Notes de Cours ELE8307

Olivier Sirois

2017-10-10

Contents

Chapter 1

Cours 1 - Architecture des FPGAs

1946 : ENIAC
plusieurs caractéristique (2 moteurs), occupait beuacoup d'espace. 5000 additions par seconde (ou 40 division par seconde). 385 mul par second, 3 racine carrée par sec..

merveilleux a l'époque.

par contre, c'étais pas simple à programmer, il fallait faire les connections manuellement. C'étais effectivement dans les premiers ordinateurs. un 'bug' était réellement un insecte qui allait se placer dans l'ordinateur, il y en avait en moyenne 1 par jour, et sa pouvait frier toute le système..

C'est avec sa que Von Neumann à eu son idée géniale..

INSÉRÉ SLIDE 6

Par contre, c'étais tellement lourd de changé l'opération entre chaque instruction que Von Neumann à ajouté une nouvelle connection entre le contrôleur de la mémoire (chemin pointillé bleu). C'est de la ou est née l'idée de mettre les instructions en mémoire pour que le contrôleur puisse manuellement aller lire les instructions. C'est de la ou est née le paradigme classique, une section de la mémoire était réservé pour les instructions tandis que le reste était utilisé pour les données.

1.1 Charactéristiques

1.1.1 Niveau sur reconfiguration

Le programme

le niveau le plus élevé sur lequel nous alons travaillé.

jeu d'instruction

un instruction est une série d'opération qui va etre fait entre les registre. Nous allons définir nous même notre jeu d'instruction comme étant des séquences de micro-ops effectié sur les registre/ALUs. sa devient plus lourd à programmer/debugger

fonctions logique de haut-niveau

(ALU, coprocessor). À l'époque, beaucoup de recherche sur arch de niveau moyen, sa revient mais sous la forme de 'coarse-grained' sur FPGA (IP block). On était capable de venir configuré un chemin de donnée auto-synchroniser composé d'ALU.. etc.

on reste toujours au niveau de traitement de donnée.

fonctions logique de bas niveau

(LUT,PLA, porte, transistor). au niveau binaire. On peut changer le comportement des LUT sur les FPGA, (comprend les PLA). Sa peut même aller jusqu'au niveau du transistor. Évidemment, c'est beaucoup plus compliquer sauf que le gain de performance en vaut la chandelle.

Interconnexions

On peut même programmer les interconnexions entre les différentes composante logique de bas niveau. Sa donne une reconfigurabilité presque infini. Juste en interconnectant les transistor de manière astucieuse on a un degrée de configuration infini. Toute la complexité peut se résumer à la complexité de leur interconnexion. Évidemment, les interconnexion programmable prend beaucoup de place.. 80% de la configurabilité du FPGA est dans les interconnexions.

1.1.2 Taille

Le nombre d'élément configurable.. le nom le dit.

On essaie toujours de ramener sa a un équivalent en porte logique (NOR). sauf que c'est de la bouette.. souvent, les compagnies font leur benchmark sur des

circuits qui sont hyperspécifique et qui se fit exactement sur le FPGA. on peut souvent voir un facteur 8 entre ce qu'ils annoncent et ce qui fit vraiment sur le FPGA.

Maintenant on va surement parler d'élément reconfigurable (LUT à 4 entrées.. etc.). C'est beaucoup plus représentatif de la performance/taille. Il faut vraiment avoir une notion poussé du systèmes pour pouvoir le comparés.

1.1.3 Vitesse

le délais, fréquence d'opérations

1.1.4 Caractéristique physique

Les tension d'entrées, d'opérations

1.1.5 Méthode de configuration

protocole de configuration (série, parallèle, maître-esclave, cryptage..).

Aujourd'hui, presque toute les technologies propriétaire est dans le chip, alors souvent les compagnies offrent des options de cryptage pour pouvoir encrypter la mémoire sur le chips pour pouvoir protéger les designs des compagnies (et les votres).
Persistence. Si on coupe le courant est-ce que le circuit reste ?

1.1.6 reconfiguration partielle en temps réelle

Certain FPGA offrent des fonctionnalités ou il peuvent changer certaines zones en opration. C'est comme un circuit mutant qui peut se modifier pendant qu'il roule. Il peut aller chercher des sous-configurations en mémoire et la loader directement sur le fabric

1.1.7 Disponibilité d'IP

On peut avoir accès à des gros blocs de construction IP. C'est bloque sont super complexe en générale, souvent les compagnies peuvent vendre leur bloque pour que les concepteur puissent les utiliser. Souvent, en temps que designer, c'est mieux de pouvoir ce servir de modules déja concus pour accélérer le processus de développement + plus de performance

1.2 Architecture de système numérique

INSÉRÉ SLIDE 8

1.2.1 Mémoire

La mémoire est le circuit combinatoir configurable par excellence, en fait, c'est sa le but.

Avec de la mémoire, on peut théoriquement réalisé n'importe quel circuit combinatoire. En utilisant la mémoire de particulière, on peut créer des effets particulière. On peut utiliser certains bits de la mémoire pour faire des contrôles spéciales (bits D0 = 1 seulement lorsque addr=0x8), cela donne l'équivalent d'un ET logique, vue que l'addresse est un partout. On peut alors, en utilisant la valeur en mémoire, recrée une table de karnaugh dans la mémoire et on utilise la valeur de l'addresse pour faire une sorte de 'query' de la table de Karnaugh.

Avec cette puissante fonctionnalité, on peut recréer n'importe qu'elle fonction logique. n'importe quel circuit combinatoire peut s'implanter avec une mémoire.

Techniquement, aucun calcul n'est réalisé, la valeurdes fonctions est mémoirsé pour chaque combinaison des entrées. On peut alors implémenter:
$D * 2^A$
$A = D + K$

fonctions.

Décodeur texte

On peut voir que c'est magique pouvoir faire un circuit avec de la mémoire, sauf qu'avec l'exemple de décodage, sa prend pas grande chose pour faire un circuit de 659 milles annees lumiere de cotes..

C'est magique lorsque le nombre d'entrées est petits.

En allant plus loins, on peut voir qu'on pourrait travailler lettre par lettre. À ce moment, on peut voir qu'à 10 entrées notre mémoire ne prend que dans les kilobits, ce qui est très manageable.

En changeant le scaling on peut souvent rendre notre design sur une grosseur qui fait en sorte qu'on puisse l'implémenter. Un autre exemple du compromis taille/vitesse.

l'importance de l'architecture ne peut pas être plus mis en évidence. Au niveau circuit on a les dimension:

- x
- y
- temps.

1.2.2 PAL - PLA

INSÉRÉ SLIDE 15

l'idée c'est d'avoir une matrice de somme de signal logique reconfigurable. Avec n'importe quel somme de produit, on peut faire n'importe quel circuit. Une architecture reconfigurable (PLA) est beaucoup plus attirantes étant donnée sa reconfigurabilité pour la sortie. Évidemment, vue qu'il y a deux niveau de reconfigurabilité, c'est un peu plus lent.

On est quand même limité avec ces circuits. On pouvait avoir des modèles purement combinatoire ou des modèls avec des bascules.

INSÉRÉ EXEMPLE DE TABLE DE CARNAUGH

1.2.3 PLD

Après les PAL-PLA, nous avons eu un CPLD. On ne peut pas décrire exactement c'est quoi. l'idée c'est de réutiliser des technologies existantes en rajoutant une matrice d'interconnexion

INSÉRÉ SLIDE 18

Notre programmation est alors très efficaces pour les petits produits, on rajoute alors une notion hiérarchiques dans la programmation de logiques

1.2.4 FPGA

INSÉRÉ SLIDE 19

nous avons en fait le même genre de logique qu'avant, avec les macro-cells qui sont une genre de version modifié d'un PAL-PLA. par contre, avec les interconnexions nous avons un niveau de hiérarchie beaucoup plus élevés.

INSÉRÉ SLIDE 20

Chacune cellules pouvait être utilité comme purement combinatoire ou comme une fonction de mémorisation.

Vue globale

Faire un registre avec des macro-cells c'était assez lourd, sa prenait deux macro-cells pour faire un registre de 32 bits. trop de ressources pour l'importance du registres. Sachant sa, ils ont garder des macro-cells mais en rajoutant des blocks déidés de mémoire. n'importe quel macro-cells était capable de communiquer entre eux et avec des registres mémoire.

INSÉRÉ SLIDE 21, 22, 23 et 24

Une macro-cells (cellule logique). étant en fait un LUT avec un peu de logique additionel (carry chain + cascade chain).

INSÉRÉ SLIDES xc3000

INSÉRÉS SLIDES xc4000

rajout d'une petite LUT pour la XC4000, (logic function of f,g,..). La grande force de Xilinx était de réutiliser les LUTS pour faire de la mémoire utilisable par les utilisateurs. Ils se sont dit qu'avec une cellule logique avec des LUTS, sa empeche pas l'utilisateur de les utilisater comme mémoire, pour pouvoir bénificier sous forme de logique additionel.

SLIDE 32

FPGA récents - Altera

- APEX
  - utilisation simultané de SOP
  - mémoire adressable par contenu
  - IO multitension
  - PLL intégrée pour multiplier la fréquence
  - nombreux standard IO
- Excalibur
  - processeur ARM9 intégrée
- Stratix
  - Blocs DSP
  - mémoire TriMatrix - différentes tailles
  - Adaptive logic module - cellule de calcule combinatoire mystérieuse

  - PLL avancés
  - Adaptive des impédances intégrés
  - IP matériels (interface mémoire, PCIe..)

INSÉRÉ SLIDE 37

Maintenant on commence a retrouver des floating points sous forme de Hard blocks sur les FPGA, ils deviennent tellement gros qu'on peut commencer à voir des résultats intéressant sur FPGA. Les gros compétiteurs des FPGA sont les GPU, depuis quelques années, les GPU sont utilisé par les scientifique pour faire du caclculs de haute performance. Gros combats pour le marché de calculs haute performance.

Un autre fait important est l'impédance intégrés, dépendamment de l'entrée, sa peut faire osciller le courant sur le FPGA ce qui pourrait en causer sa destructions. Étant donnée tous ce qu'on a sur le FPGA, on a pas la place pour rajouter des résistances dans les FPGA. Les producteur ont alors mis des résistances variable qui peuvent être rajuster on the fly pour pouvoir adapter l'indépendance des entrées-sorties. Sa fonctionne..

FPGA récents - Xilinx

- Virtex
  - Présence de mémoires
- Virtex II
  - PLL avancés
  - nombreux IO

FINIR...

nouvelle tendance - SOC

- Altera
  - Cyclone V
  - Stratix 10 SOC
- Xilinx
  - Zynq
  - Zynq (Quad core)

1.3 Cartes utilisé pour les laboratoires

1.3.1 DE2- board

INSÉRÉ DE2-USER MANUAL FIGURE 2.1

Caractéristique du FPGA

- 33k LE
- 105 M4k RAM blocks
- 483k total RAM bits
- 35 embedded multipliers
- 4 PLLs
- 475 user IO pins
- fineline..

la plupart du board est configuré pour le lab.

1.3.2 Cyclone 2 - FPGA sur le DE2

EP2C35*

toute l'information est fournit dans le fichier de documentation du FPGA. l'élément de logique du FPGA ressemble beaucoup à ce qu'il y a dans les figures précédentes. Évidemment, il manque le cascade chain.

les cellules ont les modes normales et arithmétique. arithméqieu - LUT splitter en deux pour le carry.

il faut comprendre aussi qu'il y a deux niveau d'interconnexion (globale / locale). plus on va loin, plus sa prend du temps et plus la fréquence va en etre affecter. Dans le cours, on aura pas besoin de vraiment jouer avec sa.

Il est pratiquement impossible de se rendre à la vitesse maximal d'un FPGA. Il faut littéralement qu'on fait le routage à la main et sa commence à devenir très compliqué.

les blocks M4K de mémoire peuvent avoir des modes différents, les modes sont expliquer dans le fichier de spécifications du FPGA.

Pour la plupart des notions plus avancés, il faut lire la datasheet.

1.3.3 Lab

l'horloge du clavier / fpga ne sont pas synchroniser. le but du lab est de corriger le code VHDL pour pouvoir corriger cette erreur de syncronisation. les erreurs sont causer par manque de synchronisme.

Il va faloir resynchroniser les signaux , signaux de donnée deviennet signaux Dhorloge, sa passe dans une bascule pour pouvoir les resynchroniser. Le but c'Est d'illustrer que c'Est important de travailler en synchronisme avec le FPGA.

1.3.4 VGA

voir la documentation.

Chapter 2

Cours 2 -