

A20-gpio

Generated by Doxygen 1.8.9.1

Fri Jun 19 2015 16:30:14



# Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Namespace List . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List . . . . .	7
<b>5</b>	<b>Namespace Documentation</b>	<b>9</b>
5.1	A20 Namespace Reference . . . . .	9
5.1.1	Detailed Description . . . . .	9
5.1.2	Enumeration Type Documentation . . . . .	9
5.1.2.1	pull_resistor_t . . . . .	9
<b>6</b>	<b>Class Documentation</b>	<b>11</b>
6.1	A20::GPIO Class Reference . . . . .	11
6.1.1	Detailed Description . . . . .	11
6.1.2	Member Function Documentation . . . . .	11
6.1.2.1	free . . . . .	11
6.1.2.2	get_input . . . . .	12
6.1.2.3	get_output . . . . .	12
6.1.2.4	init . . . . .	12
6.1.2.5	set_periphery_mode . . . . .	13
6.2	A20::GPIO_common Class Reference . . . . .	13
6.2.1	Detailed Description . . . . .	13
6.2.2	Member Function Documentation . . . . .	14
6.2.2.1	get_port . . . . .	14
6.2.2.2	get_type . . . . .	14
6.3	A20::GPIO_exception Class Reference . . . . .	14

6.3.1	Detailed Description	14
6.3.2	Constructor & Destructor Documentation	14
6.3.2.1	GPIO_exception	14
6.4	A20::GPIO_input Class Reference	15
6.4.1	Detailed Description	15
6.4.2	Member Function Documentation	15
6.4.2.1	get	15
6.4.2.2	get_port	15
6.4.2.3	get_type	16
6.4.3	Friends And Related Function Documentation	16
6.4.3.1	GPIO	16
6.5	A20::GPIO_output Class Reference	16
6.5.1	Detailed Description	16
6.5.2	Member Function Documentation	17
6.5.2.1	get_port	17
6.5.2.2	get_type	17
6.5.2.3	set	17
6.5.3	Friends And Related Function Documentation	17
6.5.3.1	GPIO	17
6.6	A20::GPIO_periphery Class Reference	17
6.6.1	Detailed Description	18
6.6.2	Member Function Documentation	18
6.6.2.1	get_port	18
6.6.2.2	get_type	18
6.6.3	Friends And Related Function Documentation	18
6.6.3.1	GPIO	18
6.7	sunxi_gpio Struct Reference	18
6.7.1	Member Data Documentation	19
6.7.1.1	cfg	19
6.7.1.2	dat	19
6.7.1.3	drv	19
6.7.1.4	pull	19
6.8	sunxi_gpio_int Struct Reference	19
6.8.1	Member Data Documentation	19
6.8.1.1	cfg	19
6.8.1.2	ctl	19
6.8.1.3	deb	19
6.8.1.4	sta	19
6.9	sunxi_gpio_reg Struct Reference	19
6.9.1	Member Data Documentation	20

6.9.1.1	<a href="#">gpio_bank</a>	20
6.9.1.2	<a href="#">gpio_int</a>	20
6.9.1.3	<a href="#">res</a>	20
<b>7</b>	<b>File Documentation</b>	<b>21</b>
7.1	<a href="#">A20-gpio.cpp File Reference</a>	21
7.2	<a href="#">A20-gpio.hpp File Reference</a>	21
7.2.1	<a href="#">Typedef Documentation</a>	22
7.2.1.1	<a href="#">uint16_t</a>	22
7.2.1.2	<a href="#">uint8_t</a>	22
7.3	<a href="#">examples/ex.cpp File Reference</a>	22
7.3.1	<a href="#">Function Documentation</a>	22
7.3.1.1	<a href="#">main</a>	22
7.4	<a href="#">gpio_lib.c File Reference</a>	22
7.4.1	<a href="#">Function Documentation</a>	23
7.4.1.1	<a href="#">sunxi_gpio_get_cfgpin</a>	23
7.4.1.2	<a href="#">sunxi_gpio_init</a>	23
7.4.1.3	<a href="#">sunxi_gpio_input</a>	23
7.4.1.4	<a href="#">sunxi_gpio_output</a>	23
7.4.1.5	<a href="#">sunxi_gpio_pullup</a>	23
7.4.1.6	<a href="#">sunxi_gpio_set_cfgpin</a>	23
7.4.2	<a href="#">Variable Documentation</a>	23
7.4.2.1	<a href="#">SUNXI_PIO_BASE</a>	23
7.5	<a href="#">gpio_lib.h File Reference</a>	23
7.5.1	<a href="#">Macro Definition Documentation</a>	25
7.5.1.1	<a href="#">GPIO_BANK</a>	25
7.5.1.2	<a href="#">GPIO_CFG_INDEX</a>	25
7.5.1.3	<a href="#">GPIO_CFG_OFFSET</a>	25
7.5.1.4	<a href="#">GPIO_NUM</a>	25
7.5.1.5	<a href="#">GPIO_PUL_INDEX</a>	25
7.5.1.6	<a href="#">GPIO_PUL_OFFSET</a>	25
7.5.1.7	<a href="#">SUNXI_GPA</a>	25
7.5.1.8	<a href="#">SUNXI_GPB</a>	25
7.5.1.9	<a href="#">SUNXI_GPC</a>	25
7.5.1.10	<a href="#">SUNXI_GPD</a>	25
7.5.1.11	<a href="#">SUNXI_GPE</a>	25
7.5.1.12	<a href="#">SUNXI_GPF</a>	25
7.5.1.13	<a href="#">SUNXI_GPG</a>	25
7.5.1.14	<a href="#">SUNXI_GPH</a>	25
7.5.1.15	<a href="#">SUNXI_GPI</a>	25

7.5.1.16	SUNXI_GPIO_A	25
7.5.1.17	SUNXI_GPIO_A_NR	25
7.5.1.18	SUNXI_GPIO_B	25
7.5.1.19	SUNXI_GPIO_B_NR	25
7.5.1.20	SUNXI_GPIO_C	25
7.5.1.21	SUNXI_GPIO_C_NR	25
7.5.1.22	SUNXI_GPIO_D	25
7.5.1.23	SUNXI_GPIO_D_NR	25
7.5.1.24	SUNXI_GPIO_E	25
7.5.1.25	SUNXI_GPIO_E_NR	25
7.5.1.26	SUNXI_GPIO_F	26
7.5.1.27	SUNXI_GPIO_F_NR	26
7.5.1.28	SUNXI_GPIO_G	26
7.5.1.29	SUNXI_GPIO_G_NR	26
7.5.1.30	SUNXI_GPIO_H	26
7.5.1.31	SUNXI_GPIO_H_NR	26
7.5.1.32	SUNXI_GPIO_I	26
7.5.1.33	SUNXI_GPIO_I_NR	26
7.5.1.34	SUNXI_GPIO_INPUT	26
7.5.1.35	SUNXI_GPIO_NEXT	26
7.5.1.36	SUNXI_GPIO_OUTPUT	26
7.5.1.37	SUNXI_GPIO_PER	26
7.5.1.38	SUNXI_PULL_DOWN	26
7.5.1.39	SUNXI_PULL_NONE	26
7.5.1.40	SUNXI_PULL_UP	26
7.5.1.41	SW_PORTC_IO_BASE	26
7.5.2	Enumeration Type Documentation	26
7.5.2.1	sunxi_gpio_number	26
7.5.3	Function Documentation	26
7.5.3.1	sunxi_gpio_get_cfgpin	26
7.5.3.2	sunxi_gpio_init	26
7.5.3.3	sunxi_gpio_input	27
7.5.3.4	sunxi_gpio_output	27
7.5.3.5	sunxi_gpio_pullup	27
7.5.3.6	sunxi_gpio_set_cfgpin	27
7.5.4	Variable Documentation	27
7.5.4.1	SUNXI_PIO_BASE	27

# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">A20</a> . . . . .	9
-------------------------------	---





## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

A20::GPIO . . . . .	11
A20::GPIO_common . . . . .	13
A20::GPIO_input . . . . .	15
A20::GPIO_output . . . . .	16
A20::GPIO_periphery . . . . .	17
runtime_error	
A20::GPIO_exception . . . . .	14



## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">A20::GPIO</a>	The main class of the module. First of all, you need to call the <a href="#">init()</a> method, to initialize the sunxi driver. After that you can obtain <a href="#">GPIO</a> via <a href="#">get_input</a> , <a href="#">get_output</a> and <a href="#">set_periphery_mode</a>	11
<a href="#">A20::GPIO_common</a>	An abstract class for <a href="#">GPIO_input</a> and <a href="#">GPIO_output</a> . Probably, you don't have to use this (internal library use)	13
<a href="#">A20::GPIO_exception</a>	A generic exception for all module	14
<a href="#">A20::GPIO_input</a>	The class that represents a <a href="#">GPIO</a> input. You cannot instantiate this class directly, but you need to call <a href="#">GPIO::get_input</a> to have a pointer of this class. Also, you cannot free the pointer, but you need to use <a href="#">GPIO::free</a>	15
<a href="#">A20::GPIO_output</a>	The class that represents a <a href="#">GPIO</a> output. You cannot instantiate this class directly, but you need to call <a href="#">GPIO::get_output</a> to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use <a href="#">GPIO::free</a>	16
<a href="#">A20::GPIO_periphery</a>	The class that represents a <a href="#">GPIO</a> periphery. You cannot instantiate this class directly, but you need to call <a href="#">GPIO::set_periphery_mode</a> to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use <a href="#">GPIO::free</a> . Currently, you cannot do anything with this type of <a href="#">GPIO</a>	17



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

<a href="#">A20-gpio.cpp</a>	21
<a href="#">A20-gpio.hpp</a>	21
<a href="#">examples/ex.cpp</a>	22



## Chapter 5

# Namespace Documentation

### 5.1 A20 Namespace Reference

#### Classes

- class [GPIO](#)

*The main class of the module. First of all, you need to call the [init\(\)](#) method, to initialize the sunxi driver. After that you can obtain [GPIO](#) via [get\\_input](#), [get\\_output](#) and [set\\_periphery\\_mode](#).*

- class [GPIO\\_common](#)

*An abstract class for [GPIO\\_input](#) and [GPIO\\_output](#). Probably, you don't have to use this (internal library use).*

- class [GPIO\\_exception](#)

*A generic exception for all module.*

- class [GPIO\\_input](#)

*The class that represents a [GPIO](#) input. You cannot instantiate this class directly, but you need to call [GPIO::get\\_input](#) to have a pointer of this class. Also, you cannot free the pointer, but you need to use [GPIO::free](#).*

- class [GPIO\\_output](#)

*The class that represents a [GPIO](#) output. You cannot instantiate this class directly, but you need to call [GPIO::get\\_output](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#).*

- class [GPIO\\_periphery](#)

*The class that represents a [GPIO](#) periphery. You cannot instantiate this class directly, but you need to call [GPIO::set\\_periphery\\_mode](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#). Currently, you cannot do anything with this type of [GPIO](#).*

#### Enumerations

- enum [pull\\_resistor\\_t](#) { [NONE](#) =0, [PULL\\_UP](#) =1, [PULL\\_DOWN](#) =2 }

*The enumeration for pullup/pulldown resistors. The values match sunxi values for pull resistors.*

#### 5.1.1 Detailed Description

The namespace for [A20](#)

#### 5.1.2 Enumeration Type Documentation

##### 5.1.2.1 enum [A20::pull\\_resistor\\_t](#)

The enumeration for pullup/pulldown resistors. The values match sunxi values for pull resistors.

## Enumerator

***NONE***  
***PULL\_UP***  
***PULL\_DOWN***



# Chapter 6

## Class Documentation

### 6.1 A20::GPIO Class Reference

The main class of the module. First of all, you need to call the `init()` method, to initialize the sunxi driver. After that you can obtain `GPIO` via `get_input`, `get_output` and `set_periphery_mode`.

```
#include <A20-gpio.hpp>
```

#### Static Public Member Functions

- static void `init ()` throw (`GPIO_exception`)

*Initialize the submodule sunxi. You have to call this before any other method.*

*Exceptions*

<code>GPIO_exception</code>	<i>In case of initialization error.</i>
-----------------------------	-----------------------------------------

- static `GPIO_input * get_input (uint16_t port, pull_resistor_t pull_resistor=NONE)` throw (`GPIO_exception`)  
*Set the `GPIO` indicated as input, enable the eventually specified pull resistor, and returns the corresponding object to read data. If you want to change the type of `GPIO`, you must free the previous object via `GPIO::free`.*
- static `GPIO_output * get_output (uint16_t port)` throw (`GPIO_exception`)  
*Set the `GPIO` indicated as output and returns the corresponding object to write data. If you want to change the type of `GPIO`, you must free the previous object via `GPIO::free`.*
- static void `set_periphery_mode (uint16_t port)` throw (`GPIO_exception`)  
*Set the `GPIO` indicated as periphery and returns the corresponding object. If you want to change the type of `GPIO`, you must free the previous object via `GPIO::free`.*
- static void `free (uint16_t port)` throw (`GPIO_exception`)  
*Free the corresponding object.*

#### 6.1.1 Detailed Description

The main class of the module. First of all, you need to call the `init()` method, to initialize the sunxi driver. After that you can obtain `GPIO` via `get_input`, `get_output` and `set_periphery_mode`.

##### Note

This is a static class, you cannot (and you don't need to) initialize an object of this type.

#### 6.1.2 Member Function Documentation

##### 6.1.2.1 void A20::GPIO::free ( uint16\_t port ) throw GPIO\_exception) [static]

Free the corresponding object.

## Parameters

<i>port</i>	the number of <a href="#">GPIO</a> to free according to kernel enumeration (see datasheet)
-------------	--------------------------------------------------------------------------------------------

## Note

If the object you ask to free is an output [GPIO](#), this method transform the output in input for precaution. If you don't want this behaviour, define a constant named `GPIO_NO_SAFE_RESET_TO_INPUT`

## Exceptions

<a href="#">GPIO_exception</a>	In case of generic error or if you are trying to free a non setted <a href="#">GPIO</a> .
--------------------------------	-------------------------------------------------------------------------------------------

#### 6.1.2.2 `GPIO_input * A20::GPIO::get_input ( uint16_t port, pull_resistor_t pull_resistor = NONE ) throw GPIO_exception) [static]`

Set the [GPIO](#) indicated as input, enable the eventually specified pull resistor, and returns the corresponding object to read data. If you want to change the type of [GPIO](#), you must free the previous object via [GPIO::free](#).

## Parameters

<i>port</i>	the number of <a href="#">GPIO</a> to set as input according to kernel enumeration (see datasheet)
<i>pull_resistor</i>	specify the type of pull resistor to enable in this <a href="#">GPIO</a> . Refer to <code>pull_resistor_t</code> for available values. By default no pull resistor is enabled.

## Exceptions

<a href="#">GPIO_exception</a>	In case of configuration error or if you are trying to set as input a non freed <a href="#">GPIO</a> (for example because you create a <a href="#">GPIO_output</a> on the same <a href="#">GPIO</a> ).
--------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Note

Before you can read from this gpio, you need to wait some microseconds. We already put in this method a sleep of 100us. If you don't want this behaviour, define a constant named `GPIO_NO_WAIT`.

#### 6.1.2.3 `GPIO_output * A20::GPIO::get_output ( uint16_t port ) throw GPIO_exception) [static]`

Set the [GPIO](#) indicated as output and returns the corresponding object to write data. If you want to change the type of [GPIO](#), you must free the previous object via [GPIO::free](#).

## Parameters

<i>port</i>	the number of <a href="#">GPIO</a> to set as output according to kernel enumeration (see datasheet)
-------------	-----------------------------------------------------------------------------------------------------

## Exceptions

<a href="#">GPIO_exception</a>	In case of configuration error or if you are trying to set as output a non freed <a href="#">GPIO</a> (for example because you create a <a href="#">GPIO_input</a> on the same <a href="#">GPIO</a> ).
--------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Note

Before you can write to this gpio, you need to wait some microseconds. We already put in this method a sleep of 100us. If you don't want this behaviour, define a constant named `GPIO_NO_WAIT`.

#### 6.1.2.4 `void A20::GPIO::init ( ) throw GPIO_exception) [static]`

Initialize the submodule sunxi. You have to call this before any other method.

## Exceptions

<a href="#">GPIO_exception</a>	In case of initialization error.
--------------------------------	----------------------------------

6.1.2.5 void A20::GPIO::set\_periphery\_mode ( uint16\_t port ) throw GPIO\_exception) [static]

Set the [GPIO](#) indicated as periphery and returns the corresponding object. If you want to change the type of [GPIO](#), you must free the previous object via [GPIO::free](#).

## Parameters

<i>port</i>	the number of <a href="#">GPIO</a> to set as periphery according to kernel enumeration (see datasheet)
-------------	--------------------------------------------------------------------------------------------------------

## Exceptions

<a href="#">GPIO_exception</a>	In case of configuration error or if you are trying to set as periphery a non freed <a href="#">GPIO</a> (for example because you create a <a href="#">GPIO_input</a> on the same <a href="#">GPIO</a> ).
--------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Note

Before you can use this gpio, you need to wait some microseconds. We already put in this method a sleep of 100us. If you don't want this behaviour, define a constant named GPIO\_NO\_WAIT.

The documentation for this class was generated from the following files:

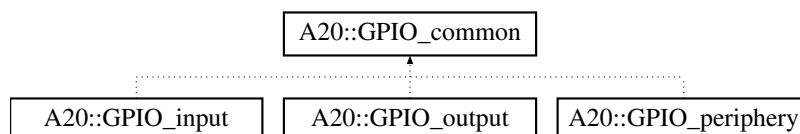
- [A20-gpio.hpp](#)
- [A20-gpio.cpp](#)

## 6.2 A20::GPIO\_common Class Reference

An abstract class for [GPIO\\_input](#) and [GPIO\\_output](#). Probably, you don't have to use this (internal library use).

```
#include <A20-gpio.hpp>
```

Inheritance diagram for A20::GPIO\_common:



### Public Member Functions

- virtual [uint8\\_t get\\_type](#) () const =0  
*It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).*
- virtual [uint16\\_t get\\_port](#) () const =0  
*It returns the current port number as requested in constructor.*

### 6.2.1 Detailed Description

An abstract class for [GPIO\\_input](#) and [GPIO\\_output](#). Probably, you don't have to use this (internal library use).

## 6.2.2 Member Function Documentation

### 6.2.2.1 `virtual uint16_t A20::GPIO_common::get_port( ) const` `[pure virtual]`

It returns the current port number as requested in constructor.

#### Returns

the port number according to linux enumeration (see datasheet)

Implemented in [A20::GPIO\\_periphery](#), [A20::GPIO\\_output](#), and [A20::GPIO\\_input](#).

### 6.2.2.2 `virtual uint8_t A20::GPIO_common::get_type( ) const` `[pure virtual]`

It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).

#### Returns

0 if input, 1 if output, 2 if periphery

Implemented in [A20::GPIO\\_periphery](#), [A20::GPIO\\_output](#), and [A20::GPIO\\_input](#).

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

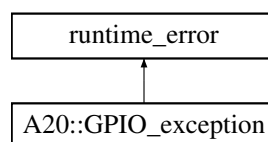
- [A20-gpio.hpp](#)

## 6.3 A20::GPIO\_exception Class Reference

A generic exception for all module.

```
#include <A20-gpio.hpp>
```

Inheritance diagram for A20::GPIO\_exception:



### Public Member Functions

- [GPIO\\_exception](#) (std::string s)

### 6.3.1 Detailed Description

A generic exception for all module.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 `A20::GPIO_exception::GPIO_exception( std::string s )` `[inline]`

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

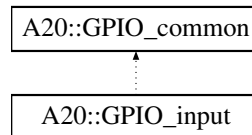
- [A20-gpio.hpp](#)

## 6.4 A20::GPIO\_input Class Reference

The class that represents a [GPIO](#) input. You cannot instantiate this class directly, but you need to call [GPIO::get↵\\_input](#) to have a pointer of this class. Also, you cannot free the pointer, but you need to use [GPIO::free](#).

```
#include <A20-gpio.hpp>
```

Inheritance diagram for A20::GPIO\_input:



### Public Member Functions

- [uint8\\_t get\\_type](#) () const  
*It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).*
- [uint16\\_t get\\_port](#) () const  
*It returns the current port number as requested in constructor.*
- [bool get](#) () const  
*It returns the current value of the [GPIO](#).*

### Friends

- class [GPIO](#)

#### 6.4.1 Detailed Description

The class that represents a [GPIO](#) input. You cannot instantiate this class directly, but you need to call [GPIO::get↵\\_input](#) to have a pointer of this class. Also, you cannot free the pointer, but you need to use [GPIO::free](#).

#### 6.4.2 Member Function Documentation

##### 6.4.2.1 bool A20::GPIO\_input::get ( ) const

It returns the current value of the [GPIO](#).

##### Returns

true if it reads a logical '1' on the [GPIO](#), false otherwise.

##### 6.4.2.2 uint16\_t A20::GPIO\_input::get\_port ( ) const [inline], [virtual]

It returns the current port number as requested in constructor.

##### Returns

the port number according to linux enumeration (see datasheet)

Implements [A20::GPIO\\_common](#).

#### 6.4.2.3 `uint8_t A20::GPIO_input::get_type ( ) const` `[inline],[virtual]`

It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).

##### Returns

0 if input, 1 if output, 2 if periphery

Implements [A20::GPIO\\_common](#).

### 6.4.3 Friends And Related Function Documentation

#### 6.4.3.1 `friend class GPIO` `[friend]`

The documentation for this class was generated from the following files:

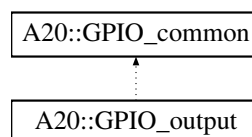
- [A20-gpio.hpp](#)
- [A20-gpio.cpp](#)

## 6.5 A20::GPIO\_output Class Reference

The class that represents a [GPIO](#) output. You cannot instantiate this class directly, but you need to call [GPIO↔O::get\\_output](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#).

```
#include <A20-gpio.hpp>
```

Inheritance diagram for A20::GPIO\_output:



### Public Member Functions

- `uint8_t get_type ( ) const`  
*It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).*
- `uint16_t get_port ( ) const`  
*It returns the current port number as requested in constructor.*
- `void set (bool) const`  
*Set the logical output of a [GPIO](#).*

### Friends

- class [GPIO](#)

#### 6.5.1 Detailed Description

The class that represents a [GPIO](#) output. You cannot instantiate this class directly, but you need to call [GPIO↔O::get\\_output](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#).

## 6.5.2 Member Function Documentation

### 6.5.2.1 uint16\_t A20::GPIO\_output::get\_port( ) const [inline], [virtual]

It returns the current port number as requested in constructor.

Returns

the port number according to linux enumeration (see datasheet)

Implements [A20::GPIO\\_common](#).

### 6.5.2.2 uint8\_t A20::GPIO\_output::get\_type( ) const [inline], [virtual]

It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).

Returns

0 if input, 1 if output, 2 if periphery

Implements [A20::GPIO\\_common](#).

### 6.5.2.3 void A20::GPIO\_output::set( bool value ) const

Set the logical output of a [GPIO](#).

Parameters

<i>value</i>	if true it puts logical '1' on the <a href="#">GPIO</a> , '0' otherwise.
--------------	--------------------------------------------------------------------------

## 6.5.3 Friends And Related Function Documentation

### 6.5.3.1 friend class GPIO [friend]

The documentation for this class was generated from the following files:

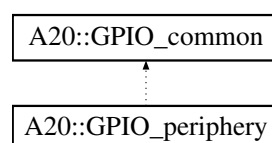
- [A20-gpio.hpp](#)
- [A20-gpio.cpp](#)

## 6.6 A20::GPIO\_periphery Class Reference

The class that represents a [GPIO](#) periphery. You cannot instantiate this class directly, but you need to call [GPIO::set\\_periphery\\_mode](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#). Currently, you cannot do anything with this type of [GPIO](#).

```
#include <A20-gpio.hpp>
```

Inheritance diagram for A20::GPIO\_periphery:



## Public Member Functions

- [uint8\\_t get\\_type](#) ( ) const  
*It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).*
- [uint16\\_t get\\_port](#) ( ) const  
*It returns the current port number as requested in constructor.*

## Friends

- class [GPIO](#)

### 6.6.1 Detailed Description

The class that represents a [GPIO](#) periphery. You cannot instantiate this class directly, but you need to call [GPIO::set\\_periphery\\_mode](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#). Currently, you cannot do anything with this type of [GPIO](#).

### 6.6.2 Member Function Documentation

#### 6.6.2.1 [uint16\\_t A20::GPIO\\_periphery::get\\_port](#) ( ) const `[inline], [virtual]`

It returns the current port number as requested in constructor.

##### Returns

the port number according to linux enumeration (see datasheet)

Implements [A20::GPIO\\_common](#).

#### 6.6.2.2 [uint8\\_t A20::GPIO\\_periphery::get\\_type](#) ( ) const `[inline], [virtual]`

It returns the type of [GPIO](#) (input, output or periphery). Probably, you don't have to use this (internal library use).

##### Returns

0 if input, 1 if output, 2 if periphery

Implements [A20::GPIO\\_common](#).

### 6.6.3 Friends And Related Function Documentation

#### 6.6.3.1 `friend class GPIO` `[friend]`

The documentation for this class was generated from the following files:

- [A20-gpio.hpp](#)
- [A20-gpio.cpp](#)



## Chapter 7

# File Documentation

### 7.1 A20-gpio.cpp File Reference

```
#include "A20-gpio.hpp"
#include "gpio_lib.h"
#include <unistd.h>
```

#### Namespaces

- [A20](#)

### 7.2 A20-gpio.hpp File Reference

```
#include <vector>
#include <stdexcept>
#include <string>
```

#### Classes

- class [A20::GPIO\\_exception](#)  
*A generic exception for all module.*
- class [A20::GPIO\\_common](#)  
*An abstract class for [GPIO\\_input](#) and [GPIO\\_output](#). Probably, you don't have to use this (internal library use).*
- class [A20::GPIO\\_input](#)  
*The class that represents a [GPIO](#) input. You cannot instantiate this class directly, but you need to call [GPIO::get\\_input](#) to have a pointer of this class. Also, you cannot free the pointer, but you need to use [GPIO::free](#).*
- class [A20::GPIO\\_output](#)  
*The class that represents a [GPIO](#) output. You cannot instantiate this class directly, but you need to call [GPIO::get\\_output](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#).*
- class [A20::GPIO\\_periphery](#)  
*The class that represents a [GPIO](#) periphery. You cannot instantiate this class directly, but you need to call [GPIO::set\\_periphery\\_mode](#) to have a pointer of an object of this type. Also, you cannot free the pointer, but you need to use [GPIO::free](#). Currently, you cannot do anything with this type of [GPIO](#).*
- class [A20::GPIO](#)  
*The main class of the module. First of all, you need to call the [init\(\)](#) method, to initialize the sunxi driver. After that you can obtain [GPIO](#) via [get\\_input](#), [get\\_output](#) and [set\\_periphery\\_mode](#).*

## Namespaces

- [A20](#)

## Typedefs

- typedef unsigned char [uint8\\_t](#)
- typedef unsigned short [uint16\\_t](#)

## Enumerations

- enum [A20::pull\\_resistor\\_t](#) { [A20::NONE](#) =0, [A20::PULL\\_UP](#) =1, [A20::PULL\\_DOWN](#) =2 }
- The enumeration for pullup/pulldown resistors. The values match sunxi values for pull resistors.*

### 7.2.1 Typedef Documentation

#### 7.2.1.1 typedef unsigned short [uint16\\_t](#)

#### 7.2.1.2 typedef unsigned char [uint8\\_t](#)

## 7.3 examples/ex.cpp File Reference

```
#include "../A20-gpio.hpp"
#include <assert.h>
#include <iostream>
```

## Functions

- int [main](#) ()

### 7.3.1 Function Documentation

#### 7.3.1.1 int [main](#) ( )