プログラム・ノート

オーバーレイ構造のプログラム
—主記憶装置の有効利用のために—

研究開発室　磯　本　征　雄

1．はじめに

フォートラン・プログラムが計算機により実行可能であるためには，そのプログラムが実行時に必要とする主記憶装置の大きさを，計算機の主記憶装置の大きさより小さくしておかなければならない。ところが現実にはプログラムが必要とする主記憶装置の大きさが実際に使用される計算機の主記憶装置の大きさを超える事がしばしばある。このような場合，周辺機器（ディスク・パック，磁気テープ等）を用いるとか，その他の工夫によりプログラム中の変数の数を少なくする事がよくおこなわれている。

一方，フォートラン・プログラム自体についていえば，プログラム全体が常時主記憶装置上になければならないということはほとんどない。むしろある瞬間に実行されているのはプログラムの内の一部分であるのが普通である。従って実際に実行されるプログラムの部分のみを主記憶装置上にロード(Load) し，実行されていない部分は主記憶装置からはずしておくことができる。そしてこの主記憶装置上のプログラムの実行が完了した時点で，次に実行すべきプログラム部分を主記憶装置上にロードすればよい。この時，新しくロードされるプログラムは，旧いプログラムの上からオーバーレイ(Over Lay) すれば良い。この手続きを順次くりかえすならば，原理的に限られた大きさの主記憶装置によって無限に大きなプログラムを実行する事が可能となる。このようにフォートラン実行プログラムの実行時に必要とする主記憶装置の大きさを少なくする方法として，「オーバーレイ構造」のプログラムにする手法がある。

ここでは，フォートラン 700 システムにおけるオーバーレイ構造のプログラム及びその手法について説明する。

2．プログラムの構造

2.1　フェーズ，チェイン及びノード名

一般にプログラムは1つの主プログラムといくつかのサブプログラムから構成されている。これらの主プログラム及びサブプログラム群を主記憶装置上にロードする時のロードの状態を「プログラムの構造」という。プログラムの構造を記述するための言語に「フェーズ(phase)」，「チェイン(chain)」及び「ノード名(node name)」がある。

図1は，フェーズ，チェイン及びノード名を説明するためにかかれたプログラムの構造の例である。太い縦線はプログラム部分であり上から下へ主記憶装置上での番地の小さなものから

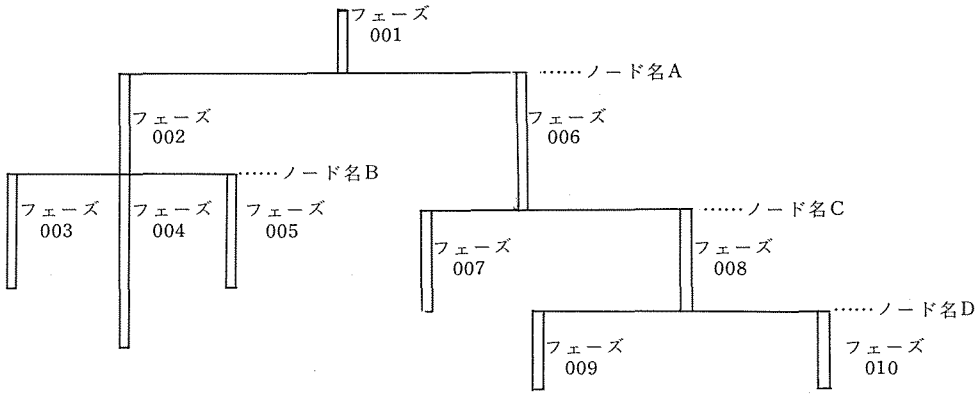


図 1

大きなものへと相対的位置を示している。細い横線は相対的番地の同一部分を示すものである。

フェーズとは，実行プログラムが実行されるときに制御プログラムによって主記憶装置内に読み込まれる（ロードされる）プログラムの単位をいう。すなわち主記憶装置内への入出力は常にフェーズ単位でおこなわれる。したがってオーバーレイ構造をもたないプログラムは1つのフェーズより成っている。図1は多重フェーズのプログラムの例である。実行プログラムがいくつのフェーズから構成されているかは，その実行プログラムをリンクロードするときに用いられたリンケージ・ローダーへの制御カードにより決められる。このフェーズを決める制御カードのことをフェーズ・カードと呼びます。一般に，フェーズの個数は，実行プログラムをリンクロードするときに使用したPHASEカードの数に等しい。図1の例にみるように，フェーズ番号が001のフェーズをルート・フェーズ(root phase) といい，ルートフェーズには必ず1つの主プログラムがなければならない。以下順次フェーズごとにフェーズ番号がフェーズ・カードにより3桁の数字で与えられる。

ノード名とは，主記憶装置内にロードされたフェーズ相互間の相対的な位置関係を示すものである。図1において細い横線はそれぞれノードでありこれにつけた名前がノード名である。例えば図1において1本の横線から下方向へ伸びた太い縦線は，それらのフェーズがいずれも主記憶装置内で同一の番地からはじまって高位番地に向ってロードされる事を示している。したがって同一ノード名をもつフェーズは主記憶装置内に同時に存在することはできない。このようにノード名は，同一のノード名をもつフェーズが実行時に同一番地から主記憶装置内に格納されるようにリンケージ・ローダに指示するためのものである。当然各フェーズでの主記憶装置内での格納開始番地は原則としてノード名を使って指定される。但しノード名は6文字以内の英数字により与えられる。

チェインとは，ルート・フェーズから最下端のフェーズ（自分の下にさらに連らなるフェーズをもたないフェーズ）迄を主記憶装置内で直列に結んだ一連のフェーズの集りを言う。例えば図1の場合は次のようなチェインより成っている。

ルート・フェーズ―フェーズ002―フェーズ003

ルート・フェーズ―フェーズ002―フェーズ004

ルート・フェーズ―フェーズ002―フェーズ005

ルート・フェーズ―フェーズ006―フェーズ007

ルート・フェーズ―フェーズ006―フェーズ008―フェーズ009

ルート・フェーズ―フェーズ006―フェーズ008―フェーズ010

一つのチェインに属する全フェーズは主記憶装置上に同時に存在する。この結果フェーズ相互の間の引用について次の事がいえる。

1）ルート・フェーズ内にある主プログラムあるいは手続き副プログラムからは，どのフェーズ内にある手続き副プログラムをも引用できる。

2）ルート・フェーズ以外のフェーズ内にある手続き副プログラムからは，そのフェーズが属するチェイン内のフェーズに含まれる手続き副プログラムを引用できる。

3）あるフェーズ内のプログラム単位からはそのフェーズの下に連らなっているいかなるフェーズ内の外部手続きをも引用できる。

このようなフェーズ，チェイン及びノード名により制御されるプログラム（例えば図1）を「樹状構造のプログラム」ともいう。これら制御カードについての説明を次節でおこなう。

2.2 多重フェーズ・プログラムのリンクロードと実行

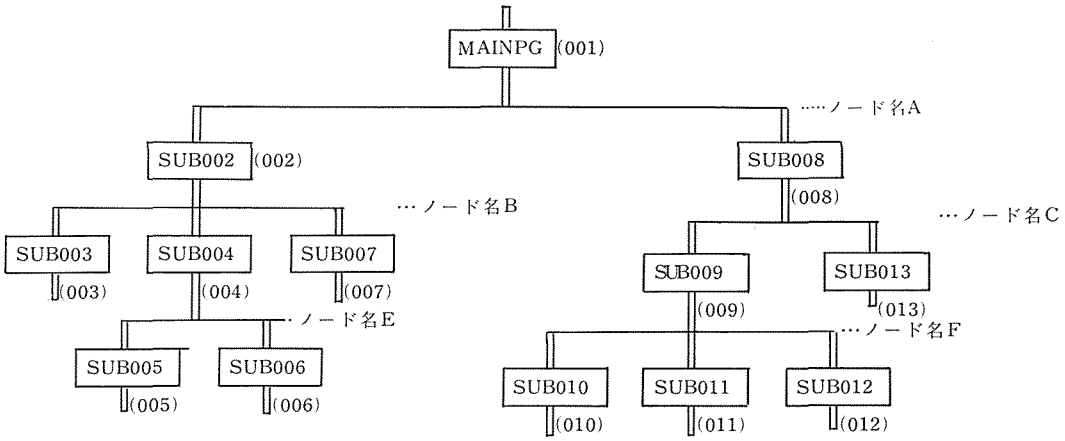
複数のフェーズから成るプログラムを多重フェーズ・プログラムという。多重フェーズ・プログラムを実行するには，リンクロード時に制御カードによってプログラムの構造を定めなければならない。図2には，多重フェーズ・プログラムとその制御カードの例を示した。但し図中の（　）でかこまれた数字はフェーズ番号を示す。また[　]でかこまれた部分は，このフェーズにつけられた名前である。図2のような構成になったプログラムを樹状構造のプログラムという。

樹状構造をもつ多重フェーズ・プログラムのリンク・ロードは，各々のチェインごとにおこなわなければならない。そして制御カードの順序はチェイン内のルートフェーズからはじめて，順次樹状構造の末端に向ってすすめる。次のチェインのロードへ移る際には，樹状構造の内の重複する部分のフェーズは改めてロードしなおす必要はなく，直前にロードされたチェインからの分岐点（ノード）以下のフェーズ群を順次樹状構造の末端に向ってロードすればよい。図2の場合，まずルート・フェーズ(MAINPG)がロードされる。続いてノードAで2つの枝に分かれるがここではフェーズ番号002(SUB002)がロードされている。その下のノードBにおいて3つの枝に分かれている。ここでは，まず左端のフェーズ003(SUB003)がロードされている。これだけの制御カードにより，

MAINPG(001)―SUB002(002)―SUB003(003)

なるチェインのリンクロードが完了する。次のチェインは前のチェインとノードBのところで枝分れしている。したがってノードBより上の部分は改めてロードしなおす必要はなく，ノー

図2



（　）内：フェーズ番号

□内：プログラムにつけたTITLE又はサブプログラム名

制御カード

COL9	16 21
	PHASEEXAMPLE
	CALLMAINPG
A	PHASE
	CALL SUB002
B	PHASE
	CALL SUB003
B	PHASE
	CALL SUB004
E	PHASE
	CALL SUB005
E	PHASE
	CALL SUB006
B	PHASE
	CALL SUB007 ↑

COL9	16 21
↱ A	PHASE
	CALL SUB008
C	PHASE
	CALL SUB009
F	PHASE
	CALL SUB010
F	PHASE
	CALL SUB011
F	PHASE
	CALL SUB012
C	PHASE
	CALL SUB013

ドB以下のフェーズをロードすればよい。ノードB以下のフェーズ番号004(SUB004)をまずロードし，ノードEの所では再び2つの枝に分れているが，ここではフェーズ番号005(SUB005)をロードして一つのチェインのロードを完了し，つづいて次のチェインのロードに移るのだが，今の場合はフェーズ番号006(SUB006)をロードするのみで第3番目のチェインのロードは終る。なぜならば，今の場合は，フェーズ006に関しては，ノードEより上にあるフェーズがすべて直前にロードされたチェインと重複しているからである。この事は，ノードBの所での分岐の状態と取扱いの上では原理的に同じである。以下ロードの手順についてはこのような方法をくりかえしてゆけば良い。

フェーズ及びノードを定める制御カードは

```
6COL.          16          21
MON$$ EXEQ LINKLOAD,,,,,,,,EXTEND
```

の制御カードの次に入れる。ノード名は各々9カラム目"PHASE"は16カラム目からパンチされる(図2参照)。PHASEカードの次のカードには，そのフェーズに相当するプログラム名が16カラム目よりパンチされる。

ルート・フェーズに対するフェーズ番号は必ず001でなければならない。ルート・フェーズ以外のフェーズのフェーズ番号は，直前にロードされたフェーズのフェーズ番号より1だけ大きい数字が自動的にわりあてられる。但しPHASEカード上にフェーズ番号を指定することにより任意のフェーズ番号を割当てることもできる。この場合には，後からリンクロードされるフェーズは直前にリンクロードされたフェーズより必ず大きいフェーズ番号をもたなければならない。

フェーズ番号は001から997までの範囲内の任意の3桁の10進数を使用できる。そして1つの実行プログラムは最大213個のフェーズに分割できる。但し磁気テープをJOBファイルとして使用する場合には，997個までのフェーズに分割してもよい。但しこれらフェーズ番号はプログラムの実行の制御の渡る順序には何ら影響しない。

以上は，多重フェーズ・プログラムを樹状構造のプログラムに組合せる場合のリンケージ・ローダーへの制御カードの説明であった。これら樹状構造のプログラムを実際にどのように実行するかは，フォートラン・プログラムの内容により決められる。これらについては次節で説明する。

3．オーバーレイの手法

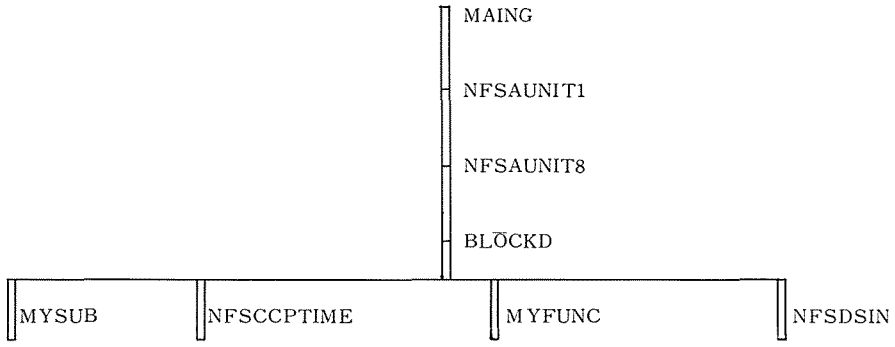
ここで言うオーバーレイとは，主記憶装置上にあるフェーズに対して，次にロードされるフェーズが前者のフェーズと同じノード名をもつ場合，旧いフェーズの上から重なって旧いフェーズをこわしてロードされることをいう。この手法としてはMODEL 700においては「自動オーバーレイ」，サブルーチン「PHASEL」と「PHASER」及びサブルーチン「CHAIN」によるものの3つの手法がある。

以下これら個々のものについてそれらの特徴と手法を説明する。

3.1 自動オーバーレイ

自動オーバーレイとは，リンクロード時に制御カードの指定により，オーバーレイされる各々のサブプログラムに対してフェーズ番号及びノード名を指定しておけば，プログラムの実行時に引用された外部手続きが必要に応じて自動的に主記憶装置上に次々とロードされてゆく手法である。したがってフォートラン・プログラムの中では，外部手続きに対する引用文以外には特に実行及び制御の手順を明示する必要はない。したがってフォートラン・ソース・プログラムとしては，1つのフェーズより成るプログラムの場合と何ら変らない。この点は，次に説明する「PHASELとPHASER」及び「CHAIN」の場合との大きな違いである。但しフェー

図3



制御カード

```
                COL6            9           16      21
①               MON $$                      EXEQ LINKLOAD,,,,,,,EXTEND
②                                           PHASEEXAMPLE
③                                           SCALLMYSUB
④                                           SCALLMYFUNC
⑤                                           SCALLNFSCCPTIME
⑥                                           SCALLNFSDSIN
⑦                                           CALL NFSAUTOVLY
⑧                                           CALL MAINPG
⑨                                           CALL NFSAUNIT1
⑩                                           CALL NFSAUNIT8
⑪                                           CALL BLOCKD
⑫                           NODE1           PHASE
⑬                                           CALL MYSUB
⑭                           NODE1           PHASE
⑮                                           CALL NFSCCPTIME
⑯                           NODE1           PHASE
⑰                                           CALL MYFUNC
⑱                           NODE1           PHASE
⑲                                           CALL NFSDSIN
```

ズ番号及びノード名の指定については前節に説明したとおりである。

以下「自動オーバーレイ」について例題によって説明する。図3には比較的単純な，多重フェーズ・プログラムの例を示した。自動オーバーレイの手法に対する制御カードの特徴は，主としてルート・フェーズの部分にある。図3に示されたプログラムは，ノード名NODE 1の下にフェーズ4個からなっている。サブプログラムMYSUB, NFSCCPTIME, MYFUNC及びNFSDSINが各々主記憶装置上の同じ番地にオーバーレイされる。遂次カードの説明をおこな

う中で，各々の用語の説明もおこなうことにする。

カード①; リンクロードを命令するシステム制御カードである。自動オーバーレイの手法においては，EXTEND オプションを必ず使用しなければならない。

カード②; 以下のカードは，次のフェーズカードがあらわれるまですべてルート・フェーズに関するものであることを示す。但し EXAMPLE は，ジョブ全体につけられた名前である。

カード③; SCALLMYSUB によりサブルーチン MYSUB のリンクロードを一時保留する。サブルーチン MYSUB については後にフェーズ・カードによりフェーズ番号が与えられる。SCALL はこのようにサブプログラムのリンク・ロードを一時保留する機能をもつ。この制御カードによりルート・フェーズの中からサブルーチン MYSUB が除外される。但し SCALL は必ずルートフェーズカードの後になければならない。

カード④; SCALL の機能は③と同じ。関数 MYFUNC のリンクロードは一時保留され，ルート・フェーズの中から除外される。

カード⑤; SCALL の機能は③と同じ。サブルーチン CPTIME のリンクロードは一時保留され，ルート・フェーズの中から除外される。NFSCCPTIME は基本外部サブルーチン CPTIME のことである。頭の部分“NFSC” については NEAC シリーズ 2200 FORTRAN プログラミング説明書の表３－１を参照して下さい。

カード⑥; SCALL の機能は③と同じ。関数 SIN のリンクロードは，一時保留され，ルート・フェーズの中から除外される。NFSDSIN は基本外部関数 SIN のことである。頭の部分“NFSD” については NEAC シリーズ 2200 FORTRAN 700 プログラミング説明書の表３－１を参照して下さい。

カード⑦; このカードによりオーバーレイ・コントローラー NFSAUTOVLY がリンク・ロードされる。NFSAUTOVLY は，実行時の自動オーバーレイを制御するプログラムである。このプログラムは必ずルート・フェーズで CALL しなければならない。CALL は，このようにサブプログラムをフェーズの中にリンクする機能をもつ。

カード⑧; このカードにより，フォートラン・ソース・プログラムの内の主プログラム (MAINPG) をルート・フェーズの中にリンクする。

カード⑨; このカードにより，ファイル・テーブル・サブプログラム (NFSAUNIT1) をルート・フェーズとしてリンクロードする。ファイル・テーブル・サブプログラムは必ず，ルート・フェーズで CALL しなければならない。

カード⑩; このカードにより，ファイル・テーブル・サブプログラム (NFSAUNIT8) をルート・フェーズの中にリンクする。ファイル・テーブル・サブプログラムをルート・フェーズで CALL しなければならない事はカード⑨の場合と同じである。

カード⑪; ブロック・データ・サブプログラムBLOCKDをリンクロードする。

ここまでの制御カードは，ルート・フェーズに関するものである。以下では，ノード名NODE 1の下にぶらさがった，相互にオーバーレイされるフェーズについての制御カードである。

カード⑫; ノード名NODE1とPHASEを指定する。今の場合はフェーズ番号が002となる。

カード⑬; カード⑫にひきつづいて，ノード名NODE1でフェーズ番号002としてサブルーチンMYSUBをリンクロードする事を示す。CALLについては，カード⑦の場合と同じ機能である。ルート・フェーズでリンク・ロードを保留されていたサブルーチンMYSUBはこのカードによりリンク・ロードされるのである。

カード⑭⑮; サブルーチンCPTIMEが，ノード名NODE 1でフェーズ番号003としてリンクロードされる。

カード⑯⑰; 関数MYFUNCが，ノード名NODE 1でフェーズ番号004としてリンクロードされる。

カード⑱⑲; 組込み関数SINが，ノード名NODE 1でフェーズ番号005としてリンクロードされる。

これら制御カードにより，サブルーチンMYSUB，CPTIME及び関数MYFUNC，SIN等が各々MAINPGからの引用に応じて適宜，主記憶装置上へオーバーレイされる。ここでは，最も単純な例題を示したが，さらに複雑な樹状構造のプログラムについても，前節の要領に従って制御カードをつくれば良い。

多重エントリー及び共通ブロックについては自動オーバーレイの際にはいくつかの注意しなければならない点がある。

多重エントリをもつ手続き副プログラムの自動オーバーレイ

多重エントリをもつ手続き副プログラムとは，ENTRY文によって複数個の入口をつくられた手続き副プログラムである(FORTRAN 700 プログラミング説明書参照)。

多重エントリをもつ手続き副プログラムをオーバーレイ・フェーズに置くためには，その副プログラム内にあるすべての手続き名をルートフェーズのリンクロード時にSCALLカードで指定しておかなければならない。後にフェーズでその手続き副プログラムをCALLカードによりリンクロードする時には，SUBROUTINE文あるいはFUNCTION文によって定義された手続き名のみを指定すればよい。

わかりやすくするために次のような例を示す。

（例）

```
主プログラム
      ≀
CALL SUBO(A, B)
      ≀
P(I)=FUNCO(X, Y)
      ≀
CALL SUB1(A)
      ≀
P(I)=FUNC1(Y)
      ≀
STOP
END
```

引用される外部手続き

```
SUBROUTINE SUBO(A, B)
      ≀
ENTRY SUB1(A)
      ≀
RETURN
END
```

```
REAL FUNCTION FUNCO(X, Y)
      ≀
ENTRY FUNC1(Y)
      ≀
RETURN
END
```

これらフォートラン・プログラムを実行するのに次の様な2種類の方法が可能である。

1）1つのフェーズからなる実行プログラムをつくる場合。

```
MON$$   EXEQ LINKLOAD,,,,,,,EXTEND
        PHASEEXAMPLE
        CALL MAINPG
```

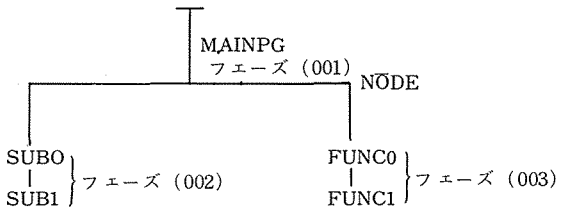
この場合にはすべての主プログラム及び手続き副プログラムが常時主記憶装置上にあるのでここではほとんど問題にすべきことはない。

2）3つのフェーズからなる実行プログラムをつくる場合。

但しここではSUBO及びSUB1とFUNCO及びFUNC1は互に引用しあう事の全くない副プログラムであるとする。この時制御カードは次のようにつくられる。

図4

```
COL1      9      COL6     COL21
MON$$            EXEQ     LINKLOAD,,,,,,,EXTEND
                 PHASE    EXAMPLE
                 SCALL    SUBO
                 SCALL    SUB1
                 SCALL    FUNCO
                 SCALL    FUNC1
                 CALL     NFSAUTOVLY
                 CALL     MAINPG
          NODE   PHASE
                 CALL     SUBO
          NODE   PHASE
                 CALL     FUNCO
```



このプログラムの実行時において，外部手続きはSUB0, FUNC0, SUB1, FUNC1の順序で引用されている。外部手続SUB0又はSUB1が引用された時にはフェーズ002が主記憶装置上にロードされ，外部手続きFUNC0又はFUOC1が引用された時にはフェーズ003が主記憶装置上にロードされる。ここの例ではフェーズは交互に主記憶装置上にロードされるが，一般には引用された手続き副プログラムを含むフェーズがすでに主記憶装置上にロードされている場合には改めてロードされることはない。

共通ブロック (COMMON BLOCK) 内のデータ

共通ブロックはプログラム部分とは別に主記憶装置内でその領域をもつ。したがってフェーズのオーバーレイの際にはプログラム部分とは別に，主記憶装置内での値を持ったまま保存されている。

1）無名共通ブロック内での変数や配列の内容は，フェーズの再ロードによっても全く変らない。

2）名前付き共通ブロック内のデータは，その共通ブロックをもつフェーズがロードされる毎に初期状態に戻される。但し初期値設定副プログラムにより初期値が設定されていれば，その初期値に戻される。名前付共通ブロックは，そのブロック名をもつプログラム単位が最初にリンクロードされたフェーズ内にとられている。

このように共通ブロックのもつ性質を充分に確認したうえでフェーズのオーバーレイを実行する事が重要である。

3.2 基本外部サブルーチン PHASEL と PHASER

基本外部サブルーチンPHASELとPHASERをフォートラン・プログラム中で使うことにより，自動オーバーレイと同じく，樹状構造のプログラムの各々のフェーズのロードをプログラマの責任においておこなうことができる。但し自動オーバーレイの手法を用いたプログラム内においてこれらのサブルーチンを使用することはできない。また自動オーバーレイの場合との違いは，フォートラン・プログラム中にサブルーチンPHASERとPHASELを組み込まなければならない事と，これらのサブルーチンの引用によって主記憶装置内にロードされる各々のフェーズに必ず1つの主プログラムがなければならない事の二点です。

サブルーチンPHASERとPHASELは各々次の機能をもっている。

CALL PHASEL(i)

i：整数型の算術式で2～997の間の値をとる。

この文の実行によりiの値をフェーズ番号としてもつフェーズが主記憶装置内にロードされる。そして実行の制御は，その新たにロードされたフェーズ内の主プログラムに渡る。

CALL PHASER

CALL PHASEL(i) 文によってロードされたフェーズ内においてこの文が実行されると，最後に実行されたCALL PHASEL(i) 文の次の実行文に実行の制御が返る。

各々のフェーズが主記憶装置上に次々とロードされてゆき，旧いフェーズの上からオーバーレイされてゆくため，遂次主記憶装置内にあるフェーズの状態を確認しながらプログラムの実行を制御することが大切である。以下にサブルーチンPHASER及びPHASEL(i)を用いる場合に注意しなければならないであろうと思われる事項を列記しておきます。

1）CALL PHASER文の累積実行回数は，CALL PHASEL(i) の累積実行回数よりもいかなる時点においても超えてはならない。

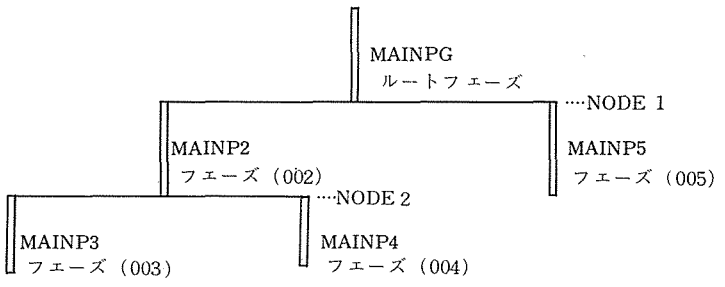
2）CALL PHASER文を実行しても，返るべきフェーズがすでに主記憶装置内でオーバーレイのため消えてしまっている場合には，その返るべきフェーズへは返ることはできない。

3）1つのチェイン内で，より下にあるフェーズ内の副プログラムを異なるフェーズより引用することはできない。(自動オーバーレイの場合との違い)

4）CALL PHASEL(i) 文によりロードされるフェーズは，その文を実行するフェーズと一般に同じチェインになければならない。

これら基本外部サブルーチンの引用によって実行されるプログラムは，リンクロード時にノード名を用いる事により樹状構造を定めなければならない。図5にはPHASEL(i) 及びPHASER を用いたフォートラン・プログラム及びこれらの実行に対するリンケージローダーへの制御カードの例を示した。図5中で前半はフォートラン・プログラムであり，後半はこれらプログラムに対する制御カードである。なお，基本外部サブルーチンPHASELとPHASERを使う場合には，フェーズごとにTITLEカードによりプログラム名をつけておかなければならない。TITLEカードは各フェーズの主プログラムの先頭につけられます。TITLE カードについて詳しくはFORTRANプログラミング説明書を参照して下さい。

図5　樹状構造プログラムの構造（PHASEL及びPHASERを使用）



フォートランプログラム及びシステム制御カード

```
COL 1      6        11
  $JOB             6000GS0000
  $LIMIT           001, 00050, 00000
  $SPECIAL
        MON$$      ASGN MW8, WORKFILEGO, S
        MON$$      ASGN MGO, MW8
        MON$$S01   EXEQ FORTRAN, , , , , , RLS
    TITLEMAINPG
  C MAIN
          DIMENSION A(500)
          DO 10 I=1, 500
     10   A(I)=1.0
          WRITE(6, 100)
    100   FORMAT(1H1, 6X, ▼THE TEST OF THE PHASEL AND PHASER▼//
         1            10X, ▼* * * * MAINPG (0.5KWS) * * * *▼)
          CALL PHASEL(002)
          CALL PHASEL(005)
          STOP
          END
    TITLEMAINP2
  C MAINP2
          DIMENSION A(30000)
          DO 10 I=1, 30000
       10 A(I)=1.0
          WRITE(6, 100)
      100 FORMAT(1H3, 20X, ▼* * * * MAIP2 (30KWS) * * * *▼)
          CALL PHASEL (003)
          CALL PHASEL (004)
          CALL PHASER
          STOP
          END
    TITLEMAINP3
  C MAINP3
          DIMENSION A(10000)
          DO 10 I=1, 10000
       10 A(I)=1.0
          WRITE(6, 100)
```

```
  100 FORMAT(1H3, 30X, ▼* * * * MAINP3 (10KWS) * * * *▼)
      CALL PHASER
      STOP
      END
 TITLEMAINP4
C MAINP4
      DIMENSION A(10000 )
      DO 10 I=1, 10000
   10 A(I)=1.0
      WRITE(6, 100)
  100 FORMAT(1H3, 30X, ▼* * * * MAINP4 (10KWS) * * * *▼)
      CALL PHASER
      STOP
      END
 TITLEMAINP5
C MAINP5
      DIMENSION A(40000)
      DO 10 I=1, 40000
   10 A(I)=1.0
      WRITE(6, 100)
  100 FORMAT(1H3, 20X, ▼* * * * MAINP5 (40KWS) * * * *▼)
      CALL PHASER
      STOP
      END

    COL6          16
      MON$$       COND NOGO
      MON$$       ASGN LIB, LO, FIL=ZZRELPUBLC, PMT
      MON$$       ASGN MLB, DO, FIL=ZZRELOCATS, PMT
      MON$$       ASGN MW1 WORKFILE01, L
      MON$$       ASGN MW3, WORKFILE03, L
      MON$$       ASGN MW9, WORKFILEJB, S
      MON$$       ASGN MJB, MW9
      MON$$02     EXEQ LINKLOAD, , , , , , EXTEND, , , RLS
                  PHASECENTER
                  CALL MAINDG
          NODE1   PHASE
                  CALL MAINP2
          NODE2   PHASE
                  CALL MAINP3
          NODE2   PHASE
                  CALL MAINP4
```

```
      NODE1    PHASE
               CALL MAINP5
MON$$          COND NOGO
MON$$          ASGN MR5, TPA1
MON$$          ASGN MR7, L1, FIL=DIRBINNNU
MON$$          UEPOPNO, N
MON$$          DSGN 100K
MON$$S03       EXEQ CENTER, MJB, , 09, M
```

出力結果

```
THE TEST OF THE PHASEL AND PHASER
    **** MAINPG (0.5KWS) ****
                **** MAIP2 (30KWS) ****
                            **** MAINP3 (10KWS) ****
                            **** MAINP4 (10KWS) ****
                **** MAINP5 (40KWS) ****
プログラムの MEMORY SIZE                0340K
```

3.3 基本外部サブルーチン CHAIN

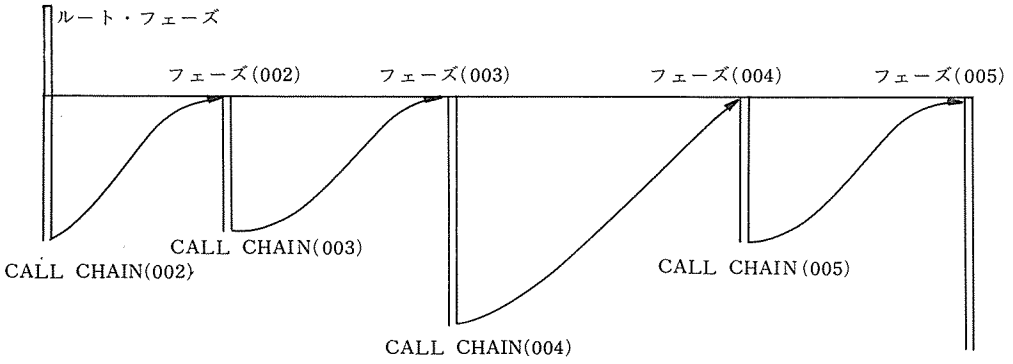
基本外部サブルーチン CHAIN は，樹状構造をもたない，すなわちプログラムの制御を次のフェーズに移した後に元のプログラム単位を含むフェーズに再び制御を戻す必要がない場合に使用される。したがって CHAIN の引用によってオーバーレイされてしまったフェーズは再びロードされることはない。基本外部サブルーチン CHAIN は次の機能をもつ

CALL CHAIN(i)

i：整数型の算術式で 2～997 の範囲の値をとることができる。

この文の実行により i の値をフェーズ番号としてもつフェーズが主記憶装置内にロードされ，そのフェーズ内の主プログラムに実行の制御が移る。

図6　チェイン・ジョブの場合のプログラム構造



チェイン・ジョブのフォートラン・プログラム及びシステム制御カード

```
COL 1          11
 $JOB          6000GS0000
 $LIMIT        001,00050,00000
 $SPECIAL
       MON$$      ASGN MW8,WORKFILEGP,S   GO
       MON$$      ASGN MGO,MW8
       MON$$S01   EXEQ FORTRAN,,,,,,RLS
   TITLEMAINPG
COL 1
 C     MAINPG
       DIMENSION A(10000)
       DO 10 I=1,10000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H5,5X,▼**** MAINPG (10KWS) ****▼)
       CALL SUB11
       CALL SUB12
       CALL CHAIN(002)
       STOP
       END

       SUBROUTINE SUB11
       DIMENSION A(10000)
       DO 10 I=1,10000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H ,20X,▼**** SUB11 (10KWS) ****▼)
       RETURN
       END

       SUBROUTINE SUB12
       DIMENSION A(10000)
       DO 10 I=1,10000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H ,20X,▼**** SUB12 (10KWS) ****▼)
       RETURN
       END
```

```
   TITLEMAINP2
COL 1
 C   MAINP2
       DIMENSION A(10000)
       DO 10 I=1,10000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H5,5X,▼* * * * MAINP2(10KWS) * * * *▼)
       CALL SUB21
       CALL SUB22
       CALL CHAIN(003)
       STOP
       END

       SUBROUTINE SUB21
       DIMENSION A(10000)
       DO 10 I=1,10000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H ,20X,▼* * * * SUB21 (10KWS) * * * *▼)
       RETURN
       END

       SUBROUTINE SUB22
       DIMENSION A(10000)
       DO 10 I=1,10000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H ,20X,▼* * * * SUB22 (10KWS) * * * *▼)
       RETURN
       END

   TITLEMAINP3
COL 1
 C   MAINP3
       DIMENSION A(30000)
       DO 10 I=1,30000
    10 A(I)=1.0
       WRITE(6,100)
   100 FORMAT(1H5,5X,▼* * * * MAINP3(30KWS) * * * *▼)
       CALL SUB31
       CALL CHAIN(004)
       STOP
       END
```

```
      SUBROUTINE SUB31
      DIMENSION A(10000)
      DO 10 I=1,10000
   10 A(I)=1.0
      WRITE(6,100)
  100 FORMAT(1H ,20X,▼* * * * SUB31 (10KWS) * * * *▼)
      RETURN
      END

  TITLEMAINP4
COL 1
 C   MAINP4
      DIMENSION A(30000)
      DO 10 I=1,30000
   10 A(I)=1.0
      WRITE(6,190)  100
  100 FORMAT(1H5,5X,▼* * * * MAINP4(30KWS) * * * *▼)
      CALL CHAIN(005)
      STOP
      END

  TITLEMAINP5
COL 1
 C   MAINP5
      DIMENSION A(20000)
      DO 10 I=1,20000
   10 A(I)=1.0
      WRITE(6,100)
  100 FORMAT(1H5,5X,▼* * * * MAINP5 (20KWS) * * * *▼)
      CALL SUB51
      STOP
      END


      SUBROUTINE SUB51
      DIMENSION A(10000)
      DO 10 I=1,10000
   10 A(I)=1.0
      WRITE(6,100)
  100 FORMAT(1H ,20X,▼* * * * SUB51(10KWS) * * * *▼)
      CALL SUB511
```

```
      CALL SUB512
      CALL SUB513
      RETURN
      END

      SUBROUTINE SUB511
      DIMENSION A(5000)
      DO 10 I=1,5000
   10 A(I)=1.0
      WRITE(6,100)
  100 FORMAT(1H ,30X,▼* * * * SUB511 (5KWS) * * * *▼)
      RETURN
      END

      SUBROUTINE SUB512
      DIMENSION A(5000)
      DO 10 I=1,5000
   10 A(I)=1.0
      WRITE(6,100)
  100 FORMAT(1H ,30X,▼* * * * SUB512 (5KWS) * * * *▼)
      RETURN
      END

      SUBROUTINE SUB513
      DIMENSION A(5000)
      DO 10 I=1,5000
   10 A(I)=1.0
      WRITE(6,100)
  100 FORMAT(1H ,30X,▼* * * * SUB513 (5KWS) * * * *▼)
      RETURN
      END
```

```
6           16      21
MON$$       COND  NOGO
MON$$       ASGN  LIB, L0, FIL=ZZRELPUBLC, PMT
MON$$       ASGN MLB, D0, FIL=ZZRELOCATS, PMT
MON$$       ASGN  MW1, WORKFILE01, L
MON$$       ASGN  MW3, WORKFILE03, L
MON$$       ASGN  MW9, WORKFILEJB, S
MON$$       ASGN  MJB, MW9
MON$$S02    EXEQ  LINKLOAD,,,,,,EXTEND,,,RLS
            PHASECENTER
            CALL NFSAUNIT7
            CALLMAINPG
            PHASE
            BASE1MAINPG
            CALL  MAINP2
            PHASE
            BASE1MAINPG
            CALL  MAINP3
            PHASE
            BASE1MAINPG
            CALL  MAINP4
            PHASE
            BASE1MAINPG
            CALL  MAINP5

MON$$       COND NOGO
MON$$       ASGN MR7, L1, FIL=DIRBINNNU
MON$$       UEPOPNO, N
MON$$       DSGN 100K
MON$$S03    EXEQ CENTER, MJB,,09,M
```

出力結果

```
＊＊＊＊ MAINPG (10KWS) ＊＊＊＊
                    ＊＊＊＊ SUB11 (10KWS) ＊＊＊＊
                    ＊＊＊＊ SUB12 (10KWS) ＊＊＊＊

＊＊＊＊ MAINP2 (10KWS) ＊＊＊＊
                    ＊＊＊＊ SUB21 (10KWS) ＊＊＊＊
                    ＊＊＊＊ SUB22 (10KWS) ＊＊＊＊
```

```
＊＊＊＊ MAINP3 (30KWS) ＊＊＊＊
                    ＊＊＊＊ SUB31 (10KWS) ＊＊＊＊

＊＊＊＊ MAINP4 (30KWS) ＊＊＊＊

＊＊＊＊ MAINP5 (20KWS) …＊＊＊  ＊
                    ＊＊＊＊ SUB511 (5KWS) ＊＊＊＊
                    ＊＊＊＊ SUB512 (5KWS) ＊＊＊＊
                    ＊＊＊＊ SUB513 (5KWS) ＊＊＊＊

        プログラムの
          MEMORY SIZE          0376KCH
```

図6は，基本外部サブルーチンCHAINが引用される場合のフォートラン・プログラムの例である。PHASEカードの次にある

```
16Col
BASE1MAINPG
```

のカードはルート・フェーズの中のMAINPGをベースにしてそれ以下のフェーズがオーバーレイされることを示すものである。

基本外部サブルーチンCHAINを引用する場合，次の点に注意しなければならない。

1）ファイル・テーブル・サブプログラム及びFORTRAN実行時モニター等は，ルート・フェーズの常駐部分になければならない。

2）各フェーズは必ず主プログラムをもたなければならない。そしてそれぞれのフェーズのロードにより主記憶装置内でつぶされ，再びロードされることはない。

4．むすび

限られた大きさの主記憶装置で大きなプログラムの実行が可能となる事の意義の大きさは利用者個々により異なるでしょう。しかし現用計算機システムにおいて，単フェーズ・プログラムのままでは実行不可能な大容量プログラムが，多重フェーズ・プログラムにする事により実行可能となる事の意義は非常に大きい。また現実には，必要なコアー・サイズの大きさにより受付ジョブ区分がなされ，しかも大容量プログラムの処理が著しく不利となる状態にあっては，フェーズを分割する事により，最大必要コアー・サイズを大幅に小さくできることの意義はさらに大きくなるでしょう。

実行中の作業領域への周辺機器の利用と共に，ここで説明した多重フェーズ・プログラムを応用されまして，計算機システムをより有効に利用されます事を望みます。なお，ここでの制御カードの説明は一般的な形式でおこないましたが，実際の場合には簡易コントロールカードにより省略化できる場合もあります。この点につきましては「N-700ジョブコントロールカードの手引き」を参照して下さい。