Technical Document - Confidential

GSM PROTOCOL STACK

G23

BZ – BUZZER DRIVER

DRIVER INTERFACE

Document Number:	8415.007.99.004
Version:	0.5
Status:	Draft
Approval Authority:	
Creation Date:	1999-Nov-17
Last changed:	2015-Mar-08 by
File Name:	Buzzer_api.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
1999-Nov-17	TSE		0.1		1
1999-Nov-25	TSE		0.2		2
1999-Dec-06	TSE		0.3		3
1999-Dec-13	TSE		0.4		4
2003-May-20	XINTEGRA		0.5	Draft	

Notes:

1. Initial version

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 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, AT Command Interface
- MMI and MMI Framework (MFW)
- Test and integration support tools.

This document describes the functional interface of the G23 Buzzer driver. This driver is used to play single tones or melodies by the buzzer and to control the buzzers functions. The addresses and masks for the Buzzers control registers are hardware depending and must be set for each integration. Melodies are played by use of timers and control structures for the buzzer tones. Melodies are files with records of tone descriptions. If plugging /unplugging of a buzzer could be detected, those events have to be signaled to the upper layers via the callback function.

2 Interface description of the Buzzer driver

2.1 Data types

Name	Description
T_DESCR	Buzzer tone description
T_ACT_TONE	actual output tone description
T_BZ_CB_FUNC	call-back function type
UBYTE	unsigned 8 bit integer data type
BYTE	signed 8 bit integer data type
USHORT	unsigned 16 bit integer data type
SHORT	signed 16 bit integer data type

2.1.1 T_BZ_CB_FUNC

Definition:

```
typedef void (*T_BZ_CB_FUNC) (T_DRV_SIGNAL* in_SignalIDPtr) ;
```

Description:

This type defines a call-back function. This function is called when a device has stopped playing a sound or melody. For more details refer to the description of the status call-back function in the chapter describing the exported functions.

2.1.2 T_DESCR Type – Buzzer Tone Description

Definition:

```
typedef struct
{
  USHORT      frequency;
  UBYTE       volume;
  USHORT      length;
} T_DESCR;
```

Description:

This data type represents the characteristics frequency, volume and duration of a single buzzer tone.

Data element	Description
Frequency	Frequency of the tone in Hz.
Volume	Volume of buzzer during playing this tone (0= min, 0xFF = max).
Length	Length of the tone in milliseconds (0: endless)

2.1.3 T_ACT_TONE Type – Description of current buzzer status

Definition:

```
typedef struct
{
  UBYTE             status;
  UBYTE             call_tone;
  UBYTE             type;
  T_DESCR       * descr;
  int               FileSize;
  UBYTE             volume;
  UBYTE             style;
  USHORT            descr_index;
} T_ACT_TONE;
```

Description:

This data type represents the current characteristics of the buzzer.

Data element	Description
Status	BUZZER_SILENT , NO_TONE, BUZZER_ON, TONE_SILENT
Call_tone	Index of tone in tone table (for the use of a tone generator)
Type	BUZZER or AUDIO : play tone over buzzer or speaker?
Descr	Actual tone description
FileSize	Size of the description file in multiples of T_DESCR
Volume	Actual buzzer volume
Style	Repeats of played file
Descr_index	Actual record number in tone description file

2.2 Constants

Name	Description
BZ_CNTL0	Buzzer control register 0
BZ_CNTL1	Buzzer control register 1
BZ_CNTL2	Buzzer control register 2
BZ_CNTL3	Buzzer control register 3
BZ_LEVEL	Buzzer level control register
BZ_ON	Buzzer on/off bit in control register
BZ_NOTCONFIGURED	Device is not configured
BZ_BUSY	Device or driver is busy or already in use
BZ_FOREVER	The melody/sound shall be played until manually stopped
BZ_SIGTYPE_SOUNDEND	Indicating the device has stopped playing a sound
DRV_BUFFER_FULL	The internal buffer is exhausted
DRV_DISABLED	Driver is not enabled
DRV_ENABLED	Driver is enabled
DRV_NOTCONFIGURED	Driver is not configured
DRV_INITFAILURE	Driver initialization failed
DRV_INITIALIZED	Driver is already initialized
DRV_INTERNAL_ERROR	Unspecified internal driver error
DRV_INPROCESS	The requested function is currently being executed
DRV_INVALID_PARAMS	One or more parameters are out of range or invalid
DRV_NOTCONFIGURED	Driver is not configured
DRV_OK	Return value indicating the function completed successfully
DRV_UNKNOWN	unknown device accessed

2.3 Functions

Name	Description
BZ_Init	Initialization of the driver
BZ_Exit	Termination of the driver
BZ_Enable	activate the buzzer
BZ_Disable	stops the buzzer
BZ_PlayStruct	play a single tone
BZ_Volume	set the buzzer volume
BZ_KeyBeep	Audio feedback to user after keypress
BZ_PlayFile	Start playing a melody/sound
BZ_Abort	Abort/stop playing the current melody/sound
BZ_SetCallBack	Define a call-back called when the end status is reached
BZ_CheckHW	Check, whether a buzzer is connected or not

2.3.1 BZ_Init – Driver Initialization

Definition:

```
USHORT  BZ_Init(
        USHORT              DrvHandle
        T_BZ_CB_FUNC        in_StatusCallbackPtr
        T_DRV_EXPORT **     DrvInfo
          ) ;
```

Parameters:

Name	Description
DrvHandle	unique handle for this driver
in_StatusCallbackPtr	pointer to callback function
DrvInfo	pointer to the driver parameters (see GDI specification document for a description of T_DRV_EXPORT).

Return values:

Name	Description
DRV_OK	Initialization successful
DRV_INITIALIZED	Driver already initialized

Description

The function initializes the driver's internal data.

The driver exports its properties like its name, the functions to access driver functionality and a bitfield called flags by the parameter DrvInfo. If the driver is called by ISR, Bit (0) in the bitfield is set, otherwise this bit is cleared.

The driver stores the DrvHandle and passes it over the SignalID to the calling process every time the callback function is called.

2.3.2 BZ_Exit - Termination of the driver

Definition:

```
void BZ_Exit (void)
```

Parameters:

Name	Description
-	

Return values:

Name	Description
-	

Description

The function is called when the driver functionality is no longer needed. The function de-allocates the resources.

2.3.3 BZ_Enable – activate the buzzer

Definition:

USHORT BZ_Enable(void) ;

Parameters:

Name	Description

-

Return values:

Name	Description
DRV_OK	Function successful
BZ_BUSY	The device is already open

Description

The function activates the buzzer. The buzzer is now ready to play tones.

2.3.4 BZ_Disable – stops the buzzer

Definition:

USHORT BZ_Disable(void) ;

Parameters:

Name	Description
Caller	handle of calling process

Return values:

Name	Description
DRV_OK	Function successful
DRV_ACCESS_DENIED	The device is opened by another process

Description

The function stops the buzzer. The buzzer stops playing melodies or tones immediately. Any further calls of BZ_Tone() will have no effect until BZ_Enable() is called.

2.3.5 BZ_PlayStruct – play a single tone

Definition:

```
void BZ_PlayStruct (
    T_DESCR             tone);
```

Parameters:

Name	Description
tone	tone description

Return values:

Name	Description
-	-

Description

The function starts playing a tone with the settings in the tone description.

2.3.6 BZ_Volume – set the buzzer volume

Definition:

USHORT BZ_Volume(UBYTE v);

Parameters:

Name	Description
v	Buzzer volume (0=min, 0xFF = max).

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	One or more parameters are out of range or not allowed in that combination

Description

The function sets the Buzzer volume immediately.

2.3.7 BZ_KeyBeep – Audio feedback to user after keypress

Definition:

```
void BZ_KeyBeep(void);
```

Parameters:

Name	Description
-	

Return values:

Name	Description
-	-

Description

The function produces a tone over the buzzer to give the user a feedback after a keypress. The tone can be equal for every key or a unique tone (perhaps DTMF) for each key.

2.3.8 BZ_Playfile – Play a sound or melody

Definition:

```
USHORT BZ_Playfile
(
    T_ACT_TONE      in_Sound
    int             in_Repeats
) ;
```

Parameters:

Name	Description
in_Sound	Description of buzzer parameters for paying the sound / melody.
in_Repeats	Identifies the number of repetitions of the sound.

Return values:

Name	Description
DRV_OK	Sound is started successfully
DRV_INVALID_PARAMS	One or more parameters are out of range or not valid

Description

This function is used to play a sound. If the device is currently playing a sound, this sound will be stopped and instead the new sound will be played. The function returns immediately after the “play process” has been activated. If the calling process should be notified when the sound has stopped playing automatically, the process has to set a call-back function or a signal using the BZ_SetCallback function.

2.3.9 BZ_AbortSound – Abort playing a sound or melody

Definition:

```
USHORT BZ_AbortSound
(
 void
) ;
```

Parameters:

Name	Description
-	

Return values:

Name	Description
DRV_OK	Sound is stopped successfully
DRV_ACCESS_DENIED	Sound could not be stopped

Description

This function is used to manually abort playing a sound. When calling this function whether the call-back will be called nor a signal will be set indicating that the device has stopped playing a sound.

2.3.10 BZ_SetCallBack – Set a call-back function

Definition:

```
USHORT  BZ_SetCallBack
(
    T_BZ_CB_FUNC        in_StatusCallbackPtr,
) ;
```

Parameters:

Name	Description
in_StatusCallbackPtr	This parameter points to the function that is called when the status of a device has changed. This parameter can be NULL if you don't wish to be informed about status changes.

Return values:

Name	Description
DRV_OK	Function completed successful
DRV_ACCESS_DENIED	Driver is in use by another process

Description

This function is used to set call-back function the driver calls when the status of the a specific device has changed. This function can be called at any time after initialization. To remove a call-back call this function setting the parameter for the corresponding call-back to NULL.

2.3.11 BZ_CheckHW - Check, whether a buzzer is connected or not

Definition:

```
USHORT BZ_CheckHW
(
    USHORT          DeviceID) ;
```

Parameters:

Name	Description
DeviceID	Device to check (if multiple devices supported), ID's set by driver.

Return values:

Name	Description
TRUE	device detected
FALSE	device not found

Description

This function checks the presence of the specified (buzzer) device.

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/>