

# **DE10-Lite VGA IP**

Version 2.0

11/20/2020 8:36:00 AM



# Table of Contents

File Index .....	1
File Documentation.....	1
HAL/inc/char_map.h.....	1
HAL/inc/DE10_Lite_VGA_Driver.h .....	2
HAL/inc/draw_vga.h .....	3
Index .....	6

---

# File Index

## File List

Here is a list of all documented files with brief descriptions:

<b>HAL/inc/char_map.h</b>	1
<b>HAL/inc/DE10_Lite_VGA_Driver.h</b>	2
<b>HAL/inc/draw_vga.h</b>	3

---

# File Documentation

## HAL/inc/char\_map.h File Reference

```
#include <alt_types.h>
```

### Macros

- **#define TTY\_CHAR\_OFFSET 0x20**  
*TTY character position offset from standard ASCII code.*
- **#define TTY\_CHAR\_WIDTH 8**  
*Width of a tty character.*
- **#define TTY\_CHAR\_HEIGHT 8**  
*Height of a tty character.*

### Functions

- **alt\_u32 upper\_char\_line** (const alt\_u8 char\_pos)  
*Returns the 32-bit representation of the top half of the requested character.*
- **alt\_u32 lower\_char\_line** (const alt\_u8 char\_pos)  
*Returns the 32-bit representation of the bottom half of the requested character.*

---

## Detailed Description

Content for 8x8 pixels TinyFont

### Author

Calle Melander  
Jens Lind

**Version**

2.0

**Date**

2017-2020

**Copyright**

AGSTU AB

---

**Function Documentation****alt\_u32 lower\_char\_line (const alt\_u8 char\_pos)**

Returns the 32-bit representation of the bottom half of the requested character.

**Parameters**

<i>char_pos</i>	Character position using ASCII code but remove TTY_CHAR_OFFSET.
-----------------	---

**alt\_u32 upper\_char\_line (const alt\_u8 char\_pos)**

Returns the 32-bit representation of the top half of the requested character.

**Parameters**

<i>char_pos</i>	Character position using ASCII code but remove TTY_CHAR_OFFSET.
-----------------	---

---

**HAL/inc/DE10\_Lite\_VGA\_Driver.h File Reference**

```
#include <system.h>
#include <io.h>
```

**Macros**

- **#define CANVAS\_WIDTH** 320
- **#define CANVAS\_HEIGHT** 240
- **#define CANVAS\_SIZE** (CANVAS\_WIDTH \* CANVAS\_HEIGHT)
- **#define pixel\_addr(x, y)** ((CANVAS\_WIDTH \* (y) + (x)) \* 4)  
*Returns the address offset from VGA base for pixel at given screen coordinate (x, y)*
- **#define write\_pixel(x, y, rgb)** IOWR\_32DIRECT(DE10\_LITE\_VGA\_IP\_0\_BASE, pixel\_addr(x, y), (rgb))  
*Writes a pixel with the specified color (rgb) on given screen coordinate (x, y).*
- **#define read\_pixel(x, y)** IORD\_32DIRECT(DE10\_LITE\_VGA\_IP\_0\_BASE, pixel\_addr(x, y))  
*Returns the current color value for given screen coordinate (x, y)*

## Enumerations

- enum **Color** { **Col\_Black** = 0, **Col\_Blue**, **Col\_Green**, **Col\_Cyan**, **Col\_Red**, **Col\_Magenta**, **Col\_Yellow**, **Col\_White** }  
*8-bit color depth*
- 

## Detailed Description

Drivers for interfacing with VGA component for the DE10-Lite board. Functionality is implemented as macros that writes to and reads from registers.

### Author

Linus Eriksson  
Jens Lind

### Version

2.0

### Date

2017-2020

### Copyright

AGSTU AB

---

## HAL/inc/draw\_vga.h File Reference

```
#include <alt_types.h>
```

## Functions

- void **clear\_screen** (alt\_u32 color)  
*This function clears the screen by writing the color value to all pixels on the screen.*
- void **draw\_hline** (alt\_u32 x0, alt\_u32 y0, alt\_u32 length, alt\_u32 color)  
*Draws a horizontal line of specified length. The function may fail if not constrained to screen.*
- void **draw\_vline** (alt\_u32 x0, alt\_u32 y0, alt\_u32 length, alt\_u32 color)  
*Draws a vertical line of specified length. The function may fail if not constrained to screen.*
- void **draw\_angled\_line** (alt\_u32 x0, alt\_u32 y0, alt\_u32 x1, alt\_u32 y1, alt\_u32 color)  
*Draws a slanting line between two specified coordinates. The function may fail if either coordinate is outside of the screen.*
- void **draw\_circle** (alt\_32 x0, alt\_32 y0, alt\_u32 radius, alt\_u32 color)  
*Writes a circle with the specified radius and color at the center coordinate (x0, y0).*
- void **draw\_filled\_circle** (alt\_32 x0, alt\_32 y0, alt\_u32 radius, alt\_u32 color)  
*Writes a circle with the specified radius and color at the center coordinate (x0, y0). The circle is filled with the same color.*
- void **tty\_print** (alt\_32 x0, alt\_32 y0, const char \*sz\_tty, alt\_u32 color, alt\_u32 BGcolor)  
*Prints a string on the screen letter by letter.*

- void **char\_print** (alt\_32 x0, alt\_32 y0, const char tty\_char, alt\_u32 color, alt\_u32 BGcolor)  
*Prints a 8x8 pixels character with the specified color and background color on the coordinate (x1, y1). The character table is in the header file **char\_map.h**.*
- void **int\_print** (alt\_32 x0, alt\_32 y0, int data, alt\_u32 data\_l, alt\_u32 color, alt\_u32 BGcolor)  
*Prints an integer to screen digit by digit. It will prefix the number with sign (+/-).*

---

## Detailed Description

Interface defining functions for drawing lines, circles, text and integer on VGA

### Author

Calle Melander  
Jens Lind

### Version

2.0

### Date

2017-2020

### Copyright

AGSTU AB

---

## Function Documentation

**void char\_print (alt\_32 x0, alt\_32 y0, const char tty\_char, alt\_u32 color, alt\_u32 BGcolor)**

Prints a 8x8 pixels character with the specified color and background color on the coordinate (x1, y1). The character table is in the header file **char\_map.h**.

### Parameters

<i>x0</i>	Top left corner horizontal position. If negative, the character will be partially cropped.
<i>y0</i>	Top left corner vertical position. If negative, the character will be partially cropped.
<i>tty_char</i>	The ASCII character to print.
<i>color</i>	Color to use for the character.
<i>BGcolor</i>	Color to use for the background of the character.

**void clear\_screen (alt\_u32 color)**

This function clears the screen by writing the color value to all pixels on the screen.

### Parameters

<i>color</i>	Color to fill the screen with.
--------------	--------------------------------

**void draw\_angled\_line (alt\_u32 x0, alt\_u32 y0, alt\_u32 x1, alt\_u32 y1, alt\_u32 color)**

Draws a slanting line between two specified coordinates. The function may fail if either coordinate is outside of the screen.

#### Parameters

<i>x0</i>	Horizontal start of the line.
<i>y0</i>	Vertical start of the line.
<i>x1</i>	Horizontal end of the line. If equal to x0, consider using draw_vline instead.
<i>y1</i>	Vertical end of the line. If equal to y0, consider using draw_hline instead.
<i>color</i>	Color of the line.

**void draw\_circle (alt\_32 x0, alt\_32 y0, alt\_u32 radius, alt\_u32 color)**

Writes a circle with the specified radius and color at the center coordinate (x0, y0).

#### Parameters

<i>x0</i>	Horizontal position for centre of circle.
<i>y0</i>	Vertical position for centre of circle.
<i>radius</i>	Radius of circle.
<i>color</i>	Color for circle border.

**void draw\_filled\_circle (alt\_32 x0, alt\_32 y0, alt\_u32 radius, alt\_u32 color)**

Writes a circle with the specified radius and color at the center coordinate (x0, y0). The circle is filled with the same color.

#### Parameters

<i>x0</i>	Horizontal position for centre of circle.
<i>y0</i>	Vertical position for centre of circle.
<i>radius</i>	Radius of circle.
<i>color</i>	Color for circle border and fill.

**void draw\_hline (alt\_u32 x0, alt\_u32 y0, alt\_u32 length, alt\_u32 color)**

Draws a horizontal line of specified length. The function may fail if not constrained to screen.

#### Parameters

<i>x0</i>	Horizontal start of the line.
<i>y0</i>	Vertical start of the line.
<i>length</i>	Horizontal distance of line - line ends at (x0+length, y0)
<i>color</i>	Color of the line.

**void draw\_vline (alt\_u32 x0, alt\_u32 y0, alt\_u32 length, alt\_u32 color)**

Draws a vertical line of specified length. The function may fail if not constrained to screen.



### Parameters

<i>x0</i>	Horizontal start of the line.
<i>y0</i>	Vertical start of the line.
<i>length</i>	Vertical distance of line - line ends at (x0, y0+length)
<i>color</i>	Color of the line.

**void int\_print (alt\_32 x0, alt\_32 y0, int data, alt\_u32 data\_l, alt\_u32 color, alt\_u32 BGcolor)**

Prints an integer to screen digit by digit. It will prefix the number with sign (+/-).

### Parameters

<i>x0</i>	Top left corner horizontal position. If negative, the number will be partially cropped.
<i>y0</i>	Top left corner vertical position. If negative, the number will be partially cropped.
<i>data</i>	The integer to print.
<i>data_l</i>	The number of digits to use. It will prefix with 0s if the number of digits are more than required.
<i>color</i>	Color to use for the number.
<i>BGcolor</i>	Color to use for the background of the number.

**void tty\_print (alt\_32 x0, alt\_32 y0, const char \* sz\_tty, alt\_u32 color, alt\_u32 BGcolor)**

Prints a string on the screen letter by letter.

### Parameters

<i>x0</i>	Top left corner horizontal position. If negative, the string will be partially cropped.
<i>y0</i>	Top left corner vertical position. If negative, the string will be partially cropped.
<i>sz_tty</i>	The ASCII characters to print.
<i>color</i>	Color to use for the characters.
<i>BGcolor</i>	Color to use for the background of the characters.

---

## Index

char_map.h	draw_vga.h, i
lower_char_line, i	draw_vga.h
upper_char_line, i	char_print, i
char_print	clear_screen, i
draw_vga.h, i	draw_angled_line, i
clear_screen	draw_circle, i
draw_vga.h, i	draw_filled_circle, i
draw_angled_line	draw_hline, i
draw_vga.h, i	draw_vline, i
draw_circle	int_print, i
draw_vga.h, i	tty_print, i
draw_filled_circle	draw_vline
draw_vga.h, i	draw_vga.h, i
draw_hline	HAL/inc/char_map.h, i

HAL/inc/DE10_Lite_VGA_Driver.h, i	char_map.h, i
HAL/inc/draw_vga.h, i	tty_print
int_print	draw_vga.h, i
draw_vga.h, i	upper_char_line
lower_char_line	char_map.h, i