Qualcomm Technologies, Inc.

Comprehensive AT Command Set in AMSS Software

Application Note

80-VR432-1 D

July 1, 2015

Confidential and Proprietary – Qualcomm Technologies, Inc.

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or websites to: DocCtrlAgent@qualcomm.com.

Restricted Distribution: Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	June 2009	Initial release
B	October 2010	Numerous changes were made to this document; it should be read in its entirety.
C	December 2012	Numerous changes were made to this document; it should be read in its entirety.
D	July 2015	Numerous changes were made to this document; it should be read in its entirety.

Contents

1 Introduction... 8

1.1 Purpose.. 8
1.2 Conventions 8
1.3 Technical assistance 8

2 AT Command Support.. 9

2.1 AT command set 10
2.2 Basic commands 12
2.3 DTE-TA/DCE interface commands 12
2.4 General commands.. 16
2.4.1 PCCA STD-101 commands 17
2.5 Call control commands 18
2.5.1 UMTS call control commands 18
2.5.2 ITU-T V.25ter call control commands... 23
2.5.3 ITU-T V.25ter data compression commands 24
2.5.4 Network service-related commands.. 25
2.6 Mobile equipment commands.. 29
2.7 UMTS packet domain commands.. 33
2.7.1 UMTS-specific packet domain commands 33
2.7.2 Packet domain modem compatibility commands 39
2.8 LTE-specific packet domain commands... 40
2.9 SMS commands 42
2.10 UMTS vendor-specific commands 45
2.11 Fax commands 62
2.12 Synchronous data mode commands.. 65
2.13 IS-707 AT command set for AMSS... 65
2.14 Proprietary CDMA AT command set 79

3 AT Command Processor... 89

3.1 ATCoP and related source files.. 89
3.2 ATCoP overview.. 93
3.2.1 General architecture and control flow... 93
3.2.2 SIO data preprocessor.. 97
3.2.3 AT command parser... 97
3.2.4 AT command processor 98
3.2.5 AT command forwarding.. 99
3.2.6 ATCoP command tables 100
3.2.7 Command table entries 101
3.2.8 Command tables 115
3.3 ATCoP asynchronous event handler tables.. 116

3.3.1 Asynchronous event handler table structure 116
3.3.2 Asynchronous event table entry 116
3.3.3 Asynchronous event handler table 118
3.4 Adding a new AT command 118
3.4.1 Adding a new asynchronous AT command 124

A List of Unsupported Commands 133

B References 138

B.1 Related documents 138

Figures

Figure 3-1 Basic ATCoP architecture and data flow ... 94
Figure 3-2 Control flow for AT commands processing ... 96
Figure 3-3 AT command table structure ... 100
Figure 3-4 Relationship between command data structures ... 107
Figure 3-5 Asynchronous event handler table structure ... 116

Tables

Table 2-1 Column headings ... 10
Table 2-2 Basic commands ... 12
Table 2-3 ITU-T V.25ter DTE-DCE interface commands ... 12
Table 2-4 UMTS general commands ... 16
Table 2-5 PCCA STD-101 commands ... 17
Table 2-6 UMTS call control commands ... 18
Table 2-7 ITU-T V.25ter call control commands ... 23
Table 2-8 ITU-T V.25ter data compression commands ... 24
Table 2-9 Network service-related commands ... 25
Table 2-10 Mobile equipment commands ... 29
Table 2-11 UMTS-specific packet domain commands ... 33
Table 2-12 Packet domain modem compatibility commands ... 40
Table 2-13 LTE-specific Packet domain modem compatibility commands ... 40
Table 2-14 SMS text and PDU mode commands ... 42
Table 2-15 Vendor-specific commands ... 45
Table 2-16 Fax ITU-T T.31 commands for GSM ... 62
Table 2-17 ITU-T V.80ter synchronous data mode commands ... 65
Table 2-18 [S9] Table 7.2-1, extended AT configuration commands (1 of 3) ... 65
Table 2-19 [S9] Table 7.2-1, extended AT configuration commands (2 of 3) ... 66
Table 2-20 [S9] Table 7.2-1, extended AT configuration commands (3 of 3) ... 67
Table 2-21 [S9] Table 7.3.1-1, fax parameters (1 of 2) ... 68
Table 2-22 [S9] Table 7.3.1-1, fax parameters (2 of 2) ... 69
Table 2-23 [S9] Table 7.3.2-1, fax action commands ... 71
Table 2-24 [S9] Table 7.4.1-1, CDMA AT parameter commands (1 of 4) ... 72
Table 2-25 [S9] Table 7.4.1-1, CDMA AT parameter commands (2 of 4) ... 73
Table 2-26 [S9] Table 7.4.1-1, CDMA AT parameter commands (3 of 4) ... 74
Table 2-27 [S9] Table 7.4.1-1, CDMA AT parameter commands (4 of 4) ... 76
Table 2-28 [S9] Table 7.4.1-2, cellular AT command extensions in support of voice services ... 77
Table 2-29 [S9] Table 7.4.1-3, cellular AT commands for packet data services ... 78
Table 2-30 [S9] Table 7.4.2-1, cellular result codes ... 78
Table 2-31 CDMA vendor-specific AT commands ... 79

Table 3-1 ATCoP source files..89
Table 3-2 Data services task source files..93
Table 3-3 AT command attributes 103
Table 3-4 Mode argument...105
Table 3-5 Execution function return values..106
Table 3-6 Command syntax flags 114
Table 3-7 Command categories 115
Table 3-8 Asynchronous event handler function return values 118
Table A-1 List of unsupported commands...133

1 Introduction

1.1 Purpose

This document specifies the AT command set for all Qualcomm Technologies, Inc. (QTI) chipsets. It groups AT commands into several categories. Each category identifies all AT commands applicable to certain functionalities, specifies the supported targets, and describes the applicable mode.

NOTE: Information in this document is preliminary and subject to change, and does not represent a commitment on the part of QTI.

1.2 Conventions

Function declarations, function names, type declarations, attributes, and code samples appear in a different font, for example, `#include`.

Shading indicates content that has been added or changed in this revision of the document.

1.3 Technical assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies, Inc. (QTI) at https://createpoint.qti.qualcomm.com/.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

2 AT Command Support

The following categories of AT commands will be supported for all RATs:

- DTE-DCE interface
- General

The following categories of AT commands will be supported for 3GPP/3GPP2 or both:

- Call control
- Network service-related
- Mobile equipment
- UMTS packet domain
- LTE packet domain
- TD-SCDMA
- Short Message Service (SMS)
- Vendor-specific
- Fax
- Synchronous Data mode

The following categories of AT commands will be supported for 3GPP2 only:

- IS-707.3 Table 7.2-1, Extended AT Configuration Commands
- IS-707.3 Table 7.3.1-1, Fax Parameters
- IS-707.3 Table 7.4.1-1, CDMA AT Parameter Commands
- IS-707.3 Table 7.4.1-2, Cellular AT Command Extensions in Support of Voice Services
- IS-707.3 Table 7.4.1-3, Cellular AT Commands for Packet Data Services
- CDMA Vendor-specific AT Commands

The TA/DCE will only accept AT commands at one fixed data rate. Automatic baud rate detection will not be supported.

2.1 AT command set

Supported AT commands for each category are defined in the sections that follow. The tables contain column headings defining the AT command. A description of these column headings is given in Table 2-1.

Table 2-1 contains details such as supported packages and applicable modes. There are different kinds of targets, e.g., 3GPP-only, 3GPP2-only, and Multimode. Multimode targets support for both 3GPP and 3GPP2, but not simultaneously. The applicable mode for some of these commands will be 3GPP-only, which means they are expected to work in only 3GPP service. Similarly, 3GPP2-only commands are expected to work in only 3GPP2 service. Mode "Any" commands shall work in both 3GPP and 3GPP2 service.

Table 2-1 Column headings

Heading	Description
Command	Defines the AT command name in its ASCII character string format
Description	Short explanation of the command and its values
[S1] or [S2] or ITU-T T.31 requirement IS 707.3	ETSI requirement column, specifies whether the AT command set specifications classify the command as one of the following: ▪ Mandatory ▪ Optional ▪ Not applicable (N/A)
Explanation	Provides insight into the reasoning behind the implementation

Heading	Description
Supported packages	▪ 1h07+ – MSM6240/6246/6260/6270/6290/7200/7225/7501/7601/76xx ▪ 1h08+ – QSC60x5, MSM6800, MDM8200, QSD8650 ▪ 1h09+ – SCMM ▪ 1h10+ – MDM9600, MDM8220, MSM7x30 ▪ MPSS NI 1.0+ – MSM8960.LA.1.0/QX.1.0/WP.1.0/MDM9615.TN.1.0 ▪ MPSS NI 2.0+ – MSM8960.LA.1.5/WP.1.5/QX.1.2, MDM9615.LE.1.0/TN.1.7(F), MSM8930.LA.1.0/WP.1.0/3.0 ▪ MPSS NI 3.0+ – MSM8960.LA.1.7/WP.1.7/WP.2.0, MDM9615.LE.2.0./TN.2.7(F), MSM8930.WP.1.3/2.0/3.1/LA 1.5/1.7, MSM8230.QX.1.0 ▪ MPSS NI 4.0+ – MSM8960.LA.2.0/2.2/QX.2.0, MDM9615.LE.3.0/T.N.2.1/2.3/3.0/3.7 ▪ MPSS NI 5.0+ – MSM8960.LA.2.3, MDM9615.LE.4.0,MSM8930.LA.4.0 ▪ MPSS NI 6.0+ – MSM8960 LA/QX, MDM9615 LE/TN, MSM8930 LA ▪ MPSS DI 1.0+ – MSM8974 LA/QX/WA/WP/LT, MDM9625 TN/LE/MN ▪ MPSS DI 2.0+ – MSM8974 LA/WP/QX, MDM9625 LE/TN/MN, MSM8926 LA/WP ▪ MPSS DI 3.0+ – MSM8974 LA, MDM9625 TN/LE, MSM8926 LA/LF ▪ MPSS DI 4.0+ – MSM8974 LA/WP, MDM9625 TN, MSM8926 LA/WP/LB ▪ MPSS TR 1.0+ – MSM8626 LA/LF, MSM8610 LA ▪ MPSS TR 2.0+ – MSM8626 LA/WP, MSM8226 LT, MSM8610 LA/LF, MSM8612 WP ▪ MPSS TR 3.0+ – MSM8610 LA ▪ MPSS DPM 1.0+ – MSM8916 LA/WP, MSM8939 LA ▪ MPSS DPM 2.0+ – MSM8916 WP/LT, MSM8939 LA ▪ MPSS BO 1.0+ – MDM9635M TN/LE/MN ▪ MPSS BO 2.0+ – MDM9635M LE/TN/MN, MSM8992 LA/WP, MSM8994 LA/WP ▪ MPSS JO 1.0+ – MSM8909 LA/WP/LF, MDM9607 LE, MSM8937 LA ▪ MPSS TH 1.0+ – MDM9640 LE, MDM9645 MN ▪ MPSS TH 2.0+ – MDM9640 LE, MDM9645 MN, MSM8996 LA/WP, MSM8978 LA ▪ MPSS TA 1.0+ – MSM8952 LA/WP ▪ MPSS TA 2.0+ – MSM8976 LA, MSM8953 LA/WP ▪ MPSS AT 1.0+ – MDM9650 LE, MDM9655 MN ▪ MPSS AT 2.0+ – MSM8998 LA/WP
Applicable mode[1]	Specifies in which modes the AT command works ▪ Any ▪ 3GPP ▪ 3GPP2

[1] Mode specifies the targets and RAT combinations that are applicable:

- 3GPP – GSM/UMTS/LTE/TD-SCDMA
- 3GPP2 – CDMA, 1X-DO

2.2 Basic commands

Basic commands are listed and described in Table 2-2.

Table 2-2 Basic commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
AT	Command line prefix	N/A		1h07+	Any
at	Command line prefix	N/A		1h07+	Any
A/	Reexecution of previously executed AT command	N/A		1h07+	Any
a/	Reexecution of previously executed AT command	N/A		1h07+	Any
+++	Switching from Online Date mode to Online Command mode	N/A	Currently supported for UMTS CS data only	1h07+	Any

2.3 DTE-TA/DCE interface commands

The DTE-TA/DCE interface commands are as indicated in Table 2-3. Command support will be as specified in [S1] and [S3].

Table 2-3 ITU-T V.25ter DTE-DCE interface commands

Command	Description	Explanation	Supported packages	Applicable mode
S3	Command line termination character; values per specifications		1h07+	Any
S4	Response formatting character; values per specifications		1h07+	Any
S5	Command line editing character; values per specifications		1h07+	Any
E<value>	Command echo ▪ <value> 0 – Disables echo ▪ <value> 1 – Enables echo		1h07+	Any
Q<value>	Result code suppression ▪ <value> 0 – Disables result codes ▪ <value> 1 – Enables result codes		1h07+	Any
V<value>	DCE response format ▪ <value> 0 – Displays result codes in numeric form ▪ <value> 1 – Displays result codes in verbose form		1h07+	Any

Command	Description	Explanation	Supported packages	Applicable mode
X<value>	Result code selection and call progress monitoring control ▪ <value> 0 – Sends a CONNECT message when a connection is established by blind dialing; ignores dial tone and busy signal ▪ <value> 1 – Enables additional result code CONNECT<rate>; disables dial tone and busy detection ▪ <value> 2 – Enables additional result codes CONNECT <rate> and NO DIALTONE; disables busy detection; enables dial tone detection ▪ <value> 3 – Enables additional result codes CONNECT <rate> and BUSY; enables busy detection; disables dial tone detection ▪ <value> 4 – Enables additional result codes CONNECT <rate>, BUSY, and NO DIALTONE; enables busy and dial tone detection		1h07+	Any
&C<value>	Circuit 109 DCE RLSD (DCD) behavior ▪ <value> 0 – Circuit 109 (CF) always ON ▪ <value> 1 – Circuit 109 (CF) ON in accordance with the specified service ▪ <value> 2 – Circuit 109 (CF) always on except wink on channel disconnect	&C2 is a QTI implementation	1h07+	Any
&D<value>	Circuit 108 DTE DTR behavior ▪ <value> 0 – Ignores circuit 108/2 (CD) ▪ <value> 1 – Enters Online Command state following ON-to-OFF transition of circuit 108/2 ▪ <value> 2 – Enters Command state following ON-to-OFF transition of circuit 108/2	Applicable only for UMTS CSD	1h07+	Any
&S	DSR control management Default – Always ON (&S0)		1h07+	Any
&W	Stores the V250 registers and S-registers into NV memory		1h07+	Any
&E <n>	Controls the display of data rate to be either serial rate or wireless connection speed. Default – Serial/DTE rate <n> = 0 Serial/DTE rate (default) = 1 wireless connection speed		1h07+	Any

Command	Description	Explanation	Supported packages	Applicable mode
+IPR= <rate>	Fixed DTE rate; this numeric extended-format parameter specifies the data rate at which the DCE will accept commands; auto baud rate detection is not supported <rate> – 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 921600, 2000000, 2900000, 3000000, 3200000, 3686400, 4000000	Default DTE rate fixed at 115200 bps; default DTE rate can be changed by $QCTER command	1h07+	Any
+ICF= <format>, <parity>	DTE-DCE character framing; this extended-format compound parameter is used to determine the local serial port start-stop (asynchronous) character framing that the DCE shall use while accepting DTE commands and while transmitting information text and result codes to the DTE: ▪ <format> – 3 ▪ <parity> – Values per specifications	QTI R_m interface fixed at 8 data bits, no parity, 1 stop bit; error returned for any other parameters	1h07+	Any
+IFC= <DCE by DTE>, <DTE by DCE>	DTE-DCE local flow control; this extended-format compound parameter is used to control the operation of local flow control between the DTE and DCE; values per specifications	Hardware and software flow control supported for asynchronous service	1h07+	Any
Z<value>	Resets to default configuration; values per specifications; command processed regardless of SIM state	Resets configuration; does not change DCE baud rate or PDP context profiles	1h07+	Any
&F<value>	Sets to factory-defined configuration (effect is implementation-dependent); values per specifications; command processed regardless of SIM state	Same behavior as Z except it changes baud rate to default value	1h07+	Any
I	Requests identification information; no value accepted; command processed regardless of SIM state		1h07+	Any
+GMI	Requests manufacturer identification; command processed regardless of SIM state		1h07+	Any
+GMM	Requests model identification; command processed regardless of SIM state		1h07+	Any
+GMR	Requests revision identification; command processed regardless of SIM state		1h07+	Any
+GSN	Requests product serial number identification; command processed regardless of SIM state		1h07+	Any

Command	Description	Explanation	Supported packages	Applicable mode
+GCAP	Requests complete capabilities list	In GSM mode, unit outputs: ▪ +GCAP ▪ +CGSM ▪ +FCLASS ▪ +DS In WCDMA mode, unit outputs: ▪ +GCAP ▪ +CGSM ▪ +ES	1h07+	Any
S0=<value>	Enables/disables automatic answering ▪ <value> 0 – Automatic answering is disabled ▪ <value> 1 to 255 – Enable automatic answering on the ring number specified		1h07+	Any
S2	Command line escape character	Currently supported for CS data	1h07+	
S6=<value>	Pauses before blind dialing; values per specifications		1h07+	Any
S7=<value>	Number of seconds to establish end-to-end data connection; values per specifications		1h09+	Any
S8=<value>	Number of seconds to pause when ‘,’ is encountered in dial string; values per specifications	Asynchronous data command; command accepted; no action taken	1h07+	Any
S10= <value>	Number of tenths of a second from carrier loss to disconnect; values per specifications	Asynchronous data command; command accepted; no action taken	1h07+	Any
S30=<n>	Setting of inactive timer value for disconnection when no user data is exchanged during a data call <n> – 0 to 255	Supported only for CS data	1h07+	Any
S103= <value>	Assigning an identifier in incoming call subaddress		1h07+	Any
S104= <value>	Assigning an identifier in outgoing call subaddress		1h07+	Any
L<value>	Monitors speaker loudness; values per specifications		1h07+	Any
M<value>	Monitors Speaker mode; values per specifications		1h07+	Any
\S	Prints the AT command settings		1h07+	Any
\V	Enables/disables the display of extended result code in response to a data call		1h07+	Any

Command	Description	Explanation	Supported packages	Applicable mode
%V	Requests revision identification; command processed regardless of SIM state	Similar to +CGMR/ +GMR command	1h07+	Any

2.4 General commands

For UMTS, support of the generic UMTS commands is indicated in Table 2-4. Command support is specified in [S1].

Table 2-4 UMTS general commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGMI	Requests manufacturer identification; command processed regardless of SIM state	Optional		1h07+	Any
+CGMM	Requests model identification; command processed regardless of SIM state	Optional		1h07+	Any
+CGMR	Requests revision identification; command processed regardless of SIM state	Optional		1h07+	Any
+CGSN	Requests product serial number identification; command processed regardless of SIM state	Optional		1h07+	Any
+CSCS=<chset>	Selects TE character set; values per specifications <chset> – IRA, GSM, UCS2	Mandatory when other commands that use this setting are implemented		1h07+	Any
+CIMI	Requests international mobile subscriber identity	Optional		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CMUX	Used to enable/disable the 3GPP TS 27.010 multiplexing protocol control channel.	Mandatory, if 3GPP TS 27.010 supported in the MT/TA.	Enabled only when Data MUX feature is enabled.	MPSS DI 2.0+	3GPP

2.4.1 PCCA STD-101 commands

For UMTS, support of the cellular network selection commands is as indicated in Table 2-5. Command support shall be as specified in [S4].

Table 2-5 PCCA STD-101 commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+WS46	Selects wireless network	Optional	Only query command Supported, Values accepted but no action taken		3GPP

2.5 Call control commands

2.5.1 UMTS call control commands

Call control commands are supported as indicated in Table 2-6. Command support will be as specified in [S1].

Table 2-6 UMTS call control commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CSTA= <type>	Selects type of address; values per specifications <type> ▪ 129 ▪ 145	Mandatory when nondefault value is allowed		1h07+	3GPP
D <dial string>;	Originates a voice call; semicolon suffix indicates voice; supported dial modifiers are: ▪ > – Direct dial from phonebook ▪ Values per specifications ▪ Command is abortable ▪ Command processed regardless of SIM state for emergency call dial string ▪ Multiple calls; originates another call, if already an active call is present making the existing call on hold	Optional		1h07+	3GPP
+CMOD	Call mode <mode> – 0	Mandatory when alternating mode calls are supported		1h07+	3GPP
+CHUP	Hang-up voice call	Mandatory		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CBST= <speed>, <name>, <ce>	Selects the circuit-switched bearer service <name> with data rate <speed>, and connection element <ce> when data calls are originated; parameter values supported are: ▪ <speed> (in bps) ▫ 0 – Autobaud ▫ 7 – 9600 (V.32) ▫ 12 – 9600 (V.34) ▫ 14 – 14400 (V.34) ▫ 16 – 28800 (V.34) ▫ 17 – 33600 (V.34) ▫ 39 – 9600 (V.120) ▫ 43 – 14400 (V.120) ▫ 48 – 28800 (V.120) ▫ 51 – 48000 (V.120) ▫ 71 – 9600 (V.110) ▫ 75 – 14400 (V.110) ▫ 80 – 28800 (V.110) ▫ 81 – 38400 (V.110) ▫ 83 – 56000 (X.31 flag stuffing, UDI/RDI) 84 – 64000 bps (X.31 flag stuffing, UDI) ▫ 116 – 64000 bps ▫ 134 – 64000 bps (multimedia) ▪ <name> ▫ 0 – Data circuit asynchronous (UDI or 3.1 kHz modem) ▫ 1 – Data circuit synchronous (UDI or 3.1 kHz modem) ▫ 4 – Data circuit asynchronous (RDI) ▪ <ce> ▫ 0 – Data transparent ▫ 1 – Data nontransparent	Mandatory		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CRLP= <iws>, <mws>, <T1>, <N2>	Alters the RLP parameters used when nontransparent circuit-switched data calls are originated. For Vers 0 and 1, the following parameter values are supported: ▪ <iws> – 0 to 61 frames ▪ <mws> – 0 to 61 frames ▪ <T1> – 38 to 255 x 10 ms ▪ <N2> – 1 to 255 retransmits For Ver 2, the following parameter values are supported: ▪ <iws> – 0 to 488 frames ▪ <mws> – 0 to 488 frames ▪ <T1> – 42 to 255 x10 ms ▪ <N2> – 1 to 255 retransmits	Mandatory	RLP Vers 0, 1, and 2 are supported	1h07+	3GPP
+CR= <mode>	Service reporting control; values per specifications	Mandatory		1h07+	3GPP
+CEER	Extended error report	Optional	Report the reason for the last call failure to set up or release; both CS and PS domain call types are reported	1h07+	3GPP
+CRC= <mode>	Cellular result codes; values per specifications	Mandatory		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CHSN	HSCSD nontransparent call configuration; parameter values supported are: ▪ <wAiur> (in bps) ▫ 0 – TA shall calculate a proper value from currently selected fixed network user rate (<speed> Sub parameter from +CBST command) ▫ 2 – 14400 ▫ 4 – 28800 ▫ 7 – 57600 ▪ <wRx> ▫ 0 – TA shall calculate a proper value from currently selected ▪ <wAiur> and <codings> ▪ <topRx> ▫ 0 – TA shall calculate a proper value from currently selected ▪ <wAiur> and <codings> ▪ <codings> ▫ 0 – All supported codings are accepted	Mandatory		1h07+	3GPP
+CVHU= [<mode>]	Voice hang-up control <mode> ▪ 0 – ATH disconnects ▪ 1 – ATH ignored	Optional	Defaulted to 1	1h07+	3GPP
+CV120	V.120 rate adaptation protocol	Mandatory when V.120 interworking is supported by ME		1h07+	3GPP
+VTS=<dtmf>	Inband DTMF generation <dtmf> tone is transmitted (default duration is 20 ms)	Optional		1h08+	3GPP
+CECALL= <type_of_eCall>	Trigger an eCall to the network <type_of_eCall> – Integer type ▪ 0 – Test call ▪ 1 – Reconfiguration call eCall ▪ 2 – Manually initiated eCall ▪ 3 – Automatically initiated eCall	Optional		MPSS NI 1.0+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+ CGPIAF= [<IPv6_AddressFormat> [,<IPv6_SubnetNotation> [,<IPv6_LeadingZeros> [,<IPv6_CompressZeros>]]]]	Set format to print IPv6 address parameters. ▪ IPv6_AddressFormat ▪ IPv6_SubnetNotation ▪ IPv6_LeadingZeros ▪ IPv6_CompressZeros ▫ 0 – Disable ▫ 1 – Enable	Optional		MPSS NI 3.0+	3GPP
+CEN= [<reporting>]	Report emergency numbers as received from the network. ▪ Reporting ▫ 0 – Disable reporting ▫ 1 – Enable reporting	Optional		MPSS NI 3.0+	3GPP
+CVMOD= [<voice_mode>]	Set the command for Voice Call mode for making a MO voice call. This command will update the NV item based on Voice mode. ▪ voice_mode ▫ 0 – CS_ONLY ▫ 1 – VOIP_ONLY ▫ 2 – CS_PREFERRED ▫ 3 – VOIP_PREFERRED	Optional		MPSS NI 3.0+	3GPP
+CSTF= [<mode>]	Set the time format of the time information. This command will update the NV item based on mode. ▪ mode ▫ 1 – HH:MM (24-hour clock) ▫ 2 – HH:MM a.m./p.m. ▫ 3-7 – Manufacturer specific	Optional		MPSS NI 3.0+	3GPP
+CSDF= [[<mode>] [,<auxmode>]]	This command sets the date format. ▪ mode – 1 to 7 ▪ aux mode – 1 to 2	Optional		MPSS NI 4.0+	3GPP
+CERP= <EAPsessionid>,<EAPparameter>	This command allows a TE to retrieve EAP session parameters/derived keys. ▪ EAP session ID – The identifier of the EAP session ▪ EAP parameter – Only key, status, and identity are supported	Optional		MPSS NI 2.0+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CEAP= <dfname>, <EAPMethod>, <EAP packet data>	This command allows a TE to exchange EAP packets with the UICC or the ME. ▪ EAP processing is handled in MT, therefore, <dfname> is not treated ▪ EAP method – Only EAP-AKA (23) and EAP-SIM (18) are supported ▪ EAP packet data : string type in hexadecimal character format	Optional		MPSS NI 2.0+	3GPP
+CUAD	This command asks the MT to discover what applications are available for selection on the UICC.	Optional		MPSS NI 2.0+	3GPP
+CMAR[2]	Master reset.	Optional		MPSS NI 1.0+	3GPP
+CRSL[2]	Ringer sound level **Note:**Only command forwarding supported	Optional		MPSS NI 1.0+	3GPP
+CDIS[2]	Display control **Note:** Only command forwarding supported	Optional		MPSS NI 1.0+	3GPP

2.5.2 ITU-T V.25ter call control commands

For UMTS, support the call control commands as indicated in Table 2-7. Command support shall be as specified in [S1] and [S3].

Table 2-7 ITU-T V.25ter call control commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
D <dial string>	Dial circuit-switched data call; supported dial modifiers are: ▪ > – Direct dial from phonebook ▪ I/I – Calling line identification restriction ▪ G/g – Closed user group Values per specifications; command is abortable	Mandatory		1h07+	Any

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
T	Selects tone dialing This commands will have no impact in our code.	Mandatory (ignored in GSM/UMTS)		1h07+	Any
P	Selects pulse dialing This command will have no impact in our code.	Mandatory (ignored in GSM/UMTS)		1h07+	Any
A	Answers incoming call; command is abortable	Mandatory		1h07+	3GPP
H<value>	Hook control command to terminate call in progress; does not terminate voice calls; values per specifications	Mandatory		1h07+	3GPP
O<value>	Returns to Online Data state from Online Command state; values per specifications	Mandatory		1h07+	3GPP

2.5.3 ITU-T V.25ter data compression commands

Support for data compression commands as indicated in Table 2-8. Command support shall be as specified in [S1] and [S3].

Table 2-8 ITU-T V.25ter data compression commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+DR= <value>	Reports use of V.42bis using intermediate result code before going to Online Data state after call answer or origination; values per specifications	Mandatory	No action taken in CDMA	1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+DS= <dir>, <neg>, <P1>, <P2>	Controls V.42bis data compression; values per specifications; parameter values supported are: ▪ <dir> – 0 to 3 ▪ <neg> – 0 ▪ <P1> – 512 to 2048 ▪ <P2> – 6	Mandatory	No action taken in CDMA	1h07+	3GPP

2.5.4 Network service-related commands

Support the network service-related commands as indicated in Table 2-9. Command support shall be as specified in [S1].

Table 2-9 Network service-related commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CNUM	Subscriber number	Optional		1h07+	Any
+CREG=[<n>]	Network registration; parameter values supported are: ▪ <n> – 0, 1, 2	Optional	No action taken in CDMA Additional parameters added in Ver 9; 27007 specification is supported from 1h10+	1h07+	3GPP
+COPS=[<mode> [,<format> [,<oper>]]]	Operator selection; parameter values supported are: ▪ <mode> – 0, 1, 2, 3, 4 ▪ <format> – 0, 1, 2	Optional		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CLCK= <fac>, <mode>, <passwd>, <class>	Locks, unlocks, or interrogates an ME or a network facility; values per specifications; command is abortable; parameter values supported are: ▪ <fac> ▫ AB ▫ AC ▫ AG ▫ AI ▫ AO ▫ IR ▫ OI ▫ OX ▫ SC ▫ FD ▫ PN ▫ PU ▫ PP ▫ PC ▫ PF ▪ <mode> – 0 to 2 ▪ <class> – 1 to 255	Mandatory		1h07+	3GPP
+CPWD= <fac>, <oldpwd>, <newpwd>	Sets new password for a facility lock function; values per specifications; parameter values supported are: ▪ <fac> ▫ AB ▫ AC ▫ AG ▫ AI ▫ AO ▫ IR ▫ OI ▫ OX ▫ P2 ▫ SC	Optional		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CLIP=[<n>]	Calling line identification presentation; values per specifications <n> – 0, 1	Optional		1h07+	3GPP
+CLIR=[<n>]	Calling line identification restriction; values per specifications <n> – 0, 1,2	Optional		1h07+	3GPP
+COLP=[<n>]	Connected line identification presentation; values per specifications <n> – 0, 1	Optional		1h07+	3GPP
+CDIP=[<n>]	Called line identification presentation; values per specifications <n> – 0, 1	Optional		1h07+	3GPP
+CCUG= <n>, <index>, <info>	Controls closed user group supplementary service; values per specifications	Optional		1h07+	3GPP
+CCFC= <reason>, <mode>, <number>, <type>, <class>, <subaddr>, <satype>, <time>	Controls call forwarding supplementary service; values per specifications	Mandatory		1h07+	3GPP
+CCWA=[<n> [,<mode> [,<class>]]]	Call waiting supplementary services; command input values per specifications. ▪ Unsolicited result code <class>. ▫ Only 1 or 2 reported. ▪ Unsolicited result code optional <alpha> and <CLI validity> ▫ Not supported	Optional		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CHLD=[<n>]	Call-related supplementary services <n> – 0, 1, 1X, 2, 2x, 3, 4	Optional		1h07+	3GPP
+CTFR= <number>[,<type> [,<subaddr> [,<satype>]]]	Call deflection supplementary service; values as per specifications	Optional		1h07+	3GPP
+CUSD= <n>, <str>, <dcs>	Controls unstructured supplementary service data; values per specifications	Optional		1h07+	3GPP
+CAOC[= <mode>]	Advice of charge; values per specifications	Optional		1h07+	3GPP
+CSSN= [<n>[,<m>]]	Supplementary service notifications; values per specifications	Optional		1h07+	3GPP
+CLCC	Lists current calls	Optional	Only lists current circuit-switched calls	1h07+	Any
+CPOL= [<index>] [, <format> [,<oper> [,<GSM_AcT>, <GSM_Compact_AcT>, <UTRAN_AcT>]]]	Preferred operator list; parameter values supported are: ▪ <index> – 1 to 85 ▪ <format> – 0, 1, 2 ▪ <GSM_AcT> – 0/1 ▪ <GSM_ Compact_AcT> – 0/1 ▪ <UTRAN_ ACT> – 0/1	Optional	Additional parameters added in Ver 9; 27007 specification is supported from 1h10+	1h07+	3GPP
+COPN	Reads operator names	Optional		1h07+	3GPP
+CPLS=<list>	Selection of Preferred PLMN LIST <list>=0, 1, 2	Optional		1h07+	3GPP

2.6 Mobile equipment commands

Mobile equipment control and status commands are supported as indicated in Table 2-10. Command support will be as specified in [S1].

Table 2-10 Mobile equipment commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CPAS	Reports phone activity status; only states ready, ringing, and call in progress are reported; command processed when ME in Limited Service state	Mandatory		1h07+	3GPP
+CFUN=[<fun>[,<rst>]][3]	Sets phone functionality; values per specifications; parameter values supported are: ▪ <fun> – 0, 1,4 (as per specification) ▫ 5 (Factory Test Mode) ▫ 6 (reset) ▫ 7 (offline) ▪ <rst> – 0,1	Optional		1h07+	Any
+CPIN=<pin>, <newpin>	Enters PIN; values per specifications; only SIM PIN/PUK and PIN2/PUK2 supported; command processed when ME in limited service state	Mandatory		1h07+	3GPP
+CBC[2]	Reports battery charge	Optional		1h07+	Any
+CSQ	Reports signal quality; values per specifications	Optional	For CDMA command, the name is +CCSQ and response differs from UMTS; only RSSI value reporting supported in CDMA	1h07+	3GPP
+CMEC=[<keyp>[,<disp>[,<ind>]]][2]	Mobile equipment control code; parameter values supported are: ▪ <keyp> – 0, 1, 2 ▪ <disp> – 0 ▪ <ind> – 0	Mandatory when any keypad, display, or indicator command is implemented		1h07+	Any
+CSS[2]	Command screen size	Optional		MPSS NI 1.0+	Any

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CKPD=<keys> [,<time> [,<pause>]][2]	Keypad control ▪ <keys> – String of characters representing keys ▪ <time>,<pause> – 0...255	Mandatory for ME not supporting +CPIN command and not having MMI	Not all character codes supported; not supported in thin UI multiprocess or builds. <pause> not supported from MPSS NI 1.0 and later.	1h07+	Any
+CPBS= <storage>, <password>	Selects phonebook memory storage; values per specifications <storage> ▪ SM ▪ LD ▪ DC ▪ FD ▪ MC ▪ ME ▪ RC ▪ EN ▪ ON	Optional	Need to set $QCPBMPREF command to select the appropriate phonebook preference.	1h07+	Any
+CPBR= <index1>, <index2>	Reads phonebook entries; values per specifications	Optional	Need to set $QCPBMPREF command to select the appropriate phonebook preference.	1h07+	Any
+CPBF= <find text>	Finds phonebook entries; values per specifications	Optional	Need to set $QCPBMPREF command to select the appropriate phonebook preference.	1h07+	Any
+CPBW= <index>, <number>, <type>, <text>, <group>, <adnumber>, <adtype>, <secondtext>, <email>	Writes phonebook entry; values per specifications	Optional	Need to set $QCPBMPREF command to select the appropriate phonebook preference.	1h07+	Any

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CCLK=<time>	Clock; values per specifications <time> – yy/MM/dd,hh:mm:ss ± zz	Optional	3GPP2 does not support automatic time update via NITZ.	1h07+	Any
+CSIM= <length>, <command>	Generic SIM access <length> – 10 to 520	Optional	Only default channel supported.	1h07+	3GPP
+CRSM= <command> [,<fileid> [,<P1>,<P2>, <P3>[,<data>,< pathid>]]]	Restricted SIM access; values per specifications	Optional	Only default channel supported.	1h07+	3GPP
+CACM= [<passwd>]	Accumulated call meter; values per specifications	Optional		1h07+	3GPP
+CAMM= [<acmmax> [,<passwd>]]	Accumulated call meter maximum; values per specifications	Optional		1h07+	3GPP
+CPUC= <currency>, <ppu> [,<passwd>]	Price per unit and currency table; values per specifications	Optional		1h07+	3GPP
+CLAC	Lists all available AT commands	Optional		1h07+	3GPP
+CTZU= <on/off>	Automatic time zone update <onoff> – 0, 1	Optional		1h07+	3GPP
+CTZR=<onoff>	Time zone reporting <onoff> – 0, 1	Optional		1h07+	3GPP
+CMEE= <n>	Reports mobile equipment error; values per specifications	Mandatory		1h07+	Any
+CLVL= <n>[2]	Used to set the loudspeaker volume level <n> – 0 to 5	Optional		1h07+	Any
+CMUT= <n>[2]	Used to disable/enable the uplink voice transmission <n> – 0, 1	Optional		1h07+	Any

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CBKLT=<state>[,<duration>][2]	Enable or disable the backlight of the MT's main display. <state>: ▪ 0 – Disable ▪ 1 – Enable for the duration specified ▪ 2 – Enable indefinitely ▪ 3 – Enable for a short duration specified by the UE manufacturer (default) <duration>: xxxx, in seconds	Optional		1h08+	Any
+CTSA=<action>,<x>,<y>[2]	Emulates touchscreen action on the mobile equipment; values per specification: <action>: 0-3 <x>,<y>: coordinates on screen	Optional		1h08+	Any

2.7 UMTS packet domain commands

2.7.1 UMTS-specific packet domain commands

Packet domain commands specific to UMTS are supported as indicated in Table 2-11. Command support will be as specified in [S1].

Table 2-11 UMTS-specific packet domain commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGDCONT= <cid>, <PDP_Type>, <APN>, <PDP_addr>, <d_comp>, <h_comp>	Sets PDP context parameter values for a PDP context identified by connection identifier; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <PDP_type> ▫ IP ▫ IPv6 ▫ IPv4v6 ▫ PPP ▪ <d_comp> – 0, 2 ▪ <h_comp> – 0, 4	Mandatory	See $QCPDPP command for connection authentication parameters Additional parameters added in Ver 9; 27007 specification is supported from 1h10+ <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
+CGDSCONT= [<cid> ,<p_cid> [,<d_comp> [,<h_comp>]]]	Defines secondary PDP context; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <p_cid> – 1 to 24 ▪ <d_comp> – 0, 2 ▪ <h_comp> – 0, 4	Optional	<cid> and <p_cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGTFT=[<cid>, [<packet filter identifier>, <evaluation precedence index> [,<source address and subnet mask> [,<protocol number (ipv4) / next header (ipv6)> [,<destination port range> [,<source port range> [,<ipsec security parameter index (spi)> [,<type of service (tos) (ipv4) and mask / traffic class (ipv6) and mask> [,<flow label (ipv6)>]]]]]]]]	Traffic flow template; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <packet filter identifier> – 1, 2 ▪ <evaluation precedence index> – 0 to 255 ▪ <source address and subnet mask> – 0.0.0.0.0.0.0.0 to 255.255.255.255. 255.255.255.255 ▪ <protocol number (ipv4)/next header (ipv6)> – 0 to 255 ▪ <destination port range> – 0.0 to 65535.65535 ▪ <source port range> – 0.0 to 65535.65535 ▪ <ipsec Security parameter index (spi)> – 0 to FFFFFFFF ▪ <type of service (tos) (ipv4) and mask/traffic class (ipv6) and mask> –0.0 to 255.255 ▪ <flow label (ipv6)> – 0 to FFFFF	Optional	<cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
+CGQREQ= <cid>, <precedence>, <delay>, <reliability>, <peak>, <mean>	Set the QoS profile used in Activate PDP Context Request message; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <precedence> – 1 to 3 ▪ <delay> – 1 to 4 ▪ <reliability> – 1 to 5 ▪ <peak> – 1 to 4 ▪ <mean> – 1 to 18, 31[1]	Optional	Setting these parameters will reset +CGEQREQ and +CGEQMIN to defaults <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGQMIN= <cid>, <precedence>, <delay>, <reliability>, <peak>, <mean>	Sets minimum acceptable profile against the negotiated profile in Activate PDP Context Accept message; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <precedence> – 1 to 3 ▪ <delay> – 1 to 4 ▪ <reliability> – 1 to 5 ▪ <peak> – 1 to 4 ▪ <mean> – 1 to 18, 31[1]	Optional	Command accepted, no action taken; setting these parameters will reset +CGEQREQ and +CGEQMIN to defaults <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP

+CGEQREQ= <cid>, <Traffic_ class>, <maximum_ bitrate_UL>, <maximum_ bitrate_DL>, <Guaranteed_ bitrate_UL>, <Guaranteed_ bitrate_DL>, <Delivery_ order>, <Maximum_ SDU_size>, <SDU_error_ ratio>, <Residual_bit_ error_ratio>, <Delivery_of_ erroneous_ SDUs>, <Transfer_ delay>, <Traffic_ handling_ priority>, <Source Statistics Descriptor>, <Signaling Indication>	Sets the UMTS QoS profile that is used in the Activate PDP Context Request message; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <Traffic_class> – 0 to 4 ▪ <maximum_ bitrate_UL> – 0 to 384 ▪ <maximum_ bitrate_DL> – 0 to 384 ▪ <Guaranteed_ bitrate_UL> – 0 to 384 ▪ <Guaranteed_ bitrate_DL> – 0 to 384 ▪ <Delivery_order> – 0 to 2 ▪ <Maximum_SDU_ size> – 0 to 1520 ▪ <SDU_error_ratio> ▫ 0E0 ▫ 1E1 ▫ 1E2 ▫ 7E3 ▫ 1E3 ▫ 1E4 ▫ 1E5 ▫ 1E6 ▪ <Residual_bit_ error_ratio> ▫ 0E0 ▫ 5E2 ▫ 1E2 ▫ 5E3 ▫ 4E3 ▫ 1E3 ▫ 1E4 ▫ 1E5 ▫ 1E6 ▫ 6E8 ▪ <Delivery_of_erroneous_SDUs> – 0 to 3 ▪ <Transfer_delay> – 0, 100 to 4000 ▪ <Traffic_handling_ priority> – 0 to 3[1] ▪ <Source Statistics Descriptor> – 0 to 1 ▪ <signaling Indication> – 0 to 1	Optional	Setting these parameters will reset +CGQMIN and +CGQREQ to defaults. If HSUPA is supported, then upper limit for <maximum_ bitrate_UL> and <Guaranteed _bitrate_UL> is 2048. If HSDPA is supported, then upper limit for <maximum_ bitrate_DL> and <Guaranteed _bitrate_DL> is 7168. <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP

+CGEQMIN = <cid>, <Traffic_class>, <maximum_ bitrate_UL>, <maximum_ bitrate_DL>, <Guaranteed_ bitrate_UL>, <Guaranteed_ bitrate_DL>, <Delivery_ order>, <Maximum_ SDU_size>, <SDU_error_ ratio>, <Residual_bit_ error_ratio>, <Delivery_of_ erroneous_ SDUs>, <Transfer_ delay>, <Traffic_ handling_ priority>, <Source Statistics Descriptor>, <Signaling Indication>	Sets the UMTS QoS profile used in the Activate PDP Context Request message; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <Traffic_class> – 0 to 4 ▪ <maximum_ bitrate_UL> – 0 to 384 ▪ <maximum_ bitrate_DL> – 0 to 384 ▪ <Guaranteed_ bitrate_UL> – 0 to 384 ▪ <Guaranteed_ bitrate_DL> – 0 to 384 ▪ <Delivery_order> – 0 to 2 ▪ <Maximum_SDU_ size> – 0 to 1520 ▪ <SDU_error_ratio> ▫ 0E0 ▫ 1E1 ▫ 1E2 ▫ 7E3 ▫ 1E3 ▫ 1E4 ▫ 1E5 ▫ 1E6 ▪ <Residual_bit_ error_ratio> ▫ 0E0 ▫ 5E2 ▫ 1E2 ▫ 5E3 ▫ 4E3 ▫ 1E3 ▫ 1E4 ▫ 1E5 ▫ 1E6 ▫ 6E8 ▪ <Delivery_of_erroneous_SDUs> – 0 to 3 ▪ <Transfer_delay> – 0, 100 to 4000 ▪ <Traffic_handling_ priority> – 0 to 3[1] ▪ <Source Statistics Descriptor> – 0 to 1 ▪ <signaling Indication> – 0 to 1	Optional	Setting these parameters will reset +CGQMIN and +CGQREQ to defaults. If HSUPA is supported, then upper limit for <maximum_ bitrate_UL> and <Guaranteed _bitrate_UL> is 2048. If HSDPA is supported, then upper limit for <maximum_ bitrate_DL> and <Guaranteed _bitrate_DL> is 7168. <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGATT= [<state>]	Attaches or detaches from the packet domain service; values per specifications; parameter values supported are: <state> ▪ 0 – Detached ▪ 1– Attached	Optional		1h07+	3GPP
+CGACT= [<state> [,<cid> [,<cid>[,…]]]]	Activates or deactivates the specified PDP context(s); values per specifications; parameter values supported are: ▪ <state> ▫ 0 – Deactivated ▫ 1 – Activated ▪ <cid> – 1 to 24	Optional	<cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
+CGCMOD= [<cid> [,<cid> [,…]]]	PDP context modify; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24	Optional	<cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
+CGDATA= [<L2P> ,[<cid> [,<cid> [,…]]]]	Enters Data state	Optional if dial command can be used to specify packet domain operation		1h07+	3GPP
+CGPADDR= [<cid> [,<cid> [,…]]]	Shows PDP address; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24	Optional	<cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
+CGCLASS= [<class>]	Sets the GPRS mobile class; values per specifications; parameter values supported are: ▪ A – Class A mode of operation	Optional	Supports only Class A	1h07+	3GPP
+CGEREP= [<mode>[,<bfr>]]	Controls sending of unsolicited result codes; values per specifications; parameter values supported are: ▪ <mode> – 0 to 2 ▪ <bfr> – 0 to 1	Optional		1h07+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGREG=[<n>]	Controls the presentation of unsolicited GPRS network registration status; values per specifications; parameter values supported are: ▪ <n> – 0, 1, 2	Optional		1h07+	3GPP
+CGSMS= [<service>]	Service preference that will be used to send mobile-originated SMS messages; parameter values supported are: ▪ <service> ▫ 0 – Packet domain ▫ 1 – Circuit switched ▫ 2 – Packet domain preferred ▫ 3 – Circuit-switched preferred	Optional		1h07+	3GPP
+QCGEQMIN	Same as +CGTFT for EPC profiles.	Optional		MPSS BO 2.0+	3GPP
+QCGEQOS	Same as +CGEQOS for EPC profiles.	Optional		MPSS BO 2.0+	3GPP
+QCGEQREQ	Same as +CGEQREQ for EPC profiles.	Optional		MPSS BO 2.0+	3GPP
+QCGQMIN	Same as +CGQMIN for EPC profiles.	Optional		MPSS BO 2.0+	3GPP
+QCGQREQ	Same as +CCGQREQ for EPC profiles.	Optional		MPSS BO 2.0+	3GPP
+QCGTFT	Same as +CCGTFT for EPC profiles.	Optional		MPSS BO 2.0+	3GPP

[1] Set values are saved across power cycles.

2.7.2 Packet domain modem compatibility commands

Existing modem compatibility commands for packet domain operation is supported as indicated in Table 2-12. Command support will be as specified in [S1].

Table 2-12 Packet domain modem compatibility commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
D	Dial (request packet domain service or packet domain IP service); values per specifications; support *98# and *99#; optional connection parameter validated against defined PDP contexts (see +CGDCONT command); supported dial modifiers are: ▪ *<cid> – PDP context definition Command is abortable	Optional if +CGDATA command is supported		1h07+	3GPP

2.8 LTE-specific packet domain commands

NOTE: This section was added to this document revision.

Existing modem compatibility commands for LTE packet domain operation is supported as indicated in Table 2-13. Command support will be as specified in [S1].

Table 2-13 LTE-specific Packet domain modem compatibility commands

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CEREG=[<n>]	Controls the presentation of unsolicited EPS network registration status; values per specifications; parameter values supported are: <n> – 0, 1, 2	Optional	Command supported in multimode or LTE-enabled targets	1h10+	3GPP
+CGEQOS=[<cid> [,<QCI> [,<DL_GBR>, <UL_GBR> [,<DL_MBR>, <UL_MBR]]]]	Sets EPS Quality of Service parameters; values per specifications; parameter values supported are: ▪ <cid> – 1 to 24 ▪ <qci> – 1 to 9 <DL_GBR>, <UL_GBR>,<DL_MBR>,<UL_MBR> -- Value in kbit/sec	Optional	Command supported in multimode or LTE-enabled targets <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h10+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CGCONTRDP=[<p_cid>]	Returns the relevant information of the PDP context.; values are reported as per specification; <p_cid> – 1 to 24	Optional	Command supported in multimode or LTE-enabled targets <p_cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h10+	3GPP
+CGSCONTRDP=[<cid>]	Returns the relevant information of the secondary PDP context.; values are reported as per specification; <cid> – 1 to 24	Optional	Command supported in multimode or LTE-enabled targets <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h10+	3GPP
+CGTFTRDP=[<cid>]	Returns the relevant information of the traffic flow template; values are reported as per specification; <cid> – 1 to 24	Optional	Command supported in multimode or LTE-enabled targets <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h10+	3GPP
+CGEQOSRDP=[<cid>]	Returns the relevant information of the Quality of Service parameters; values are reported as per specification; <cid> – 1 to 24	Optional	Command supported in multimode or LTE-enabled targets <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h10+	3GPP

Command	Description	[S1] requirement	Explanation	Supported packages	Applicable mode
+CEMODE=[<mode>]	Set the ME to operate according to the specified mode; values are reported as per specification; <mode> – 0 :PS mode of operation ▪ 1 – CS/PS mode 1 of operation ▪ 2 – CS/PS mode 2 of operation	Optional	Command supported in multimode or LTE-enabled targets. CS/PS mode is supported only when CSFB feature is present; only query and test command supported.	1h10+	3GPP
+CNMPSD	Indicates that no application on the MT is expected to exchange data.	Optional	Command supported in WCDMA and TDS modes.	MPSS DI 3.0+	3GPP

2.9 SMS commands

For UMTS, the SMS text and PDU mode AT commands are supported as indicated in Table 2-14. Command support will be as specified in [S2]. The message service failure result code +CMS ERROR are supported as specified in [S2].

Table 2-14 SMS text and PDU mode commands

Command	Description	[S2] requirement	Explanation	Supported packages	Applicable mode
+CSMS=<service>	Selects message service; values per specifications	Mandatory	Commands have no impact on 3GPP2 SMS	1h07+	3GPP

Command	Description	[S2] requirement	Explanation	Supported packages	Applicable mode
+CPMS= <mem1>, <mem2>, <mem3>	Preferred message storage; values per specifications ▪ <mem1> ▫ SM ▫ ME ▫ MT ▫ SR ▪ <mem2> ▫ ME ▫ MT ▫ SM ▫ SR ▪ <mem3> ▫ ME ▫ MT ▫ SM ▫ SR	Mandatory		1h07+	3GPP
+CMGF= <mode>	Message format; values per specifications	Mandatory		1h07+	3GPP
+CSCA= <sca>, <tosca>	Service center address; values per specifications	Mandatory		1h07+	3GPP
+CSMP= <fo>, <vp>, <pid>, <dcs>	Sets Text mode parameters; values per specification; GSM 7-bit, 8-bit and UCS2 data coding schemes supported	Mandatory		1h07+	3GPP
+CSDH= <show>	Shows Text mode parameters; values per specifications	Mandatory		1h07+	3GPP
+CSCB=[<mode> [,<mids> [,<dcss>]]]	Selects cell broadcast message types; values per specifications	Optional		1h07+	3GPP

Command	Description	[S2] requirement	Explanation	Supported packages	Applicable mode
+CNMI= <mode>, <mt>, <bm>, <ds>, <bfr>	New message indications to TE; values per specifications ▪ <mode> – 0 to 2 ▪ <mt> – 0 to 3 ▪ <bm> – 0, 2 ▪ <ds> – 0, 2 ▪ <bfr> – 0, 1	Mandatory when any new message indications implemented	<bm> value of 2 applies only in GSM mode	1h07+	3GPP
+CMGL= <stat>	Lists message; values per specifications	Optional		1h07+	3GPP
+CMGR= <index>	Reads message; values per specifications	Optional		1h07+	3GPP2
+CNMA	Acknowledges new message; values per specifications	Mandatory	Currently, supported in thin UI builds only	1h07+	3GPP
+CMGS= <da>, <toda>	Sends message; values per specifications	Optional		1h07+	3GPP
+CMSS= <index>, <da>, <toda>	Sends message from storage; values per specifications	Optional		1h07+	3GPP
+CMGW= <oa/da>, <tooa/toda>, <stat>	Writes message to memory; values per specifications	Optional		1h07+	3GPP
+CMGD= <index>, <deflag>	Deletes message; values per specifications	Optional		1h07+	3GPP
+CMGC=<fo>, <ct>[,<pid> [,<mn>[,<da> [,<toda>]]]] <CR>text is entered <ctrl-Z/ESC>	Sends command; values per specifications	Optional		1h07+	3GPP
+CMMS= [<n>]	Indicates more messages to send	Optional		1h07+	3GPP
+CSAS[=0]	Saves the settings of +CSCA, +CSMP, +CSCB into NV	Optional	Currently only one profile (0) is supported	1h07+	3GPP
+CRES[=0]	Restores the settings of +CSCA, +CSMP, +CSCB into NV	Optional	Currently only one profile (0) is supported	1h07+	3GPP

2.10 UMTS vendor-specific commands

For UMTS, the vendor-specific AT commands are supported as indicated in Table 2-15.

Table 2-15 Vendor-specific commands

Command	Description	Operation	Supported packages	Applicable mode
$QCCLR	Clears mobile error log	Accepts no arguments	1h07+	Any
&V	Dumps configuration parameters	Dumps the status of all AT parameters applicable to the current operating mode, including the single-letter parameters not otherwise readable. Accepts no arguments.	1h07+	Any
&C2	Circuit 109 (carrier detect pin) behavior – Wink	Setting winks (briefly transitions off, then back on) the R_m port carrier detect pin when data calls end	1h07+	Any
$QCTER= <rate>	Sets TE-DCE baud rate; baud rates supported are identical to +IPR command	Sets the TE-DCE rate at which DCE will accept commands; this data rate also becomes the default and is stored in NV RAM, changing the +IPR command default rate. Test command, i.e., $QCTER=?, returns the rates supported and query command, i.e., $QCTER?, returns the rate last issued by the $QCTER command or the default rate.	1h07+	Any
$QCDNSP= <address>	Sets primary DNS IP address	Sets the default primary IP address used for Domain Name Services (DNS); used only if no DNS server address is received over the air during PDP context activation. The value is stored in NVRAM.	1h07+	Any
$QCDNSS= <address>	Sets secondary DNS IP address	Sets the default secondary IP address used for DNS; used only if no DNS server address is received over the air during PDP context activation. The value is stored in NVRAM.	1h07+	Any

Command	Description	Operation	Supported packages	Applicable mode
$QCPDPP=<cid>, <auth_type>, <password>, <username>	Sets authentication for PDP-IP packet data calls ▪ <cid> – 1 to 24 ▪ <auth_type> ▫ 0 – None ▫ 1 – PAP ▫ 2 – CHAP	Defines authentication parameters on a per connection basis; the value of <auth_type> determines what additional parameters are required, as follows: ■ 0 – Neither username nor password accepted ■ 1 – Username and password accepted ■ 2 – Only password (secret) accepted Query command, i.e., $QCPDPP?, does not display password values and only displays username. <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
$QCPWRDN	Used to power down the UE	Returns OK and powers down the UE. Accepts no arguments.	1h07+ to 1h10 only	Any
$QCDGEN=<cid>,<data length>	Generates data over +CGACT activated PDP context ▪ <cid> – 1 to 24 ▪ <data length> – Greater than 21	Supported only during PDP context activated by +CGACT <cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h07+	3GPP
$QCSIMSTAT=<n>	Disables/enables the display of the status of the SIM <n> ▪ 0 – Disables the feature ▪ 1 – Enables the feature	Upon enabling, SIM status is sent as an unsolicited result code onto the terminal. Status: ▪ SIM INIT ▪ SIM ERROR ▪ UNKNOWN	1h07+	3GPP
$QCCNMI = <mode>, <mt>, <bm>, <ds>, <bfr>	New message indications to TE; values per specifications ▪ <mode> – 0 to 2 ▪ <mt> – 0 to 3 ▪ <bm> – 0, 2 ▪ <ds> – 0, 2 ▪ <bfr> – 0, 1	This is a special case of [S2] CNMI. The behavior is exactly same for all the parameters except for <mt>=2 where the route configuration will be set as similar to <mt>=0; it does not change the NV-830 settings.	1h07+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCPINSTAT?	Retrieves all PIN status of ME	Only the Read command is valid	1h07+	3GPP
$QCPDPLT = <n>	Enables/disables tolerance for long delays is PDP call setup <n> ▪ 0 – Disables waiting ▪ 1 – Enables waiting		1h07+	3GPP
$QCSYSMODE?	Returns current system mode of ME	Provides details about the support of LTE/HSUPA/ HSDPA; only the Read command is valid; 3GPP2 support added for MPSS NI 3.0 and later	1h07+	Any

Command	Description	Operation	Supported packages	Applicable mode
$QCPDPCFGE =<profile_id>, < APN disable flag>, <timer_value>,< APN class>,<APN bearer >, <max_pdn_conn_per_block>, <max_pdn_conn_timer>, <pdn_req_wait_timer>, <emergency_calls_supported>, <operator_reserved_pco>, <mcc>, <mnc>	Sets PDN teardown time interval ▪ <profile_id> – 1 to 24 ▪ <enable/disable> ▫ 0 – Enable ▫ 1 – Disable ▪ <timer_value> 0 – 122820 ▪ <APN class> – 0 – 5 <APN bearer> ▫ 0x1 – GSM type ▫ 0x2 – UMTS type ▫ 0x4 – LTE type ▫ 0x8 –TDS type and ▫ FF – All types ▪ <max_pdn_conn_per_block> 0 – 1023 ▪ <max_pdn_conn_timer> 0 – 3600 ▪ <pdn_req_wait_timer> 0 – 1023 ▪ <emergency_calls_supported> 0 – 1 ▪ <operator_reserved_pco> 65280 – 65535 ▪ <mcc> 0 – 999 ▪ <mnc> 0 - 999	<APN bearer> support added MPSS NI 3.0 and later <profile_id> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	1h10+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$ECALL= <ecall_session>, [<activation_type> , <type_of_call>]	Triggers an emergency call to the network; based on the configuration selected, it can be used to either trigger a manual or automatic voice or emergency call. ▪ <ecall_session> ▫ 1 – Start eCall session ▫ 0 – End/stop eCall session ▪ <type_of_call > ▫ 0 – Test eCall (originate voice call) ▫ 1 – Emergency eCall (originate EMERGENCY call) ▪ <activation_type> ▫ 0 – Manual eCall ▫ 1 – Automatic eCall		1h10+	3GPP
+PACSP	Indicates network selection menu availability to user or not (ENS support); valid responses are: ▪ +PACSP0 – If the menu should be disabled ▪ PACSP1 – If enabled		1h09+	3GPP
+CQI=<n>	Enable/disable reporting of channel quality indicator: <n>=0-1 ▪ 0 – Enable ▪ 1 – Disable		1h10+	3GPP
$QCCTM=<n>	Enable/disable cellular text telephone modem support: <n>=0-1 ▪ 0 – Disable ▪ 1 – Enable		1h07+	Any

Command	Description	Operation	Supported packages	Applicable mode
$QCBANDPREF=<pref_term >,<band_pref>	Ability to set/get the band preferences: ▪ <pref_term > ▫ 0 – Until a power cycle ▫ 1 – Permanent mode change. ▪ <band_pref> – “1,2,3..”		1h09+	Any
^PREFMODE=<pref_mode>	Ability to set/get the Network mode preferences ▪ <pref_mode> – 0,2,4,8		1h09+	3GPP2
*CNTI=<n>	Controls the listing of the technologies UE is capable of ▪ <n>: 0,1,2		1h09+	3GPP
^CAVE=<randu>	Sends CAVE authentication-related parameter (RANDU) to data card Data card returns the response ▪ <randu>: 3 bytes		1h08+	3GPP2
^MD5=<chapid>,<chapchallengelength><CR><chapchallenge><ctrl-Z/ESC>	Request data card to do MD5 authentication ▪ <chapid> – 0 to 255 ▪ <chapchallenge length> – 1 to 253 ▪ <chapchallenge> – Defined in the previous parameter in bytes		1h08+	3GPP2
^VPM	Generates key		1h08+	3GPP2
^SSDUPD=<randssd>	Sends RANDSSD to data card Data card calculates and reports result ▪ <randssd> – 7 bytes		1h08+	3GPP2
^SSDUPDCFM=<authbs>	Sends updated SSD confirmation (AUTHBS) ▪ <authbs> – 18 bits		1h08+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
^UIMAUTH=?	Check authentication algorithm supported: <auth>"CAVE", "MD5".		1h08+	3GPP2
^GSN	Returns MEID/ESN of the data card		1h08+	3GPP2
^CPIN	This is similar to +CPIN command in 3GPP2 mode.		1h08+	3GPP2
^CPBR	This is similar to +CPBR command in 3GPP2 mode.		1h08+	3GPP2
^CPBW	This is similar to +CPBW command in 3GPP2 mode.		1h08+	3GPP2
^CPBF	This is similar to +CPBF command in 3GPP2 mode.		1h08+	3GPP2
$QCDEFPROF= < family > < subs id > < profile id >	Sets the given profile number as default profile for the family of the specified technology and subscription. ▪ <profile_id> – 1 to 16 ▪ <Subs ID> 1 to 3 – For non-DSDS target SUBS ID is limited to 1. ▪ <family> 1 to 16		MPSS NI 1.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCMRUE= <index> <rat> <band> <channel> or <PLMN>	Command is used to edit/set MRU database. ▪ <index> – 0 to 11 ▪ <rat> ▫ 0 – CDMA ▫ 1 – HDR ▫ 2 – GSM ▫ 3 – WCDMA ▫ 4 – LTE ▪ <band> ▫ Band CDMA/ HDR – 0 to 19 (except 2) ▫ Band GSM – 0 to 8 ▫ Band UMTS – 0 to 9 ▫ Band LTE – 0 to 26 <channel> – 0 to 2047 ▪ <PLMN> String type PLMN ID	This command edits the MRU database on the UE. When <rat> is not CDMA/ HDR then fourth argument is PLMN id. For entering automatic 3GPP MRU entries, PLMN ID should be a NULL string.	MPSS NI 1.0+	Any
$QCMRUC	Command is used to clear/delete MRU database <NA>Name only supported		MPSS NI 1.0+	Any

Command	Description	Operation	Supported packages	Applicable mode
$QCAPNE= <Profile ID> <APN Class> <APN Name> <PDP Type> <APN Bearer type> <APN Flag> < APN Timer>	Command is used to edit APN values in APN table. ▪ <Profile ID> – 1 to 24 ▪ <APN class> – 0 to 4 ▪ <APN Name> – String type, maximum length is 100 ▪ <PDP Type> – String type, maximum length is 8 ▪ <APN Bearer type> – String type, maximum length is 3 ▪ <APN Flag> <enable/ disable> – 0 to 1 ▪ <timer_value> 0 to 122820	<Profile ID> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	MPSS NI 1.0+	3GPP
$QCPDPIMSCFGE= <Profile ID>, <P-CSCF Address Flag>, <DHCP Flag>, <CN Flag>	Command is used to edit PDP profile registry ▪ <Profile id> – 1 to 24 ▪ Flags have values 0 to 1 <enable disable>	<Profile id> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	MPSS NI 1.0+	3GPP
$QCCLAC	Lists all available AT commands		MPSS NI 2.0+	Any
^SPN	This command is used to display the brand information from EF-SPN. <NA>Name only supported.		MPSS NI 3.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCRMCALL =<Action>, <Instance> [,<IP Type> [,<Tech Pref > [,<umts profile number> [,<cdma profile number > [,<APN>]]]]]	Command triggers an RmNet call based on <Action> parameter which is typically a start of an RmNet Call or stop of a RmNet call. ▪ < Action > ▫ 0 – Stop ▫ 1 – Start ▪ <Instance> 1 to RMNET_ NUM_LAPTOP_ INSTANCES ▪ <IP Type> ▫ 1 – Ipv4 ▫ 2 – Ipv6 ▫ 3 – Ipv4v6 ▪ <Tech Pref> ▫ 1 – 3GPP2 ▫ 2 – 3GPP ▪ <umts_profile> 1 to 24 ▪ <APN > String type, maximum length is 100	<umts_profile> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	MPSS NI 3.0+	Any
$QCDRX= <drx coefficient>	This command provides the ability to set/get the drx coefficient. <drx_coefficient> – 0 is not specified by MS ▪ 6 CN = 6, T = 32 ▪ 7 CN = 7, T = 64 ▪ 8 CN = 8, T = 128 ▪ 9 CN = 9, T = 256		MPSS NI 3.0+	Any
^ DSCI= <dsci_val >	This command enables/disables ^DSCI URC(call event). ^DSCI:<id>, <idr>, <stat>,<type>, <mpty>,<number>, <num_type>, [<bs_type>][,cause]		MPSS NI 2.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
^CARDMODE	^CARDMODE command retune mode of the card among SIM or USIM. <NA>Name only supported.		MPSS NI 2.0+	Any
^SYSCONFIG= <mode>, <acqorder>, <roam>, <srvdomain>	This command set system configuration. ▪ <mode> system mode reference: ▫ 2– Automatically select ▫ 13 – GSM ONLY ▫ 14 – WCDMA only ▫ 15 – TD-SCDMA only ▫ 16 – No change ▪ <acqorder> – network accessing order reference ▫ 0 – Automatically ▫ 1 – GSM first, UTRAN second ▫ 2 – UTRAN first, GSM second ▫ 3 – No change ▪ <roam> – roaming support ▫ 0 – Not support ▫ 1 – Can roam ▫ 2 – No change ▪ <srvdomain> – Domain configuration ▫ 0 – CS_ONLY ▫ 1 – PS_ONLY ▫ 2 – CS_PS ▫ 3 – ANY ▫ 4 – No change		MPSS NI 2.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
^SYSINFO	The command inquires current system message, e.g., system service status, domain, roam, system mode, UIM card status, etc. <NA>Name only supported.		MPSS NI 2.0+	Any
$QCSQ	The command returns RSCP, ECIO, SIR, PATHLOSS, and RSSI if present in the following format $QCSQ: <rscp>,<ecio>, <sir>,<pathloss>, <rssi>. Name only and test command supported.		MPSS NI 3.0+	3GPP
$QCRPW	Reporting received radio signal power in a scale of 0 to 75. Name only and test command supported.		MPSS NI 3.0+	3GPP
$QCANTE	Reporting number of antenna bars in a scale of 0 to 4 based on RSSI value. Name only and test command supported.		MPSS NI 3.0+	3GPP
$QCDGEN= <Profile ID> <length>	Command is used to send a specified quantity of PS data as IP packet(s) over a primary PDP-IP context already activated by +CGACT command. Minimum length of packet is 21. Action command only supported.		1h07+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCSIMAPP= <QCSIMAPPVAL>	This command is applicable only for Multi SIM targets. User can select a SIM slot. <QCSIMAPPVAL> parameter range is based on number of SIMs supported. ▪ <QCSIMAPPVAL> ▫ 0 – SIM slot 1 ▫ 1 – SIM slot 2 ▫ 2 – SIM slot 3		1h08+	Any
$QCPBMPREF= <Preference>	This command is used to choose the application and local or global phonebook type. ▪ <Preference> ▫ 0 – Slot 1 global phonebook ▫ 1 – App 1 local phonebook ▫ 2 – Slot 2 global phonebook ▫ 3 – App 2 local phonebook ▫ 4 – Slot 3 global phonebook ▫ 5 – App 3 local phonebook		1h08+	Any
$QCRSRP	This command is used to display neighbor cell information cell ID, EARFCN, RSRP. <NA|QU> only read command supported.		MPSS NI 4.0+	3GPP
$QCRSRQ	This command is used to display neighbor cell info, Cell id, EARFCN, RSRQ. <NA|QU> only read command supported.		MPSS NI 4.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCACQDBC	Used to clear ACQ database. Name only command supported.		MPSS DI 2.0+	3GPP
$QCATMOD	Used to indicate the SIOLIB AT command processing state. Name only, read and test command supported.		MPSS NI 5.0+	Any
$QCCOPS	Same as +COPS command and only exception is that PLMN can be specified in automatic PLMN selection mode as well.		MPSS DI 2.0+	3GPP
$QCHCOPS	A version of +COPS read command for hybrid stack. <NA|QU> only read command supported.		MPSS DI 2.0+	3GPP
$QCHCREG	A version of +CREG read command for hybrid stack. <NA|QU> only read command supported.		MPSS DI 2.0+	3GPP
^MODE=<present>	Controls the presentation of ^MODE unsolicited result code. <present> 0 - Disable 1 - Enable		MPSS DI 1.0+	3GPP
$QCSIMT	Used to indicate SIM type. Name only and read command supported.		MPSS DI 3.0+	Any

Command	Description	Operation	Supported packages	Applicable mode
$QCNSP=<mode_pref>,<net_sel_pref>,<acq_order_pref>	Configures the network selection preferences. <mode_pref> 0 - Automatic 1 - GSM 2 - WCDMA 6 - LTE <net_sel_pref> 0 - Automatic 1 - Manual 2 - Limited Service <acq_order_pref> 0 - Automatic 1 - GSM 2 - WCDMA 3 – LTE		MPSS DI 3.0+	3GPP
$QCRCIND=<present>	Controls the presentation of unsolicited remote call indications. <present> 0 - Disable 1 - Enable		MPSS DI 3.0+	3GPP
$QCCSGCOPS=<cops_mode>,<plmn_format>,<plmn_id>,<cops_rat>,<csg_id>,<csg_rat>	Used for manual PLMN search and PLMN selection of CSG cells. <csg_id> - 4 byte unsigned number <csg_rat> 0 – Automatic 1 – GSM 2 – WCDMA 3 – LTE		MPSS DI 3.0+	3GPP
$QCPDPCFGEXT=<cid>,<mbim_context>,<roaming disable flag value>,<roaming disable timer value>	Used to set and get MBIM context type. <cid> - 1 to 24 <mbim_context> - 4 byte unsigned number <roaming disable flag> - 0, 1 <roaming disable timer value> - 0 to 255	<cid> values 17 to 24 are supported from MPSS JO 1.0+ onwards.	MPSS DI 3.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCVOIPM=<session_expires>,<min_session_expiry>,<rtp_rtcp_inactivity_timer>,<ringing_timer>,<ringback_timer>,<amr_wb_enable>,<scr_amr_enable>,<scr_amr_wb_enable>, <amr_mode_str>, <amr_wb_mode_str>	Used to configure IMS parameters. <session_expires> - 90 to 65535 <min_session_expiry> - 90 to 65535 <rtp_rtcp_inactivity_timer> - 0 to 20 <ringing_timer> - 30 to 45 <ringback_timer> - 35 to 50 <amr_wb_enable> 0 - Disable 1 - Enable <scr_amr_enable> 0 - Disable 1 - Enable <scr_amr_wb_enable> 0 - Disable 1 – Enable <amr_mode_str> - String type <amr_wb_mode_str> - String type		MPSS DI 4.0+	3GPP

Command	Description	Operation	Supported packages	Applicable mode
$QCPRFCRT=OPRT:<Value>,PERSIST:<Value>,TECH:<Value>,PID:<Value>,APNNAME:<Value>,PDPTYPE:<Value>	Used to create or delete any 3GPP2/EPC profile. This command uses tag:<value> syntax for all the parameters which allows users to specify these parameters in any order. OPRT, PERSIST, TECH, PID, APNNAME and PDPTYPE are tags defined for parameters in this command. OPRT:<Value> 1 - Create profile 2 - Delete profile TECH:<Value> 1 - 3GPP2 2 - EPC (Not recommended) PERSIST:<Value> 1 - Non persistent 2 - Persistent Mandatory in case of creation PID:<Value> - 100 to 178 range of allowed profile numbers APNNAME:<Value> - String type PDPTYPE:<Value> - String type		MPSS BO 1.0+	Any

Command	Description	Operation	Supported packages	Applicable mode
$QCPRFMOD=PID:<Value>,ParamName:<ParamValue> There can be 15 profile parameters with syntax ParamName:<ParamValue>.	Used to configure parameters for 3GPP2/EPC profile. This command uses tag:<value> syntax for all the parameters which allows users to specify these parameters in any order except for first parameter. PID and various ParamName tags defined for parameters in this command. First parameter of this command must be PID. PID:<Value> - 100 to 178 range of allowed profile numbers		MPSS BO 1.0+	Any

[2] Unless explicitly mentioned, all the actions (test, read, and action) are accepted on these commands.

2.11 Fax commands

The Fax commands specific to GSM are supported as indicated in Table 2-16. Command support will be as specified in [S6].

Table 2-16 Fax ITU-T T.31 commands for GSM

Command	Description	ITU-T T.31 requirement	Explanation	Supported packages	Applicable mode
+FCLASS=<value>	Sets the service class of a facsimile DCE <value> – 0, 1	Mandatory		1h07+	3GPP
+FTS= <time>	Transmits silence; directs the DCE to stop transmitting for the specified amount of time (in 10 ms increments) <time> – 0 to 255	Mandatory		1h07+	3GPP

Command	Description	ITU-T T.31 requirement	Explanation	Supported packages	Applicable mode
+FRS= <time>	Receives silence; directs the DCE to listen for silence from the remote end and report back OK when silence has been detected for the specified amount of time (in 10 ms increments); command is abortable <time> – 0 to 255	Mandatory		1h07+	3GPP
+FTM= <mod>	Transmits message; directs the DCE to transmit T.30 facsimile message data using the specified modulation rate (mod * 100 bps) <mod> – 72, 73, 74, 96, 97, 98	Mandatory		1h07+	3GPP
+FRM= <mod>	Receives message; directs the DCE to receive T.30 facsimile message data using the specified modulation rate (mod * 100 bps); command is abortable <mod> – 72, 73, 74, 96, 97, 98	Mandatory		1h07+	3GPP
+FTH= <mod>	Transmits HDLC; directs the DCE to transmit T.30 HDLC data using the specified modulation rate (mod * 100 bps) <mod> – 3	Mandatory		1h07+	3GPP
+FRH= <mod>	Receives HDLC; directs the DCE to receive T.30 HDLC data using the specified modulation rate (mod * 100 bps); command is abortable <mod> – 3	Mandatory		1h07+	3GPP

Command	Description	ITU-T T.31 requirement	Explanation	Supported packages	Applicable mode
+FAR= <value>	Adaptive rate control; disables the DCE’s ability to adaptively detect the selected message carrier or V.21 Control message and to adjust +FRM processing accordingly <value> – 0	Mandatory	Adaptive rate control is disabled for the OTA interface	1h07+	3GPP
+FDD= <value>	Doubles escape character replacement; controls the DCE to use the <DLE><SUB> pair to encode consecutive <1/0><1/0> in data <value> – 0, 1	Mandatory		1h07+	3GPP
+FCL= <time>	Carrier loss timeout; sets the duration (<time>*100 ms) used by DCE to terminate the session if no activity is detected on the carrier, i.e., the OTA interface <time> – 0 to 255	Mandatory		1h07+	3GPP
+FIT= <time>, <action>	DTE inactivity timeout; sets the duration (in seconds) used by the DCE to terminate the session if the DTE fails to respond ▪ <time> – 0 to 255 ▪ <action> – 0, 1	Mandatory	<action> 0 and 1 are treated the same, i.e., terminate the T.31 session	1h07+	3GPP

2.12 Synchronous data mode commands

For UMTS, the Synchronous Data mode commands specific to WCDMA are supported as indicated in Table 2-17. Command support will be as specified in [S5].

Table 2-17 ITU-T V.80ter synchronous data mode commands

Command	Description	ITU-T V.80ter	Explanation	Supported packages	Applicable mode
+ES= <orig_rqst>, <orig_fbk>, <ans_fbk>	Enables Synchronous mode; values per specifications ▪ <orig_rqst> – 6 ▪ <orig_fbk> – undefined ▪ <ans_fbk> – 8	Mandatory		1h07+	3GPP
+ESA= <trans_idle>, <framed_idle>, <framed_un_ov>, <hd_auto>, <crc_type>, <nrzi_en>, <sync1>, <sync2>	Preferred message storage; values per specifications ▪ <trans_idle> – 0 ▪ <framed_idle> – undefined ▪ <framed_un_ov> – Undefined ▪ <hd_auto> – undefined ▪ <crc_type> – 0 ▪ <nrzi_en> – 0 ▪ <sync1> – 0 to 255 ▪ <sync2> – undefined	Mandatory		1h07+	3GPP

2.13 IS-707 AT command set for AMSS

All of the tables in this section were copied from IS-707 specifications, as indicated in the table titles. Table 2-18 through Table 2-20 are sheets 1 through 3 of [S9] Table 7.2-1 of extended AT configuration commands.

Table 2-18 [S9] Table 7.2-1, extended AT configuration commands (1 of 3)

Cmd	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+EB	Break handling in error control operation. This extended-format compound parameter is used to control the manner of V.42 operation on the PSTN link (if present in the IWF).	Async – Req Pkt – Opt	Remote async/fax command Not relevant for packet service	1h07+	3GPP2

Cmd	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+EFCS	This extended-format numeric parameter controls the use of the 32-bit frame check sequence option in V.42 on the PSTN link (if present in the IWF).	Async – Req Pkt – N/A	Remote async/fax command Not relevant for packet service	1h07+	3GPP2
+ER	Error control reporting. This extended-format numeric parameter controls whether the extended-format +ER: intermediate result code is transmitted from the IWF over the U_m interface.	Async – Req Pkt – Opt	Remote async/fax command Not relevant for packet service	1h07+	3GPP2
+ESR	This extended-format numeric parameter controls the use of the selective repeat (SREJ) option in V.42 on the PSTN link (if present in the IWF).	Async – Req Pkt – N/A	Remote async/fax command Not relevant for packet service	1h07+	3GPP2
+ETBM	This extended-format compound parameter controls the handling of data remaining in IWF buffers upon service termination.	Async – Req Pkt – Opt	Remote async/fax command Not relevant for packet service	1h07+	3GPP2

Parameter values per IS-131

Table 2-19 [S9] Table 7.2-1, extended AT configuration commands (2 of 3)

Cmd	Description	IS-707 requirement and implementation status	Explanation	Supported packages	Applicable mode
+MA	Modulation automode control. This extended-format compound parameter is a list of modulations that the base station may use to connect with the remote DCE in Automode operation, for answering or originating data calls, as additional alternatives to the modulation specified in the +MS command.	Async – Req Pkt – N/A	Remote async/fax command	1h07+	3GPP2

Cmd	Description	IS-707 requirement and implementation status	Explanation	Supported packages	Applicable mode
+MR	Modulation reporting control. This extended-format numeric parameter controls whether the extended-format +MCR:<carrier> and +MRR:<rate> intermediate result codes are transmitted from the IWF to the MS (MS).	Async – Req Pkt – N/A	Remote async/fax command	1h07+	3GPP2
+MS	Modulation selection. This extended-format compound parameter is used to control the manner of operation of the modulation capabilities in the IWF.	Async – Req Pkt – N/A	Remote async/fax command	1h07+	3GPP2

Parameter values per IS-131

Table 2-20 [S9] Table 7.2-1, extended AT configuration commands (3 of 3)

Cmd	Description	IS-707 requirement and implementation status	Explanation	Supported packages	Applicable mode
+MV18R	V.18 reporting control. This extended-format numeric parameter controls whether the extended-format +MV18R result code is transmitted from the IWF to the MS.	Async – Opt Pkt – N/A	Remote async/fax command	1h07+	3GPP2
+MV18S	V.18 selection. This extended-format compound parameter is used to control the manner of operation of the V.18 capabilities (if present in the IWF).	Async – Opt Pkt – N/A	Remote async/fax command	1h07+	3GPP2

Parameter values per IS-131

Table 2-21 and Table 2-22 are sheets 1 and 2 of IS-707.3, Table 7.3.1-1 of fax parameters.

Table 2-21 [S9] Table 7.3.1-1, fax parameters (1 of 2)

Parameter	Value per	Description	Explanation	Supported packages	Applicable mode
+FAA	[S10]	Adaptive-answer parameter; see +FCLASS	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FAP	[S11]	Addressing and Polling capabilities parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FBO	[S10]	Phase-C data-bit-order parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FBS	[S10]	Buffer size; read-only parameter	Local	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FBU	[S10]	HDLC-frame-reporting parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FCC		DCE-capabilities parameters	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
VR	[S10]		Vertical-resolution subparameter		
[BR]	0 1 2 3[1]		Bitrate subparameter ▪ 2400 bps ▪ 4800 bps ▪ 7200 bps ▪ 9600 bps		
WD	[S10]		Page-width subparameter		
[LN]	[S10][2]		Page-length subparameter		
[DF]	[S10][2]		Data-compression-format subparameter		
[EC]	[S10][2]		Error-correction subparameter		
BF	[S10]		Binary-file-transfer subparameter		
ST	[S10]		Scan-time-per-line subparameter		3GPP2

Parameter	Value per	Description	Explanation	Supported packages	Applicable mode
[+FCLASS]	0 1 2.0[3]	Service-class selection parameter ▪ Class-0 ▪ [Class-1 support unavailable] ▪ Class-2.0 fax service ([S10])	Remote; mobile will return ERROR for +FCLASS=1	1h07+ Deprecated for 3GPP2 mode from MPSS TH 1.0+ onwards.	3GPP
+FCQ	[S10]	Copy-quality-checking parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
[+FCR]	[S10][2]	Capability-to-receive parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FCT	[S10]	DTE Phase-C timeout parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FEA	[S10]	Phase-C received EOL-alignment parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2

[1] Use of option 3 may cause degradations in the quality of certain faxes.

[2] Some values for this parameter are optional in [S10]. In this standard, all parameters of this command shall be supported.

[3] Class 2.0 represents [S10].

[4] Support of packet protocol is optional.

[5] Values 4 and 5 of this parameter are optional.

Table 2-22 [S9] Table 7.3.1-1, fax parameters (2 of 2)

Parameter	Value per	Description	Explanation	Supported packages	Applicable mode
+FFC	[S10]	Format-conversion parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FHS	[S10]	Call-termination-status parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FIE	[S10]	Procedure-interrupt-enable parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2

Parameter	Value per	Description	Explanation	Supported packages	Applicable mode
+FIS	[S10]	Current-session negotiation parameters	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
[+FLI]	[S10][2]	Local-ID-string parameter (TSI or CSI)	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FLO	[S10][2]	Flow-control-select parameter	Local	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FLP	[S10]	Indicate-document-to-poll parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FMI	[S10]	Request DCE manufacturer identification		1h07+	3GPP2
+FMM	[S10]	Request DCE model		1h07+	3GPP2
+FMR	[S10]	Request DCE revision		1h07+	3GPP2
[+FMS]	[S10][2]	Minimum-phase-C-speed parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FNR	[S10]	Negotiation-message-reporting control parameters	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FNS	[S10]	Nonstandard-frame FIF parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FPA	[S11]	Selective polling address parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
[+FPI]	[S10][2]	Local-polling-ID-string parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FPR	[S10]	Serial-port-rate-control parameter	Local; mobile will accept only 19200	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
[+FPS]	[S10][5]	Page-status parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2

Parameter	Value per	Description	Explanation	Supported packages	Applicable mode
+FPW	[S11]	Password parameter (sending or polling)	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
[+FRQ]	[S10][2]	Receive-quality-threshold parameters	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FRY	[S10]	ECM-retry-value parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FSA	[S11]	Subaddress parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
[+FSP]	[S10][2]	Request-to-poll parameter	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2

[1] Use of option 3 may cause degradations in the quality of certain faxes.

[2] Some values for this parameter are optional in [S10]. In this standard, all parameters of this command shall be supported.

[3] Class 2.0 represents [S10].

[4] Support of packet protocol is optional.

[5] Values 4 and 5 of this parameter are optional.

Table 2-23 [S9] Table 7.3.2-1, fax action commands

Command	Description	Explanation	Supported packages	Applicable mode
+FDR	Receive Phase-C data	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FDT	Transmit Phase-C data	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FIP	Initialize facsimile parameters	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+FKS	Terminate session	Remote	1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2

Table 2-24 through Table 2-27 are sheets 1 through 4 of [S9], Table 7.4.1-1 of CDMA AT parameter commands.

Table 2-24 [S9] Table 7.4.1-1, CDMA AT parameter commands (1 of 4)

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CXT=<value>	Cellular extension ▪ 0 – Do not pass unrecognized commands to the IWF ▪ 1 – When detecting an unrecognized AT command, open transport layer connection and pass unrecognized command to the IWF	Async – Req Pkt – N/A		1h07+	3GPP2
+CFG="<string>"	Configuration string The string (up to and including the termination character) will be stored by the MT2 and sent to the base station prior to dialing. Each transmission of an AT+CFG command from the TE2 replaces the contents of the previous string. The string may be up to 248 characters.	Async – Req Pkt – N/A No action taken		1h07+	3GPP2
+CAD?	Query analog or digital service Returns: ▪ 0 – If no service is available ▪ 1 – If CDMA digital service is available ▪ 2 – If TDMA digital service is available ▪ 3 – If analog service is available (values 4 to 255 reserved)	Async – Opt Pkt – Opt	2 (TDMA) is not supported.	1h07+	3GPP2

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CDR	U_m interface data compression reporting This extended-format numeric parameter controls whether the extended-format +CDR: intermediate result code is transmitted by the MT2. The result code is the same as for the [S12] +DR: result code.	Async – Req Pkt – N/A No action taken		1h07+	3GPP2

Table 2-25 [S9] Table 7.4.1-1, CDMA AT parameter commands (2 of 4)

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CDS	U_m Interface Data Compression: This extended-format compound parameter controls the V.42*bis* data compression function on the U_m interface. The command format is the same as for the [S12] +DS command.	Async – Req Pkt – N/A No action taken	Currently, Qualcomm mobile does not support V.42bis compression. Mobile will only accept 0 as a valid setting.	1h07+	3GPP2

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CRM=<value>	Set R_m interface protocol ▪ 0 – Asynchronous data or fax ▪ 1 – Packet data service, relay Layer R_m interface ▪ 2 – Packet data service, network layer R_m interface, PPP ▪ 3 – Packet data service, network Layer R_m interface, SLIP ▪ 4 – STU-III service ▪ 5 to 127 – Reserved for future use ▪ 128 to 255 – Reserved for manufacturer-specific use **Note:** The default value for the +CRM parameter shall be 0 if this value is supported by the MT2. If 0 is not supported, the default +CRM value shall be manufacturer specific.	Async – Req Pkt – Req	Mode selection occurs automatically based on data received	1h07+	3GPP2

Table 2-26 [S9] Table 7.4.1-1, CDMA AT parameter commands (3 of 4)

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CQD= <value>	Command state inactivity timer (see 3.9.1.3) ▪ 0 – Ignored ▪ 1 to 255 – Release call after 5x<value> sec have elapsed without activity. The default <value> shall be 10, corresponding to 50 sec.	Async – Req Pkt – N/A No action taken	Remote async/fax command	1h07+	3GPP2
+CCRC=[1] <value>	Cellular result codes (see Table 7.4.2-1) ▪ 0 – Disable cellular result codes ▪ 1 – Enable cellular result codes	Async – Req Pkt – N/A	Remote async/fax	1h07+	3GPP2

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CMIP?	MS IP Address Read-only; Returns the temporary IP address of the MS	Async – Req Pkt – N/A		1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+CBIP?	Base Station IP Address Read-only; Returns the base station's IP address	Async – Req Pkt – N/A		1h07+ Deprecated from MPSS TH 1.0+ onwards.	3GPP2
+QCCSS?[1]	Serving System. Read-only. Returns <BandClass>,<Band>, <SID> Band Class ▪ C – MS is registered with a cellular system ▪ P – MS is registered with a PCS system ▪ Band ▪ CA – MS is registered with cellular A-band system ▪ CB – MS is registered with cellular B-band system ▪ PA – MS is registered with PCS A-band system ▪ PB – MS is registered with PCS B-band system ▪ PC – MS is registered with PCS C-band system ▪ PD – MS is registered with PCS D-band system ▪ PE – MS is registered with PCS E-band system ▪ PF – MS is registered with PCS F-band system ▪ Z – MS is not registered ▪ SID ▪ 0 to 16383 – MS is registered with the system indicated ▪ 99999 – MS is not registered	Async – Req Pkt – N/A		1h07+	3GPP2

[1]Command name has changed because the same command name exists in [S1].

Table 2-27 [S9] Table 7.4.1-1, CDMA AT parameter commands (4 of 4)

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CCSQ[1]	Query received signal quality; Returns the Signal Quality Measure <SQM> and the Frame Error Rate <FER> as follows: ▪ <SQM> ▫ 0 to 31 – SQM (see note below) ▫ 99 – SQM is not known or is not detectable ▫ All other values are reserved ▪ <FER> ▫ 0 – <0.01% ▫ 1 – 0.01% to less than 0.1% ▫ 2 – 0.1% to less than 0.5% ▫ 3 – 0.5% to less than 1.0% ▫ 4 – 1.0% to less than 2.0% ▫ 5 – 2.0% to less than 4.0% ▫ 6 – 4.0% to less than 8.0% ▫ 7 – ≥8.0% ▫ 99 – <FER> is not known or is not detectable. All other values are reserved.	Async – Req Pkt – N/A	—	1h07+	3GPP2
+QCMUX = <n>	Select multiplex option ▪ 1 – Option 1 ▪ 2 – Option 2	Async – Req Pkt – N/A		1h07+	3GPP2
+CFC=<value>	U_m interface fax compression ▪ 0 – No compression ▪ 1 – V.42bis compression with parameters as set by the +CDS command ▪ 2 – Modified the modified read compression	Async – Req Pkt – N/A No Action Taken	Current Qualcomm mobile does not support V.42bis compression. Mobile will only accept 0 as a valid parameter.	1h07+	3GPP2

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+QCPIN= <pin>, <newpin>	This is similar to +CPIN command in 3GPP2 mode.			MPSS NI 2.0+	3GPP2
+QCLCK= <fac>, <mode>, <passwd>, <class>	This is similar to +CLCK command in 3GPP2 mode. Supported facilities are “SC” and “FD”.			MPSS NI 2.0+	3GPP2
+QCPWD= <fac>, <oldpwd>, <newpwd>	This is similar to +CPWD command in 3GPP2 mode. Supported facilities are “SC” and “P2”.			MPSS NI 2.0+	3GPP2
+QCIMI	This is similar to +CIMI command in 3GPP2 mode.<NA> Name only supported.			MPSS NI 2.0+	3GPP2

The exact meaning of the SQM is manufacturer-defined. The lowest quality reported by SQM is defined as value 00. The highest quality reported by SQM is defined as value 31.

Table 2-28 [S9] Table 7.4.1-2, cellular AT command extensions in support of voice services

Command	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CHV<value>	Hang-up voice ▪ 0 – Hang-up voice call ▪ 1 to 255 – Reserved	Async – Req Pkt – N/A		1h07+	3GPP2
+CDV <dial string>	Dial command for voice calls The format of <dial string> is identical to that for the ATD command. This command does not cause the MT2 to change to the online state.	Async – Req Pkt – N/A		1h07+	3GPP2

1

Table 2-29 [S9] Table 7.4.1-3, cellular AT commands for packet data services

Command	Description	Explanation	Supported Packages	Applicable mode
+CTA=<value>	Set/read/test U_m packet data inactivity timer ▪ 0 – Traffic channel not released during inactive periods ▪ 1 to 255 – Release the traffic channel after <value> 1-sec intervals have elapsed since last sending or receiving RLP data frames on the U_m interface ▪ 20 – (Default)	Relevant only for packet service operation	1h07+	3GPP2

2

3

Table 2-30 [S9] Table 7.4.2-1, cellular result codes

Result code	Description	IS-707 requirement	Explanation	Supported packages	Applicable mode
+CPROG: ANSWER	Indicates remote DCE has answered	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
+CPROG: BONGTONE	Billing tone was detected	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
+CPROG: DIALING <number>	Indicates PSTN dialing	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
+CPROG: DIALTONE	Dial tone was detected	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
+CPROG: QUIET ANSWER	Indicates quiet answer	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
+CPROG: RINGING	Indicates PSTN ringing	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
+CPROG: VOICE	Voice detected on the PSTN connection	Async – Req Pkt – N/A Mobile supports	Remote async/fax command	1h07+	3GPP2
RING <service option>	Specifies active service option; the <service option> shall be “ASYNC,” “FAX,” or “STU-III”	Async – Req Pkt – N/A		1h07+	3GPP2

2.14 Proprietary CDMA AT command set

Table 2-31 lists the CDMA vendor-specific AT commands.

Table 2-31 CDMA vendor-specific AT commands

Command	Description	Operation	Supported packages	Applicable mode
$QCCAV	Answer incoming voice call	Provides a means to answer an incoming voice call via an AT command; accepts no arguments	1h07+	Any
$QCCHV	Hang up incoming voice call	Provides a means to hang up an incoming voice call via an AT command; accepts no arguments	1h07+	Any
$QCQNC	Enable/ disable Quick Net Connect (QNC)	▪ 0 – Disables QNC capability; packet originations will use the packet data service option number ▪ 1 – Enables QNC capability; packet originations will use the async data service option number	1h07+	3GPP2
$QCPREV	Protocol revision in use	Returns one of the following codes: ▪ 1 – JSTD008 ▪ 3 – IS_95A ▪ 4 – IS_95B ▪ 6 – IS_2000 ▪ 7 – IS_2000_REL_A ▪ 8 – IS_2000_REL_B ▪ 9 – IS_2000_REL_C ▪ 10 – IS_2000_REL_C_MI ▪ 11 – IS_2000_REL_D; accepts no arguments	1h07+	3GPP2
$QCSO	Set data service option number set; saves to nonvolatile memory	▪ 0 – Pre-707 SO numbers (RS 1: Async 4, G3 Fax 5, packet 7; RS 2: Async 12, G3 Fax 13, packet 15) ▪ 1 – Proprietary SO numbers (RS 1: Async 4, G3 Fax 5, packet 7; RS 2: Async 0x8021, G3 Fax 0x8022, packet 0x8020) ▪ 2 – IS-707 SO numbers (RS 1: Async 0x1004, G3 Fax 0x1005, packet 0x1007; RS 2: Async 12, G3 Fax 13, packet 15)	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCMDR	Set Medium Data Rate (MDR) (also known as HSPD) setting	Valid values are 0 to 3 ▪ 0 – MDR service only. The mobile will originate with SO22 or SO25. The mobile will not negotiate to any other service option if SO22 and SO25 are unavailable. ▪ 1 – MDR service, if available. The mobile will originate with SO22 or SO25, but will negotiate to a low-speed packet service option if MDR is not available. The mobile will not negotiate to SO33. ▪ 2 – LSPD only. The mobile will originate a low-speed packet call only. The mobile will not negotiate to SO22, SO25, or SO 3. ▪ 3 – SO33, if available. The mobile will negotiate to MDR or low-speed packet service options if SO33 is not available.	1h07+	3GPP2
$QCSCRM	Enable/ disable mobile from SCRMing	▪ 0 – Mobile never SCRMs ▪ 1 – Mobile can SCRM as needed ▪ Only applies to SO33 calls; this value is stored in NV, the default is 1	1h07+	3GPP2
$QCTRTL	Enable/ disable R-SCH throttling	▪ 0 – Mobile never throttles R-SCH ▪ 1 – Mobile can throttle R-SCH as needed Only applies to SO33 calls; this value is stored in NV, the default is 1 *For MSM5000, MSM5105, and MSM5100 ASICs only.	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCHDRC	For 1xEV, select a fixed DRC value	▪ 0 – NULL rate ▪ 1 – 38.4 kbps ▪ 2 – 76.8 kbps ▪ 3 – 153.6 kbps ▪ 4 – 307.2 kbps (short) ▪ 5 – 307.2 kbps (long) ▪ 6 – 614.4 kbps (short) ▪ 7 – 614.4 kbps (long) ▪ 8 – 921.6 kbps ▪ 9 – 1228.8 kbps (short) ▪ A – 1228.8 kbps (long) ▪ B – 1843.2 kbps ▪ C – 2457.6 kbps ▪ D – 1536.0 kbps ▪ E – 3072.0 kbps ▪ F – VarRate mode The default value is F for variable rate mode. **Note:** Command only applies to 1xEV connections. **Note:** Test feature only.	1h07+	3GPP2
$QCHDRET	For 1xEV-DO Rev A, configures forced early termination test mode on the reverse link	Valid values are: ▪ 0 – Normal RL packet behavior (termination based on ARQ Ack/Nak) ▪ 1 – Forced RL packet early termination after 1 subpacket ▪ 2 – Forced RL packet early termination after 2 subpackets ▪ 3 – Forced RL packet early termination after 3 subpackets ▪ 4 – Forced RL packet termination after 4 subpackets The default value is 0. Note that this only applies when RTC MAC Subtype 3 protocol is negotiated. **Note:** Test feature only.	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCHDRR	For 1xEV, select a fixed reverse rate	Valid values for default RTC MAC, transmission rate (kbps) are: ▪ 1 – 9.6 ▪ 2 – 19.2 ▪ 3 – 38.4 ▪ 4 – 76.8 ▪ 5 – 153.6 ▪ F – VarRate Mode Valid values for subtype 3 RTC MAC, maximum payload size (bits) are: ▪ 1 – 96 ▪ 2 – 224 ▪ 3 – 480 ▪ 4 – 736 ▪ 5 – 992 ▪ 6 – 1504 ▪ 7 – 2016 ▪ 8 – 3040 ▪ 9 – 4064 ▪ A – 6112 ▪ B – 8160 ▪ C – 12256 ▪ F – VarRate Mode The default value is F for variable rate mode. **Note:** Command only applies to 1xEV connections. **Note:** Test feature only.	1h07+	3GPP2
$QCHDRT	For 1xEV, toggle early termination	▪ 0 – Early termination disabled ▪ 1 – Early termination enabled The default value is 1. **Note:** Command only applies to 1xEV connections. **Note:** Test feature only.	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCMIP[1]	Enable/ disable Mobile IP	▪ 0 – Mobile IP disabled, Simple IP only ▪ 1 – Mobile IP preferred In the initial MIP registration, if the network does not support Mobile IP, the mobile automatically reverts to Simple IP (forces a PPP renegotiation by sending an LCP C-Req). However, if a Mobile IP session is registered, and the mobile enters a network that does not support Mobile IP, it will drop the session and inform the upper layers of the failure, e.g., by dropping DCD to a laptop. ▪ 2 – Mobile IP only The mobile will make data calls only when Mobile IP is supported in the network. During an MIP session, if the mobile hands off to a network that does not support MIP, the mobile will drop the session and inform the upper layers of the failure, e.g., by dropping DCD to a laptop. This value is stored in NV. The default value is 0. **Note:** When the AT$QCMIP value is changed to 1 or 2, this modifies the value of AT+CRM to 2. AT+CRM with a value of 2 enables network model operation. Changing the value to 0 will reset the AT+CRM to its original value. **Note:** This change is not supported by DMSS 5105 Release 1.0 Commercial. **Note:** When the AT$QCMIP value is changed to 1 or 2, this modifies the value of AT$QCMDR to 3. AT$QCMDR=3 means that the mobile tries SO33 when it is in a cdma2000 network that advertises P_REV 6 or higher. When AT$QCMIP >0 and an attempt is made to set AT$QCMDR to less than 3, the mobile will return ERROR. **Note:** When the AT$QCMIP value is set to 1 or 2, this changes the value of AT$QCPKND to 0. This means that the mobile must see a dial string (such as ATDT#777) on the serial interface before it will originate packet data calls. When AT$QCMIP >0 and an attempt is made to set AT$QCPKND to 1, the mobile returns ERROR.	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCMIPEP	Enable/ disable currently active profile	▪ 0 – Disables the currently active profile (profile is unavailable until it is reenabled) ▪ 1 – Enables the currently active profile	1h07+	3GPP2
$QCMIPGETP	Return all information corresponding to the specified profile number	▪ If no profile number is entered, all information corresponding to the currently active profile is returned. ▪ If there is no profile associated with the specified number, an error is returned.	1h07+	3GPP2
$QCMIPMASPI	Set MN-AAA SPIs for the currently active profile	Two arguments – SPI value and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPMASSX	Set MN-AAA shared secret for the currently active profile in HEX	Two arguments – Hexadecimal number and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPMHSPI	Set MN-HA SPIs for the currently active profile	Two arguments – SPI value and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPMHSS	Set MN-HA shared secrets for the currently active profile	Two arguments – String corresponding to the shared secret to be stored and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** Double quotes are only required if the string contains a comma. **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCMIPMHSSX	Set MN-HA shared secret for the currently active profile in HEX	Two arguments – Hexadecimal number and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPNAI	Set the NAI for the currently active profile	Two arguments – String corresponding to NAI to be stored and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** Double quotes are only required if the string contains a comma. **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPP	Select MIP user profile to be active	Takes a profile number between 0 and 5. This value is stored in NV. This AT command is expected to be used by users to configure dialup networking.	1h07+	3GPP2
$QCMIPRT	Set the reverse tunneling currently active profile	First argument ▪ 0 – Do not request reverse tunneling ▪ 1 – Request reverse tunneling Second argument ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPT[1]	Enable/ disable the use of RFC 2002bis authentication	▪ 0 – Use of RFC 2002bis authentication is disabled; RFC 2002 style authentication is used instead ▪ 1 – Use of RFC 2002bis authentication is enabled	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCMIPMASS	Set MN-AAA shared secrets for the currently active profile	Two arguments – String corresponding to the shared secret to be stored and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** Double quotes are only required if the string contains a comma. **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPHA	Set home addr for the currently active profile in string	Two arguments – Home addr string and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPSHA	Set secondary home addr for the currently active profile in string	Two arguments – Secondary home addr string and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCMIPPHA	Set primary home addr for the currently active profile in string	Two arguments – Primary home addr string and one of the following: ▪ 0 – Do not commit to NV ▪ 1 – Commit to NV **Note:** If the value provisioned is not committed to NV, the temporary values will be deleted at the end of the following call or if $QCMIPP is called.	1h07+	3GPP2
$QCVAD	Prearrange-ment setting; respond to Page message that has a voice service option with a page response that has a data service option	▪ 0 – Off ▪ 1 – Fax for next call ▪ 2 – Fax for all calls ▪ 3 – Async for next call ▪ 4 – Async for all calls	1h07+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCNMI	Similar to +CNMI, except that <ds> and <bm> are not supported in CDMA	▪ <mode> = 0,1,2 ▪ <mt> = 0,1,2,3 ▪ <bfr> = 0,1	1h08+	3GPP2
$QCSMP	Similar to +CSMP but with a different parameters ▪ tid – Teleservice ID ▪ vpf – Validity Period Format ▪ vp – Validity Period ▪ ddtf – Deferred Delivery Time Format ▪ ddt – Deferred Delivery Time	= tid,vpf,vp,ddtf,ddt; <tid> = 4095 - 4102 <vpf> = 0 - 1 <vp>=string[22] <ddtf> = 0 - 1 <ddt> = string[22] <vpf> = 0 Absolute, 1 Relative <ddtf> = 0 Absolute, 1 Relative	1h08+	3GPP2
$QCPMS	Similar to +CPMS	= <mem1>,<mem2>,<mem3>; <mem> is any of ME or MT or SM	1h08+	3GPP2
$QCMGR	Similar to +CMGR	= <index>	1h08+	3GPP2
$QCMGS	Similar to +CMGS	= <da> , <toda> followed by SMS string	1h08+	3GPP2
$QCMSS	Similar to +CMSS	= <index>[,<da>[,<toda>]]	1h08+	3GPP2
$QCMGD	Similar to +CMGD	= <index>	1h08+	3GPP2
$QCMGL	Similar to +CMGL	= <stat>	1h08+	3GPP2
$QCMGF	Similar to +CMGF but only Text mode is supported in CDMA	= <mode>	1h08+	3GPP2
$QCMGW	Similar to +CMGW	= <da> , <toda> followed by SMS string	1h08+	3GPP2

Command	Description	Operation	Supported packages	Applicable mode
$QCBCIP	Sets the multicast IP addresses that the laptop wants to listen to for BCMCS service	The parameter for the AT command is a string that has 3 IPv4 addresses. This function checks to ensure that there are 3 IPv4 addresses and they are all either in the multicast range or 0.0.0.0 If this is true, then the BCMCS IP addresses that the laptop wants to listen to are set to these three IP addresses, and they are also written to EFS. If the number of IP addresses in the string is not 3 or at least one of them is not a valid multicast IP address (or 0.0.0.0), *all* the BCMCS IP addresses that the laptop is interested in are left untouched.	1h07+ Deprecated from MPSS JO 1.0+ onwards	3GPP2
$QCBCEN	Enables or disables BCMCS service to the laptop	▪ 0 – Disables BCMCS service to the laptop ▪ 1 – Enables BCMCS service to the laptop	1h07+ Deprecated from MPSS JO 1.0+ onwards	3GPP2
$QCQOSPRI	Sets desired value of QoS nonassured priority adjustment	Range is from 0 to 255	1h07+	3GPP2

[1] This AT command is for test purposes only and should not be changed by the mobile phone user.

[2] Command forwards to apps processor. Refer to Section 3.2.4.3.

[3] +CFUN=1,1 command forward to apps processor.

3 AT Command Processor

This chapter describes the AT Command Processor (ATCoP) implementation in AMSS software.

It is intended for software engineers who want to add or modify AT commands. It assumes that you are familiar with the AT command processing requirements in [S1], [S2], [S3], and [S6].

3.1 ATCoP and related source files

The ATCoP implementation consists of the source files listed in Table 3-1.

Table 3-1 ATCoP source files

Filename	Description
dsat_v.h	Definitions, functions, and data structures used by modules external to ATCoP
dsati.h	Definitions, functions, and data structures used internal to ATCoP
dsatclient.c	ATCoP client asynchronous interface
dsatclient_ex.c	ATCoP client asynchronous interface
dsatclient.h	External definitions, functions, and data structures required for the ATCoP client asynchronous interface
dsatclienti.h	Internal definitions, functions, and data structures required for the ATCoP client asynchronous interface
dsatdiagif.c	Deprecated ATCoP– Diag interface; file is not present from MPSS NI 2.0+
dsatdiagif.h	Deprecated ATCoP – Diag interface; file is not present from MPSS NI 2.0+
dsatprep.c	Preprocesses data received from serial driver
dsatprep_ex.c	Preprocesses data received from serial driver
dsatpar.c	AT command parser that parses AT command line into token structures
dsatcmdp.c	ATCoP that looks up command contained in token structure and calls appropriate command processing function from command table to process command
dsatcmdp_ex.c	ATCoP that looks up command contained in token structure and calls appropriate command processing function from command table to process command
dsatrsp.c	AT command response generating and formatting utilities
dsatrsp_ex.c	AT command response generating and formatting utilities
dsatutil.c	General ATCoP utilities
dsatutil_ex.c	General ATCoP utilities
dsatparm.c	Common AT parameter type command processing
dsatparm.h	Common AT parameter type command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatact.c	Common AT action type command processing
dsatact_ex.c	Common AT action type command processing
dsatact.h	Common AT action type command processing definitions, functions, and data structures used by modules internal to ATCoP

Filename	Description
dsatvend.c	Common AT vendor-specific type command processing
dsatvend.h	Common AT vendor-specific type command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatctab.c	Common AT command tables
dsatctab_ex.c	Common AT command tables
dsatctab.h	Common AT command table definitions, functions, and data structures used by modules internal to ATCoP
dsatcmif.c	Common call manager interface
dsatcmif_ex.c	Common call manager interface
dsatcmif.h	Common call manager interface definitions, functions, and data structures used by modules internal to ATCoP
dsatvoice.c	Common voice call processing control
dsatvoice_ex.c	Common voice call processing control
dsatvoice.h	Common voice call processing control definitions, functions, and data structures used by modules internal to ATCoP
dsatsms.c	Common short message service command processing
dsatsms_ex.c	Common short message service command processing
dsatsmsu.c	Common short message service command processing utilities
dsatsmsu_ex.c	Common short message service command processing utilities
dsatsms.h	Common short message service command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatsmsi.h	Common short message service command processing definitions, functions, and data structures used by modules internal to ATCoP short message service unit
dsatme.c	Common mobile equipment command processing
dsatme_ex.c	Common mobile equipment command processing
dsatme.h	Common mobile equipment command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatetsicall.c	ETSI call control command processing
dsatetsicall_ex.c	ETSI call control command processing
dsatetsicall.h	ETSI call control command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatetsicmif.c	ETSI commands call manager interface
dsatetsicmif_ex.c	ETSI commands call manager interface
dsatetsicmif.h	ETSI commands call manager interface definitions, functions, and data structures used by modules internal to ATCoP
dsatetsipkt.c	ETSI packet data command processing
dsatetsipkt_ex.c	ETSI packet data command processing
dsatetsipkt.h	ETSI packet data command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatetsime.c	ETSI mobile equipment command processing
dsatetsime_ex.c	ETSI mobile equipment command processing
dsatetsime.h	ETSI mobile equipment command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatetsismsc.c	ETSI short message service command processing
dsatetsismsa.c	ETSI short message service asynchronous event processing

Filename	Description
dsatetsismsa_ex.c	ETSI short message service asynchronous event processing
dsatetsismsu.c	ETSI short message service command processing utilities
dsatetsisms.h	ETSI short message service command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatetsismsi.h	ETSI short message service command processing definitions, functions, and data structures used by modules internal to ATCoP short message service unit
dsatetsictab.c	ETSI AT command tables
dsatetsictab_ex.c	ETSI AT command tables
dsatetsictab.h	ETSI AT command table definitions, functions, and data structures used by modules internal to ATCoP
dsatgsmfax.c	GSM fax command processing
dsatgsmfax_ex.c	GSM fax command processing
dsatgsmfax.h	GSM fax command processing definitions, functions, and data structures used by modules internal to ATCoP
dsatapi.h	ATCoP functions that are exported for multiprocessor builds; applicable only for multiprocessor targets
dsatmmtgt.c	Top-level AT command table, arrays of command table pointers; target-specific command tables and asynchronous event handler table; defines AT command set supported by the target.
dsatmmtgt_ex.c	Top-level AT command table, arrays of command table pointers; target-specific command tables and asynchronous event handler table; defines AT command set supported by the target.
dsat707ext.c	IS-707 specific extended commands
dsat707ext_ex.c	IS-707 specific extended commands
dsat707ext.h	Contains the definitions of data structures, defined and enumerated constants and function prototypes required for IS-707 mode
dsat707extctab.c	IS-707 command table
dsat707extctab_ex.c	IS-707 command table
dsat707extctab.h	IS-707 AT command table definitions, functions, and data structures used by modules internal to ATCoP
dsat707fax.c	IS-707 specific fax-related AT commands. File is not present from MPSS TH 1.0+ onwards.
dsat707fax.h	Contains the definitions of data structures, defined and enumerated constants and function prototypes required for IS-707 mode fax commands. File is not present from MPSS TH 1.0+ onwards.
dsat707faxctab.c	FAX-related AT modem command table. File is not present from MPSS TH 1.0+ onwards.
dsat707faxctab_ex.c	FAX-related AT modem command table. File is not present from MPSS TH 1.0+ onwards.
dsat707faxctab.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for FAX-related commands specific to IS-707 mode. File is not present from MPSS TH 1.0+ onwards.
dsat707hdr.c	IS-707 specific HDR-related AT commands.
dsat707hdr.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for IS-707 mode HDR commands
dsat707hdrctab.c	IS-707 HDR command table

Filename	Description
dsat707hdrctab_ex.c	IS-707 HDR command table
dsat707hdrctab.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for HDR related command table
dsat707mip.c	IS-707 specific Mobile IP-related AT commands
dsat707mip.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for IS-707 mode Mobile IP-related commands
dsat707mipctab.c	IS-707 MIP-related command table
dsat707mipctab_ex.c	IS-707 MIP-related command table
dsat707mipctab.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for MIP-related command table
dsat707pstats.c	IS-707 specific Protocol Statistics AT commands
dsat707pstats.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for IS-707 mode Protocol Statistics-related commands
dsat707pstatsctab.c	IS-707 specific Protocol Statistics AT command table
dsat707pstatsctab_ex.c	IS-707 specific Protocol Statistics AT command table
dsat707pstatsctab.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for Protocol Statistics related command table
dsat707sms.h	Contains the definitions of data structures, defined and enumerated constants and function prototypes required for the executing the SMS commands internally.
dsat707smsa.c	IS-707 short message service asynchronous event processing
dsat707smsa_ex.c	IS-707 short message service asynchronous event processing
dsat707smsc.c	IS-707 short message service command processing
dsat707smsctab.c	IS-707 SMS command table
dsat707smsctab_ex.c	IS-707 SMS command table
dsat707smsctab.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for IS-707 SMS-related command table
dsat707smsu.c	IS-707 short message service command processing utilities
dsat707smsu_ex.c	IS-707 short message service command processing utilities
dsat707util.c	Contains generic function definitions used by the IS707 AT commands
dsat707util.h	Contains the definitions of data structures, defined and enumerated constants and function prototypes required for the executing the utility functions
dsat707vend.c	IS 707 vendor-specific commands
dsat707vend.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for the executing the IS 707 Vendor commands
dsat707vendctab.c	IS 707 vendor-specific command table
dsat707vendctab_ex.c	IS 707 vendor-specific command table
dsat707vendctab.h	Contains the definitions of data structures, defined and enumerated constants, and function prototypes required for the executing the IS 707 Vendor tables
dsatdl.c	Contains functions that are used for dynamic loading of AT commands.
dsatdl_ex.c	Contains functions that are used for dynamic loading of AT commands.

Filename	Description
dsatdl.h	Contains the definitions of data structures, defines and enumerated constants, and function prototypes required for the data services AT command processor dynamic loading.

Data services task source files containing top-level task dispatcher functionality are listed in Table 3-2. These files require modification when adding asynchronous command event handling to ATCoP.

Table 3-2 Data services task source files

Filename	Description
dstask.h	Definitions, functions, and data structures used by modules external and internal to data services task
dstaski.h	Definitions, functions, and data structures used by modules internal to data services task
dstask_v.h	Definitions, functions, and data structures used by modules external to data services task
dstask.c	Data services task and top-level dispatcher for data services task

Only a subset of the above files will need to be modified based on the type of command being added.

3.2 ATCoP overview

The ATCoP implements the parsing and processing of modem AT commands for packet-switched data support, circuit-switched data support, SMS, and GSM FAX support as per the requirements specified in [S1], [S2], [S3], and [S6].

3.2.1 General architecture and control flow

The ATCoP implementation consists of a processing engine containing lists of the supported AT commands. The processing engine parses the AT commands and executes commands from command tables.

The ATCoP processing engine processes AT commands received from the serial port (the R_m interface) in the AT Command state. Figure 3-1 depicts the basic architecture and data flow of ATCoP.

During the initialization, the shim layer between SIO driver and data services (SIOLIB) registers with the Runtime Device Mapper (RDM) for all of the port opening and closing functionalities.

Whenever a device is connected (or disconnected), the RDM invokes the SIOLIB call back function to notify the opening (or closing) of the corresponding port. SIOLIB then raises a respective signal to itself (DS task). Figure 3-1 illustrates the basic ATCoP architecture and data flow, and indicates a USB being plugged in.

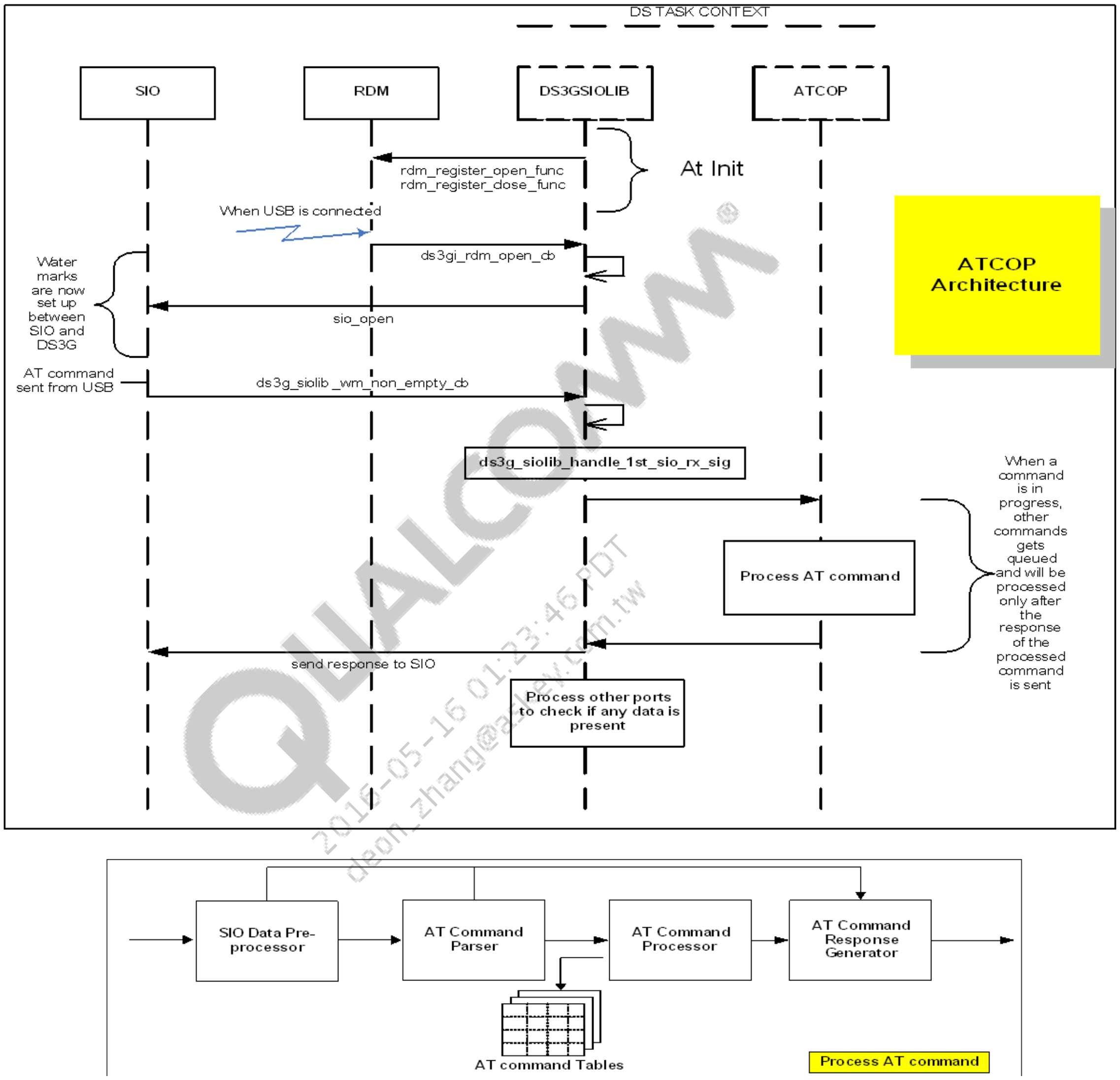
DS TASK CONTEXT
SIO
RDM
DS3GSIOLIB
ATCOP
rdm_register_open_func
rdm_register_dose_func
At Init
When USB is connected
ds3gi_rdm_open_cb
Water marks are now set up between SIO and DS3G
sio_open
AT command sent from USB
ds3g_siolib _wm_non_empty_cb
ds3g_siolib_handle_1st_sio_rx_sig
ATCOP Architecture
Process AT command
When a command is in progress, other commands gets queued and will be processed only after the response of the processed command is sent
send response to SIO
Process other ports to check if any data is present
SIO Data Pre-processor
AT Command Parser
AT Command Processor
AT Command Response Generator
AT command Tables
Process AT command

Figure 3-1 Basic ATCoP architecture and data flow

In the signal handler, the SIOLIB registers watermarks and various callbacks for events, e.g., non-empty watermark, low watermark, high watermark, with SIO.

Whenever data is queued into the SIO driver from the USB, it will invoke appropriate callbacks, e.g., non_empty watermark, etc., to raise a signal to the SIOLIB. If the USB is selected as a modem port (modem ports are selected through UI), then data from USB will be routed to ATCoP via SIOLIB.

In the signal handler, the ATCoP extracts the data and processes the AT commands and returns the result to SIO via SIOLIB. This job of extraction and processing by the ATCoP is spread across several modules. These modules include:

- SIO data preprocessor

- AT command parser
- ATCoP
- AT command response generator

AT command data received by the serial drivers from the DTE is first processed by the SIO data preprocessor. A NULL-terminated command line is produced and passed to the AT command parser.

The NULL-terminated command line is parsed by the AT command parser. A token structure is produced by the parser for each command parsed and is queued for processing by the ATCoP. The parser places a token structure on the queue for each command contained within the command line before the ATCoP is invoked.

AT commands are defined in command tables, with each command table entry containing a pointer to an associated command execution function.

The ATCoP performs a table lookup on each command token structure removed from the token queue. If a match is found, the associated command execution function is called to process the command.

The AT command response generator provides utilities that format command responses, generate result codes, and send responses to the DTE.

The ATCoP processes an AT command line one command at a time. If any command within the command line produces an error, command responses are generated for any commands processed before the error. An error result code is then generated to terminate the command line response and the remaining commands in the command line are dropped without further processing.

Only after the completion of execution of the current AT command will it service other modem ports if any data is present.

Figure 3-2 illustrates the control flow for AT commands processing.

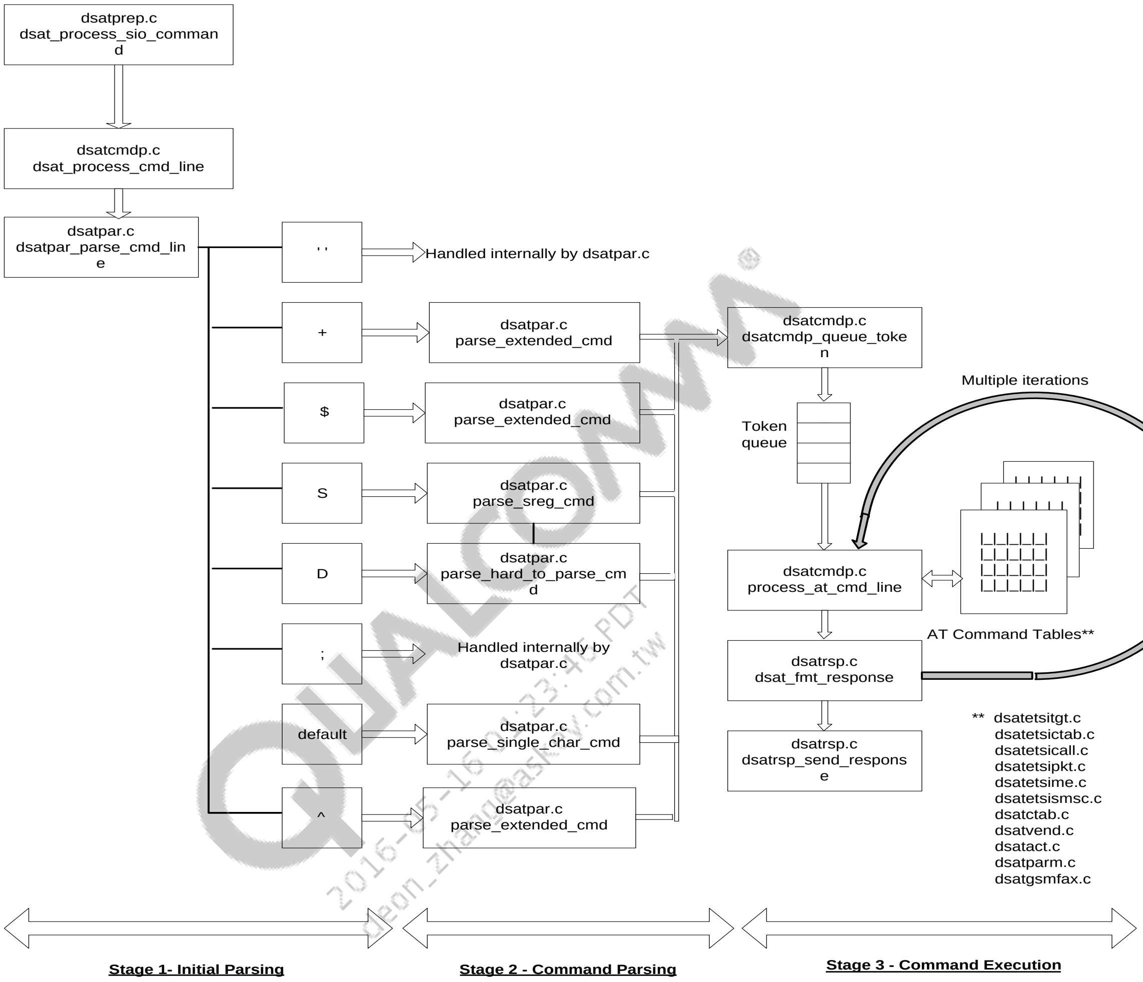

Figure 3-2 Control flow for AT commands processing

Stage 1 – Initial Parsing

The leading "AT" is stripped out of the command line by function dsat_process_sio_command. A NULL- terminated command line is then passed to the function dsat_process_cmd_line for subsequent parsing and processing. Parsing of the command line is done by function dsatpar_parse_cmd_line. The leading character of each AT command is examined and the appropriate parsing functions are called based on the category of AT command. Each AT command name (including the leading symbol, for example, $QCDMG) is parsed from the command line, as well as all associated command parameters, and placed within a token data structure. One token structure is produced for each command within the command line and queued up for command processing, once command line parsing is completed.

The code corresponding to this stage, namely dsat_process_sio_command, dsat_process_cmd_line, and dsatpar_parse_cmd_line, should not be modified by software engineers who intend to add or modify AT commands.

Stage 2 – Command Parsing

A parse function, depending on the leading character of the command, parses the AT command, filling a token data structure with the parsed information, and returns a result. If the result is OK (which means arguments, the number of arguments, and special processing code, etc., are stored), the token data structure is placed on a queue for later processing by the AT command processor. A token structure is placed on the queue for each command contained within the command line.

For example, the parse function parse_extended_cmd is invoked for extended or proprietary AT commands.

Stage 3 – Command Execution

A token structure for each command within the command line is placed on the queue by function dsatcmdp_queue_token. After parsing completes, the function process_at_cmd_line is called to process each token structure on the queue. Each command token structure is removed from the queue and the command is searched for within the command tables. The top level command table within file dsatetsitgt.c, which points to command tables within files dsatctab.c and dsatetsictab.c, defines the AT command set supported. If the command is found in the table, the processing function contained within the table for that particular command is called to execute the command. The response returned by the command, if any, is formatted with function dsat_fmt_response. This occurs for each token within the token queue. Command responses are sent to the DTE by calling function dsatrsp_send_response.

AT command tables and the associated AT command execution functions should be the focus for engineers adding or modifying new AT commands. New AT commands must be added to the command tables. Typically a new command execution function will need to be written for each new command added.

3.2.2 SIO data preprocessor

The SIO data preprocessor processes data received from the TE, producing a NULL-terminated AT command line as output to the parser. This is the normal mode of operation for the SIO data preprocessor and is referred to as the AT Command mode. The SIO data preprocessor also operates in two other modes: SMS Message Entry mode and Abort Command mode.

SMS Message Entry mode is entered only after an SMS send or write message AT command is received. SMS Message Entry mode allows actual SMS message entry by the TE. Data received in this mode is processed as an SMS message.

Abort Command mode is entered only after an abortable AT command is received, and in the case of an SMS command, only after the message has been received from the TE. While operating in Abort Command mode, any character received from the TE initiates an abort of the AT command currently executing.

3.2.3 AT command parser

The AT command parser takes a NULL-terminated AT command line, with the AT removed, as input from the SIO data preprocessor. It then parses each command and its associated arguments contained within the command line. Parsing occurs as shown and as described in Stage 2 of Figure 3-2. A token data structure is produced for each command parsed and queued for the ATCoP. Refer to Section 3.2.7.5 for details of the token data structure.

3.2.4 AT command processor

Token data structures for each command parsed in the command line are then input to the ATCoP. Once the entire command line is parsed and token structures queued, the ATCoP reads each token structure from the queue and processes it. All token structures contained in the queue are treated as part of a single command line. Any command generating an error results in all remaining commands in the queue being dropped.

The ATCoP is table-driven. Based on the ME operating mode and information contained within the token data structure, a linear search is performed to find the command in one of the command tables. When a match is found, the command processing function is called to process the command. It returns the response generated by the command, if any, and the command result code.

When the ME is operating in the Limited Service state due to a missing SIM or pending a SIM PIN validation, i.e., emergency Calls Only mode, the command processor will accept only those commands with the RESTRICTED attribute set in the command table. AT commands that can be processed while in the Limited Service state in this AMSS software implementation are identified in Chapter 2 .

3.2.4.1 Aborting AT commands

Some AT commands may be aborted after the command is issued but prior to completion. AT commands that can be aborted in this AMSS software implementation are identified in Chapter 2

The ATCoP determines which commands are abortable by use of the command tables. The AT command tables contain a pointer to an abort function for each command entry. Refer to Section 3.2.6 for details on command table structure. Abortable commands have this pointer set to the abort command handler for that particular command. Commands that are not abortable have this pointer set to NULL.

The ATCoP looks at the value of the abort function pointer to determine if the command can be aborted. If so, the SIO Data Preprocessor mode is set to Abort Command mode. Any character received before the command completion calls the command abort handler function to abort the command.

3.2.4.2 Asynchronous event handling

In order to process certain commands, such as some SMS commands, asynchronous events must be handled within the ATCoP. Asynchronous events are events received by the data services task from other tasks or possibly the data services task itself. Commands requiring asynchronous events to complete processing will be referred to as asynchronous commands from this point forward.

The ATCoP asynchronous event handler receives events from the data services task dispatcher and processes them. The event handler contains a lookup table. Refer to Section 3.2.4.2 for details on the structure of the asynchronous event handler table. Each table entry contains a function pointer to an event handler function. Event handler functions are called when ATCoP asynchronous events are received based on operating mode and the type of event received.

The control flow for AT command processing described in Figure 3-2 is slightly different for asynchronous commands. In Stage 3, once the processing function within the command table is called to begin execution of the command, it returns a value of DSAT_ASYNC_CMD. This instructs the ATCoP that this command requires asynchronous events to complete processing and to stop further processing of the tokens queued for the current command line until the asynchronous events are received to complete processing of the asynchronous AT command. When the ATCoP is waiting for any asynchronous events, it drops any new commands that are received from the SIO preprocessor. The function process_at_cmd_line is then exited at this point.

When ATCoP asynchronous events are received by the data services task dispatcher, the function dsat_process_async_cmd is called to process the event. This results in the appropriate event handler function being called from the asynchronous event handler table. When a handler function returns a value of something other than DSAT_ASYNC_CMD or DSAT_ASYNC_EVENT, this indicates that the current AT command processing has completed, and processing of the remaining commands in the command line should resume. The function process_at_cmd_line is invoked at this point to complete processing of the remaining tokens in the queue, as well as format and send the command responses. Refer to Section 3.3.2.2 for further information on the asynchronous event handler function return values.

3.2.4.3 AT command response generator

The AT command response generator provides the following services:

- Generates a formatted result code response based on an input result code
- Formats a response from a response generated by a specific command processing function or asynchronous event handler
- Sends a formatted response to destination based on operating mode and AT Command mode

Formatting of responses is based on DCE-to-TE interface AT parameters. These parameters determine how a response is formatted or if a response is sent at all. Various AT parameters also define different categories of intermediate result codes and error responses. The response generator module provides services to generate all of these various result codes and error responses.

3.2.5 AT command forwarding

AT command forwarding provides a mechanism by which registered AT commands received from the Terminal Equipment (TE) can be forwarded from the modem to a control point on the apps processor where they need to be processed. The responses for these AT commands can then be forwarded from the apps processor to the modem to be sent back to the TE.

This provides flexibility for the apps processor control point to define a command behavior different than what is supported by the modem. QMI_AT (see [Q2]) service is used by ATCoP for command forwarding mechanism, e.g., +CFUN=1,1, +CKPD,+CSS commands are forwarded to the apps processor where they need to be processed.

3.2.6 ATCoP command tables

This section contains a description of the ATCoP command tables, including the structure of a command table, and the different types of command tables and their functions. For a complete list of the AT commands supported in AMSS, refer to Chapter 2.

3.2.6.1 Command table structure

The ATCoP is table-driven. Based on information contained within the token data structure generated by the parser, a linear search is performed to find the command in one of the command tables. When a match is found, the command execution function is called to process the command. It then returns the response generated by the command, if any, as well as the command result code. The structure of the command-processing table is illustrated in Figure 3-3.

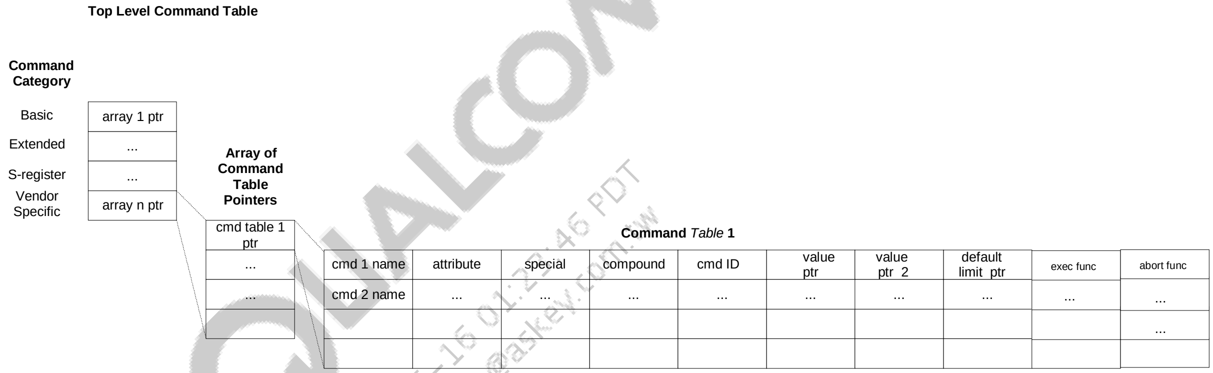

Figure 3-3 AT command table structure

The table is structured such that each row represents a particular category of commands, i.e., basic, extended, S-register, or vendor-specific. The columns of the table are independent of the operating mode. This provides support for multimode operation.

Each location in the table contains a pointer to an array of AT command table pointers for a particular category of command. The last element of each array contains a NULL pointer to define the end of the array. AT command tables contain command names, arguments, attributes, etc., and command processing and abort command handler function pointers. The function pointers identify the function that the command processor must call to execute the specific command and the function to call to abort the command, if the command is abortable.

Token structures produced by the parser for each command contain information that identifies the category of AT commands. Structuring the rows of the table to correspond to command categories reduces the number of commands that must be searched during command lookup. Having an array of command table pointers for each command category allows command tables to be structured so they can be shared across modes with less repetition of tables. Command tables containing particular groups of commands can be set up to be shared across multiple modes.

3.2.6.2 AT command tables and command processing functionality

AT command tables are used by the ATCoP to process each command. The tables contain entries for each AT command, defining how each command is processed and what processing function must be called to process each command. Refer to Section 3.1 for information on the AT command tables.

Command processing functionality is generally distributed across multiple files. Refer to Section 3.2.6.1 for information on where the processing functionality for a particular command is defined.

3.2.7 Command table entries

The ATCoP command tables contain entries of type dsati_cmd_type (and dsati_cmd_ex_type from MPSS BO 2.0+ onwards). The dsati_cmd_type (and dsati_cmd_ex_type from MPSS BO 2.0+ onwards) data structure(s) and the associated dflm_type, def_list_type, mixed_def_s_type, and token_struct_type data structures are described in detail in this section. These and other data structures used internally by the ATCoP command processor are defined in dsati.h.

3.2.7.1 dsati_cmd_type (and dsati_cmd_ex_type from MPSS BO 2.0+ onwards)

Before MPSS DI 2.0+

dsati_cmd_type is the type for a command table entry. This is defined as follows:

```
typedef struct dsati_cmd_struct
{
  byte          name[20];
  uint32        attrib;
  byte          special;
  byte          compound;
  uint32        cmd_id;
  const void   *val_ptr;
  const void   *val_ptr2;
  const void   *def_lim_ptr;
  dsat_result_enum_type (*proc_func)( dsat_mode_enum_type,
                                      const struct dsati_cmd_struct*,
                                      const tokens_struct_type*,
                                      dsm_item_type* );
  boolean (*abort_func)( const struct dsati_cmd_struct* );
} dsati_cmd_type;
```

MPSS DI 2.0+ onwards and before MPSS BO 2.0+

dsati_cmd_type is the type for a command table entry. This is defined as follows:

```
typedef struct dsati_cmd_struct
{
  byte          name[20];
  uint32        attrib;
  byte          special;
  byte          compound;
  uint32        cmd_id;
  const void   *def_lim_ptr;
  dsat_result_enum_type (*proc_func)( dsat_mode_enum_type,
                                      const struct dsati_cmd_struct*,
                                      const tokens_struct_type*,
                                      dsm_item_type* );
  boolean (*abort_func)( const struct dsati_cmd_struct* );
} dsati_cmd_type;
```

dsatutil_get_val_from_cmd_id() function is introduced to replace val_ptr and val_ptr2 members present before MPSS DI 2.0+

MPSS BO 2.0+ onwards

dsati_cmd_type and dsati_cmd_ex_type are the types for a command table entry. These are defined as follows:

```
typedef struct dsati_cmd_struct
{
  char *name;
  uint32 attrib;
  byte special;
  byte compound;
  uint16 cmd_id;
  const void *def_lim_ptr;
} dsati_cmd_type;

typedef struct dsati_cmd_ex_struct
{
  uint16 cmd_id;
  dsat_result_enum_type (*proc_func)( dsat_mode_enum_type,
                                      const struct dsati_cmd_struct*,
                                      const tokens_struct_type*,
                                      dsm_item_type* );
} dsati_cmd_ex_type;
```

The fields in the dsati_cmd_type (and dsati_cmd_ex_type from MPSS BO 2.0+ onwards) data structure(s) are:

- name
  - Name of the AT command, including the preceding +, $, ^, or &, and the terminating NULL
  - Examples – +IPR, $QCDMG, S6, &C, Z

- attrib
  - 32-bit mask that specifies one or more command attributes

Table 3-3 lists various command attributes and their meanings, and example AT commands, if any, that have each attribute.

Table 3-3 AT command attributes

Attribute	Description	Examples
ATTRIB_NONE	AT command has no attributes	&V
CONFIG	AT command has one or more arguments	S3
FCLASS2P0	Fax Class 2.0 command; the ERROR result code will be returned unless +FCLASS=2.0; not used for ETSI commands	
LOCAL_TEST	Test command is executed locally; this attribute should be set for commands that are classified as local in IS-707-A.3; for future support.	
ALWAYS	Parameter values are always transmitted to the base station in the configuration string; this is used for AT commands whose default values are manufacturer-specific; for future support	
SIMPLE_DEC	Format for the parameters in the configuration string is a 3-digit decimal with leading zeros	&C, E
STRING	AT command arguments are strings, in quotes	+CSCA, +CSMP
HEX	AT command arguments are numeric values in hexadecimal format (without any leading 0x or trailing h)	
EXTENDED	AT command uses extended syntax, i.e., the command begins with + or $); the extended command syntax is specified in ITU-T V.25ter	+IPR, $QCDMG
READ_ONLY	AT command does not have any arguments, but returns a value	+GMI, +GSN
DOWN_LOAD	Parameter values (if different from the default) are transmitted to the base station in the configuration string; for future support	S3
LIST	AT command has arguments that are numeric, alphanumeric, or strings without quotes	+IPR
MIXED_PARAM	AT command arguments could be a string of some allowed values, a numerical value in certain ranges, or a string of limited length. Each argument is dealt with individually, based on type. See the description of mixed_def_s_type in Section 3.2.7.4 for details.	+CSCA, +CSMP
NO_QUOTE	No quotes are used for display of string arguments in response to read command	+CSCA, $QCDNSP
YES_QUOTE	Opposite of no_quote; the string argument is quoted for display. Both YES_QUOTE and NO_QUOTE are needed since there are cases when the argument does not care about quotes, i.e., the attribute is neither YES_QUOTE nor NO_QUOTE. In this case, the value is processed as a literal.	+CPBS, +CSCA
RESTRICTED	AT command is processed when ME is operating in Limited Service state; without attribute, command is rejected by the command processor	+CGMI, +CPAS
NO_DISPLAY	Parameter is not displayed in response to read command	$QCPDPP
BROKEN_ RANGE	AT command has numeric argument that has discontinuous numeric range	+CGQREQ

Attribute	Description	Examples
DUAL_STACK	AT command has value pointers for both STACKs; applicable for DUAL SIM DUAL STANBY (DSDS) targets. It's deprecated from MPSS JO 1.0+ onwards.	
UMTS_CMD	AT command will give a desired result in UMTS RAT only.	+COPS
CDMA_CMD	AT command will give a desired result in CDMA RAT only.	+CDV
DO_PREF_CMD	AT command is a DO preferred command.	$QCNMI
COMMON_CMD	AT command is independent from the RAT.	+CLCC
NO_RESET	Reset commands (Z and F) do not change the parameter value	+CGSMS
ABORT_CMD	AT command is abortable.	+COPS
MULTI_INDEX	AT command has multi-dimensioned parameter set.	+CGACT
MULTI_SUBS	AT command is Subscription/Session based. Added from MPSS JO 1.0+ onwards.	+COPS
MULTI_SLOT	AT command is Slot based. Added from MPSS JO 1.0+ onwards.	+CSIM

- special
  - The special processing code, if any, or SPECIAL_NONE. The special processing codes are defined in dsati.h. This field should always be set to SPECIAL_NONE. It is used only for backwards compatibility with existing software.
- compound
  - Number of arguments for numeric commands or the maximum string length (excluding the terminating NULL) for string commands
  - Examples are:
    - For S0, which takes a single numeric argument, compound = 1
    - For a command with string argument declared having a size of 30, compound = 29
    - –
- cmd_id
  - Each command is associated with a unique command identification (cmd_id).
  - Value pointers (val_ptr) of each command can be accessed by cmd_id.
- val_ptr and val_ptr2 (replaced with dsatutil_get_val_from_cmd_id() function from MPSS DI 2.0+ onwards)
  - Val_ptr is the pointer to the variable that stores the argument value for the first stack.
  - Val_ptr2 is the pointer to the variable that stores the argument value for the second stack (DSDS targets only).
  - Val_ptr and val_ptr2 follow the same set of rules, the only difference is val_ptr represents the first stack and val_ptr2 represents the second stack.
  - If the variable is an array (for commands that have more than one argument), this is a pointer to the first element in the array.
  - For commands with the LIST attribute, the values stored in *val_ptr are not the actual parameter values; instead, they are indices into the array pointed to by list_v (see Section 3.2.7.3 for details).

    - For commands with the MIXED_PARAM attribute, the pointer points to the array of dsat_mixed_param_val_type. dsat_mixed_param_val_type is a union type of string or numerical value. Whether a member of the array is a string or a numerical value depends on the attribute the corresponding member of the array def_lim_ptr points to (see Section 3.2.7.4 for details).
    - This is a pointer to a variable of one of the following types:
        - dsat_num_item_type for commands with numeric arguments, or commands with the LIST attribute
        - dsat_string_item_type for commands with string arguments
        - dsat_mixed_param_val_type for commands with mixed type of arguments; the arguments could be a string of some allowed values, a numerical value in certain ranges, or a string of limited length (see Section 3.2.7.4 for details)
        - NULL for commands with no arguments
- def_lim_ptr
    - Defines the default value and the allowed range for the arguments
    - This is a pointer to a variable of one of the following types:
        - dflm_type (see Section 3.2.7.2 for details) for commands with the CONFIG attribute
        - def_list_type (see Section 3.2.7.3 for details) for commands with the LIST attribute
        - mixed_def_s_type (see Section 3.2.7.4 for details) for commands with the MIXED_PARAM attribute
        - NULL for commands with the STRING attribute or for commands with no arguments
- proc_func
    - Function pointer to function that is called to execute the command defined by this command table entry
    - The function pointer arguments are:
        - dsat_mode_enum_type defines the current AT Command mode. Table 3-4 lists the modes and their meanings.

Table 3-4 Mode argument

Mode	Description
DSAT_CMD	AT Command mode of operation
DSAT_ONLINE_DATA	AT Online Data mode of operation
DSAT_ONLINE_CMD	AT Online Command mode of operation

        - dsati_cmd_struct* is a pointer to the dsati_cmd_struct structure entry for the particular command that contains this command table entry.
        - token_struct_type* is a pointer to a parser token structure defined in Section 3.2.7.5. The token structure contains information required to process the command.

  - dsm_item_type* is a pointer to the DSM buffer to store the command response in. DSM buffers can be chained if command response exceeds the size of the DSM buffer passed in.
- ❐ The function return type should be one of the values defined in Table 3-5.

Table 3-5 Execution function return values

Value	Description
DSAT_OK	OK result indicating command executed successfully
DSAT_ERROR	ERROR result indicating command execution failed
DSAT_NO_RSP	Result to indicate no command response should be sent to DTE by ATCoP; used for commands that do not generate an immediate response, i.e., the D command
DSAT_CMD_ERR_RSP	Sends command-specific error response placed in DSM buffer passed into this function to DTE only, and sends no result code to DTE as final response to this command
DSAT_ASYNC_CMD	Indicates the command being processed is an asynchronous command, and asynchronous command events will be received later by the Data Services' task dispatcher. These asynchronous events will indicate when the command processing has completed and the remaining commands in the command line can be processed.

- ■ abort_func
  - ❐ Function pointer to function called to abort the command defined by this command table entry
  - ❐ Function pointer value not equal to NULL indicates the command defined by this command table entry is abortable
  - ❐ Function pointer arguments – dsati_cmd_struct* is a pointer to the dsati_cmd_struct structure entry for the particular command that contains this command table entry
  - ❐ Function return type is a Boolean indicating data call should be aborted by DSMGR if value is TRUE; otherwise, no action should be taken

Figure 3-4 illustrates the relationship between dsati_cmd_type and other command data structures.

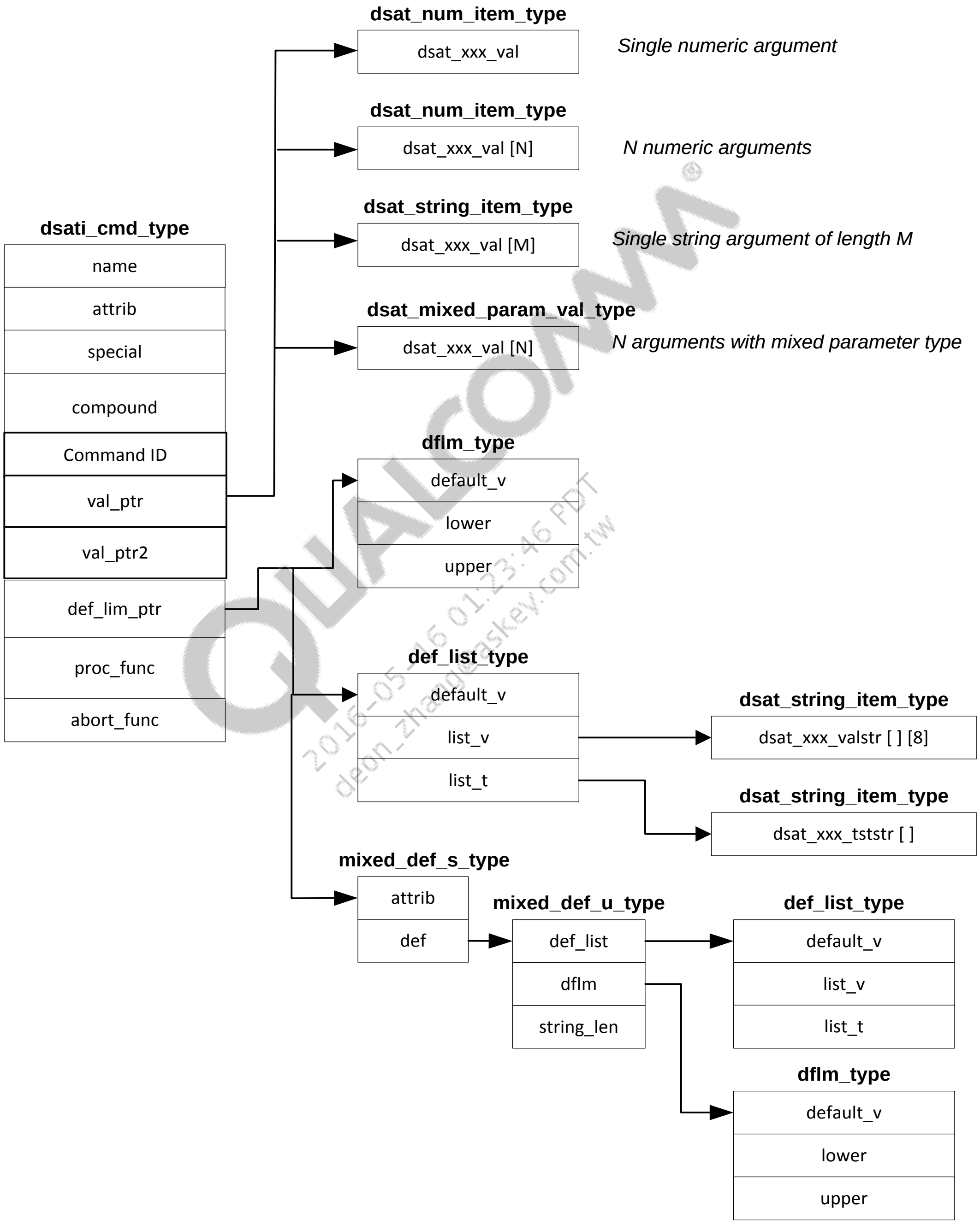

Figure 3-4 Relationship between command data structures

3.2.7.2 dflm_type

The dflm_type is used to define the default value and the lower and upper bounds of the arguments for AT commands with numeric arguments. The dflm_type data structure is defined as follows:

```
typedef struct
{
  dsat_num_item_type        default_v;
  const dsat_num_item_type  lower;
  const dsat_num_item_type  upper;
} dflm_type;
```

- The AT command arguments must be numeric, and the allowed set of values for each argument must be a range of consecutive numbers, i.e., if an AT command has an argument with the allowed values 0, 1, and 255, then the dflm_type cannot be used. The def_list_type should be used, and the command should be declared as type LIST in the command table.
- default_v is the default value of the argument, and lower and upper are the lower and upper limits of the allowed range of values that the argument can assume.
- For AT commands that have more than one argument, each argument has an associated dflm_type data structure.

Examples

The S7 register stores the maximum number of seconds to wait in order to establish an end-to-end data connection. The default value is 50 sec, and the allowed range is 1 to 255 sec. The dflm_type data structure for the S7 register is defined as:

```
LOCAL const dflm_type dsat_s7_dflm [] =
{
  { 50 , 1  , 255  },
} ;
```

The +IFC command, which controls local flow control between the DTE and the ME, has the following syntax:

```
AT+IFC = <m>, <n>
```

where:

- m = 0 to 3, default is 2
- n = 0 to 2, default is 2

The dflm_type data structure for +IFC is defined as follows:

```
const dflm_type dsat_ifc_dflm [] =
{
  { 2  , 0  , 3    },
  { 2  , 0  , 2    },
} ;
```

3.2.7.3 def_list_type

The def_list_type is used to define the default value and the allowed range of values of the arguments for AT commands of type LIST. The def_list_type data structure is defined as:

```
typedef struct
{
  dsat_num_item_type           default_v;
  const dsat_string_item_type  (*list_v)[8];
  const dsat_string_item_type  *list_t;
} def_list_type;
```

- The AT command arguments may be numeric, alphanumeric, or strings. For numeric arguments, the allowed range of values need not be consecutive.
- default_v represents the default value of the argument; it is an index into the array pointed to by list_v.
- list_v is a pointer to an array of 8-byte strings that represent all the valid values that the argument can assume. The last element in the array must be a NULL string to terminate the argument list. The AT command processor performs a string match with the elements in this array in order to determine whether an argument’s value is within the valid range.
- list_t is a pointer to a string that is returned verbatim in response to a test command.
- For AT commands that have more than one argument, each argument has an associated def_list_type structure.

Example

The +IPR command, which specifies the data rate on the R_m interface, has the following data structures for defining the default value and valid range:

```
LOCAL const dsat_string_item_type dsat_ipr_valstr [][8] =
{ "0", "45", "50", "75", "110", "150", "300", "600", "1200",
  "2400", "4800", "9600", "19200", "38400", "57600", "115200", "" };
LOCAL const dsat_string_item_type dsat_ipr_tststr [] =
  {
"(0),(45, 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, "
   "19200, 38400, 57600, 115200)" };
LOCAL def_list_type dsat_ipr_list =
{
  12,  /* 19200 */
  dsat_ipr_valstr,
  dsat_ipr_tststr
} ;
```

For the example above, the default value for +IPR will be set to dsat_ipr_valstr[12], which is the string 19200. The response issued for AT+IPR=? will be:

```
(0), (45, 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400,
57600, 115200)
```

3.2.7.4 mixed_def_s_type and dsat_mixed_param_val_type

The mixed_def_s_type is used to hold the default and possible range of values of mixed parameter type arguments of an AT command. Mixed parameter type means the multiple arguments of one AT command can be different types, such as a set of some allowed values, a numerical value in a certain range, or a string of limited length. If the command has N arguments of different types, the default and possible range of values of the arguments are contained in a mixed_def_s_type array of N length. The type of i-th argument of the mixed_def_s_type array, i.e., mixed_argus[], is defined by mixed_argus[i].attrib.

If an AT command has N arguments of different parameter types, the current values of the arguments are contained in dsat_mixed_param_val_type array of length N. The array is used in conjunction with mixed_def_s_type array described in the above paragraph. The i-th element of the dsat_mixed_param_val_type array holds the current value of the i-th argument of an AT command, while the i-th element of mixed_def_s_type array holds the default and possible range of values of the AT command.

The mixed_def_s_type and dsat_mixed_param_val_type are defined as follows:

```
typedef struct
{
  const uint32 attrib;
  const mixed_def_u_type *def;
} mixed_def_s_type;

typedef union
{
  const def_list_type def_list;   /* when attrib contains CONFIG and
                                     LIST */
  const dflm_type     dflm;       /* when attrib contains CONFIG but no
                                     LIST */
  const word          string_len; /* when attrib is STRING, this is the
                                     max length of the string val,
excluding
                                     terminating NULL */
} mixed_def_u_type;

typedef union
{
  dsat_string_item_type *string_item;
  dsat_num_item_type     num_item;
} dsat_mixed_param_val_type;
```

The attrib field of mixed_def_s_type determines which forms the mixed_def_u_type and dsat_mixed_param_val_type really take. It is a 32-bit mask containing the attribute of the argument. See Table 3-3 for command attribute definitions.

Argument types based on the attrib field include:

- If both the CONFIG and LIST bits of attrib are on but STRING is off, this argument is of type LIST. The def union field of mixed_def_s_type takes the form of def_list_type. This argument's current value of union type dsat_mixed_param_val_type takes the form of dsat_num_item_type. It is a number to index into the list of allowed values. See Section 3.2.7.3 for details of how def_list_type is used.
- If the CONFIG bit of attrib is on but both LIST and STRING are off, this argument is a number with lower and upper bound. The def union field of mixed_def_s_type takes the form of dflm_type. This argument's current value of union type dsat_mixed_param_val_type takes the form of dsat_num_item_type. It is the current value of this argument. See Section 3.2.7.2 for details of how dflm_type is used.
- If the STRING bit of attrib is on but LIST is off, this argument is a string with limited length. The def union field of mixed_def_s_type takes the form of word, which contains the maximum length of the string (excluding the terminating NULL). This argument's current value of union type dsat_mixed_param_val_type takes the form of dsat_string_item_type. It is a NULL-terminated string containing the current value of this argument.

Example

For illustrative purposes, suppose the command $EXAM exists and it contains three arguments. The first argument is a string with length 60. The second argument is a list that can be one of the following values, 129, 145, 146, or 147, with a default value of 129. The third argument is a number ranging from 4 to 66 with a default value of 4. This command has the following data structures and command table entry:

```
/* $EXAM Command */
#define MAX_STRING_LEN 60
word dsat_exam_first_string_len = MAX_STRING_LEN;

LOCAL const mixed_def_s_type dsat_exam_first_def_val =
{
  CONFIG | STRING | YES_QUOTE,
  (mixed_def_u_type *) &dsat_exam_first_string_len
};

LOCAL const dsat_string_item_type dsat_exam_second_valstr [][8] =
{
  "129", "145", "146", "147", ""
};

LOCAL const dsat_string_item_type dsat_exam_second_tststr [] =
{
  "(129, 145, 146, 147)"
};

LOCAL const def_list_type dsat_exam_second_def_list =
{
  0, dsat_exam_second_valstr, dsat_exam_second_tststr
};

LOCAL const mixed_def_s_type dsat_exam_second_def_val =
{
  CONFIG | LIST | LOCAL_TEST,
 (mixed_def_u_type *) &dsat_exam_second_def_list
};

LOCAL const dflm_type dsat_exam_third_dflm[] =
{
  { 4, 4, 66 }
};
```

```
LOCAL const mixed_def_s_type dsat_exam_third_def_val =
{
  CONFIG | LOCAL_TEST,
  (mixed_def_u_type *) &dsat_exam_third_dflm
};

LOCAL const mixed_def_s_type *dsat_exam_mixed_dfl[3] =
{
  &dsat_exam_first_def_val,
  &dsat_exam_second_def_val,
  &dsat_exam_third_def_val
};

LOCAL dsat_string_item_type exam_first_val[MAX_STRING_LEN+1];
dsat_mixed_param_val_type dsat_exam_val[3] =
{
  exam_first_val, 0, 4
};

Command table entry dsati_cmd_type:

  { "$EXAM",    EXTENDED | CONFIG | MIXED_PARAM | LIST | LOCAL_TEST,
    SPECIAL_NONE,    3 , DSAT_TABLE_EXAM_IDX,    &dsat_exam_val[0],NULL,
    &dsat_exam_mixed_dfl[0],dsatparm_exec_param_cmd,           NULL },
```

3.2.7.5 tokens_struct_type

The tokens_struct_type is used to define the information generated by the parser for each AT command parsed. It is passed to the command execution function pointed to by proc_func to provide information necessary to process the command. The tokens_struct_type data structure is defined as:

```
typedef struct
{
  byte  *working_at_line;
  byte *name;
  unsigned int op;
  byte *arg[MAX_ARG];
  unsigned int args_found;
  byte *end_of_line;
  cmd_category_enum_type cmd_category;
} tokens_struct_type;
```

- working_at_line is a pointer to the AT command line containing this command.
- name is a pointer to the name of the AT command.
- op is a bit mask of the syntax flags defined in Table 3-6. They are used to specify the syntax used in the AT command. Note, some basic commands consist of a command name followed by a number, with no = in the syntax. This implied = still results in the EQ bit mask being set. An example of this is the E<value> command.

Table 3-6 Command syntax flags

Mode	Description
NA	Name field found in command
EQ	Equal “=” found in command
QU	Query “?” found in command
AR	Argument field found in command

- arg is an array of pointers that point to each argument found in the command.
- args_found is a count of the number of arguments that were found in the command.
- end_of_line is a pointer to the end of the command contained within the memory pointed to by working_at_line. It points to the last NULL character that terminates the command.
- cmd_category defines the category of command this falls into.

Table 3-7 identifies and describes the various command categories.

Table 3-7 Command categories

Mode	Description
BASIC_AT_CMD	Basic AT commands that consist of a single character or “&”, followed by a single alphabetic character and single numeric value, e.g., E0 and D1; basic single character action commands also fall into this category, e.g., D, A, and H
SREG_AT_CMD	S-register AT commands such as S0 and S7
EXT_AT_CMD	Extended AT commands that begin with a “+” character, e.g., +IPR and +IFC
VENDOR_AT_CMD	Vendor-specific AT commands that begin with a “$” and “^” character, e.g., $QCDMG, ^CPBR

3.2.8 Command tables

The top level command table and arrays of command table pointers are in the file dsatmmtgt.c. Refer to Section 3.2.6.1 for information on command table structure.

There are four different command categories, as defined in Section 3.2.7.5. For each category of commands, command tables exist that consist of entries of type dsati_cmd_type. Command tables are in the files dsatctab.c and dsatetsictab.c.

- dsat_basic_table[] – For common basic commands
- dsat_basic_action_table[] – For common basic action commands
- dsat_sreg_table[] – For common S-register commands
- dsat_ext_table[] – For common extended commands
- dsat_vendor_table[] – For common vendor-specific commands
- dsatetsi_ext_table[] – For ETSI-specific extended commands

Each of these categories of commands is parsed using different syntax rules, except for extended and vendor-specific commands; hence, different command tables are needed. Extended and vendor-specific commands are separated into different command tables to reduce the number of commands that must be searched when processing a command.

3.3 ATCoP asynchronous event handler tables

This section describes the ATCoP asynchronous event-handling table, including the structure of an event handler entry.

3.3.1 Asynchronous event handler table structure

ATCoP asynchronous event handling is table-driven. ATCoP asynchronous events received from the data services task dispatcher are passed to the ATCoP asynchronous event handler. The ATCoP event handler then looks up events in the asynchronous event handler table, as illustrated in Figure 3-5.

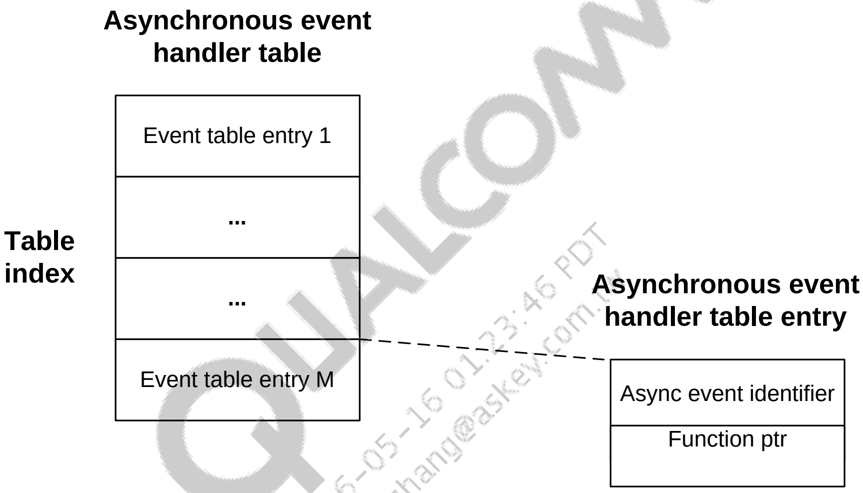

Figure 3-5 Asynchronous event handler table structure

The table is organized such that each entry in the table contains an asynchronous event identifier and an function pointer.

3.3.2 Asynchronous event table entry

The ATCoP asynchronous event handler table contains entries of type dsati_async_event_table_entry_type. The dsati_async_event_table_entry_type data structure and dsati_async_event_handler_type are described in detail in this section. These and other data structures used internally by the ATCoP are defined in dsati.h.

3.3.2.1 dsati_async_event_table_entry_type

```
typedef struct
{
  const ds_cmd_enum_type event;
  const dsati_async_event_handler_type event_handler;
} dsati_async_event_table_entry_type;
```

The fields in the dsati_async_event_table_entry_type data structure are:

- event
  - Asynchronous event to be handled; events are commands to the Data Services task that are routed to ATCoP for asynchronous event handling by the Data Services task dispatcher
  - Events are defined in ds_cmd_enum_type in the file dstask.h; ATCoP events are defined between DS_AT_ASYNC_EVENT_START (DS_CMD_ATCOP_ASYNC_EVENT_START from MPSS BO 2.0+ onwards) and DS_AT_ASYNC_EVENT_END (DS_CMD_ATCOP_ASYNC_EVENT_END from MPSS BO 2.0+ onwards), within the ds_cmd_enum_type
- event_handler
  - Event handler is function pointer; event handler function pointer is independent of Operating mode

3.3.2.2 dsati_async_event_handler_type

```
Before MPSS JO 1.0+: -
typedef dsat_result_enum_type (*dsati_async_event_handler_type)
( dsat_mode_enum_type, ds_cmd_type* );
```

```
MPSS JO 1.0+ onwards: -
typedef dsat_result_enum_type (*dsati_async_event_handler_type)
                                                   ( ds_cmd_type* );
```

The function pointer arguments are:

- dsat_mode_enum_type defines the current AT command mode; these modes defined in Table 3-4 (this argument is not used from MPSS JO 1.0+ onwards and current AT command mode in dsatcmdp_at_state global is used instead of this)
- ds_cmd_type* is a pointer to data services task command type structure defined in dstask.h; this structure contains:
  - Queue link so it can be placed on the Data Services task command queue
  - Event defined in ds_cmd_enum type
  - Union of data specific to each event type that can occur

The function return type should be one of the values defined in Table 3-8.

Table 3-8 Asynchronous event handler function return values

Value	Description
DSAT_OK	OK result indicating asynchronous command execution has completed and executed successfully
DSAT_ERROR	ERROR result indicating asynchronous command execution has completed and execution failed
DSAT_NO_RSP	Result to indicate asynchronous command execution has completed and no command response should be sent to DTE by the ATCoP
DSAT_CMD_ERR_RSP	Indicates asynchronous command execution has completed and the command-specific error response placed in DSM buffer passed to command processor with call to dsatcmdp_handle_async_cmd_rsp() function is sent to DTE only; no result code is sent to DTE as final response to this command
DSAT_ASYNC_CMD	Indicates the asynchronous command being processed has not completed and more asynchronous events associated with this command will be received
DSAT_ASYNC_EVENT	Indicates the asynchronous event being processed is not associated with the processing of an AT command; this allows asynchronous events to be handled that are not directly associated with the processing of an AT command

3.3.3 Asynchronous event handler table

The asynchronous event handler table is located in dsatmmtgt.c (and also dsatmmtgt_ex.c from MPSS BO 2.0+ onwards). Refer to Section 3.4.1 for information on the table structure.

The asynchronous event handler table is async_event_table[] (and also async_event_table_ex[] from MPSS BO 2.0+ onwards).

An event handler function pointer is given for each ATCoP asynchronous event listed in Table 3-8.

3.4 Adding a new AT command

The ATCoP is easily extensible for supporting new AT commands. Follow these steps to add a new AT command:

1. Select a unique name for the command. The command name can be up to 10 characters in length, including the leading +, $, ^, or &.
2. Determine the command attributes. The various commands attributes and their meanings are listed in Table 3-3.
3. Create a unique command identity (cmd_id) for the command and add this command ID into the corresponding command index table.
4. Create a variable to store the argument value(s). If the command has one or more arguments, declare a variable of the appropriate type in dsatctab.c or dsatetsictab.c to store the argument values. The variable should be doubled for dual stack operations. The command variable should not export via extern, command variables must be accessed by get and set utilities defined in dsatutil.c. If the argument values are referenced by modules external to ATCoP, declare get and set utilities in dsat_v.h and protect it by critical section.

5. Define the default value and the allowed range for the command arguments. For commands with simple numeric arguments, use the dflm_type. Initialize the default value and the lower and upper bounds. For commands of type LIST, use the def_list_type. Create the list_v and list_t data structures, and initialize the default value. For commands of type mixed parameter, create and initialize the appropriate dflm_type and def_list_type for each parameter of this type. For string arguments, this is not applicable. All of these variables should be declared in dsatctab.c or dsatetsictab.c. For AT commands whose defaults and/or limits need to be initialized at runtime, this initialization may be performed in the function, dsatctab_data_init.

6. Before MPSS DI 2.0+, make sure val_ptr2 is not set to NULL, if DUAL_STACK attribute is defined in the table entry. Usually, val_ptr2 would contain the address of the expanded variable.

7. Add the command execution function, if any. If the command requires special processing code outside the generic parameter processing functions defined in dsatparm.c, add a new command execution function to the appropriate command processing file (dsatparm.c, dsatact.c, dsatvend.c, dsatetsicall.c, dsatetsipkt.c, dsatetsime.c, dsatetsismsc.c, dsatgsmfax.c). Take advantage of any AT command processing utilities exported by dsati.h or parameter processing exported by dsatparm.h.

8. Add the abort command function, if any. If the command is abortable, an abort command function must be defined in the appropriate command processing file. The command processing files are the same as those identified in step 7.

9. Add the command to the appropriate command table. First, determine which command table the AT command belongs to, based on the command syntax and command type. Then, add the dsati_cmd_type entry to the selected command table. Initialize the dsati_cmd_type command table entry. Initialize the dsati_cmd_type command table entry to point to this new command execution function. If no new special processing code is required, initialize the command table entry to point to the appropriate parameter processing function or NULL, depending on the command response desired. Initialize the abort command function pointer to NULL if the command is not abortable; otherwise, initialize it to point to the abort command function.

10. Compile and test. The AT command is now ready to use.

Example 1

Suppose that you want to add a new AT command, AT$MYCMD, which has two numeric arguments for the first stack. The first argument takes values from 0 to 7, with a default value of 0, and the second argument takes values from 0 to 3, with a default value of 2. Command has no impact on the second stack, therefore, the second stack value pointer (val_ptr2) will be NULL. The command is not abortable. The changes to be made are:

1. Name – The command name is $MYCMD.
2. Attributes – This command has the following attributes:
   - CONFIG – The command has two arguments.
   - EXTENDED – The command uses extended syntax (begins with $).

3. Argument storage – In dsatctab.c:

```
dsat_num_item_type      dsat_mycmd_val[2];
```

For references, use get and set utilities:

- Setting value 2 at index 1 of array dsat_mycmd_val.

```
DSATUTIL_SET_VAL(DSAT_VENDOR_MYCMD_IDX,0,1,2)
```

- Getting value at index 1 of array dsat_mycmd_val.

```
dsatutil_get_val(
             DSAT_VENDOR_MYCMD_IDX,0,1, NUM_TYPE)
```

4. Defaults and limits – In dsatctab.c:

```
LOCAL const dflm_type dsat_mycmd_dflm[] =
{
  { 0, 0, 7 },  /* argument 1: default = 0, range = 0 – 7 */
  { 2, 0, 3 }   /* argument 2: default = 2, range = 0 – 3 */
};
```

5. Special processing – Adding a special attribute will invoke the special processing. Once the appropriate table entry is initialized, this attribute is parsed for and the appropriate functionality needs to be performed.

An example of this command execution function is:

```
dsat_result_enum_type dsatvend_exec_mycmd_cmd
(
  dsat_mode_enum_type mode,             /*  AT command mode             */
  const dsati_cmd_type *cmd_table,      /*  Ptr to cmd in command table */
  const tokens_struct_type *tok_ptr,    /*  Command tokens from parser  */
  dsm_item_type *res_buff_ptr           /*  Place to put response       */
)
{
  dsat_result_enum_type result;

   /*----------------------------------------
     Perform command parameter processing :
   ----------------------------------------*/
   result = dsatparm_exec_param_cmd( mode,
                                     cmd_table,
                                     tok_ptr,
                                     res_buff_ptr );

   /*----------------------------------------
     Perform special processing :
   ----------------------------------------*/
   result = mycmd_action( );

   return result;

} /* dsatvend_exec_mycmd_cmd */
```

6. Abort command processing – Since the command is not abortable, no abort command function needs to be defined.

7. Command table entry – Since the command uses vendor-specific (extended) syntax, it should be added to the dsat_vendor_table[] in dsatctab.c (dsat_vendor_table[] in dsatctab_ex.c and dsat_vendor_table_ex[] in dsatctab.c from MPSS BO 2.0+ onwards). The command table entry looks like this:

Before MPSS DI 2.0+

```
{ "$MYCMD",    EXTENDED | CONFIG,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2,   &dsat_mycmd_val[0],NULL,
        &dsat_mycmd_dflm[0],   dsatvend_exec_mycmd_cmd,      NULL },
```

MPSS DI 2.0+ onwards and before MPSS BO 2.0+

```
{ "$MYCMD",    EXTENDED | CONFIG,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2, &dsat_mycmd_dflm[0],
  dsatvend_exec_mycmd_cmd,      NULL },
```

MPSS BO 2.0+ onwards and before MPSS JO 1.0+

dsat_vendor_table[] in dsatctab_ex.c: -

```
{ "$MYCMD",    EXTENDED | CONFIG,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2, &dsat_mycmd_dflm[0] },
```

dsat_vendor_table_ex[] in dsatctab.c: -

```
{ DSAT_VENDOR_MYCMD_IDX, dsatvend_exec_mycmd_cmd, NULL },
```

MPSS JO 1.0+ onwards

dsat_vendor_table[] in dsatctab_ex.c: -

```
{ "$MYCMD",    EXTENDED | CONFIG,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2, &dsat_mycmd_dflm[0] },
```

dsat_vendor_table_ex[] in dsatctab.c: -

```
{ DSAT_VENDOR_MYCMD_IDX, dsatvend_exec_mycmd_cmd },
```

8. Compile and test – The AT command is now installed in the command table. Recompile the files that were modified and execute.

Example 2

In case AT$MYCMD needs to be supported for DSDS, we need to define the DUAL_STACK attibute in the following table entry:

1. Name – The command name is $MYCMD.
2. Attributes
   - CONFIG – The command has two arguments.
   - EXTENDED – The command uses extended syntax (begins with $).
   - DUAL_STACK – Dual Stack-specific attribute.
3. Argument storage – In dsatctab.c:

```
dsat_num_item_type      dsat_mycmd_val[2][2];
```

For references, use gets and set utilities:

- Setting value 2 at index 1 for second stack

```
DSATUTIL_SET_VAL(DSAT_VENDOR_MYCMD_IDX,1,1,2)
```

- Getting value for second stack at index 1 of array dsat_mycmd_val

```
dsatutil_get_val(
                 DSAT_VENDOR_MYCMD_IDX,1,1, NUM_TYPE)
```

4. Defaults and limits – In dsatctab.c:

```
LOCAL const dflm_type dsat_mycmd_dflm[] =
{
  { 0, 0, 7 },  /* argument 1: default = 0, range = 0 – 7 */
  { 2, 0, 3 }   /* argument 2: default = 2, range = 0 – 3 */
};
```

5. Special processing – Adding a special attribute will invoke the special processing. Once the appropriate table entry is initialized, this attribute is parsed for and the appropriate functionality needs to be performed.

   An example of this command execution function is:

```
dsat_result_enum_type dsatvend_exec_mycmd_cmd
(
  dsat_mode_enum_type mode,             /*  AT command mode             */
  const dsati_cmd_type *cmd_table,      /*  Ptr to cmd in command table */
  const tokens_struct_type *tok_ptr,    /*  Command tokens from parser  */
  dsm_item_type *res_buff_ptr           /*  Place to put response       */
)
{
  dsat_result_enum_type result;

   /*--------------------------------------
     Perform command parameter processing :
   ----------------------------------------*/
   result = dsatparm_exec_param_cmd( mode,
                                     cmd_table,
                                     tok_ptr,
                                     res_buff_ptr );

   /*---------------------------------------
     Perform special processing :
   ----------------------------------------*/
   result = mycmd_action( );

   return result;

} /* dsatvend_exec_mycmd_cmd */
```

6. Abort command processing – Since the command is not abortable, no abort command function needs to be defined.

7. Command table entry – Since the command uses vendor-specific (extended) syntax, it should be added to the dsat_vendor_table[] in dsatctab.c (dsat_vendor_table[] in dsatctab_ex.c and dsat_vendor_table_ex[] in dsatctab.c from MPSS BO 2.0+ onwards). The command table entry looks like this:

   Before MPSS DI 2.0+

```
{ "$MYCMD",    EXTENDED | CONFIG | DUAL_STACK,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2,   &dsat_mycmd_val[0][0],
  &dsat_mycmd_val[1][0],&dsat_mycmd_dflm[0],   dsatvend_exec_mycmd_cmd,
  NULL },
```

MPSS DI 2.0+ onwards and before MPSS BO 2.0+

```
{ "$MYCMD",    EXTENDED | CONFIG | MULTI_STACK,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2, &dsat_mycmd_dflm[0],
  dsatvend_exec_mycmd_cmd,      NULL },
```

MPSS BO 2.0+ onwards and before MPSS JO 1.0+

dsat_vendor_table[] in dsatctab_ex.c: -

```
{ "$MYCMD",    EXTENDED | CONFIG | MULTI_STACK,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2, &dsat_mycmd_dflm[0] },
```

dsat_vendor_table_ex[] in dsatctab.c: -

```
{ DSAT_VENDOR_MYCMD_IDX, dsatvend_exec_mycmd_cmd, NULL },
```

MPSS JO 1.0+ onwards

dsat_vendor_table[] in dsatctab_ex.c: -

```
{ "$MYCMD",    EXTENDED | CONFIG | MULTI_SUBS,
  SPECIAL_NONE, DSAT_VENDOR_MYCMD_IDX,   2, &dsat_mycmd_dflm[0] },
```

dsat_vendor_table_ex[] in dsatctab.c: -

```
{ DSAT_VENDOR_MYCMD_IDX, dsatvend_exec_mycmd_cmd },
```

8. Compile and test – The AT command is now installed in the command table. Recompile the files that were modified and execute.

3.4.1 Adding a new asynchronous AT command

To add an asynchronous command, the steps defined Section 3.4 must be followed with the following differences:

1. Command execution function must return DSAT_ASYNC_CMD. This value must be returned by the command execution function to notify the ATCoP that this is an asynchronous command. The command execution function typically initiates the command processing by calling on the services of another task. The other task then responds by generating an asynchronous event.
2. Add an asynchronous event handler function. An event handler function must be defined to handle the event. Add the new event handler function to the appropriate command processing file (dsatparm.c, dsatact.c, dsatvend.c, dsatetsicall.c, dsatetsipkt.c, dsatetsime.c, dsatetsismsa.c). The event handler must return a value other than DSAT_ASYNC_CMD or DSAT_ASYNC_EVENT to indicate the asynchronous command processing has completed.

3. Add the asynchronous event identifier and event structure to the Data Services task. The event identifier must be added to ds_cmd_enum_type in file dstask_v.h. Add the event identifier between the DS_AT_ASYNC_EVENT_START ((DS_CMD_ATCOP_ASYNC_EVENT_START from MPSS BO 2.0+ onwards) and DS_AT_ASYNC_EVENT_END (DS_CMD_ATCOP_ASYNC_EVENT_END from MPSS BO 2.0+ onwards) items in the enumeration type to ensure the event is handled as an ATCoP asynchronous event. Add this new event identifier to ATCoP commands in the switch statement of function dsi_process_cmds. This function is located in the file dstask.c. Define and add any new event parameters required to be processed by the event handler to the ds_cmd_type structure defined in dstask_v.h.
4. Add the event and handler to the asynchronous event handler table. Add a new dsati_async_event_table_entry_type to the asynchronous event handler table. The table is located in the file dsatmmtgt.c. Initialize the table entry to point to the new event handler function and with the event identifier.

Example

Suppose that you want to make the new AT command, AT$MYCMD, added in the previous example, an asynchronous command. This asynchronous command requires a single asynchronous event to complete processing of the command.

- Difference 1 – Command execution function returns DSAT_ASYNC_CMD

Suppose that a special function, mycmd_action(), which is invoked each time the AT$MYCMD is issued, requests information from another task. The information is then sent to ATCoP from another task by calling the provided callback function to generate an asynchronous event. Since this is a vendor-specific command, these functions must be defined in dsatvend.c. They are given here:

```
dsat_result_enum_type dsatvend_exec_mycmd_cmd
(
  dsat_mode_enum_type mode,             /*  AT command mode             */
  const dsati_cmd_type *cmd_table,      /*  Ptr to cmd in command table */
  const tokens_struct_type *tok_ptr,    /*  Command tokens from parser  */
  dsm_item_type *res_buff_ptr           /*  Place to put response       */
)
{

   /*--------------------------------------
     Perform command parameter handling.
     . . . .
   ----------------------------------------*/

   /*---------------------------------------
     Request information from another task :
   ----------------------------------------*/
   mycmd_action( my_callback_func );

   return DSAT_ASYNC_CMD;
} /* dsatvend_exec_mycmd_cmd */
```

Before MPSS BO 2.0+

```
LOCAL void my_callback_func
(
  other_task_data_type    *cmd
)
{
  ds_cmd_type *cmd_buf;

  /* Send asynchronous event to data services task */
  if( (cmd_buf = ds_get_cmd_buf()) == NULL)
  {
    ERR_FATAL("No ds command buffer",0,0,0);
  }
  else
  {
    cmd_buf->hdr.cmd_id = DS_AT_MY_ASYNC_CMD;
    cmd_buf->cmd.my_cmd.data = *cmd;
    ds_put_cmd( cmd_buf );
  }

}/* my_callback_func */
```

MPSS BO 2.0+ onwards and before MPSS JO 1.0+

```
LOCAL void my_callback_func
(
  other_task_data_type    *cmd
)
{
  ds_cmd_type *cmd_buf;
  dsat_cmd_buf_type *cmd_data;

  /* Send asynchronous event to data services task */
  cmd_buf = dsat_get_cmd_buf(FALSE);
  cmd_data = (dsat_cmd_buf_type *)cmd_buf->cmd_payload_ptr;
  cmd_buf->hdr.cmd_id = DS_CMD_ATCOP_MY_ASYNC_CMD;
  cmd_data->data.cm.my_cmd.data = *cmd;
  ds_put_cmd( cmd_buf );
}/* my_callback_func */
```

MPSS JO 1.0+ onwards

```
LOCAL void my_callback_func
(
  other_task_data_type    *cmd
)
{
  ds_cmd_type *cmd_buf;
  ds_at_mycmd_type *my_cmd;

  /* Send asynchronous event to data services task */
  cmd_buf = dsat_get_cmd_buf(sizeof(ds_at_cm_call_event_type), FALSE);
  my_cmd = cmd_buf->cmd_payload_ptr;
  cmd_buf->hdr.cmd_id = DS_CMD_ATCOP_MY_ASYNC_CMD;
  my_cmd->data = *cmd;
  ds_put_cmd( cmd_buf );
}/* my_callback_func */
```

- Difference 2 – Async event handler function

An asynchronous event is received from the other task when the requested information is available. This occurs when the other task calls the callback function previously defined to pass the data to ATCoP. The asynchronous event handler function defined here handles the event and indicates command processing has completed with the result returned.

```
dsat_result_enum_type dsatvend_my_async_event_handler
(
  dsat_mode_enum_type mode,
  ds_cmd_type *cmd_ptr
)
{
  dsm_item_type *response_buffer;

  /* Create command response and return pointer to DSM item
     containing response. */
  response_buffer = build_mycmd_response( cmd_ptr );

  /* Format response and queue up for transmission to DTE. */
  dsatcmdp_handle_async_cmd_rsp( response_buffer );

  /* Indicate command processing was completed successfully. */
  return DSAT_OK;

} /* dsatvend_my_async_event_handler */
```

- Difference 3 – Add new event

Before MPSS BO 2.0+

Add the new event identifier and command structure to the file dstask_v.h:

```
/*- - - - - - - - - - - - - - - - - - - - - - - - - - - -
  Commands that may be sent to the Data Services Task.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
typedef enum
{
  /* 3G Dsmgr sub-task commands .... */

  DS_RDM_OPEN_CMD,

  /* ATCoP sub-task commands */

  DS_AT_ASYNC_EVENT_START,          /* Start of ATCoP asynchronous events
*/
  DS_AT_SMS_MSG_CMD,
  DS_AT_MY_ASYNC_CMD,
  DS_AT_ASYNC_EVENT_END               /* End of ATCoP asynchronous events
*/

} ds_cmd_enum_type;
/*- - - - - - - - - - - - - - - - - - - - - - - - - - - -
  Command Data types
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
typedef struct
{
  other_task_data_type data;
} ds_at_mycmd_type;

/*- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  Command type: The command header identifies the command.
  If the command has any parameters, they are specified in
  the union.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
typedef struct
{
  ds_cmd_hdr_type    hdr;                               /* Command header
*/

  union
  {
    ds_rdm_open_type       rdm_open;
    ds_at_sms_msg_type         sms_msg;
    ds_at_mycmd_type           my_cmd;
  } cmd;
} ds_cmd_type;
```

Add the new event identifier to switch statement in dstask.c:

```
void dsi_process_cmds( void )
{
  ds_cmd_type  *cmd_ptr;

  /*---------------------------------------------------------
    Get commands from the command queue until the queue is
    empty. For each command received, dispatch the command
    to the appropriate subtask.
  ---------------------------------------------------------*/
  while( (cmd_ptr = (ds_cmd_type *)q_get( &dsi_cmd_q )) != NULL )
  {

    switch( cmd_ptr->hdr.cmd_id )
    {
      /*- - - - - - - - - - - - - - - - - - - - - - - - - - -
           3G Dsmgr Commands
      - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
      case DS_RDM_OPEN_CMD:
        ds3g_process_cmds( cmd_ptr );
        break;
      /*- - - - - - - - - - - - - - - - - - - - - - - - - - -
        ATCoP Commands
      - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
      case DS_AT_SMS_MSG_CMD:
      case DS_AT_MY_ASYNC_CMD:
        dsat_process_async_cmd( cmd_ptr );
        break;

    default:
      ERR_FATAL( "Invalid DS task command: %d", cmd_ptr->hdr.cmd_id, 0, 0
);

    } /* switch */
    /*------------------------------------------------------
      Return the command buffer to the free command queue.
    ------------------------------------------------------*/
    q_put( &dsi_cmd_free_q, &cmd_ptr->hdr.link );

  } /* while */
} /* dsi_process_cmds() */
```

MPSS BO 2.0+ onwards

Add the new event identifier and command structure to the file dstask_v.h:

```
/*- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
-
  Commands that may be sent to the Data Services Task.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
*/
typedef enum
{
  DS_MIN_CMD = 0,

  /*---------------------------------------------------------------------------
--
    DSMGR Commands (1 - 999)
  ---------------------------------------------------------------------------
*/
...
  DS_CMD_RDM_OPEN                              = 21,
  /* Cmd from RDM to open a serial port */
...
/*================ AT COMMAND PROCESSOR (values 6000 - 6999)
================*/

   /*------------------------------------------------------------------------
---
     ATCoP asynchronous events
   -------------------------------------------------------------------------
-*/
   DS_CMD_ATCOP_ASYNC_EVENT_START              = 6000,
   /* Start of ATCoP asynchronous events */

 /*--------------------------------------------------------------------------
-
     CDMA and ETSI SMS AT asynchronous events
   -------------------------------------------------------------------------
-*/

   DS_CMD_ATCOP_SMS_ERR_CMD                    = 6001,
   /* Error Event from UASMS */
...
   DS_CMD_ATCOP_MY_ASYNC_CMD                   = 6049,
   /* My asynchronous command */

   DS_CMD_ATCOP_ASYNC_EVENT_END                  = 6050,
   /* End of ATCoP asynchronous events */
...
  DS_MAX_CMD

} ds_cmd_enum_type;
/*- - - - - - - - - - - - - - - - - - - - - - - - - - -
  Command Data types
- - - - - - - - - - - - - - - - - - - - - - - - - - -*/
typedef struct
{
  other_task_data_type data;
} ds_at_mycmd_type;
```

- Difference 4 – Event handler table entry

Before MPSS BO 2.0+

Add the new asynchronous event handler to the table defined in dsatmmtgt.c:

```
const dsati_async_event_table_entry_type async_event_table[ ] =
{
  {DS_AT_SMS_MSG_CMD,        dsatetsismsa_msg_handler },
  {DS_AT_MY_ASYNC_CMD,       dsatvend_my_async_event_handler },

  {DS_AT_ASYNC_EVENT_END,    NULL}
};
```

MPSS BO 2.0+ onwards

Add the new asynchronous event handler to the table defined in dsatmmtgt.c:

```
const dsati_async_event_table_entry_type async_event_table_entries[ ] =
{
  { DS_CMD_ATCOP_CM_PH_INFO_CMD,     dsatcmif_cm_ph_event_handler_ex },
  { DS_CMD_ATCOP_TIMER_EXPIRED_CMD,  dsatutil_dispatch_timer_handler },
  { DS_CMD_ATCOP_MY_ASYNC_CMD,       dsatvend_my_async_event_handler },

  { DS_CMD_ATCOP_ASYNC_EVENT_END,    NULL}
};
```

Add the new asynchronous event handler to the table defined in dsatmmtgt_ex.c as well if this can be called without AT command i.e. as part of other module events:

```
const dsati_async_event_table_entry_type async_event_table_entries_ex[ ] =
{
  { DS_CMD_ATCOP_CM_PH_INFO_CMD,     dsatcmif_cm_ph_event_handler_ex },
  { DS_CMD_ATCOP_MY_ASYNC_CMD,       dsatvend_my_async_event_handler },

  { DS_CMD_ATCOP_ASYNC_EVENT_END,    NULL}
};
```

The asynchronous AT command is now added. Recompile the files that were modified and execute.

A List of Unsupported Commands

Table A-1 lists specific AT commands for each category of unsupported commands for quick reference.

Table A-1 List of unsupported commands

Command	Description	Specification requirement
DTE–DCE interface commands (ITU-T V.25ter)		
+ILRR	DTE-DCE local rate reporting	Optional
General commands [S1]		
Generic DCE control commands (ITU-T V.25ter)		
+GCI	Country of installation	Mandatory for DCE with adjustable operating parameters required for operation in more than one country
Call control commands [S1]		
+CHSD	High-Speed Circuit-Switched Data (HSCSD) device parameters	Mandatory for HSCSD
+CHST	HSCSD transparent call configuration	Mandatory for transparent HSCSD
+CHSC	HSCSD current call parameters	Optional
+CHSR	HSCSD parameters report	Mandatory for HSCSD
+CHSU	HSCSD automatic user-initiated upgrading	Optional
+CHSA	HSCSD nontransparent asymmetry configuration	Mandatory for nontransparent Enhanced Circuit Switched Data (ECSD)
+CSNS	Single numbering scheme	Optional
+CSIL	Silence command	Optional
Network service-related commands [S1]		
+CAEMLPP	Enhanced Multilevel Precedence and Preemption (eMLPP) priority registration and interrogation	Mandatory for ME without MMI and supplementary service eMLPP is implemented
+CPPS	eMLPP priority subscriptions	Mandatory for an MT supporting AT commands only and eMLPP is implemented
+CFCS	Fast call setup conditions	Mandatory for an MT supporting AT commands only and eMLPP is implemented
+CAAP	Automatic answer for eMLPP service	Mandatory for an MT supporting AT commands only and eMLPP is implemented
+CPNET	Preferred network indication	Optional
+CPNSTAT	Preferred network status	Optional
+CPSB	Current packet-switched bearer	Optional

Command	Description	Specification requirement
+CNAP	Calling name identification presentation	Optional
+COLR	Connected line identification restriction status	Optional
+CSSAC	Service-specific access control restriction status	Optional
+CNEM	Network emergency bearer services support	Optional
+CUUS1	User-to-user signaling supplementary service 1	Optional
+CECUG	Enhanced closed user group	Optional. This command can replace the command +CCUG.
+CCFCU	Communication forwarding number and conditions with URI support	Optional
+CMWI	Message waiting indication control	Optional
Mobile equipment commands [S1]		
+CALA	Alarm	Optional
+CSCC	Secure control command	Optional
+CALM	Alert Sound mode	Optional
+CRSL	Ringer sound level	Optional
+CVIB	Vibrator mode	Optional
+CCWE	Call meter maximum event	Optional
+CPWC	Power class	Optional
+CLAN	Set language	Optional
+CLAE	Language event	Optional
+CSGT	Set greeting text	Optional
+CSVM	Set voicemail number	Optional
+CRMC	Ring melody control	Optional
+CRMP	Ring melody playback	Optional
+CALD	Delete alarm	Optional
+CAPD	Postpone or dismiss an alarm	Optional
+CPROT	Enter Protocol mode	Optional
+CGLA	Generic UICC logical channel access	Optional
+CRLA	Restricted UICC logical channel access	Optional
+CCHO	Open logical channel	Optional
+CCHC	Close logical channel	Optional
+CMOLR	MO location request	Optional
+CSO	Command screen orientation	Optional
+CDSB	Command display screen boundary	Optional
+CPOS	Positioning control	Optional

Command	Description	Specification requirement
+CPOSR	Positioning reporting	Optional
+CMTLR	MT location request notification	Optional
+CMTLRA	MT location request disclosure allowance	Optional
+CBCAP	Battery capacity	Optional
+CBCON	Battery connection status	Optional
+CBCHG	Battery charger status	Optional
+CISRVCC	IMS single radio voice call continuity	Optional
+CIREP	IMS network reporting	Optional
+CPINR	Remaining PIN retries	Optional
+CSUS	Set card slot	Optional
+CAVIMS	Availability for voice calls with IMS	Optional
+CESQ	Extended signal quality	Optional
+CPNER	Primary notification event reporting	Optional
+CIREG	IMS registration information	Optional
+CASIMS	Availability for SMS using IMS	Optional
+CMCCS	Monitor of current calls	Optional
+CLCCS	List of current calls	Optional. Recommended when +CHLD command is implement. This command supports all types of numbers (including SIP URIs) and can replace +CLCC.
+CSRA	Supported radio accesses	Optional. Mandatory when +CFUN supports <fun>=128.
+CCSFB	Circuit switched fallback	Optional
+CMOLRE	Report MO location request error	Optional
+CNEC	Report network error codes	Optional
Packet domain commands [S1])		
+CGEQNEG	Negotiated 3G QoS profile	Optional
+CGCLOSP	Configure local octet stream PAD parameters	Optional
+CGCLPAD	Configure local triple-X PAD parameters (GPRS only)	Optional
+CGANS	Manual response to a network request for PDP context activation	Optional
+CGDEL	Delete nonactive PDP contexts	Optional
+CSCON	Signaling connection status	Optional
+CGAUTH	Define PDP context authentication parameters	Optional
+CIPCA	Initial PDP context activation	Optional
+CEUS	UE’s usage setting for EPS	Optional

Command	Description	Specification requirement
+CEVDP	UE's voice domain preference E-UTRAN	Optional
+CVDP	UE's voice domain preference UTRAN	Optional
+CMMIVT	UE's mobility management IMS voice termination	Optional
+CEPPI	Power Preference Indication for EPS	Optional
S0	Automatic answer (automatic response to a network request for PDP context activation)	Optional
A	Answer (manual acceptance of a network request for PDP context activation)	Optional
H	Hook control (manual rejection of a network request for PDP context activation)	Optional
Commands for VGCS and VBS [S1]		
+CAJOIN	Accept an incoming voice group or voice broadcast call	Mandatory for MT supporting AT commands only and supporting VGCS.
+CAREJ	Reject an incoming voice group or voice broadcast call	Mandatory for MT supporting AT commands only and supporting VGCS.
+CAHLD	Leave an ongoing voice group or voice broadcast call	Mandatory for MT supporting AT commands only and supporting VGCS.
+CAPTT	Talker access for voice group call	Mandatory for MT supporting AT commands only and supporting VGCS.
+CAULEV	Voice group call uplink status presentation	Mandatory for MT supporting AT commands only and supporting VGCS.
+CALCC	List current voice group and voice broadcast calls	Mandatory for MT supporting AT commands only and supporting VGCS.
+CACSP	Voice group or voice broadcast call state attribute presentation	Mandatory for MT supporting AT commands only and supporting VGCS.
+CANCHEV	NCH support indication	Mandatory for MT supporting AT commands only and supporting VGCS.
+COTDI	Originator to dispatcher information	Mandatory for MT supporting AT commands only and supporting VGCS.
+CEPTT	Short data transmission during ongoing VGCS	Mandatory for MT supporting AT commands only and supporting VGCS.
+CGIPC	Group ID prefixes capability	Mandatory for MT supporting AT commands only and supporting VGCS.
D	Request VGCS or VBS service 'D'	Mandatory for MT supporting AT commands only and supporting VGCS.
H	Termination of a voice group or voice broadcast call	Mandatory for MT supporting AT commands only and supporting VGCS.
+CGCS	VGCS subscriptions and GId status	Mandatory for MT supporting AT commands only and supporting VGCS.
+CBCS	VBS subscriptions and GId status	Mandatory for MT supporting AT commands only and supporting VGCS.
Commands for USIM application toolkit [S1]		
+CUSATR	Read USAT profile	Optional

Command	Description	Specification requirement
+CUSATW	Write USAT profile	Optional
+CUSATD	Profile download upon start-up	Optional
+CUSATA	Activate USAT profile	Optional
+CUSATT	Send USAT terminal response	Optional
+CUSATE	Send USAT envelope command	Optional
Commands for enhanced support of dialing [S1]		
+CDU	Dial URI	Optional
+CDUP	Dial URI from phonebook	Optional
+CHCCS	Hang up current calls	Mandatory when + CDU is implemented in the TA.
+CDEFMP	Define media profile	Optional
+CCMMD	Control and modify media description	Optional
+CHUPCD	Hang up for call established with CDU	Optional
Commands from TIA IS-101 [S1]		
+VBT	Buffer threshold setting	Optional
+VCID	Calling number ID presentation	Optional
+VGR	Rx gain selection	Optional
+VGT	Tx gain selection	Optional
+VIP	Initialize voice parameters	Optional
+VIT	Inactivity timer	Optional
+VLS	Line selection	Optional
+VRX	Rx Data state	Optional
+VSM	Select compression method	Optional
+VTD	Tone duration	
+VTX	Tx Data state	
SMS commands [S2]		
+CESP	Enter SMS Block Mode protocol	Mandatory when Block mode is implemented

B References

B.1 Related documents

Title	Number
Qualcomm Technologies, Inc.	
Application Note: Software Glossary for Customers	CL93-V3077-1
QMI AT 1.2, QMI Access Terminal Svc Spec	80-VB816-22
Standards	
AT command set for 3GPP User Equipment (UE)	3GPP TS 27.007 (Release 9)
Equipment (DTE - DCE) interface for Short Message Service (SMS) and Cell Broadcast Service (CBS)	3GPP TS 27.005 (Release 7)
Serial Asynchronous Automatic Dialing and Control	ITU-T V.25ter (Jul 1997)
Data Transmission Systems and Equipment — Serial Asynchronous Automatic Dialing and Control for Character Mode DCE on Wireless Data Services	PCCA STD-101 (Jun 1995)
In-band DCE control and synchronous data modes for asynchronous DTE	ITU-T V.80 (Aug 1996)
Asynchronous Facsimile DCE Control, Service Class I	ITU-T T.31 (Aug 1995)
Data Service Options for Spread Spectrum Systems	TIA/EIA/IS-707-A-2 (Mar 2001)
Data Transmission Systems and Equipment – Serial Asynchronous Automatic Dialing and Control – Extended Command Syntax	TIA/EIA-615 (Dec 1993)
Data Service Options for Wideband Spread Spectrum Systems	TIA/EIA/IS-707 (Feb 1998)
Asynchronous Facsimile DCE Control Standard, Service Class 2	TIA/EIA-592 (Aug 1990)
Facsimile Digital Interfaces - Amendments to TIA/EIA-592 to Support ITU-T T.30	TIA/EIA/ IS-134 (1993)
Data Transmission Systems and Equipment - Extensions to Serial Asynchronous Dialing and Control	TIA/EIA IS-131
Terminal Equipment to User Equipment (TE-UE) multiplexer protocol	3GPP TS 27.010