
Read the Docs Template Documentation

Release

Read the Docs

Mar 10, 2018

Contents

1	Component: epaper-29-dke	1
1.1	Header File	1
1.2	Functions	1
1.3	Structures	6
1.4	Macros	6
1.5	Type Definitions	7
1.6	Enumerations	7

Component: epaper-29-dke

This component and documentation is based almost entirely on [epaper](#) component for another type of display developed by [esp-iot-solution](#) team.

1.1 Header File

- [epaper-29-dke/epaper-29-dke.h](#)

1.2 Functions

epaper_handle_t **iot_epaper_create** (*spi_device_handle_t bus*, *epaper_conf_t *epconf*)
Create and init epaper and return a epaper handle.

Return

- handle of epaper

Parameters

- *bus*: handle of spi device
- *epconf*: configure struct for epaper device

esp_err_t **iot_epaper_delete** (*epaper_handle_t dev*, *bool del_bus*)
delete epaper handle_t

Return

- `ESP_OK` Success
- `ESP_FAIL` Fail

Parameters

- dev: object handle of epaper
- del_bus: whether to delete spi bus

void **iot_epaper_clean_paint** (*epaper_handle_t dev*, int *colored*)
clear display frame buffer

Parameters

- dev: object handle of epaper
- colored: to set display color

int **iot_epaper_get_width** (*epaper_handle_t dev*)
get paint width

Return

- paint width

Parameters

- dev: object handle of epaper

void **iot_epaper_set_width** (*epaper_handle_t dev*, int *width*)
set paint width

Parameters

- dev: object handle of epaper
- width: paint width

int **iot_epaper_get_height** (*epaper_handle_t dev*)
get paint height

Return

- paint height

Parameters

- dev: object handle of epaper

void **iot_epaper_set_height** (*epaper_handle_t dev*, int *height*)
set paint height

Parameters

- dev: object handle of epaper
- paint: height

int **iot_epaper_get_rotate** (*epaper_handle_t dev*)
get paint rotate

Return

- current rotation

Parameters

- dev: object handle of epaper

void **iot_epaper_set_rotate** (*epaper_handle_t dev*, int *rotate*)
set paint rotate

Parameters

- dev: object handle of epaper
- rotation:

unsigned char ***iot_epaper_get_image** (*epaper_handle_t dev*)
get display data

Return

- Pointer to display data

Parameters

- dev: object handle of epaper

void **iot_epaper_draw_string** (*epaper_handle_t dev*, int *x*, int *y*, const char **text*, *epaper_font_t *font*, int *colored*)
draw string start on point(x,y) and save on display data array, screen will display when call iot_epaper_display_frame function.

Parameters

- dev: object handle of epaper
- x: poing (x)
- y: poing (y)
- text: display string
- font: Font style
- colored: display color

void **iot_epaper_draw_pixel** (*epaper_handle_t dev*, int *x*, int *y*, int *colored*)
draw pixel and save on display data array, screen will display when call iot_epaper_display_frame function.

Parameters

- dev: object handle of epaper
- x: point (x)
- y: point (y)
- colored: display color

void **iot_epaper_draw_char** (*epaper_handle_t dev*, int *x*, int *y*, char *ascii_char*, *epaper_font_t *font*, int *colored*)
draw char and save on display data array, screen will display when call iot_epaper_display_frame function.

Parameters

- dev: object handle of epaper
- x: poing (x)

- `y`: poing (`y`)
- `ascii_char`: display char
- `font`: font style
- `colored`: display color

void **iot_epaper_draw_line** (*epaper_handle_t dev*, int *x0*, int *y0*, int *x1*, int *y1*, int *colored*)

draw line start on point(`x0,y0`) end on point(`x1,y1`) and save on display data array, screen will display when call `iot_epaper_display_frame` function.

Parameters

- `dev`: object handle of epaper
- `x0`: poing (`x0`)
- `y0`: poing (`y0`)
- `x1`: poing (`x1`)
- `y1`: poing (`y1`)
- `colored`: display color

void **iot_epaper_draw_horizontal_line** (*epaper_handle_t dev*, int *x*, int *y*, int *width*, int *colored*)

draw horizontal line start on point(`x,y`) and save on display data array, screen will display when call `iot_epaper_display_frame` function.

Parameters

- `dev`: object handle of epaper
- `x`: poing (`x`)
- `y`: poing (`y`)
- `width`: line width
- `colored`: display color

void **iot_epaper_draw_vertical_line** (*epaper_handle_t dev*, int *x*, int *y*, int *height*, int *colored*)

draw vertical line start on point(`x,y`) and save on display data array, screen will display when call `iot_epaper_display_frame` function.

Parameters

- `dev`: object handle of epaper
- `x`: poing (`x`)
- `y`: poing (`y`)
- `line`: height
- `display`: color

void **iot_epaper_draw_rectangle** (*epaper_handle_t dev*, int *x0*, int *y0*, int *x1*, int *y1*, int *colored*)

draw rectangle point(`x0,y0`) (`x1,y1`) and save on display data array, screen will display when call `iot_epaper_display_frame` function.

Parameters

- dev: object handle of epaper
- x0: point(x0,y0)
- y0: point(x0,y0)
- x1: point(x1,y1)
- y1: point(x1,y1)
- colored: display color

void **iot_epaper_draw_filled_rectangle** (*epaper_handle_t dev*, int *x0*, int *y0*, int *x1*, int *y1*, int *colored*)

draw fill rectangle point(x0,y0) (x1,y1) and save on display data array, screen will display when call iot_epaper_display_frame function.

Parameters

- dev: object handle of epaper
- x0: point(x0,y0)
- y0: point(x0,y0)
- x1: point(x1,y1)
- y1: point(x1,y1)
- colored: display color

void **iot_epaper_draw_circle** (*epaper_handle_t dev*, int *x*, int *y*, int *radius*, int *colored*)

draw a circle at point(x,y) and save on display data array, screen will display when call iot_epaper_display_frame function.

Parameters

- dev: object handle of epaper
- x: point(x,y)
- y: point(x,y)
- colored: display color

void **iot_epaper_draw_filled_circle** (*epaper_handle_t dev*, int *x*, int *y*, int *radius*, int *colored*)

draw a fill circle at point(x,y) and save on display data array, screen will display when call iot_epaper_display_frame function.

Parameters

- dev: object handle of epaper
- x: point(x,y)
- y: point(x,y)
- radius: radius of the circle
- colored: display color

void **iot_epaper_wait_idle** (*epaper_handle_t dev*)

wait until idle

Parameters

- dev: object handle of epaper

void **iot_epaper_reset** (*epaper_handle_t dev*)
reset device

Parameters

- dev: object handle of epaper

void **iot_epaper_display_frame** (*epaper_handle_t dev*, **const** unsigned char **frame_buffer*)
display frame, refresh screen

Parameters

- dev: object handle of epaper

void **iot_epaper_sleep** (*epaper_handle_t dev*)

After this command is transmitted, the chip would enter the deep-sleep mode to save power. The deep sleep mode would return to standby by hardware reset. The only one parameter is a check code, the command would be executed if check code = 0xA5. You can use `iot_epaper_reset()` to awaken and `EPD_Init()` to initialize.

Parameters

- dev: object handle of epaper

1.3 Structures

```
struct epaper_font_t
struct epaper_paint_t
struct epaper_conf_t
```

1.4 Macros

```
COLORED
UNCOLORED
EPD_WIDTH
EPD_HEIGHT
E_PAPER_DRIVER_OUTPUT_CONTROL
E_PAPER_DEEP_SLEEP_MODE
E_PAPER_SW_RESET
E_PAPER_MASTER_ACTIVATION
E_PAPER_DISPLAY_UPDATE_CONTROL_2
E_PAPER_WRITE_RAM
E_PAPER_WRITE_VCOM_REGISTER
```

```
E_PAPER_WRITE_LUT_REGISTER
E_PAPER_SET_RAM_X_ADDRESS_START_END_POSITION
E_PAPER_SET_RAM_Y_ADDRESS_START_END_POSITION
E_PAPER_SET_RAM_X_ADDRESS_COUNTER
E_PAPER_SET_RAM_Y_ADDRESS_COUNTER
```

1.5 Type Definitions

```
typedef void *epaper_handle_t
```

1.6 Enumerations

```
enum epaper_rotate_t
```

Values:

```
E_PAPER_ROTATE_0
E_PAPER_ROTATE_90
E_PAPER_ROTATE_180
E_PAPER_ROTATE_270
```


C

COLORED (C macro), 6

E

E_PAPER_DEEP_SLEEP_MODE (C macro), 6

E_PAPER_DISPLAY_UPDATE_CONTROL_2 (C macro), 6

E_PAPER_DRIVER_OUTPUT_CONTROL (C macro), 6

E_PAPER_MASTER_ACTIVATION (C macro), 6

E_PAPER_ROTATE_0 (C++ enumerator), 7

E_PAPER_ROTATE_180 (C++ enumerator), 7

E_PAPER_ROTATE_270 (C++ enumerator), 7

E_PAPER_ROTATE_90 (C++ enumerator), 7

E_PAPER_SET_RAM_X_ADDRESS_COUNTER (C macro), 7

E_PAPER_SET_RAM_X_ADDRESS_START_END_POSITION (C macro), 7

E_PAPER_SET_RAM_Y_ADDRESS_COUNTER (C macro), 7

E_PAPER_SET_RAM_Y_ADDRESS_START_END_POSITION (C macro), 7

E_PAPER_SW_RESET (C macro), 6

E_PAPER_WRITE_LUT_REGISTER (C macro), 6

E_PAPER_WRITE_RAM (C macro), 6

E_PAPER_WRITE_VCOM_REGISTER (C macro), 6

epaper_conf_t (C++ class), 6

epaper_font_t (C++ class), 6

epaper_handle_t (C++ type), 7

epaper_paint_t (C++ class), 6

epaper_rotate_t (C++ type), 7

EPD_HEIGHT (C macro), 6

EPD_WIDTH (C macro), 6

I

iot_epaper_clean_paint (C++ function), 2

iot_epaper_create (C++ function), 1

iot_epaper_delete (C++ function), 1

iot_epaper_display_frame (C++ function), 6

iot_epaper_draw_char (C++ function), 3

iot_epaper_draw_circle (C++ function), 5

iot_epaper_draw_filled_circle (C++ function), 5

iot_epaper_draw_filled_rectangle (C++ function), 5

iot_epaper_draw_horizontal_line (C++ function), 4

iot_epaper_draw_line (C++ function), 4

iot_epaper_draw_pixel (C++ function), 3

iot_epaper_draw_rectangle (C++ function), 4

iot_epaper_draw_string (C++ function), 3

iot_epaper_draw_vertical_line (C++ function), 4

iot_epaper_get_height (C++ function), 2

iot_epaper_get_image (C++ function), 3

iot_epaper_get_rotate (C++ function), 2

iot_epaper_get_width (C++ function), 2

iot_epaper_reset (C++ function), 6

iot_epaper_set_height (C++ function), 2

iot_epaper_set_rotate (C++ function), 3

iot_epaper_set_width (C++ function), 2

iot_epaper_sleep (C++ function), 6

iot_epaper_wait_idle (C++ function), 5

UNCOLORED (C macro), 6