# SWC_LMD

Version v1.0

7/14/2023 8:58:00 PM

# 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:
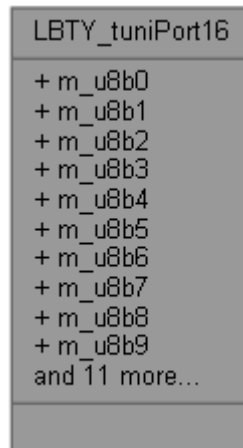


### Data Fields

- 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:
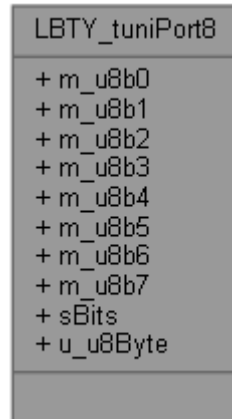


## Data Fields

- 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](#)

# File Documentation

## LMD_cfg.c File Reference

```
#include "LBTY_int.h"
#include "LMD_cfg.h"
```
Include dependency graph for LMD_cfg.c:
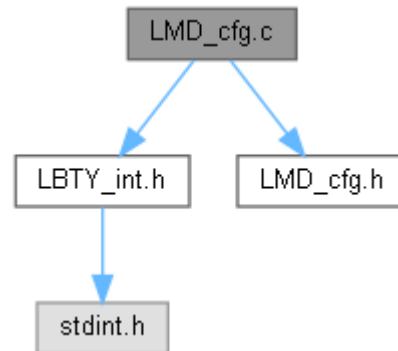


### Variables

- const u8 IMAGES [][LMD_MAT_LEN]
- const u8 IMAGES_LEN = sizeof(IMAGES)/8u

---

### Variable Documentation

**const u8 IMAGES[][LMD_MAT_LEN]**

**const u8 IMAGES_LEN = sizeof(IMAGES)/8u**

# LMD_cfg.h File Reference

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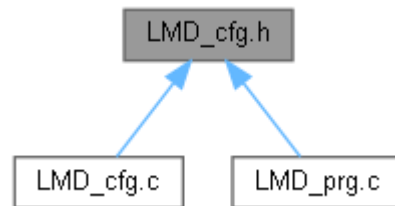


## Macros

- #define LMD_MAX_ARRAY_LEN  128u
- #define LMD_REFRESH_RATE  1u
- #define LMD_DELAY  5u
- #define LMD_CHAR_DELAY  2u
- #define LMD_CHAR_RATE  5u
- #define LMD_MAX_ROW_NUM  8u
- #define LMD_MAX_COL_NUM  8u
- #define LMD_MUX_TYPE  LMD_COL_MUX
- #define LMD_MAT_LEN  LMD_MAX_COL_NUM
- #define LMD_ROW_PORT  C
- #define LMD_ROW_ACTIVE  LMD_ACTIVE_HIGH
- #define LMD_COL_PORT  B
- #define LMD_COL_ACTIVE  LMD_ACTIVE_LOW

## Variables

- const u8 IMAGES [][LMD_MAT_LEN]
- const u8 IMAGES_LEN

## Macro Definition Documentation

**#define LMD_CHAR_DELAY  2u**

**#define LMD_CHAR_RATE  5u**

**#define LMD_COL_ACTIVE  LMD_ACTIVE_LOW**

**#define LMD_COL_PORT  B**

**#define LMD_DELAY  5u**

**#define LMD_MAT_LEN  LMD_MAX_COL_NUM**

**#define LMD_MAX_ARRAY_LEN  128u**

**#define LMD_MAX_COL_NUM  8u**

**#define LMD_MAX_ROW_NUM  8u**

**#define LMD_MUX_TYPE  LMD_COL_MUX**

**#define LMD_REFRESH_RATE  1u**

**#define LMD_ROW_ACTIVE  LMD_ACTIVE_HIGH**

**#define LMD_ROW_PORT  C**

---

## Variable Documentation

**const u8 IMAGES[][LMD_MAT_LEN][extern]**

**const u8 IMAGES_LEN[extern]**

## LMD_cfg.h

```
1 /* *************************************************************************** */
2 /* ********************* FILE DEFINITION SECTION ************************** */
3 /* *************************************************************************** */
4 /* File Name   : LMD_cfg.h                                                  */
5 /* Author      : MAAM                                                       */
6 /* Version     : v01.2                                                      */
7 /* date        : Apr 8, 2023                                                */
8 /* *************************************************************************** */
9 /* ********************** HEADER FILES INCLUDES *************************   */
10 /* *************************************************************************** */
11
12 #ifndef LMD_CFG_H_
13 #define LMD_CFG_H_
14
15 /* *************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ******************** */
17 /* *************************************************************************** */
18
19 /* *************************************************************************** */
20 /* *********************** MACRO/DEFINE SECTION *********************** */
21 /* *************************************************************************** */
22 /*
23 @ --- @ --- @ --- @
24 |     |     |     |---------- ROW0  }
25 @ --- @ --- @ --- @                 }
26 |     |     |     |--------- ROW1   }
27 @ --- @ --- @ --- @                 }<== OUTPUT
28 |     |     |     |---------- ROW2  }
29 @ --- @ --- @ --- @                 }
30 |     |     |     |--------- ROW3   }
31 |     |     |     |
32 COL0  COL1  COL2  COL3             <== OUTPUT
33 */
34
35 #define LMD_MAX_ARRAY_LEN       128u
36 #define LMD_REFRESH_RATE        1u
37 #define LMD_DELAY               5u
38 #define LMD_CHAR_DELAY          2u      //25u
39 #define LMD_CHAR_RATE           5u      //50u
40
41 #define LMD_MAX_ROW_NUM         8u
42 #define LMD_MAX_COL_NUM         8u
43
44 #define LMD_MUX_TYPE            LMD_COL_MUX
45 #define LMD_MAT_LEN             LMD_MAX_COL_NUM
46
47 #define LMD_ROW_PORT            C
48 #define LMD_ROW_ACTIVE          LMD_ACTIVE_HIGH
49
50 #define LMD_COL_PORT            B
51 #define LMD_COL_ACTIVE          LMD_ACTIVE_LOW
52
53 /* *************************************************************************** */
54 /* ************************** CONST SECTION *************************** */
55 /* *************************************************************************** */
56
57 /* *************************************************************************** */
58 /* ************************** VARIABLE SECTION ************************ */
59 /* *************************************************************************** */
60 extern const u8 IMAGES[][LMD_MAT_LEN];
61 extern const u8 IMAGES_LEN;
62 /* *************************************************************************** */
63 /* ************************** FUNCTION SECTION ************************ */
64 /* *************************************************************************** */
65
66 #endif /* LMD_CFG_H_ */
67 /*********************** E N D (LMD_cfg.h) ***************************/
```
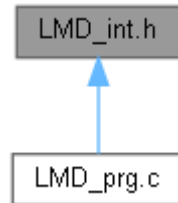
# LMD_int.h File Reference

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



## Functions

- void **LMD_vidInit** (void)
- **LBTY_tenuErrorStatus LMD_u8Display** (**u8** *const pu8Display)
- **LBTY_tenuErrorStatus LMD_u8Char** (**u8** u8Char)
- **LBTY_tenuErrorStatus LMD_u8String** (**u8** *pu8Str)
- void **LMD_vidSetString** (**u8** *pu8Str)
- **LBTY_tenuErrorStatus LMD_u8Update** (void)

---

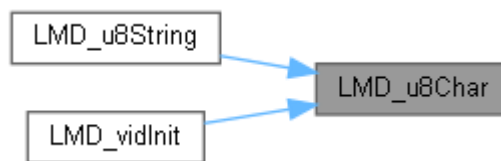## Function Documentation

### LBTY_tenuErrorStatus LMD_u8Char (u8   *u8Char*)

```
102                                                    {
103     LBTY_tenuErrorStatus  u8RetErrorState = LBTY_OK;
104     for(u8 i = LMD_CHAR_RATE ; i-- ; ){
105         if((u8RetErrorState = LMD_u8Display((u8*)IMAGES[u8Char]))){
106             break;
107         }
108     }
109     return u8RetErrorState;
110 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



### LBTY_tenuErrorStatus LMD_u8Display (u8 *const   *pu8Display*)

TODO: Update LDM with decoder

```
66                                                             {
67     LBTY_tenuErrorStatus u8RetErrorState = LBTY_OK;
68
70
71 #if LMD_MUX_TYPE == LMD_COL_MUX
72     for(u8 i = LMD_MAX_ROW_NUM ; i-- ; ){
73         GPIO_u8SetPortValue  (LMD_ROW_PORT, pu8Display[i]);
74         GPIO_u8TogglePinValue(LMD_COL_PORT, i);
75         vidMyDelay_ms(LMD_DELAY);
76         //GPIO_u8TogglePinValue(LMD_COL_PORT, i);
77         u8RetErrorState = GPIO_u8SetPortValue    (LMD_COL_PORT,
(u8)~LMD_COL_ACTIVE);
78     }
79 #endif
```

```
80 #if LMD_MUX_TYPE == LMD_ROW_MUX
81     for(u8 i = LMD_MAX_COL_NUM ; i-- ; ){
82         u8 u8ColChar = LBTY_u8ZERO;
83         for(u8 j = LMD_MAX_COL_NUM ; j-- ; ){
84             u8ColChar |= GET_BIT(pu8Display[j], i)<<(j);
85         }
86
87         GPIO_u8SetPortValue  (LMD_COL_PORT, ~u8ColChar);
88         GPIO_u8TogglePinValue(LMD_ROW_PORT, i);
89         vidMyDelay_ms(LMD_DELAY);
90         //GPIO_u8TogglePinValue(LMD_ROW_PORT, i);
91         u8RetErrorState = GPIO_u8SetPortValue    (LMD_ROW_PORT,
(u8)~LMD_ROW_ACTIVE);
92     }
93 #endif
94     return u8RetErrorState;
95 }
```

Here is the caller graph for this function:



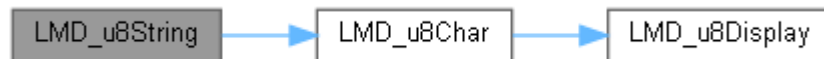### LBTY_tenuErrorStatus LMD_u8String (u8 *   *pu8Str*)

```
117                                              {
118     LBTY_tenuErrorStatus  u8RetErrorState = LBTY_OK;
119     while(*pu8Str){
120         if((u8RetErrorState = LMD_u8Char(*pu8Str))){
121             break;
122         }
123         if((u8RetErrorState = LMD_u8Char(' '))){
124             break;
125         }
126         pu8Str++;
127     }
128     return u8RetErrorState;
129 }
```

Here is the call graph for this function:



### LBTY_tenuErrorStatus LMD_u8Update (void )

```
143                                              {
144     LBTY_tenuErrorStatus u8RetErrorState = LBTY_OK;
145     static u8 u8Refresh = LBTY_u8ZERO;
146     static u8 u8Shift   = LBTY_u8ZERO;
147     static u8 u8index   = LBTY_u8ZERO;
148     static u8 pu8Char[LMD_MAT_LEN] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00};
149
150     if(u8Refresh++ > LMD_REFRESH_RATE){
151         u8Refresh =  LBTY_u8ZERO;
152         u8 u8TempChar = pu8Array_GLB[u8index];
153         for(u8 i = 0 ; i<LMD_MAT_LEN ; i++){
154             pu8Char[i] = pu8Char[i] >> 1 | (GET_BIT(IMAGES[u8TempChar][i],
u8Shift)) << (LMD_MAT_LEN - 1);
155         }
156         if(u8Shift<LMD_MAT_LEN){
157             u8Shift++;
158         }else{
159             u8Shift = LBTY_u8ZERO;
160             if(!pu8Array_GLB[++u8index]){
161                 u8index = 0;
162                 u8RetErrorState = LBTY_NOK;
163             }
164         }
165     }
166
```

13

```
167      LMD u8Display(pu8Char);
168      return u8RetErrorState;
169 }
```

Here is the call graph for this function:



### void LMD_vidInit (void )

```
52                        {
53      GPIO_u8SetPortDirection(LMD ROW PORT, PORT_OUTPUT);
54      GPIO_u8SetPortValue    (LMD ROW PORT, LMD ROW ACTIVE);
55      GPIO_u8SetPortDirection(LMD COL PORT, PORT_OUTPUT);
56      GPIO u8SetPortValue    (LMD COL PORT, LMD COL ACTIVE);
57
58      LMD u8Char(0);
59 }
```

Here is the call graph for this function:



### void LMD_vidSetString (u8 *   *pu8Str*)

```
136                                     {
137      u8 i = 0;
138      for( ; *pu8Str ; i++){
139          pu8Array GLB[i] = *pu8Str++;
140      }
141      pu8Array GLB[i] = *pu8Str;
142 }
```

# LMD_int.h

```c
1 /* *************************************************************************** */
2 /* ********************* FILE DEFINITION SECTION ************************* */
3 /* *************************************************************************** */
4 /* File Name   : LMD_int.h                                               */
5 /* Author      : MAAM                                                    */
6 /* Version     : v01.2                                                   */
7 /* date        : Apr 8, 2023                                             */
8 /* *************************************************************************** */
9 /* ********************** HEADER FILES INCLUDES ************************** */
10 /* *************************************************************************** */
11
12 #ifndef LMD_INT_H_
13 #define LMD_INT_H_
14
15 /* *************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ******************** */
17 /* *************************************************************************** */
18
19 /* *************************************************************************** */
20 /* ************************* MACRO/DEFINE SECTION ************************* */
21 /* *************************************************************************** */
22
23 /* *************************************************************************** */
24 /* ************************** CONST SECTION ****************************** */
25 /* *************************************************************************** */
26
27 /* *************************************************************************** */
28 /* ************************** VARIABLE SECTION ************************** */
29 /* *************************************************************************** */
30
31 /* *************************************************************************** */
32 /* ************************* FUNCTION SECTION ************************** */
33 /* *************************************************************************** */
34
35 /* *************************************************************************** */
36 /* Description :    LED Matrix Display initialization                    */
37 /* Input       :    void                                                 */
38 /* Return      :    void                                                 */
39 /* *************************************************************************** */
40 extern void LMD_vidInit(void);
41
42 /* *************************************************************************** */
43 /* Description :    LED Matrix Display Put char Output                   */
44 /* Input       :    pu8Display                                           */
45 /* Return      :    LBTY_tenuErrorStatus                                 */
46 /* *************************************************************************** */
47 extern LBTY_tenuErrorStatus LMD_u8Display(u8* const pu8Display);
48
49 /* *************************************************************************** */
50 /* Description :    LED Matrix Display Put Show char                     */
51 /* Input       :    u8Char                                              */
52 /* Return      :    LBTY_tenuErrorStatus                                 */
53 /* *************************************************************************** */
54 extern LBTY_tenuErrorStatus LMD_u8Char(u8 u8Char);
55
56 /* *************************************************************************** */
57 /* Description :    LED Matrix Display Show char                         */
58 /* Input       :    u8Char                                              */
59 /* Return      :    LBTY_tenuErrorStatus                                 */
60 /* *************************************************************************** */
61 extern LBTY_tenuErrorStatus LMD_u8String(u8* pu8Str);
62
63 /* *************************************************************************** */
64 /* Description :    LED Matrix Display Update shifting                   */
65 /* Input       :    void                                                 */
66 /* Return      :    LBTY tenuErrorStatus                                 */
67 /* *************************************************************************** */
68 extern void LMD_vidSetString(u8* pu8Str);
69 extern LBTY_tenuErrorStatus LMD_u8Update(void);
70
71
72 #endif /* LMD_INT_H_ */
```
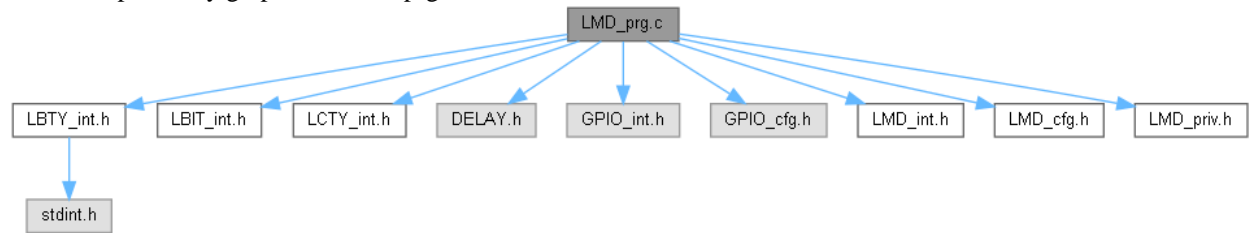
```
73 /************************* E N D (LMD_int.h) ****************************/
```

# LMD_prg.c File Reference

```
#include "LBTY_int.h"
#include "LBIT_int.h"
#include "LCTY_int.h"
#include "DELAY.h"
#include "GPIO_int.h"
#include "GPIO_cfg.h"
#include "LMD_int.h"
#include "LMD_cfg.h"
#include "LMD_priv.h"
```
Include dependency graph for LMD_prg.c:



## Functions

- void **LMD_vidInit** (void)
- **LBTY_tenuErrorStatus** **LMD_u8Display** (**u8** *const pu8Display)
- **LBTY_tenuErrorStatus** **LMD_u8Char** (**u8** u8Char)
- **LBTY_tenuErrorStatus** **LMD_u8String** (**u8** *pu8Str)
- void **LMD_vidSetString** (**u8** *pu8Str)
- **LBTY_tenuErrorStatus** **LMD_u8Update** (void)

## Variables

- static **u8** **pu8Array_GLB** [**LMD_MAX_ARRAY_LEN**]

## Function Documentation
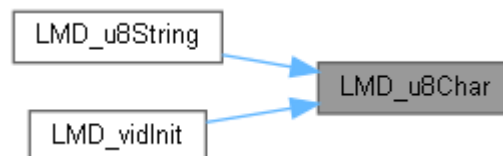
### **LBTY_tenuErrorStatus** LMD_u8Char (**u8**  *u8Char*)

```
102                                              {
103      LBTY_tenuErrorStatus  u8RetErrorState = LBTY_OK;
104      for(u8 i = LMD_CHAR_RATE ; i-- ; ){
105          if((u8RetErrorState = LMD_u8Display((u8*)IMAGES[u8Char]))){
106              break;
107          }
108      }
109      return u8RetErrorState;
110 }
```
Here is the call graph for this function:


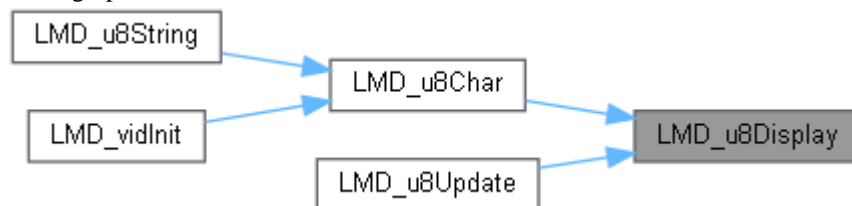
Here is the caller graph for this function:

**LBTY_tenuErrorStatus LMD_u8Display (u8 \*const   *pu8Display*)**

TODO: Update LDM with decoder

```
66                                                               {
67       LBTY_tenuErrorStatus u8RetErrorState = LBTY_OK;
68
70
71  #if LMD_MUX_TYPE == LMD_COL_MUX
72       for(u8 i = LMD_MAX_ROW_NUM ; i-- ; ){
73            GPIO_u8SetPortValue  (LMD_ROW_PORT, pu8Display[i]);
74            GPIO_u8TogglePinValue(LMD_COL_PORT, i);
75            vidMyDelay_ms(LMD_DELAY);
76            //GPIO_u8TogglePinValue(LMD_COL_PORT, i);
77            u8RetErrorState = GPIO_u8SetPortValue    (LMD_COL_PORT,
(u8)~LMD_COL_ACTIVE);
78       }
79  #endif
80  #if LMD_MUX_TYPE == LMD_ROW_MUX
81       for(u8 i = LMD_MAX_COL_NUM ; i-- ; ){
82            u8 u8ColChar = LBTY_u8ZERO;
83            for(u8 j = LMD_MAX_COL_NUM ; j-- ; ){
84                u8ColChar |= GET_BIT(pu8Display[j], i)<<(j);
85            }
86
87            GPIO_u8SetPortValue  (LMD_COL_PORT, ~u8ColChar);
88            GPIO_u8TogglePinValue(LMD_ROW_PORT, i);
89            vidMyDelay_ms(LMD_DELAY);
90            //GPIO_u8TogglePinValue(LMD_ROW_PORT, i);
91            u8RetErrorState = GPIO_u8SetPortValue    (LMD_ROW_PORT,
(u8)~LMD_ROW_ACTIVE);
92       }
93  #endif
94       return u8RetErrorState;
95  }
```

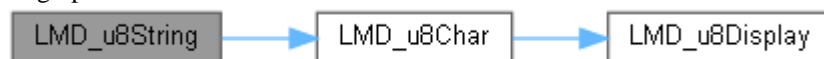Here is the caller graph for this function:



**LBTY_tenuErrorStatus LMD_u8String (u8 \*   *pu8Str*)**

```
117                                                               {
118      LBTY_tenuErrorStatus  u8RetErrorState = LBTY_OK;
119      while(*pu8Str){
120          if((u8RetErrorState = LMD_u8Char(*pu8Str))){
121              break;
122          }
123          if((u8RetErrorState = LMD_u8Char(' '))){
124              break;
125          }
126          pu8Str++;
127      }
128      return u8RetErrorState;
129  }
```

Here is the call graph for this function:



**LBTY_tenuErrorStatus LMD_u8Update (void )**

```
143                                                               {
144      LBTY_tenuErrorStatus u8RetErrorState = LBTY_OK;
145      static u8 u8Refresh = LBTY_u8ZERO;
146      static u8 u8Shift   = LBTY_u8ZERO;
147      static u8 u8index   = LBTY_u8ZERO;
148      static u8 pu8Char[LMD_MAT_LEN] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00};
```

```
149
150     if(u8Refresh++ > LMD_REFRESH_RATE){
151         u8Refresh =  LBTY_u8ZERO;
152         u8 u8TempChar = pu8Array_GLB[u8index];
153         for(u8 i = 0 ; i<LMD_MAT_LEN ; i++){
154             pu8Char[i] = pu8Char[i] >> 1 | (GET_BIT(IMAGES[u8TempChar][i],
u8Shift)) << (LMD_MAT_LEN - 1);
155         }
156         if(u8Shift<LMD_MAT_LEN){
157             u8Shift++;
158         }else{
159             u8Shift = LBTY_u8ZERO;
160             if(!pu8Array_GLB[++u8index]){
161                 u8index = 0;
162                 u8RetErrorState = LBTY_NOK;
163             }
164         }
165     }
166
167     LMD_u8Display(pu8Char);
168     return u8RetErrorState;
169 }
```

Here is the call graph for this function:
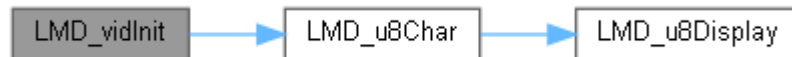


### void LMD_vidInit (void )

```
52                          {
53     GPIO_u8SetPortDirection(LMD_ROW_PORT, PORT_OUTPUT);
54     GPIO_u8SetPortValue    (LMD_ROW_PORT, LMD_ROW_ACTIVE);
55     GPIO_u8SetPortDirection(LMD_COL_PORT, PORT_OUTPUT);
56     GPIO_u8SetPortValue    (LMD_COL_PORT, LMD_COL_ACTIVE);
57
58     LMD_u8Char(0);
59 }
```

Here is the call graph for this function:



### void LMD_vidSetString (u8 *   pu8Str)

```
136                              {
137     u8 i = 0;
138     for( ; *pu8Str ; i++){
139         pu8Array_GLB[i] = *pu8Str++;
140     }
141     pu8Array_GLB[i] = *pu8Str;
142 }
```
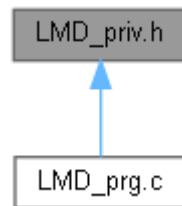
---

## Variable Documentation

### u8 pu8Array_GLB[LMD_MAX_ARRAY_LEN][static]

# LMD_priv.h File Reference

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



## Macros

- #define LMD_ACTIVE_HIGH   PORT_High
- #define LMD_ACTIVE_LOW   PORT_Low
- #define LMD_ROW_MUX   1u
- #define LMD_COL_MUX   2u

## Macro Definition Documentation

**#define LMD_ACTIVE_HIGH   PORT_High**

**#define LMD_ACTIVE_LOW   PORT_Low**

**#define LMD_COL_MUX   2u**

**#define LMD_ROW_MUX   1u**

## LMD_priv.h

Go to the documentation of this file.

```c
1  /* ************************************************************************** */
2  /* ********************* FILE DEFINITION SECTION ************************** */
3  /* ************************************************************************** */
4  /* File Name   : LMD_priv.h                                               */
5  /* Author      : MAAM                                                     */
6  /* Version     : v01.2                                                    */
7  /* date        : Apr 8, 2023                                             */
8  /* ************************************************************************** */
9  /* *********************** HEADER FILES INCLUDES *********************** */
10 /* ************************************************************************** */
11
12 #ifndef LMD_PRIV_H_
13 #define LMD_PRIV_H_
14
15 /* ************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ****************** */
17 /* ************************************************************************** */
18
19 /* ************************************************************************** */
20 /* *********************** MACRO/DEFINE SECTION *********************** */
21 /* ************************************************************************** */
22
23 #define LMD_ACTIVE_HIGH         PORT_High
24 #define LMD_ACTIVE_LOW          PORT_Low
25
26 #define LMD_ROW_MUX             1u
27 #define LMD_COL_MUX             2u
28
29 /* ************************************************************************** */
30 /* *************************** CONST SECTION ************************** */
31 /* ************************************************************************** */
32
33 /* ************************************************************************** */
34 /* ************************** VARIABLE SECTION *********************** */
35 /* ************************************************************************** */
36
37 /* ************************************************************************** */
38 /* ************************** FUNCTION SECTION ********************** */
39 /* ************************************************************************** */
40
41
42 #endif /* LMD_PRIV_H_ */
43 /************************ E N D (LMD_priv.h) ****************************/
```

# main.c File Reference

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

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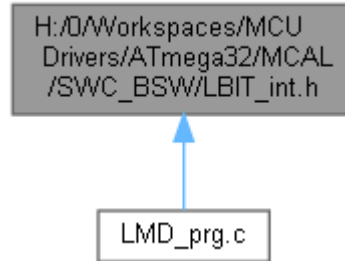


## Macros

- #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

```c
#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

```
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:
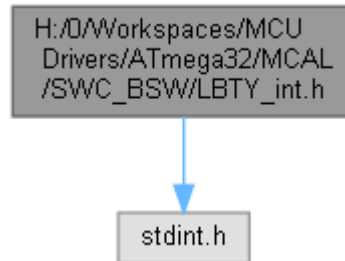


This graph shows which files directly or indirectly include this file:
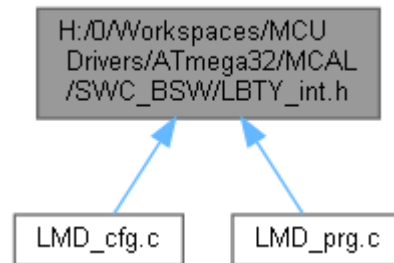


## Data Structures

- union **LBTY_tuniPort8**union **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_PO INTER | |
| LBTY_INDEX_O UT_OF_RANGE | |
| LBTY_NO_MAS TER_CHANNEL | |
| LBTY_READ_ER ROR | |
| LBTY_WRITE_E RROR | |
| LBTY_UNDEFIN ED_ERROR | |
| LBTY_IN_PROG RESS | |

```
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

```
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
```

Go to the documentation of this file.

33

```c
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:
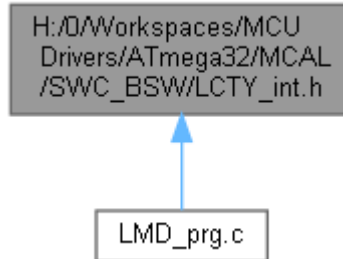


## Macros

- #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

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