Altair SLC Migration Guide for z/OS

Version: 2023.5

www.altair.com

Contents

Introduction

This guide will help guide you install the Altair SLC on the z/OS platform. It also contains sections that describe:

- How to use Altair SLC
- What to do if you have any existing SAS language programs and any data associated with them.

Overview

To migrate to Altair SLC on z/OS, you need to consider how you will migrate:

- JCL
- Data
- SAS language Programs
- MXG
- Stored macro librariesand format catalogs
- Format catalogs

You should also take into account performance issues.

JCL migration

You need to make a small number of changes to your JCL when you migrate to Altair SLC software.

WPSHOST

The executable program name in the Altair SLC load library is `WPSHOST`. In general, this is called from `WPSPROC`, but if you need to call Altair SLC directly, you should use a statement similar to:

```
//<STEPNAME> EXEC PGM=WPSHOST
```

WPSPROC

The Altair SLC `CNTL` library provided with the installation contains a member called `WPSPROC`. This member contains a JCL procedure called `WPSPROC` that sets up any required DD statements for a default configuration. If your site uses a common `SASPROC` JCL procedure, modify jobs to use Altair SLC by changing:

- The `PROCLIB JCLLIB` statement (if used) to point to wherever the `WPSPROC` JCL is held, and
- `//<STEPNAME> EXEC SASPROC` to `//<STEPNAME> EXEC WPSPROC`

Required DDNAMEs

If your site does not use a common JCL procedure, you need to make more wide-ranging changes, as each JCL job needs to be modified. The DD statements need to be modified, added or removed to suit Altair SLC. These changes vary from site to site depending on usage.

DD statements for the following DDNAMEs are essential for initialising Altair SLC:

`CEEDUMP`	Output of language environment abnormal termination dump files
`CEEOPTS`	Optional overriding Language Environment (LE) options.
`CEERPT`	Output of language environment CEEreport information
`CONFIG`	The location where default run-time options are specified.
`DFSPARMS`	Basic host sort parameters.
`DSNAOINI`	Default DB2 connection configuration.

DSNTRACE	For DB2 trace output (can be very large)
NEWS	Default 'welcome' message for placing at the top of SASLOG output, but only if it is specified in the CONFIG file.
SASHELP	A Altair SLC-formatted data library created by the installation of Altair SLC. Holds support files such as translation tables for language encoding.
SASLOG	Program source listing, progress notes, diagnostics.
SASLIST	Listing output from various procedures, as used in the SAS language program.
SETINIT	The name of the file holding the applied license, once Altair SLC has been installed and licensed.
SORTMSGS	Output generated by call to the host SORT utility, depending on the parameters specified in the DFSPARMS file
STEPLIB	Gives the name of the load library where the WPSHOST program is held, unless the library is specified in the LNKLST.
SYSOUT	For language environment output error output stream.
	Various SYSOUT files are required. These can be omitted, re-directed to disk files, or ignored using the *DUMMY* parameter.
SYSPRINT	For language environment standard output stream
WORK	A temporary sequential dataset used by Altair SLC to hold the WORK library and all other transient data while executing a SAS language program.
WPSAOINI	Altair SLC-specific DB2 configuration. Only accessed if required.
WPSFONTS	Location of default TrueType distributed with Altair SLC.
WPSTRACE	Output of Altair SLC trace service output

Program migration

Some syntax in the SAS language is not supported by Altair SLC. You can find out which of your programs contain unsupported SAS language syntax, in one of the following ways:

- Manual inspection
- Automated static analysis
- Dynamic analysis

Manual inspection

You can manually check the language syntax elements used in your programs against those listed in the *Altair SLC Reference for Language Elements* supplied with Altair SLC. Although this document is updated each time a new version of Altair SLC is released, we cannot guarantee that the list of language syntax elements is exhaustive.

Automated static analysis

Altair SLC provides a tool that can analyse programs and generate reports of the SAS language used by the programs. Two reports are produced:

- A usage report, showing all usage of syntax provided by the SAS language.
- A compatibility report, showing those syntax elements of the SAS language that are not supported by Altair SLC.

The analyser can process a single file or a complete directory of files including files in subdirectories.

The tool runs on Altair Analytics Workbench on PC. There is no z/OS-based equivalent. To use it with files produced for the z/OS environment, you must first download the files to the workstation environment. The files must be downloaded in ASCII text format.

Preparing and downloading files can be done in number of ways. For example:

- Convert the mainframe files to XMIT format and transfer them to the PC in binary mode using, for example, FTP or the TN3270 file transfer utility. The XMIT files can then be unpacked using a PC-based tool such as XMIT Manager. Because the XMIT Manager cannot handle PDSE files, these should be copied to a PDS (Partitioned Dataset) before being transferred.
- Download the mainframe files using FTP in text mode - that is, convert the files to ASCII from EBCDIC during the download process.

- Copy the source files into a directory in USS HFS/ZFS space. Use pax to transcode these files into ASCII and combine them into a single entity for transferring to the workstation, where they can be unpackaged and analysed.

An extra consideration with any of these solutions is with SAS language statements held in files of 80-byte card-images. It is frequently the case that such records have sequence numbers in character positions 73-80. The code analyser cannot handle such sequence numbers with any certainty, so they must be removed prior to transferring the data to a PC.

The SAS language statement `PROC PDSCOPY INDD=<inddname> OUTDD=<outddname>;` can be used to accomplish this. The value specifed to the `OUTDD` option must point to a file or PDS that has `LRECL=72` defined. This way, each language statement is truncated to 72 characters in length, and so any potential sequence numbers are removed.

Contact World Programming for details of obtaining a workstation version of Altair SLC suitable for performing static code analysis, and for advice on file transfer.

Dynamic analysis

World Programming can analyze a live, production z/OS system to discover how it uses syntax from the SAS language. World Programming uses specialist tools and services to do this. The analysis can provide invaluable insight for larger sites before and during any migration project to Altair SLC.

Migrating data

The contents of existing data libraries produced using the SAS System on the z/OS platform, and stored on various sources, including DASD and tape media, can be migrated to Altair SLC. Various methods can be used to achieve this.

Altair SLC can read and process disk-based SAS System data libraries (SAS V6 and later) directly using the SASDASD library engine, but cannot write to such libraries. Any SAS System data libraries that need to be written to or otherwise updated must be migrated before being processed by Altair SLC. You can do this using the COPY procedure. Altair SLC includes full support for standard physical sequential multi-volume data libraries.

Altair SLC cannot read catalogs held in a SAS System data library. Such entities must be rebuilt from the original source in Altair SLC.

If you are migrating custom formats, and the original FORMAT procedure source is unavailable, the data can be migrated by using PROC FORMAT CNTLOUT=xxx; statement in the SAS System to produce a file that can be input by Altair SLC using the PROC FORMAT CNTLIN=xxx; statement.

The Altair SLC SASSEQ engine can be used to read standard SAS System tape-based data libraries created with the V7TAPE, V8TAPE and V9TAPE data library engines. Altair SLC will also write data to such libraries, using the SASTAPE engine. Catalogs held in such libraries will be skipped unless deliberately rebuilt using Altair SLC

Migrating data from SAS System DASD data libraries

The contents of a SAS System DASD data library can be migrated to Altair SLC in one step, using the Altair SLC SASDASD engine to copy data into Altair SLC DASD data library format.

Example migration jobs

Example batch job JCL and macros in the SAS language are provided in three members in the supplied <WPSPFX>.CNT Llibrary that was created by the Altair SLC installation procedure.:

- The @SAS2WPS member shows a simple disk-to-disk migration method. The program code is held as part of the member.

- The @SAS2WPX member shows a more complex method, which uses an intermediate tape-based file to provide a backup copy of the original SAS System data library. This member also features a comparison of the two sets of data after migration is complete. The program code is held as part of the member, but it uses macros that are defined in `<WPSPFX>.CNTL` (XMIGRATE). Note that XMIGRATE is not a batch job to be submitted; it simply provides code that is used in the @SAS2WPX job.
- SAS System format catalogs can be migrated using a `PROC FORMAT CNTLOUT` and `CNTLIN` sequence. Member @FMT2WPS provides an example JCL sequence to accomplish such a migration. `<WPSPFX>.CNTL` (XMIGRATE) is used to provide code.
- Also provided is member `<WPSPFX>.CNTL` (@COMPARE). This is an example job that is intended to compare the contents of two data libraries before and after migration.

Migrating stored macro libraries

Stored macro libraries generated by the SAS System must be migrated to Altair SLC before use.

If the original source program that created the stored macros is available, run the program in Altair SLC after making suitable changes to the appropriate DD statements. Altair SLC will not write to a SAS System Library, so new Altair SLC-based libraries must be allocated and used.

If the original source program that created the stored macros is not available, the stored macros might have been generated with the / STORE SOURCE options specified. If so, the macro source can be retrieved using the `%COPY <macroname>/SOURCE;` statement. This results in the original source code being replicated in the `SASLOG`. You can then use this source code in Altair SLC to generate a stored macro library.

If the originating source code is not available, contact Altair for further assistance.

Migrating format catalogs

Altair SLC will not process `FORMAT`s held in a catalog that was created using the SAS System. The entire catalog must be re-created by inputting the original source code to Altair SLC.

If the original source code is not available, then a `FORMAT` catalog that is usable by Altair SLC can be created by:

1. Generating an intermediate file by using a `PROC FORMAT CNTLOUT=xxx;` statement, then:
2. Using this intermediate file as input to Altair SLC using a `PROC FORMAT CNTLIN=xxx;` statement.

This pair of operations is provided for in the skeleton JCL provided in `<WPSPFX>.CNTL` (@FMT2WPS)

MXG migration considerations

MXG installations vary considerably in size and complexity, so it is difficult to recommend a one size fits all migration procedure. There are, however, a few common considerations that must be catered for.

Altair SLC version

We always recommend that clients should use the latest General Availability (GA) version of Altair SLC. In certain circumstances, the solution to an observed problem may be that the client needs to upgrade to the latest maintenance version, for which special instructions will be issued. As a rule, the latest GA version(s) are always immediately available on the WPL download site.

Upgrading to a later version of Altair SLC is a procedure that requires a complete re-installation each time. As a result, the older versions of the software may be retained on the system, to be used for comparison purposes.

MXG version

Upgrade versions of MXG are made available on a (approximately) monthly basis. The monthly issue of the software is known as a minor release, and there is an annual 'major release' as well. All changes made to MXG code are documented in the CHANGES member of the distributed source library.

Altair SLC will work with MXG 25.11 or later. As a general rule, you should aim to be using the latest release of MXG at all times. As with Altair SLC, the MXG install isnot complex, and old versions of MXG can be retained for comparison purposes.

MXG FORMATs Library

MXG uses a large number of bespoke `FORMAT`s to aid interpretation of that data that it processes. It is entirely possible that new `FORMAT`s are added or that existing ones change with each release of MXG. Equally, it may be that processing `FORMAT`s by Altair SLC may change from time to time.

Altair SLC cannot process data held in a `FORMAT`s library that was created by using the SAS System. As a result, the `FORMAT`s library MUST be rebuilt using Altair SLC before trying to use it, and we recommend that the MXG `FORMAT`s library is rebuilt every time that Altair SLC or MXG is upgraded.

The <WPSPFX>.CNTL (@MXGFMTS) member is provided to aid in this process. The JCL there is written in such a way that an MXG FORMATs library may have a name that features both Altair SLC and MXG version numbers, but use of that facility is entirely driven by your site's file naming conventions.

CICSIFUE exit

Both CICS and DB2 regions can be modified so that any monitor data they generate can be passed to the SMF data recorder for processing. The amount of data that can be produced by either application can be extremely large, so further parameterisation is possible that causes the data to be compressed prior to it being sent to SMF. Once compressed, the data will need to be decompressed before being processed by MXG scripts and added to existing data libraries.

The MXG distribution provides an assembler program in the CICSIFUE member that will accomplish the required decompression processing. The program can be assembled and linkage edited into a load library by tailoring the JCL held in the MXG-supplied member EXITCICS.

To use EXITCICS with Altair SLC, you must change the parameter string passed to the LKED step from:

```
//LKED EXEC PGM=IEWBLINK,
//          PARM='XREF,LIST',
//          COND=(0,NE,ASM)
```

to:

```
//LKED EXEC PGM=IEWBLINK,
//          PARM='XREF,LIST,LET,RMODE=ANY,AMODE=31,RENT,REUS,REFR',
//          COND=(0,NE,ASM)
```

Without this change, any SAS language program that uses the exit will abort with a U4038 abend code, even if there is no compressed data in the input file.

The target load library might feature in the system-wide LNKLST but if not, then the STEPLIB DD statement in the MXGWPSV4 JCL must be modified to concatenate the library together with the load library.

Once that is done, the statement:

```
%LET SMFEXIT=CICS;
```

must be inserted as one of the earliest statements in the SAS language program used to process the SMF data files.

If the CICSIFUE exit is not prepared, then MXG will dynamically decompress data that has been decompressed by CICS and/or DB2, but only if the data was collected in conventional MANx files. MXGs dynamic decompression is not supported for data being read directly from LOGGER files. If compressed data is held in LOGGER files then it must be extracted to a normal sequential file before being input to MXG.

It must be noted that the dynamic decompression process uses considerably more CPU time that the assembled exit.

Just as with the FORMATs library, we recommend that the CICSIFUE exit is re-assembled upon any change of either MXG or Altair SLC.

MXG SOURCLIB and USERID SOURCLIB

The MXG source library that is delivered with each installation package is a PDS containing a large number of members that are selectively brought in to play in a given script with the use of the %INCLUDE statement. Each such code segment may %%INCLUDE further segments also.

It is intended that the contents of the this distributed PDS are never changed. Any required changes should be made to duplicate members copied in to the so-called USERID SOURCLIB, which is another PDS that holds modified MXG code. This pair of libraries is accessed by the SOURCLIB concatenation in the JCL used to run Altair SLC with MXG. The actual JCL will be similar to:

```
//SOURCLIB DD DSN=SITE.SPECIFIC.MXG.CODE,DISP=SHR
//         DD DSN=MXG.V3102.SOURCLIB,DISP=SHR
```

Given these statements, any %INCLUDE statement that refers to SOURCLIB will search the *SITE.SPECIFIC.MXG.CODE* library first.

Note:

It is essential that the segments of MXG that are copied to the USERID SOURCLIB and modified there are kept up-to-date with the release of MXG in use. It may be that a piece of tailored code ceases to work because support for that piece of code has changed with a later version of MXG.

Such a problem can easily cascade through the remainder of the affected program, and the original cause of the problem becomes easy to miss, given the huge amount of diagnostic information that may result. World Programming will endeavour to support a site affected by such problems, but true responsibility lies with the site.

MXGWPSV4 procedure

Member MXGWPSV4 in the MXG SOURCLIB is a set of JCL that invokes Altair SLC and prepares that session for use with MXG. It is essentially the same as the <WPSPFX>.CNTL (WPSPROC) JCL, but with a few more DD statements added. We recommend that this procedure is used for all invocations of SAS language programs that use MXG. Careful study of the MXG SOURCLIB will reveal the existence of member MXGWPSV3 also, but as at version 36.09 of MXG, there is no practical difference between the two members.

A copy of the MXGWPSV4 member appears below:

```
//* JCL PROCEDURE EXAMPLE TO EXECUTE MXG PROGRAMS UNDER WPS V2 ON Z/OS
//* LAST UPDATED: OCT  9, 2009.  CHANGE 27.239.
//MXGWPSV4 PROC WPSHLQ='HLQ.WPS',                HLQ FOR WPS LIBRARIES
//  MXGHLQ='HLQ.MXG',           HIGH LEVEL QUALIFIER FOR MXG LIBRARIES
```

```
//  WORK='500,500',              WORK SPACE, CYLINDERS
//  WORKVOL=5,                   NUMBER OF WORK VOLUMES THAT CAN BE USED
//  WORKMDL='HLQ.WPS.SASHELP',   MODEL WORK SPACE
//  SORT=500,                    SORT WORK SPACE, CYLINDERS
//  SYSPARM='',                                 /* PROGRAM PARAMETERS  */
//  OPTIONS=''                                  /* WPS OPTIONS         */
//*
//*******************************************************************
//*  USES "PRODUCTION" WPS FORMATS LIBRARY (INCOMPATIBLE WITH SAS)    **
//*  //LIBRARY   WPSHLQ.MXG.LIBRARY.WPSDATA  -  WPS FORMAT LIBRARY    **
//*                                                                   **
//*  USES "PRODUCTION" MXG DATA SET NAMES OF                          **
//*                                                                   **
//*     DDNAME         DSNAME                CONTENTS               **
//*  //SOURCLIB  MXG.USERID.SOURCLIB     - INSTALLATION TAILORING    **
//*  //  CONCAT  MXG.MXG.SOURCLIB        - MXG MASTER SOURCE LIBRARY **
//*  //LIBRARY   WPS.MXG.FORMATS.WPSDATA - FORMAT LIBRARY            **
//*                                                                   **
//*  PLEASE NOTE: YOU MUST EXAMINE YOUR INSTALLATIONS "REAL" WPS      **
//*  PROCEDURE IN YOUR PROC LIB, AND MAKE SURE THAT ALL OF THE        **
//*  DATA SETS THAT ARE IN THE //STEPLIB CONCATENATION IN THE "REAL" **
//*  PROC ARE ALSO LISTED IN THE SAME ORDER IN THIS "MXGWPSV8" PROC.  **
//*                                                                   **
//*  VENDOR:  MERRILL CONSULTANT'S MXG SOFTWARE  214-351-1966         **
//*                                                                   **
//*  THE DEFAULT SYMBOLICS DEFINED ABOVE ASSUME YOUR HIGH LEVEL       **
//*   QUALIFIER FOR WPS LIBRARIES IS "WPS" AND YOUR HIGH              **
//*   LEVEL QUALIFIER FOR MXG LIBRARIES IS "MXG".                     **
//*                                                                   **
//*  COMPARE WITH YOUR EXISTING JCL PROC FOR WPS UNDER MVS.
//*                                                                   **
//* NOTE: YOU CAN TAILOR THIS JCL PROCEDURE INTO YOUR "USERID.SOURCLIB"
//*       PDS, AND USE A JCLLIB STATEMENT TO ACCES THE PROCEDURE, IF
//*       YOU DO NOT WANT TO PUT IT IN YOUR SYSTEM PROCLIB.
//*       THE SYNTAX OF YOUR JCL THEN WOULD BE
//*           //MYJOB1  JOB  ....
//*           //MYJCL   JCLLIB ORDER=MXG.USERID
//*           //STEP1   EXEC MXGWPSV4
//*                                                                   **
//*******************************************************************
//*
//WPS      EXEC PGM=WPSHOST,REGION=0M,
//  PARM=('&OPTIONS SYSPARM=''&SYSPARM''')
//STEPLIB  DD DISP=SHR,DSN=&WPSHLQ..LOAD
//WORK      DD UNIT=(SYSDA,&WORKVOL),SPACE=(CYL,(&WORK),,,ROUND),
//             LIKE=&WORKMDL
//SORTWK01  DD UNIT=SYSDA,SPACE=(CYL,(&SORT))
//SORTWK02  DD UNIT=SYSDA,SPACE=(CYL,(&SORT))
//SORTWK03  DD UNIT=SYSDA,SPACE=(CYL,(&SORT))
//SASAUTOS DD DISP=SHR,DSN=&WPSHLQ..CNTL
//CONFIG   DD DISP=SHR,DSN=&WPSHLQ..CNTL(CONFIG)
//         DD DISP=SHR,DSN=&MXGHLQ..MXG.SOURCLIB(CONFIGW2)
//SASHELP  DD DISP=SHR,DSN=&WPSHLQ..SASHELP
//SETINIT  DD DISP=SHR,DSN=&WPSHLQ..SETINIT
//PLUGCFG  DD DISP=SHR,DSN=&WPSHLQ..CNTL(PLUGCFG)
//ODSCSS   DD DISP=SHR,DSN=&WPSHLQ..CNTL(ODSCSS)
//CEEOPTS  DD DISP=SHR,DSN=&WPSHLQ..CNTL(CEEOPTS)
//SASLOG   DD SYSOUT=*
//SASLIST  DD SYSOUT=*
//WPSFONTS DD DISP=SHR,DSN=&WPSHLQ..FONTS
```

```
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//WPSTRACE DD SYSOUT=*
//CEEDUMP  DD SYSOUT=*
//CEERPT   DD SYSOUT=*
//DFSPARM  DD DISP=SHR,DSN=&WPSHLQ..CNTL(DFSPARM)
//SORTMSGS DD SYSOUT=*
//WPSAOINI DD DISP=SHR,DSN=&WPSHLQ..CNTL(WPSAOINI)
//DSNAOINI DD DISP=(NEW,DELETE),DSN=&&DSNAOINI,
//            DSORG=PS,RECFM=FB,LRECL=80,
//            SPACE=(TRK,1),UNIT=SYSDA
//NEWS     DD DISP=SHR,DSN=&WPSHLQ..CNTL(NEWS)
//INSTREAM  DD UNIT=SYSDA,SPACE=(CYL,(1,20)),
//              RECFM=FB,LRECL=80,BLKSIZE=0
//MXGTEMP   DD UNIT=SYSDA,SPACE=(CYL,(1,20))
//LIBRARY  DD DISP=SHR,DSN=&WPSHLQ..MXG.FORMATS.WPSDATA
//SOURCLIB DD DISP=SHR,DSN=&MXGHLQ..USERID.SOURCLIB
//          DD DISP=SHR,DSN=&MXGHLQ..MXG.SOURCLIB
//          PEND
```

The `CONFIG` DD statement points firstly to the standard list of options held in the `<WPSPFX>.CNTL` (CONFIG) member. These options are then replaced or augmented by further options held in `<MXGHLQ>.SOURCLIB` (CONFIGW2). This extra `CONFIG` list features the line:

```
INITSTMT='%INCLUDE SOURCLIB(VMXGINIT);%VMXGINIT;RUN;'
```

The `VMXGINIT` code segment defines a macro that is then scheduled by the `%VMXGINIT` statement in the above line. It is this code that initialises an environment for MXG to run in a WPS session. If the `VMXGINIT` code member is ever corrupted or changed in any way, or if `%VMXGINIT` is not used for some reason, then attempted use of MXG will result in numerous problems.

To make this JCL useable across the installation, default values for the *WPSHLQ* and *MXGHLQ* symbolic variables must be made. Also note the *WORKMDL* symbolic contains *the value of* the *WPSHLQ* symbolic, NOT the `&WPSHLQ` substitution variable itself. A permanent change needs to be made here, too. Following a successful test of the changes, the modified MXGWPSV4 procedure can be copied to a library on the system-wide PROCLIB concatenation.

MXG data libraries

Depending on the level of tailoring of MXG at your site, there may be up to four data libraries that need to be catered for. These are:

- The Performance Database, or `PDB`. Contains the data that is generated as part of processing an input file of – usually – SMF data by MXG.
- The `SPIN` library. Contains data for tasks that were not completed at the time of dumping the data for input to MXG.
- The `CICSTRAN` library. Contains data for CICS tasks. This file is frequently very large and so is often held on tape.
- The `DB2ACCT` library. Contains data for DB2 tasks. This file is frequently very large and so it might be held on tape, but it is almost certainly processed and maintained on a shorter-term basis than that used for the PDB.

If MXG is used in exactly the way prescribed in the associated documentation, the `PDB` library will be copied through to weekly and monthly roll-up versions. These files need to be handled in just the same wa as the primary `PDB` library.

The `PDB` and `SPIN` libraries are normally held on disk and so they are relatively safe from corruption by a mis-managed migration to Altair SLC. Prior to migration, these files would be maintained by use of the SAS System, and Altair SLC simply will not write to such files. In order to use Altair SLC to update these files, they must be copied before first use. `<WPSPFX>.CNTL` members @SAS2WPS and/or @SAS2WPX can be used for this. The difference between the two is that @SAS2WPX will generate a tape-based backup copy of the input data library.

If the `CICSTRAN` and/or the `DB2ACCT` libraries are held on disk, then they will need to be migrated in the same fashion before first use. If, however, they are on tape, then extra care must be taken. With tape-based data libraries, Altair SLC creates the entire library in a single program step, and so the library is completely recreated each time it is processed. Unlike disk-based files, there is no possibility of updating a file held in a tape-based library. For this reason, the file `DSName` that is associated with a tape-based library created and maintained by the SAS System must not be used by Altair SLC until after migration is complete. Altair SLC will from but not write to a data library created by the SAS System. The SAS System will neither read or write an Altair SLC data library.

MXG migration process

A general list of the points to be covered appears below; this might be used to arrive at a checklist of tasks to be performed, for your own change control purposes.

Ensure enough extra resources are available.

We strongly recommend a period of parallel running to cover the migration from the SAS System with MXG to Altair SLC with MXG. A sudden death cutover should not be considered, but the value of your site's Capacity Planning, Performance Measurement and/or Accounting data is the true controlling influence here. If there is to be a period of parallel running, then the amount of disk and tape storage, and the amount of CPU time used by these functions will be at least double for the duration of the migration process.

Upgrade to the latest Altair SLC version.

If the version of Altair SLC currently installed at your site is not the latest GA version available, then an upgrade is highly recommended. Whatever version is currently installed, it should be verified to be smoothly running and readily available.

Install latest MXG version.

If the version of MXG currently installed at your site is not the latest version available, then an upgrade is highly recommended. Whatever version is currently installed, it should be verified to be smoothly running and readily available.

Create local version of MXGWPSV4.

We recommend use of the MXG-supplied `MXGWPSV4` JCL to invoke Altair SLC with MXG. Take a copy of the JCL and modify it as required, then place it in a library in the system-wide `PROCLIB` concatenation. Verify Altair SLC and MXG are availability and running.

Create new MXG FORMATs library.

Modify and use the supplied @MXGFMTS jobstream to create an Altair SLC `FORMAT`s library. Remember that the output file will be different from the one created by the SAS System. The SAS System will not process the Altair SLC-based version of the `FORMAT`s library and vice versa.

Create new CICSIFUE exit.

If required, modify and use a copy of the MXG-supplied `EXITCICS` jobstream to create a specific version of the `CICSIFUE` exit. Versions of `CICSIFUE` used by the SAS System and by Altair SLC are different, and they cannot co-exist in the same load library. A separate load library will have to created for the duration of any parallel running, and a change to the JCL STEPLIB statement will be required to cover the new library name.

Examine the code in the MXG 'USERID SOURCLIB'.

It may be that some of your tailored MXG code in your `USERID SOURCLIB` is specific for the SAS System. Only close examination will reveal such potential problems. It is best to copy the `USERID SOURCLIB` to another library for use with Altair SLC. JCL changes will be necessary to cover this.

Create migrated copies of all affected data libraries.

The `PDB` and `SPIN` libraries, and the CICSACCT and DB2ACCT libraries along with any other long-term roll up summary libraries should be migrated by copying them under control of Altair SLC. The @SAS2WPS and @SAS2WPX members that are supplied in the `<WPSPFX>.CNTL` library can be adapted for this purpose.

Attempt a parallel run and compare outputs.

Having satisfied local requirements regarding the above changes, it should now be possible to attempt a parallel run of both the SAS system and Altair SLC, and then compare the outputs. In general terms, MXG is used to process SMF data collected on a daily basis. The same input SMF file should be used as input, and then the updated data libraries should be compared to satisfy local acceptance criteria. The supplied @COMPARE JCL member can be used to accomplish such a comparison. There should be no differences, but if there are and the source of those differences is not immediately obvious, then please contact Altair support.

MXG reporting

Reporting from MXG tends to vary from site to site. Some notes are provided below which may prove useful.

Standard MXG reporting

Most reports from MXG should run with no additional migration work required.

Custom MXG reporting

It is not possible to provide exact instructions for migrating and validating user-written reporting processes in a generic guide such as this. This is best dealt with on an individual basis, please contact Altair for help.

In certain circumstances, the layout of the printed report output layout might differ slightly in Altair SLC. In some cases, this might require minor adjustments to processes that consume such output.

Appendix A - examples

This section contains the following examples.

Example – JCL for simple migration to Altair SLC

This example code is provided in the `<wpspfx>.CNTL` library as member @SAS2WPS.

```
// <add a jobcard here>
//PROCLIB  JCLLIB ORDER=(<wpspfx>.CNTL)
//*
//* SYMBOL SASDL IS THE DSNAME OF THE SAS DATA-LIBRARY ON DASD
//*        SET SASDL=<sas-dasd-data-library>
//*
//* SYMBOL WPSDL IS THE DSNAME OF THE WPS DATA-LIBRARY ON DASD
//*        SET WPSDL=<wps-dasd-data-library>
//*
//* NOTE : USE REGION=0M TO OBTAIN MAXIMUM AVAILABLE MEMORY
//*
//*-------------------------------------------------------------------*/
//* SAMPLE JOB TO MIGRATE A SAS DASD DATA LIBRARY TO WPS DASD         */
//*-------------------------------------------------------------------*/
//*
//* (1)  ADD A SUITABLE JOBCARD
//* (2)  CHANGE <wpspfx> TO THE WPS INSTALLATION DATASET PREFIX
//* (3)  CHANGE <sas-dasd-data-library> TO THE SOURCE SAS DATASET
//* (4)  CHANGE <wps-dasd-data-library> TO THE WPS TARGET DATASET
//* (5)  CHANGE <wps-procedure-name> TO THE WPS PROCEDURE NAME
//* (6)  SUBMIT THIS JOB AND THEN CHECK THE OUTPUT
//* (7)  CHECK FOR A JOB RETURN CODE OF ZERO
//*
//*-------------------------------------------------------------------*/
```

```
//* TO MIGRATE A SAS TAPE DATA LIBRARY TO WPS DASD,                    */
//*   REPLACE 'LIBNAME SASDL SASDASD;' WITH 'LIBNAME SASDL SASSEQ;'    */
//*-------------------------------------------------------------------*/
//*
//WPS      EXEC <wps-procedure-name>
//SOURCLIB DD DISP=SHR,DSN=<wpspfx>.CNTL
//SASDL    DD DISP=SHR,DSN=&SASDL,
//            DCB=BUFNO=32
//WPSDL    DD DISP=(NEW,CATLG),DSN=&WPSDL,
//            UNIT=SYSDA,SPACE=(TRK,(9000,900),RLSE)
//*
//SYSIN    DD DATA,DLM='++'

*OPTIONS SOURCE SOURCE2 MPRINT MACROGEN MLOGIC;

 LIBNAME SASDL SASDASD;
*LIBNAME SASDL SASSEQ;

PROC DATASETS LIB=WPSDL KILL; RUN;
PROC COPY IN=SASDL OUT=WPSDL MEMTYPE=DATA; RUN;

RUN;

++ END OF //SYSIN
```

Example – JCL for migration to Altair SLC via tape

This example code is provided in the `<wpspfx>.CNTL` library as member @SAS2WPX.

```
// <add a jobcard here>
//PROCLIB  JCLLIB ORDER=(<wpspfx>.CNTL)
//*
//* SYMBOL SASDL IS THE DSNAME OF THE SAS DATA-LIBRARY ON DASD
//*        SET SASDL=<sas-dasd-data-library>
//*
//* SYMBOL SASTL IS THE DSNAME OF THE SAS DATA-LIBRARY ON TAPE
//*         SET SASTL=<sas-tape-data-library>
//*
//* SYMBOL WPSDL IS THE DSNAME OF THE WPS DATA-LIBRARY ON DASD
//*         SET WPSDL=<wps-dasd-data-library>
//*
//* NOTE : USE REGION=0M TO OBTAIN MAXIMUM AVAILABLE MEMORY
//*
//*-------------------------------------------------------------------*/
//* SAMPLE JOB TO MIGRATE A SAS DASD DATA LIBRARY TO WPS DASD          */
//*-------------------------------------------------------------------*/
//*
//* (1)  ADD A SUITABLE JOBCARD
//* (2)  CHANGE <wpspfx> TO THE WPS INSTALLATION DATASET PREFIX
//* (3)  CHANGE <sas-dasd-data-library> TO THE SOURCE SAS DATASET
//* (4)  CHANGE <sas-tape-data-library> TO THE SAS TAPE DATASET
//* (5)  CHANGE <wps-dasd-data-library> TO THE WPS TARGET DATASET
//* (6)  CHANGE <sas-procedure-name> TO THE SAS PROCEDURE NAME
```

```
//* (7)  CHANGE <wps-procedure-name> TO THE WPS PROCEDURE NAME
//* (8)  SUBMIT THIS JOB AND THEN CHECK THE OUTPUT
//* (9)  CHECK FOR A JOB RETURN CODE OF ZERO
//*
//*-------------------------------------------------------------------*/
//*
//*===================================================================
//* EXPORT A SAS DATA LIBRARY FROM DASD TO TAPE (NOT COMPRESSED)
//*===================================================================
//SAS      EXEC <sas-procedure-name>
//SOURCLIB DD DISP=SHR,DSN=<wpspfx>.CNTL
//SASDL    DD DISP=SHR,DSN=&SASDL
//SASTL    DD DSN=&SASTL,DISP=(,CATLG,DELETE),
//            UNIT=(CART,,DEFER),VOL=(,,,10),LABEL=EXPDT=98007
//*
//SYSIN    DD DATA,DLM='++'

*OPTIONS SOURCE SOURCE2 MPRINT MACROGEN MLOGIC;
%INCLUDE SOURCLIB(XMIGRATE);
%LET VERBOSE=YES;
%SD2ST(SASDL=SASDL,SASTL=SASTL);

RUN;

++ END OF //SYSIN
//*
//*===================================================================
//* IMPORT A SAS DATA LIBRARY FROM TAPE INTO A WPS DATA LIBRARY ON DASD
//*===================================================================
//*
//WPS      EXEC <wps-procedure-name>
//SOURCLIB DD DISP=SHR,DSN=<wpspfx>.CNTL
//SASTL    DD DISP=SHR,DSN=&SASTL,
//            DCB=BUFNO=32
//WPSDL    DD DISP=(NEW,CATLG),DSN=&WPSDL,
//            UNIT=SYSDA,SPACE=(TRK,(9000,900),RLSE)
//*
//SYSIN    DD DATA,DLM='++'
*OPTIONS SOURCE SOURCE2 MPRINT MACROGEN MLOGIC;
%INCLUDE SOURCLIB(XMIGRATE);
%LET VERBOSE=YES;
%ST2WD(SASTL=SASTL,WPSDL=WPSDL);
RUN;
++ END OF //SYSIN
```

Example – JCL for format migration to Altair SLC

This example code is provided in the `<wpspfx>.CNTL` library as member @FMT2WPS.

```
// <add a jobcard here>
//PROCLIB  JCLLIB ORDER=(<wpspfx>.CNTL)
//*
//* SYMBOL SASFL IS THE DSNAME OF THE SAS FORMAT LIBRARY
```

```
//         SET SASFL=<sas-dasd-fmt-library>
//*
//* SYMBOL WPSFL IS THE DSNAME OF THE WPS FORMAT LIBRARY
//          SET WPSFL=<wps-dasd-fmt-library>
//*
//* NOTE : USE REGION=0M TO OBTAIN MAXIMUM AVAILABLE MEMORY
//*
//*-------------------------------------------------------------------*/
//* SAMPLE JOB TO MIGRATE A SAS FORMAT LIBRARY TO WPS                  */
//*-------------------------------------------------------------------*/
//*
//* (1)  ADD A SUITABLE JOBCARD
//* (2)  CHANGE <wpspfx> TO THE WPS INSTALLATION DATASET PREFIX
//* (3)  CHANGE <sas-dasd-fmt-library> TO THE SAS SOURCE FORMAT LIBRARY
//* (5)  CHANGE <wps-dasd-fmt-library> TO THE WPS TARGET FORMAT LIBRARY
//* (6)  CHANGE <sas-procedure-name> TO THE SAS PROCEDURE NAME
//* (7)  CHANGE <wps-procedure-name> TO THE WPS PROCEDURE NAME
//* (8)  SUBMIT THIS JOB AND THEN CHECK THE OUTPUT
//* (9)  CHECK FOR A JOB RETURN CODE OF ZERO
//*
//*-------------------------------------------------------------------*/
//*
//*====================================================================
//* EXPORT A SAS FORMAT LIBRARY TO A TEMPORARY FILE
//*====================================================================
//SAS      EXEC <sas-procedure-name>
//SOURCLIB DD DISP=SHR,DSN=<wpspfx>.CNTL
//SASFL    DD DISP=SHR,DSN=&SASFL
//SASDL    DD DSN=&&TEMPFILE,DISP=(NEW,PASS),
//            UNIT=SYSDA,SPACE=(TRK,(9000,900),RLSE)
//*
//SYSIN    DD DATA,DLM='++'

*OPTIONS SOURCE SOURCE2 MPRINT MACROGEN MLOGIC;
%INCLUDE SOURCLIB(XMIGRATE);

%LET VERBOSE=YES;
%SF2SD(SASFL=SASFL,SASDL=SASDL);

RUN;

++ END OF //SYSIN
//*
//*====================================================================
//* CREATE A WPS FORMAT LIBRARY FROM THE TEMPORARY FILE
//*====================================================================
//*
//WPS      EXEC <wps-procedure-name>
//SOURCLIB DD DISP=SHR,DSN=<wpspfx>.CNTL
//SASDL    DD DISP=SHR,DSN=&&TEMPFILE
//WPSFL    DD DISP=(NEW,CATLG),DSN=&WPSFL,
//            UNIT=SYSDA,SPACE=(TRK,(9000,900),RLSE)
//*
//SYSIN    DD DATA,DLM='++'

*OPTIONS SOURCE SOURCE2 MPRINT MACROGEN MLOGIC;
%INCLUDE SOURCLIB(XMIGRATE);

%LET VERBOSE=YES;
```

```
%SD2WF(SASDL=SASDL,WPSFL=WPSFL);

RUN;

++ END OF //SYSIN
```

Example – Altair SLC migration macros

This example code is provided in the <wpspfx>.CNTL library as member XMIGRATE.

```
  %*********************************************************************;
  %* SAS SOURCE LIBRARY CONTAINING MACROS TO SUPPORT THE SAS TO WPS    *;
  %* MIGRATION PROCESS. SEE SAS2WPS MEMBER OF CNTL LIBRARY FOR AN      *;
  %* EXAMPLE JCL JOB THAT USES THESE MACROS.                           *;
  %*********************************************************************;


%MACRO WPSORSAS();
  %*********************************************************************;
  %* WPSORSAS                                                          *;
  %* DESCRIPTION: MACRO TO CHECK WHETHER WPS OR SAS IS RUNNING         *;
  %* ARGUMENTS:   NONE                                                 *;
  %* RETURNS:     WPS OR SAS                                           *;
  %*********************************************************************;
  %IF %SYSPROD(WPS) = 1 %THEN
  %DO;
    WPS
  %END;
  %ELSE
  %DO;
    SAS
  %END;
%MEND WPSORSAS;


%MACRO SD2ST(SASDL=SASDL, SASTL=SASTL);
  %*********************************************************************;
  %* SD2ST                                                             *;
  %* DESCRIPTION: MACRO TO EXPORT A SAS DASD LIBRARY TO SAS TAPE       *;
  %*  ARGUMENTS:                                                       *;
  %*         SASDL - THE SOURCE SAS DASD DATA LIBRARY                  *;
  %*         SASTL - THE TARGET SAS TAPE DATA LIBRARY                  *;
  %* NOTE1: THIS MUST BE RUN FROM SAS NOT WPS                          *;
  %*********************************************************************;
  OPTIONS COMPRESS=NO;
  LIBNAME &SASTL TAPE;
  PROC COPY IN=&SASDL OUT=&SASTL NOCLONE MEMTYPE=DATA; RUN;
%MEND SD2ST;


%MACRO ST2WD(SASTL=SASTL, WPSDL=WPSDL);
  %*********************************************************************;
  %* ST2WD                                                             *;
  %* DESCRIPTION: MACRO TO IMPORT A SAS TAPE LIBRARY TO WPS DASD       *;
  %*  ARGUMENTS:                                                       *;
  %*         SASTL - THE SOURCE SAS TAPE DATA LIBRARY                  *;
```

```
  %*        WPSDL - THE TARGET WPS DASD DATA LIBRARY                    *;
  %*                                                                    *;
  %* NOTE1: THIS MUST BE RUN FROM WPS NOT SAS                           *;
  %*********************************************************************;
  LIBNAME &SASTL SASSEQ;
  PROC DATASETS LIB=&WPSDL KILL; RUN;
  PROC COPY IN=&SASTL OUT=&WPSDL; RUN;
  %IF "&VERBOSE" EQ "YES" %THEN
  %DO;
    TITLE2 "CONTENTS OF &WPSDL WPS DASD DATA LIBRARY";
    PROC DATASETS LIB=&WPSDL;
    RUN;
  %END;
%MEND ST2WD;


%MACRO S2WCOMP(SASDL=SASDL, WPSDL=WPSDL, METHOD=, CRITERION=);
  %*********************************************************************;
  %* S2WCOMP                                                           *;
  %* DESCRIPTION: COMPARE SAS TAPE LIBRARY WITH WPS DASD LIBRARY       *;
  %*  ARGUMENTS:                                                       *;
  %*        SASDL     - A SAS TAPE DATA LIBRARY                       *;
  %*        WPSDL     - A WPS DASD DATA LIBRARY                       *;
  %*        METHOD    - METHOD FOR PROC COMPARE                       *;
  %*        CRITERION - CRITERION FOR PROC COMPARE                    *;
  %* NOTE1: THIS MUST BE RUN FROM WPS NOT SAS                          *;
  %*********************************************************************;
  TITLE1 "S2WCOMP - SAS TO WPS DATA MIGRATION LIBRARY COMPARISON";
  TITLE2 "CONTENTS OF &SASDL SAS TAPE DATA LIBRARY";
  PROC DATASETS LIB=&SASDL;
  RUN;
  %IF "&VERBOSE" EQ "YES" %THEN
  %DO;
    TITLE2 "CONTENTS OF &WPSDL WPS DASD DATA LIBRARY";
    PROC DATASETS LIB=WPSDL; RUN;
  %END;

  PROC SQL;
  CREATE TABLE MEMNAMES AS
  SELECT MEMNAME FROM DICTIONARY.MEMBERS
  WHERE LIBNAME LIKE "&SASDL" AND MEMTYPE = 'DATA';

  DATA _NULL_;
  SET MEMNAMES END=LAST;
  LENGTH SASCODE £80;
  SASCODE =
      '%COMPRMBR('
    || "SASLIB=&SASDL"
    ||",WPSLIB=&WPSDL"
    ||",MEMNAME="||TRIM(MEMNAME)
    ||",METHOD=&METHOD"
    ||",CRITERION=&CRITERION"
    ||")" ;
  CALL EXECUTE(SASCODE);
%MEND S2WCOMP;

%MACRO COMPRMBR(SASLIB=SASDL, WPSLIB=WPSDL, MEMNAME=MEMNAME,
                METHOD=, CRITERION=);
  %*********************************************************************;
  %* COMPRMBR                                                          *;
```

```
  %* DESCRIPTION: COMPARE INDIVIDUAL SAS MEMBER WITH WPS MEMBER       *;
  %*  ARGUMENTS:                                                      *;
  %*        SASDL     - THE SAS DATA LIBRARY                        *;
  %*        WPSDL     - THE WPS DATA LIBRARY                        *;
  %*        MEMNAME   - THE MEMBER NAME                             *;
  %*        METHOD    - METHOD FOR PROC COMPARE                     *;
  %*        CRITERION - CRITERION FOR PROC COMPARE                  *;
  %* NOTE1: IF &MXG MACRO VARIABLE IS SET TO YES THEN ZDATE ZTIME     *;
  %*        VARIABLES WILL BE EXCLUDED FROM THE COMPARISON          *;
  %*        THIS IS HELPFUL WHEN COMPARING MXG PDB LIBRARIES        *;
  %*****************************************************************;
  TITLE1 "COMPRMBR - SAS TO WPS DATA MIGRATION MEMBER COMPARISON";
  TITLE2 "MEMBER NAME: &MEMNAME";

  %IF "&VERBOSE" EQ "YES" %THEN
  %DO;
    TITLE3 "SAS DATA LIBRARY: &SASLIB";
    PROC CONTENTS DATA=&SASLIB..&MEMNAME;
    PROC PRINT DATA=&SASLIB..&MEMNAME(OBS=8) LABELANDNAME;
    TITLE3 "WPS DATA LIBRARY : &WPSLIB";
    PROC CONTENTS DATA=&WPSLIB..&MEMNAME;
    PROC PRINT DATA=&SASLIB..&MEMNAME(OBS=8) LABELANDNAME;
  %END;

  TITLE2 "COMPARE RESULTS";
  PROC COMPARE BASE=&WPSLIB..&MEMNAME
    COMP=&SASLIB..&MEMNAME
    OUT=WORK.SWMCOMP_&MEMNAME
    OUTNOEQUAL
 %IF "&METHOD" NE "" %THEN %DO;
 METHOD=&METHOD
 %END;
 %IF "&CRITERION" NE "" %THEN %DO;
 CRITERION=&CRITERION
 %END;
    NOTE;
    %IF "&MXG" EQ "YES" %THEN
    %DO;
    EXCLUDEVAR ZDATE ZTIME;
    %END;
    RUN;

  TITLE "RESULTS OF COMPARE OF MEMBER: &MEMNAME";
  PROC PRINT DATA=WORK.SWMCOMP_&MEMNAME; RUN;
%MEND COMPRMBR;

%MACRO SF2SD(SASFL=SASFL,SASDL=SASDL);
  %*****************************************************************;
  %* SF2TM                                                         *;
  %* DESCRIPTION: MACRO TO UNLOAD A SAS FORMAT LIBRARY TO A SAS DASD  *;
  %*              LIBRARY MEMBER                                   *;
  %*  ARGUMENTS:                                                   *;
  %*        SASFL - THE SOURCE SAS FORMAT LIBRARY                  *;
  %*        SASDL - THE TEMPORARY SAS DATA LIBRARY                 *;
  %* NOTE1: THIS MUST BE RUN FROM SAS NOT WPS                      *;
  %*****************************************************************;
  %IF "&VERBOSE" EQ "YES" %THEN
  %DO;
    PROC FORMAT LIBRARY=SASFL FMTLIB;RUN;
  %END;
```

```
  PROC FORMAT LIBRARY=SASFL CNTLOUT=SASDL.FORMATS;RUN;
%MEND SF2SD;

%MACRO SD2WF(SASDL=SASDL,WPSFL=WPSFL);
  %*******************************************************************;
  %* SD2WF                                                           *;
  %* DESCRIPTION: MACRO TO CREATE A WPS FORMAT LIBRARY FROM A SAS    *;
  %*              DATA LIBRARY MEMBER                                *;
  %*  ARGUMENTS:                                                     *;
  %*        SASDL - THE SOURCE SAS DATA LIBRARY                      *;
  %*        WPSFL - THE TARGET WPS FORMAT LIBRARY                    *;
  %*                                                                 *;
  %* NOTE1: THIS MUST BE RUN FROM WPS NOT SAS                        *;
  %*******************************************************************;
  PROC FORMAT LIBRARY=WPSFL CNTLIN=SASDL.FORMATS;RUN;
  %IF "&VERBOSE" EQ "YES" %THEN
  %DO;
    PROC FORMAT LIBRARY=WPSFL FMTLIB;RUN;
  %END;
%MEND SD2WF;
```

Example – JCL for comparing migration output

This example code is provided in the `<wpspfx>.CNTL` library as member @COMPARE.

```
// <add a jobcard here>
//PROCLIB  JCLLIB ORDER=(<wpspfx>.CNTL)
//*
//* SYMBOL SASDL IS THE DSNAME OF THE SAS DATA LIBRARY ON DASD
//         SET SASDL=<sas-dasd-data-library>
//*
//* SYMBOL WPSDL IS THE DSNAME OF THE WPS DATA LIBRARY ON DASD
//         SET WPSDL=<wps-dasd-data-library>
//*
//* NOTE : USE REGION=0M TO OBTAIN MAXIMUM AVAILABLE MEMORY
//*
//*-------------------------------------------------------------------*/
//* SAMPLE JOB TO RUN SAS TO WPS COMPARISON                           */
//*-------------------------------------------------------------------*/
//*
//* (1) ADD A SUITABLE JOBCARD
//* (2) CHANGE <wpspfx> TO THE WPS INSTALLATION DATASET PREFIX
//* (3) CHANGE <sas-dasd-data-library> TO THE SAS DASD DATA LIBRARY
//* (4) CHANGE <wps-dasd-data-library> TO THE WPS DASD DATA LIBRARY
//* (5) CHANGE <sas-formats-library> TO THE MXG FORMATS LIBRARY IF
//*            NECESSARY
//* (6) CHANGE <migration-macro-library> TO THE MIGRATION MACRO
//*            LIBRARY
//* (7) SUBMIT THIS JOB
//* (8) CHECK FOR A JOB RETURN CODE OF ZERO
//* (9) CHECK SASLIST FOR COMPARISON OUTPUT
//*
//*-------------------------------------------------------------------*/
```

```
//* TO COMPARE A SAS TAPE DATA LIBRARY TO WPS DASD,                   */
//*   REPLACE 'LIBNAME SASDL SASDASD;' WITH 'LIBNAME SASDL SASSEQ;' */
//*-------------------------------------------------------------------*/
//*
//*
//@COMPARE EXEC WPSPROC
//SASDL    DD DISP=SHR,DSN=&SASDL
//WPSDL    DD DISP=SHR,DSN=&WPSDL
//LIBRARY  DD DISP=SHR,DSN=<sas-formats-library>  <= IF NECESSARY
//MTSOURCE DD DISP=SHR,DSN=<migration-macro-library>
//SYSIN DD DATA,DLM='++'

OPTIONS SOURCE SOURCE2 MPRINT MLOGIC MACROGEN;
OPTIONS FMTSEARCH=(LIBRARY.FORMATS WORK.FORMATS);
%INCLUDE MTSOURCE(XMIGRATE);
%LET VERBOSE = NO;
%LET MXG = YES;

 LIBNAME SASDL SASDASD;
*LIBNAME SASDL SASSEQ;

%S2WCOMP(SASDL=SASDL,WPSDL=WPSDL);

RUN;
++ END OF //SYSIN
```

Example – MXG FORMATS library build

This example code is provided in the `<wpspfx.>CNTL` library as member @MXGFMTS.

```
// <add a jobcard here>
//PROCLIB  JCLLIB ORDER=(<wpspfx>.CNTL)
//*
//*-------------------------------------------------------------------*/
//* SAMPLE JOB TO RUN MXG FORMATS LIBRARY BUILDING JOB                */
//*-------------------------------------------------------------------*/
//*
//* (1) ADD A SUITABLE JOBCARD
//* (2) CHANGE <wpspfx> TO THE WPS INSTALLATION DATASET PREFIX
//* (3) CHANGE <mxgpfx> TO THE MXG SOURCLIB DATASET PREFIX
//* (4) CHANGE <mxgwpspfx> TO THE MXG+WPS DATASET PREFIX
//* (5) SUBMIT THIS JOB AND THEN CHECK THE OUTPUT
//* (6) CHECK FOR A JOB RETURN CODE OF ZERO
//*
//*-------------------------------------------------------------------*/
//*
//*===================================================================
//*
//ALLOC    EXEC PGM=IEFBR14
//LIBRARY  DD DISP=(NEW,CATLG),
//            DSN=<mxgwpspfx>.LIBRARY.WPSDATA,
//            SPACE=(TRK,(70,20)),
//            DCB=(RECFM=FS,LRECL=27648,BLKSIZE=27648),
//            DSORG=PS
//*
//*===================================================================
```

```
//*
//@MXGFMTS EXEC WPSPROC,CONFIG='<mxgpfx>.SOURCLIB(CONFIGW2)'
//SOURCLIB DD DISP=SHR,DSN=<mxgpfx>.USERID.SOURCLIB
//         DD DISP=SHR,DSN=<mxgpfx>.SOURCLIB
//LIBRARY  DD DISP=OLD,DSN=<mxgwpspfx>.LIBRARY.WPSDATA
//SYSIN    DD DATA,DLM='++'

*OPTIONS SOURCE SOURCE2 MPRINT MLOGIC MACROGEN;

OPTIONS FMTSEARCH=(LIBRARY);
%INCLUDE SOURCLIB(FORMATS);
RUN;

++ END OF //SYSIN
```

Example – JCL for a sample MXG BUILDPDB

This example code is provided in the `<wpspfx>.CNTL` library as member @MXGPDB.

```
// <add a jobcard here>
//PROCLIB  JCLLIB ORDER=(<wpspfx>.CNTL)
//*
//*-------------------------------------------------------------------*/
//* SAMPLE JOB TO RUN EMPTY MXG BUILDPDB JOB                          */
//*-------------------------------------------------------------------*/
//*
//* (1) ADD A SUITABLE JOBCARD
//* (2) CHANGE <wpspfx> TO THE WPS INSTALLATION DATASET PREFIX
//* (3) CHANGE <mxgpfx> TO THE MXG SOURCLIB DATASET PREFIX
//* (3) CHANGE <mxgwpspfx> TO THE MXG+WPS DATASET PREFIX
//* (4) SUBMIT THIS JOB AND THEN CHECK THE OUTPUT
//* (5) CHECK FOR A JOB RETURN CODE OF ZERO
//*
//*-------------------------------------------------------------------*/
//*
//*===================================================================
//*
//ALLOC    EXEC PGM=IEFBR14
//CICSTRAN DD DISP=(NEW,CATLG),
//            DSN=<mxgwpspfx>.CICSTRAN.WPSDATA,
//            SPACE=(TRK,(450,450))
//DB2ACCT  DD DISP=(NEW,CATLG),
//            DSN=<mxgwpspfx>.DB2ACCT.WPSDATA,
//            SPACE=(TRK,(450,450))
//PDB      DD DISP=(NEW,CATLG),
//            DSN=<mxgwpspfx>.PDB.WPSDATA,
//            SPACE=(CYL,(600,150))
//SPIN     DD DISP=(NEW,CATLG),
//            DSN=<mxgwpspfx>.SPIN.WPSDATA,
//            SPACE=(CYL,(10,100))
//*
//*===================================================================
//*
```

```
//@MXGPDB  EXEC WPSPROC,CONFIG='<mxgpfx>.SOURCLIB(CONFIGW2)'
//SOURCLIB DD DISP=SHR,DSN=<mxgpfx>.USERID.SOURCLIB
//         DD DISP=SHR,DSN=<mxgpfx>.SOURCLIB
//LIBRARY  DD DISP=SHR,DSN=<mxgwpspfx>.LIBRARY.WPSDATA
//CICSTRAN DD DISP=OLD,DSN=*.ALLOC.CICSTRAN
//DB2ACCT  DD DISP=OLD,DSN=*.ALLOC.DB2ACCT
//PDB      DD DISP=OLD,DSN=*.ALLOC.PDB
//SPIN     DD DISP=OLD,DSN=*.ALLOC.SPIN
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(150,150))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(150,150))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(150,150))
//SMF      DD DUMMY
//SYSIN    DD DATA,DLM='++'

*OPTIONS SOURCE SOURCE2 MPRINT MLOGIC MACROGEN;

OPTIONS FMTSEARCH=(LIBRARY);
%INCLUDE SOURCLIB(BUILDPDB);
RUN;

++ END OF //SYSIN
```

Legal Notices

Trademarks

Altair SLC™, Altair Analytics Workbench™, and Altair SLC Hub™ are registered trademarks or trademarks of Altair Engineering, inc. (r) or ® indicates a Community trademark.

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.

All other trademarks are the property of their respective owner.

General Notices

Altair Engineering, inc. is not associated in any way with the SAS Institute.

Altair SLC is not the SAS System.

The phrases "SAS", "SAS language", and "language of SAS" used in this document are used to refer to the computer programming language often referred to in any of these ways.

The phrases "program", "SAS program", and "SAS language program" used in this document are used to refer to programs written in the SAS language. These may also be referred to as "scripts", "SAS scripts", or "SAS language scripts".

The phrases "IML", "IML language", "IML syntax", "Interactive Matrix Language", and "language of IML" used in this document are used to refer to the computer programming language often referred to in any of these ways.

Altair SLC includes software developed by third parties. More information can be found in the THANKS or acknowledgments.txt file included in the Altair SLC installation.