Утвержден

ИМЕС.469535.110 Д6-ЛУ

Модуль аналого-цифрового ввода вывода AIC324.

Библиотека функций.

Руководство пользователя

Редакция 1.1

ИМЕС.469535.110Д6

Содержание

Список обновлений и дополнений к документу 4
1 Введение....5
2 Общие сведения 6
2.1 Инсталляция и использование библиотеки функций 6
2.2 Общие правила использования библиотеки 7
2.3 Ограничения....8
3 Описание основных функций библиотеки 9
3.1 Однократное аналого-цифровое преобразование 9
3.2 Многократное аналого-цифровое преобразование с усреднением результатов....9
3.3 Периодическое аналого-цифровое преобразование 9
3.4 Однократное цифро-аналоговое преобразование 11
3.5 Генератор периодических сигналов....11
3.6 Цифровой ввод-вывод....11
3.7 Автокалибровка 12
3.8 Прямой доступ к памяти (DMA) 13
4 Функции библиотеки 14
4.1 Функции обработки ошибок 14
dscGetLastError 14
dscGetErrorString....14
4.2 Функции инициализации 15
dscInit 15
dscFree....15
dscInitBoard....16
dscFreeBoard 16
dscGetBoardMacro....17
4.3 Функции аналого-цифрового преобразования 18
dscADSetSettings....18
dscADSetChannel 18
dscADSample....18
dscADSampleInt....19
dscADSampleAvg....21
dscADScan....21
dscADScanAvg....22
dscADScanInt....22
4.4 Функции цифро-аналогового преобразования 24
dscDASetPolarity....24
dscDASetSettings....24
dscDAConvert 24
dscDAConvertScan 25
4.5 Функции цифрового ввода-вывода....26
dscDIOSetConfig....26
dscDIOInputByte 27
dscDIOOutputByte....27
dscDIOInputBit 28
dscDIOOutputBit....28
dscDIOSetBit....28
dscDIOClearBit....29
4.6 Функции калибровки 29
dscSetCalMux 29
dscADAutocal....29
dscDAAutoCal 30
dscADCalVerify....30
dscDACalVerify....30
aicSetCalPolicy....31
aicSetTemperature 31
aicSetCalSource....31
4.7 Прочие функции....32
dscRegisterRead....32

dscRegisterWrite 32
dscInp 33
dscOutp 33
dscSleep 33
dscGetTime 34
dscGetEEPROM 34
dscSetEEPROM 34
dscSetTrimDac 35
dscGetStatus 35
dscCancelOp 35
dscCancelOpType 36
aicGetDifferential 36
4.8 Функции пересчёта единиц измерения 37
dscADCodeToVoltage 37
dscVoltageToADCode 37
dscVoltageToDACode 38
dscDACodeToVoltage 38
4.9 Функции управления генератором сигналов 39
dscWGCommand 39
dscWGConfigSet 39
dscWGBufferSet 39
4.10 Функции управления пользовательскими прерываниями 40
dscSetUserInterruptFunction 40
dscSetUserInterruptFunctionType 40
dscUserInt 40
4.11 Функции управления счётчиком 8254 42
dscCounterRead 42
dscCounterSetRateSingle 42
dscCounterSetRate 42
dscCounterDirectSet 43
aicSetCounterSettings 43
4.12 Макросы 43
4.13 Коды ошибок 45
4.14 Типы данных 46
DSCCB 46
DSCADSETTINGS 47
DSCADSCAN 48
ERRPARAMS 48
DSCAIOINT 48
DSCDASETTINGS 50
DSCDACS 50
DSCS 50
DSCUserInterruptFunction 51
DSCUSERINTFUNCTION 51
DSCUSERINT 52
DSCCS 52
DSCCR 53
DSCWGCONFIG 53
DSCADCALPARAMS 53
DSCDACALPARAMS 54
AICCS 54
5 Примеры программ 55
5.1 Периодическое аналого-цифровое преобразование 55

Список обновлений и дополнений к документу

№ редакции	Краткое описание изменений	Дата
1.0	Начальная версия руководства	05.07.2011
1.1	Добавлена информация о драйвере для ОС Windows	29.01.2013

1 Введение

Библиотека функций AIC (далее библиотека) обеспечивает интерфейс на языке C, C++ к модулю аналого-цифрового ввода-вывода AIC324. Она свободно распространяется с указанным модулем и может быть загружена с сетевых ресурсов производителя.

Предполагается, что читатель знаком со средствами разработки на языке C, C++, а так же знаком с операционными системами GNU/Linux, DOS, QNX6, Windows XP. Также предполагается знание функциональных свойств и аппаратных особенностей применяемых модулей. Рекомендуется изучить документ «Модуль AIC324. Руководство пользователя» прежде чем приступить к разработке программного обеспечения для него.

Библиотека может быть использована в операционных системах GNU/Linux, DOS, QNX6, Windows XP с помощью языков программирования C или C++.

Библиотека состоит из:

Для ОС GNU/Linux:

- бинарного файла статической библиотеки GNU ar для архитектуры x86;
- заголовка библиотеки, написанного на языке C;
- исходного кода модуля ядра Linux и сценария сборки для GNU make.

Для DOS:

- бинарного файла статической библиотеки OpenWatcom 1.9 для архитектуры x86;
- заголовка библиотеки, написанного на языке C;

Для QNX6:

- бинарного файла статической библиотеки GNU ar для архитектуры x86;
- заголовка библиотеки, написанного на языке C.

Для Windows:

- бинарных файлов динамической библиотеки win32 dll, lib;
- заголовка библиотеки, написанного на языке C;
- системного драйвера sys и inf-файла установщика;
- библиотек соустановщика системного драйвера.

Библиотека поддерживает основные функции модуля AIC324, такие как:

- инициализация модуля;
- аналого-цифровое преобразование;
- цифро-аналоговое преобразование;
- операции с цифровыми входами/выходами;
- автокалибровка;
- управление генератором сигналов;
- управление счётчиками;
- работа с энергонезависимой памятью.

2 Общие сведения

2.1 Инсталляция и использование библиотеки функций

2.1.1 Инсталляция в ОС GNU/Linux

Для установки и использования библиотеки необходимо скопировать файл статической библиотеки libaic.a и заголовок библиотеки *aicud.h* в директорию с программным проектом и использвать ключи компоновщика gcc:

```
LFLAGS = -L. -laic -lpthread -lrt
```

При использовании функций библиотеки, работающих с аппаратными прерываниями от модуля, необходимо наличие в целевой системе загруженного модуля ядра и специального файла устройства `/dev/aic`. Для этого потребуются заголовки и файл конфигурации ядра Linux в точности той же версии, которая используется на целевом устройстве. Последовательность команд для сборки и установки:

```
$ cd <директория с архивом библиотеки>
$ tar -jxf aicud.tar.bz2
$ cd aicud/linux/module
$ make -C '<путь к заголовкам ядра Linux>' SUBDIRS=`pwd` modules
```

Получившийся файл *aickp.ko* скопировать на целевое устройство, и выполнить от пользователя root:

```
$ mknod /dev/aic c 250 0
$ insmod ./aickp.ko aic_major=250
```

Последние две строчки листинга необходимо выполнять на целевом устройстве каждый раз после перезагрузки ОС, поэтому рекомендуется добавить их в стартовый сценарий (например, `/etc/rc.local`), указав правильный полный путь к файлу модуля ядра *aickp.ko*.

> ***Примечание:*** *в примере загрузки модуля ядра указан так называемый major number равный 250. Вы можете изменить это число при необходимости.*

2.1.2 Инсталляция в ОС DOS/FreeDOS

Для установки и использования библиотеки необходимо скопировать файл статической библиотеки libaic.lib и заголовок библиотеки *aicud.h* в директорию с программным проектом и добавить к команде компоновщика OpenWatcom wlink:

```
l libaic.lib
```

2.1.3 Инсталляция в ОС QNX6

Инструкции аналогичны пункту 2.1.1 «Инсталляция в ОС GNU/Linux», за исключением того, что модуль ядра не требуется.

2.1.4 Инсталляция в ОС Windows XP

- Открыть мастер установки оборудования (Пуск – Панель управления – Установка

оборудования), нажать «Далее»;

- После поиска устройств Plug-n-Play выбрать пункт «Да, устройство уже подсоединено» и нажать «Далее»;
- В списке классов устройств выбрать «Добавление нового устройства» и нажать «Далее»;
- Выбрать «Установка оборудования, выбранного из списка вручную» и нажать «Далее»;
- Выбрать «Показать все устройства» и нажать «Далее»;
- Нажать «Установка с диска»;
- Нажать «Обзор» и указать путь к файлу aic324.inf, нажать «ОК»;
- В списке устройств выбрать «Fastwel AIC324 PC104 Driver», нажать «Далее», затем ещё раз «Далее»;
- При появлении вопроса о цифровой подписи драйвера выбрать «Установить драйвер»;
- Дождаться завершения процесса установки;
- Открыть менеджер устройств. Для этого открыть «Пуск», правой кнопкой мыши вызвать контекстное меню пункта «Компьютер», выбрать «Свойства», затем перейти на вкладку «Оборудование» и нажать «Диспетчер устройств»;
- В дереве устройств раскрыть список «Fastwel I/O»
- Правой кнопкой мыши нажать на вновь установленный драйвер платы, в контекстном меню выбрать «Свойства»;
- Перейти на вкладку «Ресурсы», снять отметку «Автоматическая настройка»;
- Выбрать конфигурацию из выпадающего списка «Конфигурация». Для использования функции прямого доступа к памяти (DMA) выбрать «Основная конфигурация 0001», в противном случае установить «Основная конфигурация 0002»;
- Выбрать пункт «Диапазон ввода-вывода», затем нажать «Изменить»;
- Установить диапазон адресов соответственно конфигурации джамперов на плате, нажать «ОК»;
- Выбрать пункт «Прерывание», нажать «Изменить»;
- Установить номер линии прерывания соответственно конфигурации джамперов на плате, нажать «ОК»;
- Выбрать пункт «DMA» (если используется), нажать «Изменить»;
- Установить номер канала DMA соответственно конфигурации джамперов на плате, нажать «ОК»;
- Нажать «ОК», подтвердить назначение ресурсов, закрыть окно диспетчера устройств;
- Процедуру повторить для каждого установленного в системе модуля.
- Подключить к проекту библиотеку aic-api.lib и заголовочный файл aicud.h.

2.2 Общие правила использования библиотеки

Все функции библиотеки возвращают код ошибки типа BYTE. Расшифровку кодов см. в разделе 4.13 «Коды ошибок».

Перед использованием библиотеки необходима её инициализация вызовом функции **dscInit**. Для

получения дескриптора аппаратного модуля необходимо вызвать **dscInitBoard** с параметром boardmacro соответствующим идентификатору используемого модуля. Для модуля AIC324 этот идентификатор равен DSC_AIC324. Для поиска нужного идентификатора по названию модуля можно воспользоваться функцией **dscGetBoardMacro**.

Если в приложении ОС GNU/Linux или Windows планируется работа с функциями, использующими аппаратные прерывания (**dscADSampleInt, dscADScanInt**, **dscUserInt**), то на время работы приложения должен быть загружен модуль ядра, а так же создан файл символьного устройства, см. пункт 2.1.1 «Инсталляция в ОС GNU/Linux» или «Инсталляция в ОС Windows XP».

Для завершения работы с библиотекой вызвать **dscFree**.

2.3 Ограничения

Библиотека функций поддерживает до десяти одновременно работающих модулей AIC324. Программы, использующие библиотеку должны иметь доступ к пространству ввода-вывода (выполняться с правами суперпользователя). Если по соображениям безопасности работа от пользователя root не допускается, то возможно использование функции POSIX `setuid(2)`. В таком случае пользователь, от которого идёт работа должен иметь права чтения и записи на файл устройства модуля ядра.

При использовании QNX и DOS версий драйвера функции пользовательских прерываний выполняются непосредственно в контексте прерывания, что накладывает на код соответствующие ограничения: функция должна быть скомпилирована с отключенной проверкой стека (для DOS), в теле функции не должны выполняться операции, занимающие длительное время, не должны выполняться функции библиотеки AICUD и libc.

Для использовании DMA аппаратная платформа должна иметь в своём составе контроллер DMA Intel 8237 или программно совместимый аналог.

3 Описание основных функций библиотеки

3.1 Однократное аналого-цифровое преобразование

При выполнении преобразования, получается цифровое значение амплитуды аналогового сигнала, приложенного к одному из выбранных каналов. Это значение может быть преобразовано к значению входного напряжения с использованием библиотечной функции **dscADCodeToVoltage**. Обычно это напряжение приводится к физическим величинам по формулам, представляемым производителем датчика.

Данные функции не используют аппаратные прерывания, запуск АЦП выполняется в «ручном» режиме.

Функция однократного преобразования: **dscADSample**.

Последовательность использования:

- установить режим аналогового тракта (диапазон напряжений, полярность, усиление), номер канала и загрузить подстроечные коэффициенты вызовом **dscADSetSettings**;
- выполнить **dscADSample**;
- сменить номер канала с помощью **dscADSetChannel**;
- выполнить **dscADSample** и т.д.

Преобразование можно выполнить либо для одного канала, либо для нескольких каналов за один вызов функции, при этом каналы перебираются последовательно. Такой режим называется режимом «сканирования». Период переключения каналов настраивается в пределах от 4 до 20 мкс.

Функция «сканирования» каналов: **dscADScan**.

Последовательность использования:

- установить режим аналогового тракта (диапазон напряжений, полярность, усиление), интервал переключения каналов и загрузить подстроечные коэффициенты вызовом **dscADSetSettings**;
- вызвать **dscADScan**, повторять вызов с различными номерами каналов.

3.2 Многократное аналого-цифровое преобразование с усреднением результатов

Функция аналогична однократному преобразованию, за исключением того, что цифровое значение амплитуды аналогового сигнала является арифметическим средним от заданного количества преобразований.

Функции многократного преобразования: **dscADSampleAvg**, **dscADScanAvg**.

3.3 Периодическое аналого-цифровое преобразование

Для дискретизации функции переменного напряжения необходим периодический запуск преобразования с жёстко фиксированным периодом, чего невозможно добиться «ручным» запуском. Для этого в модуле AIC324 реализован аналог микросхемы-счётчика Intel 8254, выход которого подключен к сигналу запуска АЦП. Также возможен запуск АЦП с помощью импульса на контакте

внешнего разъёма модуля для синхронизации с другими модулями.

Функции запуска процесса сбора данных: **dscADSampleInt**, **dscADScanInt**.

Функция получения состояния процесса: **dscGetStatus**.

Функции принудительного завершения процесса сбора данных: **dscCancelOp**, **dscCancelOpType**.

Функции **dscADSampleInt**, **dscADScanInt** запускают асинхронный процесс сбора данных в отдельном потоке ОС. Настройки этого процесса задаются структурой **DSCAIOINT**. Требуемое количество отсчётов АЦП устанавливается в поле num_conversions, частота преобразований — в поле conversion_rate . Отсчёты накапливаются в аппаратном буфере FIFO, по достижении порога заполнения (поле fifo_depth) генерируется прерывание. Порог заполнения должен быть в пределах 6-1024. Если использование FIFO запрещено, то прерывание генерируется на каждый отсчёт АЦП или цикл сканирования. Запрет/разрешение использования FIFO устанавливается с помощью поля fifo_enab.

При использовании библиотеки в ОС Linux существует ещё один программный буфер FIFO, находящийся в модуле ядра. Порог заполнения этого промежуточного буфера контролируется полем dump_threshold, по достижении которого происходит копирование данных из пространства ядра в пространство пользователя.

Параметры conversion_rate, fifo_enab, fifo_depth задают частоту аппаратных прерываний, а dump_threshold — темп поступления данных в буфер пользователя и частоту вызова пользовательской функции (см. раздел 4.10 «Функции управления пользовательскими прерываниями»). Установка неправильных значений может привести к переполнению аппаратного FIFO и потере данных, замедлению или «зависанию» компьютера из-за слишком высокой частоты аппаратных прерываний. Эти параметры должны отвечать следующим условиям (символ «%» обозначает остаток от целочисленного деления):

- 6 ≤ fifo_depth ≤ 1024;
- fifo_depth % (high_channel — low_channel + 1) = 0;
- dump_threshold % fifo_depth = 0;
- num_conversions % dump_threshold = 0.

В зависимости от значения поля *cycle* процесс сбора данных будет однократным или циклическим. При *cycle* = FALSE по достижению *num_conversions* процесс остановится, и в пользовательском буфере будет num_conversions последовательных семплов. При *cycle* = TRUE данные в буфере пользователя будут перезаписываться. Во избежание потери данных пользователь должен контролировать ход выполнения операции с помощью **dscGetStatus** и своевременно обрабатывать поступившие .

Алгоритм работы:

- установить режим аналогового тракта (диапазон напряжений, полярность, усиление), и загрузить подстроечные коэффициенты вызовом **dscADSetSettings**;
- выделить память для кольцевого буфера отсчётов АЦП;
- выделить переменную для сохранения указателя записи, приравнять нулю;
- запустить процесс сбора данных **dscADSampleInt**, **dscADScanInt**;
- вызвать **dscSleep**, задержку выбрать исходя из размера буфера и частоты дискретизации;
- вызвать **dscGetStatus**, получить значение указателя записи кольцевого буфера, проверить кольцевой буфер на переполнение, скопировать или обработать отсчёты АЦП в кольцевом буфере по индексам от переменной до указателя записи, приравнять переменную указателю записи;
- повторять последние два пункта по мере необходимости;
- завершить процесс вызовом **dscCancelOp**. Если помимо этого запущен ещё какой-либо асинхронный процесс, то вместо **dscCancelOp** вызвать **dscCancelOpType** с параметром int_type = INT_TYPE_AD.

Пример программы находится в разделе 5.1 «Периодическое аналого-цифровое преобразование».

3.4 Однократное цифро-аналоговое преобразование

При выполнении цифро-аналогового преобразования цифровой код преобразуется в напряжение, которое выдаётся и удерживается на одном или нескольких аналоговых выходах. Для преобразования значения напряжения в цифровой код можно воспользоваться функцией библиотеки AICUD **dscVoltageToDACode**.

Функции цифро-аналогового преобразования: **dscDAConvert**, **dscDAConvertScan**.

Алгоритм работы:

- установить режим аналогового тракта (диапазон напряжений, полярность), и загрузить подстроечные коэффициенты вызовом **dscDASetSettings**;
- произвести преобразование вызовом **dscDAConvert**, или **dscDAConvertScan**, если требуется одновременное обновление напряжений на нескольких каналах ЦАП.

3.5 Генератор периодических сигналов

В модуле AIC324 существует блок, позволяющий превратить каналы ЦАП в программируемый генератор периодических сигналов произвольной формы. Для этого в память модуля надо загрузить массив кодов ЦАП для одного или нескольких каналов, настроить источник тактирования и подать команду за запуск генерации. Источником тактирования может служить внутренний счётчик 8254, перепад напряжения на контакте внешнего разъёма, либо команда генератору. На каждый такт генератор выбирает из внутренней памяти 1, 2 или 4 кода и оправляет в соответствующий этому коду канал. Номера канала для каждого кода задаётся индивидуально на этапе программирования памяти генератора. Указатель чтения сбрасывается в 0 когда доходит до заданного порога.

Функции управления генератором: **dscWGConfigSet**, **dscWGBufferSet**, **dscWGCommand**.

Алгоритм работы:

- установить режим ЦАП (диапазон напряжений, полярность), и загрузить подстроечные коэффициенты вызовом **dscDASetSettings**;
- остановить и сбросить генератор в исходное состояние вызовами **dscWGCommand** с параметрами `WG_CMD_PAUSE` и `WG_CMD_RESET`;
- сформировать и загрузить в память модуля массив напряжений вызовами **dscWGBufferSet**;
- настроить счётчик вызовом **aicSetCounterSettings**, если необходимо;
- настроить генератор с помощью **dscWGConfigSet**;
- запустить генератор вызовом **dscWGCommand** с параметром `WG_CMD_START`;
- тактировать генератор периодическим вызовом **dscWGCommand** с параметром `WG_CMD_INC`, если необходимо.

3.6 Цифровой ввод-вывод

Цифровой ввод-вывод обеспечивает аналог микросхемы Intel 8255, которая представляет собой три 8-ми разрядных порта ввода-вывода. Настройка направления цифровых портов производится функцией **dscDIOSetConfig**, формат конфигурационного байта совпадает с форматом регистра управления микросхемы Intel 8255. Поддерживаются битовые и байтовые операции чтения/записи.

Функции цифрового ввода-вывода: **dscDIOSetConfig**, **dscDIOInputByte**, **dscDIOOutputByte**, **dscDIOInputBit**, **dscDIOOutputBit**, **dscDIOSetBit**, **dscDIOClearBit**.

3.7 Автокалибровка

Для повышения точности аналого-цифрового и цифро-аналогового преобразований в модуле AIC324 существует схема подстройки опорного напряжения и смещения нуля с помощью специализированных ЦАП (TrimDAC). Подстроечные коэффициенты TrimDAC хранятся в энергонезависимой памяти, для каждого диапазона преобразователя и диапазона температур имеется свой набор коэффициентов. Для установки коэффициентов нужно присвоить полю *load_cal* в структурах **DSCADSETTINGS**, **DSCDASETTINGS** значение TRUE при вызове функций **dscADSetSettings** и **dscDASetSettings**.

Для корректировки подстроечных коэффициентов существует алгоритм автокалибровки. Пользователь может инициировать процесс функциями **dscADAutocal**, **dscDAAutoCal**.

> ***Примечание:*** *в процессе автокалибровки на выходах ЦАП могут появляться флуктуации напряжения от половины до полного диапазона напряжений ЦАП. Перед запуском автокалибровки ЦАП все устройства, использующие выходы ЦАП, должны быть отсоединены.*

Процесс автокалибровки занимает от 1 до 3 секунд на диапазон. По окончании процесса коэффициенты TrimDAC сохраняются в область энергонезависимой памяти, соответствующую текущей температуре платы.

Номер диапазона АЦП и ЦАП, для которого будет проводится калибровка выбирается соответствующим полем в структурах **DSCADSETTINGS**, **DSCDASETTINGS** и соответствует таблице:

DSCADCALPARAMS.adrange				
Значение	Полярность	Диапазон	Усиление	Диапазон напряжений АЦП
0	UNIPOLAR	RANGE_5	GAIN_1	0 .. +5В
1	UNIPOLAR	RANGE_5	GAIN_2	0 .. +2.5В
2	UNIPOLAR	RANGE_5	GAIN_4	0 .. +1.25В
3	UNIPOLAR	RANGE_5	GAIN_8	0 .. +0.625В
4	UNIPOLAR	RANGE_10	GAIN_1	0 .. +10В
5	UNIPOLAR	RANGE_10	GAIN_2	0 .. +5В
6	UNIPOLAR	RANGE_10	GAIN_4	0 .. +2.5В
7	UNIPOLAR	RANGE_10	GAIN_8	0 .. +1.25В
8	BIPOLAR	RANGE_5	GAIN_1	-5 .. +5В
9	BIPOLAR	RANGE_5	GAIN_2	-2.5 .. +2.5В
10	BIPOLAR	RANGE_5	GAIN_4	-1.25 .. +1.25В
11	BIPOLAR	RANGE_5	GAIN_8	-0.625 .. 0.625В
12	BIPOLAR	RANGE_10	GAIN_1	-10 .. +10В
13	BIPOLAR	RANGE_10	GAIN_2	-5 .. +5В
14	BIPOLAR	RANGE_10	GAIN_4	-2.5 .. +2.5В
15	BIPOLAR	RANGE_10	GAIN_8	-1.25 .. +1.25В

DSCDACALPARAMS.darange_calibrate			
Номер	Полярность	Диапазон	Диапазон напряжений ЦАП
0	UNIPOLAR	5	0 .. +5В
1	UNIPOLAR	10	0 .. +10В
2	BIPOLAR	5	-2.5 .. +2.5В
3	BIPOLAR	10	-5 .. +5В
4	BIPOLAR	20	-10 .. +10В

Существует набор коэффициентов, записанный на этапе производства платы, который находится в защищённой от записи области памяти. Управление алгоритмом загрузки коэффициентов при вызове

функций установки режима осуществляется функцией **aicSetCalPolicy**. По умолчанию после инициализации модуля установлен режим LOADCAL_DEFAULT: при отсутствии температурных калибровок загружаются заводские.

Существует функция выбора температурного режима платы **aicSetTemperature**, позволяющая управлять областью откуда будут взяты температурные калибровки.

3.8 Прямой доступ к памяти (DMA)

Модуль AIC324 поддерживает использование ISA DMA в блочном режиме (код 0x84 для конфигурационного регистра контроллера DMA I8237). Использование прямого доступа к памяти позволяет освободить центральный процессор от ожидания выполнения копирования данных из аппаратного буфера модуля в память при выполнении оцифровки сигналов с высокими частотами семплирования. В операционных системах QNX6 и GNU/Linux для памяти-назначения используется промежуточный буфер, выделенный с учётом ограничений контроллера I8237, поэтому никаких требований к буферу не предъявляется. Для выполнения преобразований **dscADSampleInt** с использованием DMA в DOS пользователь должен выделить память так, чтобы буфер не пересекал границу сегмента памяти (64 Кбайт), иначе будет возвращена ощибка DE_BUFFER_ROLLOVER, см. 4.13 «Коды ошибок».

4 Функции библиотеки

4.1 Функции обработки ошибок

dscGetLastError

BYTE dscGetLastError(ERRPARAMS *errparams)

Возвращает в полях структуры параметра *errparams* код и строку расшифровки последней ошибки.

> ***Примечание:*** *не рекомендуется использование при многопоточной работе с драйвером при использовании нескольких аппаратных модулей, так как код последней ошибки — это глобальная переменная, и её значение может быть изменено другим потоком, работающим с другим модулем.*

- **Входные параметры**
  errparams - указатель на результат работы функции.
- **Возвращаемое значение**
  Код ошибки.
- **Пример использования**

```
# include <aicud.h>

int main() {
  BYTE rc;
  if(dscInit(DSC_VERSION)) {
    ERRPARAMS err;
    dscGetLastError(&err);
    printf("dscInit() failed with code %i: %s\n",
            err.ErrCode, err.errstring);
    return EXIT_FAILURE;
  }
  /* успешная инициализация */
```

dscGetErrorString

const char* dscGetErrorString(BYTE error_code)

Возвращает строковую расшифровку переданного параметром кода ошибки. Если код ошибки выходит за пределы возможных, то возвращается строка соответствующая коду `DE_INVALID_ERROR_CODE «INVALID ERROR CODE, PLEASE IGNORE»`

- **Входные параметры**
  error_code - код ошибки, полученный от предыдущих вызовов функций библиотеки.

- **Возвращаемое значение**
  Указатель на строку расшифровки кода ошибки, завершённую кодом «0».

4.2 Функции инициализации

dscInit

BYTE dscInit(WORD version);

Выполняет инициализацию драйвера и проверку версии заголовочных файлов. Возвращает ошибку `DE_DSCUDH_INVALID` с случае несовпадения номера версии заголовка с номером, определённым на этапе компиляции библиотеки драйвера.

- **Входные параметры**
  version - версия заголовочных файлов, определена как макрос DSC_VERSION в aicud.h.

- **Возвращаемое значение**
  Код ошибки.

- **Пример использования**

```
# include <aicud.h>

int main() {
  BYTE rc;
  rc = dscInit(DSC_VERSION);
  if(rc != DE_NONE) {
    printf("dscInit() failed: %s\n", dscGetErrorString(rc));
    return EXIT_FAILURE;
  }
  /* успешная инициализация */
```

dscFree

BYTE dscFree(void)

Отменяет все текущие операции по всем аппаратным модулям под управлением драйвера и освобождает все ресурсы.

- **Входные параметры**
  Нет.

- **Возвращаемое значение**

Код ошибки.

dscInitBoard

BYTE dscInitBoard(BYTE boardtype, DSCCB* dsccb, DSCB* board)

Инициализация аппаратного модуля.

Для linux-версии драйвера программа в момент вызова этой функции должна обладать правами суперпользователя, т. к. происходит вызов `ioperm` (2) и обращение к портам ввода-вывода. В QNX6 производится вызов функций `ThreadCtl(_NTO_TCTL_IO, 0)` и `mmap_device_io`, поэтому вызывающий пользователь должен обладать соответствующими привелегиями. При отсутствии прав доступа возвращается ошибка.

- **Входные параметры**

  boardtype - макрос, указывающий на тип аппаратного модуля;
  dsccb - настройки аппаратного модуля, см. описание структуры DSCCB;
  board - возвращаемый дескриптор аппаратного модуля.

- **Возвращаемое значение**

  Код ошибки;.
  Дескриптор аппаратного модуля по указателю *board*.

- **Пример использования**

```
# include <aicud.h>

int main() {
  int rc;
  DSCCB dsccb;
  DSCB board;

  if(dscInit(DSC_VERSION))
    return EXIT_FAILURE;

  dsccb.base_address = 0x100;
  dsccb.int_level = 7;
  rc = dscInitBoard(DSC_AIC324, &dsccb, &dscb);
  if(rc != DE_NONE) {
    printf("dscInitBoard() failed: %s\n", dscGetErrorString(rc));
    return EXIT_FAILURE;
  }
  ...
```

dscFreeBoard

BYTE dscFreeBoard(DSCB board)

Отменяет все текущие операции с аппаратным модулем, соответствующим дескриптору board и

освобождает связанные с ним ресурсы.

- **Входные параметры**
  board - дескриптор аппаратного модуля.

- **Возвращаемое значение**
  Код ошибки.

dscGetBoardMacro

BYTE dscGetBoardMacro(char* boardtype, BYTE* macro)

Поиск числового идентификатора аппаратного модуля по его названию.

- **Входные параметры**
  boardtype - строка названия модуля;
  macro - указатель для сохранения идентификатора модуля.

- **Возвращаемое значение**
  Код ошибки;
  Идентификатор модуля по указателю *macro.*

- **Пример использования**

```
# include <aicud.h>

int main() {
  int rc;
  DSCCB dsccb;
  DSCB board;
  BYTE boardtype;

  if(dscInit(DSC_VERSION))
    return EXIT_FAILURE;

  rc = dscGetBoardMacro("aic324", &boardtype);
  if(rc != DE_NONE) {
    fprintf(stderr, "dscGetBoardMacro() failed: %s\n", dscGetErrorString(rc));
    return EXIT_FAILURE;
  }
  dsccb.base_address = 0x100;
  dsccb.int_level = 7;
  rc = dscInitBoard(boardtype, &dsccb, &dscb);
  ...
```

4.3 Функции аналого-цифрового преобразования

dscADSetSettings

BYTE dscADSetSettings(DSCB board, DSCADSETTINGS* settings)

Устанавливает настройки аналогового тракта, текущий номер канала и подстроечные коэффициенты из энергонезависимой памяти, соответствующие режиму тракта. При невозможности загрузки калибровочных коэффициентов в соответствии с установленным функцией **aicSetCalPolicy** правилом возвращается ошибка DE_HW_FAILURE.

> ***Примечание:*** *при вызове функции с параметром settings.load_cal=TRUE время выполнения функции увеличивается, т. к. происходит чтение подстроечных коэффициентов из энергонезависимой памяти. Подобнее см. пункт 3.7 «Автокалибровка».*

- **Входные параметры**

board - дескриптор аппаратного модуля;
settings - настройки аппаратного модуля, см. описание структуры DSCADSETTINGS.

- **Возвращаемое значение**

Код ошибки.

- **Пример использования**

см. описание функции dscADSample.

dscADSetChannel

BYTE dscADSetChannel(DSCB board, BYTE low_channel, BYTE high_channel)

Устанавливает номера каналов однократного преобразования и режима сканирования.

Параметр *low_channel* задаёт текущий канал, т. е. тот канал по которому будут выполняться дальнейшие преобразования с помощью функций dscADSample, **dscADSampleAvg**.

- **Входные параметры**

board - дескриптор аппаратного модуля;
low_channel - нижний канал сканирования;
high_channel - верхний канал сканирования.

- **Возвращаемое значение**

Код ошибки.

dscADSample

BYTE dscADSample(DSCB board, DSCSAMPLE* sample)

Выполняет одиночное аналого-цифровое преобразование по текущему каналу.

Примечание: *Для получения корректного результата аналоговый тракт и номер канала должны быть предварительно настроены вызовом функции* ***dscADSetSettings****.*

- **Входные параметры**

board - дескриптор аппаратного модуля;
sample - указатель на результат преобразования.

- **Возвращаемое значение**

Код ошибки.

- **Пример использования**

Примечание: *здесь и далее код инициализации драйвера и аппаратного модуля опущен, см. пример использования функции* ***dscInitBoard****.*

```
#include <aicud.h>

BYTE rc;
DSCB board;
DSCADSETTINGS settings;
DSCSAMPLE sample;

...

/* Step 1 */
settings.current_channel = 0;
settings.gain = GAIN_1;
settings.polarity = BIPOLAR;
settings.range = RANGE_10;
settings.load_cal = TRUE;

if((rc = dscADSetSettings(board, settings)) != DE_NONE)
    return rc;

/* Step 2 */
if((rc = dscADSample(board, &sample)) != DE_NONE)
    return rc;

...
```

dscADSampleInt

BYTE dscADSampleInt(DSCB board, DSCAIOINT* dscaioint)

Выполняет аналого-цифровое преобразование с автоматическим запуском по событию.

Событием может служить таймер или перепад на внешнем выводе. Выполняется одно преобразование АЦП на событие. Преобразования выполняются до достижения нужного количества отсчётов, либо в цикле до отмены с помощью **dscCancelOp** или **dscCancelOpType**. При циклическом процессе (при cycle=TRUE в **DSCAIOINT**) положение указателя записи определяется с помощью

dscGetStatus. Функция запускает асинхронный процесс чтения отсчётов АЦП в отдельном потоке операционной системы, поэтому пользователь должен отслеживать момент завершения операции самостоятельно, периодически вызывая **dscGetStatus** и анализируя результат.

> ***Примечание:*** *Для получения корректного результата аналоговый тракт должен быть предварительно настроен вызовом функции* ***dscADSetSettings.***

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  dscaioint - настройки аналого-цифрового преобразования.

- **Возвращаемое значение**
  Код ошибки;
  Массив отсчётов АЦП в памяти по указателю в поле *sample_values* структуры **DSCAIOINT**.

- **Пример использования**

```
#include <aicud.h>

DSCB dscb;
DSCADSETTINGS dscadsettings;
DSCAIOINT dscaioint;
DSCS dscs;

...

/* Step 1 */

dscadsettings.current_channel = 0;
dscadsettings.gain = GAIN_1;
dscadsettings.range = RANGE_10;
dscadsettings.polarity = BIPOLAR;
dscadsettings.load_cal = TRUE;

if ((result = dscADSetSettings(dscb, &dscadsettings)) != DE_NONE)
    return result;

/* Step 2 */

dscaioint.num_conversions = 1024;
dscaioint.conversion_rate = 10000;
dscaioint.cycle = FALSE;
dscaioint.internal_clock = TRUE;
dscaioint.low_channel = 0;
dscaioint.external_gate_enable = FALSE;
dscaioint.internal_clock_gate = FALSE;
dscaioint.fifo_enab = TRUE;
dscaioint.fifo_depth = 256;
dscaioint.dump_threshold = 512;

dscaioint.sample_values = (WORD*) malloc(sizeof(DSCSAMPLE) *
```

```
          dscaioint.num_conversions);
memset(dscaioint.sample_values, 0, sizeof(DSCSAMPLE) * dscaioint.num_conversions);

if ((result = dscADScanInt(dscb, &dscaioint)) != DE_NONE)
    return result;

/* Step 3 */
while(1) {
    dscGetStatus(dscb, &dscs);
    if(dscs.total_transfers >= 1024 || dscs.op_type == OP_TYPE_NONE)
          break;
    dscSleep(50);
}
```

dscADSampleAvg

BYTE dscADSampleAvg(DSCB board, DFLOAT* average, int count)

Выполняет аналого-цифровое преобразование с усреднением.

Вызов функции аналогичен вызову **dscADSample** count раз, с усреднением результата преобразования.

> ***Примечание:*** *Для получения корректного результата аналоговый тракт и номер канала должны быть предварительно настроены вызовом функции* ***dscADSetSettings****.*

- **Входные параметры**

board - дескриптор аппаратного модуля;
average - указатель на память для сохранения усреднённого результата;
int_count - количество усреднений.

- **Возвращаемое значение**

Код ошибки;
Усреднённый по *count* отсчётам код АЦП в параметре *average.*

dscADScan

BYTE dscADScan(DSCB board, DSCADSCAN* dscadscan, DSCSAMPLE* sample_values)

Выполняет одиночное аналого-цифровое преобразование по нескольким каналам последовательно (сканирование).

В структуре DSCADSCAN поля low_channel и high_channel задают соответственно нижний и верхний каналы, по которым происходит преобразование. Интервал между семплами по каналам задаётся предварительным вызовом dscADSetSettings, параметром scan_interval в структуре DSCADSETTINGS. Массив по указателю sample_values должен быть размером не менее dscadscan.high_channel - dscadscan.low_channel + 1 элементов.

- **Входные параметры**

board - дескриптор аппаратного модуля;
dscadscan - настройки сканирования;
sample_values - указатель для сохранения отсчётов АЦП.

- **Возвращаемое значение**
  Код ошибки;
  Отсчёты АЦП по указателю *sample_values*.

dscADScanAvg

BYTE dscADScanAvg(DSCB board, DSCADSCAN* dscadscan, DFLOAT* average_values, int count)

Выполняет аналого-цифровое преобразование по нескольким каналам (сканирование) с усреднением.

Параметры dscadscan.sample_values и average_values должны указывать на массив размером не менее dscadscan.high_channel - dscadscan.low_channel + 1 элементов. Вызов функции аналогичен вызову **dscADScan** count раз с поканальным усреднением результатов преобразования.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  dscadscan - настройки сканирования, указатель на массив промежуточных значений;
  average_values - указатель для сохранения усреднённых отсчётов АЦП.

- **Возвращаемое значение**
  Код ошибки;
  Усреднённые отсчёты АЦП по указателю *average_values*.

dscADScanInt

BYTE dscADScanInt(DSCB board, DSCAIOINT* dscaioint)

Выполняет аналого-цифровое преобразование по нескольким каналам (сканирование) с автоматическим запуском по событию.

Событием может служить таймер или перепад на внешнем выводе. Выполняется high_channel - low_channel + 1 преобразований АЦП на событие. Параметры high_channel и low_channel задаются в структуре DSCAIOINT. Преобразования выполняются до достижения нужного количества отсчётов, либо в цикле до отмены с помощью dscCancelOp или dscCancelOpType. При циклическом процессе (при cycle=TRUE в DSCAIOINT) положение указателя записи определяется с помощью dscGetStatus. Функция запускает асинхронный процесс чтения отсчётов АЦП в отдельном потоке операционной системы, поэтому пользователь должен отслеживать момент завершения операции самостоятельно, периодически вызывая dscGetStatus и анализируя результат.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  dscadscan - настройки сканирования;
  sample_values - указатель для сохранения отсчётов АЦП.

- **Возвращаемое значение**
  Код ошибки;

Отсчёты АЦП.

- **Пример использования**

```
...

DSCB dscb;
DSCADSETTINGS dscadsettings;
DSCAIOINT dscaioint;

...

/* Step 1 */

dscadsettings.current_channel = 0;
dscadsettings.gain = 0;
dscadsettings.range = 0;
dscadsettings.polarity = 0;
dscadsettings.load_cal = 0;

/* Step 2 */

if ((result = dscADSetSettings(dscb, &dscadsettings)) != DE_NONE)
        return result;

/* Step 3 */

dscaioint.num_conversions = 1024;
dscaioint.conversion_rate = 10000;
dscaioint.cycle = FALSE;
dscaioint.internal_clock = TRUE;
dscaioint.low_channel = 0;
dscaioint.high_channel = 3;
dscaioint.external_gate_enable = FALSE;
dscaioint.internal_clock_gate = FALSE;
dscaioint.fifo_enab = TRUE;
dscaioint.fifo_depth = 256;
dscaioint.dump_threshold = 512;

/* Step 4 */

dscaioint.sample_values = (WORD*) malloc(sizeof(WORD) *
        dscaioint.num_conversions);
memset(dscaioint.sample_values, 0, sizeof(WORD) * dscaioint.num_conversions);

/* Step 5 */

if ((result = dscADScanInt(dscb, &dscaioint)) != DE_NONE)
                return result;
```

4.4 Функции цифро-аналогового преобразования

dscDASetPolarity

BYTE dscDASetPolarity(DSCB board, BYTE polarity)

Установка режима работы ЦАП.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  polarity - BIPOLAR устанавливает биполярный режим работы, UNIPOLAR - униполярный.

- **Возвращаемое значение**
  Код ошибки.

dscDASetSettings

BYTE dscDASetSettings(DSCB board, DSCDASETTINGS * dasettings)

Установка режима и диапазона напряжений ЦАП.

При невозможности загрузки калибровочных коэффициентов в соответствии с установленным функцией **aicSetCalPolicy** правилом возвращается ошибка DE_HW_FAILURE.

> ***Примечание:*** *для модулей AIC324 возможные диапазоны напряжений — 5.0, 10.0, 20.0; реальный диапазон входных напряжений зависит от режима полярности. Подробнее см. в Руководстве пользователя AIC324.*

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  dasettings - настройки ЦАП.

- **Возвращаемое значение**
  Код ошибки.

dscDAConvert

BYTE dscDAConvert(DSCB board, BYTE channel, DSCDACODE output_code)

Производит одиночное цифро-аналоговое преобразование по заданному каналу. В результате выполнения функции на канале channel цифро-аналогового преобразователя устанавливается и удерживается напряжение, соответствующее режиму работы ЦАП и коду output_code.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  channel - номер канала ЦАП;
  output_code - код ЦАП.

- **Возвращаемое значение**
  Код ошибки.

- **Пример использования**

```
#include <aicud.h>
DSCB board;
...
DSCDACODE code;
DSCDASETTINGS settings;
ERRPARAMS err;

settings.polarity = UNIPOLAR;
settings.range = 10.0;
settings.load_cal = TRUE;
rc = dscDASetSettings(board, &settings);
if(rc != DE_NONE) {
    dscGetLastError(&err);
    fprintf(stderr, "dscDASetSettings: %s", err.errstring);
    return -1;
}
rc = dscVoltageToDACode(board, settings, 5.0, &code);
if(rc != DE_NONE) {
    dscGetLastError(&err);
    fprintf(stderr, "dscVoltageToDACode: %s", err.errstring);
    return -1;
}
rc = dscDAConvert(board, 0, code);
if(rc != DE_NONE) {
    dscGetLastError(&err);
    fprintf(stderr, "dscDAConvert: %s", err.errstring);
    return -1;
}
```

dscDAConvertScan

BYTE dscDAConvertScan(DSCB board, DSCDACS *dscdacs)

Производит одиночное цифро-аналоговое преобразование одновременно по нескольким каналам. В результате выполнения функции на каналах цифро-аналогового преобразователя **x**, для которых выполняется условие dscdacs.channel_enable[**x**]==TRUE устанавливается и удерживается напряжение, соответствующее режиму работы ЦАП и коду dscdacs.output_code[**x**].

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  dscdacs - параметры преобразования.

- **Возвращаемое значение**
  Код ошибки.

- **Пример использования**

```
#include <aicud.h>
DSCB board;
...
DSCDACS scan;
DSCDACODE code[4];
DSCDASETTINGS settings;
ERRPARAMS err;

settings.polarity = UNIPOLAR;
settings.range = 10.0;
settings.load_cal = TRUE;
rc = dscDASetSettings(board, &settings);
if(rc != DE_NONE) {
    dscGetLastError(&err);
    fprintf(stderr, "dscDASetSettings: %s", err.errstring);
    return -1;
}
memset(&scan, 0, sizeof(scan));
scan.channel_enable[1] = TRUE;
scan.channel_enable[2] = TRUE;
code[1] = 0x1000;
code[2] = 0x2000;
scan.output_codes = code;

rc = dscDAConvertScan(board, &scan);
if(rc != DE_NONE) {
    dscGetLastError(&err);
    fprintf(stderr, "dscDAConvertScan: %s", err.errstring);
    return -1;
}
```

4.5 Функции цифрового ввода-вывода

dscDIOSetConfig

BYTE dscDIOSetConfig(DSCB board, BYTE* config_bytes)

Записывает конфигурационный байт. Подробнее см. описание микросхемы Intel 8255.

- **Входные параметры**

board - дескриптор аппаратного модуля;
config_bytes - указатель на байт конфигурации.

- **Возвращаемое значение**

Код ошибки.

- **Пример использования**

```
#include <aicud.h>
BYTE rc;
BYTE config_byte;

config_byte = 0x9b;  // set all ports on chip to input

if ((rc = dscDIOSetConfig(dscb, &config_byte)) != DE_NONE)
{
    fprintf(stderr, "dscDIOSetConfig: %s\n", dscGetErrorString(rc));
    return rc;
}
```

dscDIOInputByte

BYTE dscDIOInputByte(DSCB board, BYTE port, BYTE* digital_value)

Записывает по указателю digital_value значение, полученное из цифрового порта port.

- **Входные параметры**

board - дескриптор аппаратного модуля;
port - индекс цифрового порта;
digital_value - указатель для результата.

- **Возвращаемое значение**

Код ошибки;
Цифровое значение по указателю digital_value.

dscDIOOutputByte

BYTE dscDIOOutputByte(DSCB board, BYTE port, BYTE digital_value)

Выводит в цифровой порт port значение digital_value.

- **Входные параметры**

board - дескриптор аппаратного модуля;
port - индекс цифрового порта;
digital_value - байт для вывода.

- **Возвращаемое значение**

Код ошибки.

dscDIOInputBit

BYTE dscDIOInputBit(DSCB board, BYTE port, BYTE bit, BYTE* digital_value)

Записывает по указателю digital_value булевое состояние бита bit порта port.

- **Входные параметры**

board - дескриптор аппаратного модуля;
port - индекс цифрового порта;
bit - номер бита, 0-7;
digital_value - указатель для сохранения результата TRUE или FALSE.

- **Возвращаемое значение**

Код ошибки;
Состояние бита по указателю digital_value.

dscDIOOutputBit

BYTE dscDIOOutputBit(DSCB board, BYTE port, BYTE bit, BYTE digital_value)

Установка на разряде bit цифрового порта port уровня напряжения, соответствующего булевому значению digital_value.

- **Входные параметры**

board - дескриптор аппаратного модуля;
port - индекс цифрового порта;
bit - номер бита, 0-7;
digital_value - состояние бита, TRUE или FALSE.

- **Возвращаемое значение**

Код ошибки.

dscDIOSetBit

BYTE dscDIOSetBit(DSCB board, BYTE port, BYTE bit)

Установка на разряде bit цифрового порта port уровня напряжения, соответствующего булевому значению TRUE.

- **Входные параметры**

board - дескриптор аппаратного модуля;
port - индекс цифрового порта;
bit - номер бита, 0-7.

- **Возвращаемое значение**

Код ошибки.

dscDIOClearBit

BYTE dscDIOClearBit(DSCB board, BYTE port, BYTE bit)

Установка на разряде bit цифрового порта port уровня напряжения, соответствующего булевому

значению FALSE.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  port - индекс цифрового порта;
  bit - номер бита, 0-7.

- **Возвращаемое значение**
  Код ошибки.

4.6 Функции калибровки

dscSetCalMux

BYTE dscSetCalMux(DSCB board, BOOL on)

Подключение калибровочного мультиплексора.

При вызове с параметром *on*=TRUE АЦП переключается с входов на мультиплексор, используемый в процессе автокалибровки. Внутренняя функция драйвера, пользователь не должен вызывать её самостоятельно.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  on - состояние мультиплексора.

- **Возвращаемое значение**
  Код ошибки.

dscADAutocal

BYTE dscADAutoCal(DSCB board, DSCADCALPARAMS* params)

Автоматическая калибровка АЦП.

Процесс занимает от 1 до 3 секунд на режим. Найденные коэффициенты подстроечных ЦАП сохраняются в область энергонезависимой памяти, соответствующей текущей температуре платы. Диапазон АЦП выбирается полем *adrange* в стркутуре **DSCADCALPARAMS**.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  params - настройки процесса автокалибровки.

- **Возвращаемое значение**
  Код ошибки;
  Значения ошибки смещения и опоры в полях *ad_offset* и *ad_gain* структуры *params.*

dscDAAutoCal

BYTE dscDAAutoCal(DSCB board, DSCDACALPARAMS* params)

Автоматическая калибровка ЦАП.

Процесс занимает от 1 до 3 секунд на режим. Найденные коэффициенты подстроечных ЦАП сохраняются в область энергонезависимой памяти, соответствующей текущей температуре платы.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  params - настройки процесса автокалибровки.

- **Возвращаемое значение**
  Код ошибки;
  Значения ошибки смещения и опоры в полях *offset* и *gain* структуры *params*.

dscADCalVerify

BYTE dscADCalVerify(DSCB board, DSCADCALPARAMS* params)

Проверка отклонения кода АЦП от опорных значений.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  params - настройки процесса автокалибровки.

- **Возвращаемое значение**
  Код ошибки;
  Значения ошибки смещения и опоры в кодах АЦП полях *ad_offset* и *ad_gain* структуры *params.*

dscDACalVerify

BYTE dscDACalVerify(DSCB board, DSCDACALPARAMS* params)

Проверка отклонения напряжения ЦАП от опорных значений.

Возвращается ошибка установки напряжения ЦАП в кодах, соответствующих биполярному режиму АЦП, диапазон 10 В, усиление 1.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  params - настройки процесса автокалибровки.

- **Возвращаемое значение**
  Код ошибки;
  Значения ошибки смещения и опоры в кодах АЦП в полях *offset* и *gain* структуры *params.*

aicSetCalPolicy

BYTE aicSetCalPolicy(DSCB board, BYTE policy)

Установка правила загрузки калибровочных коэффициентов.

Служит для управления поведением функций **dscADSetSettings** и **dscDASetSettings** вызванных с параметром *load_cal*=TRUE в структурах **DSCADSETTINGS** и **DSCDASETTINGS**.

- **Входные параметры**

board - дескриптор аппаратного модуля;
policy - код правила, см. таблицу ниже.

Название	Код	Поведение функций установки режима
LOADCAL_DEFAULT	0	Правило по умолчанию. Функция пытается загрузить пользовательский набор коэффициентов, соответствующий температуре платы, при его отсутствии происходит попытка чтения набора, записанного на этапе производства модуля, при ошибке возвращается DE_HW_FAILURE.
LOADCAL_TEMPERATURE	1	Загружается пользовательский набор коэффициентов, соответствующий температуре платы, при его отсутствии возвращается ошибка DE_HW_FAILURE.
LOADCAL_CALIBRATE_MODE	2	Внутренний режим. Используется в процессе автокалибровки. Поведение аналогично режиму по умолчанию, за исключением того, что при отсутствии заводских калибровок загружаются значения по умолчанию.
LOADCAL_MANUFACTURING_MODE	3	Загружается набор коэффициентов, записанный на этапе производства модуля при его отсутствии возвращается ошибка DE_ HW_FAILURE.

- **Возвращаемое значение**

Код ошибки.

aicSetTemperature

aicSetTemperature(DSCB board, SWORD temperature)

Установка температурного режима платы.

Позволяет управлять областью откуда будут загружаться или куда сохраняться калибровочные коэффициенты при последующих вызовах **dscADSetSettings**, **dscDASetSettings**, **dscADAutocal**, **dscDAAutoCal**. Сохраняется до освобождения платы или указании нового значения температуры. При указании специального параметра TEMPERATURE_AUTO (число -273) температура будет измеряться датчиком температуры модуля по необходимости.

- **Входные параметры**

board - дескриптор аппаратного модуля;
temperature - температура платы, или специальный параметр TEMPERATURE_AUTO.

aicSetCalSource

aicSetCalSource(DSCB board, BYTE source)

Установка источника напряжения на входе АЦП.

Необходима в процессе автокалибровки. Пользователь не должен вызывать эту функцию самостоятельно.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  source - код источника, см. таблицу ниже.

Название	Код	Источник напряжения
CAL_5V	0	Источник опорного напряжения +5 Вольт
CAL_GND	1	0 Вольт
CAL_DAC0	2	Выход ЦАП 0
CAL_DAC1	3	Выход ЦАП 1
CAL_DAC2	4	Выход ЦАП 2
CAL_DAC3	5	Выход ЦАП 3
CAL_AUX	6	Выход дополнительного ЦАП

- **Возвращаемое значение**
  Код ошибки.

4.7 Прочие функции

В этот раздел вошли функции, облегчающие прямой доступ к плате, переносимость программ, функции доступа к энергонезависимой памяти и управления асинхронными операциями с аппаратными модулями.

dscRegisterRead

BYTE dscRegisterRead(DSCB board, WORD address, BYTE* data)

Чтение регистра со смещением address относительно базового адреса аппаратного модуля.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  address - смещение регистра относительно базового адреса модуля;
  data - указатель для сохранения результата.

- **Возвращаемое значение**
  Код ошибки;
  Значение регистра по указателю data.

dscRegisterWrite

BYTE dscRegisterWrite(DSCB board, WORD address, BYTE data)

Запись значения data в регистр со смещением address относительно базового адреса аппаратного модуля.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  address - смещение регистра относительно базового адреса модуля;

data - значение регистра.

- **Возвращаемое значение**

Код ошибки.

dscInp

BYTE dscInp(DWORD address, BYTE* value)

Чтение регистра ввода-вывода с абсолютным адресом address.

- **Входные параметры**

board - дескриптор аппаратного модуля;
address - адрес регистра;
value - указатель для сохранения результата.

- **Возвращаемое значение**

DE_NONE;
Значение регистра по указателю value.

dscOutp

BYTE dscOutp(DWORD address, BYTE value)

Запись регистра ввода-вывода с абсолютным адресом address.

- **Входные параметры**

board - дескриптор аппаратного модуля;
address - адрес регистра;
value - значение регистра.

- **Возвращаемое значение**

DE_NONE.

dscSleep

BYTE dscSleep(DWORD ms)

Пауза в выполнении программы не менее, чем ms миллисекунд. Точность зависит от операционной системы.

- **Входные параметры**

ms - длительность задержки в миллисекундах.

- **Возвращаемое значение**

DE_NONE.

dscGetTime

BYTE dscGetTime(DWORD* ms)

Возвращает значение системного счётчика в миллисекундах. Разрешение счётчика зависит от операционной системы.

- **Входные параметры**
  ms - указатель для сохранения результата.

- **Возвращаемое значение**
  Код ошибки;
  Значение системного счётчика по указателю ms.

dscGetEEPROM

BYTE dscGetEEPROM(DSCB board, DWORD address, BYTE* data)

Читает байт из энергонезависимой памяти по адресу address.

Примечание: *для модуля AIC324 пользователю доступна область адресов 0x1c00 — 0x5fff.*

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  address - адрес в энергонезависимой памяти;
  byte - указатель для сохранения значения.

- **Возвращаемое значение**
  Код ошибки;
  Значение байта в энергонезависимой памяти по указателю *data*.

dscSetEEPROM

BYTE dscSetEEPROM(DSCB board, DWORD address, BYTE data)

Записывает байт data в энергонезависимую память по адресу address.

Примечание: *для модуля AIC324 пользователю доступна область адресов 0x1c00 — 0x5fff. Диапазон адресов 0x0 — 0x1bff используется для хранения калибровочных коэффициентов, адреса 0x6000 и выше используется для хранения заводских настроек и недоступна для записи. Запись в эти адреса может привести к неправильной работе платы.*

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  address - адрес в энергонезависимой памяти;
  data - значение для записи.

- **Возвращаемое значение**
  Код ошибки.

dscSetTrimDac

BYTE dscSetTrimDac(DSCB board, DWORD trimDac, BYTE value)

Устанавливает код подстроечного ЦАП trimDAC. Адреса и назначение подстроечных ЦАП см. в руководстве пользователя используемого модуля. Внутренняя функция драйвера, пользователь не должен вызывать эту функцию самостоятельно.

Внимание! *При установке некорректных значений в подстроечные ЦАП возможна неправильная работа модуля.*

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  trimDac - адрес подстроечного ЦАП;
  value - код ЦАП.

- **Возвращаемое значение**
  Код ошибки.

dscGetStatus

BYTE dscGetStatus(DSCB board, DSCS* status)

Чтение статуса асинхронных операций. Применяется совместно с dscADSampleInt, dscADScanInt.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  trimDac - указатель для сохранения результата.

- **Возвращаемое значение**
  Код ошибки;
  Состояние текущих операций ввода-вывода.

- **Пример использования**
  см. пример использования функции dscADSampleInt.

dscCancelOp

BYTE dscCancelOp(DSCB board)

Принудительно завершает все асинхронные операции. Если ни одной не запущено, функция возвращает DE_NONE_IN_PROGRESS.

- **Входные параметры**
  board - дескриптор аппаратного модуля.

- **Возвращаемое значение**
  Код ошибки.

dscCancelOpType

BYTE dscCancelOpType(DSCB board, DWORD int_type)

Принудительно завершает асинхронную операцию, соответствующие флагу *i*nt_type. Если ни одной не запущено, функция возвращает DE_NONE_IN_PROGRESS. *i*nt_type должен быть одним из INT_TYPE_*, см. «Флаги текущих операций».

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  int_type - флаг операции, которую необходимо отменить, см. «Флаги текущих операций».

- **Возвращаемое значение**
  Код ошибки.

aicGetDifferential

BYTE aicGetDifferential(DSCB board, BYTE *state)

Возвращает состояния джамперов, управляющих режимом работы входных коммутаторов. Для модуля AIC324 бит 0 значения state соответствует джамперу J21, бит 1 — J22. Установленный бит соответствует установленному джамперу, дифференциальному режиму работы соответствующих каналов. Подробнее см. руководство пользователя используемого модуля.

> ***Примечание:*** *драйвер AICUD не отслеживает режим входного каскада, пользователь должен следить за состоянием джамперов и использовать только те каналы, которые задействованы в текущей конфигурации.*

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  state - указатель для сохранения состояния джамперов.

- **Возвращаемое значение**
  Код ошибки;
  Состояние джамперов по указателю *state*.

4.8 Функции пересчёта единиц измерения

dscADCodeToVoltage

BYTE dscADCodeToVoltage(DSCB board, DSCADSETTINGS adsettings, DSCSAMPLE adcode, DFLOAT *voltage)

Рассчитывает напряжение на входе исходя из кода АЦП и настроек аналогового тракта.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  adsettings - настройки аналогового тракта;

adcode - код АЦП;
voltage - указатель для сохранения рассчитанного значения напряжения.

- **Возвращаемое значение**

Код ошибки;
Рассчитанное значение напряжения по указателю voltage, В.

- **Пример использования**

```
#include <aicud.h>

DSCB dscb;
...
ERRPARAMS err;
DSCADSETTINGS adsettings;
DSCSAMPLE sample;
DFLOAT voltage;
BYTE rc;

if ((rc = dscADSample(dscb, &sample)) != DE_NONE)
{
    dscGetLastError(&err);
    fprintf(stderr, "dscADSample: %s\n", err.errstring);
    return rc;
}
if ((rc = dscADCodeToVoltage(dscb, adsettings, sample, &voltage)) != DE_NONE)
{
    dscGetLastError(&err);
    fprintf(stderr, "dscADCodeToVoltage: %s\n", err.errstring);
    return rc;
}
printf(«Sample: 0x%.4x, voltage: %5.32fV\n», sample, voltage);
```

dscVoltageToADCode

BYTE dscVoltageToADCode(DSCB board, DSCADSETTINGS adsettings, DFLOAT voltage, DSCSAMPLE *adcode)

Преобразование, обратное dscADCodeToVoltage. Рассчитывает код АЦП исходя из данного напряжения и настроек аналогового тракта.

- **Входные параметры**

board - дескриптор аппаратного модуля;
adsettings - настройки аналогового тракта;
voltage - напряжение, В;
adcode - указатель для сохранения рассчитанного кода АЦП.

- **Возвращаемое значение**

Код ошибки;
Рассчитанный код АЦП по указателю adcode.

dscVoltageToDACode

BYTE dscVoltageToDACode(DSCB board, DSCDASETTINGS dasettings, DFLOAT voltage, DSCDACODE *dacode)

Преобразовать напряжение в код ЦАП.

- **Входные параметры**

board - дескриптор аппаратного модуля;
dasettings - настройки аналогового тракта;
voltage - значение напряжения;
dacode - указатель для сохранения кода ЦАП.

- **Возвращаемое значение**

Код ошибки;
Рассчитанный код ЦАП по указателю dacode.

dscDACodeToVoltage

BYTE dscDACodeToVoltage(DSCB board, DSCDASETTINGS dasettings, DSCDACODE dacode, DFLOAT *voltage)

Преобразование, обратное **dscVoltageToDACode**.

- **Входные параметры**

board - дескриптор аппаратного модуля;
dasettings - настройки аналогового тракта;
dacode - код ЦАП;
voltage - указатель для сохранения значения напряжения.

- **Возвращаемое значение**

Код ошибки;
Рассчитанное значение напряжения по указателю voltage, В.

4.9 Функции управления генератором сигналов

dscWGCommand

BYTE dscWGCommand(DSCB board, DWORD cmd)

Команда генератору сигналов.

- **Входные параметры**

board - дескриптор аппаратного модуля;
cmd - код команды WG_CMD_*, см. раздел 4.12 «Генератор сигналов».

- **Возвращаемое значение**

Код ошибки.

dscWGConfigSet

BYTE dscWGConfigSet(DSCB board, DSCWGCONFIG *config)

Задание конфигурации генератора сигналов.

- **Входные параметры**

board - дескриптор аппаратного модуля;
settings - конфигурация генератора, см. описание структуры **DSCWGCONFIG**.

- **Возвращаемое значение**

Код ошибки.

dscWGBufferSet

BYTE dscWGBufferSet(DSCB board, DWORD address, DSCDACODE value, DWORD channel, BOOL simul)

Запись отсчёта ЦАП в память аппаратного модуля.

- **Входные параметры**

board - дескриптор аппаратного модуля;
address - адрес слова в памяти аппаратного модуля;
value - код ЦАП;
channel - номер канала ЦАП для которого предназначается семпл;
simul - не используется.

- **Возвращаемое значение**

Код ошибки.

4.10 Функции управления пользовательскими прерываниями

dscSetUserInterruptFunction

BYTE dscSetUserInterruptFunction(DSCB board, DSCUSERINTFUNCTION* dscuserintfunc)

Установка пользовательской функции для всех типов операций.

> ***Примечание:*** *для DOS и QNX6 версий библиотеки на пользовательскую функцию накладываются ограничения, см раздел 2.3 «Ограничения».*

- **Входные параметры**

board - дескриптор аппаратного модуля;
dscuserintfunc - указатель на пользовательскую функцию.

- **Возвращаемое значение**

Код ошибки.

dscSetUserInterruptFunctionType

BYTE dscSetUserInterruptFunctionType(DSCB board, DSCUSERINTFUNCTION* dscuserintfunc, DWORD int_type)

Установка пользовательской функции для операции, заданной параметром *int_type*.

> ***Примечание:*** *для DOS и QNX6 версий библиотеки на пользовательскую функцию накладываются ограничения, см пункт 2.3 «Ограничения».*

- **Входные параметры**

board - дескриптор аппаратного модуля;
dscuserintfunc - указатель на пользовательскую функцию;
int_type - тип операции, один из INT_TYPE_*, см раздел 4.12 «Флаги текущих операций».

- **Возвращаемое значение**

Код ошибки.

dscUserInt

BYTE dscUserInt(DSCB board, DSCUSERINT *dscuserint, DSCUserInterruptFunction function)

Настройка выполнения функции по событию. Событием может служить либо таймер, либо перепад уровня на внешнем выводе.

> ***Примечание:*** *для DOS и QNX6 версий библиотеки на пользовательскую функцию накладываются ограничения, см пункт 2.3 «Ограничения».*

- **Входные параметры**

board - дескриптор аппаратного модуля
dscuserint - параметры события
function - указатель на функцию для выполнения

- **Возвращаемое значение**

Код ошибки.

- **Пример использования**

В этом примере происходит настройка драйвера на выполнение пользовательской функции с частотой 10 герц, источником прерываний служит таймер 0. Функции инициализации и освобождения модуля опущены.

```
#include <dscud.h>

unsigned long userint_counter;

#ifdef DOS
#pragma check_stack(off)
#endif
void userIntFunction(void *parm)
{
    userint_counter++;
}
```

```
#ifdef DOS
#pragma check_stack()
#endif

BYTE execUserInt(void)
{
    int rc;
    ERRPARAMS err;
    DSCUSERINT uint;

    uint.intsource = USER_INT_SOURCE_INTERNAL;
    uint.rate = 10.0;
    uint.clksource = 0;
    uint.counter = COUNTER_0;
    userint_counter = 0;
    rc = dscUserInt(board, &uint, userIntFunction);
    if(rc != DE_NONE) {
          dscGetLastError(&err);
          fprintf(stderr, "dscUserInt failed: %s (%i)", err.errstring,
                      err.ErrCode);
          return rc;
    }
    return DE_NONE;
}
```

4.11 Функции управления счётчиком 8254

dscCounterRead

BYTE dscCounterRead(DSCB board, DSCCR* dsccr)

Чтение состояния счётчиков C0-C2 аналога микросхемы Intel 8254. Подробнее о регистрах счётчика см. описание микросхемы Intel 8254.

- **Входные параметры**
  board - дескриптор аппаратного модуля
  dsccr - состояния счётчиков

- **Возвращаемое значение**
  Код ошибки.
  dsccr - прочитанные состояния счётчиков

dscCounterSetRateSingle

BYTE dscCounterSetRateSingle(DSCB board, float hertz, DWORD ctr)

Установка частоты выдачи счётчиком тактовых сигналов. Параметр ctr должен быть одним из COUNTER_*, см. раздел 4.12 «Флаги счётчиков».

> ***Примечание:*** *функция модифицирует только регистры аналога 8254, для управления источником тактирования счётчика необходимо либо прямое обращение к регистрам аппаратного модуля в соответствии с руководством пользователя, либо использование* ***aicSetCounterSettings.***

- **Входные параметры**

board - дескриптор аппаратного модуля;
hertz - желаемая частота выдачи тактовых импульсов;
ctr - один из флагов счётчика, см. раздел 4.12 «Флаги счётчиков».

- **Возвращаемое значение**

Код ошибки.

dscCounterSetRate

BYTE dscCounterSetRate(DSCB board, float hertz)

Установка частоты выдачи тактовых сигналов счётчиком, отвечающим за тактирование АЦП.

> ***Примечание:*** *в модуле AIC324 за тактирование АЦП отвечают последовательно включенные счётчики 1 и 2, поэтому для этого модуля функция эквивалентна вызову* ***dscCounterSetRateSingle(board, herts, COUNTER_1_2)****.*

- **Входные параметры**

board - дескриптор аппаратного модуля
hertz - состояния счётчиков
ctr - один из флагов счётчика, см. раздел 4.12 «Флаги счётчиков».

- **Возвращаемое значение**

Код ошибки.

dscCounterDirectSet

BYTE dscCounterDirectSet(DSCB board, BYTE code, DWORD data, BYTE ctr_number)

Ручная настройка счётчиков. Предназначена для прямой установки коэффициента деления и управляющего регистра одного из счётчиков Intel 8254.

> ***Примечание:*** *функция модифицирует только регистры аналога 8254, для управления источником тактирования счётчика необходимо либо прямое обращение к регистрам аппаратного модуля в соответствии с руководством пользователя, либо использование* ***aicSetCounterSettings.***

- **Входные параметры**

board - дескриптор аппаратного модуля;
code - управляющий регистр;
data - делитель;
ctr_number - номер устанавливаемого счётчика 8254, в диапазоне 0-2.

- **Возвращаемое значение**
  Код ошибки.

aicSetCounterSettings

BYTE aicSetCounterSettings(DSCB board, DWORD counter, AICCS *settings)

Настройка счётчика, источника тактирования счётчика, управление выходом счётчика.

- **Входные параметры**
  board - дескриптор аппаратного модуля;
  counter - один из флагов счётчика, см. раздел 4.12 «Флаги счётчиков»;
  settings - настройки счётчика, см. описание структуры AICCS.

- **Возвращаемое значение**
  Код ошибки.

4.12 Макросы

Название	Значение	Комментарий
Настройки АЦП (DSCADSETTINGS)		
RANGE_5	0	Поле *range*, диапазон напряжений АЦП 0 - 5 В или -5 - +5 В в зависимости от поля *polarity*
RANGE_10	1	Поле *range*, диапазон напряжений АЦП 0 - 10 В или -10 - +10 В в зависимости от поля *polarity*
BIPOLAR	0	Поле *polarity*, биполярный режим АЦП
UNIPOLAR	1	Поле *polarity*, униполярный режим АЦП
GAIN_1	0	Поле *gain, у*силение 1
GAIN_2	1	Поле *gain, у*силение 2
GAIN_4	2	Поле *gain, у*силение 4
GAIN_8	3	Поле *gain, у*силение 8
SCAN_INTERVAL_4	5	Поле *scan_interval*, интервал 4 мкс
SCAN_INTERVAL_5	3	Поле *scan_interval*, интервал 5 мкс
SCAN_INTERVAL_9	4	Поле *scan_interval*, интервал 9 мкс
SCAN_INTERVAL_10	2	Поле *scan_interval*, интервал 10 мкс
SCAN_INTERVAL_15	1	Поле *scan_interval*, интервал 15 мкс
SCAN_INTERVAL_20	0	Поле *scan_interval*, интервал 20 мкс
Флаги текущих операций		
INT_TYPE_AD	0x01	Операция аналого-цифрового преобразования
INT_TYPE_DA	0x02	Операция цифро-аналогового преобразования
INT_TYPE_DIOIN	0x04	Генерация прерываний по внешнему событию
INT_TYPE_USER	0x08	Не используется
INT_TYPE_COUNTER	0x10	Генерация прерываний по таймеру

INT_TYPE_DIOREAD	0x20	Не используется
INT_TYPE_OPTO	0x40	Не используется
INT_TYPE_DIOWRITE	0x80	Не используется
Режимы выполнения пользовательской функции (DSCUSERINTFUNCTION)		
USER_INT_CANCEL	0	Отмена пользовательской функции
USER_INT_AFTER	1	Функция выполняется после отработки прерывания стандартным отработчиком
USER_INT_INSTEAD	2	Функция выполняется вместо стандартного отработчика, не поддерживается в текущей версии драйвера
Генератор сигналов		
WG_CH_PER_FRAME_1	0	Количество каналов на такт генератора = 1
WG_CH_PER_FRAME_2	1	Количество каналов на такт генератора = 2
WG_CH_PER_FRAME_4	2	Количество каналов на такт генератора = 4
WG_SRC_MANUAL	0	Ручное тактирование вызовом **dscWGCommand** с командой WG_CMD_INC
WG_SRC_CTR0	1	Источник тактирования — счётчик 0 Intel 8254
WG_SRC_CTR12	2	Не поддерживается драйвером AICUD
WG_SRC_EXT	3	Источник тактирования — перепад на внешнем выводе
WG_CMD_START	0	Команда запуска генератора сигнала
WG_CMD_PAUSE	1	Команда останова генератора сигнала
WG_CMD_RESET	2	Команда сброса адреса генератора сигнала
WG_CMD_INC	3	Команда тактирования генератора сигнала
Флаги счётчиков		
COUNTER_0	0x00	Счётчик 0
COUNTER_1	0x01	Не используется
COUNTER_2	0x02	Не используется
COUNTER_0_1	0x04	Не используется
COUNTER_1_2	0x08	Последовательно включенные счётчики 1 и 2
COUNTER_0_1_2	0x10	Не используется
Аппаратные модули (dscInitBoard)		
DSC_AIC324	0	Модуль AIC324

4.13 Коды ошибок

Название	Код	Строковое значение	Комментарий
DE_NONE	0	NO ERRORS REPORTED	Нормальное выполнение функции
DE_HW_FAILURE	1	HARDWARE FAILURE	Аппаратная ошибка
DE_SW_FAILURE	2	SOFTWARE FAILURE	Программная ошибка
DE_HW_NOT_SUPPORTED	3	OPERATION NOT SUPPORTED BY HARDWARE	Операция не поддерживается аппаратным модулем
DE_SW_NOT_SUPPORTED	4	OPERATION NOT	Операция не поддерживается

		SUPPORTED BY SOFTWARE	драйвером
DE_INVALID_PARM	5	INVALID PARAMETER	Неверный параметр функции
DE_ALTERNATE_IN_PROGRESS	6	ALTERNATE INTERRUPT OPERATION IN PROGRESS	Запущена другая операция, использующая тот же тип прерываний
DE_NONE_IN_PROGRESS	7	NO INTERRUPT OPERATION IN PROGRESS	Ни одной операции не запущено
DE_BUFFER_ROLLOVER	8	POINTER + SIZE OF DATA OVERRUN BUFFER	Буфер пересекает границу сегмента памяти
DE_INVALID_ERROR_CODE	9	INVALID ERROR CODE, PLEASE IGNORE	Неизвестная ошибка
DE_ALREADY_PAUSED	10	OPERATION ALREADY PAUSED	Не используется
DE_OVERFLOW	11	AM9513A COUNTER FUNCTION OVERFLOWED OR BUFFER OVERFLOW	Не используется
DE_INVALID_FUNC	12	INVALID FUNCTION FOR SELECTED BOARD	Не используется
DE_DSCUDH_INVALID	13	LIBRARY HEADER VERSION MISMATCH	Несоответствие версий библиотеки и заголовка
DE_INVALID_BOARD	14	INVALID BOARD TYPE SPECIFIED	Неизвестный тип аппаратного модуля
DE_BOARD_LIMIT_REACHED	15	MAXIMUM NUMBER OF BOARDS REACHED	Максимальное количество модулей
DE_BOARD_BUSY	16	ATTEMPT TO ACQUIRE BOARD TIMED OUT	Модуль занят другим потоком
DE_INVALID_WINDRVR_HANDLE	17	INVALID WINDRIVER HANDLE RETURNED	Не используется
DE_INVALID_WINDRVR_VERSION	18	INVALID WINDRIVER VERSION	Не совпадают версии модуля ядра и библиотеки
DE_BAD_WINDRVR_BOARD_INIT	19	WINDRIVER BOARD INITIALIZATION FAILURE	Не используется
DE_OPERATION_TIMED_OUT	20	OPERATION TIMED OUT	Таймаут ожидания операции
DE_INVALID_WINDRVR_KP	21	KERNEL PLUGIN INITIALIZATION FAILURE	Не загружен модуль ядра или отсутствует файл устройства
DE_THREAD_FAILURE	22	THREAD INITIALIZATION FAILURE	Ошибка запуска потока
DE_MUTEX_FAIL	23	MUTEX CREATION OR LOCKING OPERATION FAILED	Ошибка создания или захвата мьютекса

4.14 Типы данных

В данном разделе описаны структуры данных определённых как тип, т. е. с использованием оператора `typedef`. В данном Руководстве Пользователя описаны лишь те поля, которые используются драйвером AICUD, остальные существуют для совместимости с Универсальным Драйвером DSCUD фирмы Diamond Systems.

Название	Тип данных	Комментарий
BYTE	unsigned char	

WORD	unsigned short	
SWORD	signed short	
DWORD	unsigned long	
LONG	signed long	
FLOAT	float	
DFLOAT	double	
BOOL	int	
TRUE	BOOL 1	
FALSE	BOOL 0	
DSCSAMPLE	signed short	Код АЦП
DSCDACODE	unsigned long	Код ЦАП
DSCB	signed short	Дескриптор аппаратного модуля

DSCCB

Используется как параметр функции dscInitBoard, для поиска и настройки плат AIC

```
typedef struct {
  BYTE boardtype;
  DSCB boardnum;
  WORD base_address;
  BYTE int_level;
  BOOL RMM_external_trigger;
  BOOL RMM_external_trigger_c3;
  WORD EMM_IOAddr[8];
  WORD EMM_Interrupt[8];
  BYTE clkfrq0;
  BYTE clkfrq1;
  BYTE clksel1;
  WORD address_space;
  BYTE DAC_Config;
  LONG clock_freq;
  BYTE dma_level;
  BYTE int_level1;
  BYTE int_level2;
  BYTE int_level3;
  WORD fpga;
  BYTE SMM_AD_resolution;
  BOOL RMM_mode_16bit;
  BOOL RMM_DAC_0_3_external_enable;
  BOOL RMM_DAC_4_7_external_enable;
} DSCCB;
```

Название	Тип	Назначение	Комментарий
boardtype	BYTE	Вход	Тип аппаратного модуля, см. функции **dscGetBoardMacro**

base_address	WORD	Вход	Базовый адрес регистров аппаратного модуля на шине ISA
int_level	BYTE	Вход	Номер линии прерывания

DSCADSETTINGS

Используется как параметр функции dscADSetSettings для настройки аналогового тракта

```
typedef struct {
  BYTE current_channel;
  BYTE gain;
  BYTE range;
  BYTE polarity;
  BYTE load_cal;
  BYTE scan_interval;
  BYTE addiff;
} DSCADSETTINGS;
```

Название	Тип	Назначение	Комментарий
current_channel	BYTE	Вход	Установка текущего канала для последующих вызовов функций **dscADSample, dscADSampleAvg**
gain	BYTE	Вход	Усиление, GAIN_1 - GAIN_8
range	BYTE	Вход	Диапазон напряжений: RANGE_5, RANGE_10
polarity	BYTE	Вход	Полярность тракта: UNIPOLAR, BIPOLAR
load_cal	BYTE	Вход	При *load_cal*=TRUE происходит загрузка коэффициентов TrimDAC из EEPROM, соответствующим устанавливаемым настройкам для повышения точности преобразования, см. пункт 3.7 «Автокалибровка»
scan_interval	BYTE	Вход	Временной интервал между переключением каналов в режиме сканирования: SCAN_INTERVAL_4 - SCAN_INTERVAL_20
addiff	BYTE	Выход	Возвращает режим работы входных коммутаторов. Формат аналогичен результату работы ф-ции **aicGetDifferential**

DSCADSCAN

Параметр функции dscADScan.

```
typedef struct {
 BYTE      low_channel;
  BYTE      high_channel;
  DSCSAMPLE *sample_values;
  BYTE      gain;
} DSCADSCAN;
```

Название	Тип	Назна	Комментарий

		чение	
low_channel	BYTE	Вход	Нижний канал сканирования
high_channel	BYTE	Вход	Верхний канал сканирования
sample_values	DSCSAMPLE*	Вход	Указатель на память объёмом (*high_channel* - *low_channel* + 1)*sizeof(DSCSAMPLE)

ERRPARAMS

Параметр функции dscGetLastError.

```
typedef struct {
  BYTE ErrCode;
  const char *errstring;
} ERRPARAMS;
```

Название	Тип	Назначение	Комментарий
ErrCode	BYTE	Выход	Код последней ошибки, см. раздел 4.13 «Коды ошибок»
errstring	const char *	Выход	Строка с описанием ошибки

DSCAIOINT

Параметр функций dscADSampleInt, dscADScanInt.

```
typedef struct {
 DWORD num_conversions;
  FLOAT conversion_rate;
  FLOAT conversion_rate_final;
  BOOL  cycle;
  BOOL  internal_clock;
  BYTE  low_channel;
  BYTE  high_channel;
  BOOL  external_gate_enable;
  BOOL  internal_clock_gate;
  DSCSAMPLE  *sample_values;
  BOOL  fifo_enab;
  WORD  fifo_depth;
  DWORD dump_threshold;
  BOOL  channel_align;
  BYTE  clksource;
} DSCAIOINT;
```

Название	Тип	Назначение	Комментарий

num_conversions	DWORD	Вход	Размер буфера, в отсчётах АЦП типа DSCSAMPLE
conversion_rate	FLOAT	Вход	Желаемая частота преобразования
conversion_rate_final	FLOAT	Выход	Реально установленная частота преобразования
cycle	BOOL	Вход	*cycle*=TRUE - циклическое заполнение, *cycle*=FALSE - однократное заполнение буфера
internal_clock	BOOL	Вход	При *internal_clock*=TRUE тактирование происходит от внутреннего счётчика, аналога Intel 8254
low_channel	BYTE	Вход	Нижний канал режима сканирования, см. **dscADScanInt**
high_channel	BYTE	Вход	Верхний канал режима сканирования, см. **dscADScanInt**
external_gate_enable	BOOL	Вход	Использование сигнала разрешения тактирования АЦП (EXTGATE/DIN2) при тактировании от внешнего источника. При *external_gate_enable*=TRUE и *internal_clock*=FALSE низкий уровень на входе EXTGATE/DIN2 запрещает преобразования АЦП, высокий — разрешает.
internal_clock_gate	BOOL	Вход	Использование сигнала разрешения тактирования АЦП (EXTGATE/DIN2) при тактировании от внутреннего источника. При *internal_clock_gate*=TRUE и *internal_clock*=TRUE низкий уровень на входе EXTGATE/DIN2 запрещает преобразования АЦП, высокий — разрешает.
sample_values	DSCSAMPLE*	Вход	Указатель на буфер, куда будут считываться семплы АЦП
fifo_enab	BOOL	Вход	*fifo_enab*=TRUE включает FIFO АЦП
fifo_depth	WORD	Вход	Порог заполнения FIFO, при котором генерируется прерывание. 3-1024
dump_threshold	DWORD	Вход	Размер порции данных в семплах, которая выгружается из модуля ядра ОС. Для linux-версии определяет частоту вызова пользовательской функции.
clksource	BYTE	Вход	Источник тактирования счётчика, 0, 1 внутренний, 2 - внешний

DSCDASETTINGS

Настройки цифро-аналогового преобразователя, параметр функции dscDASetSettings.

```
typedef struct {
  BYTE polarity;
  BYTE load_cal;
  FLOAT range;
  BYTE gain;
  BYTE dasim;
  BYTE daPolEn;
 BYTE daUR;
  BYTE daURChannel;
} DSCDASETTINGS;
```

Название	Тип	Назначение	Комментарий
polarity	BYTE	Вход	Полярность тракта: UNIPOLAR, BIPOLAR

load_cal	BOOL	Вход	При *load_cal*=TRUE происходит загрузка коэффициентов TrimDAC из EEPROM, соответствующим устанавливаемым настройкам для повышения точности преобразования
range	FLOAT	Вход	Диапазон напряжений. Для модуля AIC324 корректные значения: 5.0, 10.0, 20.0

DSCDACS

Параметры цифро-аналогового преобразования. Аргумент функции dscDAConvertScan.

```
#define DACS_MAX_CHANNELS 16

typedef struct {
  BOOL channel_enable[DACS_MAX_CHANNELS];
  DSCDACODE *output_codes;
} DSCDACS;
```

Название	Тип	Назначение	Комментарий
channel_enable	BOOL[16]	Вход	Флаг активности канала в преобразовании по нескольким каналам ЦАП. *channel_enable[x]* = 1 означает, что *output_codes[x]* участвует в преобразовании
output_codes	DSCDACODE*	Вход	Буфер с кодами ЦАП

DSCS

Статус преобразования. Аргумент функции dscGetStatus. Применяется совместно с dscADSampleInt, dscADScanInt.

```
typedef struct {
  DWORD op_type;
  DWORD transfers;
  DWORD total_transfers;
  DWORD da_transfers;
  DWORD da_total_transfers;
  DWORD overflows;
  BYTE  paused;
  BYTE  output;
  WORD  base_address;
  BYTE  int_level;
  BOOL  analog;
} DSCS;
```

Название	Тип	Назначение	Комментарий

op_type	DWORD	Выход	Объединённые по ИЛИ флаги текущих операций INT_TYPE_*, см. раздел 4.12 «Флаги текущих операций»
transfers	DWORD	Выход	Положение указателя записи в буфере АЦП
total_transfers	DWORD	Выход	Общее количество семплов АЦП с начала операции
da_transfers	DWORD	Выход	Положение указателя чтения в буфере ЦАП
da_total_transfers	DWORD	Выход	Общее количество семплов ЦАП с начала операции
overflows	DWORD	Выход	Общее количество переполнений с начала операции. Если overflows ненулевое, то произошло переполнение буфера FIFO, и результат может быть искажён

DSCUserInterruptFunction

Указатель на пользовательскую функцию отработки прерывания.

```
typedef void (*DSCUserInterruptFunction) (void* parameter);
```

DSCUSERINTFUNCTION

Параметр функций dscSetUserInterruptFunction, dscSetUserInterruptFunctionType.

```
typedef struct {
 DSCUserInterruptFunction func;
  BYTE int_mode;
  DWORD int_type;
} DSCUSERINTFUNCTION;
```

Название	Тип	Назначение	Комментарий
func	DSCUserInterruptFunction	Вход	Указатель на функцию
int_mode	BYTE	Вход	Режим выполнения, см. раздел 4.12 «Режимы выполнения пользовательской функции»
int_type	DWORD	Вход	Тип прерывания, см. раздел 4.12 «Флаги текущих операций»

DSCUSERINT

Параметр функции dscUserInt.

```
typedef struct {
  BYTE  intsource;
  BYTE  counter;
```

```
  BYTE  clksource;
  FLOAT rate;
  DWORD int_type;
  DSCUserInterruptFunction func;
} DSCUSERINT;
```

Название	Тип	Назначение	Комментарий
intsource	BYTE	Вход	Источник прерываний, см. «»
counter	BYTE	Вход	Номер счётчика
clksource	BYTE	Вход	Источник тактирования счётчика
rate	FLOAT	Вход	Частота прерываний от счётчика
int_type	DWORD	Выход	Тип операции который присвоил драйвер
func	DSCUserInterruptFunction	Выход	Указатель на пользовательскую функцию (для совместимости с Diamond API)

DSCCS

Входит в состав DSCCR. Отражает состояние счётчика Intel 8254.

```
typedef struct {
  DWORD value;
  BYTE status;
} DSCCS;
```

Название	Тип	Назначение	Комментарий
value	DWORD	Выход	Значение счётчика
status	BYTE	Выход	Регистр состояния счётчика

DSCCR

Параметр функции dscCounterRead. Используется для чтения состояний счётчиков Intel 8254.

```
typedef struct {
  DWORD value;
  BYTE status;
} DSCCS;
```

Название	Тип	Назначение	Комментарий
counter0	DSCCS	Выход	Состояние счётчика 0
counter1	DSCCS	Выход	Состояние счётчика 1
counter2	DSCCS	Выход	Состояние счётчика 2

DSCWGCONFIG

Конфигурация генератора периодических сигналов. Параметр функции dscWGConfigSet.

```
typedef struct {
  DWORD depth;
  DWORD ch_per_frame;
  DWORD source;
} DSCWGCONFIG;
```

Название	Тип	Назначение	Комментарий
depth	DWORD	Вход	Размер кольцевого буфера генератора в отсчётах ЦАП, абсолютное значение от 64 до 1024 с шагом 64
ch_per_frame	DWORD	Вход	Количество каналов ЦАП на один такт генератора. Макрос WG_CH_PER_FRAME_*, см. раздел 4.12 «Генератор сигналов»
source	DWORD	Вход	Источник тактирования генератора WG_SRC_*, см. раздел 4.12 «Генератор сигналов»

DSCADCALPARAMS

Настройки автокалибровки АЦП. Параметр функций **dscADAutocal**, **dscADCalVerify**.

Название	Тип	Назначение	Комментарий
adrange	BYTE	Вход	Номер режима для автокалибровки. 255 для проведения калибровки по всем режимам. Соответствие чисел режимам см. в разделе 3.7 «Автокалибровка»
ad_offset	FLOAT	Выход	Смещение относительно опорного значения в кодах АЦП
ad_gain	FLOAT	Выход	Ошибка усиления относительно опорного значения в кодах АЦП

DSCDACALPARAMS

Настройки автокалибровки ЦАП. Параметр функций **dscDAAutoCal**, **dscDACalVerify**.

Название	Тип	Назначение	Комментарий
darange_calibrate	BYTE	Вход	Номер режима для автокалибровки. 255 для проведения калибровки по всем режимам. Соответствие чисел режимам см. в разделе 3.7 «Автокалибровка»
offset	FLOAT	Выход	Смещение относительно опорного значения в кодах АЦП
gain	FLOAT	Выход	Ошибка усиления относительно опорного значения в кодах АЦП

AICCS

Настройки счётчика. Параметр функции aicSetCounterSettings.

```
typedef struct {
  BOOL internal_clk;
  BOOL out_enable;
  BOOL gate_enable;
  DFLOAT rate;
  DWORD divider;
} AICCS;
```

Название	Тип	Назначение	Комментарий
internal_clk	BOOL	Вход	*internal_clk==TRUE* устанавливает внутренний источник тактирования
out_enable	BOOL	Вход	*out_enable==TRUE* разрешает вывод тактирующих импульсов на вывод внешнего разъёма
gate_enable	BOOL	Вход	Использования сигнала разрешения тактирования (GATE0/DIN0). При *gate_enable==TRUE* низкий уровень на входе GATE0/DIN0 запрещает тактирование счётчика.
rate	DFLOAT	Вход	Желаемая частота импульсов при внутреннем тактировании, имеет смысл при *internal_clk==TRUE*
divider	DWORD	Вход	Делитель частоты внешнего тактирования, имеет смысл при *internal_clk==FALSE*

5 Примеры программ

5.1 Периодическое аналого-цифровое преобразование

В этом примере происходит обработка 262144 отсчётов АЦП по каналу 0 с частотой преобразования 100 кГц. Данные обрабатываются функцией process_data(). В теле функции process_data() в момент вызова в буфере *buf* доступно *count* отсчётов. Размер кольцевого буфера — 16384 отсчёта.

Команда компиляции и сборки примера:

```
gcc -c -I. -O2 -o ad.o ad.c
gcc ad.o -o ad -L. -laic -lpthread -lrt
```

ad.c:

```
/*
 * Пример программы периодического аналого-цифрового преобразования.
 * Prosoft 2011
 */

#include <stdio.h>
#include <stdlib.h>

#include <aicud.h>

#define NUM_CONVERSIONS 16384

void process_data(DSCSAMPLE *buf, int count)
{
    // do something
}

int main()
{
    int rc;
    unsigned long last_total_transfers;
    unsigned long last_transfers;
    unsigned long new_sample_count;

    DSCADSETTINGS adsettings;
    DSCB board;
    DSCCB brd_conf;
    DSCAIOINT aio;
    DSCS status;

    if((rc = dscInit(DSC_VERSION)) != DE_NONE) {
```

```
        fprintf(stderr, "dscInit: %s\n", dscGetErrorString(rc));
        exit(1);
    }
    brd_conf.base_address = 0x100;
    brd_conf.int_level = 7;

    if((rc = dscInitBoard(DSC_AIC324, &brd_conf, &board)) != DE_NONE) {
        fprintf(stderr, "dscInitBoard: %s\n", dscGetErrorString(rc));
        exit(1);
    }
    aio.num_conversions = NUM_CONVERSIONS;
    aio.conversion_rate = 100000.0;
    aio.cycle = TRUE;
    aio.internal_clock = TRUE;
    aio.low_channel = 0;
    aio.external_gate_enable = FALSE;
    aio.internal_clock_gate = FALSE;
    aio.sample_values = malloc(sizeof(DSCSAMPLE)*NUM_CONVERSIONS);
    if(aio.sample_values == NULL) {
        perror("malloc");
        exit(1);
    }
    aio.fifo_enab = TRUE;
    aio.fifo_depth = 1024;
    aio.dump_threshold = 4096;
    aio.clksource = 0;

    adsettings.current_channel = 0;
    adsettings.gain = GAIN_1;
    adsettings.range = RANGE_10;
    adsettings.polarity = BIPOLAR;
    adsettings.load_cal = TRUE;
    adsettings.scan_interval = 0;

    rc = dscADSetSettings(board, &adsettings);
    if(rc != DE_NONE) {
        fprintf(stderr, "dscADSetSettings: %s", dscGetErrorString(rc));
        exit(1);
    }
    rc = dscADSampleInt(board, &aio);
    if(rc != DE_NONE) {
        fprintf(stderr, "dscADSampleInt: %s", dscGetErrorString(rc));
        exit(1);
    }
    last_total_transfers = 0;
    last_transfers = 0;
    do {
        dscSleep(20);
        dscGetStatus(board, &status);
```

```
        new_sample_count = status.total_transfers - last_total_transfers;
        if(!new_sample_count)
            continue;

        if(new_sample_count > NUM_CONVERSIONS) {
            dscCancelOp(board);
            fprintf(stderr, "Not processing data fast enough. %ld samples lost",
                    new_sample_count - NUM_CONVERSIONS);
            break;
        }
        if(status.overflows) {
            dscCancelOp(board);
            fprintf(stderr, "FIFO Overflow");
            break;
        }
        if (status.transfers > last_transfers) {
            process_data(&aio.sample_values[last_transfers],
                    status.transfers - last_transfers);
        } else {
            process_data(&aio.sample_values[last_transfers],
                    NUM_CONVERSIONS - last_transfers);
            process_data(&aio.sample_values[0], status.transfers);
        }
        last_transfers = status.transfers;
        last_total_transfers = status.total_transfers;
    } while(last_total_transfers < NUM_CONVERSION * 16);

    rc = dscCancelOp(board);
    if(rc != DE_NONE)
        fprintf(stderr, "dscCancelOp: %s\n", dscGetErrorString(rc));

    free(aio.sample_values);
    dscFreeBoard(board);
    dscFree();
    return EXIT_SUCCESS;
}
```