Chapitre 9

TD - Fonctions de base

Ce TD traite de la logique combinatoire et comprend les exercices suivants :

1. 9.1 : Simplification algébrique d'équations.
2. 9.2 : Simplification d'équations par tableau de Karnaugh.
3. 9.5 : Décodage.
4. 9.6 page 143 : Génération de fonctions.

9.1 Simplification algébrique

On considère qu'une équation booléenne est simplifiée si le nombre d'apparition des variables dans l'équation est le plus petit possible.

1. En utilisant les propriétés et théorèmes de l'algèbre de Boole, simplifiez l'expression : $S = (a + b + c) \cdot (\bar{a} + \bar{d} \cdot e + f) + \overline{(\bar{d} + e)} \cdot \bar{a} \cdot c + a \cdot b$

9.2 Simplification par tableau de Karnaugh

La méthode de Karnaugh permet de simplifier les fonctions logiques ayant peu de variables, à partir de la table de vérité de la fonction.

1. Simplifiez les deux fonctions F et G suivantes (cf. Tab. 9.1 page suivante et 9.2 page suivante) après avoir transformé leur table de vérité en tableau de Karnaugh.

9.3 Fonction F

- Les entrées sont a, b, c, d, e.
- Une variable d'entrée à « X » indique qu'elle peut être à 0 ou à 1.

9.4 Fonction G

- Les entrées sont a, b, c et d,
- i indique la valeur de la combinaison (ou minterme) en notation décimale,
- le « $-$ » indique que G peut prendre indifféremment la valeur 0 ou 1.

9.5 Décodage

Le décodeur est un circuit combinatoire à l'entrée duquel est appliqué un code binaire de n bits. Ce circuit possède N sorties (avec $N = 2^n$, en général). A chaque valeur du code d'entrée, il y a une seule sortie à l'état haut, toutes les autres sont à l'état bas. Les entrées d'un décodeur sont

e	d	c	b	a	F
0	X	X	X	X	0
1	0	0	X	X	0
1	0	1	0	X	1
1	0	1	1	X	0
1	1	0	0	0	0
1	1	0	0	1	1
1	1	0	1	X	1
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	X	0

TAB. 9.1: *Table de vérité de la fonction F.*

i	d	c	b	a	G
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	–
3	0	0	1	1	0
4	0	1	0	0	–
5	0	1	0	1	–
6	0	1	1	0	–
7	0	1	1	1	1
8	1	0	0	0	–
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

TAB. 9.2: *Table de vérité de la fonction G.*

souvent appelées *adresses*, car elles expriment en binaire le numéro décimal de la sortie activée. Les décodeurs peuvent être utilisés pour l'adressage de mémoires et la génération de fonctions logiques.

Décodeur BCD

Le BCD (« binary coded decimal ») est un code de 4 bits dont seules les 10 premières combinaisons de 0 à 9 sont employées. Les combinaisons restantes de 10 à 15 ne sont jamais utilisées. Un décodeur BCD est donc un décodeur qui a 4 entrées et 10 sorties.

1. Réalisez ce décodeur en considérant que si l'une des 6 combinaisons non autorisées est à l'entrée, toutes les sorties sont à l'état inactif « 0 ».

Décodeur de grande capacité

Si le nombre N est très élevé, on peut imaginer réaliser le décodage en cascadant des décodeurs de tailles moins importantes.

1. Par exemple essayez de concevoir un décodeur binaire 5 entrées / 32 sorties à partir de 2 décodeurs binaires 4 entrées / 16 sorties.
2. Quelle doit être la modification à apporter au décodeur 4 entrées / 16 sorties pour créer facilement le décodeur 5 entrées ?
3. Concevez un décodeur binaire 8 entrées / 256 sorties en utilisant le décodeur 4 entrées / 16 sorties précédemment modifié.

9.6 Génération de fonctions

Un transcodeur ou convertisseur est un circuit combinatoire à x entrées et y sorties. A chaque code d'entrée de x bits correspond un code de sortie y bits. Les décodeurs que nous avons étudiés dans l'exercice précédent sont donc des cas particuliers de transcodeurs.

On désire réaliser la fonction de transcodage d'un code BCD vers un code « 2 parmi 5 ». Dans le code « 2 parmi 5 », il y a toujours deux bits à « 1 » et 3 bits à « 0 ». La table de vérité est indiquée ci-dessous dans la Tab. 9.3.

- i indique la valeur de la combinaison (ou minterme) en décimal,
- les entrées sont a, b, c, d, e,
- les sorties sont $F_4F_3F_2F_1F_0$,
- si $i > 9$ l'état des sorties est indifférent.

i	d	c	b	a	F_4	F_3	F_2	F_1	F_0
0	0	0	0	0	1	1	0	0	0
1	0	0	0	1	0	0	0	1	1
2	0	0	1	0	0	0	1	0	1
3	0	0	1	1	0	0	1	1	0
4	0	1	0	0	0	1	0	0	1
5	0	1	0	1	0	1	0	1	0
6	0	1	1	0	0	1	1	0	0
7	0	1	1	1	1	0	1	0	0
8	1	0	0	0	1	0	0	0	1
9	1	0	0	1	1	0	0	1	0

Tab. 9.3: *Table de vérité de la fonction de conversion BCD → « 2 parmi 5 ».*

1. Réalisez la fonction à l'aide :
   (a) D'un décodeur BCD et quelques portes.
   (b) De multiplexeurs.

   Pour cela écrivez l'équation logique d'un multiplexeur 16 entrées (et donc 4 entrées de sélection) et comparez à l'expression d'une fonction logique quelconque à 4 entrées.

TD • Simplification algébrique :

dépôt TPT

$S = (a+b+c).(\bar{a}+\bar{d}.e+f) + \overline{(\bar{d}+e)}.\bar{a}.c + a.b$

$= a\bar{d}c + af + \bar{a}b + b\bar{d}e + bf + \bar{a}c + c\bar{d}e + cf + \bar{a}c\bar{d}\bar{e} + ab$

$= a(\bar{d}e + f) + b + ac + c\bar{d}e + f$

$= (a+c)(\bar{d}e + f) + b + \bar{a}c$

$= \bar{d}e + f + b + ac$

	$(a+c)x + \bar{a}c = ax + \bar{a}c$	
$c=0$	ax	ax
$c=1$	$x+\bar{a}$	$ax+\bar{a}$

wolframalpha.

• Simplification par tableau de Karnaugh.

e	d	c	b	a	F
0	x	x	x	x	0
↑	0	0	x	x	0
↑	0	↑	0	x	↑
↑	0	↑	↑	x	0
↑	↑	0	0	0	0
↑	↑	0	0	↑	↑
↑	↑	0	↑	x	↑
↑	↑	↑	0	0	↑
↑	↑	↑	0	↑	0
↑	↑	↑	↑	x	0

dc \ ba	00	01	11	10
00	0	0	0	0
01	1	1	0	0
11	1	0	0	0
10	0	1	1	1

$F = e.(\bar{d}c\bar{b} + \bar{b}\bar{a}c + d\bar{c}(a+b))$.

$F = e.(d+c)(\bar{c}+\bar{b}).(\bar{d}+\bar{c}+\bar{a}).(c+b+a)$.

python

i	d	c	b	a	G
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	—
3	0	0	1	1	0
4	0	1	0	0	—
5	0	1	0	1	—
6	0	1	1	0	—
7	0	1	1	1	1
8	1	0	0	0	—
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	1

dc \ ba	00	01	11	10
00	1	1	0	—
01	—	—	1	—
11	0	1	1	0
10	—	1	0	1

5 gates

vhdl

dc \ ba	00	01	11	10
00	1	1	0	1
01	0	1	1	0
11	0	1	1	0
10	1	1	0	1

$G = ac + \bar{1}(ab+c)$

$= ac + (\bar{a}+\bar{b})\bar{c}$

$= ac + \bar{a}\bar{c} + \bar{b}\bar{c}$

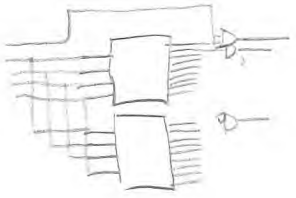
- **BCD.**

d	c	b	a	S_9	S_8	S_7	S_6	S_2	S_1	S_0
0	0	0	0	0	0	0		0	0	1
0	0	0	1	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	1	0	0
⋮										
1	0	0	1	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
⋮										
1	1	1	1							

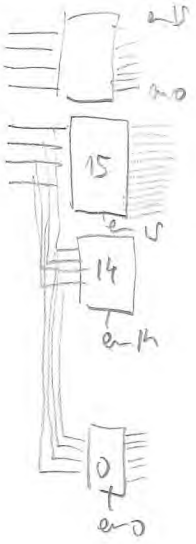
Nul si l'une des 6 entrées inutiles est présente

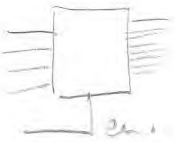
$S_0 = \bar{d}.\bar{c}.\bar{b}.\bar{a}$
$S_1 = \bar{d}.\bar{c}.\bar{b}.a$
$S_2 = \bar{d}.\bar{c}.b.\bar{a}$
⋮
$S_9 = d.\bar{a}.\bar{b}.a$

- $5 \to 32$: 2 $(4 \to 16)$.



- $8 \to 256$: $4 \to 16$.



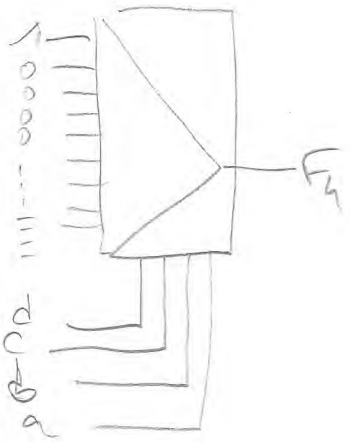


- BCD $\longrightarrow$ 2 out of 5

a) $F_4(d,c,b,a) = S_0 \vee S_7 \vee S_8 \vee S_5$

...

b)



(a or b or c) and (not(a) or not(d) and e or f) or not(not(d) or e) and not(a) and c or (a an

Input:

$$((a \vee b \vee c) \wedge (\neg a \vee (\neg d \wedge e) \vee f)) \vee (\neg (\neg d \vee e) \wedge \neg a \wedge c) \vee (a \wedge b)$$

((a OR b OR c) AND ((NOT a) OR ((NOT d) AND e) OR f))
OR ((NOT ((NOT d) OR e)) AND (NOT a) AND c) OR (a AND b)

$e_1 \vee e_2 \vee \ldots$ is the logical OR function
$\neg$ expr is the logical NOT function
$e_1 \wedge e_2 \wedge \ldots$ is the logical AND function

Minimal forms: Text notation

DNF	$(a \wedge \neg d \wedge e) \vee (a \wedge f) \vee (\neg a \wedge c) \vee b$
CNF	$(\neg a \vee b \vee \neg d \vee f) \wedge (\neg a \vee b \vee e \vee f) \wedge (a \vee b \vee c)$
ANF	$(a \wedge c) \veebar (a \wedge e) \veebar (a \wedge f) \veebar (b \wedge c) \veebar (a \wedge b \wedge c) \veebar (a \wedge b \wedge e) \veebar (a \wedge b \wedge f) \veebar (a \wedge d \wedge e) \veebar (a \wedge e \wedge f) \veebar (a \wedge b \wedge d \wedge e) \veebar (a \wedge b \wedge e \wedge f) \veebar (a \wedge d \wedge e \wedge f) \veebar (a \wedge b \wedge d \wedge e \wedge f) \veebar b \veebar c$
NOR	$(\neg a \barwedge b \barwedge \neg d \barwedge f) \barwedge (\neg a \barwedge b \barwedge e \barwedge f) \barwedge (a \barwedge b \barwedge c)$
NAND	$(a \uparrow \neg d \uparrow e) \uparrow (a \uparrow f) \uparrow (\neg a \uparrow c) \uparrow \neg b$
AND	$\neg (a \wedge \neg b \wedge d \wedge \neg f) \wedge \neg (a \wedge \neg b \wedge \neg e \wedge \neg f) \wedge \neg (\neg a \wedge \neg b \wedge \neg c)$
OR	$\neg (\neg a \vee d \vee \neg e) \vee \neg (\neg a \vee \neg f) \vee \neg (a \vee \neg c) \vee b$

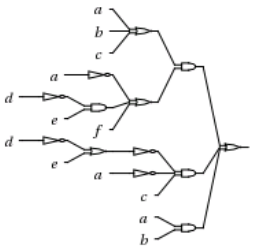
$e_1 \veebar e_2 \veebar \ldots$ is the logical XOR function
$e_1 \barwedge e_2 \barwedge \ldots$ is the logical NOR function
$e_1 \uparrow e_2 \uparrow \ldots$ is the logical NAND function

Other forms: Text notation

ESOP	$(\neg b \wedge c \wedge f) \veebar (a \wedge \neg b \wedge \neg c \wedge f) \veebar (\neg a \wedge \neg b \wedge c \wedge \neg e \wedge \neg f) \veebar (\neg b \wedge c \wedge \neg d \wedge e \wedge \neg f) \veebar (\neg a \wedge \neg b \wedge c \wedge d \wedge e \wedge \neg f) \veebar (a \wedge \neg b \wedge \neg c \wedge \neg d \wedge e \wedge \neg f) \veebar b$
IMPLIES	$(a \Rightarrow \neg ((d \Rightarrow \neg (\neg f \Rightarrow b)) \Rightarrow \neg (\neg d \Rightarrow \neg (\neg e \Rightarrow (\neg f \Rightarrow b))))) \Rightarrow \neg (\neg a \Rightarrow \neg (\neg c \Rightarrow b))$
ITE	$(a \wedge ((d \wedge (f \vee (\neg f \wedge b))) \vee (\neg d \wedge (e \vee (\neg e \wedge (f \vee (\neg f \wedge b))))))) \vee (\neg a \wedge (c \vee (\neg c \wedge b)))$

$p \Rightarrow q$ represents the logical implication $p \Rightarrow q$

Logic circuit:

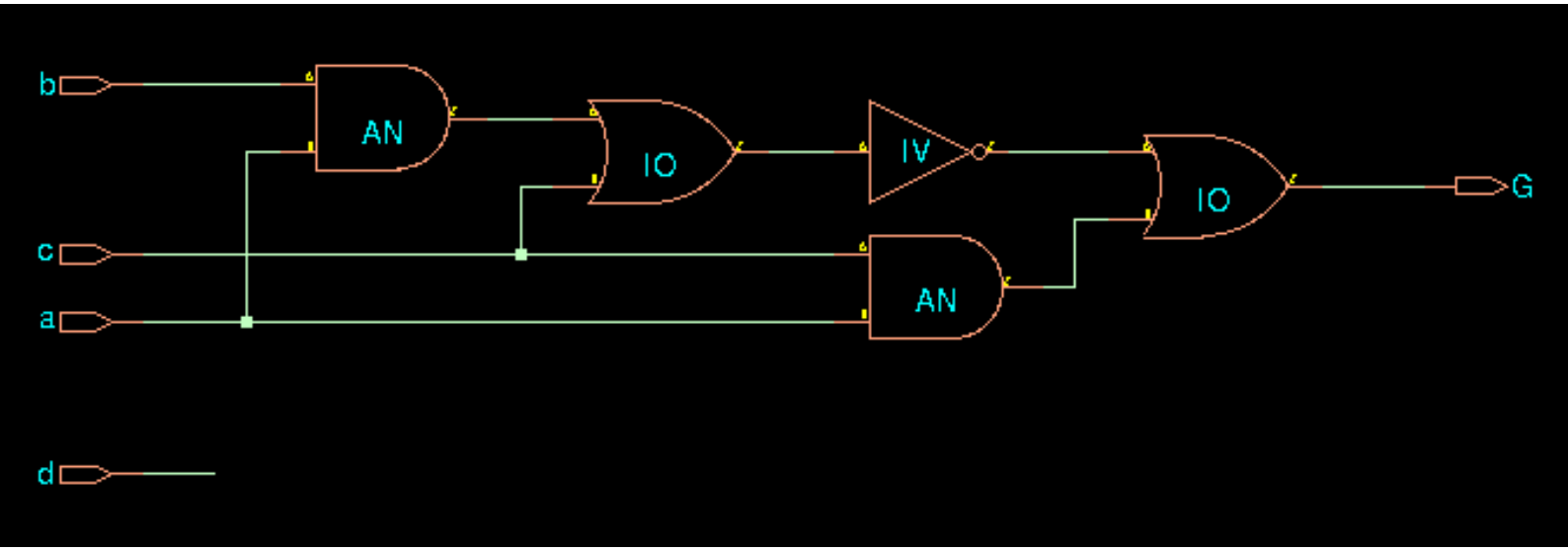


Truth density:

$$\frac{25}{32} = 78.125\%$$

Boolean operator number:

18 446 654 665 375 350 528 with variable ordering $\{a, b, c, d, e, f\}$


b
c
a
d
AN
IO
IV
AN
IO
G