

BuguRTOS native lib

4.1.0

Создано системой Doxygen 1.8.17

1	Алфавитный указатель структур данных	2
1.1	Структуры данных	2
2	Список файлов	2
2.1	Файлы	2
3	Структуры данных	2
3.1	Структура <code>bgrt_priv_cond_t</code>	2
3.1.1	Подробное описание	3
3.1.2	Поля	3
3.2	Структура <code>bgrt_priv_ipc_t</code>	3
3.2.1	Подробное описание	3
3.2.2	Поля	3
3.3	Структура <code>bgrt_priv_mtx_t</code>	4
3.3.1	Подробное описание	4
3.3.2	Поля	4
3.4	Структура <code>bgrt_priv_sem_t</code>	4
3.4.1	Подробное описание	5
3.4.2	Поля	5
4	Файлы	5
4.1	Файл <code>bugurtos/doc/doxygen/bugurt_port.h</code>	5
4.1.1	Макросы	6
4.1.2	Функции	8
4.2	Файл <code>bugurtos/libs/native/cond.h</code>	9
4.2.1	Подробное описание	10
4.2.2	Типы	10
4.2.3	Функции	10
4.3	Файл <code>bugurtos/libs/native/ipc.h</code>	12
4.3.1	Подробное описание	13
4.3.2	Типы	13
4.3.3	Функции	13
4.4	Файл <code>bugurtos/libs/native/mutex.h</code>	15
4.4.1	Подробное описание	15
4.4.2	Типы	15
4.4.3	Функции	15
4.5	Файл <code>bugurtos/libs/native/native.h</code>	17
4.5.1	Подробное описание	18
4.5.2	Макросы	18
4.6	Файл <code>bugurtos/libs/native/sem.h</code>	19
4.6.1	Подробное описание	20
4.6.2	Типы	20
4.6.3	Функции	20
	Предметный указатель	23

1 Алфавитный указатель структур данных

1.1 Структуры данных

Структуры данных с их кратким описанием.

bgrt_priv_cond_t	Условная переменная	2
bgrt_priv_ipc_t	Конечная точка IPC	3
bgrt_priv_mtx_t	Мьютекс	4
bgrt_priv_sem_t	Счётный семафор	4

2 Список файлов

2.1 Файлы

Полный список файлов.

bugurtos/doc/doxygen/bugurt_port.h		5
bugurtos/libs/native/cond.h	Заголовок условных переменных	9
bugurtos/libs/native/ipc.h	Заголовок IPC	12
bugurtos/libs/native/mutex.h	Заголовок мьютексов	15
bugurtos/libs/native/native.h	Нативный API	17
bugurtos/libs/native/sem.h	Заголовок счётных семафоров	19

3 Структуры данных

3.1 Структура `bgrt_priv_cond_t`

Условная переменная.

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

Поля данных

- `bgrt_sync_t` [wait](#)

3.1.1 Подробное описание

Условная переменная.

Условные переменные используются в сочетании с мьютексами для синхронизации процессов по событиям. Процесс может заблокировать себя на условной переменной. Другой процесс может возобновить выполнение 1 или всех процессов, заблокированных на условной переменной.

3.1.2 Поля

3.1.2.1 `wait` `bgrt_sync_t` `bgrt_priv_cond_t::wait`

Список ожидающих процессов.

Объявления и описания членов структуры находятся в файле:

- `bugurtos/libs/native/cond.h`

3.2 Структура `bgrt_priv_ipc_t`

Конечная точка IPC.

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

Поля данных

- `bgrt_sync_t` `wait`
- `void *` `msg`

3.2.1 Подробное описание

Конечная точка IPC.

Используется для реализации блокирующего синхронного или асинхронного протокола IPC.

3.2.2 Поля

3.2.2.1 `msg` `void*` `bgrt_priv_ipc_t::msg`

Указатель на буфер с сообщением.

3.2.2.2 wait bgrt_sync_t bgrt_priv_ipc_t::wait

Список ожидающих процессов.

Объявления и описания членов структуры находятся в файле:

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

3.3 Структура bgrt_priv_mtx_t

Мьютекс.

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

Поля данных

- bgrt_sync_t [wait](#)

3.3.1 Подробное описание

Мьютекс.

Используется для управления доступом к общим ресурсам, в тех случаях, когда общий ресурс нужен в течение долгого времени. Поддерживается произвольная вложенность мьютексов.

Предупреждения

Мьютексы захватываются и освобождаются только процессами. Нельзя делать это из обработчиков прерываний.

Мьютекс должен освободить **ИМЕННО ТОТ** процесс, который его захватил.

3.3.2 Поля

3.3.2.1 wait bgrt_sync_t bgrt_priv_mtx_t::wait

Список ожидающих процессов.

Объявления и описания членов структуры находятся в файле:

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

3.4 Структура bgrt_priv_sem_t

Счётный семафор.

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

Поля данных

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

3.4.1 Подробное описание

Счётный семафор.

Счётные семафоры используются для синхронизации процессов. Не рекомендуется их использовать для организации доступа к общим ресурсам, т.к. здесь нет управления приоритетами. Счётный семафор может быть захвачен 1 процессом, а освобождён другим.

3.4.2 Поля

3.4.2.1 `counter` `bgrt_cnt_t` `bgrt_priv_sem_t::counter`

Счётчик ресурсов.

3.4.2.2 `lock` `bgrt_lock_t` `bgrt_priv_sem_t::lock`

Спин-блокировка.

3.4.2.3 `wait` `bgrt_sync_t` `bgrt_priv_sem_t::wait`

Список ожидающих процессов.

Объявления и описания членов структуры находятся в файле:

- `bugurtos/libs/native/sem.h`

4 Файлы

4.1 Файл `bugurtos/doc/doxygen/bugurt_port.h`

Макросы

- `#define BGRT_INT_LOCK()`
Запретить прерывания.
- `#define BGRT_INT_FREE()`
Разрешить прерывания.
- `#define BGRT_KBLOCK`
Текущий блок ядра.
- `#define BGRT_CURR_PROC`
Текущий процесс.
- `#define BGRT_ISR(v)`
Шаблон обработчика прерывания.
- `#define BGRT_ATM_INIT_ISR(map_ptr)`
Инициализация атомарной карты.
- `#define BGRT_ATM_BSET_ISR(map_ptr, msk)`
Поставить биты в 1 по маске.
- `#define BGRT_ATM_BGET_ISR(map_ptr, msk)`
Считать биты по маске.
- `#define BGRT_ATM_BCLR_ISR(map_ptr, msk)`
Сбросить значения битов по маске.

Функции

- void `bgrt_atm_init` (`bgrt_map_t *map_ptr`)
Инициализация атомарной карты.
- void `bgrt_atm_bset` (`bgrt_map_t *map_ptr`, `bgrt_map_t msk`)
Поставить биты в 1 по маске.
- `bgrt_map_t bgrt_atm_bget` (`bgrt_map_t *map_ptr`, `bgrt_map_t msk`)
Считать биты по маске.
- `bgrt_map_t bgrt_atm_bclr` (`bgrt_map_t *map_ptr`, `bgrt_map_t msk`)
Сбросить биты по маске.

4.1.1 Макросы

4.1.1.1 `BGRT_ATM_BCLR_ISR` `#define BGRT_ATM_BCLR_ISR(
map_ptr,
msk)`

Сбросить значения битов по маске.

Предупреждения

Для вызова из обработчиков прерываний/критических секций!

Аргументы

<code>map_ptr</code>	Указатель на атомарную карту.
<code>msk</code>	Маска.

Возвращает

Последнее состояние интересующих битов.

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

Считать биты по маске.

Предупреждения

Для вызова из обработчиков прерываний/критических секций!

Аргументы

map_ptr	Указатель на атомарную карту.
msk	Маска.

Возвращает

Состояние векторов прерываний.

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

Поставить биты в 1 по маске.

Аргументы

map_ptr	Указатель на атомарную карту.
msk	Маска.

4.1.1.4 `BGRT_ATM_INIT_ISR` `#define BGRT_ATM_INIT_ISR(
map_ptr)`

Инициализация атомарной карты.

Предупреждения

Для вызова из обработчиков прерываний/критических секций!

Аргументы

map_ptr	Указатель на атомарную карту.
---------	-------------------------------

4.1.1.5 `BGRT_CURR_PROC` `#define BGRT_CURR_PROC`

Текущий процесс.

4.1.1.6 `BGRT_INT_FREE` `#define BGRT_INT_FREE()`

Разрешить прерывания.

4.1.1.7 BGRT_INT_LOCK #define BGRT_INT_LOCK()

Запретить прерывания.

4.1.1.8 BGRT_ISR #define BGRT_ISR(v)

Шаблон обработчика прерывания.

Аргументы

v	Идентификатор обработчика прерывания.
---	---------------------------------------

4.1.1.9 BGRT_KBLOCK #define BGRT_KBLOCK

Текущий блок ядра.

4.1.2 Функции

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

Сбросить биты по маске.

Аргументы

map_ptr	Указатель на атомарную карту.
msk	Маска.

Возвращает

Последнее состояние маскированных битов.

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

Считать биты по маске.

Аргументы

<code>map_ptr</code>	Указатель на атомарную карту.
<code>msk</code>	Маска.

Возвращает

Состояние векторов прерываний.

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

Поставить биты в 1 по маске.

Предупреждения

Для вызова из обработчиков прерываний/критических секций!

Аргументы

<code>map_ptr</code>	Указатель на атомарную карту.
<code>msk</code>	Маска.

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

Инициализация атомарной карты.

Аргументы

<code>map_ptr</code>	Указатель на атомарную карту.
----------------------	-------------------------------

4.2 Файл `bugurtos/libs/native/cond.h`

Заголовок условных переменных.

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

Структуры данных

- `struct bgrt_priv_cond_t`
Условная переменная.

Определения типов

- `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_cond_t bgrt_cond_t`

Функции

- `bgrt_st_t bgrt_cond_init_cs (bgrt_cond_t *cond)`
Инициализация условной переменной из обработчика прерывания или критической секции.
- `bgrt_st_t bgrt_cond_init (bgrt_cond_t *cond)`
Инициализация условной переменной.
- `bgrt_st_t bgrt_cond_wait (bgrt_cond_t *cond, bgrt_mtx_t *mutex)`
Встать в список ожидания условной переменной.
- `bgrt_st_t bgrt_cond_signal (bgrt_cond_t *cond)`
Возобновить работу 1 процесса ожидающего условной переменной.
- `bgrt_st_t bgrt_cond_broadcast (bgrt_cond_t *cond)`
Возобновить работу всех процессов ожидающих условной переменной.

4.2.1 Подробное описание

Заголовок условных переменных.

4.2.2 Типы

4.2.2.1 `bgrt_cond_t` `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_cond_t bgrt_cond_t`

Смотри `bgrt_priv_cond_t`;

4.2.3 Функции

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

Возобновить работу всех процессов ожидающих условной переменной.

Возобновляет работу всех процессов из списка ожидающих.

Предупреждения

Вызывать только при захваченном мьютексе!

Аргументы

<code>cond</code>	Указатель на условную переменную.
-------------------	-----------------------------------

Возвращает

`BGRT_ST_OK` в случае успеха, или номер ошибки.

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

Инициализация условной переменной.

Аргументы

<code>cond</code>	Указатель на условную переменную.
-------------------	-----------------------------------

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

Инициализация условной переменной из обработчика прерывания или критической секции.

Аргументы

<code>cond</code>	Указатель на условную переменную.
-------------------	-----------------------------------

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

Возобновить работу 1 процесса ожидающего условной переменной.

Возобновляет работу головы списка ожидающих процессов.

Предупреждения

Вызывать только при захваченном мьютексе!

Аргументы

<code>cond</code>	Указатель на условную переменную.
-------------------	-----------------------------------

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

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

Встать в список ожидания условной переменной.

Останавливает вызвавший процесс и ставит его в список ожидания.

Аргументы

cond	Указатель на условную переменную.
mutex	Указатель на мьютекс, защищающий условную переменную.

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

4.3 Файл `bugurtos/libs/native/ipc.h`

Заголовок IPC.

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

Структуры данных

- struct `bgrt_priv_ipc_t`
Конечная точка IPC.

Определения типов

- typedef typedefBGRT_CDECL_BEGIN struct `bgrt_priv_ipc_t` `bgrt_ipc_t`

Функции

- `bgrt_st_t bgrt_ipc_init_cs (bgrt_ipc_t *endpoint)`
Инициализация конечной точки IPC из критической секции или обработчика прерывания.
- `bgrt_st_t bgrt_ipc_init (bgrt_ipc_t *endpoint)`
Инициализация конечной точки IPC.
- `bgrt_st_t bgrt_ipc_send (bgrt_ipc_t *out, void *msg)`
Посылка данных процессу через IPC.
- `bgrt_st_t bgrt_ipc_wait (bgrt_ipc_t *in, BGRT_PID_T *pid, bgrt_flag_t block)`
Переход процесса к ожиданию получения данных через IPC.
- `bgrt_st_t bgrt_ipc_reply (bgrt_ipc_t *in, BGRT_PID_T pid)`
Разблокировать процесс, от которого получено сообщение.

4.3.1 Подробное описание

Заголовок IPC.

4.3.2 Типы

4.3.2.1 `bgrt_ipc_t` `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_ipc_t bgrt_ipc_t`

Смотри `bgrt_priv_ipc_t`;

4.3.3 Функции

4.3.3.1 `bgrt_ipc_init()` `bgrt_st_t bgrt_ipc_init (`
`bgrt_ipc_t * endpoint)`

Инициализация конечной точки IPC.

Аргументы

<code>endpoint</code>	Указатель на конечную точку.
-----------------------	------------------------------

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

Инициализация конечной точки IPC из критической секции или обработчика прерывания.

Аргументы

<code>endpoint</code>	Указатель на конечную точку.
-----------------------	------------------------------

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

Разблокировать процесс, от которого получено сообщение.

Аргументы

in	Указатель на конечную точку IPC.
pid	Идентификатор отправителя.

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

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

Посылка данных процессу через IPC.

Пересылает указатель на буфер с сообщением через IPC, отправители блокируются и ждут своей очереди, получатель наследует приоритеты отправителей.

Аргументы

out	Указатель на конечную точку IPC.
msg	Указатель на буфер с сообщением.

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

```
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 )
```

Переход процесса к ожиданию получения данных через IPC.

Для указания отправителя или получения указателя на отправитель используется буфер, адрес которого передаётся вторым аргументом.

Аргументы

in	Указатель на конечную точку IPC.
pid	Указатель на буфер идентификатора отправителя.
block	Флаг блокировки вызывающего процесса, если не 0, то вызывающий процесс блокируется до готовности сообщения.

Возвращает

`BGRT_ST_OK` в случае успеха, или номер ошибки.

4.4 Файл `bugurtos/libs/native/mutex.h`

Заголовок мьютексов.

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

Структуры данных

- `struct bgrt_priv_mtx_t`
Мьютекс.

Определения типов

- `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_mtx_t bgrt_mtx_t`

Функции

- `bgrt_st_t bgrt_mtx_init_cs (bgrt_mtx_t *mutex, bgrt_prio_t prio)`
Инициализация мьютекса из критической секции, или обработчика прерываний.
- `bgrt_st_t bgrt_mtx_init (bgrt_mtx_t *mutex, bgrt_prio_t prio)`
Инициализация мьютекса.
- `bgrt_st_t bgrt_mtx_try_lock (bgrt_mtx_t *mutex)`
Попытка захвата мьютекса.
- `bgrt_st_t bgrt_mtx_lock (bgrt_mtx_t *mutex)`
Захват мьютекса.
- `bgrt_st_t bgrt_mtx_free (bgrt_mtx_t *mutex)`
Освобождение мьютекса.

4.4.1 Подробное описание

Заголовок мьютексов.

4.4.2 Типы

4.4.2.1 `bgrt_mtx_t` `typedef typedefBGRT_CDECL_BEGIN struct bgrt_priv_mtx_t bgrt_mtx_t`

Смотри `bgrt_priv_mtx_t`;

4.4.3 Функции

4.4.3.1 `bgrt_mtx_free()` `bgrt_st_t bgrt_mtx_free (`
`bgrt_mtx_t * mutex)`

Освобождение мьютекса.

Если список ожидающих процессов пуст - вызывающий процесс освобождает мьютекс, если список не пуст - ставит на выполнение голову списка. Также происходит обработка флагов, при необходимости вызывающий процесс останавливается.

Аргументы

mutex	Указатель на мьютекс.
-------	-----------------------

Возвращает

BGRТ_ST_OK в случае успеха, или номер ошибки.

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

Инициализация мьютекса.

Аргументы

mutex	Указатель на мьютекс.
prio	Приоритет мьютекса.

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

Инициализация мьютекса из критической секции, или обработчика прерываний.

Да, инициировать из обработчика прерывания можно!

Аргументы

mutex	Указатель на мьютекс.
prio	Приоритет мьютекса.

4.4.3.4 bgrt_mtx_lock() bgrt_st_t bgrt_mtx_lock (
 bgrt_mtx_t * mutex)

Захват мьютекса.

Если мьютекс свободен - процесс захватывает его и продолжает выполняться, если уже занят - процесс останавливается и записывается в список ожидающих.

Аргументы

mutex	Указатель на мьютекс.
-------	-----------------------

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

4.4.3.5 bgrt_mtx_try_lock() bgrt_st_t bgrt_mtx_try_lock (
 bgrt_mtx_t * mutex)

Попытка захвата мьютекса.

Если мьютекс свободен - процесс захватывает его и продолжает выполняться, если уже занят - процесс продолжает выполнение.

Аргументы

mutex	Указатель на мьютекс.
-------	-----------------------

Возвращает

BGRT_ST_OK - если удалось захватить, BGRT_ST_ROLL - если не удалось.

4.5 Файл bugurtos/libs/native/native.h

Нативный API.

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

Макросы

- #define bgrt_proc_init_cs bgrt_priv_proc_init
Инициация процесса из обработчика прерывания или критической секции.
- #define bgrt_proc_run_cs bgrt_priv_proc_run
Запуск процесса из обработчика прерывания или критической секции.
- #define bgrt_proc_run BGRT_PROC_RUN
Запуск процесса.
- #define bgrt_proc_stop_cs bgrt_priv_proc_stop
Останов процесса из обработчика прерывания или критической секции.
- #define bgrt_proc_stop BGRT_PROC_STOP
Останов процесса.
- #define bgrt_proc_restart_cs bgrt_priv_proc_restart
Перезапустить процесс из обработчика прерывания.
- #define bgrt_proc_restart BGRT_PROC_RESTART
Перезапустить процесс.
- #define bgrt_proc_self_stop BGRT_PROC_SELF_STOP

- Останов вызывающего процесса.
• `#define bgrt_proc_wd_reset BGRT_PROC_RESET_WATCHDOG`
Сброс вачдог.
- `#define bgrt_proc_lock BGRT_PROC_LOCK`
Запретить останов процесса.
- `#define bgrt_proc_free BGRT_PROC_FREE`
Разрешить останов процесса.
- `#define bgrt_proc_set_prio BGRT_PROC_SET_PRIO`
Установить приоритет процесса.

4.5.1 Подробное описание

Нативный API.

В этом файле определены имена функций, макросов и типов данных, а так же включены другие заголовки.

4.5.2 Макросы

4.5.2.1 `bgrt_proc_free` `#define bgrt_proc_free BGRT_PROC_FREE`

Разрешить останов процесса.

4.5.2.2 `bgrt_proc_init_cs` `#define bgrt_proc_init_cs bgrt_priv_proc_init`

Инициация процесса из обработчика прерывания или критической секции.

4.5.2.3 `bgrt_proc_lock` `#define bgrt_proc_lock BGRT_PROC_LOCK`

Запретить останов процесса.

4.5.2.4 `bgrt_proc_restart` `#define bgrt_proc_restart BGRT_PROC_RESTART`

Перезапустить процесс.

4.5.2.5 `bgrt_proc_restart_cs` `#define bgrt_proc_restart_cs bgrt_priv_proc_restart`

Перезапустить процесс из обработчика прерывания.

4.5.2.6 `bgrt_proc_run` `#define bgrt_proc_run BGRT_PROC_RUN`

Запуск процесса.

4.5.2.7 `bgrt_proc_run_cs` `#define bgrt_proc_run_cs bgrt_priv_proc_run`

Запуск процесса из обработчика прерывания или критической секции.

4.5.2.8 `bgrt_proc_self_stop` `#define bgrt_proc_self_stop BGRT_PROC_SELF_STOP`

Останов вызывающего процесса.

4.5.2.9 `bgrt_proc_set_prio` `#define bgrt_proc_set_prio BGRT_PROC_SET_PRIO`

Установить приоритет процесса.

4.5.2.10 `bgrt_proc_stop` `#define bgrt_proc_stop BGRT_PROC_STOP`

Останов процесса.

4.5.2.11 `bgrt_proc_stop_cs` `#define bgrt_proc_stop_cs bgrt_priv_proc_stop`

Останов процесса из обработчика прерывания или критической секции.

4.5.2.12 `bgrt_proc_wd_reset` `#define bgrt_proc_wd_reset BGRT_PROC_RESET_WATCHDOG`

Сброс вачдог.

4.6 Файл `bugurtos/libs/native/sem.h`

Заголовок счётных семафоров.

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

Структуры данных

- struct `bgrt_priv_sem_t`
Счётный семафор.

Определения типов

- typedef typedefBGRT_CDECL_BEGIN struct `bgrt_priv_sem_t` `bgrt_sem_t`

Функции

- `bgrt_st_t bgrt_sem_init_cs (bgrt_sem_t *sem, bgrt_cnt_t count)`
Инициализация семафора из обработчика прерывания или критической секции.
- `bgrt_st_t bgrt_sem_init (bgrt_sem_t *sem, bgrt_cnt_t count)`
Инициализация семафора.
- `bgrt_st_t bgrt_sem_lock (bgrt_sem_t *sem)`
Захват семафора.
- `bgrt_st_t bgrt_sem_try_lock (bgrt_sem_t *sem)`
Попытка захвата семафора.
- `bgrt_st_t bgrt_sem_free (bgrt_sem_t *sem)`
Освобождение семафора.
- `bgrt_st_t bgrt_sem_free_cs (bgrt_sem_t *sem)`
Освобождение семафора из обработчика прерывания.

4.6.1 Подробное описание

Заголовок счётных семафоров.

4.6.2 Типы

4.6.2.1 `bgrt_sem_t` typedef typedefBGRT_CDECL_BEGIN struct `bgrt_priv_sem_t` `bgrt_sem_t`

Смотри `bgrt_priv_sem_t`;

4.6.3 Функции

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

Освобождение семафора.

Если список ожидающих захвата семафора пуст, то счётчик семафора увеличиваем на 1. Если не пуст - возобновляем работу головы списка.

Аргументы

sem	Указатель на семафор.
-----	-----------------------

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

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

Освобождение семафора из обработчика прерывания.

Предупреждения

У семафора не должно быть хозяина!!!

Если список ожидающих захвата семафора пуст, то счётчик семафора увеличиваем на 1. Если не пуст - возобновляем работу головы списка.

Аргументы

sem	Указатель на семафор.
-----	-----------------------

Возвращает

BGRT_ST_OK в случае успеха, или номер ошибки.

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

Инициализация семафора.

Аргументы

sem	Указатель на семафор.
count	Начальное значение счётчика.

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

Инициализация семафора из обработчика прерывания или критической секции.

Аргументы

sem	Указатель на семафор.
count	Начальное значение счётчика.

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

Захват семафора.

Если значение счётчика семафора больше 0, то процесс уменьшает счётчик семафора на 1 и продолжает выполняться. Если значение счётчика семафора равно 0, процесс останавливается и встаёт в список ожидающих освобождения семафора.

Аргументы

sem	Указатель на семафор.
-----	-----------------------

Возвращает

`BGRT_ST_OK` в случае успеха, или номер ошибки.

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

Попытка захвата семафора.

Если значение счётчика семафора больше 0, то процесс уменьшает счётчик семафора на 1 и продолжает выполняться. Если значение счётчика семафора равно 0, процесс просто продолжает выполняться.

Аргументы

sem	Указатель на семафор.
-----	-----------------------

Возвращает

`BGRT_ST_OK` в случае успеха, или номер ошибки.

Предметный указатель

bgrt_atm_bclr
 bugurt_port.h, 8
BGRT_ATM_BCLR_ISR
 bugurt_port.h, 6
bgrt_atm_bget
 bugurt_port.h, 8
BGRT_ATM_BGET_ISR
 bugurt_port.h, 6
bgrt_atm_bset
 bugurt_port.h, 9
BGRT_ATM_BSET_ISR
 bugurt_port.h, 7
bgrt_atm_init
 bugurt_port.h, 9
BGRT_ATM_INIT_ISR
 bugurt_port.h, 7
bgrt_cond_broadcast
 cond.h, 10
bgrt_cond_init
 cond.h, 11
bgrt_cond_init_cs
 cond.h, 11
bgrt_cond_signal
 cond.h, 11
bgrt_cond_t
 cond.h, 10
bgrt_cond_wait
 cond.h, 12
BGRT_CURR_PROC
 bugurt_port.h, 7
BGRT_INT_FREE
 bugurt_port.h, 7
BGRT_INT_LOCK
 bugurt_port.h, 7
bgrt_ipc_init
 ipc.h, 13
bgrt_ipc_init_cs
 ipc.h, 13
bgrt_ipc_reply
 ipc.h, 13
bgrt_ipc_send
 ipc.h, 14
bgrt_ipc_t
 ipc.h, 13
bgrt_ipc_wait
 ipc.h, 14
BGRT_ISR
 bugurt_port.h, 8
BGRT_KBLOCK
 bugurt_port.h, 8
bgrt_mtx_free
 mutex.h, 15
bgrt_mtx_init
 mutex.h, 16
bgrt_mtx_init_cs
 mutex.h, 16
bgrt_mtx_lock
 mutex.h, 16
bgrt_mtx_t
 mutex.h, 15
bgrt_mtx_try_lock
 mutex.h, 17
bgrt_priv_cond_t, 2
 wait, 3
bgrt_priv_ipc_t, 3
 msg, 3
 wait, 3
bgrt_priv_mtx_t, 4
 wait, 4
bgrt_priv_sem_t, 4
 counter, 5
 lock, 5
 wait, 5
bgrt_proc_free
 native.h, 18
bgrt_proc_init_cs
 native.h, 18
bgrt_proc_lock
 native.h, 18
bgrt_proc_restart
 native.h, 18
bgrt_proc_restart_cs
 native.h, 18
bgrt_proc_run
 native.h, 18
bgrt_proc_run_cs
 native.h, 19
bgrt_proc_self_stop
 native.h, 19
bgrt_proc_set_prio
 native.h, 19
bgrt_proc_stop
 native.h, 19
bgrt_proc_stop_cs
 native.h, 19
bgrt_proc_wd_reset
 native.h, 19
bgrt_sem_free
 sem.h, 20
bgrt_sem_free_cs
 sem.h, 21
bgrt_sem_init
 sem.h, 21
bgrt_sem_init_cs
 sem.h, 21
bgrt_sem_lock
 sem.h, 22
bgrt_sem_t
 sem.h, 20
bgrt_sem_try_lock

- sem.h, 22
- bugurt_port.h
 - bgrt_atm_bclr, 8
 - BGRT_ATM_BCLR_ISR, 6
 - bgrt_atm_bget, 8
 - BGRT_ATM_BGET_ISR, 6
 - bgrt_atm_bset, 9
 - BGRT_ATM_BSET_ISR, 7
 - bgrt_atm_init, 9
 - BGRT_ATM_INIT_ISR, 7
 - BGRT_CURR_PROC, 7
 - BGRT_INT_FREE, 7
 - BGRT_INT_LOCK, 7
 - BGRT_ISR, 8
 - BGRT_KBLOCK, 8
- bugurtos/doc/doxygen/bugurt_port.h, 5
- bugurtos/libs/native/cond.h, 9
- bugurtos/libs/native/ipc.h, 12
- bugurtos/libs/native/mutex.h, 15
- bugurtos/libs/native/native.h, 17
- bugurtos/libs/native/sem.h, 19
- cond.h
 - bgrt_cond_broadcast, 10
 - bgrt_cond_init, 11
 - bgrt_cond_init_cs, 11
 - bgrt_cond_signal, 11
 - bgrt_cond_t, 10
 - bgrt_cond_wait, 12
- counter
 - bgrt_priv_sem_t, 5
- ipc.h
 - bgrt_ipc_init, 13
 - bgrt_ipc_init_cs, 13
 - bgrt_ipc_reply, 13
 - bgrt_ipc_send, 14
 - bgrt_ipc_t, 13
 - bgrt_ipc_wait, 14
- lock
 - bgrt_priv_sem_t, 5
- msg
 - bgrt_priv_ipc_t, 3
- mutex.h
 - bgrt_mtx_free, 15
 - bgrt_mtx_init, 16
 - bgrt_mtx_init_cs, 16
 - bgrt_mtx_lock, 16
 - bgrt_mtx_t, 15
 - bgrt_mtx_try_lock, 17
- native.h
 - bgrt_proc_free, 18
 - bgrt_proc_init_cs, 18
 - bgrt_proc_lock, 18
 - bgrt_proc_restart, 18
 - bgrt_proc_restart_cs, 18
 - bgrt_proc_run, 18
 - bgrt_proc_run_cs, 19
 - bgrt_proc_self_stop, 19
 - bgrt_proc_set_prio, 19
 - bgrt_proc_stop, 19
 - bgrt_proc_stop_cs, 19
 - bgrt_proc_wd_reset, 19
- sem.h
 - bgrt_sem_free, 20
 - bgrt_sem_free_cs, 21
 - bgrt_sem_init, 21
 - bgrt_sem_init_cs, 21
 - bgrt_sem_lock, 22
 - bgrt_sem_t, 20
 - bgrt_sem_try_lock, 22
- wait
 - bgrt_priv_cond_t, 3
 - bgrt_priv_ipc_t, 3
 - bgrt_priv_mtx_t, 4
 - bgrt_priv_sem_t, 5