LLD CPHS Interface

Project	G23-GSM Protocol Stack
Document Type	Technical Documentation
Title	LLD CPHS Interface
Author	Marc Droste-Franke
Creation Date	24 October, 2002
Last Modified	25 July, 2005
ID and Version	8462.730.02.006
Status	Being Processed

0 Document Control

0.1 Document History

ID	Author	Date	Status
8462.730.02.002	CLB	29 October, 2002	Creation
8462.730.02.002	CLB	11 November, 2002	Being Processed
8462.730.02.002	KGT	11 December, 2002	Being Processed
8462.730.02.002	KGT	21 January, 2003	Being Processed
8462.730.02.003	TLU	15 April, 2003	Being Processed
8462.730.02.004	TLU	30 April, 2003	Being Processed
8462.730.02.005	MDF	13 September, 2004	Being Processed
8462.730.02.006	AJS	25 July, 2005	Being Processed

0.2 References, Abbreviations, Terms

[TI 7010.801]	7010.801, References and Vocabulary, Texas Instruments.

Table of Contents

1 Introduction

G23 is a software package implementing Layers 2 and 3 of the ETSI-defined GSM air interface signalling protocol, and as such represents that part of a GSM mobile station's protocol software which is both, platform and manufacturer independent. Therefore, G23 can be viewed as a building block providing standardised functionality through generic interfaces for easy integration.

The G23 suite of products consists of the following items:

- Layers 2 and 3 for speech & short message services,
- Layers 2 and 3 for fax & data services,
- Application Control Interface,
- Slim MMI [02.30] and
- Test and integration support tools.

2 Overview

2.1 General

The CPHS functionality is to be implemented as a separate module of ACI. AT-Interpreter and MFW will access its functionalities by means of an AT-command based interface using some newly defined AT-commands. Another interface also should allow internal communication with ACI. The aim of this document is the description of these two interfaces.

The CPHS module provides the whole functionality described in the document CPHS Phase 2 version 4.2 from 27 February 1997 with the exception of the "Network and Service Provider Lock" section. The latter has indeed already been implemented as an extra module in ACI (see doc 8463.600).

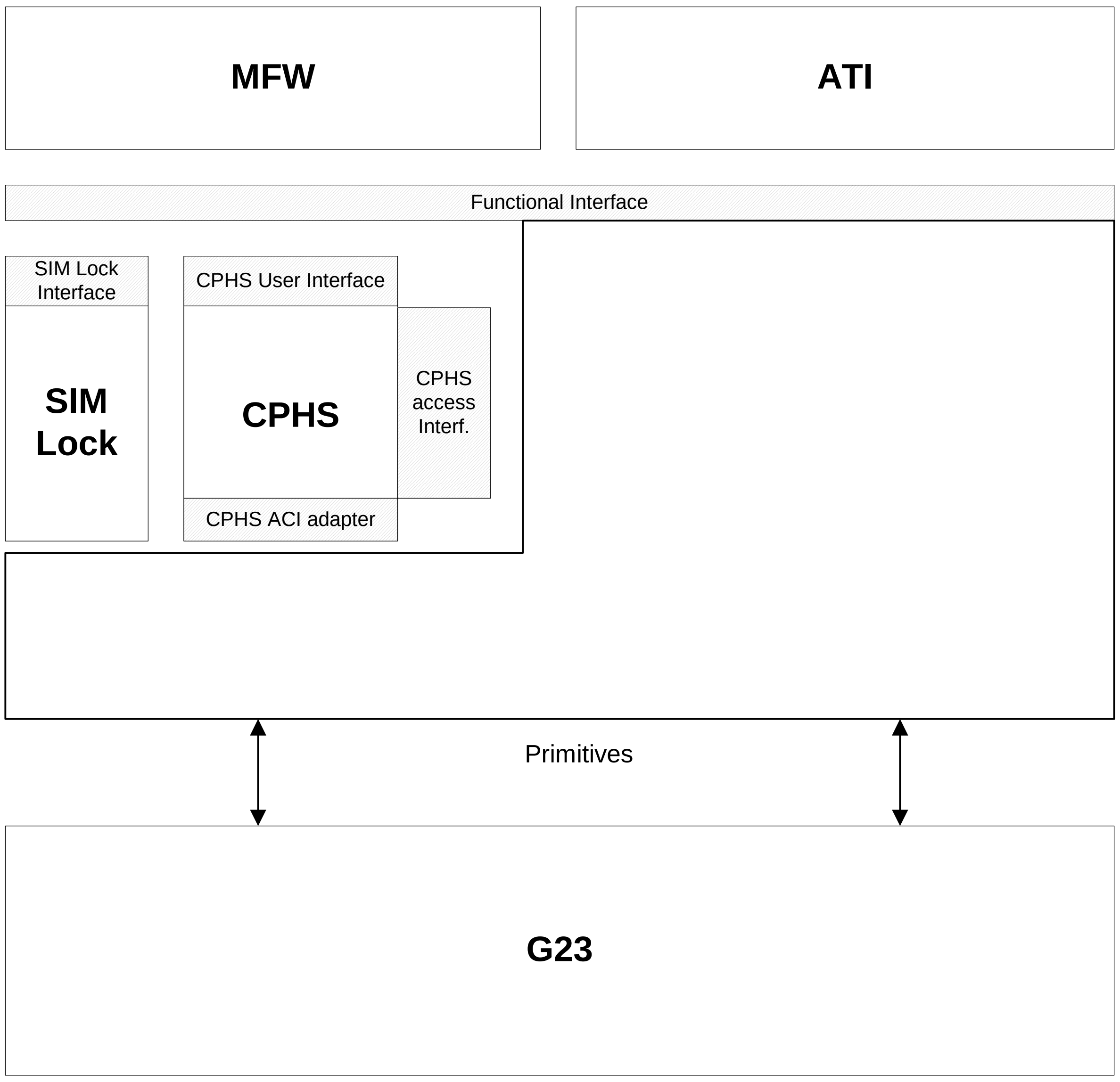
The module caches most of the CPHS SIM values. This allows functions to return immediately and avoids the use of a callback mechanism.

2.2 Functionality

The CPHS library provides the following features:

- Network Operator Name
- Home Country Roaming Indicator
- Voice Message Waiting Indicator
- Diverted Call Indicator
- Current Line Indicator
- Alternate Line Service (ALS)
- Emergency Calling (dialing 999)
- Call Forwarding flag
- Voice message waiting flag
- Customer Service Profile (CSP)
- CPHS Information
- Mailbox Numbers
- Information Numbers

2.3 Architecture



3 Interface Description

3.1 Basic Principle

The CPHS functionality is implemented in a separated module of ACI. To ensure the independence of this module, three interfaces are here defined.

The first one is a so-called *user interface*. This interface will be used by any application willing to use CPHS functionality. In the current application, this interface is used by the Command Handler part of ACI. Some operations may be asynchronous (message sent to lower entity such as SIM: stack returns before having the answer). In such cases, the CPHS module will use a callback previously passed by ACI during initialisation phase.

The CPHS module needs also a so-called *access interface* so that the protocol stack (through ACI) may exchange information with it. This interface may be used by the Command Handler or the Protocol Stack Adapter.

In some cases, the CPHS module will also have to use some ACI features, such as accessing data on the SIM. The CPHS module should as much as possible use the ACI/MFW functional interface in theses cases. However, given the current implementation of this interface, it is only possible with function immediately returning successfully (synchronous case). This is due to the current impossibility to redirect callbacks. Asynchronous ACI features should only be necessary for SIM accessing: for this case corresponding cmhSIMfunctions may be used. However in order to limit ACI dependency, calls of such functions shall be encapsulated in an ACI adapter layer in the CPHS module.

This document also contains the description of an AT-command-based interface for CPHS features. AT commands and corresponding Command Handler functions (for MFW/BMI use) here described give full control over CPHS functionalities for any type of user (ATI or MFW).

3.2 CHANGES

3.3 General Data Types

3.3.1 T_CPHS_RET

Definition:

```
typedef enum
{
  CPHS_FAIL                    = -1,
  CPHS_OK,
  CPHS_EXEC,
  CPHS_BUSY,
  CPHS_NOT_INIT,
} T_CPHS_RET;
```

Elements:

CPHS_FAIL	unspecific failure
CPHS_OK	successful execution
CPHS_EXEC	execution in progress
CPHS_BUSY	module is already in use… Retry later

CPHS_NO_INIT	module has not been initialised

Description:

These are the different return codes possible when calling a CPHS interface function.

3.3.2 T_CPHS_LINES

Elements:

CPHS_LINE_NULL	line parameter is irrelevant
CPHS_LINE1	standard voice line
CPHS_LINE_DATA	bearer data line
CPHS_LINE_FAX	bearer fax line
CPHS_LINE2	auxiliary speech line (in case of ALS)

Description:

These are the different lines CPHS functionalities may apply to.

Definition:

```
#define  CPHS_LINE_NULL      (0x0000)
#define  CPHS_LINE1          (0x0001)
#define  CPHS_LINE_DATA      (0x0002)
#define  CPHS_LINE_FAX       (0x0004)
#define  CPHS_LINE2          (0x0100)
```

These values have been chosen for consistency with the concept of classes in AT commands definitions such as CCFC or CLCK.

Type definition:

```
#define  T_CPHS_LINES        USHORT
```

The reason why no enum is being used here is that CPHS lines are to be used in bit fields. So it is more handy to have these #defines in order to handle CPHS lines.

3.3.3 T_CPHS_CB

Definition:

```
typedef enum
{
  CPHS_INIT_RES =      1,
  CPHS_ROAM_IND,
  CPHS_VOICE_MAIL_IND,
  CPHS_VOICE_MAIL_RES,
  CPHS_CFU_RES
} T_CPHS_CB;
```

Elements:

CPHS_INIT_RES	Result of initialisation
CPHS_ROAM_IND	Roaming indicator
CPHS_VOICE_MAIL_IND	Incoming voice mail indication

CPHS_VOICE_MAIL_RES	Read/Write voice mail flag result
CPHS_CFU_RES	Read/Write call diverted flag result

Description:

These are the different contexts where the CPHS module may call the ACI callback.

3.3.4 T_CPHS_PARAMS

```
typedef struct
{
  T_CPHS_CB cb_type;
  T_CPHS_RET operation_result;
  UBYTE set_flag;
  T_CPHS_LINES line;
} T_CPHS_PARAMS;
```

Elements:

cb_type	Callback type of ACI callback
operation_result	Return value of the CPHS interface functions
set_flag	Auxiliary setflag
line	Different lines CPHS functionalities may apply to

Description:

This is the structure passed by CPHS module when it has to inform the user. Cb_type informs about what CPHS functionality is involved and thus how to interpret the set_flag data.

3.3.5 T_CPHS_USER_CB

```
typedef void T_CPHS_USER_CB( T_CPHS_PARAMS *params );
```

Description:

Prototype type of callback to be used by the CPHS module to inform user (ATI/MFW).

3.4 General Functions

3.4.1 cphs_check_status

Prototype:

T_CPHS_RET cphs_check_status(void);

Parameters:

void

Return:

CPHS_BUSY	CPHS module is busy
CPHS_OK	CPHS module is initialised and ready to process
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

This function checks whether CPHS module has been initialised or if it is currently being used.

3.4.2 cphs_line_makes_sense

Prototype:

BOOL cphs_line_makes_sense(T_CPHS_LINES *line***);**

Parameters:

line	Different lines CPHS functionalities may apply to

Return:

TRUE	line is a CPHS line
FALSE	line is not a CPHS line

Description:

This function returns TRUE if the given line is a valid CPHS line..

3.5 START/END/REFRESH CPHS MODULE

3.5.1 ACI/CPHS Module functional interface

3.5.1.1 cphs_start

Prototype:

T_CPHS_RET cphs_start (T_CPHS_USER_CB **cphs_user_cb***);**

Parameters:

cphs_user_cb( )	contains callback to be used by the CPHS module to inform user (ATI/MFW).

Return:

CPHS_EXEC	CPHS module is being initialised
CPHS_FAIL	an error occurred (e.g. parameter passed is a NULL pointer)
CPHS_BUSY	CPHS has already been started to initialise
CPHS_OK	CPHS module already initialised

Description:

Initialises the CPHS module (read some parameters from the SIM).

3.5.1.2 cphs_refresh_data

Prototype:

T_CPHS_RET cphs_refresh_data (void);

Parameters:

void

Return:

CPHS_EXEC	CPHS module is being initialised
CPHS_FAIL	an error occurred
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Refresh the cached CPHS parameter from SIM.

3.5.1.3 cphs_stop

Prototype:

T_CPHS_RET cphs_stop (void);

Parameters:

void

Return:

CPHS_OK	execution OK
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Ends CPHS module functionalities. **cphs_start( )** will be needed before using the CPHS library again.

3.5.2 AT command

3.5.2.1 AT% CPHS

Table 1: %CPHS parameter command syntax

Command	Possible response(s)
`%CPHS=<init_mode>`	*+CME ERROR: <err>*
`%CPHS?`	`%CPHS: <init_mode>` *+CME ERROR: <err>*
`%CPHS=?`	`%CPHS: (`list of supported `<init_mode>`s`)` *+CME ERROR: <err>*

Description
This set command allows initialising and closing CPHS functionalities. Initialising is mandatory if the user aims at using CPHS functionality (e.g. retrieving voice waiting message). When initialising, the CPHS module proceeds to retrieve data from SIM to cache them internally, this allowing direct availability of these data. Initialising means also enabling all CPHS related indications (e.g. Call Diverted Indicator, Call Waiting Message Indicator, etc…).
Closing CPHS functionality will stop CPHS indications to be displayed. It will also free the related memory needed to cache CPHS data.
If setting fails in an ME error, `+CME ERROR: <err>` is returned. Refer subclause 9.2 for `<err>` values.
Read command returns the current state of CPHS functionalities: initialised, refreshing (only while refreshing is in progress. When progress is done, initialised will be shown) or closed.
Test command returns supported `<init_mode>`-values.

Defined values
`<init_mode>`: integer type value indicating:

0 – Closes CPHS functionalities (default).

1 – Initializes CPHS functionalities.

2 – Refreshes CPHS data cached from SIM.

3.5.3 ACI/MFW functional interface

3.5.3.1 Data types

3.5.4 T_ACI_CPHS_INIT

Definition:

```
typedef enum
{
  ACI_CPHS_CLOSE      = 0,
  ACI_CPHS_INIT,
  ACI_CPHS_REFRESH,
  ACI_CPHS_BUSY
} T_ACI_CPHS_INIT;
```

Elements:

ACI_CPHS_CLOSE	close CPHS functionalities
ACI_CPHS_INIT	initialised CPHS functionalities

ACI_CPHS_REFRESH	refreshes CPHS data
ACI_CPHS_BUSY	CPHS module is busy

Description:

These are the different values of parameter init_cphs for sAT_PercentCPHS and qAT_PercentCPHS.

3.5.4.1 sAT_PercentCPHS

Prototype:

T_ACI_RETURN sAT_PercentCPHS(T_ACI_CMD_SRC *srcId*, **T_ACI_CPHS_INIT** *init_cphs*);

Parameters:

init_cphs	Initialising/closing/refreshing CPHS module.

Return:

AT_EXCT	CPHS module is initialising
AT_CMPL	operation successful (can occur: e.g. end CPHS)
AT_FAIL	an error occurred

Description:

Starts/ends/refreshes CPHS functionalities.
When initialising, cphs_start( ) will be called with parameter cphs_user_cb = &cmhCPHS_user_cb( ), then void cmhCPHS_user_cb(T_CPHS_PARAMS *params) is to be implemented in the command handler part of ACI (file cmh_cphs.c). The function will then take care of providing the right user with the right data (using the right R_AT callback).

3.5.4.2 qAT_PercentCPHS

Prototype:

T_ACI_RETURN qAT_PercentCPHS(T_ACI_CMD_SRC *srcId*, **T_ACI_CPHS_INIT** * *init_cphs*);

Parameters:

init_cphs	Contains status of CPHS module.

Return:

AT_CMPL	status of CPHS module has been successfully retrieved
AT_FAIL	an error occurred

Description:

Queries the status of CPHS functionalities.

3.5.5 MSC

```
[Starts CPHS functionality or refreshes CPHS data from SIM]
AT%CPHS=1 or AT%CPHS=2
OK

ATI/MFW           CMH              CPHS                 PSA/CMH           SIM
   |               |                 |                    |                |
   |               |                 |                    |                |
 (1): CPHS not initialised yet
   |  sAT_%CPHS(1) |                 |                    |                |
   *-------------->*                 |                    |                |
   |               |   cphs_start()  |                    |                |
   |               *---------------->*                    |                |
   |               |                 |                    |                |
  (end of 1)
  OR
 (2): CPHS already initialised
   |  sAT_%CPHS(2) |                 |                    |                |
   *-------------->*                 |                    |                |
   |               |cphs_refresh_data|                    |                |
   |               *---------------->*                    |                |
   |               |                 |                    |                |
                  (end of 2)
                  (3)
   |               |                 | cmhSIM_ReadTranspEF|                |
   |               |                 *------------------->*                |
   |               |                 |                    |                |
   |               |                 |                    |  SIM_READ_REQ  |
   |               |                 |                    *===============>*
   |               |                 |                    |                |
   |               |    CPHS_EXEC    |                    |                |
   |               *<--%ret result%--*                    |                |
   |               |                 |                    |                |
   |    AT_EXCT    |                 |                    |                |
   *<-%ret result%-*                 |                    |                |
   |               |                 |                    |                |
   |               |                 |                    |  SIM_READ_CNF  |
   |               |                 |                    *<===============*
   |               |                 |                    |                |
   |               |                 | cphs_sim_data_cb() |                |
   |               |                 *<-------------------*                |
                  (end of 3) : (3) can be repeated if needed

   |               |                 |                    |                |
   |               |cmhCPHS_user_cb()|                    |                |
   |               | CPHS_INIT_RES   |                    |                |
   |               *<----------------*                    |                |
   |               |                 |                    |                |
   |   rAT_OK( )   |                 |                    |                |
   *<--------------*                 |                    |                |
   |               |                 |                    |                |


[Ends CPHS functionality]
AT%CPHS=0
OK

ATI/MFW           CMH              CPHS                 PSA               SIM
   |               |                 |                    |                |
   | sAT_%CPHS(0)  |                 |                    |                |
   *-------------->*                 |                    |                |
   |               |                 |                    |                |
   |               |   cphs_stop()   |                    |                |
   |               *---------------->*                    |                |

                  (1): CPHS not initialised yet
   |               |                 |                    |                |
   |               |  CPHS_NOT_INIT  |                    |                |
   |               *<--%ret result%--*                    |                |
   |               |                 |                    |                |
   |   rAT_CME()   |                 |                    |                |
   *<--------------*                 |                    |                |
   |               |                 |                    |                |
                  (end of 1)
                 OR
                  (2): CPHS had been initialised
   |               |                 |                    |                |
   |               |    CPHS_OK      |                    |                |
   |               *<--%ret result%--*                    |                |
   |               |                 |                    |                |
   |   rAT_OK( )   |                 |                    |                |
   *<--------------*                 |                    |                |
  |              |                |                  |              |
             (end of 2)
  |              |                |                  |              |
```

3.6 NETWORK OPERATOR NAME

3.6.1 ACI/CPHS Module functional interface

3.6.1.1 cphs_get_opn

Prototype:

T_CPHS_RET cphs_get_opn (CHAR ******longname*, **UBYTE** **max_longname*, **CHAR** ******shortname*, **UBYTE** **max_shortname*);

Parameters:

longname	operator long name
max_longname	maximal length of long name: IN: should contain the size of memory allocated for *longname*. OUT: If memory is not enough, then function returns with CPHS_FAIL and max_longname will contain the amount of memory needed. Otherwise contains amount of written characters.
shortname	operator short name
max_shortname	maximal length of short name: IN: should contain the size of memory allocated for *shortname*. OUT: If memory is not enough, then function returns with CPHS_FAIL and max_shortname will contain the amount of memory needed. Otherwise contains amount of written characters.

Return:

CPHS_OK	execution OK
CPHS_BUSY	CPHS module is busy
CPHS_FAIL	error occurred: if *max_buffer is not 0, then reason is memory for *buffer is not enough: check max_buffer for amount of memory needed.
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Returns the operator long name and short name if available.

3.6.2 AT command

3.6.2.1 AT% CPOPN

Table 2: %CPOPN parameter command syntax

Command	Possible response(s)
`%CPOPN?`	%CPOPN: <LONG_NAME>[,<SHORT_NAME>] *+CME ERROR: <err>*
`%CPOPN=?`	

Description

%CPOPN: "LONG NAME"[, "SHORT NAME"] will be displayed as intermediate result if operation is successful (short name is optional: it will be shown only if SIM supports this feature).

This query command returns the long and short names of the operator as stored in the corresponding SIM fields.

Defined values

`<LONG_NAME>`: string type indicating: Operator Long Name

`<SHORT_NAME>`: string type indicating: Operator Short Name (optional: may not be present in SIM)

3.6.3 ACI/MFW functional interface

3.6.3.1 qAT_PercentCPOPN

Prototype:

T_ACI_RETURN qAT_PercentCPOPN(T_ACI_CMD_SRC *srcId*, **CHAR** **longname*,
UBYTE **max_longname*, **CHAR** **shortname*, **UBYTE** * *max_shortname*);

Parameters:

longname	operator long name
max_longname	maximal length of long name: IN: should contain the size of memory allocated for *longname*. OUT: If memory is not enough, then function returns with AT_FAIL and max_longname will contain the amount of memory needed. Otherwise contains amount of written characters.
shortname	operator short name
max_shortname	maximal length of short name: IN: should contain the size of memory allocated for *shortname*. OUT: If memory is not enough, then function returns with AT_FAIL and max_shortname will contain the amount of memory needed. Otherwise contains amount of written characters.

Return:

AT_CMPL	successfully completed
AT_FAIL	an error occurred: if *max_buffer is not 0, then reason is memory for *buffer is not enough: check max_buffer for amount of memory needed.

Description:

Queries long and short names of the operator.

3.6.4 MSC

```
[Gets operator long and/or short name]
AT%COPN?
%COPN: "operator long name","op"
OK

ATI/MFW             CMH              CPHS                 PSA              SIM
   |                 |                 |                    |                 |
   |  qAT_%CPOPN()   |                 |                    |                 |
   *---------------->*                 |                    |                 |
   |                 |                 |                    |                 |
   |                 | cphs_get_opn()  |                    |                 |
   |                 *---------------->*                    |                 |
   |                 |                 |                    |                 |
   |                 |    CPHS_OK      |                    |                 |
   |                 *<--%ret result%--*                    |                 |
   |                 |                 |                    |                 |
   |     AT_CMPL     |                 |                    |                 |
   *<--%ret result%--*                 |                    |                 |
   |                 |                 |                    |                 |
```

3.7 HOME COUNTRY ROAMING INDICATOR

3.7.1 ACI/CPHS Module functional interface

3.7.1.1 cphs_roaming_ind

Prototype:

void cphs_roaming_ind(UBYTE *roaming_status***);**

Parameters:

roaming_status	roaming status
	Values:
	#define CPHS_ROAMING_ON (1)
	#define CPHS_ROAMING_OFF (0)

Description:

Informs the CPHS module that the *Roaming* state has changed.

3.7.2 AT command

3.7.2.1 %CPROAM

Table 3: %CPROAM parameter unsolicited result syntax

Unsolicited Result
`%CPROAM:` <roam_status>

Description

This unsolicited result will be displayed whenever the mobile's Roaming registration state has changed.

Defined values

< roam_status>: integer type value indicating:

0	– Mobile is not in Roaming state.

`1` – Mobile is in Roaming state (mobile is registered to an other than the home network).

3.7.3 ACI/MFW functional interface

3.7.3.1 rAT_PercentCPROAM

Prototype:

void rAT_PercentCPROAM(UBYTE *roam_status***);**

Parameters:

roaming_status	roaming status
	Values:
	#define CPHS_ROAMING_ON (1)
	#define CPHS_ROAMING_OFF (0)

Return:

void

Description:

Informs ATI/MFW that the mobile's *Roaming* state has changed.

3.7.4 MSC

```
[CPHS Roaming Indicator]
Unsolicited Message:
%CPROAM: 1

ATI/MFW          CMH              CPHS                  PSA             MM/GMM
   |               |                 |                    |                  |
 (GPRS case)
   |               |                 |                    |GMMREG_ATTACH_CNF|
   |               |                 |                    *<================*
   |               |                 |                    |    "Roaming"     |
   |               |                 |                    |                  |
   OR
 (GSM case)
   |               |                 |                    |  MMREG_REG_CNF   |
   |               |                 |                    *<=================*
   |               |                 |                    |    "Roaming"     |
   |               |                 |                    |                  |

   |               |                 | cphs_roaming_ind() |                  |
   |               |                 |  CPHS_ROAMING_ON   |                  |
   |               |                 *<-------------------*                  |
   |               |                 |                    |                  |
   |               |cmhCPHS_user_cb()|                    |                  |
   |               | CPHS_ROAM_IND   |                    |                  |
   |               *<----------------*                    |                  |
   |               |                 |                    |                  |
   |rAT_%CPROAM(1) |                 |                    |                  |
   *<--------------*                 |                    |                  |
   |               |                 |                    |                  |
   |               |                 |                    |                  |
   |               |                 |                    |                  |
```

3.8 VOICE MESSAGE WAITING

3.8.1 ACI/CPHS Module functional interface

3.8.1.1.1 cphs_voice_mail_ind

Prototype:

void cphs_voice_mail_ind(UBYTE *set_flag,* **T_CPHS_LINES** *line***);**

Parameters:

set_flag	waiting flag is to be set/erased on the SIM
line	line to which apply received Voice Mail Indication.

Return:

void

Description:

Indication to the CPHS module that a Voice Mail related SMS has been received from the network.

3.8.1.2 Voice Message Waiting Indicator Setting/Clearing/Reading

3.8.1.2.1 cphs_set_waiting_flag

Prototype:

T_CPHS_RET cphs_set_waiting_flag(UBYTE *set_flag,* **T_CPHS_LINES** *lines***);**

Parameters:

set_flag — waiting flag is to be set/erased on the SIM

Values:

```
#define CPHS_SET_WAITING_FLAG  (1)
#define CPHS_ERASE_WAITING_FLAG (0)
```

lines — lines associated to waiting flag indicator: it is a bit mask, so it can be used to change the flag status for several lines at once.

Return:

CPHS_EXEC	operation in progress
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Controls state of Waiting Indicator Flag on the SIM.

3.8.1.2.2 cphs_get_waiting_flag

Prototype:

T_CPHS_RET cphs_get_waiting_flag(UBYTE **flag_set,* **T_CPHS_LINES** *line***);**

Parameters:

flag_set — contains flag status.

Values:

```
#define CPHS_FLAG_DEACTIVATED  (0)
#define CPHS_FLAG_ACTIVATED  (1)
#define CPHS_FLAG_NOT_PRESENT  (2)
```

	#define CPHS_FLAG_ERROR (3)
line	line associated to waiting flag indicator: it is NOT a bit mask !!

Return:

CPHS_OK	operation successful
CPHS_FAIL	an error occurred (ex: line contains more than one line)
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Retrieves state of waiting indicator flag for a given bearer capability from the CPHS cached data.

3.8.2 AT command

3.8.2.1 % CPVWI

Table 4: %CPVWI parameter command syntax

Command	Possible response(s)
`%CPVWI=<mode>,<line>`	`%CPVWI: <status>[,<line>]` *+CME ERROR: <err>*
`%CPVWI?`	*+CME ERROR: <err>*
`%CPVWI=?`	`%CPVWI:` (list of supported `<mode>s,<line>s)` *+CME ERROR: <err>*

Table 5: %CPVWI parameter unsolicited result syntax

Unsolicited Result
`%CPVWI: <status>,<line>`

Description

This set command allows setting, clearing, querying of the status of the Voice Waiting Message Flag for one or several lines.

When querying, intermediate result is %CPVWI: status(,lines) (lines is optional: either Voice Waiting Message flag is set for no line and intermediate result is %CPVWI: 0, or Voice Waiting Message flag is set for at least one line and intermediate result is %CPVWI= 1,set_lines).

An unsolicited result %CPVWI: status,line will be displayed upon receiving a Voice Message Waiting SMS from the network.

Defined values

`<mode>`: integer type value indicating:

0	– Clear Voice Waiting Message Flag.
1	– Set Voice Waiting Message Flag.
2	– Query Voice Waiting Message Flag status.

`<line>`: bit field type value indicating possible sum of following values:

1	– Line 1
2	– Data
4	– Fax
256	– Line 2

`<status>`: integer type value indicating:

0	– Clear
1	– Set

3.8.3 ACI/MFW functional interface

3.8.3.1 sAT_PercentCPVWI

Prototype:

T_ACI_RETURN sAT_PercentCPVWI(T_ACI_CMD_SRC *srcId*, **UBYTE** *flag_set,* **T_CPHS_LINES** *lines***);**

Parameters:

flag_set	flag status. Values: #define CPHS_SET_WAITING_FLAG (1) #define CPHS_ERASE_WAITING_FLAG (0)
lines	line to which change applies: it is a bit mask, so it can be used to change the flag status for several lines at once.

Return:

AT_EXCT	operation in progress
AT_FAIL	an error occurred

Description:

Allows setting or clearing the voice waiting message flag for any line.

3.8.3.2 qAT_PercentCPVWI

Prototype:

T_ACI_RETURN qAT_PercentCPVWI(T_ACI_CMD_SRC *srcId*, **UBYTE** * *flag_set,* **T_CPHS_LINES** *line***);**

Parameters:

flag_set	flag status. Values: #define CPHS_FLAG_DEACTIVATED (0) #define CPHS_FLAG_ACTIVATED (1) #define CPHS_FLAG_NOT_PRESENT (2) #define CPHS_FLAG_ERROR (3)
line	line to which query applies: it is NOT a bit mask !!

Return:

AT_CMPL	successfully completed
AT_FAIL	an error occurred: (ex: line contains more than one line)

Description:

Queries the current setting for voice waiting message flag for each line.

3.8.3.3 rAT_PercentCPVWI

Prototype:

void rAT_PercentCPVWI(UBYTE *flag_set,* **T_CPHS_LINES** *line***);**

Parameters:

flag_set	flag status. Values: #define CPHS_SET_WAITING_FLAG (1)

	#define CPHS_ERASE_WAITING_FLAG (0)
line	line to which query applies

Return:

void

Description:

Informs ATI/MFW that a new Voice Message is awaiting retrieval.

3.8.4 MSC

```
[CPHS Voice Message Waiting Indicator]
Unsolicited Message:
%CPVWI: 1,1        New Voice Waiting Message for line 1.

ATI/MFW           CMH              CPHS                  PSA              SMS
   |               |                 |                    |                |
   |               |                 |                    |  MNSMS_MSG_IND |
   |               |                 |                    *<===============*
   |               |                 |                    |“Voice Msg Wait”|
   |               |                 |                    |                |
   |               |                 | cphs_voice_mail_ind|                |
   |               |                 *<-------------------*                |
   |               |                 |                    |                |
   |               |                 |psaSIM_AccessSIMData|                |
   |               |                 *------------------->*                |
   |               |                 |                    |                |
   |               |                 |                    | SIM_UPDATE_REQ |
   |               |                 |                    *===============>*
   |               |                 |                    |                |
   |               |                 |                    |                |
   |               |                 |                    | SIM_UPDATE_CNF |
   |               |                 |                    *<===============*
   |               |                 |                    |                |
   |               |                 | cphs_sim_data_cb() |                |
   |               |                 *<-------------------*                |
   |               |                 |                    |                |
   |               |cmhCPHS_user_cb()|                    |                |
   |               | VOICE_MAIL_IND  |                    |                |
   |               *<----------------*                    |                |
   |               |                 |                    |                |
   |  rAT_%CPVWI() |                 |                    |                |
   *<--------------*                 |                    |                |
   |               |                 |                    |                |
```

```
[CPHS Set/Clear Waiting Flag Indicator]
AT%CPVWI=0          Clears flag indicators for all lines.
OK

ATI/MFW           CMH                  CPHS                  PSA              SIM
   |               |                     |                    |               |
   | sAT_%CPVWI()  |                     |                    |               |
   *-------------->*                     |                    |               |
   |               |                     |                    |               |
   |               |cphs_set_waiting_flag|                    |               |
   |               *-------------------->*                    |               |
   |               |                     |                    |               |
   |               |                     |psaSIM_AccessSIMData|               |
   |               |                     *------------------->*               |
   |               |                     |                    |               |
   |               |                     |                    |SIM_ UPDATE_REQ|
   |               |                     |                    *==============>*
   |               |                     |                    |               |
   |               |      CPHS_EXEC      |                    |               |
   |               *<----%ret result%----*                    |               |
   |               |                     |                    |               |
   |     AT_EXCT   |                     |                    |               |
   *<-%ret result%-*                     |                    |               |
   |               |                     |                    |               |
   |               |                     |                    |SIM_UPDATE_CNF |
   |               |                     |                    *<==============*
   |               |                     |                    |               |
   |               |                     | cphs_sim_data_cb() |               |
   |               |                     *<-------------------*               |
   |               |                     |                    |               |
   |               |  cmhCPHS_user_cb()  |                    |               |
   |               | CPHS_VOICE_MAIL_RES |                    |               |
   |               *<--------------------*                    |               |
   |               |                     |                    |               |
   |   rAT_OK( )   |                     |                    |               |
   *<--------------*                     |                    |               |
   |               |                     |                    |               |
```

```
[CPHS Get Waiting Flag Indicator]
AT%CPVWI=2         Queries for all lines: 1 waiting msg for line 1 and line 2,
                   none for fax and data
%CPVWI: 1,257
OK

ATI/MFW             CMH                       CPHS             PSA
   |                 |                          |               |
   | qAT_%CPVWI()    |                          |               |
   *---------------->*                          |               |
   |                 |                          |               |
   |                 | cphs_get_waiting_flag( ) |               |
   |                 *------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |     AT_CMPL     |                          |               |
   *<--%ret result%--*                          |               |
   |                 |                          |               |
```

3.9 DIVERTED CALL INDICATOR and CALL FORWARDING FLAG

3.9.1 ACI/CPHS Module functional interface

3.9.1.1.1 cphs_set_cfu_flag

Prototype:

T_CPHS_RET cphs_set_cfu_flag(UBYTE *cfu_set*, **T_CPHS_LINES** *lines***);**

Parameters:

cfu_set	Call Forwarding Unconditional flag is to be set/erased on the SIM Values: #define CPHS_SET_CFU_FLAG (1) #define CPHS_ERASE_CFU_FLAG (0)
lines	line(s) to which CFU status change applies (in case of ALS): this is a bit mask.

Return:

CPHS_EXEC	operation in progress
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Controls state of Unconditional Call Forwarding Flag on the SIM.

3.9.1.1.2 cphs_get_fwd_flag

Prototype:

T_CPHS_RET cphs_get_fwd_flag(UBYTE **cfu_set,* **T_CPHS_LINES** *line***);**

Parameters:

cfu_set	flags whether Call Forwarding Unconditional is active or not Values: #define CPHS_FLAG_DEACTIVATED (0) #define CPHS_FLAG_ACTIVATED (1) #define CPHS_FLAG_NOT_PRESENT (2) #define CPHS_FLAG_ERROR (3)
line	line to which CFU status applies. (This is NOT a bitmask)

Return:

CPHS_OK	operation successful
CPHS_FAIL	an error occurred (ex: line contains more than one line)
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Retrieves the Call Forwarding flag status stored on the SIM.

3.9.2 AT command

3.9.2.1 % CPCFU

Table 6: %CPCFU parameter command syntax

Command	Possible response(s)
`%CPCFU=<mode>,<line>`	`%CPCFU: <status>[,<line>]` *+CME ERROR: <err>*
`%CPCFU?`	*+CME ERROR: <err>*
`%CPCFU=?`	`%CPCFU:` (list of supported `<mode>s,<line>s)` *+CME ERROR: <err>*

Description

This set command allows setting, clearing, querying of the status of the Call Diverted Flag for one or several lines.

This query command allows querying the current status of the Call Diverted Flag on the SIM for one or several lines.

When querying, intermediate result is %CPCFU: status(,lines) (lines is optional: either Unconditional Call Forwarding has not been activated for any line and intermediate result is %CPCFU: 0, or Unconditional Call Forwarding has been activated for at least one line and intermediate result is %CPCFU= 1,set_lines).

Defined values

`<mode>`: integer type value indicating:

`0`	– Clear Call Diverted Flag
`1`	– Set Call Diverted Flag.

2	– Query Call Diverted Flag status.

`<line>`: bit field type value indicating possible sum of following values:

1	– Line 1
2	– Data
4	– Fax
256	– Line 2

`<status>`: integer type value indicating:

0	– Activated
1	– Deactivated

3.9.3 ACI/MFW functional interface

3.9.3.1 sAT_PercentCPCFU

Prototype:

T_ACI_RETURN sAT_PercentCPCFU(T_ACI_CMD_SRC *srcId***, UBYTE** *cfu_set*, **T_CPHS_LINES** *lines***);**

Parameters:

cfu_set	flag status. Values: #define CPHS_SET_CFU_FLAG (1) #define CPHS_ERASE_CFU_FLAG (0)
lines	line to which change applies: it is a bit mask, so it can be used to change the flag status for several lines at once.

Return:

AT_EXCT	operation in progress
AT_FAIL	an error occurred

Description:

Allows setting or clearing the Call Forwarding Unconditional flag for any line.

3.9.3.2 qAT_PercentCPCFU

Prototype:

T_ACI_RETURN qAT_PercentCPCFU(T_ACI_CMD_SRC *srcId***, UBYTE** **cfu_set*, **T_CPHS_LINES** *line***);**

Parameters:

cfu_set	flags whether Call Forwarding Unconditional is active or not Values: #define CPHS_FLAG_DEACTIVATED (0) #define CPHS_FLAG_ACTIVATED (1) #define CPHS_FLAG_NOT_PRESENT (2) #define CPHS_FLAG_ERROR (3)
line	line to which CFU query applies (this is not a bit mask)

Return:

AT_CMPL	successfully completed

AT_FAIL	an error occurred (ex: line contains more than one line)

Description:

Queries the current call forwarding flag state.

3.9.4 MSC

```
[CPHS Set/Clear CFU Flag]
AT%CPCFU=0          Clears flag for all lines.
OK

ATI/MFW           CMH                  CPHS                  PSA             SIM
   |               |                     |                    |               |
   | sAT_%CPCFU()  |                     |                    |               |
   *-------------->*                     |                    |               |
   |               |                     |                    |               |
   |               |  cphs_set_cfu_flag  |                    |               |
   |               *-------------------->*                    |               |
   |               |                     |                    |               |
   |               |                     |psaSIM_AccessSIMData|               |
   |               |                     *------------------->*               |
   |               |                     |                    |               |
   |               |                     |                    |SIM_ UPDATE_REQ|
   |               |                     |                    *==============>*
   |               |                     |                    |               |
   |               |      CPHS_EXEC      |                    |               |
   |               *<----%ret result%----*                    |               |
   |               |                     |                    |               |
   |     AT_EXCT   |                     |                    |               |
   *<-%ret result%-*                     |                    |               |
   |               |                     |                    |               |
   |               |                     |                    |SIM_UPDATE_CNF |
   |               |                     |                    *<==============*
   |               |                     |                    |               |
   |               |                     | cphs_sim_data_cb() |               |
   |               |                     *<-------------------*               |
   |               |                     |                    |               |
   |               |  cmhCPHS_user_cb()  |                    |               |
   |               |    CPHS_CFU_RES     |                    |               |
   |               *<--------------------*                    |               |
   |               |                     |                    |               |
   |   rAT_OK( )   |                     |                    |               |
   *<--------------*                     |                    |               |
   |               |                     |                    |               |
```

```
[CPHS Get CFU Flag]
AT%CPCFU=2,1        Queries for line 1.
%CPCFU: 0           CFU is inactive for line 1.
OK

ATI/MFW             CMH                       CPHS             PSA
   |                 |                          |               |
   |  qAT_%CPCFU()   |                          |               |
   *---------------->*                          |               |
   |                 |                          |               |
   |                 |   cphs_get_fwd_flag( )   |               |
   |                 *------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |     AT_CMPL     |                          |               |
   *<--%ret result%--*                               |                 |
   |                 |                               |                 |
```

3.10 CURRENT LINE INDICATOR and ALS

3.10.1 ACI/CPHS Module functional interface

3.10.1.1.1 cphs_get_line

Prototype:

T_CPHS_RET cphs_get_line(UBYTE *srcId,* **UBYTE** *call_id***, T_CPHS_LINES** **line,* **CHAR** **line_desc,* **UBYTE** **max_line_desc***);**

Parameters:

call_id	call id of line being queried. This value is the same as the call id described in GSM 02.30 subclause 4.5.5.1 (also see AT+CLCC). If not present (value 255), then current active line is to be returned.
line	contains line of queried call, or current active line.
line_desc	line identification in SIM
max_line_desc	maximal length of line_desc: *IN:* should contain the size of memory allocated for *line_desc*. *OUT:* If memory is not enough, then function returns with CPHS_FAIL and *max_line_desc* will contain the amount of memory needed. Otherwise contains amount of written characters.

Return:

CPHS_OK	execution OK
CPHS_FAIL	error occurred: if *max_buffer is not 0, then reason is memory for *buffer is not enough: check max_buffer for amount of memory needed.
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:
Informs the CPHS module what current line is active for a MT call.

3.10.2 AT command

3.10.2.1 % CPALS

Table 7: %CPALS parameter command syntax

Command	Possible response(s)

`%CPALS=<call_id>`	`%CPALS: <line>,<MSISDN Id>`
	`+CME ERROR: <err>`

`%CPALS?`	`%CPALS: <line>,<MSISDN Id>`
	`+CME ERROR: <err>`

`%CPALS=?`	

Description
This set command allows querying the bearer of a given call. The user has thus the possibility to query the line of a current call.
The query command returns the current active line.

Defined values

`<call_id>`: integer type value indicating:

call id of line being queried. This value is the same as the call id described in GSM 02.30 subclause 4.5.5.1 (also see AT+CLCC).

`<line>`: integer type value indicating possible sum of following values:

1	– Line 1
2	– Data
4	– Fax
256	– Line 2

`<MSISDN id>`: string type value indicating:

This is the MSISDN identification as found on the SIM. If no MSISDN identification can be found on the SIM then the default strings "Line 1", "Data", "Fax" and "Line 2" will be applied.

3.10.3 ACI/MFW functional interface

3.10.3.1 qAT_PercentCPALS

Prototype:

T_ACI_RETURN qAT_PercentCPALS(T_ACI_CMD_SRC *srcId*, **UBYTE** *call_id*, **T_CPHS_LINES** **line*, **CHAR** **line_desc*, **UBYTE** **max_line _desc*);

Parameters:

call_id	call id of line being queried. This value is the same as the call id described in GSM 02.30 subclause 4.5.5.1 (also see AT+CLCC). If not present (value 255), then current active line is to be returned.
line	contains line of queried call, or current active line.
line_desc	line identification in SIM
max_line _desc	maximal length of line_desc: *IN:* should contain the size of memory allocated for *line_desc*. *OUT:* If memory is not enough, then function returns with AT_FAIL and *max_line_desc* will contain the amount of memory needed. Otherwise contains amount of written characters.

Return:

AT_CMPL	successfully completed
AT_FAIL	error occurred: if *max_buffer is not 0, then reason is memory for *buffer is not enough: check max_buffer for amount of memory needed.

Description:

Gives the user the possibility to query at any moment the current active line and for any call its related line (e.g. when receiving a MT call)

3.10.4 MSC

```
[CPHS Alternate Line Service]
RING
AT+CLCC
+CLCC: 1,1,4,0,0   // incoming voice MT call with cId 1
OK
AT%CPALS=1          // queries line of MT call
%CPALS: 2          // call aimed at line 2 MSISDN
OK
AT%CPALS?          // queries current active line
%CPALS: 1          // active line is line 1 MSISDN
OK

ATI/MFW           CMH              CPHS                    PSA               CC
   |               |                  |                     |                 |
   |               |                  |                     | MNCC_SETUP_IND  |
   |               |                  |                     *<================*
   |               |                  |                     |                 |
   |               |         cmhCC_IncomingCall( )          |                 |
   |               *<---------------------------------------*                 |
   |               |                  |                     |                 |
   |  rAT_RING( )  |                  |                     |                 |
   *<--------------*                  |                     |                 |
   |               |                  |                     |                 |

   |               |                  |                     |                 |
   |               |                  |                     |                 |
   | qAT_%CPALS    |                  |                     |                 |
   *-------------->*                  |                     |                 |
   |               |                  |                     |                 |
   |               | cphs_get_line()  |                     |                 |
   |               *----------------->*                     |                 |
   |               |                  |                     |                 |
   |               |     CPHS_OK      |                     |                 |
   |               *<--%ret result%---*                     |                 |
   |               |                  |                     |                 |
   |    AT_CMPL    |                  |                     |                 |
   *<-%ret result%-*                  |                     |                 |
   |               |                  |                     |                 |
   |               |                  |                     |                 |
```

3.11 EMERGENCY CALL 999

This feature is already covered by later GSM recommendation. This means CPHS requirements are already fulfilled in the software current implementation.

3.12 CPHS INFORMATION and CUSTOMER SPECIFIC PROFILE

3.12.1 Data Types

Some macros will be made available to handle CPHS info and CSP:

#define CPHS_CHECK_SST(sst, service, attribute)

Has value TRUE if service has "attribute" status.

Attribute:

ALLOCATED	1
ACTIVATED	2

Services:

CSP	1
SST	2
Mailbox Numbers	3
OpName Shortform	4
Information numbers	5

3.12.2 ACI/CPHS Module functional interface

3.12.2.1.1 cphs_get_cphs_info

Prototype:

T_CPHS_RET cphs_get_cphs_info(UBYTE **phase*, **USHORT** **sst*);

Parameters:

phase	*either CPHS Phase 1: 1 or CPHS Phase 2: 2*
sst	*service table: 2 bytes in Hexadecimal format as stored on SIM*

Return:

CPHS_OK	execution OK
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Retrieves the information concerning CPHS.

3.12.2.1.2 cphs_get_csprof

Prototype:

T_CPHS_RET cphs_get_csprof(CHAR **csp*, **UBYTE** **max_csp_length*);

Parameters:

csp	customer service profile: format is the one described in CPHS4_2.ww6
max_csp_length	maximal length of *csp*: *IN:* should contain the size of memory allocated for *csp*. *OUT:* If memory is not enough, then function returns with CPHS_FAIL and *max_csp_length* will contain the amount of memory needed. Otherwise contains amount of written characters.

Return:

CPHS_OK	execution OK
CPHS_FAIL	error occurred: if *max_csp_length* is not 0, then reason is memory for *csp* is not enough: check *max_csp_length* for amount of memory needed.
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Retrieves the Customer Service Profile.

3.12.3 AT command

3.12.3.1 % CPINF

Table 8: %CPINF parameter command syntax

Command	Possible response(s)

`%CPINF?`	`%CPHS: <phase>,<sst>[,<csp>]`
	`+CME ERROR: <err>`

`%CPINF=?`	

Description
This query command allows query the CPHS phase, the CPHS service table and the customer service profile.

Defined values

`<phase>`: integer type value indicating:

`1` – Phase 1.

`2` – Phase 2.

`<sst>`: 2 bytes Hexadecimal value indicating:

CPHS Service Table: 2 bytes (format HEX) as defined in CPHS4_2.ww6.

`<csp>`: Hexadecimal string indicating:

Customer Service Profile. Format is a hexadecimal string as following: "A1B1A2B2A3B3…" where byte An is the service group code (see CPHS B.4.7.1) and byte Bn the services Byte (see CPHS B.4.7.1)

3.12.4 ACI/MFW functional interface

3.12.4.1 qAT_PercentCPINF

Prototype:

T_ACI_RETURN qAT_PercentCPINF(T_ACI_CMD_SRC *srcId*, **UBYTE** **phase*, **USHORT** **sst*, **CHAR** **csp*, **UBYTE** **max_csp_size*);

Parameters:

phase	either CPHS Phase 1: 1 or CPHS Phase 2: 2
sst	service table: 2 bytes in Hexadecimal format as stored on SIM
csp	customer service profiles
max_cps_size	maximal length of *csp*: *IN:* should contain the size of memory allocated for *csp*. *OUT:* If memory is not enough, then function returns with CPHS_FAIL and *max_csp_length* will contain the amount of memory needed. Otherwise contains amount of written characters.

Return:

AT_CMPL	successfully completed
AT_FAIL	error occurred: if *max_buffer is not 0, then reason is memory for *buffer is not enough: check max_buffer for amount of memory needed.

Description:

Queries the CPHS phase, service table and customer service profiles.

3.12.5 MSC

```
[CPHS Get Information and Customer Service Profile]

Ex: Phase 2, Service Table where “CSP” and “OpName Shortform” are allocated and
activated and the rest is not allocated. CSP contains Service Groups “Call
Offering”, “Call Completion”, “ValueAdded Services” and “Phase Service 2+”.

AT%CPINF?
%CPINF: 2,C300,01F804F0C00109FA
OK


ATI/MFW             CMH                       CPHS             PSA
   |                 |                          |               |
   |                 |                          |               |
   |  qAT_%CPINF()   |                          |               |
   *---------------->*                          |               |
   |                 |                          |               |
   |                 |  cphs_get_cphs_info( )   |               |
   |                 *------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |                 |                          |               |
   |                 |     phs_get_csprof( )    |               |
   |                 *------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |     AT_CMPL     |                          |               |
   *<--%ret result%--*                          |               |
   |                 |                          |               |
```

3.13 READ MAILBOX NUMBERS

3.13.1 Data Types

```
typedef struct
{
 T_CPHS_LINES      line;
 CHAR              number[40];    // 10 BCD in 6F17 and 10 BCD in EXT 1 at most
 UBYTE             toa;           // type of address
 CHAR              alpha_id[22];
} T_CPHS_MB
```

3.13.2 ACI/CPHS Module functional interface

3.13.2.1.1 cphs_read_mb_number

Prototype:

T_CPHS_RET cphs_read_mb_number(UBYTE *rec_id*, **T_CPHS_MB** **mailbox_entry*);

Parameters:

rec_id	Id of queried mailboxnumber
mailbox_entry	contains all information concerning queried mailbox number

Return:

CPHS_OK	execution OK
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Returns mailboxnumber information stored on SIM at a given record.

3.13.2.1.2 cphs_first_free

Prototype:

T_CPHS_RET cphs_first_free(UBYTE ****first_free*);**

Parameters:

first_free	Index of first unused entry in mailbox list

Return:

CPHS_OK	execution OK
CPHS_FAIL	an error occurred: e.g. no information numbers cached

Description:

Returns the index of the first unused entry in the mailbox list.

3.13.3 AT command

3.13.3.1 AT% CPMB

Table 9: %CPMB parameter command syntax

Command	Possible response(s)
`%CPMB=<record_id>`	`%CPMB: <record_id>,<line>,<"number">,<toa>, <"alpha id">` *+CME ERROR: <err>*

`%CPMB?`	`%CPMB: <first>`
	`+CME ERROR: <err>`

`%CPMB=?`	`%CPMB: (max value of <record_id>) on the SIM` *+CME ERROR: <err>*

Description

This set command allows retrieving of the mailbox number by means of its record id on the SIM.
The query command returns the index of the first free entry in the list.
Test command returns the number of the existing record entries of mailbox numbers on the SIM.

Defined values

`<record_id>`: integer type value indicating:

SIM record id of mailbox

`<line>`: integer type value indicating:

`1` – Line 1

2	– Data
4	– Fax
256	– Line 2

`<number>`: string type value indicating:

mailbox number.

`<toa>`: Integer type value indicating:

type of address (coded as in GSM 04.08)

`<alpha id>`: string type value indicating:

alpha identifier related to mailbox.

<first> : Integer type value indicating:

index of the first free entry in the list.

3.13.4 ACI/MFW functional interface

3.13.4.1 qAT_PercentCPMB

Prototype:

T_ACI_RETURN qAT_PercentCPMB(T_ACI_CMD_SRC *srcId*, **UBYTE** *rec_id*, **T_CPHS_LINES** line, **CHAR** ***number, **T_ACI_TOA_TON** ton, **T_ACI_TOA_NPI** npi, **CHAR** *alpha_id, **UBYTE** *first**);**

Parameters:

rec_id	Id of queried mailbox number
line	line to which apply received Mailbox Number
number	mailbox telephone number
ton	type-of-number of mailbox number
npi	numbering-plan-identifier of mailbox number
alpha_id	alpha identifier related to mailbox
first	first unused entry

Return:

AT_CMPL	successfully completed
AT_FAIL	an error occurred

Description:

Deals with both the set and query aspects of the %CPMB command. If *rec_id* is present, indicating a set operation, it will return information relating to the specified mailbox number (via parameters *line*, *number*, *ton*, *npi* and *alpha_id*). If *rec_id* is not present, indicating a query operation, it will return the index of the first unused entry in the list (via parameter *first*).

3.13.5 MSC

```
[CPHS Get Mailbox Number]
AT%CPMB=?
%CPMB: 4
OK

AT%CPMB=2
%CPMB: 2,256,"012341234",129,"MB LINE 2"
OK


ATI/MFW             CMH                          CPHS             PSA
   |                 |                           |                |
   |                 |                           |                |
   |   qAT_%CPMB()   |                           |                |
   *---------------->*                           |                |
   |                 |                           |                |
   |                 |  cphs_read_mb_number( )   |                |
   |                 *------------------------->*                 |
   |                 |                           |                |
   |                 |         CPHS_OK           |                |
   |                 *<------%ret result%-------*                 |
   |                 |                           |                |
   |     AT_CMPL     |                           |                |
   *<--%ret result%--*                           |                |
   |                 |                           |                |
```

3.14 WRITE MAILBOX NUMBERS

3.14.1 Data Types

typedef struct

{

UBYTE data[CPHS_MAX_MB_ALPHA_LEN];

UBYTE len;

} **T_CPHS_PB_TEXT**;

3.14.2 ACI/CPHS Module functional interface

3.14.2.1.1 cphs_write_mb_number

Prototype:

T_CPHS_RET cphs_write_mb_number (UBYTE *srcId*, **UBYTE** *rec_id*, **UBYTE** **tag*, **UBYTE** *tag_len*, **UBYTE** *bcd_len*, **UBYTE** **number*, **UBYTE** *ton_npi*);

Parameters:

source_id	Id of the calling source
rec_id	Id of queried mailbox number
tag	pointer to the alpha indentifier tag
tag_len	length of the alpha identifier tag
bcd_len	length of the bcd encoded number

number	pointer to the bcd encoded number
ton_npi	ton/npi value of number

Return:

CPHS_EXEC	executing
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Writes a mailbox entry with given parameters to SIM.

3.14.2.1.2 cphs_get_mb_parameter

Prototype:

T_CPHS_RET cphs_get_mb_parameter (SHORT **firstIdx*, **SHORT** **lastIdx*, **UBYTE** **nlength*, **UBYTE** **tlength* **);**

Parameters:

firstIdx	pointer to store first index of mailbox phonebook
lastIdx	pointer to store first index of mailbox phonebook
nlength	pointer to the alpha indentifier tag
tlength	length of the alpha identifier tag

Return:

CPHS_OK	execution OK
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Returns the mailbox phonebook related parameters.

3.14.3 AT command

3.14.3.1 AT% CPMBW

Table 10: %CPMBW parameter command syntax

Command	Possible response(s)
%CPMBW=<record_id>[,[<number>],[<toa>],[<text>]]	*+CME ERROR: <err>*
%CPMBW=?	%CPMBW: (list of supported <record_id>s),<nlength>, (list of supported <type>s),<tlength> *+CME ERROR: <err>*

Description

Execution command writes phonebook entry in location number <record_id> of the CPHS mailbox phonebook of the SIM. Entry fields written are phone number `<number>` (in the format `<toa>`) and text `<text>` associated with the number. If <number> is omitted, phonebook entry is deleted. If writing fails in an ME error, `+CME ERROR: <err>` is returned. Refer subclause 9.2 of 3GPP 07.07 for `<err>` values.
Test command returns location range supported by the CPHS mailbox phonebook as a compound value, the maximum length of `<number>` field, supported number formats of the storage, and the maximum length of `<text>` field. If ME is not currently reachable, `+CME ERROR: <err>` is returned. Refer subclause 9.2 of 3GPP 07.07 for `<err>` values.

Defined values

<record_id>: integer type values in the range of location number of mailbox phonebook memory.

Attention:

Since there is no support for the parameter the following convention is used
(refer to the CPHS specification)

<record_id>	line_id	line description
1	1	Line 1
2	2	Data
3	4	Fax
4	256	Line 2

<number>: string type phone number

<toa>: type of address octet in integer format (refer GSM 04.08).

<text>: string type field of maximum length <tlength>

<nlength>: integer type value indicating the maximum length of field <number>

<tlength>: integer type value indicating the maximum length of field <text>

3.14.4 ACI/MFW functional interface

3.14.4.1 sAT_PercentCPMBW

Prototype:

T_ACI_RETURN sAT_PercentCPMBW(T_ACI_CMD_SRC *srcId*, **SHORT** *index*, **CHAR** **number*, **T_ACI_TOA** **type*, **T_CPHS_PB_TEXT** **text*);

Parameters:

srcId	ID of using source
index	index of record
number	pointer to mailbox telephone number
type	pointer to ton/npi of mailbox number in ACI struct
texi	pointer to numbering-plan-identifier of mailbox number

Return:

AT_EXCT	successfully executing
AT_FAIL	an error occurred

Description:

Writes an entry given by index to the CPHS mailbox phonebook SIM storage and updates the respective ME Cache for this entry.

3.14.4.2 tAT_PercentCPMBW

Prototype:

T_ACI_RETURN tAT_PercentCPMBW (T_ACI_CMD_SRC *srcId*, **SHORT** **firstIdx*, **SHORT** **lastIdx*, **UBYTE** **nlength*, **UBYTE** **tlength* **);**

Parameters:

srcId	ID of using source
firstIdx	pointer to store first index of mailbox phonebook

lastIdx	pointer to store first index of mailbox phonebook
nlength	pointer to the alpha indentifier tag
tlength	length of the alpha identifier tag

Return:

AT_OK	successfully executed
AT_FAIL	an error occurred

Description:

Returns the CPHS mailbox phonebook related parameters.

3.14.5 MSC

```
[CPHS Set Mailbox Number]
AT%CPMBW=2,"012341234",129,"MB LINE 2"
OK

AT%CPMBW=?
%CPMBW:
OK
```

```
ATI/MFW             CMH                       CPHS             PSA
   |                 |                          |               |
   |                 |                          |               |
   |   sAT_%CPMBW()  |                          |               |
   *---------------->*                          |               |
   |                 |                          |               |
   |                 |  cphs_write_mb_number( ) |               |
   |                 *------------------------->*               |
   |                 |                          |               |
   |                 |                        SIM_UPDATE_RECORD_REQ
   |                 |                          *-------------->*
   |                 |                          |               |
   |                 |                        SIM_UPDATE_RECORD_CNF
   |                 |                          *<--------------*
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |     AT_CMPL     |                          |               |
   *<--%ret result%--*                          |               |
   |                 |                          |               |
   |                 |                          |               |
   |   tAT_%CPMBW()  |                          |               |
   *---------------->*                          |               |
   |                 |                          |               |
   |                 | cphs_get_mb_parameter( ) |               |
   |                 *------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |     AT_CMPL     |                          |               |
   *<--%ret result%--*                          |               |
   |                 |                          |               |
```

3.15 INFORMATION NUMBERS

3.15.1 Data Types:

```
typedef struct
{
 UBYTE      element_index;          // index of element described in this structure
 UBYTE      index_level;
 CHAR       alpha_tag[16];
 CHAR       number[12];
 UBYTE      type_of_address;
 BOOL       premium_flag;           // should it be charged with premium price?
 BOOL       network_flag;           // is it network specific ?
} T_CPHS_INF_NUM
typedef enum
{
 CPNUMS_MODE_EXPLORE   = 1,   // should apply to a folder to be explored
 CPNUMS_MODE_QUERY     = 2    // queries information related to an entry
} T_CPHS_CPNUMS_MODE
```

3.15.2 ACI/CPHS Module functional interface

3.15.2.1.1 cphs_read_info_nb

Prototype:

T_CPHS_RET cphs_read_info_nb (UBYTE *element_idx,* **T_CPHS_INF_NUM** **inf_num*);

Parameters:

element_idx	Index of element chosen
inf_num	structure containing information related to an element

Return:

CPHS_OK	execution OK
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Returns whole information of a given element.

3.15.2.1.2 cphs_explore_info_nbs

Prototype:

T_CPHS_RET cphs_explore_info_nbs(UBYTE *element_idx,* **UBYTE** **inf_num_indexes,* **UBYTE** **max_elmts*);

Parameters:

element_idx	Index of folder element to be explored
inf_num_list	returns list of indexes of elements contained in folder

max_elmts	limits amount of bytes to be written in inf_num_indexes. If numbers of elements to be written is bigger, then function returns with CPHS_EXEC and amount of elements will be written in max_elmts (so that function caller knows how many memory it has to allocate to retrieve the whole list).

Return:

CPHS_OK	operation successful.
CPHS_EXEC	memory for list was too small. Necessary amount is written in max_elmts.
CPHS_FAIL	an error occurred: e.g. element is no folder.
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

If element_idx has to be a folder. A list of the indexes of the elements contained in the folder will be built and written in inf_num_indexes. Function caller has previously informed of the size of inf_num_indexes by means of max_elmts. If this size is too small then function returns with AT_EXEC and will write the necessary size in max_elmts.

3.15.2.1.3 cphs_info_num_get_max

Prototype:

T_CPHS_RET cphs_info_num_get_max(UBYTE * *max_index***);**

Parameters:

max_index	maximum count of information numbers

Return:

CPHS_OK	execution OK
CPHS_FAIL	an error occurred: e.g. no information numbers cached
CPHS_BUSY	CPHS module is busy
CPHS_NOT_INIT	CPHS module has not been initialised

Description:

Returns the maximum count of available information numbers.

3.15.3 AT command

3.15.3.1 % CPNUMS

Table 11: %CPNUMS parameter command syntax

Command	Possible response(s)
`%CPNUMS=<element_id>,<mode>`	`%CPNUMS: <element_id>,<alpha tag>,<number>, <index_level>,<premium_flag>, <network_flag>` *+CME ERROR: <err>*
`%CPNUMS=?`	list the whole table: `%CPNUMS: <element_id>,<alpha tag>,<number>, <index_level>,<premium_flag>,<network_flag>` *+CME ERROR: <err>*

Description

This set command has 2 modes: exploring and querying. Exploring is only allowed for elements of type folder. Exploring returns the elements belonging to folder having id "element_id" (To be able to start, an extra element ROOT is here defined with element_id 0. It is of type folder). Querying element returns information related to element having id "element_id".

Test command queries a list of all entries to be found on the SIM.

Defined values

`<element_id>`: integer type value indicating:

Each information number entry on the SIM gets a unique Id from the CPHS module.

`0` – default Id for element ROOT (to be used for getting the first elements: exploring ROOT)

`<mode>`: integer type value indicating:

`1` – Exploring element (only if element is a folder): returns elements belonging to folder.

`2` – Querying element: returns information related to element.

`<alpha tag>`: string type value indicating:

alpha tag of element

`<number>`: string type value indicating:

telephone number of element (empty string if element is a folder)

`<index_level>`: integer type value indicating:

index level of element. The elements belonging to ROOT have index level 1.

`<premium_flag>`: integer type value indicating:

`0` – Premium flag is not set for element.

1 – Premium flag is set for element.

`<network_flag>`: integer type value indicating:

`0` – Network specific flag is not set for element.

1 – Network specific flag is set for element.

3.15.4 ACI/MFW functional interface

3.15.4.1 qAT_PercentCPNUMS

Prototype:

T_ACI_RETURN tAT_PercentCPNUMS(T_ACI_CMD_SRC *srcId***);**

Parameters:

void

Return:

AT_CMPL	successfully completed
AT_FAIL	an error occurred

Description:

Queries whole list.

3.15.4.2 sAT_PercentCPNUMS

Prototype:

T_ACI_RETURN sAT_PercentCPNUMS(T_ACI_CMD_SRC *srcId***, UBYTE** *element_index***, T_CPHS_CPNUMS_MODE** *mode***);**

Parameters:

element_index	index of element to be explored
mode	exploring (only for folders) or querying

Return:

AT_CMPL	successfully completed
AT_FAIL	an error occurred

Description:

Explore folder if mode is exploring, otherwise queries element information.

3.15.4.3 rAT_PercentCPNUMS

Prototype:

void rAT_PercentCPNUMS (UBYTE *element_index,* **UBYTE** *index_level,* **CHAR ****alpha_tag,* **CHAR** **number,* **BOOL** *premium_flag,* **BOOL** *network_flag***);**

Parameters:

element_index	index of element
index_level	index level of element (starts with 1)
alpha_tag	alpha tag of element
number	telephone number of element
premium_flag	premium flag for element (should it be charged with premium price?)
network_flag	network flag for element (is it network specific?)

Return:

void

Description:

For n elements contained in an explored folder, this function will be called n times so that ATI/MFW knows what elements are included in a given folder.
If element was queried then function will be called once: it will contain the information related to this phone number.

3.15.5 MSC

```
[CPHS Get Information Numbers]

Ex: Example here is the one presented in CPHS recommendation B 4.6.1 - Figure 2.

AT%CPNUMS=0,1        lists elements from root.
%CPNUMS: 1,"TRAVEL","",1,0,0
%CPNUMS: 8,"WEATHER","",1,0,0
%CPNUMS: 11,"ENTERTAINMENT","",1,0,0
OK

AT%CPNUMS=9,1      (Note: 9 is no Folder: Error)
ERROR

AT%CPNUMS=9,2
%CPNUMS: 9,"North","323",2,0,0
OK



AT%CPNUMS=?
%CPNUMS: 1,"TRAVEL","",1,0,0
%CPNUMS: 2,"TAXIS","",2,0,0
%CPNUMS: 3,"Computercabs","111",3,0,0
%CPNUMS: 4,"Dial-a-cab","132",3,0,0
%CPNUMS: 5,"AIRPORTS","",2,0,0
%CPNUMS: 6,"Heathrow","345",3,0,0
%CPNUMS: 7,"Gatwick","651",3,0,0
%CPNUMS: 8,"WEATHER","",1,0,0
%CPNUMS: 9,"North","323",2,0,0
%CPNUMS: 10,"South","597",2,0,0
%CPNUMS: 11,"ENTERTAINMENT","",1,0,0
%CPNUMS: 12,"Ticketmaster","562",2,0,0
OK

ATI/MFW             CMH                        CPHS             PSA
   |                 |                           |                |
   |                 |                           |                |
   |  tAT_%CPNUMS()  |                           |                |
   *---------------->*                           |                |
   |                 |                           |                |
   |                 |                           |                |
   |                 | cphs_info_num_get_max( )  |                |
   |                 *-------------------------->*                |
   |                 |                           |                |
   |                 |         CPHS_OK           |                |
   |                 *<------%ret result%-------*                |
   |                 |                           |                |
   |                 |                           |                |
   |                 |  cphs_read_info_nb(1)     |                |
   |                 *-------------------------->*                |
   |                 |                           |                |
   |                 |         CPHS_OK           |                |
   |                 *<------%ret result%-------*                |
   |                 |                           |                |
   | rAT_%CPNUMS(1)  |                           |                |
   *<----------------*                           |                |
   ...              ...                         ...              ...
   |                 |                          |               |
   |                 |  cphs_read_info_nb(11)   |               |
   |                 *-------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   | rAT_%CPNUMS(11) |                          |               |
   *<----------------*                          |               |
   |                 |                          |               |
   |                 |  cphs_read_info_nb(12)   |               |
   |                 *-------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   | rAT_%CPNUMS(12) |                          |               |
   *<----------------*                          |               |
   |                 |                          |               |
   |     AT_CMPL     |                          |               |
   *<--%ret result%--*                          |               |
   |                 |                          |               |
```

```
AT%CPNUMS=8,1      lists elements from “WEATHER”
%CPNUMS: 9,”North”,”323”,2,0,0
%CPNUMS: 10,”South”,”597”,2,0,0
OK
```

```
ATI/MFW             CMH                       CPHS            PSA
   |                 |                          |               |
   |                 |                          |               |
   |sAT_%CPNUMS(8,1) |                          |               |
   *---------------->*                          |               |
   |                 |                          |               |
   |                 |                          |               |
   |                 | cphs_explore_info_nbs( ) |               |
   |                 *-------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |                 |                          |               |
   |                 |  cphs_read_info_nb(9)    |               |
   |                 *-------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |  rAT_%CPNUMS(9) |                          |               |
   *<----------------*                          |               |
   |                 |                          |               |
   |                 |  cphs_read_info_nb(10)   |               |
   |                 *-------------------------->*               |
   |                 |                          |               |
   |                 |         CPHS_OK          |               |
   |                 *<------%ret result%-------*               |
   |                 |                          |               |
   |  rAT_%CPNUMS(10)|                          |               |
   *<-----------------*                          |               |
   |                  |                          |               |
   |      AT_CMPL     |                          |               |
   *<--%ret result%--*                           |               |
   |                  |                          |               |
```