SWC_KPAD

Version v1.0
7/17/2023 2:59:00 AM

Table of Contents

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

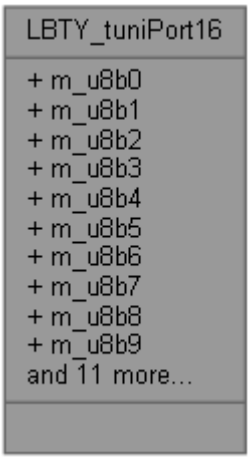
File List

Here is a list of all files with brief descriptions:

Data Structure Documentation

LBTY_tuniPort16 Union Reference

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



Data Fields

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

Field Documentation

u8 m_u8b0

u8 m_u8b1

u8 m_u8b10

u8 m_u8b11

u8 m_u8b12

u8 m_u8b13

u8 m_u8b14

u8 m_u8b15

u8 m_u8b2

u8 m_u8b3

u8 m_u8b4

u8 m_u8b5

u8 m_u8b6

u8 m_u8b7

u8 m_u8b8

u8 m_u8b9

u8 m_u8high

u8 m_u8low

struct { ... } sBits

struct { ... } sBytes

u16 u_u16Word

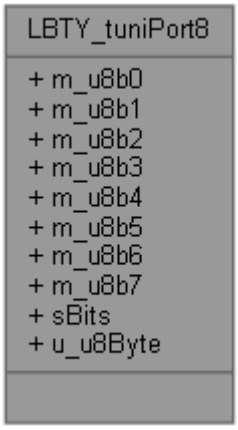
The documentation for this union was generated from the following file:

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

LBTY_tuniPort8 Union Reference

`#include <LBTY_int.h>`

Collaboration diagram for LBTY_tuniPort8:



Data Fields

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

Detailed Description

Union Byte bit by bit

Field Documentation

u8 m_u8b0

u8 m_u8b1

u8 m_u8b2

u8 m_u8b3

u8 m_u8b4

u8 m_u8b5

u8 m_u8b6

u8 m_u8b7

struct { ... } sBits

u8 u_u8Byte

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

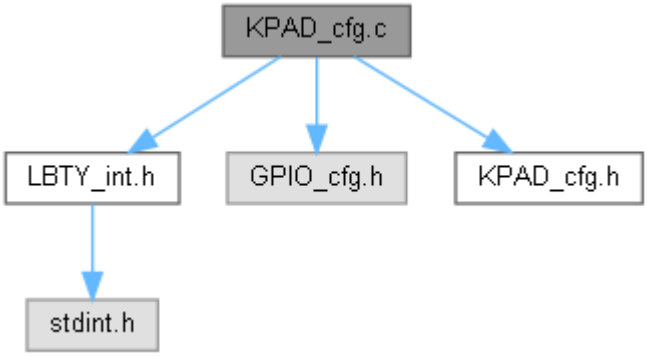
- H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h

File Documentation

KPAD_cfg.c File Reference

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

Include dependency graph for KPAD_cfg.c:



Variables

- const u8 kau8ROW_PINs_GLB [KPAD_ROW_NUM]
- const u8 kau8COL_PINs_GLB [KPAD_COL_NUM]

Variable Documentation

const u8 kau8COL_PINs_GLB[KPAD_COL_NUM]

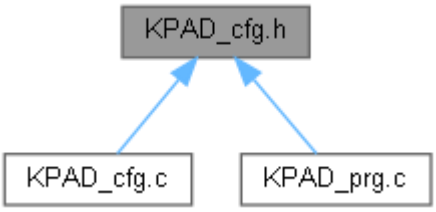
```
Initial value:= {

        KPAD_COL0

        , KPAD_COL1

        , KPAD_COL2

        , KPAD_COL3

        }
```

const u8 kau8ROW_PINs_GLB[KPAD_ROW_NUM]

```
Initial value:= {

        KPAD_ROW0

        , KPAD_ROW1

        , KPAD_ROW2

        , KPAD_ROW3

        }
```

KPAD_cfg.h File Reference

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



Macros

- #define KPAD_ROW_DIR_OUTPUT
- #define KPAD_ROW_NUM 4u
- #define KPAD_COL_NUM 4u
- #define KPAD_ROW_PORT C
- #define KPAD_COL_PORT C
- #define KPAD_ROW0 0u
- #define KPAD_ROW1 1u
- #define KPAD_ROW2 2u
- #define KPAD_ROW3 3u
- #define KPAD_COL0 4u
- #define KPAD_COL1 5u
- #define KPAD_COL2 6u
- #define KPAD_COL3 7u
- #define KPAD_ROW_MASK (1<<KPAD_ROW0)|(1<<KPAD_ROW1)|(1<<KPAD_ROW2)|(1<<KPAD_ROW3)
- #define KPAD_COL_MASK (1<<KPAD_COL0)|(1<<KPAD_COL1)|(1<<KPAD_COL2)|(1<<KPAD_COL3)
- #define DEBOUNCING_CYCLES_NUM 2u
- #define DEBOUNCING_DELAY 2u
- #define KPAD_MAX_COL 4u
- #define KPAD_MAX_ROW 4u
- #define KPAD_KEY00 '1'
- #define KPAD_KEY01 '2'
- #define KPAD_KEY02 '3'
- #define KPAD_KEY03 'A'
- #define KPAD_KEY10 '4'
- #define KPAD_KEY11 '5'
- #define KPAD_KEY12 '6'
- #define KPAD_KEY13 'B'
- #define KPAD_KEY20 '7'
- #define KPAD_KEY21 '8'
- #define KPAD_KEY22 '9'
- #define KPAD_KEY23 'C'
- #define KPAD_KEY30 '*'
- #define KPAD_KEY31 '0'
- #define KPAD_KEY32 '#'
- #define KPAD_KEY33 'D'
- #define KPAD_RELEASE '\0'

Macro Definition Documentation

#define DEBOUNCING_CYCLES_NUM 2u

#define DEBOUNCING_DELAY 2u

#define KPAD_COL0 4u

#define KPAD_COL1 5u

#define KPAD_COL2 6u

#define KPAD_COL3 7u

#define KPAD_COL_MASK (1<<KPAD_COL0)|(1<<KPAD_COL1)|(1<<KPAD_COL2)|(1<<KPAD_COL3)

#define KPAD_COL_NUM 4u

#define KPAD_COL_PORT C

#define KPAD_KEY00 '1'

Row 0

#define KPAD_KEY01 '2'

#define KPAD_KEY02 '3'

#define KPAD_KEY03 'A'

#define KPAD_KEY10 '4'

Row 1

#define KPAD_KEY11 '5'

#define KPAD_KEY12 '6'

#define KPAD_KEY13 'B'

#define KPAD_KEY20 '7'

Row 2

#define KPAD_KEY21 '8'

#define KPAD_KEY22 '9'

#define KPAD_KEY23 'C'

#define KPAD_KEY30 '*'

Row 3

```
#define KPAD_KEY31   '0'

#define KPAD_KEY32   '#'

#define KPAD_KEY33   'D'

#define KPAD_MAX_COL   4u

#define KPAD_MAX_ROW   4u

#define KPAD_RELEASE   '\0'

#define KPAD_ROW0   0u

#define KPAD_ROW1   1u

#define KPAD_ROW2   2u

#define KPAD_ROW3   3u

#define KPAD_ROW_DIR_OUTPUT

#define
KPAD_ROW_MASK   (1<<KPAD_ROW0)|(1<<KPAD_ROW1)|(1<<KPAD_ROW2)|(1<<KPAD_ROW3)

#define KPAD_ROW_NUM   4u

#define KPAD_ROW_PORT   C
```

KPAD_cfg.h

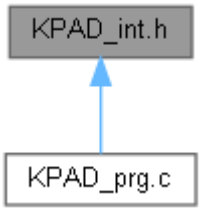
```
Go to the documentation of this file.1 /*
****************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ****************************************************************************** */
4 /* File Name   : KEYPAD_cfg.h                                                 */
5 /* Author      : MAAM                                                         */
6 /* Version     : v01.2                                                        */
7 /* date        : Mar 25, 2023                                                 */
8 /* ****************************************************************************** */
9 /* ************************ HEADER FILES INCLUDES ***********************  */
10 /* ****************************************************************************** */
11
12 #ifndef KPAD_CFG_H_
13 #define KPAD_CFG_H_
14
15 /* ****************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ****************************************************************************** */
18
19 /* ****************************************************************************** */
20 /* ************************** MACRO/DEFINE SECTION ************************** */
21 /* ****************************************************************************** */
22 /*
23 |      |      |      |
24 #      #      #      # PULL UP
25 |      |      |      |
26 @ --- @ --- @ --- @
27 |      |      |      |---------- ROW0  }
28 @ --- @ --- @ --- @                  }
29 |      |      |      |---------- ROW1  }
30 @ --- @ --- @ --- @                  }<== OUTPUT
31 |      |      |      |---------- ROW2  }
32 @ --- @ --- @ --- @                  }
33 |      |      |      |---------- ROW3  }
34 |      |      |      |
35 COL0   COL1   COL2   COL3               <== INPUT
36 */
37
38 #if defined(AMIT_KIT)
39
40 #define KPAD_MUX_TYPE           KPAD_COL_MUX
41 #define KPAD_MUX_ACTIVE         KPAD_ACTIVE_HIGH
42
43 #define KPAD_ROW_NUM            3u
44 #define KPAD_ROW_PORT           D
45 #define KPAD_ROW0               0u
46 #define KPAD_ROW1               1u
47 #define KPAD_ROW2               2u
48
49 #define KPAD_COL_NUM            3u
50 #define KPAD_COL_PORT           B
51 #define KPAD_COL0               5u
52 #define KPAD_COL1               6u
53 #define KPAD_COL2               7u
54
55 #define KPAD_ROW_MASK           (1<<KPAD_ROW0)|(1<<KPAD_ROW1)|(1<<KPAD_ROW2)
56 #define KPAD_COL_MASK           (1<<KPAD_COL0)|(1<<KPAD_COL1)|(1<<KPAD_COL2)
57
58 #elif defined(ETA32_KIT)
59
60 #define KPAD_MUX_TYPE           KPAD_ROW_MUX
61 #define KPAD_MUX_ACTIVE         KPAD_ACTIVE_LOW
62
63 #define KPAD_ROW_NUM            4u
64 #define KPAD_ROW_PORT           C
65 #define KPAD_ROW0               Eta32_Keypad_Row0
66 #define KPAD_ROW1               Eta32_Keypad_Row1
67 #define KPAD_ROW2               Eta32_Keypad_Row2
68 #define KPAD_ROW3               Eta32_Keypad_Row3
69
70 #define KPAD_COL_NUM            4u
71 #define KPAD_COL_PORT           D
72 #define KPAD_COL0               Eta32_Keypad_col0
```

```
73 #define KPAD_COL1               Eta32_Keypad_col1
74 #define KPAD_COL2               Eta32_Keypad_col2
75 #define KPAD_COL3               Eta32_Keypad_col3
76
77 #define KPAD_ROW_MASK
(1<<KPAD_ROW0)|(1<<KPAD_ROW1)|(1<<KPAD_ROW2)|(1<<KPAD_ROW3)
78 #define KPAD_COL_MASK
(1<<KPAD_COL0)|(1<<KPAD_COL1)|(1<<KPAD_COL2)|(1<<KPAD_COL3)
79
80 #elif defined(ETA32_MINI_KIT)
81
82 #define KPAD_MUX_TYPE           KPAD_ROW_MUX
83 #define KPAD_MUX_ACTIVE         KPAD_ACTIVE_LOW
84
85 #define KPAD_ROW_NUM            4u
86 #define KPAD_ROW_PORT           B
87 #define KPAD_ROW0               Eta32_mini_Keypad_Row0
88 #define KPAD_ROW1               Eta32_mini_Keypad_Row1
89 #define KPAD_ROW2               Eta32_mini_Keypad_Row2
90 #define KPAD_ROW3               Eta32_mini_Keypad_Row3
91
92 #define KPAD_COL_NUM            4u
93 #define KPAD_COL_PORT           D
94 #define KPAD_COL0               Eta32_mini_Keypad_col0
95 #define KPAD_COL1               Eta32_mini_Keypad_col1
96 #define KPAD_COL2               Eta32_mini_Keypad_col2
97 #define KPAD_COL3               Eta32_mini_Keypad_col3
98
99 #define KPAD_ROW_MASK
(1<<KPAD_ROW0)|(1<<KPAD_ROW1)|(1<<KPAD_ROW2)|(1<<KPAD_ROW3)
100 #define KPAD_COL_MASK
(1<<KPAD_COL0)|(1<<KPAD_COL1)|(1<<KPAD_COL2)|(1<<KPAD_COL3)
101
102 #else
103
104 #define KPAD_ROW_DIR_OUTPUT
105 //#define KPAD_COL_DIR_OUTPUT
106
107 #define KPAD_ROW_NUM            4u
108 #define KPAD_COL_NUM            4u
109
110 #define KPAD_ROW_PORT           C
111 #define KPAD_COL_PORT           C
112
113 #define KPAD_ROW0               0u
114 #define KPAD_ROW1               1u
115 #define KPAD_ROW2               2u
116 #define KPAD_ROW3               3u
117
118 #define KPAD_COL0               4u
119 #define KPAD_COL1               5u
120 #define KPAD_COL2               6u
121 #define KPAD_COL3               7u
122
123 #define KPAD_ROW_MASK
(1<<KPAD_ROW0)|(1<<KPAD_ROW1)|(1<<KPAD_ROW2)|(1<<KPAD_ROW3)
124 #define KPAD_COL_MASK
(1<<KPAD_COL0)|(1<<KPAD_COL1)|(1<<KPAD_COL2)|(1<<KPAD_COL3)
125
126 #endif
127
128 #define DEBOUNCING_CYCLES_NUM   2u
129 #define DEBOUNCING_DELAY        2u
130
131 #define KPAD_MAX_COL            4u
132 #define KPAD_MAX_ROW            4u
133
135 #define KPAD_KEY00              '1'
136 #define KPAD_KEY01              '2'
137 #define KPAD_KEY02              '3'
138 #define KPAD_KEY03              'A'
139
141 #define KPAD_KEY10              '4'
142 #define KPAD_KEY11              '5'
143 #define KPAD_KEY12              '6'
144 #define KPAD_KEY13              'B'
145
```

```
147 #define KPAD_KEY20              '7'
148 #define KPAD_KEY21              '8'
149 #define KPAD_KEY22              '9'
150 #define KPAD_KEY23              'C'
151
153 #define KPAD_KEY30              '*'
154 #define KPAD_KEY31              '0'
155 #define KPAD_KEY32              '#'
156 #define KPAD_KEY33              'D'
157
158 #define KPAD_RELEASE            '\0'
159
160 /* ********************************************************************** */
161 /* ***************************** CONST SECTION ***************************** */
162 /* ********************************************************************** */
163
164 /* ********************************************************************** */
165 /* ***************************** VARIABLE SECTION ************************** */
166 /* ********************************************************************** */
167
168 /* ********************************************************************** */
169 /* ***************************** FUNCTION SECTION ************************** */
170 /* ********************************************************************** */
171
172
173 #endif /* KPAD_CFG_H_ */
174 /*************************** E N D (KEYPAD_cfg.h) ****************************/
```

KPAD_int.h File Reference

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



Functions

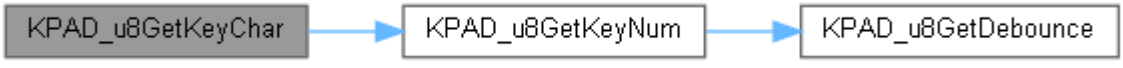
- void KPAD_vidInit (void)
- u8 KPAD_u8GetKeyNum (void)
- u8 KPAD_u8GetKeyChar (void)

Function Documentation

u8 KPAD_u8GetKeyChar (void)

```
143                                 {
144     u8 u8RetValue = LBTY_u8ZERO;
145
146     switch(KPAD_u8GetKeyNum()){
147         case  0:            u8RetValue = KPAD_KEY00;            break;
148         case  1:            u8RetValue = KPAD_KEY01;            break;
149         case  2:            u8RetValue = KPAD_KEY02;            break;
150         case  3:            u8RetValue = KPAD_KEY03;            break;
151
152         case  4:            u8RetValue = KPAD_KEY10;            break;
153         case  5:            u8RetValue = KPAD_KEY11;            break;
154         case  6:            u8RetValue = KPAD_KEY12;            break;
155         case  7:            u8RetValue = KPAD_KEY13;            break;
156
157         case  8:            u8RetValue = KPAD_KEY20;            break;
158         case  9:            u8RetValue = KPAD_KEY21;            break;
159         case 10:            u8RetValue = KPAD_KEY22;            break;
160         case 11:            u8RetValue = KPAD_KEY23;            break;
161
162         case 12:            u8RetValue = KPAD_KEY30;            break;
163         case 13:            u8RetValue = KPAD_KEY31;            break;
164         case 14:            u8RetValue = KPAD_KEY32;            break;
165         case 15:            u8RetValue = KPAD_KEY33;            break;
166
167         default:            u8RetValue = KPAD_RELEASE;          break;
168     }
169
170     return u8RetValue;
171 }
```

Here is the call graph for this function:



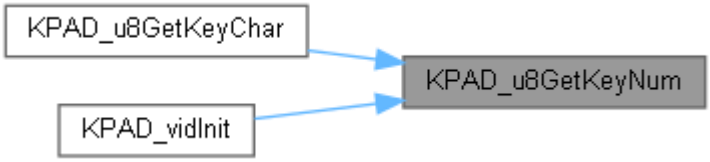
u8 KPAD_u8GetKeyNum (void)

```
96                              {
97     u8 u8RetValue = LBTY_u8MAX;
98
99 #if (KPAD_MUX_TYPE == KPAD_ROW_MUX)
100     for(u8 i = 0 ; i<KPAD_ROW_NUM ; i++){
101         GPIO_u8SetPinValue(KPAD_ROW_PORT, kau8ROW_PINs_GLB[i],
KPAD_MUX_ACTIVE);
102         for(u8 j = 0 ; j<KPAD_COL_NUM ; j++){
103             KPAD_u8GetDebounce(KPAD_COL_PORT, kau8COL_PINs_GLB[j],
&u8RetValue);
```

```
104             if(u8RetValue != KPAD_MUX_ACTIVE){
105                 u8RetValue = LBTY_u8MAX;
106                 continue;
107             }else{
108                 u8RetValue = j + (i * KPAD_MAX_ROW);
109                 break;
110             }
111         }
112         GPIO_u8SetPinValue(KPAD_ROW_PORT, kau8ROW_PINs_GLB[i],
!KPAD_MUX_ACTIVE);
113 #elif (KPAD_MUX_TYPE == KPAD_COL_MUX)
114     for(u8 i = 0 ; i<KPAD_COL_NUM ; i++){
115         GPIO_u8SetPinValue(KPAD_COL_PORT, kau8COL_PINs_GLB[i],
KPAD_MUX_ACTIVE);
116         for(u8 j = 0 ; j<KPAD_ROW_NUM ; j++){
117             KPAD_u8GetDebounce(KPAD_ROW_PORT, kau8ROW_PINs_GLB[j],
&u8RetValue);
118
119             if(u8RetValue != KPAD_MUX_ACTIVE){
120                 u8RetValue = LBTY_u8MAX;
121                 continue;
122             }else{
123                 u8RetValue = j + (i * KPAD_MAX_COL);
124                 break;
125             }
126         }
127         GPIO_u8SetPinValue(KPAD_COL_PORT, kau8COL_PINs_GLB[i],
!KPAD_MUX_ACTIVE);
128 #endif
129         if(u8RetValue == LBTY_u8MAX){
130             continue;
131         }else{
132             break;
133         }
134     }
```

Here is the call graph for this function:

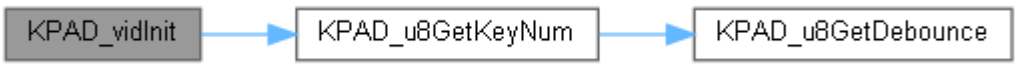
Here is the caller graph for this function:



void KPAD_vidInit (void)

```
51                          {
52 #if (KPAD_MUX_TYPE == KPAD_ROW_MUX)
53     GPIO_u8SetMaskDirection (KPAD_ROW_PORT, KPAD_ROW_MASK, PORT_OUTPUT);
54     GPIO_u8SetMaskDirection (KPAD_COL_PORT, KPAD_COL_MASK, PORT_INPUT);
55 #elif (KPAD_MUX_TYPE == KPAD_COL_MUX)
56     GPIO_u8SetMaskDirection (KPAD_COL_PORT, KPAD_COL_MASK, PORT_OUTPUT);
57     GPIO_u8SetMaskDirection (KPAD_ROW_PORT, KPAD_ROW_MASK, PORT_INPUT);
58 #endif
59     KPAD_u8GetKeyNum();
60 }
```

Here is the call graph for this function:



KPAD_int.h

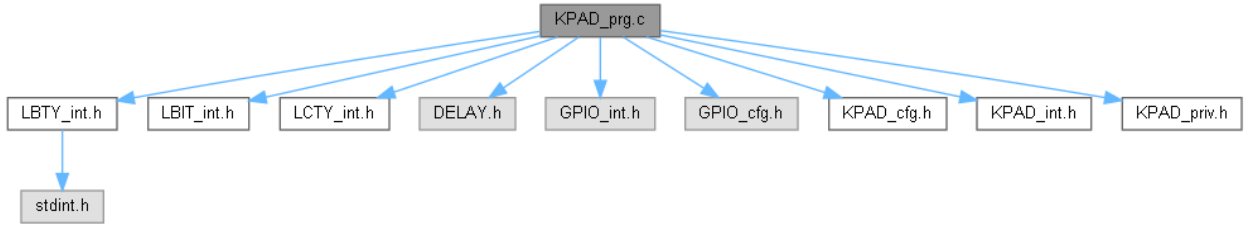
Go to the documentation of this file.

```c
/*
/* ********************************************************************************** */
/* ********************** FILE DEFINITION SECTION ************************************ */
/* ********************************************************************************** */
/* File Name    : KEYPAD_int.h                                                        */
/* Author       : MAAM                                                                */
/* Version      : v01.2                                                               */
/* date         : Mar 25, 2023                                                        */
/* ********************************************************************************** */
/* ************************ HEADER FILES INCLUDES ***************************  */
/* ********************************************************************************** */

#ifndef KPAD_INT_H_
#define KPAD_INT_H_

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

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

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

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

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

/* ********************************************************************************** */
/* Description :    Keypad initialization                                             */
/* Input       :    void                                                              */
/* Return      :    void                                                              */
/* ********************************************************************************** */
extern void KPAD_vidInit(void);

/* ********************************************************************************** */
/* Description :    Keypad Get Push Press Number                                      */
/* Input       :    void                                                              */
/* Return      :    u8                                                                */
/* ********************************************************************************** */
extern u8 KPAD_u8GetKeyNum(void);

/* ********************************************************************************** */
/* Description :    Keypad Get Input Char Value                                       */
/* Input       :    void                                                              */
/* Return      :    u8                                                                */
/* ********************************************************************************** */
extern u8 KPAD_u8GetKeyChar(void);

#endif /* KPAD_INT_H_ */
/***************************** E N D (KEYPAD_int.h) ******************************/
```

KPAD_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 "KPAD_cfg.h"
#include "KPAD_int.h"
#include "KPAD_priv.h"
```

Include dependency graph for KPAD_prg.c:



Functions

- void KPAD_vidInit (void)
- static LBTY_tenuErrorStatus KPAD_u8GetDebounce (GPIO_tenuPortNum u8PortNum, GPIO_tenuPinNum u8PinNum, pu8 pu8Value)
- u8 KPAD_u8GetKeyNum (void)
- u8 KPAD_u8GetKeyChar (void)

Variables

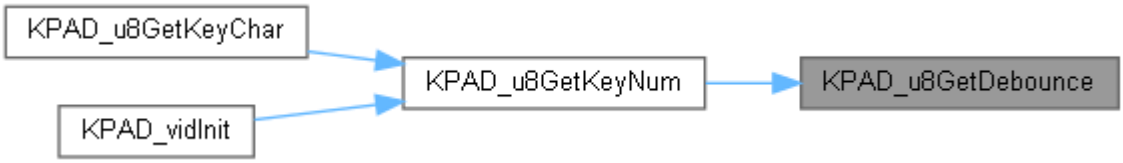
- const u8 kau8ROW_PINs_GLB [KPAD_ROW_NUM]
- const u8 kau8COL_PINs_GLB [KPAD_COL_NUM]
- return u8RetValue

Function Documentation

static LBTY_tenuErrorStatus KPAD_u8GetDebounce (GPIO_tenuPortNum *u8PortNum*, GPIO_tenuPinNum *u8PinNum*, pu8 *pu8Value*) [static]

```
69                                                  {
70      u8 u8PreValue = LBTY_u8ZERO;
71      u8 u8CurValue = LBTY_u8ZERO;
72      u8 u8DebouncingCount = LBTY_u8ZERO;
73
74      LBTY_tenuErrorStatus u8RetValue = GPIO_u8GetPinValue(u8PortNum, u8PinNum,
&u8PreValue);
75
76      while((u8DebouncingCount < DEBOUNCING_CYCLES_NUM) && (u8RetValue == LBTY_OK)){
77          vidMyDelay_ms(DEBOUNCING_DELAY);
78          u8RetValue  = GPIO_u8GetPinValue(u8PortNum, u8PinNum, &u8CurValue);
79
80          if(u8PreValue == u8CurValue){
81              u8DebouncingCount++;
82          }else{
83              u8DebouncingCount = 0;
84          }
85          u8PreValue = u8CurValue;
86      }
87      *pu8Value = u8CurValue;
88      return u8RetValue;
89  }
```

Here is the caller graph for this function:



u8 KPAD_u8GetKeyChar (void)

```
143                             {
144     u8 u8RetValue = LBTY_u8ZERO;
145 
146     switch(KPAD_u8GetKeyNum()){
147         case  0:            u8RetValue = KPAD_KEY00;            break;
148         case  1:            u8RetValue = KPAD_KEY01;            break;
149         case  2:            u8RetValue = KPAD_KEY02;            break;
150         case  3:            u8RetValue = KPAD_KEY03;            break;
151 
152         case  4:            u8RetValue = KPAD_KEY10;            break;
153         case  5:            u8RetValue = KPAD_KEY11;            break;
154         case  6:            u8RetValue = KPAD_KEY12;            break;
155         case  7:            u8RetValue = KPAD_KEY13;            break;
156 
157         case  8:            u8RetValue = KPAD_KEY20;            break;
158         case  9:            u8RetValue = KPAD_KEY21;            break;
159         case 10:            u8RetValue = KPAD_KEY22;            break;
160         case 11:            u8RetValue = KPAD_KEY23;            break;
161 
162         case 12:            u8RetValue = KPAD_KEY30;            break;
163         case 13:            u8RetValue = KPAD_KEY31;            break;
164         case 14:            u8RetValue = KPAD_KEY32;            break;
165         case 15:            u8RetValue = KPAD_KEY33;            break;
166 
167         default:            u8RetValue = KPAD_RELEASE;          break;
168     }
169 
170     return u8RetValue;
171 }
```

Here is the call graph for this function:

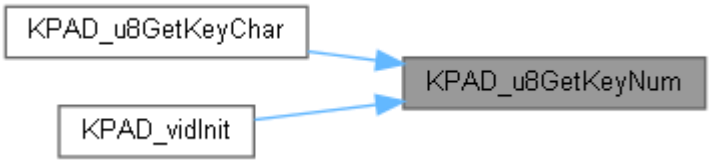
u8 KPAD_u8GetKeyNum (void)

```
96                              {
97     u8 u8RetValue = LBTY_u8MAX;
98 
99 #if (KPAD_MUX_TYPE == KPAD_ROW_MUX)
100     for(u8 i = 0 ; i<KPAD_ROW_NUM ; i++){
101         GPIO_u8SetPinValue(KPAD_ROW_PORT, kau8ROW_PINs_GLB[i],
KPAD_MUX_ACTIVE);
102         for(u8 j = 0 ; j<KPAD_COL_NUM ; j++){
103             KPAD_u8GetDebounce(KPAD_COL_PORT, kau8COL_PINs_GLB[j],
&u8RetValue);
104             if(u8RetValue != KPAD_MUX_ACTIVE){
105                 u8RetValue = LBTY_u8MAX;
106                 continue;
107             }else{
108                 u8RetValue = j + (i * KPAD_MAX_ROW);
109                 break;
110             }
111         }
112         GPIO_u8SetPinValue(KPAD_ROW_PORT, kau8ROW_PINs_GLB[i],
!KPAD_MUX_ACTIVE);
113 #elif (KPAD_MUX_TYPE == KPAD_COL_MUX)
114     for(u8 i = 0 ; i<KPAD_COL_NUM ; i++){
115         GPIO_u8SetPinValue(KPAD_COL_PORT, kau8COL_PINs_GLB[i],
KPAD_MUX_ACTIVE);
116         for(u8 j = 0 ; j<KPAD_ROW_NUM ; j++){
117             KPAD_u8GetDebounce(KPAD_ROW_PORT, kau8ROW_PINs_GLB[j],
&u8RetValue);
118 
119             if(u8RetValue != KPAD_MUX_ACTIVE){
```

```
120                 u8RetValue = LBTY_u8MAX;
121                 continue;
122             }else{
123                 u8RetValue = j + (i * KPAD_MAX_COL);
124                 break;
125             }
126         }
127         GPIO_u8SetPinValue(KPAD_COL_PORT, kau8COL_PINs_GLB[i],
!KPAD_MUX_ACTIVE);
128 #endif
129         if(u8RetValue == LBTY_u8MAX){
130             continue;
131         }else{
132             break;
133         }
134     }
```

Here is the call graph for this function:

Here is the caller graph for this function:



void KPAD_vidInit (void)

```
51                          {
52 #if (KPAD_MUX_TYPE == KPAD_ROW_MUX)
53     GPIO_u8SetMaskDirection (KPAD_ROW_PORT, KPAD_ROW_MASK, PORT_OUTPUT);
54     GPIO_u8SetMaskDirection (KPAD_COL_PORT, KPAD_COL_MASK, PORT_INPUT);
55 #elif (KPAD_MUX_TYPE == KPAD_COL_MUX)
56     GPIO_u8SetMaskDirection (KPAD_COL_PORT, KPAD_COL_MASK, PORT_OUTPUT);
57     GPIO_u8SetMaskDirection (KPAD_ROW_PORT, KPAD_ROW_MASK, PORT_INPUT);
58 #endif
59     KPAD_u8GetKeyNum();
60 }
```

Here is the call graph for this function:

Variable Documentation

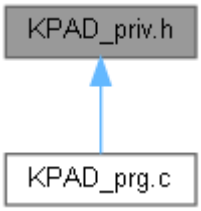
const u8 kau8COL_PINs_GLB[KPAD_COL_NUM] [extern]

const u8 kau8ROW_PINs_GLB[KPAD_ROW_NUM] [extern]

return u8RetValue

KPAD_priv.h File Reference

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



Macros

- #define KPAD_ACTIVE_HIGH PIN_High
- #define KPAD_ACTIVE_LOW PIN_Low
- #define KPAD_ROW_MUX 1u
- #define KPAD_COL_MUX 2u

Macro Definition Documentation

#define KPAD_ACTIVE_HIGH PIN_High

#define KPAD_ACTIVE_LOW PIN_Low

#define KPAD_COL_MUX 2u

#define KPAD_ROW_MUX 1u

KPAD_priv.h

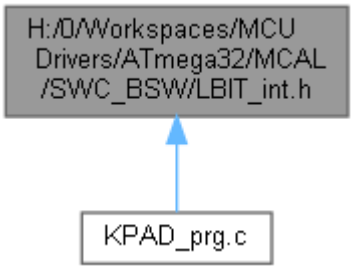
Go to the documentation of this file.

```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************** */
4 /* File Name    : KEYPAD_priv.h                                              */
5 /* Author       : MAAM                                                       */
6 /* Version      : v01.2                                                      */
7 /* date         : Mar 25, 2023                                               */
8 /* ************************************************************************** */
9 /* ************************* HEADER FILES INCLUDES ************************  */
10 /* ************************************************************************** */
11
12 #ifndef KPAD_PRIV_H_
13 #define KPAD_PRIV_H_
14
15 /* ************************************************************************** */
16 /* ********************** TYPE_DEF/STRUCT/ENUM SECTION ********************** */
17 /* ************************************************************************** */
18
19 /* ************************************************************************** */
20 /* ************************** MACRO/DEFINE SECTION ************************** */
21 /* ************************************************************************** */
22
23 #define KPAD_ACTIVE_HIGH        PIN_High
24 #define KPAD_ACTIVE_LOW         PIN_Low
25
26 #define KPAD_ROW_MUX            1u
27 #define KPAD_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 /* KPAD_PRIV_H_ */
43 /***************************** E N D (KEYPAD_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

```
#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.

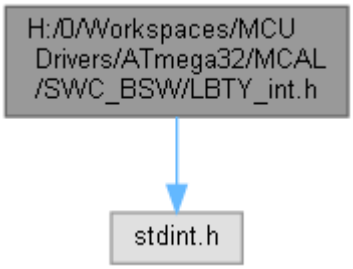
```
1 /*
****************************************************************************** */
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ****************************************************************************** */
4 /* File Name   : LBIT_int.h           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

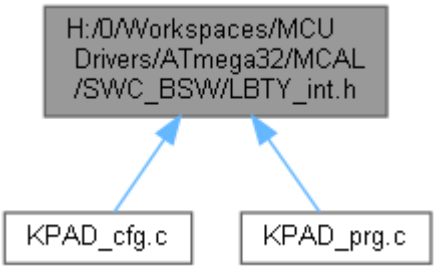
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

```
#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.

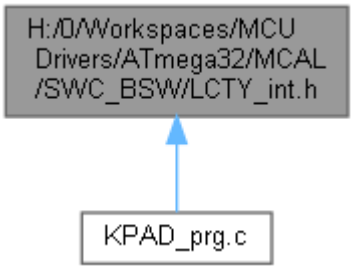
```
1 /*
2 /* ********************** FILE DEFINITION SECTION ************************** */
3 /* ************************************************************************* */
4 /* File Name   : LBTY_int.h                                                  */
5 /* Author      : MAAM                      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