République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Batna 2 – Mostefa Ben Boulaïd
Faculté de Technologie
Département d'Electronique

Thèse

Préparée au sein du Laboratoire Electronique de Puissance Commande Industrielle et du Laboratoire d'Automatique Avancée et d'Analyse des Système

Présentée pour l'obtention du diplôme de :

Doctorat en Sciences en Electronique

Option : électronique

Sous le Thème :

Conception et Implémentation d'IPs pour des Applications de Traitement Signal sur FPGA

Présentée par :

Mr RADJAH Fayçal

Devant le jury composé de :

M. SAIDI Lamine	Prof.	Université-Batna-2 Batna	Président
M. ZIET Lahcene	Prof.	Université de Sétif-1	Rapporteur
M. BENOUDJIT Nabil	Prof.	Université-Batna-2 Batna	Co-Rapporteur
M. FERHAT HAMIDA Abdelhak	Prof.	Université -Sétif-1 Sétif	Examinateur

Mars 2022

REMERCIEMENTS

J’exprime toute ma gratitude et reconnaissance à mes directeurs de thèse Monsieur ZIET Lahcene, Professeur à l’université de Sétif et Monsieur BENOUDJIT Nabil, Professeur à l’université de Barna, pour les précieux conseils qu’ils m’ont su donnés et l’attention et le sérieux avec lesquels ils ont suivi de près mes travaux de recherche.

Je remercie chaleureusement Monsieur SAIDI Lamine, Professeur à l’université de Batna, pour m’avoir fait l’honneur de présider le jury.

Je remercie également Messieurs FERHAT-Hamida Abdelhak, Professeur à l’université de Sétif pour l’honneur qu’il me fait d’avoir accepté d’examiner mon travail.

Resumé

Actuellement, les circuits reconfigurables à base d'FPGA avec leurs outils de synthèse haut niveau ''HLS'' sont devenus une technologie très prometteuse qui sera largement utilisée dans de nombreuses applications englobant tous les aspects et exigences des systèmes embarqués.
Cette thèse présente la conception de circuits électroniques de calcul de l'histogramme et de seuil de binarisation sous forme d'IPs pour la mise en œuvre d'opérations de traitement d'image morphologique incluant la segmentation, l'érosion et le filtrage linéaire. La méthode de mise en œuvre est basée sur une approche de conception matérielle/logicielle (HW/SW) utilisant des outils de synthèse de Intel-Altera (quartus, Qsys, Modelsim). La plate-forme Altera-DE2 cyclone FPGA associée à un périphérique Nios-II est utilisée pour ce travail dans lequel est connecté à un PC via une liaison JTAG. L'image capturée ou reçue via une transmission est transmise au FPGA pour subir un traitement de binarisation par la recherche d'un seuil basé sur l'histogramme. Le résultat sera ensuite renvoyé pour être affichée sur l'interface VGA ou stockée dans une base de données, en temps réel via le système réseau pour être exploité par d'autres taches. Les résultats expérimentaux démontrent la faisabilité de l'approche proposée et celle-ci peut être étendue à d'autres applications.

Abstract:

Currently, reconfigurable FPGA-based circuits with their high-level "HLS" synthesis tools have become a very promising technology that will be widely used in many applications encompassing all aspects and requirements of embedded systems.
This thesis presents the design of electronics IP circuits i such as histogram compute-IP, threshold compute-IP to implemente morphological image processing operations including segmentation, erosion and linear filtering .
The implementation method is based on hardware / software (HW / SW) co-design approach using synthesis tools from Intel-Altera (quartus, Qsys, Modelsim). The Altera-DE 2 –cyclone FPGA platform associated with a Nios-II peripheral is used for this work in which is connected to a PC via a JTAG link. The image captured or received via transmission is transmitted to the FPGA to undergo binarization processing by searching for a threshold based on the histogram. The result will then be returned to be displayed on the VGA interface or stored in a database, in real time

via the network system to be exploited by other tasks. The experimental results demonstrate the feasibility of the proposed approach and it can be extended to other applications.

ملخص

في الوقت الحالي ، أصبحت الدارات الإلكترونية القائمة على القابلة لإعادة التكوين FPGA مع أدوات التوليف عالية المستوى "HLS" تقنية واعدة للغاية سيتم استخدامها على نطاق واسع في العديد من التطبيقات التي تشمل جميع جوانب ومتطلبات الأنظمة المدمجة. تقدم هذه الأطروحة تصميم الدوائر في شكل IP مثل حساب الرسم البياني ، وحساب العتبة لتنفيذ عمليات معالجة الصور المورفولوجية بما في ذلك التجزئة والتآكل والترشيح الخطي. تعتمد طريقة التنفيذ على نهج تصميم مزدوج جهار / برمجي (HW/SW) باستخدام أدوات التوليف (Quartus, Qsys, Sopc-Builder, ModelSim ....)المقدمة من طرف الشركة العالمية Intel-Altera حيث يتم استخدام منصة Altera-DE 2 –cyclone FPGA المرتبطة بطرف الحاسوب Nios-II لهذا العمل الذي يتم توصيله بجهاز كمبيوتر عبر ارتباط JTAG.حيث يتم إرسال الصورة الملتقطة أو المستلمة عبر الإرسال إلى FPGA للخضوع لمعالجة ثنائية اللون من خلال البحث عن عتبة بناءً على الرسم البياني. Histogramme وفي النهاية سيتم إرجاع النتيجة بعد ذلك لعرضها على واجهة VGA أو تخزينها في قاعدة بيانات ، في الوقت الفعلي عبر نظام الشبكة ليتم استغلالها لمهام أخرى . وتظهر النتائج التجريبية جدوى النهج المقترح ويمكن أن تمتد لتطبيقات أخرى.

Liste des abréviations

ADAS	Advanced Driver-Assistance Systems
Add-Acc	Addition accumulation
ARM	Advanced RISC Machines
ASIC	Application Specific Integrated Circuit
CAD	Computer Aided Design
CLB	Configurable Logic Bloc
CPLD	complex Programmable Logic Device
DCT	Direct cosine Transform
DDR	*Double Data Rate*
DIBCO	Document Image Binarization Competition
DMA	Direct Access Memory
DMIPS	Dhrystone MIPS
DSP	Digital signal Processor
DSP	Digital Signal Processor
EDA	Electronic Design Automation
EDK	Evaluation Development Kit
EEPROM	Electrical Erasable Programmable read only memory
FIFO	First In First Out
FPGA	Field Programmable Gate Array
FPLD	Field Programmable Device
FPU	Flatting point unit
FSL	Fast Simplex Link
GDSII	Graphic Database System Information Interchange
HDMI	High Digital Media Interface
I2C	Inter-Integrated Circuit
IC	Integrated Circuit
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IOB	Input Output Bloc

IOE	Input Output Element
IP	Intellectual Property
JTAG	Joint Test Action Group
LAB	Logic Array Bloc
LE	Logic Element
LMB	Local Memory Bus
LSI	Large Scale Integration
LUT	look Up Table
MAC	multiplication accumulation
MIPS	Million Instructions Par Seconde
MPGA	Masque Programmable Gate Array
MSI	meddle Scale Integration
OCR	Optical character recognition
PAL	Programmable Logic Array
PLA	Programmable Array Logic
PLD	Programmable Logic Device
PLL	Phase Lock Loop
RAM	Random Access Memory
RISC	Reduce Instruction Set Computer
ROI	Region Of interest
ROM	Read Only Memory
RTL	Register Transfer Level
SB	Switch Bloc
SD	Secure Data
SDRAM	Synchronous Dynamic Random Access Memory
SOC	System On Chip
SOPC	System On Programmable Chip
SPI	Serial Peripheral Interface
SPLD	Simple Programmable Logic Device
SRAM	Static Random Access Memory

TIE	Tensilica Instruction Extension
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
VGA VHDL	Video Graphical Adapter
VHDL	Very high level scale integration Hardware Description Language
VHLSI	Very High Level Scale Integration
XPRES	Xtensa Processor Extension Synthesis

Table des matières

Chapitre III

Aperçu sur les méthodes de binarisation, implémentations et comparaisons

Table des figures

Introduction

La binarisation d'images est l'une des étapes de prétraitement les plus pertinentes, conduisant à une diminution significative de la quantité d'informations soumises à une analyse approfondie. Une telle opération est applicable dans de nombreux systèmes ; qui utilisent principalement des méthodes de reconnaissance de formes, d'objets ou de caractères qui ne nécessitent pas l'analyse de la couleur ou de la texture. Quelques bons exemples pourraient être certaines applications robotiques, y compris les suiveurs de ligne et la navigation visuelle dans des couloirs, des labyrinthes, les systèmes avancés d'assistance à la conduite (ADAS) et les véhicules autonomes avec suivi de voie, ainsi que les méthodes largement utilisées de la reconnaissance optique de caractères (OCR).

L'analyse d'images binaires peut également être appliquée avec succès dans les systèmes embarqués, utilisant des technologies FPGA dans le cas de la nécessité de grande quantités de mémoire et une forte puissance de calcul rapide. Néanmoins, les résultats appropriés de l'analyse d'images binaires, en particulier la reconnaissance de texte, dépendent de la binarisation préalable correcte. Dans certaines applications, où l'éclairage uniforme de la scène peut être assuré, (cas de scanners à plat populaires [1] ou certains scanners de livres automatisés non destructifs, même avec des caméras infrarouges supplémentaires permettant un logiciel redressant les pages de livres numérisées [2]), le seuil global le plus simple peut être suffisant. Cependant, dans de nombreuses autres situations, l'éclairage peut être non uniforme, en particulier dans les images naturelles capturées par des caméras, des méthodes adaptatives plus sophistiquées doivent donc être appliquées [3].

L'un des problèmes les plus difficiles liés à l'influence du seuillage d'images sur une analyse ultérieure est la binarisation des images de documents et par conséquent les algorithmes nouvellement développés sont généralement validés en utilisant des images de documents intentionnellement préparées contenant diverses distorsions. Pour cette raison, des ensembles de données de compétitions de binarisation d'images de documents (Document Image Binarization Contest DIBCO) [4]) bien connus sont généralement utilisés pour vérifier l'utilité et valider les avantages des méthodes de binarisation.

Les méthodes Otsu [5] et leurs approches mises en œuvre sur FPGA (Field Programmable Gate Array) donnent des résultats satisfaisants en termes de vitesse et de consommation de ressources lorsqu'elles sont analysées individuellement [5,6]. Cependant, la mise en œuvre de calculs statistiques basés sur l'histogramme pour certains types d'images, peut fournir des résultats similaires à d'autres méthodes, donc en utilisant plus de ressources du FPGA. De plus, le processus d'obtention du seuil n'est qu'une partie de l'étape de binarisation, qui est l'une des étapes les plus simples d'un système de traitement d'images.

Les systèmes de traitement, par exemple, peuvent nécessiter une capacité de mémoire pour stocker une base de données utilisée dans l'interprétation des données dans des étapes de haut niveau, qui ont occupé une partie des calculs d'histogramme. Ainsi, il propose le développement d'un algorithme pour calculer le seuil en FPGA visant la simplicité mathématique, la réduction des ressources occupées et des valeurs proches de la méthode d'Otsu.

Dans cette thèse, deux problématiques seront résolues, la première consiste l'amélioration de performances liées à l'opération de binarisation d'images destinées aux vieux documents, l'approche proposée est testé avec succès sur des base de données utilisées dans des compétitions internationales. La seconde concerne l'aspect accélération de traitement par l'introduction et la production d'IP ''Intellectual Property'' à caractères matériel dans des plateformes FPGA. Trois IPs (histogramme-compute et ISODATA-compute et Binarisation-compute) décrits avec un langage de description du matériel ont été mis en évidences pour assurer un système accéléré pour le traitement.

Cette thèse est organisée en quatre chapitres :

Le premier chapitre couvre les technologies de conception des circuits intégrés numériques. La tendance vers l'utilisation des FPGA à exigé de présenter leurs architectures Principales, leurs caractéristiques leurs types et les moyens logistiques utilisé par le flux de conception.

Le second chapitre est dédié à la notion de système sur puce (System On Chip) et la méthodologie utilisée pour intégrer un contrôleur ou processeur Hardcore ou Softcore.

Le troisième chapitre a été consacré à la présentation détaillée de binarisation d'image dédiée aux vieux documents, puique l'étude faite, concerne un exemple de traitement d'image

La méthodologie et l'approche adoptée pour la conception d'un prototype de circuit dédié à la binarisation d'image sont décrits dans le chapitre quatre. En première partie, un système

d'amélioration de performance de binarisation est présenté. La réalisation d'un système de binarisation par seuillage d'Otsu sur puce SOC contrôlé par un soft-core NIOS-II est donnée en deuxième partie.

Une synthèse est donnée en conclusion.

Chapitre 1

Technologie de la logique programmable pour la conception de circuits

1.1 Introduction

Progressivement, les systèmes et les circuits électroniques sont conçus en se servant de technologies qui offrent des capacités de prototypage, de programmabilité et de réutilisation rapides (reprogrammabilité et ré-utilisation des composants) pour :

- Minimiser le temps de développement d'un produit système.
- permettre une reconfiguration en service (par la mise à niveau normale du produit
- améliorer les performances (c.à.d. fournir des capacités de débogage de conception et l'inévitable exigence de suppression des bogues de conception),
- ou même recycler les composants électroniques pour une autre application

Ces aspects sont requis par le temps de mise sur le marché réduit, la complexité accrue des applications, et le temps d'instrumentation et de test. Le problème se pose sur la manière d'y parvenir en utilisant la gamme de technologies de circuits électroniques disponibles aujourd'hui et les pistes ouvertes. L'objectif principal du développement de l'électronique avec les capacités ci-dessus a été dans le domaine numérique car les techniques de conception et la nature des signaux numériques sont bien adaptées à la reconfiguration.

1.2 Choix de la technologie de conception

Dans le domaine du digital, le choix de la technologie de mise en œuvre est d'utiliser soit:

- une logique numérique à fonctionnalité dédiée (fixe),
- un système à processeur programmé par logiciel (conçu sur la base d'un microprocesseur, microcontrôleur, ou processeur de traitement numérique de signal, DSP)
- un dispositif logique programmable configuré par matériel (Programmable Logic Device PLD), qu'il soit simple (SPLD), complexe (CPLD) ou constitué de réseau de portes programmable pré-diffusé sur terrain (Field Programmable Gate Array FPGA).
- une mémoire pour le stockage des données (sous forme de code de programme ou code d'implémentation circuits et systèmes numériques).

Sur la figure 1.1, les composants électroniques utilisés sont des circuits intégrés (Integred Circuits IC). Ce sont des circuits électroniques conditionnés dans un boîtier approprié qui contiennent des circuits complets allant de quelques dizaines de transistors (LSI ou MSI) à des centaines de millions de transistors (VHSI), la complexité du circuit dépendant de la fonctionnalité conçue.

Dans de nombreux circuits, la technologie sous-jacente sera basée sur les circuits intégrés. Un circuit électronique complet se composera d'un certain nombre de circuits intégrés, associés à d'autres composants tels que des résistances, bobines et des condensateurs.

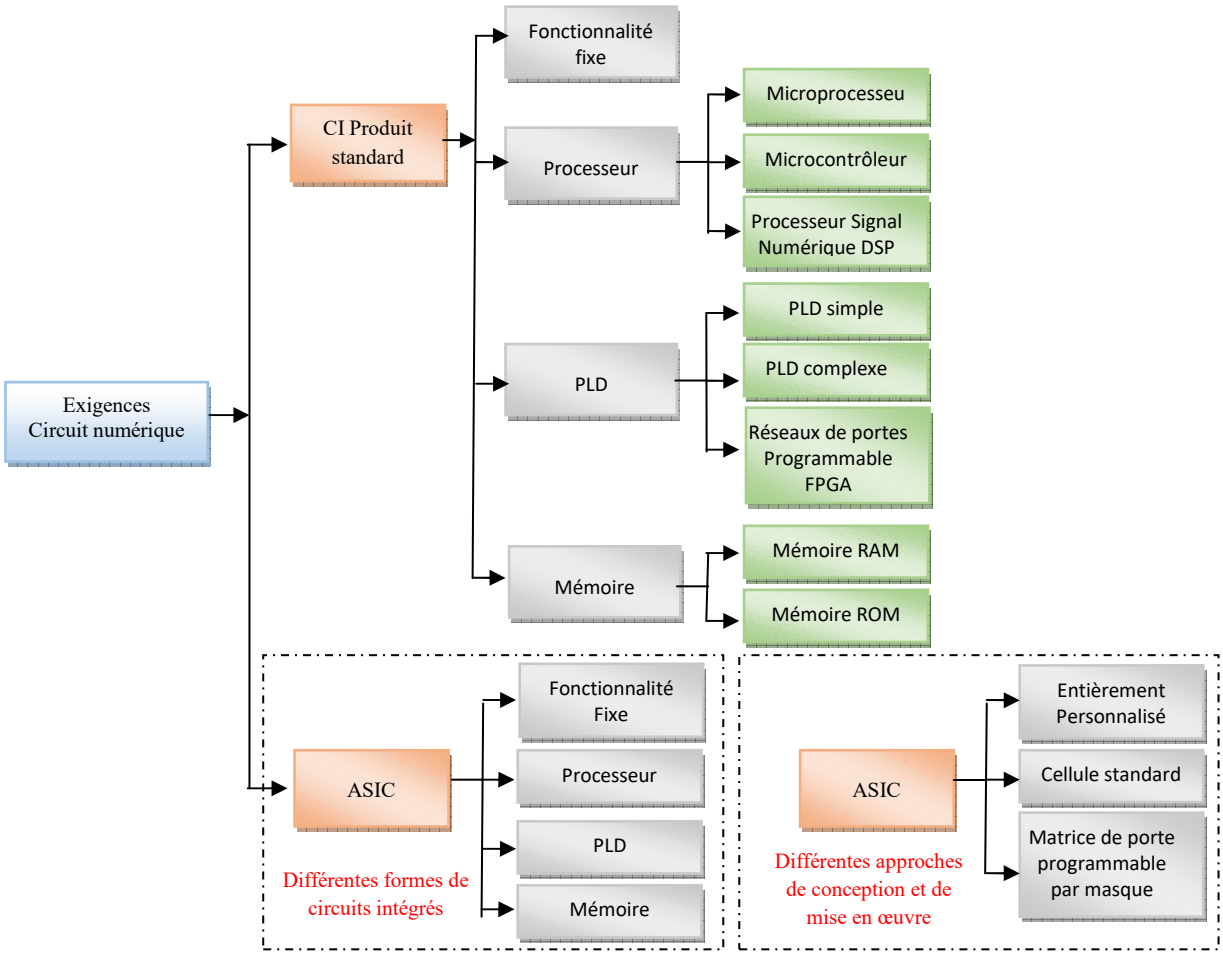


Figure 3.1.1. Technologies de réalisation des circuits numériques [7]

Cela s'applique au matériel et aux logiciels électroniques sous-jacents qui peuvent être utilisés pour concevoir un circuit pour une exigence donnée. Pour l'agencement identifié à la Figure 1.1, un ensemble donné d'exigences de circuits numériques est développé, et le rôle du concepteur est de proposer une solution qui réponde idéalement à toutes les exigences.

1.2.1 Exigences de conception

Les exigences typiques incluent [8]:

- **Contraintes de coûts :** Le processus de conception, le coût des composants, les coûts de fabrication et les coûts de maintenance et de développement futur doivent être dans des limites spécifiques.
- **Temps de conception :** La conception doit être générée dans un dans les délais.
- **Fourniture de composants :** le concepteur peut avoir les mains libres dans le choix des composants à utiliser, ou des restrictions peuvent être définies par l'entreprise ou les exigences de gestion de projet.
- **Expérience antérieure :** le concepteur peut avoir une expérience antérieure dans l'utilisation d'une technologie particulière, qui peut ou non être adaptée à la conception actuelle.
- **Formation :** Le concepteur peut avoir besoin d'une formation spécifique pour utiliser une technologie spécifique s'il n'a pas l'expérience préalable nécessaire.
- **Dispositions contractuelles :** si la conception doit être créée pour un client spécifique, le client fournira généralement un ensemble de contraintes qui seront définies dans le contrat de conception.
- **Contraintes de taille/volume :** la conception devrait être fabriquée pour s'adapter à une taille/un volume spécifique,
- **Contraintes de poids :** la conception doit respecter des restrictions de poids spécifiques (par exemple pour les applications portables telles que les téléphones portables),
- **Source d'alimentation :** le produit électronique serait soit fixe (dans un seul endroit permettant ainsi l'utilisation d'une source d'alimentation fixe) ou portable (pour être transporté à plusieurs endroits nécessitant une source d'alimentation portable (comme une batterie ou une cellule solaire),
- **Contraintes de consommation électrique :** la consommation électrique doit être faible que possible afin de minimiser les besoins en source d'alimentation, utilisable pendant une durée spécifique sur une source d'alimentation limitée, et compatible avec les meilleures pratiques dans le développement de produits électroniques environnemental.

1.2.2 Types de technologie de conception

Le choix initial pour la mise en œuvre du circuit numérique se situe entre un circuit intégré (Integred circuit IC) produit standard et un circuit intégré spécifique à l'application ‘’Application-Specific Integrated Circuit’’ ASIC [8] [18].

• **Circuit intégré produit standard IC**: est un composant électronique standard qui est conçu et fabriqué par une entreprise, disponible dans le marché auprès d'un fournisseur de composants, ou directement auprès du concepteur, pour un usage donné ou pour une plage donnée d'utilisation.

• **Circuit intégré spécifique à l'application ASIC** : Il s'agit d'un circuit intégré spécialement conçu pour une application. L’ASIC peut être conçu et fabriqué pour répondre aux exigences de conception, plutôt qu’utiliser un circuit intégré standard.

Pour de nombreuses applications, le développement d'un système électronique basé sur des circuits intégrés produit standard serait l'approche adoptée, puisque le temps et les coûts associés à la conception, à la fabrication et aux tests des ASICs peuvent être substantiels et hors du budget d'un projet de conception particulier. Entreprendre un projet de conception ASIC nécessite également l'accès à une expérience de conception de circuits intégrés et à des outils de conception assisté par ordinateur ‘’Computer Aided Design’’ CAD, en plus, un accès à une capacité de fabrication et de test appropriée.

L’approche de conception de IC de produit standard ou d'ASIC adoptée, envisage que le type du IC utilisé ou développé sera l'un des quatre types suivants :

- *Fonctionnalité fixe :* Ces circuits intégrés ont été conçus pour implémenter une fonctionnalité spécifique et ne peuvent pas être modifiés. Le concepteur utilise un ensemble de ces circuits intégrés pour mettre en œuvre une fonctionnalité de circuit globale donnée. Les modifications apportées au circuit nécessiteraient une refonte complète du circuit et l'utilisation de différents circuits intégrés à fonctionnalité fixe.

- *Processeur :* Le processeur serait plus familier à la majorité des gens, il est le plus couramment utilisé. Ce composant exécute un programme (application logicielle) pour implémenter la fonctionnalité requise. En changeant l’application logicielle, le processeur exécutera une fonction différente. Le choix du processeur dépendra du microprocesseur, du microcontrôleur ou du processeur de traitement de signal numérique ‘’Digital Signal Processor’’ (DSP).

- *Mémoire :* La mémoire sera utilisée pour stocker ou fournir un accès et permettre la modification des données et du code de programme à utiliser dans un circuit ou système

électronique à processeur. Les deux types de mémoire de base sont la mémoire morte ‘’Read Only Memory’’ ROM et la mémoire vive à accès aléatoire ‘’Random Acces Memory’’ RAM.

La ROM est utilisée pour conserver le code programme en l’absence de l'alimentation de la mémoire (stockage non volatil). Le code peut être soit fixe lors de la fabrication de la mémoire (masque ROM programmable) soit programmé électriquement une fois (PROM, ROM programmable) ou plusieurs fois. Selon le principe de la programmation et de l’effacement plusieurs versions sont à distinguer (EPROM, EEPROM et FlashROM).

Il faut noter que la PROM est aussi parfois considéré comme faisant partie de la même catégorie de circuit que la logique programmable.

La RAM est utilisée pour conserver les données et le code de programme qui nécessitent un accès rapide et la possibilité de modifier le contenu pendant le fonctionnement normal. La RAM diffère de la mémoire morte (ROM), elle peut être accédée en lecture ou en écriture dans l'application de circuit normale. Contrairement à la ROM, la RAM est considérée comme fournisseur de stockage volatile, le contenu de la RAM sera perdu lorsque l'alimentation est coupée. Il existe deux principaux types de RAM : la RAM statique (SRAM) et la RAM dynamique (DRAM).

- *Les Dispositifs logiques programmables ‘’Programmable Logic Devise’’ PLD* : Les dispositifs logiques programmables sont des circuits intégrés qui contiennent des cellules logiques numériques et une interconnexion programmable [9, 10]. L'idée de base de ces dispositifs est de permettre au concepteur de configurer les cellules logiques et de les interconnecter pour former un circuit électronique numérique au sein d'un seul puce. Dans ce cas, les ressources matérielles seront configurées pour implémenter une fonctionnalité requise. En modifiant la configuration matérielle, le PLD exploitera et réalisera une fonction différente.

Trois types de PLD sont disponibles :

- *Le dispositif logique programmable simple ‘’Simple Programmable Logic Device’’ SPLD :*

Les dispositifs logiques programmables simples (SPLD), tels que la logique de réseau programmable ‘’Programmable Array Logic’’ PAL et les réseaux logiques programmables ‘’Programmable Logique Array’’ PLA, sont utilisés depuis plus de quarante ans.

Tout d'abord, l'équation logique est minimisée et placée sous forme de somme de produits ‘’Somme of product’’ SOP, puis les produis sont configuré dans le réseau de ‘’ET logiques AND’’ et les sommes dans le réseau de ‘’OU logiques OR’’.

Les petits PLDs peuvent remplacer plusieurs anciens composants de type TTL à fonction fixe dans une conception. La plupart des PLDs contiennent une structure de type PLA dans laquelle une série de portes ANDs avec des entrées sélectionnables ou programmables alimentent une porte OR. Dans les PALs, la porte OR a un nombre fixe d'entrées et n'est pas programmable. Les portes ANDs et les portes ORs sont programmées pour implémenter directement une équation booléenne de somme de produits. Sur de nombreux PLDs, la sortie de la porte OR est connectée à une bascule dont la sortie peut ensuite être renvoyée en entrée dans le réseau de portes AND. Cela donne aux PLDs la capacité d'implémenter des machines à états simples. Un PLD peut contenir plusieurs de ces réseaux AND/OR figure 1.3.

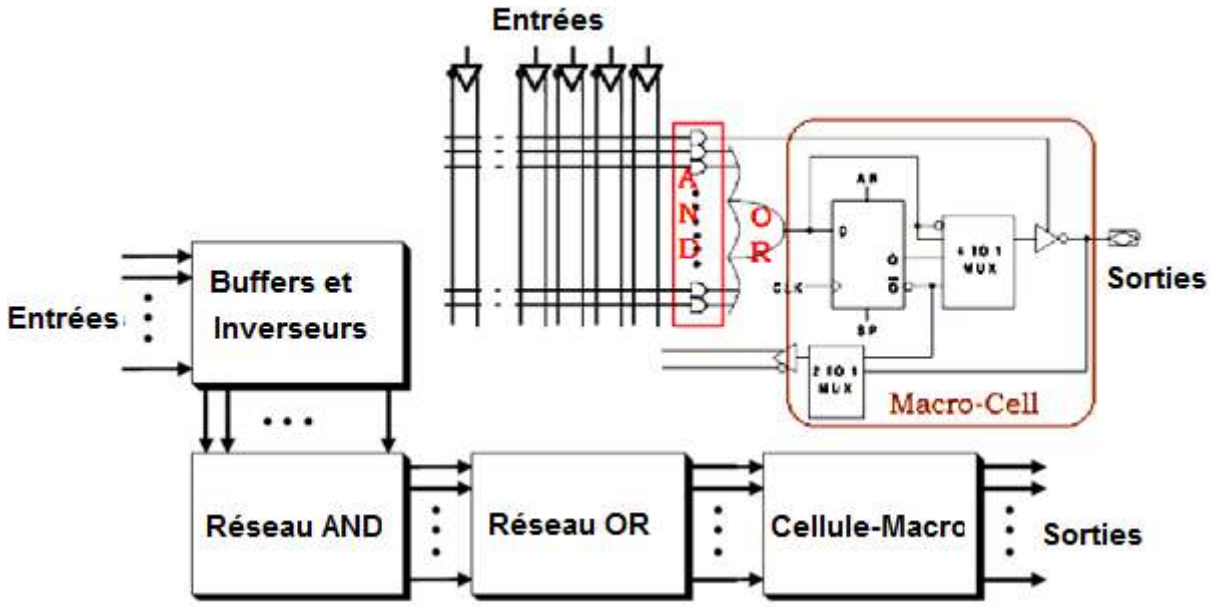


Figure 3.1.2 Structure d'un SPLD

Récemment, des densités plus élevées, une vitesse plus élevée et des avantages de coût ont permis l'utilisation de dispositifs logiques programmables dans une plus grande variété de conceptions. Les CPLDs et les FPGAs sont les dispositifs logiques programmables les plus denses et les plus avancés. Les conceptions utilisant un CPLD ou un FPGA nécessitent généralement quelques semaines d'effort d'ingénierie au lieu de plusieurs mois. Ces dispositifs sont aussi parfois appelés collectivement dispositifs logiques programmables sur site ou terrain ''Field Programmable Logic Device (FPLD)'' [7,8].

- *Le dispositif logique programmable complexe ''Complexe Programmable Logic Device'' :*

Le ''Complexe Programmable Logic Device CPLD'' présente plus de complexité par rapport au SPLD ; il s'appuie sur l'architecture SPLD et crée une conception beaucoup plus grande. Dans la

structure finale de la conception, le SPLD peut être utilisé pour intégrer les fonctions d'un certain nombre de circuits intégrés numériques discrets dans un seul composant et le CPLD peut être utilisé pour intégrer les fonctions d'un certain nombre de SPLD dans une seule puce. L'architecture CPLD est basée sur un petit nombre de blocs logiques et une interconnexion programmable globale. Une architecture CPLD générique est illustrée à la figure 1.5

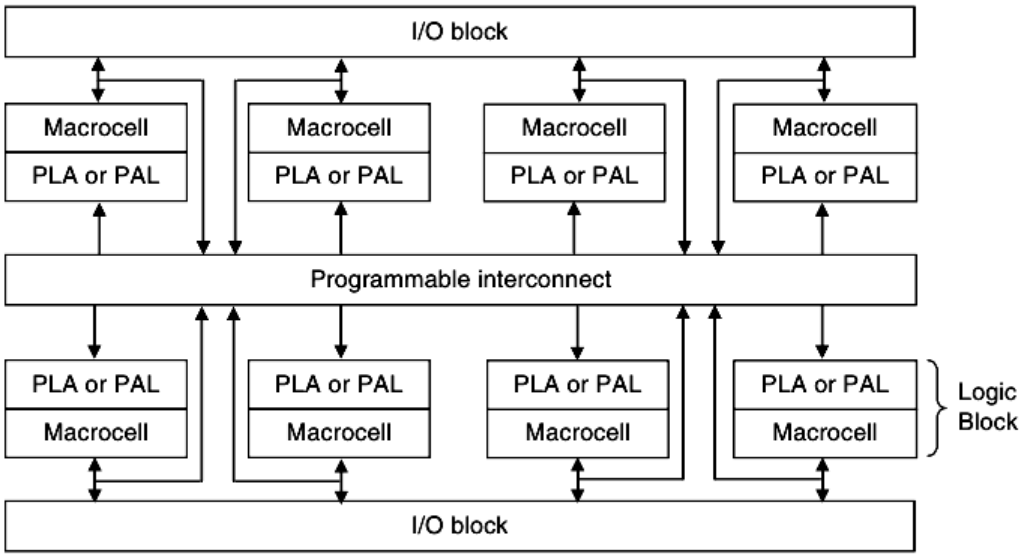


Figure 3.1.3 Architecture structurelle d'un CPLD

Le processeur et le PLD permettent au concepteur de mettre en œuvre et de modifier la fonctionnalité du circuit intégré en modifiant soit le programme logiciel, soit la configuration matérielle. [11]

- *Le réseau de portes programmables pré-diffusées sur site*, Field Programmable Gate Array (FPGA) : ce type de technologie sera vu en détail dans le paragraphe suivant.
- *Le Circuit intégré pour une application spécifique* ''Application-Specific Integrated Circuit'' **ASIC** : un ASIC peut être conçu pour créer l'une des quatre formes de circuit intégré produit standard (fonctionnalité fixe, processeur, mémoire ou PLD).

  Un ASIC serait conçu de la même manière qu'un circuit intégré de produit standard, de sorte que toute personne ayant accès à une installation de conception, de fabrication et de test ASIC peut créer un équivalent à un circuit intégré de produit standard (étant donné que les brevets et les problèmes juridiques généraux autour de la propriété intellectuelle (''Intellectual Property'' IP) les considérations relatives aux conceptions et dispositifs existants sont prises en compte.

De plus, un ASIC peut également incorporer une structure logique programmable aux côtés du matériel logique fixe. La figure 1.1 montre ce qui peut être fait avec la solution ASIC, ainsi que comment l'ASIC y parviendrait. Quatre formes différentes de IC (c'est-à-dire ce que fait le IC) qui peuvent être développées pour émuler un équivalent de IC de produit standard, et trois différentes approches de conception et de mise en œuvre. Dans une approche entièrement personnalisée, le concepteur contrôlerait tous les aspects de la conception et de la disposition des ASIC (la manière dont le circuit électronique est disposée sur la puce, qui est la pièce de matériau rectangulaire ou carré (généralement du silicium) sur laquelle les composants du circuit sont fabriqués). Cela donnerait les meilleures performances de circuit, mais serait long et coûteux à entreprendre. La conception entièrement personnalisée est principalement destinée aux circuits analogiques et à la création de bibliothèques de composants à utiliser dans une approche de conception de cellules standard semi-personnalisée. Une alternative à l'approche entièrement personnalisée utilise une approche semi-personnalisée. Ceci est subdivisé en une approche de cellule standard ou une approche de réseau de portes programmable à masque ‘’Masque Programmable Gate Array’’ MPGA. L'approche de cellule standard utilise une bibliothèque de composants de circuit de base préconçus (généralement des cellules logiques numériques) qui sont connectés dans le circuit intégré pour former le circuit global. Dans une vue simpliste, cela reviendrait à créer une conception en connectant ensemble des circuits intégrés à fonctionnalité fixe, mais au lieu d'utiliser plusieurs circuits intégrés, un seul circuit intégré est créé. Cette approche est plus rapide et moins coûteuse qu'une approche entièrement personnalisée, mais ne fournirait pas nécessairement les meilleures performances de circuit. Étant donné que seuls les circuits requis dans la conception seraient fabriqués (fabriqués), il y aurait un compromis immédiat entre les performances du circuit, le temps de conception et le coût de conception (un compromis rencontré quotidiennement par le concepteur).

L'approche MPGA est similaire à une approche de cellule standard en ce sens qu'une bibliothèque de composants est disponible et connectée, mais la disposition sur la puce (silicium) est différente. Un réseau de portes logiques est prédéterminé et le circuit est créé en créant des pistes d'interconnexion métalliques entre les portes logiques. Dans l'approche MPGA, toutes les portes logiques fabriquées sur la puce ne seraient pas nécessairement utilisées. Cela utiliserait une matrice plus grande que dans une approche de cellule standard, avec l'inclusion de portes inutilisées, mais il a l'avantage d'être plus rapide à fabriquer qu'une approche de cellule standard.

1.3 Notions sur le réseau de portes pré-diffusées sur site FPGA

Un réseau de portes pré-diffusées sur site ‘’Field Programmable Gata Array’’[11] FPGA typique est un microcircuit constitué d'un réseau d'exactement de mêmes cellules avec des connexions programmables. L'utilisateur peut configurer les fonctions à exécuter par chacune des cellules logiques et les connexions entre elles.

Les premiers FPGA ont été inventés par Ross Freeman (cofondateur de Xilinx) en 1985 et depuis lors, leur capacité logique s'est considérablement améliorée. Ces types de circuits sont devenus un choix populaire car les systèmes FPGA peuvent être reprogrammés après la fabrication pour mettre en œuvre l'application finale souhaitée par l'utilisateur. Certains FPGAs peuvent être reprogrammés à l'infini et d'autres pour une durée limitée.

En termes généraux, les FPGAs sont des puces de silicium programmables avec une collection de blocs logiques programmables entourés de blocs d'entrée/sortie qui sont assemblés via des ressources d'interconnexion programmables pour assurer le fonctionnement de n’importe quel type de circuit ou système numérique.
Les FPGAs sont développés à partir de mémoires mortes programmables (PROM) et de dispositifs logiques programmables (PLD).
Contrairement aux processeurs, les FPGAs sont de nature véritablement parallèle. Chaque tâche de traitement indépendante est affectée à une section dédiée de la puce. Par conséquent, les performances d'une partie de l'application ne sont pas affectées lorsque des tâches de traitement supplémentaires sont ajoutées.

1.3.1 Les FPGA : Architectures Principales, Caractéristiques et Types

Une architecture précise d'un FPGA varie d'un fabricant à l'autre. Une structure FPGA générique contient les éléments suivants :

• **Blocs logiques programmables ‘’Configurable Logic Bloc (CLB)’’ ou Eléments logiques ‘’Logic Element (LE)’’**: Les blocs logiques peuvent être formés de milliers de transistors à des millions de transistors. Ils implémentent les fonctions logiques requises par la conception et se composent de composants logiques variés selon une architecture petits grains ou gros grains tels que des paires de transistors, des tables de correspondance (LUT) et une logique de transport et de contrôle (bascules et multiplexeurs) et latchs.

Les FPGAs à petits grains ressemblent aux réseaux de portes ASIC, dans le sens que les CLBs ne contiennent que de petits éléments très basiques tels que des portes NANDs, des portes NORs, etc. La philosophie est que de petits éléments peuvent être connectés pour créer des fonctions plus importantes sans gaspiller trop de logique. Dans un FPGA à gros grain, où le CLB peut contenir deux ou plusieurs bascules, une conception qui n'a pas besoin de beaucoup de bascules laissera beaucoup d'entre elles inutilisées. Malheureusement, les architectures à petit grain nécessitent beaucoup plus de ressources de routage, qui prennent de la place et insèrent un retard en plus, mais améliore la souplesse de conception.

Dans une architecture à gros grain, Les CLBs contiendront suffisamment de logique pour créer une petite machine à états. Dans une architecture à petit grain, plus proche d'un véritable ASIC à matrice de portes, le CLB ne contiendra qu'une logique très basique. Le diagramme de la figure 1.4 serait considéré comme un gros bloc de grains. Il contient de la RAM pour créer des fonctions logiques combinatoires arbitraires. Il contient également des bascules pour les éléments de stockage cadencés et des multiplexeurs afin d'acheminer la logique à l'intérieur du bloc et vers et depuis des ressources externes. Les multiplexeurs permettent également la sélection de polarité et la réinitialisation et la sélection d'entrée claire. [11]

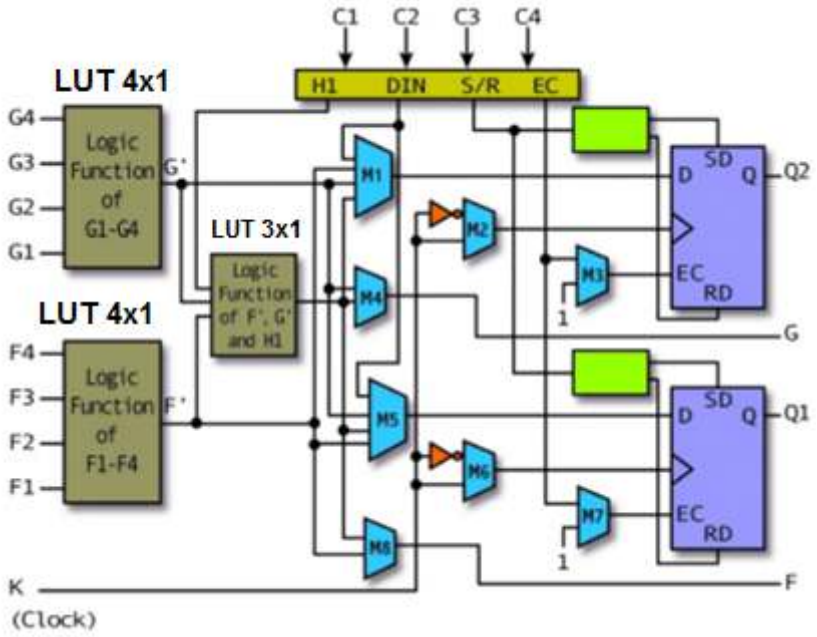


Figure 3.1.4 Bloc logique configurable simplifié

• **Blocs d'Entrées/Sorties programmables ‘’Input Output Bloc IOB’’:** Ils connectent des blocs logiques avec des composants externes via des broches d'interfaçage.

Un bloc d'E/S configurable, illustré à la figure 1.5, est utilisé pour amener des signaux sur la puce et les renvoyer à nouveau. Il se compose d'un tampon d'entrée et d'un tampon de sortie avec trois contrôles d'état et de sortie à collecteur ouvert. En général, il y a des résistances pull up sur les sorties et parfois des résistances pull down.

La polarité de la sortie peut généralement être programmée pour une sortie active haute ou basse active et souvent la vitesse de balayage de la sortie peut être programmée pour des temps de montée et de descente rapides ou lents. De plus, il y a souvent une bascule sur les sorties afin que les signaux cadencés puissent être émis directement vers les broches sans rencontrer de retard important. Cela est fait pour les entrées afin qu'il n'y ait pas beaucoup de retard sur un signal avant d'atteindre une bascule, ce qui augmenterait le temps de maintien du composant.

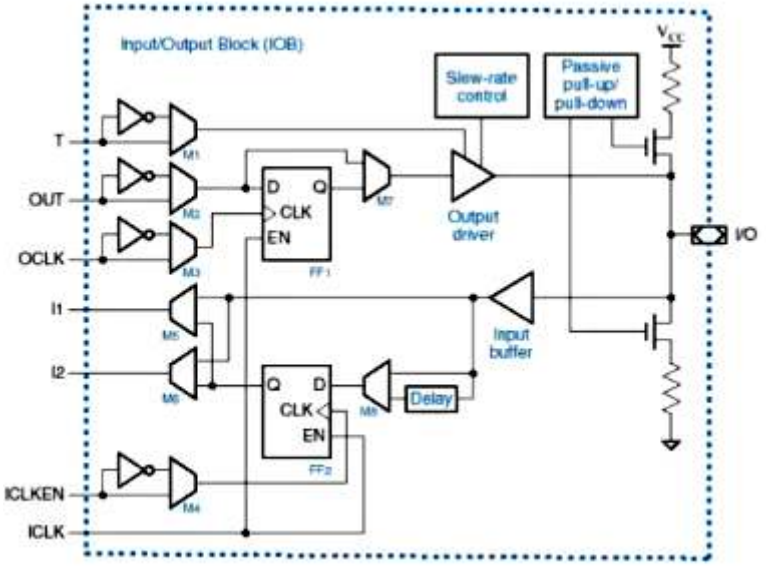


Figure 3.1.5 Structure d'un bloc entrée sortie d'un FPGA

• **Ressources d'interconnexion programmables** ''Switch Bloc (SB)'': Ce sont des interconnexions programmables électriquement (pré-posées verticalement et horizontalement) qui fournissent le chemin de routage pour les blocs logiques programmables. Les chemins de routage contiennent des segments de fil de différentes longueurs qui peuvent être interconnectés via des commutateurs programmables électriquement. La densité FPGA dépend du nombre de segments utilisés pour les chemins de routage.[12]

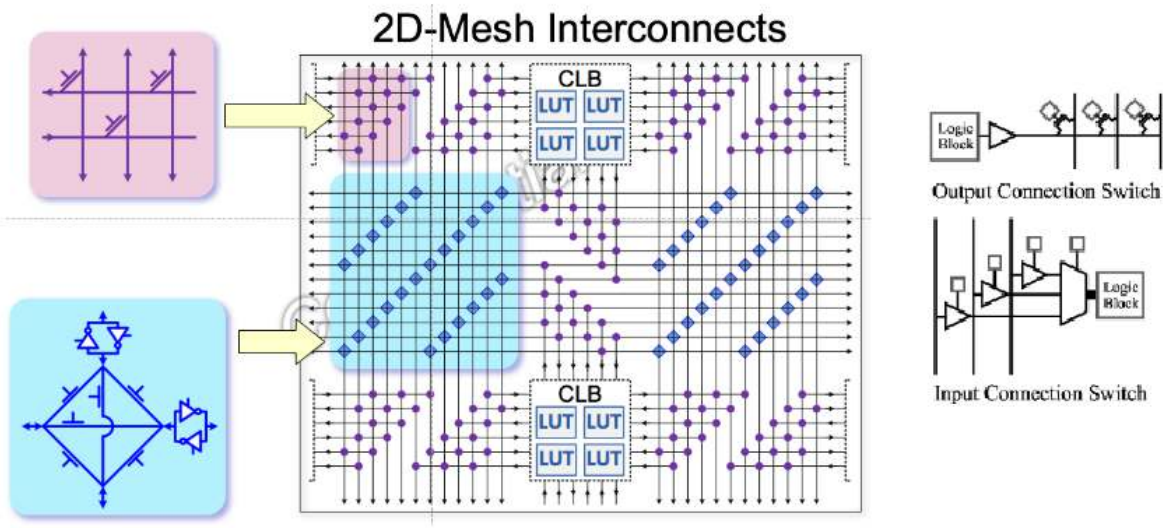


Figure 3.1.6 Interconnexion programmable FPGA

• Bloc à usage spécifique

En plus des trois blocs, les FPGAs possèdent souvent ce que l'on appelle des ressources additionnelles. La figure 1.7 illustre la disposition de ces blocs additionnels sur un FPGA. Le fonctionnement et le placement de ces blocs diffèrent d'une référence de FPGA à une autre.

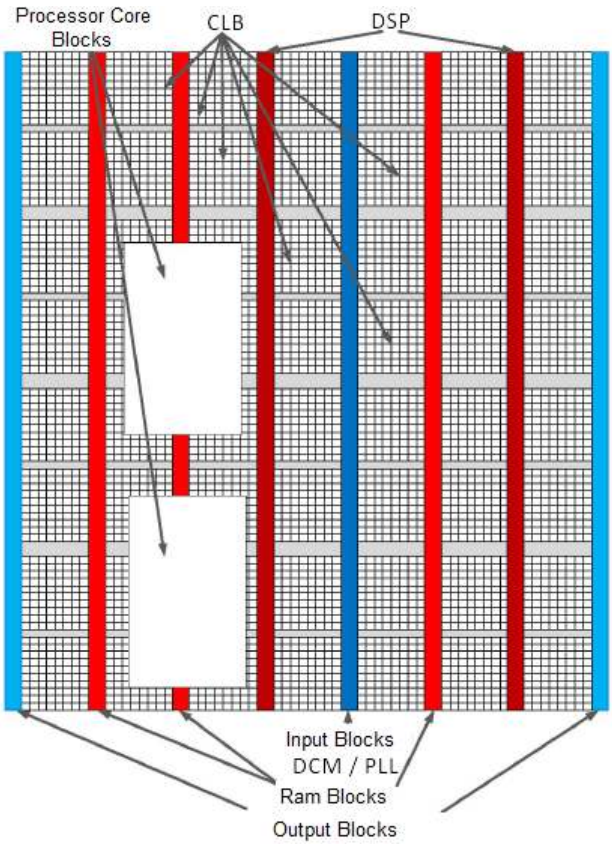


Figure 3.1.7 Structure d'un FPGA avec blocs à usage spécifiques

- **Les Cores Processurs Traitement de Signal ''Digital Signal Processor (DSP)''**

Les processeurs de traitement de signal ''Digital Signal Processor (DSP)'' sont des blocs qui permettent des conceptions plus complexes, qui peuvent consister soit en traitement numérique du signal ou seulement certains assortiments de multiplication, addition et soustraction. Comme pour les Block-RAM BRAM, il est possible de mettre en œuvre ces blocs grâce au CLB, mais il est plus efficace en termes de performances et de consommation d'énergie, d'intégrer plusieurs de ces composants au sein du FPGA. Un bloc DSP permet de réaliser un multiplicateur, un accumulateur, un additionneur, et des opérations logiques (AND, OR, NOT, et NAND) sur un bit. Il est possible de combiner les blocs DSP pour effectuer des opérations plus importantes, telles que l'addition avec virgule flottante simple, la soustraction, la multiplication, la division, et la racine carrée. Le nombre de blocs DSP est dépendant du dispositif.

- **Les processeurs embarqués**

Les processeurs embarqués ''Embedded Processor'' sont l'un des ajouts les plus importants pour le FPGA. Beaucoup de conceptions nécessitent l'utilisation d'un processeur embarqué.

Souvent, le choix d'un dispositif de FPGA avec un processeur embarqué (comme le Virtex5 de Xilinx) permet de simplifier grandement le processus de conception tout en réduisant l'utilisation des ressources et la consommation d'énergie. Le PowerPC IBM 405 et 440 sont des exemples de deux processeurs inclus dans le Virtex4 et 5 de Xilinx. Ce sont des processeurs RISC classiques qui mettent en œuvre un jeu d'instructions PowerPC.

- **Circuit d'horloge**

Des blocs d'E/S spéciaux avec des tampons d'horloge de lecteur élevés spéciaux, appelés pilotes d'horloge, sont répartis autour de la puce. Ces tampons sont connectés à des plots d'entrée d'horloge et pilotent les signaux d'horloge sur les lignes d'horloge globale décrites ci-dessus. Ces lignes d'horloge sont conçues pour des temps de skew ''timing skew'' faibles et des temps de propagation rapides. Comme nous le verrons plus tard, la conception synchrone est indispensable avec les FPGA, car l'asymétrie et les retards absolus ne peuvent pas être garantis. Ce n'est qu'en utilisant des signaux d'horloge provenant de tampons d'horloge que les retards relatifs et les temps de décalage peuvent être garantis. [13]

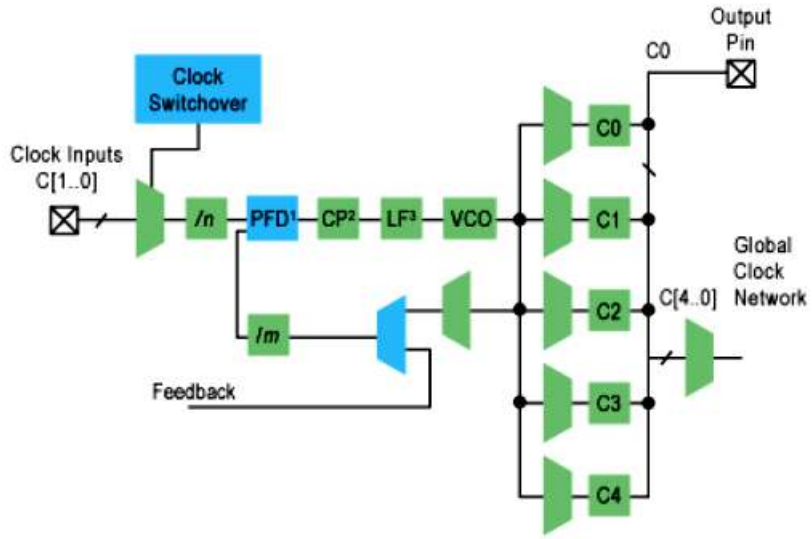


Figure 3.1.8 Circuit d'horloge

1.3.2 Principales caractéristiques et types de FPGA

L'architecture de routage affecte la densité et les performances du FPGA. Sur la base de la disposition interne des blocs, les FPGA peuvent être classés en trois catégories : [11]

• **Matrices symétriques :** Cette disposition se compose de blocs logiques disposés en lignes et colonnes d'une matrice et interconnectant des ressources entre elles. Cette matrice symétrique est entourée de blocs d'E/S qui la relient au monde extérieur Figure 1.9.

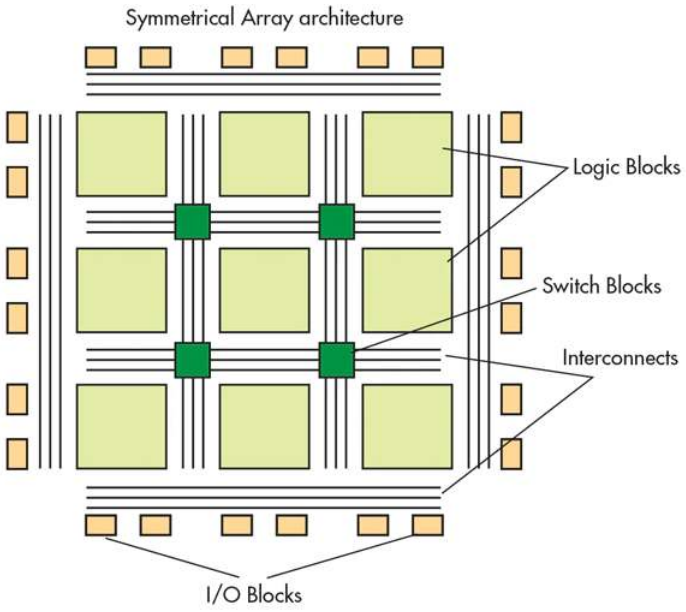


Figure 3.1.9 Structure à matrices symétriques

Les réseaux symétriques consistent en un réseau bidimensionnel de modules logiques interconnectés par des ressources d'interconnexion programmables verticales et horizontales.

• Architecture ligne : Elle alterne des lignes de ressources d'interconnexion programmables avec des lignes de blocs logiques tandis que les blocs Entrées/Sorties sont situés en périphérie des lignes figure 1.10. Une rangée peut être connectée à des rangées adjacentes via une interconnexion verticale. [12]

L'architecture basée sur les rangées se compose de rangées de blocs logiques séparés par des ressources d'interconnexion programmables.

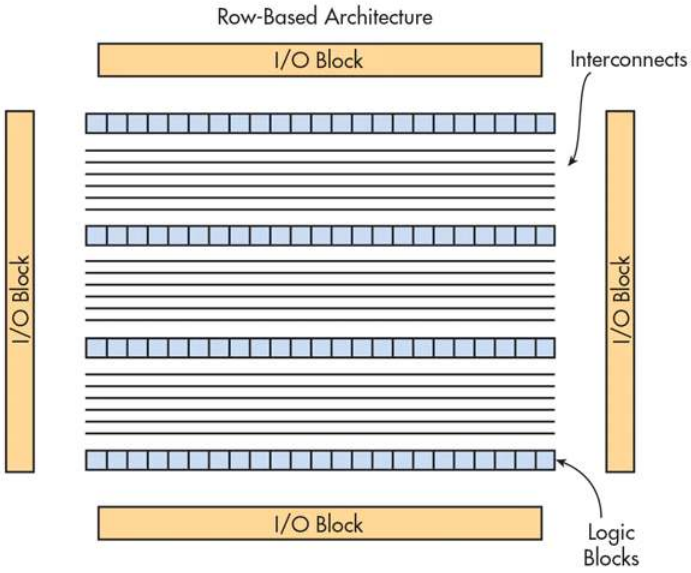


Figure 3.1.10 Structure en ligne

• PLD hiérarchiques : ils sont conçus de manière hiérarchique, le niveau supérieur ne contenant que des blocs logiques et des interconnexions. Chaque bloc logique contient un certain nombre de modules logiques. Et chaque module logique a des éléments fonctionnels combinatoires ainsi que séquentiels figure 1.11. [12]

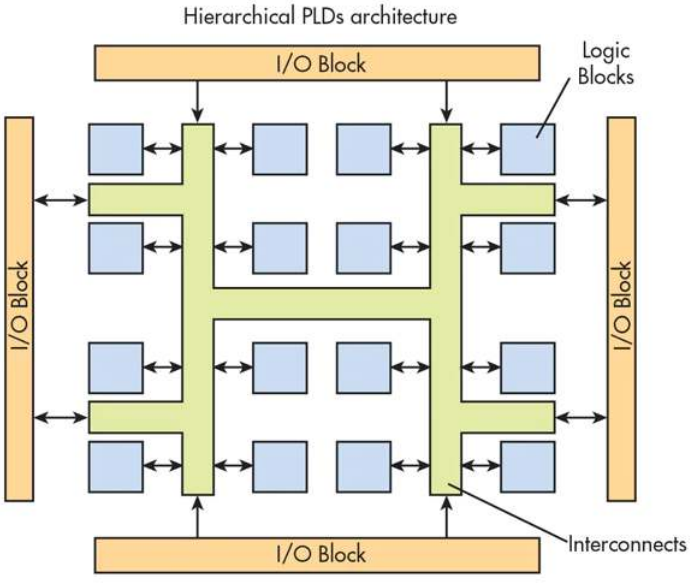


Figure 3.1.11 PLD hiérarchique

Dans un PLD hiérarchique, chaque bloc logique contient un nombre de modules logiques.

En fonction du type de technologie de programmation, les FPGAs peuvent être classés en trois catégories :

• *FPGA basés sur SRAM :* les cellules RAM statiques contrôlent le transistor de passage, les portes de transmission ou les multiplexeurs. Ils peuvent être reprogrammés au fur et à mesure de l'évolution de la conception, mais lorsque l'alimentation est coupée, la programmation est perdue et ils doivent être configurés au démarrage. Par conséquent, ils ont besoin d'une mémoire externe pour stocker le programme.

• *FPGA anti-fusibles :* Ils utilisent une technologie CMOS anti-fusible et une fois le FPGA programmé, il ne peut pas être reprogrammé. Ils conservent leur programme lorsque l'alimentation est coupée.

• *FPGA Flash :* ils utilisent des cellules à grille flottante comme commutateurs qui améliorent l'efficacité de la zone. Ils ne perdent pas d'informations lorsque le système est éteint. Cette technologie n'a pas besoin d'une mémoire externe pour stocker le programme, mais ils ne peuvent pas être reprogrammés à l'infini en raison de l'accumulation de charge dans l'oxyde.

1.3.3 Exemples de FPGA et constructeurs FPGA

1.3.3.1 Altera Cyclone FPGA Architecture [17]

La famille FPGA Cyclone est basée sur un processus SRAM en cuivre toutes couches de 1,5 V, 0,13 m, avec des densités allant jusqu'à 20 060 éléments logiques (LE) et jusqu'à 288 Kbits de RAM. Avec des fonctionnalités telles que les boucles à verrouillage de phase (PLL) pour la synchronisation et une interface dédiée à double débit de données (DDR) pour répondre aux exigences de mémoire DDR SDRAM et RAM à cycle rapide (FCRAM), les appareils Cyclone sont une solution économique pour les applications de chemin de données. Les appareils Cyclone prennent en charge diverses normes d'E/S, y compris LVDS à des débits de données allant jusqu'à 640 mégabits par seconde (Mbps) et 66 et 33 MHz, 64 et 32 bits d'interconnexion de composants périphériques (PCI), pour l'interfaçage et la prise en charge Dispositifs ASSP et ASIC. Altera propose également de nouveaux périphériques de configuration série à faible coût pour configurer les périphériques Cyclone. [16] [19]

Le dispositif Cyclone est configuré en chargeant une mémoire interne statique à accès aléatoire (SRAM). Étant donné que la SRAM est utilisée dans les FPGA, la configuration sera perdue chaque fois que l'alimentation est coupée. Dans les systèmes réels, une petite mémoire flash série externe à faible coût ou une mémoire morte programmable (PROM) est normalement utilisée pour charger automatiquement les informations de programmation du FPGA lors de la mise sous

tension de l'appareil. Les FPGA contiennent une architecture bidimensionnelle basée sur des lignes et des colonnes pour implémenter le logique utilisateur. Un réseau d'interconnexion de colonnes et de rangées fournit des connexions de signaux entre les blocs de matrice logique Logic Array Bloc (LAB) et les blocs de mémoire intégrés. Les délais d'interconnexion sont du même ordre de grandeur que les délais logiques.

La matrice logique de la Cyclone FPGA [17] se compose de LAB, avec 10 éléments logiques (Logic Element LE) dans chaque LAB. Un Logic Element est une petite unité de logique fournissant une mise en œuvre efficace des fonctions logiques de l'utilisateur. Les LAB sont regroupés en lignes et en colonnes sur l'ensemble de l'appareil. Les appareils Cyclone vont de 2 910 à 20 060 LE. Les blocs de mémoire embarqués M4K RAM sont des blocs de mémoire à double port avec 4K bits de mémoire plus la parité (4 608 bits). Ces blocs fournissent une mémoire double ou simple port de 1 à 36 bits de large jusqu'à 200 MHz. Ces blocs sont regroupés en colonnes sur l'appareil entre certains LAB. Les Cyclone EP1C6 et EP1C12 contiennent respectivement 92K et 239K bits de RAM embarquée. Chacune des broches d'E/S de l'appareil Cyclone est alimentée par un élément d'E/S (IOE) situé aux extrémités des lignes et des colonnes LAB autour de la périphérie de l'appareil. Les broches d'E/S prennent en charge diverses normes d'E/S asymétriques et différentielles. Chaque IOE contient un tampon d'E/S bidirectionnel et trois registres pour enregistrer les signaux d'entrée, de sortie et d'activation de sortie. Les dispositifs Cyclone fournissent également un réseau d'horloge mondial à faible décalage et jusqu'à deux boucles à verrouillage de phase (PLL). Le réseau d'horloge mondial se compose de huit lignes d'horloge mondiales qui traversent l'ensemble de l'appareil. Le réseau d'horloge global peut fournir des horloges pour toutes les ressources au sein de l'appareil, telles que les IOE, les LE et les blocs de mémoire. Les PLL Cyclone [17] fournissent une horloge à usage général avec multiplication/division d'horloge et déphasage ainsi que des sorties externes pour la prise en charge d'E/S différentielles à grande vitesse.

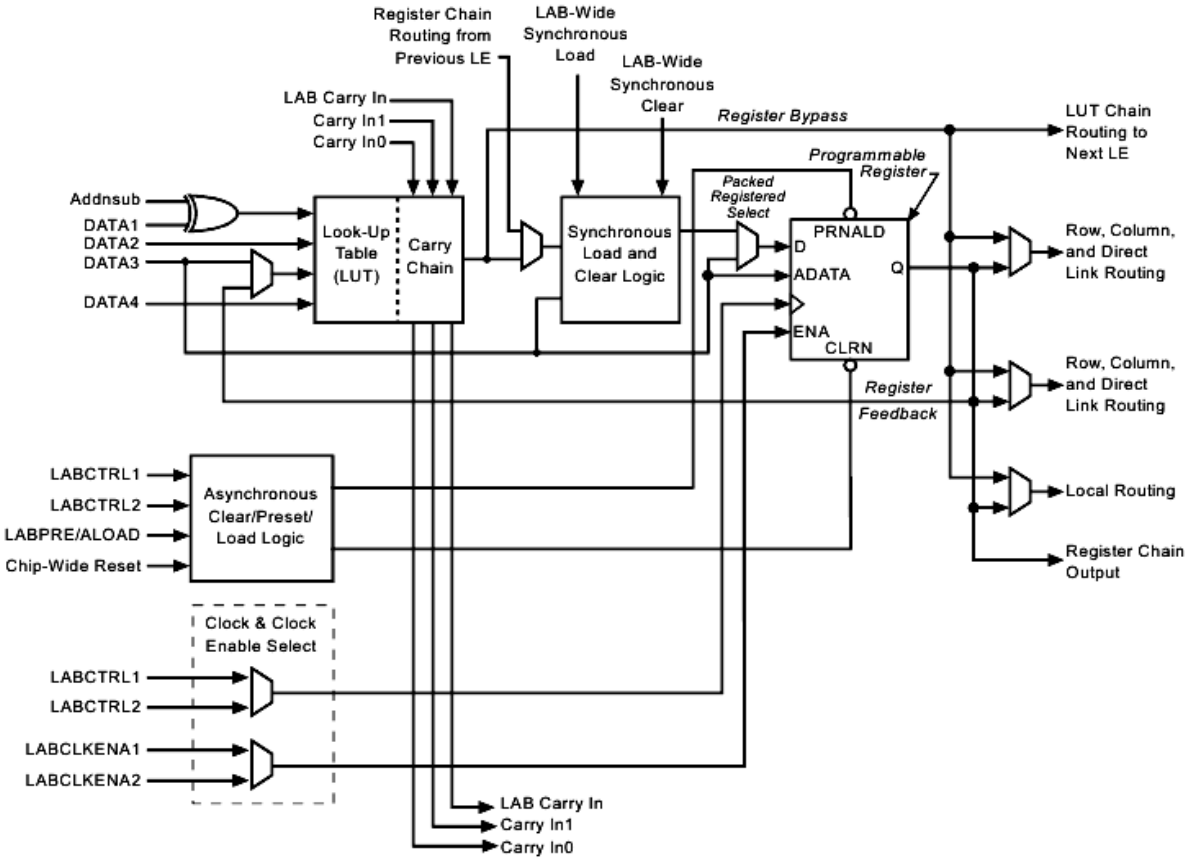


Figure 3.1.12 Structure d'un Element Logique ''LE'' de la cyclone d'Intel Altera [17]

La Figure 1.12 montre un élément logique Cyclone. Les portes logiques sont mises en œuvre à l'aide d'une table de correspondance (LUT), qui est une SRAM 16x1 haute vitesse. Quatre entrées sont utilisées pour adresser la mémoire de la LUT. La table de vérité pour le réseau de portes souhaité est chargée dans la SRAM de la LUT pendant la programmation. Une seule LUT peut donc modéliser n'importe quel réseau de portes avec quatre entrées et une sortie. Les multiplexeurs illustrés à la Figure 1.12 sont tous contrôlés par des bits dans la mémoire de configuration SRAM du FPGA.

1.3.3.2 Architecture Xilinx 4000

La famille Xilinx 4000 était une famille de dispositifs FPGA de première génération populaire avec 2000 à 180 000 portes utilisables. Il est configuré par programmation de la SRAM interne. La figure 1.13 représente l'allure d'une tranche de silicium de six pouces contenant plusieurs puces FPGA XC4010E à 10 000 grilles. [14-15]

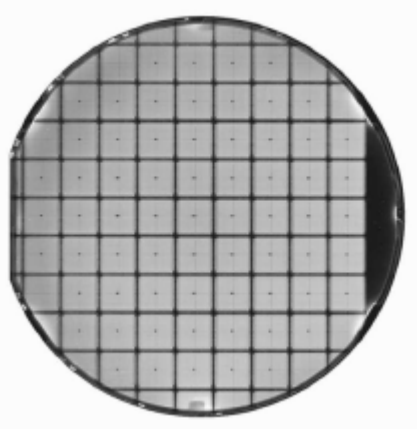


Figure 3.1.13 Représentation de plusieurs puces FPGA XC4010E sur une Tranche silicium

La figure 1.14 est une vue à contraste amélioré d'une seule puce XC4010E. Il s'agit d'une matrice 20 par 20 d'éléments logiques et les lignes d'interconnexion environnantes. Les matrices qui réussissent l'inspection et les tests au niveau de la tranche sont découpées dans la tranche et emballées dans une puce. Les rendements FPGA sont généralement de 90 % ou plus après les premiers cycles de production.

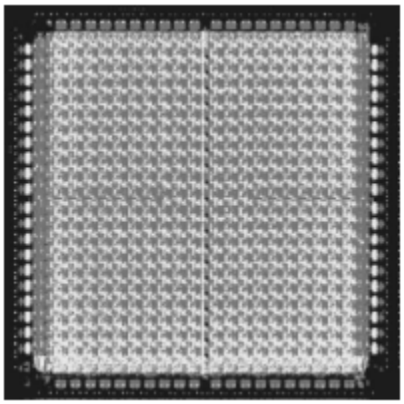


Figure 3.1.14 Allure d'une puce FPGA XC4010E

Comme le montre la figure 1.15, ce dispositif contient un élément logique plus complexe appelé bloc logique configurable (CLB). Chaque CLB contient trois tables de recherche basées sur SRAM. Les sorties des LUT peuvent être transmises à deux bascules et acheminées vers d'autres CLB. Les tables de correspondance d'un CLB peuvent également être configurées pour agir comme une RAM 16 x 2 ou une RAM 16 x 1 double port. Une logique de transport à grande vitesse est fournie entre les CLB adjacents. [13-14,15]

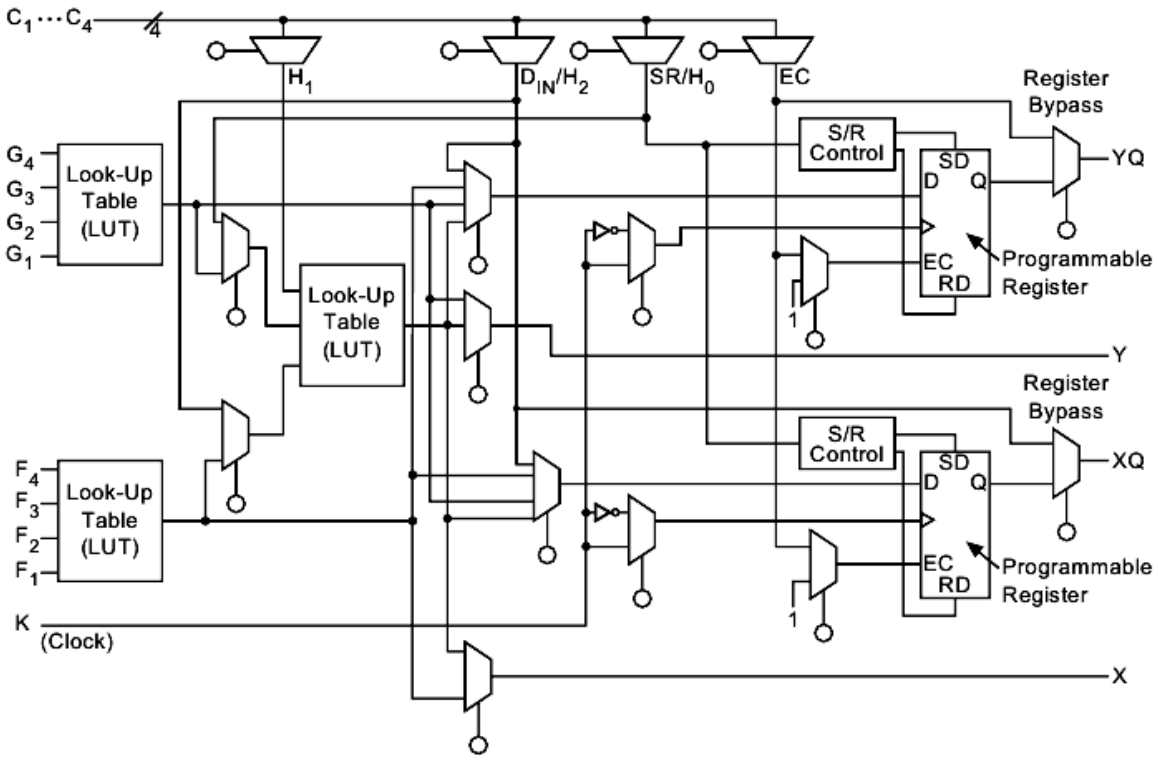


Figure 3.1.15 Structure d'un Bloc Logique configurable CLB de Xilinx [15]

1.3.3.3 Outils de conception CAO pour la logique programmable

Avec des capacités logiques d'une puce FPGA individuelle approchant les 10 000 000 de portes, la conception manuelle au niveau de la porte n'est plus une option envisageable dans les systèmes complexes. Le prototypage rapide à l'aide de langages de description matérielle (HDL), de cœurs IP et d'outils de synthèse logique a pratiquement remplacé la conception traditionnelle au niveau de la porte par une entrée de capture schématique. Ces nouveaux outils de synthèse logique basés sur les langages de description matérielle (Hardware Description Language HDL) peuvent être utilisés pour les conceptions ASIC et FPGA. Les plus utilisés sont VHDL et Verilog System-C et Verilog-C.

Le flux de conception d'outils de CAO FPGA typique est illustré à la figure 1.16. Après la saisie de la conception à l'aide d'un HDL ou d'un schéma, la conception est automatiquement traduite, optimisée, synthétisée et enregistrée sous forme de netlist. (Une netlist est une représentation textuelle d'un diagramme logique.) Une étape de simulation fonctionnelle est souvent ajoutée avant l'étape de synthèse pour accélérer les simulations de grandes conceptions. Un outil automatique adapte ensuite la conception à l'appareil en convertissant la conception pour utiliser

les éléments logiques du FPGA, d'abord en plaçant la conception dans des emplacements d'éléments logiques spécifiques dans le FPGA, puis en sélectionnant les chemins de routage du réseau d'interconnexion. Le processus de lieu et d'itinéraire peut être assez complexe et peut prendre plusieurs minutes à calculer sur des conceptions volumineuses. Sur les gros appareils, l'explosion combinatoire (croissance exponentielle) empêchera l'outil d'examiner toutes les combinaisons possibles de lieux et d'itinéraires. Lorsque les conceptions nécessitent une synchronisation critique, certains outils prennent en charge des contraintes de synchronisation qui peuvent être placées sur des lignes de signaux critiques. Ces contraintes facultatives sont ajoutées pour aider l'outil de placement et de routage à trouver un placement de conception avec des performances améliorées.

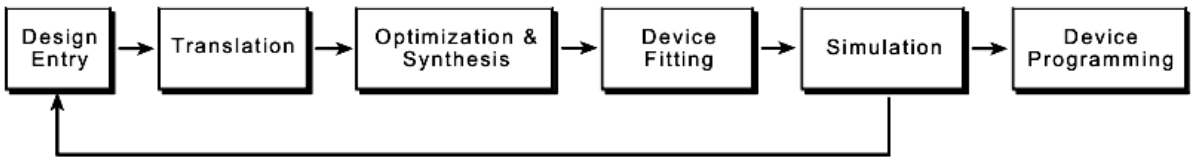


Figure.3.1.16 Flux de conception d'outils de CAO pour FPGA

Après l'emplacement et l'itinéraire, la simulation peut être effectuée en utilisant des délais réels de porte et d'interconnexion à partir d'un modèle de synchronisation détaillé de l'appareil. Bien que des erreurs puissent se produire à n'importe quelle étape, le chemin le plus courant consiste à trouver des erreurs lors d'une simulation exhaustive. La dernière étape est la programmation du périphérique et la vérification du matériel sur le FPGA.

Quelques outils de synthèse HDL prennent désormais en charge la synthèse comportementale. Contrairement aux modèles de niveau de transfert de registre ‘’Register Transfert Level (RTL), les modèles de synthèse comportementale ne spécifient pas les états et la séquence exacts des transferts de registre. Un fichier de contraintes séparé spécifie le nombre d'horloges nécessaires pour obtenir les signaux sélectionnés et l'outil génère automatiquement les automates, la logique et les transferts de registre nécessaires. Bien qu'ils ne soient pas actuellement largement utilisés pour les conceptions actuelles, de nouveaux outils de CAO FPGA apparaissent également, basés sur d'autres langages tels que C et Java. Certains de ces outils au niveau du système génèrent des modèles VHDL ou Verilog en tant qu'étape intermédiaire. Les nouveaux HDL tels que

SystemVerilog (www.systemverilog.org) et SystemC (www.systemC.org) offrent une prise en charge améliorée de la vérification.

Des outils qui génèrent automatiquement une conception FPGA à partir d'autres outils d'ingénierie tels que MATLAB-Simulink ou LabVIEW ont également été introduits. Ces outils graphiques sont principalement destinés au développement d'applications DSP pour les FPGA à l'aide d'une bibliothèque de blocs DSP spécialisés.

1.4 Conclusion

Les dispositifs logiques programmables complexes (CPLD) et les matrices ou réseaux de portes programmables sur site (FPGA) sont des dispositifs semi-conducteurs à usage général qui peuvent être programmés directement par l'utilisateur. Ce sont des dispositifs matériels très flexibles et personnalisables pouvant implémenter n'importe quelle fonction logique qu'un circuit intégré spécifique à une application pourrait exécuter, en plus, la possibilité de mettre à jour la fonctionnalité offre des avantages pour de nombreuses applications.

Les CPLD sont des dispositifs logiques à grande échelle avec des milliers d'éléments logiques programmables, une mémoire non volatile et un bloc d'E/S sur une seule puce. Les connexions entre les éléments logiques de ces puces semi-conductrices sont personnalisées par l'utilisateur avec des outils logiciels. L'architecture CPLD a des performances de synchronisation et une vitesse prévisibles, offre une gamme de capacités logiques et est souvent utilisée dans des applications portables fonctionnant sur batterie. Les FPGA sont plus denses et plus complexes que les CPLD et sont utilisés pour mettre en œuvre des conceptions plus grandes ou plus complexes.

Un FPGA n'est limité à aucune fonction matérielle prédéterminée. Un FPGA permet de programmer les caractéristiques et les fonctions du produit, de s'adapter aux nouvelles normes et de reconfigurer le matériel pour des applications spécifiques, même après l'installation du produit sur le terrain, d'où le nom « programmable sur le terrain ». Contrairement aux FPGA de génération précédente utilisant des entrées et sorties (E/S) avec une logique et des interconnexions programmables, les FPGA d'aujourd'hui se composent de divers mélanges de SRAM embarquée configurable, d'émetteurs-récepteurs à grande vitesse, d'E/S à grande vitesse, de blocs logiques et de routage.

Les PLD présentent de nombreux avantages de conception, notamment un prototypage rapide, un délai de mise sur le marché plus court et la possibilité de reprogrammer sur le terrain. Les

FPGA sont passés de leur utilisation principale en tant que plate-forme de prototypage à une large utilisation dans la production en volume grand public aujourd'hui.

Chapitre 2

Synthèse et prototypage de circuit sur puce

2.1 Introduction

Au cours de ce vingt et unième siècle, une grande miniaturisation des circuits électroniques s'est produite. Cette échelle de la miniaturisation devrait se poursuivre au cours des prochaines décennies (loi de Moore). En effet, Gordon E. Moore, cofondateur de la société Intel, démontre dès 1965 dans son article [20], que le nombre de transistors par circuit de même taille double, à prix constant, tous les ans. Il porte par la suite ce délai à 18 mois et en déduit que la puissance des ordinateurs augmente de manière significative. Actuellement la taille du transistor s'approche de plus en plus de la taille des atomes de 5 nm. Dans un tel contexte, le domaine de la conception et du prototypage des circuits sur puce (System On Chip SOC) s'est considérablement développé dans le but de fournir des produits intelligents et rentables. Sur le marché mondial, les applications de la conception basée sur le SOC dans les domaines de la transmission sans fil, du multimédia, des processeurs, des contrôleurs, du traitement d'images et des protocoles d'interface se sont considérablement développées au cours de cette décennie. Cela a un réel impact sur le coût des produits en raison de la nature concurrentielle du marché. La perception de l'évolution technologique est liée surtout et aux évolutions des algorithmes EDA et des processus pour répondre aux besoins de la conception et de la validation du SOC. De nombreux fournisseurs d'EDA comme Xilinx, Intel FPGA, Synopsys, Cadence répondent aux besoins de la conception SOC. Ces sociétés disposent de la chaîne d'outils EDA sophistiquée et de la prise en charge des cartes FPGA haute densité.

Ce chapitre fournit un aperçu général des systèmes SoC typiques et présente l'approche de conception basée sur le système sur puce programmable (System On Programmable Chip SOPC) pour la construction de systèmes SoC.

2.2 Conceptions des systèmes sur puce SOC

En raison des exigences de conception à faible puissance et à grande vitesse dans diverses applications, la conception et le prototypage du SOC sont nécessaires. Pour n'importe quel SOC, la conception comporte des blocs analogiques et numériques. La figure 2.1 donne des informations sur quelques composants du SOC.

- **Processeur et Core-processeur** :

Avec un rappel historique de l'évolution du processeur, Intel a lancé un processeur commercial 8080 en avril 1974. Il avait un bus non multiplexé avec des lignes d'adresse et de données séparées. Le bus d'adresse était large de 16 bits et le bus de données était large de 8 bits. Le boîtier utilisé pour ce processeur commercial était un DIP à 40 broches et la fréquence d'horloge était de 2 MHz.

Donc, si l'on considère l'évolution des processeurs, alors la bande passante du bus et la vitesse pour réaliser l'exécution simultanée jouent un rôle important dans la conception de tout SOC.

Dernièrement, la conception de systèmes électroniques à l'aide de microprocesseurs est devenue une partie intégrante du cycle de développement de produits de systèmes embarqués pour développer des systèmes embarqués innovants. L'utilisation des core-processeur dans la conception réduit les autres composants car des milliers de transistors sont placés sur une petite zone de silicium pour effectuer les opérations requises. Avec l'évolution des nouveaux algorithmes et techniques de conception, le coût du processeur a considérablement diminué et même les processeurs à fonctionnalités modérées sont disponibles avec un très faible coût [21].

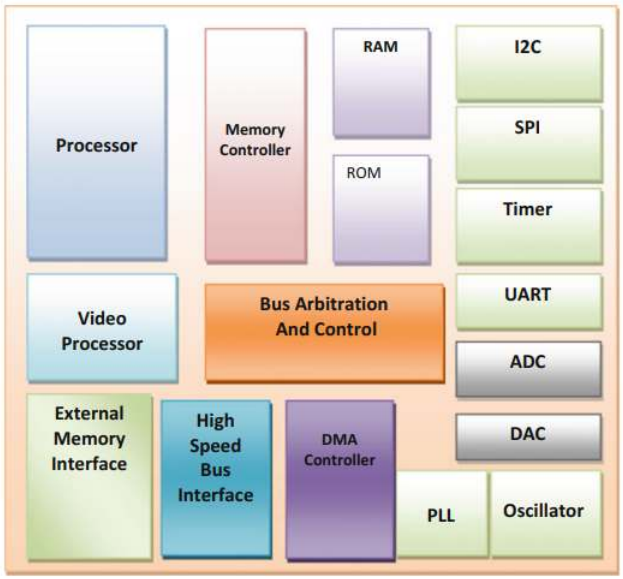


Figure 2. 1 Composants typiques dans un système SOC

La figure 2.2 donne des informations sur les 40 années de tendance dans la conception des microprocesseurs [22]

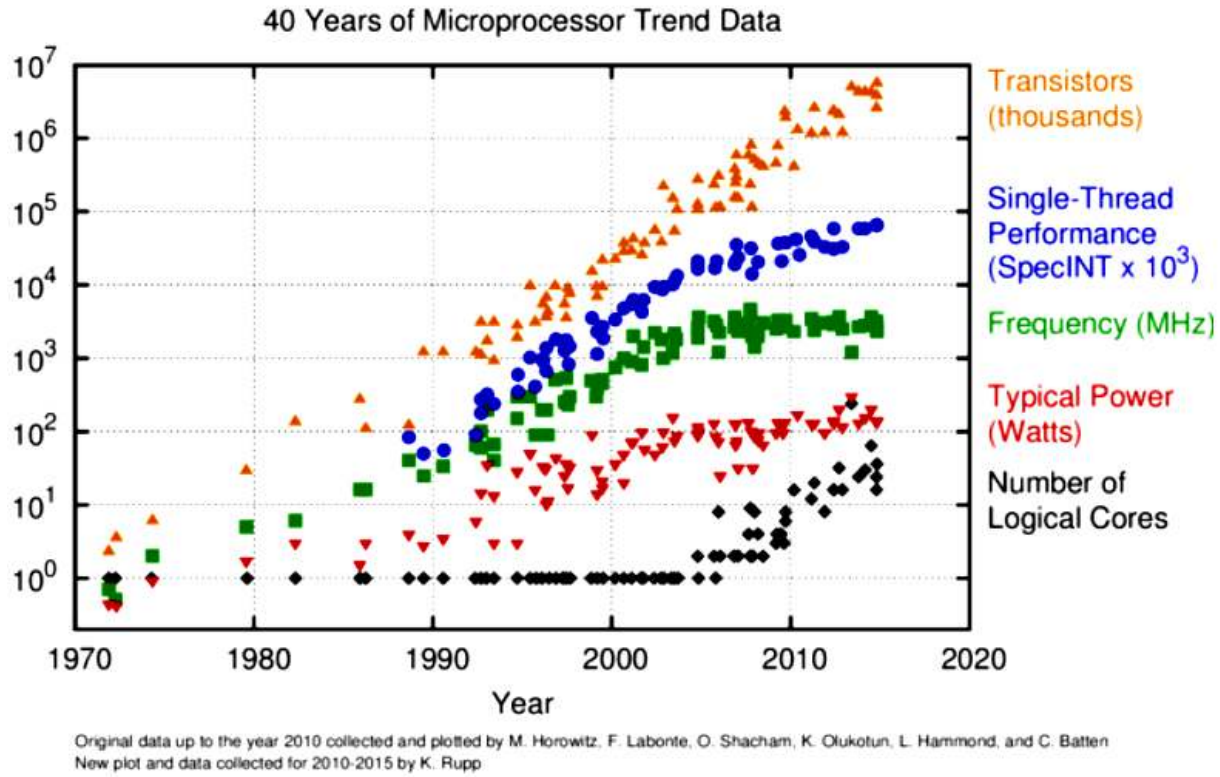


Figure 2. 2 Tendances des microprocesseurs

Les SOCs haute densité doivent comporter un ou plusieurs processeurs. L'architecture à processeurs multiples peut permettre l'exécution simultanée et le parallélisme lors de l'exécution des instructions. Dans la plupart des applications, les architectures de processeur haute vitesse et basse consommation sont nécessaires pour effectuer les opérations complexes. Ces opérations peuvent être : le transfert des données, les opérations en virgule flottante et le traitement audio vidéo. La plupart des SOC complexes ont des processeurs à usage général, des processeurs de traitement de signal, d'image et de vidéo DSP, ils sont utilisés pour améliorer les performances d'exécution globales du SOC.

- **Mémoire interne :**

En ce qui concerne le stockage interne des données, le SOC doit avoir des mémoires (RAM, ROM ou flash). Ces mémoires peuvent être des mémoires réparties ou disponibles sous forme de blocs mémoire. Les cœurs de mémoire (Memory-core) configurables peuvent être utilisés pour stocker les données nécessaires. Pour une structure processeur DSP, l'architecture peut être efficace si deux mémoires distinctes (données et programme) sont utilisées [23]. Cette stratégie peut être utile pour améliorer les performances globales de l'architecture.

- **Contrôleur de mémoire :**

Les contrôleurs DDR ou SDRAM peuvent être utilisés pour communiquer avec la mémoire DDR ou la mémoire SDRAM externe. Les contrôleurs DDR à fréquence d'horloge élevée peuvent être disponibles auprès des différents fournisseurs en tant que composant ‘’propriété intellectuelle’' (Intellectual Property IP). La synchronisation et l'utilisation d'IP éprouvées sur le plan fonctionnel peuvent réduire le temps de conception/vérification, et elles peuvent être intégrées aux composants SOC pour accomplir les tâches souhaitées [23].

- **Interface de bus à grande vitesse :**

Les performances du SOC dépendent fortement de l'architecture du bus et des schémas d'arbitrage. Pour transférer les données entre les différents composants SOC, la FIFO, des buffers, des bus sont utilisés. La logique d'interface de bus à grande vitesse peut être utilisée pour établir la communication avec l'hôte externe.

- **Interface mémoire externe :**

L'application peut nécessiter une mémoire flash ou SDRAM, ces mémoires peuvent être interfacées à l'aide de l'interface mémoire externe.

- **Contrôleurs DMA :**

Pour transférer une grande partie des données, les contrôleurs à accès directe mémoire (Direct Memory Acces DMA) peuvent être utilisés. Le transfert de données peut être établi pour la grande taille des données à grande vitesse.

- **Interfaces série :**

Les interfaces séries telles que l’I2C, le SPI, le one wire et l’UART peuvent être utilisées pour établir la communication entre les périphériques série et les composants internes du SOC

- **ADC et DAC :**

Les composants analogiques peuvent être interfacés avec les autres composants SOC à l'aide de convertisseur analogique numérique ADC et numérique analogique DAC.

- **Ressources d'horloge :**

Les oscillateurs et les boule de verrouillage de phase (phase lock loop PLL) intégrés peuvent être utilisés pour générer les horloges avec un décalage d'horloge uniforme. Le réseau de distribution d'horloge utilisant plusieurs PLL peut être utilisé pour prendre en charge le décalage d'horloge uniforme et les conceptions de domaines d'horloge multiples.

La section suivante traite du flux de conception du SOC.

2.3 Flux de conception SOC

Avec l'évolution de la technologie des processus à grande haute échelle d'intégration (Very High Scale Integration VHSI), les conceptions deviennent de plus en plus complexes et la conception basée sur le SOC est réalisable dans un temps de cycle de conception plus court en raison de la disponibilité des outils CAO (Computer Aded Design) de prototypage. La demande d'avoir un produit dans un temps de cycle de conception plus court est possible en utilisant un flux de conception efficace. La conception doit évoluer de l'étape des spécifications à la mise en fabrication finale. L'utilisation d'outils EDA (Electronic design automation) avec les fonctionnalités appropriées a permis d'avoir des conceptions sans bug avec des fonctionnalités éprouvées.[24]

Le flux de conception est illustré à la Figure 2.3 et comprend les étapes clés suivantes.

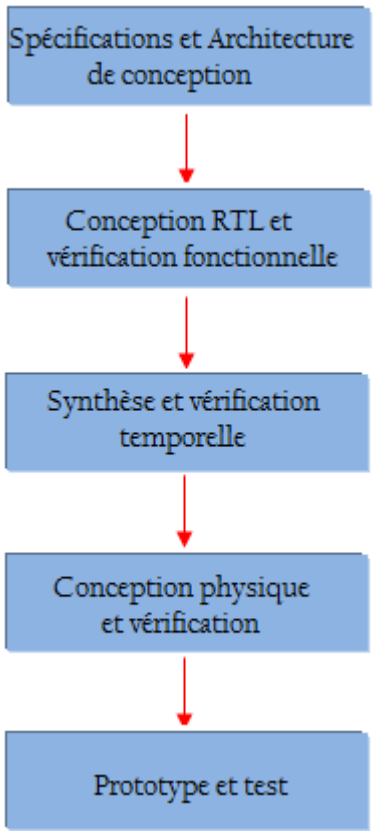


Figure 2. 3 Flux de conception SOC

2.3.1 Spécifications de conception et architecture du système

La détermination des spécifications fonctionnelles de conception pour l'ASIC ou le SOC est une phase importante. Au cours de cette phase, une étude de marché approfondie est réalisée pour figer les spécifications fonctionnelles de la conception.

Pour un smart-phone par exemple, les spécifications fonctionnelles importantes peuvent être la vitesse du processeur, les spécifications fonctionnelles du processeur, la mémoire interne, l'affichage et sa résolution, la caméra et la résolution de la caméra, les interfaces de communication externes, etc. En plus, il est essentiel d'avoir des informations sur l'assemblage mécanique et d'autres caractéristiques électriques du dispositif tel que les circuits d'alimentation et de charge de batterie et les dispositifs de sécurité. Sans oublier les contraintes environnementales et les contraintes de conception telle que la surface, la puissance. Les spécifications sont utilisées pour esquisser le plan d'étage supérieur de la puce c.à.d. l'architecture du SOC du mobile.

L'esquisse de l'architecture de milliard de portes SOC est l'une des tâches les plus difficiles car elle implique l'imagination et la compréhension réelles de l'interdépendance entre les composants matériels et logiciels. Pour éviter les frais généraux sur le processeur unique, la conception peut nécessiter plusieurs processeurs capables d'effectuer le multitâche.

Le document référentiel descriptif d'architecture est toujours élaboré à partir des spécifications de conception, il s'agit d'une représentation au niveau des blocs de la conception globale. Une équipe de professionnels expérimentés peut créer ce type de document dans le but d'être utilisé comme référence pour présenter la microarchitecture de la conception.

Le document référentiel de microarchitecture est l'abstraction de niveau inférieur des documents d'architecture, il donne des informations sur la fonctionnalité de chaque et leurs informations de synchronisation. Ce document devrait donner aussi des informations sur les IPs qui doivent être utilisées dans la conception ainsi que les détails de synchronisation et d'interface.

2.3.2 Conception RTL niveau portes-logique et vérification fonctionnelle

Pour les conceptions SOC complexes, le document de micro-architecture est utilisé comme référence par l'équipe de conception. La conception SOC utilisant des milliards de portes est divisée en plusieurs blocs, une équipe d'une centaine d'ingénieurs travaille à la mise en œuvre de la conception et à la vérification. Le concepteur niveau portes-logiques (Registre Transfert Level RTL) utilise les directives de conception et de codage recommandées lors de la mise en œuvre de

la conception RTL. Une conception RTL efficace joue toujours un rôle important pendant le cycle de mise en œuvre. Pendant ce temps, le concepteur décrit les fonctionnalités de niveau bloc et de niveau supérieur à l'aide d'un langage matériel HDL-RTL efficace.

Après l'achèvement d'une phase de conception niveau porte logique ou registre RTL efficace pour les spécifications de conception données, la fonctionnalité de conception est vérifiée à l'aide d'un simulateur standard de l'industrie. La simulation de présynthèse se fait sans aucun retard, dans l'objectif de vérifier la fonctionnalité de la conception. La pratique courante dans l'industrie consiste à vérifier la fonctionnalité en écrivant le banc de test et d'essai (test bench).

Le banc d'essai force le stimulus des signaux à la conception et surveille la sortie de la conception. Dans le scénario actuel, l'automatisation du flux de vérification et les nouvelles méthodologies de vérification ont évolué et sont utilisées pour vérifier la fonctionnalité de conception complexe dans un laps de temps plus court en utilisant le nombre requis de ressources. Le rôle de l'ingénieur de vérification est de tester les discordances fonctionnelles entre la sortie attendue et la sortie réelle. Si une inadéquation fonctionnelle est trouvée pendant la simulation, elle doit être rectifiée avant de passer à l'étape de synthèse. La vérification fonctionnelle est un processus itératif à moins que et jusqu'à ce que la conception réponde à la fonctionnalité requise. Pour de meilleurs de résultats, l'équipe d'ingénieurs de vérification utilise le document du plan de vérification. Cela peut aboutir à de meilleurs objectifs de couverture.

2.3.3 Synthèse et vérification du temps

Lorsque les exigences fonctionnelles de la conception sont satisfaites, l'étape suivante est la synthèse. L'outil de synthèse utilise le HDL (VHDL ou Verilog) RTL, les contraintes de conception et les bibliothèques en tant qu'entrées et génère le netlist au niveau de la porte en tant que sortie. La synthèse est un processus itératif jusqu'à ce que les contraintes de conception soient respectées. Comme nous l'avons mentionné, les principales contraintes de conception sont la surface, la vitesse et la puissance. Si les contraintes de conception ne sont pas respectées, la re-synthèse doit être effectuée pour réaliser une optimisation supplémentaire sur la conception RTL. Après l'optimisation, si les contraintes ne sont pas respectées, alors la modification du code RTL devra être modifiée le code RTL. L'outil de synthèse génère les rapports de zone, de vitesse et de puissance, ainsi que le netlist au niveau de la porte en tant que sortie.

La vérification de la synchronisation est effectuée en utilisant le netlist au niveau de la porte. Cette phase est utile pour trouver les non-concordances de simulation de présynthèse et de post-synthèse.

L'analyse temporelle du prélayout est également une phase importante pour corriger les violations de configuration dans la conception. Les violations de blocage peuvent être corrigées au cours d'une étape ultérieure du cycle de conception lors de l'analyse de synchronisation post-implantation.

2.3.4 Conception physique et vérification

Cela implique la planification au sol de la conception, la planification de l'alimentation, le lieu et l'itinéraire, la synthèse de l'arbre d'horloge, la vérification post-disposition, l'analyse de synchronisation statique et la génération du fichier graphique de GDSII (Graphic Database System Information Interchange) pour une conception ASIC.

2.3.5 Prototype et test

Au cours de cette phase, le prototype de conception utilisant le FPGA peut être validé et testé pour comprendre si la conception répond aux performances, au calendrier et à la fonctionnalité requis. Cette phase est un jalon chronophage et utile pour réduire les risques globaux par la détection précoce des bogues. Au fur et à mesure que la preuve de concept est validée, elle peut être utilisée pour éviter la réorientation des conceptions ASIC/SOC complexes.

2.4 Architectures soc typiques

Cette partie fournit un aperçu des systèmes SoC typiques et présente l'approche de conception basée sur le système sur puce programmable (SOPC) pour la construction de systèmes SoC. Elle aborde aussi quelques interconnexions SoC typiques et présente la structure d'interconnexion du système d'Altera, qui est la principale interconnexion sur puce envisagée pour cette recherche.

2.4.1 Cœurs de processeur SoC

Une nouvelle technologie a émergé dans les conceptions de SoC, permettant aux concepteurs d'utiliser un grand FPGA qui contient à la fois des éléments de mémoire, de logique ainsi qu'un cœur de processeur de propriété intellectuelle (IP) pour implémenter du matériel personnalisé

pour les applications SoC. Cette nouvelle approche est appelée système sur puce programmable (SOPC) [25].

Les cœurs de processeur sont classés selon deux types « hard » ou « soft » en fonction de leur flexibilité/configurabilité. Les cœurs de processeur hard sont moins configurables mais ont tendance à avoir des caractéristiques de performances plus élevées que les cœurs soft. Les cœurs de processeur hardware utilisent un cœur de processeur intégré entièrement implémenté (en silicium dédié) en plus des éléments logiques normaux du FPGA. Les cœurs de processeur matériels ajoutés à un FPGA créent une approche hybride, offrant des caractéristiques de performances qui se situent quelque part entre un circuit intégré à application spécifique (ASIC) traditionnel et un FPGA. Ils sont disponibles auprès de plusieurs fabricants avec un certain nombre de types de processeurs différents. Par exemple, Altera Corporation [28], l'un des principaux fabricants de dispositifs logiques reprogrammables, propose un cœur de processeur Advanced RISC Machines (ARM) intégré dans sa famille de FPGA APEX 20KE qui est commercialisé sous le nom de dispositif Excalibur^{TM}. Xilinx Inc., un autre fabricant leader de dispositifs logiques, fabrique une famille de FPGA appelée Virtex II pro qui comprend jusqu'à quatre cœurs de processeur PowerPC sur puce. Cypress Semiconductor propose également un système sur puce programmable (PSoC^{TM}) qui est formé sur un cœur de processeur M8C avec des blocs logiques configurables conçus pour implémenter les interfaces périphériques, qui comprennent des convertisseurs analogique-numérique, des temporisateurs, des compteurs et des UART [32]

Les cœurs de processeur soft, parfois appelés cœurs synthétisables, utilisent les éléments logiques programmables existants du FPGA pour mettre en œuvre la logique du processeur. Les noyaux souples consistent en un modèle matériel synthétisable, généralement sous la forme d'un netlist ou d'un code HDL (Hardware Description Language) extrêmement flexible et facile à optimiser [26]

Le Nios II [30], le MicroBlaze [32], le PicoBlaze [33] et le Xtensa [37,49] sont les principaux processeurs soft-core fournis par Altera [28], Xilinx et Tensilica [34] respectivement. Chacun d'entre eux a ses caractéristiques.

2.4.2 Le NIOS d'Altera

Le Nios II par Altera Corporation : Altera Corporation [28,30] est l'un des principaux fournisseurs de dispositifs logiques programmables (PLD) et de FPGA. Ils proposent les familles

de FPGA Stratix, Stratix II et Cyclone qui sont largement utilisées dans la conception de systèmes embarqués et d'applications de traitement numérique du signal (DSP). Ils fournissent également des outils de CAO associés tels que Quartus II et System-on-Programmable-Chip (SOPC) Builder qui permettent aux concepteurs de synthétiser, programmer et déboguer leurs conceptions et de construire des systèmes embarqués sur les FPGA d'Altera.

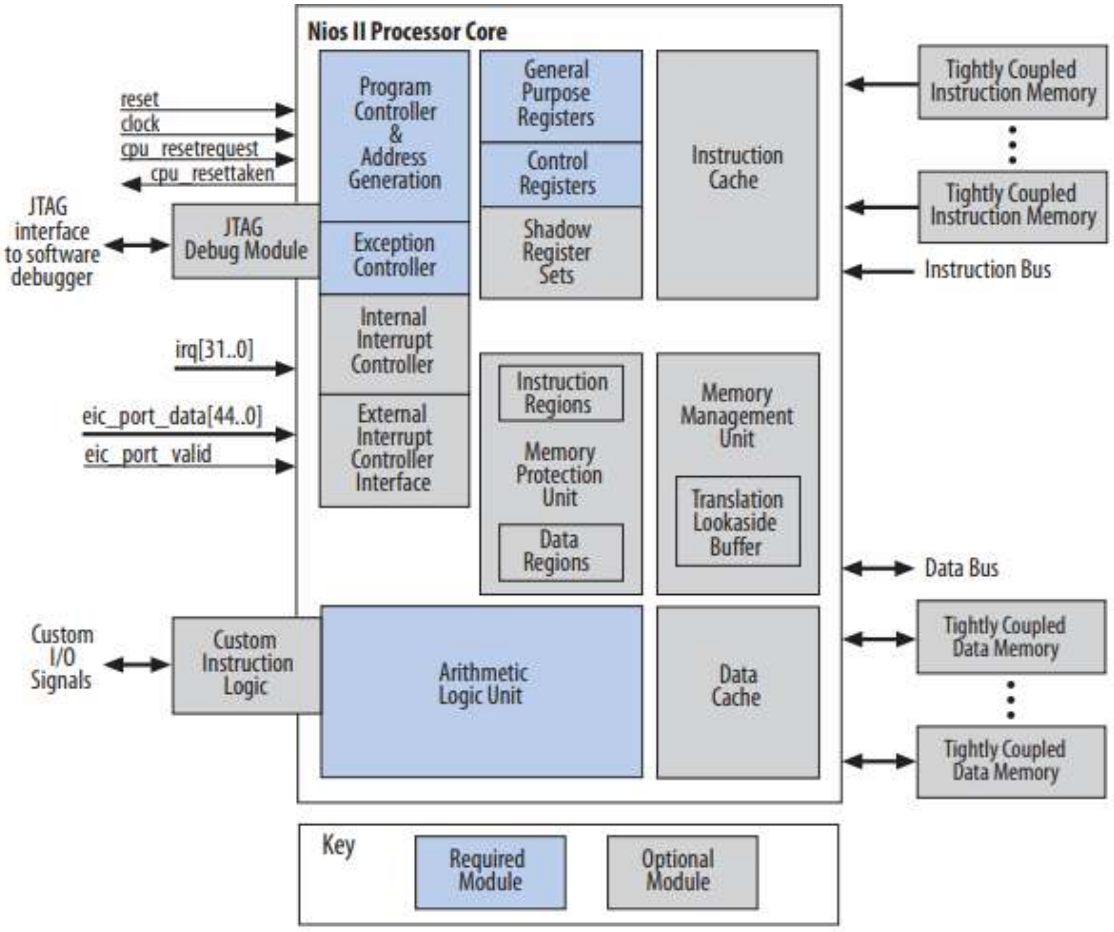


Figure 2. 4 Structure du processeur Nios-II Altera

Le processeur Nios II [29,30] est leur processeur IP phare et peut être instancié avec n'importe quelle conception de système embarqué. Ce processeur est le successeur du processeur soft-core Nios d'Altera et présente des améliorations majeures axées sur la réduction de la consommation d'éléments logiques (LE) sur un FPGA et l'amélioration des performances.

Le processeur Nios II Soft-Core est un cœur de processeur d'ordinateur à jeu d'instructions réduit (RISC) à usage général et dispose d'une architecture de mémoire Harvard. Ce noyau est largement utilisé avec les FPGA Altera et SOPC Builder. Ce processeur dispose d'une architecture de jeu d'instructions (ISA) 32 bits complète, de 32 registres à usage général, d'opérations de multiplication et de division 32x32 à instruction unique et d'instructions dédiées pour les produits de multiplication 64 bits et 128 bits. Le Nios II a également une performance

de plus de 150 Dhrystone MIPS (DMIPS) sur la famille de FPGA Stratix. Ce processeur soft-core se décline en trois versions : économique, standard et rapide. Chaque version principale modifie le nombre d'étages de pipeline, les mémoires cache d'instructions et de données et les composants matériels pour les opérations de multiplication et de division. De plus, chaque cœur varie en taille et en performances en fonction des fonctionnalités sélectionnées.

L'ajout de périphériques avec les processeurs Nios II se fait via le bus d'interface Avalon qui contient la logique nécessaire pour interfacer le processeur avec des cœurs IP standard ou des périphériques sur mesure.

2.4.3 Le Pico, MicroBlaze de Xilinx

MicroBlaze et PicoBlaze par Xilinx Incorporated : Xilinx Incorporated [31,32,33] sont les fabricants des familles Spartan et Virtex de FPGA. En outre, ils proposent également des cœurs IP souples qui ciblent leurs FPGA. Leur cœur le plus populaire et le plus utilisé est le processeur à cœur doux MicroBlaze [29,32], qui est un processeur 32 bits optimisé pour les applications embarquées. Il peut fonctionner jusqu'à 200 MHz sur un FPGA Virtex-4, et il dispose d'une architecture Harvard RISC, d'instructions 32 bits, d'un pipeline à 3 étages, d'un fichier de registre large à 32 registres, d'une unité de décalage et de deux niveaux d'interruption. La mémoire peut résider sur la puce ou en tant que périphérique externe. La mémoire sur puce est accessible par MicroBlaze à l'aide d'un bus de mémoire locale (LMB) [32], qui fournit un accès à cycle unique à la mémoire. En outre, une interface à usage général connue sous le nom de bus périphérique sur puce (OPB) peut être utilisée pour interfacer MicroBlaze avec les mémoires sur puce et hors puce ainsi qu'avec d'autres périphériques.

Le processeur MicroBlaze comprend un large ensemble de paramètres configurables qui peuvent être définis au moment de la conception, y compris une unité à virgule flottante (FPU) simple précision compatible IEEE-754 en option, un diviseur matériel, un décaleur de barillet, des caches de données et d'instructions, des capacités de gestion des exceptions, la logique de débogage du matériel et certaines fonctionnalités supplémentaires. Les concepteurs disposent également d'une interface à faible latence vers le pipeline MicroBlaze via l'interface Fast Simplex Link (FSL), qui peut inclure jusqu'à huit ports d'entrée et de sortie 32 bits dédiés [32].

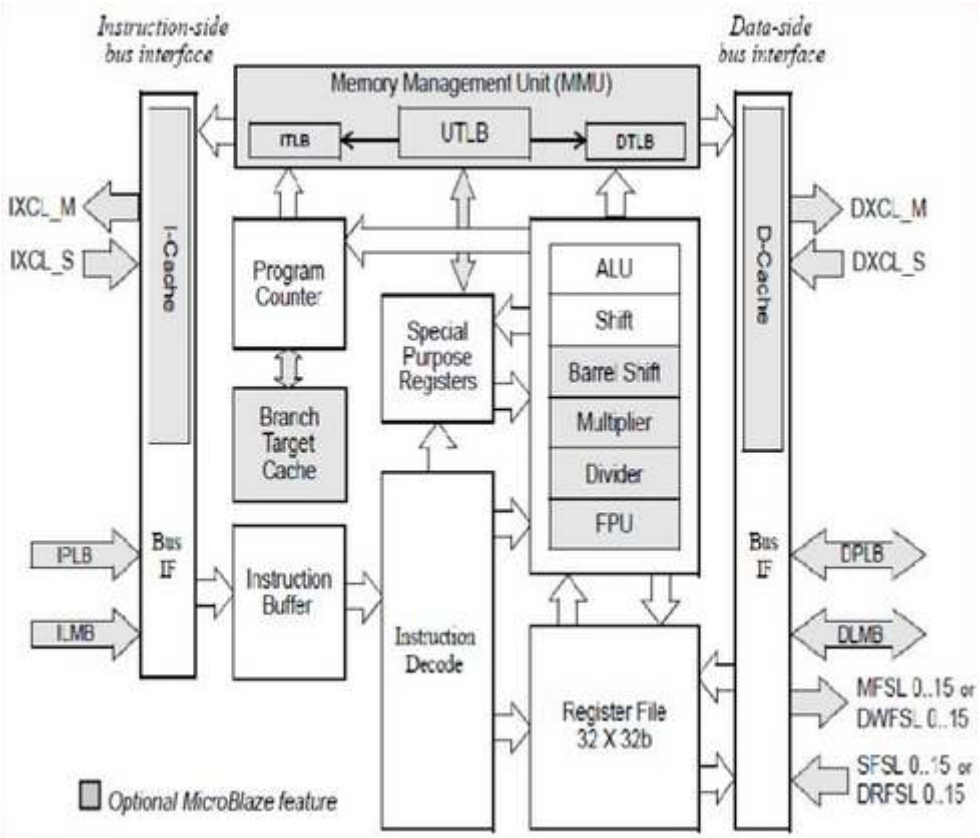


Figure 2. 5 structure du processeur MicroBlaze de Xilinx [32]

Le FSL peut être utilisé pour connecter des modules matériels personnalisés directement au pipeline afin d'accélérer les tâches urgentes. Le processeur à cœur doux MicroBlaze est destiné aux familles de FPGA Virtex et Spartan uniquement.

Xilinx propose également le kit de développement intégré (EDK) qui comprend le studio de plate-forme Xilinx et un ensemble de cœurs IP requis pour développer des systèmes intégrés à l'aide de MicroBlaze. Xilinx fournit également le processeur à cœur doux PicoBlaze [33], un microcontrôleur compact 8 bits optimisé pour les familles de FPGA Spartan-3, Virtex-II et Virtex-II Pro.

Le processeur PicoBlaze est un petit processeur à noyau souple économique qui est utile pour les applications de traitement de données simples. Série Diamond Standard et Xtensa [37,49] de Tensilica Inc.[34,50] :

2.4.4 Le Diamond, Xtens de Tensilica

Tensilica Inc. propose un certain nombre de cœurs de traitement IP souples pour la conception de systèmes embarqués. Leur série Diamond [35,36] Standard de processeurs synthétisables [36]

offre un ensemble de six processeurs prêts à l'emploi préconfiguré avec une variété de fonctionnalités.

Leur taille et leur ensemble de fonctionnalités vont des petits contrôleurs à puissance optimisée tels que le contrôleur 108Mini RISC aux grands cœurs hautes performances tels que le 545CK, qui a été optimisé pour les applications DSP. Les cœurs de la série Diamond Standard sont actuellement disponibles sous forme de descriptions HDL synthétisables des processeurs écrites en Verilog [37].

Le produit phare de Tensilica est la série Xtensa de processeurs « configurables et extensibles » [37,38,49]. Ils sont configurables en ce sens qu'ils offrent au concepteur un ensemble de paramètres prédéfinis qu'ils peuvent configurer afin d'adapter le processeur à l'application visée. Ils sont également extensibles, dans la mesure où les concepteurs peuvent également inventer des instructions personnalisées et des unités d'exécution et les intégrer directement dans le cœur du processeur.

Le processeur Xtensa est étendu à l'aide du langage Tensilica Instruction Extension (TIE) [37,39], qui est un langage de type Verilog qui peut être utilisé pour décrire des instructions personnalisées. Les concepteurs peuvent écrire le code TIE manuellement et le compiler à l'aide du compilateur TIE [40], ou ils peuvent utiliser le compilateur XPRES (Xtensa Processor Extension Synthesis) [41] pour créer automatiquement des descriptions TIE des extensions de processeur. Le compilateur XPRES peut analyser un algorithme donné écrit en C/C++ et configurer et étendre automatiquement le processeur Xtensa afin qu'il soit optimisé pour exécuter cet algorithme particulier. Enfin, le générateur de processeur Xtensa peut être utilisé pour générer des descriptions HDL du processeur personnalisé, ainsi qu'un ensemble de scripts d'automatisation de la conception électronique (EDA) et une suite complète d'outils de développement logiciel spécialement adaptés à cette conception de processeur.

Tensilica propose actuellement deux versions différentes du processeur Xtensa : le Xtensa LX et le Xtensa 6 [48,49] . Les deux sont configurables et extensibles à l'aide de TIE et du compilateur XPRES, mais ils fournissent chacun leur propre ensemble de fonctionnalités et d'options configurables. Le Xtensa LX est le produit haut de gamme de Tensilica et est optimisé pour les DSP hautes performances et d'autres applications gourmandes en données [50]. En revanche, le Xtensa 6 est conçu pour optimiser la consommation d'énergie et est bien adapté à une grande variété d'applications différentes.

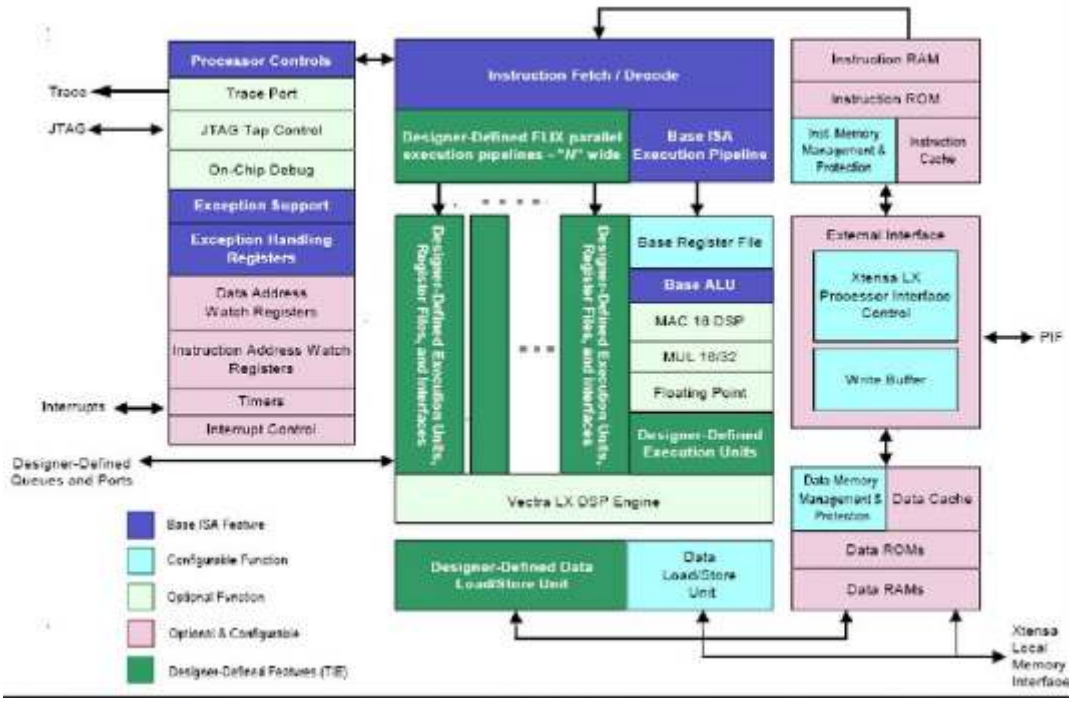


Figure 2. 6 Structure du processeur Xtensa de Tensilica [36]

2.5 Cœurs open source

Les cœurs open source sont des composants IP disponibles gratuitement dans la communauté open source [42]. Habituellement, ces types de cœurs sont utilisés dans les universités pour la recherche ainsi que dans le développement de systèmes embarqués. UT Nios [43] est un exemple de processeur open-core utilisé dans le milieu universitaire. Sun Microsystems fournit son propre processeur à cœur souple, OpenSPARC [44], qui est largement utilisé dans le développement des ASIC, mais il peut également être synthétisé pour les FPGA. Dans cette section, nous passerons en revue les processeurs soft-core LEON et OpenRISC 1200 disponibles dans la communauté open source.

LEON par Gaisler Research : Gaisler Research [45] est un fournisseur de cœurs IP et d'outils de développement de support pour les processeurs embarqués basés sur l'architecture SPARC. Leur produit principal est la gamme LEON de processeurs à cœur doux synthétisables et la bibliothèque associée de cœurs IP, appelée GRLIB IP Library [47]. Plusieurs versions successives du processeur LEON ont été développées, et actuellement Gaisler Research maintient et fournit les processeurs LEON2 et LEON3.

LEON2 [46] et LEON3 [47] sont des modèles VHDL open source d'un cœur de traitement 32 bits entièrement conforme à l'architecture standard IEEE-1754 SPARC V8. Les cœurs sont livrés

avec des unités entières compatibles SPARC V8 avec des unités matérielles de multiplication, de division et de MAC. LEON2 dispose d'un pipeline à 5 étages, tandis que LEON3 a une profondeur de pipeline de 7 étages. Les deux cœurs disposent d'une architecture de mémoire Harvard et d'un sous-système de cache associatif en ensemble configurable. Le nombre de registres dans leurs fichiers de registres est paramétrable au sein de la norme SPARC V8 (2 à 32 registres). LEON2 et LEON3 fournissent également une interface à l'un des nombreux cœurs d'unités à virgule flottante (FPU) disponibles ainsi qu'à des coprocesseurs personnalisés. Ils incluent également la prise en charge d'une unité de débogage en option, de minuteries, de chiens de garde, d'UART et de contrôleurs d'interruption.

OpenRISC 1200 : OpenRISC 1200 [42] est l'un des processeurs à noyau ouvert les plus populaires disponibles sur OpenCores.org. Ce processeur soft-core est doté d'une architecture RISC 32 bits et 64 bits adaptée à de nombreuses applications, notamment les dispositifs de mise en réseau et de télécommunication, le divertissement à domicile, les produits de consommation et les applications automobiles. Ce processeur est optimisé pour des performances élevées, une faible consommation d'énergie et une polyvalence dans une large gamme d'applications.

Le processeur présente une architecture Harvard contenant des caches de données et d'instructions séparés, chacun d'une taille de 8 Ko. Il dispose d'un ISA 32 bits contenant le jeu d'instructions de base OpenRISC (ORBIS32), un pipeline scalaire à 5 étapes à émission unique offrant un débit soutenu et une exécution d'instructions à cycle unique sur la plupart des instructions. Les performances maximales qu'il peut fournir sont de 250 DMIPS à une fréquence d'horloge de 250 MHz. OpenRISC a la capacité d'étendre l'architecture du jeu d'instructions en ajoutant des composants supplémentaires tels qu'un FPU. Jusqu'à 8 unités peuvent être ajoutées au noyau et contrôlées via des registres ou des instructions personnalisées définies par l'utilisateur. Ce processeur peut être synthétisé et téléchargé sur les FPGA Altera et Xilinx et prend en charge les systèmes d'exploitation temps réel embarqués tels que Linux, µLinux et le système d'exploitation temps réel OAR RTEMS. Pour le développement de logiciels, des outils sont disponibles qui permettent aux développeurs de compiler des programmes écrits en C/C++, Java et Fortran pour s'exécuter sur le processeur OpenRISC.

Le tableau I ci-dessous présente une comparaison des fonctionnalités et caractéristiques de plusieurs processeurs.

Tableau 1 Comparison des processeurs soft- core pour FPGA [51]

	Nios II (Fast Core)	**MicroBlaze**	**Xtensa XL**	**OpenRISC 1200**	**LEON3**
Maximum MHz	200 (FPGA)	200 (FPGA)	350 (ASIC)	300 (ASIC)	400/125 (ASIC/FPGA)
ASIC/FPGA Technology	– /Stratix and Stratix II	– /Virtex-4	0.13 µm/ –	0.18 µm/ –	0.13 µm/Not given
Reported DMIPS	150 DMIPs	166 DMIPs	–	250 DMIPS	85 DMIPs
ISA	32-bit RISC	32-Bit RISC	32-Bit RISC	32-bit RISC	32 or 64-bit RISC
Cache Memory (I/D)	Up to 64 KB	Up to 64 KB	Up to 32 KB (1)	Up to 64 KB	Up to 256 KB
Floating Point Unit (optional)	IEEE-754	IEEE-754	IEEE-754	As peripheral	IEEE-754
Pipeline	6 Stages	3 Stages	5 Stages	5 Stages	7 Stages
Custom Instructions	Up to 256 Instructions	None	Unlimited	Unspecified limit	None
Register File Size	32	32	32 or 64	32	2 to 32
Implementation	FPGA	FPGA	FPGA, ASIC	FPGA, ASIC	FPGA, ASIC
Area	700-1800 LEs	1269 LUTs	0.26 mm2	N/A	N/A

La première colonne présente les caractéristiques à travers lesquelles les processeurs sont comparés. Les colonnes suivantes montrent les fonctionnalités disponibles à l'intérieur de chaque processeur soft-core.

Comme indiqué dans le tableau, LEON3 a la fréquence de fonctionnement la plus élevée pour l'implémentation ASIC mais a la plus faible pour l'implémentation FPGA. Les processeurs à cœur souple Nios II et Microblaze ont la fréquence de fonctionnement la plus élevée pour

l'implémentation FPGA. Cependant, les fréquences pour la mise en œuvre de l'ASIC ne sont pas mentionnées.

Tous les processeurs soft-core ont des unités à virgule flottante optionnelles qui sont soit incluses dans leur architecture, soit instanciées en tant que périphérique connecté au cœur.

Xtensa XL est le noyau le plus exible parmi tous ceux répertoriés dans le tableau. Cela est dû au fait que les concepteurs peuvent créer un nombre illimité d'instructions personnalisées et d'unités d'exécution en utilisant le langage TIE et les intégrer directement dans le cœur du processeur. De plus, Xtensa propose également un grand nombre de paramètres configurables parmi lesquels le concepteur peut choisir. Nios II a également la capacité d'étendre son jeu d'instructions en ajoutant jusqu'à 256 instructions personnalisées. MicroBlaze n'a pas ce genre de fonctionnalité.

Les processeurs soft-core Nios II et Microblaze sont ciblés et optimisés principalement pour l'implémentation FPGA. En revanche, les trois autres cœurs ne sont optimisés pour aucune technologie cible spécifique.

Chaque noyau étudié présente des caractéristiques de performances et des fonctionnalités différentes qui conviennent à des applications spécifiques.

Les concepteurs de systèmes embarqués doivent choisir un cœur de processeur en fonction des exigences et des contraintes de performances de leur application particulière.

La travail abordé dans cette thèse de recherche utilise le cœur du processeur logiciel Nios II d'Altera et adopte l'approche de conception SOPC pour la construction de systèmes SoC multiprocesseurs.

2.6 Système sur puce Programmable SOPC et Flux de conception SOPC

Le flux traditionnel des outils de CAO commerciaux suit généralement un chemin allant du langage de description matérielle (HDL) ou de l'entrée de conception schématique à la synthèse et aux outils de placement et d'acheminement jusqu'à la programmation du FPGA. Les fabricants de FPGA fournissent des outils de CAO tels que le logiciel Quartus-II d'Altera et le logiciel ISE de Xilinx, qui guident le concepteur tout au long de ce processus. Comme le montre la figure 2.7, l'ajout d'un cœur de processeur et les outils qui lui sont associés sont un sur-ensemble des outils traditionnels. La synthèse standard, le lieu et l'itinéraire, et la fonctionnalité de programmation

sont toujours nécessaires, et dans le cas d'Altera et de Xilinx, les mêmes outils de CAO (Quartus II ou ISE) sont utilisés pour implémenter ces blocs.

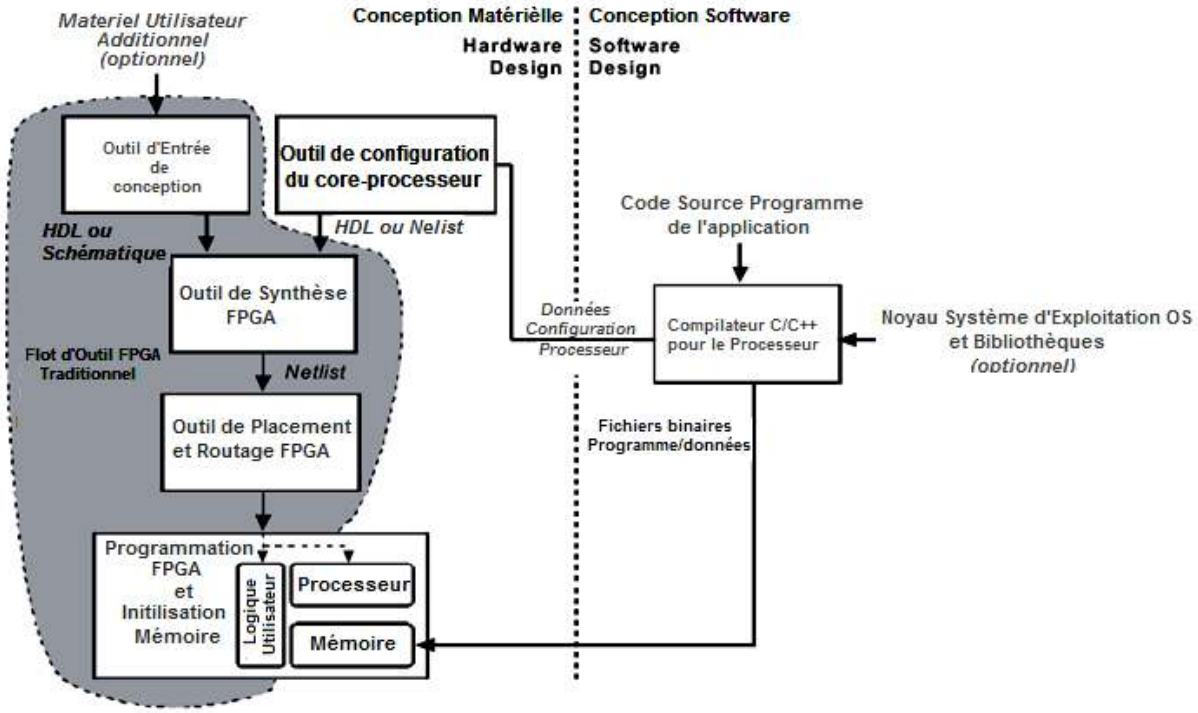


Figure 2. 7 Le flux d'outils CAO pour la conception SOPC

2.6.1 Outils de configuration du cœur du processeur

Aujourd'hui, un certain nombre de cœurs de processeur prédéfinis sont disponibles auprès de diverses sources. Les cœurs de processeurs publics sous licence GPL peuvent être trouvés sur le Web (par exemple, www.opencores.org et www.leox.org), tandis que des sociétés telles qu'Altera (processeur Nios II), Xilinx (processeur MicroBlaze) et Tensilica (processeur Xtensa) fournissent leurs processeurs et/ou outils de développement moyennant des frais.

Les cœurs de processeur fournis par les fabricants de FPGA sont généralement optimisés manuellement pour la famille de FPGA spécifique utilisée et, en tant que tels, sont plus efficacement mis en œuvre sur le FPGA qu'un cœur conçu par un simple concepteur (en particulier compte tenu des contraintes de temps et de ressources de la plupart des projets). De plus, les sociétés FPGA fournissent des outils de support complets pour faciliter la personnalisation et l'utilisation de leurs cœurs, y compris des compilateurs et des débogueurs de haut niveau ciblés sur les cœurs personnalisés

Dans le cas d'Altera et de Xilinx, le bloc Processor Core Configuration Tool illustré à la Fig. 2.7 est réalisé dans une interface graphique conviviale qui permet au concepteur de personnaliser le processeur pour un projet particulier. Une capture d'écran de l'assistant de configuration du processeur d'Altera est illustrée à la figure 2.8. Les paramètres configurables peuvent inclure la largeur du chemin de données, la mémoire, l'espace d'adressage et les périphériques (y compris les E/S à usage général définies arbitrairement, les UART, les contrôleurs Ethernet, les contrôleurs de mémoire, etc.). Une fois les paramètres du processeur spécifiés dans l'interface graphique, le cœur du processeur est généré sous la forme d'un fichier HDL (dans Altera) ou d'un fichier netlist (dans Xilinx). Ce fichier peut ensuite être inclus dans une conception HDL ou schématique traditionnelle à l'aide des outils de CAO standard.

Des affectations de broches spécifiques avec une logique utilisateur supplémentaire peuvent être incluses à ce stade comme toute autre conception de FPGA. Ensuite, la conception matérielle complète (cœur du processeur et toute logique utilisateur supplémentaire) est compilée (synthèse, lieu et route, etc.), et le FPGA peut être programmé avec le fichier résultant à l'aide des outils standard. La conception matérielle est terminée et la logique FPGA a été déterminée.

2.6.2 Compilateur de haut niveau pour le cœur du processeur

Comme indiqué sur la figure 2.7 (partie droite), l'étape suivante consiste à écrire et à compiler le programme qui sera exécuté sur le cœur du processeur logiciel. Lorsque l'outil de configuration de cœur de processeur génère les fichiers HDL ou netlist, il crée également un certain nombre de fichiers de bibliothèque et leurs fichiers d'en-tête C associés qui sont personnalisés pour le cœur de processeur spécifique généré. Un compilateur C/C++ destiné à ce processeur est également fourni. Le concepteur peut alors programmer des programmes autonomes à exécuter sur le processeur. En option, le concepteur peut compiler du code pour un système d'exploitation ciblé pour le cœur du processeur. Plusieurs systèmes d'exploitation pour le Nios II sont disponibles auprès de fournisseurs tiers ainsi que les uC/*OS* open source pris en charge par la communauté [52,53]

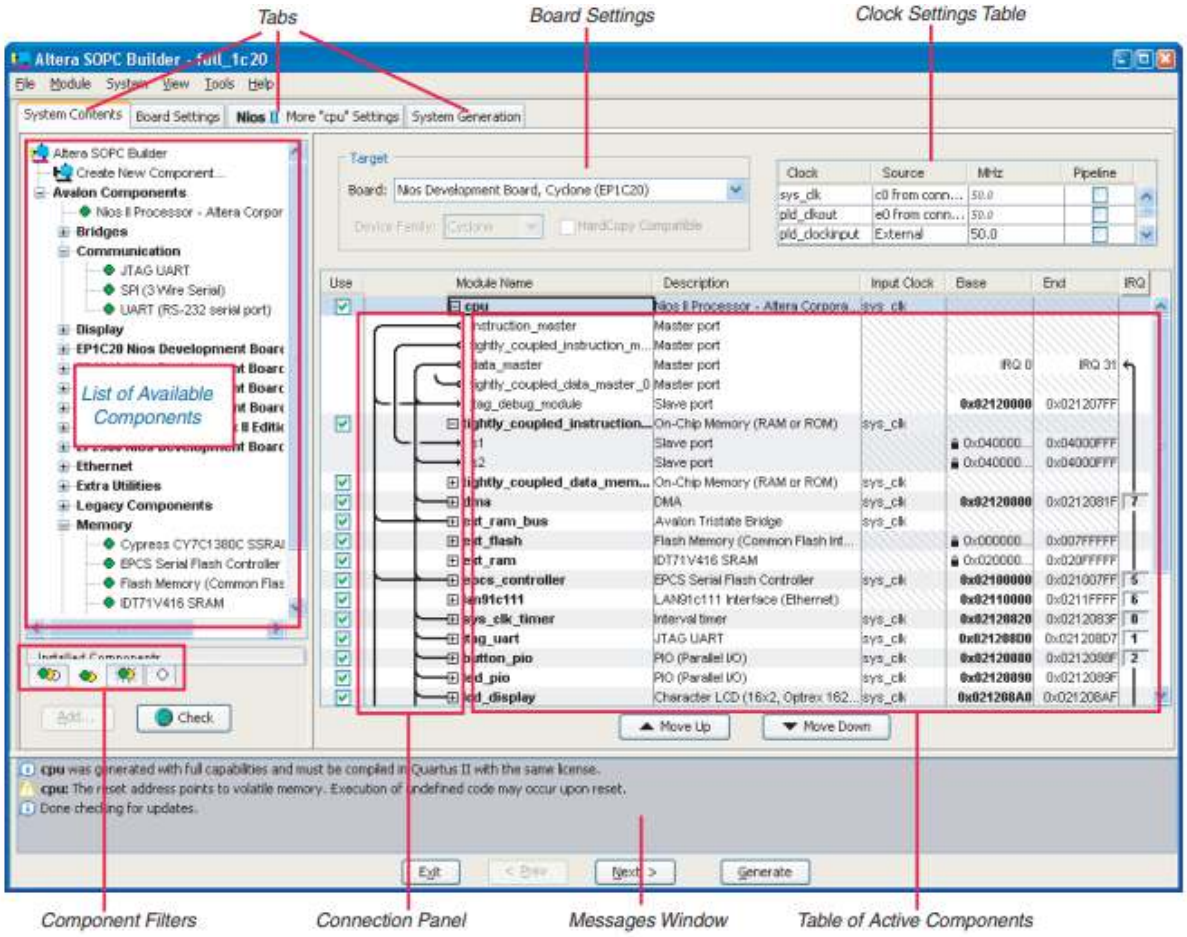


Figure 2. 8 Interface graphique de l'outil de configuration pour le Soft-core processeur Nios II.

2.6.3 Mémoire

Une fois qu'un fichier binaire programme/données a été généré, il doit être chargé dans les mémoires programme et données du processeur. Ce chargement peut se faire de plusieurs manières en fonction de la configuration mémoire du processeur à portée de main. Si le programme d'application est petit et peut tenir dans les blocs mémoire disponibles sur le FPGA, alors le programme peut être initialisé dans la mémoire lorsque la configuration matérielle est programmée. Cette initialisation se fait via les outils FPGA standard, tels que le logiciel Quartus II d'Intel Altera ou le logiciel Vivado ISE de Xilinx. Cependant, la mémoire sur puce est généralement très limitée et cette solution n'est généralement pas une option réaliste. La plupart des systèmes SoPC ont une ou deux petites puces de mémoire externes en plus du FPGA. Si des contrôleurs de mémoire sont nécessaires, ils sont implémentés à l'aide d'une logique FPGA interne.

2.6.3.1 Initialisation de la mémoire de programme

Dans un environnement de prototypage, le programme d'application sera très probablement modifié plusieurs fois avant que le programme final ne soit définitif. Dans ce cas, il faut prévoir la possibilité de télécharger le code de l'application depuis un PC vers la mémoire d'une carte FPGA. Cette fonctionnalité, généralement appelée « bootloader » ou « moniteur de démarrage », peut être implémentée dans un logiciel ou un matériel.

Un bootloader logiciel est composé de code qui est chargé dans une mémoire sur puce et commence à s'exécuter à la mise sous tension. Ce programme est suffisamment petit (1-2 Ko) pour tenir dans la plupart des mémoires sur puce, et sa fonction principale est de recevoir un fichier binaire de programme du PC de développement, de le charger dans la mémoire externe, puis de lancer l'exécution du nouveau code. De cette façon, un nouveau programme peut être stocké dans une mémoire externe (SRAM, SDRAM, mémoire Flash, etc.) en le téléchargeant sur une interface USB (ou autre) à la volée sans avoir à recharger la configuration matérielle du FPGA. Xilinx fournit un moniteur de démarrage pour son processeur à cœur MicroBlaze qui inclut la possibilité de télécharger un programme binaire via USB (ou une autre interface), de le stocker en mémoire et de démarrer l'exécution du code. [54]

Ils fournissent également une version plus améliorée appelée XMDstub qui ajoute des capacités de débogage. Les processeurs Nios hérités d'Altera comprenaient un bootloader logiciel; cependant, un bootloader matériel est la solution préférée dans Nios II.

Un bootloader matériel fournit des fonctionnalités très similaires à un bootloader logiciel ; cependant, il est implémenté dans la logique matérielle au sein du cœur du processeur. En règle générale, le processeur sera mis en pause ou bloqué lors de la mise sous tension et le chargeur de démarrage matériel aura un accès direct à la mémoire ou aux registres de mémoire dans le chemin de données du processeur. Le matériel du chargeur de démarrage peut démarrer et arrêter le processeur et contrôler le téléchargement d'un programme via l'interface USB, JTAG ou série vers les emplacements de mémoire souhaités. Le chargeur de démarrage matériel d'Altera fait partie du module de débogage JTAG, qui réside dans le processeur Nios II. Cette logique utilise l'interface JTAG avec le PC pour recevoir le code d'exécution, puis écrit le code dans la mémoire appropriée. Enfin, le matériel du chargeur de démarrage initie le compteur de programme du

processeur avec l'adresse de début du code qui vient d'être téléchargé et libère le bit de pause pour permettre au processeur de commencer à exécuter le code téléchargé.

2.6.3.2 Stockage externe non volatile

Le code du programme d'application peut être stocké sur une mémoire flash externe, une EEPROM ou d'autres formes de mémoire non volatile. Comme avec la plupart des systèmes embarqués, les disques durs sont rarement utilisés dans les dispositifs SoPC plus petits car ils ont une durée de vie plus courte que les autres composants de mémoire non volatile et la plupart des systèmes ne nécessitent pas de très grandes quantités de stockage non volatile. Le programme d'application et le code du système d'exploitation peuvent être préprogrammés dans le module de mémoire externe (pour un cycle de production) ou un programme de démarrage peut être utilisé pour stocker le programme d'application dans une mémoire non volatile. Pour les applications à faible vitesse, le code peut être exécuté directement à partir de la mémoire externe. Cependant, si une fonctionnalité à grande vitesse est requise, un concepteur peut utiliser trois mémoires, comme illustré à la figure 2.9. Dans ce schéma, la mémoire sur puce est initialisée avec un chargeur de démarrage, qui gère le mouvement du programme d'application entre les mémoires. (La mémoire sur puce est remplacée par un programme chargeur de démarrage matériel sur certains systèmes (Bootlaoder), y compris le processeur Nios II.)

La mémoire volatile rapide (c'est-à-dire SDRAM) est utilisée pour stocker le programme d'application pendant l'exécution, tandis que la mémoire non volatile plus lente (c'est-à-dire Flash ou EEPROM) est utilisée pour le stockage permanent du programme d'application. Le chargeur de démarrage peut être modifié pour initialiser le système, récupérer un programme à partir de la mémoire non volatile, le stocker dans la mémoire volatile la plus rapide, puis démarrer son exécution à partir de la mémoire la plus rapide. Ce schéma offre les avantages d'un stockage permanent, d'une exécution rapide et de la possibilité de modifier le programme d'application en cas de besoin. Bien sûr, cela se fait au détriment de la mémoire supplémentaire.

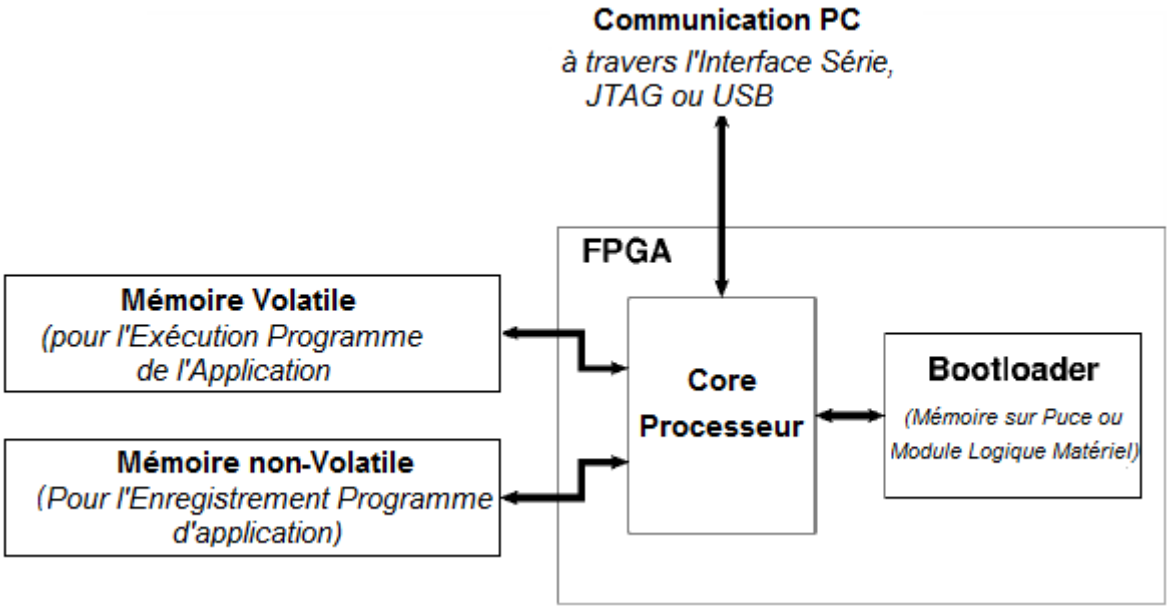


Figure 2. 9 Flexibilité et performance d'un système SoPC par arrangement de mémoires sur puce et mémoires

2.6.4 Cartes de développement SoPC

Pour permettre aux concepteurs d'apprendre le flux d'outils compliqué et aussi un démarrage précoce des projets de co-conception matériel/logiciel, la plupart des fournisseurs de FPGA proposent des cartes de développement SoPC. Ces cartes offrent un grand FPGA avec plusieurs mégaoctets de mémoire externe et une variété de fonctionnalités d'E/S intégrées capables de prendre en charge un cœur de processeur logiciel. Beaucoup incluent également des FPGA avec des cœurs de processeur matériels durs. Alors qu'un PCB personnalisé est en cours de conception pour le nouveau produit basé sur SoPC, le travail peut commencer en parallèle sur les tâches de configuration matérielle et de développement logiciel à l'aide de la carte de développement (en supposant que la carte possède un périphérique FPGA et des fonctionnalités d'E/S similaires).

Deux nouvelles cartes de développement SoPC d'Altera et de Xilinx sont représentées sur la Figure 2..10 Ces cartes prennent toutes deux en charge un large éventail d'interfaces matérielles d'E/S, notamment VGA, audio, PS/2, USB, Ethernet, E/S série, parallèle E/S, écrans LCD, LED, commutateurs et broches d'E/S à usage général supplémentaires sur des en-têtes pouvant être connectés à du matériel utilisateur externe.

La carte Altera DE2 de la Figure 2..10-a prend en charge la conception SoPC à l'aide du processeur à cœur souple Nios II sur la famille Cyclone II FPGA.

La carte de développement Atlys de la figure 2.10-b est une plate-forme de développement de circuits numériques complète et prête à l'emploi basée sur un FPGA Xilinx Spartan-6 LX45,

niveau de vitesse -3. Le grand FPGA et la collection embarquée de périphériques haut de gamme, y compris Gbit Ethernet, HDMI Video, 128 Mo de mémoire DDR2 16 bits et les ports USB et audio font de la carte Atlys un hôte idéal pour une large gamme de systèmes numériques, y compris le processeur intégré conceptions basées sur MicroBlaze de Xilinx . Les FPGA sur les deux cartes contiennent des circuits de multiplication matériels entiers qui peuvent être utilisés pour les applications DSP. Un câble USB connecté à un système de développement PC est généralement utilisé pour télécharger les données de configuration matérielle et les logiciels du FPGA sur les dispositifs de mémoire de la carte. Les outils de développement SoPC prennent également en charge le débogage à distance sur les cartes.

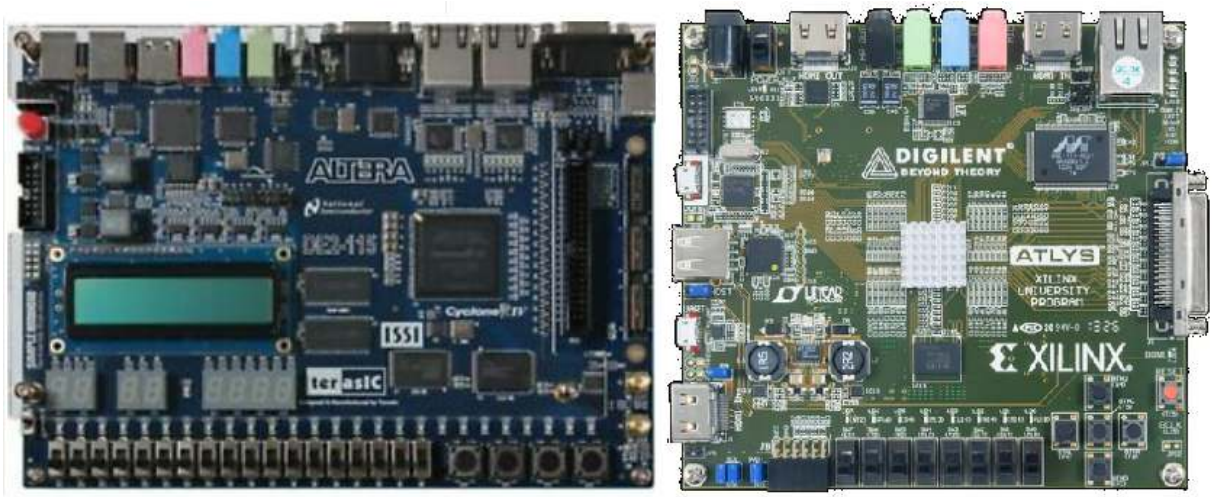


Figure 2. 10 Carte de développement et prototypage a : DE2 d'Intel-Altera b : Atlys de Xilinx

2.6.5 Systèmes d'exploitation embarqués pour les systèmes SoPC

De nombreux systèmes embarqués nécessitent désormais le multitâche, la planification, les threads et parfois la prise en charge de la mise en réseau. Dans de tels cas, un système d'exploitation intégré commercial est généralement utilisé plutôt que de développer un système d'exploitation personnalisé pour des produits individuels. Certains fournisseurs de FPGA fournissent un micronoyau pour leurs dispositifs. Linux, Nucleus PLUS, NORTi, Wind River VxWorks AE X, OSE RTOS et KROS sont disponibles pour les systèmes SoPC d'Altera via des fournisseurs tiers. Linux, QNX Neutrino, Wind River et uC/OS-II RTOS sont disponibles pour les systèmes SoPC de Xilinx [27,32, 33].

Pour les conceptions plus petites qui ne nécessitent pas une prise en charge complète du système d'exploitation, des cœurs IP et des logiciels de prise en charge pour la communication et la mise en réseau de base sont fournis. Des outils tiers sont également disponibles avec une prise en

charge complète de la mise en réseau, notamment des piles de protocoles TCP/IP complètes, des communications USB et Bluetooth.

Les frais de licence et les accords juridiques pour le système d'exploitation, la prise en charge du réseau et d'autres cœurs IP peuvent augmenter le coût global du système SoPC, et ils doivent être soigneusement évalués au début du processus de conception. Les problèmes d'IP ont tendance à être moins impliqués lorsque l'on traite directement avec les principaux fournisseurs de puces FPGA ; Cependant, le coût des cœurs IP peut encore varier considérablement, allant de la gratuité à des dizaines de milliers de dollars.

2.7 Conclusion

Ce chapitre donne un aperçu des systèmes et des conceptions de prototype de circuit sur puce. L'approche SoPC doit être envisagée pour de nombreux systèmes embarqués. Dans de nombreux cas, cela peut réduire le temps et les coûts de développement. Les exceptions sont les systèmes bas de gamme facilement implémentés sur un microcontrôleur mono-puce à faible coût, les circuits produits en très grande quantité, les composants qui nécessitent de très hautes performances et ceux qui nécessitent une très faible consommation d'énergie. Les fournisseurs traditionnels de processeurs et d'ASIC se sont rendu compte de l'avantage de l'approche SoPC utilisant des FPGAs pour une logique utilisateur personnalisé. Plus d'informations sur les dispositifs, outils et systèmes SoPC spécifiques sont disponibles auprès de fabricants tels qu'Altera, Xilinx, Cypress Semiconductor, Stretch Incorporated et Tensilica.

Chapitre 3

Aperçu sur les méthodes de binarisation, implémentations et comparaisons

3.1 Introduction

La problématique de seuillage qui sera discutée s'inscrit dans le contexte général de traitement de documents anciens. Dans cette partie de ce manuscrit, différents types de binarisation cités dans la littérature sont présentées. L'intérêt de cette étude est orienté vers la binarisation des documents en niveaux de gris, puisque les documents en couleurs, peuvent être convertis fidèlement en niveaux de gris.

3.2 Binarisation

La binarisation est une opération de prétraitement d'images qui sert à convertir une image en couleur ou niveau de gris en une image binaire, où l'avant plan est en couleur noir et l'arrière-plan en blanc (ou l'inverse). La binarisation peut se faire par plusieurs techniques, telles que le seuillage, classification, etc. Elle représente donc un outil très utilisé dans la segmentation d'images pour extraire des objets de leurs fonds en fonction d'un certain seuil.

Techniquement parlant, soit une image I de taille (M (hauteur) x N (largeur)), $I(x, y)$ la valeur représentant le niveau de gris du pixel au point coordonnées *(x, y)*.

Étant donné un seuil T :

- Nous définissons les pixels de l'objet, ceux ayant le niveau de gris inférieur à ***T***, et les pixels de fond, ceux ayant le niveau de gris supérieur à ***T***.

- Donc, l'image binarisée g est déterminée comme suit :

$$g(x,y)=\begin{cases}1 & if \quad f(x,y)>T\\ 0 & if \quad f(x,y)\le T\end{cases} \tag{3.1}$$

3.3 Les méthodes de binarisation

Les méthodes de binarisation ont été classées selon les informations qu'elles exploitent, en six catégories [55]

- méthodes basées sur la forme de l'histogramme de l'image,
- méthodes basées sur le clustering,
- méthodes basées sur l'entropie,
- méthodes basées sur les attributs d'objet,
- méthodes spatiales,
- méthodes locales.

Selon plusieurs travaux de recherche [56,57], les techniques de binarisation par seuillage d'images en niveaux de gris peuvent être classées en trois catégories :

- Seuillage global : consistant à partitionner l'image en deux classes grâce à un seuil optimal qui sera calculé à partir d'une mesure globale sur toute l'image. L'histogramme est une mesure utilisée le plus souvent dans les méthodes de seuillage. Dans ce cas, le seuil attendu est celui qui distingue le plus possible les deux classes : fond et objet.
- Seuillage local : où un seuil est calculé localement pour chaque pixel centré dans une fenêtre définie préalablement, se basant sur l'information se trouvant dans son voisinage.
- Seuillage hybrides [58] : consiste à combine des informations globales et locales pour attribuer les pixels à l'une des deux classes.

3.4 Méthodes de binarisation globale

Les méthodes de seuillage globale reposent sur l'exploitation de l'histogramme de toute l'image I . L'histogramme est une courbe monodimensionnelle qui caractérise la distribution des niveaux de gris, il est décrit par la fonction $h(k)$

$$h(k) = \sum_{i=1}^{N} \sum_{j=1}^{M} (I(i,j) == k) \qquad (2)$$

Où :

$h(k)$ étant le nombre de pixels ayant le niveau de gris k, $k \in [0:255]$

$N \times M$ le nombre total de pixels dans l'image.

A partir de l'allure de l'histogramme, on peut déduire le nombre de classes ainsi que la position relative des seuils. L'histogramme est uni-modal, s'il est formé d'un seul pic représentant les pixels de l'objet ou ceux du fond. Il est bimodal s'il est caractérisé par deux modes séparés par une vallée, il indique l'existence d'un objet sur un fond.

L'histogramme multimodal comporte plus de deux modes séparés par des vallées, il indique la présence de plusieurs classes dans l'image (chaque mode correspond à une classe). Les seuils doivent être localisés dans les vallées [58].

En pratique, il est rare de trouver un histogramme qui présente des modes bien distincts. En effet, la plupart des images présentent des histogrammes bruités caractérisés par des modes non discernables.

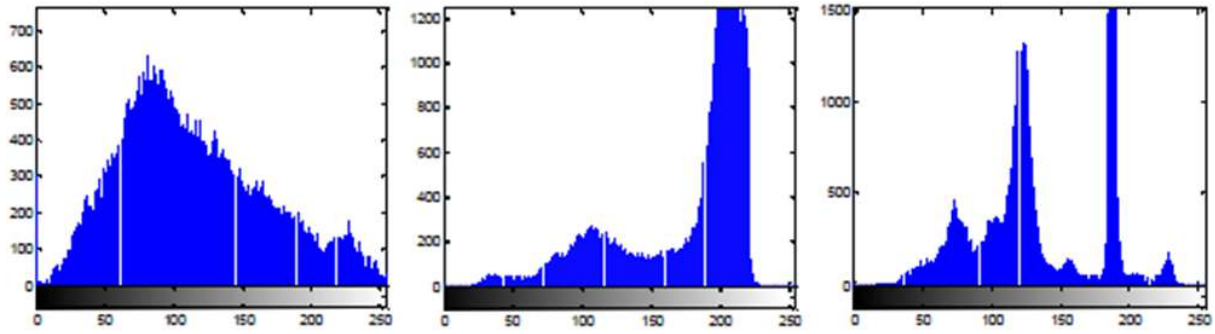


Figure 3. 1 Allures idéales d'un histogramme. (a): unimodal, (b): bimodal, (c) multimodal

L'application d'un seuil particulier T est équivalent à classer chaque pixel comme étant soit une partie de l'arrière-plan (fond) ou au premier plan (objet). Ainsi, l'ensemble de tous les pixels de l'image est divisée en deux ensembles disjoints C_1 et C_2 tel que:

$$C_1 = \{0,1,...T\} \quad et \quad C_2 = \{T+1,...255\} \tag{3}$$

Avant de donner le principe des différents algorithmes de binarisation, passons en revue quelques faits élémentaires au sujet des informations qui peuvent être dérivée de l'histogramme de l'image par les équations suivantes :

- p Représentent la probabilité des niveaux de gris telles que :

$$p(i) = \frac{h(i)}{N} \tag{4}$$

- P Représente la probabilité moyenne de l'image tels que :

$$P(i) = \sum_{i=0}^{k} p(i) \tag{5}$$

- μ Désignent l'écart type de l'image tels que :

$$\mu(i) = \frac{\sum_{i=0}^{k} p(i) \times i}{P(i)} \tag{6}$$

Si un seuillage à un niveau T est réalisé, le nombre de pixel assigné à chaque classe C_1 et C_2 (background et foreground) est:

$$n_1(T) = |C_1| = \sum_{i=0}^{T} h(i) \quad et \quad n_2(T) = |C_2| = \sum_{i=T+1}^{k} h(i) \tag{7}$$

Donc :

$$n_1(T) + n_2(T) = |C_1 \cup C_2| = N \times M \tag{8}$$

La probabilité, la probabilité moyenne ainsi que l'écart type correspondant à chaque classe C_1 ou C_2 peuvent être calculées à partir de l'histogramme pour chaque seuil T où :

μ_1, μ_2 Désignent respectivement l'écart type de chaque classe objet et fond de l'image tels que :

$$\mu_1 = \frac{\sum_{k=0}^{T} p(k)k}{P_1} \tag{9}$$

$$\mu_2 = \frac{\sum_{k=T+1}^{255} p(k)k}{P_2} \tag{10}$$

p_1, p_2 Représentent respectivement les probabilités moyennes de chaque classes objet et fond telles que :

$$P_1 = \sum_{k=0}^{T} p(k) \tag{11}$$

$$P_2 = \sum_{k=T+1}^{255} p(k) \tag{12}$$

p Représentent la probabilité des niveaux de gris telles que :

$$p(i) = \frac{h(i)}{N} \tag{13}$$

3.4.1 Méthode d'Otsu [60]

Elle est considérée comme la méthode de référence dans le domaine du seuillage d'histogrammes. Dans cette méthode, l'opération de seuillage est vue comme une séparation (ou partitionnement) des pixels d'une image en deux classes « fond et objet » à partir d'un seuil T qui minimise la variance intra-classe et maximise la variance interclasse. La classe « fond » regroupe tous les pixels ayant un niveau de gris supérieur au seuil T alors que la classe « objet » contient tous les pixels de niveau de gris inférieur à T .

Soient δ_W^2 la variance d'une intra-classe, δ_B^2 la variance interclasse et δ_T^2 la variance totale telle que :

$$\delta_B^2 = P_1 P_2 (\mu_2 - \mu_1)^2 \tag{14}$$

$$\delta_W^2 = P_1 (i - \mu)^2 \tag{15}$$

$$\delta_T^2 = \delta_W^2 + \delta_B^2 = \sum_{i=1}^{T} P_1 (i - \mu_1)^2 + \sum_{i=T+1}^{255} P_2 (i - \mu_2)^2 \tag{16}$$

Dans la littérature la variance interclasse est la plus utilisée et cela pour des raisons de simplicité. Le seuil optimal T est celui qui maximise cette variance :

$$T = \arg \max_{0 \le T \le L-1} \delta_B^2 = \arg \max_{0 \le T \le L-1} \left(P_1 P_2 (\mu_2 - \mu_1)^2 \right) \tag{17}$$

3.4.2 Méthode de kitller

La méthode J.Kittler et J.Illingworth [61], considère que l'image a un histogramme bimodale de niveaux de gris qui représente la densité de probabilité $p(i)$ d'un mélange de populations formées des niveaux de gris des deux classe objets et du fond, formons une distribution normale avec un écart type μ , une variance σ. Cet algorithme permettra de repérer le seuil optimal qui minimise la probabilité d'erreur de classification. Donc le seuil optimal T est le niveau de gris qui minimise le critère J (t) suivant :

$$J(t) = 1 + 2\left[P_1(t)\log(\sigma_1(t)) + P_2(t)\log(\sigma_2(t))\right] - 2\left[P_1(t)\log(P_1(t)) + P_2(t)\log(P_2(t))\right] \tag{18}$$

3.4.3 Méthode d'ISODATA

Le binarisation par ISODATA [62] consiste à trouver un seuil en séparant l'histogramme en deux classes itérativement avec la connaissance a priori des valeurs associées à chaque classe. Cette méthode commence par la division de l'intervalle de valeurs non-nulles de l'histogramme en deux parties équidistantes, et ensuite prendre p_1, p_2 comme les probabilités moyennes arithmétiques des deux classes. Répéter jusqu'à la convergence, le calcul du seuil optimal T comme l'entier le plus proche de $\frac{P_1 + P_2}{2}$, et mettre à jour les deux probabilité moyennes P_1, P_2.

Cette méthode sera plus présentée et fera l'objet d'une implémentation matérielle.

3.4.4 Méthode de Pun

Pun [63] a proposé que la maximisation de l'entropie est un critère optimal pour le binarisation de l'image. Le seuil optimal T est choisi comme le niveau de gris qui maximise la somme de l'entropie de l'objet H_1 et l'entropie du fond H_2.

$$S_{optimal} = \max_t \left(H_1(t) + H_2(t)\right) = \max_t \left(-\sum_{i=0}^{t} \left(p(i) \times \log p(i)\right) - \sum_{i=t+1}^{255} \left(p(i) \times \log p(i)\right) \right) \quad (19)$$

3.4.5 Méthode de KAPUR

La méthode proposée par Kapur et al. [64] c'est une extension de la méthode de Pun qui est basée sur l'entropie qui considère le premier plan et l'arrière-plan de l'image comme deux sources de signal différentes. A la déférence de cette dernière, la méthode de Kapur prend en compte la distribution de probabilité moyenne de l'objet et du fond dans la détermination de l'entropie de division. Ainsi, le seuil optimal T est lorsque la somme des entropies des deux classes atteint son maximum.

$$S_{optimal} = \max_t \left(H_1(t) + H_2(t)\right) = \max_t \left(-\sum_{i=0}^{t} \left(\frac{p(i)}{P_1} \times \log\left(\frac{p(i)}{P_1} \right) \right) - \sum_{i=t+1}^{255} \left(\frac{p(i)}{P_2} \times \log\left(\frac{p(i)}{P_2} \right) \right) \right) \quad (20)$$

3.4.6 Méthode de Cheng et Chen

La méthode de Cheng et Chen [65] est basée sur le principe la partition floue, le choix du seuil se fait par la maximisation de l'entropie. Considérons les deux ensembles flous objet et fond ou les fonctions d'appartenance sont définis par

$$\mu_{objet} = \begin{cases} 1 & x \le a \\ \dfrac{x-c}{a-c} & a < x < c \\ 0 & x \ge c \end{cases} \qquad \mu_{fond} = \begin{cases} 0 & x \le a \\ \dfrac{x-a}{c-a} & a < x < c \\ 1 & x \ge c \end{cases} \tag{21}$$

Dans cette méthode, le seuil de binarisation optimal T est choisi comme le niveau de gris dont la fonction d'appartenance égale 0.5 et donc c'est le centre de l'intervalle $\left[a_{opt}, c_{opt}\right]$, tel que a_{opt}, c_{opt} sont les valeurs de a, c en maximisant l'entropie de la division donnée par :

$$H = -\left(P_1 \times \log P_1 + P_2 \times \log P_2\right) \tag{22}$$

$$P_1 = \sum_{i=0}^{L-1} \mu_1(i) \times p(i) \quad , \quad P_2 = \sum_{i=0}^{L-1} \mu_2(i) \times p(i) \tag{23}$$

3.4.7 Méthode de Li-Lee

Li et Lee [66] ont proposé une méthode de seuillage d'images où le regroupement des niveaux de gris en deux classes (objet et fond) est fondé sur la minimisation de l'entropie croisée. Le seuil optimal T est calculé de façon à minimiser la somme de l'entropie croisée de l'objet et du fond où:

$$\eta(t) = \eta_1(t) + \eta_2(t) = \sum_{i=0}^{t} i \times h_i \times \log\left(\frac{i}{\mu_1(t)}\right) + \sum_{i=t+1}^{255} i \times h_i \times \log\left(\frac{i}{\mu_2(t)}\right) \tag{24}$$

3.5 Quelques méthodes de binarisation locale

Il existe des images pour lesquelles il est difficile d'effectuer une bonne binarisation au moyen d'un seul seuil, si celui-ci est le même pour chaque point de l'image. Ceci peut être dû à une non uniformité de l'éclairage de l'image ou au fait que les différents objets qui composent l'image ont des dynamiques de luminances différentes. Pour cela, les méthodes de binarisation locales sont mieux adaptées.

A l'inverse des méthodes globales qui ne considèrent que la valeur du pixel, les méthodes locales prennent en considération la valeur des pixels voisins pour le calcul des seuils. Deux stratégies sont souvent employées. La première consiste à diviser, en premier lieu, l'image en fenêtres non chevauchantes. Dans un deuxième temps, un seuil est déterminé dans chaque fenêtre selon la nature de l'histogramme local. Un seuil égal à zéro est affecté à la fenêtre possédant un

histogramme uni-modal tandis qu'une procédure de seuillage est appliqué pour déterminer un seul seuil dans le cas d'un histogramme bimodal ou multimodal. La figure suivante illustre le principe du seuillage local.

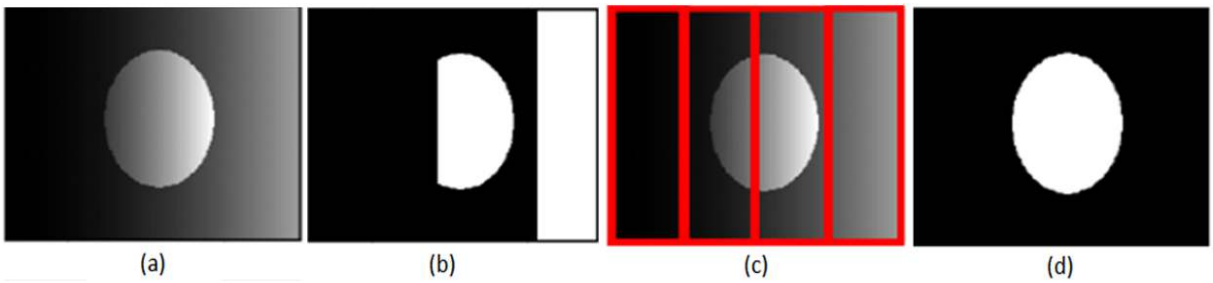


Figure 3. 2 Illustration du seuillage adaptatif. (a) Image originale avec un fond non uniforme, (b) Image segmentée avec un seuil global, (c) Image originale découpée en sous images, (d) Image segmentée avec un seuillage adaptatif.

3.5.1 Méthode de Bernsen

Bernsen propose que le seuil local soit calculé comme la moyenne des intensités de niveau de gris minimale $I_{\min}(i,j)$ et maximale $I_{\max}(i,j)$ dans la fenêtre carré w centré sur le pixel $I(i,j)$. Ainsi pour chaque pixel de coordonnés (x, y), le seuil local est donné par [67]

$$T_w(i,j) = \frac{I_{\max}(i,j) + I_{\min}(i,j)}{2} \tag{25}$$

Avec un contraste calculé par :

$$C_w(i,j) = I_{\max}(i,j) - I_{\min}(i,j) \tag{26}$$

Cependant si le mesure du contraste $C_w(i,j)$ est inférieur à un certain seuil S, alors le voisinage consiste en une seule classe: fond ou objet. La méthode de Bernsen est très sensible au bruit du fond. A cause de la prise en compte du maximum et du minimum uniquement, dans le cas où la fenêtre est uniquement sur du fond, le bruit sera interprété comme objet, car le seuil sera bas.

3.5.2 Méthode de Niblack

L'algorithme de Niblack [68] est une amélioration de la méthode de Bernsen qui calcule un seuil local pour chaque pixel $I(i,j)$. Le seuil T est calculé en fonction de la moyenne m et de l'écart-type σ de tous les pixels de la fenêtre carré w centrée sur le pixel de cordonné (i,j). Ainsi le seuil $T_w(i,j)$ est donné par :

$$T_w(i,i) = m + k \times \sigma_w(i,j) \tag{27}$$

$$T_w(i,j) = m + k \times \sqrt{\frac{\sum (I_w(i,j) - m)}{NP}} \tag{28}$$

Le paramètre k est utilisé pour déterminer le nombre des pixels des contours considérés comme des pixels d'objet et il prend des valeurs négatives, k est fixé à (-0.2) par les auteurs et NP représente les nombre des pixels de la fenêtre carré w et m est la valeur moyenne des pixels de la fenêtre carré w. Cette méthode donne de bons résultats car le seuil dépond du pixel et de l'information extraite à partir de son voisinage, mais n'est pas efficace quand le fon n'est pas uniforme.

3.5.3 Méthode de Sauvola

L'algorithme de Sauvola est une amélioration de celui de Niblack [69], ou il insère un paramètre R qui est le rang dynamique de l'écart type de la valeur de gris de l'image, et cela pour donner plus de performance pour les documents avec un fond uniformes et aussi pour avoir augmenté la performance pour les documents avec un fond contenant des textures claire ou bien une illumination inégal [69]. Dans la modification de Sauvola le seuil de binarisation est donnée par:

$$T_w(i,j) = m \times \left[1 + k \times \left(\frac{\sigma_w(i,j)}{R} - 1 \right) \right] \tag{29}$$

Le paramètre R est le rang dynamique de l'écart-type σ et le paramètre k prend des valeurs positives dans l'intervalle $[0.2,0.5]$. Dans la littérature on prend R =128 et $k = 0.5$. Dans [71], la méthode de Sauvola se montre plus efficace que la celle de Niblack dans le cas où le niveau de gris de l'objet est proche du niveau de gris 0, et celui du fond est proche du niveau de gris 255. Cependant dans les images où les niveaux de gris des pixels du fond et de l'objet sont proches, les résultats sont peu satisfaisants.

3.5.4 Méthode de Wolf

Pour résoudre les problèmes rencontrés avec l'algorithme de Sauvola en présence de document de faible contraste, Wolf et al [70]. Proposent de normaliser le contraste et la moyenne de niveaux de gris de l'image. Ainsi le seuil ce calcule par :

$$T_w(i,j) = (1-k)\times m + (k\times M) + k\times\frac{\sigma_w(i,j)}{R}\times(m-M) \quad (30)$$

Wolf propose de fixé $k = 0.5$ et M c'est le niveau de gris minimum de toute l'image et R c'est l'écart-type maximum obtenu sur toutes les fenêtres. Cette méthode surpasse dans la plupart des cas les méthodes de Niblack et de Sauvola. Cependant, si les valeurs de gris d'arrière-plan sont très différentes alors une dégradation de ses performances est observée. Cela est dû au fait que les valeurs des paramètres M, R sont calculées à partir de l'image global.

3.5.5 La méthode Nick

L'avantage majeur de cette méthode est qu'elle améliore considérablement la binarisation des images de page « blanches » et claires [71], et dans le cas où l'image présente de faible contraste, en déplaçant vers le bas, le seuil de binarisation

Le calcul du seuil est réalisé comme suit :

$$T_w = m + k\sqrt{\frac{(\sum I_w(i,j)^2 - m^2)}{NP}} \quad (31)$$

tel que :

k est le facteur de Niblack est il varie entre $[-0.1,-0.2]$ selon les besoins de l'application

3.6 Méthode hybride

Certaines méthodes adaptatives de binarisation de document [72-76] ont été rapportées dans la littérature de binarisation des documents ancien. En particulier, une approche adaptative de binarisation de document qui consiste à l'estimation du seuil de binarisation on se basant sur l'estimation de l'arrière-plan du document. Par exemple, Gatos et al. [74] utilise la méthode de binarisation de Sauvola [73] pour générer l'image binaire du document pour l'estimation de l'arrière-plan. Moghaddam et al. [77] utilise une approche adaptative et itérative de calcule de moyenne des pixels de l'arrière-plan. De plus, certaines méthodes ce base sur la détection du contour. Par exemple, Chen et al. [76] propose une méthode qui nous donne l'image binaire du document on se basant sur les informations issu de la détection de contour. Moghaddam et al. [81] Au lieu de cela, utilise la localisation de la région du texte puis l'estimation du seuil de l'image localement .Su et al. [76] essayez également de localiser le contour en utilisant un contraste d'image qui est évalué en fonction du maximum et minimum local.

Les méthodes de binarisation hybride traitent les images des documents anciens en niveaux de gris, considèrent que les caractéristiques spécifiques d'un document tel que le texte ou les graphiques sont placés sur le premier plan et le fond comme deuxième plan. Les images d'entrée sont décrites par l'équation :

$$I(i, j) = r, r \in [0,1] \tag{32}$$

Où x et y sont les coordonnées horizontale et verticale d'un pixel dans une image, et r peut prendre n'importe quelle valeur entre 0 et 1, la valeur 0 représentant un pixel noir et 1 la valeur d'un pixel blanc. L'objectif est la transformation de tous les intermédiaires de pixels évalués pour finalement trouver sur la valeur de fond, le pixel vaut 1 ou, sur le premier plan, la valeur de pixel vaut 0. Ainsi, une technique globale peut être utilisée. En plus de cela, une méthode pour résoudre ces problèmes de documents anciens doit être proposée qui conduit à l'approche hybride.

3.6.1 Algorithme de GATOS

Le principe de cette méthode est de Chercher à estimer le fond d'une image, pour ensuite faire un seuillage sur la différence entre le fond et l'image d'origine, en faisant une combinaison des différentes méthodes ayant des propriétés complémentaires pour réaliser une binarisation automatique capable de traiter des documents dégradés [72]. Il décrit sa méthode comme une succession de cinq phases.

- La première améliore la qualité globale du document.
- L'étape suivante consiste à estimer la valeur du fond de l'image. Cette opération est réalisée en deux phases.
- Une première estimation de l'avant plan est effectuée en utilisant l'algorithme de Sauvola.
- Une fois les régions détectées, les pixels objet sont remplacés par une interpolation avec les pixels voisins.

$$B(x,y) = \begin{cases} I(x;y) & si \quad S(x,y) = 0 \\ \dfrac{\sum_{x-dxy-dy}^{x+dxy+dy} \sum (I(i_x, i_y)(1 - S(i_x, i_y))}{\sum_{x-dxy-dy}^{x+dxy+dy} \sum (1 - S(i_x, i_y))} & si \quad S(x,y) = 1 \end{cases} \tag{33}$$

Avec :

$B(x,y)$: Image du fond estimée.

$I(x,y)$: Image d'origine après le filtrage de Wiener.

$S(x,y)$: Image binaire obtenue à partir de la segmentation de Sauvola.

A la fin, en comparant l'image filtrée avec l'image du fond pour avoir un seuillage final, Les zones de texte sont détectées si la distance entre l'image filtrée et l'image de fond est supérieure à un seuil d.

Ce seuil d doit varier en fonction de l'image du fond, pour que dans le cas d'un fond sombre, le seuil soit fixé plus bas pour pouvoir séparer le texte du fond. Cette opération est donnée par la formule suivante [72] :

$$T(x,y) = \begin{cases} 1 & Si \quad B(x,y) - I(x,y) > d(B(x,y) \\ 0 & Sinon \end{cases} \tag{34}$$

Avec :

$T(x,y)$: Image binaire finale.

d : Fonction de seuillage.

Le seuil d est donc fonction de la distance moyenne δ entre l'image du fond et l'image filtrée. Dans une première version d= k * δ. Cependant, dans le cas d'une image sombre, le seuil k (qui est fixé à 0,8) n'est plus bon. Pour que la méthode soit correcte dans le cas oùl'illumination du document ne serait pas constante, cette constante est adaptée en fonction de l'intensité moyenne du fond. Pour cela une fonction sigmoïde permettant d'adapter en douceur la constante est utilisée. Finalement, d est calculé ainsi :

$$d(B(x,y)) = q \times \delta \times \left(\frac{(1-p_2)}{1+\exp\left(\frac{-4 \times B(x,y)}{b \times (1-p_1)} + \frac{2 \times (1+p_1)}{1-p_1} \right)} \right) \tag{35}$$

$d(B(x,y))$: Seuillage final en fonction de l'image binaire.

$q, p_1 \; et \; p_2$: Seuillage final en fonction de l'image binaire.

$b(x)$: moyenne de la couleur du fond dans une fenêtre entourant le point (x,y).

Les résultats obtenus par cette méthode sont très intéressants. D'où la distance entre le texte réel et les caractères reconnus est difficile à détecter comparativement avec les méthodes (Otsu, Niblack, Sauvola et al, Kim et al). Cette méthode, bien que performante, ne permet pas de binariser un texte blanc sur un fond noir.

3.6.2 Algorithme de Lu

L'algorithme de Lu [76] est classé parmi les premières méthodes en compétition DIBCO 2009, il est constitué de plusieurs parties pour l'extraction du texte à partir du fond

- L'estimation de l'arrière-plan est réalisée par encliquetage de polynôme sur l'ensemble du document. Cela se fait par l'échantillonnage de chaque ligne de l'image et la création d'une courbe sur la base des valeurs d'intensité de cette ligne.
- Ensuite, un ajustement polynomial est appliqué, pour reconstruire la courbe des intensités de la ligne échantillonnée. En faisant cela, nous pouvons mieux isoler les pixels qui sont plus sombres que le fond, dans chaque ligne. Nous faisons l'estimation polynômiale horizontale et verticale.
- Une fois avoir une bonne approximation de l'arrière-plan, l'étape suivante de l'algorithme, est la compensation du contraste de l'image en utilisant l'image à partir de l'approximation du fond avec la formule :

$$I' = \frac{C}{BG} \times I \tag{36}$$

Avec I qui représente l'image originale, C le contraste de l'image, BG la valeur approchée du fond, I' est la nouvelle image.

Cette étape nous donnera une image avec un fond plus clair que l'image originale. Une approche locale est utilisée par une fenêtre glissante. Ceci est accompli en regardant les pixels voisins d'une fenêtre et la comparaison au centre de la fenêtre avec les pixels de contour de la fenêtre.

Si le nombre de pixels du contour de la fenêtre est supérieur à un nombre minimum et si l'intensité du pixel central est plus petite que la moyenne des pixels de contour d'intensité à l'intérieur de la fenêtre, alors le pixel central est considéré comme texte. La taille de la fenêtre est déterminée en évaluant la largeur du texte par le biais d'une mesure de distance entre les pixels de contour et choisissant la distance la plus fréquente dans l'histogramme. L'étape finale de cet algorithme est d'avoir l'image binarisée en utilisant des opérateurs morpho logiques.

3.6.3 Méthode de SU Boland [78]

Le principe de cette méthode est déterminé par la soustraction des images calculées par la dilatation et l'érosion des opérateurs morphologiques, qui donne des contours autours des textes comme résultat. Afin de minimiser les dégradations du fond, cet algorithme utilise la normalisation qui est calculée par la somme des résultats de dilatation et l'érosion plus un petit constant. Cette méthode décrite, évalue et classe la tache de binarisation en cinq étapes comme suit :

- Le lissage polynomial de l'image.
- Estimation du fond du document.
- Détection du texte en utilisant les contours actifs.
- Estimation de seuil local.
- Méthodes basées sur le post-traitement.

L'image est ensuite seuillée en utilisant une fenêtre glissante d'une manière similaire à l'algorithme de Lu [73]. Si le niveau de gris d'un pixel est inférieur à la moyenne des niveaux de gris des pixels du contour (fenêtre de voisinage), ou le nombre des pixels de contour doit être supérieur à un nombre minimum donné, ce pixel est classé comme négatif.

3.6.4 Algorithme de R.Hedjam et M.Cheriet [81]

Le procédé proposé par cette méthode est constitué de deux principales étapes :

- la première étape consiste à binariser l'image du document en éliminant les informations de dégradation et de ne garder que la partie noire du texte. Dans ce cas, les motifs d'interférence, le bruit, et les parties avec l'intensité faibles seront effacées. En utilisant la méthode de binarisation Sauvola pour produire une image binarisée avec un seuil optimum k.
- La deuxième étape récupère les parties de texte manquantes (perdus dans l'étape précédente) en utilisant une binarisation locale basée sur la classification des maximums de vraisemblances.

Cela nous permet de prendre deux modèles gaussiens, l'un pour le fond et l'autre pour le texte :

$$\begin{cases} P(u(x)|H_0) = P(u(x)|text) = \dfrac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(\dfrac{-(u(x)-u_t)^2}{2\sigma_t^2} \right) \\ et \\ P(u(x)|H_1) = P(u(x)|fond) = \dfrac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(\dfrac{-(u(x)-u_b)^2}{2\sigma_b^2} \right) \end{cases} \quad (37)$$

u_w et σ_w représentent la moyenne et la variance respectivement avec $w \in \{t,b\}, u(x)$ représente le niveau de gris d'un pixel. A la fin il faut résoudre le problème de classification équivalent au problème de minimisation suivant :

$$u_{fond}(x) = \arg\min_{w\in\{t,b\}} \left\{ \frac{(u(x)\_u_w)^2}{2\sigma_w^2} \right\} \quad (38)$$

Ce modèle à donné des bons résultats pour la restauration des documents anciens. Mais elle ne parvient pas à récupérer les petites boucles en caractères, sur tout quand ils appartiennent à un fond très dégradé.

3.7 Implémentation d'algorithmes de calcul du seuil de binarisation

Le Seuillage automatique est un domaine très ancien, mais toujours très actif dans la recherche. De nombreuses techniques ont été développées pour cette tâche, allant de solutions ad hoc simples à des algorithmes complexes avec des fondations théoriques solides, tel que documenté dans plusieurs revues et études d'évaluation dans la littérature.

La binarisation d'images est également considérée comme une technique de «segmentation par seuillage», elle est donc souvent classée sous ce terme. Dans cette partie nous allons donner quelques implémentations des méthodes de base, les plus utilisées pour le calcul du seuil, la majorité de ces méthodes utilise la notion de distribution de probabilité et histogramme. Pour cela, nous commençons par les méthodes simples, puis par les méthodes de seuillage globales et enfin les méthodes locales.

3.7.1 Implémentation d'une méthode simple de calcul du seuil

La valeur de seuil ne doit pas être fixée d'une certaine manière, mais sur la base du contenu de l'image. Dans le cas le plus simple, nous pourrons utiliser comme seuil de binarisation la moyenne ou la médiane de tous les pixels de l'image, ou même aussi la moyenne entre la valeur maximum et minimum des pixels.

Comme valeur du seuil la valeur de la moyenne :

$$T = mean(I) \tag{39}$$

Comme valeur du seuil la valeur de la médiane :

$$T = median(I) \tag{40}$$

Comme valeur du seuil la moyenne entre la valeur max et min :

$$T = \frac{Max(I) + Min(I)}{2} \tag{41}$$

1: ***Simple_Threshold*** (I, T)

Input: I --Image original.

Return: T -- La valeur du seuil optimal.

2: $I_{\min} \leftarrow Min(I)$ -- Valeur minimal d'intensité du pixel

3: $I_{\max} \leftarrow Max(I)$ -- Valeur maximal d'intensité du pixel

4: $T \leftarrow \frac{I_{\min} + I_{\max}}{2}$ -- Calcule du seuil

5: *Return* T .

Algorithme 1 : Calcule simple d'un seuil de binarisation

3.7.2 Implémentation méthode globale de calcul du seuil

L'approche globale du calcule du seuil est calculé à partir d'une mesure globale sur toute l'image (par exemple, l'histogramme). Dans cette partie on va présenter deux implémentations d'algorithme de seuillage global.

L'algorithme d'Otsu suppose que l'image originale contient des pixels classés en deux, dont les distributions d'intensité sont inconnues. L'objectif est de trouver un seuil T tel que l'arrière-plan et l'avant plan sont au maximum séparés, ce qui signifie d'après l'histogramme qu'ils sont :

- Chacune soit aussi étroite que possible (avoir des écarts minimes)
- Leurs centres (moyens) sont les plus éloigné.

L'algorithme suivant donne une implémentation de l'algorithme d'Otsu pour le calcul du seuil par la maximisation de la variance interclasse, il suppose que l'image est en niveaux de gris K avec un total de N pixels.

1:***OTSU_Threshold*** (h)

Input : h --Histogram.

Return: *T* **--** La valeur du seuil optimal

2: $k \leftarrow Size\ (h)$ -- Nombre des niveaux d'intensité

3 : $N \leftarrow Sum\ (h)$ --Nombre de pixel

4 : $T \leftarrow 0,\ \sigma^2_{b\max} \leftarrow 0$

5 : *for* $q \leftarrow 0 .. k$ *do*

6 : $P_1 \leftarrow 0, P_2 \leftarrow 0,\ \mu_1 \leftarrow 0, \mu_2 \leftarrow 0$

7: *for* $i \leftarrow 0 .. q$ *do*

8: $P_1 \leftarrow P_1 + \frac{h(i)}{N}$ -- Calcule de la probabilité moyenne de la classe objet

9: *End for*

10: *for* $i \leftarrow q+1 .. k$ *do*

11: $P_2 \leftarrow P_2 + \frac{h(i)}{N}$ -- Calcule de la probabilité moyenne de la classe fond

12: *End for*

13: *for* $i \leftarrow 0 .. q$ *do*

14: $\mu_1 \leftarrow \mu_1 + \left(\frac{h(i)}{N}\right) \times k$ -- Calcule de l'écart type de la classe objet

15: *End for*

16 $\mu_1 \leftarrow \frac{\mu_1}{P_1}$

17: *for* $i \leftarrow q+1 .. k$ *do*

18: $\mu_2 \leftarrow \mu_2 + \left(\frac{h(i)}{N}\right) \times k$ -- Calcule de l'écart type de la classe fond

19: *End for*

20: $\mu_2 \leftarrow \frac{\mu_2}{P_2}$

20: $\sigma^2{}_b \leftarrow P_1 \times P_2 \times (\mu_1 - \mu_2)^2$

21: *if* $\sigma^2{}_b > \sigma^2{}_{b\max}$ *do*

22: $\sigma^2{}_{b\max} \leftarrow \sigma^2{}_b$ -- Maximisation de la variance pour trouver le seuil

23: $T \leftarrow q$

24: *end if*

25: *End for*

26: *Return* T

Algorithme 2 : Calcule du seuil de binarisation par la méthode d'OTSU

Pour la méthode d'ISODATA elle détermine l'intervalle [min, max] des valeurs non nulles de l'histogramme puis elle fait une estimation initiale du seuil, (en prenant la moyenne ou la médiane de l'image entière). Cela divise l'ensemble de pixels en deux classes (objet et fond). Ensuite, la moyenne de deux classe est calculée et le seuil est repositionné centré entre les deux moyennes jusqu'au moment où l'algorithme converge.

1 : ***ISODATA_Threshold*** (h)

Input: h --Histogram

Return *:* T --La valeur du seuil optimal

2 : $k \leftarrow Size\ (h)$ -- Nombre des niveaux d'intensité

3 : $N \leftarrow Sum\ (h)$ -- Nombre de pixel

4 : $T_0 \leftarrow 0,\ T_1 \leftarrow 0,\ m_1 \leftarrow 0, m_2 \leftarrow 0$

5 : $T_0 \leftarrow round\left(\frac{Sum(h(1:k)\times(1:k))}{N}\right)$ --Calculé l'intensité moyenne de l'image

6: $m_1 \leftarrow \frac{Sum(h(1:T_0)\times(1:T_0))}{Sum(h(1:T_0))}$ -- Calculé la moyenne en dessous du seuil

7 : $m_2 \leftarrow \frac{Sum(h(T_0+1:256)\times(T_0+1:256))}{Sum(h(T_0+1:256))}$ -- Calculé la moyenne en dessus du seuil

8 : $T_1 = round\left(\frac{m_1+m_2}{2}\right)$ -- Calculé le nouveau seuil

9 : *While* $(|T_1 - T_0| \geq 1)$ do

10 : $T_0 \leftarrow T_1$

11 : $m_1 \leftarrow \frac{Sum(h(1:T_0)\times(1:T_0))}{Sum(h(1:T_0))}$

12 : $m_2 \leftarrow \frac{Sum(h(T_0+1:256)\times(T_0+1:256))}{Sum(h(T_0+1:256))}$

-- Répété la même procédure pour les autres itérations a une condition que la différence entre le nouveau et l'ancien seuil soit supérieure à 1

13 : $T_1 = round\left(\frac{m_1+m_2}{2}\right)$

14 : *End While*

15 : $T \leftarrow T_1$

16 : Return T

Algorithme 3: Calcule du seuil de binarisation par la méthode D'ISODATA

3.7.3 Implémentation d'une méthode locale de calcul du seuil

Dans beaucoup de situation, un seul seuil fixe pour tout l'image ne peut être approprié pour classifier les pixels de toute l'image, surtout lorsque l'image présente des zones avec des arrière-plans non uniformément éclairées. La solution est de recourir à chercher des seuils locaux adaptatifs [82]. Pour cela nous trouvons dans la littérature plusieurs méthodes de seuillage locale.

La méthode de Bernsen [67] c'est une méthode locale adaptative dont le seuil est calculé pour chaque pixel de l'image. Ainsi le seuil est le centre entre le niveau de gris le plus haut et le plus bas dans une fenêtre carré de taille w centré pour chaque pixel de coordonnés *(x, y)*.

1: ***Bernsen_Threshold*** (I, w)

Input*:* I, w -- Image originale, *w* fenêtre carrée

Return*:* T -- Valeurs du seuil pour chaque pixel de l'image

2: $(M, N) \leftarrow$ *Size* (I) -- Taille de l'image

3: *Create map Q : M × N → w*

4: $n \leftarrow 0$

5: *for all pixell (i, j)* do

6: $n \leftarrow n + 1$

7: $I_w \leftarrow map(I, w)$ -- Les pixels de la fenêtre centré sur le pixel *I (i,j)*

8: $I_{\min} = \min I_w$ -- Valeur minimal de dans la fenêtre *w*

7: $I_{\max} = \max I_w$ -- Valeur maximal de dans la fenêtre *w*

9: $T_n = \left(\frac{I_{\max} + I_{\min}}{2} \right)$

10: *End for*

11: $T \leftarrow T_n$

12: *return* T

Algorithme 4: Calcule du seuil de binarisation par la méthode Bernsen

La méthode de Niblack calcule le seuil localement pour chaque pixel en glissant la fenêtre rectangulaire de taille w sur toute l'image. Le seuil T est calculé en utilisant la moyenne et l'écart-type de tous les pixels dans la fenêtre (voisinage du pixel en question).

1: **Niblack_Threshold** (I, w, k)

Input: I, w, k -- Image originale, *w* taille fenêtre carrée, *k* paramètre fixé à -0,2

Return: T -- Valeurs du seuil pour chaque pixel de l'image

2: $(M, N) \leftarrow Size(I)$ -- Taille de l'image

3: *Create map Q : M × N → w*

4: $n \leftarrow 0$

5: *for all pixeII (i, j)* do

6: $n \leftarrow n + 1$

7: $I_w \leftarrow map(I, w)$ -- Les pixels de la fenêtre centré sur le pixel I (i,j)

8: $T_n \leftarrow m + k \times \sigma$ -- m c'est la moyenne des pixels et σ c'est écart type dans la fenêtre *w*

9 : $T_n \leftarrow mean(I_w) + k \times \sqrt{mean((I_w - mean(I_w))^2)}$

10: *End for*

11: $T \leftarrow T_n$

12: *return* T

Algorithme 5: Calcule du seuil de binarisation par la méthode Niblack

3.8 **Résultats de binarisation sur quelques images de vieux documents**

Dans cette partie, nous allons présenter quelques résultats en appliquant les différents algorithmes de binarisation (Global, Local) implémentés en Matlab.

Le choix c'est porté sur plusieurs images de la base des données DIBCO [83] avec ses versions du 2009 jusqu'à 2014. Nous nous contentons de présenter les tests surtout sur les types d'images qui présentent :

- Des dégradations dues à un éclairage non uniforme
- Des taches dues à un vieillissement.

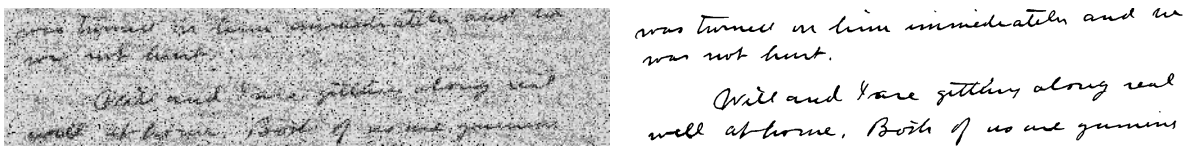
- Transparence de l'arrière-plan

(a) (b)

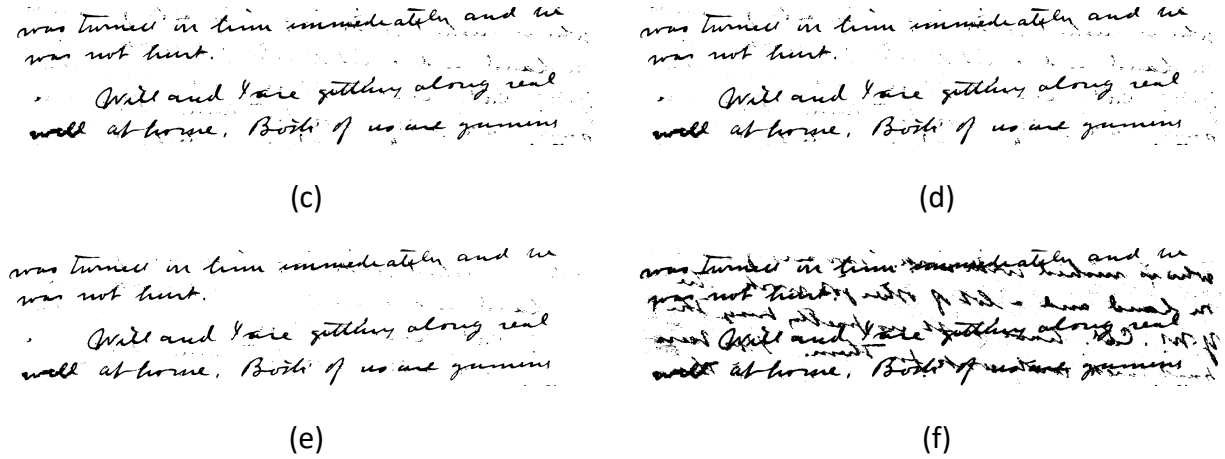
(c) (d)

(e) (f)

Figure 3. 3 Résultat de binarisation sur la base de données DIBCO 2009 [83]. (a) image original, (b) image de référence, (c) méthode d'OTSU, (d) méthode d'ISODATA, (e) méthode de Bernsen, (f) méthode de Niblack



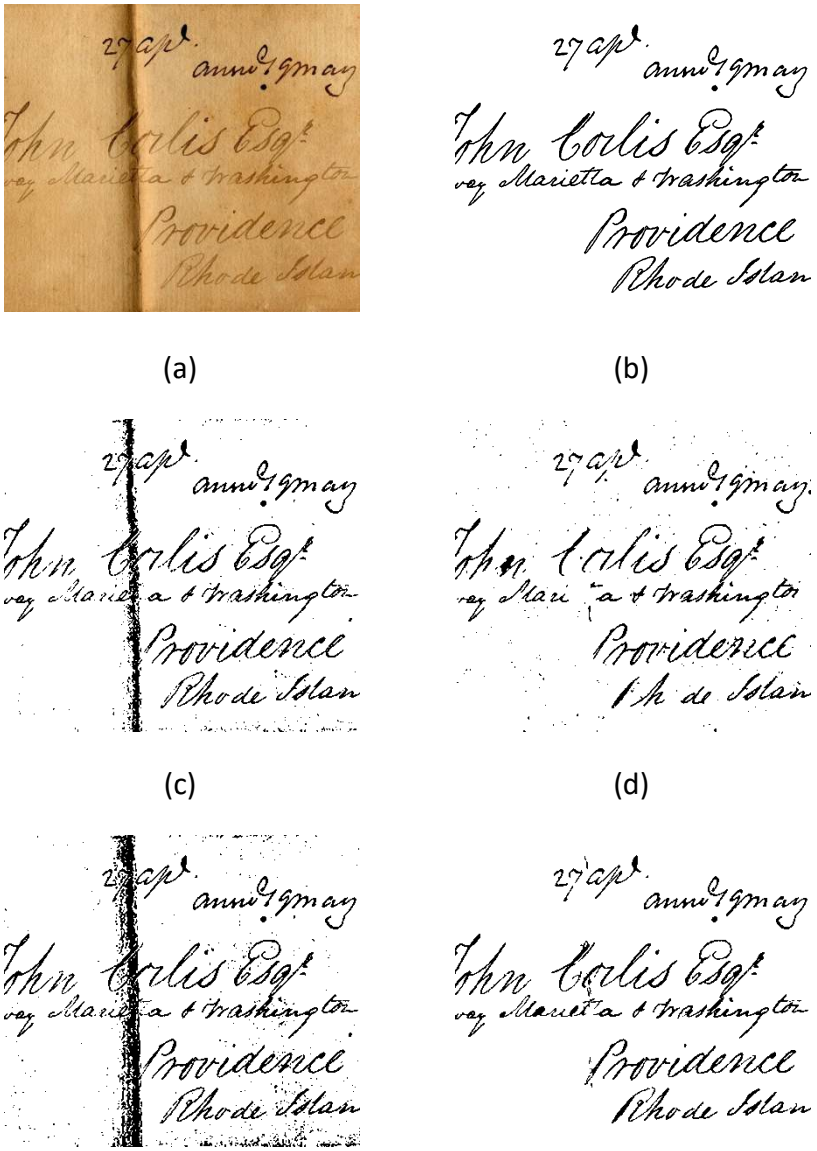
(a) (b)



(c) (d)

(e) (f)

Figure 3. 4 Résultat sur la base de données DIBCO 2010. [83]. (a) image original, (b) image de référence, (c) méthode d'OTSU, (d) méthode d'ISODATA, (e) méthode de Bernsen, (f) méthode de Niblack

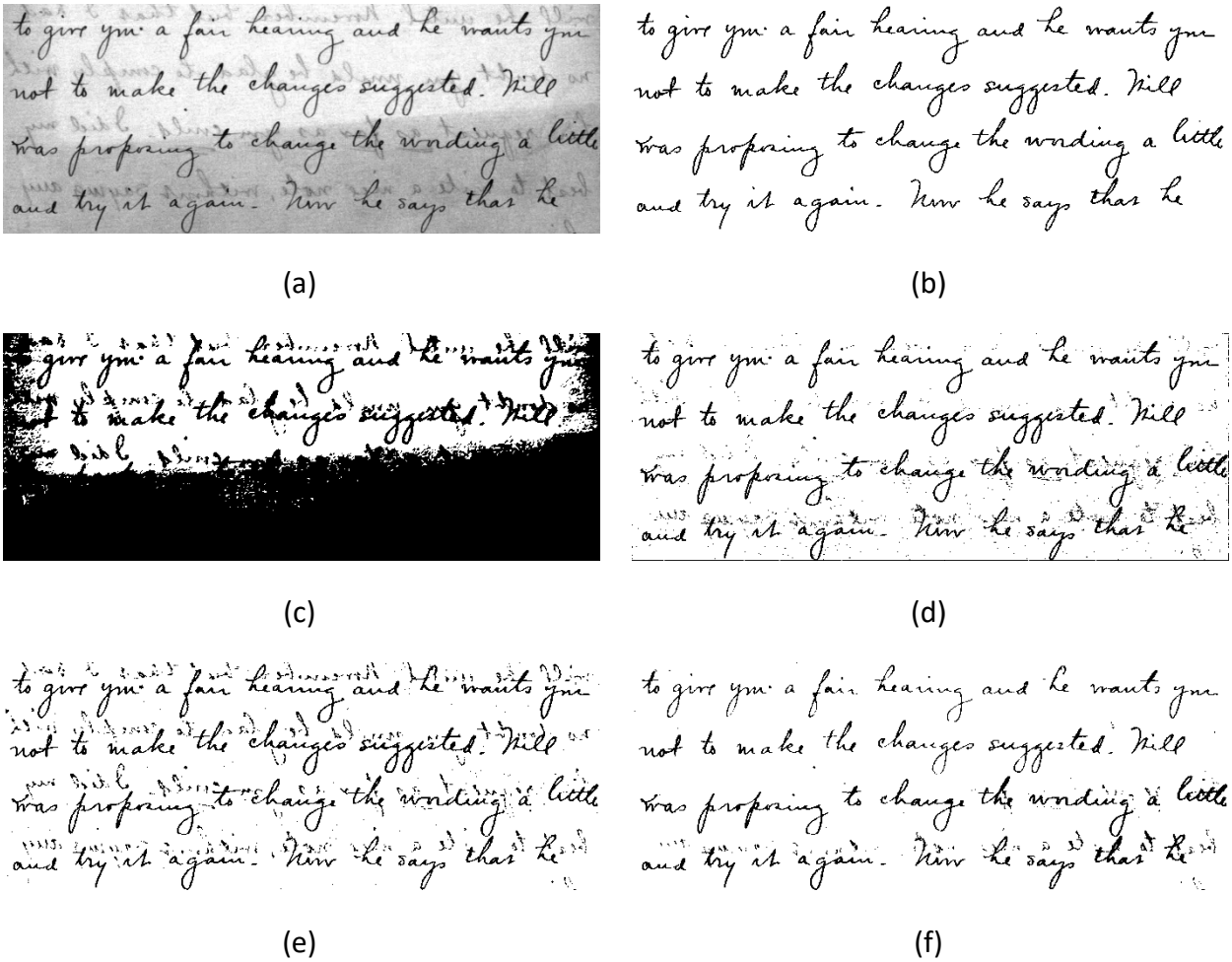


(a) (b)

(c) (d)

(e) (f)

Figure 3. 5 Résultat sur la base de données DIBCO 2011. [83]. (a) image original, (b) image de référence, (c) méthode d'OTSU, (d) méthode d'ISODATA, (e) méthode de Bernsen, (f) méthode de Niblack

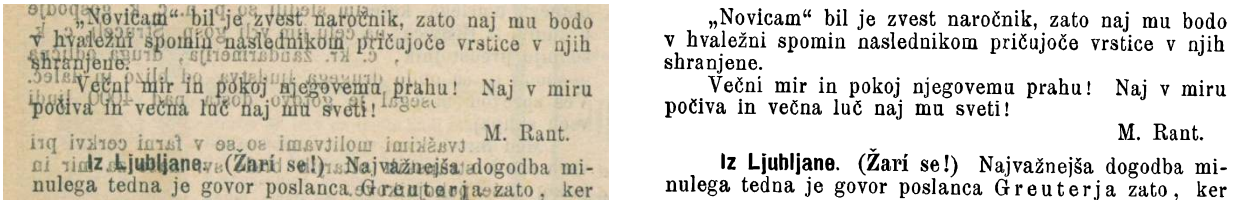


(a) (b)

(c) (d)

(e) (f)

Figure 3. 6 Résultat sur la base de données DIBCO 2012. [83]. (a) image original, (b) image de référence, (c) méthode d'OTSU, (d) méthode d'ISODATA, (e) méthode de Bernsen, (f) méthode de Niblack



(a) (b)

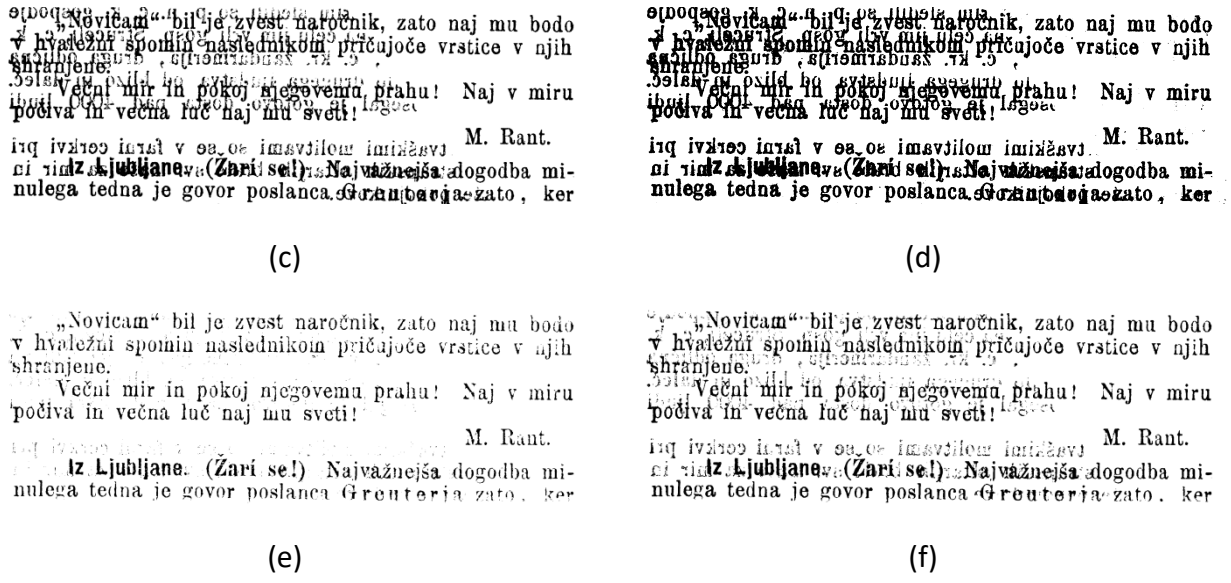


Figure 3. 7 Résultat sur la base de données DIBCO 20013. [83]. (a) image original, (b) image de référence, (c) méthode d'OTSU, (d) méthode d'ISODATA, (e) méthode de Bernsen, (f) méthode de Niblack

3.9 Conclusion

Dans ce chapitre, nous avons présenté différentes méthodes de binarisation et donnée quelques implémentations, en particulier les méthodes utilisant l'analyse par histogramme. Les méthodes de binarisation peuvent être regroupées en trois catégories

- Les méthodes globales sont plus ou moins rapides et faciles à mettre au point, mais elle présente des limitations surtout lorsque l'image présente des variations de luminance dans le font et l'objet c.à.d. lorsque l'histogramme de l'image est multimodal, ainsi que la nécessité de définir les paramètres manuellement (taille de la fenêtre, facteur k).
- Les méthodes locales, elles sont plus efficaces, plus complexes et sont composées de filtres et d'opérations existantes, ainsi elles nécessitent des capacités de calcul plus en fonction du choix de la taille de la fenêtre.
- Les algorithmes hybrides, combine en même temps entre les techniques global et locale, ainsi ils peuvent être rapide comme les méthodes globales, et fournissant des résultats de haute qualité comme les algorithmes de binarisation locales.

Nous pouvons conclure après ces tests qu'il est impossible d'affirmer d'une façon précise qu'une méthode est plus performante qu'une autre méthode pour un seul type de dégradation de document. C'est pour cette raison que les méthodes de binarisation sont évaluées sur un groupe

d'images minutieusement choisi. La décision de la performance de la méthode est prise en fonction de plusieurs types de mesure sur la moyenne de l'ensemble d'images.

Chapitre 4

Conception d'un système de binarisation d'images sur une puce programmable prototypée sur un FPGA

4.1 INTRODUCTION

Le processus de segmentation d'images ou de vidéos est requis en tant qu'étape de prétraitement dans plusieurs applications de traitement et d'analyse d'images, telles que la recherche des objets présentés dans l'image, la détermination de la région d'intérêt (ROI) à partir d'une image, la recherche d'un document image en reconnaissance optique de caractères (OCR) et détection d'objets en mouvement dans la reconnaissance de la marche humaine, la biométrie ou le suivi de cible [84,85].

L'utilisation d'images segmentées en mode binarisé réduit la charge de calcul globale dans une application spécifique. Le moyen utile de binariser une image est de définir un seuil et de séparer l'arrière-plan et le premier plan en fonction de l'intensité des pixels.

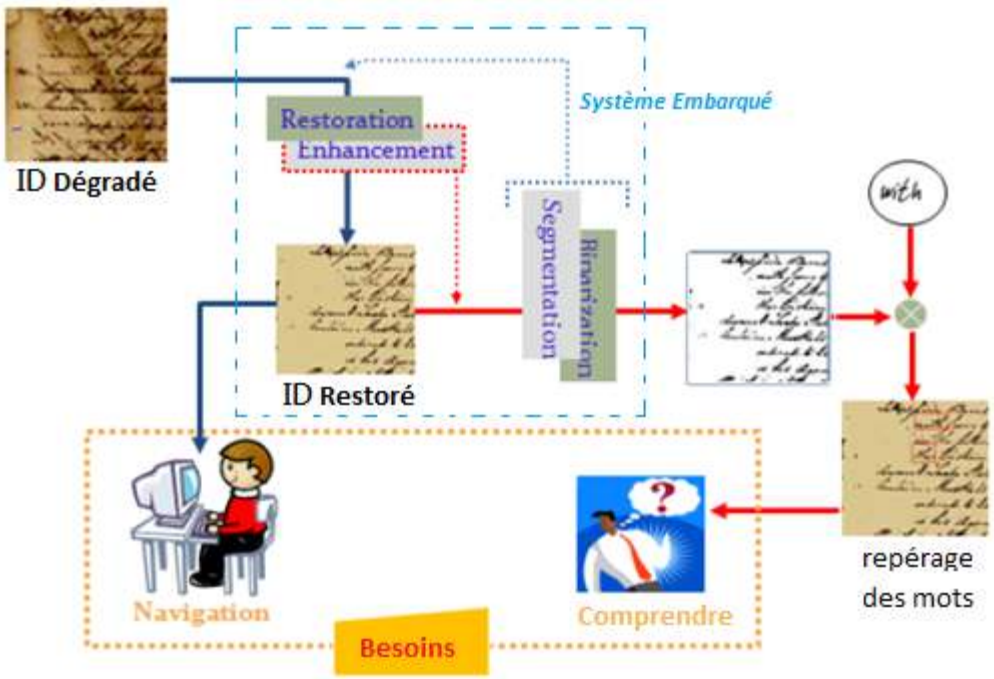
Le processus de préservation numérique (Figure 4 .1) par exemple évolue à partir de la création des images de haute résolution jusqu'à la génération de différentes versions de ces images qui présentent différents aspects comme la qualité et la taille. Une caractéristique commune entre toutes ces images, c'est qu'elles montrent la détérioration existante dans les documents originaux. La restauration et le rehaussement numérique, dans un contexte d'images de documents anciens dégradés, peuvent être vus comme des processus de transformation qui retrouvent l'aspect original de l'image du document ancien qui montre un certain état de dégradation.

Plusieurs méthodes (algorithmes) telles que Otsu, Isodata, Brensen, Niblack, Sauvola, sont mis en œuvre pour déterminer la valeur du seuil qui décide de l'ensemble des pixels qui prendront la couleur la plus blanche et ceux qui prendront la valeur la plus noire [86,87].

Ces traitements sont nécessaires non seulement pour améliorer la qualité de l'image du document, mais aussi pour améliorer les résultats des opérations ultérieures de segmentation et de reconnaissance. Ainsi, par le nettoyage des dégradations dans les images de documents dégradés, le rapport de l'interprétation du contenu sera plus important et plus significatif. Le traitement numérique des documents a pour but de partager et de diffuser l'information entre plusieurs types d'utilisateurs. L'information à diffuser sera utilisée selon des objectifs bien précis (Figure 4 .1), d'où la nécessité d'accélérer le traitement par des applications matérielles embarquées spécialisées et l'utilisation d'outils très performants.

Il est recommandé d'utiliser des ressources matérielles (FPGA), car dans le cas d'utilisation de processeurs classiques (processeurs classiques d'ordre général) et des processeurs de traitement de signal (DSP), les opérations arithmétiques et la demande de mémoire prennent un temps consistant, surtout lorsque la taille des images est importante.[89,90].

Ce chapitre présente en premier lieu une méthode de binarisation fondée sur l'utilisation d'une approche de non linéarisation du fond de l'image basé sur la classification d'image selon l'allure de l'histogramme dans le but d'améliorer les performances de la qualité d'images de vieux documents restaurés. Elle décrit en second lieu, une manière d'accélérer et d'implémenter un système de binarisation par la technique ISODATA dans une puce pour un système embarqué.



5 Figure 4 .1 Système de processus de préservation numérique

4.2 MÉTHODE BASÉE SUR LA NON LINÉARISATION DU NIVEAU D'INTERVALLE DE FOND ET L'ANALYSE "DCT"

Dans cette partie, une nouvelle technique de prétraitement pour l'opération de binarisation des vieux documents est présentée. Cette technique vise à réaliser, moyennant l'allure de l'histogramme, une opération non uniforme de filtrage sur le fond '' background'' d'une image afin d'améliorer les performances de la binarisation sans développer un nouveau Framework pour le calcul du seuil de binarisation.

4.2.1 Description de la méthode

La nouvelle approche se base essentiellement sur une classification de l'image, en image bruitée ''Noisy-image'' (image non uniformément éclairée ou transparente, rétroversion, taches d'humidité, décoloration ou tâche d'encre) ou image non bruitée ''Noiseless-image'', déterminée à partir d'une décision effectuée par l'algorithme de calcul du nombre de pics ''Sharp-Peack'' (Figure 4 .2). Selon l'état de cette décision, l'image subira soit un traitement par la transformée en cosinus DCT ou un traitement par niveau logarithmique.

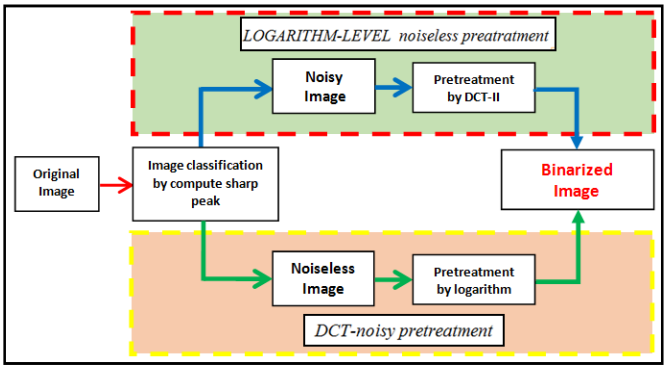


Figure 4 .2 Diagramme de la méthode proposée

4.2.2 Classification de l'image par Compute-Sharp-Peak

La classification de l'image en ''Noisy-image'' ou ''Noiseless-image'' se fait automatiquement en appliquant un algorithme pour calculer le nombre de pics de l'histogramme (Sharp-Peak) [92] à partir de l'histogramme de l'image algorithme-1. Afin d'augmenter la précision de détection des pics quatre positions voisines précédentes et quatre suivantes pour chaque valeur de niveau de gris sont prises. Si le nombre de pic aigu est supérieur à trois, il s'agit donc d'une image bruitée.

Algorithme 1

Algorithme ComputeSharpPeak (Image I)

/ Algorithme pour calculer le nombre de pics ''Sharp Peaks'' à partir de l'histogramme de l'image. */*

Entrée: image I

Renvoi: le nombre de pics dans l'histogramme d'une image

Begin

1. Calcul de l'Histogram de l'image (I);

2. Calculer la fréquence freq(k) des occurrences de pixels pour chaque valeur du niveau de gris k dans l'intervalle fermé [0..255]

3.

/ Les 2 positions voisines précédentes et 2position suivantes pour chaque valeur du niveau de gris k sont prises de manière circulaire respectivement en p1, p2, n1 et en n2. */*

pour k = 0 à 255,
  si (k=0) alors
    p1(k)=255; p2(k)=254; n1(k)=1; n2(k)=2;
  sinon
  si (i=1) alors
    p1(k)=0; p2(k)=255; n1(k)=2; n2(k)=3;
  sinon
  si (i=254) alors
    p1(k)=253; p2(k)=252; n1(k)=255; n2(k)=0;
  sinon
  si (i=255) alors
    p1(k)=254; p2(k)=253; n1(k)=0; n2(k)=1;
  sinon
    p1(k)=(k-1); p2(k)=(k-2); n1(k)=(k+1); n2(k)=(k+2);
  fin si
fin pour

4.

/ Si la fréquence d'une valeur de gris k est supérieure à celle de ses deux fréquences de valeurs de gris adjacentes gauche et droite, alors k est identifié comme* ***un pic ''Sharp Peak.*/***

n = 0;
*/*n est le nombre de pics dans l'image, initialisé à 0*/*
pour k=0 à255
si *freq(k) > max(freq(p1(k)), freq(p2(k)), freq(n1(k)), freq(n2(k)))* alors
/ max() renvoie la valeur maximale des argument */*
  n = n+1;
/ un pic est identifié pour la valeur de gris k */*
ph(n) = freq(k);
/ La valeur du ième choix est enregistrée à ph(i) */*
  fin si
fin pour

5.

/ La fréquence de crête moyenne est calculée à partir des crêtes enregistrées à l'étape 4 */*

avg-peak =∑ph(i)/n;

6. Un pic est défini comme un pic net, où la fréquence d'occurrences de pixels pour la valeur d'échelle de gris correspondante est supérieure à la fréquence de pic moyenne, avg-peak, calculée ci-dessus.
7. Renvoyez le nombre de **sharp peaks**.
Fin.

4.2.3 Prétraitement d'image sans bruit avec le niveau logarithmique LOGARITHM-LEVEL

Dans l'étape de prétraitement de la ''Noiseless-image'', un étalement des intervalles de bande des pixels constituant la zone mise en évidence est réalisé. Dans le but d'obtenir une meilleure classification des états des pixels, une action non linéaire faisant intervenir des expressions logarithmiques [93] de tous les pixels formant la zone mixte (fond et texte) (voir algorithme) est effectuée (Figure 4.3)

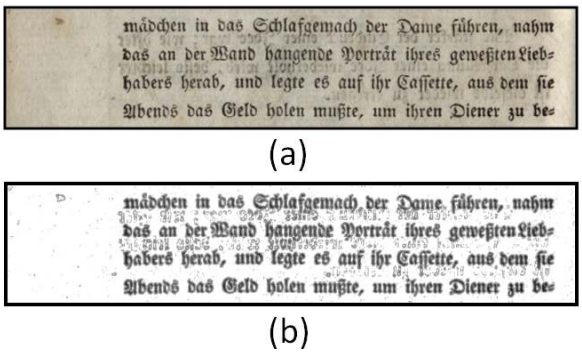


Figure 4 .3 Prétraitement par le niveau logarithmique (a) image originale, (b) image après prétraitement.

La non linéarisation dans l'intervalle des pixels des niveaux de gris est obtenue par l'expression:

$$X(i,j) = I(i,j) \times log_{MOY}(I(i,j)) \quad (43)$$

Ou :

X(i, j) : est le niveau de gris du nouveau pixel

$I(i,j)$: est le niveau de gris de l'image.

MOY : est la moyenne du niveau de gris de la fenêtre.

L'algorithme suivant donne les étapes de ce prétraitement.

Algorithme 2

Algorithme Logarithme-level (Image I)
/ Algorithme pour un prétraitement utilisant le niveau logaritmique */*
Entrée: image I
Renvoi: une image prétraité et binarisée avec OTSU

Begin

1:

*/*calcul de la moyenne du niveau de gris dans l'image*/*

Moyenne_image=$\sum niveauxdegrisdetouslespixelsdel'image/(nombretotaldepixels)$

2 : répéter les étapes 3, 4, et 5 pour tous les pixels de l'image

**/Pour tous les pixels de l'image (i=2 à largeur de l'image, j=2 à longueur de l'mage)*/*

3: Centrer une fenêtre de 3x3 sur le pixel X(i,j)

4: Calculer la valeur minimale Min, maximal Max et la moyenne Moy_f dans la fenêtre

5 : Calculer $D = Max - Moy\_f$ *et* $D' = Moy\_f - Min$

6: si $D < D'$*/*il s'agit du même voisinage*/*

si $PixelX(i,j) > Moyenne/2$

/* il s'agit d'un pixel de fond ''background''*/

sinon $New\_PixelX(i,j) = PixelX(i,j) * (log10(PixelX(i,j))/log10(Moy\_f))$

fin si

sinon $New\_Pixel\,X(i,j) = Pixel\,X(i,j)) * (log10(Pixel\,X(i,j))\,/log10(Moy\_f))$

fin si

End

7: Binariser l'image prétraitée par l'algorithme d'Otsu

4.2.4 Prétraitement d'image cas bruitée par la DCT [91]

La transformée en cosinus discrète (DCT) est une technique qui permet de représenter un signal en composante de fréquence élémentaire. Elle est largement utilisée dans les techniques de compression telle que JPEG. Elle convertit chaque valeur de pixel d'une image en sa valeur de fréquence correspondante.

La DCT, et surtout la DCT-II est particulièrement utilisée en traitement du signal l'image. La DCT possède en effet une excellente propriété de regroupement de l'énergie : l'information est principalement portée par les coefficients basses fréquences appelés coefficients d'approximation.

La DCT opère dans son domaine de fréquence équivalent en partitionnant la matrice de pixels d'image en blocs de taille NxN, N dépend du type de l'image. S'il s'agit d'une image en (niveau de gris) noir et blanc sur 8 bits, les blocs sont de $8x8$ et tous les ombrages de couleur noir et blanc peuvent être exprimés en 8 bits, de même pour une image en couleur de 24 bits, le bloc est taille N=24 la complexité temporelle peut augmenter, il serait préférable c'est d'opérer sur une composante de couleur individuelle pour une image couleur.

Mathématiquement parlant, la DCT est une fonction linéaire inversible $R^N \rightarrow R^N$ ou de manière équivalente une matrice carrée $N \times N$ inversible.

Le développement des algorithmes de calcul rapide des transformées DCT se basent sur la possibilité de décomposer la matrice de définition sous forme d'un produit de matrices dont le calcul est plus simple, et sert à diminuer le nombre de multiplications scalaires, en profitant des

identités remarquables de périodicité et symétries des fonctions sinusoïdales. Ainsi, on peut décomposer toute transformée DCT de RN en transformées plus simples.

Il existe plusieurs variantes de la DCT.

- La DCT-I $X_k = \frac{1}{2}(x_0 + (-1)^k x_{N-1}) + \sum_{n=1}^{N-2} x_n \cos\left(\frac{\pi}{N-1} n.k\right)$ (44)

- La DCT-II $X_k = \sum_{n=0}^{N-7} x_n \cos[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k]$ (45)

Dans le cas de traitement d'image, la DCT-II multidimensionnelle est utilisée :

Dans l'étape de prétraitement de l'image cas bruitée, un nouveau prétraitement en appliquant la transformée en cosinus discrète (DCT), avant l'opération de calcul du seuil de binarisation, est proposé.

La DCT-II sur des blocs de $4x4$ pixels est appliquée pour obtenir les coefficients d'approximation DCT; les 16 niveaux des différents pixels du bloc dans le domaine fréquentiel sont remplacés par la valeur du coefficient DC. Une fois cette étape terminée, l'image filtrée sera obtenue par l'opération inverse de la DCT-II^{-1}. Cette image sera par la suite binarisée par un algorithme de calcul de seuil (OTSU par exemple). Le schéma fonctionnel de la méthode proposée est donné dans la Figure 4.4.

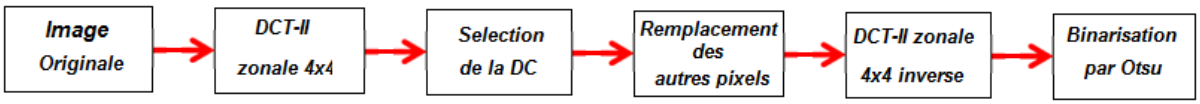


Figure 4 .4 Etapes d'amélioration de la binarisation par DCT pour images bruités.

4.2.5 Résultats expérimentaux et discussion

La méthode de binarisation proposée est évaluée sur un ensemble de test sur des images de la base de données DIBCO'09 [83]. Cet ensemble se compose de cinq images manuscrites et cinq images imprimées à la machine pour lesquelles sont associées les images de vérité terrain, des exemples d'images de l'ensemble de données pour les images imprimées et manuscrites sont affichées sur la Figure 4 .5 a, b et leurs respectives de vérité terrain sont représentées sur la Figure 4 .5 c,d. Cet ensemble de données fournit une collection d'images qui ont subi différents types de dégradation.

Figure 4 .5 Images de la base dibco

Les procédures d'évaluation utilisées comprennent un ensemble de mesures qui ont été largement utilisées à des fins d'évaluation dans la binarisation des documents anciens ces mesures consistent en :

• La mesure F ou ‘’F-Measure’’ est définie comme une moyenne harmonique de précision et de rappel [88].

$$FM = \frac{2\times rappel\times précision}{rappel+précision} \quad rappel = \frac{VP}{VP+FN}, Précission = \frac{VP}{VP+FP} \quad (46)$$

TP, FP and *FN* denote the true-positive, false-positive and false-negative values, respectively

- Le PSNR est une mesure de la proximité d'une image à une autre. Par conséquent, plus la valeur du PSNR est élevée, plus la similarité entre l'image binarisée et l'image de vérité terrain est élevée.

$$PSNR = 10lo\,g\left(\frac{C^2}{MSE}\right) \quad MSE = \frac{\sum_{x=1}^{M}\sum_{y=1}^{N}(I(x,y)-I'(x,y))^2}{MN} \quad (47)$$

• Mesure de distorsion réciproque de distance (DRD)

DRD mesure la distorsion visuelle dans les images de documents binaires.

$$DRD_K = \sum_{i=-2}^{2}\sum_{j=-2}^{2}|GT_K(i,j) - B_K(x,y)| \times W_{Nm}(i,j) \quad (48)$$

Les valeurs de test sont la valeur moyenne des dix images pour chaque mesure d'évaluation dans le tableau 4.1.

Figure 4 .6 Résultats de binarisation sur deux types de document par différentes techniques

Tableau 4.1 comparaison des résultats par les outils de mesure

Algorithm	***F-Measure***	PSNR	***DRD***	***Precision***
OTSU	75,880865	14,510213	25,106339	69,725939
Kittler	66,421235	11,462507	40,976129	55,163069
ISODATA	73,546903	14,041202	25,280317	72,915196
Niblack	48,793725	7,233527	88,397341	34,736934
Sauvola	76,400945	15,559229	6,733586	95,998365
Bernsen	76,590380	14,971854	10,881221	84,803941
Proposed	87,2025181	17.3203707	4,4992576	88,8057364

Pour évaluer l'approche, l'algorithme est testé sur deux groupes d'images (manuscrites et imprimées), le résultat et le test de performance de la méthode ont été comparés à d'autres méthodes existantes dans la littérature (OTSU, Kittler, ISODATA, Niblack, Sauvola, Bernsen). Les résultats trouvés, exprimés par figures précédentes et dans le tableau précédent ont montré que les prétraitements améliorent considérablement les performances de l'opération de binarization.

4.4 Architecture de conception avec la technologie sopc

Contrairement à d'autres conceptions où les IPs sont directement traduites par des outils de cogénération et de co-simulation avec un générateur de système ou un constructeur de DSP [94, 95,103], cette conception utilise deux parties combinant matériel et logiciel dans un système programmable sur puce SOPC.

Le SOPC réalisé est un système embarqué comportant en plus des IPs conçu et réalisées histogramme, ISODATA et binarisation (séparation Fond-text), un processeur NIOS-II, une mémoire On-Chip, des interfaces qui englobent une SRAM, une SDRAM, mémoire-flash, une interface SPI pour carte SD, une interface VGA pour l'affichage, un contrôleur DMA pour l'accès rapide mémoire et un module Ethernet pour l'accès réseau et une interface pour caméra. Le module interface JTAG-UART et le module JTAG-Debug servent respectivement pour le transfert de la configuration et de l'application pour le débogage. Le bus Avalon Switch Fabric communique les différentes informations entre les unités et une unité de calcul autour de deux IPs ISODATA-HISTOGRAM décrit avec un langage de description HDL figure 4.7.

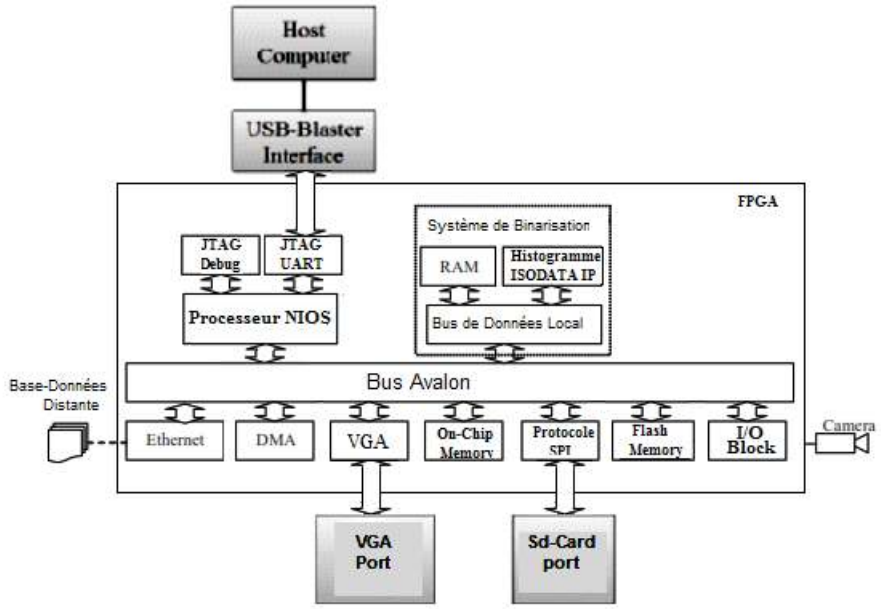


Figure 4 .7 Architecture SOPC d'une conception de binarisation d'images

La mise en œuvre du système complet implique une procédure de conception conjointe co-design matériel-logiciel. La stratégie est décrite sur la Figure 4 .8

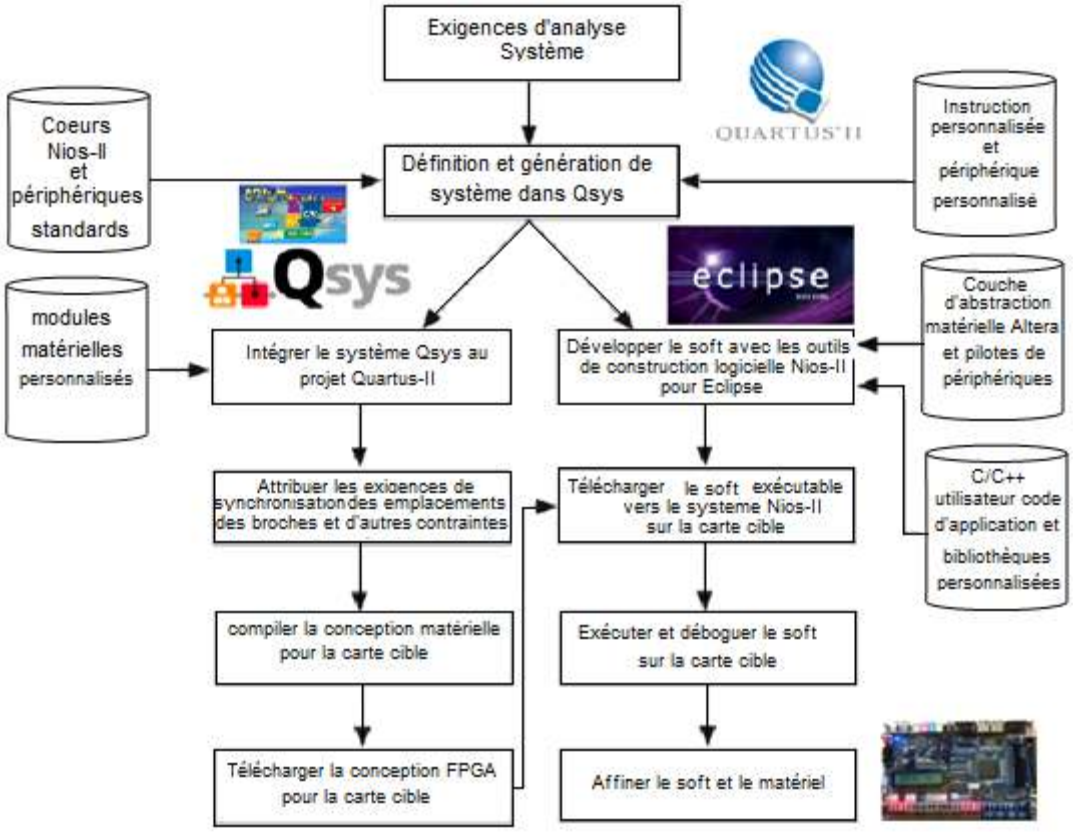


Figure 4 .8 Stratégie de Co-Design soft-hard

Dans la suite de ce paragraphe seront présentées, la partie matérielle avec les IPs essentiels et la partie logicielle avec un exemple d'étapes de gestion.[103]

4.4.1 Phase de conception du matériel

La capacité du FPGA à inclure un processeur logique a apporté une nouvelle dimension à l'utilisation de tels dispositifs. Dans lequel le processeur NIOS II fourni par ALTERA a une fonction similaire à un microcontrôleur qui intègre un processeur et comprend des périphériques et de la mémoire sur une seule puce.

La phase de conception matérielle concerne principalement la configuration du système qui est composé du soft-core NIOS II et d'un ensemble de périphériques. Cela peut être fait par le logiciel QUARTUS II, qui représente une suite d'outils CAO développés par ALTERA. La conception du matériel peut être effectuée avec la conception de l'environnement appelée « QSYS Builder ou SOPC-Builder ». [96, 97,98,103]

4.4.1.1 Processeur NIOS II et périphériques standards

- **Le processeur Nios**

Un NIOS II est un processeur soft core 32 bits se compose d'un ensemble de périphériques sur puce, d'une mémoire sur puce, de GPIO et d'autres composants utiles, tous connectés au bus Avalon pour générer un système [58, 59]. Opposé à un microcontrôleur fixe standard, un processeur NIOS II est configurable. L'ajout et la suppression de composants ou de fonctionnalités sur un système permettent d'atteindre facilement les objectifs de performances en termes de surface. Le processeur à cœur souple peut être ciblé sur n'importe quelle famille de FPGA ALTERA et n'est pas fixé dans le silicium.

Dans le système proposé, un processeur de classe économique NIOS II (version libre d'Altara) est choisi. De nombreux paramètres peuvent être définis lors de sa configuration. L'utilisateur peut choisir entre 16 ou 32 bits de largeur dans le chemin de données, la taille des registres de données, ainsi que la taille du cache et les instructions personnalisées.

A travers une application, ce processeur transfert les informations d'une image contenue dans la carte SD ou ne camera vers une mémoire, il déclenche l'opération de binarisation par l'implication de différents IPs. L'image binarisée est ensuite reconstruite sera re-stocké pour être afficher ou diffuser via un réseau.

b- Périphériques supplémentaires

En association avec le contrôleur Nios-II, plusieurs périphériques sont utilisés

- **VGA Video Graphics Array**

VGA est une norme industrielle, ou sont défini de nombreux paramètres tels que la résolution d'affichage, les taux de rafraîchissement, la synchronisation du signal de synchronisation, la polarité du signal et le niveau électrique des signaux RVB. Le moniteur affiche une trame de données d'image de M lignes et de N lignes de pixels. M×N est dit sur la résolution d'affichage. Le détail d'une trame de données d'image démontrée est que le contrôleur VGA lit d'abord des données de ligne à partir de la mémoire d'affichage de son correspondant, les envoie aux pixels correspondants sur le moniteur, puis affiche la ligne suivante. La trame suivante de données d'image sera affichée jusqu'à ce que toutes les lignes ci-dessus aient été complètement affichées. En une seconde, le nombre d'images affichées par un moniteur VGA correspond aux taux de rafraîchissement de l'affichage. Dans un taux de rafraîchissement suffisamment élevé, les yeux humains sentiront que l'image est continue, plutôt que d'être montré une ligne et une ligne. Pour un moniteur VGA ordinaire, son fil de sortie contient au total 5 signaux.[99,100]

- R, V, B : signal trichromatique.

- HS : signal de synchronisation de ligne.
- VS : signal de synchronisation de champ.

La synchronisation de ces 5 signaux doit respecter les "normes de l'industrie VGA", c'est-à-dire 640*480*60Hz. La Figure 4 .9 est la synchronisation du balayage de ligne et du balayage de champ de VGA et les tableaux 3, 4 sont leurs paramètres de synchronisation.

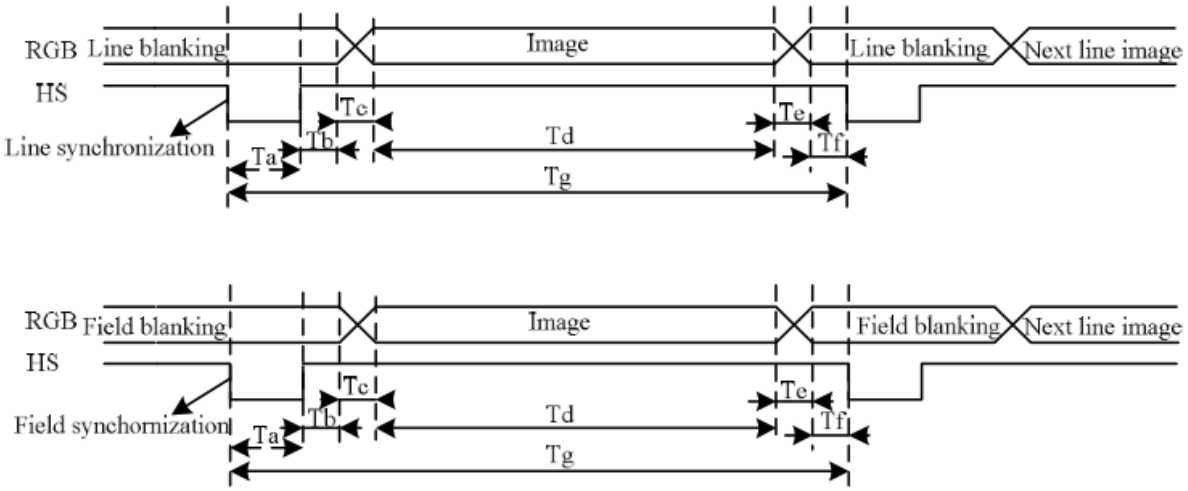


Figure 4 .9 Diagramme de synchronisation du balayage de ligne et du balayage de champ

Ta est la tête de synchronisation de ligne, Td est l'image de ligne, Tg est le cycle de ligne.

Tableau 2. Exigences de synchronisation de balayage de ligne (Unité : pixels, sortie d'un intervalle de temps de pixel

Positions correspondantes	T_f	T_a	T_b	T_c	T_d	T_e	T_g
Temps（pixels）	8	96	40	8	640	8	800

Tableau 2. Exigences de synchronisation de balayage de champ (Unité : lignes, sortie d'un intervalle de temps de ligne)

Positions correspondantes	T_f	T_a	T_b	T_c	T_d	T_e	T_g
Temps（pixels）	2	2	25	8	480	8	525

- **Contrôleur DMA**

Afin d'améliorer l'efficacité du processeur, cette conception utilise le DMA pour compléter les données d'image massives qui sont transmises de la SRAM à la mémoire FIFO du module de contrôle VGA. Avec le contrôleur DMA, un canal de transfert DMA entre le contrôleur VGA et la SRAM est établi, il permet au matériel de lire automatiquement les informations sur les pixels. Le module de contrôle de transmission DMA est illustré à la Figure 4 .3 [100]

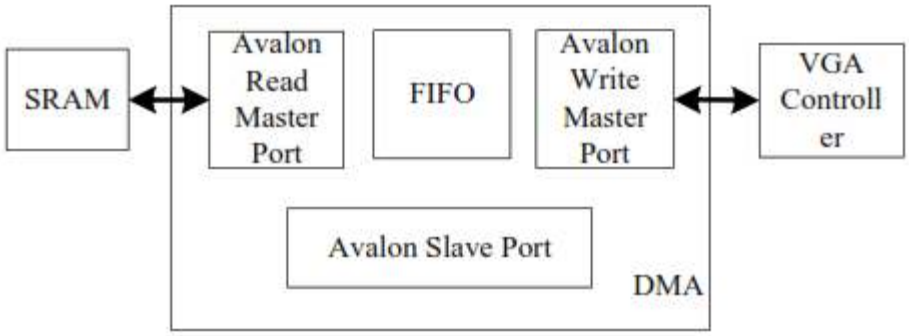


Figure 4 .10 Structure de transfert par le contrôle DMA

Dans le processus de transmission DMA, le CPU initialise le DMA par interruption dans un premier temps, ouvre le canal de transfert DMA, de sorte que le transfert n'est pas une intervention du CPU. Ensuite, le DMA lit directement les données d'image dans la SRAM vers la FIFO. Après avoir reçu le signal du contrôleur VGA, les données dans la FIFO sont lues vers le contrôleur VGA jusqu'à la fin de la transmission des données.[102]

- **Interface SPI de la carte Sd :**

Une carte SD est un périphérique de stockage de données. Il est souvent utilisé dans les dispositifs numériques tels que les appareils photo, les caméscopes, les lecteurs MP3, etc. Il est très pratique et permet aux données qu'y sont stockées d'être transférées sur plusieurs appareils.
La carte ALTERA DE2 contient un port pour carte SD. Ce port peut connecter une carte SD à la carte au système embarqué sur l'FPGA. Cela permet d'accéder à une grande quantité de données en utilisant un protocole sériel assuré par un core-IP SPI d'altera . Dans le système conçu les informations de l'image dans la SD sont transférées vers la SDRAM.[100]
L'IP Core-SPI est un circuit matériel qui implémente protocole de communication SPI "Serial Peripheral Interface" sur la carte DE2 [101]. SPI est une liaison de données série synchrone qui fonctionne avec un seul maître et plusieurs esclaves (au moins un).
Les broches utilisées dans SPI sont Fig. IV:

- SD_CLK : Utilisé pour contrôler la vitesse d'horloge de la carte SD.
- SD_DATA : Utilisé pour envoyer des commandes à la carte SD.

- SD_CMD : Utilisé pour le transfert de données.

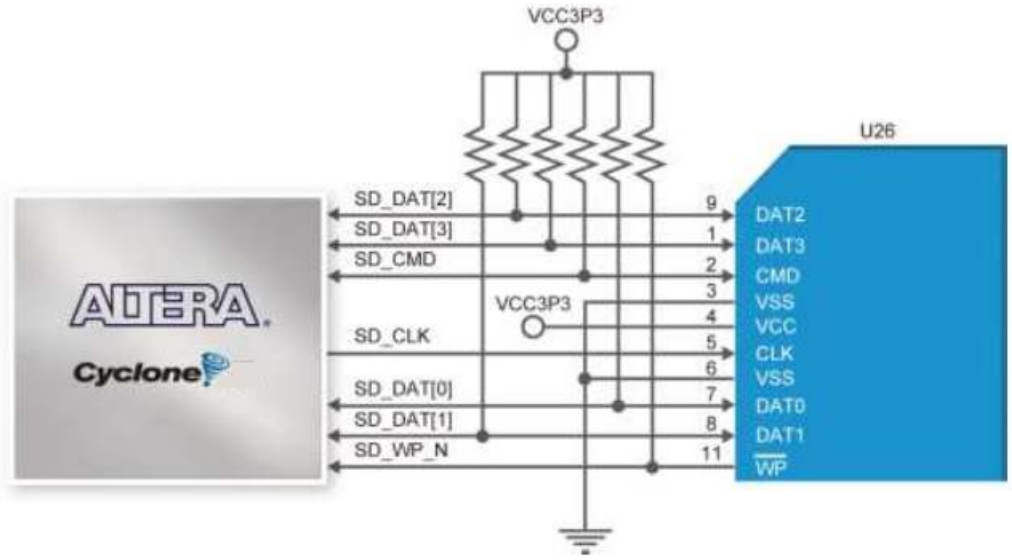


Figure 4 .11 SPI Interface for Sd-Card [SPI Sdcrd DE2]

- **Contrôleur Ethernet**

La carte DE2 fournit une prise en charge Ethernet via deux puces PHY Ethernet Marvell 88E1111 .La puce 88E1111 avec émetteur-récepteur Gigabit Ethernet 10/100/1000 Mbps intégré prend en charge les interfaces MAC GMII/MII/RGMII/TBI. [100]

Le Tableau 4.3 décrit les paramètres par défaut des deux puces. La Figure 4 .12 montre la configuration de la connexion entre le Gigabit Ethernet PHY (ENET0) et le FPGA.

Tableau 4.3 Paramètre émetteur/récepteur $88^{E}1111$

Configuration	*Description*	*Default Value*
PHYADDR[4:0]	PHY Address in MDIO/MDC Mode	10000 for Enet0;10001 for Enet1
ENA_PAUSE	Enable Pause	1-Default Register 4.11:10 to 11
ANEG[3:0]	Auto negotiation configuration for copper modes	1110-Auto-neg, advertise all capabilities, prefer master
ENA_XC	Enable Crossover	0-Disable
DIS_125	Disable 125MHz clock	1-Disable 125CLK
HWCFG[3:0]	Hardware Configuration Mode	1011/1111 RGMII to copper/GMII to copper
DIS_FC	Disable fiber/copper interface	1-Disable
DIS_SLEEP	Energy detect	1-Disable energy detect
SEL_TWSI	Interface select	0-Select MDC/MDIO interface
INT_POL	Interrupt polarity	1-INTn signal is active LOW
75/50OHM	Termination resistance	0-50 ohm termination for fiber

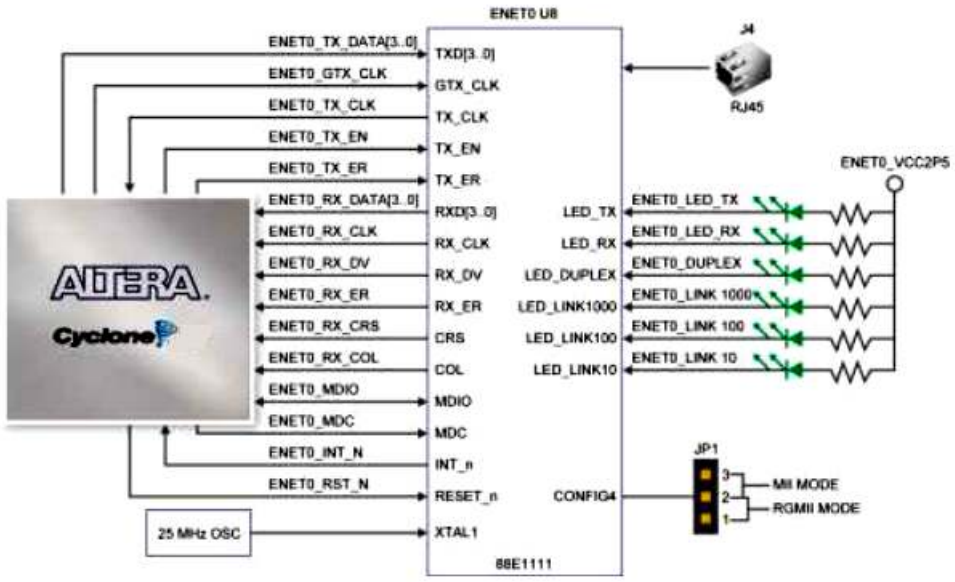


Figure 4 .12 Le schéma de structure du module matériel du système

Le matériel global comprend principalement la puce principale EP35F672C6 qui est la puce de la série Cyclone II d'Altera, périphérique, circuit d'horloge, circuit de réinitialisation/reconfiguration, alimentation, SDRAM, FLASH, circuit de mémoire SRAM et d'autres composants.

L'unité centrale et tous les périphériques du noyau NIOS II sont conçus sur mesure par l'outil QSys ou SOPC Builder, ce module d'outils configure principalement le processeur de Nios, JTAG, UART, le tristate bus avalon, mémoire sur puce, contrôleur SDRAM , interface flash commune.

Dans le cas de ce projet, le système est conçu sur puce programmable « SOPC » pour un système de segmentation-binarisation d'images basé sur la technique de seuillage ISODATA Figure 4 .1.

4.4.1.2 configurations et instanciation des modules

Dans cette architecture, le NIOS, processeur soft core, livré par Altera contrôle tous les éléments instanciés dans l'architecture par le bus Avalon.

Pour différentes tâches de la procédure de segmentation-binarisation, les propriétés intellectuelles (IP) du programme universitaire Custom Altra telles que : stockage, affichage, transfert mémoire, acquisition et transmission sont utilisées et instanciées en association avec les modules de segmentation à seuil proposés.

La mise en œuvre du système complet implique une procédure de co-conception matériel-logiciel :

• La fonctionnalité SOPC Builder, qui connecte en conséquence les composants logiciels et matériels pour construire un système informatique complet qui peut être contrôlé sur des puces FPGA et est également capable de produire automatiquement l'interconnexion adéquate. Figure 4.16.

• Framework IDE Eclipse et plug-ins Eclipse C development toolkit (CDT). Toutes les tâches de développement logiciel, y compris l'édition, la construction et le débogage, peuvent être accomplies à l'aide de NIOS II IDE [71].

4.4.2 Seuil d'image d'iso-data matérielles

Le seuillage par l'algorithme ISODATA [62] consiste à trouver un seuil en séparant l'histogramme en deux classes de manière itérative avec la connaissance préalable des valeurs associées à chaque classe. Cette méthode commence par diviser l'intervalle en valeurs non nulles représentant la population de fond C0 et la population de premier plan C1 de l'histogramme en deux parties équidistantes, puis calcule les moyennes arithmétiques m1 et m2 de chaque classe. Répéter le calcul du seuil T jusqu'à convergence vers la valeur la plus proche de (m1 + m2) / 2, et à chaque fois mettre à jour les deux moyennes m1 et m2 : voir les étapes de l'algorithme ci-dessous.

Algorithme 3

Algorithm1: Isodata" threshold selection based on the iterative method by Ridler and Calvard

1: K ← size (h) -- *number of intensity levels*

2: T ← mean (h, 0, K−1) -- *set initial threshold to overall mean*

3: **repeat**

4: C_0← count (h, 0, T) -- *background level population*

5: C_1← count (h, T+1, K−1) -- *foreground level population* $\boldsymbol{Count(h, a, b)} = \sum_{g=a}^{b} \boldsymbol{h(g)}$

6: **If** (C_0=0) (C_1=0) **then**

7: **return -1** -- *background or foreground empty*

8: m1← mean (h, 0, T) -- *background mean*

9: m2← mean (h, T+1, K−1) -- *foreground mean*

10: T'← T -- *keep previous threshold*

11: T ← [m1 + m2]/2

12: **Until** T = T' -- *end of the loop if no-change*

13: **return T**

L'histogramme d'une image en niveaux de gris, rend compte du nombre de pixels dans une image en fonction de la valeur du pixel. Les deux principales étapes associées à l'utilisation des histogrammes permettent de construire l'histogramme, d'en extraire des données et de les utiliser pour le traitement de l'image.

$$h[i] = \sum_{x,y} \begin{cases} +1, when & I[x,y] = i \\ 0, & otherwise \end{cases} \qquad (49)$$

L'histogramme est alors un diagramme utilisé dans l'analyse des données numériques, qui décrit combien de pixels d'une image ou d'une trame vidéo ont une certaine intensité. Il est souvent requis pour de nombreuses applications en traitement d'images et vidéo ou pour mesure d'évaluation. La Figure 4 .13 montre comment utiliser des compteurs ou de la mémoire pour implémenter l'histogramme d'une image. [106]

Pour cette application, une mémoire avec des données à double port et une adresse à double port est utilisée avec un décodeur et un additionneur d'accumulateur.

Les pixels de l'image d'entrée sont accumulés pour chaque niveau de gris de l'image, à la fin, la mémoire contient des valeurs qui représentent l'histogramme.

L'unité de calcul de la méthode de seuil Isodata, Figure 4.14, se compose de plusieurs parties et fonctionne selon l'algorithme1.

Après avoir obtenu l'histogramme, figure 4.13, les statistiques des valeurs des pixels de niveau se trouvent à l'intérieur d'une mémoire à double port d'entrée et à double port d'adresse et seront divisées en deux classes C1 et C2 à l'aide du registre de seuil. Au début ce registre contient un seuil initial puis les valeurs des seuils calculés dans les itérations suivantes.

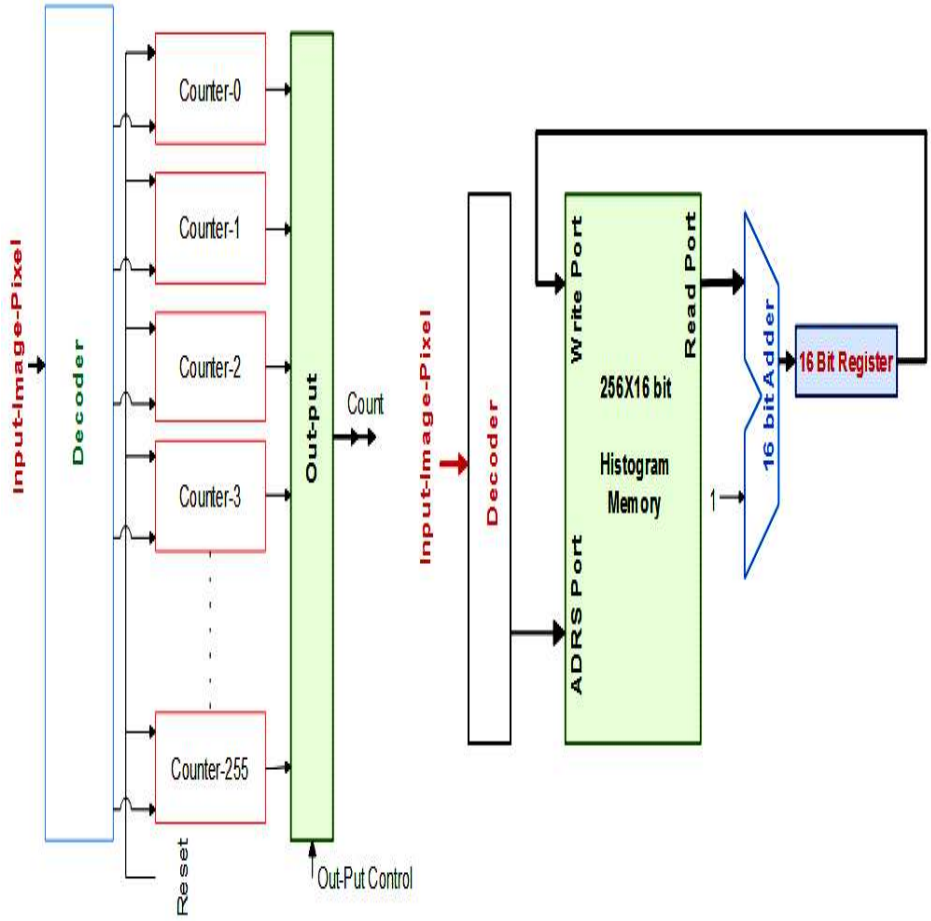


Figure 4 .13 Circuit de mise en œuvre d'histogramme avec compteur VS mémoire

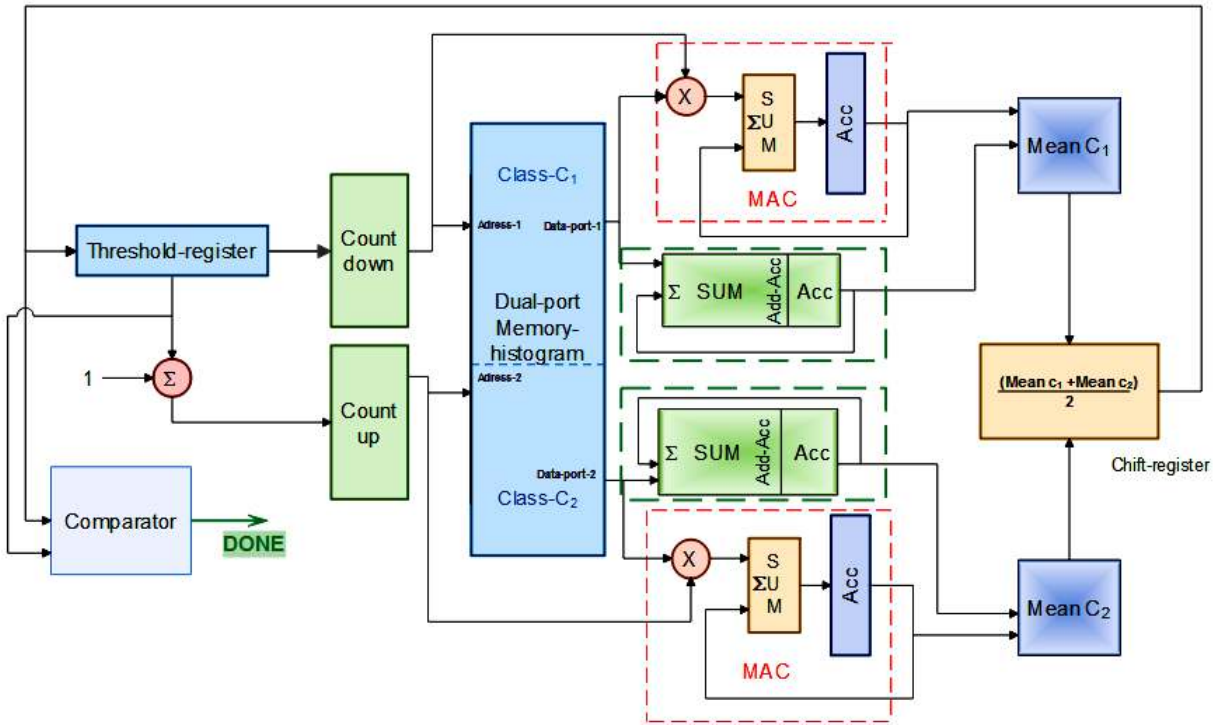


Figure 4 .14 Unité de calcul à seuil ISODATA [106]

Pour scruter les deux zones classe-C1 et classe-C2, l'unité de calcul utilise un décompteur initié par la valeur du registre de seuil et un décompteur initié par la valeur seuil +1. Cela rend le traitement plus rapide pour le processus d'itérations.

Les deux sorties de la mémoire sont pondérées et cumulées respectivement par le contenu des valeurs du décompteur et du décompteur par le module MAC cumulé de multiplication. En même temps, les valeurs de chaque classe (sortie de données de mémoire) sont accumulées par un additionneur d'accumulation "Add-Acc". Les résultats obtenus par les opérations MAC et ADD-ACC conduisent à obtenir les moments respectifs pour chaque classe de l'histogramme.

La moyenne de ces moments produite par un registre à décalage droit est transférée au registre à seuil pour une nouvelle itération si cette valeur n'est pas égale à la valeur calculée à l'itération précédente, sinon un signal fait est envoyé au processeur pour indiquer la fin de cette étape et le début de l'étape de binarisation. [106]

La dernière unité utilisée dans le processus de segmentation est la construction de l'image binirisée. Dans cette phase, chaque pixel de la mémoire-image est transformé en arrière-plan (pixel=0) ou en avant-plan (pixel=255) selon la valeur du seuil Isodata Figure 4 .15.

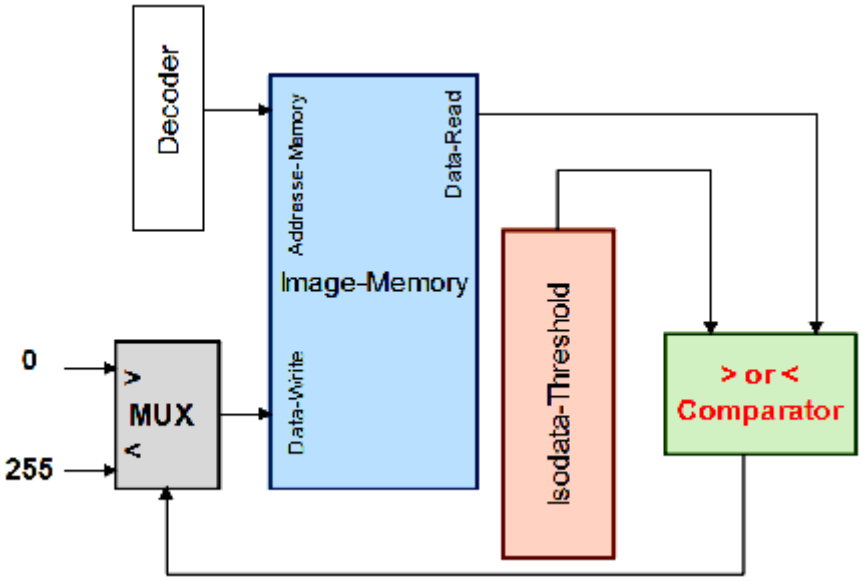


Figure 4 .15 Circuit de binarisation d'image (séparation Fond-text) [106]

La fonctionnalité SOPC Builder [106], qui connecte en conséquence les composants logiciels et matériels pour construire un système informatique complet qui peut être contrôlé sur n'importe laquelle des puces FPGA et est également capable de produire automatiquement une logique d'interconnexion. Figure 4.16.

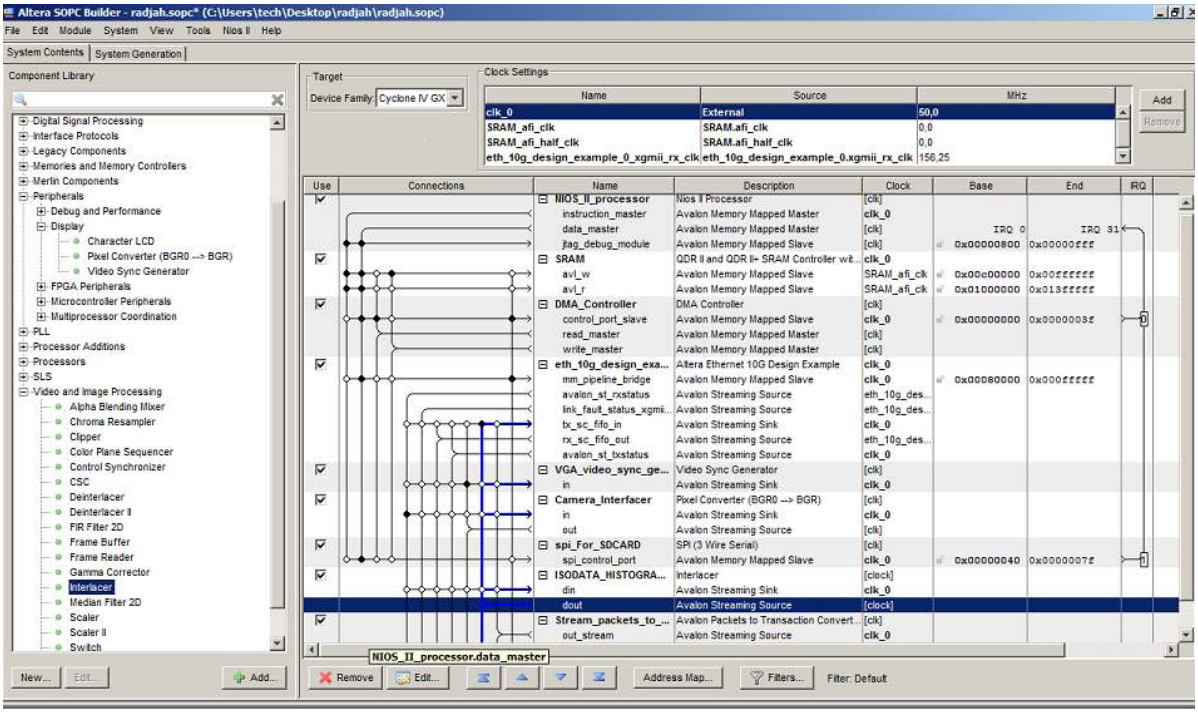


Figure 4 .16 Conception et configuration du système par Qsys ou Sopc-builder

4.4.3 Phase de conception du logiciel

Afin de tester le SOPC, un programme d'application pour le processeur NIOS II doit être écrit. L'environnement de développement intégré est une interface utilisateur graphique (GUI) de développement logiciel pour le processeur NIOS II. Il est basé sur le framework Eclipse IDE et les plug-ins Eclipse C development toolkit (CDT). Toutes les tâches de développement logiciel, y compris l'édition, la construction et le débogage, peuvent être accomplies à l'aide de NIOS II IDE.

L'application commence par les initialisations des différentes interfaces et variables. Il vérifie ensuite la présence d'un élément de stockage ou de capture (SD-Card dans ce cas ou caméra en cours d'utilisation). Le processeur NIOS donne le départ du traitement confié au module matériel de binarisation. Une fois terminé, un signal est émis indiquant la fin du traitement. L'image binarisée est maintenant prête à être affichée ou stockée sur la carte SD Figure 4.6.

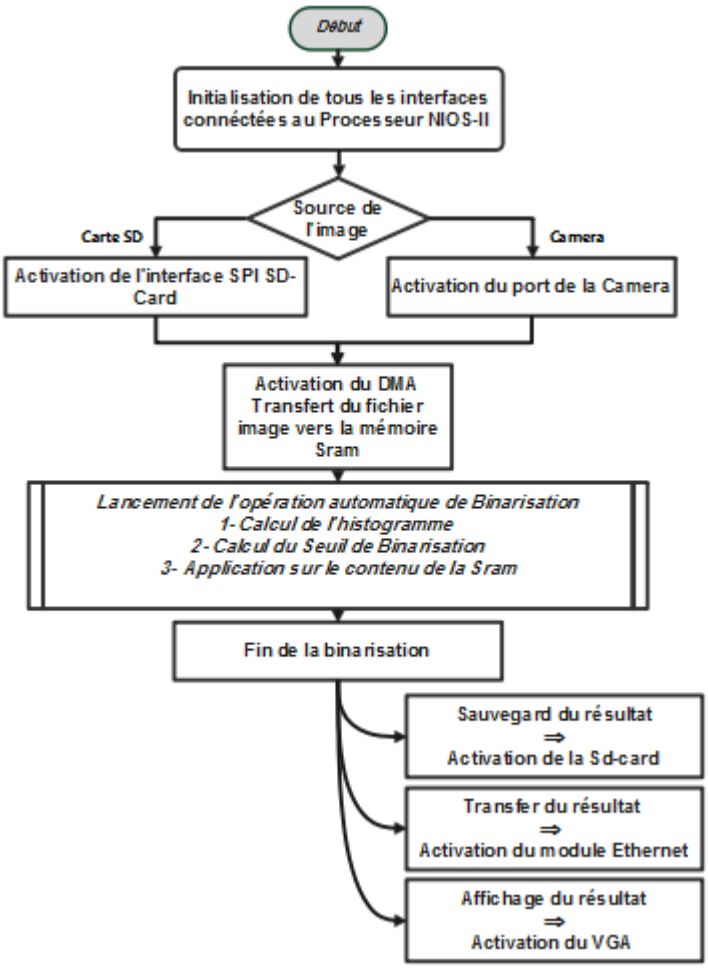


Figure 4 .17 Algorithme logiciel pour processeur NIOS-II [106]

4.4.5 Synthèse et simulation et discussion

Le résultat de synthèse obtenu pour l'architecture de conception est présenté dans le tableau 4.4. La conception est décrite en utilisant le VHDL structural et synthétisé sur le Cyclone FPGA II EP2C35F672C6 [105]. La vue au niveau RTL d'une partie du module MAC est représentée sur la figure 4.7 et le compte rendu de son opération de synthèse est représenté sur la Figure 4 .8.

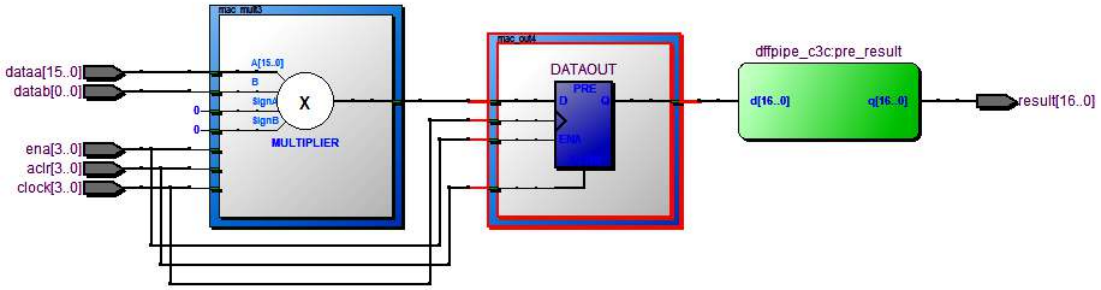


Figure 4 .18 Exemple de niveau de transfert de registre RTL d'une partie du système

Tableau 4.4: consommation de ressources dans le FPGA

Modules	Eléments Logiques (Logic Elements)	Registres (Registers)	Bits Mémoire (Memory Bits)	Multiplieurs DSP (DSP Multipliers)
Dual-Port Histogram Memory	-	-	4 K	-
MAC	33 x2	32 x2	-	2 x2
SUMAcc	17 x2	16 x2	-	-
Increment	8	-	-	-
Threshold Register	8	8	-	-
Count-UP Count-Down	8 x2	8 x2	-	-
Comparator	5	1	-	-
Means-C1 Means-C2	858 x2	-	-	-
Total	1853	121	4 K	4

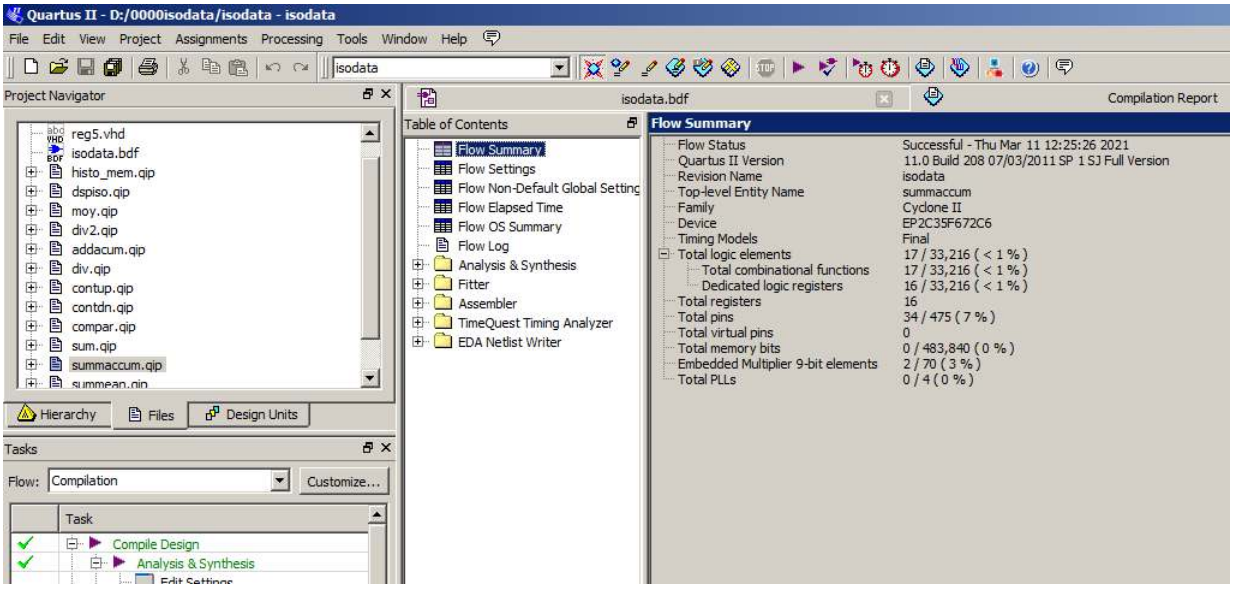
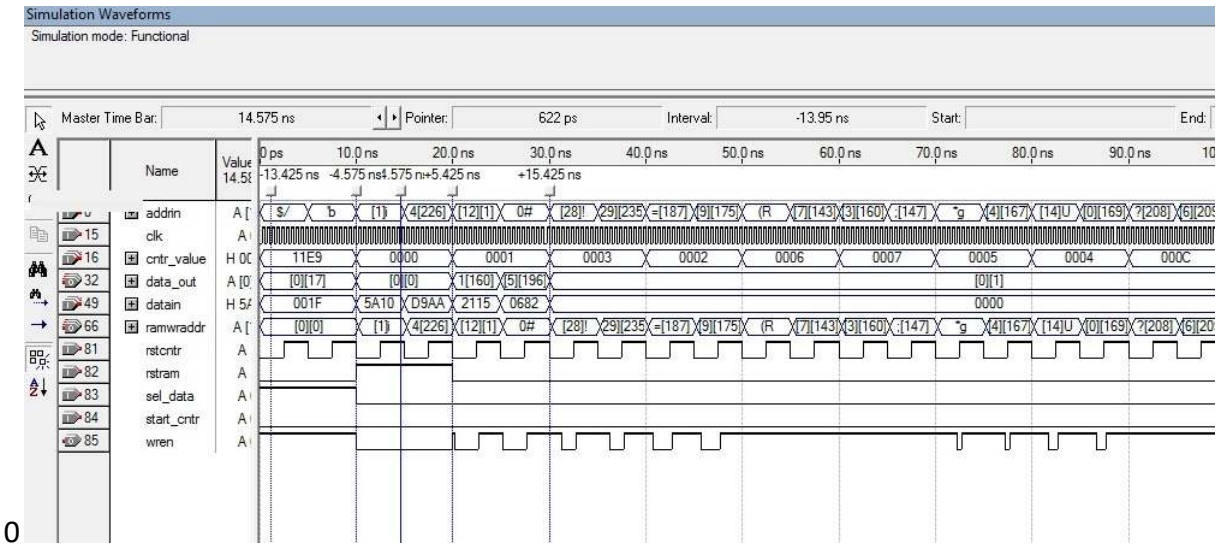


Figure 4 .19Ancien dix. Rapport de compilation de synthèse fourni par Quartus Compilation

4.4.6 Résultats et discussion

Pour mettre en évidence l'efficacité de ce travail, l'application développée est soumise à divers tests.

Pour valider le fonctionnement du module d'histogramme et le module de calcul ISODATA ; nous avons procédé par une simulation temporelle. Les temps trouvés montrent une latence acceptable vise à vis au temps de calcul sur un système à usage général (PC) et cela malgré que la fréquence de traitement choisie est uniquement de quelque dizaines de mega-hertz. Figure 4.20.



0

Figure 4 .20 . Résultats de la simulation d'histogramme

La figure 4.21.a et la figure 4.21.b montrent respectivement le DE2-Kit, l'état de sortie de la carte de prototypage DE2 avec la configuration initialisée. La figure 4.21.c montre l'affichage du système vierge sans charger le programme dans la mémoire flash du programme.

La Figure 4 .22 montre l'affichage des images de test (images originales d'anciennes impressions et d'écriture manuscrite) [83]. Les résultats de l'opération de segmentation sont donnés par la Figure 4 .23.

Nous avons remarqué que les résultats obtenus sont très proches de ceux obtenus à l'aide d'applications logicielles, malgré la faible précision utilisée du nombre de bits pour représenter les données de traitement.

Figure 4 .21 Altera DE2-kit and primary display

Figure 4 .22 Exemple of Original Old-Documents Images display

Figure 4 .23 Old-Document images Threshold by the SOPC-System

Ce choix d'image est justifié pour voir l'amélioration de la latence d'exécution par rapport à d'autres travaux notamment les travaux de Khitas et al [104] pour mettre en œuvre sa méthode.

Par rapport aux procédures conventionnelles, effectuées sur un Pentium IV 3.00 Ghz, la vitesse d'exécution est nettement meilleure lorsque le NIOS est cadencé par une horloge de seulement 100 Mhz. Cette amélioration de la vitesse est due à l'aspect parallèle de l'exécution des tâches dans les systèmes matériels.

4.5 CONCLUSION

L'implémentation matérielle peut être plus avantageuse notamment pour les applications de traitement d'image ou de vidéo puisque le pipeline est plus adapté et les modules FPGA sont dédiés à ce type de tâches

Ce travail propose un schéma de conception de pré-traitement d'images basé sur SOPC sur carte DE2. Les objectifs du projet étaient d'examiner les développements dans la conception de systèmes embarqués et les tendances futures, et d'explorer le prototypage rapide au niveau de la carte à l'aide de FPGA (DE2).

Un bon exemple est l'application conçue de binarisation d'images par calcul du seuil ISODATA analysant l'histogramme. Il montre que l'utilisation de l'approche hardware dans les conceptions d'images donne de meilleurs résultats en termes de vitesse, de taille et de coût.

L'idée suivante serait de continuer à augmenter le niveau de complexité de la mise en œuvre pour le traitement d'image typique pour les applications intégrées d'OCR, d'analyse d'image de document, d'extraction de motif et de prétraitement d'empreintes digitales.

Conclusion Générale

L'objectif de cette thèse est la contribution au développement et à la conception d'un système de binarisation embarqué dans des équipements d'acquisition d'image pour la numérisation de document. L'approche utilisée est une philosophie utilisant une conception conjointe (*codesign*) logicielle/matérielle. Une élaboration de modules IPs (*Intellectual Property*) en HDL (*High Description Language*) puis leurs instanciation dans la conception en compagnie avec d'autres composants délivrés par des concepteurs Altera ou Xilinx, dans le cadre des accords programmes interuniversitaire « University Program (*UP*) », a constitué la partie matérielle de la conception. L'utilisation du processeur NIOS-II à travers un programme développé en langage C a permis le contrôle et la gestion de ce prototype. Le système a été implémenté sur une carte kit DE2 sous la suite de l'environnement de développement Quartus-II d'Intel Altera.

Le choix de l'application faisant l'objet de ce travail s'est porté sur un système de numérisation dédié au traitement des vieux documents, particulièrement un rehaussement par binarisation. Deux modules principaux, (calcul de l'histogramme et seuillage par la technique ISODATA) plus le module de séparation fond-text ont été développés en technologie FPGA pour permettre une accélération et une optimisation du traitement.

Les Images de la base de donnée DIBCO ont été utilisés pour valider la conception et montrer sa performance par l'utilisation de quelques outils de mesure en particulier la F-Mesure et le PSNR.

Les résultats de test ont montré l'efficacité de l'approche de point de vu accélération de traitement, embarquement et portabilité. Cette approche est à conseiller pour être étendue à d'autres techniques de binarisation tel que les techniques adaptatives.

References

[1] Anne Ramsden, Mel Collier, Clare Davies, Anil Sharma, Dian Zhao, '' ELINOR – Electronic Library Project'', International Institute for electronics library research, De Montfort university Milton Keynes, British library 1998

[2] De Cher Threinen-Pendarvis, ''The Photoshop and Painter Artist Tablet Book: Creative Techniques in Digital Painting using Wacom and iPasd'', Peatchpit press 2014

[3] Trier, O. D., and Jain, A. K., ''Goal-Directed Evaluation of Image Binarisation Methods, IEEE Transactions on PAMI, 17(12), pp 1191-1201, 1995.

[4] [dibco] Ioannis Pratikakis, Konstantinos Zagoris, George Barlas and Basilis Gatos, '' ICDAR2017 Competition on Document Image Binarization (DIBCO 2017)'', 14th IAPR International Conference on Document Analysis and Recognition.

[5] Wang Jianlai, Yang Chunling, Zhu Min, Wang Changhui, ''Implementation of Otsu's thresholding process based on FPGA'', 2009 4th IEEE Conference on Industrial Electronics and Applications, 25-27 May 2009, Xi'an, China, DOI: 10.1109/ICIEA.2009.5138252

[6] Elham Ashari, Richard I. Hornsey, ''FPGA implementation of real-time adaptive image thresholding'', Proceedings of SPIE - The International Society for Optical Engineering DOI: 10.1117/12.566861

[7] H.C. Hsieh et al., Third generation architecture boosts speed and density of FPGAs, in: Custom Integrated Circuits Conf., pp. 31.2.1-31.2.7, 1990.

[8] G. Bostock, Programmable Logic Devices: Technology and Applications (McGraw-Hill, New York, 1988).

[9] D. Pellerin and M. Holley, Practical Design Using Programmable Logic (Prentice-Hall, Englewood Cliffs, N.J.,1991).

[10] P.K. Lala, Digital System Design Using Programmable Logic Devices (Prentice-Hall, Englewood Cliffs, N.J., 1990).

[11] J. Rose, R. Francis et al., Architecture of FPGAs: The effect of logic block functionality on area efficiency, Journalof Solid State Circuits, Vol. 25, pp. 1217-1225, 1990.

[12] K. E1-Ayat, A. E1-Gamal et al., A CMOS electrically configurable gate array, Journal of Solid State Circuits, Vol. 24,pp. 752-761, 1989.

[13] The Programmable Logic Data Book 2000

[14] XILINX, 2100 Logic Drive, San Jose, California, 95214, The Programmable Gate Array Book, 1991 ed

[15] XC4000E and XC4000X Series Field Programmable Gate Arrays

[16] Altera Data Book, 1990 ed., October.

[17] Cyclone Device Handbook, Volume 1

[18] M. Bolton, Digital Systems Design with Programmable Logic, Electronic Systems Engineering Series (Addison-Wesley, Reading, Mass., 1990).

[19] S. Singh, J. Rose et al., The effect of logic block architecture on FPGA performance, Journal of Solid State Circuits,Vol. 27, pp. 281-287, 1992.

[20] Gordon E Moore et al. Cramming more components onto integrated circuits. Proceedings of the IEEE, 86(1) :8285, 1998.

[21] Michael S Shur, Yoshi Nishi, Hiroshi Iwai, Hei Wong (Editor) Frontiers in Electronics - Proceedings of the Wofe-04 (Selected Topics in Electronics and Systems)

[22] https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/

[23] Marilyn Wolf,'' Computers as Components Principles of Embedded Computing System Design'', Morgan Kaufmann, Elsevier 2017, https://doi.org/10.1016/C2015-0-04103-X

[24] Laung-Terng Wang, Yao-Wen Chang and Kwang-Ting (Tim) Cheng, '' Electronic Design Automation'', Morgan Kaufmann, Elsevier 2009, https://doi.org/10.1016/S1875-9661(08)X0006-4.

[25] James O. Hamblen, Tyson S. Hall, Michael D. Furman , '' Rapid Prototyping of Digital Systems: SOPC Edition'', Springer-Verlag US 2008

[26] IEEE Standard for Verilog ® Hardware Description Language IEEE Std 1364™-2005

[27] Nicolas Maussang , Peggy Zwolinski and Daniel Brissaud, Product-service system design methodology: from the PSS architecture design to the products specifications, Journal of Engineering Design, 2009, Taylor & Francis, SN - 0954-4828 doi: 10.1080/09544820903149313

[28] Altera Corporation Website, www.altera.com, June 2006

[29] Nios II Processor Handbook",

[30] https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/nios2/n2cpu-nii5v1gen2.pdf
[31] Xilinx Incorporated Website, www.xilinx.com, June 2006
[32] MicroBlaze Processor Reference Guide", Xilinx Corporation, October 5, 2005
[33] PicoBlaze 8-bit Embedded Microcontroller User Guide", Xilinx Corporation, November 21, 2005
[34] Tensilica Incorporated Website, www.tensilica.com, June 2006
[35] Tensilica Diamond Standard Series Product Brief", Tensilica Incorporated, 2006
[36] Tom R. Halfhill, "Tensilica's Preconfigured Cores", Microprocessor Report, March 20, 2006
[37] Xtensa Overview, www.tensilica.com/products/overview.htm, June 2006
[38] Create TIE, www.tensilica.com/products/create tie.htm, Septembe 2006
[39] TIE Compiler, www.tensilica.com/products/tie compiler.htm, September 2006
[40] XPRES Compiler, http://www.tensilica.com/products/xpres.htm,September 2006
[41] Opencores.org Website, www.opencores.org, June 2006
[42] UT Nios Homepage, www.eecg.toronto.edu/˜plavec/utnios.html, June 2006
[43] OpenSPARC Website, www.opensparc.org, June 2006
[44] Gaisler Research Website, www.gaisler.com, June 2006
[45] Mishra, P. and Dutt, N., "Architectural description languages for programmable embedded systems", IEEE Proceedings of Computers and Digital Techniques, May 2005, pp. 285–297
[46] LEON2 Processor User's Manual XST Edition", Gaisler Research, July 2005
[47] GRLIB IP Core User's Manual", Gaisler Research, February 2006
[48] Broadcom Website, www.broadcom.com, June 2006
[49] Broadcom CALISTOTM BCM1500 Leverages Multiple Xtensa Cores
[50] for VoIP ASSP", Tensilica Incorporated Success Story, 2001
[51] [28]Mariem Makni et al "A Comparison and Performance Evaluation of FPGA Soft-cores for Embedded Multi-core Systems", 2016 11th International Design & Test Symposium (IDT)154, Hammamet, Tunisia, DOI: 10.1109/IDT.2016.7843032
[52] www.niosforum.com,
[53] Nios II Software Developer Handbook, https://www.intel.com
[54] A graphical bootloader for a Altera DE2 FPGA board (with Nios processor), https://github.com/janneku/nios-bootloader
[55] B. Sankur, « Survey over image thresholding techniques and quantitative performance evaluation », J. Electron. Imaging, vol. 13, no 1, pp. 146, janv. 2004.
[56] B. Su, S. Lu, C. L. Tan, « Combination of Document Image Binarization Techniques », in 2011 International Conference on Document Analysis and Recognition, Beijing, China, 2011, pp. 22-26.
[57] N. Arica, F. T. Yarman-Vural, « An overview of character recognition focused on off- line handwriting », IEEE Trans. Syst. Man Cybern. Part C Appl. Rev., vol. 31, no 2, pp. 216-233, mai 2001.
[58] A. Kefali, T. Sari, M. Sellami, « Evaluation de plusieurs techniques de seuillage d'images de documents Arabes anciens », in 5ème Symposium International Image, Multimédia, Application Graphique et Environnement (IMAGE'2009), Biskra,Algeria, 2009.
[59] S. M. Ismail, S. N. H. Sheikh Abdullah, F. Fauzi, « Statistical Binarization Techniques for Document Image Analysis », J. Comput. Sci., vol. 14, no 1, pp. 23-36, janv. 2018.
[60] N. Otsu, « A Threshold Selection Method from Gray-Level Histograms », IEEE Trans. Syst. Man Cybern., vol. 9, no 1, pp. 62-66, janv. 1979.
[61] J. Kittler, M. Hatef, R. P. W. Duin, J. Matas, « On combining classifiers », IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, no 3, pp. 226-239, mars 1998.
[62] V. FLAVIO, « Thresholding Using the ISODATA Clustering Algorithm », IEEE Trans. Syst. Man Cybern., vol. 10, no 11, pp. 771-774, 1980.
[63] T. Pun, ''A new method for grey-level picture thresholding using the entropy of the histogram'', Signal Process., vol. 2, no 3, pp. 223-237, juill. 1980.
[64] J. N. Kapur, P. K. Sahoo, A. K. C. Wong, « A new method for gray-level picture thresholding using the entropy of the histogram », Comput. Vis. Graph. Image Process., vol. 29, no 3, pp. 273-285, mars 1985.
[65] H. D. Cheng, J.-R. Chen, J. Li, « Threshold selection based on fuzzy c-partition entropy approach », Pattern Recognit., vol. 31, no 7, pp. 857-870, juill. 1998.
[66] C. H. Li, C. K. Lee, « Minimum cross entropy thresholding », Pattern Recognit., vol. 26, no 4, pp. 617-625, avr. 1993.
[67] J. Bernsen, « Dynamic thresholding of gray level images », in Proceedings of the International Conference on Pattern Recognition (ICPR '86), pp. 1251-1255, 1986.

[68] W. Niblack, An introduction to digital image processing. Englewood Cliffs, N.J: Prentice-Hall International, 1986.
[69] J. Sauvola, M. Pietikᴐinen, « Adaptive document image binarization », Pattern Recognit., vol. 33, no 2, pp. 225-236, févr. 2000.
[70] C. Wolf, J. M. Jolion, F. Chassaing, '' Extraction de texte dans des vidéos : le cas de la binarisation'', Proceedings of 13ème Congrès Francophone de Reconnaissance des Formes et Intelligence Artificielle, pp. 145-152, 2002.
[71] K. Khurshid, I. Siddiqi, C. Faure, N. Vincent, « Comparison of Niblack inspired
[72] binarization methods for ancient documents », présenté à IS&T/SPIE Electronic
[73] Imaging, San Jose, CA, 2009, pp. 724.
[74] B. Gatos, I. Pratikakis, S. J. Perantonis, « Adaptive degraded document image binarization », Pattern Recognit., vol. 39, no 3, pp. 317-327, mars 2006.
[75] [19] S. Lu, B. Su, C. L. Tan, « Document image binarization using background estimation and stroke edges », Int. J. Doc. Anal. Recognit. IJDAR, vol. 13, no 4, pp. 303-314, déc. 2010.
[76] S. J. Lu, C. L. Tan, « Binarization of Badly Illuminated Document Images through Shading Estimation and Compensation », in Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Parana, Brazil, 2007, pp. 312-316.
[77] R. F. Moghaddam, M. Cheriet, « Application of Multi-Level Classifiers and Clustering for Automatic Word Spotting in Historical Document Images », in 2009 10th International Conference on Document Analysis and Recognition, Barcelona, Spain, 2009, pp. 511-515.
[78] Q. Chen, Q. Sun, P. Ann Heng, D. Xia, « A double-threshold image binarization method based on edge detector », Pattern Recognit., vol. 41, no 4, pp. 1254-1267, avr. 2008.
[79] B. Su, S. Lu, C. L. Tan, « Binarization of historical document images using the local maximum and minimum », in Proceedings of the 8th IAPR International Workshop on Document Analysis Systems - DAS '10, Boston, Massachusetts, 2010, pp. 159-166.
[80] M. van Herk, « A fast algorithm for local minimum and maximum filters on rectangular and octagonal kernels », Pattern Recognit. Lett., vol. 13, no 7, pp. 517-521, juill. 1992.
[81] R. Hedjam, R. F. Moghaddam, M. Cheriet, « A spatially adaptive statistical method for the binarization of historical manuscripts and degraded document images », Pattern Recognit., vol. 44, no 9, pp. 2184-2196, sept. 2011.
[82] B. Bataineh, S. N. H. S. Abdullah, K. Omar, « An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows », Pattern Recognit. Lett., vol. 32, no 14, pp. 1805-1813, oct. 2011.
[83] Base de données DIBCO: KHhttp://utopia.duth.gr/~ipratika/DIBCO2013/resources.html
[84] N. Chaki et al., ''Exploring Image Binarization Techniques, Studies in Computational Intelligence'', DOI: 10.1007/978-81-322-1907-1_1, Springer India 2014.
[85] Wan Azani Mustafa et al.,'' Binarization of Document Images: A Comprehensive Review'', International Conference on Green and Sustainable Computing (ICoGeS) 2017
[86] Moghaddam, R.F., Cheriet, M. ''AdOtsu: an adaptive and parameter less generalization of Otsu's method for document image binarization''. Pattern Recogn. 45(6), 2419–2431 (2012).
[87] Gatos, B., Pratikakis, I., Perantonis, S.J.: ''Adaptive degraded document image binarization''. Pattern Recogn. 39(3), 317–327 (2006).
[88] Nancy Chinchor,'' MUC-4 EVALUATION METRICS'', Proceedings of the 4th conference on Message understanding, June 1992 Pages 22–29, https://doi.org/10.3115/1072064.1072067
[89] Fahad Siddiqui, Sam Amiri, Umar Ibrahim Minhas, Tiantai Deng, Roger Woods, Karen Rafferty and Daniel ''Crookes FPGA-Based Processor Acceleration for Image Processing Applications'' : Journal of Imaging 2019, 5, 16, doi:10.3390/jimaging5010016.
[90] Tomasz Kryjak, Mateusz Komorkiewicz Marek Gorgon, ''Real-time hardware–software embedded vision system for ITS smart camera implemented in Zynq SoC''. Journal of Real-Time Image Processing volume 15, pages123–159(2018).
[91] Jie Wei,'' Image segmentation based on situational DCT descriptors'', Pattern Recognition Letters 23(1-3):295-302, elseiver, https://doi.org/10.1016/S0167-8655(01)00124-6.
[92] Felix Scholkmann , Jens Boss, Martin Wo, ''An Efficient Algorithm for Automatic Peak Detection in Noisy Periodic and Quasi-Periodic Signals'', Algorithms MDPI 2012, 5, 588-603, doi:10.3390/a504058
[93] Ziet Lahcene, Khitas Mehdi and Radjah Fayçal, ''Implementation of image histogram for image binarization'', International Conference On Advances In Science, Istambul 31 august-2 september 2016, Istumbul TURKEY.

[94] Hebibi Amar, Arres Bartil, Lahcene Ziet,'' Comparison of two new methods for implementation BPSK modulator using FPGA'', Indonesian Journal of Electrical Engineering and Computer Science, august 2020 DOI: 10.11591/ijeecs.v19.i2.pp819-827.

[95] Taoufik Saidani et al., '' Using Xilinx System Generator for Real Time Hardware Co-simulation of Video Processing System'' Lecture Notes in Electrical Engineering 60:227-236 10.1007/978-90-481-8776-8_20, April 2020.

[96] Kun Zhang, et al, '' Design and Implementation of a Dual-IP Core UAV Flight Control System Based on Qsys'', 3rd International Conference on Mechatronics and Intelligent Robotics (ICMIR-2019), Procedia Computer Science 166 (2020) 180–186, ScienceDirect https://doi.org/10.1016/j.procs.2020.02.045

[97] S. Moslehpour, K. Jenab, Srikar Valiveti, GPS TIME RECEPTION USING ALTERA SOPC BUILDER AND NIOS II: APPLICATION IN VGA,TRAIN POSITIONING Published 10 March 2012, International Journal of Industrial Engineering & Production Research

[98] Marwa Fradi; Wajih Elhadj Youssef; Machout Mohsen, '' The design of an embedded system (SOPC) for an image processing application'', Proceedings of the IEEE International Conference on Control, Automation and Diagnosis (ICCAD), Conference Location: Hammamet, Tunisia, 23 October 2017

[99] Chan Boon Cheng, Asral Bahari Jambek, '' SOC integration for video processing application'' Bulletin of Electrical Engineering and Informatics Vol. 8, No. 1, March 2019, pp. 223~230 DOI: 10.11591/eei.v8i1.1396

[100] DE2 Development and Education Board User Manual, http://www.ece.tufts.edu/~hchang/ee129-f06/project/project2/DE2_UserManual.pdf

[101] Liang Hong-wei, Li Jian-ai, Kan Ling-ling, '' Implementation of SD Card Music Player Using Altera DE2-70'' Computer Science, International Conference on Multimedia and Signal Processing 2011

[102] Embedded Peripherals IP User Guide Updated for Intel Quartus Prime Design Suite: 21.4 ID: 683130UG-01085 Version: 2021.12.13, https://www.intel.in/content/dam/altera-

[103] Khitas Mehdi, Ziet Lahcene, Saad Bouguezl, ''Improved Degraded Document Image Binarization Using Median Filter for Background Estimation'', Elektronika ir Elektrotechnika, DOI: 10.5755/j01.eie.24.3.20982 June 2018.

[104] Phaklen Ehkan et al, ''Artificial Neural Network for Character Recognition on Embedded-Based FPGA'', Lecture Notes in Electrical Engineering 309:281-287, January 2014

[105] Radjah Fayçal, Ziet Lahcene, Benoudjit Nabil, ''ISODATA SOPC-FPGA implementation of image segmentation using NIOS-II processor'', Vol. 22, No. 2, May 2021, pp. 818~825 ISSN: 2502-4752, DOI: 10.11591/ijeecs.v22.i2.pp818-825

[106] www/global/en_US/pdfs/literature/ug/ug_embedded_ip.pdf 12-12-2021

[107] Kun Zhang, et al, '' Design and Implementation of a Dual-IP Core UAV Flight Control System Based on Qsys'', 3rd International Conference on Mechatronics and Intelligent Robotics (ICMIR-2019), Procedia Computer Science 166 (2020) 180–186, ScienceDirect https://doi.org/10.1016/j.procs.2020.02.045