async_select

User Manual

Reference:	TLT-XXX-MAN-async_select-async_select
Version:	2.0.2
Revision:	XXX

Confidentiality & Intellectual Property

Table of Contents

Chapter 1. Data Structure Documentation

1.1. async_select_Request struct Reference

1.1.1. Data Fields

- int32_t fd
- int32_t java_thread_id
- int64_t absolute_timeout_ms
- SELECT_Operation operation
- struct async_select_Request * next

An asynchronous select request.

Detailed Description

Sanity check between the expected version of the configuration and the actual version of the configuration. If an error is raised here, it means that a new version of the CCO has been installed and the configuration async_select_configuration.h must be updated based on the one provided by the new CCO version.

Definition at line 45 of file async_select.c

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

- async_select.c

1.1.2. Field Documentation

Chapter 2. File Documentation

2.1. async_select.h File Reference

```
#include <stdint.h>
```

```
#include <sni.h>
```

2.1.1. Enumerations

- enum SELECT_Operation {
  SELECT_READ,
  SELECT_WRITE
  }

  Select operations list.

2.1.2. Functions

- int32_t non_blocking_select (int32_t fd, SELECT_Operation operation)

  Execute a select() for the given file descriptor and operation without blocking.

- int32_t async_select (int32_t fd, SELECT_Operation operation, int64_t timeout_ms, SNI_callback callback)

  Executes asynchronously a select() operation for the given file descriptor. This function will suspend the execution of the current Java thread using SNI_suspendCurrentJavaThreadWithCallback(). Once the select() succeeds the Java thread is resumed and the given SNI callback is called.

- int32_t async_select_init (void)

  Initialize the async_select component. This function must be called prior to any call of async_select().

- void async_select_notify_closed_fd (int32_t fd)

  Notifies the async_select task that a file descriptor has been closed. On some systems the close of a file descriptor does not unblock the select that's why we need to notify the async_select task.

Detailed Description

Asynchronous network select API.

Author: . MicroEJ Developer Team

Version: . 2.0.2

Date: . 13 November 2020

Definition in file C:/Jenkins/workspace/M0172_CCO-Async-Select/bsp-async_select/target~/ccomponentWorking/bsp/net/inc/async_select.h

2.2. async_select_configuration.h File Reference

```
#include <stdint.h>
```

```
#include <sni.h>
```

2.2.1. Macros

- #define ASYNC_SELECT_CONFIGURATION_VERSION (2)

  Compatibility sanity check value. This define value is checked in the implementation to validate that the version of this configuration is compatible with the implementation.

- #define MAX_NB_ASYNC_SELECT (16)

  Maximum number of asynchronous select that can be done at the same moment.

- #define ASYNC_SELECT_TASK_STACK_SIZE (2048)

  async_select task stack size in bytes.

- #define ASYNC_SELECT_TASK_NAME "AsyncSelect"

  async_select task name.

- #define ASYNC_SELECT_TASK_PRIORITY (12)

  async_select task priority.

- #define ASYNC_SELECT_MUTEX_NAME "AsyncSelectMutex"

  async_select mutex name.

- #define ASYNC_SELECT_POLLING_MODE_TIMEOUT_MS (100)

  Timeout in milliseconds used when the async_select task cannot allocate a socket for notifications.

- #define ASYNC_SELECT_CLOSE_UNBLOCK_SELECT

  Set this define if a file descriptor close unblocks the select.

Detailed Description

Asynchronous network select configuration.

Author: . MicroEJ Developer Team

Version: . 2.0.2

Date: . 13 November 2020

Definition in file C:/Jenkins/workspace/M0172_CCO-Async-Select/bsp-async_select/target~/ccomponentWorking/bsp/net/inc/async_select_configuration.h

2.3. async_select.c File Reference

```
#include "async_select.h"
```

```
#include "async_select_configuration.h"
```

```
#include <string.h>
```

```
#include <sys/socket.h>
```

```
#include <sys/select.h>
```

```
#include <netinet/in.h>
```

```
#include <stdbool.h>
```

```
#include <unistd.h>
```

```
#include "LLNET_Common.h"
```

2.3.1. Data Structures

- struct async_select_Request

  An asynchronous select request.

2.3.2. Macros

- #define async_select_get_current_time_ms LLMJVM_IMPL_getCurrentTime__Z(1)

2.3.3. Typedefs

- typedef struct async_select_Request async_select_Request

  An asynchronous select request.

2.3.4. Variables

- static async_select_Request all_requests

  Pool of requests. Used to reserve MAX_NB_ASYNC_SELECT async select requests.

- static async_select_Request * free_requests_fifo

  Linked-list of free requests that can be allocated using async_select_allocate_request().

- static async_select_Request * used_requests_fifo

  Linked-list of used requests.

- static fd_set read_fds

  File descriptor set for SELECT_READ requests.

- static fd_set write_fds

  File descriptor set for SELECT_WRITE requests.

- static volatile int32_t notify_fd_cache

  Used to unblock select() function call.

- static volatile uint8_t async_select_fifo_initialized

  set to one once the FIFOs are initialized.

2.3.5. Functions

- void async_select_lock (void)

  Enter critical section for the async_select component.

- void async_select_unlock (void)

  Exit critical section for the async_select component.

- int64_t LLMJVM_IMPL_getCurrentTime__Z (uint8_t system)

External function used to retrieve currentTime (defined in LLMJVM)

- static void async_select_do_select (void)

  Executes the select() operation for the file descriptors referenced by the received requests.

- static void async_select_update_notified_requests (void)

  After the execution of the select() operation, update the status of the requests that have been notified by the select() or have reached the timemout.

- static int32_t async_select_get_notify_fd (void)

  Returns the file descriptor created just to unlock the select() when we want to notify the async_select task that a new request has been sent.

- static async_select_Request * async_select_allocate_request (void)

  Find a free request and returns it. The returned request is not put it in the used requests FIFO. It must be either put in the used requests FIFO using async_select_send_new_request() or put back in the free requests FIFO on error using async_select_free_unused_request().

- static async_select_Request * async_select_free_used_request (async_select_Request * request, async_select_Request * previous_request_in_used_fifo)

  Remove the given request from the used FIFO and put it in the free FIFO.

- static void async_select_free_unused_request (async_select_Request * request)

  Put the given request in the free FIFO. The request must not be in the used FIFO.

- static int32_t async_select_send_new_request (async_select_Request * request)

  Notifies the async_select task that a new request must be managed.

- static void async_select_notify_select (void)

  Unlock the select operation.

- void async_select_request_fifo_init (void)

  Initializes the requests FIFOs. This function must be called prior to any call of async_select(). It can be called several times.

- int32_t non_blocking_select (int32_t fd, SELECT_Operation operation)

  Execute a select() for the given file descriptor and operation without blocking.

- int32_t async_select (int32_t fd, SELECT_Operation operation, int64_t timeout_ms, SNI_callback callback)

  Executes asynchronously a select() operation for the given file descriptor.

- void async_select_notify_closed_fd (int32_t fd)

Notifies the async_select task that a file descriptor has been closed. On some systems the close of a file descriptor does not unblock the select that's why we need to notify the async_select task.

- void async_select_task_main ( )

  The entry point for the async_select task. This function must be called from a dedicated task.

Detailed Description

Asynchronous network select implementation.

Author: .   MicroEJ Developer Team

Version: .   2.0.2

Date: .   13 November 2020

Definition in file C:/Jenkins/workspace/M0172_CCO-Async-Select/bsp-async_select/target~/ccomponentWorking/bsp/net/src/async_select.c

2.4. async_select_osal.c File Reference

```
#include "async_select.h"
```

```
#include "async_select_configuration.h"
```

```
#include "osal.h"
```

```
#include <stddef.h>
```

2.4.1. Variables

- static OSAL_task_handle_t async_select_task

  async_select OS task.

- static OSAL_mutex_handle_t async_select_mutex

  Mutex used for critical sections.

2.4.2. Functions

- void async_select_request_fifo_init (void)

Initializes the requests FIFOs. This function must be called prior to any call of async_select(). It can be called several times.

- void async_select_task_main (void)

  The entry point for the async_select task. This function must be called from a dedicated task.

- void async_select_lock (void)

  Enter critical section for the async_select component.

- void async_select_unlock (void)

  Exit critical section for the async_select component.

- static int32_t async_select_start_task (void)

  Start RTOS task and init RTOS specific structures.

- OSAL_task_stack_declare (async_select_task_stack , ASYNC_SELECT_TASK_STACK_SIZE)

  Stack of the async_select task.

- int32_t async_select_init ( )

  Initialize the async_select component. This function must be called prior to any call of async_select().

Detailed Description

Asynchronous network select implementation over OSAL API.

Author: .   MicroEJ Developer Team

Version: .   2.0.2

Date: .   13 November 2020

Definition in file C:/Jenkins/workspace/M0172_CCO-Async-Select/bsp-async_select/target~/ccomponentWorking/bsp/net/src/async_select_osal.c