

---

# **Read the Docs Template Documentation**

*Release*

**Read the Docs**

**Dec 30, 2017**



---

# Contents

---

|          |                                |          |
|----------|--------------------------------|----------|
| <b>1</b> | <b>Component: epaper-29-ws</b> | <b>1</b> |
| 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        |



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-ws/epaper-29-ws.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 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)
- 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_BOOSTER_SOFT_START_CONTROL
E_PAPER_GATE_SCAN_START_POSITION
E_PAPER_DEEP_SLEEP_MODE
E_PAPER_DATA_ENTRY_MODE_SETTING
E_PAPER_SW_RESET
E_PAPER_TEMPERATURE_SENSOR_CONTROL
```

`E_PAPER_MASTER_ACTIVATION`  
`E_PAPER_DISPLAY_UPDATE_CONTROL_1`  
`E_PAPER_DISPLAY_UPDATE_CONTROL_2`  
`E_PAPER_WRITE_RAM`  
`E_PAPER_WRITE_VCOM_REGISTER`  
`E_PAPER_WRITE_LUT_REGISTER`  
`E_PAPER_SET_DUMMY_LINE_PERIOD`  
`E_PAPER_SET_GATE_TIME`  
`E_PAPER_BORDER_WAVEFORM_CONTROL`  
`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`  
`E_PAPER_TERMINATE_FRAME_READ_WRITE`

## 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\_BOOSTER\_SOFT\_START\_CONTROL (C macro), 6

E\_PAPER\_BORDER\_WAVEFORM\_CONTROL (C macro), 7

E\_PAPER\_DATA\_ENTRY\_MODE\_SETTING (C macro), 6

E\_PAPER\_DEEP\_SLEEP\_MODE (C macro), 6

E\_PAPER\_DISPLAY\_UPDATE\_CONTROL\_1 (C macro), 7

E\_PAPER\_DISPLAY\_UPDATE\_CONTROL\_2 (C macro), 7

E\_PAPER\_DRIVER\_OUTPUT\_CONTROL (C macro), 6

E\_PAPER\_GATE\_SCAN\_START\_POSITION (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\_DUMMY\_LINE\_PERIOD (C macro), 7

E\_PAPER\_SET\_GATE\_TIME (C macro), 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\_TEMPERATURE\_SENSOR\_CONTROL (C macro), 6

E\_PAPER\_TERMINATE\_FRAME\_READ\_WRITE (C macro), 7

E\_PAPER\_WRITE\_LUT\_REGISTER (C macro), 7

E\_PAPER\_WRITE\_RAM (C macro), 7

E\_PAPER\_WRITE\_VCOM\_REGISTER (C macro), 7

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

ior\_epaper\_draw\_filled\_rectangle (C++ function), 5

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\_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

## U

UNCOLORED (C macro), 6