

CASL シミュレータの開発とアセンブラ教育

松田 勲* 松岡 輝 齋藤 聡 村上紀之

Development of CASL simulator and assembler training

Isao MATSUDA Akira MATSUOKA Satoshi SAITO Toshiyuki MURAKAMI

The CASL assembler system was developed for use in assembler language education and to help students understand the internal processing of a computer. This system consists of two parts, the CASL assembler and a COMET simulator-tracer. In designing the system, stress was laid on the trace function so as to allow the student to study on the CASL while confirming the internal movement of the data in the computer each time a command is executed. This system was used in practical training at the Faculty of Electronic Engineering.

1. 序論

近年、コンピュータは LSI 技術の発達などにより、高性能、低価格化の傾向にあり、特定の技術者だけでなく、一般の人々のなかにも広く普及して来た。しかし、これらの人々が、なにかコンピュータに仕事をさせようとするとき、その様々な仕事に対するソフトウェアが必ずしも用意されているとは限らない。その場合、自分自身でソフトウェアの開発を行うことが必要となり、そのためには、プログラミング言語の習得を必要とする。このようなとき、ユーザーは、アセンブラ言語よりも、BASIC 等の高級言語を用いて、プログラミングを始める場合が多いと思われる。高級言語は、アセンブラ言語などに比べて処理速度は遅いが、最近のコンピュータの高性能化により、それほど大きなプログラムでない限り、十分な処理速度を実現してくれるし、高級言語の方が何と云っても分かり易い。しかし、高級言語は、ハードウェアを意識しないでプログラミングができるため、コンピュータの動作を理解することを目的とする場

合、あるいは制御用のプログラムを組む際には、ハードウェアと直接的な関係をもつ、アセンブラ言語や機械語を理解する必要がある。CASL は、昭和 62 年より、通産省情報処理技術者試験に使用されているアセンブラ言語で、仮想コンピュータ COMET のためのプログラミング言語である。このアセンブラ言語は、機械語命令が 23 個と少なく、基本的なアセンブラ言語であるため、アセンブラをこれから学ぶ人にとって、適したものであると思われる。

本研究では、電子工学科におけるアセンブラ言語の教育とコンピュータの動作の理解を深めることを目的として、この CASL をパーソナルコンピュータ (PC-9800 シリーズ) 上でシミュレートできるように、CASL アセンブラシステムを開発した¹⁾²⁾。CASL アセンブラとして市販されているもの³⁾もあるが、これらは、教育目的が、筆者らが意図したものと、必ずしも合致したものではない。本システムは、アセンブラ言語の入門者が、容易に CASL および COMET について理解できるように工夫しており、情報処理技術者試験の受験者にも十分学習の助けとなり得るものである。なお、

本システムのプログラミングは、松岡、齋藤、村上が担当し、総括およびこれを用いた実習は、松田が担当した。

2. COMET 計算機とアセンブラ言語 CASL

情報処理技術者試験では、プログラム作成能力を試す言語として、COBOL, FORTRAN, PL/I, アセンブラ (CASL) の 4 言語を指定している。このなかで、CASL は、第 2 種試験においては選択であるが、第 1 種試験においては必須であり、第 1 種試験合格を目指すものにとって、習得しなければならない基本的な言語である。

CASL は、仮想計算機 COMET のアセンブラ言語であるが、擬似命令、マクロ命令、機械語命令からなっており、スタック、比較演算機能、分岐命令、シフト命令など豊富な命令と、機能を有している。また、CASL はハードウェアに直結したアセンブラである。ハードウェアとしての COMET は、1 語 16 ビットの計算機であって、アクセスできるアドレスは、0 番地から 65535 番地までである。記憶装置と制御装置との相互のデータの受け渡しには、レジスタを使用する必要があるが、COMET においては、数種のレジスタのうち、プログラムカウンタ (PC)、フラグレジスタは専用レジスタであるが、これらを除いた、アキュムレータ、指標レジスタ、スタックポインタなどは、汎用レジスタと兼用する。したがってレジスタは、一機能としてではなく、多様に利用される。

3. システムの設計および製作

本システムは、アセンブラ言語の入門者および情報処理技術者試験の受験者を対象としているため、アセンブラの実用性よりも、CASL を理解することに重点をおいて、設計、製作を行った。そのために、命令を実行している様子 (レジスタ、メモリ、スタックなどデータの変化の様子) を表示するトレーサの機能を重視した。本システムは、アセンブラ言語を機械語に翻訳するための、CASL アセンブラと、COMET 計算機をシミュレートし、その様子をトレースするための COMET シミュレータ、トレーサより構成されて

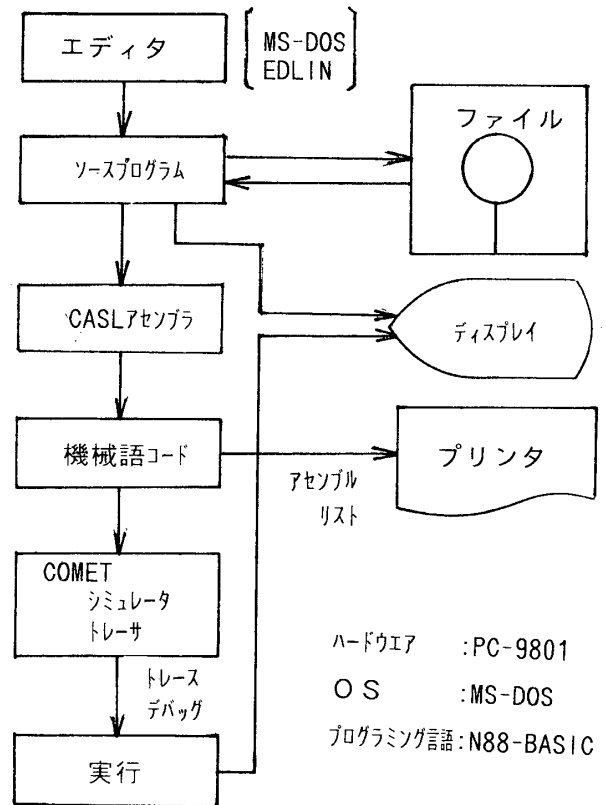


図 1 システム構成

いる。実際に使用する際には、このほかにアセンブラプログラムを作成するためのエディタも必要であるが、MS-DOS* 提供のものを基本的に利用することとし、少しの機能を付け加えた程度にとどめた。また、システムを開発し、実際に使用するためのハードウェアとして、NEC 社製 PC-9801 を使い、OS は MS-DOS、プログラミング言語は N 88-BASIC を用いた。図 1 にシステムの構成図を示す。本システムの処理の種類としては、①ソースプログラムの作成および修正、②セーブ、③ロード、④アセンブル、⑤アセンブル結果の出力、⑥デバッグ、⑦トレース、⑧実行等を用意したが、これらの処理は、すべて図 2 に示すようなメニュー画面で選択できるようになっている。

3.1 CASL アセンブラ

CASL アセンブラの機能は、アセンブラ言語 CASL を用いて作成したプログラムを、仮想計算機 COMET が理解できるように、機械語に翻訳し、オブジェクトファイルを生成することであり、以下の機能をもたせた。

①文法上の誤りのチェックとエラー表示

- ② IN, OUT マクロ命令の展開
- ③ 命令部のコード化
- ④ オペランド部のコード化
- ⑤ ラベル名のアドレス化
- ⑥ アセンブル結果の画面表示

アセンブル結果の画面表示の例を、図 3 (A), (B) に示す。ここでは、エディタを用いて作成したソースプログラムが、便宜上つけた行番号とともに画面の右側に示される。同じ行の左側には、対応する機械語コードが、それが記憶されるアドレスとともに示されている。また、アセンブルによって、

アドレス	機械語コード	行番号	ソースプログラム
すこし時間がかかります			
000		100	PROB10 START
002	1010 0020	110	LD GR1, DATA1
004	1020 0021	120	LD GR2, DATA2
006	1030 0022	130	LD GR3, DATA3
008	4010 0021	140	CPA GR1, DATA2
00A	6100 0018	150	JMI NEXT2
00C	4010 0022	160	CPA GR1, DATA3
00E	6100 0014	170	JMI NEXT1
010	1110 0023	180	ST GR1, KOTAE
012	6400 001E	190	JMP OWARI
014	1130 0023	200	NEXT1 ST GR3, KOTAE
016	6400 001E	210	JMP OWARI
018	4020 0022	220	NEXT2 CPA GR2, DATA3
01A	6100 0014	230	JMI NEXT1
01C	1120 0023	240	ST GR2, KOTAE
01E		250	OWARI EXIT
020	000F	260	DATA1 DC 15
021	0009	270	DATA2 DC 9
022	0007	280	DATA3 DC 7
023		290	KOTAE DS 1
024		300	END

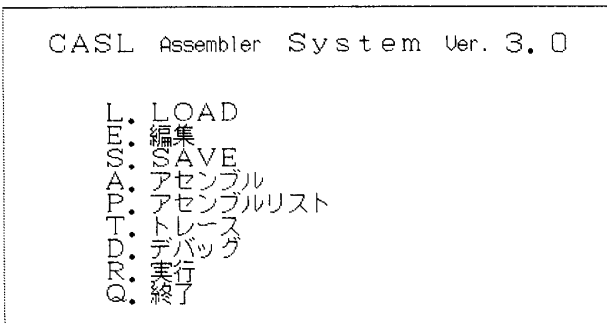


図 2 メニュー画面 各行の先頭の記号を入力することによって、処理が選択できる。

L E
アセンブルエラーはありません
キーをどれか押してください

図 3 (A) アセンブル結果の画面表示 (ソースプログラムに誤りのない場合) エディタでは、実線 E より右側部分のみ表示される。また、ソースプログラムを、ディスクから“LOAD”したときは、行番号が付け加えられて、実線 L より右側の部分が表示される。なお、実線 E および L は、実際の画面では表示されない。

000		100	PROB10	START		
002	1010 0020	110		LD	GR1, DATA1	
>004	1000 0021	120		LD	PR2, DATA2	<GR>レジスタ
006	1030 0022	130		LD	GR3, DATA3	
008	4010 0021	140		CPA	GR1, DATA2	
00A	6100 0018	150		JMI	NEXT2	
00C	4010 0022	160		CPA	GR1, DATA3	
00E	6100 0014	170		JMI	NEXT1	
>010	1110 0000	180		ST	GR1, KKTAE	オペランドのラベルが見つからない
012	6400 001E	190		JMP	OWARI	
014	1130 0023	200	NEXT1	ST	GR3, KOTAE	
016	6400 001E	210		JMP	OWARI	
018	4020 0022	220	NEXT2	CPA	GR2, DATA3	
>01A		230		JMH	NEXT1	命令
01C	1120 0023	240		ST	GR2, KOTAE	
01E		250	OWARI	EXIT		
>020	0000	260	DATA1	DC	#F	オペランド
021	0009	270	DATA2	DC	9	
022	0007	280	DATA3	DC	7	
023		290	KOTAE	DS	1	
024		300		END		

エラーの数 = 4
キーをどれか押してください

図 3 (B) アセンブル結果の画面表示 (ソースプログラムに誤りのある場合) 誤りが発見された行を>で示し、エラー内容に関する簡単なメッセージが表示される。

ラベル名も同時にアドレス化されている。

3.2 COMET シミュレータおよびトレーサ

COMET シミュレータでは、プログラムの実行は、プログラムカウンタが指示するアドレスからプログラム中の命令を取り出し、解読することから始まる。「一命令の実行が終了するごとにプログラムカウンタに2を加える……」などプログラム実行の過程として、記憶装置のなかにどのような形でデータが格納され、どのように記憶装置から取り出されるのか、命令はどのように実行されていくのかなど、計算機の動作過程を、トレースモードで体験できるようにした。

トレースモードでは、実行開始番地からコードを読み、それによって与えられる動作を1ステップずつ実行して行く。このときトレーサは、実行の様子が分かり易いように、1行実行するごとに、実行している命令とそれに対応する機械語コード、レジスタおよびメモリの内容をディスプレイに表示する。そして適当なキーを押すことによって次のステップに移行する。トレースモードで実行中の例を、図4に示す。なお、適当にキーを押

す代わりに、場合によって、数字をキー入力すれば、その秒数だけ停止して次の命令に進む。

また、デバッグモードで実行するとソースプログラムとレジスタおよびPCの内容を、一覧表の形で画面表示し、これによりプログラム実行過程で変化するレジスタ等の内容を追えるようになっている。トレースモード、デバッグモードいずれの場合も、プログラムにエラーがある場合は、アセンブルエラーがあることを表示して先へは進まない。

実行モードであれば、IN, OUT 命令を実行し、レジスタ、PCの最終的な実行結果を、ディスプレイに表示する。

3.3 エディタ

ソースプログラムを作成するためにエディタを使用するが、本システムでは基本的には、MS-DOS 提供のエディタ EDLIN を使用してプログラムの作成を行う。

4. 電子情報工学実験 II における実習

「電子情報工学実験 II」(電子工学科3年次、後

アドレス	行番号	ソースプログラム	レジスタ (16進表示)	(10進表示)	レジスタ (16進表示)	(10進表示)
000	100	PROB10 START	GR0=X'0000'	0	GR1=X'000F'	15
002	110	LD GR1,DATA1	GR2=X'0009'	9	GR3=X'0007'	7
004	120	LD GR2,DATA2	GR4=X'003A'	58	PC=X'000E'	14
006	130	LD GR3,DATA3	FR = 0	← フラグレジスタ	↑ プログラムカウンタ	
008	140	CPA GR1,DATA2	OBJ-CD=4010	0022	← 現在実行中の機械語コード	
00A	150	JMI NEXT2	DATA1 =000F			
>00C	160	CPA GR1,DATA3	DATA2 =0009			
00E	170	JMI NEXT1	DATA3 =0007			
010	180	ST GR1,KOTAE	KOTAE =0000			
012	190	JMP OWARI				
014	200	NEXT1 ST GR3,KOTAE				
016	210	JMP OWARI				
018	220	NEXT2 CPA GR2,DATA3				
01A	230	JMI NEXT1				
01C	240	ST GR2,KOTAE				
01E	250	OWARI EXIT				
020	260	DATA1 DC 15				
021	270	DATA2 DC 9				
022	280	DATA3 DC 7				
023	290	KOTAE DS 1				
024	300	END				

Q:終了 数字:スリープタイム R:リセット 他:続ける
 ヌッて=

15:33:28

図4 トレースモードで実行中の画面 左半分は、ソースプログラムで、>は現在実行中の行を示しており、この例では、160行を実行中である。右半分には、この行を実行後のレジスタの内容、対応する機械語コード、ラベルで指定された領域の内容を示す。

期)において本システムを用いて実習を行った。基本的で入門者向けレベルの問題から、アセンブラ言語をある程度理解できている中級者向けレベルの問題まで、各レベルのものを、あらかじめ用意し、各学生に、各レベルの問題を含めて8問ずつ課題として与えた。課題は2週間で修了するものとし、提出レポートには、プログラムリスト、プログラム作成の考え方、フローチャートとそれに対応する説明を記述することを求め、提出時にはその場で、筆者が指定する2~3課題について実際にCASLを起動し、トレースモードでディスプレイの表示を追いながら、プログラムの実行過程の説明を求めた。システムの操作と、命令の種類が少なく全体で10行程度のごく基本的な問題については全員習得できていた。これより命令の数が多くなったり、ラベルや、入出力命令を使うものになると、理解の程度は、個人差が大きかった。理解の程度の低い方では、10進数から16進数への変換ができなかったり、データの記憶場所とアドレスとの関係が不明であったりするものもあった。高い方では、課題では要求していないようなところまで自分で工夫を凝らして、自信たっぷり説明するものなど、千差万別であったが、ここであげたような例はわずかである。多くは、CASLの入門書や、情報処理技術者試験の受験参考書等に出ているものを参考にしたり、あるいは友人から流用したか、一つの問題に対しての解答プログラムは、問題の解釈の仕方、考え方によって、2~3タイプの解答に分類できた。なかには、ほとんど同じ、あるいは、変数名が変わっている程度のものもみられた。説明を求めた時点では、課題として与えた問題については、一応の理解を示してはいたが、他の類題を解ける力があるかどうか疑問に思われる学生もいた。いずれにせよ、時間が足りないようである。

また学生のパソコン使用状況や、本システムの改良のための資料を得るために、次の項目について、調査した。

①パソコン歴 ②主に使う言語 ③主に使うエディタ ④主に使うハードウェア ⑤使っている周辺機器 ⑥主に使うOS ⑦CASLシミュレー

タを使った感想 ⑧あれば良いと思う機能 ⑨内蔵のエディタについて ⑩問題について ⑪あれば良いと思う問題 ⑫その他、上記項目についての補足

調査結果を要約すると

(イ)パソコン歴については、86%のものが、3年半以内ということで、ほとんどの学生は、大学に入ってからパソコンに触れるようになったこと、そのうちの多くは授業のみで使った経験者であることがわかった。4年以上の経験者は、14%で、数は少ないが、経験がそれ以下のものにくらべて、やや理解度がよかった。

(ロ)主に使う言語としては、Basic, Pascalが圧倒的に多かった。Pascalは、電子工学科では1年次の「ソフトウェア工学概論」、「情報処理演習」でプログラミング言語として教育している。Pascalの使用率を1としたとき、Basicは1.7, Fortranは0.17, Cは0.29であった。

(ハ)エディタとしては、MIFESを使っているものが多く、ワープロソフト一太郎をエディタとして使っている場合もほぼ同数であった。

(ニ)OSは圧倒的にMS-DOSの利用が多かった。

(ホ)今回製作したCASLシミュレータについては、「使い易い」10%、「使いにくい」16%、「その他」74%であった。これは内蔵のエディタについての、「使い易い」6%、「使いにくい」21%、「その他」73%とほぼ同様の割合であることから、エディタを改良することがシミュレータの使い易さの改良につながると思われる。またエディタの機能として追加要望の多かったのは、「改行機能」、「データのクリヤ」(エディタの内容を一度で全部消去)であった。またシステムに対しては、「ファイルの一覧機能」、「トレースのときに元の行にもどる機能」、「トレースのときに日本語で分かり易く表示される機能」、「エラーについてもっと詳しい表示」等の要望があった。

(ヘ)問題については、「むずかしい」72%、「まあまあ」28%、「やさしい」0%であったが、おそらく、今回初めてアセンブラ言語でプログラミングをしたというものがほとんどであろうと思われる

ので、そういう点からも2週間という期間は、命令の数が少ないとはいえ、CASL アセンブラを理解するには短かすぎると思われる。

5. 反省および今後の課題

実習を行った結果では、機械語コードとアドレスづけ、または、メモリ上のデータとの関係の理解が不足している。例えば、レジスタを用いてアドレスを指定するような場合、アドレスの指定の仕方によっては、メモリの内容が変わってしまうことなど…。

今回は、アセンブラシステムの製作に時間を注ぎ、それに比べて、問題づくりにはそれほど時間をかけなかった。今回は、CASL そのものの理解を深める目的で、プログラミングを目的としたような問題を出したが、今後はさらに、アセンブラ教育導入の目的とする計算機（機械語）とのつながりが理解できるような問題も含めるようにしたい。ワンボードワイコンなどでは、ニーマニックでプログラムを組んでから、人間が機械語にアセンブルする（ハンドアセンブル）ことで、機械語コードのアドレスづけや、ラベルのアドレス化等の理解、体験ができるが、それに代わるものを工夫したい。このような学習過程を経て、計測・制御等への応用についての考え方や、理解が深まり、インターフェースの作成などへの発展につながればよいと考えている。また逆に、BASIC 等の高級言語との対応で、アセンブラ言語の特長の理解の助けとなるような課題づくりも試みたい。

システムについては、COMET シミュレータは仕様通りのものができており、現在のところ問題はないと考えているが、学習上の便宜のために作ったトレーサ機能や、エディタについては改良等を進める必要がある。

本システムは、現在はパソコン PC 9801 上で動くが、多人数教育を考えた場合、VAX に移植することも考えている。

6. 結論

本システムの開発によって、パーソナル・コンピュータ PC-9801 上で CASL および COMET

がシミュレーションできるようになり、電子工学科で実習に用いた。本システムを使用することにより、アセンブラ言語の入門者や、情報処理技術者試験の受験者が、参考書などだけで CASL を学習するよりも、実際に仮想コンピュータ COMET の実行過程を体験することにより、理解がより効率的になると思われる。本システムの特徴をまとめると

- ①アセンブラ言語 CASL の仕様に従って、エディタでソースファイルを作成し、実際にアセンブラを動作させることで、プログラムの構造、文法および命令の使い方などを効率良く学習できる
- ②仮想的に COMET を作り、COMET 上でトレースモードによって1命令ごとの動作を確認しながら学習できる
- ③必要な各種のメッセージ、画面表示はできるだけ日本語を用いたため分かりやすいものになっている。

本システムを利用することにより、計算機のハードウェアとソフトウェアの接点に関する理解が深まり、インターフェースの設計などの入門に対して、理解の助けとなることを期待している。

なお、本研究は、日本私学振興財団の私立大学等経常費補助金特別補助「特色ある教育研究」を受けて行われた研究の一部であり、ここに関係各位に深く感謝する。

参考文献

- 1) 松田, 松岡:「CASL アセンブラシステムの開発」, 第 49 回応用物理学学会学術講演会 (1988) 秋期
 - 2) 松田, 斎藤, 村上:「CASL アセンブラシステムの開発 II」, 第 37 回応用物理学関係連合講演会 (1990) 春季
 - 3) 例えば, NEC パーソナルコンピュータ PC-9800 シリーズ COMET シミュレータ「PC CASL」
- 本システムを製作するにあたり全体的に参考にした文献は
- 4) 下条哲司:「CASL」, オーム社 (1986)
 - 5) 杉原敏夫:「COMET と CASL」, オーム社 (1987)
 - 6) 日高哲郎:「アセンブラ CASL 演習」, 共立出版 (1987)
 - 7) 八鍬幸信:「CASL の完全解析」, 技術評論社 (1989)
 - 8) NEC パーソナルコンピュータ PC-9800 シリーズ COMET シミュレータ「PC CASL」ユーザーズマニュアル

* MS-DOS は米国マイクロソフト社の登録商標である。