Département Informatique, Microélectronique, Automatique

Rapport de projet

Simon Malthieu

2014

Reconfiguration d'un FPGA

Table des matières

Remerciement

Je remercie Alexandre Boé, qui m'a encadré pendant ce projet et a toujours été disponible pour m'aider tout au long du projet. Je remercie aussi Thierry Flamen pour m'avoir permis de réaliser ma carte de test avec l'une de ses cartes pré-imprimées. Je remercie enfin Mickael Coronado à l'origine du projet. Il fut aussi disponible pour établir le cahier des charges et me conseiller.

Introduction

Ce rapport présente le travail effectué dans le cadre du projet de 4ème année à Polytech Lille. C'est un projet en collaboration avec une entreprise de recherche et développement en électronique, INODESIGN. Le projet a duré 9 semaines à raison de 8 heures par semaine, et a été encadré par Alexandre Boé, Thomas Vantroy, professeurs à Polytech Lille, et Mickael Coronado, gérant de INODESIGN.

Ce rapport rend compte des différentes solutions techniques mises en place dans le cadre de ce projet. Après une présentation du sujet, les solutions materielles et logicielles seront décrites avant d'aborder le développement du programme et de l'interface.

1 Présentation du sujet

Le projet porte sur la reconfiguration d'un FPGA. Les FPGA sont des circuits logiques programmables, leur principal intérêt est qu'ils peuvent être configuré pour des utilisations très précises, et sont donc d'une grande performance dans l'application pour laquelle ils ont été configuré. D'un autre coté, une fois configuré, ils ne peuvent être utilisé pour autre chose. Le but du projet est de pouvoir reconfigurer facilement un circuit FPGA grâce à une EEPROM, que le FPGA vient lire à chaque mise sous tension, et à un mini-ordinateur Beaglebone Black, qui sert d'interface de contrôle.

Le cahier des charges défini avec Mr Coronado est le suivant :

- Concevoir un programme, exécuté par un ordinateur embarqué de type Beaglebone Black, permettant d'écrire dans une EEPROM de type M25P80 un fichier bitstream de FPGA ;
- Le programme devra aussi pouvoir vérifier l'intégrité des données écrites dans la mémoire ;
- Développer une interface web permettant de contrôler le programme : Elle doit permettre de téléverser un fichier sur l'ordinateur et exécuter le programme d'écriture avec ce fichier.

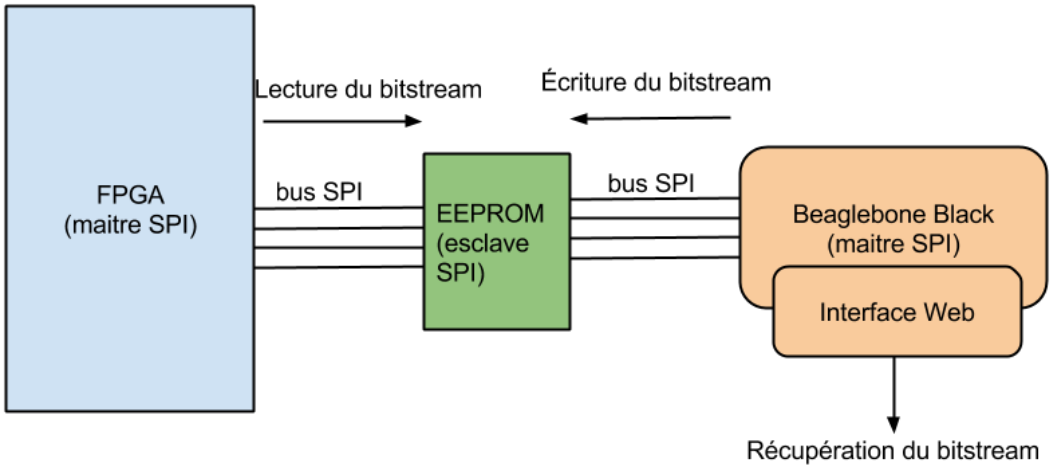

FIGURE 1 – Schéma du projet

2 Solutions techniques

2.1 Matériel

2.1.1 Beaglebone Black

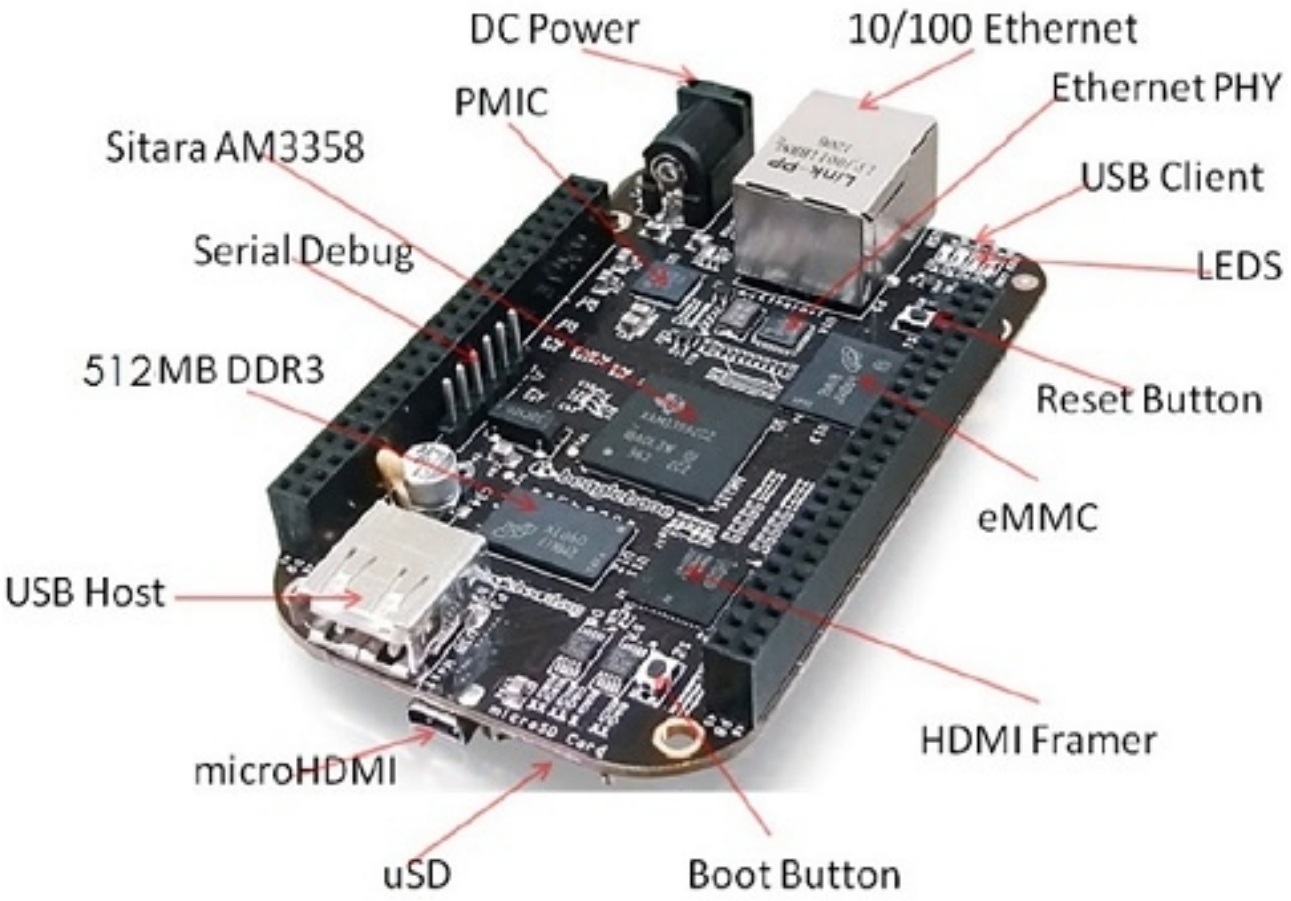

FIGURE 2 – Le Beaglebone Black et ses différentes interfaces

Le Beaglebone Black est un mini-ordinateur sous la forme d'une carte électronique. Il possède un processeur Texas Instrument AM3358, 512 Mo de RAM (DDR3), 2 Go de mémoire flash eMMC et un processeur de traitement graphique. Il est bon marché (environ 50 euros), et a été conçu dans une optique open-source. Ainsi il existe une communauté active de développeurs sur cette plateforme, permettant d'obtenir de nombreuses ressources utiles, des drivers, des systèmes d'exploitation optimisés pour cette plateforme ou les schémas de la carte électronique (pour fabriquer des extensions par exemple).

Ce ne sont pas les seuls intérêts de cette plateforme. Contrairement au Raspberry Pi, le plus célèbre de cette famille de micro-ordinateur, le processeur AM3358 peut être acheté séparément, afin de concevoir une carte spécifique par exemple. Ainsi, le Beaglebone Black est adapté au prototypage d'application. Le programme développé pour ce projet peut donc être porté sur une carte spécifique incluant le FPGA à configurer.

2.1.2 EEPROM

Une EEPROM est une mémoire ROM, donc ne peut qu'être lue, mais contrairement à une PROM, qui ne peut être programmée qu'une seule fois, l'EEPROM peut être effacée grâce à un courant électrique. Elle peut donc être effacée et reprogrammée plusieurs fois.

Dans le cadre de la configuration d'un FPGA par une EEPROM, le circuit FPGA possède une interface lui permettant de communiquer directement avec la puce mémoire via SPI. L'EEPROM choisie doit être compatible avec le FPGA, c'est à dire que non seulement elle doit posséder une interface SPI, mais doit aussi être compatible avec les commandes de lecture envoyés par le FPGA, car il n'y a pas d'intermédiaire.

Pour ce projet, une EEPROM compatible avec les FPGA Spartan 3 de Xilinx est choisie : La M25P80 de chez Micron, une EEPROM de 8 Méga bits.

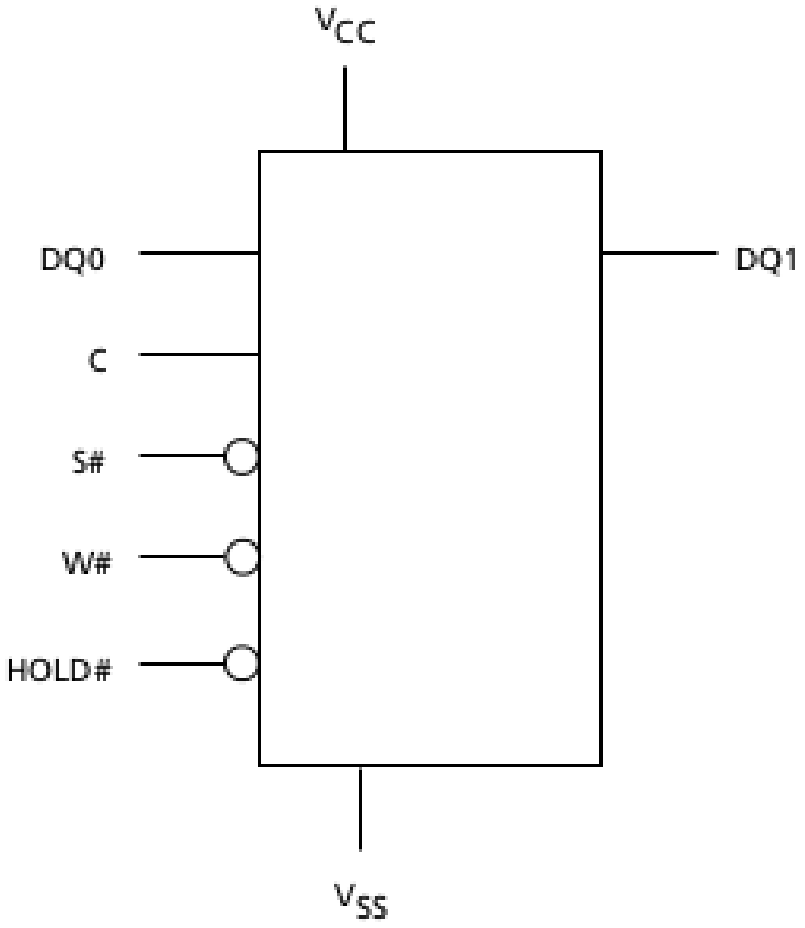

FIGURE 3 – Entrée et sortie de l'EEPROM

Description des entrées/sorties :

Vcc : Entrée d'alimentation (entre 2,5 et 3,7 V). Elle est branchée sur la sortie 3,3 V de la BBB ;

DQO : Entrée Master Out Slave In (MOSI) de la liaison SPI. C'est ici que la BBB va envoyer les commandes ;

DQ1 : Sortie Master In Slave Out (MISO) de la liason SPI ;

C : Horloge de la liaison SPI (vitesse maximum de transmission : 75 MHz ;

S# : Chip select de la liaison SPI, doit être à 0 durant la transmission ;

W# : Write protected, permet de bloquer l'écriture de façon électrique ;

HOLD# : Permet de mettre en pause une transmission ;

2.1.3 Carte de test

La M25P80 étant un composant CMS, il n'est pas possible de l'enficher directement sur une plaque d'essai (breadboard). Il faut la souder sur une carte de test. Dans un premier temps, une carte enfichable directement sur la plateforme Beaglebone Black est envisagée, et un PCB est élaboré sous Altium Designer après avoir récupéré les empreintes du composant sur le site internet d'Altium.

Par manque de temps, elle ne fut pas imprimée. En collaboration avec Thierry Flamen, l'EEPROM est soudée sur un typon pré-imprimé pour les composants CMS. Des broches sont ensuite soudées de chaque coté pour permettre d'enficher la carte sur une breadborad, et ainsi relier l'EEPROM au beaglebone Black par des fils.

Finalement, la carte de test branchée sur la breadboard présente un intérêt par rapport à une « Cape »(c'est le nom des cartes d'extensions de la Beaglebone Black, équivalent du shield Arduino) : il est facile de brancher un analyseur logique pour vérifier les signaux envoyés et reçus par la Beaglebone. Cela fut d'une grande aide lors du débogage.

2.2 Logiciel

2.2.1 Interface de communication SPI de l'EEPROM

L'EEPROM possède une interface SPI pour communiquer avec l'extérieur. C'est un bus de donnée série synchrone fonctionnant selon un schéma maître-esclave. Le bus possède 4 signaux logiques :

SCLK L'horloge générée par le maître servant à synchroniser la transmission ;

MOSI Liaison de données envoyées par le maître aux esclaves ;

MISO Liaison de données envoyées par l'esclave quand il est sélectionné ;

SS Chip select, permet de sélectionner l'esclave avec lequel le maître souhaite communiquer.

Dans notre cas, le maître est le Beaglebone, et l'esclave l'EEPROM.

La communication avec l'EEPROM M25P80 se fait toujours de la même manière : Un octet de commande, suivi éventuellement d'octets d'adresses et de données. Elle supporte un ensemble de commandes décrites ci-dessous.

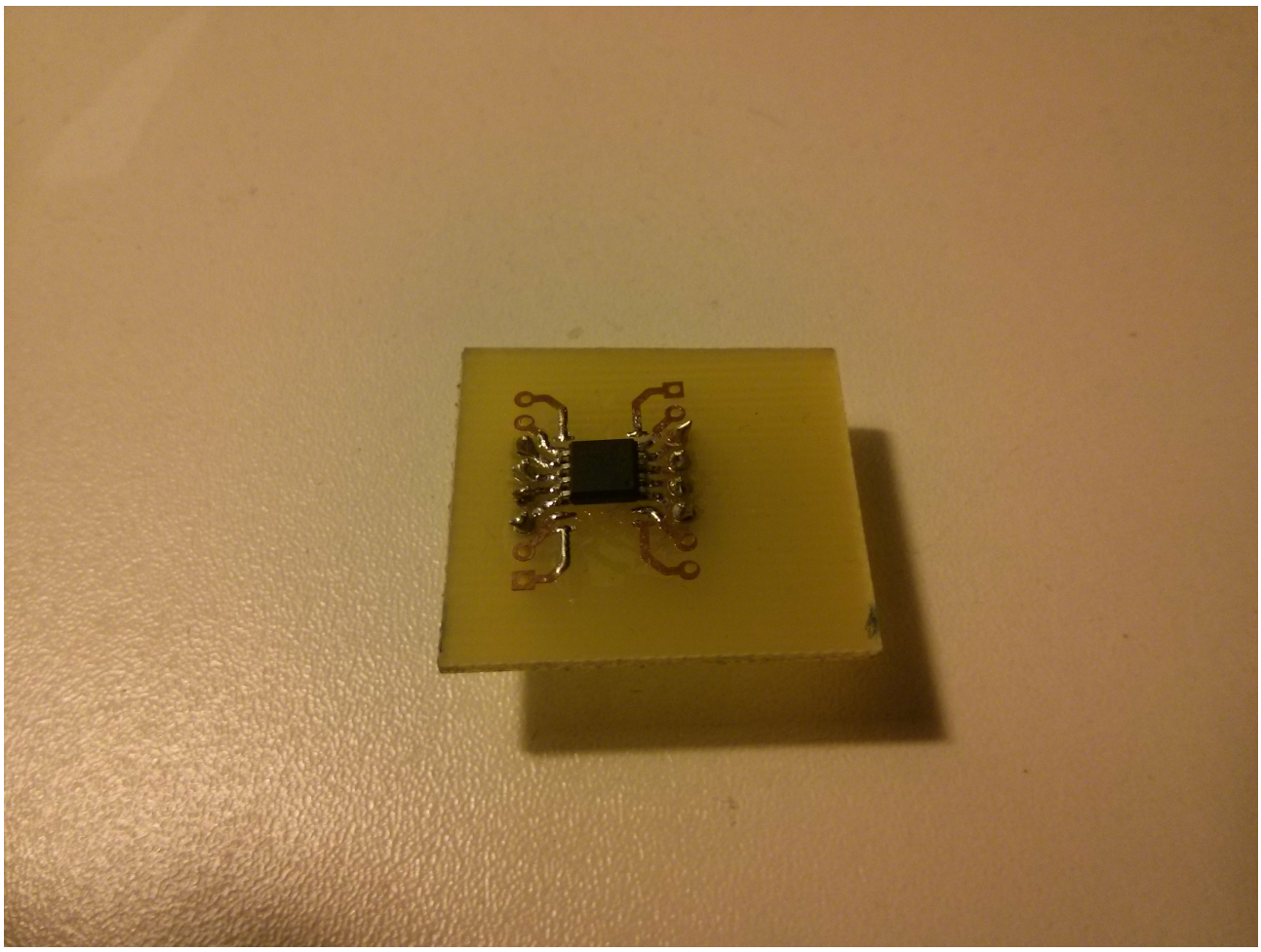

FIGURE 4 – Carte de test

Par exemple, pour lire à partir de l'adresse 0x0A0000, il faut en premier lieu sélectionner l'EEPROM en mettant le signal SS à 0, puis envoyer consécutivement sur la liaison MOSI l'octet de commande de lecture READ DATA BYTE, 03h en l'occurrence, et les 3 octets d'adresse 0Ah, 00h et 00h. Ensuite, tant que SS est à 0, l'EEPROM envoie les octets à partir de cette adresse, en incrémentant d'adresse à chaque octet.

L'EEPROM est divisée en 15 secteurs de 256 pages chacun, et chaque page contient 256 bits. L'EEPROM étant seulement effaçable par courant électrique, on ne peut donc pas écraser les données. Afin d'écrire à l'interieur, il faut d'abord effacer la mémoire, par secteur (SECTOR ERASE), ou la mémoire entière (BULK ERASE).

2.2.2 Programme en C

Le Beaglebone black ayant toute les caractéristiques d'un ordinateur, il est relativement aisé d'y installer un système d'exploitation Linux, et ainsi profiter de tous ses avantages pour développer des programmes. Le système installé sur la mémoire eMMC de 2 Go est Angstrom, un système d'exploitation basé sur le noyau Linux et optimisé pour les systèmes embarqués comme les smartphones ou les plateformes comme la Beaglebone Black.

Le système d'exploitation possède tous les paquets nécessaire pour développer rapidement. Pour la programmation, il n'est pas nécessaire d'en installer d'autres. SSH pour se connecter à

Command Name	One-Byte Command Code		Bytes		
			Address	Dummy	Data
WRITE ENABLE	0000 0110	06h	0	0	0
WRITE DISABLE	0000 0100	04h	0	0	0
READ IDENTIFICATION	1001 1111	9Fh	0	0	1 to 20
	1001 1110	9Eh			
READ STATUS REGISTER	0000 0101	05h	0	0	1 to ∞
WRITE STATUS REGISTER	0000 0001	01h	0	0	1
READ DATA BYTES	0000 0011	03h	3	0	1 to ∞
READ DATA BYTES at HIGHER SPEED	0000 1011	0Bh	3	1	1 to ∞
PAGE PROGRAM	0000 0010	02h	3	0	1 to 256
SECTOR ERASE	1101 1000	D8h	3	0	0
BULK ERASE	1100 0111	C7h	0	0	0
DEEP POWER-DOWN	1011 1001	B9h	0	0	0
RELEASE from DEEP POWER-DOWN	1010 1011	ABh	0	0	1 to ∞

FIGURE 5 – Commandes supportées par l'EEPROM (tableau issu de la datasheet)

la plateforme, GCC pour compiler et Vim pour éditer les fichiers.

Pour accéder au bus SPI du Beaglebone Black il faut paramétrer le « device tree », qui permet de définir dans quel mode sont les PINS des header (il y en a 2 de 46 pins chacun). Par exemple, les pin destinés au SPI peuvent aussi être utiliser pour de la communication audio (McASP) ou être simplement des pins génériques (GPIO). C'est un fichier lu par le système au démarrage qui permet cela.

Angstrom possède un driver pour contrôler la liaison SPI du Beaglebone Black, et fourni ainsi une couche d'abstraction sur les fonctions du noyau. Une fois le bus SPI accessible, un périphérique virtuel est disponible dans /dev/ sous le nom de spidev1.0. Il permet d'envoyer des trames directement sur le bus spi grâce à des fonctions read() et write().

La communication avec Read et write étant Half-duplex, il est possible de faire de la communication Full-duplex (écriture et lecture simultanées) grâce à la fonction ioctl(). ioctl prend en paramètre une structure spi_ioc_transfer, qui caractérise la trame qui va être envoyée. Ses champs sont décris ci-dessous :

tx_buf Buffer de transmission envoyé sur MOSI ;

rx_buf Buffer de réception rempli avec la réponse de l'esclave sur MISO ;

len Longueur de la transmission, c'est à dire le nombre d'octets pendant lequel le chip select doit être à l'état bas ;

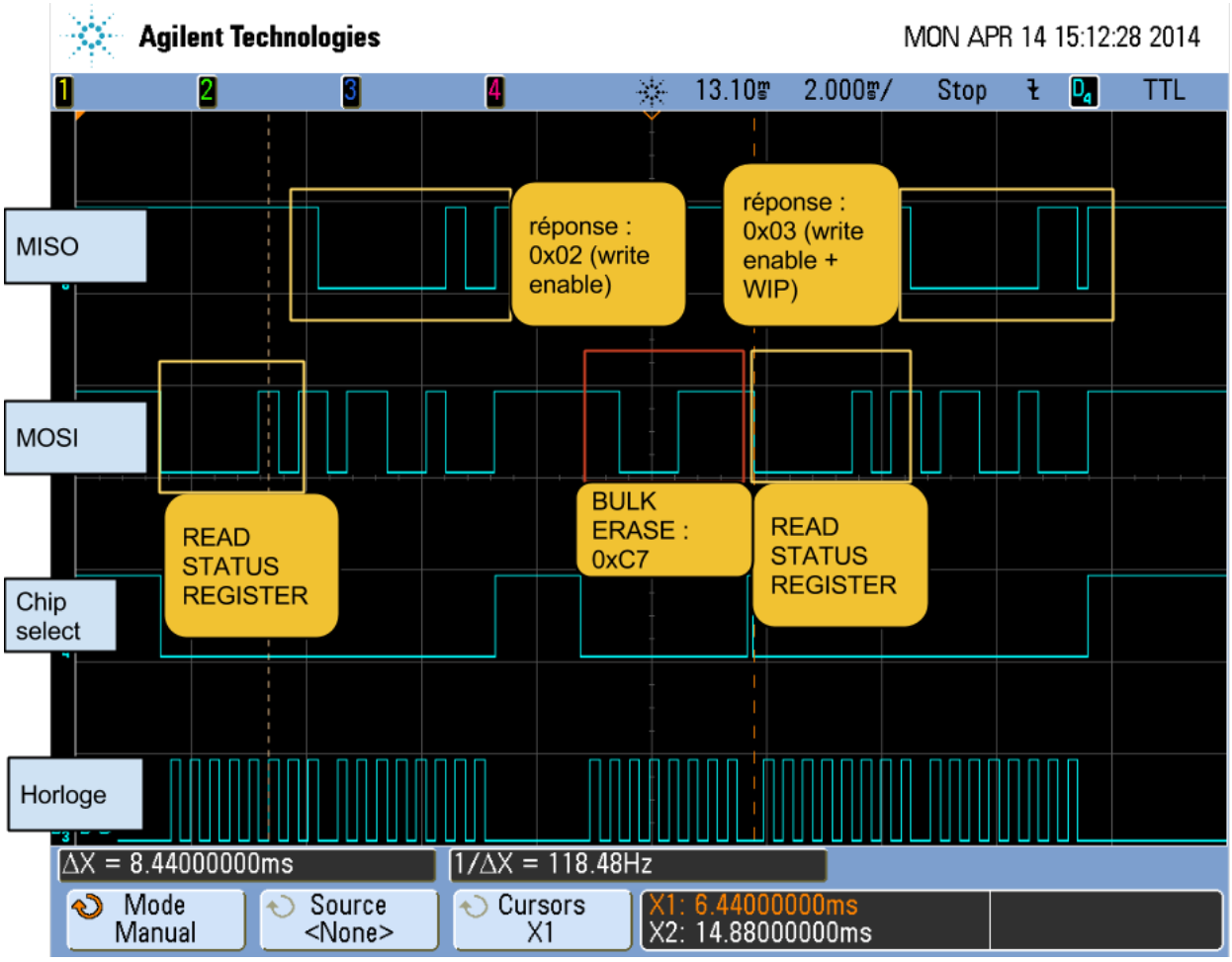

FIGURE 6 – Exemple de trame SPI

speed_hz Spécifie la vitesse de la transmission en Hertz ;

delay_usecs Délai après le dernier bit transféré avant de mettre le Chip Select à 1 ;

cs_change Permet de désélectionner l'esclave ou non entre deux transferts.

La configuration d'un FPGA est contenu dans un fichier « bitstream », où le routage et les connexions à faire dans le FPGA sont écrites. C'est ce type de fichier qui est destiné à être écrit dans la mémoire. Le programme permet donc de donner en paramètre un nom de fichier à écrire dans la mémoire.

Le FPGA peut être au coeur d'un système embarqué sensible et diffcile d'accès. C'est pour cette raison qu'il faut s'assurer de son bon fonctionnement, et notamment vérifier l'intégrité du bitstream écrit dans l'EEPROM. Le programme relit donc entièrement la mémoire et stocke son contenu dans un fichier « bitstream.test ». la comparaison entre le bitstream original et sa copie réalisée à partir de la mémoire seront comparés par le programme unix « cmp »via l'interface web.

Voici les différentes étapes de l'algorithme permettant d'écrire le bitstream puis de lire la mémoire, avec les commandes transmises en SPI entre parenthèses :

1. Activation de l'écriture afin de pouvoir effacer (WRITE ENABLE) ;
2. Effacement complet de la mémoire. Cette étape dure environ 6 secondes (BULK ERASE) ;

3. Lecture en boucle du registre de status pour vérifier que la mémoire n'est pas occupée en écriture. Si c'est le cas, le bit d'écriture en cours (WIP) est positionné à 1 (READ STATUS REGISTER) ;
4. De nouveau write enable ;
5. Ouverture du fichier de bitstream ;
6. Écriture par morceau de 256 octets (taille d'une page de la mémoire) (PAGE PROGRAM) ;
7. Lecture de la mémoire par paquet de 4096 octets : Ce n'est pas la mémoire qui impose cette limite mais la fonction ioctl(), qui n'autorise pas les transfert en SPI de plus de 4096 octets (READ DATA BYTES) ;
8. Stockage du buffer de lecture dans un fichier afin qu'il soit comparer au bitstream original par l'interface web.

Le programme fonctionne, mais il est vrai que donner un nom de fichier en paramètre d'un programme en ligne de programme puis lancer un autre programme de comparaison n'est pas pratique : il faut se connecter à l'ordinateur, copier le fichier bitstream puis exécuter le programme.

Ainsi pour résoudre ces problèmes et améliorer l'expérience utilisateur une interface web a été développée.

2.2.3 Interface Web

Pour créer l'interface web, des logiciels supplémentaires sont nécessaires : lighttpd pour le serveur Web, et son module Fast-CGI permettant d'activer PHP. PHP, le langage exécuté coté serveur le plus répandu. Ici, il sert à uploader le bitstream de l'utilisateur, puis à exécuter directement le programme d'écriture de l'EEPROM depuis le serveur web.

La fonction PHP exec() permet d'exécuter facilement un programme contenu sur le serveur (ici le beaglebone), et de récupérer les lignes de la sortie standard dans un tableau. De cette façon, les erreurs sont récupérées dans un tableau et affichées sur la page web.

C'est aussi cette fonction qui est utilisée pour vérifier l'intégrité des données. Elle éxecute le programme cmp (compare) sur le bitstream uploadé et le fichier créé à partir de la lecture de la mémoire. Ce programme ne renvoie rien si les deux fichiers sont identiques. L'interface web détermine ainsi si le bitstream a été corrompu lors du transfert.

Les pages sont en HTML simple.

3 Amélioration et perspective

Ce projet en collaboration avec INODESIGN a été proposé par le gérant de cette entreprise dans le cadre du stage que je vais effectuer dans leur locaux cet été. Ce projet constitue donc une partie du stage. Dans ce cadre là, les améliorations vont pouvoir se concrétiser facilement. Initialement, le projet devait aussi porter sur la reconfiguration via liaison J-TAG. Néanmoins,

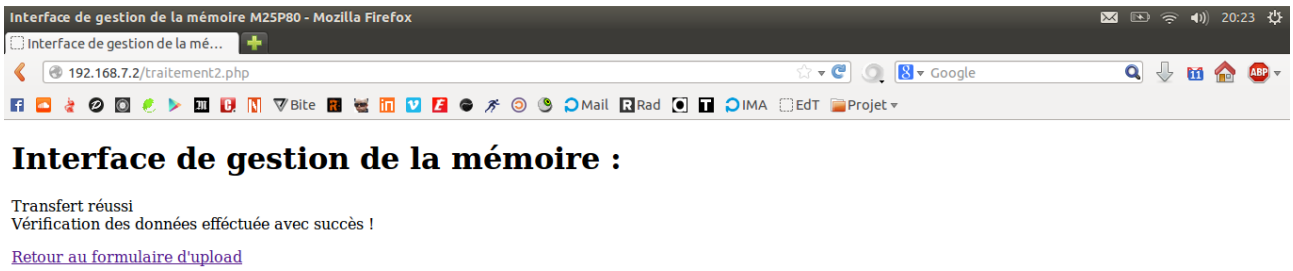

FIGURE 7 – Page de traitement après une écriture réussie

l'interface web a été privilégié afin d'obtenir un rendu plus « visuel »pour le rendu. C'est un axe qui pourra être développé plus tard, car c'est une fonction recherchée par INODESIGN. Le Beaglebone Black n'ayant pas d'interface J-TAG, il faudra émuler les signaux J-TAG avec les pins GPIO.

L'interface web est aussi grandement améliorable. Non seulement au niveau visuel, en ajoutant du CSS, mais aussi au niveau fonctionnalités : Gérer les bitstreams présents sur le serveur, écrire et lire l'EEPROM en un clic, mettre l'EEPROM en DEEP POWER MODE (un mode veille où la mémoire ne consomme plus que quelques micro ampères), de même qu'améliorer la sécurité du site web. Il est vrai que pour le moment l'interface permet de téléverser n'importe quel fichier sur le serveur. Il serait intéressant de vérifier que les fichiers uploadés sont bien des bitstreams, et ajouter un module d'authentification.

4 Conclusion

Ce projet fut intéressant à bien des égards, mais n'a pas été sans difficultés. Une des difficultés rencontrées n'a pas été technique, mais à porté sur la gestion de projet. Faire un projet tout seul n'est pas si facile que je le pensais. Il est vrai qu'on a une vision globale du projet, mais une difficulté peut bloquer tout le processus, car on manque constamment de recul pour avoir de nouvelles idées. Le processus de développement est plus lent. Bien sûr, c'est agréable de ne pas être dépendant d'un binôme, mais cela entraîne plus de responsabilités.

La deuxième grosse difficulté a été les problèmes électriques. J'ai plus de compétences en informatique, j'ai donc eu du mal à cerner le problème, m'obstinant à chercher une solution logicielle.

Enfin, j'ai sous-éstimé l'importance de la datasheet. Une connaissance plus approfondie de celle-ci m'aurait permit d'avancer plus vite et d'éviter beacoup d'erreurs, notamment à propos du fonctionnement des EEPROM et du mode d'adressage de la M25P80.

J'ai tout de même appris pleins de chose durant ce projet, notamment sur la liaison SPI, les mémoires EEPROM et les plateformes comme la beaglebone black. J'ai aussi appris à lire efficacement une datasheet, ce qui est plus important que ça en à l'air.