Table of Contents

1 Introduction

Many times while developing software systems, the need arises for a way to access specific members of a Partitioned Data Set or PDS. To accomplish this in JCL is as simple as coding DSN with a members (i.e. DSN=SOME.PDS(MEMBER). To accomplish this in an assembler program, it can be done using various data management macros (i.e. BLDL,FIND etc). However, programs coded in other languages cannot access members of a PDS except via the JCL while treating the member as a sequential dataset.

I came up with a solution to the problem of accessing members of a PDS from a program. I found a program in the TK3 system called NCZ93205 which provides access to a PDS and any member within it. With some enhancements, a routine to sequentially read the PDS directory and access any and all members of the PDS.

There are three sample programs – PDSUNLDC, PDSUNLDP and PDSCANR – that demonstrate how to process an entire PDS. PDSUNLD? Reads the PDS directory and for each member write an IEBUPDTE statement to an output dataset followed by the member. This output dataset can be used as input to the IEBUPDTE or IEBUPDTX to create/recreate a PDS. Remember that these are samples to demonstrate how to use the subroutines and not to replace other utilities like IEBCOPY.

PDSCANR is a program to scan an entire PDS for given strings of text and produce a report of the

members that have matches to the given strings.

2 Installation

To install the PDSUTIL package:

1. download the PDSUTIL.zip OR clone github repository Oldbuzzard54 PDS-Access.
2. Extract the PDSUTIL.XMIT file, the RECEIVE and the optional INSTALL JCL.
3. Some transfer protocols do not create “card image” files. To ensure correct transfers, create an empty dataset on the host (MVS) – HERC01.PDSUTIL.XMIT as a sequential dataset (i.e. DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120,DSORG=PS))
4. Transfer the PDSUTIL.XMIT file to the host as a binary file HERC01.PDSUTIL.XMIT.
5. Submit the RECEIVE JCL. This job will create the HERC01.PDSUTIL.ASM, . COB, .LOADLIB, .PLI and .PROJECT datasets. All completion codes should be 0000.
6. That is it. You can now run submit the TESTIT JCL. You can delete the .XMIT and .PROJECT datasets since they are no longer needed.
7. OPTIONALLY submit the INSTALL JCL This job will generate and link the executables into HERC01.PDSUTIL.LOADLIB. All completion codes should be 0000 except for the two PL/1 and one COBOL compiles. The PL/1 will end with 0008 and COBOL with 0004 completion codes.
8. OPTION Due to limitations in XMIT370 and RECV370, the HERC01.PDSUTIL.LOADLIB was created with a BLKSIZE of 4096. If you want to merge these modules to another library with a different BLKSIZE, you will need to 1) edit INSTALL.JCL to create the loadlib with a different BLKSIZE or 2) edit the INSTALL2.JCL to relink with a different BLKSIZE.

3 Execution

There are no special requirements for execution other than the DD statement for the PDS in the JCL. If so desired, the GETPDS and GETPDSP can be dynamically loaded using DYNALDA and DYNALDP respectively. These can be found on Jay Moseley’s (jaymoseley.com) SYSCPK.

4 Coding

This section explains how to code your COBOL or PL/1 programs to use the routine. COBOL copybooks and PL/1 includes are provided for the parameters to the routines.

4.1 COBOL

In the working storage section, you will need to use the COPY statement to access the parms. For example:

```
01  PDSGET-PARAMETERS COPY PDSGETPA.
```

The LIB option must be given to the COBOL compiler from the exec statement:

…..,PARM=’LIB,….other parms’

Use the following to call PDSGET:

```
CALL 'GETPDS'   USING PDSGET-REQUEST,
                      PDSGET-MEMBER,
                      PDSGET-RECORD.
```

4.2 PL/1

In the program code somewhere so these DECLAREs are in the scope of variables to the CALL statement. I suggest within the MAIN procedure so the variable are known within the entire program. You will need to use the %INCLUDE statement to access the parms. For example:

```
%INCLUDE PDSGETPA;
```

The MACRO option must be given to the PL/1 compiler from the exec statement:

…..,PARM=’MACRO,….other parms’

Use the following to call PDSGETP:

```
CALL GETPDSP (PDSGET_REQUEST_1,
              PDSGET_MEMBER_1,
              PDSGET_RECORD80_1,
              PDSGET_RETURN_CODE_1);
```

4.3 Request Codes

This section explains request codes. The explanations are generic. Unless otherwise stated, the text applies to both COBOL and PL/1. After each call to the routine, the return code should be checked. In COBOL, the special register RETURN-CODE is used. In PL/1, the parameter return_code should be used.

4.3.1 Open Request

The request in the parm area is set to the request code. Member is set to the DDNAME of the PDS to be accessed. If left blank, PDSIN is assumed. The record area is not used but must be given in the call.

4.3.2. Close Request

The request in the parm area is set to the request code. Member and record area are not used but must be given in the call.

4.3.3 Locate Request

The request in the parm area is set to the request code. Member is set to the name of the member to be accessed. The read routine will be positioned to start with the first record in the member. If the member is found, the record area will be populated. If member has ISPF/RPF statistics, the record area will be populated. If the member is not found, the record area will be spaces.

4.3.4 Read Request

The request in the parm area is set to the request code. Upon return, the next record to be read is returned in the record area or a return code indicating end of member.

4.3.5 Start Request

The request in the parm area is set to the request code. The PDS directory is positioned to start a directory browse at the first member greater than or equal to the member name given. Record area is not returned.

4.3.6 Next Request

The request in the parm area is set to the request code. The PDS directory is advanced to the next member to be accessed. Record area is not returned. Read requests can be issued.

4.3.7 COBOL Return Codes

The special register RETURN-CODE is used to test the return codes. For all requests, 0 means success. For all requests, 8 means failure which should be considered as fatal. For most requests, 4 means the request was not successful but not fatal. For NEXT, RC 4 indicate end of directory. For LOCATE, RC 4 indicates member not found. For READ, RC 4 indicate end of member.

4.3.8 PL/1 Return Codes

A special register like COBOL’s RETURN-CODE does not exist. So the forth parm is added to the call. For all requests, 0 means success. For all requests, 8 means failure which should be considered as fatal. For most requests, 4 means the request was not successful but not fatal. For NEXT, RC 4 indicate end of directory. For LOCATE, RC 4 indicates member not found. For READ, RC 4 indicate end of member.

4.4 RPF/ISPF Statistics

These statistic are obtained from the PDS directory if present. Otherwise they are all blank. Dates are returned in Julian format (YYYYDDD). Time fields are UNSIGNED packed fields. For COBOL, there are instructions in the copybook how to use the provided fields to do the conversion. For PL/1, there is a function (PDS_TIME_CONVERSION) is included in the INCLUDE See the INCLUDE.

5 COBOL Copybook

```
****************** GETPDSPA V1.2.0 ****************************
*****************************************************************
*                                                               *
* NAME: GETPDSPA - PARM LIST DEFINITIONS FOR GETPDS V1.3.0      *
*                                                               *
* NOTE: THERE IS A PL/1 VERSION OF THIS.  THEY SHOULD BE KEPT   *
*       IN SYNC.                                                *
*                                                               *
*****************************************************************
*                                                               *
*  (C) COPYRIGHT 2017 EDWARD G LISS   ALL RIGHTS RESERVED       *
*  (C) COPYRIGHT 2023 EDWARD G LISS   ALL RIGHTS RESERVED       *
*                                                               *
*  THIS SOURCE CODE AS WELL AS ANY OBJECT CODE RESULTING FROM   *
*  THIS SOURCE CODE MAY BE DISTRIBUTED FREELY PROVIDED NO FEE   *
*  IS CHARGED AND FOR NON-COMERCIAL PURPOSES.                   *
*                                                               *
*  FOR COMMERCIAL DISTRIBUTION RIGHTS, CONTACT THE COPYRIGHT    *
*  OWNER.                                                       *
*                                                               *
*****************************************************************
*                                                               *
* REVISION HISTORY                                              *
* ------  ----------------------------------------------------- *
* V1.1.0  INITIAL VERSION.                                      *
* V1.2.0  ADDED THE DEFINITION OF MEMBERS STATS AND ADDED       *
*         WORK AREAS FOR STATISTICS CODE CONVERSION.            *
*                                                               *
*****************************************************************
 01  PDSGET-PARAMETERS.
     02  PDSGET-REQUEST          PIC S9(08)  COMP.
     02  PDSGET-MEMBER           PIC X(08).
     02  PDSGET-RECORD           PIC X(255).
     02  FILLER                  REDEFINES PDSGET-RECORD.
         03  PDSGET-RECORD80     PIC X(80).
         03  FILLER              PIC X(75).
     02  FILLER                  REDEFINES PDSGET-RECORD.
         03  PDS-C               PIC X.
         03  PDS-VERSION         PIC X.
         03  PDS-MOD             PIC X.
         03  PDS-NULL            PIC XX.
         03  PDS-DATE-CREATED    PIC S9(7) COMP-3.
         03  PDS-DATE-UPDATED    PIC S9(7) COMP-3.
         03  PDS-TIME-CHANGED-H  PIC X.
         03  PDS-TIME-CHANGED-M  PIC X.
         03  PDS-CURRENT-LINES   PIC S9(4) COMP.
         03  PDS-INITIAL-LINES   PIC S9(4) COMP.
         03  PDS-CHANGED-LINES   PIC S9(4) COMP.
         03  PDS-USER-ID         PIC X(8).
```



```
 01  PDSGET-REQUEST-CODES.
     02  PDSGET-REQUEST-OPEN     PIC S9(08)  COMP    VALUE +0.
     02  PDSGET-REQUEST-START    PIC S9(08)  COMP    VALUE +1.
     02  PDSGET-REQUEST-LOCATE   PIC S9(08)  COMP    VALUE +4.
     02  PDSGET-REQUEST-NEXT     PIC S9(08)  COMP    VALUE +5.
     02  PDSGET-REQUEST-READ     PIC S9(08)  COMP    VALUE +8.
     02  PDSGET-REQUEST-CLOSE    PIC S9(08)  COMP    VALUE +12.

*
* PDSGET_RETURN_CODE DEFINITIONS
*       RETURN_CODE 0 = SUCCESSFUL COMPLETION OF REQUEST
*       RETURN_CODE 8 = SERIOUS ERROR.
*       RETURN_CODE 4 = DEPENDS ON REQUEST CODE
*            REQUEST_CODE    MEANING
*                 0          PDS COULD NOT BE OPENED.
*                 1          PDS COULD NOT BE OPENED.
*                 4          MEMBER NOT FOUND.
*                 5          END OF DIRECTORY.
*                 8          END OF CURRENT MEMBER.
*

 01  PDSGET-WORK-AREAS.
*
*   PDS-TIME-CHANGED -H AND -M ARE UNSIGNED PACKED DECIMALS
*   THEY MUST TO CONVERTED TO SIGNED PACK DECIMAL TO USE THEM.
*   THE FOLLOWING AREAS CAN BE USED FOR THAT.
*
*   MOVE PDS-TIME-CHANGED-? TO PDS-TIME-CHAR.
*   MOVE PDS-TIME-WORK      TO PDS-4 DIGIT-TIME.
*
*   PDS-TIME WILL HAVE A USABLE FORM OF THE HOUR OR MIN.
*
*   THIS CONVERSION MUST TAKE PLACE FOR BOTH THE -H AND =M
*   PDS-TIME-CHANGED- FIELDS.
*
     05  PDS-TIME-CONVERSION.
         10 PDS-TIME-WORK        PIC S9(5) COMP-3 VALUE ZERO.
     05  FILLER                  REDEFINES PDS-TIME-CONVERSION.
         10  FILLER              PIC X.
         10  PDS-TIME-CHAR       PIC X.
         10  FILLER              PIC X.

     05  PDS-4-DIGIT-TIME        PIC 9(4).
     05  FILLER                  REDEFINES PDS-4-DIGIT-TIME.
         10  FILLER              PIC X.
         10  PDS-TIME            PIC 99.
         10  FILLER              PIC X.

****************** GETPDSPA V1.2.0 END ************************
```

6 PL/1 INCLUDE

```
/********************** GETPDSPA V1.2.0 ****************************/
/*******************************************************************/
/*                                                                 */
/* NAME: GETPDSPA - PARM LIST DEFINITIONS FOR GETPDS V1.3.0        */
/*                                                                 */
/* NOTE: THERE IS A COBOL VERSION OF THIS.  THEY SHOULD BE KEPT    */
/*       IN SYNC.                                                  */
/*                                                                 */
/*******************************************************************/
/*                                                                 */
/*  (C) COPYRIGHT 2017 EDWARD G LISS   ALL RIGHTS RESERVED         */
/*  (C) COPYRIGHT 2023 EDWARD G LISS   ALL RIGHTS RESERVED         */
/*                                                                 */
/*  THIS SOURCE CODE AS WELL AS ANY OBJECT CODE RESULTING FROM THIS  */
/*  SOURCE CODE MAY BE DISTRIBUTED FREELY PROVIDED NO FEE IS CHARGED */
/*  AND FOR NON-COMERCIAL PURPOSES.                                */
/*                                                                 */
/*  FOR COMMERCIAL DISTRIBUTION RIGHTS, CONTACT THE COPYRIGHT OWNER. */
/*                                                                 */
/*******************************************************************/
/*                                                                 */
/* REVISION HISTORY                                                */
/* ------  ------------------------------------------------------  */
/* V1.1.0  INITIAL VERSION.                                        */
/* V1.2.0  ADDED THE DEFINITION OF MEMBERS STATS AND ADDED         */
/*         WORK AREAS FOR STATISTICS CODE CONVERSION.              */
/*                                                                 */
/*******************************************************************/

    DECLARE
       1  PDSGET_PARAMETERS   STATIC ALIGNED,
           2  PDSGET_REQUEST_OPEN   INITIAL(0)  FIXED BINARY(31),
           2  PDSGET_REQUEST_START  INITIAL(1)  FIXED BINARY(31),
           2  PDSGET_REQUEST_LOCATE INITIAL(4)  FIXED BINARY(31),
           2  PDSGET_REQUEST_NEXT   INITIAL(5)  FIXED BINARY(31),
           2  PDSGET_REQUEST_READ   INITIAL(8)  FIXED BINARY(31),
           2  PDSGET_REQUEST_CLOSE  INITIAL(12) FIXED BINARY(31);
    DECLARE
       1   PDSGET_REQUEST_1      STATIC,
           2  PDSGET_REQUEST     FIXED BINARY(31),
       1   PDSGET_MEMBER_1       STATIC,
           2  PDSGET_MEMBER      CHAR(8),     /* DDNAME FOR OPEN */
       1   PDSGET_RECORD80_1     STATIC,
           2  PDSGET_RECORD80    CHAR(80),
       1   PDSGET_RETURN_CODE_1  STATIC,
           2  PDSGET_RETURN_CODE FIXED BINARY(31);
/*******************************************************************/
/*                                                                 */
/* PDSGET_RETURN_CODE DEFINITIONS                                  */
/*       RETURN_CODE 0 = SUCCESSFUL COMPLETION OF REQUEST          */
```



```
/*       RETURN_CODE 8 = SERIOUS ERROR.                             */
/*       RETURN_CODE 4 = DEPENDS ON REQUEST CODE                    */
/*            REQUEST_CODE    MEANING                               */
/*                 0          PDS COULD NOT BE OPENED.              */
/*                 1          PDS COULD NOT BE OPENED.              */
/*                 4          MEMBER NOT FOUND.                     */
/*                 5          END OF DIRECTORY.                     */
/*                 8          END OF CURRENT MEMBER.                */
/*                                                                  */
/********************************************************************/
    DECLARE
       PDSGET_PTR   POINTER,
       1  PDSGET_STATISTICS          BASED(PDSGET_PTR),
          2  PDS_C                     CHAR(1),
          2  PDS_VERSION               CHAR(1),
          2  PDS_MOD                   CHAR(1),
          2  PDS_NULL                  CHAR(2),
          2  PDS_DATE_CREATED          FIXED DECIMAL(7,0),
          2  PDS_DATE_UPDATED          FIXED DECIMAL(7,0),
          2  PDS_TIME_CHANGED_H        CHAR(1),
          2  PDS_TIME_CHANGED_M        CHAR(1),
          2  PDS_CURRENT_LINES         FIXED BINARY(15),
          2  PDS_INITIAL_LINES         FIXED BINARY(15),
          2  PDS_CHANGED_LINES         FIXED BINARY(15),
          2  PDS_USER_ID               CHAR(8);

     /*****************************************************************
      *                                                               *
      *   PDS-TIME-CHANGED -H AND -M ARE UNSIGNED PACKED DECIMALS     *
      *   THEY MUST TO CONVERTED TO SIGNED PACK DECIMAL TO USE THEM.  *
      *   THE PDS_TIME_CONVERSION FUNCTION CAN BE USED FOR THAT.      *
      *                                                               *
      *   THIS CONVERSION MUST TAKE PLACE FOR BOTH THE -H AND =M      *
      *   PDS-TIME-CHANGED- FIELDS.                                   *
      *                                                               *
      *****************************************************************/
    DECLARE PDS_TIME_CONVERSION ENTRY RETURNS(CHAR(2));
PDS_TIME_CONVERSION:PROC(A_BYTE) RETURNS(CHAR(2));
    DECLARE A_BYTE             CHAR(1);
    DECLARE
          PDS_FILLER1_PTR      POINTER,
          PDS_FILLER4_PTR      POINTER,

          PDS_TIME_WORK        FIXED DECIMAL(5,0) INITIAL(0),

       1  PDS_FILLER1                 BASED(PDS_FILLER1_PTR),
             10  FILLER2              CHAR(1),
             10  PDS_TIME_CHAR        CHAR(1),
             10  FILLER3              CHAR(1),
       1  PDS_4_DIGIT_TIME            PIC '(4)9',
       1  FILLER4                     BASED(PDS_FILLER4_PTR),
             10  FILLER5              CHAR(1),
             10  PDS_TIME             CHAR(2),
             10  FILLER6              CHAR(1);
```



```
    PDS_FILLER1_PTR = ADDR(PDS_TIME_WORK);
    PDS_FILLER4_PTR = ADDR(PDS_4_DIGIT_TIME);

    PDS_TIME_WORK = 0;
    PDS_TIME_CHAR = A_BYTE;
    PDS_4_DIGIT_TIME = PDS_TIME_WORK;
    RETURN(PDS_TIME);
END PDS_TIME_CONVERSION;
/********************* GETPDSPA V1.2.0  END ************************/
```