SWC_PUSH

Version v1.0
7/16/2023 3:38:00 AM

Table of Contents

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

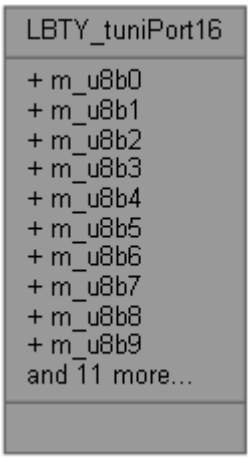
File List

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

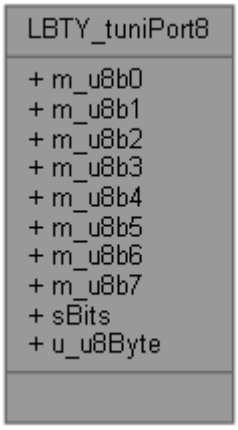
The documentation for this union was generated from the following file:

- 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

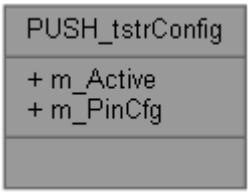
The documentation for this union was generated from the following file:

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

PUSH_tstrConfig Struct Reference

: type define of structure for GPIO Push Configuration

`#include <PUSH_int.h>`
Collaboration diagram for PUSH_tstrConfig:



Data Fields

- PUSH_tenuActive m_Active
- GPIO_tstrPinConfig m_PinCfg

Detailed Description

: type define of structure for GPIO Push Configuration

Type : struct **Unit** : None

Field Documentation

PUSH_tenuActive m_Active

Push Active

GPIO_tstrPinConfig m_PinCfg

Pin Configuration

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

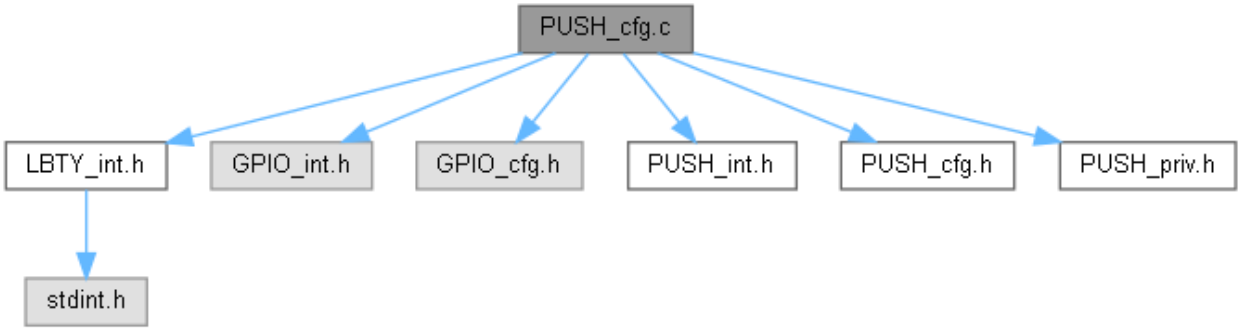
File Documentation

main.c File Reference

PUSH_cfg.c File Reference

```
#include "LBTY_int.h"
#include "GPIO_int.h"
#include "GPIO_cfg.h"
#include "PUSH_int.h"
#include "PUSH_cfg.h"
#include "PUSH_priv.h"
```

Include dependency graph for PUSH_cfg.c:



Variables

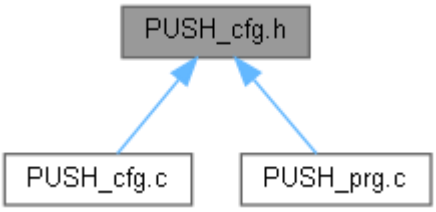
- const PUSH_tstrConfig kau8PushConfiguration_LGB [PUSH_Num]

Variable Documentation

const PUSH_tstrConfig kau8PushConfiguration_LGB[PUSH_Num]

PUSH_cfg.h File Reference

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



Macros

- #define PUSH0 0
- #define PUSH0_PORT D
- #define PUSH0_PIN GPIO_INT0
- #define PUSH0_ACTIVE PUSH_Active_High
- #define PUSH1 1
- #define PUSH1_PORT D
- #define PUSH1_PIN GPIO_INT1
- #define PUSH1_ACTIVE PUSH_Active_High
- #define PUSH2 2
- #define PUSH2_PORT B
- #define PUSH2_PIN GPIO_INT2
- #define PUSH2_ACTIVE PUSH_Active_High
- #define PUSH3 3
- #define PUSH3_PORT B
- #define PUSH3_PIN AMIT_B0
- #define PUSH3_ACTIVE PUSH_Active_High
- #define PUSH4 4
- #define PUSH4_PORT B
- #define PUSH4_PIN AMIT_B4
- #define PUSH4_ACTIVE PUSH_Active_High
- #define PUSH5 5
- #define PUSH5_PORT B
- #define PUSH5_PIN AMIT_B5
- #define PUSH5_ACTIVE PUSH_Active_High
- #define PUSH6 6
- #define PUSH6_PORT B
- #define PUSH6_PIN AMIT_B6
- #define PUSH6_ACTIVE PUSH_Active_High
- #define PUSH7 7
- #define PUSH7_PORT B
- #define PUSH7_PIN AMIT_B7
- #define PUSH7_ACTIVE PUSH_Active_High
- #define DEBOUNCING_CYCLES_NUM 5
- #define DEBOUNCING_DELAY 5

Macro Definition Documentation

#define DEBOUNCING_CYCLES_NUM 5

#define DEBOUNCING_DELAY 5

#define PUSH0 0

#define PUSH0_ACTIVE PUSH_Active_High

#define PUSH0_PIN GPIO_INT0

#define PUSH0_PORT D

#define PUSH1 1

#define PUSH1_ACTIVE PUSH_Active_High

#define PUSH1_PIN GPIO_INT1

#define PUSH1_PORT D

#define PUSH2 2

#define PUSH2_ACTIVE PUSH_Active_High

#define PUSH2_PIN GPIO_INT2

#define PUSH2_PORT B

#define PUSH3 3

#define PUSH3_ACTIVE PUSH_Active_High

#define PUSH3_PIN AMIT_B0

#define PUSH3_PORT B

#define PUSH4 4

#define PUSH4_ACTIVE PUSH_Active_High

#define PUSH4_PIN AMIT_B4

#define PUSH4_PORT B

#define PUSH5 5

#define PUSH5_ACTIVE PUSH_Active_High

#define PUSH5_PIN AMIT_B5

#define PUSH5_PORT B

#define PUSH6 6

#define PUSH6_ACTIVE PUSH_Active_High

#define PUSH6_PIN AMIT_B6

#define PUSH6_PORT B

#define PUSH7 7

#define PUSH7_ACTIVE PUSH_Active_High

#define PUSH7_PIN AMIT_B7

#define PUSH7_PORT B

PUSH_cfg.h

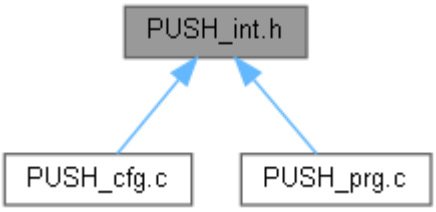
Go to the documentation of this file.

```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************** */
4 /* File Name    : PUSH_cfg.h                                                  */
5 /* Author       : MAAM                                                        */
6 /* Version      : v01.2                                                       */
7 /* date         : Mar 30, 2023                                                */
8 /* ************************************************************************** */
9 /* ************************ HEADER FILES INCLUDES ************************  */
10 /* ************************************************************************** */
11
12 #ifndef PUSH_CFG_H_
13 #define PUSH_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 PUSH0           0
26 #define PUSH0_PORT      D
27 #define PUSH0_PIN       AMIT_PUSH0
28 #define PUSH0_ACTIVE    PUSH_Active_High
29
30 #define PUSH1           1
31 #define PUSH1_PORT      D
32 #define PUSH1_PIN       AMIT_PUSH1
33 #define PUSH1_ACTIVE    PUSH_Active_High
34
35 #define PUSH2           2
36 #define PUSH2_PORT      D
37 #define PUSH2 PIN       AMIT PUSH2
38 #define PUSH2_ACTIVE    PUSH_Active_High
39
40 #elif defined(ETA32_KIT)
41
42 #define PUSH0           0
43 #define PUSH0 PORT      D
44 #define PUSH0_PIN       Eta32_Keypad_col0
45 #define PUSH0_ACTIVE    PUSH_Active_Low
46
47 #define PUSH1           1
48 #define PUSH1_PORT      D
49 #define PUSH1 PIN       Eta32 Keypad col1
50 #define PUSH1_ACTIVE    PUSH_Active_Low
51
52 #define PUSH2           2
53 #define PUSH2_PORT      D
54 #define PUSH2 PIN       Eta32 Keypad col2
55 #define PUSH2_ACTIVE    PUSH_Active_Low
56
57 #define PUSH3           3
58 #define PUSH3_PORT      D
59 #define PUSH3_PIN       Eta32_Keypad_col3
60 #define PUSH3 ACTIVE    PUSH Active Low
61
62 #elif defined(ETA32_MINI_KIT)
63
64 #define PUSH0           0
65 #define PUSH0_PORT      D
66 #define PUSH0 PIN       Eta32 mini Keypad col0
67 #define PUSH0_ACTIVE    PUSH_Active_Low
68
69 #define PUSH1           1
70 #define PUSH1_PORT      D
71 #define PUSH1_PIN       Eta32_mini_Keypad_col1
72 #define PUSH1_ACTIVE    PUSH_Active_Low
```

```
73
74 #define PUSH2           2
75 #define PUSH2_PORT      D
76 #define PUSH2_PIN       Eta32_mini_Keypad_col2
77 #define PUSH2_ACTIVE    PUSH_Active_Low
78
79 #define PUSH3           3
80 #define PUSH3_PORT      D
81 #define PUSH3_PIN       Eta32_mini_Keypad_col3
82 #define PUSH3_ACTIVE    PUSH_Active_Low
83
84 #else
85
86 #define PUSH0           0
87 #define PUSH0_PORT      D
88 #define PUSH0_PIN       GPIO_INT0
89 #define PUSH0_ACTIVE    PUSH_Active_High
90
91 #define PUSH1           1
92 #define PUSH1_PORT      D
93 #define PUSH1_PIN       GPIO_INT1
94 #define PUSH1_ACTIVE    PUSH_Active_High
95
96 #define PUSH2           2
97 #define PUSH2_PORT      B
98 #define PUSH2_PIN       GPIO_INT2
99 #define PUSH2_ACTIVE    PUSH_Active_High
100
101 #define PUSH3            3
102 #define PUSH3_PORT       B
103 #define PUSH3_PIN        AMIT_B0
104 #define PUSH3_ACTIVE     PUSH_Active_High
105
106 #define PUSH4            4
107 #define PUSH4_PORT       B
108 #define PUSH4_PIN        AMIT_B4
109 #define PUSH4_ACTIVE     PUSH_Active_High
110
111 #define PUSH5            5
112 #define PUSH5_PORT       B
113 #define PUSH5_PIN        AMIT_B5
114 #define PUSH5_ACTIVE     PUSH_Active_High
115
116 #define PUSH6            6
117 #define PUSH6_PORT       B
118 #define PUSH6_PIN        AMIT_B6
119 #define PUSH6_ACTIVE     PUSH_Active_High
120
121 #define PUSH7            7
122 #define PUSH7_PORT       B
123 #define PUSH7_PIN        AMIT_B7
124 #define PUSH7_ACTIVE     PUSH_Active_High
125
126 #endif
127
128 #define DEBOUNCING_CYCLES_NUM   5
129 #define DEBOUNCING_DELAY        5
130
131 /* ********************************************************************************* */
132 /* ***************************** CONST SECTION ***************************** */
133 /* ********************************************************************************* */
134
135 /* ********************************************************************************* */
136 /* ***************************** VARIABLE SECTION ************************** */
137 /* ********************************************************************************* */
138
139 /* ********************************************************************************* */
140 /* ***************************** FUNCTION SECTION ************************** */
141 /* ********************************************************************************* */
142
143
144 #endif /* PUSH CFG H  */
145 /*************************** E N D (PUSH_cfg.h) ******************************/
```

PUSH_int.h File Reference

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



Data Structures

struct PUSH_tstrConfig*: type define of structure for GPIO Push Configuration*

Enumerations

- enum PUSH_tenuActive { PUSH_Active_Low = (u8)0u, PUSH_Active_High }
- enum PUSH_tenuStatus { PUSH_RELEASED = (u8)0u, PUSH_PRESSED }

Functions

- void PUSH_vidInit (u8 u8PushNum)
- void PUSH_vidInitAll (void)
- LBTY_tenuErrorStatus PUSH_u8GetDebounce (u8 u8PushNum, u8 *pu8State)
- LBTY_tenuErrorStatus PUSH_u8GetPushState (u8 u8PushNum, u8 *pu8State)

Enumeration Type Documentation

enum PUSH_tenuActive

Enumerator:

PUSH_Active_Low	
PUSH_Active_High	

```
19              {
20      PUSH_Active_Low = (u8)0u,
21      PUSH_Active_High
22  }PUSH_tenuActive;
```

enum PUSH_tenuStatus

Enumerator:

PUSH_RELEASED	
PUSH_PRESSED	

```
24              {
25      PUSH_RELEASED = (u8)0u,
26      PUSH_PRESSED
27  }PUSH_tenuStatus;
```

Function Documentation

LBTY_tenuErrorStatus PUSH_u8GetDebounce (u8 *u8PushNum*, u8 * *pu8State*)

```
72                                                                          {
73      u8 u8PreValue = LBTY_u8ZERO;
74      u8 u8CurValue = LBTY_u8ZERO;
75      u8 u8DebouncingCount = LBTY_u8ZERO;
76
77      PUSH_tstrConfig* pstrPush =
78              (PUSH_tstrConfig*)&kau8PushConfiguration_LGB[u8PushNum];
79      LBTY_tenuErrorStatus u8RetValue  =
80              GPIO_u8GetPinValue(pstrPush->m_PinCfg.m_Port,
pstrPush->m_PinCfg.m_Pin, &u8PreValue);
81
82      while((u8DebouncingCount < DEBOUNCING_CYCLES_NUM) && (u8RetValue == LBTY_OK)){
83          vidMyDelay_ms(DEBOUNCING_DELAY);
84          u8RetValue  = GPIO_u8GetPinValue(pstrPush->m_PinCfg.m_Port,
pstrPush->m_PinCfg.m_Pin, &u8CurValue);
85
86          if(u8PreValue == u8CurValue){
87              u8DebouncingCount++;
88          }else{
89              u8DebouncingCount = 0;
90          }
91          u8PreValue = u8CurValue;
92      }
93
94      *pu8State = u8CurValue;
95      return u8RetValue;
96  }
```

Here is the caller graph for this function:

LBTY_tenuErrorStatus PUSH_u8GetPushState (u8 *u8PushNum*, u8 * *pu8State*)

```
104                                                                                 {
105      LBTY_tenuErrorStatus u8RetValue  = PUSH_u8GetDebounce(u8PushNum, pu8State);
106
107      *pu8State = (*pu8State == kau8PushConfiguration_LGB[u8PushNum].m_Active) ?
PUSH_PRESSED : PUSH_RELEASED;
108      return u8RetValue;
109  }
```

Here is the call graph for this function:

void PUSH_vidInit (u8 *u8PushNum*)

```
49                                  {
50      //  PUSH_tstrConfig* pstrPush =
51      //          (PUSH_tstrConfig*)&kau8PushConfiguration_LGB[u8PushNum];
52      //  GPIO_u8SetPinDirection(pstrPush->m_PinCfg.m_Port,
pstrPush->m_PinCfg.m_Pin, PIN_INPUT);
53      GPIO_u8PinInit(kau8PushConfiguration_LGB[u8PushNum].m_PinCfg);
54  }
```

void PUSH_vidInitAll (void)

```
61                                  {
62      for(u8 i = PUSH_Num ; i-- ; ){
63          GPIO_u8PinInit(kau8PushConfiguration_LGB[i].m_PinCfg);
64      }
65  }
```

PUSH_int.h

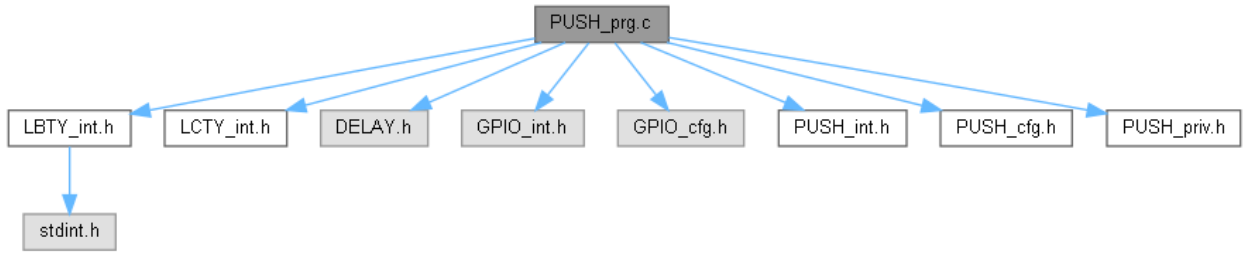
```
Go to the documentation of this file.1 /*
****************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ****************************************************************************** */
4 /* File Name    : PUSH_int.h                                                  */
5 /* Author       : MAAM                                                        */
6 /* Version      : v01.2                                                       */
7 /* date         : Mar 30, 2023                                                */
8 /* ****************************************************************************** */
9 /* ************************ HEADER FILES INCLUDES ************************  */
10 /* ****************************************************************************** */
11
12 #ifndef PUSH_INT_H_
13 #define PUSH_INT_H_
14
15 /* ****************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ****************************************************************************** */
18
19 typedef enum{
20     PUSH_Active_Low = (u8)0u,
21     PUSH_Active_High
22 }PUSH_tenuActive;
23
24 typedef enum{
25     PUSH_RELEASED = (u8)0u,
26     PUSH_PRESSED
27 }PUSH_tenuStatus;
28
29
/*******************************************************************************************
*******************************/
30
33 typedef struct{
34     PUSH_tenuActive     m_Active;
35     GPIO_tstrPinConfig  m_PinCfg;
36 }PUSH_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 Push Button direction                           */
56 /* Input       :    u8PushNum                                                   */
57 /* Return      :    void                                                        */
58 /* ****************************************************************************** */
59 extern void PUSH_vidInit(u8 u8PushNum);
60
61 /* ****************************************************************************** */
62 /* Description : Initialize the All Push Button with Configurations             */
63 /* Input       :    void                                                        */
64 /* Return      :    void                                                        */
65 /* ****************************************************************************** */
66 extern void PUSH_vidInitAll(void);
67
68 /* ****************************************************************************** */
69 /* Description :    Get the Push Button DeBouncing                              */
70 /* Input       :    u8PushNum                                                   */
71 /* Input/Output:    pu8State                                                    */
72 /* Return      :    LBTY_tenuErrorStatus                                        */
```

```
/* ********************************************************************************* */
extern LBTY_tenuErrorStatus PUSH_u8GetDebounce(u8 u8PushNum, u8* pu8State);

/* ********************************************************************************* */
/* Description :    Get the Push Button value                                        */
/* Input       :    u8PushNum                                                        */
/* Input/Output:    pu8State                                                         */
/* Return      :    LBTY_tenuErrorStatus                                             */
/* ********************************************************************************* */
extern LBTY_tenuErrorStatus PUSH_u8GetPushState(u8 u8PushNum, u8* pu8State);

#endif /* PUSH_INT_H_ */
/*************************** E N D (PUSH_int.h) ****************************/
```

PUSH_prg.c File Reference

```
#include "LBTY_int.h"
#include "LCTY_int.h"
#include "DELAY.h"
#include "GPIO_int.h"
#include "GPIO_cfg.h"
#include "PUSH_int.h"
#include "PUSH_cfg.h"
#include "PUSH_priv.h"
```

Include dependency graph for PUSH_prg.c:



Functions

- void PUSH_vidInit (u8 u8PushNum)
- void PUSH_vidInitAll (void)
- LBTY_tenuErrorStatus PUSH_u8GetDebounce (u8 u8PushNum, u8 *pu8State)
- LBTY_tenuErrorStatus PUSH_u8GetPushState (u8 u8PushNum, u8 *pu8State)

Variables

- const PUSH_tstrConfig kau8PushConfiguration_LGB [PUSH_Num]

Function Documentation

LBTY_tenuErrorStatus PUSH_u8GetDebounce (u8 *u8PushNum*, u8 * *pu8State*)

```
72                                                                          {
73     u8 u8PreValue = LBTY_u8ZERO;
74     u8 u8CurValue = LBTY_u8ZERO;
75     u8 u8DebouncingCount = LBTY_u8ZERO;
76
77     PUSH_tstrConfig* pstrPush =
78             (PUSH_tstrConfig*)&kau8PushConfiguration_LGB[u8PushNum];
79     LBTY_tenuErrorStatus u8RetValue  =
80             GPIO_u8GetPinValue(pstrPush->m_PinCfg.m_Port,
pstrPush->m_PinCfg.m_Pin, &u8PreValue);
81
82     while((u8DebouncingCount < DEBOUNCING_CYCLES_NUM) && (u8RetValue == LBTY_OK)){
83         vidMyDelay_ms(DEBOUNCING_DELAY);
84         u8RetValue  = GPIO_u8GetPinValue(pstrPush->m_PinCfg.m_Port,
pstrPush->m_PinCfg.m_Pin, &u8CurValue);
85
86         if(u8PreValue == u8CurValue){
87             u8DebouncingCount++;
88         }else{
89             u8DebouncingCount = 0;
90         }
91         u8PreValue = u8CurValue;
92     }
93
94     *pu8State = u8CurValue;
95     return u8RetValue;
96 }
```

Here is the caller graph for this function:

LBTY_tenuErrorStatus PUSH_u8GetPushState (u8 *u8PushNum*, u8 * *pu8State*)

```
104                                                                          {
105     LBTY_tenuErrorStatus u8RetValue  = PUSH_u8GetDebounce(u8PushNum, pu8State);
106
107     *pu8State = (*pu8State == kau8PushConfiguration_LGB[u8PushNum].m_Active) ?
PUSH_PRESSED : PUSH_RELEASED;
108     return u8RetValue;
109 }
```

Here is the call graph for this function:

void PUSH_vidInit (u8 *u8PushNum*)

```
49                                  {
50     //  PUSH_tstrConfig* pstrPush =
51     //          (PUSH_tstrConfig*)&kau8PushConfiguration_LGB[u8PushNum];
52     //  GPIO_u8SetPinDirection(pstrPush->m_PinCfg.m_Port,
pstrPush->m_PinCfg.m_Pin, PIN_INPUT);
53     GPIO_u8PinInit(kau8PushConfiguration_LGB[u8PushNum].m_PinCfg);
54 }
```

void PUSH_vidInitAll (void)

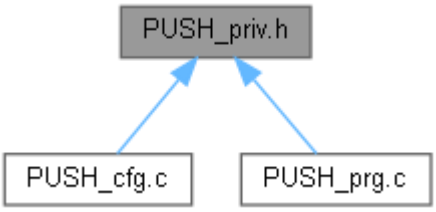
```
61                                  {
62     for(u8 i = PUSH_Num ; i-- ; ){
63         GPIO_u8PinInit(kau8PushConfiguration_LGB[i].m_PinCfg);
64     }
65 }
```

Variable Documentation

const PUSH_tstrConfig kau8PushConfiguration_LGB[PUSH_Num] [extern]

PUSH_priv.h File Reference

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



Enumerations

- enum PUSH_tenuPushNum { PUSH_Num }

Enumeration Type Documentation

enum PUSH_tenuPushNum

Enumerator:

PUSH_Num	

```
19              {
20 #ifdef PUSH0
21     PUSH_0 = (u8)0u
22 #endif
23 #ifdef PUSH1
24     ,PUSH_1
25 #endif
26 #ifdef PUSH2
27     ,PUSH_2
28 #endif
29 #ifdef PUSH3
30     ,PUSH_3
31 #endif
32 #ifdef PUSH4
33     ,PUSH_4
34 #endif
35 #ifdef PUSH5
36     ,PUSH_5
37 #endif
38 #ifdef PUSH6
39     ,PUSH_6
40 #endif
41 #ifdef PUSH7
42     ,PUSH_7
43 #endif
44     ,PUSH_Num
45 }PUSH_tenuPushNum;
```

PUSH_priv.h

Go to the documentation of this file.

```c
/*
 ********************************************************************************** */
/* ********************** FILE DEFINITION SECTION ************************** */
/* ********************************************************************************** */
/* File Name    : PUSH_priv.h                                                       */
/* Author       : MAAM                                                              */
/* Version      : v01.2                                                             */
/* date         : Mar 30, 2023                                                      */
/* ********************************************************************************** */
/* ************************ HEADER FILES INCLUDES ************************  */
/* ********************************************************************************** */

#ifndef PUSH_PRIV_H_
#define PUSH_PRIV_H_

/* ********************************************************************************** */
/* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
/* ********************************************************************************** */

typedef enum{
#ifdef PUSH0
    PUSH_0 = (u8)0u
#endif
#ifdef PUSH1
    ,PUSH_1
#endif
#ifdef PUSH2
    ,PUSH_2
#endif
#ifdef PUSH3
    ,PUSH_3
#endif
#ifdef PUSH4
    ,PUSH_4
#endif
#ifdef PUSH5
    ,PUSH_5
#endif
#ifdef PUSH6
    ,PUSH_6
#endif
#ifdef PUSH7
    ,PUSH_7
#endif
    ,PUSH_Num
}PUSH_tenuPushNum;

/* ********************************************************************************** */
/* ************************** MACRO/DEFINE SECTION ************************** */
/* ********************************************************************************** */

/* ********************************************************************************** */
/* ***************************** CONST SECTION ****************************** */
/* ********************************************************************************** */

/* ********************************************************************************** */
/* ***************************** VARIABLE SECTION *************************** */
/* ********************************************************************************** */

/* ********************************************************************************** */
/* ***************************** FUNCTION SECTION *************************** */
/* ********************************************************************************** */


#endif /* PUSH_PRIV_H_ */
/***************************** E N D (PUSH_priv.h) ******************************/
```

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.

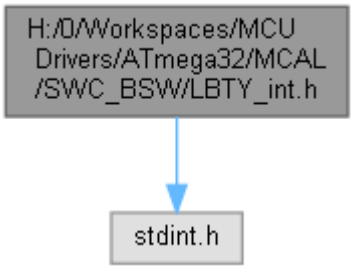
```
1 /*
******************************************************************************* */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ******************************************************************************* */
4 /* File Name   : LBIT_int.h                                                    */
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) ******************************/
```

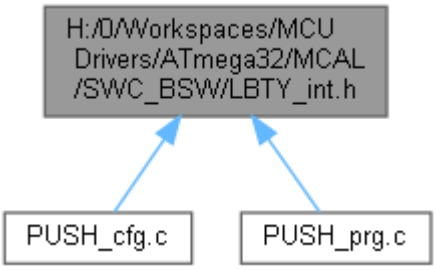
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

```
#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.

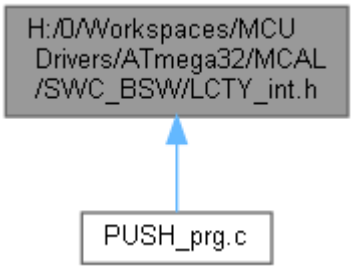
```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************* */
4 /* File Name   : LBTY_int.h                                                  */
5 /* Author      : MAAM                                                        */
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

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



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) ****************************/
```