BP30
GDD Interface Specification

Edition 2006

Published by Neonseven s.r.l.,
Viale Stazione di Prosecco, 15

34010 Sgonico (Trieste) Italy

For questions on technology, delivery and prices please contact the Neonseven Offices in Italy Sgonico and Gorizia

Attention Please!

The information herein is given to describe certain components and shall not be considered as warranted characteristics.

Terms of delivery and rights to technical change reserved.

We hereby disclaim any and all warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact Neonseven.

Neonseven technologies may only be used in life-support devices or systems with the express written approval of Neonseven, if a failure of such technologies can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

Table of Contents

1 Document Mission/Scope

1.1 Mission

This document contains the interface of the Graphics Device Driver (GDD) module.

1.2 Scope

This document is addressed to SW developers who want to modify GDD module for their HW platform and dedicate SW solution.

2 List of Acronyms

Abbreviation / Term	Explanation / Definition
APOXI	Application Programming Oriented X Interface. Object oriented application
PEC	Peripheral Event Controller
SDL	Specification Description Language
GDD	Graphics Device Driver
MMI	Man Machine Interface

3 Introduction

This document describes the interface functions of the Graphics Device Driver (GDD). The N7 GDD driver is designed for the E-GOLDradio GSM baseband system chips and abstracts the Comneon APOXI MMI application from various graphics hardware, i.e.

- LCD controller

The N7 GDD driver is the customized release of the DWD GDD driver for BP30 platform. The latter is built on a generic concept, which is prepared for supporting different kinds of HW and it is easily extended. Some of the main features of the N7 GDD driver are:

- Flexible HW interface to LCDs: Serial/Parallel, 8bit/16bit.
- High speed data transmission (PEC).
- LCD only configuration.
- OSE operating system.

4 Overall interface description

This subsection lists the different interface functions of the GDD driver.

4.1 Overall GDD driver interface functions

4.1.1 LCD related interface functions

Name	Parameters	Function Description
`GDD_lcd_init(id)`	LCD id	LCD controller and contrast initialization
`GDD_lcd_update(id, *vram, left, top, right, bottom)`	Pointer to rectangle to update and coordinates	Updates the requested rectangle of display
`GDD_lcd_uniform_update(id, *linbuffer, left, top, right, bottom)`	Pointer to image to display and coordinates	Updates the requested rectangle of linear buffer
`GDD_lcd_set_contrast(id, contrast)`	Id, contrast	Setting new contrast in the LCD controller
`GDD_lcd_get_contrast_limits(id, *limits)`	Id, pointer to limits	Getting the contrast limits (min., max. and default)
`GDD_lcd_set_power_save_mode(id, mode)`	Id, power save mode	Setting the LCD power save mode (on , off)
`GDD_lcd_shutdown(id)`	Id	Call this function before the power supply is shutdown

4.2 Detailed interface specification

4.2.1 GDD data types

4.2.1.1 gdd_result_type

Prototype:

```
typedef enum
{
  GDD_RESULT_ERROR,
  GDD_RESULT_OK,
  GDD_RESULT_CAM_JPEG_FILE_SIZE_ERROR,                //Not Used
  GDD_RESULT_CAM_JPEG_CODEC_BUSY,                     //Not Used
  GDD_RESULT_CAM_JPEG_ENCODE_TIMEOUT,                 //Not Used
  GDD_RESULT_CAM_JPEG_DECODE_TIMEOUT,                 //Not Used
  GDD_RESULT_CAM_JPEG_FIFO_FULL_TIMEOUT,              //Not Used
  GDD_RESULT_CAM_LCD_AUTO_TRANSFER_RELEASE_TIMEOUT,   //Not Used
  GDD_RESULT_CAM_APPLICATION_CONTROL_FAILURE,         //Not Used
  GDD_RESULT_CAM_LEVEL_OUT_OF_RANGE,                  //Not Used
  GDD_RESULT_CAM_JPEG_ENCODE_OVERFLOW,                //Not Used
  GDD_RESULT_CAM_JPEG_ENCODE_SIZE_LIMIT_VIOLATION,    //Not Used
  GDD_RESULT_INITIALIZATION_FAILURE,                  //Not Used
  GDD_RESULT_CAM_IMAGE_SIZE_ERROR,                    //Not Used
  GDD_RESULT_CAM_IMAGE_RESIZER_ERROR                  //Not Used
} gdd_result_type;
```

Description:

Generic return code uses by all GDD driver interface functions. If it success `GDD_RESULT_OK` is returned, otherwise `GDD_RESULT_ERROR`.

4.2.1.2 gdd_hw_id_type

Prototype:

```
typedef enum
{
  GDD_MAIN_LCD = 250,
  GDD_SUB_LCD,                                        //Not Used
  GDD_GDC,                                            //Not Used
  GDD_CAM,                                            //Not Used
  GDD_MAIN_CAM = GDD_CAM,                             //Not Used
  GDD_SUB_CAM,                                        //Not Used
  GDD_MAX_HW_ID                                       //Not Used
} gdd_hw_id_type;
```

Description:

Generic hardware id used by most GDD driver interface functions.

4.2.1.3 gdd_limits_type

Prototype:

```
typedef struct
{
  signed int min;
  signed int def;
  signed int max;
} gdd_limits_type;
```

Description:

This type is used for getting the minimum, maximum contrast limits. Further, the default value is returned.

4.2.1.4 gdd_lcd_power_save_mode_type

Prototype:

```
typedef enum
{
  GDD_LCD_POWER_SAVE_ON,
  GDD_LCD_POWER_SAVE_OFF
} gdd_lcd_power_save_mode_type;
```

Description:

This type is used for setting the power save mode of the LCD controller. Use `GDD_LCD_POWER_SAVE_ON` to activate power save mode and `GDD_LCD_POWER_SAVE_OFF` to deactivate.

4.2.2 GDD driver interface functions

4.2.2.1 GDD_lcd_init()

Prototype:
`gdd_result_type GDD_lcd_init(gdd_hw_id_type hw_id)`

Input parameters:

- hw_id.
  e.g. `GDD_MAIN_LCD`

Return value:

- `GDD_RESULT_OK`

Description:
`GDD_lcd_init()` must be called before any use of the GDD driver functionality is operational. The main purpose of this function is to initiate all peripherals used, and make sure that the LCD controller(s) are initialized properly, i.e. both hardware- and software reset. If a valid contrast value is given in static NV, this is used to set the LCD contrast; otherwise, a default value is used. The entire image of the LCD panel is cleared (black), waiting for the initial LCD update from MMI/APOXI.

GDD_lcd_init()	
State	Action
DISABLED	-
AWAIT_LCD_INIT	Initialize the LCD controllers, set initial contrast and clear LCD panel. Enter AWAIT_LCD_UPDATE state.
AWAIT_LCD_UPDATE	-
IDLE	-
CAM_ENABLED	-
CAM_CAPTURED	-

4.2.2.2 GDD_lcd_update()

Prototype:
`void GDD_lcd_update(SDL Pid proc_id, ushort res_code, gdd_hw_id_type hw_id, ubyte shuge *vram, ushort left, ushort top, ushort right, ushort bottom)`

Input parameters:

- `SDL Pid proc_id,`
- `ushort res_code,`
- `gdd_hw_id_type hw_id,`
  e.g. `GDD_MAIN_LCD`
- `ubyte shuge *vram,`
  Data stream pointer that has to be updated. I.e. the pointer has to be fixed at the top left corner (left, top) of the rectangle, which has to be updated.
- `Ushort left,`
- `ushort top,`
- `ushort right,`
- `ushort bottom,`
  Coordinates specifies the rectangle area to be updated.

Description:
This function is used by the MMI/APOXI to update a current rectangle (left, top, right, bottom) of the display RAM area.

GDD_lcd_update()	
State	Action
DISABLED	-
AWAIT_LCD_INIT	-
AWAIT_LCD_UPDATE	Turn on and update LCD panel. Enter IDLE state.
IDLE	Update LCD panel
CAM_ENABLED	Suspend camera viewfinder mode, update LCD panel and resume camera viewfinder mode
CAM_CAPTURED	Update LCD panel

4.2.2.3 GDD_lcd_set_contrast()

Prototype:
```
gdd_result_type GDD_lcd_set_contrast(gdd_hw_id_type hw_id, sshort level)
```

Input parameters:

- `gdd_hw_id_type hw_id`
- `sshort level`
  New contrast value

Return value:

- `GDD_RESULT_OK`

Description:
This function is used to modify the contrast value of the LCD controller.

GDD_lcd_set_contrast()	
State	Action
DISABLED	-
AWAIT_LCD_INIT	-
AWAIT_LCD_UPDATE	Set LCD contrast before first LCD update
IDLE	Set LCD contrast
CAM_ENABLED	-
CAM_CAPTURED	-

4.2.2.4 GDD_lcd_get_contrast_limits()

Prototype:

```
gdd_result_type GDD_lcd_get_contrast_limits(gdd_hw_id_type hw_id,
                                            gdd_limits_type *dst)
```

Input parameters:

- `gdd_hw_id_type hw_id`
- `gdd_limits_type *dst`
  Pointer to where the limit values should be stored.

Return value:

- `GDD_RESULT_OK`
  `GDD_RESULT_ERROR`, if unsupported `hw_id` or if GDD driver is not initialized

Description:
This function is used to get the limit values of the display. The default value is the center value set during the production.

GDD_lcd_get_contrast_limits()	
State	Action
DISABLED	Return GDD_RESULT_ERROR
AWAIT_LCD_INIT	Return GDD_RESULT_ERROR
AWAIT_LCD_UPDATE	Get LCD contrast limits
IDLE	Get LCD contrast limits
CAM_ENABLED	Return GDD_RESULT_ERROR
CAM_CAPTURED	Return GDD_RESULT_ERROR

4.2.2.5 GDD_lcd_set_power_save_mode()

Prototype:

```
gdd_result_type GDD_lcd_set_power_save_mode(gdd_hw_id_type hw_id,
                                            gdd_lcd_power_save_mode_type mode)
```

Input parameters:

- `gdd_hw_id_type hw_id`
  `GDD_MAIN_LCD` or `GDD_SUB_LCD`
- `gdd_lcd_power_save_mode_type mode`
  `GDD_LCD_POWER_SAVE_ON` or `GDD_LCD_POWER_SAVE_OFF`

Return value:

- `GDD_RESULT_OK`

Description:

This function is activating or deactivating the power save mode of the LCD controller specified by `hw_id`. Not all controllers own the ability to reduce power consumption. Use `GDD_LCD_POWER_SAVE_ON` to activate power save mode and `GDD_LCD_POWER_SAVE_OFF` to deactivate.

GDD_lcd_set_power_save_mode()	
State	Action
DISABLED	-
AWAIT_LCD_INIT	-
AWAIT_LCD_UPDATE	-
IDLE	Set LCD power save mode
CAM_ENABLED	-
CAM_CAPTURED	-

4.2.2.6 GDD_lcd_shutdown()

Prototype:

```
gdd_result_type GDD_lcd_shutdown(gdd_hw_id_type hw_id)
```

Input parameters:

- `gdd_hw_id_type hw_id`

Return value:

- `GDD_RESULT_OK`
  `GDD_RESULT_ERROR`, if hw_id is not supported

Description:

This function must be called before the power supply is shut down.

GDD_lcd_shutdown()	
State	Action
DISABLED	Return GDD_RESULT_ERROR
AWAIT_LCD_INIT	Return GDD_RESULT_ERROR
AWAIT_LCD_UPDATE	Return GDD_RESULT_ERROR
IDLE	Shutdown LCD controller
CAM_ENABLED	Return GDD_RESULT_ERROR
CAM_CAPTURED	Return GDD_RESULT_ERROR

5 References

5.1 External

5.2 Internal

Title	Doc ID
GDD Technical Specification	AH01.SW.TS.000015

6 Document change report

	Change Reference		Record of changes made to previous released version	
Rev	**Date**	**CR**	**Section**	**Comment**
1.0	25/06/2004	N.A.	Document creation	
1.0	13/01/2006	N.A	Document updated to BP30 Platform	

7 Approval

Revision	Approver(s)	Date	Source/signature
1.0	Stefano Godeas	25/06/2004	Document stored on server
1.1	Stefano Godeas	13/01/2006	Document stored on server

8 Annex 1

None