[04_03]九州大学大型計算機センター広報 : 4(3)

https://doi.org/10.15017/1467976

出版情報：九州大学大型計算機センター広報. 4 (3), pp.1-54, 1971-06-26. 九州大学大型計算機センター
バージョン：
権利関係：

制御文のいろいろ

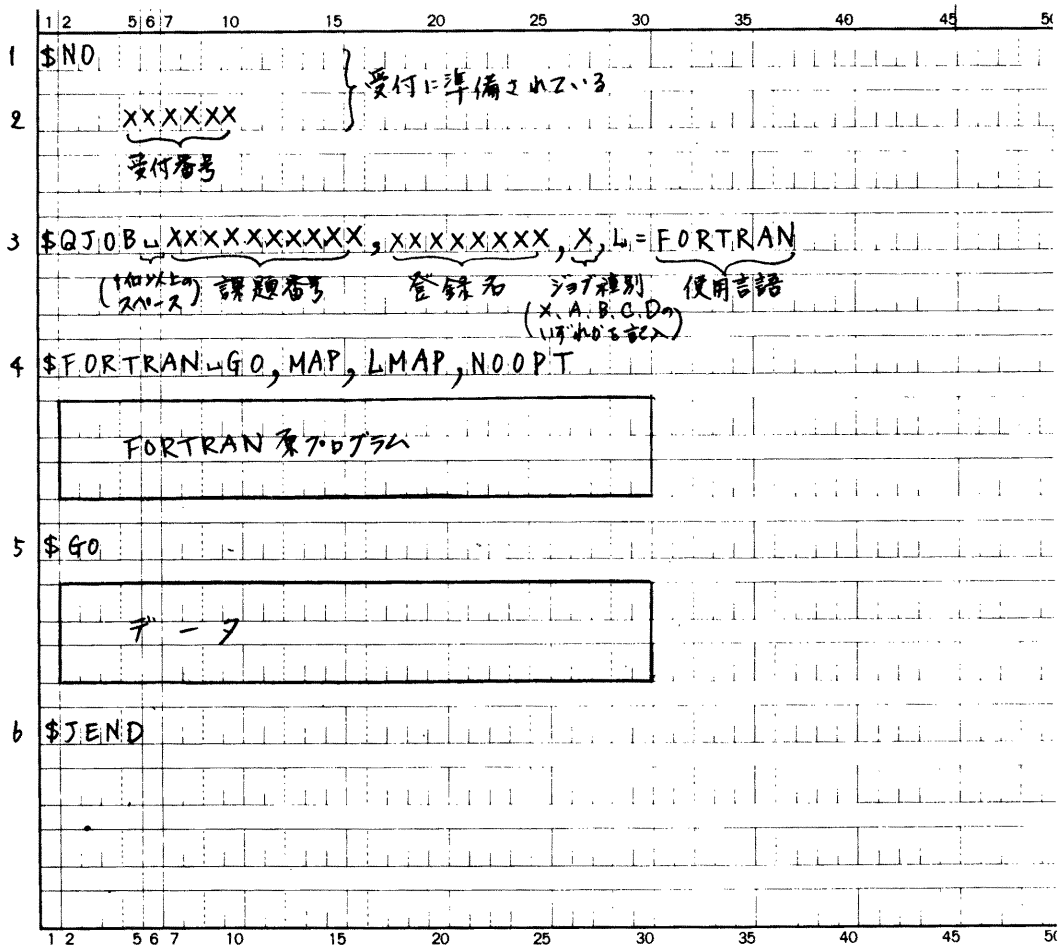
3月下旬に、ファイル装置を主とした機器構成の変更があり、それと共にソフトウェアの面でも大幅な変更がありました。センターでは制御プログラムの大幅なレベルアップに伴い、制御文もそれに合わせ能率の良いものをと考え、一部変更いたしました。詳細については、すでにセンターニュース（№14）でお知らせしましたが、ここではいくつかの例をあげながら、制御文の使用法について説明いたします。

1．FORTRANプログラムの実行（SSLを使用しない場合）
2．FORTRANプログラムの実行（SSLを使用する場合）
3．ALGOLプログラムの実行
4．FORTRANプログラムとFASPプログラムを結合して実行
5．コンパイルだけしたいとき
6．カード出力があるとき
7．LP出力量を少なくしたいとき
　（イ）FORTRANのマップを出さないとき
　（ロ）プログラムのリストを出さないとき
　（ハ）LIEDのマップを出さないとき
8．プログラムデバッグ中の制御文とデバッグ終了後の制御文

1．FORTRANプログラムの実行

科学用サブルーチンライブラリ（ＳＳＬ）やセンターで作成したライブラリ等をプログラム中で使用している時は　2．を読んでください。

例1

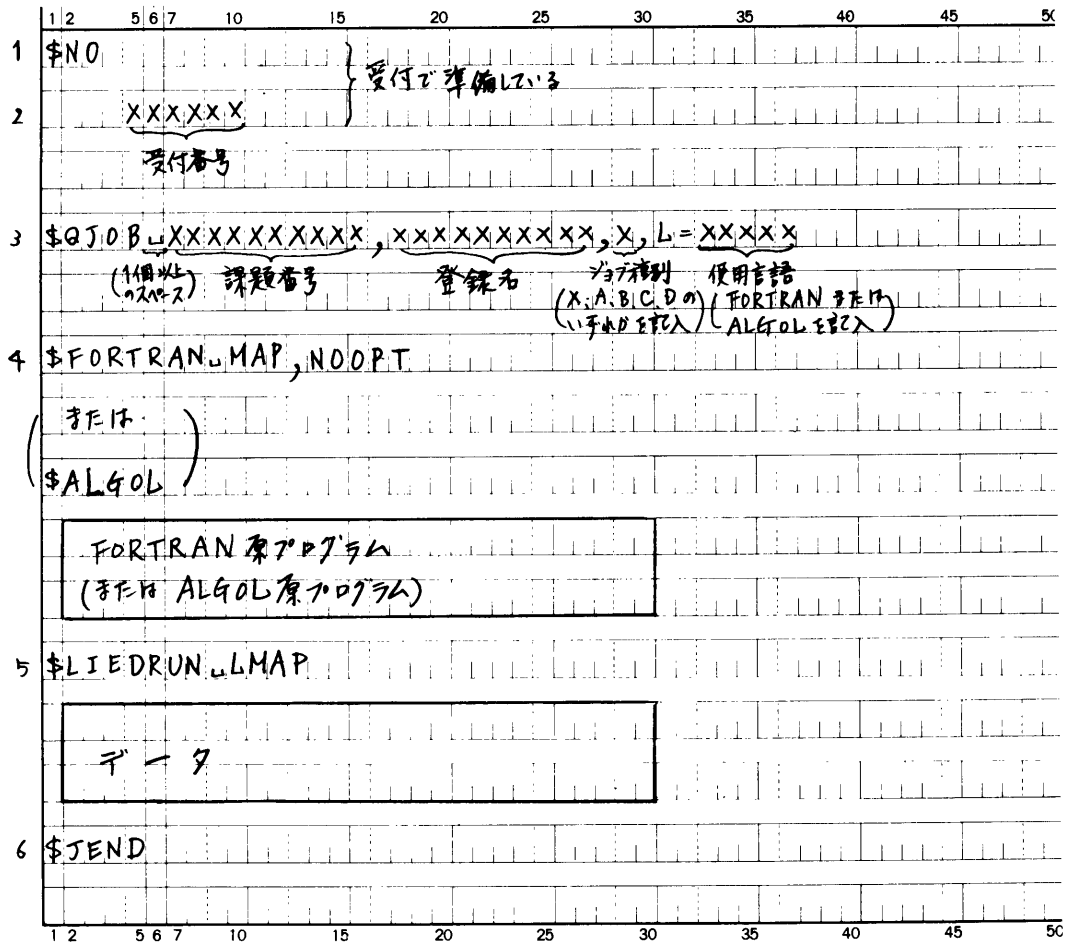


2．FORTRANプログラムの実行

科学用サブルーチンライブラリ（ＳＳＬ）やセンターで作成したライブラリ等をプログラム中で使用しているとき。

3．ALGOLプログラムの実行

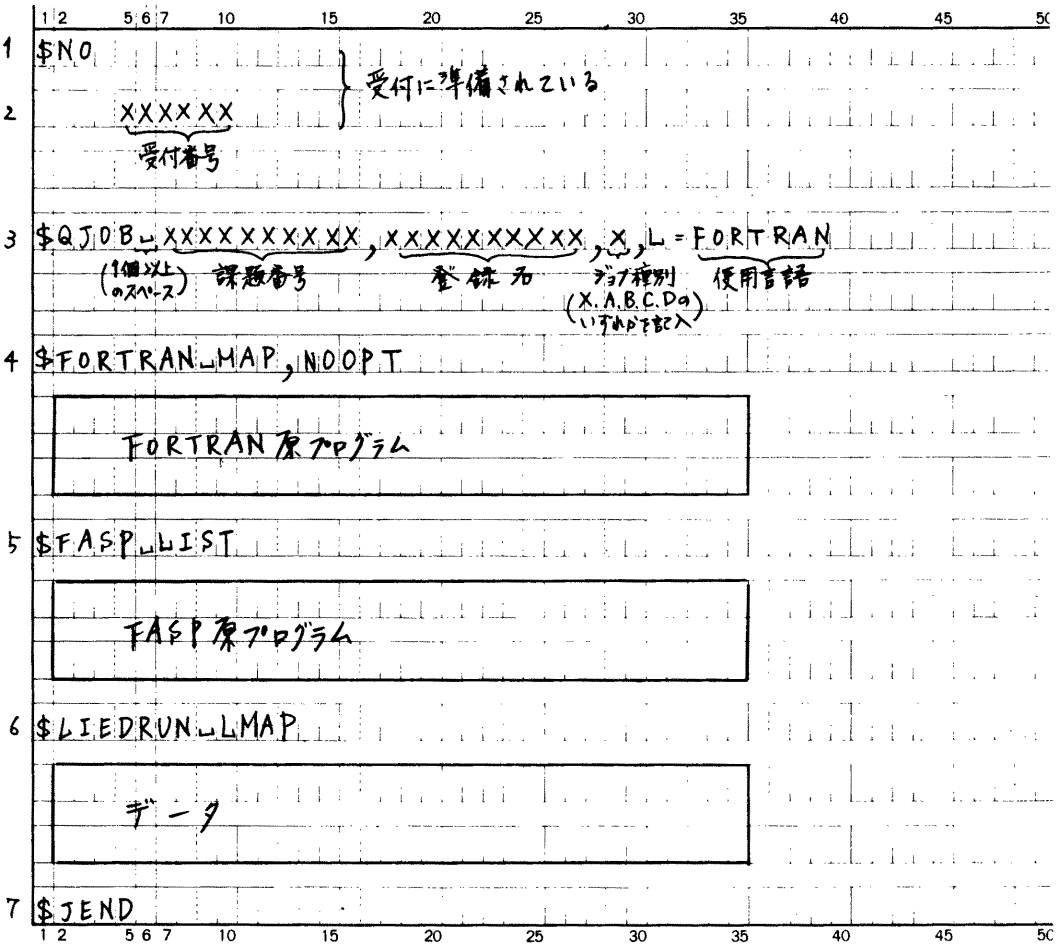
例2



4．FORTRANプログラムとFASPプログラムを結合して実行する場合

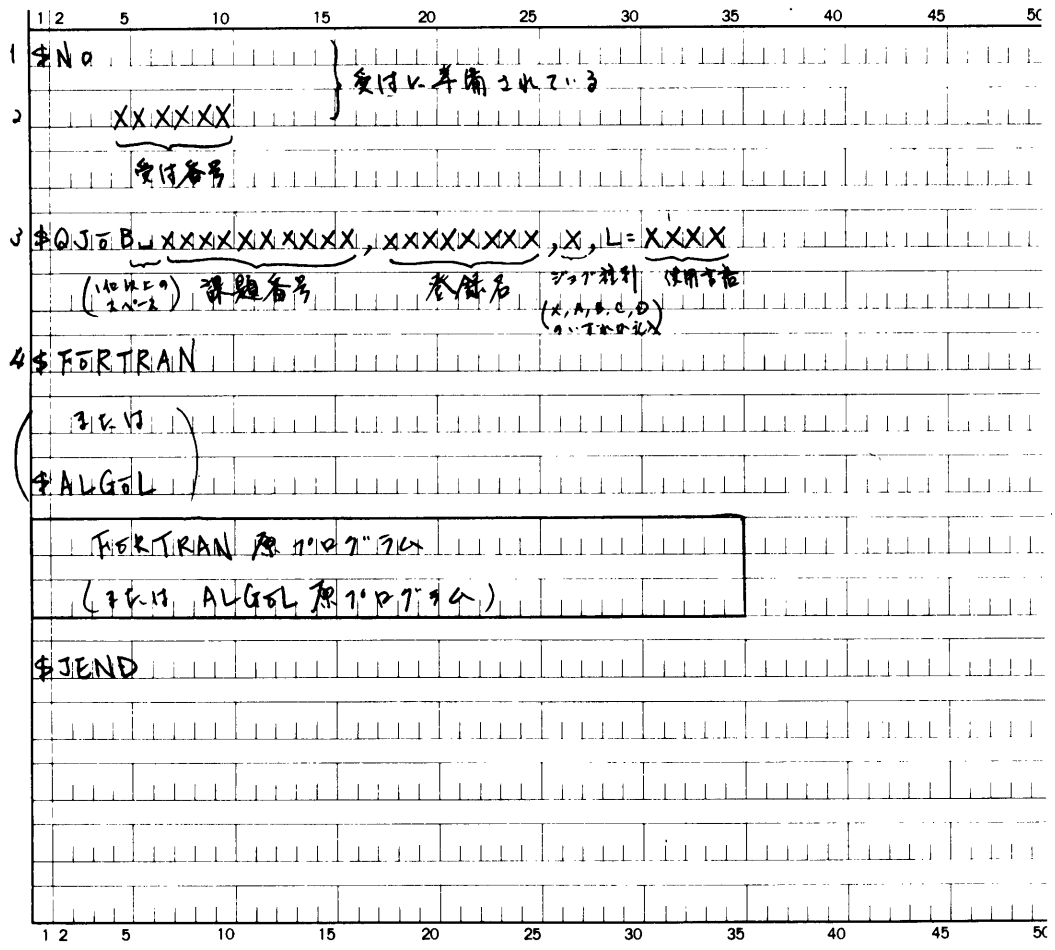
科学用サブルーチンライブラリやセンターで作成したライブラリ等も使用できます。

例3



5．コンパイルだけしたいとき

例4

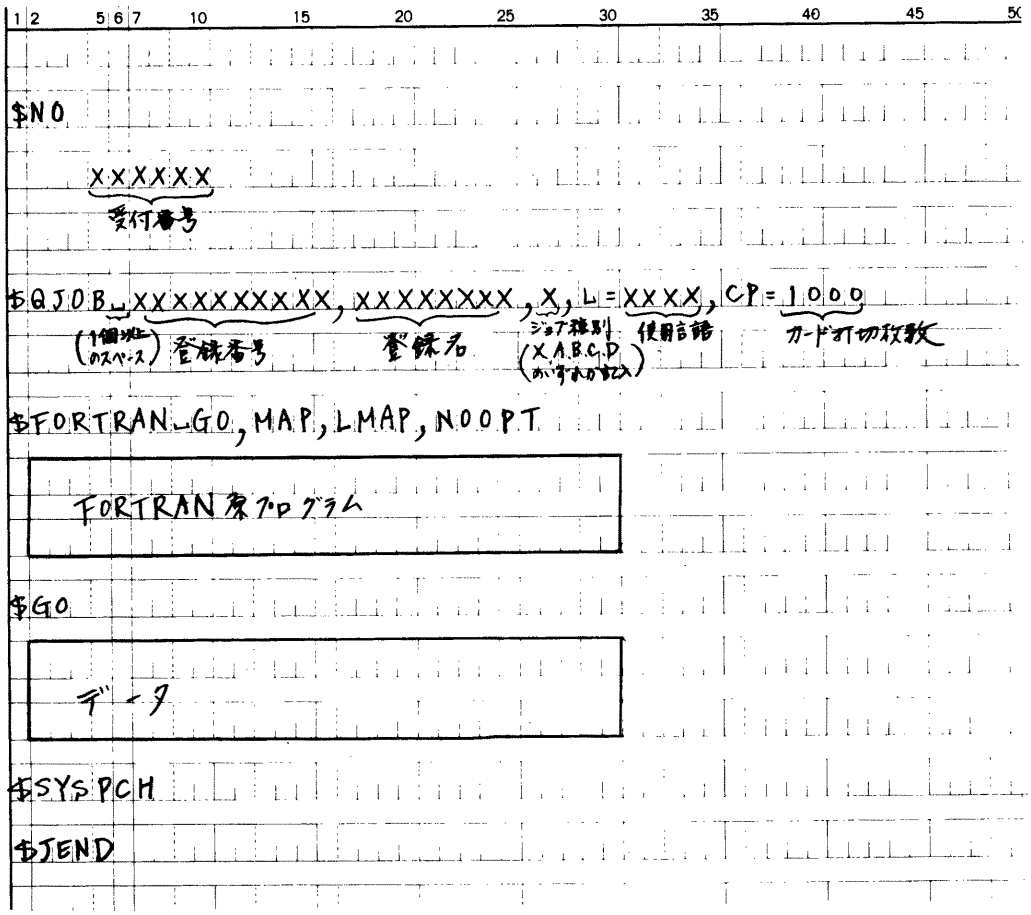


6．カード出力があるとき

（イ）QJOBカードにCP＝n のパラメータを入れる。

（ロ）データカードの最後に　$SYSPCH の制御文を入れる。

例5



7．LP出力量を少なくしたいとき

出力量打切りなどのため、出力量を少なくしたいときがあります。その場合は計算結果の出力量を少なくすること以外に次のような方法があります。

（イ）FORTRANのマップを出さないとき。

$FORTRANのパラメータMAPを除く。

例6　$FORTRAN　GO　（例1のとき）

または

$FORTRAN　　（例2のとき）

（ロ）プログラムリストを出さないとき

$FORTRAN, $ALGOLの制御文にNOLISTというパラメータを追加する。

例7．1　$FORTRAN␣NOLIST

または

$FORTRANT␣GO, NOLIST（※）　（例1のとき）

例7．2　$ALGOL␣NOLIST　（例2のとき）

（ハ）LIEDのマップを出さないとき

$FORTRANまたは$LIEDRUNのパラメータLMAPを除く。

例8．1　$FORTRAN␣GO, MAP　（例1のとき）

例8．2　$LIEDRUN　（例2のとき）

※　この場合はLIEDのリストも出力されない。

LIEDのリストを出さない場合として「$LIEDRUN NOLIST」もありますが、これらは頁数の節約には全然なりません。

注：実行時のエラーが出たときで、FORTRANのマップ、LIEDのマップがないときはプログラム相談を受けに行っても門前払いになることが少なくないので、できる限りマップはとった方が良いようです。少なくともLIEDのマップは必ずとるようにしておいた方が良いでしょう。

8．プログラムデバッグ中の制御文とデバッグ終了後の制御文

実際にプログラムを書いて、計算機で実行させるとき、最初にコンパイルエラーがあり、それを修正し、次にテストデータをもって実行テストを行なうという順になるはずです。普通よほど単純な論理でできたプログラムでない限り、期待する結果が出ないことの方が多い、つまり、論理ミスがあるわけです。従って、何回かのテストランをし、論理ミスを修正した後、期待している計算ができるわけです。この間使用する制御文は次のようにすると良いでしょう。

・デバッグ中

例1または2の制御文

・デバッグ終了後

例1または2の制御文で$FORTRANのパラメータNOOPTをとる。

ALGOLの場合はこの制御文のままです。

FORTRANの場合NOOPTを取る（OPTにする）と実行プログラムが最適化されるという利点があります。

最適化というのは実行速度を早くするために、配列の扱い等を効率良くコンパイルすることです。しかしこの場合の欠点として、FORTRANのDEBUG文が使用できないこと、実行時にエラーが起ったとき、発生個所がわかりにくくなります。