

wlan_stationdocumentation

Generated by Doxygen 1.8.8

Thu Oct 2 2014 21:20:09

Contents

1	Module Index	1
1.1	Modules	1
2	File Index	3
2.1	File List	3
3	Module Documentation	5
3.1	Getting_started_sta	5
3.1.1	Detailed Description	6
3.1.2	Macro Definition Documentation	6
3.1.2.1	APPLICATION_NAME	6
3.1.2.2	APPLICATION_VERSION	6
3.1.2.3	HOST_NAME	6
3.1.2.4	NO_OF_ATTEMPTS	6
3.1.2.5	OSI_STACK_SIZE	6
3.1.2.6	PING_INTERVAL	6
3.1.2.7	PING_PKT_SIZE	6
3.1.2.8	PING_TIMEOUT	6
3.1.3	Enumeration Type Documentation	7
3.1.3.1	e_AppStatusCodes	7
3.1.4	Function Documentation	7
3.1.4.1	main	7
3.1.4.2	SimpleLinkGeneralEventHandler	8
3.1.4.3	SimpleLinkHttpServerCallback	9
3.1.4.4	SimpleLinkNetAppEventHandler	9
3.1.4.5	SimpleLinkSockEventHandler	9
3.1.4.6	SimpleLinkWlanEventHandler	10
3.1.4.7	WlanStationMode	11
3.1.5	Variable Documentation	11
3.1.5.1	g_ucConnectionBSSID	12
3.1.5.2	g_ucConnectionSSID	12
3.1.5.3	g_ulGatewayIP	12

3.1.5.4	g_ulPingPacketsRecv	12
3.1.5.5	g_ulStatus	12
4	File Documentation	13
4.1	common/common.h File Reference	13
4.1.1	Macro Definition Documentation	14
4.1.1.1	ASSERT_ON_ERROR	14
4.1.1.2	BSSID_LEN_MAX	14
4.1.1.3	CLR_STATUS_BIT	14
4.1.1.4	CLR_STATUS_BIT_ALL	14
4.1.1.5	CLR_STATUS_BIT_ALL	14
4.1.1.6	DBG_PRINT	15
4.1.1.7	ERR_PRINT	15
4.1.1.8	FAILURE	15
4.1.1.9	GET_STATUS_BIT	15
4.1.1.10	IS_CONNECT_FAILED	15
4.1.1.11	IS_CONNECTED	15
4.1.1.12	IS_IP_ACQUIRED	15
4.1.1.13	IS_IP_LEASED	15
4.1.1.14	IS_NW_PROCSR_ON	16
4.1.1.15	IS_P2P_DEV_FOUND	16
4.1.1.16	IS_P2P_REQ_RCVD	16
4.1.1.17	IS_PING_DONE	16
4.1.1.18	IS_SMART_CFG_START	16
4.1.1.19	LOOP_FOREVER	16
4.1.1.20	SECURITY_KEY	17
4.1.1.21	SECURITY_TYPE	17
4.1.1.22	SET_STATUS_BIT	17
4.1.1.23	SL_STOP_TIMEOUT	17
4.1.1.24	SPAWN_TASK_PRIORITY	17
4.1.1.25	SSID_LEN_MAX	17
4.1.1.26	SSID_NAME	17
4.1.1.27	SUCCESS	17
4.1.1.28	UART_PRINT	17
4.1.1.29	UNUSED	17
4.1.2	Enumeration Type Documentation	18
4.1.2.1	e_StatusBits	18
4.2	common/gpio_if.c File Reference	18
4.2.1	Macro Definition Documentation	19
4.2.1.1	PIN_LED1	19

4.2.1.2	PIN_LED2	19
4.2.1.3	PIN_LED3	19
4.2.2	Function Documentation	19
4.2.2.1	GPIO_IF_ConfigureNIntEnable	19
4.2.2.2	GPIO_IF_Get	20
4.2.2.3	GPIO_IF_GetPortNPin	20
4.2.2.4	GPIO_IF_LedConfigure	21
4.2.2.5	GPIO_IF_LedOff	22
4.2.2.6	GPIO_IF_LedOn	23
4.2.2.7	GPIO_IF_LedStatus	23
4.2.2.8	GPIO_IF_LedToggle	24
4.2.2.9	GPIO_IF_Set	25
4.2.3	Variable Documentation	25
4.2.3.1	g_ucLED1Pin	26
4.2.3.2	g_ucLED2Pin	26
4.2.3.3	g_ucLED3Pin	26
4.2.3.4	g_uiLED1Port	26
4.2.3.5	g_uiLED2Port	26
4.2.3.6	g_uiLED3Port	26
4.3	common/gpio_if.h File Reference	26
4.3.1	Enumeration Type Documentation	27
4.3.1.1	ledEnum	27
4.3.1.2	ledNames	27
4.3.2	Function Documentation	28
4.3.2.1	GPIO_IF_ConfigureNIntEnable	28
4.3.2.2	GPIO_IF_Get	28
4.3.2.3	GPIO_IF_GetPortNPin	29
4.3.2.4	GPIO_IF_LedConfigure	29
4.3.2.5	GPIO_IF_LedOff	30
4.3.2.6	GPIO_IF_LedOn	31
4.3.2.7	GPIO_IF_LedStatus	32
4.3.2.8	GPIO_IF_LedToggle	32
4.3.2.9	GPIO_IF_Set	34
4.4	common/startup_gcc.c File Reference	35
4.4.1	Function Documentation	35
4.4.1.1	__attribute__	35
4.4.1.2	_c_int00	36
4.4.1.3	_sbrk	36
4.4.1.4	ResetISR	36
4.4.1.5	vPortSVCHandler	37

4.4.1.6	xPortPendSVHandler	37
4.4.1.7	xPortSysTickHandler	37
4.4.2	Variable Documentation	37
4.4.2.1	__init_data	37
4.4.2.2	_bss	38
4.4.2.3	_data	38
4.4.2.4	_ebss	38
4.4.2.5	_edata	38
4.4.2.6	_eheap	38
4.4.2.7	_etext	38
4.4.2.8	_heap	38
4.5	common/uart_if.c File Reference	38
4.5.1	Macro Definition Documentation	39
4.5.1.1	IS_SPACE	39
4.5.2	Function Documentation	39
4.5.2.1	ClearTerm	39
4.5.2.2	Error	40
4.5.2.3	GetCmd	41
4.5.2.4	InitTerm	41
4.5.2.5	Message	42
4.5.2.6	Report	43
4.5.2.7	TrimSpace	44
4.5.3	Variable Documentation	45
4.5.3.1	ilen	45
4.6	common/uart_if.h File Reference	45
4.6.1	Macro Definition Documentation	46
4.6.1.1	CONSOLE	46
4.6.1.2	CONSOLE_PERIPH	46
4.6.1.3	SYSCLK	46
4.6.1.4	UART_BAUD_RATE	46
4.6.1.5	UART_IF_BUFFER	46
4.6.2	Function Documentation	46
4.6.2.1	ClearTerm	46
4.6.2.2	DispatcherUARTConfigure	47
4.6.2.3	DispatcherUartSendPacket	47
4.6.2.4	Error	47
4.6.2.5	GetCmd	47
4.6.2.6	InitTerm	48
4.6.2.7	Message	48
4.6.2.8	Report	49

4.6.2.9	TrimSpace	50
4.6.3	Variable Documentation	50
4.6.3.1	g_ucUARTBuffer	50
4.7	inc/pinmux.h File Reference	50
4.7.1	Function Documentation	51
4.7.1.1	PinMuxConfig	51
4.8	src/main.c File Reference	51
4.9	src/pinmux.c File Reference	52
4.9.1	Function Documentation	53
4.9.1.1	PinMuxConfig	53
Index		55

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Getting_started_sta	5
-------------------------------	---

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

common/common.h	13
common/gpio_if.c	18
common/gpio_if.h	26
common/startup_gcc.c	35
common/uart_if.c	38
common/uart_if.h	45
inc/pinmux.h	50
src/main.c	51
src/pinmux.c	52

Chapter 3

Module Documentation

3.1 Getting_started_sta

Macros

- #define [APPLICATION_NAME](#) "WLAN STATION"
- #define [APPLICATION_VERSION](#) "1.1.0"
- #define [HOST_NAME](#) "www.google.be"
- #define [PING_INTERVAL](#) 1000 /* In msecs */
- #define [PING_TIMEOUT](#) 3000 /* In msecs */
- #define [PING_PKT_SIZE](#) 20 /* In bytes */
- #define [NO_OF_ATTEMPTS](#) 3
- #define [OSI_STACK_SIZE](#) 2048

Enumerations

- enum [e_AppStatusCodes](#) { [LAN_CONNECTION_FAILED](#) = -0x7D0, [INTERNET_CONNECTION_FAILED](#) = [LAN_CONNECTION_FAILED](#) - 1, [DEVICE_NOT_IN_STATION_MODE](#) = [INTERNET_CONNECTION_FAILED](#) - 1, [STATUS_CODE_MAX](#) = -0xBB8 }

Functions

- void [WlanStationMode](#) (void *pvParameters)
Start simplelink, connect to the ap and run the ping test.
- void [SimpleLinkWlanEventHandler](#) (SIWlanEvent_t *pWlanEvent)
The Function Handles WLAN Events.
- void [SimpleLinkNetAppEventHandler](#) (SINetAppEvent_t *pNetAppEvent)
This function handles network events such as IP acquisition, IP leased, IP released etc.
- void [SimpleLinkHttpServerCallback](#) (SIHttpServerEvent_t *pHttpEvent, SIHttpServerResponse_t *pHttpResponse)
This function handles HTTP server events.
- void [SimpleLinkGeneralEventHandler](#) (SIDeviceEvent_t *pDevEvent)
This function handles General Events.
- void [SimpleLinkSockEventHandler](#) (SISockEvent_t *pSock)
- void [main](#) ()

Variables

- unsigned long `g_ulStatus` = 0
- unsigned long `g_ulPingPacketsRecv` = 0
- unsigned long `g_ulGatewayIP` = 0
- unsigned char `g_ucConnectionSSID` [SSID_LEN_MAX+1]
- unsigned char `g_ucConnectionBSSID` [BSSID_LEN_MAX]

3.1.1 Detailed Description

3.1.2 Macro Definition Documentation

3.1.2.1 `#define APPLICATION_NAME "WLAN STATION"`

Definition at line 88 of file main.c.

Referenced by `main()`.

3.1.2.2 `#define APPLICATION_VERSION "1.1.0"`

Definition at line 89 of file main.c.

3.1.2.3 `#define HOST_NAME "www.google.be"`

Definition at line 91 of file main.c.

3.1.2.4 `#define NO_OF_ATTEMPTS 3`

Definition at line 99 of file main.c.

3.1.2.5 `#define OSI_STACK_SIZE 2048`

Definition at line 101 of file main.c.

Referenced by `main()`.

3.1.2.6 `#define PING_INTERVAL 1000 /* In msecs */`

Definition at line 96 of file main.c.

3.1.2.7 `#define PING_PKT_SIZE 20 /* In bytes */`

Definition at line 98 of file main.c.

3.1.2.8 `#define PING_TIMEOUT 3000 /* In msecs */`

Definition at line 97 of file main.c.

3.1.3 Enumeration Type Documentation

3.1.3.1 enum e_AppStatusCodes

Enumerator

LAN_CONNECTION_FAILED

INTERNET_CONNECTION_FAILED

DEVICE_NOT_IN_STATION_MODE

STATUS_CODE_MAX

Definition at line 104 of file main.c.

3.1.4 Function Documentation

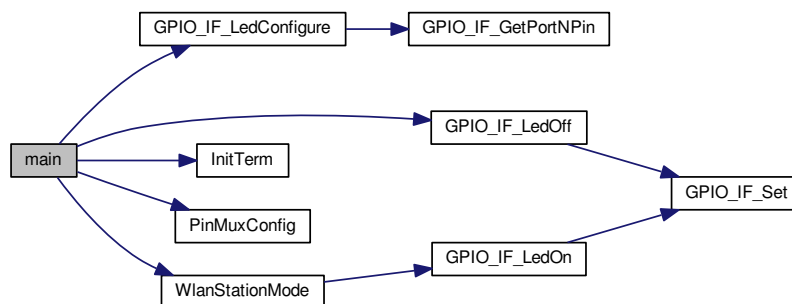
3.1.4.1 void main (void)

Definition at line 941 of file main.c.

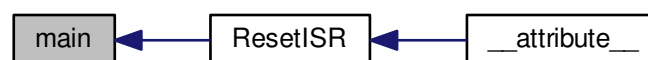
References APPLICATION_NAME, ERR_PRINT, GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), InitTerm(), LED1, LED2, LED3, LOOP_FOREVER, MCU_ALL_LED_IND, OSI_STACK_SIZE, PinMuxConfig(), SPAWN_TASK_PRIORITY, and WlanStationMode().

Referenced by ResetISR().

Here is the call graph for this function:



Here is the caller graph for this function:



3.1.4.2 void SimpleLinkGeneralEventHandler (SIdDeviceEvent_t * *pDevEvent*)

This function handles General Events.

Parameters

in	<i>pDevEvent</i>	- Pointer to General Event Info
----	------------------	---------------------------------

Returns

None

Definition at line 388 of file main.c.

References UART_PRINT.

3.1.4.3 void SimpleLinkHttpServerCallback (SIHttpServerEvent_t * *pHttpEvent*, SIHttpServerResponse_t * *pHttpResponse*)

This function handles HTTP server events.

Parameters

in	<i>pServerEvent</i>	- Contains the relevant event information
in	<i>pServer</i> ↔ <i>Response</i>	- Should be filled by the user with the relevant response information

Returns

None

Definition at line 373 of file main.c.

3.1.4.4 void SimpleLinkNetAppEventHandler (SINetAppEvent_t * *pNetAppEvent*)

This function handles network events such as IP acquisition, IP leased, IP released etc.

Parameters

in	<i>pNetAppEvent</i>	- Pointer to NetApp Event Info
----	---------------------	--------------------------------

Returns

None

Definition at line 323 of file main.c.

References g_ulGatewayIP, g_ulStatus, SET_STATUS_BIT, STATUS_BIT_IP_AQUIRED, and UART_PRINT.

3.1.4.5 void SimpleLinkSockEventHandler (SISockEvent_t * *pSock*)

This function handles socket events indication

Parameters

in	<i>pSock</i>	- Pointer to Socket Event Info
----	--------------	--------------------------------

Returns

None

Definition at line 409 of file main.c.

References UART_PRINT.

3.1.4.6 void SimpleLinkWlanEventHandler (SIWlanEvent_t * *pWlanEvent*)

The Function Handles WLAN Events.

Parameters

in	<i>pWlanEvent</i>	- Pointer to WLAN Event Info
----	-------------------	------------------------------

Returns

None

Definition at line 236 of file main.c.

References CLR_STATUS_BIT, g_ucConnectionBSSID, g_ucConnectionSSID, g_ulStatus, SET_STATUS_BIT, STATUS_BIT_CONNECTION, STATUS_BIT_IP_AQUIRED, and UART_PRINT.

3.1.4.7 void WlanStationMode (void * pvParameters)

Start simplelink, connect to the ap and run the ping test.

This function starts the simplelink, connect to the ap and start the ping test on the default gateway for the ap

Parameters

in	<i>pvParameters</i>	- Pointer to the list of parameters that can be passed to the task while creating it
----	---------------------	--

Returns

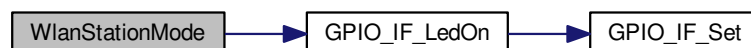
None

Definition at line 790 of file main.c.

References DEVICE_NOT_IN_STATION_MODE, GPIO_IF_LedOn(), LOOP_FOREVER, MCU_EXECUTE_SUCCESS_IND, MCU_ORANGE_LED_GPIO, SL_STOP_TIMEOUT, and UART_PRINT.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:

**3.1.5 Variable Documentation**

3.1.5.1 unsigned char g_ucConnectionBSSID[BSSID_LEN_MAX]

Definition at line 121 of file main.c.

Referenced by SimpleLinkWlanEventHandler().

3.1.5.2 unsigned char g_ucConnectionSSID[SSID_LEN_MAX+1]

Definition at line 120 of file main.c.

Referenced by SimpleLinkWlanEventHandler().

3.1.5.3 unsigned long g_ulGatewayIP = 0

Definition at line 119 of file main.c.

Referenced by SimpleLinkNetAppEventHandler().

3.1.5.4 unsigned long g_ulPingPacketsRecv = 0

Definition at line 118 of file main.c.

3.1.5.5 unsigned long g_ulStatus = 0

Definition at line 117 of file main.c.

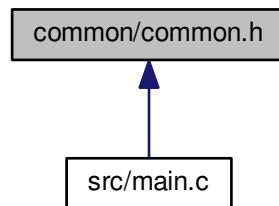
Referenced by SimpleLinkNetAppEventHandler(), and SimpleLinkWlanEventHandler().

Chapter 4

File Documentation

4.1 common/common.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define [SSID_NAME](#) "YOUR_WIFI_SSID" /* AP SSID */
- #define [SECURITY_TYPE](#) SL_SEC_TYPE_WPA_WPA2/* Security type (OPEN or WEP or WPA*/
- #define [SECURITY_KEY](#) "password" /* Password of the secured AP */
- #define [SSID_LEN_MAX](#) 32
- #define [BSSID_LEN_MAX](#) 6
- #define [UART_PRINT](#) Report
- #define [DBG_PRINT](#) Report
- #define [ERR_PRINT](#)(x) [Report](#)("Error [%d] at line [%d] in function [%s] \n\r",x,__LINE__,__FUNCTION__)
- #define [LOOP_FOREVER](#)()
- #define [ASSERT_ON_ERROR](#)(error_code)
- #define [SPAWN_TASK_PRIORITY](#) 9
- #define [SL_STOP_TIMEOUT](#) 200
- #define [UNUSED](#)(x) ((x) = (x))
- #define [SUCCESS](#) 0
- #define [FAILURE](#) -1
- #define [CLR_STATUS_BIT_ALL](#)(status_variable) (status_variable = 0)
- #define [SET_STATUS_BIT](#)(status_variable, bit) status_variable |= (1<<(bit))
- #define [CLR_STATUS_BIT](#)(status_variable, bit) status_variable &= ~(1<<(bit))

- `#define CLR_STATUS_BIT_ALL(status_variable) (status_variable = 0)`
- `#define GET_STATUS_BIT(status_variable, bit) (0 != (status_variable & (1 << (bit))))`
- `#define IS_NW_PROCSR_ON(status_variable)`
- `#define IS_CONNECTED(status_variable)`
- `#define IS_IP_LEASED(status_variable)`
- `#define IS_IP_ACQUIRED(status_variable)`
- `#define IS_SMART_CFG_START(status_variable)`
- `#define IS_P2P_DEV_FOUND(status_variable)`
- `#define IS_P2P_REQ_RCVD(status_variable)`
- `#define IS_CONNECT_FAILED(status_variable)`
- `#define IS_PING_DONE(status_variable)`

Enumerations

- `enum e_StatusBits {
STATUS_BIT_NWP_INIT = 0, STATUS_BIT_CONNECTION, STATUS_BIT_IP_LEASED, STATUS_BIT_IP_AQUIRED,
STATUS_BIT_SMARTCONFIG_START, STATUS_BIT_P2P_DEV_FOUND, STATUS_BIT_P2P_REQ_RECEIVED, STATUS_BIT_CONNECTION_FAILED,
STATUS_BIT_PING_DONE }`

4.1.1 Macro Definition Documentation

4.1.1.1 `#define ASSERT_ON_ERROR(error_code)`

Value:

```
{\
    if (error_code < 0) \
    {\
        ERR_PRINT(error_code);\
        return error_code;\
    }\
}
```

Definition at line 84 of file common.h.

4.1.1.2 `#define BSSID_LEN_MAX 6`

Definition at line 64 of file common.h.

4.1.1.3 `#define CLR_STATUS_BIT(status_variable, bit) status_variable &= ~(1 << (bit))`

Definition at line 134 of file common.h.

Referenced by SimpleLinkWlanEventHandler().

4.1.1.4 `#define CLR_STATUS_BIT_ALL(status_variable) (status_variable = 0)`

Definition at line 135 of file common.h.

4.1.1.5 `#define CLR_STATUS_BIT_ALL(status_variable) (status_variable = 0)`

Definition at line 135 of file common.h.

4.1.1.6 #define DBG_PRINT Report

Definition at line 73 of file common.h.

4.1.1.7 #define ERR_PRINT(x) Report("Error [%d] at line [%d] in function [%s] \n\r",x,__LINE__,__FUNCTION__)

Definition at line 74 of file common.h.

Referenced by main().

4.1.1.8 #define FAILURE -1

Definition at line 97 of file common.h.

4.1.1.9 #define GET_STATUS_BIT(status_variable, bit) (0 != (status_variable & (1<<(bit))))

Definition at line 136 of file common.h.

4.1.1.10 #define IS_CONNECT_FAILED(status_variable)

Value:

```
GET_STATUS_BIT(status_variable,\n                STATUS_BIT_CONNECTION_FAILED\n            )
```

Definition at line 152 of file common.h.

4.1.1.11 #define IS_CONNECTED(status_variable)

Value:

```
GET_STATUS_BIT(status_variable,\n                STATUS_BIT_CONNECTION)
```

Definition at line 140 of file common.h.

4.1.1.12 #define IS_IP_ACQUIRED(status_variable)

Value:

```
GET_STATUS_BIT(status_variable,\n                STATUS_BIT_IP_AQUIRED)
```

Definition at line 144 of file common.h.

4.1.1.13 #define IS_IP_LEASED(status_variable)

Value:

```
GET_STATUS_BIT(status_variable,\n                STATUS_BIT_IP_LEASED)
```

Definition at line 142 of file common.h.

4.1.1.14 #define IS_NW_PROCSR_ON(*status_variable*)

Value:

```
GET_STATUS_BIT(status_variable, \
                                STATUS_BIT_NWP_INIT)
```

Definition at line 138 of file common.h.

4.1.1.15 #define IS_P2P_DEV_FOUND(*status_variable*)

Value:

```
GET_STATUS_BIT(status_variable, \
                                STATUS_BIT_P2P_DEV_FOUND)
```

Definition at line 148 of file common.h.

4.1.1.16 #define IS_P2P_REQ_RCVD(*status_variable*)

Value:

```
GET_STATUS_BIT(status_variable, \
                                STATUS_BIT_P2P_REQ_RECEIVED)
```

Definition at line 150 of file common.h.

4.1.1.17 #define IS_PING_DONE(*status_variable*)

Value:

```
GET_STATUS_BIT(status_variable, \
                                STATUS_BIT_PING_DONE)
```

Definition at line 154 of file common.h.

4.1.1.18 #define IS_SMART_CFG_START(*status_variable*)

Value:

```
GET_STATUS_BIT(status_variable, \
                                STATUS_BIT_SMARTCONFIG_START
)
```

Definition at line 146 of file common.h.

4.1.1.19 #define LOOP_FOREVER()

Value:

```
{ \
    while(1); \
}
```

Definition at line 78 of file common.h.

Referenced by main(), and WlanStationMode().

4.1.1.20 `#define SECURITY_KEY "password" /* Password of the secured AP */`

Definition at line 62 of file common.h.

4.1.1.21 `#define SECURITY_TYPE SL_SEC_TYPE_WPA_WPA2/* Security type (OPEN or WEP or WPA*/`

Definition at line 61 of file common.h.

4.1.1.22 `#define SET_STATUS_BIT(status_variable, bit) status_variable |= (1<<(bit))`

Definition at line 133 of file common.h.

Referenced by SimpleLinkNetAppEventHandler(), and SimpleLinkWlanEventHandler().

4.1.1.23 `#define SL_STOP_TIMEOUT 200`

Definition at line 94 of file common.h.

Referenced by WlanStationMode().

4.1.1.24 `#define SPAWN_TASK_PRIORITY 9`

Definition at line 93 of file common.h.

Referenced by main().

4.1.1.25 `#define SSID_LEN_MAX 32`

Definition at line 63 of file common.h.

4.1.1.26 `#define SSID_NAME "YOUR_WIFI_SSID" /* AP SSID */`

Definition at line 60 of file common.h.

4.1.1.27 `#define SUCCESS 0`

Definition at line 96 of file common.h.

4.1.1.28 `#define UART_PRINT Report`

Definition at line 72 of file common.h.

Referenced by SimpleLinkGeneralEventHandler(), SimpleLinkNetAppEventHandler(), SimpleLinkSockEvent↵
Handler(), SimpleLinkWlanEventHandler(), and WlanStationMode().

4.1.1.29 `#define UNUSED(x) ((x) = (x))`

Definition at line 95 of file common.h.

4.1.2 Enumeration Type Documentation

4.1.2.1 enum e_StatusBits

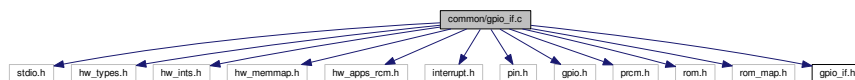
Enumerator

STATUS_BIT_NWP_INIT
STATUS_BIT_CONNECTION
STATUS_BIT_IP_LEASED
STATUS_BIT_IP_AQUIRED
STATUS_BIT_SMARTCONFIG_START
STATUS_BIT_P2P_DEV_FOUND
STATUS_BIT_P2P_REQ_RECEIVED
STATUS_BIT_CONNECTION_FAILED
STATUS_BIT_PING_DONE

Definition at line 102 of file common.h.

4.2 common/gpio_if.c File Reference

```
#include <stdio.h>
#include "hw_types.h"
#include "hw_ints.h"
#include "hw_memmap.h"
#include "hw_apps_rcm.h"
#include "interrupt.h"
#include "pin.h"
#include "gpio.h"
#include "prcm.h"
#include "rom.h"
#include "rom_map.h"
#include "gpio_if.h"
Include dependency graph for gpio_if.c:
```



Macros

- `#define PIN_LED1 9`
- `#define PIN_LED2 10`
- `#define PIN_LED3 11`

Functions

- void `GPIO_IF_LedConfigure` (unsigned char ucPins)
- void `GPIO_IF_LedOn` (char ledNum)
Turns a specific LED Off.
- void `GPIO_IF_LedOff` (char ledNum)

Turns a specific LED Off.

- unsigned char [GPIO_IF_LedStatus](#) (unsigned char ucGPIONum)

This function returns LED current Status.

- void [GPIO_IF_LedToggle](#) (unsigned char ucLedNum)

Toggles a board LED.

- void [GPIO_IF_GetPortNPin](#) (unsigned char ucPin, unsigned int *puiGPIOPort, unsigned char *pucGPIOPin)
- void [GPIO_IF_ConfigureNIntEnable](#) (unsigned int uiGPIOPort, unsigned char ucGPIOPin, unsigned int uiIntType, void(*pfnIntHandler)(void))
- void [GPIO_IF_Set](#) (unsigned char ucPin, unsigned int uiGPIOPort, unsigned char ucGPIOPin, unsigned char ucGPIOValue)
- unsigned char [GPIO_IF_Get](#) (unsigned char ucPin, unsigned int uiGPIOPort, unsigned char ucGPIOPin)

Variables

- unsigned int [g_uiLED1Port](#) = 0
- unsigned int [g_uiLED2Port](#) = 0
- unsigned int [g_uiLED3Port](#) = 0
- unsigned char [g_ucLED1Pin](#)
- unsigned char [g_ucLED2Pin](#)
- unsigned char [g_ucLED3Pin](#)

4.2.1 Macro Definition Documentation

4.2.1.1 #define PIN_LED1 9

Definition at line 82 of file gpio_if.c.

Referenced by [GPIO_IF_LedConfigure\(\)](#), [GPIO_IF_LedOff\(\)](#), and [GPIO_IF_LedOn\(\)](#).

4.2.1.2 #define PIN_LED2 10

Definition at line 83 of file gpio_if.c.

Referenced by [GPIO_IF_LedConfigure\(\)](#), [GPIO_IF_LedOff\(\)](#), and [GPIO_IF_LedOn\(\)](#).

4.2.1.3 #define PIN_LED3 11

Definition at line 84 of file gpio_if.c.

Referenced by [GPIO_IF_LedConfigure\(\)](#), [GPIO_IF_LedOff\(\)](#), and [GPIO_IF_LedOn\(\)](#).

4.2.2 Function Documentation

4.2.2.1 void [GPIO_IF_ConfigureNIntEnable](#) (unsigned int *uiGPIOPort*, unsigned char *ucGPIOPin*, unsigned int *uiIntType*, void(*)*(void) pfnIntHandler*)

Configures the GPIO selected as input to generate interrupt on activity

Parameters

<i>uiGPIOPort</i>	is the GPIO port address
-------------------	--------------------------

<i>ucGPIOPin</i>	is the GPIO pin of the specified port
<i>uiIntType</i>	is the type of the interrupt (refer gpio.h)
<i>pfnIntHandler</i>	is the interrupt handler to register

This function

1. Sets GPIO interrupt type
2. Registers Interrupt handler
3. Enables Interrupt

Returns

None

Definition at line 372 of file gpio_if.c.

4.2.2.2 unsigned char GPIO_IF_Get (unsigned char *ucPin*, unsigned int *uiGPIOPort*, unsigned char *ucGPIOPin*)

Set a value to the specified GPIO pin

Parameters

<i>ucPin</i>	is the GPIO pin to be set (0:39)
<i>uiGPIOPort</i>	is the GPIO port address
<i>ucGPIOPin</i>	is the GPIO pin of the specified port

This function

1. Gets a value of the specified GPIO pin

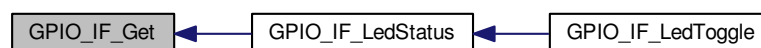
Returns

value of the GPIO pin

Definition at line 447 of file gpio_if.c.

Referenced by GPIO_IF_LedStatus().

Here is the caller graph for this function:



4.2.2.3 void GPIO_IF_GetPortNPin (unsigned char *ucPin*, unsigned int * *puiGPIOPort*, unsigned char * *pucGPIOPin*)

Get the port and pin of a given GPIO

Parameters

<i>ucPin</i>	is the pin to be set-up as a GPIO (0:39)
<i>puiGPIOPort</i>	is the pointer to store GPIO port address return value
<i>pucGPIOPin</i>	is the pointer to store GPIO pin return value

This function

1. Return the GPIO port address and pin for a given external pin number

Returns

None.

Definition at line 338 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure().

Here is the caller graph for this function:



4.2.2.4 void GPIO_IF_LedConfigure (unsigned char ucPins)

GPIO Enable & Configuration

Parameters

<i>ucPins</i>	is the bit-pack representation of 3 LEDs LSB:GP09-GP10-GP11:MSB
---------------	---

Returns

None

Definition at line 123 of file gpio_if.c.

References g_ucLED1Pin, g_ucLED2Pin, g_ucLED3Pin, g_uiLED1Port, g_uiLED2Port, g_uiLED3Port, GPIO_IF_GetPortNPin(), LED1, LED2, LED3, PIN_LED1, PIN_LED2, and PIN_LED3.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.5 void GPIO_IF_LedOff (char *ledNum*)

Turns a specific LED Off.

Turn LED Off

Parameters

<i>ledNum</i>	is the LED Number
---------------	-------------------

Returns

none

Definition at line 217 of file `gpio_if.c`.

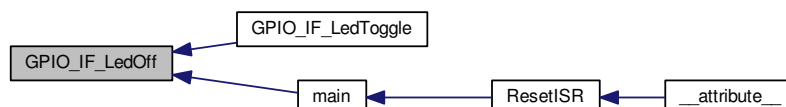
References `g_ucLED1Pin`, `g_ucLED2Pin`, `g_ucLED3Pin`, `g_uiLED1Port`, `g_uiLED2Port`, `g_uiLED3Port`, `GPIO_IF_Set()`, `MCU_ALL_LED_IND`, `MCU_ASSOCIATED_IND`, `MCU_CLIENT_CONNECTED_IND`, `MCU_EXECUTE_FAIL_IND`, `MCU_EXECUTE_SUCCESS_IND`, `MCU_GREEN_LED_GPIO`, `MCU_IP_ALLOC_IND`, `MCU_ON_IN_D`, `MCU_ORANGE_LED_GPIO`, `MCU_RED_LED_GPIO`, `MCU_SENDING_DATA_IND`, `MCU_SERVER_INIT_IND`, `PIN_LED1`, `PIN_LED2`, and `PIN_LED3`.

Referenced by `GPIO_IF_LedToggle()`, and `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.6 void GPIO_IF_LedOn (char *ledNum*)

Turns a specific LED Off.

Turn LED On

Parameters

<i>ledNum</i>	is the LED Number
---------------	-------------------

Returns

none

Definition at line 162 of file gpio_if.c.

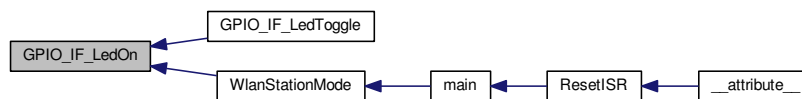
References g_ucLED1Pin, g_ucLED2Pin, g_ucLED3Pin, g_uiLED1Port, g_uiLED2Port, g_uiLED3Port, GPIO_IF_Set(), MCU_ALL_LED_IND, MCU_ASSOCIATED_IND, MCU_CLIENT_CONNECTED_IND, MCU_EXECUTE_FAIL_IND, MCU_EXECUTE_SUCCESS_IND, MCU_GREEN_LED_GPIO, MCU_IP_ALLOC_IND, MCU_ON_IND, MCU_ORANGE_LED_GPIO, MCU_RED_LED_GPIO, MCU_SENDING_DATA_IND, MCU_SERVER_INIT_IND, PIN_LED1, PIN_LED2, and PIN_LED3.

Referenced by GPIO_IF_LedToggle(), and WlanStationMode().

Here is the call graph for this function:



Here is the caller graph for this function:

4.2.2.7 unsigned char GPIO_IF_LedStatus (unsigned char *ucGPINum*)

This function returns LED current Status.

Parameters

in	<i>ucGPINum</i>	is the GPIO to which the LED is connected MCU_GREEN_LED_GPIO
----	-----------------	--

Returns

1: LED ON, 0: LED OFF

Definition at line 272 of file gpio_if.c.

References `g_ucLED1Pin`, `g_ucLED2Pin`, `g_ucLED3Pin`, `g_uiLED1Port`, `g_uiLED2Port`, `g_uiLED3Port`, `GPIO_IF_Get()`, `MCU_GREEN_LED_GPIO`, `MCU_ORANGE_LED_GPIO`, and `MCU_RED_LED_GPIO`.

Referenced by `GPIO_IF_LedToggle()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.8 void GPIO_IF_LedToggle (unsigned char *ucLedNum*)

Toggles a board LED.

Toggle the Led state

Parameters

<i>ledNum</i>	is the LED Number
---------------	-------------------

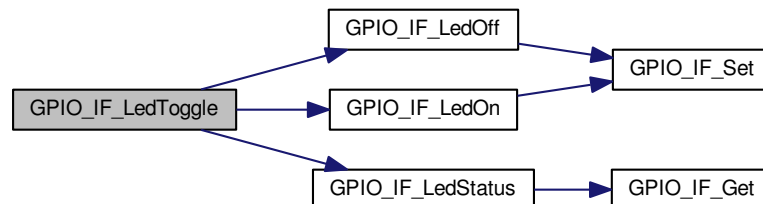
Returns

none

Definition at line 309 of file gpio_if.c.

References GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

Here is the call graph for this function:



4.2.2.9 void GPIO_IF_Set (unsigned char *ucPin*, unsigned int *uiGPIOPort*, unsigned char *ucGPIOPin*, unsigned char *ucGPIOValue*)

Set a value to the specified GPIO pin

Parameters

<i>ucPin</i>	is the GPIO pin to be set (0:39)
<i>uiGPIOPort</i>	is the GPIO port address
<i>ucGPIOPin</i>	is the GPIO pin of the specified port
<i>ucGPIOValue</i>	is the value to be set

This function

1. Sets a value to the specified GPIO pin

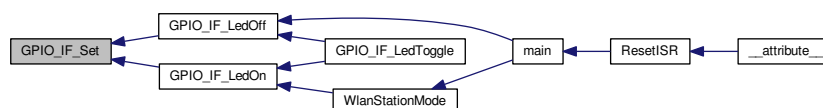
Returns

None.

Definition at line 416 of file gpio_if.c.

Referenced by GPIO_IF_LedOff(), and GPIO_IF_LedOn().

Here is the caller graph for this function:



4.2.3 Variable Documentation

4.2.3.1 unsigned char g_ucLED1Pin

Definition at line 80 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

4.2.3.2 unsigned char g_ucLED2Pin

Definition at line 80 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

4.2.3.3 unsigned char g_ucLED3Pin

Definition at line 80 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

4.2.3.4 unsigned int g_uiLED1Port = 0

Definition at line 79 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

4.2.3.5 unsigned int g_uiLED2Port = 0

Definition at line 79 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

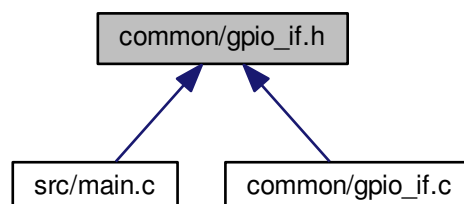
4.2.3.6 unsigned int g_uiLED3Port = 0

Definition at line 79 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure(), GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

4.3 common/gpio_if.h File Reference

This graph shows which files directly or indirectly include this file:



Enumerations

- enum `ledEnum` { `NO_LED`, `LED1` = 0x1, `LED2` = 0x2, `LED3` = 0x4 }
- enum `ledNames` {
`NO_LED_IND` = `NO_LED`, `MCU_SENDING_DATA_IND` = `LED1`, `MCU_ASSOCIATED_IND`, `MCU_IP_AL-`
`LOC_IND`,
`MCU_SERVER_INIT_IND`, `MCU_CLIENT_CONNECTED_IND`, `MCU_ON_IND`, `MCU_EXECUTE_SUCCE-`
`SS_IND`,
`MCU_EXECUTE_FAIL_IND`, `MCU_RED_LED_GPIO`, `MCU_ORANGE_LED_GPIO`, `MCU_GREEN_LED_`
`GPIO`,
`MCU_ALL_LED_IND` }

Functions

- void `GPIO_IF_GetPortNPin` (unsigned char ucPin, unsigned int *puiGPIOPort, unsigned char *pucGPIOPin)
- void `GPIO_IF_ConfigureNIntEnable` (unsigned int uiGPIOPort, unsigned char ucGPIOPin, unsigned int ui-
IntType, void(*pfnIntHandler)(void))
- void `GPIO_IF_Set` (unsigned char ucPin, unsigned int uiGPIOPort, unsigned char ucGPIOPin, unsigned char
ucGPIOValue)
- unsigned char `GPIO_IF_Get` (unsigned char ucPin, unsigned int uiGPIOPort, unsigned char ucGPIOPin)
- void `GPIO_IF_LedConfigure` (unsigned char ucPins)
- void `GPIO_IF_LedOn` (char ledNum)
Turns a specific LED Off.
- void `GPIO_IF_LedOff` (char ledNum)
Turns a specific LED Off.
- unsigned char `GPIO_IF_LedStatus` (unsigned char ucGPIONum)
This function returns LED current Status.
- void `GPIO_IF_LedToggle` (unsigned char ucLedNum)
Toggles a board LED.

4.3.1 Enumeration Type Documentation

4.3.1.1 enum ledEnum

Enumerator

`NO_LED`
`LED1`
`LED2`
`LED3`

Definition at line 53 of file `gpio_if.h`.

4.3.1.2 enum ledNames

Enumerator

`NO_LED_IND`
`MCU_SENDING_DATA_IND`
`MCU_ASSOCIATED_IND`
`MCU_IP_ALLOC_IND`
`MCU_SERVER_INIT_IND`
`MCU_CLIENT_CONNECTED_IND`

MCU_ON_IND
MCU_EXECUTE_SUCCESS_IND
MCU_EXECUTE_FAIL_IND
MCU_RED_LED_GPIO
MCU_ORANGE_LED_GPIO
MCU_GREEN_LED_GPIO
MCU_ALL_LED_IND

Definition at line 62 of file gpio_if.h.

4.3.2 Function Documentation

4.3.2.1 void GPIO_IF_ConfigureNIntEnable (unsigned int *uiGPIOPort*, unsigned char *ucGPIOPin*, unsigned int *uiIntType*, void(*)(void) *pfnIntHandler*)

Configures the GPIO selected as input to generate interrupt on activity

Parameters

<i>uiGPIOPort</i>	is the GPIO port address
<i>ucGPIOPin</i>	is the GPIO pin of the specified port
<i>uiIntType</i>	is the type of the interrupt (refer gpio.h)
<i>pfnIntHandler</i>	is the interrupt handler to register

This function

1. Sets GPIO interrupt type
2. Registers Interrupt handler
3. Enables Interrupt

Returns

None

Definition at line 372 of file gpio_if.c.

4.3.2.2 unsigned char GPIO_IF_Get (unsigned char *ucPin*, unsigned int *uiGPIOPort*, unsigned char *ucGPIOPin*)

Set a value to the specified GPIO pin

Parameters

<i>ucPin</i>	is the GPIO pin to be set (0:39)
<i>uiGPIOPort</i>	is the GPIO port address
<i>ucGPIOPin</i>	is the GPIO pin of the specified port

This function

1. Gets a value of the specified GPIO pin

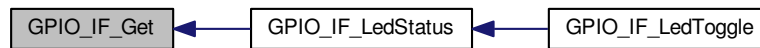
Returns

value of the GPIO pin

Definition at line 447 of file gpio_if.c.

Referenced by GPIO_IF_LedStatus().

Here is the caller graph for this function:



4.3.2.3 void GPIO_IF_GetPortNPin (unsigned char *ucPin*, unsigned int * *puiGPIOPort*, unsigned char * *pucGPIOPin*)

Get the port and pin of a given GPIO

Parameters

<i>ucPin</i>	is the pin to be set-up as a GPIO (0:39)
<i>puiGPIOPort</i>	is the pointer to store GPIO port address return value
<i>pucGPIOPin</i>	is the pointer to store GPIO pin return value

This function

1. Return the GPIO port address and pin for a given external pin number

Returns

None.

Definition at line 338 of file gpio_if.c.

Referenced by GPIO_IF_LedConfigure().

Here is the caller graph for this function:



4.3.2.4 void GPIO_IF_LedConfigure (unsigned char *ucPins*)

GPIO Enable & Configuration

Parameters

<i>ucPins</i>	is the bit-pack representation of 3 LEDs LSB:GP09-GP10-GP11:MSB
---------------	---

Returns

None

Definition at line 123 of file gpio_if.c.

References g_ucLED1Pin, g_ucLED2Pin, g_ucLED3Pin, g_uiLED1Port, g_uiLED2Port, g_uiLED3Port, GPIO_IF_↵_GetPortNPin(), LED1, LED2, LED3, PIN_LED1, PIN_LED2, and PIN_LED3.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:

**4.3.2.5 void GPIO_IF_LedOff (char ledNum)**

Turns a specific LED Off.

Turn LED Off

Parameters

<i>ledNum</i>	is the LED Number
---------------	-------------------

Returns

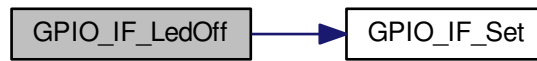
none

Definition at line 217 of file gpio_if.c.

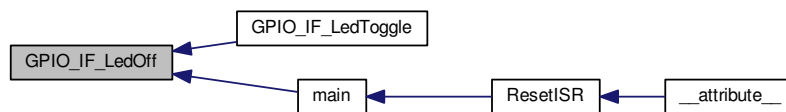
References g_ucLED1Pin, g_ucLED2Pin, g_ucLED3Pin, g_uiLED1Port, g_uiLED2Port, g_uiLED3Port, GPIO_I↵F_Set(), MCU_ALL_LED_IND, MCU_ASSOCIATED_IND, MCU_CLIENT_CONNECTED_IND, MCU_EXECUTE_↵_FAIL_IND, MCU_EXECUTE_SUCCESS_IND, MCU_GREEN_LED_GPIO, MCU_IP_ALLOC_IND, MCU_ON_IN_↵D, MCU_ORANGE_LED_GPIO, MCU_RED_LED_GPIO, MCU_SENDING_DATA_IND, MCU_SERVER_INIT_IND, PIN_LED1, PIN_LED2, and PIN_LED3.

Referenced by GPIO_IF_LedToggle(), and main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.6 void GPIO_IF_LedOn (char *ledNum*)

Turns a specific LED Off.

Turn LED On

Parameters

<i>ledNum</i>	is the LED Number
---------------	-------------------

Returns

none

Definition at line 162 of file gpio_if.c.

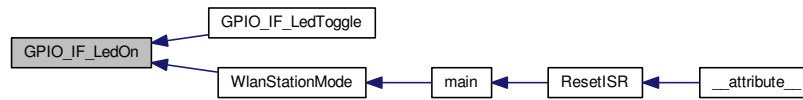
References g_ucLED1Pin, g_ucLED2Pin, g_ucLED3Pin, g_uiLED1Port, g_uiLED2Port, g_uiLED3Port, GPIO_IF_Set(), MCU_ALL_LED_IND, MCU_ASSOCIATED_IND, MCU_CLIENT_CONNECTED_IND, MCU_EXECUTE_FAIL_IND, MCU_EXECUTE_SUCCESS_IND, MCU_GREEN_LED_GPIO, MCU_IP_ALLOC_IND, MCU_ON_IND, MCU_ORANGE_LED_GPIO, MCU_RED_LED_GPIO, MCU_SENDING_DATA_IND, MCU_SERVER_INIT_IND, PIN_LED1, PIN_LED2, and PIN_LED3.

Referenced by GPIO_IF_LedToggle(), and WlanStationMode().

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.7 unsigned char GPIO_IF_LedStatus (unsigned char ucGPIONum)

This function returns LED current Status.

Parameters

in	<i>ucGPIONum</i>	is the GPIO to which the LED is connected MCU_GREEN_LED_GPIO
----	------------------	--

Returns

1: LED ON, 0: LED OFF

Definition at line 272 of file `gpio_if.c`.

References `g_ucLED1Pin`, `g_ucLED2Pin`, `g_ucLED3Pin`, `g_uiLED1Port`, `g_uiLED2Port`, `g_uiLED3Port`, `GPIO_IF_Get()`, `MCU_GREEN_LED_GPIO`, `MCU_ORANGE_LED_GPIO`, and `MCU_RED_LED_GPIO`.

Referenced by `GPIO_IF_LedToggle()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.8 void GPIO_IF_LedToggle (unsigned char ucLedNum)

Toggles a board LED.

Toggle the Led state

Parameters

<i>ledNum</i>	is the LED Number
---------------	-------------------

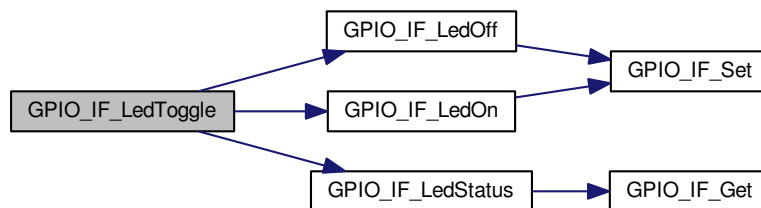
Returns

none

Definition at line 309 of file gpio_if.c.

References GPIO_IF_LedOff(), GPIO_IF_LedOn(), and GPIO_IF_LedStatus().

Here is the call graph for this function:



4.3.2.9 void GPIO_IF_Set (unsigned char *ucPin*, unsigned int *uiGPIOPort*, unsigned char *ucGPIOPin*, unsigned char *ucGPIOValue*)

Set a value to the specified GPIO pin

Parameters

<i>ucPin</i>	is the GPIO pin to be set (0:39)
<i>uiGPIOPort</i>	is the GPIO port address
<i>ucGPIOPin</i>	is the GPIO pin of the specified port
<i>ucGPIOValue</i>	is the value to be set

This function

1. Sets a value to the specified GPIO pin

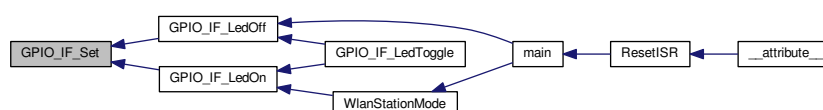
Returns

None.

Definition at line 416 of file gpio_if.c.

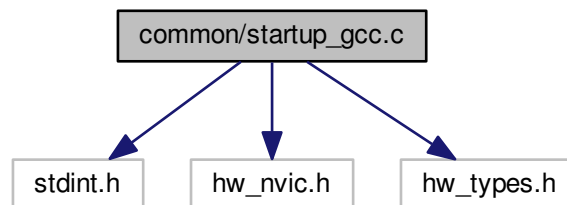
Referenced by GPIO_IF_LedOff(), and GPIO_IF_LedOn().

Here is the caller graph for this function:



4.4 common/startup_gcc.c File Reference

```
#include <stdint.h>
#include "hw_nvic.h"
#include "hw_types.h"
Include dependency graph for startup_gcc.c:
```



Functions

- void [ResetISR](#) (void)
- void [_c_int00](#) (void)
- void [vPortSVCHandler](#) (void)
- void [xPortPendSVHandler](#) (void)
- void [xPortSysTickHandler](#) (void)
- int [main](#) (void)
- [__attribute__](#) ((section(".intvecs")))
- void * [_sbrk](#) (unsigned int incr)

Variables

- unsigned long [_heap](#)
- unsigned long [_eheap](#)
- uint32_t [_etext](#)
- uint32_t [_data](#)
- uint32_t [_edata](#)
- uint32_t [_bss](#)
- uint32_t [_ebss](#)
- uint32_t [__init_data](#)

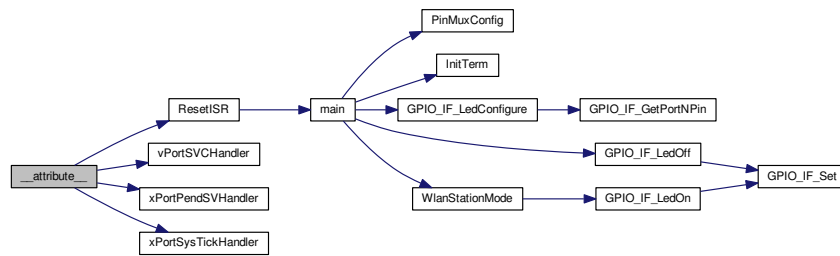
4.4.1 Function Documentation

4.4.1.1 [__attribute__](#) ((section(".intvecs")))

Definition at line 94 of file startup_gcc.c.

References [__init_data](#), [_bss](#), [_data](#), [_ebss](#), [_edata](#), [_etext](#), [ResetISR\(\)](#), [vPortSVCHandler\(\)](#), [xPortPendSVHandler\(\)](#), and [xPortSysTickHandler\(\)](#).

Here is the call graph for this function:



4.4.1.2 void _c_int00 (void)

4.4.1.3 void* _sbrk (unsigned int *incr*)

Definition at line 325 of file startup_gcc.c.

References _eheap, and _heap.

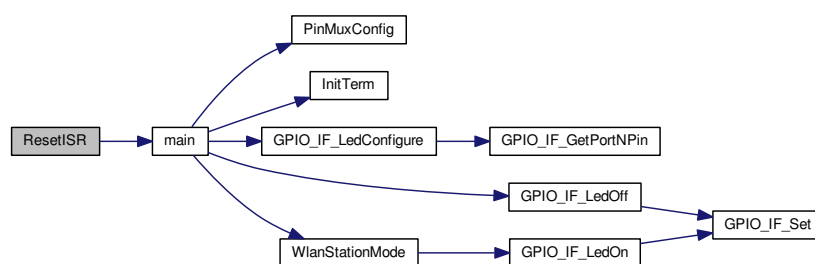
4.4.1.4 void ResetISR (void)

Definition at line 214 of file startup_gcc.c.

References __init_data, _edata, and main().

Referenced by __attribute__().

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.1.5 void vPortSVCHandler (void)

Referenced by `__attribute__()`.

Here is the caller graph for this function:



4.4.1.6 void xPortPendSVHandler (void)

Referenced by `__attribute__()`.

Here is the caller graph for this function:



4.4.1.7 void xPortSysTickHandler (void)

Referenced by `__attribute__()`.

Here is the caller graph for this function:



4.4.2 Variable Documentation

4.4.2.1 uint32_t __init_data

Referenced by `__attribute__()`, and `ResetISR()`.

4.4.2.2 uint32_t_bss

Referenced by `__attribute__()`.

4.4.2.3 uint32_t_data

Referenced by `__attribute__()`.

4.4.2.4 uint32_t_ebss

Referenced by `__attribute__()`.

4.4.2.5 uint32_t_edata

Referenced by `__attribute__()`, and `ResetISR()`.

4.4.2.6 unsigned long _eheap

Referenced by `_sbrk()`.

4.4.2.7 uint32_t_etext

Referenced by `__attribute__()`.

4.4.2.8 unsigned long _heap

Referenced by `_sbrk()`.

4.5 common/uart_if.c File Reference

```
#include <stdarg.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "hw_types.h"
#include "hw_memmap.h"
#include "prcm.h"
#include "pin.h"
#include "uart.h"
#include "rom.h"
#include "rom_map.h"
#include "uart_if.h"
```

Include dependency graph for `uart_if.c`:



Macros

- #define [IS_SPACE](#)(x) (x == 32 ? 1 : 0)

Functions

- void [InitTerm](#) ()
- void [Message](#) (const char *str)
- void [ClearTerm](#) ()
- void [Error](#) (char *pcFormat,...)
- int [GetCmd](#) (char *pcBuffer, unsigned int uiBufLen)
- int [TrimSpace](#) (char *pcInput)
- int [Report](#) (const char *pcFormat,...)

Variables

- unsigned int [ilen](#) =1

4.5.1 Macro Definition Documentation

4.5.1.1 #define IS_SPACE(x) (x == 32 ? 1 : 0)

Definition at line 56 of file `uart_if.c`.

Referenced by `TrimSpace()`.

4.5.2 Function Documentation

4.5.2.1 void ClearTerm (void)

Clear the console window

This function

1. clears the console window.

Returns

none

Definition at line 127 of file `uart_if.c`.

References `Message()`.

Here is the call graph for this function:



4.5.2.2 void Error (char * *pcFormat*, ...)

Error Function

Parameters

--	--

Definition at line 142 of file uart_if.c.

References Message().

Here is the call graph for this function:



4.5.2.3 int GetCmd (char * *pcBuffer*, unsigned int *uiBufLen*)

Get the Command string from UART

Parameters

<i>pcBuffer</i>	is the command store to which command will be populated
<i>ucBufLen</i>	is the length of buffer store available

Returns

Length of the bytes received. -1 if buffer length exceeded.

Definition at line 165 of file uart_if.c.

References `CONSOLE`, and Report().

Here is the call graph for this function:



4.5.2.4 void InitTerm (void)

Initialization

This function

1. Configures the UART to be used.

Returns

none

Definition at line 80 of file uart_if.c.

References `CONSOLE`, `CONSOLE_PERIPH`, and `UART_BAUD_RATE`.

Referenced by `main()`.

Here is the caller graph for this function:

**4.5.2.5 void Message (const char * str)**

Outputs a character string to the console

Parameters

<i>str</i>	is the pointer to the string to be printed
------------	--

This function

1. prints the input string character by character on to the console.

Returns

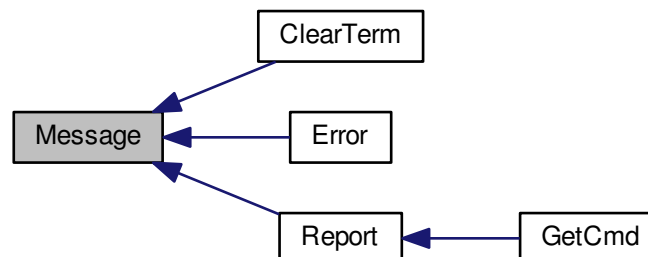
none

Definition at line 103 of file uart_if.c.

References `CONSOLE`.

Referenced by `ClearTerm()`, `Error()`, and `Report()`.

Here is the caller graph for this function:



4.5.2.6 int Report (const char * *pcFormat*, ...)

prints the formatted string on to the console

Parameters

<i>format</i>	is a pointer to the character string specifying the format in the following arguments need to be interpreted.
<i>[variable</i>	number of] arguments according to the format in the first parameters This function 1. prints the formatted error statement.

Returns

count of characters printed

Definition at line 278 of file uart_if.c.

References Message().

Referenced by GetCmd().

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.7 int TrimSpace (char * *pclnput*)

Trim the spaces from left and right end of given string

Parameters

<i>Input</i>	string on which trimming happens
--------------	----------------------------------

Returns

length of trimmed string

Definition at line 239 of file uart_if.c.

References IS_SPACE.

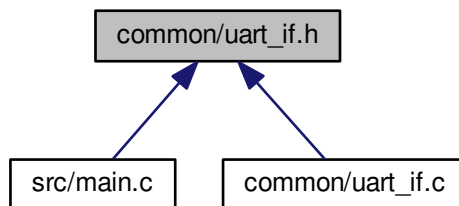
4.5.3 Variable Documentation

4.5.3.1 unsigned int ilen =1

Definition at line 66 of file uart_if.c.

4.6 common/uart_if.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- `#define` [UART_BAUD_RATE](#) 115200
- `#define` [SYSCLK](#) 80000000
- `#define` [CONSOLE](#) UARTA0_BASE
- `#define` [CONSOLE_PERIPH](#) PRCM_UARTA0
- `#define` [UART_IF_BUFFER](#) 64

Functions

- void [DispatcherUARTConfigure](#) (void)
- void [DispatcherUartSendPacket](#) (unsigned char *inBuff, unsigned short usLength)
- int [GetCmd](#) (char *pcBuffer, unsigned int uiBufLen)
- void [InitTerm](#) (void)
- void [ClearTerm](#) (void)
- void [Message](#) (const char *format,...)
- void [Error](#) (char *format,...)
- int [TrimSpace](#) (char *pcInput)
- int [Report](#) (const char *format,...)

Variables

- unsigned char [g_ucUARTBuffer](#) []

4.6.1 Macro Definition Documentation

4.6.1.1 `#define` `CONSOLE_UARTA0_BASE`

Definition at line 58 of file `uart_if.h`.

Referenced by `GetCmd()`, `InitTerm()`, and `Message()`.

4.6.1.2 `#define` `CONSOLE_PERIPH_PRCM_UARTA0`

Definition at line 59 of file `uart_if.h`.

Referenced by `InitTerm()`.

4.6.1.3 `#define` `SYSCLK 80000000`

Definition at line 57 of file `uart_if.h`.

4.6.1.4 `#define` `UART_BAUD_RATE 115200`

Definition at line 56 of file `uart_if.h`.

Referenced by `InitTerm()`.

4.6.1.5 `#define` `UART_IF_BUFFER 64`

Definition at line 63 of file `uart_if.h`.

4.6.2 Function Documentation

4.6.2.1 `void` `ClearTerm` (`void`)

Clear the console window

This function

1. clears the console window.

Returns

none

Definition at line 127 of file `uart_if.c`.

References `Message()`.

Here is the call graph for this function:



4.6.2.2 void DispatcherUARTConfigure (void)

4.6.2.3 void DispatcherUartSendPacket (unsigned char * *inBuff*, unsigned short *usLength*)

4.6.2.4 void Error (char * *pcFormat*, ...)

Error Function

Parameters

--	--

Definition at line 142 of file uart_if.c.

References Message().

Here is the call graph for this function:



4.6.2.5 int GetCmd (char * *pcBuffer*, unsigned int *uiBufLen*)

Get the Command string from UART

Parameters

<i>pcBuffer</i>	is the command store to which command will be populated
<i>ucBufLen</i>	is the length of buffer store available

Returns

Length of the bytes received. -1 if buffer length exceeded.

Definition at line 165 of file uart_if.c.

References CONSOLE, and Report().

Here is the call graph for this function:



4.6.2.6 void InitTerm (void)

Initialization

This function

1. Configures the UART to be used.

Returns

none

Definition at line 80 of file uart_if.c.

References `CONSOLE`, `CONSOLE_PERIPH`, and `UART_BAUD_RATE`.

Referenced by `main()`.

Here is the caller graph for this function:



4.6.2.7 void Message (const char * str)

Outputs a character string to the console

Parameters

<i>str</i>	is the pointer to the string to be printed
------------	--

This function

1. prints the input string character by character on to the console.

Returns

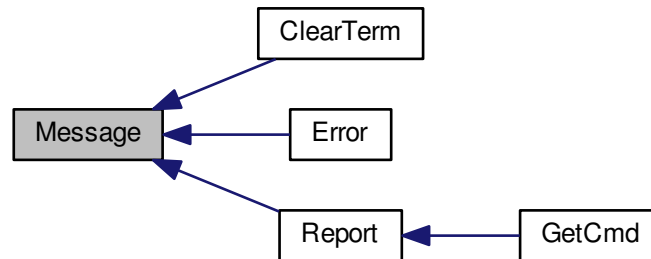
none

Definition at line 103 of file `uart_if.c`.

References `CONSOLE`.

Referenced by `ClearTerm()`, `Error()`, and `Report()`.

Here is the caller graph for this function:

**4.6.2.8 int Report (const char * *pcFormat*, ...)**

prints the formatted string on to the console

Parameters

<i>format</i>	is a pointer to the character string specifying the format in the following arguments need to be interpreted.
[<i>variable</i>	number of] arguments according to the format in the first parameters This function 1. prints the formatted error statement.

Returns

count of characters printed

Definition at line 278 of file `uart_if.c`.

References `Message()`.

Referenced by `GetCmd()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.6.2.9 int TrimSpace (char * *pcInput*)

Trim the spaces from left and right end of given string

Parameters

<i>Input</i>	string on which trimming happens
--------------	----------------------------------

Returns

length of trimmed string

Definition at line 239 of file uart_if.c.

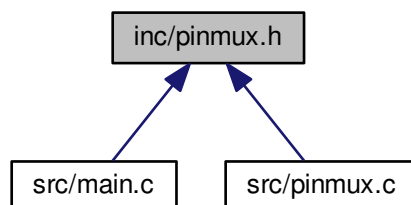
References IS_SPACE.

4.6.3 Variable Documentation

4.6.3.1 unsigned char g_ucUARTBuffer[]

4.7 inc/pinmux.h File Reference

This graph shows which files directly or indirectly include this file:



Functions

- void [PinMuxConfig](#) (void)

4.7.1 Function Documentation

4.7.1.1 void PinMuxConfig (void)

Definition at line 56 of file pinmux.c.

Referenced by main().

Here is the caller graph for this function:



4.8 src/main.c File Reference

```

#include <stdlib.h>
#include <string.h>
#include "simplelink.h"
#include "hw_types.h"
#include "hw_ints.h"
#include "rom.h"
#include "rom_map.h"
#include "interrupt.h"
#include "prcm.h"
#include "utils.h"
#include <osi.h>
#include "gpio_if.h"
#include "uart_if.h"
#include "common.h"
#include "pinmux.h"

```

Include dependency graph for main.c:



Macros

- `#define APPLICATION_NAME "WLAN STATION"`
- `#define APPLICATION_VERSION "1.1.0"`
- `#define HOST_NAME "www.google.be"`
- `#define PING_INTERVAL 1000 /* In msecs */`
- `#define PING_TIMEOUT 3000 /* In msecs */`
- `#define PING_PKT_SIZE 20 /* In bytes */`
- `#define NO_OF_ATTEMPTS 3`
- `#define OSI_STACK_SIZE 2048`

Enumerations

- enum `e_AppStatusCodes` { `LAN_CONNECTION_FAILED` = -0x7D0, `INTERNET_CONNECTION_FAILED` = `LAN_CONNECTION_FAILED` - 1, `DEVICE_NOT_IN_STATION_MODE` = `INTERNET_CONNECTION_FAILED` - 1, `STATUS_CODE_MAX` = -0xBB8 }

Functions

- void `WlanStationMode` (void *pvParameters)
Start simplelink, connect to the ap and run the ping test.
- void `SimpleLinkWlanEventHandler` (SIWlanEvent_t *pWlanEvent)
The Function Handles WLAN Events.
- void `SimpleLinkNetAppEventHandler` (SINetAppEvent_t *pNetAppEvent)
This function handles network events such as IP acquisition, IP leased, IP released etc.
- void `SimpleLinkHttpServerCallback` (SIHttpServerEvent_t *pHttpEvent, SIHttpServerResponse_t *pHttpResponse)
This function handles HTTP server events.
- void `SimpleLinkGeneralEventHandler` (SIDeviceEvent_t *pDevEvent)
This function handles General Events.
- void `SimpleLinkSockEventHandler` (SISockEvent_t *pSock)
- void `main` ()

Variables

- unsigned long `g_ulStatus` = 0
- unsigned long `g_ulPingPacketsRecv` = 0
- unsigned long `g_ulGatewayIP` = 0
- unsigned char `g_ucConnectionSSID` [`SSID_LEN_MAX`+1]
- unsigned char `g_ucConnectionBSSID` [`BSSID_LEN_MAX`]

4.9 src/pinmux.c File Reference

```
#include "pinmux.h"
#include "hw_types.h"
#include "hw_memmap.h"
#include "hw_gpio.h"
#include "pin.h"
#include "rom.h"
#include "rom_map.h"
#include "gpio.h"
#include "prcm.h"
```

Include dependency graph for pinmux.c:



Functions

- void [PinMuxConfig](#) (void)

4.9.1 Function Documentation

4.9.1.1 void PinMuxConfig (void)

Definition at line 56 of file pinmux.c.

Referenced by main().

Here is the caller graph for this function:



Index

`__attribute__`
 `startup_gcc.c`, 35

`__init_data`
 `startup_gcc.c`, 37

`_bss`
 `startup_gcc.c`, 37

`_c_int00`
 `startup_gcc.c`, 36

`_data`
 `startup_gcc.c`, 38

`_ebss`
 `startup_gcc.c`, 38

`_edata`
 `startup_gcc.c`, 38

`_eheap`
 `startup_gcc.c`, 38

`_etext`
 `startup_gcc.c`, 38

`_heap`
 `startup_gcc.c`, 38

`_sbrk`
 `startup_gcc.c`, 36

APPLICATION_NAME
 `Getting_started_sta`, 6

APPLICATION_VERSION
 `Getting_started_sta`, 6

ASSERT_ON_ERROR
 `common.h`, 14

BSSID_LEN_MAX
 `common.h`, 14

CLR_STATUS_BIT
 `common.h`, 14

CLR_STATUS_BIT_ALL
 `common.h`, 14

CONSOLE
 `uart_if.h`, 46

CONSOLE_PERIPH
 `uart_if.h`, 46

ClearTerm
 `uart_if.c`, 39
 `uart_if.h`, 46

`common.h`
 ASSERT_ON_ERROR, 14
 BSSID_LEN_MAX, 14
 CLR_STATUS_BIT, 14
 CLR_STATUS_BIT_ALL, 14
 DBG_PRINT, 14
 e_StatusBits, 18
 ERR_PRINT, 15
 FAILURE, 15
 GET_STATUS_BIT, 15
 IS_CONNECT_FAILED, 15
 IS_CONNECTED, 15
 IS_IP_ACQUIRED, 15
 IS_IP_LEASED, 15
 IS_NW_PROCSR_ON, 15
 IS_P2P_DEV_FOUND, 16
 IS_P2P_REQ_RCVD, 16
 IS_PING_DONE, 16
 IS_SMART_CFG_START, 16
 LOOP_FOREVER, 16
 SECURITY_KEY, 16
 SECURITY_TYPE, 17
 SET_STATUS_BIT, 17
 SL_STOP_TIMEOUT, 17
 SPAWN_TASK_PRIORITY, 17
 SSID_LEN_MAX, 17
 SSID_NAME, 17
 STATUS_BIT_CONNECTION, 18
 STATUS_BIT_CONNECTION_FAILED, 18
 STATUS_BIT_IP_AQUIRED, 18
 STATUS_BIT_IP_LEASED, 18
 STATUS_BIT_NWP_INIT, 18
 STATUS_BIT_P2P_DEV_FOUND, 18
 STATUS_BIT_P2P_REQ_RECEIVED, 18
 STATUS_BIT_PING_DONE, 18
 STATUS_BIT_SMARTCONFIG_START, 18
 SUCCESS, 17
 UART_PRINT, 17
 UNUSED, 17
 `common/common.h`, 13
 `common/gpio_if.c`, 18
 `common/gpio_if.h`, 26
 `common/startup_gcc.c`, 35
 `common/uart_if.c`, 38
 `common/uart_if.h`, 45

DBG_PRINT
 `common.h`, 14

DEVICE_NOT_IN_STATION_MODE
 `Getting_started_sta`, 7

DispatcherUARTConfigure
 `uart_if.h`, 46

DispatcherUartSendPacket
 `uart_if.h`, 47

`e_AppStatusCodes`

- Getting_started_sta, 7
- e_StatusBits
 - common.h, 18
- ERR_PRINT
 - common.h, 15
- Error
 - uart_if.c, 39
 - uart_if.h, 47
- FAILURE
 - common.h, 15
- g_ucConnectionBSSID
 - Getting_started_sta, 11
- g_ucConnectionSSID
 - Getting_started_sta, 12
- g_ucLED1Pin
 - gpio_if.c, 25
- g_ucLED2Pin
 - gpio_if.c, 26
- g_ucLED3Pin
 - gpio_if.c, 26
- g_ucUARTBuffer
 - uart_if.h, 50
- g_uiLED1Port
 - gpio_if.c, 26
- g_uiLED2Port
 - gpio_if.c, 26
- g_uiLED3Port
 - gpio_if.c, 26
- g_ulGatewayIP
 - Getting_started_sta, 12
- g_ulPingPacketsRecv
 - Getting_started_sta, 12
- g_ulStatus
 - Getting_started_sta, 12
- GET_STATUS_BIT
 - common.h, 15
- GPIO_IF_ConfigureNIntEnable
 - gpio_if.c, 19
 - gpio_if.h, 28
- GPIO_IF_Get
 - gpio_if.c, 20
 - gpio_if.h, 28
- GPIO_IF_GetPortNPin
 - gpio_if.c, 20
 - gpio_if.h, 29
- GPIO_IF_LedConfigure
 - gpio_if.c, 21
 - gpio_if.h, 29
- GPIO_IF_LedOff
 - gpio_if.c, 22
 - gpio_if.h, 30
- GPIO_IF_LedOn
 - gpio_if.c, 22
 - gpio_if.h, 31
- GPIO_IF_LedStatus
 - gpio_if.c, 23
 - gpio_if.h, 32
- GPIO_IF_LedToggle
 - gpio_if.c, 24
 - gpio_if.h, 32
- GPIO_IF_Set
 - gpio_if.c, 25
 - gpio_if.h, 34
- GetCmd
 - uart_if.c, 41
 - uart_if.h, 47
- Getting_started_sta, 5
 - APPLICATION_NAME, 6
 - APPLICATION_VERSION, 6
 - DEVICE_NOT_IN_STATION_MODE, 7
 - e_AppStatusCodes, 7
 - g_ucConnectionBSSID, 11
 - g_ucConnectionSSID, 12
 - g_ulGatewayIP, 12
 - g_ulPingPacketsRecv, 12
 - g_ulStatus, 12
 - HOST_NAME, 6
 - INTERNET_CONNECTION_FAILED, 7
 - LAN_CONNECTION_FAILED, 7
 - main, 7
 - NO_OF_ATTEMPTS, 6
 - OSI_STACK_SIZE, 6
 - PING_INTERVAL, 6
 - PING_PKT_SIZE, 6
 - PING_TIMEOUT, 6
 - STATUS_CODE_MAX, 7
 - SimpleLinkGeneralEventHandler, 7
 - SimpleLinkHttpServerCallback, 9
 - SimpleLinkNetAppEventHandler, 9
 - SimpleLinkSockEventHandler, 9
 - SimpleLinkWlanEventHandler, 9
 - WlanStationMode, 11
- gpio_if.c
 - g_ucLED1Pin, 25
 - g_ucLED2Pin, 26
 - g_ucLED3Pin, 26
 - g_uiLED1Port, 26
 - g_uiLED2Port, 26
 - g_uiLED3Port, 26
 - GPIO_IF_ConfigureNIntEnable, 19
 - GPIO_IF_Get, 20
 - GPIO_IF_GetPortNPin, 20
 - GPIO_IF_LedConfigure, 21
 - GPIO_IF_LedOff, 22
 - GPIO_IF_LedOn, 22
 - GPIO_IF_LedStatus, 23
 - GPIO_IF_LedToggle, 24
 - GPIO_IF_Set, 25
 - PIN_LED1, 19
 - PIN_LED2, 19
 - PIN_LED3, 19
- gpio_if.h
 - GPIO_IF_ConfigureNIntEnable, 28
 - GPIO_IF_Get, 28
 - GPIO_IF_GetPortNPin, 29

- GPIO_IF_LedConfigure, [29](#)
- GPIO_IF_LedOff, [30](#)
- GPIO_IF_LedOn, [31](#)
- GPIO_IF_LedStatus, [32](#)
- GPIO_IF_LedToggle, [32](#)
- GPIO_IF_Set, [34](#)
- LED1, [27](#)
- LED2, [27](#)
- LED3, [27](#)
- ledEnum, [27](#)
- ledNames, [27](#)
- MCU_ALL_LED_IND, [28](#)
- MCU_ASSOCIATED_IND, [27](#)
- MCU_CLIENT_CONNECTED_IND, [27](#)
- MCU_EXECUTE_FAIL_IND, [28](#)
- MCU_EXECUTE_SUCCESS_IND, [28](#)
- MCU_GREEN_LED_GPIO, [28](#)
- MCU_IP_ALLOC_IND, [27](#)
- MCU_ON_IND, [27](#)
- MCU_ORANGE_LED_GPIO, [28](#)
- MCU_RED_LED_GPIO, [28](#)
- MCU_SENDING_DATA_IND, [27](#)
- MCU_SERVER_INIT_IND, [27](#)
- NO_LED, [27](#)
- NO_LED_IND, [27](#)
- HOST_NAME
 - Getting_started_sta, [6](#)
- INTERNET_CONNECTION_FAILED
 - Getting_started_sta, [7](#)
- IS_CONNECT_FAILED
 - common.h, [15](#)
- IS_CONNECTED
 - common.h, [15](#)
- IS_IP_ACQUIRED
 - common.h, [15](#)
- IS_IP_LEASED
 - common.h, [15](#)
- IS_NW_PROCSR_ON
 - common.h, [15](#)
- IS_P2P_DEV_FOUND
 - common.h, [16](#)
- IS_P2P_REQ_RCVD
 - common.h, [16](#)
- IS_PING_DONE
 - common.h, [16](#)
- IS_SMART_CFG_START
 - common.h, [16](#)
- IS_SPACE
 - uart_if.c, [39](#)
- ilen
 - uart_if.c, [45](#)
- inc/pinmux.h, [50](#)
- InitTerm
 - uart_if.c, [41](#)
 - uart_if.h, [47](#)
- LAN_CONNECTION_FAILED
 - Getting_started_sta, [7](#)
- LED1
 - gpio_if.h, [27](#)
- LED2
 - gpio_if.h, [27](#)
- LED3
 - gpio_if.h, [27](#)
- LOOP_FOREVER
 - common.h, [16](#)
- ledEnum
 - gpio_if.h, [27](#)
- ledNames
 - gpio_if.h, [27](#)
- MCU_ALL_LED_IND
 - gpio_if.h, [28](#)
- MCU_ASSOCIATED_IND
 - gpio_if.h, [27](#)
- MCU_CLIENT_CONNECTED_IND
 - gpio_if.h, [27](#)
- MCU_EXECUTE_FAIL_IND
 - gpio_if.h, [28](#)
- MCU_EXECUTE_SUCCESS_IND
 - gpio_if.h, [28](#)
- MCU_GREEN_LED_GPIO
 - gpio_if.h, [28](#)
- MCU_IP_ALLOC_IND
 - gpio_if.h, [27](#)
- MCU_ON_IND
 - gpio_if.h, [27](#)
- MCU_ORANGE_LED_GPIO
 - gpio_if.h, [28](#)
- MCU_RED_LED_GPIO
 - gpio_if.h, [28](#)
- MCU_SENDING_DATA_IND
 - gpio_if.h, [27](#)
- MCU_SERVER_INIT_IND
 - gpio_if.h, [27](#)
- main
 - Getting_started_sta, [7](#)
- Message
 - uart_if.c, [42](#)
 - uart_if.h, [48](#)
- NO_LED
 - gpio_if.h, [27](#)
- NO_LED_IND
 - gpio_if.h, [27](#)
- NO_OF_ATTEMPTS
 - Getting_started_sta, [6](#)
- OSI_STACK_SIZE
 - Getting_started_sta, [6](#)
- PIN_LED1
 - gpio_if.c, [19](#)
- PIN_LED2
 - gpio_if.c, [19](#)
- PIN_LED3

- gpio_if.c, [19](#)
- PING_INTERVAL
 - Getting_started_sta, [6](#)
- PING_PKT_SIZE
 - Getting_started_sta, [6](#)
- PING_TIMEOUT
 - Getting_started_sta, [6](#)
- PinMuxConfig
 - pinmux.c, [53](#)
 - pinmux.h, [51](#)
- pinmux.c
 - PinMuxConfig, [53](#)
- pinmux.h
 - PinMuxConfig, [51](#)
- Report
 - uart_if.c, [42](#)
 - uart_if.h, [49](#)
- ResetISR
 - startup_gcc.c, [36](#)
- SECURITY_KEY
 - common.h, [16](#)
- SECURITY_TYPE
 - common.h, [17](#)
- SET_STATUS_BIT
 - common.h, [17](#)
- SL_STOP_TIMEOUT
 - common.h, [17](#)
- SPAWN_TASK_PRIORITY
 - common.h, [17](#)
- SSID_LEN_MAX
 - common.h, [17](#)
- SSID_NAME
 - common.h, [17](#)
- STATUS_BIT_CONNECTION
 - common.h, [18](#)
- STATUS_BIT_CONNECTION_FAILED
 - common.h, [18](#)
- STATUS_BIT_IP_AQUIRED
 - common.h, [18](#)
- STATUS_BIT_IP_LEASED
 - common.h, [18](#)
- STATUS_BIT_NWP_INIT
 - common.h, [18](#)
- STATUS_BIT_P2P_DEV_FOUND
 - common.h, [18](#)
- STATUS_BIT_P2P_REQ_RECEIVED
 - common.h, [18](#)
- STATUS_BIT_PING_DONE
 - common.h, [18](#)
- STATUS_BIT_SMARTCONFIG_START
 - common.h, [18](#)
- STATUS_CODE_MAX
 - Getting_started_sta, [7](#)
- SUCCESS
 - common.h, [17](#)
- SYSCLK
 - uart_if.h, [46](#)
- SimpleLinkGeneralEventHandler
 - Getting_started_sta, [7](#)
- SimpleLinkHttpServerCallback
 - Getting_started_sta, [9](#)
- SimpleLinkNetAppEventHandler
 - Getting_started_sta, [9](#)
- SimpleLinkSockEventHandler
 - Getting_started_sta, [9](#)
- SimpleLinkWlanEventHandler
 - Getting_started_sta, [9](#)
- src/main.c, [51](#)
- src/pinmux.c, [52](#)
- startup_gcc.c
 - __attribute__, [35](#)
 - __init_data, [37](#)
 - _bss, [37](#)
 - _c_int00, [36](#)
 - _data, [38](#)
 - _ebss, [38](#)
 - _edata, [38](#)
 - _eheap, [38](#)
 - _etext, [38](#)
 - _heap, [38](#)
 - _sbrk, [36](#)
 - ResetISR, [36](#)
 - vPortSVCHandler, [36](#)
 - xPortPendSVHandler, [37](#)
 - xPortSysTickHandler, [37](#)
- TrimSpace
 - uart_if.c, [44](#)
 - uart_if.h, [50](#)
- UART_BAUD_RATE
 - uart_if.h, [46](#)
- UART_IF_BUFFER
 - uart_if.h, [46](#)
- UART_PRINT
 - common.h, [17](#)
- UNUSED
 - common.h, [17](#)
- uart_if.c
 - ClearTerm, [39](#)
 - Error, [39](#)
 - GetCmd, [41](#)
 - IS_SPACE, [39](#)
 - ilen, [45](#)
 - InitTerm, [41](#)
 - Message, [42](#)
 - Report, [42](#)
 - TrimSpace, [44](#)
- uart_if.h
 - CONSOLE, [46](#)
 - CONSOLE_PERIPH, [46](#)
 - ClearTerm, [46](#)
 - DispatcherUARTConfigure, [46](#)
 - DispatcherUartSendPacket, [47](#)
 - Error, [47](#)
 - g_ucUARTBuffer, [50](#)

- GetCmd, [47](#)
- InitTerm, [47](#)
- Message, [48](#)
- Report, [49](#)
- SYSCLK, [46](#)
- TrimSpace, [50](#)
- UART_BAUD_RATE, [46](#)
- UART_IF_BUFFER, [46](#)

- vPortSVCHandler
 - startup_gcc.c, [36](#)

- WlanStationMode
 - Getting_started_sta, [11](#)

- xPortPendSVHandler
 - startup_gcc.c, [37](#)
- xPortSysTickHandler
 - startup_gcc.c, [37](#)