# 四則演算と計算の限界

情報Ⅰ　第44回授業

08コンピュータとプログラミング

対応ファイル：22exp44.xls

# 【復習】論理回路（教p.124）

- 真（=1）と偽（=0）の2通りの値（「真理値」という）だけであらわされる、情報の演算を「論理演算」という。
- CPUは、いくつかの論理演算を行う「論理回路」が組み合わさることで複雑な計算を行っている。
- 論理回路の基本となるのが「ANDゲート」「ORゲート」「NOTゲート」で、この3つの論理ゲートの組み合わせでさまざまな論理回路を構成する。

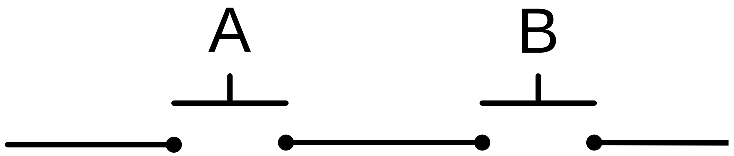
# 【復習】ANDゲート（論理積ゲート）


真理値表

| 入力 | | 出力 |
|---|---|---|
| A | B | F |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

AとBの両方（A and B）が1の場合のみ「1」を出力し、その他の場合は「0」を出力

スイッチの「直列」配置



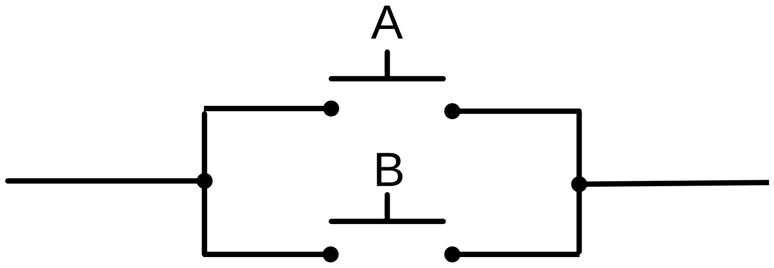
# 【復習】ORゲート（論理和ゲート）


AとBの少なくとも一方（A or B）が1の場合「1」を出力し、両方「0」の場合は「0」を出力

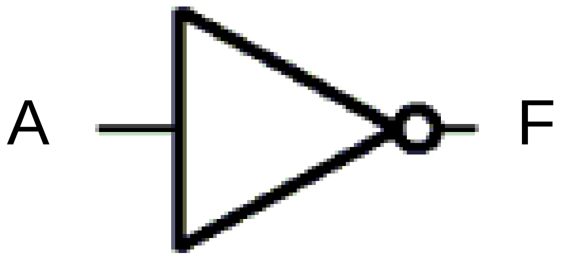
スイッチの「並列」配置

真理値表

| 入力 | | 出力 |
|---|---|---|
| A | B | F |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



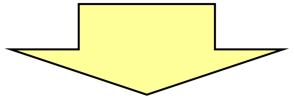
# 【復習】NOTゲート(否定ゲート)



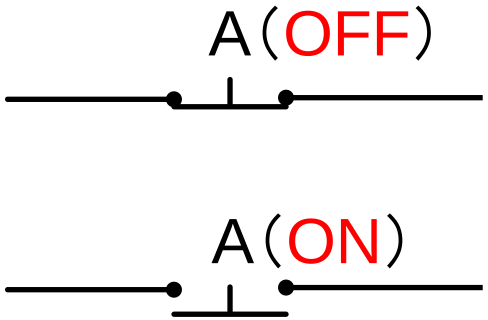

真理値表

| 入力 | 出力 |
|---|---|
| A | F |
| 0 | 1 |
| 1 | 0 |

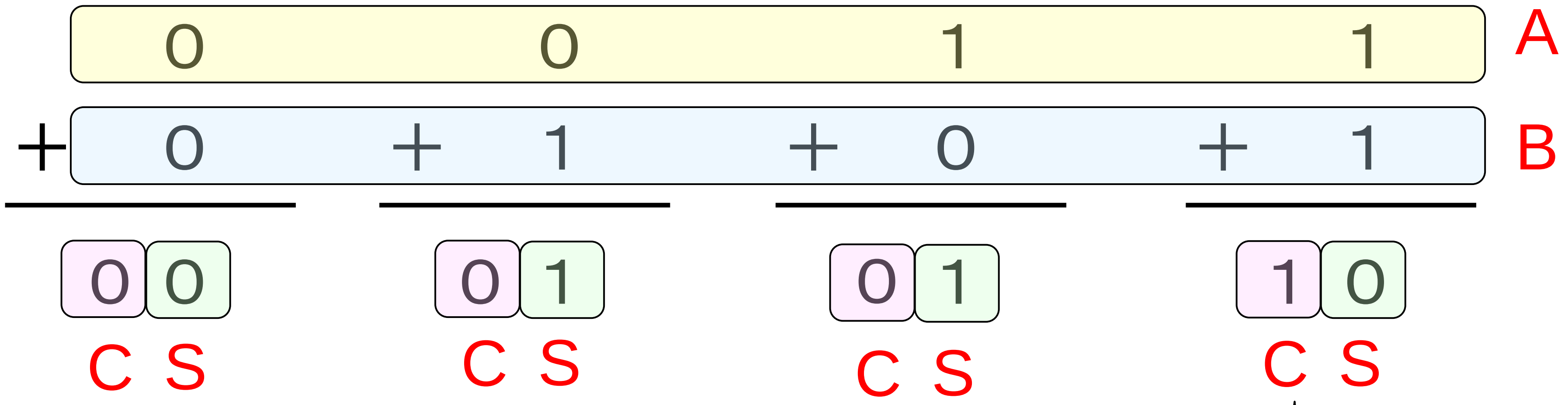
Aが0の場合は「1」を出力し、
1の場合は「0」を出力
(逆の出力(not)を行う)



スイッチを押すと消灯する電球



# 1ケタの2進法の足し算




入出力を整理

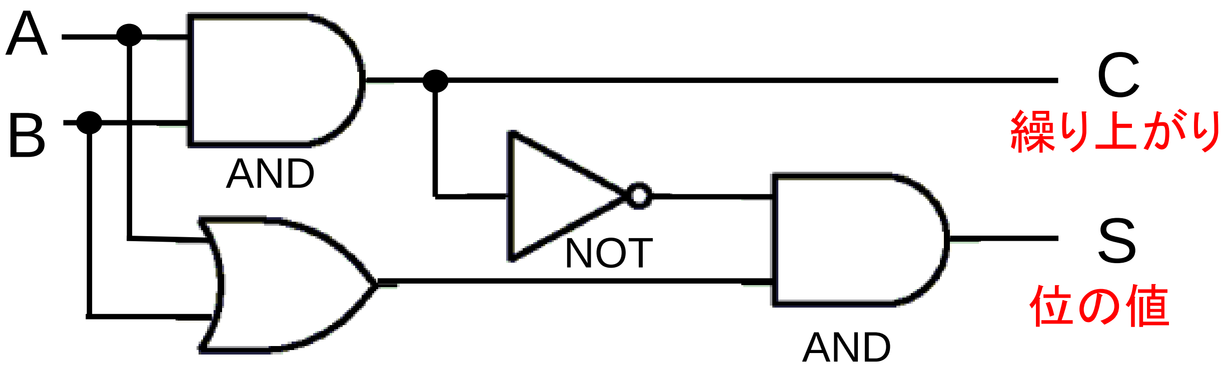
| 入力 | | 出力 | |
|---|---|---|---|
| A | B | C | S |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

繰り上がり

# 【復習】半加算回路（Half Adder）

AND、OR、NOTゲートを組み合わせて、1ケタの２進法の「足し算」を行う回路。

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



# 【復習】2ケタ以上の足し算

$$\begin{array}{r} 7\ 8\ 9 \\ +\quad 5\ 1\ 3 \\ \hline \boxed{1}\ \boxed{1}\ \boxed{1}\ \ \\ 1\ 3\ 0\ 2 \end{array} \qquad \begin{array}{r} 1\ 1\ 0\ 1 \\ +\quad 1\ 0\ 1\ 1 \\ \hline \boxed{1}\ \boxed{1}\ \boxed{1}\ \boxed{1}\ \ \\ 1\ 1\ 0\ 0\ 0 \end{array}$$

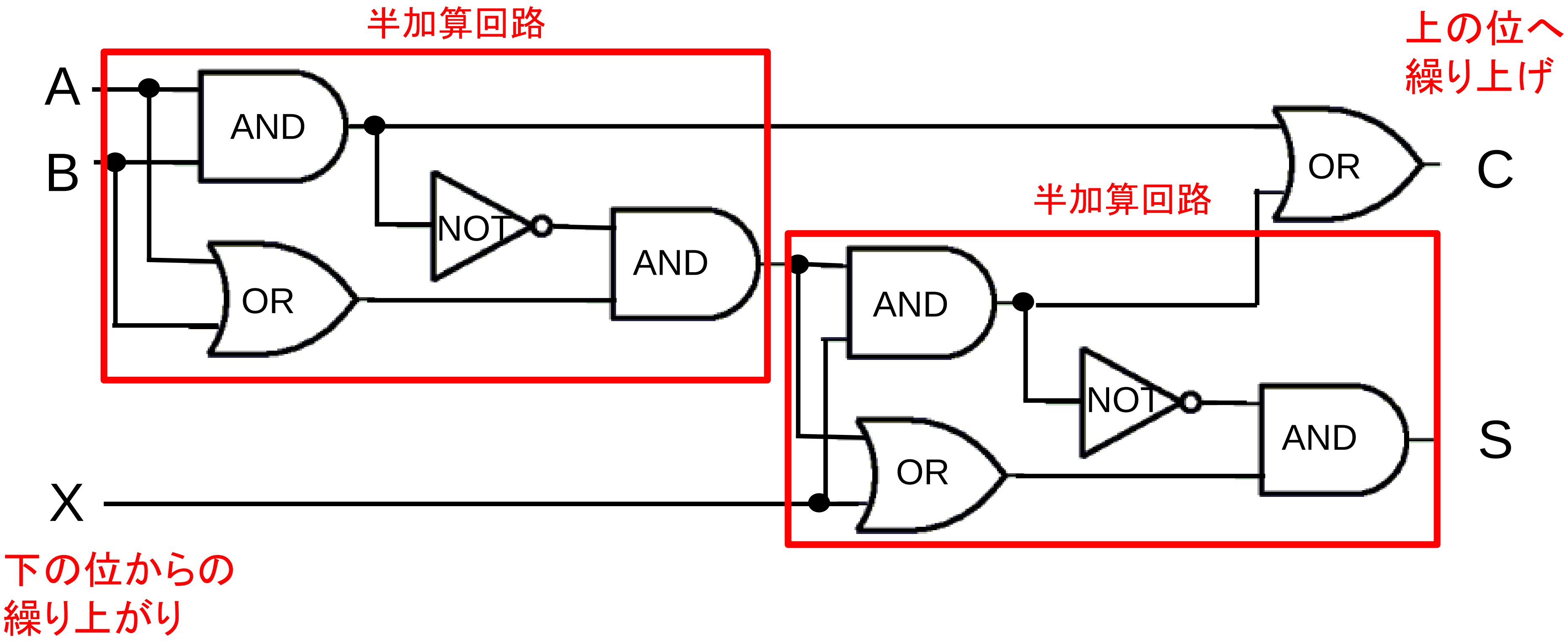
「その位の値」（S）

＝「足される数」＋「足す数」＋「前桁の繰り上り」

（A）　（B）　（X）

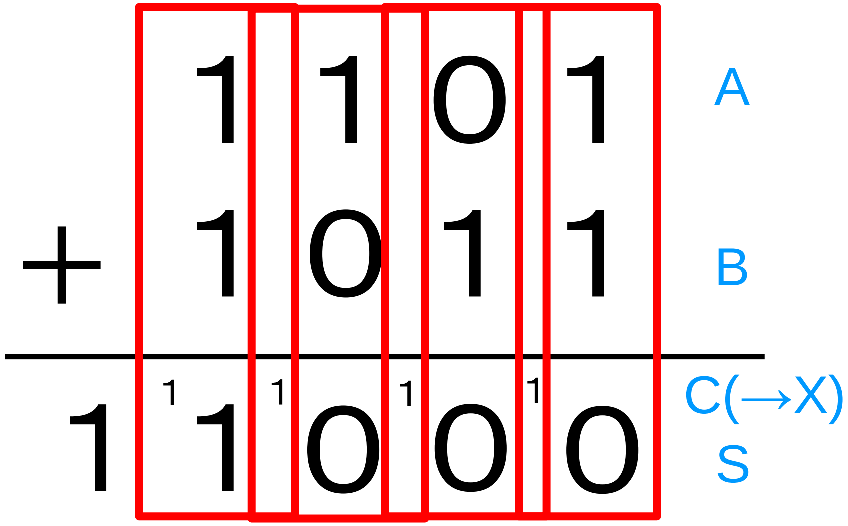
# 【復習】全加算回路（Full Adder）教p.17

下の桁からの「繰り上がり」も取り入れた回路

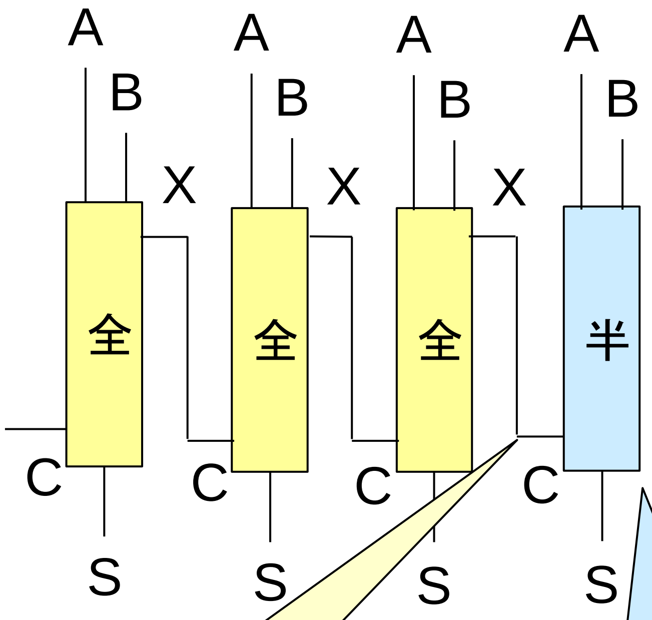



単純な回路の組み合わせで複雑な計算ができるようになる！

# 【復習】加算回路を用いた4桁の計算




「足し算」ができることは分かっている！




繰り上がりのCが、次の全加算機のXの値となっていく

1桁目は前の桁の繰り上がりが無い→半加算機で良い

# コンピュータで四則演算をするには？

「四則演算」とは、
「＋」「－」「×」「÷」の計算のことです。

# コンピュータでの「掛け算」

- 同じ数を掛ける数だけ足す。
  - 例） ３２×５
    - ３2を5回足す
- ２進を2倍する時は桁をずらす。（「Shift」と言うこともある）
  - 例） $156_{(10)} \times 10 = 1560_{(10)}$ （←小数点をずらす）
  - 例） $1011_{(2)} \times 2 = 10110_{(2)}$ （←小数点をずらす）

# 補数（p.126）

補数：「元の数」と「補数」を足した場合に

桁上がりが発生する数のうち「最小」の数

例：（１０進法で）３２７８の補数

→ １００００ － ３２７８ ＝ ６７２２

練習： 次の数の補数を求めてみよう

（１） ７

（２） ２９

# 補数と「引き算」

同じ桁数の「引き算」を行う時は、

・補数を足し、　→　足し算！

・桁上がりを無視する

例：（10進法で）5632 － 3278

→　3278の補数：6722

10000 − 3278

→　5632 ＋ 6722 ＝ 12354

無視！

→　2354

※　5632＋（10000－3278）とも考えられる！

# ２の補数

２進法では、補数（２の補数）は簡単に求まる！

例：101101（２） の補数

→　101101　の0と1を反転　010010

→　これを足せば、111111

→　1000000は、111111　の1つあと

→　よって、010010　＋　１　＝　010011

練習：　次の数の「２の補数」を求めてみよう

（１）　1010

（２）　11111

# ２の補数を利用した「引き算」

例）　101011－1100

001100 のビット反転 110011

よって、２の補数は 110100

桁を
揃える!

110011+1

```
   101011
+) 110100
---------
  1011111
```

無視!

→ 011111

→ 11111

練習：　２の補数を利用し、

100110－1101　を計算しよう

# 練習：　100110－1101

例）　100110－1101

桁を揃える!　001101

ビット反転　110010

1を足す　110011　←　２の補数

100110

+）110011

---

繰り上がりを無視　1011001 → 011001

→ 11001

# 引き算と「割り算」

- 割り算を行うには、
  - 「割られる数」から「割る数」を引けるまで引いていく
  - 引けた回数が商、残った数が余り

例）３５÷９

　　３５－９＝26

　　26―９＝17

　　１７－９＝　８（＜９）　←　３回引けたから、

　　　　　　　　　　　　　　　商：３　余り：８