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

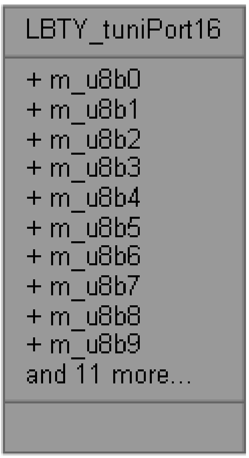
Here is a list of all files with brief descriptions:

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

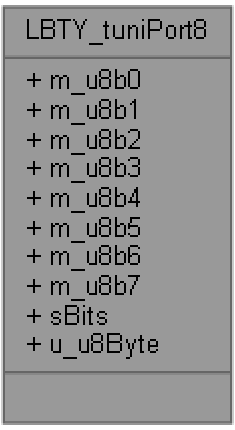
The documentation for this union was generated from the following file:

- 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:

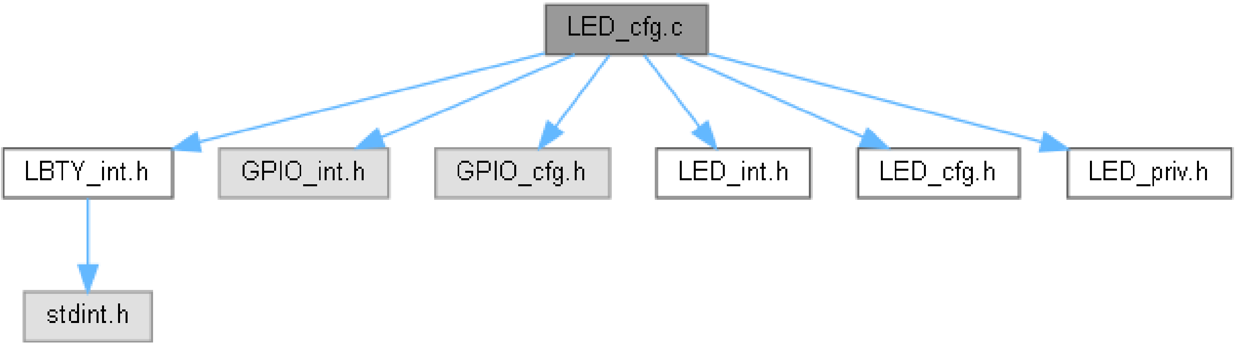
LED_int.h

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]

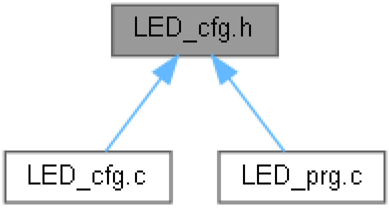
Variable Documentation

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

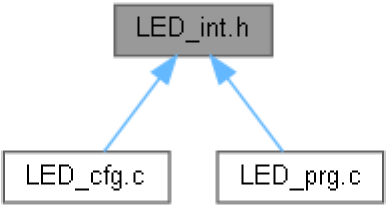
Go to the documentation of this file.

```
/* *************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name    : LED_cfg.h                                                   */
/* Author       : MAAM                                                        */
/* Version      : v01.2                                                       */
/* date         : Apr 8, 2023                                                 */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES ************************** */
/* *************************************************************************** */

#ifndef LED_CFG_H_
#define LED_CFG_H_

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

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

#if defined(AMIT_KIT)

#define LED0            0
#define LED0_PORT       D
#define LED0_PIN        AMIT_LED0
#define LED0_CON        LED_Forward

#define LED1            1
#define LED1_PORT       D
#define LED1_PIN        AMIT_LED1
#define LED1_CON        LED_Forward

#define LED2            2
#define LED2_PORT       D
#define LED2_PIN        AMIT_LED2
#define LED2_CON        LED_Forward

#elif defined(ETA32_KIT)

#define LED0            0
#define LED0_PORT       B
#define LED0_PIN        Eta32_LED_R
#define LED0_CON        LED_Forward

#define LED1            1
#define LED1_PORT       A
#define LED1_PIN        Eta32_LED_G
#define LED1_CON        LED_Forward

#define LED2            2
#define LED2_PORT       A
#define LED2_PIN        Eta32_LED_B
#define LED2_CON        LED_Forward

#define LED3            3
#define LED3_PORT       A
#define LED3_PIN        Eta32_LED_Y
#define LED3_CON        LED_Forward

#elif defined(ETA32_MINI_KIT)

#define LED0            0
#define LED0_PORT       C
#define LED0_PIN        Eta32_mini_LED_R
#define LED0_CON        LED_Forward

#define LED1            1
#define LED1_PORT       C
#define LED1_PIN        Eta32_mini_LED_G
#define LED1_CON        LED_Forward
```

```

#define LED2            2
#define LED2_PORT       C
#define LED2_PIN        Eta32_mini_LED_B
#define LED2_CON        LED_Forward

#else

#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

#endif

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

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

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


#endif /* LED_CFG_H_ */
/************************** 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	

```
             {
    LED_Forward = (u8)0u,
    LED_Reverse
}LED_tenuConnection;
```

enum LED_tenuStatus

Enumerator:

LED_OFF	
LED_ON	

```
             {
    LED_OFF = (u8)0u,
    LED_ON
}LED_tenuStatus;
```

Function Documentation

LBTY_tenuErrorStatus LED_u8SetOFF (u8 *u8LedNum*)

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

LBTY_tenuErrorStatus LED_u8SetON (u8 *u8LedNum*)

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

LBTY_tenuErrorStatus LED_u8Toggle (u8 *u8LedNum*)

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

void LED_vidInit (u8 *u8LedNum*)

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

void LED_vidInitAll (void)

```
                         {
    for(u8 i = LED_Num ; i-- ; ){
        GPIO_u8PinInit(kau8LEDConfiguration_LGB[i].m_PinCfg);
    }
}
```

LED_int.h

Go to the documentation of this file.

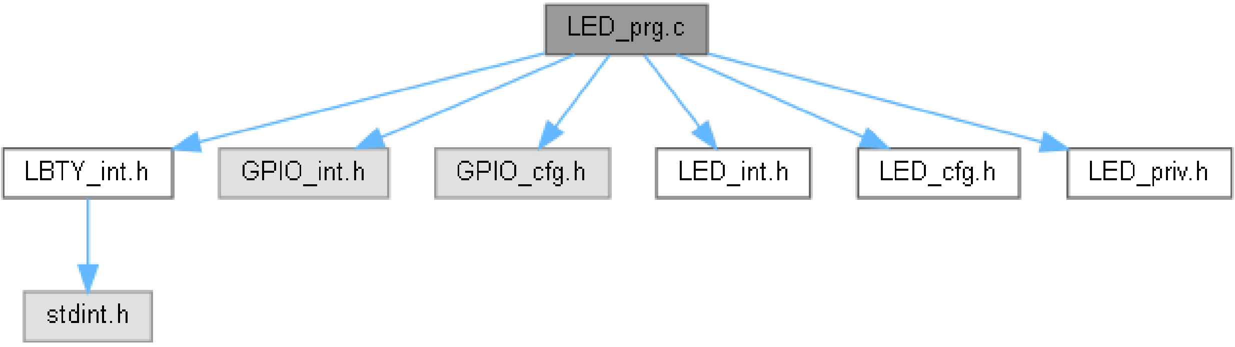
```c
/*
************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name   : LED_int.h                                                  */
/* Author      : MAAM                                                       */
/* Version     : v01.2                                                      */
/* date        : Apr 8, 2023                                                */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES **************************  */
/* ************************************************************************** */

#ifndef LED_INT_H_
#define LED_INT_H_

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

typedef enum{
    LED_Forward = (u8)0u,
    LED_Reverse
}LED_tenuConnection;

typedef enum{
    LED_OFF = (u8)0u,
    LED_ON
}LED_tenuStatus;


/**************************************************************************************
*******************************/

typedef struct{
    LED_tenuConnection  m_Connection;
    GPIO_tstrPinConfig  m_PinCfg;
}LED_tstrConfig;

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

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

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

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

/* ************************************************************************** */
/* Description : Initialize the LED Button direction                         */
/* Input       :    u8LedNum                                                  */
/* Return      :    void                                                      */
/* ************************************************************************** */
extern void LED_vidInit(u8 u8LedNum);

/* ************************************************************************** */
/* Description : Initialize the All LED Button with Configurations           */
/* Input       :    void                                                      */
/* Return      :    void                                                      */
/* ************************************************************************** */
extern void LED_vidInitAll(void);

/* ************************************************************************** */
/* Description :    Set the LED ON                                            */
/* Input       :    u8LedNum                                                  */
/* Return      :    LBTY_tenuErrorStatus                                      */
/* ************************************************************************** */
```

```c
extern LBTY_tenuErrorStatus LED_u8SetON(u8 u8LedNum);

/* ************************************************************************** */
/* Description :    Set the LED OFF                                           */
/* Input       :    u8LedNum                                                  */
/* Return      :    LBTY_tenuErrorStatus                                      */
/* ************************************************************************** */
extern LBTY_tenuErrorStatus LED_u8SetOFF(u8 u8LedNum);

/* ************************************************************************** */
/* Description :    Toggle the LED                                            */
/* Input       :    u8LedNum                                                  */
/* Return      :    LBTY_tenuErrorStatus                                      */
/* ************************************************************************** */
extern LBTY_tenuErrorStatus LED_u8Toggle(u8 u8LedNum);

#endif /* LED_INT_H_ */
/*************************** 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*)

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

LBTY_tenuErrorStatus LED_u8SetON (u8 *u8LedNum*)

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

LBTY_tenuErrorStatus LED_u8Toggle (u8 *u8LedNum*)

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

void LED_vidInit (u8 *u8LedNum*)

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

void LED_vidInitAll (void)

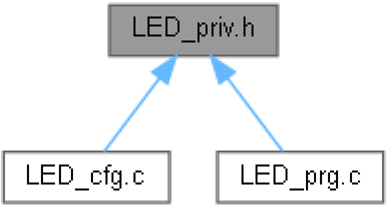
```
                          {
    for(u8 i = LED_Num ; i-- ; ){
        GPIO_u8PinInit(kau8LEDConfiguration_LGB[i].m_PinCfg);
    }
}
```

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	

```
            {
#ifdef LED0
    LED_0 = (u8)0u
#endif
#ifdef LED1
    ,LED_1
#endif
#ifdef LED2
    ,LED_2
#endif
#ifdef LED3
    ,LED_3
#endif
#ifdef LED4
    ,LED_4
#endif
#ifdef LED5
    ,LED_5
#endif
#ifdef LED6
    ,LED_6
#endif
#ifdef LED7
    ,LED_7
#endif
    ,LED_Num
}LED_tenuLedNum;
```

LED_priv.h

Go to the documentation of this file.

```c
/* ************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name    : LED_priv.h                                                  */
/* Author       : MAAM                                                        */
/* Version      : v01.2                                                       */
/* date         : Apr 8, 2023                                                 */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES **************************  */
/* ************************************************************************** */

#ifndef LED_PRIV_H_
#define LED_PRIV_H_

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

typedef enum{
#ifdef LED0
    LED_0 = (u8)0u
#endif
#ifdef LED1
    ,LED_1
#endif
#ifdef LED2
    ,LED_2
#endif
#ifdef LED3
    ,LED_3
#endif
#ifdef LED4
    ,LED_4
#endif
#ifdef LED5
    ,LED_5
#endif
#ifdef LED6
    ,LED_6
#endif
#ifdef LED7
    ,LED_7
#endif
    ,LED_Num
}LED_tenuLedNum;

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

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

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

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

#endif /* LED_PRIV_H_ */
/*************************** 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

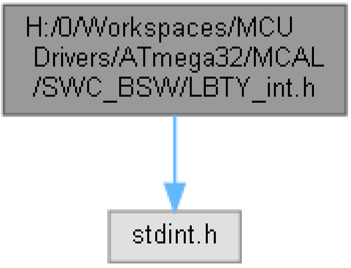
Go to the documentation of this file.

```c
/* *************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************* */
/* File Name   : LBIT_int.h                                                 */
/* Author      : MAAM                                                       */
/* Version     : v01                                                        */
/* date        : Mar 24, 2023                                               */
/* description : Bitwise Library                                            */
/* ************************************************************************* */
/* ************************ HEADER FILES INCLUDES *************************  */
/* ************************************************************************* */

#ifndef LBIT_INT_H_
#define LBIT_INT_H_

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

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

#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 ASSIGN_BIT(REG,bit,value)                   do{                              \
                                                     REG &= ~(0x01u<<bit);          \
                                                     REG |=  ((value & 0x01u)<<bit); \
                                                    }while(0)
*/

/*            bits together in an u8 register          */
#define CON_u8Bits(b7, b6, b5, b4, b3, b2, b1, b0)                                    \
(0b##b7##b6##b5##b4##b3##b2##b1##b0)

/*            bits together in an u16 register         */
#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)

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

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

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


#endif /* LBIT_INT_H_ */
/*************************** E N D (LBIT_int.h) ******************************/
```

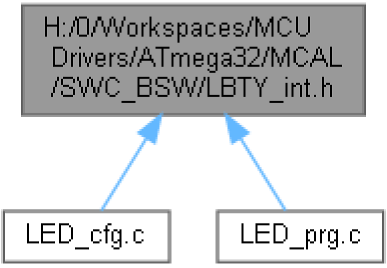
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

`#include <stdint.h>`

Include dependency graph for LBTY_int.h:


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

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


H:/0/Workspaces/MCU Drivers/ATmega32/MCAL /SWC_BSW/LBTY_int.h
LED_cfg.c
LED_prg.c

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	

```
               {
    LBTY_TRUE = 0x55,
    LBTY_FALSE = 0xAA
} 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	

```
              {
    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        /* Error is not available, wait for availability */
} LBTY_tenuErrorStatus;
```

enum LBTY_tenuFlagStatus

Flag Status type

Enumerator:

LBTY_RESET	
LBTY_SET	

```
              {
    LBTY_RESET = 0,
    LBTY_SET = !LBTY_RESET
} LBTY_tenuFlagStatus;
```

LBTY_int.h

Go to the documentation of this file.

```
/* ************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name   : LBTY_int.h                                                   */
/* Author      : MAAM                                                         */
/* Version     : v01                                                          */
/* date        : Mar 23, 2023                                                 */
/* description : Basic Library                                                */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES **************************  */
/* ************************************************************************** */

#ifndef _LBTY_INT_H_
#define _LBTY_INT_H_

#include <stdint.h>

/* ************************************************************************** */
/* ******************************* TYPE_DEF SECTION ************************* */
/* ************************************************************************** */

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;

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

/*************************************************************************/
#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 )

```

```c
#define LBTY_u64ZERO    ((u64)0x0000000000000000ULL)
#define LBTY_u64MAX     ((u64)0xFFFFFFFFFFFFFFFFULL)
#define LBTY_s64MAX     ((u64)0x7FFFFFFFFFFFFFFFLL )
#define LBTY_s64MIN     ((u64)0x8000000000000000LL )

/* ************************************************************************** */
/* ****************************** ENUM SECTION ****************************** */
/* ************************************************************************** */

typedef enum {
    LBTY_RESET = 0,
    LBTY_SET = !LBTY_RESET
} LBTY_tenuFlagStatus;

typedef enum {
  LBTY_TRUE = 0x55,
  LBTY_FALSE = 0xAA
} LBTY_tenuBoolean;

typedef enum {
  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        /* Error is not available, wait for availability */
} LBTY_tenuErrorStatus;

/* ************************************************************************** */
/* ****************************** STRUCT SECTION **************************** */
/* ************************************************************************** */

typedef union {
  struct {
    u8 m_u8b0 :1;       // LSB
    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;       // MSB
  } sBits;
  u8 u_u8Byte;
} LBTY_tuniPort8;

typedef union {
  struct  {
    u8 m_u8b0  :1;      // LSB
    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;      // MSB
  } sBits;
  struct {
    u8 m_u8low;
    u8 m_u8high;
  } sBytes;
  u16 u_u16Word;
} LBTY_tuniPort16;

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


#endif /* _LBTY_INT_H_ */
/*************************** 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.

```
/* ************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name   : LCTY_int.h                                                   */
/* Author      : MAAM                                                         */
/* Version     : v00                                                          */
/* date        : Apr 26, 2023                                                 */
/* description : Compiler Library                                             */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES **************************  */
/* ************************************************************************** */

#ifndef LCTY_INT_H_
#define LCTY_INT_H_

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

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

/* prog memory attribute */
#define LCTY_PROGMEM                __attribute__((__progmem__))

/* pure attribute */
#define LCTY_PURE                   __attribute__((__pure__))

/* Abstraction for inlining */
//#define LCTY_INLINE               static inline
#define LCTY_INLINE                 __attribute__((always_inline)) static inline

/* define function as interrupt handler */
#define LCTY_INTERRUPT              __attribute__((interrupt))

/* Memory packed to pass Memory padding */
#define CTY_PACKED                  __attribute__((__packed__))

/* Const attribute */
#define LCTY_CONST                  __attribute__((__const__))

/* place variable in direct page */
#define LCTY_DPAGE                  __attribute__((dp))

/* do not place variable in direct page */
#define LCTY_NODPAGE                __attribute__((nodp))

/* Sections */
#define LCTY_SECTION(section)       __attribute__((section( # section)))

/* Abstraction for assembly command */
# define LCTY_ASM(cmd)              __asm__ __volatile__ ( # cmd ::)

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

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

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


#endif /* LCTY_INT_H_ */
/*************************** E N D (LCTY_int.h) ******************************/
```