

Conception de circuits numériques et architecture des ordinateurs

Équipe pédagogique :

Marie Badaroux (TD), Julie Dumas (TD+TP), Florence Maraninchi (TD+TP),
Olivier Muller (TD+TP), **Frédéric Pétrot** (CM+TD), Pierre Ravenel (TD+TP)
Eduardo Tomasi Ribeiro (TP), Lionel Rieg (TD+TP),
Sébastien Viardot (TD+TP Apprentis)



Année universitaire 2024-2025

- C1 Codage des nombres en base 2, logique booléenne, portes logiques, circuits combinatoires
- C2 Circuits séquentiels
- C3 Construction de circuits complexes
- C4 Micro-architecture et fonctionnement des mémoires
- C5 Machines à état
- C6 Synthèse de circuits PC/PO
- C7 Optimisation de circuits PC/PO
- C8 Interprétation d'instructions - 1
- C9 Interprétation d'instructions - 2
- C10 **Interprétation d'instructions - 3**
- C11 Introduction aux caches

Plan détaillé du cours d'aujourd'hui

1 Processeur et jeu d'instruction

- Introduction
- Présentation de l'ISA RISC-V
- Formats et exemples
- Implantation PC/PO abstraite
- Pour conclure

Plan

1 Processeur et jeu d'instruction

- Introduction
- Présentation de l'ISA RISC-V
- Formats et exemples
- Implantation PC/PO abstraite
- Pour conclure

Introduction

Rappel de l'épisode précédent

Implantation PC/PO de machines à pile et à accumulateur

Modes d'adressage

Aujourd'hui un exemple détaillé du processeur RISC-V

Architecture inspirée du MIPS, symptomatique de la mouvance RISC
décrite dans *Computer Architecture, a Quantitative Approach*

David Patterson et John Hennessy (Turing award 2017)

Fort intérêt dans la communauté matérielle et open-source

Caractéristiques globales de l'architecture

- bus de données et d'adresse sur 32 bits
- taille de l'instruction : 32 bits
- ISA 3-adresses
- 32 registres opérandes des instructions, noté x0 à x31
 - registres x1 à x31 : usage général
 - registre x0 : vaut toujours 0
- registre pc : adresse de l'instruction
- registre ir : instruction en cours d'exécution

Exemples d'instructions

Quelques instructions assembleur

Entre registres

<code>add x2, x3, x5</code>	$x2 := x3 + x5$
-----------------------------	-----------------

Avec une constante sur 12 bits signée (0xfffff800-0xffffffff, 0x000-0x7ff)

<code>addi x2, x3, 0xfffffbad</code>	$x2 := x3 + -1107$
--------------------------------------	--------------------

Branchement conditionnel

<code>beq x5, x9, étiquette</code>	$pc := \begin{cases} \text{étiquette} & \text{si } x5 = x9 \\ pc + 4 & \text{sinon} \end{cases}$
------------------------------------	--------------------------------------------------------------------------------------------------

Chargement à partir de la mémoire

<code>lw x1, 24(x2)</code>	$x1 := mem_4[x2 + 24]$
----------------------------	------------------------

Branchement inconditionnel

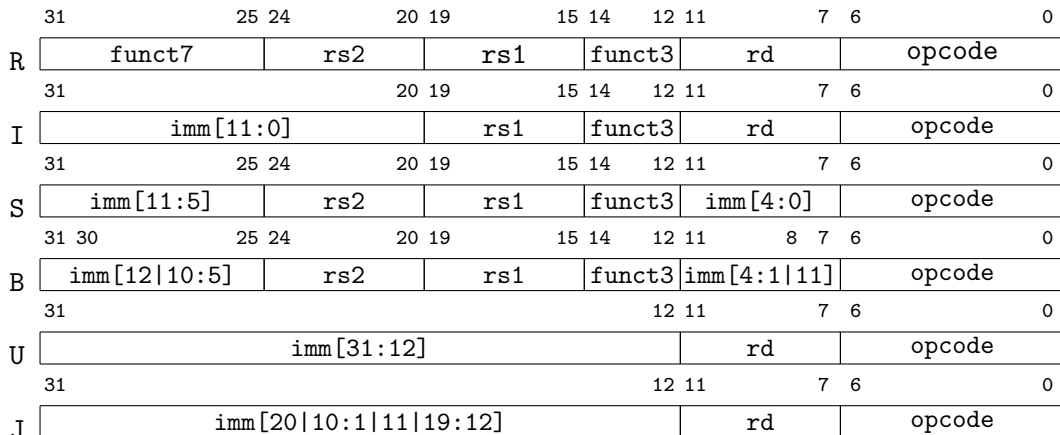
<code>jal x1, étiquette</code>	$x1 := pc + 4, pc := \text{étiquette}$
--------------------------------	----------------------------------------

Modes d'adressage

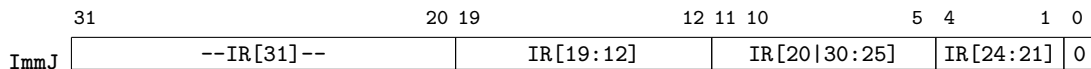
3 modes d'adressage

- *registre*
5 bits dans l'instruction
- *immédiat*
5 bits dans l'instruction, pour les décalages
12 bits signés dans l'instruction, différentes interprétations possibles
20 bits dans l'instruction
- *indirect registre (+ déplacement)*
 - 5 bits pour identifier le registre + 12 bits pour la constante
 - relatif à pc (plusieurs cas possibles)

Format des instructions



Construction des immédiateurs



Exemple détaillé d'instructions

Forme générique de quelques instructions de type R

add rd, rs1, rs2

$$r_d := r_{s1} + r_{s2}$$

sub rd, rs1, rs2

$$r_d := r_{s1} - r_{s2}$$

Codage

add x14, x23, x21

0x015b8733

sll x9, x24, x9

0x009c14b3

Exemple détaillé d'instructions

Forme générique de quelques instructions de type I

addi rd, rs1, imm12

beq rs1, rs2, imm12

lw rd, imm12(rs1)

sh rs2, imm12(rs1)

$$r_d := r_{s1} + \mathbf{ImmI}$$

$$pc := \begin{cases} pc + 4 & \text{si } r_{s1} \neq r_{s2} \\ pc + \mathbf{ImmB} & \text{sinon} \end{cases}$$

$$r_d := mem_4[r_{s1} + \mathbf{ImmI}]$$

$$mem_2[r_{s1} + \mathbf{ImmS}] := r_{s2}[15 : 0]$$



Attention, la construction des immédiate varie selon l'instruction

Codage

addi x10,x26,47	0x02fd0513
0x14d0 : bne x14,x15,0x14dc	0x00f71663

Exemple détaillé d'instructions

Forme générique de l'unique instruction de type J

`jal rd, imm20`

$rd := pc + 4; pc \leftarrow pc + \mathbf{ImmJ}$



Si `rd` est `x0`, c'est un simple saut

Codage

`0x16fc: jal x0,0x17b4`

`0x0b80006f`

`0x26d0: jal x1,0x2648`

`0xf79ff0ef`

Résumé de l'ISA du RISC-V

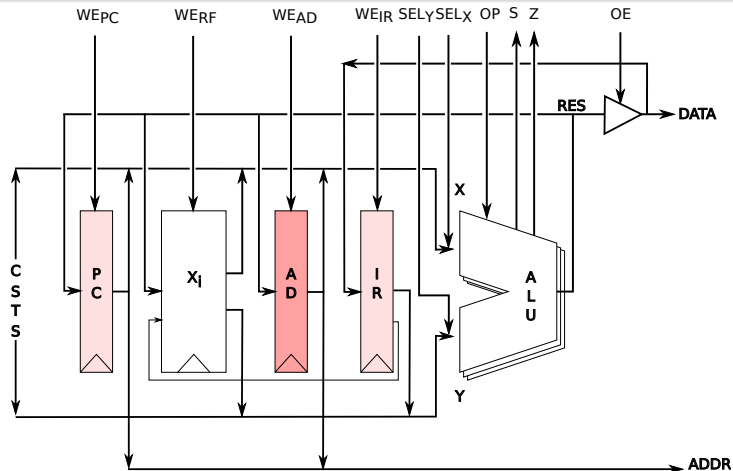
Construction d'un chemin de données « abstrait » sur lequel ces instructions peuvent être exécutées.

- Registres implicites : pc , ir
- Banc de 32 registres, x_i , à 2 accès en lecture au même cycle
- Constantes : adresse du reset, incrément de pc , ...
- Registre invisible du logiciel : AD contient l'adresse effective nécessaire aux *load* et *store*
- Unité de calcul : ALU

Résumé de l'ISA du RISC-V

- Utilisation de ir :
 - Dépend totalement du format de l'instruction
 - Immédiats :
 - sur 5 bits pour les décalages
 - sur 12 bits étendu de signe à 32 bits, multiplié par 2 pour les sauts, tel quel sinon
 - sur 20 bits étendu de signe à 32 bits multiplié par 2 pour les sauts, tel quel sinon
 - construits de manière très « surprenante » !
 - Champs r_d , r_{s1} et r_{s2} directement utilisés pour indiquer la destination et les opérandes au banc de registres

Partie opérative simplifiée

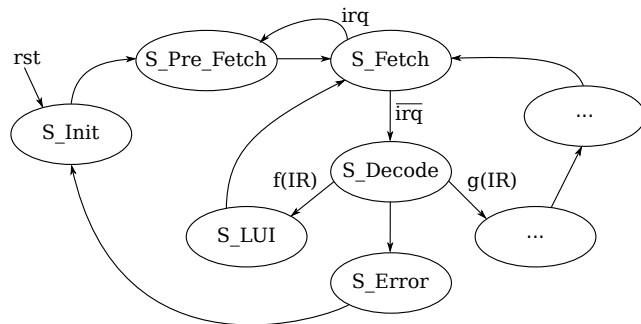


Simplifications



Principe très abstrait qui sera raffiné en TD

Survol de la partie contrôle simplifiée



Simplifications



Principe très abstrait qui sera raffiné en TD

Ajouts nécessaires pour être utilisable

Interruptions et exceptions

- ensemble de registres spécifiques, les *control and status registers*
- si exception, inhibition de l'écriture dans les registres généraux
- testées identiquement au *reset*, avant exécution instruction
- sauvegarde de l'adresse de l'instruction à laquelle reprendre après le traitement
- saut à une adresse stockée dans le csr *mtvec* à laquelle se trouve le gestionnaire

4 modes de fonctionnement

- machine : on peut tout faire, e.g. bios, grub, uboot, ... *superbaremetal*
- superviseur : on peut presque tout faire, e.g. Linux tourne dans ce mode
- utilisateur : processus classique d'un OS (applis)
- hyperviseur : on se retrouve en 3A/master pour causer Xen ou Virtualbox

Hall of fame

Quelques processeurs 8 bit PC/PO ayant marqué leur époque : 75-85

- Intel 8080
- Zilog Z80
- Motorola 6800
- MOSTech 6502

Quelques processeurs 16 et ou 32 bit PC/PO ayant marqué leur époque : 75-89

- DEC PDP-11^a
- Intel 8086
- Motorola 68000

a. Machine sur laquelle Unix et le langage C ont été développés