

SWC_INT

Version v1.0

7/16/2023 12:16:00 AM

Table of Contents

Data Structure Index.....	2
File Index.....	3
Data Structure Documentation	4
GICR_type.....	4
GIFR_type.....	6
LBTY_tuniPort16.....	8
LBTY_tuniPort8.....	10
MCUCR_type.....	12
MCUCSR_type	14
SREG_type.....	15
File Documentation	17
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBIT_int.h	17
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBIT_int.h	20
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h.....	22
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h.....	27
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LCTY_int.h.....	30
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LCTY_int.h.....	31
INT_cfg.c	32
INT_cfg.h	33
INT_int.h.....	35
INT_prg.c	40
INT_priv.h.....	43
main.c	47
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_MCU/INTP.h	48
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_MCU/INTP.h	51
Index.....	Error! Bookmark not defined.

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

<u>GICR_type</u> (: Type define of Union bit field of "General INT Control Register")	4
<u>GIFR_type</u> (: Type define of Union bit field of "General INT Flag Register"	
)	6
<u>LBTY_tuniPort16</u>	8
<u>LBTY_tuniPort8</u>	10
<u>MCUCR_type</u> (: Type define of Union bit field of "MCU Control Register"	
)	12
<u>MCUCSR_type</u> (: Type define of Union bit field of "Control and Status Register"	
)	14
<u>SREG_type</u> (: Type define of Union bit field of "General INT Control Register")	15

File Index

File List

Here is a list of all files with brief descriptions:

H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBIT_int.h	17
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h	22
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LCTY_int.h	30
INT_cfg.c	32
INT_cfg.h	33
INT_int.h	35
INT_prg.c	40
INT_priv.h	43
main.c	47
H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_MCU/INTP.h	48

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](#)

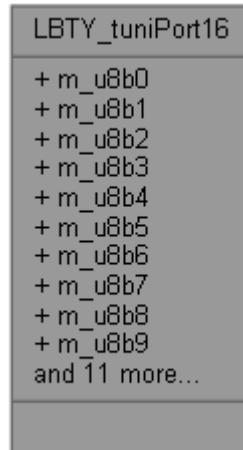
The documentation for this union was generated from the following file:

[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

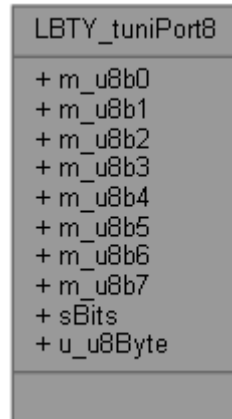
The documentation for this union was generated from the following file:

- 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:

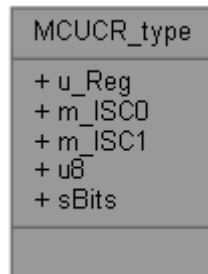
- H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/[LBTY_int.h](#)

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:

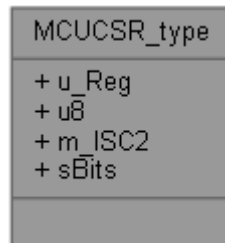
[INT_priv.h](#)

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](#)

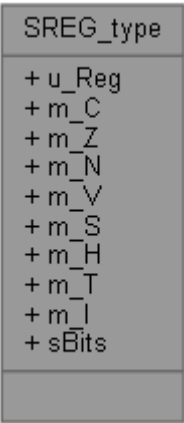
The documentation for this union was generated from the following file:

[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))
```

```

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
26
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 \
55 \
56 \
57 \
58 \
59 \
60 \
61 \
62 \
63 \
64 \
65 \
66 \
67 \
68 \
69 \
70 \
71 \
72 \
73 \
74 \
75 \
76 \
77 \
78 \
79 \
80 \
81 \
82 \
83 \
84 \
85 \
86 \
87 \
88 \
89 \
90 \
91 \
92 \
93 \
94 \
95 \
96 \
97 \
98 \
99 \
100 \
101 \
102 \
103 \
104 \
105 \
106 \
107 \
108 \
109 \
110 \
111 \
112 \
113 \
114 \
115 \
116 \
117 \
118 \
119 \
120 \
121 \
122 \
123 \
124 \
125 \
126 \
127 \
128 \
129 \
130 \
131 \
132 \
133 \
134 \
135 \
136 \
137 \
138 \
139 \
140 \
141 \
142 \
143 \
144 \
145 \
146 \
147 \
148 \
149 \
150 \
151 \
152 \
153 \
154 \
155 \
156 \
157 \
158 \
159 \
160 \
161 \
162 \
163 \
164 \
165 \
166 \
167 \
168 \
169 \
170 \
171 \
172 \
173 \
174 \
175 \
176 \
177 \
178 \
179 \
180 \
181 \
182 \
183 \
184 \
185 \
186 \
187 \
188 \
189 \
190 \
191 \
192 \
193 \
194 \
195 \
196 \
197 \
198 \
199 \
200 \
201 \
202 \
203 \
204 \
205 \
206 \
207 \
208 \
209 \
210 \
211 \
212 \
213 \
214 \
215 \
216 \
217 \
218 \
219 \
220 \
221 \
222 \
223 \
224 \
225 \
226 \
227 \
228 \
229 \
230 \
231 \
232 \
233 \
234 \
235 \
236 \
237 \
238 \
239 \
240 \
241 \
242 \
243 \
244 \
245 \
246 \
247 \
248 \
249 \
250 \
251 \
252 \
253 \
254 \
255 \
256 \
257 \
258 \
259 \
260 \
261 \
262 \
263 \
264 \
265 \
266 \
267 \
268 \
269 \
270 \
271 \
272 \
273 \
274 \
275 \
276 \
277 \
278 \
279 \
280 \
281 \
282 \
283 \
284 \
285 \
286 \
287 \
288 \
289 \
290 \
291 \
292 \
293 \
294 \
295 \
296 \
297 \
298 \
299 \
300 \
301 \
302 \
303 \
304 \
305 \
306 \
307 \
308 \
309 \
310 \
311 \
312 \
313 \
314 \
315 \
316 \
317 \
318 \
319 \
320 \
321 \
322 \
323 \
324 \
325 \
326 \
327 \
328 \
329 \
330 \
331 \
332 \
333 \
334 \
335 \
336 \
337 \
338 \
339 \
340 \
341 \
342 \
343 \
344 \
345 \
346 \
347 \
348 \
349 \
350 \
351 \
352 \
353 \
354 \
355 \
356 \
357 \
358 \
359 \
360 \
361 \
362 \
363 \
364 \
365 \
366 \
367 \
368 \
369 \
370 \
371 \
372 \
373 \
374 \
375 \
376 \
377 \
378 \
379 \
380 \
381 \
382 \
383 \
384 \
385 \
386 \
387 \
388 \
389 \
390 \
391 \
392 \
393 \
394 \
395 \
396 \
397 \
398 \
399 \
400 \
401 \
402 \
403 \
404 \
405 \
406 \
407 \
408 \
409 \
410 \
411 \
412 \
413 \
414 \
415 \
416 \
417 \
418 \
419 \
420 \
421 \
422 \
423 \
424 \
425 \
426 \
427 \
428 \
429 \
430 \
431 \
432 \
433 \
434 \
435 \
436 \
437 \
438 \
439 \
440 \
441 \
442 \
443 \
444 \
445 \
446 \
447 \
448 \
449 \
450 \
451 \
452 \
453 \
454 \
455 \
456 \
457 \
458 \
459 \
460 \
461 \
462 \
463 \
464 \
465 \
466 \
467 \
468 \
469 \
470 \
471 \
472 \
473 \
474 \
475 \
476 \
477 \
478 \
479 \
480 \
481 \
482 \
483 \
484 \
485 \
486 \
487 \
488 \
489 \
490 \
491 \
492 \
493 \
494 \
495 \
496 \
497 \
498 \
499 \
500 \
501 \
502 \
503 \
504 \
505 \
506 \
507 \
508 \
509 \
510 \
511 \
512 \
513 \
514 \
515 \
516 \
517 \
518 \
519 \
520 \
521 \
522 \
523 \
524 \
525 \
526 \
527 \
528 \
529 \
530 \
531 \
532 \
533 \
534 \
535 \
536 \
537 \
538 \
539 \
540 \
541 \
542 \
543 \
544 \
545 \
546 \
547 \
548 \
549 \
550 \
551 \
552 \
553 \
554 \
555 \
556 \
557 \
558 \
559 \
560 \
561 \
562 \
563 \
564 \
565 \
566 \
567 \
568 \
569 \
570 \
571 \
572 \
573 \
574 \
575 \
576 \
577 \
578 \
579 \
580 \
581 \
582 \
583 \
584 \
585 \
586 \
587 \
588 \
589 \
590 \
591 \
592 \
593 \
594 \
595 \
596 \
597 \
598 \
599 \
600 \
601 \
602 \
603 \
604 \
605 \
606 \
607 \
608 \
609 \
610 \
611 \
612 \
613 \
614 \
615 \
616 \
617 \
618 \
619 \
620 \
621 \
622 \
623 \
624 \
625 \
626 \
627 \
628 \
629 \
630 \
631 \
632 \
633 \
634 \
635 \
636 \
637 \
638 \
639 \
640 \
641 \
642 \
643 \
644 \
645 \
646 \
647 \
648 \
649 \
650 \
651 \
652 \
653 \
654 \
655 \
656 \
657 \
658 \
659 \
660 \
661 \
662 \
663 \
664 \
665 \
666 \
667 \
668 \
669 \
670 \
671 \
672 \
673 \
674 \
675 \
676 \
677 \
678 \
679 \
680 \
681 \
682 \
683 \
684 \
685 \
686 \
687 \
688 \
689 \
690 \
691 \
692 \
693 \
694 \
695 \
696 \
697 \
698 \
699 \
700 \
701 \
702 \
703 \
704 \
705 \
706 \
707 \
708 \
709 \
710 \
711 \
712 \
713 \
714 \
715 \
716 \
717 \
718 \
719 \
720 \
721 \
722 \
723 \
724 \

```

```

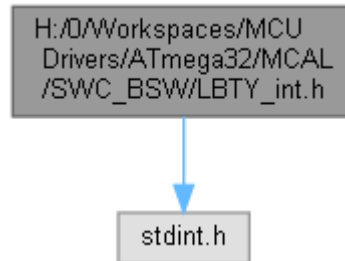
65 (0b##b15##b14##b13##b12##b11##b10##b9##b8##b7##b6##b5##b4##b3##b2##b1##b0)
66
67 /* ***** */
68 /* ***** CONST SECTION ***** */
69 /* ***** */
70
71 /* ***** */
72 /* ***** VARIABLE SECTION ***** */
73 /* ***** */
74
75 /* ***** */
76 /* ***** FUNCTION SECTION ***** */
77 /* ***** */
78
79
80 #endif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

```

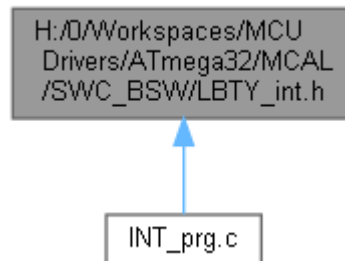

H:/0/Workspaces/MCU Drivers/ATmega32/MCAL/SWC_BSW/LBTY_int.h File Reference

#include <stdint.h>

Include dependency graph for LBTY_int.h:



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



Data Structures

- union [LBTY_tuniPort8](#) union [LBTY_tuniPort16](#)

Macros

- #define [__IO](#) volatile
- #define [__O](#) volatile
- #define [__I](#) volatile const
- #define [LBTY_u8vidNOP](#)()
- #define [LBTY_NULL](#) ((void *) 0U)
- #define [LBTY_u8ZERO](#) ((u8)0x00U)
- #define [LBTY_u8MAX](#) ((u8)0xFFU)
- #define [LBTY_s8MAX](#) ((s8)0x7F)
- #define [LBTY_s8MIN](#) ((s8)0x80)
- #define [LBTY_u16ZERO](#) ((u16)0x0000U)
- #define [LBTY_u16MAX](#) ((u16)0xFFFFU)
- #define [LBTY_s16MAX](#) ((u16)0x7FFF)
- #define [LBTY_s16MIN](#) ((u16)0x8000)
- #define [LBTY_u32ZERO](#) ((u32)0x00000000UL)
- #define [LBTY_u32MAX](#) ((u32)0xFFFFFFFFUL)
- #define [LBTY_s32MAX](#) ((u32)0x7FFFFFFFL)
- #define [LBTY_s32MIN](#) ((u32)0x80000000L)
- #define [LBTY_u64ZERO](#) ((u64)0x0000000000000000ULL)
- #define [LBTY_u64MAX](#) ((u64)0xFFFFFFFFFFFFFFFFULL)
- #define [LBTY_s64MAX](#) ((u64)0x7FFFFFFFFFFFFFFFL)
- #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)0x7FFFFFFFFFFFFFFFL)`

#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
22 typedef uint8_t u8 ;
23
24 typedef uint16_t u16;
25
26 typedef uint32_t u32;
27
28 typedef uint64_t u64;
29
30
31 typedef int8_t s8 ;
32
33 typedef int16_t s16;
34
35 typedef int32_t s32;
36
37 typedef int64_t s64;
38
39
40 typedef float f32;
41
42 typedef double f64;
43
44
45 typedef u8* pu8 ;
46
47 typedef u16* pu16;
48
49 typedef u32* pu32;
50
51 typedef u64* pu64;
52
53
54 typedef s8* ps8 ;
55
56 typedef s16* ps16;
57
58 typedef s32* ps32;
59
60 typedef s64* ps64;
61
62
63 /* ***** */
64 /* ***** MACRO/DEFINE SECTION ***** */
65 /* ***** */
66
67 /*****
68 #define __IO volatile
69 #define __O volatile
70 #define __I volatile const
71 *****/
72
73 #define LBTY_u8vidNOP()
74 #define LBTY_NULL ((void *) 0U)
75
76 #define LBTY_u8ZERO ((u8)0x00U)
77 #define LBTY_u8MAX ((u8)0xFFU)
78 #define LBTY_s8MAX ((s8)0x7F )
79 #define LBTY_s8MIN ((s8)0x80 )
80
81 #define LBTY_u16ZERO ((u16)0x0000U)
82 #define LBTY_u16MAX ((u16)0xFFFFU)
83 #define LBTY_s16MAX ((u16)0x7FFF )
84 #define LBTY_s16MIN ((u16)0x8000 )
85
86 #define LBTY_u32ZERO ((u32)0x00000000UL)
87 #define LBTY_u32MAX ((u32)0xFFFFFFFFUL)
88 #define LBTY_s32MAX ((u32)0x7FFFFFFF )
89 #define LBTY_s32MIN ((u32)0x80000000L )
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2
```

```

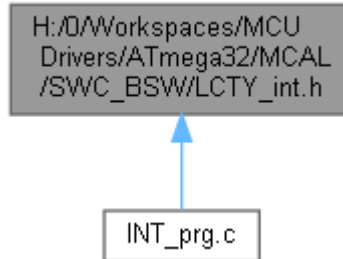
80 #define LBTY_u64ZERO      ((u64)0x0000000000000000ULL)
81 #define LBTY_u64MAX       ((u64)0xFFFFFFFFFFFFFFFFULL)
82 #define LBTY_s64MAX       ((u64)0x7FFFFFFFFFFFFFFFLL )
83 #define LBTY_s64MIN       ((u64)0x8000000000000000LL )
84
85 /* ***** */
86 /* ***** ENUM SECTION ***** */
87 /* ***** */
88
89 typedef enum {
90     LBTY_RESET = 0,
91     LBTY_SET = !LBTY_RESET
92 } LBTY_tenuFlagStatus;
93
94
95 typedef enum {
96     LBTY_TRUE = 0x55,
97     LBTY_FALSE = 0xAA
98 } LBTY_tenuBoolean;
99
100
101 typedef enum {
102     LBTY_OK = (u16)0,
103     LBTY_NOK,
104     LBTY_NULL_POINTER,
105     LBTY_INDEX_OUT_OF_RANGE,
106     LBTY_NO_MASTER_CHANNEL,
107     LBTY_READ_ERROR,
108     LBTY_WRITE_ERROR,
109     LBTY_UNDEFINED_ERROR,
110     LBTY_IN_PROGRESS /* Error is not available, wait for availability */
111 } LBTY_tenuErrorStatus;
112
113
114 /* ***** */
115 /* ***** STRUCT SECTION ***** */
116 /* ***** */
117
118 typedef union {
119     struct {
120         u8 m_u8b0 :1; // LSB
121         u8 m_u8b1 :1;
122         u8 m_u8b2 :1;
123         u8 m_u8b3 :1;
124         u8 m_u8b4 :1;
125         u8 m_u8b5 :1;
126         u8 m_u8b6 :1;
127         u8 m_u8b7 :1; // MSB
128     } sBits;
129     u8 u_u8Byte;
130 } LBTY_tuniPort8;
131
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)))

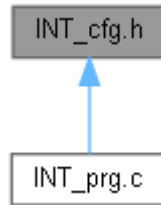
LCTY_int.h

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

INT_cfg.c File Reference

INT_cfg.h File Reference

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

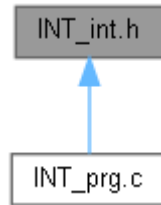


INT_cfg.h

```
Go to the documentation of this file.1 /*
*****
2 /* ***** FILE DEFINITION SECTION ***** */
3 /* ***** */
4 /* File Name      : INT_cfg.h */
5 /* Author         : MAAM */
6 /* Version        : v01.2 */
7 /* date           : Mar 26, 2023 */
8 /* ***** */
9 /* ***** HEADER FILES INCLUDES ***** */
10 /* ***** */
11
12 #ifndef INT_CFG_H_
13 #define INT_CFG_H_
14
15 /* ***** */
16 /* ***** TYPE_DEF/STRUCT/ENUM SECTION ***** */
17 /* ***** */
18
19 /* ***** */
20 /* ***** MACRO/DEFINE SECTION ***** */
21 /* ***** */
22
23 #if defined(AMIT_KIT)
24
25 #define INT_PUSH          INT0
26
27 #elif defined(ETA32_KIT)
28
29 #define INT_IR_RECEIVER   INT0
30
31 #elif defined(ETA32_MINI_KIT)
32
33 #define INT_PUSH          INT0
34
35 #else
36
37 #endif
38
39 /* ***** */
40 /* ***** CONST SECTION ***** */
41 /* ***** */
42
43 /* ***** */
44 /* ***** VARIABLE SECTION ***** */
45 /* ***** */
46
47 /* ***** */
48 /* ***** FUNCTION SECTION ***** */
49 /* ***** */
50
51
52 #endif /* INT_CFG_H_ */
53 /***** 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	

```
19 {
20     INT0 = (u8)0u,
21     INT1,
22     INT2
23 } INT\_tenuPin;
```

enum [INT_tenuSenseControl](#)

Enumerator:

INT_Low_Level	
INT_Logic_Change	
INT_Falling_Edge	
INT_Rising_Edge	
INT2_Falling_Edge	
INT2_Rising_Edge	

```
25 {
26     INT\_Low\_Level = (u8)0u,
```

```

27     INT Logic Change,
28     INT Falling Edge,
29     INT Rising Edge,
30
31     INT2_Falling_Edge = (u8)0u,
32     INT2_Rising_Edge
33 } INT_tenuSenseControl; // Interrupt Sense Control

```

Function Documentation

void INT_vidDisable (u8 u8INT_Num)

```

111                                     {
112     switch(u8INT_Num) {
113         case INT0:          S_GICR->sBits.m_INT0E = LBTY RESET;          break;
114         case INT1:          S_GICR->sBits.m_INT1E = LBTY RESET;          break;
115         case INT2:          S_GICR->sBits.m_INT2E = LBTY RESET;          break;
116         default:            break;
117     }
118 }

```

void INT_vidEnable (u8 u8INT_Num)

```

97                                     {
98     switch(u8INT_Num) {
99         case INT0:          S_GICR->sBits.m_INT0E = LBTY SET;            break;
100         case INT1:          S_GICR->sBits.m_INT1E = LBTY SET;            break;
101         case INT2:          S_GICR->sBits.m_INT2E = LBTY SET;            break;
102         default:            break;
103     }
104 }

```

Here is the caller graph for this function:



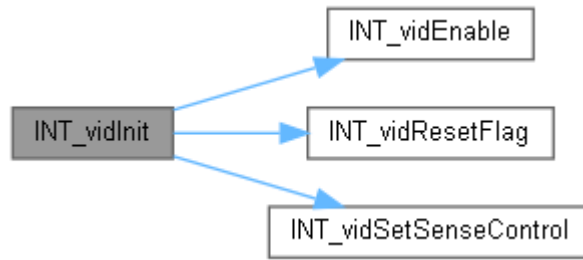
void INT_vidInit (u8 u8INT_Num)

```

53                                     {
54     switch(u8INT_Num) {
55         case INT0:
56             GPIO_u8SetPinDirection(INT0_PORT, INT0_PIN, PIN_INPUT);
57             INT_vidSetSenseControl(u8INT_Num, INT0_SC);
58             INT_vidEnable(u8INT_Num);
59             INT_vidResetFlag(u8INT_Num);
60             break;
61         case INT1:
62             GPIO_u8SetPinDirection(INT1_PORT, INT1_PIN, PIN_INPUT);
63             INT_vidSetSenseControl(u8INT_Num, INT1_SC);
64             INT_vidEnable(u8INT_Num);
65             INT_vidResetFlag(u8INT_Num);
66             break;
67         case INT2:
68             GPIO_u8SetPinDirection(INT2_PORT, INT2_PIN, PIN_INPUT);
69             INT_vidSetSenseControl(u8INT_Num, INT2_SC);
70             INT_vidEnable(u8INT_Num);
71             INT_vidResetFlag(u8INT_Num);
72             break;
73         default:
74             break;
75     }
76 }

```

Here is the call graph for this function:



void INT_vidResetFlag (u8 u8INT_Num)

```

139 {
140     switch(u8INT_Num) {
141         case INT0:      S GIFR->sBits.m_INT0F = LBTY RESET;      break;
142         case INT1:      S GIFR->sBits.m_INT1F = LBTY RESET;      break;
143         case INT2:      S GIFR->sBits.m_INT2F = LBTY RESET;      break;
144         default:        break;
145     }
146 }
  
```

Here is the caller graph for this function:



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

```

153 {
154     if(*pvidCallback == LBTY NULL) return;
155     switch(u8INT_Num) {
156         case INT0:      INT0 pvidCallback = pvidCallback;      break;
157         case INT1:      INT1 pvidCallback = pvidCallback;      break;
158         case INT2:      INT2 pvidCallback = pvidCallback;      break;
159         default:        break;
160     }
161 }
  
```

void INT_vidSetFlag (u8 u8INT_Num)

```

125 {
126     switch(u8INT_Num) {
127         case INT0:      S GIFR->sBits.m_INT0F = LBTY SET;      break;
128         case INT1:      S GIFR->sBits.m_INT1F = LBTY SET;      break;
129         case INT2:      S GIFR->sBits.m_INT2F = LBTY SET;      break;
130         default:        break;
131     }
132 }
  
```

void INT_vidSetSenseControl (u8 u8INT_Num, INT_tenuSenseControl u8INT_SC)

```

83 {
84     switch(u8INT_Num) {
85         case INT0:      S MCUCR->sBits.m_ISC0 = u8INT_SC;      break;
86         case INT1:      S MCUCR->sBits.m_ISC1 = u8INT_SC;      break;
87         case INT2:      S MCUCSR->sBits.m_ISC2 = u8INT_SC;      break;
88         default:        break;
89     }
90 }
  
```

Here is the caller graph for this function:



INT_int.h

```
Go to the documentation of this file.1 /*
*****
2 /* ***** FILE DEFINITION SECTION ***** */
3 /* ***** */
4 /* File Name : INT_int.h */
5 /* Author : MAAM */
6 /* Version : v01.2 */
7 /* date : Mar 26, 2023 */
8 /* ***** */
9 /* ***** HEADER FILES INCLUDES ***** */
10 /* ***** */
11
12 #ifndef INT_INT_H_
13 #define INT_INT_H_
14
15 /* ***** */
16 /* ***** TYPE_DEF/STRUCT/ENUM SECTION ***** */
17 /* ***** */
18
19 typedef enum {
20     INT0 = (u8)0u,
21     INT1,
22     INT2
23 }INT_tenuPin;
24
25 typedef enum{
26     INT_Low_Level = (u8)0u,
27     INT_Logic_Change,
28     INT_Falling_Edge,
29     INT_Rising_Edge,
30
31     INT2_Falling_Edge = (u8)0u,
32     INT2_Rising_Edge
33 }INT_tenuSenseControl; // Interrupt Sense Control
34
35 /* ***** */
36 /* ***** MACRO/DEFINE SECTION ***** */
37 /* ***** */
38
39 /* ***** */
40 /* ***** CONST SECTION ***** */
41 /* ***** */
42
43 /* ***** */
44 /* ***** VARIABLE SECTION ***** */
45 /* ***** */
46
47 /* ***** */
48 /* ***** FUNCTION SECTION ***** */
49 /* ***** */
50
51 /* ***** */
52 /* Description : Initialize the INT pins direction and SenseControl */
53 /* Input : u8INT_Num */
54 /* Return : void */
55 /* ***** */
56 extern void INT_vidInit(u8 u8INT_Num);
57
58 /* ***** */
59 /* Description : Set the SenseControl */
60 /* Input : u8INT_Num, u8INT_SC */
61 /* Return : void */
62 /* ***** */
63 extern void INT_vidSetSenseControl(u8 u8INT_Num, INT_tenuSenseControl u8INT_SC);
64
65 /* ***** */
66 /* Description : Enable the INT */
67 /* Input : u8INT_Num */
68 /* Return : void */
69 /* ***** */
70 extern void INT_vidEnable(u8 u8INT_Num);
71
72 /* ***** */
```

```

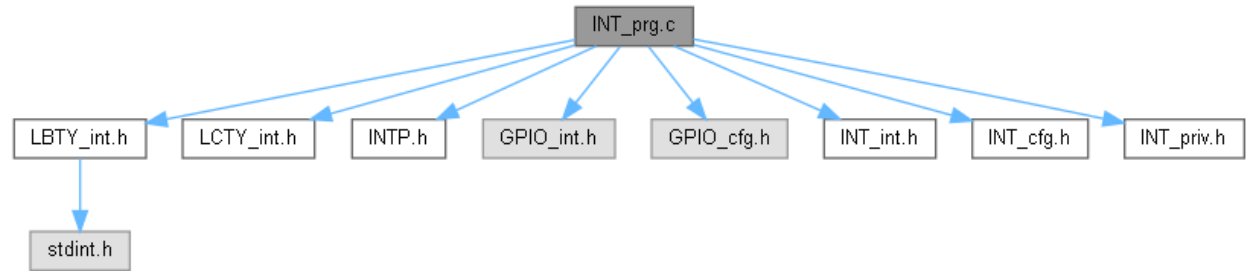
73 /* Description :    Disable the INT                                     */
74 /* Input       :    u8INT_Num                                           */
75 /* Return      :    void                                                */
76 /* ***** */
77 extern void INT_vidDisable(u8 u8INT_Num);
78
79 /* ***** */
80 /* Description :    Set the INT Flag                                     */
81 /* Input       :    u8INT_Num                                           */
82 /* Return      :    void                                                */
83 /* ***** */
84 extern void INT_vidSetFlag(u8 u8INT_Num);
85
86 /* ***** */
87 /* Description :    Reset the INT Flag                                   */
88 /* Input       :    u8INT_Num                                           */
89 /* Return      :    void                                                */
90 /* ***** */
91 extern void INT_vidResetFlag(u8 u8INT_Num);
92
93 /* ***** */
94 /* Description :    Interrupt Callback                                   */
95 /* Input       :    void                                                */
96 /* Return      :    void                                                */
97 /* ***** */
98 extern void INT_vidSetCallBack(u8 u8INT_Num, void (*pvidCallback)(void));
99
100 #endif /* INT_INT_H_ */
101 /***** 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)

```
111                                     {
112     switch(u8INT_Num) {
113         case INT0:                    S_GICR->sBits.m_INT0E = LBTY_RESET;      break;
114         case INT1:                    S_GICR->sBits.m_INT1E = LBTY_RESET;      break;
115         case INT2:                    S_GICR->sBits.m_INT2E = LBTY_RESET;      break;
116         default:                      break;
117     }
118 }
```

void INT_vidEnable (u8 u8INT_Num)

```
97                                     {
98     switch(u8INT_Num) {
99         case INT0:                    S_GICR->sBits.m_INT0E = LBTY_SET;        break;
100         case INT1:                    S_GICR->sBits.m_INT1E = LBTY_SET;        break;
101         case INT2:                    S_GICR->sBits.m_INT2E = LBTY_SET;        break;
```

```

102         default:                break;
103     }
104 }

```

Here is the caller graph for this function:



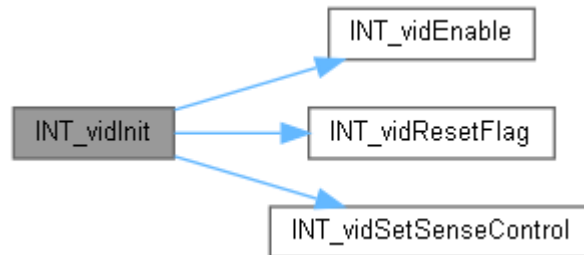
void INT_vidInit (u8 u8INT_Num)

```

53     {
54     switch(u8INT_Num) {
55     case INT0:
56         GPIO_u8SetPinDirection(INT0_PORT, INT0_PIN, PIN_INPUT);
57         INT_vidSetSenseControl(u8INT_Num, INT0_SC);
58         INT_vidEnable(u8INT_Num);
59         INT_vidResetFlag(u8INT_Num);
60         break;
61     case INT1:
62         GPIO_u8SetPinDirection(INT1_PORT, INT1_PIN, PIN_INPUT);
63         INT_vidSetSenseControl(u8INT_Num, INT1_SC);
64         INT_vidEnable(u8INT_Num);
65         INT_vidResetFlag(u8INT_Num);
66         break;
67     case INT2:
68         GPIO_u8SetPinDirection(INT2_PORT, INT2_PIN, PIN_INPUT);
69         INT_vidSetSenseControl(u8INT_Num, INT2_SC);
70         INT_vidEnable(u8INT_Num);
71         INT_vidResetFlag(u8INT_Num);
72         break;
73     default:
74         break;
75     }
76 }

```

Here is the call graph for this function:



void INT_vidResetFlag (u8 u8INT_Num)

```

139     {
140     switch(u8INT_Num) {
141     case INT0:        S GFR->sBits.m_INT0F = LBTY RESET;        break;
142     case INT1:        S GFR->sBits.m_INT1F = LBTY RESET;        break;
143     case INT2:        S GFR->sBits.m_INT2F = LBTY RESET;        break;
144     default:          break;
145     }
146 }

```

Here is the caller graph for this function:



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

```

153     {
154     if(*pvidCallback == LBTY_NULL)        return;
155     switch(u8INT_Num) {
156     case INT0:        INT0_pvidCallback = pvidCallback;        break;
157     case INT1:        INT1_pvidCallback = pvidCallback;        break;
158     case INT2:        INT2_pvidCallback = pvidCallback;        break;
159     default:          break;
160     }
161 }

```

void INT_vidSetFlag ([u8](#) [u8INT_Num](#))

```

125                                     {
126     switch(u8INT_Num) {
127         case INT0:                 S\_GIFR->sBits.m_INT0F = LBTY\_SET;           break;
128         case INT1:                 S\_GIFR->sBits.m_INT1F = LBTY\_SET;           break;
129         case INT2:                 S\_GIFR->sBits.m_INT2F = LBTY\_SET;           break;
130         default:                   break;
131     }
132 }
```

void INT_vidSetSenseControl ([u8](#) [u8INT_Num](#), [INT_tenuSenseControl](#) [u8INT_SC](#))

```

83                                     {
84     switch(u8INT_Num) {
85         case INT0:                 S\_MCUCR->sBits.m_ISC0 = u8INT\_SC;           break;
86         case INT1:                 S\_MCUCR->sBits.m_ISC1 = u8INT\_SC;           break;
87         case INT2:                 S\_MCUCSR->sBits.m_ISC2 = u8INT\_SC;           break;
88         default:                   break;
89     }
90 }
```

Here is the caller graph for this function:



ISR ([EXT_INT0_vect](#))

```

168     {
169     INT0\_pvidCallback();
170 }
```

ISR ([EXT_INT1_vect](#))

```

177     {
178     INT1\_pvidCallback();
179 }
```

ISR ([EXT_INT2_vect](#))

```

186     {
187     INT2\_pvidCallback();
188 }
```

Variable Documentation

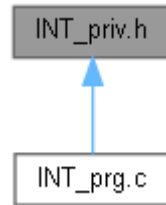
void(* [INT0_pvidCallback](#)) (void) (void) [static]

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.1 /*
*****
2 /* ***** FILE DEFINITION SECTION ***** */
3 /* ***** */
4 /* File Name      : INT_priv.h */
5 /* Author         : MAAM */
6 /* Version        : v01.2 */
7 /* date           : Mar 26, 2023 */
8 /* ***** */
9 /* ***** HEADER FILES INCLUDES ***** */
10 /* ***** */
11
12 #ifndef INT_PRIV_H_
13 #define INT_PRIV_H_
14
15 /* ***** */
16 /* ***** TYPE_DEF/STRUCT/ENUM SECTION ***** */
17 /* ***** */
18
21 typedef union{
22     u8 u_Reg;
23     struct {
24         IO u8 : 6;
25         IO u8 m ISC2 : 1;
26         IO u8 : 1;
27     }sBits;
28 }MCUCSR_type;
29
30 /*****
31
34 typedef union{
35     u8 u_Reg;
36     struct {
37         IO u8 m ISC0 : 2;
38         IO u8 m ISC1 : 2;
39         IO u8 : 4;
40     }sBits;
41 }MCUCR_type;
42
43 /*****
44
47 typedef union{
48     u8 u_Reg;
49     struct {
50         IO u8 : 5;
51         IO u8 m INT2F : 1;
52         IO u8 m INT0F : 1;
53         IO u8 m INT1F : 1;
54     }sBits;
55 }GIFR_type;
56
57 /*****
58
61 typedef union{
62     u8 u_Reg;
63     struct {
64         IO u8 : 5;
65         IO u8 m INT2E : 1;
66         IO u8 m INT0E : 1;
67         IO u8 m INT1E : 1;
68     }sBits;
69 }GICR_type;
70
71 /* ***** */
72 /* ***** MACRO/DEFINE SECTION ***** */
73 /* ***** */
74
76 #define S_MCUCSR ((MCUCSR_type* const)0x54U)
77 #define MCUCSR (*(volatile u8* const)0x54U)
78
80 #define S_MCUCR ((MCUCR_type* const)0x55U)
81 #define MCUCR (*(volatile u8* const)0x55U)
82
```



```

84 #define S_GIFR          ((GIFR_type* const)0x5AU)
85 #define GIFR            (*(volatile u8* const)0x5AU)
86
88 #define S_GICR          ((GICR_type* const)0x5BU)
89 #define GICR            (*(volatile u8* const)0x5BU)
90
91 /* ***** */
92
93 #define INT0_PORT        D
94 #define INT0_PIN         GPIO_INT0
95 #define INT0_SC          INT_Rising_Edge
96
97 #define INT1_PORT        D
98 #define INT1_PIN         GPIO_INT1
99 #define INT1_SC          INT_Low_Level
100
101 #define INT2_PORT        B
102 #define INT2_PIN         GPIO_INT2
103 #define INT2_SC          INT2_Falling_Edge
104
105 /* ***** */
106 /* ***** CONST SECTION ***** */
107 /* ***** */
108
109 /* ***** */
110 /* ***** VARIABLE SECTION ***** */
111 /* ***** */
112
113 /* ***** */
114 /* ***** FUNCTION SECTION ***** */
115 /* ***** */
116
117
118 #endif /* INT_PRIV_H_ */
119 /***** E N D (INT_priv.h) *****/

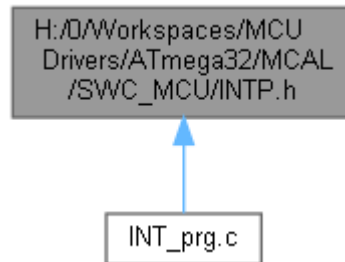
```

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)

```

142                                     {
143     S\_SREG->sBits.m_I = LBTY\_RESET;
144     //cli();
145 }

```

[LCTY_INLINE](#) void INTP_vidEnable (void)

```

137                                     {
138     S\_SREG->sBits.m_I = LBTY\_SET;
139     //sei();
140 }

```

INTP.h

```
Go to the documentation of this file.1 /*
*****
2 /* ***** FILE DEFINITION SECTION ***** */
3 /* ***** */
4 /* File Name      : INTP.h */
5 /* Author         : MAAM */
6 /* Version        : v01.2 */
7 /* date           : Apr 26, 2023 */
8 /* ***** */
9 /* ***** HEADER FILES INCLUDES ***** */
10 /* ***** */
11
12 #ifndef INTP_H_
13 #define INTP_H_
14
15 /* ***** */
16 /* ***** TYPE_DEF/STRUCT/ENUM SECTION ***** */
17 /* ***** */
18
21 typedef union{
22     u8 u_Reg;
23     struct {
24         IO u8 m_C : 1;
25         IO u8 m_Z : 1;
26         IO u8 m_N : 1;
27         IO u8 m_V : 1;
28         IO u8 m_S : 1;
29         IO u8 m_H : 1;
30         IO u8 m_T : 1;
31         IO u8 m_I : 1;
32     }sBits;
33 }SREG type;
34
35 /* ***** */
36 /* ***** MACRO/DEFINE SECTION ***** */
37 /* ***** */
38
40 #define S SREG ((SREG type* const)0x5FU)
41 #define SREG (*(volatile u8* const)0x5FU)
42
43 /* ***** */
44
45 #define wdr() __asm__ __volatile__ ("wdr")
46 #define sei() __asm__ __volatile__ ("sei" ::)
47 #define cli() __asm__ __volatile__ ("cli" ::)
48 #define reti() __asm__ __volatile__ ("reti" ::)
49
50 #ifndef _VECTOR
51 #define _VECTOR(N) __vector_ ## N
52 #endif
53
54 #define ISR_BLOCK
55 #define ISR_NOBLOCK __attribute__((interrupt))
56 #define ISR_NAKED __attribute__((naked))
57 #define ISR_ALIASOF(v) __attribute__((alias( STRINGIFY(v))))
58
59 #define ISR(vector, ...) \
60     void vector(void) __attribute__((signal)) __VA_ARGS__; \
61     void vector(void)
62 /*
63 #define ISR(vector, ...) \
64     void vector(void) __attribute__((signal,used,externally_visible)) __VA_ARGS__; \
65     void vector(void)
66 */
67 /*****
68 */
69
70 Address      Labels  Code      Comments
71 $000         jmp RESET      ; Reset Handler
72 $002         jmp EXT_INT0    ; IRQ0 Handler
73 $004         jmp EXT_INT1    ; IRQ1 Handler
74 $006         jmp EXT_INT2    ; IRQ2 Handler
75 $008         jmp TIM2_COMP    ; Timer2 Compare Handler
76 $00A         jmp TIM2_OVF     ; Timer2 Overflow Handler
```

```

77 $00C          jmp TIM1_CAPT          ; Timer1 Capture Handler
78 $00E          jmp TIM1_COMPA         ; Timer1 CompareA Handler
79 $010          jmp TIM1_COMPB         ; Timer1 CompareB Handler
80 $012          jmp TIM1_OVF           ; Timer1 Overflow Handler
81 $014          jmp TIM0_COMP          ; Timer0 Compare Handler
82 $016          jmp TIM0_OVF           ; Timer0 Overflow Handler
83 $018          jmp SPI_STC            ; SPI Transfer Complete Handler
84 $01A          jmp USART_RXC          ; USART RX Complete Handler
85 $01C          jmp USART_UDRE         ; UDR Empty Handler
86 $01E          jmp USART_TXC          ; USART TX Complete Handler
87 $020          jmp ADC                ; ADC Conversion Complete Handler
88 $022          jmp EE_RDY             ; EEPROM Ready Handler
89 $024          jmp ANA_COMP           ; Analog Comparator Handler
90 $026          jmp TWI                ; Two-wire Serial Interface Handler
91 $028          jmp SPM_RDY            ; Store Program Memory Ready Handler
92 ;
93 $02A          RESET: ldi r16,high(RAMEND) ; Main program start
94 $02B          out SPH,r16            ; Set Stack Pointer to top of RAM
95 $02C          ldi r16,low(RAMEND)
96 $02D          out SPL,r16
97 $02E          sei                    ; Enable interrupts
98 $02F          <instr> xxx
99 ... ..
100 */
101
102 #define EXT_INT0_vect          _VECTOR(1) /* External Interrupt Request 0 */
103 #define EXT_INT1_vect          _VECTOR(2) /* External Interrupt Request 1 */
104 #define EXT_INT2_vect          _VECTOR(3) /* External Interrupt Request 2 */
105 #define TIMER2_COMP_vect       _VECTOR(4) /* Timer/Counter2 Compare Match */
106 #define TIMER2_OVF_vect        _VECTOR(5) /* Timer/Counter2 Overflow */
107 #define TIMER1_CAPT_vect       _VECTOR(6) /* Timer/Counter1 Capture Event */
108 #define TIMER1_COMPA_vect      _VECTOR(7) /* Timer/Counter1 Compare Match A */
109 #define TIMER1_COMPB_vect      _VECTOR(8) /* Timer/Counter1 Compare Match B */
110 #define TIMER1_OVF_vect        _VECTOR(9) /* Timer/Counter1 Overflow */
111 #define TIMER0_COMP_vect       _VECTOR(10) /* Timer/Counter0 Compare Match */
112 #define TIMER0_OVF_vect        _VECTOR(11) /* Timer/Counter0 Overflow */
113 #define SPI_STC_vect           _VECTOR(12) /* Serial Transfer Complete */
114 #define USART_RXC_vect         _VECTOR(13) /* USART, Rx Complete */
115 #define USART_UDRE_vect        _VECTOR(14) /* USART Data Register Empty */
116 #define USART_TXC_vect         _VECTOR(15) /* USART, Tx Complete */
117 #define ADC_vect               _VECTOR(16) /* ADC Conversion Complete */
118 #define EE_RDY_vect            _VECTOR(17) /* EEPROM Ready */
119 #define ANA_COMP_vect          _VECTOR(18) /* Analog Comparator */
120 #define TWI_vect               _VECTOR(19) /* 2-wire Serial Interface */
121 #define SPM_RDY_vect           _VECTOR(20) /* Store Program Memory Ready */
122
123 #define _VECTORS_SIZE          84
124
125 /* ***** */
126 /* ***** CONST SECTION ***** */
127 /* ***** */
128
129 /* ***** */
130 /* ***** VARIABLE SECTION ***** */
131 /* ***** */
132
133 /* ***** */
134 /* ***** FUNCTION SECTION ***** */
135 /* ***** */
136
137 LCTY_INLINE void INTP_vidEnable(void){
138     S_SREG->sBits.m_I = LBTY_SET;
139     //sei();
140 }
141
142 LCTY_INLINE void INTP_vidDisable(void){
143     S_SREG->sBits.m_I = LBTY_RESET;
144     //cli();
145 }
146
147 #endif /* INTP_H */
148 /***** E N D (INT.h) *****/

```