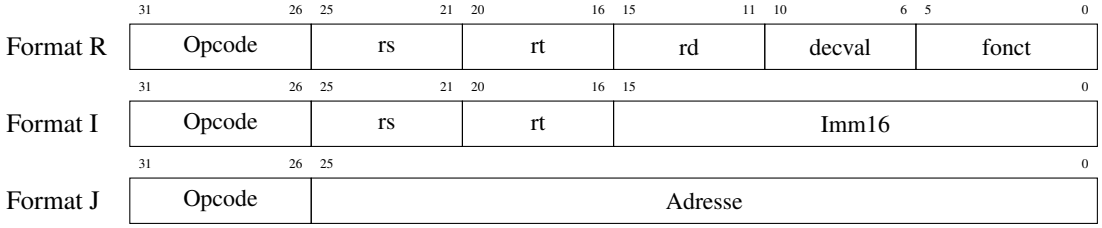
Feuille de travaux dirigés n° 6
Instructions et chemins de données

Instructions

Exercice 6.1

On rappelle les formats pour les instructions du MIPS :



Le processeur MIPS a 32 registres généraux. Les codes (opcode/foncteur) des instructions utilisées dans cet exercice sont présentés dans le tableau ci-dessous :

Opération	Mnémonique	Format	Opcode	Foncteur
Addition	`add`	R	000000	110010
`add rd, rs, rt` : $R[r_d] \leftarrow R[r_s] + R[r_t]$				
Addition Immédiat	`addi`	I	001000	
`addi rt, rs, Imm` : $R[r_t] \leftarrow R[r_s] + \text{SignExt(Imm)}$				
Soustraction	`sub`	R	000000	110100
`sub rd, rs, rt` : $R[r_d] \leftarrow R[r_s] - R[r_t]$				
Chargement	`lw`	I	100011	
`lw rt, Imm(rs)` : $R[r_t] \leftarrow \text{MEM}\,[R[r_s] + \text{SignExt(Imm)}]$				
Rangement	`sw`	I	101011	
`sw rt, Imm(rs)` : $\text{MEM}\,[R[r_s] + \text{SignExt(Imm)}] \leftarrow R[r_t]$				
Décalage à droite	`srl`	R	000000	000010
`srl rd, rt, decval` : $R[r_d] \leftarrow R[r_t] >> \text{decval}$				

1. Quel est l'intervalle maximal pour l'opérande immédiat de `addi` ?
2. Quel est l'intervalle maximal pour les pas de décalage de l'instruction `srl` ? Justifier ;
3. Quelles sont les instructions qui correspondent aux codages suivants :
   `0x23bdffe0`
   `0xafa50024`
   `0x8fae001c`
4. Quels seraient les codages en binaire et en hexadécimal des instructions suivantes :

```
sw $ra, 20($sp)
sw $fp, 16($sp)
addi $fp, $sp, 32
lw $v0, 0($fp)
srl $v0, $v0, 4
add $v0, $a0, $v0
sub $v0, $v0, $a1
```

Exercice 6.2

On considère une machine à pile M_0, une machine à accumulateur M_1 et une machine à registres M_2. Ces machines disposent des instructions décrites ci-dessous :

M_0	M_1	M_2
`add`	`load X` (acc $\leftarrow X$)	`load Ri, X`
`sub`	`store X` ($X \leftarrow$ acc)	`store Ri, X`
`push X`	`add X` (acc $\leftarrow$ acc $+ X$)	`add Ri, Rj, Rk` ($R_i \leftarrow R_j + R_k$)
`pop X`	`sub X` (acc $\leftarrow$ acc $- X$)	`sub Ri, Rj, Rk` ($R_i \leftarrow R_j - R_k$)

1. Écrire les séquences de code suivantes pour les machines M_0, M_1 et M_2 (on suppose que les variables `A`, `B`, `C` et `D` sont initialement en mémoire :

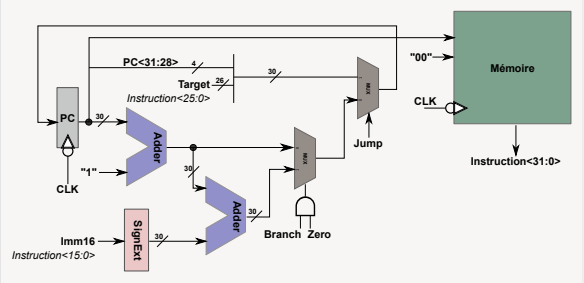
`A = B + C`	`A = B + C`
	`B = A + C`
	`D = A - B`

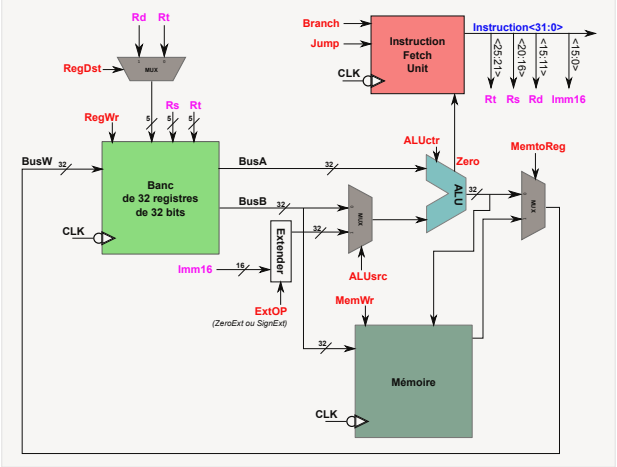
2. Comparer ces machines en taille de code statique et en nombre d'accès à la mémoire.

Chemins de données

Exercice 6.3

On souhaite ajouter au chemin de données mono-cycle vu en cours et rappelé ci-dessous l'instruction `jal`.





1. Ajouter au chemin de données tous les éléments nécessaires à la gestion de `jal` (lignes de données, modules combinatoires/séquentiels, signaux de contrôle) ;
2. Étendre la table ci-dessous de façon à tenir compte des signaux de contrôle ajoutés.

Foncteur	100000	100010	*Don't care*				
Opcode	000000	000000	001101	100011	101011	000100	000010
	`add`	`sub`	`ori`	`lw`	`sw`	`beq`	`j`
`RegDst`	1	1	0	0	x	x	x
`ALUSrc`	0	0	1	1	1	0	x
`MemtoReg`	0	0	0	1	x	x	x
`RegWrite`	1	1	1	1	0	0	0
`MemWrite`	0	0	0	0	1	0	0
`Branch`	0	0	0	0	0	1	0
`Jump`	0	0	0	0	0	0	1
`ExtOp`	x	x	0	1	1	x	x
`ALUctr`	Add	Sub	Or	Add	Add	Sub	xxx