

プログラムの動作について

プログラムは必ずアドレス 0 から実行され、END命令の実行で停止する。プログラムが無限ループに陥った場合、端末から割り込み(Control-C)をかけて停止させるとよい。

プログラムの最後の命令が END でなかった場合、機械語の最後に自動的に END が追加される。また、未定義シンボルがあった場合、各シンボルをラベルとするメモリが2バイトずつ自動的に確保される。これは原著のプログラムをそのまま実行するための措置であり、このことを前提としてプログラムを記述すべきではない。

シミュレータ起動時に `-t` オプションを指定すると、実行されるコマンドとレジスタの状態が逐一表示される。

拡張機能について

このシミュレータの実装で規定した事項、および追加、拡張した機能について述べる。

1. プログラムは大文字、小文字を区別しない。ラベル名として使えるシンボルは7文字までしか認識しない。ラベル名は先頭が英文字、2文字目以降は英数文字を使う。命令、レジスタ名、その他システム定義の名前はラベル名としては使えない。システムの予約語は以下の通り。

```
COPY, ADD, SUB, MUL, DIV, CMP, JMP, JNB, JB, IN, OUT,
CALL, RTN, PUSH, POP, END, AX, BX, CX, DX, RES, RANDOM
```

区切り文字には空白のほかにタブが使える。

2. 「;」から行末まではコメントと見なされる。空白行、またはコメントだけの行があってもよい。
3. アドレッシングはバイトアドレスのみで、アドレス空間は12ビット（4096バイト）のみである。
4. データは2バイトからなるワード（16ビット符号付き整数）単位で扱う。
5. レジスタは AX から DX までの4つが使える。いずれも 16ビット符号付き整数である。
6. 即値形式でオペランドに8ビット符号付き整数を直接指定できるようにした。値は「#」で書き始める。ただし、即値形式が利用できるのは COPY命令、CMP命令および算術演算命令である。
7. 間接アドレッシングが利用できる命令を COPY命令に限定した（原著では明確でない）。
8. レジスタDXをスタックポインタとして PUSH, POP命令および CALL, RTN命令が使える。これらの命令を使ってサブルーチンの記述が可能となっている。プログラムの実行開始直後、レジスタDXは 4094 (0xffe) に設定される。なお、スタックおよびこれらの関連命令の追加は原著のプログラムを動作させる上で何ら問題とはならない。
9. 連続した複数バイトからなる領域を確保するために、RESというディレクティブを追加した。
10. これは原著にもある機能だが、RANDOM というアドレスから乱数を取り出せる。値を読み込むたびに 0 ~ 32767 の範囲の異なる値（乱数）が得られる。

使い方

`./p88 [オプション] ソースファイル`

オプション	<code>-t</code>	トレースモードで実行	<code>-c</code>	文法チェックのみ（実行しない）
	<code>-d</code>	機械語（中間コード）を表示	<code>-h</code>	このヘルプを表示

命令の書き方と意味

以下で、AX, BX とある位置には任意のレジスタを記述できる。memには数値またはアドレスを表すラベルを記述する。#8 には「#」に引き続いて8ビット符号付き整数（-128 ~ 127の範囲の数値）を記述できる。

END	プログラムを停止させる。オペランドなし。
COPY AX,BX	レジスタBXの内容をレジスタAXにコピー
COPY AX,mem	アドレスmemのメモリ内容をレジスタAXにコピー
COPY AX,c(BX)	レジスタBXの値をアドレスとするメモリ内容をレジスタAXにコピー
COPY AX,mem+c(BX)	memにレジスタBXの値を加算したものをアドレスとし、その内容をレジスタAXにコピー。 (memが0なら c(BX) と同じ。BXが0なら mem と同じ。)
COPY AX,#8	8ビットの符号付き整数をレジスタAXにコピー
COPY mem,AX	レジスタAXの値を、アドレスmemのメモリに格納。
COPY c(BX),AX	レジスタAXの値を、レジスタBXの値をアドレスとするメモリに格納。
COPY mem+c(BX),AX	memにレジスタBXの値を加算したものをアドレスとするメモリに、レジスタAXの値を格納。
ADD AX,BX	レジスタAXにレジスタBXの内容を加える。結果はAXに残る。
ADD AX,mem	レジスタAXにアドレスmemのメモリ内容を加える。結果はAXに残る。
ADD AX,#8	レジスタAXに8ビットの符号付き整数を加える。結果はAXに残る。
SUB、MUL、DIV	これらはADDと同じ形式で、それぞれ減算、乗算、除算を行う。 ただし、0で除算をすると実行時エラーとなってプログラムは停止する。
CMP AX,BX	レジスタAXとレジスタBXの内容を比較。
CMP AX,mem	レジスタAXとアドレスmemのメモリ内容を比較。
CMP AX,#8	レジスタAXに8ビットの符号付き整数を比較。 これらの結果、AXの方が小さい場合、CFはB(Below)になる。AXの方が大きいか等しい場合、CFはNB(Not Below)になる。
JMP mem	無条件でアドレスmemに分岐。
JNB mem	CFがNBの場合、アドレスmemに分岐。
JB mem	CFがBの場合、アドレスmemに分岐。
CALL mem	スタックポインタ（レジスタDX）を2バイト分デクリメントし、スタックのその位置に現在のプログラムカウンタに2を加えた値を格納した後、アドレスmemに分岐する。
RTN	スタックからアドレスを取り出し、スタックポインタ（レジスタDX）を2バイト分インクリメントしてから、取り出したアドレスに分岐する。
PUSH AX	スタックポインタ（レジスタDX）を2バイト分デクリメントし、スタックのその位置にレジスタAXの値を格納する。
POP AX	スタックから取り出した値をレジスタAXにコピーし、スタックポインタ（レジスタDX）を2バイト分インクリメントする。
IN AX	端末からレジスタAXに整数（16ビット符号付き）を読み込む。
OUT AX	レジスタAXの内容を端末に表示する。

データ領域の確保

n	値 n を持つ2バイトのメモリ領域を確保する。n は数値またはラベル。
RES n	n バイトの大きさを持つメモリ領域を確保する。