الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Filière Électronique
Spécialité Electronique des Systèmes Embarqués

Réalisé par

SOUDAOUI Khadidja

&

ABDELMALEK Yasmine

Conception et prototypage d'un nœud de capteur à base de circuit FPGA

Encadré par : Mme. IZABOUDJEN Nouma

Mme. BOUGHRIRA Hamida

Année Universitaire 2018-2019

Remerciements

Les travaux présentés dans ce manuscrit ont été réalisés au sein de la division microélectronique et nanotechnologies (DMN) du Centre de Développement des Technologies Avancées (CDTA) à Alger.

Nous remercions tout d'abord **DIEU** qui nous a donné la force et le courage pour pouvoir réaliser ce travail.

Nous tenons à exprimer nos sincères remerciements et reconnaissances, tout d'abord, à notre encadreur **Mme. IZEBOUDJEN Nouma** d'avoir proposé ce sujet qui nous a mené vers la découverte du monde de l'expérimental et qui nous a bien accueilli pendant toute la période de notre stage, elle nous a bien dirigé afin de pouvoir réaliser ce travail de mémoire, nous n'oublions pas de la remercier pour sa disponibilité et ses conseils précieux.

De même, nous adressons notre profonde gratitude à notre promotrice **Mme.BOUGHRIRAHamida** pour sa confiance, son soutien et tous ses conseils précieux.

Nous n'oublions pas de remercier vivement **Mme LAZIB Dalila,** ingénieur chercheur au sein de la DMN**,** pour la formation qu'elle nous a donné sur la conception des circuits imprimés ainsi que les différents conseils.

Aussi, nous tenons à remercier **Mme Titri Sabrina**, chercheur au sein de la division DMN, équipe TDCSD, pour les informations sur la conception FPGA et l'outil VIVADO de Xilinx.

Nous remercions également les membres du jury qui ont accepté d'évaluer notre travail.

Sur un plan plus personnel, nous remercions vivement nos parents pour leurs encouragements et leur soutien pendant notre cursus, ainsi que tous les membres de nos chères familles et tous nos amis.

Enfin, nous remercions tous ceux qui ont contribué de près ou de loin au bon déroulement de ce travail.

Dédicaces

J'exprime mes sentiments les plus profonds et je dédie ce

Modeste travail

A mon père, ma mère, pour l'éducation qu'ils ont su me donner et qui m'a permis avec l'aide de ***DIEU*** *d'arriver là où je suis.*

A Mes chères sœurs ***BAYA*** *et* ***IMAN*** *et* ***SALSABILE tASNIME.*** *Chères frères* ***AMAR*** *et* ***AYOUB*** *et* ***SALAH AL DINE,***

A toute ma famille.

A tous mes amis ***Yasmine*** *et* ***Nabila*** *et tous ceux qui m'ont aidé afin de réaliser ce travail.*

KHADIDJA

Dédicaces

*Au **DIEU** Tout-puissant, qui m'aime toujours. . .*

Je dédie ce mémoire

A mes chers parents, ma mère et mon père pour leur patience, leur amour, leur soutien et leur encouragement

*A mes Chères sœurs **Faiza, Sidra** MES Chères frères **Oussama** et **Marwane***

A toute ma famille.

*A tous mes amis **kawter** et **assia** et tous ceux m'aiment et qui m'ont aidé afin de réaliser ce travail.*

YASMINE

ملخص

عقدة المستشعر عبارة عن نظام يتكون أساسًا من جهاز استشعار واحد أو أكثر لالتقاط البيانات، ووحدة معالجة البيانات، ودائرة اتصال هذا، نحن مهتمون بتصميم ، ودائرة إدارة الطاقة. كجزء من مشروع درجة ماستر(Ethernet، وBluetooth، وZIGBEE، وWIFI) خارجية ينقسم المشروع إلى خطوتين رئيسيتين: الخطوة الأولى هي تنفيذ البنية الداخلية لوحدة FPGA. النماذج الأولية لعقدة الاستشعار المبنية على دائرة والخطوة الثانية هي تصميم نموذج أولي جديد لوحة دوائر مبنية حول دائرة XILINX ARTIX-7 FPGA معالجة عقدة الاستشعار على دارة ، عرض من سبعة أجزاء، UART) ، وجهازي استشعار (مقياس التسارع، جهاز استشعار درجة الحرارة) وعدد من الأجهزة الطرفيةFPGA NEO430 يتكون من معالج (SOC) ، إلخ). في الخطوة الأولى، قمنا بتطبيق بنية الأجهزة لوحدة المعالجة. يمثل الأخير نظامًا على رقاقةJTAG (VHDL). الخصائص الاختيارية) الموصوفة بالكامل في لغة وصف الأجهزة فائقة السرعة) IPs وعدد من مكونات soft تُستخدم أداة Vivado في الخطوة الثانية، قمنا بتصميم الدائرة المطبوعة لعقدة المستشعر، Xilinx Artix-7 FPGA. على شريحة SOC لتجميع وتنفيذ وتحليل قدرة FPGA (XDC) لهذا السبب، ونظرا لتعقيد المشروع، اعتمدنا منهجية تصميم تستخدم ملفات قيود تصميم Altium designer. باستخدام أداة ونتائج تحليل الطاقة لاستغلالها كملف إدخال للمشروع. تصميم ثنائي الفينيل متعدد الكلور. النتائج التي تم الحصول عليها تنافسية فيما يتعلق بالمنتجات الأخرى وتُظهر أن عقدة المستشعر تستهلك طاقة ثابتة تبلغ 97 ميجا وات وتردد 100 ميجا هرتز، كما أن ثنائي الفينيل متعدد الكلور النهائي عبارة عن دائرة مطبوعة مكونة من 8 طبقات وأبعاد 10 سم × 10 سم.

كلمات البحث : Altium ، مصممVIVADO ، أداةFPGA ARTIX-100T، SOC، Neo430 عقدة الاستشعار، معالج

Résumé

Un nœud de capteurs est un système composé principalement d'un ou plusieurs capteurs pour capter des données, d'une unité de traitement de ces données, d'un circuit de communication avec l'extérieur (WIFI, ZIGBEE, Bluetooth, Ethernet), et d'un circuit de gestion de l'alimentation. Dans le cadre de ce projet de fin d'études Master, nous nous intéressons à la conception et le prototypage d'un nœud de capteur construit à base d'un circuit FPGA. Le projet est divisé en deux grandes étapes : la première étape consiste à implémenter l'architecture interne de l'unité de traitement du nœud de capteur sur un circuit FPGA XILINX ARTIX-7 et la deuxième étape consiste à concevoir le prototype d'un nouveau circuit imprimé construit autour du circuit FPGA, de deux capteurs (accéléromètre, capteur de température) et d'un certain nombre de périphériques (UART, afficheur sept segment, JTAG, etc.). Dans la première étape nous avons implémenté l'architecture hardware de l'unité de traitement. Cette dernière représente un système sur puce (SOC) composé du processeur soft NEO430 et d'un certain nombre de composants IPs (intelectual properties) décrits entièrement en VHDL (Very high speed Hardware Description Language). L'outil Vivado est utilisé pour la synthèse, l'implémentation et l'analyse de la puissance du SOC sur le circuit FPGA Artix-7 de Xilinx. Dans la deuxième étape, nous avons conçu le circuit imprimé du nœud de capteur, moyennant l'outil Altium designer. Pour cela et vu la complexité du projet, nous avons adopté une méthodologie de conception qui consiste à utiliser les fichiers de contraintes de conception sur FPGA (XDC) et les résultats d'analyse de la puissance pour les exploiter comme fichier d'entrée pour la conception du circuit imprimé. Les résultats obtenus sont concurrentiels par rapports à d'autres produits et montrent que le nœud de capteur consomme une puissance statique de 97mWatt, une fréquence de 100 Mhz, et le PCB final est un circuit imprimé de 8 couches et de dimensions de 10cmx10cm.

Mots clés : nœud de capteur, processeur Neo430, SOC, FPGA ARTIX-100T, outil VIVADO, Altium-designer

Abstract

A sensor node is a system consisting mainly of one or more sensors for capturing data, a data processing unit, an external communication circuit (WIFI, ZIGBEE, Bluetooth, Ethernet), and a power management circuit. As part of this Master degree project, we are interested in the design and prototyping of a sensor node built on an FPGA circuit. The project is divided into two main steps: the first step is to implement the internal architecture of the sensor node processing unit on a XILINX ARTIX-7 FPGA circuit and the second step is to design the prototype of a new circuit board built around the FPGA circuit, two sensors (accelerometer, temperature sensor) and a number of peripherals (UART, seven-segment display, JTAG, etc.). In the first step we implemented the hardware architecture of the processing unit. The latter represents a system-on-a-chip (SOC) consisting of the NEO430 soft processor and a number of IPs (intelectual properties) components fully described in Very High Speed Hardware Description Language (VHDL). The Vivado tool is used for synthesizing, implementing and analyzing the power of the SOC on the Xilinx Artix-7 FPGA chip. In the second step, we designed the printed circuit of the sensor node, using the tool Altium designer. Because of this, and given the complexity of the project, we adopted a design methodology that uses FPGA (XDC) design constraints files and power analysis results to exploit them as an input file for the project. PCB design. The results obtained are competitive with respect to other products and show that the sensor node consumes a static power of 97mWatt, a frequency of 100 Mhz, and the final PCB is a printed circuit of 8 layers and dimensions of 10cmx10cm.

Keywords : sensor node, Neo430 processor, SOC, FPGA ARTIX-100T, VIVADO tool, Altium-designer

Listes des acronymes et abréviations

A Ampère

BRAM Bloc RAM

CAN Convertisseur Analogique Numérique

CMOS Complementary Metal Oxide Semiconductor

CPLD Complex Programmable Logic Device

CPU Central Processing Unit

CTL CenturyLink

CLB Configurable Logic Bloc

CT Criitical Temperature

CPU Central Processing Unit

DARPA Defence Advanced Research Projects Agency

DDR Double Data Rate

DSN Distriuted Sensor Network

DSP Digital Signal Processor

DMA Direct Memory Access

DDR Double Data Rate

DRC Design Rul Check

EEPROM Electrically Erasable Programmable Read Only Memory

FPGA Field Programmable Read Only Memory

GPS Global Positioning System

Ghz Giga Hertz

IOB Input Output Bloc

I2C Internet Integrated Circuit

IP Iintellectual Property

JTAG Joint Test Action Group

Khz Kilo Hertz

LCD Liquid Crystal Display

LE Logic Elements

LSB Least Significant Bit

LUT Look Up Table

MAC Multiply and Accumulate : Medium Access Control

MEMS Micro Electro-Mecanical Sensor

MSB Most Significant Bit

Mbps stands for Megabits per second

MHz Mega Hertz

Mm Millimeter

Mbps stands for Megabits per second

Mb Mega Byte

Mo Mega octet

PCB Printed Circuit Board

PAL Programmable Logic Array

PLD programmable Logic Device

RCSF Réseaux de Capteur Sans Fils

ROM Read Only Memory

RTL Register Transfer Level

SCL Serial Clock

SDA Serial Data

Asic Application Specific Integreted Circuit

SFU Special Function Unit

SOC System On Chip

SPI Serial Peripheral Logic Device

SRAM Static Programmable Logic Device

TFBGA Thin-Prole Fine Pitch Ball Grid Array

Um Micrometer

USB Universal Serial Environment

UART Universal Asynchronous Receiver Transmitter

UART Universal Asynchronous Receiver Transmitter

V Volt

VHDL Very high speep Hardware Description Language

W Watt

XDC Xilinx Design Constraint

W Watt

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

De nos jours l'électronique est présente dans la moindre de nos activités quotidiennes, les voitures, les cafetières, les téléphones portables, les appareils photo et bien sûr dans les ordinateurs. Toutes ces applications nécessitent la réalisation de cartes électroniques, d'un ensemble de composants tel que des résistances, condensateurs ou circuits intégrés réunis sur une plaque et reliés électriquement entre eux, de manière à former un circuit complexe destiné à un usage précis. Cela nous amène donc à nous demander : Quels sont les différentes étapes, de la conception à la fabrication, dans la réalisation d'un circuit imprimé ?

Depuis leur apparition au début du 20ème siècle, la réalisation des circuits imprimés a évolué de manière significative. Et, avec l'évolution de la technologie de fabrication et le besoin sans cesse croissant de la miniaturisation des circuits électroniques, actuellement, les circuits imprimés sont caractérisés par une grande densité d'intégration et, de ce fait, nécessitent plusieurs couches pour pouvoir interconnecter les différents composants d'une même carte. On parle alors, de circuits imprimés multicouches dont le nombre pouvant atteindre 23 couches ou plus.

Dans ce contexte, l'objectif de ce projet de fin d'étude master porte sur la conception d'un nouveau circuit imprimé permettant l'implémentation d'un nœud de capteur construit à base d'un circuit programmable de type FPGA (Fiel programmable gate Array) et d'un ensemble de périphériques (UART, capteur de température, accéléromètre, afficheur sept segment, Alimentation, etc.).

Le travail en question s'inscrit dans le cadre des travaux de recherche de l'équipe : « Tools Digital Integrated System and Circuit Design » (TDCSD) de la division microélectronique et nanotechnologie du CDTA et s'insère dans le cadre du projet socioéconomique : N°14/CDTA/DGRSDT : « Nœud de capteur intelligent reprogrammable ».

Dans sa globalité, le projet consiste à proposer une nouvelle solution pour l'implémentation d'un nœud de capteur à base d'un circuit FPGA de la famille Virtex-7 Artix-7, permettant un compromis entre la faible consommation de puissance offerte par les microcontrôleurs et la rapidité de traitement offerte par les circuits FPGA; ce qui permet de relever les défis actuels liés aux applications des nœuds de capteurs : Internet des objets, multimédia, classification, etc.

Pour notre part, notre mission consiste, d'abord, à implémenter sur FPGA l'architecture de l'unité de traitement d'un nœud de capteur moyennant l'outil VIVADO de Xilinx, et ensuite utiliser les fichiers de contraintes de conception ainsi que les résultats d'implémentation issues de l'outil VIVADO de Xilinx, pour les exploiter dans la conception d'un circuit imprimé multicouches et optimisé en terme de surface, et ce moyennant l'outil Altium Designer.

L'architecture hardware de l'unité de traitement représente un système sur puce (SOC) construit autour du processeur soft NEO430 et d'un certain nombre de composants IPs (intelectual properties) décrits entièrement en VHDL (Very high speed Hardware Description Language). Le processeur soft NEO430 et les composants IPs appartiennent au domaine publique (opensource/opencores) et leur code source est téléchargeable gratuitement. L'outil Vivado est utilisé pour la synthèse, l'implémentation et l'analyse de la puissance du SOC sur le circuit FPGA Artix-7 de Xilinx hébergé sur la carte de développement Nexys-4 DDR de Diligent.

La partie conception du circuit imprimé de notre système est basée sur l'outil Altium Designer. Cet outil est utilisé pour saisir des schémas électriques, les vérifier, les simuler et aller jusqu'à la conception du circuit imprimé. La grande force d'Altium Designer c'est les nombreux outils qu'il intègre par défaut. Ce logiciel représente une seule solution informatique pour concevoir et développer un montage électronique ainsi que la simulation, le routage, l'édition et la fabrication de PCB, en allant du schéma jusqu'à la programmation des composants de ce schéma.

Ce mémoire comporte quatre chapitres, le premier chapitre présente une généralité sur les circuits FPGAs et les circuits imprimés.

Le seconde chapitre, est consacré aux nœuds de capteurs basés sur FPGA, où il sera question de présenter l'architecture générale de ces dernier et leur domaines d'application, leurs défis actuels et comment les FPGAs modernes peuvent adresser ces défis. Un état de l'art des différents travaux de recherche portant sur l'intégration des FPGAs dans les nœuds de capteurs sera aussi présenté dans ce chapitre.

Le troisième chapitre concerne l'implémentation matérielle de notre SOC, où il sera question de décrire l'architecture de notre système sur puce avec les différentes étapes et les résultats d'implémentation.

Le quatrième chapitre est dédié à la Conception du circuit imprimé du nœud de capteur basé sur FPGA où il sera question de présenter la méthodologie de conception que nous avons adopté ainsi que les résultats obtenus.

Et enfin, nous terminerons par une conclusion générale.

Chapitre I Généralité sur les circuits FPGA et les circuits imprimés

I.1 Introduction

Grace à sa reconfiguration facile et illimitée, un circuit FPGA (Field Programmable Gate Array), est capable d'implémenter des circuits et systèmes complexes, pour différents domaine d'applications. Par ailleurs, la conception à base de circuits FPGA, nécessite des outils dédiés et une méthodologie de conception permettant de transformer une description schématique ou en langage VHDL, en un fichier «Bitstream» qui sera programmé dans le circuit FPGA.

Dans le même contexte, et afin de concevoir un circuit imprimé à base de circuit FPGA, il est nécessaire de comprendre la technologie des circuits imprimé, des outils de conception de ces derniers et de définir une méthodologie de conception appropriée.

Dans ce chapitre, nous présenterons d'abord les concepts de base sur les circuits FPGA, en se basant sur les circuits de Xilinx. Ensuite, nous présenterons la technologie des circuits imprimés.

I.2 Les circuits reprogrammables

I.2.1 Structure de base

Un circuit intégré classique contient :

- des portes logiques.
- des connexions entre les portes logiques.
- des éléments de mémorisation (registres et/ou mémoires).

- des entrées/sorties.
- une (ou des) horloge(s).

Un circuit reconfigurable (circuit qui peut être reconfiguré après sa fabrication) a les mêmes éléments de base qu'un circuit statique avec la notion de reconfigurable. Partant du principe qu'une fonction logique peut s'exprimer sous forme canonique (somme de produits), cette Fonction logique peut être représentée par un réseau logique programmable constitué d'une Matrice ET et d'une matrice OU. De cette façon, en agissant sur les liaisons de la matrice ET ou la matrice OU, on peut changer la fonction réalisée. C'est ce qu'on appelle programmable/reconfigurable. Parmi les circuits reconfigurables basés sur ce principe, on cite les PLA (Programmable Logic Array) où la matrice ET et la matrice OU sont reconfigurables, et les circuits PAL (Programmable Array Logic) où seule la matrice ET est reconfigurable. Une fonction logique reconfigurable peut aussi être implémentée par les tables de vérité (Look Up Tables : LUT) construites à base de mémoire SRAM par exemple. Dans ce cas, l'adresse d'une case mémoire constitue une combinaison des entrées d'une fonction. Le contenu de la case mémoire correspond à la valeur que prend la fonction pour cette combinaison [1].

En combinant des cellules/réseaux logiques reconfigurables tels que les LUTs et les matrices ET et OU, on obtient des circuits logiques reconfigurables PLD (Programmable Logic Device). Les circuits PLD peuvent se décomposer en trois catégories : les SPLDs (Simple PLDs), les CPLDs (Complex PLDs) et les FPGAs. Les SPLDs sont des circuits composés d'un bloc d'entré, d'une matrice ET, d'une matrice OU et d'un bloc de sortie. Dans cette catégorie, on trouve les circuits PLA et PAL dont on a parlé précédemment. Les circuits CPLDs regroupent un ensemble de circuits reconfigurables.

Les FPGAs sont composés d'un réseau de blocs logiques, de cellules d'entrée/sortie et de ressources d'interconnexion totalement flexibles. La différence entre les FPGAs et les CPLDs est au niveau de l'architecture. Alors qu'un CPLD est constitué d'un ensemble de PLDs interconnectés à la matrice globale d'interconnexion, un FPGA a ses propres blocs logiques et chaque bloc peut implémenter une fonction logique. Ces blocs sont connectés, par des commutateurs reconfigurables, pour implémenter une fonction logique complète, à la différence d'un CPLD où des fonctions logiques complètes sont implémentées individuellement par les PLDs [1].

I.2.2 Les Avantages et les inconvénients

Les circuits reconfigurables sont apparus comme solution au manque de flexibilité des SoCs classiques tels que ceux basés sur les ASICs (Application Specific Integrated Circuits). Comme solution à ces problèmes, les circuits reconfigurables sont utilisés. Ces circuits ont pour avantages la conception rapide et la reconfiguration. Ce deuxième avantage permet aux circuits reconfigurables de changer de configuration selon l'application.

De plus, il est possible D'effectuer la reconfiguration durant l'exécution de l'application comme c'est le cas pour les FPGAs. Cela résout beaucoup de problèmes qui peuvent se produire au cours de l'exécution et qui peuvent être liés à la tolérance aux fautes ou à la performance.

En outre, la reconfiguration peut être effectuée plusieurs fois. Un autre avantage des circuits reconfigurables tels que les FPGAs est que les SoCs construits à base de ces circuits ont une durée de conception réduite en comparaison à celles des ASICs. Ceci est dû au fait que les changements sur les FPGAs peuvent être immédiats avant la fabrication, ce qui fait des FPGAs une solution pour le prototypage des systèmes à base d'ASICs.

L'inconvénient des SoCs reconfigurables est leur coût en comparaison aux SoCs personnalisés comme les ASICs. Alors que les systèmes personnalisés ont un coût de conception important, ils ont un coût unitaire de puce moins important que celui des systèmes reconfigurables [1].

I.2.3 Les circuits FPGAs de Xilinx

Les circuits FPGA du fabricant XILINX, structurés sous forme des matrices, utilisent deux types de cellules de base, les cellules logiques appelées CLB (Configurable logic Bloc), et les cellules d'entrées/sorties appelées IOB, Ces différentes cellules sont reliées entre elles par un réseau d'interconnexions configurable. La figure I.1 montre es différents blocs d'un circuit FPGA.

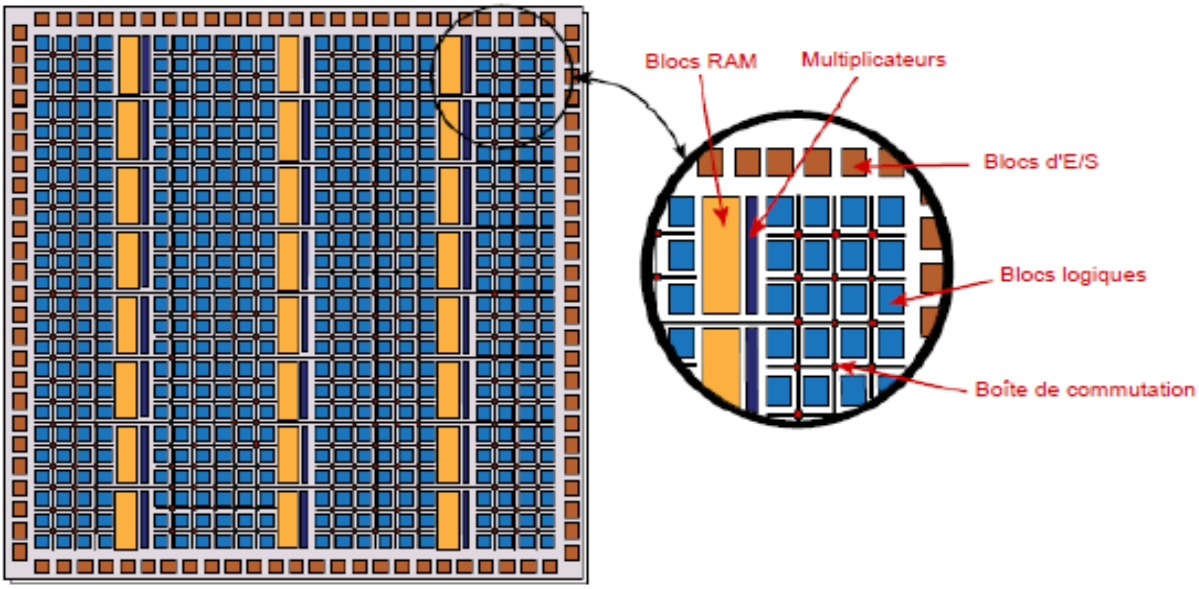


Figure I.1. Les différents blocs d'une FPGA Xilinx

I.2.4 Les blocs logiques CLBs (Configurable Logic Blocs)

Sont des éléments déterminants des performances du FPGA et contenant les fonctions logiques combinatoires et séquentielles.

- La partie combinatoire permet de réaliser des fonctions arithmétiques et logiques de complexité variable. Il est en effet possible d'utiliser plusieurs méthodes de synthèse dont les principales sont : la synthèse de fonctions à 4 ou 5 variables avec des portes classiques ET, OU et NON, AND, OR et NOR,la synthèse de fonctions combinatoires à l'aide de mémoires vives. Dans ce dernier cas, on dit aussi réalisation de fonctions logique par LUT (Look-Up Table).
- La partie séquentielle comporte en règle générale une ou deux bascules de type D.

Suivant le fabricant du circuit, ces blocs contiennent un nombre différent de portes logiques et de bascules à l'intérieur d'un bloc CLB (Configurable logico).

I.2.5 Les cellules d'entrées/sorties IOBs (Input Output Blocs)

Ces blocs permettent l'interface entre les broches du composant FPGA et la logique interne développée à l'intérieur du composant. Pour adapter les signaux suivants :

- Alimentation.
- Signaux d'horloge.

- Signaux de configuration du FPGA.
- Signaux de test.

  Les IOBs permettent l'interconnexion de la logique interne aux ports d'entrées et de Sorties du FPGA.

Les IOBs ont leur propre mémoire de configuration, elles stockent les standards de tension et la direction des ports. Ces blocs sont présents sur toute la périphérie du circuit FPGA. Chaque bloc IOB contrôle une broche du composant et il peut être reconfiguré en entrée ou en sortie, en signaux bidirectionnels ou être inutilisé (haute impédance)

I.2.6 Evolution des circuits FPGAs de Xilinx

Depuis leurs apparitions au milieu des années 80, les possibilités offertes par les circuits FPGAs n'ont cessé d'augmenter, ils ont connu une évolution et une révolution spectaculaire, comme suit :

Tableau I.1.Evolution des circuits Xilinx

Gamme (Génération)	Année	Taux d'intégration (Porte logique)	Caractéristiques
XC 2000	1985	600 ~ 1500 (2 µm)	
XC3000,	1987	1000 ~ 6000 (1.2 et 0.8 µm)	
XC4000	1991	2000 ~500000 (0,8 µm)	
XC5200	1995	6000~18000 (0,5 µm)	
XC6000	1996	10.000 ~100.000	
SPARTAN	1998		
VIRTEX	1998	0.5 ~1 millions	Blocs RAMs, horloges
VIRTEX-E	1999	0.5~4 millions	RAMs, multiplieurs
VIRTEX-II	2000	0.4~ 8 millions	plus de RAMs, Multiplieurs
VIRTEX-II-PRO	2002		RAM, multiplieurs, les processeurs Microblaze, POWERPC et IBM
VIRTEX-4	2004	90nm	RAM, multiplieur, DSP, LUT 4 entrée
VIRTEX-5	2006	65nm	LUT 6 entrées
VIRTEX-6	2009	40nm	
SERIE-7 (KINTEX, ARTIX, VIRTEX-7),	2010	28nm	processeur ARM CORTEX-9
SERIR UltraScale et UltraScale+(KINTEX, VIRTEX ,ZYNQ)	2013-2016	16nm et 20nm	Consommation d'énergies faible

En un seul mot, l'évolution des FPGAs commerciaux concerne l'amélioration du niveau de densité d'intégration qui est liée à l'évolution de la technologie microélectronique car nous sommes passés d'un niveau d'intégration de 2µm les premières familles XC2000 à 16 nm pour la dernières famille de la série UltratScale(KINTEX,VERTEX,ZYNO).

La révolution concerne l'intégration des mémoires, des multiplieurs, des processeurs, des circuits DSP dans un seul circuit FPGA. Cette évolution dans la structure des FPGAs a bien évidemment été accompagnée par une évolution des outils de développement. Il fallait en

effet des outils de plus en plus performants pour tirer profit de ces structures qui devenaient à la fois de plus en plus grandes, mais aussi de plus en plus hétérogènes [3].

I.2.7 Les circuits FPGAs de la série-7

La série 7 des circuits FPGAs comprend 3 familles de circuits : ARTIX7, KINTEX7 et VIRTEX7 comme le montre la figure I.2 Chacune de ces familles du circuit FPGA offre des performances en termes de nombre de cellules logiques, circuits DSP, transceivers (émetteurs-récepteurs : Ethernet), mémoire, nombre d'entrée sortie et tension d'utilisation.

	ARTIX-7	KINTEX-7	VIRTEX-7
	Lowest Power & Cost	Industry's Best Price/Performance	Industry's Highest System Performance
Logic Cells	20K – 355K	30K – 410K	285K – 2,000K
DSP Slices	40 – 700	120 – 1540	700 – 3,960
Max. Transceivers	4	16	80
Transceiver Performance	3.75Gbps	6.6Gbps 10.3Gbps	10.3Gbps 13.1Gbps 28Gbps
Memory Performance	800Mbps	2133Mbps	2133Mbps
Max. SelectIO™	450	500	1200
SelectIO™ Voltages	3.3V and below	3.3V and below 1.8V and below	3.3V and below 1.8V and below

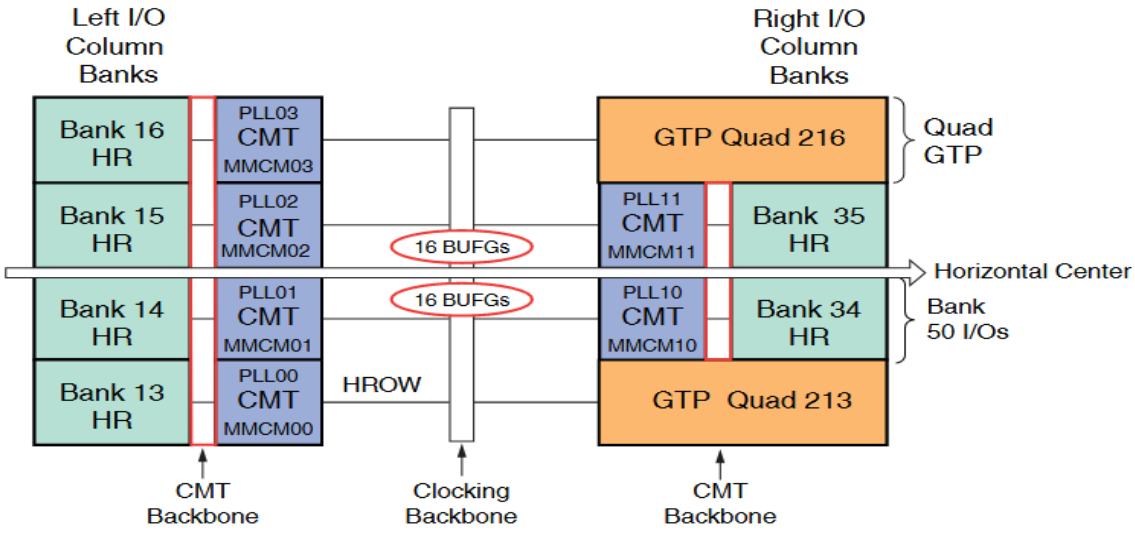
Figure I.2. Les différentes familles des circuits FPGAs série-7

Comparée aux circuits KINTEX et VIRTEX-7, la famille ARTIX-7 offre les meilleurs performances en terme de consommation de puissance et de coût. Les circuits FPGAs de la famille KINTEX-7 et VIRTEX-7 sont conçues pour être utilisées dans les applications industrielles. La famille VIRTEX-7 est utilisée dans les systèmes à très haute performances alors que la famille Kintex-7 offre un compromis entre le prix et les performances. Dans notre travail nous utilisons le circuit FPGA ARTIX-7 [1].

I.2.7.1. Architecture interne du circuit FPGA ARTIX7-100T

Dans ce paragraphe on présente l'architecture interne du circuit FPGA ARTIX-100T qui sera utilisé pour l'implémentation du nœud de capteur de notre projet. La figure I.3 montre les différents blocs de ce circuit [24] et qui sont :

- **Les BANKs**: chaque bank représente un bloc d'entré sortie (I/O) qui partagent la même source d'alimentation en général. Le circuit ARTIX-7 100T, possède les blocs suivants: **BANK13**, **BANK14, BANK15**, **BANK16**, **BANK35, BANK34**. Les pins I/O du BANKs 13 sont réservés par le constructeur et ne sont pas accessible. Le BANK 16 est partiellement accessible. Les pins I/O des autres BANKs sont accessibles par le concepteur.
- **Les CMT** : Blocs des circuits « Clock manager » : gestion de l'horloge
- **Les BUFG** : « Global Clock Buffer », buffer utilisé par l'outil dans le routage de l'horloge
- **GTP** : blocs « serial Transciever » Emetteur- Recepteur, non accessible au concepteur dans le circuit ARTIX-100T.



Figue I.3 LES BANKS d'un circuit FPGA –ARTIX7-100T

I.2.8 Différents domaines d'application des FPGAs

- Informatique : Périphériques spécialisés.
- Machinerie industrielle : Contrôleur pour machines.
- Télécommunications : Traitement d'images, Filtrage.

- Instrumentation : Équipement médical, Prototypage.
- Transport : Contrôle d'avions et métros.
- Aérospatiale : Satellites, Radar, la détection ou la surveillance…etc.

I.2.9 Les cinq principaux atouts de la technologie FPGA

- Performances.
- Coût.
- Fiabilité.
- Temps de mise sur le marché.
- Maintenance à long terme.

I.3 Les Systèmes sur puce Soc

Un système sur puce ou SoC (System-on-Chip), appelé encore système monopuce, désigne l'intégration d'un système complet sur une seule pièce de silicium, résultant de la cohabitation sur silicium de nombreuses fonctions déjà complexes en elles-mêmes : processeurs, DSP, mémoires, bus, convertisseurs, blocs analogiques, etc. Il doit comporter au minimum une unité de traitement de logiciel (une CPU) et doit dépendre d'aucun (ou de très peu) de composants externes pour exécuter sa tâche [1].

La figure I.4 présente un exemple du système monopuce typique. Il se compose d'un cœur de processeur (CPU), d'un processeur de signal numérique (DSP), de la mémoire embarquée, et de quelques périphériques tels qu'un DMA et un contrôleur d'E/S. Le CPU peut exécuter plusieurs tâches via l'intégration d'un OS. Le DSP est habituellement responsable de décharger le CPU en faisant le calcul numérique sur les signaux de provenance du convertisseur A/N [1].

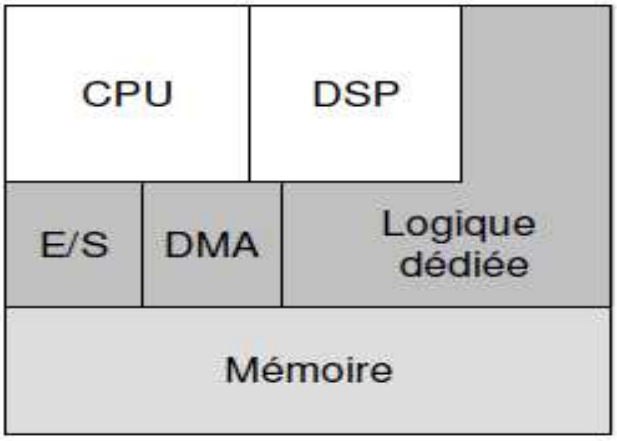


Figure I.4. Un exemple de système mono puce (SoC)

Le système pourrait être construit exclusivement de composants existants, ou d'une combinaison de composants existants et de solutions faites sur mesure

I.4 Le circuit imprimé, qu'est-ce que c'est ?

Le PCB (Printed Circuit Board) ou circuit imprimé est un support, la plupart du temps une plaque, qui sert à lier de manière électrique divers composants électroniques dans l'optique de concevoir un complexe électronique. Le circuit imprimé qui est également appelé carte électronique par bien de personnes comprend un assemblage de couches de cuivre très fines que l'on scinde avec un matériau isolant.

Afin de pouvoir obtenir tout un ensemble de pistes, l'on procède au gravage des couches de cuivre à partir d'un procédé chimique que l'on termine par la suite avec des pastilles. Pour que les pistes puissent par ailleurs être protégées de l'oxydation et des court-circuit, l'on recouvre le plus souvent le circuit imprimé avec une couche de vernis.

Il faut préciser ici que les pistes sont des éléments qui permettent de relier électriquement plusieurs parties du circuit imprimé. Dès qu'elles sont perforées, les pastilles permettent quant à elles d'établir une liaison électrique entre les couches de cuivre ou entre les différents composants qui ont été soudés dans le circuit imprimé. Dans certains cas, les pastilles qui n'ont pas été perforées peuvent permettre de procéder à la soudure des composants dont le montage a été effectué en surface.

I.4.1 Méthodologie de conception d'une carte électronique

La figure I.5 décrit le flot de conception d'un circuit imprimé (PCB). De manière générale Un projet PCB contient 2 documents :

_ Une schématique, décrivant le schéma électrique de l'application. Les composants électroniques sont représentés par des symboles et reliés par des interconnexions électriques.

_Un document PCB, décrivant le placement physique des composants sur la carte et

Le Routage des interconnexions

I.4.2 Flot de conception des circuits imprimés

Le flot utilisé dans la conception des PCB est constitué des phases suivantes [5] :

- ***La première phase***

Définition du schéma électrique du circuit en sélectionnant dans une librairie les différents symboles électriques des composants et les reliant. Ce schéma doit être complet et bien précis.

- ***La deuxième phase***

Attribution d'une empreinte pour chaque symbole électrique.

- ***La troisième phase***

Vérification des règles électriques an de détecter plusieurs erreurs telles qu'un pin d'entrée flottante, des connexions d'alimentations manquantes, connexion d'une broche d'alimentation vers un port de sortie etc. Cette vérification cation est réalisée en compilant le projet.

- ***La quatrième phase***

Transfert de toutes les empreintes des composants d'une manière aléatoire sur la surface de routage.

- ***La cinquième phase***

Définition du lay-out du PCB (le contour et la forme globale, les trous de fixation et placement des composants sur la carte en prenant en considération certains points comme :

- Les contraintes mécaniques : bords d'entrée, de sortie et d'alimentation.
- perturbations émises ou reçues.
- La taille, l'encombrement et le type des composants : hauteur et surface des composants.
- Le type des signaux à véhiculer : analogique, numérique.

- La sixième phase

Routage des interconnexions qui vont être matérialisées par des lignes métalliques en utilisant le mode de routage automatique ou manuel. Le routage est soumis à de nombreuses contraintes et règles telles que :

- l'isolation entre les conducteurs : distance minimale entre conducteurs (pistes, vias, pads).
- La largeur minimale et/ou maximale pour une piste selon le courant véhiculé.
- Les couches de cuivre nécessaires pour le routage et les couches spécifiques à des alimentations ou des plans de masse.
- Les dimensions des via

- La septième phase

Vérification des règles de routage.

- La huitième phase

génération des fichiers de sortie nécessaires à la fabrication de la carte comme le chier de format Gerber (un chier de contrôle numérique utilisé par un traceur de photo) qui décrit les couches de connexions électriques et le chier de format Excellons qui indique l'emplacement et la taille des trous de perçage. La figure (I.5) décrit le flot typique de conception d'un circuit imprimé [23]

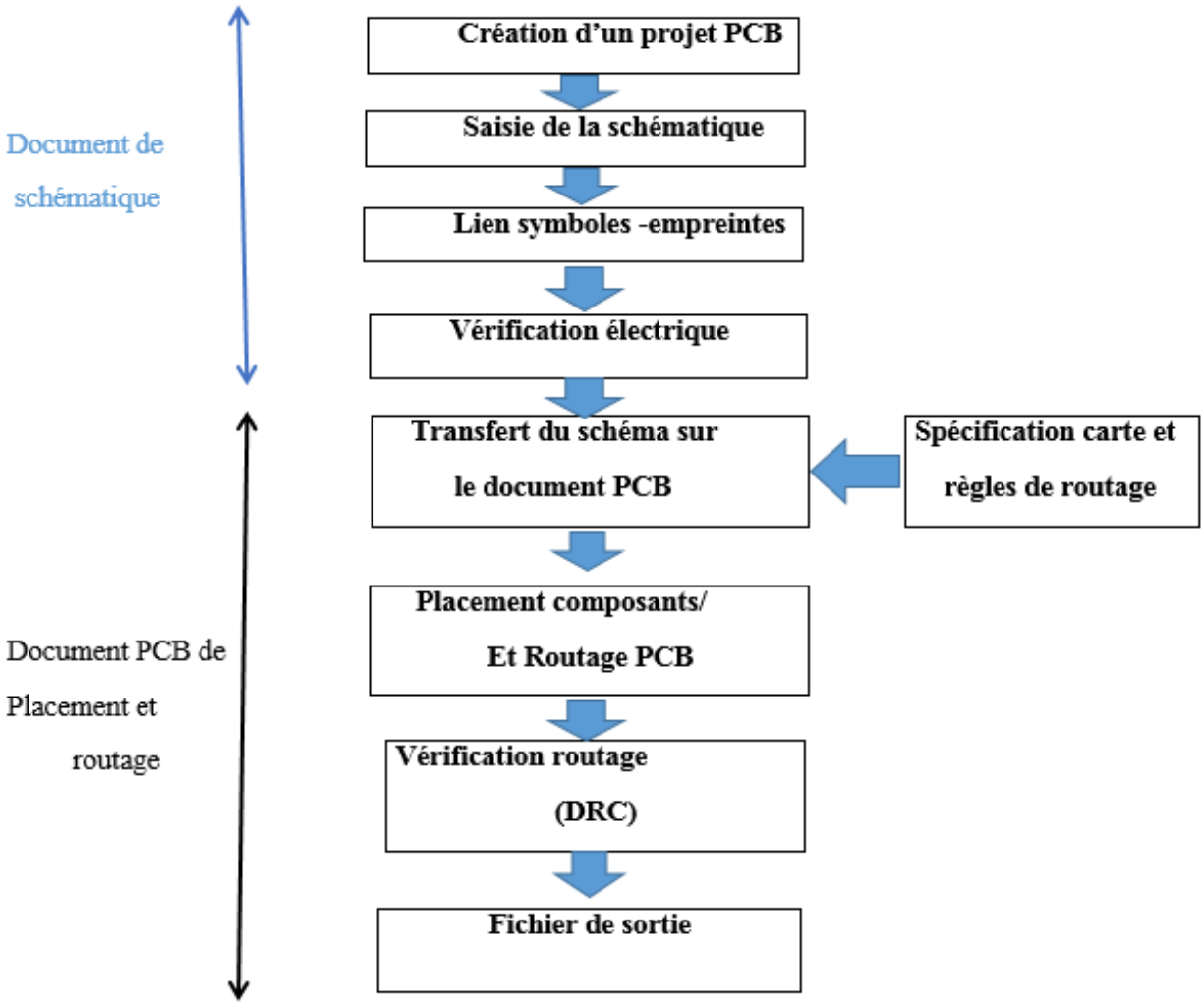


Figure I.5. Flot de conception d'un circuit imprimé (PCB)

I.5 Intégration des circuits FPGA dans un circuit imprimé

Afin d'intégrer un circuit FPGA dans un circuit imprimé, les concepteurs avaient tendance à utiliser le processus traditionnel de boîte noire, ils s'intéressaient seulement à la façon d'insérer le circuit sur la carte en ignorant son architecture. Cependant, les augmentations des performances et de la densité d'intégration du FPGA (le nombre de pins du FPGA varie entre 680 et 1760) combinées à la complexité croissante des systèmes électroniques, ont créé de nombreux dés de conception de systèmes basés sur FPGA.

Tous ces facteurs se traduisent par une complexité du PCB en termes d'augmentation des connections inter composants et du nombre des tensions d'alimentation, de placement

relatif des composants qui interfacent avec le circuit FPGA, et de routage etc. [23]. (Figure I.6)

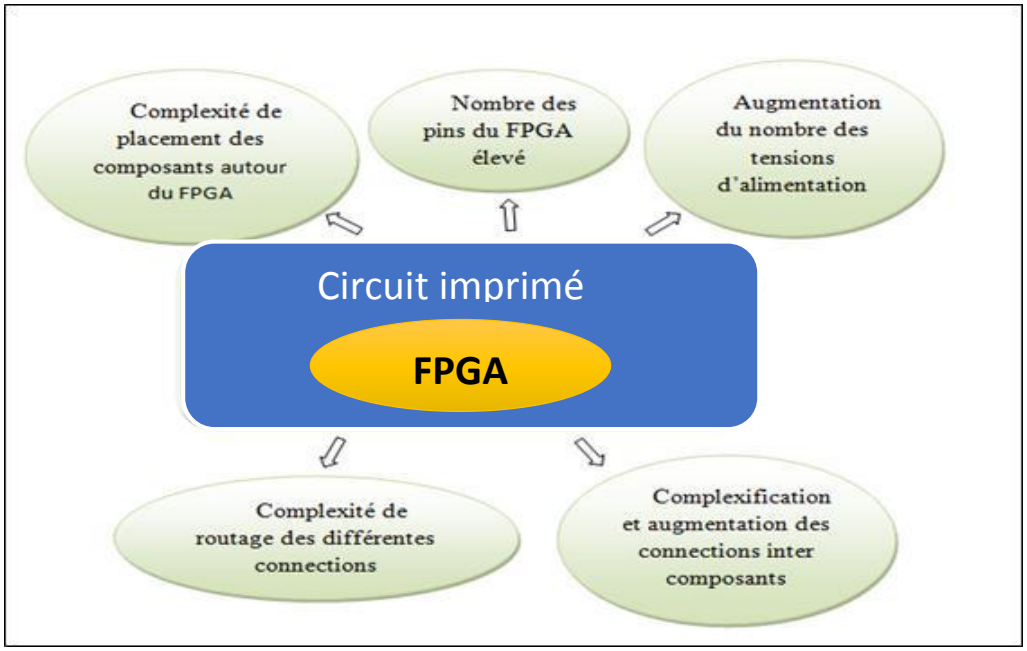


Figure I.6.montre la complexité associée à l'intégration d'un FPGA dans un PCB

I.6 Conclusion

En conclusion, Dans la première partie de ce chapitre, nous avons présenté les circuits reconfigurables d'une manière générale, Ensuite nous avons présenté les spécificités des architectures FPGA, Puis nous avons fait une description générale des circuits FPGAs en particulier et la description générale des systèmes sur puces.

Dans la deuxième partie de ce chapitre, nous avons présenté les circuits imprimés, la méthodologie de conception d'un PCB de manière générale ainsi que le défis auxquels sont confrontés les concepteurs lorsqu'ils intègrent un circuit FPGA, ce qui nécessité une méthodologie de conception plus appropriée. Cette dernière sera présentée dans le chapitre

Dans le prochain chapitre, on présentera des généralités et un état de l'art sur les nœuds de capteurs basés sur FPGA.

Chapitre II Nœuds de capteur basés sur FPGA

II.1 Introduction

Dans ce chapitre nous allons présenter l'architecture générale des nœuds de capteur ainsi leur domaine d'application. Nous nous intéressons par la suite au développement des plateformes FPGAs dans les nœuds de capteurs et les solutions qu'elles peuvent offrir pour répondre aux défis actuels.

II.2 Présentation de l'architecture matérielle d'un nœud de Capteur

Un nœud de capteurs est un système composé d'un microprocesseur/microcontrôleur, d'un ou plusieurs capteurs, ainsi que d'une interface de communication. Les tâches d'un nœud de capteurs dans le milieu industriel sont très variées. Il mesure des données liées à un procédé et son environnement tels que températures, accélérations ou pression, il effectue un prétraitement de celles-ci et les transmet à un automate ou à une passerelle.

D'un point de vue architecturelle un nœud de capteur contient quatre unités de base : une l'unité de captage, une l'unité de traitement, une l'unité de transmission, et une l'unité de contrôle d'énergie. Selon le domaine d'application, il peut aussi contenir des modules supplémentaires tels qu'un système de localisation GPS, ou bien un système générateur d'énergie (cellule solaire).la figure II.2 montre l'architecture générale d'un nœud de capteur.

Un RCSF est composé d'un ensemble de nœuds capteurs. Ces nœud capteurs sont organisés en champs « sensor fields » voir figure II.1. Chacun de ces nœuds a la capacité de collecter des données et de les transférer au nœud passerelle (dit "sink" en anglais ou puits) par l'intermédiaire d'une architecture multi-sauts. Le puits transmet ensuite ces données par Internet ou par satellite à l'ordinateur central «Gestionnaire de taches» pour analyser ces donner et prendre des décisions.

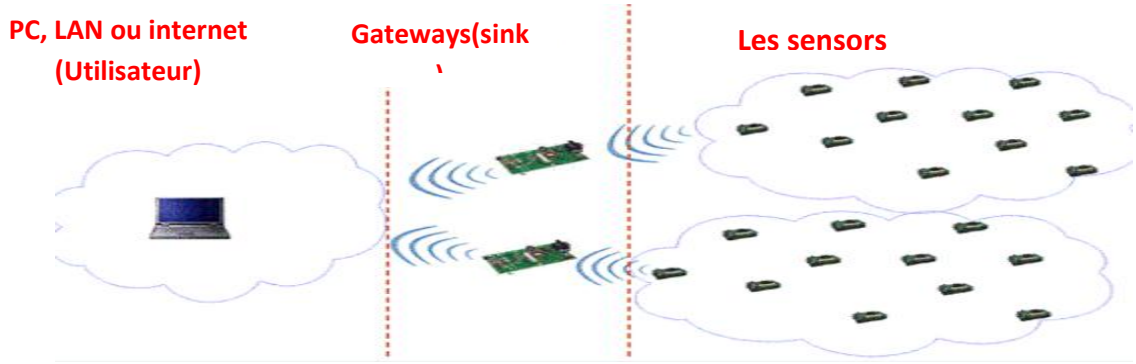


Figure II.1. Architecture d'un RCSF

II.2.1 Unité de captage (capteur)

Le capteur est généralement composé de deux sous-unités : le récepteur et le transducteur. Il fournit des signaux analogiques, basés sur le phénomène observé, au convertisseur Analogique/Numérique(CAN). Ce dernier transforme ces signaux en un signal numérique compréhensible par l'unité de traitement. Un nœud de capteur peut avoir un ou plusieurs unités de captage [4].

II.2.2 Unité de traitement (processeur)

C'est l'unité qui nous intéresse le plus dans notre travail, cette unité recueille des données de l'unité de captage ; elle effectue un traitement sur ces données (si nécessaire) et décide quand et où les envoyer. Elle doit exécuter différents programmes et protocoles de communication. Les types des unités de traitement qui peuvent être utilisés dans un nœud de capteur incluent les microcontrôleurs, les circuits DSPs (Digital Signal Processors), les circuits FPGAs et les ASICs (Application Specific Integrated Circuit) [5].

Parmi toutes ces alternatives, le microcontrôleur a été le processeur le plus utilisé dans les nœuds de capteurs à cause de son bon prix et sa faible consommation énergétique, cependant ces dernières années les principaux fournisseurs des FPGAs se sont concentrés sur la minimisation de la consommation d'énergie et la réduction des prix comme nous verrons dans la troisième section de ce chapitre.

II.2.3 Unité de transmission

Elle fait toutes les émissions et réceptions des données sur un médium. Elle peut être de type optique, ou de type radiofréquence.

- Les communications de type optique sont robustes vis-à-vis des interférences électriques. Néanmoins, ne pouvant pas établir de liaisons à travers des obstacles, elles présentent l'inconvénient d'exiger une ligne de vue permanente entre les entités communicantes.
- Les unités de transmission de type radiofréquence comprennent des circuits de modulation/démodulation, filtrage et multiplexage ; ceci implique une augmentation de la complexité et du coût de production du micro-capteur [4].

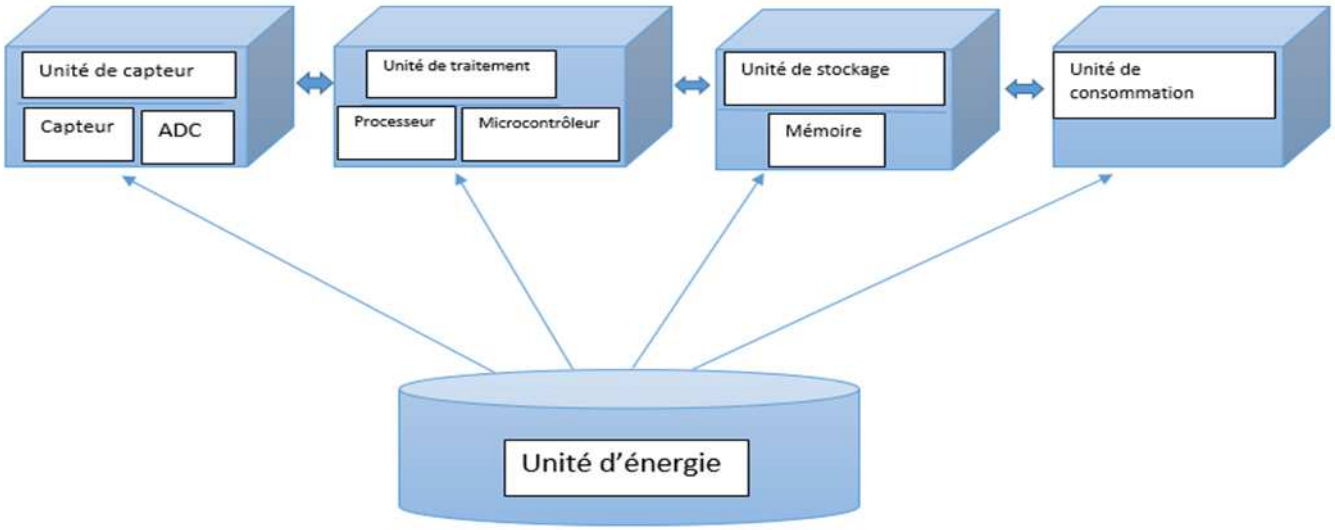


Figure II.2. Architecture matérielle d'un nœud de capteur

II.2.4 Unité de contrôle d'énergie

Un micro-capteur est muni d'une ressource énergétique (généralement une batterie). Étant donné sa petite taille, cette ressource énergétique est limitée et généralement non-remplaçable. Ceci fait souvent de l'énergie la ressource la plus précieuse d'un réseau de capteurs, car elle influe directement sur la durée de vie des micro-capteurs et donc du réseau entier. L'unité de contrôle d'énergie constitue donc une partie essentielle du système [4].

II.2.5 Unité de stockage

L'unité de stockage inclut la mémoire de programme (dont les instructions sont exécutées par le processeur) et la mémoire de données (pour conserver des données fournies par l'unité de captage et d'autres donnés locales).la talle de cette mémoire est limitée essentiellement par les considérons économiques et s'améliorera aussi probablement au fil des années [5].

II.3 Les Applications typiques des nœuds de capteur

La plus grande excitation à propos des nœuds de capteurs provient de l'utilisation d'un grand nombre de nœuds qui communiquent entre eux et forment des réseaux sans fil (RCSF).La miniaturisation, l'adaptabilité, le faible cout et la communication sans fil permettent à ces réseaux de capteur d'envahir plusieurs domaines d'application. Voici quelque exemple d'application potentielle dans différent domaines : la figure II.3 montre les déférentes applications possibles des nœuds de capteur.

- **Applications militaires :** Comme dans le cas de la majorité des technologies, le domaine militaire a été un moteur initial pour le développement des réseaux de capteurs. Des tests concluants ont déjà été réalisés dans ce domaine par l'armée américaine dans le désert de Californie, le projet DSN (Distributed Sensor Network) au DARPA (Defense Advanced Research Projects Agency) était l'un des premiers projets dans les années 80 ayant utilisés les réseaux de capteurs pour rassembler des données distribuées. Un réseau de capteurs peut être déployé dans un endroit stratégique ou hostile, afin de surveiller les mouvements des forces ennemies, ou analyser le terrain avant d'y envoyer des troupes (détection d'intrusions, détection des armes chimiques, biologiques ou radiations nucléaires) [7].

- **Applications liés à la sécurité :** Les structures d'avions, navires, automobiles, métros, etc. pourraient être suivies en temps réel par des réseaux de capteurs, de même que les réseaux de circulation ou de distribution de l'énergie. Les altérations de structure d'un bâtiment, d'une route, d'un quai, d'une voie ferrée, d'un pont ou d'un barrage hydroélectrique pourraient être détectées par des capteurs préalablement intégrés dans les murs ou dans le béton, sans alimentation électrique ni connexions filaires[6].

Certains capteurs ne s'activant que périodiquement peuvent fonctionner durant des années, voire des décennies. Un réseau de capteurs de mouvements peut constituer un système d'alarme distribué qui servira à détecter les intrusions sur un large secteur. Déconnecter le système ne serait plus aussi simple, puisqu'il n'existe pas de point critique. La surveillance de routes ou voies ferrées pour prévenir des accidents avec des animaux ou des êtres humains ou entre plusieurs véhicules est une des applications envisagées des réseaux de capteurs. [6].

Figure II.3. Exemples d'applications des RCSF [9-12]

- **Applications médicales :** Dans l'application de la médecine, les réseaux de capteur peuvent être utilisés pour assurer une surveillance permanente des organes vitaux de l'être humaine grâce à des micro-capteurs qui pourront être avalés au implantés sous la peau. Ils peuvent aussi faciliter le diagnostic de quelque maladies en effectuant des mesures physiologiques telles que : la tentions artérielle, battements du cœur, l'aide des capteurs ayant chacun une tache bien particulière.

Les données physiologiques collectées par les capteurs peuvent être stockées pendant une longue durée pour le suivi d'un patient pour une ultérieure décision médicale [8].

- **Applications sportifs :** l'évolution des réseaux de capteurs est utilisée de plus en plus dans le domaine sportif, à savoir les systèmes de surveillance, les systèmes de calcule de trajectoires (comme dans le tennis), systèmes de détection d'erreur d'arbitrage (comme dans le football indiquent si le balle a franchi la ligne de but) [7].
- **Applications commerciales :** Il est possible d'intégrer des nœuds de capteur au processus de stockage et de livraison. Le réseau pourra être utilisé pour connaitre la position, état de la direction d'un paquet ou d'une cargaison. Un client attendant un paquet peut alors avoir un avis de livraison en temps réel et connaître la position du paquet. Des entreprises manufacturières, via des réseaux de capteurs pourraient suivre le procédé de production à partir des matières premières jusqu'au produit final livré. Grâce aux réseaux de capteurs, les entreprises pourraient offrir une meilleure qualité de service tout en réduisant leurs coûts. Les produits en fin de vie pourraient être mieux démontés et recyclés ou réutilisés si les micro-capteurs en garantissent le bon état [8].
- **Applications domestiques :** Le déploiement des capteurs de mouvement et de température dans les futures maisons dites intelligentes permet d'automatiser plusieurs opérations domestiques telles que : la lumière qui s'éteint et la musique qui se met en état d'arrêt quand la chambre est vide, la climatisation et le chauffage s'ajustent selon les points multiples de mesure, le déclenchement d'une alarme par le capteur anti-intrusion quand un intrus veut accéder à la maison [8].
- **Applications environnementales :**les réseau de capteur peuvent être utilisés pour surveiller les changement environnementaux ,ils servent à déterminer les valeurs de certains paramètres a un endroit donné ,comme par exemple :la température ,la

Pression atmosphériques a un endroit donné, comme par exemple : la température, la pression atmosphérique, etc.

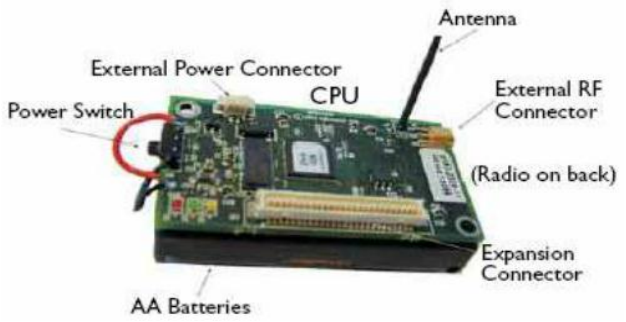
En dispersent des nœuds de capteur dabs la nature, on peut détecter des événements tels que des feux de forêts, des tempêtes au des inondations, ceci permet une intervention beaucoup plus rapide et efficace des secours. Dans le domaine de l'agriculture, les capteurs peuvent être utilisés pour réagir convenablement aux changements climatiques, par exemple en déclenchant le processus d'arrosage lors de la détection de zones sèches dans un champ agricole [8].

II.4 Etat de l'art sur la technologie des nœuds de capteurs

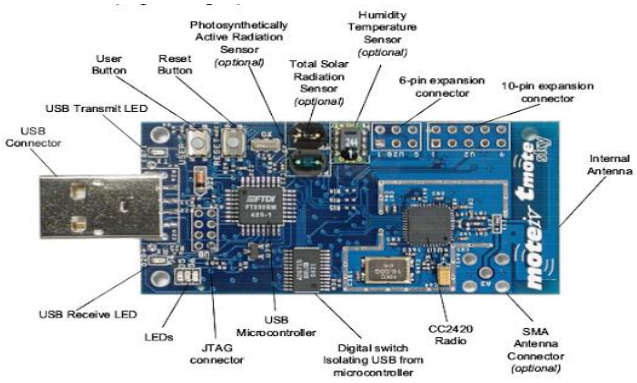
Les nœuds de capteurs peuvent être implémentés à base de microcontrôleurs, de processeurs de circuits FPGA ou bien sur des ASICs (Application Specific Integrated circuit)

II.4.1 Nœuds de capteur basé sur l'utilisation Microcontrôleur/ Microprocessus

Les microcontrôleurs et microprocesseurs ont été largement utilisés dans l'implémentation hardware des nœuds de capteurs et pour une variété d'applications [13]. La technologie des microcontrôleurs a permis aux concepteurs et utilisateurs une option simple pour réaliser leur nœud de capteur (WSN) et monter des applications appropriées. La plupart des plateformes WSN antérieures utilisaient un microcontrôleur comme contrôleur principal pour effectuer plusieurs tâches de traitement des données issues du capteur et pour la gestion de la consommation d'énergie. La Figure II.4, montre quelques exemples de plateformes de nœud de capteurs commercialisés et le tableau II.1, montre les différentes performances de ces derniers [14-18].



MICA [16]



TMOTE-SKY [18]

IMOTE [19]

TELOS [20]

Figure II.4. Exemples de nœuds de capteurs basés sur microcontrôleurs/ microprocessus

A titre d'exemple, la famille de nœuds de capteurs, MICA, est équipée d'un microcontrôleur Atmel de 8 bits, d'un flash programmable et d'un module de communication basé sur la modulation d'amplitude (TR1000 ASK), la fréquence est de 8Mhz alors que la puissance consommée ne dépasse pas les 27mW. Par contre le nœud de capteur MICA2 possède des performances meilleures en termes de fréquence (8 Mhz) et une communication basée sur la modulation de fréquence (CR1000, FSK), mais il consomme plus de puissance que son prédécesseur (89 mW).

Les plateformes TMOTE-SKY et TELOS sont basées sur l'utilisation du microcontrôleur MSP430 du fabriquant Texas-Instrument, avec une fréquence de 8 Mhz, un circuit de communication basé sur la modulation de phase (CC2420, OQPSK), et une puissance de 32 mW. Par ailleurs, la plateforme IMOTE2 utilise le microprocesseur Intel-PXA271, une communication basée sur la modulation de phase (CC2420, OQPSK), une fréquence entre 13-400Mhz. Cependant, la puissance consommée est supérieure aux autres nœuds de capteurs (86mW) [13].

Tableau II.1. Nœuds de capteurs basés sur l'utilisation des Microcontrôleurs et Microprocesseurs [13]

Plateforme	Microcontrôleur/ Microprocesseur	capteurs	Fréquence (Mhz)	Communication	Puissance (mW)
MICA [7]	Microcontrôleur Atmel 8bits	Température, humidité, accéléromètre, etc.	4	TR1000, ASK	27
MICA2 [8]	Microcontrôleur Atmel 8bits	Température, humidité, accéléromètre, lumière	8	CR1000, FSK	89
TMOTE SKY [9]	Microcontrôleur TiMSP430		8	CC2420, OQPSK	32
TELOS [10]	Microcontrôleur TiMSP430		8	CC2420, OQPSK	32
IMOTE2 [11]	Microprocesseur Intel PXA271		13-400	CC2420, OQPSK	86
TSgaTe	Microcontrolleur 32 bits ARM Cortex@M3		72		-

De manière générale, les nœuds de capteurs basés sur microcontrôleurs ou microprocesseurs, sont simples à réaliser et consomment moins de puissance, néanmoins l'un des inconvénients c'est la faible fréquence ce qui pose un problème pour les applications utilisant des algorithmes complexes et nécessitant des performances en temps de réponse.

II.4.2 Nœud de capteurs basés sur l'utilisation des circuits FPGA

La demande de capteurs intelligents flexibles, de petite taille, forte densité d'intégration, haute précision, faible consommation, constitue une tendance actuelle du marché. Par ailleurs, l'intégration des fonctions intelligentes implique des exigences supplémentaires en temps de calcul. C'est pour cette raison que les circuits FPGA, sont considérés comme des candidats potentiels et incontournables aux défis actuels et futurs dans les applications nécessitant l'utilisation des nœuds de capteurs [Projet socio-économique NCIR.

Dans [21], les auteurs ont présenté une revue des différents capteurs basés sur FPGA. Ils montrent que la plate-forme Xilinx Artix-7, a une consommation de puissance statique similaire à celle de l'ancienne carte Spartan-3. En plus, elle offre de grandes performances en terme de surface et de mémoire (Bloc RAM et LUT) (Tableau II.2)

Tableau II.2 **:** Consommation d'énergie et tension de fonctionnement des cartes Xilinx [1]

Plate-forme	modèle	Tension (V)	Nombre de LUTs	Block RAM	Consommation d'énergie statique (mW)
Spartan-3	XC3S200	1.2	4,320	216	41
Spartan-3E	XC3S250E	1.2	5,508	216	51
Spartan-6	XC6SLX100	1.2	101,261	4,824	67
Virtex-4	XC4VLX200	1.2	200,448	6,048	1,278
Virtex-5	XC5VLX220	1	138,240	6,912	1,985
Virtex-6	XC6VLX240T	1	241,152	14,976	1,977
Virtex-7	XC7VX330T	1	326,400	27,000	141
Kintex-7	XC7K160T	1	162,240	11,700	74
Artix-7	XC7A100T	1	101,440	4,860	41

Par ailleurs, le tableau II.3, montre que la consommation d'énergie statique de la série Altera V est supérieure à celle de la série 7 de Xilinx.

Tableau II.3. Consommation d'énergie et tension de fonctionnement des cartes Altera [1]

Plate-forme	modèle	Tension (V)	Nombre de LUTs	Block RAM	Consommation d'énergie statique (mW)
Cyclone	EP1C6	1.5	5,980	92	60
Cyclone II	EP2C8	1.5	8,256	165	40
Cyclone V	5CEFA9	1.10	301,000	12,200	206
Stratix	EP1S25	1.5	25,660	635	450
Stratix II	EP230	1.5	33,880	663	86
Stratix V	5SRR9	1	840,000	12,800	880
Arria V	57GXMA1D	1	75,000	8,000	197

Les résultats des tableaux II.2 et II.3, montre que la plateforme de Xilinx à base du circuit FPGA ARTIX-7, offre de meilleurs performances par rapport aux circuits FPGA de XILINX et aussi comparé aux autres technologies, comme celle de ALTERA, CYCLONE et STARTIX [1]

II.5 Conclusion

Dans ce chapitre, nous avons constaté que l'utilisation des plateformes FPGAs, particulièrement ceux de Xilinx ARTIX-7 est promotrice pour faire face aux différents défis actuels des nœuds de capteurs.

Notamment que les fournisseurs commencent à proposer des plates-formes optimisées, pour la consommation d'énergie, et capables de fonctionner à très faible puissance avec un prix réduit, ce qui permet aux nœuds de capteurs de bénéficier d'une capacité de traitement supplémentaire, pour répondre avec défis auxquels font fasses l'application actuelles.

Dans le prochain chapitre, nous allons présenter l'implémentation matérielle adoptée pour la conception du nœud de capteur sur la carte Nexys-4 DDR, à base du circuit FPGA- Artix-7.

Chapitre III Implémentation Matérielle

III.1 Introduction

Dans ce chapitre, nous allons présenter l'architecture hardware de l'unité de traitement du nœud de capteur et son implémentation sur FPGA .Il est important de signaler que l'architecture de base liée à l'unité de traitement est un système sur puce (SOC) construit autour du processeur NEO430 et de certains composants IPs. Ces derniers sont du domaine publique (opensource/opencores) et leur code VHDL est téléchargeable gratuitement.

Afin de valider la fonctionnalité de notre (SOC), nous avons ciblé deux modes d'utilisation correspondant à deux applications différentes :

- Le premier mode concerne l'utilisation de notre SOC avec un capteur de température Permettant de mesurer et d'afficher la température externe de l'environnement

- Le deuxième mode concerne l'utilisation de notre SOC avec un accéléromètre afin de mesurer et d'afficher l'accélération sur trois axes (X, Y, Z).

Nous commencerons d'abord par présenter la synoptique générale et l'architecture du système liée à l'implémentation du nœud du capteur sur FPGA. Ensuite, nous présenterons la méthodologie de conception sur FPGA, l'architecture interne détaillée du SOC en question, ainsi que les résultats de synthèse et d'implémentation sur FPGA.

III.2 Synoptique générale du nœud de capteur sur FPGA

La figure III.1 représente la synoptique générale de l'implémentation du nœud de capteur sur FPGA. La partie digitale correspondant à l'unité de traitement de notre nœud de capteur, est un SOC implémenté au sien du circuit FPGA Artix7 de Xilinx. Le circuit FPGA communique avec deux capteurs numériques : un capteur de température ADT7420 pour acquérir la température de l'environnement extérieur et un accéléromètre ADXL362 pour acquérir l'accélération sur les trois axes [1].

Un afficheur 7 segments est utilisé pour afficher la valeur de la température et de l'accélération totale.

Le circuit FPGA Artix 7, le capteur de température ADT7420, l'accéléromètre ADXL362 et l'afficheur 7 segments sont tous des composants implantés dans la carte de développement Nexys4 DDR de Digilent [1].

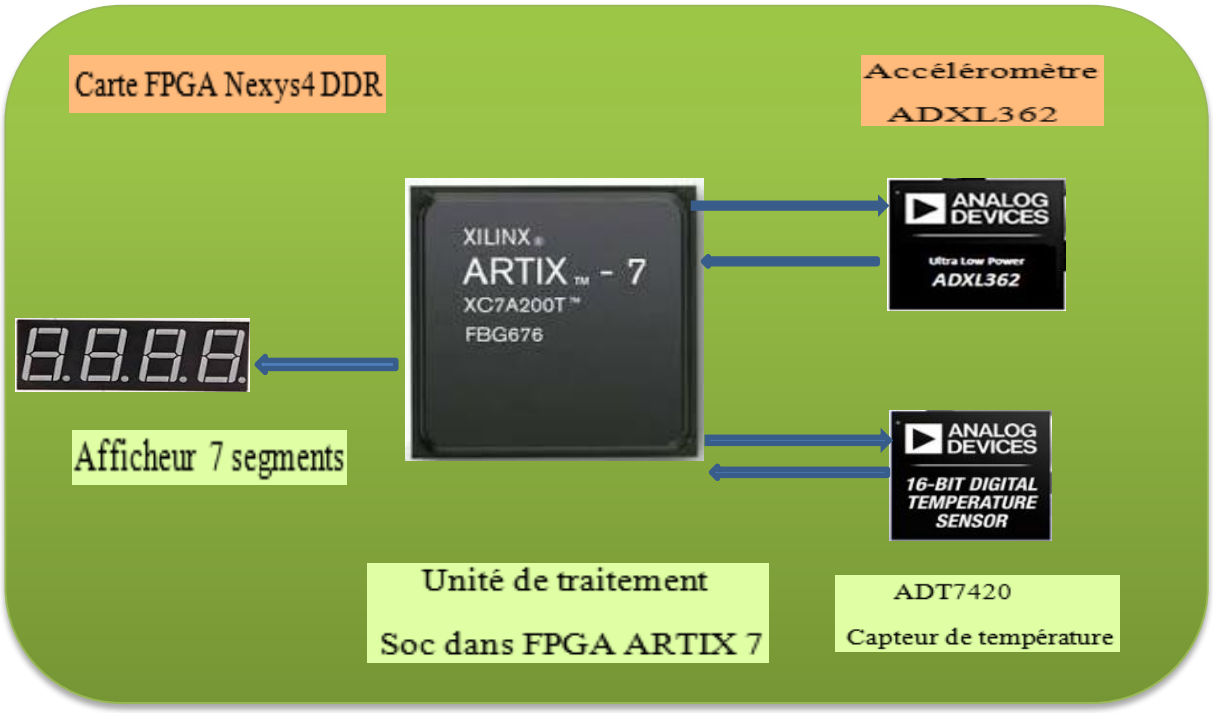


Figure III.1.synoptique général de l'implémentation sur FPGA du nœud de capteur

III.3 Méthodologie de conception du SOC sur FPGA

La figure III.2 montre le flot de conception adopté pour l'implémentation hardware du SOC. D'abord, on commence par spécifier la fonction générale du système comme décrit dans la section précédente, la prochaine étape serait de concevoir notre système sur puce (SOC) à travers la description VHDL et de le relier avec les autre composants (laptop, capteurs et afficheur 7 segments)

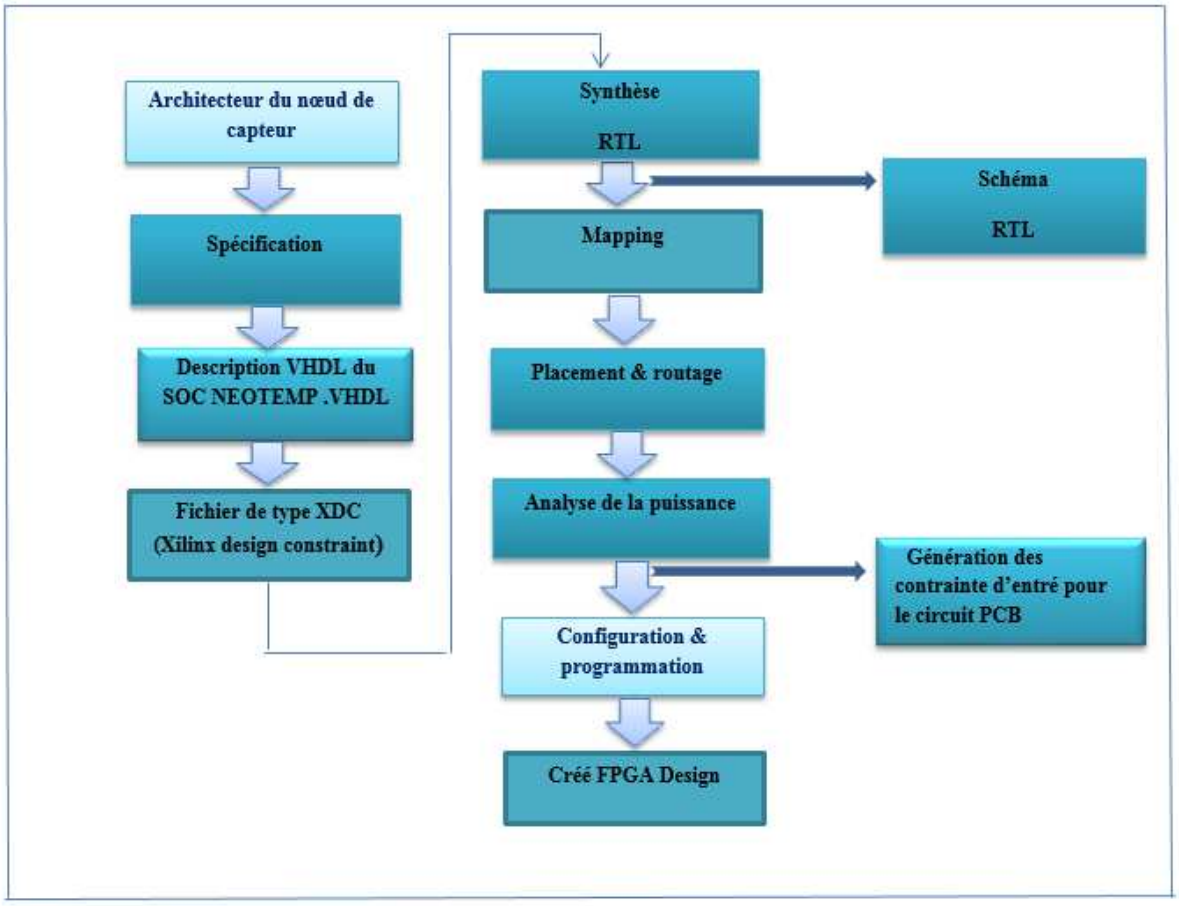


Figure III.2 flot de conception

III.4 Présentation de l'architecteur interne détaillée du système

La figure III.3 représente l'architecture interne détaillée du système, constitué principalement du système NEO430 et de l'interface série TempSensorCtl. Cette dernière assure la communication entre le capteur de température ADT7420 et le système NEO430. La communication avec l'accéléromètre est assurée moyennant le module SPI du système NEO430. La communication avec l'afficheur 7 segments est assurée moyennant la sortie PIO Et la communication avec le PC est assurée Moyennant l'UART (Universal, assynchronous Receiver transmitter) [1].

Le système à base du processeur NEO430 et l'interface TempSensorCtl forment un système sur puce SOC. L'outil Vivado de Xlinix est utilisé pour synthèse et l'implémentation de ce système sur le circuit FPGA Artix-7 de Xilinx hébergé sur la carte Nexys 4 DDR de diligent [1].

Dans la suite de cette section, nous présenterons les différents éléments du système.

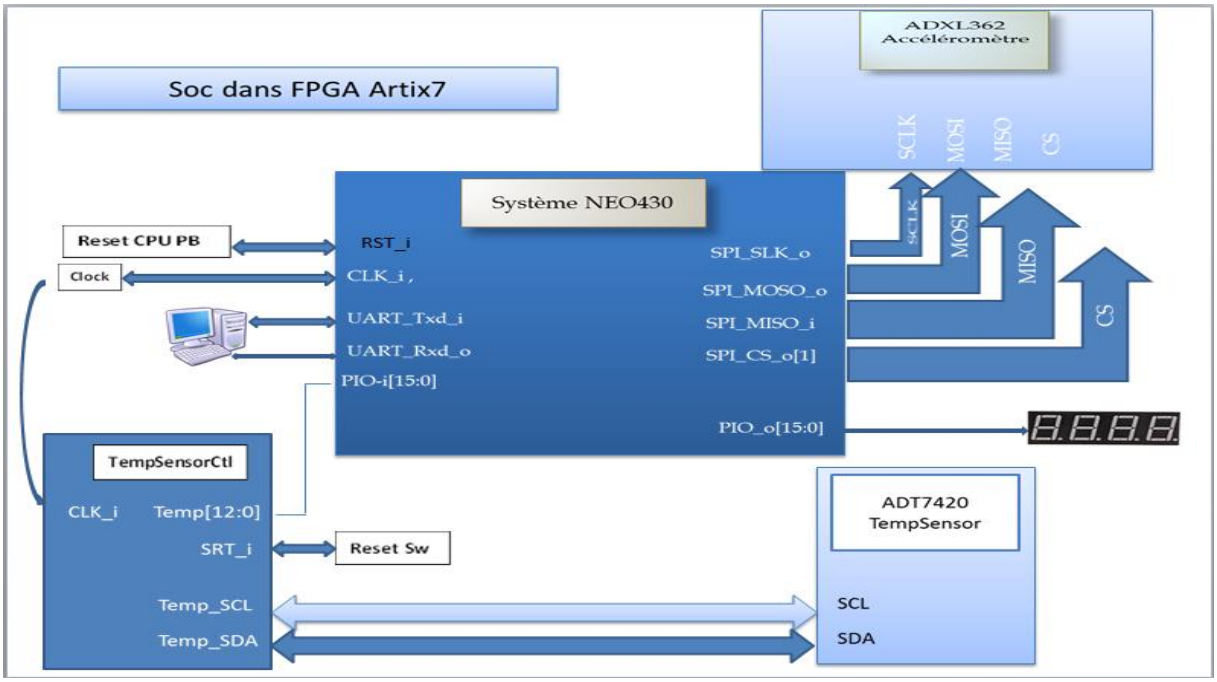


Figure III.3.présentation de l'architecteur hardware du système implémenté sur FPGA

III.4.1 Le système NEO430

Le système à processeur NEO430 est constitué de plusieurs modules comme le montre la figure III.4

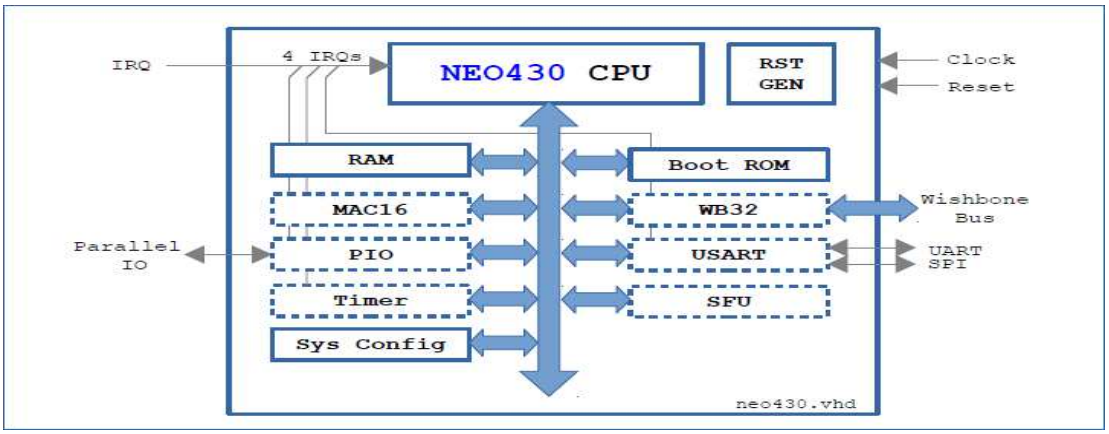


Figure III.4.Le système NEO430Stephan Nolting, « NEO430 Processor », Juillet 2016[20]

- **NEO430CPU** : Le NEO430 est un processeur soft open source 16 bits RISC décrit en VHDL qui utilise une seule mémoire pour les données et instruction (Architecture Va Neumann) et qui est aussi 100% compatible avec le jeu d'instruction du fameux MSP430 de Texas Instrument. C'est le cœur du système sur puce, il supporte aussi tous les modes d'adressages du MSP430. Il inclut une unité de contrôle, une unité arithmétique et logique (ALU), un générateur d'adresse et un groupe de registre [20]

- **Mémoire interne RAM (Component VHDL memory.vhd) :** Elle sert comme une mémoire commune pour les données et les instructions du programme exécuté. La taille de cette mémoire peut être configurée en utilisant le fichier package comme le montre la capture ci-dessous. La taille maximale autorisée est de 60KB.

```
69  constant mem_size_c : natural := 8*1024; -- bytes, max 60kB
```

Etant donné que cette mémoire est volatile, elle ne peut pas être utilisée pour garder le programme d'application de façon permanente, par conséquent on utilise le bootloader pour copier une image du programme à partir d'une ROM ajoutée au processeur comme on va voir par la suite.

- **Boot ROM (Component VHDL boot_rom.vhd):** Comme son nom déjà indique, le boot Rom contient l'image du bootloader en lecture seul qui est exécuté juste après le redémarrage du système. La taille de cette mémoire est aussi configurable à travers le fichier packa (Capture ci-dessous)

```
-- Boot ROM images (max size: 2048 bytes) --
constant std_boot_size_c      : natural := 2048;
```

- **Unité multiplicateur-accumulateur (MAC 16) (Component VHDL mac16.vhd)) :** Unité optionnelle qui performe des opérations de multiplications suivies par additions sur des donnés de 16 bits**.**

- **Interface bus Wishbone (WB32) (Component VHDL wb_interface.vhd) : C'est** un module d'interface qui utilise des bus d'adresses et données de 32bit pour ajouter d'autre IPs au système. Ce module peut être exclus du design en utilisant le fichier package [20]

- **ROM Wishbone (Component VHDL wb_mem.vhd) :**

Cette mémoire est ajoutée à travers l'interface Wishbone afin de stocker l'image du programme d'application.

- **Ports d'entrés sorties parallèle (PIO) (Component VHDL parallel_io.vhd) :**

Unité qui offre un port d'entré de 16 bits et un port de sortie de 16 bits pour communiquer avec le monde extérieur**.** Dans notre système les lignes d'entrés de ce module sont utilisées pour acquérir l'information de la température et lignes de sorties pour animer les afficheurs 7 segments. Cette unité peut être exclus durant la synthèse du système en utilisant le fichier package [20]

- **USART (Universal Synchronous Asynchronous Receiver and Transmitter/Component VHDL usart.vhd):**

Offre une interface de communication série entre le système et le monde extérieur. Deux sous modules sont implémentés : un sous module **UART** et un sous module **SPI**

(Serial Peripheral Interface) qui peuvent fonctionner en parallèle. Le module **UART** est utilisé pour communiquer avec l'ordinateur à travers un port série pour la programmation du processeur et la lecture des données par contre le module **SPI** est utilisé pour communiquer avec l'accéléromètre ADXL362 (voir figure III.4). Par default cette unité est toujours synthétisée mais elle peut être désactivée en utilisant le fichier package [20]

- **Temporisateur (Timer) (Component VHDL timer.vhd) :** Module nécessaire pour la majorité des applications temps réel, il génère des interruptions dans des intervalles de temps bien spécifiés. Cette unité est nécessaire pour le bon fonctionnement du bootloader mais elle peut toujours être exclus de l'implémentation en utilisant le fichier package [20].

- **Unité de Fonctions Spéciales (SFU) (Component VHDL sfu.vhd) :** Implémente des fonctions spéciales qui peuvent être utile pour des applications spécifiques. Toutes ces fonctions sont implémentées en hardware parce que leur implémentation en software nécessite beaucoup d'instructions comme compter le nombre de bit à '1'. Cette unité peut aussi être désactivée à travers le fichier package [20].

- **Module de configuration système (Component VHDL sysconfig.vhd) :** Donne accès à plusieurs informations du système et il permet aussi la configuration des vecteurs d'interruption [20].

III.4.2 Capteur de température ADT7420

L'ADT7420 hébergé sur la carte Nexys4 DDR est un capteur de température numérique de haute précision offrant des performances de pointe sur une large gamme industrielle, logés dans un boîtier de 4mm×4mm. Il contient un capteur de température et un CAN (Convertisseur Analogique Numérique) de 16 bits pour surveiller et numériser la température à une résolution de 0,0078 ° C. La résolution du CAN, par défaut, est réglée sur 13 bits (0.0625 °C) et c'est la résolution utilisée pour notre implémentation. La résolution du CAN est un paramètre programmable par l'utilisateur qui peut être modifié via l'interface série [21].

Le contrôle de l'ADT7420 s'effectue via l'interface série compatible I2C représenté par **TempSensorCtl** dans la figure III.3. L'ADT7420 est connecté à ce bus en esclave et est sous le contrôle d'un appareil maître.

Le schéma de la figure III.5 représente les différentes connections d'interface conseillé par le constructeur et implémenté sur la carte Nexys4.

Comme toutes les stations compatibles I2C, l'ADT7420 utilise 7 bits d'adresse, les 5 MSBs de cette adresse sont liés physiquement à 10010 à l'intérieur du capteur. Les pins A0 et A1 représentent les 2 LSBs de cette adresse ce qui nous donne le choix entre 4 adresses possibles. Sur la carte Nexys4, ces deux pins sont liés au Vcc ce qui veut dire que l'adresse du capteur est fixée à 1001011 [21].

Les pins **SCL** et **SDA** représentent le **serial clock** et **serial data** respectivement de la communication I2C. Le pin **CT** (Critical Température) s'active quand une température critique configurée par l'utilisateur est atteinte et le pin d'interruption **INT** s'active quand la température dépasse un seuil maximum prédéfini ou chute en dessous un seuil minimum prédéfini.

Les pins **A0**, **A1**, **SCL**, **SDA**, **CT** et **INT** sont du type open-drain, donc des résistances de pull-up sont nécessaires pour leurs fonctionnements comme le montre la figure III.5.

L'ADT7420 contient plusieurs registres qui permettent la configuration du capteur, la lecture de température, la spécification des seuils de température pour générer des interruptions et des alarmes. Chacun de ces registres est accessible avec une adresse unique [21].

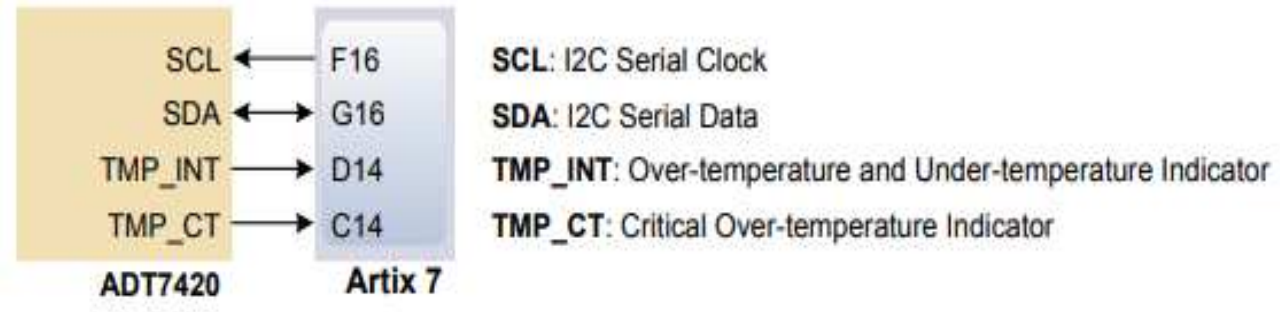


Figure III.5. Interface de capteur de température

III.4.3 L'accéléromètre ADXL362

L'ADXL362 est un accéléromètre MEMS à 3 axes à puissance ultra-faible qui consomme moins de 2 µA à un débit de sortie de 100 Hz et 270 nA en mouvement déclenché le mode de réveil. Contrairement à accéléromètres utilisant des cycles de puissance pour atteindre une puissance faible consommation, l'ADXL362 n'alias pas les signaux d'entrée par sous-échantillonnage, il échantillonne la totalité de la bande passante du capteur débits de données.L'ADXL362 fournit toujours une résolution de sortie de 12 bits ,8 bits Des données formatées sont également fournies pour une efficacité accrue sur un seul octet. Transferts quand une résolution inférieure est suffisante. [21]

Figure III.6. Interface de L'accéléromètre ADXL362.

La mesure des plages de ± 2 g, ± 4 g et ± 8 g sont disponibles, avec une résolution de 1 mg / LSB dans la plage de ± 2 g. Pour les applications où un niveau de bruit inférieure à la normale de 550 µg / √Hz de l'ADXL362, l'un des deux modes de bruit les plus faibles peut être sélectionné avec une augmentation minimale du courant d'alimentation. En plus de sa très faible consommation électrique, l'ADXL362 possède de nombreuses fonctionnalités pour permettre une véritable réduction de la consommation au niveau du système. [21]

Pin No.	Mnemonic	Description
1	$V_{DD I/O}$	Supply Voltage for Digital I/O.
2	NC	No Connect. Not internally connected.
3	Reserved	Reserved. Can be left unconnected or connected to GND.
4	SCLK	SPI Communications Clock.
5	Reserved	Reserved. Can be left unconnected or connected to GND.
6	MOSI	Master Output, Slave Input. SPI serial data input.
7	MISO	Master Input, Slave Output. SPI serial data output.
8	$\overline{CS}$	SPI Chip Select, Active Low. Must be low during SPI communications.
9	INT2	Interrupt 2 Output. INT2 also serves as an input for synchronized sampling.
10	Reserved	Reserved. Can be left unconnected, or connected to GND.
11	INT1	Interrupt 1 Output. INT1 also serves as an input for external clocking.
12	GND	Ground. This pin must be grounded.
13	GND	Ground. This pin must be grounded.
14	V_S	Supply Voltage.
15	NC	No Connect. Not internally connected.
16	GND	Ground. This pin must be grounded.

Figure III.7.Configuration des broches (vue de dessus) de l'accéléromètre ADXL362 [21]

III.4.4 Principe de fonctionnement de l'ADXL362

La composante mobile du capteur est une structure à base de poly-silicium qui est construite sur une plaque de silicium. Les ressorts à base de poly-silicium suspendent la structure sur la surface de la plaque et fournissent une résistance aux forces d'accélération. La déviation de la structure est mesurée à l'aide d'un condensateur qui est composé en une plaque fixe indépendante et une plaque mobile, fixée à la masse en mouvement. L'accélération dévie la structure et déséquilibre le condensateur différentiel, ce qui entraine une sortie du capteur dont l'amplitude est proportionnelle à l'accélération.

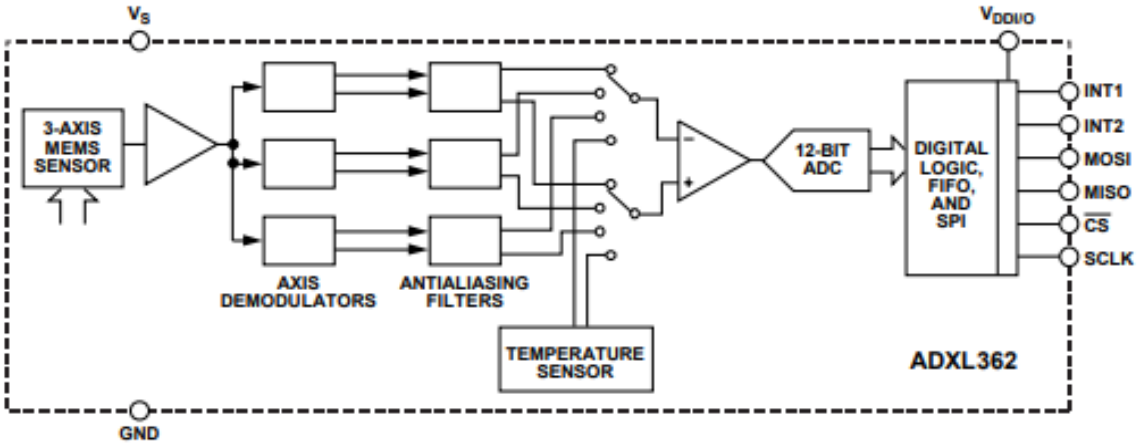


Figure III.8.Schéma fonctionnel de l'ADXL362 [21]

III.4.5 La communication Série SPI

L'ADXL362 communique via un SPI à 4 fils et fonctionne comme un esclave. Ignorer les données transmises de l'ADXL362 au périphérique principal pendant les notes à l'ADXL362. Comme le montrent les figures 36 à 40, la broche MISO est dans une position haute. État d'impédance, détenu par un détenteur de bus, sauf lorsque l'ADXL362 envoie des données en lecture (pour conserver la puissance du bus). Câblez l'ADXL362 pour la communication SPI comme indiqué dans le schéma de connexion de la figure 35. L'horloge SPI recommandée les vitesses sont comprises entre 1 MHz et 8 MHz, avec une charge maximale de 12 pF. Le schéma de synchronisation SPI suit CPHA = CPOL = 0.les connexions entre l'ADXL362 et le microcontrôleur NEO430 sont montré dans la figure III.9

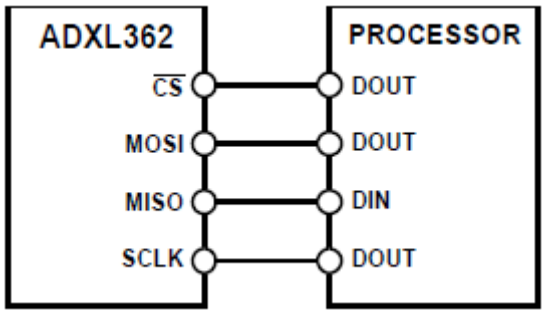


Figure III.9. Diagramme de connexion d'une liaison série SPI 4 fils

III.5 La plateforme NEXY4-DDR

La carte FPGA Nexys 4 est une plateforme de développement de circuit numérique prêt-à-utiliser, complète basée sur le tout dernier réseau de portes programmables Artix-7T. Avec

son FPGA large et de grande capacité, des mémoires externes généreuses, et un grand nombre de ports USB, Ethernet, et autres, la Nexys4 peut accueillir des modèles allant des circuits combinatoires préliminaires à de puissants processeurs embarqués. Plusieurs périphériques intégrés dont un accéléromètre, un capteur de température, un microphone numérique MEM, un amplificateur de haut-parleur et de nombreux périphériques d'E/S permettent au Nexys4 d'être utilisé pour une large gamme de modèles, sans avoir besoin d'autres composants. Le FPGA Artix-7 est optimisé pour la logique de haute performance, et offre plus de capacité, plus de performances et plus de ressources que les modèles précédents

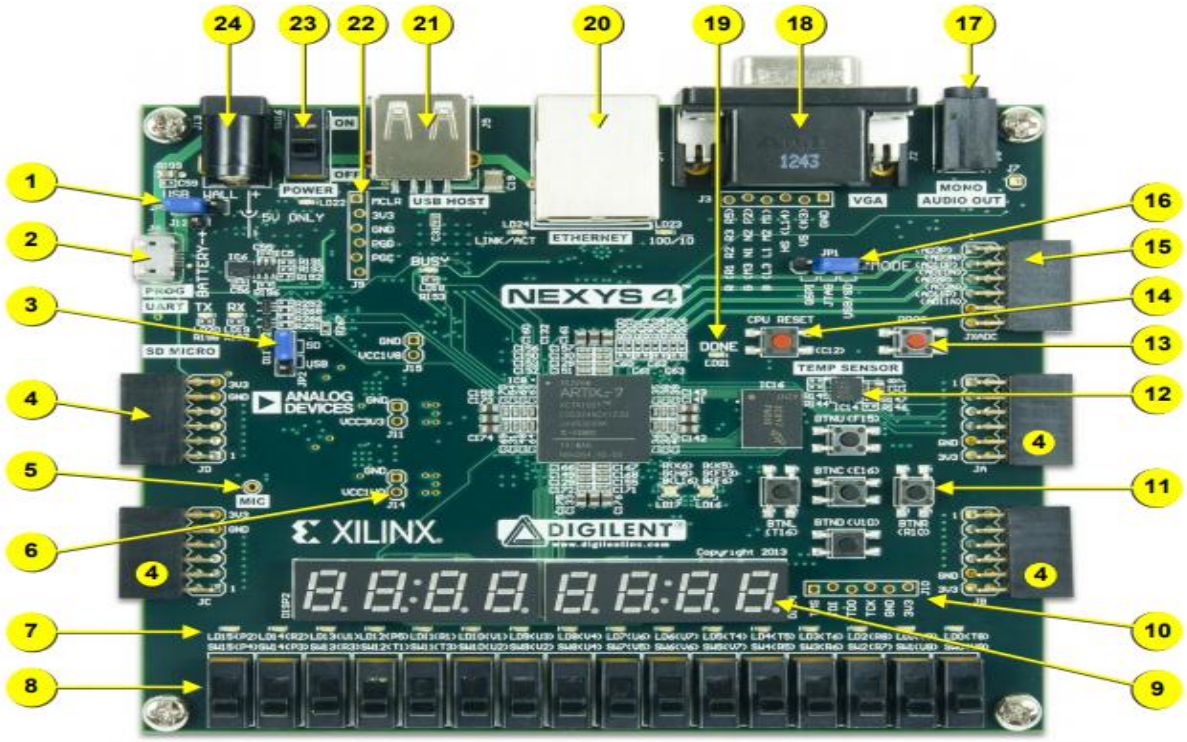


Figure III.10. Plateforme de développement Nexy4

Tableaux III.1.Les déférent pines de Nexy4 [21]

Callout	Component Description	Callout	Component Description
1	Power select jumper and battery header	13	FPGA configuration reset button
2	Shared UART/ JTAG USB port	14	CPU reset button (for soft cores)
3	External configuration jumper (SD / USB)	15	Analog signal Pmod connector (XADC)
4	Pmod connector(s)	16	Programming mode jumper
5	Microphone	17	Audio connector
6	Power supply test point(s)	18	VGA connector
7	LEDs (16)	19	FPGA programming done LED
8	Slide switches	20	Ethernet connector
9	Eight digit 7-seg display	21	USB host connector
10	JTAG port for (optional) external cable	22	PIC24 programming port (factory use)
11	Five pushbuttons	23	Power switch
12	Temperature sensor	24	Power jack

III.5.1 Les entrés et les sortie

La carte Nexys4 comprend deux DEL tricolores, seize commutateurs à glissière, six boutons poussoirs, seize DEL individuelles et ’afficheur à sept segments à huit chiffres, Les bouton-poussoir et les commutateurs à glissière sont connectés au FPGA via des résistances en série pour éviter les dommages causés par des courts-circuits par inadvertance [21].

III.6 Implémentation du système sur puce

Le logiciel Xilinx Vivado 2018 est un outil de conception de circuit pour la famille 7 et la famille Ultrascale des FPGA de Xilinx .ce logiciel permet essentiellement d’effectuer la différente étape propres à la synthèse de circuit numérique suer FPGA.

Utilisant l’outil Vivado, l’implémentation de notre système sur puce se fait en plusieurs étapes :

III.6.1 Création de projet

Un projet permet de regrouper plusieurs fichier sources, dans le projet crée et comme première étape on ajoute les différents fichiers source de type VHDL des différent modules (IPs) du système basé sur le processeur NEO430 décrit dans la section 4.1 en s’assurant que le

fichier NEO430_top. VHD est en tête d'hiérarchie des fichiers de l'instance du système de processeur et on ajoute aussi le fichier VHDL DE L4INTERFACE série TempSensor CTL comme source décrivant notre système sur puce. Après la création des deux instances, il est temps maintenant de les relier à travers un fichier VHDL global (NEO430TEMP) qui sera notre TOPLEVEL ENTITY utilisant la technique du PORT MAP. La figure III.11 montre la hiérarchie des différents fichiers sources du code VHDL décrivant l'architecture du SOC.

III.6.2 Affectation des pins

Le FPGA sur lequel nous allons implanter notre système sur puce comporte un grand nombre de pin d'entrés /sortie, cette étape consiste a créé un fichier du type XDC(Xilinx Design Contraint) qui sert a relie les différentes entrés /sortie de notre système montrées dans la figure a des pins spécifiques du FPGA qui sont eux même reliés aux autres composants de la carte de développement Nexys4. L'annexe « 1 » donne une partie du fichier de contraintes XDC.

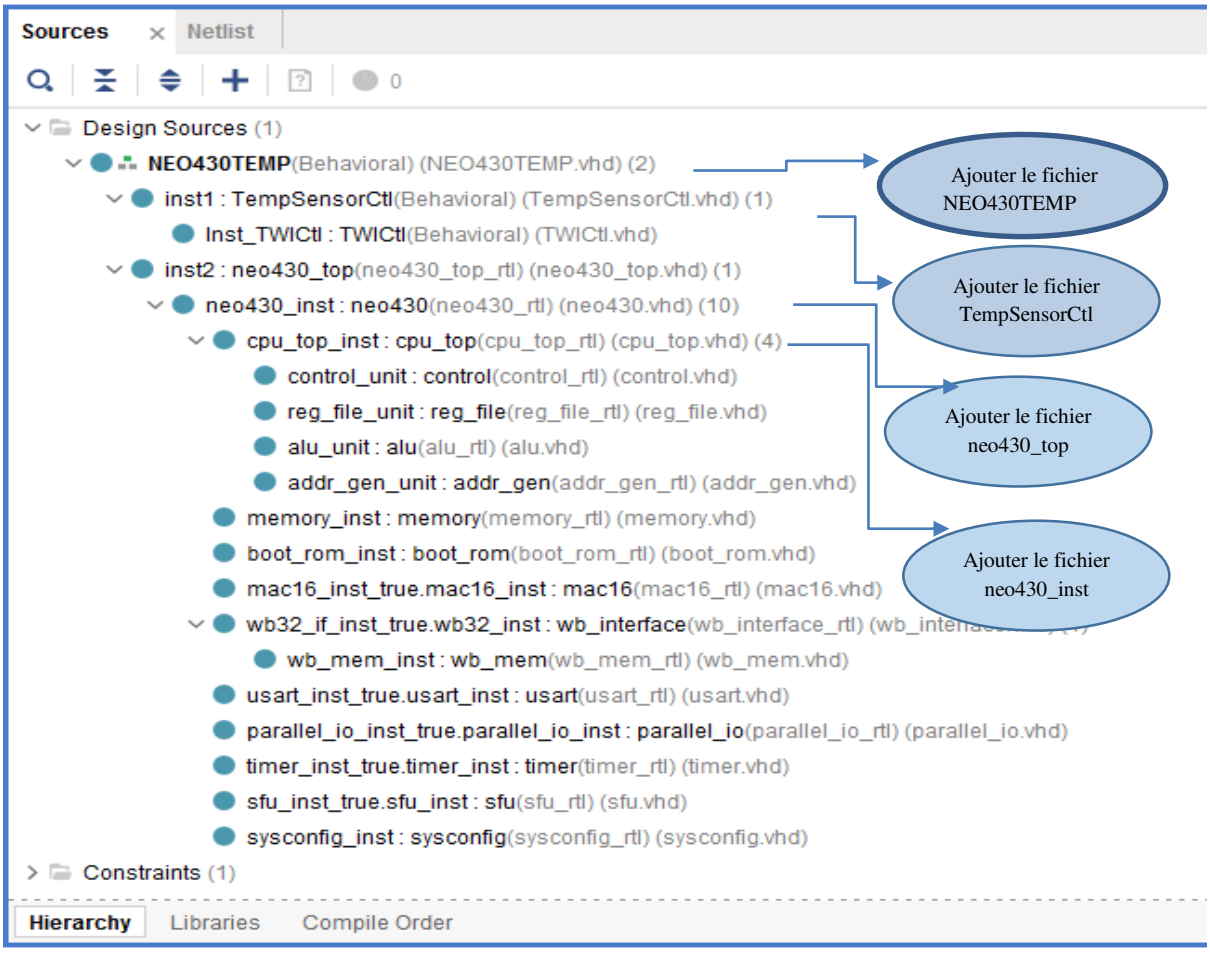


Figure III.11. Hiérarchie des différents fichiers source VHDL [22]

III.6.3 Fréquence de l'horloge

Dans notre système la fréquence de l'horloge correspond à celle de la carte et est fixée à 100 MHZ

III.6.4. Résultats de Synthèse

La synthèse consiste à traduire la description de notre système décrit en VHDL en blocs et composantes disponibles sue le FPGA choisi comme les LUTs, les bascules, les blocs RAMs, etc. La figure représente une partie du circuit de l'unité arithmétique et logique obtenue après l'opération de la synthèse.

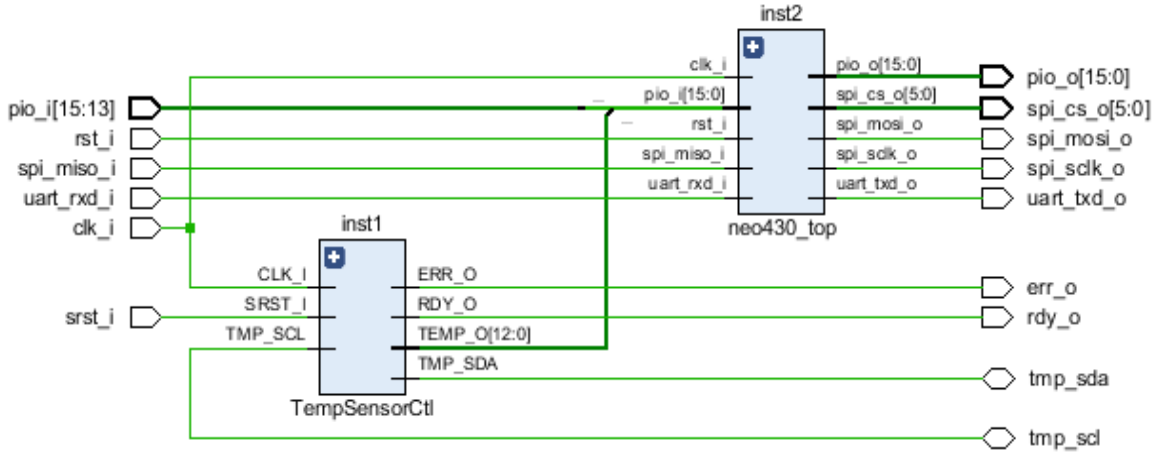


Figure III.12. Schéma RTL du SOC obtenue après la synthèse du système [22]

Tableau III.2.ulistration des ressources du FPGA après la synthèse du système [22]

Resource	utilisation	avaible	Utilisation %
LUT	1081	63400	1.71
LUTRAM	33	19000	0.17
FF	780	126800	0.62
BRAM	3.50	135	2.59
DSP	1	240	0.42
IO	37	210	17.62
BUFG	1	32	3.13

Le tableau résume l'utilisation des ressources dans l' FPGA après synthèse du système. Ces résultats obtenus montrent une utilisation de 1, 17% de LUT, 0.17% de RAM, 0.62% de FF, 2.59% de mémoire BRAM, 17 % d'entrée sortie, 1 circuit DSP, et 3.13% de BUFG. On constate que notre système SOC ne consomme pas beaucoup de ressources du circuit FPGA Artix 7 ce qui laisse l'espace pour élargir notre système et rajouter d'autres fonctionnalités en fonction des applications du nœud de capteurs.

III.6.5 Résultats d'implémentation

Cette étape est divisée en trois sous étapes :

- **Transformation (mapping) :** consiste à regrouper les composantes obtenues lors de la synthèse dans des blocs spécifiques du FPGA.

- **Disposition (placement) :** choisir des endroits spécifiques sur le FPGA ou disposer les blocs utilisés, et choisir les pins du FPGA correspondant aux ports d'entrée et de sortie.
- **Routage (routing) :** consiste à établir des connexions électriques entre les blocs utilisés.

La figure III.13 montre les réultats d'implémentation dans le circuit FPGA

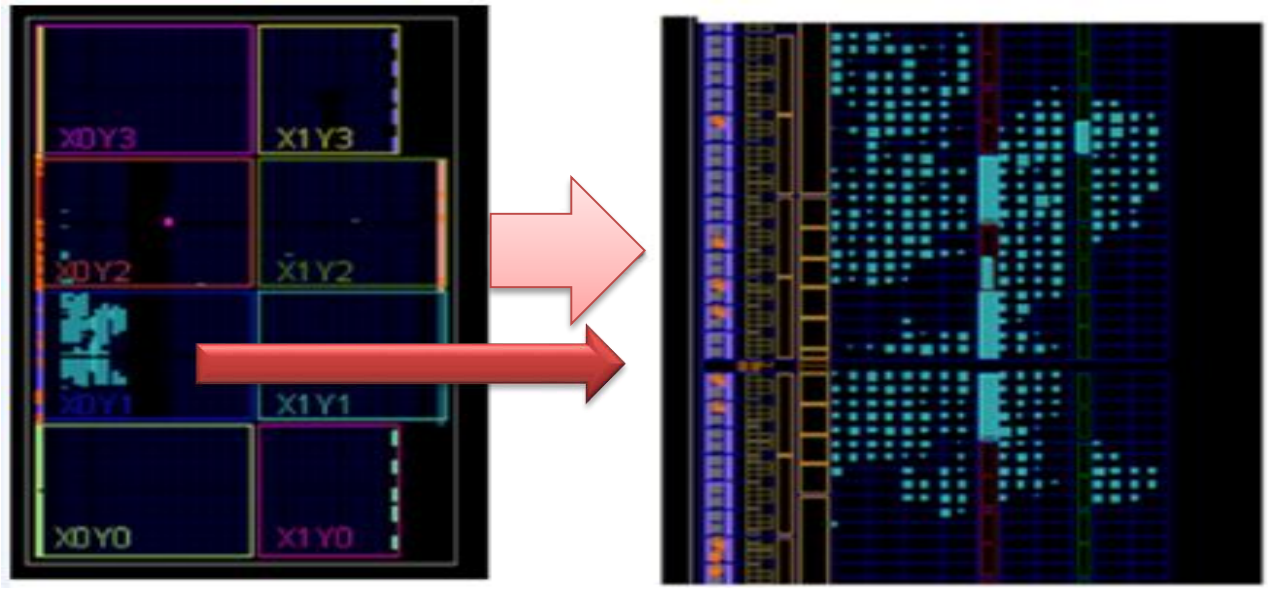


Figure III.13.Résultats d'implémentation [22]

III.6.6 Estimation de la puissance dans L'outille Vivado : XPOWER Analyzer

Cette section couvre l'analyse de la puissance à l'aide du module "XPower ANALYSER " intégré dans l'IDE Vivado. La Figure III.14 montre l'interface graphique du module d'analyse de la puissance [référence document].

L'outil XPOWER permet au concepteur d'analyser et de prévoir la puissance dissipée du circuit FPGA en fonction de certains paramètres relatifs à l'environnement dans lequel sera placé le circuit FPGA et qui sont liés aux variations de la température (Tem Grad :commercial, industriel, militaire, etc), du process de la technologie de fabrication (Process: typique, maximun), de la charge externe que doit supporter le circuit FPGA (output load:), du débit de l'air qui circule autour du circuit FPGA (Airflow:), de la présence ou nom d'un radiateur (heat sink:), ainsi que des dimensions du circuit imprimé lui-même.

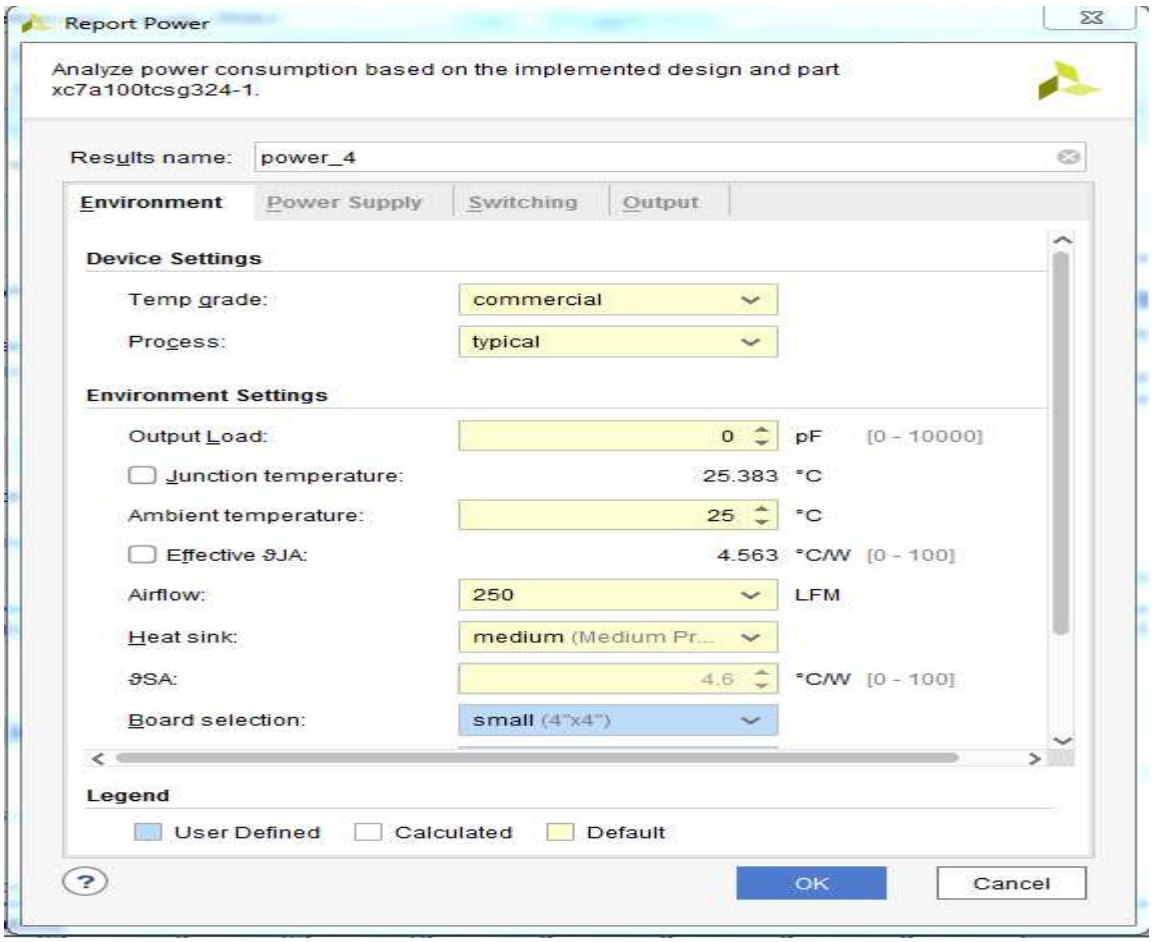


Figure III.14.Report Power Dialog Box [22]

Pour notre cas, et selon les spécifications du projet, nous avons considéré les paramètres liés à un produit commercial, un process normal (typique), et nous avons choisi d'analyser la puissance du circuit FPGA en fonction des dimensions de la carte PCB. Pour cela nous avons choisi les dimensions de la carte (4"x4" inch), c.a.d (10cmx10cm).

III.6.6.1 Résultats d'analyse de la puissance

La figure ci-dessous III.15 donne une estimation de la puissance dissipée

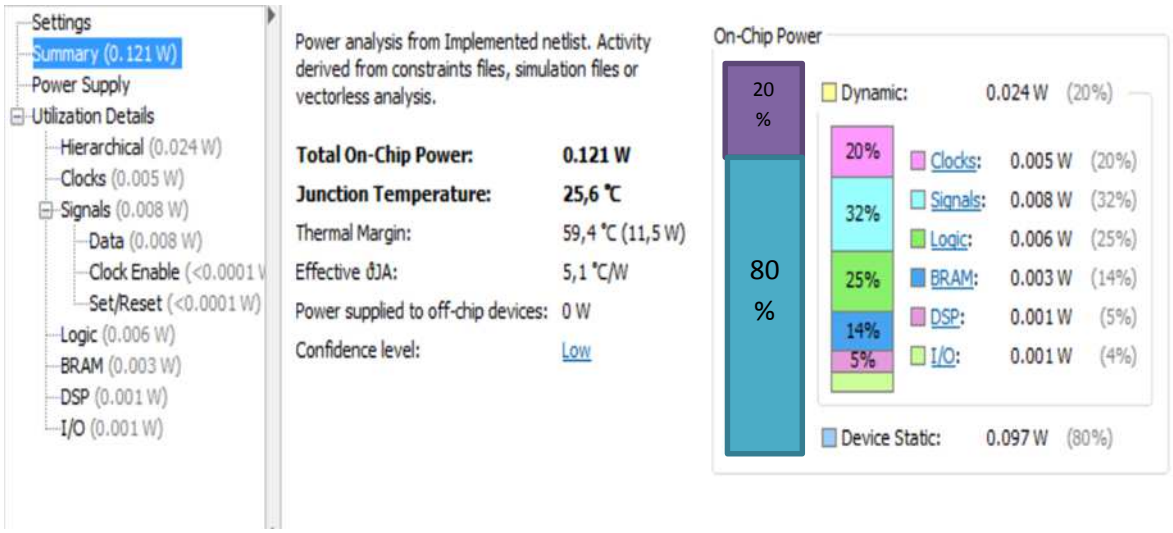


Figure III.15. Estimation de la puissance totale dissipée par le nœud de capteur sur FPGA [22]

Les résultats obtenues, montrent que pour les un circuit imprimé de (4"x4"), le circuit consomme une puissance totale de 121 mW, dont 24 mW de puissance dynamique et 97mW de puissance statique. A ce stade, aucun problème liés à la possibilité de sur-chauffage de la carte après réalisation n'est mentionné par l'outil.

Par ailleurs, les résultats de l'analyse de la puissance permettent d'identifier les différentes tensions d'alimentations utilisées dans le circuit FPGA. Ces tensions d'alimentation sont (tableau III.3)

- VCCINT (1V) :

- VCCAUX (1.8 V)
- VCCO (3.3V)
- VCCBRAM (1 V)

VCCadc (1.8 V) Tension utilisé pour alimenter le convertisseur analogique numérique de la carte

Tableau III.3. Les différentes tensions d'alimentation du circuit FPGA [22]

Supply Source	Voltage (V)	Total (A)	Dynamic (A)	Static (A)
Vccint	1.000	13.484	13.247	0.237
Vccaux	1.800	0.065	0.018	0.047
Vcco33	3.300	0.141	0.137	0.004
Vcco25	2.500	0.000	0.000	0.000
Vcco18	1.800	0.000	0.000	0.000
Vcco15	1.500	0.000	0.000	0.000
Vcco135	1.350	0.000	0.000	0.000
Vcco12	1.200	0.000	0.000	0.000
Vccaux_io	1.800	0.000	0.000	0.000
Vccbram	1.000	0.033	0.026	0.007
MGTAVcc	1.000	0.000	0.000	0.000
MGTAVtt	1.200	0.000	0.000	0.000
Vccadc	1.800	0.020	0.000	0.020

III.7 Conclusion

Dans ce chapitre, la description matérielle et le flot de conception du système implémenté ont été présentés, le système est composé de deux sous-systèmes : un système sur puce à base du processeur NEO430 implémenté sur la carte FPGA Artix7 de Xilinx à l'aide de l'outil Vivado et un autre système situé hors puce FPGA qui consiste de l'unité de captage sous la forme d'un capteur de température et un accéléromètre, un afficheur 7 segments. Aussi, Nous avons présenté également les résultats de synthèse, d'implémentation et d'analyse de la puissance. Les

fichiers de contraintes de conception du SOC, ainsi que les résultats issus de l'analyse de la puissance représentent une étape importante pour entamer la conception du circuit imprimé.

La conception du circuit imprimé fera l'objet du prochain chapitre

Chapitre IV Conception du circuit imprimé du nœud de capteur basé sur FPGA

IV.1 Introduction

La conception d'un circuit imprimé représente une tâche nécessaire avant la phase de production, elle est généralement accomplie par l'entremise des logiciels spécifiques.

L'objectif de ce chapitre et de montrer la stratégie adoptée pour la conception du circuit imprimé du nœud de capteur basé sur FPGA moyennant l'outil Altium Designer, tout en exploitant les résultats de synthèse et d'implémentation de l'architecture pour obtenir les fichiers de sortie nécessaires à la fabrication

IV.2 Méthodologie de conception du PCB

La figure IV.1 ci-dessous montre la méthodologie adoptée pour la conception de notre circuit PCB.

Il est important de signaler que pour la conception d'un circuit imprimé à base d'un circuit FPGA, il faut suivre la méthodologie de conception des circuits PCB en général (présenté dans le chapitre 1, paragraphe), à laquelle il faut prendre en considération les résultats de synthèse et d'implémentation issues de l'outil VIVADO : le fichier de contraintes de conception, les résultats du schéma RTL, les résultats d'analyse de la puissance (dimensionnement de la carte et différentes tensions d'alimentation).

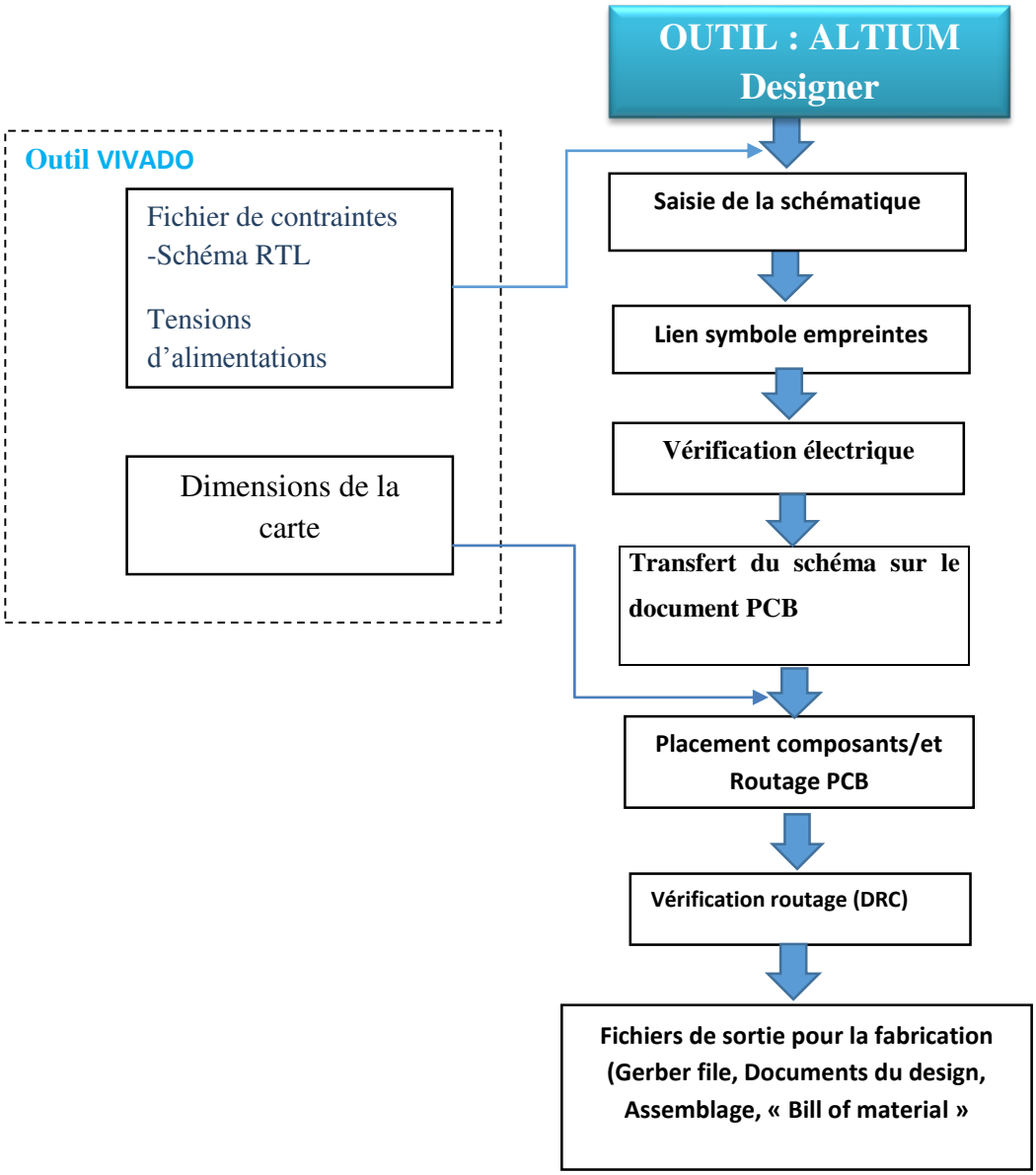


Figure IV.1 la méthodologie adoptée pour la conception de notre circuit PCB.

Dans la suite de cette section, on présentera l'outil Altium designer que nous avons utilisé pour la conception de notre PCB

IV.2.1 Présentation de l'outil Altium designer

Altium Designer est un puissant outil de CAO électronique développé par l'entreprise Altium. Ce logiciel permet de saisir des schémas électriques, les vérifier, les simuler et aller jusqu'à la conception du circuit imprimé.

La grande force d'Altium Designer c'est les nombreux outils qu'il intègre par défaut. Ce logiciel représente une seule solution informatique pour concevoir et développer un montage électronique ainsi que la simulation, le routage, l'édition et la fabrication de PCB, en allant du schéma jusqu'à la programmation des composants de ce schéma. Voici en outres les principales caractéristiques de ce logiciel :

- L'outil unique permet de simplifier le travail entre l'éditeur de schématique, la simulation, la réalisation du circuit imprimé et toutes les autres fonctionnalités
- Réaliser un prototype via le logiciel grâce aux différents outils ; incluant un outil de visualisation 3D
- Adapté pour l'avenir (prend en compte certaines nouvelles technologies et mise à jour régulière)
- Nombreuses bibliothèques de composants actuels. [25]

Les différentes fenêtres et menus de l'outil sont représentés sur la figure IV.2

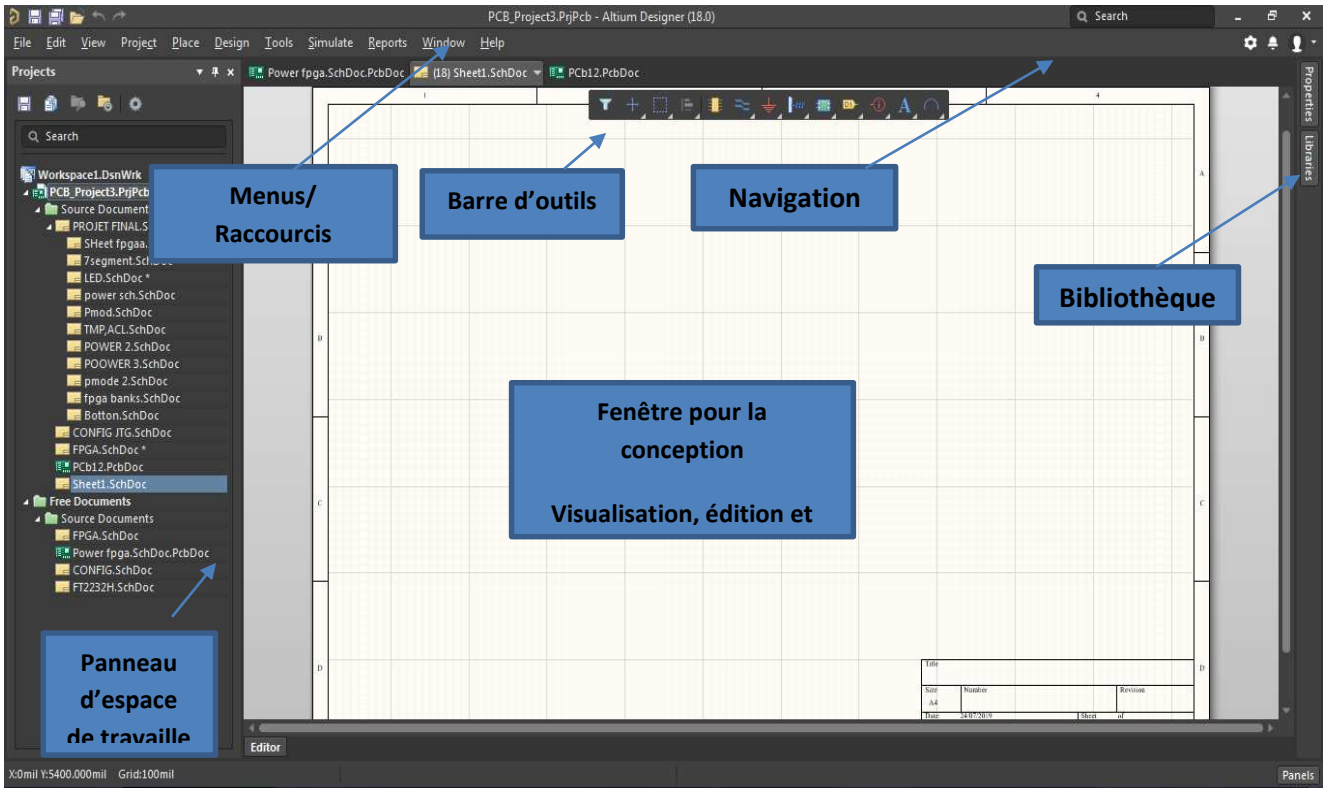


Figure IV.2 : Environnement d'Altium Designer

IV.3 Conception du circuit imprimé du nœud de capteur basé sur FPGA

IV.3.1 Présentation de l'architecture générale du PCB (schéma Électrique) :

La figure (IV.3) montre l'architecture générale du circuit imprimé du nœud de capteur la carte FPGA. Ce dernier est composé des blocs suivants :

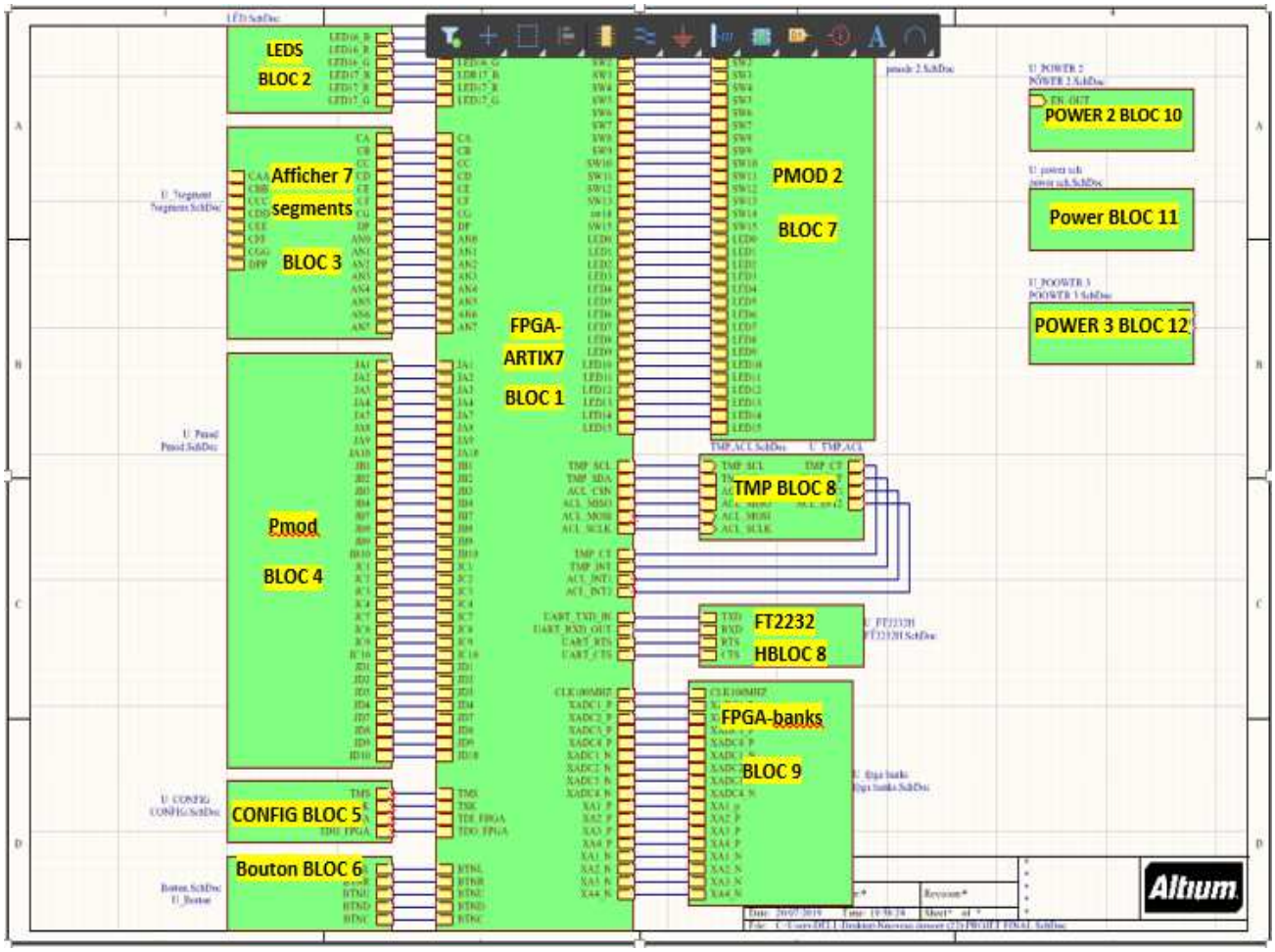


Figure IV.3 : Schéma bloc de passerelle de nœud de capteur basé sur FPGA

- **Le circuit FPGA BANKS Xilinx Artix-7 100T(BLOC1) :** il contient le Bank 14, le Bank 15, le Bank 34 et le Bank 35 et le Bank U-CONFIG (Voir section1.2.7.1), voici la figure IV.4 :

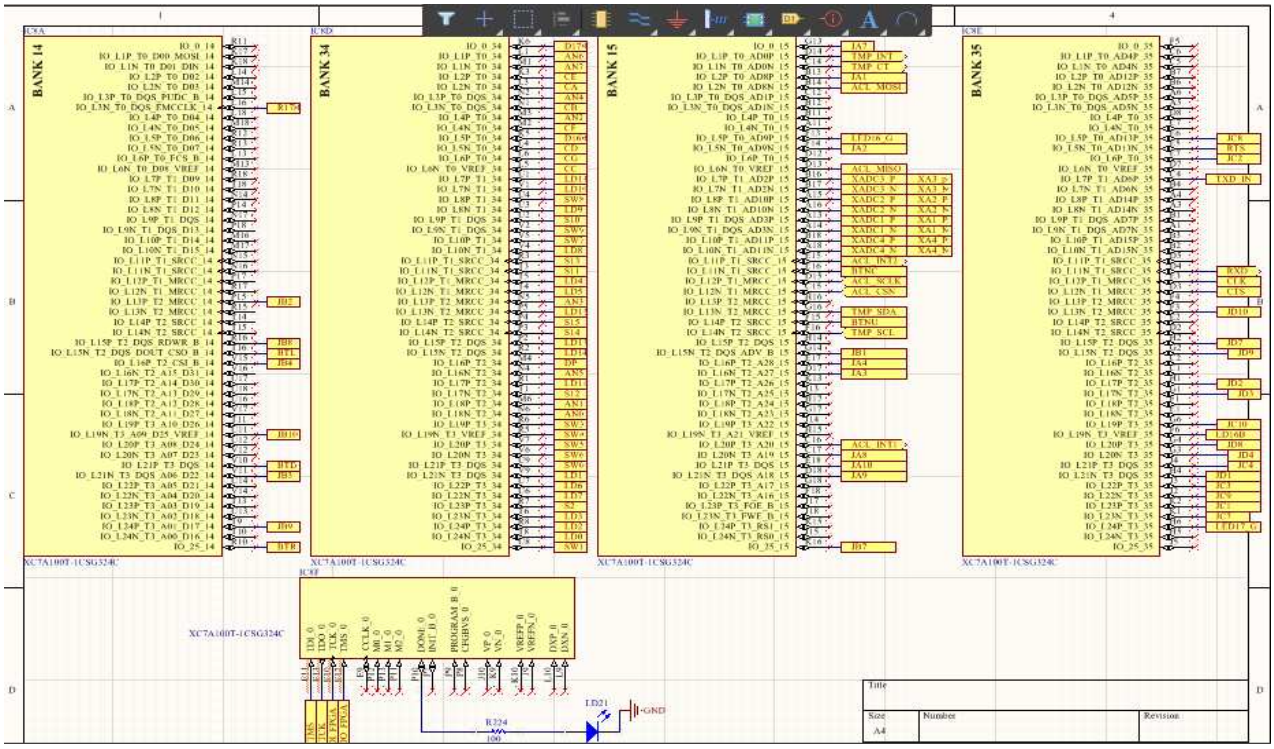


Figure IV.4. La schématique interne du circuit FPGA et ses BANKS

- **L'Afficheur 7 segment (BLOC 2) :** La carte Nexys4 DDR contient deux écrans LED à sept segments à anode commune à quatre chiffres, configurés pour se comporter comme un seul écran à huit chiffres. Chacun des huit chiffres est composé de sept segments disposés en forme de «4.3», avec une LED intégrée dans chaque segment. Les voyants de segment peuvent être allumés individuellement, de sorte que l'un des 128 modèles puisse être affiché sur un chiffre en éclairant certains segments de LED et en laissant les autres sombres, Parmi ces 128 modèles possibles, les dix correspondants aux chiffres décimaux sont le plus utile. (voir la schématique de la figure IV.5).

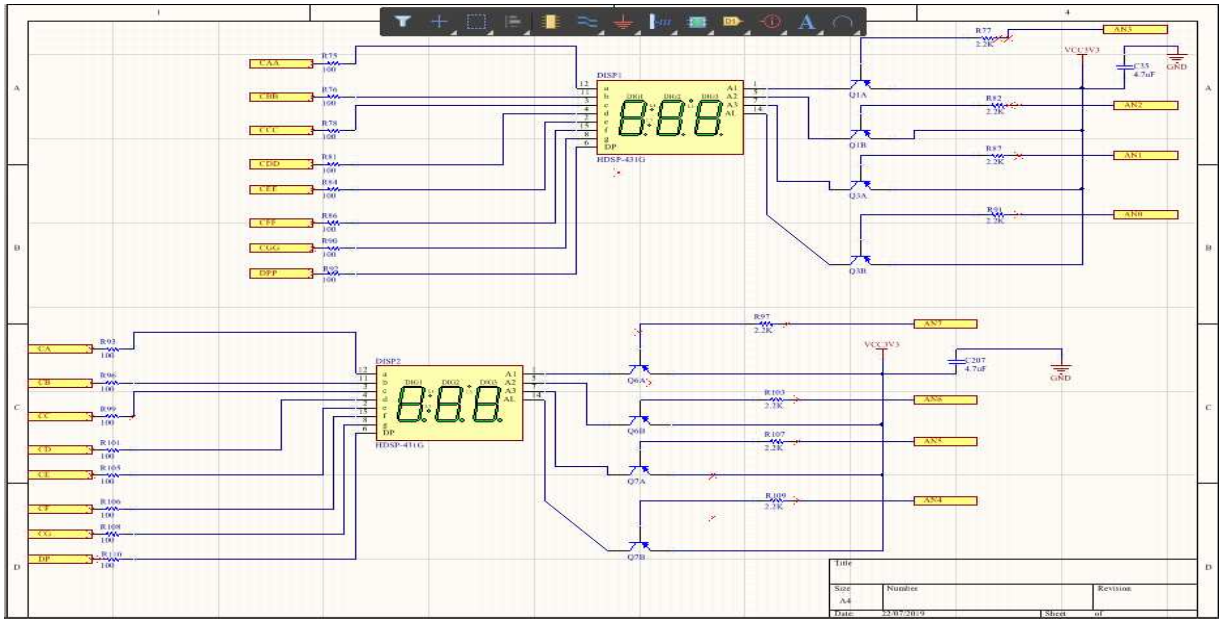


Figure IV.5. Schématique de l'afficheur 7 segments

- **Le capteur de température et l'accéléromètre (BLOC 8) :** La carte FPGA comprend un capteur de température Analog Device ADT7420. Le capteur offre une résolution allant Jusqu'à 16 bits avec une précision typique meilleure que 0,25 degrés Celsius. L'interface entre la température capteur et FPGA est montré.

La carte FPGA comprend un accéléromètre analogique pour appareil ADXL362. L'ADXL362 est un accéléromètre MEMS à 3 axes qui consomme moins de 2 µA à un débit de sortie de 100Hz et 270 nA en mode réveil déclenché par un mouvement .Contrairement aux accéléromètres qui utilisent le cycle de puissance pour réduire la consommation d'énergie, l'ADXL362 ne crée pas de repli sur les signaux d'entrée par sous-échantillonnage; il échantillonne toute la bande passante du capteur à tous les débits de données. L'ADXL362 fournit toujours une résolution de sortie de 12 bits. Des données au format 8 bits sont également fournies pour des transferts plus efficaces sur un octet lorsqu'une résolution inférieure est suffisante. Les plages de mesure de ± 2 g, ± 4 g et ± 8 g sont disponibles avec une résolution de 1 mg / LSB sur la plage de ± 2 g. Le FPGA peut communiquer avec l'ADXL362 via l'interface SPI. Lorsque l'ADXL362 est en mode de mesure, il mesure et stocke en continu les données d'accélération dans les registres X-data, Y-data et Z-data. L'interface entre le FPGA et l'accéléromètre peut être vu Foire la figure IV.6

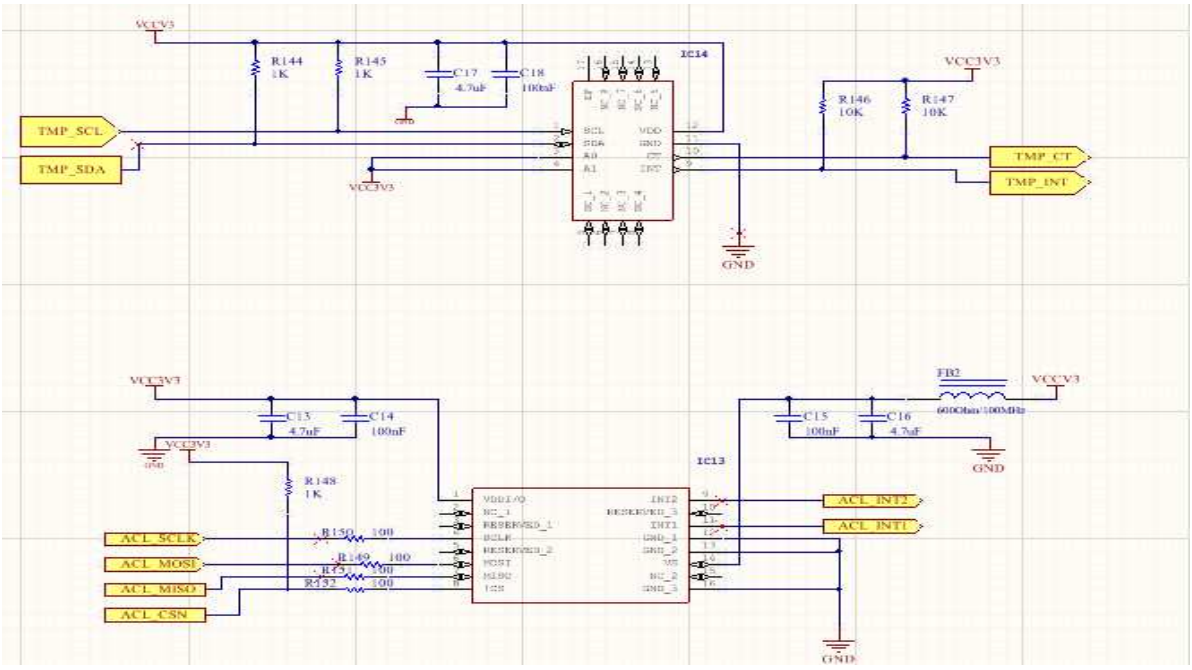


Figure IV.6.Schématique de capteur de température et accéléromètre

L'Alimentation (BLOC 10, 11, 12)

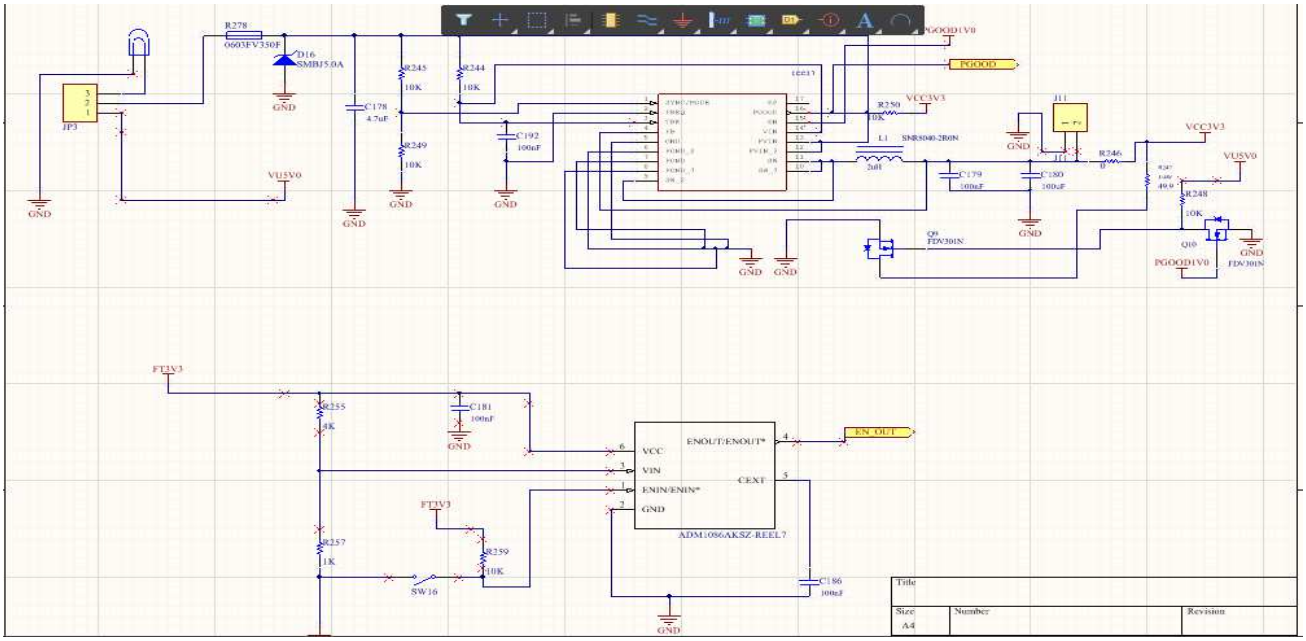


Figure IV.7.schématique de L'alimentation

IV.4 Création de la carte

Nous avons créé le document PCB intitulé où nous allons importer les empreintes des composants. Pour la création de la carte et la définition de ses paramètres, nous avons utilisé l'outil PCB Board Wizard. Les étapes suivies sont :

- Lancement de l'outil PCB Board Wizard (figure IV.8)

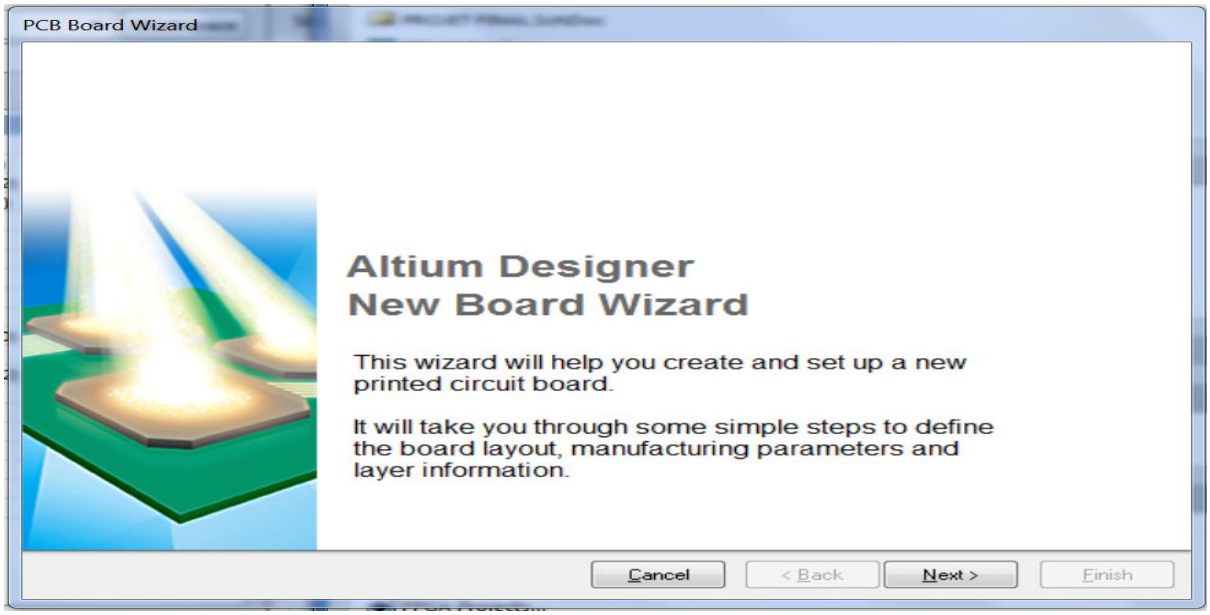


Figure IV.8.Fenêtre PCB Board wizard

- de l'unité de mesure (métrique) (figure IV.9)

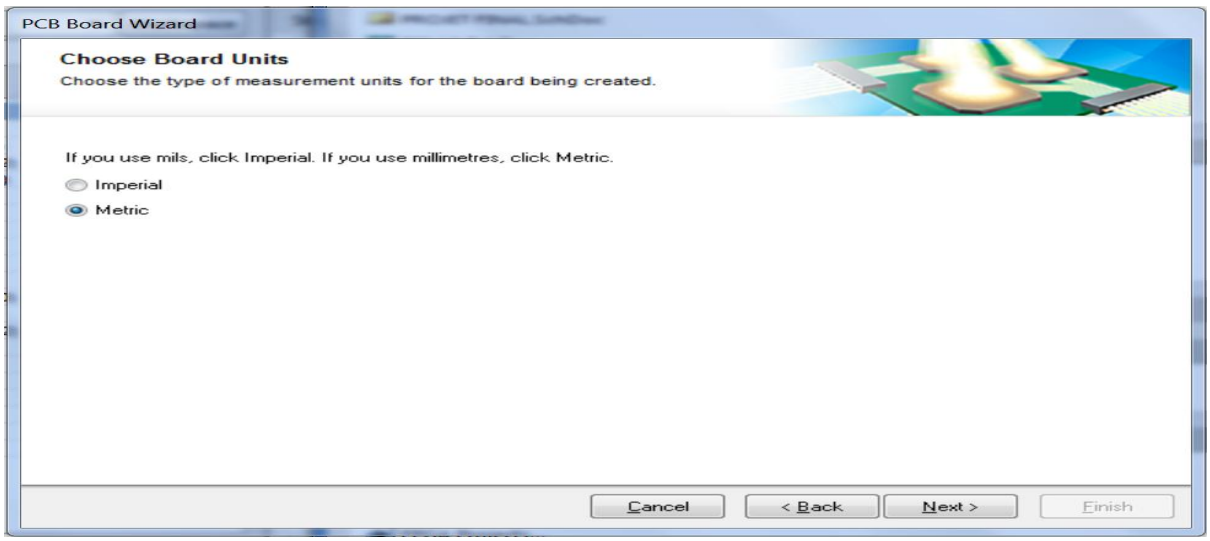


Figure IV.9. Choix de l'unité de mesure

- Choix du format de la carte et ses dimensions

Le format de la carte est rectangulaire.

Les dimensions de la carte sont : (la largeur : 100 mm, la longueur : 100 mm).

La distance minimale standard entre les déférents objets (piste, via..) et les bords Extérieurs de la carte est de 0.254mm. Dans notre cas, nous l'avons fixée à 1.3mm pour éviter que le cuivre soit endommagé lors de la définition du contour de la carte pendant la fabrication (figure IV.10)

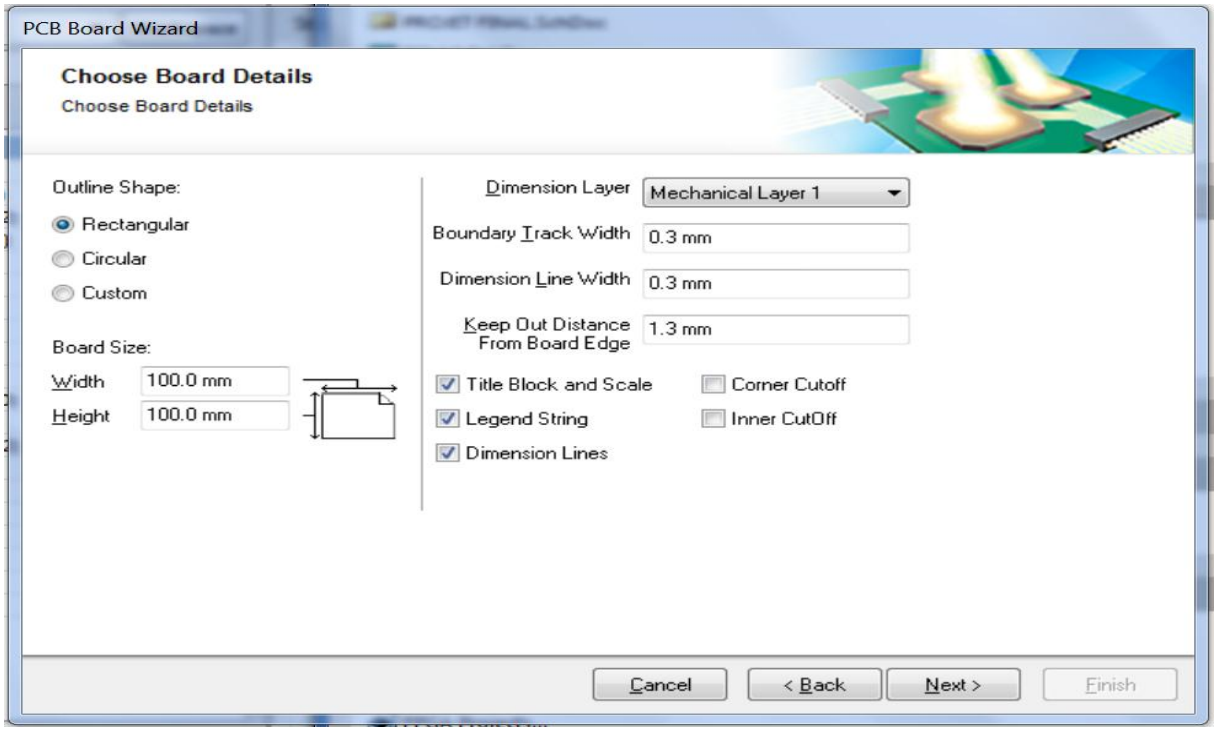


Figure IV.10. Choix de format de la carte

- Choix du nombre de couches pour les signaux et pour les plans d'alimentation

Le nombre de couches total : 8 couches (6 couches pour les signaux et 2 plans d'alimentation) (foire figure IV.11)

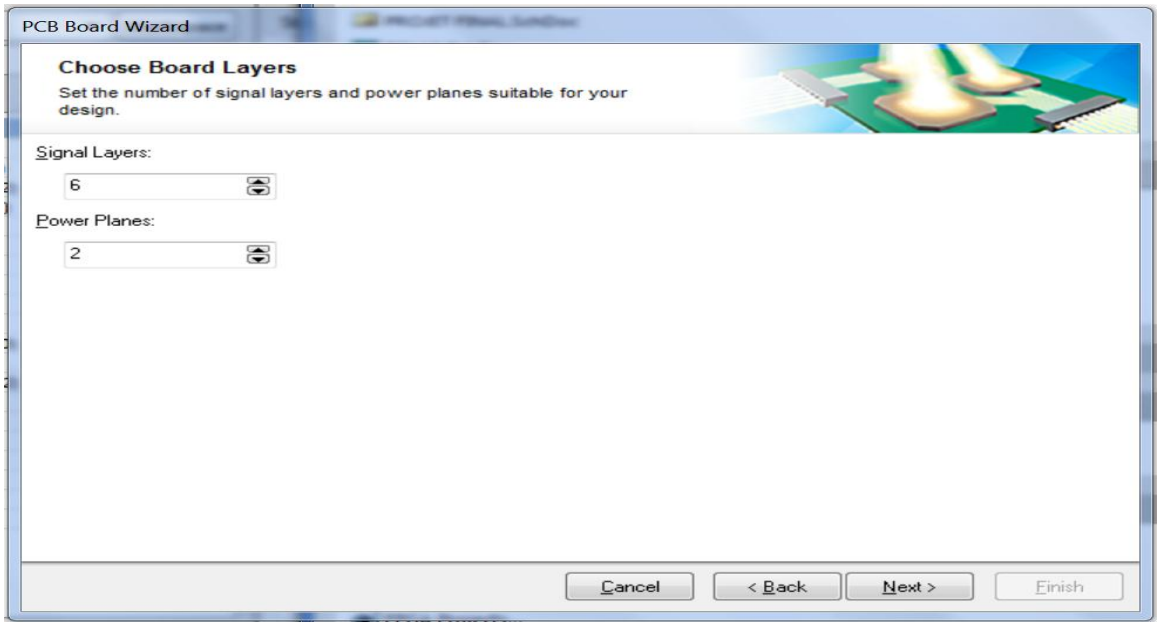


Figure IV.11.Choix du nombre de couches

- Choix du type des vias

Les vias à utiliser dans la carte sont de type traversant seulement (figure IV.12)

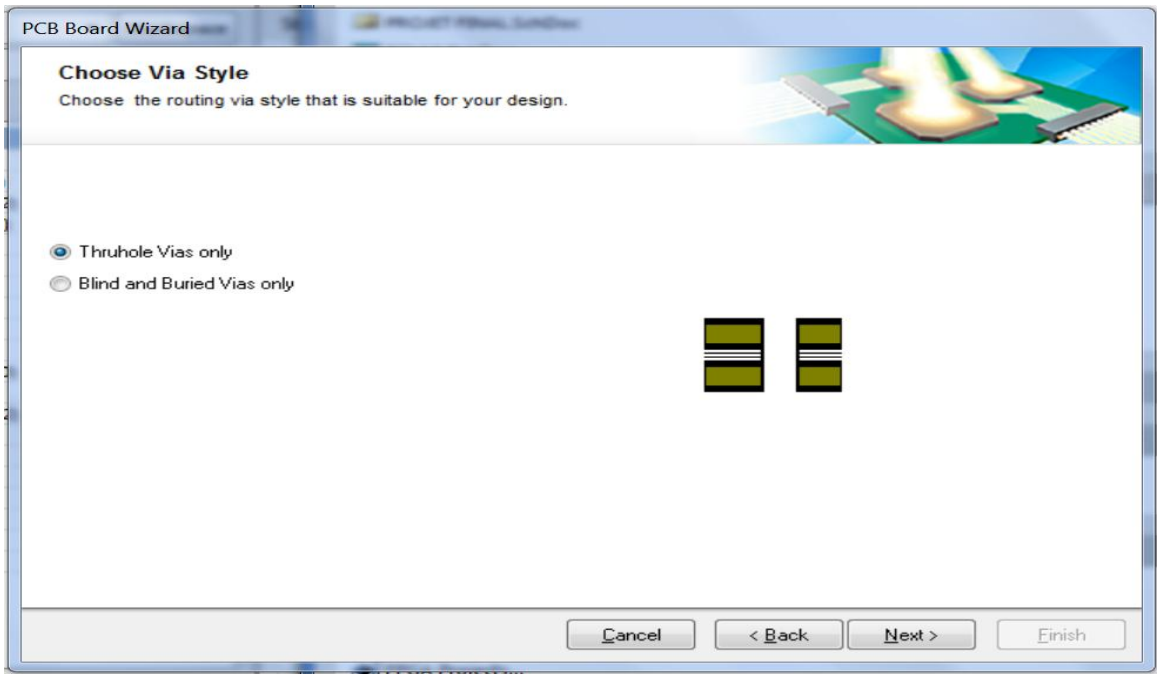


Figure IV.12. Choix de type des vias

- Choix des dimensions des vias, de la largeur des pistes et de l'espacement entre deux conducteurs (figure IV.13)

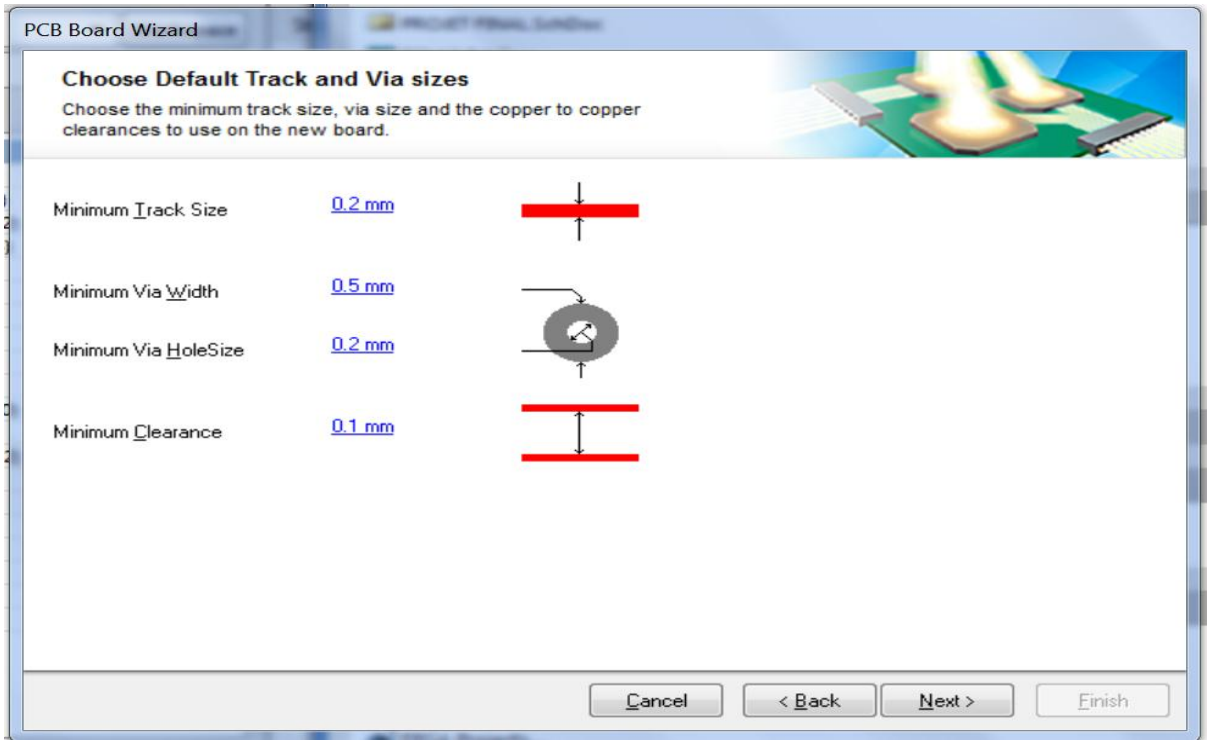


Figure IV.13. Choix des dimensions des vias, la largeur des pistes et l'espacement entre deux Conducteurs.

- Définition de la largeur des pistes

Pour définir la largeur des pistes, il faut avoir une idée sur l'intensité du courant qui circule dans la carte. Plus le courant parcourant les pistes est élevé plus les pistes doivent être larges.

Les courants circulant dans la carte sont faibles, et vu la complexité de cette dernière (un fort encombrement de composants), la largeur minimale de la piste définie vaut 0.2mm.

- Les dimensions des vias

Un ratio maximum entre le diamètre du trou et l'épaisseur du circuit imprimé (appelé aspect ratio) percé doit être respecté. Il est néanmoins admis qu'un ratio de 5 :1 peut être utilisé par la plupart des fabricants.

L'épaisseur totale du circuit imprimé est obtenue dans la fenêtre « Layer Stack Manager », elle est égale à 2.29992 mm La division de cette valeur sur 5 donne 0.45, donc le diamètre des vias ne doit pas être inférieur à 0.46mm.

- Définition de l'espacement entre deux conducteurs

Il s'agit de l'espacement entre deux pistes, deux vias ou bien entre une piste et un via. D'après la norme IPC-2221A qui est la norme principale générique pour la conception d'un circuit imprimé, l'espacement minimal entre deux conducteurs pour les PCB avec une plage de voltage (0-15v) et de la catégorie « assemblage » vaut 0.13 mm

- Une fois ces paramètres sont réglés, la surface de la carte apparait dans le plan de travail (figure IV.14)

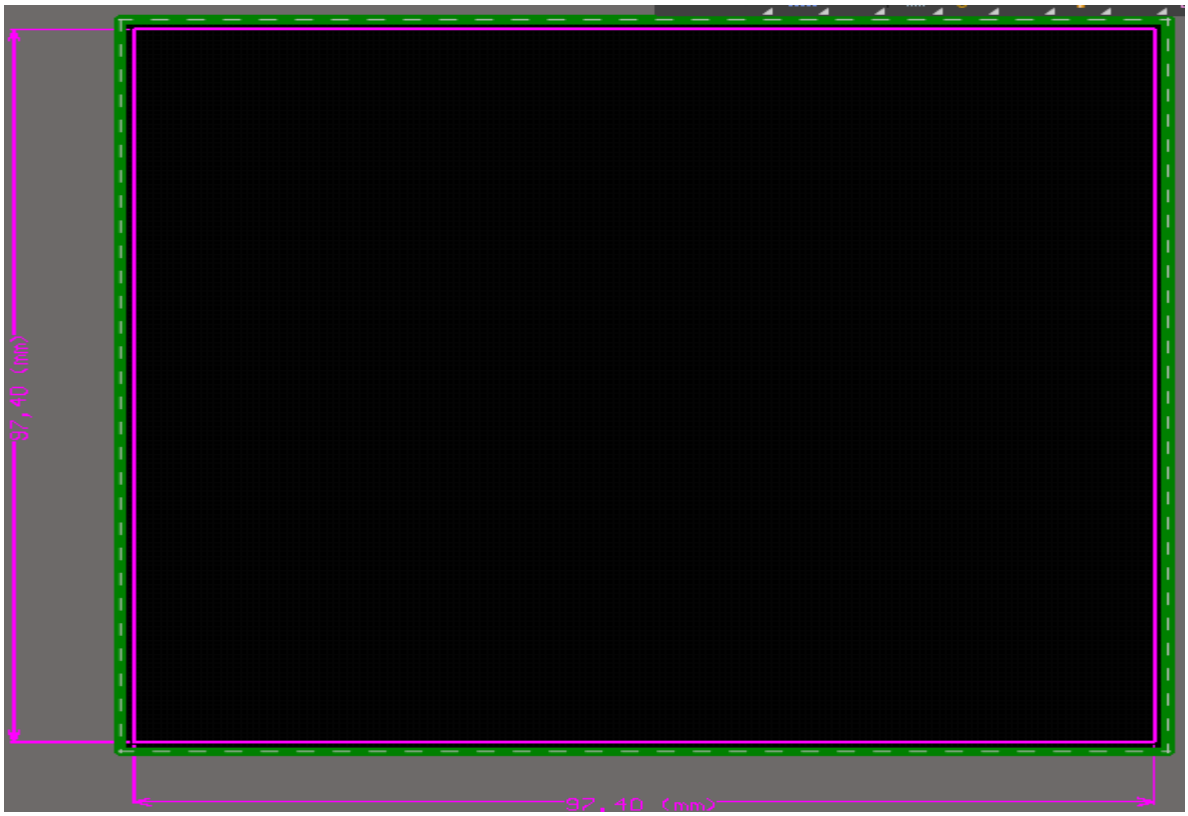


Figure IV.14. La surface de la carte

IV.4.1 Import des composent dans la surface du PCB

Dans cette étape, tous les composants et leurs interconnexions sont importés à partir des documents schématiques vers le document PCB (figure IV.15)

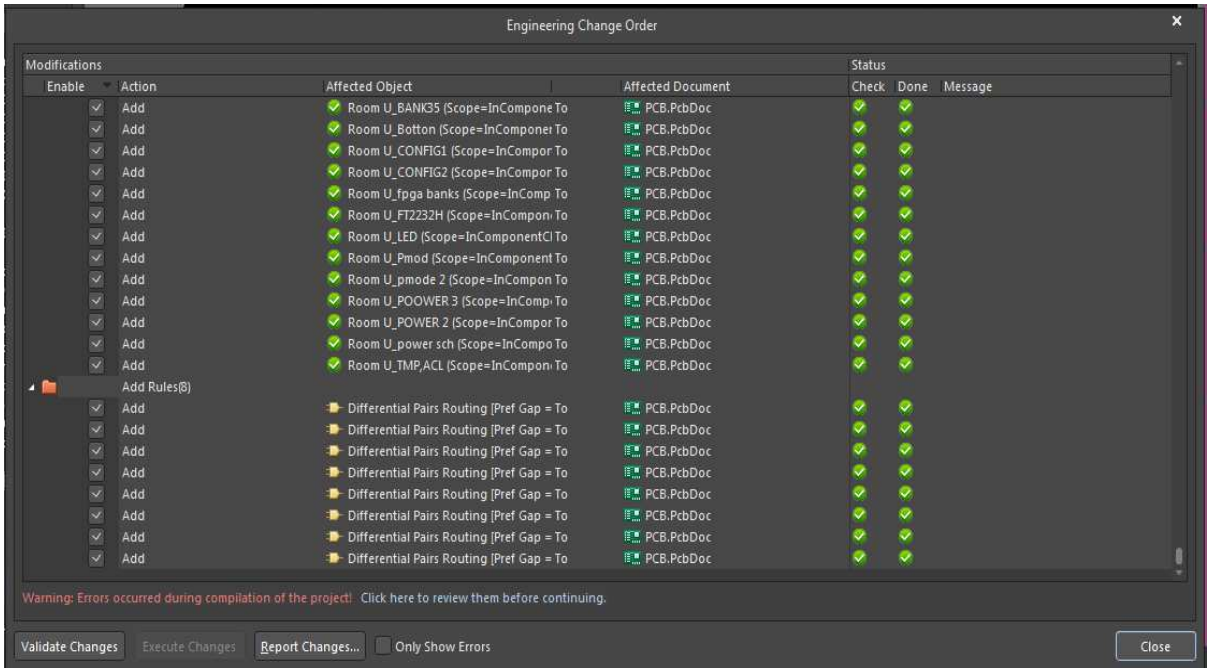


Figure IV.15. Import des composent vers le PCB

Après validation (l'outil vérifie la disponibilité des empreintes des composants), les composants apparaissent dans le plan de travail et ils sont prêts à être placés. Les connexions sont matérialisées par des fils très ns reliés directement entre les pins des composants (figure IV.16)

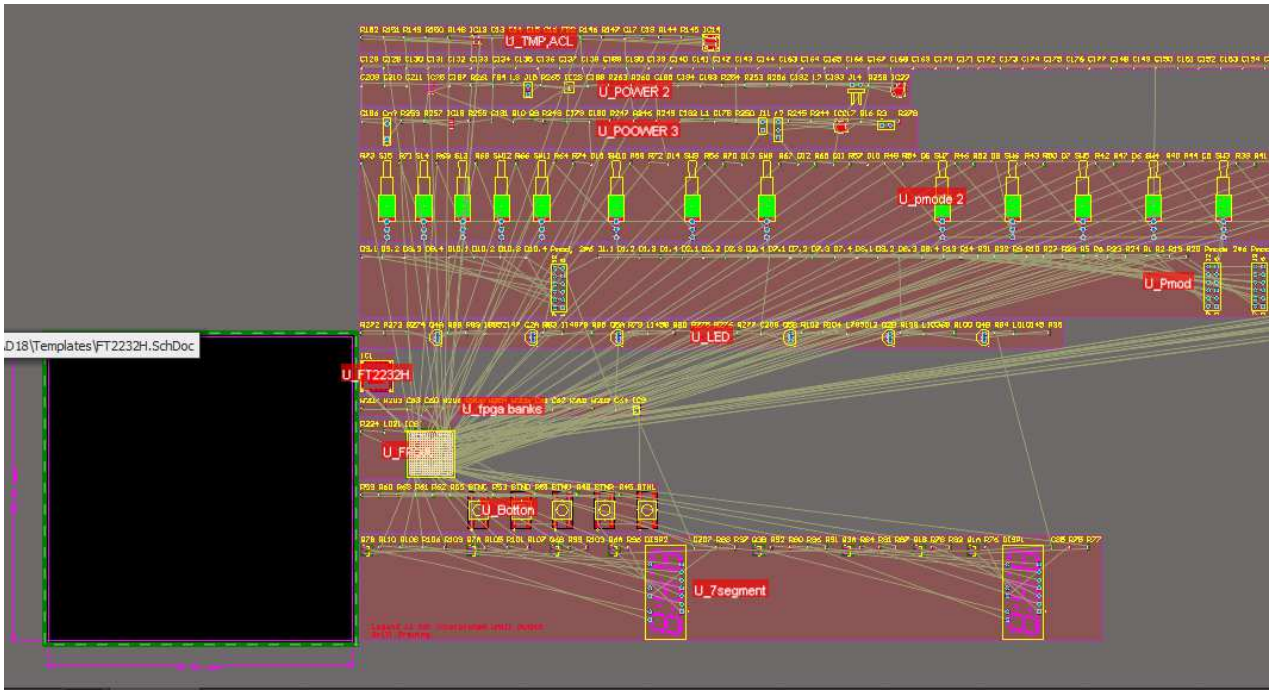


Figure IV.16.La carte avec les composants non placés

IV.4.2 Le placement

L'étape de placement des composants constitue un processus important qui est à la fois un art et une science, nécessitant une considération stratégique sur les catégories des composants (PMODE, FBGA,) et la surface disponible sur la carte. En fait, la conception de PCB est 90% de placement et 10% de routage. Placer les composants avec précision rend le routage plus facile, et donne à la carte une meilleure performance.

Le nombre de composants à placer sur la carte est 413 composants, dont un est de type BGA et un autre de type TFBGA (Thin-Prole Fine Pitch Ball GridArray) qui est une version plus petite et plus mince de BGA.

Pour notre carte, nous avons suivi les étapes suivantes :

- Placement du circuit FPGA au centre de la carte car il est relié à tous les autres composants.(figure IV.17)

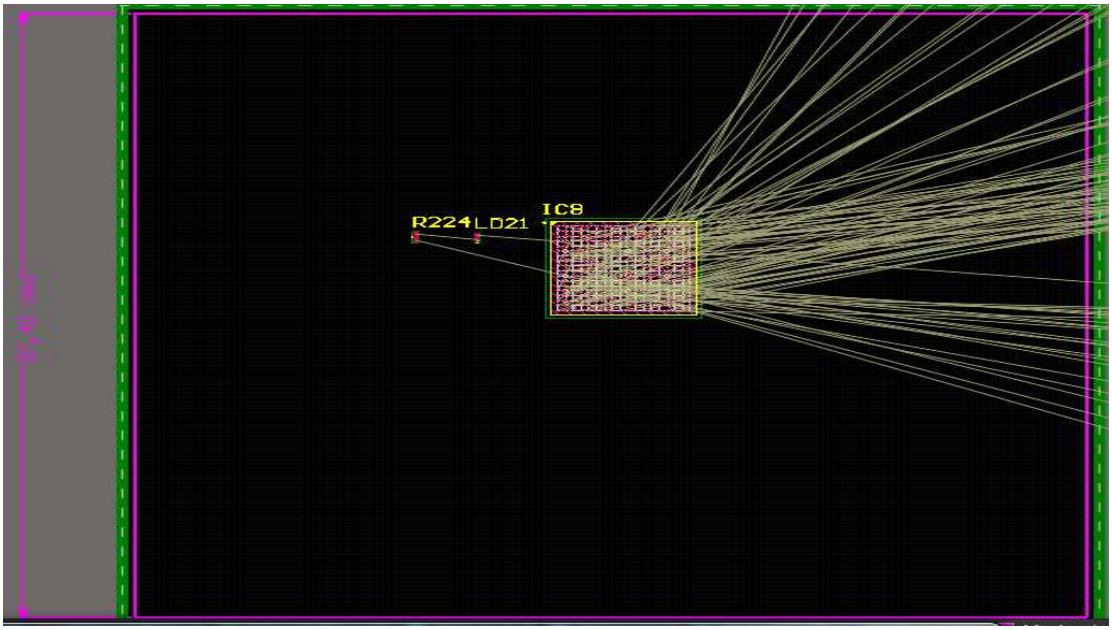


Figure IV.17. Placement du circuit FPGA

- Placement des autres composants de la manière dont laquelle ils sont regroupés dans le schéma électrique afin de minimiser la longueur des connexions.

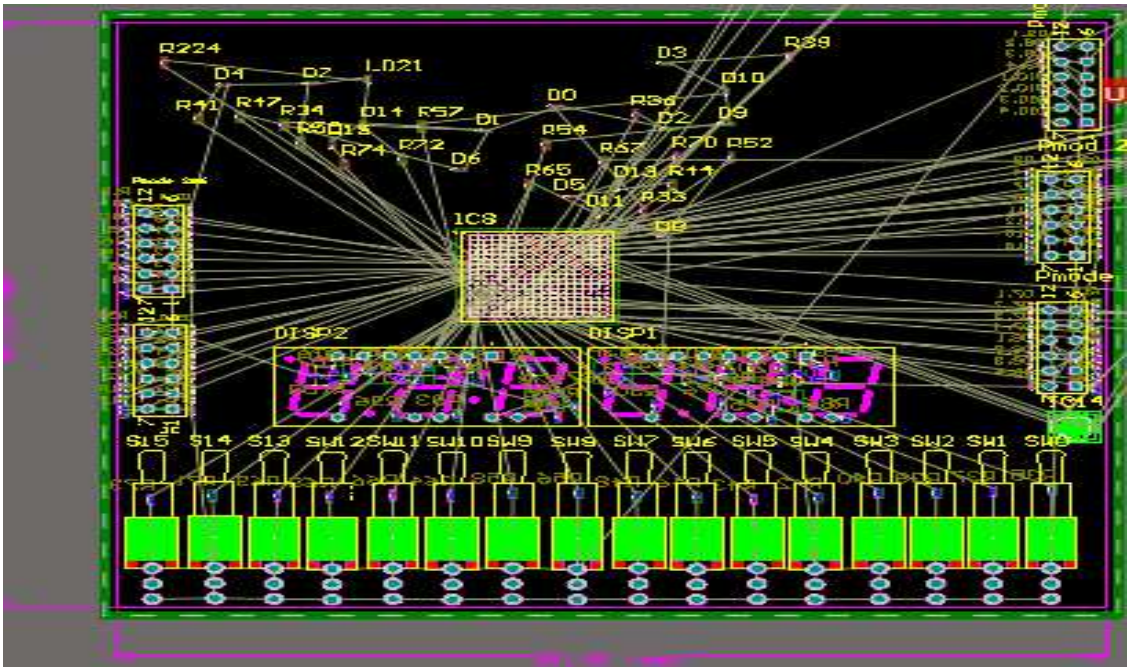


Figure IV.18. Exemple de placement des composants selon la carte FPGA

- Placement des composants traversant qui nécessitent le perçage sur la face supérieure de la carte pour que la soudure soit sur la face inférieure de celle-ci.

- Placement des composants montés en surface sur les deux faces de la carte.

- Orientation des composants similaires comme les résistances dans la même direction pour faciliter le soudage lors de l'assemblage.

- Placement des différents boutons, et LEDs sur la face supérieure de la carte. Chaque LED est placée à côté du composant dont elle indique l'état.
- Les condensateurs au Tantale peuvent être grands et difficiles à placer très près du circuit FPGA .Vu qu'ils fournissent une énergie à basse fréquence qui n'est pas aussi sensible à l'emplacement, ils peuvent être placés n' importe où sur la carte.
- Le placement des condensateurs céramiques 0402 est critique. Tout condensateur de ce type placé à l'extérieur de l'empreinte du FPGA que ce soit sur la face

supérieure ou inférieure, doit se trouver à moins de 0.5 pouce de la bordure extérieure de celle-ci

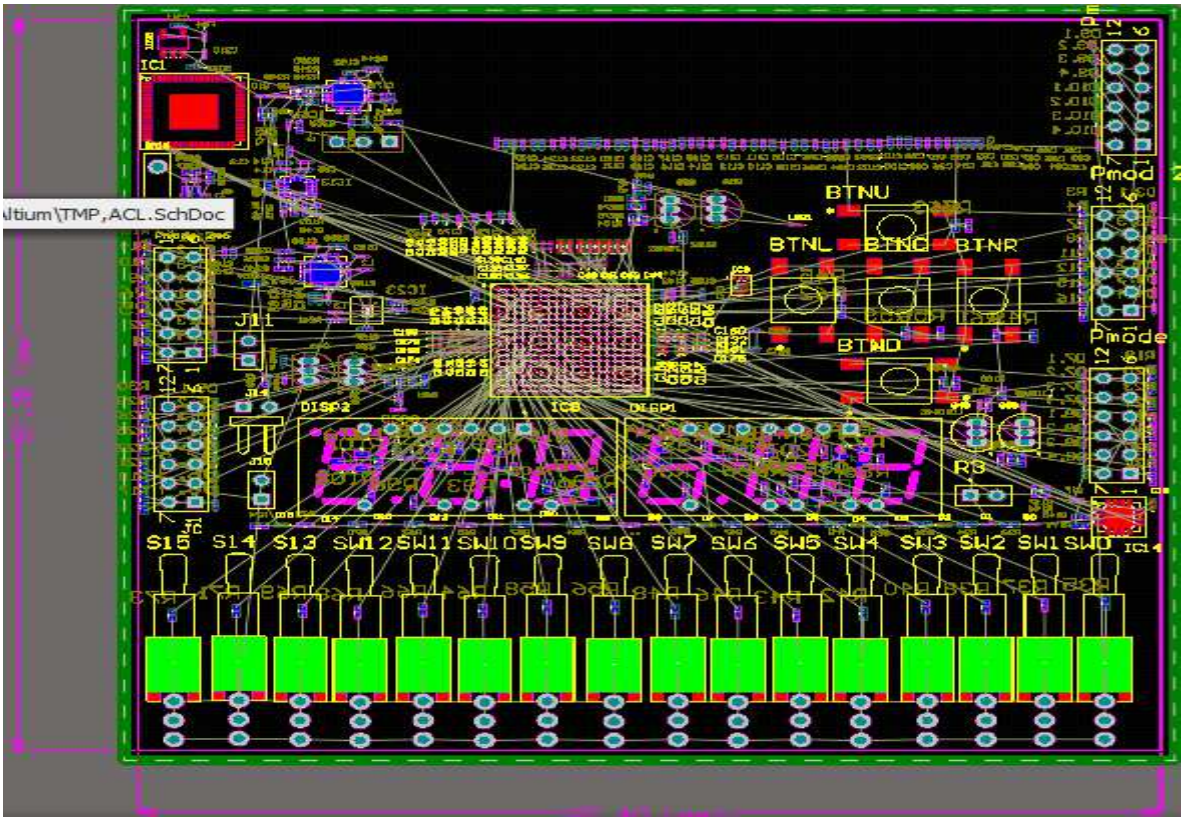


Figure IV.19.placement de tous les composants

- Pour les autres composants, il n'y a pas en général de raison logique pour commencer par un composant plutôt que par un autre. Les solutions envisagées peuvent cependant influer directement sur le tracé des pistes en le rendant simple ou compliqué. Au final, 195 sont placés sur la face supérieure du PCB tandis que 210 sont placés sur sa face inferieure.

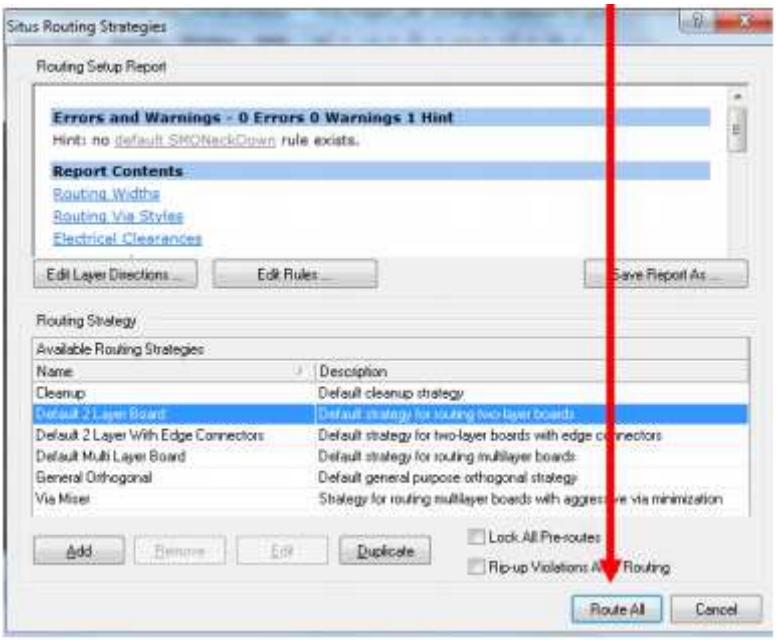
IV.4.3 Le routage

Cette opération consiste à chercher le meilleur passage pour une piste circulant parmi les autres pistes. Néanmoins, certaines solutions posent des problèmes de passage pour d'autres pistes, ce qui oblige à recommencer un autre routage, ou même de déplacer un composant.

Parmi les difficultés de routage de PCB les plus courantes se trouvent le routage de PMODE comportant un nombre élevé de broches tels que l'afficheur 7 segments. Pour cela, nous avons commencé le routage automatique directement pour éviter les problèmes de routage manuelle, cette dernier et très compliquer pour router manuellement.

- **Routage automatique :**

⇒ **Menu Autoroute ⇒ ALL ⇒ Route ALL**



Intérêt du routage automatique :

⇒ Un dessin de qualité respectant toutes les règles CEM ne peut être obtenu par un routage complet.

⇒ Nous n'utiliserons donc pas le routage automatique pour router notre carte.

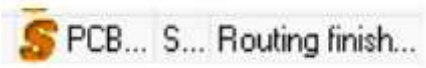
⇒ Toutefois cette fonction peut nous permettre de valider le placement de nos composants.

⇒ Pour une carte complexe, il est conseillé de dessiner plusieurs placements de composants.

⇒ En lançant le routeur automatique pour chaque placement vous pouvez comparer les résultats et retenir le placement optimal. (voir 8.1 Informations relatives au dessin de la carte)

⇒ Vous reprendrez ensuite le routage en manuel.

⇒ Lorsque le routage est fini vous obtenez le message :



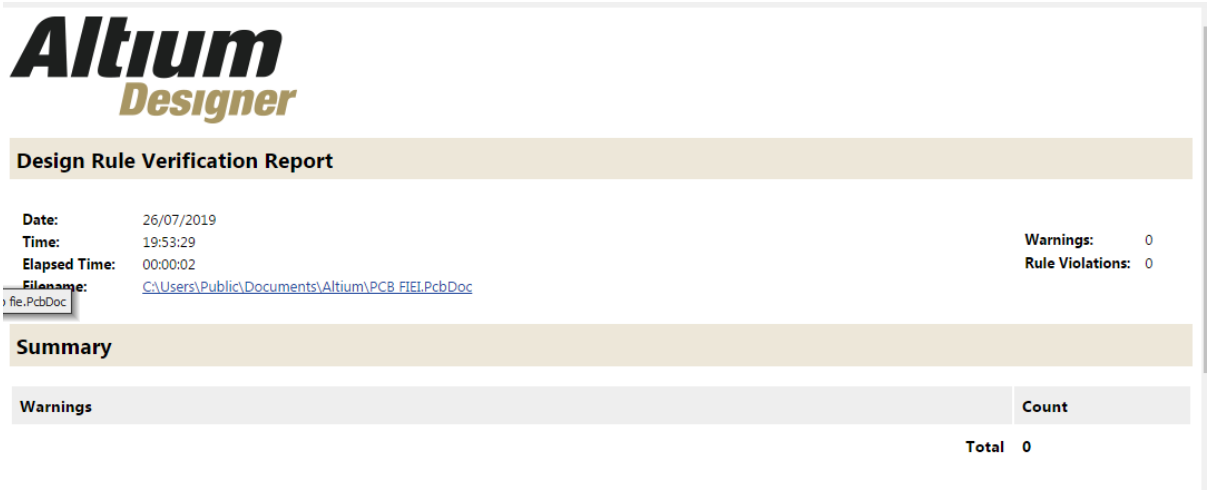
IV.4.4 Les avantage d'Active Route d'Altium Designer

L'un des grands avantages d'Active Route est qu'il fait partie intégrante de l'environnement de conception unifié d'Altium Designer. Cela élimine donc toute conversion compliquée ou transfert de données entre différents outils. Ainsi, il suffit de sélectionner les nœuds dont Active Route se chargera et d'appuyer sur le bouton « go ».

Active Route n'est pas un routeur automatique conventionnel, mais plutôt un routeur interactif guidé qui assure un routage de haute qualité pour un ensemble de nœuds choisis. Aussi, il prendra en charge les longueurs de nœud correspondantes ainsi que les paires différentielles [25].

IV.4.5 Validation du routage (DRC)

Une fois le routage terminé, il convient de s'assurer que les règles de dessin ont été respectées. Normalement, le routeur automatique nous indique les violations de règles de design. Cette validation s'appelle Design Rule Check (DRC).On y accède en cliquant sur **Tool/Design Rule Check** [25]. Au début du routage l'outil nous affiché 500 erreurs, que nous avons corrigé une par une et à la fin nous avons résolu tous les problèmes pour obtenir le résultat suivant (figure IV.20)


Altium Designer

Design Rule Verification Report

Date: 26/07/2019
Time: 19:53:29
Elapsed Time: 00:00:02
Filename: C:\Users\Public\Documents\Altium\PCB FIEI.PcbDoc

Warnings: 0
Rule Violations: 0

Summary

Warnings	Count
Total	0

Figure IV.20. Rapport de DRC

Les figures IV.21 et IV.22 .IV.33 montrent la carte en fin de routage en deux dimensions (2D) et trois dimensions (3D)

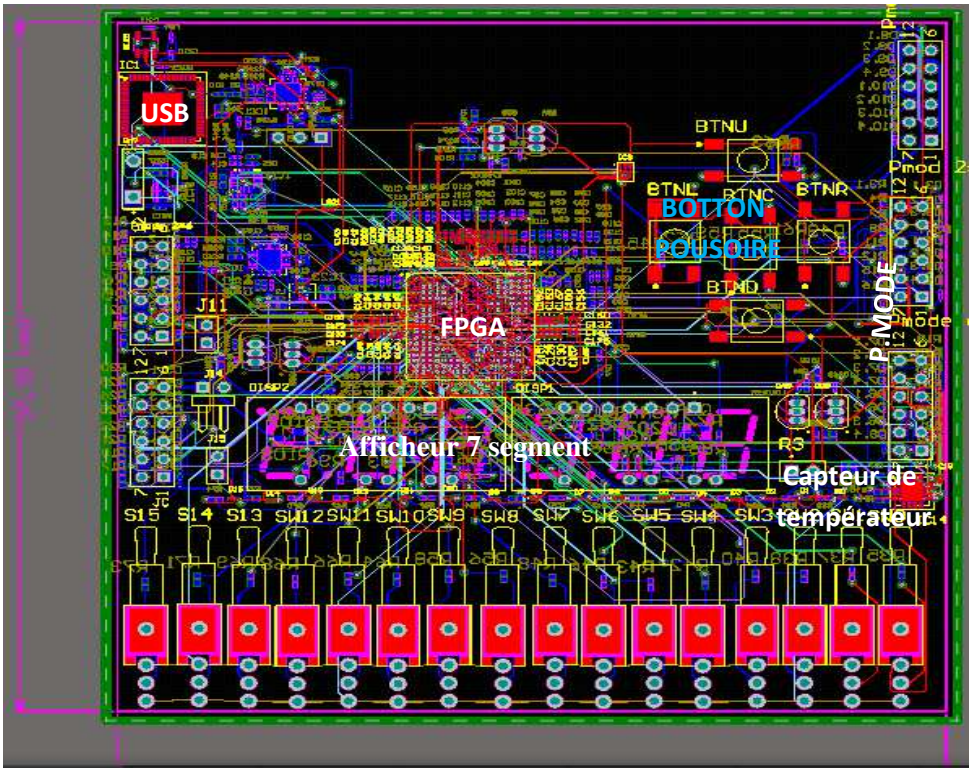


Figure IV.21. Le circuit imprimé en 2 dimensions

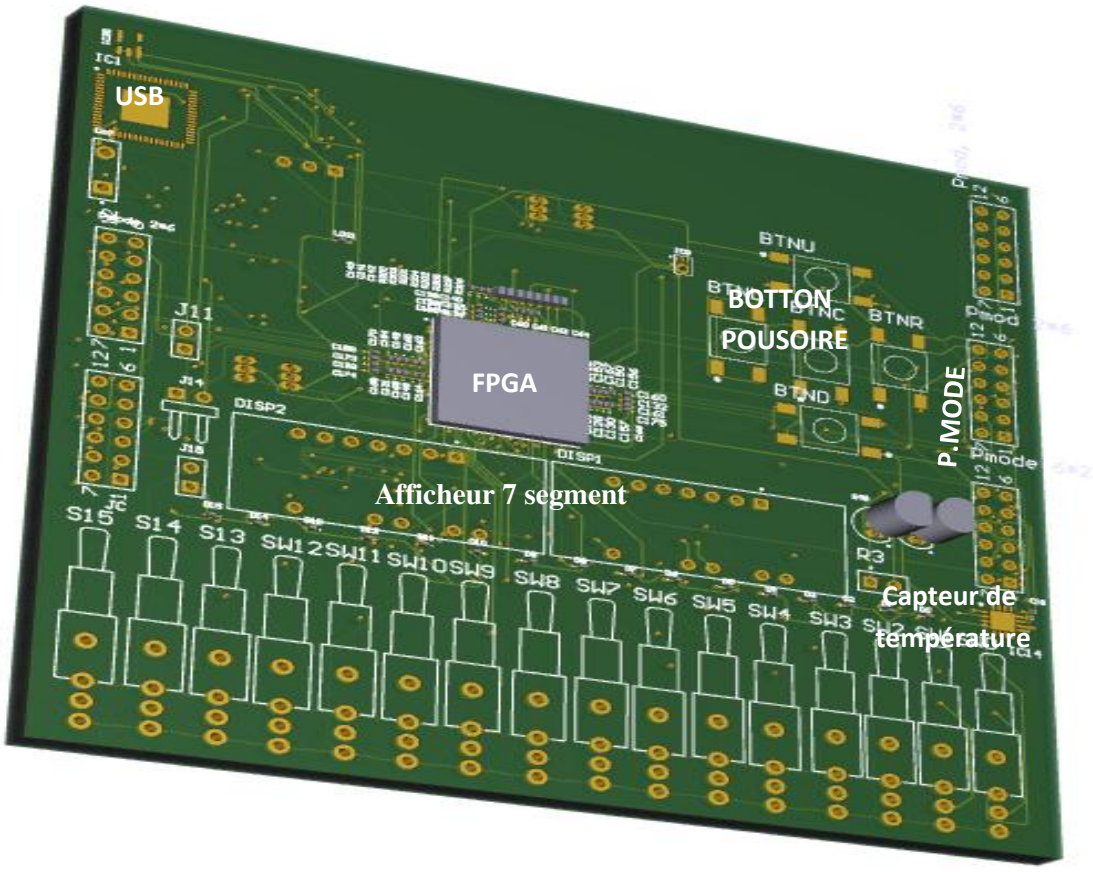


Figure IV.22. La face supérieure du circuit imprimé en 3 dimensions

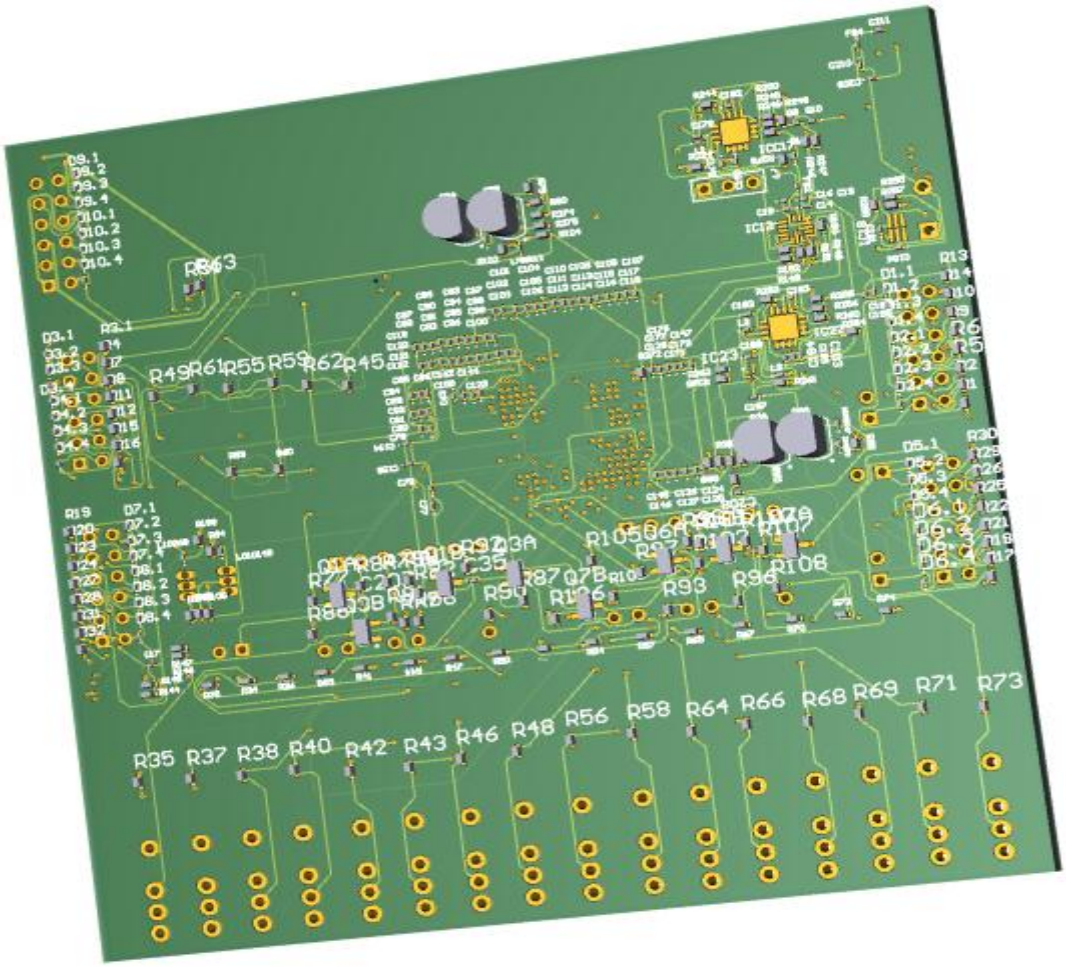


Figure IV.23.La face inférieure du circuit imprimé en 3 dimensions

IV.4.6 Création des fichiers de sortie

Altium propose différents fichiers de sortie standards tels que les fichiers Gerber. Créer un fichier de sortie standard est très utile lorsque le designer du PCB envoie son design au fabricant car cela assure la compatibilité des formats. Le fabricant pourra vérifier le design avec ces propres outils et apporter des modifications. Néanmoins, la fabrication consiste à transférer l'image de chaque couche du PCB sur des masques. Une manière simple d'assurer ce transfert est de créer une image à l'échelle du PCB [25].

- ***La nomenclature***

La nomenclature « bill of matériel « (catégorie : reports outputs) qui est un document de type rapport fournissant une liste de tous les composants requis pour construire le produit en spécifiant leurs différentes caractéristiques (voir annexe C). Le prix total estimé des composants utilisés revient à environ 792 Euro (106 920 Da).

- ***Le fichier gerber***

Le format gerber est très utilisé dans l'industrie des circuits imprimés. Il contient la description des diverses couches de connexions électriques (les pistes, les vias).Nous besoin d'un fichier image Gerber pour chacune des couches de conducteur, des couches de masque de soldat et des couches de légende. Nous suggérons que même si les images masque de soldat « solder mask » sont les mêmes pour les deux côtés, vous soumettez quand même un fichier. Bien que nous puissions accepter ces fichiers dans de nombreux formats. Le format gerber est très utilisé dans l'industrie des circuits imprimés. Il contient la description des diverses couches de connexions électriques (les pistes, les vias).

- ***Le dessin d'assemblage***

Ce fichier comporte un dessin qui indique la disposition des composants sur le circuit imprimé (catégorie : assembly outputs) Voire la figure IV.24

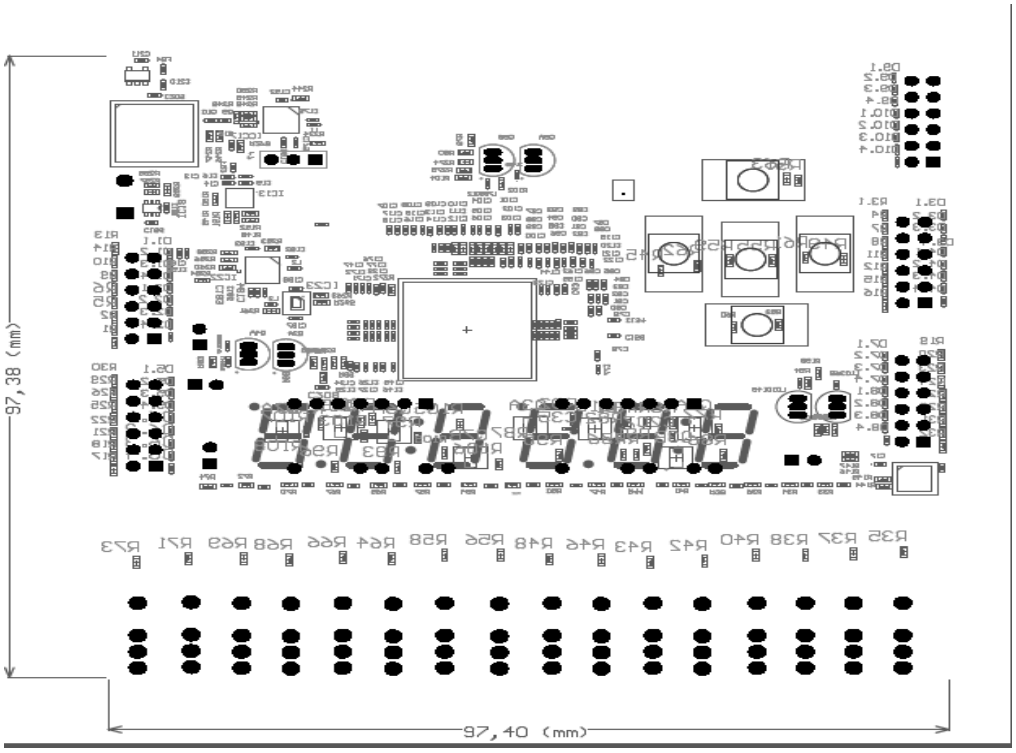


Figure IV.24. Dessin d'assemblage

IV.5 Conclusion

Dans ce chapitre, nous avons présenté la démarche adoptée pour la conception d'un circuit imprimé en utilisant l'outil de conception Altium Designer, depuis le schéma électrique jusqu'au routage en passant par le placement des composants sur la carte. La conception d'une telle carte nécessite un respect strict des règles de conception. Pour cela, nous avons essayé de prendre le maximum de ces règles et de les appliquer. Le résultat est un ensemble de fichiers qui peuvent être exploités en vue de la fabrication finale.

Conclusion générale

L'objectif du projet est de réaliser un circuit imprimé (PCB) multicouches d'un nœud de capteur basé sur FPGA. L'architecture de ce dernier est un système embarqué sur FPGA.

Vu la complexité du projet, nous avons proposé une méthodologie de conception basée sur l'utilisation de l'outil de conception des circuits PCB Altium Designer , en exploitant les résultats de synthèse, d'implémentation et d'analyse de l'architecture issues de l'outil VIVADO de Xilinx, pour en obtenir le fichier Gerber qui doit être transmis au fabriquant pour la fabrication du PCB.

L'utilisation de l'outil VIVADO est nécessaire car elle nous permet d'identifier les entrées/sorties du circuit FPGA. En outre, les résultats de synthèse, d'implémentation et d'analyse de la puissance, nous ont aidés à régler quelques paramètres nécessaires à la conception du circuit imprimé, à savoir : les circuits qui interfacent avec le FPGA, les différentes tensions d'alimentation au sein du PCB, le nombre de couches estimé du PCB ainsi que la surface estimée de celui-ci.

A partir du schéma électrique, nous avons proposé un plan de placement des différents composants sur la carte en tenant compte des contraintes liées à l'alimentation et aussi à certains composants comme le circuit FPGA, les régulateurs de tension, les résistances et les différents types de condensateurs .

Après le placement des composants vient le routage, nous avons proposé aussi un plan de routage tout en respectant un ensemble de règles imposés par les constructeurs des circuits FPGAs et par la technologie des PCB multicouches. Enfin, nous avons réussi à router 100% des pins. Les dimensions finales de la carte sont : 10x10 cm (4x4), et elle nécessite huit couches en total : deux couches pour les plan d'alimentation et six couches pour les signaux.

L'un des principaux avantages de cette méthodologie de conception est la réduction de la complexité et du temps de routage.

A la fin de la conception du PCB, nous avons généré plusieurs fichiers pour la fabrication du circuit imprimé comme le fichier Gerber et aussi la nomenclature "bill of matériel" et Le dessin d'assemblage.

Les résultats obtenus tant au niveau synthèse, analyse de la puissance qu'au niveau conception du circuit imprimé sont concluants et performants par rapports aux autres travaux de la littérature [21], puisque, nous avons réussi à diminuer à la fois, la puissance dissipée ainsi que la dimension de la carte [23].

Ce travail nous a permis d'approfondir nos connaissances et d'acquérir un savoir dans le domaine de la conception des PCB en général et plus précisément les PCB à base du FPGA.

Comme perspectives de ce travail, il serait intéressant de voir la possibilité de simuler le fonctionnement de la carte sur l'outil Altium Designer avant de l'envoyer pour la fabrication.

Bibliographie

[1] BENZID Sofiane « *Conception d'un système sur FPGA pour le traitement des données d'un nœud de capteurs »,* Mémoire de master, Ecole Nationale Supérieure de Technologie, 2017.

[2] C. Maxfield, Fpga architectures from 'a' to 'z': Part 1. EE Times Design, 2006.

[3] www.pcb-design.fr.

[4]https://fr.wikipedia.org/wiki/R%C3%A9seau_de_capteurs_sans_fil#Applications

[5] Yacine Younes, « Minimisation d'énergie dans un réseau de capteurs », mémoire de magister, Université Mouloud Mammeri de Tizi-Ouzou, Algérie, 2012

[6]https://fr.wikipedia.org/wiki/R%C3%A9seau_de_capteurs_sans_fil#Applications

[7]www.howstuffworks.com

[8] Mohamed BENAZZOUZ, « Surveillance de tout point d'une zone d'intérêt à l'aide d'un réseau de capteur multimédia sans fil », mémoire de magister, Ecole nationale supérieure d'informatique Oued- Smar Alger, Algérie, 2013.

[9]https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKw

[10]https://www.google.com/url?sa=i&source=images&cd=&ved=2ahUKEwi1mJD_sZ7jAhW

[11]https://www.google.com/url?sa=i&source=images&cd=&ved=2ahUKEwif_qGjtJ7jAhXM

[12]http://www.actepi.fr/new_actepi/Images/produits/saa/diagnostic2.jpg

[13]Antonio de la Piedra and al., « SensorSystemsBased on FPGAs : A Survey », Journal Sensors 2012, 12, ISSN: 1424-8220, pp: 12235-12264; doi:10.3390/s120912235.

[14]http://www.actepi.fr/new_actepi/Images/produits/saa/diagnostic2.jpg

[15] http://ww1.microchip.com/downloads/en/DeviceDoc/2467S.pdf

[16]http://www.xbow.com/pdf/ClassroomKits Press release.pdf

[17] http://www.pervcomconsulting.com/wsn.html.

[18] http://www.libelium.com/products/waspmote.

[19] Antonio de la Piedra, An Braeken and AbdellahTouhafi, « Sensor Systems Based on FPGAs and Their Applications »,*Sensors* open-access journal, University of Brussels, Belgium, revue 2012.

[20]Stephan Nolting, « NEO430 Processor », Juillet 2016

[21] Nexys 4 DDR FPGA Booard Reference Manual:reference.digilentinc.com.

[22]www.xilinx.com.

[23]D. Soumia et A.AIT ALLALA « Conception d'une carte de prototypage FPGA pour un serveur VoIP », Mémoire de master, Ecole Nationale Supérieure de Technologie, 2017.

[24] Docuent de xilinx : « 7 series FPGA Packaging and Pinout product spécification : UG475»: www. Xilinx.com

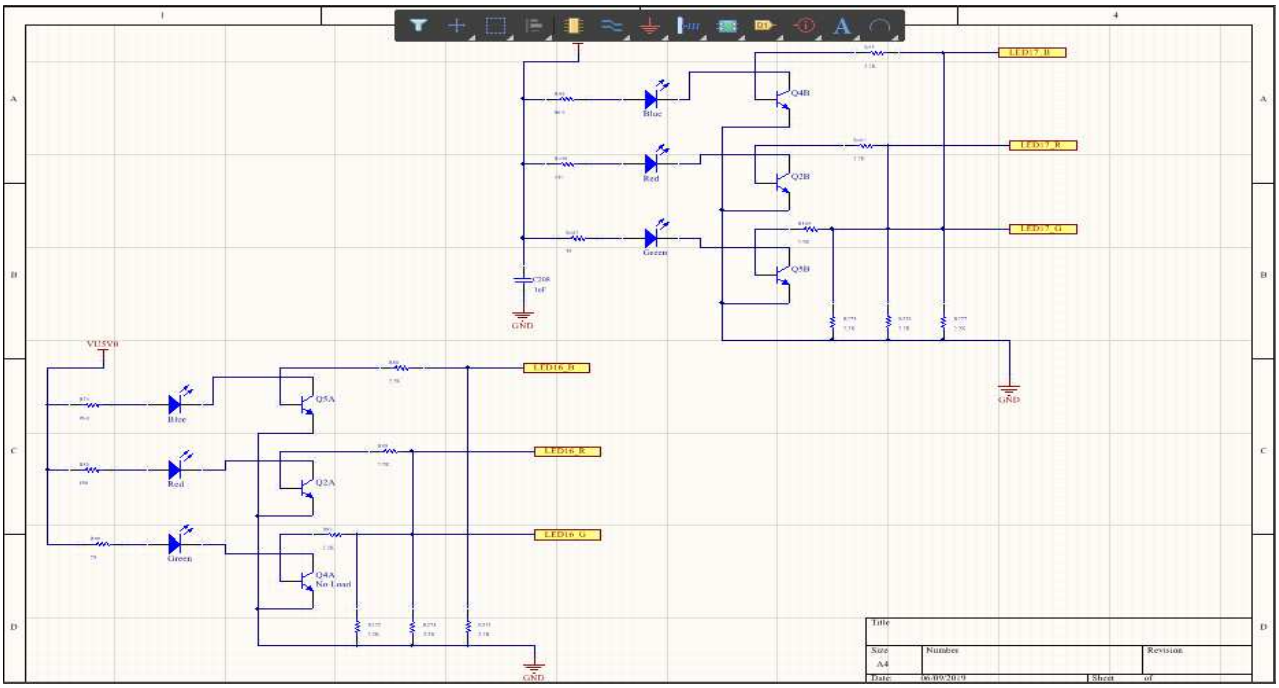
Annexe

Annexe A

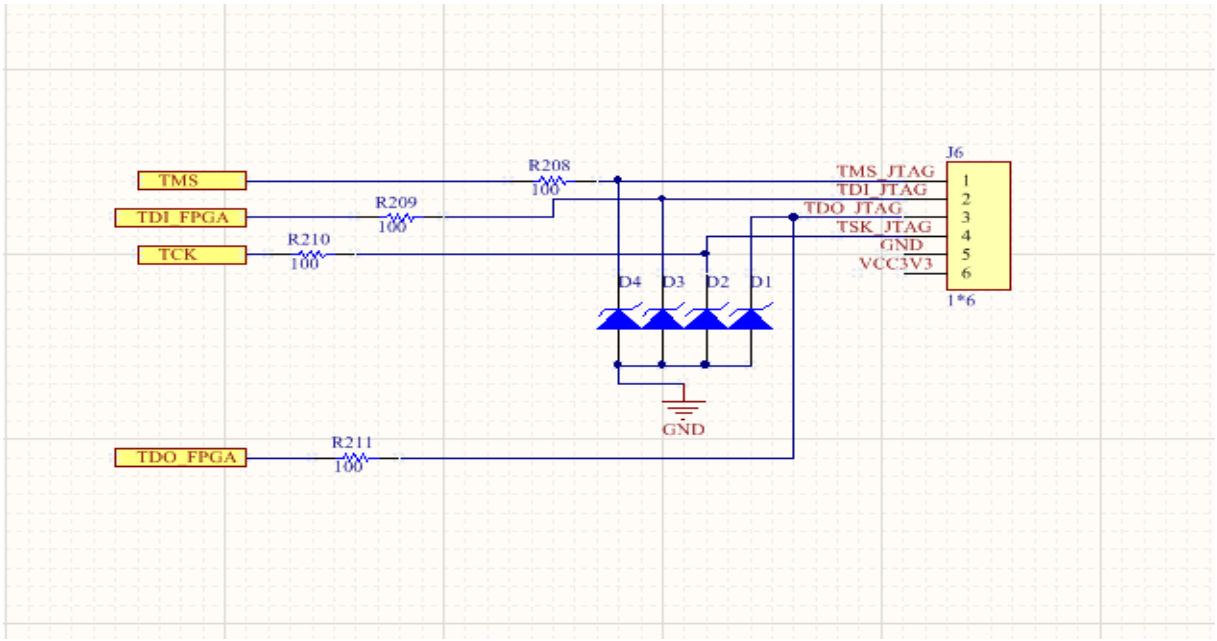
Schéma électrique du la carte électronique (quelques feuilles de schématique)

L'alimentation

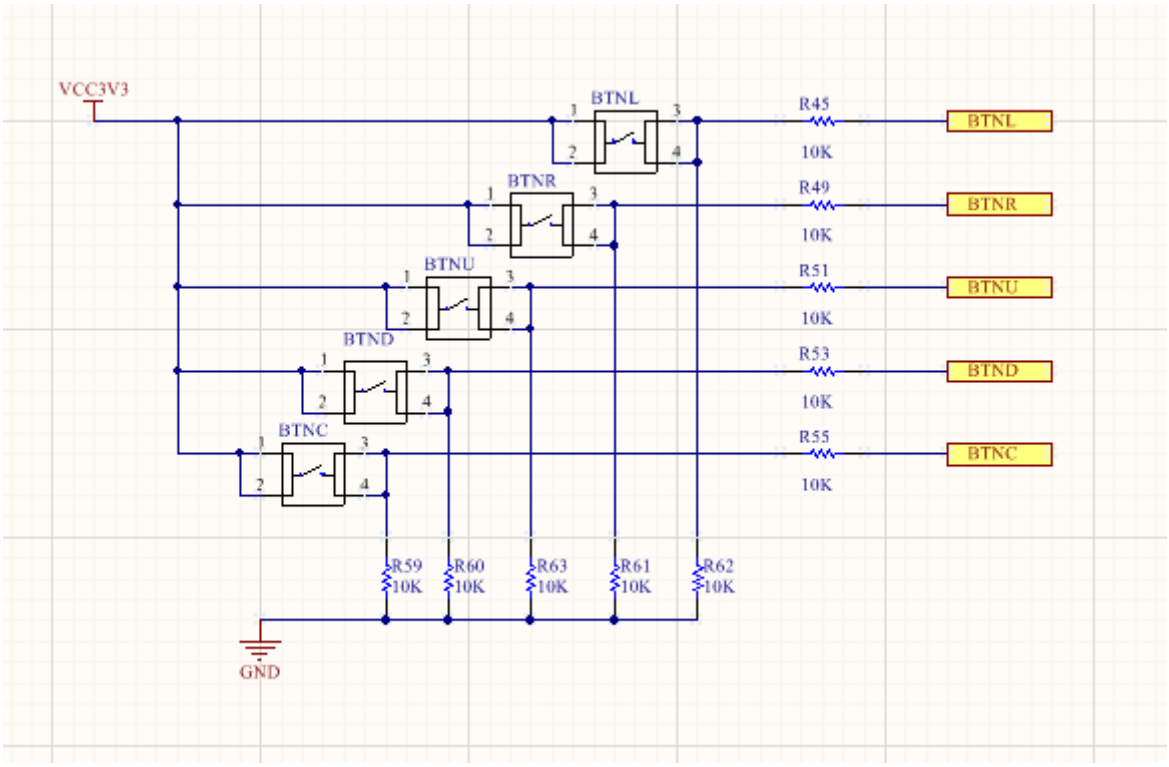
Les LED



Config JTG



Les Botton poussoir


VCC3V3
BTNL
BTNR
BTNU
BTND
BTNC
R45
10K
R49
10K
R51
10K
R53
10K
R55
10K
BTNL
BTNR
BTNU
BTND
BTNC
R59
10K
R60
10K
R63
10K
R61
10K
R62
10K
GND

Annexe B

Fichier de contrainte (XDC)

```
## Clock signal

#set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz

#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];


##Switches

#set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO_L24N_T3_RS0_15 Sch=sw[0]

#set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]

#set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]

#set_property -dict { PACKAGE_PIN R15 IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]

#set_property -dict { PACKAGE_PIN R17 IOSTANDARD LVCMOS33 } [get_ports { SW[4] }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]

#set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { SW[5] }]; #IO_L7N_T1_D10_14 Sch=sw[5]

#set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { SW[6] }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]

#set_property -dict { PACKAGE_PIN R13 IOSTANDARD LVCMOS33 } [get_ports { SW[7] }]; #IO_L5N_T0_D07_14 Sch=sw[7]

#set_property -dict { PACKAGE_PIN T8 IOSTANDARD LVCMOS18 } [get_ports { SW[8] }]; #IO_L24N_T3_34 Sch=sw[8]

#set_property -dict { PACKAGE_PIN U8 IOSTANDARD LVCMOS18 } [get_ports { SW[9] }]; #IO_25_34 Sch=sw[9]

#set_property -dict { PACKAGE_PIN R16 IOSTANDARD LVCMOS33 } [get_ports { SW[10] }]; #IO_L15P_T2_DQS_RDWR_B_14 Sch=sw[10]

#set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { SW[11] }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]

#set_property -dict { PACKAGE_PIN H6 IOSTANDARD LVCMOS33 } [get_ports { SW[12] }]; #IO_L24P_T3_35 Sch=sw[12]

#set_property -dict { PACKAGE_PIN U12 IOSTANDARD LVCMOS33 } [get_ports { SW[13] }]; #IO_L20P_T3_A08_D24_14 Sch=sw[13]

#set_property -dict { PACKAGE_PIN U11 IOSTANDARD LVCMOS33 } [get_ports { SW[14] }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]

#set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { SW[15] }]; #IO_L21P_T3_DQS_14 Sch=sw[15]


## LEDs

#set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { LED[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]

#set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { LED[1] }]; #IO_L24P_T3_RS1_15 Sch=led[1]

#set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { LED[2] }]; #IO_L17N_T2_A25_15 Sch=led[2]

#set_property -dict { PACKAGE_PIN N14 IOSTANDARD LVCMOS33 } [get_ports { LED[3] }]; #IO_L8P_T1_D11_14 Sch=led[3]

#set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { LED[4] }]; #IO_L7P_T1_D09_14 Sch=led[4]

#set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { LED[5] }]; #IO_L18N_T2_A11_D27_14 Sch=led[5]
```

#set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { LED[6] }]; #IO_L17P_T2_A14_D30_14 Sch=led[6]

#set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { LED[7] }]; #IO_L18P_T2_A12_D28_14 Sch=led[7]

#set_property -dict { PACKAGE_PIN V16 IOSTANDARD LVCMOS33 } [get_ports { LED[8] }]; #IO_L16N_T2_A15_D31_14 Sch=led[8]

#set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { LED[9] }]; #IO_L14N_T2_SRCC_14 Sch=led[9]

#set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { LED[10] }]; #IO_L22P_T3_A05_D21_14 Sch=led[10]

#set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { LED[11] }]; #IO_L15N_T2_DQS_DOUT_CSO_B_14 Sch=led[11]

#set_property -dict { PACKAGE_PIN V15 IOSTANDARD LVCMOS33 } [get_ports { LED[12] }]; #IO_L16P_T2_CSI_B_14 Sch=led[12]

#set_property -dict { PACKAGE_PIN V14 IOSTANDARD LVCMOS33 } [get_ports { LED[13] }]; #IO_L22N_T3_A04_D20_14 Sch=led[13]

#set_property -dict { PACKAGE_PIN V12 IOSTANDARD LVCMOS33 } [get_ports { LED[14] }]; #IO_L20N_T3_A07_D23_14 Sch=led[14]

#set_property -dict { PACKAGE_PIN V11 IOSTANDARD LVCMOS33 } [get_ports { LED[15] }]; #IO_L21N_T3_DQS_A06_D22_14 Sch=led[15]

#set_property -dict { PACKAGE_PIN R12 IOSTANDARD LVCMOS33 } [get_ports { LED16_B }]; #IO_L5P_T0_D06_14 Sch=led16_b

#set_property -dict { PACKAGE_PIN M16 IOSTANDARD LVCMOS33 } [get_ports { LED16_G }]; #IO_L10P_T1_D14_14 Sch=led16_g

#set_property -dict { PACKAGE_PIN N15 IOSTANDARD LVCMOS33 } [get_ports { LED16_R }]; #IO_L11P_T1_SRCC_14 Sch=led16_r

#set_property -dict { PACKAGE_PIN G14 IOSTANDARD LVCMOS33 } [get_ports { LED17_B }]; #IO_L15N_T2_DQS_ADV_B_15 Sch=led17_b

#set_property -dict { PACKAGE_PIN R11 IOSTANDARD LVCMOS33 } [get_ports { LED17_G }]; #IO_0_14 Sch=led17_g

#set_property -dict { PACKAGE_PIN N16 IOSTANDARD LVCMOS33 } [get_ports { LED17_R }]; #IO_L11N_T1_SRCC_14 Sch=led17_r

##7 segment display

#set_property -dict { PACKAGE_PIN T10 IOSTANDARD LVCMOS33 } [get_ports { CA }]; #IO_L24N_T3_A00_D16_14 Sch=ca

#set_property -dict { PACKAGE_PIN R10 IOSTANDARD LVCMOS33 } [get_ports { CB }]; #IO_25_14 Sch=cb

#set_property -dict { PACKAGE_PIN K16 IOSTANDARD LVCMOS33 } [get_ports { CC }]; #IO_25_15 Sch=cc

#set_property -dict { PACKAGE_PIN K13 IOSTANDARD LVCMOS33 } [get_ports { CD }]; #IO_L17P_T2_A26_15 Sch=cd

#set_property -dict { PACKAGE_PIN P15 IOSTANDARD LVCMOS33 } [get_ports { CE }]; #IO_L13P_T2_MRCC_14 Sch=ce

#set_property -dict { PACKAGE_PIN T11 IOSTANDARD LVCMOS33 } [get_ports { CF }]; #IO_L19P_T3_A10_D26_14 Sch=cf

#set_property -dict { PACKAGE_PIN L18 IOSTANDARD LVCMOS33 } [get_ports { CG }]; #IO_L4P_T0_D04_14 Sch=cg

#set_property -dict { PACKAGE_PIN H15 IOSTANDARD LVCMOS33 } [get_ports { DP }]; #IO_L19N_T3_A21_VREF_15 Sch=dp

#set_property -dict { PACKAGE_PIN J17 IOSTANDARD LVCMOS33 } [get_ports { AN[0] }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]

#set_property -dict { PACKAGE_PIN J18 IOSTANDARD LVCMOS33 } [get_ports { AN[1] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]

#set_property -dict { PACKAGE_PIN T9 IOSTANDARD LVCMOS33 } [get_ports { AN[2] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]

#set_property -dict { PACKAGE_PIN J14 IOSTANDARD LVCMOS33 } [get_ports { AN[3] }]; #IO_L19P_T3_A22_15 Sch=an[3]

#set_property -dict { PACKAGE_PIN P14 IOSTANDARD LVCMOS33 } [get_ports { AN[4] }]; #IO_L8N_T1_D12_14 Sch=an[4]

#set_property -dict { PACKAGE_PIN T14 IOSTANDARD LVCMOS33 } [get_ports { AN[5] }]; #IO_L14P_T2_SRCC_14 Sch=an[5]

#set_property -dict { PACKAGE_PIN K2 IOSTANDARD LVCMOS33 } [get_ports { AN[6] }]; #IO_L23P_T3_35 Sch=an[6]

#set_property -dict { PACKAGE_PIN U13 IOSTANDARD LVCMOS33 } [get_ports { AN[7] }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]

##Buttons

#set_property -dict { PACKAGE_PIN C12 IOSTANDARD LVCMOS33 } [get_ports { CPU_RESETN }]; #IO_L3P_T0_DQS_AD1P_15 Sch=cpu_resetn

#set_property -dict { PACKAGE_PIN N17 IOSTANDARD LVCMOS33 } [get_ports { BTNC }]; #IO_L9P_T1_DQS_14 Sch=btnc

#set_property -dict { PACKAGE_PIN M18 IOSTANDARD LVCMOS33 } [get_ports { BTNU }]; #IO_L4N_T0_D05_14 Sch=btnu

#set_property -dict { PACKAGE_PIN P17 IOSTANDARD LVCMOS33 } [get_ports { BTNL }]; #IO_L12P_T1_MRCC_14 Sch=btnl

#set_property -dict { PACKAGE_PIN M17 IOSTANDARD LVCMOS33 } [get_ports { BTNR }]; #IO_L10N_T1_D15_14 Sch=btnr

#set_property -dict { PACKAGE_PIN P18 IOSTANDARD LVCMOS33 } [get_ports { BTND }]; #IO_L9N_T1_DQS_D13_14 Sch=btnd

##Pmod Headers

##Pmod Header JA

#set_property -dict { PACKAGE_PIN C17 IOSTANDARD LVCMOS33 } [get_ports { JA[1] }]; #IO_L20N_T3_A19_15 Sch=ja[1]

#set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMOS33 } [get_ports { JA[2] }]; #IO_L21N_T3_DQS_A18_15 Sch=ja[2]

#set_property -dict { PACKAGE_PIN E18 IOSTANDARD LVCMOS33 } [get_ports { JA[3] }]; #IO_L21P_T3_DQS_15 Sch=ja[3]

#set_property -dict { PACKAGE_PIN G17 IOSTANDARD LVCMOS33 } [get_ports { JA[4] }]; #IO_L18N_T2_A23_15 Sch=ja[4]

#set_property -dict { PACKAGE_PIN D17 IOSTANDARD LVCMOS33 } [get_ports { JA[7] }]; #IO_L16N_T2_A27_15 Sch=ja[7]

#set_property -dict { PACKAGE_PIN E17 IOSTANDARD LVCMOS33 } [get_ports { JA[8] }]; #IO_L16P_T2_A28_15 Sch=ja[8]

#set_property -dict { PACKAGE_PIN F18 IOSTANDARD LVCMOS33 } [get_ports { JA[9] }]; #IO_L22N_T3_A16_15 Sch=ja[9]

#set_property -dict { PACKAGE_PIN G18 IOSTANDARD LVCMOS33 } [get_ports { JA[10] }]; #IO_L22P_T3_A17_15 Sch=ja[10]

##Pmod Header JB

#set_property -dict { PACKAGE_PIN D14 IOSTANDARD LVCMOS33 } [get_ports { JB[1] }]; #IO_L1P_T0_AD0P_15 Sch=jb[1]

#set_property -dict { PACKAGE_PIN F16 IOSTANDARD LVCMOS33 } [get_ports { JB[2] }]; #IO_L14N_T2_SRCC_15 Sch=jb[2]

#set_property -dict { PACKAGE_PIN G16 IOSTANDARD LVCMOS33 } [get_ports { JB[3] }]; #IO_L13N_T2_MRCC_15 Sch=jb[3]

#set_property -dict { PACKAGE_PIN H14 IOSTANDARD LVCMOS33 } [get_ports { JB[4] }]; #IO_L15P_T2_DQS_15 Sch=jb[4]

#set_property -dict { PACKAGE_PIN E16 IOSTANDARD LVCMOS33 } [get_ports { JB[7] }]; #IO_L11N_T1_SRCC_15 Sch=jb[7]

#set_property -dict { PACKAGE_PIN F13 IOSTANDARD LVCMOS33 } [get_ports { JB[8] }]; #IO_L5P_T0_AD9P_15 Sch=jb[8]

#set_property -dict { PACKAGE_PIN G13 IOSTANDARD LVCMOS33 } [get_ports { JB[9] }]; #IO_0_15 Sch=jb[9]

#set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports { JB[10] }]; #IO_L13P_T2_MRCC_15 Sch=jb[10]

##Pmod Header JC

#set_property -dict { PACKAGE_PIN K1 IOSTANDARD LVCMOS33 } [get_ports { JC[1] }]; #IO_L23N_T3_35 Sch=jc[1]

#set_property -dict { PACKAGE_PIN F6 IOSTANDARD LVCMOS33 } [get_ports { JC[2] }]; #IO_L19N_T3_VREF_35 Sch=jc[2]

#set_property -dict { PACKAGE_PIN J2 IOSTANDARD LVCMOS33 } [get_ports { JC[3] }]; #IO_L22N_T3_35 Sch=jc[3]

#set_property -dict { PACKAGE_PIN G6 IOSTANDARD LVCMOS33 } [get_ports { JC[4] }]; #IO_L19P_T3_35 Sch=jc[4]

#set_property -dict { PACKAGE_PIN E7 IOSTANDARD LVCMOS33 } [get_ports { JC[7] }]; #IO_L6P_T0_35 Sch=jc[7]

#set_property -dict { PACKAGE_PIN J3 IOSTANDARD LVCMOS33 } [get_ports { JC[8] }]; #IO_L22P_T3_35 Sch=jc[8]

#set_property -dict { PACKAGE_PIN J4 IOSTANDARD LVCMOS33 } [get_ports { JC[9] }]; #IO_L21P_T3_DQS_35 Sch=jc[9]

#set_property -dict { PACKAGE_PIN E6 IOSTANDARD LVCMOS33 } [get_ports { JC[10] }]; #IO_L5P_T0_AD13P_35 Sch=jc[10]

##Pmod Header JD

#set_property -dict { PACKAGE_PIN H4 IOSTANDARD LVCMOS33 } [get_ports { JD[1] }]; #IO_L21N_T3_DQS_35 Sch=jd[1]

#set_property -dict { PACKAGE_PIN H1 IOSTANDARD LVCMOS33 } [get_ports { JD[2] }]; #IO_L17P_T2_35 Sch=jd[2]

#set_property -dict { PACKAGE_PIN G1 IOSTANDARD LVCMOS33 } [get_ports { JD[3] }]; #IO_L17N_T2_35 Sch=jd[3]

#set_property -dict { PACKAGE_PIN G3 IOSTANDARD LVCMOS33 } [get_ports { JD[4] }]; #IO_L20N_T3_35 Sch=jd[4]

#set_property -dict { PACKAGE_PIN H2 IOSTANDARD LVCMOS33 } [get_ports { JD[7] }]; #IO_L15P_T2_DQS_35 Sch=jd[7]

#set_property -dict { PACKAGE_PIN G4 IOSTANDARD LVCMOS33 } [get_ports { JD[8] }]; #IO_L20P_T3_35 Sch=jd[8]

#set_property -dict { PACKAGE_PIN G2 IOSTANDARD LVCMOS33 } [get_ports { JD[9] }]; #IO_L15N_T2_DQS_35 Sch=jd[9]

#set_property -dict { PACKAGE_PIN F3 IOSTANDARD LVCMOS33 } [get_ports { JD[10] }]; #IO_L13N_T2_MRCC_35 Sch=jd[10]

##Pmod Header JXADC

#set_property -dict { PACKAGE_PIN A14 IOSTANDARD LVDS } [get_ports { XA_N[1] }]; #IO_L9N_T1_DQS_AD3N_15 Sch=xa_n[1]

#set_property -dict { PACKAGE_PIN A13 IOSTANDARD LVDS } [get_ports { XA_P[1] }]; #IO_L9P_T1_DQS_AD3P_15 Sch=xa_p[1]

#set_property -dict { PACKAGE_PIN A16 IOSTANDARD LVDS } [get_ports { XA_N[2] }]; #IO_L8N_T1_AD10N_15 Sch=xa_n[2]

#set_property -dict { PACKAGE_PIN A15 IOSTANDARD LVDS } [get_ports { XA_P[2] }]; #IO_L8P_T1_AD10P_15 Sch=xa_p[2]

#set_property -dict { PACKAGE_PIN B17 IOSTANDARD LVDS } [get_ports { XA_N[3] }]; #IO_L7N_T1_AD2N_15 Sch=xa_n[3]

#set_property -dict { PACKAGE_PIN B16 IOSTANDARD LVDS } [get_ports { XA_P[3] }]; #IO_L7P_T1_AD2P_15 Sch=xa_p[3]

#set_property -dict { PACKAGE_PIN A18 IOSTANDARD LVDS } [get_ports { XA_N[4] }]; #IO_L10N_T1_AD11N_15 Sch=xa_n[4]

#set_property -dict { PACKAGE_PIN B18 IOSTANDARD LVDS } [get_ports { XA_P[4] }]; #IO_L10P_T1_AD11P_15 Sch=xa_p[4]

##Accelerometer

#set_property -dict { PACKAGE_PIN E15 IOSTANDARD LVCMOS33 } [get_ports { ACL_MISO }]; #IO_L11P_T1_SRCC_15 Sch=acl_miso

#set_property -dict { PACKAGE_PIN F14 IOSTANDARD LVCMOS33 } [get_ports { ACL_MOSI }]; #IO_L5N_T0_AD9N_15 Sch=acl_mosi

#set_property -dict { PACKAGE_PIN F15 IOSTANDARD LVCMOS33 } [get_ports { ACL_SCLK }]; #IO_L14P_T2_SRCC_15 Sch=acl_sclk

#set_property -dict { PACKAGE_PIN D15 IOSTANDARD LVCMOS33 } [get_ports { ACL_CSN }]; #IO_L12P_T1_MRCC_15 Sch=acl_csn

#set_property -dict { PACKAGE_PIN B13 IOSTANDARD LVCMOS33 } [get_ports { ACL_INT[1] }]; #IO_L2P_T0_AD8P_15 Sch=acl_int[1]

#set_property -dict { PACKAGE_PIN C16 IOSTANDARD LVCMOS33 } [get_ports { ACL_INT[2] }]; #IO_L20P_T3_A20_15 Sch=acl_int[2]

##Temperature Sensor

#set_property -dict { PACKAGE_PIN C14 IOSTANDARD LVCMOS33 } [get_ports { TMP_SCL }]; #IO_L1N_T0_AD0N_15 Sch=tmp_scl

#set_property -dict { PACKAGE_PIN C15 IOSTANDARD LVCMOS33 } [get_ports { TMP_SDA }];

#IO_L12N_T1_MRCC_15 Sch=tmp_sda

#set_property -dict { PACKAGE_PIN D13 IOSTANDARD LVCMOS33 } [get_ports { TMP_INT }]; #IO_L6N_T0_VREF_15 Sch=tmp_int

#set_property -dict { PACKAGE_PIN B14 IOSTANDARD LVCMOS33 } [get_ports { TMP_CT }]; #IO_L2N_T0_AD8N_15 Sch=tmp_ct

##Omnidirectional Microphone

#set_property -dict { PACKAGE_PIN J5 IOSTANDARD LVCMOS33 } [get_ports { M_CLK }]; #IO_25_35 Sch=m_clk

#set_property -dict { PACKAGE_PIN H5 IOSTANDARD LVCMOS33 } [get_ports { M_DATA }]; #IO_L24N_T3_35 Sch=m_data

#set_property -dict { PACKAGE_PIN F5 IOSTANDARD LVCMOS33 } [get_ports { M_LRSEL }]; #IO_0_35 Sch=m_lrsel

##PWM Audio Amplifier

#set_property -dict { PACKAGE_PIN A11 IOSTANDARD LVCMOS33 } [get_ports { AUD_PWM }]; #IO_L4N_T0_15 Sch=aud_pwm

#set_property -dict { PACKAGE_PIN D12 IOSTANDARD LVCMOS33 } [get_ports { AUD_SD }]; #IO_L6P_T0_15 Sch=aud_sd

##USB-RS232 Interface

#set_property -dict { PACKAGE_PIN C4 IOSTANDARD LVCMOS33 } [get_ports { UART_TXD_IN }]; #IO_L7P_T1_AD6P_35 Sch=uart_txd_in

#set_property -dict { PACKAGE_PIN D4 IOSTANDARD LVCMOS33 } [get_ports { UART_RXD_OUT }]; #IO_L11N_T1_SRCC_35 Sch=uart_rxd_out

#set_property -dict { PACKAGE_PIN D3 IOSTANDARD LVCMOS33 } [get_ports { UART_CTS }]; #IO_L12N_T1_MRCC_35 Sch=uart_cts

#set_property -dict { PACKAGE_PIN E5 IOSTANDARD LVCMOS33 } [get_ports { UART_RTS }]; #IO_L5N_T0_AD13N_35 Sch=uart_rts