逆アセンブラの試作

鶴丸　弘昭*・吉田　博美**・高倉　聡**

Inverse Assembler

by

Hiroaki TSURUMARU

(Department of Electronics)

Hiromi YOSHIDA and Satoshi TAKAKURA

(FUJITSU Co. Ltd.)

This paper is concerned with the design and implementation of the inverse assembler which is used for analyzing MELCOM-70 system programs. The inverse assembler translates machine-code programs into symbolic assembly language. It works on a two-pass basis, that is, it makes two scans of the machine-code program. The first pass builds up a symbol table by collecting internal symbols and external symbols, and the second pass does the actual translation, using the internal and external symbols generated in the first pass.

The inverse assembler designed here has the following four features.

1. The generated internal symbols represent a certain content of each address.
2. User is able to use fifty external symbols which fit a content of each address.
3. Series internal symbols are generated even to a program separated into two segments.
4. The data word is converted into a number of four hexadecimal digits.

1. まえがき

最近，マイコンやミニコンなど計算機の急速な普及に伴い，それらの効率よい利用を目的に，システムの改良，拡張または開発に興味を持つ人が多くなってきている．このようなシステムの改良や開発を行うためには，計算機のハードウェアについての知識も当然必要ではあるが，計算機のシステム・プログラムを解析し，その仕組を理解することも不可欠だと思われる．しかし，一般にシステム・プログラムは機械語で提供される場合が多く，その解読には多くの時間と熟練を要する．そこで，機械語で表わされたプログラムの解読を容易にする手段の一つとして，「逆アセンブラ」の利用がある．「逆アセンブラ」とは，機械語で表わされたプログラムをアセンブラ語で表わされたプログラムに変換する手続き（プログラム）のことである．「逆アセンブラ」の原理そのものは，非常に簡単ではあるが，計算機の種類により，機械語やアセンブラ語が異なれば，その計算機用の「逆アセンブラ」を作成する必要がある．

我々は，卒業研究として研究室にあるMELCOM−70ミニコンピュータ用の「逆アセンブラ」の試作を行った．本論文は，それをまとめ整理したものである．「逆

昭和55年6月16日受理

*電子工学科

**富士通㈱

アセンブラ」の試作では，次の4つの点を考慮している．

(i) 計算機内で記憶場所の性格をある程度考慮してシンボルを作り出すこと．

(ii) 使用者が，解読ずみの部分に適当なシンボルを割り当てることができること．

(iii) 1つのプログラムが，分割して記憶されている場合でも，一連のシンボルを使用できること．

(iv) ラベル（シンボル）により，データ語の判定を行うこと．

2．命令語の種類と形式

MELCOM−70では，1語が16ビットで構成されており，その各ビットに，その位置に従って右から左へ番号をつけ，ビットn（n＝0〜15）と呼ぶ．語は使用目的により命令語やデータ語になったりするが，その区別はプログラムの実行により計算機がどのように取り扱うかで決まる．

MELCOM−70では75個の基本的な命令語があり，以下に示すように10種の形式に分類される（なお，アセンブラ語の表現形式と機械語との対応については参考文献6を参照）．

(1)	レジスタ・メモリ演算命令	(4個)
(2)	レジスタ・オペランド演算命令	(4個)
(3)	レジスタ・レジスタ演算命令	(8個)
(4)	レジスタ変更命令	(5個)
(5)	分岐命令	(4個)
(6)	テスト命令	(6個)
(7)	シフト命令	(6個)
(8)	入出力命令	(14個)
(9)	制御命令	(17個)
(10)	コール命令	(1個)

3．逆アセンブラ・プログラム

3．1　全体構成

試作した逆アセンブラのゼネラル・フローチャートを図1に示す．これは，2つのパス，すなわち，a）シンボル・テーブルの作成，b）アセンブラ語への変換とに大きく分けることができる．

ここで，処理番地とは逆アセンブラしたいメモリ領域の先頭番地と最終番地，外部シンボルとは使用者が適当に決めるシンボル，内部シンボルとは計算機が作り出すシンボル，メモリ参照命令とは2．で示したレジスタ・メモリ演算命令と分岐命令（ただしインデックス修飾指定のものは除く），ラベルとはメモリ上にある番地につけられたシンボル，およびニーモニック・コードとは機械語の基本的な命令機能を表現した記号（参考文献7を参照），のことである．

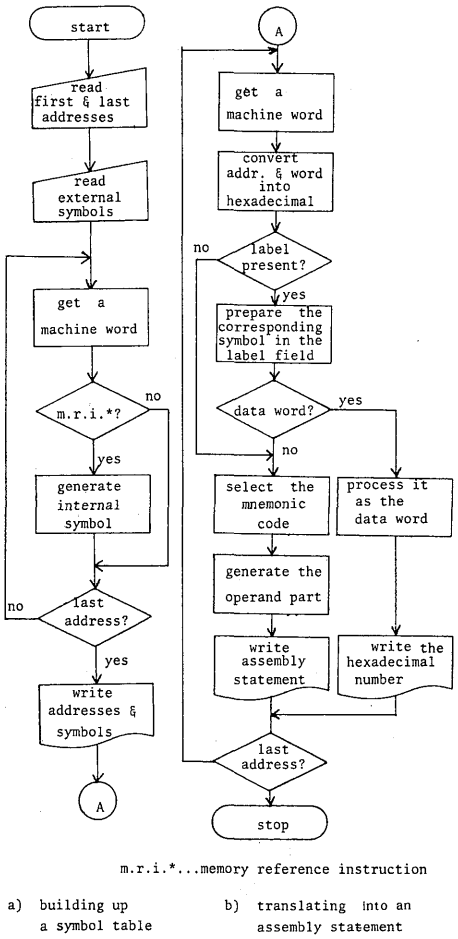

Fig. 1　General flow of the inverse assembler

このプログラムはアセンブラ言語で書かれており，手続き部が817ステップの命令語よりなり，データ部が360語の作業用領域，200語の外部シンボル・テーブル用の領域，および内部シンボルが登録されるたびに拡張される内部シンボル・テーブル用の領域よりなる．図2に，メモリ・マップを示す．

また，プログラムの簡素化や効率化のため，以下に示すような9種のサブルーチンを用いている．

(1) 内部シンボルの性格を表わす文字を作成するサブルーチン．

(2) スキップ条件に対応する記号を選択するサブルーチン．

(3) OPAe（実効アドレス）を計算するサブルーチン．

(4) レジスタのアスキー・コードを作成するサブルー

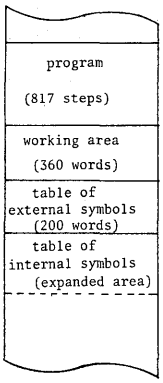

Fig. 2　Memory map of the inverse assembler

チン.

(5) ニーモニック・コードを選択し格納するサブルーチン.

(6) アスキー・コード16進数を2進数に変換するサブルーチン.

(7) 入力に関するサブルーチン.

(8) 出力に関するサブルーチン.

(9) 2進数をアスキー・コード16進数に変換するサブルーチン.

3．2　シンボル・テーブルの作成

(1) 内部シンボル・テーブルの作成

記算機内で作られた内部シンボルとそれに対応するアドレスとを組にして記憶する領域を内部シンボル・テーブルと呼ぶ. 内部シンボル・テーブルでは，図3に示すように，1つの内部シンボルに対して2語が割

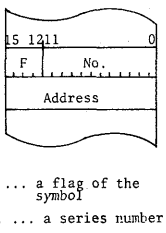

Fig. 3　Structure of an internal symbol

り当てられる.

最初の1語のビット15～12の4ビットには，内部シンボルの性格を表わす情報（F）が格納される. 内部シンボルの性格は，メモリ参照命令の種類によって次のように決めることができる. 内部シンボルの性格とFおよび出力用文字との対応表を表1に示す.

a）ロード命令（LD），加減算命令（AD，SB）の番地部に現われたら「定数」(CONSTANT).

b）ストア命令（ST）の番地部に現われたら「番地」(ADDRESS).

c）1つだけ増せという命令（ISZ）の番地部に現われたら「変数」(VERIABLE).

d）無条件分岐（BRN）の番地部に表われたら「プログラム・ポイント」(PROGRAM POINT).

e）サブルーチン・ジャンプ命令（BSS，BRS）の番地部に現われたら「サブルーチンの先頭」(SUBROUTINE HEAD).

Table 1　Bit patterns, meanings and output notations of F

bit patterns of F 15 14 13 12	meanings of F and their output notations	
0 0 0 0	constant	C
0 0 0 1	address	A
0 0 1 0	variable	V
0 0 1 1	program point	P
0 1 0 0	subroutine head	S
0 1 0 1	constant*	CK
0 1 1 0	address*	AK
0 1 1 1	variable*	VK
1 0 0 0	program point*	PK
1 0 0 1	subroutine head*	SK

*・・・indirect addressing

残りのビット11～0の12ビットは，16進3桁の一連番号（No.）を表わす. たとえば，プログラム・ポイントはP000からPFFFまで最大4096個が，間接指定の場合は，PK00からPKFFまで最大256個が順々に定義されていく.

次の1語には，その内部シンボルに対応するアドレスが格納される.

内部シンボル・テーブルへのシンボルの登録は，次のようにして行われる. すなわち，メモリ参照命令が来るたびに，実効アドレスが計算され，そのアドレスが登録済みのアドレスであるかどうかが判定され，未登録のアドレスであれば新しいシンボルが作り出され，シンボル・テーブルに登録される. シンボル・テーブルは，新しいシンボルが登録されるたびに，2語ずつ順次拡張されていく.

(2) 外部シンボル・テーブルの作成

一般に，プログラムの能率的な解析の方法は，まず

部分的に処理内容を解析し，解明された部分に適切なラベル（シンボル）を付け，それを手掛りに順次解析を進めて行くことである．ところで，内部シンボルは，その場所の性格をある程度（表1参照）表わしてはいるが，内容まで考慮したシンボルにはなっていない．そこで，試作した逆アセンブラには処理内容の解明された部分に適切なシンボルを利用者が割り当てることができるような機能を付加している．このようなシンボルを外部シンボルと呼ぶ．ただし，シンボルの定義には，少々の制限を設けている．

外部シンボルとそれに対応するアドレスとを組にして記憶する領域を外部シンボル・テーブルと呼ぶ．外部シンボル・テーブルでは，図4に示すように1つの外部シンボルに対して4語が割り当てられる．

最初の3語には，アスキー・コードで6文字以内のシンボル（ＡＣ1～ＡＣ6）が格納され，最後の1語にはそのシンボルに対応するアドレスが格納される．

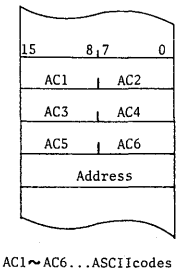

Fig. 4　Structure of an external symbol

使用者の指定により，外部シンボルが外部シンボルテーブルに登録される．外部シンボル・テーブルは，シンボルが登録されるたびに4語ずつ順次拡張されていく．このテーブルには，最大50個まで外部シンボルが格納できる．

(3)　シンボルとそのアドレスの出力領域

シンボル（内部および外部）の示すアドレスが処理番地内の場合は，出力リストにそのシンボルが現われるので，その出力リスト上でシンボルとアドレスの対応がつく．しかし処理番地外の場合は，シンボルが出力リストに現われないので，その出力リスト上ではシンボルとアドレスの対応がつかなくなる．そこで，処理番地外のシンボルが存在する場合には，出力リストの前に，そのシンボルとそれの示すアドレスとの対応を出力するようにしている．図5にシンボルとそれの示すアドレスの出力領域（11語）を示す．この出力領域は，シンボル欄（4語）とアドレス欄（2語）およびいくつかの空白の入っている語（5語）から構成されている．図5に示す内容を出力すれば，次の4．で示す図7の第8行目と同じ出力が得られる．

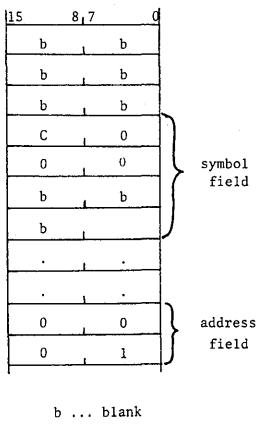

Fig. 5　Formation of the symbol and its address in output area (This output corresponds to the line (8) in Fig. 7)

3．3　アセンブラ語への変換処理

(1)　変換されたアセンブラ語の出力領域の構成

アセンブラ語に変換された命令語(ステートメント)を出力するための出力領域を図6に示す．この出力領域は，アセンブラ語のステートメントの表現形式を考慮して，一行分の出力様式として，次のような欄から構成されている．アドレス欄（3語），機械語欄（2語），ラベル欄（5語），コード欄（2語），オペランド欄（8語）およびいくつかの空白（6語）．

図6に示す内容を出力すれば，次の4．で示す図7の第13行目と同じステートメント（出力）が得られる．

(2)　アセンブラ語への変換処理

図1の(b)に示した手順に従って，アセンブラ語への変換処理を，図6の出力領域と対比させながら簡単に説明する．

(i)　アドレスと機械語の16進表示への変換

アドレスと機械語をアスキー・コードで16進4桁の数に変換して，それぞれの欄に格納する．

(ii)　ラベル選択

対象処理番地（処理の対象となっている機械語が入っている番地）と同じアドレスが，シンボル・テーブル上にアドレス（図3，図4のAddress）として

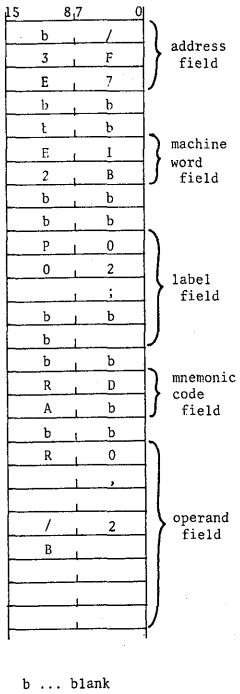

Fig. 6 Formation of the address and the machine word and the translated assembly statement in output area (This output corresponds to the line (13) in Fig. 7)

存在するかどうかを調べ，存在すればそのアドレスと対になっているシンボルをラベル欄に格納する．存在しなければ，ラベル欄には空白が入る．

(iii) データ語の判定と変換処理

一般に，定数，番地，変数および相対指定は，データとみなせるから，それらが格納されている番地にラベルが付いていれば，そのラベルによってデータ語としての判定が可能になる．以下，(a)，(b)，(c)の三つの場合に分けて語の区別を行う．

(a) ラベル欄に格納されたシンボルが，A，C，VまたはKではじまる内部シンボルであれば，その内容（機械語）は，データ語と判定され，アスキー・コードで16進4桁の数に変換され，コード欄に格納される．外部シンボルでデータ語を示す場合は，A～Fの文字ではじまるシンボルを用いればよい．

(b) ラベル欄のシンボルがサブルーチンの先頭を表わすシンボル（内部シンボルはSで，外部シンボルは：ではじまるシンボル）であれば，サブルーチン・エントリ擬似命令（：：：）をコード欄に格納する．そして，その次の対象処理番地に入っている機械語はアセンブラ語に変換しない．すなわち，アドレスと機械語だけを出力し，ステートメントが出力される所は空白のままにしておく．

(c) ラベル欄のシンボルが上記以外のシンボルであるか，またはラベル欄にシンボルがなければ命令語とみなす．

(iv) ニーモニック・コードの選択

あらかじめ，すべての機械語のオペレーション部とニーモック・コードとの対応表を作成しておき，ここでは，その対応表を検索することによって対象処理番地の機械語のニーモニック・コードを選択しコード欄に格納する．

(v) オペランド部の処理

ニーモニック・コード以外の残された部分（オペランド部とよぶ）を次の2つに分けて処理する．結果は変換されたアセンブラ語の出力領域の所定の場所にそれぞれ格納される．

(a) メモリ参照命令の場合は，まずアドレス方式の決定やレジスタの指定などアドレス修飾部の処理を行い，次に実効アドレスを作成しシンボルテーブルを参照してシンボルを決定する．

(b) メモリ参照命令以外の場合は，機械語の形式に従って，レジスタの指定，スキップ条件に対応する記号の選択，入出力機器番号の作成，またはコール命令の値の作成などを行う．

4．使用例

試作した逆アセンブラの使用例を図7に示す．処理番地は，16進表示で，3FE3から3FFF番地までであるが，図7では，アセンブラ・プログラム・リスト（出力）の一部分しか示していない．以下，図7での番号に従って簡単に説明する．なお，入出力はシステム・タイプライタのキーボード部およびプリンタ部を用いて行う．

(1) … 逆アセンブルしたいメモリ領域の処理先頭番地（16進表示で入力）．

(2) … 逆アセンブルしたいメモリ領域の処理最終番地（16進表示で入力）．

(3) … "SYMBOL～？" は外部シンボルの有無についての機械からの出力．外部シンボルを使用する場合は "Y" を，しない場合は "N" を入力．

(4) … "SYMBOL……" は外部シンボルの指定について機械からの出力．"：YOMI" は指定した外部シンボル（の入力）．

```
3FE3                                                    (1)
3FFF                                                    (2)
SYMBOL READ (Y OR N) ?Y                                 (3)
 SYMBOL    .... :YOMI                                   (4)
 ADDRESS   .... /3FE4                                   (5)
SYMBOL READ (Y OR N) ?N                                 (6)
 ***   INVERSE ASSEMBLE   ***

 CONVERSION FROM /3FE3 TO /3FFF                         (7)

     SYMBOL   ADDRESS
      C000    ....0001                                  (8)

     ASSEMBLER PROGRAM LIST

 /3FE3    810C              BRN   P000                  (9)
 /3FE4    3FF8    :YOMI;    :::                         (10)
 /3FE5    0001                                          (11)
 /3FE6    A800              LDI   R1,00                 (12)
 /3FE7    E12B    P002;     RDA   R0,/2B                (13)
 /3FF8    E02B    P001;     SNS0  /2B                   (14)
 /3FE9    81FE              BRN   P001                  (15)
 /3FEA    A21F              ANI   R0,/1F                (16)
 /3FEB    DC5A              RLCZ  R1,4,SCZ              (17)
 /3FEC    C341              ADR   R1,R0,SKP             (18)
 /3FED    C341              ADR   R1,R0,SKP             (19)
 /3FEE    99F5              BRN   :YOMI                 (20)
 /3FEF    81F7              BRN   P002                  (21)
```

Fig. 7 An example of the inverse assembler performance

(5) … 〝ADDRESS～／〞は指定した外部シンボルのアドレスについての機械からの出力．“3FE4”は，16進表示による“：YOMI”に対応するアドレス（の入力）．

(6) … (3)と同じである．Nの入力により，逆アセンブルの処理が開始される．

(7) … 処理番地の復唱

(8) … 処理番地外のシンボルとそれの示すアドレスとの対応表

(9) … (9)以下は，機械語からアセンブラ語に変換された，アセンブラ・プログラム・リストの一部．

アセンブラ・プログラム・リストの印字（出力）が完了すると機械の状態が入力状態に戻るように逆アセンブラは作成されているので，続けて処理番地を入力すれば，一連のプログラムと見なされて，連続したシンボル番号のもとで逆アセンブルが実行される．別々のプログラムとして逆アセンブルする場合は，一旦，機械を“ストップ”状態にし，再び実行開始すればよい．

5．あとがき

卒論で試作した逆アセンブラの特徴は，まえがきにも述べている通り，次の4つに集約される．

(1) 内部シンボルが記憶場所の性格をある程度表わしている．

(2) 使用者が，ラベルとして適当な外部シンボルを使用できる．

(3) 分割して記憶されている場合でも，一連のシンボルが利用できる．

(4) データ語の判定を行う．

この中で，(4)の項に関して，機械語のレベルで，データ語と命令語との区別を完全に行うことは，非常に困難である．ラベルがあれば，それによって区別は可能であるが，すべてのデータ語にラベルがついているとはかぎらないからである．ラベルなしの語は，すべて命令語とみなして変換処理を行っている．しかし，プログラミングの際，データ語はまとめて書くのが普通であるから，いくつかの語でもデータ語と判別できれば，それは解析の有効な手掛りとなると考えられる．また，特殊な場合として，ラベルだけでデータ語と判定した語が，実際は命令語であるということも起り得る．その意味で，完全なアセンブラ言語に変換されるのではないことを付記しておく．なお，逆アセンブラの作成において，プログラミングの分担や有意義な討論等で御協力頂いた本学卒業生染川聡一郎（リコー株式会社），吉田三千尋（武蔵株式会社）の両氏に謝意を表します．

おわりに，MELCOM－70ミニコンピュータシステムの導入に際し多大の御支援を頂いた本学電気工学科林田武一教授，富安隆一教授，又，日頃御指導御鞭撻いただく九州大学工学部吉田将教授に感謝いたします．

参考文献

1）森口繁一：〝システムあばき事始め－逆アセンブラについて－〞，ミニコン，P 82～87，共立出版（1974）

2）吉田博美：〝逆アセンブラの作成〞，長崎大学工学部卒業論文（昭和54年2月）

3）吉田三千尋：〝逆アセンブラの作成〞，長崎大学工学部卒業論文（昭和54年2月）

4）染川聡一郎：〝逆アセンブラの作成〞，長崎大学工学部卒業論文（昭和54年2月）

5）高倉聡：〝逆アセンブラの作成と応用〞，長崎大学工学部卒業論文（昭和55年2月）

6）MELCOM－70 システム説明書，資料番号JM－HS00－01A〈33B2〉，三菱電機（1973）

7）MELCOM－70 アセンブラ・システム説明書，資料番号JM－SR00－01A〈59B7〉，三菱電機（1973）

8）MELCOM－70 ベーシック・オペレーティングシステム説明書，資料番号JM－SR00－31A〈57A0〉，三菱電機（1975）

9）D. W. Barron: Assemblers and Loaders, American Elsevier Inc. (1969)