ISA Groupe G

Pierre-Alexandre Bazin, Mathis Bouverot & Arthur Rousseau

1 Mémoire

La RAM et la ROM sont sur 2^{16} bits, avec donc des adresses sur 16 bits. Les entiers manipulés seront toujours signés et sur 16 bits.

2 Registres

Il y a 8 registres nommés rax, rbx, rcx, rdx, rex, rfx, rhx, rz (registre nul).
Ces registres sont sur 16 bits.
Codage des registres :

rz	rax	rbx	rcx	rdx	rex	rfx	rhx
000	001	010	011	100	101	110	111

3 Instructions

Les instructions sont sur 32 bits. Leur schéma est le suivant :

`imm`	`funct7`	`funct3`	`rs2`	`rs1`	`rd`	`opcode`

avec

- `opcode` sur 3 bits
- `funct3` sur 3 bits
- `funct7` sur 1 bit
- `rd`, `rs2`, `rs1` sur 3 bits (8 registres)
- `imm` sur 16 bits

Instructions possibles :

Inst	Nom	Opcode	funct3	funct7	Description
`add`	ADD	011	000	0	`rd = rs1 + rs2`
`sub`	SUB	011	000	1	`rd = rs1 - rs2`
`or`	OR	011	100	0	`rd = rs1 or rs2`
`nand`	NAND	011	100	1	`rd = rs1 nand rs2`
`xor`	XOR	011	001	0	`rd = rs1 xor rs2`
`nxor`	NXOR	011	001	1	`rd = rs1 nxor rs2`
`and`	AND	011	101	0	`rd = (rs1 and rs2)`
`nor`	NOR	011	101	1	`rd = rs1 nor rs2`

sll	Shift left logical	011	011	0	rd = rs1 « rs2
srl	Shift right logical	011	010	0	rd = rs1 » rs2
sra	Shift right Arith	011	010	1	rd = rs1 » rs2
seq	Set equal	011	111	1	rd = (rs1 = rs2)?1:0
slt	Set less than	011	110	1	rd = (rs1 < rs2)?1:0
addi	ADD (immediate)	001	000	0	rd = rs1 + imm
subi	SUB (immediate)	001	000	1	rd = rs1 - imm
ori	OR (immediate)	001	100	0	rd = rs1 or imm
nandi	NAND (immediate)	001	100	1	rd = rs1 nand imm
xori	XOR (immediate)	001	001	0	rd = rs1 xor imm
nxori	NXOR (immediate)	001	001	1	rd = rs1 nxor imm
andi	AND (immediate)	001	101	0	rd = rs1 and imm
nori	NOR (immediate)	001	101	1	rd = rs1 nor imm
slli	Shift left logical (immediate)	001	011	0	rd = rs1 « imm
srli	Shift right logical (immediate)	001	010	0	rd = rs1 » imm
srai	Shift right Arith (immediate)	001	010	1	rd = rs1 » imm
seqi	Set equal (immediate)	001	111	1	rd = (rs1 = imm)?1:0
slti	Set less than (immediate)	001	110	1	rd = (rs1 < imm)?1:0
lw	Load word	000	000	0	rd=M[rs1]
sw	Store word	010	000	0	M[rs1]=rs2
beq	Branch ==	110	100	1	if(rs1 == rs2) PC=imm
bne	Branch !=	110	101	1	if(rs1 != rs2) PC=imm
ble	Branch ⩽	110	110	1	if(rs1 <= rs2) PC=imm
blt	Branch <	110	010	1	if(rs1 < rs2) PC=imm
bge	Branch ⩾	110	011	1	if(rs1 >= rs2) PC=imm
bgt	Branch >	110	111	1	if(rs1 > rs2) PC=imm
jal	Jump and link	111	000	0	rd=PC+4; PC=imm
jalr	Jump and link reg	101	000	0	rd=PC+4; PC=rs1+imm
jr	Jump reg	100	000	0	PC=rs1+imm
jmp	Jump	110	001	1	PC=imm

On ajoute du sucre syntaxique:

Inst	Nom	Description
mov	MOV	rd=rs1+0
movi	MOV (immediate)	rd=imm+0

4 Interface

Le simulateur ne pouvant fonctionner à la fréquence d'horloge, on pourra le synchroniser avec celle-ci, en le faisant tourner à un sous-multiple de la vitesse d'horloge.
On ajoute une fonction `_print` qui permettra d'afficher le contenu d'un registre (entier). Celle-ci permettra d'afficher l'horloge.