Современные устройства сбора данных

L-502/E-502/E16

Руководство программиста

Ревизия 1.1.13
Сентябрь 2024

Автор руководства:
Борисов Алексей

ООО “Л Кард”

117105, г. Москва, Варшавское ш., д. 5, корп. 4, стр. 2

тел.: +7 (495) 785-95-25
факс: +7 (495) 785-95-14

Адреса в Интернет:

http://www.lcard.ru

E-Mail:

Отдел продаж: sale@lcard.ru
Техническая поддержка: support@lcard.ru
Отдел кадров: job@lcard.ru

Таблица 1: Ревизии текущего документа

Ревизия	**Дата**	**Описание**
1.0.0	27.06.2012	Первая ревизия данного документа
1.0.1	22.11.2012	Добавлено описание использования библиотеки с программами на C# и в LabView, добавлено описание установки для ОС Linux, а также описание функций для циклического вывода
1.0.2	20.02.2013	Добавлена ссылка на исходные коды SDK. Описание установки пакетов для Linux вынесено в отдельный документ
1.0.3	16.02.2015	Исправлена последовательность шагов для работы с модулем при синхронном потоковом выводе. Добавлено примечание о передачи массивов в качестве выходных параметров в LabView
1.1.0	02.06.2015	Описание изменено в соответствии с изменениями, внесенными в библиотеку для поддержки модуля E-502 (введение общих и специализированных функций). Включено краткое описание различий модулей с программной стороны. Добавлены отдельные главы, описывающие настройку модуля при работе по Ethernet и поиск модулей в локальной сети.
1.1.1	06.07.2015	Добавлено описание возможности ожидания завершения установки циклического сигнала, добавленной в версии 1.1.2 библиотеки. В разделе отличий модулей при описании наличия ARM-контроллера в E-502 указан путь для скачивания обновлений прошивки с рекомендацией обновления.
1.1.2	10.07.2015	Добавлено описание нового алгоритма расчета максимального размера циклического сигнала для E-502 с прошивкой ARM 1.0.3 и выше
1.1.3	28.07.2015	Добавлено описание функций X502_SetExtRefFreqValue() и X502_GetRefFreqValue()
1.1.4	29.06.2016	Указано, что установка частоты вывода доступна в L-502, начиная с версии 0.5 прошивки ПЛИС. Рекомендация при синхронном выводе на ЦАП предворительно асинхронно установить начальные значения. Добавлено описание функций X502_CheckFeature() и X502_OutGetStatusFlags().
1.1.5	03.08.2016	Добавлено описание использования библиотеки в Visual Basic 6
1.1.6	23.08.2016	Изменена ссылка во вступлении на обновленное общее низкоуровневое описание программиста для L-502 и E-502

1.1.7	16.11.2016	Добавлено описание функций X502_CalcAdcFreq(), X502_CalcDinFreq(), X502_CalcOutFreq()
1.1.8	27.02.2015	При описании синхронного и асинхронного режимов работы добавлено описание ограничения их совместного использования при запуске синхронного ввода-вывода от внешнего сигнала
1.1.9	25.12.2019	Добавлено описание функции X502_IsOpened(). Добавлено общее описание процедуры открытия соединения с модулем, где в частности описана ситуация, когда соединение может остаться открытым, несмотря на возвращенную ошибку при его открытии
1.1.10	21.09.2020	Исправление ссылок на исходные коды. Переименование lqmeasstudio в X502Studio
1.1.11	26.07.2022	В информацию о модуле добавлены флаги для определения типа ЦАП и типа процессора
1.1.12	11.04.2024	Для E502-P1 добавлено описание функций для работы с метками времени
1.1.13	30.09.2024	Поддержка E16 со стороны x502api

Оглавление

1 О чем этот документ **11**

2 Установка и подключение библиотеки к проекту **12**
2.1 Подключение библиотеки при написании программы на языках C/C++ 13
2.2 Использование библиотеки в проекте на Delphi 13
2.3 Использование библиотеки в проекте на C# 14
2.4 Использование библиотеки в проекте LabView 16
2.5 Использование библиотеки в Visual Basic 6 17
2.6 64-битная версия библиотеки 17
2.7 Установка библиотеки и драйвера для ОС Linux 18
2.8 Исходные коды SDK 20

3 Общий подход к работе с библиотекой **21**
3.1 Отличия при работе с модулям L-502, E-502 и E16 21
3.1.1 Отличие возможностей модулей 21
3.1.2 Общие и специализированные функции для работы с модулем . . 23
3.1.3 Совместимость проектов, разработанных до введения библиотеки x502api 24
3.2 Общий алгоритм для работы с модулем 24
3.2.1 Работа с модулем при синхронном вводе 25
3.2.2 Работа с модулем при синхронном потоковом выводе 25
3.2.3 Работа с модулем при циклическом выводе 26
3.2.4 Работа с модулем при асинхронном вводе-выводе 26
3.3 Создание и освобождение описателя модуля 26
3.4 Открытие связи с модулем 27
3.4.1 Общее описание процедуры установки связи с модулем 27
3.4.2 Установка связи с модулем L-502 по интерфейсу PCI-Express . . . 28
3.4.3 Установка связи с модулем E-502 по интерфейсу USB 29
3.4.4 Установка связи с модулем E-502 по интерфейсу Ethernet 29
3.4.5 Установка связи с модулями с использованием записей о устройстве 29
3.5 Режимы работы с сигнальным процессором и без него 32
3.6 Установка настроек модуля 32
3.6.1 Настройка последовательности опроса каналов АЦП 33
3.6.2 Настройка частоты синхронного ввода/вывода 34
3.6.3 Коэффициент усреднения для логического канала (не реализовано в E16) 35
3.6.4 Настройка режимов синхронизации 36
3.7 Синхронный и асинхронный режимы работы 36
3.7.1 Асинхронный режим работы 37

3.7.2 Синхронный режим работы . . . 38
3.7.3 Циклический вывод . . . 40
3.7.4 Размер буфера и шаг для синхронного режима . . . 42
3.8 Особенности работы по интерфейсу Ethernet и настройка сетевых параметров . . . 43
3.9 Обнаружение модулей в локальной сети . . . 45

4 Константы, типы данных и функции библиотеки 47
4.1 Константы и перечисления . . . 47
4.1.1 Константы и макроопределения . . . 47
4.1.2 События поиска сетевых сервисов . . . 50
4.1.3 Коды ошибок библиотеки . . . 50
4.1.4 Интерфейс соединения с модулем . . . 55
4.1.5 Флаги, управляющие поиском присутствующих модулей . . . 55
4.1.6 Флаги для управления цифровыми выходами . . . 56
4.1.7 Константы для выбора опорной частоты . . . 56
4.1.8 поля io_mode в регистре io_hard . . . 56
4.1.9 Диапазоны измерения для канала АЦП . . . 56
4.1.10 Диапазоны измерения для канала АЦП E16 . . . 57
4.1.11 Режим измерения для логического канала . . . 57
4.1.12 Режимы синхронизации . . . 57
4.1.13 Флаги, управляющие обработкой принятых данных . . . 58
4.1.14 Флаги для обозначения синхронных потоков данных . . . 58
4.1.15 Константы, определяющие тип передаваемого отсчета из ПК в модуль . . . 58
4.1.16 Режим работы модуля . . . 59
4.1.17 Номера каналов ЦАП . . . 59
4.1.18 Флаги, используемые при выводе данных на ЦАП . . . 59
4.1.19 Номера каналов для передачи потоков данных . . . 59
4.1.20 Цифровые линии, на которых можно включить подтягивающие резисторы . . . 60
4.1.21 Флаги, определяющие наличие опций в модуле и наличие необходимых параметров . . . 60
4.1.22 Тип содержимого строки с расположением устройства . . . 62
4.1.23 Флаги для режима циклического вывода . . . 63
4.1.24 Дополнительные возможности модуля . . . 64
4.1.25 Флаги состояния для синхронного вывода . . . 64
4.2 Типы данных . . . 64
4.2.1 Запись о устройстве . . . 64
4.2.2 Калибровочные коэффициенты диапазона. . . . 65
4.2.3 Калибровочные коэффициенты модуля . . . 65
4.2.4 Информация о модуле L-502/E-502. . . . 65
4.2.5 Описатель конфигурации сетевого интерфейса . . . 66
4.2.6 Описатель контекста поиска устройств в сети . . . 66
4.2.7 Описатель сетевого сервиса . . . 66
4.2.8 Внутренняя информация записи о устройстве . . . 67
4.2.9 Описатель модуля . . . 67
4.2.10 Список серийный номеров . . . 67
4.3 Функции . . . 67

4.3.1 Функции для создания и освобождения описателя модуля 67
4.3.1.1 Создание описателя модуля 67
4.3.1.2 Освобождение описателя модуля 68
4.3.2 Функции для открытия и получения информации о модуле 68
4.3.2.1 Получение списка серийных номеров модулей L-502. . . . 68
4.3.2.2 Открытие модуля L-502 по его серийному номеру 69
4.3.2.3 Получение списка серийных номеров модулей E-502, подключенных по USB. 69
4.3.2.4 Работает аналогично E502_UsbGetSerialList, только для модулей E16. 70
4.3.2.5 Открытие модуля E-502, подключенного по USB, по его серийному номеру 70
4.3.2.6 Работает аналогично E502_OpenUsb, только для модулей E16. 70
4.3.2.7 Открытие модуля E-502 по IP-адресу 71
4.3.2.8 Открытие модуля E-16 по IP-адресу 71
4.3.2.9 Закрытие соединения с модулем 72
4.3.2.10 Проверка, открыто ли соединение с модулем 72
4.3.2.11 Получение информации о модуле 72
4.3.3 Функции для работы с записями об устройстве 73
4.3.3.1 Получить список записей, соответствующих подключенным модулям L502. 73
4.3.3.2 Получить список записей, соответствующих подключенным модулям E502. 74
4.3.3.3 Аналогично E502_UsbGetDevRecordsList, получить список записей, соответствующих подключенным модулям E16. 74
4.3.3.4 Аналогично E502_UsbGetDevRecordsList, получить список записей, соответствующих подключенным модулям E14-440. 74
4.3.3.5 Получить список записей, соответствующих подключенным модулям E502. 75
4.3.3.6 Создание записи о устройстве с указанным IP-адресом . 75
4.3.3.7 Создание записи о устройстве с указанным IP-адресом . 76
4.3.3.8 Установка TCP-порта управляющего соединения для записи о устройстве 76
4.3.3.9 Установка TCP-порта соединения передачи данных для записи о устройстве 77
4.3.3.10 Создание записи о устройстве по описателю сетевого сервиса 77
4.3.3.11 Открыть соединение с модулем по записи о устройстве . 78
4.3.3.12 Освобождение записей об устройствах 78
4.3.4 Функции для изменения настроек модуля 79
4.3.4.1 Передача установленных настроек в модуль 79
4.3.4.2 Установка параметров логического канала 79
4.3.4.3 Установка количества логических каналов 80
4.3.4.4 Получение количества логических каналов 80
4.3.4.5 Установка делителя частоты сбора для АЦП 80
4.3.4.6 Установка значения межкадровой задержки для АЦП . . 81

4.3.4.7 Установка делителя частоты синхронного ввода с цифровых линий. 81
4.3.4.8 Установка делителя частоты синхронного вывода 82
4.3.4.9 Установка частоты сбора АЦП 83
4.3.4.10 Установка частоты синхронного ввода с цифровых входов 84
4.3.4.11 Установка частоты синхронного вывода 84
4.3.4.12 Получить текущие значения частот сбора АЦП 85
4.3.4.13 Установка значения внутренней опорной частоты синхронизации 85
4.3.4.14 Установка значения внешней опорной частоты синхронизации 86
4.3.4.15 Установка значения для аналоговой синхронизации (только для E16) 86
4.3.4.16 Получение значения опорной частоты синхронизации . . 87
4.3.4.17 Установка режима генерации частоты синхронизации . . 87
4.3.4.18 Установка режима запуска частоты синхронизации . . . 88
4.3.4.19 Установить режим работы модуля 88
4.3.4.20 Получение текущего режима работы модуля 89
4.3.4.21 Установить коэффициенты для калибровки значений АЦП 89
4.3.4.22 Получение текущих калибровочных коэффициентов АЦП 90
4.3.4.23 Установить коэффициенты для калибровки значений ЦАП 90
4.3.4.24 Получение текущих калибровочных коэффициентов ЦАП 91
4.3.4.25 Рассчет частоты сбора АЦП 92
4.3.4.26 Рассчет частоты сбора АЦП 93
4.3.4.27 Рассчет частоты синхронного ввода с цифровых входов . 94
4.3.4.28 Рассчет частоты синхронного ввода с цифровых входов . 94
4.3.4.29 Рассчет частоты синхронного вывода 95
4.3.4.30 Рассчет частоты синхронного вывода 96
4.3.5 Функции асинхронного ввода-вывода 96
4.3.5.1 Асинхронный вывод данных на канал ЦАП 96
4.3.5.2 Асинхронный вывод данных на цифровые выходы 97
4.3.5.3 Асинхронный ввод значений с цифровых входов 98
4.3.5.4 Асинхронный ввод одного кадра АЦП 99
4.3.6 Функции для работы с синхронным потоковым вводом-выводом . 100
4.3.6.1 Разрешение синхронных потоков на ввод/вывод 100
4.3.6.2 Запрещение синхронных потоков на ввод/вывод 100
4.3.6.3 Получить значение, какие синхронные потоки разрешены 101
4.3.6.4 Запуск синхронных потоков ввода/вывода 101
4.3.6.5 Останов синхронных потоков ввода/вывода 101
4.3.6.6 Проверка, запущен ли синхронный ввод/вывод 102
4.3.6.7 Чтение данных АЦП и цифровых входов из модуля . . . 102
4.3.6.8 Передача потоковых данных ЦАП и цифровых выходов в модуль 103
4.3.6.9 Обработка принятых отсчетов АЦП от модуля 104
4.3.6.10 Обработка принятых от модуля данных 105
4.3.6.11 Обработка принятых от модуля данных с пользовательскими данными 106
4.3.6.12 Подготовка данных для вывода в модуль 107
4.3.6.13 Получить количество отсчетов в буфере потока на ввод . 108

4.3.6.14 Получить размер свободного места в буфере потока на вывод . . . 108
4.3.6.15 Получить номер следующего ожидаемого логического канала АЦП для обработки . . . 109
4.3.6.16 Начало подготовки вывода синхронных данных . . . 109
4.3.6.17 Начало загрузки циклического сигнала на вывод . . . 110
4.3.6.18 Установка ранее загруженного циклического сигнала на вывод . . . 110
4.3.6.19 Останов вывода циклического сигнала . . . 111
4.3.6.20 Проверка, завершена ли установка или останов циклического сигнала . . . 112
4.3.6.21 Чтение флагов статуса вывода . . . 113
4.3.6.22 Установка размера буфера для синхронного ввода или вывода . . . 113
4.3.6.23 Установка шага при передаче потока на ввод или вывод . 114
4.3.7 Функции для настройки сетевых параметров модуля E502 . . . 114
4.3.7.1 Получение текущего IP-адреса устройства . . . 114
4.3.7.2 Создание описателя конфигурации сетевого интерфейса . 114
4.3.7.3 Освобождение описателя конфигурации сетевого интерфейса. . . . 115
4.3.7.4 Чтение текущей сетевой конфигурации интерфейса . . . 115
4.3.7.5 Запись сетевой конфигурации интерфейса . . . 116
4.3.7.6 Копирование содержимого сетевой конфигурации интерфейса . . . 116
4.3.7.7 Определение, разрешен ли интерфейс Ethernet. . . . 117
4.3.7.8 Разрешение интерфейса Ethernet. . . . 117
4.3.7.9 Определение, разрешено ли автоматическое получение параметров IP. . . . 117
4.3.7.10 Разрешение автоматического получения параметров IP. . 118
4.3.7.11 Получить состояние автоматического получения параметров IP. . . . 118
4.3.7.12 Определение, разрешен ли пользовательский MAC-адрес 118
4.3.7.13 Определение, разрешен ли пользовательский MAC-адрес 119
4.3.7.14 Получение установленного статического IP-адреса . . . 119
4.3.7.15 Установка статического IP-адреса . . . 119
4.3.7.16 Получение установленной статической маски подсети . . 120
4.3.7.17 Установка статической маски подсети . . . 120
4.3.7.18 Получение установленного статического адреса шлюза . 120
4.3.7.19 Установка статического адреса шлюза . . . 121
4.3.7.20 Получение установленного пользовательского MAC-адреса 121
4.3.7.21 Установка пользовательского MAC-адреса . . . 121
4.3.7.22 Получение заводского MAC-адреса устройства . . . 122
4.3.7.23 Получение установленного имени экземпляра устройства 122
4.3.7.24 Установка имени экземпляра устройства . . . 123
4.3.7.25 Установка нового пароля для смены конфигурации . . . 123
4.3.8 Функции для поиска модулей в локальной сети . . . 124
4.3.8.1 Начало сеанса поиска модулей в локальной сети . . . 124
4.3.8.2 Получение информации о изменении присутствия модулей в локальной сети . . . 125

4.3.8.3 Останов сеанса поиска модулей в локальной сети 126
4.3.8.4 Освобождение описателя сетевого сервиса 126
4.3.8.5 Получить имя экземпляра по описателю сервиса 126
4.3.8.6 Получить серийный номер модуля по описателю сетевого сервиса 127
4.3.8.7 Получить имя усройства модуля по описателю сетевого сервиса 127
4.3.8.8 Получить IP адрес сетевого сервиса 127
4.3.8.9 Проверка, указывают ли оба описателя на один экземпляр сервиса 128
4.3.9 Функции для работы с сигнальным процессором 128
4.3.9.1 Загрузка прошивки сигнального процессора BlackFin. . . 128
4.3.9.2 Проверка, загружена ли прошивка BlackFin. 129
4.3.9.3 Чтение блока данных из памяти сигнального процессора 129
4.3.9.4 Запись блока данных в память сигнального процессора . 130
4.3.9.5 Передача управляющей команды сигнальному процессору 131
4.3.10 Функции для работы с Flash-памятью модуля 132
4.3.10.1 Чтение блока данных из Flash-памяти 132
4.3.10.2 Запись блока данных во Flash-память модуля 132
4.3.10.3 Стирание блока во Flash-памяти 133
4.3.10.4 Разрешение записи в пользовательскую область Flash-памяти 133
4.3.10.5 Запрет записи в пользовательскую область Flash-памяти 133
4.3.11 Дополнительные вспомогательные функции 134
4.3.11.1 Получить версию драйвера модуля L-502. 134
4.3.11.2 Перевод модуля E-502 в режим загрузчика 134
4.3.11.3 Перезагрузка прошивки ПЛИС 135
4.3.11.4 Передача управляющей команды контроллеру Cortex-M4. 135
4.3.11.5 Получить версию библиотеки 136
4.3.11.6 Получение строки об ошибке 136
4.3.11.7 Моргание светодиодом 136
4.3.11.8 Установка подтягивающих резисторов на входных линиях 137
4.3.11.9 Проверка поддержки модулем заданной возможности . . 137

5 Метки времени (только для E502-P1) 138
5.1 Режим синхронизации времени по протоколу PTP 138
5.1.1 Работа с модулем при использовании меток времени 139
5.1.2 Описание регистров PTP 139
5.1.3 PTP-совместимое оборудование 140
5.2 Константы и перечисления 140
5.2.1 Константы и макроопределения 140
5.2.2 Структура для хранения контекста при обработке потока слов “на ввод” с включенными метками времени 141
5.2.3 Тип данных для хранения времени Время хранится в секундах прошедшее с начала этой эпохи (00:00:00 UTC, 1 Января 1970 года) Дробный формат хранения 32.31: 32 целых бит, 31 дробных бит, старшие 32 бита - секунды, младшие 31 бит сабсекунды = 1 / (1<<31) секундсабсекунды, ... 142
5.3 Функции для работы с метками времени 142

5.3.1 Инициализация tstp_state, в нем хранится текущий контекст для операций с метками времени из потока “на ввод”. 142
5.3.2 Обработать очередное слово wrd из потока “на ввод”. 143
5.3.3 Узнать время текущего обработанного слова 143
5.3.4 Возвращает признак того что часы синхронизированы 143

Глава 1

О чем этот документ

Данный документ предназначен в первую очередь для программистов, которые собираются писать свои программы для работы с модулями L-502, E-502 и E16 с использованием предоставляемой фирмой "Л Кард" библиотеки.

В данном документе рассматривается вопрос подключения библиотеки к проекту пользователя, дается подробное описание интерфейсных функций, предоставляемых библиотекой, а также дается описание основных подходов к использованию этих интерфейсных функций.

Сама библиотека написана на языке *C* и все объявления функций и типов, а также примеры в данном документе приводятся на языке *C*, однако все привязки к другим языкам программирования являются лишь обертками над библиотекой *C* и все функции, типы и параметры сохраняют свое значения и для других языков программирования. Поэтому этот документ необходимо прочитать и пользователям, пишущим на других языках программирования. Кроме того в документе дается описание отличий и общих принципов использования библиотек на поддерживаемых других языках программирования. Примеры для других языков можно установить вместе ''L-Card L502/E502 SDK'' .

В настоящем документе не рассматриваются какие-либо вопросы, касающиеся подключения сигналов и характеристик модуля, а также лишь в общем затрагиваются принципы работы самого модуля. Эти вопросы рассматриваются в ''Руководстве пользователя L-502'' и ''Руководстве пользователя E-502'', с которыми рекомендуется ознакомится перед прочтением данного документа.

Также в данном документе не рассматривается задача написания своей прошивки для сигнального процессора модуля и работа с модулем без использования предоставляемой "Л Кард" библиотеки. Эти вопросы рассматриваются в ''Низкоуровневом описании программиста''.

Глава 2

Установка и подключение библиотеки к проекту

Для написания собственного программного обеспечения, работающего с модулями L-502 и E-502, необходимо выполнить следующее:

1. Установить драйвер для модулей:
   - Для работы с модулями, подключенными по интерфейсу PCI Express, необходим специальный драйвер, предоставляемый фирмой “Л Кард”
   - Для работы с модулями, подключенными по интерфейсу USB, используется библиотека `libusb-1.0`. Под Windows сама библиотека включена в стандартную библиотеку **e502api.dll**, а в качестве драйвера используется стандртный драйвер `WinUSB`. Под ОС Linux необходимо установить библиотеку `libusb-1.0`, при этом никаких специальных драйверов не требуется.
   - Для работы с модулями, подключенными по интерфейсу Ethernet (TCP/IP), специальных драйверов не требуется. Для возможности выполнять поиск модулей в локальной сети должна быть установлена соответствующая служба (см. главу Обнаружение модулей в локальной сети).
2. Установить необходимые динамические библиотеки (.dll для Windows или .so для Linux) в директорию, присутствующую в соответствующей переменной окружения, либо в директорию с проектом. Динамическая библиотека необходима при написании программ на любом языке программирования, так как все привязки к языкам работают через указанные библиотеки.. Всего предоставляется три библиотеки:
   - **x502api** — содержит общие функции для обоих модулей. Должна включаться в любой проект, работающий с одним из модулей, за исплючением проектов, написанных только для L-502 до появления библиотеки x502api, которые могут использовать только l502api
   - **l502api** — содержит специфические функции для модуля L-502, а также функции, оставленные для совместимости с проектами, написанными до появления x502api.
   - **e502api** — содержитспецифические функции для модуля E-502.
3. Подключить библиотеку к проекту.

Для ОС Windows предоставляется общий установщик "L-Card L502/E502 SDK" , который автоматически устанавливает все необходимые драйвера, динамические библиотеки в системную директорию, а также все файлы, необходимые для подключения библиотеки к проекту и примеры в указанную директорию. В дальнейшем в данной главе с помощью SDK_DIR будет обозначаться указанная при установке "L-Card L502/E502 SDK" директория.

Установка для ОС Linux описана в разделе Установка библиотеки и драйвера для ОС Linux .

Подключение к проекту зависит от используемого языка и среды программирования.

2.1 Подключение библиотеки при написании программы на языках C/C++

При написании на C/C++ при подключении необходимо выполнить следующее:

1. Включить в проект заголовочный файл "l502api.h" и/или "e502api.h", добавив при этом в проекте к путям для заголовочных файлов директорию SDK_DIR/include.
2. Добавить в проект файл линкера нужных библиотек для используемого компилятора:
   - **Microsoft Visual C++** : SDK_DIR/lib/msvc
   - **Microsoft Visual C++** 64-битный компилятор (подробнее о 64-битной версии описано ниже): SDK_DIR/lib/msvc64
   - **Borland C++/Borland C++ Builder** : SDK_DIR/lib/borland
   - **Borland C++/Borland C++ Builder** 64-битный компилятор: SDK_DIR/lib/borland64
   - **MinGW** : SDK_DIR/lib/mingw
   - **MinGW** 64-битный компилятор: SDK_DIR/lib/mingw64

   Примеры программ на *C* находятся в **SDK_DIR/examples/c**. Специальные примеры для **Borland C++ Builder** — в **SDK_DIR/examples/CppBuilder**.

2.2 Использование библиотеки в проекте на Delphi

Для написания программ на *Delphi* с использованием библиотеки для работы с модулями L-502 и E-502, необходимо включить в проект программы файл **SDK_DIR/pas/x502api.pas**, а также файл **SDK_DIR/pas/l502api.pas** и/или **SDK_DIR/pas/e502api.pas**, которые представляют собой обертку над библиотеками на *C*. В файлах проекта, которые используют типы и функции из этого документа, необходимо подключить модуль x502api, а также l502api и/или e502api с помощью `uses x502api;`, `uses l502api;` и `uses e502api;` соответственно. При этом для 64-битного компилятора используются те же файлы, что и для 32-битного (см. 64-битная версия библиотеки).

Следует отметить, что по сравнению с версиями до поддержки E-502, файл SDK_DIR/pas/lpcieapi.pas больше не используется. Т.к. этот файл не должен был использоваться в проектах напрямую, то он не включен в новых версиях библиотеки.

Все функции, типы и константы библиотеки отображаются в *Delphi* один к одному, за исключением следующих моментов:

- все строки (серийные номера, строки с описанием кодов ошибок) преобразуются оберткой в тип `string`, который используется стандартно для представления строк в *Delphi* (следует не забывать, что в последних версиях среды этот тип представляет собой юникодную строку). Исключением является структура t_x502_info с информацией о модуле, в которой строки представлены массивом `AnsiChar` фиксированной длины.

- все функции, работающие с массивами, принимают в качестве параметра открытый массив (open array parameter), что означает, что в эти функции можно передать как статический массив, так и динамический (установив предварительно его длину с помощью `SetLength()`). При этом, так как массивы Delphi содержат в себе длину, то в функции L502_GetSerialList() и E502_UsbGetSerialList(), а также L502_GetDevRecordsList() и E502_UsbGetDevRecordsList() отдельно передавать размер массива не требуется. Однако в функциях для работы с данными (например, X502_Recv()) длина передается так же, как и в функциях *C*, чтобы можно было использовать для приема не обязательно весь массив. При этом дополнительно проверяется, что переданная отдельным параметром длина не превышает реальную длину массива. В случае превышения будет возвращена ошибка X502_ERR_INSUFFICIENT_ARRAY_SIZE.

  Пример программы на *Delphi* находится в **SDK_DIR/examples/Delphi**.

2.3 Использование библиотеки в проекте на C#

Для написания программ, работающих с модулями L-502 и E-502, на языке *C#* (или на любом другом, поддерживаемым NetFramework), реализована специальная библиотека-обертка lpcieNet.dll. Она использует описанные ранее библиотеки на *C*, в которых реализована вся логика работы с устройством. Установщик позволяет установить lpcieNet.dll в системный кэш (GAC), что позволяет не копировать библиотеку вместе с Вашим проектом. Однако, к сожалению, Visual Studio не позволяет добавлять в проект ссылки из системного кеша и Вам все равно придется саму ссылку делать на локальную копию (просто ее не обязательно будет распространять вместе с проектом). Библиотека в кеше имеет преимущество перед локальной и будет всегда использованна именно она, если установлена.

Для использования библиотеки достаточно добавить в проект ссылку на lpcieNet.dll и в исходниках подключить нужные пространства имен:

```
using x502api;
using lpcieapi;
```

Для совместимости с программами, написанными до включения поддержки L-502, оставлена старая версия всех определений в пространстве имен l502api со старой версией класса L502.

По сравнению с функциями языка *C* в обертке *C#* сделаны следующие изменения:

- Так как *C#* — объектно ориентированный язык, то для управления модулями созданы специальные классы L502 и E502, которые наследуются от общего класса X502.

- Все функции, которые принимают описатель модуля первым параметром реализованы методами классов L502, E502 или X502, при этом сам префикс L502_, E502_ или X502_ не используется. Например, вместо `L502_Open(hnd, serial)`, используется `hnd.Open(serial)`. Для сетевой конфигурации модуля E-502 используется отдельный класс E502.EthConfig.

- Функции X502_Create() и X502_Free() вызывается в конструкторе и деструкторе класса X502, отдельными функциями не реализованы. Аналогично обстоит дело с X502_FreeDevRecordList() и классом X502.DevRec, а также с функциями E502_EthConfigCreate() и E502_EthConfigFree() и классом E502.EthConfig.

- Для создания нужного объекта устройства (L502 или E502) по имени модуля реализован статический метод `X502.Create(devname)`.

- Функции, которые не требуют экземпляра модуля (не принимают описатель первым параметром), реализованы как статические функции соответствующих классов. Например `X502_GetErrorString(err)` раеализована как `X502.GetErrorString(err)`.

- Функции типа Get/Set, которые принимают описатель модуля и один параметр, реализованы в виде свойств (properties). Например, вместо `X502_SetLChannelCount(hnd, value)`, используется `hnd.LChannelCount = value`. Однако следует быть внимательным, так как неправильно устанавливаемое значение вызовет исключение `X502.Exception`.

- Константы объявлены внутри классов и без префикса L502_, E502_ или X502_.

- Перечисления также объявлены как перечисления внутри класса и без префикса `X502_ПЕРЕЧИСЛЕНИЕ`. Например не X502_SYNC_INTERNAL, а `X502.Sync.INTERNAL`.

- Коды ошибок, так как планируется использовать общие коды ошибок и для будущих модулей, вынесены в перечисление ERR в классе lpcie.

- Все функции, использующие в C строки в виде `char *`, используют в обертке строки типа `String`.

- Функции `L502.GetSerialList()`, `E502.UsbGetSerialList()`, а также функции `L502.GetDevRecordsList()` и `E502.UsbGetDevRecordsList()` возвращают созданный внутри функции динамический массив строк/записей о устройстве (а не заполняет переданный), который уже содержит длину. Поэтому дополнительный параметр размера массива не требуется.

- Как и в Delphi, при работе с массивами данных передается длина дополнительным параметром, так как можно принимать меньше данных, чем в выделенном массиве.

- Функции, принимающие указатели в *C*, принимают параметры со спецификаторами out или ref, в зависимости от того, должна ли быть переменная проинциализирована перед вызовом функции или является выходным параметром.

Пример программы на *C#* находится в **SDK_DIR/examples/cs**.

2.4 Использование библиотеки в проекте LabView

Вы можете управлять модулями L-502 и E-502 из *LabView*, используя тот факт, что *LabView* поддерживает управляемые библиотеки NetFramework. Соответственно, Вы имеете доступ ко всем функциям, которые реализует оболочка *C#* lpcieNet.dll, т.е. все доступные функции библиотеки, с учетом отличий, описанных в предыдущем разделе).

В отличие от Visual Studio LabView автоматически подхватывает .Net библиотеки из системного кэша (GAC) и Вы можете ссылаться на нее, а не хранить локальную копию вместе с программой.

Для работы с классами .Net в LabView есть специальная панель **Connectivity -> .Net** .

Вам необходимо использовать следующие блоки:

- **Constructor Node** - создает объект. Должен быть создан для каждого модуля L-502 или E-502, с которым будете работать. При создании LabView предложит выбрать библиотеку и класс (нужно выбрать lpcieNet.dll и L-502 или E-502 из пространства x502api). Одним из выходов этого блока является ссылка на объект, которая используются как вход для остальных блоков для работы с модулем. Также с помощью конструктора создается объект логического канала со всеми настройками. Альтернативным вариантом создания объекта модуля является использование метода X502.Create, который создает нужный объект по имени модуля.

- **Close Reference** - закрывает и удаляет объект. Должен вызываться для каждого созданного объекта по завершению работы.

- **Invoke Node** - вызов функции (метода класса). При работе с объектом на вход подается ссылка и входные параметры, а на выходе — выходные параметры и обновленная ссылка (которая должна использоваться для блоков, которые будут вызваны после текущего). Входная ссылка и определяет, методы какого объекта используются (после заведения ссылки на вход имя класса появится в верхней строке, а при нажатии на вторую будет предложен на выбор его метод). Для функций, которые не работают с конкретным объектом (GetErrorString, GetSerialList и т.д. — эти функции статические и при выборе помечены [S] в начале) ссылку на вход подавать не обязательно. Однако они все равно принадлежат классу, который надо выбрать нажав правой кнопкой на блок и далее **Select Class/.Net**

- **Property Node** - используется для установки или получения свойств. Через свойства устанавливается часть параметров (логическая таблица, режим синхронизации) и также можно получить информацию о модуле (в виде класса, каждое поле которого является также отдельным свойством). Можно устанавливать несколько свойств одним блоком, расширив его вниз. Также с помощью свойств можно задавать константы из перечислений (что может быть более понятно, чем просто подавать на вход числа) - в этом случае надо выбрать класс перечисление, а каждому значению соответствует свое свойство.

Следует отметить особенность передачи массивов в качестве выходных параметров. Сами функции библиотеки для эффективности не выделяют массивов данных внутри себя, а используют переданные массивы для сохранения результатов. Поэтому такие параметры в *LabView* являются входными и выходными одновременно. На вход необходимо подать массив размера, достаточного для сохранения результатов (при этом

сами данные не имеют значения), а в качестве выходного параметра возвращается тот же массив, но уже заполненный результатами выполнения функции. Примером таких параметров могут служить параметр **buf** функции X502_Recv(), параметры **adc_data** и **din_data** функции X502_ProcessData(), а также параметр **out_buf** функции X502_PrepareData().

Примеры программ на *LabView* находится в **SDK_DIR/examples/LabView**.

2.5 Использование библиотеки в Visual Basic 6

Для работы с модулями из программы на *Visual Basic 6* нужно в проект добавить файлы модулей **x502api.bas**, **e502api.bas** и **l502api.bas**, которые можно взять из примера **SDK_DIR/examples/vb6/x502_general**. Файлы содержат объявления всех типов, констант и функций. Параметры функций совпадают с функциями языка *C*, за исключением следующих особенностей:

- так как в Visual Basic не используются указатели, то для всех описателей используется просто тип `Long`
- Функции L502_GetSerialList(), E502_UsbGetSerialList(), а также функции L502_GetDevRecordsList() и E502_UsbGetDevRecordsList() принимают в качестве параметра динамический массив и изменяют его размер в соответствии с количеством найденных элементов. Соответственно, дополнительные явные параметры, задающие размер массива на вход и количество найденных элементов не требуются.
- Строки в функциях преобразовываются в строки Visual Basic автоматически. Строки же в структурах представлены массивом байт. Для перевода их в String можно воспользоваться функцией `X502_StrConvert()`
- Для освобождения ресурсов одной записи дополнительно введена функция `X502_FreeDevRecord()`, вызывающая X502_FreeDevRecordList() для одного элемента, чтобы не преобразовывать элемент в массив вручную

Следует отметить, что при отладке из среды *Visaul Basic 6* при останове незавершенной программы, если на момент останова были незакрытые соединения с модулями, то эти соединения могут не закрываться автоматически до перезапуска среды. Соответственно, т.к. количество одновременных соединений ограничено, то модуль может быть не видим в списке найденных устройств или с ним нельзя будет установить соединение до перезапуска среды.

2.6 64-битная версия библиотеки

На 64-биной версии Windows могут выполняться программы, как собранные 32-битным, так и 64-битным компилятором, поэтому большинство программ для Windows существует только в 32-битном варианте. 64-битный компилятор используют как правило для программ, работающих с большим количеством данных, так как это позволяет иметь процессу виртуальное пространство больше 4 Гбайт.

Для 64-битной Windows установщик "`L-Card L502/E502 SDK`" ставит в соответствующие системные директории 32-битную версию библиотек, так и 64-битную. При

этом директория **Windows/system32** указывает на одну из этих директорий в зависимости от разрядности самого приложения, которое обращается по указанному пути. Для 32-битного приложения в **Windows/system32** находятся 32-битные библиотеки, а 64-битные в**Windows/Sysnative**, а для 64-битного в **Windows/system32** находятся 64-битные, а **Windows/Sysnative** не существует. При этом в **Windows/SysWOW64** всегда лежат 32-битные библиотеки и она всегда существует независимо от разрядности приложения.

При загрузке приложения, если используются системные библиотеки, то они ищутся по путям из переменной окружения PATH, среди которых есть **Windows/system32**. Так как эта директория ссылается на разные места в зависимости от разрядности запускаемого приложения, то выбор библиотеки нужной разрядности из **Windows/system32** происходит автоматически. Если библиотеки распространяются вместе с программой, то нужно следить, чтобы разрядность собранного приложения и библиотек в одной директории совпадала.

Единственным отличием при написании программ на *C или C++* является необходимость подключить lib-файл в соответствии с разрядностью используемого компилятора.

Для программ на *Delphi* необходимо только указать, для какой платформы будет собираться проект (win32 или win64) и собранная программа будет использовать библиотеку той разрядности, для которой программа была скомпилирована.

Программы на языке *C#* (или на любом другом, использующем NetFramework) компилируются в машинный код при выполнении. При этом один раз созданная программа может выполняться как на 32-битной версии, так и на 64-битной версии виртуальной машины NetFramework (в проекте можно указать явно, для какой разрядности NetFramework предназначена программа). Таким образом, одна и та же программа в 32-битной версии Windows будет выполняться с использованием 32-битной версии библиотек, а в 64-битной — с использованием 64-битной. Для библиотеки .Net разрядность определяется разрядностью использующего его приложения.

Соответственно, в проекте на *LabView*, который использует .Net библиотеку, разрядность используемой библиотеки определяется разрядностью используемой среды *LabView*.

2.7 Установка библиотеки и драйвера для ОС Linux

Для установки драйвера и библиотеки под ОС Linux существует два варианта:

- Воспользоваться готовыми собранными пакетами, предоставляемыми “Л Кард”. Это рекомендованный способ для дистрибутивов, для которых предоставляются собранные пакеты. Список поддерживаемых дистрибутивов можно посмотреть в документе “Использование внешних репозиториев 'Л Кард' для дистрибутивов Linux”
- Скачать исходные коды “L-Card L502/E502 SDK” и собрать их самостоятельно (подробнее в следующем разделе).

Для примеров работы с модулями L-502 и E-502 на *C* под Linux можно скачать архив с исходниками SDK и посмотреть на примеры в **api/x502api/examples/msvc**. Несмотря на название, они могут быть собраны GCC под ОС Linux. Для каждого примера есть makefile (с комментариями), а также файл CMakeList.txt для предпочитающих сборку с использованием cmake .

О том, как подключить внешний репозиторий, установить собранные пакеты и о преимуществах данного метода установки описано в документе “Использование внешних репозиториев 'Л Кард' для дистрибутивов Linux”. Здесь же будет приведен список самих пакетов, использующихся при работе с модулями L-502 и E-502, с указанием зависимостей. При подключении внешнего репозитория зависимости разрешаются автоматически (за исключением пакета **lpcie-dkms** , о чем описано ниже). При установке пакетов вручную без подключения внешнего репозитория, следует учитывать зависимости при установке пакетов (например библиотеки следует ставить в следующем порядке: сперва **libx502api1**, затем **libl502api1** и **libe502api1**, и только затем при необходимости **libx502api1-dev** или **libx502api1-devel**).

Для работы с модулями L-502 и E-502 используются пакеты:

- **libx502api1-dev** или **libx502api1-devel** — Пакет с файлами для разработчика: заголовочные файлы и ссылки на библиотеки нужной версии. Нужен при написании своих программ с использованием описанных в данном документе библиотек (зависит от **libx502api1**, **libl502api1** и **libe502api1**, благодаря чему сами библиотеки тоже ставятся автоматически при установке файлов разработчика)

- **libx502api1**, **libl502api1** и **libe502api1** — Пакеты непосредственно с библиотеками нужной версии. Если вы распространяете свою программу, то Вам достаточно включить в зависимости только пакет, соответствующие используемым библиотекам (без пакета с файлами для разработчика), что при создании rpm и deb пакетов выполняется автоматически. Пакет **libx502api1** содержит библиотеку общих функций, которая используются в **libl502api1** и **libe502api1**, поэтому последние зависят от первой. Пакет **libe502api1** также зависит от пакета с библиотекой `libusb-1.0` для данного дистрибутива, чтобы она устанавливалась автоматически, а также ставит правила udev для предоставления прав доступа к устройству E-502, подключенного по USB.

- **lpcie-dkms** — Пакет с исходниками драйвера (модуля ядра) для работы с модулями по интерфейсу PCI-Express (L-502), использующий систему сборки внешних модулей dkms (подробнее описано ниже).

- **lxfw-update** — Утилита для обновления прошивок ПЛИС модулей L-502 и E-502. Пакет включает в себя последнюю версию прошивок и скрипты l502-fpga-update-all.sh и e502-fpga-update-all.sh для обновления прошивок всех найденных устройств L-502 или E-502 соответственно.

Так как драйвер должен быть собран под конкретную версию ядра (а ядро может обновляться в одной версии дистрибутива или даже могут использоваться разные варианты ядра), то драйвер не может распространятся в уже собранном виде. Его сборка выполняется непосредственно при установке пакета. При этом необходимо предварительно поставить пакет с заголовочными файлами текущего ядра (обычно в пакетах с именами **linux-headers** или **kernel-devel**). Для некоторых дистрибутивов может быть несколько вариантов ядра (и соответственно несколько пакетов), более того Вы можете использовать свое ядро. Именно по этой причине пакет не задан зависимостью для **lpcie-dkms**, в отличие от других зависимостей. Узнать текущую версию ядра можно командой: `uname -r`. Убедиться, что нужные файлы установлены можно проверив наличие файлов в директории **/lib/modules/**‘uname -r‘**/build** (обычно это ссылка на заголовки ядра в **/usr/src/linux-<version>** или **/usr/src/kernels/<version>**).

Если заголовки текущего ядра установлены, то при установке пакета **lpcie-dkms** будет выполнена сборка драйверов с использованием DKMS (пакет dkms входит в зависимости **lpcie-dkms**, как и make и gcc, необходимые для сборки). DKMS это достаточно широко распространенная система сборки и управления внешними модулями ядра (она находится в основном репозитории большинства дистрибутивов Linux, хотя ее нет в OpenSuse, но для нее распространяется пакет dkms через тот же Open Build System).

DKMS позволяет:

- централизованно отслеживать, какие сторонние модули ядра, какие их версии и для каких версий ядра установлены (dkms status)
- хранит всегда исходники драйвера разных версий в централизованном месте (**/usr/src/lpcie-<version>**)
- позволяет автоматически пересобирать драйвер при переходе на новое ядро
- позволяет в любой момент удалить драйвер или любую его версию и все связанные с ним файлы (dkms remove -m lpcie -v <version> –all)

Таким образом, хотя в пакете находятся исходники драйвера, а не собранный драйвер, установка мало чем отличается от установки других пакетов, кроме того, что требует дополнительной установки заголовочных файлов ядра и установка пакета требует значительного времени на сборку.

При установки нового ядра модуль будет пересобран под него автоматически либо при установке пакета, либо при первом входе в систему с новым ядром.

2.8 Исходные коды SDK

Исходные коды всех составных компонентов SDK открыты. Пользователю предоставляется доступ на чтение к репозиторию системы контроля версий `git`, расположенному по адресу `https://gitlab.com/l-card/acq/devices/x502/sdk/lpcie_sdk`.

Также возможно скачать архив со всеми исходными кодами SDK по ссылке `https://lcard.ru/download/lpcie_sdk_src.zip`.

Инструкции по сборке находятся в файле исходников INSTALL.txt.

Глава 3

Общий подход к работе с библиотекой

3.1 Отличия при работе с модулям L-502, E-502 и E16

3.1.1 Отличие возможностей модулей

Функциональные возможности модулей L-502, E-502 и E16 очень схожи, однако есть существенные отличая.

1. Основное отличие заключается в использованных интерфейсах — PCI-Express для L-502 и USB или Ethernet для E-502 и E16. В связи с этим несколько отличается процедура установки связи с устройством. Кроме того, работа по интерфейсу Ethernet требует настройки дополнительных параметров.

2. В E-502 используется дополнительно контроллер ARM Cortex-M4 для реализации логики интерфейсов (USB/Ethernet). У данного контроллера есть своя прошивка, которую можно обновлять и в которой могут быть реализованы дополнительные возможности (в первую очередь при работе через Ethernet). Рекомендуется всегда использовать последнюю версию прошивки, которую можно скачать по адресу https://lcard.ru/e502-m4.bin и обновить с помощью программы X502Studio. Кроме того, теоретически возможно создание пользователем своей прошивки для данного контроллера, так как “Л Кард” предоставляет исходные коды данной прошивки (git-репозиторий с исходниками прошивки можно найти по адресу https://gitlab.com/l-card/acq/devices/x502/e502/firmware/e502_m4), однако какого-либо руководства по данной прошивке “Л Кард” для этого не предоставляет. Также возможно рассмотрение предложений по заказу на доработку прошивки ARM Cortex-M4 под нужные пользователю задачи.

3. Существует ограничение скорости передачи данных для E-502. Если L-502 позволяет одновременно осуществить ввод и вывод всех данных с аналоговых каналов и цифровых линий на максимальной скорости, то в модуле E-502 есть ограничения, которые зависят от используемого интерфейса:

   - при работе по USB суммарная максимальная скорость передачи составляет порядка 5 млн. отсчетов в секунду. Т.е. допустимо использовать например ввод с АЦП и цифровых линий на 2 МГц и при этом вывод только на один канал ЦАП/DOUT на 1 МГц, или на два на 500 КГц. Если идет ввод только с АЦП на 2 МГц, то возможно использовать все три канала вывода на 1 МГц и

т.д. При этом это ограничение обусловлено ограничением скорости интерфейса между контроллером ARM Cortex-M4 и ПЛИС, а не самим интерфейсом USB.

- при работе по Ethernet (TCP/IP) максимальная скорость уже ограничена скоростью передачи по сети по протоколу TCP. При работе модуля только на ввод скорость ограничена 2.5 млн. отсчетов в секунду. Скорость на вывод и влияние ее на скорость ввода будет уточнена в дальнейшем. Для вывода рекомендуется по возможности использовать циклический режим вывода. Следует также учитывать загрузку самой сети при передаче данных по Ethernet, так как она может сильно влиять на максимальную скорость передачи.

4. В связи с ограничением скорости передачи в E-502 поддерживается возможность задать общий делитель для частоты вывода , который можно задать с помощью X502_SetOutFreqDivider() или X502_SetOutFreq(). В L-502 также была введена данная возможность в версии прошивки ПЛИС 0.5, таким образом для ее использования может быть необходимо обновить прошивку.

5. При циклическом выводе в L-502 буфер находится в драйвере на ПК, а в E-502 хранение циклического буфера реализовано внутри памяти контроллера ARM Cortex-M4 модуля, что позволяет избежать загрузки интерфейса и ПК во время работы, однако приводит к ограничению размера буфера. Т.е. при работе по сети циклический вывод не влияет на ограничение интерфейса, однако ограничение в суммарную скорость в 5 млн. отсчетов в секунду для модуля сохраняется. Для версии прошивки ARM ниже 1.0.3 буфер вывода в 3 млн. отсчетов (суммарно на все каналы) делился на 2 равные части (одна используется для вывода сигнала, другая для загрузки следующего, чтобы можно было сменить сигнал без останова предыдущего), поэтому один сигнал ограничен в 1.5 млн. отсчетов. Начиная с 1.0.3 буфер может делится произвольно (в зависимости от размера загружаемого сигнала), поэтому размер сигнала ограничен значением — 3 млн. отсчетов минус размер выдаваемого сейчас сигнала (0 — если при загрузке не идет генерация предыдущего). Таким образом, если не используется возможность смены сигнала на лету (т.е. всегда после X502_OutCycleSetup() идет X502_OutCycleStop() или X502_StreamsStop() до следующей загрузки сигнала), то можно для одного сигнала использовать все 3 млн. отсчетов. Также следует учитывать, что передача сигнала в ARM контроллер занимает время и при необходимости выдачи сигнала одновременно с запуском сбора необходимо дождаться загрузки сигнала, что можно сделать например с помощью флага X502_OUT_CYCLE_FLAGS_WAIT_DONE в X502_OutCycleSetup().

6. При работе по интерфейсу Ethernet не реализована функция X502_GetSendReadyCount(), а функция X502_GetRecvReadyCount() гарантировано работает только под ОС Windows.

7. При чтении значений цифровых входов для E-502 старшие линии DI14, DI15, DI16 объединены с линиями синхронизации DI_SYN2, CONV_IN и START_IN соответственно. При этом, так как в обоих модулях линии 17 и 18 объединены с DI_SYN1 и DI_SYN2, то значение 18-ой и 14-ой линии для E-502 всегда одинаковы.

8. Настройки подтяжек для цифровых входов в E-502 отличаются от L-502 (см. описание типа t_x502_pullups)

9. В E-502 используется другая микросхема ЦАП, которая также планируется к использованию в последующих ревизиях модуля L-502.

10. E16 по цоколевке сигнальных разъемов и диапазонам измерений входного сигнала это продолжение линейки модулей E14. API верхнего уровня сделано совместимым с e502api. Для задания частоты работы ЦАП и АЦП используется опорная частота 48 Мгц. Есть режим совместимости с модулем E14-440 и работы с E14-440 API. В качестве интерфейсного процессора используется контроллер WCH32V307. Сигнальные разъемы по цоколевке несовместимы с E502!

11. В E16 как и в E14 реализован старт сбора по уровню аналогового сигнала - выше/ниже уровня, при пересечении границы уровня снизу-вверх/сверху-вниз.

12. Идентификатор USB устройства для E16 (VID_2A52&PID_0E16) отличается от E502 и E14, поэтому в некоторых случаях придется установить новые драйвера.

13. В E16 для циклического вывода (вывода из внутреннего буфера без внешней подкачки), доступен один буфер размером 11264 слов (в новых прошивках кол-во может измениться).

14. В отличие от E502, в E16 нет выделенных выходов синхронизации START_OUT и CONV_OUT, но их роль могут исполнять выходы TRIG или INT, которые можно переключать на вход или на выход, при конфигурации “на вход” эти контакты можно использовать как START_IN или CONV_IN.

15. Контакты 17 и 18 цифрового разъёма сделаны унифицированными, по умолчанию на них выведены +3,3В и GND как в E14-440, при подаче команды включения реле (см. t_x502_pullups) на эти выводы выводится +15 и -15В как в E14-140.

3.1.2 Общие и специализированные функции для работы с модулем

В связи с тем, что большая часть функциональности модулей L-502, E-502 и E16 совпадают, то большинство функций реализовано общими для обоих модулей. Все общие функции реализованы в библиотеке **x502api**. При этом названия функций и констант начинаются с X502_, а типов с t_x502_. Соответственно, для работы с обоими модулями используется один и тот же тип описателя модуля t_x502_hnd.

Основное отличие при работе, зависящее от интерфейса связи с модулем, заключается в процедуре установки самой связи. Эти функции реализованы в отдельных библиотеках **l502api** и **e502api** для модуля L-502, E-502 и E16 соответственно. При установки связи, вся необходимая информация о том, как работать с модулем по нужному интерфейсу, сохраняется внутри непрозрачного описателя модуля, а пользователь после открытия связи может работать с модулем одинаково с использованием одних и тех же общих функций из **x502api** независимо от типа модуля и используемого интерфейса.

Также в отдельную группу специализированных функций выделены функции по настройке интерфейса Ethernet, которые реализованы только в **e502api** и описаны в разделе Функции для настройки сетевых параметров модуля E502.

3.1.3 Совместимость проектов, разработанных до введения библиотеки x502api

Для реализации полной совместимости с проектами, работающими только с модулем L-502 и разработанными до введения поддержки работы с модулем E-502 (которая введена с версии 1.1.0) и создания общей библиотеки, в **l502api** реализованы функции из данной библиотеки предыдущих версий (1.0.x). При этом эти объявления этих функций и соответсвующих типов вынесены в отдельный файл “l502api_compat.h”, который включается из “l502api.h” для сохранения совместимости. Эти типы определены через общие типы из “x502api.h”, а функции реально только вызывают аналогичные общие функции из **x502api**. Соответственно, проекты, разработанные для предыдущей версии без учета общих функций, должны корректно собираться и при новой версии библиотек. Главное отличие, которое следует учитывать, что если эти проекты распространяются с новой версией **l502api**, то необходимо распространять и библиотеку **x502api**, так как ее функции используются функциями новой версии **l502api**.

Так как данные функции и типы полностью повторяют большинство обобщенных функций (за исключением префиксов в названиях), то они не приводятся в данном документе.

3.2 Общий алгоритм для работы с модулем

Данный раздел описывает типичную последовательность вызова функций для работы с модулями L-502, E-502 и E16. Более подробно каждый шаг будет описан в последующих разделах.

Типичная последовательность вызовов имеет следующий вид:

1. При работе с модулями по интерфейсам PCI-Express или USB получить список серийных номеров с помощью функций L502_GetSerialList() и E502_UsbGetSerialList(), соответственно, или получить список записей о модулях с помощью L502_GetDevRecordsList() и E502_UsbGetDevRecordsList(). При работе по Ethernet можно использовать функции обнаружения устройств в локальной сети, описанные в главе Обнаружение модулей в локальной сети , или, если известен IP-адрес устройства, перейти сразу к пункту 2.

2. Если в системе присутствует нужный модуль, создать описатель модуля с помощью X502_Create().

3. Установить соединение с модулем. При использовании записей о устройстве открытие всегда выполняется с помощью X502_OpenByDevRecord(). При использовании серийного номера используются L502_Open() или E502_OpenUsb() для L-502 и E-502, подключенного по USB, соответственно. Чтобы установить связь с модулем по Ethernet с использованием IP-адреса, необходимо использовать функцию E502_OpenByIpAddr().

4. При необходимости, получить дополнительную информацию о устройстве с помощью X502_GetDevInfo() (в частности для проверки наличия сигнально процессора BlackFin).

5. При наличии сигнального процессора (и желании работать с его использованием) загрузить прошивку сигнального процессора с помощью X502_BfLoadFirmware().

6. Установка параметров модуля с помощью набора функций для изменения настроек модуля (названия функций начинаются с X502_Set)
7. Передача установленных параметров в модуль с помощью X502_Configure().
8. Работа с модулем в синхронном и/или асинхронном режиме (описана в последующих подразделах).
9. Закрытие модуля X502_Close().
10. Освобождение описателя модуля функцией X502_Free().

3.2.1 Работа с модулем при синхронном вводе

Типичная работа с модулем при синхронном вводе состоит из следующих шагов:

1. Разрешение нужных синхронных потоков (АЦП и/или цифровых данных) с помощью X502_StreamsEnable().
2. Запуск синхронных потоков X502_StreamsStart().
3. Чтение принятых данных из модуля с помощью X502_Recv().
4. Обработка прочитанных данных с помощью X502_ProcessData(), X502_ProcessAdcData() или X502_ProcessDataWithUserExt().
5. При необходимости приема и обработки следующего блока, переход к пункту 3.
6. Останов синхронных потоков с помощью X502_StreamsStop().

3.2.2 Работа с модулем при синхронном потоковом выводе

Типичная работа с модулем при синхронном выводе состоит из следующих шагов:

1. Установка начальных значений для ЦАП с помощью асинхронного вывода X502_AsyncOutDac().
2. Разрешение нужных синхронных потоков (каналы ЦАП, цифровые выходы) с помощью X502_StreamsEnable().
3. Запуск предварительной загрузки данных на вывод с помощью X502_PreloadStart().
4. Подготовка блока данных на запись с помощью X502_PrepareData().
5. Запись подготовленного блока в модуль с помощью X502_Send().
6. При необходимости повторить пункты 4. и 5. нужное количество раз. При этом общий размер предварительно загруженных данных не должен превысить размер буфера (по умолчанию 9 МСлов)
7. Запуск синхронных потоков вызовом X502_StreamsStart().
8. Каждый раз при необходимости подгрузить новые данные в буфер выполнить пункты 4. и 5.
9. По завершению работы выполнить останов синхронных потоков с помощью X502_StreamsStop().

3.2.3 Работа с модулем при циклическом выводе

Для выставления циклического сигнала без подкачки типичная последовательность выглядит так:

1. Установка начальных значений для ЦАП с помощью асинхронного вывода X502_AsyncOutDac().
2. Разрешение нужных синхронных потоков (каналы ЦАП, цифровые выходы) с помощью X502_StreamsEnable().
3. Выделение циклического буфера указанного размера с помощью X502_OutCycleLoadStart().
4. Загрузка данных указанного размера для циклического вывода с помощью одного или нескольких вызовов X502_Send().
5. Сделать загруженный сигнал активным с помощью X502_OutCycleSetup() с флагом X502_OUT_CYCLE_FLAGS_WAIT_DONE.
6. Запустить синхронный ввод-вывод через X502_StreamsStart().
7. При необходимости вывести новый сигнал выполнить шаги 3.-5.
8. По завершению работы остановить синхронный ввод-вывод с помощью X502_StreamsStop() или только циклический вывод через X502_OutCycleStop().

3.2.4 Работа с модулем при асинхронном вводе-выводе

Типичная работа при асинхронном вводе-выводе состоит из вызова одной из функций:

- X502_AsyncInDig() - асинхронный ввод значений цифровых линий
- X502_AsyncOutDig() - асинхронный вывод значений на цифровые линии
- X502_AsyncOutDac() - асинхронный вывод значения на один из каналов ЦАП
- X502_AsyncGetAdcFrame() - асинхронный прием одного кадра АЦП

3.3 Создание и освобождение описателя модуля

Вся работа с модулями L-502, E-502 и E16 осуществляется через описатель модуля типа t_x502_hnd. Описатель модуля представляет собой непрозрачный указатель на структуру, которая хранит всю информацию о модуле и состоянии соединения с ним. Пользователь не имеет прямого доступа к полям структуры и все действия с модулем выполняются посредством вызова соответствующих функций библиотеки, которые принимают описатель модуля в качестве первого параметра.

Перед попыткой установить связь с модулем необходимо создать описатель, вызвав функцию X502_Create(), которая выделяет память под структуру, инициализирует ее поля значениями по умолчанию и возвращает указатель на нее — описатель модуля.

После того как работа с модулем завершена, выделенная функцией X502_Create() память должна быть освобождена посредством вызова X502_Free(). После освобождения описатель уже не может использоваться.

3.4 Открытие связи с модулем

3.4.1 Общее описание процедуры установки связи с модулем

Для работы с модулем необходимо установить с ним соединение с использованием ранее созданного описателя.

Функции для установки соединения с модулем зависят от используемого интерфейса и описаны в последующих подразделах.

Если функция открытия связи с модулем выполнена успешно, то соединение с модулем установлено и дальше можно использовать любые функции управления модулем. По завершению работы с модулем необходимо закрыть соединение с помощью X502_Close().

Если же функция открытия связи вернула ошибку, то это может соответствовать одной из следующих ситуаций:

- Связь с модулем не удалось установить, т.к. модуль не найден или не отвечает корректно на запросы. В этом случае по завершению вызова функции описатель модуля остается в исходном состоянии, не связанном с каким-либо соединением и его можно использовать только для открытия нового соединения.

- Связь с модулем установлена, но обнаружены проблемы в работе модуля. В этом случае соединение с модулем все равно остается открытым, однако большая часть функций работы с модулем не доступна.

При необходимости различить эти два разных состояния можно с помощью функции X502_IsOpened(), которая проверяет, открыто ли соединение с модулем в данный момент для указанного описателя. В большинстве случаев явно различать эти ситуации не требуется и достаточно вызвать функцию X502_Close(), которая закроет соединение, если оно не закрыто, при любой ошибке открытия соединения. Если функцию закрытия не вызывать по ошибке открытия соединения, то соединение может остаться открытым до освобождения памяти описателя с помощью X502_Free() или до повторного открытия соединения с помощью данного описателя.

Соединение может оставаться открытым при следующих ошибках:

- X502_ERR_FPGA_NOT_LOADED — прошивка ПЛИС не была успешно загружена и ПЛИС находится в нерабочем состоянии. Ошибка актуальна только для модуля E-502, т.к. в модуле L-502 в ПЛИС реализован сам интерфейс PCI-Express и без загрузки прошивки ПЛИС модуль не будет даже определен в системе. В модуле E-502 прошивка ПЛИС загружается микроконтроллером из Flash-памяти при старте модуля. Ошибка может возникать, если повреждена информация во Flash-памяти или по каким-то причинам возникла ошибка при ее чтении из Flash-памяти или при ее загрузке в ПЛИС. В данном режиме доступны только функции работы с Flash-памятью, функции управления сетевыми настройками модуля и функция перезагрузки прошивки ПЛИС (E502_ReloadFPGA()).

- X502_ERR_REF_FREQ_NOT_LOCKED — ошибка захвата опорной частоты синхронизации, идущей от гальваноотвязанной части управления вводом-выводом. Хотя доступ к Flash-памяти и регистрам ПЛИС может работать нормально, любые функции ввода-вывода аналоговых или цифровых данных не могут быть осуществлены при данной ошибке.

3.4.2 Установка связи с модулем L-502 по интерфейсу PCI-Express

Для начала работы с модулем необходимо установить с ним связь с помощью функции L502_Open(). Для различия модулей используется их серийные номера.

Получить список серийных номеров всех найденных модулей L-502 можно с помощью функции L502_GetSerialList(). Данная функция принимает плоский массив, в который будут сохранены найденные серийные номера, и максимальное количество серийных номеров, которое можно сохранить в переданный массив.

В простейшем случае можно задать максимальное значение модулей и для серийных номеров использовать статически выделенный массив:

```
#define MAX_MODULES_CNT  16

char serial_list[MAX_MODULES_CNT][X502_SERIAL_SIZE];
int32_t get_list_res = L502_GetSerialList(serial_list, MAX_MODULES_CNT, 0, NULL);
if (get_list_res<0) {
    /* Ошибка получения списка серийных номеров */
} else if (get_list_res==0) {
     /* Не найдено ни одного модуля */
}  else  {
     /* Найдено get_list_res модулей */
}
```

В общем случае для работы с произвольным максимальным количеством модулей можно воспользоваться третьим параметром функции для получения количества найденных модулей в системе. При этом в качестве массива серийных номеров можно передать нулевой указатель и указать нулевой размер массива. После этого можно динамически выделить массив под полученное количество серийных номеров и повторно вызвать L502_GetSerialList() для получения серийных номеров всех модулей L-502:

```
uint32_t dev_cnt;
int32_t res;

/* Получаем количество модулей в системе */
res  = L502_GetSerialList(NULL, 0, 0, &dev_cnt);
if (res<0)  {
     /* Ошибка получения списка серийных номеров */
}  else if (dev_cnt==0)  {
     /* Не найдено ни одного модуля */
}  else {
    /* Выделяем плоский массив под dev_cnt серийных номеров размером
       dev_cnt*L502_SERIAL_SIZE */
    t_x502_serial_list serial_list=
           (t_x502_serial_list)
           malloc(dev_cnt*X502_SERIAL_SIZE);
    if (serial_list == NULL)  {
         /* Ошибка выделения памяти */
    }  else  {
         res = L502_GetSerialList(serial_list, dev_cnt, 0,
                                  NULL);
         if (res>0) {
             /* Получено res серийных номеров */
         }
         /* Освобождаем выделенный массив под серийные номера */
         free(serial_list);
    }
}
```

Следует отметить, что с одним модулем одновременно может быть установлено только одно соединение. При попытке открыть модуль, с которым уже установлено соединение через другой описатель (возможно, в другой программе) L502_Open() вернет

X502_ERR_DEVICE_ACCESS_DENIED. При этом L502_GetSerialList() по умолчанию возвращает список всех серийных номеров модуля, включая те, с которыми уже установлено соединение. Если нужно получить список только тех устройств, с которыми еще не установлено соединения, то в L502_GetSerialList() можно передать флаг X502_GETDEVS_FLAGS_ONLY_NOT_OPENED.

Если в качестве серийного номера в L502_Open() передать нулевой указатель или пустую строку, то будет предпринята попытка открыть первый модуль, с которым удастся успешно установить соединение. Если ни с одним модулем установить соединение не удалось, то будет возвращена ошибка, полученная при попытки открыть последний модуль. То есть, при наличии двух модулей L-502 в системе, первый вызов L502_Open() установит соединение с первым модулем L-502, второй вызов - со вторым, а третий вернет уже ошибку доступа X502_ERR_DEVICE_ACCESS_DENIED.

3.4.3 Установка связи с модулем E-502 по интерфейсу USB

При работе по USB алгоритм установления связи абсолютно аналогичен открытию модуля L-502, с той разницей, что для получения списка серийных номеров используется функция E502_UsbGetSerialList(), а для открытия модуля E-502 по серийному номеру используется E502_OpenUsb().

3.4.4 Установка связи с модулем E-502 по интерфейсу Ethernet

Установить связь с модулем по Ethernet можно как по явно заданному IP-адресу устройства, так и воспользоваться функциями обнаружения устройств в локальной сети, подробно описанными в главе Обнаружение модулей в локальной сети .

При установке связи по IP-адресу достаточно вызвать функцию E502_OpenByIpAddr().

Следует отметить, что в отличие от других интерфейсов, для корректной работы по Ethernet должны быть настроены необходимые параметры, о чем описано в главе Особенности работы по интерфейсу Ethernet и настройка сетевых параметров .

3.4.5 Установка связи с модулями с использованием записей о устройстве

Одним из минусов функций открытия по серийному номеру является то, что в момент открытия необходимо знать, устройству, подключенному по какому интерфейсу, соответствует данный номер, чтобы выбрать соответствующую функцию для открытия устройства. Для избежания этого введен специальный тип t_x502_devrec, соответствующий записи о найденном устройстве. Данный тип содержит информацию о найденном устройстве (название, серийный номер, интерфейс, флаги с поддерживающимися возможностями и т.п.), а также содержит всю необходимую информацию о том, как установить связь и работать с соответствующим устройством. Соответственно, от устройства и интерфейса зависят только функции получения записей о устройстве, а установка связи осуществляется общей функцией X502_OpenByDevRecord().

Это позволяет получить все необходимые записи на отдельном этапе и сохранить их в общий массив, а в дальнейшем уже выполнить открытие связи по нужным записям, не разбирая отдельно, что это за устройство и по какому интерфейсу подключено.

Особенностью является необходимость пользователю очищать память, выделенную на этапе инициализации записи об устройстве. Очистка памяти выполняется с помощью

X502_FreeDevRecordList(). Запись об устройстве используется исключительно внутри X502_OpenByDevRecord() и при желании может быть освобождена сразу после вызова данной функции, если список записей больше не нужен. Также следует учесть, что перед тем, как проинициализированную ранее запись передавать в функцию для получения или инициализации новой записи (например для обновления списка), необходимо ее сперва очистить для избежания утечек памяти.

Доступны следующие функции для инициализации записей:

- L502_GetDevRecordsList() инициализирует записи, соответствующие подключенным модулям L-502 по интерфейсу PCI-Express. По параметрам и использованию аналогична L502_GetSerialList() с учетом необходимости освобождать записи.
- E502_UsbGetDevRecordsList() инициализирует записи, соответствующие подключенным модулям E-502 по интерфейсу USB. По параметрам и использованию аналогична E502_UsbGetSerialList() с учетом необходимости освобождать записи.
- E502_MakeDevRecordByIpAddr() инициализирует запись для установления соединения с модулем E-502 с заданным адресом по интерфейсу Ethernet.
- E502_MakeDevRecordByEthSvc() инициализирует запись для установления соединения с модулем E-502, соответствующем автоматически обнаруженному сервису, по интерфейсу Ethernet

Ниже приведен пример, который создает записи для всех найденных модулей, подключенных по интерфейсам USB и PCI-Express. Кроме того, предполагается, что в массиве **ip_addr_list** содержится **ip_cnt** адресов, для которых также создаются записи, а определение **TCP_CONNECTION_TOUT** задает таймаут в мс на подключение по сетевому интерфейсу. Затем предоставляется выбор нужного устройства, после чего с ним устанавливается связь и все записи после этого очищаются.

```
uint32_t ip_addr_list[] = { .... } ; /* список ip-адресов устройств  */
uint32_t ip_cnt = ... ; /* размер этого списка */

uint32_t pci_devcnt = 0;
uint32_t usb_devcnt = 0;
int32_t fnd_devcnt = 0; /* общее кол-во найденных записей */
t_x502_devrec *devrec_list = NULL; /* список записей о устройствах */
t_x502_hnd hnd = NULL; /* описатель открытого устройства */

/* получаем количество подключенных устройств по интерфейсам PCI и USB */
L502_GetDevRecordsList(NULL, 0, 0, &pci_devcnt);
E502_UsbGetDevRecordsList(NULL, 0, 0, &usb_devcnt);

if ((pci_devcnt+usb_devcnt + ip_cnt) != 0) {
    /* выделяем память для массива для сохранения найденного
        количества записей */
    devrec_list = malloc((pci_devcnt + usb_devcnt + ip_cnt) *
                         sizeof(t_x502_devrec));

    if (devrec_list != NULL) {
        unsigned i;
        /* получаем записи о модулях L-502, но не больше pci_devcnt */
        if (pci_devcnt!=0) {
            int32_t res = L502_GetDevRecordsList(&devrec_list[fnd_devcnt],
                                                 pci_devcnt, 0, NULL);
            if (res >= 0) {
                fnd_devcnt += res;
            }
        }
        /* добавляем записи о модулях E-502, подключенных по USB, в конец массива */
        if (usb_devcnt!=0) {
            int32_t res = E502_UsbGetDevRecordsList(&devrec_list[fnd_devcnt],
```

```
                                                usb_devcnt, 0, NULL);
            if (res >= 0) {
                fnd_devcnt += res;
            }
        }

        /* создаем записи для переданного массива ip-адресов */
        for (i=0; i < ip_cnt; i++) {
            if (E502_MakeDevRecordByIpAddr(&devrec_list[fnd_devcnt],
                                           ip_addr_list[i], 0,
                                           TCP_CONNECTION_TOUT) == X502_ERR_OK) {
                fnd_devcnt++;
            }
        }
    }
}

if (fnd_devcnt != 0) {
  uint32_t dev_ind;

  /* обработка списка и выбор нужного устройства, индекс которого
    для примера сохраняется в dev_ind */
   .....


    if ( <устройство выбрано> )
        hnd = X502_Create();
        if (hnd==NULL) {
            /* Ошибка создания описателя модуля! */
        } else {
            /* устанавливаем связь с модулем по записи */
            int32_t err = X502_OpenByDevRecord(hnd, &devrec_list[dev_ind]);
            if (err != X502_ERR_OK) {
                /* ошибка установления соединения */
                X502_Free(hnd);
                hnd = NULL;
            }
        }
    }

    /* освобождение ресурсов действительных записей из списка */
    X502_FreeDevRecordList(devrec_list, fnd_devcnt);
}
/* очистка памяти самого массива */
free(devrec_list);


if (hnd != NULL) {
    /* работа с модулем */

    X502_Close(hnd);
    X502_Free(hnd);
}
```

В данном примере все ресурсы записей освобождаются сразу после выбора нужного устройства и установления с ним связи. При желании можно использовать другой подход, например, сохраняя записи вместе с созданными описателями устройства, чтобы в любой нужный момент можно было открыть, закрыть и снова открыть устройство, освобождая каждую запись только в момент завершения работы или получения нового списка устройств. При этом следует учитывать, что копировать запись о устройстве можно, но X502_FreeDevRecordList должна вызываться только один раз на одну копию каждой записи.

Следует отметить, что все функции получения серийных номеров и установления связи с модулями из предыдущих разделов реально реализованы через описанные в данном разделе функции и являются лишь обертками, созданными для упрощения процедуры поиска и установления связи с устройствами.

3.5 Режимы работы с сигнальным процессором и без него

Модули L-502 и E-502 могут работать в двух режимах:

- В штатном режиме (X502_MODE_FPGA) вся обработка данных выполняется аппаратно в ПЛИС модуля, а управление модулем осуществляется путем прямой записи значений в регистры ПЛИС. В этом режиме доступны все штатные функции сбора данных, однако у пользователя нет возможности расширить функциональные возможности самого модуля. Этот режим доступен для всех модификаций L-502 и E-502.

- В режим работы с сигнальным процессором (X502_MODE_DSP) (отсутствует в E16) всё управление сбором данных выполняется сигнальным процессором BlackFin и все потоки данных на ввод и вывод идут через него. Таким образом, пользователь может путем создания модифицированной прошивки BlackFin реализовать дополнительные возможности (например, обратную связь в режиме реального времени). Этот режим доступен только для модификаций L-502-P-G, L-502-P-G-D и E-502-P-EU-D. Узнать о наличие сигнального процессора программно можно также по флагу X502_DEVFLAGS_BF_PRESENT в флагах в записи о устройстве или в флагах информации о модуле, которая может быть получена после установления связи с модулем через функцию L502_GetDevInfo().

При включении питания модуль всегда находится в штатном режиме работы без использования сигнального процессора. Для работы сигнального процессора необходимо предварительно загрузить в него программу (прошивку). Это можно сделать из файла формата ldr с помощью функции X502_BfLoadFirmware(). После этого модуль будет автоматически переведен в режим с сигнальным процессором.

При необходимости, можно специально переключить режим работы с помощью X502_SetMode(). Это может потребоваться, например, если прошивка загружена в BlackFin по интерфейсу JTAG. Кроме того, следует иметь ввиду, что при открытии связи с устройством не производится изменение режима работы модуля. Т.е. если одна программа установила режим X502_MODE_DSP, то при последующем открытии модуля из другой программы этот режим сохранится. В этой связи может понадобиться явно перевести модуль в штатный режим с помощью X502_SetMode(). Поэтому если программа не предполагает, что модуль мог работать в режиме X502_MODE_DSP и выполнять какие-то функции, которые не нужно прерывать, а работает с модулем с “чистого листа”, рекомендуется сразу после установки связи установить явно нужный режим работы.

Все настройки модуля и работа с синхронным вводом-выводом должны выполняться после установки нужного режима.

В любой момент можно узнать текущий режим работы с помощью X502_GetMode().

3.6 Установка настроек модуля

Перед использованием модуля, как правило, необходимо выполнить настройку его параметров. Сперва все настройки записываются в поля структуры описателя модуля с помощью функций, начинающихся с X502_Set, которые будут описаны в последующих

подразделах, после чего установленные параметры передаются в модуль с помощью X502_Configure().

3.6.1 Настройка последовательности опроса каналов АЦП

Модули L-502, E-502 и E16 представляют собой АЦП с последовательной коммутацией каналов. То есть измерение нескольких каналов происходит последовательно, путем переключения входного коммутатора АЦП. Как и в большинстве моделей "Л Кард" последовательность опроса каналов задается с помощью управляющей таблицы логических каналов АЦП. Всего таблица может содержать от одного до X502_LTABLE_MAX_CH_CNT логических каналов.

Каждый логический канал задает следующие параметры:

- номер физического канала, с которого производится измерение. Номер физического канала задается, считая от 0, то есть 0 означает первый канал, 1 – второй и т.д. Таким образом, в дифференциальном режиме номер канала может быть от 0 до 15, а в режиме измерения с общей землей — от 0 до 31.
- режим измерения АЦП из t_x502_lch_mode.
- используемый диапазон измерения (из t_x502_adc_range или t_e16_adc_range).
- коэффициент усреднения по заданному логическому каналу (см. раздел Коэффициент усреднения для логического канала (не реализовано в E16) , не реализовано в E16).

Задать параметры логического канала с нужным номером можно с помощью функции X502_SetLChannel(), а количество логических каналов в управляющей таблице — с помощью X502_SetLChannelCount().

Например, необходимо измерить сперва напряжение входа X1 относительно общей земли для диапазона +/-10В, затем измерить значение на 16 канале в дифференциальном режиме (между входами X16 и Y16) с диапазоном +/-1В, а затем измерить напряжение между Y1 и общей землей (17 канал в режиме с общей землей) с диапазоном +/-0.2В (Назначение выводов сигнального разъема и подключение сигналов к модулю описано в "Руководстве пользователя"). В этом случае настройка логической таблицы будет выглядеть следующим образом:

```
/* устанавливаем 3 логических канала */
int32_t err = X502_SetLChannelCount(hnd, 3);
if (err == X502_ERR_OK) {
    /* первый логический канал соответствует измерению 1 канала
          относительно общей земли */
    err = X502_SetLChannel(hnd,0,0,X502_LCH_MODE_COMM,
      X502_ADC_RANGE_10,0);
}
if (err == X502_ERR_OK) {
     /* второй логический канал соответствует измерению 16 канала
        в диф. режиме */
     err = X502_SetLChannel(hnd,1,15,X502_LCH_MODE_DIFF,
      X502_ADC_RANGE_1, 0);
}
if (err == X502_ERR_OK) {
    /* третий логический канал - измерение 17-го канала
       относительно общей земли */
    err = X502_SetLChannel(hnd,2,16,X502_LCH_MODE_COMM,
      X502_ADC_RANGE_02, 0);
}
```

```
if (err == X502_ERR_OK) {
    /* установка других настроек */
}
if (err == X502_ERR_OK) {
    /* передаем настройки в модуль */
    err = X502_Configure(hnd,0);
}

if (err != X502_ERR_OK) {
    /* произошла ошибка при настройке параметров... */
}
```

После завершения измерения с настройками, соответствующими последнему логическому каналу, следует измерение, соответствующее снова первому (с нулевым номером) логическому каналу. Последовательность измерений соответствующая одному проходу логической таблицы называется кадром.

При желании, между завершением измерения, соответствующего последнему логическому каналу кадра, и началом измерения, соответствующего первому логическому каналу следующего кадра, может быть вставлена межкадровая задержка.

3.6.2 Настройка частоты синхронного ввода/вывода

Все частоты потокового сбора и выдачи данных основываются на опорной частоте синхронизации. В качестве опорной частоты может использоваться внутренний источник частоты или внешний. В первом случае, опорная частота может быть 2МГц либо 1.5МГц. По умолчанию используется 2МГц. Изменить ее можно с помощью функции X502_SetRefFreq().

При внешней опорной частоты может использоваться сигнал с произвольной частотой до 1.5 МГц. При этом для того, чтобы функции библиотеки могли оптимально подобрать настройки передачи данных (если они не задаются вручную), а также для того, чтобы корректно работали функции подбирающие делители (см. ниже) для получения нужных частот ввода/вывода необходимо задать значение внешней опорной частоты, которая будет подана. Это значение можно задать с помощью функции X502_SetExtRefFreqValue().

Частота сбора АЦП получается с помощью деления значения опорной частоты на установленный коэффициент, который может быть в диапазоне от 1 до X502_ADC_FREQ_DIV_MAX. Кроме того, как уже упоминалось в предыдущем разделе, между измерением последнего логического канала одного кадра и началом следующего кадра, может быть добавлена межкадровая задержка. Межкадровая задержка задается в виде количества периодов опорной частоты синхронизации.

Делитель частоты сбора АЦП и количество периодов опорной частоты для межкадровой задержки можно задать явно с помощью функций X502_SetAdcFreqDivider() и X502_SetAdcInterframeDelay() соответственно. Вместо этих функций для удобства можно использовать функцию X502_SetAdcFreq(), которой можно передать значения частоты сбора АЦП и частоты кадров в Герцах, а функция сама рассчитает нужный делитель и значение межкадровой задержки, чтобы полученные частоты были наиболее близки к указанным. При этом функция вернет реально установившиеся значения частот.

Под частотой сбора АЦП (f_acq) понимается величина, обратная времени одного преобразования, соответствующего одному логическому каналу. Под частотой кадров (f_frame) понимается величина, обратная времени от начала измерения первого логического канала одного кадра до начала измерения первого логического канала следующего кадра. Это частота соответствует частоте сбора для одного логического канала.

Ниже приведена диаграмма, иллюстрирующая на примере сбора при заданных трех логических каналах, как определяются упомянутые выше частоты.

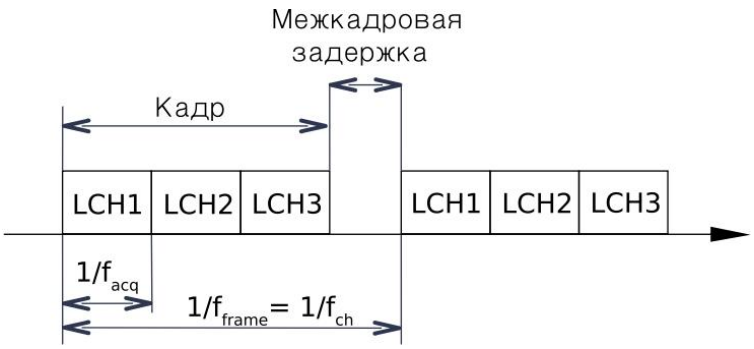

Рис. 3.1: Диаграмма сбора АЦП для трех логических каналов

Если межкадровая задержка не нужна, то можно передать нулевой указатель в качестве второго параметра X502_SetAdcFreq(), тогда будет использоваться всегда нулевая межкадровая задержка (т.е. измерение следующего кадра начнется сразу после завершения предыдущего).

Помимо синхронного ввода с АЦП модули L-502, E-502 и E16 позволяют осуществлять синхронный ввод с цифровых входов (количество зависит от типа модуля, что описано в разделе Отличие возможностей модулей). Также как и для синхронного сбора данных АЦП, частота синхронного цифрового ввода определяется как опорная частота, деленная на коэффициент, который можно установить с помощью X502_SetDinFreqDivider(). Так же можно вызывать функцию X502_SetDinFreq(), чтобы она рассчитала этот коэффициент для получения наиболее близкой частоты к указанной. Для синхронного ввода цифровых линий нет ни логической таблицы (так как каждый раз считывается значение всех цифровых входов и передается в виде одного слова), ни межкадровой задержки — при запуске синхронного ввода все измерения выполняются через одинаковые промежутки времени. При этом частота для ввода с цифровых линий может отличаться от частоты сбора АЦП.

Также модули L-502, E-502 и E16 позволяют осуществить синхронный вывод параллельно на два канала ЦАП (опция) и цифровые выводы. При этом максимальная частота вывода каждого канала в два раза меньшей значения опорной частоты. Для модуля E-502, а также L-502 с прошивкой ПЛИС версии 0.5 или выше, можно установить делитель частоты вывода (относительно опорной частоты в диапазоне от X502_OUT_FREQ_DIV_MIN до X502_OUT_FREQ_DIV_MAX) либо явно с помощью функции X502_SetOutFreqDivider(), либо вызвать функцию X502_SetOutFreq(), чтобы она подобрала такой делитель так, чтобы частота была наиболее близка к заданной. При этом частота задается общая для всех потоков вывода.

3.6.3 Коэффициент усреднения для логического канала (не реализовано в E16)

Реально микросхема АЦП всегда работает на частоте равной опорной частоте синхронизации. В случае, если частота сбора АЦП установлена меньше, чем опорная частота синхронизации, то на одно измерение значения логического канала приходится n измерений АЦП (n = f_acq/f_ref — отношение установленной частоты сбора АЦП к опорной частоте дискретизации).

При отключенном усреднении, первые n-1 измерений отбрасывается, что увеличивает время установления сигнала. При необходимости можно использовать несколько последних отсчетов (navg) для получения результирующего значения. Тогда результирующего значение будет являться средним между navg последними измерениями, однако это сокращает соответственно время на установления сигнала. Естественно navg всегда меньше либо равно n. Кроме того navg не может превышать максимального значения, равного X502_LCH_AVG_SIZE_MAX.

Значение navg задается последним параметром функции X502_SetLChannel(). Значение равное 1 означает отсутствие усреднения. Значения равное 0 означает, что коэффициент усреднения может быть выбран по усмотрению библиотеки. В текущей реализации значение 0 аналогично значению 1, но это может быть изменено в последующих версиях.

3.6.4 Настройка режимов синхронизации

По умолчанию в качестве опорной частоты синхронизации используется внутренняя частота модуля, а запуск всех синхронных измерений осуществляется при выполнении функции X502_StreamsStart().

Однако, при необходимости, возможно задать как внешний источник опорной частоты, так и внешний сигнал запуска синхронного сбора/выдачи данных.

Для этого могут быть использованы входы цифрового разъема DI_SYN1 и DI_SYN2 (может использоваться как фронт, так и спад одного из этих сигналов), либо также может использоваться разъем синхронизации для организации синхронного сбора данных по принципу ведущий-ведомые. В модуле E16 для этих же целей используются контакты INT (19 контакт цифрового разъема) и TRIG (20 контакт аналогового разъема), при этом в E16 нет выделенных портов (START/CONV)_(IN/OUT), а TRIG и INT могут быть настроены как на вход, так и на выход и играть роль входов или выходов сигналов START и CONV.

Выбор внешнего сигнала для задания опорной частоты синхронизации задается с помощью X502_SetSyncMode(), а условие запуска с помощью функции X502_SetSyncStartMode(). Следует отметить, что если задано внешнее событие запуска, то для того, чтобы модуль перешел в режим ожидания этого события, необходимо вызвать X502_StreamsStart().

Останов синхронного сбора/выдачи данных всегда осуществляется программно с помощью X502_StreamsStop().

При использовании разъема синхронизации для организации сбора данных по принципу ведущий-ведомые, для ведущего модуля источником опорной частоты синхронизации остается внутренняя частота (режим X502_SYNC_INTERNAL), а каждый ведомый модуль использует опорную частоту и/или признак запуска сбора от внешнего мастера, т.е. для каждого ведомого модуля должен быть установлен режим X502_SYNC_EXTERNAL_MASTER.

3.7 Синхронный и асинхронный режимы работы

Для модулей L-502, E-502 и E16 доступны следующие данные на ввод:

- отсчеты с АЦП
- значения цифровых входов

Также модуль может быть использована для вывода:

- отсчетов на первый канал ЦАП
- отсчетов на второй канал ЦАП
- значений на цифровые выходы

Таким образом, имеется 2 канала на ввод и 3 канала на вывод.

Каждый из этих каналов может работать как в синхронном режиме, так и асинхронно. При этом каждый канал может быть настроен индивидуально, то есть можно выполнять, например, асинхронный ввод цифровых линий на фоне синхронного потокового сбора с АЦП или выводить на один канал ЦАП сигнал в синхронном потоковом режиме, в то время как в другой выставлять значения асинхронно. Единственное исключение — невозможно осуществить асинхронный ввод с АЦП на фоне синхронного сбора данных с цифровых входов.

Также существует ограничение использования асинхронного режима вместе с синхронным в случае запуска синхронного ввода-вывода по внешнему условию. В момент времени, когда модуль ожидает внешний сигнал запуска синхронного ввода-вывода (т.е. после того, как была вызвана функция X502_StreamsStart(), но до того, как возникло внешнее условие старта), асинхронный ввод или вывод невозможен.

3.7.1 Асинхронный режим работы

При включении питания все каналы находятся в асинхронном режиме. В асинхронном режимы при вызове функции асинхронного ввода/вывода производится однократный ввод или вывод указанной информации. При этом задержка от вызова функции до непосредственно момента измерения данных для ввода или выставления указанного значения на выходе для вывода точно не определена. Также точно не может быть определена задержка между двумя последовательными операциями ввода/вывода.

Плюсом асинхронного режима является простота его использования — достаточно одного вызова требуемой функции:

- X502_AsyncInDig() - асинхронный ввод значений цифровых линий
- X502_AsyncOutDig() - асинхронный вывод значений на цифровые линии
- X502_AsyncOutDac() - асинхронный вывод значения на один из каналов ЦАП

Для однократного ввода данных с АЦП используется функция X502_AsyncGetAdcFrame(), которая выполняет ввод одного кадра данных АЦП. В отличии от других функций асинхронного ввода-вывода, перед вызовом данной функции необходимо выполнить настройку модуля: необходимо задать управляющую таблицу АЦП (см. Настройка последовательности опроса каналов АЦП). Измерение логических каналов внутри кадра происходит синхронно с заданной частотой сбора АЦП. Асинхронным является ввод самих кадров, то есть задержка между измерением кадров при последовательном вызове X502_AsyncGetAdcFrame() не определена.

Например, код для выполнения однократного измерения с 7-го физического канала в дифференциальным режиме с диапазоном +/-0.5В может выглядеть следующим образом:

```
/* устанавливаем 1 логический канал в управляющей таблице */
int32_t err = X502_SetLChannelCount(hnd, 1);
if (err == X502_ERR_OK) {
    /* логический канал соответствует измерению 7 канала
        в диф. режиме */
    err = X502_SetLChannel(hnd,0,6,X502_LCH_MODE_DIFF,L502_ADC_RANGE_05,0
      );
}
if (err == X502_ERR_OK) {
    /* передаем настройки в модуль */
    err = X502_Configure(hnd,0);
}
if (err == X502_ERR_OK) {
    /* Считываем кадр данных АЦП из одного отсчета */
    double val;
    err = X502_AsyncGetAdcFrame(hnd,
            X502_PROC_FLAGS_VOLT, 1000, &val);
    if (err == X502_ERR_OK) {
        /* верно считали значение val */
    }
}
```

3.7.2 Синхронный режим работы

В синхронном режиме ввод или вывод данных осуществляется с заданной частотой, то есть время между соседними измерениями или выводом соседний отсчетов определено. Частоты сбора для каждого канала задаются относительно общей опорной частоты синхронизации (подробнее см. главу "Настройка частоты синхронного ввода/вывода") и запуск синхронного ввода-вывода для всех каналов осуществляется одновременно.

Для запуска синхронного режима, необходимо сперва с помощью функции X502_StreamsEnable() разрешить синхронный режим по требуемым каналам, а затем запустить синхронный ввод/вывод по всем разрешенным каналам с помощью X502_StreamsStart().

При синхронном вводе модуль производит измерения с заданной частотой и сам передает данные по интерфейсу в буфер (для L-502 этот буфер находится в драйвере и передается модулем с использованием BusMaster DMA, а для E-502 — буфер выделяется бибилотекой). Принятые в буфер данные могут быть прочитаны программой с помощью X502_Recv().

Аналогично, для синхронного вывода, модуль сам по мере необходимости считывает данные из буфера и выводит считанные отсчеты с заданной частотой. Данные в буфер драйвера должны быть предварительно записаны с помощью X502_Send(). При этом, если к моменту вывода очередного отсчета данные в буфер драйвера не поступили, то будет выведено предыдущее значение.

В драйвере или библиотеке выделяется всего два буфера — один на прием и один на передачу. То есть значения для синхронного ввода с цифровых линий и отсчеты АЦП передаются одним потоком, также одним потоком передаются все данные на вывод. Каждый отсчет передается в виде 32-битного слова, содержащего дополнительную информацию, включающую в себя признак, к какому типу данных относится данный отсчет.

Разбор принятых данных на отсчеты АЦП и значения цифровых выводов осуществляется с помощью X502_ProcessData(). Помимо этого, данная функция также может осуществить перевод отсчетов АЦП в Вольты. Следует учесть, что в отличие от некоторых других изделий "Л Кард", применение калибровочных коэффициентов осуществляется аппаратно и значения уже приходят в виде 24-битных отсчетов с уже примененными коэффициентами.

В 32-битном слове, соответствующем отсчету АЦП, передается дополнительно режим измерения и номер физического канала. X502_ProcessData() сравнивает эти значения с теми, которые были заданы при настройке управляющей таблицы, чтобы убедиться в корректности принимаемых данных. При этом X502_ProcessData() ожидает, что с ее помощью будут обрабатываться все принятые данные.

Данные от АЦП приходят в том порядке, в котором производятся измерения, т.е. сперва измерения соответствующие всем логическим каналам первого кадра, затем второго и т.д. В этом же порядке X502_ProcessData() возвращает и преобразованные отсчеты АЦП. При этом в X502_ProcessData() можно передавать и нецелое количество кадров (например, если запущен синхронный ввод с цифровых линий, то заранее сложно определить, сколько в принятом блоке данных содержится отсчетов с цифровых линий, а сколько отсчетов с АЦП), в этом случае X502_ProcessData() обработает и выдает все отсчеты, включая отсчеты нецелого кадра, а при следующем ее вызове проверяет, что отсчеты АЦП начинаются с логического канала, следующего за последним обработанным до этого каналом. Какой логический канал ожидается следующим для обработки можно узнать с помощью функции X502_GetNextExpectedLchNum().

Например, пусть в управляющей таблице АЦП установлено 7 логических каналов. Если был принят блок данных от модуля содержащий 5 отсчетов АЦП (и произвольное количество значений цифровых входов, если включен синхронный ввод с цифровых линий) и обработан с помощью X502_ProcessData(), то X502_ProcessData() вернет преобразованные 5 отсчетов АЦП, соответствующие логическим каналом с индексами 0,1,2,3,4. Следующий логический канал, который ожидается для обработки — логический канал с индексом 5, поэтому X502_GetNextExpectedLchNum() вернет значение 5. Если обработать следующий блок данных, содержащий 5 следующих отсчетов, то X502_ProcessData() вернет отсчеты соответствующие каналам с индексами 5,6,0,1,2. Т.е. с помощью вызова X502_GetNextExpectedLchNum() можно узнать, какому логическому отсчету будет соответствовать первый элемент выходного массива при последующем вызове X502_ProcessData().

Следует учесть, что по умолчанию буфер в драйвере или библиотеке рассчитан на количество отсчетов, которое будет введено за 4с непрерывного сбора. Если вовремя не считывать данные с помощью X502_Recv(), то произойдет переполнение буфера в драйвере и часть данных, для которых не нашлось в буфере места, будет потеряна. При последующем появлении места в буфере, в то место в потоке, где произошел разрыв непрерывного потока данных, будет вставлено слово, представляющее собой сообщение о переполнении буфера. Если X502_ProcessData() во входном массиве обнаружит это слово, то функция вернет ошибку X502_ERR_STREAM_OVERFLOW. При этом, как и в случае возникновения других ошибок обработки, все отсчеты, которые были до возникновения ошибки будут обработаны и возвращены в выходных массивах (размеры которых будут соответственно обновлены).

Аналогично, для формирования общего потока на вывод в требуемом формате используется функция X502_PrepareData(), принимающая данные из трех массивов и сохраняющая их во внешний массив. Если какой-либо из источников не должен использоваться, то в качестве массива передается нулевой указатель. Для каналов, которые не были настроены на синхронный режим с помощью X502_StreamsEnable(), входной массив на анализируется и данные из него не используются..

Следует отметить, что если для синхронного ввода инициализация потока передачи происходит по X502_StreamsStart(), так как данные начнут поступать только после запуска синхронного ввода, то с синхронным выводом дело обстоит несколько иначе. Так как по X502_StreamsStart() уже должна начаться выдача синхронных данных, то

часть данных уже должна быть загружена в модуль. Таким образом, после разрешения синхронного вывода по нужным каналам с помощью X502_StreamsEnable() и до запуска синхронного вывода с помощью X502_StreamsStart() необходимо осуществить предзагрузку части данных синхронного потока. Для этого следует вызвать функцию X502_PreloadStart(), по которой в драйвере или библиотеке будет выделен буфер на передачу и инициализирован поток на передачу, а затем записать часть синхронных данных в буфер драйвера с помощью X502_PrepareData() и X502_Send(). Если этого не сделать, то синхронный вывод начнется лишь после того, как данные будут записаны в модуль и не будет привязан к началу синхронного сбора/выдачи данных.

Кроме того, для синхронной выдачи данных на ЦАП рекомендуется предварительно установить начальные значения на ЦАП с помощью функции асинхронного вывода. В противном случае при начале синхронного вывода может быть небольшой переходный процесс от значения на ЦАП, которое было до запуска синхронного вывода, до выставления первых нужных значений, т.к. ЦАП имеет свой фильтр и ограничения на скорость изменения сигнала.

3.7.3 Циклический вывод

Для модуля L-502, начиная с версии 1.0.4 драйвера и библиотеки, а также для модуля E-502 (начиная с версии 1.1.0), введена поддержка циклического вывода на ЦАП и цифровые выводы. Этот режим позволяет загрузить сигнал полностью в буфер внутри драйвера (для L-502) или процессора Cortex-M4 (для E-502), содержимое которого будет циклически выводиться без необходимости дальнейшей подкачки.

Данные для загрузки в циклический буфер подготавливаются также как и для потокового вывода с помощью X502_PrepareData() и записываются с помощью X502_Send() и могут содержать комбинацию данных на оба канала ЦАП и на цифровые выводы. Циклический вывод является вариантом синхронного вывода и для его работы нужно разрешить нужные потоки на вывод через X502_StreamsEnable() и должен быть запущен синхронный ввод-вывод через X502_StreamsStart(). Также как и с обычным потоковым выводом, часть каналов может использоваться для вывода циклического сигнала, а часть — асинхронно. Однако нельзя часть каналов вывода использовать в циклическом режиме, а часть в потоковом режиме с подкачкой (естественно, циклический режим на вывод можно использовать с потоковым на ввод).

Для вывода циклического сигнала используется двойная буферизация — то есть может быть выделено два буфера, пока из одного сигнал циклически выводится, в другой может подгружаться следующий сигнал. Смена сигнала происходит по концу периода предыдущего. При этом после записи одного сигнала необходимо, чтобы успела пройти смена сигналов перед тем как можно будет загружать следующий, в противном случае функция X502_OutCycleLoadStart() вернет ошибку (что можно использовать как признак, что буфер еще не готов для загрузки нового сигнал). Проверку завершения смены сигнала можно сделать при соответствующих версиях ПО (см. описание функции X502_OutCycleCheckSetupDone()) и явно с помощью функции X502_OutCycleCheckSetupDone() или используя флаг X502_OUT_CYCLE_FLAGS_WAIT_DONE во время вызова X502_OutCycleSetup(), чтобы функция вернула управление только после выполнения смены сигналов.

Для загрузки сигнала сперва вызывается функция X502_OutCycleLoadStart(), в которой задается размер циклического буфера, который будет использован для хранения отсчетов всех используемых каналов вывода. Например, если нужно использовать два канала ЦАП, на каждый из которых вывести сигнал из 1000 точек, то размер дол-

жен быть указан 2000. После этого отсчеты загружаются как и при потоковом выводе с подкачкой с помощью функций X502_PrepareData() и X502_Send(). При этом суммарно должно быть записано ровно столько же отсчетов, сколько было указано при вызове X502_OutCycleLoadStart(). После загрузки по вызову X502_OutCycleSetup() происходит переключение на загруженный буфер, при этом, в зависимости от текущего состояния, это приводит к следующему:

- если синхронный ввод-вывод не запущен (не было вызова X502_StreamsStart()) то начинается предзагрузка циклического сигнала в модуль, однако реально выдача сигнала начнется только при вызове X502_StreamsStart() (или по внешнему условию запуска). Это позволяет привязать начало вывода первого отсчета циклического сигнала к началу ввода. При этом необходимо вызывать X502_OutCycleSetup() с флагом X502_OUT_CYCLE_FLAGS_WAIT_DONE, чтобы гарантировать, что загрузка сигнала завершится до X502_StreamsStart() (актуально в первую очередь для E-502, где передача данных идет по интерфейсу и может занимать значительное время).

- если синхронный ввод-вывод запущен, но не было выведено ни одного циклического сигнала до этого, то по X502_OutCycleSetup() начинается вывод циклического сигнала.

- если синхронный ввод-вывод запущен и уже выводится предыдущий циклический сигнал, то после вызова X502_OutCycleSetup() в драйвере (или модуле для E-502) будет взведен флаг о необходимости переключить сигналы. После этого события по достижению конца предыдущего циклического буфера будет произведена смена буферов вывода. Таким образом это позволяет производить смену сигнала всегда в известной точке. При смене сигнала происходит освобождение старого буфера и только после реальной смены можно будет вызвать следующий раз X502_OutCycleLoadStart() для загрузки следующего сигнала. При использовании флага X502_OUT_CYCLE_FLAGS_WAIT_DONE, функция вернет управление только в момент, когда сама смена уже произойдет (если эта возможность поддерживается ПО).

Останов циклического вывода можно осуществить одним из следующий способов:

- X502_OutCycleStop() останавливает циклический вывод после вывода последней точки на границе циклического буфера. То есть эта функция используется, чтобы циклический вывод был завершен точно в известной точке и на выходах остались значения, соответствующие последним отсчетам в циклическом сигнале. При этом сама функция не дожидается останова вывода, если не указан флаг X502_OUT_CYCLE_FLAGS_WAIT_DONE.

- X502_StreamsDisable() с указанием всех используемых каналов вывода приводит к немедленному завершению вывода и освобождением всех буферов. На выходах останутся значения, которые были в момент вызова. После этого можно будет заново разрешить и проинициализировать вывод как в циклическом так и в потоковом режиме с подкачкой.

- X502_StreamsStop() приводит к немедленному останову всех потоков и остановку генерации опорной частоты синхронизации. На выходе остаются те значения, которые были в момент вызова функции.

3.7.4 Размер буфера и шаг для синхронного режима

В данном разделе приводится дополнительная информация о том, как можно настроить дополнительные параметры, управляющие передачей потока данных в синхронном режиме между модулем и ПК. Эти параметры по умолчанию настраиваются библиотекой автоматически. Предполагается, что большинству пользователей должны подойти автоматически настраиваемые параметры и данный раздел не является обязательным. Однако для случаев, когда автоматически определенные параметры не подходят, пользователь может задать их самостоятельно. Для этого в этом разделе приводится описание, как выбирается размер буфера и шаг библиотекой, что означают эти параметры и как их можно настроить вручную.

Как уже было сказано в предыдущем разделе прием и передача синхронных данных осуществляется через буфера в драйвере или библиотеке – один буфер на прием, один на передачу.

Выделение буфера на ввод осуществляется по X502_StreamsStart(), если был разрешен хотя бы один источник для синхронного ввода. Выделение буфера на вывод осуществляется по X502_PreloadStart().

При этом размер буфера определяется автоматически библиотекой в зависимости от установленной частоты передачи данных. Размер буфера рассчитывается так, чтобы его хватило на 4 секунды при синхронном вводе и на 3 секунды при синхронном выводе.

Вторым параметром, характеризующим передачу, является шаг передачи. Для модуля L-502 этот параметр определяет шаг прерываний. Передача данных между буфером драйвера и модулем осуществляется непосредственно самим модулем по DMA. При этом, чтобы драйвер мог узнать о том, что данные были записаны в буфер или прочитаны из него, при передаче определенного количества отсчетов модуль генерирует прерывание. То есть реально драйвер “узнает” о том, что были переданы данные только после того как будет передано заданное количество отсчетов, называемое в данной главе шагом прерываний (Точнее сказать, не позже, чем будет передано заданное количество отсчетов, так как драйвер может прочитать значение счетчика переданных данных из модуля и по другим условиям).

Таким образом, малый шаг прерываний позволяет драйверу раньше узнает о принятых или переданных данных, но приводит к большей загрузке системы. Библиотека рассчитывает шаг прерываний так, чтобы прерывания происходили с частотой 64 раза в секунду.

Для модуля E-502 этот параметр определяет используемый размер запроса по USB. Кроме того при вводе данные ставятся на передачу в ПК при из размере равным шагу, но при этом при отсутствие поступления новых данных могут быть поставлен на передачу и меньший объем данных.

Если пользователя по каким-либо причинам не устраивают эти значения он может настроить их вручную с помощью функций X502_SetStreamBufSize() и X502_SetStreamStep(). Эти функции должны быть вызваны до инициализации потоков передачи (до X502_StreamsStart() или X502_PreloadStart()).

В частности, случаями когда значения библиотеки могут не устроить, могут быть следующие:

- Пользователь использует свою прошивку BlackFin и использует каналы синхронных данных для передачи пользовательских данных, которые сильно изменяют скорость передачи данных. В этом случае библиотека не может правильно определить частоту передачи, так как не знает скорости передачи пользовательских данных.

- Пользователь изменяет каналы, которые используются в синхронном режиме, на лету (после X502_StreamsStart()) и при этом скорости передачи по этим каналам существенно отличаются. Так как расчет размера буфера выполняется при инициализации канала, то он осуществляется только по тем каналам, которые были разрешены на тот момент. Если, например, был разрешен только синхронный сбор с АЦП на относительно небольшой частоте, то буфер будет выделен также небольшой. При этом, если после запуска сбора данных будет разрешен синхронный ввод с цифровых линий на частоте 2МГц, то вероятнее всего этот буфер окажется недостаточного размера, и с большой вероятностью произойдет его переполнение. Если же оба этих потока были разрешены изначально, а потом синхронный ввод цифровых линий будет запрещен, то рассчитанный изначально шаг прерывания будет слишком большим и данные от медленного потока АЦП будут обновляться с большими задержками. Если же частоты каналов соизмеримы, то включение/отключение одного из них не приведет к существенному изменению параметров. Изменение шага прерывания и размера буфера при запущенном сборе данных на текущий момент невозможно.

3.8 Особенности работы по интерфейсу Ethernet и настройка сетевых параметров

Если USB-интерфейс в модуле E-502 всегда работает и не требует дополнительной конфигурации, то для работы по интерфейсу Ethernet необходимо выполнить настройку параметров интерфейса и разрешить данный интерфейс. Следует отметить, что при разрешенном Ethernet-интерфейсе с модулем можно работать как по USB, так и по Ethernet. При этом сбор/генерация данных могут выполняться одновременно только по одному интерфейсу (по которому пришла команда на запуск сбора/выдачи).

Основными параметрами для работы по Ethernet являются:

- IP-адрес устройства. Записывается как 4 цифры от 0 до 255 (модуль поддерживает только протокол IPv4), разделенные точками (например, 192.168.0.10). Состоит из адреса подсети и адреса устройства внутри подсети. Последний должен быть уникальным внутри подсети.

- Маска подсети. Определяет какая часть адреса относится к адресу подсети, а какая к адресу устройства. Маска 255.255.255.0 означает, что первые 3 цифры (192.168.0) обозначают адрес подсети, последняя цифра (10) - адрес устройства в подсети.

- IP-адрес шлюза. Используется только когда модуль E-502 и хост, с которого выполняется управление модулем, находятся в разных подсетях. Модуль передает пакеты по адресу шлюза, если адрес назначения находится не в той же подсети, что и модуль. При работе в локальной сети не используется.

- MAC-адрес модуля (6 цифр от 0 до 255, которые записываются в 16-ричном формате). Физический адрес устройства, который должен быть уникален внутри локальной сети. В “Л Кард” для каждого модуля прописывается свой заводской MAC-адрес, который нельзя изменить. Однако, при необходимости, пользователь может задать свой пользовательский MAC адрес и разрешить его использование вместо заводского. При этом всегда есть возможность вернутся к заводскому MAC-адресу, запретив пользовательский.

- Имя экземпляра устройства. Уникальное имя данного экземпляра в виде строки (до 64 английских символов или 32 русских). Используется для возможности автоматического обнаружения модулей в локальной сети (подробнее в главе Обнаружение модулей в локальной сети).

Для работы в первую очередь необходимо задать правильные IP-параметры (адрес, маску и, при необходимости, адрес шлюза), подробнее о чем описано в соответствующем разделе FAQ.

IP-параметры модуля E-502 могут быть установлены 3-мя способами:

- Заданы вручную (статические параметры). Пользователь сам должен позаботиться о том, чтобы адрес принадлежал нужной подсети и был уникальный в ней.

- Получены автоматически от DHCP-сервера. Если задано автоматическое получение адреса и в локальной сети присутствует DHCP-сервер, то модуль делает запрос к нему и использует выделенные DHCP-сервером IP-адреса.

- Может использоваться автоматически получаемый локальный (link-local) адрес (в соответствии с RFC3927). Адрес выбирается случайным образом в диапазоне от 169.254.1.0 до 169.254.254.255 и проверяется, что в сети нет другого устройства с таким адресом (если есть, то идет попытка выбора следующего адреса и т.д.). Это делает возможным подключение к устройству в локальной сети без специальной конфигурации. Следует однако отметить, что т.к. адрес действителен только в одной сети, то два разных устройства в разных сетях могут иметь одинаковый link-local адрес, что приводит к тому, что если у ПК несколько активных интерфейсов (и на обоих используется link-local адрес или наоборот обычный адрес), то хост не знает на каком интерфейсе искать нужное устройство. Т.е. для подключения по link-local адресу на ПК должен быть либо один активный сетевой интерфейс, либо на нужном интерфейсе должен использоваться link-local адрес, а на другом статический или полученный по DHCP адрес.

При включении автоматического получения адреса модуль выбирает себе link-local адрес (и проверяет его уникальность), параллельно выполняя поиск в сети DHCP-сервера. Если DHCP сервер не обнаружен, то используется link-local адрес. Как только будет обнаружен DHCP-сервер, то предпочтение отдается полученному от него адресу (в частности при старте в сети с DHCP сервером модуль может некоторое время до получения адреса от него использовать link-local адрес). Подобный алгоритм используется при автоматическом получении адреса в частности в ОС Windows, а также и во многих дистрибутивах Linux (иногда предоставляя возможность отдельного разрешения DHCP и link-local адреса). Следует также иметь ввиду, что автоматически получаемый адрес модуль проверяет на уникальность в сети, поэтому существует задержка в несколько секунд от подключения к сети модуля до назначения адреса. Автоматическое получение адреса не требует дополнительных настроек, однако при этом неизвестен адрес устройства со стороны ПК для установления соединения. Для решения этой проблемы возможно использование процедуры поиска устройств в локальной сети, описанной в следующем разделе.

Изменить сетевые настройки можно с помощью программы X502Studio.

Также изменение сетевых настроек модуля возможно программным образом через API библиотеки. Для этого существует отдельный тип описателя конфигурации t_e502_eth_config_hnd. Для изменения конфигурации нужно выполнить следующие шаги:

1. Создать описатель конфигурации с помощью E502_EthConfigCreate().

2. Прочитать текущую конфигурацию устройства с помощью E502_EthConfigRead() (соединение с модулем должно быть установлено)

3. Можно получить нужные параметры с помощью функций E502_EthConfigGetXXX() и/или установить новые значения с помощью функций E502_EthConfigSetXXX().

4. После завершения изменений можно записать измененную конфигурацию в модуль с помощью E502_EthConfigWrite(). Модуль сохранит новую конфигурацию в энергонезависимой памяти, запрещает Ethernet-интерфейс, после чего снова его переинициализирует уже с новыми параметрами. При этом, если соединение с устройством было выполнено по Ethernet, то для дальнейшей работы нужно разорвать соединение и установить заново (используя новые параметры)

Для избежания непреднамеренного изменения конфигурации по сети, конфигурация может быть защищена простым паролем. Пока пароль не установлен в качестве пароля нужно передавать пустую строку. Установка нового пароля выполняется аналогично любым другим изменениям параметров конфигурации (с помощью E502_EthConfigSetNewPassword()).

В случае, если пароль забыт, то можно установить соединение по USB и изменить конфигурацию (включая пароль), введя в качестве текущего пароля серийный номер модуля.

3.9 Обнаружение модулей в локальной сети

В отличие от интерфейсов USB и PCI-Express, для интерфейса Ethernet нет стандартной возможности определения подключенных устройств. Однако есть ряд протоколов, реализация которых позволяет обнаружить устройства заданного типа в локальной сети. Для этой возможности модуль E-502 поддерживает протоколы mDNS (в соответствии с RFC6762) и DNS-SD (RFC6763). В соответствии с ними, каждое устройство при подключении объявляет набор сервисов, который оно поддерживает.

Чтобы различать экземпляры устройств, поддерживающие одинаковый тип сервисов, у каждого экземляра есть свое уникальное имя. Это имя задается во время конфигурации модуля. Если оно не установлено, то в качестве имени экземпляра используется “E502_<серийный_номер>”, однако пользователь может задать свое имя, характеризующее назначение конкретного модуля для более наглядной идентификации. Кроме имени экземпляра у каждого сервиса может быть набор текстовых параметров, описывающих данный экземпляр (для E-502 это параметры задающие имя типа устройства (поле devname, значение равно всегда E-502) и серийный номер (поле serial).

В соответствии с этим протоколом у хоста в локальной сети есть возможность найти все экземпляры заданного сервиса в сети. Для обнаружения должна быть запущена соответствующая служба (или демон), отслеживающая изменения наличия устройств в сети, а уже функции e502api работают с данной службой. В ОС Linux в качестве реализации данного протокола используется демон Avahi, который включен в большинство современных дистрибутивов и входит в стандартную установку (или нужно установить соответствующий пакет вручную). В ОС Windows используется служба Bonjour, которая штатно не установлена, однако установщик включен в “L-Card L502/E502 SDK” и

данная служба будет установлена при выборе соответствующего пункта (следует отметить, что так как служба является отдельным продуктом, который может использоваться и другим ПО, то она не удаляется автоматически при удалении "L-Card L502/E502 SDK" . При необходимости следует вручную удалить службу через установку и удаление программ в "Панели управления").

Использование данного API позволяет автоматически обнаруживать подключенные в локальной сети устройства, во многом подобно другим интерфейсам. Однако существуют следующие особенности:

- Обнаружение модуля связано с посылкой пакетов и приемом ответов, которые могут быть потеряны при определенных условиях и потребовать переповторов. Это выполняется на уровне протокола и невидимо для пользователя, однако надо иметь ввиду, что обнаружение устройства может потребовать некоторого времени.

- Так как отключение отслеживается на уровне протокола, нет оповещения о физическом отключении кабеля. Соответственно при выключении питания или выдергивании кабеля отключение модуля может быть не обнаружено на протяжении длительного времени.

Для поиска устройств следует вызвать E502_EthSvcBrowseStart(), после чего использовать полученный контекст поиска устройств в сети для последующих вызовов E502_EthSvcBrowseGetEvent(). Каждый вызов возвращает информацию максимум об одном событии и немедленно возвращает управление, как только оно произошло. Каждому новому обнаруженному модулю соответствует событие E502_ETH_SVC_EVENT_ADD. В случае изменения параметров (например адреса) приходит событие E502_ETH_SVC_EVENT_CHANGED, а при исчезновении (при условии описанных выше особенностей) — E502_ETH_SVC_EVENT_REMOVE. Для каждого события возвращается описатель сетевого сервиса, по которому можно определить, какому модулю соответствует событие (узнать имя экземпляра и серийный номер модуля), а также запросить IP-адрес модуля. Для каждого события (кроме случая, когда событие не обнаружено и возвращен код E502_ETH_SVC_EVENT_NONE) этот описатель необходимо освободить с помощью E502_EthSvcRecordFree() как только он станет ненужным. В простейшем случае можно вызвать E502_EthSvcBrowseGetEvent() пока не будет обнаружено появление нужного устройства или не истечет таймаут на поиск устройства. При желании также можно использовать периодический вызов E502_EthSvcBrowseGetEvent() для постоянного мониторинга устройств в сети. В любом случае, когда поиск устройств закончен, необходимо вызвать E502_EthSvcBrowseStop().

Установить связь с модулем по описателю сетевого сервиса можно как вручную по полученному адресу через E502_EthSvcRecordResolveIPv4Addr(), так создать запись о устройстве с помощью E502_MakeDevRecordByEthSvc() для последующего открытия через X502_OpenByDevRecord().

Глава 4

Константы, типы данных и функции библиотеки

4.1 Константы и перечисления

4.1.1 Константы и макроопределения

Константа	Значение	Описание
X502_LTABLE_MAX_CH_CNT	256	Максимальное количество логических каналов в таблице
E16_LTABLE_MAX_CH_CNT	128	Максимальное количество логических каналов в таблице для E16
X502_ADC_RANGE_CNT	6	Количество диапазонов для измерения напряжений
E16_ADC_RANGE_CNT	4	
X502_ADC_COMM_CH_CNT	32	Количество каналов АЦП в режиме с общей землей
X502_ADC_DIFF_CH_CNT	16	Количество каналов АЦП в дифференциальном режиме
X502_LCH_AVG_SIZE_MAX	128	Максимальное значение для аппаратного усреднения по логическому каналу
X502_ADC_FREQ_DIV_MAX	(1024*1024)	Максимальное значения делителя частоты АЦП
E16_ADC_FREQ_DIV_MAX	(0x800000)	Максимальное значения делителя частоты АЦП для модуля E16 по отношению к опорной частоте E16_REF_FREQ_48000KHZ
X502_DIN_FREQ_DIV_MAX	(1024*1024)	Максимальное значение делителя частоты синхронного цифрового ввода

E16_DIN_FREQ_DIV_MAX	(0x800000)	Максимальное значение делителя частоты синхронного цифрового ввода для модуля E16 по отношению к опорной частоте E16_REF_FREQ_48000KHZ
E16_ADC_FREQ_DEFAULT	500000	Значения частоты АЦП по умолчанию, используется в функции X502_SetAdcFreq и X502_CalcAdcFreq2 если параметр f_acq = 0
E16_DIN_FREQ_DEFAULT	500000	Значения частоты цифрового входа DIN по умолчанию, используется в функции X502_SetDinFreq и X502_CalcDinFreq2 если параметр f_din = 0
E16_OUT_FREQ_DEFAULT	500000	Значения частоты ЦАП и цифровых выходов по умолчанию, используется в функции X502_SetOutFreq и X502_CalcOutFreq2 если параметр f_dout = 0
X502_OUT_FREQ_DIV_MIN	2	Минимальное значение делителя частоты синхронного вывода
X502_OUT_FREQ_DIV_MAX	1024	Максимальное значение делителя частоты синхронного вывода
E16_OUT_FREQ_DIV_MAX	0x8000	Максимальное значение делителя частоты синхронного вывода для модуля E16 по отношению к опорной частоте E16_REF_FREQ_48000KHZ
X502_OUT_FREQ_DIV_DEFAULT	2	Значение делителя частоты вывода по умолчанию (которое также всегда используется в L-502 с версией прошивки ПЛИС ниже 0.5)
X502_OUT_FREQ_DIV_MIN_REF_LF	1	Минимальное значение делителя частоты синхронного вывода при внешней опорной частоте ниже X502_OUT_FREQ_REF_LF_VAL

X502_OUT_FREQ_REF_LF_VAL	700000	Значение внешней опорной частоты, ниже которой разрешено устанавливать значение делителя частоты генерации до X502_OUT_FREQ_DIV_MIN_REF_LF
X502_ADC_INTERFRAME_DELAY_MAX	(0x1FFFFF)	Максимальное значение межкадровой задержки для АЦП
X502_BF_CMD_DEFAULT_TOUT	500	Таймаут по умолчанию для выполнения команды к BlackFin
X502_ADC_SCALE_CODE_MAX	6000000	Код АЦП, соответствующий максимальному значению шкалы
E16_ADC_SCALE_CODE_MAX	(0x80 * 0x7fff)	Код АЦП, соответствующий максимальному значению шкалы
X502_DAC_SCALE_CODE_MAX	30000	Код ЦАП, соответствующий максимальному значению шкалы
E16_DAC_SCALE_CODE_MAX	0x7fff	
X502_DEVNAME_SIZE	32	Максимальное количество символов в строке с названием устройства
X502_SERIAL_SIZE	32	Максимальное количество символов в строке с серийным номером
X502_IMPLEMENTATION_SIZE	16	Максимальное количество символов в строке с идентификатором board implementation
X502_LOCATION_STR_SIZE	64	Максимальное количество символов в строке с описанием подключения
X502_MAC_ADDR_SIZE	6	Размер MAC-адреса для Ethernet интерфейса
X502_INSTANCE_NAME_SIZE	64	Размер строки с описанием экземпляра устройства
X502_PASSWORD_SIZE	32	Максимальный размер строки с паролем на настройки
X502_EXT_REF_FREQ_MAX	1500000	Максимально возможное значение внешней опорной частоты
E502_P1_EXT_REF_FREQ_MAX	2000000	Максимально возможное значение внешней опорной частоты для E502-P1

X502_FLASH_USER_SIZE	0x100000	Размер пользовательской области Flash-памяти
X502_BF_REQ_TOUT	500	Стандартный таймаут на выполнение запроса к BlackFin в мс
X502_DAC_RANGE	5.	Диапазон ЦАП в вольтах
E16_DAC_RANGE	10.	
X502_DAC_CH_CNT	2	Количество каналов ЦАП
X502_DOUT_LINES_CNT	16	Количество цифровых выходов у модуля
X502_STREAM_IN_MSG_OVERFLOW	0x01010000	слово в потоке, означающее, что произошло переполнение
X502_DEVREC_SIGN	0x4C524543	Значение поля сигнатуры в записи о устройстве t_x502_devrec. Признак, что запись действительна (устанавливается функциями по получению записей о устройствах)

4.1.2 События поиска сетевых сервисов

Тип: t_e502_eth_svc_event		
Описание: Коды событий, возникающих при поиске сетевых сервисов, возвращаемые функцией E502_EthSvcBrowseGetEvent()		
Константа	**Значение**	**Описание**
E502_ETH_SVC_EVENT_NONE	0	Ни одного события не произошло
E502_ETH_SVC_EVENT_ADD	1	Обнаружено появление нового сетевого сервиса
E502_ETH_SVC_EVENT_REMOVE	2	Обнаружено исчезновение ранее доступного сетевого сервиса
E502_ETH_SVC_EVENT_CHANGED	3	Изменение параметров ранее обнаруженного сетевого сервиса

4.1.3 Коды ошибок библиотеки

Тип: t_x502_errs		
Описание: Коды ошибок библиотеки		
Константа	**Значение**	**Описание**
X502_ERR_OK	0	Функция выполнена без ошибок
X502_ERR_INVALID_HANDLE	-1	В функцию передан недействительный описатель модуля
X502_ERR_MEMORY_ALLOC	-2	Ошибка выделения памяти
X502_ERR_ALREADY_OPENED	-3	Попытка открыть уже открытое устройство
X502_ERR_DEVICE_NOT_FOUND	-4	Устройство с заданными параметрами не найдено в системе

X502_ERR_DEVICE_ACCESS_DENIED	-5	Доступ к устройству запрещен (Как правило из-за того, что устройство уже открыто в другой программе)
X502_ERR_DEVICE_OPEN	-6	Ошибка открытия устройства
X502_ERR_INVALID_POINTER	-7	В функцию передан недействительный указатель
X502_ERR_STREAM_IS_RUNNING	-8	Функция не может быть выполнена при запущенном потоке сбора данных
X502_ERR_RECV	-9	Ошибка чтения данных синхронного ввода
X502_ERR_SEND	-10	Ошибка записи данных для синхронного вывода
X502_ERR_STREAM_OVERFLOW	-11	Произошло переполнение внутреннего буфера для потока синхронного ввода
X502_ERR_UNSUP_STREAM_MSG	-12	Неизвестное сообщение в потоке синхронного ввода
X502_ERR_MUTEX_CREATE	-13	Ошибка создания системного мьютекса
X502_ERR_MUTEX_INVALID_HANDLE	-14	Неверный описатель мьютекса
X502_ERR_MUTEX_LOCK_TOUT	-15	Истекло время ожидания освобождения мьютекса
X502_ERR_MUTEX_RELEASE	-16	Ошибка освобождения мьютекса
X502_ERR_INSUFFICIENT_SYSTEM_RESOURCES	-17	Недостаточно системных ресурсов
X502_ERR_NOT_IMPLEMENTED	-18	Данная возможность еще не реализована
X502_ERR_INSUFFICIENT_ARRAY_SIZE	-19	Недостаточный размер массива
X502_ERR_FPGA_REG_READ	-20	Ошибка чтения регистра FPGA
X502_ERR_FPGA_REG_WRITE	-21	Ошибка записи регистра FPGA
X502_ERR_STREAM_IS_NOT_RUNNING	-22	Сбор данных уже остановлен
X502_ERR_INTERFACE_RELEASE	-23	Ошибка освобождения интерфейса
X502_ERR_THREAD_START	-24	Ошибка запуска потока
X502_ERR_THREAD_STOP	-25	Ошибка останова потока
X502_ERR_DEVICE_DISCONNECTED	-26	Устройство было отключено
X502_ERR_IOCTL_INVALID_RESP_SIZE	-27	Неверный размер ответа на управляющий запрос
X502_ERR_INVALID_DEVICE	-28	Неверный тип устройства
X502_ERR_INVALID_DEVICE_RECORD	-29	Недействительная запись о устройстве
X502_ERR_INVALID_CONFIG_HANDLE	-30	Неверный описатель конфигурации модуля

X502_ERR_DEVICE_NOT_OPENED	-31	Связь с устройством закрыта или не была установлена
X502_ERR_INVALID_OP_FOR_IFACE	-32	Данная операция не доступна для текущего интерфейса связи с устройством
X502_ERR_FPGA_NOT_LOADED	-33	Не загружен ПЛИС модуля
X502_ERR_INVALID_USB_CONFIGURATION	-34	Неверная конфигурация USB-устройства
X502_ERR_INVALID_SVC_BROWSE_HANDLE	-35	Неверный описатель контекста поиска устройств в сети
X502_ERR_INVALID_SVC_RECORD_HANDLE	-36	Неверный описатель записи о сервисе
X502_ERR_DNSSD_NOT_RUNNING	-37	Не запущена программа обнаружения устройств в локальной сети
X502_ERR_DNSSD_COMMUNICATION	-38	Ошибка при обращении к программе обнаружения устройств в локальной сети
X502_ERR_SVC_RESOLVE_TIMEOUT	-39	Превышен таймаут запроса параметров автообнаруженного сетевого устройства
X502_ERR_INSTANCE_NAME_ENCODING	-40	Ошибка в кодировке имени экземпляра устройства
X502_ERR_INSTANCE_MISMATCH	-41	Экземпляры модулей не совпадают
X502_ERR_NOT_SUP_BY_FIRMWARE	-42	Возможность не поддерживается текущей версией прошивки устройства
X502_ERR_NOT_SUP_BY_DRIVER	-43	Возможность не поддерживается текущей версией драйвера устройства
X502_ERR_OUT_CYCLE_SETUP_TOUT	-44	Превышено время ожидания установления циклического сигнала на вывод
X502_ERR_UNKNOWN_FEATURE_CODE	-45	Неизвестный код поддерживаемой возможности
X502_ERR_INVALID_LTABLE_SIZE	-102	Задан неверный размер логической таблицы
X502_ERR_INVALID_LCH_NUMBER	-103	Задан неверный номер логического канала
X502_ERR_INVALID_LCH_RANGE	-104	Неверно задано значение диапазона АЦП
X502_ERR_INVALID_LCH_MODE	-105	Неверно задан режим измерения для логического канала
X502_ERR_INVALID_LCH_PHY_NUMBER	-106	Неверно задан номер физического канала при настройке логического

X502_ERR_INVALID_LCH_AVG_SIZE	-107	Неверно задан размер усреднения для логического канала
X502_ERR_INVALID_ADC_FREQ_DIV	-108	Неверно задан делитель частоты сбора данных АЦП
X502_ERR_INVALID_DIN_FREQ_DIV	-109	Неверно задан делитель частоты синхронного ввода цифровых линий
X502_ERR_INVALID_MODE	-110	Неверно задан режим работы модуля
X502_ERR_INVALID_DAC_CHANNEL	-111	Неверный номер канала ЦАП
X502_ERR_INVALID_REF_FREQ	-112	Неверный код выбора опорной частоты синхронизации
X502_ERR_INVALID_INTERFRAME_DELAY	-113	Неверно задано значение межкадровой задержки
X502_ERR_INVALID_SYNC_MODE	-114	Неверно задан режим синхронизации
X502_ERR_INVALID_STREAM_CH	-115	Неверно задан номер потока данных
X502_ERR_INVALID_OUT_FREQ_DIV	-116	Неверно задан делитель частоты синхронного вывода
X502_ERR_REF_FREQ_NOT_LOCKED	-131	Ошибка захвата опорной частоты синхронизации
X502_ERR_IOCTL_FAILD	-132	Управляющий запрос к драйверу завершен с ошибкой
X502_ERR_IOCTL_TIMEOUT	-133	Истек таймаут ожидания завершения выполнения управляющего запроса к драйверу
X502_ERR_GET_INFO	-134	Ошибка получения информации о устройстве от драйвера
X502_ERR_DIG_IN_NOT_RDY	-135	За время ожидания не было считано новое слово с цифровых линий
X502_ERR_RECV_INSUFFICIENT_WORDS	-136	Принято недостаточно слов от модуля
X502_ERR_DAC_NOT_PRESENT	-137	Попытка выполнить операцию, требующую наличие ЦАП, при его отсутствии
X502_ERR_SEND_INSUFFICIENT_WORDS	-138	Передано недостаточно слов в модуль
X502_ERR_NO_CMD_RESPONSE	-139	Не пришло ответа на переданную команду
X502_ERR_PROC_INVALID_CH_NUM	-140	Неверный номер канала в обрабатываемом потоке синхронного ввода
X502_ERR_PROC_INVALID_CH_RANGE	-141	Неверный код диапазона в обрабатываемом потоке синхронного ввода

X502_ERR_FLASH_INVALID_ADDR	-142	Задан неверный адрес во Flash-памяти
X502_ERR_FLASH_INVALID_SIZE	-143	Задан неверный размер блока данных при работе с Flash-памятью
X502_ERR_FLASH_WRITE_TOUT	-144	Истек таймаут ожидания завершения записи во Flash-память
X502_ERR_FLASH_ERASE_TOUT	-145	Истек таймаут ожидания завершения стирания блока Flash-памяти
X502_ERR_FLASH_SECTOR_BOUNDARY	-146	Заданная область для стирания Flash-памяти нарушает границу блока в 4 Кбайт
X502_ERR_SOCKET_OPEN	-147	Не удалось открыть сокет для соединения
X502_ERR_CONNECTION_TOUT	-148	Превышено время подключения
X502_ERR_CONNECTION_CLOSED_BY_DEV	-149	Соединение закрыто другой устройством
X502_ERR_SOCKET_SET_BUF_SIZE	-150	Не удалось установить заданный размер буфера сокета
X502_ERR_NO_DATA_CONNECTION	-151	Соединение для передачи данных не установлено
X502_ERR_NO_STREAM_END_MSG	-152	Не удалось дождаться сообщения о завершении потока
X502_ERR_CONNECTION_RESET	-153	Соединение было сброшено другой стороной
X502_ERR_HOST_UNREACHABLE	-154	Не удалось найти хост с указанным адресом
X502_ERR_TCP_CONNECTION_ERROR	-155	Ошибка установления TCP-соединения
X502_ERR_LDR_FILE_OPEN	-180	Не удалось открыть файл прошивки BlackFin
X502_ERR_LDR_FILE_READ	-181	Ошибка чтения из фала прошивки BlackFin
X502_ERR_LDR_FILE_FORMAT	-182	Неверный формат файла прошивки BlackFin
X502_ERR_LDR_FILE_UNSUP_FEATURE	-183	Используются возможность LDR-файла, недоступные при записи прошивки BlackFin по HDMA
X502_ERR_LDR_FILE_UNSUP_STARTUP_ADDR	-184	Неверный стартовый адрес программы в прошивке BlackFin
X502_ERR_BF_REQ_TIMEOUT	-185	Истек таймаут выполнения запроса на чтения/запись памяти BlackFin

X502_ERR_BF_CMD_IN_PROGRESS	-186	Команда для BlackFin все еще находится в процессе обработки
X502_ERR_BF_CMD_TIMEOUT	-187	Истекло время выполнения управляющей команды процессором BlackFin
X502_ERR_BF_CMD_RETURN_INSUF_DATA	-188	Возвращено недостаточно данных в ответ на команду к BlackFin
X502_ERR_BF_LOAD_RDY_TOUT	-189	Истек таймаут ожидания готовности процессора BlackFin к записи прошивки
X502_ERR_BF_NOT_PRESENT	-190	Попытка выполнить операцию для которой нужен сигнальный процессор при отсутствии сигнального процессора в модуле
X502_ERR_BF_INVALID_ADDR	-191	Неверный адрес памяти BlackFin при записи или чтении по HDMA
X502_ERR_BF_INVALID_CMD_DATA_SIZE	-192	Неверный размер данных, передаваемых с управляющей командой в BlackFin

4.1.4 Интерфейс соединения с модулем

Тип: t_x502_iface		
Описание: Интерфейс соединения с модулем		
Константа	**Значение**	**Описание**
X502_IFACE_UNKNOWN	0	Неизвестный интерфейс
X502_IFACE_USB	1	Устройство подключено по USB
X502_IFACE_ETH	2	Устройство подключено по Ethernet через TCP/IP
X502_IFACE_PCI	3	Устройство подключено по PCI/PCIe

4.1.5 Флаги, управляющие поиском присутствующих модулей

Тип: t_x502_getdevs_flags		
Описание: Флаги, управляющие поиском присутствующих модулей		
Константа	**Значение**	**Описание**
X502_GETDEVS_FLAGS_ONLY_NOT_OPENED	1	Признак, что нужно вернуть серийные номера только тех устройств, которые еще не открыты

4.1.6 Флаги для управления цифровыми выходами

Тип: t_x502_digout_word_flags		
Описание: Флаги управления цифровыми выходами. Могут быть объединены через логическое “ИЛИ” со значениями цифровых выходов при асинхронном выводе с помощью X502_AsyncOutDig() или переданы в X502_PrepareData() при синхронном выводе.		
Константа	**Значение**	**Описание**
X502_DIGOUT_WORD_DIS_H	0x00020000	Запрещение (перевод в третье состояние) старшей половины цифровых выходов
X502_DIGOUT_WORD_DIS_L	0x00010000	Запрещение младшей половины цифровых выходов
E16_DIGOUT_WORD_DIS	0x00030000	Для модуля E16 поддерживается только полное запрещение цифровых выходов

4.1.7 Константы для выбора опорной частоты

Тип: t_x502_ref_freq		
Описание: Константы для выбора опорной частоты		
Константа	**Значение**	**Описание**
E16_REF_FREQ_48000KHZ	48000000	Частота 48МГц для E16
X502_REF_FREQ_2000KHZ	2000000	Частота 2МГц
X502_REF_FREQ_1500KHZ	1500000	Частота 1.5МГц

4.1.8 поля io_mode в регистре io_hard

Тип: t_x502_io_mode		
Описание: поля io_mode в регистре io_hard		
Константа	**Значение**	**Описание**
X502_MODE_REF_FREQ_2000	0	Частота 2МГц
X502_MODE_REF_FREQ_1500	2	Частота 1.5МГц
E16_MODE_REF_FREQ_48000	3	Частота 48МГц для E16

4.1.9 Диапазоны измерения для канала АЦП

Тип: t_x502_adc_range		
Описание: Диапазоны измерения для канала АЦП		
Константа	**Значение**	**Описание**
X502_ADC_RANGE_10	0	Диапазон +/-10V
X502_ADC_RANGE_5	1	Диапазон +/-5V
X502_ADC_RANGE_2	2	Диапазон +/-2V
X502_ADC_RANGE_1	3	Диапазон +/-1V
X502_ADC_RANGE_05	4	Диапазон +/-0.5V
X502_ADC_RANGE_02	5	Диапазон +/-0.2V

4.1.10 Диапазоны измерения для канала АЦП E16

Тип: t_e16_adc_range		
Описание: Диапазоны измерения для канала АЦП E16		
Константа	**Значение**	**Описание**
E16_ADC_RANGE_10000	0	Диапазон +/-10V
E16_ADC_RANGE_2500	1	Диапазон +/-2.5V
E16_ADC_RANGE_625	2	Диапазон +/-0.625V
E16_ADC_RANGE_156	3	Диапазон +/-0.156V

4.1.11 Режим измерения для логического канала

Тип: t_x502_lch_mode		
Описание: Режим измерения для логического канала		
Константа	**Значение**	**Описание**
X502_LCH_MODE_COMM	0	Измерение напряжения относительно общей земли
X502_LCH_MODE_DIFF	1	Дифференциальное измерение напряжения
X502_LCH_MODE_ZERO	2	Измерение собственного нуля

4.1.12 Режимы синхронизации

Тип: t_x502_sync_mode		
Описание: Режимы задания источника частоты синхронизации и признака начала синхронного ввода-вывода		
Константа	**Значение**	**Описание**
X502_SYNC_INTERNAL	0	Внутренний сигнал
X502_SYNC_EXTERNAL_MASTER	1	От внешнего мастера по разъему межмодульной синхронизации
X502_SYNC_DI_SYN1_RISE	2	По фронту сигнала DI_SYN1
X502_SYNC_DI_SYN1_FALL	3	По спаду сигнала DI_SYN1
X502_SYNC_DI_SYN2_RISE	6	По фронту сигнала DI_SYN2
X502_SYNC_DI_SYN2_FALL	7	По спаду сигнала DI_SYN2
E16_SYNC_TRIG_RISE	2	Только для E16! Важно правильно установить направление TRIG или INT на вход снятием флага E16_MODE_TRIG_OUTPUT или E16_MODE_INT_OUTPUTПо фронту сигнала TRIG
E16_SYNC_TRIG_FALL	3	По спаду сигнала TRIG
E16_SYNC_INT_RISE	6	По фронту сигнала INT
E16_SYNC_INT_FALL	7	По спаду сигнала INT
E16_SYNC_ADC_ABOVE_LEVEL	8	Аналоговая синхронизация старта, только для E16 Выше уровня
E16_SYNC_ADC_BELOW_LEVEL	9	Ниже уровня
E16_SYNC_ADC_EDGE_RISE	10	По переходу сигнала снизу вверх
E16_SYNC_ADC_EDGE_FALL	11	По переходу сигнала сверху вниз

4.1.13 Флаги, управляющие обработкой принятых данных

Тип: t_x502_proc_flags		
Описание: Флаги, управляющие обработкой принятых данных		
Константа	**Значение**	**Описание**
X502_PROC_FLAGS_VOLT	0x00000001	Признак, что нужно преобразовать значения АЦП в вольты
X502_PROC_FLAGS_DONT_CHECK_CH	0x00010000	Признак, что не нужно проверять совпадение номеров каналов в принятых данных с каналами из логической таблицы. Может использоваться при нестандартной прошивке BlackFin при передаче в ПК не всех данных.

4.1.14 Флаги для обозначения синхронных потоков данных

Тип: t_x502_streams		
Описание: Флаги для обозначения синхронных потоков данных		
Константа	**Значение**	**Описание**
X502_STREAM_ADC	0x01	Поток данных от АЦП
X502_STREAM_DIN	0x02	Поток данных с цифровых входов
X502_STREAM_DAC1	0x10	Поток данных первого канала ЦАП
X502_STREAM_DAC2	0x20	Поток данных второго канала ЦАП
X502_STREAM_DOUT	0x40	Поток данных на цифровые выводы
X502_STREAM_ALL_IN	X502_STREAM_ADC | X502_STREAM_DIN	Объединение всех флагов, обозначающих потоки данных на ввод
X502_STREAM_ALL_OUT	X502_STREAM_DAC1 | X502_STREAM_DAC2 | X502_STREAM_DOUT	Объединение всех флагов, обозначающих потоки данных на вывод

4.1.15 Константы, определяющие тип передаваемого отсчета из ПК в модуль

Тип: t_x502_stream_out_wrd_type		
Описание: Константы, определяющие тип передаваемого отсчета из ПК в модуль		
Константа	**Значение**	**Описание**
X502_STREAM_OUT_WORD_TYPE_DOUT	0x0	Цифровой вывод
X502_STREAM_OUT_WORD_TYPE_DAC1	0x40000000	Код для 1-го канала ЦАП
X502_STREAM_OUT_WORD_TYPE_DAC2	0x80000000	Код для 2-го канала ЦАП

4.1.16 Режим работы модуля

Тип: t_x502_mode		
Описание: Режим работы модуля		
Константа	**Значение**	**Описание**
X502_MODE_FPGA	0	Все потоки данных передаются через ПЛИС минуя сигнальный процессор BlackFin
X502_MODE_DSP	1	Все потоки данных передаются через сигнальный процессор, который должен быть загружен прошивкой для обработки этих потоков
X502_MODE_DEBUG	2	Отладочный режим

4.1.17 Номера каналов ЦАП

Тип: t_x502_dac_ch		
Описание: Номер каналов ЦАП для указания в X502_AsyncOutDac()		
Константа	**Значение**	**Описание**
X502_DAC_CH1	0	Первый канал ЦАП
X502_DAC_CH2	1	Второй канал ЦАП

4.1.18 Флаги, используемые при выводе данных на ЦАП

Тип: t_x502_dacout_flags		
Описание: Флаги, комбинацию которых можно передать в X502_AsyncOutDac() или X502_PrepareData(), чтобы определить действия, которые должны выполнить эти функции с переданным значением перед выводом их на ЦАП		
Константа	**Значение**	**Описание**
X502_DAC_FLAGS_VOLT	0x0001	Указывает, что значение задано в Вольтах и при выводе его нужно перевести его в коды ЦАП. Если флаг не указан, то считается, что значение изначально в кодах
X502_DAC_FLAGS_CALIBR	0x0002	Указывает, что нужно применить калибровочные коэффициенты перед выводом значения на ЦАП.

4.1.19 Номера каналов для передачи потоков данных

Тип: t_x502_stream_ch		
Описание: Номера каналов для передачи потоков данных		
Константа	**Значение**	**Описание**
X502_STREAM_CH_IN	0	Общий канал на ввод
X502_STREAM_CH_OUT	1	Общий канал на вывод

4.1.20 Цифровые линии, на которых можно включить подтягивающие резисторы

Тип: t_x502_pullups		
Описание: Флаги, определяющие на каких цифровых входах должны быть включены подтягивающие резисторы. Для разных модулей доступны разные наборы флагов.		
Константа	**Значение**	**Описание**
X502_PULLUPS_DI_H	0x01	Старшая половина цифровых входов (только для L502)
X502_PULLUPS_DI_L	0x02	Младшая половина цифровых входов (только для L502)
X502_PULLUPS_DI_SYN1	0x04	Линия SYN1
X502_PULLUPS_DI_SYN2	0x08	Линия SYN2
X502_PULLDOWN_CONV_IN	0x10	Подтяжка к 0 линии межмодульной синхронизации CONV_IN (только для E502)
X502_PULLDOWN_START_IN	0x20	Подтяжка к 0 линии межмодульной синхронизации START_IN (только для E502)
E16_MODE_TRIG_OUTPUT	0x04	E16: Переключение TRIG на выход, иначе вход
E16_MODE_INT_OUTPUT	0x08	E16: Переключение INT на выход, иначе вход
E16_MODE_TRIG_START_OUTPUT	0x10	E16: Сигнал START на выходе TRIG, иначе CONV (TRIG должен быть включен на выход)
E16_MODE_INT_START_OUTPUT	0x20	E16: Сигнал START на выходе INT, иначе CONV (INT должен быть включен на выход)
E16_MODE_RELAY_ON	0x40	E16: Включить реле

4.1.21 Флаги, определяющие наличие опций в модуле и наличие необходимых параметров

Тип: t_x502_dev_flags		
Описание: Флаги, определяющие наличие опций в модуле и наличие необходимых параметров		
Константа	**Значение**	**Описание**
X502_DEVFLAGS_DAC_PRESENT	0x00000001	Признак наличия двухканального канального ЦАП
X502_DEVFLAGS_GAL_PRESENT	0x00000002	Признак наличия гальваноразвязки
X502_DEVFLAGS_BF_PRESENT	0x00000004	Признак наличия сигнального процессора BlackFin

X502_DEVFLAGS_PROC_TYPE	0x00000038	3 бита, отвечающие за тип используемого процессора/контроллера (0 — ADSP-BF523)
X502_DEVFLAGS_DAC_TYPE	0x000000C0	2 бита, отвечающие за тип используемого ЦАП (0 — AD5542AARUZ, 1 — DAC8581IPW)
X502_DEVFLAGS_IFACE_SUPPORT_USB	0x00000100	Признак, что устройство поддерживает интерфейс USB
X502_DEVFLAGS_IFACE_SUPPORT_ETH	0x00000200	Признак, что устройство поддерживает Ethernet
X502_DEVFLAGS_IFACE_SUPPORT_PCI	0x00000400	Признак, что устройство поддерживает интерфейс PCI/PCI-Express
X502_DEVFLAGS_INDUSTRIAL	0x00008000	Признак, что устройство выполнено в индустриалном исполнении
X502_DEVFLAGS_FLASH_DATA_VALID	0x00010000	Признак, что во Flash-памяти присутствует информация о модуле
X502_DEVFLAGS_FLASH_ADC_CALIBR_VALID	0x00020000	Признак, что во Flash-памяти присутствуют действительные калибровочные коэффициенты АЦП
X502_DEVFLAGS_FLASH_DAC_CALIBR_VALID	0x00040000	Признак, что во Flash-памяти присутствуют действительные калибровочные коэффициенты ЦАП
X502_DEVFLAGS_FPGA_LOADED	0x00800000	Признак, что присутствует прошивка ПЛИС и она успешно была загружена
X502_DEVFLAGS_DEVREC_OPENED	0x01000000	Признак, что устройство уже открыто (действителен только внутри t_x502_devrec)
X502_DEVFLAGS_FPGA_AWAITING_GD32	0x02000000	Признак, что присутствует прошивка ПЛИС и она успешно была загружена ожидаем GD32 для того чтобы остальные регистры работали
X502_DEVFLAGS_GD_PRESENT	0x02000000	Признак, что присутствует GD32 и это E502-P1

4.1.22 Тип содержимого строки с расположением устройства

Тип: t_x502_location_type		
Описание: Данное поле определяет содержимое поля location в структуре t_x502_devrec		
Константа	**Значение**	**Описание**
X502_LOCATION_TYPE_NONE	0	В поле расположения устройства не содержится информации
X502_LOCATION_TYPE_ADDR	1	В поле расположения устройства содержится строка с адресом устройства
X502_LOCATION_TYPE_INSTANCE_NAME	2	В поле расположения устройства содержится строка с именем экземпляра

4.1.23 Флаги для режима циклического вывода

Тип: t_x502_out_cycle_flags		
Описание: Данные флаги могут быть переданы в X502_OutCycleSetup() и X502_OutCycleStop()		
Константа	**Значение**	**Описание**
X502_OUT_CYCLE_FLAGS_FORCE	0x01	Флаг указывает, что останов или смена сигнала могут произойти без ожидания конца цикла предыдущего сигнала. Это позволяет выполнить переключение быстрее (однако все равно может быть поставлено на передачу до 256 КСемплов, которые должны будут быть переданы), но точка смены или останова может быть в любом месте периода
X502_OUT_CYCLE_FLAGS_WAIT_DONE	0x02	Флаг указывает, что функция должна дождаться полной загрузки сигнала и установки сигнала на вывод (для X502_OutCycleSetup()) или завершения генерации циклического сигнала (для X502_OutCycleStop()). Без него функции только посылают команду модулю, возвращая сразу управление. Данное ожидание может занимать значительное время в зависимости от размера сигнала (а также размера предыдущего сигнала в случае смены или останова генерации без X502_OUT_CYCLE_FLAGS_FORCE). Данную проверку можно сделать и отдельной функций X502_OutCycleCheckSetupDone(). Данный флаг имеет значения только в тех случаях, когда поддерживается функция X502_OutCycleCheckSetupDone(), в противном случае он игнорируется.

4.1.24 Дополнительные возможности модуля

Тип: t_x502_features		
Описание: Коды возможностей модуля, которые могут поддерживаться или нет в зависимости от типа модуля, версий прошивок и т.п.		
Константа	**Значение**	**Описание**
X502_FEATURE_OUT_FREQ_DIV	1	Поддержка установки делителя частоты вывода, отличного от X502_OUT_FREQ_DIV_DEFAULT
X502_FEATURE_OUT_STATUS_FLAGS	2	Возможность чтения флагов состояния вывода с помощью X502_OutGetStatusFlags()

4.1.25 Флаги состояния для синхронного вывода

Тип: t_x502_out_status_flags		
Описание: Флаги состояния для синхронного вывода		
Константа	**Значение**	**Описание**
X502_OUT_STATUS_FLAG_BUF_IS_EMPTY	0x01	Флаг указывает, что в настоящее время буфер в модуле на передачу пуст
X502_OUT_STATUS_FLAG_BUF_WAS_EMPTY	0x02	Флаг указывает, что было опустошение буфера на вывод с начала старта синхронного ввода-вывода или с момента последнего чтения статуса с помощью X502_OutGetStatusFlags() (в зависимости от того, что было последним)

4.2 Типы данных

4.2.1 Запись о устройстве

Тип: t_x502_devrec		
Описание: Структура, описывающая устройство, по которой с ним можно установить соединение		
Поле	**Тип**	**Описание поля**
sign	uint32_t	Признак действительной структуры. Если запись действительна (соответствует какому-либо устройству), то должен быть равен X502_DEVREC_SIGN)
devname	char [X502_DEVNAME_SIZE]	Название устройства
serial	char [X502_SERIAL_SIZE]	Серийный номер
location	char [X502_LOCATION_STR_SIZE]	Описание подключения (если есть)

flags	uint32_t	Флаги из t_x502_dev_flags, описывающие устройство
iface	uint8_t	Интерфейс, по которому подключено устройство
location_type	uint8_t	Определяет, что именно сохранено в поле location (одно значение из t_x502_location_type)
res	char [122]	Резерв
internal	t_x502_devrec_inptr *	Непрозрачный указатель на структуру с дополнительной информацией, необходимой для открытия устройства

4.2.2 Калибровочные коэффициенты диапазона.

Тип: t_x502_cbr_coef		
Описание: Структура содержит калибровочные значения смещения нуля и коэффициента шкалы для одного диапазона АЦП или ЦАП.Результирующее значение АЦП вычисляется как (val - offs) * k, где val - неоткалиброванное значение		
Поле	**Тип**	**Описание поля**
offs	double	смещение нуля
k	double	коэффициент шкалы

4.2.3 Калибровочные коэффициенты модуля

Тип: t_x502_cbr		
Описание: Структура, содержащая все калибровочные коэффициенты, которые используются модулем L-502/E-502		
Поле	**Тип**	**Описание поля**
adc	t_x502_cbr_coef [X502_ADC_RANGE_CNT]	Калибровочные коэффициенты АЦП
res1	uint32_t [64]	Резерв
dac	t_x502_cbr_coef [X502_DAC_CH_CNT]	Калибровочные коэффициенты ЦАП
res2	uint32_t [20]	Резерв

4.2.4 Информация о модуле L-502/E-502.

Тип: t_x502_info		
Описание: Структура, содержащая постоянную информация о модуле L-502/E-502, которая как правило не изменяется после открытия		
Поле	**Тип**	**Описание поля**
name	char [X502_DEVNAME_SIZE]	Название устройства (“L502” или “E502” или “E16”)
serial	char [X502_SERIAL_SIZE]	Серийный номер

devflags	uint32_t	Флаги из t_x502_dev_flags, описывающие наличие в модуле определенных опций
fpga_ver	uint16_t	Версия ПЛИС (старший байт - мажорная, младший - минорная)
plda_ver	uint8_t	Версия ПЛИС, управляющего аналоговой частью
board_rev	uint8_t	Ревизия платы
mcu_firmware_ver	uint32_t	Версия прошивки контроллера Cortex-M4. Действительна только для модуля E-502
flash_size	uint32_t	Размер внешней flash-памяти в байтах
factory_mac	uint8_t [X502_MAC_ADDR_SIZE]	Заводской MAC-адрес — действителен только для устройств с Ethernet-интерфейсом
res	uint8_t [106]	Резерв
cbr	t_x502_cbr	Заводские калибровочные коэффициенты (из Flash-памяти)

4.2.5 Описатель конфигурации сетевого интерфейса

Тип: t_e502_eth_config_hnd
Описание: Непрозрачный указатель на структуру, содержащую параметры конфигурации сетевого интерфейса модуля E-502. Пользовательской программе не доступны поля структуры напрямую, а только через функции библиотеки. Описатель конфигурации создается с помощью E502_EthConfigCreate() и в конце работы освобождается с помощью E502_EthConfigFree(). Как правило все настройки не должны заполняться пользователем вручную, обычно сперва они считываются из устройства с помощью E502_EthConfigRead(), после чего часть настроек можно изменить и сохранить в модуль через E502_EthConfigWrite()

4.2.6 Описатель контекста поиска устройств в сети

Тип: t_e502_eth_svc_browse_hnd
Описание: Указатель на непрозрачную структуру с информацией о состоянии текущего сеанса поиска устройств в сети. Создается при начале поиска вызовом E502_EthSvcBrowseStart() и уничтожается с помощью E502_EthSvcBrowseStop()

4.2.7 Описатель сетевого сервиса

Тип: t_e502_eth_svc_record_hnd
Описание: Указатель на непрозрачную структуру с информацией о сервисе в сети, соответствующем одному модулю E-502. Используется при автоматическом обнаружении устройств в локальной сети. Создается при вызове E502_EthSvcBrowseGetEvent() и уничтожается с помощью E502_EthSvcRecordFree()

4.2.8 Внутренняя информация записи о устройстве

Тип: t_x502_devrec_inptr
Описание: Непрозрачная структура с информацией, достаточной для установления с ним связи. Зависит от типа устройства, интерфейса подключения и не доступна пользователю напрямую, а используется библиотекой в X502_OpenByDevRecord()

4.2.9 Описатель модуля

Тип: t_x502_hnd
Описание: Непрозрачный указатель на структуру, содержащую информацию о настройках модуля и текущем соединении с ним. Пользовательской программе не доступны поля структуры напрямую, а только через функции библиотеки. Функции управления модулем принимают описатель модуля своим первым параметром. Описатель модуля создается с помощью X502_Create() и в конце работы освобождается с помощью X502_Free().

4.2.10 Список серийный номеров

Тип: t_x502_serial_list
Описание: Тип определяет массив серийных номеров для количества модулей, определяемого на этапе работы программы.

4.3 Функции

4.3.1 Функции для создания и освобождения описателя модуля

4.3.1.1 Создание описателя модуля

Формат: t_x502_hnd X502_Create (void)
Описание: Создание описателя модуля, для последующей работы с модулем E-502 или L-502. В случае успешного выделения памяти инициализирует поля описателя значениями по умолчанию.
Возвращаемое значение: NULL в случае ошибки, иначе - описатель модуля

4.3.1.2 Освобождение описателя модуля

Формат: `int32_t X502_Free (t_x502_hnd hnd)`
Описание: Освобождение памяти, выделенной под описатель модуля с помощью `X502_Create()`. После этого описатель уже использовать нельзя, независимо от возвращенного значения!
Параметры: **hnd** — Описатель устройства
Возвращаемое значение: Код ошибки

4.3.2 Функции для открытия и получения информации о модуле

4.3.2.1 Получение списка серийных номеров модулей L-502.

Формат: `int32_t L502_GetSerialList (char serials[] [X502_SERIAL_SIZE], uint32_t size, uint32_t flags, uint32_t *devcnt)`
Описание: Функция возвращает список номеров всех найденных модулей L-502, независимо от того, открыты они сейчас или нет. Если нужен список только тех модулей, которые не открыты (то есть только тех, с которыми можно установить соединение), то для этого можно передать в функцию флаг `X502_GETDEVS_FLAGS_ONLY_NOT_OPENED`.
Параметры: **serials** — Массив размером size*`X502_SERIAL_SIZE` байт, в который будут сохранены серийные номера найденных модулей. Может быть NULL, если size=0, а devcnt!=NULL, в случае, если нужно только получить количество модулей в системе. **size** — Определяет, сколько максимально серийных номеров может быть сохранено в массив serial. Будут сохранены только первые size серийных номеров. Может быть 0, если serials=NULL **flags** — Флаги из `t_x502_getdevs_flags`, определяющие поведение функции. **devcnt** — Если devcnt!=NULL, то в данную переменную сохраняется общее число найденных модулей L502 (может быть больше size).
Возвращаемое значение: Если <0 - код ошибки, иначе количество сохраненных серийных номеров в массиве serials (всегда <= size)

4.3.2.2 Открытие модуля L-502 по его серийному номеру

Формат: `int32_t L502_Open (t_x502_hnd hnd, const char *serial)`

Описание:

Функция устанавливает связь с модулем L-502 по его серийному номеру. После успешного выполнения этой функции, пользователь получает эксклюзивный доступ к модулю через описатель модуля. До закрытия связи с помощью `X502_Close()` никто другой установить связь с модулем не сможет (будет возвращена ошибка `X502_ERR_DEVICE_ACCESS_DENIED`).

Если в качестве серийного номера передан NULL или пустая строка, то будет установлена связь с первым найденным модулем, с которым получится успешно ее установить. Если в системе нет ни одного модуля, то будет возвращена ошибка `X502_ERR_DEVICE_NOT_FOUND`. Если в системе присутствуют модули L-502, но соединение ни с одним из них установить не удалось, то будет возвращена ошибка, полученная при попытке установить соединение с последним найденным модулем.

После завершения работы с устройством соединение должно быть закрыто с помощью `X502_Close()`.

Параметры:

hnd — Описатель устройства.

serial — Указатель на строку с серийным номером открываемого модуля или NULL.

Возвращаемое значение:

Код ошибки.

4.3.2.3 Получение списка серийных номеров модулей E-502, подключенных по USB.

Формат: `int32_t E502_UsbGetSerialList (char serials[][X502_SERIAL_SIZE], uint32_t size, uint32_t flags, uint32_t *devcnt)`

Описание:

Функция возвращает список номеров всех найденных модулей E-502, независимо от того, открыты они сейчас или нет.

Функция на данный момент не поддерживает флаг `X502_GETDEVS_FLAGS_ONLY_NOT_OPENED`.

Параметры:

serials — Массив размером size*`X502_SERIAL_SIZE` байт, в который будут сохранены серийные номера найденных модулей. Может быть NULL, если size=0, а devcnt!=NULL, в случае, если нужно только получить количество модулей в системе.

size — Определяет, сколько максимально серийных номеров может быть сохранено в массив serial. Будут сохранены только первые size серийных номеров. Может быть 0, если serials=NULL

flags — Флаги из `t_x502_getdevs_flags`, определяющие поведение функции.

devcnt — Если devcnt!=NULL, то в данную переменную сохраняется общее число найденных модулей E502 (может быть больше size).

Возвращаемое значение:

Если <0 - код ошибки, иначе количество сохраненных серийных номеров в массиве serials (всегда <= size)

4.3.2.4 Работает аналогично E502_UsbGetSerialList, только для модулей E16.

Формат: `int32_t E16_UsbGetSerialList (char serials[][X502_SERIAL_SIZE], uint32_t size, uint32_t flags, uint32_t *devcnt)`

Описание:

4.3.2.5 Открытие модуля E-502, подключенного по USB, по его серийному номеру

Формат: `int32_t E502_OpenUsb (t_x502_hnd hnd, const char *serial)`

Описание:

Функция устанавливает связь с модулем E-502, подключенным по интерфейсу USB, по его серийному номеру.

После успешного выполнения этой функции, пользователь получает эксклюзивный доступ к модулю через описатель модуля. До закрытия связи с помощью `X502_Close()` никто другой установить связь с модулем не сможет (будет возвращена ошибка `X502_ERR_DEVICE_ACCESS_DENIED`).

Если в качестве серийного номера передан NULL или пустая строка, то будет установлена связь с первым найденным модулем, с которым получится успешно ее установить. Если в системе нет ни одного модуля, то будет возвращена ошибка `X502_ERR_DEVICE_NOT_FOUND`. Если в системе присутствуют модули E-502, но соединение ни с одним из них установить не удалось, то будет возвращена ошибка, полученная при попытке установить соединение с последним найденным модулем.

После завершения работы с устройством соединение должно быть закрыто с помощью `X502_Close()`.

Параметры:

hnd — Описатель устройства.

serial — Указатель на строку с серийным номером открываемого модуля или NULL.

Возвращаемое значение:

Код ошибки.

4.3.2.6 Работает аналогично E502_OpenUsb, только для модулей E16.

Формат: `int32_t E16_OpenUsb (t_x502_hnd hnd, const char *serial)`

Описание:

4.3.2.7 Открытие модуля E-502 по IP-адресу

Формат: `int32_t E502_OpenByIpAddr (t_x502_hnd hnd, uint32_t ip_addr, uint32_t flags, uint32_t tout)`

Описание:

Функция устанавливает связь с модулем E-502, подключенным по интерфейсу Ethernet, для которого установлен указанный адрес IPv4.

После завершения работы с устройством соединение должно быть закрыто с помощью `X502_Close()`.

Параметры:

hnd — Описатель устройства.

ip_addr — IPv4 адрес модуля в виде 32-битного слова. Для адреса "a.b.c.d" ip_addr = (a<<24)|(b<<16)|(c<<8)|d.

flags — Флаги, управляющие работой функции. Резерв, должны быть всегда 0.

tout — Время на установления подключения в мс. Если подключение не удастся завершить за заданное время, то функция вернет ошибку.

Возвращаемое значение:

Код ошибки.

4.3.2.8 Открытие модуля E-16 по IP-адресу

Формат: `int32_t E16_OpenByIpAddr (t_x502_hnd hnd, uint32_t ip_addr, uint32_t flags, uint32_t tout)`

Описание:

Функция устанавливает связь с модулем E-16, подключенным по интерфейсу Ethernet, для которого установлен указанный адрес IPv4.

После завершения работы с устройством соединение должно быть закрыто с помощью `X502_Close()`.

Параметры:

hnd — Описатель устройства.

ip_addr — IPv4 адрес модуля в виде 32-битного слова. Для адреса "a.b.c.d" ip_addr = (a<<24)|(b<<16)|(c<<8)|d.

flags — Флаги, управляющие работой функции. Резерв, должны быть всегда 0.

tout — Время на установления подключения в мс. Если подключение не удастся завершить за заданное время, то функция вернет ошибку.

Возвращаемое значение:

Код ошибки.

4.3.2.9 Закрытие соединения с модулем

Формат: `int32_t X502_Close (t_x502_hnd hnd)`
Описание: Функция разрывает соединение с модулем E-502/L-502, если оно было ранее установлено (в противном случае ничего не делает). Описатель модуля не освобождается. Память под описатель модуля должна быть освобождена вызовом `X502_Free()`.
Параметры: **hnd** — Описатель модуля.
Возвращаемое значение: Код ошибки.

4.3.2.10 Проверка, открыто ли соединение с модулем

Формат: `int32_t X502_IsOpened (t_x502_hnd hnd)`
Описание: Функция проверяет, открыто ли в данный момент соединение с модулем. Если соединение открыто, то функция возвращает `X502_ERR_OK`. Если же закрыто, то функция возвращает ошибку `X502_ERR_DEVICE_NOT_OPENED`.
Параметры: **hnd** — Описатель модуля.
Возвращаемое значение: Код ошибки.

4.3.2.11 Получение информации о модуле

Формат: `int32_t X502_GetDevInfo (t_x502_hnd hnd, t_x502_info *info)`
Описание: Получение информации о модуле L-502/E-502, с которым установлена связь.
Параметры: **hnd** — Описатель модуля. **info** — Информация о модуле (смотри описание типа t_x502_info).
Возвращаемое значение: Код ошибки.

4.3.3 Функции для работы с записями об устройстве

4.3.3.1 Получить список записей, соответствующих подключенным модулям L502.

Формат: `int32_t L502_GetDevRecordsList (t_x502_devrec *list, uint32_t size, uint32_t flags, uint32_t *devcnt)`

Описание:

Функция находит все подключенные модули L-502 и инициализирует записи о каждом найденном устройстве и сохраняет их в переданный список (если не нулевой). Возвращенные в списке записи должны быть очищены после использования с помощью `X502_FreeDevRecordList()` (также в случае повторного вызов `L502_GetDevRecordsList()` с тем же массивом записей, записи, полученные при предыдущем вызове, должны быть сперва очищены).

Параметры:

list — Массив для сохранения записей о найденных устройствах. Должен содержать место для сохранения не менее size записей. Может быть NULL, если size=0, а devcnt!=NULL, в случае, если нужно только получить количество модулей в системе.

size — Определяет, сколько максимально записей может быть сохранено в массив list. Будут сохранены только первые size записей, если устройств найденно больше.

flags — Флаги из `t_x502_getdevs_flags`, определяющие поведение функции.

devcnt — Если не нулевой указатель, то в данную переменную сохраняется общее число найденных модулей L-502 (может быть больше size).

Возвращаемое значение:

Если <0 — код ошибки, иначе количество сохраненных записей о найденных устройствах (всегда <= size). Именно на этот размер нужно сделать в дальнейшем `X502_FreeDevRecordList()` для освобождения памяти, выделенной под информацию, на которую ссылается запись.

4.3.3.2 Получить список записей, соответствующих подключенным модулям E502.

Формат: `int32_t E502_UsbGetDevRecordsList (t_x502_devrec *list, uint32_t size, uint32_t flags, uint32_t *devcnt)`

Описание:

Функция находит все подключенные по интерфейсу USB модули E-502 и инициализирует записи о каждом найденном устройстве и сохраняет их в переданный список (если не нулевой). Возвращенные в списке записи должны быть очищены после использования с помощью `X502_FreeDevRecordList()` (также в случае повторного вызов `E502_UsbGetDevRecordsList()` с тем же массивом записей, записи, полученные при предыдущем вызове, должны быть сперва очищены).

Параметры:

list — Массив для сохранения записей о найденных устройствах. Должен содержать место для сохранения не менее size записей. Может быть NULL, если size=0, а devcnt!=NULL, в случае, если нужно только получить количество модулей в системе.

size — Определяет, сколько максимально записей может быть сохранено в массив list. Будут сохранены только первые size записей, если устройств найдено больше.

flags — Флаги из `t_x502_getdevs_flags`, определяющие поведение функции.

devcnt — Если не нулевой указатель, то в данную переменную сохраняется общее число найденных модулей E-502, подключенных по интерфейсу USB (может быть больше size).

Возвращаемое значение:

Если <0 — код ошибки, иначе количество сохраненных записей о найденных устройствах (всегда <= size). Именно на этот размер нужно сделать в дальнейшем `X502_FreeDevRecordList()` для освобождения памяти, выделенной под информацию, на которую ссылается запись.

4.3.3.3 Аналогично E502_UsbGetDevRecordsList, получить список записей, соответствующих подключенным модулям E16.

Формат: `int32_t E16_UsbGetDevRecordsList (t_x502_devrec *list, uint32_t size, uint32_t flags, uint32_t *devcnt)`

Описание:

4.3.3.4 Аналогично E502_UsbGetDevRecordsList, получить список записей, соответствующих подключенным модулям E14-440.

Формат: `int32_t E440_UsbGetDevRecordsList (t_x502_devrec *list, uint32_t size, uint32_t flags, uint32_t *devcnt)`

Описание:

4.3.3.5 Получить список записей, соответствующих подключенным модулям E502.

Формат: `int32_t E502_UsbGetDevRecordsList2 (t_x502_devrec *list, uint32_t size, uint32_t flags, uint32_t *devcnt, uint16_t idVendor, uint16_t idProduct)`

Описание:

Делает тоже самое что и E502_UsbGetDevRecordsList, отличие в том что можно указать idVendor и idProduct USB устройства

Параметры:

idVendor — idVendor

idProduct — idProduct

Возвращаемое значение:

Если <0 — код ошибки, иначе количество сохраненных записей о найденных устройствах (всегда <= size). Именно на этот размер нужно сделать в дальнейшем `X502_FreeDevRecordList()` для освобождения памяти, выделенной под информацию, на которую ссылается запись.

4.3.3.6 Создание записи о устройстве с указанным IP-адресом

Формат: `int32_t E502_MakeDevRecordByIpAddr (t_x502_devrec *devrec, uint32_t ip_addr, uint32_t flags, uint32_t tout)`

Описание:

Данная функция инициализирует запись о устройстве, подключенном по интерфейсу Ethernet, с указанным IPv4 адресом. Данная функция только создает запись, но не проверяет наличие соответствующего устройства. Подключение к модулю выполняется аналогично другим записям через `X502_OpenByDevRecord()`.

Параметры:

devrec — Указатель на запись устройства, которая должна быть создана и заполнена нужными параметрами.

ip_addr — IPv4 адрес модуля в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`).

flags — Флаги. Резерв, должны быть всегда 0.

tout — Время для установления подключения в мс. Данное время сохраняется в записи и используется при последующем вызове `X502_OpenByDevRecord()`. Если подключение не удастся завершить за это время, то функция `X502_OpenByDevRecord()` вернет ошибку.

Возвращаемое значение:

Код ошибки

4.3.3.7 Создание записи о устройстве с указанным IP-адресом

Формат: `int32_t E502_MakeDevRecordByIpAddr2 (t_x502_devrec *devrec, uint32_t ip_addr, uint32_t flags, uint32_t tout, char const *devname)`

Описание:

Работает аналогично функции E502_MakeDevRecordByIpAddr. Для поддержки E16 добавлен параметр *devname.

Параметры:

devrec — Указатель на запись устройства, которая должна быть создана и заполнена нужными параметрами.

ip_addr — IPv4 адрес модуля в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`).

flags — Флаги. Резерв, должны быть всегда 0.

tout — Время для установления подключения в мс. Данное время сохраняется в записи и используется при последующем вызове `X502_OpenByDevRecord()`. Если подключение не удастся завершить за это время, то функция `X502_OpenByDevRecord()` вернет ошибку.

devname — Имя устройства “E16” или “E502”.

Возвращаемое значение:

Код ошибки

4.3.3.8 Установка TCP-порта управляющего соединения для записи о устройстве

Формат: `int32_t E502_EthDevRecordSetCmdPort (t_x502_devrec *devrec, uint16_t cmd_port)`

Описание:

Данная функция позволяет изменить TCP-порт управляющего соединения модуля E-502. Это может быть необходимо, если модуль E-502 и хост, с которого необходимо установить соединение, находятся в разных сетях и адрес модуля E-502 не доступен из сети хоста. В этом случае требуется настройка проброса портов на маршрутизаторе и при наличие более одного такого модуля E-502, т.к все соединения идут с маршрутизитором, то различить эти модули можно только по TCP-порту, если настроить разные порты при пробросе. В этом случае помимо порта управляющего соединения, необходимо изменить и порт соединения для передачи данных, вызвав `E502_EthDevRecordSetDataPort()`.

Данная функция должна быть вызвана для записи, созданной до этого с помощью `E502_MakeDevRecordByIpAddr()` и до открытия соединения с помощью `X502_OpenByDevRecord()`.

Параметры:

devrec — Указатель на запись устройства, в которой нужно изменить управляющий TCP-порт.

cmd_port — Новое значение TCP-порта для управляющего соединения

Возвращаемое значение:

Код ошибки

4.3.3.9 Установка TCP-порта соединения передачи данных для записи о устройстве

Формат: `int32_t E502_EthDevRecordSetDataPort (t_x502_devrec *devrec, uint16_t data_port)`

Описание:

Функция аналогична E502_EthDevRecordSetCmdPort(), но изменяет TCP-порт для соединения, по которому идет обмен данных потоков ввода-вывода.

Параметры:

devrec — Указатель на запись устройства, в которой нужно изменить управляющий TCP-порт.

data_port — Новое значение TCP-порта для соединения передачи данных

Возвращаемое значение:

Код ошибки

4.3.3.10 Создание записи о устройстве по описателю сетевого сервиса

Формат: `int32_t E502_MakeDevRecordByEthSvc (t_x502_devrec *devrec, t_e502_eth_svc_record_hnd svc, uint32_t flags, uint32_t tout)`

Описание:

Данная функция инициализирует запись о устройстве, подключенном по интерфейсу Ethernet, соответствующему сетевому сервису, на который указывает переданный описатель сетевого сервиса. Этот описатель может быть получен с помощью функций поиска сетевых сервисов, соответствующих модулям E-502, в локальной сети. Данная функция только создает запись, но не проверяет наличие соответствующего устройства. Подключение к модулю выполняется аналогично другим записям через X502_OpenByDevRecord(). Вся необходимая информация из описателя сетевого сервиса сохраняется в записи о устройстве, т.е. после вызова данной фунции при желании описатель сетевого сервиса можно сразу освобождать с помощью E502_EthSvcRecordFree().

Параметры:

devrec — Указатель на запись устройства, которая должна быть создана и заполнена нужными параметрами.

svc — Описатель сетевого сервиса, полученный с помощью E502_EthSvcBrowseGetEvent().

flags — Флаги. Резерв, должны быть всегда 0.

tout — Время для установления подключения в мс. Данное время сохраняется в записи и используется при последующем вызове X502_OpenByDevRecord(). Если подключение не удастся завершить за это время, то функция X502_OpenByDevRecord() вернет ошибку.

Возвращаемое значение:

Код ошибки

4.3.3.11 Открыть соединение с модулем по записи о устройстве

Формат: `int32_t X502_OpenByDevRecord (t_x502_hnd hnd, const t_x502_devrec *devrec)`

Описание:

Функция устанавливает соединение с модулем E-502 или L-502 по записи об этом устройстве. Необходимые действия зависят от того, на какое устройство подключенное по какому интерфейсу ссылается запись. Сами записи создаются специальными функциями (свои для каждого типа модуля и интерфейса подключения) и не должны изменяться пользователем вручную.

Параметры:

hnd — Описатель модуля.

devrec — Запись о устройстве, содержащая необходимую информацию для установления с ним связи

Возвращаемое значение:

Код ошибки.

4.3.3.12 Освобождение записей об устройствах

Формат: `int32_t X502_FreeDevRecordList (t_x502_devrec *list, uint32_t size)`

Описание:

Функция очищает ресурсы, выделенные при инициализации записи о устройстве под информацию, необходимую для открытия устройства. Данная функция должна вызываться после инициализации записи о устройстве одной из соответствующих функций после того, когда запись уже не нужна. После установки связи с устройством через `X502_OpenByDevRecord()` запись не используется в дальнейшем и ее можно при желании сразу освободить, не закрывая соединения с устройством. Функция может очистить сразу несколько записей из массива (если очищается одна, то в качестве размера допустимо указывать 1).

Параметры:

list — Массив записей о устройстве или указатель на единственную запись, ресурсы которой (которых) нужно освободить.

size — Количество записей в массиве

Возвращаемое значение:

Код ошибки.

4.3.4 Функции для изменения настроек модуля

4.3.4.1 Передача установленных настроек в модуль

Формат: `int32_t X502_Configure (t_x502_hnd hnd, uint32_t flags)`

Описание:

Функция выполняет запись текущих настроек (которые были установлены с помощью функций X502_SetXXX) в модуль. Должна вызываться перед запуском потока данных.

Параметры:

hnd — Описатель модуля.

flags — Флаги (резерв - должно быть равно 0).

Возвращаемое значение:

Код ошибки.

4.3.4.2 Установка параметров логического канала

Формат: `int32_t X502_SetLChannel (t_x502_hnd hnd, uint32_t lch, uint32_t phy_ch, uint32_t mode, uint32_t range, uint32_t avg)`

Описание:

Функция устанавливает параметры заданного логического канала в логической таблице АЦП.

Параметры:

hnd — Описатель модуля.

lch — Номер логического канала. (от 0 до `X502_LTABLE_MAX_CH_CNT`-1 или до `E16_LTABLE_MAX_CH_CNT`-1 для E16)

phy_ch — Номер физического канала АЦП, начиная с 0 (0-15 для дифференциального режима, 0-31 для режима с общей землей)

mode — Режим измерения канал АЦП (значение типа `t_x502_lch_mode`)

range — Диапазон измерения канала (значение типа `t_x502_adc_range`)

avg — Коэффициент усреднения по каналу (не реализовано в E16). Нулевое значение соответствует значению коэффициента, определенного библиотекой. Для явного задания коэффициента усреднения нужно перед значение от 1 (отсутствие усреднения) до `X502_LCH_AVG_SIZE_MAX`. В случае если значение усреднения превышает делитель частоты, то это значение будет скорректировано

Возвращаемое значение:

Код ошибки.

4.3.4.3 Установка количества логических каналов

Формат: `int32_t X502_SetLChannelCount (t_x502_hnd hnd, uint32_t lch_cnt)`

Описание:

Функция устанавливает количество логических каналов в логической таблице АЦП.

Параметры:

hnd — Описатель модуля

lch_cnt — Количество логических каналов (от 1 до `X502_LTABLE_MAX_CH_CNT` или до `E16_LTABLE_MAX_CH_CNT` для E16)

Возвращаемое значение:

Код ошибки

4.3.4.4 Получение количества логических каналов

Формат: `int32_t X502_GetLChannelCount (t_x502_hnd hnd, uint32_t *lch_cnt)`

Описание:

Функция возвращает установленное ранее с помощью `X502_SetLChannelCount()` количество логических каналов в управляющей таблице АЦП.

Параметры:

hnd — Описатель модуля

lch_cnt — Количество логических каналов

Возвращаемое значение:

Код ошибки

4.3.4.5 Установка делителя частоты сбора для АЦП

Формат: `int32_t X502_SetAdcFreqDivider (t_x502_hnd hnd, uint32_t adc_freq_div)`

Описание:

Частота сбора АЦП получается как результат деления опорной частоты синхронизации (как в случае внешней, так и внутренней) на делитель, устанавливаемый этой функцией.

Альтернативой этой функции служит `X502_SetAdcFreq()`, которая рассчитывает этот делитель на основании переданной требуемой частоты сбора АЦП.

Параметры:

hnd — Описатель модуля.

adc_freq_div — Делитель частоты АЦП (от 1 до `X502_ADC_FREQ_DIV_MAX`).

Возвращаемое значение:

Код ошибки.

4.3.4.6 Установка значения межкадровой задержки для АЦП

Формат: `int32_t X502_SetAdcInterframeDelay (t_x502_hnd hnd, uint32_t delay)`

Описание:

Функция устанавливает межкадровую задержку для АЦП, то есть количество периодов опорной частоты синхронизации, которое будет пропущено после проведения измерения последнего канала логической таблицы до проведения измерения, соответствующего первому логическому каналу следующего кадра.

Альтернативой может являться функция `X502_SetAdcFreq()`, которая рассчитывает значение межкадровой задержки по заданным параметрам частоты сбора и частоты следования кадров (частоты сбора на логический канал).

Параметры:

hnd — Описатель модуля.

delay — Значение межкадровой задержки (от 0 до `X502_ADC_INTERFRAME_DELAY_MAX`)

Возвращаемое значение:

Код ошибки.

4.3.4.7 Установка делителя частоты синхронного ввода с цифровых линий.

Формат: `int32_t X502_SetDinFreqDivider (t_x502_hnd hnd, uint32_t din_freq_div)`

Описание:

Частота синхронного ввода данных с цифровых входов получается как результат деления опорной частоты синхронизации на делитель, устанавливаемый этой функцией.

Альтернативой этой функции служит `X502_SetDinFreq()`, которая рассчитывает этот делитель на основании переданной требуемой частоты синхронного ввода с цифровых линий.

Параметры:

hnd — Описатель модуля.

din_freq_div — Делитель частоты синхронного ввода с цифровых линий (от 1 до `X502_DIN_FREQ_DIV_MAX`).

Возвращаемое значение:

Код ошибки.

4.3.4.8 Установка делителя частоты синхронного вывода

Формат: `int32_t X502_SetOutFreqDivider (t_x502_hnd hnd, uint32_t out_freq_div)`

Описание:

Частота синхронного вывода данных получается как результат деления опорной частоты синхронизации на делитель, устанавливаемый этой функцией. Используется общая частота вывода для каждого канала ЦАП и для цифровых линий (вывод осуществляется параллельно). Частота вывода не может быть больше половины опорной частоты.

Альтернативой этой функции служит `X502_SetOutFreq()`, которая рассчитывает этот делитель на основании переданной требуемой частоты синхронного вывода.

Для модуля L-502, чтобы была возможность установить делитель, отличный от `X502_OUT_FREQ_DIV_DEFAULT`, необходимо обновить прошивку ПЛИС до версии 0.5 или выше. Для модуля E-502 возможность всегда поддерживается. Проверить программно наличие данной возможности можно с помощью функции `X502_CheckFeature()`.

Параметры:

hnd — Описатель модуля.

out_freq_div — Делитель частоты синхронного вывода (от `X502_OUT_FREQ_DIV_MIN` до `X502_OUT_FREQ_DIV_MAX`).

Возвращаемое значение:

Код ошибки.

4.3.4.9 Установка частоты сбора АЦП

Формат: `int32_t X502_SetAdcFreq (t_x502_hnd hnd, double *f_acq, double *f_frame)`

Описание:

Функция подбирает делитель частоты АЦП так, чтобы полученная частота сбора была наиболее близка к указанной в параметре f_acq. Функция возвращает в этом же параметре реальную частоту, которая была установлена.

Так же функция может подобрать значение межкадровой задержки так, чтобы частота следования кадров (частота сбора на логический канал) была наиболее близка к указанному значению. Для этого следует передать требуемое значение в переменной f_frame (в ней также по завершению будет возвращено значение установленной частоты). Если в качестве f_frame передан нулевой указатель, то будет установлена нулевая межкадровая задержка.

Если необходимо изменить значение опорной частоты, то данная функция должна быть вызвана после `X502_SetSyncMode()` и `X502_SetRefFreq()` / `X502_SetExtRefFreqValue()`, в противном случае полученные делители будут давать неверное значение частоты.

Если устанавливается частота кадров, то функция должна вызываться после того, как было заданно нужное количество логических каналов в управляющей таблице с помощью `X502_SetLChannelCount()`.

При использовании внешней опорной частоты синхронизации эта функция будет давать верный результат, только если эта внешняя частота соответствует значению, установленному с помощью `X502_SetRefFreq()`.

Параметры:

hnd — Описатель модуля.

f_acq — На входе принимает требуемое значения частоты сбора АЦП в Герцах. На выходе возвращает реально установленное значение частоты.

f_frame — На входе принимает требуемое значение частоты сбора кадров (частоты сбора на логический канал) АЦП в Герцах. На выходе возвращает реально установленное значение. Если передан нулевой указатель, то устанавливает максимальную частоту сбора кадров (нулевую межкадровую задержку).

Возвращаемое значение:

Код ошибки.

4.3.4.10 Установка частоты синхронного ввода с цифровых входов

Формат: `int32_t X502_SetDinFreq (t_x502_hnd hnd, double *f_din)`

Описание:

Функция подбирает делитель частоты ввода значений с цифровых входов так, чтобы полученная частота ввода была наиболее близка к указанной. Функция возвращает в этом же параметре реальную частоту, которая была установлена.

Если необходимо изменить значение опорной частоты синхронизации, то данная функция должна быть вызвана после `X502_SetSyncMode()` и `X502_SetRefFreq()`/`X502_SetExtRefFreqValue()`, в противном случае полученный делитель будет давать неверное значение частоты.

При использовании внешней опорной частоты синхронизации эта функция будет давать верный результат, только если эта внешняя частота соответствует значению, установленному с помощью `X502_SetRefFreq()`.

Параметры:

hnd — Описатель модуля.

f_din — На входе принимает требуемое значения частоты ввода с цифровых входов в Герцах. На выходе возвращает реально установленное значение частоты.

Возвращаемое значение:

Код ошибки.

4.3.4.11 Установка частоты синхронного вывода

Формат: `int32_t X502_SetOutFreq (t_x502_hnd hnd, double *f_dout)`

Описание:

Функция подбирает делитель частоты синхронного вывода так, чтобы полученная частота была наиболее близка к указанной. Функция возвращает в этом же параметре реальную частоту, которая была установлена.

Если необходимо изменить значение опорной частоты синхронизации, то данная функция должна быть вызвана после `X502_SetSyncMode()` и `X502_SetRefFreq()` / `X502_SetExtRefFreqValue()`, в противном случае полученный делитель будет давать неверное значение частоты.

При использовании внешней опорной частоты синхронизации эта функция будет давать верный результат, только если эта внешняя частота соответствует значению, установленному с помощью `X502_SetRefFreq()`.

Для модуля L-502, чтобы была возможность установить частоту, отличную от опорной частоты, деленной на `X502_OUT_FREQ_DIV_DEFAULT`, необходимо обновить прошивку ПЛИС до версии 0.5 или выше. Для модуля E-502 возможность всегда поддерживается. Проверить программно наличие данной возможности можно с помощью функции `X502_CheckFeature()`.

Параметры:

hnd — Описатель модуля.

f_dout — На входе принимает требуемое значения частоты синхронного вывода в Герцах. На выходе возвращает реально установленное значение частоты.

Возвращаемое значение:

Код ошибки.

4.3.4.12 Получить текущие значения частот сбора АЦП

Формат: `int32_t X502_GetAdcFreq (t_x502_hnd hnd, double *f_acq, double *f_frame)`

Описание:

Функция возвращает ткущие установленные для модуля значения частоты сбора и частоты кадров АЦП (частоты на логический канал) в Герцах, которые были установлены до этого с помощью `X502_SetAdcFreq()` или с помощью функций `X502_SetAdcFreqDivider()` / `X502_SetAdcInterframeDelay()`.

Параметры:

hnd — Описатель модуля.

f_acq — Если не NULL, то на выходе возвращается текущее значение частоты сбора АЦП.

f_frame — Если не NULL, то на выходе возвращается текущее значение частоты кадров АЦП.

4.3.4.13 Установка значения внутренней опорной частоты синхронизации

Формат: `int32_t X502_SetRefFreq (t_x502_hnd hnd, uint32_t freq)`

Описание:

Функция задает значение внутренней опорной частоты синхронизации, от которой получаются все частоты синхронного ввода/вывода посредством деления на определенный делитель.

Данная функция при внутренней опорной частете выбирает одну из двух доступных частот в 2МГц или 1.5 МГц(2МГц является значением по умолчанию), для задания которых можно введены константы из `t_x502_ref_freq`.

При использовании внешней опорной частоты следует использовать `X502_SetExtRefFreqValue()`.

Для модуля E-502 вывод на ЦАП при опорной частоте 1.5 МГц работает только для версии прошивки PLDA 1 или выше.

Параметры:

hnd — Описатель модуля.

freq — Значение из `t_x502_ref_freq`, которое задает выбранную опорную частоту.

Возвращаемое значение:

Код ошибки.

4.3.4.14 Установка значения внешней опорной частоты синхронизации

Формат: `int32_t X502_SetExtRefFreqValue (t_x502_hnd hnd, double freq)`

Описание:

При установке внешней опорной частоты (вызов `X502_SetSyncMode()` со занчением, отличным от `X502_SYNC_INTERNAL`) данная функция позволяет задать частоту внешней опорной частоты, котороя может быть любая, но не превышать 1.5 МГц.

Данная функция не влияет на настройки самого модуля, однако установка корректного значения позволяет правильно установить нужную частоту сбора функциями `X502_SetAdcFreq()`, `X502_SetDinFreq()` и `X502_SetOutFreq()`, а также корректно рассчитать значения по умолчанию для размера буфера и шага передачи данных между модулем и ПК.

Данная функция доступна в библиотеке версии 1.1.4 или выше.

Параметры:

hnd — Описатель модуля.
freq — Значение внешней опорной частоты в Гц.

Возвращаемое значение:

Код ошибки.

4.3.4.15 Установка значения для аналоговой синхронизации (только для E16)

Формат: `int32_t X502_SetAdcSyncStartValue (t_x502_hnd hnd, uint32_t phy_ch, uint32_t mode, uint32_t range, double value)`

Описание:

Параметры:

hnd — Описатель модуля.
phy_ch — Номер физического канала АЦП, начиная с 0 (0-15 для дифференциального режима, 0-31 для режима с общей землей)
mode — Режим измерения канал АЦП (значение типа `t_x502_lch_mode`)
range — Диапазон измерения канала (значение типа `t_x502_adc_range`)
value — Значение в Вольтах.

Возвращаемое значение:

Код ошибки.

4.3.4.16 Получение значения опорной частоты синхронизации

Формат: `int32_t X502_GetRefFreqValue (t_x502_hnd hnd, double *freq)`

Описание:

Данная функция возвращает текущее значение опорной частоты синхронизации, которое используется библиотекой в функциях `X502_SetAdcFreq()`, `X502_SetDinFreq()` и `X502_SetOutFreq()`, а также при рассчете параметров передачи данных между модулем и ПК.

При внутренней опорной частоте используется значение, установленное `X502_SetRefFreq()` (1.5 или 2 Мгц), при внешней — частота, установленная с помощью функции `X502_SetExtRefFreqValue()`.

Данная функция доступна в библиотеке версии 1.1.4 или выше.

Параметры:

hnd — Описатель модуля.

freq — Значение внешней опорной частоты в Гц.

Возвращаемое значение:

Код ошибки.

4.3.4.17 Установка режима генерации частоты синхронизации

Формат: `int32_t X502_SetSyncMode (t_x502_hnd hnd, uint32_t sync_mode)`

Описание:

Функция устанавливает кто будет генератором опорной частоты синхронизации - сам модуль или будет использоваться внешний сигнал.

В режиме `X502_SYNC_INTERNAL` модуль сам будет генерировать для себя частоту синхронизации с частотой, заданной `X502_SetRefFreq()`. При этом запуск генерации будет осуществлен по вызову `X502_StreamsStart()` или по условию, заданому в `X502_SetSyncStartMode()`, а останов по `X502_StreamsStop()`.

В остальных режимах сбор будет осуществляться по внешнему сигналу синхронизации.

Параметры:

hnd — Описатель модуля.

sync_mode — Значение из `t_x502_sync_mode`, определяющее кто будет источником частоты синхронизации.

Возвращаемое значение:

Код ошибки.

4.3.4.18 Установка режима запуска частоты синхронизации

Формат: `int32_t X502_SetSyncStartMode (t_x502_hnd hnd, uint32_t sync_start_mode)`

Описание:

Функция устанавливает условие запуска синхронного ввода/вывода данных.

Если с помощью `X502_SetSyncMode()` установлен режим синхронизации `X502_SYNC_INTERNAL`, то по заданному данной функцией условию модуль начнет генерировать частоту синхронизации, в противном случае по заданному условию модуль начнет использовать внешне заданную частоту синхронизации (т.е. до выполнения условия сигнал синхронизации на заданном входе будет игнорироваться).

Режимы задания условия запуска синхронизации имеют те же значения, что и режимы задания самой частоты (см. тип `t_x502_sync_mode`). В случае `X502_SYNC_INTERNAL` запуск осуществляется при выполнении функции `X502_StreamsStart()`, в противном случае - после выполнения `X502_StreamsStart()` модуль начинает ожидать заданного данной функцией условия. Т.е. даже при задании внешних источников синхронизации, все равно необходимо вызывать `X502_StreamsStart()`.

Параметры:

hnd — Описатель модуля.

sync_start_mode — Значение из `t_x502_sync_mode`, определяющее условие запуска частоты синхронизации.

Возвращаемое значение:

Код ошибки.

4.3.4.19 Установить режим работы модуля

Формат: `int32_t X502_SetMode (t_x502_hnd hnd, uint32_t mode)`

Описание:

Функция устанавливает режим работы модуля, который определяет будет ли потоки данных обрабатывать ПЛИС или сигнальный процессор BlackFin. При включении питания модулем всегда управляет ПЛИС. После загрузки прошивки с помощью `X502_BfLoadFirmware()` модуль переходит в режим управления сигнальным процессором.

Данная функция может использоваться для ручной установки режима, например, для возврата в режим управления ПЛИС или для переключения в режим управления сигнальным процессором, если прошивка уже была загружена (например, через JTAG интерфейс при отладке).

Параметры:

hnd — Описатель модуля.

mode — Режим работы модуля из `t_x502_mode`.

Возвращаемое значение:

Код ошибки.

4.3.4.20 Получение текущего режима работы модуля

Формат: `int32_t X502_GetMode (t_x502_hnd hnd, uint32_t *mode)`
Описание: Функция возвращает текущий режим работы модуля.
Параметры: **hnd** — Описатель модуля. **mode** — В данном параметре возвращается текущий режим работы модуля (из `t_x502_mode`).
Возвращаемое значение: Код ошибки.

4.3.4.21 Установить коэффициенты для калибровки значений АЦП

Формат: `int32_t X502_SetAdcCoef (t_x502_hnd hnd, uint32_t range, double k, double offs)`
Описание: Функция записывает в ПЛИС коэффициенты для калибровки значений АЦП. При открытии модуля, библиотека считывает калибровочные коэффициенты из защищенной области Flash-памяти модуля и записывает их в ПЛИС для выполнения калибровки на лету. Результирующее значение АЦП вычисляется по формуле (val - offs) * k, где val — некалиброванное значение. Данная функция позволяет изменить используемые коэффициенты в то время, пока не запущен синхронный сбор данных. При этом изменяются только текущие коэффициенты, а заводские калибровочные коэффициенты из Flash-памяти сохраняют свое значение и при следующем открытии будут восстановлены.
Параметры: **hnd** — Описатель модуля. **range** — Диапазон АЦП (из `t_x502_adc_range`). **k** — Устанавливаемое значение коэффициента шкалы. **offs** — Устанавливаемое значение смещения нуля.
Возвращаемое значение: Код ошибки.

4.3.4.22 Получение текущих калибровочных коэффициентов АЦП

Формат: `int32_t X502_GetAdcCoef (t_x502_hnd hnd, uint32_t range, double *k, double *offs)`

Описание:

Функция возвращает текущие калибровочные коэффициенты для заданного диапазона измерения АЦП. Эти коэффициенты могут отличаться от заводских значений, сохраненных во Flash-памяти модуля, например, если пользователь использовал `X502_SetAdcCoef()` для установки своих коэффициентов.

Параметры:

hnd — Описатель модуля.
range — Диапазон АЦП (из `t_x502_adc_range`).
k — В данной переменной возвращается текущий коэффициент шкалы.
offs — В данной переменной возвращается текущее смещение нуля.

Возвращаемое значение:

Код ошибки.

4.3.4.23 Установить коэффициенты для калибровки значений ЦАП

Формат: `int32_t X502_SetDacCoef (t_x502_hnd hnd, uint32_t ch, double k, double offs)`

Описание:

Функция устанавливает калибровочные коэффициенты для заданного канала АЦП, которые будут использоваться функциями x502api для калибровки выводимых значений ЦАП, если указан фалаг `X502_DAC_FLAGS_CALIBR`.

Откалиброванное значение ЦАП в кодах получается как (val + offs) * k, где val — некалиброванное значение (в кодах).

При открытии модуля, библиотека считывает калибровочные коэффициенты из защищенной области Flash-памяти модуля и использует их.

Данная функция нужна только если пользователь хочет использовать свои коэффициенты. При этом она не изменяет значения во Flash-памяти, т.е. при следующем открытии модуля коэффициенты будут снова восстановлены из Flash-памяти.

Параметры:

hnd — Описатель модуля.
ch — Канал ЦАП (из `t_x502_dac_ch`).
k — Устанавливаемое значение коэффициента шкалы.
offs — Устанавливаемое значение смещения нуля.

Возвращаемое значение:

Код ошибки.

4.3.4.24 Получение текущих калибровочных коэффициентов ЦАП

Формат: `int32_t X502_GetDacCoef (t_x502_hnd hnd, uint32_t ch, double *k, double *offs)`

Описание:

Функция возвращает текущие калибровочные коэффициенты для заданного канала ЦАП. Эти коэффициенты могут отличаться от заводских значений, сохраненных во Flash-памяти модуля, например, если пользователь использовал `X502_SetDacCoef()` для установки своих коэффициентов.

Параметры:

hnd — Описатель модуля.
ch — Канал ЦАП (из `t_x502_dac_ch`).
k — В данной переменной возвращается текущий коэффициент шкалы.
offs — В данной переменной возвращается текущее смещение нуля.

Возвращаемое значение:

Код ошибки.

4.3.4.25 Рассчет частоты сбора АЦП

Формат: `int32_t X502_CalcAdcFreq (double ref_freq, uint32_t lch_cnt, double *f_acq, double *f_frame, uint32_t *result_freq_div, uint32_t *result_frame_delay)`

Описание:

Исходя из заданных параметров, функция подбирает делитель частоты АЦП и значение межкадровой задержки так, чтобы полученные частоты были наиболее близки к заданным, и возвращает полученные значения частот.

В отличие от `X502_SetAdcFreq()`, данная функция предназначена получения скорректированной частоты без использования описателя модуля и только рассчитывает результирующие параметры, не изменяя настройки.

Для модуля E16 в параметре ref_freq следует передавать E16_REF_FREQ_48000KHZ, либо использовать функцию X502_CalcAdcFreq2!

Параметры:

ref_freq — Значение опорной частоты в Гц (внешней или внутренней)

lch_cnt — Количество логических каналов, которое будет использовано. Необходимо для расчета межкадровой задержки. Если в качестве f_frame передан нулевой указатель, то может быть равно 0.

f_acq — На входе принимает требуемое значения частоты сбора АЦП в Герцах. На выходе возвращает рассчитанное значение частоты, которая может быть установлена.

f_frame — На входе принимает требуемое значение частоты сбора кадров (частоты сбора на логический канал) АЦП в Герцах. На выходе возвращает рассчитанное значение. Если передан нулевой указатель, то задержка рассчитана не будет. Если передано значение меньше или равное нуля, то будет рассчитана максимальная частота кадров (с нулевой межкадровой задержкой).

result_freq_div — В данном параметре возвращается рассчитанное значения делителя частоты АЦП. Может быть передан нулевой указатель, если это значение явно знать не требуется.

result_frame_delay — В данном параметре возвращается рассчитанное значение межкадровой задержки. Может быть передан нулевой указатель, если это значение явно знать не требуется.

Возвращаемое значение:

Код ошибки.

4.3.4.26 Рассчет частоты сбора АЦП

Формат: `int32_t X502_CalcAdcFreq2 (t_x502_hnd hnd, double ref_freq, uint32_t lch_cnt, double *f_acq, double *f_frame, uint32_t *adc_freq_div, uint32_t *adc_frame_delay)`

Описание:

Исходя из заданных параметров, функция подбирает делитель частоты АЦП и значение межкадровой задержки так, чтобы полученные частоты были наиболее близки к заданным, и возвращает полученные значения частот.

В отличие от `X502_SetAdcFreq()`, данная функция предназначена получения скорректированной частоты и только рассчитывает результирующие параметры, не изменяя настройки. Описатель модуля в этой функции нужен для того чтобы различать модули E16 и X502.

Параметры:

hnd — Описатель модуля.

ref_freq — Значение опорной частоты в Гц (внешней или внутренней)

lch_cnt — Количество логических каналов, которое будет использовано. Необходимо для расчета межкадровой задержки. Если в качестве f_frame передан нулевой указатель, то может быть равно 0.

f_acq — На входе принимает требуемое значения частоты сбора АЦП в Герцах. На выходе возвращает рассчитанное значение частоты, которая может быть установлена.

f_frame — На входе принимает требуемое значение частоты сбора кадров (частоты сбора на логический канал) АЦП в Герцах. На выходе возвращает рассчитанное значение. Если передан нулевой указатель, то задержка рассчитана не будет. Если передано значение меньше или равное нуля, то будет рассчитана максимальная частота кадров (с нулевой межкадровой задержкой).

result_freq_div — В данном параметре возвращается рассчитанное значения делителя частоты АЦП. Может быть передан нулевой указатель, если это значение явно знать не требуется.

result_frame_delay — В данном параметре возвращается рассчитанное значение межкадровой задержки. Может быть передан нулевой указатель, если это значение явно знать не требуется.

Возвращаемое значение:

Код ошибки.

4.3.4.27 Рассчет частоты синхронного ввода с цифровых входов

Формат: `int32_t X502_CalcDinFreq (double ref_freq, double *f_din, uint32_t *result_freq_div)`

Описание:

Исходя из заданных параметров, функция подбирает делитель частоты ввода значений с цифровых входов так, чтобы полученная частота ввода была наиболее близка к указанной, и возвращает полученное значение частоты.

В отличие от X502_SetDinFreq(), данная функция предназначена получения скорректированной частоты без использования описателя модуля и только рассчитывает результирующие параметры, не изменяя настройки.

Для модуля E16 в параметре ref_freq следует передавать E16_REF_FREQ_48000KHZ, либо использовать функцию X502_CalcDinFreq2!

Параметры:

ref_freq — Значение опорной частоты в Гц (внешней или внутренней)

f_din — На входе принимает требуемое значения частоты ввода с цифровых входов в Герцах. На выходе возвращает рассчитанное значение частоты, которое может быть установлено.

result_freq_div — В данном параметре возвращается рассчитанное значения делителя частоты синхронного ввода цифровых линий. Может быть передан нулевой указатель, если это значение явно знать не требуется.

Возвращаемое значение:

Код ошибки.

4.3.4.28 Рассчет частоты синхронного ввода с цифровых входов

Формат: `int32_t X502_CalcDinFreq2 (t_x502_hnd hnd, double ref_freq, double *f_din, uint32_t *result_freq_div)`

Описание:

Исходя из заданных параметров, функция подбирает делитель частоты ввода значений с цифровых входов так, чтобы полученная частота ввода была наиболее близка к указанной, и возвращает полученное значение частоты.

В отличие от X502_SetDinFreq(), данная функция только рассчитывает результирующие параметры, не изменяя настройки.

Параметры:

hnd — Описатель модуля.

ref_freq — Значение опорной частоты в Гц (внешней или внутренней)

f_din — На входе принимает требуемое значения частоты ввода с цифровых входов в Герцах. На выходе возвращает рассчитанное значение частоты, которое может быть установлено.

result_freq_div — В данном параметре возвращается рассчитанное значения делителя частоты синхронного ввода цифровых линий. Может быть передан нулевой указатель, если это значение явно знать не требуется.

Возвращаемое значение:

Код ошибки.

4.3.4.29 Рассчет частоты синхронного вывода

Формат: `int32_t X502_CalcOutFreq (double ref_freq, double *f_dout, uint32_t *result_freq_div)`

Описание:

Исходя из заданных параметров, функция подбирает делитель частоты синхронного вывода так, чтобы полученная частота была наиболее близка к указанной, и возвращает полученное значение частоты.

В отличие от `X502_SetOutFreq()`, данная функция предназначена получения скорректированной частоты без использования описателя модуля и только рассчитывает результирующие параметры, не изменяя настройки.

Функция предполагает, что модуль поддерживает изменение частоты вывода (см. требования к модулю для этого в описании функции `X502_SetOutFreq()`).

Для модуля E16 в параметре ref_freq следует передавать E16_REF_FREQ_48000KHZ, либо использовать функцию X502_CalcOutFreq2!

Параметры:

ref_freq — Значение опорной частоты в Гц (внешней или внутренней)

f_dout — На входе принимает требуемое значения частоты синхронного вывода в Герцах. На выходе возвращает рассчитанное значение частоты, которое может быть установлено.

result_freq_div — В данном параметре возвращается рассчитанное значения делителя частоты синхронного вывода. Может быть передан нулевой указатель, если это значение явно знать не требуется.

Возвращаемое значение:

Код ошибки.

4.3.4.30 Рассчет частоты синхронного вывода

Формат: `int32_t X502_CalcOutFreq2 (t_x502_hnd hnd, double ref_freq, double *f_dout, uint32_t *result_freq_div)`

Описание:

Исходя из заданных параметров, функция подбирает делитель частоты синхронного вывода так, чтобы полученная частота была наиболее близка к указанной, и возвращает полученное значение частоты.

В отличие от `X502_SetOutFreq()`, данная функция предназначена получения скорректированной частоты без использования описателя модуля и только рассчитывает результирующие параметры, не изменяя настройки.

Функция предполагает, что модуль поддерживает изменение частоты вывода (см. требования к модулю для этого в описании функции `X502_SetOutFreq()`).

Параметры:

hnd — Описатель модуля.

ref_freq — Значение опорной частоты в Гц (внешней или внутренней)

f_dout — На входе принимает требуемое значения частоты синхронного вывода в Герцах. На выходе возвращает рассчитанное значение частоты, которое может быть установлено.

result_freq_div — В данном параметре возвращается рассчитанное значения делителя частоты синхронного вывода. Может быть передан нулевой указатель, если это значение явно знать не требуется.

Возвращаемое значение:

Код ошибки.

4.3.5 Функции асинхронного ввода-вывода

4.3.5.1 Асинхронный вывод данных на канал ЦАП

Формат: `int32_t X502_AsyncOutDac (t_x502_hnd hnd, uint32_t ch, double data, uint32_t flags)`

Описание:

Функция выводит указанное значение на указанный канал ЦАП. Значение может быть задано как в кодах, так и в Вольтах, и к нему могут быть применены калибровочные коэффициенты (определяется флагами).

Функция может вызываться либо когда синхронный сбор не запущен, либо при запущенном сборе данных, если синхронный вывод по этому каналу ЦАП не разрешен.

Функция не работает в случае, если модуль находится в ожидании внешнего условия для запуска синхронного ввода (была вызвана `X502_StreamsStart()` при внешней синхронизации старта, но сам сигнал запуска еще не возник).

Параметры:

hnd — Описатель модуля.

ch — Номер канала ЦАП (из `t_x502_dac_ch`).

data — Выводимое значение на ЦАП (в кодах или вольтах)

flags — Флаги из `t_x502_dacout_flags`.

Возвращаемое значение:

Код ошибки.

4.3.5.2 Асинхронный вывод данных на цифровые выходы

Формат: `int32_t X502_AsyncOutDig (t_x502_hnd hnd, uint32_t val, uint32_t msk)`

Описание:

Функция выводит указанное значение на цифровые выходы модуля. Формат значения аналогичен `X502_PrepareData()` - в младших 16 битах указывается выводимое значение, а в старшие - флаги (с помощью которых можно перевести одну из половин в третье состояние).

Функция может вызываться либо когда синхронный сбор не запущен, либо при запущенном сборе данных, если синхронный вывод по цифровым линиям не разрешен.

Можно использовать маску, чтобы вывести только на часть выводов, оставив остальные неизменными, однако следует учесть, что после открытия связи с модулем необходимо сперва сделать вывод на все линии, после чего уже можно использовать маску при последующих вызовах.

Функция не работает в случае, если модуль находится в ожидании внешнего условия для запуска синхронного ввода (была вызвана `X502_StreamsStart()` при внешней синхронизации старта, но сам сигнал запуска еще не возник).

Параметры:

hnd — Описатель модуля.

val — Младшая половина - выводимое значение, старшая - флаги из `t_x502_digout_word_flags`.

msk — Маска - указанные в маске биты не будут изменяться с предыдущего выведенного состояния (распространяется и на старшую половину val).

Возвращаемое значение:

Код ошибки.

4.3.5.3 Асинхронный ввод значений с цифровых входов

Формат: `int32_t X502_AsyncInDig (t_x502_hnd hnd, uint32_t *din)`

Описание:

Функция считывает текущее значение цифровых входов. При этом синхронный сбор цифровых входов не должен быть запущен (не разрешен поток `X502_STREAM_DIN`).

Так как модули E-502/L-502 не поддерживают аппаратно асинхронный ввод, то если на момент вызова этой функции не запущен синхронный ввод/вывод с помощью `X502_StreamsStart()`, то данная функция на время выполнения запускает синхронный сбор и останавливает его как только будет получено одно новое значение цифровых входов, для модулей E16 этого не требуется.

Для модуля E16 17 бит - состояние входа INT, 18 бит - TRIG

Функция не работает в случае, если модуль находится в ожидании внешнего условия для запуска синхронного ввода (была вызвана `X502_StreamsStart()` при внешней синхронизации старта, но сам сигнал запуска еще не возник).

Параметры:

hnd — Описатель модуля.

din — При успешном выполнении в этой переменной возвращается текущее состояние цифровых входов. Действительны младшие 18 бит, старшие 14 - резерв. Часть битов при этом объединены с линиями синхронизации, при этом это объединение зависит от типа модуля. Подробнее описано в разделе с различиями модулей E-502 и L-502. Резервные биты могут быть использованы в последующих версиях, не следует считать, что они всегда будут равны нулю!

Возвращаемое значение:

Код ошибки.

4.3.5.4 Асинхронный ввод одного кадра АЦП

Формат: `int32_t X502_AsyncGetAdcFrame (t_x502_hnd hnd, uint32_t flags, uint32_t tout, double *data)`

Описание:

Функция производит однократный ввод кадра в соответствии с заранее установленной логической таблицей. Частота сбора АЦП соответствует частоте, установленной с помощью `X502_SetAdcFreq()`. Частота следования кадров значения не имеет. Сам кадр вводится синхронно, но при последовательном вызове `X502_AsyncGetAdcFrame()` для измерения нескольких кадров задержка между этими кадрами не определена.

Функция так же выполняет обработку принятых данных АЦП, аналогично `X502_ProcessAdcData()`, и принимает набор флагов, аналогичный `X502_ProcessAdcData()`.

Для работы этой функции не должен быть разрешен синхронный ввод АЦП и цифровых линий.

Так как аппаратно асинхронный ввод в плате отсутствует, то эта функция в случае не запущенного потока запускает его внутри себя, принимает один кадр данных и после этого останавливает синхронный сбор.

Параметры:

hnd — Описатель модуля.

flags — Флаги из t_x502_proc_flags

tout — Таймаут на выполнение функции в мс

data — Массив, в котором в случае успеха будут возвращены отсчеты кадра АЦП. Должен быть размером, достаточным для хранения отсчетов типа double в количестве, равном количеству установленных логических каналов в управляющей таблице АЦП.

Возвращаемое значение:

Код ошибки.

4.3.6 Функции для работы с синхронным потоковым вводом-выводом

4.3.6.1 Разрешение синхронных потоков на ввод/вывод

Формат: `int32_t X502_StreamsEnable (t_x502_hnd hnd, uint32_t streams)`

Описание:

Функция разрешает прием/передачу для указанных потоков Не указанные потоки сохраняют свое разрешенное или запрещенное состояние. Может вызываться как до `X502_Configure()`, так и после. Разрешенные потоки устанавливаются как правило до вызова `X502_StreamsStart()`.

При желании в некоторых ситуациях можно изменять состав разрешенных потоков во время запущенного сбора данных, однако если эти потоки сильно различаются в частоте, то рассчитанные библиотекой значения буфера и шага прерывания могут не подходить для изменившихся значений (см. `Размер буфера и шаг для синхронного режима`)

Параметры:

hnd — Описатель модуля.

streams — Набор флагов `t_x502_streams`, указывающих, какие потоки должны быть разрешены.

Возвращаемое значение:

Код ошибки.

4.3.6.2 Запрещение синхронных потоков на ввод/вывод

Формат: `int32_t X502_StreamsDisable (t_x502_hnd hnd, uint32_t streams)`

Описание:

Функция запрещает передачу синхронных данных для указанных потоков. Не указанные потоки сохраняют свое разрешенное или запрещенное состояние. Функция, противоположная по смыслу `X502_StreamsEnable()`.

Параметры:

hnd — Описатель модуля.

streams — Набор флагов `t_x502_streams`, указывающих, какие потоки должны быть запрещены.

Возвращаемое значение:

Код ошибки.

4.3.6.3 Получить значение, какие синхронные потоки разрешены

Формат: `int32_t X502_GetEnabledStreams (t_x502_hnd hnd, uint32_t *streams)`

Описание:

Функция позволяет получить набор флагов, которые указывают, какие синхронные потоки сейчас разрешены.

Параметры:

hnd — Описатель модуля.

streams — Набор флагов `t_x502_streams`, указывающих, какие потоки сейчас разрешены.

Возвращаемое значение:

Код ошибки.

4.3.6.4 Запуск синхронных потоков ввода/вывода

Формат: `int32_t X502_StreamsStart (t_x502_hnd hnd)`

Описание:

Функция запуска синхронных потоков данных. Все синхронные потоки тактируются от общей опорной частоты. Если был установлен внутренний старт синхронизации, то синхронизация потоков начнется при выполнении данной функции, в противном случае по данной функции модуль перейдет в состояние ожидания внешнего признака начальной синхронизации.

Также функция осуществляет инициализацию канала DMA на ввод данных из платы, если был разрешен поток АЦП или синхронного ввода цифровых линий, и инициализацию канала DMA на вывод, если был разрешен хотя бы один поток на вывод, но не была вызвана функция `X502_PreloadStart()` (однако в этом случае начало вывода не совпадет с началом ввода).

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.6.5 Останов синхронных потоков ввода/вывода

Формат: `int32_t X502_StreamsStop (t_x502_hnd hnd)`

Описание:

Функция останова синхронных потоков ввода/вывода данных. После выполнению этой функции модуль завершает генерацию опорной частоты синхронизации (или использовать внешнюю частоту синхронизации) и останавливает синхронную передачу данных.

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.6.6 Проверка, запущен ли синхронный ввод/вывод

Формат: `int32_t X502_IsRunning (t_x502_hnd hnd)`

Описание:

Функция проверяет запущен ли синхронный ввод вывод с помощью `X502_StreamsStart()` или какой-либо внутренней логикой в прошивки BlackFin. Если сбор данных не запущен, то функция возвращает ошибку `X502_ERR_STREAM_IS_NOT_RUNNING`, если запущен, то нулевой код ошибки

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.6.7 Чтение данных АЦП и цифровых входов из модуля

Формат: `int32_t X502_Recv (t_x502_hnd hnd, uint32_t *buf, uint32_t size, uint32_t tout)`

Описание:

Функция считывает данные от модуля, которые были приняты в промежуточный буфер в драйвере или библиотеке. Функция принимает отсчеты в специальном индексном формате, в котором содержится информация, что это за данные (значения цифровых входов или отсчеты АЦП) и дополнительная информация для АЦП (номер канала, режим). Для разбора полученных отсчетов используется функция `X502_ProcessData()`.

Если в буфере сейчас находится меньше отсчетов, чем было запрошено, то функция будет ожидать пока придет заданное количество данных или пока не истечет указанный таймаут. В последнем случае функция возвратит столько отсчетов, сколько было в буфере при истечении таймаута.

Количество готовых для чтения отсчетов в буфере драйвера можно при желании узнать с помощью функции `X502_GetRecvReadyCount()`.

До вызовов `X502_Recv()` синхронный поток сбора данных должен быть уже запущен с помощью `X502_StreamsStart()`.

Параметры:

hnd — Описатель модуля.
buf — Буфер, в которые будут сохранены отсчеты.
size — Количество считываемых отсчетов (32-битных слов).
tout — Таймаут на прием данных в мс.

Возвращаемое значение:

Если < 0 - код ошибки. Если $>= 0$ - количество считанных слов.

4.3.6.8 Передача потоковых данных ЦАП и цифровых выходов в модуль

Формат: `int32_t X502_Send (t_x502_hnd hnd, const uint32_t *buf, uint32_t size, uint32_t tout)`

Описание:

Функция записывает данные на передачу в промежуточный буфер, после чего эти данные будут переданы в модуль. Данные должны быть в специальном формате, который определяет, что это за данные (цифровые выходы, канал ЦАП1 или канал ЦАП2). Подготовить данные в нужном формате можно с помощью `X502_PrepareData()`.

Если промежуточный буфер на передачу заполнен, то функция будет ждать пока он не освободится или пока не истечет указанный таймаут. Количество свободного места в буфере можно при желании узнать с помощью функции `X502_GetSendReadyCount()`.

Возвращение означает, что данные записаны в промежуточный буфер, а не то что они уже дошли до модуля и выведены.

Перед вызовом этой функции должна быть запущена предзагрузка данных на вывод с помощью `X502_PreloadStart()`.

Параметры:

hnd — Описатель модуля.
buf — Буфер со словами, которые необходимо передать модулю
size — Количество передаваемых отсчетов (32-битных слов).
tout — Таймаут на передачу (в буфер драйвера) данных в мс.

Возвращаемое значение:

Если < 0 - код ошибки. Если $>= 0$ - количество записанных слов.

4.3.6.9 Обработка принятых отсчетов АЦП от модуля

Формат: `int32_t X502_ProcessAdcData (t_x502_hnd hnd, const uint32_t *src, double *dest, uint32_t *size, uint32_t flags)`

Описание:

Функция выполняет обработку отсчетов АЦП, прочитанных с помощью `X502_Recv()`. Функция проверяет служебную информацию из входного массива и переводит отсчеты АЦП либо в коды, либо в вольты (если указан флаг `X502_PROC_FLAGS_VOLT`).

Функция используется, когда не запущен синхронный ввод с цифровых линий и отсчеты АЦП являются единственными приходящими от модуля данными (если в принятом потоке будут другие данные - то они будут отброшены).

Если запущен синхронный ввод с цифровых линий, то следует использовать `X502_ProcessData()`, которая выделяет данные с цифровых линий в отдельный массив.

Параметры:

hnd — Описатель модуля.

src — Входной массив отсчетов, принятых с помощью `X502_Recv()`.

dest — Массив, в который будут сохранены преобразованные данные от АЦП.

size — На входе - количество слов в массиве src, на выходе - количество сохраненных преобразованных значений в массиве dest

flags — Набор флагов из `t_x502_proc_flags`

Возвращаемое значение:

Код ошибки.

4.3.6.10 Обработка принятых от модуля данных

Формат: `int32_t X502_ProcessData (t_x502_hnd hnd, const uint32_t *src, uint32_t size, uint32_t flags, double *adc_data, uint32_t *adc_data_size, uint32_t *din_data, uint32_t *din_data_size)`

Описание:

Функция выполняет обработку данных, прочитанных с помощью `X502_Recv()`. Функция проверяет служебную информацию из входного массива, разбивает данные на два массива - данные от АЦП, которые переводятся в тип double, и данные от синхронного цифрового ввода.

Данные от АЦП так же могут быть переведены в вольты. При этом данные АЦП приходят от модуля уже откалиброванными с помощью калибровочных коэффициентов, так как калибровка выполняется аппаратно. Если данные АЦП не переводятся в Вольты и при этом не были изменены заводские калибровочные коэффициенты, то возвращенное значение равное `X502_ADC_SCALE_CODE_MAX` соответствует напряжению равному максимальному для используемого диапазона.

Кроме того, функция разбирает сообщения, передаваемые в потоке данных (например, сообщение о переполнении буфера).

Параметры:

hnd — Описатель модуля.

src — Входной массив отсчетов, принятый с помощью `X502_Recv()`.

size — Количество отсчетов (32-битных слов) в массиве src.

flags — Набор флагов из `t_x502_proc_flags`, управляющих поведением функции. Может быть указано несколько флагов через логическое “ИЛИ”.

adc_data — Массив, в который будут сохранены данные от АЦП, преобразованные в соответствии с указанными флагами. Может быть NULL, если не нужно сохранять данные от АЦП (тогда adc_data_size должен быть тоже NULL, или в переменной передан размер 0).

adc_data_size — На входе в данном параметре передается резмер буфера adc_data. Если данных от АЦП во входном массиве будет больше adc_data_size, то в adc_data будет сохранено только первые adc_data_size отсчетов. На выходе при успешном завершении функции в данную переменную записывается количество сохранных отсчетов АЦП. Указатель может быть равен NULL, если adc_data = NULL

din_data — Массив, в который будут сохранены отчеты с синхронного цифрового ввода. Каждое слово соответствуют состоянию всех цифровых входов в формате, описанном в функции `X502_AsyncInDig()`.

din_data_size — Аналогично параметру adc_data_size в этом параметре передается размер буфера din_data в отсчетах, а на выходе сохраняется количество реально сохраненных отсчетов цифрового ввода. Может быть NULL, если din_data = NULL.

Возвращаемое значение:

Код ошибки.

4.3.6.11 Обработка принятых от модуля данных с пользовательскими данными

Формат: `int32_t X502_ProcessDataWithUserExt (t_x502_hnd hnd, const uint32_t *src, uint32_t size, uint32_t flags, double *adc_data, uint32_t *adc_data_size, uint32_t *din_data, uint32_t *din_data_size, uint32_t *usr_data, uint32_t *usr_data_size)`

Описание:

Функция аналогична `X502_ProcessData()`, но позволяет также выделить пользовательские данные из потока. Пользовательскими данными считаются все отсчеты, которые не являются данными АЦП, данными цифрового ввода или сообщениями. Пользовательские данные складываются без изменений в массив usr_data (если он не равен нулю). Данная функция предназначена в первую очередь для программистов, которые будут использовать модифицированную прошивку сигнального процессора BlackFin.

Параметры:

hnd — Описатель модуля.

src — Входной массив отсчетов, принятый с помощью `X502_Recv()`.

size — Количество отсчетов (32-битных слов) в массиве src.

flags — Набор флагов из `t_x502_proc_flags`.

adc_data — Массив, в который будут сохранены данные от АЦП (см. `X502_ProcessData()`).

adc_data_size — см. `X502_ProcessData()`

din_data — Массив, в который будут сохранены отчеты с синхронного цифрового ввода. См. `X502_ProcessData()`.

din_data_size — см. `X502_ProcessData()`.

usr_data — Массив, в который будут сохранены пользовательские данные без изменения их формата.

usr_data_size — В этом параметре передается размер буфера usr_data а на выходе сохраняется количество реально сохраненных отсчетов пользовательских данных. Может быть NULL только если usr_data = NULL.

Возвращаемое значение:

Код ошибки.

4.3.6.12 Подготовка данных для вывода в модуль

Формат: `int32_t X502_PrepareData (t_x502_hnd hnd, const double *dac1, const double *dac2, const uint32_t *digout, uint32_t size, int32_t flags, uint32_t *out_buf)`

Описание:

Функция принимает данные из трех массивов - данные на цифровые выходы, отсчеты первого и второго канала ЦАП. В качестве массива может быть передан нулевой указатель, если данные из этого источника не требуются. Все используемые массивы должны быть одинакового размера и функция их равномерно перемешивает в общий поток, преобразуя в нужный для модуля формат.

Выходной массив должен будет содержать n*size отсчетов, где n - количество используемых входных массивов (от 1 до 3).

Значения цифровых выходов представляют собой 32-битные слова, младшие 16-бит которых определяют значения выводов, а старшие - флаги из `t_x502_digout_word_flags`, которые могут использоваться в частности для перевода одной (или обеих) из половин выводов в третье состояние.

В качестве значений ЦАП могут использоваться как коды, так и Вольты, в зависимости от переданных флагов, и к выводимым значениям могут быть применены калибровочные коэффициенты. Если используются коды ЦАП с включенной калибровкой, то код `X502_DAC_SCALE_CODE_MAX` определяет код, соответствующий +5V.

Параметры:

hnd — Описатель модуля.

dac1 — Входной массив отсчетов первого канала ЦАП или NULL, если не используется.

dac2 — Входной массив отсчетов второго канала ЦАП или NULL, если не используется.

digout — Входной массив со значениями цифровых выводов или NULL, если не используется.

size — Размер каждого из используемых входных массивов.

flags — Флаги, управляющие работой функции, из `t_x502_dacout_flags`.

out_buf — Выходной массив, в который будут сохранены сформированные отсчеты. Должен быть размера n*size (n - количество используемых входных массивов)

Возвращаемое значение:

Код ошибки.

4.3.6.13 Получить количество отсчетов в буфере потока на ввод

Формат: `int32_t X502_GetRecvReadyCount (t_x502_hnd hnd, uint32_t *rdy_cnt)`
Описание: Функция возвращает количество отсчетов, которые были приняты из модуля во внутренний буфер и готовы для считывания с помощью `X502_Recv()`. То есть если в `X502_Recv()` передать значение, которое вернула данная функция, то `X502_Recv()` вернет это количество данных без ожидания (так как они уже в буфере). При работе по Ethernet данная функция возвращает корректное значение только для ОС Windows.
Параметры: **hnd** — Описатель модуля. **rdy_cnt** — Количество готовых к приему отсчетов.
Возвращаемое значение: Код ошибки.

4.3.6.14 Получить размер свободного места в буфере потока на вывод

Формат: `int32_t X502_GetSendReadyCount (t_x502_hnd hnd, uint32_t *rdy_cnt)`
Описание: Функция возвращает количество отсчетов, соответствующее свободному месту в буфере на передачу в модуль. Это количество отсчетов гарантированно может быть передано с помощью `X502_Send()` без ожидания. Данная функция не реализована при работе по интерфейсу Ethernet (TCP).
Параметры: **hnd** — Описатель модуля. **rdy_cnt** — Количество слов, которому соответствует свободное место в буфере на передачу.
Возвращаемое значение: Код ошибки.

4.3.6.15 Получить номер следующего ожидаемого логического канала АЦП для обработки

Формат: `int32_t X502_GetNextExpectedLchNum (t_x502_hnd hnd, uint32_t *lch)`

Описание:

Функция возвращает номер логического канала АЦП, который должен быть обработан первым при следующем вызове `X502_ProcessData()`/ `X502_ProcessAdcData()` в случае, если поток данных непрерывен.

По сути, это номер логического канала, следующий за логическим каналом последнего обработанного до этого отсчета АЦП. Может быть использовано при обработке блоков данных не кратных целому количеству кадров. Если перед `X502_ProcessData()` вызывать данную функцию, то она вернет номер логического канала, соответствующий первому отсчету АЦП, обработанному последующим вызовом `X502_ProcessData()`.

Например, если установлено 7 логических каналов, а в `X502_ProcessData()` передано для обработки кратное 7 количество отсчетов, то последующий вызов `X502_GetNextExpectedLchNum()` вернет номер канала равный 0 (так как обработано целое число кадров и ожидается снова начало кадра). Если в `X502_ProcessData()` передан массив с 7*n + 5 отсчетами АЦП, то следующим ожидаемым каналом будет логический канал с номером 5 (обработаны каналы 0,1,2,3,4 из неполного кадра).

Параметры:

hnd — Описатель модуля.
lch — Номер логического канала (начиная с нуля).

Возвращаемое значение:

Код ошибки.

4.3.6.16 Начало подготовки вывода синхронных данных

Формат: `int32_t X502_PreloadStart (t_x502_hnd hnd)`

Описание:

Функция должна вызываться перед началом предзагрузки потоковых синхронных данных на вывод. Для начала выдачи синхронных данных одновременно с началом синхронного ввода, к моменту начала сбора часть данных должна быть уже загружена в модуль до вызова `X502_StreamsStart()`.

Данная функция инициализирует канал обмена на передачу данных на вывод. После вызова этой функции можно загрузить часть данных на вывод с помощью `X502_Send()`.

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.6.17 Начало загрузки циклического сигнала на вывод

Формат: `int32_t X502_OutCycleLoadStart (t_x502_hnd hnd, uint32_t size)`

Описание:

По вызову этой функции в драйвере (для L502) или в памяти контроллера модуля (для E502) выделяется место под циклический буфер на вывод. Должна вызываться перед загрузкой циклических данных с помощью `X502_Send()`.

Для успешного выполнения должен быть свободный буфер (используется двойная буферизация) - т.е. функция не может быть вызвана сразу после предыдущего `X502_OutCycleSetup()`. Кроме того не должен был быть использован потоковый вывод.

Для L-502 размер максимальный размер буфера определяется только размером, который позволяет выделить система на уровне драйвера. Для E-502 размер ограничен памятью встроенного контроллера. Подробнее смотри в описании различий E-502 и L-502.

Параметры:

hnd — Описатель модуля.

size — Количество отсчетов в выводимом циклическом сигнале суммарно для всех используемых каналов вывода.

Возвращаемое значение:

Код ошибки.

4.3.6.18 Установка ранее загруженного циклического сигнала на вывод

Формат: `int32_t X502_OutCycleSetup (t_x502_hnd hnd, uint32_t flags)`

Описание:

По вызову этой функции ранее загруженный циклический буфер становится активным. Если синхронный ввод-вывод запущен (через `X502_StreamsStart()`), то по этой функции сигнал будет выдаваться на выход, иначе выдача начнется при запуске синхронного ввода-вывода.

Если до этого уже выводился циклический сигнал, то смена на новый произойдет в конце цикла предыдущего сигнала, если не указан флаг `X502_OUT_CYCLE_FLAGS_FORCE`.

Если не указан флаг `X502_OUT_CYCLE_FLAGS_WAIT_DONE`, то функция только дает команду на установку сигнала, не ожидая непосредственной смены сигнала или загрузки сигнала. В частности для одновременного запуска синхронного ввода и вывода необходимо сделать загрузку первого циклического сигнала с данным флагом, чтобы гарантировать, что сигнал полностью загружен к моменту запуска синхронного ввода-вывода через `X502_StreamsStart()`.

Данная функция должна быть вызвана только после вызова `X502_OutCycleLoadStart()` и загрузки указанного в ней количества отсчетов в буфер!

Параметры:

hnd — Описатель модуля.

flags — Флаги из `t_x502_out_cycle_flags`.

Возвращаемое значение:

Код ошибки.

4.3.6.19 Останов вывода циклического сигнала

Формат: `int32_t X502_OutCycleStop (t_x502_hnd hnd, uint32_t flags)`

Описание:

По вызову этой функции прекращается выдача ранее установленного циклического сигнала с помощью `X502_OutCycleSetup()`. Остановка осуществляется после выдачи последнего отсчета в периоде, что позволяет знать какие значения останутся на выходах.

При вызове же `X502_StreamsStop()` (или при запрещении всех потоков на вывод через `X502_StreamsDisable()`) останов всех потоков происходит сразу и точная точка останова неизвестна.

При этом необходимо учитывать, что сама функция по умолчанию только делает запрос на останов, а реально останов произойдет позже. Если вызвать `X502_StreamsStop()` до завершения останова, то последний отсчет будет неизвестен, т.е. необходимо дождаться завершения останова, для чего может например быть использован флаг `X502_OUT_CYCLE_FLAGS_WAIT_DONE`.

Параметры:

hnd — Описатель модуля.
flags — Флаги из `t_x502_out_cycle_flags`.

Возвращаемое значение:

Код ошибки.

4.3.6.20 Проверка, завершена ли установка или останов циклического сигнала

Формат: `int32_t X502_OutCycleCheckSetupDone (t_x502_hnd hnd, uint32_t *done)`

Описание:

Функция проверяет, завершена ли установка циклического сигнала после вызова `X502_OutCycleSetup()` или завершена ли остановка генерации циклического сигнала после вызова `X502_OutCycleStop()`. По своему назначению аналогична флагу `X502_OUT_CYCLE_FLAGS_WAIT_DONE` в описанных выше функциях, но позволяет выполнить ожидание вручную (с проверкой других условий).

Функция доступна в библиотеки, начиная с версии 1.1.2, при этом для работы функции необходима версия прошивки ARM не ниже 1.0.2 для модуля E-502 или версия драйвера не ниже 1.0.9 для L-502. В отличие от флага, если данные условия не выполняются, то функция явно вернет ошибку `X502_ERR_NOT_SUP_BY_FIRMWARE` или `X502_ERR_NOT_SUP_BY_DRIVER`.

Ожидание завершения может быть необходимо при вызове `X502_OutCycleSetup()` при загрузке первого сигнала до вызова `X502_StreamsStart()`, если требуется, чтобы выдача первых отсчетов на ЦАП совпала по времени с моментом запуска ввода, т.к. иначе загрузка сигнала в модуль может не завершиться к моменту запуска и выдача сигнала начнется с задержкой (актуально для E502).

При последующих вызовах `X502_OutCycleSetup()` для смены уже установленного сигнала установка считается завершенной после завершения загрузки сигнала и непосредственной смены выдаваемого сигнала. Эту проверку можно использовать, чтобы явно узнать, что смена сигнала завершилась и можно уже загружать следующий циклический сигнал.

При вызове `X502_OutCycleStop()` ожидание завершения может использоваться перед вызовом `X502_StreamsStop()`, чтобы убедится (если это необходимо), что генерация была завершена именно на последней точке загруженного циклического сигнала.

Параметры:

hnd — Описатель модуля.

done — 0, если есть незавершенный запрос на установку или останов циклического сигнала, 1 — в противном случае (включая случай, когда выдача циклического сигнала вообще не ведется)

Возвращаемое значение:

Код ошибки.

4.3.6.21 Чтение флагов статуса вывода

Формат: `int32_t X502_OutGetStatusFlags (t_x502_hnd hnd, uint32_t *status)`

Описание:

Функция читает значения флагов статуса синхронного вывода из регистра статуса. В частности по флагу `X502_OUT_STATUS_FLAG_BUF_WAS_EMPTY` можно проверить, не было ли опустошения буфера с момента запуска синхронного вывода, чтобы убедиться, что не было разрыва сигнала из-за неподкаченных вовремя данных.

Параметры:

hnd — Описатель модуля.

status — Флаги статуса — набор битов из `t_x502_out_status_flags`, объединенных через логическое “ИЛИ”.

Возвращаемое значение:

Код ошибки.

4.3.6.22 Установка размера буфера для синхронного ввода или вывода

Формат: `int32_t X502_SetStreamBufSize (t_x502_hnd hnd, uint32_t ch, uint32_t size)`

Описание:

Функция устанавливает размер буфера, который используется для временного хранения данных на прием или на передачу. Предназначена для случаев, когда пользователя по каким-либо причинам не удовлетворяет рассчитываемое библиотекой значение по умолчанию.

Если функцией X502_SetStreamBufSize был установлен не нулевой размер буфера, то этот размер будет использоваться начиная с первого вызова X502_StreamsStart или X502_StreamsEnable и пересчитываться далее не будет, чтобы библиотека сама рассчитала размер буфера надо вызвать X502_SetStreamBufSize с нулевым размером и он рассчитается при первом X502_StreamsStart или X502_StreamsEnable.

Параметры:

hnd — Описатель модуля.

ch — Определяет, устанавливается размер буфера на ввод или на вывод (значение из `t_x502_stream_ch`).

size — Размер буфера в 32-битных отсчетах, 0 - библиотека будет сама рассчитывать значение по умолчанию

Возвращаемое значение:

Код ошибки.

4.3.6.23 Установка шага при передаче потока на ввод или вывод

Формат: `int32_t X502_SetStreamStep (t_x502_hnd hnd, uint32_t dma_ch, uint32_t step)`
Описание: Функция устанавливает шаг передачи данных (шаг генерации прерываний для PCI-Express или размер запроса для USB) при передаче синхронного потока данных на ввод или на вывод. Данная функция предназначена для пользователей, которых не устроит автоматически рассчитываемое библиотекой значение.
Параметры: **hnd** — Описатель модуля. **dma_ch** — Определяет, шаг передачи устанавливается на ввод или на вывод (значение из `t_x502_stream_ch`). **step** — Шаг прерывания в 32-битных отсчетах
Возвращаемое значение: Код ошибки.

4.3.7 Функции для настройки сетевых параметров модуля E502

4.3.7.1 Получение текущего IP-адреса устройства

Формат: `int32_t E502_GetIpAddr (t_x502_hnd hnd, uint32_t *ip_addr)`
Описание: Функция возвращает IP-адрес устройства по которому было установлено соединение. То есть связь с устройством должна быть уже установлена и кроме того, установлена именно по интерфейсу Ethernet.
Параметры: **hnd** — Описатель устройства **ip_addr** — Текущий IPv4 адрес модуля в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`).
Возвращаемое значение: Код ошибки

4.3.7.2 Создание описателя конфигурации сетевого интерфейса

Формат: `t_e502_eth_config_hnd E502_EthConfigCreate (void)`
Описание: Создание описателя конфигурации сетевого интерфейса. В случае успешного выделения памяти инициализирует поля описателя значениями по-умолчанию.
Возвращаемое значение: NULL в случае ошибки, иначе - описатель модуля

4.3.7.3 Освобождение описателя конфигурации сетевого интерфейса.

Формат: `int32_t E502_EthConfigFree (t_e502_eth_config_hnd cfg)`
Описание: Освобождение памяти, выделенной под описатель конфигурации сетевого интерфейса с помощью `E502_EthConfigCreate()`. После этого описатель уже использовать нельзя, независимо от возвращенного значения!
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса
Возвращаемое значение: Код ошибки

4.3.7.4 Чтение текущей сетевой конфигурации интерфейса

Формат: `int32_t E502_EthConfigRead (t_x502_hnd hnd, t_e502_eth_config_hnd cfg)`
Описание: Функция читает текущие параметры интерфейса и сохраняет их в структуру, на которую указывает созданный с помощью `E502_EthConfigCreate()` описатель конфигурации сетевого интерфейса. Соединение с устройством при этом должно быть установлено, но может быть установлено по любому поддерживаемому интерфейсу.
Параметры: **hnd** — Описатель устройства из которого нужно считать конфигурацию **cfg** — Описатель конфигурации сетевого интерфейса
Возвращаемое значение: Код ошибки

4.3.7.5 Запись сетевой конфигурации интерфейса

Формат: `int32_t E502_EthConfigWrite (t_x502_hnd hnd, t_e502_eth_config_hnd cfg, const char *passwd)`

Описание:

Функция передает модулю конфигурацию сетевого интерфейса, которую модуль должен сохранить в своей энергонезависимой памяти.

При успешном выполнении данной функции модуль отключает Ethernet-интерфейс, настраивает его на новые параметры и снова его инициализирует, поэтому если соединение с устройством установлено по сети, то дальнейшая работа с устройством будет уже не возможна — необходимо закрыть связь с устройством и установить ее заново.

Для изменения конфигурации необходимо передать пароль для конфигурации (пустая строка, если пароль не был установлен). При работе по USB интерфейсу в качестве пароля можно передать текущий серийный номер устройства (для случая, если забыт установленный пароль).

Параметры:

hnd — Описатель устройства из которого нужно считать конфигурацию
cfg — Описатель конфигурации сетевого интерфейса
passwd — Строка с паролем для изменения конфигурации

Возвращаемое значение:

Код ошибки

4.3.7.6 Копирование содержимого сетевой конфигурации интерфейса

Формат: `int32_t E502_EthConfigCopy (t_e502_eth_config_hnd src_cfg, t_e502_eth_config_hnd dst_cfg)`

Описание:

Функция выполняет копирование всех параметров одной созданной конфигурации в другую конфирурацию, создавая полную копию.

Параметры:

src_cfg — Описатель исходной сетевой конфигурации интерфейса, содержимое которой нужно скопировать.
dst_cfg — Описатель сетевой конфигурации интерфейса, в которую нужно скопировать содержимое исходной конфигурации

Возвращаемое значение:

Код ошибки

4.3.7.7 Определение, разрешен ли интерфейс Ethernet.

Формат: `int32_t E502_EthConfigGetEnabled (t_e502_eth_config_hnd cfg, uint32_t *en)`
Описание: Функция возвращает, разрешен ли интерфейс Ethernet в указанной конфигурации. Если интерфейс не разрешен, то Ethernet контроллер полностью отключен.
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса **en** — Если интерфейс разрешен, то в данной переменной возвращается 1, иначе — 0
Возвращаемое значение: Код ошибки

4.3.7.8 Разрешение интерфейса Ethernet.

Формат: `int32_t E502_EthConfigSetEnabled (t_e502_eth_config_hnd cfg, uint32_t en)`
Описание: Функция устанавливает, разрешена ли работа по интерфейсу Ethernet. Если интерфейс не разрешен, то Ethernet контроллер полностью отключен.
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса **en** — 0 означает запрет интерфейса Ethernet, 1 — разрешение
Возвращаемое значение: Код ошибки

4.3.7.9 Определение, разрешено ли автоматическое получение параметров IP.

Формат: `int32_t E502_EthConfigGetAutoIPEnabled (t_e502_eth_config_hnd cfg, uint32_t *en)`
Описание: Функция возвращает, разрешено ли автоматическое получение параметров IP (IP-адрес, маска подсети, адрес шлюза) с использованием DHCP/linklocal или используются статически заданные параметры.
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса **en** — Если разрешено автоматическое получение параметров, то возвращается 1, иначе — 0
Возвращаемое значение: Код ошибки

4.3.7.10 Разрешение автоматического получения параметров IP.

Формат: `int32_t E502_EthConfigSetAutoIPEnabled (t_e502_eth_config_hnd cfg, uint32_t en)`
Описание: Функция устанавливает, разрешено ли автоматическое получение параметров IP (IP-адрес, маска подсети, адрес шлюза) с использованием DHCP/linklocal или используются статически заданные параметры.
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса **en** — Если разрешено автоматическое получение параметров, то возвращается 1, иначе — 0
Возвращаемое значение: Код ошибки

4.3.7.11 Получить состояние автоматического получения параметров IP.

Формат: `int32_t E502_EthConfigGetAutoIPState (t_e502_eth_config_hnd cfg, uint32_t *state)`
Описание: Функция возвращает, получил ли модуль параметры IP (IP-адрес, маска подсети, адрес шлюза) с использованием DHCP/linklocal
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса **state** — Состояние автоматического получения параметров IP
Возвращаемое значение: Код ошибки

4.3.7.12 Определение, разрешен ли пользовательский MAC-адрес

Формат: `int32_t E502_EthConfigGetUserMACEnabled (t_e502_eth_config_hnd cfg, uint32_t *en)`
Описание: Функция возвращает, разрешен ли MAC-адрес, заданный пользователем, или используется заводской MAC-адрес.
Параметры: **cfg** — Описатель конфигурации сетевого интерфейса. **en** — Если разрешен пользовательский MAC-адрес, то возвращается 1, иначе (если используется заводской) — 0
Возвращаемое значение: Код ошибки

4.3.7.13 Определение, разрешен ли пользовательский MAC-адрес

Формат: `int32_t E502_EthConfigSetUserMACEnabled (t_e502_eth_config_hnd cfg, uint32_t en)`

Описание:
Функция возвращает, разрешен ли MAC-адрес, заданный пользователем, или используется заводской MAC-адрес.

Параметры:
cfg — Описатель конфигурации сетевого интерфейса.
en — Если разрешен пользовательский MAC-адрес, то возвращается 1, иначе (если используется заводской) — 0

Возвращаемое значение:
Код ошибки

4.3.7.14 Получение установленного статического IP-адреса

Формат: `int32_t E502_EthConfigGetIPv4Addr (t_e502_eth_config_hnd cfg, uint32_t *ip_addr)`

Описание:
Функция возвращает заданный в конфигурации статический IP-адрес, который используется устройством, если запрещено автоматическое получение IP-параметров.

Параметры:
cfg — Описатель конфигурации сетевого интерфейса.
ip_addr — Заданный IP-адрес в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`).

Возвращаемое значение:
Код ошибки

4.3.7.15 Установка статического IP-адреса

Формат: `int32_t E502_EthConfigSetIPv4Addr (t_e502_eth_config_hnd cfg, uint32_t ip_addr)`

Описание:
Функция устанавливает в конфигурации заданный статический IP-адрес, который будет использоваться устройством, если запрещено автоматическое получение IP-параметров.

Параметры:
cfg — Описатель конфигурации сетевого интерфейса.
ip_addr — Устанавливаемый IP-адрес в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`).

Возвращаемое значение:
Код ошибки

4.3.7.16 Получение установленной статической маски подсети

Формат: `int32_t E502_EthConfigGetIPv4Mask (t_e502_eth_config_hnd cfg, uint32_t *mask)`

Описание:

Функция возвращает заданное в конфигурации значение маски подсети, которая используется устройством, если запрещено автоматическое получение IP-параметров.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

mask — Маска подсети в виде 32-битного слова (аналогично параметру ip_addr функции E502_OpenByIpAddr()).

Возвращаемое значение:

Код ошибки

4.3.7.17 Установка статической маски подсети

Формат: `int32_t E502_EthConfigSetIPv4Mask (t_e502_eth_config_hnd cfg, uint32_t mask)`

Описание:

Функция устанавливает в конфигурации значение маски подсети, которая будет использоваться устройством, если запрещено автоматическое получение IP-параметров.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

mask — Устанавливаемое значение маски подсети в виде 32-битного слова (аналогично параметру ip_addr функции E502_OpenByIpAddr()).

Возвращаемое значение:

Код ошибки

4.3.7.18 Получение установленного статического адреса шлюза

Формат: `int32_t E502_EthConfigGetIPv4Gate (t_e502_eth_config_hnd cfg, uint32_t *gate)`

Описание:

Функция возвращает заданное в конфигурации значение адреса шлюза, который используется устройством, если запрещено автоматическое получение IP-параметров.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

gate — Адрес шлюза в виде 32-битного слова (аналогично параметру ip_addr функции E502_OpenByIpAddr()).

Возвращаемое значение:

Код ошибки

4.3.7.19 Установка статического адреса шлюза

Формат: `int32_t E502_EthConfigSetIPv4Gate (t_e502_eth_config_hnd cfg, uint32_t gate)`

Описание:

Функция устанавливает в конфигурации значение адреса шлюза, который будет использоваться устройством, если запрещено автоматическое получение IP-параметров.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

gate — Устанавливаемое значение адреса шлюза в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`).

Возвращаемое значение:

Код ошибки

4.3.7.20 Получение установленного пользовательского MAC-адреса

Формат: `int32_t E502_EthConfigGetUserMac (t_e502_eth_config_hnd cfg, uint8_t *mac)`

Описание:

Функция возвращает заданное в конфигурации значение пользовательского MAC-адреса, который используется устройством при явном его разрешении (см. `E502_EthConfigSetUserMACEnabled()`).

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

mac — Пользовательский MAC-адрес устройства в виде массива из `X502_MAC_ADDR_SIZE` байт в порядке записи адреса

Возвращаемое значение:

Код ошибки

4.3.7.21 Установка пользовательского MAC-адреса

Формат: `int32_t E502_EthConfigSetUserMac (t_e502_eth_config_hnd cfg, const uint8_t *mac)`

Описание:

Функция устанавливает в конфигурации значение пользовательского MAC-адреса, который будет использоваться устройством при явном его разрешении (см. `E502_EthConfigSetUserMACEnabled()`).

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

mac — Устанавливаемое значение пользовательского MAC-адрес устройства в виде массива из `X502_MAC_ADDR_SIZE` байт в порядке записи адреса

Возвращаемое значение:

Код ошибки

4.3.7.22 Получение заводского MAC-адреса устройства

Формат: `int32_t E502_EthConfigGetFactoryMac (t_e502_eth_config_hnd cfg, uint8_t *mac)`

Описание:

Функция возвращает значение заводского MAC-адреса устройства, которому соответствует переданная первым параметром конфигурация. Заводской MAC-адрес, используемый устройством по-умолчанию, записывается производителем (в "Л Кард") при производстве устройства вместе с его серийным номером и не может быть изменен пользователем. Если пользователю нужно изменить MAC-адрес устройства, то он должен задать пользовательский MAC-адрес с помощью `E502_EthConfigGetUserMac()` и разрешить его использование через `E502_EthConfigSetUserMACEnabled()`. При этом всегда есть возможность снова вернуться к использованию оригинального заводского MAC-адреса.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

mac — Заводской MAC-адрес устройства в виде массива из `X502_MAC_ADDR_SIZE` байт в порядке записи адреса

Возвращаемое значение:

Код ошибки

4.3.7.23 Получение установленного имени экземпляра устройства

Формат: `int32_t E502_EthConfigGetInstanceName (t_e502_eth_config_hnd cfg, char *name)`

Описание:

Функция возвращает заданное пользователем имя экземпляра устройства. Данное имя может использоваться для обнаружения устройства в сети. Если не задано, то используется имя, образованное названием устройства и его серийным номером. Данное имя должно быть уникально в пределах сети.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

name — Оканчивающаяся нулем строка с заданным именем экземпляра устройства в формате UTF-8. Массив должен быть рассчитан на `X502_INSTANCE_NAME_SIZE` байт данных.

Возвращаемое значение:

Код ошибки.

4.3.7.24 Установка имени экземпляра устройства

Формат: `int32_t E502_EthConfigSetInstanceName (t_e502_eth_config_hnd cfg, const char *name)`

Описание:

Функция устанавливает имя экземпляра устройства, которое может использоваться для обнаружения устройства локальной в сети.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

name — Оканчивающаяся нулем строка с заданным именем экземпляра устройства в формате UTF-8. Максимальный размер массива (включая завершающий ноль) составляет `X502_INSTANCE_NAME_SIZE` байт данных.

Возвращаемое значение:

Код ошибки.

4.3.7.25 Установка нового пароля для смены конфигурации

Формат: `int32_t E502_EthConfigSetNewPassword (t_e502_eth_config_hnd cfg, const char *new_passwd)`

Описание:

Функция устанавливает новое значение пароля, которое должно будет использоваться для смены конфигурации через `E502_EthConfigWrite()`.

При этом значение при сохранении конфигурации с установленным новым паролем необходимо для успешной смены конфигурации в `E502_EthConfigWrite()` передать значение пароля, которое было установлено до этого. Если функция завершится успешно, то для последующего изменения конфигурации в `E502_EthConfigWrite()` нужно будет передавать уже новое установленное значение пароля.

Параметры:

cfg — Описатель конфигурации сетевого интерфейса.

new_passwd — Оканчивающаяся нулем строка, содержащая новое значение пароля для смены конфигурации сетевого интерфейса. Максимальный размер массива (включая завершающий ноль) составляет `X502_PASSWORD_SIZE` байт данных.

Возвращаемое значение:

Код ошибки.

4.3.8 Функции для поиска модулей в локальной сети

4.3.8.1 Начало сеанса поиска модулей в локальной сети

Формат: `int32_t E502_EthSvcBrowseStart (t_e502_eth_svc_browse_hnd *pcontext, uint32_t flags)`

Описание:

При вызове данной функции запускается процесс поиска сервисов, соответствующих модулям E-502, в локальной сети и создается контекст текущего сеанса поиска. Этот контекст используется для дальнейших вызовов `E502_EthSvcBrowseGetEvent()`. После завершения поиска должна быть вызвана функция `E502_EthSvcBrowseStop()`. Для запуска сеанса необходима запущенная служба (демон) обнаружения — поддерживаются Bonjour для ОС Windows и Avahi для ОС Linux.

Параметры:

pcontext — Указатель, в который при успешном выполнении сохраняется контекст сеанса поиска устройств.

flags — Флаги (резерв). Должен всегда передаваться 0.

Возвращаемое значение:

Код ошибки.

4.3.8.2 Получение информации о изменении присутствия модулей в локальной сети

Формат: `int32_t E502_EthSvcBrowseGetEvent (t_e502_eth_svc_browse_hnd context, t_e502_eth_svc_record_hnd *svc, uint32_t *event, uint32_t *flags, uint32_t tout)`

Описание:

Данная функция позволяет как получить список присутствующих модулей (сетевых сервисов) в локальной сети, так и отслеживать в дальнейшем изменение их состояния.

Функция ждет первого изменения состояния и возвращает информацию о нем. Информация состоит из события (появление сетевого сервиса, исчезновение, изменение параметров) и из описателя сетевого сервиса, которому соответствует событие.

После начала поиска с помощью `E502_EthSvcBrowseStart()` контекст не содержит информации о наличие сетевых сервисов. Если уже есть подключенные в локальной сети модули E-502, то информация о них будет возвращена в следующих `E502_EthSvcBrowseGetEvent()` с событием `E502_ETH_SVC_EVENT_ADD`, за каждый вызов по одному устройству.

Если за заданный таймаут не произошло ни одного изменения, то функция завершится успешно по окончанию таймаута и вернет событие `E502_ETH_SVC_EVENT_NONE`.

При желании можно продолжать вызвать данную функцию для непрерывного отслеживания состояния подключения модулей.

Параметры:

context — Описатель контекста поиска, созданный при вызове `E502_EthSvcBrowseStart()`.

svc — Если возвращенное событие не равно `E502_ETH_SVC_EVENT_NONE`, то в данной переменной сохраняется созданный описатель сетевого сервиса, соответствующего указанному событию. Этот описатель должен быть всегда уничтожен вручную с помощью `E502_EthSvcRecordFree()`.

event — В данную переменную сохраняется код события (один из `t_e502_eth_svc_event`). Если за указанное время не произошло ни одного события, то возвращается код `E502_ETH_SVC_EVENT_NONE`.

flags — В данной переменной сохраняются дополнительные коды флагов (резерв). Может быть передан нулевой указатель, если значение флагов не интересует.

tout — Таймаут (в мс) на время ожидания события

Возвращаемое значение:

Код ошибки.

4.3.8.3 Останов сеанса поиска модулей в локальной сети

Формат: `int32_t E502_EthSvcBrowseStop (t_e502_eth_svc_browse_hnd context)`

Описание:

При вызове данной функции процесс поиска сетевых сервисов, соответствующий указанному контексту, останавливается. Все ресурсы, выделенные на этапе `E502_EthSvcBrowseStart()` освобождаются. Контекст с этого момента становится недействительным. Вызову `E502_EthSvcBrowseStart()` всегда должен соответствовать последующий вызов `E502_EthSvcBrowseStop()` для корректного освобождения ресурсов.

Параметры:

context — Описатель контекста поиска, созданный при вызове `E502_EthSvcBrowseStart()`.

Возвращаемое значение:

Код ошибки.

4.3.8.4 Освобождение описателя сетевого сервиса

Формат: `int32_t E502_EthSvcRecordFree (t_e502_eth_svc_record_hnd svc)`

Описание:

Освобождение памяти, выделенной под описатель сетевого сервиса при вызове `E502_EthSvcBrowseGetEvent()`.

Параметры:

svc — Описатель сетевого сервиса

Возвращаемое значение:

Код ошибки

4.3.8.5 Получить имя экземпляра по описателю сервиса

Формат: `int32_t E502_EthSvcRecordGetInstanceName (t_e502_eth_svc_record_hnd svc, char *name)`

Описание:

Функция возвращает имя экземпляра сервиса. Это имя соответствует имени, которое установлено в сетевых настройках модуля, соответствующего указанному сервису, с помощью `E502_EthConfigSetInstanceName()`. Следует отметить, что данное имя, в отличие от остальных строк, представлено в кодировке UTF-8, которая совпадает с обычной ASCII строкой только для символов английского алфавита. Функция не выполняет запросов к самому модулю.

Параметры:

svc — Описатель сетевого сервиса

name — Массив на `X502_INSTANCE_NAME_SIZE` байт, в который будет сохранено название экземпляра

Возвращаемое значение:

Код ошибки

4.3.8.6 Получить серийный номер модуля по описателю сетевого сервиса

Формат: `int32_t E502_EthSvcRecordGetDevSerial (t_e502_eth_svc_record_hnd svc, char *serial)`

Описание:

Функция возвращает серийный номер модуля E-502, соответствующего сетевому сервису, на который указывает переданный описатель. Функция не выполняет запросов к самому модулю.

Параметры:

svc — Описатель сетевого сервиса

serial — Массив на `X502_SERIAL_SIZE` байт, в который будет сохранен серийный номер

Возвращаемое значение:

Код ошибки

4.3.8.7 Получить имя усройства модуля по описателю сетевого сервиса

Формат: `int32_t E502_EthSvcRecordGetDevName (t_e502_eth_svc_record_hnd rec, char *devname)`

Описание:

Функция возвращает имя устройства модуля (“E502” или “E16”), соответствующего сетевому сервису, на который указывает переданный описатель. Функция не выполняет запросов к самому модулю.

Параметры:

svc — Описатель сетевого сервиса

devname — Массив на `X502_DEVNAME_SIZE` байт, в который будет сохранено имя устройства

Возвращаемое значение:

Код ошибки

4.3.8.8 Получить IP адрес сетевого сервиса

Формат: `int32_t E502_EthSvcRecordResolveIPv4Addr (t_e502_eth_svc_record_hnd svc, uint32_t *addr, uint32_t tout)`

Описание:

Функция получает IP-адрес модуля E-502, соответствующего сетевому сервису, на который указывает переданный описатель. Функция при необходимости может выполнять запросы к самому модулю для получения этого адреса, если информации о адресе нет в кеше.

Параметры:

svc — Описатель сетевого сервиса

addr — IP-адрес модуля в виде 32-битного слова (аналогично параметру ip_addr функции `E502_OpenByIpAddr()`)

tout — Время ожидания ответа от модуля в случае необходимости выполнить запрос для установления адреса.

Возвращаемое значение:

Код ошибки

4.3.8.9 Проверка, указывают ли оба описателя на один экземпляр сервиса

Формат: `int32_t E502_EthSvcRecordIsSameInstance (t_e502_eth_svc_record_hnd svc1, t_e502_eth_svc_record_hnd svc2)`
Описание: Функция проверяет, указывают ли оба описателя сервисов на один и тот же экземпляр. Если приложение сохраняет список описателей сервисов при их обнаружении, то данная функция может использоваться, например, при событиях `E502_ETH_SVC_EVENT_REMOVE` или `E502_ETH_SVC_EVENT_CHANGED`, чтобы понять, какой записи в сохраненном списке соответствует событие (т.е. функция `E502_EthSvcBrowseGetEvent()` вернет новый описатель, но указывающий на тот же экземпляр, что и при событии `E502_ETH_SVC_EVENT_ADD`)
Параметры: **svc1** — Первый описатель сетевого сервиса для сравнения **svc2** — Второй описатель сетевого сервиса для сравнения
Возвращаемое значение: Код ошибки. Возвращает `X502_ERR_OK`, если оба описателя указывают на один экземпляр.

4.3.9 Функции для работы с сигнальным процессором

4.3.9.1 Загрузка прошивки сигнального процессора BlackFin.

Формат: `int32_t X502_BfLoadFirmware (t_x502_hnd hnd, const char *filename)`
Описание: Функция загружает прошивку сигнального процессора из указанного файла в процессор и запускает ее, проверяет правильность загрузки путем получения версии прошивки (через специальную команду). Прошивка должна быть в бинарном формате LDR.
Параметры: **hnd** — Описатель модуля. **filename** — Имя файла с загружаемой прошивкой.
Возвращаемое значение: Код ошибки.

4.3.9.2 Проверка, загружена ли прошивка BlackFin.

Формат: `int32_t X502_BfCheckFirmwareIsLoaded (t_x502_hnd hnd, uint32_t *version)`

Описание:

Функция передает команды процессору BlackFin для получения версии прошивки и ее состояния. Успешное выполнение команд свидетельствует о том, что в BlackFin загружена действительная прошивка. Кроме того прошивке передается информация о модуле (наличие опций, версия ПЛИС и т.д.) для внутреннего использования. В случае успеха модуль переводится в режим DSP.

Данная функция может служить для проверки, была ли загружена прошивка раньше (чтобы не загружать повторно) или для проверки была ли она загружена через JTAG-интерфейс.

Параметры:

hnd — Описатель модуля.

version — Если указатель не нулевой, то в данной переменной возвращается версия прошивки BlackFin в случае успешной проверки.

Возвращаемое значение:

Код ошибки.

4.3.9.3 Чтение блока данных из памяти сигнального процессора

Формат: `int32_t X502_BfMemRead (t_x502_hnd hnd, uint32_t addr, uint32_t *regs, uint32_t size)`

Описание:

Функция считывает блок данных напрямую из памяти процессора. Может быть прочитаны данные, как из внутренней памяти (L1), так и из внешней SDRAM. Для выполнения этой функции в BlackFin должна быть загружена его прошивка.

Функция предназначена в первую очередь для пользователей, пишущих свою программу для сигнального процессора.

Параметры:

hnd — Описатель модуля.

addr — Адрес памяти, начиная с которого будет считан блок данных.

regs — Массив, в который будут сохранено прочитанное содержимое памяти.

size — Количество считываемых 32-битных слов.

Возвращаемое значение:

Код ошибки.

4.3.9.4 Запись блока данных в память сигнального процессора

Формат: `int32_t X502_BfMemWrite (t_x502_hnd hnd, uint32_t addr, const uint32_t *regs, uint32_t size)`

Описание:

Функция записывает блок данных напрямую в памяти процессора BlackFin. Блок данных должен быть всегда кратен 8 32-битным словам (32 байтам). Запись может осуществляться как во внутреннюю память (L1), так и во внешнюю SDRAM. Для выполнения этой функции в BlackFin должна быть загружена его прошивка.

Функция предназначена в первую очередь для пользователей, пишущих свою программу для сигнального процессора.

Следует быть осторожным, т.к. запись в область данных, используемую программой может привести к ее неработоспособности.

Параметры:

hnd — Описатель модуля.

addr — Адрес памяти, начиная с которого будет записан блок данных.

regs — Массив с данными для записи в сигнальный процессор.

size — Количество записываемых данных в 32-битных словах (должно быть кратно 8).

Возвращаемое значение:

Код ошибки.

4.3.9.5 Передача управляющей команды сигнальному процессору

Формат: `int32_t X502_BfExecCmd (t_x502_hnd hnd, uint16_t cmd_code, uint32_t par, const uint32_t *snd_data, uint32_t snd_size, uint32_t *rcv_data, uint32_t rcv_size, uint32_t tout, uint32_t *recvd_size)`

Описание:

Функция предназначена для передачи пользовательских управляющих команд процессору для пользователей, пишущих свою прошивку BlackFin.

Управление работой сигнального процессора штатным образом осуществляется через управляющие команды, которые записываются в специальную область памяти сигнального процессора. Сигнальный процессор обрабатывает команду и по завершению записывает в эту же область результат.

Команды деляться на стандартные, которые используются библиотекой x502api и реализованы в штатной прошивке сигнального процессора и пользовательские, которые пользователь может определять по своему усмотрению. Пользовательские команды начинаются с кода X502_BF_CMD_CODE_USER (0x8000).

Параметры:

hnd — Описатель модуля.

cmd_code — Код команды - определяет, что за команда выполняется.

par — Параметр, передаваемый с командой (значение зависит от кода команды).

snd_data — Опциональные данные, передаваемые вместе с командой. Если данные не передаются, то должен передаваться нулевой указатель и snd_size = 0.

snd_size — Количество 32-битных слов, передаваемых в snd_data

rcv_data — Массив, в который будут переданы данные, возвращенные процессором по завершению команды. Если данные не должны возвращаться, то должен передаваться нулевой указатель, а rcv_size = 0.

rcv_size — Количество 32-битных слов, которое ожидается, что вернет процессор по выполнению команды. Массив rcv_data должен быть рассчитан на данное количество слов.

tout — Таймаут в течении которого будет ожидаться, когда процессор завершит выполнение команды. Функция возвратит управление либо по завершению команды, либо по таймауту.

recvd_size — Если не является нулевым указателем, то в эту переменную будет сохранено количество 32-битных слов, которое реально вернул процессор после выполнения команды (процессор имеет право вернуть меньше данных, чем запрашивалось в rcv_size).

Возвращаемое значение:

Код ошибки. Если процессор выполнил команду с ненулевым кодом завершения, то этот код и будет возвращен функцией.

4.3.10 Функции для работы с Flash-памятью модуля

4.3.10.1 Чтение блока данных из Flash-памяти

Формат: `int32_t X502_FlashRead (t_x502_hnd hnd, uint32_t addr, uint8_t *data, uint32_t size)`

Описание:

Функция считывает массив данных из Flash-памяти модуля в массив, переданный пользователем. Для считывания не нужно специальное разрешение - оно доступно всегда.

Параметры:

hnd — Описатель модуля.
addr — Адрес начала блока.
data — Массив, куда будут сохранены считанные данные (должен быть не меньше size байт).
size — Количество байт для чтения.

Возвращаемое значение:

Код ошибки.

4.3.10.2 Запись блока данных во Flash-память модуля

Формат: `int32_t X502_FlashWrite (t_x502_hnd hnd, uint32_t addr, const uint8_t *data, uint32_t size)`

Описание:

Функция записывает переданный массив данных во Flash-память модуля. Эта область должна быть предварительно стерта с помощью `X502_FlashErase()` и до начала изменения должна быть вызвана функция `X502_FlashWriteEnable()`, чтобы разрешить любое изменение содержимого Flash-памяти. Пользователю для записи доступны только первые `X502_FLASH_USER_SIZE` байт Flash-памяти.

Параметры:

hnd — Описатель модуля.
addr — Адрес начала блока.
data — Массив с записываемыми данными (должен быть не меньше size байт).
size — Количество байт для записи.

Возвращаемое значение:

Код ошибки.

4.3.10.3 Стирание блока во Flash-памяти

Формат: `int32_t X502_FlashErase (t_x502_hnd hnd, uint32_t addr, uint32_t size)`

Описание:

Функция стирает блок во Flash-памяти модуля (все ячейки будут читаться как 0xFF). Адрес и размер должны быть кратны 4096 байт! Перед вызовом этой функции должна быть разрешена запись в пользовательскую область с помощью `X502_FlashWriteEnable()`.

Параметры:

hnd — Описатель модуля.
addr — Адрес начала блока (должен быть кратен 4К).
size — Количество байт для стирания (кратно 4К).

Возвращаемое значение:

Код ошибки.

4.3.10.4 Разрешение записи в пользовательскую область Flash-памяти

Формат: `int32_t X502_FlashWriteEnable (t_x502_hnd hnd)`

Описание:

Функция разрешает запись в пользовательскую область Flash-памяти (первые `X502_FLASH_USER_SIZE` байт). Должна быть вызвана до того, как можно будет использовать `X502_FlashErase()` и `X502_FlashWrite()` для изменения содержимого пользовательской области памяти. После завершения изменений следует вызвать `X502_FlashWriteDisable()`.

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.10.5 Запрет записи в пользовательскую область Flash-памяти

Формат: `int32_t X502_FlashWriteDisable (t_x502_hnd hnd)`

Описание:

Функция запрещает запись в пользовательскую область Flash-памяти модуля (первые `X502_FLASH_USER_SIZE` байт). Должна быть вызвана после того, как нужные данные в пользовательской области были изменены с помощью `X502_FlashErase()` и `X502_FlashWrite()`, чтобы защитить пользовательскую область от случайной изменения в дальнейшем.

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.11 Дополнительные вспомогательные функции

4.3.11.1 Получить версию драйвера модуля L-502.

Формат: `int32_t L502_GetDriverVersion (t_x502_hnd hnd, uint32_t *ver)`

Описание:

Функция возвращает версию драйвера, установленного для указанного открытого устройства. Версия возвращается в виде 32-битного числа. Строковое представление возвращенной версии - четыре числа, старшее соответствует старшему байту, младшее - младшему.

Старший байт - мажорная версия, второй по старшинству байт - минорная, третий - ревизия, четвертый - номер сборки (не используется - всегда 0).

Это та версия, которая отображается в диспетчере устройств в Windows или с помощью modinfo в Linux.

Данная функция доступна только для устройств с интерфейсом PCI/PCI-Express (L502)

Параметры:

hnd — Описатель модуля.

ver — 32-битное число, представляющее собой версию драйвера

Возвращаемое значение:

Код ошибки.

4.3.11.2 Перевод модуля E-502 в режим загрузчика

Формат: `int32_t E502_SwitchToBootloader (t_x502_hnd hnd)`

Описание:

Функция переводит устройство в режим загрузчика для возможности обновления прошивки контроллера Cortex-M4 модуля E-502 с помощью утилиты lboot.

В зависимости от используемого текущего интерфейса для соединения с модулем, модуль переводится в режим загрузки прошивки по интерфейсу USB (если соединение было по USB) или по TFTP (если соединение было по интерфейсу Ethernet).

При успешном вызове данной функции дальнейшая работа с текущем соединением невозможна, т.е. единственным допустимым следующим вызовом является `X502_Close()`.

При переходе в загрузчик находится в режиме загрузчика порядка 30с после чего, если не поступало запросов на перепрошивку загрузчик возвращает управление штатной прошивке. Пока модуль находится в режиме загрузчика с ним невозможно установить соединение с помощью функций данной библиотеки.

Параметры:

hnd — Описатель устройства.

Возвращаемое значение:

Код ошибки.

4.3.11.3 Перезагрузка прошивки ПЛИС

Формат: `int32_t E502_ReloadFPGA (t_x502_hnd hnd)`

Описание:

По данной команде контроллер Cortex-M4 модуля E-502 выполняет сброс ПЛИС и загрузку прошивки ПЛИС из внутренней Flash-памяти.

Это сервисная функция, которая используется главным образом для обновления прошивки ПЛИС.

Параметры:

hnd — Описатель устройства.

Возвращаемое значение:

Код ошибки.

4.3.11.4 Передача управляющей команды контроллеру Cortex-M4.

Формат: `int32_t E502_CortexExecCmd (t_x502_hnd hnd, uint32_t cmd_code, uint32_t par, const uint8_t *snd_data, uint32_t snd_size, uint8_t *rcv_data, uint32_t rcv_size, uint32_t tout, uint32_t *recvd_size)`

Описание:

Функция предназначена для передачи пользовательских управляющих команд контроллеру для случая модифицированной прошивки Cortex-M4.

Параметры:

hnd — Описатель модуля.

cmd_code — Код команды - определяет, что за команда выполняется.

par — Параметр, передаваемый с командой (значение зависит от кода команды).

snd_data — Опциональные данные, передаваемые вместе с командой. Если данные не передаются, то должен передаваться нулевой указатель и snd_size = 0.

snd_size — Количество байт, передаваемых в snd_data

rcv_data — Массив, в который будут переданы данные, возвращенные процессором по завершению команды. Если данные не должны возвращаться, то должен передаваться нулевой указатель, а rcv_size = 0.

rcv_size — Количество байт, которое ожидается, что вернет контроллер по выполнению команды. Массив rcv_data должен быть рассчитан на данное количество слов.

tout — Таймаут в течении которого будет ожидаться, когда контроллер завершит выполнение команды. Функция возвратит управление либо по завершению команды, либо по таймауту.

recvd_size — Если не является нулевым указателем, то в эту переменную будет сохранено количество байт, которое реально вернул контроллер после выполнения команды (контроллер имеет право вернуть меньше данных, чем запрашивалось в rcv_size). Если указатель нулевой, то возвращаение меньшего количества данных считается ошибкой.

Возвращаемое значение:

Код ошибки.

4.3.11.5 Получить версию библиотеки

Формат: `uint32_t X502_GetLibraryVersion (void)`

Описание:

Функция возвращает версию библиотеки x502api. Версия возвращается в виде 32-битного числа. Строковое представление возвращенной версии — четыре числа, старшее соответствует старшему байту, младшее — младшему.

Старший байт — мажорная версия, второй по старшинству байт — минорная, третий — ревизия, четвертый — номер сборки (не используется — всегда 0)

Возвращаемое значение:

32-битное число, представляющее собой версию библиотеки

4.3.11.6 Получение строки об ошибке

Формат: `const char* X502_GetErrorString (int32_t err)`

Описание:

Функция возвращает строку, соответствующую переданному коду ошибки. В настоящее время возвращается всегда русская версия строки (возможно в будущем будет возможность сменить язык глобальной функцией).

Следует учесть, что в ОС Windows строка возвращается в стандартной для Windows кодировке CP1251, в то время как в Linux используется кодировка UTF-8.

Параметры:

err — Код ошибки, для которого нужно вернуть строку.

Возвращаемое значение:

Указатель на строку, соответствующую коду ошибки

4.3.11.7 Моргание светодиодом

Формат: `int32_t X502_LedBlink (t_x502_hnd hnd)`

Описание:

При вызове этой функции, если не запущен синхронный ввод/вывод, происходит кратковременное затухание красного цвета светодиода на передней панели L-502 или светодиода LED1 модуля E-502. Может быть использована для визуальной идентификации модуля после его открытия.

При запущенном синхронном вводе/выводе данный светодиод всегда горит зеленым цветом и данная функция не влияет на его состояние.

Параметры:

hnd — Описатель модуля.

Возвращаемое значение:

Код ошибки.

4.3.11.8 Установка подтягивающих резисторов на входных линиях

Формат: `int32_t X502_SetDigInPullup (t_x502_hnd hnd, uint32_t pullups)`

Описание:

Функция может использоваться для включения подтягивающих резисторов на цифровых входах. Для разных модулей подтягивающие резисторы реализованы на разных входах. Для модулей E-502 и L-502 можно включать их на линиях SYN1 и SYN2. Для L-502 можно отдельно задавать включены или отключены подтяжки на младшей или старшей половине цифровых линий. Для E-502 можно включить подтягивающие к нулю резисторы на входах межмодульной синхронизации. Для E-16 линий TRIG и INT могут быть как входом, так и выходом для сигналов START и CONV.

На не указанных линиях подтягивающие резисторы будут отключены, если они были включены до этого.

При включении питания все подтягивающие резисторы отключены.

Параметры:

hnd — Описатель модуля.

pullups — Флаги (из `t_x502_pullups`), определяющие, на каких линиях включены подтягивающие резисторы.

Возвращаемое значение:

Код ошибки.

4.3.11.9 Проверка поддержки модулем заданной возможности

Формат: `int32_t X502_CheckFeature (t_x502_hnd hnd, uint32_t feature)`

Описание:

Функция используется для проверки, поддерживается ли определенная возможность из `t_x502_features` для данного модуля с текущими версиями прошивок.

Если возможность поддерживается, то будет возвращен код `X502_ERR_OK`.

Данная функция доступна в библиотеке версии 1.1.6 или выше.

Параметры:

hnd — Описатель модуля.

feature — Значение из `t_x502_features`, определяющие, какую возможность необходимо проверить.

Возвращаемое значение:

Если возможность поддерживается, то возвращается `X502_ERR_OK`, иначе — код ошибки

Глава 5

Метки времени (только для E502-P1)

5.1 Режим синхронизации времени по протоколу PTP

В модификации E502-P1 добавлен механизм внутренних часов с возможностью их плавной подстройки. Также добавлена поддержка протокола PTP для синхронизации времени по сети Ethernet и реализована возможность вставки меток времени в поток данных “на ввод”, Таким образом при включении меток времени появилась возможность вычислять точное время для конкретных слов из потока. E502-P1 поддерживает только Peer-to-Peer режим протокола PTP, поэтому сервер реального времени с которым происходит синхронизация должен быть настроен именно на этот режим. В режиме PTP Peer-to-Peer каждый узел сети (в т.ч. коммутаторы) должен быть PTP - совместимым и учитывать задержку до ближайшего соседнего, иначе работа PTP синхронизации не возможна.

Если все условия для работы PTP соблюдены, то синхронизация внутреннего значения времени (внутренних часов) в E502-P1 должна заработать автоматически, устройство будет получать значение точного времени с сервера и корректировать внутренние часы.

Для того чтобы метки времени появились в потоке “на ввод”, необходимо специально включить такой режим через соотвтствующее API, по умолчанию этот режим выключен.

Для работы с PTP в E502-P1 были добавлены несколько новых регистров. Ранее зарезервированный бит в слове “на ввод” теперь означает что слово является меткой времени и для нее вводится признак “захват PTP”.

Признак “захват PTP” - атрибут слова метки времени, означает что для потока слов полученных между двумя метками времени с установленным признаком захвата можно вычислить время каждого слова с определенной точностью.

Для обеспечения требуемой точности синхронизации в E502-P1 введены несколько настроек.

- **Допустимая рассинхронизация часов, +-нс/с.** При каждом получении значения точного времени от сервера, E502-P1 вычисляет разность значения внутренних часов со значением принятым от сервера. Далее эта разность делится на время прошедшее после последней синхронизации (вычисленное по внутренним часам). Вычисленное значение сравнивается с заданным параметром. Синхронизация считается успешной, если вычисленное значение не выходит за заданный интервал.

- **Минимальное кол-во успешных синхронизаций.** Для того чтобы признак “захват PTP” появился, должно произойти минимальное количество успешных синхронизаций. В случае неудачной синхронизации, признак “захват PTP” сбрасывается и цикл ожидания минимального количества успешных синхронизаций повторяется.

5.1.1 Работа с модулем при использовании меток времени

Работа с метками времени из API реализована отдельно от функций управления потоком и функций получения данных из потока.

1. Включение вставки меток времени в поток данных “на ввод”: X502_FpgaRegWrite(hnd, E502_REGS_ARM_TIME_CTRL, 1);
2. Установка настроек захвата: X502_FpgaRegWrite(hnd, E502_REGS_PTP_LOCK_LIMIT (ptp_limit_ns & 0xffff) | (ptp_req_hits << 16))

1. Узнать состояние признака “захват PTP”: X502_FpgaRegRead(E502_REGS_ARM_TIME &status);
2. Инициализировать контекст для обработки меток времени: X502_tstp_init(&tstp_state, adc_freq, din_freq).
3. Обработка меток времени из прочитанных данных с помощью X502_tstp_process_wrd(&tstp_state, cur_wrd), где cur_wrd - очередное слово полученное с помощью X502_Recv()

5.1.2 Описание регистров PTP

E502_REGS_ARM_TIME_CTRL
Address: 0x110

Bit	Name	Access	Description	Reset Value
[1]	STATUS	R	признак “захват PTP” 1: признак “захват PTP” присутствует, условия захвата выплнены 0: признак “захват PTP” отсутствует	0
[0]	ENABLE	RW	1: добавлять метки времени в поток ввода 0: поток ввода без меток времени	0

E502_REGS_PTP_LOCK_LIMIT
Address: 0x706

Bit	Name	Access	Description	Reset Value
[31:16]	ENABLE	RW	Минимальное кол-во успешных синхронизаций	0
[15:0]	STATUS	R	Допустимая рассинхронизация часов, +-нс/с	0

5.1.3 PTP-совместимое оборудование

Оборудование, совместная работа которого с E502-P1 по протоколу PTP была проверена.

Сервер времени: блок коррекции времени ЭНКС-2 производства ООО «Инженерный центр „Энергосервис“» (www.enip2.ru) Сетевой коммутатор с поддержкой PTP: блок L-VIMS-SWITCH производства ООО “Л КАРД” (www.lcard.ru)

5.2 Константы и перечисления

5.2.1 Константы и макроопределения

Константа	Значение	Описание
TSP_NSEC_PER_SEC	(1000000000)	
TSP_WRD1_SSEC_LEN	(24)	
TSP_WRD2_SSEC_LEN	(7)	
TSP_WRD2_SEC_LEN	(19)	
TSP_WRD3_SEC_LEN	(13)	
TSP_SSEC_WIDTH	(31)	
TSP_WRD_NUM	(((wrd) >> 26) & 3)	
TSP_WRD0_ADC_CLK_NUM_MASK	(0x3ffffff)	
TSP_WRD1_LOCK_MASK	(1 << 25)	
TSP_WRD1_FMARK_MASK	(1 << 24)	
TSP_WRD1_SSEC_MASK	((1 << TSP_WRD1_SSEC_LEN) - 1)	
TSP_WRD2_SEC_MASK	(0x3ffff80)	
TSP_WRD2_SSEC_MASK	((1 << TSP_WRD2_SSEC_LEN) - 1)	
TSP_WRD2_SECSSEC_MASK	((1 << (TSP_WRD2_SEC_LEN + TSP_WRD2_SSEC_LEN)) - 1)	
TSP_WRD3_SEC_MASK	((1 << TSP_WRD3_SEC_LEN) - 1)	
TSP_WRD1_IS_FMARK	(!!((wrd) & TSP_WRD1_FMARK_MASK))	
TSP_WRD1_IS_LOCK	(!!((wrd) & TSP_WRD1_LOCK_MASK))	
TSP_WRD1_GET_SSEC	((wrd) & TSP_WRD1_SSEC_MASK)	
TSP_WRD2_GET_SECSSEC	((uint64_t)((wrd) & TSP_WRD2_SECSSEC_MASK) << TSP_WRD1_SSEC_LEN)	
TSP_WRD2_GET_SSEC	((wrd) & TSP_WRD2_SSEC_MASK)	

TSP_WRD3_GET_SEC	((uint64_t)((wrd) & TSP_WRD3_SEC_MASK) << (TSP_WRD1_SSEC_ LEN + TSP_WRD2_ SSEC_LEN + TSP_ WRD2_SEC_LEN))	
SSEC_MAX	(0x7fffffff)	
TSTP_SSEC_TO_NSEC	(time > 0 ? ((uint32_ t)((((double)((time) & SSEC_MAX)) / (1U << TSP_ SSEC_WIDTH)) * TSP_NSEC_PER_ SEC)) : ((uint32_ t)((((double)((time*	(-1)) & SSEC_MAX)) / (1U << TSP_SSEC_ WIDTH)) * TSP_NSEC_ PER_SEC)))
TSTP_SECSSEC_TO_SEC	(time > 0 ? (uint32_t)((time) >> TSP_SSEC_WIDTH) : (uint32_t)((time * -1) >> TSP_SSEC_ WIDTH))	
TSTP_SECSSEC_TO_SEC_DOUBLE	(((double)(time)) / (1U << TSP_SSEC_ WIDTH))	
TSTP_SEC_TO_SSEC	(((uint64_t)(time)) << TSP_SSEC_WIDTH)	

5.2.2 Структура для хранения контекста при обработке потока слов “на ввод” с включенными метками времени

Тип: t_x502_tstp_state		
Описание: Структура для хранения контекста при обработке потока слов “на ввод” с включенными метками времени		
Поле	**Тип**	**Описание поля**
wrd	uint32_t [4]	значение слов последней метки времени
adc_freq	uint32_t	частота АЦП
din_freq	uint32_t	частота DIN
tstp_start_time	t_x502_tstptime	время первой метки времени
tstp_mark_rcvd	bool	признак, что первая метка времени получена

cur_wrd	uint32_t	значение текущего обрабатываемого слова
processed_wrds	uint32_t	кол-во обработанных слов из потока
adcwrds_after_tstp	uint32_t	кол-во слов АЦП после последней метки времени
dinwrds_after_tstp	uint32_t	кол-во слов DIN после последней метки времени
wrds_after_tstp	uint32_t	общее кол-во слов после последней метки времени
last_tstp_time	t_x502_tstptime	время последней метки времени

5.2.3 Тип данных для хранения времени Время хранится в секундах прошедшее с начала этой эпохи (00:00:00 UTC, 1 Января 1970 года) Дробный формат хранения 32.31: 32 целых бит, 31 дробных бит, старшие 32 бита - секунды, младшие 31 бит сабсекунды = 1 / (1<<31) секундсабсекунды,

Тип: t_x502_tstptime

Описание: Тип данных для хранения времени Время хранится в секундах прошедшее с начала этой эпохи (00:00:00 UTC, 1 Января 1970 года) Дробный формат хранения 32.31: 32 целых бит, 31 дробных бит, старшие 32 бита - секунды, младшие 31 бит сабсекунды = 1 / (1<<31) секунд

- сабсекунды,

5.3 Функции для работы с метками времени

5.3.1 Инициализация tstp_state, в нем хранится текущий контекст для операций с метками времени из потока “на ввод”.

Формат: void X502_tstp_init (t_x502_tstp_state *tstp_state, uint32_t adc_freq, uint32_t din_freq)

Описание:

Данная функция инициализирует структуру в которой хранится контекст для работы с метками времени Необходимо указывать на частоту АЦП и DIN т.к. время для слов между двумя метками будет считаться в зависимости от частоты.

Параметры:

tstp_state — Указатель на существующую струтуру t_x502_tstp_state
adc_freq — Частота АЦП
din_freq — Частота DIN

5.3.2 Обработать очередное слово wrd из потока “на ввод”.

Формат: `void X502_tstp_process_wrd (t_x502_tstp_state *tstp_state, uint32_t wrd)`

Описание:

Функция должна быть вызвана только один раз и последовательно для каждого слова полученного из X502_Recv

Параметры:

tstp_state — Указатель на струтуру t_x502_tstp_state предварительно инициализированную через tstp_init()

wrd — Слово из потока “на ввод”

5.3.3 Узнать время текущего обработанного слова

Формат: `void X502_tstp_get_curwrd_time (t_x502_tstp_state *tstp_state, t_x502_tstptime *ret)`

Описание:

Узнать время текущего слова которое до этого было обработано функцией tstp_process_wrd() Формат времени: 32бита - секунды, 31 бит сабсекунды = 1 / (1<<31) секунд

Параметры:

tstp_state — Указатель на струтуру t_x502_tstp_state предварительно инициализированную через tstp_init()

ret — Указатель на t_x502_tstptime по которому будет сохранено расчитанное значение времени для текущего слова

5.3.4 Возвращает признак того что часы синхронизированы

Формат: `bool X502_tstp_get_lock (t_x502_tstp_state *tstp_state)`

Описание:

Возвращает признак “захват PTP” для текущего обработанного слова

Параметры:

tstp_state — Указатель на струтуру t_x502_tstp_state предварительно инициализированную через tstp_init()

Возвращаемое значение:

true: присутствует признак “захват PTP” для текущего обработанного слова, false: признак “захват PTP” отсутствует