計算機システムの基礎(第7回配布)

第3章　論理回路

（1）集合　　集合、補集合、集合の要素、集合の基本演算

（2）2値論理　と　基本論理回路　　ブール代数、公理、定理、基本論理回路

（3）組み合わせ回路

　　回路図解析と回路図設計　（論理式から論理回路へ）

　　真理値表　と　カルノー図、　主加法標準形、

　　半加算器と全加算器、

　　簡単化、エンコーダとデコーダ

（4）順序回路

担当：　福井大学　大学院工学研究科
情報・メディア工学専攻
森　眞一郎　(moris@u-fukui.ac.jp)

[レポート課題] (論理式と論理回路)

(1) 以下の論理式を証明せよ

(a) $A \bullet C + \overline{A} \bullet B = (A+B) \bullet (\overline{A}+C)$

(b) $\overline{A \bullet \overline{B \bullet \overline{C}}} + B \bullet C = \overline{A} + B$

(2) 上記(a)の左辺の論理式に対応する論理回路、ならびに
右辺の論理式に対応する論理回路を設計して図示せよ。

(3) 教科書P.117　問題7の (1)~(3)の論理式に対応する論理回路を設計せよ。

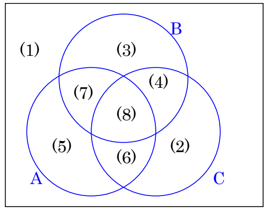
(A4　レポート用紙提出のこと。表紙をつける必要はないが、1枚目の上側余白に学生番号、氏名を記入のこと
両面を使って解答してよいが、複数ページに跨る場合は必ず　ホッチキス留め　すること。)

最小項と最大項（1）

[用語の定義]最小項と最大項
- 最小項：
  - 全ての変数が真または偽の形で含まれている論理積項
- 最大項：
  - 全ての変数が真または偽の形で含まれている論理和項

例　3変数の場合

A	B	C	Y
0	0	0	(1)
0	0	1	(2)
0	1	0	(3)
0	1	1	(4)
1	0	0	(5)
1	0	1	(6)
1	1	0	(7)
1	1	1	(8)



ベン図

最小項はベン図上での8つの区画のうち
いずれか一つの区画に対応

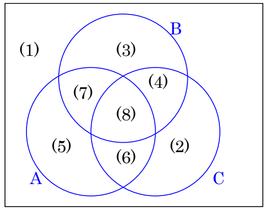
最大項はベン図上での8つの区画のうち
いずれか一つの区画を除いた区画に対応

最小項と最大項（2）

[用語の定義]最小項と最大項
- 最小項：
  - 全ての変数が真または偽の形で含まれている論理積項
- 最大項：
  - 全ての変数が真または偽の形で含まれている論理和項

例　3変数の場合

A	B	C	Y
0	0	0	(1)
0	0	1	(2)
0	1	0	(3)
0	1	1	(4)
1	0	0	(5)
1	0	1	(6)
1	1	0	(7)
1	1	1	(8)



ベン図

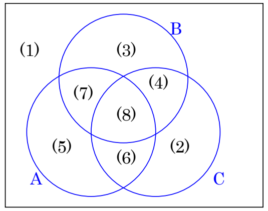
最小項はベン図上での8つの区画のうち
いずれか一つの区画に対応

A・(～B)・C

最大項はベン図上での8つの区画のうち
いずれか一つの区画を除いた区画に対応

(～A)+B+(～C)
....抜いた区画はA・(～B)・C

最小項と最大項(3)



ベン図

最小項はベン図上での8つの区画のうち
いずれか一つの区画に対応

$A \bullet \overline{B} \bullet C$

最大項はベン図上での8つの区画のうち
いずれか一つの区画を除いた区画に対応

$\overline{A} + B + \overline{C}$

....除いた区画は $A \bullet \overline{B} \bullet C$

対応関係は「ド・モルガンの定理」を使えば一意にわかる!!

$$\overline{X \bullet Y \bullet Z} = \overline{X} + \overline{Y} + \overline{Z}$$

$$\overline{X + Y + Z} = \overline{X} \bullet \overline{Y} \bullet \overline{Z}$$

標準形(1)

[用語の定義]積和形と和積形

- 積和形(sum of products form):
  - 論理関数が論理積項の和として論理式表現された場合、その論理式は積和形表現であると呼ぶ。
  - 特に、全ての論理積項が最小項で表現されている場合は、
    標準積和形 (あるいは加法標準形)と呼ぶ。
- 和積形(product of sums form):
  - 論理関数が論理和項の積として論理式表現された場合、その論理式は和積形表現であると呼ぶ。
  - 特に、全ての論理和項が最大項で表現されている場合は、
    標準和積形 (あるいは乗法標準形)と呼ぶ。

標準形(2)

[用語の定義]積和形と和積形

- 積和形(sum of products form):
  - 積和形表現
    f(A、B、C) = A・B + B・C + A・C ... 積 の 和
  - 標準積和形表現(あるいは加法標準形表現)
    f(A、B、C) = A・B・C+A・B・(~C)+A・(~B)・C+(~A)・B・C
    ... 最小項 の 和
- 和積形(product of sums form):
  - 和積形表現
    g(A,B,C) = (~A + ~B)(~B + ~C)(~A + ~C) ... 和 の 積
  - 標準和積形 (あるいは乗法標準形)
    g(A,B,C) = (~A + ~B + ~C)(~A + ~B + C)(~A + B + ~C)(A + ~B + ~C)
    ... 最大項の積

組み合わせ論理回路の合成

積和型あるいは和積型論理式の場合(2段)

積和型の場合 AND-OR型ゲート構成 で合成	和積型の場合 OR-AND型ゲート構成 で合成

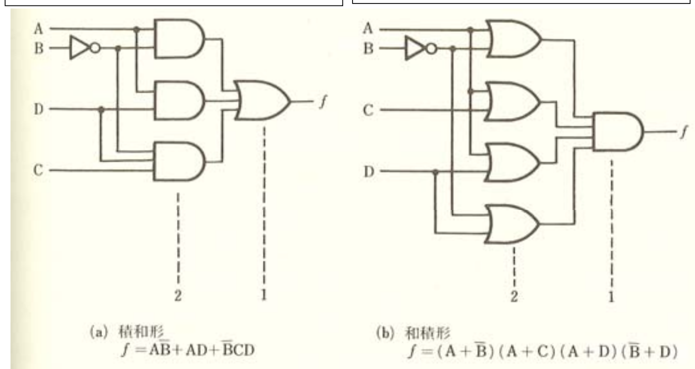
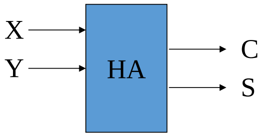


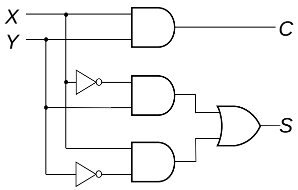
図 4.16 2段構成のゲート回路の例

半加算器(HA:Half Adder)

- 入力X、Yが与えられたときに、その桁の和S (Sum)と桁上がりC (Carry)を出力する回路



X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



$C = X \bullet Y$

$S = \overline{X} \bullet Y + X \bullet \overline{Y} = X \oplus Y$

[復習2]2進数の加算(その2)

- n桁の場合

000 + 000 = 0000
000 + 001 = 0001
…
001 + 011 = 0100
…
101 + 011 = 1000
…
111 + 111 = 1110

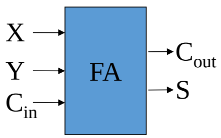
i桁目の計算では、入力Xi, Yiと i-1桁目からの桁上げC_{i-1}から、その桁の和Siと桁上げCiを生成する

101+011= (0100+0010)+(1+1) — 1桁目
= (0100+0010)+ 10
= (0100+0000)+(00+10+10) — 2桁目
= (0100+0000)+(100)
…

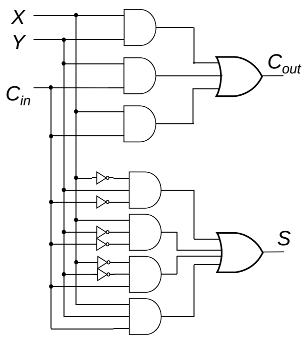
2項の和では C_{i-1} は 0 or 1

全加算器(FA: Full Adder)

- 入力X, Yと下位桁からの桁上げCinから、その桁の和Sと桁上げCoutを生成する回路



X	Y	C_{in}	C_{out}	S
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1



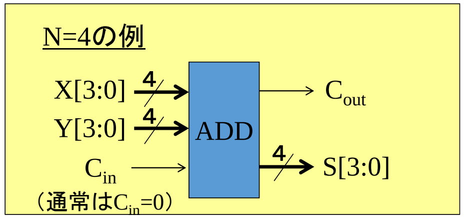
$C_{out} = X \bullet Y + Y \bullet C_{in} + X \bullet C_{in}$

$S = \overline{X} \bullet Y \bullet \overline{C_{in}} + X \bullet \overline{Y} \bullet \overline{C_{in}} + \overline{X} \bullet \overline{Y} \bullet C_{in} + X \bullet Y \bullet C_{in}$

C_{out}は多数決、
Sは3変数の排他的論理和

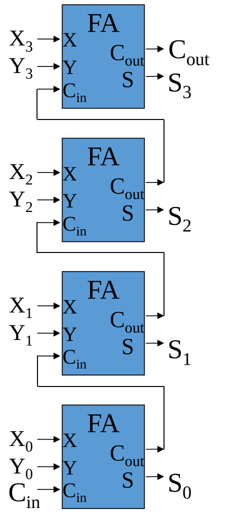
Nビット加算器(ADD:Adder)

教科書p.205



- リプルキャリ方式(右図)
  - FAをN個直列に接続して構成
    - i桁目の桁上げ出力C_{out}をi+1桁目の桁上げ入力C_{in}に接続
  - 回路は簡単であるが、桁上げ信号(Carry)が逐次的に伝播するため遅い

高速化手法の例
　桁上げ選択方式、桁上げ先見方式



復号器(Decoder)/符号器(Encoder)

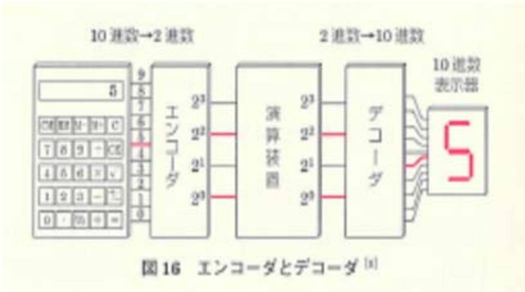


図16 エンコーダとデコーダ

2進化10進
符号器
(BCDエンコーダ)

入力	出力
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

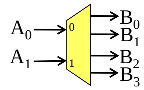
出力サイズが変化する
符号器の例 (モールス符号の親戚)

入力	出力
A	101100
B	111010100
C	111011100
D	1110100
E	100
F	101011100
G	1111100
…	
T	1100
…	
Z	111110100

デコーダ(decoder)/デマルチプレクサ(demultiplexer)

2^N個の出力のうち、Nビットの入力Aに対応する出力のみ1をそれ以外の出力には0を出す回路

「例1」2-to-4デコーダ(2入力デマルチプレクサ)



回路図は自分で考えて見ましょう

A_1	A_0	B_3	B_2	B_1	B_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

教科書p.102

エンコーダ(符号化器、encoder)

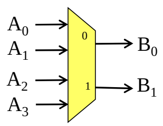
Nビットの入力を、Mビットの出力に符号化する回路

・一般のエンコーダはN＝2^Mとしてデコーダの逆の動作を示す。
(この場合、入力禁止の組み合わせが生じる)
・より広義には、符号化ルールを定めると当該ルールに対応した任意のエンコーダを作ることができる。

「例1」4-to-2エンコーダ

A_3	A_2	A_1	A_0	B_1	B_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

入力制限1:入力が1となるのは1ビットのみ
入力制限2:必ずいづれかの入力ビットが1



$$B_0 = A_3 + A_1$$
$$B_1 = A_3 + A_2$$

組み合わせ論理回路の簡単化

1. 準備 (加法標準形 と 真理値表)
2. 簡単化の目的
3. 簡単化とは?
4. 簡単化の手法
5. 公式を使った簡単化
6. カルノー図を使った簡単化
   1. カルノー図とは?
   2. カルノー図の作り方
   3. カルノー図を使った簡単化

標準形

[用語の定義]積和形と和積形

- 積和形(sum of products form):
  - 論理関数が**論理積項の和**として論理式表現された場合、その論理式は積和形表現であると呼ぶ。
  - 特に、全ての論理積項が**最小項**で表現されている場合は、
    標準積和形 (あるいは加法標準形)と呼ぶ。

- 和積形(product of sums form):
  - 論理関数が論理和項の積として論理式表現された場合、その論理式は和積形表現であると呼ぶ。
  - 特に、全ての論理和項が最大項で表現されている場合は、
    標準和積形 (あるいは乗法標準形)と呼ぶ。

標準形

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

真理値表 と 加法標準形 の対応関係

- 加法標準形
  - 真理値表で関数値が「1」となる入力変数の組み合わせに対して、**変数が「1」のところは変数自身、変数が「0」のところは変数の偽(否定)**を対応させて変数全部の論理積をつくり、出来上がった最小項の論理和を求める。

$$f(A,B,C) = A \bullet B \bullet C + A \bullet B \bullet \overline{C} + A \bullet \overline{B} \bullet C + \overline{A} \bullet B \bullet C$$

簡単化の目的

まずは直感的に....

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

左の真理値表であらわされる論理関数Yを
加法標準形で書くと

$$Y = \overline{A} \bullet \overline{B} \bullet \overline{C} \bullet D + \overline{A} \bullet \overline{B} \bullet C \bullet D + \overline{A} \bullet B \bullet \overline{C} \bullet D + \overline{A} \bullet B \bullet C \bullet D$$

となるが、……よく見ると

$$Y = \overline{A} \bullet D$$

とも等価である。

簡単化の目的

- 見てわかりやすい、扱いやすい
- 論理回路が 速く/小さく/安全に できる！！
  - より工学的には、
    1. 回路の遅延時間を短くすること
    2. 回路の総量(面積)を小さくすること
    3. 回路の故障を少なくすること、

       ……

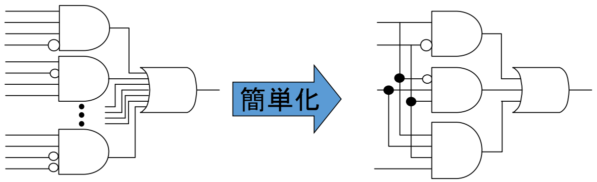
注意！！

簡単化 ⊂ 最適化

Eg.) 消費電力最小化
遅延のばらつき最小化

簡単化とは？

- 加法標準形論理式の簡単化とは？
  1. 積項（AND素子）の数を減らすこと
     ＝ 論理和（OR素子）の入力数を減らすこと
  2. 積項（AND素子）の入力数を減らすこと



加法標準形 ⇒ 最小項の論理和
真理値表で1になる1行を1つの積項（AND）で表現し、それらの積項を論理和（OR）したもの。

簡単化の原理

[公式]（A＋～A）＝1
[公式]（A＋A）＝A
[公式]（A＋1）＝1、A･1＝A
を使って式を短くする

これだけです！！

例1

$$f(A,B,C) = A \bullet \overline{B} \bullet C + \overline{A} \bullet \overline{B} \bullet C$$

$$= (A + \overline{A}) \bullet \overline{B} \bullet C = 1 \bullet \overline{B} \bullet C = \overline{B} \bullet C$$

例2

$$f(A,B,C) = A \bullet \overline{B} \bullet C + \overline{B} \bullet C$$

この続きは自分でやってみましょう！

重要な公理、定理

すべての変数は
1または0の何れかの値をとる

$x+1=1$	$x+0=x$
$x \cdot 1=x$	$x \cdot 0=0$
$\overline{0}=1$	$\overline{1}=0$

- 対合律(2重否定)
  $\overline{\overline{x}} = x$
- 補元律
  $x \cdot \overline{x} = 0$
  $x + \overline{x} = 1$
- べき等律
  $x \cdot x = x$
  $x + x = x$
- 交換律
  $x \cdot y = y \cdot x$
  $x + y = y + x$
- 結合律
  $(x \cdot y) \cdot z = x \cdot (y \cdot z)$
  $(x + y) + z = x + (y + z)$
- 分配律
  $x + y \cdot z = (x + y) \cdot (x + z)$
  $x \cdot (y + z) = x \cdot y + x \cdot z$
- 吸収律
  $x \cdot (x + y) = x$
  $x + x \cdot y = x$
- ド・モルガンの定理
  $\overline{x \cdot y} = \overline{x} + \overline{y}$
  $\overline{x + y} = \overline{x} \cdot \overline{y}$

公式を使った簡単化

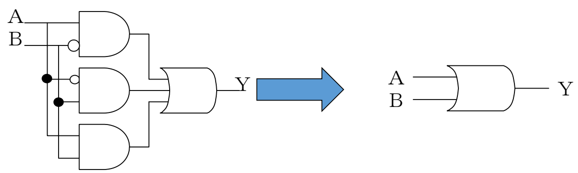
- 練習1

$$Y = A \bullet \overline{B} + \overline{A} \bullet B + A \bullet B$$

$=$

$=$

$= A + B$



公式を使った簡単化

• 練習2

$Y=\overline{A}\bullet B\bullet C+A\bullet\overline{B}\bullet C+A\bullet B\bullet\overline{C}+A\bullet B\bullet C$

$=$

$=$

$=B\bullet C+A\bullet C+A\bullet B$　　（多数決回路）

簡単化前　3入力AND素子 x4 ＋ 4入力OR素子
簡単化後　2入力AND素子 x3 ＋ 3入力OR素子

公式を使った簡単化

• 練習3

$Y=A\bullet\overline{B}+B\bullet\overline{C}+\overline{B}\bullet C+\overline{A}\bullet B$

$=$.

$=$.

$=$.

$=A\bullet\overline{B}+B\bullet\overline{C}+\overline{A}\bullet C$

（別解）$=A\bullet\overline{C}+\overline{B}\bullet C+\overline{A}\bullet B$

簡単化前　2入力AND素子 x 4 ＋ 4入力OR素子
簡単化後　2入力AND素子 x 3 ＋ 3入力OR素子

公式を使った簡単化

[結論]　大変だ！

これではあまりにも生産性が悪い！！
何とかならないか？