SWC_SEG

Version v1.0
7/14/2023 8:41:00 PM

Table of Contents

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

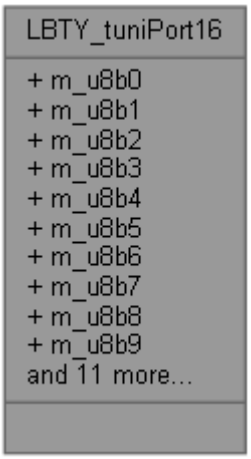
File List

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

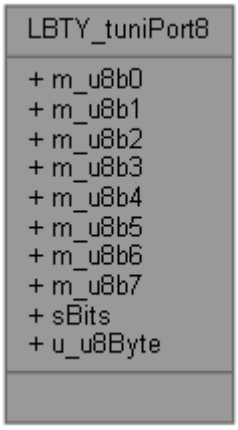
u16 u_u16Word

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

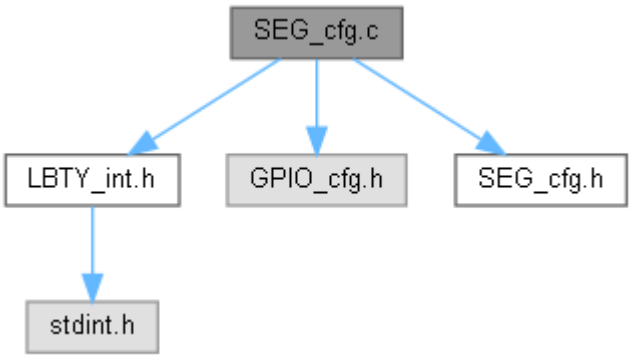
File Documentation

main.c File Reference

SEG_cfg.c File Reference

```
#include "LBTY_int.h"
#include "GPIO_cfg.h"
#include "SEG_cfg.h"
```

Include dependency graph for SEG_cfg.c:



Variables

- const u8 kau8SegPins [] = {0, 1, 2, 3, 4, 5, 6, 7}
- const u8 kau8SegDigits []

Variable Documentation

const u8 kau8SegDigits[]

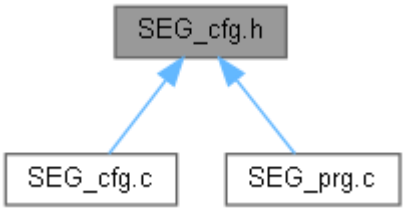
```
Initial value:= {
        0x3F,
        0x06,
        0x5B,
        0x4F,
        0x66,
        0x6D,
        0x7D,
        0x07,
        0x7F,
        0x67,
        0x77,
        0x7C,
        0x39,
        0x5E,
        0x79,
        0x71

}
```

const u8 kau8SegPins[] = {0, 1, 2, 3, 4, 5, 6, 7}

SEG_cfg.h File Reference

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



Macros

- #define SEG_PORT_COM0 C
- #define SEG_PIN_COM0 0u
- #define SEG_PORT_COM1 C
- #define SEG_PIN_COM1 1u
- #define SEG_PORT_COM2 C
- #define SEG_PIN_COM2 2u
- #define SEG_PORT_COM3 C
- #define SEG_PIN_COM3 3u
- #define SEG_PORT_COM4 C
- #define SEG_PIN_COM4 4u
- #define SEG_PORT_COM5 C
- #define SEG_PIN_COM5 5u
- #define SEG_PINS 8u
- #define SEG_PORT_DATA B
- #define SEG_a 0u
- #define SEG_b 1u
- #define SEG_c 2u
- #define SEG_d 3u
- #define SEG_e 4u
- #define SEG_f 5u
- #define SEG_g 6u
- #define SEG_PORT_DOT B
- #define SEG_h 7u
- #define SEG_FLOAT_DOT 1u
- #define SEG_FLOAT_MUL 10u
- #define SEG_DELAY 5u
- #define SEG_NUM_DELAY 15u
- #define SEG_NUM_RATE 25u

Macro Definition Documentation

#define SEG_a 0u

#define SEG_b 1u

#define SEG_c 2u

#define SEG_d 3u

#define SEG_DELAY 5u

#define SEG_e 4u

#define SEG_f 5u

#define SEG_FLOAT_DOT 1u

#define SEG_FLOAT_MUL 10u

#define SEG_g 6u

#define SEG_h 7u

#define SEG_NUM_DELAY 15u

#define SEG_NUM_RATE 25u

#define SEG_PIN_COM0 0u

#define SEG_PIN_COM1 1u

#define SEG_PIN_COM2 2u

#define SEG_PIN_COM3 3u

#define SEG_PIN_COM4 4u

#define SEG_PIN_COM5 5u

#define SEG_PINS 8u

#define SEG_PORT_COM0 C

#define SEG_PORT_COM1 C

#define SEG_PORT_COM2 C

#define SEG_PORT_COM3 C

#define SEG_PORT_COM4 C

```
#define SEG_PORT_COM5   C

#define SEG_PORT_DATA   B

#define SEG_PORT_DOT   B
```

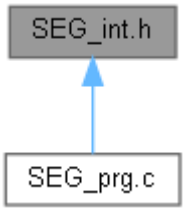
SEG_cfg.h

```
Go to the documentation of this file.1 /*
********************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ********************************************************************************** */
4 /* File Name   : SEG_cfg.h                                                        */
5 /* Author      : MAAM                                                             */
6 /* Version     : v01.2                                                            */
7 /* date        : Mar 25, 2023                                                     */
8 /* ********************************************************************************** */
9 /* ************************ HEADER FILES INCLUDES ************************  */
10 /* ********************************************************************************** */
11
12 #ifndef SEG_CFG_H_
13 #define SEG_CFG_H_
14
15 /* ********************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ********************************************************************************** */
18
19 /* ********************************************************************************** */
20 /* ************************** MACRO/DEFINE SECTION ************************** */
21 /* ********************************************************************************** */
22
23 #if defined(AMIT_KIT)
24
25 #define SEG_DECODER
26 #define SEG_PORT_COM0       C
27 #define SEG_PIN_COM0        AMIT_7Seg_COM0
28 #define SEG_PORT_COM1       C
29 #define SEG_PIN_COM1        AMIT_7Seg_COM1
30
31 #define SEG_PINS            4u
32 #define SEG_PORT_DATA       C
33 #define SEG_A               AMIT_7Seg_A
34 #define SEG_B               AMIT_7Seg_B
35 #define SEG_C               AMIT_7Seg_C
36 #define SEG_D               AMIT_7Seg_D
37
38 #define SEG_PORT_DOT        C
39 #define SEG_h               AMIT_C0
40
41 #elif defined(ETA32_KIT)
42
43 #define SEG DECODER
44 #define SEG_PORT_COM0       B
45 #define SEG_PIN_COM0        Eta32_7Seg_COM0
46 #define SEG_PORT_COM1       B
47 #define SEG_PIN_COM1        Eta32_7Seg_COM1
48 #define SEG_PORT_COM2       A
49 #define SEG PIN COM2        Eta32 7Seg COM2
50 #define SEG_PORT_COM3       A
51 #define SEG_PIN_COM3        Eta32_7Seg_COM3
52
53 #define SEG_PINS            4u
54 #define SEG PORT DATA       B
55 #define SEG_A               Eta32_7Seg_A
56 #define SEG_B               Eta32_7Seg_B
57 #define SEG_C               Eta32_7Seg_C
58 #define SEG_D               Eta32_7Seg_D
59
60 #define SEG PORT DOT        B
61 #define SEG_h               Eta32_LED_R
62
63 #elif defined(ETA32_MINI_KIT)
64
65 #define SEG_PORT_COM0       C
66 #define SEG PIN COM0        Eta32 mini 7Seg COM0
67 #define SEG_PORT_COM1       C
68 #define SEG_PIN_COM1        Eta32_mini_7Seg_COM1
69
70 #define SEG_PINS            8u
71 #define SEG_PORT_DATA       A
72 #define SEG_a               Eta32_mini_7Seg_A
```

```
73 #define SEG_b               Eta32_mini_7Seg_B
74 #define SEG_c               Eta32_mini_7Seg_C
75 #define SEG_d               Eta32_mini_7Seg_D
76 #define SEG_e               Eta32_mini_7Seg_E
77 #define SEG_f               Eta32_mini_7Seg_F
78 #define SEG_g               Eta32_mini_7Seg_G
79
80 #define SEG_PORT_DOT        B
81 #define SEG_h               Eta32_mini_7Seg_Dot
82
83 #else
84
85 #define SEG_PORT_COM0       C
86 #define SEG_PIN_COM0        0u
87 #define SEG_PORT_COM1       C
88 #define SEG_PIN_COM1        1u
89 #define SEG_PORT_COM2       C
90 #define SEG_PIN_COM2        2u
91 #define SEG_PORT_COM3       C
92 #define SEG_PIN_COM3        3u
93 #define SEG_PORT_COM4       C
94 #define SEG_PIN_COM4        4u
95 #define SEG_PORT_COM5       C
96 #define SEG_PIN_COM5        5u
97
98 #define SEG_PINS            8u
99 #define SEG_PORT_DATA       B
100 #define SEG_a               0u
101 #define SEG_b               1u
102 #define SEG_c               2u
103 #define SEG_d               3u
104 #define SEG_e               4u
105 #define SEG_f               5u
106 #define SEG_g               6u
107
108 #define SEG_PORT_DOT        B
109 #define SEG_h               7u
110
111 #endif
112
113 #define SEG_FLOAT_DOT       1u
114 #define SEG_FLOAT_MUL       10u
115
116 #define SEG_DELAY           5u
117 #define SEG_NUM_DELAY       15u
118 #define SEG_NUM_RATE        25u
119
120 /* ********************************************************************** */
121 /* ***************************** CONST SECTION ***************************** */
122 /* ********************************************************************** */
123
124 /* ********************************************************************** */
125 /* ***************************** VARIABLE SECTION ************************** */
126 /* ********************************************************************** */
127
128 /* ********************************************************************** */
129 /* ***************************** FUNCTION SECTION ************************** */
130 /* ********************************************************************** */
131
132
133 #endif /* SEG CFG H  */
134 /************************** E N D (SEG_cfg.h) ******************************/
```

SEG_int.h File Reference

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



Functions

- void SEG_vidInit (void)
- void SEG_vidDisplay (u16 u16NumValue, u8 u8Dot)
- void SEG_vidDisplayFloat (f32 f32NumValue)
- void SEG_vidDisplayFloat_Blink (f32 f32NumValue)
- void SEG_vidDisplayNum (u16 u16NumValue)
- void SEG_vidDisplayNum_Blink (u16 u16NumValue)
- void SEG_vidDispalyDigit (u8 u8DigitValue, u8 u8PortCom, u8 u8PinCom, u8 u8Dot)
- void SEG_vidSetNum (u16 u16NumValue, u8 u8Dot)
- LBTY_tenuErrorStatus SEG_u8Update (void)

Function Documentation

LBTY_tenuErrorStatus SEG_u8Update (void)

```
247                                             {
248     LBTY_tenuErrorStatus u8RetErrorState = LBTY_OK;
249     static u8 u8Com = LBTY_u8ZERO;
250 
251     u16 u16NumValue = u16NumValue_GLB;
252     u8 u8Dot        = u8Dot_GLB;
253 
254     switch(u8Com){
255 #ifdef SEG_PIN_COM0
256     case SEG_COM0:
257         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM0, SEG_PIN_COM0,
(u8Dot == SEG_COM0));
258         break;
259 #endif
260 
261 #ifdef SEG_PIN_COM1
262     case SEG_COM1:
263         u16NumValue /= 10u;
264         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM1, SEG_PIN_COM1,
(u8Dot == SEG_COM1));
265         break;
266 #endif
267 
268 #ifdef SEG_PIN_COM2
269     case SEG_COM2:
270         u16NumValue /= 10u;
271         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM2, SEG_PIN_COM2,
(u8Dot == SEG_COM2));
272         break;
273 #endif
274 
275 #ifdef SEG_PIN_COM3
276     case SEG_COM3:
277         u16NumValue /= 10u;
278         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM3, SEG_PIN_COM3,
(u8Dot == SEG_COM3));
279         break;
280 #endif
```

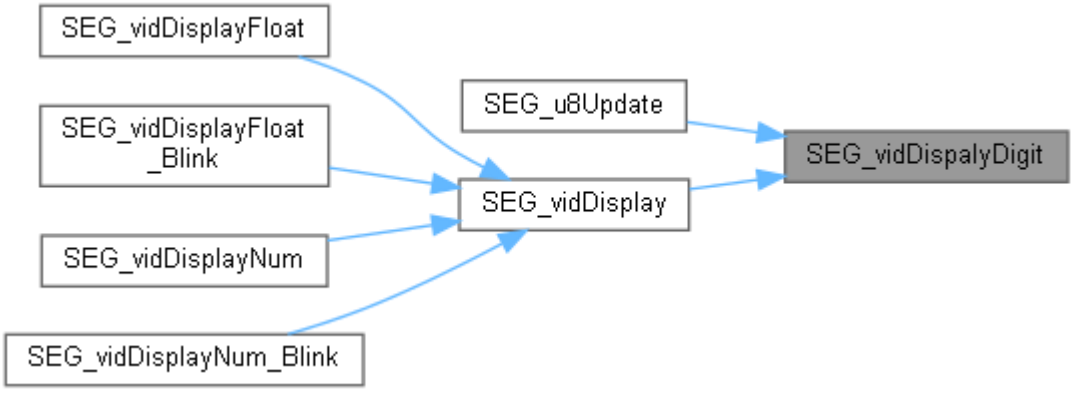
```
281
282 #ifdef SEG_PIN_COM4
283     case SEG_COM4:
284         u16NumValue /= 10u;
285         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM4, SEG_PIN_COM4,
(u8Dot == SEG_COM4));
286         break;
287 #endif
288
289 #ifdef SEG_PIN_COM5
290     case SEG_COM5:
291         u16NumValue /= 10u;
292         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM5, SEG_PIN_COM5,
(u8Dot == SEG_COM5));
293         break;
294 #endif
295     default:
296         break;
297     }
298     u8Com = (u8Com + 1) % SEG_NUM;
299
300     if(!u8Com){
301         u8RetErrorState = LBTY_NOK;
302     }
303     return u8RetErrorState;
304
305 }
```

Here is the call graph for this function:

void SEG_vidDispalyDigit (u8 *u8DigitValue*, u8 *u8PortCom*, u8 *u8PinCom*, u8 *u8Dot*)

```
220
{
221
222 #ifdef  SEG_DECODER
223     u8 u8PortValue_LOC = 0;
224     GPIO_u8GetPortValue(SEG_PORT_DATA, &u8PortValue_LOC);
225     for(u8 i = SEG_PINS ; i-- ; ){
226         GPIO_u8SetPinValue  (SEG_PORT_DATA, kau8SegDecoderPort[i],
GET_BIT(u8DigitValue, i));
227     }
228 #else
229     GPIO_u8SetPortValue(SEG_PORT_DATA, kau8SegDigits[u8DigitValue] << SEG_a);
230 #endif
231
232     GPIO_u8SetPinValue (SEG_PORT_DOT, SEG_h, u8Dot);
233     GPIO_u8SetPinValue (u8PortCom, u8PinCom, PIN_High);
234     vidMyDelay_ms(SEG_DELAY);
235     GPIO_u8SetPinValue (u8PortCom, u8PinCom, PIN_Low);
236 }
```

Here is the caller graph for this function:



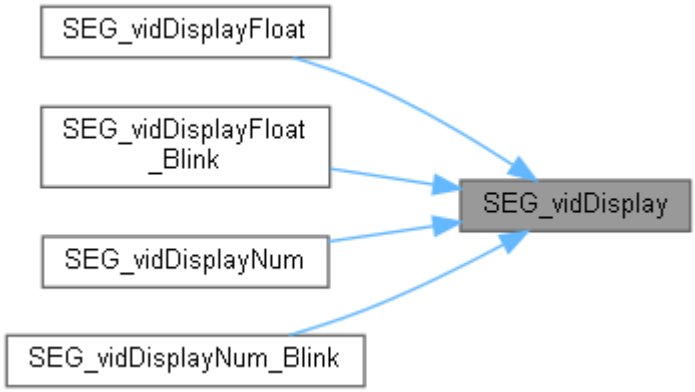
void SEG_vidDisplay (u16 *u16NumValue*, u8 *u8Dot*)

```
127                                                 {
```

```
128 #ifdef SEG_PIN_COM0
129     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM0, SEG_PIN_COM0, (u8Dot ==
SEG_COM0));
130 #endif
131
132 #ifdef SEG_PIN_COM1
133     u16NumValue /= 10u;
134     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM1, SEG_PIN_COM1, (u8Dot ==
SEG_COM1));
135 #endif
136
137 #ifdef SEG_PIN_COM2
138     u16NumValue /= 10u;
139     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM2, SEG_PIN_COM2, (u8Dot ==
SEG_COM2));
140 #endif
141
142 #ifdef SEG_PIN_COM3
143     u16NumValue /= 10u;
144     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM3, SEG_PIN_COM3, (u8Dot ==
SEG_COM3));
145 #endif
146
147 #ifdef SEG_PIN_COM4
148     u16NumValue /= 10u;
149     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM4, SEG_PIN_COM4, (u8Dot ==
SEG_COM4));
150 #endif
151
152 #ifdef SEG_PIN_COM5
153     u16NumValue /= 10u;
154     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM5, SEG_PIN_COM5, (u8Dot ==
SEG_COM5));
155 #endif
156
157 }
```

Here is the call graph for this function:

Here is the caller graph for this function:



void SEG_vidDisplayFloat (f32 *f32NumValue*)

```
164                                             {
165     SEG_vidDisplay((u16)(f32NumValue * SEG_FLOAT_MUL), SEG_FLOAT_DOT);
166 }
```

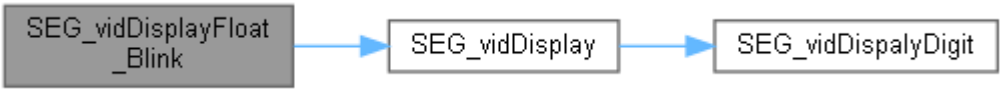
Here is the call graph for this function:

void SEG_vidDisplayFloat_Blink (f32 *f32NumValue*)

```
168                                                 {
169     static u8 u8Tick = LBTY_u8ZERO;
170     static u8 u8Blink = LBTY_SET;
171
172     if((++u8Tick > SEG_NUM_DELAY)){
```

```
173         u8Blink = LBTY_RESET;
174         if(u8Tick >= SEG_NUM_RATE){
175             u8Tick = LBTY_u8ZERO;
176             u8Blink = LBTY_SET;
177         }
178     }
179
180     if(u8Blink){
181         SEG_vidDisplay((u16)(f32NumValue * SEG_FLOAT_MUL), SEG_FLOAT_DOT);
182     }else{
183         vidMyDelay_ms(SEG_DELAY * SEG_NUM * 2);
184     }
185 }
```

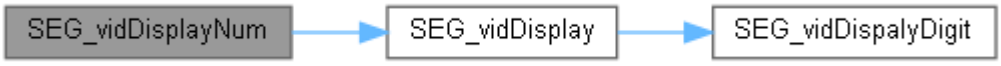
Here is the call graph for this function:



void SEG_vidDisplayNum (u16 *u16NumValue*)

```
192                                             {
193     SEG_vidDisplay(u16NumValue, LBTY_u8MAX);
194 }
```

Here is the call graph for this function:



void SEG_vidDisplayNum_Blink (u16 *u16NumValue*)

```
196                                                 {
197     static u8 u8Tick = LBTY_u8ZERO;
198     static u8 u8Blink = LBTY_SET;
199
200     if((++u8Tick > SEG_NUM_DELAY)){
201         u8Blink = LBTY_RESET;
202         if(u8Tick >= SEG_NUM_RATE){
203             u8Tick = LBTY_u8ZERO;
204             u8Blink = LBTY_SET;
205         }
206     }
207
208     if(u8Blink){
209         SEG_vidDisplay(u16NumValue, LBTY_u8MAX);
210     }else{
211         vidMyDelay_ms(SEG_DELAY * SEG_NUM * SEG_DELAY);
212     }
213 }
```

Here is the call graph for this function:

void SEG_vidInit (void)

```
59                      {
60
61 #ifdef SEG_PIN_COM0
62     GPIO_u8SetPinDirection(SEG_PORT_COM0, SEG_PIN_COM0, PIN_OUTPUT);
63     GPIO_u8SetPinValue    (SEG_PORT_COM0, SEG_PIN_COM0, PIN_Low);
64 #endif
65
66 #ifdef SEG_PIN_COM1
67     GPIO_u8SetPinDirection(SEG_PORT_COM1, SEG_PIN_COM1, PIN_OUTPUT);
68     GPIO_u8SetPinValue    (SEG_PORT_COM1, SEG_PIN_COM1, PIN_Low);
69 #endif
70
71 #ifdef SEG_PIN_COM2
72     GPIO_u8SetPinDirection(SEG_PORT_COM2, SEG_PIN_COM2, PIN_OUTPUT);
73     GPIO_u8SetPinValue    (SEG_PORT_COM2, SEG_PIN_COM2, PIN_Low);
74 #endif
75
76 #ifdef SEG_PIN_COM3
77     GPIO_u8SetPinDirection(SEG_PORT_COM3, SEG_PIN_COM3, PIN_OUTPUT);
78     GPIO_u8SetPinValue    (SEG_PORT_COM3, SEG_PIN_COM3, PIN_Low);
```

```
79 #endif
80
81 #ifdef SEG_PIN_COM4
82     GPIO_u8SetPinDirection(SEG_PORT_COM4, SEG_PIN_COM4, PIN_OUTPUT);
83     GPIO_u8SetPinValue    (SEG_PORT_COM4, SEG_PIN_COM4, PIN_Low);
84 #endif
85
86 #ifdef SEG_PIN_COM5
87     GPIO_u8SetPinDirection(SEG_PORT_COM5, SEG_PIN_COM5, PIN_OUTPUT);
88     GPIO_u8SetPinValue    (SEG_PORT_COM5, SEG_PIN_COM5, PIN_Low);
89 #endif
90
91 #ifdef  SEG_DECODER
92     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_A, PIN_OUTPUT);
93     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_A, PIN_Low);
94     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_B, PIN_OUTPUT);
95     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_B, PIN_Low);
96     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_C, PIN_OUTPUT);
97     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_C, PIN_Low);
98     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_D, PIN_OUTPUT);
99     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_D, PIN_Low);
100 #else
101     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_a, PIN_OUTPUT);
102     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_a, PIN_Low);
103     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_b, PIN_OUTPUT);
104     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_b, PIN_Low);
105     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_c, PIN_OUTPUT);
106     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_c, PIN_Low);
107     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_d, PIN_OUTPUT);
108     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_d, PIN_Low);
109     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_e, PIN_OUTPUT);
110     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_e, PIN_Low);
111     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_f, PIN_OUTPUT);
112     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_f, PIN_Low);
113     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_g, PIN_OUTPUT);
114     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_g, PIN_Low);
115 #endif
116
117 GPIO_u8SetPinDirection(SEG_PORT_DOT, SEG_h, PIN_OUTPUT);
118 GPIO_u8SetPinValue    (SEG_PORT_DOT, SEG_h, PIN_Low);
119
120 }
```

void SEG_vidSetNum (u16 *u16NumValue*, u8 *u8Dot*)

```
243                                                 {
244     u16NumValue_GLB = u16NumValue;
245     u8Dot_GLB       = u8Dot;
246 }
```

SEG_int.h

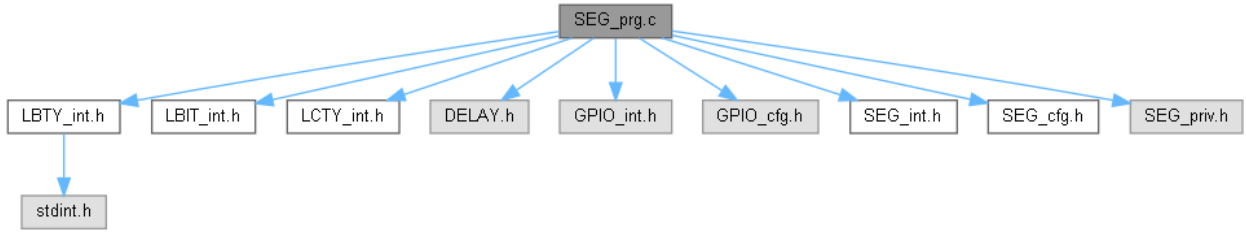
```
Go to the documentation of this file.1 /*
****************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ****************************************************************************** */
4 /* File Name   : SEG_int.h                                                    */
5 /* Author      : MAAM                                                         */
6 /* Version     : v01.2                                                        */
7 /* date        : Mar 25, 2023                                                 */
8 /* ****************************************************************************** */
9 /* ************************ HEADER FILES INCLUDES ************************  */
10 /* ****************************************************************************** */
11
12 #ifndef SEG_INT_H_
13 #define SEG_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 :    7-Seg initialization                                      */
37 /* Input       :    void                                                      */
38 /* Return      :    void                                                      */
39 /* ****************************************************************************** */
40 extern void SEG_vidInit(void);
41
42 /* ****************************************************************************** */
43 /* Description :    7-Seg Display Value                                       */
44 /* Input       :    u16NumValue, u8Dot                                        */
45 /* Return      :    void                                                      */
46 /* ****************************************************************************** */
47 extern void SEG_vidDisplay(u16 u16NumValue, u8 u8Dot);
48
49 /* ****************************************************************************** */
50 /* Description :    7-Seg Display Real Num Value                              */
51 /* Input       :    f32NumValue,                                              */
52 /* Return      :    void                                                      */
53 /* ****************************************************************************** */
54 extern void SEG_vidDisplayFloat(f32 f32NumValue);
55 extern void SEG_vidDisplayFloat_Blink(f32 f32NumValue);
56
57 /* ****************************************************************************** */
58 /* Description :    7-Seg Display Num Value                                   */
59 /* Input       :    u16NumValue                                               */
60 /* Return      :    void                                                      */
61 /* ****************************************************************************** */
62 extern void SEG_vidDisplayNum(u16 u16NumValue);
63 extern void SEG_vidDisplayNum_Blink(u16 u16NumValue);
64
65 /* ****************************************************************************** */
66 /* Description :    7-Seg Display Digit with Dot                              */
67 /* Input       :    u8DigitValue, u8PortCom, u8PinCom, u8Dot                  */
68 /* Return      :    void                                                      */
69 /* ****************************************************************************** */
70 extern void SEG_vidDispalyDigit(u8 u8DigitValue, u8 u8PortCom, u8 u8PinCom, u8 u8Dot);
71
72 /* ****************************************************************************** */
```

```
73 /* Description :    7-Seg Display Update Digit                                */
74 /* Input       :    void                                                      */
75 /* Return      :    LBTY_tenuErrorStatus                                      */
76 /* ************************************************************************ */
77 extern void SEG_vidSetNum(u16 u16NumValue, u8 u8Dot);
78 extern LBTY_tenuErrorStatus SEG_u8Update(void);
79 
80 #endif /* SEG_INT_H_ */
81 /************************** E N D (SEG_int.h) ****************************/
```

SEG_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 "SEG_int.h"
#include "SEG_cfg.h"
#include "SEG_priv.h"
```

Include dependency graph for SEG_prg.c:



Functions

- void SEG_vidInit (void)
- void SEG_vidDisplay (u16 u16NumValue, u8 u8Dot)
- void SEG_vidDisplayFloat (f32 f32NumValue)
- void SEG_vidDisplayFloat_Blink (f32 f32NumValue)
- void SEG_vidDisplayNum (u16 u16NumValue)
- void SEG_vidDisplayNum_Blink (u16 u16NumValue)
- void SEG_vidDispalyDigit (u8 u8DigitValue, u8 u8PortCom, u8 u8PinCom, u8 u8Dot)
- void SEG_vidSetNum (u16 u16NumValue, u8 u8Dot)
- LBTY_tenuErrorStatus SEG_u8Update (void)

Variables

- static u16 u16NumValue_GLB
- static u8 u8Dot_GLB
- const u8 kau8SegPins []
- const u8 kau8SegDigits []

Function Documentation

LBTY_tenuErrorStatus SEG_u8Update (void)

```
247                                             {
248     LBTY_tenuErrorStatus u8RetErrorState = LBTY_OK;
249     static u8 u8Com = LBTY_u8ZERO;
250
251     u16 u16NumValue = u16NumValue_GLB;
252     u8 u8Dot        = u8Dot_GLB;
253
254     switch(u8Com){
255 #ifdef SEG_PIN_COM0
256     case SEG_COM0:
257         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM0, SEG_PIN_COM0,
(u8Dot == SEG_COM0));
258         break;
259 #endif
260
261 #ifdef SEG_PIN_COM1
```

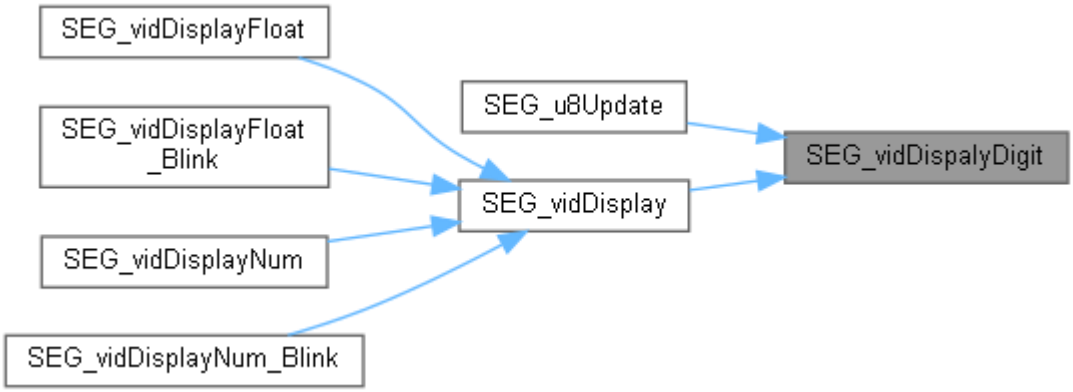
```
262     case SEG_COM1:
263         u16NumValue /= 10u;
264         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM1, SEG_PIN_COM1,
(u8Dot == SEG_COM1));
265         break;
266 #endif
267
268 #ifdef SEG_PIN_COM2
269     case SEG_COM2:
270         u16NumValue /= 10u;
271         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM2, SEG_PIN_COM2,
(u8Dot == SEG_COM2));
272         break;
273 #endif
274
275 #ifdef SEG_PIN_COM3
276     case SEG_COM3:
277         u16NumValue /= 10u;
278         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM3, SEG_PIN_COM3,
(u8Dot == SEG_COM3));
279         break;
280 #endif
281
282 #ifdef SEG_PIN_COM4
283     case SEG_COM4:
284         u16NumValue /= 10u;
285         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM4, SEG_PIN_COM4,
(u8Dot == SEG_COM4));
286         break;
287 #endif
288
289 #ifdef SEG_PIN_COM5
290     case SEG_COM5:
291         u16NumValue /= 10u;
292         SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM5, SEG_PIN_COM5,
(u8Dot == SEG_COM5));
293         break;
294 #endif
295     default:
296         break;
297     }
298     u8Com = (u8Com + 1) % SEG_NUM;
299
300     if(!u8Com){
301         u8RetErrorState = LBTY_NOK;
302     }
303     return u8RetErrorState;
304
305 }
```

Here is the call graph for this function:

void SEG_vidDispalyDigit (u8 *u8DigitValue*, u8 *u8PortCom*, u8 *u8PinCom*, u8 *u8Dot*)

```
220
{
221
222 #ifdef  SEG_DECODER
223     u8 u8PortValue_LOC = 0;
224     GPIO_u8GetPortValue(SEG_PORT_DATA, &u8PortValue_LOC);
225     for(u8 i = SEG_PINS ; i-- ; ){
226         GPIO_u8SetPinValue  (SEG_PORT_DATA, kau8SegDecoderPort[i],
GET_BIT(u8DigitValue, i));
227     }
228 #else
229     GPIO_u8SetPortValue(SEG_PORT_DATA, kau8SegDigits[u8DigitValue] << SEG_a);
230 #endif
231
232     GPIO_u8SetPinValue (SEG_PORT_DOT, SEG_h, u8Dot);
233     GPIO_u8SetPinValue (u8PortCom, u8PinCom, PIN_High);
234     vidMyDelay_ms(SEG_DELAY);
235     GPIO_u8SetPinValue (u8PortCom, u8PinCom, PIN_Low);
236 }
```

Here is the caller graph for this function:

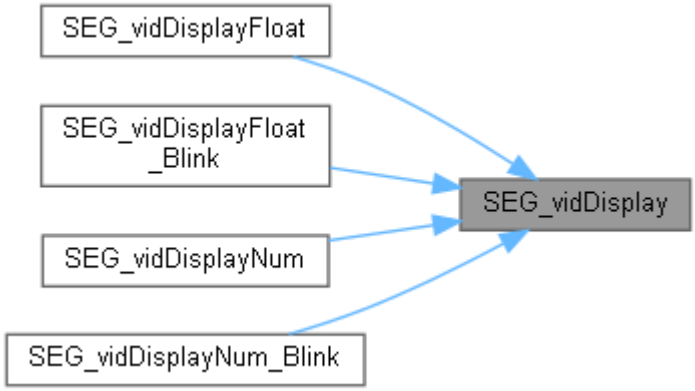


void SEG_vidDisplay (u16 *u16NumValue*, u8 *u8Dot*)

```
127                                                 {
128 #ifdef SEG_PIN_COM0
129     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM0, SEG_PIN_COM0, (u8Dot ==
SEG_COM0));
130 #endif
131
132 #ifdef SEG_PIN_COM1
133     u16NumValue /= 10u;
134     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM1, SEG_PIN_COM1, (u8Dot ==
SEG_COM1));
135 #endif
136
137 #ifdef SEG_PIN_COM2
138     u16NumValue /= 10u;
139     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM2, SEG_PIN_COM2, (u8Dot ==
SEG_COM2));
140 #endif
141
142 #ifdef SEG_PIN_COM3
143     u16NumValue /= 10u;
144     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM3, SEG_PIN_COM3, (u8Dot ==
SEG_COM3));
145 #endif
146
147 #ifdef SEG_PIN_COM4
148     u16NumValue /= 10u;
149     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM4, SEG_PIN_COM4, (u8Dot ==
SEG_COM4));
150 #endif
151
152 #ifdef SEG_PIN_COM5
153     u16NumValue /= 10u;
154     SEG_vidDispalyDigit(u16NumValue % 10u, SEG_PORT_COM5, SEG_PIN_COM5, (u8Dot ==
SEG_COM5));
155 #endif
156
157 }
```

Here is the call graph for this function:

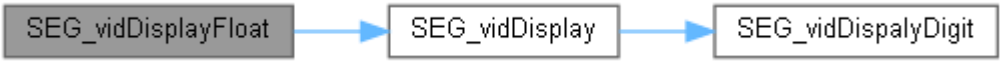
Here is the caller graph for this function:



void SEG_vidDisplayFloat (f32 *f32NumValue*)

```
164                                                 {
165     SEG_vidDisplay((u16)(f32NumValue * SEG_FLOAT_MUL), SEG_FLOAT_DOT);
166 }
```

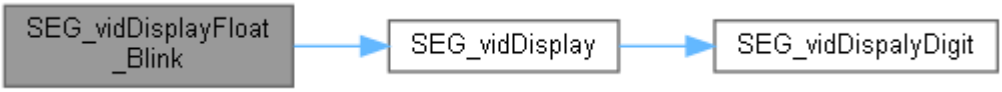
Here is the call graph for this function:



void SEG_vidDisplayFloat_Blink (f32 *f32NumValue*)

```
168                                                         {
169     static u8 u8Tick = LBTY_u8ZERO;
170     static u8 u8Blink = LBTY_SET;
171 
172     if((++u8Tick > SEG_NUM_DELAY)){
173         u8Blink = LBTY_RESET;
174         if(u8Tick >= SEG_NUM_RATE){
175             u8Tick = LBTY_u8ZERO;
176             u8Blink = LBTY_SET;
177         }
178     }
179 
180     if(u8Blink){
181         SEG_vidDisplay((u16)(f32NumValue * SEG_FLOAT_MUL), SEG_FLOAT_DOT);
182     }else{
183         vidMyDelay_ms(SEG_DELAY * SEG_NUM * 2);
184     }
185 }
```

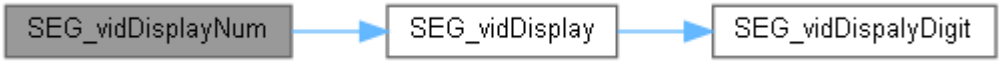
Here is the call graph for this function:



void SEG_vidDisplayNum (u16 *u16NumValue*)

```
192                                                 {
193     SEG_vidDisplay(u16NumValue, LBTY_u8MAX);
194 }
```

Here is the call graph for this function:

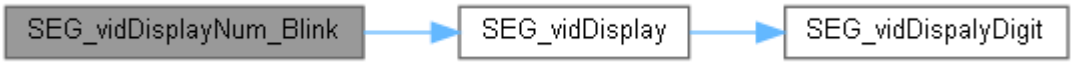


void SEG_vidDisplayNum_Blink (u16 *u16NumValue*)

```
196                                                 {
197     static u8 u8Tick = LBTY_u8ZERO;
198     static u8 u8Blink = LBTY_SET;
199 
200     if((++u8Tick > SEG_NUM_DELAY)){
201         u8Blink = LBTY_RESET;
202         if(u8Tick >= SEG_NUM_RATE){
203             u8Tick = LBTY_u8ZERO;
204             u8Blink = LBTY_SET;
205         }
```

```
206     }
207
208     if(u8Blink){
209         SEG_vidDisplay(u16NumValue, LBTY_u8MAX);
210     }else{
211         vidMyDelay_ms(SEG_DELAY * SEG_NUM * SEG_DELAY);
212     }
213 }
```

Here is the call graph for this function:



void SEG_vidInit (void)

```
59                      {
60
61 #ifdef SEG_PIN_COM0
62     GPIO_u8SetPinDirection(SEG_PORT_COM0, SEG_PIN_COM0, PIN_OUTPUT);
63     GPIO_u8SetPinValue    (SEG_PORT_COM0, SEG_PIN_COM0, PIN_Low);
64 #endif
65
66 #ifdef SEG_PIN_COM1
67     GPIO_u8SetPinDirection(SEG_PORT_COM1, SEG_PIN_COM1, PIN_OUTPUT);
68     GPIO_u8SetPinValue    (SEG_PORT_COM1, SEG_PIN_COM1, PIN_Low);
69 #endif
70
71 #ifdef SEG_PIN_COM2
72     GPIO_u8SetPinDirection(SEG_PORT_COM2, SEG_PIN_COM2, PIN_OUTPUT);
73     GPIO_u8SetPinValue    (SEG_PORT_COM2, SEG_PIN_COM2, PIN_Low);
74 #endif
75
76 #ifdef SEG_PIN_COM3
77     GPIO_u8SetPinDirection(SEG_PORT_COM3, SEG_PIN_COM3, PIN_OUTPUT);
78     GPIO_u8SetPinValue    (SEG_PORT_COM3, SEG_PIN_COM3, PIN_Low);
79 #endif
80
81 #ifdef SEG_PIN_COM4
82     GPIO_u8SetPinDirection(SEG_PORT_COM4, SEG_PIN_COM4, PIN_OUTPUT);
83     GPIO_u8SetPinValue    (SEG_PORT_COM4, SEG_PIN_COM4, PIN_Low);
84 #endif
85
86 #ifdef SEG_PIN_COM5
87     GPIO_u8SetPinDirection(SEG_PORT_COM5, SEG_PIN_COM5, PIN_OUTPUT);
88     GPIO_u8SetPinValue    (SEG_PORT_COM5, SEG_PIN_COM5, PIN_Low);
89 #endif
90
91 #ifdef  SEG_DECODER
92     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_A, PIN_OUTPUT);
93     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_A, PIN_Low);
94     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_B, PIN_OUTPUT);
95     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_B, PIN_Low);
96     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_C, PIN_OUTPUT);
97     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_C, PIN_Low);
98     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_D, PIN_OUTPUT);
99     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_D, PIN_Low);
100 #else
101     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_a, PIN_OUTPUT);
102     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_a, PIN_Low);
103     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_b, PIN_OUTPUT);
104     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_b, PIN_Low);
105     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_c, PIN_OUTPUT);
106     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_c, PIN_Low);
107     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_d, PIN_OUTPUT);
108     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_d, PIN_Low);
109     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_e, PIN_OUTPUT);
110     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_e, PIN_Low);
111     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_f, PIN_OUTPUT);
112     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_f, PIN_Low);
113     GPIO_u8SetPinDirection(SEG_PORT_DATA, SEG_g, PIN_OUTPUT);
114     GPIO_u8SetPinValue    (SEG_PORT_DATA, SEG_g, PIN_Low);
115 #endif
116
117 GPIO_u8SetPinDirection(SEG_PORT_DOT, SEG_h, PIN_OUTPUT);
118 GPIO_u8SetPinValue    (SEG_PORT_DOT, SEG_h, PIN_Low);
119
```

```
120 }
```

void SEG_vidSetNum (u16 *u16NumValue*, u8 *u8Dot*)

```
243                                                 {
244     u16NumValue_GLB = u16NumValue;
245     u8Dot_GLB       = u8Dot;
246 }
```

Variable Documentation

const u8 kau8SegDigits[] `[extern]`

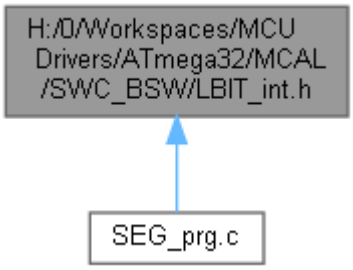
const u8 kau8SegPins[] `[extern]`

u16 u16NumValue_GLB `[static]`

u8 u8Dot_GLB `[static]`

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

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

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

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

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

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

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

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

LBIT_int.h

Go to the documentation of this file.

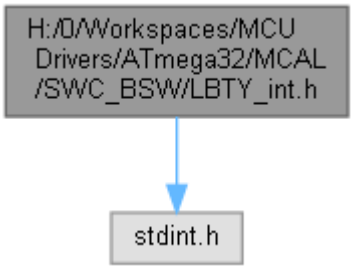
```
1 /*
**************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************** */
4 /* File Name    : LBIT_int.h                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

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

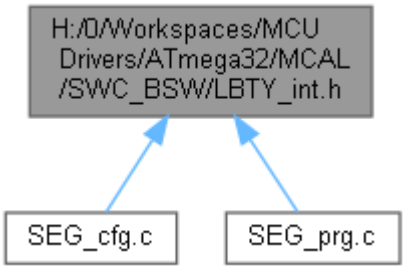
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

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

Include dependency graph for LBTY_int.h:



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



Data Structures

- union LBTY_tuniPort8union LBTY_tuniPort16

Macros

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

Typedefs

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

Enumerations

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

Macro Definition Documentation

#define __I volatile const

#define __IO volatile

#define __O volatile

#define LBTY_NULL ((void *) 0U)

#define LBTY_s16MAX ((u16)0x7FFF)

#define LBTY_s16MIN ((u16)0x8000)

#define LBTY_s32MAX ((u32)0x7FFFFFFFL)

#define LBTY_s32MIN ((u32)0x80000000L)

#define LBTY_s64MAX ((u64)0x7FFFFFFFFFFFFFFFLL)

#define LBTY_s64MIN ((u64)0x8000000000000000LL)

#define LBTY_s8MAX ((s8)0x7F)

#define LBTY_s8MIN ((s8)0x80)

#define LBTY_u16MAX ((u16)0xFFFFU)

#define LBTY_u16ZERO ((u16)0x0000U)

#define LBTY_u32MAX ((u32)0xFFFFFFFFUL)

#define LBTY_u32ZERO ((u32)0x00000000UL)

#define LBTY_u64MAX ((u64)0xFFFFFFFFFFFFFFFFULL)

#define LBTY_u64ZERO ((u64)0x0000000000000000ULL)

#define LBTY_u8MAX ((u8)0xFFU)

#define LBTY_u8vidNOP()

#define LBTY_u8ZERO ((u8)0x00U)

Data Types Limitation

Typedef Documentation

typedef float f32

Standard Real Decimal number

typedef double f64

typedef s16* ps16

typedef s32* ps32

typedef s64* ps64

typedef s8* ps8

Standard Pointer to Signed Byte/Word/Long_Word

typedef u16* pu16

typedef u32* pu32

typedef u64* pu64

typedef u8* pu8

Standard Pointer to Unsigned Byte/Word/Long_Word

typedef int16_t s16

typedef int32_t s32

typedef int64_t s64

typedef int8_t s8

Standard Signed Byte/Word/Long_Word

typedef uint16_t u16

typedef uint32_t u32

typedef uint64_t u64

typedef uint8_t u8

Data Types New Definitions Standard Unsigned Byte/Word/Long_Word

---

Enumeration Type Documentation

enum LBTY_tenuBoolean

Boolean type

Enumerator:

LBTY_TRUE	
LBTY_FALSE	

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

enum LBTY_tenuErrorStatus

Error Return type

Enumerator:

LBTY_OK	
LBTY_NOK	
LBTY_NULL_POINTER	
LBTY_INDEX_OUT_OF_RANGE	
LBTY_NO_MASTER_CHANNEL	
LBTY_READ_ERROR	
LBTY_WRITE_ERROR	
LBTY_UNDEFINED_ERROR	
LBTY_IN_PROGRESS	

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

enum LBTY_tenuFlagStatus

Flag Status type

Enumerator:

LBTY_RESET	
LBTY_SET	

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

LBTY_int.h

Go to the documentation of this file.

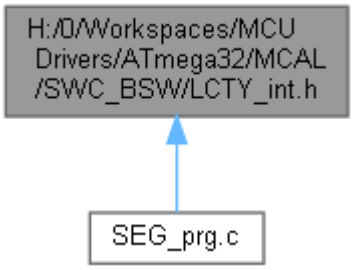
```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************* */
4 /* File Name   : LBTY_int.h                                                  */
5 /* Author      : MAAM                      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

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

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

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

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



Macros

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

Macro Definition Documentation

#define CTY_PACKED __attribute__((__packed__))

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

#define LCTY_CONST __attribute__((__const__))

#define LCTY_DPAGE __attribute__((dp))

#define LCTY_INLINE __attribute__((always_inline)) static inline

#define LCTY_INTERRUPT __attribute__((interrupt))

#define LCTY_NODPAGE __attribute__((nodp))

#define LCTY_PROGMEM __attribute__((__progmem__))

#define LCTY_PURE __attribute__((__pure__))

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

LCTY_int.h

Go to the documentation of this file.

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