Honeywell

Control Language Process Manager Data Entry

PM11-500

Honeywell

Implementation
Process Manager - 2

Control Language Process Manager Data Entry

PM11-500
Release 500
9/95

Copyright, Trademarks, and Notices

Revision 01 - September 1, 1995

About This Publication

This publication provides instructions for the following tasks:

- Creating CL/PM source files, and entering and editing data in the source files
- Using the Engineering Personality's Utilities to manage and manipulate files related to CL/PM
- Using the CL/PM Compiler commands and options that are part of the COMMAND PROCESSOR activity on the Engineering Main Menu

This publication supports TDC 3000^{X} software releases 500.

Table of Contents

INTRODUCTION
Section 1

This section provides

- *A description of the intended use for this publication.*
- *References to other publications needed or useful for compiling and loading CL/PM sequences.*

1.1 WHO THIS PUBLICATION IS INTENDED FOR

This publication is prepared for use by people such as process engineers, control-system engineers, and application engineers who need to install custom functions in a TDC 3000X System, using the Control Language/Process Manager (CL/PM).

1.2 HOW TO USE THIS PUBLICATION

Use this publication when

- You need to determine what work must be accomplished before you can install or modify a CL/PM sequence program. Section 2 should be especially useful for this purpose.
- You are ready to install a new CL/PM sequence program or to change an existing CL/PM sequence program.

The expected use of Sections 2 through 5 of this publication is as follows:

- **Section 2 – Prerequisites**—Use this section to determine what must be available to install or to change CL/PM sequence programs and what startup tasks must be completed.
- **Section 3 – Typical CL/PM Data Entry Sessions**—Use this section to review the major steps in installing or changing a CL/PM sequence program and the recommended order of those steps.
- **Section 4 – Engineer's Keyboard**—Use this section when you don't know or are not sure of the effect of pressing a key on the Engineer's Keyboard.
- **Section 5 – Commands**—Use this section to determine the detailed steps in executing the commands used to install or to change CL/PM sequence programs, and for a description of the functions of the CL/PM compiler.
- **Section 6 – Error Indications and Recovery**—This section provides guidance in recovering from human errors.

- **Index**—An alphabetized list of CL/PM data entry topics with references to paragraph numbers in Sections 1 through 6.

Before using the procedures in Sections 3 through 5 of this publication, you should review the publications mentioned under 1.3.

1.3 RELATED PUBLICATIONS

The following are publications that contain reference information related to CL/PM data entry, or provide procedures that are related to CL/PM data entry.

- *Control Language/Process Manager Reference Manual, PM27-510,* in the *Implementation/Process Manager - 2* binder. This manual defines the Control Language, including its environment. The language's elements, syntax, and rules are defined. Also defined are the relationships of the parts of CL/PM sequence programs and their sizes and limitations.

- **Control Functions**—This is a set of two publications, one that covers data acquisition and control functions that are independent of the nodes that participate in the functions, and one for PMMs.

  System Control Functions, SW09-501, in the *Implementation/Startup & Reconfiguration - 2* binder

  Process Manager Control Functions and Algorithms, PM09-500, in the *Implementation/Process Manager - 1* binder

- *Engineer's Reference Manual, SW09-505*, in the *Implementation/Startup & Reconfiguration - 2* binder. Provides guidance and tips that are useful in designing and implementing the system configuration, and in starting up the TDC 3000^X System.

- **Parameter References**—This publication provides information about all standard parameters, including parameter names, data types, value ranges, and access levels.

  Process Manager Parameter Reference Dictionary, PM09-540, in the *Implementation/Process Manager - 2* binder

- *System Startup Guide - Cartridge Drive, SW11-504*, in the *Implementation/Startup & Reconfiguration - 1* binder. We recommend that you use this guide the first time you use any of the activities on the Engineering Main Menu. It provides step-by-step procedures for all available configuration activities. This guide defines 31 system startup tasks. Task 28 is Build, Compile, and Load CL/PM Sequence programs.

- *Data Entity Builder Manual, SW11-511*, in the *Implementation/Engineering Operations - 1* binder. This publication describes the use of the Data Entity Builder (DEB). The DEB is used for eight of the activities on the Engineering Main Menu. Of those eight, data-point building is the most important to CL/PM data entry, because it is used to build the data points that CL/PM sequence programs work with.

- *Command Processor Operation, SW11-507,* in the *Implementation/Startup & Reconfiguration - 1* binder. This publication provides instructions for the use of all of the functions available through the Command Processor selection on the Engineering Main Menu. Utilities are used to set up default volume pathnames, to manage volumes and files, including those used by CL/PM, to catalog and list the content of volumes and files, and to display and print the content of CL/PM files.

- *Text Editor Operation, SW11-506,* in the *Implementation/Engineering Operations - 3* binder. This publication defines how to use the Text Editor. It is used to create CL/PM source files on user volumes, and to enter and edit information in those files.

- *Process Operations Manual, SW11-501*, in the *Operation/Process Operations* Binder. This manual provides instructions for use of the Universal Station's Operator personality. CL/PM (sequence) programs are loaded in the PMM through this personality.

PREREQUISITES FOR A CL/PM DATA ENTRY SESSION
Section 2

Use this section to determine what must be available to install or to change CL/PM programs, and what startup tasks must be completed.

2.1 PRECEDING SYSTEM STARTUP TASKS COMPLETED

You can create CL/PM source files, enter data in them, and edit them at any time the Universal Station's Universal Personality or Engineering Personality is loaded. CL/PM programs can be compiled and loaded ONLY after configuration and loading of PMMs (System Startup Task 32) is complete. See the Tasks shown in the *System Startup Guide* for further information.

CL/PM data entry consists of creating source files, compiling them, and loading them into PMMs, where they will be executed.

A sequence program (the .PO object file, which must be on Volume &Enn—refer to heading 3.2.2) is loaded into a PMM Process Module Data Point (a sequence slot in the PMM) from the Process Module Detail display in the Universal Station's Process Operator personality. To load a CL/PM program, at least one US must be running the Operator Personality, so System Startup Task 30 must be completed.

2.2 USER VOLUMES ESTABLISHED WITH SUFFICIENT CAPACITY

You will store your CL/PM source files on user-volume removable media (floppies or cartridge disks) or in user volumes on an HM. The CL Compiler creates listing files, error-listing files, and object files on the volume that contains the source file (Table 2-1 lists all of the volumes used in CL/PM data entry). You will find it easiest to use the HM.

Whether you use removable media or the HM, you must configure one or more user volumes with sufficient capacity to store all of your source files and the Compiler-generated files.

Unless you create an unusually large number of CL programs, the CL user volumes will be much smaller than those that contain IDFs with DEB-built entities. Guidelines for estimating the size of CL-user volumes are provided under 7.2 in the *Engineer's Reference Manual*.

You can use Task 4 (removable media) or Task 10 (HM) to configure your user volumes (see heading 3.2.2). User volumes should be configured in Task 10 during the initial startup, but the HM volume configuration can be changed to include user volumes by using the on-line reconfiguration procedures (see *Network Data Entry* in the *Implementation/Startup & Reconfiguration - 1* binder). For a more detailed explanation of floppy formatting, refer to *Command Processor Operation* in the *Implementation/Engineering Operations - 1* binder.

We recommend that you periodically copy any CL-user files you have on an HM to removable media so that you can recover them should something cause the data on the HM to be lost. Use the Copy command to do this.

2.3 ENGINEERING PERSONALITY RUNNING AND NEEDED VOLUMES AVAILABLE

All of the activities described in this publication take place at a Universal Station with the Engineering Personality running. To begin any of these activities, the Engineering Main Menu must be on the screen. To load the Engineering Personality and to get the Main Menu on the screen, you can use Task 2 of the *System Startup Guide*, or refer to the *Engineer's Digest* (pocket guide).

Table 2-1 lists all of the volumes and files involved in installing and changing CL/PM programs. These volumes should be available on an HM or on removable media as needed. It is easier to have them on an HM, so mounting and dismounting disks isn't necessary.

Table 2-1 — Volumes and Files Used in Installing and Changing CL/PM Programs

Volume	File Name and Suffix	Use
&OV1	(Several)	Engineering Overlay 1—contains the Text Editor, Utilities, and Data Entity Builder (DEB)
&OV2	(Several)	Engineering Overlay 2--contains the CL Compiler
&NMG	(Several)	Contains the NIM GDFs
&Enn	(see 3.9)	Contains CL/PM object (.PO) files
CL	(filename).CL	CL/PM source file (created by the user)
"	(uppssiii).PO	CL/PM object file (created by the Compiler)
"	(filename).LS	CL/PM listing file (created by the Compiler)
"	(filename).LE	CL/PM error listing file (created by the Compiler)

Note: CL is a user volume that can have any user-defined name. "CL" is the default name that appears on the Modify Default Volume Path Names display. You can change the name on that display to the name of your user volume.

2.3.1 Universal Station and Volume Handling Guidelines

CL data entry requires a Universal Station that has an Engineer's Keyboard. It's useful to have a printer connected to the US so that you can make printed records of your work. If you are using removable media, the US should have two drives connected to it, so that the transfers will be faster than if they have to go over the LCN.

Execution of any command that can change the system database or the process database requires the key switch on the Operator Keyboard to be in the ENGR position. It is easiest to check the switch when you start a session and leave it in ENGR until you are finished.

It is very helpful to have another Universal Station in the same console with the Operator Personality running. You can then look at the detail displays for the points associated with your CL/PM programs, to verify that parameter values mentioned in this publication are as described, and to see the effects of your work.

It is easiest to have one or more user volumes for your CL/PM files (see Table 2-1) on an HM (Startup Task 10) and to have the complete software complement on an HM (Startup Task 16). If you use removable media, it's easiest to use one user volume for all CL/PM files, and you need to mount and dismount disks, as needed. For example,

- When using the Text Editor or Utilities, &OV1 should be in one drive and your user volume in the other.
- When you enter CL from the COMMAND PROCESSOR to use the CL Compiler, &OV2 should be in one drive and your user volume in the other.

Before you begin any activity on the Engineering Main Menu, you should select SUPPORT UTILITIES and check the Modify Default Volume Path Name display to see that the correct sources ($Fn or NET) and the correct volume names are entered. If not, change and then press ENTER.

2.4 PREREQUISITES FOR A CL/PM COMPILE AND LOAD SESSION

Before CL/PM programs can be compiled the following must be true:

A. The **PMM Node Configuration** must be built and loaded (System Startup Task 32). This entity must specify a nonzero number of process module slots to be able to run sequences.

B. If the compilation of the CL/PM program will add words to the **NIM Library**, there must be enough space left in that entity.

C. The **Process Module Data Point,** to which the sequence will be loaded, plus any other data points that the sequence references by entity name, must be built and loaded.

Before CL/PM programs can be loaded, the following must be true:

A. The PMM that will receive the CL/PM program must be loaded and running with the Process Module state of OFF. Any PMMs that contain Data Points that are referred to by the CL/PM program should be in OK state in order to avoid run time errors.

B. There must be sufficient sequence memory in the slot to accept the CL/PM program.

C. The preparations described under 3.2.2 must be completed; this includes ensuring that the .PO (object file) is in volume &Enn, where nn = UCN number the PMM is on (UCN number of the bound data point).

TYPICAL CL/PM DATA ENTRY SESSIONS
Section 3

3.1 HOW TO USE THIS SECTION

Typical CL/PM data entry sessions are described in this section for these two reasons:

- To provide an overview of typical sessions for new users.
- To define the order in which we recommend you accomplish the major steps.

References to detailed instructions for these major steps are provided.

3.2 INSTALLING A CL/PM SEQUENCE PROGRAM

Use the following major steps to install a CL/PM sequence program or to change one.

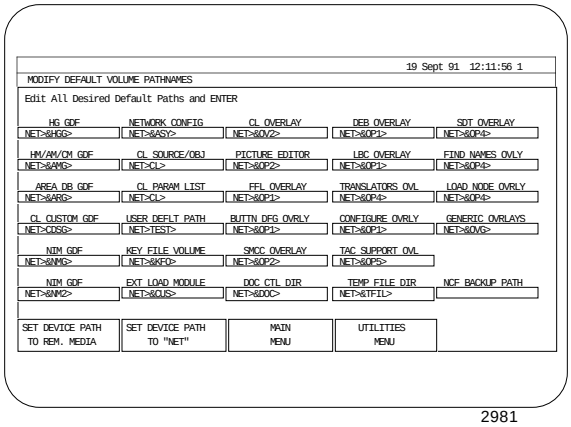
1. From the Engineering Main Menu, select SUPPORT UTILITIES and then select MODIFY VOLUME PATHS. On the Modify Default Volume Path Names Display, check that the pathnames for the following volumes are correct (see Table 2-1):

   CL SOURCE/OBJ
   examples; $F2>CL> or NET>CL>

   CL OVERLAY
   examples; $F1>&OV2>
   or NET>&OV2>

   USER DEFLT PATH
   examples; $F2>CL>
   or NET>CL>

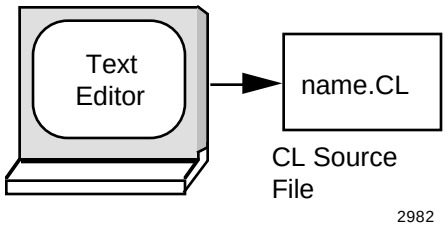
   If necessary, correct the pathnames and press ENTER. The display reappears with all pathnames in blue.



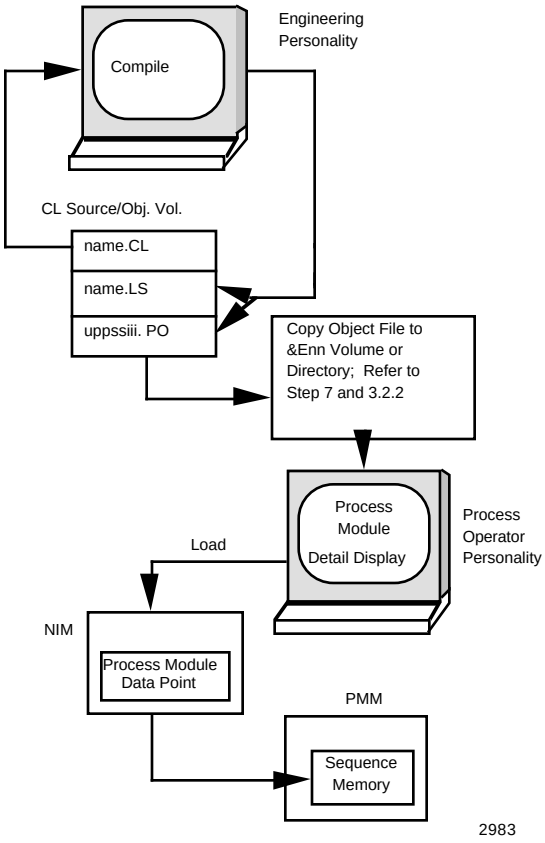
2981

> **NOTE**
>
> If you are using removable media, just before you enter CL from the COMMAND PROCESSOR display, mount &OV2 in the drive indicated by its pathname. Once the compiler overlay is read in, you no longer need the &OV2 disk/diskette.

2. Use the Text Editor to create the source file (see 5.2.1) on the CL Source/Object Volume named in step 1.



3. Compile the CL source file (see 5.3.2). If you use the -UL option, sequence names, phase names, step names, subroutine names, abnormal condition handler names, self-defining enumeration state names, and SEND string items are automatically entered in the NIM Library.



If errors are detected, an error file

(filename).LE

is created on the CL Source/Obj volume, where (filename) is the name of your source file.

If no errors are detected, these files are created on the CL Source/Obj volume:

(filename).LS

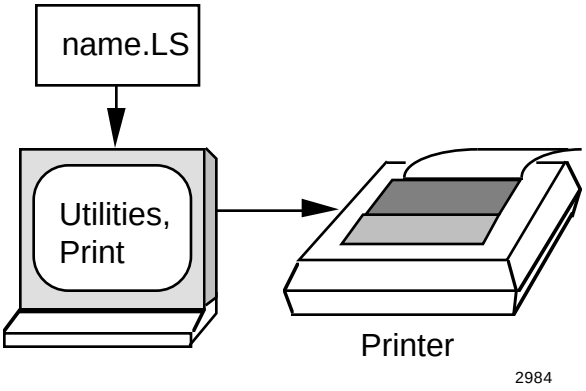
(uppssiii).PO

Where (filename) is the name of your source file and (uppssiii) is the object file name, in the following format:

u	is the UCN number of the bound data point (because only one digit is provided, UCNs 10 through 20 are represented by the alphabetic characters A through K)
pp	is the PMM number of the bound data point (1-64)
ss	is the sequence slot number (1-160, where A0 = 100, B0 = 110, . . .and G0 = 160)
iii	is the sequence name index (2-999) in the NIM library

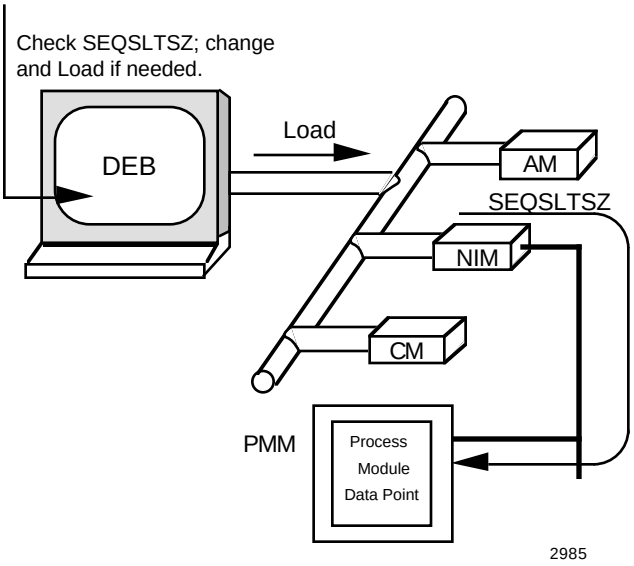
4. Use the Print command (see 5.1) to look at the listing (.LS) file.

   Find and write down the object size information from the listing. The object size is stated as blocks of 32 words.

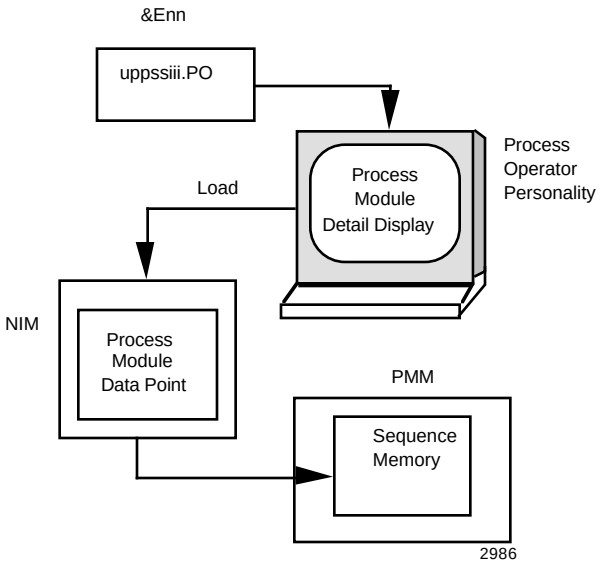


5. Use the DEB to reconstitute the Process Module point for this sequence (see the *Data Entity Builder Manual*). Make sure the value in parameter SEQSLTSZ allocates enough sequence memory space to accommodate the object size determined in step 4. It's a good idea to make it large enough to accept future compilations, as well.

6. When the PED-status line indicates no errors, no missing data, and no unENTERed data, write the Process Module point to an IDF and load it in its node.



7. Prepare to load the .PO object file according to 3.2.2 (the .PO file must be in volume &Enn), then load the program through the Process Module display in the US On Process Personality.



3.2.1 NIM Sequence Library

The NIM maintains one sequence library for all sequences on its UCN. The library consists of 3000 entries of 8-character strings. All process modules in a NIM use the same library. The library is initialized to all blanks at initial NIM startup.

For CL/PM, the strings listed in the following table must be defined in the NIM library of the PMM of the bound data point:

Table 3-1 — Strings in NIM Library

String	Available Entry Numbers
Sequence Names	2 – 999
Phase Names, Step Names, SEND string item, self-defining enumeration state names, abnormal condition handler names, subroutine names	2 – 3000

Entry 1 always contains blanks (used in displays as the step name for any phase, subroutine or handler without a step).

Names are added to the sequence library as follows:

- Sequence names are assigned in decreasing order from 999 to 2.
- All other names are assigned in decreasing order from 3000 to 2. Locations for self defined enumerations are found by searching in descending order for the first available group of contiguous locations. The individual state names then are entered to those locations in ascending order.

There are two methods of adding words to the NIM library:

- Let the CL Compiler do it automatically by selecting the –UL compile command option. If you compile without the –UL option, a .LE file is generated that lists needed words not found in the NIM library. Note that the –UL switch enters names into the library only on a clean compile. If there are compilation errors, names are not added.

CAUTION

Once a compilation that uses the NIM library has been run, it is best to use caution if changing entries in the library by the second method (using the DEB). Changing entries could cause incorrect messages and/or the display of wrong sequence, phase or step names. If you must make changes, you should repeat all compilations that use the library, to ensure that all names are correct.

- Use the DEB to key words into the NIM Library entity. You can reconstitute the library one third at a time by specifying $NuuLIBn, where uu = UCN number and n = 1, 2, or 3 (for first , second, or third 1000 entries). Once the library is reconstituted, you can key in new words or change words, and then reload; or you can build a fresh version, initializing all words to all blanks as follows:

  – select Network Interface Module at the Engineering Personality
  – select the Library Configuration box on the NIM Build Type Select menu
  – enter Network number (UCN number) and library number (1, 2, or 3 for first. second, or third 1000 entries).
  – load

Sequence Library Example:

If library contents are as follows

entry	contents
3000	hi
2999	bye
2998	sam
2997	blank
2996	blank
2995	blank
2994	blank

and the self-defining enumeration red/blue/green is added to the library, the library now appears as

entry	contents
3000	hi
2999	bye
2998	sam
2997	green
2996	blue
2995	red
2994	blank

If a sequence name has been entered in the library, another name (example, phase name) with the same name will reference the sequence library entry. That is, the string will not be re-entered in entries 1000-3000; however, if a nonsequence name has been entered in the library in an entry from 1000-3000, a sequence name by the same name must be re-entered in an entry from 2-999.

3.2.2 Prepare to Load the .PO Object File

The CL/PM object file(s) (.PO) must be available on a volume whose name is &Enn, where nn is the UCN number. There are two ways to make such a volume available:

- Use the Engineering Personality's Command Processor's Create Directory command to create the directory &Enn on the user volume that has your .PO file, on another user volume, or on a disk/diskette.
- In System Startup Task 10, configure a "CL/PM UCN" volume for each UCN needing a download volume.

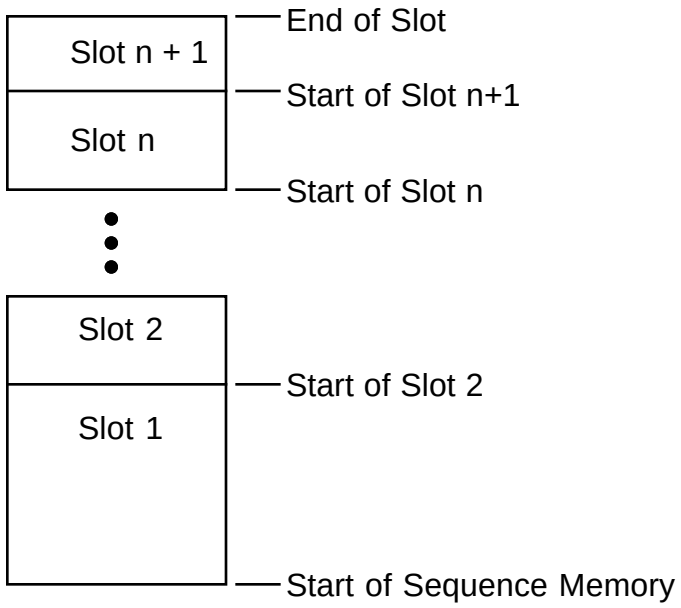


Figure 3-1 — Example of Sequence Slots in Sequence Memory 2987

Use the Engineering Personality's Command Processor's Copy or Move command to copy/move your .PO file to the &Enn volume or directory. This volume/directory could be used to store your tested (debugged) CL/PM programs, keeping them separate from untested CL/PM programs on your user volume.

Sequence programs are loaded into the slot of the sequence program's bound data point. Layout of sequence slots in memory are shown in Figure 3-1. (The number of sequence slots is user configurable in the PM Node Configuration entity.) If you reload a changed sequence whose size has changed, you may not be able to load it. Here are some download guidelines:

- The size of each sequence slot is stored in parameter SEQSLTSZ in the Process Module data point. The total of all the sequence-slot sizes in the PMM cannot exceed the available sequence memory. If a sequence program is larger than its configured slot size, it cannot be loaded.
- The location in memory where a sequence begins is determined by the sum of all of the preceding slot sizes.

- You can increase or decrease a sequence slot size on-line without affecting other already loaded (or running) sequences as long as the total memory for the box is not exceeded.

- Sequence slot size is specified as a number of 32-word blocks.

3.3 MOVING CL/PM APPLICATIONS BETWEEN SYSTEMS

CL/PM structures can be moved from one LCN-based TDC 3000^X System to another, only in CL/PM source file form. Object files and IDFs contain internal representations of names that apply to only the system they were created on; specifically,

- **Data points** in the destination system must be built using the **DEB's Exception Build** command.

- **CL/PM source files** must be recompiled on the destination system, with the associated Process Module data points and NIM Sequence Library entities (see 3.2.1) loaded in the NIM(s) in the destination system.

It is likely that before you do any of these activities you will need to edit the CL/PM source files on the destination system, to change point names and to tailor the source file for the destination system.

ENGINEER'S KEYBOARD
Section 4

Use this section when you don't know or are not sure of the effect of pressing a key on the Engineer's Keyboard.

4.1 CL/PM DATA ENTRY KEY FUNCTIONS

The Engineer's Keyboard is required for interaction with the Engineering Personality's Utilities, Text Editor, and CL/PM Programs activities. Most of the keys function as you would expect. Figure 4-1 clarifies the functions of several of the keys.

Several keys have dual functions and dual markings—symbols on the top of the key and other symbols on the front edge of the key. The secondary function, indicated on the front edge of these keys, is obtained when one of the two CTL keys is held down while the function key is pressed.

The generic-function keys, F1 through F12, are used to execute Text Editor commands, as described in Section 3 of the *Text Editor Operation* manual. These keys are not used in the Utilities and CL/PM Programs activities.

The color and behavior keys (BKGND, BLINK, INTEN, WHT, BLK, CYAN, BLUE, MAGN, RED, YEL, GRN) don't operate with the Utilities, Text Editor, and CL/PM Programs activities.

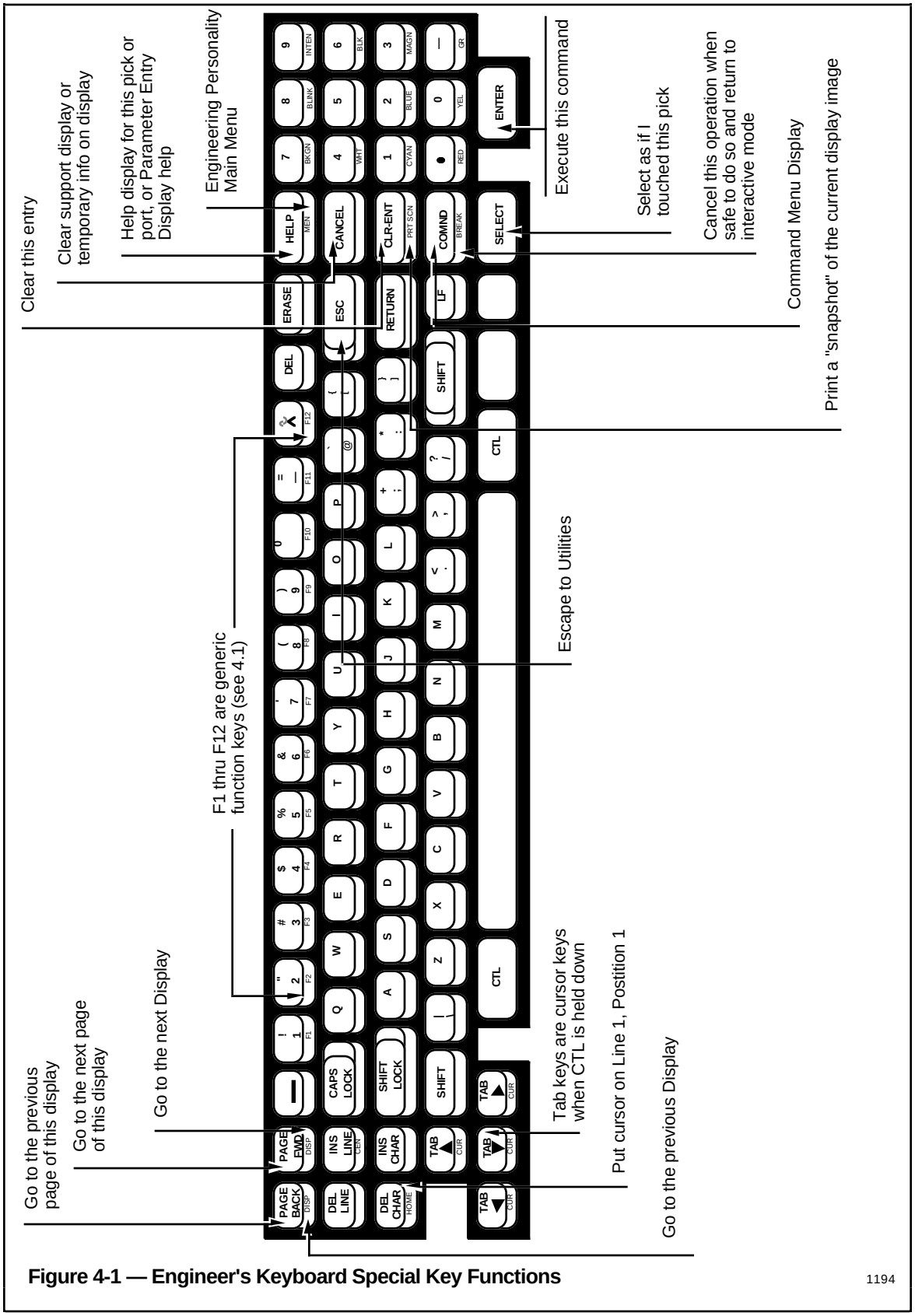


Figure 4-1 — Engineer's Keyboard Special Key Functions

COMMANDS
Section 5

This section provides

- *An overview of Text Editor and Utilities commands that are related to CL/PM data entry (see 5.1 and 5.2).*
- *Detailed descriptions of the CL/PM Compile commands (see 5.3).*

Section 3 describes how these commands are used to install and to modify CL/PM programs. Section 6 provides guidelines for recovering from errors indicated as the Compile commands are executed.

5.1 USING UTILITY COMMANDS IN CL/PM DATA ENTRY

The default-volume pathnames and utilities are used in CL/PM data entry. The Modify Default Volume Pathnames display is called up by selecting the `SUPPORT UTILITIES` pick on the Engineering Main Menu and then selecting `MODIFY VOLUME PATHS` on the Support Utility Menu. Alternately, you can select the COMMAND PROCESSOR pick on the Engineering Main Menu, and then type SP and press ENTER. The utilities are called up by selecting the `COMMAND PROCESSOR` pick on the Engineering Main Menu.

The following are the functions that you will probably use most in CL/PM data entry (command-file utilities Execute Command, Prompt Out, Pause, and End are described under 5.4):

- **Modify Default Volume Pathnames**—This function is used to set up the pathnames to the files that are created and used in CL/PM data entry. Examples of such pathnames are provided in this publication under the following heading:

  3.2—Installing a CL/PM Sequence Program

  Instructions for calling up the Modify Volume Pathnames Display are in paragraph 4.3 in the *Command Processor Operation* manual. Instructions for changing the default pathnames are in Section 6 of that publication.

- **Copy**—Used to copy a file or files from one volume to another, or on the same volume. We recommend that you copy all of your files to removable media, in case a file is lost or damaged. The Copy and Copy Volume commands are described under 5.6 and 5.7 in the *Command Processor Operation* manual.

- **Create**—Used to name a volume on a floppy diskette or cartridge disk and to initialize the diskette/disk so that it can be used on the TDC 3000^X System. This command is used to create user volumes that you use in CL/PM data entry, or on which you make backup copies of HM files. This command is described under 5.8 in the *Command Processor Operation* manual.

- **Create Directory**—Used to create a directory under an existing volume. An example is described in subsection 3.2.2 in this publication, Prepare to Load the PO Object File.

- **List Names**—Used to list the attributes of a volume and its files. You can use this command to determine if specific files are in a volume, the time and date they were created, their size, and other information. For your records, you might want to use the Data Out command to direct the list to a printer as well as to the screen. The List Names command is described under 5.13 in the *Command Processor Operation* manual.

- **Data Out**—Directs the output generated by a subsequent Print Command to a printer or file as well as to the display screen, or to the screen only. This command is executed directly from the COMMAND PROCESSOR. To direct the output to go to the screen and a printer, on the Control Language Compiler display, key in a command as in this example:

  ```
  DO $Pn
  ```

  and press ENTER. In this command, n is the printer number (which is available on the Console Status Display). To direct that the output is to go to only the screen (this is the default situation), key in DO and press ENTER.

- **Print**—Used to display or to print the content of a file. Use this command to look at your listing (.LS) and error (.LE) files. This command is executed directly from the COMMAND PROCESSOR. Key in a command as in this example:

  ```
  PR NET>CL>VLR_CALC.LE
  ```

  and press ENTER. When the whole content of the requested file has been displayed, "Print Complete" appears at the bottom of the display area. If a Data Out command has also directed the output to a printer, the content of the file is simultaneously displayed and printed.

- **Delete**—Deletes a file from a volume. If you make changes to CL/PM source files, you may want to copy the original file to a file with a slightly different name, make the changes in the new file, install those changes, test them, and when the modified structure is operating properly, delete the original file. For example, you might name your original file CALC1 and the new file CALC2. When CALC2 is compiled, installed, and successfully running, you could delete CALC1. The delete command is described under 5.10 in the Command Processor *Operation* manual.

5.2 USING TEXT EDITOR COMMANDS IN CL/PM DATA ENTRY

The following procedures describe the Text Editor functions you will probably use most frequently in CL/PM data entry. For more detailed descriptions of Text Editor functions, refer to the *Text Editor Operation* manual.

5.2.1 How to Create a CL/PM Source (.CL) File

1. You access the Text Editor by the COMMAND PROCESSOR pick on the Engineering Main Menu display.

2. Key in "ED" followed by the pathname for the file to be created. Be sure to include the .CL suffix. For example,

```
ED NET>CL>CALC1.CL
```

3. Press ENTER. A blank Text Editor display appears, ready for you to key in your source-file information.

4. Key in your source file. The form and content of source files for all CL structures are in the *Control Language/Process Manager Reference Manual*.

5. After you have keyed in and visually checked your entire source file, end the Text Editor session by pressing LF, then E, and RETURN (or CTL F1, followed by CTL F2). The source file is transferred from the Text Editor's temporary file to the user volume that you named in step 2, and your source (.CL) file is now ready to compile.

5.2.2 How to Copy a File, Edit it, and Store it in a New File

Often, you will use several similar CL/PM structures, and you can save time by copying a source file, editing the copy, and storing the result in a new file.

1. You access the Text Editor by the COMMAND PROCESSOR pick on the Engineering Main Menu display.

2. Key in ED followed by the pathname for the file to be copied. Be sure to include the .CL suffix. For example,

```
ED NET>CL>CALC1.CL
```

3. Press ENTER. The first 23 lines of the file are displayed.

4. Edit the copy of the file that is now on the Text Editor display to make the changes that are needed in the new file. When you finish, leave the cursor on the first character you want in the new file.

5. Hold CTL and press F7. Four new prompters appear (`F1=Define`, `F2=Get`, `F3=Put`, and `F4=UnProt`).

6. Hold CTL and press F1. A prompter appears that asks for the pathname for the new file.

7. Key in the pathname of the new file and press ENTER. Here is an example of the pathname:

   ```
   CALC2.CL
   ```

8. Hold CTL and press F3. A prompter appears that asks how many lines to output. If you want to store the whole file, key in 9999 (this is more lines than the largest file); otherwise, key in the number of lines you want to store in the new file.

9. Press ENTER. You now have two copies of the file, the original one that you copied and the new one that you named in step 7. In this example, they are both on the same volume. If a different volume is needed, precede the filename in step 7 with the entire name of the required path.

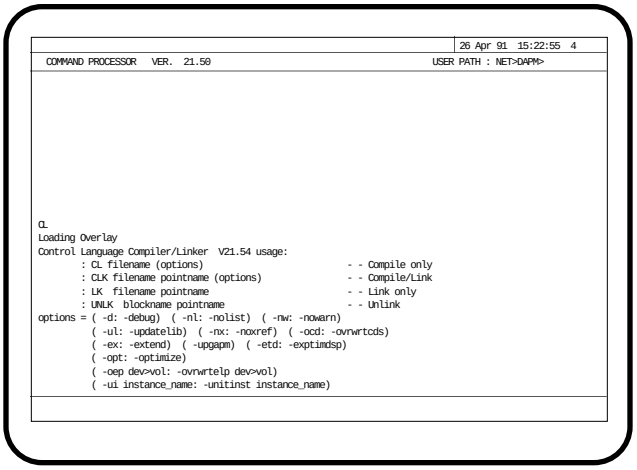
5.3 CL/PM COMPILER COMMANDS

The CL Compiler commands are available through the `COMMAND PROCESSOR` pick on the Engineering Main Menu. The Compiler reads CL/PM source (.CL) files and creates object files from them. Object files contain the CL/PM structures in the form that is executed by the destination node. The object files are loaded into the Process Manager Module in which the bound data point resides.

5.3.1 Compiler Command Descriptions

This Control Language Compiler Display appears when you type `CL` on the COMMAND PROCESSOR display and press ENTER or execute any CL Command from the COMMAND PROCESSOR.

When the display first appears, the cursor is in the command-entry/prompter region, as shown here. The area between the command-entry/prompter region and the title line displays the results of command execution.



2989

5.3.1.1 Commands

For CL/PM, only the Compile command can be executed from this display. Execution is requested by keying in the command line with any options and pressing ENTER.

- **Compile**—Compiles CL/PM source files, creating object and listing files, or if errors are detected, an error file. Form:

```
CL (filename) -(options)
```

where: (filename) is the name of the CL/PM source file.

The source file must have a .CL suffix, but use of the suffix in the compile command is optional.

> **NOTE**
>
> The remaining three commands shown on the screen (Link, Unlink, and Compile and Link) apply only to CL/AM programs. CL/PM programs are not linked by the Link command. They are loaded from the Process Module Detail display in the Universal Station's Operator personality. See the *Process Operations Manual* and section 3.2.2 of this document.

5.3.1.2 Compile Command Options

The following options are available for the Compile command. The options are requested by keying in "–" followed by the option name for each option. More than one option can be included, and each is separated by a space. For example, a command line that requests several CL/PM options would have this form:

```
CL CALC2 -D -UL -NW -NL -NX –OEP dev>vol –HE
```

Following is the full set of CL compile options (note that some do not apply to CL/PM):

- **Debug**—Compile %DEBUG lines in the source file. See 3.3.3 in the *Control Language/Process Manager Reference Manual.* Form:

```
-D  (or -DEBUG)
```

- **Update Library**—Update the NIM library (see 3.2.1). Form:

```
-UL  (or -UPDATELIB)
```

- **No Warnings**—Do not list notes and warnings in .LE or .LS files. Form:

```
-NW  (or -NOWARN)
```

- **No Listing**—Do not generate a full listing file (.LS); list only error lines. Form:

  `-NL` (or `-NOLIST`)

  The -NL option automatically causes the -NX option to be selected; therefore, the CL/APM cross reference is omitted from the listing and both -NL and -NX are displayed as selected options at the bottom of the listing.

- **No Cross Reference**—Omit printing the CL cross reference (symbol table) at the end of the listing. Form:

  `-NX` (or `-NOXREF`)

- **Override Equipment List Name**—Overrides the device name and volume name specified in an %INCLUDE_EQUIPMENT_LIST statement and/or the User Default Path. Form:

  `-OEP device>vol` (or `-OVRWRTELP device>vol`)

  Example: `-OEP NET>ELB`

- **Unit Instance Only**—Only the one specified unit instance of an included Equipment List will be compiled. Form:

  `-UI unit_instance_name` (or `-UNITINST unit_instance_name`)

  If no errors are detected during compilation of the unit instance selected,
  - Listing (.LS) and object (.NO) files are generated only for that unit instance. Existing .LS and .NO files for that unit instance are overwritten.
  - No error listing (.LE) file is generated for that unit instance and any existing .LE file for that unit instance is deleted.

  If any error is detected during compilation of the unit instance selected,
  - An error listing (.LE) file is generated that includes information only for that unit instance. If a .LE file for that unit instance already exists, it is overwritten.
  - No listing (.LS) or object (.NO) file is generated, thus any existing .LS and .NO files for that unit instance are not replaced.

- **Optimize Object Generation**—For any unit instance in an included Equipment List for which compatibility is maintained, no new object file is created. Compatibility is maintained when there has been no change to alias names or their data types for a unit instance since the previous compilation (established by a comparison of the time stamps in the source and existing object files). If the -UI option also is used, only the selected unit instance can be affected. Thus, if compatibility of that unit instance is maintained, an object is not created. Form:

  `-OPT (or -OPTIMIZE)`

  If no errors are detected during compilation of a unit instance,
  - Listing (.LS) file is generated. Existing .LS file is overwritten.
  - No error listing (.LE) file is generated and any existing .LE file is deleted.

  If any error is detected during compilation of a unit instance,
  - An error listing (.LE) file is generated that includes information for that unit instance. If a .LE file already exists, it is overwritten.
  - No listing (.LS) or object (.NO) file is generated, thus any existing .LS and .NO files for that unit instance are not replaced.

- **Heap Expansion**—Extends the amount of heap memory available for CL compilations if certain requirements are met. Form:

  `-HE (or -HEAPEXP)`

CAUTION

The heap expansion option requires software release R410 or later, Universal Personality, and a Universal Station with a minimum of 5 megawords of memory. (Without the heap expansion option, the minimum memory size for a Universal Station with Universal Personality and R410 software is 4 megawords.)

If the requirements in the caution box above are met, you can use the heap expansion switch (option) to provide more heap memory for CL compilations. This option can be especially useful in compiling large CL programs that use the Equipment List function. CL programs that use Equipment List require more heap memory than programs that do not use Equipment List, and this reduces the number of blocks of object code that the CL Compiler can generate. The following table shows the number of blocks of object code that the compiler can generate with and without the heap expansion option for programs that use Equipment List and programs that do not use Equipment List:

Controller	Number of Blocks of Object Code			
	With Heap Expansion Option (UP with 5 Mw Memory)		Without Heap Expansion Option (EP or UP)	
	With Equipment List	Without Equipment List	With Equipment List	Without Equipment List
APM	500 (16,000 words)	500 (16,000 words)	276 (8,832 words)	392 (12,544 words)
PM	336 (10,752 words)	336 (10,752 words)	223 (7,136 words)	256 (8,192 words)
MC/AMC	512 (8,192 words)	512 (8,192 words)	356 (5,696 words)	512 (8,192 words)

The maximum amount of heap memory available in the Engineering and Universal Personalities without expanded memory is 320,000 words. The maximum amount of heap memory available in the Universal Personality (with 5 Mw memory) is 700,000 words if the heap expansion option is used.

NOTE

To terminate a compilation in progress, press the combination of the "CTL" and "BREAK" keys on the engineering keyboard. The CL compiler recognizes the Break key during any of its three passes or during the listing generation.

5.3.1.3 Compilation Results

After a successful compilation, the status of the object file is reported both on the screen and in the listing (.LS) file. No files are modified on an unsuccessful compilation.

The following are examples of the messages that report compilation status:

For a compilation that results in a new .PO object file;

```
File A0302997.PO created
```

For a compilation that overwrites an existing .PO object file;

```
File A0302998.PO overwritten
```

5.3.2 How to Invoke Compiler Commands

5.3.2.1 Compilation Precaution

Simultaneous CL/PM Compilations—An incorrect compilation may occur if two CL/PM compilations occur simultaneously (on different Engineering Personality USs), using the same NIM library. Only one of the compilations will successfully modify the library; the other one may contain erroneous names. To avoid this, verify that one such compilation is complete, before another is attempted.

5.3.2.2 Volume Handling and Pathnames

The COMMAND PROCESSOR display doesn't provide prompters to request that required volumes be mounted in a disk/diskette drive for a CL compilation. If you don't make a needed volume available, an error message appears, indicating that the required volume or file could not be found. After you mount the required volume, retype your command line and press ENTER.

The pathnames used in executing Compiler commands are the default pathnames on the Utilities' Modify Default Volume Pathnames display.

The descriptions of typical CL/PM data-entry sessions in Section 3 provide examples of volume pathnames, and suggestions for handling disk/diskette. Table 2-1 lists all volumes involved in CL/PM data entry.

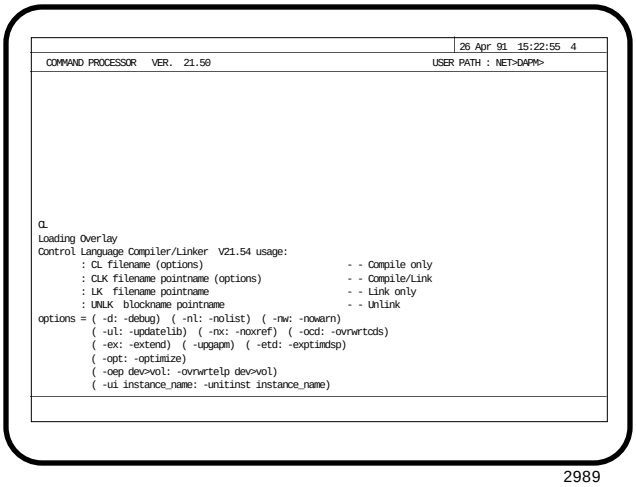
5.3.2.3 Invoking a Compile Command

1. If you are using some Engineering Personality activity other than the COMMAND PROCESSOR, hold CTL and press MENU to return to the Engineering Main Menu.

2. To invoke CL:

   Select the COMMAND PROCESSOR, type `CL` and press ENTER (Engineering Personality Overlay volume &OV2 must be available on a disk/diskette or an HM). The Control Language Compiler display appears.

3. The summary of CL commands and options appear as shown here.

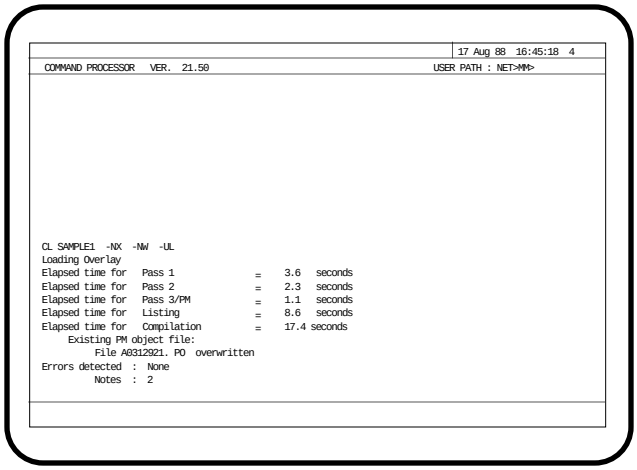


2989

4. Check that your CL/PM source file is available on the CL Source/Obj volume.

   Key in a Compile command line, and any options needed (see 5.3.1.2). Here we keyed in "`cl sample1`." "Sample1" is the name of our source file.

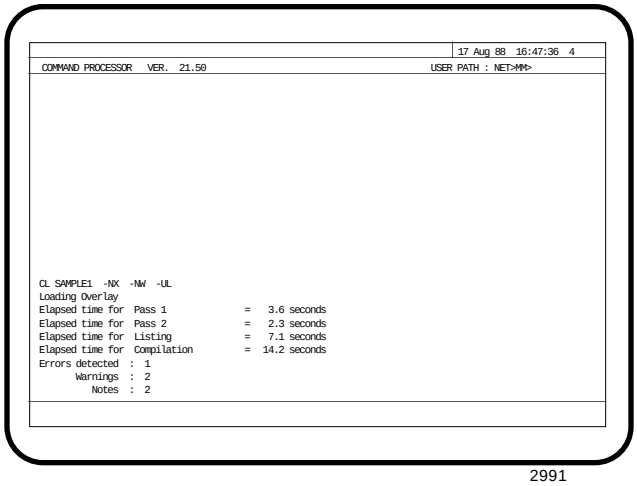
5. Press ENTER. The Compiler makes several passes through the source file and each of these is noted on the display.
   As each pass is processed, the line number of the current line being compiled is displayed on the Command Line to show progress of the compilation. Comment lines in the CL program are skipped and their line numbers are not displayed. During compiler processing of internal executive routines, a zero (0) line number is displayed.
   Here, the compiler made three passes and generated the listing in 17.4 seconds, and detected no errors. The listing file (.LS) and the object file (.PO) are now available on the CL Source/Obj volume.



2990

   If errors are detected, they are indicated on a line like the one at the bottom of this display, the listing and object files are not created, and instead, an error file (.LE) is created on the CL Source/Obj volume.

   Section 6 provides guidelines for recovering from errors.



2991

5.4 HOW TO USE COMMAND FILES

CL and Command Processor commands can be included in command files to cause them to be executed without human interaction. The command processor commands that facilitate the execution of command files are available through the COMMAND PROCESSOR.

`DO (pathname)`	Data Out
`P (pathname)`	Print
`EC (pathname)`	Execute Command File
`PO (pathname)`	Prompt Out
`Pause`	Pause (stop execution) until the operator presses ENTER.
`End`	End Command File Execution and this Utilities session.

where: (pathname) is a complete file pathname for either disk/diskette or HM.

See Section 5.1 for a description of the Data Out and Print commands.

The EC command directs the Universal Station to take commands from the specified file, rather than from the keyboard. The commands are displayed on the screen as they are read from the file. The responses to the commands also go to the screen, just as if they were being executed through interaction with an operator. Command execution continues until an End command is encountered or the end of the file is reached, and you are returned to the Command Processor.

You can abort EC file execution by holding CTL and pressing COMND. When the command being executed is complete, execution stops, and you are returned to the Command Processor.

"&P" can be included in a command file to indicate comments that are not considered as commands. "&P" and the text that follows it appear on the screen as they are encountered, and if so specified by a PO command, they are also printed or appended to a file.

The PO command directs the US to copy the commands and responses as they appear on the screen, to the specified printer, or to append them to the specified file. This continues until an End command or the end of the file terminates execution, or until a PO command with no pathname is executed.

The PAUSE command causes command execution to cease. Command execution continues when the ENTER button is pressed. The primary purpose of the PAUSE command is to allow you to mount a disk/diskette during the execution of an Execute file.

We recommend that you use a .EC suffix on your command files to identify their purpose. The EC and PROMPTOUT commands don't require this specific suffix, but you will find a .EC suffix a help in remembering the purpose of the file.

As an example, you could use the Text Editor to create a file in a volume named "EC" with the file name of "ECFILE.EC," like this:

```
PO NET>CL>SESSION.EC
&P This command file compiles two CL blocks
CL PUMP1

CL PUMP2
END
```

Then on the Control Language Compiler display, key in

```
EC NET>EC>ECFILE.EC
```

and press ENTER. The PO command directs the commands, the &P line, and all responses to file SESSION.EC. The two CL commands are then executed, and the End command terminates the session and returns you to the Command Processor.

If you wanted to review the commands and responses that were stored in SESSION.EC, you could key in

```
PR NET>CL>SESSION.EC
```

Then press ENTER. The content of the file appears on the display.

ERROR INDICATIONS AND RECOVERY
Section 6

Use parts 6.1 and 6.2 of this section to recover from errors detected by the Compiler commands. Part 6.3 provides information that will help you to recover from errors that are detected as a CL/PM program is running in a Process Manager Module.

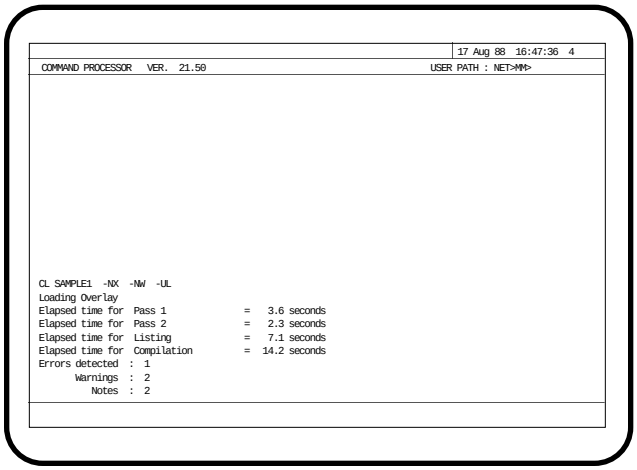
6.1 HOW ERRORS ARE INDICATED

CL/PM data entry errors are indicated two ways:

- On the COMMAND PROCESSOR Display.
- In error listing (.LE) files that the Compiler generates when it detects errors.

As each CL/PM command is executed, messages appear on the Control Language Compiler display that indicate progress as the command is executed and that provide information that may or may not indicate errors.

The content of these messages must be considered, to determine if corrective action is needed or not. For example, this display indicates that a Compile command was executed; however, the last message on this display indicates that one error was detected during the compilation. That error can be found in the error listing (.LE) file.



2991

6.1.1 CL Compiler Error Messages

An alphabetical list of all of the error messages generated by the compile command is provided below. The information in the Source/Severity column has the following meaning:

- **Source**

  CX—Communication and data access messages

  P1, P2, P3PM—Compiler messages generated on pass 1, pass 2, or pass 3. These messages appear in the error listing (.LE) file.

The major activity in each of the Compiler passes is as follows:

Pass 1—CL syntax checking. Note: The compiler lists only the first syntax error it finds in each source code statement.

Pass 2—Parameter type checking

P3PM—Object code generated for a CL/PM compilation

- **Severity** (listed from most severe to least severe);

  Fatal—An error that prevents further command execution or that prevents the CL Compiler from executing beyond the point where the error occurred

  Severe—A severe, but not fatal error

  Error—An error that did not prevent command execution

  Note—The Compiler has done something that may be surprising or noteworthy

  Warning—The user has done something noteworthy.

Messages	Source/Severity
%INCLUDE_EQUIPMENT_LIST directive is not supported for CL/AM	P2-Fatal
%INCLUDE_EQUIPMENT_LIST directive only allowed as the first line	P1-Error
%INCLUDE_EQUIPMENT_LIST is not declared as the first line	P1-Error
&ASY volume not found	CX-Fatal
(filename) could not be created:<reason>	CX-Fatal
(filename) could not be deleted <reason>	CX-Fatal
(filename) could not be read: <reason>	CX-Fatal
(filename) Handler actually defined as (filename)	P2-Error
(filename) Not Found	P2-Error
(tagname.param) compile type does not match actual type	CX-Fatal
(tagname.param) does not exist	LK-Fatal
(tagname.param) is an array of point IDs that do not all refer to the same LCN	LK-Fatal
<name> cannot be redeclared	P2-Error
<name> expected	P1-Error
<name> is not a legal %RELAX option	P2-Error
<name> not found	P2-Error
<name> was not found in the system and -UL was not selected	CX-Error
A local variable is required here	P2-Error
A Loop may have only one REPEAT statement	P2-Error
A number cannot be divided by a time	P2-Error

(Continued)

Messages (Continued)	Source/Severity
A remote variable is required here	P2-Error
ALARM time must be an integer constant (0-9999) for MC	P3MC/PM/APM-Error
Aliasing of Bound Data Point	P2-Error
Alias Name does not exist in Equipment List Object File	P1-Error
Allowed only before first Block in a Package or in Block data declarations	P2-Error
All enabled abnormal handlers are disabled	P2-Note
Argument multiply defined	P2-Error
Argument must be a Data Point Param of type Data Point	P2-Error
Argument must be a Data Point Param or a LOCAL	P2-Error
Argument must be an unsubscripted data point parameter	P2-Error
Argument must be of type ENUMERATION	P2-Error
Argument must be of type String Literal	P3APM-Error
Array expected	P2-Error
Array index must be of type number	P3MC/PM/APM-Error
Array too large	P3-Fatal
Arrays cannot be compared	P2-Error
Attempt to fetch element of off node array of strings: <pointname>.<parameter>	LK-Error
Attempt to redefine (name) as a parameter name	P2-Error
Attempt to redefine parameter (name) with different type	P2-Error
Attempt to store to (tagname.param) whose access lock is View Only	P2-Error
Attempted Aliasing	P2-Error
Attempted indirection	P2-Error
Attribute previously defined	P2-Error
Background Block Active on Bound Data Point—retry Link or Unlink	LK-Fatal
BACKGROUND is allowed only in Platform Subroutines and/or Functions	P2-Error
Block already exists at another Insertion Point	LK-Fatal
Block exceeds maximum number of prefetches allowed per block	LK-Fatal
Block insertion point must be BACKGRND when calling background routines	P3-Error
Block not on the Bound Data Point	LK-Fatal
Both KEEPENB and abnormal handlers not allowed in a Phase heading	P2-Error
Bound Data Point at maximum CLs	LK-Fatal
Bound Data Point has already been processed in a previous compilation	P2-Error
Bound Data Point is active	LK-Fatal
Bound Data Point not configured for CLs	LK-Fatal
C-Link data inconsistent in READ/WRITE: may indicate that no C-Link exists	P3MC-Error
Cannot create .GD file, FMS error: <reason>	CX-Fatal
Cannot declare a data type with REFERENCE_N	P2-Error
Cannot fetch values from an OUT argument	P2-Error
Cannot access this parameter from this sequence program	P3PM/APM-Error
Cannot generate code for this point reference	P3PM/APM-Error
Cannot get default pathname	CX-Error

(Continued)

Messages (Continued)	Source/Severity
Cannot have LOCAL data definitions with the data type of Entity ID in a CL Block	P2-Error
Cannot INITIATE non Process Module Point	P3MC/PM/APM-Error
Cannot INITIATE own normal sequence	P3MC/PM/APM-Error
Cannot Link to a Library subroutine	LK-Fatal
Cannot loop in Phase Heading	P3MC/PM/APM-Error
Cannot loop in Step Heading	P3MC/PM/APM-Error
Cannot modify a local constant or enumeration literal	P2-Error
Cannot modify a Point ID—use Data Entity Builder	P2-Error
Cannot modify an expression	P2-Error
Cannot modify an IN argument	P2-Error
Cannot modify function	P2-Error
Cannot modify read-only parameter	P3MC/PM/APM-Error
Cannot multiply time by time	P2-Error
Cannot pass a member of an array of data Points	P2-Error
Cannot Read Pointer from Global memory	CX-Fatal
Cannot Read/Write subroutine argument	P3MC/PM/APM-Error
Cannot Read/Write this parameter	P3MC/PM/APM-Error
Cannot Reference @1.@2	LK-Fatal
Cannot reference data points with names greater than 8 characters across LCNs	P2-Error
Cannot reference Off-LCN $AMIF Point	P2-Error
Cannot reference off-LCN points in a sequence program	P2-Error
Cannot Restart in emergency handler	P3MC/PM/APM-Error
Cannot SEND Enumeration types with blank state names	P3MC/PM/APM-Error
Cannot SEND Enumerations	P2-Error
Cannot SEND mode type value	P3MC-Error
Cannot send Arrays	P2-Error
Cannot send Conditions	P2-Error
Cannot store to an IOL parameter in a READ statement	P3PM-Error
Cannot use a parameter which requires a Prefetch in a CALL statement	P3PM-Error
Cannot use an entire array	P3MC/PM/APM-Error
Cannot use WHEN ERROR clause when INITIATE own handler	P3MC/PM/APM-Error
Checksum Error on Linker/Loader communication	LK-Fatal
Code size exceeds maximum of 12288 words	P3APM-Error
Compiler memory overflow! Program too large or statement too complex	CX-Fatal
Composite Data Point must reside in a MC	P3MC/PM/APM-Error
Constant is too big	P3MC/PM/APM-Error
Corrupted Data Base	LK-Fatal
Code Size exceeds maximum of 8192 words	P3PM-Fatal
Could not finish reading file (filename) <reason>	CX-Fatal
Could not get file attribute (filename) <reason>	CX-Fatal
Could not get memory in AM	LK-Fatal
Could not go to End of File (filename) <reason>	CX-Fatal
Could not open %Include_Set file @1: @2	CX-Error
Could not open file (filename) <reason>	CX-Fatal
Could not write to file (filename):<reason>	CX-Fatal
Custom Set @1 is not loaded in the AM	LK-Error

(Continued)

Messages (Continued)	Source/Severity
DA error adding name to global directory, Build status=(value)	CX-Fatal
DA error converting entity name, minor_stat: (value)	CX-Error
DA error getting enumeration definition	CX-Fatal
DA error getting Point Parameters, se_stat: (value)	CX-Error
DA ext/int conv error, <reason>	CX-Fatal
DA int/ext conversion error, codes=(values)	CX-Fatal
Data Access Disk I/O Error:<reason>	CX-Fatal
Data Access error: Enumeration Set not found	CX-Error
Data Point @1 does not exist. Rebuild Equipment List.	P2-Fatal
Data Point @1 does not exist	LK-Fatal and Error
Data Point @1 not found—all nodes not operational	LK-Fatal and Error
Data Point Identifier expected	P2-Error
Data Point is too large	LK-Fatal
Data Point must be a MC point	P3MC-Error
Data Point must be in different MC from Bound Data Point	P3MC-Error
Data Point must be in the same UCN node as the bound data point	P3PM/APM-Error
Data Point must be in the same MC as Bound Data Point	P3MC-Error
Data Point must be on same Hiway as Bound Data Point	P3MC-Error
Data type DATA_POINT_ID must be followed by REFERENCE	P2-Error
Data Type name may not appear here	P2-Error
Data Type was not found in system	P2-Error
Declared size exceeds actual array bounds	P2-Error
Declared with invalid Data Type	P2-Error
Default value exceeds Port Size	P3CDS-Warning
Defined but not referenced	P2-Warning
Delaying for busy Loader	LK-Note
Destination must be a Data Point with a Mailbox	P2-Error
Different version of object exists in Unit <number> Usecount = <number>	LK-Fatal
Disk/Memory revision mismatch on Global Name File	CX-Fatal
Does not match Heading Name	P2-Error
Duplicate Declaration	P2-Error
ELSE cannot begin a Statement	P2-Error
Empty Custom Data Segment not permitted	P2-Error
Empty Object File	LK-Fatal
Empty SEND statements are not permitted in sequence programs	P3MC/PM/APM-Error
End of File expected	P1-Error
End of Line expected	P1-Severe
Entity Name could not be converted	CX-Error
Enumeration <name> is not valid on the UCN	CX-Error
Enumeration (name) not found—all nodes not operational	LK-Fatal
Enumeration literal expected	P2-Error
Enumeration literal may not appear here	P2-Error
Enumeration literal not implemented	P3MC-Error
Enumeration literal not in this Type	P2-Error
Enumeration name conflicts with Parameter List Name in Custom Library	CX-Error
Enumeration Set does not match actual enumeration set	CX-Error
Enumeration set <name> was not found in custom name library	P3CDS-Error
Enumeration state <name> not found - all nodes not operational	LK-Fatal

(Continued)

Messages (Continued)	Source/Severity
Enumeration state not valid for this sequence program	P3PM/APM-Error
Enumeration type of this object is invalid	P3MC-Error
Enumeration types with blank state names are not supported	CX-Error
Equipment List Object version mismatch - Translate or rebuild Equipment List	CX-Error
Error attempting to get parameter information; S:@1 M:@2 PA:@3	CX-Fatal
Error during attempt to get parameter information; PA Status:<status>	CX-Fatal
Error during attempt to get SDE state names; Minor Status:<reason>	CX-Fatal
EU string truncated	P3CDS-Warning
Exceeded maximum of 3000 characters allowed for Value statements	P3CDS-Fatal
Expression is invalid in READ or WRITE statement	P3MC/PM/APM-Error
Expression is not constant or cannot be evaluated	P2-Error
Expression too complicated for evaluation	P3-Fatal
Fetching UCN Box Point parameters NN, FL, or TIME is not allowed in foreground CL	P3/LK-Error
File Manager Request Block—Allocation Error	CX-Fatal
File suffix is not Equipment List Object Suffix (QO)	P1-Error
Foreground fetch of Array Point <tagname>.TIME which has > 160 elements	LK-Error
Function Result must be scalar or string	P2-Error
Global Name File full	CX-Error
Handler already enabled for this condition	P2-Error
Handler contains multiple RESTARTS	P2-Error
Handler expected	P2-Error
Hardware location is required	P3MC/PM/APM-Error
Hardware location must be NN or FL	P3MC/PM-Error
Hardware location must be NN, FL, TIME, STR8, STR16, STR32, or STR64	P3APM-Error
Heading expected here	P1-Error
Heap memory requirements exceed available heap memory	CX-Fatal
I/O card type of bound MC is inconsistent	P3MC/PM/APM-Error
Identifier cannot be resolved in this context	P2-Error
Identifier expected here	P1-Error
Identifier length error	CX-Error
Identifier must be PM, APM, or MC—defaulted to PM	P2-Error
Identifier must be within 8 characters	P3MC-Error
Illegal character	P1-Error
Illegal compiler directive	P1-Severe
Illegal external Name	CX-Error
Illegal use of "%"	P1-Error
Incorrect format string	P3APM-Error
Index Type mismatch	P2-Error
INITIATE Point.Param not allowed in this sequence program	P2-Error
INITIATE SEQ not allowed in this sequence program	P2-Error
Insertion Point invalid for build type	LK-Error
Internal communication error (IDB) - Contact Honeywell TAC	CX-Fatal

(Continued)

Messages (Continued)	Source/Severity
Internal error; Notify your Honeywell Representative:	
Converting LCN Name	LK-Fatal
Invalid access key	CX-Error
Invalid Array Base type	P2-Error
Invalid CALL argument	P3MC/PM/APM-Error
Invalid constant data type	P3MC/PM/APM-Error
Invalid Declaration	P3MC/PM/APM-Fatal
Invalid declaration for a Sequence Program	P2-Error
Invalid File Name	CX-Fatal
Invalid File Pathname or Pathname not specified	P1/CX-Error
Invalid in a Sequence	P2-Error
Invalid in Blocks	P2-Error
Invalid in this context	P2-Severe
Invalid %Include_Set file name, must be 6 characters or less	P1-Error
Invalid Insertion Point (name)	LK-Error
Invalid insertion point for configured algorithm	CX-Error
Invalid mode enumeration for this point	P3MC/PM/APM-Error
Invalid Module Number, Entity does not exist	CX-Error
Invalid parameter on this point	P3MC/PM/APM-Error
Invalid point for this sequence	P2-Error
Invalid point type for this compilation	LK-Error
Invalid point type for this compilation	P2-Error
Invalid SEND destination	P3MC/PM/APM-Error
Invalid SEND item	P3MC/PM/APM-Error
Invalid SEND WAIT option for this Destination	P3MC/PM/APM-Error
Invalid Slot Number, Entity does not exist	CX-Error
Invalid State Name	P2-Error
Invalid subscript for this parameter	P3MC/PM/APM-Error
Invalid time expression	P3MC/PM/APM-Error
Invalid to indirect through an off-node point	LK-Fatal
Keylock error on attempt to store to (tagname.parameter)	LK-Error
Labels must be unique within a Program Block	P2-Error
Library full of Names/Strings, id not added	P3PM-Error
Library full of Sequence/Phase/Step id, id not added	P3MC/PM/APM-Error
Library full of Strings, String not added	P3MC/PM/APM-Error
Library Subroutine not permitted in a Package	P2-Error
Library Subroutine Object corrupted	LK-Fatal
Linking/Unlinking to points on other LCNs is not supported	LK-Fatal
Loader communication error	LK-Fatal
Loader still busy after one retry	LK-Fatal
Loader time out error	LK-Fatal
Loader/Data Access communication problem	LK-Fatal
Loader/Kernel communications problem	LK-Fatal
Local Subroutine cannot have a Bound Data Point	P2-Error
Local time not synchronized with network	CX-Note
Local variables may not be data point identifiers	P2-Error
Logical condition expected	P2-Error
Logical conditions cannot be compared	P2-Error
Logical or Enumeration type expected here	P2-Error
Loop Index not declared	P2-Error

(Continued)

Messages (Continued)	Source/Severity
Loop was never repeated	P2-Warning
Lower bound must be less than upper bound	P2-Error
Malformed Identifier	P1-Error
Malformed number	P1-Error
Malformed Point Identifier	CX-Fatal
MAN, AUTO, CASC, and COM may be stored but not compared	P3MC-Error
Missing Abnormal Handler ID	P1-Error
Missing Closing Quote	P1-Error
Missing END	P1-Error
More than 12 IOL parameter fetches in one step	P3PM/APM-Error
MSF Error - Code:@1	P1/P2-Fatal
MSF FM Error - @1:@2	P1/P2-Fatal
MSF.LO is not installed in this node	P1-Fatal
Multiple locals assigned to this hardware location	P2-Error
Multiple stores of strings are not permitted	P2-Error
Must be a BDP Parameter	P2-Error
Must be a Parameter List	P2-Error
Must be a positive integer	P2-Error
Must be an integer	P2-Error
Must be APM resident point	CX-Error
Must be in the RESTART section of an ABNORMAL Handler	P2-Error
Must be MC resident point	CX-Error
Must be PM resident point	CX-Error
Must contain executable statements	P2-Error
Must link to point (name)	LK-Fatal
Name/String not found in Library and -UL was not selected	P3PM-Error
Name too long	P3-Error
Nested CALL invalid	P3MC/PM/APM-Error
Nested PACKAGE heading ignored	P2-Severe
Network in the EL Object does not match with compilation target	P2-Fatal
No available entries in Library for Sequence names, Sequence name not added	P3PM/APM-Error
No corresponding item for this Variable	P2-Error
No response from the remote AM node	LK-Fatal
Not a parameter	P2-Error
Not a Process Module point	CX-Error
Not a Subroutine	P2-Error
Not Enough Contiguous Memory Error	LK-Fatal
Not enough memory	CX-Fatal
Not supported	All-Error
Not supported on target machine	P3-Fatal
Not valid for AM	P2-Error
Null point ID only permitted in point ID arrays	LK-Fatal
Number of values differs from number of array elements	P2-Error
Numeric literal expected here	P1-Error
Object code memory requirement exceeds maximum allowable object size	LK-Fatal
Object corrupted in transmission from US to AM	LK-Fatal
Object File Access Error	LK-Fatal
Object File Corrupted	LK-Fatal

(Continued)

Messages (Continued)	Source/Severity
Object file FMRB allocation error	LK-Fatal
Obsolete syntax - this name is ignored	P2-Error
Old File (filename) cannot be safely renamed (filename)	CX-Fatal
Only one %INCLUDE_EQUIPMENT_LIST directive is allowed	CX-Error
Only one Custom Data definition is permitted	P2-Error
Only valid in a Custom Data Segment	P2-Error
Parameter (name) not found—all nodes not operational	LK-Fatal
Parameter (name) not on this Data Point	P2-Error
Parameter conflict	P2-Error
Parameter Declaration expected	P2-Severe
Parameter Description truncated	P3CDS-Warning
Parameter List <name> was not found in custom name library	P3CDS-Error
Parameter List File (filename) is empty	P2-Error
Parameter List name conflicts with Enumeration Set Name in Custom Library	CX-Error
Parameter name conflicts with Segment name in Custom Library	CX-Error
Parameter name not allowed in this context	P2-Error
Parameter references used in a subscript must be On-Node	P3APM-Error
Parameters data type not supported	P2-Error
Passing enumeration literals as arguments in a CALL is not allowed	P3MC-Error
Phase Heading contains multiple ALARM clauses	P2-Error
Point ID change only valid after unlink and link	P3-Warning
Point Identifiers cannot be compared	P2-Error
Point is not NIM resident	CX-Error
Point must be declared in an EXTERNAL statement prior to the LOCAL declaration	P2_Error
Point must be on the same UCN as the Bound Data Point	CX-Error
Point must be resident on the UCN	CX-Error
PPS parameter not logical	P2-Fatal
Program is too large	P3MC/PM/APM-Error
PRTFL parameter of composite point contains only the status of the input slot	P3MC-Warning
Quoted string is too long. Maximum length allowed is <value>	P1-Error
Reference List entry created for each element of this array	P3-Note
References an off-LCN point; must link to IPP or Backgrnd Insertion Point	LK-Fatal
Remote variable in READ/WRITE must be in different box from Bound Data Point	P3MC/PM/APM-Error
Remote variable in READ/WRITE/INITIATE must be on same C-LINK as Bound Data Pt	P3MC-Error
RESUME must have PHASE destination	P3MC/PM/APM-Error
Routine must begin with step	P2-Error
Routine Out of Order	P2-Error
Same object exists with different access key in Unit <name>. Use count = <value>	LK-Fatal
Segment name conflicts with standard parameter or param name in Custom Library	CX-Error
Segment size exceeds maximum by <number> bytes	P3CDS-Error
Sequence must begin with a PHASE	P2-Error
Sequence/Phase/Step ID must be defined in Library	P3MC/PM/APM-Error

(Continued)

Messages (Continued)	Source/Severity
Source File contains no code	P1-Fatal
Source File is empty	CX-Fatal
Statements not valid in Custom Data Segments	P2-Error
String exceeded maximum allowable length and was shortened to <value> characters	P3CDS-Warning
String is valid only in SEND statement	P3MC/PM-Error
String literal expected here	P1-Error
String literal type can only be general (G), integer (I), or real (R) format	P3APM-Error
String must be defined in Library	P3MC/PM/APM-Error
String must be within 8 characters	P3MC/PM/APM-Error
Subroutine/Function name must be less than or equal to 80 characters	CX-Error
Subscript must be an integer constant	P3MC/PM/APM-Error
Subscript outside array bounds	P3-Error
Subscript used too deep in an indirect reference	P3-Error
Syntax Error	P1-Severe
System Name conflict	CX-Fatal
Tagname access references only the upper subslot of this dual digital parameter	P3MC-Warning
Target does not define a loop	P2-Error
Temp File (filename) could not be renamed:(filename)	CX-Fatal
The AM must be configured with at least one BACKGROUND task	LK-Error
The argument being passed in is not a Data Point Parameter	P2-Error
The argument being passed in is not an Enumeration type	P2-Error
The Data Point Name @1 is invalid	LK-Fatal
The loop counter must be a LOCAL variable	P2-Error
The referenced parameter list (filename) was not found	P2-Error
The target of a CALL statement must be a subroutine	P2-Error
There are too many parameter references in this block	LK-Fatal
This array limited to one dimension	P2-Error
This data type may only be compared equal/unequal	P2-Error
This Include Set file has an error on line @1, column @2	CX-Error
This is not an array	P2-Error
This LOCAL variable is mapped to a Serial Interface IOP parameter	P3APM-Warning
This parameter name cannot be used in a Custom Data Segment	P3CDS-Error
Time expression may not be followed by a WHEN clause	P2-Error
Time may be wrong	CX-Note
Too few arguments	P2-Error
Too many arguments	P2-Severe
Too many Array elements	P3CDS-Fatal
Too many array elements individually initialized with the VALUE statement	P3CDS-Error
Too many Build Visible Parameters	P3CDS-Fatal
Too many constants in program	P3MC/PM/APM-Error
Too many enumerations	P2-Error
Too many Links to Unit	LK-Fatal
Too many Parameters in Segment, Entity Builder Limit	P3CDS-Error
Too many Segments in Package, Entity Builder Limit	P3CDS-Error

(Continued)

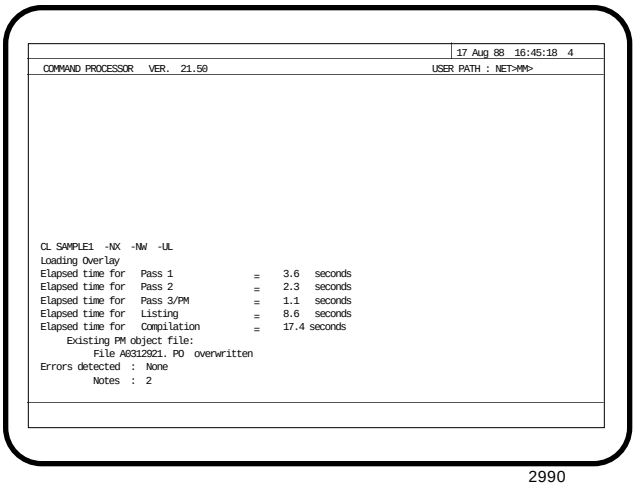
Messages (Continued)	Source/Severity
Too many SEND items	P3MC/PM/APM-Error
Too many statements in a Step/Routine	P3MC/PM/APM-Error
Too many variables in this statement	P3MC/PM/APM-Error
Total Variables exceeds available memory	P3-Fatal
Type (name) expected	P2-Error
Type mismatch between variable type and hardware location	P3MC-Error
Type mismatch	P2-Error
Undefined Enumeration	CX-Error
Undefined Link Error	LK-Fatal
Undefined Symbol	P2-Error
Unexpected Attribute Statement	P2-Error
Unexpected End of File	P1-Error
Unexpected Record	LK-Fatal
Unexpected text after end of program	P1-Severe
Unit Instance name does not exist in Equipment List Object File	P2-Fatal
Unlink required—another block at specified Insertion Point	LK-Fatal
Unlink required—block already on Bound Data Point	LK-Fatal
Unmatched END statement ignored	P2-Severe
Unrecognizable Point Type	CX-Error
Unresolved forward reference	P2-Error
Use standard enumeration @1 instead of declaring enumeration states	P2-Error
Value cannot be Converted	P3-Fatal
Value does not match parameter type	P2-Error
Variables requiring prefetch are not permitted in the Phase alarm clause	P3APM-Error
Wrong Data Type for this Operator	P2-Error
Wrong number of Dimensions	P2-Error

6.2 CORRECTING CL/PM SOURCE FILE ERRORS

The following are the steps used to correct an actual, but simple, CL/PM source file error. A similar process can be used to correct all CL/PM source file errors.

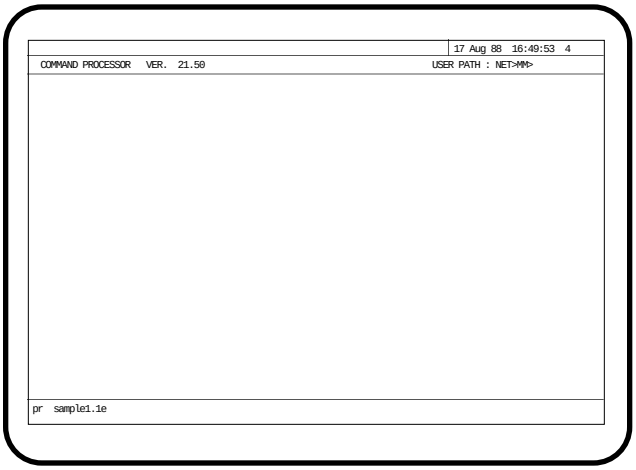
1. This is how the display appeared when compilation of the CL/PM source file was completed. The last line on this display indicates that the Compiler detected one error.

   We need to look at the content of the error listing (.LE) file to find the error.



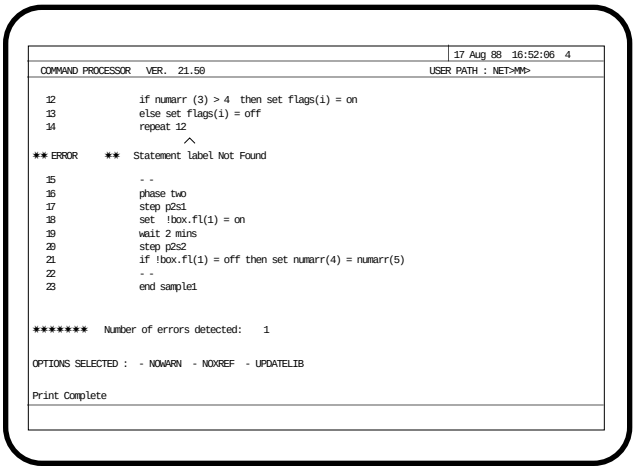
2990

2. We keyed in "PR SAMPLE1.LE" followed by ENTER, to request that the error listing be displayed.



2994

3. The entire .LE file is displayed, line-by-line. When the end of the file is reached, a "Print Complete" message appears. You can now use PAGE FWD and PAGE BACK to page through the file to find the error messages.



2995

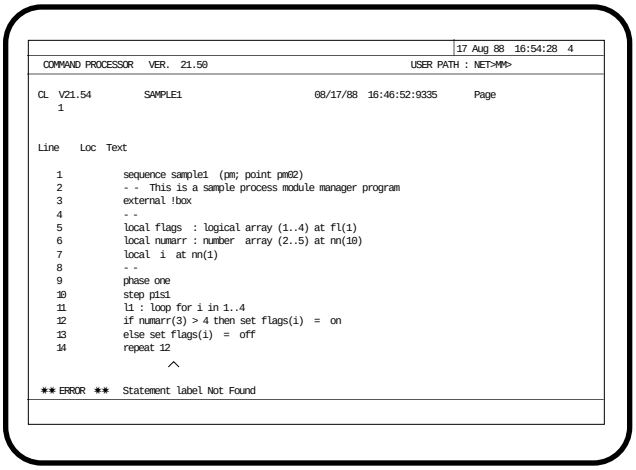
4. We found our error on the first page of the .LE file. The "**ERROR ** Statement Label Not Found" message is just below the line with the error. The "^" points to the beginning of the field with the error.

   The Compiler found that the REPEAT label could not be found. The statement should have read

   ```
   repeat l1
   ```

   instead of

   ```
   repeat l2
   ```



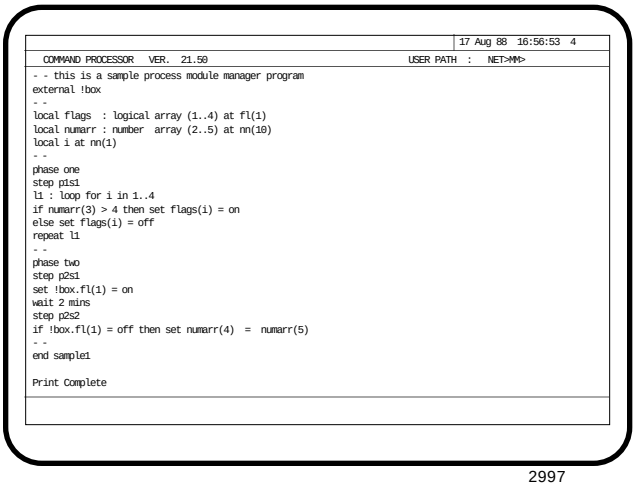
2996

5. Type ED followed by the pathname of the source file that needs to be corrected, example: "SAMPLE1.CL", and then press ENTER (be sure that the .CL file is available in the floppy drive or on an HM).

6. The first 23 lines of the .CL file appear. You can page forward and backward through this display. Our error is about halfway down the first page.

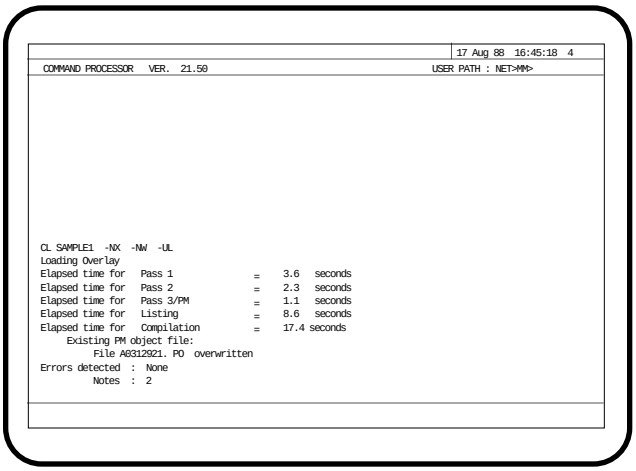
   We keyed in l1 instead of l2 to correct the error. Then we pressed LF, followed by E, and RETURN, to transfer the corrected file to the volume it came from.

7. The display returns to the Command Processor display.



2997

8. Type `cl sample1`, press ENTER to recompile the CL/PM source file. This display shows that this time the compilation took 17.4 seconds, and there are now no errors.



2990

In the example of steps 1 through 7, an error was detected on the second Compiler pass, and pass 3 did not take place. For each pass to complete, the preceding pass must be error free. Once you have corrected an error in an early pass, you may still have more errors to correct in subsequent passes.

6.3 HOW TO DETERMINE THE LOCATION OF A CL/PM RUNTIME ERROR

The CL listing (.LS) file includes program-code location information. This information is in the "Loc" column in the example below.

For CL/PM programs, the number that appears in the "Loc" column for each executable statement is the number that will appear in the statement field of the Process Module Detail Display for the bound data point, when the statement is executed.

The syntax of the IF statement can result in "hidden" statements for which no "Loc" number can be displayed. The statements in the consequent of IF statements can be written on the same line as the rest of the IF statement. This results in more than one executable statement for each CL/PM source code statement. In this case, the number in the "Loc" column is the number for the IF statement. The numbers for the statements in the consequent are not displayed. If it is important to know the "Loc" numbers for such statements, we suggest that each IF statement be written on a separate line, using the continuation character in column one. Here is a program that has two versions of the same IF statement, one containing hidden statements and the other with no hidden statements.

```
Line   Loc          Text

  1            SEQUENCE locnum(PM ; POINT nm05pm01)

  2            PHASE ph01
  3
  4            --EXAMPLE OF AN IF statement with hidden statements

  5      1     IF f1(01) THEN (SEND : "W"; SET f1(02) = OFF)

  6      4     ELSE IF f1(02) THEN SEND : "X"

  7      6     ELSE (SEND : "Y"; SEND : "Z")
  8
  9            --The same IF statement with no hidden statements

 10      8     IF f1(01) THEN

 11      9     &    (SEND : "W";

 12      10    & SET f1(02) = OFF)

 13      11    ELSE IF f1(02) THEN

 14      12    &     SEND : "X"

 15      13    ELSE (SEND : "Y";

 15      14    &     SEND : "W")

 17

 18           END locnum
```

Note that the END statement is counted as an executable statement even though a statement number is not printed on the listing.

Value stores by CL/PM to I/O module parameters are not performed until the next preemption point is encountered. Thus, on these types of stores (called poststores), the program is past the store statement(s) before any store failure can be detected; therefore, the statement "Loc" number displayed for a poststore failure will be that of the following preemption point. All poststore failures fail with error F171 (Communication error in IOL access).

Example:

```
Line   Loc          Text

   1           Step Three
   2     1     IF LG002J03.L(1) = OFF
   3     2     & THEN SET D1713J03.PVFL = ON
   4     3     SEND : "PVFL MAY BE OFF"
   5     4     WAIT 10 SECS
```

In this example, the pvsource of the digital input latched data point D173J03 is auto, so the CL/PM program cannot store to the PVFL. The program fails with an F171 error, but the statement "Loc" number will be 4, not 2.

Index

Index

READER COMMENTS

Honeywell IAC Automation College welcomes your comments and suggestions to improve future editions of this and other publications.

You can communicate your thoughts to us by fax, mail, or toll-free telephone call. We would like to acknowledge your comments; please include your complet name and address

BY FAX: Use this form; and fax to us at (602) 313-4108

BY TELEPHONE: In the U.S.A. use our toll-free number 1*800-822-7673 (available in the 48 contiguous states except Arizona; in Arizona dial 1-602-313-5558).

BY MAIL: Use this form; detach, fold, tape closed, and mail to us.

Title of Publication: **Control Language/Process Manager Data Entry** Issue Date: **9/95**

Publication Number: **PM11-500**

Writer: **Maria Nelson**

COMMENTS: _____

_____

_____

_____

_____

_____

_____

_____

RECOMMENDATIONS:_____

_____

_____

_____

_____

_____

_____

_____

NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

TELEPHONE _____ FAX _____

(If returning by mail, please tape closed; Postal regulations prohibit use of staples.)

Communications concerning technical publications should be directed to:

Automation College
Industrial Automation and Control
Honeywell Inc.
2820 West Kelton Lane
Phoenix, Arizona 85023-3028

FOLD FOLD

From: _____

NO POSTAGE
NECESSARY
IF MAILED
IN THE USA

BUSINESS REPLY MAIL

FIRST CLASS **PERMIT NO. 4332** PHOENIX, ARIZONA

POSTAGE WILL BE PAID BY ....

Honeywell

Industrial Automation and Control
2820 West Kelton Lane
Phoenix, Arizona 85023-3028

Attention: Manager, Quality

FOLD FOLD

Cut Along Line

Additional Comments:

Honeywell

Industrial Automation and Control
Honeywell Inc.
16404 North Black Canyon Highway
Phoenix, Arizona 85023-3099

Helping You Control Your World