# 目次

第 4 章

# 論理回路と数学での論理

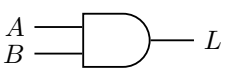
この章では，情報 I で扱う「論理回路」の原理について，数学で扱う論理の考え方（背理法の原理や同値の概念など）と併せて解説していきます。

## 4.1 基本の論理回路と真理表

身の回りの電気機器は，スイッチが押されたり，センサーがある値以上または以下の数値を観測されたりすると，何らかの反応を示すようになっています。スイッチは ON か OFF，センサーは反応するか否かのように 2 つの状態で表現でき，特に **0 か 1** あるいは **TRUE（真）か FALSE（偽）**の 2 値で表すことができます。このように入出力情報を 0 または 1 で表現できるデジタルな電子回路で，以下の 3 つの基本回路（AND，OR，NOT）の組み合わせで実現できるものを**論理回路**といい，その入出力の対応をまとめた表を**真理表（真理値表）**といいます。以下では「0 が FALSE」，「1 が TRUE」として対応させて考え，ここでいう真理値とは「TRUE/FALSE」のことを表します。

ちなみに 0，1 のデータはコンピュータの内部では電圧の高低で表現されます。

### AND 回路（論理積回路）



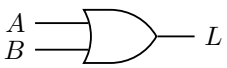

| 入力 | | 出力 |
|---|---|---|
| $A$ | $B$ | $L$ |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

AND 回路は，$A$，$B$ の入力がともに 1（TRUE）のときのみ，出力が 1（TRUE），それ以外は 0（FALSE）とするものです。$A$ と $B$ が両方ともに真であるときのみ，その結果を真とする「$A$ かつ $B$」という論理演算に対応し，「$A \wedge B$」と記します。

例えばパソコンで文字入力をする場面で，キーボードの SHIFT キーと A キーを同時に押すと大文字の「A」が表示できますが，この場合，$A$:「SHIFT キーが押される」，$B$:「A のキー

が押される」，$L$:「大文字の A が表示される」にそれぞれ対応しています[*1]。

### OR 回路（論理和回路）

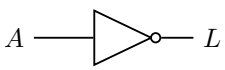



| 入力 | | 出力 |
|---|---|---|
| $A$ | $B$ | $L$ |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

OR 回路は，$A$，$B$ いずれかの少なくとも一方の入力が 1（True）のとき，結果を 1 とするもの，つまり，$A$ か $B$ の少なくとも一方が真のとき，結果を真とする「$A$ または $B$」という論理演算に対応し，「$A \vee B$」で表します。

例えば，人が誰も乗っていない 2 基のエレベータがあり，いずれか一方の「↑」ボタンが押されると「上の階に行くエレベータが止まる」という仕組みを表現することができます。2 つの「↑」ボタンが同時に押されても同じ結果を示します。

### NOT 回路（否定回路）




| 入力 | 出力 |
|---|---|
| $A$ | $L$ |
| 0 | 1 |
| 1 | 0 |

NOT 回路は，入力された 1，0 の情報を 0，1 に反転させる回路です。「$\overline{A}$」で表します。

### 重要な注意：「OR（または）」は「少なくとも一方」であって「どちらか一方」ではない

レストランのランチメニューで「パンまたはライスがつきます」といった場合は通常「どちらか一方のみ（either A or B）」を表しますが，論理の世界での「または」（OR）は「少なくとも一方」の意味になります。「どちらか一方」を表す論理回路として，あとで扱う XOR 回路というものがあります。

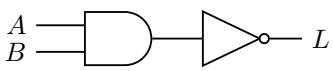
### 注意：論理回路の実装はブラックボックスでよい

AND 回路と OR 回路はそれぞれスイッチの直列回路と並列回路に対応させることができますが，NOT 回路はそうはいきません。実際には例えばトランジスタという素子を複数利用して実現されていて，なかなか複雑なものとなっています。ここではあくまでこれらの基本回路を組み合わせた「論理」に注目して，考えていくことにします。

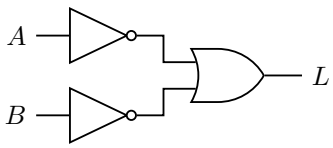
[*1] もし SHIFT キーが押されていないまま，A のキーを入力すると，（大文字ではなく）小文字の a が表示されるという結果になります

**例題 4.1: 基本回路の組み合わせ〜NAND 回路とド・モルガンの法則**

(1) 次の回路は **NAND 回路**とよばれるものである。対応する真理表を作成しなさい。(出力結果は $A \wedge B$ の否定であることから，$\overline{A \wedge B}$ と表す。NAND の N は NOT の意)




(2) (1) と以下に示す回路の 2 つについて，入出力の結果が一致することを真理表で確認し，ド・モルガンの法則 $\overline{A \wedge B} \Leftrightarrow \overline{A} \vee \overline{B}$ が成り立つことを示せ ((2) の出力結果は $\overline{A}$ と $\overline{B}$ を OR 回路で結んでいるので $\overline{A} \vee \overline{B}$ と表す)。

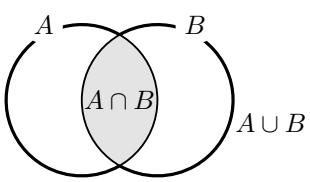



(3) あるタクシーには，2 つの後部座席 $A, B$ のシートベルトが締められているときだけ，「シートベルトを着用してください」のランプ表示が消えるようになっています。(1) (2) の論理回路の入力の $A, B$ は「各席のシートベルトが締められているとき 1，そうでないとき 0」とし，出力 $L$ は「『シートベルトを着用してください』のランプが点いているとき 1，消えているとき 0」とします。この例を用いて，ド・モルガンの法則が成り立っていることを説明せよ。

(考え方) 例えば (1) では，まず $A, B$ の入力がともに 1 のとき，AND 回路で 1 が出力され，NOT 回路で反転されて 0 が出力されます。このように $A, B$ の入力値の組み合わせ 4 通りについて順に調べます。

## 重要な注意：論理と集合の対応とド・モルガンの法則

数学でベン図というものを学びます。これは $A$，$B$ を条件としたとき，$A$，$B$ を満たすものの集合（**真理集合**）を下図のように重なりを持つように記したもので，$A$ と $B$ 両方満たす部分がこの重なりの部分に対応し，$A \cap B$ と書きます。また，$A$ と $B$ の少なくとも一方を満たすものの集合は，$A$，$B$ のいずれかで囲まれる部分に対応し，$A \cup B$ と書きます（論理演算の $\wedge$ と $\vee$ に対応していますが，微妙に異なることに注意)。

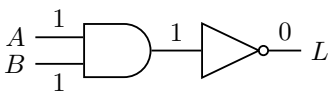



この図を利用すると，$A \cap B$ 以外の部分 $\overline{A \cap B}$ と，$A$ 以外の部分 $\overline{A}$ と $B$ 以外の部分 $\overline{B}$ を合わせた部分 $\overline{A} \cup \overline{B}$ とが一致する，つまり $\overline{A \cap B} = \overline{A} \cup \overline{B}$ が成り立つことがわかります。これと

$\overline{A \cup B} = \overline{A} \cap \overline{B}$ を集合の**ド・モルガンの法則**といいます。例題の（2）はこれの論理演算版に相当します。あとで紹介しますが，論理回路はベン図との対応で考えるとわかりやすくなります。

### 例題 4.1 の解答

（1）例えば $A$，$B$ が 1 のときは，AND 回路の出力が 1 で，NOT 回路を通ると 0 に反転します。それ以外の組み合わせは入力に 0（FALSE）が含まれるので，AND 回路の出力が 0 で，NOT 回路で 1 に反転します。




| 入力 | | 途中 | 出力 |
|---|---|---|---|
| $A$ | $B$ | $A \wedge B$ | $L(\overline{A \wedge B})$ |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

（2）例えば $A$，$B$ がともに 1 のとき，OR 回路に 2 つとも 0 が入力されて 0 が出力されます。$A$ が 1 で $B$ が 0 のときは，OR 回路に 0 と 1 が入力されて，1 が出力されます。他同様，$A$，$B$ がいずれか 0 のとき，OR 回路に 1（TRUE）が少なくとも 1 つ入力されるので，出力は 1 になります。

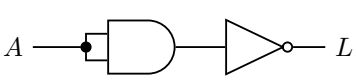
| 入力 | | 途中 | | 出力 |
|---|---|---|---|---|
| $A$ | $B$ | $\overline{A}$ | $\overline{B}$ | $L(\overline{A} \vee \overline{B})$ |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |

以上から，真理表の結果が完全に一致し，ド・モルガンの法則 $\overline{A \wedge B} \Leftrightarrow \overline{A} \vee \overline{B}$ が成り立つことがわかります。

（3）（1）は「A，B のシートベルトがともに締められている」以外のとき，『シートベルト着用』の表示が点灯されることを表します。つまり「A が締められていないか B が締められていないかの一方が少なくとも成り立つ」とき，『シートベルト着用』の表示が出ることになり，これは（2）の回路の説明に合致することから，ド・モルガンの法則が成り立つことがわかります。

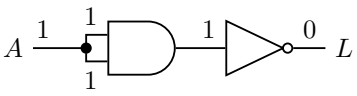
### 回路の分岐と NAND 回路・XOR 回路

例題 4.1（1）の NAND 回路を利用した次の回路を考えます。AND 回路の手前で入力 $A$ が 2 つに分岐しています。




これは $A$ の入力値が，黒丸の結節点で分岐して，AND 回路の 2 つの入力値となることを表

します。つまり $A$ の入力値が 1 のとき，AND 回路の入力値は 2 つとも 1 となって 1 が出力，NOT 回路で反転して 0 が出力されます。また $A$ の入力値が 0 のとき，AND 回路の入力値は 2 つとも 0 となって 0 が出力，NOT 回路で反転して 1 が出力されます。

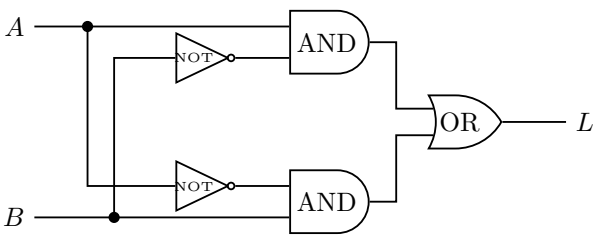



| $A$ | $L$ |
|---|---|
| 1 | 0 |
| 0 | 1 |

つまり，この入出力結果は NOT 回路に一致することがわかります。

この回路で最初に分岐したあとは NAND 回路を表しているので，NOT 回路が NAND 回路だけで表せたことになります。あとで紹介しますが，AND 回路も OR 回路も NAND 回路だけで表すことができ，実際の論理回路は NAND 回路のみで表すことができます。

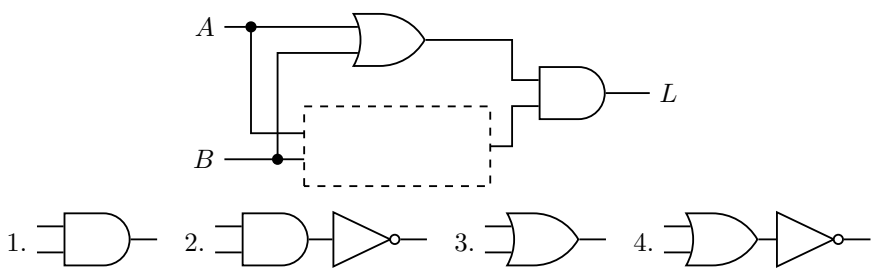
**例題 4.2: 回路の分岐・XOR 回路**




ある部屋の照明は同じ部屋の 2 つのボタン A，B で制御できるようになっていて，入出力の結果は上の論理回路で表されます。ここで入力 $A$，$B$ は「各ボタンが ON のときを 1，OFF のとき 0」を表し，出力 $L$ は「照明がついているとき 1，消えているとき 0」を表します。ただし，回路の分岐は図の黒点で示された 2 点のみとします（この回路を **XOR 回路**といいます）。

（1）入出力結果を示す真理表を作成し，照明の制御を例に論理回路の意味を解釈せよ。

（2）XOR 回路は次の図に示す回路と同じ入出力結果になります。空欄にあてはまる回路として適切なものをあとの 1〜4 の中から一つ選びなさい。




**例題 4.2 の考え方**

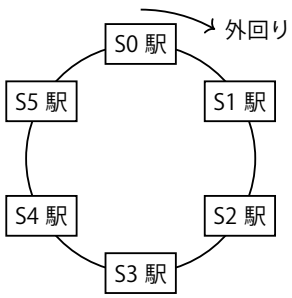
（1）例えば $A$，$B$ が 1（スイッチが ON）のときは，上の AND 回路には $A$ から 1 が入力，$B$ からは手前の NOT 回路で反転された 0 が入力される結果として，0 が出力されます。同様に

(8) 製品 A，B，C の個数を入れる配列 num_abc について，「num_abc = 0」を記述する場所として最も適切な位置を①～④の中から一つ選びなさい。

(9) 空欄 ソ ～ チ にあてはまる内容をそれぞれ答えなさい。

## 問題 11.5（環状線の鉄道運賃表の作成・配列と関数の利用）

ある鉄道路線は環状線となっていて，S0～S5 までの 6 つの駅があります。駅間の距離は S0-S1，S1-S2，…，S5-S0 駅間の順に，配列 dist_list = [2,3,2,4,3,2] に格納されています（単位は km）。




この路線の 2 駅 p，q 間の，外回り（outer）と内回り（inner）の距離が短いほうについて，移動距離を算出する min_dist 関数を定義します。

さらに，この鉄道会社の運賃は 3 km 未満の移動が 150 円で，3 km 以上は 2 km ごとに 30 円ずつ加算して計算されます。

この路線の 2 駅間について，運賃表 fare_list を作成することにします。

**問 1**　この環状路線一周の総移動距離 loop_dist を以下のコードで求めます。例えば S0-S1 間の距離は dist_list[0] = 2，S0-S5 駅間は dist_list[5] で取り出します。range(a,b) は「a 以上 b 未満」，len(a) は「配列 a の要素数」を表します。

```
# 駅間の距離(外回り)
dist_label = ['0-1','1-2','2-3','3-4','4-5','5-0']
dist_list = [2,3,2,4,3,2]

# 1周の距離
loop_dist = [ ア ]
for i in range(0,[ イ ]):
  loop_dist = [ ウ ] + [ エ ][i]
print(loop_dist)
```

(1) 空欄 ア にあてはまる内容を答え，空欄 イ ～ エ にあてはまる内容を以下の解答群から 1 つずつ選びなさい（同じものを複数回選んでも構わない）。

イ ～ エ の解答群

1. dist_label　2. dist_list　3. loop_dist　4. len(dist_list)
5. len(dist_list)-1　6. len(dist_list)+1

**問 2**　次に Sp 駅と Sq 駅（p, q は 0～5 のいずれかの整数）間の外回りの移動距離（outer_dist）と，内回り（inner_dist）の移動距離の小さいほうを求める関数 min_dist を次のコードで定義します。

```
# Sp駅とSq駅間の,内回り外回りの近いほうの距離を求める
def min_dist(dist_list,p,q):
  loop_dist = [ア]    # さきほどの1周の距離を求めるコードを転記
  for i in range(0,[イ]):
    loop_dist = [ウ] + [エ][i]

  if p > q:      # p > qのとき，p,qの値を入れ替える
    [オ]

  outer_dist = 0
  for i in range(p, [カ]):
    outer_dist = [キ] + [エ][i]
  inner_dist = [ク] - [ケ]
  if outer_dist >= inner_dist:
    return [コ]
  else:
    return [サ]
```

(2) [オ] の 3 行で，次の（a）～（c）を並べ替えて順に実行します。正しい順番を示したものをあとの 1～6 の中から 1 つ選びなさい。

（a）`p = q`　（b）`q = temp`　（c）`temp = p`

1.（a）（b）（c）　2.（a）（c）（b）　3.（b）（a）（c）　4.（b）（c）（a）
5.（c）（a）（b）　6.（c）（b）（a）

(3) 空欄 [カ]～[サ] にあてはまる内容を，以下の解答群から 1 つずつ選びなさい（同じものを複数回選んでも構わない）。

[カ]～[サ] の解答群

1. `loop_dist`　2. `dist_list`　3. `inner_dist`　4. `outer_dist`
5. `len(dist_list)`　6. `len(dist_list)-1`　7. `len(dist_list)+1`　8. `q`
9. `q-1`　10. `q+1`

(4) 次のコードの実行結果を答えなさい。

```
print(min_dist(dist_list,5,1))
```

**問 3**　次に，電車での移動距離が `dist` [km] のときの運賃を求める関数 `fare` を以下のコードで定義します。例えば 3 km 以上 5 km 未満のときは 180 円となります。