



**+CIND – Indicator control  
&  
+CMER – Mobile equipment event reporting**

Project	G23-GSM Protocol Stack
Document Type	Technical Documentation
Title	+CIND – Indicator control & +CMER – Mobile equipment event reporting
Author	Thomas Schott
Creation Date	14.11.2003 09:58
Last Modified	3/8/2015 8:45:00 PM
ID and Version	8462.724.04.001
Status	Planned

Copyright © 2003 Texas Instruments, Inc. All rights reserved.

Texas Instruments Proprietary Information – Strictly Private

# 1 Document Control

© Copyright Texas Instruments, Inc. 2003  
All rights reserved.

Every effort has been made to ensure that the information contained in this document is accurate at the time of printing. However, the software described in this document is subject to continuous development and improvement. Texas Instruments reserves the right to change the specification of the software. Information in this document is subject to change without notice and does not represent a commitment on the part of Texas Instruments. Texas Instruments accepts no liability for any loss or damage arising from the use of any information contained in this document.

The software described in this document is furnished under a license agreement and may be used or copied only in accordance with the terms of the agreement. It is an offence to copy the software in any way except as specifically set out in the agreement. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of Texas Instruments.

## 1.1 Document History

ID	Author	Date	Status
8462.724.04.001	Thomas Schott	13.11.2003	Planned

## 1.2 References, Abbreviations, Terms

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

## 2 Table of Contents

<b>2</b>	<b>TABLE OF CONTENTS</b>	<b>1</b>
<b>3</b>	<b>INTRODUCTION</b>	<b>3</b>
3.1	AT-command syntax	3
3.2	Mobile Equipment event reporting +CMER	4
3.3	Internal ACI behavior (MS Cs)	6
3.3.1	Calling the +CIND from terminal located outside	6
3.3.2	Calling the +CIND? from terminal located outside	6
3.3.3	Calling the +CMER from terminal located outside	7
3.3.4	Calling the +CMER? from terminal located outside	7
3.3.5	Raising of unsolicited message +CIEV (field strength changed)	8
3.3.6	Raising of unsolicited message +CIEV (SIM -full/available)	8
3.3.7	Raising of unsolicited message +CIEV (switch data mode → command mode)	9
<b>4</b>	<b>MODIFICATIONS IN NON-ACI MODULES</b>	<b>10</b>
4.1	Modification in SMS entity	10
4.1.1	Modification in sms_tlf.c	10
4.1.2	Modification in mnsms.doc & mnsms.sap	10
<b>5</b>	<b>MODIFICATIONS IN ACI MODULES</b>	<b>11</b>
5.1	Modifications in ATI	11
5.1.1	Module ati_cmd.c	11
5.1.2	Module ati_mm.c	12
5.1.3	Module ati_ret.c	12
5.1.4	Module aci_cmd.h	13
5.2	Modifications in CMH	14
5.2.1	Module aci_cmh.h	14
5.2.2	Module cmh.h	17
5.2.2.1	Additional command-handler parameter in cmh-shared parameter and type	17
5.2.3	module cmh_f.c	18
5.2.4	Module cmh_mms.c	18
5.2.5	Module cmh_mmq.c	19
5.2.6	Module cmh_mm.h	19
5.2.7	Module cmh_mmr.c	19
5.3	Modifications in PSA	20
5.3.1	Module psa_mm.h	20
<b>6</b>	<b>TEST</b>	<b>21</b>
6.1	Simulation test	21
6.1.1	'SMS full' test at simulation	21
6.1.1.1	ASC610	21
6.1.1.2	ASC611	21

**6.2 Target test 21**

6.2.1	Signal strength test on target-----	21
6.2.2	'SMS full' test on target-----	22

## 3 Introduction

To support the Serie60 product family (SymbionOS) the indicating of the signal quality and the SIM full event have to be support at the terminal interface. This indication will be done via the unsolicited message +CIEV.

This document describes the modifications to be done to provide the +CIND & +CMER commands to control the +CIEV output for the RSSI and SMS full event.

### 3.1 AT-command syntax

Command	Possible response(s)
+CIND=[<ind>[,<ind>]]	+CME ERROR: <err>
+CIND?	+CIND: <ind>[,<ind>]] +CME ERROR: <err>
+CIND= ?	+CIND: ("signal", (0-5)), ("smsfull", (0-1)) +CME ERROR: <err>

#### Description

Set command is used to set the values of ME indicators "signal" and "smsfull". <ind> value 0 means that the indicator is off (or in state which can be identified as "off"-state), 1 means that indicator is on (or in a state which is more substantial than "off"-state), 2 is more substantial than 1, and so on. The "smsfull" indicator is a simple on/off style element, it has values 0 and 1. If ME does not allow setting of indicators or ME is not currently reachable, +CME ERROR: <err> is returned. If parameter is empty field, indicator shall remain in the previous value.

Read command returns the status of both ME indicators. If ME is not currently reachable, +CME ERROR: <err> is returned.

Test command returns pairs, where string value <descr> is a maximum 16 character description of the indicator and compound value is the allowed values for the indicator. If ME is not currently reachable, +CME ERROR: <err> is returned.

#### Defined values

<ind>: integer type value, which shall be in range of corresponding <descr>

<descr> values reserved by the present document and their <ind> ranges:

"signal"            signal quality (0-5)

"smsfull"           a short message memory storage in the MT has become full (1), or memory locations are available (0); i.e. the range is (0-1)

### 3.2 Mobile Equipment event reporting +CMER

Command	Possible response(s)
+CMER=[<mode>[,<ind>[,<bfr>]]]	+CME ERROR: <err>
+CMER?	+CMER: <mode>,<keyp>,<disp>,<ind>,<bfr>
+CMER=?	+CMER: (list of supported <mode>s),(list of supported <keyp>s),(list of supported <disp>s),(list of supported <ind>s),(list of supported <bfr>s)

#### Description

Set command enables or disables sending of unsolicited result codes from ACI to application in the case of indicator state changes.

<mode> controls the processing of unsolicited result codes specified within this command. <bfr> controls the effect on buffered codes when <mode> 1 or 2 is entered. If the protocol stack does not support settings, +CME ERROR: <err> is returned.

Test command returns the modes supported by the TA as compound values.

The buffer in ACI for the unsolicited result codes has the size '1' for storing of one indication.

#### Defined values

<mode>:

- 0 buffer unsolicited result codes in the ACI; if ACI result code buffer is full, the oldest ones will be discarded  
*Interpretation: all indications are buffered, regardless to other settings and ACI → terminal link.*
- 1 discard unsolicited result codes when ACI → terminal link is reserved (e.g. in on-line data mode); otherwise forward them directly to the terminal  
*Interpretation: all indications are forwarded to terminal, except in case of ACI → terminal link reservation: here the indications are discarded.*
- 2 buffer unsolicited result codes in the ACI when ACI → terminal link is reserved (e.g. in on-line data mode) and flush them to the terminal after reservation; otherwise forward them directly to the TE  
*Interpretation: all indications are forwarded to terminal, except in case of ACI → terminal link reservation: here the indications are buffered. If the ACI → terminal link comes back to COMMAND-Mode, the buffered indications are sent.*

#### 3 not supported

<keyp>: **not supported**

<disp>: **not supported**

<ind>:

- 0 no indicator event reporting  
*Interpretation: no indications are sent always in each cases.*
- 1 indicator event reporting using result code +CIEV: <ind>,<value>. <ind> indicates the indicator order number (as specified for +CIND) and <value> is the new value of indicator. Only those indicator events, which are not caused by +CIND shall be indicated by the TA to the TE  
*Interpretation: Only indications which was not selected by the +CIND command are sent.*

- 2 indicator event reporting using result code +CIEV: <ind>,<value>. All indicator events shall be directed from ACI to application

Interpretation: *Only indications which was selected by the +CIND command are sent.*

<bfr>:

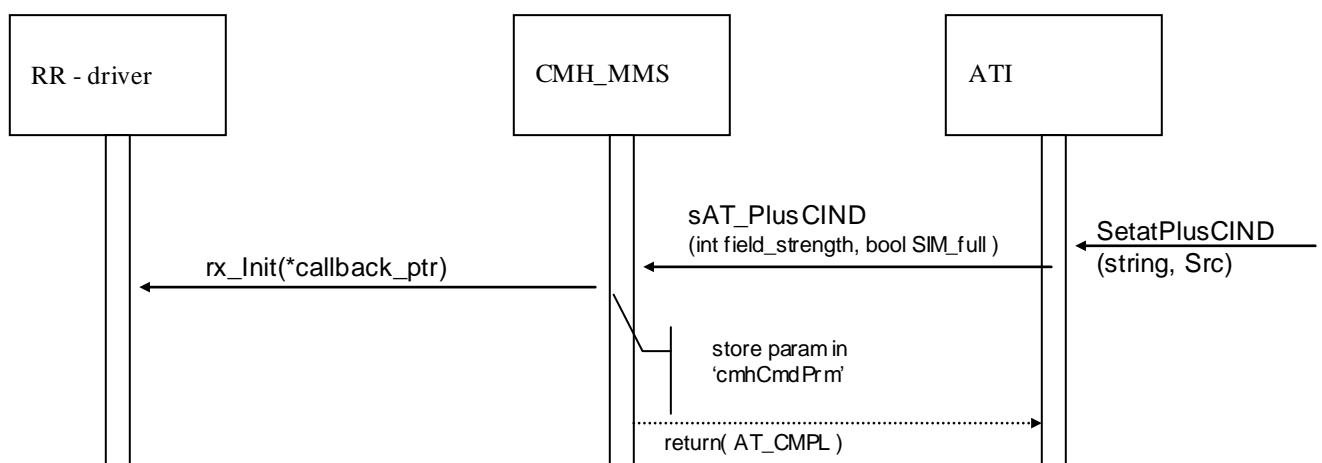
- 0 TA buffer of unsolicited result codes defined within this command is cleared when <mode> 1...3 is entered
- 1 TA buffer of unsolicited result codes defined within this command is flushed to the TE when <mode> 1...2 is entered (OK response shall be given before flushing the codes)

### 3.3 Internal ACI behavior (MSCs)

Only the behavior for the TCS2.1.1. will be described here as follow. For further versions e.g. TCS2.1.x (x>1) a behavior with support of primitives are foreseen.

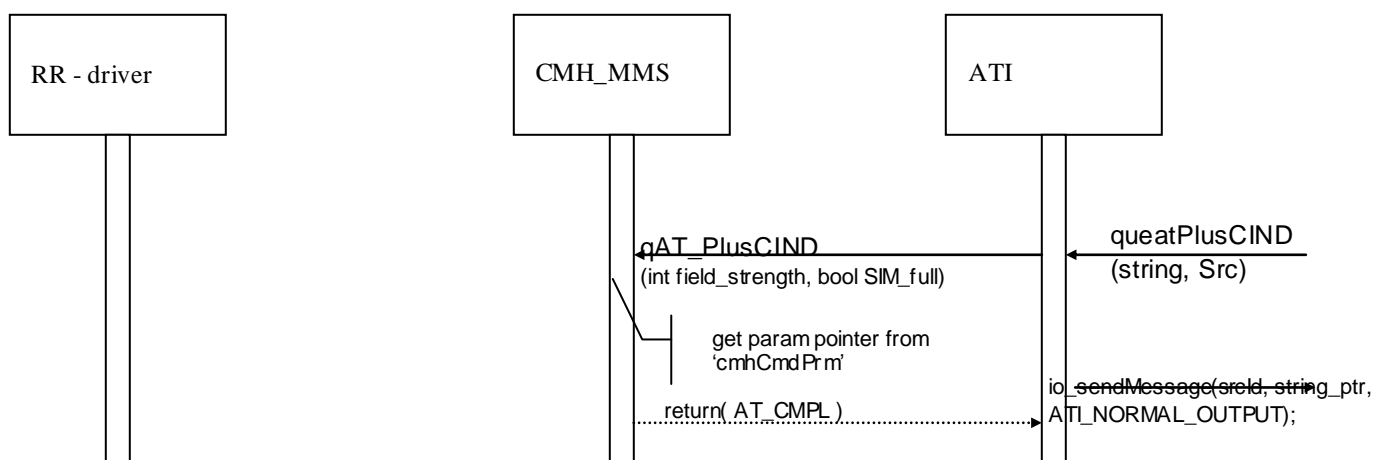
#### 3.3.1 Calling the +CIND from terminal located outside

After enter the +CIND command at AT-Interface, for example by using the terminal program, the parameter will be stored into the shared parameter inside the command handler and a primitive will send to the RR-entity to configure the RR the unsolicited RSSI-indication primitive has to be send to ACI.



#### 3.3.2 Calling the +CIND? from terminal located outside

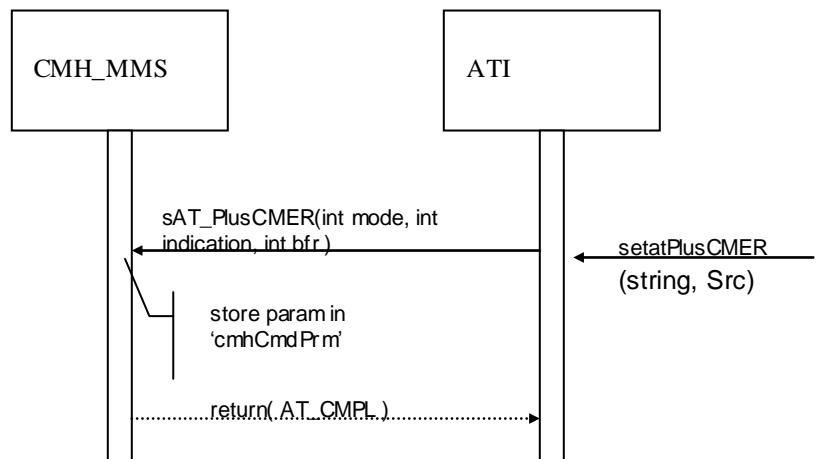
After enter the %CIND? command at AT-Interface, for example by using the terminal program, the configured values for the SIM-full- and field strength indication will be get from command handler and display on the terminal screen.





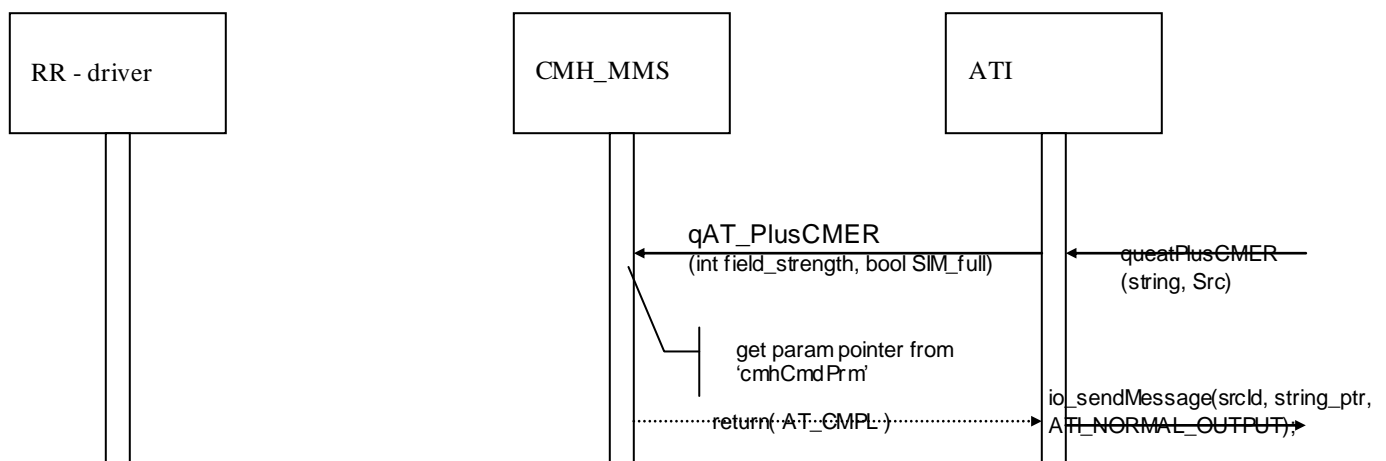
### 3.3.3 Calling the +CMER from terminal located outside

After enter the +CMER command at AT-Interface, for example by using the terminal program, the parameter will be stored into the shared parameter inside the command handler. Depending on these parameters the unsolicited message +CIEV will be raise, buffer or discard.



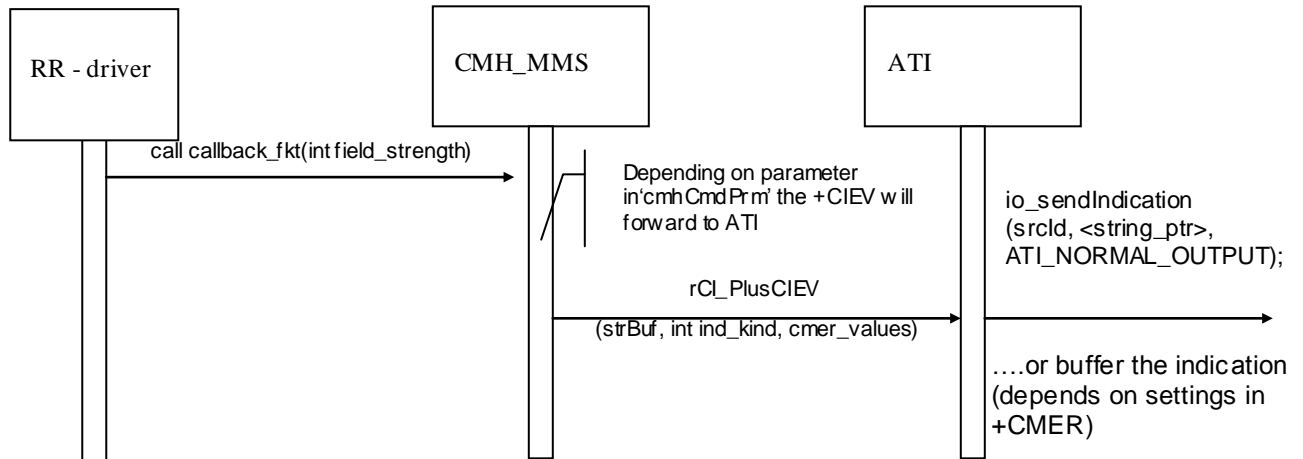
### 3.3.4 Calling the +CMER? from terminal located outside

After enter the %CMER? command at AT-Interface, for example by using the terminal program, the values for the CIEV mode, -indication type and buffering (bfr) will be get from command handler and display on the terminal screen.



### 3.3.5 Raising of unsolicited message +CIEV (field strength changed)

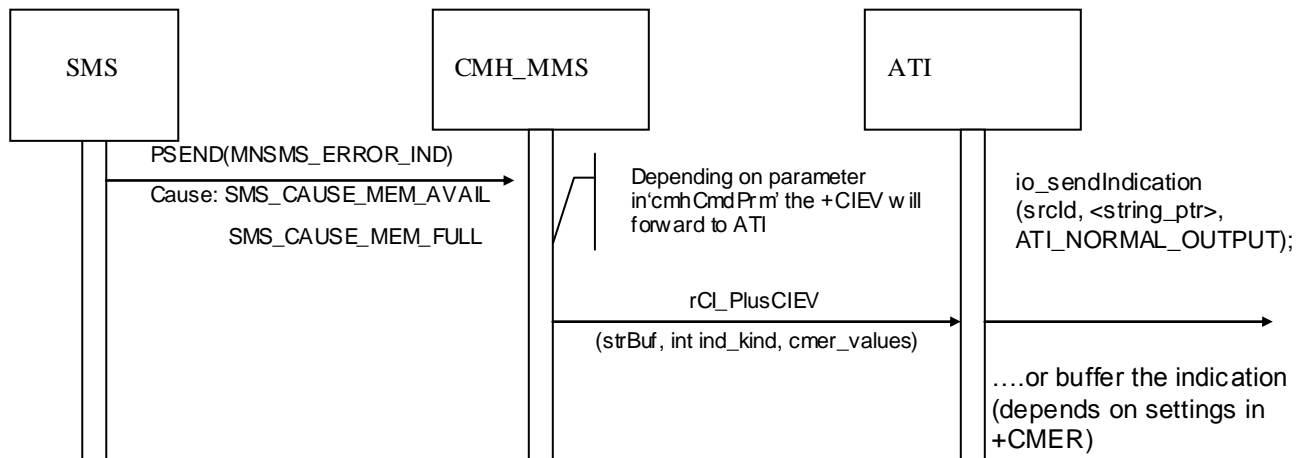
Depending on settings in 'cmhCmdPrm' (+CMER) a string will be send to all connected sources to inform about the field strength change.



### 3.3.6 Raising of unsolicited message +CIEV (SIM-full/available)

Depending on settings in 'cmhCmdPrm' (+CMER) a string will be send to all connected sources to inform about the SIM-full state change.

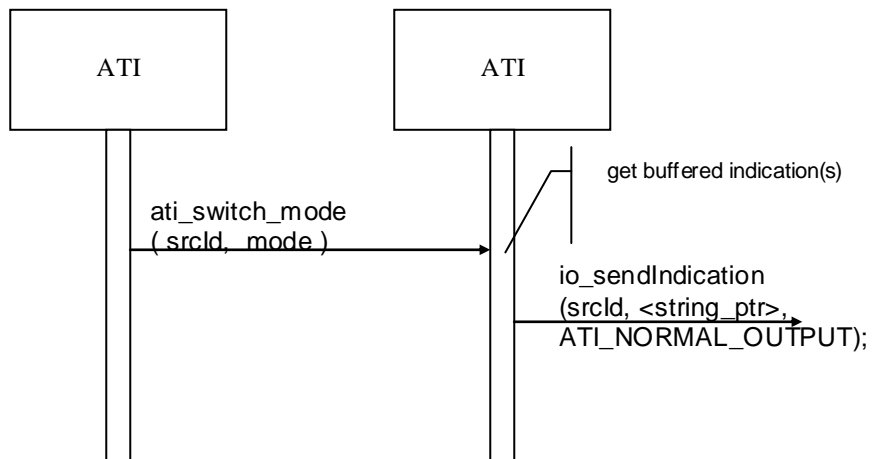
Note, that only the SIM-full state changes will be displayed.



### 3.3.7 Raising of unsolicited message +CIEV (switch data mode → command mode)

If the user has switched from data mode to command mode, a mechanism has to be check if a indication message was buffered. In this case now these indications must be flush now to terminal application.

Note, that only the last reported status will be displayed.



## 4 Modifications in non-ACI modules

### 4.1 Modification in SMS entity

A new cause in the primitive MNSMS\_ERROR\_IND has to be defined. This cause reports if memory locations are available again after the memory storage has become full. The cause for the SMS\_FULL\_IND exists already in the SMS SAP document.

#### 4.1.1 Modification in sms\_tlf.c

An additional call of tl\_sms\_error\_ind shall be necessary, with the new cause parameter whether the SMS storage is become full or empty.

```
/* inform the ACI about SIM-full/avail via SMS error indication */
usErrorParam = avail ? SMS_CAUSE_MEM_AVAIL : SMS_CAUSE_MEM_FULL;
tl_mnsms_error_ind(usErrorParam);
```

Then, the tl\_sms\_error\_ind(error\_value) function will send a SMS error primitive to ACI.

#### 4.1.2 Modification in mnsms.doc & mnsms.sap

A new cause value to indicate that the SMS storage has available space has to be add.  
For this topic also the new mnsms.sap document must be changed also by SAPE.

## 5 Modifications in ACI modules

### 5.1 *Modifications in AT!*

#### 5.1.1 Module ati\_cmd.c

- Additional prototypes

- **EXTERN T\_ATI\_RSLT setatPlusCIND (char \*cl, UBYTE srcId);**
- **EXTERN T\_ATI\_RSLT queatPlusCIND (char \*cl, UBYTE srcId);**
- **EXTERN T\_ATI\_RSLT setatPlusCMER (char \*cl, UBYTE srcId);**
- **EXTERN T\_ATI\_RSLT queatPlusCMER (char \*cl, UBYTE srcId);**

- Additional row in

LOCAL const ATCommand cmds [] = {

:

```
{ "%CIND", AT_CMD_CIND, setatPlusCIND, test_gen, queatPlusCIND,  
"%s: (\signal\", (0-5)), (\smsfull\", (0-1))\"},  
{ \"%CMER\", AT_CMD_CMER, setatPlusCMER, test_gen, queatPlusCMER,  
"%s: (0-2), (0), (0), (0-2), (0,1)\"},  
:  
}
```

- Changes in ati\_switch\_mode()

additional call to **cmd\_flushCIEVBuf**(src\_params->src\_id);

### 5.1.2 Module ati\_mm.c

- Implement new functions
  - T\_ATI\_RSLT **setatPlusCIND**(char \*cl, UBYTE srcId) {  
:  
sAT\_PlusCIND(int signal\_strength, bool sms\_full);  
:  
}
  - T\_ATI\_RSLT **queatPercentCIND** (char \*cl, UBYTE srcId) {  
:  
qAT\_PercentCIND(int field\_strength, bool sms\_full);  
:  
ci\_remTrailCom(string\_ptr, pos);  
io\_sendMessage(srcId, string\_ptr, ATI\_NORMAL\_OUTPUT);  
}
  - T\_ATI\_RSLT **setatPlusCMER**(char \*cl, UBYTE srcId) {  
:  
sAT\_PlusCMER(int mode, int ind, int bfr);  
:  
}
  - T\_ATI\_RSLT **queatPercentCMER** (char \*cl, UBYTE srcId) {  
:  
qAT\_PercentCMER(int mode, int ind, int bfr);  
:  
ci\_remTrailCom(string\_ptr, pos);  
io\_sendMessage(srcId, string\_ptr, ATI\_NORMAL\_OUTPUT);  
}
  - **cmd\_flushCIEVBuf**(src\_params->src\_id)  
→ this function will send the buffered CIEV indications to the terminal when the  
ACI→terminal link is switched back to text mode.
- New global buffer for storing the indications while data mode  
**LOCAL T\_CIEV\_BUFFER atCievBuf[CIEV\_BUF\_SIZE];**

### 5.1.3 Module ati\_ret.c

- Implement new functions to format output string and put it out to terminal  
GLOBAL void **rCI\_PercentCIEV** ( int indication\_kind, int indication\_value,  
T\_ACI\_MM\_IND\_PARAM sIndicationParam )  
{  
UBYTE srcId = srcId\_cb;  
/\* check settings in 'cmhCmdPrm' and depends on, send the indication or not \*/  
sprintf( /\* here format string \*/ );  
io\_sendIndication(srcId, <string\_ptr>, ATI\_NORMAL\_OUTPUT);  
}

#### 5.1.4 Module aci\_cmd.h

- Insert prototype `EXTERN void cmd_flushCievBuf(UBYTE srcId);`
- Insert CIEV buffer

**typedef struct**

```
{  
    UBYTE          uiSrcId;  
    T_ACI_MM_CIND_TYPE tIndication;  
} T_CIEV_BUFFER_VALUES;
```

**typedef struct**

```
{  
    UBYTE          uiLastIndex;  
    T_CIEV_BUFFER_VALUES tBufferValues;  
} T_CIEV_BUFFER;
```

- Insert CIEV buffer size  
**#define CNMI\_BUF\_SIZE 4**

## 5.2 Modifications in CMH

### 5.2.1 Module aci\_cmh.h

- Additional row before AT\_CMD\_MAX → define new AT-command

```
typedef struct {  
:  
AT_CMD_CIND,  
AT_CMD_CMER,  
AT_CMD_MAX  
} T_ACI_AT_CMD
```

- New enum type for +CIND

```
typedef enum  
{  
    CIND_SIGNAL_INDICATOR_INVALID = -1,  
    CIND_SIGNAL_INDICATOR_LVL0,  
    CIND_SIGNAL_INDICATOR_LVL1,  
    CIND_SIGNAL_INDICATOR_LVL2,  
    CIND_SIGNAL_INDICATOR_LVL3,  
    CIND_SIGNAL_INDICATOR_LVL4,  
    CIND_SIGNAL_INDICATOR_LVL5,  
    CIND_SIGNAL_INDICATOR_TYPE_MAX  
} T_ACI_CIND_SIGNAL_TYPE;
```

- New enum type for +CIND

```
typedef enum  
{  
    CIND_SMSFULL_INDICATOR_INVALID = -1,  
    CIND_SMSFULL_INDICATOR_MEMAVAIL,  
    CIND_SMSFULL_INDICATOR_MEMFULL,  
    CIND_SMSFULL_INDICATOR_MAX  
} T_ACI_CIND_SMSFULL_TYPE;
```



- New enum type for +CMER

```
typedef enum
{
    CMER_MODE_INVALID = -1,
    CMER_MODE_0,
    CMER_MODE_1,
    CMER_MODE_2,
    CMER_MODE_TYPE_MAX
} T_ACI_CMER_MODE_TYPE;
```

- New enum type for +CMER

```
typedef enum
{
    CMER_INDICATOR_INVALID = -1,
    CMER_INDICATOR_0,
    CMER_INDICATOR_1,
    CMER_INDICATOR_2,
    CMER_INDICATOR_TYPE_MAX
} T_ACI_CMER_IND_TYPE;
```

- New enum type for +CMER

```
typedef enum
{
    CMER_BFR_INVALID = -1,
    CMER_BFR_0,
    CMER_BFR_1,
    CMER_BFR_TYPE_MAX
} T_ACI_CMER_BFR_TYPE;
```

new prototypes for set and query of AT-command +CIND inside command handler

```
EXTERN T_ACI_RETURN sAT_PlusCIND (int signal_strength, bool sms_full);  
EXTERN T_ACI_RETURN qAT_PlusCIND (int *signal_strength, bool *sms_full);
```

new prototypes for set and query of AT-command +CMER inside command handler

```
EXTERN T_ACI_RETURN sAT_PlusCMER (int mode, int ind, int bfr);  
EXTERN T_ACI_RETURN qAT_PlusCMER (int *mode, int *ind, int *bfr );
```

- new prototypes to format terminal output string and put it out to terminal

```
EXTERN void rAT_PlusCIEV ( void );  
EXTERN void rAT_PlusCIEV ( int ind_kind, int ind_value );  
  
EXTERN void rCI_PlusCIEV ( void );  
EXTERN void rCI_PlusCIEV (int ind_kind, int ind_value );
```

## 5.2.2 Module cmh.h

### 5.2.2.1 Additional command-handler parameter in cmh-shared parameter and type

- new MM command handler type

```
typedef struct
{
    T_ACI_CIND_SIGNAL_TYPE          sCindSignalParam;
    T_ACI_CIND_SMSFULL_TYPE         sCindSmsFullParam;
} T_ACI_MM_CIND_VAL_TYPE;
```

```
typedef struct
{
    T_ACI_CMER_MODE_TYPE            sCmerModeParam;
    T_ACI_CMER_IND_TYPE             sCmerIndParam;
    T_ACI_CMER_BFR_TYPE             sCmerBfrParam;
} T_ACI_MM_CMER_VAL_TYPE;
```

```
typedef struct
{
    T_ACI_MM_CIND_VAL_TYPE          sMmCINDSettings;
    T_ACI_MM_CMER_VAL_TYPE          sMmCMERSettings;
} T_ACI_IND_MODE ;
```

- additional MM command handler parameter

```
typedef struct mmCmdPrm            /* command parameters related to MM */
{
    :
    T_ACI_IND_MODE sIndicationParam;
} T_MM_CMD_PRM;
```

### 5.2.3 module cmh\_f.c

initialize the new command shared parameter with 'invalid'-value

```
for( tSrc=CMD_SRC_LCL; tSrc<CMD_SRC_MAX; tSrc++ )
{ /* initialize the shared params for %CSCN */
  cmhPrm[tSrc].mmCmdPrm... = ...INVALID;
  :
  :
}
```

new row at RAT\_JmpTbl

```
CB_VC( PercentCIEV ), /* RAT_CIEV */
```

### 5.2.4 Module cmh\_mms.c

- New function must be implement for storing the parameter in command-handler shared parameter

```
sAT_PlusCIND (T_ACI_CMD_SRC srcId, int signal_strength, int sms_full );
{
  /* store the data into the 'cmhPrm'-shared parameter */
  rx_Init(rx_Cb); /* initialize callback_Ptr to receive the signal strength */
}
```

```
sAT_PlusCMER (T_ACI_CMD_SRC srcId, int *mode, int *ind_type, int *bfr_type);
{
  /* store the data into the 'cmhPrm'-shared parameter */
}
```

- Changes in '**rx\_Cb()**' to forward the signal quality to ATI via '**cmhMM\_sendCIEV()**'

### 5.2.5 Module cmh\_mmq.c

- New function must be implement to get the parameter pointer in command-handler shared parameter

```
qAT_PlusCIND(T_ACI_CMD_SRC srcId, int *signal_strength, int *sms_full)
{
    /* get the pointer to the 'cmhPrm'-shared parameter */
}
```

```
qAT_PlusCIND(T_ACI_CMD_SRC srcId, int *mode, int *ind_type, int *bfr_type)
{
    /* get the pointer to the 'cmhPrm'-shared parameter */
}
```

### 5.2.6 Module cmh\_mm.h

- New function prototype to forwarding the facilities to ATI against settings in 'cmhCmdPrm'

```
GLOBAL SHORT cmhMM_sendCIEV(int *signal_strength, int *sms_full );
```

### 5.2.7 Module cmh\_mmr.c

- New function must be implement to forwarding the facilities to ATI against settings in 'cmhCmdPrm'

```
GLOBAL SHORT cmhMM_sendCIEV(int *signal_strength, int *sms_full)
{
    /* check settings in 'cmhCmdPrm' and depends on, send the indication or not */
    R_AT( RAT_CIEV, tSrc )( signal_strength, sms_full, sIndicationParam );
    return( 0 );
    :
}
```

## ***5.3 Modifications in PSA***

### **5.3.1 Module psa\_mm,h**

- Receive the SIM-full primitive and forward it to the ACI command handler for further processing.
- Receive the signal strength primitive from RR and forward it to the ACI command handler for further processing.
- Send the RR-config primitive to configure the RR-driver for the signal strength primitive

## 6 Test

### 6.1 Simulation test

A simulation test is only possible for the SMS full part, because for the signal part, a PRIMITIVE does not exist.

#### 6.1.1 'SMS full' test at simulation

The simulation test cases ASC610 and ASC611 tests the correct behavior of the SMS-full indication.

##### 6.1.1.1 ASC610

This test tests the various +CIND and +CMER settings. All possible settings in both commands are tested.

##### 6.1.1.2 ASC611

This test tests the behavior of ACI if an SMS\_ERROR\_INDICATION(...MEM\_FULL/...MEM\_AVAIL) is incomes. In each cases a "+CIEV : smsfull, 0"- or "+CIEV : smsfull, 1"-indication must be appear.

### 6.2 Target test

The target test comprises at least the test of the signal strength. This can be done after coding the changes inside ACI.

Because of needed implementations outside ACI a test of the SIM full indication maybe cannot be possible. It depends on development of other departments, if this feature can be test or not.

#### 6.2.1 Signal strength test on target

This test comprises the test of the behavior of the D-Sample while various +CIND & +CMER settings and signal levels. The various signal levels will be simulate via shielding of the antenna.

These tests take the behavior of the +CIEV output after switching the +CIEV-message on via +CMER and/or +CIND at commands:

Signal strength test 1 (full setting of +CIND → all field strength changes will be displayed)

Type AT+CIND=5 in terminal

shielding the antenna

wait for the corresponding +CIEV message (e.g. +CIEV: signal, 4)

the +CIEV-message will appears every once if the field strength has been changed

Signal strength test 2 (full setting of +CMER → all field strength changes will be displayed)

Type AT+CMER=0,0,0,2 in terminal

shielding the antenna

wait for the corresponding +CIEV message (e.g. +CIEV: signal, 4)

the +CIEV-message will appear every once if the field strength has been changed

#### Signal strength test 3

(partial setting of +CIND → the +CIEV appears only then if the signal strength value has been changed lower than set signal strength while +CIND command )

Type AT+CIND=5 in terminal

shielding the antenna

wait for the corresponding +CIEV message (e.g. +CIEV: signal, 4)

- the +CIEV-message will appear every once if the field strength has been changed

#### 2) Signal strength test 4

(switch of the +CIND output via clear all values at +CMER and/or +CIND command)

type AT+CIND=0,0 in terminal

type AT+CIND=0,0,0,0 in terminal

shielding the antenna

wait for the corresponding +CIEV message (e.g. +CIEV: signal, 4)

- a +CIEV-message must not appear !!

#### 3) Signal strength test 5

(do test 1-4 in combine with %CSQ is switched on)

- type AT+%CSQ=1 in terminal (additional %CSQ values will appear every then if the signal strength value has been changed)
- do test 1-4

#### 4) Signal strength test 6

(test buffering of +CIEV indications in case of UART data mode)

- AT+CFUN=1
- AT+CPIN="xyzw"
- AT+COPS=0
- AT+CIND=5
- a +CIEV-message will appear every once if the field strength has been changed
- AT+CBST=7,0,1
- ATD0303822699 (dial the 'chat noir' mailbox)
- a +CIEV-message must not appear !! Every once, if the +CIEV indication should be appears, this will be buffer (check this via using PCO2).
- type (fast) '+++ ' at the terminal -> the data mode will be switched off
- the last four buffered +CIND indications will be display

### 6.2.2 'SMS full' test on target

The SMS full test to show the "+CIEV : smsfull, 0"- or "+CIEV : smsfull, 1"-indication must also be done at target. In order to do this, some SMS are received until the "+CIEV : smsfull, 1"-indication is received. After deletion of some SMS from the SMS phone storage, the "+CIEV : smsfull, 0"-indication must be appear.

*Prerequisite:* +CIND=,1 and +CMER=1,,,2,1