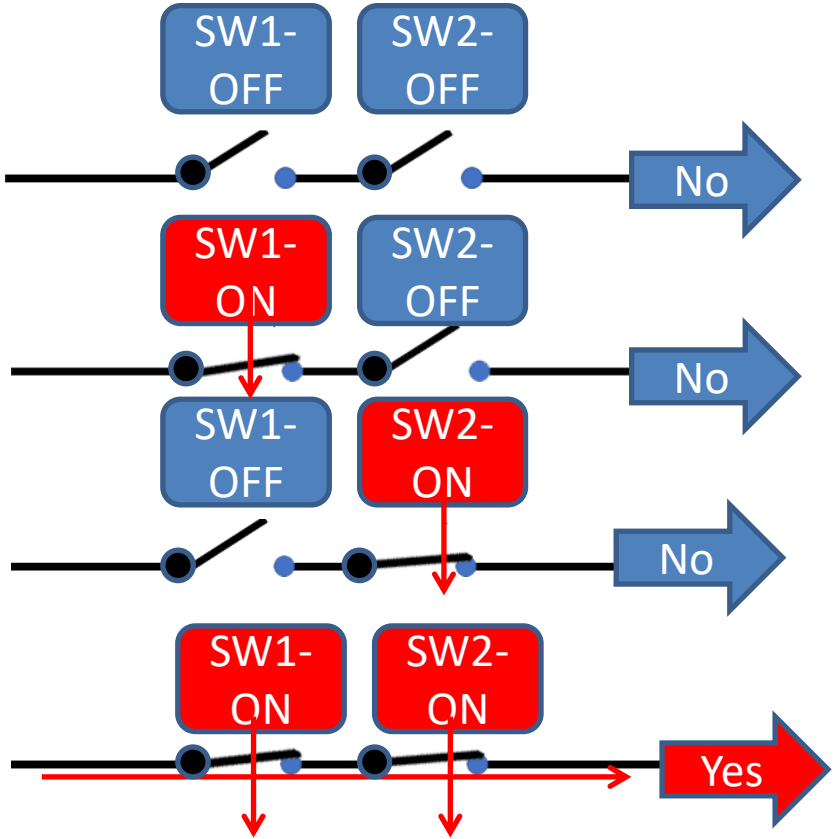
情報Ⅰ

1-8　　論理回路
1-8-1　基本論理回路AND
1-8-2　基本論理回路OR
1-8-3　基本論理回路NOT
1-8-4　真理値表完成問題
1-8-5　半加算器回路
1-8-6　全加算器回路
1-8-7　NAND回路

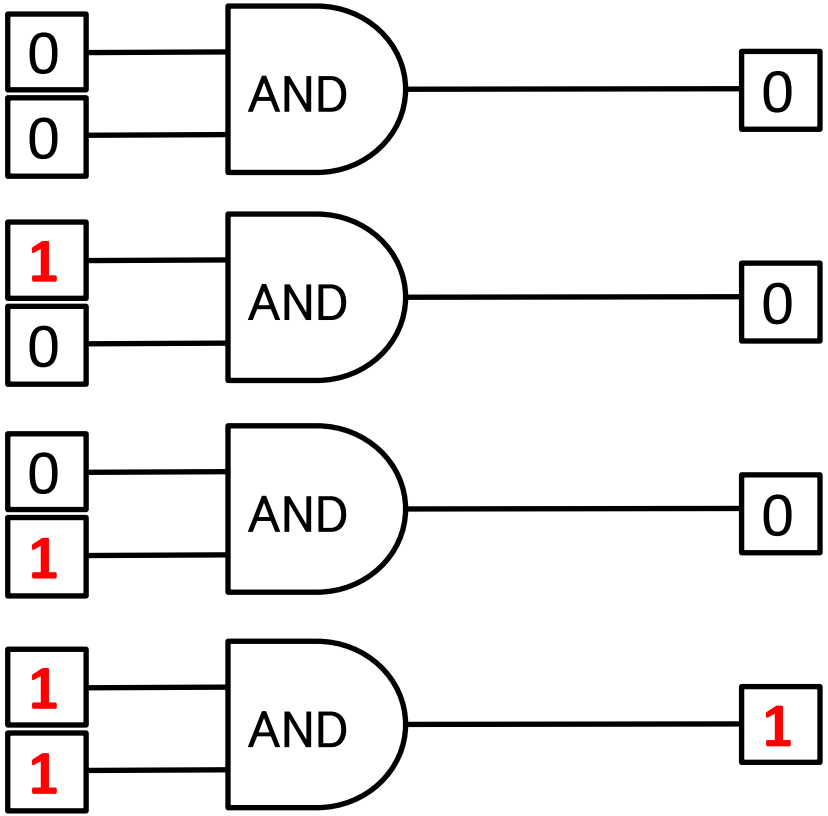
1-8-1　AND回路



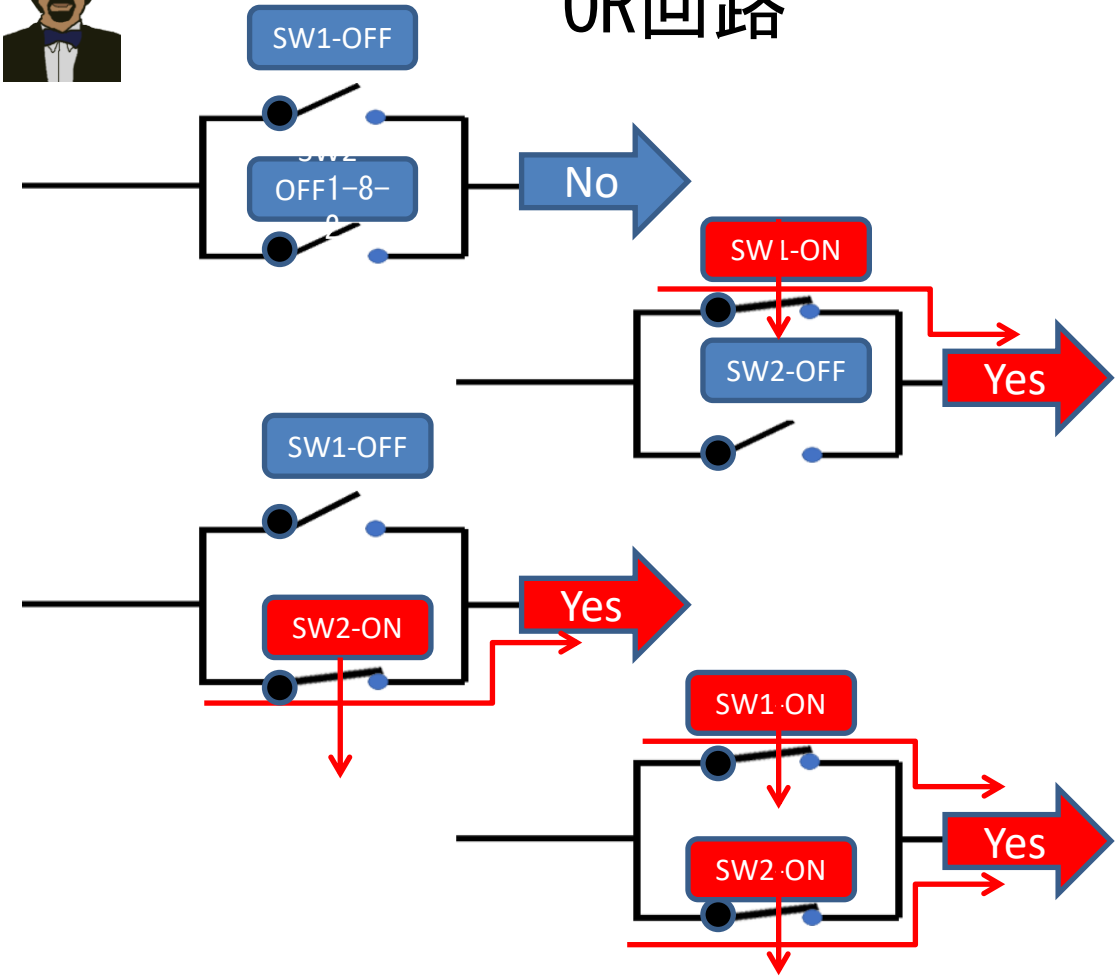
SW	電流
OFF➡入力0	No➡出力0
ON➡入力1	Yes➡出力1

AND回路の真理値表

INPUT-1	INPUT-2	OUTPUT
0	**0**	**0**
0	1	0
1	0	0
1	1	1


0
0
AND
0
1
0
AND
0
0
1
AND
0
1
1
AND
1

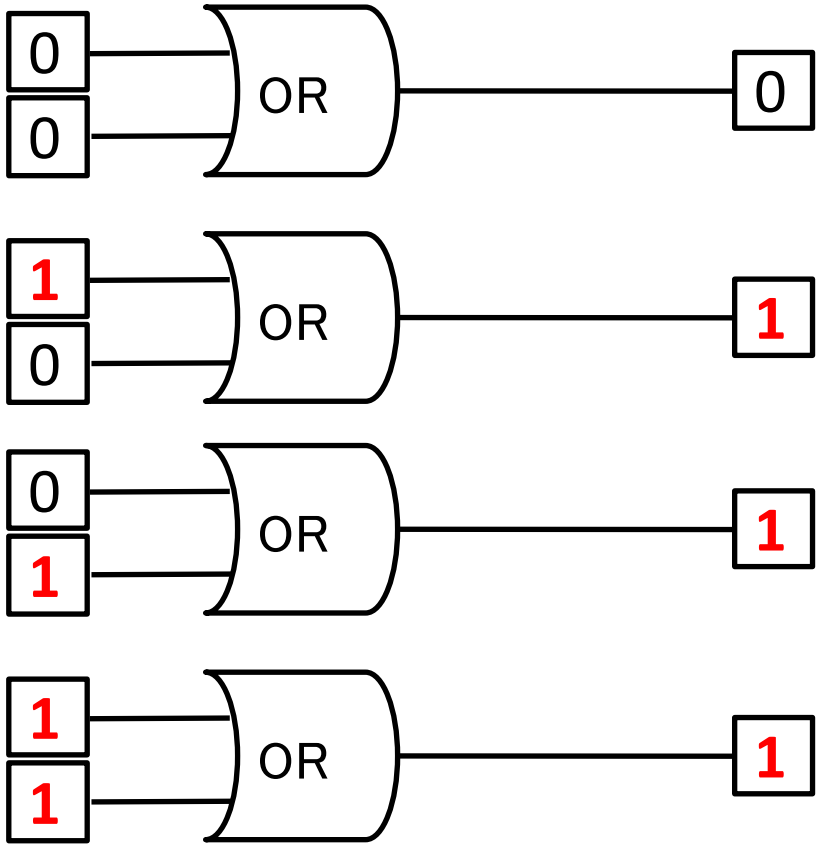
OR回路



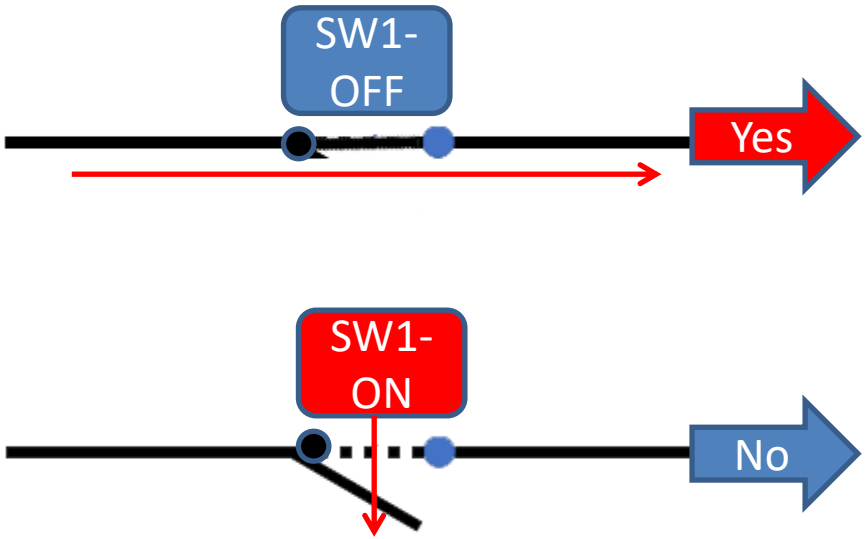
SW	電流
OFF➡入力0	No➡出力0
ON➡入力1	Yes➡出力1

OR回路の真理値表

INPUT-1	INPUT-2	OUTPUT
0	0	0
0	1	1
1	0	1
1	1	1


0
0
OR
0
1
0
OR
1
0
1
OR
1
1
1
OR
1

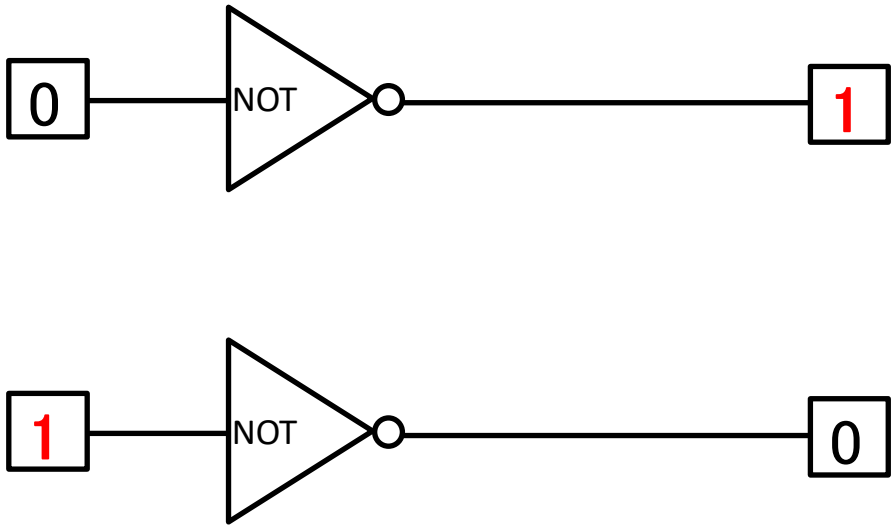
1-8-3　NOT回路



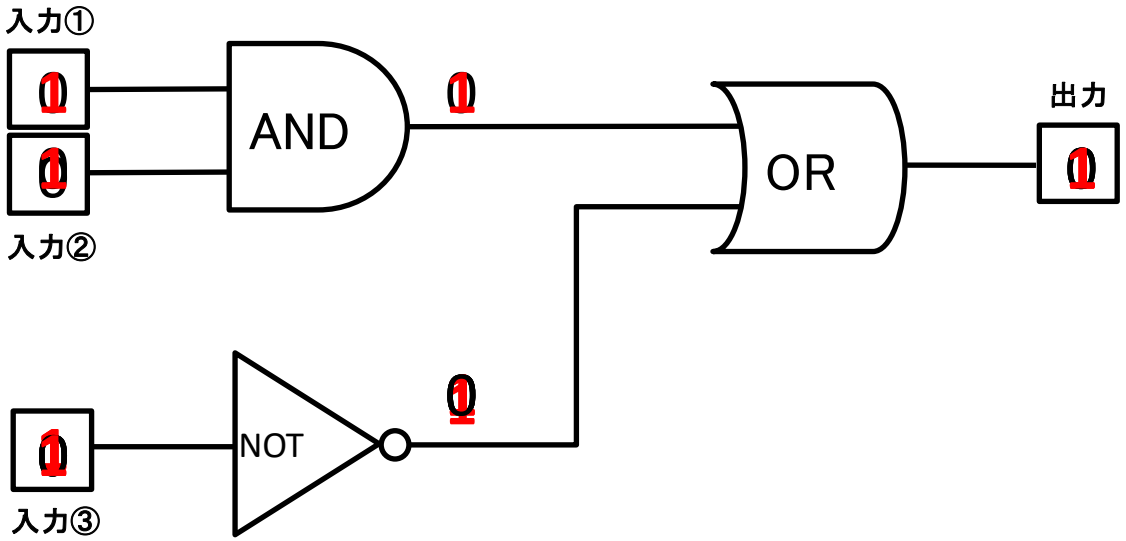
SW	電流
OFF➡入力0	Yes➡出力1
ON➡入力1	No➡出力0

NOT回路の真理値表

INPUT-1	OUTPUT
0	1
1	0


0
NOT
1
1
NOT
0

1-8-4　論理回路練習問題1：右の真理値表を完成させよう

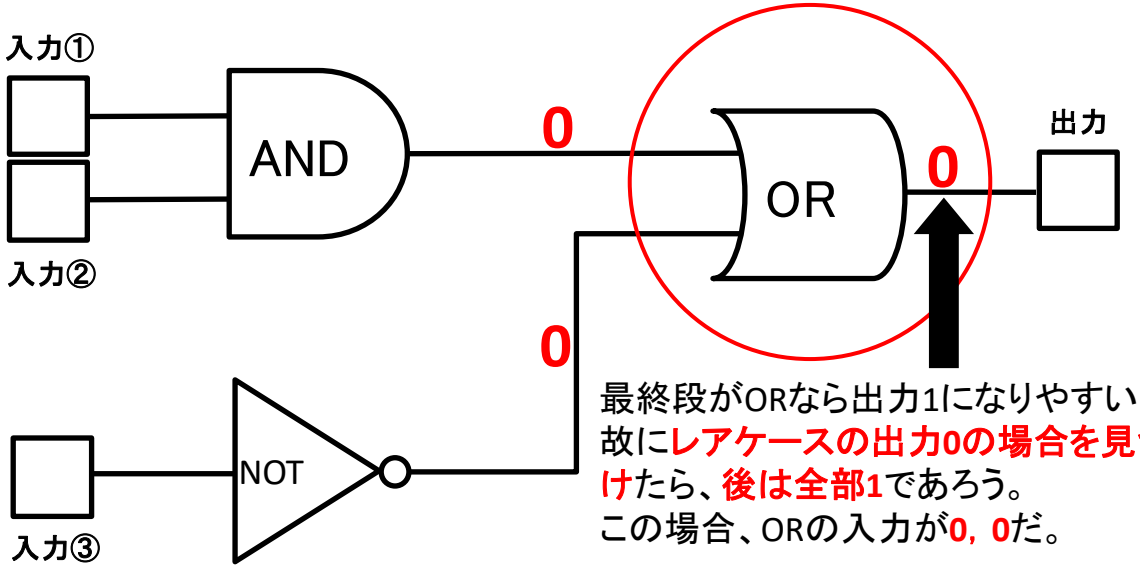


真理値表

①	②	③	出力
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

論理回路問題補足解法技術

ANDが0を出す場合は、その入力が「2入力が1，1である」場合以外
・・・条件x



最終段がORなら出力1になりやすい。
故にレアケースの出力0の場合を見つけたら、後は全部1であろう。
この場合、ORの入力が0，0だ。

NOTが0を出す場合は、その入力が「入力1である」場合
・・・条件y

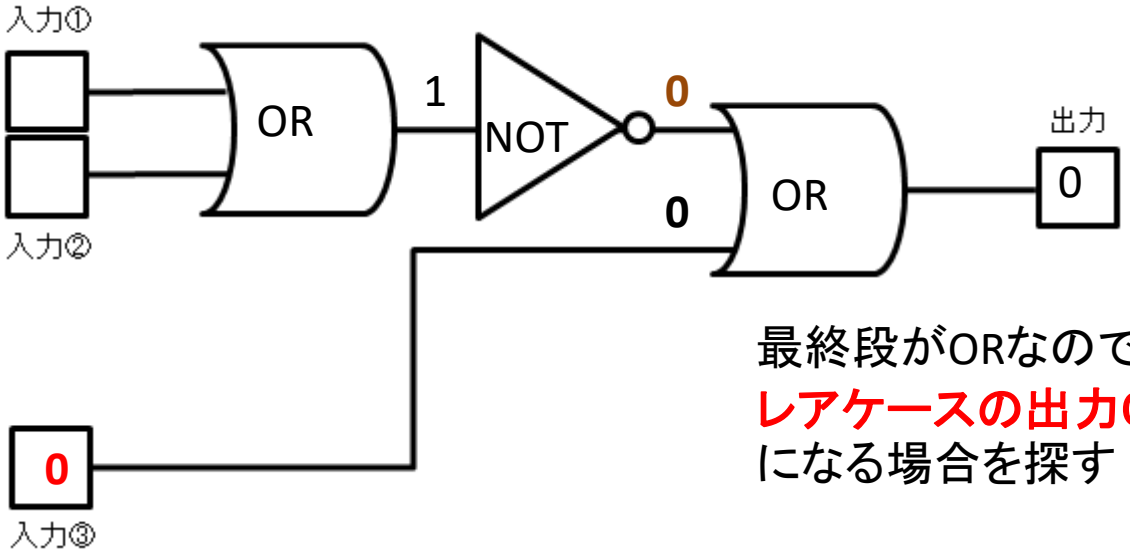
真理値表

①	②	③	出力
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

条件x、条件yが重なる場合は出力0あとは全部1

1-8-4　論理回路練習問題2：右の真理値表を完成させよう

①②が０，０以外は
OR回路の出力は1



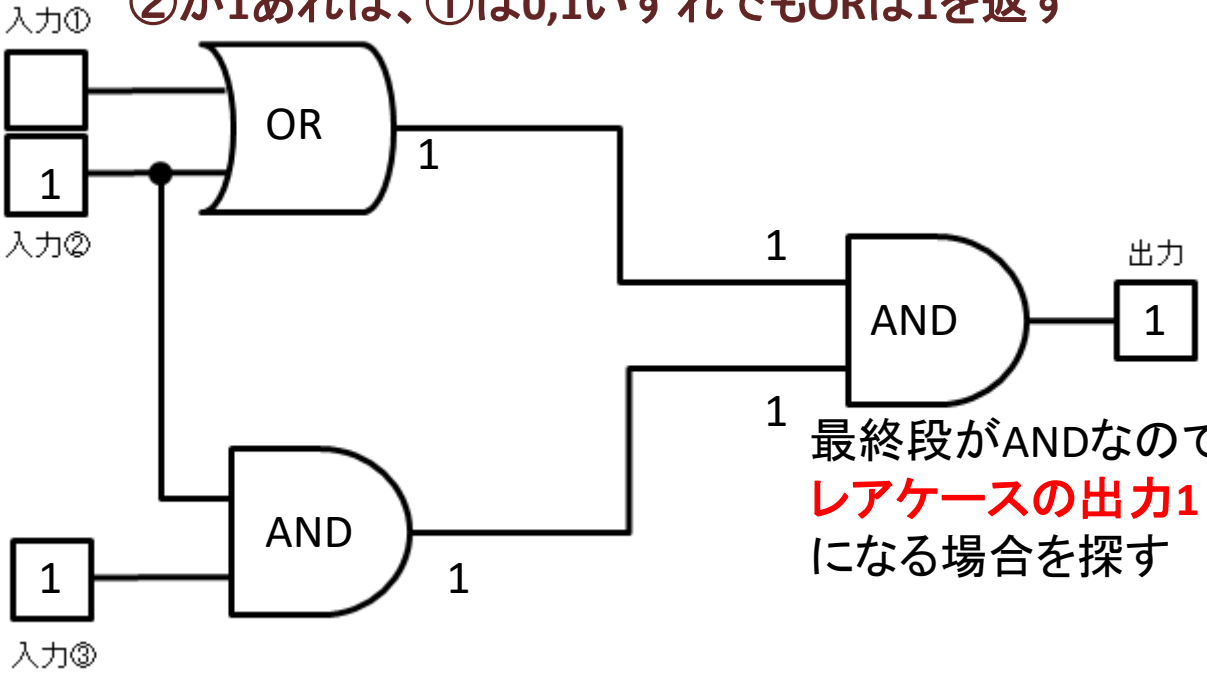
最終段がORなので
レアケースの出力0
になる場合を探す

真理値表

①	②	③	出力
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

1-8-4　論理回路練習問題3：右の真理値表を完成させよう

②が1あれば、①は0,1いずれでもORは1を返す

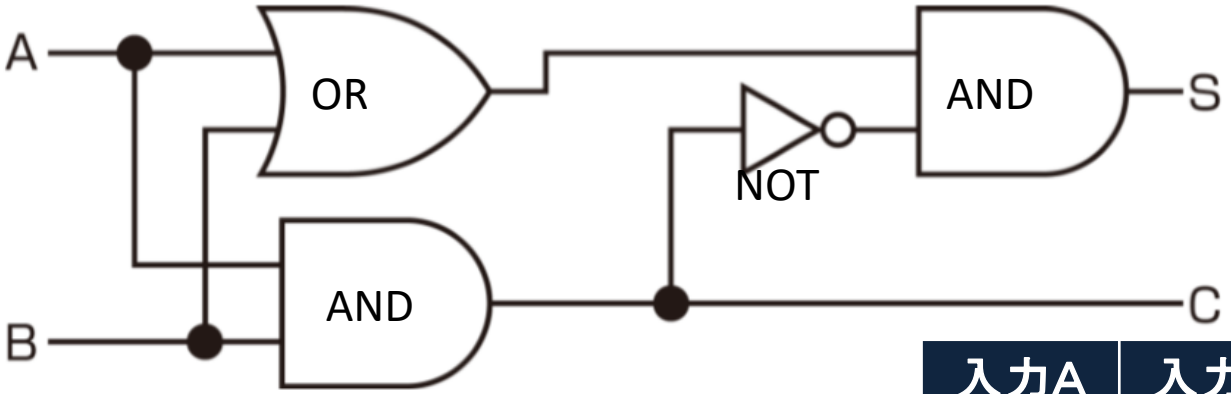


最終段がANDなので
レアケースの出力1
になる場合を探す

真理値表

①	②	③	出力
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

1-8-5　半加算器：右の真理値表を完成させよう
出力Ｓ，Ｃは何を示すか？



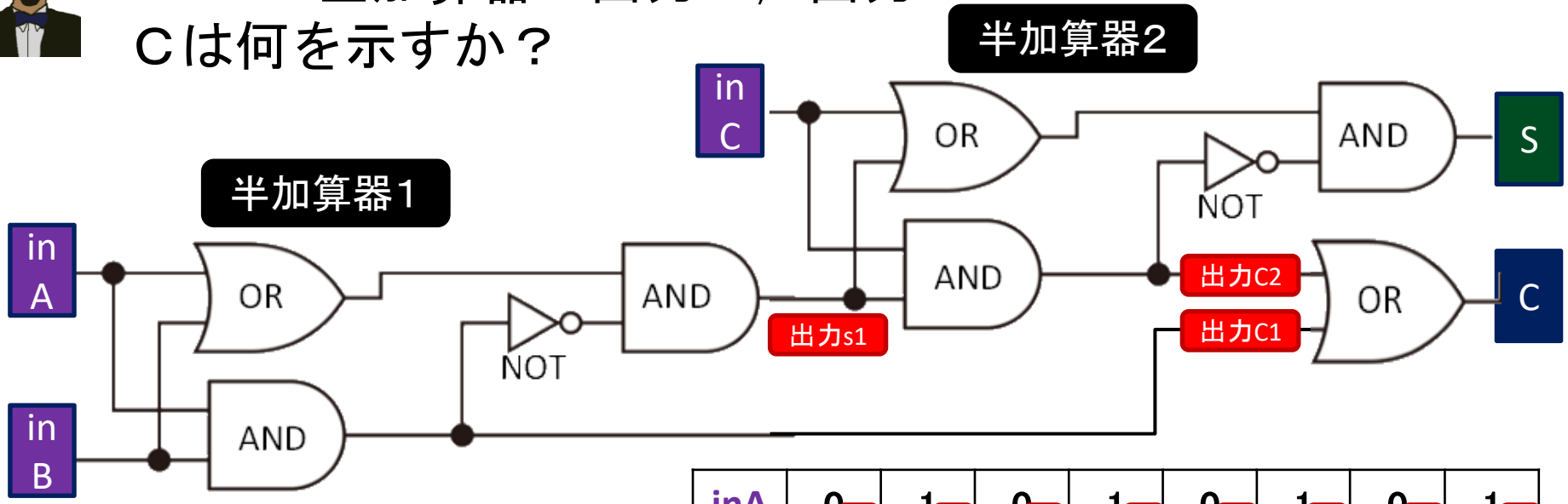
回路中の●は
枝分かれしている
ことを示す

・出力Ｓ：加算結果の１桁目
　出力Ｃ：加算結果の１桁上げの有無

・2bit(2入力)の加算：半加算器で可能
・桁上がり分を含む3bit(3入力)の加算
　➡次の工夫が必要

入力A	入力B	出力S	出力C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

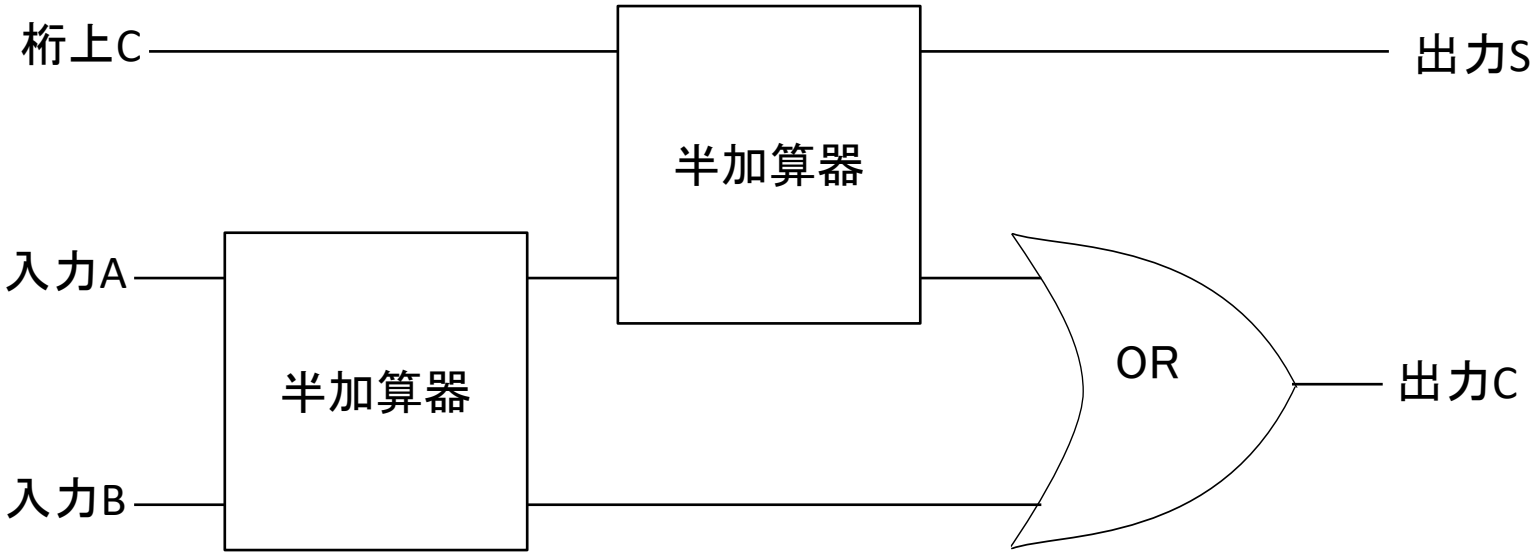
1-8-6　全加算器：出力S，出力Cは何を示すか？

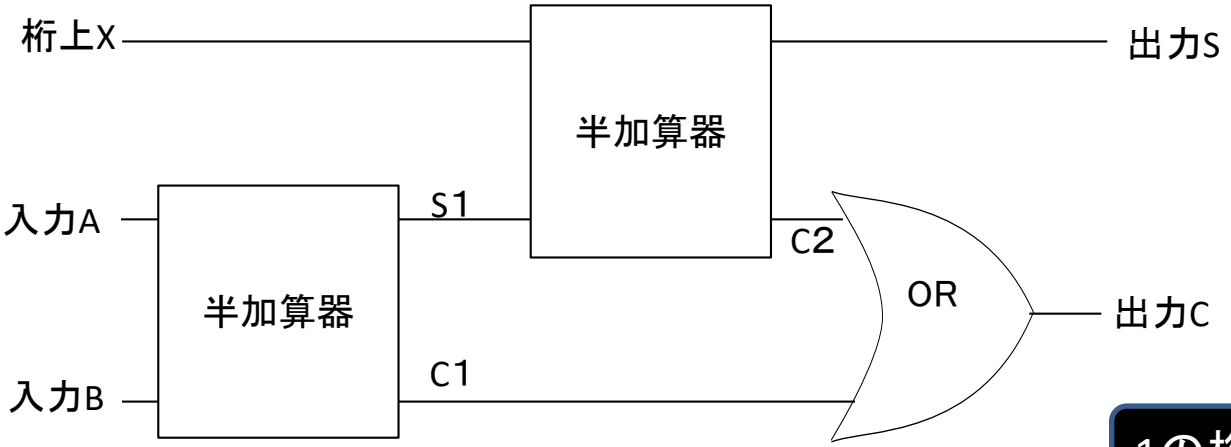


出力S:3入力加算結果の1桁目
出力C: 3入力加算結果の2桁目

・桁上がり分を含む3bit(3入力)の加算:全加算器

inA	0	1	0	1	0	1	0	1
inB	0	0	1	1	0	0	1	1
inC	0	0	0	0	1	1	1	1
S	0	1	1	0	1	0	0	1
C	0	0	0	1	0	1	1	1


桁上C
半加算器
出力S
入力A
半加算器
OR
出力C
入力B

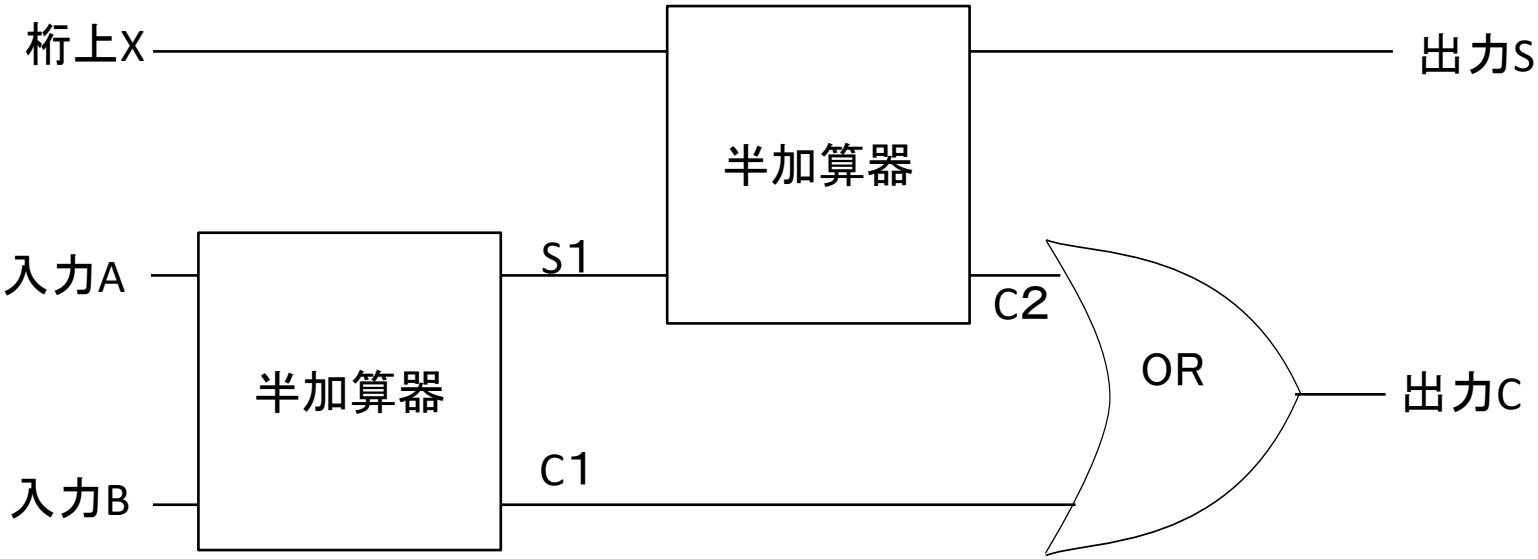


0 0 0
0 0 0
0 0 0
0 0 0
0 0

1の桁　2の桁

桁上X	入力A	入力B	中間S1	中間C1	中間C2	出力S	出力C
0	0	0	0	0			
0	0	1	1	0			
0	1	0	1	0			
0	1	1	0	1			
1	0	0	0	0			
1	0	1	1	0			
1	1	0	1	0			
1	1	1	0	1			

1 1 1
1 1 1
1 1 1
1 1 1


桁上X
半加算器
出力S
入力A
S1
半加算器
C2
OR
出力C
入力B
C1

1-8-6　全加算器・半加算器の必要数　　4bitの場合

	全加算器	全加算器	全加算器	半加算器
	1	1	1	
	0	1	1	1
＋	0	1	0	1
	1	1	0	0

☜桁上がり

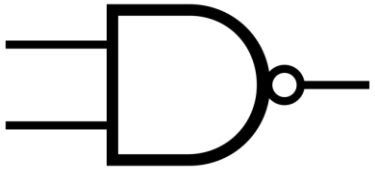
		7
+		5
	1	2

4(bit)の加算➡　3個の全加算器、1個の半加算器が必要

n(bit)の加算➡(n-1)個の全加算器、1個の半加算器が必要

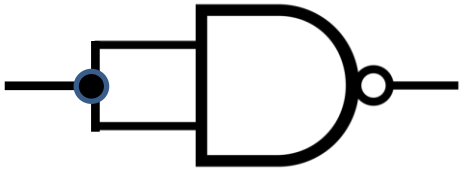
処理するビット数を上げる➡多くの素子を必要とする➡コスト増

1-8-7　ＮＡＮＤ回路：NOT AND、すなわちAND回路の反転

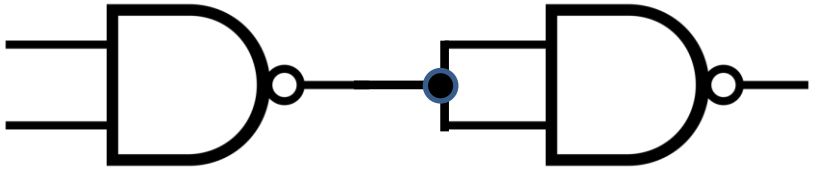


入力①	入力②	出力
0	0	1
0	1	1
1	0	1
1	1	0

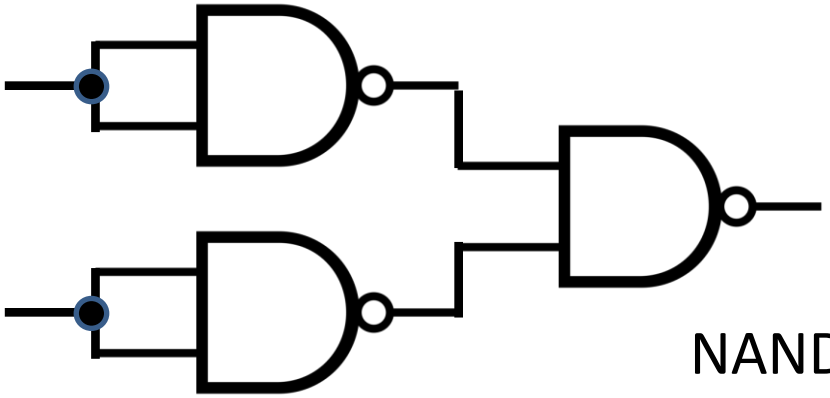
NAND回路の組合せで
あらゆる回路を組むことができる
➡大量生産でコストカット可能



上はNAND回路で組んだNOT回路
では、NAND回路でAND回路、OR回路を
組む方法は？



NAND回路で組んだ　AND回路



NAND回路で組んだ　OR回路