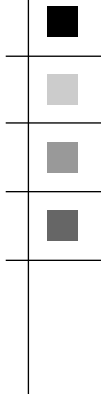




INSTALLATION INSTRUCTIONS

Installation Instructions for the SAS/C[®] 7.50 Cross-Platform Software for Windows Environments





INSTALLATION INSTRUCTIONS

**Installation Instructions for the
SAS/C[®] 7.50
Cross-Platform Software for
Windows Environments**



The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *Installation Instructions for the SAS/C® 7.50 Cross-Platform Software for Windows Environments*, Cary, NC: SAS Institute Inc., 2004.

Installation Instructions for the SAS/C® 7.50 Cross-Platform Software for Windows Environments

Copyright © 2004 SAS Institute Inc., Cary, NC, USA.

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, by any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

Limited permission is granted to store this copyrighted material in your system and display it on terminals, to print only the number of copies required for use by those persons responsible for installing and supporting the licensed SAS programs for which this material has been provided, and to modify the material to meet specific installation requirements. The SAS Institute copyright notice must appear on all printed versions of this material or extracts thereof, and on the display medium when the material is displayed. Permission is not granted to reproduce or distribute the material except as stated above.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software by the government is subject to restrictions as set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

Table of Contents

Getting Started	v
Purpose	v
Conventions	v
Terms	v
Federal Government Rights Notice	v
Chapter 1: Introduction	7
Installation	7
Chapter 2: Disk Space Requirements	9
Chapter 3: Installing SAS/C Cross-Platform Software on your PC System	11
Installing from a CD-ROM	11
Running setup.exe	11
Registry Updates	12
Updating the Registry after Installation	12
Uninstalling the SAS/C Cross-Platform Software	13
Transferring Files to MVS	13
Using the IDE to Transfer Files to the Mainframe.	14
Chapter 4: Independent Mainframe SAS/C Cross-Platform Software Installation	15
Installing the Independent SAS/C Cross-Platform Software on MVS.	16
Chapter 5: Mainframe SAS/C Cross-Platform Software Installation	21
Installing SAS/C CSL on MVS	22
Chapter 6: Installation Verification	25
Verify SAS/C Cross-Platform Compiler Installation	25
Verify SAS/C Cross-Platform C++ Development System Installation	26
Chapter 7: Code Maintenance	27
SAS/C Usage Notes	27
How to Receive Updates for SAS/C Cross-Platform Software	27
How to Apply Updates	27
sascc.cfg Configuration File	27
Product License Maintenance for the SAS/C Cross-Platform C Compiler	28
Product License Maintenance for the SAS/C Cross-Platform C++ Software	29
Chapter 8: Customizing SAS/C Installation	31
Making the Transient Library Available to C Programs on MVS	31
Customizing SAS/C Help Information on MVS	34
Customizing Temporary File Defaults on MVS	35
Customizing Dynamic Allocation Defaults on MVS.	37
Customizing Data-in-Virtual Defaults on MVS.	41
Customizing CICS Library Support	43
Customizing TCP/IP Defaults on MVS	51
Appendix A: Utility Command Formats	55
dset	55
objzap	56
zap.	60
Appendix B: Reference Publications	61

Getting Started

Purpose

This document describes the installation of SAS/C® Cross-Platform Software for the following operating systems:

- Windows 95
- Windows 98
- Windows NT, Windows 2000, and Windows XP.

The products included with this release are:

- SAS/C Cross-Platform Compiler
- SAS/C Cross-Platform C++ Development System
- SAS/C Cross-Platform Resident Library for OS/390

Additionally, this guide contains valuable information on product maintenance and other topics that will be useful in the future. Therefore, we urge you to keep this document for later reference.

Note: For the purpose of this documentation, z/OS, OS/390, and MVS are synonymous terms.

Conventions

This section covers the typographical conventions this manual uses. You will see several type styles in this manual. Style conventions are summarized here:

<code>roman</code>	is the basic type style used for most text in this manual.
<code>monospace</code>	is used for language elements in the text and in programming code.
<i>italic</i>	is used for environment variable elements that are specific to the user's site. It is also used to indicate terms that are defined in text.

Terms

Independent is defined as a customer who has licensed SAS/C Cross-Platform Software independently of any SAS/C mainframe software.

Federal Government Rights Notice

If your installation is a Federal Government site or a Federal Government Prime Contractor site, you are responsible for the information contained in a usage rights notice that has been included on the installation media. Please review the Government Rights Notice information contained in the file `govnote.txt` as soon as the installation is completed.

1 Introduction

This document provides instructions for installing the SAS/C Cross-Platform Software on a PC. Please read these instructions before you attempt to install any SAS/C Cross-Platform Software product on Windows 95, Windows 98, Windows NT, Windows 2000, and Windows XP.

Installation

The basic install process is the same, regardless of the combination of SAS/C products your site has licensed. Every installation consists of the following steps.

1. Verify the disk space needed on your PC system. See “Disk Space Requirements” on page 9.
2. Load the SAS/C Cross-Platform software products on your PC system. See “Installing SAS/C Cross-Platform Software on your PC System” on page 11.
3. If you are not currently a licensed SAS/C mainframe customer (i.e., an independent customer), you will need to install a subset of the cross-platform software on your mainframe. See “Independent Mainframe SAS/C Cross-Platform Software Installation” on page 15.

Note: If you are an independent customer, installation of a subset of SAS/C CSL redistributable files (see step 4 below) are automatically installed during step 3.

4. If you are a currently licensed mainframe customer and plan to use the cross-platform facilities of the SAS/C debugger on MVS but do not have SAS/C CSL installed at your site, you will need to install a subset of the SAS/C CSL redistributable files. See “Mainframe SAS/C Cross-Platform Software Installation” on page 21.
5. Review the `readme.txt` file in the installation directory. This file contains directions that will guide you to the SAS/C online tutorial and help files. Included in the tutorial and help files is information for setting options at the source and project level, as well as automating the FTP process within the Microsoft Visual C++ IDE. For more information on automating the FTP process, see “Transferring Files to MVS” on page 13.
6. Validate the installation. See “Installation Verification” on page 25.
7. Perform code maintenance, if necessary. See “Code Maintenance” on page 27.
8. Optionally, perform SAS/C installation customizations. See “Customizing SAS/C Installation” on page 31.

2

Disk Space Requirements

Use the following table to determine the minimum disk space required to install the product combination that your site has licensed on your PC system.

TABLE 1.

SAS/C Cross-Platform	Size in Megabytes
Compiler	33
C++ Development System	22
Resident Library for MVS	10.0

In addition to the space needed for the products listed above, you will need approximately 10.0 megabytes of space on your PC system. This space will be used to load the files needed for the mainframe installation of the SAS/C Debugger and SAS/C CSL redistributable files. These redistributable files require no additional license to be installed on a mainframe and are part of the SAS/C library and the SAS/C CSL products.

Your installation media may contain maintenance in the form of SAS/C Cross-Platform software updates. See “Code Maintenance” on page 27 for additional information. The installation file, `setup.exe`, will extract the updates for the Typical Setup Type, or will optionally extract the updates for the Custom Setup Type from the installation CD-ROM. If you choose to install the updates, you will need to provide additional disk space.

3

Installing SAS/C Cross-Platform Software on your PC System

This chapter explains how to install the SAS/C Cross-Platform Software from CD-ROM.

Caution: Relocating the SAS/C Cross-Platform Software. The installation process sets up a configuration file named `sascc.cfg`. This file contains important location information used by the C and C++ compiler drivers. If you move the software after the initial installation, you will need to modify the two environment variables inserted in the `sascc.cfg` configuration file. See “Code Maintenance” on page 27 for more information.

Updates to the SAS/C Cross-Platform Compiler may be included on the installation media. You will be given the option to install updates at installation time.

Installing from a CD-ROM

Before installing the SAS/C Cross-Platform Software from CD-ROM, make sure you know the name of the CD-ROM drive attached to your PC system.

To install from CD-ROM, follow these steps:

1. Place the SAS/C Cross-Platform Software CD-ROM in your PC's CD-ROM drive.
2. Perform **one** of the following series of steps to install the SAS/C Cross-Platform software on Windows 95, Windows 98, Windows NT, Windows 2000, and Windows XP:

- Use the Start button on the Taskbar to install the software:

- Select the Start button on the Taskbar.
- Select the Run... option.
- Type the CD-ROM drive name followed by `\setup.exe` in the Run dialog box.
- Select the OK button.

Or

- Use Windows Explorer to install the software:

- Open a Windows Explorer session.
- Select the icon for your PC's CD-ROM drive.
- Select `setup.exe`.

Running setup.exe

InstallShield was used to create the `setup.exe` file. To install the SAS/C Cross-Platform Software you have to run the `setup.exe` program located on the CD-ROM.

Several dialog boxes will provide instructions or prompt you for information when you run the `setup.exe` program to install the SAS/C Cross-Platform Software. Following is a list of the dialog boxes and their instructions or prompts:

1. The Welcome dialog box advises you to close all Windows applications before proceeding with the installation. Select the Next button after you have closed all the applications.

2. The Legal Notices dialog box advises you to acknowledge the preceding notices. Select the **Yes** button to continue the installation.
3. The User Information dialog box prompts you for your name and company details. Select the **Next** button after entering this information.
4. The Choose Destination Location dialog box gives you the option to override the default installation location on your PC, `C:\Program Files\SAS Institute\SASC756`. By selecting the **Browse** button, you can determine an alternative destination for the installation. Select the **Next** button after you have completed any changes you want to make on this dialog box.
5. The Setup Type dialog box describes the two variations you can select for the setup. The options are as follows:
 - **Typical** (default installation)
Install all licensed products.
 - **Custom**
Install selected components of the licensed products.
6. The Select Components dialog box opens **ONLY** if you selected **Custom** in the Setup Type dialog box. You can select which components to install from a list that details the products you have licensed. A dynamic calculation of the space required is made to ensure you have enough disk space available for the installation.
7. The Select Sub-components dialog box appears if you click the **Change** button on the Select Components dialog box. The Select Sub-components dialog box enables the discrete selection of sub-components for each of the licensed products. Click the **Continue** button to return to the Select Components dialog box, which allows you to select more sub-components of a licensed product.
8. The Microsoft IDE Customization dialog box gives you the option to run the SAS/C Cross-Platform Software from within the Microsoft Visual C++ development environment. If you select this option, see “Registry Updates” on page 12 for details about updates to the Registry. These updates do not occur if you select the **No** option on the Microsoft IDE Customization dialog box. Select the **Next** button after you choose an option on this dialog box.
9. The Microsoft Visual C++ Version dialog box prompts you to enter the version of Microsoft Visual C++ currently installed on your PC. The Registry is updated according to the version you select. This dialog box opens only if you selected the **YES** option on the Microsoft IDE Customization dialog box.
10. The Setup Complete dialog box appears as the final dialog box. Setup has completed copying files to your computer, and the Registry is updated. It is advisable to select the option “Yes. I want to restart my computer now.” to ensure that the installation correctly updates the Registry.

Registry Updates

If you selected the option to run the SAS/C Cross-Platform Software from within the Microsoft Visual C++ development environment, your Registry has been updated according to the version of Microsoft Visual C++ you have installed on your PC.

See the online help files for more information on Registry updates. You can access the online help by typing the following command in a DOS shell:

```
sashelp.hlp
```

Updating the Registry after Installation

If you selected **No** in the Microsoft IDE Customization dialog box during the initial installation, you can still update the Registry value entries to ensure correct implementation of the sascc370

driver within the Microsoft Visual C++ IDE by executing the `setup.exe` again from the CD-ROM containing the SAS/C Cross-Platform product. In the Choose Destination Location dialog box, the `installdir`, which reflects the current location of the SAS/C++ Cross-Platform software already installed on your PC, is automatically taken from the environment variable `SASCDEV` and entered into the appropriate text box. Select the **Custom** option in the Setup Type dialog box. When the Select Components dialog box appears, deselect all the individual components, then select the **Next** button. Select **Yes** in the Microsoft IDE Customization dialog box, and then select the version of the Microsoft software currently installed on your PC in the subsequent Microsoft Visual C++ Version dialog box. This procedure will update the Registry and will not impact the SAS/C Cross-Platform software already installed.

Uninstalling the SAS/C Cross-Platform Software

After you install the SAS/C Cross-Platform software, a file named `SASC.isu` is placed in the `installdir` which enables you to uninstall the software. Uninstalling the software will completely remove the SAS/C products from your PC.

If you selected to run the SAS/C Cross-Platform software from within the Microsoft Visual C++ environment, use the following procedure to uninstall the software:

1. Run `uninstall42`, `uninstall50`, or `uninstall60` from a DOS shell to return the Registry to its original settings. Use `uninstall42` if you are using Microsoft Visual C++ Version 4.2 with the SAS/C Cross-Platform software. Use `uninstall50` if you are using Microsoft Visual C++ Version 5.0 with the SAS/C Cross-Platform software. Use `uninstall60` if you are using Microsoft Visual C++ Version 6.0 with the SAS/C Cross-Platform software.
2. Select the **Add/Remove Programs** icon in the Control Panel.
3. Select `SASC 7.50` in the **Add/Remove Programs Properties** dialog box.

If you selected **NOT** to run the SAS/C Cross-Platform software from within the Microsoft Visual C++ environment, use the following procedure to uninstall the software:

1. Select the **Add/Remove Programs** icon in the Control Panel.
2. Select `SASC 7.50` in the **Add/Remove Programs Properties** dialog box.

Transferring Files to MVS

During the SAS/C Cross-Platform software installation the following batch files are copied to the appropriate directories:

TABLE 2.

Files	Destination Directory
<code>ftp_mvs.bat / ftp_mvs.dat</code>	<code>installdir\lib\mvs\sascindp</code>
<code>ftp_mvs_csl.bat / ftp_mvs_csl.dat</code>	<code>installdir\sasccsl\mvs</code>

`installdir` is the location on your PC allocated for your installation of the software.

Note: See “Independent Mainframe SAS/C Cross-Platform Software Installation” on page 15 and “Mainframe SAS/C Cross-Platform Software Installation” on page 21 for information about the above batch files and transferring the files to MVS systems.

Using the IDE to Transfer Files to the Mainframe

If you integrated the SAS/C product into the Microsoft Visual C++ IDE, you can automate the process of FTPing files from your PC to the mainframe with the following procedure:

1. Select the **Project** pulldown from the toolbar.
2. Select the **Settings** option.

The Project Settings dialog box appears.

3. Scroll through the tabs to the **Post-build step** tab and select it.
4. Enter the pathname to the FTP batch file in the **Post-build command** list box.
5. Click on the **OK** pushbutton.

For more information on automating the FTP process, refer to “Executing FTP Batch Files” under the Installation topic in the online help files.

4

Independent Mainframe SAS/C Cross-Platform Software Installation

The instructions in this chapter are for SAS/C sites who licensed the SAS/C Cross-Platform Software independently of any SAS/C mainframe software. If you are a currently licensed SAS/C mainframe site, see “Installation” on page 7 for the installation steps you need to follow.

If you are installing SAS/C Cross-Platform Software on MVS, refer to “Installing the Independent SAS/C Cross-Platform Software on MVS” on page 16.

Caution: If you are a currently licensed SAS/C mainframe site, **do not** use the instructions in this chapter.

Installing the Independent SAS/C Cross-Platform Software on MVS

Perform the following steps to install SAS/C Cross-Platform Software on MVS.

1. Transfer Files to MVS

When you ran the installation file, `setup.exe`, the `lib\mvs\sascindp`, `lib\mvs\sascindp\csl`, and `lib\cics\sascindp` directories were created in your installation location or directory. These directories contain the files needed for the MVS SAS/C installation.

MVS data sets need to be allocated and specific files need to be transferred from your PC to MVS. The data sets that need to be allocated are listed in “Allocate Data Sets for Installation” on page 16. The files that need to be transferred to MVS are listed in “Which Files to Transfer to MVS” on page 17.

You may use your preferred method to allocate the data sets and move the files, or you may use the files, `ftp_mvs.bat` and `ftp_mvs.dat`, that were installed in `\lib\mvs\sascindp` in your installation location. If you choose to use `ftp_mvs.bat`, you need to modify the **Prefix** and **Path** values in file `ftp_mvs.dat`. You will also need to modify the **Path** value in `ftp_mvs.bat`. **Prefix** should specify the high-level prefix to use for the MVS data set names, and **Path** is your installation location. You also need to change the **Userid** and **Password** values in file `ftp_mvs.dat`. After modifying these four values, execute `ftp_mvs.bat` by issuing the following command from the DOS prompt:

```
ftp_mvs
```

Allocate Data Sets for Installation

If you choose not to use the FTP batch file, you will need to allocate the following data sets to hold files that will be transferred from your PC system.

TABLE 3.

Data Set	Attributes
<i>prefix</i> .CNTL	SPACE=(6160,(5,1,1)),RECFM=FB, DSORG=PO, LRECL=80, BLKSIZE=6160
<i>prefix</i> .CLIST	SPACE=(6160,(5,1,1)),RECFM=FB, DSORG=PO, LRECL=80, BLKSIZE=6160
<i>prefix</i> .DBGHELP	SPACE=(4080,(300,1)), RECFM=FBS, DSORG=PS, LRECL=80, BLKSIZE=4080
<i>prefix</i> .LMPORT.OBJ	SPACE=(3120,(180,1)), RECFM=FB, DSORG=PS, LRECL=80, BLKSIZE=3120
<i>prefix</i> .PORTLIB	SPACE=(6120,(2200,1,1)),RECFM=V, DSORG=PO, LRECL=255, BLKSIZE=6120

Which Files to Transfer to MVS

Transfer the following SAS/C files from your PC to MVS using the FTP batch file or another method that you prefer.

TABLE 4.

File	Definition
instlmvs.jcl	is the install JCL. Move this file to <i>prefix</i> .CNTL(INSTLMVS).
dbghelp.clt	is the SAS/C Debugger help file clist. Move this file to <i>prefix</i> .CLIST(L\$DBHELP).
sascdbg.clt	is the clist that invokes the remote debugger. Move this file to <i>prefix</i> .CLIST(SASCDBG).
cicsaloc.jcl	is JCL used for CICS customizations. Move this file to <i>prefix</i> .CNTL(CICSALOC).
cicscsd	is used for CICS customizations. Move this file to <i>prefix</i> .CNTL(CICSCSD).
cicsdct	is used for CICS customizations. Move this file to <i>prefix</i> .CNTL(CICSDCT).
cicspct	is used for CICS customizations. Move this file to <i>prefix</i> .CNTL(CICSPCT).
cicsppt	is used for CICS customizations. Move this file to <i>prefix</i> .CNTL(CICSPPT).
dbghelp.hyp	is the SAS/C Debugger help file. Move this file to <i>prefix</i> .DEGHELP.
lmpport.obj	is the LMPORT utility object file. Move this file to <i>prefix</i> .LMPORT.OBJ.
mvsrtl.lmp	is the SAS/C transient library. Move this file to <i>prefix</i> .PORTLIB(LINKLIB).
tsoload.lmp	is the TSO command processor support and TSO environment variable support library. Move this file to <i>prefix</i> .PORTLIB(TSOLOAD).
cicsrtl.lmp	is the SAS/C CICS transient load library. Move this file to <i>prefix</i> .PORTLIB(CICSLOAD).
cslrtl.lmp	is the SAS/C CSL transient library. Move this file to <i>prefix</i> .PORTLIB(CSLRTL).
rdload.lmp	is the SAS/C CSL redistributable utilities. Move this file to <i>prefix</i> .PORTLIB(RDLOAD).
smpload.lmp	is the SAS/C CSL sample library. Move this file to <i>prefix</i> .PORTLIB(SMPLOAD).

2. Edit the Installation Job, INSTLMVS

You should find the installation job in *prefix*.CNTL(INSTLMVS). The first three lines of the INSTLMVS job contain the job card to be used for executing the job. Modify the

ACCOUNT-CODE and *PROGRAMMER-NAME* values to reflect those used by your site. The next few lines in the INSTLMVS job have parameters that need to be modified. These parameters are:

TABLE 5.

Parameter	Definition
PREFIX= 'SASC.PREFIX'	should reflect the <i>prefix</i> value used in "Allocate Data Sets for Installation" on page 16.
PORTLIB= 'YOUR.PORTLIB.HERE'	this is <i>prefix</i> . PORTLIB used in "Allocate Data Sets for Installation" on page 16.
OBJDS= 'YOUR.LMPORT.OBJ'	this is <i>prefix</i> . LMPORT . OBJ used in "Allocate Data Sets for Installation" on page 16.

3. Run the Installation Job, INSTLMVS

After modifying the JCL according to your site requirements, submit the INSTLMVS job. The INSTLMVS job will create the SAS/C libraries listed below.

TABLE 6.

Library	Definition
<i>prefix</i> . LINKLIB	SAS/C transient library
<i>prefix</i> . TSOLOAD	SAS/C TSO command processor support and TSO environment variable support library
<i>prefix</i> . CICSLOAD	SAS/C CICS transient load library
<i>prefix</i> . CSL . LOADLIB	SAS/C CSL transient library
<i>prefix</i> . CSL . RD . LOAD	SAS/C CSL redistributable utility library - contains NFS User Commands
<i>prefix</i> . CSL . SAMPLE . LOAD	SAS/C CSL sample library

4. Modify the SAS/C Debugger Help File clist

During installation, the SAS/C Debugger help file clist was installed in *prefix* . CLIST(L\$DBHELP). The file is similar to this:

```

/* COPYRIGHT (C) 1998 BY SAS INSTITUTE INC. ALL RIGHTS RESERVED. */
/*
/* NAME:          L$DBHELP                      (L$DBHELP) */
/* PRODUCT:      SAS/C                          */
/*
/* PURPOSE:      ALLOCATE DEBUGGER HELP FILE    */
/*
ALLOCATE FILE(DBGHELP) DA(`%PREFIX..DBGHELP') SHR

```

You will need to edit this clist and substitute the name of your SAS/C debugger help library in place of ` %PREFIX..DBGHELP' .

5. Modify the Remote Debugger clist

During installation, the SAS/CDBG clist was installed in *prefix*.CLIST(SASCDBG). The file is similar to the following:

```

/*REXX*/
parse arg args
/*-----*/
/* COPYRIGHT (C) 1995 BY SAS INSTITUTE INC. ALL RIGHTS RESERVED.*/
/*
/* NAME: SASCDBG
/* PRODUCT: SAS/C
/*
/* PURPOSE: Invoke the Remote DEBUGGER as a TSO command
/*
.
.
.
.
/* Name of Transient library and remote debugger module
DBGLOAD="%PREFIX..LINKLIB(L$DBRMT)"

```

You will need to edit this clist and substitute the name of your SAS/C transient library in place of *%PREFIX.LINKLIB*.

6. Optionally, install the SAS/C Remote Debugger to run under UNIX System Services (USS)

If you wish to use the SAS/C debugger under UNIX System Services (USS), you will need to install the remote debugger, *sascdbg*, into the Hierarchical File System. To install the remote debugger, perform the following steps:

A. Transfer the remote debugger (*sascdbg*) and the installation job (*SASCCPOE*) to MVS.

During the PC installation, *sascdbg* was placed in *lib\mvs\sascindp\oemvs.tar* in your installation location or directory. You will need to perform a binary transfer of the file to MVS as:

```

prefix.OEMVS.TAR SPACE=(3120,(200,1)),RECFM=FB,
DSORG=PS,LRECL=1,BLKSIZE=3120

```

The OEMVS installation job, *SASCCPOE*, was installed during the PC installation into *lib\mvs\sascindp\sasccpoe.jcl* in your installation location or directory. You will need to transfer this file to MVS as *prefix.CNTL(SASCCPOE)*. The *prefix.CNTL* data set was created in "Allocate Data Sets for Installation" on page 16.

B. Edit the USS installation job, *SASCCPOE*.

The first three lines of *prefix.CNTL(SASCCPOE)* contain the job card to be used for executing the job. Modify the *ACCOUNT-CODE* and *PROGRAMMER-NAME* values to

reflect those used by your site. The next several lines in the SASCCPOE job have parameters that need to be modified. These parameters are:

TABLE 7.

Parameter	Definition
PREFIX=' SASC . PREFIX '	should reflect the <i>prefix</i> value used in “Allocate Data Sets for Installation” on page 16
DIR=' /usr/local '	specifies the location of ‘bin’ installation directory for the SAS/C remote debugger
UNIT=DISK	specifies the unit type of the storage device where data sets may be created.
USERID=YOURUSERID	specifies a high level prefix that is used for creating an installation data set which is later deleted.

C. Run the installation job, SASCCPOE.

After modifying the JCL according to your site requirements, submit the SASCCPOE job. SASCCPOE copies *prefix.OEMVS . TAR* into the HFS and installs *sascdbg* in the bin directory at the location specified by the DIR parameter.

Your Installation is complete for MVS.

SAS/C Cross-Platform Limited Distribution Library Files for MVS

The following list contains all of the MVS SAS/C programs and libraries that are redistributable at no charge.

```

.\lib\mvs\sascindp\mvsrtl.lmp
.\lib\mvs\sascindp\tsoload.lmp
.\lib\cics\sascindp\cicsrtl.lmp
.\lib\mvs\sascindp\dbghelp.hyp
.\lib\mvs\sascindp\dbghelp.clt
.\lib\mvs\sascindp\sascdbg.clt
.\lib\mvs\sascindp\csl\cslrtl.lmp
.\lib\mvs\sascindp\csl\rdload.lmp
.\lib\mvs\sascindp\csl\smpload.lmp
.\lib\mvs\sascindp\instlmvs.jcl
.\lib\mvs\sascindp\lmpport.obj
.\lib\mvs\libares.a
.\lib\cics\libares.a

```

5

Mainframe SAS/C Cross-Platform Software Installation

The instructions in this chapter are for SAS/C mainframe (OS/390 or z/OS) sites who currently have SAS/C release 7.50 installed, and want to use their SAS/C mainframe debugger for cross debugging. The CSL files shipped with the SAS/C Cross-Platform Software are the same files that are distributed with release 7.50 of the SAS/C CSL product. You should skip this chapter if your site has SAS/C CSL release 7.50 licensed and installed.

If you plan to use the mainframe for cross debugging, you will need to:

- ❑ Provide support for the Network File System (NFS) client. Refer to *SAS/C Cross-Platform Compiler and C++ Development System, Usage and Reference, Release 7.00* and *SAS/C Debugger User's Guide and Reference, Release 7.00* for administration and use of the NFS client support in the SAS/C debugger.

Installing SAS/C CSL on MVS

Perform the following steps to install SAS/C CSL on MVS.

1. Transfer Files to MVS

When you ran the installation file, `setup.exe`, the `sasccsl\mvs` directory was created in your installation location or directory. The `sasccsl\mvs` directory contains the files needed for the MVS SAS/C CSL redistributables installation.

You need to allocate MVS data sets to hold the redistributable files and then move the files from your PC to MVS. The data sets that need to be allocated are listed in “Allocate Data Sets for Installation” on page 22. The files that need to be transferred to MVS are listed in “Which Files to Transfer to MVS” on page 23.

You may use your preferred method to allocate the data sets and move the files. Otherwise, you may use the files, `ftp_mvs_csl.bat` and `ftp_mvs_csl.dat`, that were installed in `\sasccsl\mvs` in your installation location. If you choose to use `ftp_mvs_csl.bat`, you need to modify the **Prefix** and **Path** values in file `ftp_mvs_csl.dat`. You will also need to modify the **Path** value in `ftp_mvs_csl.bat`. **Prefix** should specify the high-level prefix to use for the MVS data set names, and **Path** is your installation location. You also need to change the **Userid** and **Password** values in file `ftp_mvs_csl.dat`. After modifying these four values, execute `ftp_mvs_csl.bat` by issuing the following command from the DOS prompt:

```
ftp_mvs_csl
```

Allocate Data Sets for Installation

If you choose not to use the FTP batch file, you will need to allocate the following data sets to hold files that will be transferred from your PC to MVS.

Note: You should use the same value for *prefix* below, that you use in the install job, CROSSCSL. See “2. Edit the Installation Job, CROSSCSL” on page 23 for more information on modifying the value of *prefix*.

TABLE 8.

Data Set	Attributes
<i>prefix</i> .CNTL	SPACE=(6160,(5,1,1)), RECFM=FB, DSORG=PO, LRECL=80, BLKSIZE=6160
<i>prefix</i> .LMPORT.OBJ	SPACE=(3120,(180,1)), RECFM=FB, DSORG=PS, LRECL=80, BLKSIZE=3120
<i>prefix</i> .PORTLLIB	SPACE=(6120,(700,1,1)), RECFM=V, DSORG=PO, LRECL=255, BLKSIZE=6120

Which Files to Transfer to MVS

Transfer the following SAS/C CSL files using the method you prefer.

TABLE 9.

File	Definition
<code>crosscsl.jcl</code>	is the install JCL. Move this file to <i>prefix</i> .CNTL(CROSSCSL).
<code>lmpport.obj</code>	is the LMPORT utility object file. Move this file to <i>prefix</i> .LMPORT.OBJ.
<code>cslrtl.lmp</code>	is the SAS/C CSL transient library. Move this file to <i>prefix</i> .PORTLIB(CSLRTL).
<code>rdload.lmp</code>	is the SAS/C CSL utility command library. Move this file to <i>prefix</i> .PORTLIB(RDLOAD).
<code>smpload.lmp</code>	is the SAS/C CSL sample command library. Move this file to <i>prefix</i> .PORTLIB(SMPLOAD).

2. Edit the Installation Job, CROSSCSL

You should find the installation job in *prefix*.CNTL(CROSSCSL). The first three lines of the CROSSCSL job contain the job card to be used for executing the job. Modify the ACCOUNT-CODE and PROGRAMMER-NAME values to reflect those used by your site. The next few lines in the CROSSCSL job have parameters that need to be modified. These parameters are:

TABLE 10.

Parameter	Definition
<code>PREFIX='SASC.PREFIX'</code>	should reflect the <i>prefix</i> value used in "Allocate Data Sets for Installation" on page 22
<code>PORTLIB='YOUR.PORTLIB.HERE'</code>	this is <i>prefix</i> .PORTLIB used in "Allocate Data Sets for Installation" on page 22
<code>OBJDS='YOUR.LMPORT.OBJ'</code>	this is <i>prefix</i> .LMPORT.OBJ used in "Allocate Data Sets for Installation" on page 22

3. Run the Installation Job, CROSSCSL

After modifying the JCL according to your site requirements, submit the CROSSCSL job. The CROSSCSL job will create the SAS/C CSL libraries listed below. You will need to make these libraries available for use in cross debugging.

TABLE 11.

Library	Definition
<i>prefix</i> .CSL.LOADLIB	SAS/C CSL transient library
<i>prefix</i> .CSL.RD.LOAD	SAS/C CSL redistributable utility library - contains NFS User Commands
<i>prefix</i> .CSL.SAMPLE.LOAD	SAS/C CSL sample library

6

Installation Verification

Verify SAS/C Cross-Platform Compiler Installation

To verify the SAS/C Cross-Platform compiler installation, compile and prelink a sample C program. You will find the sample source file `ftoc.c` in the `samples\c` directory in your installation location. To compile and prelink `ftoc.c`, run the `sascc370` driver which was installed in your host `\wnt\bin` directory in your installation location:

```
sascc370 -v installdir\samples\c\ftoc.c
```

TABLE 12.

Variable	Definition
<i>installdir</i>	your installation location or directory

Note: In this example, `sascc370` generates extra output because the `-v` option is used. (The `-v` option turns on the verbose mode of the driver, compiler, and prelinker components.) The extra output should aid in verifying that the compiler installation is correct. The output is greatly reduced without the `-v` option.

You should see output from the `sascc370` command similar to the following:

```
SAS/C Compiler Driver V7.50.06
Copyright (C) 2004 SAS Institute Inc.
set INCLUDE370='installdir\Include'

installdir\host\wnt\bin\lc1 -dCROSS370=1 -cd -hu -n! '-oc:\TEMP\
sascc2.1.q'
"installdir\samples\c\ftoc.c"
SAS/C Release 7.50.06 (Target 370 Cross Compiler)
Copyright(c) 2004 by SAS Institute Inc. All Rights Reserved.
*** No errors; No warnings; No user suppressed warnings

installdir\host\wnt\bin\lc2 '-oftoc.obj' "c:\TEMP\sascc2.1.q"
SAS/C Compiler(Phase 2)Release 7.50.06
Copyright(c) 2004 by SAS Institute Inc. All Rights Reserved.

installdir\host\wnt\bin\cool -o a.out -Linstalldir ftoc.obj
installdir/lib/mvs/libc.a
SAS/C (R) C Object Code Pre-linker Release 7.50.06
Copyright(c) 2004 by SAS Institute Inc. All Rights Reserved.

cool: Note 1010: Pre-linking completed with return code = 0
```

Verify SAS/C Cross-Platform C++ Development System Installation

To verify the SAS/C Cross-Platform C++ Development System installation, compile and prelink a sample C++ program. You will find the sample source file `ftoc.cxx` in the `samples\cxx` directory in your installation location. To compile and prelink `ftoc.cxx`, run the `sascc370` driver which was installed in your host\wnt\bin directory in your installation location.

Note: The example below creates an object file to transfer to MVS for the final linking and execution. To compile and prelink `ftoc.cxx` for MVS, issue this command:

```
sascc370 -v installdir\samples\cxx\ftoc.cxx
```

TABLE 13.

Variable	Definition
<code>installdir</code>	your installation location or directory

Note: In this example, `sascc370` generates extra output because the `-v` option is used. (The `-v` option turns on the verbose mode of the driver, compiler, and prelinker components). The extra output should aid in verifying that the compiler installation is correct. The output is greatly reduced without the `-v` option.

You should see output from the `sascc370` command similar to the following:

```
SAS/C Compiler Driver V7.50.06
Copyright (C) 2004 SAS Institute Inc.
set INCLUDE370='installdir\Include'

installdir\host\wnt\bin\cxx -Adigraph1 -Adigraph2 -DCROSS370=1
-XA -Hu '-mftoc'
"installdir\samples\cxx\ftoc.cxx" 'c:\TEMP\sascc2.1.c'
SAS/C C++ 7.50.064Copyright (c) 2004 SAS Institute Inc.

installdir\host\wnt\bin\lc1 -dCROSS370=1 -cd -hu -n! -cxx '-sftoc'
'-oc:\TEMP\sascc3.1.q' "c:\TEMP\sascc2.1.c"
SAS/C Release 7.50.06 (Target 370 Cross Compiler)
Copyright(c) 2004copyright(c) 2004 by SAS Institute Inc. All Rights
Reserved.
*** No errors; No warnings; No user suppressed warnings

installdir\host\wnt\bin\lc2 '-oftoc.obj' "c:\TEMP\sascc3.1.q"
SAS/C Compiler(Phase 2)Release 7.50.06
Copyright(c) 2004 by SAS Institute Inc. All Rights Reserved.

installdir\host\wnt\bin\cool -o a.out -Linstalldir ftoc.obj
installdir/lib/libcxx.a
installdir/lib/mvs/libc.a
SAS/C (R) C Object Code Pre-linker Release 7.50.06
Copyright(c) 2004 by SAS Institute Inc. All Rights Reserved.

cool: Note 1010: Pre-linking completed with return code = 0
```

7

Code Maintenance

SAS/C Usage Notes

The Technical Support Division of SAS Institute Inc. provides machine readable usage notes and zaps for the SAS/C products to assist you in diagnosing and correcting known problems with the products.

Usage notes are intended to be used as a reference in the event that you encounter a problem with one of the products. If your problem has already been reported, it may be documented in a usage note. The usage note will indicate whether there is a zap to correct the problem. Cautions and warnings about side effects will also be described in the usage note.

How to Receive Updates for SAS/C Cross-Platform Software

Updates to the SAS/C Cross-Platform Software are provided in the form of patches. These updates are available at the following SAS Institute web site:

<http://www.sas.com/products/sasc/index.html>

Select “Search Cross-Platform Maintenance”. This page contains links to the latest fixes and enhancements for the SAS/C Cross-Platform Software.

How to Apply Updates

The zap utility is provided as part of the SAS/C Cross-Platform Software product for applying maintenance in the form of binary patches to SAS/C Cross-Platform products. See “zap” on page 60. for the complete format of this utility.

sascc.cfg Configuration File

During the initial install process, a configuration file called `sascc.cfg` was created. `sascc.cfg` is found in the bin directory for your host:

```
installdir\host\wnt\bin\sascc.cfg
```

The compiler driver uses the values returned from `sascc.cfg` to determine the *installdir*, and thus, determine (using its own knowledge of the host on which it is running) locations of:

TABLE 14.

Component	Location
include files	<i>installdir</i> \include
standard library directory	<i>installdir</i> \lib
executables	<i>installdir</i> \host\wnt\bin

The following is an example `sascc.cfg`:

```
#
# PATH points to the <installation dir>
# INCLUDE points to <installation dir>\include
#
# The Environment variables SASDEV and SASINCLUDE
# are generated at the time of installation of the
# SAS/C C++ Cross-Platform product. Only change these
# two variables if you alter the location of the
# installation on your PC.
#
# Use the 'set' command in an MS-DOS shell

PATH=%SASCDEV%
INCLUDE=%SASCINCLUDE%
```

If you move the compiler you need to change the two environment variables SASDEV and SASINCLUDE in the Registry. To change the two environment variables:

1. Select the **Start** button on the Taskbar.
2. Select the **Run...** option.
3. Type `regedit` in the **Run** dialog box.
4. Select the **OK** button.
 - A `regedit` session opens.
5. Open the `HKEY_CURRENT_USER` handle.
6. Open the Environment subkey.
7. Double-click on SASDEV.
 - An **Edit String** dialog box appears.
8. Type the path to the new location in the **Edit String** dialog box.
9. Select **OK**.
10. Repeat Steps 7, 8, and 9 for SASINCLUDE.

Product License Maintenance for the SAS/C Cross-Platform C Compiler

The license information for SAS/C Cross-Platform C Compiler is contained within a binary license data file, `sascc_set.dat`, provided on the installation media. You can use the `dset` utility to display the information in the `sascc_set.dat` file. The `dset` utility is described in “Utility Command Formats” on page 55 and is on the installation media. The information used to update the license is known as the **setinit** zap file. A copy of the `setinit` zap file used for your site is also provided as one of the files on the installation media. It is named `sascc_set.zap`. This zap file contains data that corresponds to the data that was used to individually license your copy of the product before shipment.

Note: You do not need to initialize the license information during the installation procedure.

When you renew your license, you will be provided with a new `sascc_set.zap` file. Update the license information by running the `zap` utility provided on the installation media and described in “Utility Command Formats” on page 55. To update the license in the SAS/C Cross-Platform Compiler enter the following command:

```
zap sascc_set.zap sascc_set.dat
```

The `zap` utility modifies the `sascc_set.dat` file, which is your binary license data file.

Product License Maintenance for the SAS/C Cross-Platform C++ Software

The license information for SAS/C Cross-Platform C++ Software is contained within a binary license data file, `sascd_set.dat`, provided on the installation media. You can use the `dset` utility to display the information in the `sascd_set.dat` file. The `dset` utility is described in “Utility Command Formats” on page 55 and is on the installation media. The information used to update the license is known as the **setinit** zap file. A copy of the `setinit` zap file used for your site is also provided as one of the files on the installation media. It is named `sascd_set.zap`. This zap file contains data that corresponds to the data that was used to individually license your copy of the product before shipment.

Note: You do not need to initialize the license information during the installation procedure.

When you renew your license, you will be provided with a new `sascd_set.zap` file. Update the license information by running the `zap` utility provided on the installation media and described in, “Utility Command Formats” on page 55. To update the license in the SAS/C Cross-Platform C++ Software enter the following command:

```
zap sascd_set.zap sascd_set.dat
```

The `zap` utility modifies the `sascd_set.dat` file, which is your binary license data file.

8

Customizing SAS/C Installation

This chapter describes customizations for your SAS/C installation. In most cases these customizations are available in the form of zaps. The customization zaps should be applied to the SAS/C resident and all-resident libraries on the cross compiler platform, as well as to the SAS/C Transient Library on the mainframe. The OBJZAP utility is provided to install zaps to the SAS/C libraries on the cross compiler platform. The OBJZAP utility is documented in “objzap” on page 56. To apply the customization zaps to the SAS/C Transient Library, use the APPLYZAP utility. Refer to *A Guide for the SAS/C Compiler Consultant* for directions on using the APPLYZAP utility on the mainframe.

If you distribute all-resident applications you may need to provide a method for your customers to modify the installation defaults listed in this chapter. We recommend that you review the APPLYZAP utility, as documented in *A Guide for the SAS/C Compiler Consultant* and provide your customers with the all-resident customization zaps and the APPLYZAP utility.

Making the Transient Library Available to C Programs on MVS

The SAS/C product includes several transient libraries, which provide dynamically loaded run-time support for programs compiled with the SAS/C Compiler. Two versions of the transient library are provided: one for CICS applications and the other for non-CICS applications.

In order to increase the efficiency of C programs, the SAS/C runtime library is divided into two parts: resident and transient. Resident library modules are linked in with the executable program. Transient library modules are loaded as needed at runtime. This library structure reduces the memory requirements of C programs by including modules as needed during execution, instead of including all possibly required modules in the program at link-edit time.

Additionally, this organization allows transient library modules to be shared among many programs (if the modules are loaded into the LPA), which reduces real memory use and the I/O overhead involved in loading. Perhaps most importantly, this organization allows MVS/ESA C programs to run above the 16-megabyte line, while transient library modules that interface to I/O are loaded below the line (as required by MVS/ESA).

To have them available for loading at runtime, the appropriate transient libraries must be available on every system on which a SAS/C program is run. **To facilitate this process, the SAS/C transient libraries can be redistributed royalty-free to sites which use programs that were developed with the SAS/C product.**

Note: By default, most of the SAS/C transient library is loaded above the 16 megabyte line, but some pieces (for example, low-level I/O modules) have `AMODE 24` requirements and so must reside below the line.

Finding the Transient Library

In “Installing the Independent SAS/C Cross-Platform Software on MVS” on page 16, you copied the transient library into a disk data set. For C programs to find transient modules at runtime, you must make the modules available in one of the places where C programs search for transient modules. The search order is listed below. See “Making the CICS Transient Library Available to CICS on MVS” on page 43 for instructions on making the Transient Library available under CICS.

1. Data sets allocated to CTRANS (if any)
2. Any defined tasklib data sets. For instance, when a SAS/C program is run under ISPF, the DDname ISPLLIB is defined as a tasklib.
3. STEPLIB (if any)
4. JOBLIB (if any)
5. the LPA (Link Pack Area)
6. Linklist libraries

You can make the transient library available to a C program using any of these locations. The following describes the considerations that apply to each location.

Note: If running under ISPF, a possible search order is listed below, but consult your ISPF documentation for your exact search path.

1. CTRANS
2. tasklib (ISPLLIB)
3. STEPLIB
4. LPA
5. LINKLIST

Finding the Transient Library under the OpenEdition Shell

For programs running under the OpenEdition Shell (or invoked via the OpenEdition exec function), the following search order is used.

1. A data set named by the ddn_CTRANS environment variable, if any
2. Any data set defined by the STEPLIB environment variable
3. The LPA (Link Pack Area)
4. Linklist libraries

Also note that if you expect to run `setuid` or `setgid` programs written in SAS/C, you should put the library in either linklist or LPA. Alternatively, you can include the transient library in the STEPLIB environment variable list. In this case, you must also include the transient library data set name in the OpenEdition “sanction list” defined by the BPXPRMnn member of SYS1.PARMLIB.

CTTRANS

Use of the CTRANS DDname is convenient for testing purposes when you are installing a new version of the transient library. Since C programs search the libraries allocated to CTRANS first, you can test a new version of the transient library while an older version remains installed. Sites that are evaluating a SAS/C program, and may therefore want to remove the transient library at any time, could also use CTRANS to access the transient library. CTRANS is particularly useful in TSO because, unlike STEPLIBs, CTRANS can be allocated dynamically and thus requires no modifications to logon procedures.

STEPLIBs or JOBLIBs

STEPLIBs or JOBLIBs are useful for sites that do not want to put the transient library in the LPA or linklist. They are also useful for testing and evaluation.

LPA

The LPA is the recommended location for SAS/C transient library modules. Installing the transient library into the LPA decreases memory requirements for C programs and reduces system paging if the C library is heavily used.

Installing the transient library into the LPA simplifies JCL and TSO procedures for calling C programs since CTRANS, JOBLIBs, and STEPLIBs are not required. If you do not have enough

virtual storage available to copy the entire library to an LPA library, it is recommended that you copy a select list of modules to the LPA library. A recommended minimal list of modules to place in the LPA library is LSCARGP, LSCDIND, LSCDMGR, LSCOSEQ, LSCOTERM, LSCSIO, LSCIDDN, LSCIDSN, and LSCITSO. The debugger load module L\$DEBUG is also a very good candidate for inclusion if your site also has the SAS/C Compiler and makes frequent use of the debugger. Of course, any modules that are not copied into the LPA must be made available through one of the other locations (in this case, the linklist would usually be the most appropriate).

IMPORTANT: The modules LSCEVRG and L\$DCST must NOT be moved into the LPA. Placement of these modules into LPA will cause execution errors.

LINKLIST

Depending on your local site procedures, you would make transient library modules available in the linklist either by naming the transient library data set (copied from the installation medium in “Installing the Independent SAS/C Cross-Platform Software on MVS” on page 16) in the LNKLISTxx member of SYS1 . PARMLIB or by copying the modules into some other library whose name is already in the linklist (perhaps SYS3 . LINKLIB). This has none of the performance advantages of using the LPA but does not have the disadvantage of the LPA which decreases your system's available private area. It does share the advantage of making the transient library modules available without the need to code CTRANS, JOBLIBs, or STEPLIBs.

Customizing SAS/C Help Information on MVS

The debugger help file *prefix*.DBGHELP is NOT a TSO format help file, but has a special internal format which the SAS/C Debugger uses. From an MVS standpoint, the file is a standard physical sequential file, it cannot be a member of a PDS, and it must have the following DCB attributes when it is copied or allocated:

```
DCB= (DSORG=PS , RECFM=FBS , LRECL=80 , BLKSIZE=4080)
```

The *prefix*.DBGHELP can be made available to users by placing the *prefix*.CLIST member L\$DBHELP in your system CLIST library. When debugger help is requested and no DBGHELP DDname is allocated, the debugger attempts to invoke the L\$DBHELP CLIST to allocate the help file. The file tailoring procedures automatically update this CLIST to reflect the installation's name chosen for *prefix*.DBGHELP.

Note: *prefix*.HELP needs to be made available to TSO by either concatenating *prefix*.HELP to the DDname SYSHELP or by copying the members to some larger general-purpose library.

Customizing Temporary File Defaults on MVS

This section describes a library zap that can be used by MVS sites that want to change the defaults for temporary file UNIT and SPACE. The default UNIT is VIO, and the default SPACE allocation is (TRK,(50,0)).

C programs that issue UNIX-style I/O requests use temporary files for work areas. If the SYSTMPnn DDnames are not provided, temporary files are dynamically allocated to UNIT=VIO. This causes a problem for sites without UNIT=VIO defined.

The following generic zap must be applied to the transient library (*prefix*.LINKLIB) using APPLYZAP (see *A Guide for the SAS/C Compiler Consultant, Release 7.00* for details on using the APPLYZAP program) in order to change the temporary unit name or space allocation.

```
* NAME: Z7503101 PRODUCT: SASC CATEGORY: SPEC SYSTEM: MVS
* DATE: 02JAN04 STATUS: DZ+UT USAGE-ID: LIBRARY-C3101
*
* VIO as a unit name not defined at some installations
*
* NOTE: APPLY TO SASC.LINKLIB (TRANSIENT RUN-TIME LIBRARY IN
* LOAD MODULE FORMAT) USING THE APPLYZAP UTILITY.
*
* NOTE: THE USER MUST CODE THEIR OWN VALUES FOR UNIT,
* PRIMARY, AND SECONDARY SPACE.
* XXXXXX - PRIMARY TRACK ALLOCATION
* YYYYYY - SECONDARY TRACK ALLOCATION
* ZZZZZZZZ - UNIT NAME
*
NAME L$CITMP L$C$VIOO LINKLIB
CHECKSUM
VER 0000 0000,0032
VER 0004 0000,0000
VER 0008 E5C9,D640,4040,4040
*
REP 0000 00XX,XXXX
REP 0004 00YY,YYYY
REP 0008 ZZZZ,ZZZZ
IDRDATA Z7503101
```

Note: You do not have to replace data that you are not changing. For example, to change the unit name to DISK in the transient library, use the following:

```
NAME L$CITMP L$C$VIOO LINKLIB
VER 0008 E5C9D640
REP 0008 C4C9E2D2
```

If you are using the all-resident library, the following generic zap must be applied to `lib\mvs\libares.a` in your installation directory using `OBJZAP`. All relevant programs must be relinked. Instead of relinking, you may apply the zap to the `LCVIOO CSECT` in the appropriate user load module. Note that you must change the module name in the `NAME` statement to refer to your user load module when applying the zap to a user program.

```
*   NAME: Z7503101  PRODUCT: SASC      CATEGORY: SPEC      SYSTEM: MVS
*   DATE: 02JAN04  STATUS:  DZ+UT     USAGE-ID: LIBRARY-C3101
*
*   VIO as a unit name not defined at some installations
*
*   NOTE:          APPLY TO lib/mvs/libares.a (ALL-RESIDENT
*                 RUN-TIME LIBRARY IN OBJECT FORMAT) USING THE OBJZAP
*                 UTILITY.
*
*   NOTE:          RELINK OR ZAP ANY ALL-RESIDENT PROGRAMS WHICH INCLUDE
*                 THIS MODULE.
*
*                 NOTE: THE USER MUST CODE THEIR OWN VALUES FOR UNIT,
*                 PRIMARY, AND SECONDARY SPACE.
*                 XXXXXX - PRIMARY TRACK ALLOCATION
*                 YYYYYY - SECONDARY TRACK ALLOCATION
*                 ZZZZZZZ - UNIT NAME
*
NAME      L$CITMP  L$C$VIOO
CHECKSUM
VER       0000      0000,0032
VER       0004      0000,0000
VER       0008      E5C9,D640,4040,4040
*
REP       0000      00XX,XXXX
REP       0004      00YY,YYYY
REP       0008      ZZZZ,ZZZZ
IDRDATA  Z7503101
```

Note: You do not have to replace data that you are not changing. For example, to change the unit name to `DISK` in the transient library, use the following:

```
NAME L$CITMP L$C$VIOO
VER  0008 E5C9D640
REP  0008 C4C9E2D2
```

Customizing Dynamic Allocation Defaults on MVS

This section describes a zap that can be used by MVS sites that wish to change defaults for the dynamic allocation of new permanent data sets. (This is performed when “DSN:” or “TSO:” style file names are used by C programs calling `fopen()` or other library functions that accept a filename.) The current default is equivalent to `SPACE=(1000,(10,3,5))`. The following are examples of why you might wish to apply this zap:

- The default space allocation is too large or too small.
- Standards at your site call for use of a particular unit name or volume serial for such data sets.
- You wish to allocate data sets created by C programs using a specific SMS data class or storage class.

The following generic zap must be applied to the transient library (`prefix.LINKLIB`) using `APPLYZAP` (see *A Guide for the SAS/C Compiler Consultant, Release 7.00* for details on using the `APPLYZAP` program) in order to change the default allocations.

Certain areas being zapped are applicable only to sites using SMS (IBM's Storage Management Subsystem). Do not change these fields if SMS is not installed and operational.

```

*   NAME: Z7504048   PRODUCT: SASC       CATEGORY: SPEC   SYSTEM: MVS
*   DATE: 02JAN04   STATUS:  DZ+UT      USAGE-ID: LIBRARY-C4048
*
*   How to change the defaults for alcunit=, space= and extend= amparms
*
*   NOTE:           APPLY TO SASC.LINKLIB (TRANSIENT RUN-TIME LIBRARY IN
*                   LOAD MODULE FORMAT) USING THE APPLYZAP UTILITY.
*
*                   NOTE: THE USER MUST CODE THEIR OWN VALUES FOR PRIMARY AND
*                   SECONDARY SPACE, AVERAGE BLOCK LENGTH, NUMBER OF
*                   DIRECTORY BLOCKS, FLAGS, UNIT NAME, VOLUME, DATA CLASS,
*                   STORAGE CLASS AND MANAGEMENT CLASS. IF PADDING IS
*                   NEEDED, THESE VALUES SHOULD BE PADDED WITH BLANKS.
*
*                   IF SMS IS NOT INSTALLED AND ACTIVE, DO NOT INSTALL A DATA
*                   CLASS, STORAGE CLASS OR MANAGEMENT CLASS, AND DO NOT SET
*                   THE 08, 04, OR 02 FLAG BITS.
*
*                   XXXXXX - PRIMARY TRACK ALLOCATION
*                   YYYYYY - SECONDARY TRACK ALLOCATION
*                   LLLLLL - AVERAGE BLOCK LENGTH
*                   NNNNNN - NUMBER OF DIRECTORY BLOCKS
*                   FF      - FLAGS (80=TRACKS, 40=CYLINDERS, 20=BLOCKS
*                               08=RECORDS, 04=KILO-RECORDS,
*                               02=MEGA-RECORDS. SET THE 10 BIT TO
*                               DEFAULT RLSE=YES.)
*                   UUUUUUUUUUUUUUUUU - UNIT NAME
*                   ZZZZZZZZ - VOLUME
*                   DDDDDDDD - DEFAULT DATA CLASS
*                   SSSSSSSS - DEFAULT STORAGE CLASS
*                   MMMMMMMM - DEFAULT MANAGEMENT CLASS
*
*   END
NAME      L$CIDSN  L$C$DSN=      LINKLIB
CHECKSUM
VER       0008    0000,0A
VER       000B    0000,03
VER       000E    0003,E8
VER       0011    0000,05
VER       0014    20

```

```

VER      0015      0000,0000,0000,0000
VER      001D      0000,0000,0000
VER      0023      0000,0000,0000,0000
VER      002B      0000,0000,0000,0000
VER      0033      0000,0000,0000,0000
*
REP      0008      XXXX,XX
REP      000B      YYYY,YY
REP      000E      LLLL,LL
REP      0011      NNNN,NN
REP      0014      FF
REP      0015      UUUU,UUUU,UUUU,UUUU
REP      001D      ZZZZ,ZZZZ,ZZZZ
REP      0023      DDDD,DDDD,DDDD,DDDD
REP      002B      SSSS,SSSS,SSSS,SSSS
REP      0033      MMMM,MMMM,MMMM,MMMM
IDRDATA  Z7504048

```

Note: You do not have to replace data that you are not changing. For example, to make the default space allocation for new permanent data sets two cylinders of primary space and one cylinder of secondary space, use the following:

```

NAME L$CIDSN L$C$DSN=          LINKLIB
VER  0008 00000A
VER  000B 000003
VER  0014 20
REP  0008 000002
REP  000B 000001
REP  0014 40

```

If you are using the all-resident library, the following generic zap must be applied to `lib\mvs\libares.a` in your installation directory using OBJZAP. All relevant programs must be relinked. Instead of relinking, you may apply the zap to the `LCDSN= CSECT` in the appropriate user load module. Note that you must change the module name in the NAME statement to refer to your user load module when applying the zap to a user program.

Certain areas being zapped are applicable only to sites using SMS (IBM's Storage Management Subsystem). Do not change these fields if SMS is not installed and operational.

```

*   NAME: Z7504048  PRODUCT: SASC      CATEGORY: SPEC    SYSTEM: MVS
*   DATE: 02JAN04  STATUS:  DZ+UT    USAGE-ID: LIBRARY-C4048
*
*   How to change the defaults for alcunit=, space= and extend= amparms
*
*   NOTE:          APPLY TO lib/mvs/libares.a (ALL-RESIDENT
*                 RUN-TIME LIBRARY IN OBJECT FORMAT) USING THE OBJZAP
*                 UTILITY.
*
*   NOTE:          RELINK OR ZAP ANY ALL-RESIDENT PROGRAMS WHICH INCLUDE
*                 THIS MODULE.
*
*                 NOTE: THE USER MUST CODE THEIR OWN VALUES FOR PRIMARY AND
*                 SECONDARY SPACE, AVERAGE BLOCK LENGTH, NUMBER OF
*                 DIRECTORY BLOCKS, FLAGS, UNIT NAME, VOLUME, DATA CLASS,
*                 STORAGE CLASS AND MANAGEMENT CLASS. IF PADDING IS
*                 NEEDED, THESE VALUES SHOULD BE PADDED WITH BLANKS.
*
*                 IF SMS IS NOT INSTALLED AND ACTIVE, DO NOT INSTALL A DATA
*                 CLASS, STORAGE CLASS OR MANAGEMENT CLASS, AND DO NOT SET
*                 THE 08, 04, OR 02 FLAG BITS.
*
*                 XXXXXX - PRIMARY TRACK ALLOCATION
*                 YYYYYY - SECONDARY TRACK ALLOCATION
*                 LLLLLL - AVERAGE BLOCK LENGTH
*                 NNNNNN - NUMBER OF DIRECTORY BLOCKS
*                 FF      - FLAGS (80=TRACKS, 40=CYLINDERS, 20=BLOCKS
*                               08=RECORDS, 04=KILO-RECORDS,
*                               02=MEGA-RECORDS. SET THE 10 BIT TO
*                               DEFAULT RLSE=YES.)
*                 UUUUUUUUUUUUUUUU - UNIT NAME
*                 ZZZZZZZZ - VOLUME
*                 DDDDDDDD - DEFAULT DATA CLASS
*                 SSSSSSSS - DEFAULT STORAGE CLASS
*                 MMMMMMMM - DEFAULT MANAGEMENT CLASS
*
*   END
NAME      L$CIDSN  L$C$DSN=
CHECKSUM
VER       0008      0000,0A
VER       000B      0000,03
VER       000E      0003,E8
VER       0011      0000,05
VER       0014      20
VER       0015      0000,0000,0000,0000
VER       001D      0000,0000,0000
VER       0023      0000,0000,0000,0000
VER       002B      0000,0000,0000,0000
VER       0033      0000,0000,0000,0000
*
REP       0008      XXXX,XX

```

```

REP      000B      YYYY,YY
REP      000E      LLLL,LL
REP      0011      NNNN,NN
REP      0014      FF
REP      0015      UUUU,UUUU,UUUU,UUUU
REP      001D      ZZZZ,ZZZZ,ZZZZ
REP      0023      DDDD,DDDD,DDDD,DDDD
REP      002B      SSSS,SSSS,SSSS,SSSS
REP      0033      MMMM,MMMM,MMMM,MMMM
IDRDATA  Z7504048

```

Note: You do not have to replace data that you are not changing. For example, to make the default space allocation for new permanent data sets two cylinders of primary space and one cylinder of secondary space, use the following:

```

NAME L$CIDSN L$C$DSN=
VER  0008 00000A
VER  000B 000003
VER  0014 20
REP  0008 000002
REP  000B 000001
REP  0014 40

```

Customizing Data-in-Virtual Defaults on MVS

This section describes a zap that is useful for MVS sites that wish to change the default values used by the library in its algorithm for buffering Data-In-Virtual objects. The algorithm is described in detail in the section "I/O Functions" in *SAS/C Library Reference, Volume 1, Release 7.00*. You might wish to apply this zap if the default buffer size and/or maximum number of buffers are too large or too small for your site.

The following generic zap must be applied to the transient library (*prefix*.LINKLIB) using APPLYZAP (see *A Guide for the SAS/C Compiler Consultant, Release 7.00* for details on using the APPLYZAP program) in order to change the default buffer size and/or maximum number of buffers.

```
* NAME: Z7505342 PRODUCT: SASC CATEGORY: SPEC SYSTEM: MVS
* DATE: 02JAN04 STATUS: DZ+UT USAGE-ID: LIBRARY-C5342
*
* How to change the defaults for Data-In-Virtual amparms
*
* NOTE: APPLY TO SASC.LINKLIB (TRANSIENT RUN-TIME LIBRARY IN
* LOAD MODULE FORMAT) USING THE APPLYZAP
* UTILITY.
*
* THE USER MUST CODE THEIR OWN VALUES FOR THE DEFAULT
* BUFFER SIZE AND/OR THE DEFAULT MAXIMUM NUMBER OF BUFFERS.
*
* XXXXXXXX - DEFAULT NUMBER OF 4K PAGES
* YYYYYYYY - DEFAULT MAXIMUM NUMBER OF BUFFERS
*
* END
NAME L$CWPSO L$C$LDSO LINKLIB
CHECKSUM
VER 0000 0000,0040
VER 0004 0000,0004
REP 0000 XXXX,XXXX
REP 0004 YYY,YYY
IDRDATA Z7505342
```

Note: You do not have to replace data that you are not changing. For example, to make the default buffer size 1024K (256 - 4K pages), use the following:

```
NAME L$CWPSO L$C$LDSO LINKLIB
VER 0000 0000,0040
REP 0000 0000,0100
```

If you are using the all-resident library, the following generic zap must be applied to `lib\mvs\libares.a` in your installation directory using OBJZAP. All relevant programs must be relinked. Instead of relinking, you may apply the zap to the L\$C\$LDSO CSECT in the appropriate user load module. Note that you must change the module name in the NAME statement to refer to your user load module when applying the zap to a user program.

```

*   NAME: Z7505342   PRODUCT: SASC       CATEGORY: SPEC       SYSTEM: MVS
*   DATE: 02JAN04   STATUS:  DZ+UT      USAGE-ID: LIBRARY-C5342
*
*   How to change the defaults for Data-In-Virtual amparms
*
*   NOTE:          APPLY TO lib/mvs/libares.a (ALL-RESIDENT
*                 RUN-TIME LIBRARY IN OBJECT FORMAT) USING THE OBJZAP
*                 UTILITY.
*
*   NOTE:          RELINK OR ZAP ANY ALL-RESIDENT PROGRAMS WHICH INCLUDE
*                 THIS MODULE.
*
*                 THE USER MUST CODE THEIR OWN VALUES FOR THE DEFAULT
*                 BUFFER SIZE AND/OR THE DEFAULT MAXIMUM NUMBER OF BUFFERS.
*
*                 XXXXXXXX - DEFAULT NUMBER OF 4K PAGES
*                 YYYYYYYY - DEFAULT MAXIMUM NUMBER OF BUFFERS
*
*   END
NAME      L$CWPSO  L$C$LDSO
CHECKSUM
VER       0000    0000,0040
VER       0004    0000,0004
REP       0000    XXXX,XXXX
REP       0004    YYYY,YYYY
IDRDATA   Z7505342

```

Note: You do not have to replace data that you are not changing. For example, to make the default buffer size 1024K (256 - 4K pages), use the following:

```

NAME L$CWPSO L$C$LDSO
VER  0000 0000,0040
REP  0000 0000,0100

```

Customizing CICS Library Support

In “Installing the Independent SAS/C Cross-Platform Software on MVS” on page 16, the CICS transient library was copied into a disk data set. In order to run C programs in a CICS environment you must make the CICS transient library available to CICS, update the CICS PPT, PCT, and DCT tables, and modify the CICS start-up JCL.

Making the CICS Transient Library Available to CICS on MVS

There are two ways to make the transient library available to CICS. One way is to add the CICS transient library data set name to the DFHRPL concatenation of libraries in the CICS start-up JCL. Here is a sample DD statement:

```
//          DD DISP=SHR,DSNAME=prefix.CICLOAD
```

A second method is to place the CICS transient library routines in the MVS Link Pack Area (LPA) and take the necessary steps to build a CICS Application Load Table (ALT). Refer to the appropriate CICS manuals for further details on the steps necessary to implement an ALT.

Defining the CICS Transient Library Programs in the CICS PPT on MVS

If your site uses the RDO (Resource Definition On-line) facilities of CICS, you can use the RDO definitions in the CICSCSD member which was unloaded from the installation medium into the *prefix.CNTL* data set in “Installing the Independent SAS/C Cross-Platform Software on MVS” on page 16, as input to the IBM utility program DFHCSDUP to define the required PPT entries in your CSD (CICS System Definition) file. The RDO definitions as they appear in that member are as follows:

```
DELETE GROUP (SASC750)
DEFINE PROGRAM (LSHABTRH) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHALNK) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHARGP) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHCST) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHDEBUG) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHDEBUGM) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHDCICS) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHDMGR) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHDSPL) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHISPL) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHITD) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHITDB) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHKOPR) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHLDBCS) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHNCOM) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHNDBA) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHOSEQ) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHRCOM) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHSGNL) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHSIO) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHSTG) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHSTGM) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHXEVV) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWIO) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG1) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG2) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG3) LANGUAGE (ASSEMBLER) GROUP (SASC750)
```

```

DEFINE PROGRAM (LSHWNG4) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG5) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG6) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG7) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG8) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHWNG9) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE PROGRAM (LSHXEVV) LANGUAGE (ASSEMBLER) GROUP (SASC750)
DEFINE TRANSACTION (DEBUG) PROGRAM (LSHDEBUG) GROUP (SASC750)
LIST GROUP (SASC750) OBJECTS

```

After the group SASC750 is added to your CSD the group must be installed before CICS can use the definitions. Either use RDO to install the group each time CICS is initialized or add the group name to a list that appears in the GRPLIST operand of the SIT (System Initialization Table). See the appropriate CICS manuals for more information on RDO.

Alternatively, you can use the CICS PPT member to update your installation's PPT (processing program table). CICS PPT was unloaded from the installation medium into the *prefix*.CNTL data set in "Installing the Independent SAS/C Cross-Platform Software on MVS" on page 16. The PPT entries as they appear in that member are as follows:

```

DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHABTRH
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHALNK
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHARGP
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHCST
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHDEBUG
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHDEBUGM
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHDCICS
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHDMGR
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHDSPL
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHISPL
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHITD
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHITDB
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHKOPR
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHLDBCS
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHNCOM
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHNDBA
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHOSEQ
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHRCOM
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHSGNL
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHSIO
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHSTG
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHSTGM
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHXEVV
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWIO
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG1
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG2
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG3
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG4
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG5
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG6
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG7
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG8
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHXEVV
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHWNG9
DFHPPT TYPE=ENTRY, PGMLANG=ASSEMBLER, PROGRAM=LSHXEVV

```

Defining the SAS/C Debugger Front-end Transaction in the CICS PCT on MVS

If you plan to use the SAS/C Debugger on CICS then you must define the front-end transaction either in the PCT (Program Control Table) or via an RDO definition. If your site uses RDO to define transactions then you can use the transaction definition that is included with all the program definitions in the CICS_{CSD} member which was previously unloaded from the installation medium into the *prefix*.CNTL data set. The definition which appears in that file is as follows:

```
DEFINE TRANSACTION(DEBUG) PROGRAM(LSHDEBUG) GROUP(SASC750)
```

Alternatively, if your site uses tables to define transactions then you can use the CICS_{PCT} member of the *prefix*.CNTL data set to update your PCT. The PCT entry in that member appears as follows:

```
DFHPCT TYPE=ENTRY, TRANSID=DEBUG, PROGRAM=LSHDEBUG
```

Defining CICS Extrapartition Queues in the CICS DCT on MVS

You can use the CICS_{DCT} member to update your installation's DCT (Destination Control Table). The CICS_{DCT} member was unloaded from the installation medium into the *prefix*.CNTL data set in "Installing the Independent SAS/C Cross-Platform Software on MVS" on page 16. The DCT entries as they appear in that member are as follows:

```
STDOUT DFHDCTTYPE=SDSCI,          USED BY SAS/C FOR STDOUT      +
      DSCNAME=STDOUT,            +
      BLKSIZE=1370,              +
      RECSIZE=133,               +
      RECFORM=VARBLKA,          +
      TYPEFLE=OUTPUT,           +
      BUFNO=1                    +
STDERR DFHDCTTYPE=SDSCI,          USED BY SAS/C FOR STDERR      +
      DSCNAME=STDERR,           +
      BLKSIZE=1370,              +
      RECSIZE=133,               +
      RECFORM=VARBLKA,          +
      TYPEFLE=OUTPUT,           +
      BUFNO=1                    +
*
*      RECSIZE IS FOR WIDE REPORTS, PREFIXED WITH TERMID/TRANID,
*      PLUS FOUR BYTES FOR RECORD LENGTH (LLBB)
*
STGRPT DFHDCTTYPE=SDSCI,          USED BY SAS/C FOR =STORAGE REPORTS +
      DSCNAME=STGRPT,           +
      BLKSIZE=1450,              +
      RECSIZE=145,               133 + 8 + 4 +
      RECFORM=VARBLK,           +
      TYPEFLE=OUTPUT,           +
      BUFNO=1                    +
SASO   DFHDCTTYPE=EXTRA,          USED BY SAS/C FOR STDOUT      +
      DESTID=SASO,              +
      DSCNAME=STDOUT,           +
      RSL=PUBLIC                 ALLOW ANY TRANSACTION TO ACCESS
SASE   DFHDCTTYPE=EXTRA,          USED BY SAS/C FOR STDERR      +
      DESTID=SASE,              +
      DSCNAME=STDERR,           +
      RSL=PUBLIC                 ALLOW ANY TRANSACTION TO ACCESS
SASR   DFHDCTTYPE=EXTRA,          USED BY SAS/C FOR =STORAGE REPORTS +
      DESTID=SASR,              +
      DSCNAME=STGRPT,           +
      RSL=PUBLIC                 ALLOW ANY TRANSACTION TO ACCESS
```

The RECFORM, RECSIZE, and BLKSIZE parameters specified on the DFHDCT TYPE=SDSCI macros contain our recommended values. They may be freely modified at your discretion. You will have to COLD start your newly modified DCT for the changes to take effect.

The symbolic names of the extrapartition destinations (SASE, SASO, and SASR) are the default names used by the library. These names may be changed if that is required. See “Changing Default Names for CICS Modules and TD Queues” on page 47 for more information before making any changes to these names.

Allocating the CICS Extrapartition Transient Data Queues on MVS

This installation step consists of allocating data sets for the CICS extrapartition transient data queues. You can use the CICSALOC member which was unloaded from the installation medium into the *prefix*.CNTL file in “Installing the Independent SAS/C Cross-Platform Software on MVS” on page 16 to allocate these files.

The following is a listing of the job CICSALOC extracted from your installation medium:

```
//CICSALOC JOB (ACCOUNT INFORMATION) , 'PROGRAMMER' ,MSGCLASS=A,
//          MSGLEVEL=(1,1) ,TIME=(5,00)
// *
//*****
// * DOC:   ALLOCATE THE CICS EXTRAPARTITION TRANSIENT DATA FILES.
// * REFER:
// * DATE:
// * NOTE:  MODIFY THE SPACE REQUIREMENTS AS APPROPRIATE
//*****
// *
//ALLOCATE EXEC PGM=IEFBR14
//STDERR   DD DSN=&TDPREFIX..STDERR,
//          DISP=(NEW,CATLG) ,UNIT=&UNIT,
//          VOL=SER=&DISKVOL,
//          SPACE=(CYL,(2,2)) ,
//          DCB=(RECFM=VBA,LRECL=133,BLKSIZE=1370)
//STDOUT   DD DSN=&TDPREFIX..STDOUT,
//          DISP=(NEW,CATLG) ,UNIT=&UNIT,
//          VOL=SER=&DISKVOL,
//          SPACE=(CYL,(2,2)) ,
//          DCB=(RECFM=VBA,LRECL=133,BLKSIZE=1370)
//STGRPT   DD DSN=&TDPREFIX..STGRPT,
//          DISP=(NEW,CATLG) ,UNIT=&UNIT,
//          VOL=SER=&DISKVOL,
//          SPACE=(CYL,(2,2)) ,
//          DCB=(RECFM=VB,LRECL=145,BLKSIZE=1490)
```

Review space allocations for the data sets and change them as appropriate.

Modifying the CICS Start-up JCL on MVS

Once the extrapartition transient data queues have been allocated you must include DD statements that reference them in the CICS start-up JCL. The following are sample DD statements. Be sure to use the data set names that were specified in “Allocating the CICS Extrapartition Transient Data Queues on MVS” on page 46.

```
//STDERR   DD DISP=SHR,DSNAME=CICS.SASC.STDERR
//STDOUT   DD DISP=SHR,DSNAME=CICS.SASC.STDOUT
//STGRPT   DD DISP=SHR,DSNAME=CICS.SASC.STGRPT
```

Changing Default Names for CICS Modules and TD Queues

This section describes a zap that is useful for CICS sites that wish to change the default values used by the library for the names of the CICS transient load modules and/or the extrapartition transient data queues. Unless you have a very good reason for doing so, we recommend that you DO NOT change these defaults.

You might wish to apply this zap if the default extrapartition transient data queue names (SASE, SASO, and SASR) are already in use at your site.

Note: If you change the default three character prefix for the transient data queue names and this is not done during product installation, then all relevant user application programs must be relinked. Instead of relinking, you may apply the zap to the L\$C\$PFXH CSECT in the appropriate user load module. Note that you must change the module name in the NAME statement to refer to your user load module when applying the zap to a user program.

Two CICS transient library programs must also be zapped: LSHSTG and LSHSTGM. These programs are invoked by the library to support the =storage run-time option. Similarly, you might wish to apply this zap if the default CICS transient library program names (which, by default, all start with the three character prefix LSH) are already in use at your site.

Note: If you decide to change the three character load module name prefix you must also modify the PPT entries (see “Defining the CICS Transient Library Programs in the CICS PPT on MVS” on page 43). In addition, the load modules in *prefix*.CICSLOAD must be renamed as well. It is important that the alias names for these members be preserved, otherwise it will be impossible to apply zaps and other maintenance that you receive from the Institute.

The following generic zap may be applied to the CICS resident libraries (lib\cics\libc.a and lib\cicsvse\libc.a in your installation directory) using OBJZAP in order to change the default transient data queue and/or transient load module names.

```

*   NAME: Z7500158   PRODUCT: SASC       CATEGORY: SPEC   SYSTEM: MVS
*   DATE: 02JAN04   STATUS:  DZ+UT      USAGE-ID: LIBRARY-C0158
*
*   How to change default CICS TD queue and transient library names
*
*   NOTE:           APPLY TO SASC.CICSOBJ(OBJECT MODULE FORM CICS RESIDENT
*                   LIBRARY), SASC.CICSLIB(LOAD MODULE FORM CICS RESIDENT
*                   LIBRARY), SASC.VSEOBJ(OBJECT MODULE FORM CICS/VSE
*                   RESIDENT LIBRARY), AND SASC.CICSLOAD (LOAD MODULE FORM
*                   CICS TRANSIENT LIBRARY) USING THE APPLYZAP UTILITY.
*
*                   THE USER MUST CODE THEIR OWN VALUES FOR THE
*                   DEFAULT THREE CHARACTER PREFIX OF THE CICS
*                   EXTRAPARTITION TRANSIENT DATA QUEUE NAMES AND/OR
*                   THE CICS TRANSIENT LIBRARY PROGRAM NAMES.
*
*                   XXXX,XX - DEFAULT EXTRAPARTITION TD QUEUE NAME PREFIX
*                   YYYY,YY - DEFAULT TRANSIENT LIBRARY PROGRAM NAME PREFIX
*
*   END
NAME      L$C$PFXH L$C$PFXH      CICSOBJ CICSLIB CICSVSEO
CHECKSUM
VER       0000      E2C1,E200
VER       0004      D3E2,C800
REP       0000      XXXX,XX
REP       0004      YYYY,YY
ALIAS     L$C$PFX
CHECKSUM
IDRDATA   Z7500158
NAME      L$DSTG   L$C$PFXH      CICSLOAD

```

```

VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX
REP      0004      YYYY,YY
IDRDATA  Z7500158
NAME     L$DSTGM  L$C$PFXH      CICSLOAD
VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX
REP      0004      YYYY,YY
IDRDATA  Z7500158
NAME     L$CDBUG  L$C$PFXH      CICSLOAD
VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX
REP      0004      YYYY,YY
IDRDATA  Z7500158

```

The following generic zap may be applied to the CICS transient library.

```

*   NAME: Z7500158  PRODUCT: SASC      CATEGORY: SPEC      SYSTEM: MVS
*   DATE: 02JAN04  STATUS:  DZ+UT      USAGE-ID: LIBRARY-C0158
*
*   How to change default CICS TD queue and transient library names
*
*   NOTE:          APPLY TO SASC.CICSOBJ(OBJECT MODULE FORM CICS RESIDENT
*                  LIBRARY), SASC.CICSLIB(LOAD MODULE FORM CICS RESIDENT
*                  LIBRARY), SASC.VSEOBJ(OBJECT MODULE FORM CICS/VSE
*                  RESIDENT LIBRARY), AND SASC.CICSLOAD (LOAD MODULE FORM
*                  CICS TRANSIENT LIBRARY) USING THE APPLYZAP UTILITY.
*
*                  THE USER MUST CODE THEIR OWN VALUES FOR THE
*                  DEFAULT THREE CHARACTER PREFIX OF THE CICS
*                  EXTRAPARTITION TRANSIENT DATA QUEUE NAMES AND/OR
*                  THE CICS TRANSIENT LIBRARY PROGRAM NAMES.
*
*                  XXXX,XX - DEFAULT EXTRAPARTITION TD QUEUE NAME PREFIX
*                  YYYY,YY - DEFAULT TRANSIENT LIBRARY PROGRAM NAME PREFIX
*
*   END
NAME     L$DSTG   L$C$PFXH      CICSOBJ CICSLIB CICSVSEO
VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX
REP      0004      YYYY,YY
ALIAS    L$C$PFX
IDRDATA  Z7500158
NAME     L$DSTG   L$C$PFXH      CICSLOAD
VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX
REP      0004      YYYY,YY
NAME     L$DSTGM  L$C$PFXH      CICSLOAD
VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX

```

```
REP      0004      YYYY,YY
IDRDATA  Z7500158
NAME     L$CDEBUG  L$C$PFXH      CICSLOAD
VER      0000      E2C1,E200
VER      0004      D3E2,C800
REP      0000      XXXX,XX
REP      0004      YYYY,YY
CHECKSUM
IDRDATA  Z7500158
```

The following generic zap may be applied to the CICS resident library.

```

*   NAME: Z7500158  PRODUCT: SASC      CATEGORY: SPEC   SYSTEM: CMS/ESA
*   DATE: 02JAN04  STATUS:  DZ+UT    USAGE-ID: LIBRARY-C0158
*
*   How to change default CICS TD queue and transient library names
*
*   NOTE:          APPLY TO SASC.CICSOBJ(OBJECT MODULE FORM CICS RESIDENT
*                 LC370VSE TXTLIB(CICS/VSE RESIDENT LIBRARY) USING
*                 THE APPLYZAP UTILITY.
*
*                 THE USER MUST CODE THEIR OWN VALUES FOR THE
*                 DEFAULT THREE CHARACTER PREFIX OF THE CICS
*                 EXTRAPARTITION TRANSIENT DATA QUEUE NAMES AND/OR
*                 THE CICS TRANSIENT LIBRARY PROGRAM NAMES.
*
*                 EEEE,EE - DEFAULT EXTRAPARTITION TD QUEUE NAME PREFIX
*                 FFFF,FF - DEFAULT TRANSIENT LIBRARY PROGRAM NAME PREFIX
*
*   END
NAME      L$C$PFXH L$C$PFXH      CICSOBJ CICSVSEO
VER       0000     E2C1,E200
VER       0004     D3E2,C800
REP       0000     EEEE,EE
REP       0004     FFFF,FF
LOG Z7500158 ZAPLOG L$C$PFXH How to change default CICS TD queue

```

Customizing TCP/IP Defaults on MVS

Specifying the High-Level Prefix for TCP/IP Data Sets

The MVS file system differs from the UNIX file structure, and local security or organization considerations can affect how data sets are named. "Network Administration" of *SAS/C Library Reference, Volume 2* describes how the SAS/C Socket library searches for these data sets. The following zap is used to change the default TCP/IP prefix as described in "Network Administration." Before using this zap please read and understand "Network Administration."

The following generic zap may be applied to the transient library (*prefix*.LINKLIB) using APPLYZAP (see *A Guide for the SAS/C Compiler Consultant* for details on using the APPLYZAP program).

```

*   NAME: Z7504151   PRODUCT: SASC       CATEGORY: SPEC       SYSTEM: MVS
*   DATE: 02JAN04   STATUS:  DZ+UT      USAGE-ID: LIBRARY-C4151
*
*   Specifying the high-level prefix for TCP/IP data sets
*
*   NOTE:           APPLY TO SASC.LINKLIB (LOAD MODULE FORM TRANSIENT
*                   LIBRARY) AND SASC.ARESOBJ (ALL-RESIDENT OBJECT FORM
*                   RUNTIME LIBRARY) USING THE APPLYZAP UTILITY.
*
*                   THIS ZAP SHOULD BE USED ONLY AS A LAST RESORT. SEVERAL
*                   OTHER METHODS ARE AVAILABLE FOR FINDING TCP/IP
*                   CONFIGURATION DATA SETS. SEE SAS/C LIBRARY REFERENCE,
*                   VOLUME 2, CHAPTER 17 FOR DETAILS.
*
*                   NOTE: CODE YOUR OWN VALUE FOR THE PREFIX.
*                   UP TO 26 BYTES MAY BE SPECIFIED (REPLACING THE
*                   "XX"'s IN THE ZAP BELOW). ALL UNUSED BYTES MUST
*                   BE CHANGED TO "00". THE LAST BYTE OF THE REP
*                   SECTION MUST REMAIN "00".
*
*                   EXAMPLE: CHANGE PREFIX TO "SYS2.TCPIP2"
*                   REP 0000     E2E8,E2F2
*                   REP 0004     4BE3,C3D7
*                   REP 0008     C9D7,E5F2
*                   REP 000C     0000,0000
*                   REP 0010     0000,0000
*                   REP 0014     0000,0000
*                   REP 0018     0000,00
*
*   END
NAME      L$CNDBA  L$CSKFN$      LINKLIB
CHECKSUM
VER       0000      E3C3,D7C9
VER       0004      D700,0000
VER       0008      0000,0000
VER       000C      0000,0000
VER       0010      0000,0000
VER       0014      0000,0000
VER       0018      0000,00
*
REP       0000      XXXX,XXXX
REP       0004      XXXX,XXXX
REP       0008      XXXX,XXXX
REP       000C      XXXX,XXXX
REP       0010      XXXX,XXXX

```

```

REP      0014      XXXX,XXXX
REP      0018      XXXX,XX
IDRDATA  Z7504151

```

If you are using the all-resident library, the following generic zap may be applied to `lib\mvs\libares.a` in your installation directory using OBJZAP.

```

*   NAME: Z7504151  PRODUCT: SASC      CATEGORY: SPEC      SYSTEM: MVS
*   DATE: 02JAN04  STATUS:  DZ+UT      USAGE-ID: LIBRARY-C4151
*
*   Specifying the high-level prefix for TCP/IP data sets
*
*   NOTE:          APPLY TO SASC.LINKLIB (LOAD MODULE FORM TRANSIENT
*                 LIBRARY) AND SASC.ARESOBJ(ALL-RESIDENT OBJECT FORM
*                 RUNTIME LIBRARY) USING THE APPLYZAP UTILITY.
*
*                 THIS ZAP SHOULD BE USED ONLY AS A LAST RESORT. SEVERAL
*                 OTHER METHODS ARE AVAILABLE FOR FINDING TCP/IP
*                 CONFIGURATION DATA SETS. SEE SAS/C LIBRARY REFERENCE,
*                 VOLUME 2, CHAPTER 17 FOR DETAILS.
*
*                 NOTE: CODE YOUR OWN VALUE FOR THE PREFIX.
*                 UP TO 26 BYTES MAY BE SPECIFIED (REPLACING THE
*                 "XX"'s IN THE ZAP BELOW). ALL UNUSED BYTES MUST
*                 BE CHANGED TO "00". THE LAST BYTE OF THE REP
*                 SECTION MUST REMAIN "00".
*
*                 EXAMPLE: CHANGE PREFIX TO "SYS2.TCPIP2"
*
*                 REP 0000      E2E8,E2F2
*                 REP 0004      4BE3,C3D7
*                 REP 0008      C9D7,E5F2
*                 REP 000C      0000,0000
*                 REP 0010      0000,0000
*                 REP 0014      0000,0000
*                 REP 0018      0000,00
*
*   END
NAME     L$CNDBA  L$CSKFN$
CHECKSUM
VER      0000      E3C3,D7C9
VER      0004      D700,0000
VER      0008      0000,0000
VER      000C      0000,0000
VER      0010      0000,0000
VER      0014      0000,0000
VER      0018      0000,00
*
REP      0000      XXXX,XXXX
REP      0004      XXXX,XXXX
REP      0008      XXXX,XXXX
REP      000C      XXXX,XXXX
REP      0010      XXXX,XXXX
REP      0014      XXXX,XXXX
REP      0018      XXXX,XX
IDRDATA  Z7504151

```

Configuring the SAS/C Library to use the IBM z/OS Name Resolver

Starting with z/OS V1R2, IBM introduced a new DNS Name Resolver designed to replace all previous IBM Name Resolvers. This new Name Resolver is based on the latest version of the BIND DNS Name Resolver, version 9. Instead of a Name Resolver running in the application's Address Space, the new IBM Name resolver runs in a separate Address Space and is executed by a Started Task. In order to access this new Name Resolver, IBM also introduced two new UNIX System Services (USS) Assembler Callable Service calls for the name resolver functions, `gethostbyname(BPX1GHN)` and `gethostbyaddr(BPX1GHA)`.

By default the SAS/C Library will use the SAS/C Name Resolver when `gethostbyname()` or `gethostbyaddr()` is called. To enable the SAS/C Library to use the new USS calls for these functions and therefore use IBM's new z/OS Name Resolver the following zap needs to be applied.

The following zap may be applied to the transient library(prefix.LINKLIB) using `APPLYZAP` (see *A Guide for the SAS/C Compiler Consultant* for details on using the `APPLYZAP` program).

```
*****
*   NAME: Z7502143   PRODUCT: SASC       CATEGORY: SPEC   SYSTEM: MVS
*   DATE: 30MAY03   STATUS:  DZ+UT      USAGE-ID: LIBRARY-C2143
*
*   7.50C Transient Library zap to enable using IBM z/OS 1.2 Resolver
*
*   NOTE:           APPLY TO SASC.LINKLIB (TRANSIENT RUN-TIME LIBRARY IN
*                   LOAD MODULE FORMAT), USING THE APPLYZAP UTILITY.
*   -----
*   END
NAME      L$CNCOE  L$CNCOE=  LINKLIB
CHECKSUM
VER       000008  0000,0000
REP      000008  0000,0001
CHECKSUM 00090800
IDRDATA  Z7502143
```

If you are using the all-resident library, the following zap may be applied to `lib/mvs/libares.a` in your installation directory using `OBJZAP`.

```
*****
*   NAME: Z7502143   PRODUCT: SASC       CATEGORY: SPEC   SYSTEM: MVS
*   DATE: 30MAY03   STATUS:  DZ+UT      USAGE-ID: LIBRARY-C2143
*
*   7.50C Transient Library zap to enable using IBM z/OS 1.2 Resolver
*
*   NOTE:           APPLY TO lib/mvs/libares.a (ALL-RESIDENT
*                   RUN-TIME LIBRARY IN OBJECT FORMAT) USING THE OBJZAP
*                   UTILITY.
*
*   NOTE:           RELINK OR ZAP ANY ALL-RESIDENT PROGRAMS WHICH INCLUDE
*                   THIS MODULE.
*   -----
*   END
NAME      L$CNCOE  L$CNCOE=
CHECKSUM
VER       000008  0000,0000
REP      000008  0000,0001
CHECKSUM 00090800
IDRDATA  Z7502143
```


Appendix A Utility Command Formats

This appendix describes the utility command formats you will need when installing the SAS/C Cross-Platform software on a PC. All of these commands are described in man pages provided with the product.

dset

The `dset` utility command displays the license information for a SAS/C Cross-Platform License product. The license data is maintained in a file with the executables associated with the product, and is updated using the `zap` utility supplied with the SAS/C Cross-Platform products, along with information supplied by SAS Institute. For additional information on the `zap` utility, see “zap” on page 60.

The format for the `dset` utility command is:

```
dset license_data_file
```

The following example displays the license information associated with the SAS/C Cross-Platform Compiler.

```
dset sascc_set.dat
```

Variable	Definition
<i>sascc_set.dat</i>	the license data for SAS/C Cross-Platform Compiler
<i>sascd_set.dat</i>	the license data for SAS/C Cross-Platform C++ Development System

objzap

You can run the `objzap` utility on Windows 95, Windows 98, Windows NT, Windows 2000, and Windows XP to make changes to object modules (since several parts of the products are supplied in object form). This utility operates on object modules that may or may not be members of ar370 archives, allowing users to

- examine data and instructions
- change data and instructions
- dump any control section (CSECT) of the object.

The format for the `objzap` utility command is:

```
objzap [-aar370_archive_name | -lobject_file_location] < zapname
```

The following options are valid for `objzap`

Option	Description
-a	Allows you to provide an archive filename (only for non-370 hosts).
-ca	Computes a checksum the same way <code>applyzap</code> does.
-i	Allows you to name the input zap file
-l	Allows you to specify the SYSLIB name or directory (depending on host) that contains the member to be zapped.
-p	Use <code>ARGV[0]</code> to determine the path to AR370.
-s	Sets <code>seqfile=1</code> .
-u	Sets <code>upper=1</code> . Ensure that the output line is <256 bytes. If you are running <code>objzap</code> on a 370 host, the output appears in uppercase.

The `objzap` utility uses several control statements that specify the operations to be performed. This section includes a brief synopsis of the control statements and detailed syntax rules, followed by an example of a PC `objzap` command line.

Control Statements

This section provides an introduction to the organization of the control statements that are used to run the `objzap` utility. Following this introduction is a detailed description of each statement and a brief discussion of the syntax rules.

Organization

Control statements for the `objzap` utility are listed below. (Control statements are not case-sensitive.)

- NAME object csect
- VER offset expected-content
- REP offset replacement-data
- CHECKSUM value
- IDRDATA xxxxxxxxx
- DUMP object csect

Each set of control statements begins with a NAME statement. The VER, REP, IDRDATA, and DUMP statements following the NAME statement apply to the object module and CSECT that the NAME statement identifies.

The VER, REP, IDRDATA, and DUMP statements can appear in any order, but it is better to code all VER statements before the first REP statement to ensure that the data are verified before anything is replaced. The DUMP statement must follow the NAME statement; the CHECKSUM statement can appear anywhere in the sequence of statements. When a new NAME statement occurs, it defines a new CSECT (possibly a new object) as the object of succeeding VER, REP, IDRDATA, and DUMP statements.

Statement Descriptions

□ NAME object csect

NAME gives the identity of the object module containing the CSECT (control section) that all succeeding control statements operate on. There is no restriction on the number of NAME statements that can appear.

□ VER offset expected-content

VER compares the contents of a location in the CSECT (offset) with the expected content (expected-content) supplied by the user. If the two fields do not compare as equal, the VER operation fails and a formatted dump is provided for the csect. No further REP operations are performed until the next NAME statement occurs.

- offset

contains the hexadecimal displacement of the data in csect. The VER operation fails if offset is outside the boundaries for the CSECT specified by the NAME statement (offset can be an even or odd number of digits).

- expected-content

contains the hexadecimal representation of the data expected at the offset in csect. The data must be expressed as an even number of two hexadecimal values, for example, 4741D175 (or, with commas: 4741,D175). If commas are used to separate data, the number of digits between them must also be even. Blanks cannot be used to separate digits. A blank ends expected-content; any data following a blank are treated as commands and ignored. For data that will not fit in one 80-byte VER statement, a second statement must be used.

□ REP offset replacement-data

REP changes the data in a CSECT defined in the NAME statement. It replaces the data specified at offset with the data specified in replacement-data. The REP operation fails if offset is outside the boundaries for the CSECT specified in the NAME statement. The formats of the arguments to REP follow the same rules as the formats of the VER arguments. The VER operations should always be performed to determine what will be changed with the REP function. If more than one VER and REP operation is to be performed on a CSECT, statements should be ordered so that all VER statements appear before all REP statements. The reason for this order is to ensure that no REP statement is performed if any VER operation fails. When REP is successful, the old data are printed out and the IDR information in the object module is automatically updated. (See the IDRDATA discussion in this list.)

□ CHECKSUM <value>

The CHECKSUM statement performs one of two tasks, depending on whether the optional value argument is used. The value argument must be eight hexadecimal digits and can not contain commas. If the value argument is present, the statement compares the number specified in value and the accumulated CHECKSUM. The checking is done when `objzap` reads the control statements. If the accumulated CHECKSUM and the number specified by value are not equal, no processing is done. If the value argument is not present, the accumulated CHECKSUM is printed in hexadecimal. The accumulated CHECKSUM starts at zero and is reset to zero by each CHECKSUM statement. Only the offsets and data from REP statements are used in accumulating the CHECKSUM. The CHECKSUM statement guards effectively against typographical errors in making a change. All fixes to object modules supplied by SAS Institute Inc. contain

CHECKSUM statements, and you should not remove them. You can use a comment on the CHECKSUM statement if the CHECKSUM statement contains a value argument. (See the discussion of comments later in this chapter.)

IDRDATA xxxxxxxxxx

The IDRDATA control statement is executed only if at least one REP operation is executed. IDRDATA puts a maximum of 10 bytes of user data into bytes 1 to 10 at the location of the second IDR item (on the END card of the object that contains the CSECT). xxxxxxxxxx is the ten bytes of data, expressed without embedded blanks. Blanks are added at the right if less than 10 bytes are specified. IDRDATA is useful for tracing what zaps have been applied. Note that when a REP operation is performed, the following occurs:

- If no IDRDATA operation is specified, “UNKNOWN” is inserted in bytes 1-9 of the second IDR field.
- If an error later occurs, the string ER is inserted in bytes 11-12 on the END card. Otherwise, bytes 11-12 are blanked out.
- The Julian data (yyddd) is inserted in bytes 15-19 on the END card.

If there is more than one CSECT in the object on which a REP is performed, the IDRDATA from the last CSECT is used. If there was no previous IDR statement, “UNKNOWN” is used. It is customary for zaps supplied by SAS Institute to contain IDR statements with the release and zap numbers as identification.

DUMP <object > <csect>

DUMP or DUMPT dumps the CSECT identified in the NAME statement. The csect and object arguments are optional, and if specified, they must be the same as in the NAME statement. The output of the DUMP command is in hexadecimal format. When this command is used, the IDR data from the most recent zap (successful or unsuccessful) are printed, in addition to the contents of the CSECT. Note that a NAME statement must always come before the DUMP statement.

Comments

Comment statements: You can use comments in the obj zap command stream. Comment statements must be in the form * comment. (The number of comments is not limited.) The obj zap utility writes the comment statements to standard output. Comments included on control statements: You can also include comments on control statements, other than DUMP or DUMPT statements. In a control statement, place the comment after the last argument that the statement requires. Precede the comment with a blank. You do not need an asterisk (*) to indicate a comment on a control statement. If no arguments are present, you cannot use a comment on the control statement.

Detailed Syntax Rules

The detailed syntax rules follow:

- An obj zap operation name must be specified before any arguments in a statement.
- The statement can be entered starting in any column. Control statements can be up to 80 bytes long. (Information beyond column 72 is ignored.)
- Several blanks can separate the obj zap operation name and its first argument, but there must be at least one blank. Similarly, one or more blanks must separate arguments in the statement.
- Commas can be used in data fields other than offsets and the CHECKSUM value. Blanks are not allowed inside data fields. a blank terminates a data field.
- Values in the expected-content field (VER statement) and the replacement- data field (REP statement) must be expressed as an even number of hexadecimal digits.
- Comment statements are specified by an asterisk. Comments can also be used on control statements other than the DUMP statement. In a control statement, a space separates the comment from the last argument that the statement takes. No asterisk (*) is needed for comments on control statements.
- Control statements are not case-sensitive.

Examples

Zap an object specified in the NAME statement of the file myzap using the control statements found there:

```
objzap < myzap
```

Zap an object that is a member of myarchive.a using the control statements in myzap2.

```
objzap -amyarchive.a < myzap2
```

Zap an object that is located in the directory /my/objects using the control statements in myzap3.

```
objzap -l\my\objects < myzap3
```

zap

The zap utility command reads a zap file and applies the changes to the image. A zap file contains a series of lines which are either comments or replace commands, indicating replacement bytes. A comment line is one beginning with an asterisk (*). The entire line is taken as a comment and ignored. Replace commands begin with the keyword REP followed by the offset of where the replacement is to begin, followed by the bytes to replace. The offsets and the replacement bytes are specified as hexadecimal digits.

The format for the zap utility command is:

```
zap zap-file image-file
```

The following option is valid for zap

Option	Description
<i>zap-file</i>	The file that contains the zap information.
<i>image-file</i>	The file which is altered by the application of the zap.

An example zap file could look like:

```
*
* An example zap file
*
* The following line indicates the 8 bytes at
* offset 100 hexadecimal (256 decimal) are to be replaced.
REP 100 B2FB83431858A488
*
* The next line indicates that the next 8 bytes are to be
* replaced, because the offset is 108 (264 decimal.)
REP 108 DB581F2D03000000
```

There is no checking of the bytes which were previously in the image-file. The image file is rewritten.

In the following example, the input file zap file, myzap, is scanned for replacement lines, replacing the specified bytes in the image file, prog.

```
zap myzap prog
```

The zap utility is normally used to update a site's license information.

Appendix B Reference Publications

With Release 7.50, we are continuing our move to online documentation. You will find these books on the CD-ROM titled *SAS/C OnlineDoc™, Release 7.00*, which is included with the SAS/C software distribution package:

SAS/C® 7.50 Changes and Enhancements

Introducing SAS/C® Software, Release 7.00

SAS/C® Software: Changes and Enhancements, Release 7.00

SAS/C® CICS User's Guide, Release 7.00

SAS/C® Compiler and Library User's Guide, Release 7.00

SAS/C® Cross-Platform Compiler and C++ Development System User's Guide, Release 7.00

SAS/C® C++ Development System User's Guide, Release 7.00

SAS/C® Debugger User's Guide and Reference, Release 7.00

SAS/C® Software Diagnostic Messages, Release 7.00

A Guide for the SAS/C® Compiler Consultant, Release 7.00

SAS/C® Library Reference, Volume 1, Release 7.00

SAS/C® Library Reference, Volume 2, Release 7.00

Standard C++ Libraries, Rogue Wave titles:

Standard C++ Library General User's Guide - OEM Edition

Standard C++ Library Class Reference

Tools.h++ 8.0 Class Reference

Tools.h++ 8.0 User's Guide

For additional information refer to these documents:

SAS/C® Compiler Full-Screen Support Library User's Guide, Second Edition, Release 5.01

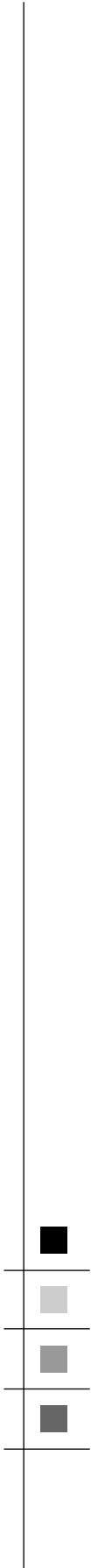
SAS/C® Compiler Interlanguage Communication Feature User's Guide, Release 4.00

SAS/C® Compiler and Library Quick Reference Guide, First Edition, Release 6.00

SAS® Technical Report C-113 SAS/C® Connectivity Support Library, Release 1.00

A Guide for the SAS/C® Compiler Consultant, Release 7.00

SAS® Technical Report C-115 The Generalized Operating System Interface for the SAS/C® Compiler Run-Time System, Release 5.50



www.sas.com/service/techsup/intro.html

SAS is the world leader in e-intelligence software and services. SAS partners with customers to turn raw data, including the vast quantity generated by e-business, into usable knowledge, and makes it available to decision-makers across the enterprise. SAS enables informed business decisions, helping its customers gain competitive advantage in their markets.