Technical Document – Confidential

GSM PROTOCOL STACK

G23

DSPL – DISPLAY DRIVER

FUNCTIONAL SPECIFICATION

Document Number:	8415.025.99.015
Version:	0.10
Status:	Draft
Approval Authority:	
Creation Date:	1998-Sep-21
Last changed:	2015-Mar-08 by XINTEGRA
File Name:	8415_025.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
1998-Sep-21	LM et al		0.1		1
1998-Dec-15	LM et al		0.2		2
1998-Dec-17	LM et al		0.3		3
1998-Dec-17	LM et al		0.4		4
1999-Mar-02	MS et al		0.5		5
1999-Mar-23	LM et al		0.6		6
1999-Jun-03	LE et al		0.7		7
2000-Feb-04	UB et al		0.8		8

2001-May-17	GSP et al		0.9		9
2003-May-16	XINTEGRA		0.10	Draft	

Notes:

1. Initial version
2. Balanced, functions removed/added
3. Editorial, functions removed/added
4. Draft release
5. English check/New document template
6. CAPI extended (_GetFontImage, _SelectFontbyID, _SelectFontbyImage, _DrawIcon, _GetIconImage, _SetWorkShadow, _SetDisplayShadow) English check document template
7. Consistency check/Submitted
8. Condat AG change
9. DSPL_TYPE_CHARACTER added

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,
- Slim MMI [02.30] and
- Test and integration support tools.

This document describes the functional interface of the G23 display driver interface. This driver is used as a programming interface between a display device, such as an LCD (liquid crystal display) and applications.

The driver has been designed to be kept as generic as possible in order to be adopted to many different display devices including graphical and character oriented displays.

The various driver capabilities can be retrieved via the dspl_GetDeviceCaps() function. The term device capabilities includes parameters such as the size of the display and the units used for calculation. The size of the display is measured in logical units. This means in the case of a character oriented display, the logical unit is character. In the case of a graphical display, the logical unit is pixel.

The upper-left corner is always the origin of the display and has the coordinates 0,0 (x,y). The lower-right corner of the display has the coordinates X_{Max-1}, Y_{Max-1}. and represents the maximum size of the display.

The driver interface supports a set of text related functions including font selection and text formatting as well as a graphical function including image drawing.

As already mentioned, the functionality available through the driver depends on the capabilities of the display device.

With the exception of some graphical features, the driver supports all functions with the known limitation that the form of representation of a text in a specific font may differ. For example, a character oriented display might not support more than one single font. Therefore, the driver displays the text by using the font that has properties as close as possible to the requested font.

Images may not be displayed at all.

2 Interface description of the DSPL driver

2.1 Data types

Name	Description
dspl_DevCaps_Type	Device capabilities data type

2.1.1 dspl_DevCaps_Type – Device capabilities data type

Definition:

```
typedef struct dspl_DevCaps _Type
{
  UBYTE   DisplayType
  USHORT Width
```

```
    USHORT Height
} ;
```

Description:

The device capabilities data type dspl_DevCaps_Type is used to identify the properties of the display used. It includes the type of the display which can be a graphical or text oriented display, a color or mono display and the dimension of the display given in rows and columns.

The number of rows and columns is measured in logical units. Depending on the type of display (see Figure 1) the unit can be characters or pixels.

The DisplayType parameter can be one or a combination of one of the display types defined in Figure 1. Depending on the type of display, different sets of functions may be supported. For example, a monochrome display may not support or may only partly support the functions dspl_SetFrgColor() and dspl_SetBkgColor().

Display Type	Value
DSPL_TYPE_CHARACTER	0
DSPL_TYPE_GRAPHIC	1
DSPL_TYPE_COLOR	2

Figure 1

2.2 Constants

Name	Description
DSPL_TXTATTR_NORMAL	Text is displayed normal
DSPL_TXTATTR_INVERS	Text is displayed inverse
DSPL_TXTATTR_FLASH	Text is displayed flashing
DSPL_TYPE_COLOR	Display supports different colors
DSPL_TYPE_GRAPHIC	Display is pixel oriented graphical display
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver
DSPL_FBOX_CURSOR_TYPE	Cursor type, filled box cursor
DSPL_OBOX_CURSOR_TYPE	Cursor type, Outlined box cursor
DSPL_TLIN_CURSOR_TYPE	Cursor type, top edge line cursor
DSPL_BLIN_CURSOR_TYPE	Cursor type, bottom edge line cursor
DSPL_SLOWFLASH_MODE	Slow flashing cursor
DSPL_FASTFLASH_MODE	Fast flashing cursor
DSPL_STATIC_MODE	Cursor is not flashing
DSPL_BMPINVERT	Combines pixels of the destination and source bitmaps using the Boolean XOR operator
DSPL_BMPAND	Combines pixels of the destination and source bitmaps using the Boolean AND operator
DSPL_BMPCOPY	Copies the source bitmap to the destination bitmap
DSPL_BMPERASE	Inverts the destination bitmap and combines the result with the source bitmap using the Boolean AND operator
DSPL_BMPPAINT	Combines pixels of the destination and source bitmaps using the Boolean OR operator

2.3 Functions

Name	Description
dspl_Init	Initialization of the driver
dspl_Exit	De-Initialization of the driver
dspl_BitBlt	Display a Bitmap
dspl_Clear	Clear the display
dspl_Enable	Turn display on/off
dspl_DrawIcon	Draw a driver internal icon
dspl_DrawLine	Draw a line from coordinate A to coordinate B
dspl_DrawRect	Draw a rectangle at a specific position
dspl_DrawEllipse	Draw an ellipse at a specific position
dspl_GetDeviceCaps	Retrieve the display capabilities (dimensions, units)
dspl_SetDeviceCaps	Set the display capabilities (dimensions, units)
dspl_GetFontImage	Retrieve the address of a driver internal font
dspl_SetCursor	Set the type of cursor and its mode
dspl_SetCursorPos	Set the position of the cursor
dspl_ShowCursor	Hide or show the cursor
dspl_SetBkgColor()	Set the background color
dspl_SetFrgColor()	Set the foreground color
dspl_SelectFontbyID	Select a driver-specific font
dspl_SelectFontbyImage	Select an application specific font
dspl_GetFontHight	Retrieve the maximum vertical size of a font
dspl_GetTextExtent	Retrieve the dimension of a given string
dspl_GetMaxTextLen	Calculate the sub-string that can be displayed in given row extent
dspl_TextOut	Display a null terminated string
dspl_SetWorkShadow	Set the address of the shadow RAM for drawing
dspl_SetDisplayShadow	Set the address of the shadow RAM of which the contents are displayed

2.3.1 dspl_Init – Driver Initialization

Definition:

```
UBYTE dspl_Init
(
    void
) ;
```

Parameters:

Name	Description
-	-

Return values:

Name	Description
DSPL_OK	Initialization successful
DSPL_INITIALIZED	Driver already initialized

Description

The function initializes the driver's internal data. The function returns DRV_INITIALIZED if the driver has already been initialized and is ready to be used or is already in use. In case of an initialization failure, which means that the driver cannot be used, the function returns DRV_INITFAILURE.

2.3.2 dspl_Exit – De-initialization of the driver

Definition:

```
void dspl_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" all allocated resources and finalizes the driver.

2.3.3 dspl_Clear – Clear a specific display region

Definition:

```
UBYTE dspl_Clear
(
        USHORT          in_X1
        USHORT          in_Y1
        USHORT          in_X2
        USHORT          in_Y2
) ;
```

Parameters:

Name	Description
in_X1	Specifies the x - coordinate of the upper-left corner
in_Y1	Specifies the y – coordinate of the upper-left corner
in_X2	Specifies the x – coordinate of the lower-right corner
in_Y2	Specifies the y - coordinate of the lower-right corner

Return values:

Name	Description
DRV_OK	Function successful
DRV_INPROCESS	The driver could not complete the clearance of the display at once. The driver is busy clearing the buffers.

Description

This function is used to clear a specific region of the display using the current background color. The region is specified by the upper left corner (X1, Y1) and the lower right corner (X2, Y2) inclusive. The background color can be set using the dspl_SetBkgColor() function (see Note).

NOTE: The display may not be a color display and therefore may not support different colors.

2.3.4 dspl_ClearAll – Clear the whole display

Definition:

```
UBYTE dspl_ClearAll
(
        void
) ;
```

Parameters:

Name	Description
-	-

Return values:

Name	Description
DRV_OK	Function successful
DRV_INPROCESS	The driver could not complete the clearance of the display at once. The driver is busy clearing the buffers.

Description

This function is used to clear the whole display.

2.3.5 dspl_Enable – Turn display on or off

Definition:

```
void dspl_Enable
(
    UBYTE                   in_Enable
) ;
```

Parameters:

Name	Description
in_Enable	Turn display on (= 1) or off (= 0)

Return values:

Name	Description
-	-

Description

This function is used to turn the display on or off. While a display is switched off, it is not possible to perform any drawing functions.

2.3.6 dspl_GetDeviceCaps – Retrieve the display capabilities

Definition:

```
void dspl_GetDeviceCaps
(
    dspl_DevCaps_Type*     out_DeviceCapsPtr
) ;
```

Parameters:

Name	Description
out_DeviceCapsPtr	Pointer to the device capabilities data type buffer

Return values:

Name	Description
-	-

Description

This function is used to retrieve the capabilities of the display device including the dimension of the display and the logical unit in which the dimension is measured. For more information about the dspl_DevCaps_Type data type, see Chapter 2.1.1.

2.3.7 dspl_SetDeviceCaps – Set the display capabilities

Definition:

```
void dspl_SetDeviceCaps
(
    dspl_DevCaps_Type*    in_DeviceCapsPtr
) ;
```

Parameters:

Name	Description
In_DeviceCapsPtr	Pointer to the device capabilities data type buffer

Return values:

Name	Description
-	-

Description

This function is used to set the capabilities of the display device including the dimension of the display and the logical unit in which the dimension is measured. For more information about the dspl_DevCaps_Type data type, see Chapter 2.1.1.

2.3.8 dspl_GetIconImage – Retrieve a Driver internal Icon Image

Definition:

```
UBYTE dspl_GetIconImage
(
    UBYTE           in_Icon
    USHORT          in_Size
    UBYTE *         out_IconImagePtr
) ;
```

Parameters:

Name	Description
in_Icon	Value identifying the icon of which the image is retrieved
in_Size	Value indicating the size of the buffer to which out_IconImagePtr points
out_IconImagePtr	Address of the buffer to which the icon image is copied

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Font not available
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to copy the image of a driver internal icon into an application specific icon buffer. The application may modify the icon.

In the case of a successful completion, the function returns DRV_OK.

If the size of the buffer to which the icon image is to be copied is too small, the driver returns DRV_INVALID_PARAMS.

If a specific driver implementation does not support this functionality, the driver returns DSPL_FCT_NOTSUPPORTED.

2.3.9 dspl_SetCursor – Set cursor type and mode

Definition:

```
UBYTE dspl_SetCursor
(
    UBYTE           in_CursorType
    UBYTE           in_CursorMode
) ;
```

Parameters:

Name	Description
in_CursorType	Type of cursor (see following table)
in_CursorMode	Mode of the cursor (see following table)

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Parameters are out of range or not allowed

Description

This function is used to change the current cursor settings. These settings include the type of cursor and the mode [e.g. static cursor (not flashing)]. A set of standard types and modes is defined in the following table. This set can be extended by user defined types and modes.

Note: The modes and types of cursors defined may not be supported in all implementations.

Type	Value
DSPL_FBOX_CURSOR_TYPE	1
DSPL_OBOX_CURSOR_TYPE	2
DSPL_TLIN_CURSOR_TYPE	3
DSPL_BLIN_CURSOR_TYPE	4

Figure 2

Mode	Value
DSPL_STATIC_MODE	1
DSPL_FASTFLASH_MODE	2
DSPL_SLOWFLASH_MODE	3

Figure 3

2.3.10 dspl_SetCursorPos – Set the cursor position

Definition:

```
UBYTE dspl_SetCursorPos
(
    USHORT          in_X
    USHORT          in_Y
) ;
```

Parameters:

Name	Description
in_X	Specifies the x – coordinate
in_Y	Specifies the y – coordinate

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Parameters are out of range or not allowed

Description

This function is used to set the current cursor position. If the function succeeds, the cursor is set to the new position.

If one of the values is out of range, the function returns DRV_INVALID_PARAMS.

The number of rows and columns the display supports can be retrieved using the function dspl_GetDevCaps(). The upper left cell has the coordinates 0,0. The values in_X and in_Y are measured in logical units depending on the capabilities of the device (see dspl_GetDevCaps()). This means that a graphical display will use pixels.

2.3.11 dspl_ShowCursor – Hide or show the cursor

Definition:

```
UBYTE dspl_ShowCursor
(
    UBYTE                   in_Show
) ;
```

Parameters:

Name	Description
in_Show	New status of the cursor (= 0 – hide, >0 - show)

Return values:

Name	Description
DSPL_CURSOR_VISIBLE	Cursor was visible
DSPL_CURSOR_INVISIABLE	Cursor was invisible

Description

This function is used to change the status of the cursor. The cursor can be visible or invisible. The function returns the previous status of the cursor.

2.3.12 dspl_SetBkgColor – Set background color

Definition:

```
UBYTE dspl_SetBkgColor
(
    UBYTE                   in_Color
) ;
```

Parameters:

Name	Description
in_Color	Color value to be set for background painting

Return values:

Name	Description
DRV_OK	Color changed
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to change the color used for background painting. If the color value is out of range, the driver returns DSPL_INVALID_PARAMS and leaves the color unchanged.

NOTE: The display may not be a color display and therefore may not support different colors.

2.3.13 dspl_SetFrgColor – Set foreground color

Definition:

```
UBYTE dspl_SetFrgColor
(
    UBYTE                   in_Color
) ;
```

Parameters:

Name	Description
in_Color	Color value to be set for foreground painting

Return values:

Name	Description
DRV_OK	Color changed
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to change the color used for foreground painting, e.g. text color. If the color value is out of range, the driver returns DSPL_INVALID_PARAMS and leaves the color unchanged.

NOTE: The display may not be a color display and therefore may not support different colors.

2.3.14 dspl_DrawIcon – Draw a Driver internal Icon

Definition:

```
UBYTE dspl_DrawIcon
(
    UBYTE       in_IconID
    USHORT      in_X1
    USHORT      in_Y1
    USHORT      in_X2
    USHORT      in_Y2
) ;
```

Parameters:

Name	Description
in_IconID	Value identifying the driver internal icon that is to be drawn
in_X1	Specifies the x - coordinate of the upper-left corner
in_Y1	Specifies the y - coordinate of the upper-left corner

Return values:

Name	Description
DRV_OK	Configuration successful
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to draw a driver internal icon. The origin of the icon is the upper left corner, defined by the parameters in_X1/in_Y1.

NOTE: The display may not be a graphical display and therefore may not support this function at all.

2.3.15 dspl_DrawLine – Draw a Line

Definition:

```
UBYTE dspl_DrawPoint
(
    USHORT          in_X1
    USHORT          in_Y1
    USHORT          in_X2
    USHORT          in_Y2
) ;
```

Parameters:

Name	Description
in_X1	Specifies the x - coordinate of the starting point
in_Y1	Specifies the y - coordinate of the starting point
in_X2	Specifies the x - coordinate of the ending point
in_Y2	Specifies the y - coordinate of the ending point

Return values:

Name	Description
DRV_OK	Configuration successful
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to draw a line from a specific location defined by the parameters in_X1/in_Y1, to a specific location defined by the parameters in_X2/in_Y2.

The display's origin is the upper left corner with the coordinates (0,0).

This function uses the current foreground color, which can be set using the dspl_SetFrgColor(), to draw the line.

NOTE1: The display may not be a graphical display and therefore may not support this function at all.

NOTE2: The display may not be a color display and therefore may not support different colors.

2.3.16 dspl_DrawRect – Draw a rectangle

Definition:

```
UBYTE dspl_DrawPoint
(
    USHORT          in_X1
    USHORT          in_Y1
    USHORT          in_X2
    USHORT          in_Y2
) ;
```

Parameters:

Name	Description
in_X1	Specifies the x - coordinate of the upper-left corner
in_Y1	Specifies the y - coordinate of the upper-left corner
in_X2	Specifies the x - coordinate of the lower-right corner
in_Y2	Specifies the y - coordinate of the lower-right corner

Return values:

Name	Description
DRV_OK	Configuration successful
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to draw a rectangle. The upper left corner of the rectangle is defined by the parameters in_X1/in_Y1. The lower right corner of the rectangle is defined by the parameters in_X2/in_Y2.

The display's origin is the upper left corner with the coordinates (0,0).

This function uses the current foreground color, which can be set using the dspl_SetFrgColor(), to draw the rectangle.

NOTE1: The display may not be a graphical display and therefore may not support this function at all.

NOTE2: The display may not be a color display and therefore may not support different colors.

2.3.17 dspl_DrawEllipse – Draw an ellipse

Definition:

```
UBYTE dspl_DrawEllipse
(
    USHORT          in_X1
    USHORT          in_Y1
    USHORT          in_X2
    USHORT          in_Y2
) ;
```

Parameters:

Name	Description
in_X1	Specifies the x - coordinate of the upper-left corner of the ellipse's bounding rectangle
in_Y1	Specifies the y - coordinate of the upper-left corner of the ellipse's bounding rectangle
in_X2	Specifies the x - coordinate of the lower-right corner of the ellipse's bounding rectangle
in_Y2	Specifies the y - coordinate of the lower-right corner of the ellipse's bounding rectangle

Return values:

Name	Description
DRV_OK	Configuration successful
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to draw an ellipse. The center of the ellipse is the center of the bounding rectangle specified by in_X1, in_Y1, in_X2 and in_Y2.

The figure drawn by this function extends up to, but does not include, the right and bottom coordinates. This means that the height of the figure is in_Y2 – in_Y1 and the width of the figure is in_X2 – in_X1.

If either the width or the height of the bounding rectangle is 0, no ellipse is drawn and the function returns DSPL_INVALID_PARAMS.

The display's origin is the upper left corner with the coordinates (0,0).

This function uses the current foreground color, which can be set using the dspl_SetFrgColor(), to draw the ellipse.

NOTE1: The display may not be a graphical display and therefore may not support this function at all.

NOTE2: The display may not be a color display and therefore may not support different colors.

2.3.18 dspl_BitBlt – Display a Bitmap

Definition:

```
UBYTE dspl_BitBlt
(
    USHORT      in_X
    USHORT      in_Y
    USHORT      in_Width
    USHORT      in_Height
    USHORT      in_Index
    void*       in_BmpPtr
    USHORT      in_Rop
) ;
```

Parameters:

Name	Description
in_X	Specifies the x - coordinate of the upper-left corner of the bit-map
in_Y	Specifies the y - coordinate of the upper-left corner of the bitmap
in_Width	Specifies the width of the bitmap
in_Height	Specifies the height of the bitmap
in_Index	Index in the bitmap
in_BmpPtr	Pointer to the bitmap
in_Rop	Raster operation

Return values:

Name	Description
DSPL_OK	Configuration successful
DSPL_INVALID_PARAMS	Parameters are out of range or not allowed
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to display a bitmap at the specified location using the raster operation provided. The bitmap format is customer specific but must include the size of the bitmap. The following table shows the possible raster operations that the driver may support.

Raster operation	Value	Description
DSPL_BMPINVERT	1	Combines pixels of the destination and source bitmaps using the Boolean XOR operator
DSPL_BMPAND	2	Combines pixels of the destination and source bitmaps using the Boolean AND operator
DSPL_BMPCOPY	4	Copies the source bitmap to the destination bitmap
DSPL_BMPERASE	8	Inverts the destination bitmap and combines the result with the source bitmap using the Boolean AND operator
DSPL_BMPPAINT	16	Combines pixels of the destination and source bitmaps using the Boolean OR operator

Figure 4

2.3.19 dspl_SelectFontbyID – Select a Driver Specific Font

Definition:

```
UBYTE dspl_SelectFontbyID
(
    UBYTE                   in_Font
) ;
```

Parameters:

Name	Description
in_Font	Value identifying the font

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Font not available
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to select a font used for displaying text. Text can be displayed using the functions dspl_TextOut1() and dspl_TextOut2().

Driver specific fonts are identified by a font ID (parameter in_Font). The definition of fonts and font identifiers is not in the scope of this document. Fonts and font identifiers have to be defined by the customer. The specific implementation of the display driver and the MMI using the driver have the knowledge about the available fonts, their identification and how to use them.

Aside from the two functions mentioned previously, the following functions are based on the knowledge of fonts:

- dspl_GetTextExtent()
- dspl_GetMaxTextLen()
- dspl_GetFontHeight()

NOTE: Font selection may not be supported in different implementations of the display driver and therefore this function may not be supported.

2.3.20 dspl_SelectFontbyImage – Select a Application Specific Font

Definition:

```
UBYTE dspl_SelectFontbyImage
(
    UBYTE *                 in_FontPtr
) ;
```

Parameters:

Name	Description
in_FontPtr	Address of the buffer containing the font

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Font not available
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to select a font used for displaying text. Text can be displayed using the functions dspl_TextOut1() and dspl_TextOut2().

Application specific fonts are identified by the parameter in_FontPtr, the address of the buffer containing the application specific font. The structure of the font image is not in the scope of this document. The structure of font images must be defined by the customer implementing the driver.

A driver may also provide applications with driver specific fonts that can be selected using the function dspl_SelectFontbyID.

NOTE: Font selection may not be supported in different implementations of the display driver and therefore this function may not be supported.

2.3.21 dspl_GetFontImage – Retrieve a Driver internal Font Image

Definition:

```
UBYTE dspl_SelectFontbyID
(
    UBYTE           in_Font
    USHORT          in_Size
    UBYTE *         out_FontImagePtr
) ;
```

Parameters:

Name	Description
in_Font	Value identifying the font of which the image is retrieved
in_Size	Value indicating the size of the buffer to which out_FontImagePtr points
out_FontPtr	Address of the buffer to which the image is copied

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Font not available
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

This function is used to copy the image of a font into an application specific font buffer. The application may modify the font.

In the case of a successful completion, the function returns DRV_OK.

If the size of the buffer to which the font image is to be copied is too small, the driver returns DRV_INVALID_PARAMS.

If a specific driver implementation does not support this functionality, the driver returns DSPL_FCT_NOTSUPPORTED.

2.3.22 dspl_GetFontHeight – Retrieve the vertical font size

Definition:

```
UBYTE dspl_GetFontHeight
(
    void
) ;
```

Parameters:

Name	Description
-	-

Return values:

Name	Description
0	Function not supported
Size	Size of the font measured in logical units depending on the display capabilities

Description

This function is used to retrieve the vertical size of the currently selected font. The function returns the height measured in logical units depending on the device capabilities (e.g. pixels or multiple of characters). Call the function dspl_SelectFont() to select a font.

NOTE: Font selection may not be supported in different implementations of the display driver and therefore this function may not be supported.

2.3.23 dspl_GetTextExtent – Retrieve the horizontal dimension of a string

Definition:

```
USHORT dspl_GetTextExtent
(
    char *          in_Text
    USHORT          in_Length

) ;
```

Parameters:

Name	Description
in_Text	string of which the size is to be calculated
in_Length	length of the text

Return values:

Name	Description
Size	Horizontal size of the given text measured in units depending on the display capabilities

Description

This function is used to calculate the size of the string, to which the parameter in_Text points. The function returns the size needed to display the text. The value of the size is measured in units depending on the device capabilities (e.g. pixels or multiple of characters). Call the function dspl_SelectFont() to select a font.

NOTE: Font selection may not be supported in different implementations of the display driver and therefore this function may not be supported in different implementations of the display driver.

2.3.24 dspl_GetMaxTextLen – Calculate the displayable sub-string of a given string

Definition:

```
USHORT dspl_GetMaxTextLen
(
    char *          in_Text
    USHORT          in_HSize
) ;
```

Parameters:

Name	Description
in_Text	0 - terminated string
in_HSize	Maximum horizontal size measured in logical units depending on the device capabilities

Return values:

Name	Description
Len	Number of characters of the given string that can be displayed in the given region

Description

This function is used to calculate the sub-string (number of characters) of a 0 – terminated string (in_Text) that can be displayed in the region given by the parameter in_Hsize. The value of in_HSize is measured in logical units (e.g. pixels or multiple of characters). Call the function dspl_SelectFont() to select a font.

NOTE: Font selection may not be supported in different implementations of the display driver and therefore this function may not be supported.

2.3.25 dspl_TextOut – Display a null terminated string

Definition:

```
UBYTE dspl_TextOut
(
    USHORT          in_X
    USHORT          in_Y
    UBYTE           in_Attrib
    char*           in_Text
) ;
```

Parameters:

Name	Description
in_X	Specifies the x - coordinate of the starting position
in_Y	Specifies the y - coordinate of the starting position
in_Attrib	Attributes for displaying the text
in_Text	Null terminated string to be displayed

Return values:

Name	Description
DRV_OK	Configuration successful
DRV_INVALID_PARAMS	Parameters are out of range or not allowed

Description

This function is used to display a text at the given location in_X/in_Y using the defined attributes (in_Attrib) and the currently selected color (foreground and background). The cursor position is left unchanged. The driver will not take the displays bounding rectangle into consideration and therefore text may be cut off.

Linefeeds and carriage returns are not supported and therefore must be handled by the upper layer. Figure 5 shows the available attributes.

Use the dspl_GetTextExtent() function to calculate the horizontal and vertical size of a text or call the dspl_GetMaxTextLen() to calculate the number of characters of a given string that can be displayed in a row with a restricted size using the currently selected font. A font can be selected using the dspl_SelectFont() function.

The background and foreground colors can be selected by calling the dspl_SetFrgColor() and dspl_SetBkgColor() function.

Attribute (see NOTE)	**Value**
DSPL_TXTATTR_NORMAL	0
DSPL_TXTATTR_INVERS	1
DSPL_TXTATTR_FLASH	2

Figure 5

NOTE1: The specific driver implementation may not support all display attributes shown in Figure 5.

NOTE2: The specific driver implementation or display device may not support color selection and/or font selection.

2.3.26 dspl_SetWorkShadow – Set the Address of the Shadow RAM for Drawing

Definition:

```
UBYTE dspl_SetWorkShadow
(
    UBYTE *                 in_ShadowPtr
) ;
```

Parameters:

Name	Description
in_ShadowPtr	Pointer to the shadow RAM used for drawing

Return values:

Name	Description
DRV_OK	Function completed successfully
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

The function defines the shadow RAM area used for writing without displaying the content.

2.3.27 dspl_SetDisplayShadow – Set the Address of the Shadow RAM

Definition:

```
UBYTE dspl_SetDisplayShadow
(
    UBYTE *                 in_ShadowPtr
) ;
```

Parameters:

Name	Description
in_ShadowPtr	Pointer to the shadow RAM of which the contents are to be displayed

Return values:

Name	Description
DRV_OK	Function completed successfully
DSPL_FCT_NOTSUPPORTED	Function is not supported by this driver

Description

The function defines the display RAM area.

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/>