入出力マクロの設計

谷川邦夫*, 寺門純一*, 磯田和男*

(昭和53年9月8日受理)

The design of I/O Macro for FORTRAN-like Formatting.

KUNIO TANIKAWA, JUN-ICHI TERAKADO and KAZUO ISODA

Abstract: – Programming in assembly language is a time-consuming task requiring almost all attention to the machine-dependent details of each instruction, specifying precise and correct operations.
And, what is more, the details of input/output processing and data conversion are too complicate for beginners to write programs.
In this paper, two macro instructions for FORTRAN-like formatting were developed in order to alleviate this difficulties to some extent.

1. まえがき

本学情報工学科に設置されているHITAC 8350 EDOS アセンブリシステムには,カードリーダからの入力用としてRDCRD, ラインプリンタへの出力用としてPROUTといった2種類のマクロが備っているが, いずれも図1に示すような簡単な入出力動作しか行なわない。実際には, 外部機器とメモリ内部とではデータの表現方法が異っているため計算機でデータの処理を行なう場合には, 図2に示すようにデータの移動の段階でデータ変換や編集を考えなければならない。しかし, これらの手順をアセンブラ言語でプログラミングするということはなみたいていの事ではない。ましてアセンブラ言語の初

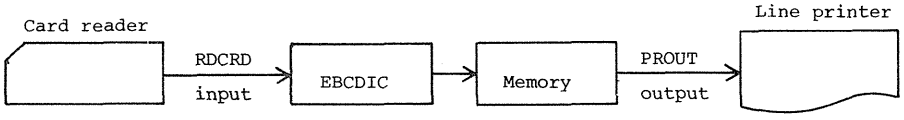

Fig. 1 Input/Output process.

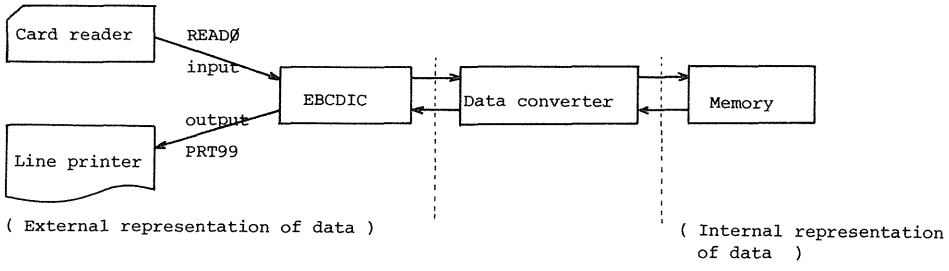

Fig. 2 Input/Output process.

* 茨城大学工学部情報工学科(日立市中成沢町)

心者にとっては，命令を並べるだけでかなりの労力と時間を要するだろう。

ここで設計された入出力マクロ（図3）は，FORTRAN風の書式制御文（フォーマット文）をDC（Define Constant）命令（図4）で与えることにより，データの変換，編集を簡単に行なえるようにしたものである。

2. 構成および機能

この入出力マクロおよびフォーマット文は，EDOSアセンブリシステムの下ではたらくように設計されている。したがって図3に示すように，他のエグゼクティブ連絡マクロと同じ形式をとっている。

Label	Opcode	Operand
(Symbol or Blank)	READØ	Label,Datal,Data2,.....
	PRT99	Label,Datal,Data2,.....

Fig. 3 Macro format.

Label	Opcode	Operand
Symbol	DC	'(S1,S2,...,Si/Sj,...,Sn)'

Sl,.... are format codes.

Fig. 4 The format statement.

オペレーション・コードはREADØ（カードリーダから入力），またPRT99（ラインプリンタへの出力）であり，オペランドにはフォーマット文のラベル，入出力並びをラベルで順に記入する。入出力並びはなくてもさしつかえないが，入出力マクロとフォーマット文は対になっているため，フォーマット文のラベルを省略することはできない。このマクロ，フォーマット文を使ってプログラムを実行させると次の4つのサブプログラムがユーザプログラムの後にリンクされる。

a）フォーマットコード・スキャン・サブルーチン

b）フォーマットコード処理サブルーチン

c）データ変換サブルーチン

d）入出力処理サブルーチン

次にフォーマット文による入出力についてのべる。図6に示すように入出力並びがData1, Data2……DataN，書式がDC命令によって'（S1，S2，……，Sm）'

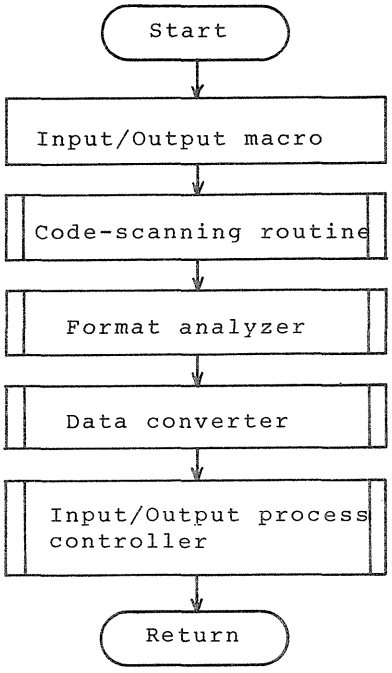

Fig. 5 Flow of subprogram.

```
                .
                .
                .
        EADØ     FMTl,DATAl,DATA2,.....,DATAn
                .
                .
                .
FMTl    DC       '(Sl,S2,S3,.........,Sm)'
                .
                .
DATAl   DS        F
DATA2   DS        F
                .
                .
                .
DATAn   DS        F
                .
                .
                .
                .
```

Fig. 6 Sample.

と与えられている場合に入出力を実行しようとすると，まずData1のアドレス計算を行ない，書式を左から見ていって，入出力並びを要求しない欄記述子を順に実行し次に実際の入出力を行なう。さらにData2のアドレス計算を行ない，前述の操作を繰り返す。したがって書式と入出力並びとの間では，実行の制御が行き来することになる。（図7）

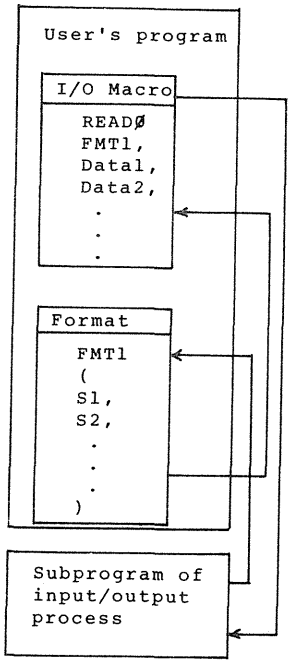

Fig. 7　Flow of control.

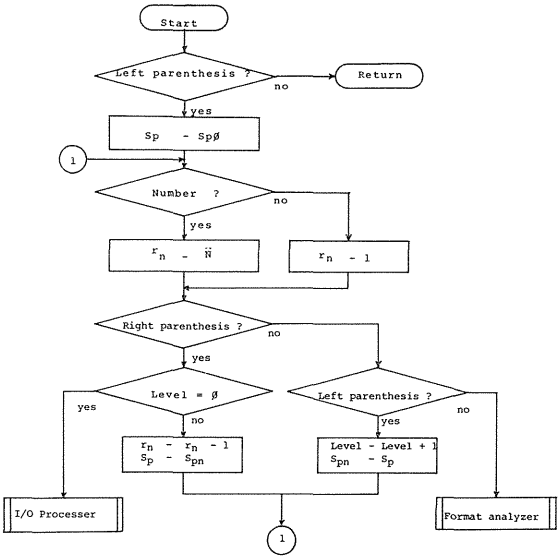

Fig. 8　Flow-chart for routine of code-scanning.

2－1　フォーマットコード・スキャン・サブルーチン

ユーザがフォーマット文を定義し，READØおよびPRT99を実行しようとしたときに使用されるルーチンでデータの入出力時にそのフォーマットコードのスキャン（走査）を行なう。

まずフォーマット文の最初の1文字が'('であることを調べ，レベルをØとおき次の1文字の走査点をSpとし走査を開始する。走査の際','は無視し，レベルごとの走査開始点Spnをもうける。したがって走査はSp＝SpØ（レベルØの走査開始点）という状態から始まることになる。図8に走査の流れを示す。

1）数字で始まる場合，この値をγn（反復回数）とし数字以外で始まる場合はγn＝1とおく。

2）')'なら3）の処理を，')'なら4）の処理を行なう。それ以外の場合は欄記述子とみなして，フォーマットコード処理サブルーチンへコントロールを移す。

3）レベルを調べ，Øなら入出力処理サブルーチンへコントロールを移す。Ø以外の場合には，γn>Øなら，そのときのレベルのSpnをSpに移し1）の処理へもどる。

4）レベルに1を加え，レベル≦2ならSpをそのときのレベルのSpnに移し1）の処理へもどる。

このフォーマット文は，欄記述子群の重なり（レベル2まで）を認めているため，重なりにレベルをつけレベルごとにSpn，γnという変数が必要になっている。また走査がフォーマット文の最後の')'（レベルØ）に達したとき，欄記述子群の重なりがある場合はレベル1の'('までもどり，Sp_1から走査を続ける。ない場合はレベルØのSpØから走査することになっている。図9に重なりについて示す。

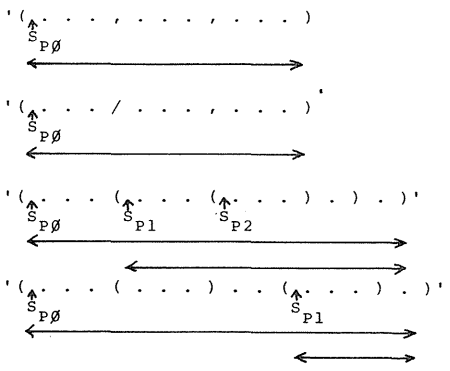

Fig. 9　The nesting of group format specifications.

2－2　フォーマットコード処理サブルーチン

2－1のルーチンでスキャンにされたコードを解析しコードが正しければ，フォーマットコードごとに用意されている各データ変換サブルーチン（入力用と出力用がある。）へコントロールを移す。（図10）

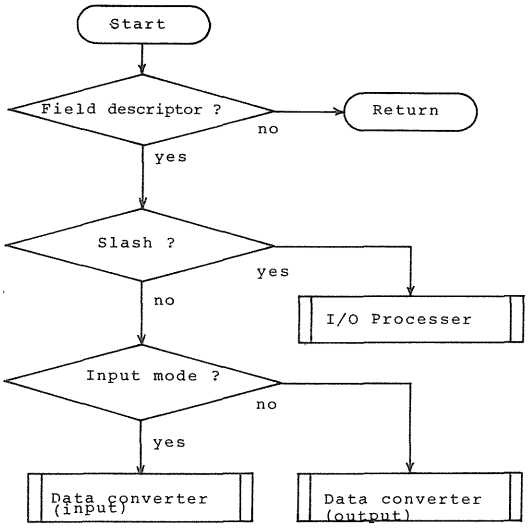

Fig. 10　Flow-chart for routine of format analyzer.

まずこのルーチンでは欄記述子かどうかのチェックを行なっている。欄記述子はωS，γSω，γSω・dいづれかの形式をとっていなければならない。欄記述子には次の3種類がありそれぞれの形に応じて入力用と出力用のデータ変換サブルーチンが用意されている。

- ○文字欄記述子　　γSω，　ωS
- ○数値欄記述子　　γSω，　γSω・d
- ○空白欄記述子　　ωS

空白欄記述子は入力並びは必要としない。またコードが'／'の場合にはただちに入出力処理サブルーチンへコントロールを移す。

2－3　データ変換サブルーチン

入出力するデータの形式を管理するもので，入出力されるデータは必らず本ルーチンにより指定の形式に変換される。各フォーマットコード（欄記述子）ごとに入力用と出力用の2種類のサブルーチンがあり，出力用サブルーチンでは編集も自動的に行なわれる。

2－4　入出力処理サブルーチン

このルーチンはデータの入出力を行なう。入力の場合カードリーダーからメモリに80文字読み込むだけの仕事のみ行ない，データのチェックは2－（1～3）のルチンにまかせている。出力の場合は，ラインプリンタにメモリから133文字（最初の1文字は制御文字）出力する。制御文字のチェックは必らず行ない，違っている場合は最初にブランクを補い出力動作に入る。

2－5　欄記述子

欄記述子の種類に応じて，各データ変換サブルーチンへコントロールが移される。

(1)　文字欄記述子

英数字，文字または記号を取り扱うために3つの型がありγCω，γZω，ωHの形式をとっている。16進データも文字欄記述子群に入れてある。

- γCω　：　英数字，文字または記号の入出力用
- γZω　：　16進データ（0～9，A～F）の入出力用
- ωH　：　フォマットコードωHに続けてω個の英数字，文字または記号を書く。出力用として使用した場合，ωHに続くω文字がプリントアウトされ入力用として使用した場合はω文字がカードリーダより読み込まれ，ωHに続くω文字が読み込まれたω文字と置き換る。

(2)　数値欄記述子

数値として取り扱われるデータに対する変換形式としては，次に示す4つの型がある。

型	形　式	内　容	外　　部
I	γIω	固定小数点（4バイト）	整数型表示
F	γFω・d	浮動小数点（4バイト）	実数型表示
E	γEω・d	浮動小数点（4バイト）	実数型表示で10のベキ数をもつ
D	γDω・d	浮動小数点（8バイト）	実数型表示で10のベキ数をもつ

ここで，ωは全体のフィールドの長さ（符号，小数点を含む）を示す符号なしの正の整数型定数。dは小数点以下の桁を示す整数型定数，γは欄記述子の繰り返される回数を示す。

a）入力用として使用した場合

整数型データを入力する場合にはI型を使用し，実数型データの入力にはF，E，D型を使用する。数値の頭

のブランクは無視され，それ以外のブランクはøとみなす。実数型データの場合は，データの小数点の位置がdより優先され，指数部はデータの右の4文字とし，指数部の始まりは十，一，E，Dのいずれかである。

b）出力用として使用した場合

整数型データの出力にはI型を使用し，ωは全体のフィールドの長さを示す。ωが実際のデータの長さより小さいときには，変換されたデータの右端から（ω－1）桁で切られ，左端に'＊'がつけられる。ωがデータの長さより大きいときは，数値の前にデータの長さをこえた分だけブランクが補なわれる。実数型データの出力にはF，E，D型を使用し，符号，小数点，指数部のエリアの確保が必要なため，F型の場合はω－d≧3，E，D型の場合はω－d≧7の条件を満足しなければならない。

数値の整数部分はI型と同じ処理が行なわれるが，小数点以下のøは無視されない。またデータのまるめは行なわない。

(3) 空白欄記述子

ωXの形式をとり，入力の場合はω文字読みとばし，出力の場合はω個のブランクをプリントする。

(4) 制御文字（改行制御）

ラインプリンタの紙送りを制御するための情報で，出力データの最初の文字で指定する。なお制御文字はプリントされない。もし指定がない（最初の文字が制御文字ではない）場合は制御文字としてブランクが出力データの前に補なわれる。

制御文字	動　　作
ブランク	1行送ってからプリント
ø	2行送ってからプリント
＋	行送りしないでプリント
1	ページ送りをしてからプリント

3. 使用上の注意と使用例

この入出力マクロを使用する場合には，次のことに注意しなければならない。

1）外部表現としてEBCDICコードを使用し，内部表現としてはバイナリコードを扱っている。したがって数値の内部表現は固定小数点表示と浮動小数点表示の2種類だけである。

2）入出力マクロで使われている記号（##FMT##）をソースプログラムで使用できない。また他のエグゼクティブ連絡マクロと同じように汎用レジスタ14および15を使用しているので，ユーザがこのレジスタを使用するときは注意が必要である。

3）データの並びは10個までにしてある。ただしマクロを展開した形で使用する場合は，並びの制限はなく

```
FLAGS   LOCTN OBJECT CODE       ADDR1 ADDR2 STMNT M  SOURCE STATEMENT
        00000                                 00001 TERA   START
                                              00002        PRINT NOGEN
        00000 05 20                           00003        BALR  2,0
        00002                                 00004        USING *,0
                                              00005        READ0 FMT1,A,B
        0001C 58 30 204A        0004C         00006        L     3,A
        00020 5A 30 204E        00050         00007        A     3,B
        00024 50 30 2052        00054         00008        ST    3,C
                                              00009        PRT99 FMT2,A,B,C
                                              00010        TERM
        0004C                                 00011 A      DS    F
        00050                                 00012 B      DS    F
        00054                                 00013 C      DS    F
        00058 4DC9F36BC9F55D                  00014 FMT1   DC    '(I3,I5)'
        0005F 4DF1C8406BC9F36B                00015 FMT2   DC    '(1H ,I3,2I5)'
                                              00016        END


// EXEC

  3   15   18


// END
```

Fig. 11 Sample program.

なる。

図11に入出力マクロとフォーマット文の使用例を示す。まずREADØによりカードリーダからデータA（＝3），B（＝15）を読み込み，次にC＝A＋Bの演算を実行し，その結果をPRT99によりラインプリンタへA，BおよびC（＝18）の値を出力している。

4. あとがき

とかくアセンブラ言語は，敬遠されがちである。それというのもFORTRAN等の高級言語で書かれたプログラムではステップ数が少なくてすむ計算のアルゴリズムでも，アセンブラ言語によるプログラミングでは，数多くの命令とステップ数が必要となるためではないだろうか。さらに入出力となると計算機内部と外部機器との関係が複雑で，プログラムを作るということはなみたいていではない。

今回我々が作成したFORTRAN風のマクロはデータの入出力に関してデータの変換，編集を自動的に行なう能力を持っている。したがってこのマクロを活用することにより，アセンブラ言語でのプログラミングの手間をはぶき，初心者でも数少い命令を覚えるだけでデータの入出力動作を含めたプログラムの作成が容易になるだろう。

参考文献

(1) 一：EDOS FCPとエグゼクティブ連絡マクロ
8300-3-108 ：㈱日立製作所

(2) 一：EDOS アセンブリ・システム
8300-3-101 ：㈱日立製作所

(3) 一：H-8350 処理装置
8350-2-001 ：㈱日立製作所

(4) 島内剛一他 ：FORTRANの実際
：サイエンスライブラリ

(5) 一：IBM system／360 and system／370 FORTRAN IV Language ：IBM corporation

(6) waiter G. RUDD ：Assembly Language programming and the IBM 360 and 370 computer ：prentice-Hall