BuguRTOS native lib

4.1.0

Generated by Doxygen 1.8.17

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

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

map_ptr	A pointer to atomic map.
msk	A mask.

Returns

Last masked bits state.

4.1.1.2 BGRT_ATM_BGET_ISR `#define BGRT_ATM_BGET_ISR(`

```
map_ptr,
msk )
```

Read masked bits.

Warning

For ISR/crit_sec usage!

Parameters

map_ptr	A pointer to atomic map.
msk	A mask.

Returns

Masked vectors state.

4.1.1.3 BGRT_ATM_BSET_ISR `#define BGRT_ATM_BSET_ISR(`

```
map_ptr,
msk )
```

Set masked bits.

Parameters

map_ptr	A pointer to atomic map.
msk	A mask.

4.1.1.4 BGRT_ATM_INIT_ISR `#define BGRT_ATM_INIT_ISR(`

```
map_ptr )
```

Atomic map initialization.

Warning

For ISR/crit_sec usage!

Parameters

map_ptr	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

v	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

map_ptr	A pointer to atomic map.
msk	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

map_ptr	A pointer to atomic map.
msk	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

map_ptr	A pointer to atomic map.
msk	A mask.

4.1.2.4 bgrt_atm_init() `void bgrt_atm_init ( bgrt_map_t * map_ptr )`

Atomic map initialization.

Parameters

map_ptr	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

cond	A bgrt_cond_t 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

cond	A bgrt_cond_t 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

cond	A bgrt_cond_t 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

cond	A bgrt_cond_t 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

cond	A bgrt_cond_t pointer.
mutex	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

endpoint	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

endpoint	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

in	An IPC endpoint pointer.
pid	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 pinter to the message buffer through IPC. Senders are blocked on IPC endpoint and wait for their turn, receiver inherits senders priorities.

Parameters

out	An IPC endpoint pointer.
msg	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

in	An IPC endpoint pointer.
pid	A sender pid buffer pointer.
block	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

mutex	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

mutex	A mutex pointer.
prio	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

mutex	A mutex pointer.
prio	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

mutex	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

mutex	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

sem	A bgrt_sem_t 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

sem	A bgrt_sem_t 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

sem	A bgrt_sem_t pointer.
count	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

sem	A bgrt_sem_t pointer.
count	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

sem	A bgrt_sem_t 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

sem	A bgrt_sem_t pointer.

Returns

BGRT_ST_OK on success, or error number.

Index