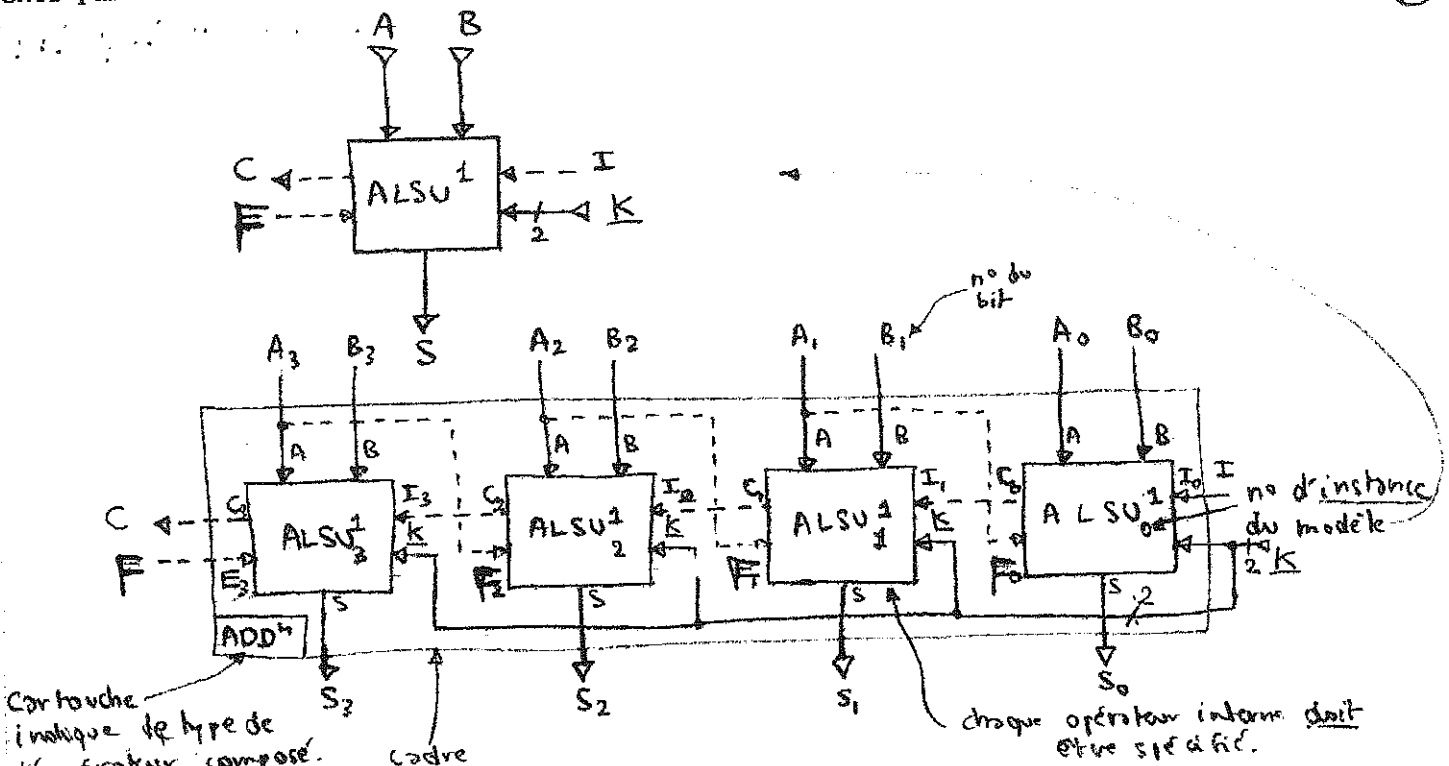


$ALSU^4$ est la composition (ou l'agrégation) de 4 instances de type ALU^1.

1. Chaque $ALSU^1_n$ s'occupe d'un bit n° n de la sortie S. Elles effectuent toutes la même opération, donc leur commande commune K est la même.

Il faut bien sûr spécifier chaque $ALSU^1$; pour cela, voyons ce que vaut la sortie S_n pour chaque opération :

K	Opération	S_n
00	PASS B	B_n
01	SHIFT RIGHT A	A_{n+1}
11	ADD	$A_n \oplus B_n \oplus C_{n-1}$
10	NAND	$\overline{A_n \cdot B_n}$

$\oplus$ = addition
$\cdot$: et

On voit bien que pour K = 11 $ALSU^4$ est un ADD^4, donc chaque $ALSU^1_n$ se comporte comme un additionneur à 1 bit ADD^1_n, qui a une entrée de retenue I_n et une sortie de retenue C_n. Bien sûr, $I_n = C_{n-1}$

On complète donc les schémas en ajoutant ces entrées et sorties, ainsi que leurs connexions. (en pointillés ici).

De même, pour K = 01, la sortie $S_n = A_{n+1}$.
Or chaque tranche n'a accès qu'à A_n.
Il faut donc bien ajouter une entrée E_n qui reçoit A_{n+1} pour la tranche n° n. $E_n = A_{n+1}$
Là aussi, on ajoute l'entrée et ses connexions sur les schémas (pointillés).

On peut maintenant spécifier rigoureusement la tranche $ALSU^1$: il a deux sorties S et C qui sont spécifiées séparément par une expression algébrique en fonction des entrées de données ABIF pour chaque commande K_1K_0

K	S	C
00	B	X
01	F	X
11	A*B*I	AB+AI+BI ← vu en cours
10	$\overline{A \cdot B}$	X

vu en cours

* = ou exclusif (XOR)
X = pas défini
\+ = OU (OR)

On note que C n'est pas utilisée pour $K \neq 11$, donc sa valeur n'est pas spécifiée : on la prendra 0 ou 1 pour simplifier la logique ensuite.

Si l'on veut établir la table de Karnaugh directement, on a 4 lignes et $2^4 = 16$ colonnes, soit au mieux 8 x 8 cases, ce qui est beaucoup.

On va donc simplifier en décomposant l'$ALSU^2$ en deux couches :

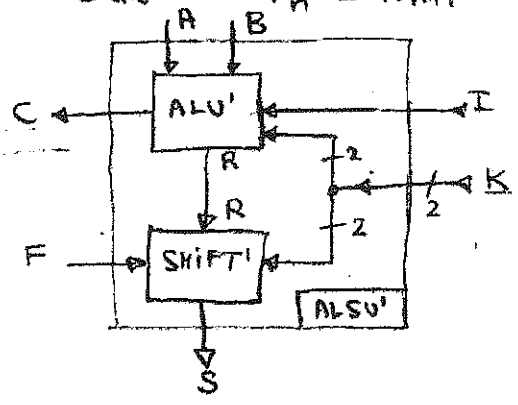
- ALU^1 qui effectue des opérations purement arithmétiques et logiques et fournit un résultat R
- $SHIFT^1$ qui effectue le décalage quand nécessaire.

* ALU^1 reçoit A, B et I et fournit R et C.
* $SHIFT^1$ reçoit le bit E et R et fournit S, mais n'a que faire de I.

Décomposons le tableau en montrant d'abord R puis S :

K	R	C	S
00	B	X	R (donc B)
01	A	X	F
11	A*B*I	AB+AI+BI	R (donc A*B*I)
10	$\overline{A \cdot B}$	X	R (donc $\overline{A \cdot B}$)

En principe, $E_n := A_{n+1}$, mais si l'on veut respecter le principe des couches, E ne doit dépendre que de R, sortie de la couche du dessus.
Donc $F_n = R_{n+1}$ et $R_{n+1} = A_{n+1}$ quand K = 01



ALU^1 et $SHIFT^1$ reçoivent chacun K pour savoir quelle opération effectuer.

On note que l'on aurait pu décomposer l'opérateur ALSU⁴ d'abord en couches (comme dans la feuille de TD) :

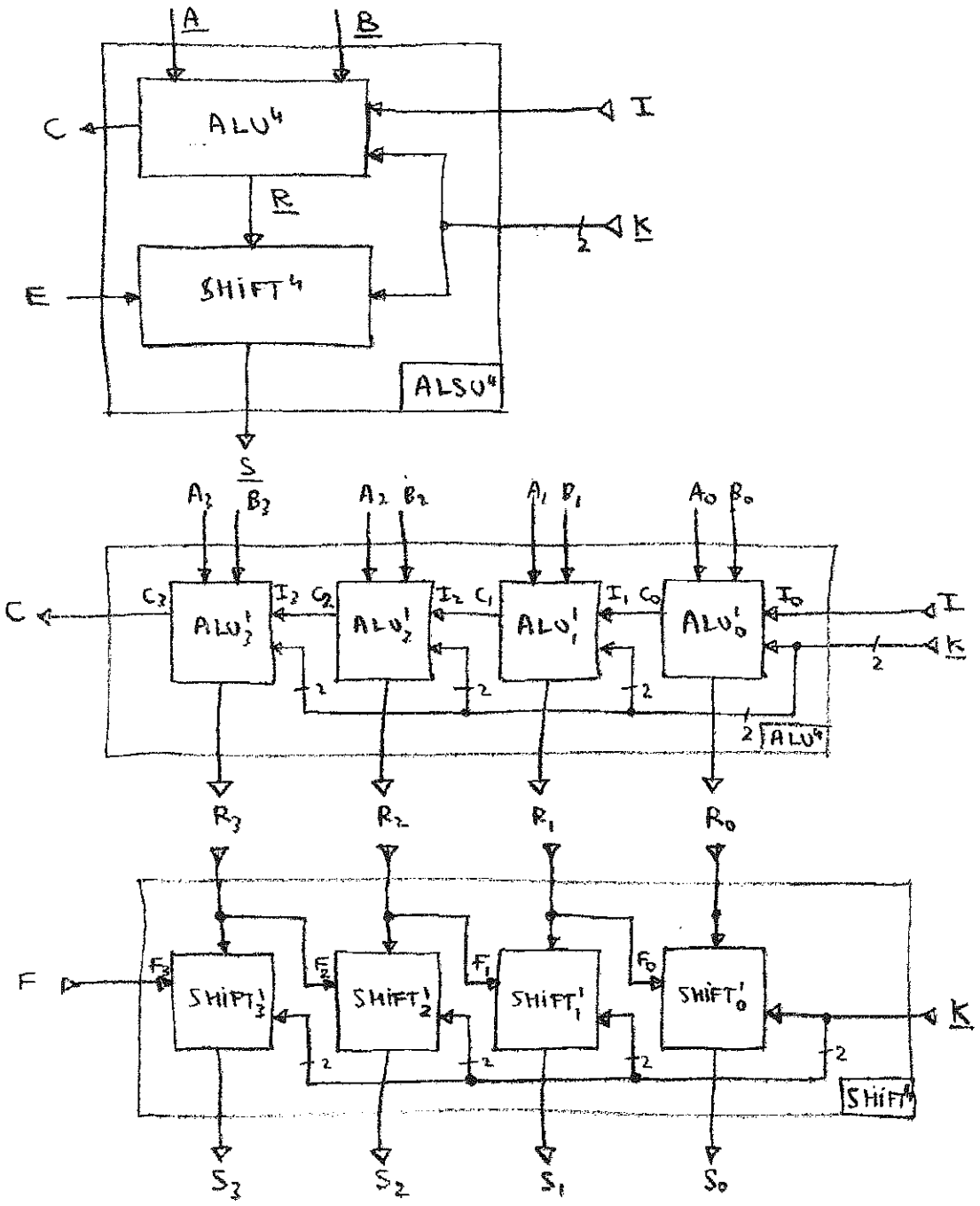


Table de Karnaugh de $R(A, B, I, K_1, K_0)$

$K_1 K_0$ \ A BI	000	001	011	010	110	111	101	100
00	0	0	1	1	1	1	0	0
01	0	0	0	0	1	1	1	1
11	0	1	0	1	0	1	0	1
10	1	1	1	1	0	0	1	1

Polynôme booléen minimal :

$$R = \underbrace{\overline{K_1}\overline{K_0}B}_{①} + \underbrace{\overline{K_1}K_0A}_{②} + \underbrace{K_1\overline{K_0}\overline{A}}_{③} + \underbrace{K_1\overline{K_0}\overline{B}}_{④} +$$

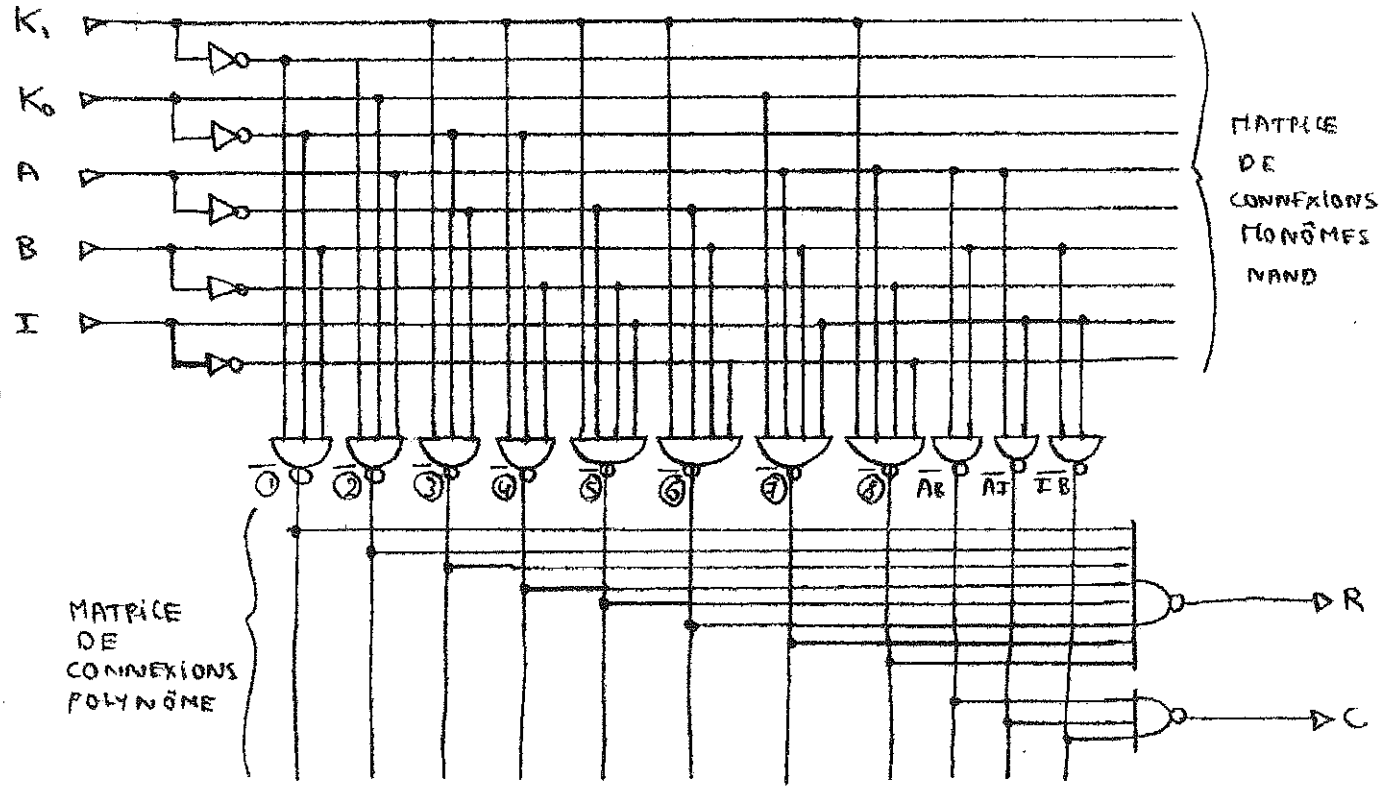
$$+ \underbrace{K_1\overline{A}\overline{B}I}_{⑤} + \underbrace{K_1\overline{A}B\overline{I}}_{⑥} + \underbrace{K_0AB\overline{I}}_{⑦} + \underbrace{K_1A\overline{B}\overline{I}}_{⑧}$$

On a 8 monômes en et.

Là où C n'est pas défini, on va lui donner la même valeur que là où il l'est, donc :

$C = AB + AI + IB$ 3 monômes en et.

Réseau logique avec des NANDs (voir synthèse suivante)

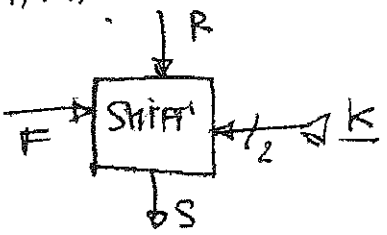


Note : En réalité, l'ALU est souvent synthétisée directement avec des transistors, comme une porte complexe.
De plus, des mises en facteurs de termes communs à toutes les tranches (et calculés une seule fois) seraient effectuées. ($\overline{K_1}\overline{K_0}$, $\overline{K_1}K_0$, $K_1\overline{K_0}$) cf. ci-après.

SHIFT'

Table de Karnaugh $S(E, R, K_1, K_0)$

K_1K_0 \ FR	00	01	11	10
00	0	① 1	1	0
01	0	0	1	1 ③
11	0	② 1	1	0
10	0	1	1	0



Polynôme booléen minimal :

$$S = \underbrace{\overline{K_0} R}_{①} + \underbrace{K_1 R}_{②} + \underbrace{\overline{K_1} K_0 F}_{③}$$

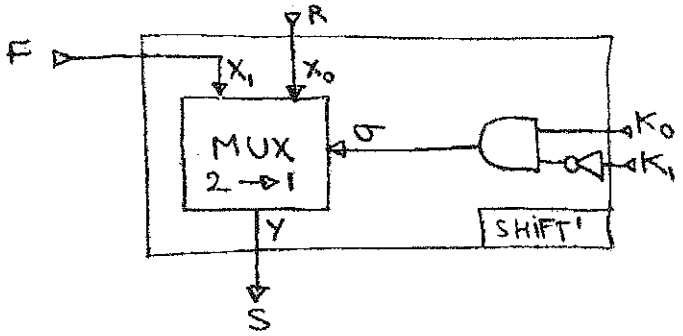
Synthèse avec un MUX :

On note que : $S = (\overline{K_0} + K_1) \cdot R + \overline{K_1} \cdot K_0 \cdot F$ (mise en facteurs)

$= \overline{K_0 \overline{K_1}} \cdot R + K_0 \overline{K_1} \cdot F$ (Morgan)

$= \overline{\sigma} \cdot R + \sigma \cdot F$ (avec $\sigma = K_0 \cdot \overline{K_1}$)

On peut donc synthétiser S avec un multiplexeur MUX à 2 voies de 1 bit vers une voie de 1 bit. d'entrée n°0 R et n°1 F, et de sélection σ



En réalité, sur un SHIFT^4 on mettrait en commun σ et utiliserait un MUX par tranche.

Le MUX est prévu à réaliser en CMOS, et l'on synthétise donc les SHIFTers avec des MUX comme ci-dessus

Synthèse avec des NORs du décaleur

Il suffit de faire la table de Karnaugh de $\overline{S}(E, R, K_1, K_0)$ soit encore de chercher les rectangles entourant des 0.

On obtient : $\overline{S} = \overline{F} \cdot \overline{R} + \overline{K_1} \cdot K_0 \cdot \overline{F} + K_1 \cdot \overline{R} + \overline{K_1} \cdot \overline{K_0} \cdot \overline{R}$

$S = \overline{\overline{S}} = \overline{\overline{F} \cdot \overline{R} + \cdots} = \overline{\overline{F + R} + \cdots} =$

$(F \downarrow R) \downarrow (K_1 \downarrow \overline{K_0} \downarrow F) \downarrow (\overline{K_1} \downarrow R) \downarrow (K_1 \downarrow K_0 \downarrow R)$

Réduction du polynôme complémentaire pour l'ALU

Nous avons tenté d'écrire un polynôme booléen de $\overline{R}$.

Mais il se peut qu'en fait le polynôme booléen de $\overline{R}$ soit plus simple. On va donc considérer les 0 comme des 1 et vice-versa :

K_1K_0 \ ABI	000	001	011	010	110	111	101	100
00	0	0	1	1	1	1	0	0
01	0	0	0	0	1	1	1	1
11	0	1	0	1	0	1	0	1
10	1	1	1	1	0	0	1	1

$$\overline{R} = \underbrace{\overline{K_1}\cdot\overline{K_0}\cdot\overline{B}}_{①} + \underbrace{\overline{K_1}\cdot K_0\cdot\overline{A}}_{②} + \underbrace{K_1\cdot\overline{K_0}\cdot A\cdot B}_{③}$$

$$+ \underbrace{K_0\cdot\overline{A}\cdot\overline{B}\cdot\overline{I}}_{④} + \underbrace{K_0\cdot\overline{A}\cdot B\cdot I}_{⑤} + \underbrace{K_1\cdot A\cdot B\cdot\overline{I}}_{⑥} + \underbrace{K_1\cdot K_0\cdot A\cdot\overline{B}\cdot I}_{⑦}$$

On a un monôme de moins, et la complexité tombe à 34 au lieu de 36.

Synthèse avec des NORs

On pourrait constituer un réseau de NANDs comme avant, et faire suivre le tout d'une porte NOT.

On peut aussi utiliser les lois de Morgan pour mettre ce polynôme sous forme d'un réseau de NORs :

$$R = \overline{\overline{R}} = \overline{\overline{K_1}\cdot\overline{K_0}\cdot\overline{B} + \overline{K_1}\cdot K_0\cdot\overline{A} + \cdots}$$

$$= \mathrm{NOR}\left(\overline{K_1}\cdot\overline{K_0}\cdot\overline{B}\,;\ \overline{K}\cdot K_0\cdot\overline{A}, \ldots\right)$$

$$= \mathrm{NOR}\left(\overline{\overline{\overline{K_1}+\overline{K_0}+\overline{B}}}, \overline{\overline{\overline{K}+\overline{K_0}+\overline{A}}}, \ldots\right) \qquad \text{(Morgan)}$$

$$= \mathrm{NOR}\left(\overline{K_1+K_0+B}, \overline{K+\overline{K_0}+A}, \ldots\right)$$

$$= \mathrm{NOR}\left(\mathrm{NOR}(K_1,K_0,B), \mathrm{NOR}(K,\overline{K_0},A), \ldots\right)$$

Synthèse directe avec des transistors

On dispose d'un polynôme booléen $\overline{R}$. Simplifions-le en 2 étapes:

1/ Mettons en facteur les termes commun à toutes les tranches (fonctions de K_1 et K_0) :

$$\overline{R} = \underbrace{(\overline{K_1}\cdot\overline{K_0})}_{\beta}\cdot\overline{B} + \underbrace{(\overline{K_1}\cdot K_0)}_{\alpha}\cdot\overline{A} + \underbrace{(K_1\cdot\overline{K_0})}_{\gamma}\cdot A\cdot B + (K_0)\cdot\overline{A}\cdot\overline{B}\cdot\overline{I} + K_0\cdot\overline{A}\cdot B\cdot I + K_1\cdot A\cdot B\cdot\overline{I} + \underbrace{(K_1 K_0)}_{\delta} A\cdot\overline{B}\cdot I$$

$\alpha = \overline{K_1}\cdot\overline{K_0}$: $= 1 \Leftrightarrow$ PASS A
$\beta = \overline{K_1}\cdot K_0$: ——— B
$\gamma = K_1\cdot\overline{K_0}$: ——— NAND
$\delta = K_1\cdot K_0$: ——— ADD

2/ Si ils sont tous nuls, ou plusieurs sont égaux à 1 $\Rightarrow$ X

$$\overline{R} = \beta\cdot\overline{B} + \alpha\cdot\overline{A} + \gamma\cdot A\cdot B + \delta\cdot\overline{A}\cdot\overline{B}\cdot\overline{I} + \delta\cdot\overline{A}\cdot B\cdot I + \delta\cdot A\cdot B\cdot\overline{I} + \delta\, A\cdot\overline{B}\cdot I$$

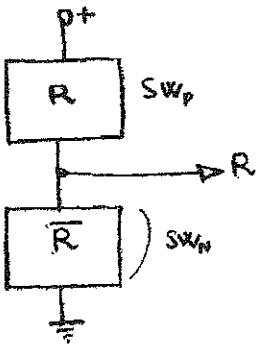
On peut ensuite effectuer des mises en facteur pour minimiser le nombre de portes. On peut le faire avec $\overline{A}$, $\overline{B}$, $A\cdot B$ et δ. Cette dernière est plus efficace :

$$\overline{R} = \alpha\cdot\overline{A} + \beta\cdot\overline{B} + \gamma\cdot A\cdot B + \delta\cdot\underbrace{(\overline{A}\cdot\overline{B}\cdot\overline{I} + \overline{A}\cdot B\cdot I + A\cdot\overline{B}\cdot I + A\cdot B\cdot\overline{I})}_{A*B*I}$$

La complexité est alors de 27 par tranche.

$R = 1 \Rightarrow \overline{R} = 0$

On peut donc décrire ce système avec des interrupteurs :



$R = 0 \Leftrightarrow \begin{cases} SW_N & \text{fermé} \\ SW_p & \text{ouvert} \end{cases}$