

RS-8232-2/58964

Cy1

# SANDIA REPORT

SAND87-1564 UC-28

Unlimited Release

Printed April 1989

## System Manager's Technical Guide for PBFA-II Control/Monitor System



8232-2/068964



00000001 -

Charles E. Simpson

Prepared by  
Sandia National Laboratories  
Albuquerque, New Mexico 87185 and Livermore, California 94550  
for the United States Department of Energy  
under Contract DE-AC04-76DP00789

1 ✓

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

**NOTICE:** This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof or any of their contractors or subcontractors.

Printed in the United States of America  
Available from  
National Technical Information Service  
U.S. Department of Commerce  
5285 Port Royal Road  
Springfield, VA 22161

NTIS price codes  
Printed copy: A03  
Microfiche copy: A01

SAND87-1564

**SYSTEM MANAGER'S TECHNICAL GUIDE FOR  
PBFAII CONTROL/MONITOR SYSTEM**

April 1989

Charles E. Simpson  
Digital Systems Division 7545  
Sandia National Laboratories  
Albuquerque, NM 87185-5800

**ABSTRACT**

This document describes the methodology for implementation and management of the PBFAII Control/Monitor Software System. It also describes software documentation using an automated software engineering tool. A prerequisite to the efficient use of this document is the information contained in a companion document titled "*The Structure and Function of the PBFAII Control System*", SAND87-1563.

## Contents

1. Introduction.....	4
2. Control/Monitor Software Management and Implementation.....	5
2.1 Control/Monitor Software Management Directory Structure.....	5
2.1.1 /CM_CODE Global Directory.....	6
2.1.2 Software Subsystem Subdirectories.....	6
2.1.3 Program Subdirectories.....	6
2.2 C/M Software Program Types.....	7
2.3 Software Compilation and Linking.....	8
2.3.1 Source Code Compilation.....	8
2.3.2 Program Linking.....	9
2.4 Executable System Assembly.....	9
2.4.1 Subsystem Control Unit Assembly .....	9
2.4.2 Graphics Engine Assembly.....	10
2.4.3 Target Host System Assembly.....	10
3.RTE-A Operating System Software.....	11
3.1 System and Snap Files.....	11
3.2 Program Directories.....	11

**Contents (continued)**

4. Control/Monitor Software Documentation .....	12
4.1 Notes on TAGS® Terminology Used.....	12
4.2 Documentation Structure.....	13
4.3 Representation of Files and Memory Partitions.....	13
4.4 Representation of System Processes.....	13
APPENDIX A - PBFAII Control/Monitor Software Management Directory Structure.....	14
APPENDIX B - Compilation Command File Example.....	25
APPENDIX C - Linker Command File Example.....	26
APPENDIX D - PBFAII Control/Monitor Executable System Assembly Directory Structure.....	27
APPENDIX E - Subsystem Control Unit BUILD Answer File Example.....	28
APPENDIX F - Subsystem Control Unit BUILD Command File Example.....	29
APPENDIX G - Graphics Engine BUILD Answer File Example.....	30
APPENDIX H - Graphics Engine BUILD Command File Example.....	31
References.....	32

# SYSTEM MANAGER'S TECHNICAL GUIDE FOR PBFAII CONTROL/MONITOR SYSTEM

## 1. Introduction

The PBFAII C/M Software System involves approximately 100,000 lines of executable code written in Hewlett-Packard's *RTE FORTRAN 77*. The code base at this time consists of approximately 1600 separate compilation units which link into 185 programs. Not counted as separate compilation units but also existing in the code base are approximately 800 'Include' files. The system executes on a distributed computer network consisting of a Target Host, five memory based Subsystem Control Units and two memory based Graphic Engines. Because of the number of software modules, resulting executable programs, implementation procedures and system and snap files involved, a methodology was developed to provide a means for consistently building and implementing the system. Also an automated software engineering system was purchased to document the software.

As is pointed out in the companion document (SAND87-1563), design of the software architecture was influenced greatly by the need to provide a flexible and extendible system in order to support future control criteria changes. Driven by these requirements, the resulting design (from a very abstract viewpoint) consists of a 'non-changing' support substrate and two primary areas within the software structure identified as being where expansion would likely be required. The management methodology and documentation procedures described in the following sections were developed to parallel the actual software architecture as much as possible in order that these systems will also easily support the expected extensions to the code base.

## **2. Control/Monitor Software Management And Implementation**

### **2.1 Control/Monitor Software Management Directory Structure**

During development the C/M software was partitioned into six on-line software subsystems and a utility subsystem as follows:

Control Action Task (CAT)  
Host Satellite Interface (HSI)  
Operator Machine Interface (OMI)  
Subsystem Control Unit (SCU)  
System Data Base (SDB)  
Timing and Firing (T&F)  
Tools and Utilities (TOOLS)

To facilitate system assembly and also to aid in configuration management, a hierarchical file structure was established on the PBFALL development computer (Node 9) to catalog and manage the C/M software. The tree structure for code storage is shown in Appendix A. As can be seen, the root of the structure is a global directory named /CM\_CODE. Immediately below the root directory are subdirectories for each of the five on-line software subsystems and a subdirectory for the utilities software. In all cases except for the OMI subsystem, directly below the subsystem subdirectories are subdirectories each of which contain files specific to one executable program. The OMI subsystem is further partitioned above the program directories. Two basic rules for management of the software are as follows.

1. Each file within the system must be unique in terms of its descriptor which consists of a directory path and a file name.
2. When compiling or linking, modules may be inherited only from the directory where the operation is being performed or from those directories in the search path above.

From a configuration management view, adherence to these two rules produces the result that modification of a module can potentially affect only those modules in the directory where the modification was made or modules in those directories in search paths below.

### **2.1.1 /CM\_CODE Global Directory**

The /CM\_CODE directory contains source and relocatable code that is global to the software system in that any source code or relocatable in this directory is referenced in code or command procedures from more than one of the lower software subsystems. This directory also contains system and snap files that are used to link programs from more than one of the lower software subsystems.

### **2.1.2 Software Subsystem Subdirectories**

Each software subsystem subdirectory contains source and relocatable code that is global only to that particular software subsystem. Any source code or relocatable in a software subsystem subdirectory is referenced in code or command procedures from more than one of the subsystem's lower program subdirectories.

### **2.1.3 Program Subdirectories**

Each program subdirectory is specific to a single program and contains source, relocatable and executable code for that program. These are the only directories within the management structure that contain executable code. Also in the CAT and SCU subsystems program directories there will be instances where the same file name appears in more than one directory. This does not present a problem because, as was pointed out earlier, within this management structure each file must have a unique descriptor which includes its particular directory search path. The use of the same name for similar processes such as control actions, recorders and archivers allows link and compile command files to be basically identical for many of the program directories. Later discussions detail the manner in which specific programs are assembled.

## 2.2 C/M Software Program Types

To aid in a better understanding of the compilation, linking and system assembly procedures to be discussed later it will be pointed out that programs in the C/M software system are of three basic types in regards to the function that each performs. The first of these types is a one-of-a-kind program of which only one copy executes within the entire software subsystem. The second type of program is a completely generic program that performs exactly the same function on more than one of the nodes or as a node specific server on the Target Host. Examples of generic programs are the host communication servers (REPOR) executing within the SCU system, report monitors (ASR00) executing on the Target Host within the HSI system and master archiver monitors (ARCV5) executing on the Target Host within the SDB system. It is necessary to link only one copy of these generic programs and then clone copies as required. The third type type of program is a 'semi-generic' program that , like a generic program, performs basically the same function on more than one node, but must be compiled and/or linked with node or function specific modules. Examples of this third type of program are the real time point recording programs (ARCH1 through ARCV5 and RECO1 through RECO5) executing within the SCU subsystem. Control action programs in the CAT subsystem are somewhat 'semi-generic' in nature in that they are all of the same basic form, however the programs require a greater variety of custom modules in the form of control kernels, facility point lists and message strings.

## 2.3 Software Compilation And Linking

### 2.3.1 Source Code Compilation

Each directory in the C/M software management structure contains a command procedure named `&COMP_<xxx>.CMD` where `<xxx>` corresponds to the name of the directory or to the name of a generic program in the directory. An example of such a compilation command file is shown in Appendix B. With reference to Appendix B note that two basic forms of compiler invocation are used as follows:

Compiler Invocation	Result
<code>ftn7x,\$file,0,-</code>	1) Searches previously specified Working Directory (directory where command file resides) for source code (\$file). 2) Compiles and places resulting relocatable in same Working Directory.
<code>ftn7x,\$source/\$file,0,\$reloc/xx.rel</code>	1) Searches source directory (\$source) for source code (\$file). 2) Finds include files without path names in previously specified working directory. 3) Compiles and places relocatable (xx.rel) in relocatable directory (\$reloc). This form allows generic modules in \$source to be compiled with customized include files in the specified Working Directory. To conform with the software management rules, the source directory (\$source) must be in the path above the Working Directory. Also \$reloc must be the same as the Working Directory.

The entire C/M software system can be compiled by executing a file named `&COMP_PBFAIL.CMD` which exists in the `/CM_CODE` global directory. Command files for compiling individual software subsystems exist in each of the software subsystem directories. These files each make use of the control structure capabilities of the RTE command interpreter to assure that the process will be halted if a compilation error occurs. If an error does occur, the operator is presented the option to 1) ignore the error, 2) correct the error and continue or 3) terminate the compilation process.

### **2.3.2 Program Linking**

Each program subdirectory contains a command file named <PROG>.LOD where <PROG> corresponds to the name of the directory or to the name of a generic program being linked. The command file is used with the linker to place an executable version of the program in the directory. An example of such a command file is shown in Appendix C. The entire C/M software system can be linked by executing a file named LINK\_PBFALL.CMD which exists in the /CM\_CODE global directory. Command files for linking individual software subsystems exist in each of the software subsystem directories. These master link command files also halt the process if a link error occurs.

## **2.4 Executable System Assembly**

After all programs have been linked into their individual subdirectories on the development computer, it is necessary to partition the executable software into node specific subsystems. This is accomplished using another hierarchical file structure on the development computer. This structure is shown in Appendix D. As can be seen, the root of the structure is a global directory named /V2\_SYS\_BUILDR. Immediately below the root directory are subdirectories for each of the five subsystem controller nodes, each of two graphic engine nodes and for simulator software.

### **2.4.1 Subsystem Control Unit Assembly**

Each of the five Subsystem Control Units (SCU) is a memory based system. To assemble these systems each of the SCU subdirectories in the /V2\_SYS\_BUILDR structure contains an answer file to be used by the RTE-A BUILD utility. An example is shown in Appendix E. As can be seen, this answer file RP's basic operating system programs from a /SATPROGRAMS subdirectory and then selectively RP's node specific programs from individual program subdirectories in the /CM\_CODE file structure. Note that the generic programs executing on each satellite can be RP'd in a non node specific manner (RECOR instead of RECO1). It is also here that generic run files in the CAT subsystem subdirectories get assigned specific run file names (/CM\_CODE/CAT/MARX\_CHARGING/CATV2.RUN to MRXCA.RUN). Note also that the answer file specifies the correct system and snap file. Also existing in each of the SCU subdirectories is a command file that invokes the RTE-A BUILD utility with the proper answer file and moves the resulting memory based system to a file manager cartridge on the Target Host. A command file example is shown in Appendix F.

## 2.4.2 Graphics Engine Assembly

Each of the two Graphics Engines (GE) is a memory based system. To assemble these systems each of the GE subdirectories in the /V2\_SYS\_BUILDR structure contains an answer file to be used by the RTE-A BUILD utility. An example is show in Appendix G. As can be seen, this answer file RP's basic operating system programs from a /PGSPROGRAMS subdirectory and then selectively RP's node specific programs from individual program subdirectories in the /CM\_CODE file structure. Note also that the answer file specifies the correct system and snap file. Also existing in each if the GE subdirectories is a command file that invokes the RTE-A BUILD utility with the proper answer file and moves the resulting memory based system to a file manager cartridge on the Target Host. A command file example is shown in Appendix H.

## 2.4.3 Target Host System Assembly

Executable code specific to the Target Host is not handled through the /V2\_SYS\_BUILDR structure but is instead copied from the individual program subdirectories to the Target Host by use of a series of command files residing in the /MANAGER directory on the Target Host. The names of these command files are of the form COPY\_<SUBSYSTEM>\_RUN.COMD where <SUBSYSTEM> corresponds to one of the software subsystems. The generic report monitors and master archiver monitors mentioned earlier are RP'd (cloned) in the Target Host welcome file.

### 3. RTE-A Operating System Software

Implementation of the PBF AII C/M Software System necessitates the generation of three different RTE-A systems. As it becomes necessary to rebuild RTE-A operating system software (as opposed to applications software), some steps must be taken in order that system files, snap files and properly linked operating system programs will be in the directories specified by the command files discussed earlier.

#### 3.1 System and Snap Files

System and snap files for the three systems must be placed in the /CM\_CODE global directory and must be named as follows:

Software Category	System File	Snap File
-----	-----	-----
Target Host	HOST_SYS.SYS	HOST_SNAP.SNP
Subsystem Control Units	SAT_SYS.SYS	SAT_SNAP.SNP
Graphics Engines	PG_SYS.SYS	PG_SNAP.SNP

#### 3.2 Program Directories

RTE-A operating system run files must be placed in global directories named as follows:

Software Category	Global Directory
-----	-----
Target Host	/TARPROGRAMS
Subsystem Control Units	/SATPROGRAMS
Graphics Engines	/PGSPROGRAMS

## 4. Control/Monitor Software Documentation

Because of the size and complexity of the C/M system, an automated software engineering tool was purchased and used to provide a documentation package. The tool is *"The Technology for the Automated Generation of Systems"* (TAGS<sup>®</sup>) marketed by Teledyne Brown Engineering [1]. The tool consists of a graphical system requirements and design language called *"Input/Output Requirements Language"* (IORL<sup>®</sup>) [2] and a series of supporting software packages one of which is a diagnostic analyzer to check that a design is complete and syntactically correct. In the future, a simulation compiler will be available for evaluation of the real time aspects of a design. Because TAGS<sup>®</sup> was used in this case to document an existing system, more emphasis was placed on providing informational documentation rather than documentation in the form required by the IORL<sup>®</sup> system analyzers to evaluate a design.

The C/M software documentation consists of a single computer workstation containing baseline documentation describing high level software components, major data flows and detailed logic for select software subsystems.

### 4.1 Notes On TAGS<sup>®</sup> Terminology Used

Training and manuals have been provided to persons from the PBFAll project and it is assumed that persons using the documentation to any great extent will be familiar with TAGS<sup>®</sup>. For the purposes of this document however the following IORL<sup>®</sup> components are summarized.

SYSTEM	---- unique name referring to an overall design
DOCUMENT	---- related diagrams and tables for one major software component
IORTD	---- input/output relationship and timing diagram
PPD	---- detailed logic flow diagram for a single predefined process
PPT	---- table of parameters local to a single PPD
DSD	---- graphic representation of system data

## **4.2 Documentation Structure**

The name of the IORL<sup>®</sup> SYSTEM for the C/M software documentation is PBFALL. Documents within this system were chosen where possible to represent the source code software subsystems described earlier.

A library structure paralleling to some extent the source code hierarchical file system was also established for cataloging and referencing PPD's. Because TAGS<sup>®</sup> at the time of this writing did not directly provide the capability, the library structure was implemented using a dummy IORTD in the top level document. This library structure can be examined by accessing IORTD-1 in the PBFALL document.

## **4.3 Representation of Files and Memory Partitions**

The C/M software makes extensive use of shared files and extended memory partitions the structures of which are defined in the documentation using PPD's. Because PPD's are normally used to define executable code, they are in this case dummy processes whose primary purpose is to allow the creation of PPT's and DSD's which actually show the data configuration. Because the PPD's are used in higher level logic diagrams, any change to a PPD representing a file or shared memory structure will be reflected as appropriate throughout the documentation.

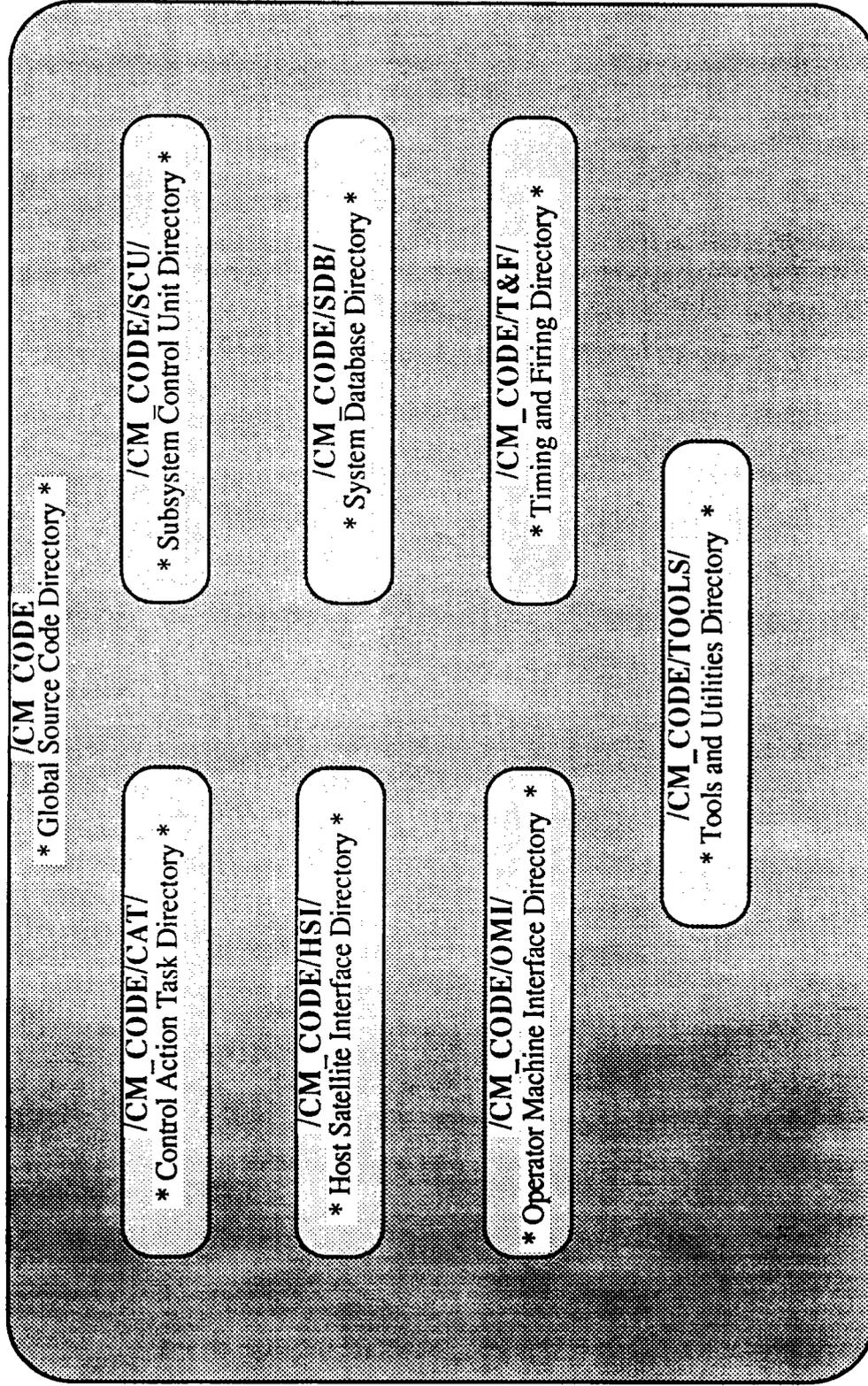
## **4.4 Representation of System Processes**

Operating system service routines (EXEC, DS-1000, IMAGE, etc) were also represented in the documentation with dummy PPD's. This provides a way to consistently specify the services throughout the documentation. Also if in the future it becomes applicable to evaluate the effects of software modifications perhaps using the forthcoming simulation compiler, the dummy PPD's are already in place and need only be expanded.

# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

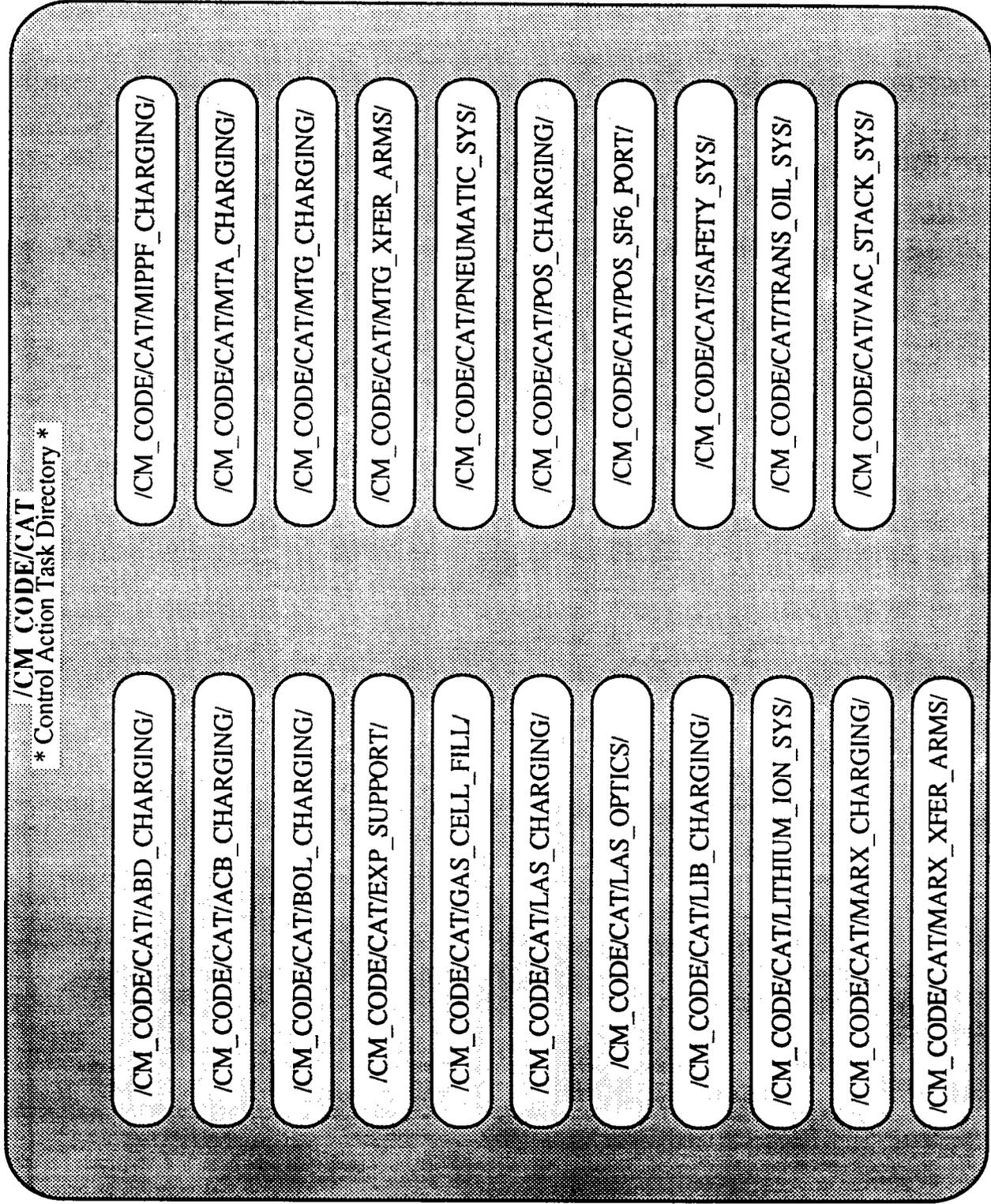
### A.1 Global Source Code Directory and Software Subsystem Subdirectories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

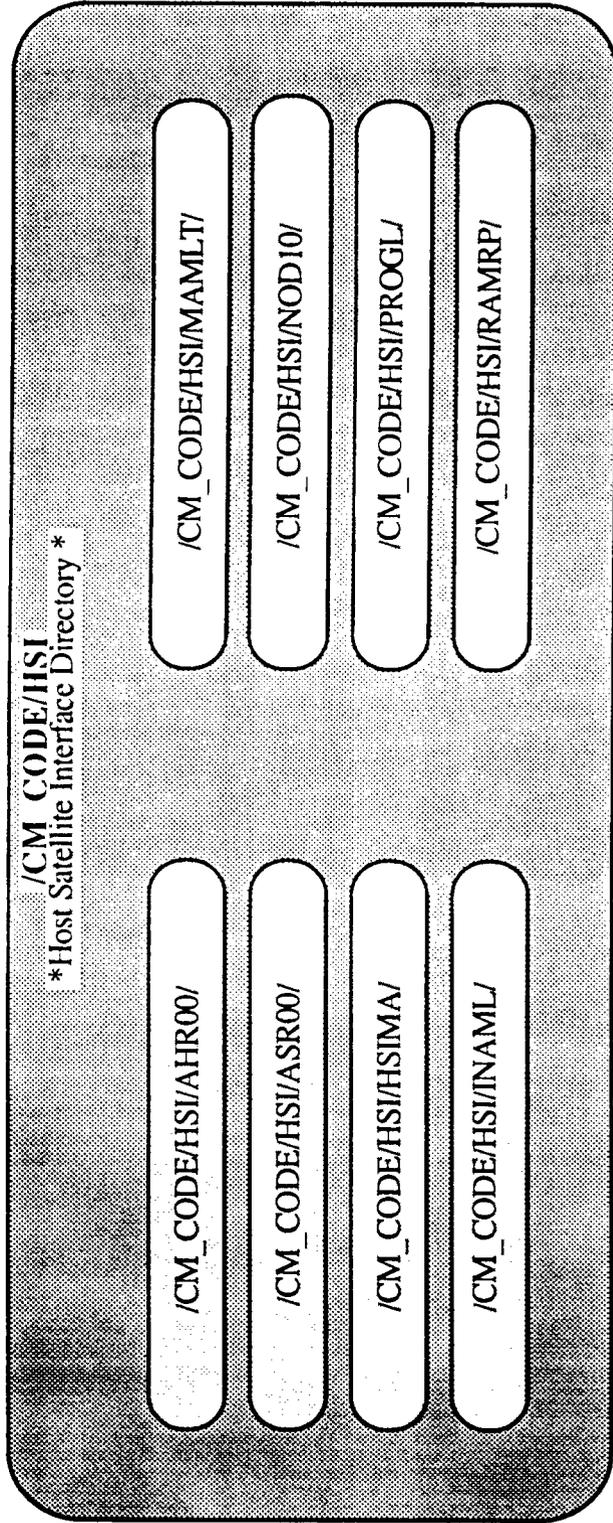
### A.2 CAT Software Subsystem Directory and CAT Program Directories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

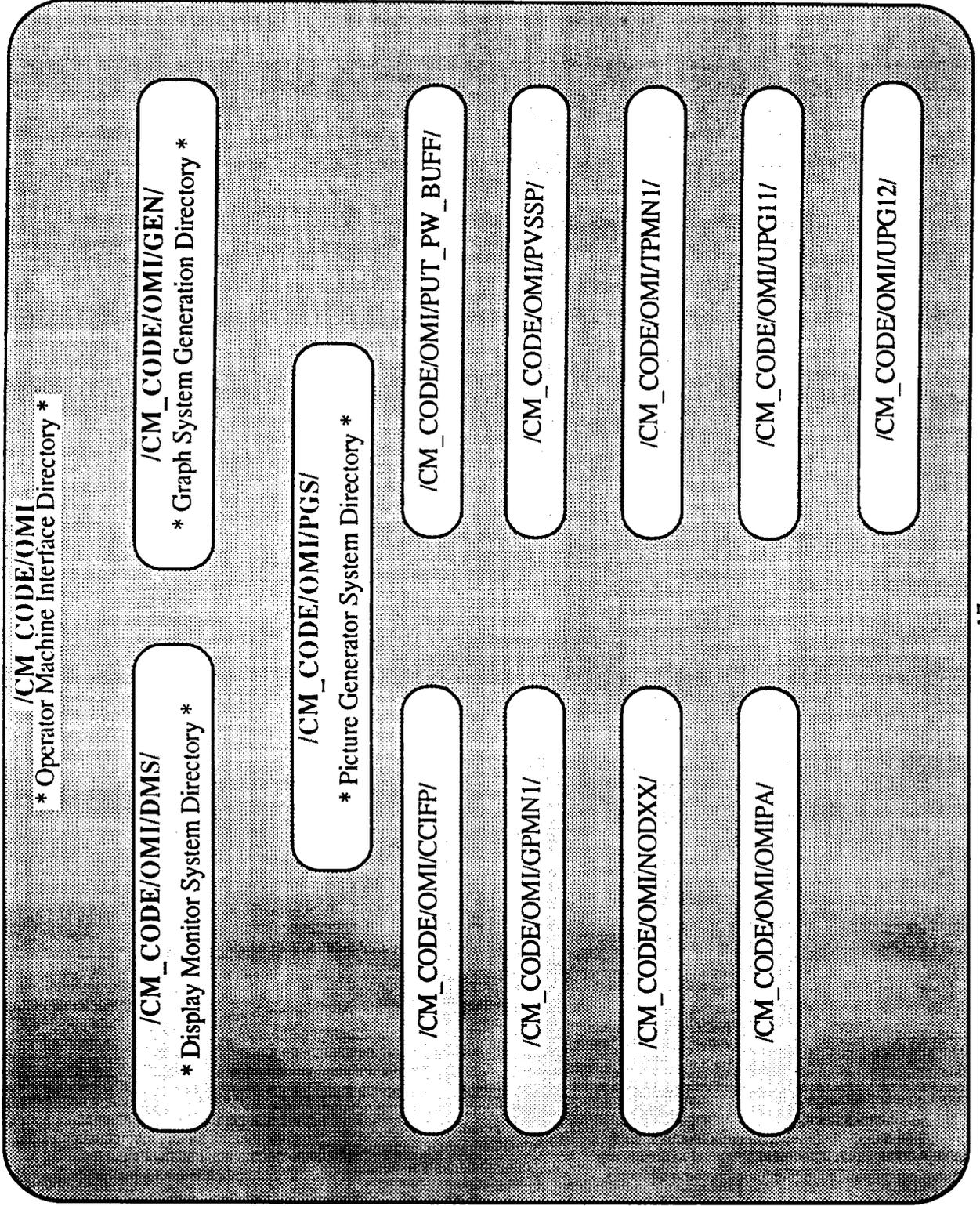
### A.3 HSI Software Subsystem Directory and HSI Program Directories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

### A.4 OMI Software Subsystem Directory Structure



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

### A.5 OMI Display Monitor System Directory And Display Monitor Program Directories

/CM\_CODE/OMI/DMS/

\* Display Monitor System Directory \*

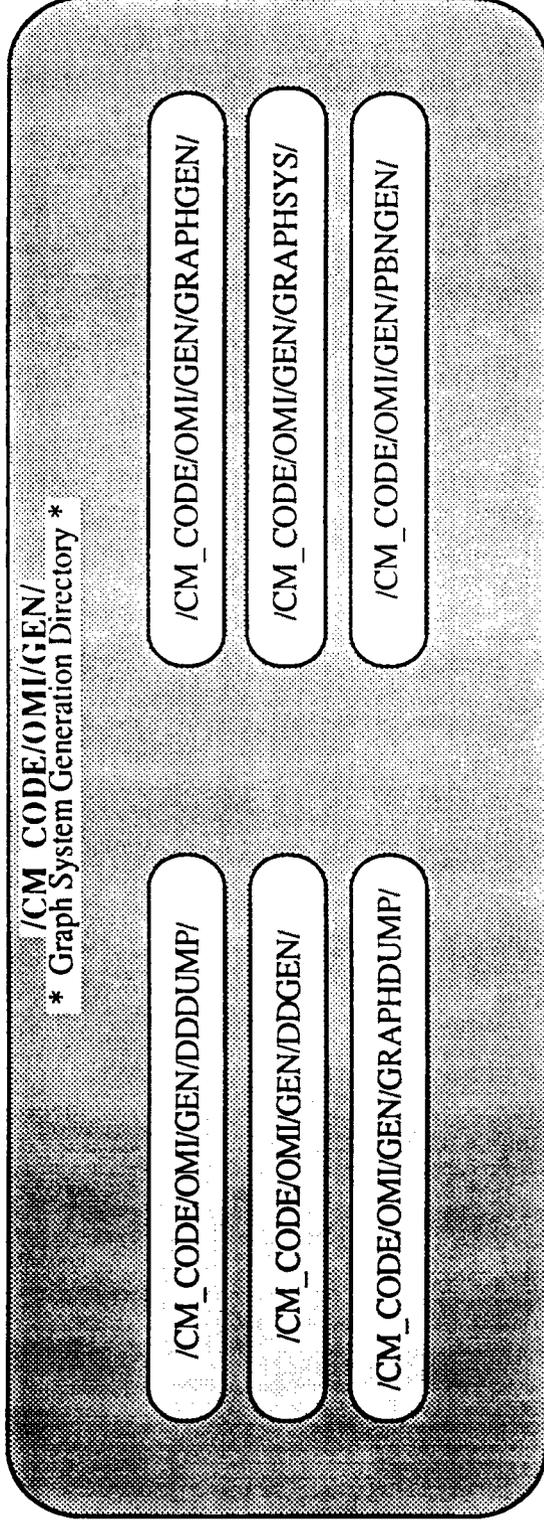
/CM_CODE/OMI/DMS/DM002/	/CM_CODE/OMI/DMS/DM019/	/CM_CODE/OMI/DMS/DM046/
/CM_CODE/OMI/DMS/DM003/	/CM_CODE/OMI/DMS/DM027/	/CM_CODE/OMI/DMS/DM047/
/CM_CODE/OMI/DMS/DM004/	/CM_CODE/OMI/DMS/DM028/	/CM_CODE/OMI/DMS/DM048/
/CM_CODE/OMI/DMS/DM005/	/CM_CODE/OMI/DMS/DM030/	/CM_CODE/OMI/DMS/DM049/
/CM_CODE/OMI/DMS/DM006/	/CM_CODE/OMI/DMS/DM031/	/CM_CODE/OMI/DMS/DM050/
/CM_CODE/OMI/DMS/DM007/	/CM_CODE/OMI/DMS/DM038/	/CM_CODE/OMI/DMS/DM052/
/CM_CODE/OMI/DMS/DM008/	/CM_CODE/OMI/DMS/DM041/	/CM_CODE/OMI/DMS/DM053/
/CM_CODE/OMI/DMS/DM009/	/CM_CODE/OMI/DMS/DM042/	/CM_CODE/OMI/DMS/DM054/
/CM_CODE/OMI/DMS/DM010/	/CM_CODE/OMI/DMS/DM043/	/CM_CODE/OMI/DMS/DM055/
/CM_CODE/OMI/DMS/DM011/	/CM_CODE/OMI/DMS/DM044/	/CM_CODE/OMI/DMS/DM063/
/CM_CODE/OMI/DMS/DM012/	/CM_CODE/OMI/DMS/DM045/	/CM_CODE/OMI/DMS/DM064/

NOTE: DM012 is cloned as DM013 thru DM018  
 DM019 is cloned as DM020 thru DM025  
 DM031 is cloned as DM032 thru DM037  
 DM055 is cloned as DM056 thru DM061

# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

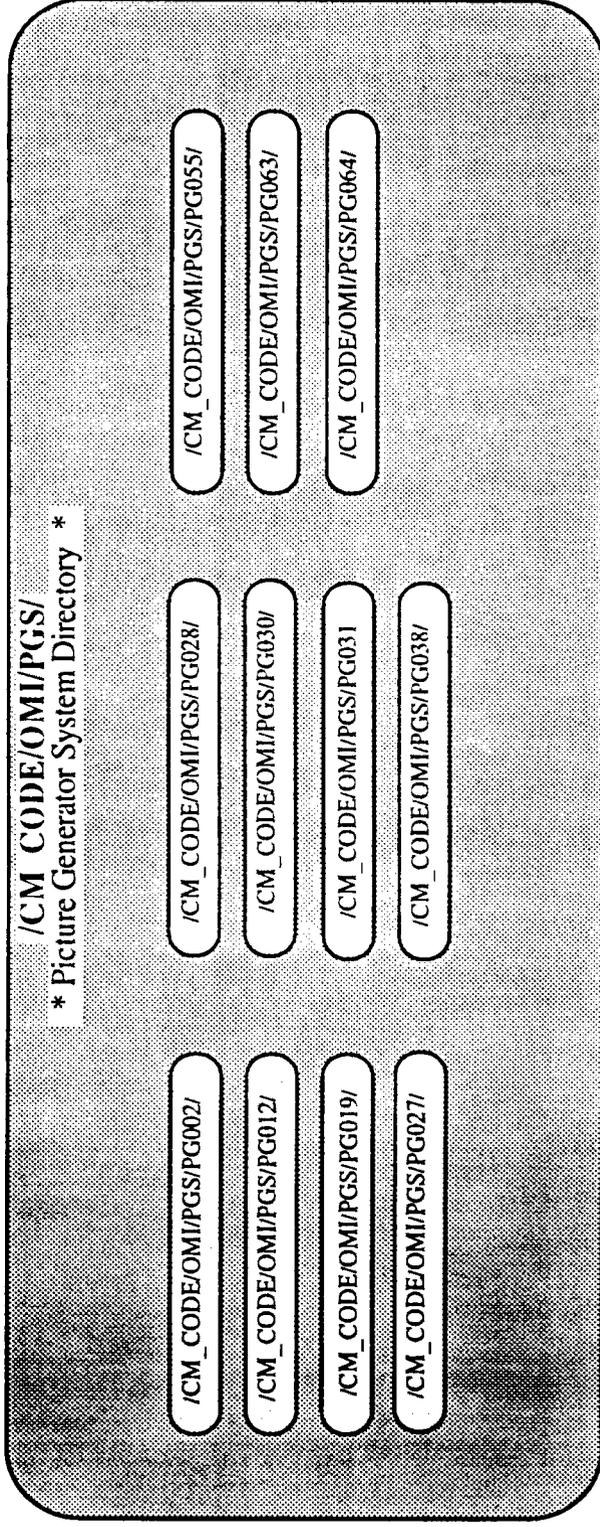
### A.6 OMI Graph System Generation Directory And Graph Generation Directories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

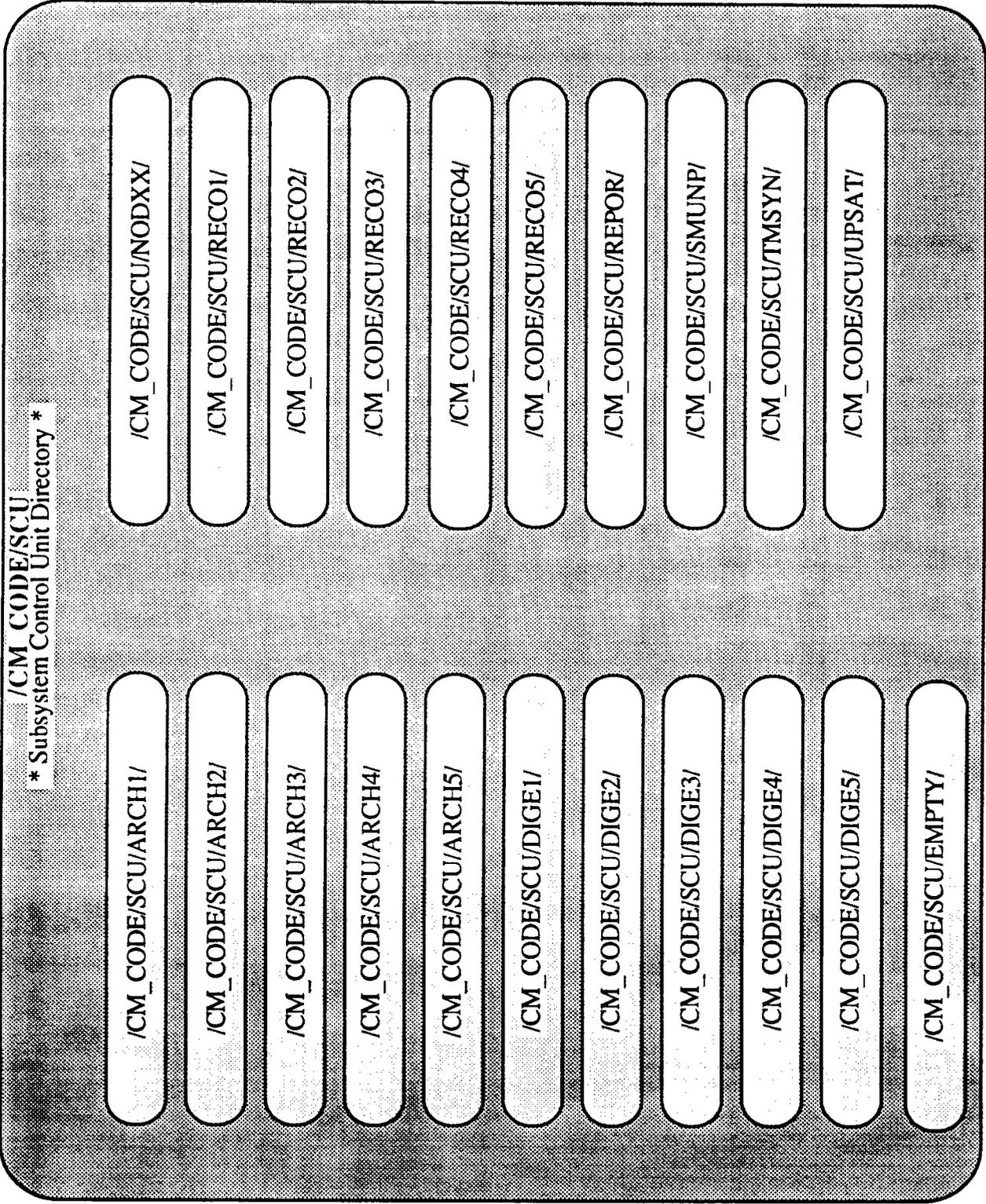
### A.7 OMI Picture Generator Directory And Picture Generator Program Directories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

### A.8 SCU Software Subsystem Directory And SCU Program Directories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

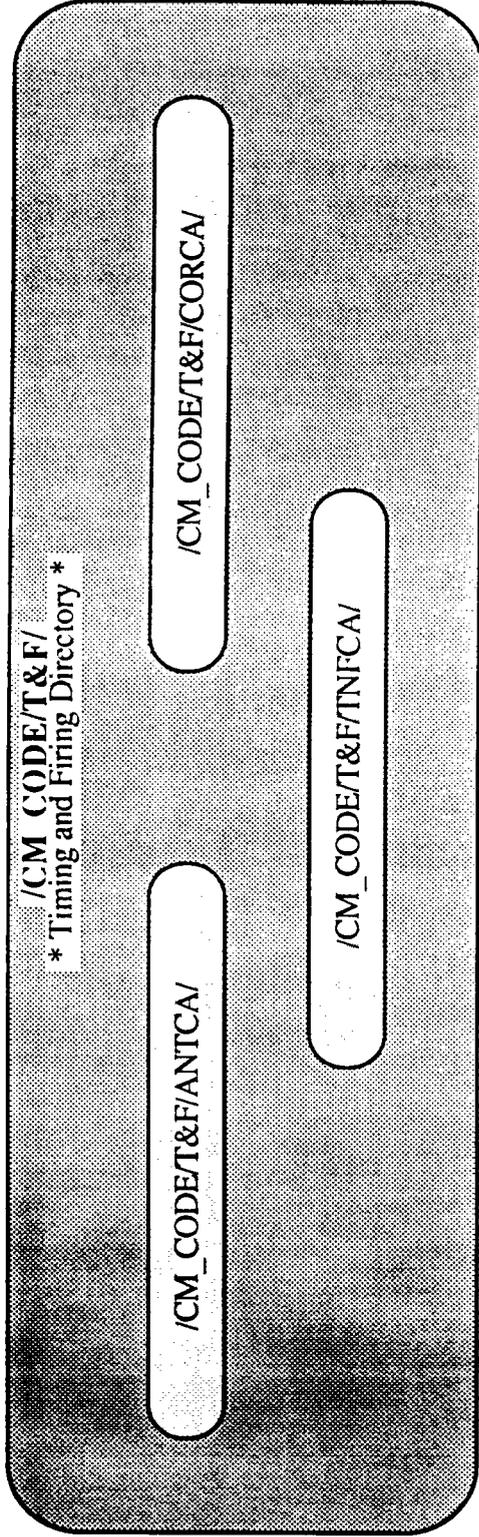
### A.9 SDB Software Subsystem Directory And SDB Program Directories

/CM_CODE/SDB/		
*System Database Directory *		
/CM_CODE/SDB/ACMON/	/CM_CODE/SDB/MBOG/	/CM_CODE/SDB/RMRLA/
/CM_CODE/SDB/ALCLR/	/CM_CODE/SDB/MDSED/	/CM_CODE/SDB/RMRML/
/CM_CODE/SDB/ALMON/	/CM_CODE/SDB/MSMON/	/CM_CODE/SDB/RMRPA/
/CM_CODE/SDB/ARCV5/	/CM_CODE/SDB/PMMI/	/CM_CODE/SDB/SET_KNOBS/
/CM_CODE/SDB/BUAMF/	/CM_CODE/SDB/FSHOT/	/CM_CODE/SDB/SIMDR/
/CM_CODE/SDB/CFINT/	/CM_CODE/SDB/PTEMP/	/CM_CODE/SDB/SIMTB/
/CM_CODE/SDB/EEFIL/	/CM_CODE/SDB/PTLOG/	/CM_CODE/SDB/SMCI/
/CM_CODE/SDB/FAMON/	/CM_CODE/SDB/PVDBM/	/CM_CODE/SDB/SOSI/
/CM_CODE/SDB/FPNCO/	/CM_CODE/SDB/PVSRV/	/CM_CODE/SDB/SQMGR/
/CM_CODE/SDB/GRBLD/	/CM_CODE/SDB/P_RCIF/	/CM_CODE/SDB/SSRG/
/CM_CODE/SDB/HBACC/	/CM_CODE/SDB/RCIF/	/CM_CODE/SDB/STMON/
/CM_CODE/SDB/IOCNF/	/CM_CODE/SDB/RECAL/	/CM_CODE/SDB/SUARF/
/CM_CODE/SDB/LNKER/	/CM_CODE/SDB/RMRGA/	/CM_CODE/SDB/SUPER/
		/CM_CODE/SDB/SYEMN/

# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

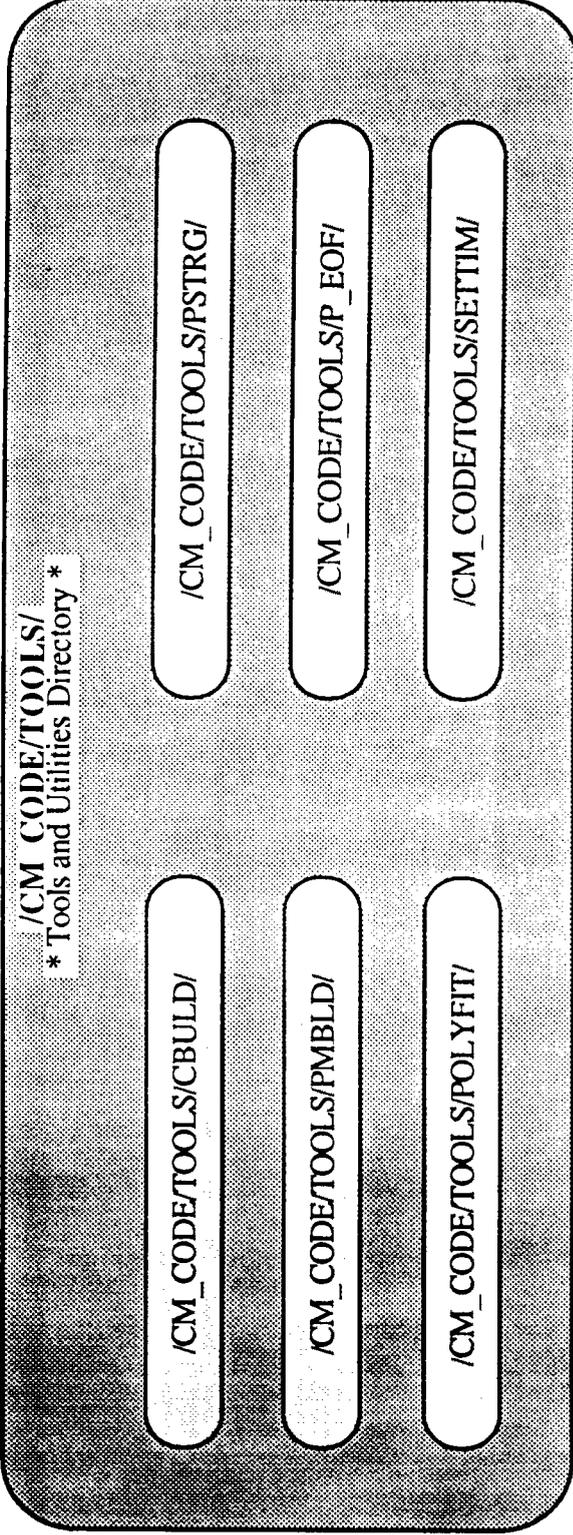
### A.10 T&F Software Subsystem Directory and T&F Program Directories



# APPENDIX A

## PBFAII Control/Monitor Software Management Directory Structure

### A.11 TOOLS Software Subsystem Directory and TOOLS Program Directories



# APPENDIX B

## Compilation Command File Example

### B.1 Portion of CATV2 Compilation Command File

```
set log = on
*-----*
* File Name: &COMP_CATV2.CMD
* Purpose:   Compiles Code in /CM_CODE/CAT/marx_charging directory
*-----*
set log = off
*
set status = good
*
set source = /cm_code/cat
set reloc = /cm_code/cat/marx_charging
*
set file = CATV2.FTN
set STATUS = bad
WHILE IS $STATUS EQ bad
DO
  echo COMPILING >>>-----> $file
  ftn7x,$file,0,-
  IF IS $return1 NE 0;
  THEN TR /cm_code/cmd_continue.cmd
  ELSE SET STATUS = good
  FI
DONE
IF IS $STATUS NE GOOD; THEN RETURN ; FI
*
      .
      .
*
set file = AUTO_MODE.FTN
set STATUS = bad
WHILE IS $STATUS EQ bad
DO
  echo COMPILING >>>-----> $file
  ftn7x,$source/$file,0,$reloc/auto_mode.rel
  IF IS $return1 NE 0;
  THEN TR /cm_code/cmd_continue.cmd
  ELSE SET STATUS = good
  FI
DONE
IF IS $STATUS NE GOOD; THEN RETURN ; FI
      .
      .
*
echo >>>=====>> $reloc COMPILATION COMPLETE
```

# APPENDIX C

## Linker Command File Example

### C.1 CATV2 Linker Command File

```
*-----*
* File Name: CATV2.LOD
* Purpose:  Loads Version 2 Control Action
*-----*
*
ll,CATV2.map
su,of
lc
*
li,$MCLIB
li,/CM_CODE/CAT/CATV2_LIBRARY.lib
*
re,ALARM_CATV2S.rel
re,AUTO_MODE.rel
re,BUILD_INPUT_CHAIN.rel
re,BUILD_OUTPUT_CHAIN.rel
re,CATV2.rel
re,CNTRL_KERNAL.rel
re,DISABLE_PNTS.rel
re,DOREPORT.rel
re,ENABLE_PNTS.rel
re,GET_RMFT_BUFFER.rel
re,GET_STATE.rel
re,INITIALIZE_MODE.rel
re,LOCAL_MODE.rel
re,MANUAL_MODE.rel
re,MRX_CHRG_CMD_DEC.rel
re,REPORT_GROUP.rel
re,REPORT_RECORD.rel
re,SEND_MILESTONES.rel
re,SET_CONTROL.rel
re,VECTOR_STATE.rel
*
re,/CM_CODE/ERRER.rel
re,/CM_CODE/PRINT_ABREG_ERR.rel
*
re,/CM_CODE/CAT/BI_VECTOR_STATE.rel
*
se,/CM_CODE/CAT/CATV2_LIBRARY.LIB
se,$MCLIB
*
en
```

# APPENDIX D

## PBFAII Control/Monitor Executable System Assembly Directory Structure

### D.1 Global System Assembly Directory and Subdirectories

/V2\_SYS\_BUILD  
\* Global Assembly Directory \*

/V2\_SYS\_BUILD/ACMP1/

\* Accelerator Components #1 Directory \*

/V2\_SYS\_BUILD/ACMP2/

\* Accelerator Components #2 Directory \*

/V2\_SYS\_BUILD/FACSAF/

\* Facility Safety Directory \*

/V2\_SYS\_BUILD/FPS1/

\* Fluid Process System #1 Directory \*

/V2\_SYS\_BUILD/FPS2/

\* Fluid Process System #2 Directory \*

/V2\_SYS\_BUILD/PGS11/

\* Node 11 Picture Generators Directory \*

/V2\_SYS\_BUILD/PGS12/

\* Node 12 Picture Generators Directory \*

/V2\_SYS\_BUILD/SIM/

\* Software Simulators Directory \*

# APPENDIX E

## Subsystem Control Unit BUILD Answer File Example

### E.1 Portion of BUILD Answer File For ACMP1 Subsystem Control Unit

```

*ACMP1:::7272
/CM_CODE/SAT_SNAP.SNP
/CM_CODE/SAT_SYS.SYS
YES,,,,,,
909,,,,,,
RP,/SATPROGRAMS/DRTR.RUN,D.RTR,
RP,/SATPROGRAMS/DSRTR.RUN,,
RP,/SATPROGRAMS/GRPM.RUN,,
RP,/SATPROGRAMS/#SEND.RUN,,
RP,/SATPROGRAMS/QCLM.RUN,,
RP,/SATPROGRAMS/RTRY.RUN,,
.
.
RP,/SATPROGRAMS/COMND.RUN,,
RP,/SATPROGRAMS/DINIT.RUN,,
RP,/SATPROGRAMS/DEMON.RUN,,
RP,/SATPROGRAMS/EMPTY.RUN,,
RP,/SATPROGRAMS/WH.RUN,
RP,/CM_CODE/SCU/NODXX/NODXX.RUN,NOD01,
RP,/CM_CODE/SCU/UPSAT/UPSAT.RUN,,
ST,0,1,I0,0,1,,
RP,/CM_CODE/SDB/IOCNF/IOCNF.RUN,,
RP,/CM_CODE/SCU/TMSYN/TMSYN.RUN,,
RP,/CM_CODE/SCU/RECO1/RECOR.RUN,,
PR,40
RP,/CM_CODE/SCU/ARCH1/ARCHV.RUN,,
PR,99
RP,/CM_CODE/SCU/REPOR/REPOR.RUN,,
PR,34
RP,/CM_CODE/SCU/DIGE1/DIGEM.RUN,,
PR,36
RP,/CM_CODE/CAT/MARX_CHARGING/CATV2.RUN,MRXCA.RUN
PR,38
RP,/CM_CODE/CAT/MTG_CHARGING/CATV2.RUN,MTGCA.RUN
PR,38
RP,/CM_CODE/CAT/MTA_CHARGING/CATV2.RUN,MTACA.RUN
PR,38
RP,/CM_CODE/CAT/LAS_CHARGING/CATV2.RUN,LTSCA.RUN
PR,38
RP,/CM_CODE/CAT/MARX_XFER_ARMS/CATV2.RUN,MRXTA.RUN
PR,38
RP,/CM_CODE/CAT/MTG_XFER_ARMS/CATV2.RUN,MTGTA.RUN
PR,38
RP,/CM_CODE/CAT/PNEUMATIC_SYS/CATV2.RUN,PNUCA.RUN
PR,38
RP,/CM_CODE/CAT/LAS_OPTICS/CATV2.RUN,LTSOP.RUN
PR,38
PT
/

```

```

* AUTO PARTION CONSTRUCNTION
* PHYSICAL MEMORY SIZE
* I/O HANDLER
* DS I/O HANDLER
* GENERAL REQUEST PROCESSOR
* UPDATE TRANSMITTER
* ERROR LOG TRANSMITTER
* RETRANSMIT_ON_ERROR
.
.

```

```

* REMAT COMMAND PROCESSOR
* NODE INITIALIZATION PROCESS
* 2250 DIAGNOSTIC PROGRAM
* 32 PAGE DUMMY PROGRAM
* SYSTEM STATUS
* DINIT INITIALIZE PROGRAM
* SCU STARTUP PROGRAM
* AUTO START UPSAT
* MC2250 CONFIG QUERY
* SATELLITE TIME SYNC PROGRAM
* POINT STATE DATA RECORDER
.
* POINT STATE DATA ARCHIVER
.
* HOST REPORT SERVER
.
* DIGITAL EVENT MONITOR

```

# APPENDIX F

## Subsystem Control Unit BUILD Command File Example

### F.1 ACMP1 BUILD Command File

```
*-----*
* File Name: /V2_SYS_BUILDR/ACMP1/BUILD_ACMP1.CMD
* Purpose:  Builds Node 1 Memory Based 2250 System
*-----*
*
WD /V2_SYS_BUILDR/ACMP1
PU *ACMP1
RU BUILD, ACMP1.ANS,ACMP1.LST
CO *ACMP1 *ACMP1:xx:20::7272>10 D
*
*
```

# APPENDIX G

## Graphics Engine BUILD Answer File Example

### G.1 Portion of BUILD Answer File For Node 11 Graphics Engine

```

*PGS11:::8944
/CM_CODE/PG_SNAP.SNP
/CM_CODE/PG_SYS.SYS
YES,,,,,,,,
1118,,,,,,,,
RP/PGSPROGRAMS/GRPM.RUN,,
RP/PGSPROGRAMS/#SEND.RUN,,
RP/PGSPROGRAMS/QCLM.RUN,,
RP/PGSPROGRAMS/RTRY.RUN,,
.
.
RP/PGSPROGRAMS/DINIT.RUN,,
RP/PGSPROGRAMS/EMPTY.RUN,,
RP/PGSPROGRAMS/WH.RUN,
RP/CM_CODE/OMI/NODXX/NODXX.RUN,NOD11,
RP/CM_CODE/OMI/UPPG11/UPPGN.RUN,,
ST,1,1,11,10,,
.
RP/CM_CODE/OMI/PGS/PG055/PG055.RUN,PG057
RP/CM_CODE/OMI/PGS/PG055/PG055.RUN,PG058
RP/CM_CODE/OMI/PGS/PG055/PG055.RUN,PG059
RP/CM_CODE/OMI/PGS/PG055/PG055.RUN,PG060
RP/CM_CODE/OMI/PGS/PG002/PG002.RUN,PG002
.
.
RP/CM_CODE/OMI/PGS/PG002/PG002.RUN,PG011
RP/CM_CODE/OMI/PGS/PG002/PG002.RUN,PG041
.
.
RP/CM_CODE/OMI/PGS/PG002/PG002.RUN,PG053
RP/CM_CODE/OMI/PGS/PG012/PG012.RUN,PG014
RP/CM_CODE/OMI/PGS/PG012/PG012.RUN,PG015
RP/CM_CODE/OMI/PGS/PG012/PG012.RUN,PG016
RP/CM_CODE/OMI/PGS/PG012/PG012.RUN,PG017
RP/CM_CODE/OMI/PGS/PG019/PG019.RUN,PG021
RP/CM_CODE/OMI/PGS/PG019/PG019.RUN,PG022
RP/CM_CODE/OMI/PGS/PG019/PG019.RUN,PG023
RP/CM_CODE/OMI/PGS/PG019/PG019.RUN,PG024
RP/CM_CODE/OMI/PGS/PG030/PG030.RUN,PG030
RP/CM_CODE/OMI/PGS/PG031/PG031.RUN,PG033
RP/CM_CODE/OMI/PGS/PG031/PG031.RUN,PG034
RP/CM_CODE/OMI/PGS/PG031/PG031.RUN,PG035
RP/CM_CODE/OMI/PGS/PG031/PG031.RUN,PG036
RP/CM_CODE/OMI/PGS/PG038/PG038.RUN,PG038
RP/CM_CODE/OMI/PGS/PG031/PG039.RUN,PG039
RP/CM_CODE/OMI/PGS/PG040/PG040.RUN,PG040
RP/CM_CODE/OMI/PGS/PG062/PG062.RUN,PG062
PT
/E

```

- \* AUTO PARTION CONSTRUCNTION
- \* PHYSICAL MEMORY SIZE
- \* GENERAL REQUEST PROCESSOR
- \* UPDATE TRANSMITTER
- \* ERROR LOG TRANSMITTER
- \* RETRANSMIT\_ON\_ERROR
- .
- .
- \* NODE INITIALIZATION PROCESS
- \* 32 PAGE DUMMY PROGRAM
- \* SYSTEM STATUS
- \* DINIT INITIALIZE PROGRAM
- \* PG STARTUP PROGRAM
- \* AUTO START UPPGN

# APPENDIX H

## Graphics Engine BUILD Command File Example

### H.1 Node 11 Graphics Engine BUILD Command File

```
*-----*
* File Name: /V2 SYS BUILDR/PGS11/BUILD PGS11.CMD
* Purpose:  Builds Node 11 Memory Based Graphics Engine
*-----*
*
PU *PGS11
RU BUILD, PGS11.ANS,PGS11.LST
CO *PGS11 *PGS11:xx:20::8944>10 D
*
*
```

## References

1. Teledyne Brown Engineering, Cummings Research Park, Huntsville, Alabama, *"Technology For The Automated Generation of Systems (TAGS)"*.
2. Teledyne Brown Engineering, Cummings Research Park, Huntsville, Alabama, *"Input/Output Requirements Language Reference Manual"*.

DISTRIBUTION:

1266 D. R. Nations (2)  
1266 E. J. Mader (EG&G) (5)  
1266 D. D. Bloomquist (2)  
3141 S. A. Landenberger (5)  
3151 W. I. Klein (3)  
7500 D. M. Olson  
3154-1 C. L. Ward for DOE/OSTI (8)  
7540 T. B. Lane  
7545 J. L. Mortley  
8524 J. A. Wackerly  
7545 C. E. Simpson (5)