Руководство (v1.4)

По работе с драйвером модуля «mPCIe – MIL1553UD»

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

Для драйверов версии 1.2 и ниже

ОС WINDOWS

28.11.2016

ООО «Новомар» 2016 г.

Оглавление

1. Введение 3
2. Установка драйвера 4
2.1 Расшифровка названия драйвера. 5
3. Список доступных IOCTL вызовов по версиям драйверов 6
4. IOCTL вызовы. 7
4.1 IOCTL_MIL1553DEV_CLEAR_DMA 8
4.2 IOCTL_MIL1553DEV_DISABLE_INTERRUPT 9
4.3 IOCTL_MIL1553DEV_DISABLE_SUBADDR 10
4.4 IOCTL_MIL1553DEV_DMA_COUNT 11
4.5 IOCTL_MIL1553DEV_ENABLE_SUBADDR 12
4.6 IOCTL_MIL1553DEV_GET_DEVICE_VER 13
4.7 IOCTL_MIL1553DEV_GET_INT_FLAGS 14
4.8 IOCTL_MIL1553DEV_GET_NBLOCK_RAW_DMA 15
4.9 IOCTL_MIL1553DEV_GET_RESOURCE_INFO 16
4.10 IOCTL_MIL1553DEV_READ_BAR 17
4.11 IOCTL_MIL1553DEV_READ_DMA_DATA_BLOCKS 18
4.12 IOCTL_MIL1553DEV_READ_DMA_DATABUF 19
4.13 IOCTL_MIL1553DEV_READ_DMA_STATUS 20
4.14 IOCTL_MIL1553DEV_READ_REG 21
4.15 IOCTL_MIL1553DEV_READ_SUBADDR_BUF 22
4.16 IOCTL_MIL1553UDx_REGISTER_INTERRUPT 23
4.17 IOCTL_MIL1553DEV_WRITE_BAR 25
4.18 IOCTL_MIL1553DEV_WRITE_REG 26
4.19 IOCTL_MIL1553DEV_WRITE_SUBADDR_BUF 27
5. Обновление драйвера 28
6. Обновление руководства 28

1. Введение

Основой драйвера для модуля **«mPCIe–1553UD»**, является файл Mil1553UD_YY_VerX.sys, где YY является разрядностью ОС (x86 или x64), для которой разработан драйвер, а X версия драйвера.

Драйвер разрабатывался и тестировался для операционных систем Microsoft Windows XP 32 bit edition, Microsoft Windows 7 32 bit edition и Microsoft Windows 7 64 bit edition.

При установке драйвера на ОС Windows XP, система выдаст предупреждение о том, что данное ПО не прошло сертификацию для Windows XP. Это связано с тем, что компания Microsoft более не поддерживает Windows XP. Нажмите кнопку «Продолжить». Драйвер установится и начнёт свою корректную работу.

Драйвер поддерживает как работу одного, так и одновременную работу нескольких устройств и присваивает им уникальные символьные имена **Mil1553UDx**, где X – индекс устройства, начиная с 0. Т.е. при одновременной работе двух и более устройств, одно будет иметь имя Mil1553UD0, второе Mil1553UD1 и т.д.

2. Установка драйвера

Установка драйвера производится стандартными средствами установки оборудования ОС Windows.

Для установки драйвера следует открыть «Диспетчер устройств», выбрать устройство с идентификатором **PCI\VEN_A203&DEV_9470&REV_13** и нажать установить драйвер. Идентификатор можно просмотреть в свойствах устройства, во вкладке «Сведения», выбрав пункт «ИД оборудования».

Далее следует выбрать кнопку «Выполнить поиск драйверов на этом компьютере», указать путь к директории драйвера и нажать «Далее».

Если система отобразит ошибку, что не удалось найти драйвер для этого устройства, значит устройство выбрано неверно. Проверьте идентификатор устройства.

Если система отобразит ошибку о том, что устройство не может начать работу (код 10), перезагрузите компьютер.

Если система отобразит ошибку о том, что не удалось проверить цифровую подпись драйвера (код 52), проверьте наличие обновления ОС KB3033929. В Windows 7 наличие обновления можно проверить по следующему пути: «Пуск → Панель управления → Система и безопасность → Просмотр установленных обновлений → Поиск: установленные обновления».

Если система отобразит сообщение, что драйвер установлен, то можно приступать к работе с устройством.

Модуль теперь можно найти в «Диспетчере устройств» в ветке «Multifunction Adapters» под именем «Novomar® MIL1553UD Controller».

2.1 Расшифровка названия драйвера.

MIL1553UD_x86_Ver1_0.sys
1 2 3 4

1	***MIL1553UD***	Название поддерживаемого модуля
2	***X86***	Разрядность ОС, для которой предназначен драйвер
2	***Ver1_0***	Версия драйвера
3	***sys***	Расширение файла

3. Список доступных IOCTL вызовов по версиям драйверов

Название вызова	Краткое описание
Список функций, добавленных **в драйвере версии 1.2**	
IOCTL_MIL1553DEV_DISABLE_INTERRUPT	Запрет прерываний
IOCTL_MIL1553Udx_REGISTER_INTERRUPT	Разрешение и ожидание прерывания
Список вызовов, доступных **в драйвере версии 1.0**	
IOCTL_MIL1553DEV_CLEAR_DMA	Очистка буфера DMA
IOCTL_MIL1553DEV_DISABLE_SUBADDR	Запрет подадреса
IOCTL_MIL1553DEV_DMA_COUNT	Чтение программного счётчика записанных блоков данных в DMA.
IOCTL_MIL1553DEV_ENABLE_SUBADDR	Разрешение подадреса
IOCTL_MIL1553DEV_GET_DEVICE_VER	Чтение номера ревизии устройства.
IOCTL_MIL1553DEV_GET_INT_FLAGS	Чтение флагов прерываний.
IOCTL_MIL1553DEV_GET_NBLOCK_RAW_DMA	Количество новых данных в буфере DMA
IOCTL_MIL1553DEV_GET_RESOURCE_INFO	Получение информации о ресурсах и состоянии.
IOCTL_MIL1553DEV_READ_BAR	Чтение данных из регистрового пространства устройства (BAR)
IOCTL_MIL1553DEV_READ_DMA_DATA_BLOCKS	Чтение данных из буфера DMA
IOCTL_MIL1553DEV_READ_DMA_DATABUF	Чтение данных из буфера DMA
IOCTL_MIL1553DEV_READ_DMA_STATUS	Чтение статуса DMA
IOCTL_MIL1553DEV_READ_REG	Чтение регистров устройства
IOCTL_MIL1553DEV_READ_SUBADDR_BUF	Чтение буфера подадреса
IOCTL_MIL1553DEV_WRITE_BAR	Запись регистров устройства
IOCTL_MIL1553DEV_WRITE_REG	Запись регистров устройства
IOCTL_MIL1553DEV_WRITE_SUBADDR_BUF	Запись буфера подадреса

4. IOCTL вызовы.

В данном описании приняты следующие соглашения и допущения:

1) по умолчанию, смещения указываются в шестнадцатеричном виде (hex);
2) по умолчанию, размеры указываются в десятичном виде;
3) по умолчанию, установленный флаг возврата говорит об удачном выполнении запроса, сброшенный же флаг возврата означает, что

- запрос был отклонен ОС;
- запрос не был выполнен драйвером из-за неправильных входных параметров.

Во всех примерах вызовов есть три неописанные переменные, это:

m_hDevice – дескриптор устройства на котором должна выполниться операция. Чтобы извлечь данные о дескрипторе устройства, используйте функцию **CreateFile**.

lpBytesReturned – указатель на переменную, которая получает размер переданных данных.

Xxx – параметр который должен быть равен либо NULL, либо ссылке на структуру OVERLAPPED**,** для того чтобы операция выполняется как перекрывающая (асинхронная) операция.

В случае удачного выполнения функция возвращает ненулевое значение. В случае ошибки возвращаемое значение равно нулю. Чтобы получить дополнительную информацию об ошибке, вызовите **GetLastError**

Вызовы сгруппированы по алфавиту.

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.

4.1 IOCTL_MIL1553DEV_CLEAR_DMA

Назначение:

По вызову данной функции программный счётчик вычитанных пакетов данных DMA уравнивается с регистром DMA_INDEX*(адрес 0x1008).

*См. Раздел 5.1.2 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Действие:

-

Примечание:

ВНИМАНИЕ!!!

ФУНКЦИЮ ИСПОЛЬЗОВАТЬ ОЧЕНЬ АККУРАТНО! ВОЗМОЖНА ПОТЕРЯ ДАННЫХ. ПЕРЕД ВЫЗОВОМ ПРОВЕРЬТЕ ВЫЧИТАНЫ ЛИ ВСЕ ДАННЫЕ.

Вход:

–

Выход:

–

Пример вызова:

```
if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_CLEAR_DMA,
  NULL, 0,
  NULL, 0,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.2 IOCTL_MIL1553DEV_DISABLE_INTERRUPT

Назначение:
Запрет прерывания m_nInt. (Функции разрешения прерывания отсутствует, т.к. прерывание разрешается в вызове IOCTL_MIL1553DEV_REGISTER_INTERRUPT).

Действие:
Функция запрещает прерывание m_nInt, записывая данные либо только в регистр INTERRUPT MASK* (адрес 1010h), либо и в регистр INTERRUPT MASK и в регистр RT_INT_REG** (адрес 2034h).
* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553UD*".
** см. Раздел 6.1.8 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
Варианты InterruptName:

- **INT_HDAT –** прерывание по заполнению ½ буфера данных DMA;
- **INT_QDAT –** прерывание по заполнению 1/16 буфера данных DMA;
- **RT_INT_MC –** прерывание по приёму КУ;
- **RT_INT_ERR** – прерывание по ошибке сообщения;
- **RT_INT_REN** – прерывание от подадреса приёма;
- **RT_INT_TEN** – прерывание от подадреса передачи;
- **INT_BC** – Прерывание режима КШ.

Вход – структура MIL1553UD_INT_NTFCN_IN:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nInt:* номер прерывания	См. п. примечание
0x04	4	*m_hEvt:*	Не используется

Выход:
-

Пример вызова:

```
HANDLE Event= CreateEvent(NULL, TRUE, FALSE, name);
MIL1553UD_INT_NTFCN_IN InputBuf;
InputBuf.m_nInt = INT_HDAT;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_DISABLE_INTERRUPT,
  &InputBuf, sizeof(InputBuf),
  NULL, 0,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.3 IOCTL_MIL1553DEV_DISABLE_SUBADDR

Назначение:

Запрет подадреса, в зависимости от выбранного режима работы.

Действие:

В зависимости от выбранного канала, а также в зависимости номера подадреса функция устанавливает в единицу 31 бит регистра RT_RCV_REG$n^{(*)}$** и сбрасывает в ноль 30 бит регистра RT_TR_REG$n^{(*)}$***.

*n – номер запрашиваемого подадреса

** см. Раздел 6.1.12 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

*** см. Раздел 6.1.13 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:

Если вызов заканчивается неудачно с ошибкой STATUS_INVALID_PARAMETER, значит значение парамера m_nNum выходит за границы диапазона 1-30.

Вход – структура MIL1553_SUBADDR:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nRT:* приём или передача	Не используется
0x04	4	*m_nNum:* номер подадреса	1-30

Выход:

-

Пример вызова:

```
MIL1553_SUBADDR InputBuf;
InputBuf.m_nNum = 1;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_DISABLE_SUBADDR,
  &InputBuf, sizeof(InputBuf),
  NULL, 0,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.4 IOCTL_MIL1553DEV_DMA_COUNT

Назначение:
Чтение программного счётчика записанных блоков данных в DMA.

Действие:
Функция забирает значение счётчика из внутренних переменных драйвера.

Примечание:
-

Вход:
–

Выход – MIL1553_DMA_COUNT_OUT:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nData:* Значение счётчика	В байтах

Пример вызова:

```
MIL1553_DMA_COUNT_OUT OutputBuf;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_DMA_COUNT,
  NULL, NULL,
  &OutputBuf, sizeof(OutputBuf),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()

nCount = OutputBuf.m_nCount;
```

4.5 IOCTL_MIL1553DEV_ENABLE_SUBADDR

Назначение:

Разрешение подадреса, в зависимости от выбранного режима работы.

Действие:

Если выбран режим передачи, то функция устанавливает в единицу 31 бит регистра RT_RCV_REG$n^{(*)}$** и устанавливает в единицу 30 бит регистра RT_TR_REG$n^{(*)}$***.

Если выбран режим приёма, то функция сбрасывает в ноль 31 бит регистра RT_RCV_REG$n^{(*)}$** и сбрасывает в ноль 30 бит регистра RT_TR_REG$n^{(*)}$***.

*n – номер запрашиваемого подадреса

** см. Раздел 6.1.12 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

*** см. Раздел 6.1.13 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:

Если вызов заканчивается неудачно с ошибкой STATUS_INVALID_PARAMETER, значит значение парамера m_nNum выходит за границы диапазона 1-30.

Вход – структура MIL1553Udx_SUBADDR:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nRT*: выбор режима работы. Если 1, то подадрес разрешается на приём, если 0, то подадрес разрешается на передачу.	
0x04	4	*m_nNum:* номер подадреса	1-30

Выход:

-

Пример вызова:

```
MIL1553_SUBADDR InputBuf;
  InputBuf.m_nRT = 1;
  InputBuf.m_nNum = 1;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_ENABLE_SUBADDR,
  &InputBuf, sizeof(InputBuf),
  NULL, NULL,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.6 IOCTL_MIL1553DEV_GET_DEVICE_VER

Назначение:
Чтение номера ревизии устройства.

Действие:
Функция читает значение ревизии устройства из конфигурационного адресного пространства PCI.

Примечание:
-

Вход:
–

Выход:

Смещение	Размер	Назначение	Примечание
0x00	4	Номер ревизии устройства	

Пример вызова:

```
UINT32 Output;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_GET_DEVICE_VER,
  NULL,NULL,
  &Output, sizeof(Output),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()

Revision = Output;
```

4.7 IOCTL_MIL1553DEV_GET_INT_FLAGS

Назначение:
Чтение флагов прерываний.

Действие:
В переменную m_nIntValue записывается значение регистра INTERRUPT* (адрес 100Ch). После этого значение регистра сбрасываются.

В переменную m_nRTValue читается значение 20-23 битов регистра HW_STAT_REG2**. Флаги переписываются последующими данными.

* см. Раздел 5.1.3 документа "*Руководство по программированию модуля "mPCIe-1553UD*".
** см. Раздел 6.1.10 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
-

Вход:
–

Выход – структура MIL1553_INT_FLAGS_OUT:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nIntValue*: Текущее значение регистра прерывний INTERRUPT	
0x04	4	*m_nRTValue*: Текущее значение регистра прерывний HW_STAT_REG2	

Пример вызова:

```
MIL1553_GET_INT_FLAGS_OUT Output;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_GET_INT_FLAGS,
  NULL,NULL,
  &Output, sizeof(Output),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()

nIntValue = Output.m_nIntValue;
nRTValue = Output.m_nRTValue;
```

4.8 IOCTL_MIL1553DEV_GET_NBLOCK_RAW_DMA

Назначение:
Чтение количества новых блоков данных, накопленных в буфере DMA (блок – 128 байт).

Действие:
Функция вычитает из значения регистра DMA_INDEX*(адреса 0x1008) значение программного счётчика вычитанных пакетов. Полученное количество блоков возвращается в переменной m-nBlock.
*См. Раздел 5.1.2 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
-

Вход:
-

Выход – структура MIL1553_GET_NBLOCK_DMA_OUT:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nBlock:* Кол-во новых блоков данных	

Пример вызова:

```
MIL1553_GET_NBLOCK_DMA_OUT Output;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553_GET_NBLOCK_RAW_DMA,
  &Input,sizeof(Input),
  &Output, sizeof(Output),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()

nBlock = Output.m_nBlock;
```

4.9 IOCTL_MIL1553DEV_GET_RESOURCE_INFO

Назначение:
Получение информации о ресурсах и состоянии.

Действие:
Функция читает значение ревизии устройства из конфигурационного адресного пространства PCI.

Примечание:
-

Вход:
–

Выход – структура MIL1553_GET_RESOURCE_INFO_OUT:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nBus* : Номер шины	
0x04	4	*m_nSlot* : Номер слота	
0x08	4	*m_nFunction* : Номер функции	
0x0C	4	*m_nDataBufPhysAddr* : Физический адрес буфера DMA	
0x10	4	*m_nDataBufSize* : Размер буфера DMA	В байтах.
0x14	4	*m_BarPhysAddr*: Физический адрес пространства регистра BAR	
0x18	128	*m_szCompileDate*: Текстовая строка, заканчивающаяся нулем, с информацией о времени и дате компиляции драйвера	

Пример вызова:

```
MIL1553_GET_RESOURCE_INFO_OUT Output;
if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_GET_DEVICE_VER,
  NULL,NULL,
  &Output, sizeof(Output),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()

pBus = Output.m_nBus;
pSlot = Output.m_nSlot;
nFunction = Output.m_nFunction;
pBufAddr = Output.m_nDataBufPhysAddr;
pBufSize = Output.m_nDataBufSize;
pBarAddr = Output.m_BarPhysAddr;
pStr = (char*)Output.m_szCompileDate;
```

4.10 IOCTL_MIL1553DEV_READ_BAR

Назначение:
Чтение данных из регистрового пространства устройства (BAR).

Действие:
Функция читает необходимое количество данных из желаемого адреса регистрового пространства устройства (BAR) в переменную pData.

Примечание:
Рекомендуется использовать если данные больше 32 бит. Если меньше или равны 32 битам, то рекомендуется использовать вызов IOCTL_MIL1553DEV_READ_REG.

Вход – структура MIL1553_READ_BAR_IN:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nOffset*: Смещение блока данных на чтение	
0x04	4	*m_nSize*: Размер блока данных	

Выход:

Смещение	Размер	Назначение	Примечание
0x00	*m_nSize*	*m_nOffset*: Считанный блок данных	

Пример вызова:

```
MIL1553_READ_BAR_IN Input;
Input.m_nOffset = nAddr;
Input.m_nSize = nSize;
Input.m_nBar = nBar;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_READ_BAR,
  &Input, sizeof(Input),
  &Data, sizeof(Data),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.11 IOCTL_MIL1553DEV_READ_DMA_DATA_BLOCKS

Назначение:

Чтение данных из буфера DMA. Данная функция читает любое требуемое значение новых блоков данных DMA.

Действие:

По номеру запрашиваемого канала функция читает данные из буфера DMA. Новые данные кладутся в переменную pData до тех пор, пока либо не наберётся запрашиваемое количество блоков, либо блоки с новыми данными не закончатся. Прочитанных блоков данных может быть меньше, чем запрашиваемых!

Примечание:

-

Вход – структура MIL1553_READ_DMA_DATA_BLOCKS_IN:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nBlocks*: Запрашиваемое кол-во блоков данных для чтения	

Выход:

Смещение	Размер	Назначение	Примечание
0x01	*m_nBlocks*128*	Считанные блоки данных	

Пример вызова:

```
MIL1553_READ_DMA_DATA_BLOCKS_IN InputBuf;
//Размер запрашиваемых данных
UINT32 nOutputSize = 2*128;
InputBuf.m_nBlocks = 2;
InputBuf.m_nCh = 1;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_READ_DMA_DATA_BLOCKS,
  &InputBuf, sizeof(InputBuf),
  &pData, nOutputSize,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
//размер прочитанных данных
Blocks = lpBytesReturned/128;
```

4.12 IOCTL_MIL1553DEV_READ_DMA_DATABUF

Назначение:

Чтение данных из буфера DMA. Данная функция вычитывает только один новый блок данных.

Действие:

Данная функция читает данные из буфера DMA только по одному блоку (блок – 128 байт). Для того, чтобы проверить есть ли готовые данные надо вызвать функцию IOCTL_MIL1553DEV_READ_DMA_STATUS.

. Если возвращаемое ей значение равно ‘1’, значит данные есть и их следует вычитать вызвав данную функцию, и снова проверить статус готовности данных DMA. И так пока статус не станет равен нулю.

Примечание:

-

Вход:

–

Выход:

Смещение	Размер	Назначение	Примечание
0x00	128	Считанный блок данных	В байтах

Пример вызова:

```
if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_READ_DMA_DATABUF,
  NULL, NULL,
  &OutputBuf, 128,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.13 IOCTL_MIL1553DEV_READ_DMA_STATUS

Назначение:
Чтение статуса DMA.

Действие:
Функция сравнивает программный счётчик вычитанных пакетов данных из DMA со значением регистра DMA_INDEX*(адрес 0x1008), если значения не равны, то новые данные появились, их следует вычитать и в OutputBuf вернётся 1. Если равны, то новых данных нет и в OutputBuf вернётся 0.
*См. Раздел 5.1.2 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
-

Вход:
–

Выход – MIL1553_READ_DMA_STATUS:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nStatus*: Статус новых данных	

Пример вызова:

```
MIL1553_READ_DMA_STATUS OutputBuf;
if(::DeviceIoControl(
  m_hDevice,
IOCTL_MIL1553_READ_DMA_STATUS,
  NULL, NULL,
  &OutputBuf, sizeof(OutputBuf),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.14 IOCTL_MIL1553DEV_READ_REG

Назначение:
Чтение данных из регистрового пространства устройства (BAR).

Действие:
Функция читает данные из желаемого адреса регистрового пространства устройства (BAR) в переменную pData.

Примечание:
Рекомендуется использовать если данные меньше, либо равны 32 битам. Если больше 32 бит, то рекомендуется использовать функцию IOCTL_MIL1553DEV_READ_BAR.

Вход – структура MIL1553_READ_REG_IN:

Смещение	Размер	Назначение	Примечание
0x00	*4*	*m_nAddress*: Адрес регистра	

Выход:

Смещение	Размер	Назначение	Примечание
0x00	4	Считанный блок данных	

Пример вызова:

```
MIL1553_READ_REG_IN InputBuf;
  InputBuf.m_nAdress = nAddr;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_READ_REG,
  &InputBuf, sizeof(InputBuf),
  pData, sizeof(pData),
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.15 IOCTL_MIL1553DEV_READ_SUBADDR_BUF

Назначение:
Чтение данных из буфера подадреса.

Действие:
В зависимости от выбранного номера буфера и номера подадреса функция вычитывает блок данных запрашиваемого подадреса из буфера RT_DATA_BUF0$n^{(*)}$** или RT_DATA_BUF1$n^{(*)}$***.
*n – номер запрашиваемого подадреса
** см. Раздел 6.1.14 документа "*Руководство по программированию модуля "mPCIe-1553UD*".
*** см. Раздел 6.1.15 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
-

Вход – структура MIL1553_RD_SUBADDR_IN:

Смещение	Размер	Назначение	Примечание
0x00	*4*	*m_nNum:* Номер подадреса	1-30
0x04	*4*	*m_nBufNum:* Выбор буфера. Если 0, то RT_DATA_BUF0n , если 1, то RT_DATA_BUF1n	

Выход:

Смещение	Размер	Назначение	Примечание
0x00	64	Считанный блок данных	

Пример вызова:

```
MIL1553_RD_SUBADDR_IN InputBuf;

InputBuf.m_nBufNum = 0;
InputBuf.m_nNum = 1;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_READ_SUBADDR_BUF,
  &InputBuf, sizeof(InputBuf),
  pData, nSize,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError();
```

4.16 IOCTL_MIL1553DEV_REGISTER_INTERRUPT

Назначение:
Разрешение и ожидание прерывания m_nInt.

Действие:
Функция передаёт драйверу событие (event) m_hEvt и информацию об ожидаемом драйвере. Драйвер разрешает прерывание m_nInt и ожидает его.
Для разрешения прерывания функция записывает данные либо только в регистр INTERRUPT MASK* (адрес 1010h), либо и в регистр INTERRUPT MASK и в регистр RT_INT_REG** (адрес 2034h).
Как только ожидаемое прерывание обрабатывается драйвером, hEvent переводится в сигнальное состояние.
* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553UD*".
** см. Раздел 6.1.8 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
Варианты m_nInt:

- **INT_HDAT –** прерывание по заполнению ½ буфера данных;
- **INT_QDAT –** прерывание по заполнению 1/16 буфера данных;
- **RT_INT_MC –** прерывание по приёму КУ;
- **RT_INT_ERR** – прерывание по ошибке сообщения;
- **RT_INT_REN** – прерывание от подадреса приёма (Прерывание не будет срабатывать, если подадрес не разрешён на приём данных. Для разрешения воспользуйтесь вызовом IOCTL_MIL1553DEV_ENABLE_SUBADDR);
- **RT_INT_TEN** – прерывание от подадреса передачи (Прерывание не будет срабатывать, если подадрес не разрешён на передачу данных. Для разрешения воспользуйтесь вызовом IOCTL_MIL1553DEV_ENABLE_SUBADDR);
- **INT_BC** – Прерывание режима КШ.

Вход – структура MIL1553UD_INT_NTFCN_IN:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nInt:* номер прерывания	См. п. примечание
0x04	4	*m_hEvt:* Event для ожидания прерывания	

Выход:
–

Пример вызова на следующей странице:

Пример вызова:

```
MIL1553UD_INT_NTFCN_IN InputBuf;
InputBuf.m_nInt = InterruptName;
InputBuf.m_hEvt = hEvent;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_REGISTER_INTERRUPT,
  &InputBuf, sizeof(InputBuf),
  NULL, NULL,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.17 IOCTL_MIL1553DEV_WRITE_BAR

Назначение:
Запись данных в регистровое пространство устройства (BAR).

Действие:
Функция записывает необходимое количество данных по желаемому адресу в регистровое пространство устройства (BAR).

Примечание:
Рекомендуется использовать если данные больше 32 бит. Если меньше или равны 32 битам, то рекомендуется использовать функцию IOCTL_MIL1553DEV_WRITE_REG.

Вход – структура MIL1553_WRITE_BAR_IN:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nOffset*: Смещение блока данных на запись	В байтах
0x04	4	*m_nSize*: Размер блока данных	В байтах
0x08	*m_nSize*	Блок данных на запись	

Выход:
–

Пример вызова:

```
MIL1553_WRITE_BAR_IN nInputSize;
pInput->m_nOffset = nAddr;
pInput->m_nSize = nSize;
pInput->m_nBar = nBar;
pInput->m_Data = Data;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_WRITE_BAR
  &InputBuf, sizeof(InputBuf),
  NULL, NULL,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.18 IOCTL_MIL1553DEV_WRITE_REG

Назначение:
Запись данных в регистровое пространство устройства (BAR).

Действие:
Функция записывает данные по желаемому адресу в регистровое пространство устройства (BAR).

Примечание:
Рекомендуется использовать если данные меньше, либо равны 32 битам. Если больше 32 бит, то рекомендуется использовать функцию IOCTL_MIL1553DEV_WRITE_BAR.

Вход – структура MIL1553_READ_BAR_IN:

Смещение	Размер	Назначение	Примечание
0x00	4	*m_nAddress:* Адрес регистра	
0x04	4	*m_nData:* Записываемый блок данных	В байтах

Выход:
–

Пример вызова:

```
MIL1553_WRITE_REG_IN InputBuf;

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553DEV_WRITE_REG,
  &InputBuf, sizeof(InputBuf),
  NULL, NULL,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

4.19 IOCTL_MIL1553DEV_WRITE_SUBADDR_BUF

Назначение:
Запись данных в буфер подадреса.

Действие:
В зависимости от выбранного номера буфера и номера подадреса функция записывает в буфер RT_DATA_BUF0$n^{(*)}$** или RT_DATA_BUF1$n^{(*)}$*** данные из переменной nData.
*n – номер запрашиваемого подадреса
** см. Раздел 6.1.14 документа "*Руководство по программированию модуля "mPCIe-1553UD*".
*** см. Раздел 6.1.15 документа "*Руководство по программированию модуля "mPCIe-1553UD*".

Примечание:
-

Вход – структура MIL1553_WR_SUBADDR_IN:

Смещение	Размер	Назначение	Примечание
0x00	*4*	*m_nNum:* Номер подадреса	1-30
0x04	*4*	*m_nBufNum:* Выбор буфера. Если 0, то RT_DATA_BUF0n , если 1, то RT_DATA_BUF1n	
0x08	*64*	*nData*: Блок данных на запись	

Выход:
-

Пример вызова:

```
MIL1553_WR_SUBADDR_IN InputBuf;
pInput.m_nBufNum = 0;
pInput.m_nNum = 1;
memcpy(&pInput.m_nData, pData, 0x40);

if(::DeviceIoControl(
  m_hDevice,
  IOCTL_MIL1553Udx_WRITE_SUBADDR_BUF,
  &InputBuf, sizeof(InputBuf),
  NULL, NULL,
  &lpBytesReturned,
  xxx))
else
  dw = GetLastError()
```

5. Обновление драйвера

Версия драйвера	Дата	Изменение
1.0	22.07.2013	- Драйвер создан
1.1	12.05.2015	- Улучшена работа вызовов: IOCTL_MIL1553DEV_ENABLE_SUBADDR IOCTL_MIL1553DEV_DISABLE_SUBADDR IOCTL_MIL1553DEV_READ_SUBADDR_BUF IOCTL_MIL1553DEV_WRITE_SUBADDR_BUF - Добавлены функции: IOCTL_MIL1553DEV_REGISTER_INTERRUPT IOCTL_MIL1553DEV_DISABLE_INTERRUPT
1.2	28.11.2016	- Обновлена цифровая подпись

6. Обновление руководства

Версия документа	Дата	Изменение
1.0	22.07.2013	Документ создан
1.1	23.12.2014	Обновлены разделы: 1. Введение 2. Установка драйвера 2.1 Расшифровка названия драйвера
1.2	24.02.2015	Добавлен раздел “Список доступных IOCTL вызовов по версиям драйверов” Обновлён титульный лист.
1.3	12.05.2015	- Исправлено описание вызова IOCTL_MIL1553DEV_GET_INT_FLAGS - Добавлены описания вызовов IOCTL_MIL1553DEV_WRITE_SUBADDR_BUF IOCTL_MIL1553DEV_READ_SUBADDR_BUF IOCTL_MIL1553DEV_ENABLE_SUBADDR IOCTL_MIL1553DEV_DISABLE_SUBADDR IOCTL_MIL1553DEV_READ_DMA_DATA_BLOCKS IOCTL_MIL1553DEV_GET_NBLOCK_RAW_DMA
1.4	28.11.2016	- Отредактирован пункт Установка драйвера