Circuits logiques combinatoires et séquentiels

Guy Bégin

7 novembre 2022

Simplification logique

Objectifs

- Pouvoir formuler une expression logique en forme canonique *Produit de sommes* ou *Somme de produits*, et convertir entre les deux formes
- Pouvoir simplifier une expression au moyen d'un diagramme de Karnaugh
- Pouvoir simplifier une expression par la méthode Quine-McCluskey
- Être familier avec les approches d'implémentation des fonctions simplifiées

Expressions équivalentes

- Un des aspects ennuyeux avec les expressions logiques est que la correspondance entre expression et fonction logique n'est pas biunivoque : plusieurs expressions différentes peuvent correspondre à une seule et même fonction.
- De plus, certaines des expressions équivalentes peuvent être plus complexes que d'autres.
- Lorsque vient le temps d'implémenter avec des portes une fonction logique, il est la plupart du temps plus efficace d'implémenter selon une expression plus simple, voir minimale.
- On doit donc considérer des approches systématiques et efficaces pour simplifier les expressions logiques.

Expressions équivalentes . . . 2

- Quand une expression Booléenne est implémentée avec des portes logiques, chaque terme nécessite une porte et chaque variable au sein d'un terme correspond à une entrée de la porte.
- On appelle littéral une variable qui apparaît dans un terme, sous forme complémentée ou non.
- Par exemple, l'expression $F = x'y'z + xz + xy'z$ compte huit littéraux.
- Si on réduit le nombre de termes, le nombre de littéraux, ou les deux, on obtiendra une expression qui sera plus simple à implémenter avec des portes.

Formes canoniques

Minterms et maxterms

- Dans une expression, une variable x peut apparaître telle qu'elle x ou complémentée x'.
- Si on considère les combinaisons possibles de deux variables via un opérateur ET, on a alors quatres possibilités : $x'y', x'y, xy', xy$.
- Chacun de ces quatres termes s'appelle un minterm.
- De façon équivalente (duale, en vérité), n variables reliées par une fonction OU peuvent donner lieu à 2^n termes distincts, appelés maxterms.

Formes canoniques . . . 2

- De façon générale, pour n variables, on aura 2^n minterms ou 2^n maxterms différents possibles.
- Pour étiqueter les différents minterms ou maxterms, on a établi une convention de numérotation.
- Le numéro d'étiquette d'un minterm est construit de la façon suivante.
- Une variable complémentée amène un bit d'étiquette 0, une variable telle qu'elle amène un bit d'étiquette 1.
- En ordonnant les bits selon l'ordre alphabétique des variables, on obtient un vecteur de bits qui donnera le numéro à assigner au minterm.

Formes canoniques . . . 3

- Par exemple, le minterm $xy'z$ donnera l'étiquette 101, donc le numéro de minterm (en équivalent décimal) 5.
- La règle pour les maxterms est duale : une étiquette 0 pour une variable telle qu'elle, et une étiquette 1 pour une variable complémentée.
- Chaque maxterm est le complément du minterm correspondant (de même numéro), et *vice versa*.

Formes canoniques . . . 4

- Dans le tableau 1, on montre les symboles de la forme m_j pour les mintems et M_j pour les maxterms, avec j qui est l'équivalent décimal de la combinaison de bits correspondante.

Table 1 – Minterms et maxterms pour trois variables

x	y	z	Minterm	Symb.	Maxterm	Symb.
0	0	0	$x'y'z'$	m_0	$x+y+z$	M_0
0	0	1	$x'y'z$	m_1	$x+y+z'$	M_1
0	1	0	$x'yz'$	m_2	$x+y'+z$	M_2
0	1	1	$x'yz$	m_3	$x+y'+z'$	M_3
1	0	0	$xy'z'$	m_4	$x'+y+z$	M_4
1	0	1	$xy'z$	m_5	$x'+y+z'$	M_5
1	1	0	xyz'	m_6	$x'+y'+z$	M_6
1	1	1	xyz	m_7	$x'+y'+z'$	M_7

Formes canoniques . . . 5

- Pour la fonction F_1 dont le tableau de vérité est le suivant :

Table 2 – Fonction de trois variables

x	y	z	F_1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Somme de produits

- on peut donc écrire

$$F_1 = xy'z' + xy'z' + xy'z + xyz' + xyz = m_2 + m_4 + m_5 + m_6 + m_7$$

- puisque ce sont les termes pour lesquels la fonction vaut 1.
- Cette forme d'expression est une forme canonique appelée *somme de produits*.

Notation compacte ... 2

- Pour simplifier la notation, on peut écrire de façon plus compacte

$$F_1 = \sum(2, 4, 5, 6, 7)$$

- où on ne met que les numéros des minterms participant à la somme.

Complément d'une fonction

- Si on veut exprimer le complément d'une fonction, on peut lire dans le tableau de vérité les combinaisons pour lesquelles la fonction vaut 0.
- En prenant un minterm pour chaque combinaison où la fonction vaut 0 et en faisant un OU de ces termes, on obtient une expression en *somme de produits* pour le complément de la fonction.
- Ainsi, pour la fonction F_1', on a

$$F_1' = m_0 + m_1 + m_3 = x'y'z' + x'y'z + x'yz$$

Produit de sommes

- Si on complémente F_1', on obtiendra naturellement F_1.
- En appliquant le théorème de DeMorgan à chaque terme, on trouve

$$F_1 = (x + y + z)(x + y + z')(x + y' + z') = M_0 \cdot M_1 \cdot M_3$$

- Cette forme d'expression est aussi une forme canonique appelée *produit de sommes*.
- Pour simplifier la notation, on peut écrire de façon plus compacte

$$F_1 = \prod(0, 1, 3)$$

- où on ne met cette fois que les numéros des maxterms participant au produit.

Somme de produits

- Pour n variables binaires, on a 2^n minterms différents possibles.
- Les minterms qui participent à la somme dans l'expression en forme canonique *somme de produits* sont ceux qui produisent un 1 dans le tableau de vérité de la fonction.
- Puisque la fonction peut valoir 0 ou 1 pour chaque minterm, le nombre total de fonctions différentes qui peuvent être définies avec n variables est de 2^{2^n}.

Conversion vers forme canonique somme de produits

- Si on veut convertir en forme canonique *somme de produits* l'expression pour une fonction qui ne serait pas sous cette forme, on commence par faire l'expansion de l'expression en forme *somme de produits*.
- Ensuite, on vérifie chaque terme pour voir si toutes les variables en font partie.
- S'il manque une ou des variables, on peut faire un ET du terme avec une expression du type $x + x'$ dans laquelle x est une variable manquante.
- Ce ET ne change pas la valeur de la fonction puisque $x + x' = 1$.
- Évidemment, on peut toujours trouver la formulation en forme canonique en se basant sur le tableau de vérité.

Conversion vers forme canonique produit de sommes

- Si on veut convertir en forme canonique *produit de sommes* l'expression pour une fonction qui ne serait pas sous cette forme, on commence par faire l'expansion de l'expression en forme *produit de sommes*.
- On peut avantageusement faire appel à la distributivité de $+$ sur $\cdot$ pour ce faire.
- Ensuite, on vérifie chaque terme pour voir si toutes les variables en font partie.
- S'il manque une ou des variables, on peut faire un OU du terme avec une expression du type $x \cdot x'$ dans laquelle x est une variable manquante.
- Ce OU ne change pas la valeur de la fonction puisque $x \cdot x' = 0$.

Conversion entre formes canoniques

- Prenons notre exemple précédent $F_1 = \sum(2, 4, 5, 6, 7)$.
- On sait que $F_1' = \sum(0, 1, 3)$.
- Si on prend le complément de F_1' par le théorème de DeMorgan, on obtient
  $F_1 = (m_0 + m_1 + m_3)' = m_0' \cdot m_1' \cdot m_3' = M_0 \cdot M_1 \cdot M_3 = \prod(0, 1, 3)$.
- En effet, de minterm à maxterm, on a $m_j' = M_j$.
- Le maxterm d'indice j est le complément du minterm de même indice j, et *vice versa*.

Formes standard

- Les expressions canoniques en *somme de produits* et en *produit de sommes* ne sont généralement pas simples, car toutes les variables doivent être présentes.
- Pour l'implémentation, on cherchera des expressions en formes *somme de produits* ou *produit de sommes* dans lesquelles les termes pourront être simplifiés.
- C'est-à-dire que les termes pourront comporter une, deux, trois, etc. variables plutôt qu'obligatoirement toutes les variables.
- Toujours pour notre fonction exemple, on peut écrire

$F_1 = x + yz'$

Implémentation à deux niveaux

- Lorsqu'on implémente une telle fonction avec des portes logiques, il faut une porte ET pour chaque terme produit (qui comporte plus d'une variable) et une porte OU pour faire la somme finale.
- On obtient une implémentation à deux niveaux.
- De façon duale, on peut également obtenir une formulation en *produit de sommes* qui aboutira à une implémentation à deux niveaux avec une porte OU par terme et une porte ET pour le produit final.

Objectifs de minimisation

Étant donné une fonction logique de n variables $z(x_1, x_2, \ldots, x_n)$, on veut déterminer une expression pour cette fonction sous la forme *Somme de Produits* (S de P) ou *Produit de Sommes* (P de S) qui

1. comporte un nombre minimum de termes produits (pour la forme S de P) ou de termes sommes (pour la forme P de S) ;
2. est telle qu'aucune expression pour z comportant le même nombre de termes n'utilise moins de littéraux.

Diagrammes de Karnaugh

- Une méthode visuelle permet de simplifier l'expression logique d'une fonction en systématisant une procédure faisant appel à un diagramme qui fait ressortir les simplifications possibles.
- Un diagramme de Karnaugh (diag-K) est constitué d'un regroupement de cellules carrées, chaque cellule correspondant à un minterm possible.
- Les cellules sont organisées de façon à ce que lorsqu'on passe d'une cellule à une cellule adjacente (horizontalement ou verticalement), un seul bit du minterm change, ce qui revient à dire qu'une seule variable passe de telle qu'elle à complémentée.
- Cela fait en sorte que si la fonction est 1 pour deux minterms adjacents, la somme des deux minterms pourra être simplifiée en un seul terme dans lequel la variable correspondant au bit qui change est absente.

Diagrammes de Karnaugh ... 2

- Par exemple, on pourrait avoir pour deux minterms adjacents

$m_5 + m_7 = xy'z + xyz = xz(y' + y) = xz.$

- Ici les deux minterms adjacents diffèrent par la variable y, qui sera donc supprimée du terme produit résultant.

Diagrammes de Karnaugh : deux variables

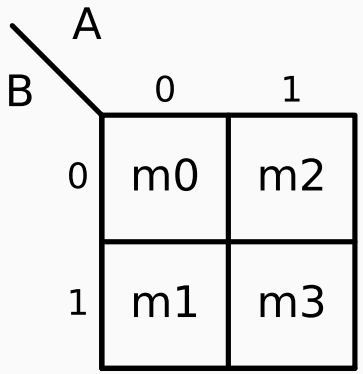


Figure 1 – Diag-K à deux variables

Diagrammes de Karnaugh : trois variables

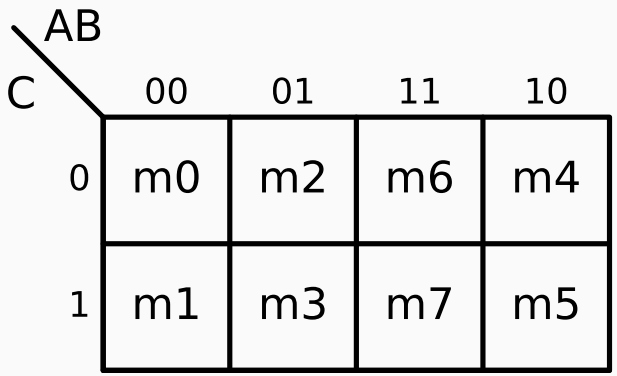


Figure 2 – Diag-K à trois variables, avec minterms

Diagrammes de Karnaugh : trois variables . . . 2

- Sur un diag-K à trois variables, on voit que les bits AB sont ordonnés selon un code Gray, de façon à ce qu'un seul des bits change lorsqu'on passe d'une cellule à la suivante horizontalement.
- L'adjacence se poursuit en bout de diagramme : par exemple, la cellule 100 (m_4) est adjacente horizontalement à la cellule 000 (m_0).

Diagrammes de Karnaugh : trois variables . . . 3

- On peut imaginer le diagramme comme replié sur lui-même pour visualiser cette adjacence.

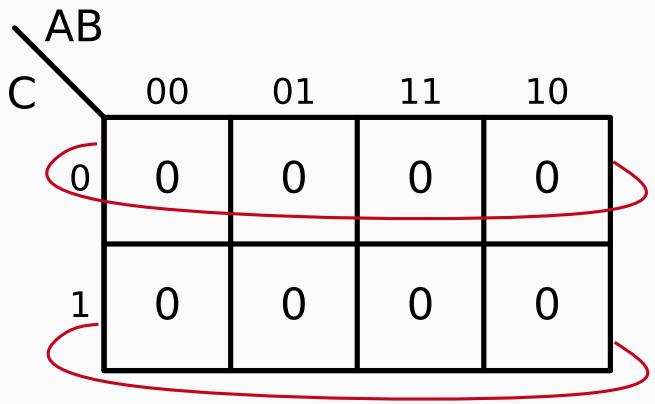


Figure 3 – Diag-K avec adjacence horizontale

Diagrammes de Karnaugh : quatre variables et plus

- Sur un diag-K à quatre variables, l'adjacence repliée est autant horizontale que verticale.
- Pour plus de quatre variables, il devient difficile d'utiliser cette méthode : les diagrammes sont de grande taille et surtout, les règles d'adjacence ne sont plus aussi facilement observables.
- Les risques d'erreurs sont plus grands.

Diagrammes de Karnaugh : quatre variables

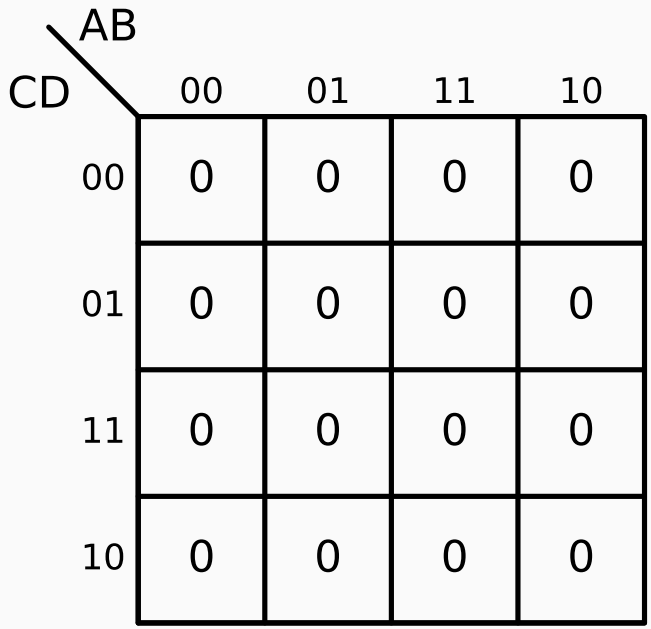


Figure 4 – Diag-K à quatre variables

Procédure de simplification

Pour utiliser un diag-K pour minimiser une fonction logique :

1. Les minterms de la fonction à minimiser sont identifiés en insérant un 1 dans la cellule correspondant à chaque minterm.
2. On cherche dans le diagramme pour trouver des regroupement de deux cellules adjacentes qui sont marquées d'un 1.
3. Chaque groupe de deux cellules 1 adjacentes est marqué comme groupe. Un même minterm peut être incorporé à plus d'un groupe.
4. Il est aussi possible de regrouper les groupes : deux groupes de 2 qui sont adjacents peuvent ainsi se regrouper en un groupe de 4. Les tailles de groupes doivent être des puissances de 2. Il est ainsi possible de créer des groupes de 2, 4, 8 ou 16 minterms.

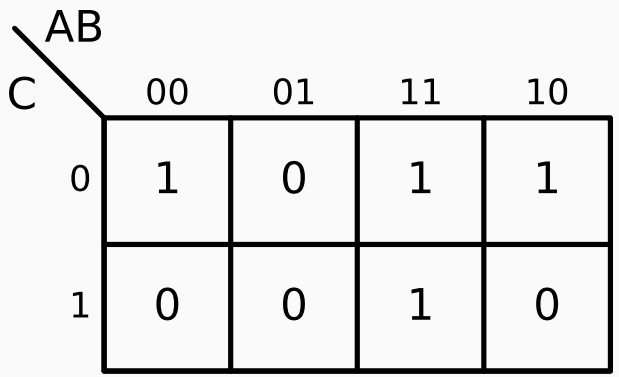
Procédure de simplification ... 2

5. Une fois tous les regroupements identifiés, il est possible de lire l'expression de la fonction en *somme de produits*. Chaque groupement correspond à un terme produit, et la ou les variables dont le bit ne change pas dans le groupe sont conservées ; les autres sont éliminées.

Exemple de simplification

Considérons par exemple la fonction $F(A, B, C) = \sum(0, 4, 6, 7)$.

- Après la première étape, on obtient



Exemple de simplification ... 2

- Après les regroupements, on obtient un diag-K comportant trois regroupements

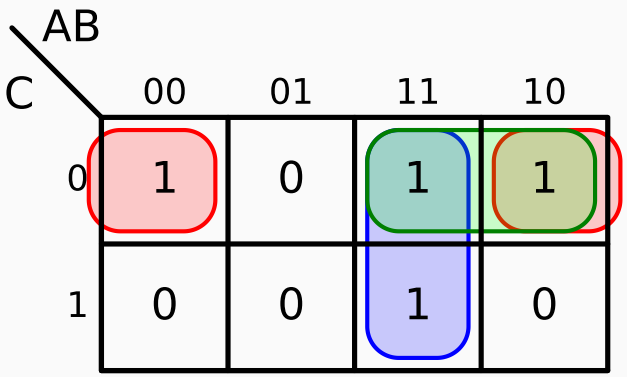


Figure 5 – Diagramme après les regroupements

Exemple de simplification ... 3

- Le groupe en rouge correspond au produit $B'C'$, celui en bleu correspond à AB et celui en vert correspond à AC'.
- L'expression finale en *somme de produits* est donc $F = B'C' + AB + AC'$.

Cas facultatifs

- Certaines fonctions sont incomplètement définies, dans le sens où certaines combinaisons d'entrées ne se produiront jamais ou seront sans conséquences si elles se produisent.
- On parle de cas indifférents ou facultatifs (en anglais, *don't care*).
- Pour la simplification, ces cas pourront être traités tantôt comme des 0, tantôt comme des 1, selon ce qui sera le plus avantageux.

Cas facultatifs ... 2

- Pour tenir compte de ces cas, les minterms seront notés avec un X dans le diagramme de Karnaugh.
- Dans l'exemple à quatre variables suivant, sur deux cas facultatifs, un seul, celui correspondant à m_7, a été traité comme un 1, ce qui a permis de créer le regroupement en bleu.
- L'autre cas facultatif, correspondant à m_2, n'a pas servi dans un regroupement, ce qui signifie qu'il a été traité comme un 0.
- La fonction résultante est donc $AC'D' + BD + AB$.

Cas facultatifs .. 3

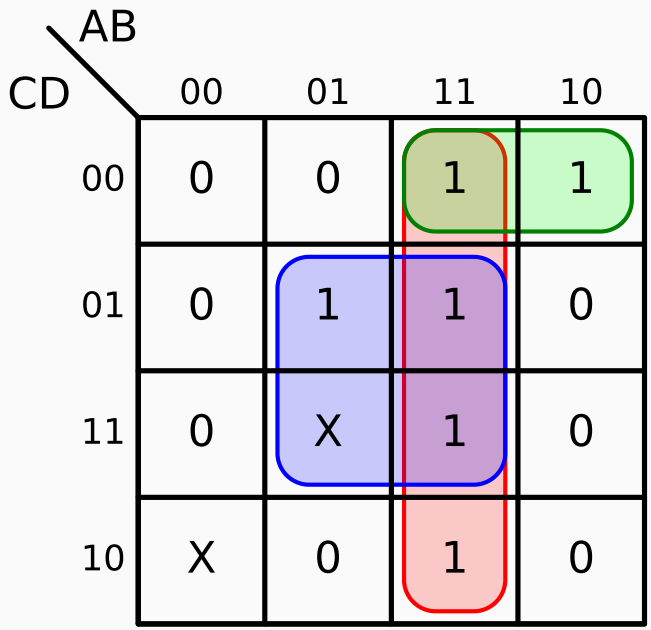


Figure 6 – Diag-K avec cas facultatifs

Impliquants

Le choix des regroupements à utiliser doit toujours viser à s'assurer que :

1. Tous les minterms de la fonction sont couverts par les regroupements choisis.
2. Le nombre de termes retenus pour l'expression est minimal.
3. Il n'y a pas de termes redondants, c'est-à-dire, qui couvrent uniquement des minterms déjà couverts.

Impliquants . . . 2

- Il y a parfois des plus d'une expression qui rencontre ces critères.
- Il est possible de systématiser le choix des termes en prenant en compte le caractère essentiel des termes.

Impliquants . . . 3

- Soit $p(X)$ un terme produit de littéraux tirés de l'ensemble de variables X.

Si, pour une fonction logique $z(X)$ définie pour le même ensemble de variables, la relation

pour tout A tel que $p(A) = 1$, $z(A) = 1$

tient, alors p est un impliquant de z.

- Cela signifie que la vérité du terme produit p implique celle de z.
- *Tout minterm de p est aussi un minterm de z.*

Exemple pour impliquants

$$z_1 = ab + bc + ab'c$$

ab, bc, $ab'c$ sont des impliquants évidents de z_1.

$a'bc$, abc', abc, ac sont aussi des impliquants de z_1.

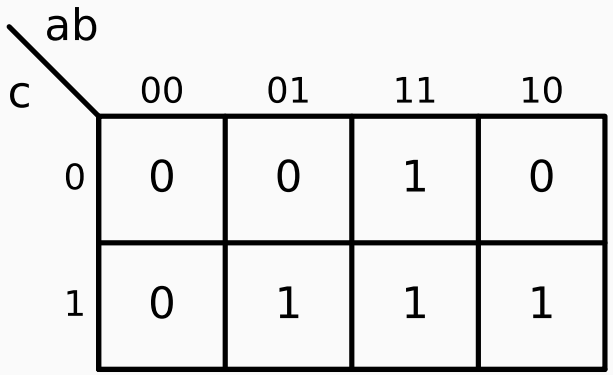


Figure 7 – Diag-K pour l'exemple des impliquants

Impliquant premier

- Un impliquant p de la fonction z est premier si n'importe quel terme produit obtenu de p en supprimant un littéral n'est pas un impliquant de z.
- Ici, ab est un impliquant premier de z_1 car ni a ni b ne sont des impliquants de z_1.
- Mais $ab'c$ n'est pas un impliquant premier de z_1 car ac est un impliquant de z_1.
- Sur un diagramme de Karnaugh, un impliquant premier (i.p.) est un groupe qui n'est contenu dans aucun autre groupe plus grand.

Couverture d'une fonction

- Un sous-ensemble d'i.p. qui contient tous les minterms d'une fonction couvre la fonction.

Une couverture minimale est une couverture avec :

1. le nombre minimal d'impliquants premiers,
2. le moins de littéraux parmi les couvertures avec nombre minimum d'implicants.

Impliquant premier essentiel

- Un i.p. est essentiel si et seulement si il couvre un minterm de la fonction qui ne peut être couvert par un autre i.p. de la fonction. Une couverture de la fonction doit contenir tous les impliquants premiers essentiels (i.p.e.).
- Un impliquant premier absolument inessentiel est un i.p. qui couvre des minterms qui sont tous couverts par les i.p.e. de la fonction.

Sélection des impliquants

Règles de sélection des impliquants :

1. Mettre de côté tous les i.p.e. Ils seront utilisés dans la solution finale.
2. Éliminer tous les i.p. absolument inessentiels.
3. Il reste à choisir parmi les i.p. inessentiels pour obtenir une couverture minimale.

Lorsque le problème est de taille réduite, on peut faire une recherche exhaustive de toutes les solutions possibles pour choisir la solution minimale.

Minimisation avec cas facultatifs

1. Lorsqu'on détermine les i.p., on doit considérer les X comme des 1, de façon à pouvoir utiliser les i.p. rendus possibles par les cas facultatifs.
2. Lors de la sélection des i.p. pour obtenir une couverture minimale, on ne doit pas ne pas essayer de couvrir les X.

Minimisation avec plusieurs fonctions

Si deux fonctions z_i etz_j ont des expressions minimales qui comportent un terme commun, une seule porte suffira pour générer ce terme au profit des deux fonctions.

Exemple de minimisation avec plusieurs fonctions

$$z_1 = ac + a'bc' + a'c'd$$

et

$$z_2 = ac + a'bc'd' + a'b'c'd$$

cd \ ab	00	01	11	10
00	0	1	0	0
01	1	1	0	0
11	0	0	1	1
10	0	0	1	1

Figure 8 – Fonction z_1

Exemple de minimisation avec plusieurs fonctions ... 2

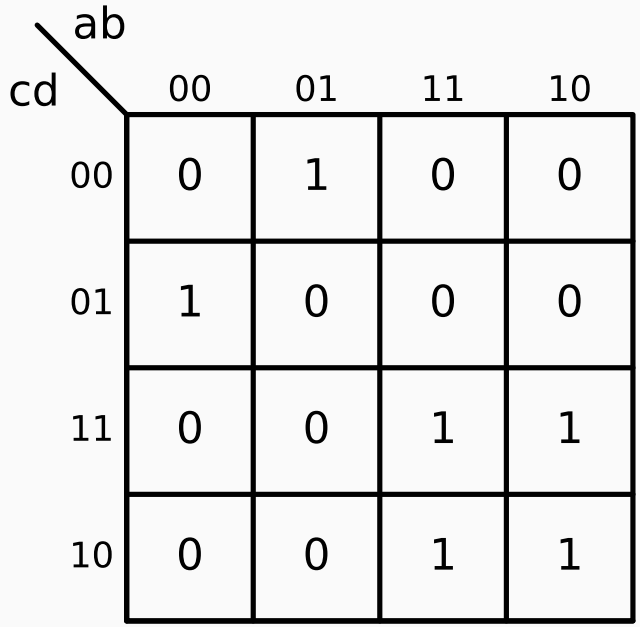


Figure 9 – Fonction z_2

Exemple de minimisation avec plusieurs fonctions ... 3

- Il est alors préférable de réutiliser les termes communs et de générer seulement les termes manquants pour la seconde fonction.
- Dans cet exemple, le terme ac sera calculé une seule fois.
- Les termes $a'bc'd'$ et $a'b'c'd$ sont nécessaires pour z_2.
- Alors, pour z_1, on fera

$$z_1 = ac + a'bc'd' + a'b'c'd + a'bc'd$$

- qui ne nous coûtera que le dernier terme produit et une somme de quatre termes.

Tableau de couverture Quine-McCluskey

- La méthode de Quine-McCluskey systématise la sélection des impliquants en se basant sur des relations qui s'expriment en fonction d'un tableau de couverture.
- Un tableau de couverture comporte une ligne pour chaque i.p. et une colonne pour chaque minterm de la fonction à minimiser z.
- Un ✓ est inscrit à l'intersection de la ligne i et de la colonne j si l'i.p. P_i de la ligne i couvre le minterm m_j de la colonne j.

Problème de minimisation Q-M

Le problème de minimisation devient alors : trouver une couverture pour la fonction z qui

1. contient le nombre minimum de lignes
2. est telle qu'aucune autre couverture à nombre de ligne minimum comprend moins d'entrées 1 et 0 dans ses codes d'impliquants de ligne.

Utilisation du tableau de couverture

- Dans le tableau de couverture, on identifie facilement les i.p.e. par les colonnes qui ne contiennent qu'un ✓.
- L'i.p. qui couvre une colonne qui ne contient qu'un ✓ est un i.p.e.
- Puisque les i.p.e. doivent faire partie de la solution finale, toutes les colonnes couvertes par des i.p.e. seront couvertes dans n'importe quelle solution.
- On peut donc éliminer ces colonnes de la suite de la recherche de la solution, de même que les lignes correspondant aux i.p.e.
- On obtient ainsi un tableau de couverture réduit.
- Il ne faut cependant pas oublier de mettre les i.p.e. dans la solution finale.

Tableau de couverture réduit

- Le tableau de couverture réduit permet de se concentrer sur la sélection des i.p. dont la sélection n'est pas évidente *a priori*.
- Considérons pour illustrer la discussion le tableau de couverture réduit suivant.
- m_c est sans doute couvert pas un i.p.e. qui n'est pas montré ici.

Tableau de couverture réduit ... 2

Table 3 – Tableau réduit

	m_a	m_b	m_c	m_d	m_e	m_f	m_g	m_h
P_A		✓			✓		✓	✓
P_B	✓	✓				✓		✓
P_C	✓				✓		✓	✓
P_D		✓						✓
P_E	✓	✓		✓	✓	✓	✓	✓

Dominance de lignes

- Une ligne P_i domine une ligne P_j (noté $P_i \supseteq P_j$) si la ligne P_i contient un ✓ dans toutes les colonnes où la ligne P_j contient un ✓.
- Ici, on a $P_B \supseteq P_D$ mais P_B ne domine pas P_A.
- On peut voir aussi que P_E domine plusieurs lignes.

Dominance de lignes . . . 2

- En général, une P_i dominante contient plus de ✓ que P_j.
- Si elles ont le même nombre de ✓ (dans les mêmes colonnes), on a $P_i = P_j$.
- Il n'y a pas de cas d'égalité ici.
- Une ligne dominée par une autre peut être éliminée du tableau de couverture à condition que son nombre de littéraux soit supérieur ou égal à celui de la ligne dominante.

Dominance de colonnes

- Une colonne m_i domine une colonne m_j (noté $m_i \supseteq m_j$) si la colonne m_i contient un ✓ dans toutes les lignes où la colonne m_j contient un ✓.
- Ici, la colonne $m_h \supseteq m_g$ mais m_b ne domine pas m_a.
- Une colonne dominant une autre peut être éliminée du tableau de couverture, car le fait que la solution finale couvre la colonne dominée assure que la colonne dominante sera couverte aussi.
- Donc ici, la colonne m_h peut être éliminée.
- En cas d'égalité, comme on a ici pour $m_e = m_g$, on peut librement choisir quelle colonne éliminer.

Implémentation des fonctions simplifiées

- Les circuits logiques simplifiés en forme *produit de sommes* ou *somme de produits* sont souvent mis en oeuvre au moyen de portes NAND ou NOR plutôt qu'avec des portes ET et OU.
- La raison est qu'il est plus simple en pratique de réaliser ces portes.

Implémentation à deux niveaux

- Une fonction en forme *somme de produits* s'implémente évidemment avec des portes ET pour les produits et une porte OU pour la somme finale.
- Considérons par exemple $F = AB + CD$.

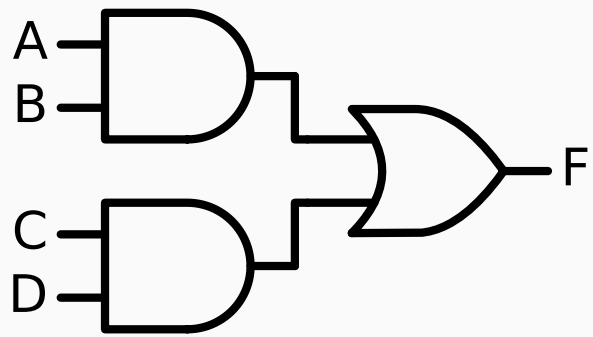


Figure 10 – *Somme de produits* pour $F = AB + CD$

Implémentation à deux niveaux ... 2

- La fonction peut aussi s'implémenter tout naturellement en faisant appel uniquement à des portes NAND.
- On peut vérifier facilement que le circuit suivant implémente la même fonction $F = ((AB)' \cdot (CD)')' = AB + CD$

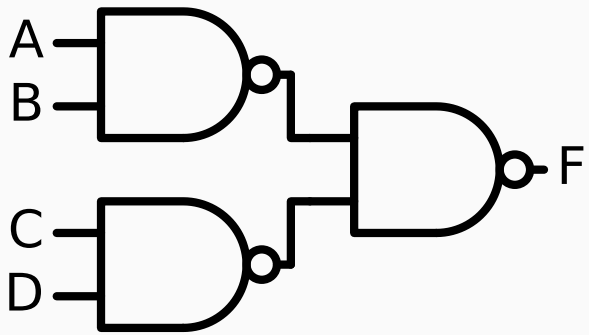


Figure 11 – *Somme de produits* NAND

Implémentation à deux niveaux . . . 3

- Cette configuration s'interprète plus facilement en représentant la porte de sortie comme une porte NOR avec les entrées complémentées (version équivalente de la porte NAND).
- En effet, la complémentation de chaque sortie de somme est compensée par la complémentation à l'entrée de la porte de sortie.

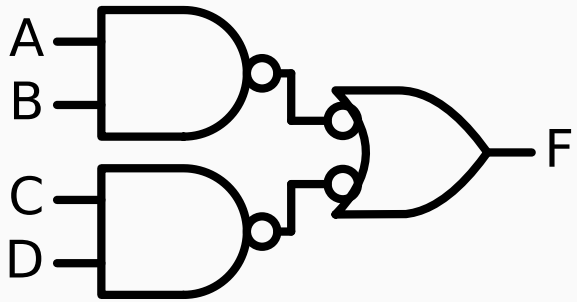


Figure 12 – *Somme de produits* NAND plus évidente