Présentation du projet de sysnum

Ryan Lahfa, Constantin Gierczak-Galle, Julien Marquet, Gabriel Doriath Döhler

Introduction

Car c'est notre projet !

Le projet se divise en deux sous-projets :

- Le processeur Minecraft avec l'ISA V-RISC-V[1] ;
- Le processeur RISC-V écrit en System Verilog et simulé avec Verilator

[1] Invention de cerveaux malades.

Plan pour Minecraft

- Motivations

Plan pour Minecraft

- Motivations
- Redstone

Plan pour Minecraft

- Motivations
- Redstone
- ISA

Plan pour Minecraft

- Motivations
- Redstone
- ISA
- Détails d'implémentation

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties
- Prototypes

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties
- Prototypes
- Icarus Verilog $\rightarrow$ Verilator et Verilog $\rightarrow$ System Verilog

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties
- Prototypes
- Icarus Verilog $\rightarrow$ Verilator et Verilog $\rightarrow$ System Verilog
- Caches, MMU

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties
- Prototypes
- Icarus Verilog $\rightarrow$ Verilator et Verilog $\rightarrow$ System Verilog
- Caches, MMU
- Wishbone

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties
- Prototypes
- Icarus Verilog $\rightarrow$ Verilator et Verilog $\rightarrow$ System Verilog
- Caches, MMU
- Wishbone
- Vérification formelle avec SymbiFlow

Plan pour RISC-V

- Fonctionnalités principales du processeur : extensions, entrées-sorties
- Prototypes
- Icarus Verilog $\rightarrow$ Verilator et Verilog $\rightarrow$ System Verilog
- Caches, MMU
- Wishbone
- Vérification formelle avec SymbiFlow
- Contrôleur VGA

Minecraft

Motivations

Minecraft : circuits logiques avec de la redstone[2].

[2] cf slide suivante

Motivations

Minecraft : circuits logiques avec de la redstone[2].

Déjà quelques implémentations existantes de CPU plus ou moins complexes.

[2] cf slide suivante

Motivations

Minecraft : circuits logiques avec de la redstone[2].

Déjà quelques implémentations existantes de CPU plus ou moins complexes.

But : implémenter un CPU 8-bits simple dans Minecraft ; contraintes surtout liées au jeu.

[2] cf slide suivante

Redstone

Redstone : poudre qui, placée au sol, forme des fils. Valeurs : 0 ou 1[3].

[3] Subtilité : il y a des histoires de puissance... Out of the scope pour cette présentation

Redstone

Redstone : poudre qui, placée au sol, forme des fils. Valeurs : 0 ou 1^3.

Un agencement d'éléments (fils de redstone, torches de redstones, blocs, etc.) forme un **circuit logique combinatoire**. Propagation non instantanée : facteur à prendre en compte (naïvement, `>= 0.1` seconde pour qu'un signal parcourt 16 blocs) $\rightarrow$ limitation en taille.

[3] Subtilité : il y a des histoires de puissance... Out of the scope pour cette présentation

Redstone

Redstone : poudre qui, placée au sol, forme des fils. Valeurs : 0 ou 1^3.

Un agencement d'éléments (fils de redstone, torches de redstones, blocs, etc.) forme un **circuit logique combinatoire**. Propagation non instantanée : facteur à prendre en compte (naïvement, `>= 0.1` seconde pour qu'un signal parcourt 16 blocs) $\rightarrow$ limitation en taille.

Quelques timings ajustés et des fonctionnalités de Minecraft permettent de faire des latchs : sauvegarde de données.

[3] Subtilité : il y a des histoires de puissance... Out of the scope pour cette présentation

ISA : V-RISC-V

V-RISC-V = Very Reduced Instruction Set Computer (-V pour le jeu de mot)

Données sur 8 bits, instructions sur 32 bits.

ISA : V-RISC-V

V-RISC-V = Very Reduced Instruction Set Computer (-V pour le jeu de mot)

Données sur 8 bits, instructions sur 32 bits.

- STORE
- LOAD
- ADD
- OR
- XOR
- LOADI
- JMP conditionnel

ISA : V-RISC-V

V-RISC-V = Very Reduced Instruction Set Computer (-V pour le jeu de mot)

Données sur 8 bits, instructions sur 32 bits.

- STORE
- LOAD
- ADD
- OR
- XOR
- LOADI
- JMP conditionnel

```
| pc : 1 | flag : 2 | or,carry,xor : 3 | read1 : 4 |
imm : 0:3 | write : 4 | imm : 4:7 | read2 : 4
```

Pseudo-instructions

Avec les instructions de base et les registres spéciaux :

- NOP
- SUB
- HALT
- PRINT
- JMP (inconditionnel)
- MOV
- NOT
- CMP

Registres

16 general purpose registers : %0 to %15.

Registres

16 general purpose registers : %0 to %15.

Largeur : 8 bits

Registres

16 general purpose registers : `%0` to `%15`.

Largeur : 8 bits

Registres spéciaux :

- `%0 = 0` $\rightarrow$ NOP
- `%1 = -1` $\rightarrow$ NOT
- `%15 = random(0, 255)`

Aspect technique

Conclusion et ouvertures

Achievements :

CPU V-RISC-V avec ROM, registres, ALU, instructions arithmétiques et logiques.

[4] O : afficheurs 7-segments ; I : sélecteurs à leviers

[5] En fait déjà presque possible...

Conclusion et ouvertures

Achievements :

CPU V-RISC-V avec ROM, registres, ALU, instructions arithmétiques et logiques.

TODO :

RAM, I/O utilisateur[4], découpage de l'espace mémoire, pipeline[5]

[4] O : afficheurs 7-segments ; I : sélecteurs à leviers

[5] En fait déjà presque possible...

Le processeur RISC-V (Sakaido, le brillant)

Fonctionnalités principales

Il s'agit d'un processeur RISC-V qui implémente RV32I[6].

[6] RV32IM était disponible à un moment