



**ÉCOLE POLYTECHNIQUE
UNIVERSITÉ DE MONTRÉAL**

DÉPARTEMENT DE GÉNIE ÉLECTRIQUE

Réalisation d'un contrôleur de moteur c.c

Projet de fin d'études soumis comme
condition partielle à l'obtention
du diplôme de baccalauréat en ingénierie

Présenté par: Rick Bélair
Matricule: 54508
Directeur de projet: Romano De Santis

Date: 18 avril 1997

CARTE CONTRÔLEUR DE MOTEURS

Ce projet concerne la conception, l'analyse, la construction, la mise en opération et la documentation d'un contrôleur de position/vitesse de moteurs à courant continu. L'objectif est de mettre en place un asservissement de position/vitesse qui puisse être utilisé au sein des robots autonomes développés par le groupe SAE Robotique.

Afin de bien répondre à l'objectif, une analyse complète du problème fut élaborée. Au delà du problème de contrôle de moteur, d'autres critères de réalisation se sont ajoutés comme: la fiabilité, la durabilité du système, la facilité d'implantation et le maintien du PC comme processeur central. La carte contrôleur réalisé se branche donc sur le BUS PC-ISA et utilise des puces de contrôles de moteurs dédiés. Comme ces puces d'asservissement possèdent leur propre librairie de fonction, la programmation est très simple et directement en langage évolué sur un environnement de PC bien connu de tous. En plus du contrôle de 4 moteurs, chaque carte possède 24 E/S programmable et 4 entrées A/N 8 bits pour y brancher des capteurs sensoriels.

L'objectif du projet est donc atteint, le robot marcheur Alexis et le robot rouleur Bazou pourront participer respectivement au Walking Machine Decathlon qui se déroule cette année à Mexico ainsi qu'au Festival des Sciences et Technologies de France.

Mots-clés: Contrôleur PID, moteur courant continu. , BUS ISA PC104.

TABLE DES MATIÈRES

	pages
CARTE CONTRÔLEUR DE MOTEURS	I
TABLE DES MATIÈRES.....	III
REMERCIEMENTS	V
LISTE DES FIGURES	VI
LISTE DES TABLEAUX.....	VII
INTRODUCTION	1
1.0 FORMULATION DU PROBLÈME	3
2.0 ÉTAPES DE DÉVELOPPEMENT DU CONTRÔLEUR	7
2.1 ANALYSE DES BESOINS ET DES CONTRAINTES	7
2.1.1 Définition du contrôleur requis.....	7
2.1.2 Limitations mécaniques	8
2.1.3 Contraintes électriques	9
2.1.4 Les coûts	9
2.1.5 Autres contraintes et besoins	9
2.2 CHOIX DE DESIGN	10
2.2.1 Étape 1: Choix du microprocesseur central	10
2.2.2 Étape 2: Le choix du contrôleur de moteur dédié.....	11+
2.2.3 Étape 3: Le standard PC104.....	11
2.2.4 Étape 4: Le circuit d'interface (FPGA)	12
2.2.5 Étape 5: Convertisseur Analogique numérique	12
3.0 FONCTIONNEMENT DE LA CARTE.....	13
3.1 DESCRIPTION DU CIRCUIT	15
3.1.1 Description du FPGA	17
3.1.2 Description et fonctionnement du PMD.....	18
3.1.3 Description du convertisseur A/N	19
4.0 PROCÉDURE D'UTILISATION DE LA CARTE.....	20
4.1 CHOIX DU IRQ (INTERUPT REQUEST).....	21
4.2 CHOIX DE L'ADRESSE DE BASE.....	22
4.3 LA PROGRAMMATION DU FPGA	23
4.4 BRANCHEMENTS.....	24
4.4.1 Connecteur encodeur	24
4.4.2 Connecteur moteurs	25
4.4.3 Connecteur I/O	27
5.0 IMPLANTATION DE LA COMMUNICATION.....	28
5.1 LA COMMUNICATION GÉNÉRAL ENTRE LE PC ET LE CIRCUIT PMD	30
5.2 COMMUNICATION ENTRE BUS PC- ISA ET LE FPGA.....	32
5.3 COMMUNICATION AVEC LES CIRCUITS DE PMD.....	36
CONCLUSION	41

RECOMMANDATIONS	43
ANNEXE A : LISTE DES COMPOSANTES	44
ANNEXE B : SCHÉMAS LOGIQUES DE LA CARTE CONTRÔLEUR.....	46
ANNEXE C : SCHÉMA DE FABRICATION DU CIRCUIT IMPRIMÉ.....	47
ANNEXE D : IMPLANTATION LOGIQUE DANS LE FPGA	48
ANNEXE E : LE CONVERTISSEUR ANALOGIQUE NUMÉRIQUE AD7824.....	49
ANNEXE F : LOGICIEL DE PROGRAMMATION PAR LE PORT PARALLÈLE	50
GLOSSAIRE.....	50
BIBLIOGRAPHIE.....	52

REMERCIEMENTS

J'aimerais exprimer ma reconnaissance à tous ceux qui ont contribué à la réalisation de ce projet: tout les employé de la compagnie Netcorp en particulier Marc Nagel, pour leur soutien technique, matériel et financier. Je tiens aussi à remercier la compagnie C.M.R pour la commandite des circuits imprimés et la compagnie Tecknor pour le don du PC industriel. Merci aussi, à toute l'équipe de SAE ROBOTIQUE, particulièrement à Richard Prescott pour le déverminage de la communication entre la carte contrôleur et le PC. Je voudrais également remercier Philippe Brais pour avoir crée une interface utilisateur exceptionnel pour la carte contrôleur ainsi que Patrick Grondin et Monsieur Romano DeSantis .

Enfin, je voudrais remercier de tout mon coeur Paule Bergeron , pour sa patience et son soutien tout au long de la réalisation du projet.

LISTE DES FIGURES

	pages
FIGURE 1 SCHÉMA BLOC GÉNÉRAL DE LA CARTE CONTRÔLEUR.....	14
FIGURE 2 IMAGE DE LA CARTE CONTRÔLEUR.....	22
FIGURE 3 CONNECTEUR DB9 ENCODEUR ET CAPTEUR DE FIN DE COURSE.....	25
FIGURE 4 AMPLIFICATEUR EN H.....	26
FIGURE 5 SHÉMA BLOC DÉTAILLÉ DE LA CARTE CONTRÔLEUR.....	29
FIGURE 6 R16.TIF AVEC READ=INPW(0x300).....	34
FIGURE 7, W_16.TIF AVEC OUTPW(0x300, 0xABCD).....	34
FIGURE 8 DIAGRAMME TEMPOREL POUR ÉCRITURE D'UNE COMMANDE.....	36
FIGURE 9 DIAGRAMME TEMPOREL D'ÉCRITURE DE DONNÉE SUR LE PMD.....	37
FIGURE 10 DIAGRAMME TEMPOREL DE LECTURE DE DONNÉE SUR LE PMD.....	37
FIGURE 11 PSEUDOCODE SHÉMATIQUE DE LA MACHINE A ÉTAT FPGA-PMD.....	40

LISTE DES TABLEAUX

	pages
TABLEAU 1 SÉLECTION DE L'ADRESSE DE BASE	23
TABLEAU 2 REGISTRE D'ÉTAT	30
TABLEAU 3 TYPE DE COMMANDE	31

INTRODUCTION

Le groupe SAE ROBOTIQUE de l'école Polytechnique de Montréal est un groupe étudiant au baccalauréat qui conçoit et fabrique des robots autonomes en vue de participer à deux compétitions. La première compétition est le « Walking Machine Decathlon » qui est réservée aux robots marcheurs autonomes. Cette année, elle se déroule au Mexique. La seconde compétition pour les robots rouleurs autonomes est le « Festival des Sciences et Technologies » qui se tient à Toulouse cette année.

Depuis sa construction en 1993, ALEXIS le robot marcheur de SAE ROBOTIQUE, n'a pu faire que huit pas en mai 1995. La défaillance a été causé par un contrôle tout ou rien très saccadé et une faiblesse mécanique dans un genou. En avril 96 dans le cadre de la compétition du WALKING MACHINE DECATHLON à l'école de Technologie supérieur, le contrôleur utilisé, n'a pas suffi à la tâche. En effet, les circuits faits de «wired wrap » et les connecteurs RJ45 (connecteur de téléphone) n'ont pu supporter les vibrations qu'implique la marche du robot. Par conséquent, l'équipe a été contrainte d'abandonner quelques heures avant le début de la compétition.

Comme les équipes de travail restent rarement plus de trois ans au sein du groupe SAE, il est presque impossible de refaire le système de contrôle au complet tout en rendant le robot autonome. Il faut donc laisser un système de contrôle fiable, facilement adaptable et en quantité suffisante pour les nouveaux robots.

Pour pouvoir répondre aux exigences de fiabilité et de disponibilité de la carte, des circuits imprimés de la carte contrôleur devront être réalisés. La carte contrôleur devra se brancher sur un PC pour préserver le système de reconnaissance visuel du robot rouleur déjà en place. Le circuit devra être composé principalement d'un contrôleur de moteurs dédiés afin d'ajouter un point de fiabilité à la carte. Ce circuit intégré diminuera grandement les risques inhérents d'erreurs de programmation en assembleur des microcontrôleurs dédiés. De plus le contrôleur dédié permettra de diminuer le temps de développement de la carte contrôleur.

Enfin ce projet concerne la conception, l'analyse, la construction, la mise en opération et la documentation d'un contrôleur de position/vitesse d'un moteur à courant continu. L'objectif est de mettre en place un asservissement de position/vitesse qui puisse être utilisé au sein des robots autonomes développés par le groupe SAE ROBOTIQUE.

Ce document présente donc les étapes de développement du contrôleur de moteur, c'est-à-dire la formulation du problème, l'analyse des besoins, les choix de design et le design détaillé de la carte réalisée. Etant donné que la carte utilise un contrôleur de moteur dédié, ce document portera surtout sur la construction de la carte et l'interface de communication PC-contrôleur dédié. Puisque le fonctionnement du circuit de contrôle est bien expliqué dans le manuel du fabricant.

1.0 FORMULATION DU PROBLÈME

La formulation du problème constitue une étape importante lors de la réalisation d'un projet de recherche et de développement. En effet, pour bien identifier les besoins, il faut bien connaître le problème. Afin de bien cerner celui-ci, nous survolerons les travaux réalisés par les équipes précédentes et les développements futurs. Ce survol présentera donc les problèmes reliés à la réalisation de robots autonomes par rapport à l'électronique de contrôle de déplacements de ces robots produits par le groupe SAE ROBOTIQUE.

Comme mentionné précédemment, le but du groupe étudiant SAE ROBOTIQUE est de concevoir des robots autonomes en vue de participer aux compétitions. Cette tâche représente une panoplie de défis autant mécanique, électrique qu'informatique. Nous ne traiterons pas ici des problèmes purement mécaniques ou informatiques, mais il faut tout de même tenir compte des interrelations possibles.

Le projet SAE ROBOTIQUE existe depuis 1986, mais les vestiges de ces civilisations antérieures sont disparus avec leurs créateurs. La dernière trace tangible que nous possédons est un circuit imprimé et quelques photos de l'équipe 1991-93. À l'époque, le robot marcheur à quatre pattes était muni d'une architecture de contrôleur entièrement fait maison. Deux cartes MC68000 jouaient le rôle de cerveau et cervelet tandis que cinq cartes munies de processeurs XC68HC711G5 contrôlaient deux moteurs chacun. Les

XC68HC711G5 de Motorola étaient, à l'époque, des prototypes ce qui rendais la tâche d'en trouver d'autres presque impossible.

Finalemnt, l'équipe de 1993-1996 a dû construire un autre contrôleur. Le concept de cerveau et de cervelet de la génération précédente a été retenu. Le cerveau a été remplacé par un PC pour rendre le développement plus aisé et le cervelet a été remplacé par un MC68332. Cette carte contrôleur fiché sur le bus du PC, servait à contrôler les 9 moteurs à courant continu du robot marcheur Alexis. À l'extérieur du PC se trouvait la boîte de jonction sur laquelle se branchait les capteurs de position et les amplificateurs de puissance. La boîte de jonction et la carte contrôleur étaient faits de « wired wrap » et de connecteurs qui supportaient mal les vibrations; ce qui à causé bien des problèmes. Présentement le système ne peut plus être utilisé puisqu'il n'est pas assez fiable. Avant de reproduire cette architecture sur circuit imprimé pour le rendre plus fiable, il est préférable de bien analyser les besoins présent.

Depuis 1995, des robots rouleurs ont aussi été construits pour participer à la compétition du Festival des sciences et technologie de la France. Les robots devaient êtres en mesure de suivre une ligne tracé sur un damier. Dans notre cas le robot détecte la ligne à l'aide d'une caméra branchée sur une carte d'acquisition vidéo Matrox. Le contrôleur de moteur utilisé en 1996 était un autre produit maison fabriqué sur carte prototype, mais cette fois-ci le microcontrôleur était un 68HC11 branché sur le port sériel.

Le principal défaut des deux cartes contrôleur est le manque de fiabilité. Mais ce n'est pas tout, ces deux cartes ont une architecture différente et un langage de programmation différent. Le code n'est donc pas compatible et la possibilité d'adaptation est restreinte. En effet, cette année le groupe voulait construire un nouveau robot marcheur à six pattes avec 18 moteurs. Mais les cartes contrôleur en place n'auraient pas pu être utilisées. En fait, les problèmes se résument au manque de standardisation, de « portabilité » et de fiabilité. Peu importe les nouveaux robots fabriqués, il faudra être en mesure d'adapter facilement et rapidement les cartes contrôleur. Le nombre de cartes contrôleur disponible devrait être suffisant pour répondre à plus que la demande de chacun des robots afin d'avoir quelques pièces de rechanges.

D'une équipe à l'autre, l'électronique de contrôle n'a pu être préservé, soit à cause de la disponibilité des pièces, de la documentation, de l'expertise mal transmis ou bien par le manque de « portabilité » du design. Le temps investie lors de la construction des cartes contrôleurs a empêchée année après année d'arrivé au but visé soit l'autonomie des robots. Jusqu'à présent, aucun capteur sensoriel n'a été utilisé sur nos robots marcheurs car les cartes contrôleur prenaient toutes les ressources disponibles .

Comme le projet SAE ROBOTIQUE regroupe des étudiants au baccalauréat de l'école Polytechnique de Montréal les limites de temps ne peuvent être négligés. En effet, Bien que le bac a une durée de 4ans la période active des membres du groupes est souvent limité à moins de 3 ans. Le temps de développement complet d'un robot autonome est

donc très limité et semble irréalisable en si peu d'année. Il faut donc laisser aux générations suivantes au moins le coeur de nos robots disponibles soit le système de contrôle pour être en mesure de rendre les robots autonomes.

Un autre facteur important est que le budget de SAE ROBOTIQUE est en majorité formé de commandites. En retour les entreprises attendent, une bonne visibilité de leur produits ou de leur emblèmes. Que ce soit lors des compétitions, à la télévision ou à l'intérieur des murs de Polytechnique. Il faut donc leur faire honneur pour se mériter leur bonnes faveurs les années suivantes, sinon la recherche de commandites sera de plus en plus fastidieuse. Le budget est donc très limité, et lors de la recherche de solution, il faut faire des demandes de commandites de produits ce qui allonge considérablement le temps de développement des robots.

2.0 ÉTAPES DE DÉVELOPPEMENT DU CONTRÔLEUR

2.1 Analyse des besoins et des contraintes

Pour répondre adéquatement aux besoins de SAEROBOTIQUE, il faut préalablement bien définir les caractéristiques du contrôleur requis et toutes les contraintes tant du côté mécanique qu'électrique. Le coût et le temps de fabrication sont aussi des facteurs à ne pas négliger.

2.1.1 Définition du contrôleur requis

Un contrôleur, dans notre cas, est un système qui permet d'asservir un ou plusieurs moteurs en position/vitesse. Pour permettre de compenser l'erreur de position et de vitesse, un PID est habituellement utilisé pour ce type d'application. Dans une optique un peu plus vaste, le contrôle des moteurs ne suffit pas. Il faut, bien entendu, asservir correctement chaque moteur pour générer la marche du robot, mais il faut également, connaître l'environnement dans lequel le robot évolue pour être en mesure d'interagir avec celui-ci. La carte contrôleur dans cette optique devra donc comporter des interfaces pour les capteurs sensoriels.

La carte contrôleur doit recevoir les signaux suivants: l'encodeur optique incrémental, les capteurs de fin de course, la commande (numérique) et les constantes d'asservissement (numériques). Le système d'asservissement donne comme sortie un signal de sens de

rotation et un signal pulsé de largeur variable (PWM pulse width modulation) qui est une fonction PID de l'erreur de vitesse/position recherchée. De plus, étant donné la rapidité du robot, la boucle d'asservissement doit être de l'ordre du ms pour avoir un contrôle en position/vitesse précis.

2.1.2 Limitations mécaniques

Puisque la marche d'Alexis génère passablement de vibrations et de chocs mécaniques, la logique de contrôle et les circuits d'interfaces devront être en mesure de supporter cet environnement. L'espace alloué pour le matériel de contrôle est limité, La superficie maximale que ce matériel peut occuper doit être inférieur à 20cm X 20 cm X 30cm. De plus, sur le robot marcheur, il faut minimiser le poids, par conséquent la masse de la logique de contrôle doit donc être la plus petite possible.

Comme le système d'asservissement doit être implanté sur Alexis (9 moteurs), Bazou (2 moteurs) et le nouveau marcheur (18 moteurs), il doit être modulaire, en plusieurs exemplaires et ce afin de permettre le travail simultané sur les robots tout en laissant l'équipement fixe dans chaque robot.

2.1.3 Contraintes électriques

La consommation énergétique doit être limitée puisque les robots doivent être autonomes et donc alimentés par des batteries. Afin de distribuer l'énergie des batteries des convertisseurs DC-DC sont utilisés. Par conséquent, l'alimentation logique est limitée à 5 volts.

2.1.4 Les coûts

Le comité SAE ROBOTIQUE est un groupe étudiant qui doit financer leurs projets par des commandites. Il faut donc tenter d'avoir le plus de commandites de matériels possibles chez les fabricants de produits car les don en argent se font rare. Le développement du contrôleur existant a coûté 10 000\$ réparti sur une période de trois ans sans avoir les performances désirées. Le budget alloué cette année pour l'électronique de contrôle est fixé à 3000\$.

2.1.5 Autres contraintes et besoins

Pour répondre le mieux possible aux besoins de la compétition, le design électrique doit être fait entièrement par l'équipe. Le temps de développement est limité à moins d'un an pour que les robots puissent participer aux compétitions. Étant le dernier de l'équipe électrique de SAE Robotique et n'ayant pas trouvé de relève, le design de la carte devait être simplifié le plus possible afin de réduire le temps de développement et faciliter la

compréhension du système par les générations futures. Afin d'accélérer et faciliter le développement du contrôleur, la programmation bas niveau (assembleur) a été évitée.

De plus, dans le but de répondre à l'objectif global de la carte contrôleur, il serait utile de prévoir quelques entrées/sorties programmables et un convertisseur A/N pour y brancher les différents capteurs .

2.2 Choix de design

Les besoins et les contraintes étant analysés, une recherche est maintenant nécessaire pour identifier les composantes s'intégrant à la carte contrôleur. Les étapes de sélection sont les suivantes:

2.2.1 Étape 1: Choix du microprocesseur central

Afin de répondre aux besoins énumérés précédemment, plusieurs choix de design ont été faits. Premièrement, l'ordinateur personnel a été préservé comme cerveau pour les robots étant donné sa facilité d'utilisation et son environnement bien connu. Pour avoir une bonne vitesse de transfert de données, la communication entre le PC et la carte contrôleur est effectuée directement sur le BUS ISA du PC.

2.2.2 Étape 2: Le choix du contrôleur de moteur dédié

Une fois l'environnement de travail défini, le choix d'un contrôleur de moteur est relativement simple. En effet, pour augmenter la fiabilité, pour diminuer le temps de développement et pour réduire l'espace occupé par la carte, un contrôleur de moteur dédié pour 4 moteurs a été choisi. Les autres modèles contrôleur n'étaient pas aussi compact et n'offraient pas autant de fonctionnalité. La disponibilité des pièces est un facteur important, c'est pourquoi, le choix s'est arrêté sur les composants de PMD (Performance Motion Device). Comme mentionner précédemment, le coût joue un rôle déterminant dans la prise de décision. Cependant dans ce cas ce critère n'a pas d'importance puisque les pièces nous ont été commanditées.

2.2.3 Étape 3: Le standard PC104

Pour pouvoir répondre aux exigences mécaniques de vibrations et aux contraintes d'espace, le standard industriel PC104 a été choisi. Ce standard est l'équivalent du BUS ISA mais en plus compact et permet d'empiler les cartes au lieu de les enfichers sur le bus de fond de panier (back plane). Les cartes du PC 104 ont une grandeur standard de 3.5 X 3.75 X 0.9 pouces ce qui répond bien aux contraintes spatiales. L'utilisation du standard PC104 avec un PC ordinaire nécessite la fabrication d'un adaptateur enfichable. La grandeur de la carte contrôleur PC 104 ne peut maintenant plus être diminuée. Cependant, pour mieux répondre aux exigences d'espace, il faut tenter de réduire le volume occupé par le PC. La compagnie *TECKNOR industrial computers inc* a donc été approchée

puisque cette compagnie fabrique des ordinateurs PC compatibles complet et compact avec le standard PC104.

2.2.4 Étape 4: Le circuit d'interface (FPGA)

Il ne reste plus qu'à choisir le circuit d'interface entre le PC et les puces de PMD. Le circuit d'interface est nécessaire pour assurer la compatibilité temporelle entre le PC et la carte contrôleur. Une façon souple et simple de réaliser cette interface est d'utiliser un FPGA (Field Programmable Gate Array). Cette puce peut être programmée par un logiciel comme View Logic pour implanter les fonctions logiques désirées. La génération de FPGA 3042 de Xilinx a été choisie pour son format « surfacemount MQFP100' » très compact, pour ses 97 E/S disponibles et parce qu'il est possible de le programmer à volonté. De plus l'environnement de travail et ces FPGA sont offerts par la compagnie NETCORP.

2.2.5 Étape 5: Convertisseur Analogique numérique

L'espace occupé par le FPGA étant réduit, il est donc possible d'ajouter un convertisseur A/N sur la carte de contrôle afin d'être en mesure d'utiliser des capteurs analogiques (jauge de contraintes, échantillonneur de courant pour moteurs...). Le choix s'est arrêté sur un convertisseur A/N populaire le AD7824, celui-ci possède 4 entrées analogiques et un temps de conversion de 2.5 μ sec.

3.0 FONCTIONNEMENT DE LA CARTE

Cette partie du document présentera en profondeur le fonctionnement de la carte contrôleur, c'est-à-dire une description physique du circuit, la procédure d'utilisation de la carte et l'interface de communication. Il est à noter que la carte a été conçue pour fonctionner avec une communication de 16 bits. Pour l'instant, seulement la communication 8 bits est implantée pour des raisons électriques que j'expliquerai dans la section de l'interface de communication sur le bus du PC. La carte peut également supporter un convertisseur A/N, cependant la logique de communication n'est pas implantée puisque le robot n'est pas encore muni de capteurs analogiques.

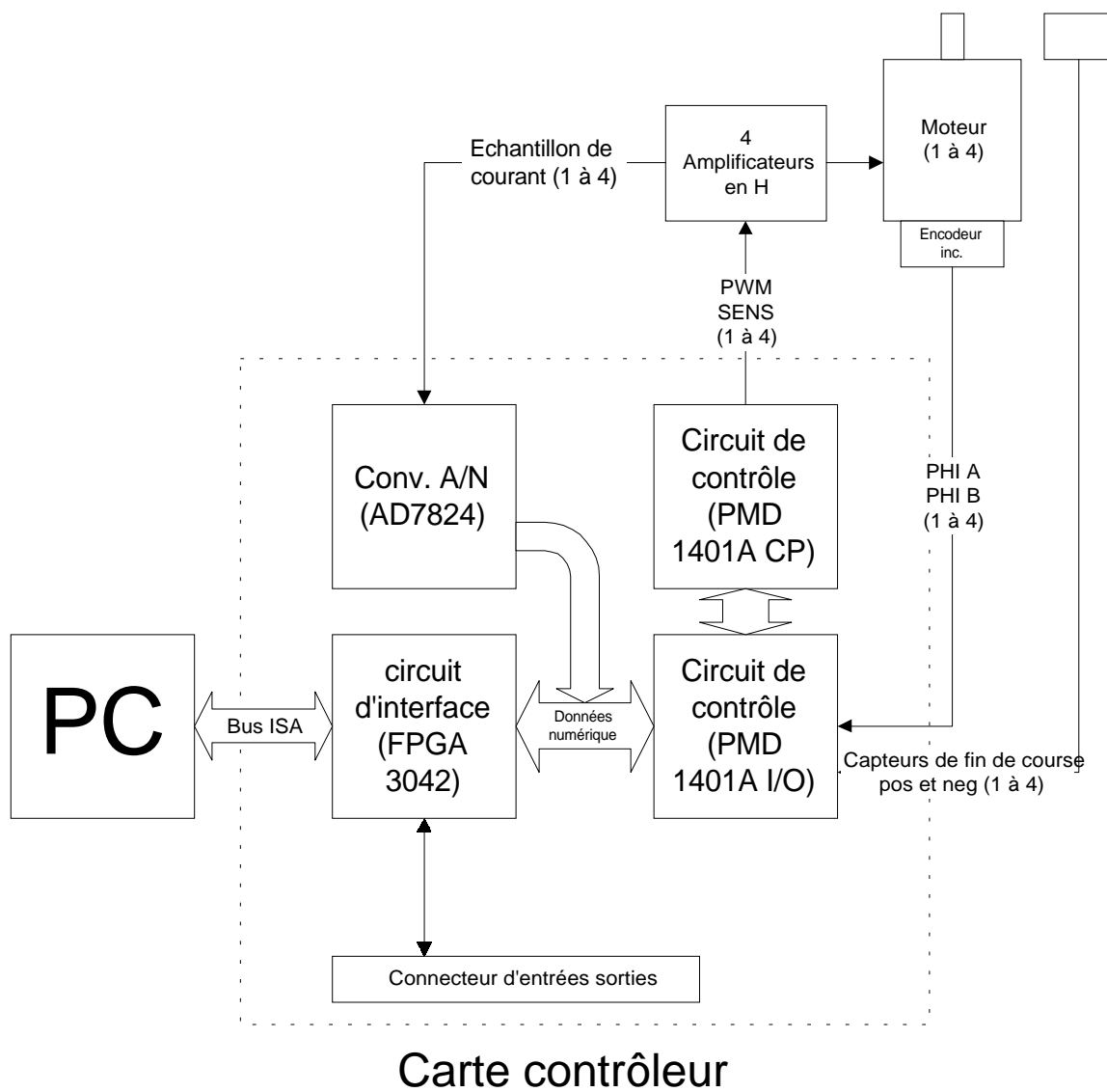


Figure 1 Schéma bloc général de la carte contrôleur

3.1 Description du circuit

Le schéma bloc (figure 1) nous donne un aperçu général des branchements logiques entre les composantes de la carte. Pour plus de détails concernant ces branchements logiques, l'emplacement ainsi que la liste des composantes avec leur numéro de pièce se référer à l'annexe A et B.

Si nous nous référons au schéma bloc général de la carte (figure 1), nous remarquons que la fonction principale de la carte est effectuée par 3 circuits intégrés, les 2 puces PMD et le FPGA. Ce dernier sert principalement d'intermédiaire de communication entre le PC et les puces de contrôle PMD pour des raisons de synchronisation. Ces raisons seront expliquées dans la section des interfaces de communication. Les puces PMD se chargent de l'asservissement des 4 moteurs CC selon une commande numérique de position et de vitesse envoyée par le PC.

Plus précisément, le circuit de contrôle PMD 1401A CP est le processeur central qui fait les calculs de compensation d'erreurs et génère les signaux de contrôle des moteurs: soit le sens de rotation du moteur et un signal pulsé de largeur variable (PWM). Ce dernier contrôle la vitesse du moteur.

Les signaux PWM et SENS sont par la suite injectés dans un circuit d'amplification en H pour ainsi transmettre un signal 24volts pulsé sur les bornes du moteur courant continu. Selon la longueur des impulsions PWM, le moteur tournera plus ou moins vite. Sur le

circuit d'amplification, il est possible, d'ajouter une très petite résistance (0.01Ω), en série avec le moteur, sur laquelle une tension est mesurée. Celle-ci est directement proportionnelle au courant tiré par le moteur selon la loi d'Ohm ($V=RI$). Il est donc possible de convertir la tension lue à l'aide du convertisseur analogique numérique. Cette tension nous permet donc d'avoir accès, à partir du PC, à une valeur numérique sur 8 bits du courant tiré par chacun des moteurs. Le courant est proportionnel à la force appliquée par chaque moteur, par conséquent le robot peut savoir si la patte touche par terre ou si le robot est en déséquilibre.

Le circuit de contrôle PMD 1401A I/O sert d'interface entrée/sortie pour communiquer avec le processeur maître (le PC dans notre cas). Cette puce permet aussi de capter les signaux provenant des encodeurs optique incrémentaux et des capteurs de fin de course. PHI A et PHI B représentent les signaux en quadrature de phase d'un encodeur optique incrémental. C'est-à-dire deux ondes carrées déphasées de 90° qui permettent de déduire le sens de rotation du moteur et d'avoir une mesure du déplacement de l'actuateur en comptant les impulsions. Sur chaque moteur, on retrouve 2 capteurs de fin de course soit le début (positive limit switch) et la fin (négative limit switch) du déplacement de l'actuateur.

Finalement, on retrouve sur le schéma bloc le circuit d'interface (FPGA 3042) qui permet de gérer les communications entre le PC, le circuit de contrôle PMD, le convertisseur A/N et les E/S programmable.

3.1.1 Description du FPGA

Le FPGA est une composante logique pouvant être programmée selon les spécifications logiques de l'utilisateur. Comme son nom l'indique le « Field Programmable Gate Array » est un champ de logique programmable. Le AT&T 3042 est formé d'un champ de 12 X 12 blocks contenant chacun une logique d'entrée combinable de 5 variables et 2 bascules. Ces blocks peuvent être interconnectés par un réseau de branchements réparti entre ces blocks. Dans le réseau d'interconnexion, il existe des portes 3 états (0, 1 et haute impédance 'Z') qui remplacent avantageusement les multiplexeurs. Un des principaux avantages de ce circuit est qu'il peut être programmé à volonté et ce directement par le port parallèle du PC ou bien par un simple PROM (Programmable Rom) sériel. La conception, la simulation et l'implantation logique, à l'intérieur du FPGA, se font rapidement à l'aide des outils de développement XACT.

Le FPGA AT&T3042 possède 97 entrées/sorties programmables, dont une partie est utilisée pour faire la connexion sur le BUS du PC, une autre partie est utilisée pour établir la communication avec le PMD et le convertisseur A/N. Le reste des E/S libres sont dirigées vers un connecteur qui nous permettra de brancher les capteurs.

3.1.2 Description et fonctionnement du PMD

Comme mentionné précédemment, les puces MC1401A de PMD (Performance Motion Device) sont un agencement de 2 circuits intégrés qui permettent de contrôler 4 moteurs CC. Ces puces utilisent directement les signaux des encodeurs incrémentaux et des deux capteurs de fin de course comme signaux de retour. Ces puces génèrent un signal PWM fonction de l'erreur PID en sortie. Le MC1401A offre plusieurs types de contrôles de trajectoire. Par exemple le type trapézoïdale génère un profil comportant les étapes de déplacement d'un actuateur. C'est-à-dire la phase d'accélération, la phase de vitesse constante et la phase de décélération de la position initiale jusqu'à la position désirée de l'actuateur. Chacun des axes (moteurs) peut être programmé indépendamment les un des autres ou en synchronisme. Le contrôleur MC1401A dispose d'une librairie de 96 commandes qui permet de configurer et d'échanger des informations concernant l'asservissement de chaque moteurs entre le contrôleur maître et le circuit de contrôle (voir le cahier de PMD en annexe).

3.1.3 Description du convertisseur A/N

Le convertisseur A/N est un circuit qui permet de transformer une tension analogique en une valeur numérique. Le AD7824 possède 4 entrées analogiques et converti à un taux de 2.5us par entrée analogique. La tension située entre 0 et 5 Volts est traduite sur mot de 8 bits. Et l'erreur de conversion est de l'ordre de la moitié du bit le moins significatif. Il est important de brancher une impédance inférieure à 100 Ω sur les entrées du convertisseur afin de respecter les limites d'opération du convertisseur (voir annexe E).

4.0 PROCÉDURE D'UTILISATION DE LA CARTE

Avant de mettre en marche la carte contrôleur, il faut faire quelques étapes préparatoires. La première étape consiste à brancher la carte sur le bus ISA du PC. Il est également nécessaire de choisir la configuration de base désirée, c'est-à-dire l'adresse de base de la carte, le mode de programmation et le numéro de l'interruption (IRQ 3 à 7). La communication entre un processeur maître et l'esclave peut se faire de deux façons, par interruption ou par la méthode d'accès successif «polling». L'interruption sert à interrompre le processeur maître pendant l'exécution de sa tâche principale. Cette façon de faire permet d'éviter au processeur central de faire des lectures inutilement. Dans notre cas, les interruptions pourront servir à avertir le PC d'une fin de déplacement ou d'une erreur. Une fois la configuration choisie, il faut, s'assurer du branchement adéquat des entrées et des sorties du système. Le mode de programmation déterminé, il faudra soit programmer le FPGA à l'aide du port parallèle ou directement utiliser la carte qui sera déjà programmée par un PROM sériel. La dernière étape consiste à programmer en langage C ou CPP la marche désirée. Pour ce faire, une librairie en CPP, des commandes des puces PMD, a été développée par l'équipe informatique du groupe SAE ROBOTIQUE. La description et le fonctionnement des puces PMD est expliqué dans le manuel du fabricant en annexe.

4.1 Choix du IRQ (interrupt request)

Plusieurs configurations sont possibles pour permettre l'interruption. Le connecteur J1 (voir figure 2) permet de choisir le numéro d'interruption pour chaque carte branchée. Il est à noter qu'une carte ne doit avoir qu'un seul numéro d'interruption. Pour chaque ligne d'interruption choisie, selon la spécification, il faut ajouter une résistance de 10 k Ω branchée à la masse. Il est bien important de vérifier que le système ne possède pas déjà cette résistance et qu'il n'utilise pas la ligne d'interruption à d'autres fins. Cette résistance de bas niveau « Pull down », nécessite l'ajout d'un cavalier entre les pins 11 et 12 du connecteur J1. Afin de sélectionner le numéro d'interruption, il suffit d'ajouter un cavalier aux numéros d'interruption correspondant indiqués sur le circuit (près du connecteur J1).

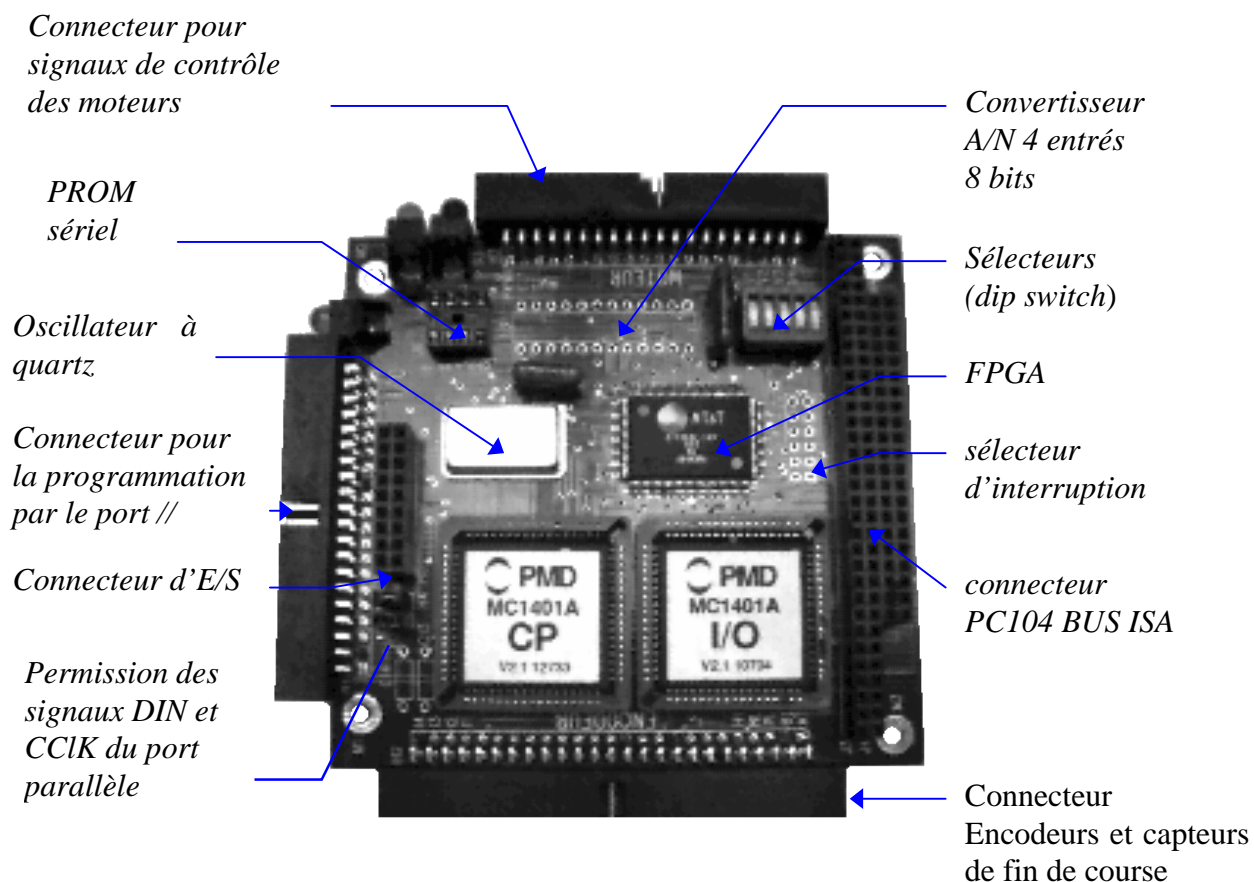


Figure 2 Image de la carte contrôleur

4.2 Choix de l'adresse de base

Chaque carte doit posséder sa propre adresse de base fixe située dans le premier mégabit (MB) d'adresse entrée/sortie du PC. Cette adresse est choisie à l'aide du sélecteur d'adresse de 4 bits (dip switch 1 à 4, voir figure 2). Les adresses situées dans le premier MB de mémoire du PC, sont décrites sur 10 bits. Ces derniers doivent être décodés par les FPGA pour savoir si la communication leur est destinée. Ainsi les bits 0 à 2 sont utilisés comme adresse interne du FPGA, les bits 3 à 6 représentent l'adresse de base sélectionnée (dip switch) et les bits 7 à 9 sont fixés et programmés dans le FPGA (hard code). Avec 4 «dip switch », 16 cartes peuvent être adressées, c'est pourquoi les bits d'adresses 7 à 9

sont fixés. On peut représenter les dix bits d'adresse par le tableau ci-dessous. Les régions ombragées représentent les bits associés aux trois chiffres hexadécimaux..

Adresse	de base du FPGA			de base choisie avec Dip switch				interne		
				dip4	dip3	dip2	dip1			
# du bit	9	8	7	6	5	4	3	2	1	0
ex.:323/hex	1	1	0	off	on	off	off	0	1	1

Tableau 1 sélection de l'adresse de base

4.3 La programmation du FPGA

Le FPGA est configuré pour être programmé de façon sérielle, soit en mode maître ou esclave. Lorsque le mode maître est sélectionné, le FPGA se programme lui-même à l'aide d'un PROM sériel. Pour ce faire, il faut que le 5^{ème} sélecteur soit à la position « on » et que les cavaliers H7 et H4 soit enlevés. Et ce pour ne pas causer de conflit avec les autres cartes par les lignes CCLK et DIN. Le PROM quant à lui doit être programmé préalablement à l'aide d'un programmeur de mémoire. Il est important de bien choisir l'option de programmation du PROM en fixant le OE (Ouput enable) actif bas. Étant donné la minceur des pattes du PROM 37LV36, il sera peut-être nécessaire de le placer sur un « socket dip 8 » pour le programmer.

Dans le mode esclave, le FPGA ne se programme pas lui-même. Ce mode implique le positionnement du 5^{ème} sélecteur (dip 5) à off. Il implique également d'ajouter les 2

cavaliers H7 et H4 afin de laisser passer les bits de données (DIN) et l'horloge de programmation (CCLK) provenant du port parallèle. Le PROM ne doit pas être utilisé dans ce mode puisque celui-ci causerait des conflits avec l'information du port parallèle.

Chaque carte est identifiée d'un numéro qui correspond au bit de 0 à 7 du port parallèle.

De cette façon, il est possible d'envoyer une configuration différente pour chacune des cartes. En effet, Il peut être utile d'avoir plusieurs configurations possibles des FPGA, par exemple une carte peut utiliser ses E/S pour contrôler un moteur pas à pas et une autre peut se charger des capteurs sensoriels. Pour l'instant, le logiciel de programmation

envoie le même fichier de configuration à tous les FPGA. Bien entendu pour programmer, il faut brancher le câble DB25 sur le port parallèle et le connecteur double à 10 broches sur le connecteur H5 avec les pins 1 correspondante (flèche sur le connecteur ribbon et pin carré sur le circuit imprimé du connecteur H5). Par la suite, il faut entrer la commande « prog378 » ou « prog278 » ou « prog3BC » selon l'adresse du port parallèle suivie du nom du fichier de configuration RBT. Les diagrammes temporels, le programme et un exemple de fichier RBT sont listés en annexe F.

4.4 Branchements

Les connecteur utilisé sont de type ruban. Le branchement a été fait pour que chaque signal soit entouré de mise à la terre.

4.4.1 Connecteur encodeur

Le connecteur utilisé est le H2 de 50 broches pour les encodeurs et les capteurs de fin de course. Il peut être divisé en 4 sections de 10 fils. Pour ce faire, on doit retirer les signaux

d'index qui ne sont pas utilisés sur nos robots. C'est 4 sections peuvent se brancher directement sur des connecteurs DB9 pour « ribbon ». Voici la disposition du connecteurs DB9.

Le signal Home est un signal d'entrée actif bas comme un capteur de fin de course mais il sert à fixer la position initiale. Dans notre cas seulement 2 capteurs de fin de course sont fixés sur le robot et l'on assigne à l'un deux la position initiale. Avec le signal Home branché, la commande Set_capt_home peut être utilisée, cette commande permet d'affecter la position à zéro au front descendant du signal home. Ce qui permet d'avoir une position initiale très précise.

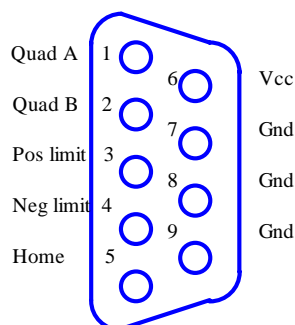


Figure 3 Connecteur DB9 encodeur et capteur de fin de course

4.4.2 Connecteur moteurs

Le connecteur H3 des moteurs possède 40 broches dont 8 sont directement reliées au FPGA. Ces 8 E/S (TFLAG(1:4) et MOT(1:4)) sont nécessaires pour maintenir la compatibilité avec les anciennes cartes amplificatrices. Afin de maintenir la compatibilité, il était nécessaire de générer à partir des signaux SENS et PWM quatre signaux

correspondant aux 4 grilles des Mosfets branchés en H. Pour ce faire, il a fallu ajouter des cavaliers pour relier PWMAG(1:4) avec TFLAG(1:4) et PWMSIGN(1:4) avec MOT(1:4). De cette façon les signaux de PWM et de SENS sont retournés dans le FPGA pour pouvoir générer les 4 lignes A, B, C et D, pour chaque moteur, ceux-ci sortent sur le connecteur I/O (voir figure 4). Étant donné que les Mosfets canal N sont plus coûteux, le signal pulsé est fait sur B ou C selon le sens de rotation.

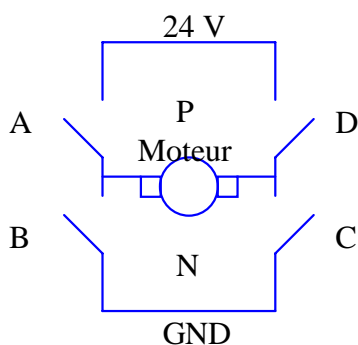


Figure 4 amplificateur en H

Si une nouvelle version d'amplificateurs en H est réalisé, le connecteur moteur pourra être utilisé tel quel avec les 8 E/S en bonus tout en libérant les 16 autres entrées sorties du connecteur I/O. Ces E/S pourront servir pour des signaux d'alerte de température ou bien pour avertir qu'un disjoncteur a brûlé. Les signaux analogiques ont été disposés à la fin du connecteur moteur pour réduire l'effet du bruit de pulsation des signaux PWM. De plus, si les moteurs n'ont pas d'échantillonneur de courant, le câble plat peut être séparé pour libérer les 4 entrées analogiques à d'autres fins.

4.4.3 Connecteur I/O

Les E/S du connecteur I/O peuvent être utilisées à toutes les sauces. Il suffit de programmer le FPGA pour pouvoir utiliser ces E/S. Jusqu'à présent, elles n'ont servi que pour la génération des signaux A B C et D des amplificateurs de moteur, mais elles serviront dans un avenir rapproché aux branchements des sonars et des capteurs infrarouges. On remarque sur le schéma logique en ANNEXE B que le connecteur contient les mêmes 4 signaux MOT(1 :4) que sur le connecteur moteur. De cette façon, si les signaux ne sont pas utilisés sur les circuits d'amplifications, ils peuvent être utilisés sur le port d'E/S.

5.0 IMPLANTATION DE LA COMMUNICATION

Nous observons sur le schéma bloc détaillé de la carte contrôleur (figure 6) les communications possibles entre le PC, le FPGA, les puces PMD et le convertisseur A/N. Pour des questions de synchronisation et de fonctionnement logique entre le PC, les puces PMD et le convertisseur A/N un intermédiaire est nécessaire ; c'est le rôle du FPGA. Avant d'aller en profondeur sur les communication PC-FPGA et FPGA-PMD, nous allons faire une brève description de l'architecture de communication. Par la suite, le BUS PC-ISA sera présenté. Et finalement les diagrammes temporel du PMD seront observé pour comprendre le fonctionnement de la machine à état de synchronisation entre le FPGA et le circuit PMD.

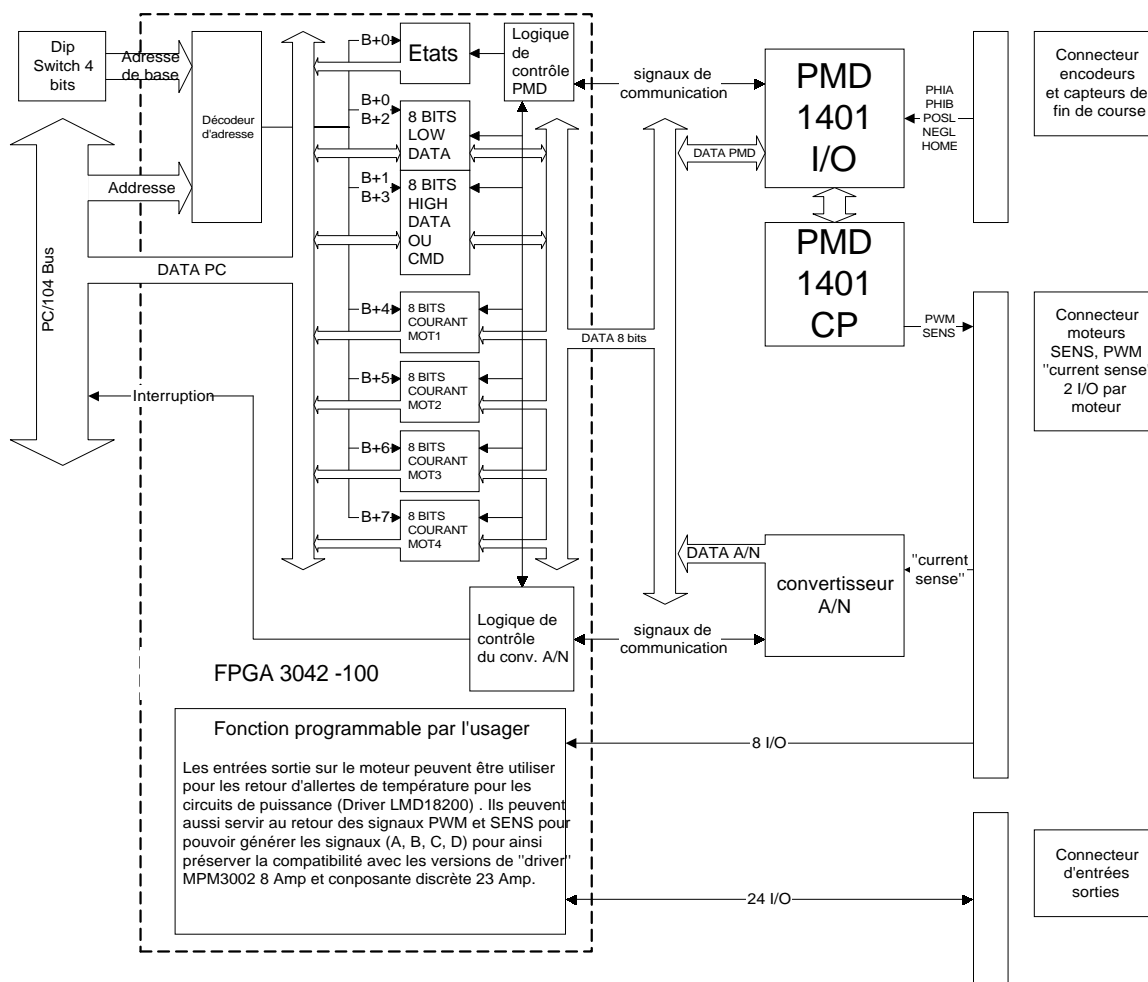


Figure 5 schéma bloc détaillé de la carte contrôleur

5.1 La communication général entre le PC et le circuit PMD

En se référant à la figure 5, on remarque le rectangle pointillé représente le FPGA et son implantation logique. On peut remarquer que le PC peut seulement écrire aux adresses de base +0, +1, +2 et +3, ces registres servent d'intermédiaire entre le PC et le circuit intégré de PMD. Pour savoir si le PC à la permission d'écrire ou lire dans ces registres d'échanges, il faut vérifier le contenu du registre d'état. On retrouve dans le tableau 2 les informations à l'intérieur du registre d'état.

Bit	7	6	5	4	3	2	1	0
information du registre d'état	-	-	-	-	-	Attend-lecture-PC	ATTEND	HostReady

Tableau 2 Registre d'état

Le signal *HostReady* indique la disponibilité du PMD pour communiquer avec le processeur maître. Le signal *ATTEND* indique la position initiale de la machine à état, c'est-à-dire la position d'attente d'écriture ou de lecture du PC. Ce dernier ne peut écrire seulement lorsque le signal *ATTEND* et le signal *HostReady* sont actif. Le signal *Attend-lecture-PC* quant à lui est un indice de la commande précédente et implique la lecture d'un mot de 32 bits. Comme il n'y a que deux registres 8 bits, le PC doit aller lire le premier 16 bits avant que la machine à état puisse aller lire la deuxième partie du mots de

32 bits. Si une commande implique la lecture d'un seul mot 16 bits, la machine à état retourne après lecture à l'état *ATTEND*. Le schéma bloc de la machine à état est expliqué dans la section suivante.

Pour envoyer une commande au PMD, il faut écrire a l'adresse de base + 1 la commande et le type de commande dans le registre de B+0. Le type de commande est définie comme suit:

Bit	7	6	5	4	3	2	1	0
information	-	-	-	-	-	32 bits	16 bits	lecture/écriture=1/0

Tableau 3 Type de commande

Le type de commande informe la machine à état du type d'accès qu'elle devra effectuer, soit une commande en lecture ou en écriture de 16bits ou 32bits ou bien seulement un écriture de donné 16 bits ou 32 bits.

Le PC devra lire et écrire les données aux adresses de base +2 et +3, c'est pour cette raison que les 2 registres d'échanges ont deux adresses qui leur sont destinées. L'adresse de base +0 et +1 pour écrire les commande et l'adresse de base +2 et +3 pour lire et écrire des données.

5.2 Communication entre BUS PC- ISA et le FPGA

Le bus PC-ISA est un descendant du bus PC-XT qui possédait une mémoire totale de 1MB et une largeur de bus de 8bits. Le BUS PC-ISA est composé de deux connecteurs : le connecteur long de 62 broches (PC-XT) et le connecteur auxiliaire long de 36 broches (ajout du BUS ISA). Il n'est donc pas surprenant de retrouver la partie haute des adresses et du bus de données sur le connecteur auxiliaire de 36 broches du BUS ISA (voir annexe B). Le standard PC-ISA doit supporter des cartes de type PC-XT pour préserver la compatibilité, mais les signaux rajoutés sur le connecteur auxiliaire modifient la fonctionnalité des accès mémoire. Par exemple les signaux de lecture et d'écriture du grand connecteur 62 broches (ancien BUS XT), ne sont activés que pour des accès mémoire dans le premier MB. Tandis que les signaux équivalents sur le connecteur auxiliaire sont actifs pour tout l'espace mémoire du bus ISA.

Les signaux utilisés sur le BUS ISA par la carte contrôleur sont les suivants : AEN*, IOR*, IOW*, SYCLK, IOCS16*, SD[0:15], SA[0:9] et les IRQ[3:7]. Selon les spécifications du BUS ISA, le signal AEN* n'est pas actif durant les accès DMA, il sert donc à différencier les types d'accès (DMA ou 1MB et moins). Les signaux IOR* et IOW* sont activés respectivement lors d'un accès en lecture ou en écriture. Le signal

IOCS16* quant à lui sert à indiquer au bus ISA que la carte peut supporter des accès mémoire de 16 bits dans la partie basse (1MB). Mais ce signal doit être activé par une porte capable de supporter un courant de 20 mA. Les spécifications du FPGA indiquent une limite de courant de sortie de 7 mA, c'est pourquoi les accès 16 bits n'ont pas été tentés directement. Toutes les autres lignes sur bus ISA sont spécifiées à 4mA. En ce qui concerne les lignes d'interruptions, il faut ajouter une résistance de $1K\Omega$ branchée à la masse pour respecter les spécifications du BUS. Il faut également faire attention de vérifier qu'il y a seulement une résistance par ligne d'interruption. Il est à noter que l'horloge de 8Mhz devrait être la référence du bus ISA. Mais comme l'horloge du BUS est souvent dérivée de l'horloge centrale du processeur et que les vitesses de processeur varient beaucoup d'une machine à une autre, il n'y a pas de spécifications officielles pour les diagrammes temporels du bus ISA.

Pour s'assurer du bon fonctionnement de la communication, des diagrammes temporels ont été échantillonnés sur le BUS ISA du PC (figure 6 et 7).

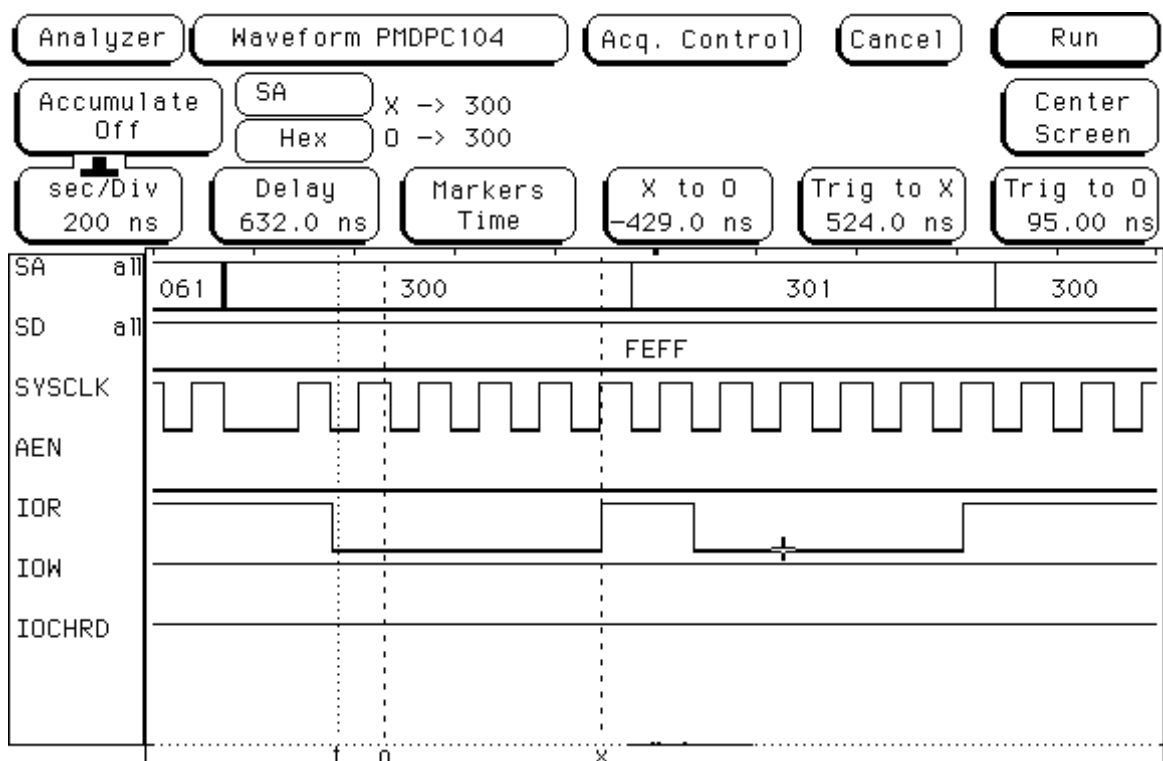


Figure 6 R16.tif avec read=inpw(0x300)

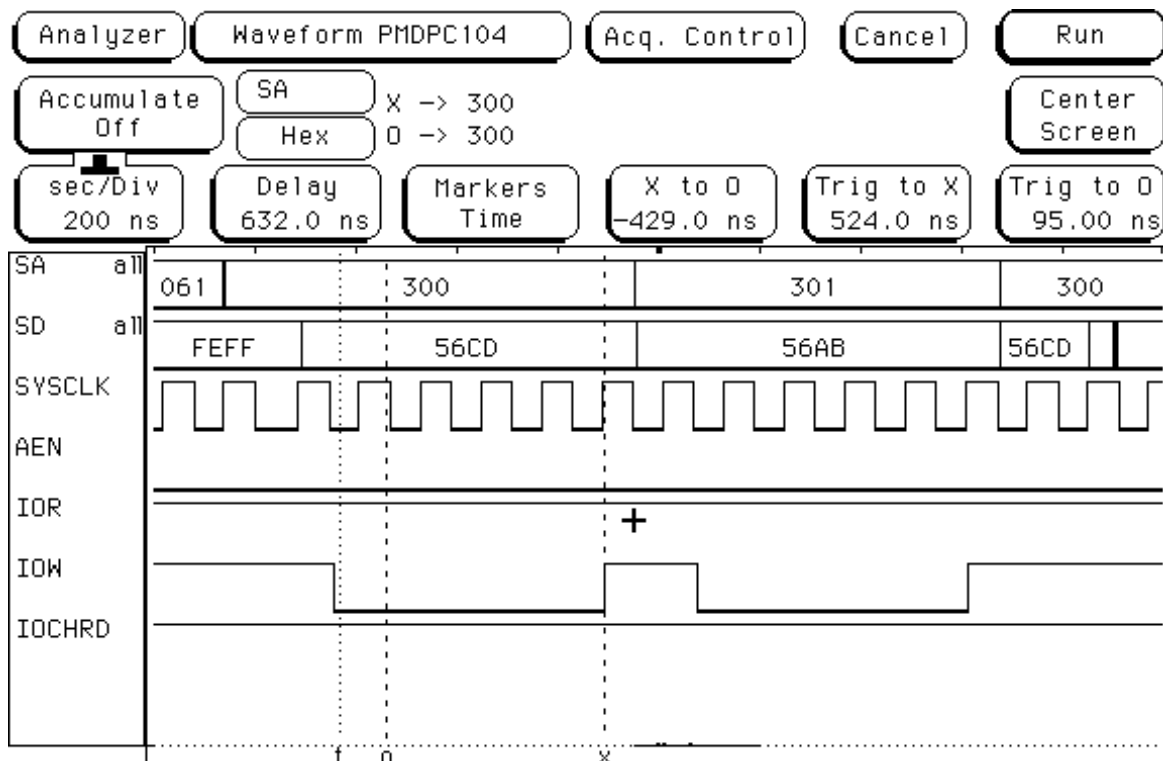


Figure 7, W_16.tif avec outpw(0x300, 0xABCD)

Les commandes en langage C `outpb` et `inpb` sont utilisées pour faire des accès 8bits dans le premier MB d'E/S du PC. Tandis que les commandes `outpw` et `inpw` sont utilisées pour faire des accès de 16 bits. Si le FPGA abaisse le signal `IOCS16` aussitôt que l'adresse de base est décodée, le mot de 16 bits est écrit d'un seul accès 16 bits. Comme le signal `IOCS16` n'est pas encore utilisé, les commandes de 16 bits génèrent un double accès 8 bits sur 2 adresses subséquentes comme les 2 figures précédentes le montrent. On peut remarquer lors d'un accès 16 bits en 2 parties (`outpw` ou `inpw` en C) que les bits les moins significatifs sont lus ou écrits par exemple, à l'adresse `0x300/hex` tandis que les bits les plus significatifs sont écrits à l'adresse `0x301/hex`. Avec une communication 16 bits d'un seul accès, la période d'écriture ou de lecture serait réduite à 4 cycles d'horloge au total tandis que la communication séparée, comme à la figure 7, prend 12 cycles d'horloge. L'implantation logique du côté du PC doit donc respecter ces spécifications temporelles que ce soit pour une horloge de 8Mhz ou de 16 Mhz.

Une façon simple pour établir la communication entre le PC et le FPGA a été d'utiliser des registres d'échanges de 8bits (mémoire de 8 bits, voir figure 5). À chacun de ces registres est associé une adresse ou deux dans certains cas. Le signal d'écriture est branché avec le décodeur d'adresse sur la permission d'écriture des registres (`input ENABLE`) afin de permettre au PC d'écrire dans le registre désiré. Lors de la lecture, l'adresse interne du FPGA est décodée pour permettre au registre en lecture d'afficher son contenu sur le BUS du PC.

5.3 Communication avec les circuits de PMD

Du coté des circuits de PMD, il faut respecter d'autres spécifications temporelles que nous pouvons observer sur les figures 9-10-11. Les signaux utilisés pour la communication entre le circuit PMD et le FPGA sont les suivants: HostSlct*, HostCmd, HostWrite*, HostRead*, HostData[7:0] et HostRdy. Le signal HostSlct* sert à avertir le PMD que nous désirons communiqué avec celui-ci. Le signal HostCmd, identifie le type d'information que le FPGA veut transférer, soit une commande, soit des données. Les signaux HostWrite et HostRead servent respectivement à faire des accès en écriture et en lecture. HostData[7:0] représente le BUS de données 8bits. Finalement le circuit PMD utilise le signal HostRdy pour avertir le processeur maître de sa disponibilité à communiquer.

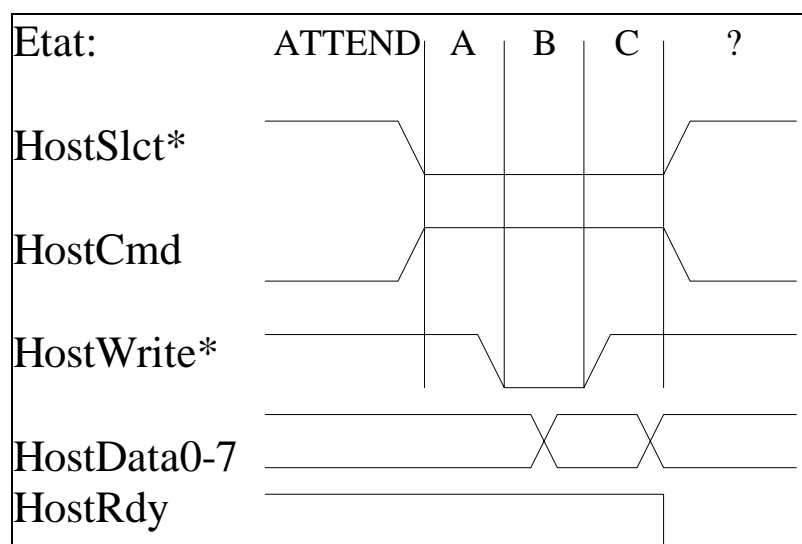


Figure 8 Diagramme temporel pour écriture d'une commande

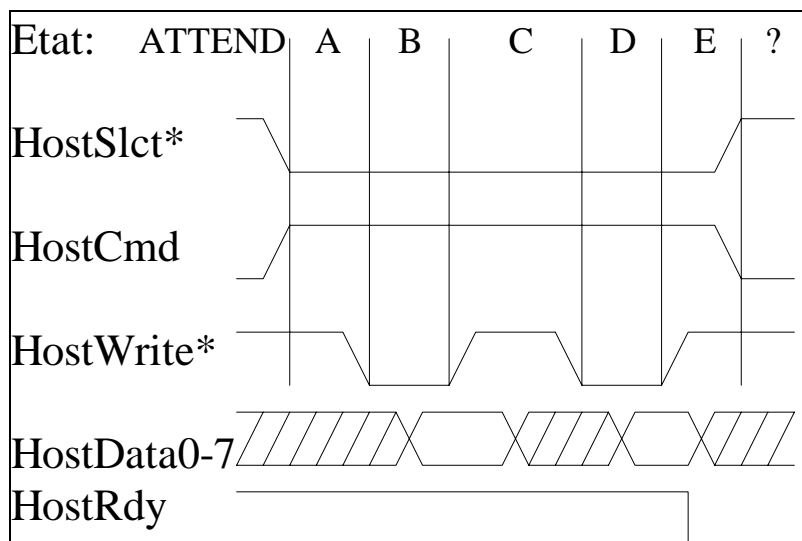


Figure 9 Diagramme temporel d'écriture de donnée sur le PMD

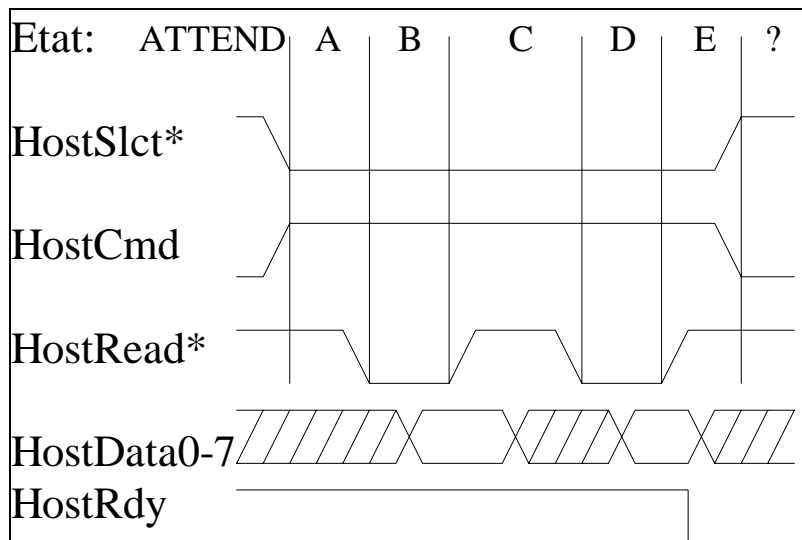


Figure 10 Diagramme temporel de lecture de donnée sur le PMD

Le PMD est un circuit esclave et ne fait qu'écouter ce que le FPGA lui dit. Le seul signal de communication que le PMD génère est le signal HostRdy pour annoncer qu'il est prêt à recevoir des informations. Afin de générer les signaux de communication selon les spécifications temporelles fixées par le circuit PMD, des états ont été assignés pour chaque étapes de communication (voir figure 8, 9 et 10). Une machine à état est utilisée pour faire le déroulement de ces étapes en contrôlant les signaux de communication tout en respectant les délais. L'étape B est l'étape la plus lente par conséquent, elle fixe le temps de transition à 50ns minimum pour toutes ces étapes.

La figure 12 représente le pseudocode schématique de la machine à état de communication implanté dans le FPGA pour communiquer avec le PMD. On retrouve les états identifiés sur les diagrammes temporels. Si l'information envoyée par le PC est une commande en écriture (commande de 8 bits, suivie de 2 ou 4 mot de 8 bit contenant des données), la machine à état parcourra les étapes A, B, C et RESET MEM. La dernière étape remet à zéro les bits COMMANDE, REG_W, R16 et R32 qui contiennent le type de communication désirée. Le bit COMMANDE est actif si l'accès désiré est une commande (le PC a écrit à l'adresse de base+0 et +1). Le bit REG_W (registred_write) permet à la machine à état de savoir si l'accès est en écriture ou en lecture. Les bits R16 et R32 identifient le type de lecture que la machine à état doit faire soit 16 bits ou 32 bits. Par exemple, si la commande envoyée par le PC est une commande de lecture d'informations sur 16bits. les bits REG_W, COMMANDE et R16 seront activés et la machine à état parcourra les états A, B, C et F. À cette étape, les bits COMMANDE et

REG_W sont effacés et la machine à état attend la remontée du signal HostRdy. Lorsque ce dernier est activé la machine à état passe par les étapes A, B, C, D, E et RESET MEM mais cette fois-ci en faisant des accès en lecture sur le PMD (comme à la figure 10). Une fois tous les bits mémoire effacés, la machine à état retourne à l'état ATTEND, pour attendre le prochain accès du PC. Prenons un autre exemple, si la commande est en lecture 32 bits, le FPGA envoie la commande, lit le premier mot de 16 bits et attend à l'étape G jusqu'à ce que le PC récupère le premier mot de 16 bits. Par la suite la machine à état lit le second mot de 16 bits et retourne à l'étape initiale ATTEND. En résumé pour effectué cette commande, la machine à état sera parcouru trois fois.

L'implantation de cette machine à état a été faite à l'aide de la méthode ONE-HOT ENCODING. Cette méthode consiste à prendre une bascule pour chaque état de la machine. Il suffit de prendre une porte OU avec autant d'entrée qu'il y a de flèche entrante dans l'état observé et la branché à l'entrée de la bascule. Par la suite, on branche les combinaisons de conditions qui sont préalablement combiné à travers des porte ET aux entrées de la porte OU (voir annexe D).

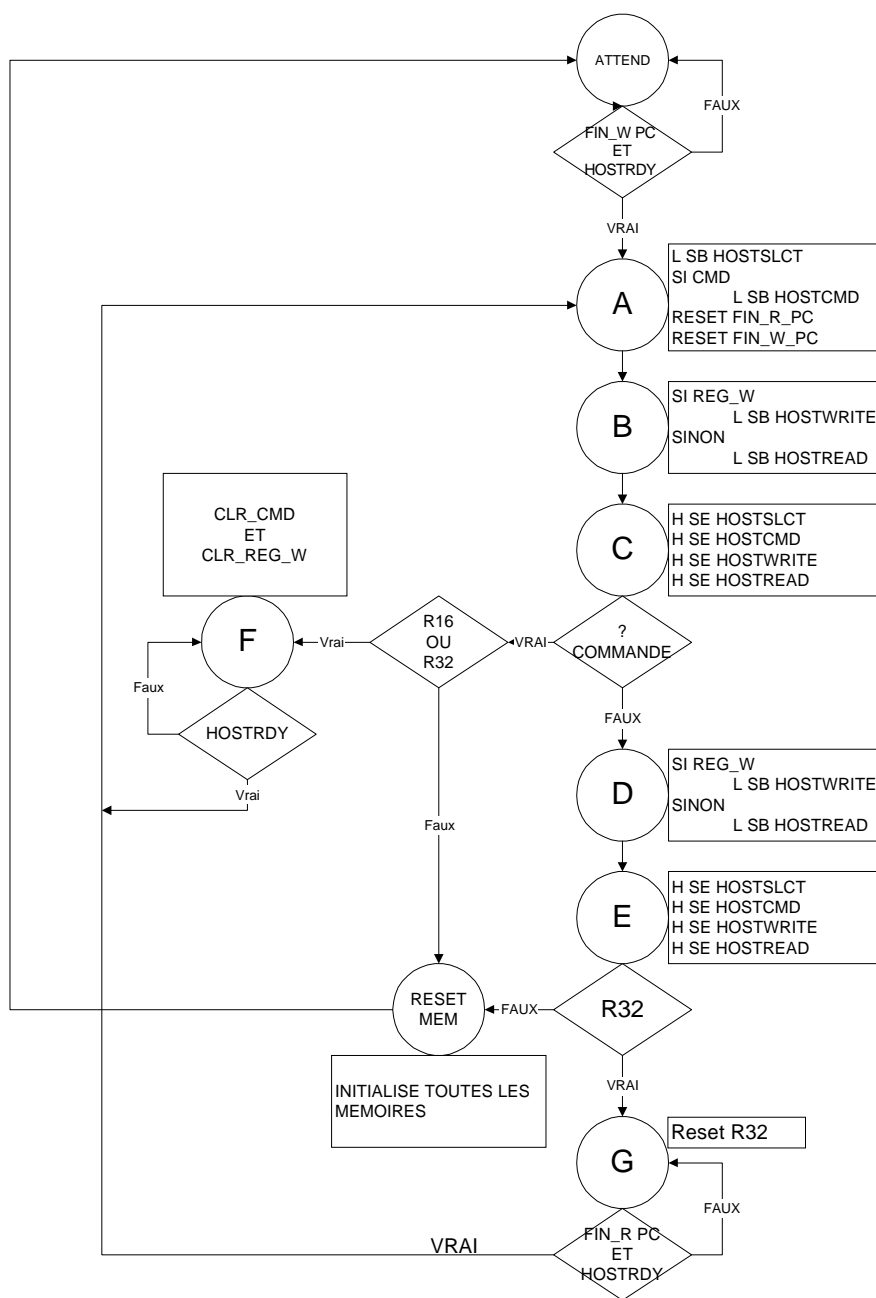


Figure 11 Pseudocode schématique de la machine à état FPGA-PMD

NOTE: SE: Le signal est désactivé à la fin de l'état (state end).
 SB: Le signal est activé au début de l'état (state beging).
 H : Niveau logique haut.
 L : Niveau logique bas.

CONCLUSION

Le but du projet était de réaliser un système d'asservissement de moteur courant continu. Afin de bien répondre au problème, une analyse des besoins et des contraintes a été réalisée. Il fallait trouver une solution qui réponde aux besoins de fiabilité, de simplicité et de compatibilité avec le PC. De plus, ce système devait être peu coûteux, réalisable en moins d'un an et il devait également être fabriqué en quantité suffisante pour répondre aux besoins des robots marcheurs et rouleur.

Afin de concevoir le système, il a fallu prendre certaines décisions. À cet égard, le PC a été retenu comme processeur central et un circuit de contrôle de moteurs dédié a été choisi. Par ailleurs, le standard PC 104, le type de FPGA et les types de connecteurs utilisés ont également été fixés.

Lors du développement les contraintes de temps et de coûts furent respectés, malgré des problèmes de communication de fichiers avec la première compagnie de PCB. Ainsi, les circuits imprimés ont finalement été fabriqués chez CMR et ils ont coûté 2400\$. De plus, la majorité des pièces qui allaient sur le circuit ont été commandités (PMD, FPGA, connecteurs, résistances, condensateurs...).

Une fois les circuits imprimés fabriqués et le deverminage physique de la carte terminée, il a fallu instaurer la méthode de programmation du FPGA par le port parallèle. C'est lorsque cette lourde étape fut terminée que les tests de communications entre le PC et le

FPGA purent commencer. Afin de s'assurer que la communication s'effectuait comme prévue les signaux furent analysés à l'aide d'un analyseur d'états. Ce dernier a permis de comprendre tous les délais de communication sur le bus ISA du PC.

La dernière étape de conception de la carte fut d'instaurer la communication entre le FPGA et les circuits PMD (puces dédiées pour les moteurs CC). Par la suite, le logiciel d'utilisation de la carte a du être adapté au traité de communication déjà en place. Cette adaptation s'est faite de façon à établir une communication la plus rapide possible entre le PC et les circuits de contrôle de moteurs (PMD) de la carte. Pour ce faire un logiciel en C++ fut créé.

Finalement, la carte répond très bien aux exigences de départ. Il ne reste plus qu'à implanter la communication avec les convertisseurs analogiques numériques et ajouter les entrées/sorties programmables pour en faire une carte d'acquisition sensorielle complète.

Les robots du groupe SAE ROBOTIQUE sont maintenant munis de tout l'équipement de contrôle nécessaire pour participer aux compétitions de cette année, au Mexique et en France.

RECOMMANDATIONS

1. Attacher un bracelet de mise à terre entre vous et le robot pour travailler sur celui-ci.
Les circuits CMOS sont très sensibles aux chocs électrostatiques.
2. Utiliser le mode de programmation « master » avec le PROM sériel lorsque les tests d'intégration seront terminés sur le FPGA.
3. Maintenir les stations de travail de PC fixe contenant tout le matériel nécessaire à leur bon fonctionnement.
4. Éviter le transfert de matériel d'un PC à l'autre pour éviter le bris et les pertes de matériels.
5. Bien s'assurer d'attribuer des adresses différentes à chacune des cartes contrôleur.

ANNEXE A : LISTE DES COMPOSANTES

liste des Composantes

Comme nous l'avons vu précédemment la carte contrôleur est composée de:

C? Condensateur

D1, D2, D3 diode électroluminescente

H1 Connecteurs E/S

H2 Connecteur d'Encodeurs optiques et capteurs de fin de course

H3 Connecteur pour moteurs (PWM, Sens, +2 E/S)par moteurs + 4 entrés A/N

H4 Sélecteur d'horloge de programmation

H5 Connecteur pour programmation avec port parallèle.

H6 Sélecteur d'alimentation pour diode

H7 Sélecteur de données d'entrées pour la programmation

J1 Sélecteur d'IRQ (interrupt request)

J3, J4, J5, J6 Connecteur du BUS ISA format PC104

J7 Connecteur inutilisé

M? Troue de fixation

O1 Horloge 25MHz (Oscillateur)

R? Résistance

U1 Puce CP (Processeur Central) MC1401A de PMD

U2 Puce I/O de PMD

U3 Convertisseur analogique numérique AD7824

U4 FPGA Xilinx ou AT&T 3042-100

U5 Sélecteur d'adresse (5 dip switch)

U6 PROM sériel pour programmer le FPGA (Microchip 37LV36)

**ANNEXE B : SCHÉMAS LOGIQUES DE LA CARTE
CONTRÔLEUR**

ANNEXE C : SCHÉMA DE FABRICATION DU CIRCUIT

IMPRIMÉ

ANNEXE D : IMPLANTATION LOGIQUE DANS LE FPGA

ANNEXE E : LE CONVERTISSEUR ANALOGIQUE

NUMÉRIQUE AD7824

**ANNEXE F : LOGICIEL DE PROGRAMMATION PAR LE
PORT PARALLÈLE**

GLOSSAIRE

WIRED WRAP: circuit monté sur plaque perforé munie de « socket » avec de grande broches. Les liaisons entre les puces se font a l'aide d'un type de tournevis qui permet d'enrouler de petit fil d'une patte à l'autre.

PWM: signal pulsé de largeur variable. Onde carré modulé en largeur selon le ratio désiré. Un signal de 1% de PWM est haut 1% du temps et bas 99% de la période de l'onde carré (Voir livre PMD p36).

ACTUATEUR : Système de commande conçu pour engendrer une force capable de produire un mouvement.

BIBLIOGRAPHIE

Actel, ACT FAMILY FIELD PROGRAMMABLE GATE ARRAY DATABOOK, mars 1991 architecture des FPGA (s) ACT1 1020 et ACT2 1240.

Analog Devices, ANALOG-TO-DIGITAL CONVERTERS, 1996

ISA SYSTEM ARCHITECTURE, Addison Wesley Publication référence et diagramme temporels du PC-ISA.

Plamondon et Deschênes, INTRODUCTION AUX MICROPROCESSEURS, E.P.M sept. 1992 référence méthodologique.

Performance Motion Devices, PMD ADVANCED MULTI-AXIS MOTION CONTROL CHIPSET, février 1996 référence de base du contrôleur. MC1401A.

Sawan et Fréchette, SYSTEMES LOGIQUES II, École Polytechnique de Montréal, Janvier 1993

WinSystems, Volume 2.1 : EMBEDDED PCS PC/104, 1995, référence de dimension physique et des signaux du BUS ISA PC-104

Xilinx, THE PROGRAMMABLE LOGIC DATA BOOK, 1993 architecture, délais et mode de programmation du FPGA XC3042 équivalent du AT&T3042.