Руководство (v1.1)

По работе с драйвером модуля “mPCIe–1553RT2”

Интерфейс ГОСТ Р 52070-2003
(MIL-STD-1553B)

Для драйверов версии 3.3 и ниже

ОС LINUX

24.02.2015

ООО “Новомар” 2015 г.

Оглавление

1. Введение.

Драйвер модуля “**mPCIe – 1553RT2**” поддерживает одновременную работу не более 100 устройств и присваивает им уникальные символьные имена вида “man_dev_x”, где x — индекс устройства, начиная с 0. Устройства находятся в папке “/dev” .

Каждый модуль представляет отдельный файл устройства в файловой системе.

Обращение к независимым каналам модуля осуществляется по номеру канала (0, 1..).

Взаимодействие с драйвером происходит посредством ioctl-команд, перечень которых находится в файле “ioctl_man.h”.

2. Установка драйвера.

1. Создайте папку /modules в корне файловой системы.
2. Поместите в созданную папку каталог man_drv, содержащую файлы drvman.ko, version.txt и ioctl_man.h.
3. Откройте файл /etc/rc.local текстовым редактором.
4. Добавьте в открытый файл перед строкой “exit 0” строку “insmod /modules/man_drv/drvman.ko”.
5. Перезагрузите компьютер.

Если Вы хотите проверить, загружен ли драйвер в ядро:

1. В терминале введите команду “lsmod”.
2. Найдите в выведенном списке “drvman”.
3. Если есть — все нормально, если нет — драйвер не работает.

Для обновления версии драйвера:

1. Посмотреть версию драйвера (прикладывается в файле version.txt в папке с драйвером) .
2. Сравнить с версией текущего драйвера (файл /modules/man_drv/version.txt).
3. Выбрать более позднюю.
4. Заменить файлы drvman.ko и version.txt в папке /modules/man_drv на новые (сохраняя имена!).
5. Перезагрузить компьютер.

3. Подключение файла с командами к разрабатываемому проекту.

1. Скопировать файл “ioctl_man.h” из папки /modules/man_drv в папку с проектом.
2. В начале проекта добавить строку:

```
#include «ioctl_man.h»
```

3. Использовать команды.

Формат описан ниже; примеры использования приложены.

4. Список доступных команд по версиям драйверов.

Название вызова	Краткое описание
Список вызовов, доступных **в драйвере версии до 3.3 и ниже**	
IOCTL_ENABLE_DMA	Разрешение работы DMA
IOCTL_DISABLE_DMA	Запрет работы DMA
IOCTL_RESET_INDEX_DMA	Сброс счётчика DMA
IOCTL_GET_NBLOCK_RAW_DMA	Кол-во новых блоков данных в буфере DMA
IOCTL_RD_RAW_DMA	Чтение буфера DMA
IOCTL_ENABLE_CANONICAL_DMA	Включение режима “CANONICAL” для DMA
IOCTL_DISABLE_CANONICAL_DMA	Выключение режима “CANONICAL” для DMA
IOCTL_VERSION	Информация о плате.
IOCTL_SET_ADDRESS	Установка адреса ОУ
IOCTL_ENABLE_CHANNEL	Разрешение работы ОУ
IOCTL_DISABLE_CHANNEL	Запрет работы ОУ
IOCTL_SET_TIMER_TIMEOUT	Установка разрешения таймера
IOCTL_RD_CH_NBLOCK_BEGIN_DMA	Чтение буфера DMA
IOCTL_ENABLE_CANONICAL_CH_MEM	Включение режима “CANONICAL” для передачи
IOCTL_DISABLE_CANONICAL_CH_MEM	Выключение режима “CANONICAL” для передачи
IOCTL_WR_REG_OFFSET	Запись в регистры
IOCTL_RD_REG_OFFSET	Чтение из регистров
IOCTL_WR_REG	Запись в регистры
IOCTL_RD_REG	Чтение из регистров
IOCTL_ENABLE_PADDR	Разрешение подадреса
IOCTL_DISABLE_PADDR	Запрет подадреса
IOCTL_WR_REG_PA	Запись в регистр подадреса
IOCTL_RD_REG_PA	Чтение из регистра подадреса
IOCTL_WR_BLOCK_BUF_PA	Запись в буфер передачи
IOCTL_RD_BLOCK_BUF_PA	Чтение из буфера передачи
IOCTL_DISABLE_INT_ALLCH_HDAT	Отключить прерывание HDAT
IOCTL_DISABLE_INT_CH_HDAT	Отключить прерывание HDAT
IOCTL_DISABLE_INT_ALLCH_QDAT	Отключить прерывание QDAT

IOCTL_DISABLE_INT_CH_QDAT	Отключить прерывание QDAT
IOCTL_DISABLE_INT_ALLCH_FLASH	Отключить прерывание FLASH
IOCTL_DISABLE_INT_CH_FLASH	Отключить прерывание FLASH
IOCTL_DISABLE_INT_CH_MC	Отключить прерывание MC
IOCTL_DISABLE_INT_CH_ERR	Отключить прерывание ERR
IOCTL_ENABLE_INT_CH_IO_SA_PID_SIG	Включение прерываний и назначение сигналов.

5. Описание использования команд драйвера.

Структура ioctl-команды:

int ioctl(int fd, IOCTL_..., unsigned long param),
где:
int fd – дескриптор файла, полученный при вызове функции open для файла устройства;
IOCTL_... – команда из набора, описанного в файле ioctl_man.h;
unsigned long param – параметр, передаваемый с командой. Содержимое параметра зависит от команды (см. описание команд ниже).

Команда, в случае удачного выполнения запроса, возвращает 0.
В случае неудачного выполнения запроса возвращается значение, отличное от 0, что означает:
1. запрос был отклонен ОС (-EBUSY и т.п.);
2. –EACCES - запрос не был выполнен драйвером (команда отсутствует, не может быть выполнена в режиме CANONICAL, некорректные параметры для данной платы/команды).

При загрузке драйвера для платы и всех каналов включается режим CANONICAL.
Режим реализован программно в драйвере.
В данном режиме недоступны команды (см. описание команд ниже), способные оказать деструктивное влияние на работу в данном режиме.
В режиме CANONICAL обеспечиваются следующие два блока функциональности (запрещаемые/разрешаемые независимо):

1. Независимое чтение информации из буфера DMA по каналам (режим включается/выключается для всех каналов платы одновременно):
   - чтение полученной информации с одного канала платы не оказывает влияние на другие каналы;
   - при чтении с канала пользователь получает данные только этого канала.

   При включении/выключении режима указатель буфера DMA платы обнуляется.
2. “Простая” работа с буферами передачи (режим включается/выключается для каждого канала независимо).
   В этом блоке реализована следующая функциональность:
   - для всех подадресов канала включается режим использования одного буфера на передачу;
   - при записи в буфер передачи данные становятся доступны для многократного считывания со стороны контроллера шины (КШ);
   - обеспечивается целостность доступных со стороны КШ данных.

   Обращение по записи/чтению возможно только к буферу 0 любого подадреса.

После загрузки драйвера запрещена работа устройства, работа DMA, все прерывания, все подадреса, все командные слова, указатель DMA сброшен в 0, во все подадреса передачи записаны 0.

6. Команды, оказывающие влияние на все каналы платы.

6.1. IOCTL_ENABLE_DMA

Назначение:

Начать (продолжить) работу DMA платы.

Входные параметры:

param – не используется

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_ENABLE_DMA, 0)) {
//ошибка
}
else {
// DMA разрешен
}
```

6.2. IOCTL_DISABLE_DMA

Назначение:

Остановить работу DMA платы.

Входные параметры:

param – не используется

недоступна в режиме CANONICAL

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_DISABLE_DMA, 0)) {
//ошибка
}
else {
// DMA запрещен
}
```

6.3. IOCTL_RESET_INDEX_DMA

Назначение:

Сбросить указатель данных буфера DMA платы. Режим работы DMA не изменяется.

Входные параметры:

param – не используется

недоступна в режиме CANONICAL

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_RESET_INDEX_DMA, 0)) {
//ошибка
}
else {
// указатель DMA сброшен
}
```

6.4. IOCTL_GET_NBLOCK_RAW_DMA

Назначение:

Получить кол-во блоков данных, накопленных в буфере DMA платы по всем каналам (блок - 128 байт).

Входные параметры:

param – указатель на переменную типа unsigned int, в которую будет помещено кол-во блоков

недоступна в режиме CANONICAL

Пример вызова:

```
#include “ioctl_man.h”
unsigned int nb;
if (ioctl(fd,  IOCTL_GET_NBLOCK_RAW_DMA, &nb)) {
//ошибка
}
else {
// nb = кол-во непрочитанных блоков в буфере DMA
}
```

6.5. IOCTL_RD_RAW_DMA

Назначение:

Считать из буфера DMA кол-во блоков DMA_STR.number_block, поместив блоки в массив по указателю DMA_STR.buf.
Значение в DMA_STR.channel не используется.
После выполнения команды указатель буфера DMA сдвигается на соответствующую кол-ву прочитанного позицию.

Входные параметры:

Param - переменная типа DMA_STR.

недоступна в режиме CANONICAL

Пример вызова:

```
#include “ioctl_man.h”
DMA_STR d;
d.number_block; // Кол-во запрашиваемых/полученных блоков
d.number_channel;
if (ioctl(fd,  IOCTL_RD_RAW_DMA, &d)) {
//ошибка
}
else {
// d.number_block = кол-во блоков, скопированных из буфера
DMA (может быть меньше
//запрошенных!)
//d.buf – скопированная из буфера DMA информация
}
```

6.6. IOCTL_ENABLE_CANONICAL_DMA

Назначение:

Включение режима “CANONICAL” работы с DMA.

Входные параметры:

param – не используется

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd, IOCTL_ENABLE_CANONICAL_DMA, 0)) {
//ошибка
}
else {
// Режим CANONICAL DMA включен, указатель буфера DMA
сброшен
}
```

6.7. IOCTL_DISABLE_CANONICAL_DMA

Назначение:

Выключение режима “ CANONICAL ” работы с DMA.

Входные параметры:

param – не используется

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd, IOCTL_DISABLE_CANONICAL_DMA, 0)) {
//ошибка
}
else {
// Режим CANONICAL DMA выключен, указатель буфера DMA
сброшен
}
```

6.8. IOCTL_VERSION

Назначение:

Запрос информации о плате.

Входные параметры:

param – переменная типа VERSION

Пример вызова:

```
#include “ioctl_man.h”
VERSION ver;
if (ioctl(fd, IOCTL_VERSION, &ver)) {
//ошибка
}
else {
// структура ver содержит информацию о плате
}
```

7. Команды работы с каналом.

7.1. IOCTL_SET_ADDRESS

Назначение:

Установка адреса ОУ AD_STR.ad (0…31) в канале AD_STR.channel (0…N).

Входные параметры:

param – переменная типа AD_STR

Пример вызова:

```
#include “ioctl_man.h”
AD_STR sad;
sad.channel = 0;
sad.ad = 3;
if (ioctl(fd,  IOCTL_SET_ADDRESS, &sad)) {
//ошибка
}
else {
// В канале 0 установлен адрес 3
}
```

7.2. IOCTL_ENABLE_CHANNEL

Назначение:

Разрешение работы ОУ канала param.

Входные параметры:

param – номер канала

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_ENABLE_CHANNEL, 0)) {
//ошибка
}
else {
// канал 0 разрешен
}
```

7.3. IOCTL_DISABLE_CHANNEL

Назначение:

Запрет работы ОУ канала param.

Входные параметры:

param – номер канала

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_DISABLE_CHANNEL, 0)) {
//ошибка
}
else {
// канал 0 запрещен }
```

7.4. IOCTL_SET_TIMER_TIMEOUT

Назначение:

Установка разрешения таймера TT.timer (1,2,4,8,16,32,64) и таймаута TT.timeout (0 – 17 мкс, 1 – 110 мкс) приема канала TT.channel.

Входные параметры:

Param – переменная типа TT

Пример вызова:

```
#include “ioctl_man.h”
TT tit;
tit.channel = 0;
tit.timer = 4;
tit.timeout = 0;
if (ioctl(fd,  IOCTL_SET_TIMER_TIMEOUT, &tit)) {
//ошибка
}
else {
// В канале 0 установлено разрешение таймера 4 мкс, таймаут 17 мкс
}
```

7.5. IOCTL_RD_CH_NBLOCK_BEGIN_DMA

Назначение:

Считать из буфера DMA кол-во блоков DMA_STR.number_block канала DMA_STR.channel, поместив блоки в массив по указателю DMA_STR.buf.
После выполнения команды указатель буфера DMA соответствующего канала сдвигается на соответствующую кол-ву прочитанного позицию.

Входные параметры:

param – указатель на структуру типа DMA_STR
доступна только в режиме CANONICAL

Пример вызова:

```
#include “ioctl_man.h”
struct {
unsigned int number_block; // Кол-во запрашиваемых/полученных блоков
unsigned int number_channel; // Номер канала
char b[20*128]; // Место под 20 блоков
} d;
d.number_block = 20;
d.channel = 1;
if (ioctl(fd,  IOCTL_RD_RAW_DMA, &d)) {
//ошибка
}
else {
// d.number_block = кол-во блоков, скопированных из буфера DMA (может быть меньше
//запрошенных!) канала 1
//d.buf0]- d.buf d.number_block*128] – скопированная из буфера DMA информация   }
```

7.6. IOCTL_ENABLE_CANONICAL_CH_MEM

Назначение:

Разрешить режим “CANONICAL” работы с буферами передачи для канала param.

Входные параметры:

param – номер канала

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_ENABLE_CANONICAL_CH_MEM, 0)) {
//ошибка
}
else {
//Режим CANONICAL MEM  для канала 0 включен
}
```

7.7. IOCTL_DISABLE_CANONICAL_CH_MEM

Назначение:

Запретить режим CANONICAL работы с буферами передачи для канала param.

Входные параметры:

param – номер канала

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_DISABLE_CANONICAL_CH_MEM, 0)) {
//ошибка
}
else {
//Режим CANONICAL MEM  для канала 0 выключен (режим буферов не изменен)
}
```

8. Команды доступа к регистрам платы.

8.1. IOCTL_WR_REG_OFFSET

Назначение:

Запись в регистр по смещению SADDR_DATA.daddr относительно базового адреса значения SADDR_DATA.data.

ВНИМАНИЕ.

Рекомендуется использовать только опытным пользователям.

При работе с данной командой необходимо учитывать возможность влияния на работу всех каналов платы.

Использование ошибочного смещения может привести к неисправности платы. Вывод в регистры не контролируется (возможно нарушение некорректной работы в режиме CANONICAL).

Входные параметры:

param – указатель на структуру типа SADDR_DATA

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA sad;
sad.daddr = 0x2000;
sad.data = 0;
// поля buf1 и channel структуры SADDR_DATA в данной
команде не используются
if (ioctl(fd,  IOCTL_WR_REG_OFFSET, &sad)) {
//ошибка
}
else {
// Значение 0 записано в регистр 0x2000
}
```

8.2. IOCTL_RD_REG_OFFSET

Назначение:

Чтение из регистра по смещению SADDR_DATA.daddr относительно базового адреса значения и копирование его в SADDR_DATA.data.

Входные параметры:

param – указатель на структуру типа SADDR_DATA

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA sad;
sad.daddr = 0x2000;
// поля buf1 и channel структуры SADDR_DATA в данной
команде не используются
if (ioctl(fd,  IOCTL_RD_REG_OFFSET, &sad)) {
//ошибка
}
else {
// В sad.data – содержимое регистра 0x2000
}
```

9. Команды доступа к регистрам канала.

9.1. IOCTL_WR_REG

Назначение:

Запись в регистр SADDR_DATA.daddr канала SADDR_DATA.channel значения SADDR_DATA.data.

Входные параметры:

param – указатель на структуру типа SADDR_DATA

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA sad;
sad.daddr = MIL_FREE_TIMER;
sad.data = 0;
sad.channel = 1;
// поле buf1 структуры SADDR_DATA в данной команде не используются
if (ioctl(fd,  IOCTL_WR_REG, &sad)) {
//ошибка
}
else {
// Значение 0 записано в регистр MIL_FREE_TIMER канала 1 (таймер сброшен)
}
```

9.2. IOCTL_RD_REG

Назначение:

Чтение из регистра SADDR_DATA.daddr канала SADDR_DATA.channel значения и копирование его в SADDR_DATA.data.

Входные параметры:

param – указатель на структуру типа SADDR_DATA

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA sad;
sad.daddr = MIL_FREE_TIMER;
sad.channel = 0;
// поле buf1 структуры SADDR_DATA в данной команде не используются
if (ioctl(fd,  IOCTL_RD_REG, &sad)) {
//ошибка
}
else {
// sad.data = текущее значение таймера  канала 0
}
```

9.3. IOCTL_ENABLE_PADDR

Назначение:

Разрешение подадреса номер SADDR_DATA_PA.paddr приема/передачи SADDR_DATA_PA.daddr (*MIL_RT_RCV_REG/ MIL_RT_TR_REG*) канала SADDR_DATA_PA.channel

Входные параметры:

param – указатель на структуру типа SADDR_DATA_PA.

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA_PA sadpa;
sadpa.daddr = MIL_RT_RCV_REG;
sadpa.paddr = 3;
sadpa.channel = 1;
// поле data  структуры SADDR_DATA в данной команде не используются
if (ioctl(fd,  IOCTL_ENABLE_PADDR, &sadpa)) {
//ошибка
}
else {
// Подадрес 3 на прием канала 1 разрешен
}
```

9.4. IOCTL_DISABLE_PADDR

Назначение:

Запрешение подадреса номер SADDR_DATA_PA.paddr приема/передачи SADDR_DATA_PA.daddr (*MIL_RT_RCV_REG/ MIL_RT_TR_REG*) канала SADDR_DATA_PA.channel

Входные параметры:

param – указатель на структуру типа SADDR_DATA_PA.

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA_PA sadpa;
sadpa.daddr = MIL_RT_TR_REG;
sadpa.paddr = 5;
sadpa.channel = 0;
// поле data  структуры SADDR_DATA в данной команде не используются
if (ioctl(fd,  IOCTL_DISABLE_PADDR, &sadpa)) {
//ошибка
}
else {
// Подадрес 5 на передачу канала 0 запрещен
}
```

9.5. IOCTL_WR_REG_PA

Назначение:

Запись в регистр SADDR_DATA_PA.daddr (*MIL_RT_RCV_REG, MIL_RT_TR_REG*) подадреса SADDR_DATA_PA.paddr канала SADDR_DATA_PA.channel значения SADDR_DATA_PA.data.

Входные параметры:

param – указатель на структуру типа SADDR_DATA_PA.

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA_PA sadpa;
sadpa.daddr = MIL_RT_RCV_REG;
sadpa.paddr = 3;
sadpa.data = 0;
sadpa.channel = 1;
// поле buf1 структуры SADDR_DATA в данной команде не используются
if (ioctl(fd,  IOCTL_WR_REG_PA, &sadpa)) {
//ошибка
}
else {
// Значение 0 записано в регистр MIL_RT_RCV_REG подадреса 3 канала 1
}
```

9.6. IOCTL_RD_REG_PA

Назначение:

Чтение из регистра SADDR_DATA_PA.daddr (*MIL_RT_RCV_REG, MIL_RT_TR_REG*) подадреса SADDR_DATA_PA.paddr канала SADDR_DATA_PA.channel значения и копирование его в SADDR_DATA_PA.data.

Входные параметры:

param – указатель на структуру типа SADDR_DATA_PA.

Пример вызова:

```
#include “ioctl_man.h”
SADDR_DATA_PA sadpa;
sadpa.daddr = MIL_RT_TR_REG;
sadpa.paddr = 7;
sadpa.channel = 0;
// поле buf1 структуры SADDR_DATA в данной команде не используются
if (ioctl(fd,  IOCTL_RD_REG_PA, &sadpa)) {
//ошибка
}
else {
// sadpa.data = значение регистра MIL_RT_TR_REG подадреса 7 канала 0
}
```

9.7. IOCTL_WR_BLOCK_BUF_PA

Назначение:

Запись блока данных SBLOCK_BUF_PA.buf[32] в буфер передачи номер SBLOCK_BUF_PA.buf (0, 1) подадреса SBLOCK_BUF_PA.paddr канала SBLOCK_BUF_PA.channel

Входные параметры:

Param - переменная типа SBLOCK_BUF_PA.

Пример вызова:

```
#include “ioctl_man.h”
SBLOCK_BUF_PA sbbpa;
sbbpa.num_buf = 0;
sbbpa.paddr = 3;
sbbpa.buf[0] = 0x5555;
sbbpa.buf[31] = 0xaaaa;
sbbpa.channel = 0;
if (ioctl(fd,  IOCTL_WR_BLOCK_BUF_PA, &sbbpa)) {
//ошибка
}
else {
// Буфер sbbpa.buf[32] записан в буфер 0 подадреса передачи 3 канала 0
}
```

9.8. IOCTL_RD_BLOCK_BUF_PA

Назначение:

Чтение блока данных из буфера передачи номер SBLOCK_BUF_PA.buf (0, 1) подадреса SBLOCK_BUF_PA.paddr канала SBLOCK_BUF_PA.channel и копирование его в SBLOCK_BUF_PA.buf[32]

Входные параметры:

Param - переменная типа SBLOCK_BUF_PA.

Пример вызова:

```
#include “ioctl_man.h”
SBLOCK_BUF_PA sbbpa;
sbbpa.num_buf = 1;
sbbpa.paddr = 8;
sbbpa.channel = 1;
if (ioctl(fd,  IOCTL_RD_BLOCK_BUF_PA, &sbbpa)) {
//ошибка
}
else {
// sbbpa.buf[32] = содержимому буфера 1 подадреса передачи 8 канала 1
}
```

10. Команды работы с прерываниями.

Механизм прерываний основан на использовании сигналов. На каждое прерывание можно назначить индивидуальные сигнал и процесс. Примеры использования приложены к драйверу.

Возможные прерывания:
SINTHIOSA_INTR_HDAT — заполнение буфера DMA на 1/8
SINTHIOSA_INTR_QDAT — заполнение буфера DMA на 1/2
SINTHIOSA_INTR_FLASH — прерывание контроллера FLASH
SINTHIOSA_INTR_MC — прерывание при получении Команды Управления
SINTHIOSA_INTR_ERR — прерывание по ошибке
SINTHIOSA_INTR_SADDR — прерывание подадреса

Команды:

10.1. IOCTL_DISABLE_INT_ALLCH_HDAT

Назначение:
Отключить посылку сигнала по прерыванию HDAT по всем каналам платы.

Входные параметры:

Param – не используется.

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_DISABLE_INT_ALLCH_HDAT, 0)) {
//ошибка
}
else {
//
}
```

10.2. IOCTL_DISABLE_INT_CH_HDAT

Назначение:
Отключить посылку сигнала по прерыванию HDAT по указанному каналу платы.

Входные параметры:

Param – номер канала.

Пример вызова:

```
#include “ioctl_man.h”
unsigned long channel = 0; // номер канала
if (ioctl(fd,  IOCTL_DISABLE_INT_CH_HDAT, &channel)) {
//ошибка
}
else {
//
}
```

10.3. IOCTL_DISABLE_INT_ALLCH_QDAT

Назначение:

Отключить посылку сигнала по прерыванию QDAT по всем каналам платы.

Входные параметры:

Param – не используется.

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_DISABLE_INT_ALLCH_QDAT, 0)) {
//ошибка
}
else {
//
}
```

10.4. IOCTL_DISABLE_INT_CH_QDAT

Назначение:

Отключить посылку сигнала по прерыванию QDAT по указанному каналу платы.

Входные параметры:

param – указатель на переменную channel типа unsigned long

Пример вызова:

```
#include “ioctl_man.h”
unsigned long channel = 0; // номер канала
if (ioctl(fd,  IOCTL_DISABLE_INT_CH_QDAT, &channel)) {
//ошибка
}
else {
//
}
```

10.5. IOCTL_DISABLE_INT_ALLCH_FLASH

Назначение:

Отключить посылку сигнала по прерыванию FLASH по всем каналам платы.

Входные параметры:

Param – не используется.

Пример вызова:

```
#include “ioctl_man.h”
if (ioctl(fd,  IOCTL_DISABLE_INT_ALLCH_FLASH, 0)) {
//ошибка
}
else {
//
}
```

10.6. IOCTL_DISABLE_INT_CH_FLASH

Назначение:

Отключить посылку сигнала по прерыванию FLASH по указанному каналу платы.

Входные параметры:

Param – указатель на переменную channel типа unsigned long

Пример вызова:

```
#include “ioctl_man.h”
unsigned long channel = 0; // номер канала
if (ioctl(fd,  IOCTL_DISABLE_INT_CH_FLASH, &channel)) {
//ошибка
}
else {
//
}
```

10.7. IOCTL_DISABLE_INT_CH_MC

Назначение:

Отключить посылку сигнала по прерыванию MC по указанному каналу платы.

Входные параметры:

param – указатель на переменную channel типа unsigned long

Пример вызова:

```
#include “ioctl_man.h”
unsigned long channel = 0; // номер канала
if (ioctl(fd,  IOCTL_DISABLE_INT_CH_MC, &channel)) {
//ошибка
}
else {
//
}
```

10.8. IOCTL_DISABLE_INT_CH_ERR

Назначение:

Отключить посылку сигнала по прерыванию ERR по указанному каналу платы.

Входные параметры:

param – указатель на переменную channel типа unsigned long

Пример вызова:

```
#include “ioctl_man.h”
unsigned long channel = 0; // номер канала
if (ioctl(fd,  IOCTL_DISABLE_INT_CH_ERR, &channel)) {
//ошибка
}
else {
//
}
```

10.9. IOCTL_ENABLE_INT_CH_IO_SA_PID_SIG

Назначение:

Включение прерываний и назначение сигналов.

Входные параметры:

param – указатель на структуру типа SINTHIOSA

Пример вызова:

```
#include “ioctl_man.h”
SINTHIOSA sin;
sin.channel = 0; // номер канала
sin.intr = SINTHIOSA_INTR_HDAT; // прерывание, по которому будет отправлен сигнал
sin.enable_tr = 0x00000001; // 0 – запретить прерывание при передаче по подаресу; 1 - разрешить (разрешено по 1, запрещено по остальным)
sin.disable_rcv = 0x00000003; //  0 – запретить прерывание при приеме по подаресу; 1 – разрешить (разрешено по 1 и 2, запрещено по остальным)
sin.proc_pid = pid; // pid процесса, на который Вы хотите получить данный сигнал
sin.sig_no = signal; // сигнал, кторый Вы хотите получить
if (ioctl(fd,  IOCTL_ENABLE_INT_CH_IO_SA_PID_SIG, &sin)) {
//ошибка
}
else {
//
}
```

11. Обновление руководства.

Версия документа	Дата	Изменение
1.0	23.11.2012	Документ создан.
1.1	24.02.2015	Изменения в оформлении документа Добавлен раздел “Список доступных команд по версиям драйверов”