

# Using IPCS with ALCS

July 2002

Mike Hannaford

ALCS Support  
Internet: [alcs@uk.ibm.com](mailto:alcs@uk.ibm.com)



---

# Contents

## **IPCS REXX execs and CLISTs** 1

- Introduction 1
- Pre-requisite reading 1
- Using ALTLIB 2
- The Symbol Table 2
- Useful IPCS Commands 2

## **The ALCS IPCS REXX execs** 3

- Screen Errors 3

## **DXCINIT - Setting the symbol DXCNUC** 5

- The ALCS Nucleus is not in the dump 5
- Using the DXCINIT REXX exec to locate the Nucleus 5

## **DXCSYM - Setting other ALCS symbols** 7

## **ALCS Dump Formatting Execs** 13

- DXCWKLST - Format the Dispatcher Task Work Lists 13
- DXCJECT - Show the SVC dump area table 13
- DXCEVCB - Show the save area chains for the dispatcher tasks. 14
- DXCAECB - Find the Type 1 storage units that are in use 15
- DXCAIOB - Find the IOCBs that are in use 15
- DXCCS0DT - Show parts of the system configuration table 15
- DXCCTKB - Show parts of the system keypoint table 15
- DXCTOD - Translate the time of day clock 15
- DXCUAREA - Show unformatted data 15



---

# IPCS REXX execs and CLISTs

---

## Introduction

IPCS is the MVS tool used to analyse machine readable dumps and traces. This document is concerned with the analysis of SVC dumps of ALCS.

There are several events that can lead to the taking of a SVC dump of the ALCS address space.

- The operator can issue a MVS DUMP command to capture the ALCS address space in flight. It is possible for ALCS to continue after the dump is captured. A dump may be taken in this way if it is thought that ALCS is hanging or looping. After dump capture ALCS can be cancelled and restarted by the operator.
- A SLIP trap event occurs against the ALCS address space which requests ACTION=SVCD
- An ALCS catastrophic error occurs and SVCDUMP=YES coded on the SCTGEN (see PTF UQ64211)

The SVC dump process captures the data quickly in an MVS dataspace, it is then written out in the background to the system dump datasets. For the dump to be of use in diagnosing an ALCS problem it must contain the whole of the ALCS private region. Ensure the system dump datasets are large enough to hold this amount of data and the relevant dump parameters are set to capture the private region.

ALCS User Group requirement ATC0119 requests that ALCS provide IPCS routines to allow examination of critical ALCS storage areas within the dump, such as DXCNUC monitor table.

This document describes the IPCS REXX execs that ALCS Support are providing to assist in this area. These execs are not rigorous, nor complete and may require tailoring for your installation. Please contact [alcs@uk.ibm.com](mailto:alcs@uk.ibm.com) for assistance or to request a new exec to do addition formatting.

---

## Pre-requisite reading

Before reading this document you should be familiar with the contents and terminology in the IPCS User's Guide. This manual is distributed on the operating system Collection CD-ROMs and is available for online viewing at the IBM Website. It has had various book numbers and versions over the years including :

- z/OS MVS IPCS User's Guide SA22-7596
- OS/390 MVS IPCS User's Guide GC28-1756
- MVS/ESA IPCS User's Guide GC28-1490, for MVS/ESA Version 5
- MVS/ESA IPCS User's Guide GC28-1631, for MVS/ESA Version 4

Of particular relevance is the chapter "**Using IPCS REXX execs and CLISTs**" which is either Chapter 6 or Chapter 7 depending which edition of the manual you have.

---

## Using ALTLIB

You can use the TSO/E ALTLIB command to define a CLIST library to an IPCS dialog session. IPCS will first search any ALTLIB library before searching other libraries.

For example, suppose you have your ALCS IPCS REXX execs in the dataset DXC.V2R2M1.IPCS. To place DXC.V2R2M1.IPCS ahead of other CLIST libraries, choose the COMMAND option from the IPCS Primary Option Menu and enter the ALTLIB command as follows:

```
==> ALTLIB ACTIVATE APPLICATION(CLIST) DATASET('DXC.V2R2M1.IPCS')
```

An ALTLIB environment is unique for each IPCS dialog screen. If you split the screen and start another IPCS dialog session, the first ALTLIB environment does not apply to both screens. You can define another ALTLIB environment for the second screen.

When you exit the IPCS dialog, the ALTLIB environment for that IPCS session disappears. See the ALTLIB command in **OS/390 MVS IPCS Commands** for more information.

---

## The Symbol Table

For every dump you process IPCS maintains a table of symbols representing data areas in the dump. The symbol table is part of the contents of the dump directory. IPCS uses the table to make future references to a dump faster and more efficient.

When you process a dump for the first time, IPCS initializes the symbol table for the dump. A symbol in IPCS is a label associated with a location in a dump. IPCS automatically creates some symbols for you. You can create your own symbols for areas of the dump using the EQUATE subcommand.

The ALCS IPCS REXX execs will create and use symbols as part of their processing. Use the IPCS command **listsym** to view the symbol table and **dropsym** and **equate** to delete and add symbols.

---

## Useful IPCS Commands

IPCS is rich with formatting capability, some useful IPCS commands are :

<b>summ regs</b>	TCB/PRB with registers
<b>summ format</b>	much more formatting than the above
<b>systrace</b>	System Trace
<b>verbx mtrace</b>	Master Trace - console messages
<b>status faildata</b>	Will format error information if SDWA is found by IPCS

---

## The ALCS IPCS REXX execs

There are several execs to assist you in dump analysis, they are presented here in a non-alphabetic order. The first exec that must be used is DXCINIT in order to set the symbol equate for the ALCS nucleus. The other execs should be able to run (assuming the data is available in the target dump) once the DXCINIT exec has run.

### Screen Errors

The execs format data and send the data to the target output point, typically this is your screen. If you do not scroll down through all the output data before ending the exec (for example, through use of PF3) then you will see a REXX Program interrupted error, eg :

```
292 *-*          "NOTE '"GENφ"' ASIS"  
    +++ RC(12) +++  
292 +++          "NOTE '"GENφ"' ASIS"  
122 +++          call put  
132 +++          call Format_SaveArea nextsave, i, nucsa  
132 +++          call Format_SaveArea nextsave, i, nucsa  
 95 +++          call Format_SaveArea nextsave, i, nucsa  
 42 +++          Call Format_Nucleus_Area nu0sv.i, i  
Error running DXCEVCB, line 292: Program interrupted  
***
```

This is to be expected as the exec is still running, the above just shows where the exec has reached (note the nested function calls). The advantage of working this way is that data is returned to the screen in a timely manner.

When running through a long chain of control blocks an exec may take a long time to complete, so be wary of using maximum scroll down (m PF8). A corrupt chain of control blocks could lead to a loop. If you get no response use Attention and halt the execution of the REXX exec.



---

## DXCINIT - Setting the symbol DXCNUC

The key to locating any ALCS data is to first find the ALCS nucleus. The ALCS nucleus, DXCNUC, contains a data area holding pointers to the ALCS CSECTs and data areas. It is important to find DXCNUC. DXCNUC resides in the ALCS monitor which is loaded into the low end of the ALCS address space. If this area is not in the dump then it likely that the dump will be of no use.

### The ALCS Nucleus is not in the dump

There are several reasons why this might be the case.

The dump may not be of the ALCS address space.

The dump did not request the private area of the ALCS address space. All ALCS data is in the private region of the ALCS address space. ALCS does not use the common areas of the address space directly.

For dumps of ALCS the dumps parameters RGN and TRT are of most importance. RGN is the REGION, the private address space, and TRT is the Trace Table which is helpful in analysing loop conditions.

Another reason for the nucleus not being in the dump is that, although it was requested, it failed to be captured. A common fault is the dump dataset is not big enough and a PARTIAL dump is captured rather than a COMPLETE dump.

If DXCNUC is not in the dump then the ALCS REXX execs will not be of use.

### Using the DXCINIT REXX exec to locate the Nucleus

The aim of the DXCINIT exec is to find the nucleus and to set the symbol DXCNUC to equate to it. This is the only purpose of this exec. If you know the address of DXCNUC in the dump then the command

**EQUATE DXCNUC xxxxx.**

where xxxxx is the hexadecimal address of the ALCS Nucleus will have the same effect as the successful running of the DXCINIT exec.

Exec completes normally with :

```
LIST 00068638 ASID(X'0031') LENGTH(X'06') CHARACTER  
ASID(X'0031') ADDRESS(068638.) KEY(70)
```



## DXCSYM - Setting other ALCS symbols

Once the symbol for the nucleus has been set, we can set up labels for the fields in the monitor interface area (the data area part of DXCNUC). The DXCSYM exec will do this. Use the command **LISTSYM** to see all the new symbols.

Symbols are set for the configuration tables (DXCCDS, DXCCDC, DXCCDD, DXCCDT) and for all the ALCS Monitor CSECTS (eg. DXCINT).

Symbols are also set for most of the control blocks and tables pointed to by the monitor interface area. These include :

Offset	Symbol	Description
0030	CTKBADR	CTKB
0034	VFACADR	VFA CONTROL AREA
0038	AGFSCTB	FILE POOL CONTROL TABLE
003C	ASYSID	ALCS NAME AREA
0040	ACPOSTI	SYSTEM STATE INDICATORS
0044	SCLKADR	ALCS SECOND CLOCK
0048	AECBCNT	ECB COUNTERS
004C	AECBCNTU	ECB COUNT UPDATE TABLE
0050	DXCMONPS	START OF ONLINE MONITOR PROGRAM
0054	DXCMONPE	END OF ONLINE MONITOR PROGRAM
0058	DXCCPTAS	START OF MONITOR TABLES AREA
005C	DXCCPTAE	END OF MONITOR TABLES AREA
0060	ADXCSAVE	ADDRESS OF MONITOR WORK AREAS
0060	DXCSAVES	START OF MONITOR WORK AREAS
0064	DXCSAVEE	END OF MONITOR WORK AREAS
0068	ASRBSAVE	ADDRESS OF SRB SAVE AREA
006C	ASRBFLAG	ADDRESS OF SRB SAVE INDEX
0070	ASRBCNT	NUMBER OF SRB SAVE AREAS
0074	AENFEXIT	ENF LISTENER USER EXIT
0078	ENFADR	ADDRESS OF SQA AREA FOR ENF
007C	ENFLEN	LENGTH OF SQA AREA FOR ENF
0088	ADXCCDS	SYSTEM TABLES
008C	ADXCCDC	COMMUNICATIONS TABLES
0090	ADXCCDD	DASD TABLES

Offset	Symbol	Description
0094	ADXCCDT	TAPE (SEQUENTIAL FILE) TABLES
0098	ADXCCDS0	RESERVED
009C	ADXCCDS1	PROGRAM CONFIGURATION TABLE
00A0	ADXCCDS2	COMMS CONFIGURATION TABLE
0340	AKPTRTN	KEYPOINT UPDATE ROUTINE
0344	ACLKRTN	CLOCK UPDATE ROUTINE
0348	ACRTSVC	CRET SERVICE ROUTINE
034C	ASLCPOL	SLC BUFFER POLICE ROUTINE
0354	ASIMRTN	TIME INITIATED SIMLOGON ROUTINE
0358	AHSOPZRO	HIGH SPEED LINE OPZERO
035C	ALSOPZRO	LOW SPEED LINE OPZERO
0360	ACOMACT0	COMMS ACTION TABLE PROC
0364	AXPOPZRO	X25 OPZERO
0368	AMROPZRO	MESSAGE ROUTER OPZERO
036C	APOOLRNG	POOL RING MANAGEMENT ROUTINE
03E8	AIOBLST	IOCB LIST
03EC	APFQLST	PAGE FAULT LIST
03F0	ARDYLST	READY LIST
03F4	ADLYLST	DELAY LIST
03F8	AINPLST	INPUT LIST
03FC	ADFRLST	DEFER LIST
0400	AIOBLST2	DEFERRED IOCB LIST
0418	ANOTQUE	NOT ECB QUEUE NBR=0
041C	ARUNQUE	RUNNING QUEUE NBR=1
0420	AGETQUE	GETECB QUEUE NBR=2
0424	ARECQUE	RECEIVE QUEUE NBR=3
0428	AVFAQUE	VFA WAIT QUEUE NBR=8
042C	AHLDQUE	HOLD WAIT QUEUE NBR=9
0430	ATRCQUE	TRACE WAIT QUEUE NBR=10
0434	ASEQQUE	SEQFILE WAIT QUEUE NBR=11
0438	AHKAQUE	UNHKA WAIT QUEUE NBR=12
043C	AVFHQUE	VFA HLD WAIT QUEUE NBR=13
0440	ASQLQUE	SQL WAIT QUEUE NBR=14
0444	AAPCQUE	APPC WAIT QUEUE NBR=15

<b>Offset</b>	<b>Symbol</b>	<b>Description</b>
0448	AMQIQUE	MQI WAIT QUEUE NBR=16
044C	ASLCQUE	SLC WAIT QUEUE NBR=17
0464	REQSWA	FIRST REQUEST SWITCH ADDRESS
0464	AKPTREQ	KEYPOINT UPDATE REQUEST SWITCH
0468	ACPUTMR	CLOCK UPDATE REQUEST SWITCH
046C	ADLYREQ	DELAY LIST SERVICE REQUEST SWITCH
0470	ADFRREQ	DEFRC LIST SERVICE REQUEST SWITCH
0474	ACRTREQ	CRET SERVICE REQUEST SWITCH
0478	AKPLREQ	SLC POLICE SERVICE REQUEST SWITCH
047C	ATASREQ	TAS SERVICE REQUEST SWITCH
0480	ATMEAVL	TAS REQUEST SWITCH
0484	ADPFREQ	DYNAMIC PAGE FIX REQUEST SWITCH
0488	AVFTREQ	VFA TIME INITIATED FILE SWITCH
048C	ASIMREQ	SIMLOGON REQUEST SWITCH
0490	ACT0REQ	COMMS ACTION TABLE 0 REQUEST
0494	ACT1REQ	COMMS ACTION TABLE 1 REQUEST
04AC	APURREQ	ENTRY PURGE (PURGC MACRO)
04B0	AGTFREQ	GTF REQUEST SWITCH
04B4	ASTVREQ	STV INTERCEPT REQUEST SWITCH
04B8	ADCRREQ	DCR INTERCEPT REQUEST SWITCH
04BC	AKLTREQ	SLC LINK TRACE INTERCEPT SWITCH
04C0	AMTBREQ	MACRO TRACE BLOCK REQUEST SWITCH
04C4	AQUIREQ	QUIESCE REQUEST SWITCH
04C8	AXCFREQ	XCF TABLE UPDATE
04CC	AGFSREQ	POOL SERVICE
04E4	AHOLDCNT	HOLD COUNTER
04E8	AWRITCNT	WRITE COUNTER
050C	ATRCPTR	MACRO TRACE BLOCK POINTERS
0510	ADUPDMP	DUPLICATE DUMP TABLE
0514	ACRTLST	CRET ITEM LIST HEADERS
0518	AEVCBLST	EVENT CONTROL BLOCK LIST
051C	ACP0DA	DIRECTLY ADDRESSED TABLE
0520	AINTTCB	INITIALIZER TCB
0524	ACH0CH	RESOURCE HOLD TABLE

<b>Offset</b>	<b>Symbol</b>	<b>Description</b>
0528	AOLDECBT	OLDEST ECB CREATE TIME TOD
052C	APRDTB	BLOCK/RECORD DESCRIPTOR TABLE
0530	ASLITB	SYSTEM LOAD INFORMATION TABLE
0534	ALIMTAB	ENTRY LIMITS TABLE
0538	AQUICTL	QUIESCE CONTROL AREA
053C	AACCESS	SERIALIZED ACCESS CONTROL AREA
0540	ADBCTAB	DATA BASE DS CONTROL TABLE
0544	ATESTACB	TEST DATA BASE DATA SET ACB
0548	AHKAECB	ANCHOR FOR UNHKA/REHKA CHAIN
054C	AKPTTAB	ADDRESS OF KEYPOINT TABLES
0550	ADXCXCFMT	XCF MEMBER TABLE
0554	APARTTAB	TEST DB POOL PARTITION TABLE
0558	ADXCXCFAT	PLEX APPLICATION CTL TABLE
0580	ADCOPTS	DATA COLLECTION OPTIONS
0584	ADMPOPTS	DUMP OPTIONS
0588	ATIMWORK	TIMER ROUTINES WORK AREA
058C	ASLCMISC	SLC MISCELLANEOUS AREA
0590	AGTFCTL	MACRO TRACE (GTF) CONTROL AREA
0594	ASQLAREA	DB2/SQL DATA AREA
0598	APPCAREA	APPC/MVS (LU6.2) DATA AREA
059C	AMQIAREA	MESSAGE QUEUEING DATA AREA
05A0	ASCRLLOG	MAX NUMBER OF SCROLL LOG LINES
05A4	ATCPAREA	TCP/IP DATA AREA
05A8	APDUAREA	PDU FACILITY CONTROL AREA
05B0	ADFTAB0	TPFDF STATIC FASTLINK HDR
05B4	ADFTAB1	TPFDF STATIC FASTLINK SLOTS
05B8	CRELSTIO	IOCB LIST POINTER
05BC	CRELSTSU	STORAGE UNIT LIST POINTER
05C0	CRELSTS2	TYPE 2 STORAGE UNIT LIST PTR
05CC	EXTLIST	MONITOR USER-EXIT TABLE
05D0	EXTSERV	MONITOR CALLABLE SERVICES
05E0	PGMCDP	SYSTEM PROGRAM TABLE ADDRESS
05E4	PGMPNT	PROGRAM NAME TABLE ADDRESS
05E8	PGMPHT	PROGRAM HASH TABLE ADDRESS

<b>Offset</b>	<b>Symbol</b>	<b>Description</b>
05EC	PGMPHTN	NUMBER OF ENTRIES IN HASH TABLE
05F0	PGMMNT	MODULE LOAD TABLE ADDRESS
05F4	PGMMNTN	MODULE LOAD TABLE ENTRY COUNT
05F8	PLIBDCB	DCB FOR PROGRAM LIBRARY DXCPLIB
05FC	CLIBDCB	DCB FOR CONFIG LIBRARY DXCCLIB
0600	HLIBDCB	DCB FOR HLL LIBRARY ROUTINES
0628	TCBCOUNT	MAXIMUM CPU LOOP TCB COUNT
062C	PGSIZE	VIRTUAL STORAGE PAGE SIZE
0634	PGFIXD	AMOUNT OF STORAGE PAGE FIXED
0638	PGFIXM	MAXIMUM STORAGE FOR PAGE FIX
063C	SUSIZE	STORAGE UNIT (SU) SIZE IN BYTES
0640	SUSIZE2	TYPE 2 SU SIZE IN BYTES
068C	PCNBENB	ENABLE PKM FOR TABLES ACCESS
0690	PCNBDIS	DISABLE PKM FOR TABLES ACCESS
0694	PCNBVAL	VALIDATE STORAGE REFERENCE
0698	PCNBCSK	CHANGE STORAGE KEY
069C	PCNBSUP	ENTER/LEAVE SUPERVISOR STATE
06A0	PCNBAPF	TURN OFF APF AUTHORIZATION
06A4	PCNBINT	INITIALIZE PKM AT START OF TASK



---

# ALCS Dump Formatting Execs

---

## DXCWKLST - Format the Dispatcher Task Work Lists

This exec runs through the work lists for ECBs and IOCBs showing the list header and a one line entry of each item on a work list.

```
ECB count, active:    0003, delay: 00A2

ENTRY DISPATCHER WORK LIST -- IOCB LIST
0C700008 TOP 0C716AF0 00000000 CDS BOT 0C71E7B0 00000000

DESC CHAIN IOCB TYPE Flag Dispense Time ECB
0C716AF0 0C704290 7F636A00 DASD ALLOCATE IOCB 00 0000000000000000 00000000
...

ENTRY DISPATCHER WORK LIST -- READY LIST
0C700020 TOP 0C724700 FD47009C CDS BOT 0C723640 00040004

DESC CHAIN DEADLOCK ECB CRI REC HOLD RES HOLD SEQ FILE LAST MACRO STATUS Dispense Time
0C724700 0C723640 00000000 0CF795A0 0246FF 00000002 00000000 00000000 SBT9 FINWC F032 IN USE B59875710E757E80
...
```

The fields are taken from :

IOCB DISPATCHER WORK LIST SUMMARY RECORD			ECB DISPATCHER WORK LIST SUMMARY RECORD		
DESC	IOCB BLOCK DESCRIPTOR		DESC	ECB BLOCK DESCRIPTOR	
CHAIN	FORWARD CHAIN	LS0PCH	CHAIN	FORWARD CHAIN	LS0PCH
IOCB	IOCB ADDRESS		DEADLOCK	DEADLOCK DETECT CHAIN	CE3DLK
TYPE	IOCB TYPE	IO0TYP	ECB	ECB ADDRESS	
Flag	IOCB FLAGS	IO0FLG	CRI	COMMS RESOURCE ID	CE3CRI
Dispense Time	DISPENSE TIME	LS0TOD	REC HOLD	RECORD HOLD COUNT	CE3RHN
ECB	ECB ADDRESS	IO0ECB	RES HOLD	RESOURCE FOLD COUNT	CE3CHN
			SEQ FILE	SEQ FILE COUNT	CE3SEQ
			LAST	PROGRAM NAME	CE2MTEP
			MACRO	MACRO REQUEST CODE	CE2MTER
				MACRO LISTING ADDRESS	CE2MTEL
			STATUS	USAGE INDICATORS	LS0PFL
			Dispense Time	DISPENSE TIME	LS0TOD

---

## DXCVECT - Show the SVC dump area table

A SVCDUMP is taken in two parts. The first part is restricted in size, but is captured with the processor disabled.

When initialising a dump in IPCS you may request to use summary dump data, if you do the disabled dump data will be viewable. This data is a truer reflection of what was in storage when the dump occurred. Data captured later in non-disabled mode may have been changed by other entries running in parallel to the dump process.

If the dump is a result of an ALCS catastrophic error and SVCDUMP=YES coded on the SCTGEN, then ALCS requests areas to be dumped as part of the disabled capture. ALCS builds a vector table to hold the addresses of the data to be captured, along with other important addresses, such as the interrupt work area (IROWA).

Depending on the circumstances at the time of the dump it may not be possible to capture any data in disabled mode in which case the vector table will contain zeroes

The DXCVECT exec uses the ALCS dump vector table to find the IROWA, it is not dependent on the DXCINIT exec, it shows the areas captured as part of the disabled dump.

NU0SV 000BBC00 PSATOLD 0096E0C8

IROWA 000BD2E0

Msg 06.27.46 SE-NODUMP OPR-123456 PROG-TSTR OFFSET-0082 CRN-FRCS0B3B

R00-07 0F857868 0F858648 00000000 00000000 0F85866B 00000000 0000000E 0F857868  
 R08-15 7F2E6018 0F8575A0 80068E22 000684E0 0F701658 000BBC00 7EF7B43E 0F857868  
 PSW 078D0000 FF2E6082 IC 0001 IL 0006

FP0-FP6 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000  
 SDWA 00926918 0F8575A0 ECB BDA 0F706138 Options 7100E0F6 UserList 00000000

Current ALCS dump vector table at 7F140000

Block	Start	End
VECT	7F140000	7F140168
CTKB	7F672E68	7F673288
TBL	0F700000	8F717100

Unformatted area at 7F140000, length 00000200

```

7F140000 000 E2E5C3C1 D9C5C140 0000A000 000A9000 0096E0C8 000BD2E0 00926918 00000000
7F140020 020 00000000 00000000 E5C5C3E3 7F1400C8 C3E3D2C2 7F1400D0 E3C2D340 7F1400D8
7F140040 040 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140060 060 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140080 080 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F1400A0 0A0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F1400C0 0C0 00000000 00000000 7F140000 7F140168 7F672E68 7F673288 0F700000 8F717100
7F1400E0 0E0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140100 100 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140120 120 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140140 140 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140160 160 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F140180 180 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F1401A0 1A0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F1401C0 1C0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
7F1401E0 1E0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
  
```

---

## DXCEVCB - Show the save area chains for the dispatcher tasks.

This exec finds the event control block list for the ALCS dispatcher tasks (the CPU Loop tasks). It also finds the Nucleus save areas for these tasks.

The first line it displays is the list of MVS EvCBs. ALCS can have several CPU Loop tasks. Each has an associated EvCB in this list with the last one in the list having the high order bit turned on.

The exec displays the save area chain for each CPU Loop task.

---

## **DXCAECB - Find the Type 1 storage units that are in use**

The exec scans the block list descriptors for Type 1 storage units. It displays a summary of the ones that are in use and for ECBs it shows information relating to the status of the ECB.

It also summarises the block usage indicator field.

---

## **DXCAIOB - Find the IOCBs that are in use**

The exec scans the block list descriptors for IOCBs. It displays a summary of the ones that are in use.

---

## **DXCCS0DT - Show parts of the system configuration table**

The exec displays part of the contents of the ALCS system configuration table.

---

## **DXCCTKB - Show parts of the system keypoint table**

The exec displays part of the contents of the ALCS system keypoint table.

---

## **DXCTOD - Translate the time of day clock**

The exec is called by some of the other execs to translate the MVS time of day (TOD) clock. It can be invoked from ipcs directly, for example **dxctod b7db0edc display** will return :

```
11:00:00 30 Jun 2002
```

---

## **DXCUAREA - Show unformatted data**

This exec is called by some of the other execs to show storage. It displays 32 bytes of data on each line, preceded by the data address and 3-character offset. It can be invoked from ipcs directly, for example **dxcuarea 7f140000 200** will display the requested area. The output would look like that in the DXCVECT example shown previously.