# SWC_INT

Version v1.0

7/16/2023 12:16:00 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

## GICR_type Union Reference

: Type define of Union bit field of "General INT Control Register"

`#include <INT_priv.h>`

Collaboration diagram for GICR_type:

### Data Fields

- u8 u_Reg
- struct {
- __IO u8: 5
- __IO u8 m_INT2E: 1
- __IO u8 m_INT0E: 1
- __IO u8 m_INT1E: 1
- } sBits

### Detailed Description

: Type define of Union bit field of "General INT Control Register"

**Type** : Union **Unit** : None

### Field Documentation

**__IO u8 m_INT0E**

External Interrupt Request Enable 0

**__IO u8 m_INT1E**

External Interrupt Request Enable 1

**__IO u8 m_INT2E**

External Interrupt Request Enable 2

**struct { ... } sBits**

**__IO u8**

Reversed

**u8 u_Reg**

---

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

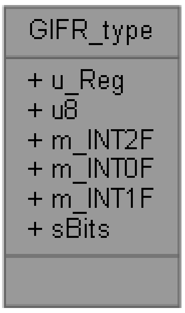
**INT_priv.h**

## GIFR_type Union Reference

: Type define of Union bit field of "General INT Flag Register"

`#include <INT_priv.h>`

Collaboration diagram for GIFR_type:



### Data Fields

- u8 u_Reg
- struct {
- __IO u8: 5
- __IO u8 m_INT2F: 1
- __IO u8 m_INT0F: 1
- __IO u8 m_INT1F: 1
- } sBits

---

### Detailed Description

: Type define of Union bit field of "General INT Flag Register"

**Type** : Union **Unit** : None

---

### Field Documentation

**__IO u8 m_INT0F**

External Interrupt Request Flag 0

**__IO u8 m_INT1F**

External Interrupt Request Flag 1

**__IO u8 m_INT2F**

External Interrupt Request Flag 2

**struct { ... } sBits**

**__IO u8**

Reversed

**u8 u_Reg**

---

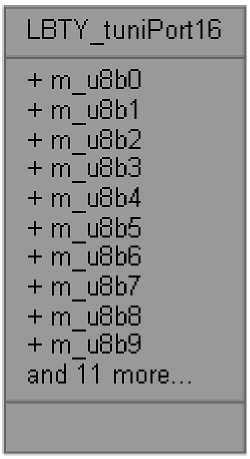
**The documentation for this union was generated from the following file:**

**INT_priv.h**

## 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**

---

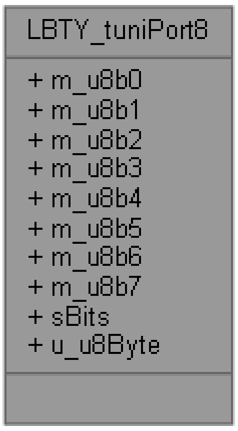
**The documentation for this union was generated from the following file:**

- 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:**

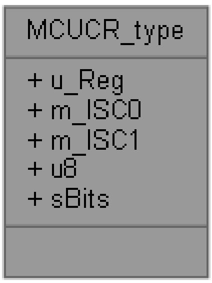
- H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h

# MCUCR_type Union Reference

: Type define of Union bit field of "MCU Control Register"

`#include <INT_priv.h>`

Collaboration diagram for MCUCR_type:



## Data Fields

- u8 u_Reg
- struct {
- __IO u8 m_ISC0: 2
- __IO u8 m_ISC1: 2
- __IO u8: 4
- } sBits

---

## Detailed Description

: Type define of Union bit field of "MCU Control Register"

**Type** : Union **Unit** : None

---

## Field Documentation

**__IO u8 m_ISC0**

Interrupt 0 Sense Control

**__IO u8 m_ISC1**

Interrupt 1 Sense Control

**struct { ... } sBits**

**__IO u8**

Reversed

**u8 u_Reg**

---

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

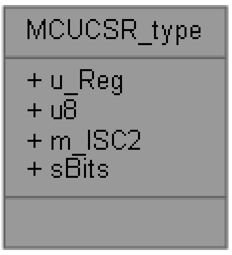
**INT_priv.h**

## MCUCSR_type Union Reference

: Type define of Union bit field of "Control and Status Register"

```
#include <INT_priv.h>
```

Collaboration diagram for MCUCSR_type:



### Data Fields

- u8 u_Reg
- struct {
- __IO u8: 6
- __IO u8 m_ISC2: 1
- } sBits

### Detailed Description

: Type define of Union bit field of "Control and Status Register"

**Type** : Union **Unit** : None

### Field Documentation

**__IO u8 m_ISC2**

Interrupt 2 Sense Control

**struct { ... } sBits**

**__IO u8**

Reversed

**u8 u_Reg**

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

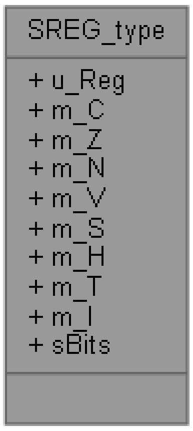
**INT_priv.h**

# SREG_type Union Reference

: Type define of Union bit field of "General INT Control Register"

`#include <INTP.h>`

Collaboration diagram for SREG_type:



## Data Fields

- u8 u_Reg
- struct {
- __IO u8 m_C: 1
- __IO u8 m_Z: 1
- __IO u8 m_N: 1
- __IO u8 m_V: 1
- __IO u8 m_S: 1
- __IO u8 m_H: 1
- __IO u8 m_T: 1
- __IO u8 m_I: 1
- } sBits

---

## Detailed Description

: Type define of Union bit field of "General INT Control Register"

**Type** : Union **Unit** : None

---

## Field Documentation

### __IO u8 m_C

Carry Flag

### __IO u8 m_H

Half Carry Flag

### __IO u8 m_I

Global Interrupt Enable

**__IO u8 m_N**

Negative Flag

**__IO u8 m_S**

Sign Bit

**__IO u8 m_T**

Bit Copy Storage

**__IO u8 m_V**

Two’s Complement Overflow Flag

**__IO u8 m_Z**

Zero Flag

**struct { ... } sBits**

**u8 u_Reg**

---

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

- H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_MCU/INTP.h

# File Documentation

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

### Macros

- #define _BV(bit)   (1u<<(bit))
- #define SET_BIT(REG,   bit)   ((REG) |=   (1u<<(bit)))
- #define CLR_BIT(REG,   bit)   ((REG) &= ~(1u<<(bit)))
- #define TOG_BIT(REG,   bit)   ((REG) ^=   (1u<<(bit)))
- #define SET_BYTE(REG,   bit)   ((REG) |=   (0xFFu<<(bit)))
- #define CLR_BYTE(REG,   bit)   ((REG) &= ~(0xFFu<<(bit)))
- #define TOG_BYTE(REG,   bit)   ((REG) ^=   (0xFFu<<(bit)))
- #define SET_MASK(REG,   MASK)   ((REG) |=   (MASK))
- #define CLR_MASK(REG,   MASK)   ((REG) &= ~(MASK))
- #define TOG_MASK(REG,   MASK)   ((REG) ^=   (MASK))
- #define GET_MASK(REG,   MASK)   ((REG) &    (MASK))
- #define SET_REG(REG)   ((REG)   = ~(0u))
- #define CLR_REG(REG)   ((REG)   =   (0u))
- #define TOG_REG(REG)   ((REG) ^= ~(0u))
- #define GET_BIT(REG,   bit)   (((REG)>>(bit)) & 0x01u)
- #define GET_NIB(REG,   bit)   (((REG)>>(bit)) & 0x0Fu)
- #define GET_BYTE(REG,   bit)   (((REG)>>(bit)) & 0xFFu)
- #define ASSIGN_BIT(REG,   bit,   value)   ((REG) = ((REG) & ~(0x01u<<(bit)))   | (((value) & 0x01u)<<(bit)))
- #define ASSIGN_NIB(REG,   bit,   value)   ((REG) = ((REG) & ~(0x0Fu<<(bit)))   | (((value) & 0x0Fu)<<(bit)))
- #define ASSIGN_BYTE(REG,   bit,   value)   ((REG) = ((REG) & ~(0xFFu<<(bit)))   | (((value) & 0xFFu)<<(bit)))
- #define CON_u8Bits(b7,   b6,   b5,   b4,   b3,   b2,   b1,   b0)

  (0b##b7##b6##b5##b4##b3##b2##b1##b0)
- #define CON_u16Bits(b15,   b14,   b13,   b12,   b11,   b10,   b9,   b8,   b7,   b6,   b5,   b4,   b3,   b2,   b1,   b0)

  (0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2##b1##b0)

---

## Macro Definition Documentation

**#define _BV( bit)   (1u<<(bit))**

**#define ASSIGN_BIT( REG,   bit,   value)   ((REG) = ((REG) & ~(0x01u<<(bit)))   | (((value) & 0x01u)<<(bit)))**

**#define ASSIGN_BYTE( REG,   bit,   value)   ((REG) = ((REG) & ~(0xFFu<<(bit)))   | (((value) & 0xFFu)<<(bit)))**

**#define ASSIGN_NIB( REG,   bit,   value)   ((REG) = ((REG) & ~(0x0Fu<<(bit)))   | (((value) & 0x0Fu)<<(bit)))**

**#define CLR_BIT( REG,   bit)   ((REG) &= ~(1u<<(bit)))**

**#define CLR_BYTE( REG,   bit)   ((REG) &= ~(0xFFu<<(bit)))**

**#define CLR_MASK( REG,   MASK)   ((REG) &= ~(MASK))**

**#define CLR_REG( REG)   ((REG)   =   (0u))**

**#define CON_u16Bits( b15,   b14,   b13,   b12,   b11,   b10,   b9,   b8,   b7,   b6,   b5, b4,   b3,   b2,   b1,   b0)**

**(0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2## b1##b0)**

**#define CON_u8Bits( b7,   b6,   b5,   b4,   b3,   b2,   b1,   b0)**

**(0b##b7##b6##b5##b4##b3##b2##b1##b0)**

**#define GET_BIT( REG,   bit)   (((REG)>>(bit)) & 0x01u)**

**#define GET_BYTE( REG,   bit)   (((REG)>>(bit)) & 0xFFu)**

**#define GET_MASK( REG,   MASK)   ((REG) &   (MASK))**

**#define GET_NIB( REG,   bit)   (((REG)>>(bit)) & 0x0Fu)**

**#define SET_BIT( REG,   bit)   ((REG) |=   (1u<<(bit)))**

Bitwise Operation

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

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

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

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

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

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

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

## LBIT_int.h

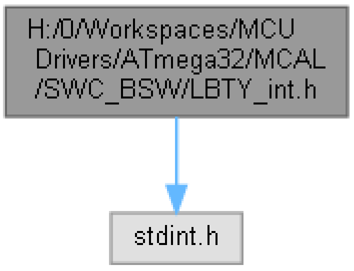
Go to the documentation of this file.

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

#ifndef LBIT_INT_H_
#define LBIT_INT_H_

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

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

#define _BV(bit)                                            (1u<<(bit))

#define SET_BIT(REG, bit)                               ((REG)  |=   (1u<<(bit)))
#define CLR_BIT(REG, bit)                               ((REG)  &= ~(1u<<(bit)))
#define TOG_BIT(REG, bit)                               ((REG)  ^=   (1u<<(bit)))

#define SET_BYTE(REG, bit)                              ((REG)  |=   (0xFFu<<(bit)))
#define CLR_BYTE(REG, bit)                              ((REG)  &= ~(0xFFu<<(bit)))
#define TOG_BYTE(REG, bit)                              ((REG)  ^=   (0xFFu<<(bit)))

#define SET_MASK(REG, MASK)                             ((REG)  |=   (MASK))
#define CLR_MASK(REG, MASK)                             ((REG)  &= ~(MASK))
#define TOG_MASK(REG, MASK)                             ((REG)  ^=   (MASK))
#define GET_MASK(REG, MASK)                             ((REG)  &    (MASK))

#define SET_REG(REG)                                    ((REG)   = ~(0u))
#define CLR_REG(REG)                                    ((REG)   =   (0u))
#define TOG_REG(REG)                                    ((REG)  ^= ~(0u))

#define GET_BIT(REG, bit)                               (((REG)>>(bit))  & 0x01u)
#define GET_NIB(REG, bit)                               (((REG)>>(bit))  & 0x0Fu)
#define GET_BYTE(REG, bit)                              (((REG)>>(bit))  & 0xFFu)

#define ASSIGN_BIT(REG, bit, value)                    ((REG) = ((REG) & ~(0x01u<<(bit))) | (((value) & 0x01u)<<(bit)))
#define ASSIGN_NIB(REG, bit, value)                    ((REG) = ((REG) & ~(0x0Fu<<(bit))) | (((value) & 0x0Fu)<<(bit)))
#define ASSIGN_BYTE(REG, bit, value)                   ((REG) = ((REG) & ~(0xFFu<<(bit))) | (((value) & 0xFFu)<<(bit)))

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

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

/*            bits together in an u16 register           */
#define CON_u16Bits(b15, b14, b13, b12, b11, b10, b9, b8, b7, b6, b5, b4, b3, b2, b1, b0)    \
(0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2##b1##b0)

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

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

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


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

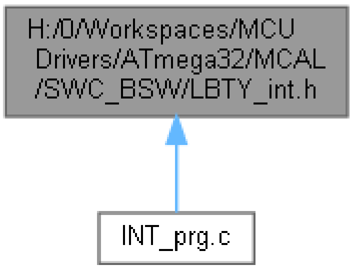
# H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

`#include <stdint.h>`

Include dependency graph for LBTY_int.h:


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

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


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

## Data Structures

- union LBTY_tuniPort8union LBTY_tuniPort16

## Macros

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

## Typedefs

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

## Enumerations

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

## Macro Definition Documentation

**#define __I   volatile const**

**#define __IO   volatile**

**#define __O   volatile**

**#define LBTY_NULL   ((void *) 0U)**

**#define LBTY_s16MAX   ((u16)0x7FFF )**

**#define LBTY_s16MIN   ((u16)0x8000 )**

**#define LBTY_s32MAX   ((u32)0x7FFFFFFFL )**

**#define LBTY_s32MIN   ((u32)0x80000000L )**

**#define LBTY_s64MAX   ((u64)0x7FFFFFFFFFFFFFFFLL )**

**#define LBTY_s64MIN   ((u64)0x8000000000000000LL )**

**#define LBTY_s8MAX   ((s8)0x7F )**

**#define LBTY_s8MIN   ((s8)0x80 )**

**#define LBTY_u16MAX   ((u16)0xFFFFU)**

**#define LBTY_u16ZERO   ((u16)0x0000U)**

**#define LBTY_u32MAX   ((u32)0xFFFFFFFFUL)**

**#define LBTY_u32ZERO   ((u32)0x00000000UL)**

**#define LBTY_u64MAX   ((u64)0xFFFFFFFFFFFFFFFFULL)**

**#define LBTY_u64ZERO   ((u64)0x0000000000000000ULL)**

**#define LBTY_u8MAX   ((u8)0xFFU)**

**#define LBTY_u8vidNOP()**

**#define LBTY_u8ZERO   ((u8)0x00U)**

Data Types Limitation

---

## Typedef Documentation

**typedef float f32**

Standard Real Decimal number

**typedef double f64**

**typedef s16* ps16**

**typedef s32* ps32**

**typedef s64* ps64**

**typedef s8* ps8**

Standard Pointer to Signed Byte/Word/Long_Word

**typedef u16* pu16**

**typedef u32* pu32**

**typedef u64* pu64**

**typedef u8* pu8**

Standard Pointer to Unsigned Byte/Word/Long_Word

**typedef int16_t s16**

**typedef int32_t s32**

**typedef int64_t s64**

**typedef int8_t s8**

Standard Signed Byte/Word/Long_Word

**typedef uint16_t u16**

**typedef uint32_t u32**

**typedef uint64_t u64**

**typedef uint8_t u8**

Data Types New Definitions Standard Unsigned Byte/Word/Long_Word

---

## Enumeration Type Documentation

**enum LBTY_tenuBoolean**

Boolean type

**Enumerator:**

| | |
|---|---|
| LBTY_TRUE | |
| LBTY_FALSE | |

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

**enum LBTY_tenuErrorStatus**

Error Return type

**Enumerator:**

| | |
|---|---|
| LBTY_OK | |
| LBTY_NOK | |
| LBTY_NULL_POINTER | |
| LBTY_INDEX_OUT_OF_RANGE | |
| LBTY_NO_MASTER_CHANNEL | |
| LBTY_READ_ERROR | |
| LBTY_WRITE_ERROR | |
| LBTY_UNDEFINED_ERROR | |
| LBTY_IN_PROGRESS | |

```
              {
  LBTY_OK = (u16)0,
  LBTY_NOK,
  LBTY_NULL_POINTER,
  LBTY_INDEX_OUT_OF_RANGE,
  LBTY_NO_MASTER_CHANNEL,
  LBTY_READ_ERROR,
  LBTY_WRITE_ERROR,
  LBTY_UNDEFINED_ERROR,
  LBTY_IN_PROGRESS        /* Error is not available, wait for availability */
} LBTY_tenuErrorStatus;
```

**enum LBTY_tenuFlagStatus**

Flag Status type

**Enumerator:**

| | |
|---|---|
| LBTY_RESET | |
| LBTY_SET | |

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

## LBTY_int.h

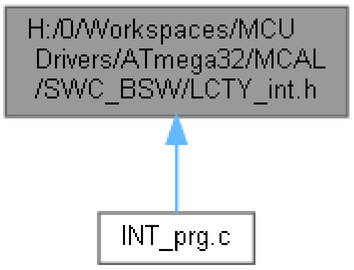
Go to the documentation of this file.

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

#ifndef _LBTY_INT_H_
#define _LBTY_INT_H_

#include <stdint.h>

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

typedef uint8_t     u8 ;
typedef uint16_t    u16;
typedef uint32_t    u32;
typedef uint64_t    u64;

typedef int8_t      s8 ;
typedef int16_t     s16;
typedef int32_t     s32;
typedef int64_t     s64;

typedef float       f32;
typedef double      f64;

typedef u8*         pu8 ;
typedef u16*        pu16;
typedef u32*        pu32;
typedef u64*        pu64;

typedef s8*         ps8 ;
typedef s16*        ps16;
typedef s32*        ps32;
typedef s64*        ps64;

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

/**************************************************************************/
#define __IO    volatile
#define __O     volatile
#define __I     volatile const
/**************************************************************************/

#define LBTY_u8vidNOP()
#define LBTY_NULL       ((void *) 0U)

#define LBTY_u8ZERO     ((u8)0x00U)
#define LBTY_u8MAX      ((u8)0xFFU)
#define LBTY_s8MAX      ((s8)0x7F )
#define LBTY_s8MIN      ((s8)0x80 )

#define LBTY_u16ZERO    ((u16)0x0000U)
#define LBTY_u16MAX     ((u16)0xFFFFU)
#define LBTY_s16MAX     ((u16)0x7FFF )
#define LBTY_s16MIN     ((u16)0x8000 )

#define LBTY_u32ZERO    ((u32)0x00000000UL)
#define LBTY_u32MAX     ((u32)0xFFFFFFFFUL)
#define LBTY_s32MAX     ((u32)0x7FFFFFFFL )
#define LBTY_s32MIN     ((u32)0x80000000L )

```

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

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

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

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

typedef enum {
  LBTY_OK = (u16)0,
  LBTY_NOK,
  LBTY_NULL_POINTER,
  LBTY_INDEX_OUT_OF_RANGE,
  LBTY_NO_MASTER_CHANNEL,
  LBTY_READ_ERROR,
  LBTY_WRITE_ERROR,
  LBTY_UNDEFINED_ERROR,
  LBTY_IN_PROGRESS         /* Error is not available, wait for availability */
} LBTY_tenuErrorStatus;

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

typedef union {
  struct {
    u8 m_u8b0 :1;        // LSB
    u8 m_u8b1 :1;
    u8 m_u8b2 :1;
    u8 m_u8b3 :1;
    u8 m_u8b4 :1;
    u8 m_u8b5 :1;
    u8 m_u8b6 :1;
    u8 m_u8b7 :1;        // MSB
  } sBits;
  u8 u_u8Byte;
} LBTY_tuniPort8;

typedef union {
  struct  {
    u8 m_u8b0  :1;       // LSB
    u8 m_u8b1  :1;
    u8 m_u8b2  :1;
    u8 m_u8b3  :1;
    u8 m_u8b4  :1;
    u8 m_u8b5  :1;
    u8 m_u8b6  :1;
    u8 m_u8b7  :1;
    u8 m_u8b8  :1;
    u8 m_u8b9  :1;
    u8 m_u8b10 :1;
    u8 m_u8b11 :1;
    u8 m_u8b12 :1;
    u8 m_u8b13 :1;
    u8 m_u8b14 :1;
    u8 m_u8b15 :1;       // MSB
  } sBits;
  struct {
    u8 m_u8low;
    u8 m_u8high;
  } sBytes;
  u16 u_u16Word;
} LBTY_tuniPort16;

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


#endif /* _LBTY_INT_H_ */
/*************************** E N D (LBTY_int.h) ******************************/
```

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

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


H:/0/Workspaces/MCU Drivers/ATmega32/MCAL /SWC_BSW/LCTY_int.h
INT_prg.c

### Macros

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

---

### Macro Definition Documentation

#### #define CTY_PACKED   __attribute__((__packed__))

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

#### #define LCTY_CONST   __attribute__((__const__))

#### #define LCTY_DPAGE   __attribute__((dp))

#### #define LCTY_INLINE   __attribute__((always_inline)) static inline

#### #define LCTY_INTERRUPT   __attribute__((interrupt))

#### #define LCTY_NODPAGE   __attribute__((nodp))

#### #define LCTY_PROGMEM   __attribute__((__progmem__))

#### #define LCTY_PURE   __attribute__((__pure__))

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

## LCTY_int.h

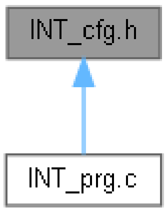
Go to the documentation of this file.

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

#ifndef LCTY_INT_H_
#define LCTY_INT_H_

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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


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

## INT_cfg.c File Reference

## INT_cfg.h File Reference

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



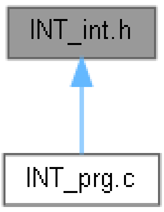
## INT_cfg.h

Go to the documentation of this file.

```
/* *************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name   : INT_cfg.h                                                    */
/* Author      : MAAM                                                         */
/* Version     : v01.2                                                        */
/* date        : Mar 26, 2023                                                 */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES *************************  */
/* ************************************************************************** */

#ifndef INT_CFG_H_
#define INT_CFG_H_

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

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

#if defined(AMIT_KIT)

#define INT_PUSH            INT0

#elif defined(ETA32_KIT)

#define INT_IR_RECEIVER     INT0

#elif defined(ETA32_MINI_KIT)

#define INT_PUSH            INT0

#else

#endif

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

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

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


#endif /* INT_CFG_H_ */
/*************************** E N D (INT_cfg.h) ******************************/
```

## INT_int.h File Reference

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



### Enumerations

- enum INT_tenuPin { INT0 = (u8)0u, INT1, INT2 }
- enum INT_tenuSenseControl { INT_Low_Level = (u8)0u, INT_Logic_Change, INT_Falling_Edge, INT_Rising_Edge, INT2_Falling_Edge = (u8)0u, INT2_Rising_Edge }

### Functions

- void INT_vidInit (u8 u8INT_Num)
- void INT_vidSetSenseControl (u8 u8INT_Num, INT_tenuSenseControl u8INT_SC)
- void INT_vidEnable (u8 u8INT_Num)
- void INT_vidDisable (u8 u8INT_Num)
- void INT_vidSetFlag (u8 u8INT_Num)
- void INT_vidResetFlag (u8 u8INT_Num)
- void INT_vidSetCallBack (u8 u8INT_Num, void(*pvidCallback)(void))

---

### Enumeration Type Documentation

#### enum INT_tenuPin

**Enumerator:**

| | |
|---|---|
| INT0 | |
| INT1 | |
| INT2 | |

```
          {
    INT0 = (u8)0u,
    INT1,
    INT2
}INT_tenuPin;
```

#### enum INT_tenuSenseControl

**Enumerator:**

| | |
|---|---|
| INT_Low_Level | |
| INT_Logic_Change | |
| INT_Falling_Edge | |
| INT_Rising_Edge | |
| INT2_Falling_Edge | |
| INT2_Rising_Edge | |

```
          {
    INT_Low_Level = (u8)0u,
    INT_Logic_Change,
    INT_Falling_Edge,
    INT_Rising_Edge,

    INT2_Falling_Edge = (u8)0u,
    INT2_Rising_Edge
}INT_tenuSenseControl;      // Interrupt Sense Control
```

---

## Function Documentation

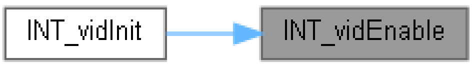
### void INT_vidDisable (u8 *u8INT_Num*)

```
                               {
    switch(u8INT_Num){
        case INT0:          S_GICR->sBits.m_INT0E = LBTY_RESET;          break;
        case INT1:          S_GICR->sBits.m_INT1E = LBTY_RESET;          break;
        case INT2:          S_GICR->sBits.m_INT2E = LBTY_RESET;          break;
        default:            break;
    }
}
```

### void INT_vidEnable (u8 *u8INT_Num*)

```
                               {
    switch(u8INT_Num){
        case INT0:          S_GICR->sBits.m_INT0E = LBTY_SET;            break;
        case INT1:          S_GICR->sBits.m_INT1E = LBTY_SET;            break;
        case INT2:          S_GICR->sBits.m_INT2E = LBTY_SET;            break;
        default:            break;
    }
}
```

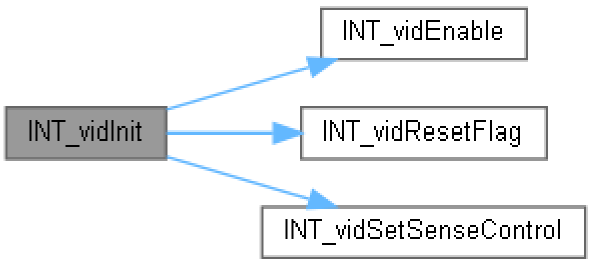
Here is the caller graph for this function:



### void INT_vidInit (u8 *u8INT_Num*)

```
                               {
    switch(u8INT_Num){
    case INT0:
        GPIO_u8SetPinDirection(INT0_PORT, INT0_PIN, PIN_INPUT);
        INT_vidSetSenseControl(u8INT_Num, INT0_SC);
        INT_vidEnable(u8INT_Num);
        INT_vidResetFlag(u8INT_Num);
        break;
    case INT1:
        GPIO_u8SetPinDirection(INT1_PORT, INT1_PIN, PIN_INPUT);
        INT_vidSetSenseControl(u8INT_Num, INT1_SC);
        INT_vidEnable(u8INT_Num);
        INT_vidResetFlag(u8INT_Num);
        break;
    case INT2:
        GPIO_u8SetPinDirection(INT2_PORT, INT2_PIN, PIN_INPUT);
        INT_vidSetSenseControl(u8INT_Num, INT2_SC);
        INT_vidEnable(u8INT_Num);
        INT_vidResetFlag(u8INT_Num);
        break;
    default:
        break;
    }
}
```

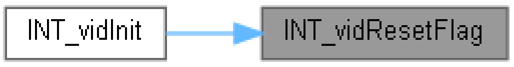
Here is the call graph for this function:



### void INT_vidResetFlag (u8 *u8INT_Num*)

```
                                    {
    switch(u8INT_Num){
        case INT0:          S_GIFR->sBits.m_INT0F = LBTY_RESET;         break;
        case INT1:          S_GIFR->sBits.m_INT1F = LBTY_RESET;         break;
        case INT2:          S_GIFR->sBits.m_INT2F = LBTY_RESET;         break;
        default:            break;
    }
}
```

Here is the caller graph for this function:



### void INT_vidSetCallBack (u8 *u8INT_Num*, void(*)(void) *pvidCallback*)

```
                                                                             {
    if(*pvidCallback == LBTY_NULL)      return;
    switch(u8INT_Num){
        case INT0:          INT0_pvidCallback = pvidCallback;           break;
        case INT1:          INT1_pvidCallback = pvidCallback;           break;
        case INT2:          INT2_pvidCallback = pvidCallback;           break;
        default:            break;
    }
}
```

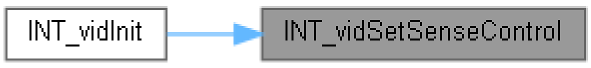
### void INT_vidSetFlag (u8 *u8INT_Num*)

```
                                  {
    switch(u8INT_Num){
        case INT0:          S_GIFR->sBits.m_INT0F = LBTY_SET;           break;
        case INT1:          S_GIFR->sBits.m_INT1F = LBTY_SET;           break;
        case INT2:          S_GIFR->sBits.m_INT2F = LBTY_SET;           break;
        default:            break;
    }
}
```

### void INT_vidSetSenseControl (u8 *u8INT_Num*, INT_tenuSenseControl *u8INT_SC*)

```
                                                                                  {
    switch(u8INT_Num){
        case INT0:          S_MCUCR->sBits.m_ISC0  = u8INT_SC;          break;
        case INT1:          S_MCUCR->sBits.m_ISC1  = u8INT_SC;          break;
        case INT2:          S_MCUCSR->sBits.m_ISC2 = u8INT_SC;          break;
        default:            break;
    }
}
```

Here is the caller graph for this function:



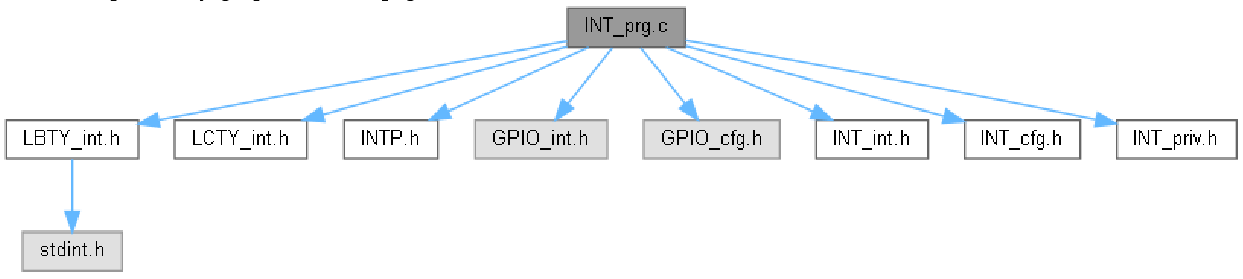
## INT_int.h

Go to the documentation of this file.
```c
/* ************************************************************************ */
/* ********************** FILE DEFINITION SECTION ************************** */
/* ************************************************************************* */
/* File Name   : INT_int.h                                                   */
/* Author      : MAAM                                                        */
/* Version     : v01.2                                                       */
/* date        : Mar 26, 2023                                                */
/* ************************************************************************* */
/* ************************ HEADER FILES INCLUDES *************************  */
/* ************************************************************************** */

#ifndef INT_INT_H_
#define INT_INT_H_

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

typedef enum {
    INT0 = (u8)0u,
    INT1,
    INT2
}INT_tenuPin;

typedef enum{
    INT_Low_Level = (u8)0u,
    INT_Logic_Change,
    INT_Falling_Edge,
    INT_Rising_Edge,

    INT2_Falling_Edge = (u8)0u,
    INT2_Rising_Edge
}INT_tenuSenseControl;      // Interrupt Sense Control

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

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

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

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

/* ************************************************************************** */
/* Description :    Initialize the INT pins direction and SenseControl       */
/* Input       :    u8INT_Num                                                 */
/* Return      :    void                                                      */
/* ************************************************************************** */
extern void INT_vidInit(u8 u8INT_Num);

/* ************************************************************************** */
/* Description :    Set the SenseControl                                      */
/* Input       :    u8INT_Num, u8INT_SC                                       */
/* Return      :    void                                                      */
/* ************************************************************************** */
extern void INT_vidSetSenseControl(u8 u8INT_Num, INT_tenuSenseControl u8INT_SC);

/* ************************************************************************** */
/* Description :    Enable the INT                                            */
/* Input       :    u8INT_Num                                                 */
/* Return      :    void                                                      */
/* ************************************************************************** */
extern void INT_vidEnable(u8 u8INT_Num);

/* ************************************************************************** */
/* Description :    Disable the INT                                            */
/* Input       :    u8INT_Num                                                  */
/* Return      :    void                                                       */
/* ************************************************************************** */
extern void INT_vidDisable(u8 u8INT_Num);

/* ************************************************************************** */
/* Description :    Set the INT Flag                                           */
/* Input       :    u8INT_Num                                                  */
/* Return      :    void                                                       */
/* ************************************************************************** */
extern void INT_vidSetFlag(u8 u8INT_Num);

/* ************************************************************************** */
/* Description :    Reset the INT Flag                                         */
/* Input       :    u8INT_Num                                                  */
/* Return      :    void                                                       */
/* ************************************************************************** */
extern void INT_vidResetFlag(u8 u8INT_Num);

/* ************************************************************************** */
/* Description :    Interrupt Callback                                         */
/* Input       :    void                                                       */
/* Return      :    void                                                       */
/* ************************************************************************** */
extern void INT_vidSetCallBack(u8 u8INT_Num, void (*pvidCallback)(void));

#endif /* INT_INT_H_ */
/*************************** E N D (INT_int.h) ******************************/
```

## INT_prg.c File Reference

```
#include "LBTY_int.h"
#include "LCTY_int.h"
#include "INTP.h"
#include "GPIO_int.h"
#include "GPIO_cfg.h"
#include "INT_int.h"
#include "INT_cfg.h"
#include "INT_priv.h"
```

Include dependency graph for INT_prg.c:



### Functions

- void INT_vidInit (u8 u8INT_Num)
- void INT_vidSetSenseControl (u8 u8INT_Num, INT_tenuSenseControl u8INT_SC)
- void INT_vidEnable (u8 u8INT_Num)
- void INT_vidDisable (u8 u8INT_Num)
- void INT_vidSetFlag (u8 u8INT_Num)
- void INT_vidResetFlag (u8 u8INT_Num)
- void INT_vidSetCallBack (u8 u8INT_Num, void(*pvidCallback)(void))
- ISR (EXT_INT0_vect)
- ISR (EXT_INT1_vect)
- ISR (EXT_INT2_vect)

### Variables

- static void(* INT0_pvidCallback )(void)
- static void(* INT1_pvidCallback )(void)
- static void(* INT2_pvidCallback )(void)

---

### Function Documentation

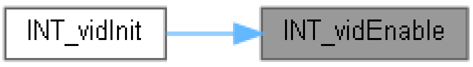
#### void INT_vidDisable (u8 *u8INT_Num*)

```
                                    {
        switch(u8INT_Num){
            case INT0:          S_GICR->sBits.m_INT0E = LBTY_RESET;         break;
            case INT1:          S_GICR->sBits.m_INT1E = LBTY_RESET;         break;
            case INT2:          S_GICR->sBits.m_INT2E = LBTY_RESET;         break;
            default:            break;
        }
    }
```

#### void INT_vidEnable (u8 *u8INT_Num*)

```
                                    {
        switch(u8INT_Num){
            case INT0:          S_GICR->sBits.m_INT0E = LBTY_SET;           break;
            case INT1:          S_GICR->sBits.m_INT1E = LBTY_SET;           break;
            case INT2:          S_GICR->sBits.m_INT2E = LBTY_SET;           break;
            default:                break;
        }
    }
```

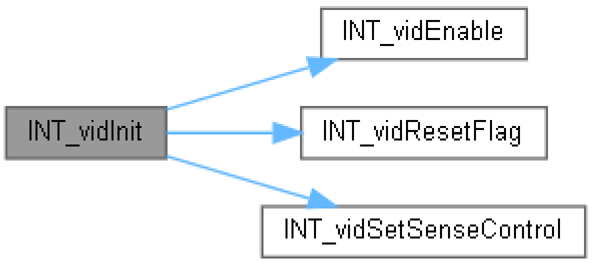
Here is the caller graph for this function:



### void INT_vidInit (u8 *u8INT_Num*)

```
                               {
    switch(u8INT_Num){
        case INT0:
            GPIO_u8SetPinDirection(INT0_PORT, INT0_PIN, PIN_INPUT);
            INT_vidSetSenseControl(u8INT_Num, INT0_SC);
            INT_vidEnable(u8INT_Num);
            INT_vidResetFlag(u8INT_Num);
            break;
        case INT1:
            GPIO_u8SetPinDirection(INT1_PORT, INT1_PIN, PIN_INPUT);
            INT_vidSetSenseControl(u8INT_Num, INT1_SC);
            INT_vidEnable(u8INT_Num);
            INT_vidResetFlag(u8INT_Num);
            break;
        case INT2:
            GPIO_u8SetPinDirection(INT2_PORT, INT2_PIN, PIN_INPUT);
            INT_vidSetSenseControl(u8INT_Num, INT2_SC);
            INT_vidEnable(u8INT_Num);
            INT_vidResetFlag(u8INT_Num);
            break;
        default:
            break;
    }
}
```

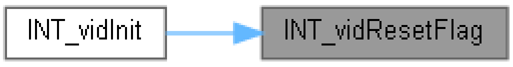
Here is the call graph for this function:



### void INT_vidResetFlag (u8 *u8INT_Num*)

```
                                     {
    switch(u8INT_Num){
        case INT0:          S_GIFR->sBits.m_INT0F = LBTY_RESET;          break;
        case INT1:          S_GIFR->sBits.m_INT1F = LBTY_RESET;          break;
        case INT2:          S_GIFR->sBits.m_INT2F = LBTY_RESET;          break;
        default:            break;
    }
}
```

Here is the caller graph for this function:



### void INT_vidSetCallBack (u8 *u8INT_Num*, void(*)(void) *pvidCallback*)

```
                                                                  {
    if(*pvidCallback == LBTY_NULL)       return;
    switch(u8INT_Num){
        case INT0:          INT0_pvidCallback = pvidCallback;          break;
        case INT1:          INT1_pvidCallback = pvidCallback;          break;
        case INT2:          INT2_pvidCallback = pvidCallback;          break;
        default:            break;
    }
}
```

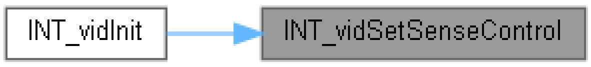
### void INT_vidSetFlag (u8   *u8INT_Num*)

```
                                    {
    switch(u8INT_Num){
        case INT0:          S_GIFR->sBits.m_INT0F = LBTY_SET;            break;
        case INT1:          S_GIFR->sBits.m_INT1F = LBTY_SET;            break;
        case INT2:          S_GIFR->sBits.m_INT2F = LBTY_SET;            break;
        default:            break;
    }
}
```

### void INT_vidSetSenseControl (u8   *u8INT_Num*, INT_tenuSenseControl   *u8INT_SC*)

```
                                                                          {
    switch(u8INT_Num){
        case INT0:          S_MCUCR->sBits.m_ISC0  = u8INT_SC;            break;
        case INT1:          S_MCUCR->sBits.m_ISC1  = u8INT_SC;            break;
        case INT2:          S_MCUCSR->sBits.m_ISC2 = u8INT_SC;            break;
        default:            break;
    }
}
```

Here is the caller graph for this function:



### ISR (EXT_INT0_vect )

```
                      {
    INT0_pvidCallback();
}
```

### ISR (EXT_INT1_vect )

```
                      {
    INT1_pvidCallback();
}
```

### ISR (EXT_INT2_vect )

```
                      {
    INT2_pvidCallback();
}
```

## Variable Documentation

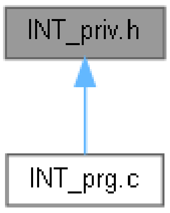
### void(* INT0_pvidCallback) (void) (void ) `[static]`

### void(* INT1_pvidCallback) (void) (void ) `[static]`

### void(* INT2_pvidCallback) (void) (void ) `[static]`

## INT_priv.h File Reference

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



### Data Structures

union MCUCSR_type: *Type define of Union bit field of "Control and Status Register"*

union MCUCR_type: *Type define of Union bit field of "MCU Control Register"*

union GIFR_type: *Type define of Union bit field of "General INT Flag Register"*

union GICR_type: *Type define of Union bit field of "General INT Control Register"*

### Macros

- #define S_MCUCSR ((MCUCSR_type* const)0x54U)
- #define MCUCSR (*(volatile u8* const)0x54U)
- #define S_MCUCR ((MCUCR_type* const)0x55U)
- #define MCUCR (*(volatile u8* const)0x55U)
- #define S_GIFR ((GIFR_type* const)0x5AU)
- #define GIFR (*(volatile u8* const)0x5AU)
- #define S_GICR ((GICR_type* const)0x5BU)
- #define GICR (*(volatile u8* const)0x5BU)
- #define INT0_PORT D
- #define INT0_PIN GPIO_INT0
- #define INT0_SC INT_Rising_Edge
- #define INT1_PORT D
- #define INT1_PIN GPIO_INT1
- #define INT1_SC INT_Low_Level
- #define INT2_PORT B
- #define INT2_PIN GPIO_INT2
- #define INT2_SC INT2_Falling_Edge

## Macro Definition Documentation

**#define GICR   (*(volatile u8* const)0x5BU)**

**#define GIFR   (*(volatile u8* const)0x5AU)**

**#define INT0_PIN   GPIO_INT0**

**#define INT0_PORT   D**

**#define INT0_SC   INT_Rising_Edge**

**#define INT1_PIN   GPIO_INT1**

**#define INT1_PORT   D**

**#define INT1_SC   INT_Low_Level**

**#define INT2_PIN   GPIO_INT2**

**#define INT2_PORT   B**

**#define INT2_SC   INT2_Falling_Edge**

**#define MCUCR   (*(volatile u8* const)0x55U)**

**#define MCUCSR   (*(volatile u8* const)0x54U)**

**#define S_GICR   ((GICR_type* const)0x5BU)**

General Interrupt Control Register

**#define S_GIFR   ((GIFR_type* const)0x5AU)**

General Interrupt Flag Register

**#define S_MCUCR   ((MCUCR_type* const)0x55U)**

MCU Control Register

**#define S_MCUCSR   ((MCUCSR_type* const)0x54U)**

MCU Control and Status Register

## INT_priv.h

Go to the documentation of this file.

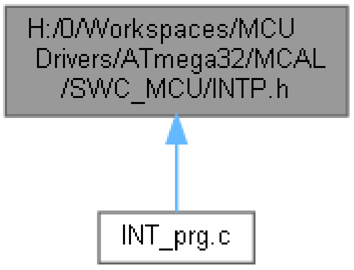
```
/* ************************************************************************ */
/* ********************** FILE DEFINITION SECTION ************************** */
/* ************************************************************************** */
/* File Name   : INT_priv.h                                                 */
/* Author      : MAAM                                                       */
/* Version     : v01.2                                                      */
/* date        : Mar 26, 2023                                               */
/* ************************************************************************** */
/* *********************** HEADER FILES INCLUDES *************************   */
/* *************************************************************************** */

#ifndef INT_PRIV_H_
#define INT_PRIV_H_

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

typedef union{
    u8 u_Reg;
    struct {
        __IO u8         : 6;
        __IO u8 m_ISC2 : 1;
        __IO u8         : 1;
    }sBits;
}MCUCSR_type;

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

typedef union{
    u8 u_Reg;
    struct {
        __IO u8 m_ISC0 : 2;
        __IO u8 m_ISC1 : 2;
        __IO u8         : 4;
    }sBits;
}MCUCR_type;

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

typedef union{
    u8 u_Reg;
    struct {
        __IO u8         : 5;
        __IO u8 m_INT2F: 1;
        __IO u8 m_INT0F: 1;
        __IO u8 m_INT1F: 1;
    }sBits;
}GIFR_type;

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

typedef union{
    u8 u_Reg;
    struct {
        __IO u8         : 5;
        __IO u8 m_INT2E: 1;
        __IO u8 m_INT0E: 1;
        __IO u8 m_INT1E: 1;
    }sBits;
}GICR_type;

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

#define S_MCUCSR        ((MCUCSR_type* const)0x54U)
#define MCUCSR          (*(volatile u8* const)0x54U)

#define S_MCUCR         ((MCUCR_type* const)0x55U)
#define MCUCR           (*(volatile u8* const)0x55U)

```

```c
#define S_GIFR          ((GIFR_type* const)0x5AU)
#define GIFR            (*(volatile u8* const)0x5AU)

#define S_GICR          ((GICR_type* const)0x5BU)
#define GICR            (*(volatile u8* const)0x5BU)

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

#define INT0_PORT       D
#define INT0_PIN        GPIO_INT0
#define INT0_SC         INT_Rising_Edge

#define INT1_PORT       D
#define INT1_PIN        GPIO_INT1
#define INT1_SC         INT_Low_Level

#define INT2_PORT       B
#define INT2_PIN        GPIO_INT2
#define INT2_SC         INT2_Falling_Edge

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

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

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


#endif /* INT_PRIV_H_ */
/*************************** E N D (INT_priv.h) ******************************/
```

## main.c File Reference

## H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_MCU/INTP.h File Reference

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



### Data Structures

union SREG_type: *Type define of Union bit field of "General INT Control Register"*

### Macros

- #define S_SREG   ((SREG_type* const)0x5FU)
- #define SREG   (*(volatile u8* const)0x5FU)
- #define wdr()   __asm__ __volatile__ ("wdr")
- #define sei()   __asm__ __volatile__ ("sei" ::)
- #define cli()   __asm__ __volatile__ ("cli" ::)
- #define reti()   __asm__ __volatile__ ("reti" ::)
- #define _VECTOR(N)   __vector_ ## N
- #define ISR_BLOCK
- #define ISR_NOBLOCK   __attribute__((interrupt))
- #define ISR_NAKED   __attribute__((naked))
- #define ISR_ALIASOF(v)   __attribute__((alias(__STRINGIFY(v))))
- #define ISR(vector, ...)
- #define EXT_INT0_vect   _VECTOR(1)   /* External Interrupt Request 0 */
- #define EXT_INT1_vect   _VECTOR(2)   /* External Interrupt Request 1 */
- #define EXT_INT2_vect   _VECTOR(3)   /* External Interrupt Request 2 */
- #define TIMER2_COMP_vect   _VECTOR(4)   /* Timer/Counter2 Compare Match */
- #define TIMER2_OVF_vect   _VECTOR(5)   /* Timer/Counter2 Overflow */
- #define TIMER1_CAPT_vect   _VECTOR(6)   /* Timer/Counter1 Capture Event */
- #define TIMER1_COMPA_vect   _VECTOR(7)   /* Timer/Counter1 Compare Match A */
- #define TIMER1_COMPB_vect   _VECTOR(8)   /* Timer/Counter1 Compare Match B */
- #define TIMER1_OVF_vect   _VECTOR(9)   /* Timer/Counter1 Overflow */
- #define TIMER0_COMP_vect   _VECTOR(10)   /* Timer/Counter0 Compare Match */
- #define TIMER0_OVF_vect   _VECTOR(11)   /* Timer/Counter0 Overflow */
- #define SPI_STC_vect   _VECTOR(12) /* Serial Transfer Complete */
- #define USART_RXC_vect   _VECTOR(13)   /* USART, Rx Complete */
- #define USART_UDRE_vect   _VECTOR(14)   /* USART Data Register Empty */
- #define USART_TXC_vect   _VECTOR(15)   /* USART, Tx Complete */
- #define ADC_vect   _VECTOR(16)   /* ADC Conversion Complete */
- #define EE_RDY_vect   _VECTOR(17) /* EEPROM Ready */
- #define ANA_COMP_vect   _VECTOR(18)   /* Analog Comparator */
- #define TWI_vect   _VECTOR(19)   /* 2-wire Serial Interface */
- #define SPM_RDY_vect   _VECTOR(20)   /* Store Program Memory Ready */
- #define _VECTORS_SIZE   84

### Functions

- LCTY_INLINE void INTP_vidEnable (void)

- LCTY_INLINE void INTP_vidDisable (void)

---

## Macro Definition Documentation

**#define _VECTOR( N)   __vector_ ## N**

**#define _VECTORS_SIZE   84**

**#define ADC_vect   _VECTOR(16)        /* ADC Conversion Complete */**

**#define ANA_COMP_vect   _VECTOR(18)        /* Analog Comparator */**

**#define cli()   __asm__ __volatile__ ("cli" ::)**

**#define EE_RDY_vect   _VECTOR(17)  /* EEPROM Ready */**

**#define EXT_INT0_vect   _VECTOR(1) /* External Interrupt Request 0 */**

Interrupt Vectors

**#define EXT_INT1_vect   _VECTOR(2) /* External Interrupt Request 1 */**

**#define EXT_INT2_vect   _VECTOR(3) /* External Interrupt Request 2 */**

**#define ISR( vector,    ...)**

```
Value:          void vector(void) __attribute__((signal)) __VA_ARGS__; \
        void vector(void)
```

**#define ISR_ALIASOF( v)   __attribute__((alias(__STRINGIFY(v))))**

**#define ISR_BLOCK**

**#define ISR_NAKED   __attribute__((naked))**

**#define ISR_NOBLOCK   __attribute__((interrupt))**

**#define reti()   __asm__ __volatile__ ("reti" ::)**

**#define S_SREG   ((SREG_type* const)0x5FU)**

Status Register

**#define sei()   __asm__ __volatile__ ("sei" ::)**

**#define SPI_STC_vect   _VECTOR(12) /* Serial Transfer Complete */**

**#define SPM_RDY_vect   _VECTOR(20)   /* Store Program Memory Ready */**

**#define SREG   (*(volatile u8* const)0x5FU)**

**#define TIMER0_COMP_vect   _VECTOR(10)   /* Timer/Counter0 Compare Match */**

**#define TIMER0_OVF_vect   _VECTOR(11)   /* Timer/Counter0 Overflow */**

**#define TIMER1_CAPT_vect   _VECTOR(6)   /* Timer/Counter1 Capture Event */**

**#define TIMER1_COMPA_vect   _VECTOR(7)   /* Timer/Counter1 Compare Match A */**

**#define TIMER1_COMPB_vect   _VECTOR(8)   /* Timer/Counter1 Compare Match B */**

**#define TIMER1_OVF_vect   _VECTOR(9)   /* Timer/Counter1 Overflow */**

**#define TIMER2_COMP_vect   _VECTOR(4)   /* Timer/Counter2 Compare Match */**

**#define TIMER2_OVF_vect   _VECTOR(5)   /* Timer/Counter2 Overflow */**

**#define TWI_vect   _VECTOR(19)   /* 2-wire Serial Interface */**

**#define USART_RXC_vect   _VECTOR(13)   /* USART, Rx Complete */**

**#define USART_TXC_vect   _VECTOR(15)   /* USART, Tx Complete */**

**#define USART_UDRE_vect   _VECTOR(14)   /* USART Data Register Empty */**

**#define wdr()   __asm__ __volatile__ ("wdr")**

---

## Function Documentation

### LCTY_INLINE void INTP_vidDisable (void )

```
                {
    S_SREG->sBits.m_I = LBTY_RESET;
    //cli();
}
```

### LCTY_INLINE void INTP_vidEnable (void )

```
                {
    S_SREG->sBits.m_I = LBTY_SET;
    //sei();
}
```

## INTP.h

Go to the documentation of this file.

```c
/*
************************************************************************** */
/* ********************** FILE DEFINITION SECTION *************************** */
/* ************************************************************************** */
/* File Name    : INT.h                                                      */
/* Author       : MAAM                                                       */
/* Version      : v01.2                                                      */
/* date         : Apr 26, 2023                                               */
/* ************************************************************************** */
/* ************************ HEADER FILES INCLUDES *************************  */
/* ************************************************************************** */

#ifndef INTP_H_
#define INTP_H_

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

typedef union{
    u8 u_Reg;
    struct {
        __IO u8 m_C : 1;
        __IO u8 m_Z : 1;
        __IO u8 m_N : 1;
        __IO u8 m_V : 1;
        __IO u8 m_S : 1;
        __IO u8 m_H : 1;
        __IO u8 m_T : 1;
        __IO u8 m_I : 1;
    }sBits;
}SREG_type;

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

#define S_SREG          ((SREG_type* const)0x5FU)
#define SREG            (*(volatile u8* const)0x5FU)

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

# define wdr()                      __asm__ __volatile__ ("wdr")
# define sei()                      __asm__ __volatile__ ("sei" ::)
# define cli()                      __asm__ __volatile__ ("cli" ::)
# define reti()                     __asm__ __volatile__ ("reti" ::)

#ifndef _VECTOR
#define _VECTOR(N)                  __vector_ ## N
#endif

#define ISR_BLOCK
#define ISR_NOBLOCK                 __attribute__((interrupt))
#define ISR_NAKED                   __attribute__((naked))
#define ISR_ALIASOF(v)              __attribute__((alias(__STRINGIFY(v))))

#define ISR(vector, ...)            \
        void vector(void) __attribute__((signal)) __VA_ARGS__; \
        void vector(void)
/*
#define ISR(vector, ...)            \
    void vector(void) __attribute__ ((signal,used,externally_visible)) __VA_ARGS__;\
    void vector(void)
 */
/*************************************************************************/
/*
Address     Labels  Code                    Comments
$000                jmp RESET           ; Reset Handler
$002                jmp EXT_INT0        ; IRQ0 Handler
$004                jmp EXT_INT1        ; IRQ1 Handler
$006                jmp EXT_INT2        ; IRQ2 Handler
$008                jmp TIM2_COMP       ; Timer2 Compare Handler
$00A                jmp TIM2_OVF        ; Timer2 Overflow Handler
```

```
$00C                jmp TIM1_CAPT      ; Timer1 Capture Handler
$00E                jmp TIM1_COMPA     ; Timer1 CompareA Handler
$010                jmp TIM1_COMPB     ; Timer1 CompareB Handler
$012                jmp TIM1_OVF       ; Timer1 Overflow Handler
$014                jmp TIM0_COMP      ; Timer0 Compare Handler
$016                jmp TIM0_OVF       ; Timer0 Overflow Handler
$018                jmp SPI_STC        ; SPI Transfer Complete Handler
$01A                jmp USART_RXC      ; USART RX Complete Handler
$01C                jmp USART_UDRE     ; UDR Empty Handler
$01E                jmp USART_TXC      ; USART TX Complete Handler
$020                jmp ADC            ; ADC Conversion Complete Handler
$022                jmp EE_RDY         ; EEPROM Ready Handler
$024                jmp ANA_COMP       ; Analog Comparator Handler
$026                jmp TWI            ; Two-wire Serial Interface Handler
$028                jmp SPM_RDY        ; Store Program Memory Ready Handler
;
$02A        RESET:  ldi r16,high(RAMEND) ; Main program start
$02B                out SPH,r16        ; Set Stack Pointer to top of RAM
$02C                ldi r16,low(RAMEND)
$02D                out SPL,r16
$02E                sei                ; Enable interrupts
$02F                <instr> xxx
... ... ...
  */

#define EXT_INT0_vect               _VECTOR(1)  /* External Interrupt Request 0 */
#define EXT_INT1_vect               _VECTOR(2)  /* External Interrupt Request 1 */
#define EXT_INT2_vect               _VECTOR(3)  /* External Interrupt Request 2 */
#define TIMER2_COMP_vect            _VECTOR(4)  /* Timer/Counter2 Compare Match */
#define TIMER2_OVF_vect             _VECTOR(5)  /* Timer/Counter2 Overflow */
#define TIMER1_CAPT_vect            _VECTOR(6)  /* Timer/Counter1 Capture Event */
#define TIMER1_COMPA_vect           _VECTOR(7)  /* Timer/Counter1 Compare Match A */
#define TIMER1_COMPB_vect           _VECTOR(8)  /* Timer/Counter1 Compare Match B */
#define TIMER1_OVF_vect             _VECTOR(9)  /* Timer/Counter1 Overflow */
#define TIMER0_COMP_vect            _VECTOR(10) /* Timer/Counter0 Compare Match */
#define TIMER0_OVF_vect             _VECTOR(11) /* Timer/Counter0 Overflow */
#define SPI_STC_vect                _VECTOR(12) /* Serial Transfer Complete */
#define USART_RXC_vect              _VECTOR(13) /* USART, Rx Complete */
#define USART_UDRE_vect             _VECTOR(14) /* USART Data Register Empty */
#define USART_TXC_vect              _VECTOR(15) /* USART, Tx Complete */
#define ADC_vect                    _VECTOR(16) /* ADC Conversion Complete */
#define EE_RDY_vect                 _VECTOR(17) /* EEPROM Ready */
#define ANA_COMP_vect               _VECTOR(18) /* Analog Comparator */
#define TWI_vect                    _VECTOR(19) /* 2-wire Serial Interface */
#define SPM_RDY_vect                _VECTOR(20) /* Store Program Memory Ready */

#define _VECTORS_SIZE               84

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

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

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

LCTY_INLINE void INTP_vidEnable(void){
    S_SREG->sBits.m_I = LBTY_SET;
    //sei();
}

LCTY_INLINE void INTP_vidDisable(void){
    S_SREG->sBits.m_I = LBTY_RESET;
    //cli();
}

#endif /* INTP_H_ */
/*************************** E N D (INT.h) ******************************/
```