EPMUX

1.0.0

Создано системой Doxygen 1.8.13

Содержание

1 Алфавитный указатель структур данных

1.1 Структуры данных

Структуры данных с их кратким описанием.

2 Список файлов

2.1 Файлы

Полный список файлов.

3 Структуры данных

3.1 Структура epmux_channel_for_line_a_t

#include <epmux.h>

Поля данных

- uint8_t ModuleNumber
- uint8_t ChannelNumber
- uint8_t Reserved [14]

3.1.1 Поля

3.1.1.1 ChannelNumber

uint8_t ChannelNumber

Номер канала в пределах модуля (номер контакта в разъёме). Нумерация начинается с единицы. 0 –отключить все каналы. Диапазон допустимых значений: 0 – 64.

3.1.1.2 ModuleNumber

uint8_t ModuleNumber

Порядковый номер модуля в цепочке устройств. Нумерация начинается с единицы. 0 –отключить все каналы.

3.1.1.3 Reserved

uint8_t Reserved[14]

Значение данного поля не должно использоваться в прикладном ПО. Для обеспечения совместимости с другими устройствами не изменяйте значение этого поля.

Объявления и описания членов структуры находятся в файле:

- epmux.h

3.2 Структура epmux_channel_for_line_b_t

#include <epmux.h>

Поля данных

- uint8_t ModuleNumber
- uint8_t ChannelNumber
- uint8_t Reserved [14]

3.2.1 Поля

3.2.1.1 ChannelNumber

uint8_t ChannelNumber

Номер канала в пределах модуля (номер контакта в разъёме). Нумерация начинается с единицы. 0 –отключить все каналы. Диапазон допустимых значений: 0 – 64.

3.2.1.2 ModuleNumber

uint8_t ModuleNumber

Порядковый номер модуля в цепочке устройств. Нумерация начинается с единицы. 0 –отключить все каналы.

3.2.1.3 Reserved

uint8_t Reserved[14]

Значение данного поля не должно использоваться в прикладном ПО. Для обеспечения совместимости с другими устройствами не изменяйте значение этого поля.

Объявления и описания членов структуры находятся в файле:

- epmux.h

3.3 Структура epmux_get_chain_structure_t

#include <epmux.h>

Поля данных

- uint8_t ChainLength
- uint8_t ChainStructure [16]
- uint8_t Reserved [16]

3.3.1 Поля

3.3.1.1 ChainLength

uint8_t ChainLength

Длина цепочки устройств. Длина цепочки равна количеству подключенных модулей.

3.3.1.2 ChainStructure

uint8_t ChainStructure[16]

Структура цепочки устройств. Показывает, какого типа модули на каких позициях расположены. Типы модулей: 1 – модуль типа A, поддерживает подключение каналов только к линии A; 2 – модуль типа AB, поддерживает подключение каналов как к линии A, так и к линии B; 0 – используется для заполнения оставшейся части массива, реальным модулям не соответствует.

3.3.1.3 Reserved

uint8_t Reserved[16]

Значение данного поля не должно использоваться в прикладном ПО. Для обеспечения совместимости с другими устройствами не изменяйте значение этого поля.

Объявления и описания членов структуры находятся в файле:

- epmux.h

3.4 Структура epmux_get_identity_information_t

#include <epmux.h>

Поля данных

- uint8_t Manufacturer [16]
- uint8_t ProductName [16]
- uint8_t ControllerName [16]
- uint8_t HardwareMajor
- uint8_t HardwareMinor
- uint16_t HardwareBugfix
- uint8_t BootloaderMajor
- uint8_t BootloaderMinor
- uint16_t BootloaderBugfix
- uint8_t FirmwareMajor
- uint8_t FirmwareMinor
- uint16_t FirmwareBugfix
- uint32_t SerialNumber
- uint8_t Reserved [8]

3.4.1 Поля

3.4.1.1 BootloaderBugfix

uint16_t BootloaderBugfix

Номер релиза версии загрузчика.

3.4.1.2 BootloaderMajor

uint8_t BootloaderMajor

Мажорный номер версии загрузчика.

3.4.1.3 BootloaderMinor

uint8_t BootloaderMinor

Минорный номер версии загрузчика.

3.4.1.4 ControllerName

uint8_t ControllerName[16]

Пользовательское имя контроллера. Может быть установлено пользователем с помощью отдельной команды.

3.4.1.5 FirmwareBugfix

uint16_t FirmwareBugfix

Номер релиза версии прошивки.

3.4.1.6 FirmwareMajor

uint8_t FirmwareMajor

Мажорный номер версии прошивки.

3.4.1.7 FirmwareMinor

uint8_t FirmwareMinor

Минорный номер версии прошивки.

3.4.1.8 HardwareBugfix

uint16_t HardwareBugfix

Номер правок этой версии железа.

3.4.1.9 HardwareMajor

uint8_t HardwareMajor

Основной номер версии железа.

3.4.1.10 HardwareMinor

uint8_t HardwareMinor

Второстепенный номер версии железа.

3.4.1.11 Manufacturer

uint8_t Manufacturer[16]

Имя производителя. Устанавливается производителем.

3.4.1.12 ProductName

uint8_t ProductName[16]

Название продукта. Устанавливается производителем.

3.4.1.13 Reserved

uint8_t Reserved[8]

3.4.1.14 SerialNumber

uint32_t SerialNumber

Серийный номер изделия.

Объявления и описания членов структуры находятся в файле:

- epmux.h

4 Файлы

4.1 Файл epmux.h

epmux API

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

Структуры данных

- struct epmux_get_identity_information_t
- struct epmux_get_chain_structure_t
- struct epmux_channel_for_line_a_t
- struct epmux_channel_for_line_b_t

Макросы

- #define EPMUX_BUILDER_VERSION_MAJOR 0
- #define EPMUX_BUILDER_VERSION_MINOR 10
- #define EPMUX_BUILDER_VERSION_BUGFIX 12
- #define EPMUX_BUILDER_VERSION_SUFFIX ""
- #define EPMUX_BUILDER_VERSION "0.10.12"
- #define EPMUX_URPC_API_EXPORT __attribute__((visibility("default")))
- #define EPMUX_URPC_CALLING_CONVENTION
- #define device_undefined (-1)
- #define result_ok 0
- #define result_error (-1)
- #define result_not_implemented (-2)
- #define result_value_error (-3)
- #define result_nodevice (-4)
- #define EPMUX_MAX_CHAIN_LENGTH 0x10
- #define EPMUX_MODULE_TYPE_A 0x1
- #define EPMUX_MODULE_TYPE_AB 0x2
- #define EPMUX_NO_MODULE 0x0

Уровень логирования

- #define LOGLEVEL_ERROR 0x01
- #define LOGLEVEL_WARNING 0x02
- #define LOGLEVEL_INFO 0x03
- #define LOGLEVEL_DEBUG 0x04

Определения типов

- typedef int device_t
- typedef int result_t
- typedef void(EPMUX_URPC_CALLING_CONVENTION * epmux_logging_callback_t) (int loglevel, const wchar_t *message, void *user_data)

Функции

- EPMUX_URPC_API_EXPORT void EPMUX_URPC_CALLING_CONVENTION epmux_logging_callback_stderr_wide (int loglevel, const wchar_t *message, void *user_data)
- EPMUX_URPC_API_EXPORT void EPMUX_URPC_CALLING_CONVENTION epmux_logging_callback_stderr_narrow (int loglevel, const wchar_t *message, void *user_data)
- EPMUX_URPC_API_EXPORT void EPMUX_URPC_CALLING_CONVENTION epmux_set_logging_callback (epmux_logging_callback_t cb, void *data)
- EPMUX_URPC_API_EXPORT device_t EPMUX_URPC_CALLING_CONVENTION epmux_open_device (const char *uri)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_libversion (char *lib_version)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_save_settings (device_t handle)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_read_settings (device_t handle)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_identity_information (device_t handle, epmux_get_identity_information_t *output)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_reset (device_t handle)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_chain_structure (device_t handle, epmux_get_chain_structure_t *output)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_all_channels_off (device_t handle)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_channel_for_line_a (device_t handle, epmux_channel_for_line_a_t *output)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_set_channel_for_line_a (device_t handle, epmux_channel_for_line_a_t *input)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_channel_for_line_b (device_t handle, epmux_channel_for_line_b_t *output)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_set_channel_for_line_b (device_t handle, epmux_channel_for_line_b_t *input)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_close_device (device_t *handle_ptr)
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_profile (device_t handle, char **buffer, void *(*allocate)(size_t))
- EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_set_profile (device_t handle, char *buffer)

4.1.1 Подробное описание

epmux API

4.1.2 Макросы

4.1.2.1 device_undefined

#define device_undefined (-1)

4.1.2.2 EPMUX_BUILDER_VERSION

#define EPMUX_BUILDER_VERSION "0.10.12"

4.1.2.3 EPMUX_BUILDER_VERSION_BUGFIX

#define EPMUX_BUILDER_VERSION_BUGFIX 12

4.1.2.4 EPMUX_BUILDER_VERSION_MAJOR

#define EPMUX_BUILDER_VERSION_MAJOR 0

4.1.2.5 EPMUX_BUILDER_VERSION_MINOR

#define EPMUX_BUILDER_VERSION_MINOR 10

4.1.2.6 EPMUX_BUILDER_VERSION_SUFFIX

#define EPMUX_BUILDER_VERSION_SUFFIX ""

4.1.2.7 EPMUX_MAX_CHAIN_LENGTH

#define EPMUX_MAX_CHAIN_LENGTH 0x10

4.1.2.8 EPMUX_MODULE_TYPE_A

#define EPMUX_MODULE_TYPE_A 0x1

4.1.2.9 EPMUX_MODULE_TYPE_AB

#define EPMUX_MODULE_TYPE_AB 0x2

4.1.2.10 EPMUX_NO_MODULE

```
#define EPMUX_NO_MODULE 0x0
```

4.1.2.11 EPMUX_URPC_API_EXPORT

```
#define EPMUX_URPC_API_EXPORT __attribute__((visibility("default")))
```

4.1.2.12 EPMUX_URPC_CALLING_CONVENTION

```
#define EPMUX_URPC_CALLING_CONVENTION
```

4.1.2.13 LOGLEVEL_DEBUG

```
#define LOGLEVEL_DEBUG 0x04
```

Уровень логирования - отладка

4.1.2.14 LOGLEVEL_ERROR

```
#define LOGLEVEL_ERROR 0x01
```

Уровень логирования - ошибка

4.1.2.15 LOGLEVEL_INFO

```
#define LOGLEVEL_INFO 0x03
```

Уровень логирования - информация

4.1.2.16 LOGLEVEL_WARNING

```
#define LOGLEVEL_WARNING 0x02
```

Уровень логирования - предупреждение

4.1.2.17 result_error

```
#define result_error (-1)
```

4.1.2.18 result_nodevice

```
#define result_nodevice (-4)
```

4.1.2.19 result_not_implemented

#define result_not_implemented (-2)

4.1.2.20 result_ok

#define result_ok 0

4.1.2.21 result_value_error

#define result_value_error (-3)

4.1.3 Типы

4.1.3.1 device_t

typedef int device_t

4.1.3.2 epmux_logging_callback_t

typedef void(EPMUX_URPC_CALLING_CONVENTION * epmux_logging_callback_t) (int loglevel, const wchar_t *message, void *user_data)

Прототип функции обратного вызова для логирования.

Аргументы

loglevel	- Уровень логирования.
message	- Сообщение.

4.1.3.3 result_t

typedef int result_t

4.1.4 Функции

4.1.4.1 epmux_all_channels_off()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_all_channels_off (
device_t handle)

Выключает все каналы всех модулей.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().

4.1.4.2 epmux_close_device()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_close_device (
device_t * handle_ptr)

Закрывает устройство.

Аргументы

handle_ptr	- Идентификатор устройства.

4.1.4.3 epmux_get_chain_structure()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_chain_structure (
device_t handle,
epmux_get_chain_structure_t * output)

Автоматическое определение / обновление структуры цепочки подключенных модулей. Позволяет узнать количество подключенных модулей, их тип и взаимное расположение в цепочке устройств. Данная команда используется не только для сбора информации, но и для автоматической конфигурации мультиплексора. Поэтому при обновлении конфигурации устройств «на горячую» (подключении / отключении новых модулей) необходимо вызвать данную команду перед установкой каналов. В противном случае возможны ошибки, связанные с неправильной нумерацией модулей.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().
out	output	- Данные, получаемые с устройства.

4.1.4.4 epmux_get_channel_for_line_a()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_channel_for_↩
line_a (

```
device_t handle,
epmux_channel_for_line_a_t * output )
```

Активный канал линии A (основная линия для вывода сигнала). Канал определяется порядковым номером модуля в цепочке и номером канала в пределах модуля (номер контакта в разъёме). Линия может быть подключена ровно к одному из каналов. При подключении нового канала, старый канал будет отключен автоматически. При изменении конфигурации оборудования «на горячую» (отключение / подключение модулей) перед установкой канала нужно обновить конфигурацию цепочки устройств с помощью команды get_chain_structure.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().
out	output	- Данные, получаемые с устройства.

4.1.4.5 epmux_get_channel_for_line_b()

```
EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_channel_for_↩
line_b (
device_t handle,
epmux_channel_for_line_b_t * output )
```

Активный канал линии B. Канал определяется порядковым номером модуля в цепочке и номером канала в пределах модуля (номер контакта в разъёме). Линия может быть подключена ровно к одному из каналов. При подключении нового канала, старый канал будет отключен автоматически. Линия B поддерживается не на всех модулях. Узнать, какие модули в текущей конфигурации цепочки поддерживают подключение каналов к линии B, можно с помощью команды get_chain↩ _structure. При изменении конфигурации оборудования «на горячую» (отключение / подключение модулей) перед установкой канала нужно обновить конфигурацию цепочки устройств с помощью команды get_chain_structure.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().
out	output	- Данные, получаемые с устройства.

4.1.4.6 epmux_get_identity_information()

```
EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_identity_↩
information (
device_t handle,
epmux_get_identity_information_t * output )
```

Возвращает идентификационную информацию об устройстве, такую как номера версий прошивки и серийный номер. Эта информация удобна для поиска нужного устройства среди списка доступных. Может быть вызвана как из прошивки, так и из бутлоадера.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().
out	output	- Данные, получаемые с устройства.

4.1.4.7 epmux_get_profile()

```
EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_get_profile (
        device_t handle,
        char ** buffer,
        void *(*)(size_t) allocate )
```

Загружает профиль с устройства.

Аргументы

in	handle	- Идентификатор устройства.
out	buffer	- Адрес указателя на выходной буфер. Память для указателя на char* должна быть выделена.
out	allocate	- Функция для выделения памяти.

4.1.4.8 epmux_libversion()

```
EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_libversion (
        char * lib_version )
```

Версия библиотеки.

Аргументы

out	lib_version	- Версия библиотеки.

4.1.4.9 epmux_logging_callback_stderr_narrow()

```
EPMUX_URPC_API_EXPORT void EPMUX_URPC_CALLING_CONVENTION epmux_logging_callback_↩
stderr_narrow (
        int loglevel,
        const wchar_t * message,
        void * user_data )
```

Простая функция логирования на stderr в узких (однобайтных) символах.

Аргументы

loglevel	- Уровень логирования.
message	- Сообщение.

4.1.4.10 epmux_logging_callback_stderr_wide()

EPMUX_URPC_API_EXPORT void EPMUX_URPC_CALLING_CONVENTION epmux_logging_callback_↩
stderr_wide (
int loglevel,
const wchar_t * message,
void * user_data)

Простая функция логирования на stderr в широких символах.

Аргументы

loglevel	- Уровень логирования.
message	- Сообщение.

4.1.4.11 epmux_open_device()

EPMUX_URPC_API_EXPORT device_t EPMUX_URPC_CALLING_CONVENTION epmux_open_device (
const char * uri)

Открывает устройство по имени name и возвращает идентификатор устройства.

Аргументы

in	name	- Имя устройства. Имя устройства имеет вид "com:port" или xi-net://host/serial или udp://host:port. Для COM устройства "port" это имя устройства в ОС. Например "com:\\.\COM3" (Windows) или "com:///dev/tty/ttyACM34" (Linux/Mac). Для сетевого (XiNet) устройства "host" это IPv4 адрес или полностью определённое имя домена, "serial" это серийный номер устройства в шестнадцатеричной системе. Например "xi-net://192.168.0.1/00001234" или "xi-net://hostname.com/89ABCDEF". Для ethernet переходника com-udp "host" это IPv4 адрес переходника, "port" это порт переходника. Например "udp://192.168.0.2:1024" Замечание: в один момент времени COM устройство может использоваться только одной программой. Если при открытии устройства возникают ошибки, нужно убедиться, что COM-порт есть в системе и что это устройство в данный момент не используется другими программами

4.1.4.12 epmux_read_settings()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_read_settings (
device_t handle)

Чтение всех настроек контроллера из flash памяти в оперативную, заменяя текущие настройки.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().

4.1.4.13 epmux_reset()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_reset (
device_t handle)

Команда для перезагрузки контроллера эквивалентная перезагрузке по питанию. В нормальной практике использоваться не должна.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().

4.1.4.14 epmux_save_settings()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_save_settings (
device_t handle)

При получении команды контроллер выполняет операцию сохранения текущих настроек во встроенную энергонезависимую память контроллера.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().

4.1.4.15 epmux_set_channel_for_line_a()

EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_set_channel_for_↩
line_a (
device_t handle,
epmux_channel_for_line_a_t * input)

Активный канал линии A (основная линия для вывода сигнала). Канал определяется порядковым номером модуля в цепочке и номером канала в пределах модуля (номер контакта в разъёме). Линия может быть подключена ровно к одному из каналов. При подключении нового канала, старый канал будет отключен автоматически. При изменении конфигурации оборудования «на горячую» (отключение / подключение модулей) перед установкой канала нужно обновить конфигурацию цепочки устройств с помощью команды get_chain_structure.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().
in	input	- Данные, отправляемые устройству.

4.1.4.16 epmux_set_channel_for_line_b()

```
EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_set_channel_for_↩
line_b (
        device_t handle,
        epmux_channel_for_line_b_t * input )
```

Активный канал линии B. Канал определяется порядковым номером модуля в цепочке и номером канала в пределах модуля (номер контакта в разъёме). Линия может быть подключена ровно к одному из каналов. При подключении нового канала, старый канал будет отключен автоматически. Линия B поддерживается не на всех модулях. Узнать, какие модули в текущей конфигурации цепочки поддерживают подключение каналов к линии B, можно с помощью команды get_chain↩ _structure. При изменении конфигурации оборудования «на горячую» (отключение / подключение модулей) перед установкой канала нужно обновить конфигурацию цепочки устройств с помощью команды get_chain_structure.

Аргументы

in	handle	- Идентификатор устройства, полученный от epmux_open_device().
in	input	- Данные, отправляемые устройству.

4.1.4.17 epmux_set_logging_callback()

```
EPMUX_URPC_API_EXPORT void EPMUX_URPC_CALLING_CONVENTION epmux_set_logging_callback (
        epmux_logging_callback_t cb,
        void * data )
```

Устанавливает функцию обратного вызова для логирования. Передача NULL в качестве аргумента отключает логирование.

Аргументы

logging_callback	указатель на функцию обратного вызова

4.1.4.18 epmux_set_profile()

```
EPMUX_URPC_API_EXPORT result_t EPMUX_URPC_CALLING_CONVENTION epmux_set_profile (
        device_t handle,
        char * buffer )
```

Загружает профиль с устройства.

Аргументы

in	handle	- Идентификатор устройства.
in	buffer	- Входной буфер, откуда будет считан профиль.

Предметный указатель