Руководство (v3.6)

По работе со статической библиотекой модуля «mPCIe–1553RT2»

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

Для библиотек версии 1.5 и ниже

ОС WINDOWS

28.11.2016

ООО «Новомар» 2016 г.

Оглавление

1. Подключение библиотеки к разрабатываемому проекту.

Для того, чтобы получить доступ к функциям драйвера, в разрабатываемый проект следует включить статическую библиотеку.

Библиотеку можно добавлять в проекты и среды разработок, такие как: Visual Studio(2005, 2008), Qt, C++ Builder 6, Borland C++ 5 и т.д.

1.1. Пример подключения библиотеки к среде разработки ПО «Microsoft Visual Studio»

К данной среде следует подключать библиотеку **MIL1553RT2_VerXXY.lib,** где XX – версия библиотеки, а Y – D – отладочная версия, R- релизная. Данные файлы находятся в архиве в папке «visual_studio».

Библиотека разрабатывалась и тестировалась только в среде «Microsoft Visual Studio **2008**».

Для подключения в настройках Microsoft Visual Studio выберите путь:

Project properties -> Configuration properties -> Linker -> Input -> Additional Dependecies

Укажите имя библиотеки.

Также следует указать компилятору путь к заголовочным файлам библиотеки, для этого выберите путь:

Project properties -> Configuration properties -> C/C++ -> Additional Include Directories

Укажите путь к папке с заголовочными файлами библиотеки.

По пути Project properties -> Configuration properties -> Linker -> Additional Library Directories следует указать путь к основному файлу библиотеки «**MIL1553RT2_VerXXY.lib**».

Пропишите в вашем проекте

```
#include «windows.h»
#include «MIL1553DeviceEx.h»
```

Далее выберите путь:

Project properties -> Configuration properties ->General.

Если вы подключаете библиотеку к разрабатываемому *.exe файлу то вам следует в графе «Use of MFC» выбрать «Use MFC in a static Library».

Если к *.dll, то «Use MFC in a Shared DLL».

Теперь вы можете вызывать функции библиотеки.

1.2. Пример подключения библиотеки к среде разработки ПО «Qt»

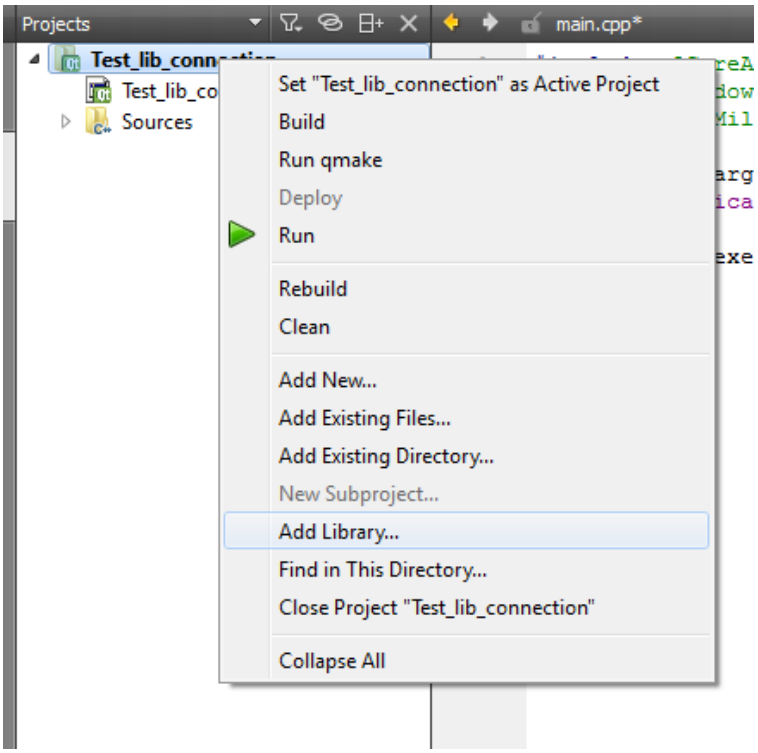
К данной среде следует подключать библиотеку «**libLibMIL1553RT2_verXX.a**», где XX – версия библиотеки, а суффикс d означает отладочную версию библиотеки.

Данные файлы находятся в архиве в папке «qt».

Библиотека разрабатывалась и тестировалась в среде Qt с компилятором **MinGW**.

Для подключения библиотеки к среде разработки Qt выполните следующие действия:

Выберите пункт «Add library» в заголовке дерева проекта.



В открывшемся окне выберите пункт «External library»;

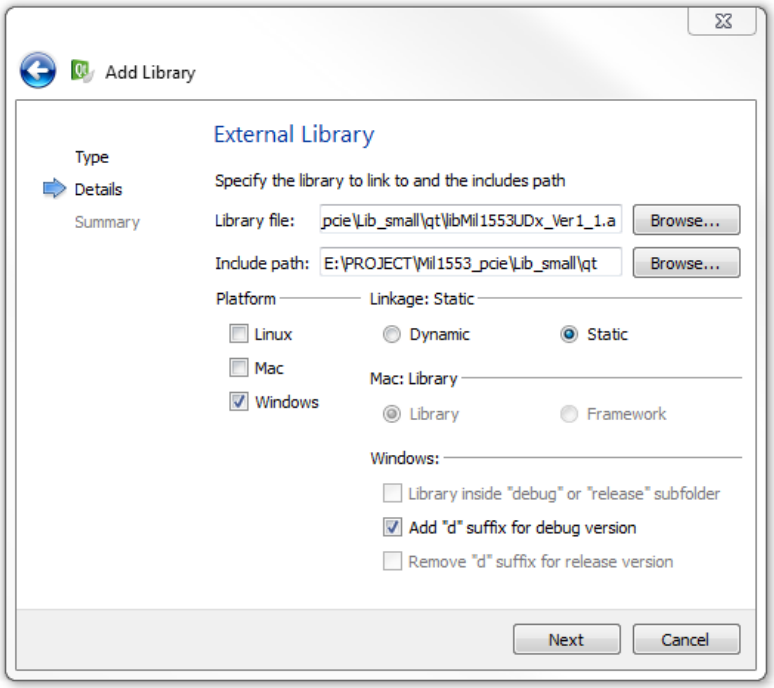
Далее в поле «Library file» выберите библиотеку без суффикса d в конце названия (релизная версия, d – отладочная);

Уберите галочки с пунктов Linux и Mac;

Выберите пункт Linkage: Static;

Установить галочку на пункте «Add «d» suffix for debug version»

Пример:



Нажмите кнопку Next и на следующем шаге кнопку Finish.
При корректном добавлении библиотеки в pro файле разрабатываемого проекта должны появиться строки следующего содержания:

```
win32:CONFIG(release, debug|release): LIBS += -L$$PWD/../ -lMil1553UDx_Ver1_1
else:win32:CONFIG(debug, debug|release): LIBS += -L$$PWD/../ -lMil1553UDx_Ver1_1d

INCLUDEPATH += $$PWD/../
DEPENDPATH += $$PWD/../

win32-g++:CONFIG(release, debug|release): PRE_TARGETDEPS += $$PWD/../libMil1553UDx_Ver1_1.a
else:win32-g++:CONFIG(debug, debug|release): PRE_TARGETDEPS += $$PWD/../libMil1553UDx_Ver1_1d.a
else:win32:!win32-g++:CONFIG(release, debug|release): PRE_TARGETDEPS += $$PWD/../Mil1553UDx_Ver1_1.lib
else:win32:!win32-g++:CONFIG(debug, debug|release): PRE_TARGETDEPS += $$PWD/../Mil1553UDx_Ver1_1d.lib
```

В вашем проекте необходимо прописать следующую строку:

#include “MIL1553UdxDevice.h”

Теперь вы можете вызывать функции библиотеки.

1.3. Расшифровка названия библиотеки.

MIL1553RT2_Ver1_0.lib
1 2 3

1	***MIL1553RT2***	Название поддерживаемого модуля
2	***Ver1_0***	Версия библиотеки
3	***Lib***	Расширение файла

2. Режим «CANONICAL»

В данном драйвере предусмотрен режим CANONICAL, обеспечивающий следующие два блока функциональности (запрещаемые/разрешаемые независимо).

В данных режимах недоступны некоторые функции. Для того чтобы понять доступна ли функция в данных режимах смотрите пункт “примечание” в описании функций. Если об этом ничего не сказано, то функция доступна как в данном режиме, так и вне его.

2.1. «CanonicalDMA»

Независимое чтение информации из буфера DMA по шинам (режим включается/выключается для всех шин устройства одновременно):

- чтение полученной информации с одной шины устройства не оказывает влияние на другие шины;
- при чтении с шины пользователь получает данные только этой шины.

При включении/выключении режима указатель буфера DMA устройства обнуляется.

Для включения используется функция EnableCanonicalDMA, для выключения DisableCanonicalDMA.

2.2. «CanonicalChMem»

“Простая” работа с буферами передачи (режим включается/выключается для каждой шины независимо).

В этом блоке реализована следующая функциональность:

- для всех подадресов шины 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.

Для включения используется функция EnableCanonicalChMem, для выключения функция DisableCanonicalChMem.

3. Список доступных функций по версиям библиотек

Название вызова	Краткое описание
Список вызовов, доступных **в библиотеке версии до 1.5 и ниже**	
BOOL Open	Открытие устройства
void Close	Закрытие устройства
BOOL GetDatabufSize	Размер буфера DMA
BOOL GetDatabufPhysAddr	Физический адрес буфера DMA.
BOOL GetBarPhysAddr	Физический адреса BAR.
BOOL GetPCILocation	Номера шины и слота, занимаемые устройством
BOOL GetRevision	Номер ревизии устройства
BOOL GetDriverCompileDate	Информация о компиляции драйвера
BOOL WriteBar	Запись регистров.
BOOL ReadBar	Чтение регистров.
BOOL ReadReg	Чтение регистров.
BOOL WriteReg	Запись регистров.
BOOL EnableDMA	Разрешение работы DMA
BOOL DisableDMA	Запрет работы DMA.
BOOL EnableCanonicalDMA	Включение режима “CanonicalDMA”
BOOL DisableCanonicalDMA	Выключение режима “CanonicalDMA”
UINT32 EnableCanonicalChMem	Включение режима “CanonicalChMem”
UINT32 DisableCanonicalChMem	Выключение режима “CanonicalChMem”
UINT32 EnableSubAddr	Разрешение подадреса.
UINT32 DisableSubAddr	Запрет подадреса.
UINT32 WorkEnable	Разрешение/запрет работы устройства
UINT32 SetRTAddress	Выбор адреса удалённого устройства.
UINT32 BusSelection	Включение/выключение шины устройства
UINT32 SetTimer	Включение/выключение и выбор шага таймера
BOOL GetProgCount	Значение программного счётчика DMA
BOOL GetDMAReadyStatus	Чтение статуса DMA.
BOOL ClearDMA	Уравнивание значений счётчиков DMA
BOOL GetNBlockRawDMA	Количество новых блоков данных DMA
UINT32 ReadDMADataOneBlock	Чтение данных из буфера DMA
BOOL ReadDMADataBlocks	Чтение данных из буфера DMA
UINT32 ReadDMACanonicalDataBlocks	Чтение данных из буфера DMA
UINT32 ReadRegSubAddr	Чтение данных из регистра подадреса

UINT32 WriteRegSubAddr	Запись данных в регистр подадреса.
UINT32 ReadRegSubAddrBuf	Чтение данных из буфера подадреса.
UINT32 WriteRegSubAddrBuf	Запись данных в буфер подадреса.
BOOL GetIntFlags	Чтение флагов прерываний.
BOOL ChangeINT_HDAT_Work	Разрешение/запрет прерывания INT_HDAT.
BOOL ChangeINT_QDAT_Work	Разрешение/запрет прерывания INT_QDAT.
BOOL ChangeINT_FLASH_Work	Разрешение/запрет прерывания INT_FLASH.
BOOL ChangeINT_RT0_SADDR_Work	Разрешение/запрет прерывания INT_RT0_SADDR.
BOOL ChangeINT_RT0_MC_ERR_Work	Разрешение/запрет прерывания INT_RT0_MC_ERR
BOOL ChangeINT_RT1_SADDR_Work	Разрешение/запрет прерывания INT_RT1_SADDR.
BOOL ChangeINT_RT1_MC_ERR_Work	Разрешение/запрет прерывания INT_RT1_MC_ERR.
UINT32 ChangeRT_INT_TEN_Work	Разрешение/запрет прерывания RT_INT_TEN.
UINT32 ChangeRT_INT_REN_Work	Разрешение/запрет прерывания RT_INT_REN.
UINT32 ChangeRT_INT_ERR_Work	Разрешение/запрет прерывания RT_INT_ERR.
UINT32 ChangeRT_INT_MC_Work	Разрешение/запрет прерывания RT_INT_MC.

4. Функции для работы с драйвером.

Все функции поделены на два типа BOOL и UINT32 по следующему критерию:

- если в функции несколько мест, где может произойти ошибка, то функция имеет тип UINT32. Каждой ошибке присвоен свой номер, который можно расшифровать, спосмотрев соответствующую таблицу;
- если в функции только одно место, где может произойти ошибка, то она имеет тип BOOL и в случае неудачи возвращает значение FALSE.

Обе функции в случае удачного выполнения возвращают значение 1/TRUE.

Каждая функция имеет пример вызова. Для того чтобы не перепечатывать код вручную, вы можете найти пример вызова функции по названию тестовой функции в файле *TestMil1553libRT2.cpp*

Пример:

Функция библиотеки:

4.1.3 BOOL GetDatabufSize(UINT32 *pBufSize)

Её пример вызова:

```
bool testGetDMASize(){
  CMIL1553DeviceEx obj;
  UINT32 nSize;
  obj.Open(0);
  if (obj.GetDatabufSize(&nSize))
    //В переменной nSize появилось
      значение размера буфера DMA.
  Else
    //Ошибка
  obj.Close();
  return true;
}
```

Название тестовой функции testGetDMASize(). По этому названию, код данной функции можно найти в файле *TestMil1553RT2lib.cpp.*

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.

При старте драйвера запрещена работа DMA. Для разрешения воспользуйтесь функцией EnableDMA().

4.1. Стандартные функции

4.1.1. BOOL Open(int nIndex)

Открытие устройства. С этой функции начинается работа с устройством.

Для открытия устройства в функцию следует передать его индекс. Если устройство в системе одно, то индекс равен ‘0’.

Если нужно работать с несколькими устройствами одновременно, то следует объявить несколько экземпляров класса:

Пример:

```
CMIL1553DeviceEx m_Device0, m_Device1;
```

Функцию Open следует вызывать с разными индексами (0, 1, 2, 3), от разных экземпляров:

```
m_Device0.Open(0); m_Device1.Open(1);
```

4.1.2. void Close()

Закрытие устройства. Для закрытия устройства функцию Close следует вызывать от тогоже экземпляра класса что и функцию Open.

Пример:

```
CMIL1553DeviceEx m_Device0;
m_Device0.Open(0);
m_Device0.Close();
```

4.1.3. BOOL GetDatabufSize(UINT32 *pBufSize)

Назначение:

Чтение размера буфера DMA.

Действие:

Функция забирает значение размера буфера DMA из внутренних переменных драйвера.

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pBufSize	Указатель на переменную в которую будет считано значение	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetDMASize(){
  CMIL1553DeviceEx obj;
  UINT32 nSize;
  obj.Open(0);
  if (obj.GetDatabufSize(&nSize))
    //В переменной nSize появилось
      значение размера буфера DMA.
  Else
    //Призошла ошибка.
  Obj.Close();
  return true;}
```

4.1.4. BOOL GetDatabufPhysAddr(UINT32 *pBufAddr)

Назначение:

Чтение физического адреса буфера DMA.

Действие:

Функция забирает начальное значение адреса буфера DMA из внутренних переменных драйвера.

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pBufAddr	Указатель на переменную в которую будет считано значение	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetDmaAddress(){
  CMIL1553DeviceEx obj;
  UINT32 nAddress;
  obj.Open(0);
  if (obj.GetDatabufPhysAddr(&nAddress))
    //В переменной nAddress появилось
      начальное значение адреса буфера DMA.
  Else
    //Призошла ошибка.
  Obj.Close();
  return true;
}
```

4.1.5. BOOL GetBarPhysAddr(UINT32 *pBarAddr)

Назначение:

Чтение начала физического адреса пространства регистров BAR.

Действие:

Функция забирает начальное значение адреса буфера DMA из внутренних переменных драйвера.

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pBarAddr	Указатель на переменную в которую будет считано значение	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetBarAddress(){
  CMIL1553DeviceEx obj;
  UINT32 nAddress;
  obj.Open(0);
  if (obj.GetBarPhysAddr(&nAddress))
    //В переменной nAddress появилось
      начальное значение адреса BAR.
  Else
    //Призошла ошибка.
  Obj.Close();
  return true;
}
```

4.1.6. BOOL GetPCILocation(UINT32 *pBus, UINT32 *pSlot)

Назначение:

Чтение номера шины и слота, занимаемые устройством.

Действие:

Функция забирает значения номера шины и слота из внутренних переменных драйвера.

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pBus	Указатель на переменную в которую будет считано значение	
UINT32 *pSlot	Указатель на переменную в которую будет считано значение	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetPCILoc(){
  CMIL1553DeviceEx obj;
  UINT32 nBus, nSlot;
  obj.Open(0);
  if (obj.GetPCILocation(&nBus, &nSlot))
    //В переменной nBus появилось значение
      занимаемой шины, а в nSlot занимаемого слота.
  Else
    //Призошла ошибка.
  Return true;
}
```

4.1.7. BOOL GetRevision(UINT32 *pRevision)

Назначение:

Чтение номера ревизии устройства.

Действие:

Функция читает значение ревизии устройства из конфигурационного адресного пространства PCI.

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 * pRevision	Указатель на переменную в которую будет считано значение	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetRevision(){
  CMIL1553DeviceEx objEx;
  UINT32 nRev;
  objEx.Open(0);
  if (objEx.GetRevision(&nRev))
    //В переменной nRev появилось
      значение ревизии устройства.
  Else
    //Призошла ошибка.
  objEx.Close();
  return true;
}
```

4.1.8. BOOL GetDriverCompileDate(std::string *pStr)

Назначение:

Чтение текстовой строки с информацией о времени и дате компиляции драйвера.

Действие:

Функция забирает значение времени и даты компиляции драйвера из внутренних переменных драйвера.

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
std::string *pStr	Указатель на переменную в которую будет скопирован текст	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetDriverCompileDate(){
  CMIL1553DeviceEx objEx;
  string sStr;
  objEx.Open(0);
  if (objEx.GetDriverCompileDate(&sStr))
   //В переменной sStr появилась
    дата и время компиляции драйвера.
  Else
    //Призошла ошибка.
  objEx.Close();
  return true;
}
```

4.1.9. BOOL WriteBar(UINT32 nAddr, const void *pData, UINT32 nSize)

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 WriteReg.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nAddr	Адрес начала блока памяти для записи	1000h – 201Ch
const void *pData	Указатель на буфер с записываемыми данными	
UINT32 nSize	Размер данных	1 – 4128 байт

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testWriteBar(){
  CMIL1553DeviceEx objEx;
  UINT32 nAddr = 0x0, nData[2];
  nData[0] = 0x11223344;
  nData[1] = 0x55667788;
  objEx.Open(0);
  if (objEx.WriteBar(nAddr, &nData, 8))
    //Данные записаны по желаемомуадресу в BAR.
  Else
    //Призошла ошибка.
  objEx.Close();
return true;
}
```

4.1.10. BOOL ReadBar(UINT32 nAddr, void *pData, UINT32 nSize)

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 функцию ReadReg.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nAddr	Адрес начала блока памяти для чтения	1000h – 201Ch
const void *pData	Указатель на буфер куда будут записываться прочтенные данные	
UINT32 nSize	Размер данных	1 – 4128 байт

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testReadBar(){
  CMIL1553DeviceEx objEx;
  UINT32 pData[2], nAddr = 0x0;
  objEx.Open(0);
  if (objEx.ReadBar(nAddr, &pData, 8))
    //В переменной pData появились
    данные прочтённые из желаемого адреса BAR.
  Else
    //Призошла ошибка.
  objEx.Close();
  return true;
}
```

4.1.11. BOOL ReadReg(UINT32 nAddr, UINT32 *pData)

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 ReadBar.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nAddr	Адрес начала блока памяти для чтения	1000h – 201Ch
UINT32 *pData	Указатель на переменную в которую будет считаны данные	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testReadReg(UINT32 nAddr){
  CMIL1553DeviceEx objEx;
  UINT32 pData;
  objEx.Open(0);
  if (objEx.ReadReg(nAddr, &pData))
    //В переменной pData появились
    данные прочтённые из желаемого адреса BAR.
  Else
    //Призошла ошибка.
  objEx.Close();
  return true;
}
```

4.1.12. BOOL WriteReg(UINT32 nAddr, UINT32 nData)

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 WriteBar.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nAddr	Адрес начала блока памяти для записи	1000h – 201Ch
UINT32 pData	Переменная с записываемыми данными	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testWriteReg(){
  CMIL1553DeviceEx objEx;
  UINT32 nAddr = 0x2000, nData = 0;
  objEx.Open(0);
  if (objEx.WriteReg(nAddr, nData))
    //Данные записаны по желаемому адресу в BAR.
  Else
    //Призошла ошибка.
  objEx.Close();
  return true;
}
```

4.2. Функции конфигурирования устройства

4.2.1. BOOL EnableDMA()

Назначение:
Разрешение работы DMA.

Действие:
Нулевой бит регистра DMA_DATA_BASE* (адрес 1000h) устанавливается в единицу.
*См. Раздел 5.1.1 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
После загрузки операционной системы работа не разрешена.

Входные данные

-

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testEnableDMA(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.EnableDMA())
    //Работа DMA разрешена
  else
    //Призошла ошибка.
  Return true;
}
```

4.2.2. BOOL DisableDMA()

Назначение:

Запрет работы DMA.

Действие:

Нулевой бит регистра DMA_DATA_BASE* (адрес 1000h) сбрасывается в ноль.
*См. См. Раздел 5.1.1 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

-

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testDisableDMA(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.DisableDMA())
    //Работа DMA запрещена
  else
    //Призошла ошибка.
  Return true;
     }
```

4.2.3. BOOL EnableCanonicalDMA()

Назначение:

Включение режима “CanonicalDMA”* работы с DMA.
*см. раздел 2.1 данного документа.

Действие:

Один программный счётчик вычитанных пакетов данных DMA обнуляется. Он заменяется на два, каждый из которых приравнивается к регистровому счётчику пакетов (регистр DMA_INDEX* (адрес 0x1008)) и содержит количество пакетов по соответствующей шине.
Включается режим “CanonicalDMA”, после которого становятся доступны новые функции и недоступны некоторые старые. Режим включается для обоих каналов одновременно.
*См. Раздел 5.1.2 документа “*Руководство по программированию модуля “mPCIe-1553RT2*”.

Примечание:

Для того чтобы понять работает какая-либо функция с данным режимом или нет, смотрите раздел “примечание” просматриваемой функции. Если про данный режим ничего не сказано, значит функция может работать как с данным режимом, так и без него.

Входные данные

-

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testEnableCanonical(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.EnableCanonicalDMA())
    //Режим “CanonicalDMA” включён.
  Else
    //Призошла ошибка.
  Return true;
}
```

4.2.4. BOOL DisableCanonicalDMA()

Назначение:

Выключение режима “CanonicalDMA”* работы с DMA.
*см. раздел 2.1 данного документа.

Действие:

Два программных счётчика пакетов, содержащие количество прочитанных пакетов по соответствующему каналу обнуляются. Вместо них возвращается один счётчик на оба канала. Он приравнивается к регистровому счётчику пакетов (регистр DMA_INDEX* (адрес 0x1008)). Выключается режим “CanonicalDMA”, после которого становятся доступны новые функции и недоступны некоторые старые. Режим включается для обоих каналов одновременно.
*См. Раздел 5.1.2 документа “*Руководство по программированию модуля “mPCIe-1553RT2*”.

Примечание:

Для того чтобы понять работает какая-либо функция без данного режима или нет, смотрите раздел “примечание” просматриваемой функции. Если про данный режим ничего не сказано, значит функция может работать как с данным режимом, так и без него.

Входные данные

-

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testDisableCanonical(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.DisableCanonicalDMA())
    //Режим “CanonicalDMA” выключён.
  Else
    //Призошла ошибка.
  Return true;
}
```

4.2.5. UINT32 EnableCanonicalChMem(UINT32 nCh)

Назначение:

Включение режима “CanonicalChMem”* работы с буферами передачи.
*см раздел 2.2 данного документа.

Действие:

Биты 15 и 16 регистра MIL$n^{(*)}$_CTRL_REG_PCI*** сбрасываются в ноль.
Также устанавливается в единицу 28 бит всех регистров MIL$n^{(*)}$_RT_TR_REG$m^{(**)}$****.
Включается режим “CanonicalChMem ”, после которого становятся доступны новые функции и недоступны некоторые старые. Режим включается только для запрашиваемого канала.
*n - – номер запрашиваемого канала.
**m – номер запрашиваемого подадреса
***См. Раздел 5.1.5 документа “*Руководство по программированию модуля “mPCIe-1553RT2*”.
**** см. Раздел 6.1.13 документа “*Руководство по программированию модуля “mPCIe-1553RT2*”.

Примечание:

Для того чтобы понять работает какая-либо функция с данным режимом или нет, смотрите раздел “примечание” просматриваемой функции. Если про данный режим ничего не сказано, значит функция может работать как с данным режимом, так и без него.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nCh

Пример вызова:

```
bool testEnableCanonicalChMem(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nCh = 0;
  if (objEx.EnableCanonicalChMem(nCh))
    //Режим “CanonicalChMem” включён.
  Else
    //Призошла ошибка.
  Return true;
}
```

4.2.6. UINT32 DisableCanonicalChMem(UINT32 nCh)

Назначение:

Выключение режима “CanonicalChMem”* работы с буферами передачи.
*см раздел 2.2 данного документа.

Действие:

Выключается режим “CanonicalChMem ”, после которого становятся доступны новые функции и недоступны некоторые старые. Режим выключается только для запрашиваемого канала.
15 и 16 биты регистра MIL$n^{(*)}$_CTRL_REG_PCI***, а также 31 биты регистров MIL$n^{(*)}$_RT_TR_REG$m^{(**)}$**** установленные при включении режима не сбрасываются.
*n – номер запрашиваемого канала.
**m – номер запрашиваемого подадреса
***См. Раздел 5.1.5 документа “*Руководство по программированию модуля “mPCIe-1553RT2*”.
**** см. Раздел 6.1.13 документа “*Руководство по программированию модуля “mPCIe-1553RT2*”.

Примечание:

Для того чтобы понять работает какая-либо функция с данным режимом или нет, смотрите раздел “примечание” просматриваемой функции. Если про данный режим ничего не сказано, значит функция может работать как с данным режимом, так и без него.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nCh

Пример вызова:

```
bool testDisableCanonicalChMem(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nCh = 0;
  if (objEx.DisableCanonicalChMem(nCh))
    //Режим “CanonicalChMem” включён.
  Else
    //Призошла ошибка.  Return true;
}
```

4.2.7. UINT32 EnableSubAddr(UINT32 nCh, UINT32 nRT, UINT32 nNum)

Назначение:

Разрешение подадреса.

Действие:

В зависимости от выбранного канала, а также в зависимости от выбранного режима передачи или приёма и номера подадреса функция устанавливает в единицу 31 бит регистров MIL$n^{(*)}$_RT_RCV_REG$m^{(**)}$*** или MIL$n^{(*)}$_RT_TR_REG$m^{(**)}$****.

*n – номер запрашиваемого канала.

**m – номер запрашиваемого подадреса

*** см. Раздел 6.1.12 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

**** см. Раздел 6.1.13 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 nRT	Выбор регистра. Если 1, то регистр передачи RT_TR_REGn, если 0, то регистр приёма RT_RCV_REGn.	0-1
UINT32 nNum	Номер подадреса	1-31

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nRT
4	Неправильное значение параметра nNum
5	Неправильное значение параметра nCh

Пример вызова:

```
bool testEnableSubAddr(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nCh = 0,
      nRT = 0,
      nNum = 3;
  if (objEx.EnableSubAddr(nCh, nRT, nNum))
    //Подадрес номер 3 канала номер 0 был разрешён
  else
    //Произошла ошибка
  return true;
}
```

4.2.8. UINT32 DisableSubAddr(UINT32 nCh, UINT32 nRT, UINT32 nNum)

Назначение:

Запрет подадреса.

Действие:

В зависимости от выбранного канала, а также в зависимости от выбранного режима передачи или приёма и номера подадреса функция сбрасывает в ноль 31 бит регистров MIL$n^{(*)}$_RT_RCV_REG$m^{(**)}$*** или MIL$n^{(*)}$_RT_TR_REG$m^{(**)}$****.

*n – номер запрашиваемого канала.

**m – номер запрашиваемого подадреса

*** см. Раздел 6.1.12 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

**** см. Раздел 6.1.13 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 nRT	Выбор регистра. Если 1, то регистр передачи RT_TR_REGn, если 0, то регистр приёма RT_RCV_REGn.	0-1
UINT32 nNum	Номер подадреса	1-31

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nRT
4	Неправильное значение параметра nNum
5	Неправильное значение параметра nCh

Пример вызова:

```
bool testDisableSubAddr(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nCh = 0,
      nRT = 0,
      nNum = 3;
  if (objEx.DisableSubAddr(nCh, nRT, nNum))
    //Подадрес номер 3 канала номер 0 был запрещён
  else
    //Произошла ошибка
  return true;
}
```

4.2.9. UINT32 WorkEnable(UINT32 nCh, bool bFlag)

Назначение:
Разрешение/запрет работы устройства.

Действие:
В зависимости от выбранного канала и значения переменной bFlag, функция либо устанавливает в единицу 23 бит регистра MIL$n^{(*)}$_CTRL_REG_PCI**, либо сбрасывает его в ноль.
*n – номер запрашиваемого канала.
** см. Раздел 5.1.5 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
Разрешение работы следует выполнять самым последним действием при инициализации платы.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
bool bFlag	Значение TRUE означает разрешение работы, FALSE запрет.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение канала

Пример вызова:

```
bool testWorkEnable(){
  CMIL1553DeviceEx objEx;
  bool Flag = true;
  objEx.Open(0);
  if(objEx.WorkEnable(0, Flag))
    //Работа канала 0 разрешена.
  Else
    //Произошла ошибка
  bjEx.Close();
  return true;
}
```

4.2.10. UINT32 SetRTAddress(UINT32 nCh, UINT32 Address)

Назначение:

Выбор адреса удалённого устройства.

Действие:

В зависимости от выбранного канала функция записывает желаемый адрес удалённого устройства в биты 5…9 регистра MIL$n^{(*)}$_CTRL_REG_PCI**, а также при необходимости устанавливает в единицу 10 бит этого же регистра.

*n – номер запрашиваемого канала.

** см. Раздел 5.1.5 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
bool bFlag	Значение TRUE означает разрешение работы, FALSE запрет.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение канала

Пример вызова:

```
bool testSetRTAddress(){
  CMIL1553DeviceEx objEx;
  UINT32 nAddress = 3;
  objEx.Open(0);
  if(objEx.SetRTAddress(0, nAddress))
    //Адрес удалённого устройства записан в регистр
  else
    //Произошла ошибка
  objEx.Close();
  return true;
}
```

4.2.11. UINT32 BusSelection(UINT32 nCh, bool bFlag)

Назначение:
Включение/выключение шины устройства.

Действие:
В зависимости от выбранного канала и значения переменной bFlag функция:
Для 0 канала:
либо устанавливает в единицу 2 бит регистра MIL$n^{(*)}$_CTRL_REG_PCI**, либо сбрасывает его в ноль.
Для 1 канала:
либо устанавливает в единицу 3 бит регистра MIL$n^{(*)}$_CTRL_REG_PCI**, либо сбрасывает его в ноль.
*n – номер запрашиваемого канала.
** см. Раздел 5.1.5 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
У каждого канала только одна шина данных. Она либо включается, либо выключается.

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
bool bFlag	Значение TRUE означает включение шины, FALSE выключение	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение канала

Пример вызова:

```
bool testBusSelection(){
  CMIL1553DeviceEx objEx;
  bool Flag = true;
  objEx.Open(0);
  if(objEx.BusSelection(0, Flag))
    //Шина данных канала 0 включена
  else
    //Произошла ошибка
  objEx.Close();
  return true;
}
```

4.2.12. UINT32 SetTimer(UINT32 nCh, UINT32 Timer)

Назначение:

Включение/выключение и выбор шага таймера.

Действие:

В зависимости от выбранного канала функция записывает желаемый шаг таймера в регистр MIL$n^{(*)}$_CTRL_REG_PCI**.

*n – номер запрашиваемого канала.

** см. Раздел 5.1.5 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 Timer	Выбранное значение шага таймера.	1 мкс, 2 мкс, 4 мкс, 8 мкс, 16 мкс, 32 мкс, 64 мкс '0' – таймер выключен

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение шага таймера
4	Неправильное значение канала

Пример вызова:

```
bool testSetTimer(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.SetTimer(0, 16))
    //значение инкремента основного таймера
       выставлено в 16 мкс
  else
    //Произошла ошибка
  objEx.Close();
  return true;

}
```

4.3. Функции для работы с данными

4.3.1. BOOL GetProgCount(UINT32 *nCount)

Назначение:
Чтение программного счётчика записанных блоков данных в DMA.

Действие:
Функция забирает значение счётчика из внутренних переменных драйвера.

Примечание:
Не доступна в режиме "CanonicalDMA". Воспользуйтесь функцией DisableCanonicalDMA().

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *nCount	Указатель на переменную в которую будут считаны данные	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetRrogCount(){
  CMIL1553DeviceEx objEx;
  UINT32 nCount;
  objEx.Open(0);
  if (objEx.GetProgCount(&nCount))
    //В переменной nCount появилось
      программного счётчика записанных блоков данных.
  Else
    cout<<"I/O error";
  objEx.Close();
  return true;
}
```

4.3.2. BOOL GetDMAReadyStatus(UINT32 *bFlag)

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 bFlag вернётся 1. Если равны, то новых данных нет и в bFlag вернётся 0.
*См. Раздел 5.1.2 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
Не доступна в режиме "CanonicalDMA". Воспользуйтесь функцией DisableCanonicalDMA().

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *bFlag	Указатель на переменную в которую будут считаны данные	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова совместно с функцией ReadDMADataOneBlock:

```
bool testReadData(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 flag=0;
  UINT32 buf[32];

  for (int i=0; i<2000; i++){
    objEx.GetDMAReadyStatus(&flag);
    if (flag){
      objEx.ReadDMADataOneBlock(&buf);
      //В перменной buf появился блок данных из DMA
      break;
    }
    else
       Новых данных нет
  }
  objEx.Close();
  return true;
}
```

4.3.3. BOOL ClearDMA()

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-1553RT2*".

Действие:

-

Примечание:

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

Входные данные

-

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testClearDMA(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ClearDMA())
    //Счётчик уравнен, функция GetDMAReadyStatus
      будет возвращать 0.
  Else
    //Произошла ошибка
  objEx.Close();
  return true;
}
```

4.3.4. BOOL GetNBlockRawDMA(UINT32 *nValue);

Назначение:

Чтение количества новых блоков данных, накопленных в буфере DMA устройства по всем каналам (блок – 128 байт).

Действие:

Функция вычитает из значения регистра DMA_INDEX*(адрес 0x1008) значение программного счётчика вычитанных пакетов. Полученное количество блоков возвращается в переменной nValue.

*См. Раздел 5.1.2 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

Не доступна в режиме "CanonicalDMA". Воспользуйтесь функцией DisableCanonicalDMA().

Входные данные

Переменная	Описание	Диапозон значений
UINT32 * nValue	Указатель на переменную в которую будут считаны данные	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetNBlockRawDMA(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nBlocks;

  if (objEx.GetNBlockRawDMA(&nBlocks))
    //В переменной nBlocks появилось количество новых блоков
данных.
  Else
    //Произошла ошибка
  return true;
}
```

4.3.5. UINT32 ReadDMADataOneBlock (void *pData)

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 GetDMAReadyStatus. 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:

Не доступна в режиме “CanonicalDMA”. Воспользуйтесь функцией DisableCanonicalDMA().

Входные данные

Переменная	Описание	Диапозон значений
void *pData	Указатель на буфер памяти в который будут считаны данные	Размер буфера не менее 128 байт

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Размер буфера pData больше 128 байт

Пример вызова совместно с функцией 2.4.2 GetDMAReadyStatus:

```
bool testReadData(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 flag=0;
  UINT32 buf[32];

  for (int i=0; i<2000; i++){
    objEx.GetDMAReadyStatus(&flag);
    if (flag){
      objEx.ReadDMADataOneBlock(&buf);
      //В перменной buf появился блок данных из DMA
      break;
    }
    else
       //Новых данных нет
  }
  objEx.Close();
  return true;
}
```

4.3.6. BOOL ReadDMADataBlocks(void *pData, UINT32 nBlocks, UINT32 *pBlocks)

Назначение:

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

Действие:

Чтобы воспользоваться функцией, рекомендуется узнать количество новых блоков данных с помощью функции GetNBlockRawDMA. Значение возвращенное этой функцией, использовать в качестве входного параметра для переменной nBlocks.
После выполнения функции указатель буфера DMA сдвигается на соответствующую количеству прочитанных данных позицию.

Примечание:

Не доступна в режиме "CanonicalDMA". Воспользуйтесь функцией DisableCanonicalDMA().

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pData	Указатель на переменную в которую будут считаны данные	
UINT32 nBlocks	Запрашиваемое количество блоков данных для чтения	
UINT32 *pBlocks	Указатель на переменную в которую будет записано считанное количество новых блоков данных. Оно может быть меньше чем запрашиваемое количество.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testReadDMADataBlocks(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 buf[64];
  UINT32 nBlocks;
  UINT32 nSize = 2;

  if (objEx.ReadDMADataBlocks(&buf, nSize, &nBlocks))
     //Если в переменной nBlocks значение не равное нулю,
   //то в переменной buf лежат вычитанные блоки данных.
  Else
    //Произошла ошибка;
  return true;
}
```

4.3.7. UINT32 ReadDMACanonicalDataBlocks(void *pData, UINT32 nCh, UINT32 nBlocks, UINT32 *pBlocks);

Назначение:

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

Действие:

По номеру запрашиваемого канала функция перебирает блоки данных в DMA, проверяя из какого канала пришли данные. Если из нужного то она кладёт данные в переменную pData до тех пор, пока либо не наберёт запрашиваемое количество блоков, либо блоки с новыми данными не закончатся.

Примечание:

Доступна только в режиме "CanonicalDMA". Воспользуйтесь функцией EnableCanonicalDMA().

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pData	Указатель на переменную в которую будут считаны данные	
UINT32 nCh	Номер шины, пришедшие данные с которой следует вычитать.	0-1
UINT32 nBlocks	Запрашиваемое количество блоков данных для чтения	
UINT32 *pBlocks	Указатель на переменную в которую будет записано считанное количество новых блоков данных. Оно может быть меньше чем запрашиваемое количество.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение шины.

Пример вызова на след. Странице

Пример вызова:

```
bool testReadDMAChDataBlocks(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 buf[64], nBlocks, nRes;
  UINT32 nSize = 2, nCh = 1;

  nRes = objEx.ReadDMACanonicalDataBlocks(&buf, nCh, nSize,
&nBlocks);
  switch (nRes){
    case 1:
      //В nBlocks появилось количество прочитанных данных и
     //в переменной bufсами данные.
      Return true;
      break;
    case 2:
      cout<<”I/O error\n”;
      return false;
      break;
    case 3:
      cout<<”Invalid bus number\n”;
      return false;
      break;
    default:
    return false;}
```

4.3.8. UINT32 ReadRegSubAddr(UINT32 nCh, UINT32 nRT, UINT32 nNum, UINT32 *pData)

Назначение:

Чтение данных из регистра подадреса.

Действие:

В зависимости от выбранного канала, а также в зависимости от выбранного режима передачи или приёма и номера подадреса функция вычитывает регистры MIL$n^{(*)}$_RT_RCV_REG$m^{(**)}$*** или MIL$n^{(*)}$_RT_TR_REG$m^{(**)}$****.

*n – номер запрашиваемого канала.

**m – номер запрашиваемого подадреса

*** см. Раздел 6.1.12 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

**** см. Раздел 6.1.13 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 nRT	Выбор регистра. Если 1, то регистр передачи RT_TR_REGn, если 0, то регистр приёма RT_RCV_REGn.	0-1
UINT32 nNum	Номер подадреса	1-31
UINT32 *pData	Указатель на переменную в которую будут считаны данные	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nRT
4	Неправильное значение параметра nNum
5	Неправильное значение параметра nCh

Пример вызова:

```
bool testReadRegSubAddr(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nCh = 0,
      nRT = 0,
      nNum = 3,
      nData;
  if (objEx.ReadRegSubAddr(nCh, nRT, nNum, &nData))
    //в переменной nData появилось значение нужного регистра
  else
    //Произошла ошибка
  return true;
}
```

4.3.9. UINT32 WriteRegSubAddr(UINT32 nCh, UINT32 nRT, UINT32 nNum, UINT32 nData)

Назначение:

Запись данных в регистр подадреса.

Действие:

В зависимости от выбранного канала, а также в зависимости от выбранного режима передачи или приёма и номера подадреса функция записывает в регистры MIL$n^{(*)}$_RT_RCV_REG$m^{(**)}$*** или MIL$n^{(*)}$_RT_TR_REG$m^{(**)}$**** данные из перменной nData.

*n – номер запрашиваемого канала.

**m – номер запрашиваемого подадреса

*** см. Раздел 6.1.12 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

**** см. Раздел 6.1.13 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 nRT	Выбор регистра. Если 1, то регистр передачи RT_TR_REGn, если 0, то регистр приёма RT_RCV_REGn.	0-1
UINT32 nNum	Номер подадреса	1-31
UINT32 nData	Данные для записи.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nRT
4	Неправильное значение параметра nNum
5	Неправильное значение параметра nCh

Пример вызова:

```
bool testWriteRegSubAddr(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nCh = 0,
      nRT = 0,
      nNum = 3,
      nData = 1;
  if (objEx.WriteRegSubAddr(nCh, nRT, nNum, nData))
    //В регистр приёма 0 канала подадреса 3 записаны данные
    // из переменной nData
  else
    //Произошла ошибка
  return true;
}
```

4.3.10. UINT32 ReadRegSubAddrBuf(UINT32 nCh, UINT32 nBufNum, UINT32 nNum, void *pData)

Назначение:

Чтение данных из буфера подадреса.

Действие:

В зависимости от выбранного канала, а также в зависимости от выбранного номера буфера и номера подадреса функция вычитывает буферы MIL$n^{(*)}$_RT_DATA_BUF0$m^{(**)}$*** или MIL$n^{(*)}$_RT_DATA_BUF1$m^{(**)}$****. Размер буфер, в который будут вычитаны данные, должен быть 64 байта.

*n – номер запрашиваемого канала.

**m – номер запрашиваемого подадреса

*** см. Раздел 6.1.14 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

**** см. Раздел 6.1.15 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

*Доступна только в режиме "*CanonicalChMem*". Воспользуйтесь функцией EnableCanonicalChMem().*

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 nBufNum	Выбор буфера. Если 0, то RT_DATA_BUF0n , если 1, то RT_DATA_BUF1n	0-1
UINT32 nNum	Номер подадреса	1-31
void *pData	Указатель на буфер куда будут считаны данные	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nBufNum
4	Неправильное значение параметра nNum
5	Неправильное значение параметра nCh

Пример вызова на след. Странице.

Пример вызова:

```
bool testReadRegSubAddrBuf(){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nBufNum = 0,
         nNum = 3,
         nCh = 0,
         nData,
         nRes;
  nRes = objEx.ReadRegSubAddrBuf(nCh, nBufNum, nNum, &nData);
  switch (nRes){
    case 1:
      //В nData появились вычитанные данные
    case 2:
      //Произолша ошибка ввода/вывода
    case 3:
    case 4:
      //Неправильные входные данные
    default:
      return false;
  }}
```

4.3.11. UINT32 WriteRegSubAddrBuf(UINT32 nCh, UINT32 nBufNum, UINT32 nNum, const void *pData, UINT32 nSize)

Назначение:

Запись данных в буфер подадреса.

Действие:

В зависимости от выбранного канала, а также в зависимости от выбранного номера буфера и номера подадреса функция записывает в буферы MIL$n^{(*)}$_RT_DATA_BUF0$m^{(**)}$*** или MIL$n^{(*)}$_RT_DATA_BUF1$m^{(**)}$**** данные из блока pData.
*n - – номер запрашиваемого канала.
**m – номер запрашиваемого подадреса
*** см. Раздел 6.1.14 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".
**** см. Раздел 6.1.15 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

Доступна только в режиме "CanonicalChMem". Воспользуйтесь функцией EnableCanonicalChMem().

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала.	0-1
UINT32 nBufNum	Выбор буфера. Если 0, то RT_DATA_BUF0n , если 1, то RT_DATA_BUF1n	0-1
UINT32 nNum	Номер подадреса	1-31
void *pData	Блок данных для записи.	
UINT32 nSize	Размер блока данных	0-64

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Неправильное значение параметра nBufNum
4	Неправильное значение параметра nNum
5	Неправильное значение параметра nCh

Пример вызова на след. Странице.

Пример вызова:

```
bool testWriteRegSubAddrBuf (){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 nBufNum = 0,
         nNum = 3,
         nCh = 0,
         nData,
         nRes;
 nRes = objEx. WriteRegSubAddrBuf(nCh,  nBufNum, nNum, &nData,
4);
  switch (nRes){
    case 1:
      //Данные были записаны
    case 2:
      //Произолша ошибка ввода/вывода
    case 3:
    case 4:
      //Неправильные входные данные
    default:
      return false;
  }}
```

4.4. Функции прерываний

4.4.1. BOOL GetIntFlags(UINT32 *pIntFlags, UINT32 *nRT1Value, UINT32 *nRT0Value)

Назначение:

Чтение флагов прерываний.

Действие:

В переменную pIntFlags читается значение регистра INTERRUPT* (адрес 100Ch). После этого флаги сбрасываются.

В переменные nRT0Value и nRT1Value читаются значения 20-23 бит регистров MIL0_HW_STAT_REG2** и MIL1_HW_STAT_REG2** соответственно. Флаги переписываются последующими данными.

* см. Раздел 5.1.3 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

** см. Раздел 6.1.10 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
UINT32 *pIntFlags	Указатель на переменную в которую будет считано значение	
UINT32 *nRT1Value	Указатель на переменную в которую будет считано значение	
UINT32 *nRT0Value	Указатель на переменную в которую будет считано значение	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testGetIntFlags(){
  CMIL1553DeviceEx objEx;
  UINT32 nIntFlags, nRT1Value, nRT0Value;
  objEx.Open(0);
  if (objEx.GetIntFlags(&nIntFlags, &nRT1Value, &nRT0Value))
    //В переменные считаны значения
  else
    //Произошла ошибка
  objEx.Close();
  return true;
}
```

4.4.2. BOOL ChangeINT_HDAT_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_HDAT.

Действие:

Нулевой бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testChangeINT_HDAT_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeINT_HDAT_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.3. BOOL ChangeINT_QDAT_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_QDAT.

Действие:

Первый бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testChangeINT_QDAT_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeINT_QDAT_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.4. BOOL ChangeINT_FLASH_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_FLASH.

Действие:

Третий бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testChangeINT_FLASH_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeINT_FLASH_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.5. BOOL ChangeINT_RT0_SADDR_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_RT0_SADDR.

Действие:

Четвёртый бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testChangeINT_RT0_SADDR_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeRT0_SADDR_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.6. BOOL ChangeINT_RT0_MC_ERR_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_RT0_MC_ERR.

Действие:

Пятый бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова:

```
bool testChangeINT_RT0_MC_ERR_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeINT_RT0_MC_ERR_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.7. BOOL ChangeINT_RT1_SADDR_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_RT1_SADDR.

Действие:

Шестой бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова на функции ChangeINT_RT0_SADDR_Work:

```
bool testChangeINT_RT0_SADDR_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeRT0_SADDR_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.8. BOOL ChangeINT_RT1_MC_ERR_Work(bool bFlag)

Назначение:

Разрешение/запрет прерывания INT_RT1_MC_ERR.

Действие:

Седьмой бит регистра INTERRUPT MASK* (адрес 1010h) либо устанавливается в единицу, либо сбрасывается в ноль.

* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:

-

Входные данные

Переменная	Описание	Диапозон значений
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
FALSE	Запрос от драйвера вернулся с ошибкой

Пример вызова на фукции ChangeINT_RT0_MC_ERR_Work:

```
bool testChangeINT_RT0_MC_ERR_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  if (objEx.ChangeINT_RT0_MC_ERR_Work(flag))
    //бит изменён
  else
    //произошла ошибка
  return true;
}
```

4.4.9. UINT32 ChangeRT_INT_TEN_Work(UINT32 nCh, bool bFlag)

Назначение:
Разрешение/запрет прерывания RT_INT_TEN.

Действие:
Нулевой бит регистра MIL$n^{(*)}$_RT_INT_REG** либо устанавливается в единицу, либо сбрасывается в ноль.
* n – номер канала
**см. Раздел 6.1.7 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
Действительно только при значении 1 бита 4 INT_RT0_SADDR или 6 INT_RT1_SADDR регистра INTERRUPT_MASK* (адрес 1010h) в зависимости от выбранного канала.
*см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала	0-1
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Не разрешён бит INT_RTn_SADDR
4	Не правильное значение параметра nCh

Пример вызова:

```
bool testChangeRT_INT_TEN_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 Result = objEx.ChangeRT_INT_TEN_Work(0, flag);
  switch (Result){
    case 1:
      cout<<"RT_INT_TEN was changed\n";
     break;
    case 2:
     cout<<"I/O error\n";
     break;
    case 3:
     cout<<"INT_RTn_SADDR is not enable\n";
     break;

    default:
       return false;
  }
  return true;
}
```

4.4.10. UINT32 ChangeRT_INT_REN_Work(UINT32 nCh, bool bFlag)

Назначение:
Разрешение/запрет прерывания RT_INT_REN.

Действие:
Первый бит регистра MIL$n^{(*)}$_RT_INT_REG** либо устанавливается в единицу, либо сбрасывается в ноль.
* n – номер канала
**см. Раздел 6.1.7 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
Действительно только при значении 1 бита 4 INT_RT0_SADDR или 6 INT_RT1_SADDR регистра INTERRUPT_MASK* (адрес 1010h) в зависимости от выбранного канала.
* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала	0-1
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Не разрешён бит INT_RTn_SADDR
4	Не правильное значение параметра nCh

Пример вызова на фукции ChangeRT_INT_TEN_Work:

```
bool testChangeRT_INT_TEN_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 Result = objEx.ChangeRT_INT_TEN_Work(0, flag);
  switch (Result){
    case 1:
      cout<<"RT_INT_TEN was changed\n";
     break;
    case 2:
     cout<<"I/O error\n";
     break;
    case 3:
     cout<<"INT_RTn_SADDR is not enable\n";
     break;

    default:
       return false;
  }
  return true;
}
```

4.4.11. UINT32 ChangeRT_INT_ERR_Work(UINT32 nCh, bool bFlag)

Назначение:
Разрешение/запрет прерывания RT_INT_ERR.

Действие:
Третий бит регистра MIL$n^{(*)}$_RT_INT_REG** либо устанавливается в единицу, либо сбрасывается в ноль.
* n – номер канала
**см. Раздел 6.1.7 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
Действительно только при значении 1 бита 5 INT_RT0_MC_ERR или 6 INT_RT1_MC_ERR регистра INTERRUPT_MASK* (адрес 1010h) в зависимости от выбранного канала.
* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала	0-1
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Не разрешён бит INT_RTn_MC_ERR
4	Не правильное значение параметра nCh

Пример вызова на фукции ChangeRT_INT_TEN_Work:

```
bool testChangeRT_INT_TEN_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 Result = objEx.ChangeRT_INT_TEN_Work(0, flag);
  switch (Result){
    case 1:
      cout<<"RT_INT_TEN was changed\n";
     break;
    case 2:
     cout<<"I/O error\n";
     break;
    case 3:
     cout<<"INT_RTn_SADDR is not enable\n";
     break;

    default:
       return false;
  }
  return true;
}
```

4.4.12. UINT32 ChangeRT_INT_MC_Work(UINT32 nCh, bool bFlag)

Назначение:
Разрешение/запрет прерывания RT_INT_MC.

Действие:
Третий бит регистра MIL$n^{(*)}$_RT_INT_REG** либо устанавливается в единицу, либо сбрасывается в ноль.
* n – номер канала
**см. Раздел 6.1.7 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Примечание:
Действительно только при значении 1 бита 5 INT_RT0_MC_ERR или 6 INT_RT1_MC_ERR регистра INTERRUPT_MASK* (адрес 1010h) в зависимости от выбранного канала.
* см. Раздел 5.1.4 документа "*Руководство по программированию модуля "mPCIe-1553RT2*".

Входные данные

Переменная	Описание	Диапозон значений
UINT32 nCh	Номер канала	0-1
bool bFlag	Если TRUE, то прерывание разрешается, если FALSE, то запрещается.	

Расшифровка кодов ошибок

код	Расшифровка
2	Запрос от драйвера вернулся с ошибкой
3	Не разрешён бит INT_RTn_MC_ERR
4	Не правильное значение параметра nCh

Пример вызова на фукции ChangeRT_INT_TEN_Work:

```
bool testChangeRT_INT_TEN_Work(bool flag){
  CMIL1553DeviceEx objEx;
  objEx.Open(0);
  UINT32 Result = objEx.ChangeRT_INT_TEN_Work(0, flag);
  switch (Result){
    case 1:
      cout<<"RT_INT_TEN was changed\n";
     break;
    case 2:
     cout<<"I/O error\n";
     break;
    case 3:
     cout<<"INT_RTn_SADDR is not enable\n";
     break;

    default:
       return false;
  }
  return true;
}
```

5. С чего начать

5.1. Набор функций для инициализации устройства на приём.

Необходимый набор:
EnableDMA
SetRTAddress
WorkEnable
BusSelection

5.2. Набор функций для чтения принятых данных

Не забывайте включать и выключать режим “CanonicalDMA”(см. раздел 2.1 данного документа).

Для чтения всех блоков данныхподряд:
ReadDMADataBlocks
Для чтения данных только из нужного канала:
ReadDMACanonicalDataBlocks

5.3. Набор функций для инициализации устройства на передачу данных

Не забывайте включать и выключать режим “CanonicalChMem”(см. раздел 2.2 данного документа).

EnableCanonicalChMem
EnableSubAddr

5.4. Набор функций для передачи данных.

Для выбора режима передачи см. раздел 6.2 “Режимы работы при передаче данных” документа “*Руководство по программированию модуля “mPCIe-1553RT2*”

4. Обновление библиотеки

Версия библиотеки	Дата	Изменение
1.4	03.07.2013	- Изменены функции ReadRegSubAddrBuf и WriteRegSubAddrBuf. Теперь возможно читать и писать данные более 32 бит.
1.5	28.11.2016	- В архив добавлена библиотека Qt.

5. Обновление руководства

Версия документа	Дата	Изменение
1.2	08.02.2013	1. При старте драйвера запрещена работа DMA. Для разрешения воспользуйтесь функцией EnableDMA. 2. Добавлена возможность вычитывания данных из DMA не только по одному блоку, но и несколькими блоками одновременно. 3. Добавлена возможность поканального вычитывания данных из DMA. 4. Изменилось название функции с ReadDMADataBuf на ReadDMADataOneBlock. 5.Изменён вызов функции: WorkEnable; SetRTAddress; BusSelection; SetTimer; Больше нет функций с индексами 0 и 1. Теперь для обращения к каналам устройства, надо послать номер желаемого канала в качестве переменной nCh. 6.Функция SetTimer изменила тип с BOOL на UINT32. 7. Список новых функций: EnableDMA; DisableDMA; EnableCanonicalDMA; DisableCanonicalDMA; EnableCanonicalChMem; DisableCanonicalChMem; EnableSubAddr; DisableSubAddr; GetNBlockRawDMA; ReadDMADataBlocks; ReadDMACanonicalDataBlocks; ReadRegSubAddr; WriteRegSubAddr; ReadRegSubAddrBuf; WriteRegSubAddrBuf; Функции для работы с прерываниями 8. Добавлен новый раздел “Расшифровка названия библиотеки”.

Продолжение на следующей странице:

1.3	06.03.2013	1. Переделано описание всех функций. Появились разделы “Назначение” – краткое описание назначения вызова; “Действие” – описание действий вызова; “Примечание” – какие-либо важные заметки о действии вызова. “Пример вызова” – код для запуска вызова. 2. Описание режима “Canonical” вынесено из раздела 1 “Введение” в отдельный раздел “Режим “Canonical”. 3. Исправлена ошибка в функции EnableCanonicalChMem. Не записывались 15 и 16 биты.
3.4	24.02.2015	1. Добавлен новый раздел “Список доступных функций по версиям библиотек” 2. Изменения в оформлении документа
3.5	20.05.2015	Изменён пункт “Подключение библиотеки к разрабатываемому проекту”
3.6	28.11.2016	Изменён пункт “Подключение библиотеки к разрабатываемому проекту”