

# Code binaire

101001010100111101000010010111010010  
001000010100101001001010000101101001010100001111010010101001110100001001011010010  
1101010101011101000010010100100101001001001001001001010100101010000111010010101



# Codage des instructions : exemple

*4 octets*

ldr rd, étiquette

0 0	@donnée	# rd
-----	---------	------

str rs, étiquette

0 1	@donnée	# rs
-----	---------	------

add rd, rs1, rs2

1 0	# rs1	# rs2	# rd
-----	-------	-------	------

add rd, rs1, #constante

1 1	# rs1	cste	# rd
-----	-------	------	------

sub rd, rs1, rs2

1 2	# rs1	# rs2	# rd
-----	-------	-------	------

cmp rs1, rs2

1 3	# rs1	# rs2	0 0
-----	-------	-------	-----

mov rd, #constante

1 4	0 0	cste	# rd
-----	-----	------	------

b étiquette

2 0	@instruction	0 0
-----	--------------	-----

bge étiquette

2 1	@instruction	0 0
-----	--------------	-----

# Décodage des instructions

	31	24	23	16	15	8	7	0
ldr rd, étiquette	0	0	@donnée				#	rd
str rs, étiquette	0	1	@donnée				#	rs
add rd, rs1, rs2	1	0	#	rs1	#	rs2	#	rd
add rd, rs1, #valeur	1	1	#	rs1	valeur	#		
sub rd, rs1, rs2	1	2	#	rs1	#	rs2	#	rd
cmp rs1, rs2	1	3	#	rs1	#	rs2	0	0
mov rd, #valeur	1	4	0	0	valeur	#		
b étiquette	2	0	@instruction				0	0
bgt étiquette	2	1	@instruction				0	0

addition :

26 25 24

0 0 0 1 0 0 0 0 ... ..

0 0 0 1 0 0 0 1 ... ..

soustraction :

26 25 24

0 0 0 1 0 0 1 0 ... ..

comparaison (=soustraction) :

26 25 24

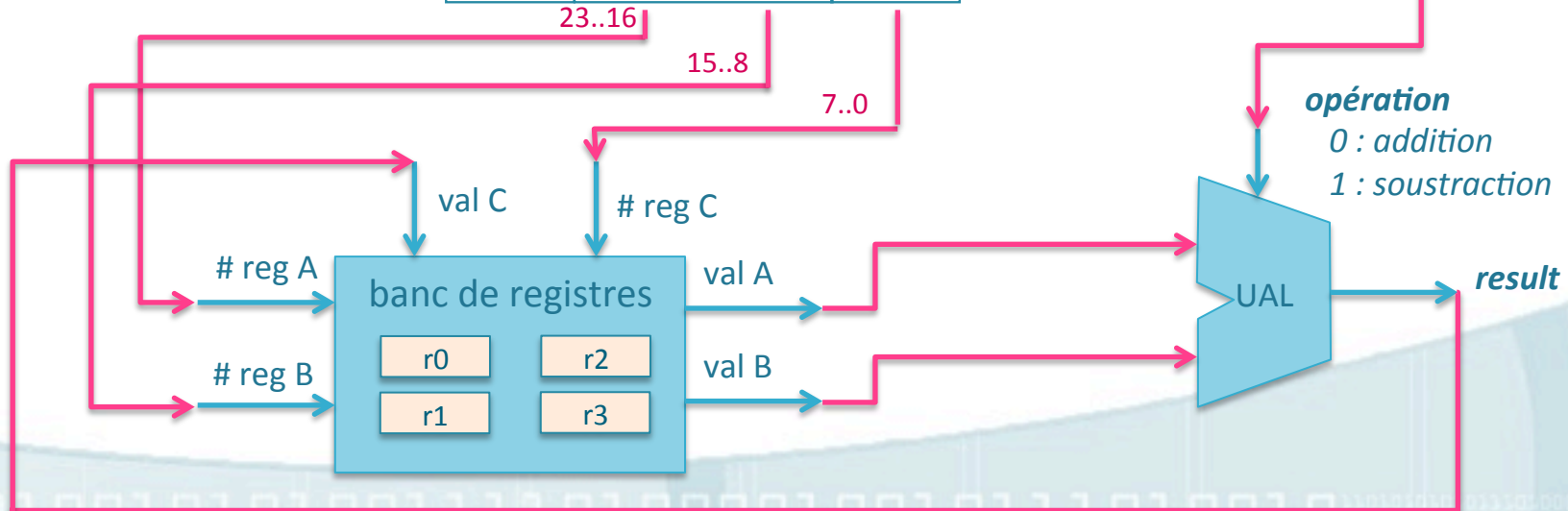
0 0 0 1 0 0 1 1 ... ..

*opération*  
0 : addition  
1 : soustraction



# Décodage des instructions

	31	24	23	16	15	8	7	0
ldr rd, étiquette	0	0	@donnée				#	rd
str rs, étiquette	0	1	@donnée				#	rs
add rd, rs1, rs2	1	0	#	rs1	#	rs2	#	rd
add rd, rs1, #valeur	1	1	#	rs1	valeur	#		
sub rd, rs1, rs2	1	2	#	rs1	#	rs2	#	rd
cmp rs1, rs2	1	3	#	rs1	#	rs2	0	0
mov rd, #valeur	1	4	0	0	valeur	#		
b étiquette	2	0	@instruction				0	0
bgt étiquette	2	1	@instruction				0	0



addition :

26 25 24

0 0 0 1 0 0 0 0 ... ..

0 0 0 1 0 0 0 1 ... ..

soustraction :

26 25 24

0 0 0 1 0 0 1 0 ... ..

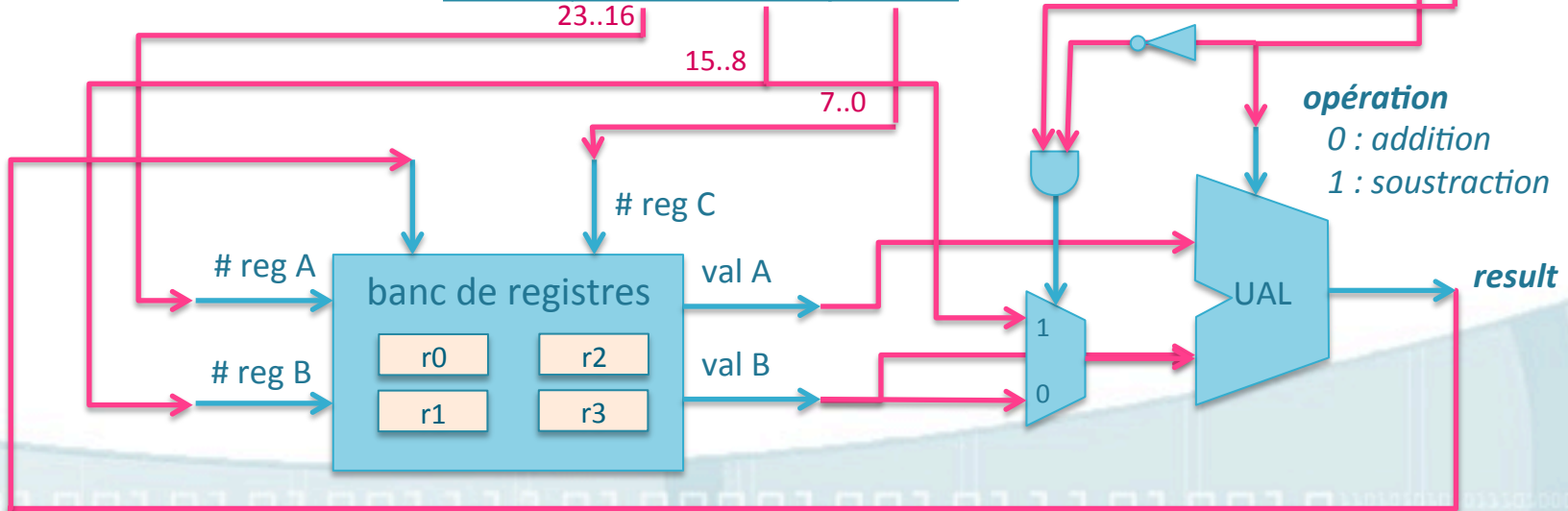
comparaison (=soustraction) :

26 25 24

0 0 0 1 0 0 1 1 ... ..

# Décodage des instructions

	31	24	23	16	15	8	7	0
ldr rd, étiquette	0	0	@donnée				#	rd
str rs, étiquette	0	1	@donnée				#	rs
add rd, rs1, rs2	1	0	#	rs1	#	rs2	#	rd
add rd, rs1, #valeur	1	1	#	rs1	valeur	#		rd
sub rd, rs1, rs2	1	2	#	rs1	#	rs2	#	rd
cmp rs1, rs2	1	3	#	rs1	#	rs2	0	0
mov rd, #valeur	1	4	0	0	valeur	#	rd	
b etiquette	2	0	@instruction				0	0
bgt etiquette	2	1	@instruction				0	0



**addition :**

26 25 24

0 0 0 1 0 0 0 0 ... ..

```
0 0 0 1 0 0 0 1 ... ..
```

**soustraction :**

26 25 24

0 0 0 1 0 0 1 0 ... ..

**comparaison (=soustraction) :**

26 25 24

```
0 0 0 1 0 0 1 1 ... ..
```

# Mémoire

```

x:      .int 4
y:      .int 3
produit: .int 0
main:   ldr r1, x
        ldr r2, y
        ldr r3, produit
        mov r4, #0
boucle: cmp r4, r2
        bge fin
        add r3, r1, r3
        add r4, r4, #1
        b  boucle
fin:    str r3, produit
    
```

adresse	contenu				
0000	00	00	00	04	
0004	00	00	00	03	
0008	00	00	00	00	
000C	00	XX	XX	01	@x
0010	00	XX	XX	02	@y
0014	00	XX	XX	03	@produit
0018	14	00	00	04	
001C	13	04	02	00	
0020	21	XX	XX	00	@fin
0024	10	01	03	03	
0028	11	04	01	04	
002C	20	XX	XX	00	@boucle
0030	01	XX	XX	03	@produit

1<sup>ère</sup> passe : codage des instructions

	adresse	contenu					adresse	contenu			
@x	0000	00	00	00	04		0000	00	00	00	04
@y	0004	00	00	00	03		0004	00	00	00	03
@produit	0008	00	00	00	00		0008	00	00	00	00
@main	000C	00	XX	XX	01	@x	000C	00	00	00	01
	0010	00	XX	XX	02	@y	0010	00	00	04	02
	0014	00	XX	XX	03	@produit	0014	00	00	08	03
	0018	14	00	00	04		0018	14	00	00	04
@boucle	001C	13	04	02	00		001C	13	04	02	00
	0020	21	XX	XX	00	@fin	0020	21	00	30	00
	0024	10	01	03	03		0024	10	01	03	03
	0028	11	04	01	04		0028	11	04	01	04
	002C	20	XX	XX	00	@boucle	002C	20	00	1C	00
@fin	0030	01	XX	XX	03	@produit	0030	01	00	08	03

2<sup>nde</sup> passe : codage des adresses