SWC_LED

Version v1.0
7/16/2023 2:59:00 AM

Table of Contents

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

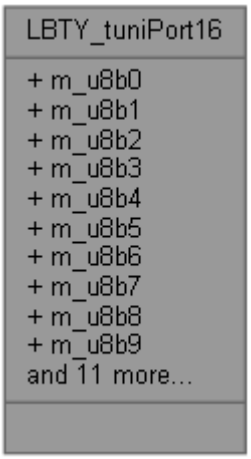
File List

Here is a list of all files with brief descriptions:

Data Structure Documentation

LBTY_tuniPort16 Union Reference

`#include <LBTY_int.h>`
Collaboration diagram for LBTY_tuniPort16:



Data Fields

- struct {
- u8 m_u8b0:1
- u8 m_u8b1:1
- u8 m_u8b2:1
- u8 m_u8b3:1
- u8 m_u8b4:1
- u8 m_u8b5:1
- u8 m_u8b6:1
- u8 m_u8b7:1
- u8 m_u8b8:1
- u8 m_u8b9:1
- u8 m_u8b10:1
- u8 m_u8b11:1
- u8 m_u8b12:1
- u8 m_u8b13:1
- u8 m_u8b14:1
- u8 m_u8b15:1
- } sBits
- struct {
- u8 m_u8low
- u8 m_u8high
- } sBytes
- u16 u_u16Word

Field Documentation

u8 m_u8b0

u8 m_u8b1

u8 m_u8b10

u8 m_u8b11

u8 m_u8b12

u8 m_u8b13

u8 m_u8b14

u8 m_u8b15

u8 m_u8b2

u8 m_u8b3

u8 m_u8b4

u8 m_u8b5

u8 m_u8b6

u8 m_u8b7

u8 m_u8b8

u8 m_u8b9

u8 m_u8high

u8 m_u8low

struct { ... } sBits

struct { ... } sBytes

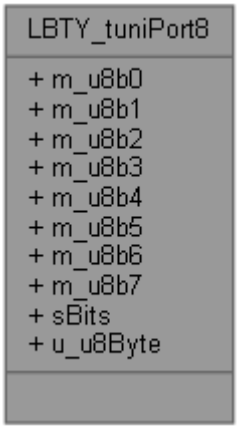
u16 u_u16Word

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

- H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h

LBTY_tuniPort8 Union Reference

`#include <LBTY_int.h>`
Collaboration diagram for LBTY_tuniPort8:



Data Fields

- struct {
- u8 m_u8b0:1
- u8 m_u8b1:1
- u8 m_u8b2:1
- u8 m_u8b3:1
- u8 m_u8b4:1
- u8 m_u8b5:1
- u8 m_u8b6:1
- u8 m_u8b7:1
- } sBits
- u8 u_u8Byte

Detailed Description

Union Byte bit by bit

Field Documentation

u8 m_u8b0

u8 m_u8b1

u8 m_u8b2

u8 m_u8b3

u8 m_u8b4

u8 m_u8b5

u8 m_u8b6

u8 m_u8b7

struct { ... } sBits

u8 u_u8Byte

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

- H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h

LED_tstrConfig Struct Reference

: type define of structure for Led Configuration

`#include <LED_int.h>`
Collaboration diagram for LED_tstrConfig:

Data Fields

- LED_tenuConnection m_Connection
- GPIO_tstrPinConfig m_PinCfg

Detailed Description

: type define of structure for Led Configuration

Type : struct **Unit** : None

Field Documentation

LED_tenuConnection m_Connection

Push Active

GPIO_tstrPinConfig m_PinCfg

Pin Configuration

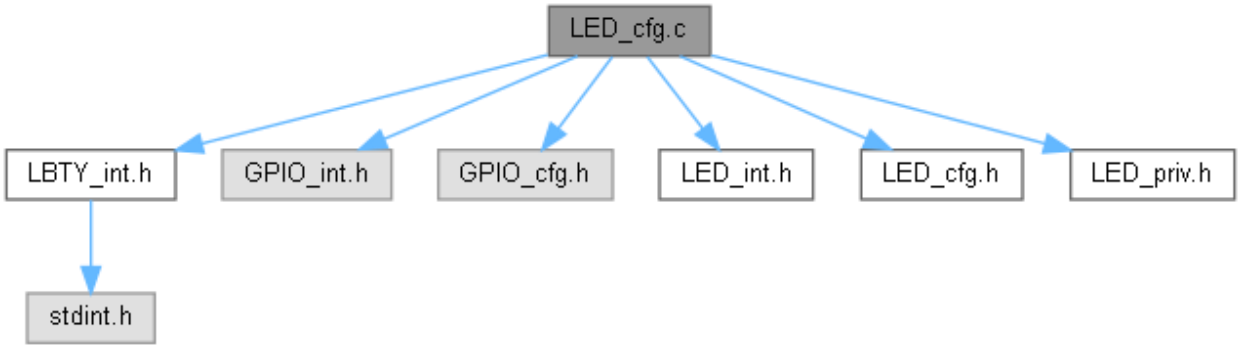
The documentation for this struct was generated from the following file:
LED_int.h

File Documentation

LED_cfg.c File Reference

```
#include "LBTY_int.h"
#include "GPIO_int.h"
#include "GPIO_cfg.h"
#include "LED_int.h"
#include "LED_cfg.h"
#include "LED_priv.h"
```

Include dependency graph for LED_cfg.c:



Variables

- const LED_tstrConfig kau8LEDConfiguration_LGB [LED_Num]

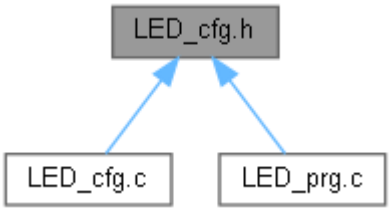
Variable Documentation

const LED_tstrConfig kau8LEDConfiguration_LGB[LED_Num]

```
Initial value:= {

         {.m_Connection = LED0_CON, .m_PinCfg =
        {.m_Port = LED0_PORT, .m_Pin = LED0_PIN, .m_Dir = PIN_OUTPUT, .m_Value =
LED0_CON}}


        ,{.m_Connection = LED1_CON, .m_PinCfg =
        {.m_Port = LED1_PORT, .m_Pin = LED1_PIN, .m_Dir = PIN_OUTPUT, .m_Value =
LED1_CON}}


        ,{.m_Connection = LED2_CON, .m_PinCfg =
        {.m_Port = LED2_PORT, .m_Pin = LED2_PIN, .m_Dir = PIN_OUTPUT, .m_Value =
LED2_CON}}


        ,{.m_Connection = LED3_CON, .m_PinCfg =
        {.m_Port = LED3_PORT, .m_Pin = LED3_PIN, .m_Dir = PIN_OUTPUT, .m_Value =
LED3_CON}}


        ,{.m_Connection = LED4_CON, .m_PinCfg =
        {.m_Port = LED4_PORT, .m_Pin = LED4_PIN, .m_Dir = PIN_OUTPUT, .m_Value =
LED4_CON}}


        ,{.m_Connection = LED5_CON, .m_PinCfg =
        {.m_Port = LED5_PORT, .m_Pin = LED5_PIN, .m_Dir = PIN_OUTPUT, .m_Value =
LED5_CON}}}
```

LED_cfg.h File Reference

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



Macros

- #define LED0 0
- #define LED0_PORT B
- #define LED0_PIN 0
- #define LED0_CON LED_Forward
- #define LED1 1
- #define LED1_PORT B
- #define LED1_PIN 1
- #define LED1_CON LED_Forward
- #define LED2 2
- #define LED2_PORT B
- #define LED2_PIN 2
- #define LED2_CON LED_Forward
- #define LED3 3
- #define LED3_PORT B
- #define LED3_PIN 3
- #define LED3_CON LED_Forward
- #define LED4 4
- #define LED4_PORT B
- #define LED4_PIN 4
- #define LED4_CON LED_Forward
- #define LED5 5
- #define LED5_PORT B
- #define LED5_PIN 5
- #define LED5_CON LED_Forward
- #define LED6 6
- #define LED6_PORT B
- #define LED6_PIN 6
- #define LED6_CON LED_Forward
- #define LED7 7
- #define LED7_PORT B
- #define LED7_PIN 7
- #define LED7_CON LED_Forward

Macro Definition Documentation

#define LED0 0

#define LED0_CON LED_Forward

#define LED0_PIN 0

#define LED0_PORT B

#define LED1 1

#define LED1_CON LED_Forward

#define LED1_PIN 1

#define LED1_PORT B

#define LED2 2

#define LED2_CON LED_Forward

#define LED2_PIN 2

#define LED2_PORT B

#define LED3 3

#define LED3_CON LED_Forward

#define LED3_PIN 3

#define LED3_PORT B

#define LED4 4

#define LED4_CON LED_Forward

#define LED4_PIN 4

#define LED4_PORT B

#define LED5 5

#define LED5_CON LED_Forward

#define LED5_PIN 5

#define LED5_PORT B

#define LED6 6

#define LED6_CON LED_Forward

#define LED6_PIN 6

#define LED6_PORT B

#define LED7 7

#define LED7_CON LED_Forward

#define LED7_PIN 7

#define LED7_PORT B

LED_cfg.h

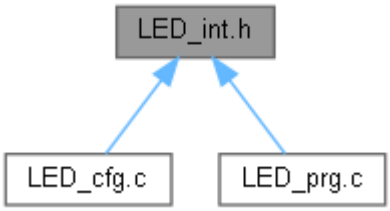
Go to the documentation of this file.

```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************** */
4 /* File Name    : LED_cfg.h                                                   */
5 /* Author       : MAAM                                                        */
6 /* Version      : v01.2                                                       */
7 /* date         : Apr 8, 2023                                                 */
8 /* ************************************************************************** */
9 /* ************************* HEADER FILES INCLUDES ************************  */
10 /* ************************************************************************** */
11
12 #ifndef LED_CFG_H_
13 #define LED_CFG_H_
14
15 /* ************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ************************************************************************** */
18
19 /* ************************************************************************** */
20 /* ************************** MACRO/DEFINE SECTION ************************** */
21 /* ************************************************************************** */
22
23 #if defined(AMIT_KIT)
24
25 #define LED0            0
26 #define LED0_PORT       D
27 #define LED0_PIN        AMIT_LED0
28 #define LED0_CON        LED_Forward
29
30 #define LED1            1
31 #define LED1_PORT       D
32 #define LED1_PIN        AMIT_LED1
33 #define LED1_CON        LED_Forward
34
35 #define LED2            2
36 #define LED2_PORT       D
37 #define LED2 PIN        AMIT LED2
38 #define LED2_CON        LED_Forward
39
40 #elif defined(ETA32_KIT)
41
42 #define LED0            0
43 #define LED0 PORT       B
44 #define LED0_PIN        Eta32_LED_R
45 #define LED0_CON        LED_Forward
46
47 #define LED1            1
48 #define LED1_PORT       A
49 #define LED1 PIN        Eta32 LED G
50 #define LED1_CON        LED_Forward
51
52 #define LED2            2
53 #define LED2_PORT       A
54 #define LED2 PIN        Eta32 LED B
55 #define LED2_CON        LED_Forward
56
57 #define LED3            3
58 #define LED3_PORT       A
59 #define LED3_PIN        Eta32_LED_Y
60 #define LED3 CON        LED Forward
61
62 #elif defined(ETA32_MINI_KIT)
63
64 #define LED0            0
65 #define LED0_PORT       C
66 #define LED0 PIN        Eta32 mini LED R
67 #define LED0_CON        LED_Forward
68
69 #define LED1            1
70 #define LED1_PORT       C
71 #define LED1_PIN        Eta32_mini_LED_G
72 #define LED1_CON        LED_Forward
```

```
73
74 #define LED2            2
75 #define LED2_PORT       C
76 #define LED2_PIN        Eta32_mini_LED_B
77 #define LED2_CON        LED_Forward
78
79 #else
80
81 #define LED0            0
82 #define LED0_PORT       B
83 #define LED0_PIN        0
84 #define LED0_CON        LED_Forward
85
86 #define LED1            1
87 #define LED1_PORT       B
88 #define LED1_PIN        1
89 #define LED1_CON        LED_Forward
90
91 #define LED2            2
92 #define LED2_PORT       B
93 #define LED2_PIN        2
94 #define LED2_CON        LED_Forward
95
96 #define LED3            3
97 #define LED3_PORT       B
98 #define LED3_PIN        3
99 #define LED3_CON        LED_Forward
100
101 #define LED4             4
102 #define LED4_PORT        B
103 #define LED4_PIN         4
104 #define LED4_CON         LED_Forward
105
106 #define LED5             5
107 #define LED5_PORT        B
108 #define LED5_PIN         5
109 #define LED5_CON         LED_Forward
110
111 #define LED6             6
112 #define LED6_PORT        B
113 #define LED6_PIN         6
114 #define LED6_CON         LED_Forward
115
116 #define LED7             7
117 #define LED7_PORT        B
118 #define LED7_PIN         7
119 #define LED7_CON         LED_Forward
120
121 #endif
122
123 /* ********************************************************************************* */
124 /* ****************************** CONST SECTION ************************************ */
125 /* ********************************************************************************* */
126
127 /* ********************************************************************************* */
128 /* ***************************** VARIABLE SECTION ********************************** */
129 /* ********************************************************************************* */
130
131 /* ********************************************************************************* */
132 /* ***************************** FUNCTION SECTION ********************************** */
133 /* ********************************************************************************* */
134
135
136 #endif /* LED_CFG_H_ */
137 /***************************** E N D (LED_cfg.h) ******************************/
```

LED_int.h File Reference

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



Data Structures

struct LED_tstrConfig: *type define of structure for Led Configuration*

Enumerations

- enum LED_tenuConnection { LED_Forward = (u8)0u, LED_Reverse }
- enum LED_tenuStatus { LED_OFF = (u8)0u, LED_ON }

Functions

- void LED_vidInit (u8 u8LedNum)
- void LED_vidInitAll (void)
- LBTY_tenuErrorStatus LED_u8SetON (u8 u8LedNum)
- LBTY_tenuErrorStatus LED_u8SetOFF (u8 u8LedNum)
- LBTY_tenuErrorStatus LED_u8Toggle (u8 u8LedNum)

Enumeration Type Documentation

enum LED_tenuConnection

Enumerator:

LED_Forward	
LED_Reverse	

```
19              {
20      LED_Forward = (u8)0u,
21      LED_Reverse
22  }LED_tenuConnection;
```

enum LED_tenuStatus

Enumerator:

LED_OFF	
LED_ON	

```
24              {
25      LED_OFF = (u8)0u,
26      LED_ON
27  }LED_tenuStatus;
```

Function Documentation

LBTY_tenuErrorStatus LED_u8SetOFF (u8 *u8LedNum*)

```
83                                                  {
84      LED_tstrConfig* pstrLed =
85              (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
86      return GPIO_u8SetPinValue(pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin
87              ,(pstrLed->m_Connection == LED_Forward) ? LED_Forward : LED_Reverse);
88  }
```

LBTY_tenuErrorStatus LED_u8SetON (u8 *u8LedNum*)

```
71                                                  {
72      LED_tstrConfig* pstrLed =
73              (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
74      return GPIO_u8SetPinValue(pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin
75              ,(pstrLed->m_Connection == LED_Forward) ? LED_Reverse : LED_Forward);
76  }
```

LBTY_tenuErrorStatus LED_u8Toggle (u8 *u8LedNum*)

```
95                                                  {
96      LED_tstrConfig* pstrLed =
97              (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
98      return GPIO_u8TogglePinValue(pstrLed->m_PinCfg.m_Port,
pstrLed->m_PinCfg.m_Pin);
99  }
```

void LED_vidInit (u8 *u8LedNum*)

```
46                                  {
47  //  LED_tstrConfig* pstrLed =
48  //          (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
49  //  GPIO_u8SetPinDirection(pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin,
PIN_OUTPUT);
50  //  GPIO_u8SetPinValue    (pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin
51  //              ,(pstrLed->m_Connection == LED_Forward) ?
pstrLed->m_PinCfg.m_Value : !pstrLed->m_PinCfg.m_Value);
52      GPIO_u8PinInit(kau8LEDConfiguration_LGB[u8LedNum].m_PinCfg);
53  }
```

void LED_vidInitAll (void)

```
60                                  {
61      for(u8_i = LED_Num_; i-- ; ){
62          GPIO_u8PinInit(kau8LEDConfiguration_LGB[i].m_PinCfg);
63      }
64  }
```

LED_int.h

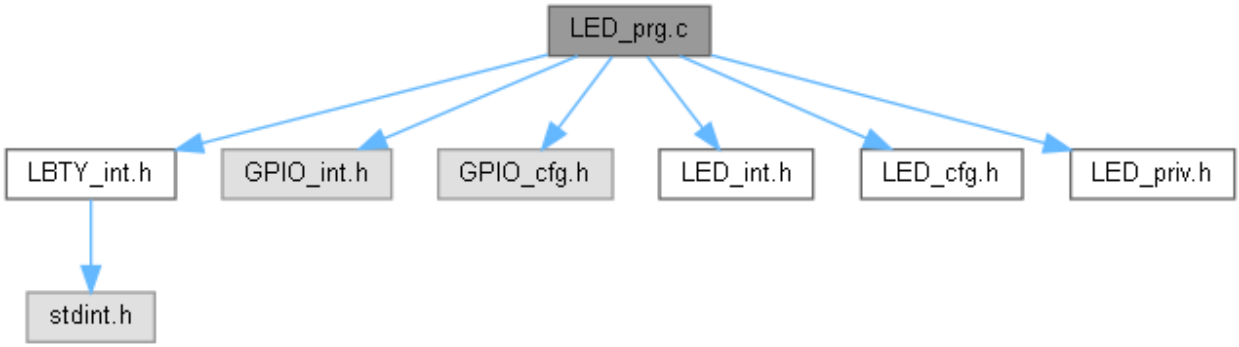
```
Go to the documentation of this file.1 /*
****************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ****************************************************************************** */
4 /* File Name    : LED_int.h                                                   */
5 /* Author       : MAAM                                                        */
6 /* Version      : v01.2                                                       */
7 /* date         : Apr 8, 2023                                                 */
8 /* ****************************************************************************** */
9 /* ************************ HEADER FILES INCLUDES ************************  */
10 /* ****************************************************************************** */
11
12 #ifndef LED_INT_H_
13 #define LED_INT_H_
14
15 /* ****************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ****************************************************************************** */
18
19 typedef enum{
20     LED_Forward = (u8)0u,
21     LED_Reverse
22 }LED_tenuConnection;
23
24 typedef enum{
25     LED_OFF = (u8)0u,
26     LED_ON
27 }LED_tenuStatus;
28
29
/*******************************************************************************************
*******************************/
30
33 typedef struct{
34     LED_tenuConnection  m_Connection;
35     GPIO_tstrPinConfig  m_PinCfg;
36 }LED_tstrConfig;
37
38 /* ****************************************************************************** */
39 /* ************************** MACRO/DEFINE SECTION **************************** */
40 /* ****************************************************************************** */
41
42 /* ****************************************************************************** */
43 /* ***************************** CONST SECTION ********************************** */
44 /* ****************************************************************************** */
45
46 /* ****************************************************************************** */
47 /* ***************************** VARIABLE SECTION ******************************* */
48 /* ****************************************************************************** */
49
50 /* ****************************************************************************** */
51 /* ***************************** FUNCTION SECTION ******************************* */
52 /* ****************************************************************************** */
53
54 /* ****************************************************************************** */
55 /* Description : Initialize the LED Button direction                            */
56 /* Input       :    u8LedNum                                                    */
57 /* Return      :    void                                                        */
58 /* ****************************************************************************** */
59 extern void LED_vidInit(u8 u8LedNum);
60
61 /* ****************************************************************************** */
62 /* Description : Initialize the All LED Button with Configurations              */
63 /* Input       :    void                                                        */
64 /* Return      :    void                                                        */
65 /* ****************************************************************************** */
66 extern void LED_vidInitAll(void);
67
68 /* ****************************************************************************** */
69 /* Description :    Set the LED ON                                              */
70 /* Input       :    u8LedNum                                                    */
71 /* Return      :    LBTY_tenuErrorStatus                                        */
72 /* ****************************************************************************** */
```

```
73 extern LBTY_tenuErrorStatus LED_u8SetON(u8 u8LedNum);
74
75 /* ************************************************************************** */
76 /* Description :    Set the LED OFF                                           */
77 /* Input       :    u8LedNum                                                  */
78 /* Return      :    LBTY_tenuErrorStatus                                      */
79 /* ************************************************************************** */
80 extern LBTY_tenuErrorStatus LED_u8SetOFF(u8 u8LedNum);
81
82 /* ************************************************************************** */
83 /* Description :    Toggle the LED                                            */
84 /* Input       :    u8LedNum                                                  */
85 /* Return      :    LBTY_tenuErrorStatus                                      */
86 /* ************************************************************************** */
87 extern LBTY_tenuErrorStatus LED_u8Toggle(u8 u8LedNum);
88
89 #endif /* LED_INT_H_ */
90 /***************************** E N D (LED_int.h) ******************************/
```

LED_prg.c File Reference

```
#include "LBTY_int.h"
#include "GPIO_int.h"
#include "GPIO_cfg.h"
#include "LED_int.h"
#include "LED_cfg.h"
#include "LED_priv.h"
```

Include dependency graph for LED_prg.c:



Functions

- void LED_vidInit (u8 u8LedNum)
- void LED_vidInitAll (void)
- LBTY_tenuErrorStatus LED_u8SetON (u8 u8LedNum)
- LBTY_tenuErrorStatus LED_u8SetOFF (u8 u8LedNum)
- LBTY_tenuErrorStatus LED_u8Toggle (u8 u8LedNum)

Variables

- const LED_tstrConfig kau8LEDConfiguration_LGB [LED_Num]

Function Documentation

LBTY_tenuErrorStatus LED_u8SetOFF (u8 *u8LedNum*)

```
83                                                  {
84     LED_tstrConfig* pstrLed =
85             (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
86     return GPIO_u8SetPinValue(pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin
87             ,(pstrLed->m_Connection == LED_Forward) ? LED_Forward : LED_Reverse);
88 }
```

LBTY_tenuErrorStatus LED_u8SetON (u8 *u8LedNum*)

```
71                                                  {
72     LED_tstrConfig* pstrLed =
73             (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
74     return GPIO_u8SetPinValue(pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin
75             ,(pstrLed->m_Connection == LED_Forward) ? LED_Reverse : LED_Forward);
76 }
```

LBTY_tenuErrorStatus LED_u8Toggle (u8 *u8LedNum*)

```
95                                                  {
96     LED_tstrConfig* pstrLed =
97             (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
98     return GPIO_u8TogglePinValue(pstrLed->m_PinCfg.m_Port,
pstrLed->m_PinCfg.m_Pin);
99 }
```

void LED_vidInit (u8 *u8LedNum*)

```
46                                  {
47 //  LED_tstrConfig* pstrLed =
48 //          (LED_tstrConfig*)&kau8LEDConfiguration_LGB[u8LedNum];
49 //  GPIO_u8SetPinDirection(pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin,
PIN_OUTPUT);
50 //  GPIO_u8SetPinValue    (pstrLed->m_PinCfg.m_Port, pstrLed->m_PinCfg.m_Pin
51 //              ,(pstrLed->m_Connection == LED_Forward) ?
pstrLed->m_PinCfg.m_Value : !pstrLed->m_PinCfg.m_Value);
52     GPIO_u8PinInit(kau8LEDConfiguration_LGB[u8LedNum].m_PinCfg);
53 }
```

void LED_vidInitAll (void)

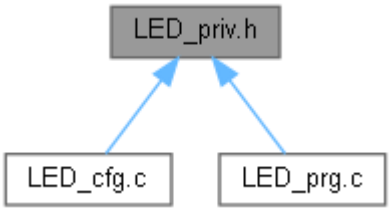
```
60                                  {
61     for(u8 i = LED_Num ; i-- ; ){
62         GPIO_u8PinInit(kau8LEDConfiguration_LGB[i].m_PinCfg);
63     }
64 }
```

Variable Documentation

const LED_tstrConfig kau8LEDConfiguration_LGB[LED_Num] [extern]

LED_priv.h File Reference

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



Enumerations

- enum LED_tenuLedNum { LED_Num }

Enumeration Type Documentation

enum LED_tenuLedNum

Enumerator:

LED_Num	

```
19              {
20 #ifdef LED0
21     LED_0 = (u8)0u
22 #endif
23 #ifdef LED1
24     ,LED_1
25 #endif
26 #ifdef LED2
27     ,LED_2
28 #endif
29 #ifdef LED3
30     ,LED_3
31 #endif
32 #ifdef LED4
33     ,LED_4
34 #endif
35 #ifdef LED5
36     ,LED_5
37 #endif
38 #ifdef LED6
39     ,LED_6
40 #endif
41 #ifdef LED7
42     ,LED_7
43 #endif
44     ,LED_Num
45 }LED_tenuLedNum;
```

LED_priv.h

```
Go to the documentation of this file.1 /*
****************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ****************************************************************************** */
4 /* File Name    : LED_priv.h                                                  */
5 /* Author       : MAAM                                                        */
6 /* Version      : v01.2                                                       */
7 /* date         : Apr 8, 2023                                                 */
8 /* ****************************************************************************** */
9 /* ************************* HEADER FILES INCLUDES ************************  */
10 /* ****************************************************************************** */
11
12 #ifndef LED_PRIV_H_
13 #define LED_PRIV_H_
14
15 /* ****************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ****************************************************************************** */
18
19 typedef enum{
20 #ifdef LED0
21     LED_0 = (u8)0u
22 #endif
23 #ifdef LED1
24     ,LED_1
25 #endif
26 #ifdef LED2
27     ,LED_2
28 #endif
29 #ifdef LED3
30     ,LED_3
31 #endif
32 #ifdef LED4
33     ,LED_4
34 #endif
35 #ifdef LED5
36     ,LED_5
37 #endif
38 #ifdef LED6
39     ,LED_6
40 #endif
41 #ifdef LED7
42     ,LED_7
43 #endif
44     ,LED_Num
45 }LED_tenuLedNum;
46
47 /* ****************************************************************************** */
48 /* ************************** MACRO/DEFINE SECTION ************************** */
49 /* ****************************************************************************** */
50
51 /* ****************************************************************************** */
52 /* ***************************** CONST SECTION ********************************** */
53 /* ****************************************************************************** */
54
55 /* ****************************************************************************** */
56 /* ***************************** VARIABLE SECTION ******************************* */
57 /* ****************************************************************************** */
58
59 /* ****************************************************************************** */
60 /* ***************************** FUNCTION SECTION ******************************* */
61 /* ****************************************************************************** */
62
63 #endif /* LED_PRIV_H_ */
64 /***************************** E N D (GPIO_priv.h) ******************************/
```

main.c File Reference

H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBIT_int.h File Reference

Macros

- #define _BV(bit) (1u<<(bit))
- #define SET_BIT(REG, bit) ((REG) |= (1u<<(bit)))
- #define CLR_BIT(REG, bit) ((REG) &= ~(1u<<(bit)))
- #define TOG_BIT(REG, bit) ((REG) ^= (1u<<(bit)))
- #define SET_BYTE(REG, bit) ((REG) |= (0xFFu<<(bit)))
- #define CLR_BYTE(REG, bit) ((REG) &= ~(0xFFu<<(bit)))
- #define TOG_BYTE(REG, bit) ((REG) ^= (0xFFu<<(bit)))
- #define SET_MASK(REG, MASK) ((REG) |= (MASK))
- #define CLR_MASK(REG, MASK) ((REG) &= ~(MASK))
- #define TOG_MASK(REG, MASK) ((REG) ^= (MASK))
- #define GET_MASK(REG, MASK) ((REG) & (MASK))
- #define SET_REG(REG) ((REG) = ~(0u))
- #define CLR_REG(REG) ((REG) = (0u))
- #define TOG_REG(REG) ((REG) ^= ~(0u))
- #define GET_BIT(REG, bit) (((REG)>>(bit)) & 0x01u)
- #define GET_NIB(REG, bit) (((REG)>>(bit)) & 0x0Fu)
- #define GET_BYTE(REG, bit) (((REG)>>(bit)) & 0xFFu)
- #define ASSIGN_BIT(REG, bit, value) ((REG) = ((REG) & ~(0x01u<<(bit))) | (((value) & 0x01u)<<(bit)))
- #define ASSIGN_NIB(REG, bit, value) ((REG) = ((REG) & ~(0x0Fu<<(bit))) | (((value) & 0x0Fu)<<(bit)))
- #define ASSIGN_BYTE(REG, bit, value) ((REG) = ((REG) & ~(0xFFu<<(bit))) | (((value) & 0xFFu)<<(bit)))
- #define CON_u8Bits(b7, b6, b5, b4, b3, b2, b1, b0)

  (0b##b7##b6##b5##b4##b3##b2##b1##b0)
- #define CON_u16Bits(b15, b14, b13, b12, b11, b10, b9, b8, b7, b6, b5, b4, b3, b2, b1, b0)

  (0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2##b1##b0)

Macro Definition Documentation

#define _BV(bit) (1u<<(bit))

#define ASSIGN_BIT(REG, bit, value) ((REG) = ((REG) & ~(0x01u<<(bit))) | (((value) & 0x01u)<<(bit)))

#define ASSIGN_BYTE(REG, bit, value) ((REG) = ((REG) & ~(0xFFu<<(bit))) | (((value) & 0xFFu)<<(bit)))

#define ASSIGN_NIB(REG, bit, value) ((REG) = ((REG) & ~(0x0Fu<<(bit))) | (((value) & 0x0Fu)<<(bit)))

#define CLR_BIT(REG, bit) ((REG) &= ~(1u<<(bit)))

#define CLR_BYTE(REG, bit) ((REG) &= ~(0xFFu<<(bit)))

#define CLR_MASK(REG, MASK) ((REG) &= ~(MASK))

#define CLR_REG(REG) ((REG) = (0u))

#define CON_u16Bits(b15, b14, b13, b12, b11, b10, b9, b8, b7, b6, b5, b4, b3, b2, b1, b0)

(0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2##b1##b0)

#define CON_u8Bits(b7, b6, b5, b4, b3, b2, b1, b0)

(0b##b7##b6##b5##b4##b3##b2##b1##b0)

#define GET_BIT(REG, bit) (((REG)>>(bit)) & 0x01u)

#define GET_BYTE(REG, bit) (((REG)>>(bit)) & 0xFFu)

#define GET_MASK(REG, MASK) ((REG) & (MASK))

#define GET_NIB(REG, bit) (((REG)>>(bit)) & 0x0Fu)

#define SET_BIT(REG, bit) ((REG) |= (1u<<(bit)))

Bitwise Operation

```
#define SET_BYTE( REG,   bit)   ((REG) |=   (0xFFu<<(bit)))

#define SET_MASK( REG,   MASK)   ((REG) |=   (MASK))

#define SET_REG( REG)   ((REG)   = ~(0u))

#define TOG_BIT( REG,   bit)   ((REG) ^=   (1u<<(bit)))

#define TOG_BYTE( REG,   bit)   ((REG) ^=   (0xFFu<<(bit)))

#define TOG_MASK( REG,   MASK)   ((REG) ^=   (MASK))

#define TOG_REG( REG)   ((REG) ^= ~(0u))
```

LBIT_int.h

Go to the documentation of this file.

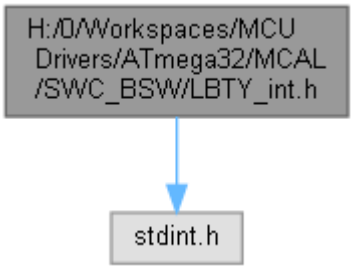
```
1 /*
******************************************************************************* */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ******************************************************************************* */
4 /* File Name   : LBIT_int.h                                                    */
5 /* Author      : MAAM                                                          */
6 /* Version     : v01                                                           */
7 /* date        : Mar 24, 2023                                                  */
8 /* description : Bitwise Library                                               */
9 /* ******************************************************************************* */
10 /* ************************ HEADER FILES INCLUDES ************************  */
11 /* ******************************************************************************* */
12
13 #ifndef LBIT_INT_H_
14 #define LBIT_INT_H_
15
16 /* ******************************************************************************* */
17 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
18 /* ******************************************************************************* */
19
20 /* ******************************************************************************* */
21 /* ************************** MACRO/DEFINE SECTION ************************** */
22 /* ******************************************************************************* */
23
24 #define _BV(bit)                                          (1u<<(bit))
25
27 #define SET_BIT(REG, bit)                                 ((REG) |=  (1u<<(bit)))
28 #define CLR_BIT(REG, bit)                                 ((REG) &= ~(1u<<(bit)))
29 #define TOG_BIT(REG, bit)                                 ((REG) ^=  (1u<<(bit)))
30
31 #define SET_BYTE(REG, bit)                                ((REG) |=  (0xFFu<<(bit)))
32 #define CLR_BYTE(REG, bit)                                ((REG) &= ~(0xFFu<<(bit)))
33 #define TOG_BYTE(REG, bit)                                ((REG) ^=  (0xFFu<<(bit)))
34
35 #define SET_MASK(REG, MASK)                               ((REG) |=  (MASK))
36 #define CLR_MASK(REG, MASK)                               ((REG) &= ~(MASK))
37 #define TOG_MASK(REG, MASK)                               ((REG) ^=  (MASK))
38 #define GET_MASK(REG, MASK)                               ((REG) &   (MASK))
39
40 #define SET_REG(REG)                                      ((REG)  = ~(0u))
41 #define CLR_REG(REG)                                      ((REG)  =  (0u))
42 #define TOG_REG(REG)                                      ((REG) ^= ~(0u))
43
44 #define GET_BIT(REG, bit)                                 (((REG)>>(bit)) & 0x01u)
45 #define GET_NIB(REG, bit)                                 (((REG)>>(bit)) & 0x0Fu)
46 #define GET_BYTE(REG, bit)                                (((REG)>>(bit)) & 0xFFu)
47
48 #define ASSIGN_BIT(REG, bit, value)                      ((REG) = ((REG) & ~(0x01u<<(bit)))
| (((value) & 0x01u)<<(bit)))
49 #define ASSIGN_NIB(REG, bit, value)                      ((REG) = ((REG) & ~(0x0Fu<<(bit)))
| (((value) & 0x0Fu)<<(bit)))
50 #define ASSIGN_BYTE(REG, bit, value)                     ((REG) = ((REG) & ~(0xFFu<<(bit)))
| (((value) & 0xFFu)<<(bit)))
51
52 /*
53 #define ASSIGN_BIT(REG,bit,value)                         do{
\
54                                                            REG &= ~(0x01u<<bit);
\
55                                                            REG |=  ((value & 0x01u)<<bit);
\
56                                                           }while(0)
57 */
58
59 /*          bits together in an u8 register          */
60 #define CON_u8Bits(b7, b6, b5, b4, b3, b2, b1, b0)
\
61
(0b##b7##b6##b5##b4##b3##b2##b1##b0)
62
63 /*          bits together in an u16 register         */
64 #define CON_u16Bits(b15, b14, b13, b12, b11, b10, b9, b8, b7, b6, b5, b4, b3, b2, b1,
b0)   \
```

```
65
(0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2##b1##b0)
66
67 /* ****************************************************************************** */
68 /* ***************************** CONST SECTION ********************************** */
69 /* ****************************************************************************** */
70
71 /* ****************************************************************************** */
72 /* ***************************** VARIABLE SECTION ******************************* */
73 /* ****************************************************************************** */
74
75 /* ****************************************************************************** */
76 /* ***************************** FUNCTION SECTION ******************************* */
77 /* ****************************************************************************** */
78
79
80 #endif /* LBIT_INT_H_ */
81 /***************************** E N D (LBIT_int.h) ******************************/
```

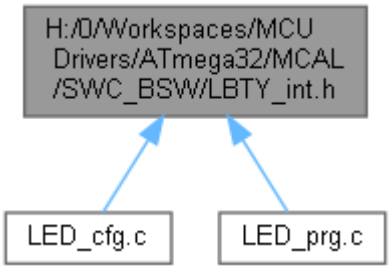
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

```
#include <stdint.h>
```

Include dependency graph for LBTY_int.h:



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



Data Structures

- union LBTY_tuniPort8union LBTY_tuniPort16

Macros

- #define __IO volatile
- #define __O volatile
- #define __I volatile const
- #define LBTY_u8vidNOP()
- #define LBTY_NULL ((void *) 0U)
- #define LBTY_u8ZERO ((u8)0x00U)
- #define LBTY_u8MAX ((u8)0xFFU)
- #define LBTY_s8MAX ((s8)0x7F)
- #define LBTY_s8MIN ((s8)0x80)
- #define LBTY_u16ZERO ((u16)0x0000U)
- #define LBTY_u16MAX ((u16)0xFFFFU)
- #define LBTY_s16MAX ((u16)0x7FFF)
- #define LBTY_s16MIN ((u16)0x8000)
- #define LBTY_u32ZERO ((u32)0x00000000UL)
- #define LBTY_u32MAX ((u32)0xFFFFFFFFUL)
- #define LBTY_s32MAX ((u32)0x7FFFFFFFL)
- #define LBTY_s32MIN ((u32)0x80000000L)
- #define LBTY_u64ZERO ((u64)0x0000000000000000ULL)
- #define LBTY_u64MAX ((u64)0xFFFFFFFFFFFFFFFFULL)
- #define LBTY_s64MAX ((u64)0x7FFFFFFFFFFFFFFFLL)
- #define LBTY_s64MIN ((u64)0x8000000000000000LL)

Typedefs

- typedef uint8_t u8
- typedef uint16_t u16
- typedef uint32_t u32
- typedef uint64_t u64
- typedef int8_t s8
- typedef int16_t s16
- typedef int32_t s32
- typedef int64_t s64
- typedef float f32
- typedef double f64
- typedef u8 * pu8
- typedef u16 * pu16
- typedef u32 * pu32
- typedef u64 * pu64
- typedef s8 * ps8
- typedef s16 * ps16
- typedef s32 * ps32
- typedef s64 * ps64

Enumerations

- enum LBTY_tenuFlagStatus { LBTY_RESET = 0, LBTY_SET = !LBTY_RESET }
- enum LBTY_tenuBoolean { LBTY_TRUE = 0x55, LBTY_FALSE = 0xAA }
- enum LBTY_tenuErrorStatus { LBTY_OK = (u16)0, LBTY_NOK, LBTY_NULL_POINTER, LBTY_INDEX_OUT_OF_RANGE, LBTY_NO_MASTER_CHANNEL, LBTY_READ_ERROR, LBTY_WRITE_ERROR, LBTY_UNDEFINED_ERROR, LBTY_IN_PROGRESS }

Macro Definition Documentation

#define __I volatile const

#define __IO volatile

#define __O volatile

#define LBTY_NULL ((void *) 0U)

#define LBTY_s16MAX ((u16)0x7FFF)

#define LBTY_s16MIN ((u16)0x8000)

#define LBTY_s32MAX ((u32)0x7FFFFFFFL)

#define LBTY_s32MIN ((u32)0x80000000L)

#define LBTY_s64MAX ((u64)0x7FFFFFFFFFFFFFFFLL)

#define LBTY_s64MIN ((u64)0x8000000000000000LL)

#define LBTY_s8MAX ((s8)0x7F)

#define LBTY_s8MIN ((s8)0x80)

#define LBTY_u16MAX ((u16)0xFFFFU)

#define LBTY_u16ZERO ((u16)0x0000U)

#define LBTY_u32MAX ((u32)0xFFFFFFFFUL)

#define LBTY_u32ZERO ((u32)0x00000000UL)

#define LBTY_u64MAX ((u64)0xFFFFFFFFFFFFFFFFULL)

#define LBTY_u64ZERO ((u64)0x0000000000000000ULL)

#define LBTY_u8MAX ((u8)0xFFU)

#define LBTY_u8vidNOP()

#define LBTY_u8ZERO ((u8)0x00U)

Data Types Limitation

Typedef Documentation

typedef float f32

Standard Real Decimal number

typedef double f64

typedef s16* ps16

typedef s32* ps32

typedef s64* ps64

typedef s8* ps8

Standard Pointer to Signed Byte/Word/Long_Word

typedef u16* pu16

typedef u32* pu32

typedef u64* pu64

typedef u8* pu8

Standard Pointer to Unsigned Byte/Word/Long_Word

typedef int16_t s16

typedef int32_t s32

typedef int64_t s64

typedef int8_t s8

Standard Signed Byte/Word/Long_Word

typedef uint16_t u16

typedef uint32_t u32

typedef uint64_t u64

typedef uint8_t u8

Data Types New Definitions Standard Unsigned Byte/Word/Long_Word

---

Enumeration Type Documentation

enum LBTY_tenuBoolean

Boolean type

Enumerator:

LBTY_TRUE	
LBTY_FALSE	

```
96              {
97   LBTY_TRUE = 0x55,
98   LBTY_FALSE = 0xAA
99 } LBTY_tenuBoolean;
```

enum LBTY_tenuErrorStatus

Error Return type

Enumerator:

LBTY_OK	
LBTY_NOK	
LBTY_NULL_POINTER	
LBTY_INDEX_OUT_OF_RANGE	
LBTY_NO_MASTER_CHANNEL	
LBTY_READ_ERROR	
LBTY_WRITE_ERROR	
LBTY_UNDEFINED_ERROR	
LBTY_IN_PROGRESS	

```
102             {
103   LBTY_OK = (u16)0,
104   LBTY_NOK,
105   LBTY_NULL_POINTER,
106   LBTY_INDEX_OUT_OF_RANGE,
107   LBTY_NO_MASTER_CHANNEL,
108   LBTY_READ_ERROR,
109   LBTY_WRITE_ERROR,
110   LBTY_UNDEFINED_ERROR,
111   LBTY_IN_PROGRESS        /* Error is not available, wait for availability */
112 } LBTY_tenuErrorStatus;
```

enum LBTY_tenuFlagStatus

Flag Status type

Enumerator:

LBTY_RESET	
LBTY_SET	

```
90              {
91    LBTY_RESET = 0,
92    LBTY_SET = !LBTY_RESET
93 } LBTY_tenuFlagStatus;
```

LBTY_int.h

Go to the documentation of this file.

```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************* */
4 /* File Name   : LBTY_int.h                                                  */
5 /* Author      : MAAM                                                        */
6 /* Version     : v01                                                         */
7 /* date        : Mar 23, 2023                                                */
8 /* description : Basic Library                                               */
9 /* ************************************************************************* */
10 /* ************************ HEADER FILES INCLUDES ************************  */
11 /* ************************************************************************* */
12
13 #ifndef _LBTY_INT_H_
14 #define _LBTY_INT_H_
15
16 #include <stdint.h>
17
18 /* ************************************************************************* */
19 /* ****************************** TYPE_DEF SECTION ************************* */
20 /* ************************************************************************* */
21
24 typedef uint8_t     u8 ;
25 typedef uint16_t    u16;
26 typedef uint32_t    u32;
27 typedef uint64_t    u64;
28
30 typedef int8_t      s8 ;
31 typedef int16_t     s16;
32 typedef int32_t     s32;
33 typedef int64_t     s64;
34
36 typedef float       f32;
37 typedef double      f64;
38
40 typedef u8*         pu8 ;
41 typedef u16*        pu16;
42 typedef u32*        pu32;
43 typedef u64*        pu64;
44
46 typedef s8*         ps8 ;
47 typedef s16*        ps16;
48 typedef s32*        ps32;
49 typedef s64*        ps64;
50
51 /* ************************************************************************* */
52 /* ************************** MACRO/DEFINE SECTION ************************* */
53 /* ************************************************************************* */
54
55 /*****************************************************************************/
56 #define __IO    volatile
57 #define __O     volatile
58 #define __I     volatile const
59 /*****************************************************************************/
60
61 #define LBTY_u8vidNOP()
62 #define LBTY_NULL       ((void *) 0U)
63
65 #define LBTY_u8ZERO     ((u8)0x00U)
66 #define LBTY_u8MAX      ((u8)0xFFU)
67 #define LBTY_s8MAX      ((s8)0x7F )
68 #define LBTY_s8MIN      ((s8)0x80 )
69
70 #define LBTY_u16ZERO    ((u16)0x0000U)
71 #define LBTY_u16MAX     ((u16)0xFFFFU)
72 #define LBTY_s16MAX     ((u16)0x7FFF )
73 #define LBTY_s16MIN     ((u16)0x8000 )
74
75 #define LBTY_u32ZERO    ((u32)0x00000000UL)
76 #define LBTY_u32MAX     ((u32)0xFFFFFFFFUL)
77 #define LBTY_s32MAX     ((u32)0x7FFFFFFFL )
78 #define LBTY_s32MIN     ((u32)0x80000000L )
79
```

```
80 #define LBTY_u64ZERO     ((u64)0x0000000000000000ULL)
81 #define LBTY_u64MAX      ((u64)0xFFFFFFFFFFFFFFFFULL)
82 #define LBTY_s64MAX      ((u64)0x7FFFFFFFFFFFFFFFLL )
83 #define LBTY_s64MIN      ((u64)0x8000000000000000LL )
84
85 /* ********************************************************************************* */
86 /* ****************************** ENUM SECTION ************************************** */
87 /* ********************************************************************************* */
88
90 typedef enum {
91     LBTY_RESET = 0,
92     LBTY_SET = !LBTY_RESET
93 } LBTY_tenuFlagStatus;
94
96 typedef enum {
97   LBTY_TRUE = 0x55,
98   LBTY_FALSE = 0xAA
99 } LBTY_tenuBoolean;
100
102 typedef enum {
103   LBTY_OK = (u16)0,
104   LBTY_NOK,
105   LBTY_NULL_POINTER,
106   LBTY_INDEX_OUT_OF_RANGE,
107   LBTY_NO_MASTER_CHANNEL,
108   LBTY_READ_ERROR,
109   LBTY_WRITE_ERROR,
110   LBTY_UNDEFINED_ERROR,
111   LBTY_IN_PROGRESS         /* Error is not available, wait for availability */
112 } LBTY_tenuErrorStatus;
113
114 /* ********************************************************************************* */
115 /* ****************************** STRUCT SECTION ************************************ */
116 /* ********************************************************************************* */
117
119 typedef union {
120   struct {
121     u8 m_u8b0 :1;       // LSB
122     u8 m_u8b1 :1;
123     u8 m_u8b2 :1;
124     u8 m_u8b3 :1;
125     u8 m_u8b4 :1;
126     u8 m_u8b5 :1;
127     u8 m_u8b6 :1;
128     u8 m_u8b7 :1;       // MSB
129   } sBits;
130   u8 u_u8Byte;
131 } LBTY_tuniPort8;
132
133 typedef union {
134   struct  {
135     u8 m_u8b0  :1;      // LSB
136     u8 m_u8b1  :1;
137     u8 m_u8b2  :1;
138     u8 m_u8b3  :1;
139     u8 m_u8b4  :1;
140     u8 m_u8b5  :1;
141     u8 m_u8b6  :1;
142     u8 m_u8b7  :1;
143     u8 m_u8b8  :1;
144     u8 m_u8b9  :1;
145     u8 m_u8b10 :1;
146     u8 m_u8b11 :1;
147     u8 m_u8b12 :1;
148     u8 m_u8b13 :1;
149     u8 m_u8b14 :1;
150     u8 m_u8b15 :1;      // MSB
151   } sBits;
152   struct {
153     u8 m_u8low;
154     u8 m_u8high;
155   } sBytes;
156   u16 u_u16Word;
157 } LBTY_tuniPort16;
158
159 /* ********************************************************************************* */
160 /* ***************************** FUNCTION SECTION *********************************** */
```

```
161 /* ************************************************************************** */
162
163
164 #endif /* _LBTY_INT_H_ */
165 /*************************** E N D (LBTY_int.h) ****************************/
```

H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LCTY_int.h File Reference

Macros

- #define LCTY_PROGMEM __attribute__((__progmem__))
- #define LCTY_PURE __attribute__((__pure__))
- #define LCTY_INLINE __attribute__((always_inline)) static inline
- #define LCTY_INTERRUPT __attribute__((interrupt))
- #define CTY_PACKED __attribute__((__packed__))
- #define LCTY_CONST __attribute__((__const__))
- #define LCTY_DPAGE __attribute__((dp))
- #define LCTY_NODPAGE __attribute__((nodp))
- #define LCTY_SECTION(section) __attribute__((section(# section)))
- #define LCTY_ASM(cmd) __asm__ __volatile__ (# cmd ::)

Macro Definition Documentation

#define CTY_PACKED __attribute__((__packed__))

#define LCTY_ASM(cmd) __asm__ __volatile__ (# cmd ::)

#define LCTY_CONST __attribute__((__const__))

#define LCTY_DPAGE __attribute__((dp))

#define LCTY_INLINE __attribute__((always_inline)) static inline

#define LCTY_INTERRUPT __attribute__((interrupt))

#define LCTY_NODPAGE __attribute__((nodp))

#define LCTY_PROGMEM __attribute__((__progmem__))

#define LCTY_PURE __attribute__((__pure__))

#define LCTY_SECTION(section) __attribute__((section(# section)))

LCTY_int.h

Go to the documentation of this file.

```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************* */
4 /* File Name   : LCTY_int.h                                                  */
5 /* Author      : MAAM                                                        */
6 /* Version     : v00                                                         */
7 /* date        : Apr 26, 2023                                                */
8 /* description : Compiler Library                                            */
9 /* ************************************************************************* */
10 /* ************************ HEADER FILES INCLUDES ************************  */
11 /* ************************************************************************* */
12
13 #ifndef LCTY_INT_H_
14 #define LCTY_INT_H_
15
16 /* ************************************************************************* */
17 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
18 /* ************************************************************************* */
19
20 /* ************************************************************************* */
21 /* ************************** MACRO/DEFINE SECTION ************************** */
22 /* ************************************************************************* */
23
24 /* prog memory attribute */
25 #define LCTY_PROGMEM                __attribute__((__progmem__))
26
27 /* pure attribute */
28 #define LCTY_PURE                   __attribute__((__pure__))
29
30 /* Abstraction for inlining */
31 //#define LCTY_INLINE               static inline
32 #define LCTY_INLINE                 __attribute__((always_inline)) static inline
33
34 /* define function as interrupt handler */
35 #define LCTY_INTERRUPT              __attribute__((interrupt))
36
37 /* Memory packed to pass Memory padding */
38 #define CTY_PACKED                  __attribute__((__packed__))
39
40 /* Const attribute */
41 #define LCTY_CONST                  __attribute__((__const__))
42
43 /* place variable in direct page */
44 #define LCTY_DPAGE                  __attribute__((dp))
45
46 /* do not place variable in direct page */
47 #define LCTY_NODPAGE                __attribute__((nodp))
48
49 /* Sections */
50 #define LCTY_SECTION(section)       __attribute__((section( # section)))
51
52 /* Abstraction for assembly command */
53 # define LCTY_ASM(cmd)              __asm__ __volatile__ ( # cmd ::)
54
55 /* ************************************************************************* */
56 /* ***************************** CONST SECTION ***************************** */
57 /* ************************************************************************* */
58
59 /* ************************************************************************* */
60 /* ***************************** VARIABLE SECTION ************************** */
61 /* ************************************************************************* */
62
63 /* ************************************************************************* */
64 /* ***************************** FUNCTION SECTION ************************** */
65 /* ************************************************************************* */
66
67
68 #endif /* LCTY_INT_H_ */
69 /*************************** E N D (LCTY_int.h) ****************************/
```