Rapport de Projet
Architecture des ordinateurs

François Poguet
Enzo Costantini

Décembre 2019

Table des figures

1 Les circuits

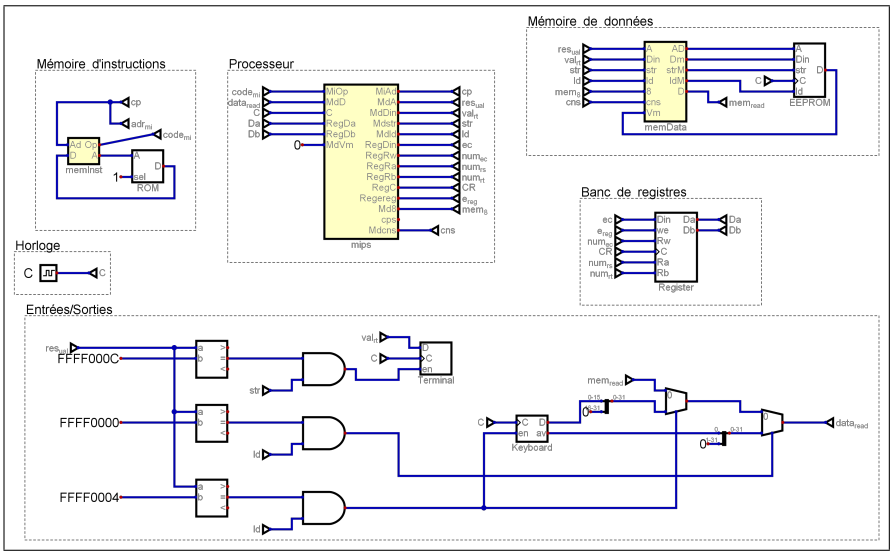


FIGURE 1 – Circuit du système (sys.dig)

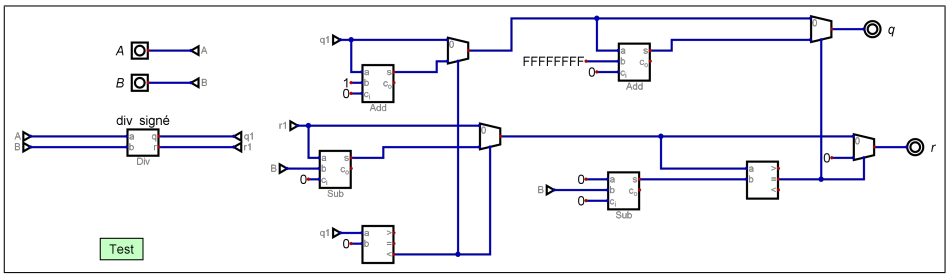


FIGURE 2 – Circuit de division signée (div.dig)

Le circuit de division signée a du être modifié pour regler des erreurs dans certains cas particuliers. (Voir figure 2)

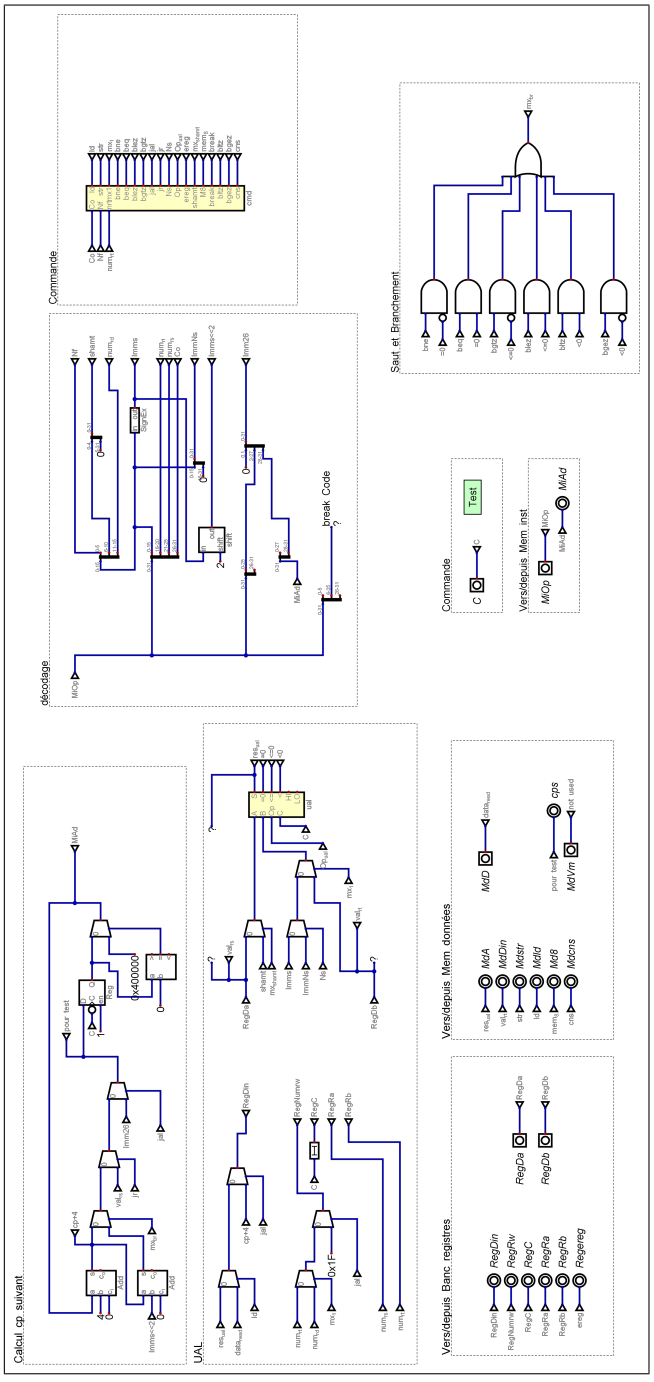


FIGURE 3 – Circuit du processeur (mips.dig)

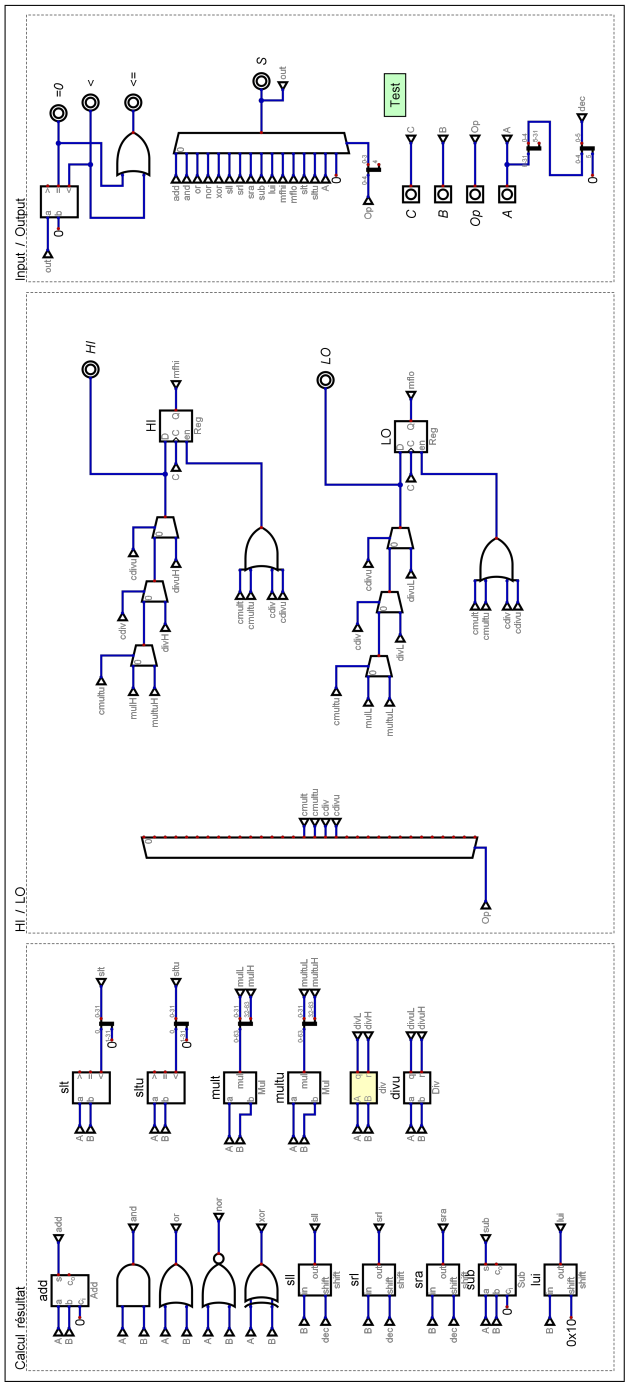


FIGURE 4 – Circuit de l'UAL (ual.dig)

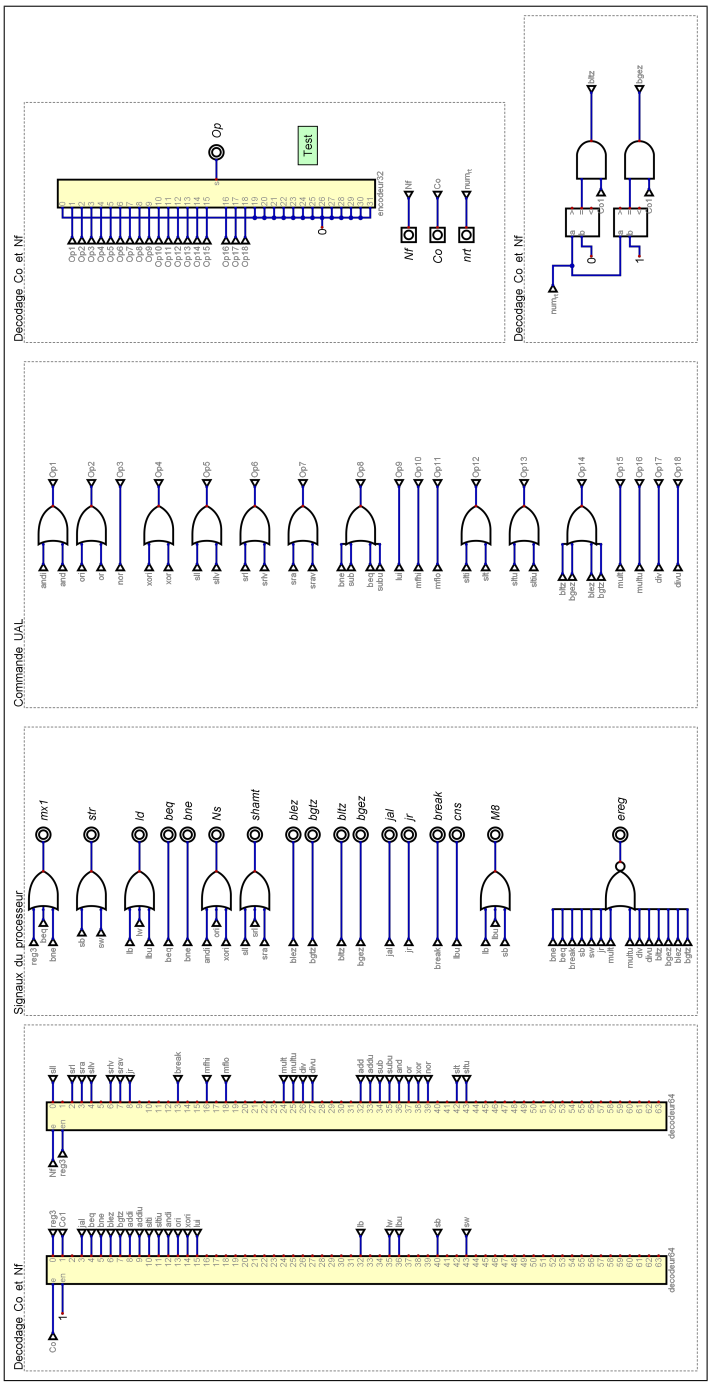


FIGURE 5 – Circuit de commande (cmd.dig)

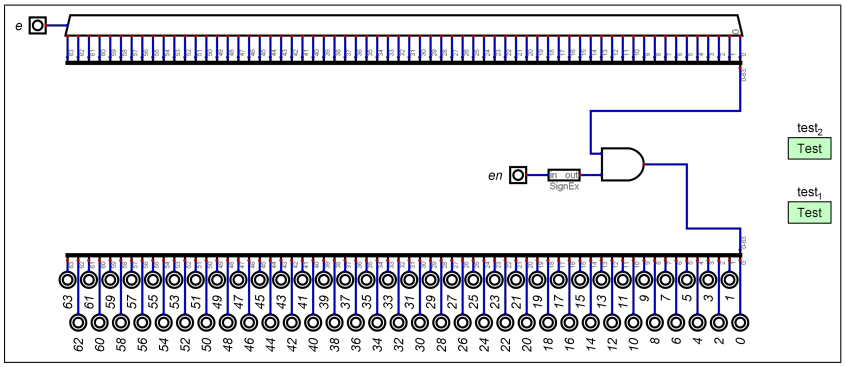


FIGURE 6 – Circuit du décodeur (decodeur64.dig)

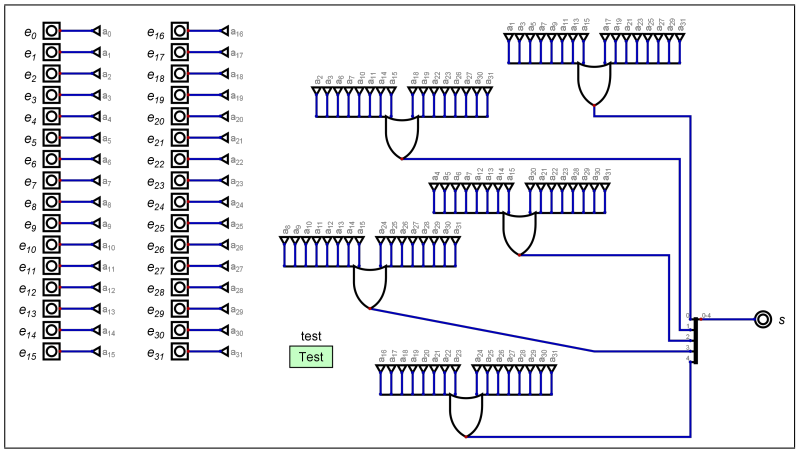
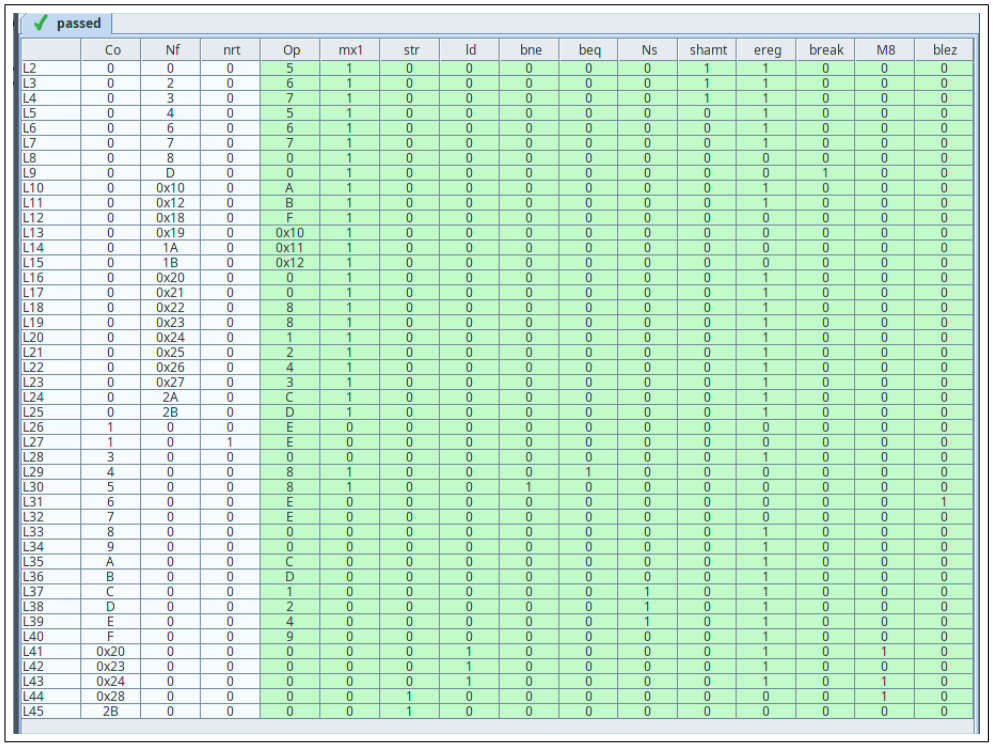


FIGURE 7 – Circuit de l'encodeur (encodeur32.dig)

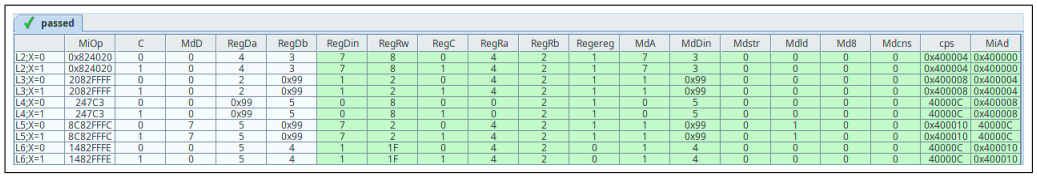
2 Les tests



passed

	Co	Nf	nrt	Op	mx1	str	ld	bne	beq	Ns	shamt	ereg	break	M8	blez
L2	0	0	0	5	1	0	0	0	0	0	1	1	0	0	0
L3	0	2	0	6	1	0	0	0	0	0	1	1	0	0	0
L4	0	3	0	7	1	0	0	0	0	0	1	1	0	0	0
L5	0	4	0	5	1	0	0	0	0	0	0	1	0	0	0
L6	0	6	0	6	1	0	0	0	0	0	0	1	0	0	0
L7	0	7	0	7	1	0	0	0	0	0	0	1	0	0	0
L8	0	8	0	0	1	0	0	0	0	0	0	0	0	0	0
L9	0	D	0	0	1	0	0	0	0	0	0	0	1	0	0
L10	0	0x10	0	A	1	0	0	0	0	0	0	1	0	0	0
L11	0	0x12	0	B	1	0	0	0	0	0	0	1	0	0	0
L12	0	0x18	0	F	1	0	0	0	0	0	0	0	0	0	0
L13	0	0x19	0	0x10	1	0	0	0	0	0	0	0	0	0	0
L14	0	1A	0	0x11	1	0	0	0	0	0	0	0	0	0	0
L15	0	1B	0	0x12	1	0	0	0	0	0	0	0	0	0	0
L16	0	0x20	0	0	1	0	0	0	0	0	0	1	0	0	0
L17	0	0x21	0	0	1	0	0	0	0	0	0	1	0	0	0
L18	0	0x22	0	8	1	0	0	0	0	0	0	1	0	0	0
L19	0	0x23	0	8	1	0	0	0	0	0	0	1	0	0	0
L20	0	0x24	0	1	1	0	0	0	0	0	0	1	0	0	0
L21	0	0x25	0	2	1	0	0	0	0	0	0	1	0	0	0
L22	0	0x26	0	4	1	0	0	0	0	0	0	1	0	0	0
L23	0	0x27	0	3	1	0	0	0	0	0	0	1	0	0	0
L24	0	2A	0	C	1	0	0	0	0	0	0	1	0	0	0
L25	0	2B	0	D	1	0	0	0	0	0	0	1	0	0	0
L26	1	0	0	E	0	0	0	0	0	0	0	0	0	0	0
L27	1	0	1	E	0	0	0	0	0	0	0	0	0	0	0
L28	3	0	0	0	0	0	0	0	0	0	0	1	0	0	0
L29	4	0	0	8	1	0	0	0	1	0	0	0	0	0	0
L30	5	0	0	8	1	0	0	1	0	0	0	0	0	0	0
L31	6	0	0	E	0	0	0	0	0	0	0	0	0	0	1
L32	7	0	0	E	0	0	0	0	0	0	0	0	0	0	0
L33	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0
L34	9	0	0	0	0	0	0	0	0	0	0	1	0	0	0
L35	A	0	0	C	0	0	0	0	0	0	0	1	0	0	0
L36	B	0	0	D	0	0	0	0	0	0	0	1	0	0	0
L37	C	0	0	1	0	0	0	0	0	1	0	1	0	0	0
L38	D	0	0	2	0	0	0	0	0	1	0	1	0	0	0
L39	E	0	0	4	0	0	0	0	0	1	0	1	0	0	0
L40	F	0	0	9	0	0	0	0	0	0	0	1	0	0	0
L41	0x20	0	0	0	0	0	1	0	0	0	0	1	0	1	0
L42	0x23	0	0	0	0	0	1	0	0	0	0	1	0	0	0
L43	0x24	0	0	0	0	0	1	0	0	0	0	1	0	1	0
L44	0x28	0	0	0	0	1	0	0	0	0	0	0	0	1	0
L45	2B	0	0	0	0	1	0	0	0	0	0	0	0	0	0

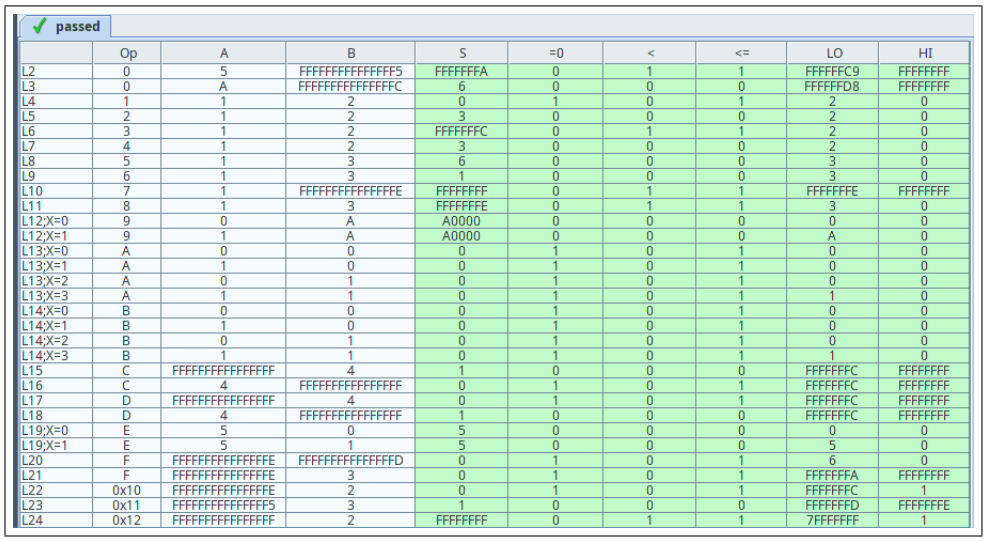
FIGURE 8 – Tests de commande



passed

	MiOp	C	MdD	RegDa	RegDb	RegDin	RegRw	RegC	RegRa	RegRb	Regereg	MdA	MdDin	Mdstr	Mdld	Md8	Mdcns	cps	MiAd
L2;X=0	0x824020	0	0	4	3	7	8	0	4	2	1	7	3	0	0	0	0	0x400004	0x400000
L2;X=1	0x824020	1	0	4	3	7	8	1	4	2	1	7	3	0	0	0	0	0x400004	0x400000
L3;X=0	2082FFFF	0	0	2	0x99	1	2	0	4	2	1	1	0x99	0	0	0	0	0x400008	0x400004
L3;X=1	2082FFFF	1	0	2	0x99	1	2	1	4	2	1	1	0x99	0	0	0	0	0x400008	0x400004
L4;X=0	247C3	0	0	0x99	5	0	8	0	0	2	1	0	5	0	0	0	0	40000C	0x400008
L4;X=1	247C3	1	0	0x99	5	0	8	1	0	2	1	0	5	0	0	0	0	40000C	0x400008
L5;X=0	8C82FFFC	0	7	5	0x99	7	2	0	4	2	1	1	0x99	0	1	0	0	0x400010	40000C
L5;X=1	8C82FFFC	1	7	5	0x99	7	2	1	4	2	1	1	0x99	0	1	0	0	0x400010	40000C
L6;X=0	1482FFFE	0	0	5	4	1	1F	0	4	2	0	1	4	0	0	0	0	40000C	0x400010
L6;X=1	1482FFFE	1	0	5	4	1	1F	1	4	2	0	1	4	0	0	0	0	40000C	0x400010

FIGURE 9 – Tests du processeur



passed

	Op	A	B	S	=0	<	<=	LO	HI
L2	0	5	FFFFFFFFFFFFFF5	FFFFFFFA	0	1	1	FFFFFFC9	FFFFFFFF
L3	0	A	FFFFFFFFFFFFFFC	6	0	0	0	FFFFFFD8	FFFFFFFF
L4	1	1	2	0	1	0	1	2	0
L5	2	1	2	3	0	0	0	2	0
L6	3	1	2	FFFFFFFC	0	1	1	2	0
L7	4	1	2	3	0	0	0	2	0
L8	5	1	3	6	0	0	0	3	0
L9	6	1	3	1	0	0	0	3	0
L10	7	1	FFFFFFFFFFFFFE	FFFFFFFF	0	1	1	FFFFFFFE	FFFFFFFF
L11	8	1	3	FFFFFFFE	0	1	1	3	0
L12;X=0	9	0	A	A0000	0	0	0	0	0
L12;X=1	9	1	A	A0000	0	0	0	A	0
L13;X=0	A	0	0	0	1	0	1	0	0
L13;X=1	A	1	0	0	1	0	1	0	0
L13;X=2	A	0	1	0	1	0	1	0	0
L13;X=3	A	1	1	0	1	0	1	1	0
L14;X=0	B	0	0	0	1	0	1	0	0
L14;X=1	B	1	0	0	1	0	1	0	0
L14;X=2	B	0	1	0	1	0	1	0	0
L14;X=3	B	1	1	0	1	0	1	1	0
L15	C	FFFFFFFFFFFFFFF	4	1	0	0	0	FFFFFFFC	FFFFFFFF
L16	C	4	FFFFFFFFFFFFFFF	0	1	0	1	FFFFFFFC	FFFFFFFF
L17	D	FFFFFFFFFFFFFFF	4	0	1	0	1	FFFFFFFC	FFFFFFFF
L18	D	4	FFFFFFFFFFFFFFF	1	0	0	0	FFFFFFFC	FFFFFFFF
L19;X=0	E	5	0	5	0	0	0	0	0
L19;X=1	E	5	1	5	0	0	0	5	0
L20	F	FFFFFFFFFFFFFFE	FFFFFFFFFFFFFFD	0	1	0	1	6	0
L21	F	FFFFFFFFFFFFFFE	3	0	1	0	1	FFFFFFFA	FFFFFFFF
L22	0x10	FFFFFFFFFFFFFFE	2	0	1	0	1	FFFFFFFC	1
L23	0x11	FFFFFFFFFFFFFF5	3	1	0	0	0	FFFFFFFD	FFFFFFFE
L24	0x12	FFFFFFFFFFFFFFF	2	FFFFFFFF	0	1	1	7FFFFFFF	1

FIGURE 10 – Tests de l'UAL

3 L'implémentation

Le programme en C ne posant pas beaucoup de problème, aucune explication n'est requise.

Le code mips, traduit à l'origine par `godbolt.org`, a été majoritairement commenté et modifié. La fonction toInt a été intégralement réécrite (voir figure 11), et d'autres parties plus minimes du code ont été améliorée. Notamment la fonction readString qui gère maintenant la suppression de caractères par la touche backspace (voir figure 12).

Quelques difficultés se sont présentées lors du chargement du programme dans le circuit du système, le "pas à pas" nous aura permis de règler nos problèmes et nous sommes arrivé au résultat attendu.

calc.asm

```
367
368 toInt :
369         # $a0 : nombre / $a1 : base / $a2 : adresse chaine res
370         sw $ra,0($sp)
371         sw $a0,-4($sp)
372         sw $t0,-8($sp)
373         bltz $a0,un # si $a0 positif on inverse et b 'un'
374         sub $a0,$0,$a0 # sinon inversion et  b 'boucle1'
375         b boucle1
376 un:
377         addi $t0,$0,'-' # premier caractere = '-'
378         sb $t0,0($a2)
379         addi $a2,$a2,1 # incrémentation adresse
380
381
382 boucle1:
383         div $a0,$a1 # n/base
384         mflo $a0 # recup du quotient
385
386         addi $a2,$a2,1 # incrémentation adresse
387         bnez $a0,boucle1 # tant que $a0 != 0
388
389         addi $t0,$0,0
390         sb $t0,0($a2) #stockage de \0 à la fin de la chaine
391         lw $a0,-4($sp) # recup valeur initiale nb
392 boucle2 :
393
394         move $t1,$a0 # $t1 = nombre
395         addi $a2,$a2,-1 # décrémentation adresse
396
397         div $t1,$a1 # nb/base
398         mfhi $t0 # recup reste
399         move $a0,$t0
400         jal forDigit
401         sb $v0,0($a2) # stockage du digit
402         move $a0,$t1
403         div $a0,$a1
404         mflo $a0 # nb = nb/base
405         bnez $a0,boucle2 # tant que nb != 0
406
407         add $v0,$0,$0 # retourne 0
408         lw $ra,0($sp)
409         lw $a0,-4($sp)
410         lw $t0,-8($sp)
411         jr $ra
```

FIGURE 11 – Code mips de la fonction toInt

```
213 readString:
214 # $a1 : adresse  tableau de caracteres
215         sw $v0,0($sp)
216         sw $t1,-4($sp)
217         sw $a0,-8($sp)
218         sw $t2,-12($sp)
219         sw $t3,-16($sp)
220 readStringBoucle :
221         getchar_console
222         add $t1,$0,$v0 # on sauvegarde $v0(caractere venant d'être tapé dans $t1
223         sb $v0,0($a0) # on stock $v0 à l'adresse de $a1
224         addi $a0,$a0,1 # on incrémente $a1
225         lw $t2,-8($sp)
226         addi $t3,$a0,-1
227         bne $t1,'\b',continue1 # si le caractere n'est pas backspace, b 'continue1'
228         addi $a0,$a0,-1 # suppression du caractère backspace
229         beq $t2,$t3,continue1 # si on est pas revenu au début de l'adresse (= la chaine est vide)
230         addi $a0,$a0,-1 # suppression du caractere tapé avant backspace
231         continue1 :
232         bne $t1,'\n',readStringBoucle # tant que la touche entrée n'est pas tapée
233         sb $0,-1($a0) # remplacement de \n par \0
234         lw $v0,0($sp)
235         lw $t1,-4($sp)
236         lw $a0,-8($sp)
237         lw $t2,-12($sp)
238         lw $t3,-16($sp)
239         jr $ra
```

FIGURE 12 – Code mips de la fonction readString