INFORMATION ABOUT THE PANDA2 SYSTEM FOR PANEL OPTIMIZATION

****

TABLE OF CONTENTS

****

I. SUMMARY

The purpose of PANDA2 is to find the minimum weight design of a stiffened panel made of laminated composite material. Of course, simple isotropic panels can also be designed. The panel can be loaded by up to 5 independent sets of loads, (Nx, Ny, Nxy, and p), in which Nx is the load in the axial direction (negative for compression) and p is the normal pressure. PANDA2 represents a more detailed treatment of certain behavior not handled by the original PANDA [1]. In particular, optimum designs can be obtained for panels with locally post-buckled skin and for panels with hat stiffeners; and stresses induced by uniform normal pressure are included in the analysis.

Optimization is carried out based on several independently treated structural models of the panel, as follows:

1. PANDA-type models [1] for general, local, and panel bucking, crippling of stiffener parts, and rolling of stiffeners with and without participation of the panel skin are included. These models are described in detail in [1]. (See Table 1 and Figs. 1-4 of [1].)

2. Buckling load factors and post-local buckling behavior are calculated for what is termed in PANDA2 a "panel module". Such a module is depicted in Figs. 1 and 2. A module includes the cross section of a stiffener plus the panel skin of width equal to the spacing between stiffeners. In this model the panel module cross section is divided into segments, each of which is discretized and analyzed via the finite difference energy method. Variation of deflection in the axial direction is assumed to be harmonic [sin(n*theta) or cos(n*theta)]. This one-dimensional discretization is similar to that used in the BOSOR programs for the analysis of shells of revolution. In fact, many of the subroutines for buckling and vibration analysis are taken from BOSOR4 and modified slightly. The modification is necessary to handle prismatic structures instead of shells of revolution. Both local and wide-column instability can be handled with the same structural model.

3. A discretized model of the entire width of the panel, treated in this case with stringers smeared out. This model is introduced only if the axial load varies across the width of the panel.

In the panels designed by PANDA2 the skin between stringers may buckle well before failure of the panel. The maximum stress components and therefore stress constraints in the optimization analysis are computed including growth and modification of the local skin buckling mode as predicted in a theory similar to that formulated by Koiter in 1943 [4].

A typical panel module is shown in Fig. 1.

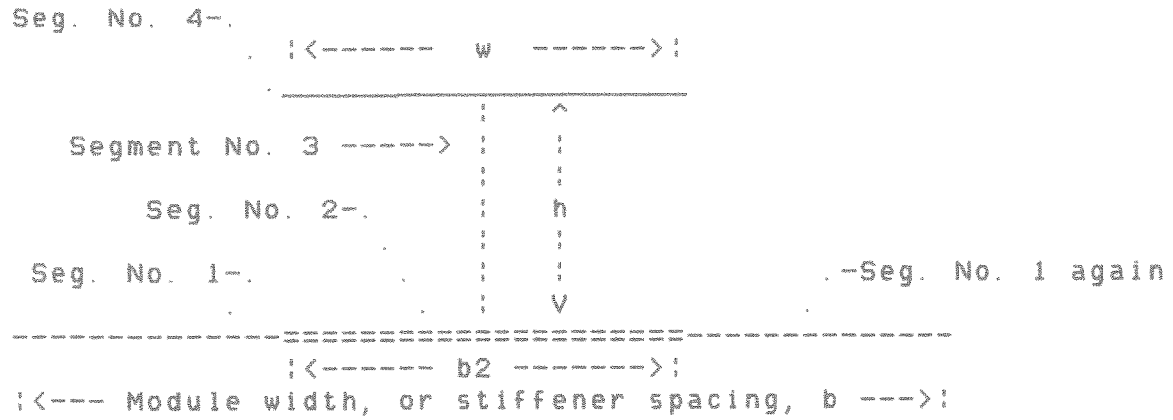


FIGURE 1 SINGLE MODULE OF A PANEL WITH T-SHAPED STRINGERS (AXIAL LOAD Nx ACTS NORMAL TO THE SCREEN)

Figure 2 shows how the single module fits into the complete, multi- module panel:

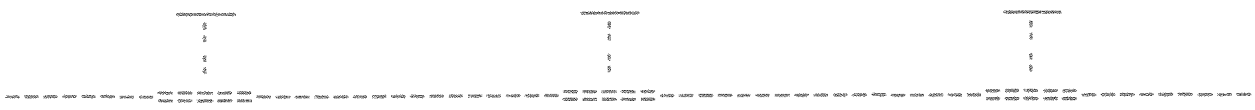


FIGURE 2 PANEL WITH T-SHAPED STRINGERS, CONSISTING OF 3 MODULES

Symmetry conditions are applied at the left and right edges of the single module. After the optimum design is obtained, you can check the accuracy of the general instability load predicted from the single-module model by running a multi-module model with BOSOR4 [2]. The input data file for this multi-module model are generated automatically by the PANDA2 system.

As with PANDA, the program PANDA2 consists of several independently executable processors which share a common data base. In the processor BEGIN you supply a starting design (perhaps a design produced by PANDA). In DECIDE you choose decision variables for the optimization analysis and their upper and lower bounds; you choose linking variables and their factors of proportionality; and you choose "escape" variables (explained below). In MAINSETUP you choose strategy parameters such as number and range of axial half-waves in the local buckling mode, number of design iterations in the optimization problem, and factors of safety for general instability and maximum stress. In PANDAOPT

you cause the analysis to be performed. PANDAOPT consists of two main branches: in one branch the structural analyses (stress, buckling and postbuckling) are performed and in the other new designs are produced by the optimizer CONMIN, written by Gary Vanderplaats [3].

****

III. HOW TO RUN PANDA2

To operate PANDA2 you must have the BOSOR4 program system on a subdirectory with the name BOSOR4. The PANDA2 system uses many of the libraries from the BOSOR4 system, and certain commands (for example PANEL followed by BOSORALL) use the BOSOR4 processors in toto.

You first activate PANDA2 commands via the command PANDA2LOG. This command must be given before you do any PANDA2 work. Upon typing the command PANDA2LOG, you see the following display on your screen:

```
 PANDA2 COMMANDS HAVE BEEN ACTIVATED.

PANDA2 commands are:
  HELPAN       (get information on PANDA2.)
  BEGIN        (you provide a starting design)
  SETUP        (PANDA2 generates BOSOR4 input)
  BOSMODEL     (batch run of BOSOR4 preprocessor)
  DECIDE       (you choose decision variables)
  MAINSETUP    (you provide strategy for batch run)
  PANDAOPT     (launch batch run of mainprocessor)
  PLOTTER      (launch batch run to plot stuff)
  CHANGE       (assign new values to parameters)
  PANEL        (generate BOSOR4 input for panel)
  CLEANPAN     (delete many files for a case)

Please consult the following sources for more
information about PANDA2:

  1.  HELPAN     file    (type HELPAN)
  2.  SAMPLE.CAS file    (sample case)
  3.  Documents listed under HELPAN OVERVIEW REF
  4.  Run tutorial options of BEGIN, DECIDE, etc.
```

You then prepare input data interactively via the BEGIN command. These data establish a starting design, material properties, temperature rise pertaining to residual stresses from curing, and in-plane loading. A summary of the data that you provided interactively is stored on a file NAME.BEG, which you will find very useful in future analyses of the same design concept. (NAME is a name that you choose and stick with for the entire case.)

You next type the command SETUP.

A BOSOR4-type of input data file is created by your typing the command SETUP. Following SETUP, you type the command BOSMODEL. This launches a batch run of a BOSOR4-type preprocessor which is almost identical to BOSORREAD (see HELP4 PROGRAMS in the BOSOR4 subdirectory).

The BOSMODEL processor creates a discretized model of a single module of the panel. When the BOSMODEL batch run is finished, you can either type the command DECIDE or the command MAINSETUP. If you choose DECIDE, you will be asked to pick decision variables and their upper and lower bounds, linked variables and linking constants, and escape variables. If you choose MAINSETUP, you will be asked to provide strategy parameters for a simple buckling analysis of a fixed design.

Suppose you choose DECIDE. This choice means that you intend to do an optimization analysis. After picking decision variables and other optimization parameters in DECIDE, you type the command MAINSETUP. Now you establish certain strategy parameters for the buckling and optimization analyses.

You then type the command PANDAOPT, which launches a batch run of the PANDA2 mainprocessor. Simple buckling analyses require up to a minute on the VAX 11/780, and 6 to 10 design iterations in an optimization analysis require about 10 to 20 minutes of CPU time. The results of the PANDAOPT run are stored in a file called NAME.OPM.

After inspecting the NAME.OPM file, you may wish to do more design iterations. If you feel the strategy parameters established in MAINSETUP are still good, simply type the command PANDAOPT again. Look at the new NAME.OPM file. Keep repeating this cycle until you have a satisfactory design. If along the way you want to change the strategy parameters, type MAINSETUP again. If you want to change which parameters are decision variables or upper or lower bounds, or if you want to change the linking variables or linking constants of proportionality, type DECIDE again.

When you are satisfied that you have a good design, you will probably want plots of the cross section and buckling modes. Give the command MAINSETUP and choose the "fixed design" strategy (no optimization). Indicate that you want plots. Then type the command PLOTTER. If you have just done a "fixed design" analysis and if you there indicated that you wanted plots, then simply type PLOTTER.

All the analysis up to this point is for a single panel module. You will want to check the accuracy of the general instability load factor (eigenvalue) from the single-module analysis by comparing it with the general instability load factor from a multi-module model that represents the entire panel. This you do by typing the commands PANEL, BOSOR4LOG, BOSORALL, and BOSORPLOT, in that order. PANEL is like SETUP, except that it sets up an input data file, NAME.ALL, for a multi-module panel. This file is a regular BOSOR4 input data file. The command BOSOR4LOG activates the BOSOR4 command set, and the command BOSORALL initiates a batch run of the BOSOR4 pre-, main-, and postprocessors. BOSORPLOT gets plots of the discretized multi-module panel and of buckling modes.

SAMPLE RUNSTREAM WITH PANDA2...

```
PANDA2LOG          (you activate PANDA2 commands. Please
```

```
                          note that you must insert the following
                          two statements in your LOGIN.COM:
                        $ASSIGN userdisk:[username.PANDA2] PANDA2
                        $PANDA2LOG :== @userdisk:[username.PANDA2]PANDA2

        BEGIN            (you establish a starting design)
        SETUP            (an input file for BOSOR4 is set up)
        BOSMODEL         (BOSOR4-type discrete model of the
                          single panel module is set up)
        DECIDE           (you establish optimization parameters)
        MAINSETUP        (you set up PANDA2 analysis strategy)
        PANDAOPT         (you launch batch run that performs
                          whatever analysis you called for
                          in MAINSETUP)
        MAINSETUP        (call for "fixed design" with plots)
        PLOTTER          (launch batch run to generate plot files)
        PANEL            (input file for multi-module panel is
                          setup. This is input to BOSOR4)
        BOSOR4LOG        (activate BOSOR4 command set)
        BOSORALL         (run BOSOR4 to get general buckling
                          of the multi-module panel)
        BOSORPLOT        (get multi-module panel plot file)
```

****

IV. PANDA2 COMMANDS

PANDA2LOG

This command activates the PANDA2 command set. You must use it before you can work with PANDA2. It is advisable to use it also when you have just worked with BOSOR4 modules and you then wish to do more work with PANDA2.

FORMAT:

```
PANDA2LOG
```

EXAMPLE OF USE IN A RUNSTREAM:

```
        PANDA2LOG        (you activate PANDA2 commands. Please
                          note that you must insert the following
                          four statements in your LOGIN.COM:
                        $ASSIGN userdisk:[username.PANDA2] PANDA2
                        $PANDA2LOG :== @userdisk:[username.PANDA2]PANDA2
                        $ASSIGN userdisk:[username.BOSOR4] BOSOR4
                        $BOSOR4LOG :== @userdisk:[username.BOSOR4]BOSOR4
                          You must have BOSOR4 in order to use PANDA2!

        BEGIN            (you establish a starting design)
        SETUP            (an input file for BOSOR4 is set up)
        BOSMODEL         (matrix templates for a panel module
                          are established for later buckling
                          analyses with PANDAOPT)
        MAINSETUP        (set up analysis strategy to be
                          used by PANDAOPT)
        PANDAOPT         (perform PANDA2 analysis)
```

BEGIN

You provide, in a conversationally interactive mode, a starting design for the panel. The interactive input is saved on a file called NAME.BEG, in which NAME is a name of your choice. In future runs of this or a similar case, you can edit the file NAME.BEG, then give the command BEGIN, and tell the system that you are using an existing file. If you goof part way through a case, you can do a CONTROL Y, edit the NAME.BEG file, and re-issue the command BEGIN, telling the system that you are adding to an existing file. The system will answer all the questions up to your goof in a "batch" mode, then return control to you. You can then complete the interactive input.

Output from the PANDA2 preprocessor BEGIN is stored in a file with the name NAME.OPB. You should print this file and inspect it to make sure that the case is as you intend it to be.

FORMAT:

```
BEGIN
```

EXAMPLE OF USE IN A RUNSTREAM:

```
PANDA2LOG          (you activate PANDA2 commands)
BEGIN              (you establish a starting design)
SETUP              (an input file for BOSOR4 is set up)
BOSMODEL           (matrix templates for a panel module
                    are established for later buckling
                    analyses with PANDAOPT)
MAINSETUP          (set up analysis strategy to be
                    used by PANDAOPT)
PANDAOPT           (perform PANDA2 analysis)
```

SETUP

This command causes to be set up a file NAME.ALL suitable as input for the BOSOR4 preprocessor to be activated by your subsequent command, BOSMODEL. NAME must be the same name that you used in BEGIN.

FORMAT:

```
SETUP
```

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN              (you establish a starting design)
SETUP              (an input file for BOSOR4 is set up)
BOSMODEL           (matrix templates for a panel module
                    are established for later buckling
                    analyses with PANDAOPT)
MAINSETUP          (set up PANDA2 analysis strategy)
PANDAOPT           (perform PANDA2 analysis)
```

BOSMODEL

This command causes BOSOR4-type templates of stiffness and load-geometric matrices to be set up, as well as arc length parameters and connectivities.

FORMAT:

```
BOSMODEL
```

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN              (you establish a starting design)
SETUP              (an input file for BOSOR4 is set up)
BOSMODEL           (matrix templates are established)
MAINSETUP          (set up PANDA2 analysis strategy)
PANDAOPT           (perform PANDA2 analysis)
```

DECIDE

You provide, in a conversationally interactive mode, strategy parameters for PANDA2 optimization. You choose which of the problem parameters are to be decision variables (allowed to change) during optimization and their lower and upper bounds. You also choose linked parameters, that is parameters that are not decision variables but vary in proportion to designated decision variables, and you assign factors of proportionality. Finally, you choose "escape" variables, that is variables that should be increased during any design iteration in which CONMIN fails to change the design. You should choose only thicknesses as escape variables.

Your interactive input is saved on a file, NAME.DEC, in which NAME is the same name you used for BEGIN, SETUP, etc. In future runs of this or a similar case, you can edit the file NAME.DEC, then give the command DECIDE, and tell the system that you are using an existing file. If you goof part way through a case, you can do a CONTROL Y, edit the NAME.DEC file, and re-issue the command DECIDE, telling the system that you are adding to an existing file. The system will answer all the questions up to your goof in a "batch" mode, then return control to you. You can then complete the interactive input.

Output from the PANDA2 processor DECIDE is stored in a file with the name NAME.OPD. You should print this file and inspect it to make sure that the case is as you intend it to be.

FORMAT:

```
DECIDE
```

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN              (you establish a starting design)
SETUP              (an input file for BOSOR4 is set up)
BOSMODEL           (BOSOR4-type discrete model set up)
DECIDE             (optimization parameters established)
MAINSETUP          (set up analysis strategy to be
                    used by PANDAOPT)
PANDAOPT           (perform PANDA2 optimization)
```

MAINSETUP

You establish strategy parameters for the PANDAOPT run to follow. Strategy parameters include:

1. type of analysis (simple buckling or optimization)
2. do you want to vary the number of axial waves?
3. factors of safety for general instability and max. strain
4. how many design iterations do you want PANDAOPT to do?
5. do you want to use KOITER local postbuckling theory?

An image of the interactive session in MAINSETUP is stored on a small file with the name NAME.OPT, which is listed at the beginning of the output file from PANDAOPT. (See PANDAOPT).

FORMAT:

MAINSETUP

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN                (you establish a starting design)
SETUP                (an input file for BOSOR4 is set up)
BOSMODEL             (BOSOR4-type discrete model set up)
DECIDE               (optimization parameters established)
MAINSETUP            (set up PANDA2 analysis strategy)
PANDAOPT             (perform PANDA2 optimization)
```

PANDAOPT

This command initiates a batch run of the PANDA2 mainprocessor. The results are stored in a file NAME.OPM, in which NAME is the same name you chose for BEGIN, SETUP, BOSMODEL, MAINSETUP, etc. To do several sets of design iterations you issue the command PANDAOPT again and again. Make sure to print out the file NAME.OPM after each batch run corresponding to each PANDAOPT. Only the latest NAME.OPM is retained.

FORMAT:

PANDAOPT

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN                (you establish a starting design)
SETUP                (an input file for BOSOR4 is set up)
BOSMODEL             (matrix templates for a panel module
                      are established for later buckling
                      analyses with PANDAOPT)
MAINSETUP            (set up analysis strategy to be
                      used by PANDAOPT)
PANDAOPT             (perform PANDA2 analysis)
```

PLOTTER

This command launches a batch run that eventually produces plots of the panel module cross section with local and wide column

buckling modes. General instability buckling modes of the entire panel width may also be plotted, depending on strategy chosen in MAINSETUP. Note: Before using PLOTTER, you must have a file NAME.OPT (input data for MAINSETUP) in which:

1. the "fixed design" analysis option was chosen (no optimization)
2. you indicated that you wanted plots.

You can, of course, produce the proper NAME.OPT file simply by giving the command MAINSETUP and answering the questions posed by it interactively.

PANEL

This command is used to find the general instability buckling load of a full-size panel built up of the panel modules such as the single panel module used in the PANDA2 optimization analysis. In this way the accuracy of the single module analysis is checked, at least with regard to general instability, which is more in doubt than is that for the local buckling analysis.

PANEL sets up a file NAME.ALL, which is subsequently used as input to the BOSOR4 buckling analysis of the multi-module panel that you initiate by subsequently typing the commands BOSOR4LOG, BOSORALL, and BOSORPLOT.

FORMAT:

```
PANEL
```

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN              (you establish a starting design)
SETUP              (an input file for BOSOR4 is set up)
BOSMODEL           (BOSOR4-type discrete model of the
                    single panel module is set up)
DECIDE             (optimization parameters established)
MAINSETUP          (set up PANDA2 analysis strategy)
PANDAOPT           (perform PANDA2 optimization of the
                    single panel module)
PANEL              (input file for multi-module panel is
                    setup. This is input to BOSOR4)
BOSOR4LOG          (activate BOSOR4 commands)
BOSORALL           (run BOSOR4 to get general buckling
                    of the multi-module panel)
BOSORPLOT          (get multi-module panel plot file)
```

BOSOR4LOG

This command activates the BOSOR4 command set. You need to use it after PANEL in order to get general instability of the entire panel width with every module treated as a multi-branched structure.

FORMAT:

```
BOSOR4LOG
```

EXAMPLE OF USE IN A RUNSTREAM:

```
PANDA0PT            (perform PANDA2 analysis)
PANEL               (input file for multi-module panel is
                     setup. This is input to BOSOR4)
BOSOR4LOG           (activate BOSOR4 commands)
BOSORALL            (run BOSOR4 to get general buckling
                     of the multi-module panel)
BOSORPLOT           (get multi-module panel plot file)
```

BOSORPLOT

Initiates batch run to set up BOSOR4 plot files.

FORMAT:

```
BOSORPLOT
```

EXAMPLE OF USE IN A RUNSTREAM:

```
PANDA0PT            (perform PANDA2 analysis)
PANEL               (input file for multi-module panel is
                     setup. This is input to BOSOR4)
BOSOR4LOG           (activate BOSOR4 commands)
BOSORALL            (run BOSOR4 to get general buckling
                     of the multi-module panel)
BOSORPLOT           (get multi-module panel plot file)
```

PLOT

This command is used to produce hard copies of plots at Lockheed Palo Alto Research Laboratories. You must have PARM.PLV and VECTOR.PLV files before you give this command or whatever the equivalent command is at your facility.

FORMAT:

```
PLOT
```

EXAMPLE OF USE IN A RUNSTREAM:

```
PANDA0PT            (perform PANDA2 analysis)
PANEL               (input file for multi-module panel is
                     setup. This is input to BOSOR4)
BOSOR4LOG           (activate BOSOR4 commands)
BOSORALL            (run BOSOR4 to get general buckling
                     of the multi-module panel)
BOSORPLOT           (get multi-module panel plot file)
PLOT                (obtain hard copies from PLV files)
```

BOSORALL

A batch run of the three BOSOR4 processors (pre, main, post) is initated for calculation of general instability of the multi-module panel. Output is stored in a file NAME.OUT;3 if the run terminated successfully. See the BOSOR4 help file for more details in case of unsuccessful termination. (To see the BOSOR4 help file, type the commands BOSOR4LOG, HELP4 COMMAND BOSORALL).

FORMAT:

BOSORALL

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN                (you establish a starting design)
SETUP                (an input file for BOSOR4 is set up)
BOSMODEL             (BOSOR4-type discrete model of the
                      single panel module is set up)
DECIDE               (optimization parameters established)
MAINSETUP            (set up PANDA2 analysis strategy)
PANDAOPT             (perform PANDA2 optimization of the
                      single panel module)
PANEL                (input file for multi-module panel is
                      setup. This is input to BOSOR4)
BOSOR4LOG            (activate BOSOR4 commands)
BOSORALL             (run BOSOR4 to get general buckling
                      of the multi-module panel)
BOSORPLOT            (get multi-module panel plot file)
```

CHANGE

You use the CHANGE command to change parameters without having to go back to BEGIN. The parameters are segregated into three groups:

1. parameters eligible to be decision variables;
2. parameters not eligible to be decision variables;
3. allowables (for example, max. strain)

Your interactive input is saved on a file, NAME.CHG, in which NAME is the same name you used for BEGIN, SETUP, etc. In future runs of this or a similar case, you can edit the file NAME.CHG, then give the command CHANGE, and tell the system that you are using an existing file. If you goof part way through a case, you can do a CONTROL Y, edit the NAME.CHG file, and re-issue the command CHANGE, telling the system that you are adding to an existing file. The system will answer all the questions up to your goof in a "batch" mode, then return control to you. You can then complete the interactive input.

FORMAT:

CHANGE

EXAMPLE OF USE IN A RUNSTREAM:

```
BEGIN                (you establish a starting design)
SETUP                (an input file for BOSOR4 is set up)
BOSMODEL             (BOSOR4-type discrete model set up)
DECIDE               (optimization parameters established)
MAINSETUP            (set up PANDA2 analysis strategy)
PANDAOPT             (perform PANDA2 optimization)
CHANGE               (change some parameters)
SETUP                (an input file for BOSOR4 is set up)
BOSMODEL             (BOSOR4-type discrete model set up)
MAINSETUP            (set up PANDA2 analysis strategy)
PANDAOPT             (perform PANDA2 optimization for
```

modified panel module)

CLEANUP

All files with the name NAME.* and *.PLV are deleted except NAME.BEG, NAME.DEC, and NAME.CHG.

****

V. PANDA2 FILES

GENERAL INFORMATION ON FILES....

Runs of PANDA2 generate many files. Most of these files have a name such as NAME.BEG or NAME.DEC, etc., in which NAME is a name that you assign the case. The most important of these files from the user's viewpoint are:

NAME.BEG NAME.DEC NAME.OPM NAME.OUT

ALL (NAME.ALL file) FILE TYPE: FORMATTED

NAME.ALL is generated by two processors, SETUP and PANEL. SETUP generates a NAME.ALL file that represents a single module of the panelPANEL generates a NAME.ALL file that represents a multi-module panel. The single-module NAME.ALL file is used by the PANDA2 system as input to the BOSMODEL processor. The multi-module NAME.ALL file is used by the PANDA2 system as input to the BOSORALL processor, which is a BOSOR4 processor.

BL1 (NAME.BL1 file) FILE TYPE: BINARY

NAME.BL1 contains contents of labelled common blocks of the BOSOR4 system of programs corresponding to the discretized model of the panel module.

BL2 (NAME.BL2 file) FILE TYPE: BINARY

NAME.BL2 contains contents of labelled common blocks of the BOSOR4 system of programs corresponding to the discretized model of the entire panel with smeared stiffeners.

DAT (NAME.DAT file) FILE TYPE: FORMATTED

NAME.DAT contains a copy of whatever interactive session is presently being performed. This file is generally not needed, but may be useful in case of accidental

distruction of the other file that contains the current interactive session (NAME.BEG, NAME.DEC, NAME.OPT, or NAME.CHG, for examples). Also, NAME.DAT provides a properly documented file corresponding to whatever interactive session you just completed. For example, you may have generated some input for NAME.BEG by VAX editing rather than by answering questions interactively. You probably did not bother to type the definitions corresponding to your new input. Therefore your VAX-edited NAME.BEG file will not be properly annotated. After completion of the BEGIN session and before you do anything else, you should replace the improperly annotated NAME.BEG file with the properly annotated NAME.DAT file.

DOC (NAME.DOC file) FILE TYPE: FORMATTED

NAME.DOC is a re-annotated copy of NAME.ALL. NAME.DOC is generated by the BOSOR preprocessor BOSORREAD. It represents a completely annotated input "deck" for BOSOR if execution of the preprocessor BOSORREAD was successful.

ERR (NAME.ERR file) FILE TYPE: FORMATTED

NAME.ERR contains list output. This file will not be present if BOSORALL ran successfully. If the case bombed, it will contain whatever output had been generated in the processor (pre-, main- or post-) that happened to be executing at the time the case bombed out. List this file for debugging purposes.

LAB (NAME.LAB file) FILE TYPE: FORMATTED

NAME.LAB contains labels for plots. This file must exist before the command BOSORPLOT is given.

OUT (NAME.OUT file) FILE TYPE: FORMATTED

NAME.OUT contains BOSOR4 output. The version number will be 3 if BOSORALL ran successfully to completion.

PLT (NAME.PLT file) FILE TYPE: BINARY

NAME.PLT contains data for plots. This file must exist before the command BOSORPLOT is given.

PLV (VECTR1.PLV and PARM.PLV files) TYPE: BINARY

*.PLV files contain data for plotter. These files are generated via the command BOSORPLOT. You can obtain hard copies of your plots by giving whatever command is appropriate at your facility. (PLOT at LPARL).

RN1 (NAME2.RN1 file) FILE TYPE: BINARY

NAME.RN1 contains BOSOR4 and PANDA2 data bases corresponding to the discretized model of the panel module.

RN2 (NAME.RN2 file) FILE TYPE: BINARY

NAME.RN2 contains BOSOR4 and PANDA2 data bases corresponding to the discretized model of the entire panel with smeared stiffeners.

NAM (NAME.NAM file) FILE TYPE: FORMATTED

NAME.NAM contains only the name of the PANDA2 case.

BEG (NAME.BEG file) FILE TYPE: FORMATTED

NAME.BEG contains a summary of the interactive session initiated with the command BEGIN (establish starting design). This file can be edited and used for future runs of the BEGIN processor, thus saving you the necessity of running through all the questions again interactively. Partial NAME.BEG files are also useful: If you make a mistake during the interactive session, you can terminate with a CONTROL Y, remove any erroneous lines at the end of the NAME.BEG file, and issue the command BEGIN again. Upon your indicating that you want to add to an existing file, the BEGIN processor will use the data in the NAME.BEG file first and return to the interactive mode at the end of this file. The data that you next provide will be added to the NAME.BEG file.

CBL (NAME.CBL file) FILE TYPE: BINARY

NAME.CBL contains contents of labelled common blocks of the PANDA2 system of programs.

DEC (NAME.DEC file) FILE TYPE: FORMATTED

NAME.DEC contains a summary of the interactive session initiated with the command DECIDE (establish decision variables). This file can be edited and used for future runs of the DECIDE processor, thus saving you the necessity of running through all the questions again

interactively. Partial NAME.DEC files are also useful: If you make a mistake during the interactive session, you can terminate with a CONTROL Y, remove any erroneous lines at the end of the NAME.DEC file, and issue the command DECIDE again. Upon your indicating that you want to add to an existing file, the DECIDE processor will use the data in the NAME.DEC file first and return to the interactive mode at the end of this file. The data that you next provide will be added to the NAME.DEC file.

OPB (NAME.OPB file) FILE TYPE: FORMATTED

NAME.OPB contains a summary of output from BEGIN. It is a good idea to list this file before proceeding with any optimization in order to insure that the case is as you intend it to be.

OPD (NAME.OPD file) FILE TYPE: FORMATTED

NAME.OPD contains a summary of output from DECIDE. It is a good idea to list this file before proceeding with any optimization in order to insure that the case is as you intend it to be.

OPM (NAME.OPM file) FILE TYPE: FORMATTED

NAME.OPM contains output from the PANDA2 mainprocessor, PANDAOPT. This is the most important file of all from the user's viewpoint, because it contains warping evaluation, bowing due to loading, general and local bifurcation buckling loads, the postbuckled state of strain and stress resultants, identification of maximum strains and strain margins, and the evolution of the design during optimization.

NOTE: make sure you print this file out after every PANDAOPT batch run. The PANDA2 system only retains the most recent version of NAME.OPM.

OPT (NAME.OPT file) FILE TYPE: FORMATTED

NAME.OPT contains a summary of the short interactive session following your command MAINSETUP. The contents of this file are listed at the beginning of NAME.OPM. The file is always saved, and you do not have to go back to MAINSETUP every time you want to launch a batch run via your command PANDAOPT. For example, after looking at the results of the first set of design iterations in the file NAME.OPM, you may want to do another set of design iterations without changing any strategy parameters. If so, you can simply type PANDAOPT again. The PANDA2 system will use the strategy parameters in the file NAME.OPT.

CHG (NAME.CHG file) FILE TYPE: FORMATTED

NAME.CHG contains a summary of the interactive session initiated with the command CHANGE (modify design). This file can be edited and used for future runs of the CHANGE processor, thus saving you the necessity of running through all the questions again interactively. Partial NAME.CHG files are also useful: If you make a mistake during the interactive session, you can terminate with a CONTROL Y, remove any erroneous lines at the end of the NAME.CHG file, and issue the command CHANGE again. Upon your indicating that you want to add to an existing file, the CHANGE processor will use the data in the NAME.CHG file first and return to the interactive mode at the end of this file. The data that you next provide will be added to the NAME.CHG file.

OPC (NAME.OPC file) FILE TYPE: FORMATTED

NAME.OPC contains a summary of output from CHANGE. It is a good idea to list this file before proceeding with any optimization in order to insure that the case is as you intend it to be. Also, if you have made substantial changes, it is a good idea before doing any optimization, to run a simple buckling analysis with the new design fixed.

****

VI. PANDA2 DOCUMENTATION

DOCUMENTATION FOR PANDA2....

(A) BACKGROUND MATERIAL ON SHELL BUCKLING....

COMPUTERIZED BUCKLING ANALYSIS OF SHELLS, Nijhoff, The Netherlands, 1985 (book by D. Bushnell)

"Buckling of Shells--Pitfall for Designers", AIAA Journal, Vol. 19, No. 9, September, 1981, pp. 1183-1226

"Computerized Analysis of Shells--Governing Equations", Computers & Structures, Vol. 18, pp.471-536, 1984

(B) MATERIAL NEEDED FOR USE OF THE BOSOR4 COMPUTER PROGRAM...

"BOSOR4--Program for stress, buckling, and vibration of complex shells of revolution," in STRUCTURAL MECHANICS SOFTWARE SERIES, vol. 1, Univ. Press of Virginia, pp11-143, 1977

BOSOR4ST.ORY (file contained herein that describes the VAX version of BOSOR4)

HELP4 (interactive help file, data in BOSOR4.HLP)

(C) MATERIAL NEEDED FOR USE OF THE PANDA2 COMPUTER PROGRAM...

"PANDA--Interactive program for minimum weight design of stiffened cylindrical panels and shells", Computers and structures, Vol. 16, pp 167-185, 1983

"PANDA2 - Program for minimum weight design of stiffened, composite, locally buckled panels", Lockheed Missiles and Space Co. Report LMSC-D067175, revised, November, 1986. To appear in Computers and Structures Journal, probably in 1987.

"Stress, buckling and vibration of prismatic shells", AIAA Journal, Vol. 9, No. 10, pp 2004-2013, Oct. 1971

PANDA2ST.ORY (File that describes PANDA2 and how to use it. You should obtain a list of this.)

TUTORIAL runs of the processors BEGIN, DECIDE, MAINSETUP, PANDAOPT, and CHANGE

HELPAN (PANDA2 HELP file)

SAMPLE.CAS (Sample case: a blade-stiffened panel under pure axial compression and under combined axial compression and in-plane shear)

HOWTO.RUN (A file that describes how tape was made.)

****

VII. RECOMPILING SOURCE FILES, REGENERATING LIBRARIES, RE-LINKING

HOW TO RECOMPILE AND RELINK PANDA2 ELEMENTS

RECOMPILE:

PANDA2 is divided into a number of source code libraries, as follows:

ARRAYS.NEW;1
B4READ.NEW;1
BEGIN.NEW;1
BOSPAN.NEW;1
BUCKLE.NEW;3
BUCPAN1.NEW;1
BUCPAN2.NEW;1
CHANGE.NEW;1
CONMAN.NEW;1
CONMIN.NEW;1
DECIDE.NEW;1
EBAND2.NEW;1
FORCES.NEW;1
GASP.NEW;1
GASPOLD.NEW;1
GETCIJ.NEW;1
GLOBST.NEW;1
HELPAN.NEW;1
KOITER.NEW;1
MAIN.NEW;1

```
MAINSETUP.NEW;2
MODE.NEW;1
NEWTPN.NEW;1
PANCOM.NEW;1
PROMPTER.NEW;1
RFIVE.NEW;1
SETUP.NEW;1
SMRCIJ.NEW;1
STOGET.NEW;1
STRAIN.NEW;1
STRUCT.NEW;1
UTIL.NEW;1
```

If you receive notice of a bug, or if you wish to modify the PANDA2 source code for any reason, then you do the following:

1. Edit the appropriate source code library, using standard editing procedures.

2. Compile the library: $FOR NAME.NEW in which NAME is one of the library names listed above.

3. After successful compilation, give the command: @GETLIB
   The GETLIB procedure is as follows:

```
$! This procedure establishes a compressed library from a
$! file of object decks and deletes the object file.
$!
$      INQUIRE  P1  "GIVE NAME OF LIBRARY FILE"
$      LIB/CREATE  'P1'.OLB  'P1'.OBJ
$      LIB/COM     'P1'.OLB
$      COPY        'P1'.OLB  *.*
$      PURGE       'P1'.OLB
$      DELETE      'P1'.OBJ;*
```

   When GETLIB asks for the name of the library file, type the appropriate name from the list given above (e.g. STRUCT).

4. When the NAME.OLB file has been successfully generated, and after you have done all your recompilations, give the appropriate linking command or commands. The various linking procedures in the PANDA2 system are as follows:

```
LINKBEG.COM;1
LINKBOSPAN.COM;1
LINKBREADVAX.COM;1
LINKCHANGE.COM;1
LINKDEC.COM;1
LINKGLOBVAX.COM;1
LINKHELP.COM;1
LINKMAINSETUP.COM;1
LINKMAINVAX.COM;1
LINKSETUP.COM;1
```

EXAMPLE:

```
$FOR MAINSETUP.NEW
$@GETLIB
 GIVE NAME OF LIBRARY FILE:  MAINSETUP
$@LINKMAINSETUP
$PURGE *.EXE
```

****

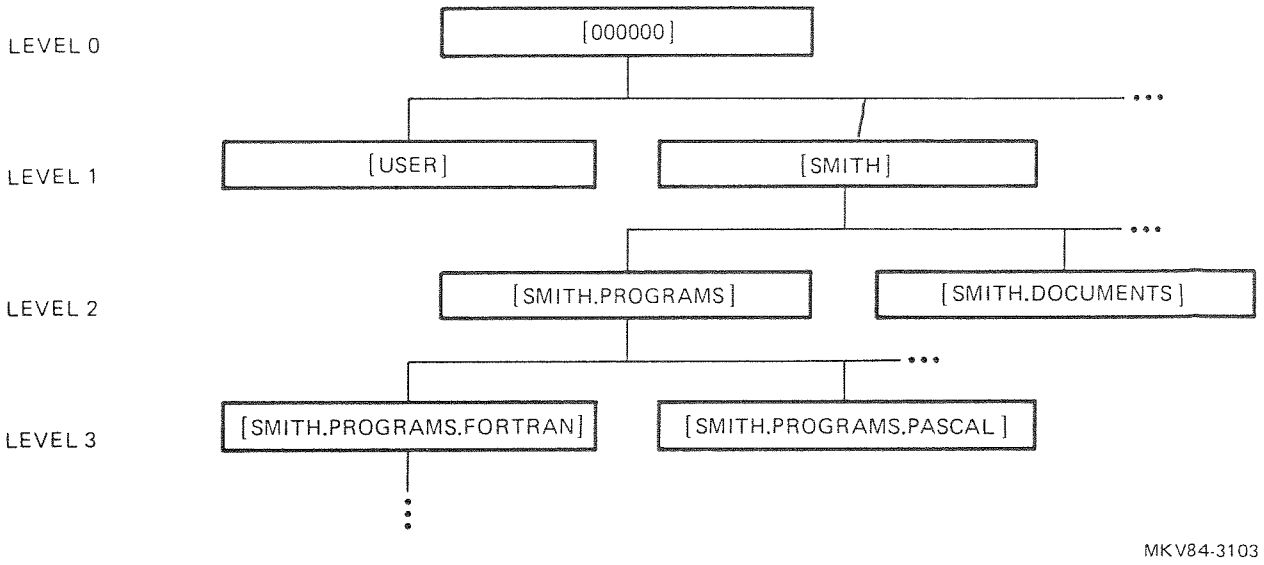


Figure 3 A Directory Hierarchy

```
$ DIRECTORY

Directory $DISK1:[SMITH]

DOCUMENTS.DIR;1      FILE.TXT;1          LOGIN.COM;2          MAIL.MAI;1
MYFILE.TXT;1         PROGRAMS.DIR;1

Total of 6 files.
$
```

Example 2 Directories Catalog Subdirectories and Other Files

FILE SPECIFICATION FORMAT AND SYNTAX

NODE::DEVICE:[DIRECTORY]NAME.TYPE;VERSION NUMBER

	description	# of characters	default
NODE	computer name	1-6	current computer logged into
DEVICE	disk or tape	1-255	$DISK1
DIRECTORY	directory name	1-39	current directory
NAME	filename	0-39	none
TYPE	file extension	0-9	depends on command: conventions- print - .LIS FORTRAN- .FOR command file - .COM run file - .EXE
VERSION NUMBER		1-32767	latest (highest) version by most commands

---

wildcard characters

*	takes the place of one or more characters
%	takes the place of one character

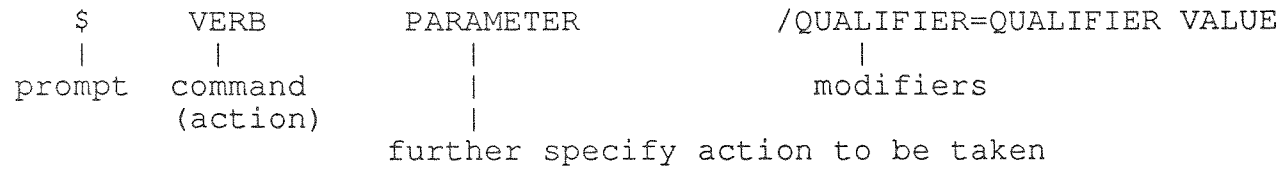
EXAMPLE:

for the following files:

FILE1.TXT;1	FILE1.TXT;2
FILE2.TXT;1	TEXTFILE.TXT;1
MYFILE.TXT;1	MYFILE.TXT;2

the wildcards:	will produce:
.;*	all files
.;2	FILE1.TXT;2 MYFILE.TXT;2
FILE%.TXT	FILE1.TXT;1 FILE1.TXT;2 FILE2.TXT;1 FILE2.TXT;2
*FILE	TEXTFILE.TXT;1 MYFILE.TXT;1 MYFILE.TXT;2

DCL COMMAND FORMAT AND SYNTAX



the verb and paramete are separated by at least blank

if the parameter is a list, separate by commas

qualifiers may be positional

examples:

$PRINT myfile,yourfile /COPIES=2	will print 1 copy of myfile and 2 copies of yourfile
$PRINT/COPIES=2 myfile,yourfile	prints 2 copies of each

commands can be abbreviated to at least (most) 4 characters

use a hyphen to continue commands to the next line

CONTROLLING TERMINAL OUTPUT

CTRL S	to halt output
CTRL Q	to continue output

CONTROLLING COMMAND OR PROGRAM EXECUTION

CTRL Y	halt execution

DCL COMMANDS

COMMAND	REQUIRED PARAMETERS	OPTIONAL PARAMETERS	DESCRIPTION
DIRectory			list files in default directory
		file-spec	list files w/file-spec
Type	filename		lists filename contents to screen
COPY	from-filename to_filename		creates a new copy of a file
RENAme	from-filename to-filename		changes the name of a file
SEARch	filename(s) "a string"		displays a list of the files containing the desired string
APPEnd	from-filename to-filename		moves a copy of the 1st file listed to the bottom of the 2nd
DIFFerence	file1 file2		displays lines in one file that are different from the other file
		/PARALLEL	lists side by side
SHOW	PROCESS		
	DEFAULT		current default disk and directory
	USERS		who currently logged in
	TIME		display date and time
		/ALL	complete information
DELete	filename;#		delete a file
		/CONFIRM	gives a confirmation prompt
PURGE	filename		delete all by latest version
	<blank>		does for all files
		/KEEP=2	keeps the 2 latest versions
HELP		command	
RECALL		#	recalls previous # command
		/ALL	recalls last 20 commands
SET	PASSWORD		change password
	DEFAULT pathname		move to pathname
	TERM/UNKNOWN		set terminal type
	VT100		

DIRECTORIES

CReate/DIRectory pathname — create a subdirectory

where pathname is enclosed in brackets
directory names separated by periods

example:

```
CR/DIR $DISK1:[HERSTROM.SUB1]
   or
       [HERSTROM.SUB1]
   or
       [.SUB1]    if located in HERSTROM directory
```

Some commands:

DIR [-] ~~move up~~ list files one level up [note: a SET DEF [-] will move you up one level]

DIR [...] list files and directories from current directory and all files and directories below

DIR [PATHNAME...] lists files and directories from pathname and below

The following commands are the only times you actually use the .DIR extension:

DIR/PROT directoryname.dir
display protection on directory file
(owner does not have delete access)

SET PROT=(O:D) directory.dir
give owner delete privilege

DELETE directory.dir
deletes a directory, all files and directories within
the directory must have been deleted previously

WARNING: The system will let set default to a directory that doesn't exist. You will not get an error until you try to use it.

FILE PROTECTION

USERS ARE DIVIDED INTO GROUPS	DEFAULT PROTECTION CODE
SYSTEM	RWED
OWNER	RWED
GROUP	RE
WORLD	-none-

CATAGORIES OF ACCESS
- READ
- WRITE
- EXECUTE
- DELETE

Some commands to show and set the protection:

SHow PROTection	displays default used by system
DIR/PROT filename	show protection on a particular file
SET PROT=(code)/DEFault	change the default protection, this is good for the current loggin session
SET PROT=(code) filename	change the protection on particular file
SET PROT filename	change the protection back to the system default

where code = catagories of access
example:
(S:REWD,O:REWD,G)
including the G gives no access
excluding the W leaves access as it was

MAIL

allows you to send messages to other users

$MAIL	invokes mail
MAIL>	mail prompt
SEND	sending a message
READ	read a message
FILE "folder name"	set up folders for mail messages
DIR/FOLDERS	give list of folders
SELECT "folder name"	
DELETE message-number	if message number is omitted, deletes the message currently reading
HELP	
EXIT	exit from mail

CREATING AN EXECUTABLE

complile, link, and run a FORTRAN program

```
$FORTRAN TEST[.FOR]          (produces TEST.OBJ)
$LINK TEST[.OBJ]             (produces TEST.EXE)
$RUN TEST[.EXE]              (executes the program)
```

CREATING A COMMAND FILE

use the .COM extension, then
@filename will execute the command file

begin every line with a $
for comments use a ! after the $
default parameters P1,P2,...P8 (use quotes around these parameters when symbols no normally used in the location, unless it is the first thing in the line)

EXAMPLE:

```
CR RUNFOR.COM
$! compile, link and run a FORTRAN PROGRAM
$FORTRAN 'P1'
$LINK 'P1'
$RUN 'P1'
$EXIT
[control] Z

@RUNFOR filename
```

LOGICAL NAMES - use in place of a file specification

```
SHOW LOGICAL logical-name                lists logical name meaning

DEFINE logical-name [pathname]
                    [pathname]filename    assign a value to a logical name
```

SYMBOLS - represents a numeric, character or logical value

```
SYMBOL =[=] "value"            assign a value to a symbol
       where  =  local
              == global
       or
         ==: value             (don't need the quotes,then)

DELETE/SYMBOL symbol            delete a symbol
```

---

to print a file on the line printer attached to CALMA (NODE C):

```
COPY filename C::TXA7:
```

to print a file on the postscript laser printer

```
PS filename
```

LARCNET NODE NAME : SCB210B

```
$! my login commands
$!
$! set terminal for vt100
$!
$set term/dev=vt100
$!
$!  enable the user of defined keypad keys
$!
$assign/process sys$login:edtini.edt $DISK1:[HERSTROM]edtini.edt
$!
$!  define keypad keys
$!
$define/key f17 "directory"/terminate
$define/key f20 "lisp"/terminate
$!
$! define symbols
$!
$sd == "set default"
$delete == "delete/confirm"
$time == "show time"
$priv :== @$disk1:[herstrom]priv
$lisp == "run lisp"
$home == "set def $disk1:[herstrom]"
$!
$!  set logical names
$!
$assign dua0:[000000.lisp]flisp.exe lisp
$assign dua0:[000000.lisp]loop.l loop.l
$assign dua0:[000000.lisp]macs.l macs.l
```

THE EDT EDITOR

EDIT filename — invokes the editor

* — prompt for the editor

commands for screen editor

Change — puts in screen editor mode
(only works for vt100 emulation)

to put the TEK4109 into VT100 mode:

hit setup
*CODE EDIT <cr>
hit setup

to return to TEK mode:

hit setup
*CODE TEK
hit setup

other commands:

help
exit - exit saving changes
quit - exit with no saving of changes

Creating a file:

CReate filename — creates on empty file and puts you in input mode

to exit : CONTROL Z

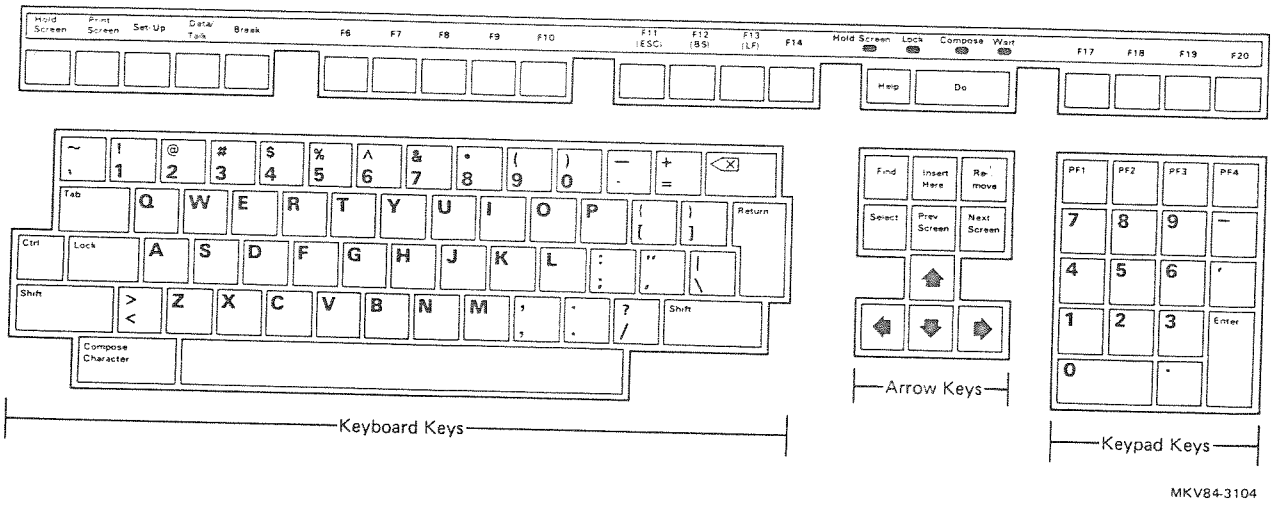


Figure 1 Keyboard for VT200-Series Terminals

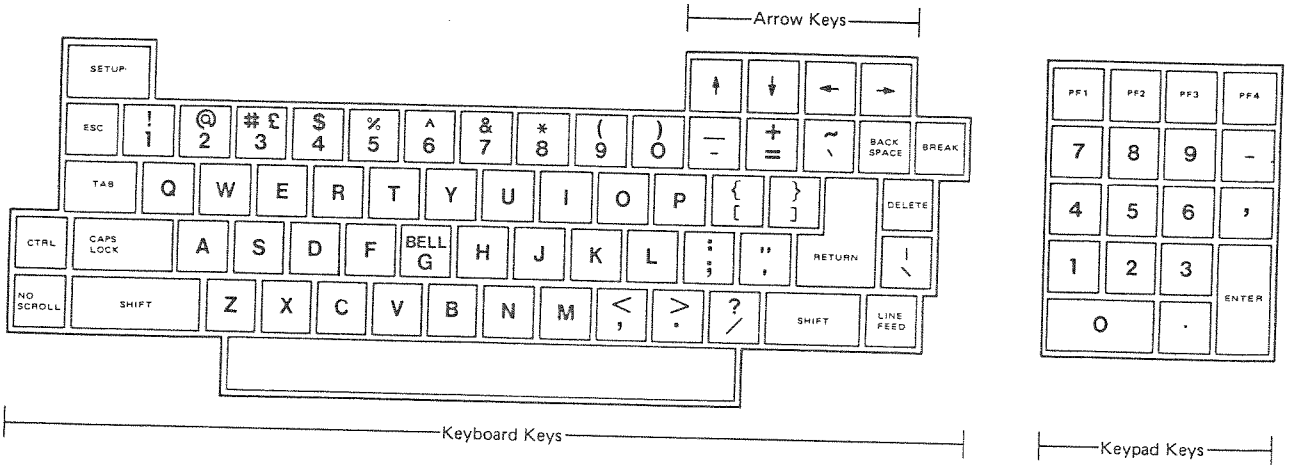


Figure 2 Keyboard for VT100-Series Terminals

Table 8 Keys Used in Editing Command Lines

Function	Key
Move the cursor to the beginning of the command line	BACK SPACE key, CTRL/H, or F12
Move the cursor to the end of the line	CTRL/E
Move the cursor one character to the left	Left arrow key or CTRL/D
Move the cursor one character to the right	Right arrow key or CTRL/F
Delete from the beginning of the line to the cursor	CTRL/U
Delete the character to the left of the cursor	DELETE key <X
Delete the word to the left of the cursor	CTRL/J, F13, or LINE FEED key
Switch between overstrike mode and insert mode	CTRL/A or F14 (as soon as hit <return> returns to overstrike mode)

USING THE TEXT EDITOR

- To invoke the EDT Editor, use the format:

  EDIT filename.typ

- If the file does not exist, EDT will create it.

- EDT does not provide a default file type. If you do not specify one, the file type is null.

- Terminating an EDT session
  - The QUIT command terminates EDT without creating an output file.
  - The EXIT command creates an output file from the contents of the text buffer, then terminates the session.
  - To override the default output file name, specify a new one with:
    EXIT newfilename.typ

- Run the utility TEACHME to learn the screen editor (reference manuals are also available). Possible only on VT100 terminals.

- Line mode editing is possible on any type of terminal.

Table 2-1: Summary of Line Editing Commands

Command	Function
CHANGE [range]	Invokes character mode editing for specified buffer
CLEAR	Deletes the contents of a text buffer
COPY [range1] TO [range2] [/QUERY]	Copies lines specified by range1 to a location in an EDT buffer specified by range2; does not delete lines from original location
DEFINE { KEY | MACRO }	Defines a new or revised key function for character mode editing, or defines a macro name
DELETE [range] [/QUERY]	Deletes a specified line or lines
EXIT [file-spec]	Terminates EDT, saving the contents of the text buffer MAIN as the output file
FILL [range]	Reformats a block of text, filling lines with the maximum number of full words without exceeding the right margin
FIND range	Establishes the first line in range as the current line
HELP [topic ...]	Displays information on the specified EDT command or function
INCLUDE file-spec [range]	Copies an external file to a location in a text buffer specified by range
INSERT [range]	Opens a text buffer for the insertion of text at the location specified by range
MOVE [range1] TO [range2] [/QUERY]	Moves lines specified by range1 to the location specified by range2, deleting the lines from the source location
PRINT file-spec [range]	Creates a listing file with the specified file name
QUIT [/SAVE]	Terminates EDT without creating an output file, optionally

Table 2-1 (Cont.): Summary of Line Editing Commands

Command	Function
REPLACE [range]	Deletes specified lines from a text buffer and leaves the buffer open for insertion of text
RESEQUENCE [range]	Assigns new line numbers to a range of lines
SET [parameter]	Sets a variety of editor operating parameters
SET[[NO]NUMBER]	Enables/disables the display of line numbers
SHOW [parameter]	Displays specified editor operating parameters
SUBSTITUTE /string1/string2/[range] [/QUERY]	Replaces string1 with string 2, either in the current line or in the specified range
[SUBSTITUTE] NEXT [/string1/string2]	Replaces string1 with string2, based either on the strings specified or on the previous SUBSTITUTE command
TAB ADJUST [-]n [range]	Shifts each line in a range of lines by a specified number of logical tab stops
[TYPE] [range]	Displays specified lines and makes the first line in range the current line; the default command
WRITE file-spec [range]	Moves a copy of specified text from a buffer to a file

Table 2-2: Single-Line Range Specifications

Specification	Meaning
.	The current line
number	The line specified by the number
'string' or "string"	The next line containing the string you specify
-'string' or -"string"	The preceding line containing the string you specify
[range] { + | - } [number]	The line that is the specified number of lines after (or before, if minus) the single line specified by range (range defaults to the current line; number defaults to 1)
BEGIN	The first line in the text buffer
END	An empty line (designated by [EOB]) following the last line of text in the text buffer

Examples:

Specification	Meaning
20.6	The line numbered 20.6
"EQUIVALENCE"	The next line that contains the string *EQUIVALENCE*
"-D_FLOATING COMPLEX"	The first preceding line that contains the string *D_FLOATING COMPLEX*
-6	The line six lines before the current line
"SUBROUTINE"+4	The line four lines after the line that contains the string *SUBROUTINE*

Table 2-3: Multiple-Line Range Specifications

Specification	Meaning
[range1] { : | THRU } [range2]	The set of lines from range1 through range2, which are single line range specifications (the default for both range1 and range2 is the current line)
[range] { # | FOR } number	The specified number of lines beginning with the single line specified by range (the default for range is the current line)
BEFORE	All lines in the buffer that precede the current line
REST	The current line and all lines in the buffer that follow it
WHOLE	The entire buffer
range, range... or range AND range AND...	All lines specified by each single line range
[range] ALL 'string'	All lines in the range containing the specified string (the default for range is the entire buffer)

Examples:

Specification	Meaning
2:6.5	Lines 2 through 6.5, inclusive
'STRUCTURE'#5	The line containing the string *STRUCTURE* and the four lines following it, for a total of five lines
.-10:.	The line 10 lines before the current line through the current line, inclusive
[illegible] ALL 'LOAD'	All lines from [illegible] through [illegible] the

^ | UP	DOWN | v	<—— LEFT	——> RIGHT

LETE	Delete character
NEFEED	Delete to beginning of word
CKSPACE	Backup to beginning of line
RL/A	Compute tab level
RL/D	Decrease tab level
RL/E	Increase tab level
RL/K	Define key
RL/R	Refresh screen
RL/T	Adjust tabs
RL/U	Delete to beginning of line
RL/W	Refresh screen
RL/Z	Exit to line mode

ess a key for help on that key.
exit, press the spacebar.

GOLD	HELP	FNDNXT FIND	DEL L UND L
PAGE COMMAND	SECT FILL	APPEND REPLACE	DEL W UND W
ADVANCE BOTTOM	BACKUP TOP	CUT PASTE	DEL C UND C
WORD CHNGCASE	EOL DEL EOL	CHAR SPECINS	ENTER SUBS
LINE OPEN LINE		SELECT RESET	