Rapport de Projet de Fin d'Études

Développement d'une application de maintenance de logiciel embarqué

Dan Pham

Maître de stage : Romain Poncet
Tuteur : Oum-El-Kheir Aktouf

Février - Juin 2011

Remerciements

À mes parents et à ma sœur qui m'ont soutenus tout au long de mes études.

Je remercie tout particulièrement mon maître de stage Romain Poncet, pour toute l'aide qu'il m'a apporté. Ses conseils et nos discussions autour des choix techniques et des méthodes à utiliser m'ont énormément motivé et appris.

Je remercie aussi Johann Lebon, Jean-Pierre Chevillon, Thierry Brionne pour le partage de leur expérience et leurs conseils dans le développement informatique et FPGA.

Je remercie enfin toutes les personnes avec lesquelles j'ai eu le plaisir de collaborer et qui ont pu par la même occasion m'aider durant toute la durée de mon stage.

Résumé

Le stage a été effectué au sein du département de recherche et développement de M.G.I. Digital Graphic Technology situé à Ivry-sur-Seine.

Le but étant d'ajouter une interconnectivité Ethernet Gigabit à la carte de contrôleur de têtes FPGA, de manière à pouvoir réceptionner des micrologiciels et de les écrire sur une mémoire flash. Les micrologiciels sont envoyés sur le lien Gigabit par le PC au moyen d'une interface homme-machine.

Ce stage se décompose en deux parties.

Le développement de la partie Ethernet Gigabit : Il faut développer du code matériel pour communiquer avec le contrôleur PHY Ethernet Gigabit pour émettre et recevoir sur le lien. L'implémentation d'une pile réseau minimale Ethernet/IP/UDP est ensuite nécessaire pour commniquer avec le PC sous Windows. Enfin, un premier protocole à fenêtre glissante doit être développé pour assurer l'intégrité des données ainsi qu'un second destiné à l'application de mise à jour du micrologiciel.

La réception du micrologiciel fractionné par le PC. Chacune des parties est écrite puis vérifiée sur la mémoire flash. Un mécanisme de récupération est mis en place de manière à disposer d'un système fonctionnel même si une erreur survient lors de la mise à jour.

Mots-clés : FPGA, pile, Ethernet, IP, UDP, Ethernet Gigabit, Verilog, C#

Abstract

This internship was made at the IT department of M.G.I. Digital Graphic Technology at Ivry-sur-Seine, France.

The goal is to add Ethernet Gigabit connectivity to a FPGA head controller board in order to receive and write firmware on a flash chip. The firmware is then sent over the Gigabit Ethernet link along graphical user interface.

This training is split into two parts.

The Gigabit Ethernet development part. Hardware code is developed to communicate with the Ethernet Gigabit PHY controller to emit and receive data on the link. Implementation of an Ethernet/IP/UDP stack is then necessary to communicate with a desktop under Windows. Finally, a sliding window protocol must be developed to insure data integrity as well as a second one to transfer firmwares.

The reception of a fractioned firmware from the PC. Each part is written and verified on the flash memory. A failsafe mechanism is setup in order to always have a working system, even if errors appear on the updated firmware.

Keywords : FPGA, stack, Ethernet, IP, UDP, Gigabit Ethernet, Verilog, C#

Glossaire

Field Programmable Gate Array (FPGA) : Composant électronique numérique programmable. Grâce, par exemple, au langage Verilog, des circuits sont conçus puis implantés au coeur des FPGA. Ils deviennent alors des composants personnalisés reprogrammables.

Hardware Description Language (HDL) : Un langage de description de matériel est un langage informatique permettant la description d'un circuit électronique. Celui-ci peut décrire les fonctions réalisées par le circuit (description comportementale) ou les portes logiques utilisées par le circuit (description structurelle). VHDL et le Verilog sont des HDL.

Processeur softcore : Un processeur softcore est une implémentation d'un processeur synthétisé à partir d'une description en HDL.

Intellectual property (IP) : Le code source d'un composant qui peut être implanté sur un système intégré.

Bitstream : Tableau de bits contenant la configuration matérielle du FPGA utilisée au démarrage.

Joint Test Action Group (JTAG) : Le terme JTAG, désigne le groupe de travail qui a conçu la norme et abusivement le terme générique Boundary Scan. Celui-ci permet entre autre de programmer le bitstream dans le FPGA.

Testbench : Fichier décrivant un ensemble de stimulis permettant la vérification du code matériel.

Lane : Une lane est constituée de deux paires différentielles, l'une permettant la transmission des données et la récepetion pour l'autre. L'Ethernet Gigabit utilise deux lanes.

Gigabit Media Independent Interface (GMII) : La GMII est une interface entre l'Ethernet Media Access Control (MAC) et la couche physique (PHY). Cette interface supporte des débits allant jusqu'à 1000 Mbit/s.

Application programming interface (API) : Lorsqu'un programme doit utiliser du code tiers, ces derniers sont dotés d'une API, afin de pouvoir les utiliser simplement sans connaître leur fonctionnement interne.

Table des matières

Première partie

Contexte du stage

Chapitre 1

Présentation de M.G.I.

1.1 Organisation de l'entreprise

M.G.I. Digital Graphic Technology a été fondée en 1982 par Edmond Abergel.
Acteur sur le marché des Industries Graphiques, M.G.I. conçoit, développe, et commercialise en France et à l'étranger, une gamme de Solutions d'Impression Numérique et de Finitions Multi-Supports Papiers & Plastiques.
La majeure partie de son chiffre d'affaires résulte de la vente de stations d'impression et de revenus accessoires comme les contrats de maintenance, les consommables propriétaires et les prestations associées accompagnant la vente.
Le siège social de la société est situé au 161 Avenue de Verdun, 94200 à Ivry-sur-Seine. MGI est une SA au capital de 7.960.331€ avec un effectif de 50 salariés. Son chiffre d'affaire pour l'année 2009 était de 17 millions d'euros [1].

1.1.1 Historique

A l'origine la société développait des solutions progicielles destinées notamment aux domaines financier (intégration fiscale) et paramédical sous le nom de Mulhousienne de Gestion Informatique (M.G.I).
A l'issue du développement d'un logiciel spécifique aux industries graphiques, les dirigeants ont identifié un segment de marché professionnel ne disposant pas de solutions dédiées : l'impression à la demande (*Print on Demand*).

La société, avec son centre de recherche et développement a ensuite conçu et développé la première Presse Numérique répondant aux attentes des professionnels des Industries Graphiques.

En 1991, la première génération des Presses Numérique Monochrome "Master Carte" rencontre un grand succès commercial et incite la société MGI à poursuivre ses efforts dans le développement et l'élargissement de sa gamme.

1.1.2 Domaines d'activité

La société concentre ses activités autour de 3 pôles :

— La recherche et le développement de nouvelles solutions comprenant la conception des produits et l'intégration de technologies.
— Le développement commercial au niveau national (au travers de sa force de vente dédiée) et international (au travers de sa filiale américaine et de son réseau de distributeurs).
— Le développement de consommables propriétaires et de services associés qui permettent de générer un chiffre d'affaire récurrent.

1.1.3 Ouverture internationale

Forte de son succès, M.G.I. s'est tournée vers l'international en développant son réseau de distributeurs. En 2005, le C.A international représente ainsi 49% du chiffre total du groupe M.G.I. et plus de 70% aujourd'hui [2]. Le réseau de distribution actuel en plus de la filiale américaine compte ainsi 22 distributeurs assurant une couverture internationale.
Cette présence est d'autant plus renforcée par la présence de l'entreprise dans les salons et conventions internationales de l'Industrie Graphique comme le DRUPA (Allemagne) Intergraphic, Graphitec France, Poligrafinter (Moscou), Digital print World (Londres), Print 05 (Chicago), Print on demand (New York), Graphic Americas (Miami), CardAsia (Singapour).

M.G.I. est installée dans les pays et les régions suivantes :

— FRANCE : Siège mondial de M.G.I. qui dessert la France, l'Europe, le Moyen Orient, l'Afrique et la Russie.
— ETATS UNIS : Filiale installée en Floride desservant l'ensemble du continent américain (Canada, Amérique du Nord, Centrale et du Sud) et les Caraïbes.
— ASIE PACIFIQUE : Bureau installé à Singapour desservant le continent Asiatique et la zone du Pacifique.

1.1.4 Au sein du siège de M.G.I.

L'essentiel des salariés de l'entreprise est regroupé au sein du siège social à Ivry-sur-Seine, regroupant le département de recherche et développement ainsi que l'ensemble du personnel administratif et de direction. L'équipe de recherche et développement compte 15 ingénieurs spécialisés dans les domaines de l'informatique, l'électronique, la thermodynamique, la mécanique et de la chimie.

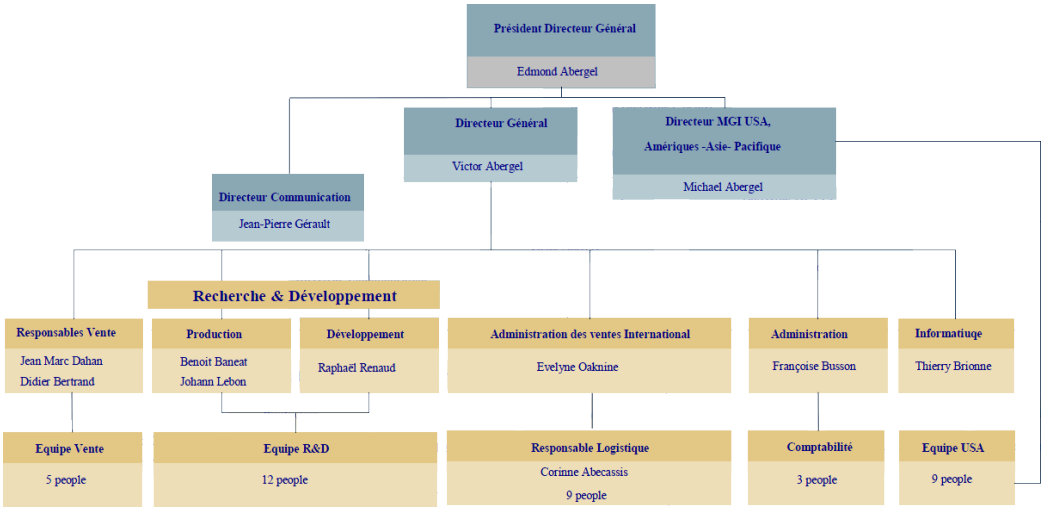

FIGURE 1.1 – Organigramme de M.G.I.

Département informatique et électronique

Le département informatique de M.G.I. est en étroite relation avec le département électronique, notamment pour l'intégration logicielle PC avec l'électronique embarqué des contrôleurs de têtes d'impression.

Les membres du département informatique et électronique possèdent les formations suivantes :

— Thierry Brionne : Directeur Informatique, présent depuis la création de l'entreprise, titulaire d'un DEA/DESS en informatique.
— Romain Poncet : Directeur Informatique adjoint, ancien élève de l'*Esisar* (Promotion 2010).
— Johann Lebon : Responsable électronique, Ingénieur en électronique de l'École supérieure d'électricité (*Supélec*).

1.1.5 Stratégies

M.G.I. a fait le choix de concentrer ses efforts et son activité spécifiquement sur des fonctions à forte valeur ajoutée. Ainsi elle maîtrise l'ensemble des étapes nécessaires à la fabrication d'un équipement (en particulier la recherche et développement, l'achat des composants, le contrôle-qualité). La société s'appuie sur un panel diversifié de sous-traitants industriels qui ne se voient alors transmettre aucun savoir-faire technologique et stratégique.

1.2 Produits

1.2.1 La presse numérique

Une presse numérique permet de réaliser la production en série de documents imprimés, de quelques dizaines à un millier d'exemplaires. Sa qualité est proche de l'imprimerie classique (technologie offset, utilisant un processus d'insolation et de gravure de plaques pour l'impression) mais sont avantage réside dans le fait qu'il soit possible d'inclure des données variables dans les documents, principalement à des fins de personnalisation.

On distingue deux filières [3] dans l'impression numérique (Figure 1.2) :

- Le *Computer-To-Press* : Proche de l'imprimerie classique à la différence que les plaques sont gravées directement sur la presse. Les presses offset numériques qui utilisent ce procédé affichent un temps de préparation d'écriture et de nettoyage entre 12 et 20 minutes, avec une impression strictement identique pour un tirage donné, et des temps de séchage significatifs mais permettant de réaliser des tirages moyens (d'environ 1.000 à 5.000 exemplaires).
- Le *Computer-To-Print* : Les systèmes électro-photographiques dont le temps de préparation est limité à quelques secondes avec une impression qui peut être totalement différente à chaque feuille s'adressent prioritairement aux marchés "à la demande" et "personnalisés". Les technologies jet d'encre et laser sont des exemples de *Computer to Print*.

M.G.I se positionne ainsi sur le secteur du *Computer to Print* en utilisant la technologie jet d'encre dont les améliorations récentes ont ouvert de nouvelles possibilités. Il est aujourd'hui possible de réaliser des impressions jet d'encre couleur avec une définition élevée, ce qui permet d'offrir une véritable alternative aux offres concurrentes actuelles que sont les copieurs numériques

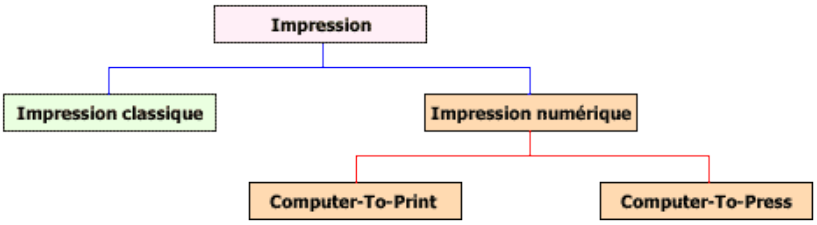

FIGURE 1.2 – Les différentes filières d'impression.

haut de gammes ou les presses numériques en proposant une grille tarifaire très abordable.

1.2.2 Quelques références

Les références présentées ci-dessous sont d'une part représentatives de la gamme des produits M.G.I. actuels [4] et d'autre part sont les produits ayant fait l'objet du travail effectué lors de ce stage.

JETcard

La *JETcard* est la première solution d'impression couleur et de personnalisation des cartes plastiques utilisant la technologie jet d'encre de M.G.I. Son fonctionnement est le suivant : la carte vierge est dans un premier temps apprêtée pour s'affranchir de la diversité des matériaux, puis passe à l'étape d'impression où est appliquée l'encre (cyan magenta jaune et noir). Ensuite, un vernis U.V. est appliqué de manière sélective ou totale sur la carte pour la protéger avant qu'elle subisse le contrôle qualité où elle pourra être éjectée en cas de non conformité. La *JETcard* permet ainsi de substituer jusqu'à 5 équipements utilisés dans la chaîne traditionnelle de production des cartes plastiques - presse offset, assembleuse, presse de lamination, puncher et encodeuse/imprimante de personnalisation ce qui en fait une solution économique et complète.

JETvarnish

La *JETvarnish* est une vernisseuse. Elle permet d'appliquer un vernis sélectif, c'est à dire qu'il est possible de choisir les zones précises de vernissage, sur un format carte allant jusqu'au 52x105mm. Afin de permettre un séchage rapide des cartes après vernissage, un séchage basé sur des lampes U.V. est appliqué. L'utilisation de la technologie jet d'encre permet ici de ne plus avoir

FIGURE 1.3 – JETcard.

recours à la création de plaques de vernissage (sorte de patrons de vernissage utilisés dans les presses offset) longues et coûteuses à mettre en place.

FIGURE 1.4 – JETvarnish.

Chapitre 2

Problématique

2.1 Présentation

2.1.1 Contexte

La société souhaite diminuer la technicité des mises à jour des produits JETcard et JETvarnish qui nécessitent actuellement un opérateur pour positionner une sonde sur chacun des contrôleurs de têtes d'impression afin de reprogrammer leur mémoire flash.

2.1.2 Besoins

Le système de contrôleur de têtes d'impression dispose actuellement d'un dispositif de mise à jour de type sériel (SPI) piloté au moyen d'une sonde par le logiciel *Xilinx iMPACT*. Le but de ce stage est d'ajouter une interconnectivité Ethernet Gigabit à une carte de développement (possèdant une spécification analogue à la carte de contrôleur de têtes d'impression) de manière à pouvoir réceptionner des micrologiciels et mettre à jour la mémoire flash de cette dernière. Les micrologiciels seront envoyés sur le lien Gigabit par le PC au moyen d'une interface homme-machine qui sera spécifiée et réalisée.
Parrallèlement, le développement de l'interface Ethernet Gigabit devra être réalisée dans l'optique de disposer d'un débit utile d'au moins 700 Mbps pour le transfert des futures données.

2.1.3 Finalités et retour sur investissement

Le travail effectué durant ce stage sera réutilisé pour permettre à terme de réceptionner via internet, le micrologiciel de chacun des contrôleurs de

têtes puis de pouvoir déclencher à distance une procédure de mise à jour. L'amélioration du dispositif de reprogrammation permettra d'automatiser le processus et d'éviter les multiples retraits/insertions de la sonde sur le connecteur du contrôleur pouvant être dommageables si l'opération est trop souvent répétée. Il permettra ainsi d'économiser un temps non négligeable sur chacune des mises à jour et d'éviter la détérioration du matériel à long terme.

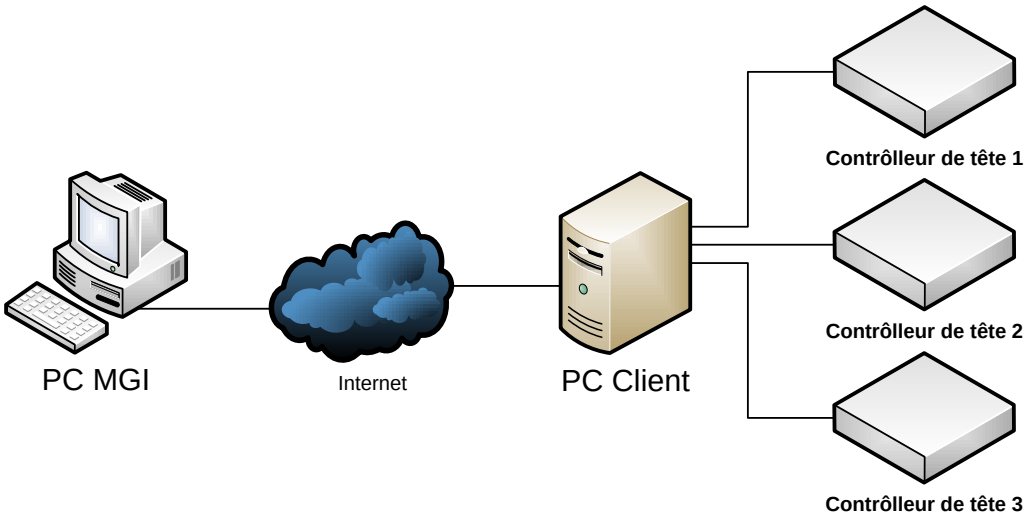

FIGURE 2.1 – Mise à jour à distance

2.1.4 Moyens mis à disposition

Matériel

Une carte d'évaluation *Xilinx FPGA SP605* est utilisée pour prototyper la future carte de contrôleur de têtes. Elle se base sur un FPGA *Spartan-6 XC6SLX45T*. Ce dernier est connecté à une multitude de composants dont en particulier :

- Un PHY *Marvell Alaska (88E1111)* pour les communications Ethernet 10/100/1000Mbps.
- Une mémoire flash 8Mo Quad SPI *Winbond*.
- 128 Mo de RAM DDR3 *Micron Technology Inc.*

Un PC récent, quadri-cœur, permet de réaliser la synthèse FPGA sous l'environnement de développement *Xilinx*. Ce dernier est aussi pourvu d'interfaces Gigabit connectées sur le bus PCI et PCI Express.

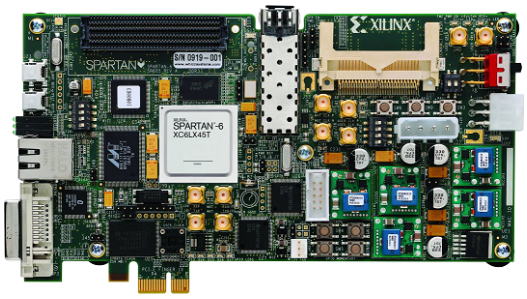

FIGURE 2.2 – Carte d'évaluation Xilinx SP605

Logiciel

Pour le développement de la cible FPGA, l'environement *Xilinx Integrated Software Environement (ISE)* est utilisé. C'est une suite de développement qui m'est familière, car utilisée lors du Projet Industriel et durant les travaux pratiques de cinquième année. Au niveau du PC, un environement *Microsoft Windows* est imposé. Pour permettre une intégration aisée à la suite logicielle existante, l'environement de développement *Visual Studio 2008 C#* est utilisé.
L'analyseur réseau *Wireshark* a eu un rôle prépondérant dans la mise au point de la partie réseau de la carte FPGA. Grâce à sa pile réseau indépendante de *Windows*, elle permet d'écouter toutes les trames réseaux mêmes si ces dernières sont erronées ou malformées. Enfin, un éditeur héxadécimal *Frhed* [5] sous license libre, a permis d'analyser la structure des fichiers Bitstream et d'éditer des micrologiciels de test.

2.1.5 Méthodologie

Ecrire du code matériel se revèle être une tâche rigoureuse : les fonctions que devra réaliser la carte FPGA étant nombreuses, il est nécessaire de possèder une spéfication claire à réaliser.
Pour vérifier le code matériel, une première simulation analytique permet de s'assurer que le code *Verilog* produit est correct du point de vue logique, sans considération des temps de propagation. A cette occasion, un test bench est décrit pour générer des stimulis en entrée du module à vérifier pour observer si les sorties sont correctes. Ensuite, un analyseur logique tel que *ChipScope*, intégré dans le FPGA, permet de connaître l'état des signaux en temps réel après implémentation de l'architecture sur le FPGA.
Une grande part d'intuition et d'expérience sont nécessaires pour faire le plus rapidement possible les bonnes hypothèses ; ce qui est crucial car certaines

hypothèses ne peuvent être vérifiées que par une synthèse complète du FPGA (lorsque des composants externes non simulables sont utilisés (tel que le PHY Ethernet)) prenant au minimum 5 minutes. Cette approche expérimentale est contraignante, mais elle oblige à bien comprendre ce que fait le code et ainsi à progresser.

Deuxième partie

Réalisation

Chapitre 3

Etat de l'art

Après l'écriture du cahier des charges fonctionnel du projet, l'état de l'art a permis de recenser les solutions existances permettant d'acheminer des données du PC à la carte FPGA de manière fiable en bénéficiant d'une vitesse de transfert au Gigabit.

3.1 Elaboration des critères d'évaluation

Le choix de travailler sous Windows a des conséquences importantes sur les premiers choix techniques. Il faut en effet gérer au minimum les couches Ethernet, IP, UDP ou TCP sur la carte FPGA. De plus, d'après une des notes technique de Microsoft, il n'est plus possible depuis *Windows XP* de créer directement des paquets IP (Raw packets). Il est alors obligatoire d'utiliser une fonction de l'*API C#* pour générer des paquets UDP et TCP ; les choix portent sur ces deux protocoles.

Dans un premier temps, une liste de critères a été mise en place pour évaluer les différentes solutions :

- *Le type de solution* : Une solution peut être logicielle, matérielle via une implémentation interne au FPGA (en code matériel : HDL, ou directement pré-cablée : Hard-macro) ou par l'utilisation de composants externes comme par exemple un microcontrôleur. Dans certains cas, on trouvera des solutions hétérogènes combinant matériel et logiciel.
- *La license* : Libre ou Payante. Payante, le coût devra être raisonnable car il pourra être multiplié par le nombre de contôleurs de têtes (de 4 à 6 par machine) suivant les modalités de la license.
- *Couches réseaux* : Chacune des solutions devra implémenter les couches Ethernet et IP puis TCP ou UDP au choix.

— *Performances* : Les performances de la solution devront permettre d'atteindre les 700 Mbps.
— *Empreinte matérielle* : L'empreinte devra être la plus faible possible pour laisser un maximum de place à la partie fonctionnelle du contrôleur de têtes.

Nom	Type	Langage	Couches	TEMAC Xilinx nécessaire?	License	
VHDL IP Stack	Hard	VHDL	UDP/IP	Non	Libre	Pile IP matérielle simple. In aussi la couche MAC mais
Silabs CP220x	MCU	N/A	TCP/IP	Non	Payante	Microcontrôleur avec PHY MCU: 30Mbps.
Microchip TCP/IP Stack	Soft	C	TCP/IP	Non	Payante	Compatible avec le microco 10Mbps. Interface SPI. Possibilité de connections Exemples de code: serveur
						La pile uIP nécessite très p Créé pour spécialement pou périphériques sous Etherne Peut être instancié comme Timer : Renvoi du dernier da Possibilité de connections

FIGURE 3.1 – Extrait de l'état de l'art.

3.2 Solution

Un tableau regroupant l'ensemble a pu être mis en place grâce aux critères préalablement définis. En résulte, 5 types de solutions dont les quatre premières ne sont pas envisageables.

— *Utiliser une hard-macro MAC avec description des autres protocoles en HDL.* La hard-macro MAC n'est pas disponible sur le FPGA utilisé.
— *Micro-contrôleur implémentant la couche MAC avec description des autres protocoles en HDL.* Les solutions avec micro-contrôleur implémentant la couche MAC ne peuvent fournir qu'une interface Ethernet de 10 Mbps.
— *Processeur softcore exécutant un OS intégrant la pile protocolaire.* La vitesse du processeur ne permet pas de traiter assez rapidement les protocoles. En résulte, un débit maximal de 40Mbps.
— *Processeur softcore exécutant le protocole en stand-alone.* Pour les mêmes raisons que la solution précédente, cette solution ne peut être envisagée.

La dernière solution et la seule viable, consiste à décrire intégralement la pile protocolaire en langage HDL. Ceci représente un travail conséquent mais nécessaire dans le but de l'obtention d'un débit de 700 Mbps sur l'interface

Gigabit. Il a été choisi de développer une couche UDP plutôt qu'une couche TCP du fait d'une plus grande complexité de cette dernière. L'intégrité des données sera néanmoins assurée par un protocole spécifique qui est spécifié et développé.
Cette étude a aussi permis de recenser une implémentation commerciale de la couche Ethernet MAC développée par *Xilinx* sous le nom d'*IP TEMAC*. Cependant, au vu de son coût, de l'impossibilité d'accéder à son code source et du gain de temps limité qu'elle peut apporter au projet, il a été décidé ne pas l'utiliser.

La solution adoptée est la suivante :

— Le PC s'appuit sur la couche réseau (Ethernet/IP/UDP) de Windows et des pilotes de la carte Ethernet Gigabit pour envoyer et recevoir les données (en violet). De ce fait, il est seulement nécessaire de développer les protocoles du gestionnaire de mise à jour (en vert).

— La carte nécessite un développement plus complexe. En effet, il faut implémenter la couche réseau ainsi que la Mise à jour du micrologiciel (en vert). Cette dernière contient non seulement les protocoles du gestionnaire de mise à jour mais aussi un module d'écriture/lecture de la mémoire flash pour enregistrer le micrologiciel. La couche réseau repose sur la couche Ethernet physique (PHY) (en violet).

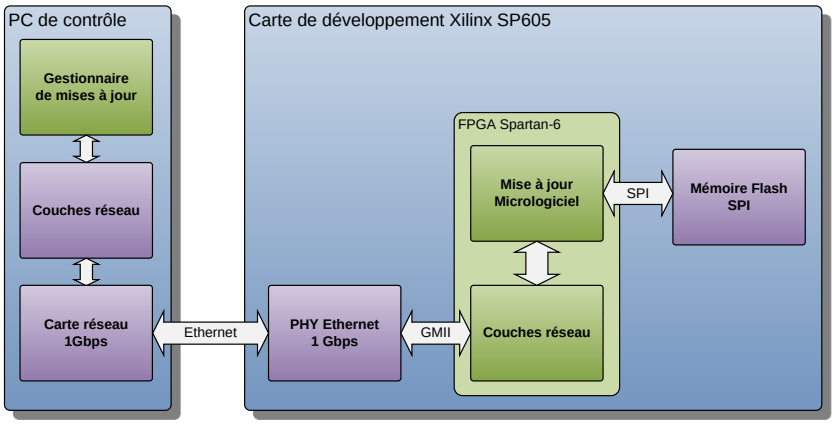

FIGURE 3.2 – Architecture globale.

Chapitre 4

Développement

Une fois la décision prise d'orienter le projet vers une solution programmée intégralement en HDL sur la carte de développement, la spécification a pu être écrite. Nous décrirons à travers ce chapitre, les détails et subtilités techniques qui ont rendu ce projet à la fois original et intéressant.

4.1 Pile protocolaire

Pour pouvoir réaliser une communication entre la carte SP605 et Windows, il faut impérativement utiliser les trois protocoles Ethernet/IP/UDP. Le fait que UDP ne garantisse pas la fiabilité des données, un protocole gérant cette dernière, appelé MGI a été développé. Enfin, un dernier protocole spécifique s'occupe lui des échanges de micrologiciels.

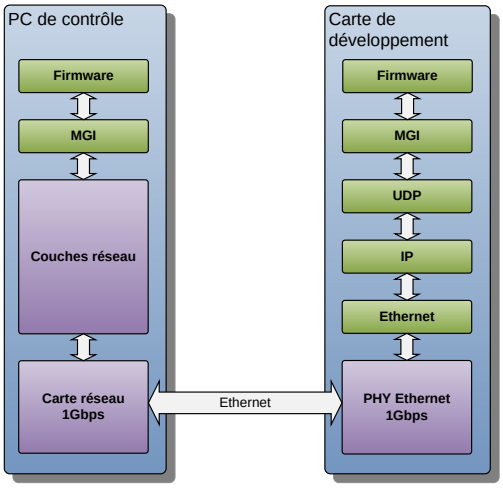

FIGURE 4.1 – Vue d'ensemble des protocoles développés.

Le fait de segmenter le protocole de fiabilité et le protocole d'échange de micrologiciels, va simplifier le remplacement de l'interface USB 2.0 par l'Ethernet Gigabit. Le diagramme de la Figure 4.1 résume les différents protocoles de communication utilisés : les protocoles identifiés en vert étant ceux développés dans ce projet et les protocoles identifiés en violet étant préexistants alors que le diagramme suivant, Figure 4.2, présente l'encapsulation des différents protocoles.

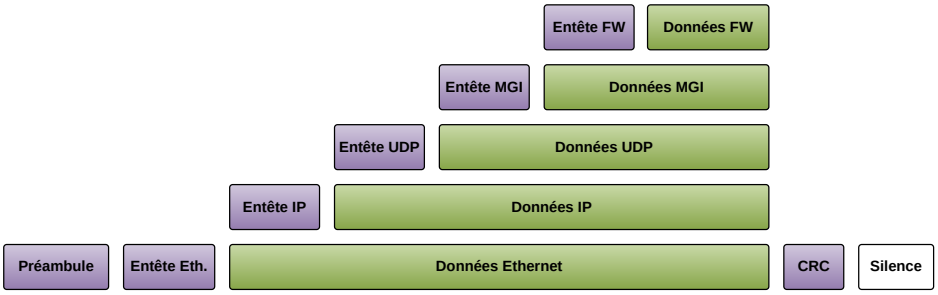

FIGURE 4.2 – Encapsulation des protocoles.

4.1.1 Couche physique

Le PHY Marvell Alaska (88E1111) Ethernet Gigabit se présente sous la forme d'un chip 128 pins au paquetage TQFP. C'est lui qui assure le codage, la gestion des collisions et la négociation du lien Ethernet. Il se connecte

d'une part au médium physique grâce à ses deux lanes et d'autre part au FPGA via l'interface (GMII). Il comporte une multitude de registres et de fonctionnalités qui permettent de choisir la vitesse du lien ou encore d'envoyer des paquets de test.

Fonctionnement de l'interface GMII

Une des grandes difficultés a été de comprendre le fonctionnemment de l'interface *Gigabit Media Independent Interface (GMII)*. En effet, seule une petite partie de la datasheet est accessible au public et les informations présentées donnent plus des détails sur sa spécification que sur son utilisation. C'est donc par recoupements avec plusieurs sites internet de développement embarqué que j'ai trouvé les informations nécessaires.

Connexion avec le FPGA

Pour la connection du PHY avec le FPGA, certains composants internes du FPGA doivent être configurés pour sélectionner les niveaux de tension et veiller à ce que les contraintes temporelles des signaux générés par l'interface GMII du PHY soient respectés. Comme cette connexion est indépendante de la couche MAC utilisée, j'ai pu trouver des indications dans le manuel d'utilisateur de l'*IP TEMAC* de *Xilinx* [6] qui y consacre une section à la connexion du PHY avec le *Spartan-6*.

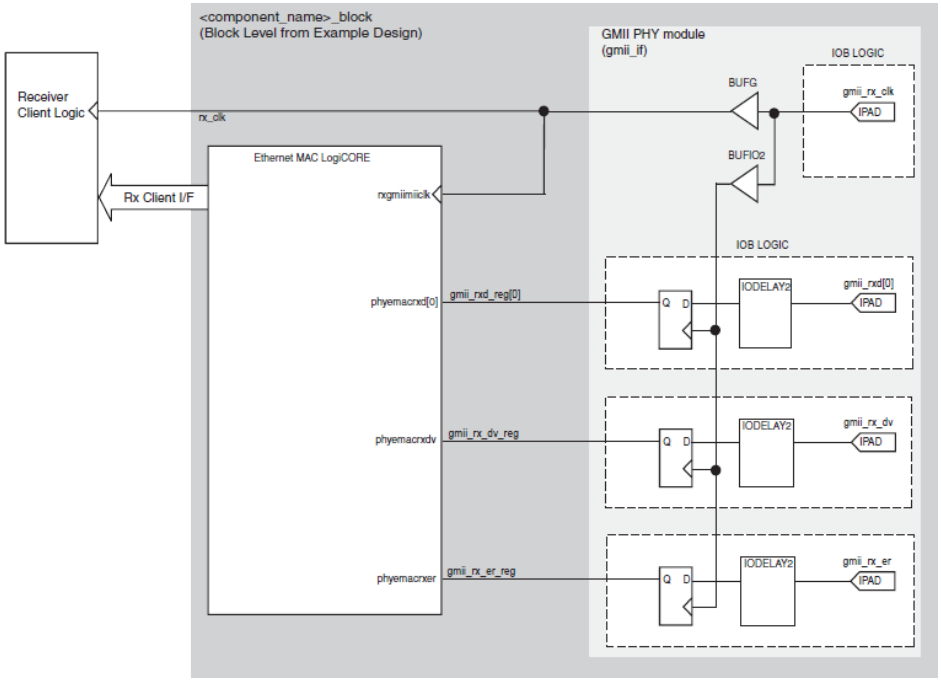

FIGURE 4.3 – Logique interne au FPGA pour s'interfacer avec le PHY.

Horloge

Le PHY fonctionne grâce une horloge de 125 MHz. Grâce à un *Digital Clock Manager (DCM)* on génére cette fréquence en interne dans le FPGA puis elle connectée en sortie sur le pin correspondant du PHY. Toutes les données transmises au PHY sont synchrones à cette horloge. A l'inverse, toutes les données transmises par le PHY sont synchrones à une seconde horloge à 125 MHz. Ces deux horloges sont légérement déphasées l'une par rapport à l'autre et il faut donc observer une grande prudence pour éviter des problèmes de synchronisme.

4.1.2 Ethernet

Les débuts de débug ont été difficiles. En effet, lorsque des bits étaient envoyés sur le lien via le PHY. La led TX du PC s'allumait. Ceci montrait que des bits étaient reçus mais rejetés par l'interface. Pour vérifier la première émission d'un paquet de la carte de développement, j'ai donc écrit un paquet en dur, possèdant toutes ces sommes de contrôles correctes. J'ai pu alors observer la réception du paquet sous Wireshark.

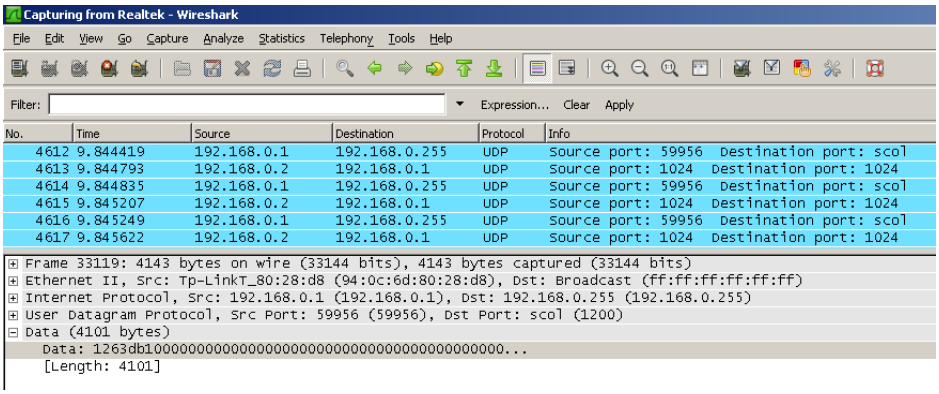

FIGURE 4.4 – Réception de paquets sous Wireshark.

Frame Check Sequence

L'étape suivante fut d'envoyer des données variables. Le FCS de l'Ethernet (implémenté par un CRC32) dépendant à la fois des données et de son entête, il faut le calculer à chaque nouveau paquet. J'ai donc adapté un algorithme de caclul de CRC32 libre trouvé sur internet pour mes besoins dont j'ai comparé les résultats avec une implémentation *C* vérifiée.
La particularité du CRC est d'être calculé en direct. A chaque top d'horloge d'émission, un octet de la trame à transmettre est positionné sur l'interface GMII. Ce même octet s'ajoute aussi à la somme de contrôle. Une fois la totalité des octets transmis, le CRC32 est envoyé sur les 4 derniers tops d'horloge.

Padding

L'envoi de message court, par exemple d'un acquitement contenant 2 octets de données utiles peut aboutir à la création d'une trame ethernet incorrecte, c'est à dire d'une taille inférieure à 46 octets. Pour pallier à ce problème, des octets de "padding" ou de bourrage sont ajoutés après les données utiles pour obtenir une trame d'au minimum 46 octets.

4.1.3 IP/UDP

Chacune des couches protocolaires (Ethernet/IP/UDP/MGI/FW) est gérée sous forme d'une machine d'état. La machine d'état (Figure 4.5) décrit le fonctionnement de la couche IP en réception. Cette machine est donc chargée d'analyser le paquet reçu afin d'extraire sa charge utile pour la transmettre à la couche suivante, la couche UDP.

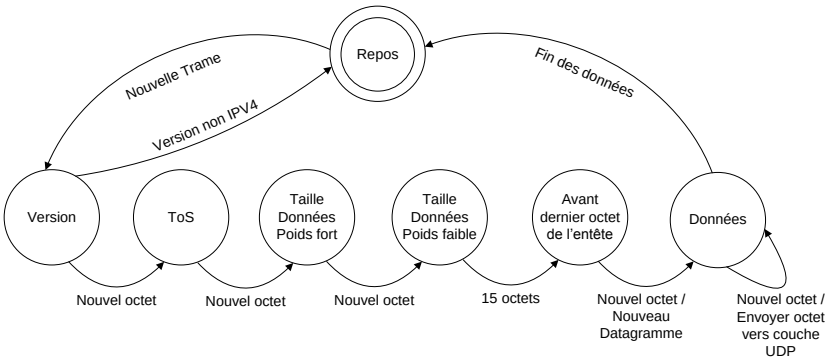

FIGURE 4.5 – Machine d'état de la couche IP en réception.

Son fonctionnement est le suivant : Lorsque la couche Ethernet a décodé l'entête d'une trame reçue, elle le signale à la couche IP pour qu'elle décode sa charge utile via le signal "Nouvelle Trame". On vérifie ensuite que le paquet soit bien en version v4. Le ToS n'est pas analysé car non utilisé. On enregistre les deux octets suivants qui indiquent la taille des N octets utiles du paquet IP. Ensuite, on finit de lire l'entête et on annonce à la couche UDP avec "Nouveau Datagramme" que le prochain octet contiendra des données à analyser. Les N octets suivants sont ensuite envoyés à la couche UDP. Le travail réalisé par la couche UDP en réception sera ensuite semblable à celui de la couche IP en observant les spécificités du protocole.

4.1.4 MGI

La couche MGI a été réalisé dans le but de d'apporter la fiabilité au protocole UDP, ce qui évite d'implémenter une couche TCP, plus complexe. Le protocole se base sur une fenêtre glissante. Sa taille est variable de manière à pouvoir répondre à l'impératif de vitesse (700 Mbps) sans pour autant dépasser la taille des buffers disponibles. Les buffers sont implémentés ici avec des FIFOs qui utilisent des blocs ram en nombre limité sur le FPGA.

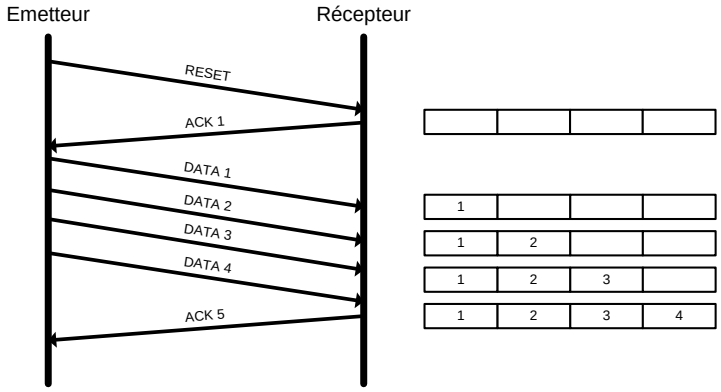

FIGURE 4.6 – Echange simple.

Le fait que la connexion soit en point à point simplifie le problème car si le paquet N+1 est reçu alors que le paquet N est attendu, alors le paquet N a été perdu. Dans ce cas le récepteur n'enverra pas d'acquittement à l'émetteur ce qui permettra à ce dernier de s'apercevoir qu'un paquet a été perdu au bout d'un temps de time-out. Pour simplifier au maximum l'implémentation sur le FPGA, lorsqu'un paquet est perdu, l'ensemble de la fenêtre d'émission est réémise. En effet, si tel n'était pas le cas, un acquittement serait envoyé pour signaler le premier paquet perdu et il serait alors nécessaire d'envoyer une information supplémentaire : la taille restante de la fenêtre.

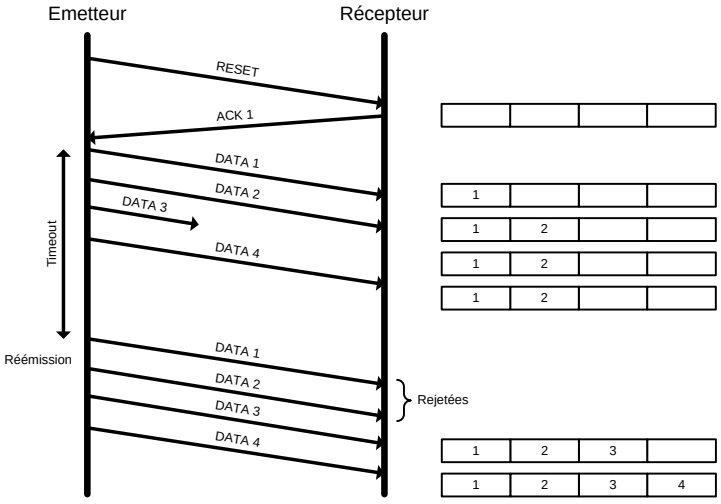

FIGURE 4.7 – Perte de paquets.

4.1.5 Firmware

Cette couche analyse les différents types de paquets transmis par la couche MGI : lecture, écriture, données, effacement et exécute l'action désirée. Pour simplifier l'implémentation, l'analyse du protocole a été séparée de l'exécution des commandes qui sont décrites dans la section suivante.

4.2 Mémoire sérielle

Les FPGAs sont constitués de bascules CMOS qui doivent être reconfigurées à chaque démarrage. Cette configuration s'effectue traditionellement via le protocole JTAG mais aussi par l'interface SPI sur des FPGAs récents comme le *Spartan-6*. Cette interface permet ainsi de rendre autonome la carte puisque cette dernière lit les données de configuration sur la mémoire SPI, au démarrage.

Lecture et écriture

Pour accèder à la flash SPI, un module contenant les différentes opérations élémentaires de lecture, d'écriture et d'effacement total de la mémoire d'une page ont été réalisées. Ce dernier se charge de générer une horloge spécifique à la puce SPI, d'envoyer les commandes, le signal d'activation (enable) en respectant les délais d'attente spécifiés par le constructeur. Un point notable dans l'écriture de la mémoire flash est l'effacement préalable de l'espace à programmer avec des '1'. En effet, la commande d'écriture peut uniquement faire passer des bits à l'état '1' vers '0' et seule une commande d'effacement peut remettre des bits de l'état '0' vers l'état '1'.

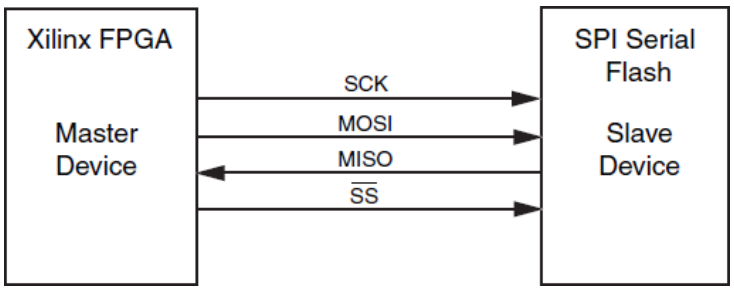

FIGURE 4.8 – Diagramme bloc de configuration de la mémoire SPI.

Ci-avant, les 4 signaux utilisés par l'interface SPI. $\overline{SS}$ permet l'activation

de la puce, MOSI et MISO sont réspectivement l'entrée et la sortie de la mémoire flash et SCK l'horloge de référence.

Mode de récupération

Un mécanisme de récupération est mis en place de manière à toujours disposer d'un système fonctionnel même si une erreur survient lors de la mise à jour.

Pour se faire, trois bitstreams doivent être au préalable enregistrés dans la mémoire flash SPI :

1. Chargé au démarrage du FPGA, il contient des bits de synchronisation ainsi que les adresses des deux bitstreams suivants.
2. "Multiboot" : Bitstream contenant le micro-logiciel courant.
3. "Golden" : Bitstream de récupération. Ce dernier n'est jamais mis à jour. Il contient une architecture minimale permettant la réception par l'interface Gigabit d'un nouveau firmware ainsi que son enregistrement à l'emplacement du bitstream "Multiboot" défaillant.

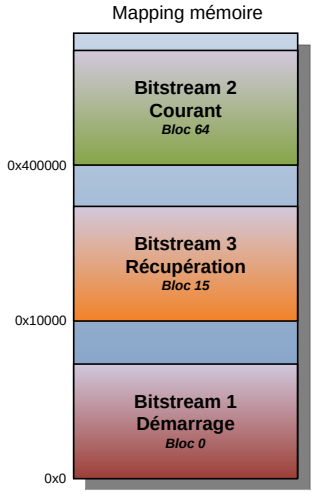

FIGURE 4.9 – Mapping mémoire de la mémoire flash.

4.3 Logiciel PC

Le logiciel PC contient l'interface Homme-Machine ainsi que les couches protocolaires MGI et Firmware.

Interface Homme-Machine

L'interface homme-machine a deux buts : permettre à l'utilisateur de sélectionner le micrologiciel qu'il désire écrire dans la mémoire flash et l'informer sur l'état de l'avancement de la mise à jour. L'interface a été réalisée avec l'éditeur graphique de *Visual Studio* qui dispose d'une bibliothéque de composants (boutton, textbox) ajoutables par un simple glisser/déposer.

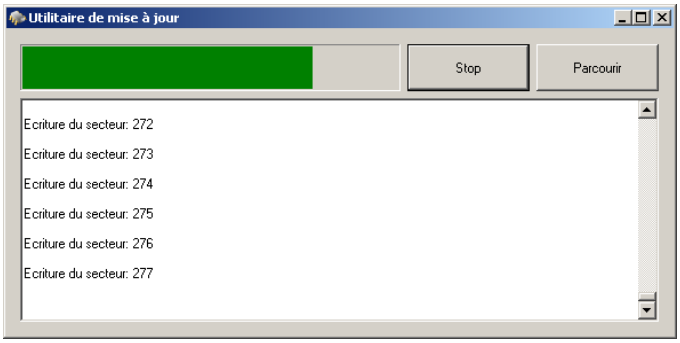

FIGURE 4.10 – Utilitaire de mise à jour.

Le bitstream

Le fichier de bitstream est créé par les outils *Xilinx* à l'issue de la synthèse du code *Verilog*. Il contient des métadonnées (nom du fichier, date d'édition, type de FPGA, taille du bitstream) ainsi qu'un tableau de bits contenant la configuration du FPGA. *Xilinx* n'a pas révélé la spécification exacte de ce fichier. Une analyse de l'entête à l'éditeur hexadécimale ainsi que quelques recherches ont permis de comprendre la structure de l'entête. L'entête est ainsi découpée en sections qui contiennent leur taille en octets (en rouge), les données (en vert) et à la fin le numéro de section sur un octet (en orange). Une fois les bits de configuration extraits du fichier bitstream, il sont envoyés par paquets de 4Ko à la carte de développement.

```
000000  00 09 0f f0 0f f0 0f f0 0f f0 00 00 01 61 00 23  ...ð.ð.ð.ð...a.#
000010  67 6f 6c 64 65 6e 5f 53 50 36 30 35 2e 6e 63 64  golden_SP605.ncd
000020  3b 55 73 65 72 49 44 3d 30 78 46 46 46 46 46 46  ;UserID=0xFFFFFF
000030  46 46 00 62 00 0e 36 73 6c 78 34 35 74 66 67 67  FF.b..6slx45tfgg
000040  34 38 34 00 63 00 0b 32 30 31 31 2f 30 34 2f 32  484.c..2011/04/2
000050  31 00 64 00 09 31 36 3a 31 32 3a 35 30 00 65 00  1.d..16:12:50.e.
000060  16 a6 74 ff ff ff ff ff ff ff ff ff ff ff ff ff  .¦tÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

FIGURE 4.11 – Entête du bitstream analysée.

Protocoles

Comme Windows implémente les couches Ethernet/IP/UDP, seules sont implémentés les couches MGI (fiabilité) et Firmware. Lors de la programmation de la couche MGI, la difficulté était de trouver des primitives d'envoi et de réception synchrones, c'est à dire qui bloquent l'exécution du programme jusqu'à ce que les données soient transmises ou reçues.

Chapitre 5

Etude financière

L'amélioration du dispositif de reprogrammation a permis d'automatiser le processus et d'éviter les multiples retraits/insertions de la sonde sur le connecteur du contrôleur pouvant être dommageables si l'opération est trop souvent répétée. Il permet ainsi d'économiser un temps non négligeable sur chacunes des mises à jour et d'éviter la détérioration du matériel à long terme.
De plus, ce projet permet de changer l'interface USB 2.0, goulot d'étranglement de l'architecture actuelle pour de l'Ethernet Gigabit.

5.1 Analyses préliminaires

5.1.1 Coût du personnel

L'avantage du dispositif est qu'il permettra à l'entreprise de réaliser elle-même la procédure de mise à jour. Des économies pourront être réalisées à la fois sur les honoraires d'un technicien formé pour effectuer les mises à jour ainsi que sur les coûts de transports, variables, suivant si un déplacement international est nécessaire. Actuellement, une mise à jour prend une journée à un technicien de maintenance. Les frais de transport s'échelonnent de 100€ à 1500€ suivant la situation géographique du client.

5.1.2 Coût du matériel

Une détérioration du connecteur permettant la mise à jour du micrologiciel est possible à long terme. Lorsqu'une mise à jour critique et obligatoire ne peut être réalisée suite à cette panne, la perte inhérente au temps de réparation peut s'élever en journées de son chiffre d'affaire. De plus, l'atteinte à son image de marque suite à l'impossibilité d'honorer ses commandes est

elle, inestimable. S'ajoute ensuite le coût des réparations s'élèvant à une cinquantaine d'euros pour chaque carte endommagée.

5.1.3 Long terme

L'interface USB 2.0, actuel goulot d'étranglement de l'architecture, ne permet pas de transmettre assez rapidement les données ceci venant nuire au nombre de cartes imprimées par minute. Les exigences en termes de vitesse d'impression augmentant constamment dans le marché de l'impression, il est nécessaire de trouver une nouvelle interface pour que M.G.I. continue à proposer des solutions compétitives répondant aux attentes du marché.

5.2 Coût du projet

5.2.1 Coût matériel

Pour ce projet, il a été nécessaire d'investir dans :

— Un kit d'évaluation *Xilinx Spartan-6* au prix de 350€ incluant non seulement la carte *SP605* mais aussi le support technique et la suite de développement complète qui permet de générer le code des micrologiciels.
— Un PC d'une valeur estimée à 1500€ .

5.2.2 Coût de développement

Le coût d'un stagiaire pour l'entreprise est de 7500€ durant les 5 mois que compte le projet. A ce titre il est intéressant de le comparer avec :

— Le coût d'un jeune ingénieur durant la même période. A raison de 3000€ par mois en incluant les charges pour l'entreprise, le coût du projet s'éleverait à 15000€ pour les 5 mois.
— Le coût d'une solution commerciale équivalente. Actuellement il n'existe pas de solution commerciale pour ce projet. Le seul élément substituable dans ce projet serait la couche réseau Ethernet MAC. Le coût de cette solution est proposé par *Xilinx* sous le nom d'*IP TEMAC* à un tarif de 1451$ pour une utilisation illimitée de l'IP pour ce projet.

5.3 Conclusion de l'étude financière

Ainsi, le coût total du projet a été de 16850€ soit une économie de 7500€ par rapport au coût d'un jeune ingénieur embauché. De plus, si un tel projet n'avait pas été mis en place, les coûts annuels engendrés par les honoraires et les déplacements d'un technicien seraient de 25000€. Cette solution est donc amortie en moins d'un an et demi.

Chapitre 6

Conclusion

Ce projet de fin d'études a été l'occasion de développer un projet FPGA conséquent en complétant mes connaissances théoriques acquises durant ma formation.

J'ai ainsi pu réfléchir à une solution nécessitant l'implémentation d'une pile protocolaire pour le support de l'Ethernet Gigabit. Son originalité réside dans le choix d'une implémentation entièrement matérielle qui bien que plus difficile à réaliser et à mettre à jour, était l'unique solution étant donné les contraintes imposées par le projet.

Ce développement m'a fait comprendre l'importance de bien modéliser le problème à ses débuts du fait que la gestion de protocoles est une tâche complexe mais qui peut être simplifiée si l'on arrive à discerner les besoins exacts du projet ; c'est ce qui rend au développement embarqué sa singularité.
Une des grandes difficultés a été de vérifier le code réalisé ; des tests benchs ont d'une part été nécessaires ainsi que des essais réels pour détecter certaines erreurs. En effet, les composants externes et les délais de propagation internes au FPGA n'étaient pas simulables.

Au terme de ce stage, c'est avec une grande satisfaction que je livre une version fonctionnelle de l'utilitaire de mise à jour qui va ainsi pouvoir être intégrée dans le prochain micrologiciel de production.

Bibliographie

[1] M.G.I. Résultats financiers M.G.I. 2009. http://www.mgi-fr.com/files/CP%20Financier%20r__sultats%202009.pdf.

[2] M.G.I. Document de base M.G.I. http://www.allegrafinance.com/index.php?option=com_docman&task=doc_download&gid=22.

[3] Jocelyne Rouis et Jean-Claude Sohm Éliane Rousset. Les presses numériques dans l'imprimerie. http://cerig.efpg.inpg.fr/icg/Dossiers/Presses_num/Chapitre_1.htm.

[4] M.G.I. Produits M.G.I. http://www.mgi-fr.com/fr/p2-produits/.

[5] Raihan Kibria. Frhed, éditeur héxadécimal sous license libre. http://frhed.sourceforge.net.

[6] Xilinx Inc. Manuel IP Tri-Mode Ethernet MAC. http://www.xilinx.com/support/documentation/ip_documentation/tri_mode_eth_mac_ug138.pdf.

Annexes

État de l'art

Cahier des charges fonctionnel

Spécification fonctionnelle