CP

SERVICE

ROUTINES

Parameter Value Conversions/Extractions - D M K C V T

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K C V T	D M K C V T B H	Convert Binary Data to Printable HEX	R1=Binary Data R14=Return Addr	RO,R1=EBCDIC	N/A
	D M K C V T H B	Convert HEX Data To Binary Format	RO=Data Length R1=Addr of Data R14=Return Addr	R1=Binary Data	CC Settings: O=Convert OK 1=Error
	D M K C V T D B	Convert Decimal Field To Binary Data	RO=Data Length R1=Addr of Data R14=Return Addr	R1=Contains Binary Data	CC Settings: O=Convert OK 1=Error
	D M K C V T B D	Convert Binary Field To Dblwrd Decimal	R1=Contains Binary Data	RO,R1=Contain the Decimal Value	N/A
	D M K C V T D T	Convert Date/Time To EBCDIC Data Format	R1=Addr of Buffer To Contain DATE Value R2=Addr of Buffer to	MM:DD:YY HH:MM:SS	If a Clock Error Occurs Then a CVT001 ABEND

NOTE: Condition Codes Differ Between Entry Point Functions

Parameter Value Conversions/Extractions - D M K S C N

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K S C N	D M K S C N R U	Locate CP Real I/O Blocks for given Real Device Addr	R1=Addr (CUU) Format R14=Return Addr	R6=RCHBLOK Addr R7=RCUBLOK Addr R8=RDEVBLOK Addr	CC Settings: 0=All Blocks OK 1=Not All Found
	D M K S C N V U	Locate CP Virtual I/O Blocks for given Virt Device Addr	R1=Addr (CUU) Format R11=VMBLOK Addr R14=Return Addr	R6=VCHBLOK Addr R7=VCUBLOK Addr R8=VDEVBLOK Addr	CC Settings: 0=All Blocks OK 1=Not all Found
	D M K S C N D C	Locate RDEVBLOK Addr For Specific SYSOWND Index Value	R0=CCPD Value R14=Return Addr	R8=RDEVBLOK Addr	0=Block Found OK 1=Not found...
	D M K S C N V S	Locate RDEVBLOK Addr For Specific VOLSER Value	R0=Length of Field R1=Addr of VOLSER R14=Return Addr	R1=RDEVBLOK Addr	0=Block Found OK 1=Not found...
	D M K S C N R D	Compute Device-Addr (CUU) from RDEVBLOK Addr	R8=RDEVBLOK Addr R14=Return Addr	R1=Contains (CUU) value	0=Convert OK 1=Bad Convert
	D M K S C N V D	Compute Device-Addr (CUU) from VDEVBLOK Addr	R8=VDEVBLOK Addr R14=Return Addr	R1=Contains (CUU) Value	0=Convert OK 1=Bad Convert
	D M K S C N A U	Locate VMBLOK Addr For Specific USERID Value	R0=Length of Field R1=Addr of USERID R14=Return Addr	R1=VMBLOK Addr or R1 Unpredicatable (See CC Settings)	CC Settings: 0=VMBLOK found 1=Not found; Reg-1 not valid

NOTE: Condition Codes Differ Between Entry Point Functions

Parameter Value Conversions/Extractions - D M K S C N

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K S C N	D M K S C N R N	Return Full Device Name from RDEVBLOK	R8=RDEVBLOK Addr R14=Return Addr	R1=Contains the Device Name.. (i.e. DASD,TAPE..)	CC Settings: 0=RDEVBLOK found
	D M K S C N V N	Return Full Device Name from VDEVBLOK	R8=VDEVBLOK Addr R11=VMBLOK Addr R14=Return Addr	(Same as Above)	CC Settings: (Same as Above)
	D M K S C N F D	Locate Next Non-Blank Field in a Command Input Buffer	R9=Addr of 18-Dblwrd Area to Search R14=Return Addr	R0=Length of Field R1=Addr of Field	CC Settings: 0=Data Found 2=End of Buffer Reached
NOTE: This Entry Pont was removed in SP3 Release.	D M K S C N L I	Search VMBLOKs for Links to MINIDISK (Not Valid for VM/SP3 Release).	R1=RDEVBLOK Addr R2=VDEVBLOK Addr R11=VMBLOK Addr R14=Return Addr	R0=No of R/O Links R1=No of R/W Links R3=VMBLOK Addr for First R/O User	CC Settings: 0=No Links Found 1=R/O Links Fnd 2=R/W Links Fnd

NOTE: Condition Codes Differ Between Entry Point Functions

Parameter Value Conversions/Extractions - D M K S C O

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K S C O	D M K S C O L I	Search VMBLOKs for Links to MINIDISK (Entry Point Valid for VM/SP3 Only)	R1=RDEVBLOK Addr R2=VDEVBLOK Addr R11=VMBLOK Addr R14=Return Addr	R0=No of R/O Links R1=No of R/W Links R3=VMBLOK Addr for First R/O User	CC Settings: 0=No Links Found 1=R/O Links Fnd 2=R/W Links Fnd
	D M K S C O N P	Find the Next Logical Path to a Device	R6=RCHBLOK Addr R7=RCUBLOK Addr R8=RDEVBLOK Addr	R6=Next RCHBLOK R7=Next RCOBLOK	CC Settings: 0=R6 & R7 Valid 1=No Path Found

NOTE: Condition Codes Differ Between Entry Point Functions

Paging Manager Interface Routines - D M K P T R

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K P T R	D M K P T R A N	Preform Virtual-to-Real Addr Translation (Including PAGIN Req)	"TRANS" Macro Usage Rx=Virt; Ry=Real Addrs (See Macros)	Ry=Real Addr after Translation or PAGIN Request..	CC Settings: 0=Page Resident 2=Xlation Error
	D M K P T R F R	Allocate a Real Page Frame from FREELIST	R1=0 R11=VMBLOK Addr SVC 8 CALL Linkage	R7=Addr of CORTABLE Entry for Page	-
	D M K P T R F T	Place Real Page Frame (CORTABLE Entry) on FREELIST	R7=Addr of CORTABLE Entry SVC 8 CALL Linkage	None	-
	D M K P T R R S	Reset All Virtual Storage Pages for User	R11=Addr of VMBLOK for User to Reset SVC 8 CALL Linkage	All Pages Placed on FLUSHLIST with all Invalid bits on...	Normal Call is From DMKSCHDL (Q-Drop Entry)
	D M K P T R L K	Lock a Real Page Frame (Increase Lock count Field ...)	R2=Real Page Addr R14=Return Addr	Real Page Frame has been Locked..	N/A
	D M K P T R U L	Unlock a Real Page Frame (Decrement the Lock Count Field...)	R2=Real Page Addr R14=Return Addr	Real Page Frame has been Unlocked..	PTR003 Abend if Real Page Frame Not Locked....

NOTE: Condition Codes Differ Between Entry Point Functions

Request High Free CP Storage - D M K F R E

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K F R E	D M K F R E	Request Free Storage (Unconditional...)	R0=Length of Storage Requested R14=Return Addr	R0=Same R1=Addr of High Free Area	-
	D M K F R E R C	Request Free Storage (Conditional Request)	R0=Length of STorage Requested R14=Return Addr	R1=Addr of High Free Area (See CC's)	CC Settings 0=Area Obtained 1=Area Not Avail
	D M K F R E T	Release Area of High Free Storage (Uncoditional....)	R0=Length of Area to be Released. R1=Addr of Area to be Released	High Free Storage Block is Released.	FREnnn ABEND: If either Len or Addr Fields are Incorrect.

NOTE: Condition Codes Differ Between Entry Point Functions

Error Message Edit & Processing - D M K E R M

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K E R M	D M K E R M S G	Construct/Edit CP Error Message from Input Parameters	(SVC 8 CALL Linkage) See Below:	Same as Input (Error Message Sent According to Input)	-

R0 = XXXXXXnn

XXXXXX = Three Character Module-ID of Caller
nn = If Not Zero, then used as Length of Data.
= If Zero, then Reg-1 will Contain the Data itself.

R1 = Either Contains the Addr of the Data to be Inserted OR Data Itself

R2 = YY ZZ MMMM

YY = Option Byte: X'80' - Return to Caller (Otherwise Issue SVC 16...)
X'40' - Caller Area will be Released
X'20' - Send to 'Operator' Userid
X'10' - Sound Alarm with Message
X'08' - Delete ERRMSG Parm Field

ZZ = Action Character: I - Information Msg
A - Action Msg
R - Response Needed
W - Warning Msg
S - Severe System Error

MMMM = Message Number (Binary Value)

R3 = LLaaaaaa

LL = Length of Area to be Released
aaaaaa = Addr of Area to be Released

R11 = Addr of VMBLOK (User will Receive Error Message)

Initiate Console Write Routine - D M K Q C N

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K Q C N	D M K Q C N W T	Construct & Initiate a Console Write Req Using Input Parms...	(SVC 8 CALL Linkage) See Below:	Same as Input (Console Msg Sent According to Input)	-

Basic Input Regs: R0 = No of Bytes in Message.
R1 = Address of Message
R2 = "CALL" Parameters
R3 = The Number of Double-Words to 'FRET' if ..'DFRET' Option was used; or Number of Imbedded Strt Fields-Diag-58 Only.
R4 = Virt Device Addr (only if "VIRDEVAD" is Used)

Reg-2 Parameters:

|Byte-0|Byte-1|Byte-2|Byte-3|

Byte-0:
- Flag & Line No. For Diag-58 Req

Byte-1:
- IMSG-X'800000' = The Message to be Sent is an Information Type.

Byte-3:
- LOGDROP-X'80' = Logoff User & Drop Line
- LOGHOLD-X'40' = Logoff User & Hold LIne
- PRIORITY-X'20' = Priority Message
- VMGENIO-X'10' = VM Generated I/O
- INHIBIT-X'08' = Don't Write Console Data
- NOAUTO-X'04' = Suppress Auto CR
- ALARM-X'02' = Sound Console Alarm
- NOTIME-X'01' = Suppress Time Stamp on Msg

Byte-2:
- HILIGHT-X'8000' = Highlight Msg on 3270 Device.
- NOTRESP-X'4000' = Write Response to User's Virt Buffer.
- SECUSER-X'2000' = Msg Sent From Disconnected User
- VIRDEVAD-X'1000' = Remote Virt Dev (R4=Addr)
- ERRMSG-X'0800' = Error Msg (Use VMMLEVEL Values)
- NORET-X'0400' = Return Immediately (No Wait)
- DFRET-X'0200' = "FRET" Message Buffer After Write
- OPERATOR-X'0100' = Send Message to Sys OPERATOR

Initiate Console Read Routine - D M K Q C O

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K Q C O	D M K Q C O R D	Construct & Initiate a Console Read Req Using Input Parms...	(SVC 8 CALL Linkage) See Below:	Same as Input (Console Read Req Based Upon Input...)	-

```
Basic Input Regs:  R0 = Maximum Number of Bytes of Input Expected
                   R1 = Address of Input Buffer
                   R2 = "CALL" Parameters(See Below)
                   R4 = Virt Device Addr (only if "VIRDEVAD" is Used)

Reg-2 Parameters:

          |Byte-0|Byte-1|Byte-2|Byte-3|
             |      |      |      |            +-----------+
+--------+   |      |      |      +-----> Byte-3 | Not Used |
|Not Used|<--+      |      |                   +-----------+
+--------+          |      |
                    |      |             +---------------------------------------+
                    |      |             | EDIT-X'0004'  = Edit Line Based on Logical
                    |      |             |                 Editing Characters
                    |      |             | UCASE-X'0002' = Translate all Characters
                    |      +-----------> |                 to Upper Case.
                    |                    | VMGENIO-X'0010' = VM Generated I/O Req
                    |                    | INHIBIT-X'0008' = Do Not Display Read
                    |                    |                   Data or Spooled.
                    v                    | VIRDEVAD-X'1000' = Remote Dev Processing
+--------------------------+             | WRTREADD-X'0001' = Write Chained to Read
|                          |             |                    For TTY Processing.
| Byte-1 Will Hold CCW OP- |             +---------------------------------------+
| Code for Full Screen     |
| Requests.                |
|                          |
+--------------------------+
```

Schedule & Reset CP Timer Requests - D M K S C H

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K S C H	D M K S C H S T	Establish a Clock Comparator Request	R1=Address of TRQBLOK for Timer Request.	R1=Same (TRQBLOK Placed on TRQBLOK Chain..)	N/A
	D M K S C H R T	Delete a TRQBLOK from the TRQBLOK Request Chain (Reset it...)	R1=Address of TRQBLOK to be deleted.	R1=Same (TRQBLOK Deleted from Chain...)	NOTE: No error checking is done by DMKSCHRT.

Allocate CP Virtual Storage Page - D M K P G U

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K P G U	D M K P G U V G	Allocate a CP Virtual Storage Page within CP Virtual Storage.	R14=Return Address	R1=Addr of Virtual Page available	ABEND PGU005 if no virtual buffers avail.
	D M K P G U V R	Release a CP Virtual Storage Page within CP Virtual Storage.	R1=Addr of Virt Stor Page. R14=Return Address	Virt Storage Page will be deallocated	ABEND PGU006 if virt page not allocated.

Stack CPEXBLOK or IOBLOK or TRQBLOK	- D M K S T K

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K S T K	D M K S T K C P	Stack CPEXBLOK on the Dispatcher's Request Queue.	R1=Addr of CPEXBLOK to be Stacked R14=Return Addr	CPEXBLOK Stacked	-
	D M K S T K I O	Stack either IOBLOK or TRQBLOK on the Dispatcher's Queue.	R1=Addr of IOBLOK or TRQBLOK R14=Return Addr	IOBLOK/TRQBLOK Stacked	-

Entry Points (within DMKSTK) Used Only for AP/MP Support:

	D M K S T K M P	Stack CPEXBLOK on the Current Processor's Queue.	R1=Addr of CPEXBLOK to be Stacked R14=Return Addr	CPEXBLOK Stacked	AP/MP Only
	D M K S T K O P	Stack CPEXBLOK on the Other Processor's Dispatcher's Queue.	R1=Addr of CPEXBLOK to be Stacked R14=Return Addr	CPEXBLOK Stacked	AP/MP Only
	D M K S T K L F	Stack CPEXBLOK LIFO for a Specific Processor's Queue.	R1=Addr of CPEXBLOK to be Stacked R14=Return Addr	CPEXBLOK Stacked	AP/MP Only

CP Directory Management Routines - D M K U D R

C P Module	Entry Point(s)	Function(s)	Input Req'ts	Output Values	Comments
D M K U D R	D M K U D R F U	Locate the Specific Directory Blcok for User (UDIRBLOK)	RO=Length of USERID R1=Addr of USERID R2=Addr of Buffer to Receive Block	R2=Addr of UDIRBLOK	CC Settings: O=User Found 1=Not Found 3=I/O Error
	D M K U D R M D	Read User's Directory Account Block (UMACBLOK) (UMACBLOK)	R1=Addr of Disp and DASD ADDR (CCPD) R2=Addr of Buffer For UMACBLOK	R2=Addr of UMACBLOK	CC Settings O=Block Read 1=Invalid Data 3=I/O Error
	D M K U D R F D	Read a Specific Dev Block From User's Directory. (UDEVBLOK)	RO=Contains Dev Addr R1=Addr of Disp and DASD ADDR (CCPD) R2=Addr of Buffer	R2=Addr of UDEVBLOK	CC Settings: O=Device Found 1=Not Found 3=I/O Error
	D M K U D R X I	Read User's IPL Block from Directory. (UIPLBLOK)	R1=Addr of Disp and DASD Addr (CCPD) R2=Addr of BUffer	R2=Addr of UIPLBLOK	CC Settings: O=Device Found 1=Invalid Data 3=I/O Error
	D M K U D R R D	Read Next Directory Block for User: Either - UMACBLOK or UDEVBLOK or UIPLBLOK	R1=Addr of Disp and DASD Addr (CCPD) R2=Addr of BUffer	R2=Addr of Block UMACBLOK or UDEVBLOK or UIPLBLOK	CC Settings: O=Block Read 1=Invalid Data 3=I/O Error
	D M K U D R R V	Release Virtual Page Used by Directory Routines as Buffer	R2=Addr of Page Buffer	None	CC Settings: O=Page Buffer Released

NOTE: Condition Codes Differ Between Entry Point Functions

CP Linkage & Exit Macros - MACROS = CALL, GOTO, RELOC and EXIT

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
C A L L	Used to Pass Control Another CP Module	CALL MODULE,PARM=Field1+Field2... - L R15,=A(MODULE) LA R2,Field1 + Field2 SVC 8 or L R15,AMODULE LA R2,Field1 + Field2 BALR R14,R15	MODULE = Name of Specific CP Module that is to be Called AMODULE = The Entry Point address of the mod- defined within the resident portion of CP. Fieldx = Any Term That Has Been Previously Defined (i.e..COPY EQU...). CALL Macro may be used to invoke either a CP Resident or Pagable Module. An Imbedded Table within the CALL Macro Determines the Correct Type of Call Sequence; either SVC or BALR Linkage..(See Example...). BALR Linkage Implies that the Module is Resident AND an its Entry Point Defined Within PSA.
G O T O	Used to Pass Control Without an Implied Return (No Return)	GOTO MODULE - L R12,AMODULE BR R12	MODULE = The Specific Name of the CP Module (CP Resident Module). The GOTO Macro is used to pass control to CP Resident Module by using R12 as its Base Register. The Address of the CP Routine must Defined within the PSA. Module using the GOTO Macro will not expect to receive control back from Module that was invoked.

CP Linkage & Exit Macros - MACROS = RELOC and EXIT

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
R E L O C	Used to Establish Addressability and to Save Registers in SAVAREA Block (SVC)	DMKXYZxx RELOC - ENTRY DMKXYZxx DMKXYZxx DS OH USING *,R12 STM RO,R11,SAVEREGS SL R12,=A(DMKXYZxx-DMKXYZ) USING DMKXYZ,R12	DMKXYZxx = Represents the Name Field on the Macro Line (Must be Specified..) The RELOC Macro is used by each Entry point within a CP Module (Resident or Pagable) if the Module will receive control via an SVC Linkage (i.e...CALL Macro). The RELOC Macro establishes Base Reg-12 for the entire module as well as the specific Entry Point. In addition, the RELOC Macro will save Reg-0 -thru- Reg-11 within the standard SVC SAVEAREA Block (Constructed by DMKSVC).
E X I T	Used to Return Control to a Previous CP Module.	EXIT - LM RO,R11,SAVEREGS SVC 12	EXIT Macro is used to pass control back to a previous CP Module (who called this Module) Reg-0 -thru- Reg-11 are restored to their previous values (when the RELOC Macro was used), and an SVC 12 is used. A Condition Code value can be set prior to the EXIT Macro invocation (See SETCC Macro); in which case it will be preserved accross the SVC call, and can be examined by the CP Module that will be given control.

CP Interface With Paging Manager - MACRO = TRANS

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
T R A N S	Used to Invoke the Paging Manager (DMKPTRAN for Address Translations.	TRANS Rx,Ry,OPT=(Options..), AFF=,ADEX=,IOER=, LCTL= Options = BRING (Bring the Page in) LOCK (Lock Page Frame) DEFER (Control will not Returned until page is Available). SYSTEM (Use the SYSTEM VMBLOK for Address Translation). IOERETN (I/O Error Exit has been provided TRANS R2,R1,OPT=(BRING,DEFER) LCTL C1,C1,VMSEG LRA R2,0(0,R1) BZ A LA R2,BRING+DEFER L R15,APTRAN SVC 8 A DS OH	Options = Input Values used by DMKPTRAN for Address Translations. (See Descriptions....) AFF = Processor Affinity (AP/MP Only) ADEX = Address Exception Exit (Address..) IOER = Paging I/O Error EXIT (Address..) LCTL = 'NO' - No LCTL Instr Used. The TRANS Macro is used to affect Virtual -to-Real address translation, and if necessary to invoke the Real Paging Manager (DMKPTRAN) to resolve the condition. Based upon the OPTIONS supplied, DMKPTRAN will pagin the page (BRING) lock it (LOCK) and will not return to the caller until completed (DEFER). If SYSTEM is specified, then the SYSTEM VMBLOK address will be loaded into Reg-11 before Control Reg-1 is loaded. If LOCK is specified, then a call to DMKPTRLK would be inserted in the Macro Expansion. Registers: Rx & Ry Ry = Contains the VIRTUAL address to be translated, this would normally be Reg-1. Rx = Will contain the REAL address when DMKPTRAN has completed its processing. NOTE: If other registers are used, then the TRANS Macro will switch them to Reg-1 & and Reg-2 respectively.

CP Module ABEND	- MACROS = ABEND

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
A B E N D	Used to Force a CP Module ABEND with Unique Code Value	ABEND n - Example: ABEND 3 Generated Code: xxx1 EQU * SVC 0 DC CL3'xxx',AL1(3)	n = Equals a Unique ABEND number within the the CP Module. xxx = Will be the specific CP Module identifier (Equated to MODID+3). The CP ABEND Macro will generated a Abend of CP using the Module Id field as further identification. In the example shown, the Abend that will be generated will be: xxx001 If the module name was DMKABC, then the Abend Code will be ' ABC001 '. Case should be taken when coding several ABEND macros within the same CP Module, to avoid duplicate Abend code values.

Construct Simple Message - MACRO = MSG

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
M S G	Used to Generate an inline message; used for DMKQCNWT Call	MSG 'Message text...' - (Sample Expansion) MSG 'This is a message' LA R0,17 LA R1,=C'This is a message'	Message Text = Equals a character-type message that will be sent to a VM User's console. Usage Notes: The MSG macro is used to generate an inline message text including the length of the message (into R0) and the address of the message (into R1). An immediate Call can be made to DMKQCNWT with options included to send the message to a VM user's console. NOTE: Registers-0 & 1 are used during the Macro invocation and are NOT saved. No other register pair can be specified on the Macro call.

Condition Code & Timer Switching - MACROS = SETCC, SWTCHVM

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
S E T C C	Used to Set a Final Condition Code within a Module before an EXIT is Made.	SETCC |0|1|2|3| - SETCC 0 CLI *+1,0 (The Condition Code is Now Set)	|0|1|2|3| = Represents the possible Condition Codes that can be set The SETCC Macro expands in-line with an instruction which will set the condition code based upon the value used (0, 1, 2 or 3). Common Usage: Prior to an EXIT Macro the SETCC Macro can be used to set a condition code. The EXIT Macro will preserve the condition code accross the SVC interrupt and pass the setting to the CP Module who initiated the CALL.
S W T C H V M	Used to Switch the CPU timer value to charge another User.	SWTCHVM - STPT VMTTIME-VMBLOK(R11) LR R11,R1 SPT VMTTIME-VMBLOK(R11)	SWTCHVM Macro is used to terminate the Time charging for the current user (Overhead Time-VMTTIME), who's address is contained in R11, and start charging the new user (address in R1) for overhead time (VMTTIME). In addition, the new user (R1 address) will be switched to R11 as the now current User. Registers: R11 = The address of the current user. R1 = The address of the new user (which will become the current user).

Establish CP Command Entry - MACRO = COMND (For Use Within DMKCFC)

C P Macro	Function(s)	Macro Format & Sample Expansion(s)	Explainations & Comments
C O M N D	Used to Create a Command Entry within DMKCFC for Command Processing.	COMND Name,Class,Min,Entry, | NCL= |,| TYPE = | - COMND FORCE,A+B,3,DMKABCXX	Name = The Actual Command Name (8-Characters Maximum Allowed) Class = The Allowable Classes Authorized for the Commandl set the condition code Min = The Minimum Allowable Abbreviation for the Command name. Entry = The Fully Qualified Entry Point Name of the Module to Process the Command. NCL=1 - Used to Allow Command Prior to LOGON TYPE = Either 'A' or 'V' for either ADCON or VCON Expansion. Default is to use VCON within Command entry. The COMND Macro is used to generate a Command Entry within DMKCFC (Command Look-up Module). Based upon the Command Entry, the allowable classes and minimum abbreviation fields will be verified by DMKCFC before the module is given control to process the command. In certain cases, the module who will receive control for the command, may further direct and invoke other CP module to process the command based upon class-type and/or other options supplied on the command line. Example woule be the 'Query' command.