2CMPA-Ott-Alan

2.1 L’UNITE ARITHMETIQUE ET LOGIQUE (UAL)

2.1.1 Le 1er jalon : Effectuer les opérateurs de l’UAL

Nous allons construire une Unité Arithmétique et Logique (UAL) 1 bit qui comportera 4 types d’opérations :
a. L’addition
b. Le NON-OU logique
c. Le NON-ET logique
d. Le OU exclusif

Ces quatre montages doivent être placés de manière à avoir leur deux entrées (notées A et B) communes et leur sortie (notées Op0 pour l’addition, Op1 pour le NON-OU logique, Op2 pour le NON-ET logique et Op3 pour le OU exclusif) distinctes.

Additionner Table de Vérité

A	B	R	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

/S = (/A.B) + (A./B)

La sortie S correspond a un OU EXCLUSIF

R = A.B
La Sortie R correspond a un ET

NON-OU Logique

A	B	Op1
0	0	1
0	1	0
1	0	0
1	1	0

Op1 = /A./B)
La Sortie Op1 correspond a un NON-OU

NON-ET Logique

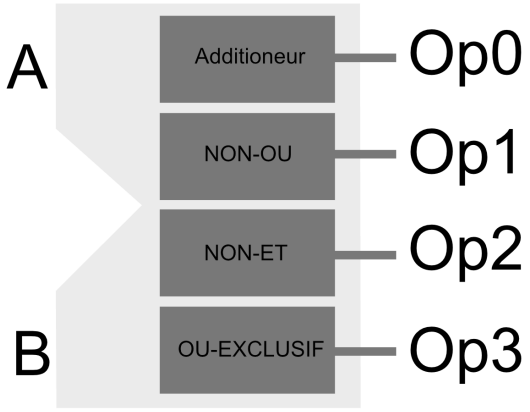
A	B	Op2
0	0	1
0	1	1
1	0	1
1	1	0

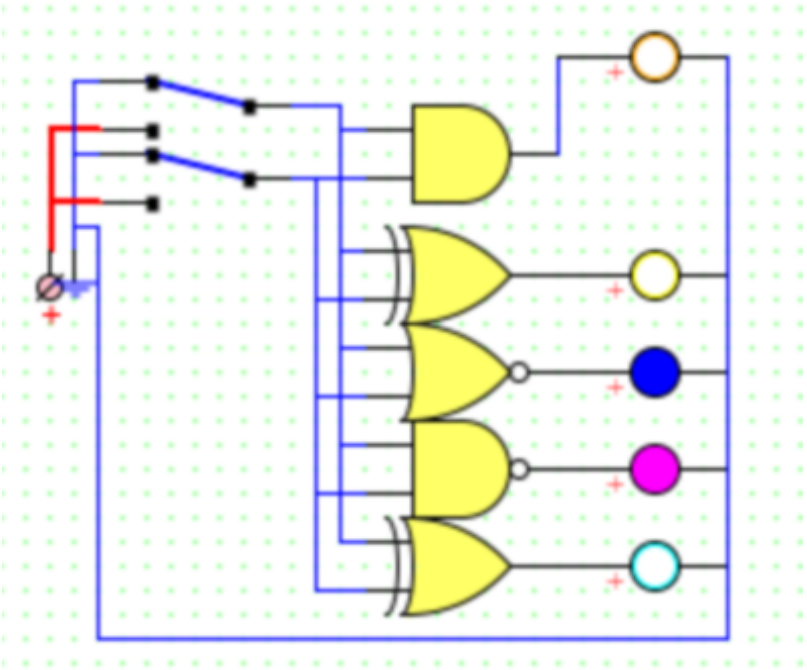
Op2 = /A./B + A./B +/A.B
La Sortie Op1 correspond a un NON-ET

OU-EXCLUSIF Logique

A	B	Op3
0	0	0
0	1	1
1	0	1
1	1	0

Op3 = /A.B + A./B
La Sortie Op1 correspond a un OU-EXCLUSIF





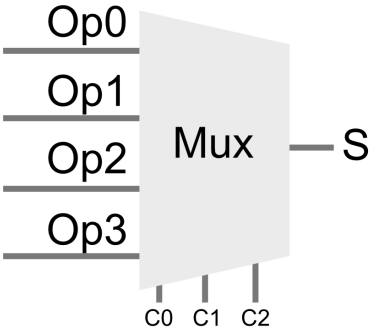
2.1.2 Le 2ème jalon : Effectuer la sortie unique

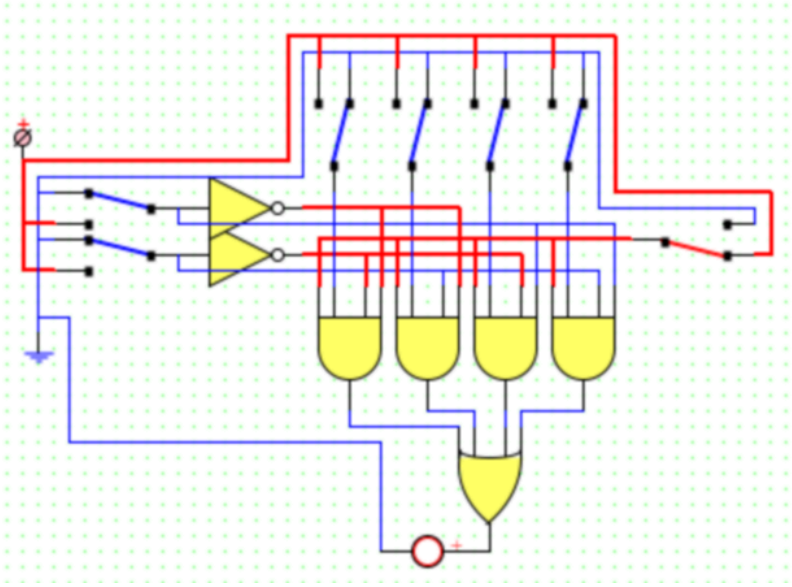
Les 4 opérateurs de l'UAL fonctionnent simultanément, cependant à un instant donné, un seul résultat nous intéresse. Pour sélectionner le bon résultat à la sortie de l'UAL, nous devons brancher un des quatre signaux Opi à la sortie S de notre UAL : nous utilisons un multiplexeur. Les signaux de contrôle de ce multiplexeur seront notés C0 et C1 (nous avons 4

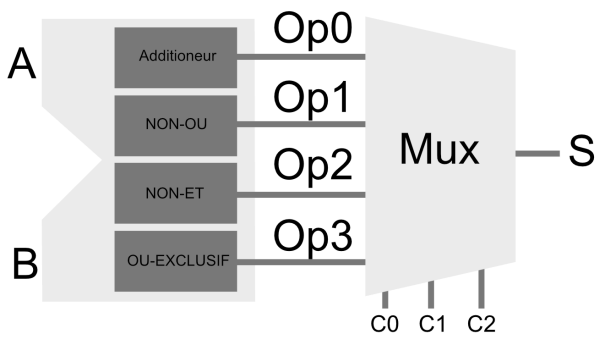
possibilités donc 2 bits de contrôle) et C2 (un signal qui permet de contrôle l'état "branché ou débranché" de la sortie S du multiplexeur).

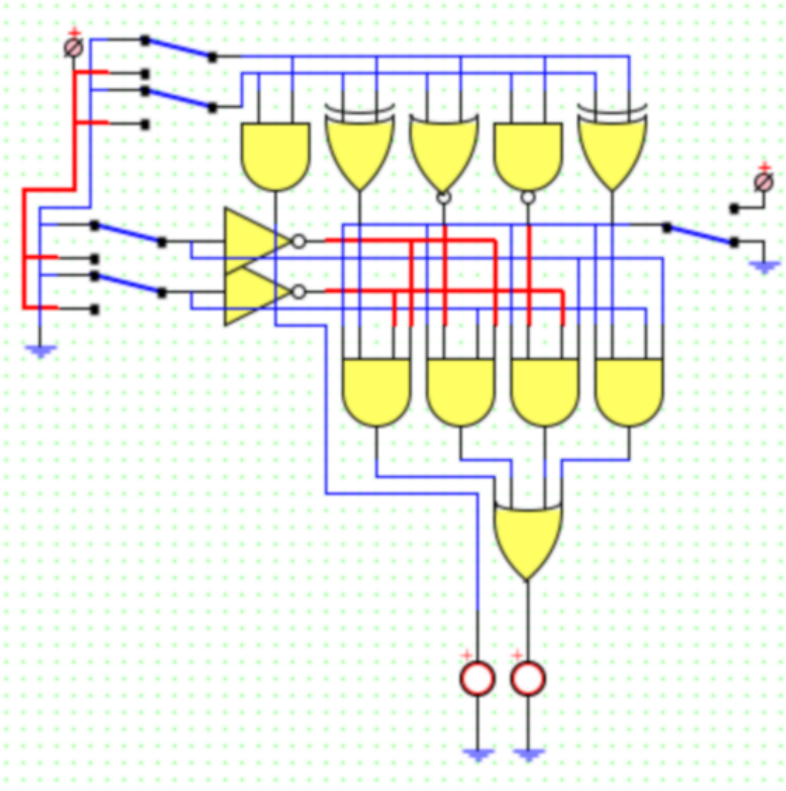
C0	C1	C2	S
0	0	1	Op0
0	1	1	Op1
1	0	1	Op2
1	1	1	Op3
0	0	0	-
0	1	0	-
1	0	0	-
1	1	0	-

S = C2(/C0./C1 + /C0.C1 + C0./C1) + C0.C1)









2.2 LES REGISTRES DES DONNEES

2.2.1 Le 3ème jalon : Effectuer les registres

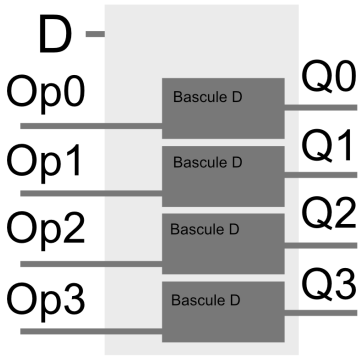
A côté du premier montage (Jalon 1 et 2), nous plaçons 4 registres de données 1 bit. Pour cela, nous utilisons 4 bascules D. L'interconnexion de ces bascules s'effectuera de la manière suivante :

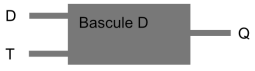
a. L'entrée D de ces 4 bascules sera commune
b. Les entrées (notées T0, T1, T2 et T3) seront distinctes
c. Les sorties (notées Q0, Q1, Q2 et Q3) seront distinctes
d. Les sorties /Q sont inutilisées

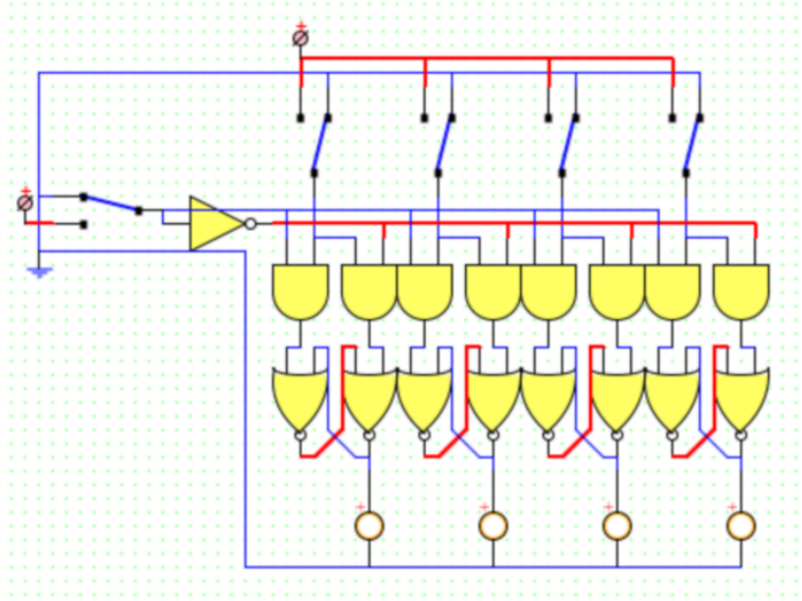
D	T	Qi
0	0	Qin-1
0	1	1

D	T	Qi
1	0	Qin-1
1	1	0

Qi = /D.T
/Qi = D.T
Qin-1 = /T







2.2.2 Le 4ème jalon : Effectuer la sélection des registres

Les 4 registres du processeur ont une entrée commune et des sorties distinctes :
a. Pour stocker l'information dans le bon registre, il faut activer le bon Ti. Comme nous avons à chaque fois un Ti à sélectionner parmi les 4, nous utiliserons un décodeur 2 vers 4.
b. Pour récupérer l'information du bon registre, il faut de nouveau utiliser un multiplexeur qui va prendre en entrée les signaux Qi et rendra en sortie un signal E.

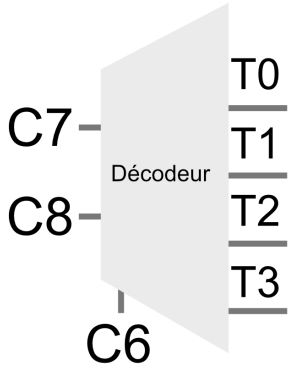
Les signaux de contrôle du multiplexeur seront notés C3, C4 et C5 et ceux du décodeur seront appelés C6, C7 et C8.

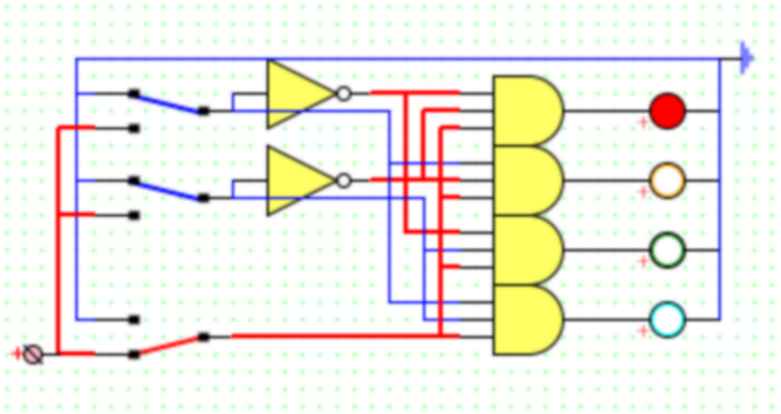
A.

C6	C7	C8	S
1	0	0	T0

C6	C7	C8	S
1	0	1	T1
1	1	0	T2
1	1	1	T3
0	0	0	-
0	0	1	-
0	1	0	-
0	1	1	-

S = C6 (/C7./C8 + C7./C8 + /C7.C8 + C7.C8)



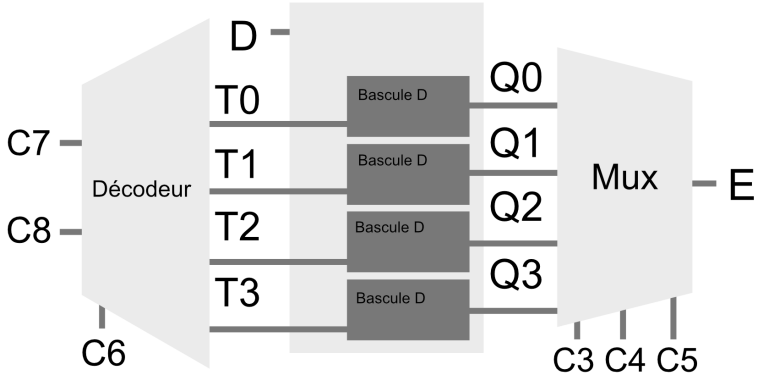


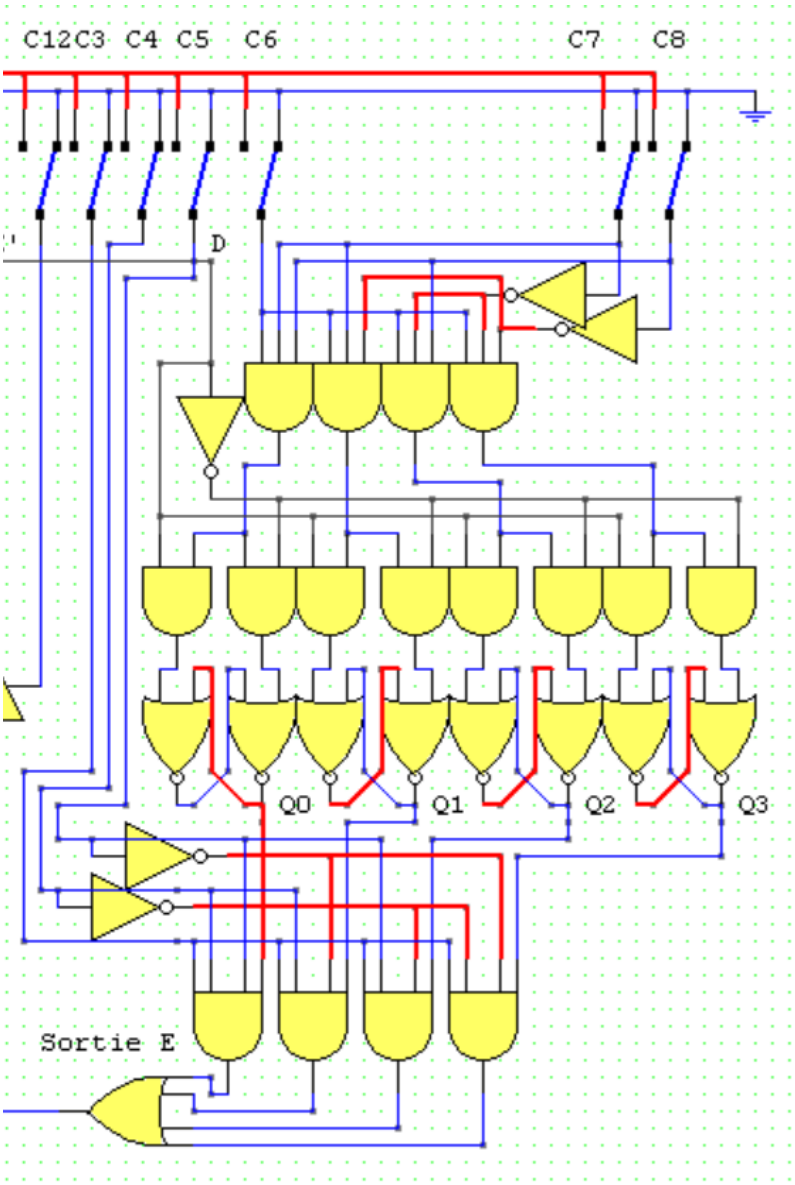
B.

C3	C4	C5	E
1	0	0	Q0
1	0	1	Q1
1	1	0	Q2
1	1	1	Q3

C3	C4	C5	E
0	0	0	-
0	0	1	-
0	1	0	-
0	1	1	-

E = C3 (/C4./C5 + C4./C5 + /C4.C5 + C4.C5)





2.3 L'INTERCONNEXION ET LE REGISTRE D'ETAT

2.3.1 Le 5ème jalon : Effectuer la connexion entre UAL et les registres

Nous disposons maintenant de deux montages l'UAL et le bloc de registres internes. Nous allons les raccorder en mettant en place un bus de données qui va être commun aux entrées et aux sorties de l'UAL et du bloc de registres. Dans notre configuration, nous devrions relier les pattes A, B, D, E et S de notre montage par un même fil qui représente donc un bus d'une largeur de 1 bit.

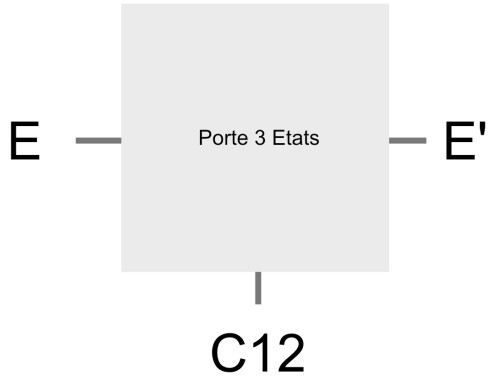
Relier directement ces pattes entre elles risque de poser des problèmes électriques. Nous devons ajouter :
a. Une porte 3 états à la sortie de notre bloc de registres (juste après la patte E, nous obtenons une patte E').
b. Une autre porte 3 états à la sortie de l'UAL (juste après la patte S, nous obtenons donc une patte S').
c. Deux registres qui permettront de stocker de façon temporaire les opérandes A et B (à placer juste avant les pattes A et B) : nous obtenons les pattes A' et B'.

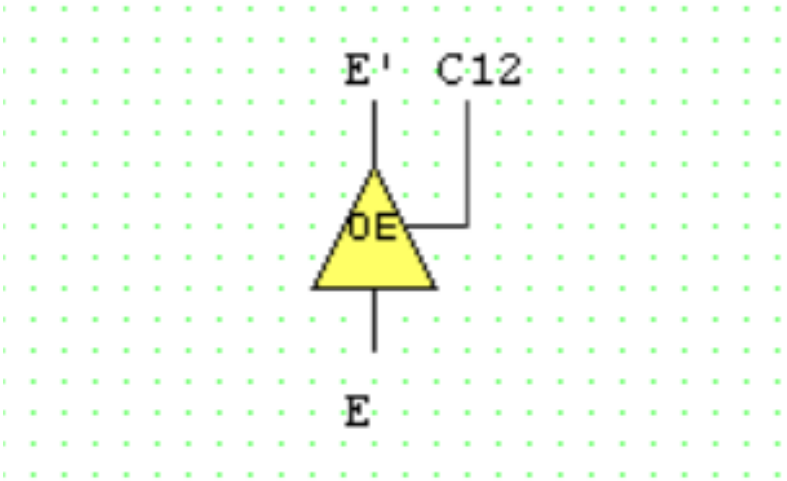
A.

E	C12	E'
0	1	0
1	1	1
0	0	-

E	C12	E'
1	0	-

E' = Ci E
/E' = Ci /E

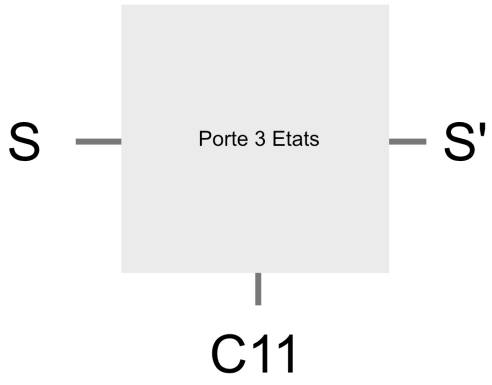


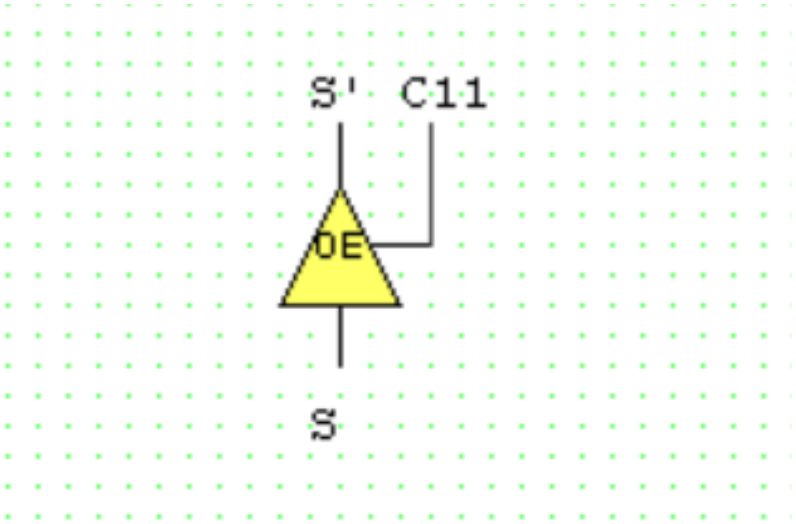


B.

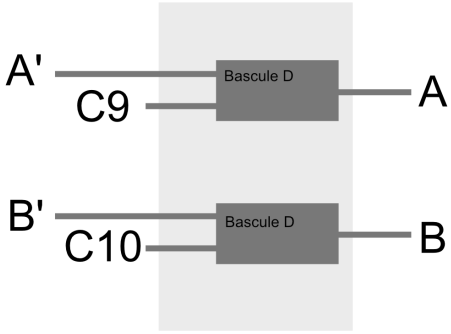
S	C11	S'
0	1	0
1	1	1
0	0	-
1	0	-

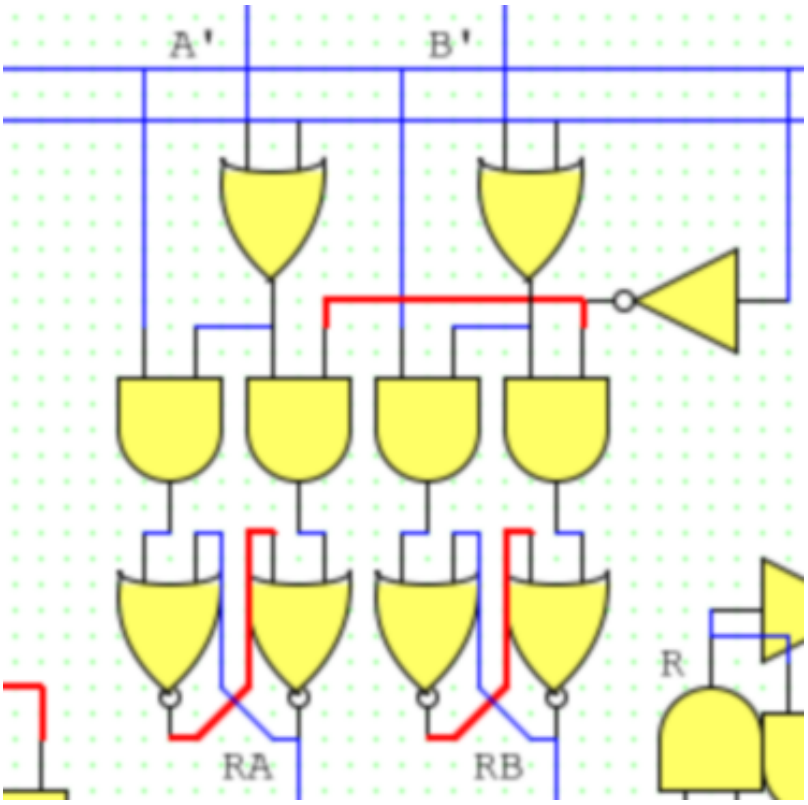
S' = Ci S
/S' = Ci /S



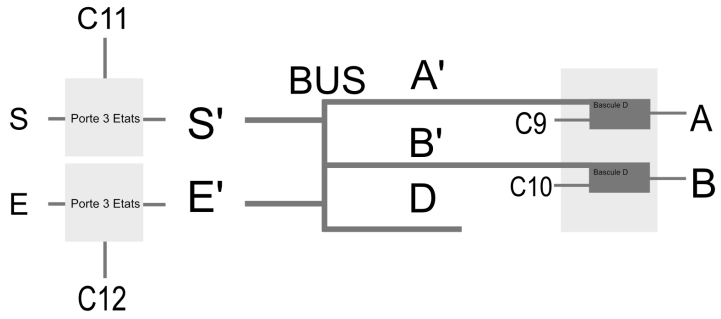


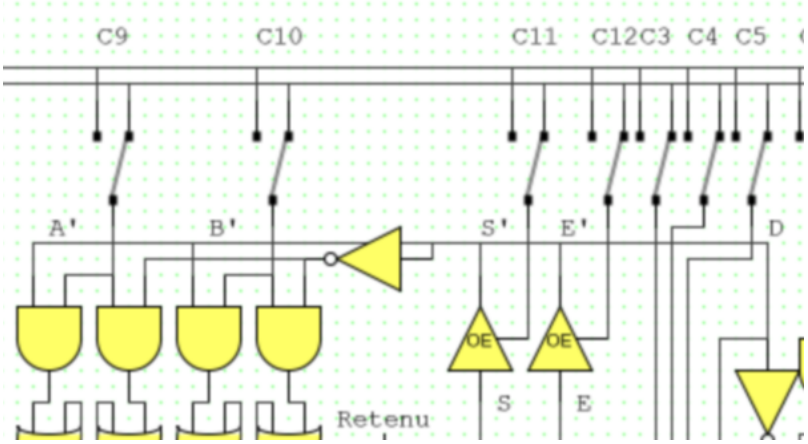
c.





Nous pouvons maintenant relier les pattes A', B', S', E' et D entre elles par un fil et nous notons C9, C10, C11 et C12 les nouvelles pattes de contrôle de ce montage.





2.3.2 Le 6ème jalon : Effectuer les flags des valeurs

Notre additionner rend un résultat sur 2 bits :
a. Le bit issu de l’addition des deux bits placés en entrée.
b. Le bit de retenu que nous n’avons pas encore utilisé.

Nous allons placer deux nouvelles bascules D qui permettront de stocker l’état de notre UAL lors de chaque opération :
a. Si l’addition provoque une retenue, le premier registre doit contenir 1 ("le bit carry")
b. Si une opération retourne un zéro, le second registre doit contenir 1 ("le bit zéro").

R	C13	C
1	1	1
0	1	0
1	0	-
0	0	-

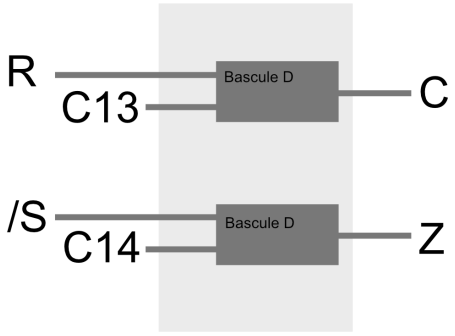
C = C13 . R

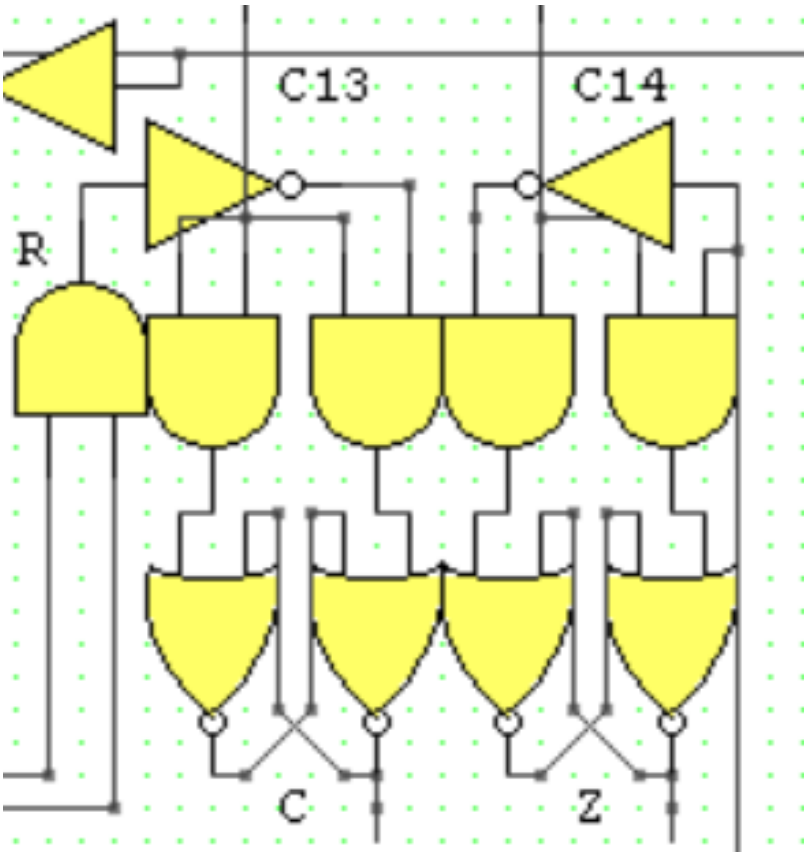
/C = C13 . /R

S	C14	Z
1	1	0
0	1	1
1	0	-
0	0	-

Z = C14 . /S

/Z = C14 . S





2.3.3 Le 7ème jalon : Effectuer le montage final

Tous les signaux Ci mis ensemble forment le bus de contrôle de notre processeur 1 bit. Dans un premier, nous allons ajouter un certain nombre de commutateurs afin de contrôler l'état des signaux Ci. Par la suite ces signaux seront câblés à d'autres éléments du processeur comme le séquenceur et le décodeur d'instructions.

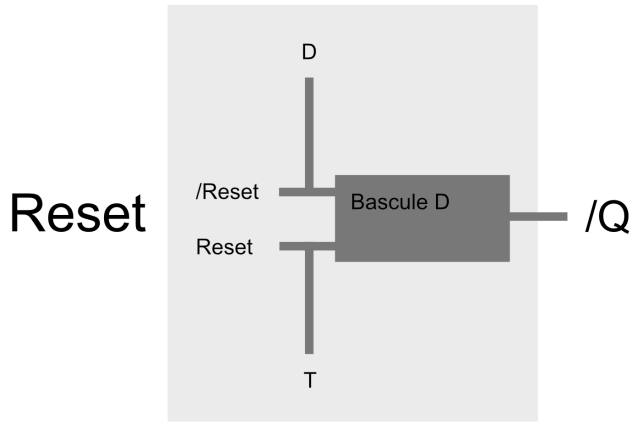
Pour faciliter la manipulation de notre processeur, nous devons ajouter un signal supplémentaire appelé RESET (qui sera commandé par un commutateur) :
a. Si le signal est à 1, cela force à enregistrer un "0" dans toutes les bascules D du montage
b. Si le signal est à 0, le montage fonctionne normalement

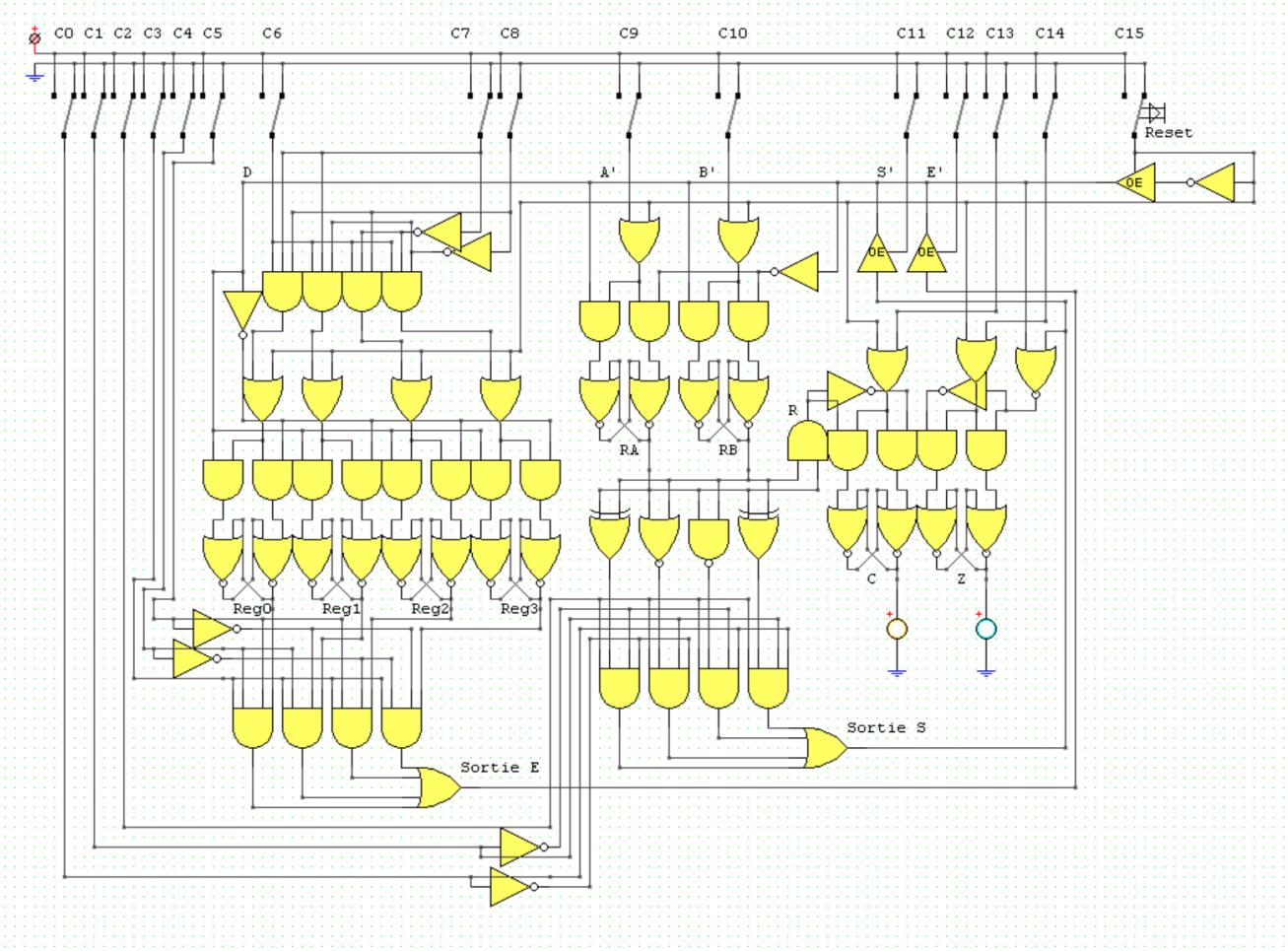
C15 (Reset)	RegA	RegB	C	Z	Reg0	Reg1	Reg2	Reg3
0	RA(n-1)	RB(n-1)	C(n-1)	Z(n-1)	R0(n-1)	R1(n-1)	R2(n-1)	R3(n-1)
1	0	0	0	0	0	0	0	0

/Sortie = C15

C15 (Reset)	Ci	Reg
1	1	0
1	0	Reg(n-1)
0	1	Reg(n-1)
0	0	Reg(n-1)

/Reg = C15 . Ci





2.3.4 BONUS

2.4 LE FONCTIONNEMENT DU MONTAGE

2.4.1 Le 8ème jalon : Effectuer la simulation à partir d'un scénario

Maintenant que nous disposons d'un processeur 1 bit, déterminer la séquence à effectuer au niveau des commutateurs Ci pour effectuer les suites d'instructions ci-dessous :

1. RESET
2. Faire un NON-OU entre Reg 0 et Reg 1
3. Stocker le résultat dans Reg 2

4. Additionner Reg 0 et Reg 2
5. Stocker le résultat dans Reg 3
6. Additionner Reg 2 et Reg 3
7. Stocker le résultat dans Reg 0

	PFort														
Etap	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1
1:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2:	0	0	0	1	0	0	1	0	0	0	1	1	1	0	0
3:	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0
4:	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1
5:	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
6:	0	0	0	1	0	0	1	0	0	0	1	1	1	0	0
7:	0	0	0	1	0	1	0	0	0	0	1	0	1	0	0
8:	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1
9:	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0
10:	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0
11:	0	0	0	0	1	0	0	1	1	1	0	0	0	1	1