Simplification par tables de Karnaugh (KAR)

Cours Conception Numérique

Orientation : Systèmes industriels (SYND)
Cours : Conception Numérique (Cnum)
Auteur : Christophe Bianchi, François Corthay, Pierre Pompili, Silvan Zahno
Date : 25 août 2022
Version : v2.1

Table des matières

1 Introduction

La réalisation de fonctions combinatoires à l'aide de portes logiques nécessite l'écriture de ces fonctions sous une forme correspondant au schéma qui sera mis en oeuvre.

La technique la plus répandue pour le développement manuel de circuits consiste à utiliser des **tables de Karnaugh**. A l'aide de ces tables, il est possible de développer une fonction quelconque sous forme de somme de produits, de produit de sommes ou de OU-exclusif de produits. Pour des fonctions avec un nombre élevé d'entrées, plusieurs techniques permettent d'en réduire la complexité pour les rendre maîtrisables.

2 Tables de Karnaugh

La table de Karnaugh donne une repésentation tabulaire d'un diagramme de Venn.

2.1 Table à 2 variables

La figure 1 présente un diagramme de Venn à 2 variables et la table de Karnaugh correspondante.

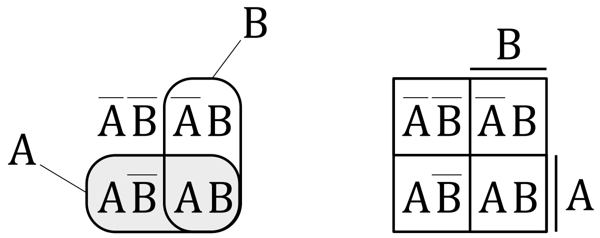


Figure 1 – Table de Karnaugh à 2 variables

2.2 Table à 3 variables

La figure 2 présente un diagramme de Venn à 3 variables et la table de Karnaugh correspondante.

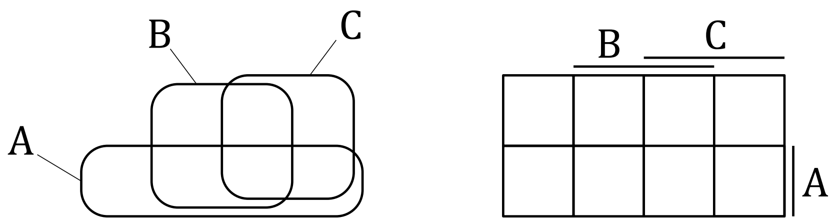


Figure 2 – Table de Karnaugh à 3 variables

2.3 Table à 4 variables

La figure 3 présente un diagramme de Venn à 4 variables et la table de Karnaugh correspondante.

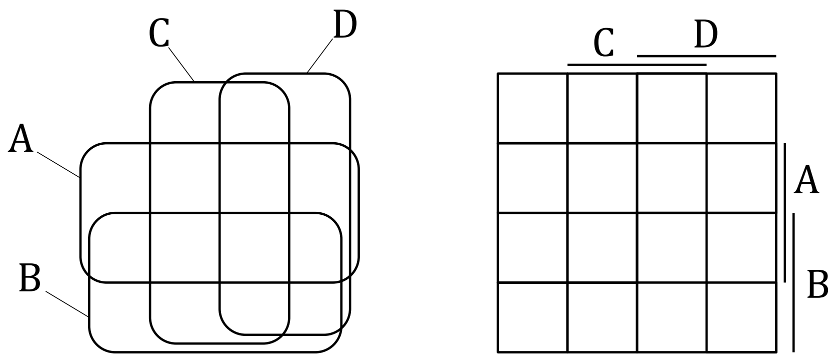


Figure 3 – Table de Karnaugh à 4 variables

2.4 Tables à plus de 4 variables

Lorsque le nombre d'ensembles dépasse 4, il n'est plus évident de dessiner les ensembles de manière à faire apparaître toutes les intersections possibles. Pour les tables de Karnaugh, la technique consiste alors à placer des tables à 4 variables et de les sélectionner ou non par les variables supplémentaires. Ceci est représenté à la figure 4 pour une table de Karnaugh à 5 variables.

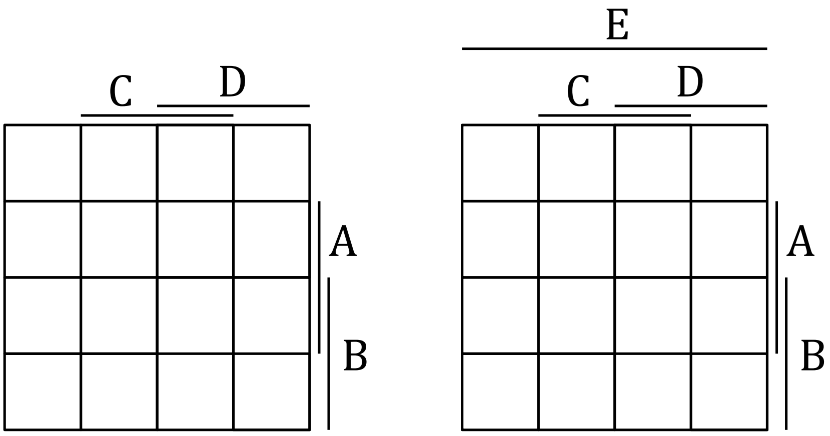


FIGURE 4 – Table de Karnaugh à 5 variables

3 Simplification sous fome de somme de produits

3.1 Représentation d'un produit

Le produit de plusieurs variables, lesquelles peuvent éventuellement êtres inversées, correspond à l'intersection des ensembles correspondants.

Dans une table de Karnaugh, un produit se représente par un ensemble de cases contiguës. Le nombre de cases est toujours une puissance de 2. Il dépend du nombre de facteurs du produit : plus le nombre de facteurs est élevé et plus le nombre de cases est petit.

3.1.1 Définition : monôme

Un monôme est un produit de plusieurs variables, lesquelles peuvent éventuellement êtres inversées.

Dans la table de Karnaugh, un monôme correspond à un ensemble de cases contiguës dont le nombre est une puissance de 2.

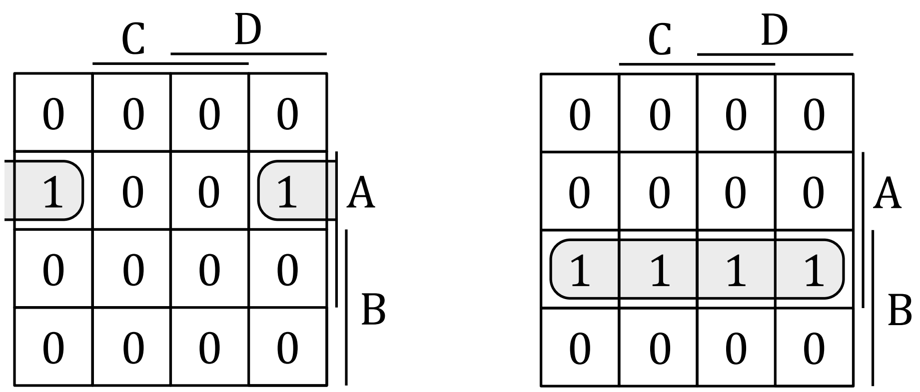
3.1.2 Définition : minterme

Un minterme est un produit de toutes les variables, lesquelles peuvent éventuellement êtres inversées.

Un minterme correspond à une case de la table de Karnaugh.

Exemples

La figure 5 présente des produits de variables dans leur table de Karnaugh.



$\bar{C}\bar{B}A$ $\quad$ BA

FIGURE 5 – Représentation de produits dans une table de Karnaugh

> Un ensemble peut dépasser le bord d'une table de Karnaugh et se développer à partir de l'autre bord qui lui fait face.
> Un produit qui comporte toutes les variables d'entrées occupe une seule case dans la table de Karnaugh. Retirer un facteur dans l'expression d'un produit revient à doubler le nombre de cases occupées.

3.2 Représentation d'une somme de produits

Une somme de produits se représente par la réunion des ensembles correspondant aux produits concernés.

3.2.1 Définition : polynôme

Un polynôme est une somme de monômes. Il correspond à une somme de produits.

Exemples

La figure 6 présente des sommes de produits de variables dans leur table de Karnaugh.

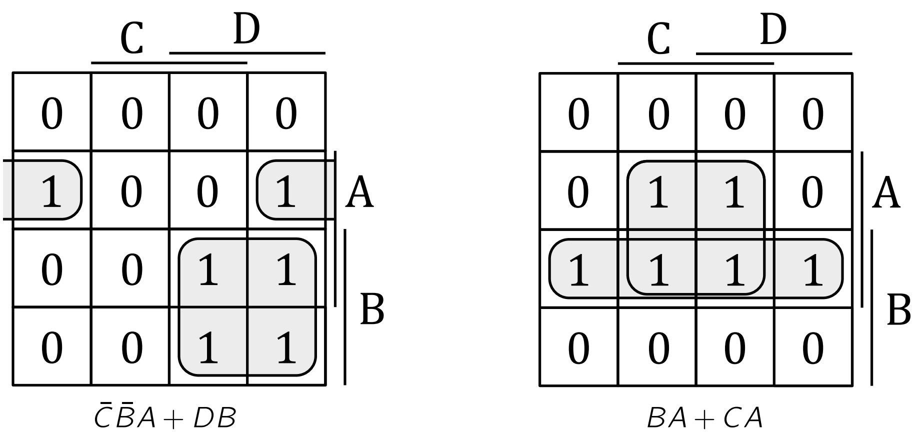


FIGURE 6 – Représentation de produits dans une table de Karnaugh

3.3 Définition : impliquant premier

Un impliquant premier est un produit de facteurs qui n'est pas inclus dans un produit plus grand.

Exemple

La figure 7 présente dans une table de Karnaugh un produit de facteurs inclus dans un ensemble plus grand.

Dans ce tableau, l'implicant DBA est contenu dans l'implicant premier DA et n'est donc pas un implicant premier.

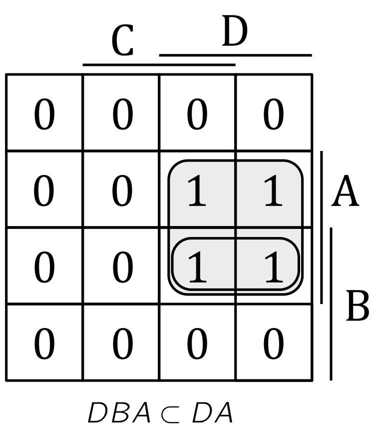


FIGURE 7 – Implicant premier

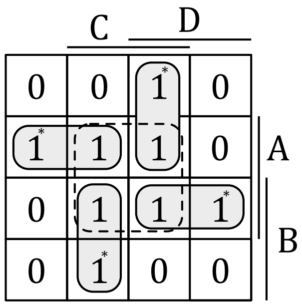
3.4 Définition : impliquant premier essentiel

Un implicant premier essentiel est un implicant premier dont une partie n'appartient qu'à lui et ne se retrouve donc dans aucun autre implicant premier.

Exemple

La figure 8 présente dans une table de Karnaugh un implicant premier de 4 cases, lequel est inutile puisque tous ses éléments sont contenus dans l'un des implicants premiers essentiels, marqués d'une astérisque.

Il est donc important de traiter d'abord les implicants premiers essentiels avant de s'occuper des autres implicants premiers, même s'ils sont de taille plus importante.



$A\bar{B}\bar{D} + CD\bar{B} + ABD + BC\bar{D} \cancel{+ CA}$

FIGURE 8 – Implicants premiers essentiel

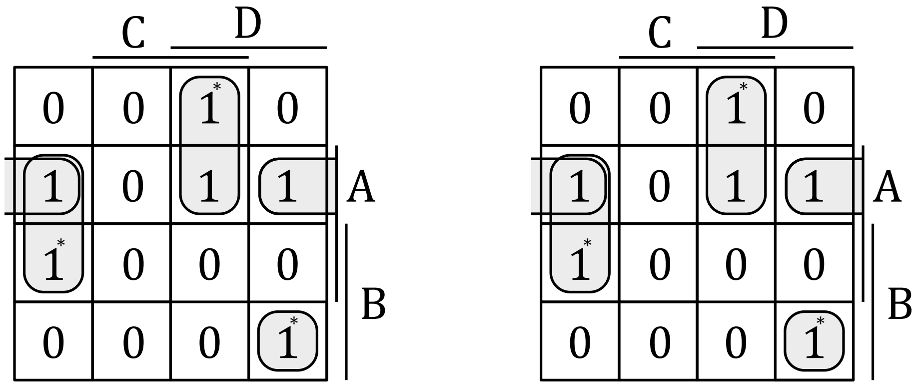
3.5 Simplification sous forme de somme de produits

La méthode de simplification sous forme de somme de produits se fait comme suit :

- remplir la table de Karnaugh de la fonction logique à simplifier,
- rechercher tous les implicants premiers essentiels et écrire le polynôme formé de la somme de ces implicants,
- pour les '1' non encore inclus dans les implicants premiers essentiels, trouver un nombre minimal d'implicants premiers pouvant les inclure et ajouter la somme de ces implicants au polynôme déjà développé jusqu'ici.

Exemple

La figure 9 présente cette méthode de simplification sous forme de somme de produits. La première table de Karnaugh contient tous les implicants premiers essentiels, marqués d'une astérisque. La deuxième table présente différents implicants premiers utilisables pour inclure la partie non encore comprise dans les implicants premiers essentiels.



$DC\bar{B}^* + \bar{D}\bar{C}A^* + D\bar{C}B\bar{A}^*$ $\qquad$ $D\bar{B}A/\bar{C}\bar{B}A$

$$\begin{cases} DC\bar{B}^* + \bar{D}\bar{C}A^* + D\bar{C}B\bar{A}^* + D\bar{B}A \\ DC\bar{B}^* + \bar{D}\bar{C}A^* + D\bar{C}B\bar{A}^* + \bar{C}\bar{B}A \end{cases}$$

FIGURE 9 – Simplification sous forme de somme de produits

3.6 Simplification de fonctions incomplètement définies

La simplification sous forme de somme de produits peut prendre avantage d'une définition incomplète dans le but de réduire l'expression polynomiale résultante.

La méthode de simplification est la même que celle pour des fonctions complètement définies, mais les états indéterminés sont inclus dans les implicants ou non selon qu'ils permettent d'augmenter ou non la taille des implicants premiers.

3.6.1 Définition : fonction incomplètement définie

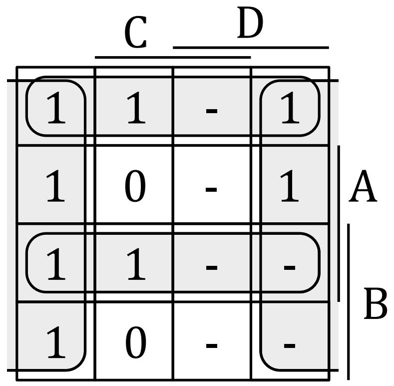
Une fonction incomplètement définie ne spécifie pas la valeur de sa sortie pour toutes les combinaisons possibles des entrées.

> Cette définition incomplète provient soit d'un cahier des charges où le résultat de la fonction n'est pas pris en compte dans certains cas (conditions dites “don't care”), soit d'un système où certaines combinaisons des entrées ne peuvent se présenter (conditions dites “never happens”).

Exemple

La figure 10 présente la simplification sous forme de somme de produits d'une fonction incomplètement définie.

Les états qui étaient indéfinis avant simplification recoivent une valeur déterminée après simplification. Ceux qui sont inclus dans les implicants de la figure sont assignés à '1' et les autres sont assignés à '0'.



$\bar{C} + \bar{B}\bar{A} + BA$

FIGURE 10 – Simplification d'une fonction incomplètement définie

4 Simplification de fonction OU-exclusif

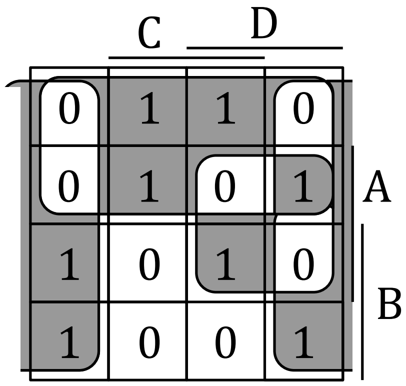
4.1 Représentation d'un OU-exclusif de produits

La fonction OU-exclusif donne une indication de parité : si le nombre de ses entrées à '1' est impair, la sortie vaut '1', sinon elle vaut '0'.

Dans une table de Karnaugh, le OU-exclusif de plusieurs implicants vaut '1' dans les cases recouvertes par un nombre impair d'ensembles et '0' dans les cases recouvertes par aucun ou par un nombre pair d'ensembles.

Exemple

La figure 11 présente l'aspect d'un OU-exclusif d'implicants dans une table de Karnaugh.



$\bar{C} \oplus \bar{B} \oplus DA$

FIGURE 11 – OU-exclusif d'implicants dans une table de Karnaugh

4.2 Simplification sous forme de OU-exclusif de produits

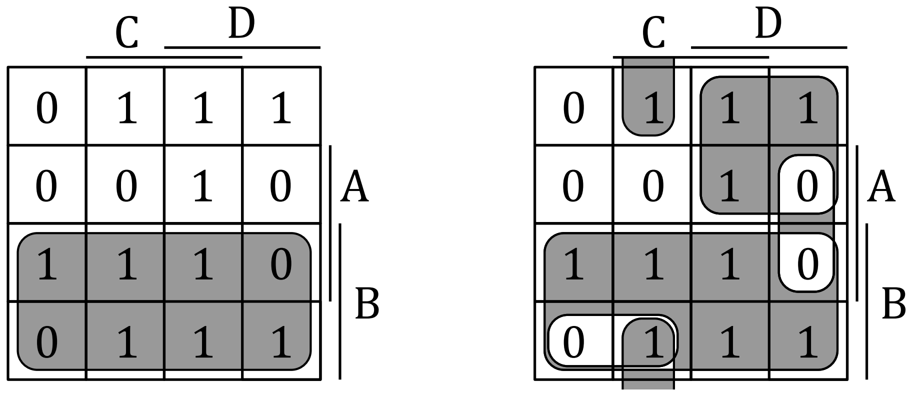
La simplification sous forme de OU-exclusif de produits ne se fait pas avec une méthode aussi directe que pour la simplification sous forme de somme de produits.

Elle peut se faire en commencant par un grand ensemble dont tous les éléments ne sont pas nécessairement à '1'. A partir de ce choix initial, d'autres ensembles sont choisis de manière à inverser les '0' en excédent et à inclure les '1' manquants.

Exemple

La figure 12 présente cette méthode de simplification sous forme de OU-exclusif de produits.

La simplification a commencé par le choix de l'ensemble B. Le choix de départ à partir de l'ensemble C ou D permet de trouver une expression plus simple avec un OU-exclusif de 4 produits



$B \oplus \bar{B}D \oplus D\bar{C}A \oplus \bar{D}B\bar{A} \oplus \bar{D}C\bar{A}$

FIGURE 12 – Simplification sous forme de OU-exclusif de produits

5 Fonctions avec un nombre élevé d'entrées

5.1 Limitation de l'utilisation de la table de Karnaugh

La simplification de fonctions par la méthode graphique de la table de Karnaugh se limite pratiquement à des fonctions de 6 variables d'entrée. Différents algorithmes existent pour réaliser cette tâche à l'aide d'ordinateurs.

La mise en équations de fonctions à un nombre élevé d'entrée peut se faire par simplification algébrique ou par réduction en blocs hiérarchiques.

5.2 Simplification algébrique

La simplification algébrique consiste à écrire une fonction sous forme d'équation et à la transformer à l'aide des propriétés de l'algèbre booléenne : associativité, commutativité, distributivité, De Morgan, ...

Exemple : comparaison d'égalité

Soient deux nombres entiers, A et B, exprimés dans la base binaire et codés sur 8 bits.

$$\begin{aligned} A &= a_7 * 2^7 + a_6 * 2^6 + \cdots + a_1 * 2^1 + a_0 * 2^0 \\ B &= b_7 * 2^7 + b_6 * 2^6 + \cdots + b_1 * 2^1 + b_0 * 2^0 \end{aligned} \tag{1}$$

Pour déterminer si ces 2 nombres sont égaux, il faut comparer les bits de poids identique de chaque nombre.

L'égalité des 2 nombres peut s'écrire :

$$\begin{aligned} A = B &\Leftrightarrow (a_7 = b_7) * (a_6 = b_6) * \cdots * (a_1 = b_1) * (a_0 = b_0) \\ A = B &\Leftrightarrow \overline{(a_7 \oplus b_7)} * \overline{(a_6 \oplus b_6)} * \cdots * \overline{(a_1 \oplus b_1)} * \overline{(a_0 \oplus b_0)} \\ A = B &\Leftrightarrow (\bar{a}_7\bar{b}_7 + a_7 b_7) * (\bar{a}_6\bar{b}_6 + a_6 b_6) * \cdots * (\bar{a}_0\bar{b}_0 + a_0 b_0) \end{aligned} \tag{2}$$

Cette fonction peut se réaliser à l'aide de portes ET et OU selon l'eq. 2 ou se développer sous forme de somme de produits ou sous une autre forme encore.

5.3 Réduction en blocs de taille maîtrisable

Une approche typique de l'ingénieur consiste à réduire un système complexe en blocs suffisamment simples pour être maîtrisés avec les techniques à disposition.

Exemple : comparaison d'égalité

La comparaison d'égalité de 2 nombre à 8 bits peut se faire en partageant chacun des nombres en 2, puis en 2 encore.

On obtient alors des blocs avec au maximum 4 entrées, parfaitement maîtrisables avec la technique des tables de Karnaugh.

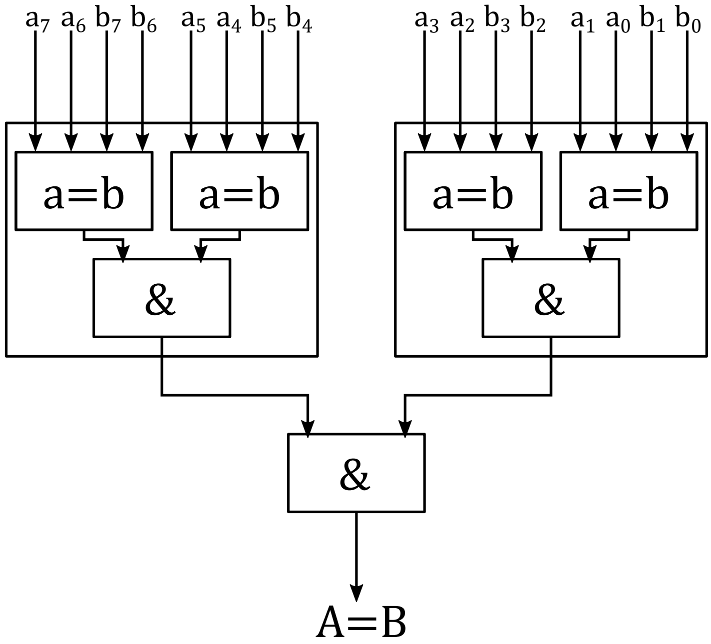
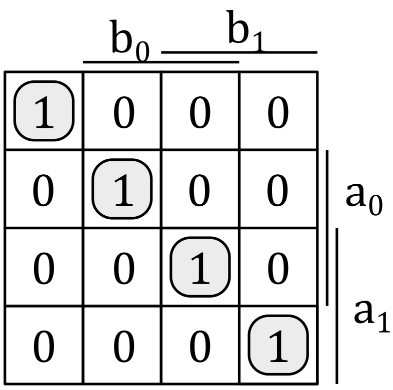


FIGURE 13 – Réduction en blocs de taille maîtrisable

L'expression ploynomiale de la fonction du bloc comparant 2 bits de chaque nombre est donnée à la figure 14.



$$\bar{a}_0\bar{b}_0\bar{a}_1\bar{b}_1 + a_0 b_0\bar{a}_1\bar{b}_1 + \bar{a}_0\bar{b}_0 a_1 b_1 + a_0 b_0 a_1 b_1$$

FIGURE 14 – Simplification de la comparaison de 2 bits par nombre

5.4 Systèmes itératifs

Le traitement de nombres peut s'appuyer sur des algorithmes mathématiques qui permettent de développer des opérations sur des nombres en opérant chiffre par chiffre et en posant des résultats intermédiaires : reports pour comparaisons et additions, produits partiels pour la multiplication, ...

Exemple : comparaison d'égalité

La comparaison de 2 nombres peut se faire de manière itérative, avec un report qui indique si les 2 nombres sont égaux jusqu'au point où se trouve ce report.

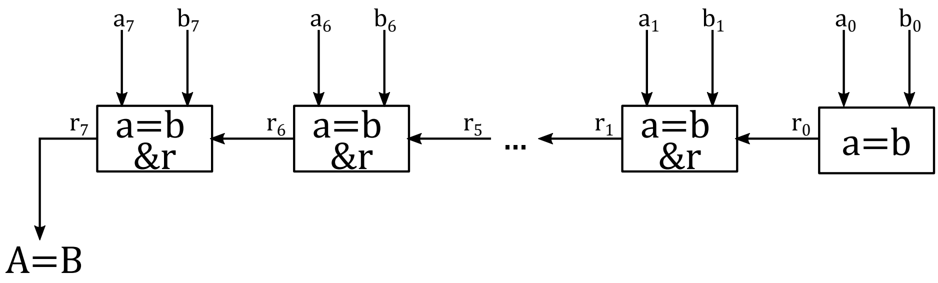
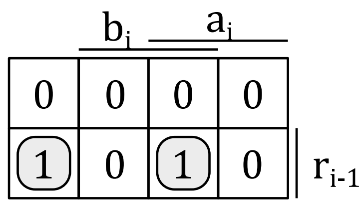


FIGURE 15 – Système itératif

Le premier bloc peut soit être utilisé tel quel avec son entrée de report fixée à $r_{-1} = 1$, soit être simplifié à l'expression $r_0 = \bar{a}_0\bar{b}_0 + a_0b_0$.

L'expression ploynomiale de la fonction du bloc itératif est donnée à la figure 16.



$$r_i = \bar{a}_i\bar{b}_i r_{i-1} + a_i b_i r_{i-1}$$

FIGURE 16 – Simplification de la fonction itérative

> Du fait des conditions de bord, le premier et le dernier bloc d'un système itératif peuvent être différents des autres blocs de la chaîne. Le premier bloc est alors soit à simplifier soit à utiliser avec des entrées constantes. Le dernier bloc peut être simplifié lorsqu'il a des sorties inutilisées.

Références

[1] Suhail Almani. *Electronic Logic Systems*. second edition. New-Jersey : Prentice-Hall, 1989.

[2] Jean Michel Bernard et Jean Hugon. *Pratique Des Circuits Logiques*. quatrième édition. Paris : Eyrolles, 1987.

[3] Klaus Elektronik Bd Beuth. *Digitaltechnik*. Vogel Buchverlag, 1990. isbn : 3-8023-1755-6.

[4] Michael D. Ciletti et M. Morris Mano. *Digital Design*. second edition. New-Jersey : Prentice-Hall, 2007.

[5] David J. Comer. *Digital Logic and State Machine Design*. Saunders College Publishing, 1995.

[6] Donald L. Dietmeyer et R. David. “Logic Design of Digital Systems”. In : *Journal of Dynamic Systems, Measurement, and Control* 94.2 (1er juin 1972), p. 174-174. issn : 0022-0434. doi : 10.1115/1.3426575. url : https://doi.org/10.1115/1.3426575 (visité le 01/06/2021).

[7] William I. Fletcher. *Engineering Approach to Digital Design*. New-Jersey : Prentice-Hall, 1980.

[8] Marcel Gindre et Denis Roux. *Electronique Numérique, Logique Combinatoire et Technologie*. Paris : McGraw-Hill, 1987.

[9] Randy H. Katz et Gaetano Borriello. *Contemporary Logic Design*. California : The Benjamin/Cummings Publishing Company Inc, 2005.

[10] David Lewin et Douglas Protheroe. *Design of Logic Systems*. second edition. Hong Kong : Springer, 2013.

[11] Daniel Mange. *Analyse et synthèse des systèmes logiques*. Editions Géorgi. T. Traité d'électricité, volume V. St Saphorin : PPUR presses polytechniques, 1995. 362 p. isbn : 978-2-88074-045-0. Google Books : 5NSdD4GRl3cC.

[12] Clive Maxfield. *Bebop to the Boolean Boogie*. Elsevier, 2009. isbn : 978-1-85617-507-4. doi : 10.1016/B978-1-85617-507-4.X0001-0. url : https://linkinghub.elsevier.com/retrieve/pii/B9781856175074X00010 (visité le 27/05/2021).

[13] Ronald J. Tocci et André Lebel. *Circuits Numériques : Théorie et Applications*. deuxième édition. Ottawa : Editions Reynald Goulet inc. / Dunod, 1996.

[14] John F. Wakerly. *Digital Design : Principles And Practices*. 3rd edition. Prentice-Hall, 2008. isbn : 0-13-082599-9.