HORLOGE

Aimeric DUCHEMIN, Amédée LEBERRE, Paul RAPHAEL, Yann Viegas

ENS Ulm

14 janvier 2025

Instruction set

Proposition 1.1 : Conventions

La taille d'un mot est de 32 bits. On représente les addresses sur 16 bits.

Instruction	*Encoding*	*Description*
ADD	01 rs1 rs2 00	rs1 <- $rs1 + $rs2
SUB	02 rs1 rs2 00	rs1 <- $rs1 - $rs2
XOR	03 rs1 rs2 00	rs1 <- $rs1 $\hat{}$rs2
OR	04 rs1 rs2 00	rs1 <- $rs1 | $rs2
AND	05 rs1 rs2 00	rs1 <- $rs1 & $rs2
ADDIMM	45 rs1 rs2 00	rs1 <- imm | $rs1
NOT	06 rs1 rs2 00	rs1 <- $rs2
LSHIFT	07 rs1 rs2 00	rs1 <- $rs1 « $rs2
RSHIFT	08 rs1 rs2 00	rs1 <- $rs1 » $rs2
LOADROM	09 rs1 rs2 00	rs1 <- M[$rs2]
LOADRAM	14 rs1 rs2 00	rs1 <- M[$rs2]

Instruction	*Encoding*	*Description*
MOVIMM	49 rs1 rs2 00	rs1 <- $imm
STORE	2A rs1 rs2 00	M[$rs2] <- $rs1
MOV	0B rs1 rs2 00	rs1 <- $rs2
NONZERO	82 rs1 rs2 00	rs1 <- $rs2
JMP	80 rs1 00 00	PC += $rs1
JMPIMM	81 00 imm	PC += imm
MUL	0D rs1 rs2 00	rs1 <- $rs1 * $rs2

Structure

Le projet est structuré en plusieurs fichiers. alu.py contient les fonctions relatives à l'ALU. registers.py permet de créer et connecter les registres. Le main gère la lecture du code dans la ROM. Enfin, utils.py contient des fonctions telles que gigamux (permettant de faire la sélection de registres, d'instructions etc)

Assembleur

Quelques qualités de l'assembleur :

- Alias de variables
- Sauts intelligents
- Interpréteur d'assembleur

Exemple : Fibonacci

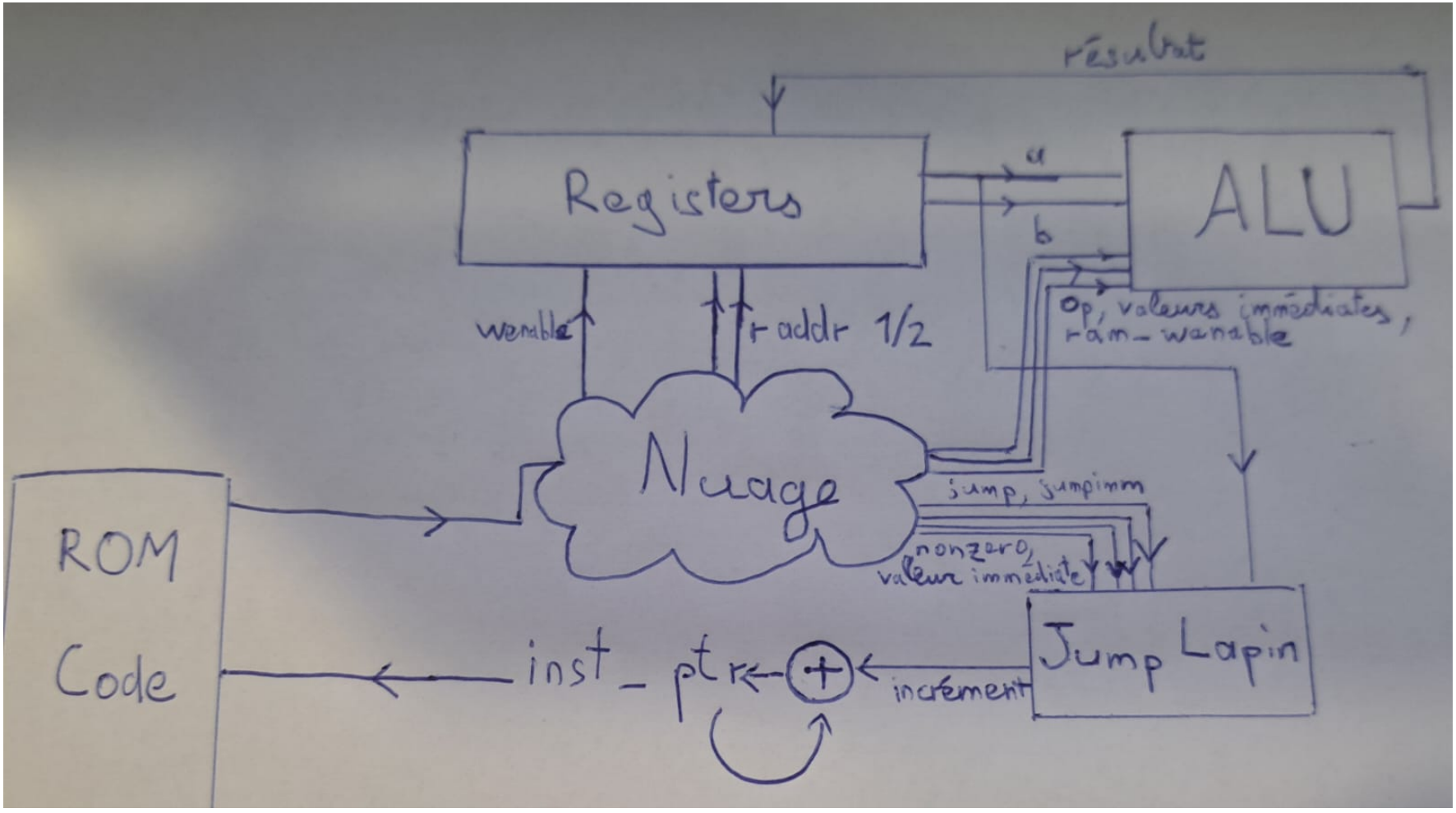
```
MOV rdx $15
MOV rbx $0
MOV rcx $1

MOV r2x $1
fibo :
MOV r1x rcx
ADD rcx rbx
MOV rbx r1x
MOV rax rcx
SUB rdx r2x
NONZERO rdx
JMP 'end
JMP 'fibo

end :
MOV rbx $42

.data
```

Exemple

```
% arg1 = rax
% valDeRetour = rcx

MOV @arg1 $31
MOV rrt 'finFoo->ici
JMP 'foo
ici :
MOV rdx @valDeRetour
JMP 'finProgramme

foo :
# code de la fonction
ADD @valDeRetour @arg1
finFoo :
JMP rrt

finProgramme :
ADD rax $0 # padding

.data
```

Simulateur

Circuit de fonctionnement



L'ALU

Compromis entre :

- Efficacité théorique (profondeur du circuit)
- Efficacité pratique (taille)
- Faisabilité et utilité pour une horloge

Addition rapide

Propagation d'une retenue dans un additionneur carry-lookahead :

$$\begin{array}{cccccc} & & \longleftarrow & & & \\ \ldots & 1 & 1 & 0 & 1 \\ \ldots & 0 & 0 & 1 & 1 \end{array}$$

Profondeur $O(\log n)$ mais taille $O(n^2)$; pour un n-addeur, profondeur et taille $O(n)$.

Multiplication rapide

Multiplication naïve : addition de n nombres.

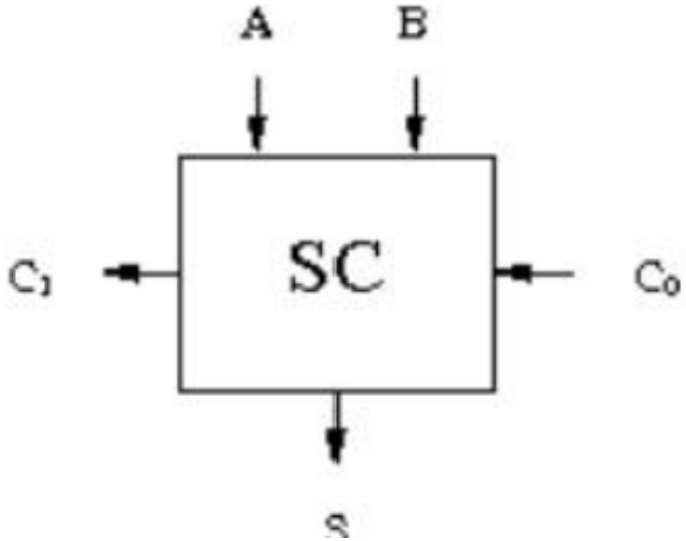


Schéma d'un fulladder.

Une rangée de fulladders multiplie le nombre de bits à additionner par $\frac{2}{3}$.

Profondeur $O(\log n)$, taille $O(n \log n)$ (avec une grosse constante).

Circuit mixte : profondeur $O(n)$, taille $O(n \log n)$.

La Clock : structure

1. Copie des valeurs de ROM dans des registres.
2. Boucle des secondes
    1. Année bissextile (utilse un critère de divisibilité par 25)
    2. Nouvelle date sans changement d'heure
    3. Jour de la semaine (congruence de Zeller)
    4. Changement d'heure
    5. Attente active pour garantir (par exemple) 1000 cycles par seconde

La Clock : critère de divisibilité

Soit $c \in \{0, 1\}$, $a \in N$. Soit d impair.
Alors $2a + c$ est divisible par d ssi $a + 2^{-1}c$ est divisible par d.

La Clock : congruence de Zeller

On considère que janvier et février sont les 13ème et 14ème mois de l'année précédente.
Soit m le mois.
Soit q le jour du mois, $k = annee \mod 100$ et $j = annee/100$ (toutes les divisions sont entières).
Soit

$$h = q + \frac{13(m+1)}{5} + k + \frac{k}{4} + \frac{j}{4} + 5j$$

Alors, le jour de la semaine pour le calendrier grégorien est

$$((h+5) \mod 7) + 1$$

Conclusion


Comment est-ce qu'on commit suicide.py ?
Aimeric1 committed 3 days ago