Rapport

El421 – Applications avancées des circuits FPGA

Implémentation d'un jeu de ping pong

Students :
Simon Lignac
Steeve Mbock

Professeur :
Richard Salvetat

Promotion 2027 Aéro 4
Lundi 13 avril 2026
Année 2025–2026

Sommaire

Chapitre 1

Introduction

Dans le monde des cartes de calcul, c'est le CPU, ou processeur, qui domine le marché ou du moins qui est ancré dans l'imaginaire collectif. Reconnue pour sa formidable puissance de calcul et sa polyvalence, c'est le premier élément auquel on pense lorsqu'on veut construire une carte électronique. Néanmoins, une concurrente existe, jouant sur la spécialisation plutôt que sur la puissance, ainsi que sur sa capacité à paralléliser les calculs. À l'heure où l'on parle de l'atteinte des limites physiques de la miniaturisation, le FPGA semble être une des solutions à ce problème, et bien plus encore. Une carte FPGA (Field-Programmable Gate Array) est une carte électronique contenant un circuit logique reprogrammable. À la différence d'un microprocesseur classique qui exécute des instructions séquentielles, un FPGA permet de définir directement le matériel : on configure des blocs logiques et des interconnexions pour créer son propre processeur, un filtre numérique, un contrôleur ou tout autre circuit.

Chapitre 2

Prise en main

2.1 Code VHDL

La prise en main de la partie code, notamment sur la manière de construire le code, n'a pas été difficile. En effet, on avait bien compris comment définir une action par `process`, ou encore la déclaration de variable. Pour ce qui est de l'architecture, elle se présente comme suit :

```
projet/
|---COMPONENT/
|   |---declaration entree sortie (physique)
|---declaration des variables/
|
|---Process(1)/
|   |--horloge
|
|---Process(2)/
|   |--commande joueur
|
|---Process(3)/
|   |--dynamique de la balle
|   |--comportement IA
|   |--gestion collision
|---Process(4)/
|   |--affichage
```

2.2 Composants instanciés

Composant	Rôle	Horloge
`clk_wiz_0`	Génère l'horloge pixel de 108 MHz à partir du 100 MHz de la carte	–
`MouseCtl`	Décode le protocole PS/2 et fournit les coordonnées (x, y) de la souris	108 MHz
`MouseDisplay`	Superpose le curseur de souris sur la trame VGA	108 MHz

TABLE 2.1 – Sous-composants du projet

Chapitre 3

Implémentation

3.1 Logique de jeu

Le jeu met en scène deux raquettes (15×100 px) et une balle (16×16 px), affichées en blanc sur fond noir. Une ligne médiane pointillée sépare le terrain.

Contrôle joueur. La raquette droite est déplacée via `BTN_I` à 6 pixels par trame (soit ≈ 360 px/s). Le déplacement est synchronisé sur le `frame_tick` pour garantir une vitesse constante.

Intelligence artificielle. La raquette gauche suit la balle verticalement à 5 px/trame, volontairement plus lente que la balle (8 px/trame horizontal), ce qui laisse une chance au joueur.

Physique de la balle. La vitesse initiale est $(dx, dy) = (8, 6)$ px/trame. Les rebonds sur les bords haut et bas inversent dy ; les collisions avec les raquettes inversent dx avec correction de position. Après sortie d'écran, un délai de 90 trames ($\approx 1{,}5$ s) précède la réapparition au centre.

```
if (ball_dx < 0) and
   (next_x <= rect_x + RECT_W) and
   (next_x + BALL_SIZE >= rect_x) and
   (next_y + BALL_SIZE >= rect_y) and
   (next_y <= rect_y + RECT_H) then
  next_x  := rect_x + RECT_W;
  ball_dx <= -ball_dx;
end if;
```

Listing 3.1 – Détection de collision – raquette gauche (IA)

Chapitre 4

Résultats et conclusion

4.1 Fonctionnalités implémentées

Génération VGA 1280 × 1024 @ 60 Hz stable ;
Balle mobile avec rebonds et réinitialisation automatique ;
Raquette joueur contrôlée par boutons ;
Raquette IA suivant la balle verticalement ;
Détection de collisions avec correction de position ;
Ligne médiane pointillée et curseur souris PS/2.

4.2 Limitations et améliorations possibles

Plusieurs axes d'amélioration ont été identifiés : l'affichage du score (les constantes sont déclarées mais la logique d'incrémentation n'est pas implémentée), le bornage explicite de la raquette joueur pour éviter un dépassement d'écran, l'ajout d'une légère imprécision aléatoire dans l'IA, et l'intégration d'effets sonores via une sortie audio de la carte.

4.3 Conclusion

Ce projet démontre la faisabilité d'un jeu vidéo interactif entièrement implémenté sur FPGA en VHDL, sans processeur ni système d'exploitation. L'architecture purement matérielle traite chaque trame en temps réel à 108 MHz. Les principaux défis relevés sont la synchronisation VGA, la gestion du pipeline de retard et la détection de collisions cohérente. Ce projet constitue une bonne illustration de la conception numérique en logique synchrone et de l'utilisation d'IP cores dans un environnement Xilinx Vivado.