2008/07/17

情報(金３　中村政隆先生)

実習＆第３･６･７章編

ごうしょう

情報(金３　中村政隆先生)

実習&第３･６･７章編

情報のシケプリ（実習で扱った範囲）を作成することになったごうしょうです。とまや君と共に分担して情報のシケプリを作ることになりました。以下注意点です。

1. 講義編とは別の人が作っているので内容的に重複する可能性があります。

2. やたらマニアックになるのは必至です。内容が工学部レベルです。校閲やわからない部分は工業高専→**工学部３年の石井**さんに手伝ってもらいました。

3. 量も多くなりそうです。早めに読み進めることをお勧めします。

4. 「情報」の教科書の説明があまりにも飛躍しているのでごうしょうが**再構築しました**。新しい教科書を書いているようなレベルです。教科書との対応ページなども書いてあります。あ、**「６．最後に」を最初に読んでください。**

5. しかしこの実習編の内容も試験に出たりするので注意が必要です。

6. 実習なので PC で実際に動かすとよいと思います。（http://lecture.ecc.u-tokyo.ac.jp/~nakamura/2008/2008_index.html）また中村先生のレジュメもわかりやすいです。CFIVE で落とせます。

7. わかりやすく説明しているつもりですが、わかりにくかったらごうしょうまで。

以下このシケプリで扱う内容…

6/6 ＩＣトレーナー実習

6/20WWW・メイルシミュレータ実習　オートマトンシミュレータ実習 Travelling Salesman ゲーム

6/27 オートマトンシミュレータ ED21 CPU シミュレータ

7/4 HTML(文 12)

適宜教科書第３章―3.2 情報通信　第６章―6.2 計算のモデル化　第７章―コンピュータの仕組み

重要な語句は赤をつけ、意識すべき語句には下線をひきました。試験範囲外・参考程度のものは薄い青色にしました。

また教科書に関しては**「教科書第*章」**または単に**「7.2.1」**のように章節番号のみの場合もあります。「P.***」と出てきた場合も教科書を参照してください。

文体がそろってなかったり、日本語としておかしい部分が多々あると思います。勘弁してください m(_ _)m

1．コンピュータの仕組み（教科書第７章）

・ディジタルな世界(参考)

アナログとディジタルの違いについてはもうよいでしょう。コンピュータの世界ではほぼすべてのデータはディジタル信号として扱われています。ところでディジタルというのは数値、という意味ですが、私たちが普段よく使っている数字(digit; １０進数が基本)とは異なり、ディジタルな世界では２進数（0 と 1 のみ）で表されています。これを実際には電圧の ON/OFF で区別しています。膨大なデータであっても２進数の桁数を増やして対応しているわけです。この桁数をビット（bit）といっています。第３章のビットとは異なることに注意してください。

ディジタルの世界では電圧の ON/OFF は２進数の０と１に対応しています。これからの説明においては電圧の ON/OFF を２進数の０と１を同値に考えます。

・真理値と論理関数(教科書 7.2.1)

コンピュータは２進数で入力、計算、出力を行っていることがお分かりいただけたでしょう。２進符号表現の演算を実現する回路は、組合わせ回路と順序回路に大別されます。組合わせ回路とは計算途中を保存できないが任意の２進演算を行うことのできる関数のような回路を言います。つまり単純な計算を行います。それに対して順序回路とは計算途中を保存する機構を持ち複雑な計算を行えるようになった回路を指します。ここでは組合せ回路について説明します。では、どのような演算ができれば任意の２進演算を行えるのでしょうか？

２進数の加算、といっても１０進数の加算と基本は同じです。記数法と位取りに注意しましょう。ここでは一番簡単な加算、つまり（１ビットの変数 x）＋（１ビットの変数 y）について考えましょう。このときの結果を s（１ビット）とし、位が上がった時に C_{out}（１ビット）を出力することにしましょう。

x	y	s	C_{out}	１０進法
0	0	0	0	0+0=0
0	1	1	0	0+1=1
1	0	1	0	1+0=1
1	1	0	1	1+1=2

（x，y は１ビットの変数　s は１ビットの結果　C_{out} は桁上げの出力）

このような関係になります。１０進数表記もしたので簡単に理解できるでしょう。このように２進演算の入出力の関係を示した表を真理値表といいます。

実際の計算の時は下位の計算結果からの桁上げを考慮するので桁上げ入力 C_{in} も考慮します。

	x	y	C_{in}	s	C_{out}	１０進数
①	0	0	0	0	0	0+0+0=0
②	0	0	1	1	0	0+0+1=1
③	0	1	0	1	0	0+1+0=1
④	0	1	1	0	1	0+1+1=2
⑤	1	0	0	1	0	1+0+0=1
⑥	1	0	1	0	1	1+0+1=2
⑦	1	1	0	0	1	1+1+0=2
⑧	1	1	1	1	1	1+1+1=3

（x，yは1ビットの変数　Cinは桁上げ入力　sは1ビットの結果　C_{out}は桁上げの出力）

2進数の各桁の演算を前の2例において入力桁数を増やした形、つまりn入力1出力の論理関数全種類と、その実行法を考えます。nビットの入力パターンは2^n通り(前の表を見れば自明)でありそれぞれの入力パターンについて0または1の出力が可能だから論理関数o_iの種類（真理表は全部で何パターンか？）は2の2^n乗となります。

それではこれらの任意の演算が行える関数は何でしょうか？それは{AND,OR,NOT}の組合わせです。（なお、説明はp.159を見てください。また、AND,OR,NOTがどのような論理関数かは**ブール代数**の項で扱います。）このすべての論理関数を実現できる演算の組み合わせ（これらを組み合わせることですべての論理演算ができる組み合わせ）を完備な組合わせと呼び、この性質を完備性といいます。

またNANDやNORは其々が単独に完備性を持ちます。単一の演算で完備となるため実際の論理回路ではほとんとがNANDとNORでできています。（※制作が簡単なNANDが実際では使われることが多いようです。）

ブール代数（7.2.2）

先ほどは組合せ回路の構成や性質をあらわすのに真理値表を用いましたが、そのほかにブール代数とMIL記法があります。

ブール代数は論理回路を数式として表わすためのもので、ANDを・（論理積）、ORを＋（論理和）、NOTを$\overline{x}$（論理否定）で表します。

AND演算（・　論理積）…2つの入力x,yにたいして両方が1でない限り0を返します。つまり2進数1ビットの掛け算をするわけです。

OR演算（＋　論理和）…2つの入力x,yに対して片方でも1であった場合は1を返し、両方0の場合のみ0をかえします。つまり2進数1ビットの足し算をします。

NOT演算（$\overline{x}$　論理否定）…1つの入力にたいして0ならば1、1ならば0を返します。つまりビットを反転させます。

わかりにくいひとも出てくるかもしれませんが、数Aの集合と同じです。

ANDを$x \cap y$、ORを$x \cup y$、で考えるとよいでしょう。

x	y	s	10進法
0	0	0	0*0=0
0	1	0	0*1=0
1	0	0	1*0=0
1	1	1	1*1=1

AND演算の真理値表

x	y	s	10進法
0	0	0	0+0=0
0	1	1	0+1=1
1	0	1	1+0=1
1	1	1	1+1=1(実際とは異なりますが、ここは説明のために便宜的にこうしてあります。なお10進数の計算に近い答えを足すのは半加算器)

OR演算の真理値表

x	S
1	0
0	1

NOT演算の真理値表

論理式（論理演算）	Boole代数式
$x \cap y$	$p \cdot q$
$x \cup y$	$p + q$
$\overline{x}$	$\overline{p}$

対応表

この他にも NAND,NOR、XOR などがあります。

NAND($\overline{x \cdot y}$)・NOR($\overline{x + y}$)…それぞれ NOT+AND　NOT+OR なので詳しい説明は省きます。AND、OR の結果を反転させればよいわけです。

XOR（排他的論理和　x⊕y）…基本的には OR 演算と同じです。しかし x,y がともに 1 であった場合は 0 を返します（排他的）。２つのビットを比較して同じならば０を、異なれば１を返します。（cf. 5．補足 ハミング符号）

x	y	NOT $\overline{x}$	AND $x \cap y$	OR $x \cup y$	NAND $\overline{x \cap y}$	NOR $\overline{x \cup y}$	XOR $x \oplus y$
1	1	0	1	1	0	0	0
1	0	0	0	1	1	0	1
0	1	1	0	1	1	0	1
0	0	1	0	0	1	1	0

様々な論理演算の真理値表

加算標準系・乗算標準系

ブール代数表現においては、数学における他の演算と同様に同じ内容の式でも様々な表現方法があります。それらの中でいかなる演算においても唯一の表現に定まる形式を標準系といいます。

加算標準系…変数の総数が n 個である際に n 個の変数からなる論理積の論理和で表現された形式です。

乗算標準系…変数の総数が n 個である際に n 個の変数からなる論理和の論理積で表現された形式です。

再簡(minimal)加算標準形

命題1　$A + A \cdot B = A$

命題2　$A \cdot p + B \cdot \overline{p} = A \cdot p + B \cdot \overline{p} + A \cdot B$

例
$$\begin{aligned} E &= x \cdot y \cdot z + \overline{x} \cdot \overline{z} + x \cdot y \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot \overline{z} \\ &= x \cdot y \cdot z + \overline{x} \cdot \overline{z} + x \cdot y \cdot z + \overline{x} \cdot \overline{y} \cdot z \\ &= x \cdot y \cdot (z + \overline{z}) + \overline{x} \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot z \\ &= x \cdot y + \overline{x} \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot z \\ &= x \cdot y + \overline{x} \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot z + \overline{x} \cdot \overline{y} \\ &= x \cdot y + \overline{x} \cdot \overline{z} + \overline{x} \cdot \overline{y} \\ &(= x \cdot y + \overline{x} \cdot \overline{z} + \overline{x} \cdot \overline{y} + y \cdot \overline{z}) \end{aligned}$$
（再簡加算標準形）19

定理１は、A＋A・B＝A(1 ＋B)＝A ってことだね。どんなものでも、１とのOR をとると１になるので、Bは消えます。 定理２は、 $A \cdot p + B \cdot \overline{p} = A \cdot p(1 + B) + B \cdot \overline{p}\ (1 + A) = A \cdot p + A \cdot p \cdot B + B \cdot \overline{p} + B \cdot p \sim \cdot A\overline{p} = A \cdot p + B \cdot \overline{p} + A \cdot B(p + \overline{p}) = A \cdot p + B \cdot \overline{p} + A \cdot B$ *という変形をしてるんだね。１＋〇＝１だから、好きなところにかけていい。１とAND とっても変化しないから。結局、定理１は無駄な項を消すテクニック、定理２は欲しい項を取り出すテクニックということです。*

- 基本積 P が E の主項であるとは、E+P=E かつ P の真部分積がその性質を持たないこと。

- 定理　任意のブール式に命題 1, 2 の変形を適用すると有限ステップで終了し、かつ E は E の主項の和になっている。

20

簡単に言うと主項っていうのは、ある式をできるだけ単純化した式の中にある、それぞれの項のこと。ある式はいろんな形に変形できるけど、「主項の和で表す」って決めたら、その形はひとつに決まります。この形のことを加法標準形といって、論理回路を設計する際の基本にするわけです。※ちなみに論理回路では AND と OR は対等だから、乗法標準形というのも存在します。こっちはある式を完璧に因数分解された形にもっていくわけです。

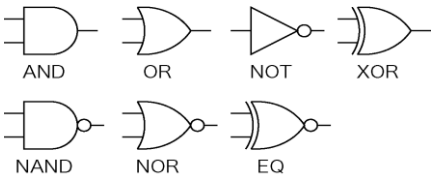
以上石井さんの解説でした。神☆正直手に負えない;;

MIL 記法（7.2.3）

論理演算を数式として表わす方法を紹介ました。もう一つ論理演算を回路図として表わす方法があります。これを MIL 記法（アメリカの軍用規格の一部）といいます。以下授業用スライドを示します。

MIL記法

- 基本的な論理演算を以下の基本素子で表現



- 基本素子のあいだの結線によって論理関数を表現

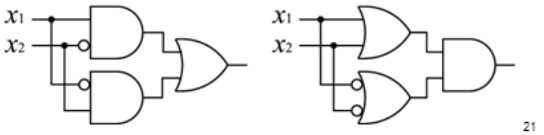
16

これらの基本素子（EQ は教科書にありますが触れません。）のあいだを結線することで論理回路を示します。なお、NOT、NAND、NOR に見られる○は否定を表します。

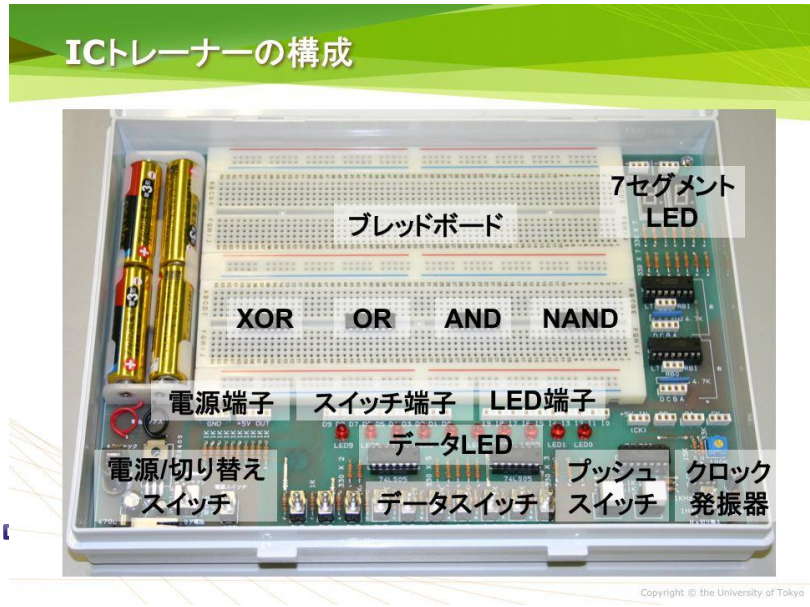
XORの標準形による表現

- XOR（排他的論理和）
  - 2入力が同じ値のとき0, 異なる値のとき1
- ブール代数によるXORの表現
  - 加算標準形　$x_1 \cdot \overline{x_2} + \overline{x_1} \cdot x_2$
  - 乗算標準形　$(x_1 + x_2) \cdot (\overline{x_1} + \overline{x_2}) =$
- MIL記法によるXORの表現

加算標準形　　乗算標準形



21



また XOR はほかの論理素子からつくるとこうなるらしいです。授業用スライドより

IC トレーナー実習(6/6)

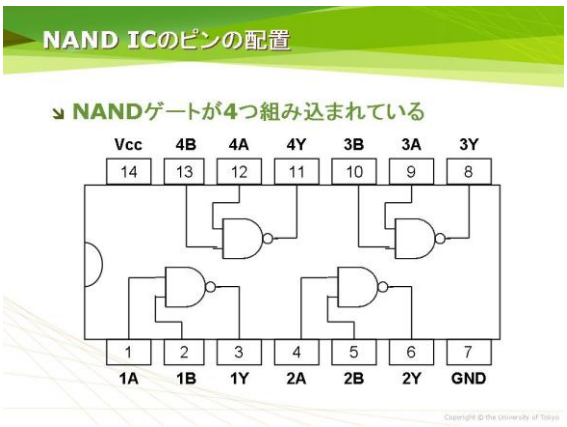
やっと実習に入れました。実習編をよんでわかりにくいところがあれば適宜戻ってください。

まずは IC トレーナーの各部分について説明しましょう。

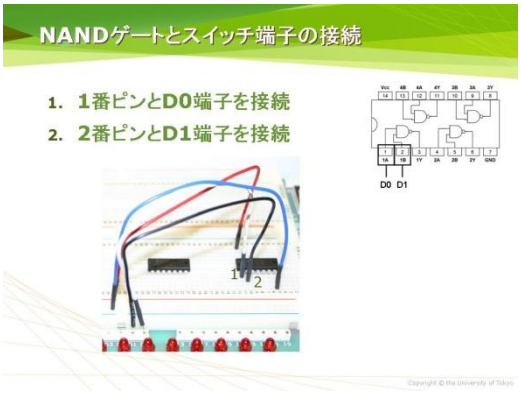
- ✧ 電源端子・電源/切り替えスイッチ…この IC 群を動かすための電源部分です。安定した電源を得るために電源用 IC が入ってます。ここはあまり問題ないでしょう
- ✧ ブレッドボード…各種電子部品やジャンパ線（部品間をつなげる為の線）を差し込むことで電子回路を組むことのできる電子基盤の一種。組み立て・組み換えが簡単なので実験用につかわれる。
- ✧ スイッチ端子・データスイッチ…端子とスイッチはつながっている。スイッチは入力のための装置として使われる。計算のための数値を入力する。
- ✧ LED 端子・データ LED…LED（発光ダイオード）を光らせることにより計算結果を出力する。スイッチ・LED は入出力装置（I/O）といえる。
- ✧ その他…今回は関係ありません。

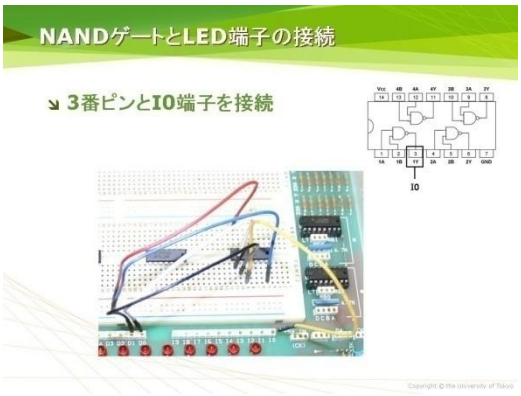
それではやっていきましょう。回路の接続方法・動作確認は飛ばして…

NAND ゲートの動作確認



一番右にある NAND IC74LS00 の中にはこのような論理回路が入っています。なお Vcc は電源＋で GND は電源－に接続します。





IC と各部品を接続します。1 A と 1 B にはスイッチ D0 と D1 を、1 Y には LED I0 を接続します。

D0	D1	I0
0	0	1
0	1	1
1	0	1
1	1	0

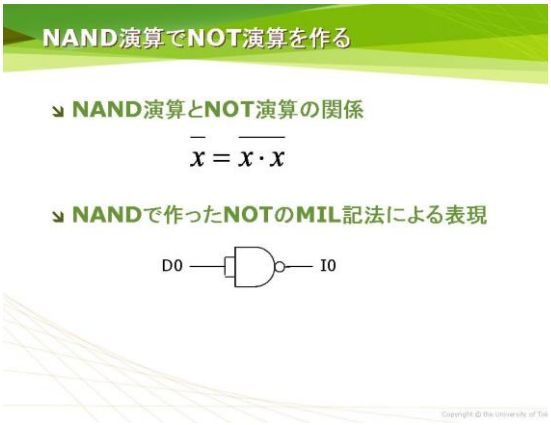
	0	1
回路	電圧低	電圧高
スイッチ	OFF（手前）	ON（奥）
LED	消灯	点灯

それでは，動作を確認します．スイッチおよび LED と端子は，SW0 と D0, SW1 と D1, LED0 と I0 がそれぞれつながっています．たとえばスイッチ SW0 を ON にし，SW1 を OFF にすると， NAND ゲートへは D0 から 1，D1 から 0 が入力され，1 が I0 に出力されますから，LED0 が点灯します．実験ボードの電源スイッチを ON にし，ス

イッチ SW0 と SW1 の ON, OFF を切り替えて， LED0 が真理値表のとおりに点灯もしくは消灯することを確認しましょう．

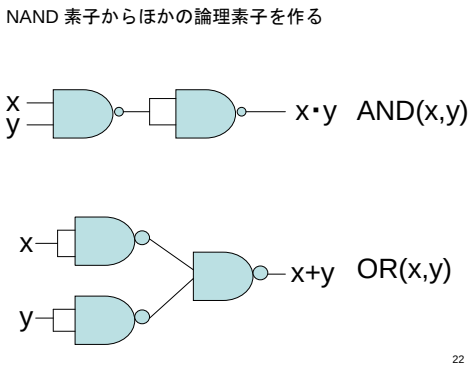
…まぁコピペです。突っ込みどころがありません。論理演算と実際の動作がリンクしたでしょうか？

NAND ゲートの応用



NAND では NOT を作ることができます。AND 部分の２入力を同じにすればいいのです。そうすると、AND{0,0}=0 AND{1,1}=1 なので、反転して NAND{0,0}=0 NAND{1,1}=1⇔ NOT{0}=1,NOT{1}=0 となります。

NAND ゲートで AND・OR 演算をする



これは実習において「必須課題」(http://lecture.ecc.u-tokyo.ac.jp/~nakamura/2008/IC-trainer-exercise.html)でした。説明を示すので、上の図において指で追っていくとわかりやすいです。

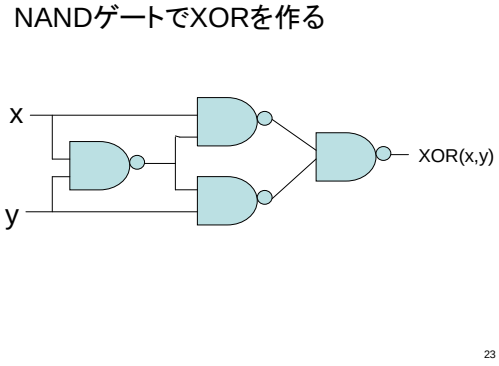
NAND ゲートで AND ゲートを作る…ひとつめの NAND では x,y ２つの入力に対して NAND 演算を行います。ふたつめの NAND ゲートを NOT 演算を行います。つまり、**NOT{** NAND{x,y}**}** というわけです。

NAND ゲートで OR 演算をする…ちょっと難しいです。実際に数値をいれて順に計算していくとよいです。x,y の入力をそれぞれ反転させて NAND 演算します。段階別にして真理値表をしめします。

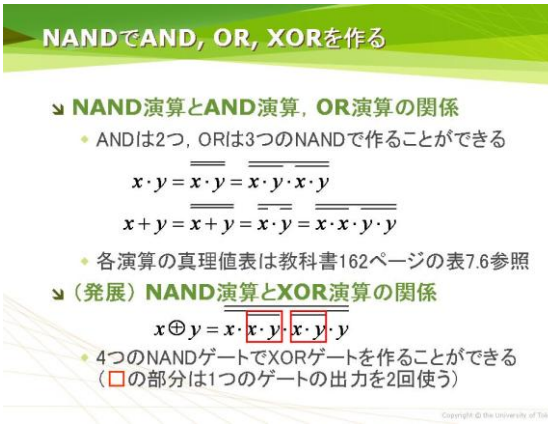
	入力	NOT	AND	NOT	s
x	1	0			
			0	1	1
y	0	1			

x,y の値を変えることでほかの場合を示すことができます。つまり **NAND{ NOT{x}, NOT{y} }**といえます。

NAND から NOT,AND,OR が作り出せました。「NAND 演算は単独で完備性を持つ」、とさきほど書きましたが、それを MIL 記法で示しました。



選択部分です。もうわかると思いますので解説はしません。ここまで MIL 記号で示してきましたが、これをブール代数で示すとこうなるらしいです。



もう意味はわかるはずです。発展的ですよね。

１ビット半加算器

それではこれらの論理 IC を組み立てることで計算をしていきましょう。すべての計算の基本である加算器についてやっていきます。加算器のビット数を増やしたりすることですべての計算ができます。１秒に１つカウントすることで時計になったり。実際石井さんは高専時代ロジック IC だけで時計を作らされそうになって焦ったそうです。IC を２０～３０個必要なのかな？カオスになることは確実。ちょっとロジック IC がコンピュータの基本になっていることが実感できるようになったと思いますよ。

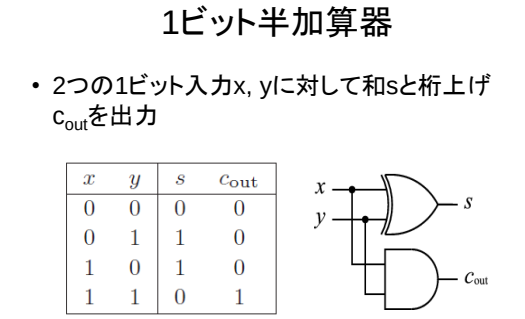
まずは１ビット半加算器。一番初めにでた真理値表を覚えていますが？１ビット＋１ビット＝２ビットの足し算です。

x	y	s	C_{out}	１０進法
0	0	0	0	0+0=0
0	1	1	0	0+1=1
1	0	1	0	1+0=1
1	1	1	1	1+1=2

これをブール代数表現や論理関数表現で表すと…

s=x⊕y=XOR{x,y}　Cout=x・y=AND{x,y}

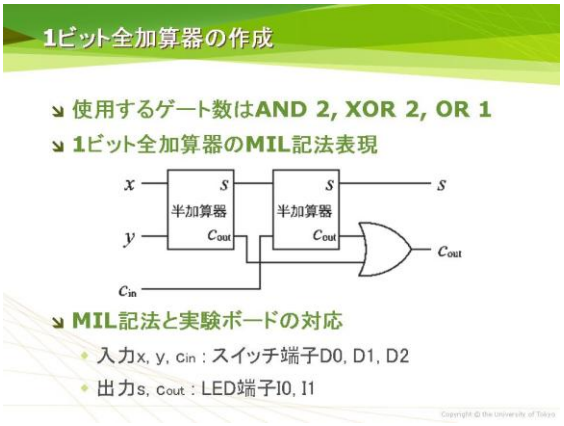
となります。これを MIL 記号で示すとこのようになります。それを１ビット半加算器といいます。



指で追うと合点できるでしょう。下位ビットの桁上げを考慮しないので半加算器と呼ばれます。

１ビット全加算器

それでは下位ビットの桁上げを考慮しましょう。



真理値表：

	x	y	C_{in}	s	C_{out}	１０進数
①	0	0	0	0	0	0+0+0=0
②	0	0	1	1	0	0+0+1=1
③	0	1	0	1	0	0+1+0=1
④	0	1	1	0	1	0+1+1=2

⑤	1	0	0	1	0	1+0+0=1
⑥	1	0	1	0	1	1+0+1=2
⑦	1	1	0	0	1	1+1+0=2
⑧	1	1	1	1	1	1+1+1=3

この全加算器を数珠つなぎにすることで何ケタもの数字を扱えるようになります。つまりｎビット全加算器ができてしまうわけです。

減算器（参考　7.3.4）

加算ができることがわかりました。それでは減算はどうするのでしょうか？そのためには入力を変更します。最上位のビットを符号ビットとして固定してしまいます。そして数値の絶対値部分を２の補数表現をつかってしめします。まぁなんのことやらわかりませんが･･･ようは正の数のビットを反転させると負の数にかわります。これらをそのまま加算すると引き算になっている･･･そうです。まぁ要はビットを反転させることで引き算ができるのです。

	符号 bit	絶対値			１０進数
正の数	0	1	0	1	+5
負の数	1	1	1	0	-2
加算結果	0	0	1	1	+3

ALU(参考　7.3.4)

そんなこんなで足し算・引き算ができました。これらをもっと変化させると加減乗除ができるらしいです。拡張していくことで様々な計算ができるユニットを ALU（Arithmetic and Logic Unit　算術・論理演算器）といいます。要はコンピュータにおいて実際に計算を行う部分です。

ED21 シミュレータ(6/27)

やっと終わりました。IC トレーナー。あれを理解するのにはこれだけの知識が必要だったのですね。

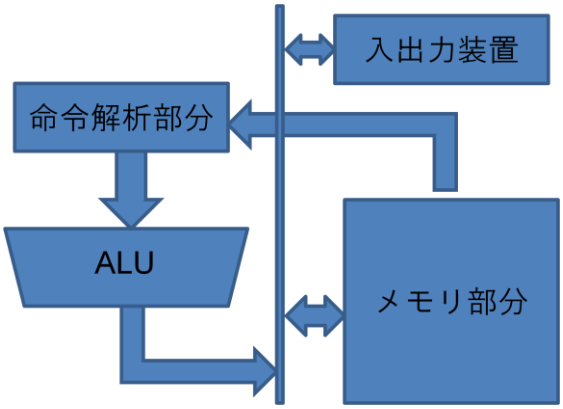
それではやっていきましょう。…このシミュレータは ECCS からでないと使えないみたいです。なお、「じょうほうの情報」(http://www.geocities.jp/infofinf/)に用意しておきます。

マイコン（参考）

このシミュレータは私たちが普段使っているパソコンを簡略化したものです。現在ではこれに近い構造をとっているのがマイコンと呼ばれているものです。マイコンは一定の目的のために機能が限定されたコンピュータ、と言えるでしょう。テレビのリモコンを作るのにパソコンは必要ないでしょうし…

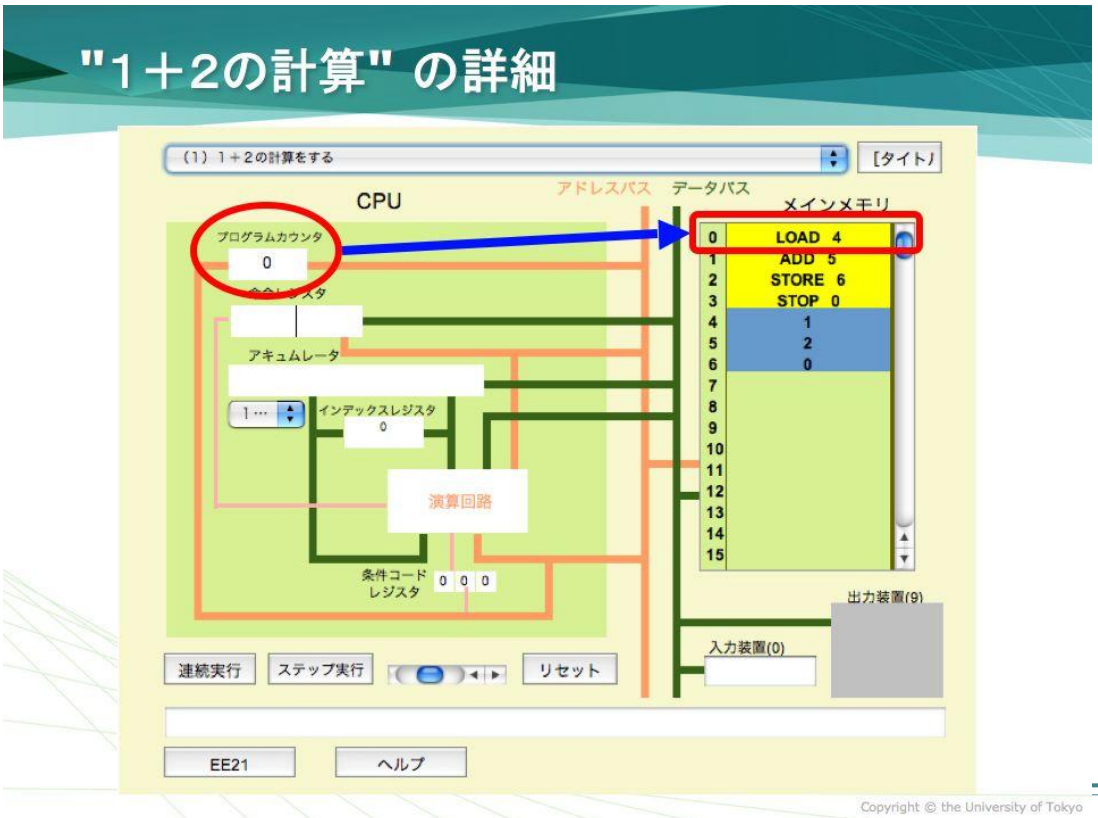
コンピュータの基本構成（参考　7.1.1）

このコンピュータの構造（アーキテクチャ）は一つのメモリの中にプログラムとデータが混在しています。これをフォンノイマン型



コンピュータと呼びます。

コンピュータはプログラムを見て計算をしていきます。そのプログラムは機械語（０と１の羅列）で書かれています。しかしこれを人が扱うことはとても難しいです。そこでこの機械語の命令を英単語に置き換えたものをアセンブリ言語と呼びます。以下の説明ではこのアセンブリ言語を使ってプログラムが書かれています。



動作（7.1.2　7.5.1）

まずはこのシミュレータの各部分の説明。

- 演算回路…おなじみ ALU などの計算を行う回路です。
- メインメモリ…プログラム（ALU がどのような計算をすべきかの命令群）やデータ（計算を行うための）を格納してあります。格納部分一つ一つにはアドレスと呼ばれる番号が振られていて、それを指定することでメモリの内容を参照します。倉庫みたいな感じ。
- プログラムカウンタ…メモリの何行目の命令を実行しているかを示すカウンタです。命令が 1 つ終了したらカウンタが一つ増えて次に移ります。
- 命令レジスタ…今実行するべき計算が何かを保存しておく場所です。アセンブリで命令｜参照すべきアドレスの順で書かれています。
- アキュムレータ…ワーキングメモリというマイコンもありますね。今行っている命令でつかう数字を保存し、この数字に数値を加減していきます。
- その他…今回は関係ないみたいです。

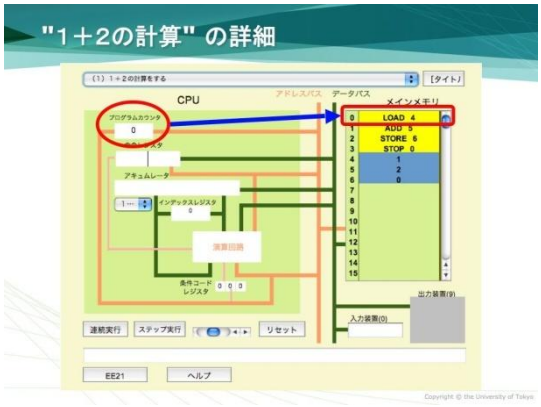
それでは実際に動かしましょう。
以下コピペです。つっこみどころがないのです。

2. "１＋２の計算" の詳細

"１＋２の計算" をするプログラムが，実際にどのように動いているのか，詳しく見てみましょう．

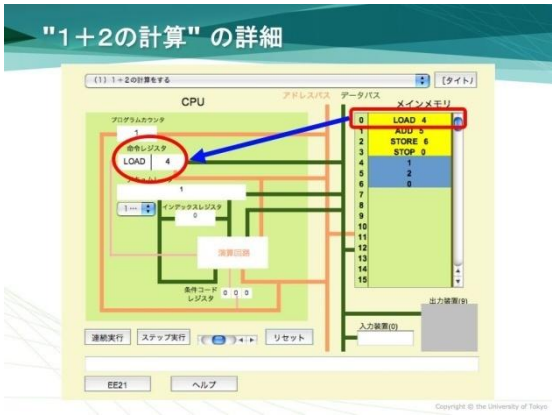
1. "プログラムカウンタ" を見て，"メインメモリ" 上の命令をとりこむ

"プログラムカウンタ" の値にしたがって，"メインメモリ" の 0 番地の命令をとりにいきます．



2. "命令レジスタ" 上の命令 "LOAD 4" を実行する

"メインメモリ" の 0 番地にあった命令 "LOAD 4" が "命令レジスタ" にとりこまれ，この "LOAD 4" を実行します．

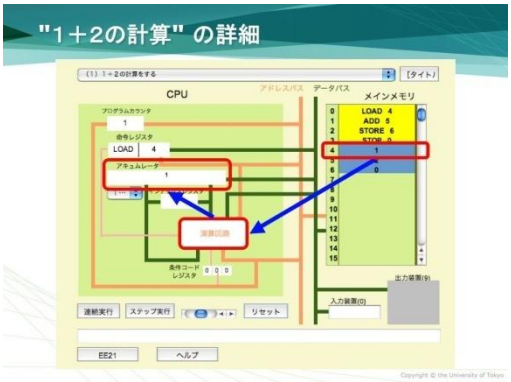


3. "LOAD 4" の結果を"アキュムレータ" に格納する

LOAD 命令は

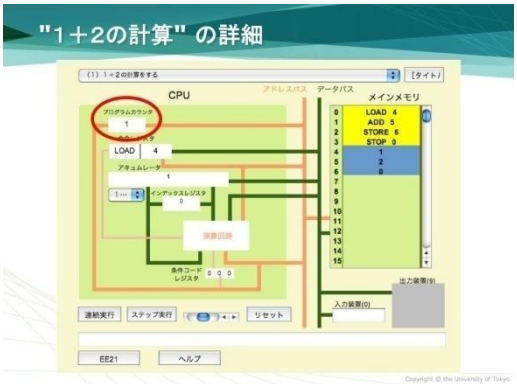
- LOAD n: n 番地の内容をアキュムレータにコピーする

という命令です．ここでは "LOAD 4" なので，4 番地の値である "1" を "アキュムレータ" にコピーします．



4. "プログラムカウンタ" の値を 1 増やす

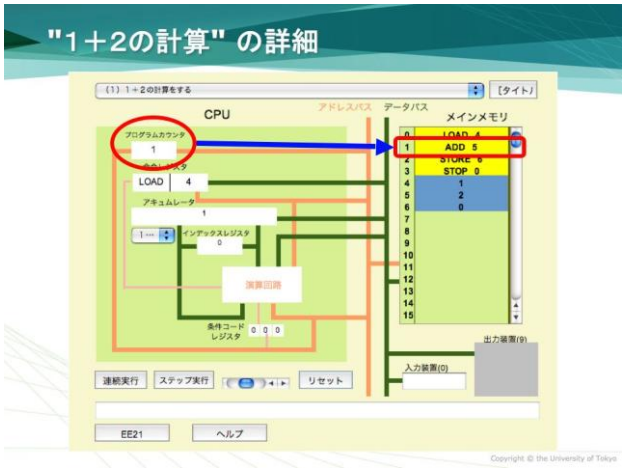
"プログラムカウンタ" の値を 1 つ増やします．



"ステップ実行"の最初のクリックで，ここまで実行が進みます．上の 2 と 3 の画像では，プログラムカウンタの値が"1" になっていますが，これは画像を取り込むための都合によるもので，本来ならここで初めて "プログラムカウンタ" の値は "1" になります．

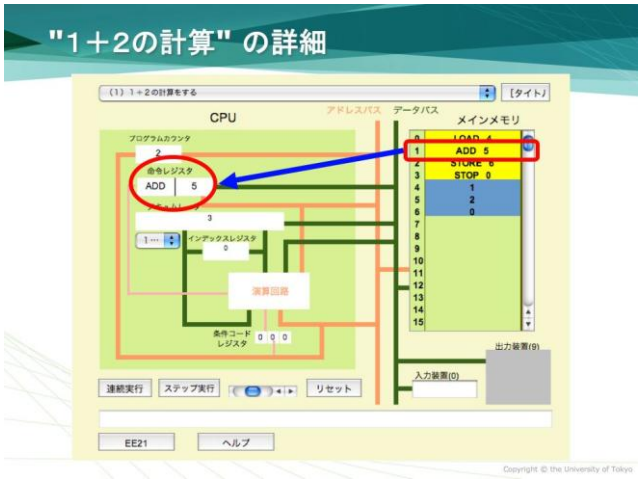
5. "プログラムカウンタ" を見て，"メインメモリ" 上の命令をとりこむ

"プログラムカウンタ" の値にしたがって，"メインメモリ" の 1 番地の命令をとりにいきます．



6. "命令レジスタ" 上の命令 "ADD 5" を実行する

"メインメモリ" の 1 番地にあった命令 "ADD 5" が "命令レジスタ" にとりこまれ，この "ADD 5" を実行します．

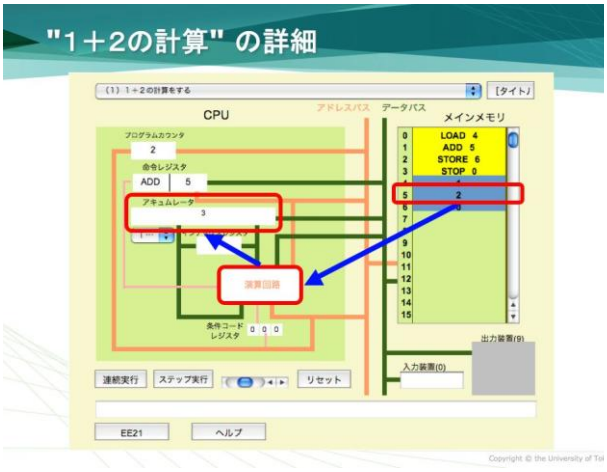


7. "ADD 5" の結果を"アキュムレータ" に格納する

ADD 命令は

- ADD n: n 番地の値をアキュムレータの値に加える

という命令です．ここでは "ADD 5" なので，5 番地の値である "2" を "アキュムレータ" の値である "1" に加えた値，すなわち "3" を "アキュムレータ" に格納します．



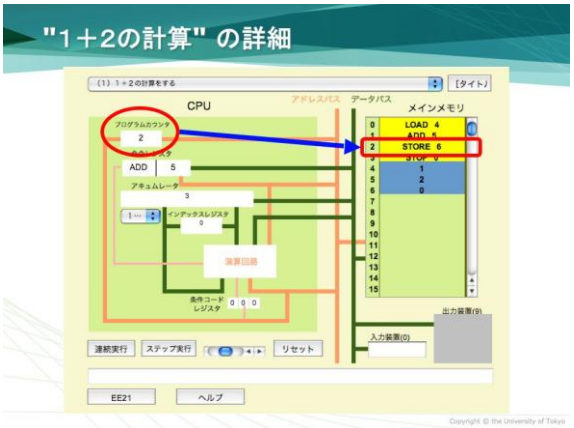
8. "プログラムカウンタ" の値を 1 増やす

"プログラムカウンタ" の値を 1 つ増やします．

"ステップ実行"の 2 回目のクリックで，ここまで実行が進みます．

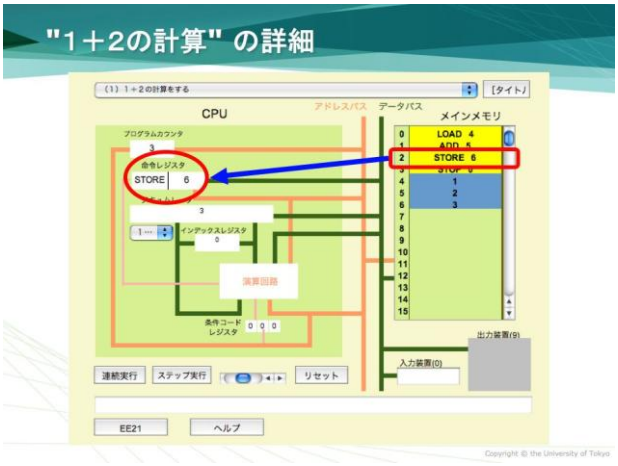
9. "プログラムカウンタ" を見て，"メインメモリ" 上の命令をとりこむ

"プログラムカウンタ" の値にしたがって，"メインメモリ" の 2 番地の命令をとりにいきます．



10. "命令レジスタ" 上の命令 "STORE 6" を実行する

"メインメモリ" の 2 番地にあった命令 "STORE 6" が "命令レジスタ" にとりこまれ，この "STORE 6" を実行します．



11. "STORE 6" により，メインメモリに"アキュムレータ"の値を格納する

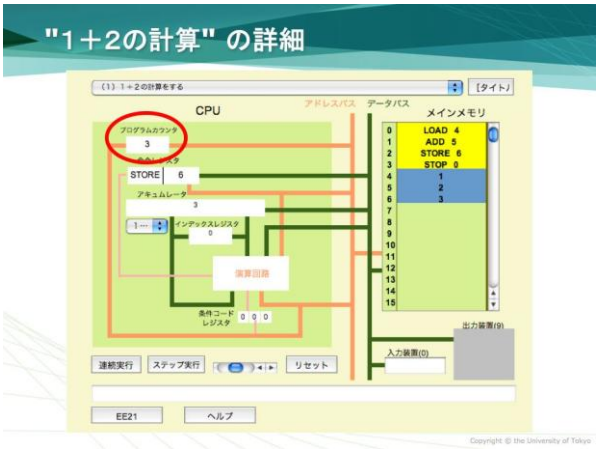
STORE 命令は

- STORE n: アキュムレータの値を n 番地に格納する

という命令です．ここでは "STORE 6" なので，"アキュムレータ" の値 "3" を 6 番地に書き込みます．

12. "プログラムカウンタ" の値を 1 増やす

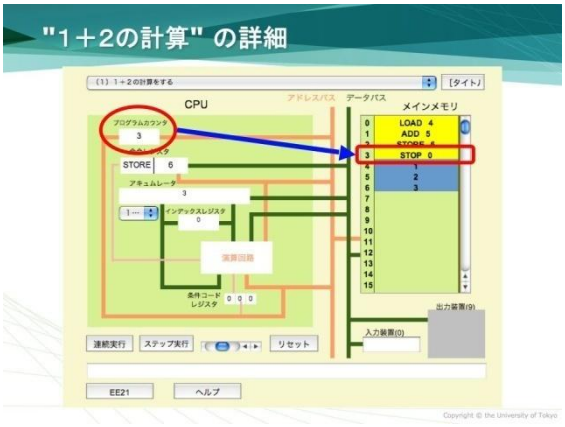
"プログラムカウンタ" の値を 1 つ増やします．



"ステップ実行"の 3 回目のクリックで，ここまで実行が進みます．

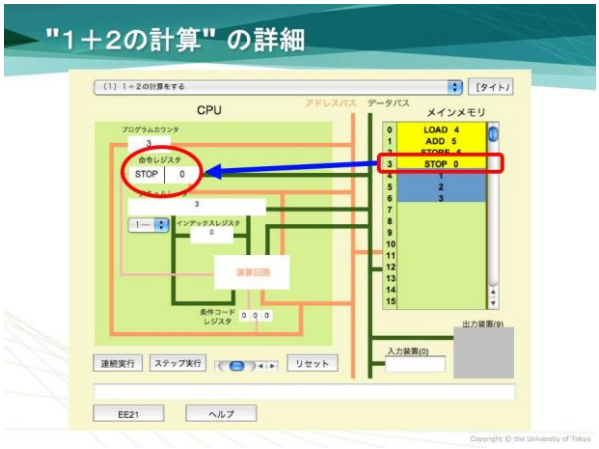
13. "プログラムカウンタ" を見て，"メインメモリ" 上の命令をとりこむ

"プログラムカウンタ" の値にしたがって，"メインメモリ" の 3 番地の命令をとりにいきます．



14. "命令レジスタ" 上の命令 "STOP 0" を実行する

"メインメモリ" の 3 番地にあった命令 "STOP 0" が "命令レジスタ" にとりこまれ，この "STOP 0" を実行します．



STOP 命令は

- STOP n: プログラムを終了する

という命令です．ここでは "STOP 0" となっていますが，0 以外の数字でもプログラムは終了します．"ステップ実行"の 4 回目のクリックで，ここまで実行が進んで，プログラムの実行は終わります．

プログラム実行の流れ

このようにプログラムの実行は，以下の 1～3 のサイクルを繰り返します．

2. "プログラムカウンタ" の値を見て，"メインメモリ" 上の命令をとりにいく
3. "命令レジスタ" 上にある命令を実行する

主な命令は以下のとおり

- アキュムレータに値を読み込む
- アキュムレータの値とメインメモリの値で演算を行う
- メインメモリにアキュムレータの値を書き出す
- プログラムの実行を終了する(この場合，実行のサイクルは止まる)

4. "プログラムカウンタ" の値を 1 増やす

使われている命令のまとめ

ここまでに利用された命令をまとめると，以下のとおりです．

- LOAD n: n 番地の内容をアキュムレータにコピーする

- ADD n: n 番地の値をアキュムレータの値に加える
- STORE n: アキュムレータの値を n 番地に格納する
- STOP n: プログラムを終了する

長いですね。いやですね。実際に ED21 シミュレータを使うことをお勧めします。

…この先は説明しなくてよいでしょう。シケ対の仕事丸投げですが、長くなる＆どうせコピペ（解説の完成度が高い）＆どーせ出ない。なので。

URL は http://lecture.ecc.u-tokyo.ac.jp/johzu/joho/Y2008/ED21/ED21/index.html です。

実際のコンピュータ(7.6)

ここからは中村スライドが中心です。比較的わかりやすいです。

ハードウェア構成(7.6.1)

パソコンの中身は一番上のようになっています。基本構造はマイコンとあまり変わりません。

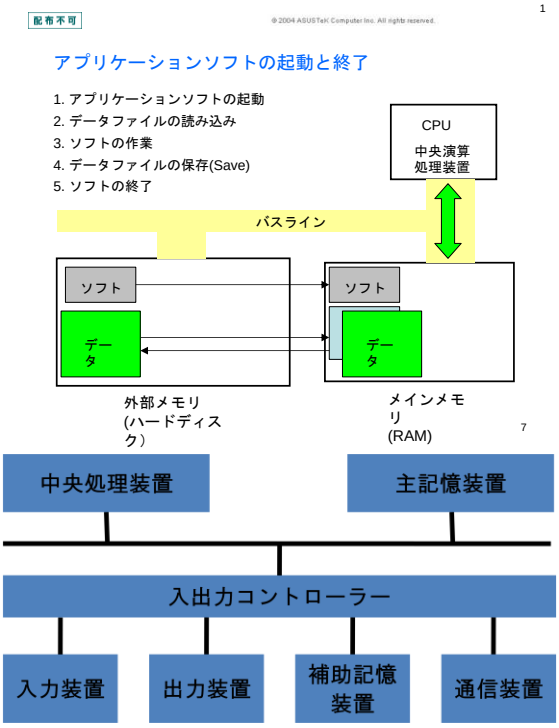
中央処理装置は CPU、主記憶装置はメモリ、入力装置はキーボード・マウス、出力装置はディスプレイ・プリンタ・スピーカ、補助記憶装置はハードディスク・USB メモリ、通信装置は LAN などのネット接続装置が主な例です。

各部分をつないでいる線…これはバスと呼ばれます。一定の形式・規格に従って各装置が通信を行う線です。

オペレーティングシステム（7.6.2）

なんだか飛躍しすぎ…と思うのは自分だけではないはず。そう思ったのがきっかけで再構築したわけですが。。

写真で見るマザーボード



オペレーティングシステム（OS）というのは基本ソフトと呼ばれます。確かに各アプリケーションで入力から出力までをすべてやってしまってもよいかもしれません。しかしそれをやるには、例え文字を打つだけでも多くの労力とプログラムがひつようになります。そこでどのソフトでも必要になるような基本的な機能を提供しよう、というのが OS です。

OS は基本的な機能の提供だけでなく PC の資源管理など、さまざまなサービスを提

供しているのです。以下 OS が提供している機能を上げていきます。

- プロセス管理…現在の PC では様々な処理を同時に提供します。例えば、ブラウザを使いながらワードをつかったり。しかし CPU で処理できるのは瞬間で１つしかありません。そこで OS は複数の処理を分割し（タイムスライシング）、スケジュールを作って実行します。
- メモリ管理…主記憶装置上にのる情報は有限です。そこで効率よく使えるように割り当て（どのデータをどの部分に保存するか）、解放（要らないデータを消去）を行います。
- 入出力管理…キーボードやマウス・さらには拡張機器の管理を行います。プロセスと周辺装置の仲介
- ファイル管理…階層的なファイルの管理を行うためのファイルシステムを提供します。
- ユーザー管理…一つのパソコンをみんなで使うための管理。これは普段行っているのでわかるでしょう。

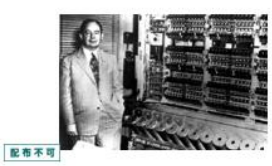
第７章の最後に開発の歴史です。何故かしら中村スライドにはあります。

開発の歴史

計算の実現機構

- プログラム内蔵方式（フォンノイマン型コンピュータ）
  - メモリ上にデータとプログラムを保持
  - 万能チューリング機械（6.2.1項）と同様に他の計算機を模倣できる

  ⇔最も初期のコンピュータ(ENIAC,1946)はプログラムを配線していた



配布不可

コンピュータの歴史

1970 年アラン・ケイ「パーソナルコンピュータ」を提唱。「コンピュータ・リテラシ」も彼の造語。**ダイナブックを具現化。ハードウェア Alto, 暫定的ソフト Smalltalk .**

（アラン・ケイはコンピュータとは高価でみんなで共有するもの、という概念が一般的だった時代に理想のパーソナルコンピュータを提唱した。ケイが XEROX のパロアルト研究所在籍時に「暫定ダイナブック」として開発したのが、Smalltalk を GUI ベース(Graphical User Interface; アイコン・マウスなどが使用されているので直感的に操作しやすい)の オペレーティングシステム に用いて動作させていた Alto で、これを目にしたスティーブ・ジョブズが Lisa 、そして Macintosh を開発するきっかけとなったとされる。ケイの論文の「ダイナミックパーソナルメディア（動的個人媒体)」を使う、「本」のようなもの。から Dynabook と名前が付けられた。）from Wikipedia

1976 年 Apple I(スティーブ・ジョブズが私財をなげうって作った愛好家用コンピュータ)翌年 Apple II 大成功を収める。

(世界で初めて個人向けに完成品コンピュータとして大量生産・大量販売されたパーソナルコンピュータである。本体はキーボードと一体化した形状であり、後にパーソナルコンピュータの標準的な形態となったキーボードと本体が分離しているセパレートタイプ（当時はデタッチャブルタイプと呼ばれた）ではない。この形状はポータブルタイプライターをイメージしたという。）from Wikipedia

1970 年代後半～80 年台前半　多くの非互換機メーカーが競合

(今のようにどのメーカーの PC でも WINDOWS が入れられるわけではなく、コンピュータの規格が異なるが故メーカー独自の OS が使われていた。)

1981 年 IBM PC 16 ビットコンピュータ　PC/AT 互換機が業界標準、アップルは非互換機路線を堅持

16 ビットというのはコンピュータが一度に 16 ビットの情報を計算できるということ。IBM が PC/AT というコンピュータを発表して以来全世界に広まり、業界標準としての地位を確立した。しかしアップルは従来のスタイル、つ

まり独自のマシーン規格を貫いた。今でもこれは続き、MAC 機に WINDOWS は基本的には入れられない。また WINDOWS 機に MAC OS はインストールできない。)

2004 年 **IBM はパーソナルコンピュータ事業を中国のレノボ・グループに売却。**ハードウエアのオープンアーキテクチャ化を大きな要因として繁栄した PC/AT 互換機であったが、最終的にはその互換性によって市場から撤退することとなった。

(PC/AT 互換機の普及の最大の要因は規格を公表したこと。今ある WIN 機は PC/AT 互換機だが、改良の結果初期の PC/AT と物理的な互換性はない。)

国内の歴史

「国産マイコン」の最初の製品は、1976 年日本電気 (NEC) より発売された TK-80 **とされる。**

(初期のマイコン。数字を表示する LED と 16 進数を入力するキーしかなかった。開発者はアセンブリ言語を自分で機械語に翻訳し入力する必要があった。一度間違えるとカオス。しかし安価・単体で使用可能であったがゆえ多くの愛好家に購入された。今の日本のコンピュータ技術開発者はこれで育ったといってもよいほどの影響力を持った。)

その後 8 ビットマイコン・BASIC と群雄割拠の時代

(BASIC というのはコンピュータのソフトを開発するための言語。いまでも健在で、Visual Basic として残っている。また数学 B の教科書に載っているのも BASIC。文法が英語にとても近くわかりやすい。)

1982 年　16 ビット CPU を採用した「PC-9800 シリーズ」が発売。**その後は** MS-DOS を採用した PC-9800 シリーズの独走態勢となった。

(PC-9800 は NEC から発売されたコンピュータ。ほかに富士通などが PC を発表していたが大ヒット。今でも秋葉原で互換機が売られているらしい)

Macintosh は、漢字 Talk7 が発売された頃から、**ハードウェアの値下げと日本語処理機能の充実により、**シェアを伸ばしていった。(漢字 TALK7 は今でもネット上にある。GUI を採用した直感的な操作性と日本語入力が大ウケ)

1995 年に GUI を大改良した Windows 95 の発売が開始されると、98 互換機のエプソンも PC/AT 互換機に転換し、国内独自パソコンの歴史は完全にピリオドが打たれた。

(結局最後は PC/AT 互換機+WINDOWS の組み合わせにやられたわけです。ちなみに最新の WIN の世界シェアは 91%　MAC は 8%で Linux は 0.5% 第 4 の OS は iPhone らしいです。さすがは Apple。しかし個人的には Linux を応援したい;
http://journal.mycom.co.jp/news/2008/02/05/045/index.html)

…まさか出るまい。こんなの出題して何になる？

以上で前半部分終了です。お疲れ様でした。

長い。そしてマニアック。仕方がない…のか？シケ対のシュミが多分に含まれているのは気のせいだろうか。

それでは次。

2．WWW・メイルシミュレータ(6/20)(教科書　3.2)

以前にクラスブログにアップしたやつを焼き直します。

WWW シミュレータ

（http://media.itc.u-tokyo.ac.jp/jsim/src/jsim/mail_www/mailsim.htm　）
ITpro(Web アクセスの基本):
http://itpro.nikkeibp.co.jp/article/COLUMN/20060215/229453/
総務省：
http://www.soumu.go.jp/joho_tsusin/security/kiso/k01.htm

・URL 入力　（教科書第３章　3.4.7）

WWW ブラウザ（つまりユーザー自身）が URL（例えば http://host.b.jp/index.html）を送信すると、まず DNS サーバーにその情報が送られます。
DNS(Domain Name System)というのは、インターネット上のサーバー(例 host.b.jp)にはそれぞれ番地のようなもの(IP アドレス　例 192.168.0.2)が割り振られています。しかしそれでは人間は扱えないので英語などで名前を割り振っておきます。それが DNS システム。
URL→DNS サーバー→IP アドレス　の順。これでユーザーは IP アドレスがわかりました。つぎはこれを使ってホームページにアクセスします。

・HTTP リクエスト

WWW ブラウザはホームページを見るために相手の IP アドレスや自分の IP アドレス、さらに欲しいもの(ページファイル)を送信します。HTTP については後述のはず。
ユーザーが送るとサーバーからメッセージヘッダ+ページファイルが送られてきます。ヘッダとは(ほんとはリクエスト行ってものもあります。１行目のみ。別にいいでしょう)にはサーバーの IP など基本情報が書いてあります。小包に送り主の情報・受け取り主の情報が書いていないと困るでしょ。これをメッセージヘッダといいます。
こうしてユーザーはページの中身を手に入れられました。次はユーザー側のソフトで表示します。

・ページ表示

先ほどのページファイルは中身の文字情報のほかに HTML タグ(今は XHTML 等色々な種類があります。)も含まれています。HTML タグとは文書をどのように配置するか、飾り付けるか…のことです。まぁ来週の実習でやるでしょ。それを WEB ブラウザ(例えば今使っている IE 7 とか、Firefox とか)が解読して表示します。

・Cookie

…は今回あまり重要じゃないですね。
Cookie とは WEB ブラウザに mixi にログオンするときのパスワードなどを覚えさせるものです。

・キャッシュ

…もあまり重要ではありません。何で入れたんだろう。。一回みたページはまた見る可能性が高いのでとっておくのです。

・TCP/IP（パケット通信）

プロコトルの一種。プロコトルとはネットワークを介してコンピュータ同士が通信を行なう上で、相互に決められた約束事の集合。通信手順、通信規約などと呼ばれることもあります。コンピュータ同士で話をするときに文法などを決めておかないと困りますよね？それと同じ。HTTP もこの一種。

で、ユーザーから何か出てくるのがパケット。パケットとは送信する情報を細かく切ったもので、ヘッダ（前述）とデータからなります。小さくすることで複数の端末からのデータを１本の信号線上に混在させることができ、帯域の分割利用が可能になるなどの利点があります。
パケットで小分けされたデータは１度に送ろうとします。しかしインターネット上で何

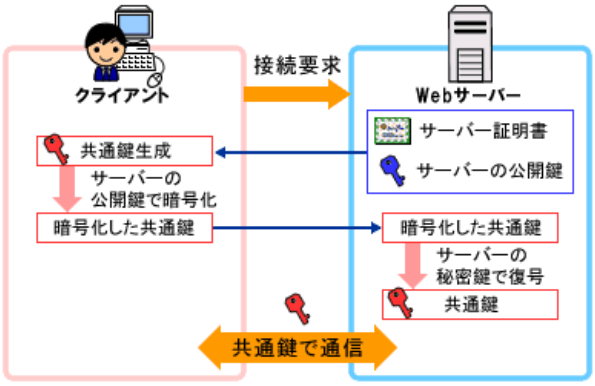
らかの理由により一部が正常に受信されないことがあります。そこで受信先は送り主に情報の再送を要求します。これによってすべての情報がそろいました。

・SSL （間違いがあるので読まないでください。すみません m(__)m）

長いです。がんばりましょう。メイルについてはまたのちほど。SSL とは HTTP 通信を暗号化して送受信する業界標準の通信プロトコル。ここでは公開鍵暗号方式が使われています。

公開鍵暗号方式とは公開鍵・秘密鍵・共通鍵の 3 種類があります。まずにサーバーから公開鍵（この正体は第三者機関が発行した電子証明書）を受け取ります。この公開鍵は第 3 者(CA)が保障します。

共通鍵はログオンするときのパスワードだと思ってください。CA が証明すべきファイルの内容が改ざんされていないことを証明したら公開鍵で共通鍵を暗号化・送信。パスワードを裸のまま流すとばれてしまいますので。サーバーはこの共通鍵を受信したら暗号化された情報をユーザーに送ります。
秘密鍵はというと…データを暗号化するための鍵。これをユーザーに教えてしまっては改ざんが容易ですよね。



粗相をお許しください。以下石井さんの解説…

この方式は、公開鍵暗号方式と共通鍵暗号方式を組合わせて使ってる。なぜかと言うと、そうするとお互いに相手を認識できるから(公開鍵だけだと、誰かが送信者になりすますことが出来てしまう)。それと、単に共通鍵の方が楽だから(PC の負荷的に)。で、この通信の流れなんだけど、まずは公開鍵暗号方式の通信をする。CA ってのは認証局のことで、届いた公開鍵が本物がどうか確認してくれるんだ。で、公開鍵の確認ができたら、メッセージとして共通鍵を暗号化して送る。こうすることで、誰にも盗聴されずに両者が共通鍵を共用することが出来て、それ以降は共通鍵暗号方式での通信が出来る、ということ。

以上です。ハイブリッド暗号方式のようです。

ここからは対応する第 3 章の解説。

3.2.2　プロコトル

(a)プロトコルの必要性

プロコトル…ネットワークを介してコンピュータ同士が通信を行なう上で、相互に決められた約束事の集合。通信手順、通信規約などと呼ばれることもある。

英語しか使えない人と日本語しか使えない人では会話ができないように、対応しているプロトコルが異なると通信することができない。

(from e-words)

(b)アプリケーションとプロトコル

我々サービスを受ける、つまり情報を要求する側のことをクライアントと呼び、その要求に応じて情報を送信する側をサーバーと呼ぶ。WWW においては HTTP というプロトコルが採用されている。

HTTP とは要求と応答で構成される単純なプロトコルである。代表的なやりとりとはクライアントが要求した文章をサーバーが返す場合で、(1)クライアントからサーバーに目的文章の URL を送信する、(2)サーバーがその文章を返す、という通信が行われる。文章の前にはヘッダと呼ばれるメッセージがつけられる。(教科書　P.47)

(c)プロトコルと階層

インターネットで現在使われている TCP/IP プロトコル群は階層的な構成が採用されている。階層化の利点は

(1)基礎的な通信部分を分離することで異なる通信メディア(ADSL や無線 LAN)でも WEB 閲覧の際は HTTP プロコトルを共有できる。

同じ下位プロコトル(例えば物理層とか)を共有することで SMTP などメールプロコトルなども使用できる。
(2)ユーザーが通信のために煩雑な手続きが取られていることに気づかずにすむ。開発者の優しさ。

プロコトルの 7 階層(3.4.2　e-word より　詳しくは教科書)

第 1 層(物理層)は、データを通信回線に送出するための電気的な変換や機械的な作業を受け持つ。ピンの形状やケーブルの特性なども第 1 層で定められる。

第 2 層(データリンク層)は、通信相手との物理的な通信路を確保し、通信路を流れるデータのエラー検出などを行なう。

第 3 層(ネットワーク層)は、相手までデータを届けるための通信経路の選択や、通信経路内のアドレス(住所)の管理を行なう。IP

第 4 層(トランスポート層)は、相手まで確実に効率よくデータを届けるためのデータ圧縮や誤り訂正、再送制御などを行なう。TCP

第 5 層(セッション層)は、通信プログラム同士がデータの送受信を行なうための仮想的な経路(コネクション)の確立や解放を行なう。

第 6 層(プレゼンテーション層)は、第 5 層から受け取ったデータをユーザーが分かりやすい形式に変換したり、第 7 層から送られてくるデータを通信に適した形式に変換したりする。

第 7 層(アプリケーション層)は、データ通信を利用した様々なサービスを人間や他のプログラムに提供する。HTTP

3.2.3　通信の秘密と相手の認証

(a)共通鍵と暗号鍵方式

暗号化する前の元の文章を平文(ひらぶん)、暗号化した分を暗号文と呼び、暗号文から平文に戻すことを復号と呼びます。暗号化は計算手順と鍵を用いて行います。共通鍵暗号方式と公開鍵暗号方式があります。

共通鍵暗号方式は暗号化と復号を共通の鍵で行う方式です。処理が早いなどの利点はありますが、共通鍵が盗聴されてしまうと漏えいしてしまう、相手ごとに鍵を渡さなければいけない、など欠点があります。そのため特定の相手との通信に向いています。また、自分の PC 内で暗号化する場合にも向いています。「文字を 3 文字ずらす」といったのも共通鍵暗号方式の一例です。初代暗号方式は DES・AES が有名です。

公開鍵暗号方式は受信する人が 2 組の鍵をつくり、片方を暗号文を作るときに用いる鍵として公開し（公開鍵）、もう片方を自分が復号するために保存しておきます（秘密鍵）。こうすることで、例え暗号文や公開鍵が流出していても復号できるのは自分の秘密鍵のみです。しかし、暗号化する際の計算時間が長くなってしまうなどの問題があるので、平文を共通鍵で暗号化し、その共通鍵を公開鍵暗号方式で送るというハイブリッド暗号が使われています。

(b)ディジタル署名

ディジタル署名とは電子的な署名の技術です。この方法の一つに一方向ハッシュ関数というのがあります。この一方向ハッシュ関数というのは公開関数ですが、関数で処理された結果（電子指紋；データの特徴）からもとのメッセージに復号することは現時点では絶対にできない点で優れています。この電子指紋を秘密鍵で再び暗号化することで署名が生成されます。

復号するときはメッセージと署名を受信します。メッセージを一方向ハッシュ関数で電子指紋 1 を作成し、それとは別に署名と公開鍵から電子指紋 2 を復号します。この 2 つを比較し一致していれば改ざんされていません。なぜなら署名をつくることができるのは秘密鍵を持つ者だけだからです。

このディジタル署名の重要な用途は公開鍵への署名です。自分が受け取った公開鍵が意図した相手の発行したものかどうかを判断するためです。

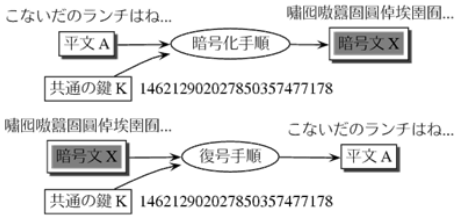
3.3.3 通信の秘密と相手の認証

- 共通鍵暗号（対称鍵暗号）
  - –送信する暗号化での鍵と受信での復号化の鍵が同じもの。(シーザー暗号など)
- 公開鍵暗号（非対称鍵暗号）
  - –暗号化と復号での鍵が違うもの(RSA方式など)

1

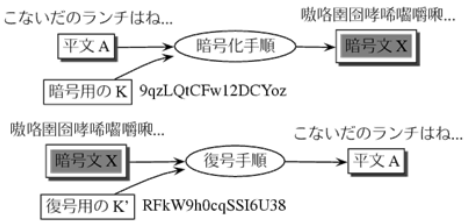
共通鍵暗号

- 一つの鍵で暗号化と復号化が両方できるモデル　　鍵を秘密に保つ必要がある



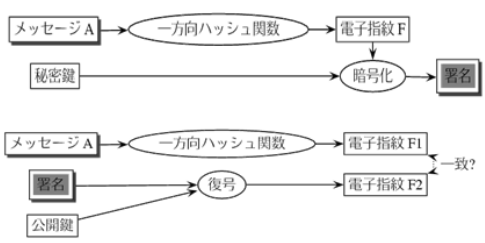
37

公開鍵暗号



38

署名と検証



39

メイルシミュレータ

以下コピペ。

メイルシミュレータ：

http://media.itc.u-tokyo.ac.jp/jsim/src/jsim/mail_www/mailsim.htm

・電子メールアドレス

アドは　abc@def.com　などとなっています。これは２つの部分に別れ、　[アカウント名]@[サーバー名]　という意味をもちます。何とかマンションの誰々さん、みたいな。

・IMAP,POP,SMTP

これは電子メイルを送受信するプロコトル。SMTP はサーバー間・サーバーとユーザー間で送信するために使います。IMAP は Outlook などのメールソフト、POP は WEB メイルにおいて使うものだと思っていてください。

・送信

いま user1@host.a.jp から user3@host.b.jp メイルを送る場合を考えましょう。メイル送信フォームが出てきます。このフォームの To,CC,Bcc について。

To というのは主な送信先。CC というのはカーボンコピーのことです。同じ内容を指定した先に送信できます。Bcc とはブラインドカーボンコピーのことで、CC と同じような機能ですが、そのとき受信先にはそのほかの受信先が見えないというのが大きな特徴です。

host.a.jp は user1 から送られたメイルのヘッダを解読して host.b.jp に送信します。

・受信

受信先の user3@host.b.jp は host.b.jp に対してサーバーにメイルが届いていないか受信問い合わせをします。さきほど user1 が送ったメイルがサーバーにあるのでそれを受信して終了。

…こんな感じです。あ、気づきましたかね？この過程で一切の暗号化がされていません。つまり誰でも読もうと思えば読めます。電子メイルのセキュリティーははがき程度だといわれています。

やっぱりどこか間違っている気がします。訂正があればお知らせします。

3．オートマトンシミュレータ(6/27　6.2.1)

http://lecture.ecc.u-tokyo.ac.jp/johzu/joho/Y2008/Automaton/Automaton/

6.2.1　機械的な計算モデル

以下引用

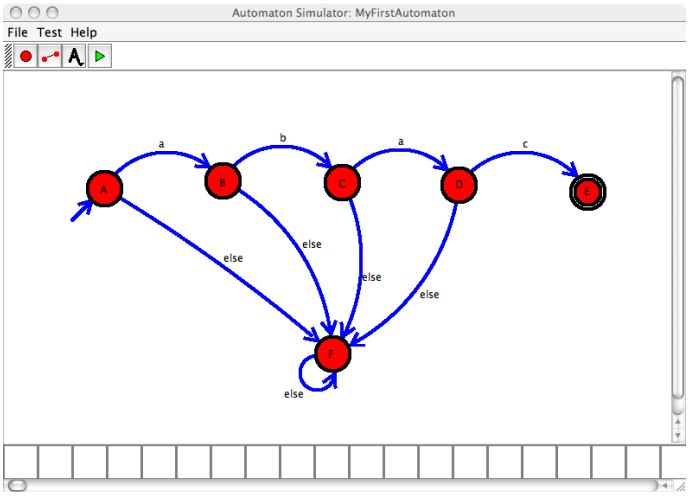
オートマトン (automaton (pl: automata)) とは「自動人形」を意味している言葉で、情報科学の分野においては、次のような特徴を持ったシステムのことである。

- 外から、連続している情報が入力される
- 内部に「状態」を保持する
- 外へ、何らかの情報を出力する

携帯電話を例にとると、キーを押すことによってさまざまな機能が使用できるが、その機能はキーと必ずしも 1 対 1 で連動しているわけではない。「5」のキーを押すことにより、ある場合には画面に 5 が現れるが、ほかのある場合には「な」が現れる。あるいは画面上のキャラクターが行動したりもする。これは今までに入力された情報によって内部の状態が変化しているからである。このように入力がなされた時点での「文脈」に対して複雑な解釈を行うような仕組みをオートマトンという。from Wikipedia

(a)有限状態機械

内部の状態および入力が有限個であるものを有限状態機械といいます。教科書では「１」「２」「３」という３つのボタンをもち、「abac」と打つと解錠できる電子錠が紹介されています。以下はオートマットシミュレータで書いた電子錠プログラム。仕方なく「a」



「ｂ」「ｃ」の３つのボタンとしています。

まぁオートマット実習の目的がよくわからないので　http://lecture.ecc.u-tokyo.ac.jp/johzu/joho/Y2008/Automaton/Automaton/　から試してください。遊んで実感してください、という話でしょう。

状態としては「これまでに正しいボタンがｎ個正しい順序で押された」「ボタンを押し間違えた」で表すとします。前者には「ボタンが押されていない」から「４つ正しい順番で押された」まで５つの状態(Ａ～Ｅ)があります。後者をＦとします。初期状態はＡであり、「１」「２」「１」「３」とこの順番で押されることで状態Ｅ（解錠）に移ります。それ

いがいの順序で押された場合は状態Ｆ（間違い）に移ります。

アルゴリズムとして示すと・・・

状態←最初の状態

While 状態≠最終状態 Do

ボタン←押されたボタン

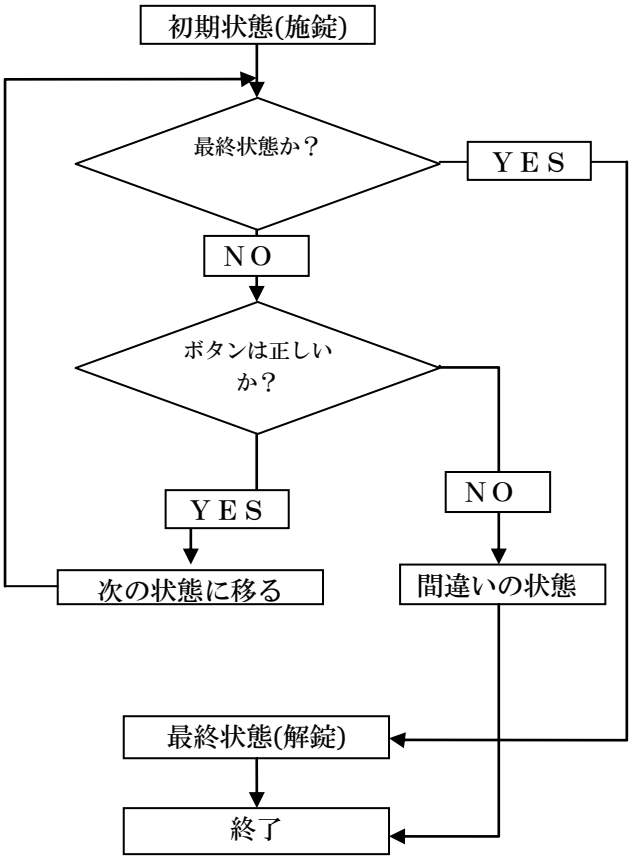
表から(状態、ボタン)に対応する次状態を引く

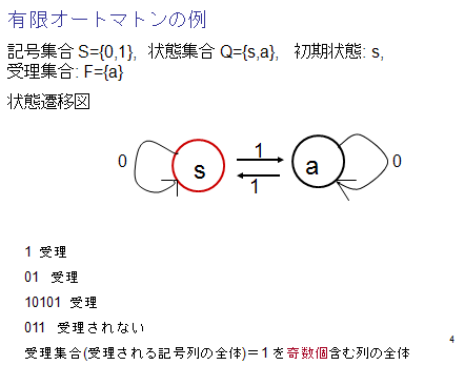
状態←次状態

Done

…なんのことやら。図で示します。

この有限状態機械「有限」は機械の中にある状態が有限であるがゆえ、扱える問題が限られているといえます。

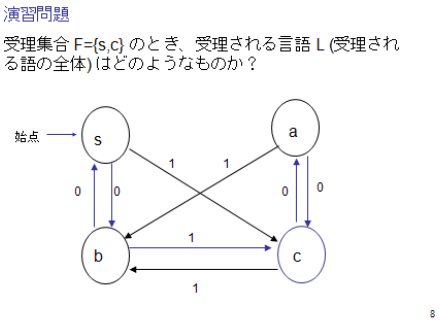




記号集合…「０」と「１」の二つの状態があること

状態集合…「Ｓ」と「Ａ」の２つの状態があるということ

受理状態…「Ａ」がゴール、最終状態であるということ。

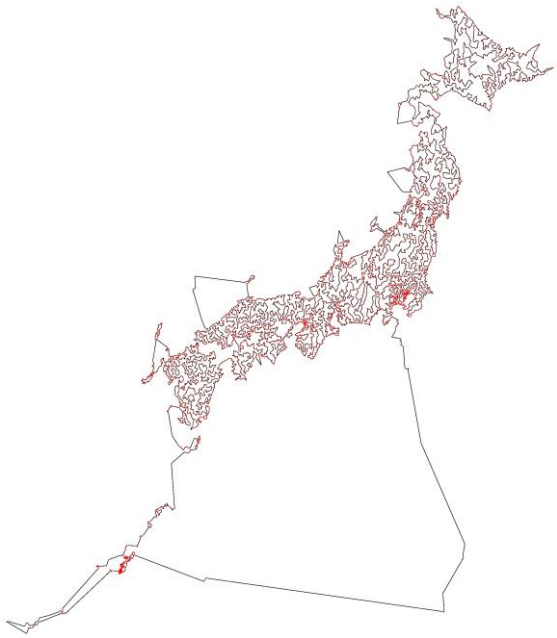


…誰か解けたら教えてください。数学が残念なので解けません(涙)

４．Travelling Salesman ゲーム（6/27　4.3.2）

セールスマン問題

路線図を考えてみましょう。出発地と目的地が決まっているとき、どの路線を使いどの駅で乗り換えれば最短なのでしょう？これを巡回セールスマン問題といいます。如何にしたら効率よく最短経路をたどれるか、という問題です。しかし「出発地→目的地」の所要時間をすべて求め比較するのには無限の時間がかかります。これを NP 困難といいます。このような問題は、厳密な最短経路を求めるのではなくできるだけ良い近似解を求める方法が研究されています。



日本全国を一筆書きするとこんな感じだそうです。中村的には「おもしろい」、と。

5．補足

・ハミング距離(07 個別問２)

ここからは過去問から気になるところを解説しました。これら以外にもわからないところがあったらごうしょうまで。まぁ自分もお手上げな場合があるので…

通信過程において、ノイズや干渉によって、どうしても送信データが正確に遅れないことがあります。それを補正するために作られたのがハミング距離です。具体例を見ていきましょう。

元のデータ W	1	F	3	9
２進数	0001	1111	0011	1001
受信結果 X	1	F	2	1
２進数	0001	1111	0010	0001
受信結果 Y	9	E	2	0
２進数	1001	1110	0010	0000
受信結果 Z	C	1	F	6
２進数	1100	0001	1111	0110

あ、E とか F とかっていうのは１６進数です。

10 進数、2 進数、16 進数対応表

10 進数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
16 進数	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10

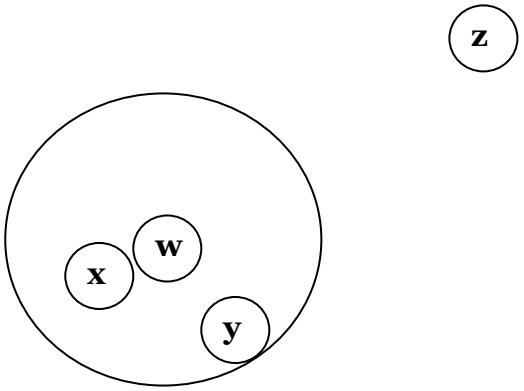
まぁあまり気にしなくていいです。下の２進数でお話をしますので。これらの３つの受信結果を受け取ったとしましょう。このとき２進数のレベルで元のデータと比較します。このとき、２進数のレベルで何ビットが異なるかを表したのがハミング距離です。

d(W,X}=2

d(W,Y)=5

d(W,Z)=12

実際には受信結果同士を比較するらしいのですが、説明が簡単なので元のデータと比較します。この結果をみると、X が一番忠実で Z が一番離れていると考えられます。この結果をもとに元のデータを復元していきます。



データ w を中心として、ハミング距離を半径として配置します。そして一定距離のなかにあるデータをもとのデータにノイズが混じったものだと見なします。この一定距離の中にある x,y を比較して、元のデータ w を復元します。

…長くなりました。定義すると

$$d(x,y)=\sum_{i=1}^{n} x_i \oplus y_i$$

です。やっていることは今までの説明と同じで、数式化しただけです。なお排他的論理和⊕は２つのビットを比較して同じならば０を、異なれば１を返します。では次の問題。

問 2 (b)

記号列 x,y のハミング距離を h(x,y)と書くとき、任意の記号列 x、y、z で h (x,y)≦ h (x,z)+ h (y,z)が成立することを示せ。

だそうです。（解答）

(左辺)=$\sum x_i \oplus y_i$

(右辺)=$\sum (x_i \oplus z_i + z_i \oplus y_i)$

$x_i \oplus y_i = 1$のとき、常に$x_i \oplus y_i + z_i \oplus y_i = 1$

$x_i \oplus y_i = 0$のとき、$x_i \oplus y_i + z_i \oplus y_i$は$z_i$の値によって 0 と 1 のどちらも取りうる。

解説は…

x　00111001001 01

z　00111011000 01

y　00111011001 01

$x_i \oplus y_i=1$ のとき、z_iは必ず片方と異なるが、もう片方とは一致する。よって常に 1 (i=11)

$x_i \oplus y_i=0$ のとき、z_iは(1)　xi=yi=zi だから h(x,z)+h(z,y)=0(2)　必ず xi≠zi または yi≠zi のどちらかだから h(x,z)+h(z,y)=1

まぁこんなもんです。難しいよね・・・

イミテーション・ゲーム(06 個別問 4)

問 4　アラン・チューリングが論文”Computing Machinery and Intelligence”, Mind, Vol.LIX,No.236,1950　で「機械は人間と同じように考えることができるか」という問題を考えるために考案したイミテーション・ゲームとはどのようなものか説明せよ。

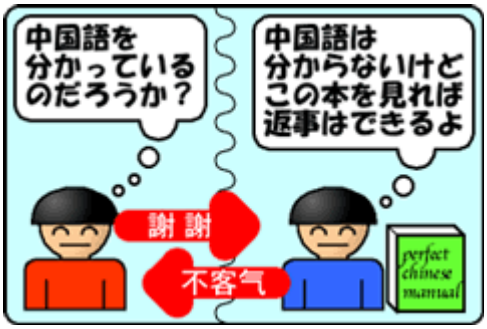
チューリングテスト

1950 年に数学者チューリングはチューリングテストという，知能があることに関する実験を提唱しました．それには，2 台のディスプレイの前にテストをする人がいます．1 台のディスプレイには隠れている別の人が，もう 1 台は人間をまねるように作られたコンピュータが受け答えした結果がそれぞれ出てきます．テストをする人はどんな質問をしてもよいとします．例えば，詩を作らせたり，音楽の感想を聞きます．また，コンピュータも人間をまねる努力をします．例えば，わざと計算に時間をかけたり，間違えたりします．こうして，テストをする人がどちらが人間でどちらがコンピュータか分からなければ，このコンピュータには知能があるとするのがチューリングテストです．

中国語の部屋

では，チューリングテストをパスすれば知能のある機械，すなわち，人工知能といえるでしょうか？これには有名な哲学者サールの『中国語の部屋』という反論があります．これは，英語しかわからない人が部屋にいます．その部屋には，中国語がわからなくても，中国語の文字を書いてあるとおりに置き換えると，中国語の受け答えができてしまう完璧な説明書があります．つまり，この部屋の人は，英語しか分かりませんが，中国語の質問に中国語で答えることができます．ということは，

中国語の受け答えができるだけでは，中国語が分かるとは限らないことになります．同様に，まるで知能があるような受け答えができるかを調べるというチューリングテストに合格しても本当に知能があるかは分からないという反論です．



もろにコピペです。（http://www.ai-gakkai.or.jp/jsai/whatsai/AItopics3.html ）

これで概要がつかめたでしょう。

ナイキスト周波数(06 共通試験共通問題 1 (a))

共通試験です。いきなりエグいですね。

共通問題 1　以下の文章には誤りがある。

(a)音楽 CD はサンプリング周波数が 44100Hz であり、理論上周波数が 44100Hz の音まで再現できる。

2.3.3　標本化定理　を参考してください。再現できるのはサンプリング周波数の 1 / 2 です。これをナイキスト周波数といいます。答えは 22050Hz。人間の耳は 20,000Hz までしか聞こえないのでこれで十分なのです。

6．最後に

始めに同じ中村先生の授業を受けている 5 組のシケプリより。

5 組シケ対よりひとこと（速読スキップ可）

情報の試験は共通問題と、各教官による個別試験から成り立っている。個別試験をしない教官もいるようだが、うちの中村政隆教官（大鬼）に尋ねてみたところ、「今のところ個別試験は行う予定です」とのこと（一昨年はやらなかったらしい）。共通問題と去年の個別試験も置く予定なので、そちらのほうも参照してほしい。ただし、情報は 05 年以前、「情報処理」という科目名で、過去問を参照する際には内容はまるで違うので気をつけていただきたい。

まず共通問題のほうだが、これは範囲から割と万遍なく出されているようである。対策は、教科書を徹底的に勉強することにある。というか、共通問題なのでそれ以上のことは出せないはず。まあ正直、教科書のほうを丁寧に勉強していただければそれが一番なのだが、一応まとめ的なものとしてこのシケプリを適宜参照すると良いだろう。シケプリだけでは不十分です（と予防線を張ってみる）。

範囲についてだが、今年度の分はここに載っている。赤文字は必修なので、試験範囲であると思われる。「要望学習項目」というものがあり、これは A,B に分かれている。教官に直接尋ねてみたところ、「大問ひとつ分が選択問題で、A,B どちらかの問題を選んで解いてもらうことになる（と思う）」とのこと。まだはっきりと聞いたわけじゃないのでよくわからないが、去年の大問 3 もそんな感じなのでたぶん間違いないだろう。まあご存知の通り、あの教官の授業はほとんど誰も聞いていないし、**それはシケ対の私とて例外ではない**。ということで、授業の内容はともかく、学習しやすいほうにしぼって学習するのが良いと思う。そこで、このシケプリでは「要望学習項目 A」のほうを推奨する。教科書を確認していただければわかる通り、B の内容というのは実に理系っぽい。我々文系の人間は無難に A を選択するのが良かろう。もちろん、余裕のある人や、理系的な問題のほうが好きだという人は B のほうを学習しても構わない。ところで、**授業のパワポを見る限りどうも B の内容を説明しているようだが、気にしてはいけない**。というか、全く必須項目でないところを説明しているみたい。あんまり聞いてないからよくわからないけど。

次に個別試験のほう。見ていただければわかるが、異常に難しい（と思う）。というか、**必須学習項目を無視しすぎ**です。なので、ちょっとまだ試験範囲のほうがはっきりわからない（「授業でやったところ全部」とか言われたら僕はもう……）ので、最終提出のときにできれば付け加えようと思うが、まずは共通問題をしっかりととることを優先したほうが無難だろう。ただし、03 年と 04 年の中村教官の個別試験を見ると、非常に内容がかぶっていることがわかる。このことから、もしかしたら 07 年と似たような箇所をついてくるという可能性もあるが、試験においてヤマをはるのは非常にリスキーなので、きちんと学習しておくにこしたことはないだろう。

敵は「大鬼」である。見かけや態度にだまされていると、思いがけず「不可」がくるかもしれない。心してかかりたい。

なお、上記の内容は不確定な情報及び主観を多分に含むものなので、信用するかどうかは自己責任でお願いします。

ああ、そうそう、**去年どっかの教官が共通試験の内容を前バラシしたらしい**ので、よそのクラスに人脈がある人は情報収集をよろしくお願いします（なんか毎年ある教官が前バラシするとかいう話も聞く）。

※追記：一応７章までの必修分野をまとめました。８章以降は各自教科書を読んでください。というのも、８章以降はシケプリでまとめるよりも、教科書の文章を読んだほうが記述問題に対応しやすいと思ったからです。分野が分野なので。あと個別試験はほぼ確実にオートマトンが出ます。

誰だか存じ上げませんが５組シケ対の方情報ありがとうございます。

共通試験の範囲

東京大学教養学部テキスト
Introduction to the New Information World
情報
川合　慧［編］東京大学出版会，275頁，2006 年 1 月．

細目次

1. 必須学習項目　**赤字と太字で示す**
2. 要望学習項目 A　緑字と下線で示す
3. 要望学習項目 B　青字と下線で示す

中村先生のページより

回 日付 場所 講義 実習

1 4/18 情報棟演習室 HWB 3. まず知っておく～ 8. 日本語の扱い / MailSuite の利用、日本語英語テキスト（入力練習用）

2 4/25 情報棟演習室 HWB 10 電子メイル / 13 ファイルシステム / プリンタの利用/ 15 エディタ / CFive の利用法 ー

3 5/2 講義室 第３章 情報理論・インターネット --情報理論、秘密鍵暗号、公開鍵暗号、ハフマン符号化 ー

4 5/9 講義室 (3.4.1 ネットワーク集合体、3.4.2 階層プロトコル IP アドレスとネットワークアドレス、ポート番号、TCP/IP, DNS の仕組み、 文字コード、ビット| WWWシミュレータ実演

第４章データモデル

ー

5 5/16 講義室 第５章プログラム・プログラム言語、第６章 アルゴリズムと計算量, ＮＰ-完全問題 // オートマトン、Turing 機械、計算可能性 | オートマトンシミュレータ実演 -

6 5/23 - 五月祭のため午後休講 -

7 5/30 講義室 第７章 CPU と論理回路 (7.1.1 コンピュータの基本構成, 7.1.2 機械語レベルのプログラム, 7.2.1 真理値表と論理関数, 7.2.3MIL 記法)8 章 WWW の仕組み、第２章 情報の表現(標本化、量子化、標本化定理(Shannon)、記号論)、 9 章 ユーザーインターフェイス、 10 章 情報技術と社会, インターネットの心理学 -

9 6/6 情報棟演習室 ＩＣトレーナー実習

必須課題 | ７章抜粋 {PDF} ＩＣトレーナー実習（外部リンク）

10 6/13 - 出張のため休講 -

11 6/20 情報棟演習室 WWW・メイル シミュレータ実習

オートマトンシミュレータ実習

Travelling Salesman ゲーム

ＰＰＴ実習 WWW・メイル シミュレータ / PPT の利用, ピクトグラムの作成

11 6/27 情報棟演習室 オートマトンシミュレータ

ED21 CPU シミュレータ HTML(文 12)、LaTeX(理 1), LaTeX 標準ファイル[tex]その PDF ファイル[PDF]

12 7/4 情報棟演習室 HTML(文 12)、LaTeX(理 1) HTML(文 12)、LaTeX(理 1)

13 7/9(水) 情報棟演習室 UNIX のファイルとディレクトリ・コマンド HTML(文 12)、LaTeX(理 1)

14 7/11 情報棟演習室 プログラミング プログラミング

共通問題の解答は別にまとめました。これもコピペです。個別 07 に関しては一部補足をつけるにとどまりました。あとは教科書みれば何とかなるでしょう。…なんとなく過去問の類題が出る気がします。

エントロピーに関して本をお借りしたので、それをスキャンしてアップしておきます。また中村スライドも PDF 化しておきます。また実習で使う Automat と ED21 シミュレータは「じょうほうの情報」にアップします。

…これらは「シケプリパック」として圧縮して「じょうほうの情報」にアップしました。

あとがき

長いですね。2 万 4 千字の超大作…とはいえ、これで全部ではないのです。。実習＋αだけですから。作成時間は 54 時間かな？多分。おかげさまで社会 1 の勉強ができませんでした。不可った(涙) もとはと言えば自分で引き受けた仕事なのでどうしようもありませんがｗ

何故ここまでできたのだろうか…？

もともとは 1 日でざっと説明を書き上げようか、というつもりでした。しかし教科書を参考にして書こうとすると、どうしても行き詰ってしまいました。説明すべきところが説明されていなかったり、論理に飛躍があったりしたのです。本 1 冊分の内容が 1 章に圧縮されていたり。「あまりにもひどい。」これを読んでもわかりにくいので、内容の順序を組み替え、補充するところから始めました。その結果とても時間がかかってしまいました。

このわかりにくい教科書をよみながら思ったのです。「この人たちは読み手に伝える意欲はあるのだろうか。」この本は 1 章ごとに別の人が書いています。「東大教養学部テキスト」であるがゆえ、内容を濃縮しようと張り切ってしまうのはわかります。しかし、そうすることで「テキスト」とはかけ離れてしまっていると思います。

…別にこの本に限ったことではありません。技術系の本全般に言えると思います。中学のころから、著者が「自分達がわかっていればよい」と思っているのかどうかはわかりませんが、人に伝える意欲に欠けている気がしていました。

「高校レベルの知識層に説明して伝えることができなければ、その人は科学を理解しているとは言えない」とは物理学者ファインマンの言葉です。

東大生は「高校レベルの知識層」としては十分だと思います。それであるが故に、本当に「科学を理解している」のだろうか、と思ってしまいます。中村先生然り。

このシケプリを作る際には「高校レベルの知識層に説明して伝えること」を最重要課題にしました。自分が中学の頃から…違った小学校の頃から高校で吹奏楽を始めるまでに吸収した「科学」を果たして理解できているのか、その判断は読者にゆだねたいと思います。

このシケプリを作るのに手伝ってくださった工学部 3 年の石井さんに感謝します。情報の試験とはなんら関係ないのにも関わらず、わざわざ時間を割いて自分の無知を正してくれました。

最後に、このシケプリを作るきっかけを与えてくれた 20 組のみんなに感謝。このシケプリ（もはや教科書な気がする…）が学内最高水準であると自負しています。相手は大鬼ですが、頑張って単位をとりましょう。質問はごうしょうまで。

それでは残りの試験も頑張ってください。

JAH！！　　7/19　22：39　ごうしょう

2008/07/20

情報 − 実習編

訂正・補足　7/20 版

ごうしょう

情報－実習編

訂正・補足　7/20 版

　ごうしょうです。早速間違いが発見されました。勉強不足ですみません m(__)m

訂正

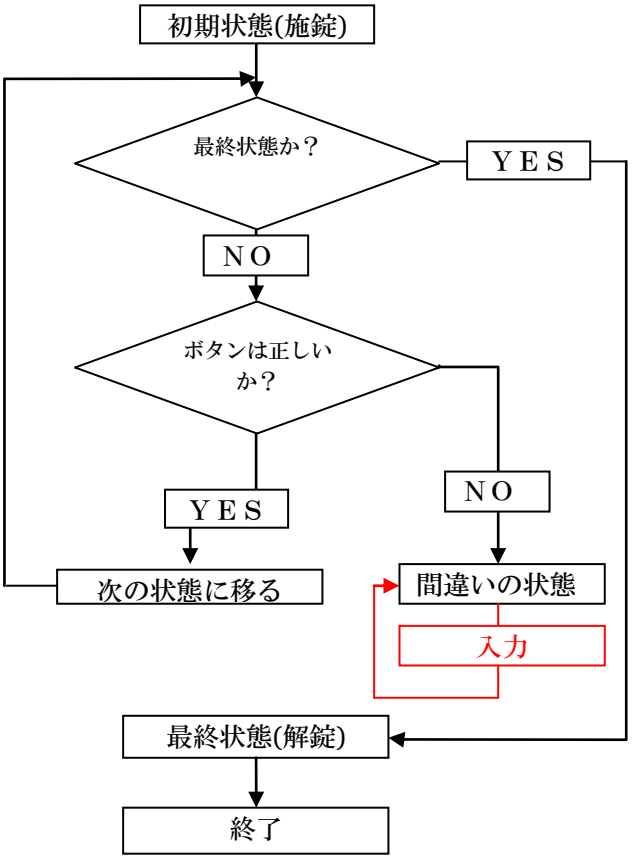
P.11 減算器(参考　7.3.4)

　まず、負の数を扱うために一番左のビット(最上位ビット)を符号ビットとします。つまり、左にひとつ余分にビットをくっつけて、それが 0 なら正、1 なら負の数を表すことに決めるということ。それから、負の数は 2 の補数で表すことにします。2 の補数で表すには、まず正の数で書いてみて、次にビットを全て反転させ(0→1、1→0 とする)、最後にそれに 1 を足せばよいです。2 の補数で書かれているものを読むときも、これと同じ手順で変換します。例えば、－7 を表すには、まず 00111 を用意して、これを反転して 11000 とし、これに 1 を加えて 11001 とします。このとき、自動的に最上位ビットが 1 になってます。それからもうひとつ、－0 が存在しないことも確認して下さい。さて、減算を計算するには、負の数の足し算とすればよいです。このとき、符号ビットまで含めた足し算を行います。例として１２－７を計算してみます。１２は 01100、－7 は 11001 なので、この二つを足して 00101(最上位からの桁上がりは無視)となり、5 が得られました。

　以上コピペです。すごい先輩です。。

3．オートマトンシミュレータ（6/27 6.2.1）

フローチャート



ハミング距離（07 個別）問 2 (b)

　記号列 x,y のハミング距離を h(x,y)と書くとき、任意の記号列 x、y、z で h (x,y)≦ h (x,z)+ h (y,z)が成立することを示せ。

　だそうです。（解答）

$$(\text{左辺})=\sum x_i \oplus y_i$$
$$(\text{右辺})=\sum (x_i \oplus z_i + z_i \oplus y_i)$$

$x_i \oplus y_i = 1$のとき、常に$x_i \oplus y_i + z_i \oplus y_i = 1$
$x_i \oplus y_i = 0$のとき、$x_i \oplus y_i + z_i \oplus y_i$は$z_i$の値によって０と２のどちらも取りうる。

解説は…

x_i　00111001001O1
z_i　00111011000O1
y_i　00111011001O1

$x_i \oplus y_i=1$ のとき、z_iは必ず片方と異なるが、もう片方とは一致する。よって常に 1 (i=11)

$x_i \oplus y_i=0$ のとき、z_iは(1)　xi=yi=zi だから h(x,z)+h(z,y)=0　(2)必ず $x_i \neq z_i$ かつ $y_i \neq z_i$ だから h(x,z)+h(z,y)=1

2006 共通問題問 2

(c) f(1,4)を実行すると r = 2（各自確認すること）→f(1,2)とf(3,4)を呼び出す。f(1,2)とf(3,4)を計算するのにf はそれぞれ 2 回ずつ必要で、よって計 7 回。

(d) 結論から言ってしまえば、2(n-1)回。

↑ (c)は f(1,4)を計算するのに何回 f が使われるかっていう問題だけど、f(1,4)自体も数に入れるんじゃないかな？だから答えは 7 回。(d)も同じように考えると 2n-1 回になる。

補足

コンピュータの基本構成（必修 7.1.1）

図 7.1 はプログラム内蔵式コンピュータ（フォン・ノイマン型コンピュータ）の最も基本的な部分を示しています。（テキストより…ちょっと荒いけど気にせず）　演算装置、主記憶装置(メインメモリ)　制御装置に大別されます。制御装置と演算装置を合わせたものを中央処理装置(CPU; Central Processing Unit)といい、この中央処理装置を一つの IC にまとめたものを MPU(Micro Processing Unit)と呼びます。と呼びます。主記憶装置(メモリ)とは複数の情報を格納し、選択的に読み書きできるものです。このメモリのなかは情報を格納する小さな箱の集まりのようなもので、その箱には一つ一つアドレスというものが割り振られています。そのアドレスを選択することで選択的に読み書きができます。

演算レジスタもデータを保存する装置ですが、演算は演算レジスタの中のデータに対して行われ、計算途中のデータや演算結果が保存されます。初期のコンピュータでは演算レジスタは一つしかなく、アキュムレータ(AC; ACcumelater)と呼ばれていました。

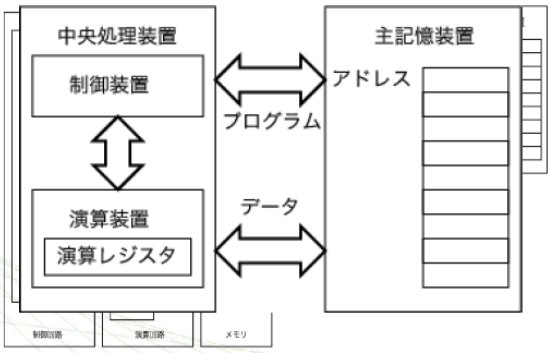


図 7.1

7.1.2　機械語レベルのプログラム例

これを要望学習項目 B とするのは酷でしょう…

コンピュータはプログラムを見て計算をしていきます。そのプログラムは機械語（0 と 1 の羅列）で書かれています。しかしこれを人が扱うことはとても難しいです。そこでこの機械語の命令を英単語に置き換えたものをアセンブリ言語と呼びます。以下の説明ではこのアセンブリ言語を使ってプログラムが書かれています。

個々のコンピュータで利用できる命令群を命令集合と呼びます。各命令は命令コードとオペランドと呼ばれる付加情報からなります。「load A」というのはアドレス「A」にあるデータを演算レジスタに「格納する」という意味。

命令集合は、一般に「データ転送命令」「演算命令」「分岐命令」に分かれます。データ転送命令(ex. load store)というのは主記憶装置上のデータを演算レジスタに書き込んだり、演算レジスタのデータを主記憶装置に書き込む命令です。演算命令(ex. add and or substract)は演算レジスタや主記憶装置のデータ四則演算や論理演算する命令です。分岐命令(ex. jump call)は指定したアドレスにプログラムの実行を移したり、条件分岐を行う命令です。

…P.156 に書かれているプログラムは ED21 シミュレータで実行するとよいでしょう。

種類	内容	意味
データ転送命令	load A	アドレス A のデータを演算レジスタに読み込む
	store A	演算レジスタのデータをアドレス A に書き込む
演算命令	add A	アドレス A のデータを演算レジスタの値に加える
	substruct A	アドレス A のデータを演算レジスタの値から引く
分岐命令	jump A	アドレス A にプログラムの実行に移す
	jumpzero A	演算レジスタのデータが 0 の場合、アドレス A にプログラムの実行を移す
その他	write	演算レジスタのデータを出力する
	halt	プログラムの実行を停止する

命令集合の例