



ARCHITECTURE DES ORDINATEURS

Projet Puissance-4



13 mars 2022
LAU King
GHAMAT Seyedehsetayesh

I. Introduction

L'assembleur MIPS est basé sur une architecture registre-registre, elle ne peut travailler que sur des données dans le banc de registres. MIPS est principalement utilisé dans les consoles de jeux vidéo. C'est pourquoi dans le cadre de notre projet, nous avons essayé d'illustrer au mieux les principes propres de l'assembleur MIPS en implémentant le projet Puissance-4.

II. Puissance-4

1- Règles du jeu :

Puissance 4 est un jeu de stratégie qui se joue à deux. Pour gagner, le joueur doit aligner une suite de 4 jetons de même couleur sur une grille comptant 6 rangées et 7 colonnes. Chaque joueur dispose de 21 jetons d'une couleur. Tour à tour, les deux joueurs placent un jeton dans la colonne de leur choix, le jeton coulisse alors jusqu'à la position la plus basse possible dans la colonne. Le vainqueur est le joueur qui réalise le premier un alignement (horizontal, vertical ou diagonal) consécutif d'au moins quatre jetons de sa couleur. Si, alors que toutes les cases de la grille de jeu sont remplies, aucun des deux joueurs n'a réalisé un tel alignement, la partie est déclarée égalité.

2- Personnalisation du jeu :

Le code initial du jeu nous a été fourni par les enseignants. Celui-ci comprenait la structure des fonctions qu'on devait remplir et la procédure d'affichage du jeu (Grille et jeton). Cependant, quelques fonctionnalités supplémentaires devaient être ajoutées afin de perfectionner le jeu, le rendre plus vivant et original comme avoir la possibilité de relancer le jeu et aussi permettre à chaque joueur de choisir une couleur différente parmi un ensemble proposé via la console.

III. Structure du programme

Au début, nous avons travaillé ensemble sur la procédure *updateRecord* nous avons mis l'ensemble des recherches en commun et choisi les fonctions les plus pertinentes et pour la procédure *winCheck* on a divisé les tâches pour être plus efficace. Par la suite, nous avons défini les parties à traiter par chacun. Setayesh s'est chargée de la partie concernant du relancement du jeu ainsi que la procédure *GameTie* et King s'est chargé de la partie *PlayerWon* et l'option de choisir une couleur de jeton par joueur.

Initialement, il fallait définir les phases par lesquelles passer pour une amélioration optimale du jeu. Pour ce faire, nous avons commencé par faire des schémas pour avoir une vision globale du code. Puis des tests ont été fait pour avoir un résultat réel et déterminer les améliorations à apporter.

Voici quelques résumés généraux sur des fonctions fournies :

La procédure *UpdateRecord*

- Détermine la position exacte où placer le jeton et met à jour l'état du jeu en mémoire, puis renvoi la position de jeton.
- Vérifier si le joueur saisie un chiffre en [1 et 7] .
- Trouver la prochaine ligne vide.
- Ajouter le jeton du joueur dans la grille de jeu.

La procédure *WinCheck*

- Détermine si le dernier jeton joué a permis de gagner.
- Horizontale : Aller vers la gauche, si on est tout à gauche on continue à droite, sinon on va à la position à notre gauche, aller vers la droite, si on est tout à droite on arrête, sinon on va à la position à notre droite.
- Vertical : On va vers le haut, si on est à la dernière ligne en haut on va vérifier en bas, sinon on va dans les positions au-dessus, on va vers le bas, si on est à la dernière ligne en bas on va vérifier en haut, sinon on regarde la position en dessous.
- Slash : On va en haut à droite, si on est à tout en haut ou on est tout à droite alors on va vérifier en bas à gauche, sinon on va en haut à droite, on va en bas à gauche, si on est tout en bas ou tout à gauche alors on arrête de vérifier, sinon on va en bas à gauche.
- Antislash : On va en haut à gauche, si on est à tout en haut ou on est tout à gauche alors on va vérifier en bas à droite, sinon on va en haut à gauche, on va en bas à droite, si on est tout en bas ou on est tout à droite alors on arrête de vérifier, sinon on va en bas à droite
- Égalité : On se met tout en haut du tableau, compteur pour le nombre de jeton de la ligne tout en haut, si dans la ligne il y a une case qui n'a pas un numéro de joueur il n'y a pas égalité, sinon on incrémente le compteur, on se met à droite, si le compteur est égal à 7 alors la ligne tout en haut remplit donc égalité.

La procédure *GameTie*

- Affiche l'égalité et demande une saisie pour recommencer le jeu ou non et vérifie que la saisie est bonne et arrête le jeu.

La procédure *PlayerWon*

- Affiche le gagnant et demande une saisie pour recommencer le jeu ou non et vérifie que la saisie est bonne et arrête le jeu.

Voici quelques résumés généraux sur des fonctions ajouter :

La procédure *Restart*

- Met des 0 dans le tableau et dessine des jetons blancs dans la grille de jeu et relance le jeu.

La procédure *choisirCouleur*

- En amont ajouter des couleurs dans le tableau *Colors* et créer un *tableauCouleur* pour mettre les numéros de couleur choisis.
- Vérifie que la saisie est bonne ensuite lance le jeu

IV. Conclusion

Ce projet a été très enrichissant car nous avons pu réaliser un certain nombre de recherches sur les instructions en MIPS avant même de pouvoir débiter la programmation. Nous avons néanmoins rencontré des difficultés : au début sur le fonctionnement des tableaux en ce qui concerne les lectures et écritures. Par la suite, sur le déplacement dans la grille de jeu et l'effacement du tableau contenant les jetons afin de relancer le jeu. Malgré tout cela, nous avons réussi à finir le projet et grâce au délai supplémentaire accordé, nous avons pu rendre le code plus lisible et y intégrer les commentaires.