バッチジョブ制御言語入門

－SX簡易形のジョブ制御言語－

システム管理掛　青井信一
業務掛　吉川勝

1．まえがき

大阪大学大型計算機センターにはスーパーコンピュータSX-2N（以下，SXと記述），汎用計算機ACOS2020（以下，ACOS）ならびにワークステーションが導入され利用者の計算処理に当たっています。

TSS 処理の場合，利用者一人一人があたかもセンターの計算機を占有しているかのように，また会話的に処理を進めることができるので，プログラムのデバッグを非常に効率よく進めることができます。一方，バッチ処理では，完成したプログラムをバッチジョブとして実行することにより TSS端末の前に長時間貼付にならなくてもよい，また，TSS では実行できない巨大（メモリあるいは演算時間）なジョブが実行できる，演算負担額の面からみてもバッチジョブの方が負担金が少なくてすむ，というような特徴があります。

ここでは，これからセンターの計算機をバッチ処理で利用される方のために，TSS端末からSXをバッチジョブ[*1]で利用するときに必要なジョブ制御言語[*2]（以下，JCL）について説明します。

2．センターの計算機とバッチ処理概要

2.1　センターの計算機システム

センターに設置されているSXの科学演算処理装置（SPU：Scientific Processing Unit）は，利用者プログラムを実行する科学技術計算エンジンともいうべき演算プロセッサ（AP：Arithmetic Processor）と，資源管理，TSS処理などのシステム制御を行う制御プロセッサ（CP：Control Processor）の二つのプロセッサから構成されています。また，ACOSの演算処理装置（EPU：Executing Processing Unit）は SXの

[*1] ジョブとは，いくつかのプログラムと，それに関連あるデータからなる利用者の仕事を処理する単位です。ジョブを構成している個々のプログラムの実行をSXではジョブステップ，ACOSではアクティビティと呼びます。

[*2] ジョブをシステムに投入し，いくつかのジョブステップを実行し，結果を出力する一連の処理機能の制御がジョブ制御であり，これらのすべてをジョブ制御言語のジョブ制御文で記述します。

ような機能分担はなく利用者プログラムの実行の他に，資源管理，TSS処理などシステムの制御も受け持っています。

スーパーコンピュータＳＸ－２Ｎ	汎用計算機ＡＣＯＳ２０２０
演算プロッセサ（ＡＰ） ・バッチ利用者プログラムの実行	**演算処理装置（ＥＰＵ）** ・ＴＳＳ処理 ・システムプログラムの実行 ・ＴＳＳ利用者プログラムの実行 ・バッチ利用者プログラムの実行 ・その他
制御プロセッサ（ＣＰ） ・ＴＳＳ処理 ・システムプログラムの実行 ・ＴＳＳ利用者プログラムの実行 ・バッチ利用者プログラムの実行[*1] ・その他	

図１　演算処理装置の機能分担

これらの二つの計算機は全く独立した計算機でありますが光ケーブルを介して接続されています。ただし，SXを利用する場合はACOSを経由して利用するバックエンドプロセッサの形になっており，ACOSに投入されたSX用のバッチジョブは自動的にSXに転送され，実行が終了すればまたACOSに逆転送されます。また，ファイルについてはACOSのファイルをSX-2Nに転送することをステージ，SX-2Nのファイルを ACOSに逆転送することをデステージと言い，JCLあるいはTSSコマンドで簡単に実行できます。

当然ACOSを利用する場合には転送／逆転送と言われる概念はありません。

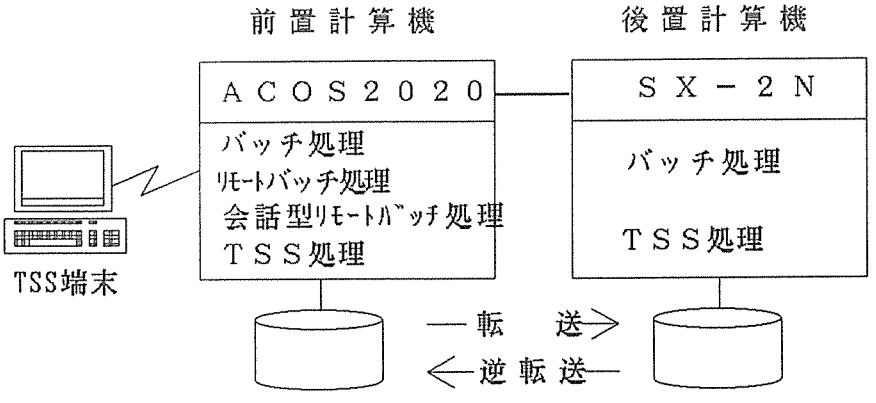


図２　バックエンドプロセッサ方式

[*1] ＳＸのＣＰでバッチジョブの利用者プログラムを実行することも可能ですが，ベクトル化されたプログラムを高速に実行するのはＡＰであり，ＣＰはあくまでもシステムの制御用とお考えください。

ハードウェアのアーキテクチュアはSXとACOSではまったく異なり（SXは1語 32ビット，ACOSは 36ビット），計算機を運用管理するオペレーティングシステムも，SXは SXOS，ACOSは ACOS-6/MVXⅡというオペレーティングシステムでTSS，バッチの利用方法も大きく異なります。

2.2 バッチジョブの利用形態

SXを利用するときには，簡易形と基本形といわれる二つの利用形態があります。本来は基本形と言われる使い方だけですが，従来からの ACOS の利用者のためにACOSの利用形態に合わせて，SXが簡単に利用できるように新たに簡易形が追加されています。

なお，ACOSには簡易形，基本形という区別はありません。

（1）SX簡易形

簡易形はSXを利用するためのファイルをACOS上に持つ使い方です。SXの利用に際し，指定するファイルはACOSのファイルであり，そのファイルはSX上のワークファイルへ自動的に転送され，処理終了時には，必要に応じてSXからACOSへ自動的に逆転送されます。

SXバッチ簡易形で使用できるソフトウェアは次の通りです。

- プログラム言語：FORTRAN77
- 性能向上支援ツール：ANALYZER/SX
  FORTRAN77言語で記述されたソースプログラムを入力し，プログラムの構造とプログラムの実行に関する種々の解析情報を出力する。
- ライブラリ：科学技術計算ライブラリASL/SX，
  数値計算ライブラリMATHLIB/SX，図形用ライブラリ

（2）SX基本形

基本形はSX上にファイルを持つ使い方であり，基本的にはSXが提供する機能をほぼ全て使用できます。簡易形と同じくジョブはSXに転送され，ジョブ終了時に結果は自動的に逆転送されますが，使用しているファイルはSX上に保存されたままです。もちろん，転送／逆転送用ジョブ制御文を使用すればファイルを転送することも逆転送することも可能です。

３．　ＳＸバッチ簡易形による利用法

３．１　SX簡易形ジョブ制御文記述規則

制御文には次のような記述規則があります。

- 第１カラム＄で始まります。
- カラム８から１５までが，コントロールフィールドで制御文の種類（ＪＯＢ，ＦＲＴ７７など）を指定します。
- カラム１６から７２までが，オペランドフィールドでジョブ制御文に対するオプションを指定します。
- オペランドフィールドの各オプションはコンマ（，）で区切ります。
- オペランドフィールドの定義の終了は空白で表します。オプションの途中に空白を使用することはできません。
- オプションの順序は決められている場合と，任意の場合があります。

３．２　ＦＯＲＴＲＡＮプログラムの構成

（１）翻訳／結合／実行ジョブ構成

簡易形ではFORTRANプログラムを翻訳し，実行する基本的なジョブ構成でしか利用できません。もちろん翻訳処理のみでも可能ですがデバッグでの翻訳であればTSSを利用[*1]するほうが賢明です。１つのジョブは利用者宣言文であるJOB文で始まり，ジョブ終了宣言文であるENDJOB文で終わります。

```
1       8       16
------------------------------------------------------
$       JOB     ;A,U,,,,JPA4                利用者宣言文
$       SX                                  簡易形ジョブ宣言文
$       FRT77   SOURCE                      コンパイラ起動文
        ソースプログラム
$       GO                          利用者プログラム結合・実行文
        データ
$       ENDJOB                              ジョブ終了宣言文
```

一般にFORTRAN77プログラムを実行するためには，次の手順が必要です。

FORTRAN77SXコンパイラによってFORTRAN77プログラムが翻訳され，コンパイルユ

[*1] ＳＸジョブのコンパイル時のエラーチェックをＴＳＳで，速報 No.176をご覧ください。

ニット（CU）が出力されます。このコンパイルユニットはそのままでは実行できず，実行時入出力ルーティン，組み込み関数，組み込みサブルーティン，利用者が作成したサブルーティン副プログラム等のコンパイルユニットの参照をリンカによって解決し，これらを結合，編集して実行プログラムを作成します。これをロードモジュール（LM）と呼び，ロードモジュールが記憶域にロードされ実行されます。

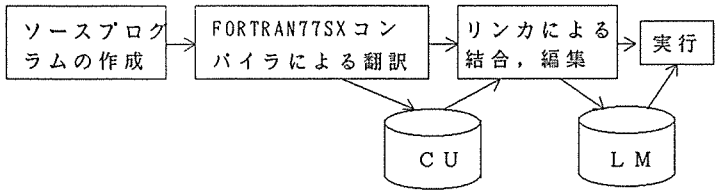


図３　翻訳から実行までの過程

（２）翻訳／結合／実行ジョブ構成の展開

簡易形でジョブを投入するとSXで実行可能なように，簡易形のジョブ制御文が基本形のジョブ制御文に変換され，プログラムおよびデータの転送／逆転送用のステップが追加されSXに転送されます。通常この基本形のジョブ制御文を意識する必要はありませんが，プログラムが正常に終了しない，など問題が発生した場合には，この基本形のJCLの出力されているイメージを調べることも必要になります。

利用者作成ジョブ

```
$      JOB
$$$$$  SX
$$$$$  FRT77
ソースプログラム
$$$$$  GO
データ
$$$$$  PRMFL
$$     ENDJOB
```

→

展開されたSXジョブ

```
$      JOB
```

SX用ワークファイル作成ステップ

```
$FORT77SX
$LINK
```

SX用ファイル転送ステップ

```
$STEP
```

S2000用ファイル逆転送ステップ

```
$      ENDJOB
```

```
$JOB S3315T,USER=W60000,ACCOUNT=W60000,CLASS=U,LIST=SOURCE,
     CPTIME=60,OUTLIM=50000;          ←― ワークファイル作成ステップ
      ≀
$LIBALLOC CU,ACOS.TEMP.S3315T_CU,UNIT=TRACK,PUBLIC,SIZE=17,NORMAL=CATNOW;
$PREALLOC ACOS.TEMP.S3315T_SL01,UNIT=TRACK,SIZE=5,INCRSZ=5,RECSIZE=
 508,BLOCKSZ=32755,MAXEXT=16,PUBLIC,RECFORM=VB,NODELR,NORMAL=CATNOW ;
      ≀
$FORT77SX COMFILE=(ACOS.TEMP.S3315T_SL01,CATLGD),CULIB=(ACOS.TEMP.S3315T_
 CU,CATLGD ),PRFILE=( APR,FILESTAT=SYSOUT),SOURCE ;   ←― 翻訳ステップ
      ≀
$STAGE DATA,PMD=WRITE,SXFILE=ACOS.TEMP.S3315T_USERFIL06,PRFILE=XPR,
 A6FILE='W60000/SX/TESTDATA',CONT=OTHER ;  ←―データ転送ステップ
```

```
$STEP PRS3315T,FILE=(ACOS.TEMP.S3315T_LM,CATLGD),CPTIME=60,OUTLIM=50000,;
      ≀
$ENDSTEP ;                              ←── 利用者プログラム実行ステップ
$DSTAGE DATA,SXFILE=ACOS.TEMP.S3315T_USERFIL06,PRFILE=XPR,A6FILE=
 'W60000/SX/TESTDATA',CONT=OTHER ;
      ≀
$ENDJOB ;                               ←── データ逆転送ステップ
```

もちろん，この展開形は使用されている簡易形のジョブ制御文により異なります。

3.3 ジョブの投入から問い合わせ・結果の取り出し・結果の見方

バッチジョブの投入，処理状況問い合わせ，結果の確認は全てTSS端末から実行できます。

（1）ジョブの投入

端末をTSSと接続し，システム選択レベル（SYSTEM?）で CARDINサブシステム[*1]を呼び出します。投入するジョブの格納されているファイルを指定し，RUNコマンドでジョブを投入します。ジョブが投入されるとシステム側から受付番号が返されます。この受付番号をSNUMB番号といい，これ以降のジョブに対する問い合わせや結果の取り出しについては，このSNUMB番号を使用します。

```
SYSTEM ?CARD                                    ＣＡＲＤＩＮサブシステム
TCMD024 R ＯＬＤまたはＮＥＷ?OLD␣/SX/KANI.J 実行するジョブの呼び出し
*RUN                                            バッチジョブの投入
CRJE001 R カード形式とディスポジション?S,L      形式と出力先の指定
CRJE002 R タブ文字と位置?!,8,16                 タブ文字の指定
CRJE350 I   ＳＮＵＭＢ名 ＃ D160T               ＳＮＵＭＢ番号
*
```

ファイル /SX/KANI.J には，次のようなイメージが格納されています。

```
0010$!JOB!;A,U,,,,JPA4
0020$!SX
0030$!FRT77!SOURCE
0050     ソースプログラム
0070$!GO
0090     データ
0110$!ENDJOB
```

[*1] ＣＡＲＤＩＮサブシステムの使用法の詳細については「ＴＳＳの手引 Ｉ（実習用）」，「TSS-AF会話型リモートバッチ説明書」第2章ＣＡＲＤＩＮサブシステムをご覧ください。

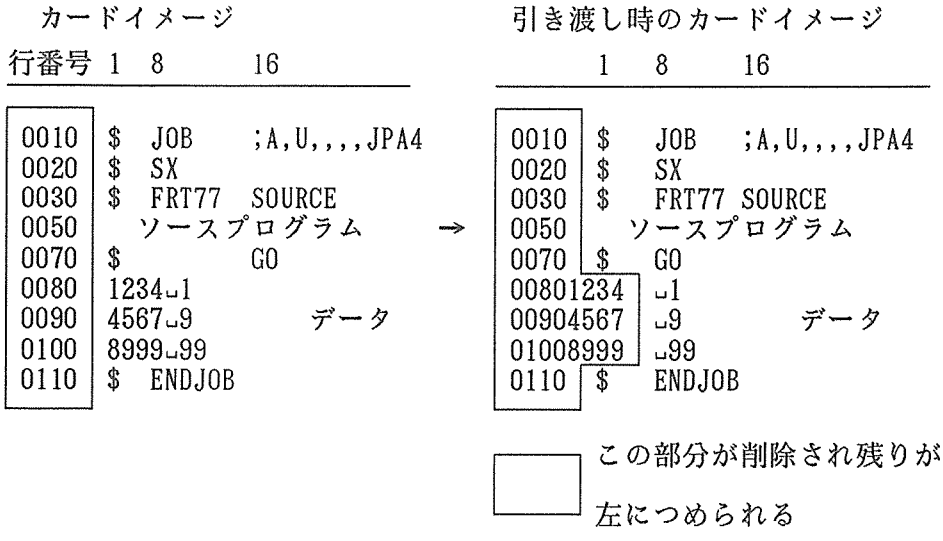
①カード形式の応答

ASIS，STRIP，MOVE，NORMの４つのオプションがあり，何れか１つを必ず指定します。

ⓐSTRIP（S）

バッチシステムに渡す前に行番号[*1]を取ります。各行の最初の数字以外の文字がカードの１桁目になります。

データの部分に１桁目から数値がある場合はこのデータも削除されますので注意してください。１桁目からデータを使用したい場合は行番号とデータの部分を＃で区切ってください。



ⓑASIS（A）

そのままの形式でバッチに渡します。行番号を含んでいないか，あるいは，データの１桁目から数字がある場合に使用します。ただし，ジョブ制御文の前に数字があるとジョブ制御文解析時にエラーとなります。

ⓒMOVE（M）

行番号をカードの73～80に移してバッチに渡します。データの１桁目に数値データがある場合 STRIPと同じ注意が必要となります。

ⓓNORM（N）

MOVEオプションの機能に加えて，標準タブ文字（：）[*2]と標準タブセッティン

[*1] 行に順番をつける目的で行番号があり，データの修正が容易にできます。「TSS-AFシステム説明書」第４章コマンドとファイルの使用法をご覧ください。

[*2] テキストやデータを入力する場合，タブ文字を用いることにより端末から冗長な空白を入力する必要がなくなります。「TSS-AFシステム説明書」第４章コマンドとファイルの使用法をご覧ください。

グ（8,16,32,73）で編集してバッチに渡します。FORTRANのプログラムで配列の下限と上限の間に：を使用している場合は，タブ文字と間違われコンパイル中に警告エラーとなりますのでご注意ください。

カードイメージ　　引渡し時のカードイメージ

```
1                                    1       8        16              73
0010$:JOB:;A,U                       $       JOB      ～              00000010
0020$:SX                             $       SX                       00000020
0030$:FRT77:SOURCE                   $       FRT77    SOURCE          00000030
0040      DIMENSION A(10:100)  →        DIMENSION A(10～100)         00000040
0050                                                                  00000050
0060      ソースプログラム                ソースプログラム             00000060
            ≀                                ≀
```

②ディスポジションの応答

LOWER（L）のオプションがあります。必要であればカード形式のオプションの後ろにカンマ（，）で区切って指定します。

投入すべきバッチジョブの英小文字部分をそのままバッチシステムに引き渡します。大文字のみの場合でも指定して問題はありません。小文字を使用し，このオプションを忘れた場合，全て大文字に変換されます。

③タブ文字と位置の応答

この質問はカード形式に対して NORM が指定されなかったときのみ行われます。データを作成するとき，利用者は桁位置を合わせるために空白を連続して打鍵する労をさけるためタブ文字を使用でき，タブ文字として，＃記号以外の文字が使用できます。

タブを使用している場合には，タブ文字の問い合わせにタブ文字と位置を指定します。例えば，！をタブ文字にし，カードイメージファイル上の8,16,40桁目にセットしてある場合には，次のように応答します。

！，８，１６，４０

もし，標準タブ文字と標準タブセットを使用している場合は，カード形式の質問に対してNORM（N）と入力すればこの質問は行われません。

タブ文字を使用していない場合は，そのままキャリッジリターンキーを押します。

④編集情報

上記のようなファイルの編集や処理の方法に関する情報を編集情報といい，コマンドの問答を少なくするため，処理すべきファイル内の１行目に，あらかじめ入れておくことができます。この情報がある場合には，通常の問答は行われません。

編集情報の一般形式は，次のとおりです。

行番号##カード形式とディスポジション␣タブ文字と位置

行番号はあっても無くても構いません。次にRUNコマンドに対する編集情報の例を示します。

- 0010##MOVE␣%,8,16,32
- ##S,L␣!,8,16,32
- 0010##N

（2）状態問い合わせ

投入したジョブの実行状態の確認はTSSコマンドのJSTS，JMONIコマンドで行います。JSTSコマンドは，コマンドが入力された時点のジョブの状態を表示するのに対し，JMONIコマンドはジョブの実行が終了するまで，状態が変わるたびにその状態を端末に表示します。いずれもSNUMB番号の指定が必要です。

```
*JMONI␣SNUMB番号
CRJE452 I C842Tはジョブ転送中です  @ 16:55:00          ……ⓐ
CRJE453 I C842T はジョブ転送が完了しました :5          ……ⓑ
CRJE458 I C842T はシスアウトの転送中です               ……ⓒ
CRJE447 I デマンドファイル内のC842Tは出力待ちですＩＤ＝Q0 @16:55
CRJE600 I NORMAL TERMINATION
*
```

ⓐ投入したバッチジョブがACOS側の転送待ち行列に登録されているか，現在SXに転送中かのどちらかの状態です。繁忙期になるとこの状態が長く続きます。

ⓑSXに転送されて実行が始まっています。

ⓒ実行が終了しACOS側に逆転送中です。

もしSNUMB番号を忘れた場合は，OBEサブシステムのJオプションでシステムに登録されているSNUMB番号の一覧を表示することができます。

```
*OBE␣J
ENTER SNUMB  ?↵
SNUMB JOB-ID      DATE  TIME  DEVICE  CLASS  STATUS
C842T C842T       06/11 16:54   Q0     SX     WAITING PRINTER
E610T MAIL        06/13 16:50   UL     B-CLAS WAITING PRINTER
FUNCTION     ?↵
*
```

システムの状況を知りたい場合は，OBEの Sオプションを使用します。

```
*OBE␣S
  SYSTEM STATUS ON 07/13/90 AT 13:34:02
 * CLASS WAITING
                ⓐ      ⓑ       ⓒ                ⓓ                ⓔ
  CLASS WAITING  EXEC  MAX-COUNT  WAIT-TIME･DATE   EXEC-TIME･DATE
 E-CLAS     0      2       6          :      /  /     12:53  07/13/90
 A-CLAS     0      1       4          :      /  /     12:40  07/13/90
     ≀
 SX(V)      2      2       2      11:58  07/13/90     10:07  07/13/90
 SX(W)      0      1       1          :      /  /     10:45  07/13/90
 * OUTPUT WAITING            * CONNECT  TERMINAL
       PRINT   PLOTTER             TSS      RBE
          1        0               51        3
FUNCTION     ?↵
*
```

ⓐ各クラスの実行待ち件数　　ⓑ各クラスの実行中の件数
ⓒ各クラスの現在の最大実行多重度　ⓓ実行待ちジョブの投入日時
ⓔ実行中ジョブの投入日時

（３）結果の取り出し

結果を取り出す方法には，JOUT/DEMANDサブシステム[*1]で結果を端末に出力する，あるいはIDカードでセンターのプリンタへ出力する，の２つがあります。

①JOUT/DEMANDサブシステムによる検索

処理結果の内容を表示させたり，不要なステップ／アクティビティの結果をプリンタに出力する前に削除できます。

システム選択レベル（SYSTEM?）で JOUTサブシステムを呼び出します。システム側からの SNUMB? の問い合わせに検索したいSNUMB番号を入力し，FUNCTION?の問い合わせにJOUT用のコマンドを入力します。

```
SYSTEM ?JOUT␣snumb番号
CRJE012 R FUNCTION?func
          ≀
```

funcには次の機能があります。

[*1] バッチジョブの結果を端末から操作できます。詳細は「TSS-AF会話型リモートバッチ説明書」第４章JOUT／DEMANDサブシステムをご覧ください。

ⓐACTIVITY␣n

nで指定したステップ／アクティビティに位置づけします。

ⓑLIST␣ALL

処理中のジョブに含まれる全レポートについて，アクティビティ番号の順に，レポートコード[*1]や行数を一覧表の形で表示します。

ⓒEPRINT␣rc

rcはレポートコードであり，現在位置づけられているアクティビティの指定したレポートコードの内容を表示します。

ⓓSPRINT␣rc

使用している端末が画面型の場合に利用でき，指定したレポートをプリンタ出力形式のまま表示します。DONEでFINC?レベルに戻ります。

ⓔKILR␣rc

位置づけられているアクティビティの指定されたレポートを削除します。

ⓕLENGTH␣n

端末に出力する１行当りの文字数を指定します。nが４の倍数でなければnより大きな最小の４の倍数値となります。160以上を指定した場合は160。

ⓖHOLD

結果を後日検索あるいはセンターのプリンタに出力するため保存します。

ⓗRELE

結果を消去します。再び検索あるいは出力することはできません。

JOUTサブシステムから抜け出すためには，RELEまたはHOLDを入力します。

```
SYSTEM ?JOUT␣C842T
CRJE012 R ファンクション?LIST␣ALL
ACTI#          RC     LINE    HOLD   CLASS
  --           $$      --      --      I   ……実行レポートイメージ
 001(FRT7V )   74       14     YES     I   ……FORTRAN ソースイメージ
 002(LINKER)   74       23     YES     I   ……リンカの編集レポート
 003(****RU)   26        2     YES     I   ……WRITE 22 の出力イメージ
 003(****RU)   25        2     YES     I   ……WRITE 21 の出力イメージ
CRJE012 R ファンクション?ACTI␣3
CRJE012 R ファンクション?PRIN␣26
        ≀
CRJE317 I レポート 26 の終了です
CRJE012 R ファンクション?KILR␣26
```

[*1] FORTRANの外部装置識別子番号を８進数になおしたもの。
WRITE(10,nn)→12，WRITE(33,nn)→41，FORTRANコンパイラのソースイメージ→74，ジョブ制御言語イメージ→$$

```
CRJE318 I レポート 26 をＫＩＬＬしました
CRJE012 R ファンクション?LIST␣ALL
ACTI#          RC     LINE    HOLD   CLASS
  --           $$      --      --      I
 001(FRT7V )   74        14    YES     I
 002(LINKER)   74        23    YES     I
 003(****RU)   26*        2    YES     I   ……*印はこのレポートが
 003(****RU)   25         2    YES     I         削除されている表示
CRJE012 R ファンクション?HOLD
SYSTEM ?
```

②ＩＤカードによる出力

センターには本館１階のオープン入出力装置室と入出力棟１階にプリンタが設置されています。これらのプリンタに出力する場合には，それぞれのプリンタが設置してある近くのデマンド出力用[*1]ＩＤカード読み取り装置にＩＤカードを入力してください。システムに保存されている結果がプリンタに出力されます。

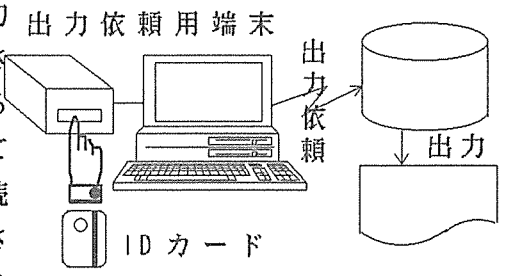


図４　デマンド出力端末

１回の要求で10件まで一度に出力されます。

（４）結果の保存

プリンタに出力されない，あるいはJOUTで消去されないジョブは，計算終了日から８日間保存されています。９日目になると強制的に消去されますのでご注意ください。なお，システムには一利用者番号につき100件のジョブしか投入できません。101[*2]件目になると新たなジョブは投入できなくなります。不用な結果は保存しないでなるべく早めに消去してください。なお，同じ利用者が一度に大量のバッチジョブを投入してもSXでは１件づつしか実行されません。

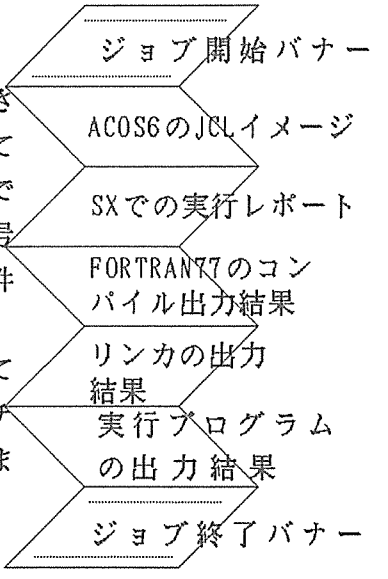


図５　出力形式

（５）出力形式

実行結果をプリンタに出力した場合，図５のような

[*1] 利用者の出力要求があった時点で結果を出力すること。

[*2] JOB COUNT LIMIT ERRORあるいはJOB COUNT LIMIT OVERメッセージが出力されます。この値は混雑期になると変更される場合があります。

形で出力されます。

①開始バナー

登録番号，SNUMB番号あるいはＪＯＢ文で指定した識別名が花文字で表示されます。

②簡易形のJCLイメージ

使用された簡易形のJCLのイメージがそのまま出力されます。

③ＳＸでの実行レポート

SXで実行可能なように展開されたJCLのイメージと実行レポートが出力されます。実行レポート上には，SXでの各ステップの実行時間と終了状態が表示されています。

```
*************************************************************************
 JOB EXECUTION LISTING
*************************************************************************
JOB INITIATION DATE AND TIME : 90-07-04 15:52:59
JNO=10 CLASS=U PRIORITY=15 INITIAL SWITCH=00000000
$JOB SU580T,USER=W60000,CLASS=U,LIST=SOURCE,CPTIME=60,OUTTLIM=50000;
$LIBALLOC CU,ACOS.TEMP.SU580T_CU,UNIT=TRACK,PUBLIC,SIZE=17,NORMAL=CATNOW;
STEP STARTED AT 15:53:00
        PROCESS GROUP NORMALLY TERMINATED.TERMINATION CODE=DONE ― ⓐ
$FORT77SX COMFILE=ACOS_SU580T_SL01,CULIB=(ACOS.TEMP.SU580T_CU,CATLGD),PRF
ILE=(APR,FILESTAT=SYSOUT),SOURCE;
        STEP STARTED AT 15:53:06
        PROCESS GROUP NORMALLY TERMINATED. TERMINATION CODE=WARNING
        STEP TERMINATED AT 15:53:07
             CPU TIME : 0.057 SEC      ―――――――――――――――― ⓑ
              ELAPSED TIME : 1.393 SEC
$LINK PRSU580T,PRIMLIB=(ACOS.TEMP.SU580T_CU,CATLGD),SENTENTRY,OUTLIB=(ACO
S.TEMP.SU580T_LM,CATLGD),PRFILE=(KPR,FILESTAT=SYSOUT),COMMAND='LIST=N,';
         STEP STARTED AT 15:53:14
        PROCESS GROUP NORMALLY TERMINATED. TERMINATION CODE=DONE
        STEP TERMINATED AT 15:53:17
        CPU TIME : 0.129 SEC      ―――――――――――――――― ⓒ
        ELAPSED TIME : 3.505 SEC
$STEP PRSU580T,FILE=(ACOS.TEMP.SU580T_LM,CATLGD),CPTIME=60,OUTLIM=50000,;
$ENDSTEP ;
        STEP STARTED AT 15:53:18
        PROCESS GROUP NORMALLY TERMINATED. TERMINATION CODE=0
        STEP TERMINATED AT 15:53:18
        CPU TIME : 1.016 SEC      ―――――――――――――――― ⓓ
        AP TIME : 1.000 SEC
        ELAPSED TIME : 0.456 SEC
        VECTOR OPERATION RATIO : 80.0 %     ―――――――――― ⓔ
$ENDJOB ;
```

ⓐ終了コード[*1]　　DONE：正常終了

0：正常終了　　FATAL：コンパイラが致命的なエラーを検出

FILEOV：ファイルがオーバーフロー　　PROCEXCP：プログラムが異常終了

TIMEOUT：演算処理装置時間オーバー

ⓑFORTRAN77/SXコンパイラの演算処理装置使用時間

ⓒリンカの演算処理装置使用時間

[*1] コードの詳細は「コードハンドブック」リターンコードをご覧ください。

ⓓ利用者プログラム実行時の科学演算処理装置使用時間（APとCPの合計）

ⓔ利用者プログラムのベクトル化率

特に利用者プログラムの実行ステップ（$STEP）の箇所の終了コードに注意してください。ファイル削除（$DEALLOC）箇所でOBJ1UNKNが出力されている場合がありますが，通常は問題ありません。

④コンパイル出力結果

コンパイル時の結果が出力されます。出力情報の詳細については「FORTRAN77,77/SXプログラミング手引書」翻訳時の出力情報をご覧ください。

⑤リンカ出力結果

リンク時のサマリリストが出力されます。リンク統計情報の最後の行に，実行時必要なAPメモリのサイズが，メガバイト単位で表示されています。

⑥実行プログラム出力結果

実行結果が出力されます。

```
1    8
-------------------
$    JOB
$    SX
$    FRT77   SOURCE    →
     ソースプログラム
          ⋮
$    FRT77   SOURCE    →
     ソースプログラム
$    GO                →
$    ENDJOB            ↘
```

$ GO 文は二つのアクティビティ／ステップに展開されます。

ジョブ実行結果

ACT#--	レポートコード $$（JCLｲﾒｰｼﾞ）
ACT#1	レポートコード 74（ｿｰｽｲﾒｰｼﾞ）
⋮	
ACT#n	レポートコード 74（ｿｰｽｲﾒｰｼﾞ）
ACT#n+1	レポートコード 74（ﾘﾝｶ編集ﾚﾎﾟｰﾄ）
ACT#n+2	プログラム実行結果。実行のレポートコードは，プログラム中の WRITE文と $ SYSOUT文の対応付けられたファイルコードを８進数に変換したもの。

⑦ジョブ終了バナー

出力リストの最終頁に登録番号，JOB文の識別名が大きく花文字で出力され，ジョブに要した費用が次の形式で出力されます。

```
**********************************************************************
*                  ⓐCURRENT USE                                      *
*ⓑJOB  INPUT-ID   ⓒJOB START DAY ⓓCPU TIME USE(AP) ⓔBASIC COST     *
*ⓕJOB OUTPUT-ID   ⓖJOB END   DAY ⓗCPU TIME USE(CP) ⓘCPU TIME COST  *
*ⓙJOB SCHED CLASS                 ⓚPRINT PAGE(LP)   ①PRINT PAGE COST*
*ⓜJOB OUTPUT CLASSⓝELAPSE TIME   ⓞPRINT PAGE(JPR)                  *
*                  ⓟCHANNEL USED TIME              ⓠTOTAL COST      *
*                                                  ⓡAVAILABLE BUDGET*
**********************************************************************
```

ⓐ現在までの利用負担額合計　ⓑｼﾞｮﾌﾞ投入端末　ⓒｼﾞｮﾌﾞ実行開始日時
ⓓAP使用時間　ⓔ利用負担金基本負担額　ⓕ出力ﾌﾟﾘﾝﾀ装置名
ⓖｼﾞｮﾌﾞ実行終了日時　ⓗCP使用時間　ⓘ利用負担金演算負担額
ⓙｼﾞｮﾌﾞ実行ｸﾗｽ　ⓚﾌﾟﾘﾝﾀ出力頁数　①利用負担金出力負担額
ⓜﾌﾟﾘﾝﾀ出力ｸﾗｽ　ⓝ経過時間　ⓞ日本語ﾌﾟﾘﾝﾀ出力頁数
ⓟ入出力ﾁｬﾈﾙ使用時間　ⓠｼﾞｮﾌﾞ利用負担額合計　ⓡ利用可能残額

3.4　ジョブ制御文の詳細

（1）JOB文

利用資格のチェックおよび利用者の識別を行います。ジョブの先頭におきます。

①形式　　1　　8　　16

$　　JOB　　；支払コード，ジョブクラス，［識別名］，
［実行レポート］，［端末識別名］，［出力装置］

②オペランドフィールド

支払コード：支払費目毎に割り当てられたコード[*1]を指定する。必須

国立学校校費などの振替分：A～F
公私立学校経費　　：G～J
科学研究費　　：K～T

ジョブクラス：使用する資源の制限にあったクラスを指定する。必須

システム	ジョブクラス	CPU[*2]時間	プログラムサイズ	出力記録数	作業用ファイル容量
AP	U	900秒（15分）	8000KW（32MB）	50000行	100MB
	V	10800秒（3時間）	24000KW（96MB）		
	W	3600秒（1時間）	48000KW（192MB）		
CP	S	180秒（3分）	4000KW（16MB）		
	T	7200秒（2時間）	8000KW（32MB）		

[*1] ※利用者が支払う計算機利用負担金の費目をコード化したもの，一人で複数の支払コードを持つことができます。

[*2] CPU時間の制限値の規定値は1分。CPはあくまでAPで有効に働くプログラムをデバッグするためのものとお考えください。

［識別名］：計算結果の送り先を指定する。省略可

ＭＡＩＬ……申請者へ郵送する。吹田地区は指定不可

ＲＭＴ……リモートバッチ端局へ出力する

その他（８文字以内）……ディスクに保存（省略時既定値）

［実行レポート］：SXではこのパラメータは意味がない。省略してください。

［端末識別名］：識別名にＲＭＴを指定したときのみ意味を持ち，出力先のリモートバッチ端局ＩＤを指定する。

［出力装置］：結果をプリンタに出力するときの装置を指定する。省略可

ＪＰＡ４……Ａ４サイズの日本語プリンタへ出力する

ＪＰＢ４……Ｂ４サイズの日本語プリンタへ出力する

省略……装置の空き状態により，どちらかのプリンタに出力される

③注意事項

- このオプションの形式はCARDINサブシステムでしか利用できません。
- オプションはこの順序で指定する必要があります。
- 識別名に MAILを指定したジョブは，実行終了後センターのプリンタに自動的に出力され，利用者番号の申請者へ郵送されます。MAIL，RMT以外のジョブはディスクに保存されたままとなります。

④例　カラム　1　8　16

```
$      JOB     ;A,V,,,,JPA4
```

支払コードＡ，ジョブクラスＶ，Ａ４サイズのプリンタに出力する

カラム　1

```
$!JOB!;K,W,,,,JPB4
```

支払コードＫ，ジョブクラスＷ，Ｂ４サイズのプリンタに出力する

タブ文字として ！ を使用している。

（２）ＳＸ文

ＪＯＢ文の次におき，ＳＸで使用する資源（ＣＰＵ時間，出力記録数など）の制限値を変更します。

①形式　1　8　16

```
$    SX     [CPTIME=tttt]
            [,OUTLIM=llllll]
            [,CUSIZE=mm]
            [,LMSIZE=nn]
            [,STAGE={YES|NO}]
```

②オペランドフィールド

CPTIME：利用者プログラムの実行時に使用するＡＰとＣＰの科学演算処理装置の上限値を秒単位で指定する。省略可。既定値は60秒。

OUTLIM：ジョブ全体のシステム標準SYSOUT（プリンタ出力）として出力する論理レコード数の上限値を指定する。省略可。既定値50000行。

CUSIZE：FORTRANのコパイル時出力される，コンパイルユニットを一時的に格納するファイルのサイズをリンク*1数で指定する。省略可。既定値は50ﾘﾝｸ。

LMSIZE：リンク時出力されるロードモジュール（実行可能形式の目的プログラム）を一時的に格納するファイルのサイズをリンク数で指定する。省略可。既定値は50リンク。

STAGE：利用者プログラムの実行が異常終了したとき，実行中に使用していたファイルの内容をACOS側のファイルに転送するか否かを指定する。既定値はＹＥＳ。

YES……異常終了しても，作成された実行結果ファイルの内容をACOS側に転送する。

NO ……異常終了した場合，実行結果ファイルの内容をACOS側には転送しない。

③注意事項

- オプションが１枚に書ききれない場合，$ ETC文で続けて書くことができます。

```
1        8        16
$        SX       CPTIME=3600,
$        ETC      OUTLIM=1000
```

- オプションの順序は任意です。
- 同一オプションを指定した場合，最後に指定したものが有効となります。
- オプションの文字列を誤った場合，そのｵﾌﾟｼｮﾝが無視され既定値が取られます。
- ＳＸのオプションとしてこの他に‘PROC’，‘CLASS’がありますが，これを指定するとジョブは削除されます。
- 指定したジョブクラスの制限値内*2でしかCPTIMEを延長できません。

④例

ｶﾗﾑ

```
1        8        16
$        SX       CPTIME=10800
```

実行時の科学演算処理装置時間を３時間に変更。但し，ジョブクラスはＶでなければならない。

---

*1 ファイルの大きさを表す単位です。１リンク＝１２Ｌリンク，１Ｌリンク＝３２０語。

*2 ジョブクラスのCPU制限値を越えるとSX CPTIME OVER メッセージが実行レポート上に出力されジョブは削除されます。

```
カラム  1
$!SX!OUTLIM=2000,CPTIME=3600
```

出力行数を2000と少なくし，中央処理装置使用時間を１時間に変更
タブ文字として ！ を使用しています。

（3）ＦＲＴ７７文

ＳＸ文の次におき，FORTRAN77/SXコンパイラを呼び出し，翻訳処理を行います。

①形式　　1　　8　　16

```
$    FRT77    オプション［, オプション］
```

オプションの形式

```
［, {SOURCE|NOSOURCE}］ ［, {MAP|NOMAP}］ ［, {FMTLIST|NOFMTLIST}］ ［, {CU|NOCU}］
［, {ELN|ILN}］ ［, {STMTID|NOSTMTID}］ ［, {TEST|NOTEST}］ ［, {DBLINE|NODBLINE}］
［, {DBG|NODBG}］ ［, {ALC|NOALC}］ ［, {BYNAME|NOBYNAME}］ ［, {MRGMSG|SEPMSG}］
［, {CHECK|NOCHECK}］ ［, {EJECT|NOEJECT}］ ［, {SETMASK|NOSETMASK}］ ［ , {SUMMARY|NOSUMMARY}］
［, {FLAG|NOFLAG}］ ［, {INCLIST|NOINCLIST}］ ［, {NCLIST|STDLIST}］ ［, {ENDMARK|NOENDMARK}］
［, {DOLLAR|NODOLLAR}］ ［, {LCASE|NOLCASE}］ ［, {VDIR|NOVDIR}］ ［, {INLOG4|NOINLOG4}］
［,XAREA= {n|mM}］ ［,COMPAT= {I|数字列}］ ［,MAIN=名前］ ［,BLKDATA=名前］
［, {FIXED|FREE|FREE2}］ ［, {OBSERVE|WARNING|FATAL}］ ［,CONSTEXT= {ALL|NOEXP|NONE}］ ［,AUTODBL［=値］］
［,OPT= ( ［ {DIV|NODIV}］ ［, {MOVE|NOMOVE}］ ［, {CHG|NOCHG}］ ) ］
［,MASK= ( ［ {ZDIV|NOZDIV}］ ［, {FLOVF|NOFLOVF}］ ［, {FLUNF|NOFLUNF}］ ［, {FXOVF|NOFXOVF}］ ) ］
［, {SAMEAREA= ( {n|mM} , {SEP|NOSEP} )|NOSAMEAREA}］
［, {INLINE|NOINLINE} ［= ( ［ {WHOLE|LOOP|IMLOOP}］ ［, {ERRCHK|NOERRCHK}］ ［,名前［,名前］］ ) ］ ］
```

〔,{VECTOR / NOVECTOR}〔=(〔{FULLMSG / INFOMSG / NOMSG}〕〔,{VERRCHK / NOVERRCHK}〕〔,LOOPCNT=数値〕
〔,{EXPAND=n / NOEXPAND}〕〔,{IMLOOP / NOIMLOOP}〕〔,VWORKSZ={n / mM}〕
〔,{DIVLOOP / NODIVLOOP}〕〔,{LOOPCHG / NOLOOPCHG}〕〔,{ASSUME / NOASSUME}〕)〕

②オペランドフィールド（オプションの下線は既定値を示します）

SOURCE：ソースプログラムリストを出力する。

NOSOURCE：ソースプログラムリストを出力しない。

MAP：相互参照リストを出力する。

NOMAP：相互参照リストを出力しない。

FMTLIST：ソースプログラムの構造を段付け等によって見やすくしたリストを出力する。

NOFMTLIST：ソースプログラムの構造を段付け等によって見やすくしたリストを出力しない。

CU：コンパイルユニットをCULIBに出力することを指定する。

NOCU：コンパイルユニットをCULIBに出力しないことを指定する。

ELN：診断メッセージ，マップ，相互参照リスト，実行時のエラー情報等の情報における，行の識別に外部行番号を用いることを指定する。

ILN：診断メッセージ，マップ，相互参照リスト，実行時のエラー情報等の情報における，行の識別にコンパイラが生成する内部行番号を用いることを指定する。

STMTID：実行時のエラーメッセージ中に行番号情報を表示させるためにコンパイルユニット中に行番号と目的コード位置の対応表を出力することを指定する。

NOSTMTID：コンパイルユニット中に行番号と目的コード位置の対応表を出力しないことを指定する。

TEST：デバッグサポートプログラムを用いたデバッグを行う。

このオプションが指定されたとき，

- 最適化は行われない。
- INLINEオプションが指定されても無視される。
- VECTORオプションが指定されても無視される。また，ベクトル化指示行でVECTORを指定しても無視される。
- OPTオプションを指定しても無視される。

NOTEST：デバッグサポートプログラムを用いたデバッグを行わない。

DBLINE：デバッグ行を翻訳の対象とする。

NODBLINE：デバッグ行を注釈行とする。

DBG：デバッグ文（SUBCHECK文およびSUBSTRCHK文）を翻訳の対象とする。

このオプションが指定されたとき，

- 最適化は行われない。
- INLINEオプションが指定されても無視される。
- OPTオプションが指定されても無視される。

NODBG：デバッグ文を翻訳の対象としない。

ALC：共通ブロック内の各々のデータを，最も参照効率がよくなるように割り付ける。

NOALC：共通ブロック内のデータの間を詰めて割り付ける。

BYNAME：仮引数である変数名で，仮引数の並びの中で斜線で囲まれていないものを位置取り（reference by location）にする。

NOBYNAME：仮引き数である変数で仮引き数の並びの中で斜線で囲まれていないものを値取り（reference by value）にする。ただし，長さが（*）で指定されている文字型の変数や斜線で囲まれている変数は常に位置取りになる。

MRGMSG：診断メッセージをソースプログラムリストの中に埋め込んで出力する。

SEPMSG：診断メッセージをソースプログラムリストと分離して出力する。

CHECK：配列要素名における添字の値の検査機能と，部分列名における部分列式の値の検査機能を無条件に働かせる[*1]。

NOCHECK：配列要素名における添字の値の検査機能と，部分列名における部分列式の値の検査機能を無条件には働かせない。SUBCHECK文，SUBSTRCHK文の有無およびDBGオプションの指定の有無により，検査機能を働かせるか否かが決まる。

EJECT：プログラム単位やリストの種類が変わるたびにリストの改ページを行う。

NOEJECT：プログラム単位やリストの種類が変わってもリストの改ページを行わず，数行の空白をはさんで出力を続ける。

SETMASK：副プログラム単位の入口・出口で，MASKオプションに従った例外制御マスク値設定を行う。

NOSETMASK：副プログラム単位の入口・出口では，例外制御マスク値の設定は行わない。

SUMMARY：プログラム単位のサマリリスト（適用されているコンパイルオプション

[*1] 詳細は「FORTRAN77,77/SXプログラミング手引書」5 プログラムのデバッグをご覧ください。

を含む）を出力する。

NOSUMMARY：プログラム単位のサマリリストは出力しない。

FLAG：日本工業規格のFORTRANの上位水準に含まれていない項目に対して，メッセージを出力する。

NOFLAG：日本工業規格のFORTRANの上位水準に含まれていない項目に対して，メッセージを出力しない。

INCLIST：ソースプログラムリストを出力する場合，INCLUDE文により組み込まれた文をソースプログラム中に表示する。

NOINCLIST：ソースプログラムリストを出力する場合，INCLUDE文により組み込まれた文はソースプログラムリスト中に表示しない。

NCLIST：コンパイラが出力するリストを日本語で出力する。

STDLIST：コンパイラが出力するリストを英文で出力する。

ENDMARK：ソースプログラムが固定形式で記述されている場合だけ有効で，ソースプログラム上で行領域（第１～72桁）と識別領域（第73～80桁）の間に区切りのマーク（｜の１文字）を挿入する。

NOENDMARK：ソースプログラムリスト上で行領域の直後に識別用領域を表示する。

DOLLAR：プログラム中の文字＄を特殊文字とする。

NODOLLAR：プログラム中の文字＄を英字とする。

LCASE：ソースプログラム中に英小文字が含まれている。英小文字がなくても構わない。プログラム中の英小文字は文字定数の中等を除いて対応する英大文字と同一視される。ソースリスト上で英小文字をそのまま表示したい場合はジョブ投入時のディスポジションでＬを指定しないと大文字に変換されて投入される。

NOLCASE：ソースプログラム中に英小文字が含まれていない。英小文字があるとエラーとなる。

VDIR：ソースプログラム中のベクトル化指示行を有効にする。

NOVDIR：ソースプログラム中のベクトル化指示行を無効にする。

INLOG4：２バイト整数型を標準の整数型，１バイト論理型を標準の論理型とみなして翻訳する。

NOINLOG4：２バイト整数型，１バイト論理型をそのままの型で翻訳する。

XAREA*1=n，XAREA=mM：指定されたnK（キロ）バイトあるいはmM（メガ）バイト以上の記憶域を占める共通ブロック，局所的配列（EQUIVALENCE文により結

---

*1 詳細は「FORTRAN77,77/SXプログラミング手引書」４．１３拡張領域をご覧ください。

合されている場合には，結合された要素全体）が，拡張領域に割り付けられることを指定する。既定値64Ｋ。

COMPAT[*1]：FORTRAN77言語と異なるFORTRAN言語で記述されたソースプログラムを翻訳する。

FIXED：ソースプログラムが固定形式で記述されている。

固定形式とは，文番号領域をカラム１～５，継続行指定領域をカラム６，文領域をカラム７～７２として記述された形式。

FREE：ソースプログラムが形式１の自由形式で記述されている。

形式１の自由形式とは，次のような形で記述されている形式をいう。

- 継続行は，直前の行が注釈行でもデバッグ行でもなく，かつ直前の行の空白でない最後の文字がハイフンである行。
- 注釈行は，継続行ではなく，かつ第１けた目の文字が引用符（”）である行。
- デバッグ行は，第１桁目の文字が星印（＊）である行。
- 開始行は，第１桁目の文字が引用符でも，星印でもなく，かつ，直前の行が注釈行でないなら，直前の行の空白でない最後の文字がハイフンでない場合の現在の行。
- 開始行の先頭が数字の列で始まっている場合にのみ，この数字の列を文番号とみなす。数字の列は５けた以下である。
- 文は，開始行および継続行の任意の桁から始め，任意の桁で終了する。

FREE2：ソースプログラムが形式２の自由形式で記述されている。

形式２の自由形式とは，

- 行の第１桁から第72桁の間に文を記述している。
- 注釈行は，第１桁目の文字がＣである行。
- デバッグ行は，第１桁目の文字が星印である行。
- 継続行は，空白でない最初の文字がアンパサンド（＆）である行。
- 開始行は，注釈行でもデバッグ行でもなく，かつ，空白でない最初の文字がアンパサンドでもない行。開始行の先頭が数字の列で始まっている場合には，この数字の列を文番号とみなす。
- 文は，開始行および，継続行の第１桁から第72桁の任意の桁から始め，任意の桁で終了する。

OBSERVE：すべてのレベルの診断メッセージを出力する。

[*1] 詳細は「FORTRAN77,77/SXプログラミング手引書」４．７他FORTRANとの互換のための機能をご覧ください。

WARNING：警告レベル以上の診断メッセージを出力する。

FATAL：重大レベル以上の診断メッセージを出力する。

CONSTEXT：数値定数の有効桁数による精度拡張を行うか否かを指定する。

- ALL：すべての数値定数に対して行う。
- NOEXP：指数の指定のない数値定数に対して行う。
- NONE：行わない。

AUTODBL[*1]：精度自動拡張機能を働かせるか否かを指定する。既定値はNONE。

MAIN：主プログラムにPROGRAM文がない場合の主プログラム名を指定する。
MAIN=英数字８文字の名前。省略したときの既定値はFTMAIN。

BLKDATA：BLOCK DATA文に名前が指定されていない場合の初期値設定副プログラム名を指定する。BLKDATA=英数字８文字の名前。省略したときの既定値はFTBLKD。

以下のオプションは複数の値から選択して括弧でくくって指定します。

OPT：最適化処理に当たり，以下の制御を指示する。

- DIV：除算の乗算化の最適化を行う。
- NODIV：除算の乗算化の最適化を行わない。
- MOVE：ループ内に不要な計算がある場合，無条件にループ外に移動する。
- NOMOVE：ループ内に不要な計算がある場合，ループ内で選択的に実行される部分にあるものは，ループ外に移動しない。
- CHG：式の評価順序の変更を行う。
- NOCHG：式の評価順序の変更を行わない。

MASK[*2]：演算例外を抑止するか否かを指定する。既定値はZDIV,FLOVF,NOFLUNF,NOFXOVF。

SAMEAREA,NOSAMEAREA[*3]：局所的な配列およびEQUIVALENCEグループをプログラム単位間で共有可能とするか否かを指定する。

INLINE：組み込み関数や乗演算で通常は外部手続きとして翻訳されるものを，指定された関数について，その引用場所にその関数の計算式のコードを埋め込んで翻訳する。

名前が指定されている場合には，指定されている関数だけを埋め込みの対象とし，一方，名前が省略されている場合には，指定できるすべての関数を

[*1] 詳細は「FORTRAN77,77/SXプログラミング手引書」４．４精度自動拡張機能をご覧ください。

[*2] 詳細は「FORTRAN77,77/SXプログラミング手引書」４．２演算例外処理をご覧ください。

[*3] 詳細は「FORTRAN77,77/SXプログラミング手引書」４．１５排他的に使われるデータ領域同士の領域共有化機能をご覧ください。

埋め込みの対象として翻訳する。

指定できる名前：SQRT,DSQRT,SIN,DSIN,COS,DCOS,EXP,DEXP,ALOG,DLOG,
ALOG10,DLOG10,IIPWR,RIPWR,DIPWR,RRPWR,DDPWR

ここでIを整数型，Rを実数型，Dを倍精度実数型とすると，IIPWRはI**I，RIPWRはR**I，DIPWRはD**I，RRPWRはR**R，DDPWRはD**Dのべき乗ルーティンを示す。

NOINLINE：INLINEと逆。名前が指定されている関数を外部手続きとして翻訳し，その他の関数は，その引用場所にその関数の計算式のコードを埋め込んで翻訳する。一方，名前が指定されていない場合には，すべての関数を外部手続きとして翻訳する。名前はINLINEで示したものと同じものが指定できる。

また，関数の計算式のコードを埋め込んで翻訳する場所を次のサブオプションで限定することができる。

- WHOLE：すべての場所にある対象関数を埋め込みの対象とする。
- LOOP：コンパイラが認識したループ中で引用されている対象関数を埋め込みの対象とする。
- IMLOOP：コンパイラが認識したループの内，最内側ループ中で引用されている対象関数だけを埋め込みの対象とする。

また，引数の値の検査を行うか否かを選択できる。

- ERRCHK：引数の値が許される範囲内にあるか否かの検査を行う。
- NOERRCHK：引数の値が許される範囲内にあるか否かの検査を行わない。

VECTOR：ソースプログラム中のベクトル化可能なDOループに対して自動ベクトル化を行う。

NOVECTOR：自動ベクトル化を行わない。

- FULLMSG：ベクトル化診断メッセージとベクトル化情報リストを出力する。
- INFOMSG：ベクトル化診断メッセージのみを出力する。
- NOMSG：ベクトル化診断メッセージとベクトル化情報リストを出力しない。
- LOOPCNT=I：DOループの繰り返し数の最大値を指定する。繰り返し数が翻訳時に不明な場合，繰り返し数をいくつに仮定するかを指定する。ただし，ループ中に使用されている配列の宣言により，繰り返し数の上限が推定できる場合には，Iの値と上限値の小さい方が取られる。Iとして1以上の値を指定する。既定値5000。
- VERRCHK：ループのベクトル化された部分にある組み込み関数の引数値のエラーチェックを行う。
- NOVERRCHK：ループのベクトル化された部分にある組み込み関数の引数値のエ

ラーチェックを行わない。

- VWORKSZ=nM：作業用ベクトル領域の大きさをメガ単位で指定する。既定値１Mバイト。１≦m≦４。コンパイラは作業用ベクトル領域の大きさが，VWORKSZオプションで指定された値を越える場合，そのDOループはベクトル化しない。
- DIVLOOP：ループ分割によるベクトル化を行う。
- NODIVLOOP：ループ分割によるベクトル化を行わない。
- IMLOOP：最深DOループのみをベクトル化の対象とする。
- NOIMLOOP：任意のDOループをベクトル化の対象とする。
- LOOPCHG：ループの入れ替えによるベクトル化を行う。
- NOLOOPCHG：ループの入れ替えによるベクトル化を行わない。
- ASSUME：仮配列あるいは共通ブロック内の配列の宣言サイズをループ長の仮定に用いる。
- NOASSUME：仮配列あるいは共通ブロック内の配列の宣言サイズをループ長の仮定に用いない。
- EXPAND=n：ループ長ｎ以下の最内側ループを展開することを指定する。ｎとして１以上の値を指定する。既定値４。
- NOEXPAND：ループ展開を行わないことを指定する。

③注意事項

- オプションの綴りを誤るとジョブは削除されます。
- マルチコンパイルは10個まで可能ですが，複数のソースファイルを $ PRMFL文で指定するときは，１つ１つのファイルの間に $ FRT77文が必要です。
- ここに記載されているオプションはAP用のオプションです。CPでは使用できないオプションもあります。

正

```
$  FRT77    ｵﾌﾟｼｮﾝ
     ｿｰｽﾌﾟﾛｸﾞﾗﾑ
$  FRT77  ｵﾌﾟｼｮﾝ
$  PRMFL   S*,～
$  FRT77  ｵﾌﾟｼｮﾝ
$  PRMFL   S*,～
$  FRT77  ｵﾌﾟｼｮﾝ
     ｿｰｽﾌﾟﾛｸﾞﾗﾑ
```

誤

```
$  FRT77    ｵﾌﾟｼｮﾝ
     ｿｰｽﾌﾟﾛｸﾞﾗﾑ ﾃﾞｯｸ
$  PRMFL   S*,～
$  PRMFL   S*,～
     ↑
最初のソースデックのみ有効となり
$ PRMFL文以降のソースプログラムは
SX側に転送されません。
```

④例

```
ｶﾗﾑ   1        8        16
      $        FRT77    SOURCE,FIXED
```

ソースプログラムは固定形式で記述されており，ソースリストを出力する。

（4）GO文

利用者プログラムを実行します。

①形式　　1　　8　　16

```
        $     GO        オプション［, オプション］
```

②オペランドフィールド

ASL ：メーカ提供の科学技術計算ライブラリを使用する。

MLIB：メーカ提供の数値計算ライブラリを使用する。

XLIB：図形処理ライブラリを使用する。

③注意事項

- １つのジョブで $ GO文は１枚しか利用できません。複数の$ GO文が挿入されていても２枚目以降は無視されます。
- 結合処理と実行の２つのステップに展開されます。
- 一度に複数のライブラリを使用できます。

④例

カラム　1　　8　　16

```
$     GO      ASL,MLIB
```

メーカ提供の科学技術計算ライブラリおよび数値計算ライブラリを使用。

（5）PRMFL文

実行時に使用するパーマネントファイル，あるいは翻訳時に使用するソースプログラムファイルを割り当てます。

①形式　　1　　8　　16

```
        $     PRMFL     FC，割り当てタイプ，アクセスモード，
                        修飾ファイル名，［初期サイズ］，
                        ［拡張サイズ］，［レコードサイズ］，
                        ［転送変換形式］
```

②オペランドフィールド

FC（ファイルコード）：プログラム中で使用しているファイル（FORTRANでは外部装置識別子）の２桁の英数字を指定する。

ソースプログラム：S＊

実行データ入力時：01≦FC≦99，但し，06，07は使用できません。

実行データ出力時：01≦FC≦99，但し，05は使用できません。

割り当てタイプ：ファイルに対する入出力動作を指定する。

Ｒ：読み込み，ジョブ終了時SXからACOSへの逆転送は行われません。

Ｗ：読み込み／書き出し，ジョブ終了時SXからACOSへの逆転送が行われます。

アクセスモード：ファイルの形式を指定する。

Ｓ：順編成ファイル，簡易形では順編成ファイルしか使用できません。

修飾ファイル名：使用するパーマネントファイルを指定する。

初期サイズ：SX側のディスクに確保するファイルの最初の大きさをリンク単位で指定する。省略可。省略時15リンク。

拡張サイズ：SX側のディスクに確保するファイルの１回の増分値をリンク単位で指定する。省略可。省略時15リンク。拡張は15回まで行われる。

レコードサイズ：ファイルへの最大入出力論理レコードサイズを指定する。省略可。省略時508バイト。１レコード509バイト以上の時指定する必要がある。最大は32731バイトです。

転送変換形式：転送ファイルのコード変換の形式を指定する。

FORM：順編成ファイル書式付きデータを標準のコード変換を行い転送／逆転送する。規定値。

NFORM8：順編成ファイルの32ビットバイナリデータ（SXでの書式無し入出力文のデータ）をそのままコード変換せずに転送／逆転送する。

③注意事項

- ファイルの形式は順編成ファイルしか使用できません。
- SX側のファイル容量は既定値でほぼ十分の大きさを確保していますが，容量不足のエラーがでた場合は，第5，6オプションを使用し拡張してください。
- データ書き込みのあるファイルに割り当てタイプを誤って読み込みのみの指定をした場合，実行は正常に終了してもACOSへのファイルの逆転送は行われません。
- SX側のディスク容量を十分にとり，実行ステップが正常に終了しSXからACOSに逆転送が行われたとき，ACOSのファイル容量が小さいとデータは途中までしか書かれません。
- ACOSのファイル構造概略

ACOSのファイル管理システム（ＦＭＳ：ﾌｧｲﾙﾏﾈｰｼﾞﾒﾝﾄｽｰﾊﾟｰﾊﾞｲｻﾞ）はカタログという管理情報子をノードとする木構造の最終位置にファイルを位置づけ，ファイルを体系的に管理しています。各利用者のカタログ構造は階層構造をなしており，ＵＭＣ（利用者マスタカタログ：利用者番号）から数えて９レベル以内でなければなりません。ＵＭＣとは，利用者カタログ構造の最高位に位置するカタログであり，利用者番号になります。ファイル記述とは，カタログ構造内の各枝の

末端に位置するカタログであり，ファイルと一対一に対応します。サブカタログとは，ＵＭＣと各ファイル記述の中間に位置するカタログをさします。修飾カタログ名，修飾ファイル名は別名カタログストリング，ファイルストリングと呼ばれることもあります。修飾名とはＵＭＣから対応するカタログまたはファイルまでのカタログ構造内の枝のノード名をスラッシュ（／）で区切り連ねた名前であり，同一名を持つ他のカタログまたはファイルとを区別します。

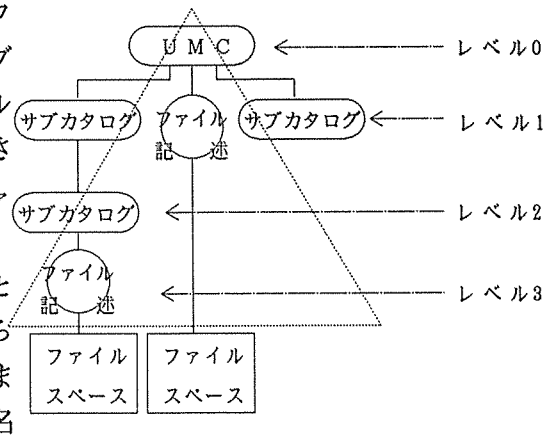


図６　ファイル構造

- ソースプログラムあるいはデータをこの $ PRMFL文で割り当てたとき，これらのプログラムあるいはデータが転送されるのはジョブが転送されるときではなく，転送されたジョブがSXで実行に入り転送用のステップが実行に入ったときです。繁忙期になると，ジョブのターンアラウンドが延びこの転送されるタイミングが非常にズレます。データファイルのアクセスなどにはご注意ください。転送用のステップが実行に入ったとき，ACOS側で使用していると転送ステップが異常終了します。また，タブ文字やカード形式の編集の対象にはなりませんのでご注意ください。
- ソースプログラムファイル転送時，ファイルの最初のレコードの先頭が数字であると，行番号付きファイルであるとみなします。行番号なしのファイルでも文番号が最初のレコードに存在すると，行番号有りとみなし行番号有りの処理を行いますのでファイル転送が正常に実行[*1]できません（ジョブ投入時のカード形式とは無関係に）。
- 実行データファイル転送時，行番号有りの処理を行いませんので，行番号付きファイルでも行番号をデータとみてそのまま転送します（ジョブ投入時のカード形式とは無関係に）。

④例

```
ｶﾗﾑ   1       8       16
      $       FRT77   SOURCE
                ≀
              WRITE(31,1000)A,B,C
                ≀
```

[*1] レポートコード７７に，STAGE ABNORMAL END エラーメッセージが出力されます。

```
$        GO
$        PRMFL   31,W,S,W60000/DATA3,,,256
```

利用者番号W60000のファイルDATA3にレコード長256バイトで外部装置識別子31のデータを書き込む。

```
カラム 1      8       16
             WRITE(10)I,J,R,D
               ~
$        GO
$        PRMFL   10,W,S,W60000/SXDATA,,,,NFORM8
$        ENDJOB
```

利用者番号W60000/SXDATAファイルにSX側の32ビットバイナリデータ（書式無し入出力文）をコード変換せずにそのまま転送／逆転送する。ACOS側ではこのファイルは読み書きできません。

（6）ＦＩＬＥ文

プログラム中で使用しているﾌｧｲﾙを一時ファイルに割り当てます。一時ファイルは SX 側に作成されますので，順編成，直接編成のファイル形式が利用でき，プログラム中では書式付き・書式無し記録の入出力ができます。

①形式

```
1      8             16
$     FILE        FC,,［初期サイズとアクセスモード］，
                  ［拡張サイズ］，［レコードサイズ］
```

②オペランドフィールド

ＦＣ（ファイルコード）：プログラム中で使用しているファイル（FORTRANでは外部装置識別子）の２桁の英数字を指定する。01≦ＦＣ≦99，但し05，06，07は使用できません。

初期サイズとアクセスモード：SX側のディスクに確保するファイルの最初の大きさをリンク単位で指定する。続いてファイルの形式，Ｌ（順編成）かＲ（直接編成）の何れかを指定する。省略可。省略時１リンクの順編成ファイル。

拡張サイズ：SX側のディスクに確保する順編成ファイルの１回の増分値をリンク単位で指定する。省略可。省略時15リンク。拡張は15回まで行われる。直接編成の時は無視される。

レコードサイズ：ファイルへの最大入出力論理レコードサイズを指定する。省略可。省略時508バイト。１レコード509バイト以上の時指定する必要がある。最大は32731バイトです。

③例

```
ｶﾗﾑ   1        8        16
      $        FRT77    SOURCE
                 ≀
               WRITE(11,1000)A,B,C,D,E,F,G
                 ≀
      $        GO
      $        FILE     11,,15L,,1024
```

初期値15ﾘﾝｸの順編成一時ファイルにレコード長1024バイトで外部装置識別子11のデータを書き込む。

（7）ＳＹＳＯＵＴ文

プログラム中で使用している外部装置識別子６以外のファイルを標準SYSOUT[*1]ファイルに割り当てます。

①形式

```
      1      8              16
      $$     ＳＹＳＯＵＴ   ＦＣ
```

②オペランドフィールド

ＦＣ（ファイルコード）：プログラム中で使用しているファイル（FORTRANでは外部装置識別子）の２桁の英数字を指定する。01≦ＦＣ≦63

③注意事項

- 外部装置識別子06，07に関してはこの文は不要です。
- １ジョブ内で11枚まで使用できます。

④例

```
ｶﾗﾑ   1        8        16
      $        FRT77    SOURCE
                 ≀
               WRITE(11,1000)A,B,C
               WRITE(21,2000)H,I,J
                 ≀
      $        GO
      $        SYSOUT   11
      $        SYSOUT   21
```

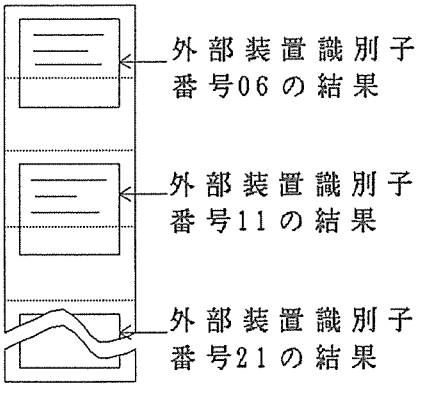


図７　標準ＳＹＳＯＵＴ出力形式

外部装置識別子11，21のデータを標準SYSOUTファイルに割り当てる。

[*1] プリンタに出力するデータで，レコードサイズは133文字に制限され，最初の１文字目が印字時の行送り文字として扱われる。標準では外部装置識別子６を持つWRITE文

（8）SELECTA文

複数の入力ファイルを１つに併合してバッチ処理システムに引き渡します。

①形式　　1　　8　　16

$　　SELECTA　ファイル記述

②オペランドフィールド

ファイル記述：使用するパーマネントファイルを指定する。

③注意事項

- CARDINサブシステムで投入するときのみ使用できます。
- CARDINサブシステムは $ SELECTA 文を見つけると，指定されたファイルの内容を呼び出し $ SELECTA 文はバッチシステムには引き渡しません。
- $ PRMFL 文と異なり，CARDINサブシステムがジョブ投入時にファイルの内容を展開しますので，ファイル転送用のステップはありません。
- RUNコマンド時のカード様式の影響を受けますので注意してください。

④例

```
*  CARD O /SXTEST1                              ····ⓐ
*  LIST                                         ····ⓑ
$IJOBI;A,U,,,,JPA4
$IFRT77ISOURCE
$ISELECTAIW60000/SOURCE1
$ISELECTAIW60000/SOURCE2
$IGOIASL,MLIB
$IENDJOB
*  RUN                                          ····ⓒ
CRJE001 R カード形式とディスポジション？A       ····ⓓ
CRJE002 R タブ文字と位置？1,8,16                ····ⓔ
CRJE350 I    SNUMB名  #  0131T
*
```

ⓐCARDINサブシステムで利用者番号直下のファイルSXTEST1を呼び出しています。

ⓑ呼び出したファイルの内容を表示しています。

ⓒジョブをバッチシステムに投入しています。

ⓓカード形式を入力

ⓔタブ文字を入力

3.5 ジョブ構成

（1）ソースプログラム・データファイルの使用

①コンパイルと実行

最も基本的なジョブ構成でコンパイルと実行を行っています。

```
0010$!JOB!;A,U,,,,JPA4                                  ……  ⓐ
0020$!SX
0030$!FRT77!SOURCE                                      ……  ⓑ
0040       read(5,10)i,j
0050    10 format(2i2)
0060       k=i+j
0070       l=i-j
0080       write(6,20)i,j,k,l
0090    20 format(1h ,4i10)
0100       stop
0110       end
0120$!GO
0130 8 4
0140$!ENDJOB
*run                                                    ……  ⓒ
CRJE001 R カード形式とディスポジション?s,l              ……  ⓓ
CRJE002 R タブ文字と位置?!,8,16                        ……  ⓔ
CRJE350 I   ＳＮＵＭＢ名 ＃ 0643T
*jmon 0643T                                            ……  ⓕ
CRJE452 I 0643T はジョブ転送中です
          ≀
CRJE447 I デマンドファイル内の 0643T は出力待ちです @ 11:02:57
CRJE600 I NORMAL TERMINATION
*
```

ⓐ支払コードA，ジョブクラスU，A4サイズのプリンタを指定

ⓑソースリストを出力しています。

ⓒCARDINサブシステムでジョブを投入

ⓓ行番号を取り，小文字をそのまま変換しないでバッチシステムに引き渡しています。

ⓔタブ文字として！を使用しています。

ⓕジョブの状態を表示

②ソースプログラムとデータをファイルから

ソースプログラムとデータをファイルから入力しています。

```
0001##S,L !,8,16                                    ……  ⓐ
0010$!JOB!;A,U
0020$!SX
0030$!FRT77!SOURCE
0040$!SELECTA!W60000/SX/TESTSOU                     ……  ⓑ
0050$!GO
0060$!PRMFL!05,R,S,W60000/SX/TESTDATA               ……  ⓒ
0070$!ENDJOB
*RUN
CRJE350 I   ＳＮＵＭＢ名 ＃ 0680T
*JMON 0680T
      ≀
CRJE600 I NORMAL TERMINATION
*JOUT 0680T LIST ALL
ACTI#  RC    LINE   HOLD  CLASS
  --   $$     --     --     I
 001   74      24    YES    I
 002   74      28    YES    I
 003   77       1    YES    I
CRJE012 R ファンクション?ACTI 3 PRIN 77             ……  ⓓ

11:44:52 STAGE ABNORMAL END. (A6)FILE BUSY          ……  ⓔ
CRJE317 I レポート 77 の終了です
CRJE012 R ファンクション?RELE                       ……  ⓕ
*REMO /SX/TESTDATA                                  ……  ⓖ
*RUN                                                ……  ⓗ
CRJE350 I   ＳＮＵＭＢ名 ＃ 0755T
*
```

ⓐCARDINサブシステムのための編集情報（行番号削除等）を入れてあります。

ⓑソースプログラムの格納されているファイルを$ SELECTA 文で指定しています。このファイルの内容はRUNコマンドでバッチシステムに引き渡すときに展開されます。ファイル/SX/TESTSOUには次のように格納されています。

カラム1

```
      0010       read(5,10)i,j
      0020    10 format(2i2)
      0030       k=i+j
      0040       l=i-j
      0050       write(6,20)i,j,k,l
      0060    20 format(1h ,4i10)
      0070       stop
      0080       end
```

この例では行番号がないと文番号が行番号とみなされコンパイルエラーとなります。正しい編集情報をこのファイルの先頭に入れるか，$ SELECTA文を$ PRMFL文に変更すると正常に実行します。但し，$ PRMFL 文の場合編集情報があるとコンパイルエラーとなります。

$ SELECTA 文の場合

正

```
1
0010      read(5,10)i,j
0020 10   format(2i2)
0030      k=i+j
0040      l=i-j
0050      write(6,20)i,j,k,l
0060 20   format(1h ,4i10)
0070      stop
0080      end
```

誤

```
1
          read(5,10)i,j
     10   format(2i2)
          k=i+j
          l=i-j
          write(6,20)i,j,k,l
     20   format(1h ,4i10)
          stop
          end
```

正

```
1
0010##s !,8,16
0020      read(5,10)i,j
0030 10   format(2i2)
0040      k=i+j
0050      l=i-j
0060      WRITE(6,20)I,J,K,L
0070 20   format(1h ,i10)
0080      stop
0090      end
```

$ PRMFL 文の場合

正

```
1
0010      read(5,10)i,j
0020 10   format(2i2)
0030      k=i+j
0040      l=i-j
0050      write(6,20)i,j,k,l
0060 20   format(1h ,4i10)
0070      stop
0080      end
```

正

```
1
          read(5,10)i,j
     10   format(2i2)
          k=i+j
          l=i-j
          write(6,20)i,j,k,l
     20   format(1h ,4i10)
          stop
          end
```

誤

```
1
0010##s !,8,16
0020      read(5,10)i,j
0030 10   format(2i2)
0040      k=i+j
0050      l=i-j
0060      WRITE(6,20)I,J,K,L
0070 20   format(1h ,i10)
0080      stop
0090      end
```

ⓒデータの格納されているファイルを $ PRMFL文で指定しています。バッチシステムに引き渡すときにファイル転送用のステップに展開されます。編集情報の対象にはなりません。ファイル/SX/TESTDATAには次のように格納されています。

```
カラム 1
␣8␣4
```

ⓓアクティビティ３のレポート77の内容を出力している。

ⓔファイル転送ステップ実行時，TSSでこのファイルが使用中であったためSXに転送できませんでした。TSSのOLDやLISTコマンド等でファイルをアクセスすると使用中の状態のままです。

ⓕ出力結果を削除しています。

ⓖTSSのREMOコマンドで使用中のファイルをクローズし利用状態を解除しています。

ⓗ再度ジョブを投入しています。

（２）一時ファイルの使用

外部装置識別子10を直接探査形式で一時ファイルに割り当てて実行しています。

```
$!JOB!;A,U
$!SX
$!FRT77!SOURCE
      OPEN(10,ACCESS='DIRECT')                    ……  ⓐ
      READ(5,10)I,J
   10 FORMAT(2I2)
      DO 100 I=1,10
           ≀
  100 WRITE(10,IREC=I)I,J,K
           ≀
      STOP
      END
$!GO
 8 4
           ≀
$!FILE!10,,10R                                    ……  ⓑ
$!ENDJOB

*RUN
CRJE001 R カード形式とディスポジション?A
CRJE002 R タブ文字と位置?!,8,16
CRJE350 I    ＳＮＵＭＢ名 ＃ 1362T
*
```

ⓐ外部装置識別子10のファイルに直接探査形式を指定しています。

ⓑ外部装置識別子10を大きさ10リンクの直接編成一時ファイルに割り当てています。

３.６　使用上の注意

①FORTRAN プログラムの翻訳は必須であり，実行のみのジョブは削除されます。

②プログラムの実行は翻訳直後に一度だけ有効です。

③SXで処理されたジョブが出力記録数オーバーで異常終了した場合，ファイルの逆転送機能は働きません。

④SXファイルの拡張について，SXディスク上に指定された拡張サイズの空き領域が存在しないとき，拡張サイズが３リンクとなり指定された大きさのファイルが確保されない場合があります。

４．　おわりに

SX簡易形で使用できるジョブ制御言語はここに説明したものだけではありません。この他にももう少しありますが，ここに説明したものだけで一応はSXを利用できると思われます。ただ，ANALYZERの使い方についてはいっさい説明してありません。日本電気説明書「MSF-6利用の手引き」及び「ANALYZER/SX説明書」をご覧ください。

なお，本稿をまとめるにあたり関西日本電気ソフトウェア株式会社久宗利恵氏に大変お世話になりました。紙面をかりてお礼申し上げます。

参考文献

GJF11-4 MSF-6利用の手引
GGB11-1 FORTRAN77言語説明書
GGB12-5 FORTRAN,77/SXプログラミング手引書
GGB14-5 ANALYZER/SX説明書