

コンピュータ・サイエンス2

第1回

CPUの構成と論理回路

人間科学科コミュニケーション専攻

白銀 純子

第1回の内容

- オリエンテーション
- 前期の復習
 - ◆ 後期の内容に関係ある部分のみ
- コンピュータの構成

オリエンテーション

授業目標

- コンピュータの基本的な仕組みを理解
 - ◆ コンピュータを自在に使いこなすための基礎的な素養
 - ◆ 情報処理技術者試験や高校の教科「情報」の教職免許に必要な知識を身につけるための基盤

学習上の注意事項

- 疑問点やわからないことをそのままにしておくと、ついてこれなくなる
 - ◆ 必ず次の講義までに解決するように!
- 授業を休んだときは、次に授業までに、必ず授業のWebページを見て内容を勉強しておくこと
 - ◆ わからないことは聞くこと!

教科書

- 教科書:「情報とコンピュータ」, 河村一樹, 和田勉, 山下和之, 立田ルミ, 岡田正, 佐々木整, 山口和紀共著, 株式会社オーム社

※授業は、教科書±αの内容になる予定なので、
教科書と授業の資料を併用して勉強すること

連絡先と資料置き場

- 連絡先

研究室: 8号館4階8413室

メールアドレス: junko@lab.twcu.ac.jp

※質問は、メールか研究室にどうぞ

- 授業Webページ

<http://www.cis.twcu.ac.jp/~junko/Science/>

※授業内容、お知らせなど

わからない言葉があるときは

- コンピュータ用語辞典で調べる
 - ◆ e-Words: <http://e-words.jp/>
 - ◆ アスキーデジタル用語辞典: <http://yougo.ascii24.com/gh/index.html>
- コンピュータ用語辞典に載っていないときは、検索エンジン(Googleなど)で調べる
 - ◆ Google: <http://www.google.co.jp/>

成績評価とレポート・試験

- 成績評価：出席：30%，レポート+期末試験：70%
- 出席：授業のWebページから毎回の設問に回答

出席の取り方(1)

- スマートフォンで授業のWebページから回答入力ページにアクセスし、前回授業の復習・その日の授業に関する問題やアンケートに回答
- 問題やアンケート: 1回の授業につき2～3回程度、授業内で発表
 - ◆ その日に出されたものすべてに回答すると1回分の出席
 - ◆ 無回答や明らかにいいかげんな回答があると、遅刻・寝ていた・早退のどれかとみなし、1回分の出席点の半分
 - どうしても回答がわからない設問は、設問の内容を書いておけばOK
 - 設問に対する回答欄の数が多い日もあるかもしれないが、その場合には、出されなかった問題については無回答でOK(解答欄は3つあるが、その日の設問は2つだけだった、など)
 - ◆ 電車の遅延の場合には、遅延証明書をもらってくること
 - 遅延証明書の裏に授業名・日付・学生番号・氏名を書いて提出することで、無回答の設問があっても、1回分の出席点
 - バスは遅延証明書が出ないので要注意(遅れても大丈夫なように余裕を持って来ること)

出席の取り方(2)

- 設問は一気には出さないで、設問に対する回答は、ノートなどにメモしておき、授業の最後に入力して提出すること
 - ◆ 1問ずつ入力して提出、ということをしてしないこと
 - すべての設問に対する回答を入力した状態で提出すること
 - ◆ 回答を、回答提出フォームに入力しておいておくと、何らかの原因で回答が消えてしまうこともありえるので、必ず回答をノートなどにメモしておくこと
 - タイムアウト(一定時間ほっておくと時間切れになる)
 - 間違って回答提出フォームの画面を消してしまう, etc.
- 何らかの理由で回答提出フォームから回答できない人は、授業の最後に出席カードを取りに来ること(授業中に取りに来ないこと)
 - ◆ 出席カードの裏に解答を書いて提出すること

授業のWebページ: <http://www.cis.twcu.ac.jp/~junko/Science/>

前期の復習

コンピュータでの情報の扱い方(p. 2)

- コンピュータが扱える情報は「0」と「1」のみ
- 大量の「0」と「1」を組み合わせて情報を表現
 - ◆ 数値: 1つの数を「0」と「1」の組み合わせで表現
 - ◆ 文字: 1つの文字を「0」と「1」の組み合わせで表現
 - ◆ 画像: 1つの点の色を「0」と「1」の組み合わせで表現
 - 画像は、色のついた点が縦横に並べられているものという扱い

2進数(p. 4)

- n進数: 数をn個の文字で表す方法
 - ◆ 10進数: 数を10個の文字で表す方法(普段使っている数の表現方法)
 - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9の10個の文字
 - ◆ 2進数: 数を2個の文字で表す方法
 - 0, 1の2個の文字

コンピュータ: 「0」と「1」で全ての情報を表現

 2進数で情報を表現する、とすることができる

10進数を2進数に変換

- 10進数の数は、2進数の表現に直すことができる

$$\begin{array}{r} 2 \overline{) 10\text{進数の数}} \\ 2 \overline{) \quad \quad \quad \text{商1} \cdots \text{余り1}} \\ 2 \overline{) \quad \quad \quad \text{商2} \cdots \text{余り2}} \\ \quad \cdot \\ \quad \cdot \\ \quad \cdot \\ 2 \overline{) \quad \quad \quad \cdot} \\ \quad \quad \quad \text{商n} \cdots \text{余りn} \end{array}$$

1. 10進数の数を2で割って商1と余り1を計算する
2. 商1を2で割って商2と余り2を計算する
3. 商2を2で割って商3と余り3を計算する
4.

商が0になるまで繰り返す
※小数の計算はしない

➡ 余りを余りnから余り1の順に左から並べたものが2進数

10進数を2進数に変換(例)

10進数の13を2進数に変換

$$\begin{array}{r} 2 \overline{)13} \\ 2 \overline{)6} \cdots \text{余り: } 1 \\ 2 \overline{)3} \cdots \text{余り: } 0 \\ 2 \overline{)1} \cdots \text{余り: } 1 \\ 0 \cdots \text{余り: } 1 \end{array}$$



$$(13)_{10} = (1101)_2$$

10進数の50を2進数に変換

$$\begin{array}{r} 2 \overline{)50} \\ 2 \overline{)25} \cdots \text{余り: } 0 \\ 2 \overline{)12} \cdots \text{余り: } 1 \\ 2 \overline{)6} \cdots \text{余り: } 0 \\ 2 \overline{)3} \cdots \text{余り: } 0 \\ 2 \overline{)1} \cdots \text{余り: } 1 \\ 0 \cdots \text{余り: } 1 \end{array}$$



$$(50)_{10} = (110010)_2$$

※矢印の方向に余りを並べる

2進数を10進数に変換

● 単純に...

1. 2進数の各桁の上にそれぞれ「2」を書く
2. 1. で書いた「2」の右肩に、右から0, 1, 2, ...と書いていく
 - $2^0, 2^1, 2^2, \dots$ ができていく

1.

2	2	2	2	2	2
1	1	1	0	1	0



2.

⁵ 2	⁴ 2	³ 2	² 2	¹ 2	⁰ 2
1	1	1	0	1	0

右から左に、0, 1, 2, ...と
番号をつける



2進数を10進数に変換

● 単純に...

3. 各桁の上の「 2^n 」と、それぞれの桁の数をかけあわせる
4. 2. の結果を足し合わせる

2.

2^5	2^4	2^3	2^2	2^1	2^0
1	1	1	0	1	0



3.

2^5	2^4	2^3	2^2	2^1	2^0
×	×	×	×	×	×
1	1	1	0	1	0

||

2^5 2^4 2^3 0 2^1 0



4.

2^5	2^4	2^3	0	2^1	0
-------	-------	-------	---	-------	---

足し合わせる
||

$2^5 + 2^4 + 2^3 + 0 + 2^1 + 0 = 58$

負の数の表現(p. 9)

● 負の数の表現方法

- ◆ **真数表現**: 2進数の一番大きな桁を符号の桁とし、この桁が0であれば正の数、1であれば負の数とする考え方
 - 「+0」と「-0」ができるなど様々な不都合
- ◆ **2の補数表現**: 負の数 X を、正の数 X (2進数)の0と1を反転させて1を加えた数で表現する方法
 - 足し算・引き算・かけ算・割り算を全て同じ回路で計算できるので好都合

実際のコンピュータでは2の補数を利用

2の補数表現[2](p. 9)

- 2の補数 = 負の数を2進数で表現したもの(コンピュータの世界では)
- 計算方法(例: -20を10桁の2進数に直す)
 1. 2の補数に直したい10進数のマイナスを取り除く
 - $(-20)_{10} \rightarrow (20)_{10}$
 2. 1. の結果を2進数に直す
 - $(20)_{10} = (0000010100)_2$
 3. 2. の結果の0と1を逆にする(0の桁を1、1の桁を0にする)

0 0 0 0 0 1 0 1 0 0



1 1 1 1 1 0 1 0 1 1

2の補数表現[2](p. 9)

- 2の補数 = 負の数を2進数で表現したもの(コンピュータの世界では)
- 計算方法(例: -20を10桁の2進数に直す)

4. 3. の結果に1を足し算する

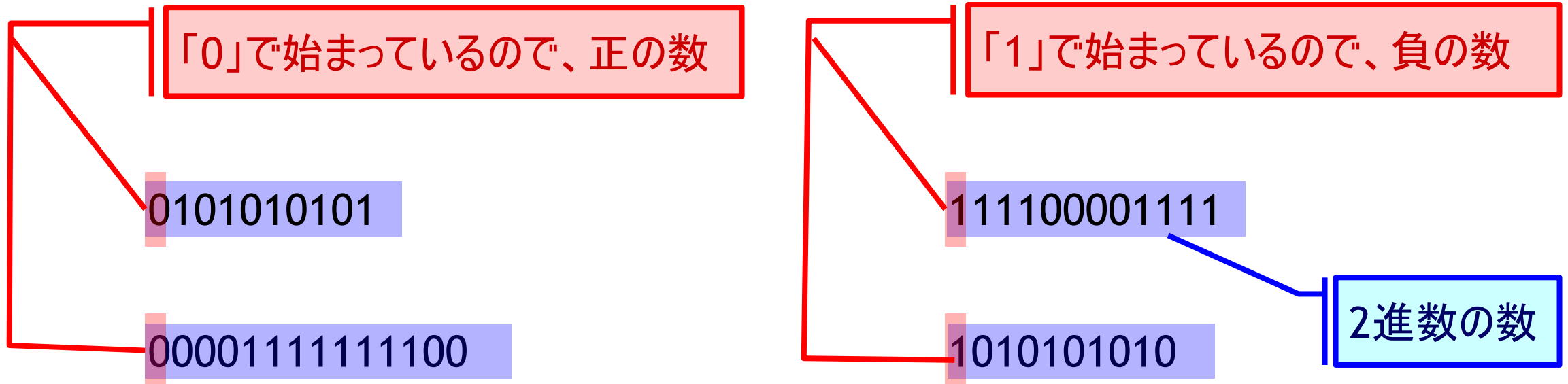
$$\begin{array}{r} 1111101011 \\ +) \quad \quad \quad 1 \\ \hline 1111101100 \end{array}$$

-20を2進数に直した結果
(2の補数 = 2進数での負の数の表現)

2進数での負の数の表現では、
「-」の記号はつけない

正の数と負の数の見分け方[3]

- 2進数を見たときに...(2の補数を考える場合)
 - ◆ 「2の補数を考える」という場合は、先頭の桁を見て、正の数か負の数かを判断
 - ◆ 「2の補数を考える」と書かれていない場合は、負の数を考えなくてOK



コンピュータの構成

コンピュータの構成(p. 39)

- ハードウェアとソフトウェアで構成
 - ◆ **ハードウェア**: 部品や周辺機器など
 - ◆ **ソフトウェア**: ハードウェアを制御して様々な処理をする手順や命令の集合

ハードウェアの構成(p.39)

- マザーボード
- 中央処理装置(CPU)
- 記憶装置
- 入出力装置
- ネットワーク接続装置
- 拡張カード
- 各種インタフェース

マザーボード(p. 34)

- 「メインボード」とも
- コンピュータの様々な部品を装着する基盤
 - ◆ コンピュータのほとんどの部品はマザーボードに接続され、マザーボードを介してやりとりする
 - ◆ 様々なスロット(差込口)を持つ
 - CPUスロット: CPUを装着する箇所
 - メモリスロット: メインメモリを装着する箇所
 - 拡張スロット: 拡張カードを装着する箇所
 - ◆ ビデオカードやサウンドカード、ネットワークカードなどの拡張カードの機能をあわせ持つものも多い

中央処理装置(p. 39)

- 「CPU(Central Processing Unit)」, 「プロセッサ」とも
- コンピュータの心臓部
- 様々なデータの処理や各装置の制御を担当
- コンピュータの速度の性能の大部分を決定付ける部品
 - ◆ コンピュータの処理速度はCPUの処理速度に大きく依存
- 人間の頭脳の中の物事を考える部分に相当

記憶装置[メインメモリ](p. 41)

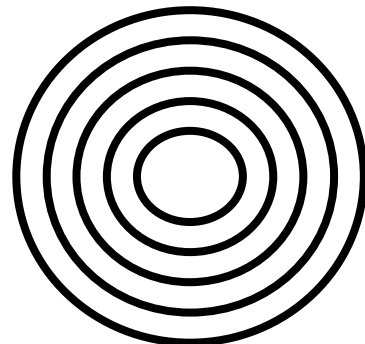
- 「主記憶装置」とも
- コンピュータ内でデータや処理内容を記憶する装置
- CPUから直接読み書きでき、他の記憶装置と比べるとデータの読み書きが非常に高速
 - ◆ ランダムアクセス
- 材料の価格が高く、多くの容量の搭載は不可能
 - ◆ 容量が多いと、それだけコンピュータの処理速度が高速
 - ◆ 最近のPCでは、2GB～8GB程度搭載
- 電源を切ると、記憶した内容が消去
 - ◆ 人間の頭脳の短期記憶の部分に相当

記憶装置[HDD][1](p. 42)

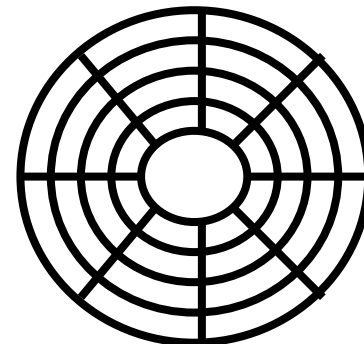
- Hard Disk Driveの略
- コンピュータの代表的な外部記憶装置の1つ
 - ◆ 主記憶装置以外の記憶装置を「外部記憶装置」または「補助記憶装置」と呼ぶ
- 円盤(複数枚)にデータを記憶する装置
 - ◆ 円盤は磁性体のディスク
- 記憶できる容量が大
 - ◆ コンピュータの記憶容量の性能を決定付ける部品
 - ◆ 材料の価格が安く、多くの容量の搭載が可能
- ランダムアクセス(Random access)の記憶装置

記憶装置[HDD][2](p. 42)

- コンピュータの記憶容量の性能を決定付ける部品
- 電源を切っても記憶した内容は記憶したまま
 - ◆ 人間の頭脳の長期記憶の部分に相当
- 振動や熱が弱点
 - ◆ 落としたりすると壊れる
- ディスクをトラックとセクタに区切り、セクタ単位でデータを保存



トラックに区切ったHDD



セクタに区切ったHDD

記憶装置[HDD][3](p. 42)

- ディスクが回転することで、データを読み書き
 - ◆ 回転して、読み書きしたいセクタをアクセスアームの位置にあわせ、アクセスアームが読み書き
 - 1分間に5000～10000回ほど回転
 - ◆ 目的のセクタにたどり着く(シーク)までに時間がかかる
 - HDDは円盤で一番内側がデータの最初
= 外側に保存されたデータのアクセスには時間がかかる
 - 目的のセクタにたどり着くまでの平均時間: **平均シーク時間**
 - ◆ アクセスアームが読み取ったデータをHDD内の一時保存場所に転送し、そこからメインメモリに転送
 - シーク時間とこれらの転送時間を合わせが時間が**アクセス時間**

記憶装置[SSD][1](p. 43)

- Solid State Driveの略
- 近年普及してきた、HDDに代わる勢いの外部記憶装置
- 半導体メモリを利用した記憶装置
 - ◆ USBメモリやデジカメのメモリカードなどで利用
- HDDより高速にデータを読み書き可能
- HDDよりもランダムアクセスの性能が良

記憶装置[SSD](p. 43)

- 電源を切っても記憶した内容は記憶したまま
 - ◆ 人間の頭脳の長期記憶の部分に相当
- 消費電力が少なく、振動にも耐性
- 材料の価格が高
 - ◆ メインメモリより安く、HDDより高い

ランダムアクセス

- 記憶媒体に記憶されている順番に関係なく、データを読み書きしていく方法

◆ 最初の方にあるデータデータも最後の方にあるデータも、発見に必要な時間は同じ

様々なデータの読み書きに必要な平均の時間が少ない

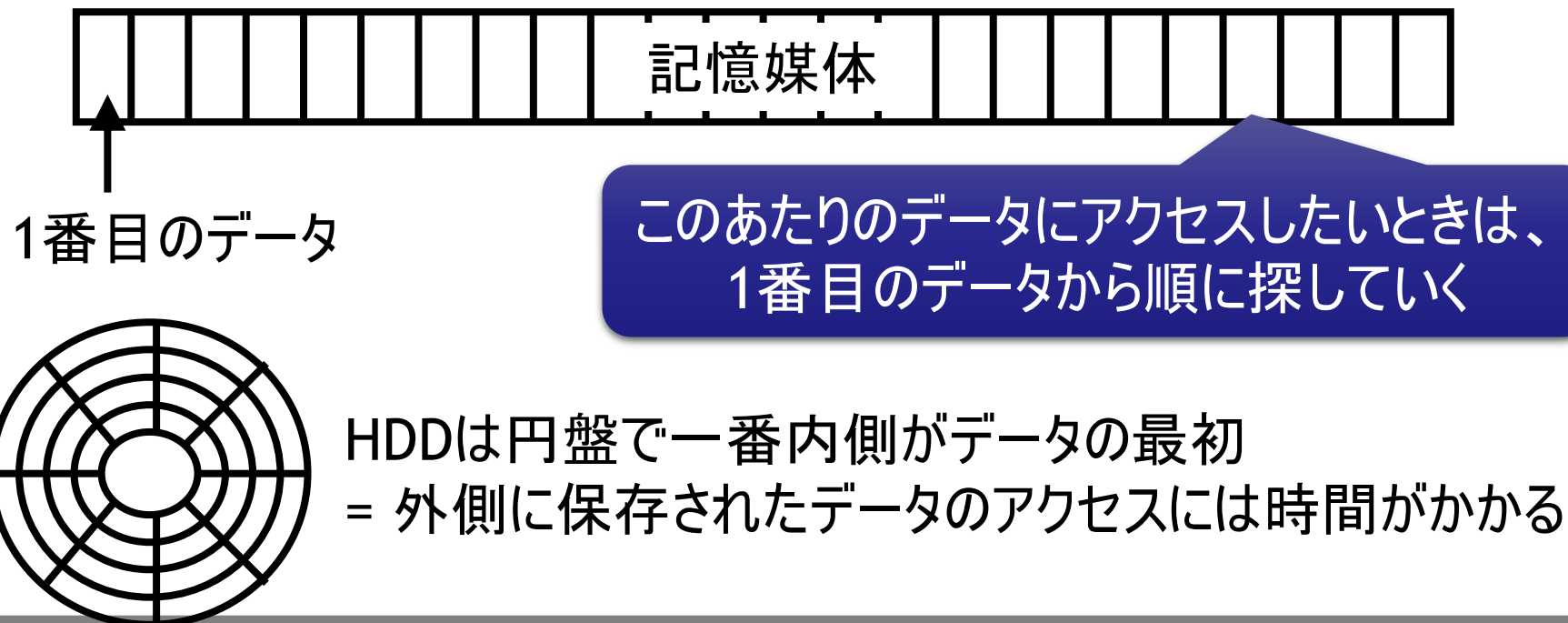


シーケンシャルアクセス

- 記憶媒体に記憶されている順番に、データを読み書きしていく方法

- ◆ 最初の方にあるデータの発見は速い
- ◆ 最後の方にあるデータの発見は遅い

様々なデータの読み書きに必要な平均の時間は多くかかる



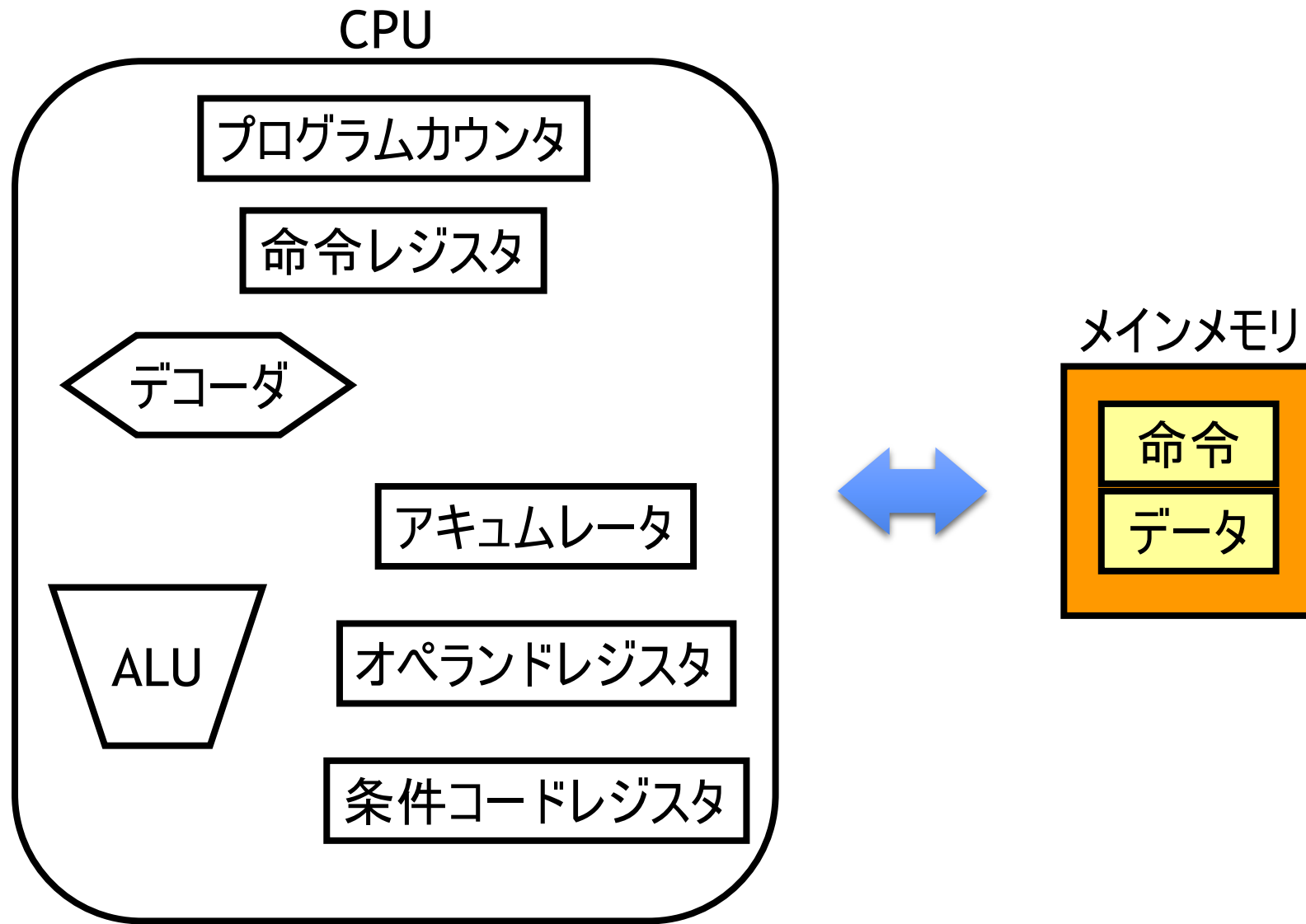
Question!

後期の内容

中央処理装置(CPU)

- プログラムの命令をメインメモリから取り出して解釈し、実行するための装置
 - ◆ プログラム: コンピュータへの命令の集合
 - 「プログラミング言語」という人間が理解できる言葉で書かれた命令の集合
 - プログラミング言語の命令を、機械語(0と1の2進数)に翻訳した命令の集合
 - ◆ 機械語のプログラムをメインメモリの中に格納
 - メインメモリの中は番地を割り振って領域が分割され、様々な命令やデータが格納されている
 - ◆ メインメモリへの命令の格納と管理もCPUの役目
 - それぞれの命令をどの番地に格納するか, etc.

CPUの構成[1](p. 39)



CPUの構成[2](p. 39)

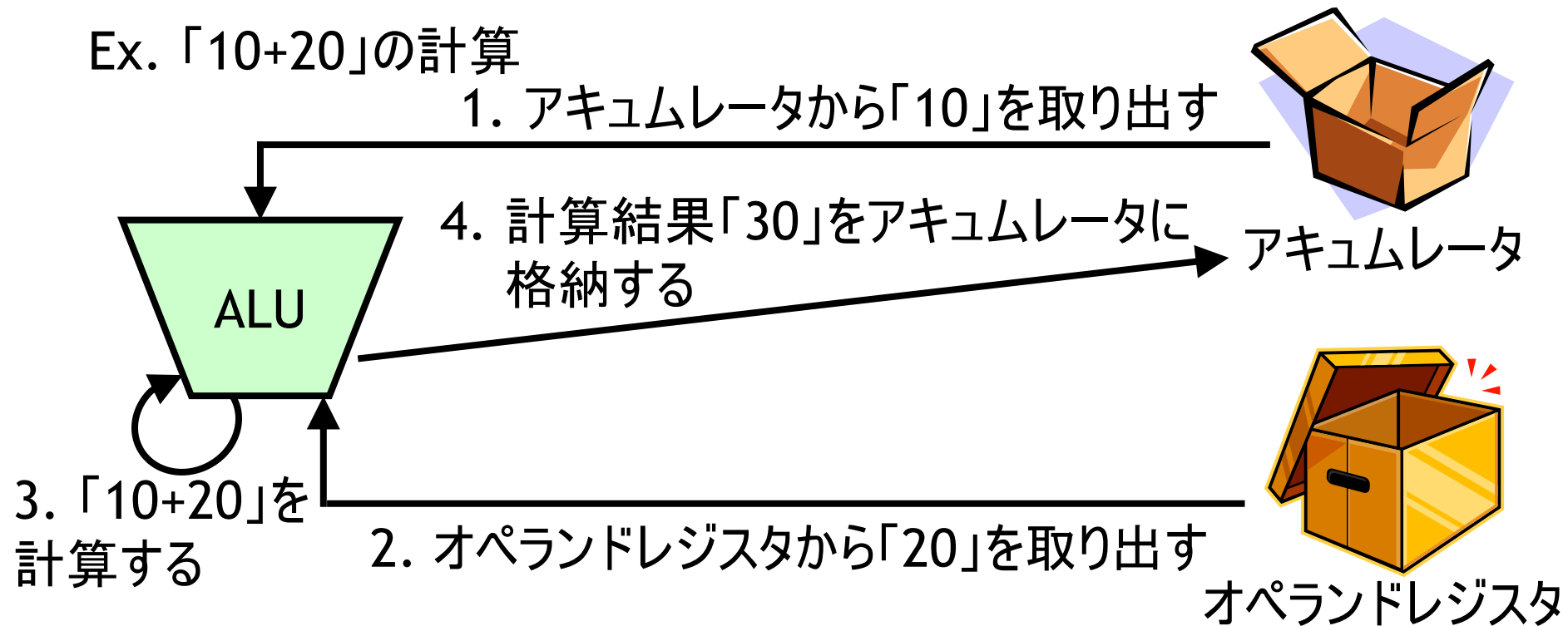
- **プログラムカウンタ**: メインメモリに格納されている命令を取り出すための番地を指定
- **命令レジスタ**: 取り出した命令を一時的に格納
 - ◆ **命令**: 命令コードとオペランドから構成
 - **命令コード**: データ転送や様々な計算、入出力処理などの処理方法
 - **オペランド**: 命令で使用するデータが格納されている番地や値など
- **デコーダ(解読器)**: 命令コードを解読し(何をすれば良いかを考え)、命令を実行するための信号を出力

CPUの構成[3](p. 40)

- **ALU(演算器)**: 演算(四則計算や論理演算など)を実行
 - ◆ メモリやレジスタの記憶されているデータを取り出し
 - ◆ 演算に使うデータ(演算数)を**アキュムレータ**に格納
 - 演算数: 「xxされる」側のデータ
 - Ex. 「A + B」の「A」
 - ◆ 演算に使うデータ(被演算数)を**オペランドレジスタ**に格納
 - 被演算数: 「xxする」側のデータ
 - Ex. 「A + B」の「B」
 - ◆ 演算結果を**アキュムレータ**に置き換え
 - ◆ **条件コードレジスタ**に条件コードを設定(必要な場合)
 - 正負の符号の判定やオーバーフローの判定など

CPUの構成[4](p. 40)

- **アキュムレータ**: 演算に使うデータや演算結果の格納場所
- **オペランドレジスタ**: 演算に使われるデータの格納場所



CPUの性能(p. 40)

- **クロック周波数**: CPUが一段階の動作を行う時間単位(サイクルタイム)
 - ◆ 単位: Hz(ヘルツ)
 - ◆ Ex. 1GHz = 1000000000Hz(10億Hz)
= 1秒間に10億回動作
 - ◆ 同じモデルのCPU同士であれば、クロックの数値の大きいものが処理が速い
 - モデル: CPUのブランドのようなもの
 - ◆ モデルが違えば、同じメーカーでも一概には比較できない
 - CPUが行う一段階分の動作は、CPUのモデルなどによって異なるため

Question!

論理回路

論理回路[1](p. 34)

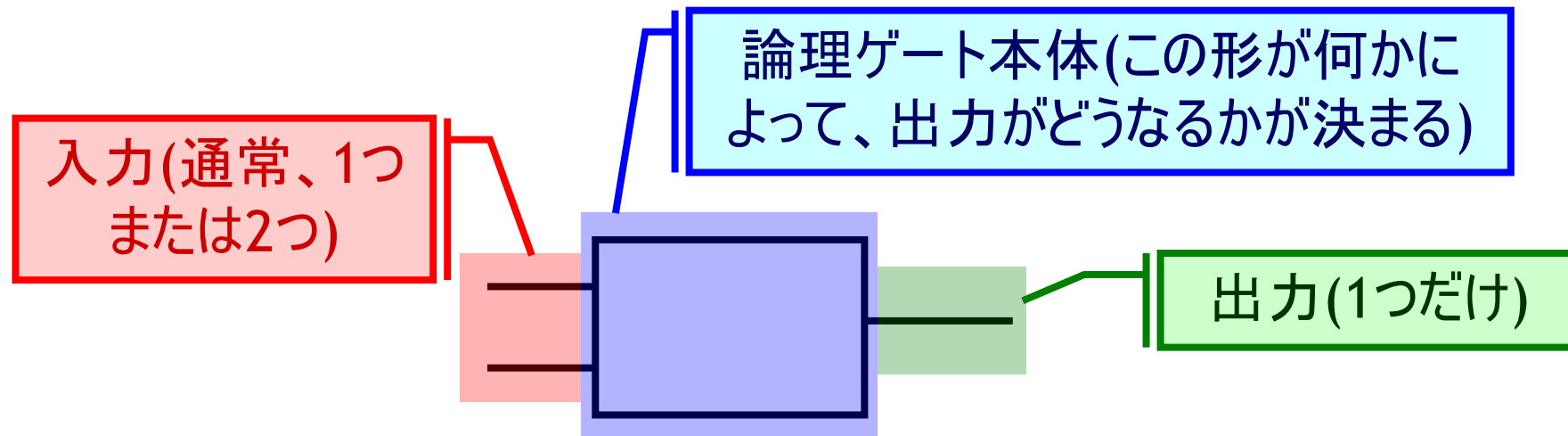
- **論理回路**: 論理演算を実現する電子回路
 - ◆ 電子回路: 電気を流すことで様々な処理をする部品
- **論理演算**: 論理型のデータ同士に対する演算
 - ◆ 論理型: 「0」または「1」の2種類のみの2進数で表現できるデータ
 - 「true」(真)と「false」(偽)で表すことも
 - ◆ 1つまたは2つのデータを入力とし、演算結果を出力
- CPUの構成要素(ALUやレジスタなど)は論理回路で構成
 - ◆ コンピュータは、様々な処理をするための回路で構成

論理回路[2](p. 36)

- 論理回路をMIL(Military standard)記号を用いて表現
 - ◆ **MIL記号**: 論理回路を構成する部品をイメージ化したもの(図として描くときに利用される絵)
 - ◆ 1つ1つの部品を「論理ゲート」と呼ぶ
 - ANDゲート
 - ORゲート
 - NOTゲート
 - NANDゲート
 - NORゲート
 - XORゲート

論理ゲート(p. 36)

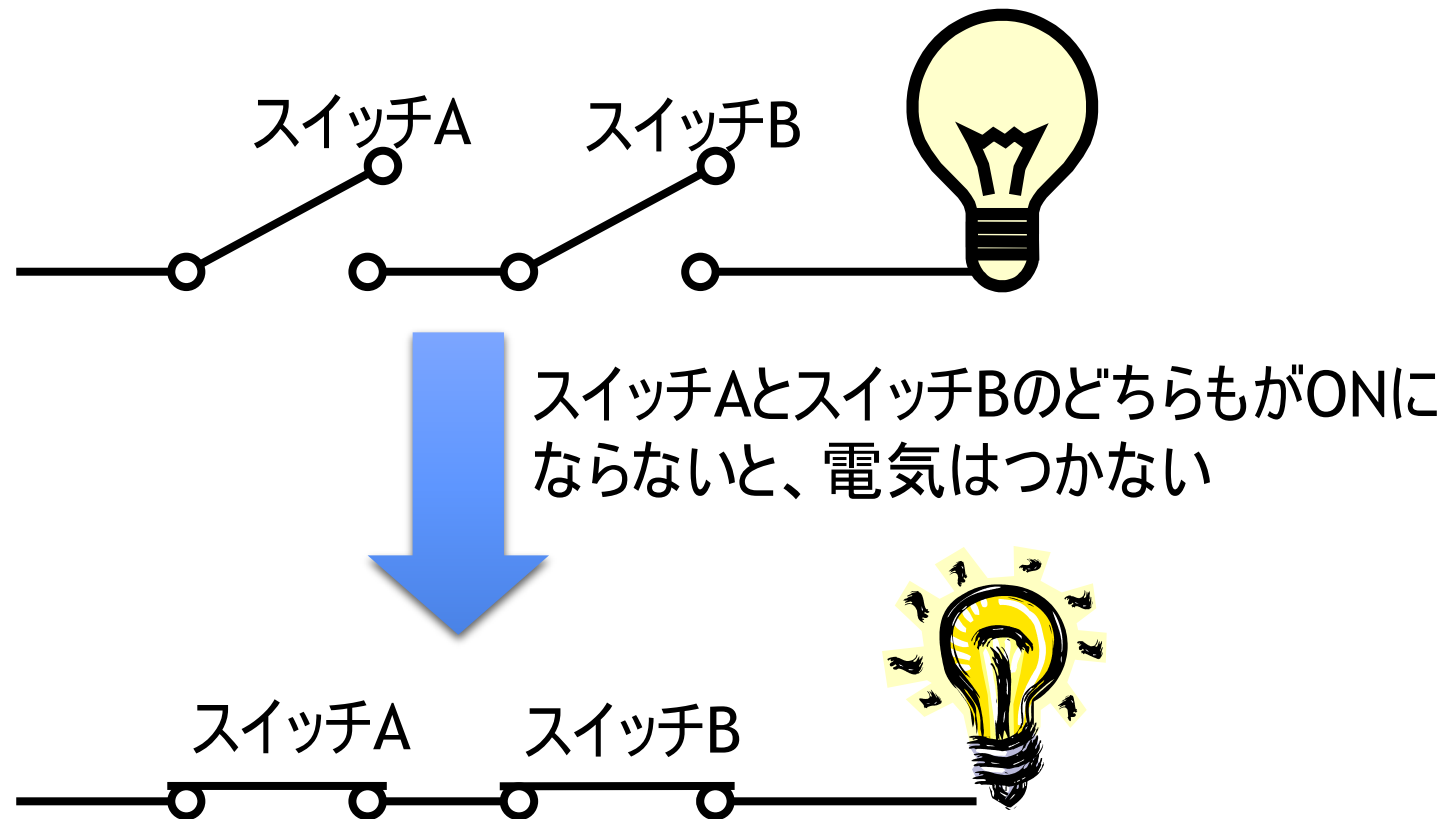
- 論理回路を構成する部品の最小単位
- 「入力」と「出力」の電気信号で構成
 - ◆ 「入力」対し、何かの処理をして「出力」とする
 - ・ 入力: 0または1の1ビット
 - ・ 出力: 0または1の1ビット
 - ◆ 1つの論理ゲートに、入力は1つまたは2つ、出力は1つ



論理ゲートの基本的な形

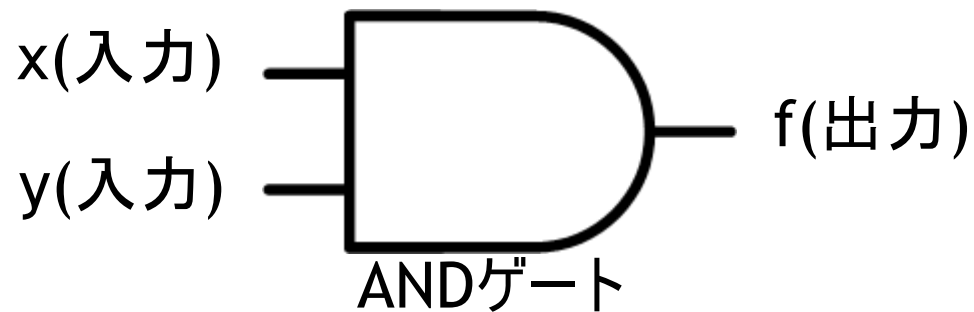
論理積[AND][1](p. 36)

- 2つの入力がどちらも「1」の場合、出力が「1」となり、2つの入力のどちらかが「0」の場合、出力が「0」となる



論理積[AND][2](p. 36)

- 論理回路は「ANDゲート」で表現
- スイッチのONを「1」、スイッチのOFFを「0」
 - ◆ ON: 線の中を電気が通っている状態
 - ◆ OFF: 線の中を電気が通っていない状態

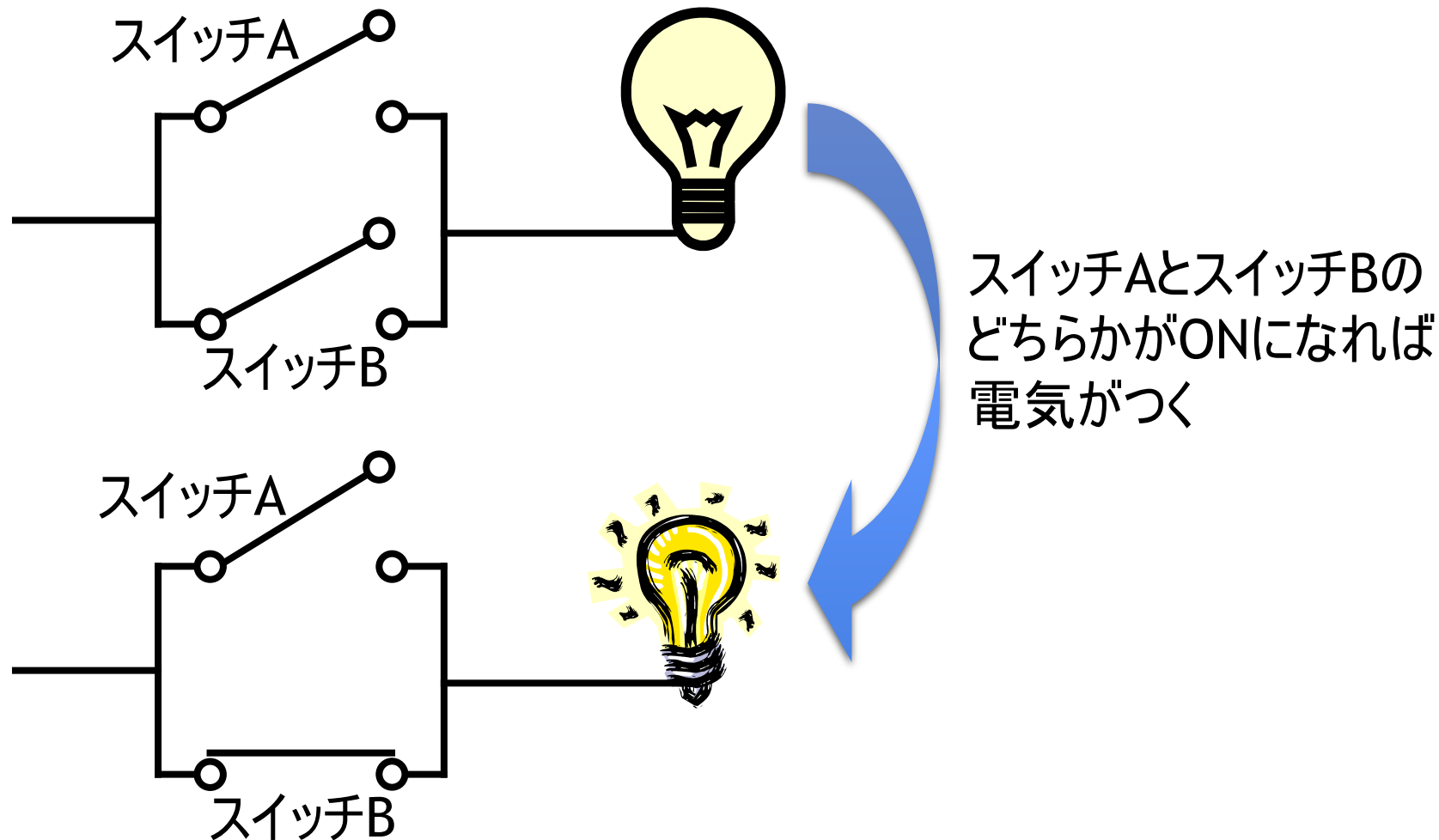


入力と出力の関係

入力		出力 (f)
x	y	
0	0	0
0	1	0
1	0	0
1	1	1

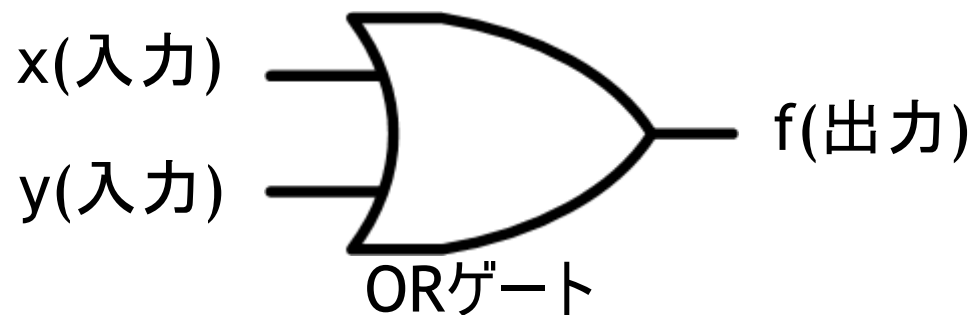
論理和[OR][1](p. 37)

- 2つの入力がどちらかが「1」の場合、出力が「1」となり、2つの入力のどちらも「0」の場合、出力が「0」となる



論理和[OR][2](p. 37)

- 論理回路は「ORゲート」で表現
- スイッチのONを「1」、スイッチのOFFを「0」
 - ◆ ON: 線の中を電気が通っている状態
 - ◆ OFF: 線の中を電気が通っていない状態

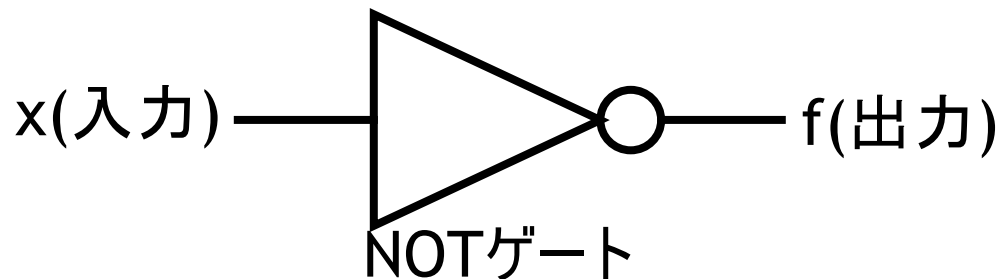


入力と出力の関係

入力		出力 (f)
x	y	
0	0	0
0	1	1
1	0	1
1	1	1

否定[NOT](p. 37)

- 1つの入力で、「0」の場合は出力が「1」となり、「1」の場合は出力が「0」となる
 - ◆ 入力の逆が出力
- 論理回路は「NOTゲート」で表現
- スイッチのONを「1」、スイッチのOFFを「0」
 - ◆ ON: 線の中を電気が通っている状態
 - ◆ OFF: 線の中を電気が通っていない状態



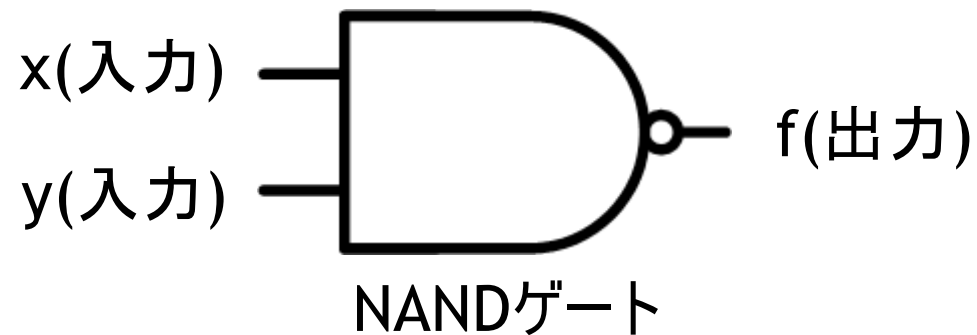
入力と出力の関係

入力(x)	出力(f)
0	1
1	0

NAND[Not AND](p. 37)

- ANDゲートと出力が逆になる

- ◆ 2つの入力がどちらも「1」の場合、出力が「0」となり、2つの入力のどちらかが「0」の場合、出力が「1」となる



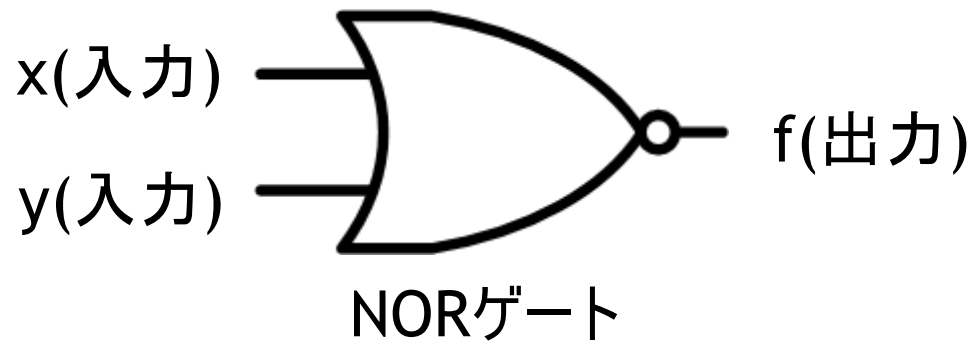
入力と出力の関係

入力		出力 (f)
x	y	
0	0	1
0	1	1
1	0	1
1	1	0

NOR[Not OR](p. 37)

- ORゲートと出力が逆になる

- ◆ 2つの入力がどちらかが「1」の場合、出力が「0」となり、2つの入力のどちらも「0」の場合、出力が「1」となる

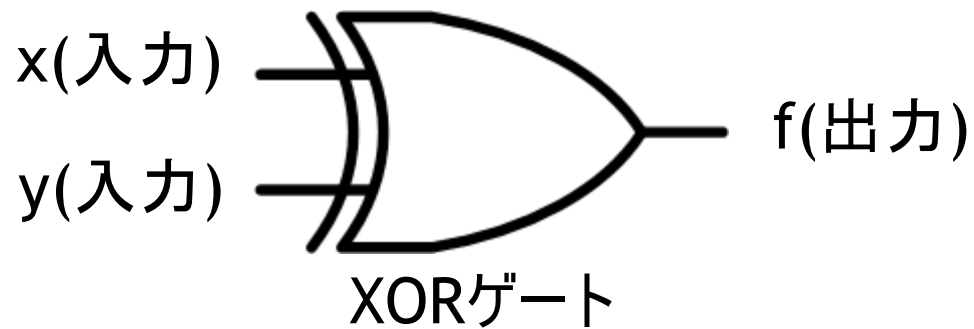


入力と出力の関係

入力		出力 (f)
x	y	
0	0	1
0	1	0
1	0	0
1	1	0

排他的論理和[XOR](p. 37)

- XOR: eXclusive OR
- 2つの入力と同じ場合は「0」となり、2つの入力が異なる場合は「1」となる



入力と出力の関係

入力		出力 (f)
x	y	
0	0	0
0	1	1
1	0	1
1	1	0

真理値表(p. 37)

- 真理値表：論理ゲートの入力と出力を表にしたもの

◆ 資格試験を受ける人：覚えよう！

◆ 資格試験を受けない人(授業だけで良い人)：覚えなくてOK

入力		出力				
x	y	AND	OR	NAND	NOR	XOR
0	0	0	0	1	1	0
0	1	0	1	1	0	1
1	0	0	1	1	0	1
1	1	1	1	0	0	0

入力xが0、入力yが0のとき、AND
ゲートの出力は0になる、という意味

入力(x)	出力(NOT)
0	1
1	0