Technical Document -Confidential

GSM PROTOCOL STACK

G23

SIM – SUBSCRIBER IDENTITY MODULE

Document Number:	6147.707.99.102
Version:	0.4
Status:	Draft
Approval Authority:	
Creation Date:	1997-Dec-11
Last changed:	2015-Mar-08 by XGUTTEFE
File Name:	6147_707.doc

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.

Change History

Date	Changed by	Approved by	Version	Status	Notes
1997-Dec-11	LE et al		0.1		1
1999-Jan-28	UB et al		0.2		2
1999-Feb-08	MS et al		0.3		3
2003-May-07	XGUTTEFE		0.4	Draft	

Notes:

1. Initial version
2. New WCS template
3. English check

Table of Contents

List of Figures and Tables

List of References

[ISO 9000:2000] International Organization for Standardization. Quality management systems - Fundamentals and vocabulary. December 2000

1.1 References

[C_8410.001] 8410.001.98.102; September 18, 1998
G23 Product Description; Condat

[C_8410.008] 8410.008.98.002; June 15, 1998
GTI Interface Description; Condat

[C_8410.003] 8410.003.98.103; September 09, 1998
Test Facilities Description; Condat

1.2 Abbreviations

IMSI	International Mobile Subscriber Identity
MMI	Man Machine Interface

1.3 Terms

---

2 Introduction

G23 is a software package implementing Layers 2 and 3 of the ETSI-defined GSM air interface signaling protocol, and as such represents the 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 standardized 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.

For a detailed reference of the G23 components, please refer to the Product Description [C_8410.001]. For detailed information regarding integration into the target system, please refer to the Generic Target Interface [C_8410.008]. For detailed information about the compiling and linking procedure, please refer the User Guide on the delivery CD.

This Technical Documentation document shows how to use the SIM object in target systems. It lists the headers involved and describes how to link SIM with other components. The customer specific functions included are listed and described.

3 Structure

3.1 Headers

The modules include several header files. Header files which are changeable by the user are marked (*). These header files are used to integrate the protocol stack entities into a specific target system.

ccdapi.h (*)

This header defines the prototypes and some constants for the Condat Coder Decoder (CCD).

sim.h

This header contains constants for the SIM application and the prototypes of the component.

cnf_sim.h (*)

Constants for the dynamic configuration of SIM are defined in this header. It is acceptable to change the commands and the parameter names for the dynamic configurations supported.

custom.h (*)

This header defines global constants for the integration of the protocol stack entity into a specific target system. The user may define the identifier of the communication resource, the supported traces, the communication method (by copying primitives or by exchanging references of primitives) , the custom specific primitive header, etc.

cus_sim.h (*)

Custom specific definitions for the protocol stack entity are located in this header. A version identifier is defined.

gsm.h

This header contains global definitions for all protocol stack entities. Depending on the definitions in custom.h, many options and traces are defined in this header.

mconst.cdg

This header is generated by the CCD compiler. It includes all message identifiers and some constants needed by the entities.

message.h

Constants for messages are defined. SIM application does not use messages.

mon_sim.h

Constants for the monitoring of SIM are defined in this header.

pconst.cdg

This header is generated by the CCD compiler. It includes all primitive identifiers and some constants needed by the entities.

pei.h (*)

Prototypes for the protocol stack entity interface are defined in this header. Some parameters and the return types of this function are changeable by the user.

prim.h

Constants for primitives are defined and service access point dependent primitive header files are included (p_sim.h).

p_sim.h

This header is generated by the CCD compiler. It includes the C-struct type definitions for the primitives of the service access point SIM. The header is included by prim.h

stddefs.h

This header contains several standard definitions used by the protocol stack entities.

string.h

This header is a the standard string header from the target compiler. It defines string and memory functions.

tok.h

The prototypes and some constants for the parse function of the TOK module are defined in this header.

vsi.h (*)

Prototypes for the virtual system interface are defined in this header. Some parameters and the return types of these functions are changeable by the user for integration into a specific target system.

3.2 Dynamic Configuration

Dynamic configuration means to change the behavior of the protocol stack entity at run-time. This is carried out by sending a string with a dedicated format as described in Test Facilities [C_8410.003]. An additional feature is to request the old configuration. This feature may be switched off by an option defined in custom.h.

The dynamic configuration string is a parameter of the pei_config () function which is part of the protocol stack entity interface (PEI). The keywords for the dynamic configurations are changeable by the static configuration mechanism.

At the moment, no dynamic configurations are defined for SIM.

3.3 Custom Specific Functions

Custom specific functions are implemented in the module sim_csf.c. It is acceptable to replace the functions in this module with functions from the customer. It is not acceptable to change parameters of the functions.

The idea behind these custom specific functions is to have a mechanism with which to configure the protocol stack entity at run-time by a source outside the protocol stack entity, for example a non erasable memory.

For the SIM application, this module contains the SIM driver calls. The whole module is replaced for the target system by the SIM driver.

```
GLOBAL USHORT SIM_ChangeCHV      (   BYTE        *   result,
                                     BYTE        *   oldCHV,
                                     BYTE        *   newCHV,
                                     BYTE            chvType,
                                     BYTE        *   size )
```

This function changes a PIN on the SIM card.

```
GLOBAL USHORT SIM_DisableCHV     (   BYTE        *   result,
                                     BYTE        *   CHV,
                                     BYTE        *   size )
```

This function disables the PIN check on the SIM Card.

```
GLOBAL USHORT SIM_EnableCHV      (   BYTE        *   result,
                                     BYTE        *   CHV,
                                     BYTE        *   size )
```

This function enables the PIN check on the SIM Card.

```
GLOBAL USHORT SIM_GetResponse    (   BYTE        *   result,
                                     BYTE            len,
                                     BYTE        *   size )
```

The SIM driver function GetResponse is carried out. This is used to get information about the SIM card.

```
GLOBAL USHORT SIM_Increase       (   BYTE        *   result,
                                     BYTE        *   dat,
                                     BYTE        *   size )
```

This SIM driver function increases a counter of SIM card.

```
GLOBAL void SIM_Init             (     void        (*insert)(void),
                                       void        (*remove)(void) )
```

This function initializes the SIM driver. It defines the call back functions for insertion and removal of the SIM card.

```
GLOBAL USHORT SIM_Invalidate       (   BYTE        *  result,
                                       BYTE        *  size )
```

This function replaces the SIM driver Call Invalidate.

```
GLOBAL USHORT SIM_ReadBinary        (  BYTE        *  result,
                                       USHORT      offset,
                                       BYTE           len,
                                       BYTE        *  size)
```

This function reads a binary field of the SIM card.

```
GLOBAL USHORT SIM_ReadRecord        (  BYTE        *  result,
                                       BYTE           mode,
                                       BYTE           recNum,
                                       BYTE           len,
                                       BYTE        *  size )
```

This functions reads a record field of the SIM card.

```
GLOBAL USHORT SIM_Rehabilitate     (   BYTE        *  result,
                                       BYTE        *  size )
```

This function replaces the SIM driver Call Rehabilitate.

```
GLOBAL int SIM_Reset             (     T_SIM_CARD  * Info )
```

This functions resets the SIM driver. As a response, the data read by the SIM driver after the ATR signal is given to the application.

```
GLOBAL USHORT SIM_RunGSMAlgo        (  BYTE        *  result,
                                       BYTE        *  rand,
                                       BYTE        *  size )
```

This function carries out the GSM authentication algorithm to get the authentication response (Sres) and the cipher key (Kc).

```
GLOBAL USHORT SIM_Seek             (   BYTE        *  result,
                                       BYTE        *  dat,
                                       BYTE           mode,
                                       BYTE           len,
                                       BYTE        *  size )
```

This functions searches after a suitable record on the SIM card.

```
GLOBAL USHORT SIM_Select           (   USHORT         id,
                                       BYTE        *  dat,
                                       BYTE        *  size )
```

This function selects an elementary field or a directory of the SIM card.

```
GLOBAL USHORT SIM_Status           (   BYTE        *  result,
                                       BYTE        *  size )
```

This function reads the status of a SIM card field.

```
GLOBAL USHORT SIM_UnblockCHV     (   BYTE       *   result,
                                     BYTE       *   unblockCHV,
                                     BYTE       *   newCHV,
                                     BYTE           chvType,
                                     BYTE       *   size )
```

This function unblocks a SIM card. This happens after too many attempts with an incorrect PIN number.

```
GLOBAL USHORT SIM_UpdateBinary   (   BYTE       *   result,
                                     BYTE       *   dat,
                                     USHORT         offset,
                                     BYTE           len,
                                     BYTE       *   size )
```

This function writes to a binary field of the SIM card. The content is updated.

```
GLOBAL USHORT SIM_UpdateRecord   (   BYTE       *   result,
                                     BYTE       *   dat,
                                     BYTE           mode,
                                     BYTE           recNum,
                                     BYTE           len,
                                     BYTE       *   size )
```

This function writes to a record of the SIM card. The content is updated.

```
GLOBAL USHORT SIM_VerifyCHV      (   BYTE       *   result,
                                     BYTE       *   chv,
                                     BYTE           chvType,
                                     BYTE       *   size )
```

This function checks a user PIN number against the PIN stored on the SIM card.

3.4 Monitoring

The monitor struct includes relevant physical parameters of the protocol stack entity. The parameters are updated continuously. This way the environment always has the possibility of accessing parameters of the protocol stack. These parameters are used to create monitor reports about a display or test system, to create statistical data, etc. outside the functionality of a protocol stack but with access to protocol stack parameters. It is acceptable to read the parameters of the monitor struct, but it absolutely not acceptable to write to the monitor struct. The first parameter of the monitor struct is the version of the protocol stack entity.

The following monitor struct is defined for the protocol stack entity:

```
typedef struct
{
   T_VERSION     *   version;
} T_MONITOR;
```

Appendices

A. Acronyms

DS-WCDMA	Direct Sequence/Spread Wideband Code Division Multiple Access

B. Glossary

International Mobile Telecommunication 2000 (IMT-2000/ITU-2000)

Formerly referred to as FPLMTS (Future Public Land-Mobile Telephone System), this is the ITU's specification/family of standards for 3G. This initiative provides a global infrastructure through both satellite and terrestrial systems, for fixed and mobile phone users. The family of standards is a framework comprising a mix/blend of systems providing global roaming. <URL: http://www.imt-2000.org/>