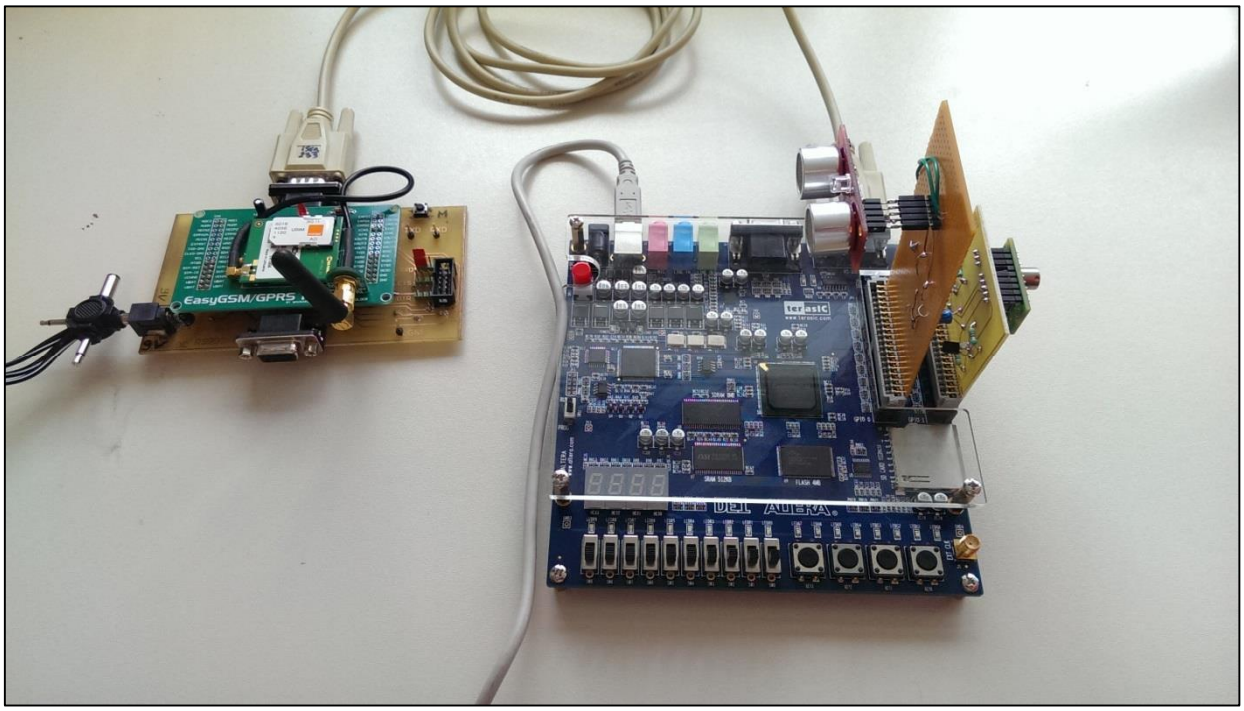
Institut Supérieur de lôÉlectronique et du Numérique
Tél. : +33 (0)2.98.03.84.00
Fax : +33 (0)2.98.03.84.10
20, rue Cuirassé Bretagne
CS 42807 - 29228 BREST Cedex 2 - **FRANCE**

Projet M1

Année scolaire 2013/2014

Réalisation d'un système de télésurveillance et d'alerte



Proposé par : LE LAY Chantal

RAOULT Corentin — Option : Systèmes embarqués
PERROT Gurvan — Option : Systèmes embarqués

Résumé

Les systèmes de télésurveillance sont de plus en plus répandus dans le monde actuel. Ce projet fait partie du domaine de la domotique. Il a pour objet : la sécurité à lôintérieur dôun habitat ou même dôun lieu public.

Ces dernières années, il a été intéressant de voir quôavec lôavancée des nouvelles technologies, il est désormais de plus en plus facile de tout contrôler à distance. Lôintitulé de ce projet est donc en parfaite adéquation avec le développement des technologies du moment.

Comment une intrusion peut-elle être détectée à lôintérieur dôun habitat ?

Pour répondre à cette problématique, il a été imposé dôutiliser le FPGA de la carte DE1. La plupart des capteurs qui sont utilisés, notamment pour des applications de télésurveillance, communiquent grâce à un protocole I2C. Pour ce faire, une IP a dû être créé car lôenvironnement sous lequel le travail a été réalisé (Quartus) ne contient pas l'IP de ce protocole de communication. Cette IP comprend le composant I2C et son interface de communication avec le processeur (Nios II) qui sera utilisée pour l'application.

De plus, pour être prévenu en cas dôintrusion et pouvoir agir rapidement, un module GSM a été relié à cette carte DE1 via une connexion UART.

Dans un premier temps l'architecture hardware du système a été créée puis le programme software a été développé.

Deux capteurs (ultrason et thermique) ont été connectés à la carte DE1 afin de comparer leurs caractéristiques et leurs performances. En ce qui concerne leur fonctionnement, lorsque le capteur ultrason détecte un changement important par rapport à la distance moyenne quôil mesure, un SMS est envoyé par lôintermédiaire du module GSM. Le second capteur (thermique) quant à lui, envoie un message dôintrusion si une variation brutale de température intervient.

Durant ce projet, un autre capteur infrarouge a également pu être étudié. Son système de communication diffère des deux capteurs ci-dessus. En effet, une seule broche permet de lire ce capteur. Une comparaison a ainsi pu être réalisée.

De plus, une autre technologie que celle du FPGA a été mise en place. Un Raspberry Pi a été utilisé afin de lire les données de ce capteur et contrôler lôenvoi dôun SMS.

Glossaire

Altera : Fabricant de composants reprogrammables (FPGA)
Avalon : Type de bus destiné à destiné à l'implémentation sur des FPGA
Bash : Bourne-Again Shell
CPU : Central Processing Unit
Eclipse : Logiciel rattaché à Quartus (création de la partie Software)
Flash : Type de mémoire utilisant des transistors
FPGA : Field Programmable Gate Array
GPIO : General Purpose Input / Output
GSM : Global System for Mobile Communications
Hardware : Architecture matérielle
I2C : Inter Integrated Circuit
IP : Intellectual Property
JTAG: Joint Test Action Group: bus servant à programmer le FGA ou le microcontrôleur
LED : Light Emitting Diode (DEL : Diode Electroluminescente)
LSB : Less Significant Bit
Modelsim : Logiciel permettant la simulation de programmes VHDL
MSB : Most Significant Bit
NMOS : Type de transistor
Offset: Valeur représentant une adresse de manière relative
PCB : Printed Circuit Board
Pin : Broche du FPGA
PIN (code) : Personal Identification Number
PIO : Programmed Input / Output
PIR: Passive InfraRed
Qsys : Logiciel rattaché à Quartus (création de la partie hardware)
Quartus : Environnement de développement
RAM : Random Access Memory
R / W : Read / Write
SIM : Subscriber Identity Module
SMS : Short Message Service
SoC : System on Chip
Soft-processor : CPU implanté sur un FPGA
Software : Instructions informatique
SoPC : System on Programmable Chip
SDRAM : Synchronous Dynamic Random Access Memory
SSH : Secure SHell
UART : Universal Asynchronous Receiver Transmitter
USB : Universal Serial Bus
VHDL : Very High Speed Integrated Circuit Hardware Description
Vim : Vi IMproved

Table des matières

1. Table des figures

2. Introduction

La télésurveillance consiste à observer un lieu à distance, grâce à lôutilisation de toutes sortes de capteurs ou caméras. Ce système a commencé à se développer à Londres dans les années 1980 suite à certains attentats. Les domaines dôapplications de ce procédé sont divers. On peut sôen servir pour la sécurité routière, celle des lieux publics (métro, gare, aéroportsé) ainsi que pour la protection de bâtiments dits « sensibles » tels quôune banque.

Cependant, même si dans la majorité des cas, les appareils utilisés sont des caméras de surveillance, différents types dôappareils peuvent être mis en place (capteur thermique, détecteur de mouvement, capteur de distance ultrasoné).

Durant le projet « réalisation dôun système de télésurveillance et dôalerte », plusieurs capteurs ont été utilisés (ultrason, infrarouge). Le système global consiste à relier les capteurs au FPGA via une liaison I2C et en cas dôintrusion dôenvoyer un SMS au propriétaire du lieu grâce à un module GSM relié au FPGA par une liaison UART. Côest le protocole de communication I2C qui a été développé car les plupart des capteurs utilisent ce bus pour transmettre leurs données.

De plus, certaines contraintes ont été posées en début de projet. En effet, le choix de la technologie FPGA était imposé (carte DE1 de chez Altera). LôISEN possédant déjà un capteur ultrason et un thermique utilisant ce système de communication. Il a donc fallu créer une IP I2C sous lôenvironnement de travail Quartus car le composant nôexistait pas dans le logiciel de création hardware Qsys. Seule lôutilisation de Quartus permet de travailler sur le FPGA de chez Altera.

Pour ce qui est du module GSM, celui-ci était déjà fourni par lôécole avec une UART comme moyen de communication.

Afin de faire fonctionner ce système de communication, un programme en langage C a, par la suite, été réalisé.

3. Gestion de projet

Afin de mener à bien ce projet, une organisation au sein du binôme a dû être mise en place. Après avoir réalisé le cahier des charges ensemble, les différentes tâches à accomplir ont été réparties au sein du groupe. Cette division des tâches permet dôavoir un meilleur rendement et dôêtre plus efficace.

De plus, pour avoir un suivi de projet et valider certaines étapes (cahier des charges, testsé), des réunions étaient régulièrement organisées avec le professeur référent (Mme Le Lay). Cela a permis de restructurer certaines parties quand cela était nécessaire, de valider les objectifs intermédiaires et de préciser ceux qui devaient encore être réalisés. Un rapport dôavancement du projet a été rendu un mois avant le rapport final afin de permettre au professeur encadrant de pouvoir analyser plus facilement lôavancement du travail. Les tâches réalisées et celles encore inaccomplies y figuraient notamment.

Par ailleurs, pour aider à la gestion de projet, certains outils ont été mis en place comme le diagramme de Gantt afin de suivre lôavancement de chacune des tâches et le retard qui pouvait être accumulé.

Organisation du travail à lôaide dôun diagramme de Gantt :

Nom de la tache	Durée estimée	Durée réel	Novembre 2013	Décembre 2013	Janvier 2014	Février 2014	Mars 2014	Avril 2014	Mai 2014
Analyse du système, cahier des charges	3 semaines	6 semaines							
Étude des capteurs	2 semaines	3 semaines							
Prise en main des logiciels	11 semaines	10 semaines							
Modélisation hardware du système de base	2 semaines	2semaines							
Implémentation du module GSM	3 semaines	2 semaines							
Création du PCB	2 semaines	2 semaines							
Modélisation bus I2C (IP et software)	6 semaines	9 semaines							
Intégration finale	5 semaines	4 semaines							
Développement du système de comparaison	7 semaines	3 semaines							
Préparation démo et rapport	3 semaines	4 semaines							

Figure 1 : Diagramme de Gantt

Dans lôensemble, la durée qui avait été estimée pour chaque objectif a été respectée. Aucun gros retard nôa été pris.

Concernant le matériel nécessaire à la bonne réalisation de ce projet, la commande dôun capteur PIR a été effectuée. Ceci afin de pouvoir analyser les avantages et les inconvénients de différents capteurs. Pour ne pas être dans lôurgence, cette commande a été faite plusieurs mois avant la date de fin de projet.

Un délai dôun mois a également été prévu pour la réalisation du rapport afin dôéviter tout stress et dôavoir un certain recul sur le travail réalisé avant de le finaliser.

Toute cette organisation et ces procédures de demande de matériels (oscilloscope, réalisation dôune carteé) étaient nécessaires à la bonne réalisation de ce projet.

Grandes phases du projet :

- Etude des besoins et des contraintes avec le client (professeur encadrant)
- Réalisation du cahier des charges
- Répartition du travail au sein du binôme
- Conception et test de la partie hardware
- Conception et test de la partie software
- Intégration des deux parties
- Evaluation des performances du système et des différents capteurs
- Réalisation du rapport final

4. Analyse fonctionnelle (cahier des charges)

Dans un premier temps, lôoutil « diagramme pieuvre » a pu être utilisé pour analyser les besoins et identifier les fonctions de service du produit. Ce diagramme met en évidence les relations entre les différents éléments du milieu environnant et le produit. Sa mise en î uvre consiste également à identifier les fonctions du produit par rapport aux éléments qui lui sont extérieurs.

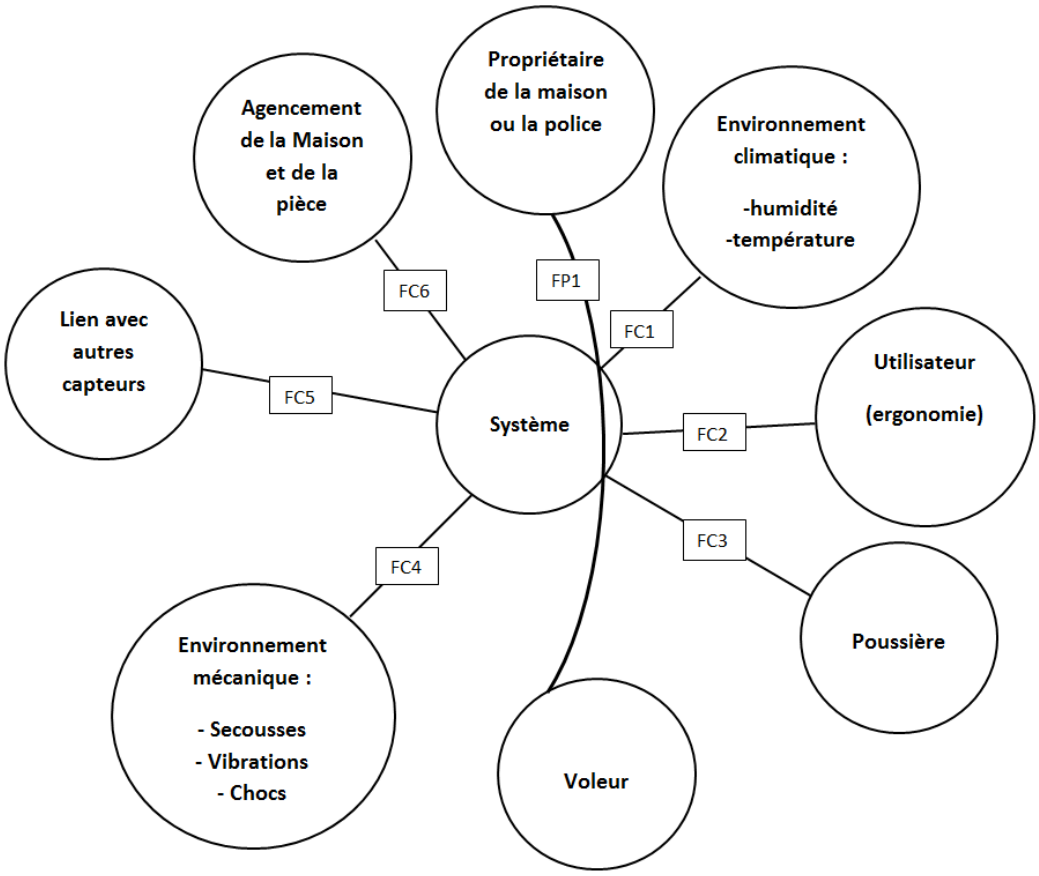


Figure 2 : Diagramme pieuvre

Fonction Principale :

FP1 : Le système doit avertir le propriétaire de la maison dôune intrusion par une personne non autorisée.

Fonctions de Contraintes :

FC1 : Le système doit être fonctionnel pour une température et une humidité dôune maison inhabitée (pas de chauffage).
FC2 : Le système doit pouvoir être utilisé et configuré facilement.
FC3 : Le système doit être protégé de la poussière.
FC4 : Le système doit résister aux contraintes physiques (lors du transport, de chutesé).
FC5 : Le système doit communiquer avec dôautres capteurs dans dôautres pièces.
FC6 : Le système doit avoir un champ de vision assez grand pour pouvoir détecter une personne nôimporte où dans une pièce.
FC7 : Le système doit être discret.

Pour ce qui est de la fonction principale du produit, celui-ci doit détecter toute intrusion au sein dûne pièce. Par ailleurs, pour les fonctions de contraintes, le capteur doit résister à lœenvironnement climato-mécanique intérieur (poussière, rayons du soleil). Ensuite, le produit doit être conforme à la réglementation ainsi quûaux normes de sécurité.

Les différents types de valeurs :

- Les valeurs dûusage : portée du capteur, discrétion (peu visible)
- Les valeurs dœestime : esthétique du produit, designé
- Les valeurs dœéchange : prix du système global

De plus, afin dœéviter tout désagrément au niveau du capteur, i.e. tous risques de chutes qui pourraient être dû à certaines secousses, on pourrait soit fixer la carte sur un socle vissé au mur ou sur un meuble.

Pour ce qui est de sa fiabilité et sa durée de vie, des tests peuvent être réalisés. Il faudrait que le capteur ait une durée de vie dûau minimum 7 ans et rester opérationnel 6 mois sans interruption.

Pour ce qui est de lœutilisation du système (mise en marche), il devra être facile pour une personne nœayant aucune compétence dans le domaine de la télésurveillance de sœen servir et éventuellement dœy apporter quelques modifications de paramétrage. Il doit être ergonomique et discret.

Définition du système

Pour définir le système, un autre outil de gestion peut être utilisé. Il sœagit du QQOQCP (Quoi, Qui, Où, Quand, Comment, Pourquoi).

Quoi ? : Un capteur permettant la détection dûune intrusion dans un habitat.
Qui ? : Le produit peut intéresser le particulier qui souhaite protéger sa maison tout comme les banques ou les villes afin de réduire les vols et les agressions.
Où ? : Le problème de sécurité peut apparaitre dans certains lieux ayant besoin dûun maximum de sécurité (banques, bijouteriesé).
Quand ? : Le capteur doit pouvoir être actif en permanence pour pouvoir garantir une sécurité totale du lieu.
Comment ? : Dans ce projet, un capteur thermique « enregistre » la température ambiante et dès quûune variation importante est détectée, un SMS est envoyé au propriétaire de la maison grâce à un module GSM. Le même principe est utilisé avec un capteur ultrason qui lit cette fois-ci la distance.
Pourquoi ? : A grande échelle (avec plusieurs capteurs répartis dans une maison), cette télésurveillance permet à une personne possédant beaucoup de bien de valeur de quitter son habitat sereinement.

Système de base

Voici un exemple de plan montrant le champ dôaction des capteurs.

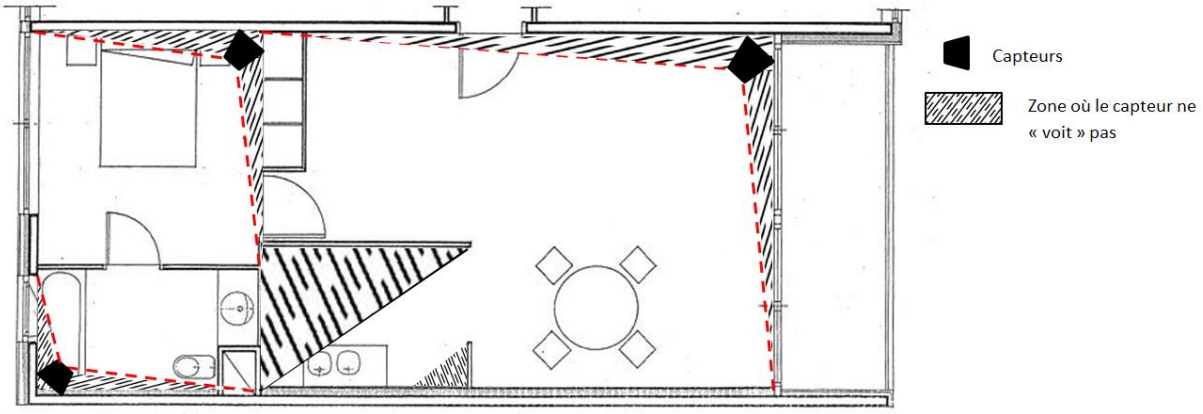


Figure 3 : Exemple de mise en action de capteurs

Sur ce plan de studio, les zones non « surveillées » par le capteur (ici en hachuré) sont négligeables par rapport à la surface totale de la pièce. De toute évidence, les espaces où les objets de valeur sont entreposés seront obligatoirement surveillés par le capteur.

Capteurs

Tous les capteurs ici placés dans un coin de chaque pièce, nécessitent seulement un angle de 90° pour être efficaces dans toutes les situations dôintrusions. Le champ dôaction du capteur peut être modélisé sous la forme dôun cône (cf schéma ci-dessous). Ainsi, même certaines pièces qui possèdent des velux peuvent être sécurisées. Le capteur ne bouge pas, il est fixe mais couvre un champ de plus de 100° ce qui est largement suffisant sôil est positionné dans un coin. Dans tout type de pièce une intrusion sera immédiatement détectée.

De plus, avec une portée avoisinant les 10 m, chaque capteur peut couvrir une pièce entière, mais lôagencement des pièces du client nôétant pas connu par les concepteurs du système, il ne peut être garanti quôil nôy ait aucun angle mort dans la pièce où le capteur sera installé. Dans des pièces moins large telles que des couloirs, le capteur détectera plus aisément un individu. L'installateur du système devra donc veiller à positionner les capteurs aux bons endroits en fonction de la configuration de la pièce et de son agencement. Pour une meilleure détection, ils pourront être placés à hauteur humaine quand cela sera possible.

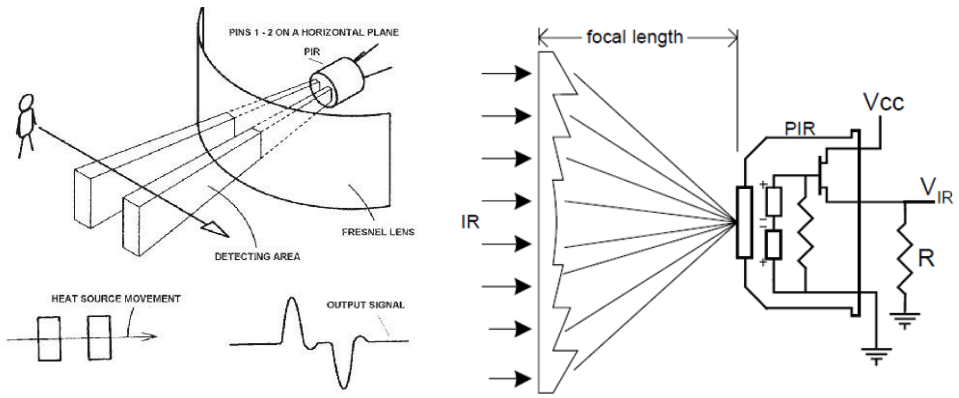


Figure 4 : Schéma de lôangle dôaction dôun capteur

Module GSM

Afin que le propriétaire ou la police soit averti de façon rapide un module GSM sera utilisé pour envoyer un SMS dès quôune intrusion est détectée. Ce dernier est fourni par lôISEN. Celui-ci peut communiquer avec la carte DE1 et le Nios II grâce à différents types de bus. La liaison UART a été choisie pour ce transfert de données car lôIP de ce protocole de communication est disponible dans lôenvironnement Quartus.

5. Développement technique

5.1. Fonctionnement des Systems on Chip (SoC)

Exemple dôun SoC simple qui utilise des blocs RAM du FPGA pour sauvegarder le programme du Nios II.

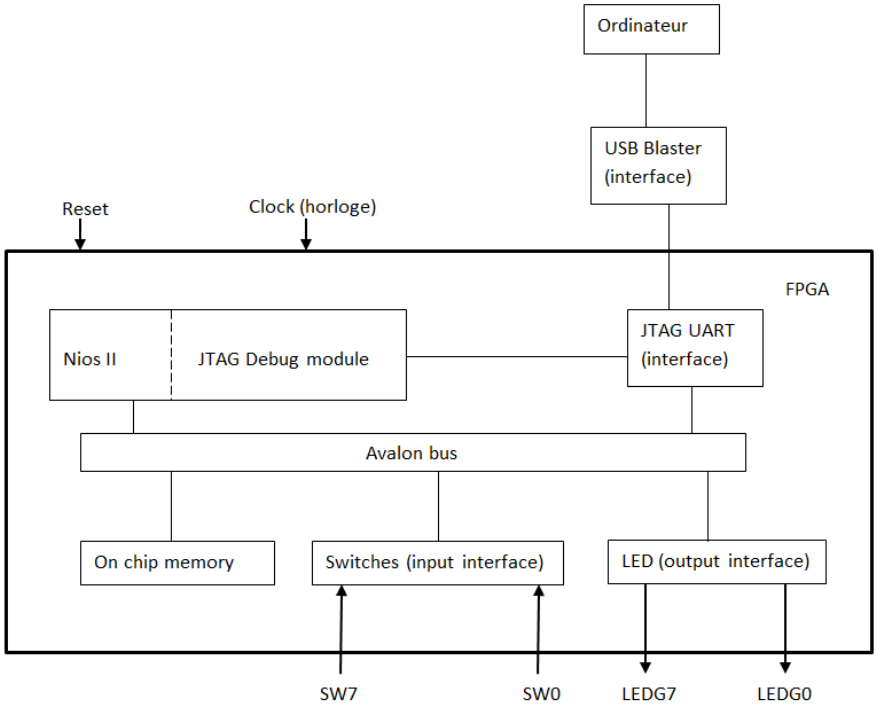


Figure 5 : Exemple dôarchitecture dôun SoC

Dans un premier temps, il faut savoir ce quôest un System on Chip. Côest un système qui intègre un processeur, de la mémoire, des périphériques I/O et des composants spécifiques sur une même puce.

On peut également définir ce quôest un SoPC (System on Programmable Chip). Côest un système complet embarqué sur une puce reprogrammable de type FPGA.
Avant de construire un système embarqué implanté sur un FPGA, il faut avant tout, analyser les besoins pour définir les composants qui constitueront le système. Puis il faut construire lôarchitecture hardware et enfin développer le programme software.
A partir de lôenvironnement de développement Quartus développé par Altera, on peut développer lôarchitecture matérielle grâce à lôoutil Qsys et la partie software avec Eclipse (logiciels intégrés à Quartus).

Le processeur qui a été utilisé durant le projet est le Nios II de chez Altera. Celui-ci est connecté aux différents périphériques via le bus Avalon, on peut ainsi utiliser les connecteurs GPIO, la mémoire on Chip, la connexion JTAGé

La carte DE1 dispose de différents périphériques de sortie. Durant la construction de la partie hardware, on peut se servir des LEDs, des boutons, des switchsé de la carte pour les applications que lôon souhaite. Toute cette partie est modulable et offre de nombreuses possibilités.

Différences entre lôimplémentation dôun programme VHDL sur le FPGA et un SoPC :

A partir dôun composant décrit en VHDL, une IP est créée afin que le Nios II puisse communiquer avec le composant et « le contrôler ». Pour ce faire, des registres accessibles en lecture ou écriture doivent être au préalable définis.

Pour créer lôIP, un fichier VHDL de décodage dôadresse est relié au composant. A chaque périphérique est associée une plage dôadresse pour accéder aux différents registres (offset) de lôIP.

Toujours sous Qsys, un système a été réalisé pour contrôler les différents éléments de la carte DE1 utiles au projet:

- les PIO pour les LEDs et les afficheurs 7 segments
- le contrôleur pour la SDRAM
- le CPU, le system ID et le timer
- lôIP I2C relié au port GPIO
- lôhorloge de base reliée à lôoscillateur (50 MHz)
- le contrôleur UART relié à la connexion RS-232
- é

Ensuite, les deux fichiers ont été compilés sous Qsys afin de créer le périphérique. En fonction des besoins, certains éléments ont peuvent être rajoutés comme le switch pour choisir le capteur.

Une fois cette étape réalisée, la partie hardware terminée a pu être compilé sous Quartus. Lôarchitecture matérielle crée et implantée dans le FPGA est reliée aux différentes broches des périphériques utiles sur la carte (SDRAM, afficheurs 7 segments, LEDsé).

Suite à cela, on passe sous Eclipse pour développer la partie software en langage C.

Synthèse du flot de conception dôun SoPC (Nios II system) :

- Analyse des besoins en termes dôarchitecture matérielle
- Création dôun projet via Quartus
- Création de lôarchitecture matérielle via Qsys
- Intégration du système Qsys dans Quartus, ajout si nécessaire de blocs logiques complémentaires, affectation des pins, synthèse, placement routage, génération fichier.sof de configuration.
- Configuration du FPGA (télécharger lôarchitecture hardware dans le FPGA) avec le fichier de configuration.

- Création du programme software (exécuté par le NIOS) en utilisant « software Build Tools for Eclipse » (utilisation du fichier .sopcinfo qui définit la plateforme hardware pour les outils software).
- Compilation, édition de lien.
- Chargement du code exécutable (projet .elf)
- Vérification du fonctionnement sur la carte.

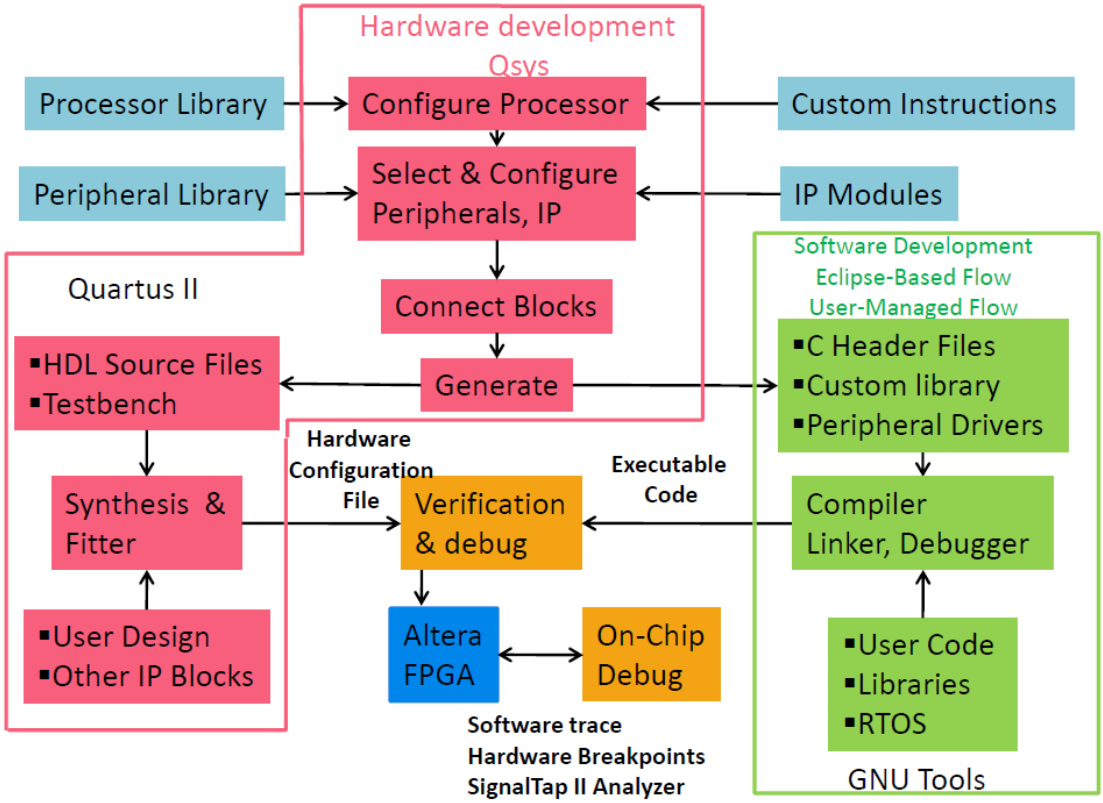


Figure 6 : Flot de conception pour le Nios II

5.2. Liaison I2C entre le capteur et le FPGA (carte DE1)

Suite à la réalisation du cahier des charges, notre première tâche consistait à « fabriquer » le périphérique I2C. Ce dernier comprend deux fichiers VHDL, le premier gère la communication I2C, le second sert à interfacer le composant avec le Nios II via le bus Avalon.

5.2.1. Présentation générale du protocole I2C

Afin dôavoir une meilleur compréhension de cette partie, voici une présentation du protocole I2C.

Ce dernier est un bus de données série synchrone. Cela veut dire que la transmission de données se passe seulement sur un fil. De plus le fait que ce protocole soit synchrone oblige à utiliser un second fil pour la transmission de lôhorloge entre les différents équipements. Enfin lôI2C est un bus bidirectionnel half-duplex, cela signifie quôil permet de transporter des informations dans les deux sens, mais pas simultanément.

Plusieurs équipements, soit maîtres, soit esclaves, peuvent être connectés au bus. Les échanges ont toujours lieu entre un seul maître et un (ou tous les) esclave(s), toujours à l'initiative du maître (jamais de maître à maître ou d'esclave à esclave). Cependant, rien n'empêche à un composant de passer du statut de maître à esclave et réciproquement.

La connexion est réalisée par l'intermédiaire de 2 lignes :

- SDA (Serial Data Line) : ligne de transmission données bidirectionnelle,
- SCL (Serial Clock Line) : ligne d'horloge de synchronisation bidirectionnelle.

Il ne faut également pas oublier la masse qui doit être commune aux équipements. Les deux lignes sont maintenues au niveau haut grâce à des résistances de pull-up.

Le type de codage utilisé est de type NRZ (Non Retour à Zéro) : c'est-à-dire que l'état logique (« 1 » ou « 0 ») est maintenu au moins sur toute la période de l'horloge.

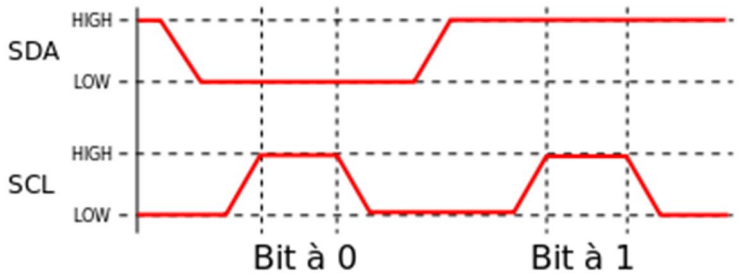


Figure 7 : Exemple de transmission de bits sur un bus I2C

Le niveau (haut ou bas) de la ligne SDA doit être maintenu stable pendant le niveau haut sur la ligne SCL pour la lecture du bit.

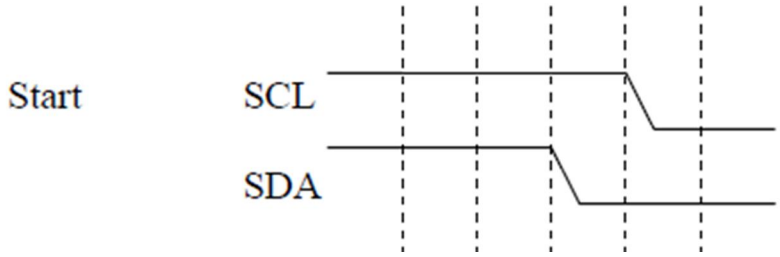
Les équipements sont donc câblés sur le bus par le principe du « ET câblé », ce qui veut dire qu'en cas d'émission simultanée de 2 équipements, la valeur « 0 » écrase la valeur « 1 ».

Ainsi on peut dire que l'état logique « 0 » ou bas est l'état « dominant », et que l'état logique « 1 » ou haut est l'état « récessif ». Lorsque le bus n'est pas utilisé, il est au niveau haut (grâce aux résistances de pull-up).

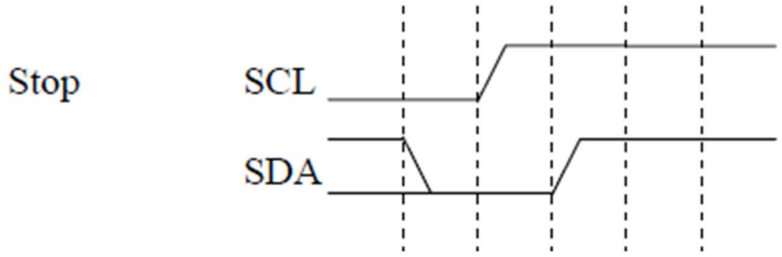
Il existe 5 vitesses de transmission pour l'I2C :

- « Standard mode » : 100 kbit/s,
- « Fast mode » : 400 kbit/s,
- « Fast plus mode » : 1 Mbit/s,
- « High-speed mode » : 3,4 Mbit/s,
- « Ultra-fast mode » : 5 Mbit/s.

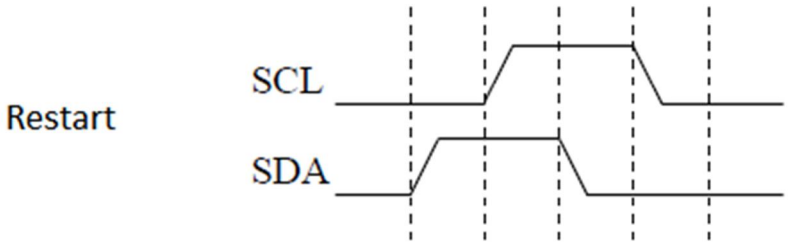
Trame I2C :



Un maitre peut initier un transfert en envoyant une séquence Start qui permet de lancer lôécriture de lôadresse sur la ligne SDA. Les lignes des signaux SCL et SDA sont à lôétat haut par défaut lorsquôelles sont inutilisées. Cette séquence met dôabord la ligne SDA à zéro puis la ligne SCL.



A lôinverse de la séquence Start, la séquence Stop a pour but de remettre les lignes à lôétat haut et ainsi de mettre fin à la communication. La ligne SCL est dôabord remise à ó1ô puis la ligne SDA. Le maitre génère cette séquence pour mettre fin à la communication.



La séquence de Restart intervient lorsque lôon passe dôune lecture à une écriture ou lôinverse. Celle-ci est également utilisée lors dôun changement dôadresse de lôéquipement I2C. Cette séquence correspond à un Stop puis un Start, à la différence que pour le Stop, cêest la ligne SDA qui est remise à lôétat haut avant la ligne SCL.

Protocole entier de la trame I2C :

Les échanges de données entre le maitre et lôesclave sont synchronisés par le signal dôhorloge SCL. Ils se font octet par octet.

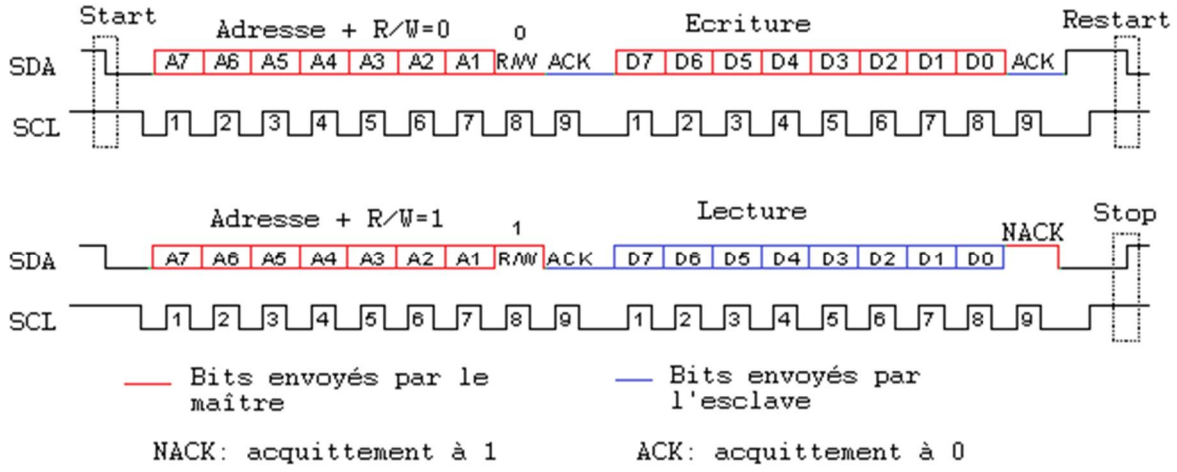


Figure 8 : Exemple de trames I2C

La procédure est lancée par un Start. Ensuite pour pouvoir faire la liaison avec le composant I2C, lôadresse correspondant à celui-ci est envoyée par le maitre sur la ligne SDA.

Cette adresse comporte 7 bits plus un bit (« R/W ») permettant de savoir si on se situe en mode écriture (bit à ó0ô) ou en mode lecture (bit à ó1ô) (8 bits en tout). Afin de savoir, si le capteur a bien reçu la première trame et que la communication est bien établie, le capteur doit envoyer au maitre un acquittement. Un acquittement correspond au niveau ó0ô sur le neuvième font dôhorloge (après un octet de données).

Ensuite, le maitre envoi les bits de données du MSB au LSB (trame dôécriture) puis lôesclave fait un acquittement pour informer le maitre de la bonne réception de lôinformation.

Une séquence de Restart est alors lancée (bit « R/W » mis à ó1ô) pour passer en mode lecture. Une nouvelle fois lôadresse plus le dernier bit de sélection du mode est acquittée. Pour lôenvoi des données, côest désormais lôesclave qui envoi les différents bits au maitre, toujours du MSB au LSB. Un acquittement du maitre a lieu à la fin de chaque octet reçu.

La lecture de données peut être effectuée tant que lôon nôa pas de bit de non acquittement qui y mettrait fin. Naturellement, côest le signal de Stop qui met fin à tout ce protocole.

Mécanisme de synchronisation de lôhorloge :

A tout moment, l'esclave peut « bloquer » la ligne SCL au niveau bas pour signaler qu'il est occupé (ralentissement de la communication). En effet, le niveau bas étant dominant, lorsque le maître fournit un niveau haut sur la ligne SCL, celle-ci est donc bloquée à lôétat bas. Le maître doit cependant continuer à fournir un niveau haut afin que celui-ci soit pris en compte une fois la ligne libérée par lôesclave. Ainsi, le cycle peut reprendre.

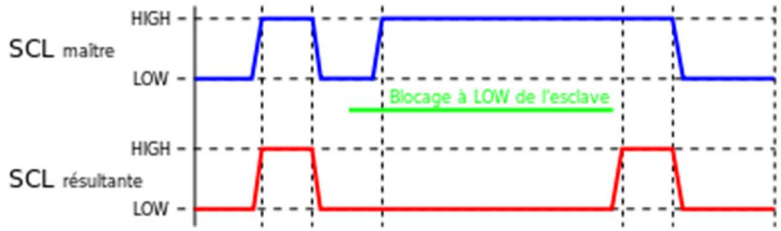


Figure 9 : Exemple dôun blocage du SCL

5.2.2. Modélisation en VHDL

Dans un premier temps, lôexplication portera sur le fichier VHDL gérant seulement le protocole I2C.

La première tâche à réaliser est celle de diviser la fréquence générée par lôoscillateur présent sur la carte DE1 (50 MHz). Ceci afin dôobtenir une des fréquences standards pour la vitesse correspondant au bus I2C. Dans ce projet, la vitesse de transfert choisie est de 400 kbit/s (ce qui équivaut à 400 kHz: « Fast mode »).

Chaque période du bus I2C est elle-même divisée en quatre sous-périodes afin de générer un déphasage entre lôhorloge du bus de données (data_clk) et celle de la ligne SCL (scl_clk). Par la suite, les données de la ligne SDA ne pourront changer que sur les fronts montants de data_clk. En revanche la ligne SCL sera identique au signal scl_clk lors dôune transmission de données. Grâce à ce diviseur, les bits de données sont maintenus pendant toute la durée de la période de la fréquence du bus. La ligne SCL est quant à elle à lôétat haut au milieu de cette période. Ceci permet dôavoir une marge de temps pour prélever lôétat des bits de données.

De plus une vérification de lôétat de la ligne SCL est faite lorsque scl_clk passe à lôétat haut. Ainsi si lôesclave impose une pause (SCL à lôétat bas), le maître le voit et maintient son état jusquôà ce que la ligne soit libérée.

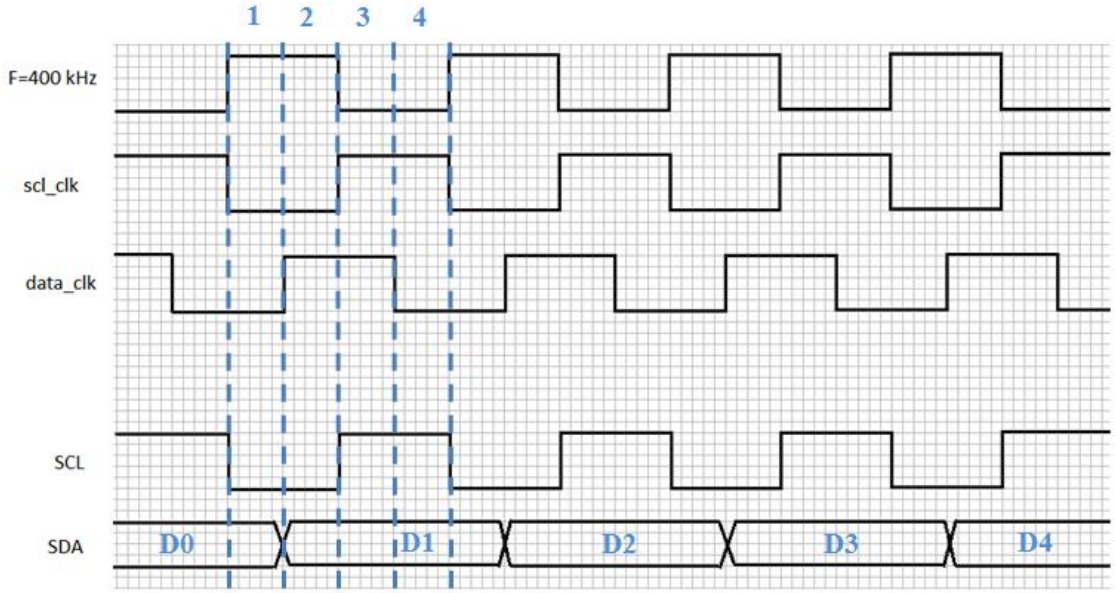


Figure 10 : Chronogramme correspondant aux différents signaux dôhorloge

Diagramme d'états pour l'I2C :

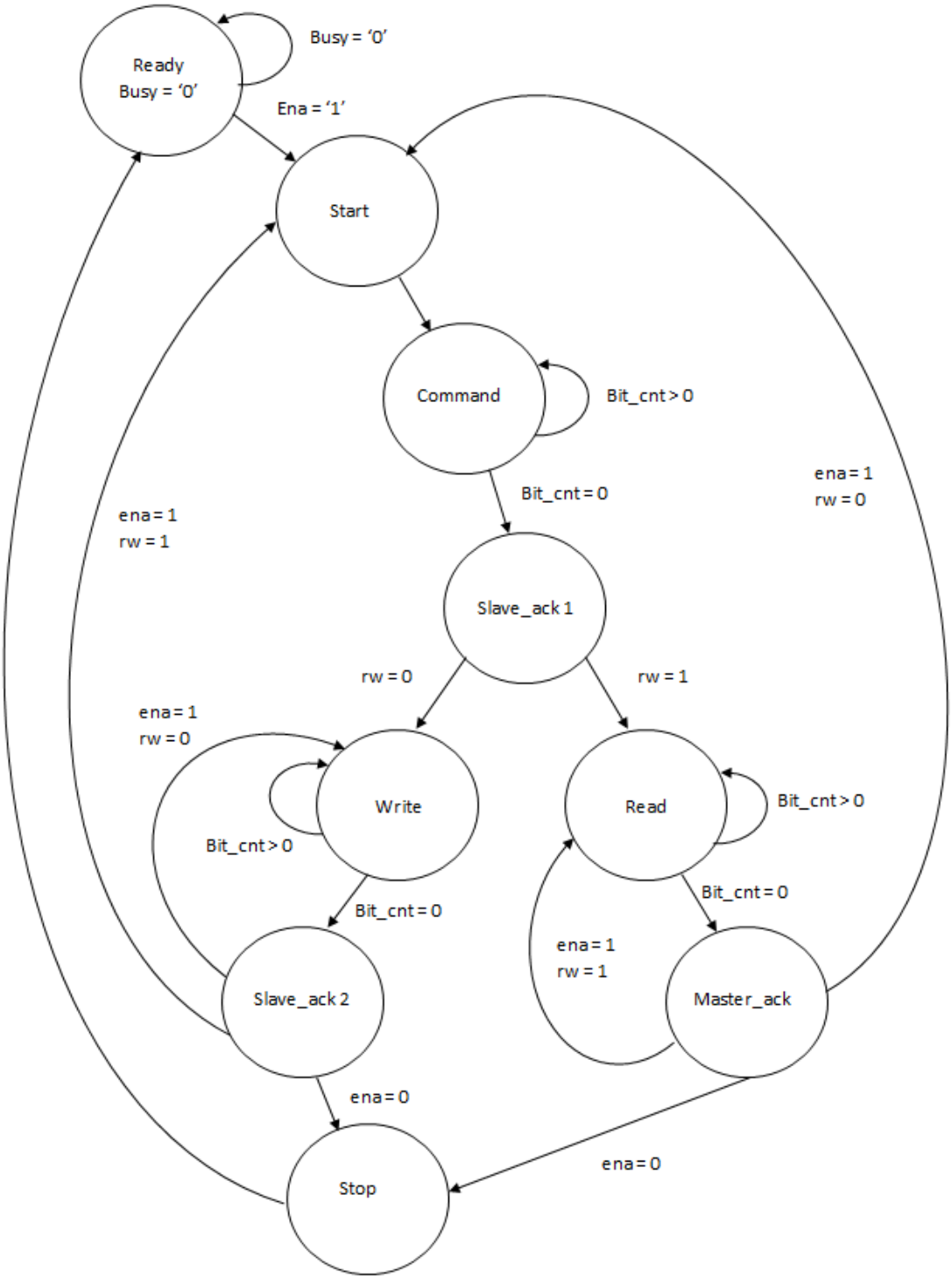


Figure 11 : Diagramme d'états pour le composant I2C

Ce diagramme comportant 9 états va permettre d'expliquer le protocole I2C avec les conditions de passage d'un état à un autre. Le premier état « ready » permet une initialisation de toutes les variables ainsi que la libération des lignes (busy = '0'). Une fois que cela est fait, le système peut être lancé grâce au Start. L'étape « command » vient alors avec l'octet pour l'adresse du composant afin de communiquer avec celui-ci. Si la communication est bien réalisée avec le composant, un acquittement est fait (état 'ack' sur le chronogramme).

A partir de cet état et en fonction du mode dôutilisation du capteur, un mode lecture ou un mode écriture peut être effectué. Cela dépendra du bit « R/W » (ó0ô pour lôécriture et ó1ô pour la lecture). On doit alors avoir un acquittement, validation de bonne réception de la commande par le capteur. Suite à cela, soit une boucle permet de rester dans le mode lecture ou écriture, soit le système redémarre depuis lôétat Start ou on stoppe toute acquisition de mesures.

5.2.3. Interface avec le Nios II (bus Avalon)

Pour permettre au composant I2C de communiquer avec le Nios II, un second fichier VHDL a été créé. Ce dernier est appelé IP_I2C et a pour simple but de faire fonctionner le fichier responsable de la communication I2C à partir du microprocesseur. Pour se faire, certains registres ont été mis en place et défini afin que le Nios II puisse commander le composant.

Offset	Name	R/W	Contenu des 8 bits			
0	addr_reg	W	Adresse du périphérique capteur / esclave			
1	data_wr_reg	W	Données à écrire à lôesclave			
2	control_reg	RW	ack_error (bit 3)	ena (bit 2)	ena_irq (bit 1)	fin_impulse (bit 0)
3	data_rd_reg	R	Octets reçus par lôesclave			
4	rw_reg	RW	Sur un seul bit : choix lecture ou écriture (bit 0)			

Le premier registre (offset 0) permet dôécrire lôadresse sur 7 bits. Un second (offset 4) permet la modification et lôécriture du bit « R/W » qui permet de compléter les 7 bits de lôadresse pour en avoir 8. Un autre registre (offset 1) permet dôécriture la donnée que lôon souhaite envoyer sur 8 bits.

Il y a enfin deux derniers registres. Un registre de contrôle (offset 2) qui permet de lancer ou dôarrêter la trame grâce à lôécriture de ó0ô sur le bit ena (Start ou masque pour le Stop). Il a également pour rôle de vérifier que les acquittements ont bien été effectués. Seulement quatre bits sont nécessaires pour gérer la communication entre le système (maitre) et le capteur I2C (esclave).

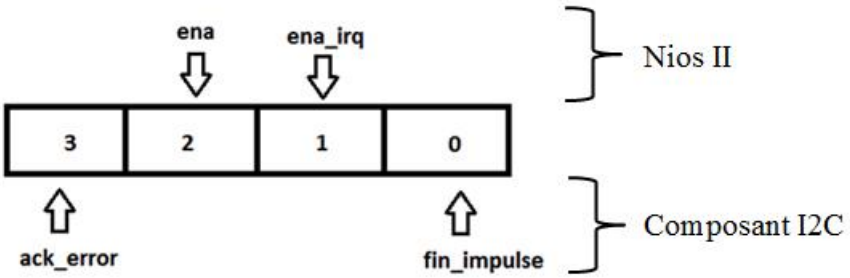


Figure 12 : Registre de contrôle

Sur ce schéma, le composant I2C peut écrire sur les bits 0 et 3 du registre de contrôle tandis que le Nios II peut écrire sur les bits 1 et 2 (bien que le bit 1 « ena_irq » ne soit pas utilisé). Tous ces registres peuvent cependant être lus par le Nios II.

Définition du registre contrôle :

- ack_error =1 → pas d'acquittement
- ena = 1 → autorisation de communication
- fin_impulse → mis à 1 par l'IP lors de l'acquittement et remis à zéro par le Nios II
- ena_irq → mis à 1 par le Nios II

Pour finir, un dernier registre (offset 3) permet de récupérer la donnée qui a été lue.

Il y a autant d'adresses que de registres contenus dans le périphérique. En effet, pour communiquer avec le capteur des registres sont nécessaires. Ce sont les offsets de ces registres qui sont utilisés dans la partie software. Une plage d'adresse est donc réservée en fonction du nombre de registres qui sont nécessaires à son fonctionnement. L'adresse et le nom des registres sont définis au préalable lors de la conception hardware.

Exemples d'utilisation des registres l'IP I2C:

Ces fonctions sont la base pour utiliser les registres de l'IP.

Ecriture de l'adresse: 0x70 dans le registre 0 :

```
IOWR(IP_I2C_0_BASE, 0, 0x70);
```

Lecture du registre 3 qui est stocké dans une variable sur 8 bits :

```
alt_u8 valeur_lue = (IORD(IP_I2C_0_BASE,3));
```

Ecriture de 4 (0100) dans le registre de contrôle → Start :

```
IOWR(IP_I2C_0_BASE, 2, 4);
```

Masque pour le Stop (ena (bit 2) = 0) → Stop :

```
IOWR(IP_I2C_0_BASE, 2, IORD(IP_I2C_0_BASE,2) & 4);
```

Afin de faciliter l'envoi et la réception de données via le bus I2C des fonctions plus simples ont été créées.

```
void I2C_write_address(alt_u32 i2c_base, alt_u8 address);
```

Cette fonction met en mémoire dans le registre de l'IP l'adresse sur 7 bits ainsi que le bit « R/W ». Pour cela elle prend l'adresse donnée sur 8 bits en argument et la scinde en 2 : la vraie adresse sur 7 bits et le bit « R/W ». Exemple : si on passe 209 (D1 en hexadécimal) en argument, cela donnera une adresse sur 7 bits égale à 104 (68 en hexadécimal) et « R/W » = 1. Par sécurité cette fonction doit toujours précéder une des 3 fonctions ci-dessous.

```
void I2C_write_data(alt_u32 i2c_base, alt_u8 data);
```

Cette fonction met en mémoire l'octet data dans le registre et lance l'écriture de l'adresse mise en mémoire par la fonction I2C_write_address puis de la donnée data sur la ligne SDA.

void I2C_write_register_slave(alt_u32 i2c_base, alt_u8 num_register, alt_u8 data);

Fonction identique à la précédente à la différence que cette dernière écrit 2 données sur la ligne : le numéro du registre et l'octet de data. Cela sert à écrire une donnée dans un registre de l'équipement I2C esclave.

alt_u8 I2C_read_data(alt_u32 i2c_base);

Fonction permettant de lancer une lecture, c'est-à-dire qu'elle écrit l'adresse (cf I2C_write_address) puis lit et retourne en sortie de fonction l'octet que lui envoie l'esclave.

5.3. Utilisation des différents capteurs

5.3.1. Capteur ultrason (SRF 08)

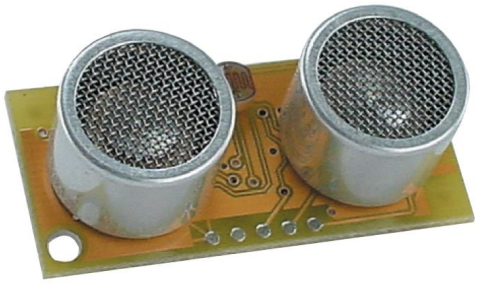


Figure 13 : Capteur ultrason

Ce capteur ultrason permet de mesurer une distance de 3cm à 6m. Il est alimenté en 5V et il fonctionne grâce à une connexion I2C. Son angle de détection est de 55°. Ce type de capteur permet également de mesurer l'intensité lumineuse. Pour mesurer la distance, une onde est envoyée, cette dernière réfléchie sur le premier obstacle qu'elle rencontre. Une fois l'onde revenue, la distance peut être obtenue dans l'unité de mesure que l'on souhaite (en cm, en pouce ou en microseconde).

Pour utiliser ce capteur, on doit d'abord écrire son adresse afin de pouvoir communiquer avec lui. Son adresse étant E0 (pour l'écriture) et E1 (pour la lecture) en hexadécimal sur 8 bits. Pour obtenir l'adresse sur 7 bits (norme I2C), on enlève le bit de poids faible qui correspond au bit « R/W ». On obtient alors une adresse sur 7 bits de 70 (en hexadécimal).

Ensuite pour initialiser une demande de donnée il faut écrire la commande qui correspond à l'unité de mesure qui est souhaité pour le résultat (cf tableaux à la fin de cette partie).

Dans ce projet, il faut écrire 81 dans le registre zéro, afin de récupérer la donnée lue en centimètres.

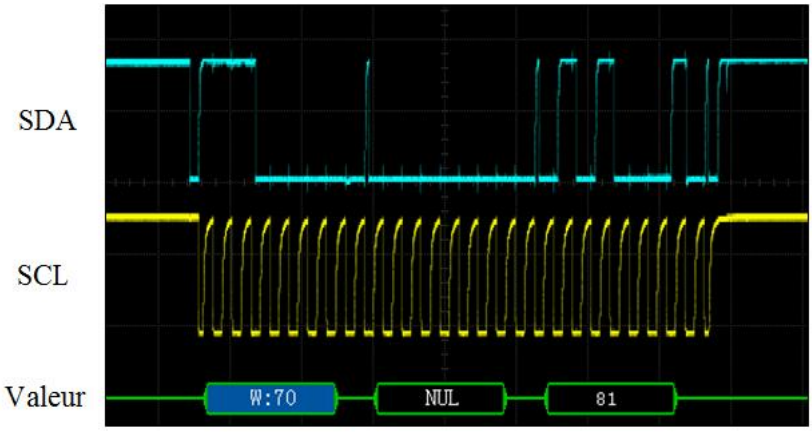


Figure 14 : 1ère trame envoyée pour lancer une prise de mesure

Avant de pouvoir lire la donnée dans le registre deux et trois (donnée sur 16 bits, un octet dans chaque registre), il faut au préalable attendre que lôonde ultrason ait eu le temps dêtre émise et de revenir. La documentation indique que ce délai est de 65 ms. Après ce temps dôattente la donnée peut être lue. Pour cela on écrit lôadresse du capteur avec le bit « R/W » égale à ó0ô et on écrit la valeur 2 pour signifier au capteur que lôon veut récupérer la donnée qui se trouve dans son deuxième registre. Ce dernier correspond aux 8 bits de poids fort de la distance lue par le capteur. Enfin, un Restart est effectué. Lôadresse est donc réécrite mais avec cette fois avec le bit « R/W » à ó1ô. Le maître libère alors la ligne pour permettre au capteur dôécrire la donnée mesurée.

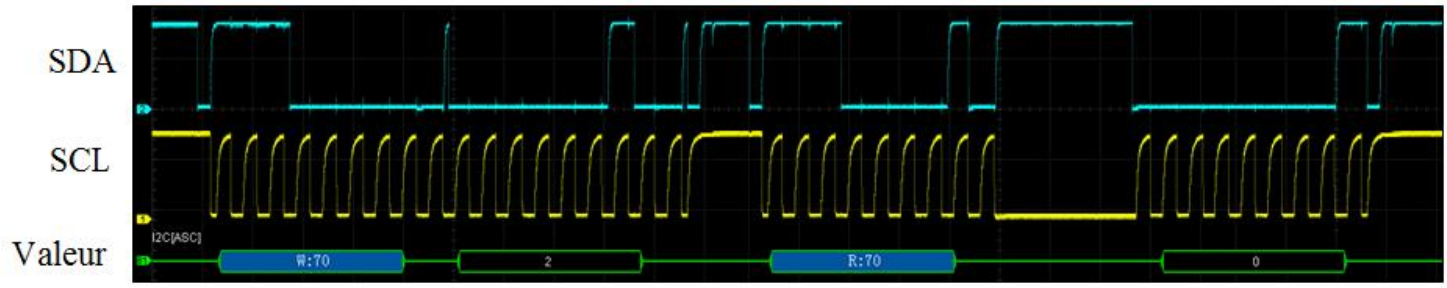


Figure 15 : Trame de lecture pour le registre 2

Les trames sont utilisées pour lire lôoctet de poids faible, avec cette fois-ci une lecture du registre 3.

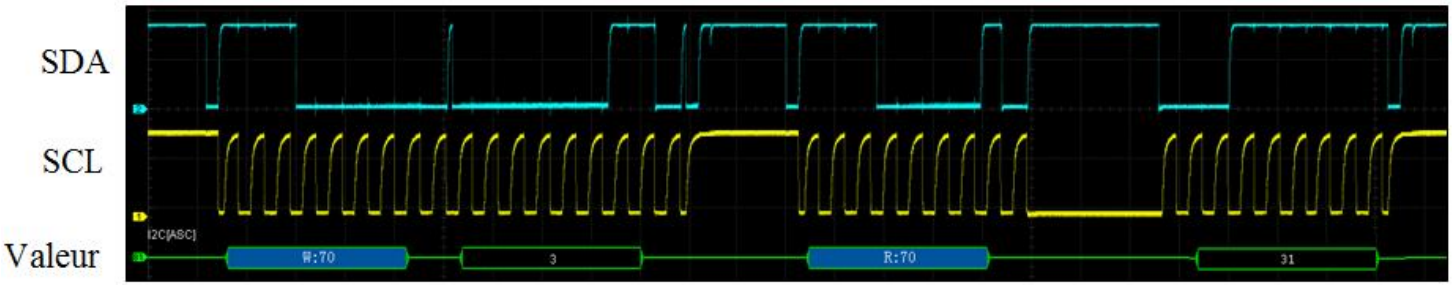


Figure 16 : Trame de lecture pour le registre 3

Le capteur ayant une précision approximative lorsquôil mesure de grandes distances, une moyenne est fait pour avoir une valeur fiable de la distance entre le capteur et le plus proche objet. Ensuite une mesure est prise tous les dixièmes de secondes et comparée à cette moyenne. Un seuil de 30 cm a été choisi car la précision du capteur peut varier jusquôà 20 cm.

Ainsi, si la différence est de plus de 30 cm alors on considère quûne personne est passée entre le capteur et lôobjet : il y a donc une intrusion.

Les valeurs sont prises à une fréquence de 10 Hz pour que la détection se fasse même si la personne passe rapidement devant le capteur.

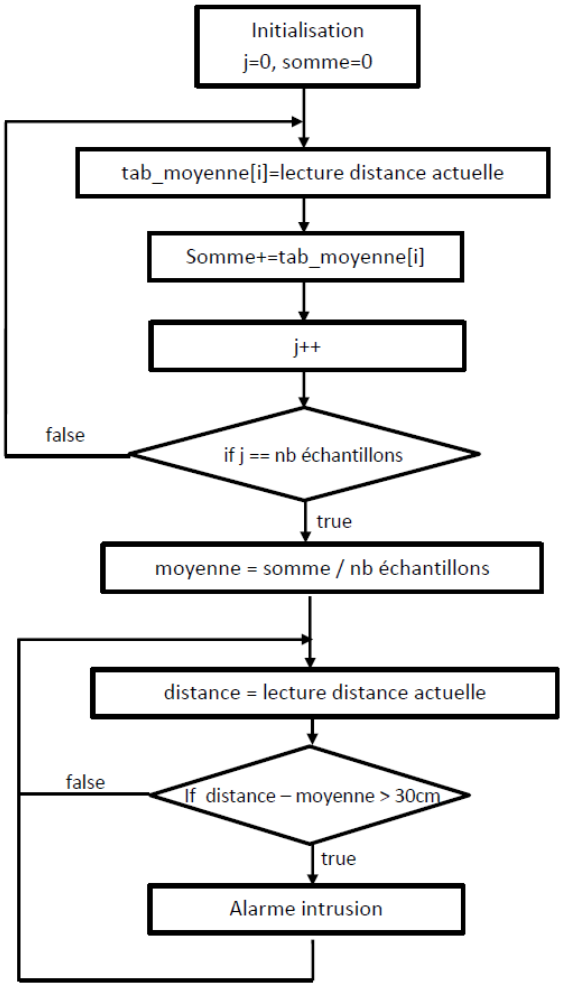


Figure 17 : Organigramme de détection par mesure de la distance

Spécifications des registres du capteur ultrason :

Adresse	Lecture	Écriture
0	N° de la version du logiciel	Registre de commande
1	Capteur de lumière	Registre gain max (3 par défaut)
2	1er écho (octet haut)	Registre portée (255 par défaut)
3	1er écho (octet bas)	N/A

Figure 18 : Registres du capteur à ultrason

Commande		Action
Décimal	Hexadécimal	
80	0x50	Mode de mesure avec le résultat en inches
81	0x51	Mode de mesure avec le résultat en centimètres
82	0x52	Mode de mesure avec le résultat en microsecondes

Figure 19 : Différentes possibilités dûunités en lecture de la donnée

5.3.2. Capteur thermique (module MTP81)

Figure 20 : Module MTP81

Ce module est un capteur thermique pouvant s'apparenter à une mini-caméra thermique basse résolution. Ce capteur est à même d'effectuer des mesures directes de température.

Le module est doté de 8 mini-zones sensibles consécutives capables de mesurer la température d'un point donné. Alimenté sous 5V, le capteur se pilote au moyen d'un bus I2C. Une sortie spéciale permet de piloter automatiquement un servomoteur afin que le module puisse balayer une zone complète pour obtenir le spectre thermique d'une large surface.

Concernant son utilisation, la première donnée qui doit être écrite correspond à l'adresse du capteur avec le bit « R/W » égale à zéro. Dans ce cas, l'adresse sur 8 bit est égale à D0 en hexadécimale. De la même manière que précédemment on obtient une adresse sur 7 bits de 68 en hexadécimal. La seconde trame à envoyer est la sélection de l'adresse du premier registre à lire du capteur. Cette dernière est égale à 0.

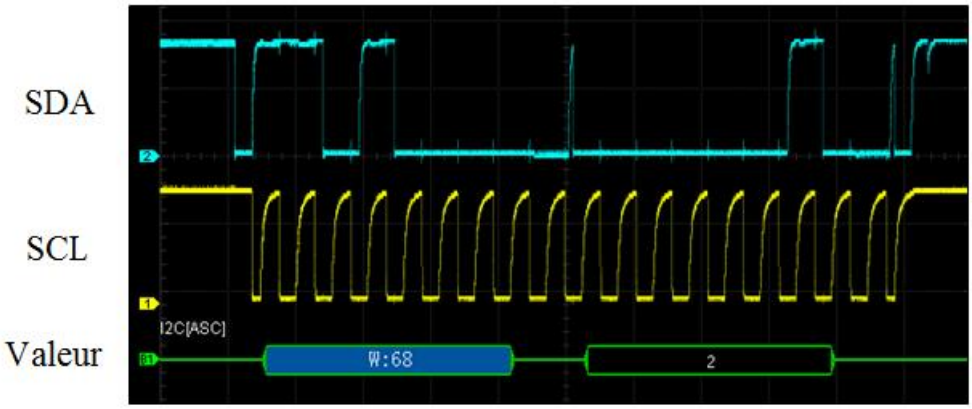


Figure 21 : Trame pour l'écriture

Ensuite pour la lecture, on réécrit l'adresse du capteur avec le bit « R/W » à « 1 » comme pour l'ultrason.

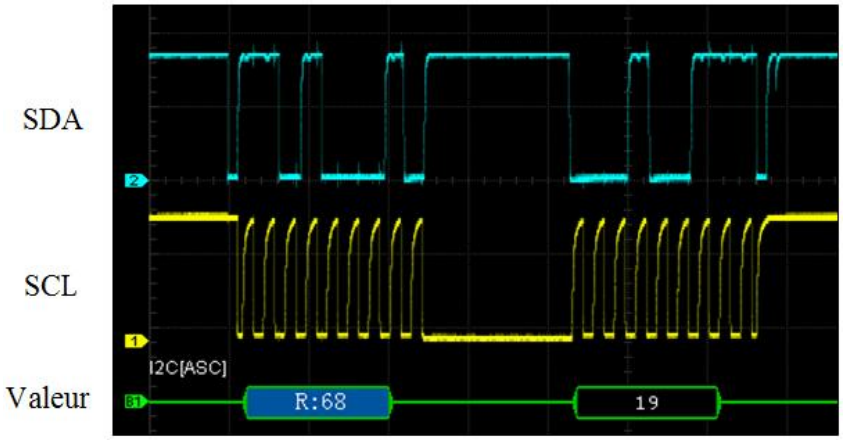


Figure 22 : Trame pour la lecture

Sur cette trame, la température de la zone qui a été lue pendant la simulation est de 19°C. Cette opération est répétée 8 fois de suite pour obtenir les valeurs, sur les 8 zones, qui seront ensuite stockées dans un tableau.

Un second tableau est, par la suite, créé avec une nouvelle lecture des 8 zones. Une comparaison est ensuite réalisée entre les valeurs de ces deux tableaux (zones par zones). Après plusieurs essais, il a été observé que le bruit thermique au sein dûne pièce peut varier jusquôà plus ou moins 2°C.

Le seuil de détection a donc été fixé pour une différence supérieure à plus ou moins 3°C. La comparaison des valeurs est réalisée entre ces deux tableaux qui sont chacun renouvelés tous les dixièmes de secondes : les valeurs mesurées deviennent les anciennes valeurs et de nouvelles valeurs sont lues.

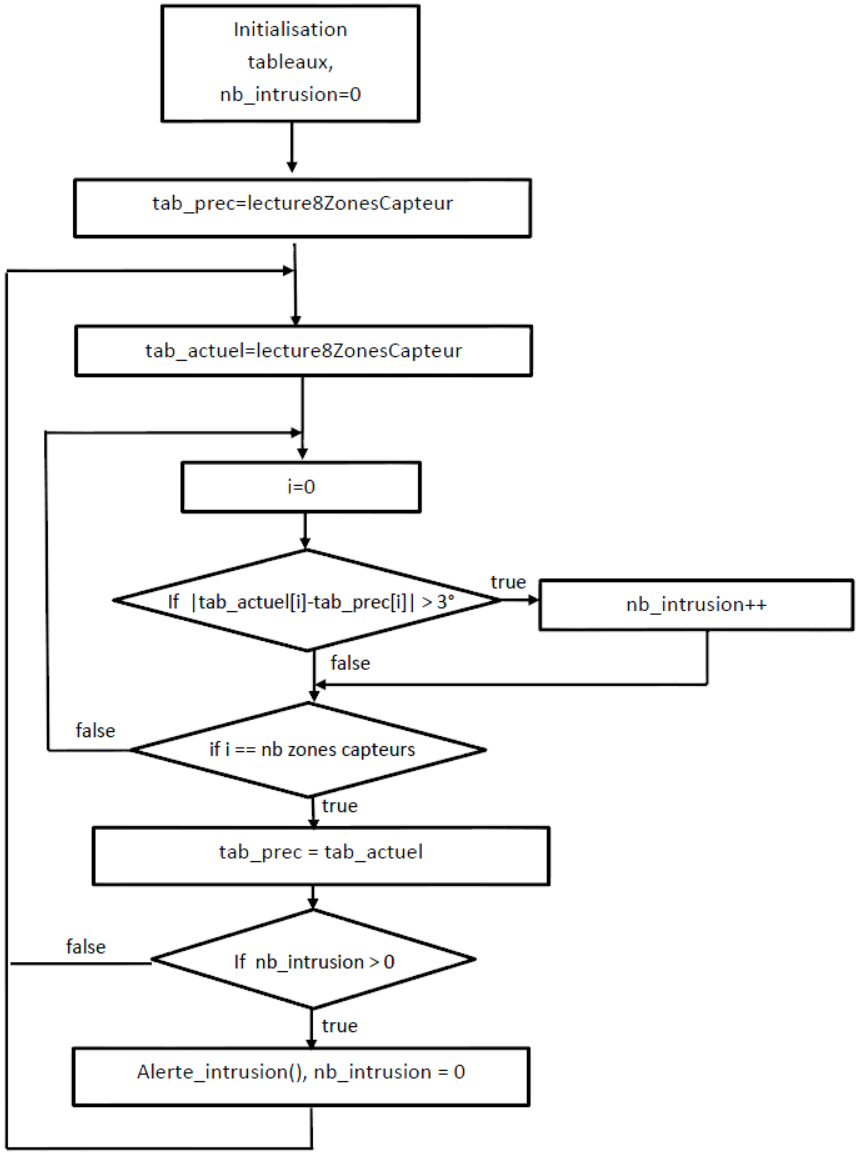


Figure 23 : Organigramme de détection par mesure de la température

5.4. Liaison UART entre le module GSM et le FPGA (carte DE1)

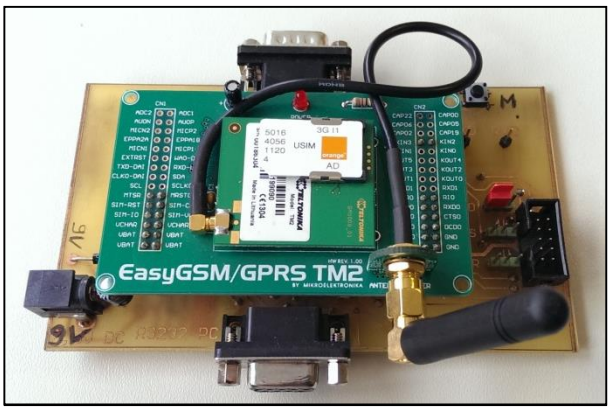


Figure 24 : Module GSM

Lô3SEN possédait déjà le module GSM avec deux connections UART (une pour le relier à un ordinateur et une autre pour le relier directement sur la carte DE1).

Afin de le mettre en route et de tester son bon fonctionnement, il a été connecté via un câble USB à un ordinateur. Ensuite, le port de communication doit être connu pour utiliser lôhyperterminal Putty. Ce terminal permet dôécrire en mode commande les trames à saisir pour sôenvoyer un SMS.

Plusieurs étapes sont alors indispensables. Lôutilisation dôune carte SIM sur le module GSM étant naturellement présent, une commande comportant le code PIN de la carte afin de la déverrouiller doit être validée.

Une fois la carte SIM déverrouillée et le numéro de téléphone du destinataire également rentré en mode commande, un SMS peut être envoyé.

Attention, afin dôéviter un bug du terminal lors du choix du message à envoyer, celui-ci doit comporter un nombre de caractères vraiment petit.

Commande dans lôhyperterminal Putty :

- **AT+CMEE=2** // autorise les codes dôerreurs en mode détaillé
  OK
  AT+CPIN="0000" // code pin
  OK
  AT+FLO=0 // pas de contrôle de flux
  OK
  AT+CMGF=1 // format message SMS = mode texte
- OK
  AT+CMGS="0683312811" // numéro destinataire
  > **message ctrl Z escape** // message
  OK
  AT+CPWROFF // extinction du module
  OK

Ensuite, un programme en C a été réalisé sous Eclipse afin de permettre au module GSM de communiquer avec le FPGA. Ainsi, le capteur et le module GSM étant reliés tous les deux au FPGA, une fois quôune intrusion est détecté dans lôhabitat via le capteur, un SMS est envoyé au propriétaire.

5.5. Création de la carte électronique

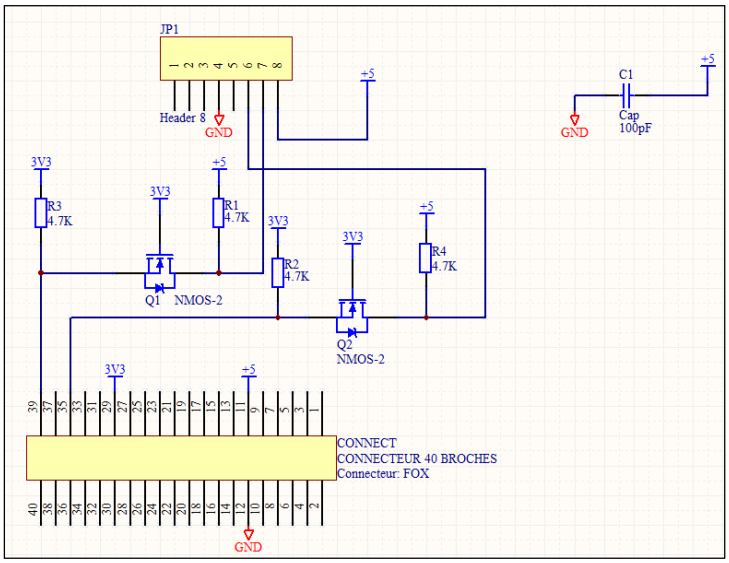


Figure 25 : Schéma électrique

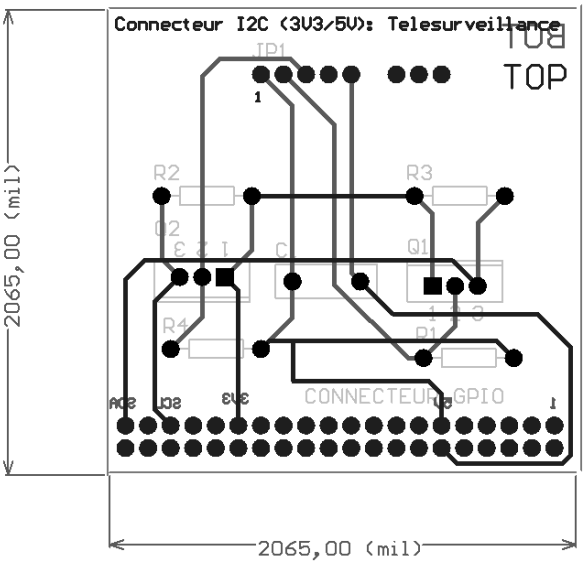


Figure 26 : PCB

Afin de pouvoir réaliser la carte électronique, la création dôun schéma électrique a avant tout été nécessaire. Pour se faire, certain composants ont dû être créé (connecteur GPIO et le connecteur pour le capteur) ainsi que leurs empreintes. Il sôagit de la forme quôils auront (nombre de pastillesé) une fois mis sur la carte. Pour les autres composants comme les résistances, les condensateurs ou les transistors, ils étaient déjà présents dans les librairies existantes.

Suite à cela, la carte finale, côest-à-dire le PCB a pu être créé. Pour se faire, les composants ont été placés sur la carte aux dimensions qui avait été choisi (celle du connecteur GPIO) et le routage a pu être effectué (liaisons entre les différents éléments). En effet, les mesures de la carte ont été faites de sorte que celle-ci soit de la largeur du connecteur afin de pouvoir la brancher correctement sur le FPGA.

Cette carte permet de relier le capteur au FPGA grâce aux broches du GPIO (connecteur HE10 de 40 broches). Concernant les composants présents, il y a quatre résistances (4,7KÝ) de pull-up reliées au 3,3V et au 5V. Ces résistances reliées à des transistors NMOS permettent dôadapter la tension de la carte qui est de 3,3V à la tension de fonctionnement du capteur qui est de 5V. Un condensateur pour le découplage (afin dôéviter tout choc brutal à lôallumage) est également présent (1µF). Plus la valeur de ce composant sera grande, plus le temps de charge sera long.

5.6. Système global

Le système global contient :

- un CPU : le Nios II (il coordonne tous les périphériques)
- une mémoire externe au FPGA (SDRAM de 8MB)
- un timer pour autoriser les interruptions
- une UART pour communiquer avec le module GSM
- deux IP I2C qui sont utilisées pour communiquer avec les deux capteurs (thermique et ultrason)
- trois PIO, un en entrée pour le switch et deux en sortie pour les LEDs et les afficheurs 7 segments
- un JTAG UART qui permet dôimplanter lôarchitecture matérielle et de programmer le software. LôUSB Blaster passe également par cette connexion (communication entre le SoPC et lôordinateur).
- un SYSTEM ID

Tous ces périphériques sont reliés entre eux grâce au bus Avalon. Pour ce qui est du JTAG UART, il communique avec lôordinateur avec lôUSB. La sortie UART est quant à elle reliée au RS232 et les deux IP aux GPIO (0 et 1) de la carte DE1. La mémoire externe est rattachée à la SDRAM de la carte.

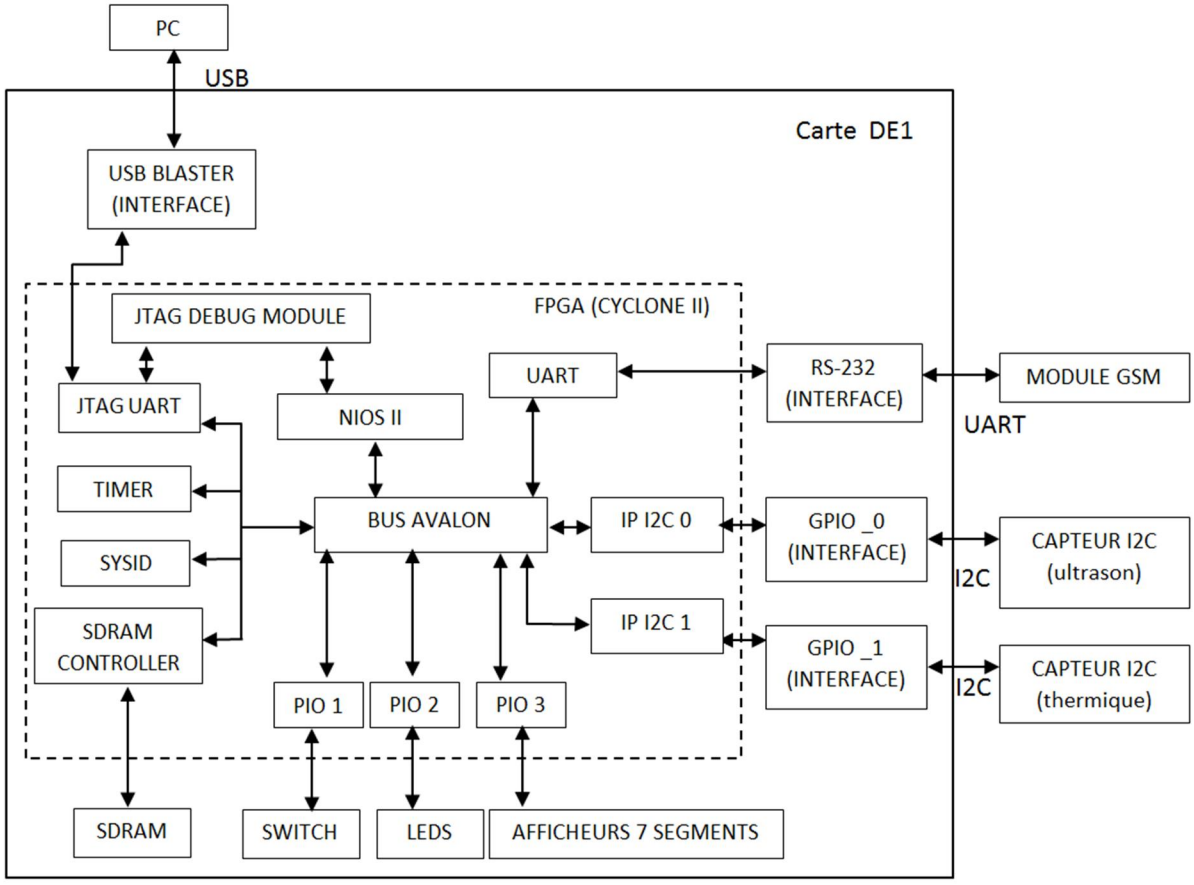


Figure 27 : Architecture globale

Analyse des ressources utilisées dans le FPGA (Cyclone II, EP2C20F484C7) :

Ressources	Nombre utilisé	Nombre total	Pourcentage utilisé
Eléments logiques	3 874	18 752	21%
Registres	2631		
Pins	82	315	26%
Mémoire interne	46 720	239 616	19%

5.7. Comparaison avec une autre technologie

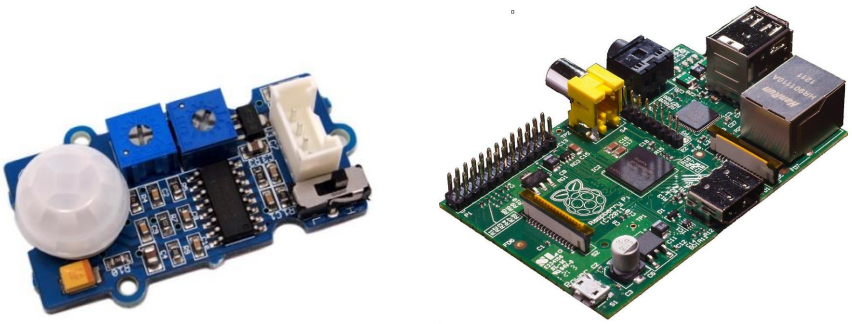


Figure 28 : Capteur PIR et Raspberry Pi

Dans le but dœétudier et de comparer le système de télésurveillance précédemment réalisé, un capteur PIR fonctionnant avec une seule sortie (état haut quand une personne est détectée, état bas sinon) a été relié à un Raspberry Pi. En effet, cette petite carte électronique est en fait un mini-ordinateur. Il contient le programme qui permet de lire la sortie du capteur et dœenvoyer un SMS grâce à un téléphone portable relié par USB. Le programme a été développé en bash grâce à lœéditeur de texte Vim et une connexion SSH.

Pour envoyer un SMS, lœutilitaire Gammu a été utilisé afin dœenvoyer les AT-Commandes au module GSM interne au téléphone mobile. Le capteur a, par défaut, sa sortie OUT à lœétat bas, quand il détecte un mouvement celle-ci passe à lœétat haut et est maintenue à cet état quelques secondes. En plus de cette sortie relié à un pin du GPIO du Raspberry Pi, deux fils permettent de lœalimenter (+5V et la masse).

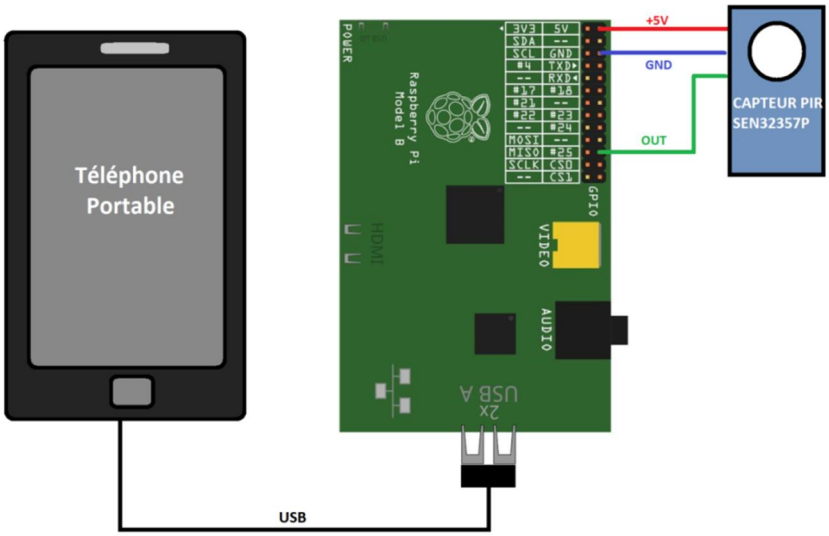


Figure 29 : Schéma du système utilisant le Raspberry Pi et le capteur PIR

Suite à la réalisation de ce montage, une comparaison a pu être rapidement faite. Cette nouvelle technologie est avant tout beaucoup moins encombrante que celle étudiée précédemment. Le module GSM a été remplacé par un simple téléphone portable et la liaison avec le capteur est beaucoup plus simple. Le code assez lourd pour créer lôIP I2C nôest alors plus nécessaire. En seulement quelques lignes de code le capteur fonctionne correctement. En effet, comparé aux capteurs utilisés auparavant, celui-ci est beaucoup plus facile dôutilisation.

Il est intéressant de voir quôavec un intitulé de projet qui est la télésurveillance, plusieurs technologies peuvent être mise en place afin de pouvoir en comparer les avantages et les inconvénients ainsi que le coût de conception qui reste un point important au sein dôune entreprise.

Cependant, la base du projet étant la réalisation et lôutilisation dôune IP I2C, cette dernière partie est beaucoup moins intéressante dôun point de vue pédagogique.

Capteurs	Avantages	Inconvénients	Portée
Capteur ultrason (SRF 08)	- données faciles à exploiter sôil y a un mouvement - très solide	- données imprécise (rayonnement trop large)	3 m
Capteur Thermique (MTP81)	- balayage large grâce à un servomoteur - résolution élevée (8 mini-zones)	- prix plutôt élevé - mesures imprécises après 2 m	5 m
Capteur PIR (SEN32357P)	- facile dôutilisation - prix faible	- peut pédagogique à lôutilisation	6 m

Estimation du prix de conception :

Carte DE1 + capteur ultrason + module GSM ≈ 90ú + 40ú + 40ú ≈ 170ú
Carte DE1 + capteur thermique (MTP81) + module GSM ≈ 90ú + 100ú + 40ú ≈ 230ú

Raspberry Pi + capteur PIR (SEN32357P) + téléphone portable ≈ 35ú + 10ú + 10ú ≈ 55ú

Cette estimation correspond aux équipements utilisés durant ce projet. Cependant, en entreprise, un simple FPGA (Flash, Oscillateuré) beaucoup moins onéreux quôune carte DE1 serait surement utilisé.

6. Tests

Ce tableau permet de récapituler tous les tests qui ont été nécessaires durant la réalisation de ce projet.

Fonction testée	Méthode et matériel	Résultat Attendu	Résultat Obtenu
Tests pour la carte électronique			
Absence de court-circuit sur la carte	Utilisation d'un ampèremètre en mode « sonore ». Si celui-ci bip cela veut dire que la connexion entre les deux broches est bien faite	Aucun court-circuit de décelé	Validé
Alimentation en 3V3 et 5V	Toujours à l'aide d'un voltmètre, on vérifie les tensions	Bonne tension aux bornes des composants souhaitées	Validé
Connexion de la carte aux ports GPIO de la carte DE1	Grâce à un oscilloscope, les trames SCL et SDA peuvent être observées	Trames corrects à l'oscilloscope (adresse, acquittementé)	Validé
Tests pour le module GSM			
Fonctionnement du module GSM	Connexion du module via un câble UART / USB à un ordinateur. Utilisation de l'hyperterminal Putty avec l'envoi de trames	Chaque trame ayant un objectif (déverrouillage code PINé), celle-ci dit être acquittée (OK). A la fin de toutes les trames, un SMS doit être envoyé	Validé (Envoi et réception d'un SMS)
Fonctionnement du module via la carte DE1	Connexion du module à la carte en UART. Envoi des trames à partir du Nios II	Lorsque toutes les trames ont été envoyées, un SMS a bien été envoyé au numéro qui était indiqué	Validé (SMS reçu)
Test de l'I2C			
Fonctionnement du module VHDL sans le Nios II	Mise de l'I2C sur les pins du GPIO de la carte DE1	Observation à l'oscilloscope des bonnes trames de l'I2C	Validé (observation des bonnes trames mais sans acquittement car le capteur n'est pas présent)

Fonctionnement de la communication entre lôIP et le Nios II	Lecture des registres de lôIP	On vérifie quôon a bien les flags et les valeurs des registres aux bons moments	Validé
Fonctionnement de lôIP contrôlée par le Nios II	Le programme écrit et lit les registres de lôIP pour envoyer les trames I2C	Observation des trames sur les pins du GPIO	Validé (observation des bonnes trames mais sans acquittement car le capteur nôest pas présent)
Test du capteur ultrason			
Fonctionnement du capteur avec la lecture dôune distance	Via la liaison I2C le capteur peut lire la donnée quôil reçoit (Code C : Eclipse)	On a souhaité retourner la valeur lue sur les afficheurs 7 segments	Validé (Valeur bien affichée)
Détection intrusion	Capteur relié par I2C	Allumage de la LED si intrusion	Validé
Test du capteur thermique (MTP81)			
Fonctionnement du capteur avec la lecture dôune température	Toujours grâce à la liaison I2C, le capteur peut lire la donnée quôil reçoit (Code C : Eclipse)	On souhaite retourner la température lue sur les afficheurs 7 segments	Validé (Température correctement affichée)
Détection intrusion	Capteur relié par I2C	Allumage de la LED si intrusion	Validé
Test des périphériques de la carte			
Changement de capteur à lôaide dôun Switch	A lôaide dôun switch, le choix du capteur peut être effectué (capteur ultrason sur GPIO 0 / SW0 à ó1ô et capteur thermique sur GPIO 1 / SW0 à ó0ô	On souhaite pour permuter le choix du capteur : SW0 à ó1ô la mesure de la distance est retournée ; SW0 à ó0ô la mesure de la température est lue	Validé (En fonction du switch, température et lecture de la distance correctement affichée)
Allumage dôune LED lors dôune intrusion	La LED LEDR0 sôallume si : -Pour le capteur ultrason : intrusion → changement rapide de la distance mesurée (différence importante avec la valeur lue précédemment) -Pour le capteur thermique : intrusion → variation brutale de la température lue	La LED LEDR0 doit sôallumer lorsquôune intrusion est détectée	Validé (Allumage correct de la LED)

Affichage des LEDs en fonction du capteur choisi	Grâce à deux LEDs (LEDR2 et LEDR3) certaine informations sont obtenues : LEDR2 → capteur ultrason choisi et LEDR3 → capteur thermique choisi	L'allumage des LEDs en fonction de la position du switch est correct. De plus, lors d'une intrusion la LED s'allume instantanément	Validé (Allumage des LEDs)
Test implémentation finale			
Fonctionnement système complet	Module GSM connecté par UART (RS-232) et capteurs reliés par I2C (GPIO)	Envoi d'un SMS lors d'une variation de la distance ou de la température	Validé
Test technologie différente (Raspberry Pi + capteur PIR SEN32375P + portable)			
Fonctionnement du capteur PIR	Capteur relié au GPIO du Raspberry Pi	Lecture du pin relié au capteur et affichage « lecture ok » quand le pin passe à '1'	Validé
Fonctionnement de l'envoi de SMS	Portable relé au Raspberry Pi par USB	Envoi d'un message et vérification sa bonne réception	Validé
Fonctionnement du système total avec l'envoi du SMS par le portable	Capteur relié au GPIO du Raspberry Pi et portable relié en USB	Envoi d'un message quand le capteur détecte une intrusion	Validé

7. Conclusion

Une première partie concernant les objectifs atteints peut être développée. En effet, à la fin d'un projet le résultat prend une grande place et permet de se positionner par rapport au cahier des charges réalisé en début de projet. L'objectif principal de ce projet était de pouvoir communiquer via une liaison I2C avec un capteur, celui-ci a été atteint. Deux capteurs possédant ce moyen de communication ont été implantés sur la carte DE1.

Avant d'arriver à ce résultat, certaines difficultés ont cependant été rencontrées. En effet, après avoir développé plusieurs parties en parallèle, il était difficile de détecter d'où venait certaines erreurs. L'utilisation de quatre logiciels comme Qsys, Quartus, Eclipse ou encore Modelsim ne favorisait pas la recherche de l'erreur dans le système. Cela engendrait donc une perte de temps assez importante. Certains objectifs du projet qui avaient été fixés n'ont ainsi pas pu être atteints.

En effet, des améliorations possibles touchant à l'IP, telles que les interruptions, n'ont pas pu être misent en place.

De plus, un module pour la mise en route d'une alarme sonore si un intrus est détecté aurait mérité d'être créé. Cela aurait naturellement dissuadé la personne de rester dans

lôhabitat. Il aurait également été intéressant de mieux organiser son temps afin de mettre en place une interface pour configurer le numéro de téléphone à appeler en cas dôintrusion ainsi quôune liaison Bluetooth ou autre entre les différentes cartes à lôintérieur dôun même bâtiment.

8. Bibliographie

- Documents papier :

Cours sur les systèmes on Chip réalisé par Mme Le Lay (option Systèmes Embarqués)
Cours sur Protel DXP réalisé par Mr Corbel (CSI3)
TD sur la commande I2C réalisé par Mr Reboux (CSI3)

- Liens internet :

Capteur thermique MTP81 :

http://www.lextronic.fr/P1728-capteur-thermique-mtp81.html

Capteur thermique PIR Grove (SEN32357P) :

http://www.seeedstudio.com/wiki/index.php?title=Twig_-_PIR_Motion_Sensor

Composant I2C :

http://opencores.org/project,i2c_master_slave

- Documents PDF :

Module GSM : GSM_GPRS_TM2 user manual EN08
Utilisation de lôhyperterminal puTTY

Capteur Ultrason : Telemetre_ultrason_sfr08_06602

9. Annexes

Script Bash pour le contrôle du capteur PIR et l'envoi de SMS avec le Raspberry Pi :

```
setup ()
{
        #initalisation du pin 25 du gpio
        echo Setup
        cd /sys/class/gpio/
        ls
        echo 25 > export
        ls
        cd gpio25/
}

attenteCapteur()
{
        echo -n "en attente du capteur ... "
        #tant que le pin 25 est à zéro on attend (on lit 'value' dans le dossier  gpio25/)
        while [ `cat value` = 0 ]; do
                sleep 0.01
        done
        echo "intrusion"
        #on envoi le SMS d'intrusion
        echo -e "alerte intrusion n: $1\n$(date +%H:%M)\n$(date +%d/%m/%Y)" | gammu --sendsms TEXT 06xxxxxxxx
        sleep 10
}


#envoi du SMS de mise en route
echo -e "allumage raspberry\n$(date +%H:%M)\n $(date +%d/%m/%Y)" | gammu --sendsms TEXT 06xxxxxxxx

#on initialise le pin25 avec la fonction setup
setup

#on initialise le nombre d'intrusion à zéro
var = 0

#while 1 car le système ne doit jamais s'arrêter
while true; do
        ((var++))
        #fonction qui bloque tant qu'il n'y a pas d'intrusion
        attenteCapteur $var
        echo inrusion numero $var
done
```