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

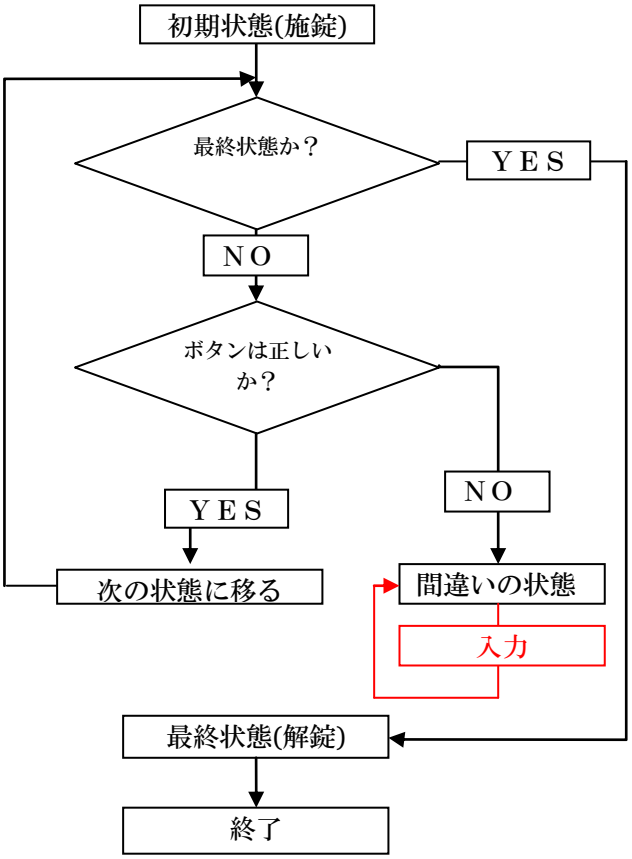
P.11 減算器(参考　7.3.4)

　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。

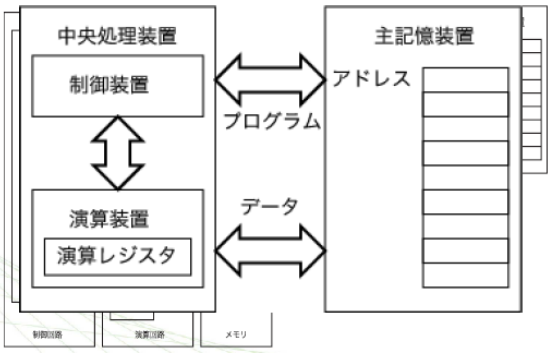


図 7.1

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