

# Rapport

**Projet d'Architecture des systèmes embarqués - Quartus Carte Radar 2D**

Membre : Vladislav Levovitch BALAYAN  
Contribution précieuse : Daniel FERREIRA LARA

Aide rédactionnelle, assistance au débogage, support pour le code, et remerciements à ceux qui m'ont guidé et apporté leur expertise : Ayoub, Nicolas, Liza, Salma, Farah, Vladislav, Maxime, Ayman, Yulin, Victor, Quentin, ChatGPT, Benjamin, Lucien Bachelard, Monsieur ABDELAZIM Abdelrahman, Monsieur DOUZE Yann et l'incroyablissime Monsieur VIATEUR.

## Remerciement

Mes remerciements vont évidemment tout d'abord à mon ami, Daniel FERREIRA LARA, qui, a su m'épauler tout au long de cette année, m'encourager et me donner tout l'espoir qu'il a pu.

Un très grand merci aussi à Monsieur MARTIN qui m'a beaucoup appris pendant nos séances encadrées depuis le cinquième semestre. Avec son collègue Monsieur PINNA, qui par le cours d'électronique numérique 1, m'ont fait découvrir une nouvelle passion pour ce langage étant l'amour parfait entre l'informatique et l'électronique.

Mes remerciements vont aussi à Monsieur ABDELAZIM qui a toujours fait preuve d'une grande pertinence dans ses réponses même lorsque mes questions étaient des plus basiques. Merci à lui et à notre très bien aimé Thibault (alias Papa des Ei) qui ont beaucoup participé à mon épanouissement personnel et qui m'ont appris à mieux structurer mes codes.

Pour finir, j'aimerais remercier toutes ces personnes extraordinaires que j'ai rencontrées cette année dans la formation de mes rêves depuis de nombreuses années, et qui, par leur simple présence, me rendaient heureux : les copains du TPE, Yulin, Daniel, Inessa, Xiaochen, Jana, Joakim, Zimeng, Nicolas, Maxime, Ayman, Victor, Quentin, Papa, Ilias, Benjamin, Nadir, et ceux du TPA, nos incroyables alternants avec une mention toute particulière pour ces perles inoubliables qui m'ont accompagné tout au long de l'année dernière dans tous nos différents travaux en traitement du signal : Liza, Jihen, Amel, Salma, Farah et Ioana, les alternants de mon groupe, Asdjad, Fatou, Enes, Hakan, Vedat, Amine, les deux Tarek, Michel, Emmanuel, et les alternants qu'on ne pouvait voir que trop rarement grâce aux restructurations mais qui ont tout de même su rayonner : Arthur, Léa, Gloire A Dieu, Wassim, Ali, Elsa, Karlitou, Sékouba, Tom et pour finir Mon Pitit Chou.

## Introduction

Dans le cadre de notre parcours universitaire en 4ème année d'école, nous sommes amenés, en Électronique Numérique 4, à réaliser et simuler un cœur de processeur mono-cycle. Pour une meilleure gestion de notre projet, nous avons, sur les conseils de Thibault, codé dans un environnement de travail pertinent avec l'IDE VScode, un suivi avec [GitHub](#) et une remise en ligne via [Google Doc](#).

Ce projet ambitieux implique la conception et la simulation de différents blocs principaux tels que l'unité de traitement, l'unité de gestion des instructions et l'unité de contrôle. L'objectif est de diviser le projet en petits composants tels que des multiplexeurs, des registres, des bancs de mémoire, ainsi qu'une Unité Arithmétique et Logique (UAL).

Cela nous permettra de simuler, d'assembler et de tester notre modèle sur une carte FPGA.

## Table des matières

<b>Remerciement.....</b>	<b>2</b>
<b>Introduction.....</b>	<b>2</b>
<b>1. Télémètre ultrason.....</b>	<b>5</b>
1.1. Implémentation en VHDL du télémètre seul.....	5
Figure 1.1.1 : Chronogramme du télémètre à ultrason.....	6
1.2. Essai sur carte FPGA du télémètre seul.....	6
Figure 1.2.1 : Test hardware du télémètre pour $d = 10 \text{ cm}$ .....	7
Figure 1.2.2 : Test hardware du télémètre pour $d = 4 \text{ cm}$ .....	7
Figure 1.2.3 : Test hardware du télémètre pour $d = 6 \text{ cm}$ .....	7
1.3. Implémentation en VHDL du télémètre compatible avec l'interface Avalon.....	8
Figure 1.3.1 : Chronogramme du télémètre à ultrason pour l'interface Avalon.....	8
1.4. Programmation logicielle sur carte FPGA du télémètre.....	9
<b>2. Servomoteur.....</b>	<b>10</b>
2.1. Implémentation en VHDL du servomoteur seul.....	10
Figure 2.1.1 : Chronogramme du servomoteur seul.....	10
2.2. Test unitaire sur carte FPGA du servomoteur seul.....	11
Figure 2.2.1 : Test hardware du servomoteur pour un angle de $90,0^\circ$ .....	11
Figure 2.2.2 : Test hardware du servomoteur pour un angle de $45,0^\circ$ .....	12
Figure 2.2.3 : Test hardware du servomoteur pour un angle de $0,0^\circ$ .....	12
Figure 2.2.4 : Test hardware du servomoteur pour un angle de $32,4^\circ$ .....	12
2.3. Implémentation en VHDL du servomoteur compatible avec l'interface Avalon.....	13
Figure 2.3.1 : Chronogramme du servomoteur compatible avec l'interface Avalon... 13	13
2.4. Programmation logicielle sur carte FPGA du servomoteur.....	13
<b>Conclusion.....</b>	<b>14</b>
<b>Annexe.....</b>	<b>15</b>

## 1. Télémètre ultrason

Le télémètre ultrason est un dispositif qui permet de mesurer une distance en utilisant la réflexion d'ondes sonores. Cet outil est largement utilisé dans des applications telles que la robotique, la cartographie et l'évitement d'obstacles. Dans notre projet, il joue un rôle crucial pour cartographier une scène en mesurant les distances entre le capteur et les obstacles dans son champ de détection.

### 1.1. Implémentation en VHDL du télémètre seul

Le capteur utilisé, le HC-SR04, fonctionne en envoyant une onde ultrasonore et en mesurant le temps que cette onde met à revenir après avoir été réfléchie par un obstacle. Ce temps est directement proportionnel à la distance entre le capteur et l'obstacle.

Pour déclencher une mesure, un signal Trigger de 10 µs est envoyé au capteur. Ce dernier génère une série d'ondes ultrasonores et active un signal Echo, dont la durée reflète le temps aller-retour de l'onde. La distance est ensuite calculée à l'aide de la formule :

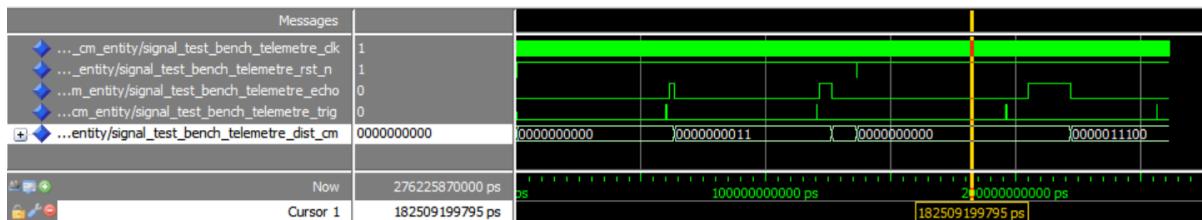
$$d = v_{son_{340}} \times t_{demi} = \frac{340}{2} t_{in} = 170t_{in}$$

où  $v_{son_{340}}$  la vitesse de propagation du son (340 m.s-1) et  $t_{demi}$  la durée mesurée, divisée par deux pour prendre en compte l'aller-retour.

Dans notre cas, avec une fréquence d'horloge de 50 MHz, chaque cycle correspond à une durée de 20 ns, et une impulsion de 1 µs représente 500 cycles. Ainsi, un centimètre de distance équivaut à environ 2 941 cycles d'horloge, soit un temps de 0,00005882s. Pour simplifier nos calculs, nous avons approximé ce facteur en arrondissant à 3 000 cycles (  $200 \times 15$  coups d'horloge ) par centimètre soit un temps de  $60 \times 1\mu s$ , introduisant une erreur minime mais acceptable pour nos besoins.

Pour l'implémentation, nous avons adapté cette formule dans notre code VHDL afin que la mesure de la durée d'Echo soit convertie en distance. Nous avons utilisé un compteur principal pour accumuler les cycles lorsque le signal Echo est actif. Cette durée est ensuite divisée par le facteur vu précédemment de 60 pour obtenir une estimation de la distance en centimètres.

Figure 1.1.1 : Chronogramme du télémètre à ultrason.



Pendant les simulations sur Modelsim, cette approximation a été testée avec plusieurs valeurs d'impulsions d'Echo. Par exemple, une impulsion de 2 ms a donné une mesure de 3 cm, correspondant à une distance calculée de 3,39 cm, tandis qu'une impulsion de 5 ms a produit 8 cm, proche de 8,47 cm théoriques. Enfin, une impulsion plus longue de 17 ms a donné 28 cm, avec une valeur réelle de 28,81cm. Ces résultats, bien qu'ils montrent une petite erreur de l'ordre d'un centimètre, confirment la validité de notre approche.

La simulation a d'abord permis d'identifier une anomalie dans le système, où la distance affichée ne se mettait à jour qu'après la réception d'un nouvel écho. Toutefois, cette logique a été revue sur les conseils de Monsieur DOUZE, et il a été décidé de conserver la dernière valeur valide tant qu'une nouvelle mesure n'est pas confirmée. Cette approche vise à éviter que des valeurs erronées ne soient affichées en cas de prélèvements trop rapides, ce qui pourrait entraîner une estimation incorrecte de la distance, comme une valeur nulle. Une telle situation pourrait laisser penser à un obstacle, alors que l'environnement pourrait en réalité être dégagé. Nous avons ainsi modifié le code pour que la valeur affichée reste stable jusqu'à la réception d'un nouvel écho confirmé.

Enfin, un test du comportement du système en cas de reset asynchrone a été effectué, confirmant que le reset interrompt correctement les opérations en cours et réinitialise tous les signaux à leur état initial.

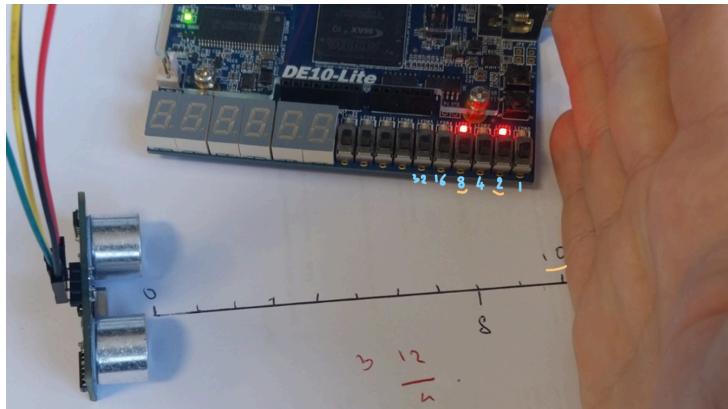
## 1.2. Essai sur carte FPGA du télémètre seul

Dans cette étape, nous avons réalisé des tests matériels en mode standalone pour vérifier le bon fonctionnement de l'implémentation VHDL du télémètre. L'objectif principal était de valider que le modèle proposé répond aux attentes en termes de précision et de cohérence des mesures. Ces essais ont également permis de corriger une erreur dans le coefficient utilisé pour convertir les cycles d'horloge en distance. Une erreur de facteur 10 avait initialement été introduite dans les calculs. Grâce à un protocole expérimental itératif, nous avons ajusté ce coefficient et obtenu des résultats précis et fiables.

Bla bla pour vérifié la proportionnalité.

Le protocole consistait à connecter le télémètre HC-SR04 aux broches GPIO de la carte FPGA. La sortie, correspondant à la distance mesurée, a été connectée aux LEDs rouges (LEDR[9..0]) de la carte DE10-Lite, permettant une visualisation directe des résultats.

Figure 1.2.1 : Test hardware du télémètre pour  $d = 10 \text{ cm}$



Lors du premier test, nous avons placé un obstacle à une distance de 10 cm du capteur. Les LEDs indiquaient bien le bit de donnée 8 + 2. Ce résultat a permis de vérifier la juste proportionnalité entre la durée du signal echo et la distance réelle, et de valider les calculs de conversion pour cette plage de mesure.

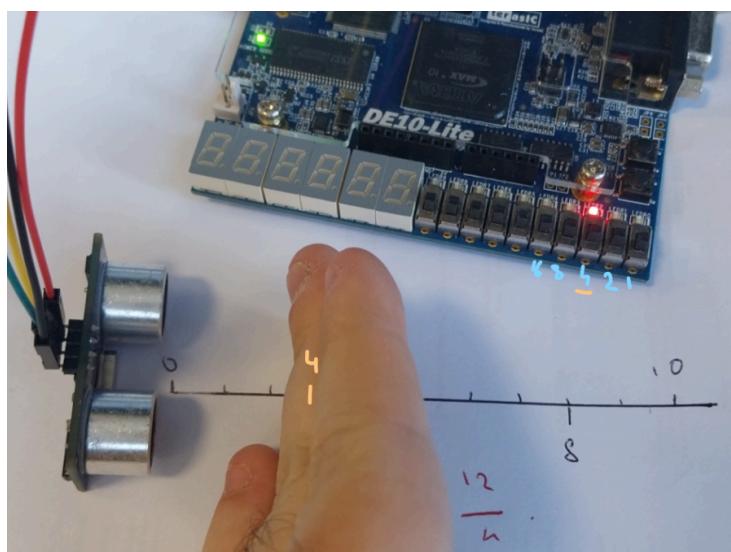
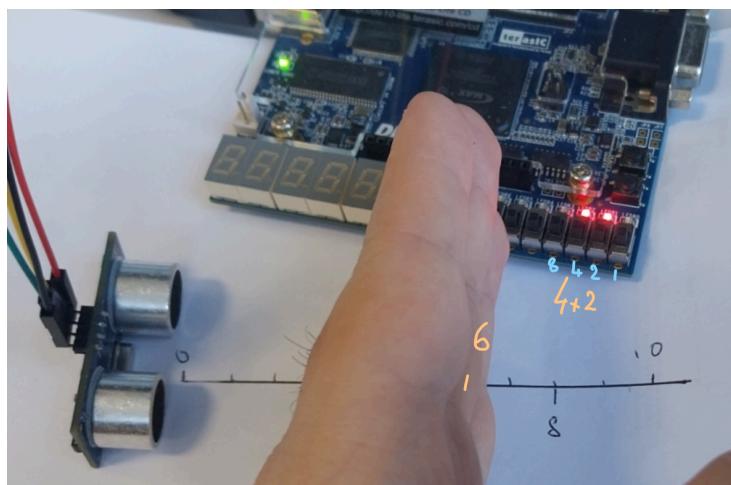


Figure 1.2.2 : Test hardware du télémètre pour  $d = 4 \text{ cm}$

Un second test a été mené avec un obstacle à 4 cm. Cette distance plus courte visait à évaluer la précision du système dans une plage très rapprochée. Les résultats étaient conformes, et aucune anomalie n'a été relevée, confirmant la robustesse du modèle même pour des distances proches.

Figure 1.2.3 : Test hardware du télémètre pour  $d = 6 \text{ cm}$

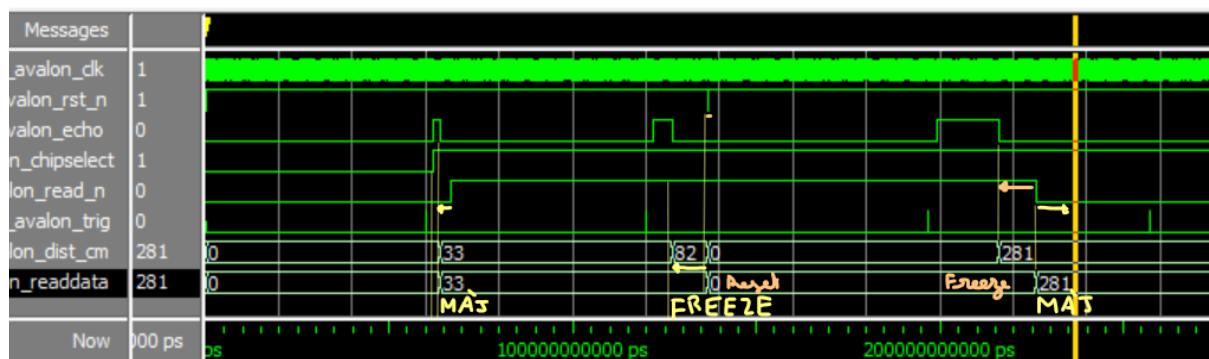


Enfin, une variation de 2 cm a été introduite, en déplaçant cette fois ci l'obstacle à 6 cm. Ce test visait à vérifier la capacité du système à détecter des variations subtiles de distance. Le télémètre a mesuré la distance correctement, avec une précision suffisante pour distinguer clairement ce petit écart de 2cm.

### 1.3. Implémentation en VHDL du télémètre compatible avec l'interface Avalon

Afin d'interconnecter le module télémètre au reste du système de manière standardisée, nous avons intégré l'interface Avalon - MM. Cette intégration garantit une compatibilité avec les autres périphériques et permet une gestion centralisée des données par le bus Avalon. Pour cela, nous avons ajouté les signaux spécifiques à Avalon comme chipselect pour l'activation du module, read\_n pour la lecture ( actif à l'état bas ) et readdata pour transmettre les données mesurées.

*Figure 1.3.1 : Chronogramme du télémètre à ultrason pour l'interface Avalon*



Les tests effectués sur cette nouvelle architecture ont confirmé son bon fonctionnement. Nous avons observé que, dans un état initial, avant toute réception d'un signal Echo, la valeur de readdata restait nulle. Lorsqu'un signal Echo est détecté, la distance mesurée est correctement calculée et mise à jour. Cependant, si la lecture est désactivée (avec read\_n à 1), la valeur de readdata reste figée, même si une nouvelle mesure est effectuée. Ce comportement garantit la cohérence des données sur le bus Avalon.

Un test particulier a été mené pour vérifier la gestion du reset. Lorsque le signal de reset est activé, tous les compteurs internes et les données sont correctement réinitialisés à zéro, assurant ainsi une remise à l'état initial du système. En outre, après une série d'impulsions d'Echo, nous avons confirmé que la distance était calculée et que les données étaient transmises sur readdata uniquement lorsque read\_n repassait à l'état bas.

Ces résultats montrent que l'implémentation Avalon fonctionne comme prévu et permet une intégration fluide du télémètre dans un système SoC.

## 1.4. Programmation logicielle sur carte FPGA du télémètre

...

## 2. Servomoteur

Dans cette deuxième partie consacrée au servomoteur, nous avons choisi de convertir les positions angulaires du servomoteur en dixièmes de degré, soit sur 10 bits d'entrée, afin d'assurer une meilleure précision et d'anticiper un affichage plus détaillé des mesures sur l'écran VGA. Les valeurs supérieures à 900 dixièmes de degré ( soit 90° ) sont volontairement ramenées à cette limite pour respecter les contraintes physiques du servomoteur.

### 2.1. Implémentation en VHDL du servomoteur seul

Le fonctionnement du servomoteur repose sur un compteur incrémental dans une période de 20 ms, correspondant à 1 000 000 cycles d'horloge. Les valeurs de position transmises, exprimées en dixièmes de degré, déterminent la durée de l'impulsion haute (entre 1 ms pour 0° et 2 ms pour 90°). Par exemple, une position intermédiaire de 45° correspond à une impulsion de 1,5 ms, tandis qu'une précision au dixième de degré est assurée grâce à un incrément minimal de 55 cycles d'horloge par 0,1°. Par ailleurs, les états angulaires sont déterminés en fonction de la valeurs du compteur régulant les impulsions.

Figure 2.1.1 : Chronogramme du servomoteur seul.



Afin de valider ce modèle, nous avons élaboré un banc de test en simulation. La première partie des tests concerne les positions limites et demi du servomoteur :  $0^\circ$ ,  $45^\circ$  et  $90^\circ$ . Nous avons observé des impulsions respectives de 1 ms, 1,5 ms et 2 ms sur le chronogramme, en parfaite cohérence avec les calculs théoriques.

Ensuite, nous avons testé la gestion d'une valeur angulaire intermédiaire, comme  $32,4^\circ$ , dont l'impulsion correspond à 1,356 ms. Cette valeur a été vérifiée avec succès, notamment par l'absence de levée des drapeaux d'avertissement en simulation, le tout avec une précision au microseconde.

Enfin, nous avons réalisé des tests sur des scénarios particuliers. Un test avec le signal reset\_n a validé la remise à zéro de l'impulsion de commande lorsque le reset est activé. De même, les positions supérieures à 900 dixièmes de degré ont été correctement ramenées à une impulsion de 2 ms, confirmant la robustesse de notre implémentation.

## 2.2. Test unitaire sur carte FPGA du servomoteur seul

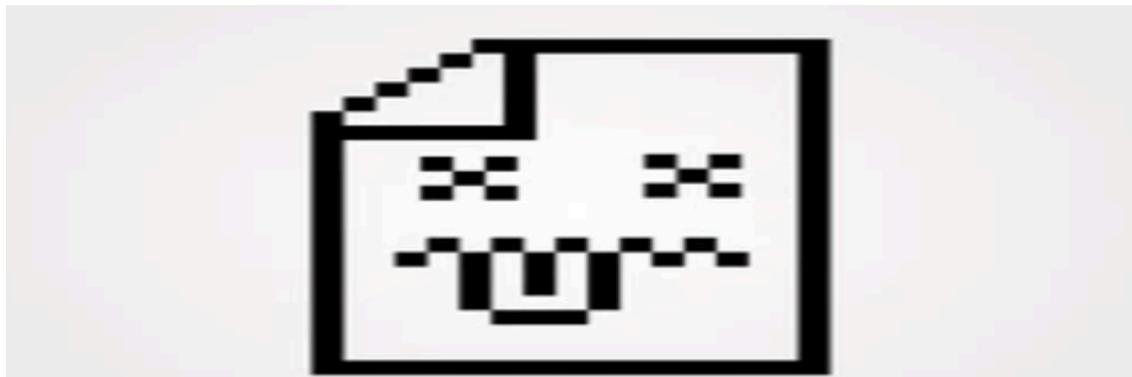
Pour valider le modèle matériel du servomoteur, des tests ont été réalisés sur carte FPGA DE10-Lite. Nous avons utilisé des interrupteurs pour simuler différentes valeurs de position angulaire. Les résultats observés sont présentés ci-dessous.

*Figure 2.2.1 : Test hardware du servomoteur pour un angle de  $90,0^\circ$*



Lors d'un premier test, nous avons réglé l'angle à  $90^\circ$ , correspondant à une impulsion haute de 2 ms. Ce résultat est en accord avec nos attentes et les simulations effectuées précédemment. Blabla en vérifiant le plafond ( tout les interrupteur allumé par ex ) + explication bit et position d'entrée souhaitée

Figure 2.2.2 : Test hardware du servomoteur pour un angle de 45,0°



Un deuxième test, avec une position de 45°, a révélé une impulsion haute de 1,5 ms, validant ainsi la gestion des positions intermédiaires Commentaire bref + switch

Figure 2.2.3 : Test hardware du servomoteur pour un angle de 0,0°



Ensuite, pour une position minimale de 0°, l'impulsion observée était bien de 1 ms, confirmant le bon fonctionnement pour l'angle zéro.

Figure 2.2.4 : Test hardware du servomoteur pour un angle de 32,4°



Enfin, une vérification à une valeur angulaire précise de 32,4° a démontré une

impulsion haute mesurée à 1,356 ms, validant la précision au dixième de degré sur la plateforme matérielle.

Les tests réalisés ont ainsi permis de confirmer que les positions limites et intermédiaires, ainsi que les cas particuliers, sont gérés conformément au modèle théorique.

### 2.3. Implémentation en VHDL du servomoteur compatible avec l'interface Avalon

Afin d'interconnecter le module servomoteur au reste du système de manière standardisée, nous avons intégré l'interface Avalon - MM. Cette intégration garantit une compatibilité avec les autres périphériques et permet une gestion centralisée des données par le bus Avalon. Pour cela, nous avons ajouté les signaux spécifiques à Avalon comme chipselect pour l'activation du module, write\_n pour l'écriture ( actif à l'état bas ) et WriteData pour transmettre les données mesurées.

*Figure 2.3.1 : Chronogramme du servomoteur compatible avec l'interface Avalon*

### 2.4. Programmation logicielle sur carte FPGA du servomoteur

## Conclusion

Bien que le projet n'ait pas été achevé et ait connu un démarrage sur les chapeaux de roues, avec un très long blocage au niveau de l'UAL et des problèmes personnels, celui-ci m'aura beaucoup apporté, notamment une grande fierté personnelle. J'espère également avoir su prouver, par la lecture de ce rapport, que j'ai compris les principaux aspects du module et ai fini par obtenir une organisation plus claire et structurée, tant de mon environnement de développement informatique que de mon code lui-même, en grande partie grâce au soutien et aux conseils de Thibault.

J'ai finalement trouvé une forme de libération et surtout une satisfaction personnelle qui me donne encore plus de motivation pour la suite de mon parcours universitaire et humain. Ce projet m'a permis de mieux comprendre la gestion des données dans le processeur, leur déplacement via les bus de données, ainsi que la signification concrète des signaux sortant des composants. J'ai également beaucoup apprécié en savoir plus sur les composants que l'on rencontre souvent sur les schémas électroniques de nos documentations techniques et mieux les comprendre.

Bien que certains n'aient pas cru en moi pendant longtemps, je reste convaincu que cela était principalement dû à un manque de communication. J'ai tenté de remédier à cela trop tardivement, je peux l'admettre, notamment en exposant mes difficultés par e-mail en raison de l'absence au niveau du stage. En outre si vous souhaitez en savoir davantage, ce dernier matin, j'avais un rendez-vous en matinée voir à l'annexe, ce qui explique pourquoi je n'étais pas présent. Par ailleurs, j'avais besoin d'une bonne préparation nonobstant mon innocence avérée ou non, ce qui, comme nous l'avons vu, s'est révélé insuffisant en raison du peu de temps accordé à celle- ci.

Pour conclure ce rapport, j'espère sincèrement, Monsieur DOUZE, que nos relations seront réellement apaisées à l'avenir. J'espère également que notre belle filière Ei, EiSE, Ei2I, ELi, SETi, MCR, GPi continuera de prospérer comme elle l'a toujours fait, grâce à Thibault HILAIRE, Dimitri GALAYKO, Yann DOUZE, Roselyne CHOTIN, Annick ALEXANDRE, Andrea PINNA, Sylvain FERUGLIO, Cécile BRAUNSTEIN, Benoît FABRE, et enfin Michel REDON, et que toutes les futures promotions s'y sentiront aussi épanouies que j'aurais pu l'être, tout comme l'a été notre grand frère Monsieur VIATEUR.

## Annexe