

## SWITCH\_driver

Generated by Doxygen 1.8.17



---

<b>1 Data Structure Index</b>	<b>1</b>
1.1 Data Structures . . . . .	1
<b>2 File Index</b>	<b>3</b>
2.1 File List . . . . .	3
<b>3 Data Structure Documentation</b>	<b>5</b>
3.1 switchmap_t Struct Reference . . . . .	5
<b>4 File Documentation</b>	<b>7</b>
4.1 D:/Technical/ITI/Embedded Computer Architecture/Tasks/CodeDocs/SWITCH.h File Reference . . .	7
4.1.1 Detailed Description . . . . .	8
4.1.2 Function Documentation . . . . .	8
4.1.2.1 Switch_GetSwitchState() . . . . .	8
4.1.2.2 switchTask() . . . . .	8
4.1.2.3 SwitchTask_GetSwitchState() . . . . .	10
4.1.2.4 SwitchTask_Init() . . . . .	10
<b>Index</b>	<b>11</b>



# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

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



## Chapter 2

# File Index

### 2.1 File List

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

D:/Technical/ITI/Embedded Computer Architecture/Tasks/CodeDocs/ <a href="#">SWITCH.h</a>	
This file is the SWITCH driver . . . . .	<a href="#">7</a>





## Chapter 3

# Data Structure Documentation

### 3.1 switchmap\_t Struct Reference

#### Data Fields

- u32 **pin**
- void \* **port**
- u32 **pullState**

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

- D:/Technical/ITI/Embedded Computer Architecture/Tasks/CodeDocs/[SWITCH.h](#)



## Chapter 4

# File Documentation

### 4.1 D:/Technical/ITI/Embedded Computer Architecture/Tasks/CodeDocs/SWITCH.h File Reference

This file is the SWITCH driver.

#### Data Structures

- struct [switchmap\\_t](#)

#### Macros

- #define **PULL\_UP** 1
- #define **PULL\_DOWN** 2
- #define **PRESSED** 1
- #define **RELEASED** 0
- #define **MAX\_COUNTS** 5

#### Functions

- ERROR\_STATUS [SwitchTask\\_Init](#) (u32 switchNum)  
*This function shall initialize specific switch by setting its pin, port, mode and configuration in a GPIO object and passing it to GPIO module.*
- ERROR\_STATUS [Switch\\_GetSwitchState](#) (u32 switchNum, u8 \*switchValue)  
*This function shall return the specified switch state which can be PRESSED or RELEASED.*
- ERROR\_STATUS [SwitchTask\\_GetSwitchState](#) (u32 switchNum, u8 \*switchValue)  
*This function shall return the specified switch state which can be PRESSED or RELEASED. Used with scheduler.*
- void [switchTask](#) (void)  
*This function represents the scheduler task to invoke switch.*

### 4.1.1 Detailed Description

This file is the SWITCH driver.

#### Author

Alzahraa Elsallakh ( [zahraaelsallakh@gmail.com](mailto:zahraaelsallakh@gmail.com))

#### Version

1.0

#### Date

2020-02-07

#### Copyright

Copyright (c) 2020

### 4.1.2 Function Documentation

#### 4.1.2.1 Switch\_GetSwitchState()

```
ERROR_STATUS Switch_GetSwitchState (
    u32 switchNum,
    u8 * switchValue )
```

This function shall return the specified switch state which can be PRESSED or RELEASED.

#### Parameters

<i>switchNum</i>	holds the index of the switch in the switch array
<i>switchValue</i>	Pointer to hold the switch value

#### Returns

ERROR\_STATUS

status\_Ok : If reading the switch state goes correctly

status\_Nok : If any error occurred during reading

#### 4.1.2.2 switchTask()

```
void switchTask (
    void )
```

This function represents the scheduler task to invoke switch.

**Parameters**

<i>void</i>	
-------------	--

**Returns**

void

**4.1.2.3 SwitchTask\_GetSwitchState()**

```
ERROR_STATUS SwitchTask_GetSwitchState (
    u32 switchNum,
    u8 * switchValue )
```

This function shall return the specified switch state which can be PRESSED or RELEASED. Used with scheduler.

**Parameters**

<i>switchNum</i>	holds the index of the switch in the switch array
<i>switchValue</i>	Pointer to hold the switch value

**Returns**

ERROR\_STATUS  
status\_Ok : If reading the switch state goes correctly  
status\_Nok : If any error occurred during reading

**4.1.2.4 SwitchTask\_Init()**

```
ERROR_STATUS SwitchTask_Init (
    u32 switchNum )
```

This function shall initialize specific switch by setting its pin, port, mode and configuration in a GPIO object and passing it to GPIO module.

**Parameters**

<i>switchNum</i>	holds the index of the switch in the switch array
------------------	---

**Returns**

ERROR\_STATUS  
status\_Ok : If the initialization is done successfully  
status\_Nok : If any error occurred during initialization

# Index

D:/Technical/ITI/Embedded Computer Architecture/Tasks/CodeDocs/SWITCH.h, [7](#)

SWITCH.h

Switch\_GetSwitchState, [8](#)

switchTask, [8](#)

SwitchTask\_GetSwitchState, [10](#)

SwitchTask\_Init, [10](#)

Switch\_GetSwitchState

SWITCH.h, [8](#)

switchmap\_t, [5](#)

switchTask

SWITCH.h, [8](#)

SwitchTask\_GetSwitchState

SWITCH.h, [10](#)

SwitchTask\_Init

SWITCH.h, [10](#)