Technical Document

LLD: FIND FIRST LOCATION IN CURRENT PHONEBOOK

Document Number:	20_04_04_02454
Version:	0.4
Status:	Draft
Approval Authority:	
Creation Date:	2005-May-11
Last changed:	2005-July-25 by Andrew Spruce
File Name:	lld_aci_cpbs_cpmb_xt6.doc

Copyright © 2003 Texas Instruments
Texas Instruments Proprietary Information – Internal Data
NET3.DOT

Important Notice

Texas Instruments Incorporated and/or its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products, software and services at any time and to discontinue any product, software or service without notice. Customers should obtain the latest relevant information during product design and before placing orders and should verify that such information is current and complete.

All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment. TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI products, software and/or services. To minimize the risks associated with customer products and applications, customers should provide adequate design, testing and operating safeguards.

Any access to and/or use of TI software described in this document is subject to Customers entering into formal license agreements and payment of associated license fees. TI software may solely be used and/or copied subject to and strictly in accordance with all the terms of such license agreements.

Customer acknowledges and agrees that TI products and/or software may be based on or implement industry recognized standards and that certain third parties may claim intellectual property rights therein. The supply of products and/or the licensing of software does not convey a license from TI to any third party intellectual property rights and TI expressly disclaims liability for infringement of third party intellectual property rights.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products, software or services are used.

Information published by TI regarding third–party products, software or services does not constitute a license from TI to use such products, software or services or a warranty, endorsement thereof or statement regarding their availability. Use of such information, products, software or services may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

No part of this document may be reproduced or transmitted in any form or by any means, electronically or mechanically, including photocopying and recording, for any purpose without the express written permission of TI.

Change History

Date	Changed by	Approved by	Version	Status	Notes
2005-May-11	Andrew Spruce		0.1	Draft	1
2005-May-16	Andrew Spruce		0.2	Draft	2
2005-May-25	Andrew Spruce		0.3	Draft	3
2005-Jul-25	Andrew Spruce		0.4	Draft	4

Notes:

1. Initial version.
2. Updated following internal review.
3. Added %CPMB?
4. File renamed, minor corrections made.

Table of Contents

1 Introduction

During the design phase of the Atlas (XT6) project, it has been determined that enhancements to the ACI are required in order to support ACT-PBListFirstFreeItemIdSearch and EVT-PBListFirstFreeItemIdSearch from the Alcatel EXP Phonebook object. This document details the necessary work.

A new command %CPBS?, an extension to +CPBS? will be implemented. This will handle all ACI-supported phonebooks except for "MBN" and "SIM_SST".

For "MBN" (CPHS Mailbox Numbers), a new command %CPMB? will be implemented.

As "SIM_SST" is not writeable, there is no need to know its first free entry. Consequently the feature will not be implemented for this list.

2 PHB

A new function, pb_first_free( ), which will determine the first free location in the specified phonebook, will be added in **phb.c** (the corresponding prototype in **phb.h**). The mechanism used will be based on the one that +CPBW uses, which is in pb_add_record( ).

```
T_PHB_RETURN pb_first_free(
  UBYTE type,
  SHORT *first_free)
{
  SHORT i,bit;
  UBYTE max_bitmap;
  UBYTE pos;

  TRACE_FUNCTION ("pb_first_free()");

  if (first_free==NULL)
    return(PHB_FAIL);

  switch (type)
  {
    case SDN:
    case ADN:
    case FDN:
    case BDN:
    case UPN:
    case ECC:
      switch (type)
      {
        case ADN:   max_bitmap=MAX_ADN_BITMAP;    break;
        case FDN:   max_bitmap=MAX_FDN_BITMAP;    break;
        case BDN:   max_bitmap=MAX_BDN_BITMAP;    break;
        case UPN:   max_bitmap=MAX_UPN_BITMAP;    break;
        case SDN:   max_bitmap=MAX_SDN_BITMAP;    break;
        case ECC:   max_bitmap=MAX_ECC_BITMAP;    break;

        default:
          return(PHB_FAIL);
      }

      bit = 0;
      for (i=0; i<max_bitmap; i++)
      {
        pos = 0;
        while ((phb_ctb[type].rcd_bitmap[i] & (1<<pos)))
        {
          bit++;
          pos++;
        }

        if ((bit%8) || !pos)
        {
          if (bit>=phb_ctb[type].max_rcd)
          {
            *first_free=0;
            return(PHB_FULL);
          }

          *first_free=bit+1;
          return(PHB_OK);
        }
      }

      *first_free=0;
      return(PHB_FULL);
```

```
    case LDN:
    case LMN:
    case LRN:
      /*
      *   It is not possible to specify an index when writing to these
      *   phonebooks. Whenever a number is added, it automatically goes
      *   in the first entry of the list, hence it could be said that
      *   the first free entry is always 1.
      */
      *first_free=1;
      return(PHB_OK);

    default:
      break;
  }

  return(PHB_FAIL);
}
```

Note that this does not include the combined ADN/FDN phonebook, a result for this would be meaningless, particularly as it cannot be written to using +CPBW.

3 CPHS

A new function, cphs_first_free( ), which will determine the first free location in the mailbox list, will be added in **cphs.c** (the corresponding prototype in **cphs.h**).

```
GLOBAL T_CPHS_RET cphs_first_free(
  UBYTE *first_free)
{
  UBYTE rec_id;

  if(cphs_cached_params==NULL)
  {
    TRACE_ERROR("cphs_cached_params==NULL");
    return(CPHS FAIL);
  }

  if (cphs cached params->mb numbers==NULL)
  {
    TRACE_ERROR("cphs_cached_params->mb_numbers==NULL");
    return(CPHS_FAIL);
  }

  for (rec_id=0;rec_id<cphs_cached_params->max_mb_numbers;rec_id++)
  {
    if (cphs_cached_params->mb_numbers[rec_id].number[0]==0)
    {
      *first_free=(UBYTE)(rec_id+1);
      return(CPHS_OK);
    }
  }

  /*
  *   List is full, indicate this with 0.
  */
  *first_free=0;
  return(CPHS_OK);
}
```

4 ACI

In **cmh_phbq.c**, the function cmh_QueryCPBS( ) will be added. This will just be a copy of qAT_PlusCPBS( ), except in that it will also call pb_first_free( ) in order to determine the first free location.

```
LOCAL T_ACI_RETURN cmh_QueryCPBS ( T_ACI_CMD_SRC  srcId,
                                   T_ACI_PB_STOR* storage,
                                   SHORT*         used,
                                   SHORT*         total,
                                   SHORT*         first )
{
.
.
.

  if (first!=NULL)
    *first=0;

  if ( pPHBCmdPrm -> cmhStor NEQ PB_STOR_NotPresent )
  {

.
.
.

    /*
    *   Now try and determine the first free location in the
    *   currently selected phonebook, but only if the output pointer
    *   is valid.
    */
    if (first!=NULL)
    {
      if (ret==PHB_OK)
      {
        ret=pb_first_free(pPHBCmdPrm->phbStor,first);

        switch(ret)
        {
          default:
          case PHB_FAIL:
            ACI_ERR_DESC(ACI_ERR_CLASS_Cme,CME_ERR_Unknown);
            return(AT_FAIL);

          case PHB_FULL:
          case PHB_OK:
            break;
        }
      }
    }
  }

  return (AT_CMPL);
}
```

The existing qAT_PlusCPBS( ) will be modified to call this new function :

```
GLOBAL T_ACI_RETURN qAT_PlusCPBS ( T_ACI_CMD_SRC  srcId,
                                   T_ACI_PB_STOR* storage,
                                   SHORT*         used,
                                   SHORT*         total )
{
  TRACE_FUNCTION ("cmh_QueryCPBS ()");
```

```
  return(cmh_QueryCPBS(srcId,storage,used,total,NULL));
}
```

Finally, the very similar qAT_PercentCPBS( ) will be implemented (the corresponding prototype will be in **aci_cmh.h**) :

```
GLOBAL T_ACI_RETURN qAT_PercentCPBS ( T_ACI_CMD_SRC  srcId,
                                      T_ACI_PB_STOR* storage,
                                      SHORT*         used,
                                      SHORT*         total,
                                      SHORT*         first )
{
  TRACE_FUNCTION ("qAT_PercentCPBS ()");

  return(cmh_QueryCPBS(srcId,storage,used,total,first));
}
```

Modifications will be made to qAT_PercentCPMB( ), including an additional argument to allow it to return the first free entry. Note that this function is now used for both set and query. The function is in **cmh_cphs.c**, the corresponding prototype in **aci_cphs.h**.

```
GLOBAL 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)
{
.
.
.

  /*
  *   The presence of rec_id determines whether or not this is a
  *   query.
  */
  if (rec_id==(UBYTE)ACI_NumParmNotPresent)
  {
    if (first==NULL)
    {
      ACI_ERR_DESC(ACI_ERR_CLASS_Cme,CME_ERR_Unknown);
      return(AT_FAIL);
    }

    /*
    *   Determine the first free location.
    */
    if (cphs_first_free(first)!=CPHS_OK)
    {
      ACI_ERR_DESC(ACI_ERR_CLASS_Cme,CME_ERR_Unknown);
      return(AT_FAIL);
    }

    return(AT_CMPL);
  }

  cphs ret = cphs read mb number(rec id, &mb number);

.
.
.
```

```
}
```

5 ATI

AT_CMD_P_CPBS will be added to T_ACI_AT_CMD in **aci_cmh.h**. The new command will be added to the 'cmds' table in **ati_cmd.c** :

```
LOCAL const ATCommand cmds [] =
{
.
.
.
    {"%CPBS", AT_CMD_P_CPBS, 0 , test_gen, queatPercentCPBS , 0},
    {NULL,0,0,0,0,0}
};
```

The %CPMB entry in this table will be modified to add the query :

```
LOCAL const ATCommand cmds [] =
{
.
.
.
CPHS_CMD("%CPMB",AT_CMD_CPMB,setatPercentCPMB,test_gen,queatPercentCPMB,"%s: 4")
.
.
.
};
```

In **ati_phb.c**, queatPercentCPBS( ) will be added. This will be based on queatPlusCPBS( ) but with the additional parameter added.

```
GLOBAL T_ATI_RSLT queatPercentCPBS(char *cl, UBYTE srcId)
{
  char *me="%CPBS: ";
  T_ACI_RETURN    ret;
  T_ACI_PB_STOR   stor;
  SHORT           used;
  SHORT           total;
  SHORT           first;
  SHORT           i;

  TRACE_FUNCTION("queatPercentCPBS()");

  ret=qAT_PercentCPBS(srcId,&stor,&used,&total,&first);

  if (ret==AT_CMPL)
  {
    for(i=0;phb_mem_names[i].name NEQ NULL;i++)
    {
      if (phb_mem_names[i].stor EQ stor)
      {
        sprintf(g_sa,"%s\"%s\",%d,%d,%d",
          me,phb_mem_names[i].name,used,total,first);
        io sendMessage(srcId, g sa, ATI NORMAL OUTPUT);
        break;
      }
    }
  }
  else if (ret==AT_FAIL)
  {
    cmdCmeError(CME_ERR_Unknown);
  }
  return (map_aci_2_ati_rslt(ret));
```

```
}
```

In **ati_cphs.c**, queatPercentCPMB( ) will be added.

```
GLOBAL T_ATI_RSLT queatPercentCPMB(char *cl, UBYTE srcId)
{
  char *me="%CPMB: ";
  T_ACI_RETURN    ret;
  UBYTE           first;

  TRACE_FUNCTION("queatPercentCPMB()");

  ret=qAT_PercentCPMB(
    srcId,
    (UBYTE)ACI_NumParmNotPresent,
    NULL,NULL,NULL,NULL,NULL,&first);

  if (ret==AT_CMPL)
  {
    sprintf(g_sa,"%s%d",me,first);
    io_sendMessage(srcId, g_sa, ATI_NORMAL_OUTPUT);
  }
  else if (ret==AT_FAIL)
  {
    cmdCmeError(CME_ERR_Unknown);
  }
  return (map_aci_2_ati_rslt(ret));
}
```

6 BAT

The script file **cmd_list.log** will be modified to include the new commands. The parameter 'storage' will then exist in both +CPBS and %CPBS, so it should be relocated to the 'shared' area :

```
-storage : STORAGE : U8 : Storage area
0 : FD : SIM fixdialling phone book
1 : DC : SIM last-dialling phone book (LD)
2 : EN : Emergency call numbers (ED)
3 : AD : Abbreviated dialling numbers
4 : BD : Barred dialling numbers
5 : RC : Last received numbers (LR)
6 : SD : Service dialling numbers
7 : MC : Last missed numbers (LM)
8 : AF : Combination of fixed and abbreviated dialling phonebook (AD + FD)
9 : UD : User person number
```

The commands themselves will then be described as follows :

```
{
+CPBS : Select Phonebook Memory Storage
[cmd
-storage : REF@shared : man
]
[query rsp
-storage : man
-used  : USED  : U8 : opt : Number of used locations in selected memory
-total : TOTAL : U8 : opt : Total number of locations in selected memory
]
}

{
%CPBS : Select Phonebook Memory Storage
[query rsp
-storage : REF@shared : man
-used  : USED  : U8 : opt : Number of used locations in selected memory
-total : TOTAL : U8 : opt : Total number of locations in selected memory
-first : FIRST : U8 : opt : First free location in selected memory
]
}

{
%CPMB : CPHS Mailbox Numbers
.
.
.
[query rsp
-first : FIRST : U8 : man : First free location
]
}
```

This will result in the following structures being added to **p_bat.h**,

```
typedef struct
{
  T_BAT_storage storage; /*<  0: 4> (enum=32bit)<->T_BAT_storage Storage area */
  S16           used;    /*<  4: 2> Number of used locations in selected memory */
  S16           total;   /*<  6: 2> Total number of locations in selected memory */
  S16           first;   /*<  8: 2> First free location in selected memory */
  U8            _align0; /*< 10: 1> alignment */
  U8            _align1; /*< 11: 1> alignment */
} T_BAT_res_que_percent_cpbs;

typedef struct
{
  U8            first;   /*<  0: 1> First free location */
  U8            _align0; /*<  1: 1> alignment */
  U8            _align1; /*<  2: 1> alignment */
  U8            _align2; /*<  3: 1> alignment */
} T_BAT_res_que_percent_cpmb;
```

and in **p_bat_val.h**, definitions for BAT_CMD_QUE_PERCENT_CPBS, BAT_RES_QUE_PERCENT_CPBS, BAT_CMD_QUE_PERCENT_CPMB and BAT_RES_QUE_PERCENT_CPMB will be created.

In **aci_bat.c**, the new command will be inserted into the appropriate tables :

```
LOCAL const BATCommand bat_cmds_without_params [] =
{
.
.
.
  { /* BAT_CMD_QUE_PERCENT_CPBS       */ qBAT_PercentCPBS    },
.
.
.
  { /* BAT_CMD_QUE_PERCENT_CPMB       */ qBAT_PercentCPMB    },



static T_map_response_2_size response_2_size[] =
{
.
.
.
  {/* BAT_RES_QUE_PERCENT_CPBS      */ sizeof(T_BAT_res_que_percent_cpbs)},
.
.
.
  {/* BAT_RES_QUE_PERCENT_CPMB      */ sizeof(T_BAT_res_que_percent_cpmb)},
```

In **aci_bat_phb.c**, qBAT_PercentCPBS( ) will be implemented (the corresponding prototype will be in **aci_bat_cmh.h**) :

```
GLOBAL T_ACI_BAT_RSLT qBAT_PercentCPBS (T_ACI_DTI_PRC_PSI *src_infos_psi,
                                        T_BAT_cmd_send    *cmd)
{
  T_ACI_BAT_RSLT ret;
  T_ACI_PB_STOR stor;
  SHORT used;
  SHORT total;
  SHORT first;

  TRACE_FUNCTION ("qBAT_PercentCPBS()");

  /*
  *   Call the corresponding qAT function. T ACI BAT RSLT is
  *   assumed to be equivalent to T_ACI_RESULT.
  */
  ret=(T_ACI_BAT_RSLT)qAT_PercentCPBS(
    src_infos_psi->srcId,&stor,&used,&total,&first);

  /*
  *   If the query completes, send the response now.
  */
  if (ret==ACI_BAT_CMPL)
  {
    T_BAT_cmd_response resp;
    T_BAT_res_que_percent_cpbs cpbs_data;

    resp.ctrl_response=BAT_RES_QUE_PERCENT_CPBS;
    resp.response.ptr_que_percent_cpbs=&cpbs_data;

    /*
    *   This relies on T_BAT_storage being identical to
    *   T_ACI_PB_STOR.
    */
    cpbs_data.storage=(T_BAT_storage)stor;

    cpbs_data.total=(S16)total;
    cpbs_data.used=(S16)used;
    cpbs_data.first=(S16)first;

    aci_bat_send(src_infos_psi,&resp);
  }

  return(ret);
}
```

In **aci_bat_cphs.c**, qBAT_PercentCPMB( ) will be implemented (the corresponding prototype will be in **aci_bat_cmh.h**) :

```
GLOBAL T_ACI_BAT_RSLT qBAT_PercentCPMB (T_ACI_DTI_PRC_PSI *src_infos_psi,
                                        T_BAT_cmd_send    *cmd)
{
  T_ACI_BAT_RSLT ret;
  UBYTE first;

  TRACE_FUNCTION ("qBAT_PercentCPMB()");

  /*
  *   Call the corresponding qAT function. T_ACI_BAT_RSLT is
  *   assumed to be equivalent to T_ACI_RESULT.
  */
  ret=(T_ACI_BAT_RSLT)qAT_PercentCPMB(
    src_infos_psi->srcId,
    (UBYTE)ACI_NumParmNotPresent,NULL,NULL,NULL,NULL,NULL,&first);

  /*
  *   If the query completes, send the response now.
  */
  if (ret==ACI_BAT_CMPL)
  {
    T_BAT_cmd_response resp;
    T_BAT_res_que_percent_cpmb cpmb_data;

    resp.ctrl_response=BAT_RES_QUE_PERCENT_CPMB;
    resp.response.ptr_que_percent_cpmb=&cpmb_data;

    cpmb_data.first=(U8)first;

    aci_bat_send(src_infos_psi,&resp);
  }

  return(ret);
}
```

7 Testing

Tapcaller test cases will be written, for example :

```
T_CASE ACIPHB091()
{
  BEGIN_CASE ("First free location")
  {
    initialize_phonebook_aciphb061();

    SEND(aci_cmd_req("AT+CPBS='FD'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'FD',0,0,0"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='DC'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'DC',3,10,1"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='EN'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'EN',3,4,4"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='MT'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'MT',1,1,0"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='BD'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'BD',0,0,0"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='RC'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'RC',0,10,1"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='SD'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'SD',0,0,0"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='MC'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'MC',0,10,1"));
    AWAIT(aci_cmd_ind("OK"));

    SEND(aci_cmd_req("AT+CPBS='AF'"));
    AWAIT(aci_cmd_ind("OK"));
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("ERROR"));

    SEND(aci_cmd_req("AT+CPBS='ON'"));
    AWAIT(aci_cmd_ind("OK"));
```

```
    SEND(aci_cmd_req("AT%CPBS?"));
    AWAIT(aci_cmd_ind("%CPBS: 'ON',0,0,0"));
    AWAIT(aci_cmd_ind("OK"));
  }
}
```

It is anticipated that the BAT commands will be tested the same way. In order to achieve this it will be necessary to modify the corresponding ATI routines to generate and process BAT messages. As this area is currently under development, and the mechanism may change significantly before the work described in this document is implemented, this will not be detailed at present.

8 Documentation

This section describes changes that will be made to existing documents.

8.1 AT Command Interface Description (8415.052)

8.1.1 %CPBS: Select Phonebook Memory Storage

Command	Possible Response(s)
%CPBS?	%CPBS: <storage>[,<used>,<total>,<first>] +CME ERROR:<err>

Description

This command is an extended version of +CPBS. It is used to obtain information about the currently selected phonebook.

Defined Values

<storage> : Currently selected phonebook memory storage area.

"FD" SIM fixdialling phone book
"DC" SIM last-dialling phone book
"EN" Emergency call numbers
"MT" Abbreviated dialling numbers
"BD" Barred dialling numbers
"RC" Last received numbers
"SD" Service dialling numbers
"MC" Last missed numbers
"AF" Combination of fixed and abbreviated dialling phonebook
"ON" User person number

<used> : Integer type value indicating the number of used locations in selected memory.

<total> : Integer type value indicating the total number of locations in selected memory.

<first> : Integer type value indicating the index of the first free location in selected memory.

8.1.2 %CPMB: CPHS Mailbox Numbers

Command	Possible Response(s)
%CPMB?	%CPBS: <first> +CME ERROR: <err>

Description

The query command returns the index of the first free entry in the list.

Defined Values

<first> : Integer type value indicating the index of the first free location in the list.

8.2 ACI Functional Interface Description (8411.802)

8.2.1 qAT_PercentCPBS( ) - Select Phonebook Memory Storage

Command Reference :

8415.052

Function Definition :

```
T_ACI_RETURN qAT_PercentCPBS (
        T_ACI_CMD_SRC srcId,
        T_ACI_PB_STOR *storage,
        SHORT *used,
        SHORT *total,
        SHORT *first);
```

Parameters:

name	buffer size	comment	
srcId	---	command source identifier	IN
storage	---	selected phonebook memory storage	OUT
used	---	used entries	OUT
total	---	total entries	OUT
first	---	first unused entry	OUT

Return:

symbolic constant	comment
AT_CMPL	successfully completed
AT_FAIL	failed

8.2.2 qAT_PercentCPMB( ) - CPHS Mailbox Numbers

Command Reference :

8415.052

Function Definition :

```
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:

name	buffer size	comment	
srcId	---	command source identifier	IN
rec_id	---	SIM record ID of mailbox	IN
line	---	line	OUT
number	---	mailbox number	OUT
ton	---	type of address	OUT
npi	---	numbering plan	OUT
alpha_id	CPHS_MAX_MB_ALPHA_LEN	associated text string	OUT
first	---	first unused entry	OUT

Return:

symbolic constant	comment
AT_CMPL	successfully completed
AT_FAIL	failed

Function Group:

CPHS

Description:

The qAT_PercentCPMB( ) function 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 about the specified mailbox number (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 (parameter *first*).

8.3 Atlas Project Phonebook Object Mapping (20_04_02_01451)

Mappings for ACT-PBListFirstFreeItemSearch / EVT-PBListFirstFreeItemSearch will be updated, as will any relevant mapping tables.

Appendices

A. Abbreviations

ACI	Application Control Interface
EXP	Alcatel Exploitation Layer
SAP	Service Access Point
SIM	Subscriber Identity Module

B. Storage Area Conversion Table

T_BAT_storage	T_PHB_TYPE	T_ACI_PB_STOR	GSM 07.07
BAT_STORAGE_FD	FDN	PB_STOR_Fd	FD
BAT_STORAGE_DC	LDN	PB_STOR_Ld	DC or LD
BAT_STORAGE_EN	ECC	PB_STOR_Ed	EN
BAT_STORAGE_AD	ADN	PB_STOR_Ad	MT or AD
BAT_STORAGE_BD	BDN	PB_STOR_Bd	BD
BAT_STORAGE_RC	LRN	PB_STOR_Lr	RC or LR
BAT_STORAGE_SD	SDN	PB_STOR_Sd	SD
BAT_STORAGE_MC	LMN	PB_STOR_Lm	MC or LM
BAT_STORAGE_AF	ADN_FDN	PB_STOR_Af	AF
BAT_STORAGE_UD	UPN	PB_STOR_Ud	ON or UD