コンピュータ基礎(4)

5章 プロセッサ

コンピュータの構成要素

- 処理装置（CPU, プロセッサ）
  （５章の授業で詳しく説明します）
  - 命令を主記憶装置から読み込んで解釈，実行する．
  - 演算装置と制御装置の２つからなる．
  - 四則演算や制御（条件判断）を行う．
- 記憶装置（３章，４章の授業で詳しくやります）
  - 主記憶装置：メインメモリ．計算機が動作している間に，処理途中のデータを一時的に記憶する．普通，電源を切ると内容が消えてしまう（揮発性）．
  - 補助記憶装置：ハードディスクなど．主記憶装置よりも大容量で，処理結果を長期的に記憶するために用いられる．電源を切手も内容は消えない（不揮発性）．
- 入出力装置（２章の授業で詳しくやります）
  - パソコンであればマウスやキーボード，ディスプレイ．
  - 家電機器の制御や画面表示なども含む．

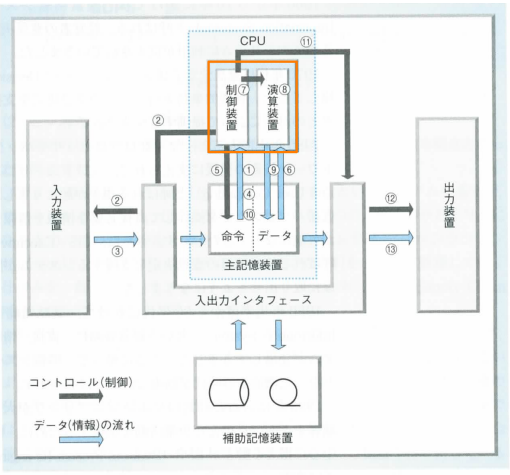
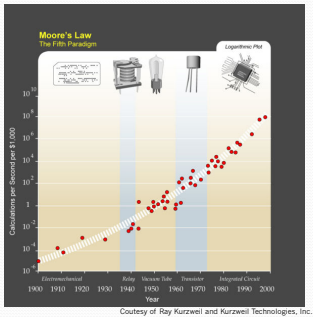


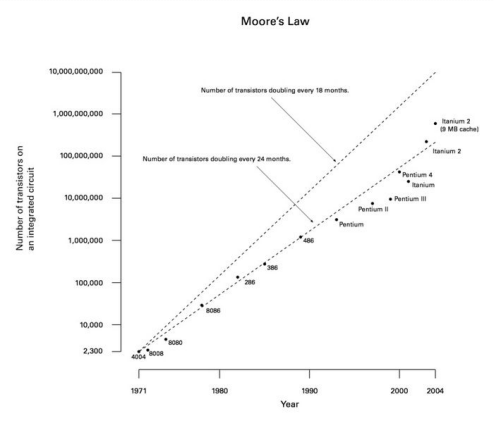
図 1-48　データの流れとコントロールの流れ

[　　]の法則

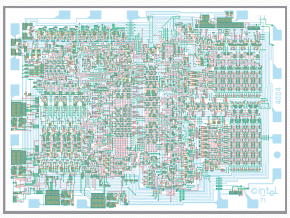
- 集積回路(IC, LSI)のトランジスタの集積密度
  - 18-24ヶ月ごとに倍になる，という経験則
  - インテルの創業者，ゴードン・ムーアが提唱



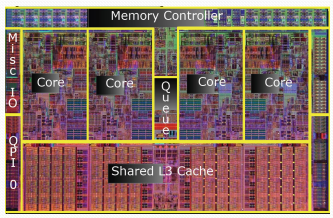
ムーアの法則



CPUの高集積化



Intel 4004，1971年
2300トランジスタ
動作クロック：731kHz



Intel Core i7，2008年
7億3100万トランジスタ
動作クロック：3.33GHz

トランジスタ数 318,000倍
経過年数　37年
$2^{(37/2)} \fallingdotseq 371{,}000$
クロック周波数も数千倍に．
１クロックでできる処理も大幅に違う．

コンピュータの速度の指標

- [　　](Million instructions per second)
  - １秒間に，いくつの命令を処理できるか．
  - 100万個単位(million)．
  - 計算機によって，１個の命令の機能が異なるため，MIPSだけで優劣が決まるわけではない．
- [　　]
  - Floating point number Operations Per Second
  - １秒間に，小数の計算をいくつ実行出来るか．
  - 科学技術計算の能力を測るのに向いている．
  - 接頭辞を付け，GFLOPS などとして使う．
  - 「京コンピュータ」は10PFLOPS.
    （P:ペタ＝10^{15}，$10\times10^{15}=1$京）

論理回路とは

- 0と1の入力から出力を決める回路

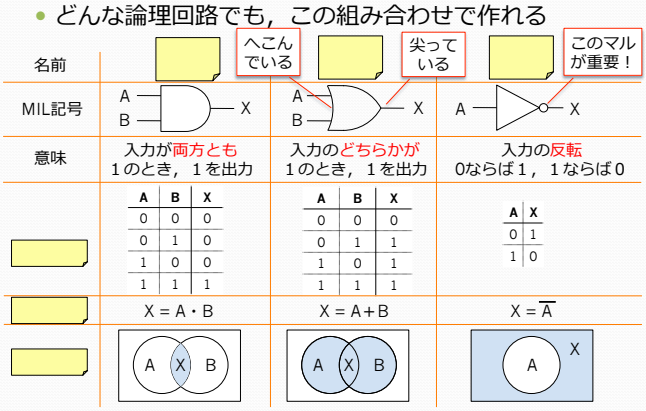
例：多数決回路（入力のうち0か1の多い方を出力する）

- 論理回路を作るには？
  - どのような部品を使うのか？
  - どのようにして設計するのか？

詳細は，２年前期「論理回路」で学習します．

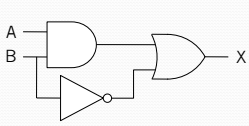
基本論理素子

- どんな論理回路でも，この組み合わせで作れる

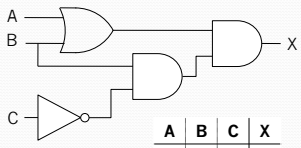


例題

- 以下の論理回路の[　　]を書け．

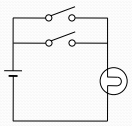


A	B	X
0	0	
0	1	
1	0	
1	1	

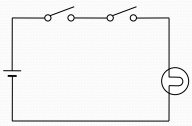


A	B	C	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

スイッチ回路



並列つなぎ
どちらかのスイッチがONなら電球が点灯するORに相当

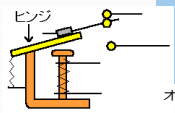


直列つなぎ
両方のスイッチがONなら電球が点灯するANDに相当

- スイッチのつなぎ方で，ANDとORの機能を作る
  - 他の回路からスイッチをON/OFF出来れば，論理回路を繋いでいくことが出来る
    → 電気によりON/OFF出来るスイッチの利用

基本論理素子の作り方

- リレー（電磁石式のスイッチ）
  - 電磁石に電気を流すとスイッチが ON になる
  - 回路が切れる方もある

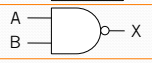
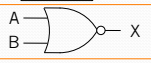
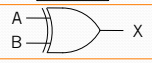


オムロンホームページより

- 基本論理素子
  - リレーを２つ，直列つなぎにすると，両方をONにしないと電気が流れない・・ANDの動作
  - リレーを２つ，並列つなぎにすると，どちらかがONなら電気が流れる・・ORの動作
  - リレーの回路が切れる側の接点を使うとNOTが作れる
- 実際に，リレーを使った計算機は存在した
  - 1950年代．計算は遅い．
  - 現在はトランジスタに置き換えられている

その他の論理回路記号

- 記号に付けられた ○ は反転を表す．
  NANDはANDの反転，NORはORの反転
- NANDかNORのどちらかだけで，
  どんな論理回路でも作ることが出来る

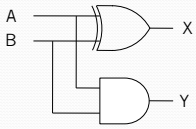
名前			
MIL記号			
意味	入力が両方とも 1のとき，0を出力	入力のどちらかが 1のとき，0を出力	入力が違えば1を出力 (排他的論理和)
真理値表	A B X 0 0 1 0 1 1 1 0 1 1 1 0	A B X 0 0 1 0 1 0 1 0 0 1 1 0	A B X 0 0 0 0 1 1 1 0 1 1 1 0

加算回路(1)

- 筆算の１けた分を考える
  - 0+0=0
  - 0+1=1, 1+0=1
  - 1+1=0 で，1繰り上がる
- A+B=X で，繰り上がり Y とすると
  - X は A と B の XOR
  - Y は A と B の AND
- これを［　　］という
  - 下の桁からの繰り上がりを
    計算できないので，「半」

```
  0 0 1 1
+ 0 1 1 0
  1 0 0 1
```

A	B	X	Y
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

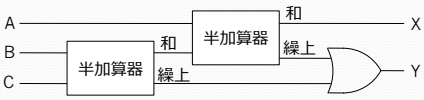


加算回路(2)

- 下の桁からのくり上がりを考える
  - 0+0+0=0
  - 0+0+1=0+1+0=1+0+0=1
  - 0+1+1, 1+0+1, 1+1+0 のときは
    その桁は０で，１繰り上がる
  - 1+1+1のときは，その桁は１で，
    １繰り上がる
- これを［　　］という
  - 半加算器を２つ使って作れる

```
  0 0 1 1
+ 0 1 1 0
  1 0 0 1
```

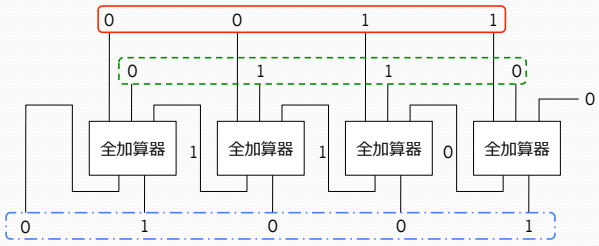
A	B	C	X	Y
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



加算回路(3)

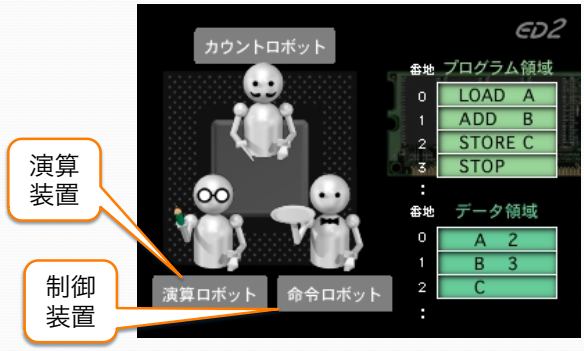
- 筆算の回路を作ることが出来る

```
  0 0 1 1
+ 0 1 1 0
  0 1 0 0 1
```



- その他の計算
  - 引き算：２の補数（反転して１を足す）して加算
  - 掛け算：シフト（桁ずらし）と加算の繰り返し

制御装置と演算装置



- 「計算を行う部分」だけでは，複雑な処理は出来ない．プログラム（命令）を解釈する部分が必要．

命令の読み出し

- 命令アドレスレジスタ（　　）
  - 現在，実行中の命令がメモリのどこにあるかを常に記憶している，特別なレジスタ．
- CPUの動作
  1. [命令取り出し（フェッチ）]命令アドレスレジスタに記憶されているアドレスから，命令を読み出す．
  2. [命令解釈（デコード）]命令によって，CPUのどの回路を働かせるかを決める．
  3. [命令実行]メモリにデータを読み書きしたり，計算を行ったりする．

クロックとレジスタ

- クロックに合わせて命令の読み込み，解釈，実行が行われる．
- 計算途中結果はCPU内部のレジスタに蓄えられる．

記憶階層

重要!!

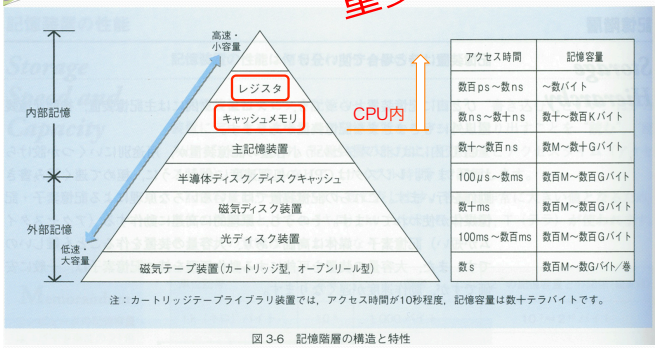


図 3-6 記憶階層の構造と特性

- 大容量のメモリほど遅い
  - 速度と記憶容量の両立のため，階層化されている

CPUの仕組み

- 制御装置
  - 主記憶装置から命令を取り出し・解釈して，他の装置（演算装置など）を動作させる
- 演算装置
  - 四則演算，論理演算，大小比較などを行う．
- [　　]
  - 計算の途中結果や，処理対象のデータが主記憶装置のどこにあるか（データのアドレス）を記憶する．
  - CPU内部に設けられ，非常に高速だが小容量．
- [　　]
  - 計算機の動作のタイミングを取る信号．
- [　　]
  - 計算機外部とデータのやりとりをする信号線．

命令の仕組み

- 命令部
  - 命令の種類を数値で表す．例えば，メモリからデータを読み出すのが 0x10，メモリへのデータの書き出しが0x20，足し算が 0x30，引き算が 0x40など．
- アドレス部
  - データを読んだり書いたりする対象のメモリのアドレスを表す．
  - 命令によっては，計算する値そのものだったり，レジスタの番号だったりもする．

CPUの動作と命令

- 命令とは？
  - 高級言語（例えばC言語）で作ったプログラムをコンパイルすると，実行ファイルができる．この中身は，CPUが直接解釈できる命令（機械語）である．
  - 機械語は，それぞれの単純な命令が二進数で表されている．
  - 変数名などの「名前」は全て，数値に置き換えられている．
    - int a, b, c; とすると，a, b, c という変数に対応するメモリの番地（アドレス：例えば 100, 104, 108 ）が決められる．
    - 例えば，C言語のプログラムで c = a + b; とすると，次のような機械語命令が作られる．
      - 100番地からデータを読み出してレジスタに入れる．
      - 104番地からデータを読み出してレジスタの値に加える．
      - レジスタの値を108番地に格納する．

CPUの入出力制御

- 計算機のデータ信号線をバスと呼ぶ
  - CPUと直接つながっているもの：内部バス．CPUと，メモリや入出力装置をつなぐ．それ以外は外部バス．
  - パソコンに内蔵する拡張カードの端子をシステムバスと呼ぶ．共通規格化されている．
- パラレルインタフェースとシリアルインタフェース
  - パラレル（並列）インタフェース：複数の信号線で同時にデータを送る．
    - 8bitのデータなら，8本の信号線で送るなど．
    - 例：PCI, SCSI, GP-IB, ATA など．
  - シリアル（直列）インタフェース：１本の信号線に1bitずつ順次送る．最近は高速化が著しい．
    - 例：RS-232C, USB, IEEE1394, Serial-ATA など（有線）
    - IrDA（赤外線），Bluetooth（電波）なども．