

JAERI-M
7337

BSAMレベルでの特殊I/Oルーチンの開発

1977年10月

岡本 正雄・滝塚 知典・和田 善之・岡田 高光*

日本原子力研究所
Japan Atomic Energy Research Institute

この報告書は、日本原子力研究所が JAERI-M レポートとして、不定期に刊行している研究報告書です。入手、複製などのお問合せは、日本原子力研究所技術情報部（茨城県那珂郡東海村）あて、お申しこしください。

JAERI-M reports, issued irregularly, describe the results of research works carried out in JAERI. Inquiries about the availability of reports and their reproduction should be addressed to Division of Technical Information, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken, Japan.

B S A M レベルでの特殊 I/O ルーチンの開発

日本原子力研究所東海研究所核融合研究部
岡本正雄・滝塚知典・和田善之^{*}・岡田高光^{*}

(1977年9月22日受理)

B SAM レベルでの特殊 I/O ルーチン「FORTXDAM」を開発・作成した。「FORTXDAM」は、大量のデータを頻繁に READ/WRITE する時、データのランダム・アクセス（もしくは、ダイレクト・アクセス）を行う時、ファイル効率を高めるために、ダブル・バッファリング（一般にはマッチ・バッファリング）の技法を用いる時、データの転送中に CPU を使って計算を行う時（非同期入出力）、などの場合に極めて有効なものである。「FORTXDAM」は原研計算センタの FACOM 230-75 の計算機システムで有効な FASP 言語で書かれており、6つの基本副プログラムで構成されていて、FORTRAN の CALL 形式で呼び出すことができる。このプログラムは、核融合研究・プラズマ物理における大規模な計算機シミュレーションにおいて特に役立つものである。

* 富士通株式会社原子力システム開発部

Special I/O Routine Based on the BSAM Level

Masao OKAMOTO, Tomonori TAKIZUKA,
Yoshiyuki WADA * and Takamitsu OKADA *

Division of Thermonuclear Fusion Research,
Tokai Research Establishment, JAERI

(Received September 22, 1977)

A Special I/O routine "FORTXDAM" has been developed, which is based on the BSAM Level. Program "FORTXDAM" is useful in input-output of large quantities of data, random access (direct access), usage of the double (or multi) buffers technique or asynchronous input-output. Written in FASP language available for the computer system FACOM 230-75 of JAERI, it consists of six basic sub-programs which can be called in FORTRAN language. FORTXDAM is useful especially in large-scale computer simulations in thermonuclear fusion and plasma physics research.

Keywords : I/O Routine, BSAM Level, Simulation, FORTXDAM Code,
FASP language

* Nuclear Power Systems Development Department, Fujitsu Limited.

目 次

1. 目的と概要	1
2. 基本サブ・ルーチンの機能と使用法	2
2.1 XDPEN	2
2.2 XDFORM	4
2.3 XDWRIT	5
2.4 XDREAD	6
2.5 XDCHEK	6
2.6 XDCLOS	7
2.7 エラー処理	7
3. 例題とテスト・ランの結果	18
4. まとめ	20
謝 辞	20
参考文献	20
付録A. 「FORTXDAM」のテスト・プログラム	22
付録B. 「FORTXDAM」のプログラム・リスト	24

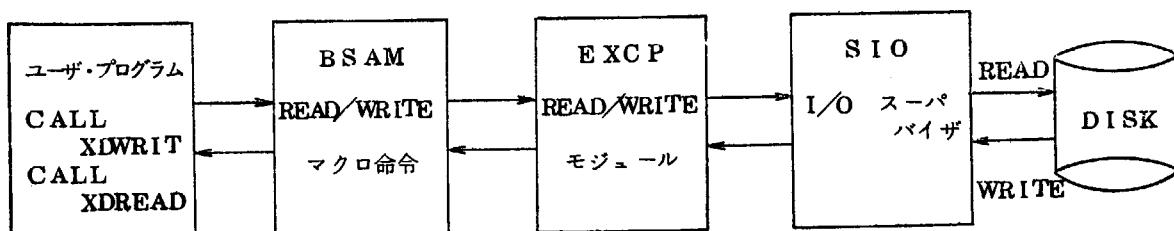
1. 目的と概要

核融合研究や、プラズマ物理の分野では、大規模な計算機シミュレーションがしばしば行なわれる。このようなシミュレーションでは、大部分が初期値問題を扱っており、大量のデータを頻繁に入出力することが多い。FORTRAN レベルで大量のデータを頻繁に入出力するとcpu時間が著しく増加するばかりでなく、ジョブの実効的な時間(through-put timeあるいはelapsed time, またはcore time)も著しく増加する。従って、大規模なシミュレーションを効率よく実行するためには、入出力に必要なcpu時間やジョブの実効的な時間を軽減する試みは、重要であると考えられる。

我々は、このような目的のために、プログラム「FORTXDAM」(FORTRAN EXTENDED DIRECT ACCESS METHOD)を開発した。このプログラムはFORTRAN-H[1]よりもさらに強力なdisk access methodを提供し、ユーザはこれを使用することにより、ジョブのcpu時間やジョブの実効的な時間を大巾に短縮することができる。すなわち、「FORTXDAM」はFACOM 230-75 M-VII FORTRAN-H言語で作成しようとするアプリケーション・プログラムが次のような方法でディスク・アクセスを行うときにcpu時間、ジョブの実効時間を大巾に軽減することを目的としている。

- 1) 大量のデータを頻繁に(もしくはループ的に)READ/WRITEするとき。
- 2) データのランダム・アクセス(ダイレクト・アクセス)を行うとき。
- 3) ファイルの効率を高めるために、ダブル・バッファリング、あるいは一般的にマルチ・バッファリング技法を用いるとき。
- 4) データの転送中にcpuを使って計算を行うとき(非同期入出力)。

「FORTXDAM」は、F230-75 M-VII データ管理BSAM(Basic Sequential Access Method)[2]を使用して作成され、FASP言語[3]で書かれている。すなわち、READ/WRITEマクロ命令で記述してある。EXCP(Execute Channel Program)とI/Oスーパーバイザ(モニタ)との関係は第1図の通りである。



第1図 I/O の処理

「FORTXDAM」はFASP言語で書かれており、6つのエントリー(XDOPEN, XDFORM, XDWRIT, XDREAD, XDCHEK, XDCLOS)を持っている。それぞれのエントリーはM-VII FORTRAN-HでCALL形式にて呼び出すことができる。CP, CNPどちらのモード[4]でも可能である。XDOPENは与えられた論理機番のファイルをオープンし、XDFORMはディスクのフォーマット・ライトを行う。XDWRITはバッファの中のデータをディスクに転送し、XDREADはその逆を行う。XDCHEKは、XDWRIT, XDREADのルーチンが呼ばれた後に同期をとり、I/Oの動作の完了状態をチェックする。XDCLOSはファイルのクローズ処理を行う。基本的な思想として、XDOPENはBSAMのOPENマクロ命令に対応し、XDWRITは、BSAMのWRITEマクロ命令と、XDREADは、READマクロ命令と対応するようにプログラムは作成されている。XDFORMは、大記憶装置(ディスク)のCKD部(カウント部、キ一部、データ部)の創造を行うものであり、WRITE, CHECKマクロ命令を使って作成されている。

2章では、6つの基本サブ・ルーチンの機能と使用法について述べる。これら6つの基本サブ・ルーチンを組合せることによって、BSAMレベルでのI/Oを実行することができる。

3章では、簡単な例題とそのテスト・ランの結果を示す。4章でまとめを行い、付録Aにテスト・プログラムを、付録Bに「FORTXDAM」のプログラム・リストを掲げる。

2. 基本サブ・ルーチンの機能と使用法

基本サブ・ルーチン「FORTXDAM」はFASP言語で作成されており、6つのサブ・ルーチン(XDOPEN, XDFORM, XDWRIT, XDREAD, XDCHEK, XDCLOS)から成る。こゝでは、これら6つのサブ・ルーチンの機能とその使用法について述べる。

2.1 XDOPEN

呼び出し形式

```
CALL XDOPEN(INOUT, LENGTH, FDNAME)
```

機能

- (1) 与えられた論理機番(INOUT)のファイルをオープンする。
 - ・ユーザの使用するファイルはすべてこのサブ・ルーチンを呼んでオープンしなければならない。
 - ・論理機番は1~98の数で与える。オールド・ファイル(すでにデータが書かれているファイル)の場合はINOUT=99と指定する。
- (2) 与えられた機番のテーブルを作成し、そのファイルに関する情報を格納する。以下

XDREAD, XDWRIT 等が呼ばれたとき, INOUT でどのファイルかを判定し, その情報を使用する。

テーブルの種類

LOGICNO : 論理機番。
 FCB : fcb (file control block) の先頭番地。
 NOTRACK : LENGTH/13023 の商。
 MINBLKSZ : NOTRACK ≠ 0 のときの LENGTH/13023。
 MAXBLKSZ : ブロック・サイズ。
 NOTRACK ≠ 0 のとき = 13023。
 NOTRACK = 0 のとき = LENGTH。
 NO1BLOCK : NOTRACK = 0 のとき
 $\left[\frac{13030 - LENGTH}{LENGTH + 135} \right] + 1.$

引数の説明

INOUT ファイルの論理機番。= 99 ならばオールド・ファイルとして処理する。
 LENGTH バッファ・サイズ(バイト単価)。
 FDNAME ファイル定義名(英文字で始まる 8 文字の英数字。必ず 8 文字必要)。

制限事項

(1) ファイル数

オープンできる(使用できる)ファイル数は最大 16 個である。

(2) バッファ・サイズ

$$LENGTH \leq 13,023 \times 16 = 208,368 \text{ バイト}$$

$$= 46,304 \text{ 語}$$

(1 語 = 4.5 バイト)。

(3) ファイル定義名

前づめで 8 文字なければならない。

(例) 8HF01□□□□

'TPDISK1 □'

注意事項

(1) LENGTH

・ワード単位のものをいったんバイト単位に直して XDPEN を CALL すること。転

送モードを TMOD=8 としているので、1ワードを 4.5 バイトとして計算する。

- ・バッファ・サイズは偶数ワードでなければならない。
- ・バッファの先頭番地は偶数番地でなければならない。（COMMONは偶数番地にとられるのでこれを利用する。）

```
(例)      COMMON/BUFFER/BUFF(100)
           ↓   ↓
           偶数番地 偶数
LENGTH=100*9/2  バイト単位に変換
CALL XDPEN(1, LENGTH, 'FO1HBBBBB')
```

(2) 13023 の意味

ディスク・パックの1トラックのバイト数が 13030 である。その範囲で最大の偶数ワードをバイトに変換すると 13023 になる。

2.2 XDFORM

呼び出し形式

```
CALL XDFORM(INOUT, MAXNB, BUFF)
```

機能

- (1) ディスクのフォーマット・ライトを行う。フォーマット・ライト (format write) とは、ファイルを創造するために媒体上に CKD 部（カウント部、キー部、データ部）を創造することである。従ってファイルは READ/WRITE する前に必ずこのルーチンでフォーマット・ライトしなければならない。
- (2) フォーマット・ライトした後、FORTXDM は内部でいったんそのファイルを閉じ、その後、ファイルの形式を update モードに変更して再びオープンする。以後そのファイルは READ/WRITE は自在にできる。

引数の説明

INOUT 論理機番。

MAXNB データ数。
データ数とは最大何回 WRITE するかの数。

BUFF READ/WRITE 時のバッファ用の変数。これは偶数番地から始まっているなければならない。

制限事項

(1) データ数

データ数が多い程ディスクを多量に使うから、FD文で適當な量を確保しなければならない。

• LENGTH ≥ 13023 のとき

$$\left(\frac{\text{LENGTH}}{13023} + 1 \right) * \text{MAXNB} \quad \text{トラック}$$

• LENGTH < 13023 のとき

$$\frac{\text{MAXNB}}{\left[\frac{13030 - \text{LENGTH}}{\text{LENGTH} + 135} \right] + 1} \quad \text{トラック}$$

2.3 XDWRIT

呼び出し形式

```
CALL XDWRIT(INOUT, NBUF, BUFFi)
```

機能

- (1) ファイルのデータ番号(NBUF)の位置にバッファの内容を書き出す。従ってこのルーチンを呼び出すときの論理機番のファイルは、すでにXDOPEN及びXDFORMでオープン、フォーマット・ライトされていなければならない。
- (2) データ番号によりTTR(ディスク上の相対的な番地)を計算してその位置から書き出す。

TT : ファイル上の相対トラック番地。0から始まる。

R : トラック上の相対ブロック番号。1から始まる。

TTRの計算式

• NOTRACK = 0 のとき

$$\begin{cases} TT = (NBUF - 1) / NO1BLOCK \\ R = NBUF - NO1BLOCK * TT \end{cases}$$

• NOTRACK ≠ 0 のとき

 i) MINBLKSZ = 0 のとき

$$\begin{cases} TT = NOTRACK * (NBUF - 1) \\ R = 1 \end{cases}$$

ii) MINBLKSZ ≠ 0 のとき

$$\begin{cases} TT = (\text{NOTRACK} + 1) * (\text{NBUF} - 1) \\ R = 1 \end{cases}$$

(3) このルーチンを呼び出した後に必ず XDCHEK で同期をとらねばならない。

引数の説明

INOUT : 論理機番。

NBUF : データ番号。

BUFF_i : i 番目のドッファの先頭番地（偶数番地）を持つ変数（例， BUFF(1, 2)）。

制限事項

(1) データ番号

データ番号は XDFORM でのデータ数 MAXNB の値を越えてはならない。

2.4 XDREAD

呼び出し形式

```
CALL XDREAD( INOUT, NBUF, BUFFi )
```

機能

(1) ファイルのデータ番号の位置からバッファに読み込む。

このルーチンは XDWRIT の WRITE マクロが READ マクロに変わっただけで他はすべて同じである。

引数の説明

INOUT : 論理機番。

NBUF : データ番号。

BUFF_i : i 番地のバッファの先頭番地（偶数番地）を持つ変数。

2.5 XDCHEK

呼び出し形式

```
CALL XDCHEK( INOUT )
```

機能

- (1) XDREAD, XDWRIT のルーチンが呼ばれた後に同期をとり, I/O の動作の完了状態をチェックする。
- XDREAD, XDWRIT のルーチンを呼んだ後には, 必ずこのルーチンを呼ばなければならない。

引数の説明

INOUT : 論理機番。

2.6 XDCLOS呼び出し形式

CALL XDCLOS (INOUT)

機能

- (1) 与えられた論理機番 (INOUT) のファイルをクローズする。
 クローズ処理をした後, このファイルは LENGTH, MAXNB 等を変更して再びオープンすることができる。オープンされていないファイルに関しては何の処理も行なわない (no-operation)。

引数の説明

INOUT : 論理機番。

2.7 エラー処理

「FORTXDAM」でエラーが発生した場合, 次のようなメッセージを出力して処理を打切る。

***** FORTXDAM ERROR MESSAGE *****

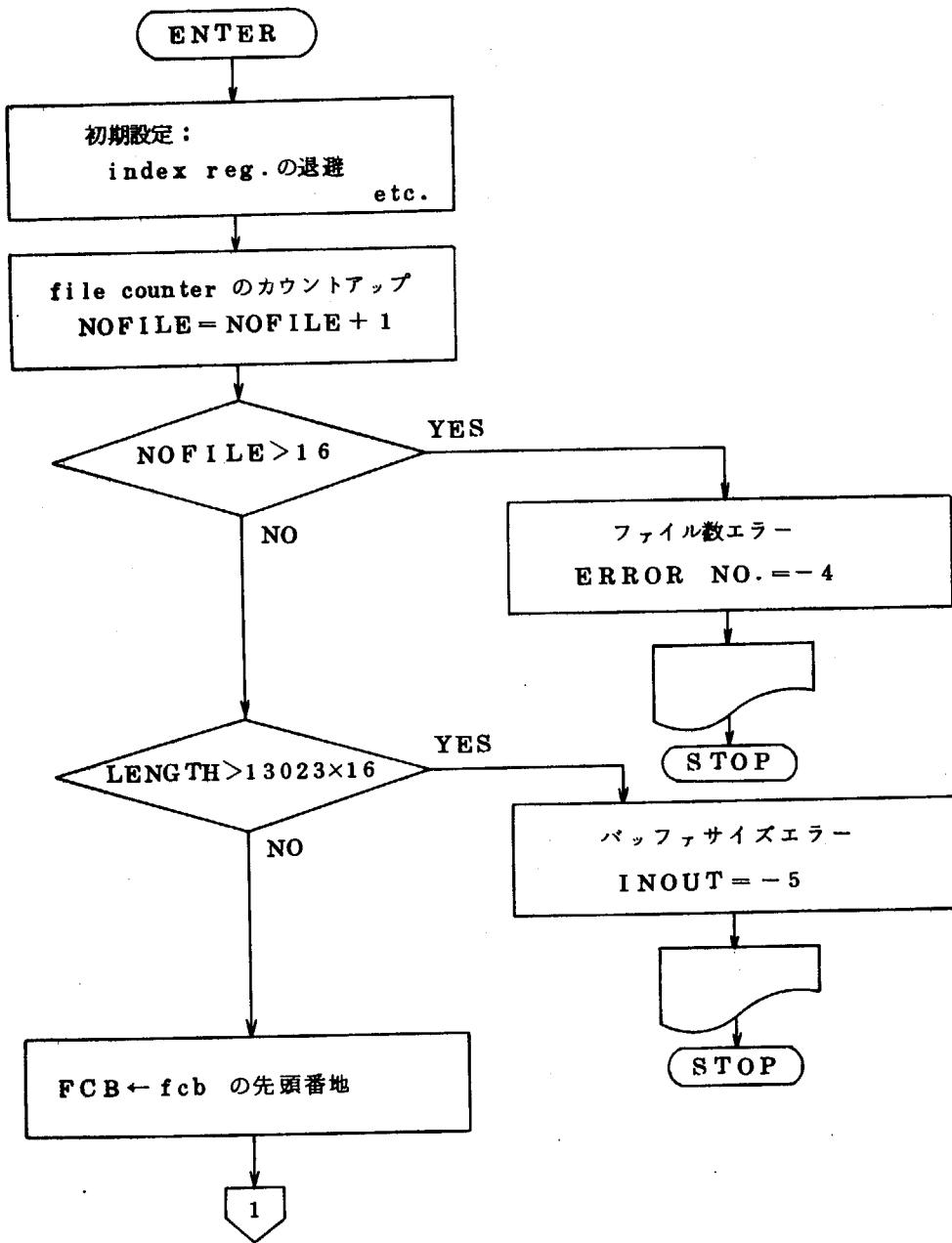
XDOPEN^① ISN = 2^② FILE = 1^③ ERROR NO. -1^④
 STATUS 400000000000^⑤

EXECUTION TERMINATED

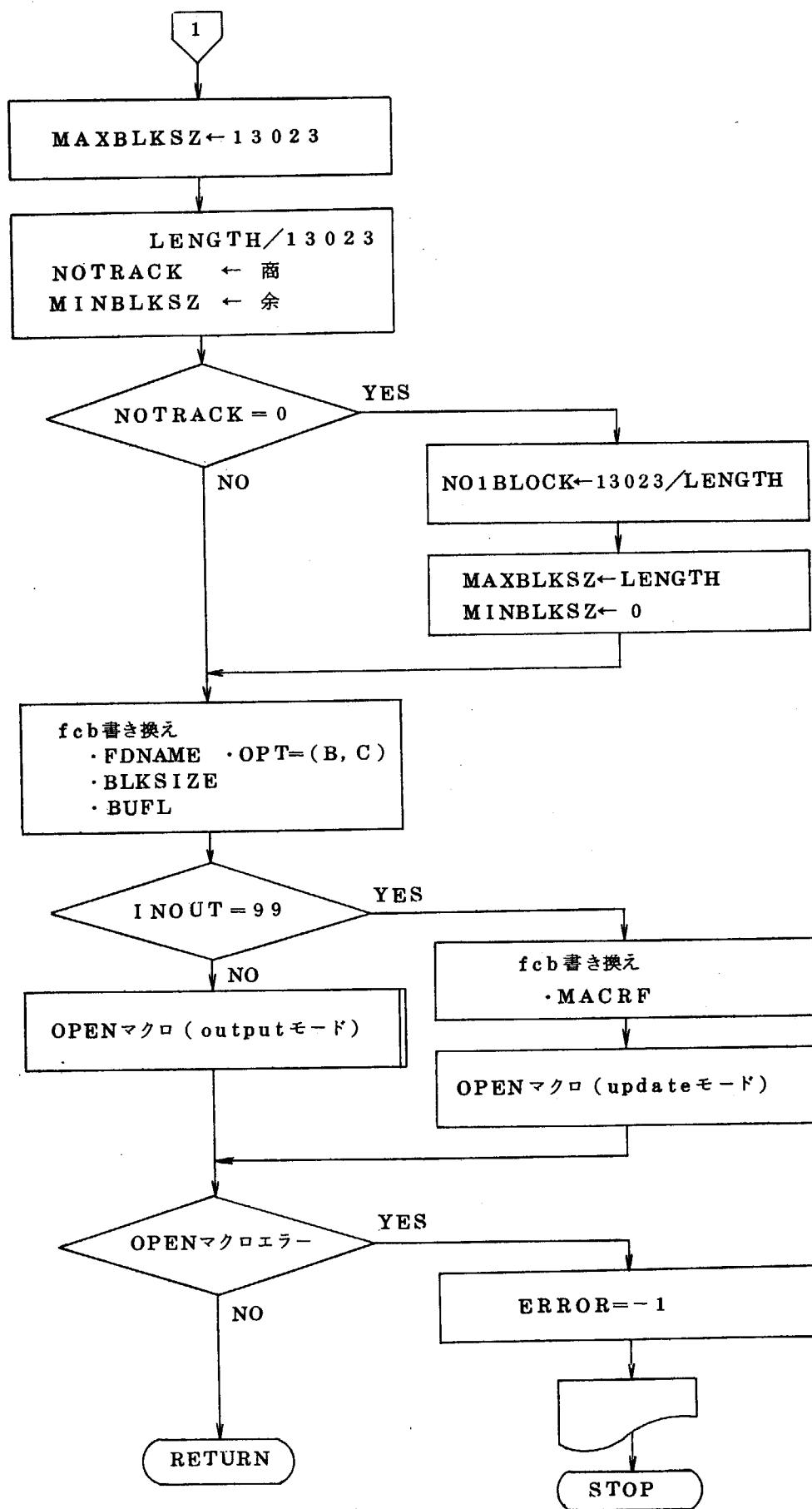
- ① エラーが発生したルーチン名
- ② ①をCALLしているプログラムの内部文番号。
- ③ エラーが発生したファイルの論理機番。
- ④ エラー番号

-1 : オープン・マクロ・エラー (XDOPEN, XDFORM)。
-2 : チェック・マクロ・エラー (XDFORM, XDCHEK)。
-3 : クローズ・マクロ・エラー (XDFORM, XDCLOS)。
-4 : ファイル数が17個以上 (XDOPEN)。
-5 : バッファ・サイズ・オーバー (XDOPEN)。
-6 : ファイルがオープンされていない (XDFORM, XDWRIT, XDREAD,
 XDCHEK)。

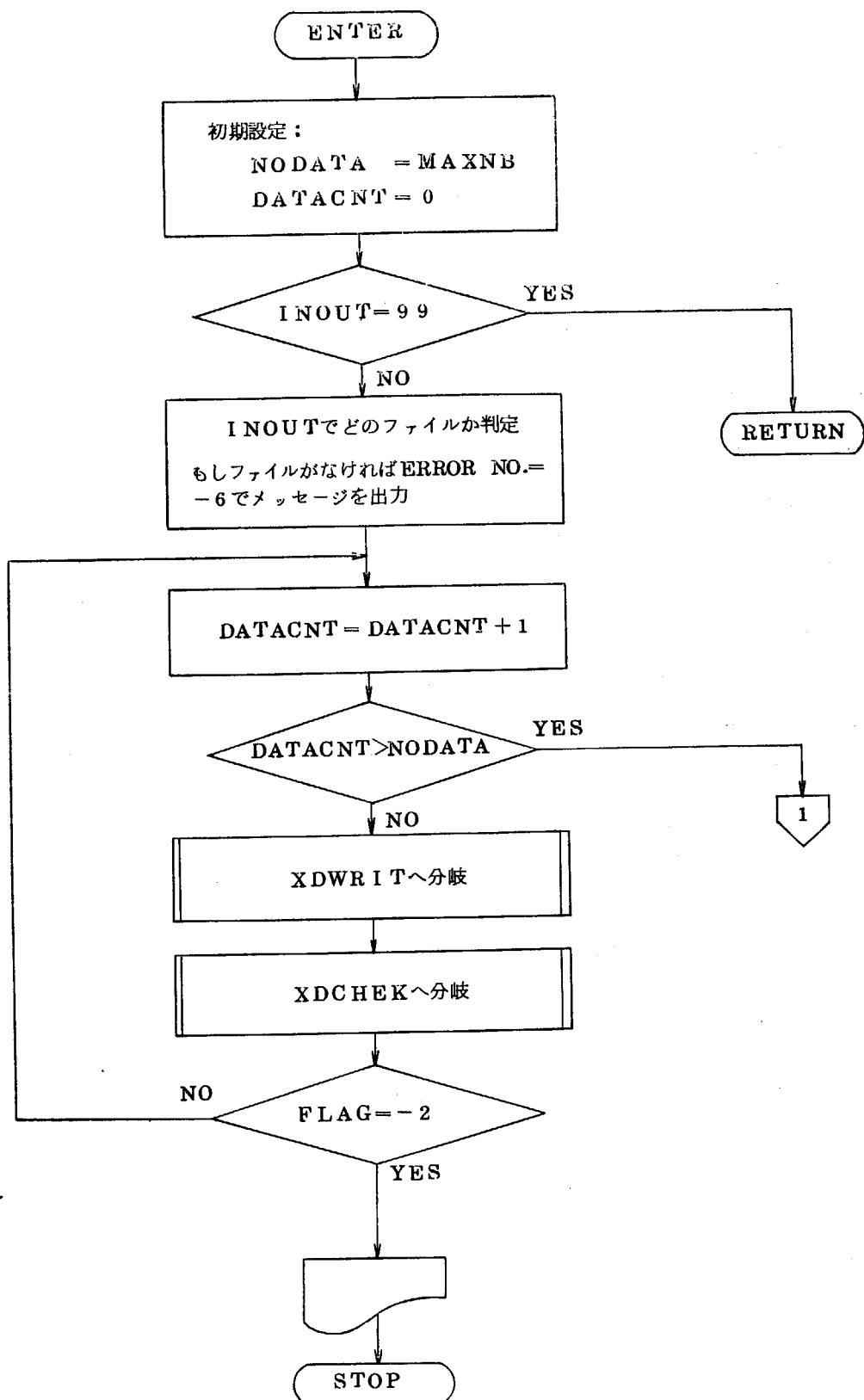
- ⑤ マクロ・エラー (-1, -2, -3)のときの通知情報 (8進数)。内容に関しては文献
[2]のOPEN, CLOSEマクロ, または, BSAMのREAD, WRITEマクロ命令の
通知情報を参照のこと。



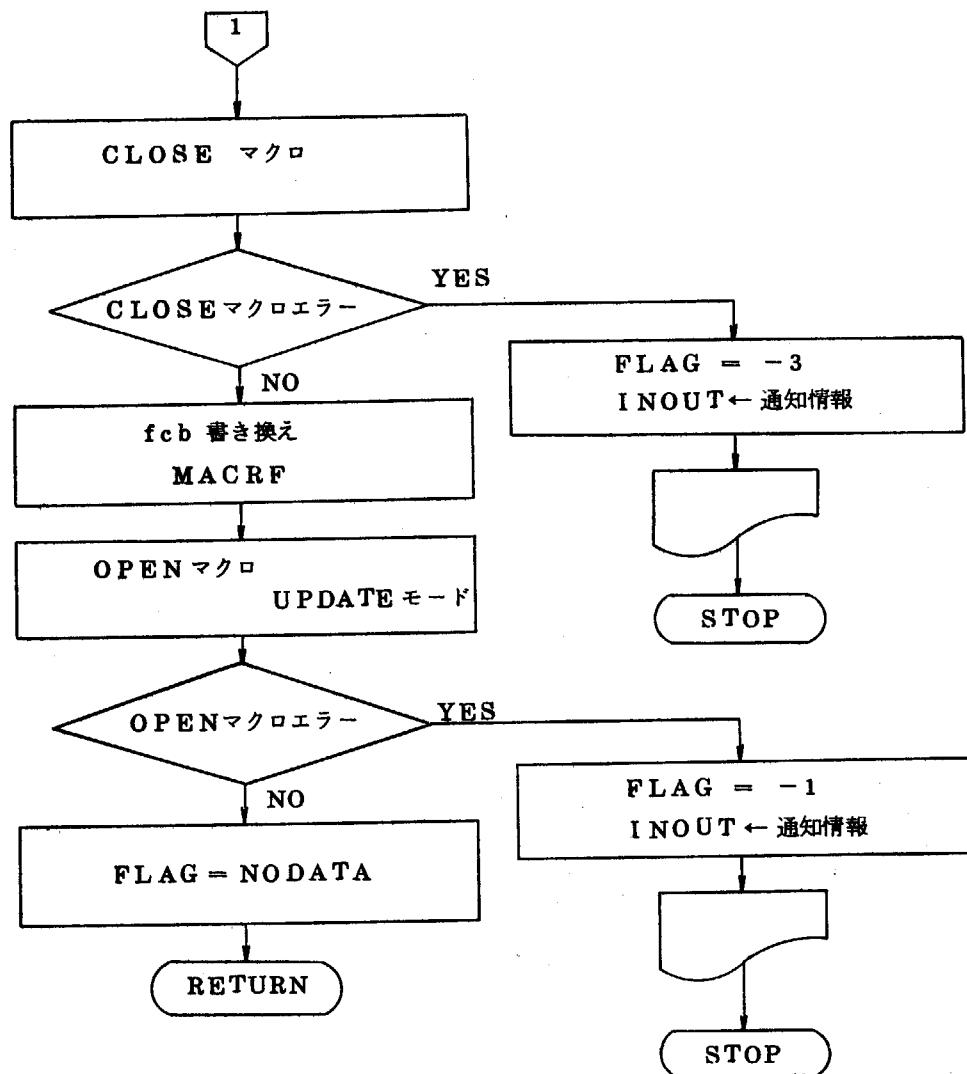
第2図 XDPEN のプログラム・フロー



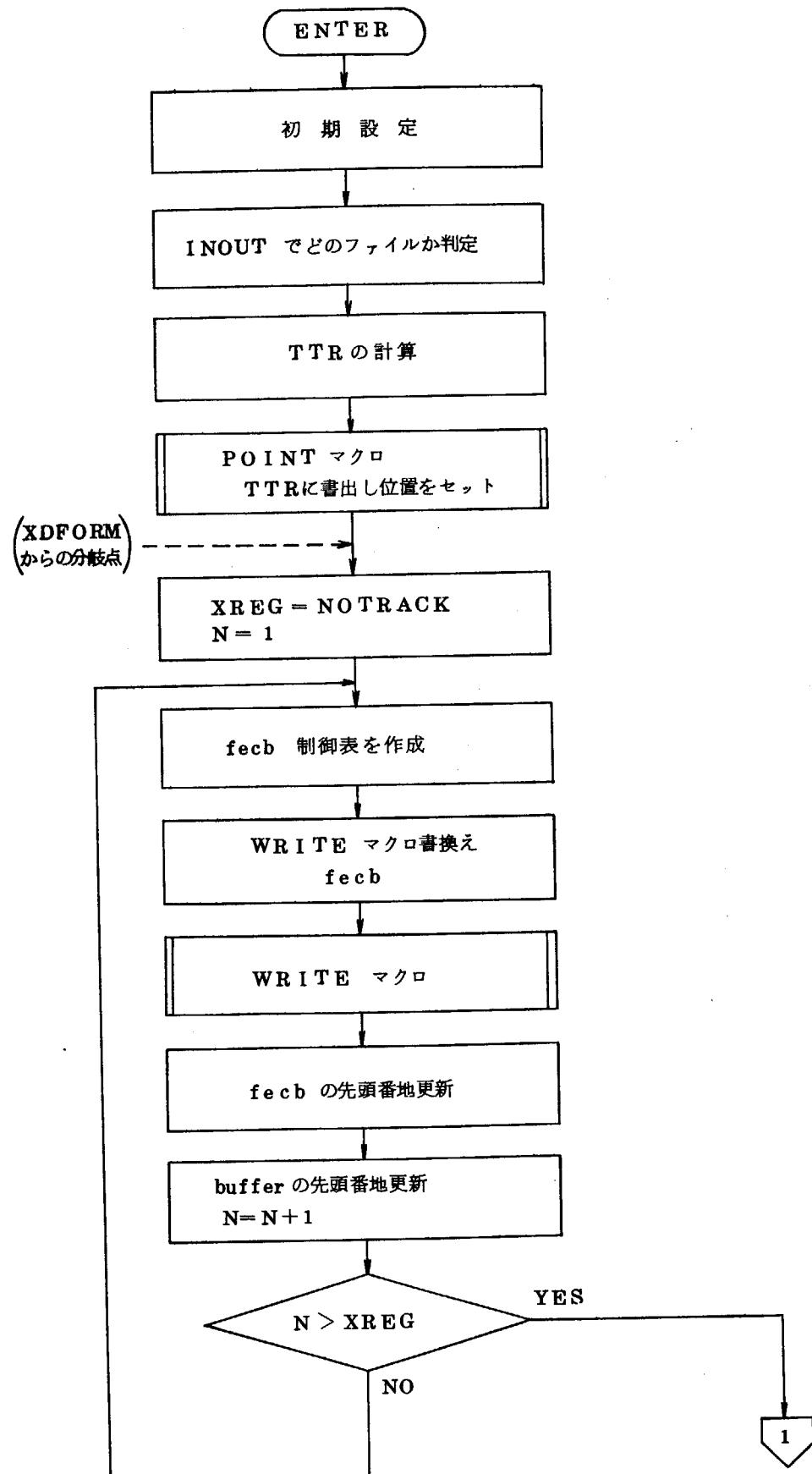
第2図 (続)



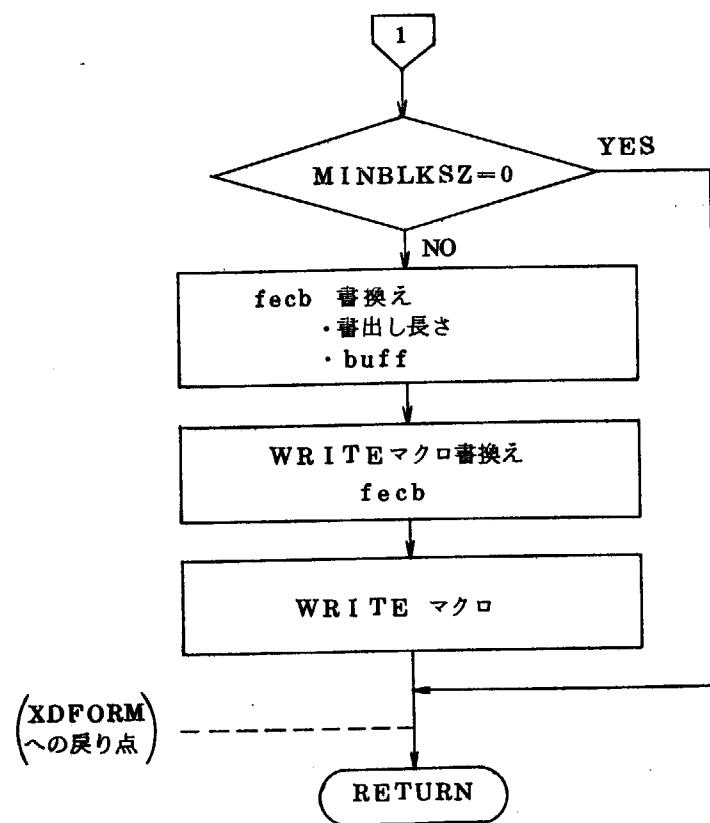
第3図 XDFORM のプログラム・フロー



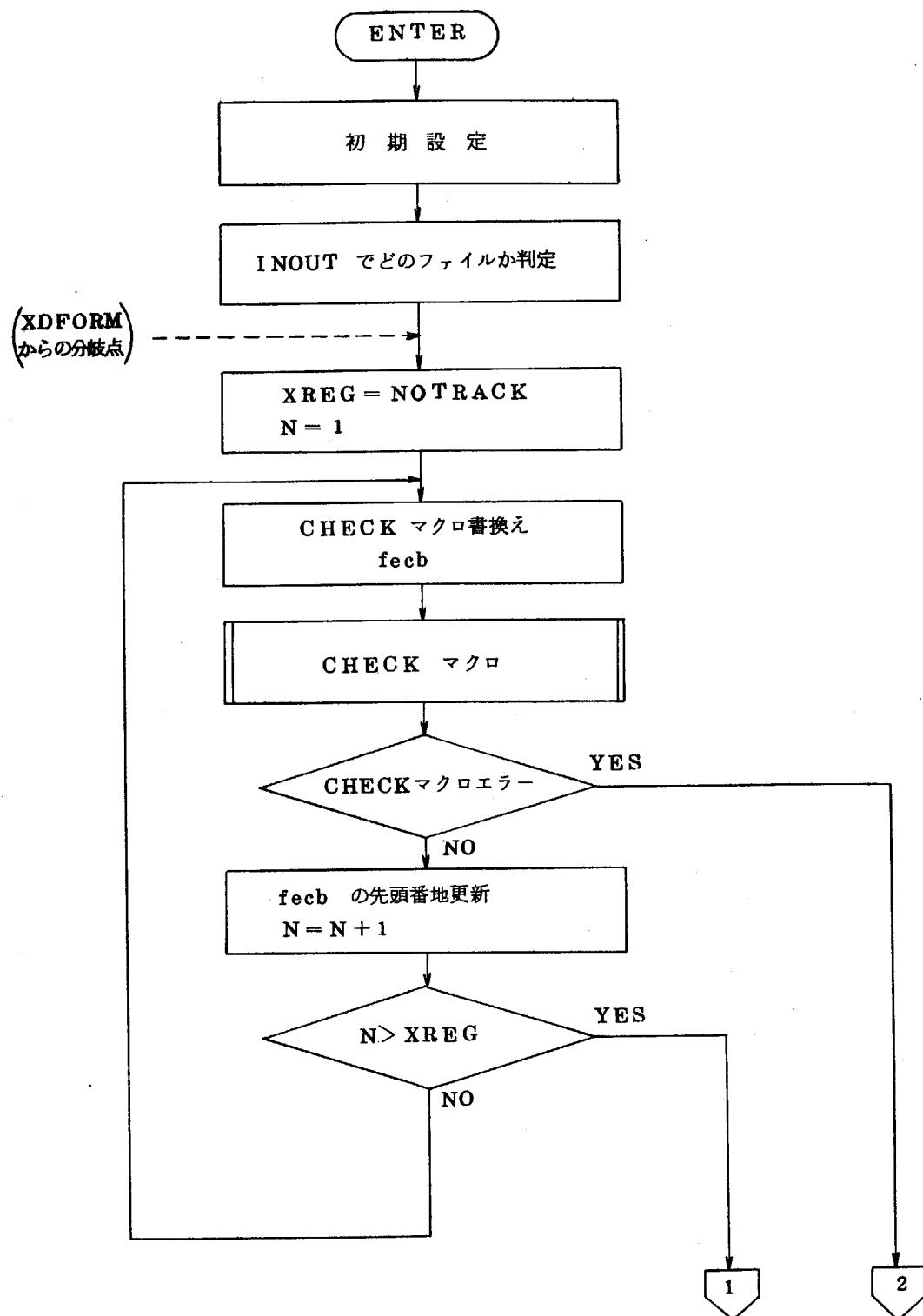
第3図 (続)



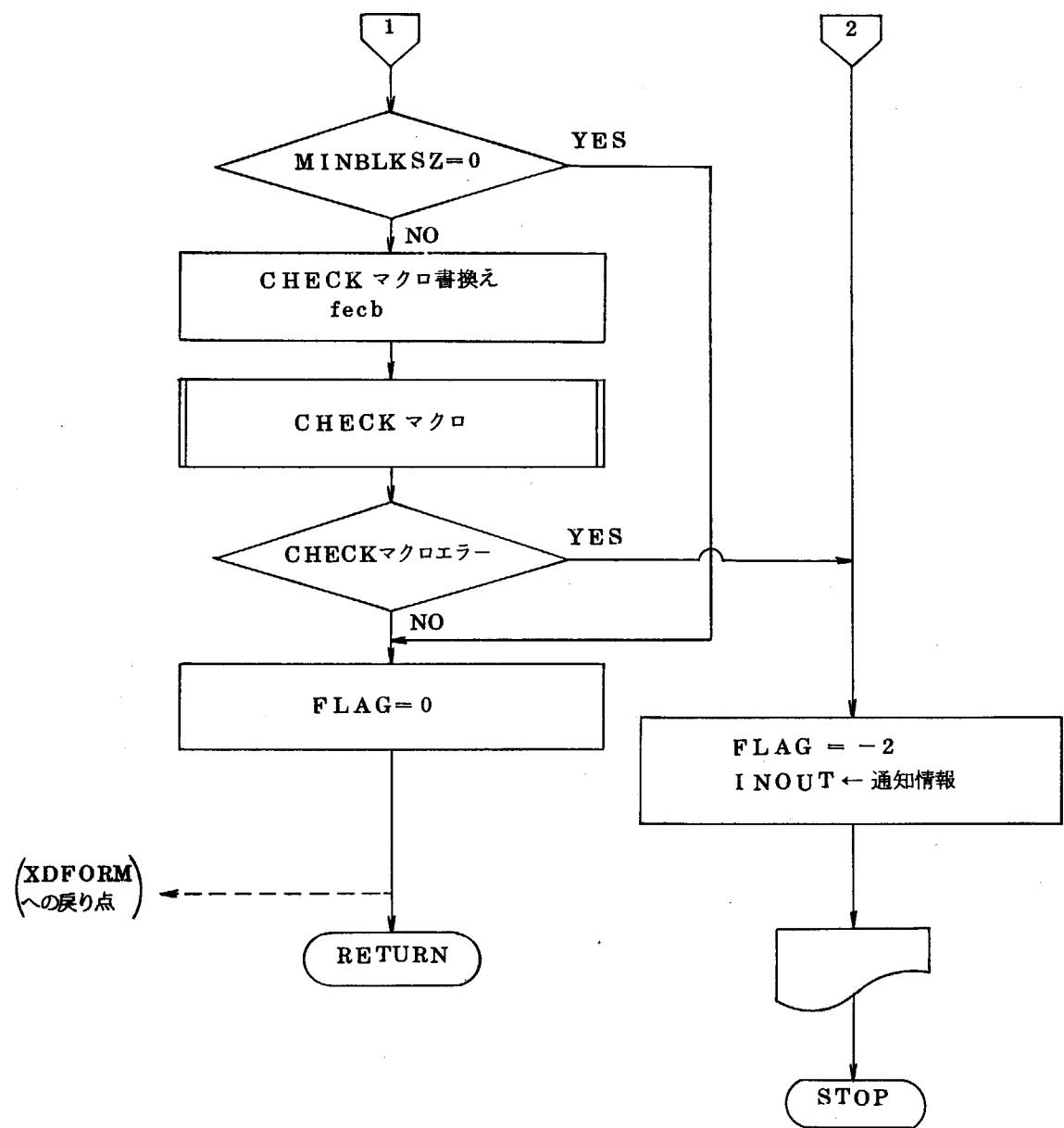
第4図 XDWRIT のプログラム・フロー



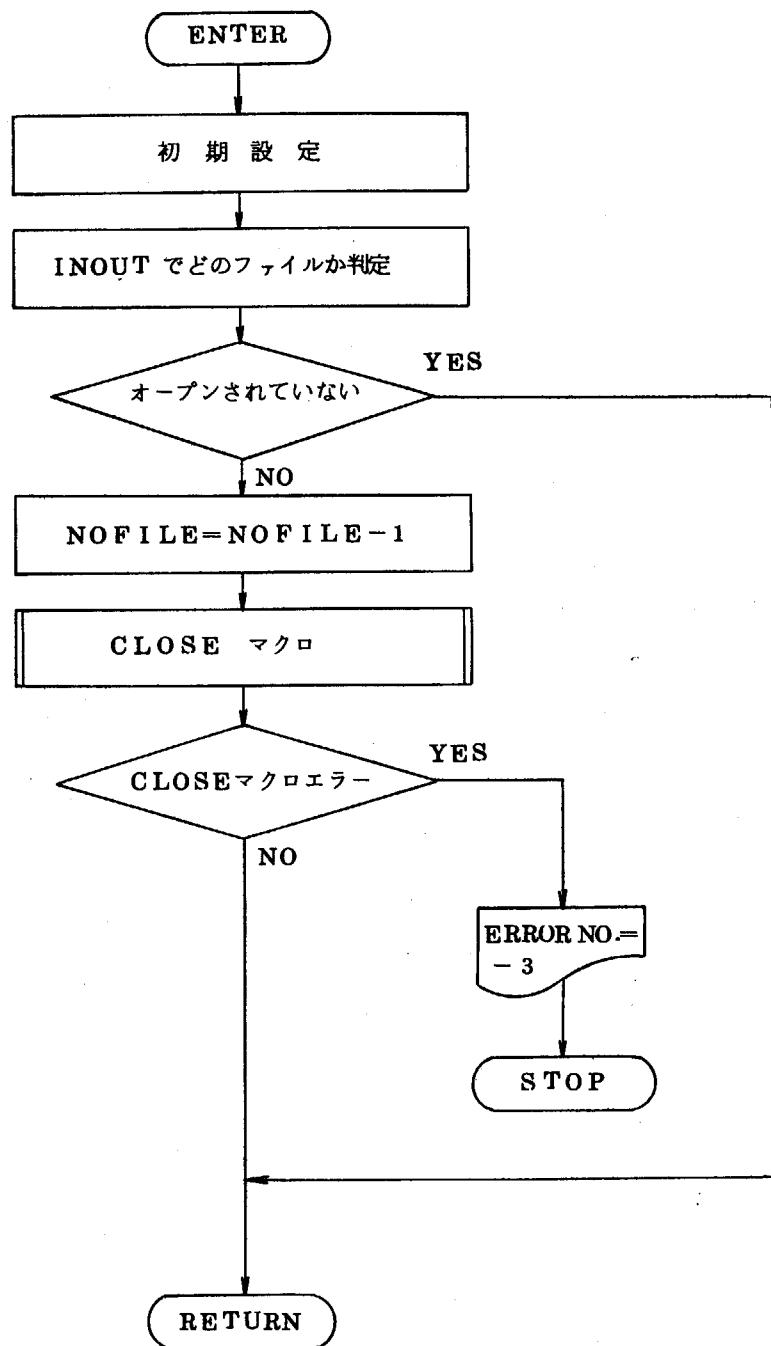
第4図(続)



第5図 XDCHEK のプログラム・フロー



第5図(続)



第6図 XDCLOS のプログラム・フロー

3. 例題とテスト・ランの結果

「FORTXDAM」は、6つの基本サブ・ルーチンから構成されているが、これら6つの基本サブ・ルーチンを組合せることにより、BSAMレベルでのI/Oを実行することができる。この節では簡単な例題とそのテスト・ランの結果を示す。テスト・プログラムを付録Aに掲げる。このテスト・プログラムでは、始めにCOMMON文で1000語のバッファを3つ定義する。XDOPENで2つのファイル（論理機番1，2：ファイル定義名(DNAME1.., DNAME2..)）をオープンする。ファイルの大きさはFD文（マクロ制御文￥LIBEDISKを使っている。）で決めてある。次にXDFORMを呼んで、2つのファイル（ディスク）にフォーマット・ライトを行う。データ数は、変数NBMXで与えられ、これはインプット・データにしてある。演算およびREAD/WRITEは2つのループで行っている。始めのループでは、バッファ1に演算結果を格納し、これをXDWRITで1のファイルのデータ番号1の場所に書きこむ。書き込み中に、演算を続行し、演算結果を2のバッファに格納する。XDCHEKで、バッファ1のデータが1のファイルに転送が終ったことが確認されると、XDWRITを呼んでバッファ2のデータを同じファイル1のデータ番号2の位置に書きこむ。これをNBMX回くり返す（上記の手続を2回と数えて）。次のループでは、3つのバッファを用いて、ファイル1から読みこんで、演算を行い、バッファの内容をファイル2に書き写している。この大ループを抜けると演算、I/Oは完了し、次の入力データを読みこみにいく。1組のデータを読みこんだあと、XDCLOSを呼んでファイルをクローズする。新しいデータに従って再びファイルをオープンし同じ演算、I/Oをくり返す。このプログラムのXDWRIT, XDREADの個所をWRITE文、READ文におきかえて、FORTRAN-H非同期I/O, 通常I/Oの場合もテストした。FORTRAN-HではXDCHEKに相当するものがWAIT文である。付録Aのプログラム・リストの中でFORTRAN-Hの非同期入出力のための文はコメント文にして掲げた。

3つのテスト・プログラムのcpu時間、ジョブの実効的時間(core-time)の結果を第1表に示す。NBMXは上記のようにくり返し計算の回数を示す。READ, WRITEの回数は3*NBMX回である。同じNBMXに対して、3つのテスト・プログラムは同時にスタックをするようにし、また、ジョブ割あて時間(cpu), コア・メモリの占有可能な大きさ、等プログラム実行の優先度をそろえてスタックした。第1表に示されている結果は会計情報でその単位はcpu-time, core-timeともに秒である。これを見ると「FORTXDAM」がcpu-time, core-timeとも格段と優れていることがわかる。FORTRAN-Hの非同期I/Oはcpu-timeが非常に長くなっているが、これは見かけ上のことで、その理由については4章で記す。FORTRANの通常I/Oの「FORTXDAM」に対する比率は、core-timeで2から20であるが、cpu-timeは、NBMXの値にかゝわらず、ほど3である。

第1表 「FORTXDAM」のテスト・ランの結果

NBMX	FORTXDAM		FORTRAN-H 非同期 I/O		FORTRAN-H 通常 I/O	
	cpu-time	core-time	cpu-time	core-time	cpu-time	core-time
10	0.224	7.891	3.119	27.129	0.897	11.304
20	0.546	21.990	6.347	47.764	1.683	18.706
30	0.888	36.811	9.593	105.494	2.479	78.831
40	1.027	29.795	12.616	65.576	3.289	77.671
50	1.358	26.132	15.844	63.735	4.031	58.802
60	1.672	67.573	19.054	82.245	4.833	194.539
70	1.830	19.752	22.089	226.920	5.628	355.267
80	2.162	24.813	25.320	232.910	6.578	529.986
90	2.446	27.986	28.541	522.427	7.208	66.150
100	2.766	40.428	31.768	600.459	8.260	72.334
200	5.329	63.374	63.310	634.658	16.499	764.184
300	8.078	106.274	95.019	287.847	24.482	152.549
400	10.645	208.250	126.610	498.363	32.608	320.259
500	13.425	305.986	158.292	332.703	40.999	438.458
600	16.160	401.367	189.990	795.513	49.174	1306.442
700	18.743	211.726	221.856	692.594	57.163	1437.268
800	21.579	292.880	253.206	580.981	65.396	645.407
900	23.937	484.110	284.870	849.006	73.371	1367.537
1000	26.840	771.305	316.839	2174.603	81.214	2744.478
2000	52.911	1112.980	633.436	5221.866	162.842	4004.876
平均		213.071		702.140		732.252

比較のためFORTRAN-Hの非同期I/O、通常I/Oのテスト・ランの結果を掲げる。cpu-time, core-timeとも会計情報でその単位は「sec.」である。

4. ま　と　め

BSAMを使用した特殊 I/O ルーチンを作成した。このプログラム「FORTXDAM」は FAST 言語で書かれており、6つの基本サブ・ルーチンから成る。この6つの基本サブ・ルーチンは FORTRAN-H の CALL 形式で呼び出すことができ、これら6つの基本ルーチンを適当に組合せ CALL することにより、FORTRAN-H のもとでのアプリケーション・プログラムの中で BSAM レベルでの I/O を実行することができる。「FORTXDAM」の簡単なテスト・プログラムを作成し、FORTRAN-H での通常 I/O、非同期 I/O を使った場合と効率を比較した。その結果、「FORTXDAM」の有効性が第1表のように示された。しかしテストはほんの一例にしかすぎない。バッファ・サイズ、演算回数、READ/WRITE の回数をいろいろ変化させて、さらにテストする必要がある。

なお、FORTRAN-H の非同期 I/O が非常に大きい cpu-time を費やしているが、これは、READ/WRITE ごとにサブ・タスクを発生させ、実測の cpu-time の他にこれらのサブ・タスクが使った cpu-time も加算されているからである。原研の計算センタの FACOM 230-75 では、サブ・タスクの cpu-time を測定する時計は 1.00 msec を単位にしている。従って、1 回のサブ・タスクを発生させると実際の cpu-time は 1.0 msec 以下であるのに、1.00 msec の時間が加算される。これは、近々、FORTRAN-H の修正で切り上げる前の実際の cpu-time が加算されるようになる予定である。FORTRAN-H の非同期 I/O の場合の実測の cpu-time は「FORTXDAM」の結果より、極くわずか長いだけであるから、サブ・タスクが費す cpu-time を正しく加算すると、FORTRAN-H 非同期 I/O の場合の全 cpu-time は、FORTXDAM によるものと、FORTRAN-H 通常 I/O によるものとの中间の値になるものと期待される。

謝　　辞

著者は、有益な議論・助言をして下さった、理論解析研究室の田中正俊室長・常松俊秀氏、および、計算センタの浅井清氏、富山峰秀氏に対し、心から感謝いたします。また、日本アイ・ビー・エム社の、宮野敏男氏、田部秀一氏の有益なコメントに対し感謝いたします。

参　考　文　献

- [1] FACOM 230 M-VII FORTRAN WH 文法書
(FACOM 230-75 75SP-6011-2, 富士通株式会社, 昭和 50 年 6 月)
- [2] FACOM 230 M-VII システムマクロ文法書 I, II, III

(FACOM 230-60/75 75SP-0431-3, 富士通株式会社, 昭和48年12月)

[3] FACOM 230 M-VI/VII FASP文法編

(FACOM 230-60/75 75SP-0811-2, 富士通株式会社, 昭和49年1月)

[4] FACOM 230-75 解説(FACOM 230-75 75EX0001-2, 富士通株式会社, 昭和48年6月)

付録A. 「FORTX DAM」のテスト・プログラム

FORTRAN-H の非同期 I/O のためのルーチンはコメント文で併記してある。

```

C
C
C***** TEST OF FORTXDM *****
C*                                *
C*                                *
C*      TEST OF FORTXDM          *
C*                                *
C*                                *
C***** LIBEDISK DNAME1 .DATA1   *
C*      LIBEDISK DNAME2 .DATA2   *
C***** *****
C
1      COMMON /BUF/ BUFF(1000,3)
2      DOUBLE PRECISION NAME
3      DOUBLE PRECISION DN1,DN2
4      DATA NAME/8TESTXD /
5      DATA DN1,DN2/8HDNAME1 ,8HDNAME2 /
6      NAMELIST /TEST/ JJMAX,NRMX,NSIZE,L1,L2
C
C *** DEFAULT ***
C
7      JJMAX=30
8      NSIZE=1000
9      NBMX=10
10     L1=1
11     L2=2
C
12     WRITE(6,666) NAME
13     666 FORMAT(////+1H ,A8)
C
14     1 READ(5,TEST,ERR=9998,END=9999)
15     WRITE(6,777)
16     777 FORMAT(1H //)
17     WRITE(6,TEST)
C
C ***** CLOSE ***
18     CALL XDCLOS(L1)
19     CALL XDCLOS(L2)
C***** *****
C
20     LENG=NSIZE*9/2
C
C ***** OPEN ***
21     CALL XDOPEN(L1,LENG,DN1)
22     CALL XDOPEN(L2,LENG,DN2)
C***** *****
C
23     NB1=NBMX
24     NB2=NBMX
C
C ***** FORM ***
25     CALL XDFORM(L1,NB1,BUFF)
26     CALL XDFORM(L2,NB2,BUFF)
C***** *****
C
27     CALL CLOCKM(MSO)
C
28     KW=1
29     LBC=1
30     DO 10 NB1,NBMX
C
C *** CALCULATION ***
C
31     IF(NB,EQ,1) CALL CLOCKM(ITIME1)
C
32     DO 20 I=1,1000
33     BUFF(1,LBC)=FLOAT(I)
34     DO 25 JJ=1,JJMAX
35     BUFF(1,LBC)=BUFF(1,LBC)*FLOAT(I)
36     BUFF(1,LBC)=BUFF(1,LBC)-FLOAT(I)
37     BUFF(1,LBC)=BUFF(1,LBC)*1.0E-8
38     BUFF(1,LBC)=BUFF(1,LBC)*BUFF(1,LBC)
39     BUFF(1,LBC)=BUFF(1,LBC)+FLOAT(I)
40
41     25 CONTINUE
42     20 CONTINUE
C
43     IF(NB,EQ,1) CALL CLOCKM(ITIME2)
44     IF(NB,EQ,1) LTIME=ITIME2-ITIME1
45     IF(NB,EQ,1) WRITE(6,*)
46     823 FORMAT(//1, CALCULATION TIME CPU =1.10*MSEC//)
C
C *** WRITE ***
47     IF(NB,EQ,1) GO TO 11
48     CALL XDCHEK(KW)
C
49     11 CONTINUE
50     LBW=LBC
      CALL XDWRT(KW,NB,BUFF(1,LBW))
      WRITE(KW,1D+KW) BUFF(1:LBW),,BUFF(1000,LBW)
C***** *****
C
51     LBC=MOD(LBC,2)+1
52     10 CONTINUE
53     CALL XDCHEK(KW)
C
54     CALL CLOCKM(MS1)
55     MS=MS1-MS0
56     WRITE(6,100) MS
C
57     KR=KW
58     KW=2

```

```

59      LBR=1          XDM01150
60      CALL XDREAD(KR,1,BUFF(1,LBR))    XDM01160
61      C           REWIND KR          XDM01170
62      C           READ(KR,1D=KR) BUFF(1,LBR)...BUFF(1000,LBR) XDM01180
63      C           LBC=1            XDM01190
64      DO 50 NB=1,NBMAX                 XDM01200
65      C           *****          XDM01210
66      C           *** READ ***     XDM01220
67      C           CALL XCHEK(KR)      XDM01230
68      C           WAIT(KR,1D=KR)     XDM01240
69      C           IF(NB,EQ,NBMAX) GO TO 51 XDM01250
70      C           NB1=NB+1          XDM01260
71      C           LBR=MOD(LBR,3)+1   XDM01270
72      C           CALL XDREAD(KR,NB1,BUFF(1,LBR))    XDM01280
73      C           READ(KR,1D=KR) BUFF(1,LBR)...BUFF(1000,LBR) XDM01290
74      C           *****          XDM01300
75      C           51 CONTINUE      XDM01310
76      C           *****          XDM01320
77      C           *****          XDM01330
78      C           *****          XDM01340
79      C           *****          XDM01350
80      C           *****          XDM01360
81      C           *** CALCULATION *** XDM01370
82      C           DO 60 I=1,1000      XDM01380
83      C           BUFF(I,LBC)=FLOAT(I)  XDM01390
84      C           DO 65 JJ=1,JJMAX      XDM01400
85      C           BUFF(I,LBC)=BUFF(I,LBC)*FLOAT(I)  XDM01410
86      C           BUFF(I,LBC)=BUFF(I,LBC)-FLOAT(I)  XDM01420
87      C           BUFF(I,LBC)=BUFF(I,LBC)*1.0E-8    XDM01430
88      C           BUFF(I,LBC)=BUFF(I,LBC)+BUFF(I,LBC)  XDM01440
89      C           BUFF(I,LBC)=BUFF(I,LBC)+FLOAT(I)  XDM01450
90      C           65 CONTINUE      XDM01460
91      C           60 CONTINUE      XDM01470
92      C           *****          XDM01480
93      C           *****          XDM01490
94      C           *****          XDM01500
95      C           *** WRITE ***     XDM01510
96      C           IF(NB,EQ,1) GO TO 52 XDM01520
97      C           CALL XCHEK(KW)      XDM01530
98      C           WAIT(KW,1D=KW)     XDM01540
99      C           52 CONTINUE      XDM01550
100     C           LBW=LBC          XDM01560
101     C           CALL XDWRIT(KW,NB,BUFF(1,LBW))  XDM01570
102     C           WRITE(KW,1D=KW) BUFF(1,LBW)...BUFF(1000,LBW) XDM01580
103     C           *****          XDM01590
104     C           *****          XDM01600
105     C           LBC=MOD(LBC,3)+1  XDM01610
106     C           50 CONTINUE      XDM01620
107     C           CALL XCHEK(KW)      XDM01630
108     C           WAIT(KW,1D=KW)     XDM01640
109     C           *****          XDM01650
110     C           CALL CLOCK(MS2)    XDM01660
111     C           MS=MS2-MS1        XDM01670
112     C           WRITE(6,100) MS    XDM01680
113     C           100 FORMAT(1H0,10X,'***** CPU = ',I6,' MSEC *****') XDM01690
114     C           *****          XDM01700
115     C           GOTO 1          XDM01710
116     C           *****          XDM01720
117     C           *****          XDM01730
118     C           STOP          XDM01740
119     C           *****          XDM01750
120     C           9999 WRITE(6,999)    XDM01760
121     C           999 FORMAT(1H ,//,,,' INVALID INPUT DATA!') XDM01770
122     C           STOP          XDM01780
123     C           *****          XDM01790

```

付録B. 「FORTX DAM」のプログラム・リスト

BINARY CARD 0000

003566	0231	4	1	000750	41	STA	NOTRACK+1	SET NOTRACK (= A)	FOR01760	443
003567	0232	4	1	000770	41	STR	MINBLKSZ+1	SET MINBLKSZ (= R)	FOR01770	444
003570	5100	4	0	003607	41	JNZA	+15	TEST NOTRACK=0	FOR01780	445
003571	0103	4	0	003410	41	LA	TRAKSIZE	YES--CALCULATE BLKSIZE	FOR01790	446
003572	4026	0	0	000007	40	AI	7		FOR01800	447
003573	0110	4	0	003411	41	S	LENGTH		FOR01810	448
003574	4031	0	0	000000	40	XAR	,		FOR01820	449
003575	0103	4	0	003411	41	LA	LENGTH		FOR01830	450
003576	4026	0	0	000207	40	AI	135		FOR01840	451
003577	0231	4	0	003420	41	STA	TEMP		FOR01850	452
003600	4023	0	0	000000	40	LAI	0		FOR01860	453
003601	0074	4	0	003420	41	D	TEMP		FOR01870	454
003602	4026	0	0	000001	40	AI	1		FOR01880	455
003603	0231	4	1	000730	41	STA	NO1BLOCK+1	SET NO1BLOCK (= A)	FOR01890	456
003604	0103	4	0	003411	41	LA	LENGTH		FOR01900	457
003605	0231	4	1	000710	41	STA	MAXBLKSZ+1	SET MAXBLKSZ (= LENGTH)	FOR01910	458
003606	0233	4	1	000070	41	STZ	MINBLKSZ+1		FOR01920	459
003607	0124	4	1	000670	41	LX+4	FCB+1		FOR01930	460
003610	0122	4	7	000002	40	LX+2	2+7	SET FDNAME	FOR01940	461
003611	0103	4	2	000000	40	LA	0+2		FOR01950	462
003612	0231	4	4	000014	40	STA	12+4		FOR01960	463
003613	0103	4	2	000001	40	LA	1+2		FOR01970	464
003614	0231	4	4	000015	40	STA	13+4		FOR01980	465
003615	4023	0	0	000143	40	LAI	99		FOR01990	466
003616	0702	4	1	000650	41	TE	LOGICNO+1		FOR02000	467
003617	4000	4	0	003637	41	J	MODFCB		FOR02010	468
003620	0103	4	1	003432	41	LA	FCB,AD+1	PREOPEN	FOR02020	469
003621	0231	4	0	003626	41	STA	*+5		FOR02030	470
003630	0640	4	4	000017	40	PREOPEN	FCB00,FLTYP=OUTPUT	PREOPEN	FOR02040	471
003631	4000	4	0	003637	41	TB4N+0	15+4	CHECK	FOR02050	478
003632	0103	4	0	004456	41	J	MODFCB		FOR02060	479
003633	0231	4	0	003456	41	LA	=-1	PREOPEN ERROR	FOR02070	480
003634	0103	4	4	000007	40	STA	NOERR	NOERR = -1	FOR02080	481
003635	0231	4	0	003457	41	LA	7+4		FOR02090	482
003636	4000	4	0	003707	41	STA	INFORM		FOR02100	483
003637	0103	4	1	000710	41	J	OPENEROR		FOR02110	484
003638	0441	4	0	000020	40	MODFCB	LA	MAXBLKSZ+1	FOR02120	485
003640	0125	4	1	000710	41	STAD	16+4	BLKSIZE = MAXBLKSZ	FOR02130	486
003641	4435	4	4	000005	40	LX+5	MAXBLKSZ+1	BUFL = MAXBLKSZ	FOR02140	487
003643	1703	4	1	000770	41	STXU+5	5+4		FOR02150	488
003644	4000	4	0	003652	41	TNZ	MINBLKSZ+1		FOR02160	489
003645	4023	0	0	000001	40	J	*+6		FOR02170	490
003646	1701	4	1	000750	41	LAI	1		FOR02180	491
003647	4000	4	0	003652	41	TL	NOTRACK+1		FOR02190	492
003650	1510	4	4	000020	40	J	*+3		FOR02200	493
003651	4000	4	0	003653	41	MB1F+0	16+4	OPTCD = B	FOR02210	494
003652	0510	4	4	000020	40	J	*+2		FOR02220	495
003653	4023	0	0	000143	40	HB1N+0	16+4	OPTCD = (C,B)	FOR02230	496
003654	1702	4	1	000650	41	LAI	99		FOR02240	497
003655	4000	4	0	003671	41	TNE	LOGICNO+1	IF(LOGICNO ,NE, 99) GO TO OPENFILE	FOR02250	498
003656	0103	4	0	004457	41	J	OPENFILE		FOR02260	499
003657	0441	4	0	000017	40	LA	=0/040040	(R,W)	FOR02270	500
003660	0103	4	1	003432	41	STAD	15+4	MACRF	FOR02280	501
003661	0231	4	0	003666	41	LA	FCB,AD+1		FOR02290	502
003670	4000	4	0	003701	41	STA	*+5		FOR02300	503
003671	0103	4	1	003432	41	OPEN	FCB00,FLTYP=UPDATE		FOR02310	504
003672	0231	4	0	003677	41	J	OPENCHECK		FOR02320	511
003701	0600	4	4	000017	40	OPENFILE	LA	FCB,AD+1	FOR02330	512
003702	4000	4	0	003712	41	STA	*+5		FOR02340	513
003703	0103	4	0	004456	41	OPEN	FCB00,FLTYP=OUTPUT		FOR02350	514
003704	0231	4	0	003456	41	TB0N+0	15+4		FOR02360	521
003705	0103	4	4	000007	40	J	OPENEXIT	OPEN NORMAL END NOERR = -1	FOR02370	522
003706	0231	4	0	003457	41	LA	=-1		FOR02380	523
003707	4023	0	0	000000	40	STA	NOERR		FOR02390	524
003710	5000	4	0	000005	65	LA	7+4		FOR02400	525
003711	0257	4	0	000005	42	STA	INFORM		FOR02410	526
003712	4650	4	0	000640	41	OPENEROR	LAI	0	FOR02420	527
003713	4000	4	7	000003	40	JOINTS	XDMESG		FOR02430	528
						SXJ+7	XDMESG	JUMP TO MESSAGE ROUTINE	FOR02440	529
						RSTX	SVINDEX	RESTORE INDEX=REG, RETURN	FOR02450	530
						J	3+7	EJECT	FOR02460	531
									FOR02470	532

004204	0231	4	3	001013	41		STA	FECB+3,3	(FECB+3)		FOR04400	743
004205	0233	4	3	001014	41		STZ	FECB+4,3	(FECB+4)		FOR04410	744
004206	0104	4	0	004211	41		LR	**3			FOR04420	745
004207	6000	0	0	000006	42		TRP	S,WRITES			FOR04430	746
004210	4000	4	0	004213	41		J	**3			FOR04440	747
004211	0000	4	0	004212	60		ADCON	**1			FOR04450	748
004212	0000	4	3	001010	60		ADCON	FECB,3			FOR04460	749
004213	5000	0	0	003761	41	WFORMBAK	NOP	WFORMRET			FOR04470	750
004214	4000	4	0	004222	41		J	**6			FOR04480	751
004215	0103	4	0	004460	41	WRITEROR	LA	==6	NOERR = -6		FOR04490	752
004216	0231	4	0	003456	41		STA	NOERR			FOR04500	753
004217	4023	0	0	000000	40		LAI	0			FOR04510	754
004220	3000	4	0	000003	65		JOINTS	XDMESG			FOR04520	755
004221	0237	4	0	000003	42		SXJ,7	XDMESG	JUMP TO MESSAGE ROUTINE		FOR04530	756
004222	4650	4	0	000640	41		RSTX	SVINDEX			FOR04540	757
004223	4000	4	7	000003	40		J	3,7	RETURN		FOR04550	758
									EJECT		FOR04560	759


```

1      SUBROUTINE XDMESG          XDM00010
2      COMMON /XDZZZ/ KIND, ISN, NOERR, INFORM, INOUT   XDM00020
3      REAL*8 XDNAMES(6)           XDM00030
4      DATA XDNAMES /8HXDOPEN ,8HXDFORM ,8HXDWRIT ,8HXDREAD ,
1           8HXDCHEK ,8HXDCLOS /
5      WRITE(6,13 XDNAMES(KIND), ISN, INOUT, NOERR      XDM00050
6      1 FORMAT('/////* **** FORTXDM ERROR MESSAGE ****'
1           //5X,A8,5X,!ISN =!15,5X,!FILE =!15,5X,!ERROR NO,!13 )
7      !IF(NOERR,GE,-3) WRITE(6,2) INFORM                XDM00070
8      2 FORMAT(6OX,STATUS '!012)                      XDM00080
9      WRITE(6,3)                         XDM00090
10     3 FORMAT('! EXECUTION TERMINATED!')
11     STOP
12     END

```