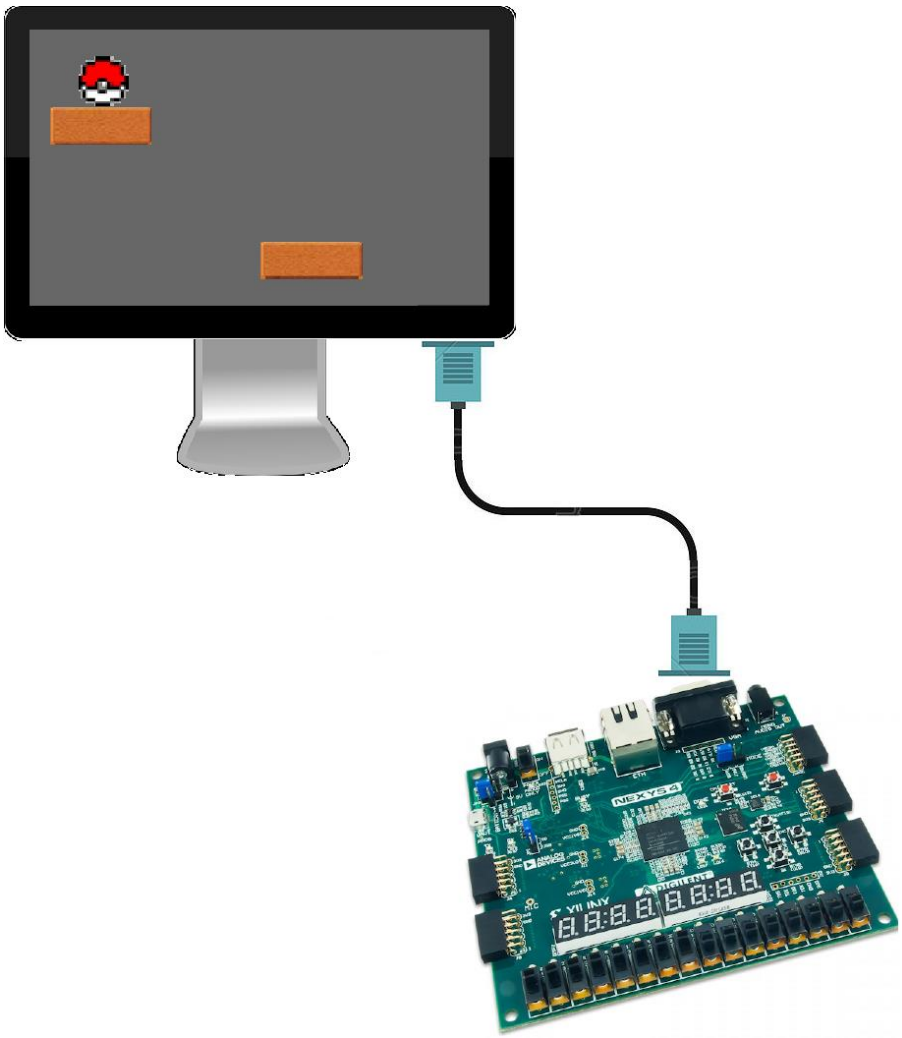
Conception d'un jeu vidéo sur FPGA



1. Présentation du projet

Objectif : concevoir un jeu vidéo sur une board FPGA (Nexys4, XILINX). Le système est décomposé en deux parties : matérielle et logicielle.

La partie matérielle est constituée d'un contrôleur vidéo (VGA), pour la gestion de l'écran de jeu, d'un contrôleur de commande en guise de manette de jeu (accéléromètre, switches et boutons poussoirs). On retrouve également une implémentation SOFT CORE d'un CPU MIPS, qui embarque l'application du jeu. D'autres éléments additionnels sont présents tels que des afficheurs 7segments et des LEDs.

La partie logicielle est embarquée dans le CPU. Elle représente l'application du jeu vidéo développée en langage C. C'est elle qui pilote la partie matérielle.

Description du jeu : le but est de positionner votre balle sur les différents obstacles qui se présentent. La position des obstacles est aléatoire et ils montent au fur et à mesure du bas de l'écran vers le haut. Chaque atterrissage sur un obstacle augmente votre score. Si vous tombez (sortie de l'écran) ou restez sur l'obstacle jusqu'à ce que celui-ci atteigne le haut de l'écran, vous avez perdu. Bonne chance !

2. Architecture matérielle

a. Contrôleur vidéo

Comme son nom l'indique, c'est lui qui est en charge de gérer l'affichage des images sur l'écran. Ces dernières sont stockées dans des mémoires ROM. Le balayage de l'écran se fait pixel par pixel (25MHz). A chaque instant de ce balayage le contrôleur vidéo fait la correspondance entre la position du balayage et la position des images à afficher. S'il y a corrélation, l'image est affichée. A noter qu'il est possible, avec les adaptations nécessaires, d'afficher plusieurs fois la même image à des positions différentes. Trois images sont présentes dans ce jeu ainsi qu'un fond d'écran.

Figure 3 - La balle

Figure 2 - L'obstacle

Figure 1 - Game Over

b. Contrôleur de commande

Le contrôleur de commande fait le lien entre l'utilisateur et le jeu. La partie principale de ce contrôleur est l'accéléromètre. C'est grâce à lui qu'il est possible de mettre en mouvement la balle. Au vu des besoins de l'application, seul l'axe Y est géré (mouvement gauche/droite sur l'écran). Les données provenant de l'accéléromètre sont récupérées par liaison SPI.

Trois switches sont disponibles pour modifier la difficulté du jeu. Ces switches font varier la vitesse de défilement des obstacles.

Deux boutons poussoirs sont disponibles pour exécuter un reset du jeu et pour démarrer une nouvelle partie (après un GAME OVER).

c. Modules additionnels

Des modules supplémentaires viennent s'intégrer au système afin d'apporter plus de fonctionnalités. On dispose de deux afficheurs 7 segments (de 4 digits chacun). L'afficheur de droite affiche le score actuel de la partie, tandis que l'afficheur de gauche affiche le record réalisé (meilleur score).

Des LEDs sont également présentes afin d'afficher les données de l'accéléromètre. Cela ajoute une touche de dynamisme sur la carte.

d. Plasma CPU

Ce qui fait le lien entre la partie matérielle et logicielle est le processeur Plasma, implémenté sur le FPGA. Par le biais de ses GPIO d'entrée et sortie, le CPU peut piloter l'ensemble des éléments présentés précédemment.

Le CPU possède une RAM interne qui contient le bootloader, programme de démarrage. Celui-ci permet d'activer la liaison UART afin de communiquer avec l'utilisateur, avant le démarrage du jeu. Il est alors possible de mettre à jour le logiciel de l'application.

Une RAM externe est utilisée pour stocker l'application du jeu vidéo. A la demande de l'utilisateur, le bootloader redirige le CPU sur cette RAM externe, ce qui lance l'application du jeu.

Mapping GPIO d'entrée :

- **9 à 0** : données de l'accéléromètre (mouvement gauche/droite de la balle)
- **12 à 10** : données des 3 switches pour modifier la vitesse du jeu
- **13** : bouton poussoir pour recommencer une partie (après GAME OVER)

Mapping GPIO de sortie :

- **9 à 0** : positions balle/obstacles (ligne & colonne)
- **12 à 10** : sélection du type de position (ligne balle / colonne balle / ligne obstacles / colonne obstacles)
- **19 à 13** : valeur sur les afficheurs 7segments (score / record)
- **22 à 20** : sélection du 7segments à modifier
- **31** : activation/désactivation de l'affichage de Game Over

Note importante sur la gestion des « sélections du type de position » & « sélection du 7segments à modifier » :

Afin d'éviter tout état intermédiaire des signaux de sélection de position et de 7segments, ces derniers sont codés en code GRAY. De cette manière, entre chaque modification du signal de sélection 1 seul et unique bit varie. Cela permet d'éviter tout état intermédiaire, qui entrainerait des sauts d'affichage des images.

Exemple : sélection affichage balle (10,30), sélection affichage obstacle (50,80). Sans code GRAY l'obstacle prendra la position de la balle (10,30) pendant quelques millisecondes, avant de prendre sa position correct (50,80). Idem concernant les valeurs des 7segments.

Cette architecture permet d'utiliser le moins possible de GPIO de sortie.

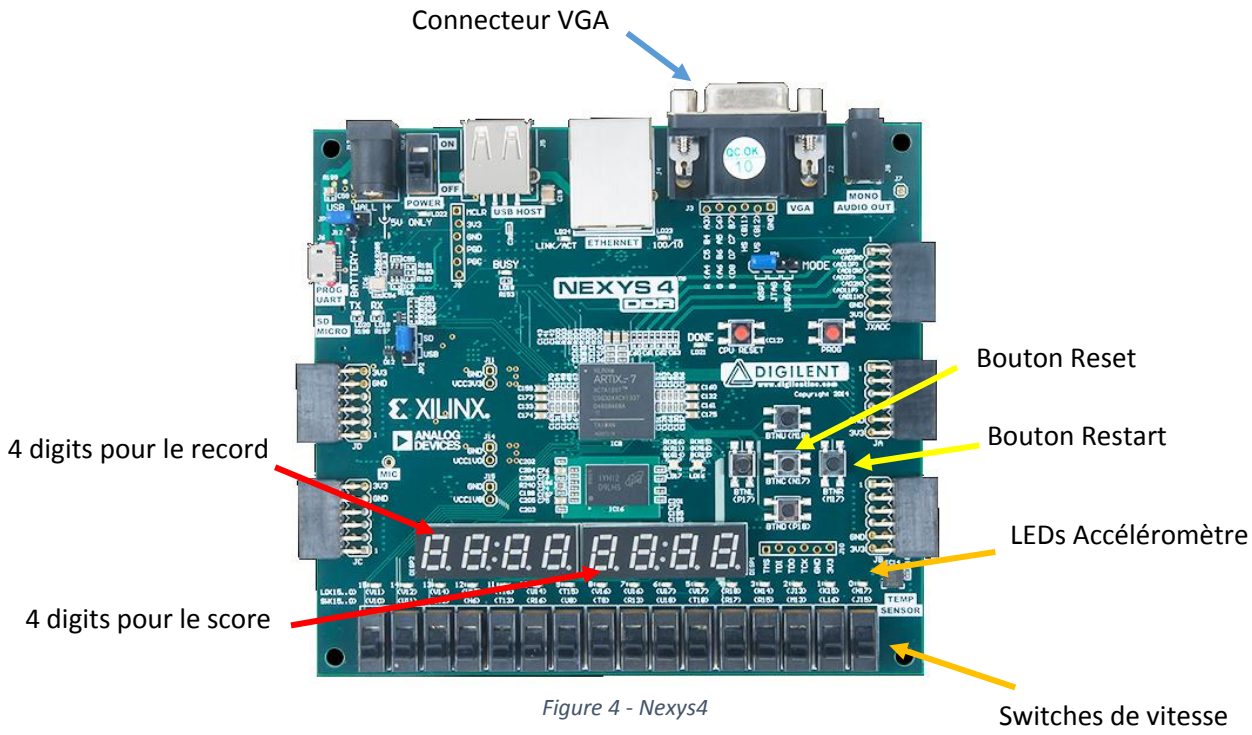


Figure 4 - Nexys4

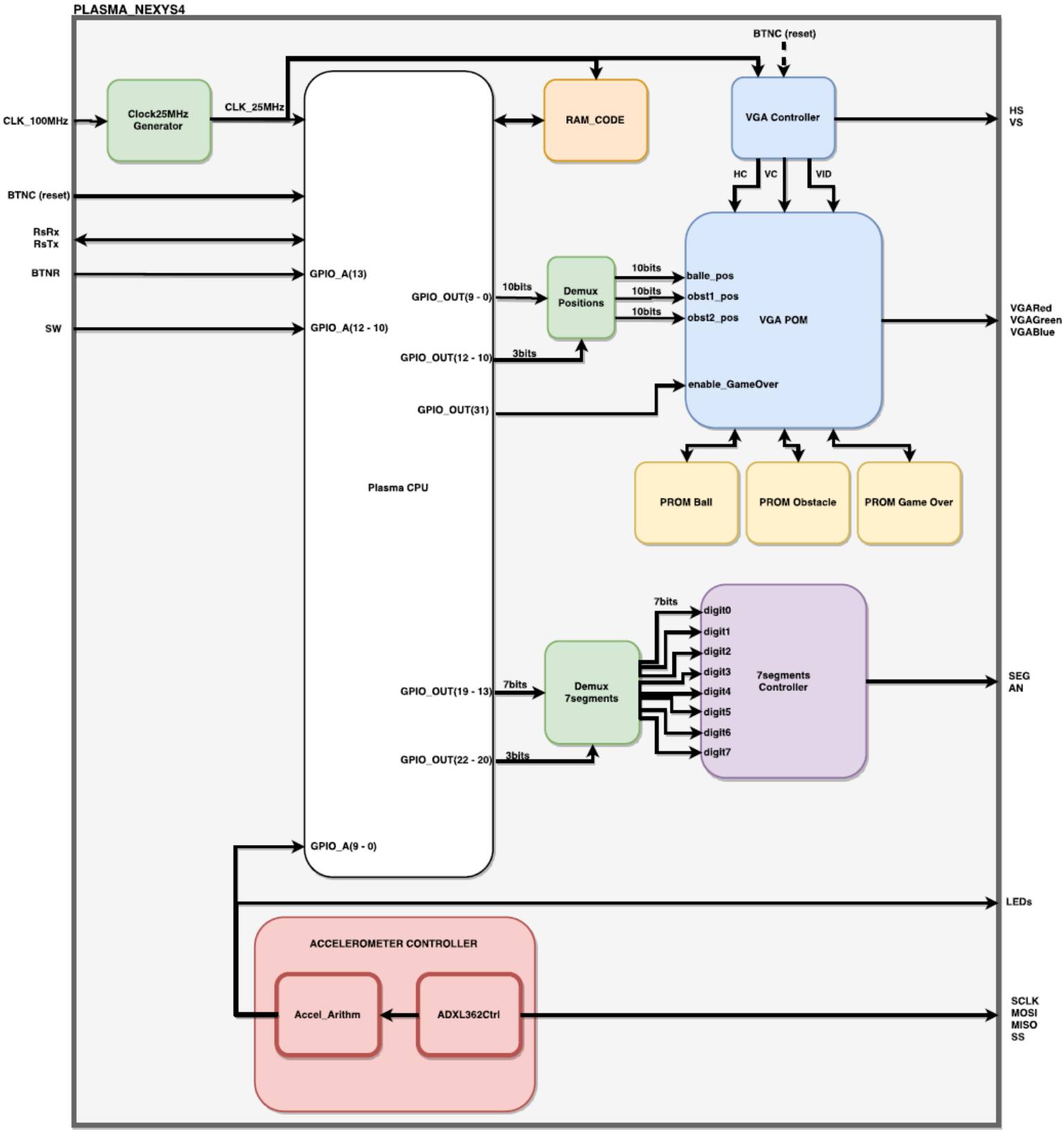


Figure 5 - Architecture matérielle

3. Architecture logicielle

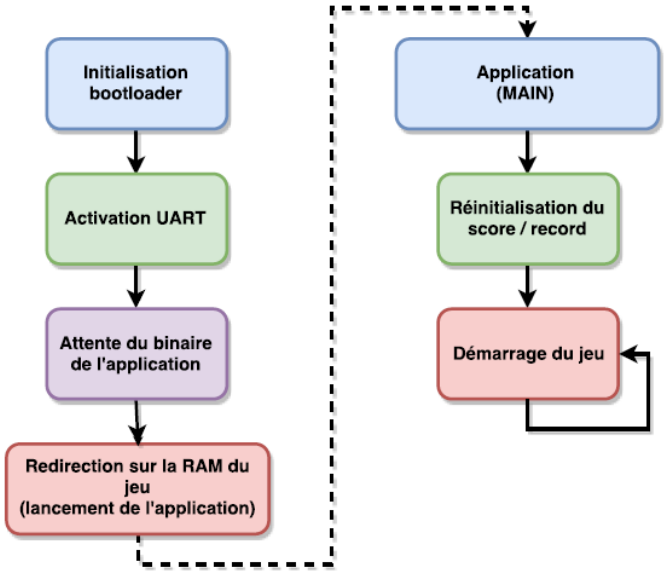


Figure 6 - Procédure de démarrage

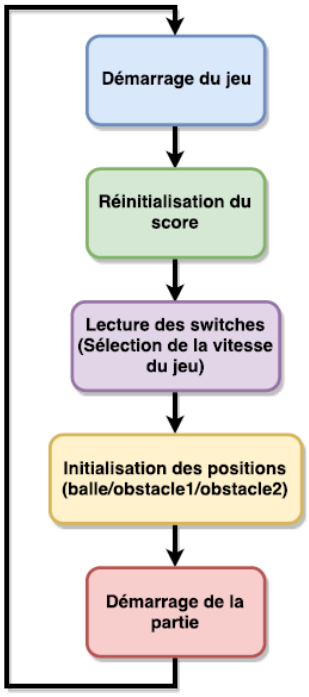


Figure 7 - Lancement du jeu

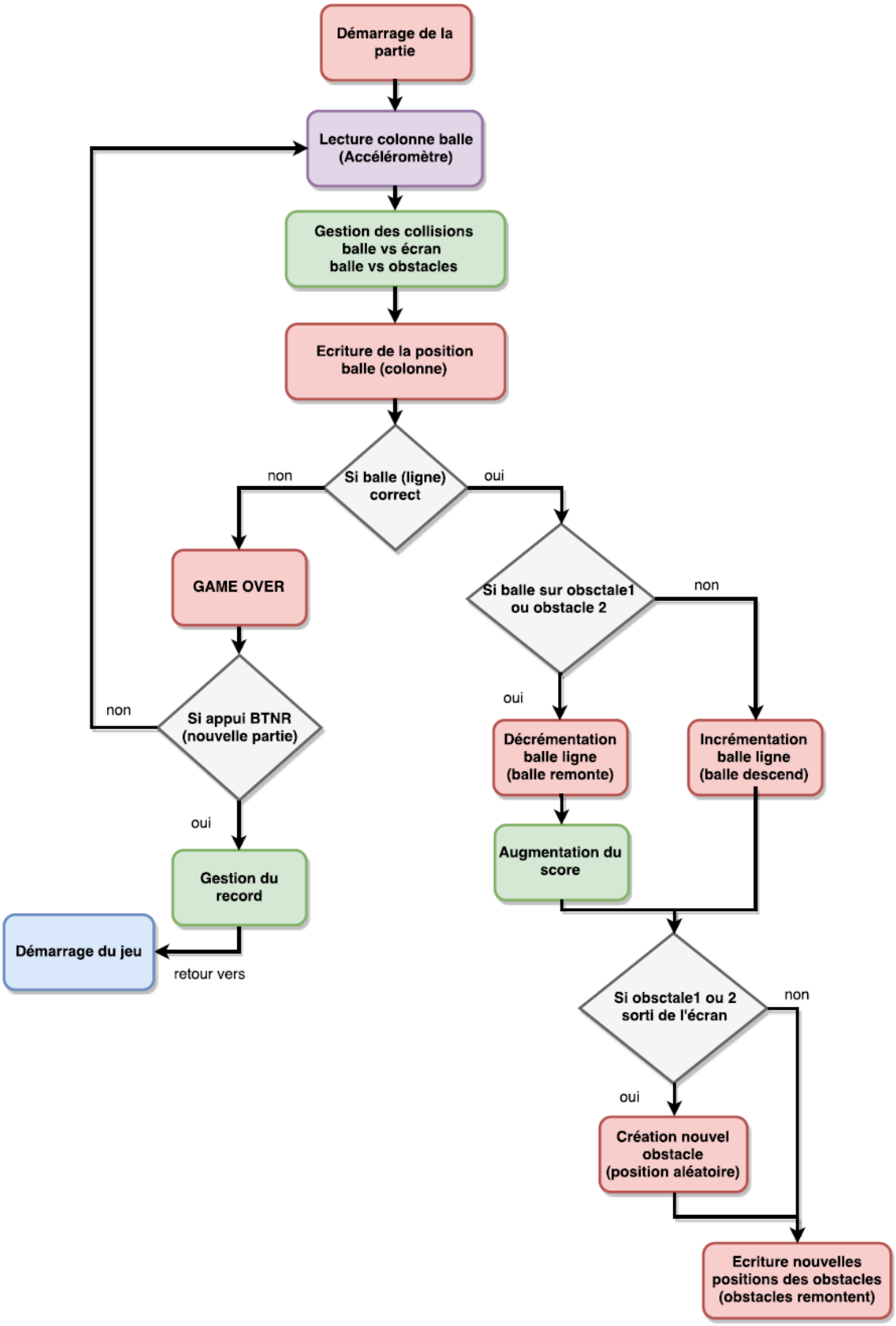


Figure 8 - Organigramme du jeu

4. Améliorations

- Afin d'apporter plus d'option au jeu, il serait intéressant de gérer l'accélération de la balle au cours de sa chute entre les différents obstacles.

- Il serait intéressant d'ajouter de la mobilité aux obstacles. Certains seraient fixes (mouvement vertical uniquement) alors que d'autres seraient mobiles (mouvement horizontal en plus du mouvement vertical).

- Un module audio pourrait également être implanté.