

BuguRTOS native lib

4.1.0

Generated by Doxygen 1.8.17

1 Data Structure Index	2
1.1 Data Structures	2
2 File Index	2
2.1 File List	2
3 Data Structure Documentation	2
3.1 bgrt_priv_cond_t Struct Reference	2
3.1.1 Detailed Description	3
3.1.2 Field Documentation	3
3.2 bgrt_priv_ipc_t Struct Reference	3
3.2.1 Detailed Description	3
3.2.2 Field Documentation	3
3.3 bgrt_priv_mtx_t Struct Reference	4
3.3.1 Detailed Description	4
3.3.2 Field Documentation	4
3.4 bgrt_priv_sem_t Struct Reference	5
3.4.1 Detailed Description	5
3.4.2 Field Documentation	5
4 File Documentation	6
4.1 bugertos/doc/doxygen/bugurt_port.h File Reference	6
4.1.1 Macro Definition Documentation	6
4.1.2 Function Documentation	8
4.2 bugertos/libs/native/cond.h File Reference	10
4.2.1 Detailed Description	10
4.2.2 Typedef Documentation	11
4.2.3 Function Documentation	11
4.3 bugertos/libs/native/ipc.h File Reference	13
4.3.1 Detailed Description	14
4.3.2 Typedef Documentation	14
4.3.3 Function Documentation	14
4.4 bugertos/libs/native/mutex.h File Reference	16
4.4.1 Detailed Description	17
4.4.2 Typedef Documentation	17
4.4.3 Function Documentation	17
4.5 bugertos/libs/native/native.h File Reference	18
4.5.1 Detailed Description	19
4.5.2 Macro Definition Documentation	19
4.6 bugertos/libs/native/sem.h File Reference	21
4.6.1 Detailed Description	22
4.6.2 Typedef Documentation	22
4.6.3 Function Documentation	22

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

bgrt_priv_cond_t	A conditional variable	2
bgrt_priv_ipc_t	An IPC endpoint	3
bgrt_priv_mtx_t	A mutex	4
bgrt_priv_sem_t	A counting semaphore	5

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

bugurtos/doc/doxygen/bugurt_port.h		6
bugurtos/libs/native/cond.h	A conditional variable header	10
bugurtos/libs/native/ipc.h	An IPC header	13
bugurtos/libs/native/mutex.h	A mutex header	16
bugurtos/libs/native/native.h	Native API	18
bugurtos/libs/native/sem.h	A counting semaphores header	21

3 Data Structure Documentation

3.1 bgrt_priv_cond_t Struct Reference

A conditional variable.

```
#include "bugurtos/libs/native/cond.h"
```

Data Fields

- bgrt_sync_t [wait](#)

3.1.1 Detailed Description

A conditional variable.

Conditional variables with mutexes are used for process-event synchronization. A process can block on conditional variable. Other process can launch one or all processes blocked on conditional variable.

3.1.2 Field Documentation

3.1.2.1 wait `bgrt_sync_t bgrt_priv_cond_t::wait`

A list of waiting processes.

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

- [bugurtos/libs/native/cond.h](#)

3.2 bgrt_priv_ipc_t Struct Reference

An IPC endpoint.

```
#include "bugurtos/libs/native/ipc.h"
```

Data Fields

- bgrt_sync_t [wait](#)
- void * [msg](#)

3.2.1 Detailed Description

An IPC endpoint.

Used for blocking synchronous or asynchronous IPC protocol implementation.

3.2.2 Field Documentation

3.2.2.1 msg void* bgrt_priv_ipc_t::msg

A message buffer pointer.

3.2.2.2 wait bgrt_sync_t bgrt_priv_ipc_t::wait

A list of waiting processes.

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

- [bugurtos/libs/native/ipc.h](#)

3.3 bgrt_priv_mtx_t Struct Reference

A mutex.

```
#include "bugurtos/libs/native/mutex.h"
```

Data Fields

- bgrt_sync_t [wait](#)

3.3.1 Detailed Description

A mutex.

Mutexes are used to control an access to common data. If your code needs yo use some common data for a long time, then you should use mutex instead of critical section. Mutex nesting is supported.

Warning

- Only a process can lock or free a mutex!
- Locked mutex can be freed only by a locker process!

3.3.2 Field Documentation

3.3.2.1 wait bgrt_sync_t bgrt_priv_mtx_t::wait

A list of waiting processes.

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

- [bugurtos/libs/native/mutex.h](#)

3.4 bgrt_priv_sem_t Struct Reference

A counting semaphore.

```
#include "bugurtos/libs/native/sem.h"
```

Data Fields

- bgrt_sync_t [wait](#)
- bgrt_cnt_t [counter](#)
- bgrt_lock_t [lock](#)

3.4.1 Detailed Description

A counting semaphore.

Counting semaphores are used for process synchronization. It is not recommended to use them in common data access control, because priority inversion is possible. A counting semaphore can be locked by one process and freed by another.

3.4.2 Field Documentation

3.4.2.1 counter `bgrt_cnt_t bgrt_priv_sem_t::counter`

A resource counter.

3.4.2.2 lock `bgrt_lock_t bgrt_priv_sem_t::lock`

A sync spin-lock.

3.4.2.3 wait `bgrt_sync_t bgrt_priv_sem_t::wait`

A list of waiting processes.

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

- [bugurtos/libs/native/sem.h](#)

4 File Documentation

4.1 bugurtos/doc/doxygen/bugurt_port.h File Reference

Macros

- #define [BGRT_INT_LOCK\(\)](#)
Disable interrupts.
- #define [BGRT_INT_FREE\(\)](#)
Enable interrupts.
- #define [BGRT_KBLOCK](#)
Current kernel block.
- #define [BGRT_CURR_PROC](#)
Current process.
- #define [BGRT_ISR\(v\)](#)
Interrupt service routine declaration template.
- #define [BGRT_ATM_INIT_ISR\(map_ptr\)](#)
Atomic map initialization.
- #define [BGRT_ATM_BSET_ISR\(map_ptr, msk\)](#)
Set masked bits.
- #define [BGRT_ATM_BGET_ISR\(map_ptr, msk\)](#)
Read masked bits.
- #define [BGRT_ATM_BCLR_ISR\(map_ptr, msk\)](#)
Clear masked bits.

Functions

- void [bgrt_atm_init](#) (bgrt_map_t *map_ptr)
Atomic map initialization.
- void [bgrt_atm_bset](#) (bgrt_map_t *map_ptr, bgrt_map_t msk)
Set bits using mask.
- bgrt_map_t [bgrt_atm_bget](#) (bgrt_map_t *map_ptr, bgrt_map_t msk)
Read masked bits.
- bgrt_map_t [bgrt_atm_bclr](#) (bgrt_map_t *map_ptr, bgrt_map_t msk)
Clear masked bits.

4.1.1 Macro Definition Documentation

4.1.1.1 BGRT_ATM_BCLR_ISR #define BGRT_ATM_BCLR_ISR(
map_ptr,
msk)

Clear masked bits.

Warning

For ISR/crit_sec usage!

Parameters

<i>map_ptr</i>	A pointer to atomic map.
<i>msk</i>	A mask.

Returns

Last masked bits state.

4.1.1.2 BGR_T_ATM_BGET_ISR `#define BGR_T_ATM_BGET_ISR(
 map_ptr,
 msk)`

Read masked bits.

Warning

For ISR/crit_sec usage!

Parameters

<i>map_ptr</i>	A pointer to atomic map.
<i>msk</i>	A mask.

Returns

Masked vectors state.

4.1.1.3 BGR_T_ATM_BSET_ISR `#define BGR_T_ATM_BSET_ISR(
 map_ptr,
 msk)`

Set masked bits.

Parameters

<i>map_ptr</i>	A pointer to atomic map.
<i>msk</i>	A mask.

4.1.1.4 BGR_T_ATM_INIT_ISR `#define BGR_T_ATM_INIT_ISR(
 map_ptr)`

Atomic map initialization.

Warning

For ISR/crit_sec usage!

Parameters

<i>map_ptr</i>	A pointer to atomic map.
----------------	--------------------------

4.1.1.5 BGRT_CURR_PROC `#define BGRT_CURR_PROC`

Current process.

4.1.1.6 BGRT_INT_FREE `#define BGRT_INT_FREE()`

Enable interrupts.

4.1.1.7 BGRT_INT_LOCK `#define BGRT_INT_LOCK()`

Disable interrupts.

4.1.1.8 BGRT_ISR `#define BGRT_ISR(
v)`

Interrupt service routine declaration template.

Parameters

<i>v</i>	An interrupt vector id.
----------	-------------------------

4.1.1.9 BGRT_KBLOCK `#define BGRT_KBLOCK`

Current kernel block.

4.1.2 Function Documentation

4.1.2.1 bgrt_atm_bclr() `bgrt_map_t bgrt_atm_bclr (`
 `bgrt_map_t * map_ptr,`
 `bgrt_map_t msk)`

Clear masked bits.

Parameters

<i>map_ptr</i>	A pointer to atomic map.
<i>msk</i>	A mask.

Returns

Last masked bits state.

4.1.2.2 bgrt_atm_bget() `bgrt_map_t bgrt_atm_bget (`
 `bgrt_map_t * map_ptr,`
 `bgrt_map_t msk)`

Read masked bits.

Parameters

<i>map_ptr</i>	A pointer to atomic map.
<i>msk</i>	A mask.

Returns

Masked vectors state.

4.1.2.3 bgrt_atm_bset() `void bgrt_atm_bset (`
 `bgrt_map_t * map_ptr,`
 `bgrt_map_t msk)`

Set bits using mask.

Warning

For ISR/crit_sec usage!

Parameters

<i>map_ptr</i>	A pointer to atomic map.
<i>msk</i>	A mask.

4.1.2.4 bgrt_atm_init() void bgrt_atm_init (
 bgrt_map_t * map_ptr)

Atomic map initialization.

Parameters

<i>map_ptr</i>	A pointer to atomic map.
----------------	--------------------------

4.2 bugurtos/libs/native/cond.h File Reference

A conditional variable header.

```
#include <bugurt.h>
#include "mutex.h"
```

Data Structures

- struct [bgrt_priv_cond_t](#)
 A conditional variable.

Typedefs

- typedef typedefBGRT_CDECL_BEGIN struct [bgrt_priv_cond_t](#) [bgrt_cond_t](#)

Functions

- bgrt_st_t [bgrt_cond_init_cs](#) ([bgrt_cond_t](#) *cond)
 A conditional variable initiation from ISR or critical section.
- bgrt_st_t [bgrt_cond_init](#) ([bgrt_cond_t](#) *cond)
 A conditional variable initiation.
- bgrt_st_t [bgrt_cond_wait](#) ([bgrt_cond_t](#) *cond, [bgrt_mtx_t](#) *mutex)
 Wait for a condition.
- bgrt_st_t [bgrt_cond_signal](#) ([bgrt_cond_t](#) *cond)
 Launch one waiting process.
- bgrt_st_t [bgrt_cond_broadcast](#) ([bgrt_cond_t](#) *cond)
 Launch all waiting processes.

4.2.1 Detailed Description

A conditional variable header.

4.2.2 Typedef Documentation

4.2.2.1 `bgrt_cond_t` `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_cond_t bgrt_cond_t`

See `bgrt_priv_cond_t`;

4.2.3 Function Documentation

4.2.3.1 `bgrt_cond_broadcast()` `bgrt_st_t bgrt_cond_broadcast (bgrt_cond_t * cond)`

Launch all waiting processes.

Launches all processes from waiting process list.

Warning

Caller must lock mutex first!

Parameters

<code>cond</code>	A <code>bgrt_cond_t</code> pointer.
-------------------	-------------------------------------

Returns

BGRT_ST_OK on success, or error number.

4.2.3.2 `bgrt_cond_init()` `bgrt_st_t bgrt_cond_init (bgrt_cond_t * cond)`

A conditional variable initiation.

Parameters

<code>cond</code>	A <code>bgrt_cond_t</code> pointer.
-------------------	-------------------------------------

4.2.3.3 `bgrt_cond_init_cs()` `bgrt_st_t bgrt_cond_init_cs (bgrt_cond_t * cond)`

A conditional variable initiation from ISR or critical section.

Parameters

<i>cond</i>	A <code>bgrt_cond_t</code> pointer.
-------------	-------------------------------------

4.2.3.4 `bgrt_cond_signal()` `bgrt_st_t bgrt_cond_signal (`
`bgrt_cond_t * cond)`

Launch one waiting process.

Launches the head of waiting process list.

Warning

Caller must lock mutex first!

Parameters

<i>cond</i>	A <code>bgrt_cond_t</code> pointer.
-------------	-------------------------------------

Returns

BGRT_ST_OK on success, or error number.

4.2.3.5 `bgrt_cond_wait()` `bgrt_st_t bgrt_cond_wait (`
`bgrt_cond_t * cond,`
`bgrt_mtx_t * mutex)`

Wait for a condition.

This function stops caller process and inserts it to conditional variable wait list.

Parameters

<i>cond</i>	A <code>bgrt_cond_t</code> pointer.
<i>mutex</i>	A pointer to a mutex which protects a conditional variable.

Returns

BGRT_ST_OK on success, or error number.

4.3 bugurtos/libs/native/ipc.h File Reference

An IPC header.

```
#include <bugurt.h>
```

Data Structures

- struct [bgrt_priv_ipc_t](#)
An IPC endpoint.

Typedefs

- typedef typedefBGRT_CDECL_BEGIN struct [bgrt_priv_ipc_t](#) [bgrt_ipc_t](#)

Functions

- [bgrt_st_t](#) [bgrt_ipc_init_cs](#) ([bgrt_ipc_t](#) *endpoint)
IPC endpoint initiation from ISR or critical section.
- [bgrt_st_t](#) [bgrt_ipc_init](#) ([bgrt_ipc_t](#) *endpoint)
IPC endpoint initiation.
- [bgrt_st_t](#) [bgrt_ipc_send](#) ([bgrt_ipc_t](#) *out, void *msg)
IPC data transmission.
- [bgrt_st_t](#) [bgrt_ipc_wait](#) ([bgrt_ipc_t](#) *in, BGRT_PID_T *pid, [bgrt_flag_t](#) block)
Wait for IPC.
- [bgrt_st_t](#) [bgrt_ipc_reply](#) ([bgrt_ipc_t](#) *in, BGRT_PID_T pid)
Unblock a sender process, which message has been received.

4.3.1 Detailed Description

An IPC header.

4.3.2 Typedef Documentation

4.3.2.1 [bgrt_ipc_t](#) typedef typedefBGRT_CDECL_BEGIN struct [bgrt_priv_ipc_t](#) [bgrt_ipc_t](#)

See [bgrt_priv_ipc_t](#);

4.3.3 Function Documentation

4.3.3.1 [bgrt_ipc_init\(\)](#) [bgrt_st_t](#) [bgrt_ipc_init](#) (
[bgrt_ipc_t](#) * endpoint)

IPC endpoint initiation.

Parameters

<i>endpoint</i>	A pointer to the endpoint.
-----------------	----------------------------

4.3.3.2 bgrt_ipc_init_cs() `bgrt_st_t bgrt_ipc_init_cs (`
`bgrt_ipc_t * endpoint)`

IPC endpoint initiation from ISR or critical section.

Parameters

<i>endpoint</i>	A pointer to the endpoint.
-----------------	----------------------------

4.3.3.3 bgrt_ipc_reply() `bgrt_st_t bgrt_ipc_reply (`
`bgrt_ipc_t * in,`
`BGRT_PID_T pid)`

Unblock a sender process, which message has been received.

Parameters

<i>in</i>	An IPC endpoint pointer.
<i>pid</i>	A sender process ID.

Returns

BGRT_ST_OK on success, or error number.

4.3.3.4 bgrt_ipc_send() `bgrt_st_t bgrt_ipc_send (`
`bgrt_ipc_t * out,`
`void * msg)`

IPC data transmission.

This function transfers a pointer to the message buffer through IPC. Senders are blocked on IPC endpoint and wait for their turn, receiver inherits senders priorities.

Parameters

<i>out</i>	An IPC endpoint pointer.
<i>msg</i>	A message buffer pointer.

Returns

BGRT_ST_OK on success, or error number.


```

4.3.3.5 bgrt_ipc_wait() bgrt_st_t bgrt_ipc_wait (
    bgrt_ipc_t * in,
    BGRT_PID_T * pid,
    bgrt_flag_t block )

```

Wait for IPC.

A buffer must be used to set or get sender process. A buffer pointer must be passed as a second parameter.

Parameters

<i>in</i>	An IPC endpoint pointer.
<i>pid</i>	A sender pid buffer pointer.
<i>block</i>	A caller block flag. If non zero, then caller is blocked until message is sent.

Returns

BGRT_ST_OK on success, or error number.

4.4 bugurtos/libs/native/mutex.h File Reference

A mutex header.

```
#include <bugurt.h>
```

Data Structures

- struct [bgrt_priv_mtx_t](#)
A mutex.

Typedefs

- typedef typedefBGRT_CDECL_BEGIN struct [bgrt_priv_mtx_t](#) [bgrt_mtx_t](#)

Functions

- [bgrt_st_t bgrt_mtx_init_cs](#) ([bgrt_mtx_t](#) *mutex, [bgrt_prio_t](#) prio)
A mutex initiation for usage in ISRs or in critical sections.
- [bgrt_st_t bgrt_mtx_init](#) ([bgrt_mtx_t](#) *mutex, [bgrt_prio_t](#) prio)
A mutex initiation.
- [bgrt_st_t bgrt_mtx_try_lock](#) ([bgrt_mtx_t](#) *mutex)
Try to lock a mutex.
- [bgrt_st_t bgrt_mtx_lock](#) ([bgrt_mtx_t](#) *mutex)
Lock a mutex.
- [bgrt_st_t bgrt_mtx_free](#) ([bgrt_mtx_t](#) *mutex)
Mutex free.

4.4.1 Detailed Description

A mutex header.

4.4.2 Typedef Documentation

4.4.2.1 bgrt_mtx_t typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_mtx_t bgrt_mtx_t

See bgrt_priv_mtx_t;

4.4.3 Function Documentation

4.4.3.1 bgrt_mtx_free() bgrt_st_t bgrt_mtx_free (
 bgrt_mtx_t * mutex)

Mutex free.

If a mutex wait list is empty, then caller process frees a mutex, else mutex wait list head gets launched.

Parameters

<i>mutex</i>	A mutex pointer.
--------------	------------------

Returns

BGRT_ST_OK on success, or error number.

4.4.3.2 bgrt_mtx_init() bgrt_st_t bgrt_mtx_init (
 bgrt_mtx_t * mutex,
 bgrt_prio_t prio)

A mutex initiation.

Parameters

<i>mutex</i>	A mutex pointer.
<i>prio</i>	A mutex priority.

4.4.3.3 bgrt_mtx_init_cs() `bgrt_st_t bgrt_mtx_init_cs (`
`bgrt_mtx_t * mutex,`
`bgrt_prio_t prio)`

A mutex initiation for usage in ISRs or in critical sections.

Parameters

<i>mutex</i>	A mutex pointer.
<i>prio</i>	A mutex priority.

4.4.3.4 bgrt_mtx_lock() `bgrt_st_t bgrt_mtx_lock (`
`bgrt_mtx_t * mutex)`

Lock a mutex.

If a mutex is free then caller process locks it and continues, else caller process stops and waits until mutex gets freed.

Parameters

<i>mutex</i>	A mutex pointer.
--------------	------------------

Returns

BGRT_ST_OK on success, or error number.

4.4.3.5 bgrt_mtx_try_lock() `bgrt_st_t bgrt_mtx_try_lock (`
`bgrt_mtx_t * mutex)`

Try to lock a mutex.

If mutex is free then caller process locks it and continues, if not caller process continues without wait.

Parameters

<i>mutex</i>	A mutex pointer.
--------------	------------------

Returns

BGRT_ST_OK - if mutex was successfully locked else - BGRT_ST_ROLL.

4.5 bugurtos/libs/native/native.h File Reference

Native API.

```
#include <bugurt.h>
#include "ipc.h"
#include "sem.h"
#include "mutex.h"
#include "cond.h"
```

Macros

- #define [bgrt_proc_init_cs](#) bgrt_priv_proc_init
A process initialization from ISR or critical section.
- #define [bgrt_proc_run_cs](#) bgrt_priv_proc_run
Run a process from ISR or critical section.
- #define [bgrt_proc_run](#) BGRT_PROC_RUN
Run a process.
- #define [bgrt_proc_stop_cs](#) bgrt_priv_proc_stop
Stop a process from ISR or critical section.
- #define [bgrt_proc_stop](#) BGRT_PROC_STOP
Stop a process.
- #define [bgrt_proc_restart_cs](#) bgrt_priv_proc_restart
Restart a process from ISR.
- #define [bgrt_proc_restart](#) BGRT_PROC_RESTART
Restart a process.
- #define [bgrt_proc_self_stop](#) BGRT_PROC_SELF_STOP
Stop a caller process.
- #define [bgrt_proc_wd_reset](#) BGRT_PROC_RESET_WATCHDOG
Reset a process watchdog.
- #define [bgrt_proc_lock](#) BGRT_PROC_LOCK
Disable a process stop.
- #define [bgrt_proc_free](#) BGRT_PROC_FREE
Enable a process stop.
- #define [bgrt_proc_set_prio](#) BGRT_PROC_SET_PRIO
Set a process priority.

4.5.1 Detailed Description

Native API.

In this file there are definitions of functions, macros and data types. Also this file includes other native lib headers.

4.5.2 Macro Definition Documentation

4.5.2.1 [bgrt_proc_free](#) #define bgrt_proc_free BGRT_PROC_FREE

Enable a process stop.

4.5.2.2 bgrt_proc_init_cs `#define bgrt_proc_init_cs bgrt_priv_proc_init`

A process initialization from ISR or critical section.

4.5.2.3 bgrt_proc_lock `#define bgrt_proc_lock BGRT_PROC_LOCK`

Disable a process stop.

4.5.2.4 bgrt_proc_restart `#define bgrt_proc_restart BGRT_PROC_RESTART`

Restart a process.

4.5.2.5 bgrt_proc_restart_cs `#define bgrt_proc_restart_cs bgrt_priv_proc_restart`

Restart a process from ISR.

4.5.2.6 bgrt_proc_run `#define bgrt_proc_run BGRT_PROC_RUN`

Run a process.

4.5.2.7 bgrt_proc_run_cs `#define bgrt_proc_run_cs bgrt_priv_proc_run`

Run a process from ISR or critical section.

4.5.2.8 bgrt_proc_self_stop `#define bgrt_proc_self_stop BGRT_PROC_SELF_STOP`

Stop a caller process.

4.5.2.9 bgrt_proc_set_prio `#define bgrt_proc_set_prio BGRT_PROC_SET_PRIO`

Set a process priority.

4.5.2.10 bgrt_proc_stop `#define bgrt_proc_stop BGRT_PROC_STOP`

Stop a process.

4.5.2.11 bgrt_proc_stop_cs `#define bgrt_proc_stop_cs bgrt_priv_proc_stop`

Stop a process from ISR or critical section.

4.5.2.12 bgrt_proc_wd_reset `#define bgrt_proc_wd_reset BGRT_PROC_RESET_WATCHDOG`

Reset a process watchdog.

4.6 bugurtos/libs/native/sem.h File Reference

A counting semaphores header.

```
#include <bugurt.h>
```

Data Structures

- struct [bgrt_priv_sem_t](#)
A counting semaphore.

Typedefs

- typedef typedefBGRT_CDECL_BEGIN struct [bgrt_priv_sem_t](#) [bgrt_sem_t](#)

Functions

- `bgrt_st_t bgrt_sem_init_cs (bgrt_sem_t *sem, bgrt_cnt_t count)`
Semaphore initiation from ISR.
- `bgrt_st_t bgrt_sem_init (bgrt_sem_t *sem, bgrt_cnt_t count)`
Semaphore initiation.
- `bgrt_st_t bgrt_sem_lock (bgrt_sem_t *sem)`
A semaphore lock.
- `bgrt_st_t bgrt_sem_try_lock (bgrt_sem_t *sem)`
Try to lock a semaphore.
- `bgrt_st_t bgrt_sem_free (bgrt_sem_t *sem)`
Semaphore free.
- `bgrt_st_t bgrt_sem_free_cs (bgrt_sem_t *sem)`
Semaphore free. For ISR usage.

4.6.1 Detailed Description

A counting semaphores header.

4.6.2 Typedef Documentation

4.6.2.1 `bgrt_sem_t` `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_sem_t bgrt_sem_t`

See `bgrt_priv_sem_t`;

4.6.3 Function Documentation

4.6.3.1 `bgrt_sem_free()` `bgrt_st_t bgrt_sem_free (bgrt_sem_t * sem)`

Semaphore free.

If semaphore wait list is empty, then counter will be increased, else semaphore wait list head will be launched.

Parameters

<i>sem</i>	A <code>bgrt_sem_t</code> pointer.
------------	------------------------------------

Returns

BGRT_ST_OK on success, or error number.

4.6.3.2 `bgrt_sem_free_cs()` `bgrt_st_t bgrt_sem_free_cs (bgrt_sem_t * sem)`

Semaphore free. For ISR usage.

Warning

A semaphore must not have an owner

If semaphore wait list is empty, then counter will be increased, else semaphore wait list head will be launched.

Parameters

<i>sem</i>	A <code>bgrt_sem_t</code> pointer.
------------	------------------------------------

Returns

BGRT_ST_OK on success, or error number.

4.6.3.3 bgrt_sem_init() `bgrt_st_t bgrt_sem_init (`
`bgrt_sem_t * sem,`
`bgrt_cnt_t count)`

Semaphore initiation.

Parameters

<i>sem</i>	A <code>bgrt_sem_t</code> pointer.
<i>count</i>	A counter start value.

4.6.3.4 bgrt_sem_init_cs() `bgrt_st_t bgrt_sem_init_cs (`
`bgrt_sem_t * sem,`
`bgrt_cnt_t count)`

Semaphore initiation from ISR.

Parameters

<i>sem</i>	A <code>bgrt_sem_t</code> pointer.
<i>count</i>	A counter start value.

4.6.3.5 bgrt_sem_lock() `bgrt_st_t bgrt_sem_lock (`
`bgrt_sem_t * sem)`

A semaphore lock.

If semaphore counter greater than zero, then it will be decreased and caller process will continue, else caller process will stop and wait until semaphore get free.

Parameters

<i>sem</i>	A <code>bgrt_sem_t</code> pointer.
------------	------------------------------------

Returns

BGRT_ST_OK on success, or error number.

4.6.3.6 bgrt_sem_try_lock() `bgrt_st_t bgrt_sem_try_lock (`
`bgrt_sem_t * sem)`

Try to lock a semaphore.

If semaphore counter greater than zero, then it will be decreased and caller process will continue, else caller process will just continue.

Parameters

<i>sem</i>	A <code>bgrt_sem_t</code> pointer.
------------	------------------------------------

Returns

BGRT_ST_OK on success, or error number.

Index

bgrt_atm_bclr
 bugurt_port.h, 8

BGRT_ATM_BCLR_ISR
 bugurt_port.h, 6

bgrt_atm_bget
 bugurt_port.h, 9

BGRT_ATM_BGET_ISR
 bugurt_port.h, 7

bgrt_atm_bset
 bugurt_port.h, 9

BGRT_ATM_BSET_ISR
 bugurt_port.h, 7

bgrt_atm_init
 bugurt_port.h, 10

BGRT_ATM_INIT_ISR
 bugurt_port.h, 7

bgrt_cond_broadcast
 cond.h, 11

bgrt_cond_init
 cond.h, 11

bgrt_cond_init_cs
 cond.h, 11

bgrt_cond_signal
 cond.h, 13

bgrt_cond_t
 cond.h, 11

bgrt_cond_wait
 cond.h, 13

BGRT_CURR_PROC
 bugurt_port.h, 8

BGRT_INT_FREE
 bugurt_port.h, 8

BGRT_INT_LOCK
 bugurt_port.h, 8

bgrt_ipc_init
 ipc.h, 14

bgrt_ipc_init_cs
 ipc.h, 15

bgrt_ipc_reply
 ipc.h, 15

bgrt_ipc_send
 ipc.h, 15

bgrt_ipc_t
 ipc.h, 14

bgrt_ipc_wait
 ipc.h, 15

BGRT_ISR
 bugurt_port.h, 8

BGRT_KBLOCK
 bugurt_port.h, 8

bgrt_mtx_free
 mutex.h, 17

bgrt_mtx_init
 mutex.h, 17

bgrt_mtx_init_cs
 mutex.h, 17

bgrt_mtx_lock
 mutex.h, 18

bgrt_mtx_t
 mutex.h, 17

bgrt_mtx_try_lock
 mutex.h, 18

bgrt_priv_cond_t, 2
 wait, 3

bgrt_priv_ipc_t, 3
 msg, 3
 wait, 4

bgrt_priv_mtx_t, 4
 wait, 4

bgrt_priv_sem_t, 5
 counter, 5
 lock, 5
 wait, 5

bgrt_proc_free
 native.h, 19

bgrt_proc_init_cs
 native.h, 19

bgrt_proc_lock
 native.h, 20

bgrt_proc_restart
 native.h, 20

bgrt_proc_restart_cs
 native.h, 20

bgrt_proc_run
 native.h, 20

bgrt_proc_run_cs
 native.h, 20

bgrt_proc_self_stop
 native.h, 20

bgrt_proc_set_prio
 native.h, 20

bgrt_proc_stop
 native.h, 20

bgrt_proc_stop_cs
 native.h, 21

bgrt_proc_wd_reset
 native.h, 21

bgrt_sem_free
 sem.h, 22

bgrt_sem_free_cs
 sem.h, 22

bgrt_sem_init
 sem.h, 23

bgrt_sem_init_cs
 sem.h, 23

bgrt_sem_lock
 sem.h, 23

bgrt_sem_t
 sem.h, 22

bgrt_sem_try_lock

sem.h, 23
 bugurt_port.h
 bgrt_atm_bclr, 8
 BGRT_ATM_BCLR_ISR, 6
 bgrt_atm_bget, 9
 BGRT_ATM_BGET_ISR, 7
 bgrt_atm_bset, 9
 BGRT_ATM_BSET_ISR, 7
 bgrt_atm_init, 10
 BGRT_ATM_INIT_ISR, 7
 BGRT_CURR_PROC, 8
 BGRT_INT_FREE, 8
 BGRT_INT_LOCK, 8
 BGRT_ISR, 8
 BGRT_KBLOCK, 8
 bugurtos/doc/doxygen/bugurt_port.h, 6
 bugurtos/libs/native/cond.h, 10
 bugurtos/libs/native/ipc.h, 13
 bugurtos/libs/native/mutex.h, 16
 bugurtos/libs/native/native.h, 18
 bugurtos/libs/native/sem.h, 21

 cond.h
 bgrt_cond_broadcast, 11
 bgrt_cond_init, 11
 bgrt_cond_init_cs, 11
 bgrt_cond_signal, 13
 bgrt_cond_t, 11
 bgrt_cond_wait, 13
 counter
 bgrt_priv_sem_t, 5

 ipc.h
 bgrt_ipc_init, 14
 bgrt_ipc_init_cs, 15
 bgrt_ipc_reply, 15
 bgrt_ipc_send, 15
 bgrt_ipc_t, 14
 bgrt_ipc_wait, 15

 lock
 bgrt_priv_sem_t, 5

 msg
 bgrt_priv_ipc_t, 3
 mutex.h
 bgrt_mtx_free, 17
 bgrt_mtx_init, 17
 bgrt_mtx_init_cs, 17
 bgrt_mtx_lock, 18
 bgrt_mtx_t, 17
 bgrt_mtx_try_lock, 18

 native.h
 bgrt_proc_free, 19
 bgrt_proc_init_cs, 19
 bgrt_proc_lock, 20
 bgrt_proc_restart, 20
 bgrt_proc_restart_cs, 20
 bgrt_proc_run, 20
 bgrt_proc_run_cs, 20
 bgrt_proc_self_stop, 20
 bgrt_proc_set_prio, 20
 bgrt_proc_stop, 20
 bgrt_proc_stop_cs, 21
 bgrt_proc_wd_reset, 21

 sem.h
 bgrt_sem_free, 22
 bgrt_sem_free_cs, 22
 bgrt_sem_init, 23
 bgrt_sem_init_cs, 23
 bgrt_sem_lock, 23
 bgrt_sem_t, 22
 bgrt_sem_try_lock, 23

 wait
 bgrt_priv_cond_t, 3
 bgrt_priv_ipc_t, 4
 bgrt_priv_mtx_t, 4
 bgrt_priv_sem_t, 5