

## PIC32 SDK

Generated by Doxygen 1.8.17



<b>1 Class Index</b>	<b>1</b>
1.1 Class List	1
<b>2 File Index</b>	<b>3</b>
2.1 File List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 p32_uart Struct Reference	5
3.1.1 Member Data Documentation	5
3.1.1.1 brg	5
3.1.1.2 mode	5
3.1.1.3 rxreg	5
3.1.1.4 sta	6
3.1.1.5 txreg	6
<b>4 File Documentation</b>	<b>7</b>
4.1 drivers/cpu.c File Reference	7
4.1.1 Function Documentation	8
4.1.1.1 cpu_clear_interrupt_enable()	8
4.1.1.2 cpu_clear_interrupt_flag()	8
4.1.1.3 cpu_get_interrupt_enable()	8
4.1.1.4 cpu_get_interrupt_flag()	8
4.1.1.5 cpu_get_peripheral_clock()	8
4.1.1.6 cpu_get_system_clock()	8
4.1.1.7 cpu_lock()	9
4.1.1.8 cpu_reset()	9
4.1.1.9 cpu_set_interrupt_enable()	9
4.1.1.10 cpu_set_interrupt_flag()	9
4.1.1.11 cpu_set_interrupt_priority()	9
4.1.1.12 cpu_unlock()	9
4.2 drivers/gpio.c File Reference	10
4.2.1 Macro Definition Documentation	10
4.2.1.1 NUM_PPS_FUNCTIONS	11
4.2.1.2 NUM_PPS_PINS	11
4.2.2 Function Documentation	11
4.2.2.1 gpio_clear_output_function()	11
4.2.2.2 gpio_lock_pps()	11
4.2.2.3 gpio_read()	11
4.2.2.4 gpio_set_input_function()	11
4.2.2.5 gpio_set_mode()	12
4.2.2.6 gpio_set_output_function()	12
4.2.2.7 gpio_unlock_pps()	12
4.2.2.8 gpio_write()	12

4.2.3 Variable Documentation . . . . .	12
4.2.3.1 ppsPinMappingFunctions . . . . .	12
4.2.3.2 ppsPinMappingPins . . . . .	12
4.3 drivers/uart.c File Reference . . . . .	13
4.3.1 Function Documentation . . . . .	13
4.3.1.1 uart_close() . . . . .	13
4.3.1.2 uart_open() . . . . .	14
4.3.1.3 uart_read() . . . . .	14
4.3.1.4 uart_set_baud() . . . . .	14
4.3.1.5 uart_set_format() . . . . .	14
4.3.1.6 uart_set_rx_pin() . . . . .	15
4.3.1.7 uart_set_tx_pin() . . . . .	15
4.3.1.8 uart_write() . . . . .	16
<b>Index</b>	<b>17</b>

# Chapter 1

## Class Index

### 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">p32_uart</a> . . . . .	5
------------------------------------	---



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

drivers/ <a href="#">cpu.c</a>	7
drivers/ <a href="#">gpio.c</a>	10
drivers/ <a href="#">uart.c</a>	13





## Chapter 3

# Class Documentation

### 3.1 p32\_uart Struct Reference

#### Public Attributes

- volatile p32\_regset [mode](#)
- volatile p32\_regset [sta](#)
- volatile p32\_regbuf [txreg](#)
- volatile p32\_regbuf [rxreg](#)
- volatile p32\_regset [brg](#)

#### 3.1.1 Member Data Documentation

##### 3.1.1.1 brg

```
volatile p32_regset p32_uart::brg
```

##### 3.1.1.2 mode

```
volatile p32_regset p32_uart::mode
```

##### 3.1.1.3 rxreg

```
volatile p32_regbuf p32_uart::rxreg
```

#### 3.1.1.4 sta

```
volatile p32_regset p32_uart::sta
```

#### 3.1.1.5 txreg

```
volatile p32_regbuf p32_uart::txreg
```

The documentation for this struct was generated from the following file:

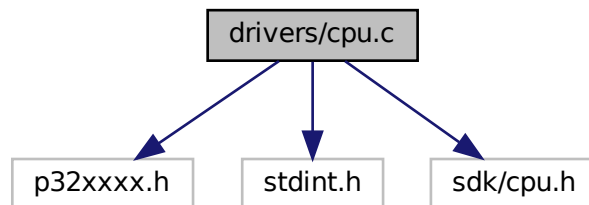
- [drivers/uart.c](#)

## Chapter 4

# File Documentation

### 4.1 drivers/cpu.c File Reference

```
#include <p32xxxx.h>
#include <stdint.h>
#include "sdk/cpu.h"
Include dependency graph for cpu.c:
```



### Functions

- `uint32_t cpu_get_peripheral_clock ()`
- `uint32_t cpu_get_system_clock ()`
- `int cpu_get_interrupt_flag (uint8_t irq)`
- `void cpu_set_interrupt_flag (uint8_t irq)`
- `void cpu_clear_interrupt_flag (uint8_t irq)`
- `void cpu_set_interrupt_enable (uint8_t irq)`
- `void cpu_clear_interrupt_enable (uint8_t irq)`
- `int cpu_get_interrupt_enable (uint8_t irq)`
- `void cpu_set_interrupt_priority (uint8_t vec, uint8_t ipl, uint8_t spl)`
- `void cpu_unlock ()`
- `void cpu_lock ()`
- `void cpu_reset ()`

## 4.1.1 Function Documentation

### 4.1.1.1 `cpu_clear_interrupt_enable()`

```
void cpu_clear_interrupt_enable (
    uint8_t irq )
```

### 4.1.1.2 `cpu_clear_interrupt_flag()`

```
void cpu_clear_interrupt_flag (
    uint8_t irq )
```

### 4.1.1.3 `cpu_get_interrupt_enable()`

```
int cpu_get_interrupt_enable (
    uint8_t irq )
```

### 4.1.1.4 `cpu_get_interrupt_flag()`

```
int cpu_get_interrupt_flag (
    uint8_t irq )
```

### 4.1.1.5 `cpu_get_peripheral_clock()`

```
uint32_t cpu_get_peripheral_clock ( )
```

### 4.1.1.6 `cpu_get_system_clock()`

```
uint32_t cpu_get_system_clock ( )
```

#### 4.1.1.7 `cpu_lock()`

```
void cpu_lock ( )
```

#### 4.1.1.8 `cpu_reset()`

```
void cpu_reset ( )
```

#### 4.1.1.9 `cpu_set_interrupt_enable()`

```
void cpu_set_interrupt_enable (
    uint8_t irq )
```

#### 4.1.1.10 `cpu_set_interrupt_flag()`

```
void cpu_set_interrupt_flag (
    uint8_t irq )
```

#### 4.1.1.11 `cpu_set_interrupt_priority()`

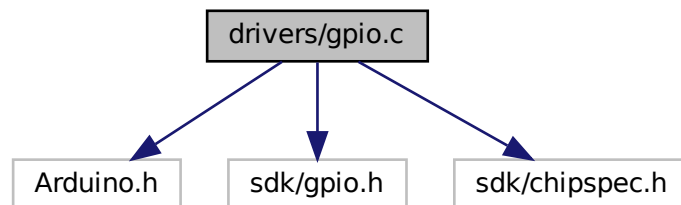
```
void cpu_set_interrupt_priority (
    uint8_t vec,
    uint8_t ipl,
    uint8_t spl )
```

#### 4.1.1.12 `cpu_unlock()`

```
void cpu_unlock ( )
```

## 4.2 drivers/gpio.c File Reference

```
#include <Arduino.h>
#include "sdk/gpio.h"
#include "sdk/chipspec.h"
Include dependency graph for gpio.c:
```



### Classes

- struct **ppsPinMapping**

### Macros

- #define **NUM\_PPS\_PINS** (sizeof([ppsPinMappingPins](#)) / sizeof([ppsPinMappingPins](#)[0]))
- #define **NUM\_PPS\_FUNCTIONS** (sizeof([ppsPinMappingFunctions](#)) / sizeof([ppsPinMappingFunctions](#)[0]))

### Functions

- void [gpio\\_set\\_mode](#) (uint8\_t pin, uint8\_t mode)
- uint8\_t [gpio\\_read](#) (uint8\_t pin)
- void [gpio\\_write](#) (uint8\_t pin, uint8\_t val)
- int [gpio\\_set\\_input\\_function](#) (uint8\_t pin, uint8\_t function)
- int [gpio\\_set\\_output\\_function](#) (uint8\_t pin, uint8\_t function)
- int [gpio\\_clear\\_output\\_function](#) (uint8\_t pin)
- void [gpio\\_unlock\\_pps](#) ()
- void [gpio\\_lock\\_pps](#) ()

### Variables

- const struct ppsPinMapping [ppsPinMappingPins](#) []
- const struct ppsPinMapping [ppsPinMappingFunctions](#) []

#### 4.2.1 Macro Definition Documentation

#### 4.2.1.1 NUM\_PPS\_FUNCTIONS

```
#define NUM_PPS_FUNCTIONS (sizeof(ppsPinMappingFunctions) / sizeof(ppsPinMappingFunctions[0]))
```

#### 4.2.1.2 NUM\_PPS\_PINS

```
#define NUM_PPS_PINS (sizeof(ppsPinMappingPins) / sizeof(ppsPinMappingPins[0]))
```

### 4.2.2 Function Documentation

#### 4.2.2.1 gpio\_clear\_output\_function()

```
int gpio_clear_output_function (
    uint8_t pin )
```

#### 4.2.2.2 gpio\_lock\_pps()

```
void gpio_lock_pps ( )
```

#### 4.2.2.3 gpio\_read()

```
uint8_t gpio_read (
    uint8_t pin )
```

#### 4.2.2.4 gpio\_set\_input\_function()

```
int gpio_set_input_function (
    uint8_t pin,
    uint8_t function )
```

#### 4.2.2.5 gpio\_set\_mode()

```
void gpio_set_mode (
    uint8_t pin,
    uint8_t mode )
```

#### 4.2.2.6 gpio\_set\_output\_function()

```
int gpio_set_output_function (
    uint8_t pin,
    uint8_t function )
```

#### 4.2.2.7 gpio\_unlock\_pps()

```
void gpio_unlock_pps ( )
```

#### 4.2.2.8 gpio\_write()

```
void gpio_write (
    uint8_t pin,
    uint8_t val )
```

### 4.2.3 Variable Documentation

#### 4.2.3.1 ppsPinMappingFunctions

```
const struct ppsPinMapping ppsPinMappingFunctions[]
```

#### 4.2.3.2 ppsPinMappingPins

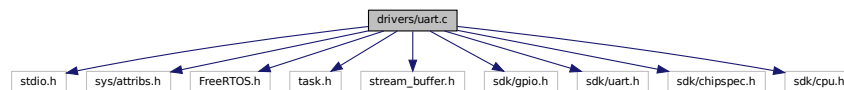
```
const struct ppsPinMapping ppsPinMappingPins[]
```



## 4.3 drivers/uart.c File Reference

```
#include <stdio.h>
#include <sys/attrs.h>
#include "FreeRTOS.h"
#include "task.h"
#include "stream_buffer.h"
#include "sdk/gpio.h"
#include "sdk/uart.h"
#include "sdk/chipspec.h"
#include "sdk/cpu.h"
```

Include dependency graph for uart.c:



### Classes

- struct [p32\\_uart](#)
- struct **uartControlDataStruct**

### Functions

- int [uart\\_write](#) (uint8\_t uart, uint8\_t byte)
- int [uart\\_read](#) (uint8\_t uart)
- int [uart\\_set\\_tx\\_pin](#) (uint8\_t uart, uint8\_t pin)
- int [uart\\_set\\_rx\\_pin](#) (uint8\_t uart, uint8\_t pin)
- int [uart\\_set\\_baud](#) (uint8\_t uart, uint32\_t baud)
- int [uart\\_set\\_format](#) (uint8\_t uart, uint8\_t format)
- int [uart\\_open](#) (uint8\_t uart)
- int [uart\\_close](#) (uint8\_t uart)

#### 4.3.1 Function Documentation

##### 4.3.1.1 [uart\\_close\(\)](#)

```
int uart_close (
    uint8_t uart )
```

#### 4.3.1.2 uart\_open()

```
int uart_open (
    uint8_t uart )
```

#### 4.3.1.3 uart\_read()

```
int uart_read (
    uint8_t uart )
```

#### 4.3.1.4 uart\_set\_baud()

```
int uart_set_baud (
    uint8_t uart,
    uint32_t baud )
```

Configure the baud rate of the selected UART

##### Parameters

<i>uart</i>	The index of the UART
<i>baud</i>	The baud rate to configure.

##### Returns

1 on success, 0 on failure

#### 4.3.1.5 uart\_set\_format()

```
int uart_set_format (
    uint8_t uart,
    uint8_t format )
```

Configure the data format for the selected UART Formats are specified in convenient macros:

- `uart8N1`
- `uart8N2`
- `uart8E1`
- `uart8E2`
- `uart8O1`
- `uart8O2`
- `uart9N1`
- `uart9N2`

## Parameters

<i>uart</i>	The index of the UART
<i>format</i>	The format to use.

## Returns

1 on success, 0 on failure

**4.3.1.6 uart\_set\_rx\_pin()**

```
int uart_set_rx_pin (
    uint8_t uart,
    uint8_t pin )
```

Configure the RX pin of the selected UART through PPS

## Parameters

<i>uart</i>	The index of the UART
<i>pin</i>	The index of the pin to assign the RX function to

## Returns

1 on success, 0 on failure

**4.3.1.7 uart\_set\_tx\_pin()**

```
int uart_set_tx_pin (
    uint8_t uart,
    uint8_t pin )
```

Configure the TX pin of the selected UART through PPS

## Parameters

<i>uart</i>	The index of the UART
<i>pin</i>	The index of the pin to assign the TX function to

## Returns

1 on success, 0 on failure

#### 4.3.1.8 `uart_write()`

```
int uart_write (
    uint8_t uart,
    uint8_t byte )
```

# Index

brg

p32\_uart, 5

cpu.c

cpu\_clear\_interrupt\_enable, 8

cpu\_clear\_interrupt\_flag, 8

cpu\_get\_interrupt\_enable, 8

cpu\_get\_interrupt\_flag, 8

cpu\_get\_peripheral\_clock, 8

cpu\_get\_system\_clock, 8

cpu\_lock, 8

cpu\_reset, 9

cpu\_set\_interrupt\_enable, 9

cpu\_set\_interrupt\_flag, 9

cpu\_set\_interrupt\_priority, 9

cpu\_unlock, 9

cpu\_clear\_interrupt\_enable

cpu.c, 8

cpu\_clear\_interrupt\_flag

cpu.c, 8

cpu\_get\_interrupt\_enable

cpu.c, 8

cpu\_get\_interrupt\_flag

cpu.c, 8

cpu\_get\_peripheral\_clock

cpu.c, 8

cpu\_get\_system\_clock

cpu.c, 8

cpu\_lock

cpu.c, 8

cpu\_reset

cpu.c, 9

cpu\_set\_interrupt\_enable

cpu.c, 9

cpu\_set\_interrupt\_flag

cpu.c, 9

cpu\_set\_interrupt\_priority

cpu.c, 9

cpu\_unlock

cpu.c, 9

drivers/cpu.c, 7

drivers/gpio.c, 10

drivers/uart.c, 13

gpio.c

gpio\_clear\_output\_function, 11

gpio\_lock\_pps, 11

gpio\_read, 11

gpio\_set\_input\_function, 11

gpio\_set\_mode, 11

gpio\_set\_output\_function, 12

gpio\_unlock\_pps, 12

gpio\_write, 12

NUM\_PPS\_FUNCTIONS, 10

NUM\_PPS\_PINS, 11

ppsPinMappingFunctions, 12

ppsPinMappingPins, 12

gpio\_clear\_output\_function

gpio.c, 11

gpio\_lock\_pps

gpio.c, 11

gpio\_read

gpio.c, 11

gpio\_set\_input\_function

gpio.c, 11

gpio\_set\_mode

gpio.c, 11

gpio\_set\_output\_function

gpio.c, 12

gpio\_unlock\_pps

gpio.c, 12

gpio\_write

gpio.c, 12

mode

p32\_uart, 5

NUM\_PPS\_FUNCTIONS

gpio.c, 10

NUM\_PPS\_PINS

gpio.c, 11

p32\_uart, 5

brg, 5

mode, 5

rxreg, 5

sta, 5

txreg, 6

ppsPinMappingFunctions

gpio.c, 12

ppsPinMappingPins

gpio.c, 12

rxreg

p32\_uart, 5

sta

p32\_uart, 5

txreg

p32\_uart, [6](#)

uart.c

uart\_close, [13](#)

uart\_open, [13](#)

uart\_read, [14](#)

uart\_set\_baud, [14](#)

uart\_set\_format, [14](#)

uart\_set\_rx\_pin, [15](#)

uart\_set\_tx\_pin, [15](#)

uart\_write, [15](#)

uart\_close

uart.c, [13](#)

uart\_open

uart.c, [13](#)

uart\_read

uart.c, [14](#)

uart\_set\_baud

uart.c, [14](#)

uart\_set\_format

uart.c, [14](#)

uart\_set\_rx\_pin

uart.c, [15](#)

uart\_set\_tx\_pin

uart.c, [15](#)

uart\_write

uart.c, [15](#)