コンピュータ基礎

ブール代数と論理回路

成蹊大学　理工学部
情報科学科

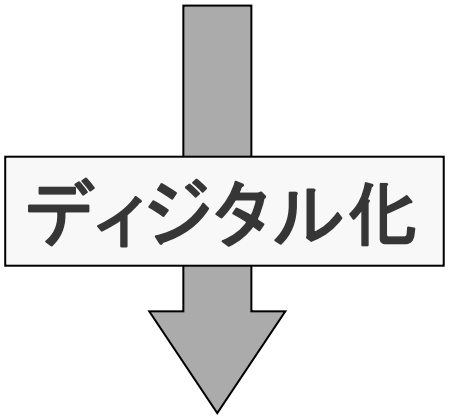
ディジタルとアナログ

- アナログ（analog）とは

  アナログ＝連続的

  本質的には無限の情報量がある

  例：音、光、温度、時間、etc.



アナログ情報の中から代表点を選ぶ＝サンプリング

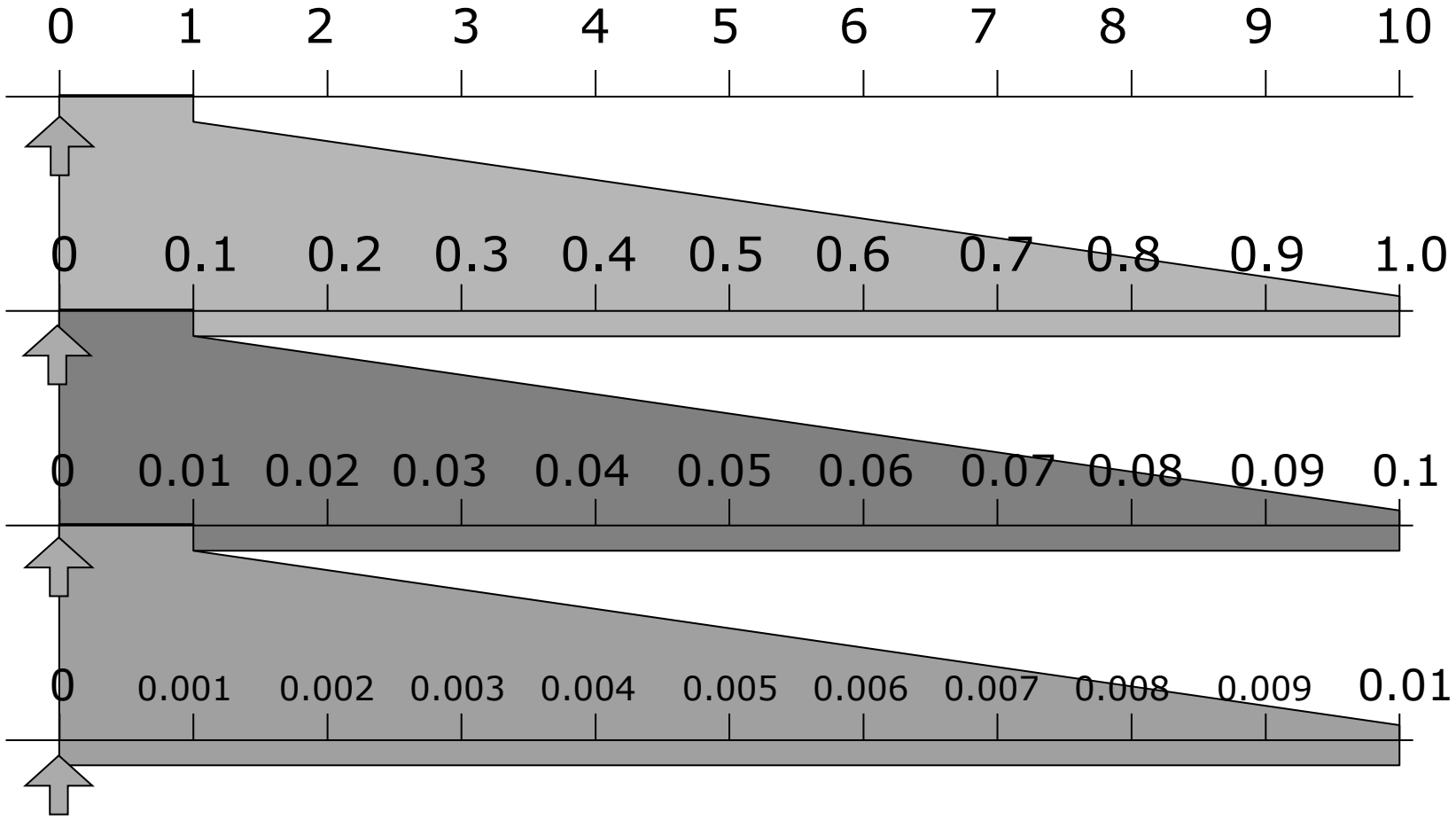
ディジタル＝離散的

有限の情報量に抑えられる

ディジタル化　　一般的には2値化

連続量とディジタル化

- 矢印の位置を読み取ってみよう。



「メモリの数値を読み取る」＝「離散化して情報を取得する」

情報の２値化

- ２値：　０ と １

　ディジタルシステム　　電気、磁気、光の利用

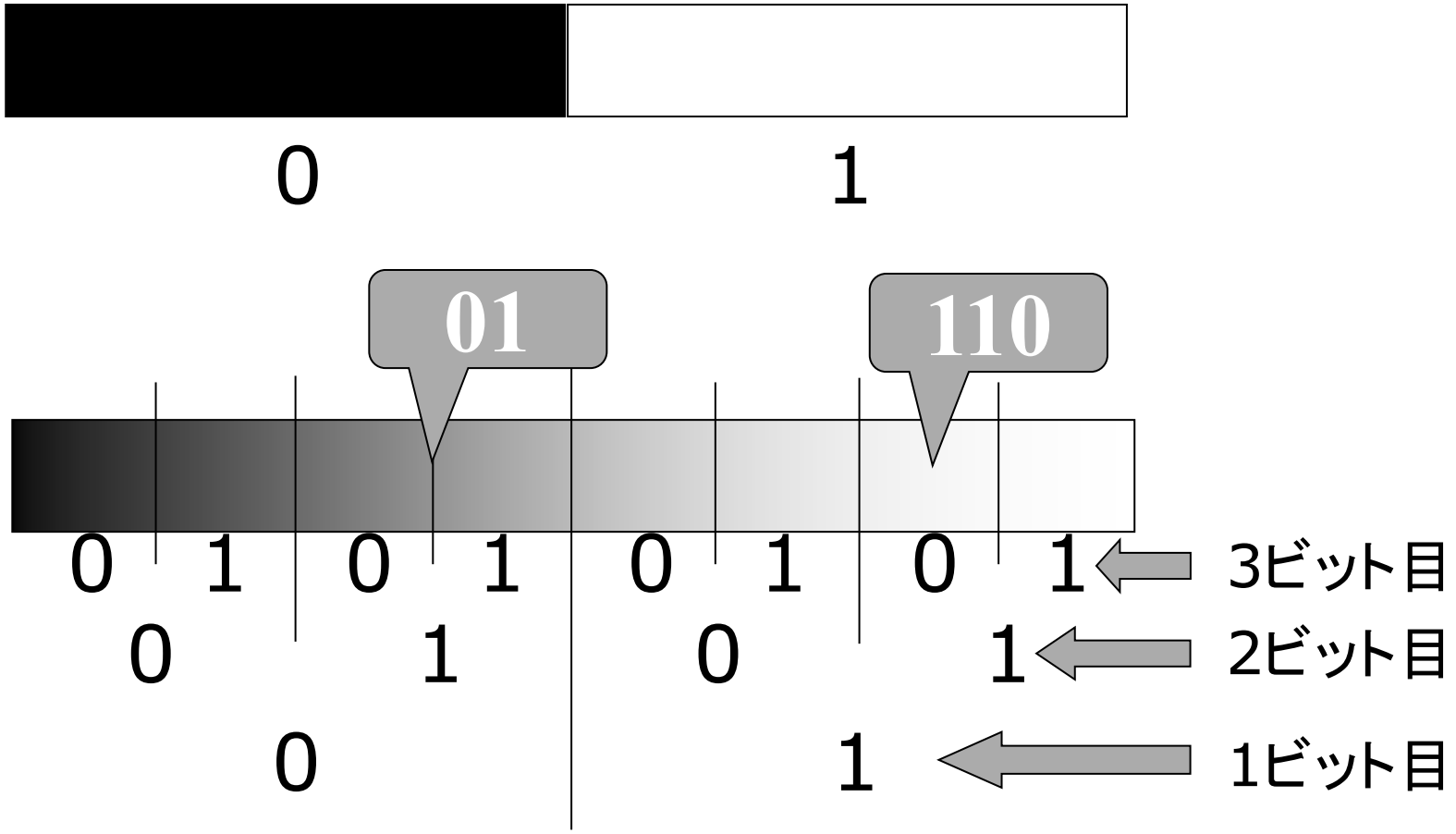
	電気	磁気	光
0	電圧High	N	消灯・遮断
1	電圧Low	S	点灯・透過

２値は、白黒はっきり区別できる表現

　「白っぽい」「黒っぽい」のような程度を精度良く表現するには0と1を複数桁並べて表現可能

情報の2値化

- 「白っぽい」「黒っぽい」のような、程度の表現



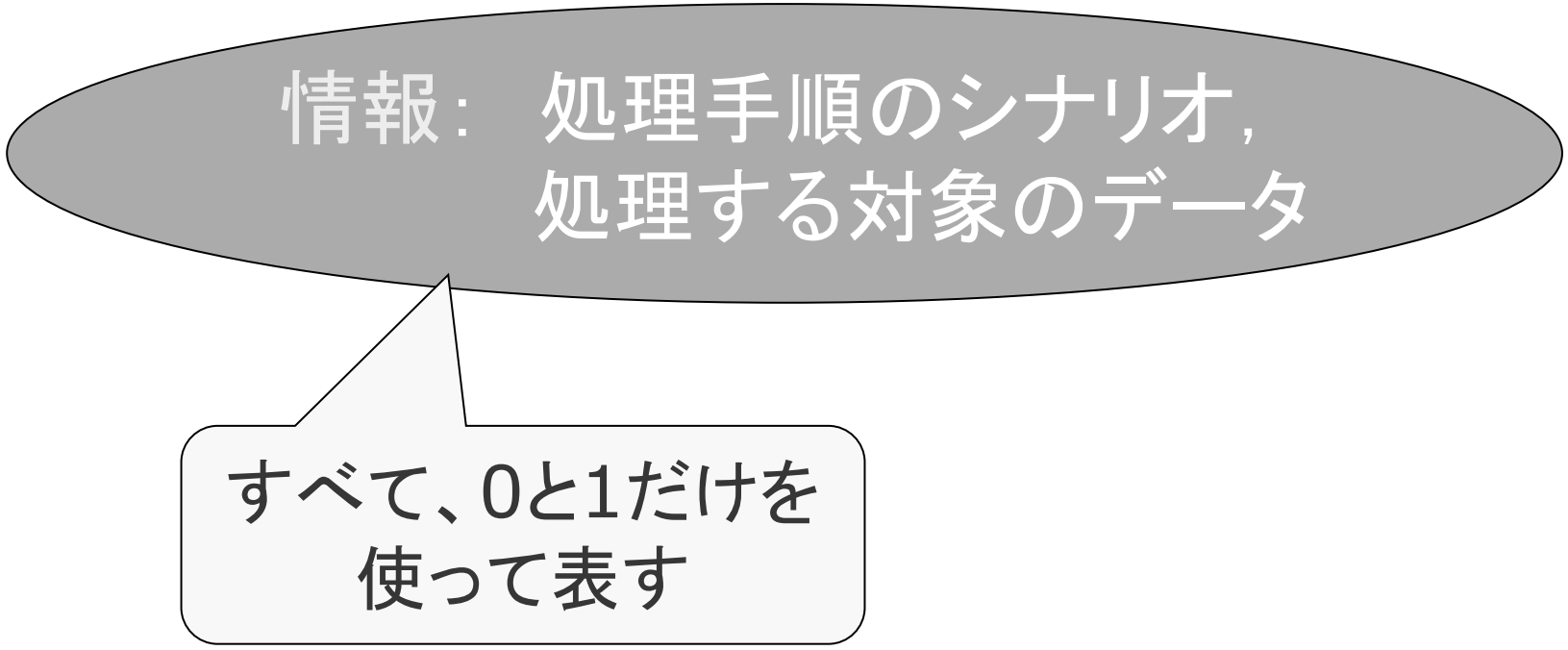
ディジタル化のメリット

- 情報量を有限にすることができる
- 情報を劣化することなく伝えやすい
- 伝えられた情報を再現しやすい

“黒っぽい”と人に伝えても、人によって程度が異なるかも知れない。
しかし、“001の黒さ”と伝えれば、「真っ黒」と「真っ白」の間を8つのレベルに分けて、真っ黒から2番目のレベルを選ぶことで、再現性が良くなる。

コンピュータの情報表現

- コンピュータは大規模ディジタルシステムの代表格



コンピュータ基礎　前半のポイント

- ディジタル化＝情報の2値化
- 情報表現
  - 整数の内部表現
  - 実数の内部表現
  - 文字の内部表現（ASCIIコード）
- 同じ情報をできる限り少なくするために
  - エントロピー、ハフマン符号（符号割当）
- 情報伝達において受信側で誤り訂正可能に
  - ハミング符号

情報の表現(数値データ:整数)

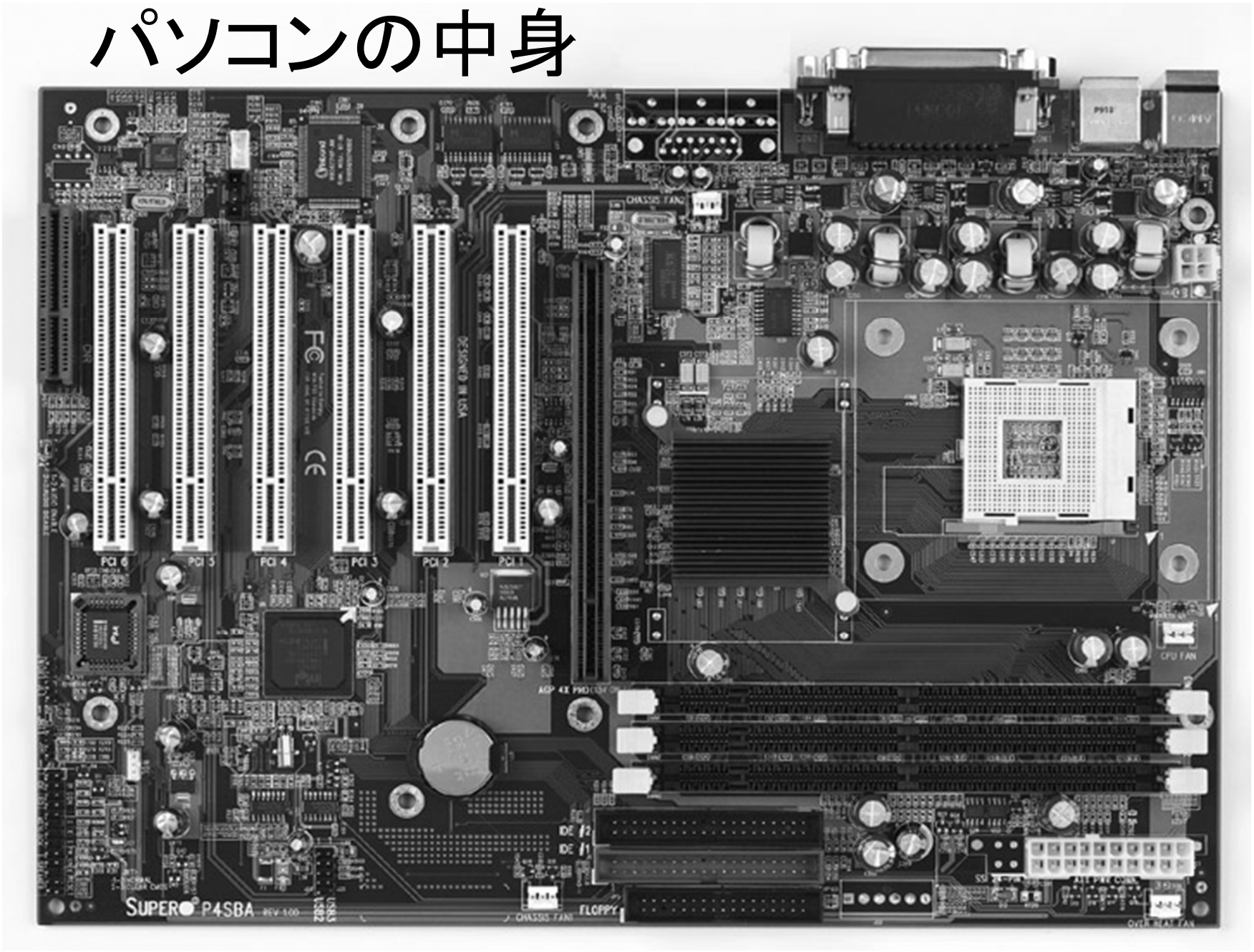
各桁で使用できる数字

10進数(decimal):0～9　　2進数(binary):0,1

8進数(octal):0～7　　16進数(hexadecimal):0～9,A～F

Dec	Bin	Oct	Hex	Dec	Bin	Oct	Hex
0	0	0	0	16	10000	20	10
1	1	1	1	17	10001	21	11
2	10	2	2	18	10010	22	12
3	11	3	3	19	10011	23	13
4	100	4	4	20	10100	24	14
5	101	5	5	21	10101	25	15
6	110	6	6	22	10110	26	16
7	111	7	7	23	10111	27	17
8	1000	10	8	24	11000	30	18
9	1001	11	9	25	11001	31	19
10	1010	12	A	26	11010	32	1A
11	1011	13	B	27	11011	33	1B
12	1100	14	C	28	11100	34	1C
13	1101	15	D	29	11101	35	1D
14	1110	16	E	30	11110	36	1E
15	1111	17	F	31	11111	37	1F

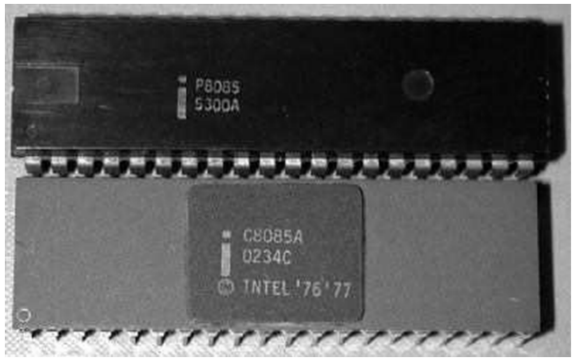
パソコンの中身



（Supermicro社製 SUPER P4SBA）

CPU
中央演算処理装置

i8085



Pentium4(表）

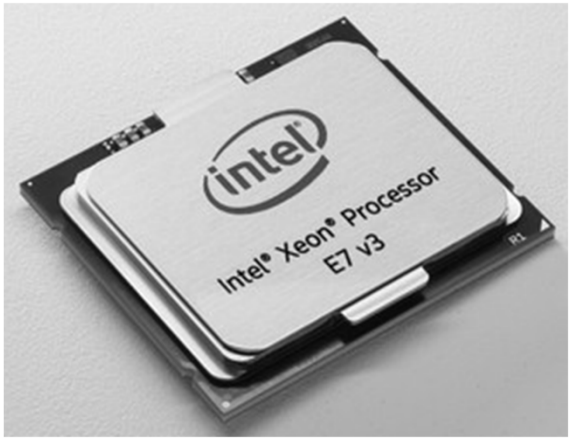
Xeon7500(表&裏）

1CPUに8コア

Core iシリーズ

1CPUに2～6コア

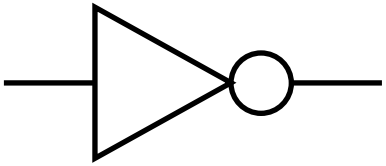
Xeon E7 v3



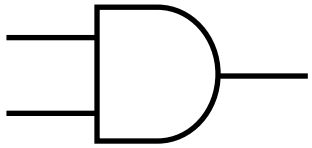
1CPUに18コア

情報をあやつる材料

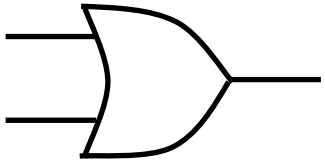
基本ゲート：論理回路の構成要素



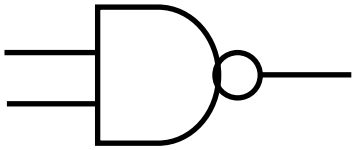
NOT
否定
1入力



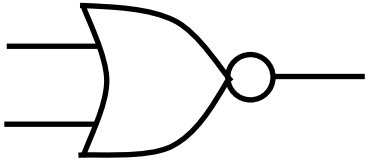
AND
論理積
n入力



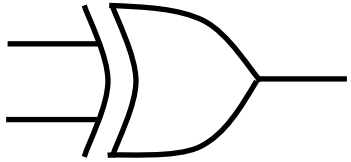
OR
論理和
n入力



NAND
n入力



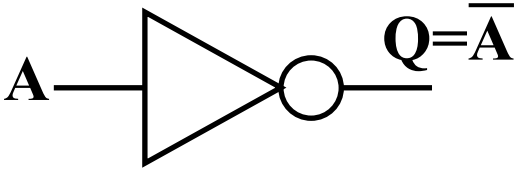
NOR
n入力



XOR
排他的論理和
n入力

どっちもゲート

どっちかゲート

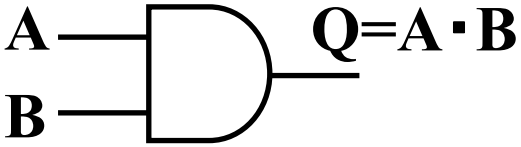


NOTゲート

否定

A	Q
0	1
1	0

入力信号を否定（反転）する

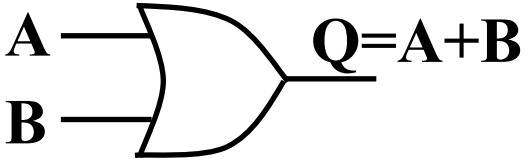


ANDゲート

論理積

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

入力がすべて1のときのみ出力が1

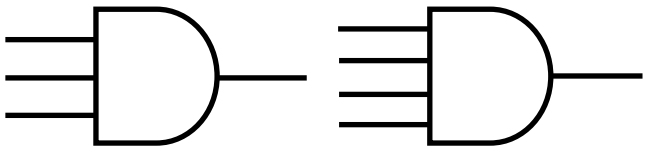


ORゲート

論理和

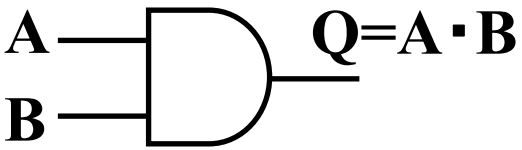
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

入力の少なくとも1つが1のとき出力が1



3入力AND

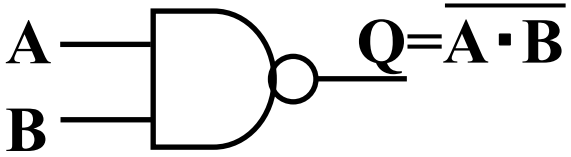
4入力AND



ANDゲート

論理積

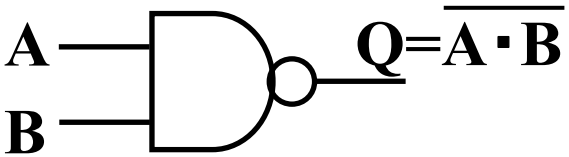
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

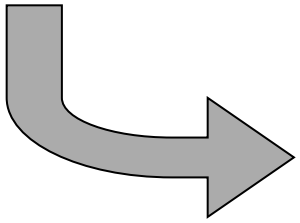


NANDゲート

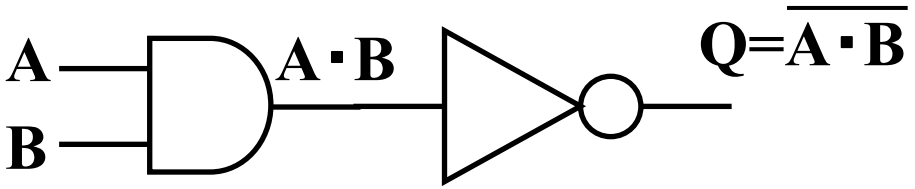
（NOT AND）

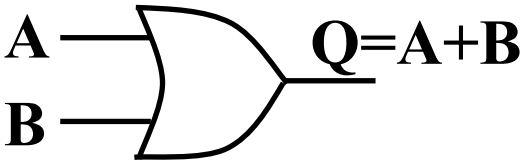
A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0





ANDゲートの出口にNOTゲートを
付加したもの同じ



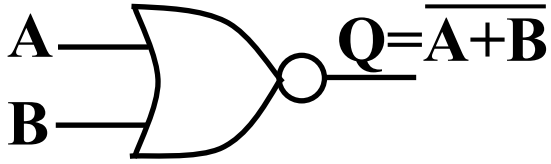


ORゲート

論理和

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

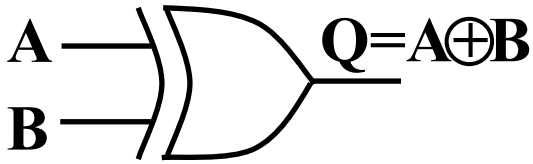
入力の少なくとも1つが1のとき出力が1



NORゲート

（NOT OR）

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0



XORゲート

排他的論理和

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

2入力が互いに異なるとき出力が1

$$A \oplus B = \overline{A} \cdot B + A \cdot \overline{B}$$

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}B$	$A\overline{B}$	$\overline{A}B+A\overline{B}$
0	0	1	1	0	0	0
0	1	1	0	1	0	1
1	0	0	1	0	1	1
1	1	0	0	0	0	0

0と1だけを扱う特別な数学
ブール代数（Boolean Algebra）［論理数学］

- ブール代数の公理

公理１　論理変数は**0**か**1**のどちらかの値をとる

公理２　$\overline{0}=1,\ \overline{1}=0$

公理３　$A+0=0+A=A,\quad A\cdot 1=1\cdot A=A$

公理４　$A+1=1+A=1,\quad A\cdot 0=0\cdot A=0$

■ ブール代数の定理

定理1　$A+\overline{A}=1$，$A\cdot\overline{A}=0$

《証明》

A	$\overline{A}$	$A+\overline{A}$
0	1	1
1	0	1

A	$\overline{A}$	$A\cdot\overline{A}$
0	1	0
1	0	0

定理2　$A+A=A$，$A\cdot A=A$

《証明》

A	A	A+A	A・A
0	0	0	0
1	1	1	1

■ ブール代数の定理（その2）

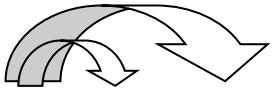
定理3　A+B=B+A，A･B=B･A　（交換則）

定理4　（A+B）+C=A+（B+C），
（A･B）･C=A･（B･C）　（結合則）

《証明》

A	B	C	A+B	(A+B)+C	B+C	A+(B+C)	A･B	(A･B)･C	B･C	A･(B･C)
0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	1	0	0	0	0
0	1	0	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	0	0	1	0
1	0	0	1	1	0	1	0	0	0	0
1	0	1	1	1	1	1	0	0	0	0
1	1	0	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	1	1

■ ブール代数の定理（その3）



定理5　A+B･C=(A+B)･(A+C),

A･(B+C)=A･B+A･C　　（分配則）

《証明》

A	B	C	B+C	A･(B+C)	A･B	A･C	A･B+A･C
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

定理5第2式　　定理5第2式

(A+B)(A+C)=(A+B)A+(A+B)C=AA+AB+AC+BC

定理2　　公理3＆定理5第2式　公理4　　公理3

=A+AB+AC+BC=A(1+B+C)+BC=A･1+BC=A+BC

- ブール代数の定理（その４）

定理６　A+A･B=A,　A･(A+B)=A　（吸収則）

《証明》

$A+A\cdot B = A\cdot 1+A\cdot B$ （公理３）
$= A\cdot(1+B)$ （分配則（定理５））
$= A\cdot 1$ （公理４）
$= A$ （公理３）

$A\cdot(A+B) = A\cdot A+A\cdot B$ （分配則（定理５））
$= A+A\cdot B$ （定理２）
$= A$ （定理６第１式）

定理７　$A+\overline{A}\cdot B=A+B$,
$A\cdot(\overline{A}+B)=A\cdot B$　（共有項則）

《証明》

$A+\overline{A}\cdot B = (A+\overline{A})\cdot(A+B)$ （分配則）
$= 1\cdot(A+B)$ （定理１）
$= A+B$ （公理３）

■ ブール代数の定理（その5）

定理8　$A\cdot B+B\cdot C+\overline{A}\cdot C=A\cdot B+\overline{A}\cdot C$,

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

（共有項則）

《証明》

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

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

吸収則

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

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

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

吸収則

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

■ ブール代数の定理（その6）

定理9　$\overline{A+B}=\overline{A}\cdot\overline{B}$,　　$\overline{A\cdot B}=\overline{A}+\overline{B}$

（ド・モルガンの定理）

《証明》

A	B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$	$\overline{A\cdot B}$	$\overline{A}+\overline{B}$
0	0						
0	1						
1	0						
1	1						

ド・モルガンの定理の意味(1)

数直線の範囲の例

赤の範囲は

$0<x$ && $x \leqq 5$　　「&&」は「かつ(=AND)」の意味

$A \cdot B$　　「$0<x$」を A、「$x \leqq 5$」を B と表すと・・・

赤ではない範囲は

$x \leqq 0$ || $5<x$　　「||」は「または(=OR)」の意味

$\overline{A} + \overline{B}$　　「$x \leqq 0$」は「$0<x$」の否定、
「$5<x$」は「$x \leqq 5$」の否定だから ・・・

つまり「赤の範囲」を否定すると「赤ではない範囲」になるので

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$

⇐ ド・モルガンの定理第2式

ド・モルガンの定理の意味(2)

双対性

ブール代数の公理・定理に、それぞれ2つずつ式があったことに注目
一方の式の両辺を否定すると、ド・モルガンの定理により、
もうひとつの式が得られる。

例: 定理7 2変数の共有項則

第1式 $A + \overline{A} \cdot B = A + B$

両辺を否定して、それぞれド・モルガンの定理を用いると・・・

$$\overline{A + \overline{A} \cdot B} = \overline{A + B}$$

$$\overline{A} \cdot \left(A + \overline{B}\right) = \overline{A} \cdot \overline{B}$$

$$X \cdot \left(\overline{X} + Y\right) = X \cdot Y$$

$\overline{A}$をXに、$\overline{B}$をYに置き換えると

⇐ 定理7の第2式

論理式と論理回路表現

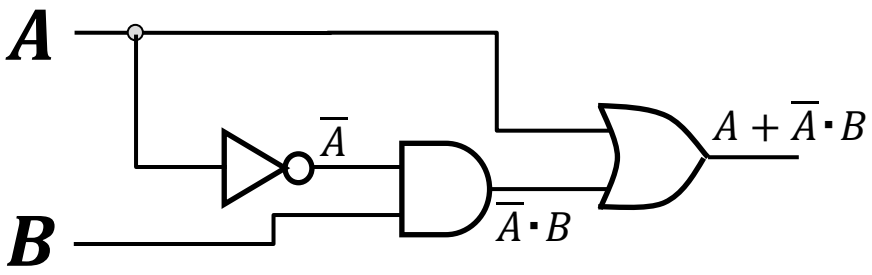
論理式とそれを表す論理回路は等価である。

- 論理式で使用されている演算子は $\overline{\ }$, ・, ＋ の3種類で、それぞれNOTゲート、ANDゲート、ORゲートで表すことができる。
  （排他的論理和⊕XORゲートもあるが、 $\overline{\ }$, ・, ＋で等価式を作ることができる）

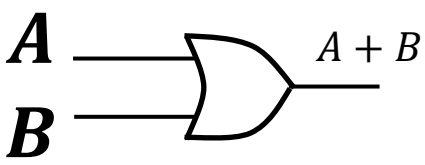
ブール代数の公理・定理を使ってなぜ論理式を簡略化するのか？

- 例えば定理7第1式 $A + \overline{A}\cdot B = A + B$ で考えると･･･

左辺の論理回路



右辺の論理回路

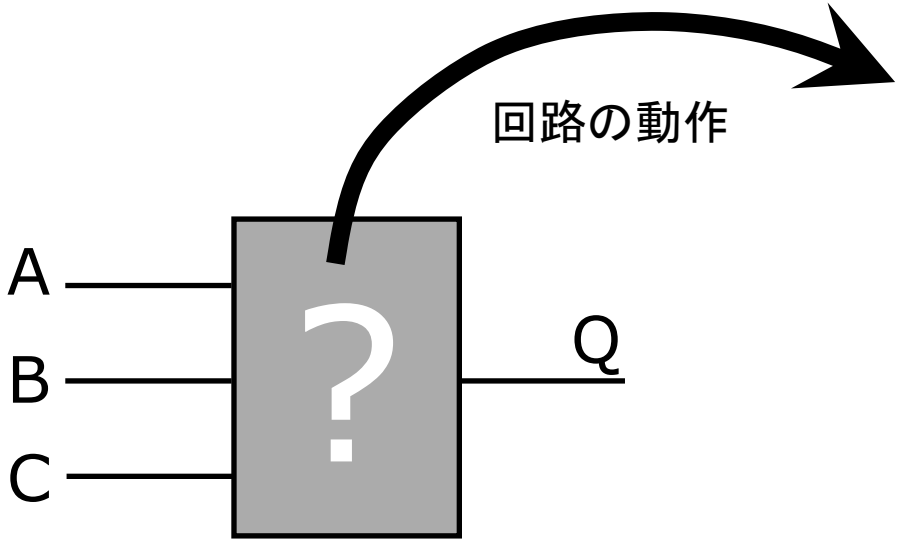


左辺よりも簡略化された右辺の論理式による回路図の方が、
使用ゲート数が少なく、結果、コストが安く、消費電力も少なく抑えられる。

組合せ論理回路設計

- 設計対象の回路は**真理値表**で表現可能

例：ある案件について、A,B,Cの３人のそれぞれがスイッチを持ち、賛成ならスイッチを押す。多数決方式で、その案件が成立したときに、出力が1となる回路を設計せよ。（３入力多数決回路）



真理値表

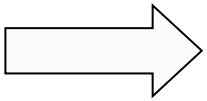
入力			出力
A	B	C	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

組合せ論理回路設計（2）

- 真理値表から論理式へ

  1つの論理式表現は1つの論理回路に対応させることができる。従って与えられた真理値表を、論理式で表す必要がある。

A	B	C	Q
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



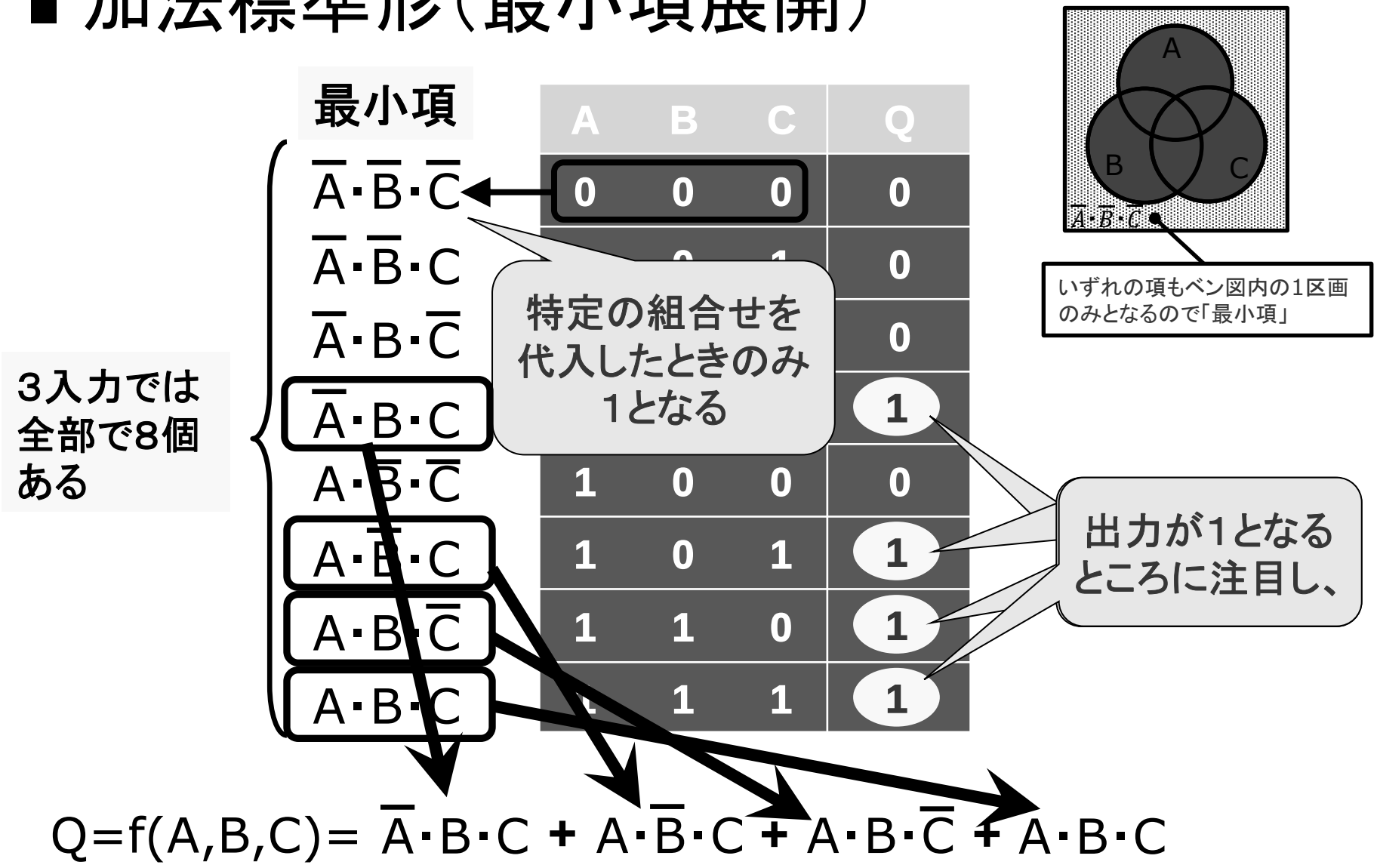
Q=f(A,B,C)

＝ ？

- 加法標準形（最小項展開）
- 乗法標準形（最大項展開）

組合せ論理回路設計(3)

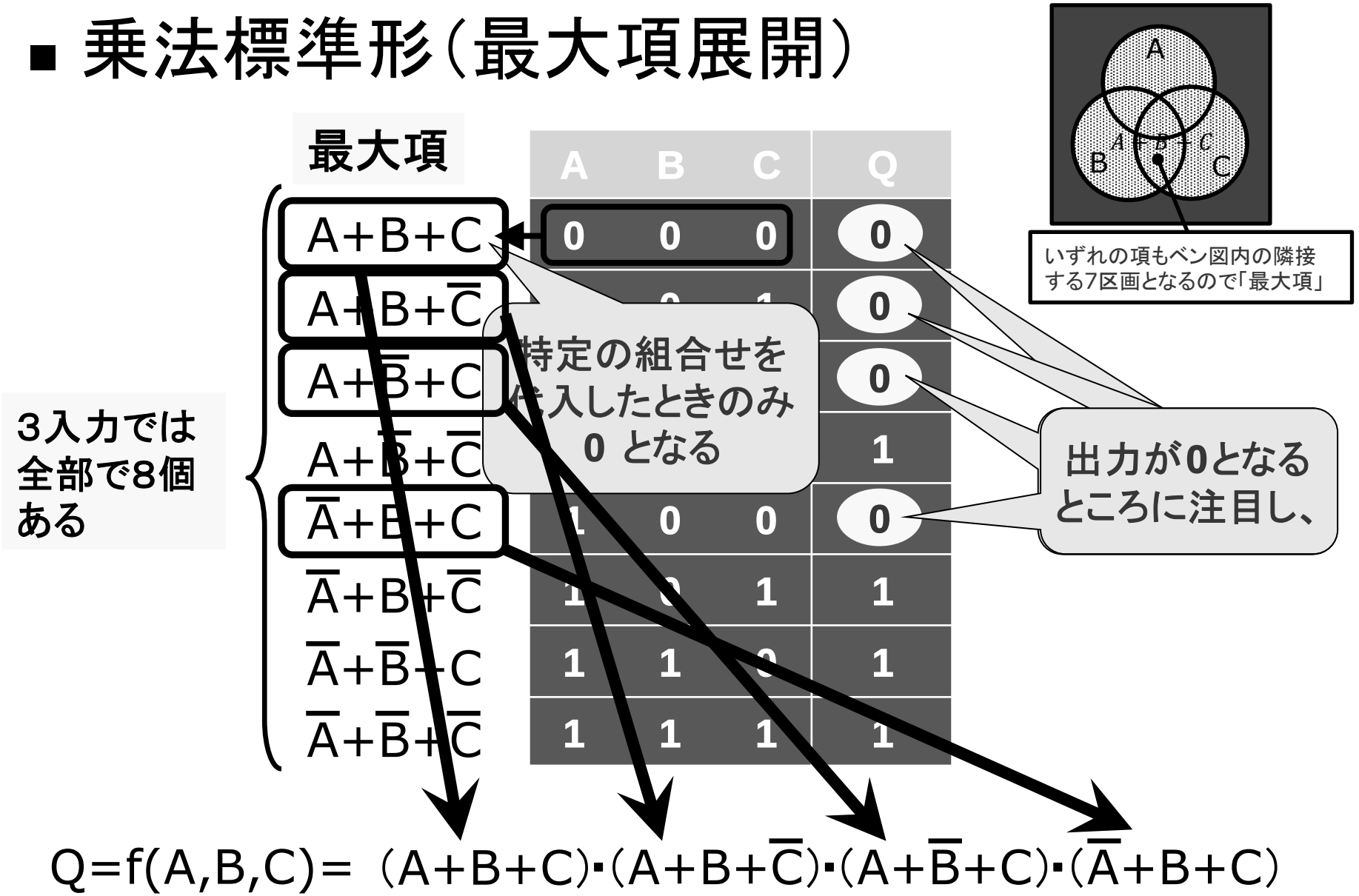
- 加法標準形(最小項展開)



$$Q=f(A,B,C)=\bar{A}\cdot B\cdot C + A\cdot\bar{B}\cdot C + A\cdot B\cdot\bar{C} + A\cdot B\cdot C$$

組合せ論理回路設計(4)

- 乗法標準形(最大項展開)



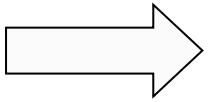
$$Q=f(A,B,C)=(A+B+C)\cdot(A+B+\overline{C})\cdot(A+\overline{B}+C)\cdot(\overline{A}+B+C)$$

ディジタル回路の作り方（2）

- 真理値表から論理式へ

  1つの論理式表現は1つの論理回路に対応させることができる。従って与えられた真理値表を、論理式で表す必要がある。

A	B	C	Q
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



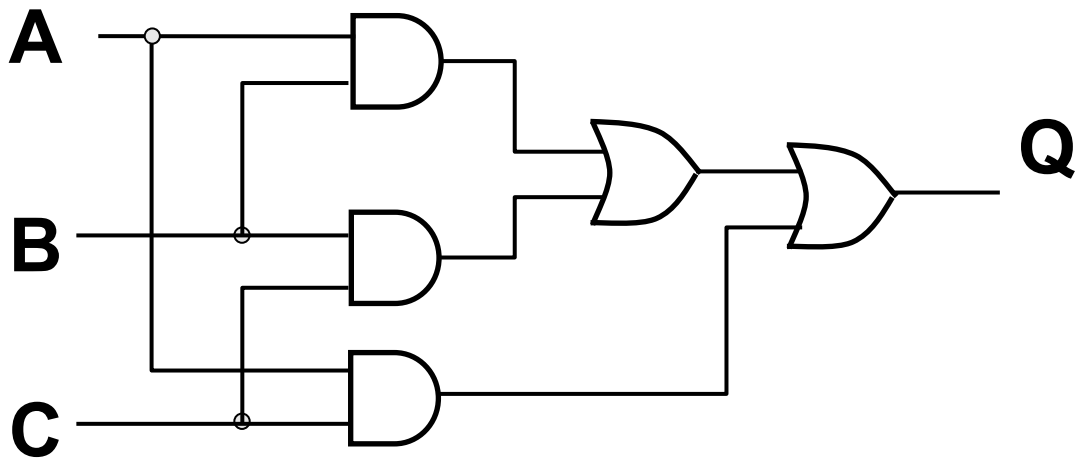
Q=f(A,B,C)

＝ ？

ディジタル回路の作り方(3)

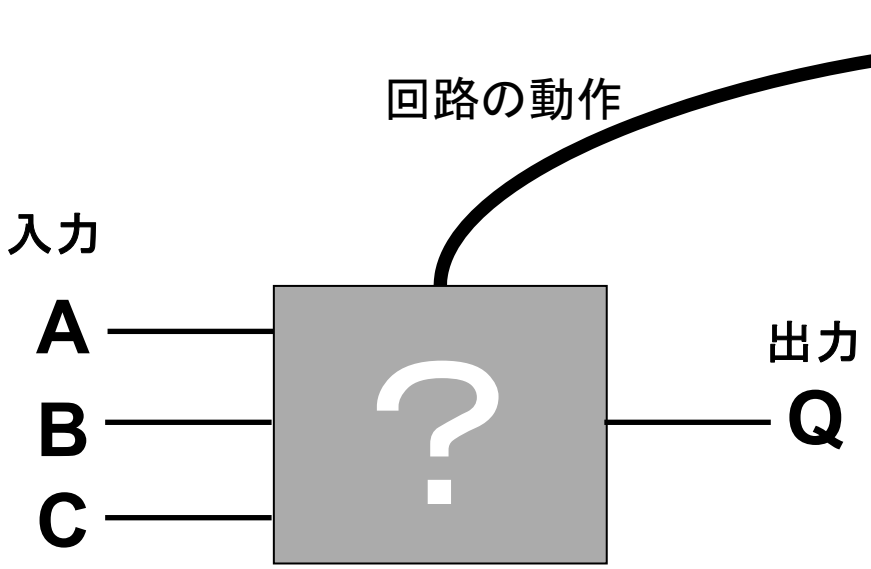
- 論理式から論理回路へ

$$Q=f(A,B,C) = A\cdot B+B\cdot C+C\cdot A$$



3入力多数決回路の例

組合せ論理回路設計の復習



真理値表

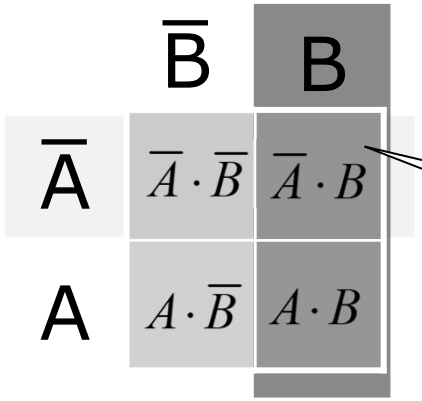
入力			出力
A	B	C	Q
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

カルノー図（Karnaugh Map）法

入力変数に応じたすべての最小項に対応するエリアを利用した簡略化手法

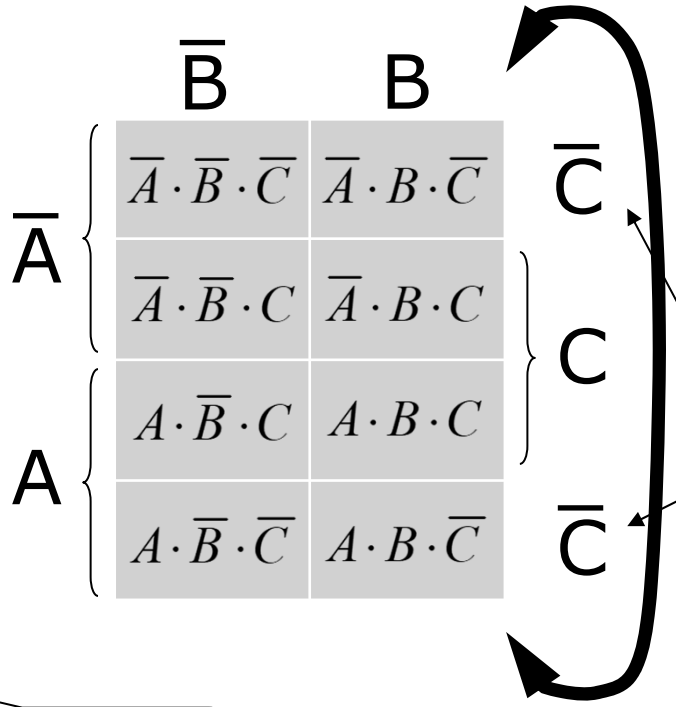
１変数の場合

２変数の場合



$\overline{A}$とBの重なり
（$\overline{A}$かつB）

３変数の場合



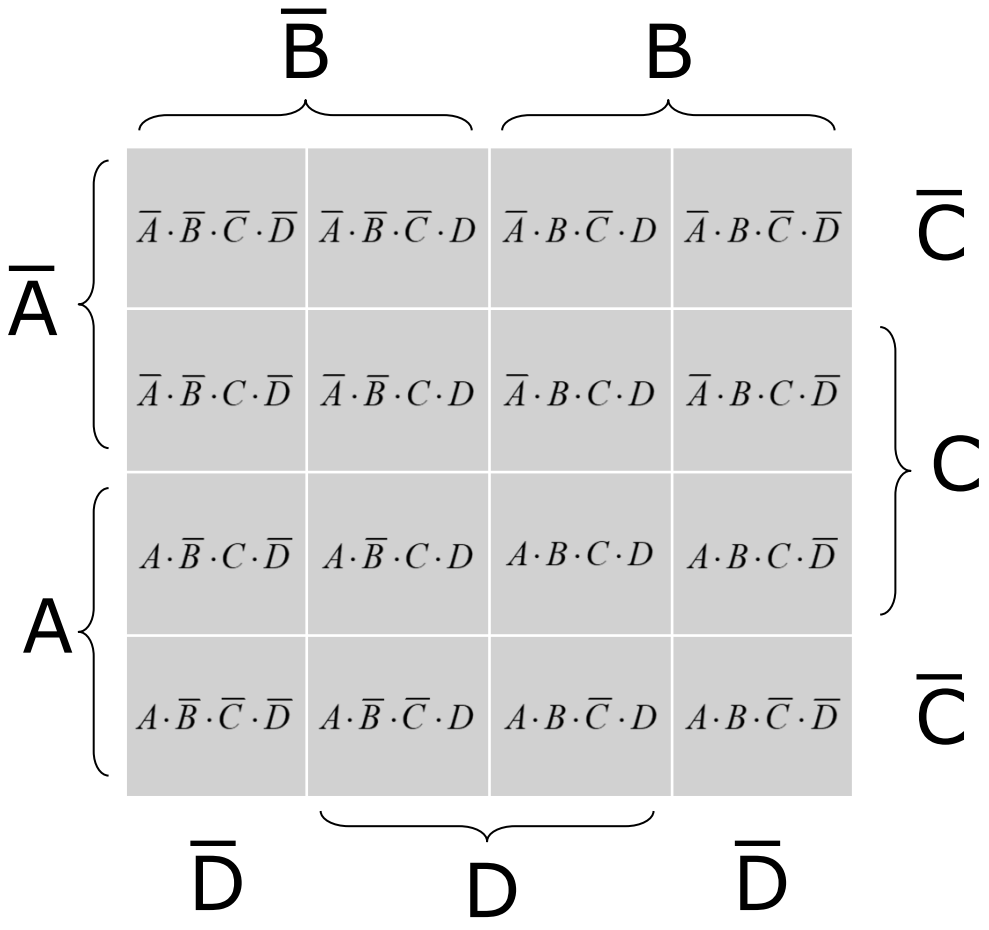
$\overline{C}$のエリアが離れているが、図の上辺と下辺がつながっていると考える

変数が1つ増えるごとに
Mapのサイズが2倍になる

カルノー図（Karnaugh Map）法

入力変数に応じたすべての最小項に対応するエリアを利用した簡略化手法

４変数の場合



参考： カルノー図とベイチ図

カルノー図

CD \ AB	00	01	11	10
00	$\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$	$\overline{A}\cdot B\cdot\overline{C}\cdot\overline{D}$	$A\cdot B\cdot\overline{C}\cdot\overline{D}$	$A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$
01	$\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot D$	$\overline{A}\cdot B\cdot\overline{C}\cdot D$	$A\cdot B\cdot\overline{C}\cdot D$	$A\cdot\overline{B}\cdot\overline{C}\cdot D$
11	$\overline{A}\cdot\overline{B}\cdot C\cdot D$	$\overline{A}\cdot B\cdot C\cdot D$	$A\cdot B\cdot C\cdot D$	$A\cdot\overline{B}\cdot C\cdot D$
10	$\overline{A}\cdot\overline{B}\cdot C\cdot\overline{D}$	$\overline{A}\cdot B\cdot C\cdot\overline{D}$	$A\cdot B\cdot C\cdot\overline{D}$	$A\cdot\overline{B}\cdot C\cdot\overline{D}$

ベイチ図

	$\overline{B}$		B	
$\overline{A}$	$\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$	$\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot D$	$\overline{A}\cdot B\cdot\overline{C}\cdot D$	$\overline{A}\cdot B\cdot\overline{C}\cdot\overline{D}$
	$\overline{A}\cdot\overline{B}\cdot C\cdot\overline{D}$	$\overline{A}\cdot\overline{B}\cdot C\cdot D$	$\overline{A}\cdot B\cdot C\cdot D$	$\overline{A}\cdot B\cdot C\cdot\overline{D}$
A	$A\cdot\overline{B}\cdot C\cdot\overline{D}$	$A\cdot\overline{B}\cdot C\cdot D$	$A\cdot B\cdot C\cdot D$	$A\cdot B\cdot C\cdot\overline{D}$
	$A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$	$A\cdot\overline{B}\cdot\overline{C}\cdot D$	$A\cdot B\cdot\overline{C}\cdot D$	$A\cdot B\cdot\overline{C}\cdot\overline{D}$
	$\overline{D}$	D		$\overline{D}$

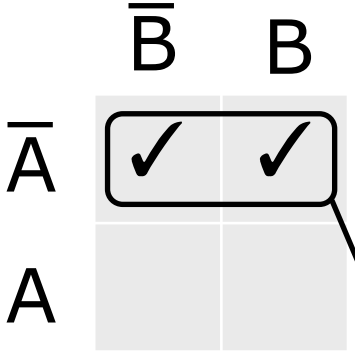
$\overline{C}$ C $\overline{C}$

論理式の簡略化の考案者が異なることから、カルノー図とベイチ図とそれぞれ呼ばれていますが、横軸・縦軸の表記方法が異なるだけで、使用している論理変数による最小項が各マスに対応する、という意味で同じ考え方をしている図です。
本講義では、特にこれらを区別していません。（カルノー図では真理値表の出力値を各セルの最小項の値として1も0も書き込む、ベイチ図では、1の値のみ書き込み、0は空欄のままにしておく、といった違いはありますが、次項以降の簡略化方法は同じで、どちらの図でも同じ結果が得られます。）

カルノー図（Karnaugh Map）法

入力変数に応じたすべての最小項に対応するエリアを利用した簡略化手法

論理式 $Q = f(A,B) = \overline{A} \cdot \overline{B} + \overline{A} \cdot B$ の簡略化を考える。



論理式に含まれる最小項のエリアに「✓」印をつける。

上下左右に隣接する2^n個の「✓」印をグループ化する

この2つの「✓」印をまとめて読むと

$$Q = \overline{A}$$

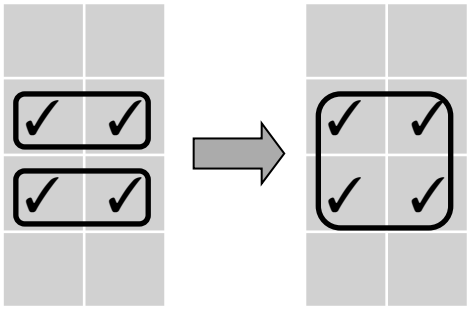
と読める。（変数Bのバーが付くエリアと付かないエリアにグループがまたがるので変数Bが消去される。
計算上は $\overline{A} \cdot (\overline{B} + B) = \overline{A} \cdot 1 = \overline{A}$ ということ。）

2^n個をグループ化したところではn個の変数が消去される。

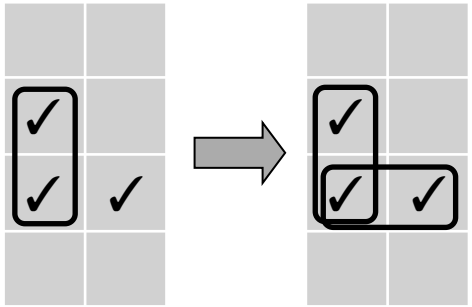
カルノー図（Karnaugh Map）法

「v」印をグループ化するときのポイント

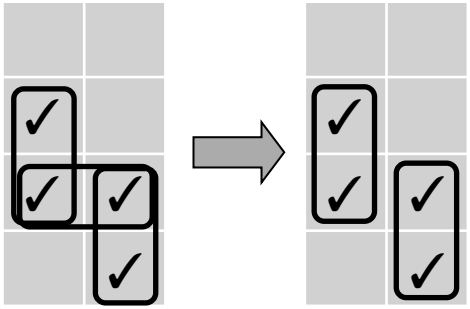
① グループはできる限り大きく作る



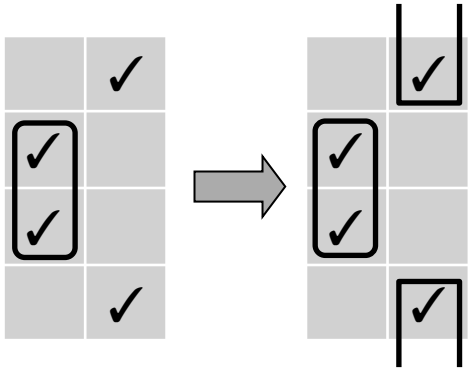
② グループを構成するとき、同じ「✓」印を別のグループに使っても良い。



③ すでに他のグループに含まれている「✓」印だけを使ってグループを構成しない。



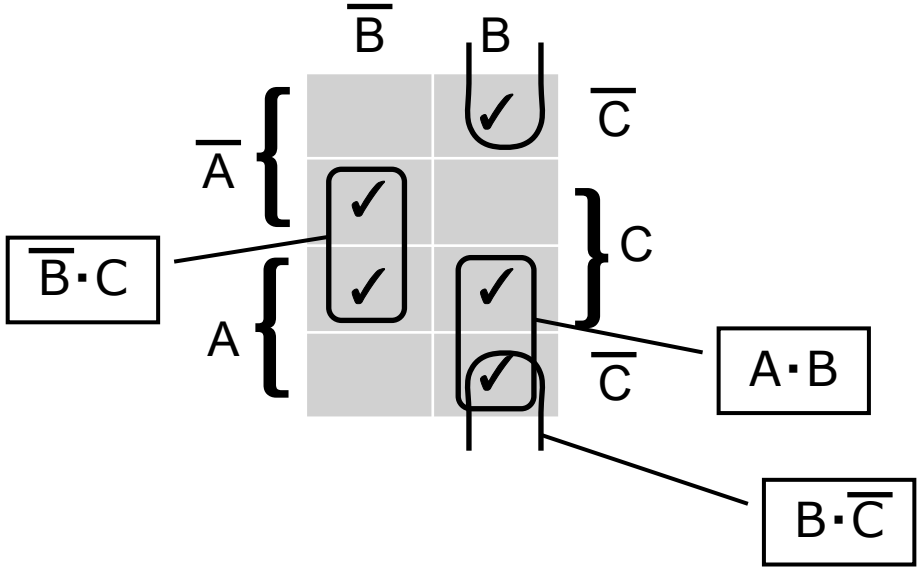
④ カルノー図の上辺と下辺、左辺と右辺はそれぞれ連続していることに注意する。



カルノー図法による論理式の簡略化

真理値表

入力			出力
A	B	C	Q
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



$$Q = A \cdot B + B \cdot \overline{C} + \overline{B} \cdot C$$

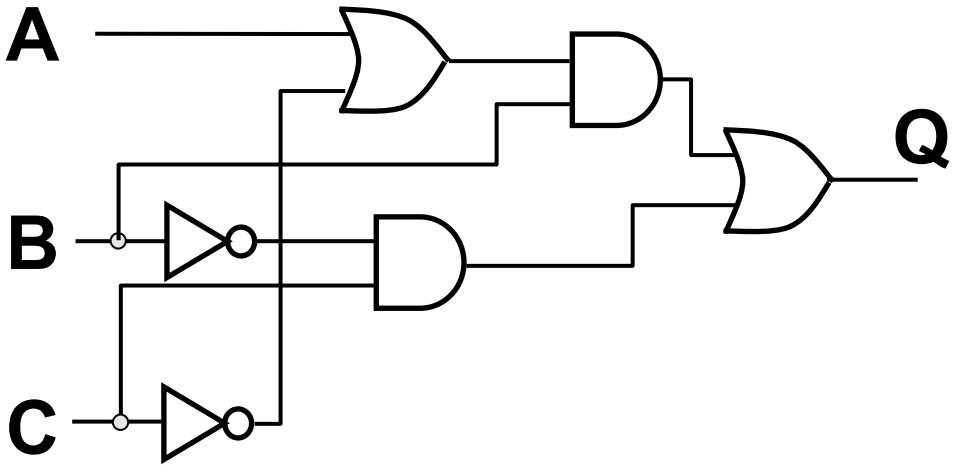
$$= A \cdot B + (B \oplus C)$$

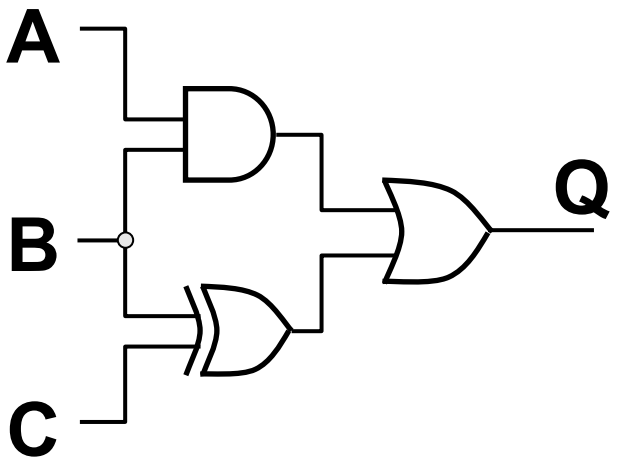
XORを使うならこのようになる。

論理式から回路図へ

$$Q = A \cdot B + B \cdot \overline{C} + \overline{B} \cdot C$$
$$= (A + \overline{C}) \cdot B + \overline{B} \cdot C$$

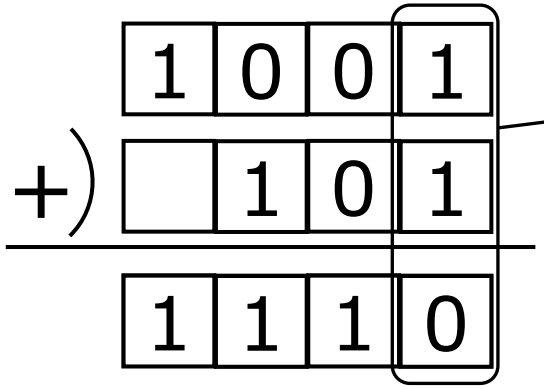
$$Q = A \cdot B + (B \oplus C)$$



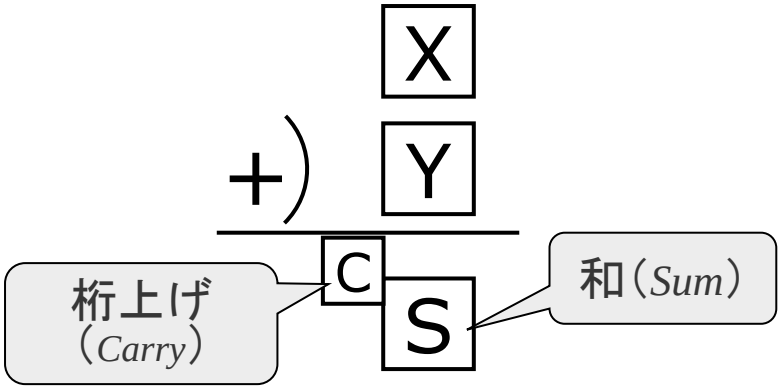


加算器・減算器

2進数の足し算を考える。　例 1001(2)+101(2)

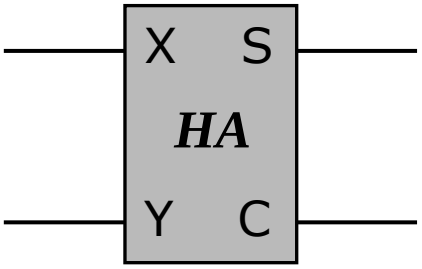


最下位ビット（***Least Significant Bit*** **=LSB**）の
加算の入力と出力を見ると･･･



入力		出力	
X	Y	S	C
0	0		
0	1		
1	0		
1	1		

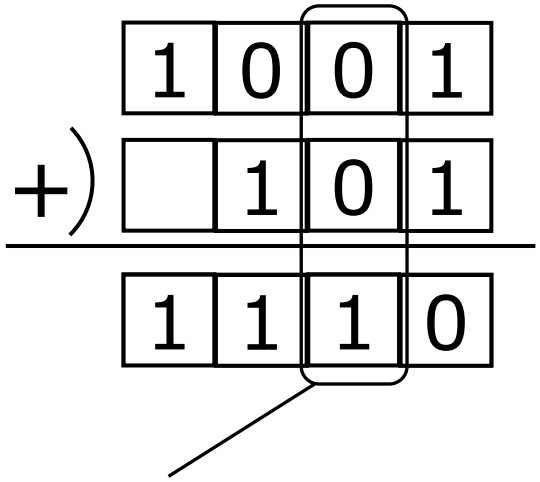
半加算器（***Half Adder***）



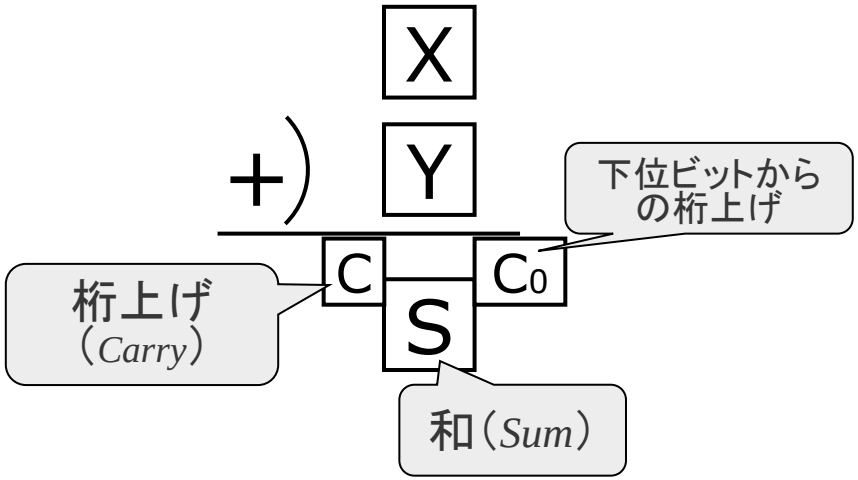
左は半加算器の真理値表

加算器・減算器

2進数の足し算を考える。　例 1001(2)+101(2)

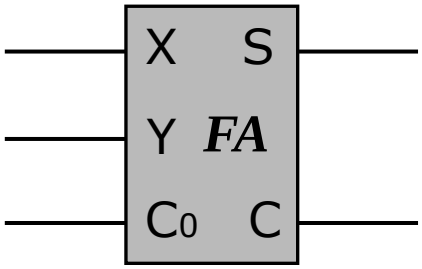


最下位ビット（LSB）以外のビットの
加算の入力と出力を見ると･･･



入力			出力	
X	Y	C_0	S	C
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

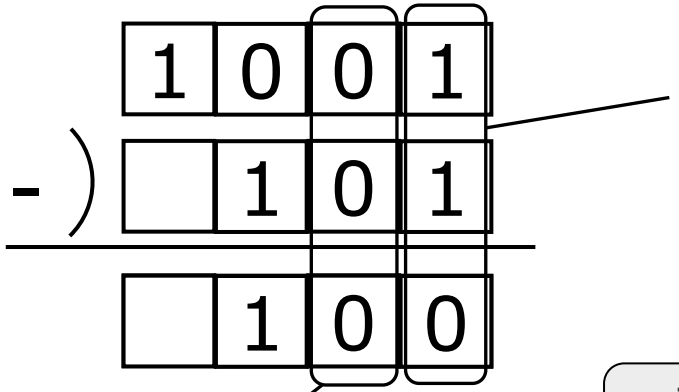
上は全加算器の真理値表



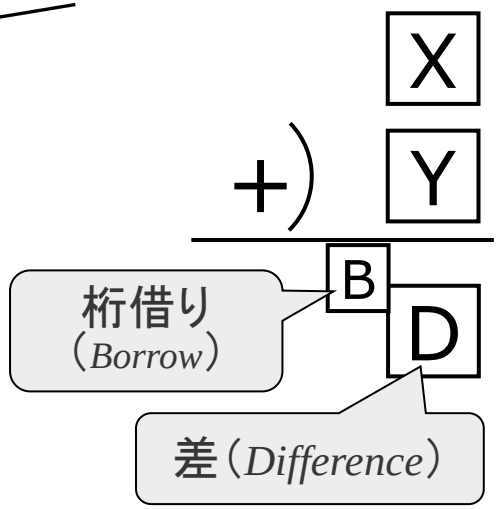
全加算器（***Full Adder***）

加算器･減算器

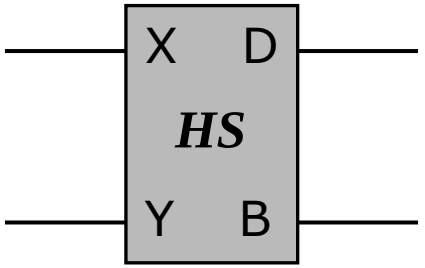
減算器についても同様に考えられる。　例 1001(2)-101(2)



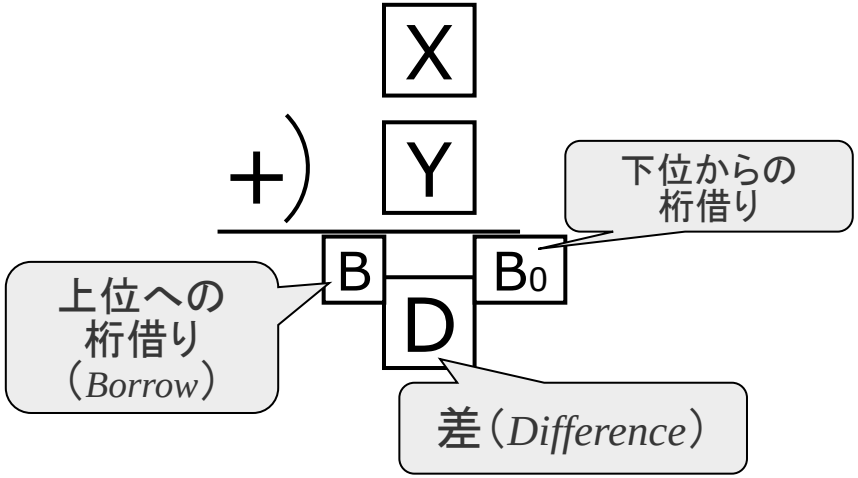
LSBの減算の入力と出力を見ると･･･

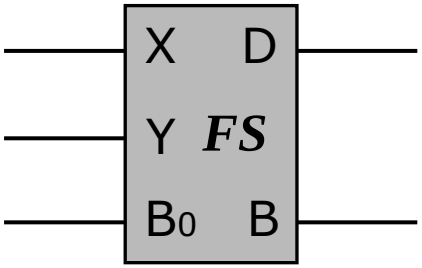


半減算器（***Half Subtractor***）



LSB以外のビットの減算の入力と出力を見ると･･･



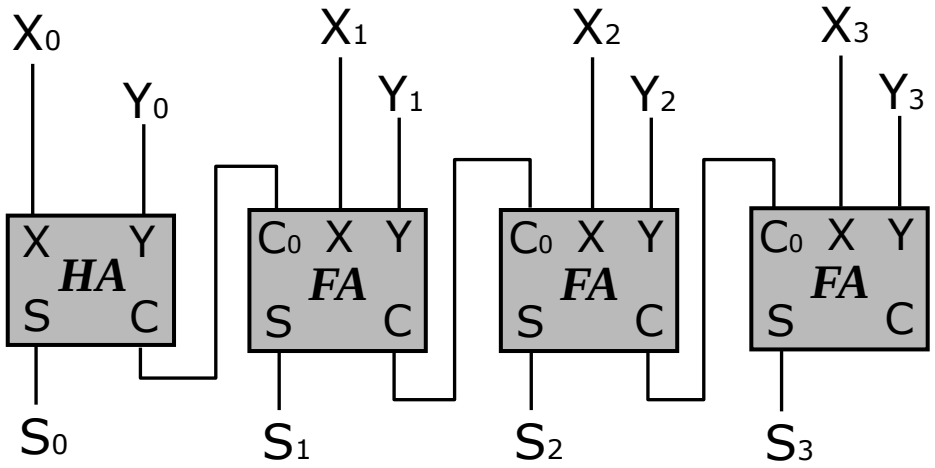


全減算器（***Full Subtractor***）

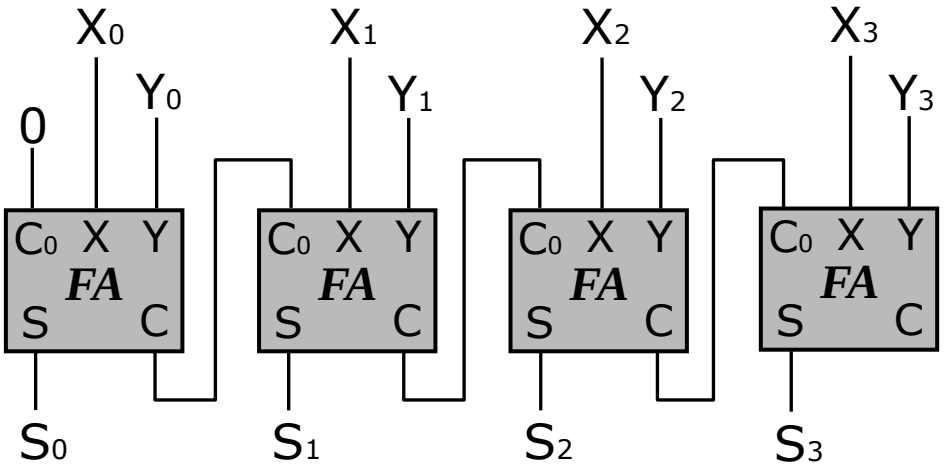
加算器

- 4ビットの加算を行うには

	X_3	X_2	X_1	X_0
+	Y_3	Y_2	Y_1	Y_0
	S_3	S_2	S_1	S_0



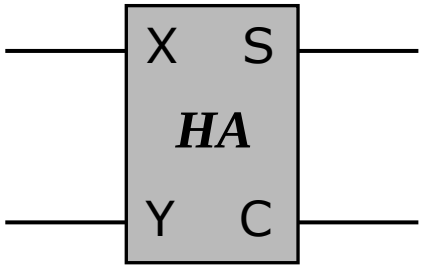
またはLSB用のHAの代わりに、次のようにFAを使うこともできる。



半加算器・全加算器の設計

- HA,FAは組合せ論理回路で設計可能

半加算器（*Half Adder*）



半加算器の真理値表

入力		出力	
X	Y	S	C
0	0		
0	1		
1	0		
1	1		

出力の論理式

$$\begin{cases} S = \\ C = \end{cases}$$

*HA*の回路図

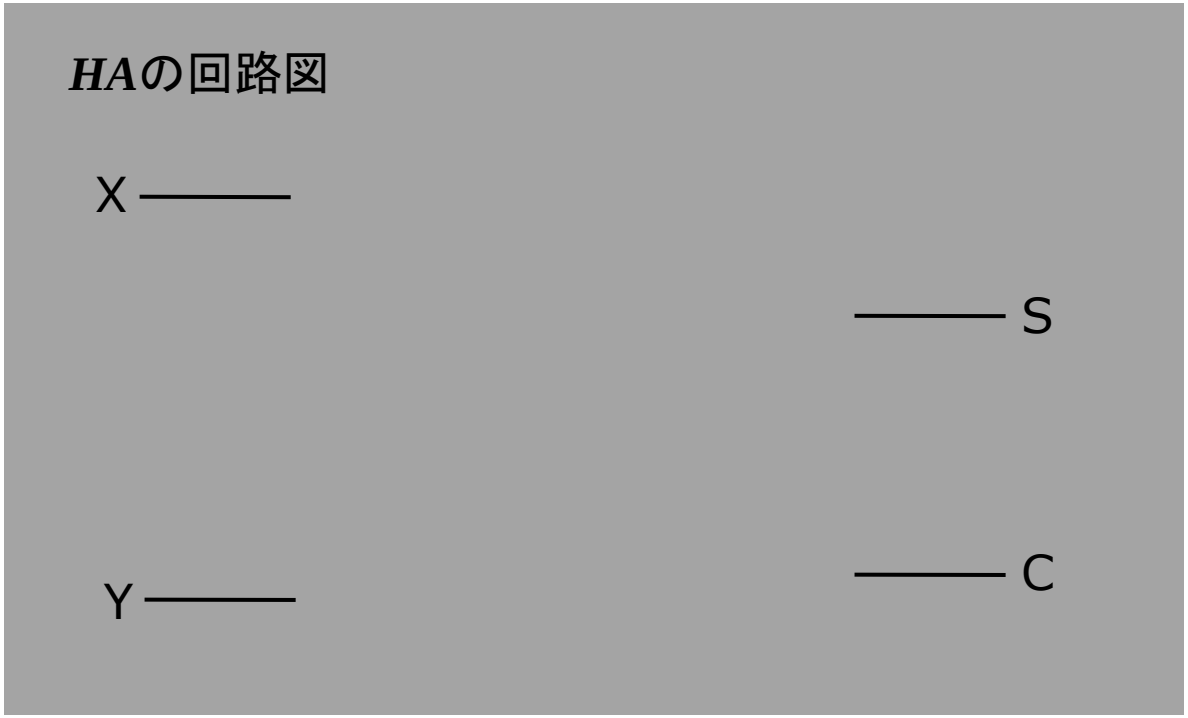
半加算器・全加算器の設計

- HAを作るのに、もしXORゲートが使用出来ないときは

出力の論理式

$$\begin{cases} S = \\ C = \end{cases}$$

***HA*の回路図**



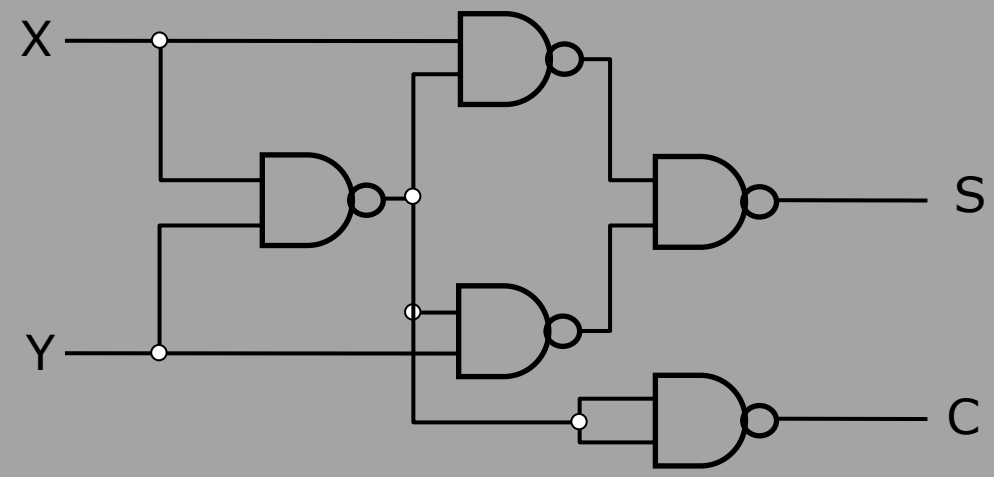
半加算器・全加算器の設計

- HAを作るのに、もしNANDゲートしか使用出来ないときは

出力の論理式

$$\begin{cases} S = \\ C = \end{cases}$$

*HA*の回路図

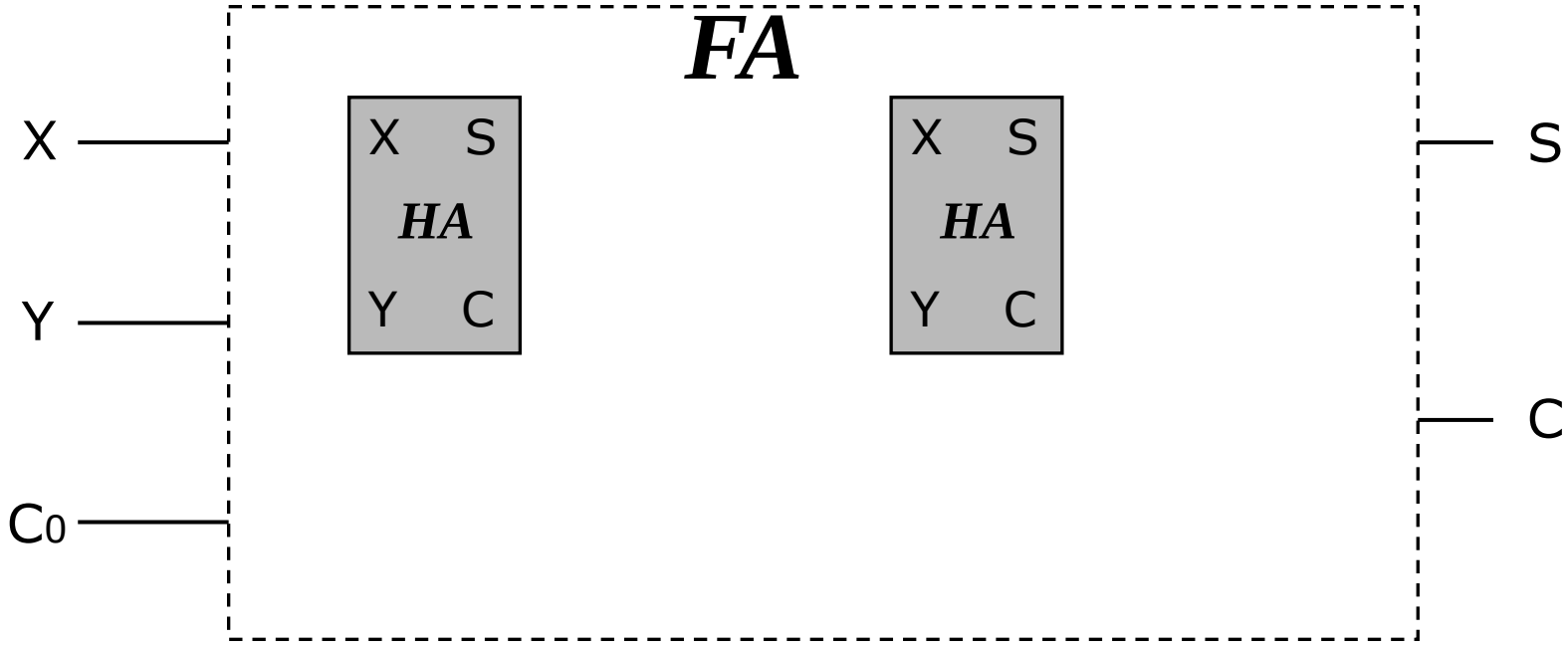


これはあとで解説する
NAND等価回路を参照

半加算器・全加算器の設計

- FAをHAと同様に真理値表から回路図を得ても良いが、ここでは、HAを利用してFAを構成してみる。

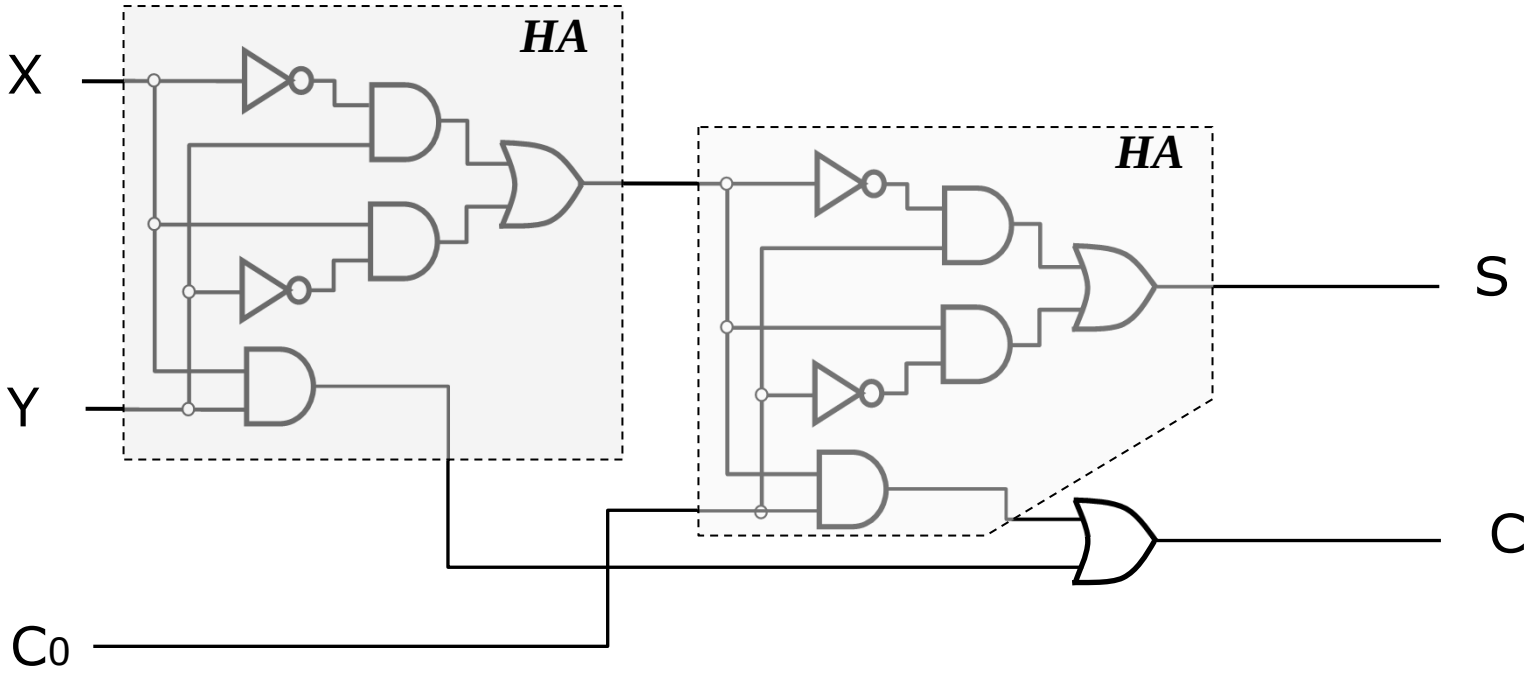
FAでは、下位からの桁上げをさらに加算しなければならないので、
以下のようにHAを2つ使って構成できることになる。



半加算器・全加算器の設計

- HAを利用したFAの詳細な構成例

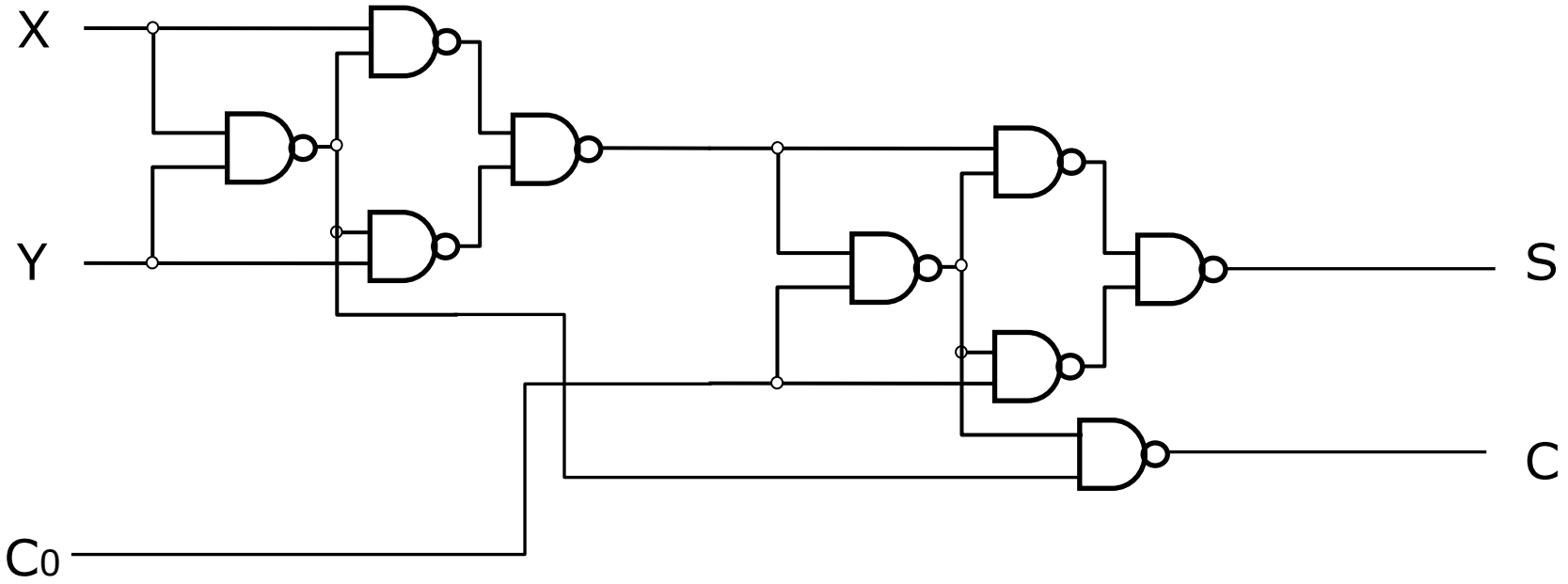
*FA*構成例1：NOT,AND,ORゲートを使用



半加算器・全加算器の設計

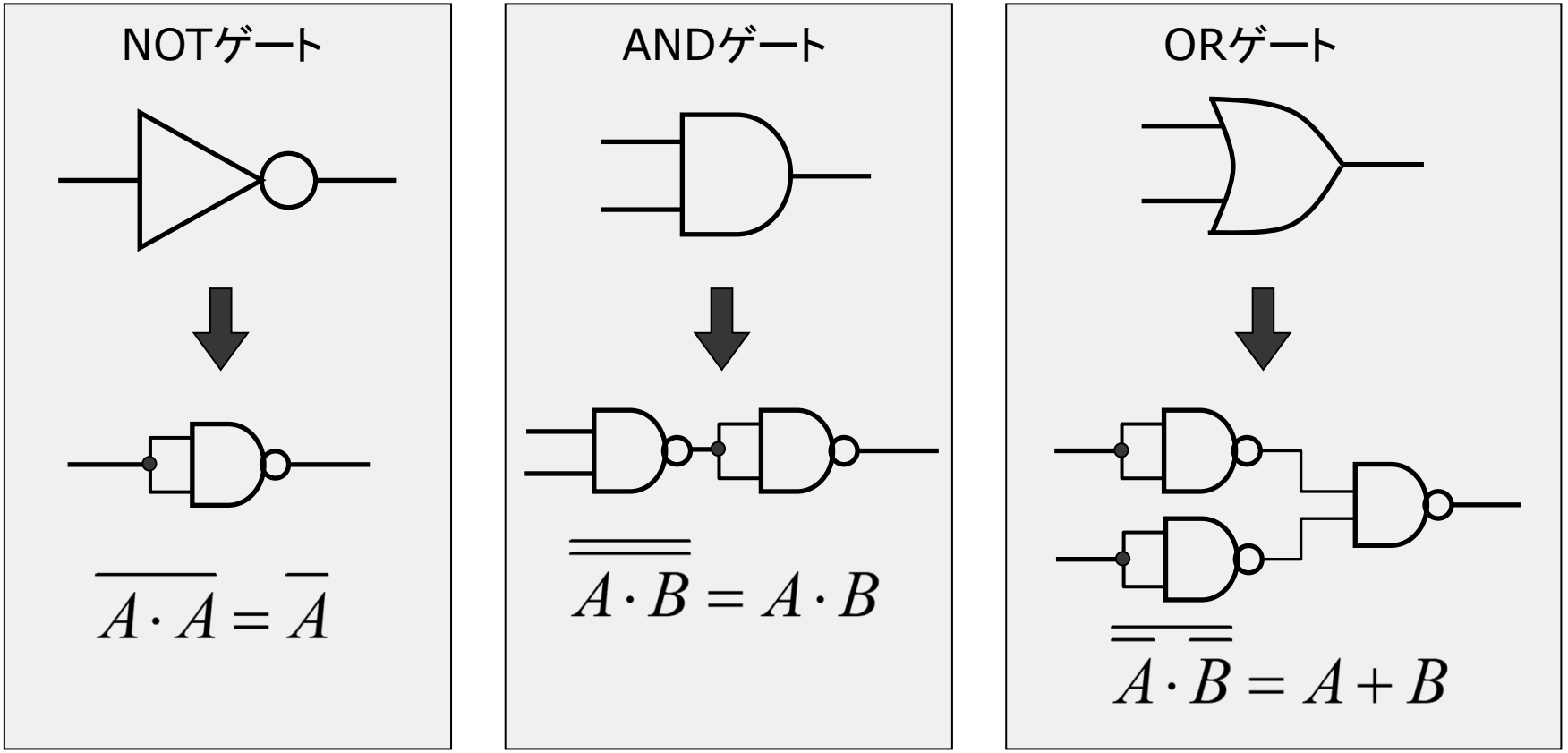
- HAを利用したFAの詳細な構成例

*FA*構成例2：NANDゲートのみを使用



NAND等価回路、NOR等価回路

- すべての論理回路は、NAND,NORのどちらかのゲートのみで構成できる。



減算器の設計

- 加算器を利用した減算器の設計

減算器を設計するために、ここまで解説してきた加算器と同様の
手順をとることができる。

しかし、ここでは減算の仕組みを以下のようにとらえて回路を構成する。

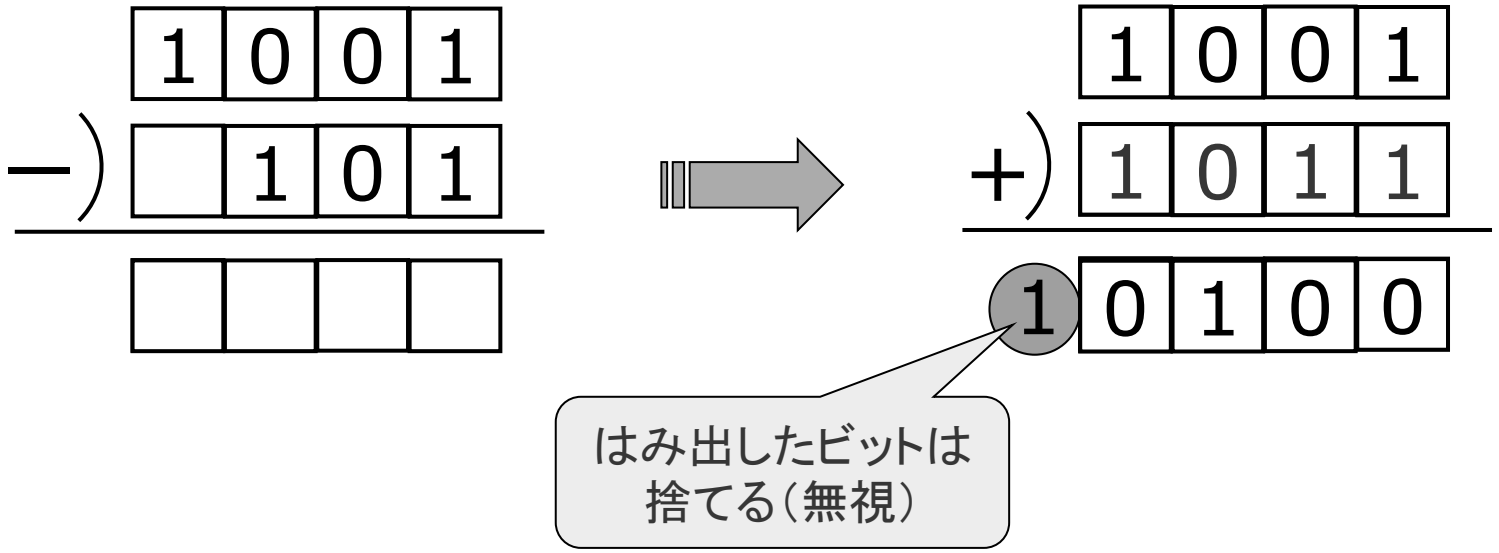
例. 1001(2)-101(2)の減算は、
1001(2)+”101(2)の2の補数“
と考えれば、加算に直して計算することが可能であると分かる。

ここで、2の補数の作り方を思い出すと、データのビット数を考慮し、
引く数の各ビットを反転し（すなわちNOTをとり）、最下位ビット（LSB）に
1を加える。この2進数を引かれる数に加えれば良い。

減算器の設計

- 加算器を利用した減算器の設計

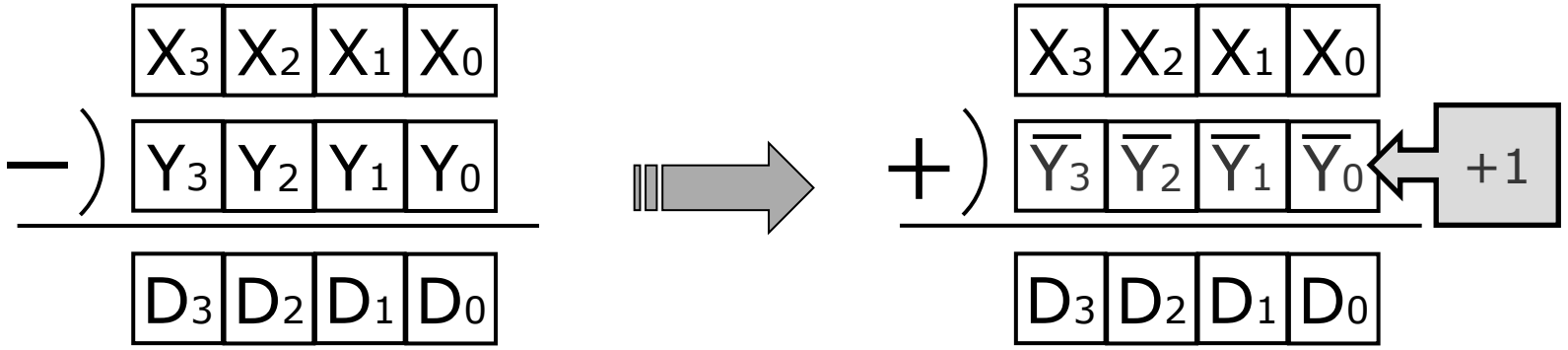
データが4ビットのときの減算の例　1001(2)-101(2)

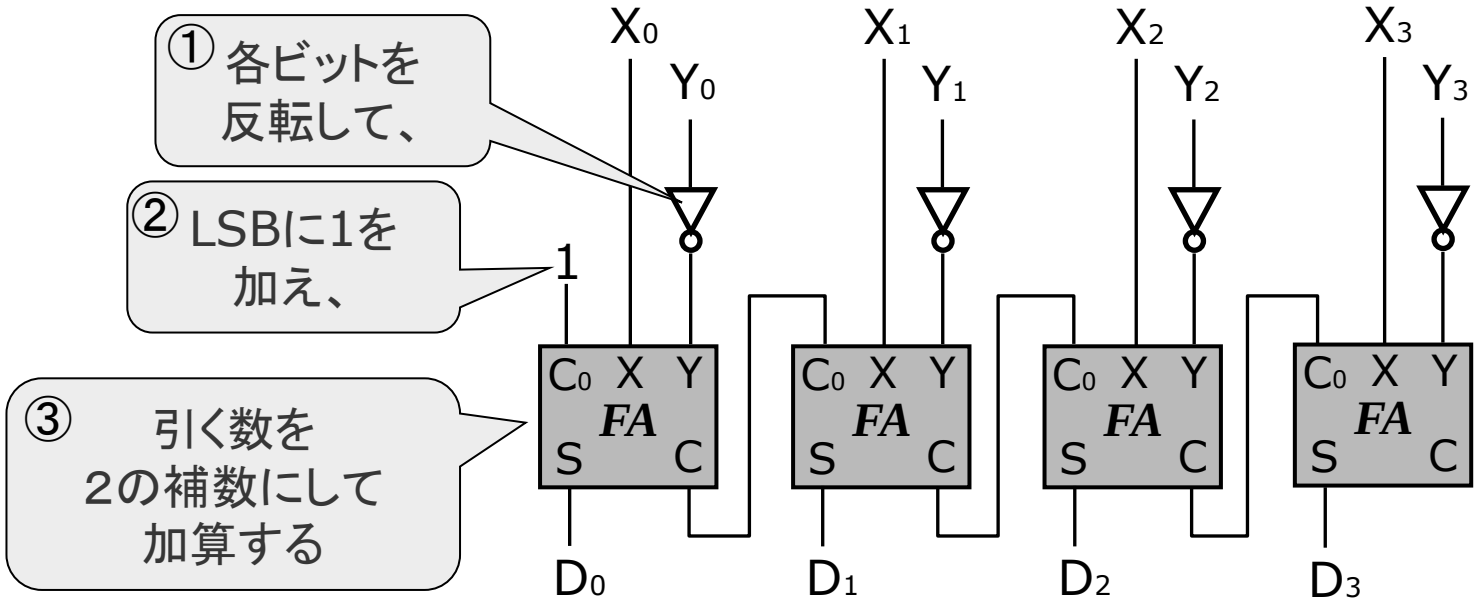


従って4ビットの減算を行う回路は次のように構成することができる。

減算器の設計

- 4ビットの減算を行うには

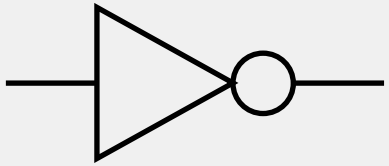




NAND等価回路、NOR等価回路

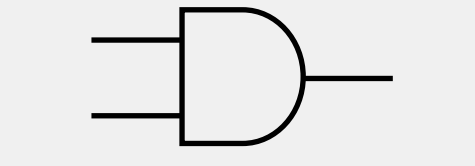
- NAND等価回路を描くために

NOTゲート



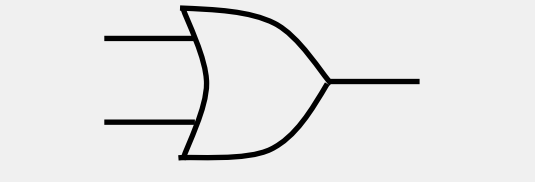
$$\overline{A \cdot A} = \overline{A}$$

ANDゲート



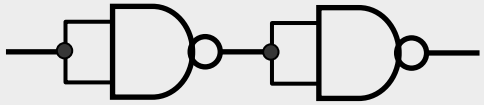
$$\overline{\overline{A \cdot B}} = A \cdot B$$

ORゲート



$$\overline{\overline{A} \cdot \overline{B}} = A + B$$

NOT,AND,ORゲートを上記のNANDゲートですべて置き換えても良いが・・・



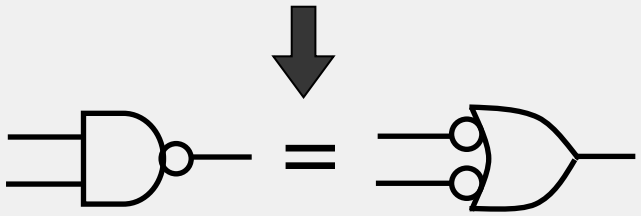
のような2連続のNOTを二重否定で、消去するのは手間

NAND等価回路、NOR等価回路を作成をしやくするゲートの記法（1）

- MIL記法：米国軍（Military）を意味する記法、ANSI記法（これまでの記法）とほぼ同じ
- NAND,NORゲートとド・モルガンの定理を見比べると・・・

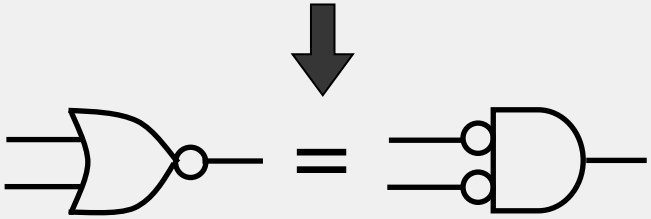
NANDゲート

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$



NORゲート

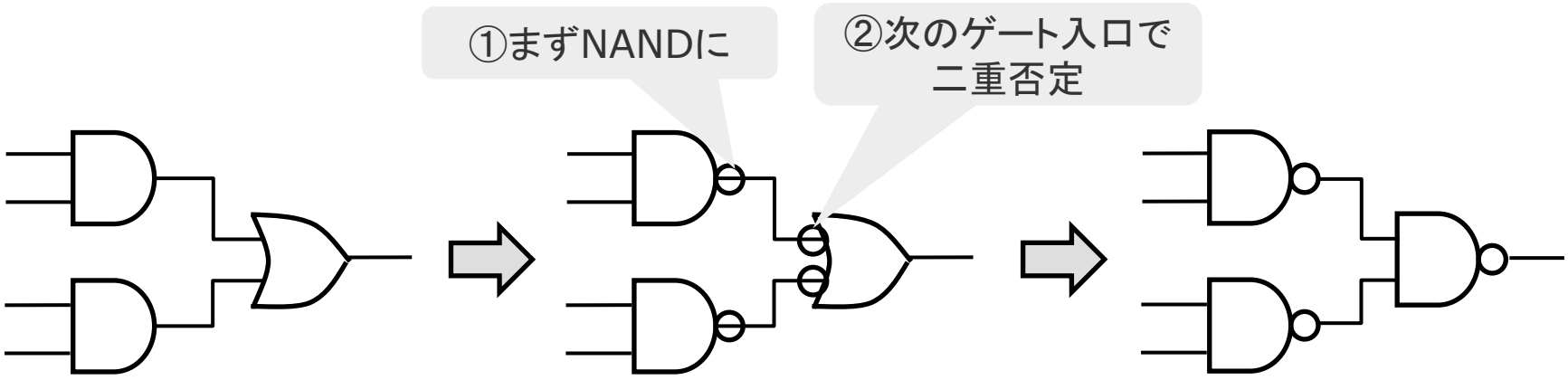
$$\overline{A + B} = \overline{A} \cdot \overline{B}$$



○記号をNOTゲートと考えれば良い

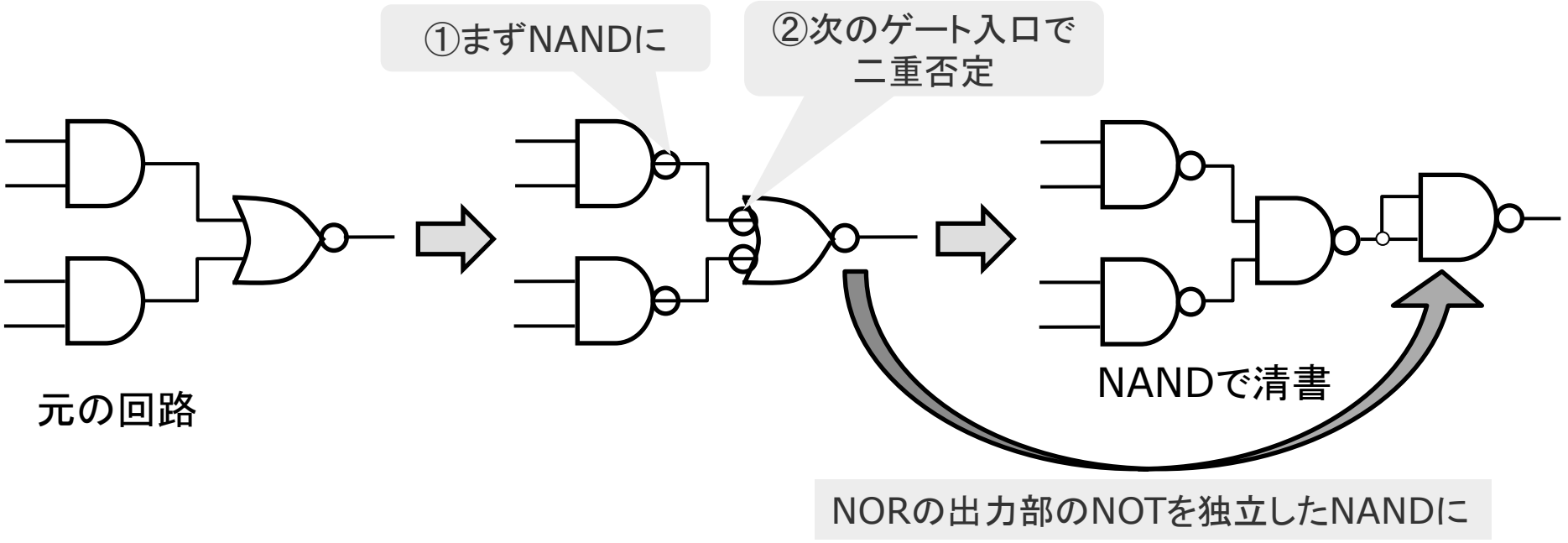
NAND等価回路、NOR等価回路を作成をしやくするゲートの記法（2）

- 例1：下記の回路のNAND等価回路を作るには・・・



NAND等価回路、NOR等価回路を作成をしやくするゲートの記法（2）

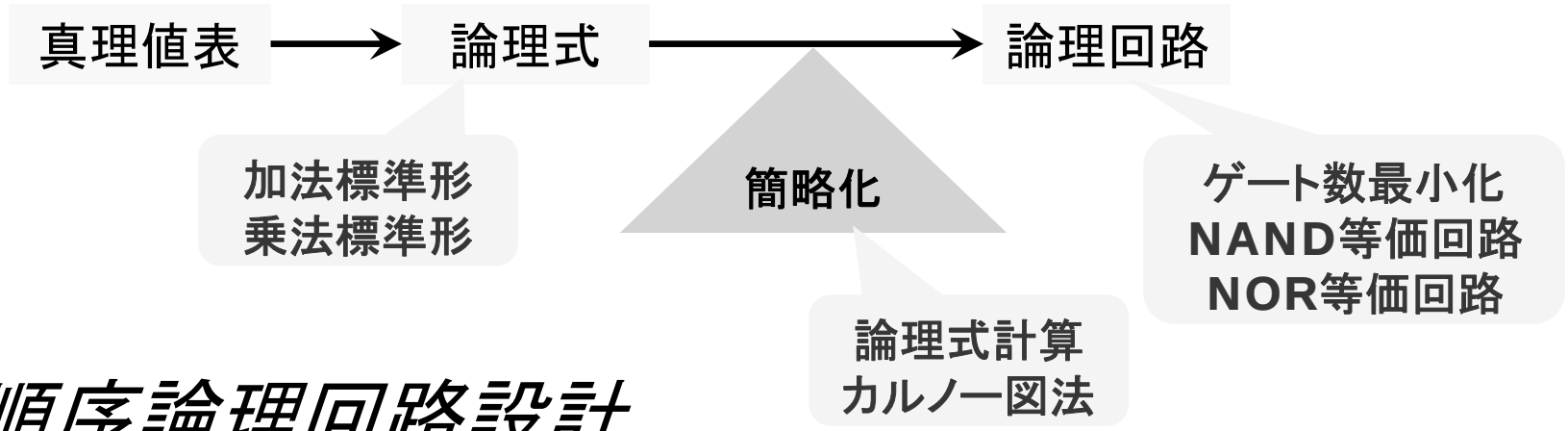
- 例2：下記の回路のNAND等価回路を作るには・・・



これらの図による変換以外にも論理式の変形によるNAND化・NOR化も可能ではある。

順序論理回路設計とは

- *組合せ論理回路設計*
  - ➢基本ゲートの組合せ



- *順序論理回路設計*
  - ➢記憶をともなう回路の設計
  - ➢“記憶”を実現する基本モジュール
  - ➢その基本モジュールと基本ゲートを組み合わせて設計（組合せ論理回路設計技術も利用）

順序論理回路設計は、2年次の「デジタルシステム」にて