CASLをベースとした総合的アセンブリ言語教育システム*

堀内　征治**・堀内　泰輔***

A System for Synthetic Learning of Assembly Language Based on CASL

Seiji HORIUCHI and Taisuke HORIUCHI

It is essential that contemporary mechanical engineer who needs to master mechanical control with a computer should learn not only assembly language but also the hardware related to that language. But the assembly language used for CPU which is available on the market has so many instructions that the learners of the language have been suffered from the confusion caused by them. In order to solve this problem, it is necessary to construct a virtual machine with a small number of instructions. In March 1987, CASL, an assembly language for a virtual machine named COMET, was adopted to the certificate examination of information processing under the auspices of the Ministry of International Trade and Industry. It seems that CASL has been suitable for the beginners because of its compactness and sufficiency.

Firstly, the CASL simulator which is able to work on any personal computer has been developed making use of C language. Secondly, the simulator was implemented into microcomputer which has the ability to control sensor or actuator via I/O port, and another system with the hardware has been developed. It has been proved that these systems are appropriate for the acquisition of the assembly language.

1. はじめに

コンピュータによる機械制御を学習するうえで，アセンブリ言語の習得は必須の条件である．しかし，アセンブリ言語は個々の命令が単純なため，プログラムが大きくなりがちでバグが発生しやすい．また，実行時の暴走も考えられる．こういった点から，初心者のアセンブリ教育には，専用に作ったシステムを用いるのが最適といえる．一方，ハードウェアの面では，制御用インターフェイスの製作も初心者にとっては難しく，配線ミスなどでコンピュータ本体を破壊する恐れがある。したがって，ハードウェアの学習にも専用のユニットが必

*　昭和62年8月　全国高等専門学校情報処理教育研究協議会，情報処理教育研究発表会において発表
**　機械工学科　助教授
***　機械工学科　助手
原稿受付　昭和62年9月30日

要となる.

以上の観点から，ソフト・ハードの両面にわたる総合的なアセンブリ言語教育システムの開発を試みた.

本システムを次の2つで構成した.

1) アセンブリ言語学習用 CASL シミュレータ

2) CASL による機械制御学習システム

以下に，これらのシステムの概要を報告する.

2. アセンブリ言語学習用 CASL シミュレータ

CASL は通産省の情報処理技術者試験の中で62年度から新たに用いられている仮想計算機 COMET 上のアセンブリ言語であり，表1のように初心者の入門用言語としてはきわめてシンプルで適切な仕様をもっている．しかし，もともと仮想的な言語であるために，理解を深めるためにはシミュレータを用いて実習する必要がある．このシミュレータとしてはこれまでに数種が発表されているが，そのほとんどは汎用性に欠ける．また，操作の点でも問題が多い.

そこで本システムの開発に当たっては，現時点での主流 OS である UNIX あるいは MS-DOS を搭載していれば機種を問わずに実行できるシステムを目標に置いた．また，初心者を対象にするため，アセンブルからオブジェクトプログラムの実行までを一括して行うシステムを考慮した.

表1 CASL の言語仕様

第1語				第2語	アセンブラ命令	
OP		GR	XR	adr		
主OP	副OP					
0	0				未使用	
1	0				LD	GR,adr,XR
	1				ST	GR,adr,XR
	2				LEA	GR,adr,XR
2	0				ADD	GR,adr,XR
	1				SUB	GR,adr,XR
3	0				AND	GR,adr,XR
	1				OR	GR,adr,XR
	2				EOR	GR,adr,XR
4	0				CPA	GR,adr,XR
	1				CPL	GR,adr,XR
5	0				SLA	GR,adr,XR
	1				SRA	GR,adr,XR
	2				SLL	GR,adr,XR
	3				SRL	GR,adr,XR
6	0	F			JPZ	adr,XR
	1	F			JMI	adr,XR
	2	F			JNZ	adr,XR
	3	F			JZE	adr,XR
	4	F			JMP	adr,XR
7	0	F			PUSH	adr,XR
	1		0	F F F F	POP	GR
8	0	F			CALL	adr,XR
	1	F	0	F F F F	RET	

以上の2点を設計の主柱に据えて開発したシミュレータを “FINE CASL”<Fast, Independent & Noble Educational system for CASL>(以下，FINE と略す) と称する.

2—1 FINE の動作環境

FINE としては前述のように UNIX および MS-DOS のもとで動作する2つの版を開発した．前者は現在のところ FACOM S-3500 (富士通) のUNIXシステムでしか実行できないが，移植は容易である．また，後者は市販の任意のMS-DOS パソコンで動作する.

2—2 FINE の構成

FINE の開発は，UNIX 上においてC言語を用いて行った．Cは移植性が高いため MS-

DOS 版への展開も容易であった．MS-DOS 版のソースサイズは36510バイト，オブジェクトサイズは32370バイト（Lattice C Ver. 3.0 による）である．

FINE の設計に当たり，前述のようなアセンブリ入門教育を効率的に行うため，次の点の実現を図った．

1) 操作ができるだけ単純になること
2) ソースプログラムの入力にフリーフォーマット方式を採用すること
3) エラーメッセージを豊富にすると共に，トレース機能を充実させること
4) プログラムや実行結果などがファイル形式で保存され，いつでも再利用できるようにすること
5) 実行を高速に行うこと

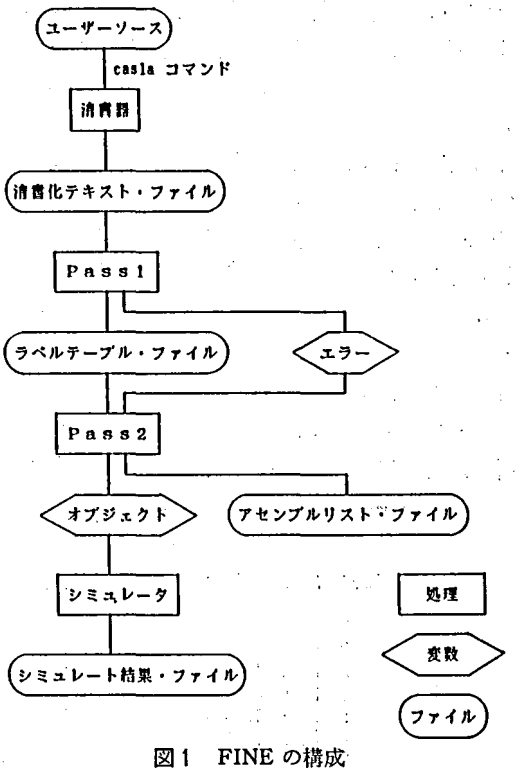


図1　FINE の構成

シミュレータは大別して図1のように4つのルーチンから成る．各ルーチンの処理概要は次のとおりである．

(1) 清書器

このルーチンでは，エディタを用いてフリーフォーマットで記述した CASL ソースプロ

```
start
lea gr1,0 ;GR1=0
lea gr1,0,gr1 ;GR3=GR1
call .rdmpd ;register dump
lea gr1,1 ;GR1=1
lea gr2,0,gr1 ;GR2=GR1
call .rdmpd ;register dump
st gr3,wrk ;GR3=(WRK)
loop add gr1,wrk ;GR1=GR1+(WRK)
call .rdmpd ;register dump
st gr2,wrk ;(WRK)=GR2
lea gr2,0,gr1 ;GR2=GR1
cpa gr1,c10000 ;if GR1<10000
jmi loop;  goto loop
exit ;end
wrk ds 1
c10000 dc 10000
end
```

(a) CASL ソースプログラム

```
start
lea     gr1,0       ;GR1=0
lea     gr3,0,gr1   ;GR3=GR1
call    .rdmpd      ;register dump
lea     gr1,1       ;GR1=1
lea     gr2,0,gr1   ;GR2=GR1
call    .rdmpd      ;register dump
st      gr3,wrk     ;GR3=(WRK)
loop    add     gr1,wrk     ;GR1=GR1+(WRK)
call    .rdmpd      ;register dump
st      gr2,wrk     ;(WRK)=GR2
lea     gr2,0,gr1   ;GR2=GR1
cpa     gr1,c10000  ;if GR1<10000
jmi     loop        ;  goto loop
exit                ;end
wrk     ds      1
c10000  dc      10000
end
```

(b) 清書化テキスト

図2　清書器による変換の一例

グラムを，標準形式（清書化テキスト）に変換する．具体的にはラベル・命令コード間などの区切りをTABに，英小文字を大文字に置き換える。図2に変換の前後のテキスト例を示す。

(2) Pass 1

清書化テキストをもとに図3に示すようなラベルテーブルを作成する．ラベルテーブルはラベルとそれに対応するアドレスの表であり，区切り記号はTABである．

```
.RDMPD  F020        LOOP    000E        WRK     001C
C10000  001D
```

図3 ラベルテーブル

(3) Pass 2

清書化テキストおよびラベルテーブルからCOMETの機械語を生成する．また，機械語とソースの対応リストであるアセンブルリストを画面に表示したり，ファイルとしての保存も行う．図4はアセンブルリストの一例である．

```
*****  CASL CROSS ASSEMBLER BY LIP(1986)  *****

LINE  ADD.  OBJECT     LABEL OPERATION OPERAND      COMMENT

   1                          START
   2  0000  12100000          LEA     GR1,0        ;GR1=0
   3  0002  12310000          LEA     GR3,0,GR1    ;GR3=GR1
   4  0004  80F0F020          CALL    .RDMPD       ;register dump
   5  0006  12100001          LEA     GR1,1        ;GR1=1
   6  0008  12210000          LEA     GR2,0,GR1    ;GR2=GR1
   7  000A  80F0F020          CALL    .RDMPD       ;register dump
   8  000C  1130001C          ST      GR3,WRK      ;GR3=(WRK)
   9  000E  2010001C  LOOP    ADD     GR1,WRK      ;GR1=GR1+(WRK)
  10  0010  80F0F020          CALL    .RDMPD       ;register dump
  11  0012  1120001C          ST      GR2,WRK      ;(WRK)=GR2
  12  0014  12210000          LEA     GR2,0,GR1    ;GR2=GR1
  13  0016  4010001D          CPA     GR1,C10000   ;if GR1<10000
  14  0018  61F0000E          JMI     LOOP         ;     goto loop
  15  001A  64F0F0B0          EXIT                 ;end
  16  001C  0000      WRK     DS      1
  17  001D  2710      C10000  DC      10000
  18                          END

ASSEMBLE END                          30 WORDS
```

図4 アセンブルリストの一例

(4) エミュレータ

Pass 2において生成された機械語をエミュレートする．FINEにはもとのCASLには定義されていないレジスタダンプなどのシステムサブルーチンを持たせた．これによりメモリーやレジスタの内容把握が容易になった．図5にレジスタダンプを伴ったエミュレート結果を示す．

なお，これらの処理はcasla というコマンドを入力するだけで連続して

```
GR0=0   GR1=0   GR2=0   GR3=0   GR4=8192
GR0=0   GR1=1   GR2=1   GR3=0   GR4=8192
GR0=0   GR1=1   GR2=1   GR3=0   GR4=8192
GR0=0   GR1=2   GR2=1   GR3=0   GR4=8192
GR0=0   GR1=3   GR2=2   GR3=0   GR4=8192
GR0=0   GR1=5   GR2=3   GR3=0   GR4=8192
GR0=0   GR1=8   GR2=5   GR3=0   GR4=8192
```

図5 エミュレート結果の一例

行われる．すなわち，操作性は極めてよいといえる．

2—3 FINE の性能テスト結果

本システムの実行速度を評価するために，FM16βFDⅡ（富士通）の RAM ディスク上で100行のソースプログラムをアセンブルしたところ約10秒を要した．この間，アセンブル時にはリスト出力も同時に行っているので，処理時間の大半は画面表示に費やされていると考えられ，アセンブル自体は高速に行われているといえる．

また，各命令の実行時間測定結果を表2に示す．測定に当たっては，各々の命令についてn回のループの所要時間を求め，次式に従って値を得た．なお，MS-DOS版は FM16β FDⅡの RAM ディスク上で測定した．

表2　各命令の実行時間

命令	所要時間（μsec）		命令	所要時間（μsec）	
	MS-DOS版	UTS版		MS-DOS版	UTS版
LD	131	60	CPA	160	75
ST	130	65	CPL	138	65
LEA	159	70	SLA	161	75
ADD	162	70	SRA	162	75
SUB	159	80	SLL	163	80
AND	160	75	SRL	165	80
OR	162	80	PUSH·POP	253	130
EOR	163	75	CALL·RET	331	155

$$\frac{\text{（命令を含むn回ループ所要時間）} - \text{（n回空ループ所要時間）}}{\text{n回}}$$

各命令の実行時間の平均は，MS-DOS では約 155 μs，UNIX 版では約73 μs である．1命令を1 μs 前後で処理する一般の16ビットマイクロプロセッサと比較すると遅いが，1秒間に6000～16000命令を実行できることから，エミュレータとしてはかなり高速であり，教育面では全く問題ないと考えられる．

また，UNIX 版は MS-DOS 版の約2倍のスピードを持つが，これは対象とした UNIXマシンが32ビットであることが要因であろう．

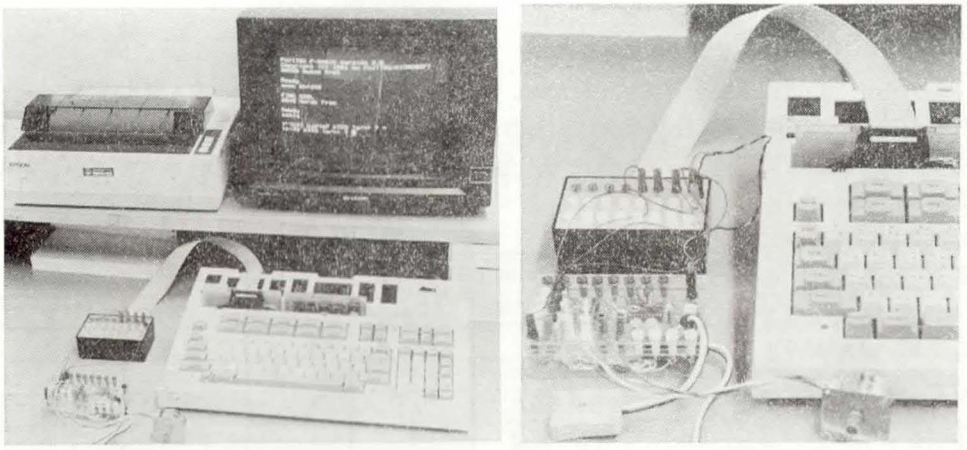
3. CASL による機械制御学習システム

前節で述べた FINE により，アセンブリ言語の入門教育は大いに効果をあげうる．しかし，この種の教育では機械制御の基礎を教授することが重要であり，これへの展開を図る必要がある．そこで前述の FINE をさらに拡張すると共に，ハードウェアコントロールを理解させるシステムとして，8ビットパソコンを用いた制御実験の学習システムを構築した。設計の方針を次の4点に置いた．

1) 制御対象をモジュール化し基礎的な回路が簡単に扱えるようにする．
2) アセンブルから機械制御への実行過程が理解しやすいようにする．
3) 初心者が犯しやすいトラブルに対処しやすい形態をとる．
4) アセンブルから実行までの操作が簡単で，かつ，各種の制御に対応できる速度を保証する．

システムの概観を図6に示す．

以下にこのシステムのソフトウェア部・インタフェース部・ハードウェア部などの概要を



(a) 全　　体　　(b) 機械制御部

図6　機械制御学習システムの概観

示す.

3—1　制御実験パソコン用シミュレータ

先ず，前述の FINE を8ビットパソコン FM-7 で実行できるように移植した．これには FM-7 の CPU である 6809用の DOH-C コンパイラを用いた.

このシミュレータは前節の UNIX および MS-DOS 版とほぼ同じであるが，入出力ファイルが使われない点が大きな違いである．すなわち，ソースプログラムは FM-7 の BASIC インタープリタを利用して，BASIC の注釈文として入力することになり，結果は画面と同時にプリンタにも出力可能である．(UNIX および MS-DOS 版ではファイル転送が必要であった.)また，制御学習を容易にするため，本システム独自の命令としてローテイト命令や,

```
FINE CASL
2048 Words Free

Ready
```

```
LINE  ADD.  OBJECT      LABEL OPERATION OPERAND
   1                          START
   2  0000  1210000A          LEA     GR1,10
   3  0002  B0F0F020    LOOP  CALL    .RDMPD
   4  0004  1211FFFF          LEA     GR1,-1,GR1
   5  0006  62F00002          JNZ     LOOP
   6  0008  64F0F0B0          EXIT
   7                          END

ASSEMBLE END                                  10 WORDS
```

```
100 ' start
110 ' lea gr1,10
120 'loop call .rdmpd
130 ' lea gr1,-1,gr1
140 ' jnz loop
150 ' exit
160 ' end
```

```
GR0=0   GR1=10  GR2=0   GR3=0
GR0=0   GR1=9   GR2=0   GR3=0
GR0=0   GR1=8   GR2=0   GR3=0
GR0=0   GR1=7   GR2=0   GR3=0
GR0=0   GR1=6   GR2=0   GR3=0
GR0=0   GR1=5   GR2=0   GR3=0
GR0=0   GR1=4   GR2=0   GR3=0
GR0=0   GR1=3   GR2=0   GR3=0
GR0=0   GR1=2   GR2=0   GR3=0
GR0=0   GR1=1   GR2=0   GR3=0
```

(a) ソースプログラム　　(b) アセンブルおよび実行結果

図7　6809用 FINE の実行例

I/O ポートの情報等を表すシステムラベルの拡張を図った.

なお, 本ソフトウェアの ROM 化により, コンピュータ起動時に即座に FINE の環境に移行できる. 図7はソースプログラムの一例とその実行結果である.

3—2　インタフェースカード

8ビット CPU と制御対象とのインタフェースをとるために, PPI (Programmable Peripheral Interface) カードを作製した.

PPI カードの中心は 8255A である. このため, I/O ポートとして 24 ビットとることができ, 多彩な制御が望める. 回路は 8255 を含め IC 3 個で構成しているので, 製作しやすく, 故障もしにくい.

また, 外部機器との接続を考慮し, +5 V および GND をコネクタに出力した. コネクタには, この電源ラインとともに, A, B, C の3つのポート各8ビット計24ビットを利用するため26ピンのコネクタを用いた.

この概観を図8に示す. 図の右が PPI カード, 左は次に述べる制御モジュールとの中継器である. 中継器は操作手順を容易にするために作成したもので, これには, 3つのポートおよび +5 V および GND 端子が用意されている.

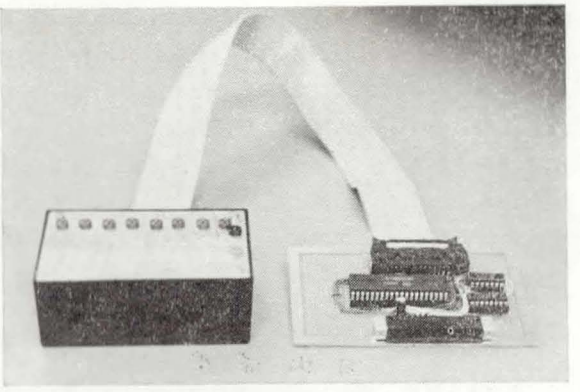


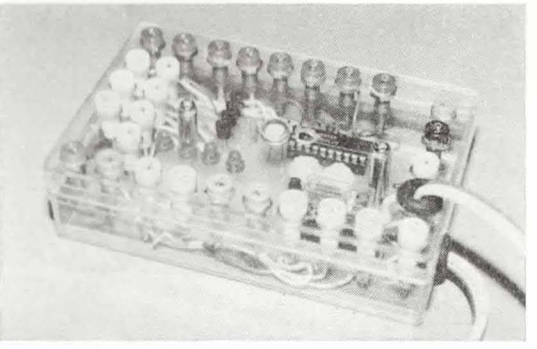
図8　中継器と PPI カード

3—3　制御モジュール

今回 CASL による機械制御の実験対象としたものは次の5点である。

1) LED の点滅制御
2) リレー制御
3) ステッピングモータの駆動
4) 光センサを用いた A/D 変換
5) センサとアクチュエータの複合制御

本システムでは, 制御対象が初心者にとっても簡単に取り扱えるようにするため, 上記の実験対象（駆動対象・センサおよび駆動回路）をすべて一体化した.



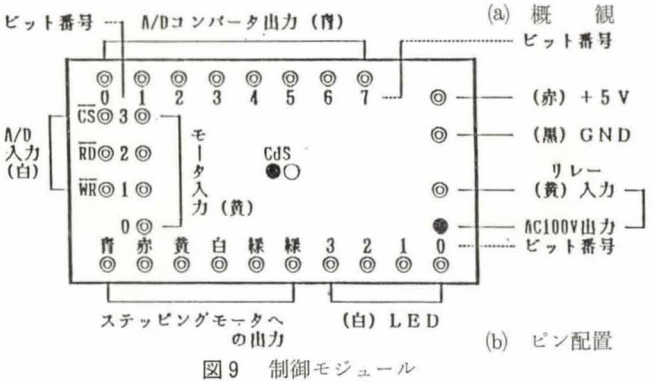


図9　制御モジュール

これを制御モジュールと称する．しかし，全くブラックボックスとしてしまうことは望ましくないので，プラスティックケースにコンパクトに埋め込み，回路の様子が理解できるように設計した．制御モジュールの概観とモジュールのピン配置を図9に示す．なお，制御モジュールは多数台必要なため，プリント基板化を行って対処した．

ユーザーは制御モジュールの適当なピン位置と中継器を接続し，CASL プログラムを実行させることにより，容易に制御の実態を把握できる．

4. むすび

アセンブリ言語の習得を志す初心者にとってのソフト・ハード両面での一貫した教育システムを開発し，それが教育用として十分満足いく結果を得たことを報告した．

本システムの特色のひとつはハード部分のモジュール化であり，上述の実験対象以外の制御も，このモジュール部分を適宜構成していくことにより可能となる．当初は初心者教育のためのシステムとして開発したが，さらに，高度な分野への発展も考慮できよう．

おわりに，本システムの構築に当たっては，本校機械工学科情報研究室に在籍した花石昌文（現長野富士通ソフト）・千原隆夫（現富士通）両君の努力が大きい．ここに深く感謝の意を表する．

参考文献

(1) 山之上・風間・堀内：機械技術者のためのマイコン制御入門，日刊工業新聞社（1983）
(2) I/O：ROM カード，工学社（1984，Vol. 1）