教育用仮想電子計算機とそのシミュレータ

教育用仮想電子計算機とそのシミュレータ

山田　攻・津国敏明・塚原　至
安部嘉一・北村正一

A Virtual Computer for Education and its Simulator

O. Yamada, T. Tukuni, I. Tukahara, K. Abe
and S. Kitamura

Abstract

To learn the computer science, it is required of us to have some preliminary knowledge of the computer hardware. It is desired to do the experiment using assembly language.

But commercial computers are too complicated and inconvenient for students to learn a survey of computer. Therefore, we designed a virtual computer EDUAC, an assembly language EDUAL, and a simulating program has been written to simulate the EDUAC.

The EDUAC has one program counter, one instruction register, one address register, one accumlator, two I/O buffer registers and a few core memories. The EDUAL has twenty-two executive instructions and five pseud-instructions.

Whenever a instruction is executed, the simulator checks whether the operation is correct or not, and prints out the content of every register.

I. はじめに

電子計算機の教育においては，実習を行ない，電子計算機を使わせてみることによって，より早く，より確実に理解できる部分が少なくない。また，そうした経験が不可欠の要素となることが多い。とくにソフトウェア関係では，オペレーティング・システムの諸概念やシステム・プログラムの内容などを理解するために，その予備知識として，機械語あるいはアセンブラー言語レベルでの電子計算機の動作を知っていることが必要になる。ハードウェアの面においても細かな回路構成の講義以前に，機械語等の使用によって，機械全体の構成とその動きを体験しておくことは非常に好ましいことと言える。

しかしながら，実際に電子計算機の機械語あるいはアセンブラー言語を覚えて，これを使用してみるためには，少なくとも１週間以上の長時間を要し，プログラマーの養成を目的とするのでない限り実施は困難である。また，単なる電子計算機の動作の理解のために，既存の電子計算機を利用するのは何かと不便が多い。

このような点にかんがみ，電子計算機の基本的な諸概念を理解させるにはいかなる電子計

算機が最適であるかを考えて，適当な教育用仮想電子計算機を設定し，そのシミュレータを作成した。

II. 教育内容との関連

電子計算機関係の教育において，最も問題となることは，その根本の基礎概念となる“記憶”とか“処理”の実際上の具体的なイメージを机上ではつかみにくいことである。それゆえ，実習に費やすことができる時間を考慮して，仮想電子計算機の主目的をこの点に絞って考えることにする。この場合その境界をどこにおくかは非常に大きな問題となるが，ここでは経験的なものから，次のような事項を理解させることを目的とする。

(1) プログラム内蔵方式の概念
(2) 電子計算機内部でのデータ表現
(3) 電子計算機が有している“命令”の実際
(4) 命令遂行の手順
(5) アドレス方式（直接，間接，相対，インデキシング）
(6) ルーチンの結合
(7) 入出力の方法
(8) 演算・記憶・入出力・制御の有機的なつながりと処理の流れ
(9) アセンブルおよびコンパイルの意味と概念

入出力は中央処理装置の側からみた“読む”，“書く”の意味を理解するにとどめ，入出力機器および二次記憶に関する知識を得ることは目的としない。また，割り込みの概念を知ることは非常に重要であるが，これ自身をとくに経験していなくても，上記の項目が理解できれば，机上で説明するだけで充分であると考えられるので省略した。

III. 方　　式

上述の目的を実現するために，具体的には次のような方式をとる。

(1) 適当な仮想電子計算機システムを設定する。
(2) この仮想電子計算機の機械語とアセングラー言語を作る。
(3) 受講者には両方の言語を教えるが，プログラムはアセンブラー言語によって組ませる。
(4) 実在する電子計算機を用いて，仮想電子計算機のためのアセンブリー・プログラムを作る。
(5) このアセングリー・プログラムによって受講者のプログラムをアセンブルし，仮想電子計算機の機械語を作り出す。
(6) 記述の誤りがなければアセンブラー言語と機械語との対応表を印刷する。

(7) 作り出された機械語のプログラムをシミュレータ・プログラムによって実行する。

(8) 実行に際しては，仮想電子計算機の状態変化を逐次印刷する。

アセンブリー・プログラムおよびシミュレータ・プログラムは実際に実行させる電子計算機のアセンブラー言語を用いて記述する。この方式において，教育効果を左右するのは，仮想電子計算機の機能という意味も含めて，いかなるアセンブラー言語を設定するかという点と，電子計算機の動作を理解しやすくするために，その状態変化をいかに表示するかという2点であると思われる。

IV. 教育用電子計算機としての必要条件

どのような電子計算機を設定するかを決めるにあたっては教育用であり，かつ金物を作成するわけではないから，経済性にからむ問題は取り除かれるが，逆に教育に要する時間的制約から必要最小限の機能をもつ電子計算機という条件が生じてくる。しかしながら，市販の小型あるいは超小型電子計算機との最も大きな相違は，「使うために便利な必要最小限」という条件が，「知っておくべき機能を一通り有している必要最小限」という条件にかわることである。例えば，仮に小型電子計算機が直接・相対・間接・インデキシングというすべてのアドレス形式を有する必要がないとしても，教育用としてはすべてのアドレス形式を使用できることが好ましい。また逆に，プログラムを作るのは，使ってみることに意味があるのであって，その演算結果を利用するのが目的ではないから，複数個のインデックス・レジスタを用意する必要はないと言える。

前述の種々の教育内容や制約を考慮すれば，教育用電子計算機と，そのアセングラー言語を定めるにあたっての必要条件，または設定の基準は次のようになると考えられる。

(1) 非常に単純化された電子計算機システムであること。

(2) 電子計算機の基本的動作を表わす命令は一通り有すること。

(3) 使用頻度の高い命令であっても，類似する機能があるか，または他の機能を理解すれば，机上の説明で充分理解できるものは省く。

(4) 市販されている電子計算機とあまりかけ離れていないこと。

(5) 命令は覚えやすいこと。具体的には，命令の数が少なく，形式が単純であって，記号表示は直観的であること。

(6) プログラムを組みやすいことはとくに条件としない。例えば，シフト命令を何種類も用意するなど。

(7) 処理の流れが理解しやすいこと

(8) 教えやすいこと。例えば，アドレス形式の変化など，機能の増加は，積み重ねの形で教えられること。

(9) 上述の条件に支障を生じなければ，後日シミュレートしている電子計算機 (FACOM 270-20) を利用する場合を考慮して，これに近い形とする。例えば記号の用い方など。

これらの条件には，互に相反する性質となるものが存在しており，すべての条件を満たすことは不可能であるが，最終的な決定に際しては II. の教育内容に沿った例題を作り，その適否を判定した。

V. 教育用仮想電子計機

前述の諸条件を考慮して，次のような教育用の仮想電子計算機 EDUAC (Educational Automatic Computer) を設定した。

コア： 128 W (1 W=16 ビット)

レジスタ類

1) プログラム・カウンター 1 個
2) 命令レジスタ 1 個
3) アドレス・レジスタ 1 個
4) アキュムレータ 1 個
5) インデックス・レジスタ 1 個
6) 入出力用のバッファ・レジスタ各 1 個

プログラム・カウンターには次に実行する命令のアドレスが入る。命令レジスタには実行する命令が入る。アドレス・レジスタには，命令のオペランド (アドレス) が入り，アドレス修飾が行なわれる場合はアドレス修飾後の，実際にアクセスするアドレスが入る。アキュムレータは，一般には，乗除算のために 2 個必要であるが，実習はごく簡単な，例えば 2×3 というような既知の小さい値に対して行なう程度で充分であるから，理解を容易にするため簡略化した。加減算も同様であるが，あふれ分はすべて無視する。したがって，あふれを表示するインジケータ等は一切省略した。入出力の制御命令もすべて省略し，入出力命令 1 個によって，アキュムレータ内のコードの出力，またはアキュムレータに入力するものとする。入出力コードと内部コードの変換はハードウェアによって行なっているとする。ただし，入出力命令が実行されると，入出力バッファに入出力コードが入る。

命令のビット構成は，アドレス形式が 2 ビット，命令の種別の表示が 5 ビット，アドレスが 9 ビットである。数値表現はすべて 2 進数で，負の数は 2 の補数表示とする。

EDUAC をシミュレートする電子計算機システムの関係から，入力には紙テープ，出力にはタイプライターを使用する。また EDUAC では 2 次記憶装置をとり扱わない。EDUAC とそれをシミュレートする電子計算機との対応をとくに意識しなくてよいように工夫しているが，多人数のプログラムを 1 ステップ毎に実行し，その様子を見せることは困難であるので，

同等のことを紙の上（印刷）で表示する。このときは，上述の各種のレジスタ類，およびコアの内容を表示し，理解を確実にする。

VI. アセンブラー言語の仕様

ここで言うアセンブラー言語とは，単なる機械語の記号表示としての意味だけでなく，電子計算機の細かな機能の決定の意味を含めている。これは，受講者は，アセンブラー言語の利用を通して，電子計算機の動作を知るため，アセンブラー言語の良し悪しによって全体が大きく影響を受けるので，アセンブラー言語の仕様と電子計算機の機能とを切り離して考えられないことによる。

教育用アセンブラー言語 EDUAL (Educational Assembly Language) は，実行命令22種と擬似命令5種からなり，記号表示と意味は次の通りである。

(実行命令)

L	(load)	(D)→Acc
ST	(store)	(Acc)→D
A	(add)	(Acc)+(D)→Acc
S	(subtract)	(Acc)−(D)→Acc
M	(multiply)	(Acc)×(D)→Acc
D	(divide)	(Acc)÷(D)→Acc
AND	(and)	(Acc) と (D) のビット毎の論理積
OR	(or)	(Acc) と (D) のビット毎の論理和
EOR	(exclusive or)	(Acc) と (D) のビット毎の排他的論理和
SETX	(set index)	N→index
AX	(add index)	N+(index)→index
J	(jump)	(D)→PC
JP	(jump plus)	(Acc)>0 ならば (D)→PC
JZ	(jump zero)	(Acc)=0 ならば (D)→PC
JM	(jump minus)	(Acc)<0 ならば (D)→PC
JZX	(jump zero index)	(Index)=0 ならば (D)→PC
JL	(jump and link)	(PC)→D, (D)+1→PC
SL	(shift left)	Acc の内容を N ビットだけ左へ移動
SR	(shift right)	Acc の内容を N ビットだけ右へ移動
READ	(read)	紙テープから1文字読み，内部コードに変換して Acc に入れる。
PRNT	(print)	Acc に示される内部コードに相当する文字を1文字印刷する。

STOP　(stop)　　電子計算機の停止。

(擬似命令)

RSV　(reserve)　　N語の領域確保。

DEC　(decimal)　　Nを10進数とみなして内部表現(2進数)に変える。

OCT　(octal)　　Nを8進数とみなして内部表現に変える。

ACNT　(address constant)　Dのアドレスを格納する。

END　(end)　　プログラムの終りを示す。

ただし，READ, RRNT, STOP, ENDを除き，すべて1個のアドレス部をもち，SETX, AX, SL, SR, RSV, DEC, OCTはアドレス部に数値Nが書かれ，他のものは記号アドレスDが書かれる。( )はそのアドレスの内容を表わす。

形式を単純化する意味では，バリアント部を持たない命令形式が望ましく，分岐命令はこの趣旨によって，条件を加味した命令を設けた。しかしながら，アドレス修飾の形式をバリアントなしで表示すると，例えば，インデックスに加えることと，インデキシングを行なって加えることの区別がつきにくく，かえって紛らわしいので，アドレス形式だけはバリアントの形をとることにした。この場合，最初はバリアントがないものとして教えても支障のないように，上記の実行命令に括孤を附して，次のような形で使用する。

L(X), DATA

括孤内がアドレス形式の指定で，次の3種類とする。

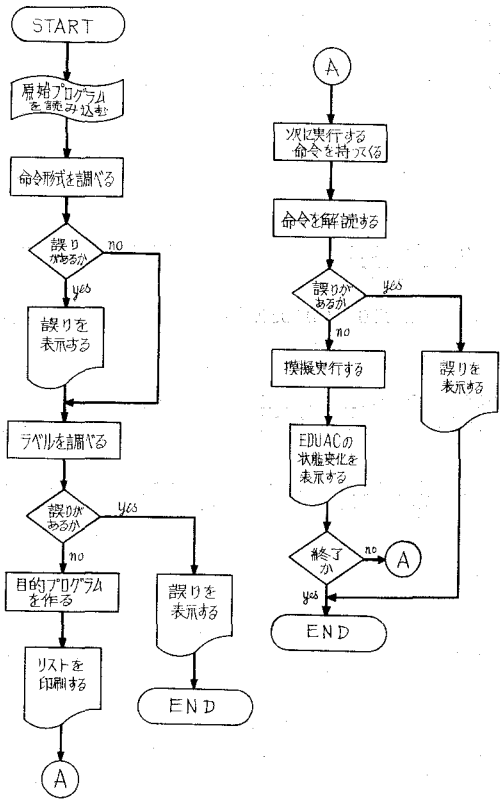
R：相対アドレス，　I：間接アドレス，　X：インデキシング

EDUALの命令中，入出力とインデックス関係を除いては，特に問題になる点はないと思われる。インデックス関係の4種の命令は，アドレス部1個という原則を守るために，独立した命令にした。インデックス関係では，このほか，インデックスの内容を指定アドレスへ転送する命令も考えられるが，実用的なプログラムをEDUALによって組むことは考えられないので省略した。このことは，入出力および各種の変換用サブルーチンを特に用意しない理由でもある。入出力命令の形式は種々考えられるが，入出力の意味を理解するという点だけに絞って考え，非常に単純化した。これは，実際の電子計算機における，入出力の難かしさとかけ離れるが，このように単純化してもなお，2進～10進の変換など多くの問題を残しており，入出力の複雑さの一端は経験できる。ただ，EDUACではコード変換はハードウェア処理であるが，その相違を知るため，シミュレータでの動作表示にこの点をつけ加えることにした。

ラベルは;を区切り記号とし，命令の末尾の区切り記号には@を用いる。

VII. アセンブリー・プログラムとシミュレータ・プログラム

この二つのプログラムの構成は図-1に示す。アセンブリー・プログラムは一般の電子計算機と全く同じであって，アセンブリー時に誤りを見い出せばその誤りの個所と種別を表示し，誤りがなければ原始プログラムと目的プログラムの対応表を印刷する。



図—1　アセンブリー・プログラムとシミュレータ・プログラムの流れ図

Fig. 1.　Flow chart of assembly program and simulator program.

シミュレータは，EDUACの機能をシミュレートすると同時に，1命令の遂行毎にEDUACの状態を表示する。

5種のレジスタ類（プログラム・カウンター，命令レジスタ，アドレス・レジスタ，アキュムレータ，インデックス・レジスタ）は1ステップ毎に必ず表示する。ST命令はコアの内容を変化させるので，コア・マップを印刷する。入出力命令では，入出力コードとそのシフト・ケースを表示する。また出力の際には，それまでに出力されたすべての文字を印字する。このことはEDUACの状態表示と，その実行結果の出力とが同一の用紙に印刷されるため，両者の区別

```
EDUAL Assemble

  Source Program                    Object Program

 1          L,      X@          000  021006 (0010001000000110)
 2          A,      Y@          001  001007 (0000001000000111)
 3          ST,     Z@          002  022010 (0010010000001000)
 4          A,      CODE@       003  001011 (0000001000001001)
 5          PRNT@                004  032000 (0011010000000000)
 6          STOP@                005  030000 (0011000000000000)
 7  X;      DEC,    1@          006  000001 (0000000000000001)
 8  Y;      DEC,    2@          007  000002 (0000000000000010)
 9  Z;      RSV,    1@          010  000000 (0000000000000000)
10  CODE;   OCT,    40@         011  000040 (0000000000100000)
11          END@

Execution


Step 000
  Prog.C.=000000   Inst.Reg.=000000   Ad.Reg.=000000
  Acc.=000000 (0000000000000000)   Index Reg.=000000
 Core Map
  021006  001007  022010  001011  032000  030000  000001  000002
  000000  000040

Step 001
  Prog.C.=000001   Inst.Reg.=021006   Ad.Reg.=000006
  Acc.=000001 (0000000000000001)   Index Reg.=000000

Step 002
  Prog.C.=000002   Inst.Reg.=001007   Ad.Reg.=000007
  Acc.=000003 (0000000000000011)   Index Reg.=000000

Step 003
  Prog.C.=000003   Inst.Reg.=022010   Ad.Reg.=000010
  Acc.=000003 (0000000000000011)   Index Reg.=000000
 Core Map
  021006  001007  022010  001011  032000  030000  000001  000002
  000003  000040

Step 004
  Prog.C.=000004   Inst.Reg.=001011   Ad.Reg.=000011
  Acc.=000043 (0000000000100011)   Index Reg.=000000

Step 005
  Prog.C.=000005   Inst.Reg.=032000   Ad.Reg.=000011
  Acc.=000043 (0000000000100011)   Index Reg.=000000
  I.Buf.=00000000 (Shift=UC)   O.Buf.=11111110 (Shift=LC)
 Out Put
  #3#

Step 006
  Prog.C.=000006   Inst.Reg.=030000   Ad.Reg.=000011
  Acc.=000043 (0000000000100011)   Index Reg.=000000
 Core Map
  021006  001007  022010  001011  032000  030000  000001  000002
  000003  000040
```

図—2 シミュレーションの一例

Fig. 2. A example of simulation.

がつかず見にくくなるのを避けるためである。アセンブルとシミュレートの結果の表示例を図-2に示す。

図-2 において，step 0 は初期状態を表わしている。各種レジスタ類およびコアの内容の表示は 8 進数で，括弧をつけた部分は 2 進数で表示している。

VIII. おわりに

現時点では，まだ EDUAL を使用した教育を行なっていないので，その教育効果を検討することはできないが，この EDUAL の前身となった言語 ESAP[1] の例からして，1 回 2 時間の講義を 3 回程度行なうことによって，目的の大半は達せられると考えられる。この教育効果については機会を改めて報告したい。

最後に，コーディングおよびデバッキングに際して，協力していただいた高野謙治，青山栄喜の両氏，およびいろいろとご検討をいただいた本学電子工学科の皆様に謝意を表します。

(昭 46. 5. 20 受理)

文　献

1) 津国敏明：教育用アセンブラー言語について，室蘭工大，昭和 45 年度卒業論文.