

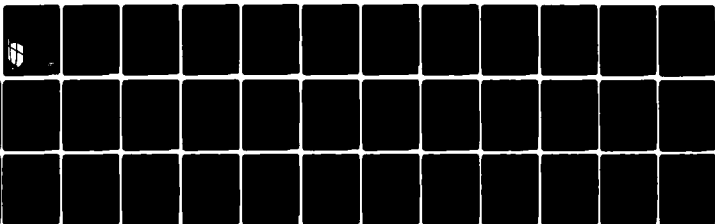
AD-A093 153

CONSTRUCTION ENGINEERING RESEARCH LAB (ARMY) CHAMPAIGN IL F/G 9/2
AUTOMATED DOCUMENTATION SYSTEM (ADS) STUB GENERATOR: DESCRIPTIO--ETC(U)
OCT 80 L LAWRIE, J BAUGH
CERL-TR-E-167

UNCLASSIFIED

NL

1 of 1
AD-A093 153



END
DATE
FILMED
1-81
DTIC

construction
engineering
research
laboratory



United States Army
Corps of Engineers
...Serving the Army
...Serving the Nation

TECHNICAL REPORT E-167
October 1980

**AUTOMATED DOCUMENTATION SYSTEM (ADS)
STUB GENERATOR: DESCRIPTION AND
USER INSTRUCTIONS**

(Comprehensive Standard for Software Development)

LEVEL II

2
JC

by
Linda Lawrie
Jean Baugh

DTIC
SELECTED
DEC 17 1980

E



Approved for public release; distribution unlimited.

80 2 15 122

AD A093153

DDC FILE

The contents of this report are not to be used for advertising, publication, or promotional purposes. Citation of trade names does not constitute an official indorsement or approval of the use of such commercial products. The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

***DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED
DO NOT RETURN IT TO THE ORIGINATOR***

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CERL-TR-E-167	2. GOVT ACCESSION NO. AD-A093 753	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AUTOMATED DOCUMENTATION SYSTEM (ADS) STUB GENERATOR: DESCRIPTION AND USER INSTRUCTIONS		5. TYPE OF REPORT & PERIOD COVERED FINAL Rept.
7. AUTHOR(s) Linda Lawrie Jean Baugh		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS U. S. ARMY CONSTRUCTION ENGINEERING RESEARCH LABORATORY P.O. Box 4005, Champaign, IL 61820		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS RDT&E Program 6.27.25A
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE October 1989
		13. NUMBER OF PAGES 35
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Copies are obtainable at the National Technical Information Service Springfield, VA 22151		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) computer program documentation Automated Documentation System		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report describes a subsystem of the Automated Documentation System (ADS) called the Stub Generator. The Stub Generator lets software development managers define which ADS documentation sections must be in the final software code. It also lets the manager monitor updated ADS documentation as initial software codes are amended and extended. The programmer is given a convenient way of developing a system in a top-down structural manner through the generation of standard program stubs.		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 405279

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Block 20 continued.

This report is written for the data processing professional and assumes the reader is familiar with Control Data Corporation (CDC) FORTRAN and UPDATE and the ADS computer system as described in the U.S. Army Construction Engineering Research Laboratory (CERL) Technical Report E-147, *The Automated Documentation System — User Manual* (CERL, February 1979).

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

FOREWORD

This research was conducted by the Energy Systems Division (ES) of the U.S. Army Construction Engineering Research Laboratory (CERL) for the Engineering and Scientific Division of the Engineering and Data Systems Office, Department of the Army, under RDT&E Program 6.27.25A, "Engineering Software Development", Task 02, "Comprehensive Standard for Software Development." Mr. Gene Manning was the Technical Monitor.

Mr. Douglas C. Hittle was the CERL Principal Investigator. Administrative support was provided by Dr. D. J. Leverenz and Mr. R. G. Donaghy, Chief of CERL-ES.

The ADS Stub Generator program was authored by Ms. J. Baugh and Ms. L. Lawrie. Appreciation is expressed to Ms. M. L. Scala for her help in writing this report.

COL Louis J. Circeo is Commander and Director of CERL, and Dr. L. R. Shaffer is Technical Director.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/ _____	
Availability Codes	
Dist	Avail and/or special
A	

CONTENTS

DD FORM 1473	1
FOREWORD	3
1 INTRODUCTION	5
Background	
Objective	
Mode of Technology Transfer	
2 OVERVIEW	5
First Session	
History File	
Creating Program Modules	
File List and Function Description	
Options	
Help	
3 CAPABILITIES AND LIMITATIONS	8
Flexible Features	
Limitations	
Output	
4 COMMAND LANGUAGE AND ONLINE HELP	9
5 GENERAL HINTS	10
6 SUMMARY	11
APPENDIX A: Sample First Session	12
APPENDIX B: Sample Run	17
APPENDIX C: Instructions for Beginning a Stub Generator Session on BCS	25
APPENDIX D: Batch Job Output	26
DISTRIBUTION	

AUTOMATED DOCUMENTATION SYSTEM (ADS) STUB GENERATOR: DESCRIPTION AND USER INSTRUCTIONS

1 INTRODUCTION

Background

A useful software tool is the product of a carefully managed and executed software development project.

In 1979, the U.S. Army Construction Engineering Research Laboratory (CERL) introduced a computer system to help guarantee successful management and execution of Army software development. This system, called the Automated Documentation System (ADS), allows internal and external documentation to proceed *simultaneously* with software development.¹

The ADS Stub Generator is a subsystem of ADS. It lets the software project manager define which ADS documentation sections must be in the final software code. It also lets the manager monitor updated ADS documentation as the initial software code is amended and extended.

This report describes how to use the ADS subsystem Stub Generator. The text is geared to the data processing professional and assumes the reader is familiar with Control Data Corporation (CDC) FORTRAN and UPDATE and the ADS computer system as described in CERL Technical Report (TR) E-147.

Objective

The objective of this study was to develop a method to ensure structured program development and adequate documentation by automatically generating standard program stubs.

Mode of Technology Transfer

The ADS Stub Generator is available from the Boeing Computer Services (BCS) Company under the U.S. Army Corps of Engineers contract for scientific and engineering teleprocessing.

¹Linda Lawrie, *The Automated Documentation System — User Manual*, Technical Report E-147/ADA067203 (U.S. Army Construction Engineering Research Laboratory, February 1979).

2 OVERVIEW

Stubs are skeletal routines which contain routine headers, preliminary ADS documentation, type statements, common block references, initial debugging output, and RETURN/ END statements. The Stub Generator relieves some of the drudgery of creating these routine stubs by building skeleton FORTRAN modules containing standard UPDATE directives, basic FORTRAN code, and ADS comments from data entered by the user. These skeleton modules are kept in the UPDATE source program library. They can be amended or extended as the system is developed.

First Session

During the first Stub Generator session, the user (usually the software project team leader) sets the desired level of internal program documentation by specifying the required ADS documentation sections (see Appendix A). At this time, the team leader can also set basic "system parameters" for the software being developed; once set, some of these system parameters cannot be changed. For example:

1. The name of the system (in the form of a three-character prefix which is used to generate file names)
2. The password (if any) associated with system files
3. The control character to be used on UPDATE directives.

A complete list of parameters which can and cannot be changed after the first Stub Generator session is in Appendix A, Figure A1. Appendix A also gives a sample of an initial Stub Generator session.

History File

The system history file is maintained by the Stub Generator. It has all the data needed to describe a software system to the Stub Generator (Figure 1). It includes:

1. A list of all options in effect during the last Stub Generator session
2. Data on modules already named or described to the Stub Generator subsystem
3. Data needed to create instructions to update and sort the source program library.

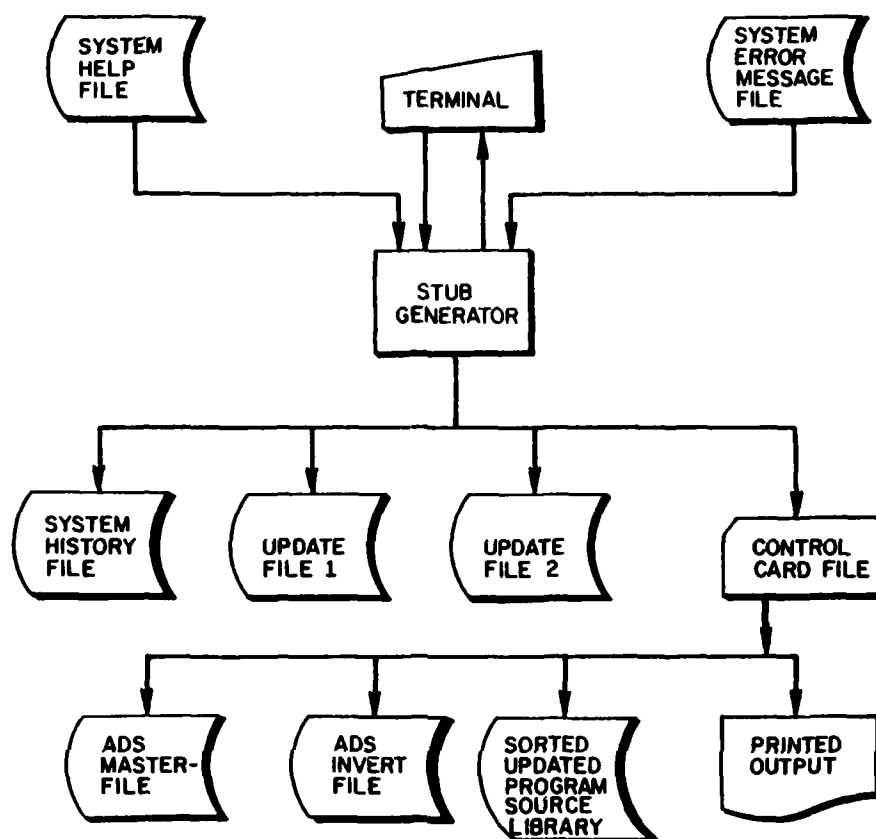


Figure 1. Stub Generator system overview.

Creating Program Modules

During the creation of program modules, the Stub Generator prompts the programmer for information needed to define each program module. The Stub Generator ensures adequate internal program documentation by:

1. Prompting with the ADS section headers in the order specified by the software manager during the first session.
2. Verifying the specified ADS section for the type of module being defined. After the required ADS comments are given to the Stub Generator, the programmer may input additional ADS sections, if desired.

Two UPDATE files can be created during each Stub Generator session. One has preliminary UPDATE directives (such as *ID or *AF) and any modules defined as COMDECKS. The other has UPDATE directives and source code for any modules defined as DECKS.

Both files are kept on disk until a programmer tells the Stub Generator to put them in the source program library. (For more details, see the explanation of the @D command in Chapter 4.)

After the Stub Generator is told to update the source program library, it generates a file of control cards. These cards are used to begin the proper job stream. (This file is sent to the job queue internally. However, the Stub Generator will save this file as

a disk file to be batched by hand, if requested.) Once the job begun by these control cards is complete, any update files will be purged; the newly sorted source program library will be saved and a printed report of the final source library will be output.

If the option to run ADS is in effect when the control card file is created, the control card job will also create and keep two ADS files. It will also produce any ADS reports requested.

A sample programmer session is given in Appendix B. (Actual usage of the Stub Generator will differ from facility to facility. Commands for beginning a session on Boeing Computer Services' (BCS) EKS system are given in Appendix C.)

File List and Function Description

Figure 1 shows the overall file system for the Stub Generator.

The system help file contains the online help messages that are given in response to the @H command (Chapters 4 and 5).

The system error message file contains the online help messages pertaining to error messages issued by the Stub Generator. These messages are given in response to the @M command (see Chapters 4 and 5).

The control card file contains the job stream that is generated and submitted by the Stub Generator. This file may be saved as a permanent disk file. If the user asks the Stub Generator to save the control card file, the permanent file name will be generated by concatenating the user-supplied prefix with CNTL (for three-character prefixes), OCNTL (for two-character prefixes), and LOCNTL (for one-

character prefixes). The batch job that is submitted will generate printed output and a sorted UPDATE source program library (OLDPL). The OLDPL contains all of the source code for the software system in a special UPDATE format.

The ADS master and invert files will also be created if the RUN ADS option is in effect. These files contain information about the code in the source program library needed to generate ADS reports.

Options

The Stub Generator optionally inserts initial debugging features when the skeleton modules (Figure 2) are created. These features "trace" program paths and parameter passing.

The initial debugging features are controlled by the logical variables TRACE and PARAMS. These variables (along with the variable DEBUG*) are assumed to be in the standard debug common block of the software system being developed. The TRACE and PARAMS flags can be turned on or off internally during debugging. Besides TRACE, PARAMS, and DEBUG, the Stub Generator assumes that the software system has a routine called TRACER.

Using these variables and the routine TRACER, the Stub Generator automatically creates the code to trace program paths and display passed parameters at routine entries and exits.

Help

Two system message files explain error messages and tell the user what the causes of the error might be. The online help features are invoked by the @H and @M commands. (For details, see Chapters 4 and 5.)

```
SUBROUTINE SUB1(X,Y)
```

```
IF (TRACE) CALL TRACER (4HSUB1,1)
```

```
IF (PARAMS) WRITE (DEBUG,*) X,Y
```

```
IF (PARAMS) WRITE (DEBUG,*) X,Y
```

```
IF (TRACE) CALL TRACER (4HSUB1,2)
```

```
RETURN
```

```
END
```

—generated other statements including the DEBUG common

—entry trace

—entry parameters

—other code

—exit parameters

—exit trace

Figure 2. Sample Stub Generator debugging statements.

3 CAPABILITIES AND LIMITATIONS

The Stub Generator automatically generates all the UPDATE directives needed to (1) create a source program library, (2) add new modules, or (3) purge existing modules. As module names are mentioned (either as the module is defined or in a context which identifies the module type), it builds a history of names to warn users of possible conflicts (e.g., using duplicate module names, using the same name in a context that implies different routine types, or creating a subroutine call to a defined subroutine whose argument list has a different number of arguments than the defined subroutine).

For example, if a new module is given the same name as a module already in the system, the Stub Generator will inform the user that the named module exists and ask if he/she wants to purge the existing version. The history may also detect usage conflicts, such as naming a module previously defined as a function in a generated subroutine call, and a few possible error conditions, such as a conflict between the number of formal parameters given in the module definition and the number of parameters given in the argument list of a generated subroutine call to that previously defined module.

Flexible Features

1. The Stub Generator accepts card image input for ADS comments. This lets the user exercise all the formatting capabilities in ADS.

2. Generated subroutine calls and user-defined code (e.g., comments, calculations, and function calls) can be put in any order in the body of the routine. The Stub Generator asks for subroutine calls and other code and requires consecutive negative responses to both requests before completing the module. The Stub Generator will automatically include "RETURN" or "STOP" and "END" statements as needed.

3. The UPDATE source library (OLDPL) is output in a standard order of COMDECKS, main programs, and subroutines and functions. To ask for a nonstandard ordering, the user assigns a level from 1 to 3 to each module when it is defined. The OLDPL will then be created in the order:

- a. COMDECKS

- b. Level 3 routines

- c. Level 2 routines

- d. Level 1 routines.

Each subgroup of either OLDPL ordering is kept in alphabetical order. Module levels are stored in the Stub Generator history file and their order maintained as modules are added or deleted.

4. Permanent files can be accessed under an alternate user number. Thus, the OLDPL and the basic system history file can be generated and maintained under a master account, and be accessed by several different subsidiary accounts.

5. A full update of the ADS master files is always done when an ADS run is requested. This ensures that all possible changes are included. (The Stub Generator creates an ADS batch job to do this; see pp 6, 7.) But if the user is working with a pre-existing OLDPL (or if there are a number of completed modules), it may be better to run ADS jobs more tailored to the specific update needs outside of the Stub Generator.

Limitations

1. The Stub Generator has no file editing capabilities other than a very limited line replacement for the job control stream and backspacing one input line. Any big change should be made by a separate UPDATE run or by redefining the defective module.

A module is redefined by entering its name in response to the request for the next routine name and by allowing the STUB Generator to purge the old routine. (Modules cannot be purged or redefined on the first run, since the purge directive is not valid when a new OLDPL is being created.) The Stub Generator does not interactively update the OLDPL; rather, it generates input files of directives and creates a batch job from a user session to accomplish the actual OLDPL modification.

2. Because the Stub Generator is not a FORTRAN compiler, a clean compile is not guaranteed by generating a module. For example, if the software system being developed is to automatically include TRACER calls, the user must define "TRACE" as a logical variable in a standard debug COMDECK which must be included in each of the system's routines. If this is not done, a FORTRAN compile error will occur. To include automatic unformatted parameter writes, "PARAMS" must be defined in the same way

*The output unit for the software system's debugging output.

as TRACE. TRACE, PARAMS, and the standard debug COMDECK refer to the special initial debugging outlined in the overview. The debug COMDECK name is defined during the first Stub Generator session and maintained in the Stub Generator history file (see Chapter 2).

Figure 2 is a sample of the debugging statements created by the Stub Generator.

Output

Batch job output lists (1) all active cards in the OLDPL, (2) any FORTRAN compilation errors, and (3) any ADS reports requested. If an ADS run was not requested, only the OLDPL list will be output. A sample batch job submitted by the Stub Generator is given in Appendix D.

4 COMMAND LANGUAGE AND ONLINE HELP

Nine commands ask for special action from the Stub Generator. The command syntax is a "@" immediately followed by a one-letter command (or the full command word). Table 1 explains each command.

The online help commands (@H and @M) tell the user what type of information the Stub Generator is asking for or what type of errors might account for a particular error message.

The @H command gives clues about the type of information the Stub Generator is currently request-

Table 1
Stub Generator Commands

Abbreviation	Full Command	Explanation
@A	@ABORT	Aborts the Stub Generator program without saving any of the current work files.
@B	@BACKSPACE	Backspaces one record on the last file written. This command must be used carefully. It may not be possible to backspace to the correct place since the Stub Generator maintains several files simultaneously. If a different prompt has intervened since the record to be replaced was displayed, (e.g., I>, M>, I>), the backspace command may not work properly. However, errors can be fixed by doing a standard UPDATE run.
@C	@COMMANDS	Lists the available commands; this list is displayed on the terminal.
@D	@DONE	Begins normal end-of-job processing to generate and submit a batch job to update the OLDPL (and run ADS, if requested).
@E	@ECHO	Toggles echoing of lines input.
@H	@HELP	Asks for online help giving more information on the question currently being asked of the user.
@M	@MESSAGE	Asks for online help giving more information on the last error message given to the user.
@M,####	@MESSAGE,####	Asks for online help for a particular error message identified by a four-digit code (####).
@P	@PRINT	Prints the last line input by the user.
@S	@SAVE	Saves the user's work files and run options without beginning a batch run. The user can then input more definitions later.

ing. It also tells the user what publications to refer to for more detailed explanations or syntactical rules.

The @M command has two forms; both give extended error messages. (Extended error messages give a detailed statement of the error, possible causes, and names of appropriate reference manuals.) Entering "@M" will result in an extended error message for the last displayed error message. Extended error messages may be retrieved by typing "@M.dddd," where "dddd" is the four-digit code preceding the brief message for which the user is requesting additional information

Up to three levels of help may be available for any particular question or error message. A "+" at the end of a help section tells the user that additional help is available.

5 GENERAL HINTS

1. A null line (RETURN) is the same as a negative response for Y/N questions, except during the first run of the Stub Generator, when basic system parameters are defined. During this first run, the default for all parameters is Y.

2. The Stub Generator will output an UPDATE input file for an existing OLDPL when the user exits a session with the @S command. The user should then edit the UPDATE input files to make sure the *AF directives indicate the correct order for the OLDPL. The permanent file names of the update input files are created by concatenating the user-supplied prefix with DTEI and DTES (for a three-character prefix), PDTEI and PDTES (for a two-character prefix), or UPDTEI and UPDTES (for a one-character prefix). For example, if the software prefix is "SAM," the UPDATE input files are SAMDTEI and SAMDTES. The user can then create a new job stream to begin the UPDATE run.

3. Abbreviations can be used to name both required and optional ADS sections. Valid abbreviations are usually made from the first three characters of the section name (Table 2).

4. When the user ends a list format with a comma, it tells the Stub Generator that the user wants to add more data to the list. For example, if a list

of COMDECKS called in a routine is entered as "COM1, COM2, COM3," the Stub Generator will ask the user for another input line. The user then can add more COMDECK names.

5. If the batch job submitted by the Stub Generator fails for any reason, the user should try to resubmit a corrected run before initiating a new Stub Generator session. The exception is when an abort has occurred after an OLDPL manipulation. In this

Table 2
ADS Categories and Valid Abbreviations

ADS Documentation Categories

Category	Abbreviation
Title	TIT
Common block title	COM TIT
Author	AUT
Date Written	DAT
References	REF
Location	LOC
Method	MET
Control cards	CON
Remarks	REM
System	SYS
Flow	FLO
Files	FIL
Algorithm	ALG
System dependencies	SYS DEP
Nonsystem externals	NON SYS EXT
Machine dependencies	MAC DEP
Implementation dependencies	IMP DEP
Variable dictionary	VAR
Revised (date)	REV
Purpose	PUR

case, the user should purge the work files. Any missing output can then be retrieved by beginning the appropriate job *outside* of the Stub Generator. If the output is not retrieved in this way, it will be produced during the next successful Stub Generator run.

6. Three types of prompts are produced by the Stub Generator and give a clue to the type of input expected:

- I> This is the standard prompt indicating that a reply to the current question is desired. The type of input desired is generally indicated in the question itself or in the online help (@H).
- M> This prompt indicates the Stub Generator is ready to accept multiple input lines. The lines will be accepted as 80-column card image lines. No processing of the line will be done other than to write it to the appropriate file.
- CD This prompt indicates the Stub Generator is ready to accept a card image input of an ADS comment. The CD is used as columns 1 and 2 of the line written and need not be repeated by the user.

7. The Stub Generator gives two classes of error messages: warning and error. Warning messages generally tell the user the Stub Generator has detected a special condition and has taken corrective action, is expecting special input (i.e., continuing input following a line ending with a comma), or has detected a condition that may result in an error (i.e., an incorrect number of arguments on a subroutine call to an already defined subroutine). Fatal error messages indicate a severe error that must be corrected and re-input before processing can continue.

6 SUMMARY

The Stub Generator gives the software development manager who relies on ADS a way of standardizing documentation during all phases of project development. By setting specific, unchangeable ADS documentation standards during the first Stub Generator session, the manager guarantees that minimum documentation goals will be met, no matter how often the software is amended or extended. The Stub Generator also lets the manager monitor project progress easily and reliably. The programmer is also given a convenient way of creating program systems in a top-down structural manner through the generation of standard program stubs.

APPENDIX A: SAMPLE FIRST SESSION

The parameters which define the basic characteristics of the software being developed are set during the first Stub Generator session. This first session is usually done by the software development team leader. The team leader can control the system characteristics and the basic ADS documentation by setting parameters which cannot be changed. In the sample session in this appendix, the system parameter library was saved by using the @S command. The team leader could also choose to define the standard debug common block and the TRACER routine to be used by the system and begin the batch run to create an OLDPL by ending the session with the @D command.

The prompt to the user is "I>". If nothing is shown on that line, the user used only the RETURN key. This key defaults to "YES" or the value indicated in Figure A1.

Parameter	Default	Change [†]		Description
		Yes	No	
Master Account	None		X	User id under which system parameters were defined and where OLDPL and system library are maintained by system.
System Prefix	None		X	3 user-specified characters that are appended to the front of standard file names to create names uniquely identifying files belonging to a particular system.
Password	Null		X	Password associated with system-maintained permanent files.
Update Control Character	*		X	Character appended to the front of UPDATE directives.
ADS Required Sections	None		X	User-specified list of ADS sections that are to be generated for each module defined. Specified headers will always be generated even if no input is received for that section.
Account Password	Null	X		Password associated with user id under which user is currently running.
List Control	Yes	X		Controls generation of "**IF -DEF, (mod name), I", "C/ LIST, NONE" and "**IF DEF, LISTALL", "C/ LIST, ALL", statements to allow control of source listing in compiler output.
Implicit Integer	Yes	X		Controls generation of "IMPLICIT INTEGER (A-Z)" statement.
Standard Debug Deck	Yes	X		User may request the generation of "**CALL" to a user-specified debug comdeck in all DECKS defined.
Debug Deck Name	DEBUGR	X		Name of user-defined debug comdeck to be specified on "**CALL".
Standard OLDPL Order	Yes	X		Standard order = comdecks, main routines, subroutines/functions in alphabetical order within each group. User may request control of ordering of all modules except comdecks.
Nonpropagating Comdecks	Yes	X		Controls generation of "NOPROP" parameter on "**COMDECK" directive.

[†] Parameters that cannot be changed after the first session are "system parameters." Changeable parameters are "run-time options."

Figure A1. Stub Generator - first session sample.

Parameter	Default	Change [†]		Description
		Yes	No	
TRACER Calls	Yes	X		Controls generation of calls to a user-defined TRACER routine upon entry and exit to subroutines/functions. Provides a subroutine trace.
Unformatted Parameter Trace	Yes	X		Controls generation of unformatted write statement with I/O list of formal parameters upon entry and exit to subroutines.
Comdeck Define Name	COMM		X	Name to be used in generation of "**IF DEF," directive preceding a block of ADS code in comdecks. Allows suppression of listing of comments.
Routine Define Name	ROUT		X	Name to be used in generation of "**IF DEF," directive preceding a block of ADS code in decks. Allows suppression of listing of comments.
ADS Run Initiation	Yes	X		Controls initiation of execution of the ADS program to generate/update master documentation files and/or produce ADS reports. ADS runs will always redocument all modules defined in the old OLDPL. If the generator is being used to generate stubs for an existing system or only reports are desired, it would be more efficient for the user to initiate an ADS run separately.

[†] Parameters that cannot be changed after the first session are "system parameters." Changeable parameters are "run-time options."

Figure A1. (Cont'd).

welcome to the bcs network
your access port is cix 67

select desired service: eks1

80/07/09. 07.51.30.
EKS1 760C.W0460.63AA 80/06/29.DS-0 18.55.11. 80/07/08.

TERMINAL 27, TTY
RECOVER/USER ID: jeanb
*** NEW SCHEDULE FOR NOS 1.4 IMPLEMENTATION. SEE MSG,NOS ***
C>get,profil/un=cer005
C>call,,jst(master=cer084,sys=sam,pw=test)
07.55.04. WELCOME TO JEAN'S INTERACTIVE STUBMAKER!
07.55.04. HANG ON WHILE I CRANK IT UP.

A D S S T U B G E N E R A T O R

INSTRUCTIONS? (Y/N)

I>y

THE ADS STUB GENERATOR SYSTEM IS DESIGNED AS AN AID TO A SYSTEMS DESIGN TEAM IN THE INITIAL STAGES OF DEVELOPING A FORTRAN SYSTEM. SYSTEM PARAMETERS ARE REQUESTED FROM THE USER TO DETERMINE OPTIONS DESIRED, AND FOR USE WITH PERMANENT FILE FUNCTIONS. THESE OPTIONS ARE RETAINED BETWEEN RUNS. SOME OPTIONS MAY BE CHANGED BY THE USER AT THE BEGINNING OF A RUN.

MOST REQUESTS ARE SELF-EXPLANATORY. IF ADDITIONAL HELP IS REQUIRED, THE USER MAY REFER TO THE USER'S GUIDE FOR MORE DETAILED INFORMATION, OR REQUEST ON-LINE HELP FOR THE SPECIFIC QUESTION OR ERROR MESSAGE. UP TO THREE LEVELS OF ON-LINE HELP ARE AVAILABLE. A "+" FOLLOWING A SECTION OF TEXT INDICATES MORE HELP FOR THAT ITEM IS AVAILABLE AND MAY BE RETRIEVED BY REPEATING THE SAME COMMAND (I.E., @H FOR ADDITIONAL HELP ON THE CURRENT QUESTION, OR @M FOR ADDITIONAL HELP ON AN ERROR MESSAGE).

AVAILABLE COMMANDS:

@A	@ABORT	ABORT PROCESSING
@B	@BACK	BACKSPACE ONE RECORD ON LAST FILE WRITTEN
@C	@COMMANDS	PRINT LIST OF AVAILABLE COMMANDS
@D	@DONE	NORMAL END OF JOB PROCESSING
@E	@ECHO	TOGGLES ECHOING OF INPUT LINES
@H	@HELP	REQUEST ON-LINE HELP FOR CURRENT QUESTION
@M	@MESSAGE	REQUEST ON-LINE HELP FOR LAST ERROR MESSAGE ISSUED
@N,####	@MESSAGE,####	REQUEST OF ON-LINE HELP FOR SPECIFIED ERROR MESSAGE
@P	@PRINT	PRINT LAST LINE INPUT
@S	@SAVE	SAVE CURRENT FILES AND EXIT PROGRAM

**WARNING: 4700 INVALID TID

ABORT SESSION? (Y/N)
 I>n
 5 UPDATE CONTROL CHAR?
 I>
 6 ADS REQ SECTIONS? (NAME1,NAME2,...,NAMEN/NONE)
 I>
 7 ACCT PW?
 I>
 8 AUTO LIST CONTROL STMT? (Y/N)
 I>
 9 AUTO IMPLICIT INTEGER? (Y/N)
 I>
 10 AUTO STD DEBUG? (Y/N)
 I>
 11 DEBUG DECKNAME?
 I>
 12 STD OLDPL ORDER? (Y/N)
 I>
 13 AUTO NOPROP CONDECKS? (Y/N)
 I>
 14 AUTO TRACER CALLS? (Y/N)
 I>
 15 AUTO PARAMETER TRACE? (Y/N)
 I>
 16 ADS CONDECK DEF NAME?
 I>
 17 ADS ROUTINE DEF NAME?
 I>
 18 RUN ADS? (Y/N)
 I>
 LIST SYSTEM PARAMS? (Y/N)
 I>y

SYSTEM PARAMETERS:

2 MASTER ACCT ID?	: CER084
3 SYS? (3 CHARS MAX)	: SAN
4 PW?	: TEST
5 UPDATE CONTROL CHAR?	: *
6 ADS REQ SECTIONS?	
16 ADS CONDECK DEF NAME?	: COMM
17 ADS ROUTINE DEF NAME?	: ROUT

LIST RUN OPTIONS? (Y/N)
 I>y

RUN OPTIONS:

8 AUTO LIST CONTROL STMT? (Y/N)	: Y
9 AUTO IMPLICIT INTEGER? (Y/N)	: Y
10 AUTO STD DEBUG? (Y/N)	: Y
11 DEBUG DECKNAME?	: DEBUGR

```

12 STD OLDPL ORDER? (Y/N)      : Y
13 AUTO NOPROP COMDECKS? (Y/N) : Y
14 AUTO TRACER CALLS? (Y/N)    : Y
15 AUTO PARAMETER TRACE? (Y/N) : Y
18 RUN ADS? (Y/N)               : Y
ANY CHANGES? (Y/N)
I>y
ENTER ALL, # OR <CR>
I>6
  6 ADS REQ SECTIONS? (NAME1,NAME2,...,NAMEN/NONE)
I>tit,com tit,aut,dat
ENTER ALL, # OR <CR>
I>
LIST SYSTEM PARAMS? (Y/N)
I>y

```

SYSTEM PARAMETERS:

```

2 MASTER ACCT ID?              : CER084
3 SYS? (3 CHARS MAX)           : SAM
4 PW?                           : TEST
5 UPDATE CONTROL CHAR?         : *
6 ADS REQ SECTIONS?
  TITLEZ=
  COMMON BLOCK TITLEZ=
  AUTHORIZ=
  DATE WRITTENZ=
16 ADS COMDECK DEF NAME?       : COMM
17 ADS ROUTINE DEF NAME?       : ROUT
LIST RUN OPTIONS? (Y/N)
I>n
ANY CHANGES? (Y/N)
I>n
MODULE NAME?
I>@s

```

```

OK. JUST A SECOND. I'M A SLOW WRITER.
08.00.42. COME AGAIN!
EXIT.
C>

```

APPENDIX B: SAMPLE RUN

The sample run in this appendix has examples of basic features like help commands and how to input ADS comments. A sample version of a TRACER routine and a debugging common block have been entered in this session and a batch run was begun to create an OLDPL. The job stream generated for the batch job is shown on p 24. This version uses KCL procedures, which restrict the range of valid UPDATE characters that can be used as the UPDATE control character. Only the characters A-Z, 0-9, and * can be passed as parameters to the procedures in the standard profile. The Stub Generator will only generate input files for UPDATE runs using other valid UPDATE control characters if the user creates a new job stream outside the Stub Generator.

Prompts to the user are "I>", "M>", and "CD". If nothing is shown on the prompt line, the user has entered only the RETURN.

ADS STUB GENERATOR

INSTRUCTIONS? (Y/N)

I>n

7 ACCT PW?

I>blastu

LIST SYSTEM PARAMS? (Y/N)

I>y

SYSTEM PARAMETERS:

2 MASTER ACCT ID? : CER084

3 SYS? (3 CHARS MAX) : SAM

4 PW? : TEST

5 UPDATE CONTROL CHAR? : *

6 ADS REQ SECTIONS?

TITLEZ=

COMMON BLOCK TITLEZ=

AUTHORZ=

DATE WRITTENZ=

16 ADS CONDECK DEF NAME? : COMM

17 ADS ROUTINE DEF NAME? : ROUT

LIST RUN OPTIONS? (Y/N)

I>y

RUN OPTIONS:

8 AUTO LIST CONTROL STMT? (Y/N) : Y

9 AUTO IMPLICIT INTEGER? (Y/N) : Y

10 AUTO STD DEBUG? (Y/N) : Y

11 DEBUG DECKNAME? : DEBUGR

12 STD OLDPL ORDER? (Y/N) : Y

13 AUTO NOPROP CONDECKS? (Y/N) : Y

14 AUTO TRACER CALLS? (Y/N) : Y

15 AUTO PARAMETER TRACE? (Y/N) : Y

18 RUN ADS? (Y/N) : Y

ANY CHANGES? (Y/N)

I>y

ENTER ALL, # OR <CR>

I>6

***ERROR: 2200 OUT OF BOUNDS

ENTER ALL, # OR <CR>

I>@m

NUMERIC VALUE OF USER INPUT EXCEEDS MINIMUM OR MAXIMUM
ALLOWABLE VALUE.+

I>@m

EXAMPLE;USER INPUT=9 FOR OLDPL LEVEL, VALID VALUES ARE 1-3.

OR USER SPECIFIED A QUERY NUMBER TO CHANGE AN OPTION THAT
IS EITHER NOT 1-18 OR IS NOT CHANGEABLE ON THIS RUN
(SYSTEM PARAMETER).

I>
LIST SYSTEM PARAMS? (Y/N)

I>n
LIST RUN OPTIONS? (Y/N)

I>n
ANY CHANGES? (Y/N)

I>n
MODULE NAME?

I>debugr
MODULE TYPE? (C,P,F,S)

I>eh
C - COMDECK
P - PROGRAM
F - FUNCTION
S - SUBROUTINE

I>c
COMMONS DEFINITION? (Y/N)

I>y
M> common /debugr/ debug,trace,params

M>
TYPE STATEMENTS? (Y/N)

I>y
M> integer debug

M>eb
M> integer debug
M> logical trace,params

M>c
M>
DIMENSION STATEMENTS? (Y/N)

I>n
ADS - REQUIRED SECTIONS

CD COMMON BLOCK TITLEZ=
CD debugr - debugging variable common
CD

OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)
I>var

CD VARIABLE DICTIONARYZ=
CD debugZ debug file name
CDcd traceZ logical variable for routine tracing calls to timer
CD paramsZ logical variable for parameter tracing
CD

OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)
I>none

COMDECK CALLS? (NAME1,NAME2,...,NAMEN/NONE)
I>none

OTHER CODE? (Y/N)
I>n

MODULE NAME?
I>tracer
MODULE TYPE? (C,P,F,S)

I>s
PARAMETER LIST? (P1,P2,...PN/NONE)

I>sname,flag
TYPE STATEMENTS? (Y/N)

I>n
DIMENSION STATEMENTS? (Y/N)

I>n
ADS - REQUIRED SECTIONS

CD TITLEZ=
CD tracer - standard subroutine trace routine
CD

CD AUTHORZ=
CD your average programmerHer
CDEp
 YOUR AVERAGE PROGRAMMRHER

CDob
CD your average programmer
CD

CD DATE WRITTENZ=
CD 11oct79
CD

OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)

I>pur,var

CD PURPOSEZ=
CD tracer prints out tracing information (at entry and exit to
CD subroutines) depending on value of flag
CD

CD VARIABLE DICTIONARYZ=
CD sname - input parameter (char format), contains the subroutine
CD name to be traced
CD flag - indicator to tracer of what action is to be done;
CD\$-- 1; enter subroutine sname
CD\$-- 2; exit subroutine sname
CD

OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)

I>none

COMDECK CALLS? (NAME1,NAME2,...,NAMEN/NONE)

I>none

SUBROUTINE CALLS? (Y/N)

I>n

OTHER CODE? (Y/N)

I>y

M>c igo is caluclated flag
M> if(flag .lt. 1.or. falg .gt. 2) goto 999
M> goto(901,902), flag


```

M> 901 continue
M>c   enter subroutine
M>     tsptr = tsptr + 1
M>     write(debug,702) tsptr,7hentered,sname
M>     goto 1000
M> 902 continue
M>c   exit subroutine
M>     if (tsptr .le. 0) tsptr = 1
M>     write(debug,902) tsptr,7hexited,sname
M> 702 format(1h ,=(1h-),a7,1x,a10)
M>     tsptr = tsptr - 1
M> 1000 continue
M>

```

SUBROUTINE CALLS? (Y/N)

I>n

OTHER CODE? (Y/N)

I>n

MODULE NAME?

I>main

MODULE TYPE? (C,P,F,S)

I>p

FILE DEFINITIONS? (F1,F2,...FN/NONE)

I>input,output,debug=output

TYPE STATEMENTS? (Y/N)

I>n

DIMENSION STATEMENTS? (Y/N)

I>n

ADS - REQUIRED SECTIONS

CD TITLEX=

CD main - a sample main program

CD

CD AUTHORIZ=

CD y.a. programmer

CD

CD DATE WRITTENZ=

CD 1200 a.d.

CD

OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)

I>none

CONDECK CALLS? (NAME1,NAME2,...,NAMEN/NONE)

I>filecon

SUBROUTINE CALLS? (Y/N)

I>y

NAME?

I>sub1

PARAMETERS? (P1,P2,...PN/NONE)

I>paran1,paran2,paran3,paran4,paran5,paran6,paran7,paran8,paran9,paran10

SUBROUTINE CALLS? (Y/N)

```

I>n
OTHER CODE? (Y/N)
I>y
M>c      comments and other code can be interspersed with generated
M>c      subroutine calls
M>
SUBROUTINE CALLS? (Y/N)
I>y
NAME?
I>sub2
PARAMETERS? (P1,P2,...PN/NONE)
I>none
SUBROUTINE CALLS? (Y/N)
I>n
OTHER CODE? (Y/N)
I>n
MODULE NAME?
I>sub2
MODULE TYPE? (C,P,F,S)
I>s
PARAMETER LIST? (P1,P2,...PN/NONE)
I>none
TYPE STATEMENTS? (Y/N)
I>n
DIMENSION STATEMENTS? (Y/N)
I>n
ADS - REQUIRED SECTIONS

```

```

CD      TITLEZ=
CD      sub2 - a sample subroutine
CD

```

```

CD      AUTHORIZ=
CD      y.a.programmer
CD

```

```

CD      DATE WRITTENZ=
CD      any date
CD

```

```

OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)

```

```

I>none

```

```

CONDECK CALLS? (NAME1,NAME2,...,NAMEN/NONE)

```

```

I>none

```

```

SUBROUTINE CALLS? (Y/N)

```

```

I>n

```

```

OTHER CODE? (Y/N)

```

```

I>n

```

```

MODULE NAME?

```

```

I>@d

```

```

**FILECOM**

```

```

THE ABOVE CONDECKS MUST BE DEFINED PRIOR TO INITIATING A RUN.
ENTER @S TO END SESSION AND SAVE FILES, OR CONTINUE DEFINITIONS.

```

```

I>filecom

```

```

MODULE TYPE? (C,P,F,S)

```

I>c
COMMONS DEFINITION? (Y/N)
I>y
M> common /filecom/ input,output
M>
TYPE STATEMENTS? (Y/N)
I>n
DIMENSION STATEMENTS? (Y/N)
I>n
ADS - REQUIRED SECTIONS

CD COMMON BLOCK TITLEZ=
CD filecom - file definition common
CD
OPTIONAL SECTIONS? (SEC1,SEC2,...SECN/NONE)
I>none
CONDECK CALLS? (NAME1,NAME2,...,NAMEN/NONE)
I>none
OTHER CODE? (Y/N)
I>n
MODULE NAME?
I>qd

OK. JUST A SECOND WHILE I GET IT ALL TOGETHER.
ADS COMMANDS? (Y/N)
I>y
M>title a sample ads run generated by the stub generator

**WARNING: 4400 TERMINATING ; SUPPLIED
M>ep
TITLE A SAMPLE ADS RUN GENERATED BY THE STUB GENERATOR

M>print wide dump for all commons;
M>print wide dump for all routines;
M>draw wide tree;
M>end;
M>
1 /JOB
2 STUBGEN,T30,CN130000,P02.
3 USER,CERO84,BLASTU. JEAND.
4 GET,PROFIL/UN=CERO05.
5 GET,OLBPL=SAHOPL/UN=0,PU=TEST,NA.
6 CALL,,UPDTE(OPL=SAHOPL,UN=0,PU=TEST,CT=P,UC=*)
7 CALL,,OSRT(OPL=SAHOPL,UN=0,PU=TEST,CT=P,UC=*)
8 CALL,,STADS(OPL=SAHOPL,UN=0,PU=TEST,CT=P,UC=*,INVT=SAMINV,MAST=SANMAS)
9 PURGE,SANDTES.
10 PURGE,SANDTE1.
11 PURGE,SANSCND/NA.
12 PURGE,SANTCHD/NA.
13 EXIT,U.
14 COPYDF,UPLIST,OUTPUT.
15 IF(FILE(ADSOUT,LO))COPYDF,ADSOUT,OUTPUT,3.
16 EXIT.
17 /EOR

ANY CHANGES? (Y/N)

I>n

DO YOU WISH TO RUN THIS JOB? (Y/N)

I>n

DO YOU WISH TO SAVE THIS DECK? (Y/N)

I>y

08.31.35. COME AGAIN!

EXIT.

C>get,samcntl

C>list,f=samcntl

/JOB

STUBGEN,T30,CN130000,P02.

USER,CER084,BLASTU. JEAND.

GET,PROFIL/UN=CER005.

GET,OLDPL=SAHOPL/UN=0,PW=TEST,NA.

CALL,,UPDTE(OPL=SAHOPL,UN=0,PW=TEST,CT=S,UC=*)

CALL,,DSRT(OPL=SAHOPL,UN=0,PW=TEST,CT=S,UC=*)

CALL,,STABS(OPL=SAHOPL,UN=0,PW=TEST,CT=S,UC=*,INVT=SAHINV,MAST=SAHNAS)

PURGE,SANDTES.

PURGE,SANDTE1.

PURGE,SAMSCND/NA.

PURGE,SAMTCND/NA.

EXIT,U.

COPYBF,UPLIST,OUTPUT.

IF(FILE(ADSOUT,LO))COPYBF,ADSOUT,OUTPUT,3.

EXIT.

/EOR

/PACK

/READ,SANDTES

/READ,SANDTE1

/EOR

*ID SORT

/EOR

YANK\$\$\$

/READ,SAMTCND

/EOR

YANK\$\$\$

/EOF

*ID ADS

*DF COMM

*DF ROUT

*DF MAIN

*DF SUB2

*DF TRACER

/EOR

/READ,SAMSCND

/EOF

EOI ENCOUNTERED.

C>subnit,samcntl,e1=cer001

07.36.15. 80/07/14.LCIIDXZ

C>bye

JOB PROCESSING CCUS 110.781

BYE 80/07/14. 07.36.30.

APPENDIX C: INSTRUCTIONS FOR BEGINNING A STUB GENERATOR SESSION ON BCS

To begin a Stub Generator session on the Cyber 175 at BCS, the user should login and retrieve the procedure file:

`GET,PROFIL/UN=CER005`

The user then calls the interactive procedure:

`CALL,,JIST(MASTER=<master account>,SYS=<system prefix>,
PW=<password>,TID=<RJE user id>)`

The parameter list for the call is:

- <master account>: The ID under which the first session is/was executed to define the system parameters. Required so that the Stub Generator can access and maintain required permanent files.
- <system prefix>: User-defined system identifier (maximum three characters).
- <password>: Password to be associated with system-maintained permanent files. Needed to allow the Stub Generator to access and maintain the proper set of system files.
- <RJE user id>: User ID to which output from the batch job is to be sent. This does not have to be specified if the user will not be beginning a batch job at the end of the session (@D command) — the resulting warning can be ignored. If the user fails to enter a TID but does wish to begin a run, the job should be aborted when requested and the procedure restarted with the proper parameters. The Stub Generator submits the batch job from the procedure; thus, an incorrect TID cannot be fixed from within the program.

APPENDIX D: BATCH JOB OUTPUT

This appendix lists output generated by the batch job begun by the Stub Generator session shown in Appendix B. The OLDPL listing shows the contents of the final sorted OLDPL. Immediately after the OLDPL listing is a short list of compile errors. These errors would not be listed if the batch job had not included an ADS run. The ADS output includes an echo of the commands received, a list of requested reports, and the requested tree of the system as it currently exists.

DIRECTIVES -III OF DE FROM -ANNUVE <DECK DATE>,YAHKSSS
 DIRECTIVES -III OF DE FROM -ANNUVE <DECK DATE>,YAHKSSS
 DIRECTIVES -III OF DE FROM -ANNUVE <DECK DATE>,YAHKSSS

UNLABELED OLDPI CONNECTION IDENTIFIERS UPDATE 1.3-498. 60/07/14. 07.36.34. PAGE 1

CORRECTION IDENTIS ARE LISTED IN CHRONOLOGICAL ORDER OF INSERTION

YAHKSSS DEBUGR FILECOM TRACER MAIN SUB2

DECKS ARE LISTED IN THE ORDER OF THEIR OCCURRENCE ON A NEW PROGRAM LIBRARY IF ONE IS CREATED BY THIS UPDATE

YAHKSSS DEBUGR FILECOM MAIN SUB2 TRACER

CORRECTION DECKS ENCOUNTERED

DEBUGR FILECOM

DECKS -OTTEN TO COMPILE FILE

MAIN SUB2 TRACER

THIS UPDATE REQUIRED 347008 WORDS OF CORE.

80/07/14, 07.16.34,

UPDATE 1.3-898,

MASTER AUDIT, IDENT CARD TOTAL

UNLARKED ULOPL

LIST OF CONTROL, ACTIVE, AND/OR INACTIVE CARDS IN DEHUGR

```

DEHUGR *CUMDECK DEHUGR,ADPROB
DEHUGR CUMDECK /DEHUGR/ DEHUGR,TRACE,PARAMS
DEHUGR INTERGEN DEHUGR
DEHUGR LOGICAL TRACE,PARAMS
C
*IF DEF,CD*
DEHUGR CD CUMDECK /DEHUGR/ DEHUGR,TRACE,PARAMS
DEHUGR CD DEHUGR - DEHUGRING VARIABLE CUMDECK
DEHUGR CD VARIABLE DICTONARY*
DEHUGR CD DEHUGR /DEHUGR/ FILE NAME
DEHUGR CD TRACE: LOGICAL VARIABLE FOR ROUTINE TRACING CALLS TO TIMER
DEHUGR CD PARAMS: LOGICAL VARIABLE FOR PARAMETER TRACING
DEHUGR *ENDIF

```

LIST OF CONTROL, ACTIVE, AND/OR INACTIVE CARDS IN FILECOM

```

FILECOM *CUMDECK FILECOM,ADPROB
FILECOM CUMDECK /FILECOM/ INPUT,OUTPUT
*IF DEF,CD*
FILECOM CD CUMDECK /FILECOM/ INPUT,OUTPUT
FILECOM CD FILECOM - FILE DEFINITION COMMON
FILECOM *ENDIF

```

LIST OF CONTROL, ACTIVE, AND/OR INACTIVE CARDS IN MAIN

```

MAIN *DECK MAIN
MAIN *IF -DEF,MAIN,1
MAIN C/ LIST,MAIN
MAIN *IF DEF,LISTALL,1
MAIN C/ LIST,ALL
MAIN PROGRAM MAIN(OUTPUT,OUTPUT,DEBUG=OUTPUT)
MAIN IMPLICIT INTEGER (A-Z)
*IF DEF,MAIN
CD TITLE*
CD MAIN - A SAMPLE MAIN PROGRAM
CD AUTHORITY
CD Y-4. PROGRAMMER
CD DATE 1200 A.D.
CD 1200 A.D.
*ENDIF
*CALL DEHUGR
*CALL FILECOM
CALL SURT(PARAM1,PARAM2,PARAM3,PARAM4,PARAM5,PARAM6,PARAM7,
PARAM8,PARAM9,PARAM10)
C COMMENTS AND OTHER CODE CAL RE INTERSPERSED WITH GENERATED
C SUBROUTINE CALLS
CALL SUB2
STOP
END

```

LIST OF CONTROL, ACTIVE, AND/OR INACTIVE CARDS IN SUR2

```

SUR2 *DECK SUR2
SUR2 *IF -DEF,SUR2,1

```


UNLABELED OLDFL

MASTER AUDIT, IDENT CARD TOTAL

UPDATE 1.3-498.

PAGE 4

80/07/14, 07.36.34.

LIST OF CONTROL, ACTIVE, AND/OR INACTIVE CARDS IN TRACER

TRACER	GO TO 1000		
TRACER	902 CONTINUE		
TRACER	C		
TRACER	EXIT SUBROUTINE		
TRACER	IF (ISPTR .LE. 0) ISPTR = 1		
TRACER	WRITE(OUTPUT,902) ISPTR,7HEXITED,SNAME		
TRACER	702 FORMAT(1H, '(1H-),A7,1X,A10)		
TRACER	ISPTR = ISPTR - 1		
TRACER	1000 CONTINUE		
TRACER	IF (PARAMS) WRITE(OUTPUT,*) SNAME, FLAG		
TRACER	RETURN		
TRACER	END		
4 ERRORS IN TRACER			
ERRORS IN			
TRACER 29=	GOTO(901,902), FLAG		
TRACER 35=	WRITE(OUTPUT,702) ISPTR,7HEXITED,SNAME		
TRACER 38=	WRITE(OUTPUT,902) ISPTR,7HEXITED,SNAME		
TRACER 44=	END		

I AM IF STATEMENT MAY BE MORE EFFICIENT THAN A 2 OR 3 BRANCH COMPUTED GO TO STATEMENT.

TRACER 29

FE SNAME101 SYMBOLIC NAME HAS TOO MANY CHARACTERS.

FE TRACER 33

FE .902 PRESENT USE OF THIS LABEL CONFLICTS WITH PREVIOUS USES.

TRACER 38

UNDEFINED STATEMENT NUMBERS 999

14 JUL 80

07.36.57.

CEPL - AHS - MESSAGE OUTPUT

PAGE 1

** WARNING: DUPLICATE WRITE REQUESTED FOR COMMON BLOCK - DEBUG - REQUEST IGNORED

TITLE A SAMPLE ADS RUN GENERATED BY THE STUD GENERATOR

PRINT WIDE DUMP FOR ALL COMMONS

PRINT WIDE DUMP FOR ALL ROUTINES

** WARNING: REQUESTED READ OF RECORD (ROUTINE) -- SUBJ , RECORD NOT ON MASTER FILE

DRAW WIDE TREE

END

14 JUL 80 07.36.57. CERL - ADS - VERSION 1.0
A SAMPLE ADS RUN GENERATED BY THE STUB GENERATOR

DUMP

TITLE.
DEBUG - DEBUGGING VARIABLE COMMON
VARIABLE DICTIONARY FOR COMMON BLOCK DEBUG.
DEBUG - DEBUG FILE NAME
PARAMS - LOGICAL VARIABLE FOR PARAMETER TRACING
TRACE - LOGICAL VARIABLE FOR ROUTINE TRACING CALLS TO TIMER
THE ROUTINES WHICH CALL DEBUG ARE --
SUB2 TRACER

MAIN

31

14 JUL 80 07.36.57. CERL - ADS - VERSION 1.0
A SAMPLE ADS RUN GENERATED BY THE STUB GENERATOR

DUMP

TITLE.
FILECOM - FILE DEFINITION COMMON
VARIABLE DICTIONARY FOR COMMON BLOCK FILECOM.
INPUT - ** NAME **
OUTPUT - ** NAME **
THE ROUTINES WHICH CALL FILECOM ARE --
MAIN

CERL - ADS - VERSION 1.0
 A SAMPLE ADS RUN GENERATED BY THE STUB GENERATOR

TITLE.
 MAIN - A SAMPLE MAIN PROGRAM

AUTHOR.
 T.A. PROGRAMMER

DATE WRITTEN.
 1200 A.D.

FILES USED IN MAIN ARE --
 DEBUG INPUT OUTPUT

VARIABLE DICTIONARY FOR ROUTINE MAIN .

- PARAM1 - ** NONE **
- PARAM2 - ** NONE **
- PARAM3 - ** NONE **
- PARAM4 - ** NONE **
- PARAM5 - ** NONE **
- PARAM6 - ** NONE **
- PARAM7 - ** NONE **
- PARAM8 - ** NONE **
- PARAM9 - ** NONE **
- PARAM10 - ** NONE **

THIS A MAIN PROGRAM OF LENGTH ***** WORDS.

ROUTINES CALLED BY MAIN ARE --
 SUB1 SUB2

COMMON BLOCKS CALLED BY MAIN ARE --
 DEBUG FILECOM

THE ROUTINES WHICH CALL MAIN ARE -- ** NONE **

AS OF 14 JUL 60-SUB2

CERL - ADS - VERSION 1.0
A SAMPLE ADS RUN GENERATED BY THE SUB2 GENERATOR

14 JUL 60 07.36.57.

DUMP

TITLE.
SUB2 A SAMPLE SUBROUTINE

AUTHOR.
V.A.PROGRAMMER

DATE WRITTEN.
ANY DATE

VARIABLE DICTIONARY FOR ROUTINE SUB2 -- ** NONE **
THIS SUBROUTINE HAS A LENGTH OF ***** WORDS.

ROUTINES CALLED BY SUB2 ARE --
TRACER

COMMON BLOCKS CALLED BY SUB2 ARE --
DERUGH

THE ROUTINES WHICH CALL SUB2 ARE --
MAIN

AS OF 14 JUL 80-TRACER

CEPL - AUS - VERSION 1.0
A SAMPLE AUS RUN GENERATED BY THE STUB GENERATOR

07.36.57.

DD-P

TITLE - STANDARD SUBROUTINE TRACE ROUTINE

AUTHOR - YOUR AVERAGE PROGRAMMER

DATE - 11/11/79

PURPOSE - TRACER PRINTS OUT TRACING INFORMATION (AT ENTRY AND EXIT TO SUBROUTINES) DEPENDING ON VALUE OF FLAG

VARIABLE DICTIONARY FOR ROUTINE TRACER .

FLAG - ** NONE **

FLAG - INDICATOR TO TRACER OF WHAT ACTION IS TO BE DONE:
-- 1; ENTER SUBROUTINE SNAME
-- 2; EXIT SUBROUTINE SNAME

SNAME - INPUT PARAMETER (CHAR FORMAT), CONTAINS THE SUBROUTINE NAME TO BE TRACED

ISPTR - ** NONE **

THIS SUBROUTINE HAS A LENGTH OF ***** MURUS.

ROUTINES CALLED BY TRACER ARE -- ** NONE **

COMMON BLOCKS CALLED BY TRACER ARE --

DERUGH

THE ROUTINES WHICH CALL TRACER ARE --

SUB2

PAGE 6
AS OF 14 JUL 80--NOT FOUND

14 JUL 80 07.36.57. CERL - ADS - VERSION 1.0
CRUSS-REF A SAMPLE ADS RUN GENERATED BY THE STUB GENERATOR

THE FOLLOWING RECORDS WERE NOT FOUND ON THE MASTER FILE --
SUB1

THE ROUTINES WHICH CALL SUB1 ARE --
MAIN

PAGE 7
AS OF 14 JUL 80--MAIN

14 JUL 80 07.36.57. CERL - ADS - VERSION 1.0
TREE A SAMPLE ADS RUN GENERATED BY THE STUB GENERATOR

MAIN-----
1*SUB1
1*SUB2-----1*TRACEH

CERL DISTRIBUTION

Chief of Engineers
ATTN: Tech Monitor
ATTN: DAEN-ASI-L (2)
ATTN: DAEN-CCP
ATTN: DAEN-CW
ATTN: DAEN-CWE
ATTN: DAEN-CWM-R
ATTN: DAEN-CWO
ATTN: DAEN-MP
ATTN: DAEN-MPC
ATTN: DAEN-MPE
ATTN: DAEN-MPO
ATTN: DAEN-MPR-A
ATTN: DAEN-RD
ATTN: DAEN-RDC
ATTN: DAEN-RDM
ATTN: DAEN-RM
ATTN: DAEN-ZC
ATTN: DAEN-ZCE
ATTN: DAEN-ZCI
ATTN: DAEN-ZCM

US Army Engineer Districts

ATTN: Library

Alaska
Al Batin
Albuquerque
Baltimore
Buffalo
Charleston
Chicago
Detroit
Far East
Fort Worth
Galveston
Huntington
Jacksonville
Japan
Kansas City
Little Rock
Los Angeles
Louisville
Memphis
Mobile
Nashville
New Orleans
New York
Norfolk
Omaha
Philadelphia
Pittsburgh
Portland
Riyadh
Rock Island
Sacramento
San Francisco
Savannah
Seattle
St. Louis
St. Paul
Tulsa
Vicksburg
Walla Walla
Wilmington

US Army Engineer Divisions

ATTN: Library

Europe
Huntsville
Lower Mississippi Valley
Middle East
Middle East (Rear)
Missouri River
New England
North Atlantic
North Central
North Pacific
Ohio River
Pacific Ocean
South Atlantic
South Pacific
Southwestern

Waterways Experiment Station

ATTN: Library

Cold Regions Research Engineering Lab

ATTN: Library

US Government Printing Office

Receiving Section/Depository Copies (2)

Defense Technical Information Center

ATTN: DDA (12)

Engineering Societies Library
New York, NY

FESA, ATTN: Library

ETL, ATTN: Library

Engr. Studies Center, ATTN: Library

Inst. for Water Res., ATTN: Library

Army Instl. and Major Activities (CONUS)

DARGOM - Dir., Inst., & Svcs.

ATTN: Facilities Engineer

ARRADCOM

Aberdeen Proving Ground

Army Matls. and Mechanics Res. Ctr.

Corpus Christi Army Depot

Harry Diamond Laboratories

Dugway Proving Ground

Jefferson Proving Ground

Fort Monmouth

Letterkenny Army Depot

Natick Research and Dev. Ctr.

New Cumberland Army Depot

Pueblo Army Depot

Red River Army Depot

Redstone Arsenal

Rock Island Arsenal

Savanna Army Depot

Sharpe Army Depot

Seneca Army Depot

Tobyhanna Army Depot

Tooele Army Depot

Watervliet Arsenal

Yuma Proving Ground

White Sands Missile Range

FORSCOM

FORSCOM Engineer, ATTN: AFEN-FE

ATTN: Facilities Engineers

Fort Buchanan

Fort Bragg

Fort Campbell

Fort Carson

Fort Davens

Fort Drum

Fort Hood

Fort Indiantown Gap

Fort Irwin

Fort Sam Houston

Fort Lewis

Fort McCoy

Fort McPherson

Fort George G. Meade

Fort Ord

Fort Polk

Fort Richardson

Fort Riley

Presidio of San Francisco

Fort Sheridan

Fort Stewart

Fort Wainwright

Vancouver Bks.

TRADOC

HQ, TRADOC, ATTN: ATEN-FE

ATTN: Facilities Engineer

Fort Belvoir

Fort Benning

Fort Bliss

Carlisle Barracks

Fort Chaffee

Fort Dix

Fort Eustis

Fort Gordon

Fort Hamilton

Fort Benjamin Harrison

Fort Jackson

Fort Knox

Fort Leavenworth

Fort Lee

Fort McClellan

Fort Monroe

Fort Rucker

Fort Sill

Fort Leonard Wood

INSCOM - Ch, Instl. Div.

ATTN: Facilities Engineer

Vint Hill Farms Station

Arlington Hall Station

WESTCOM

ATTN: Facilities Engineer

Fort Shafter

MDW

ATTN: Facilities Engineer

Cameron Station

Fort Lesley J. McNair

Fort Myer

MSC

HQ USAHSC, ATTN: HSLO-F

ATTN: Facilities Engineer

Fitzsimons Army Medical Center

Walter Reed Army Medical Center

USACC

ATTN: Facilities Engineer

Fort Huachuca

Fort Ritchie

MTMC

HQ, ATTN: MTMC-SA

ATTN: Facilities Engineer

Oakland Army Base

Bayonne MOT

Sunny Point MOT

US Military Academy

ATTN: Facilities Engineer

ATTN: Dept of Geography &

Computer Science

USAES, Fort Belvoir, VA

ATTN: ATZA-DTE-EM

ATTN: ATZA-DTE-SU

ATTN: Engr. Library

Chief Inst. Div., I&SA, Rock Island.

USA ARRCOM, ATTN: Dir., Instl & Svc

TARCOM, Fac. Div.

TECOM, ATTN: DRSTE-LG-F

TSARCOM, ATTN: STSAS-F

NARAD COM, ATTN: ORDNA-F

AMARC, ATTN: DRXMR-WE

HQ, XVIII Airborne Corps and

Ft. Bragg

ATTN: AFZA-FE-EE

HQ, 7th Army Training Command

ATTN: AETTIG-DEH (5)

HQ USAREUR and 7th Army

DDCS/Engineer

ATTN: AEAEN-EN (4)

V Corps

ATTN: AETVDEH (5)

VII Corps

ATTN: AETSDEH (5)

21st Support Command

ATTN: AEREH (5)

US Army Berlin

ATTN: AEBA-EN (2)

US Army Southern European Task Force

ATTN: AESE-ENG (5)

US Army Installation Support Activity

Europe

ATTN: AEUES-RP

8th USA, Korea

ATTN: EAFE

Cdr, Fac Engr Act (8)

AFE, Yongsan Area

AFE, 2D Inf Div

AFE, Area II Spt Det

AFE, Cp Humphreys

AFE, Pusan

AFE, Taegu

DLA ATTN: DLA-WI

USA Japan (USARJ)

Ch, FE Div, AJEN-FE

Fac Engr (Monsu)

Fac Engr (Okawa)

ROK/US Combined Forces Command

ATTN: EUSA-MNC-CFC/Engr

416th Engineer Command

ATTN: Facilities Engineering

Norton AFB

ATTN: AFRCE-MX/DEE

Team Distribution

Chief of Engineers
ATTN: DAEN-DSE
ATTN: DAEN-MPO-B
ATTN: DAEN-MPO-U
ATTN: DAEN-MPZ-A
ATTN: DAEN-MPR
ATTN: DAEN-RDL
Dept of the Army
WASH DC 20314

Chief of Engineers
ATTN: DAEN-PMS
Dept of the Army
WASH DC 20314
for forwarding to:
International Organization
for Standards
Central Secretariat
1, Rue de Varembe, 1211
Geneva, Switzerland

US Army Materiel Development
and Readiness Command
ATTN: DRCDE-DK (3)
5001 Eisenhower Ave
Alexandria, VA 22333

Commander
US Army Electronics Command
ATTN: DRSEL-TL-M/Mr. Tenzer
Fort Monmouth, NJ 07703

DOD ADP Policy Committee
Assistant Secretary of Defense
(Comptroller)
WASH DC 20301

DOD Standardization Area
Computer Aided Design and
Numerical Control
Naval Ship Engineering Center
Hyattsville, MD 20782

DOD Standardization Program
for Information Processing
Standards for Computers (IPSC)
Directorate of Data Automation
(AF/KRAX)
HQ, USAF
WASH DC 20330

Commander
7th Army Combined Arms Training
Center
ATTN: AETTM-HRD-EHD
APO New York 09407

US Army Engr Div, Europe
ATTN: Technical Library (3)
APO New York 09757

American National Standards
Institute
1430 Broadway
New York, NY 10018

ANSI X3 Committee
C/O CBEMA
1828 L Street, NW
WASH DC 20036

DOD Working Group on Computer
Documentation Standards
DOD/DOCN
The Pentagon
WASH DC 20301

DOD Working Group of Computer-
Generated Military Symbology
DOD/DISPLAY
The Pentagon
WASH DC 20301

Federal Information Processing
Standards Coordinating and
Advisory Committee
Dept of Commerce
WASH DC 20234

Library of Congress
Exchange and Gift Division
ATTN: Federal Documents Section
WASH DC 20540

Interagency Committee on Automatic
Data Processing
National Bureau of Standards
WASH DC 20234

National Bureau of Standards
Institute for Computer Sciences
and Technology
WASH DC 20234

Lawrie, Linda

Automated Documentation System (ADS) Stub Generator : description and user instructions / by Linda Lawrie, Jean Baugh. -- Champaign, IL : Construction Engineering Research Laboratory ; Springfield, VA : available from NTIS, 1980.
35 p. (Technical report ; E-167)

I. Electronic data processing documentation. I. Baugh, Jean. II. Title.
III. Series: U.S. Army Construction Engineering Research Laboratory. Technical report ; E-167.

**DATA
FILM**