МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Физический факультет

Кафедра радиофизики

Магистрально-модульные системы автоматизации

Методические указания к лабораторной работе № 2
практикума ТСАНИ

Новосибирск
2015

УДК 53:004.9(075.8)
ББК В185.505
Ф278

Фатькин Г. А., Оттмар А. В., Зорин А. В. **Магистрально-модульные системы автоматизации**: Метод. указ. ; Новосиб. гос. ун-т. – Новосибирск : РИЦ НГУ, 2015. – 28 с.

Лабораторная работа посвящена изучению принципов построения магистрально-модульных систем автоматизации, а также комплекта оборудования National Instruments, используемого в практикуме.

В задачу студента входит работа с портом ввода-вывода, ЦАП, АЦП модуля NI PXI-6251 с помощью созданной в университете библиотеки tsanilib. Ознакомление с работой магистрально-модульных систем происходит на базе специализированной системы, построенной на основе шины Avalon.

Рецензент:
Е. А. Бехтенев

Ответственная за выпуск:
О. А. Тенекеджи

.

Введение

Магистрально-модульные системы автоматизации получили очень широкое распространение благодаря возможности объединения разнородных средств автоматизации в единую систему, и лёгкости реализации алгоритмов управления. Также немаловажным является расширяемость таких систем и наличие широкого арсенала готового оборудования.

Одной из первых магистрально-модульных систем, активно применявшихся в научных исследованиях, была система КАМАК (CAMAC – Computer Automated Measurement And Control). КАМАК до сих пор используется в институтах РАН, однако последнее время от него отказываются по причине устаревания аппаратной базы. Широко распространены современные стандарты VME, CompactPCI, PCI Express. В практикуме ТСАНИ используется оборудование, выполненное в стандарте PXI Express (PXIe), который представляет собой магистрально-модульную систему.

Стандарты, описывающие магистрали (шины) современных магистрально-модульных систем, весьма объёмны и достаточно сложны для понимания неспециалистами. Поэтому с принципами их работы вы будете знакомиться на примере упрощённой специализированной магистрали, построенной на базе шины Avalon.

Кроме того, в этой работе вы ознакомитесь с используемым в практикуме комплектом оборудования: портами ввода-вывода и цифро-аналоговыми (ЦАП, DAC – digital-to-analog converter) и аналогово-цифровыми (АЦП, ADC) преобразователями, а также программными средствами.

Принципы устройства магистрально-модульных систем автоматизации

Магистрально-модульные системы автоматизации представляют собой один или несколько элементных блоков (модулей), установленных в едином каркасе (крейте) и электрически связанных с помощью общей шины (магистрали). Соответственно, в любой такой системе можно выделить основные элементы: крейт, модули, магистраль.

Крейт – это механический каркас, в который устанавливаются модули. Кроме того, в крейте располагается магистраль, блок питания, система охлаждения. Часто крейты монтируются в специальную стойку.

Модули – это электронные блоки, выполняющие различные функции: измерения, генерации сигналов, хранения информации, преобразования сигналов и другие. Модули устанавливаются в крейт и подключаются к магистрали.

Обычно в системе выделяется одно или несколько управляющих устройств (модулей), которые называются контроллерами. Контроллер выполняет все процедуры по адресации остальных устройств, чтению/записи, обработке запросов и так далее. Контроллер может являться «интеллектуальным», то есть иметь в своём составе микро-ЭВМ. Другой тип контроллеров способен лишь обеспечивать обмен данными между управляющей ЭВМ и модулями. Физическое соединение, обеспечивающее связь с ЭВМ, выполняется посредством какой-либо стандартной или нестандартной линии связи/шины: Ethernet, CANbus, RS-232 и других.

Магистраль – это совокупность электрических проводников, соединяющих различные модули. Магистраль используется для передачи команд и данных по шинам и подачи питания на приборы.

Шина – это набор проводников, сгруппированных по функциональному назначению.

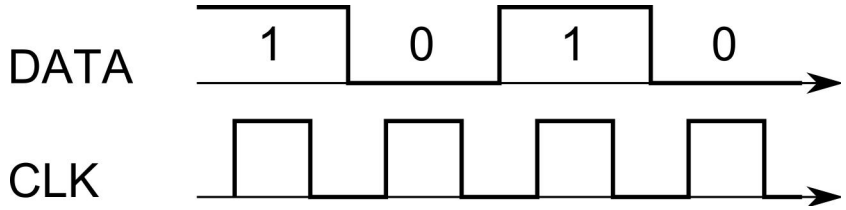

Рис. 1. Передача слова по последовательной шине

Различают два типа шин: параллельные и последовательные. По последовательным шинам одновременно передаётся только один бит, обычно синхронно с тактовым сигналом. На рис. 1 показан пример передачи по шине 1010. Понадобилось 2 провода и 4 такта.

По параллельной шине биты слова передаются по нескольким проводам одновременно. Например, передача 1010 по параллельной шине с тактовым сигналом может выглядеть так, как показано на рис. 2. Таким образом, для передачи 4 бит необходим всего 1 такт, а шина содержит 5 проводов.

Если говорить о шинах вообще, а не только о магистрально-модульных системах, то количество используемых проводников может быть весьма различным: многие шины (последовательные USB, RS-485, Ethernet, SATA) используют витые пары, то есть два проводника на один сигнал; также следует учитывать земли (в последних версиях параллельной шины PATA количество проводников удвоилось по сравнению с первыми версиями: рядом с каждым из 40 существовавших ранее проводников проложили новые земли), питания, экраны. В некоторых, обычно последовательных, шинах тактовый сигнал отсутствует, что уменьшает количество

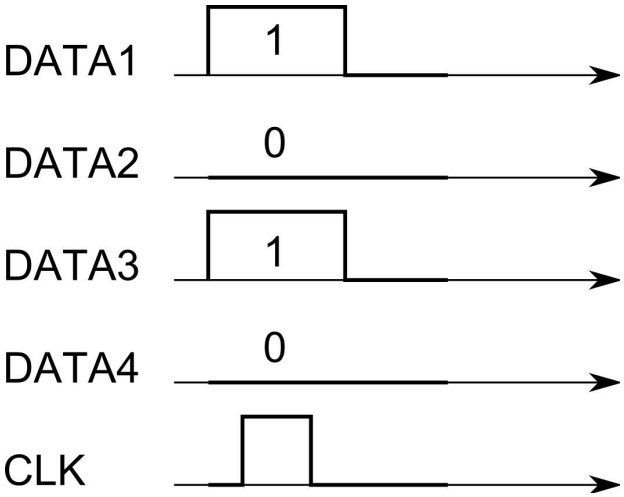

Рис. 2. Передача слова по параллельной шине

проводников за счёт усложнения приёмопередающих устройств. Данные могут быть объединены с питанием (ключ домофона, Power over Ethernet, Power-line communication).

В магистрально-модульных системах чаще всего применяются параллельные шины. Однако в последнее время прослеживается тенденция к переходу на последовательные и так называемые последовательно-параллельные шины (как, к примеру, в стандарте PCI-Express). Это объясняется в основном двумя причинами: 1) уменьшение количества проводов ведёт к уменьшению взаимных наводок и позволяет увеличить частоту сигнала, так что итоговая скорость передачи может даже возрасти; 2) уменьшение количества выводов микросхем ведёт к их удешевлению (выгода от уменьшения количества выводов значительно превышает расходы на усложнение кристалла) и уменьшению места, занимаемого на плате, что уменьшает цену конечного устройства.

Шины можно разделить на однонаправленные и двунаправленные. Однонаправленные шины позволяют передавать данные лишь в одном направлении – от источника к приёмнику. Двунаправленные шины позволяют источнику и приёмнику меняться местами. Существуют технологии (дуплекс), позволяющие использовать одни и те же провода для одновременной передачи данных в двух направлениях.

Основными на магистрали являются шины адреса, данных, управления, команд и прерываний. Существуют и другие виды шин, например, шины арбитража, состояния, которые мы рассматривать не будем.

Шина адреса предназначена для выбора модуля, установленного в крейт, а также для адресации внутри модуля (выбора субадреса). Адресация может выполняться различными способами: двоичное кодирование, географическая адресация и так далее. В системах PXIe, применяющихся в практикуме, используется географическая (позиционная) адресация – каждому месту в крейте отводится отдельная сигнальная линия, служащая для выбора блока, к которому производится обращение. Другие шины, общие для всех модулей, используются для задания субадреса.

Шина данных предназначена для передачи данных. Чаще всего разрядность шины (количество проводников, сгруппированных в шину) кратна 8: 8, 16, 24, 32 разряда. Значит, за один такт по шине могут быть переданы 1, 2, 3, 4 октета (байта).

Шина команд используется для управления операциями на магистрали. По шине передаются команды модулям, например: произвести чтение, обнулить данные, запретить генерацию запросов и тому подобное. Полный список команд, выполняемых модулем, определяется стандартом, в котором выполнена система автоматизации, и структурой модуля.

Для уменьшения общего количества проводников шины адреса, данных, команд могут выполняться *мультиплексированными,* то есть одни и те же линии используются для посылки, например, адреса и данных, но в разное время. Фиксация моментов (стробирование) передачи адреса и данных происходит по сигналам на шине управления.

Шина управления предназначена для синхронизации работы модулей или передачи информации. В состав шины управления могут входить: линии стробирования (по этим линиям передаются сигналы, разделяющие адрес, данные, и так далее на мультиплексированных шинах), тактовые линии (по ним передаются сигналы постоянной формы и частоты, чаще всего прямоугольный меандр, подаваемые на все модули одновременно для синхронизации работы модулей во времени), линии сброса и очистки (по ним передаются сигналы модулям перейти в начальное состояние).

Шина прерываний предназначена для определения того, что модуль запрашивает обработку какого-либо события: закончено измерение, требуются данные и так далее. Шины прерываний часто заводятся на управляющий процессор и позволяют аппаратно прерывать работу текущей программы для выполнения неотложных операций с модулем. При появлении сигнала на такой шине процессор переходит в заранее назначенную процедуру обработки прерываний. Прерывания часто имеют приоритеты, например, прерывание от системного таймера может быть более приоритетным, чем прерывание от внешнего устройства.

Контрольные вопросы

1. В чём отличие интеллектуальных и неинтеллектуальных контроллеров? Как вы думаете, чем определяется выбор типа контроллера для системы автоматизации?
2. Сколько проводников понадобится для реализации параллельной шины, в которой можно адресовать 7 модулей и передавать (читать и писать) в каждый модуль однобайтовые слова? В случае географической адресации модулей? А если использовать мультиплексированную шину адрес/данные?
3. Пусть тактовая частота шины 33 МГц. Сколько времени займёт передача 8-битового слова по шине, если шина параллельная, и сколько, если она последовательная? Сколько проводов потребуется в том и другом случае?
4. Если бы вам доверили разработку блока АЦП, для чего вы бы использовали шину прерываний? Опишите алгоритм процедуры обработки прерываний для этого случая.

Краткое описание PXI Express

Стандарт PXI Express

PXI Express (PXIe) – один из современных стандартов, описывающий правила построения магистрально-модульной системы. Используемое в практикуме оборудование соответствует спецификации PXI-5, разработанной в августе 2005 г. Разработкой и развитием стандарта занимается группа компаний PXI Systems Alliance.

Структура магистрали и логика работы шин в этом стандарте достаточно сложны. К счастью, детали работы шины будут скрыты от вас программным интерфейсом, поэтому в работе даётся самое общее описание этих шин.

Крейт и модули PXIe

Конструктивно PXIe-крейт и модули выполняются по стандарту «Евромеханика» [1, 2]. Возможны варианты исполнения в 3U и 6U форм-факторе (рис. 3), крейт 3U вы можете увидеть на своем рабочем месте, крейт 6U отличается от него высотой (примерно в 2 раза выше). Аналогично модули могут быть выполнены в варианте 3U и 6U.

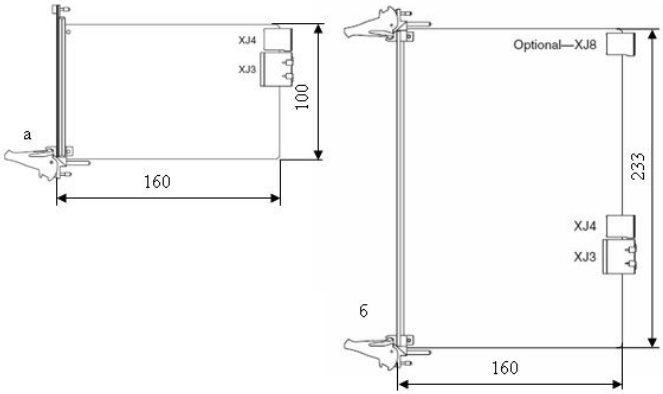

Рис. 3. Внешний вид модулей 3U (а) и 6U (б). в миллиметрах

Крейт, используемый в работе (рис.4.), имеет 8 разъёмов для установки модулей. Существуют крейты и с бóльшим количеством разъёмов для модулей (по спецификации крейт может иметь не более 31 разъёма для модулей).

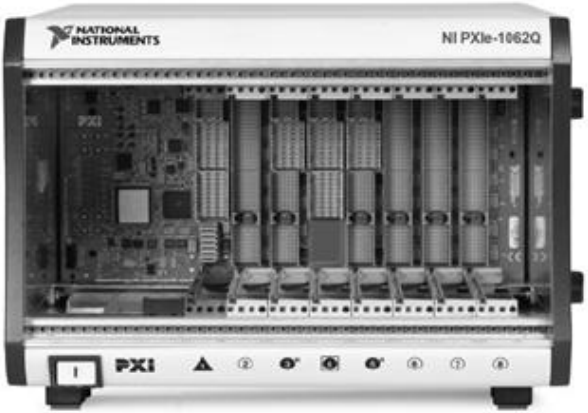

Рис. 4. Крейт NI PXIe-1062Q

В качестве контроллера используется модуль NI PXIe-8360. Он занимает один разъём, пространство слева от разъема № 1 остается свободным. К передней панели контроллера подключен кабель, соединяющий контроллер с PCIe платой NI PXIe-PCIe8361, установленной в компьютере. С помощью платы PXIe-PCIe8361 и модуля PXIe-8360 происходит управление крейтом от ЭВМ.

Особенностью конструкции PXIe является возможность использования в системе модулей, выполненных в разных стандартах семейства PCI: PXIe, cPCIe, PXI, cPCI. Для этого магистраль крейта PXIe имеет универсальный набор разъёмов.

Рис. 5. Знаки обозначения для разъёмов крейта: *а* – разъём модуля контроллера; *б* – разъём периферийных модулей PXI; *в* – гибридный разъём PXIe; *г* – разъём периферийных модулей PXIe; д – разъём тактового модуля

Рис. 6. Знаки обозначения для модулей: *а* – модуля контроллера; *б* – периферийный модуль PXI; *в* – периферийный модуль PXIe; *г* – тактовый модуль

Распознать, в какую позицию следует вставлять модуль, можно по разъемам модуля; кроме того, на каждом модуле должен присутствовать специальный знак, позволяющий распознать тип модуля, аналогичные знаки с указанием номера разъема нанесены на крейт (рис. 5, 6).

Порядок включения и выключения аппаратуры

Для того чтобы приступить к работе, необходимо включить стойку, крейт и компьютер.

Порядок включения:

1) включить стойку (чёрной кнопкой);
2) включить крейт;
3) включить компьютер.

Если вы включите сначала компьютер, а затем крейт, то в силу аппаратных особенностей системы (PXI подключается как PCI-bridge к шине PCI компьютера и определяется только при включении или перезагрузке), устройства PXI не будут доступны системе, и вы не сможете с ними работать. В таком случае достаточно перезагрузить компьютер.

Выключение производится в обратном порядке:

1) выключить компьютер;
2) выключить крейт;
3) выключить стойку (красной кнопкой).

Выключить крейт не получится до тех пор, пока компьютер работает.

Всё необходимое для работы программное обеспечение (ОС, LabWindows/CVI, драйвер NI-DAQmx) уже установлено на компьютере.

Оборудование и программные средства, используемые в работе

Модулем, которым вам предстоит в основном пользоваться в этой и нескольких последующих работах, является универсальный модуль ввода-вывода NI PXI-6251 [3], изображённый на рис. 7.

Рис. 7. Модуль NI PXI-6251

Модуль имеет в своём составе:

- 8 дифференциальных входных аналоговых каналов, подаваемых на 16-разрядный АЦП. Идентификаторы: *ai0...ai7,* от англ. Analog input – аналоговый вход;
- 2 аналоговых выходных канала – выходы 16-разрядных ЦАП. Идентификаторы: *ao0, ao1,* от англ. Analog output – аналоговый выход;
- 24 цифровых канала, разбитые на три 8-разрядных порта ввода-вывода. Идентификаторы: *port0...port2;*
- два 32-разрядных цифровых счётчика. Идентификаторы: *ctr0, ctr1.*

К модулю подключается терминальный блок, изображённый на рис. 8. На терминальный блок выведены все цифровые, входные и выходные аналоговые каналы модуля NI PXI-6251. Упрощённая схема подключения каналов показана на рис. 9.

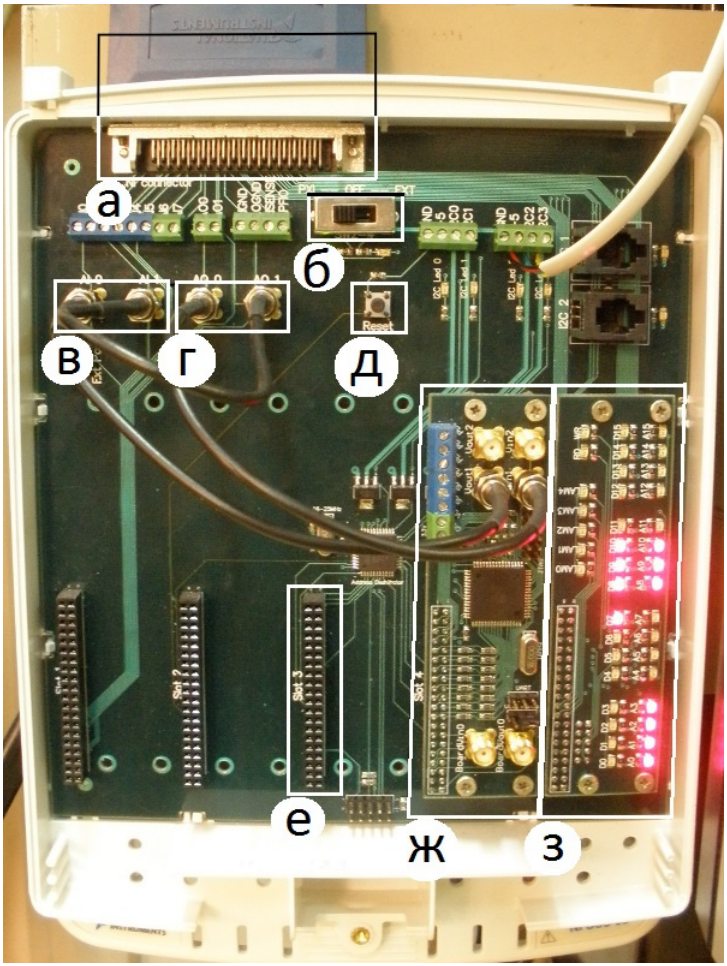

Рис. 8. Фотография терминального блока: *а* – разъём для подключения кабеля к NI PXI-6251, *б* – переключатель питания, *в* – аналоговые входы AI0, AI1, *г* – аналоговые выходы AO0, AO1, *д* – кнопка перезагрузки, *е* – разъём для подключения платы, *ж* – модуль ЦАП-АЦП, *з* – плата индикатора магистрали

Аналоговые входы и выходы выведены на разъёмы SMA, которые вы можете коммутировать нужным образом с помощью кабелей. Кабели SMA достаточно сложно вкручивать и выкручивать, поэтому подумайте, в каком порядке вам нужно скоммутировать сигналы, прежде чем приниматься за работу.

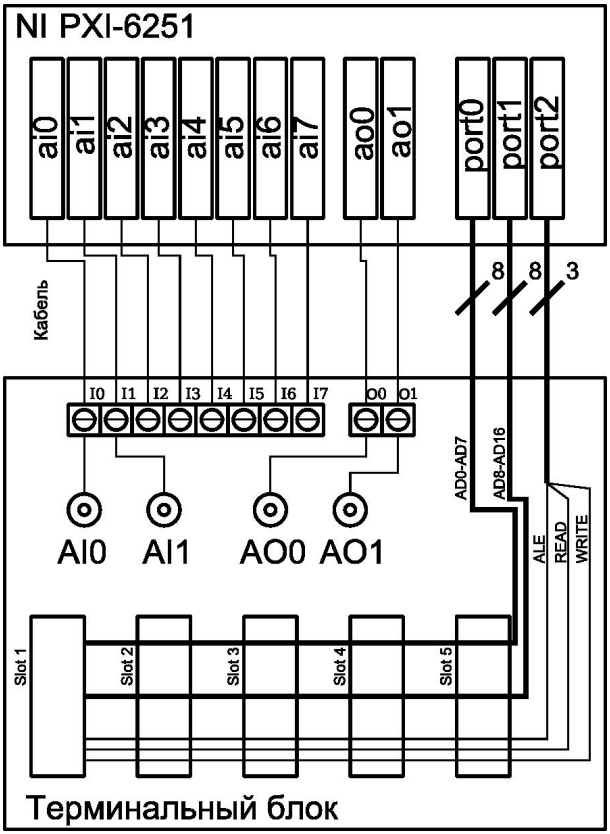

Рис. 9. Схема подключения терминального блока к NI PXI-6251.
Не показаны линии прерываний (port2 line3:7)

Включение терминального блока производится с помощью трёхпозиционного переключателя. Крайнее левое положение (PXI) соответствует включенному состоянию, среднее положение (OFF) – выключенному. Правое положение (EXT) соответствует внешнему питанию и в этой работе не используется.

Также на плате имеется кнопка перезагрузки модулей на специализированной магистрали. На терминальном блоке реализована специализированная магистраль, основанная на шине Avalon, работая с которой, вы на практике освоите принципы построения магистрально-модульных систем. В табл. 1 (см. на стр. 21) приведено соответствие выходов портов и названий сигналов на магистрали Avalon.

Работа с модулем NI PXI-6251

Работа с модулем NI PXI-6251 в среде LabWindows/CVI может осуществляется с помощью библиотеки NI DAQmx, однако в рамках ТСАНИ предлагается использовать библиотеку-обёртку tsanilib. Для её использования достаточно создать проект из шаблона «User Interface Application(TSANI)». Подробное описание функций библиотеки дано в справочных материалах.

Прежде всего вызовите функцию int ni6251Slot(int slot) с аргументом 2, если модуль вставлен во второй слот. После завершения работы следует вызвать функцию int ni6251Close( ):

```
#include "tsani.h"
...
ni6251Slot(2);
... // работа с модулем
ni6251Close();
```

Работа с аналоговыми каналами предельно проста. Для выходных каналов используется функция

```
int analogOut(int idx, double val),
```

для входных

```
int analogIn(int idx, double* val).
```

Аргумент idx – номер канала (0 или 1), val – напряжение в вольтах. В случае успеха возвращается 0.

Чтобы понять работу с цифровыми каналами, сначала разберёмся, что такое порт ввода-вывода.

Порт ввода-вывода

Порт ввода-вывода – это электронное устройство, служащее для приёма и передачи цифровых сигналов. В случае транзисторно-транзисторной логики (TTL) логический 0 соответствует напряжению 0...0,8 В, логическая 1 – 2,4...5 В. Напряжение более 5 или менее 0 В может вывести устройство из строя, а напряжение 0,8...2,4 В – средний диапазон, который различными логическими элементами трактуется по разному, при нормальной работе линии не должны находиться под таким напряжением (процесс перехода из одного состояния в другое мы не рассматриваем). Если линия порта настроена на вывод, она подключается к соответствующему источнику напряжения (направление тока может быть любым). В случае настройки линии порта на ввод напряжение определяется внешним источником сигнала, а линия имеет большой импеданс. Можно считать, что линия в режиме ввода – это вольтметр, точнее, однобитный ана-

логово-цифровой преобразователь. Обычно порт состоит из нескольких линий (8, 16, 24, 32). Количество линий называют разрядностью порта.

Модуль NI PXI-6251 имеет три 8-разрядных (всего 24 разряда) порта с TTL уровнями, причём каждая линия может быть настроена на ввод или вывод независимо от остальных. Порты позволяют выдавать и принимать сигналы с частотой до 10 МГц как с внешним, так и внутренним тактированием. Ток канала может достигать 24 мА, полный ток через все каналы 448 мА.

Программно порт можно представить тремя регистрами: чтения (read), записи (write) и маски (mask).

При записи в бит маски 0 соответствующая линия настраивается на ввод, то есть регистр записи игнорируется, а регистр чтения отображает уровень, установленный внешним устройством на линии (рис. 10).

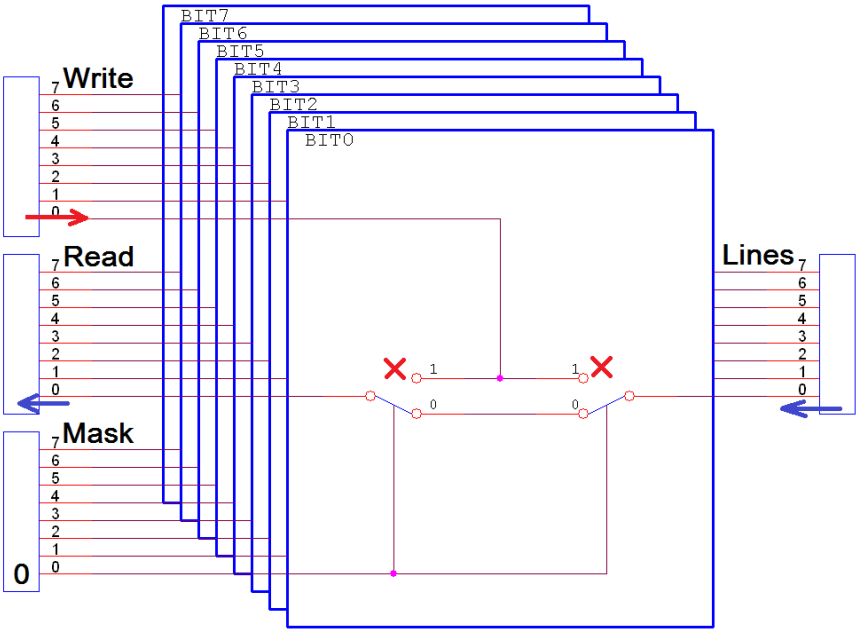

Рис. 10. Порт ввода-вывода в режиме ввода

Когда в маске записана 1, на линии оказывается напряжение, соответствующее биту в регистре записи. Он же дублируется в регистре чтения (рис. 11). Если к линии, настроенной на вывод, подключить внешний источник напряжения, последствия могут быть различными. При достаточно большом сопротивлении линии рядом с каждым устройством будет своё напряжение, но тогда нельзя говорить о едином на всей линии уровне сиг-

нала. Если же сопротивление мало, а это так в большинстве случаев, может возникнуть большой ток, выводящий устройства из строя.

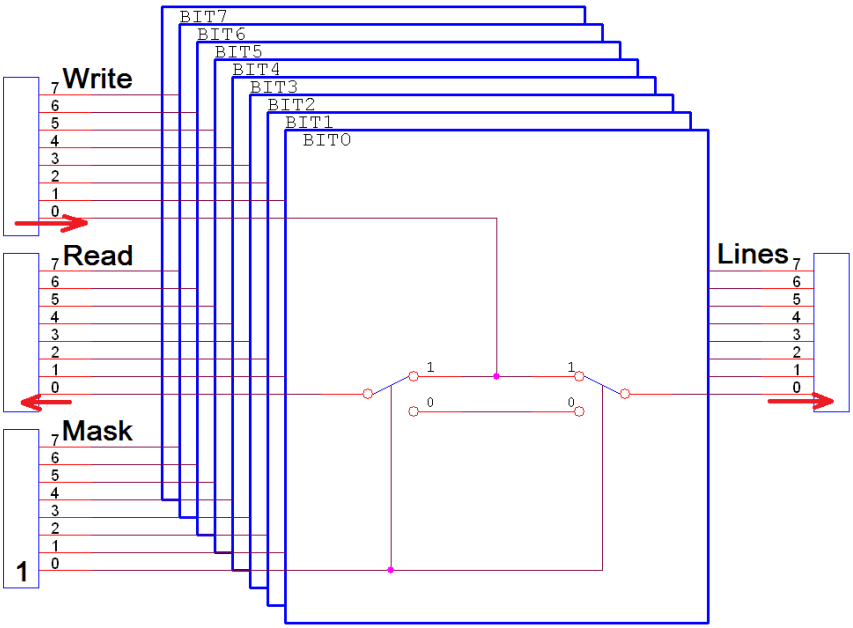

Рис. 11. Порт ввода-вывода в режиме вывода

Если перевести линию из режима вывода в режим ввода, следует подождать некоторое время, прежде чем обращаться к регистру чтения. Внешнее устройство может выставить напряжение, отличное от того, которое выставлял порт в режиме вывода. Так как ёмкость проводников ненулевая, требуется время, чтобы напряжение изменилось. Второй причиной является задержка обработки. Например, микроконтроллеры AVR (а именно такой лежит в основе платы ЦАП-АЦП магистрали Avalon) при обращении к регистру чтения выдают информацию о состоянии линии в некий момент от 0,5 до 1,5 тактов назад.

Выше описана лишь одна из возможных реализаций порта ввода-вывода. При работе с библиотекой tsanilib не требуется ждать после перевода линии из состояния записи в состояние чтения. При работе с шиной I²C вы познакомитесь со схемотехническим решением, позволяющим безопасно выводить данные на линию нескольким устройствам одновременно. Некоторые реализации портов ввода-вывода не умеют настраивать часть линий на ввод, а часть на вывод. Для защиты от изменений уровня в случае, когда линия настроена на ввод, но к ней ничего не подключено,

применяют притягивающий к земле (pull-down) или питанию (pull-up) резистор большого номинала.

Библиотека tsanilib содержит 3 функции для работы с портами ввода-вывода:

```
// запись в регистр маски:
int portMask(int portN, unsigned char mask);

// запись в регистр записи:
int portOut(int portN, unsigned char data);

// чтение из регистра чтения:
int portIn(int portN, unsigned char* data);
```

Во всех случаях аргумент portN – это номер порта (0...2). В случае успеха возвращается 0.

Ещё раз напомним, что каждая линия модуля NI PXI-6251 работает независимо (часть может быть настроена на вывод, а часть на ввод). Однако библиотека tsanilib позволяет одновременно обращаться ровно к одному регистру одного порта.

Для знакомства с портами ввода-вывода предлагается использовать плату индикатора магистрали. Если установить логическую 1 на линию 0 порта 2, на линии 1 и 2 установить 0, а остальные линии этого порта настроить на чтение, то можно считать, что к портам 0 и 1 подключены по 8 светодиодов, расположенных на индикаторе магистрали. При записи единицы в линию зажигается подключенный к ней светодиод, при записи нуля он гаснет. В практическом задании 3 объясняется, почему терминальный блок работает именно так.

Убедитесь, что оборудование работает. Для этого используйте программу из D:\tsani\тесты\.

Пример консольной программы, выполняющей несколько переключений светодиодов порта 0:

```
// обычные библиотеки LabWindows/CVI
#include <ansi_c.h>
#include <cvirte.h>
#include <userint.h>
#include "toolbox.h"

#include "tsani.h" // библиотека tsanilib

int main () {
   unsigned char temp = 0;
   ni6251Slot(2); // инициализация модуля
```

```
   // переводим модуль в режим отображения состоя-
ния линий
   portMask(2, 0x07);
   portOut(2, 0x01);

   // гасим все светодиоды
   portMask(0, 0xff);
   portOut(0, 0x00);

   // включим светодиоды 3 и 7, подождём 1 с
   portOut(0, (1<<7) | (1<<3));
   Delay(1.);

   // дополнительно включим светодиод 2
   portIn(0, &temp);
   temp |= 1<<2;
   portOut(0, temp);
   Delay(1.);

   // теперь выключим светодиод 7
   portIn(0, &temp);
   temp &= ~(1<<7);
   portOut(0, temp);
   Delay(1.);

   // переключим младшие 4 светодиода, а старшие
оставим как есть
   portIn(0, &temp);
   temp = (temp & 0xf0) | ((~temp) & 0x0f); // или
можно написать так: temp = (temp | 0x0f) &
((~temp) | 0xf0);
   portOut(0, temp);
   Delay(1.);

   ni6251Close(); // деинициализация модуля
   return 0;
}
```

Научитесь зажигать и гасить нулевой светодиод, другими словами, писать в линию 0 порта 0. В графическом интерфейсе пользователя сделайте 2 кнопки типа Command Button. Добавьте индикатор (LED), управляйте им одновременно со светодиодом.

Практическое задание 1. Работа с портом ввода-вывода

Рис.12. Графический интерфейс программы управления светодиодами

Напишите программу с пользовательским интерфейсом, аналогичным показанному на рис. 12. В обычном режиме работы (сразу после запуска программы) по нажатию на кнопку должен переключаться соответствующий индикатор в программе и светодиод на индикаторе магистрали, управляемый портом 0. Кнопка бегущего огня (Running Light) включает другой режим: каждые 0,25 секунды состояние каждого индикатора и светодиода должно меняться на то, которое было у предыдущего, а 0-й берёт состояние 7-го. Программа должна позволять переключиться обратно в обычный режим.

Обратите внимание на следующие ограничения. На первый взгляд они могут показаться ненужными, так как программу возможно написать и без их соблюдения, однако с точки зрения хорошего стиля программирования эти ограничения вполне естественны и не усложняют, а упрощают программирование.

В программе должен быть ровно один таймер. Период повторения 250 мс, никаких других задержек быть не должно. Таймер должен реализовывать только бегущий огонь. Желательно использовать функцию SetCtrlAttribute с атрибутом ATTR_ENABLED.

Состояние индикаторов всегда должно соответствовать состоянию светодиодов. Для этого в начале работы программы (после настройки порта 2 и до вызова функции RunUserInterface) погасите все индикаторы и светодиоды. В дальнейшем считайте, что актуальная информация хранится на шине, то есть при каждом событии (нажатие кнопки или тик таймера) нужно сначала прочитать данные из порта и работать именно с ними. Запрещается хранить данные в программе между событиями или читать состояния индикаторов. Индикаторы должны иметь Control Mode = Indicator и не иметь функции обратного вызова (callback).

Для управления индикаторами напишите функцию, принимающую аргумент типа unsigned char по аналогии с функцией PortOut. В теле функции достаточно вызвать SetCtrlVal по 1 разу на каждый индикатор (используйте условное выражение ?: на месте аргумента val).

Кнопка бегущего огня может быть Command Button либо Toggle Button. Остальные кнопки должны быть типа Command Button.

Все кнопки, кроме бегущего огня, должны иметь одну и ту же функцию обратного вызова (callback). Используйте параметр функции control и оператор switch.

Многие студенты замечают, что при последовательном добавлении элементов в редакторе интерфейса их численные идентификаторы в .h-файле также добавляются последовательно. Поэтому возникает соблазн обращаться к индикатором по символической константе нулевого индикатора и смещению (аналог арифметики указателей). Однако так делать нельзя. При изменении .uir-файла (например, удалите индикатор № 3, а потом создайте точно такой же) идентификаторы могут измениться, а символические константы нет. Если вам не нравится создавать 8 индикаторов и кнопок вручную (в редакторе графического интерфейса), можно сгенерировать их программно (читайте раздел Library Reference → User Interface Library → Overview → Creating a Graphical User Interface справки «LabWindows/CVI»).

Желающие могут работать с двумя портами (0 и 1) и увеличить количество индикаторов и кнопок до 16.

ЦАП и АЦП NI PXI-6251

Модуль NI PXI-6251 содержит два 16-разрядных ЦАП и 16-разрядный АЦП с мультиплексированным входом. Скорость выдачи напряжений ЦАП составляет 2,8 MS/s ($2{,}8 \cdot 10^6$ отсчётов в секунду). Диапазон выходных напряжений от −10 до +10 В. Диапазон является перестраиваемым.

АЦП измеряет сигналы в диапазоне от −10 до +10 В. Скорость измерений составляет 1 MS/s. Мультиплексор позволяет подключать вход АЦП к одному из 16 каналов. В приборе есть функция выбора входного диапазона. АЦП умеет работать с дифференциальным входом, а также измерять напряжение относительно различных точек схемы.

Для проверки работы ЦАП и АЦП (а также ЦАП и АЦП магистрали Avalon, см. задание 3) имеется тестовая программа в D:\tsani\тесты. Соединения должны быть следующие:

- AO0 → AI0
- Vout1 → AI1
- AO1 → Vin1

Практическое задание 2. Работа с ЦАП и АЦП

Напишите программу, интерфейс которой аналогичен показанному на рис. 13, которая управляет напряжением ЦАП NI PXI-6251 и измеряет выставленное напряжение с помощью АЦП NI PXI-6251. Установите пределы задания и измерения напряжений −11...+11 В. Соедините выход ЦАП (AO) и вход АЦП (AI) на терминальном блоке с помощью кабеля и измерьте сгенерированное ЦАП напряжение с помощью АЦП. Исследуйте

следующее: одинаковость измеренного и поданного напряжений; одинаковость нескольких измерений при неизменном поданном напряжении; поведение на границах диапазонов приборов.

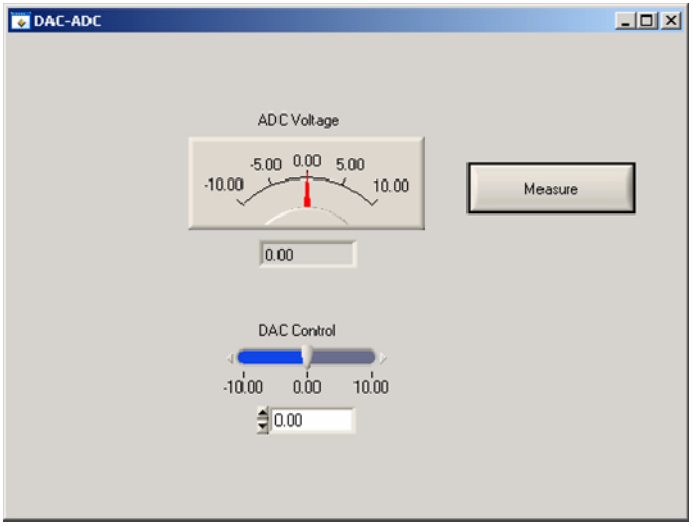

Рис. 13. Пример графического интерфейса для задания 2

Специализированная магистраль Avalon

Как уже отмечалось, протоколы современных магистралей весьма сложны для изучения, и подробности работы с ними остаются скрытыми от пользователя слоями абстракций (драйверами ОС, библиотеками функций, программными комплексами). Поэтому для изучения вам предлагается специализированная магистраль с протоколом Avalon.

Устройство и принципы работы магистрали Avalon

Магистраль Avalon реализована на терминальном блоке. Модули представляют собой небольшие платы, вставляющиеся в разъёмы. Всего разъёмов на магистрали 5, поэтому возможно одновременное подключение пяти плат. Вам предстоит написать программу, выступающую в роли контроллера магистрали.

Магистраль содержит следующие шины:

- AD0...AD15 – мультиплексированная двунаправленная шина адрес / данные;
- ALE, READ, WRITE – три сигнала, образующие шину управления. Эта шина всегда пишется контроллером, а модули только читают её.

Когда данные не передаются, ALE должен иметь низкий уровень, а READ и WRITE высокий (READ и WRITE являются инвертированными);

- INT1...INT5 – шины прерываний. Сигналы выставляются модулями (по одному прерыванию на каждый), читаются контроллером.

Соответствие линий портов шинам на магистрали приведено в табл. 1.

Таблица 1

Соответствие линий портов шинам магистрали Avalon

	line7	**line6**	**line5**	**line4**	**line3**	**line2**	**line1**	**line0**
port0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
port1	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
port2	INT5	INT4	INT3	INT2	INT1	WRITE	READ	ALE

Магистраль позволяет контроллеру производить чтение и запись 16-разрядных слов в 16-битном адресном пространстве, а модулям устанавливать и сбрасывать прерывания.

Способ адресации изображён на рис. 14. Адресная посылка делится на две части: адрес платы (три младших бита 0:2) и субадрес регистра (13 старших битов 3:15) в плате. С помощью адреса платы выбирается плата, к которой идёт обращение. Платы нумеруются с нуля слева направо: если вы хотите обратиться ко второй слева плате, то в биты адреса модуля следует записать 001_2. Адреса с 101_2 по 111_2 используются для работы с I^2C. Субадрес выбирает регистр в плате. Он может адресовать $2^{13} = 8192$ регистра. Описание регистров будет дано в разделе, описывающем соответствующую плату.

Рис. 14. Способ адресации на магистрали Avalon

Рассмотрим два типа операций, возможных на магистрали Avalon: запись 16-разрядного слова по определённому субадресу в модуль и чтение 16-разрядного слова из определённого субадреса модуля.

Цикл записи изображён на рис. 15 (по оси абсцисс отложено время, а по оси ординат – уровни соответствующих сигналов). Сначала на шину AD выставляется адрес, затем устанавливается в логическую единицу ALE (Address

Latch Enable – защёлка адреса) и удерживается не менее 4 мкс. В этот момент определяется, к какой плате идёт обращение, и эта плата запоминает, в какой субадрес будет вестись запись. Затем ALE опускается. После этого на шину AD устанавливается слово, которое нужно записать по переданному адресу, и опускается шина WRITE. В течение 4 мкс плата должна прочитать слово с шины. После этого контроллер поднимает WRITE.

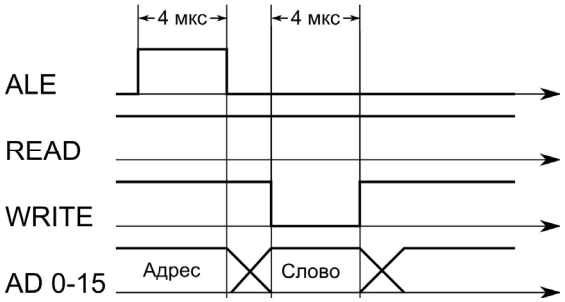

Рис. 15. Цикл записи на шине

Итак, если изначально ALE = 0, READ = 1, WRITE = 1, а шина AD настроена на запись, то процедура записи выглядит следующим образом:

- сформировать полный адрес из адреса платы и субадреса регистра
- записать полный адрес на шину AD
- записать 1 в ALE
- подождать не менее 4 мкс
- записать 0 в ALE
- записать данные на шину AD
- записать 0 в WRITE
- подождать не менее 4 мкс
- записать 1 в WRITE

Цикл чтения производится похожим образом и изображён на рис. 16. Отличие состоит в том, что вместо опускания WRITE происходит опускание READ, затем контроллер ждёт не менее 6 мкс, в течение которых модуль должен выставить данные. Теперь контроллер может прочитать их и поднять READ.

Запишем вышесказанное алгоритмически:

- сформировать полный адрес из адреса платы и субадреса регистра
- записать полный адрес на шину AD
- записать 1 в ALE
- подождать не менее 4 мкс
- записать 0 в ALE

- перевести шину AD в режим чтения
- записать 0 в READ
- подождать не менее 6 мкс
- прочитать данные с шины AD
- записать 1 в READ
- перевести шину AD в режим записи
- вернуть прочитанные данные

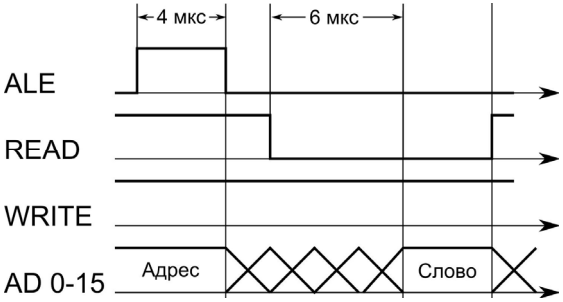

Рис. 16. Цикл чтения на шине

Так как работа с адресом в обоих случаях происходит одинаково, следует вынести эту часть кода в отдельную функцию. Задержки (4 и 6 мкс) явно прописывать необязательно, так как при использовании библиотеки tsanilib они возникнут сами собой.

В этой работе используются две платы: плата ЦАП-АЦП и плата индикатора магистрали, с которой вы уже работали в предыдущем задании. Плата ЦАП-АЦП предназначена для генерации и измерения аналоговых сигналов. Индикатор магистрали – для отображения работы магистрали. Подробнее о них будет рассказано далее.

Работа с шиной производится отдельными циклами. По окончании одного можно тут же начинать другой.

К каждой плате подведена своя шина прерывания. Такая шина всегда пишется платой, а контроллер имеет право только читать её. Обычно прерывания используются для того, чтобы сообщить о наступлении какого-либо важного события, точное время наступления которого заранее неизвестно, например, об окончании генерации или получения аналогового сигнала. К сожалению, имеющееся программно-аппаратное обеспечение не позволяет продемонстрировать существенных преимуществ прерываний над опросами по протоколу Avalon, так как сами линии прерываний всё равно приходится опрашивать. Поэтому в дальнейшем в этой работе мы будем говорить о бите готовности вместо прерываний. Снимается бит готовности обычно записью в некоторый регистр платы. Однако даже в

такой реализации чтение бита готовности гораздо быстрее чтения регистров модулей.

Плата индикатора магистрали

Плата индикатора магистрали предназначена для визуального отображения циклов шины. Светодиоды LAM0...LAM4 показывают состояние битов готовности INT1...INT5. Светодиоды RD, WR зажигаются, если на шине был произведён цикл READ или WRITE соответственно, и остаются зажжёнными до тех пор, пока не будет произведён следующий цикл.

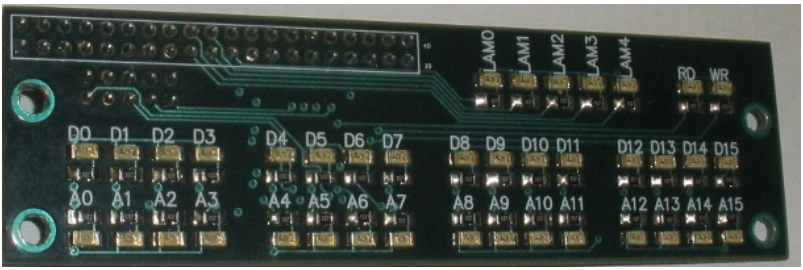

Рис. 17. Фотография модуля индикатора магистрали

Светодиоды A0...A15 показывают значение шины AD в рабочем цикле, которое запоминается по опусканию ALE. Светодиоды D0...D15 также показывают значение шины AD в рабочем цикле, но запоминаются по поднятию READ или WRITE. Таким образом, индикатор магистрали отображает состояние шины AD в последнем цикле.

Плата ЦАП-АЦП

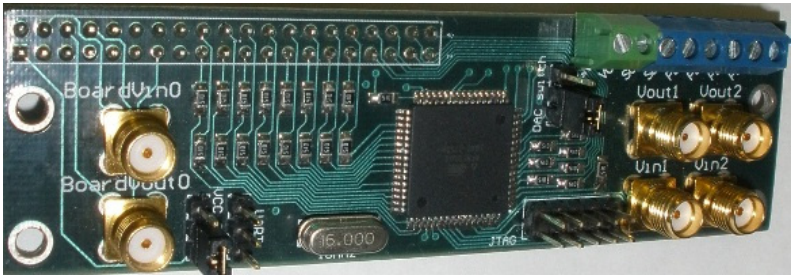

Рис. 18. Фотография модуля ЦАП-АЦП

Плата ЦАП-АЦП содержит в себе два 8-разрядных ЦАП и два 10-разрядных АЦП. ЦАП и АЦП управляются каждый своими регистрами. Работу с ЦАП и АЦП следует производить последовательно.

Цифро-аналоговый преобразователь

Диапазон ЦАП составляет 0...3,3 В. Разрядность 8 бит. Максимальная частота дискретизации 13,158 кГц (время установления выходного напряжения 76 мкс). ЦАП может выставлять прерывание после обновления (полного установления) сигнала на своём выходе. Для снятия прерывания следует использовать бит IACK в *регистре команд АЦП*. Каналы ЦАП выведены на SMA разъёмы Vout1 и Vout2. Все регистры ЦАП восьмиразрядные, то есть при записи в них старший байт отбрасывается, а записывается только младший. При чтении из них на старшие 8 бит также не стоит обращать внимания.

ЦАП управляется с помощью трёх регистров: *регистра управления ЦАП* и двух *регистров данных ЦАП*.

Регистр управления ЦАП (субадрес 0x00) содержит 3 бита:

Бит 2 = DIE (DAC Interrupt Enable) – разрешение подачи прерывания после обновления сигнала.

Бит 1 = DEN1 (DAC Enable 1) – разрешение генерации сигнала на Vout1.

Бит 0 = DEN2 – то же для Vout2.

Регистры данных ЦАП имеют субадреса 0x02 (ЦАП 1) и 0x03 (ЦАП 2).

Запись в регистр данных вызывает генерацию аналогового напряжения в соответствующем канале, если в регистре управления разрешена работа канала. Если разрешена подача прерывания, то по завершении установки напряжения будет выставлено соответствующее прерывание. Чтение из регистра возвращает записанные в нём данные.

Аналогово-цифровой преобразователь

Диапазон АЦП 0...2,56 В. Разрядность 10 бит. Входы АЦП выведены на SMA разъёмы Vin1 и Vin2. АЦП имеет память объёмом по 255 слов на каждый канал. Для управления АЦП используется 7 регистров: *регистр управления АЦП, регистр команд АЦП, регистр начального адреса ЗУ (запоминающего устройства), регистр конечного адреса ЗУ, регистр таймера, регистр данных АЦП 1, регистр данных АЦП 2*. Кроме того, есть *регистры прямого доступа к данным*. Регистры АЦП разной разрядности, поэтому при описании каждого из них указывается разрядность.

Регистр управления АЦП (разрядность 8 бит, субадрес 0x10):

X	X	X	ATS1	ATS0	AIE	AEN2	AEN1
7	6	5	4	3	2	1	0

AEN (ADC Enable) – разрешает работу соответствующего АЦП.

AIE (ADC Interrupt Enable) – разрешает устанавливать прерывание по окончании измерений. Запрет установки прерывания приводит к его сбросу.

ATS (ADC Timing Source) – выбор источника частоты дискретизации АЦП. 0 – оцифровка начинается сразу после команды START и проходит с максимальной частотой дискретизации (в этом режиме время между измерениями не фиксировано и составляет 112–115 мкс), 1 – частота дискретизации определяется внутренним таймером, 2 – данные оцифровываются по положительному фронту внешнего сигнала запуска, в этой работе не используется.

Регистр команд АЦП (разрядность 8 бит, субадрес 0x11):

X	X	X	X	X	SetI	IACK	Start
7	6	5	4	3	2	1	0

Start – запуск измерений. При чтении бит Start будет в 1 до тех пор, пока идут измерения. Как только измерения закончатся, бит Start будет установлен в 0.

IACK (Interrupt ACKnowledge) – подтверждение прерывания. Запись этого бита сбрасывает сигнал на шине прерывания.

SetI (Set Interrupt) – принудительная установка сигнала на шине прерывания. При чтении бит SetI всегда возвращает 0.

Регистр начального адреса ЗУ (разрядность 8 бит, субадрес 0x12). Используется для указания адреса (0...255), с которого начнётся запись оцифрованных значений в ЗУ.

Регистр конечного адреса ЗУ (разрядность 8 бит, субадрес 0x13). Используется для указания конечного адреса ЗУ. Всего будет оцифровано **(конечный адрес − начальный адрес + 1)** значений.

Регистры данных АЦП1 и АЦП2 (разрядность 10 бит, субадреса 0x16, 0x17). Используются для чтения оцифрованных данных по окончании измерения. Когда измерение закончено, счётчик адреса чтения для каждого из каналов устанавливается в начальный адрес. В регистры данных считываются значения из ЗУ. При каждом чтении из регистра данных счётчик соответствующего канала автоматически увеличивается на единицу, и в регистр данных записывается новое значение из ЗУ. Когда счётчик достигнет конечного адреса**,** его значение снова устанавливается в начальный адрес.

Внимание, не изменяйте регистры АЦП во время работы!

Регистр таймера (разрядность 16 бит, субадрес 0x14). Используется для установки частоты дискретизации АЦП. При установке в регистре

таймера значения t период дискретизации T вычисляется по формуле T = 1/62 500. Частота дискретизации f = 1/T. Задаваемая таким образом частота может принимать значения примерно от 0,95 Гц до 62,5 кГц. При записи 0 в регистр таймера блок будет работать практически с максимальной частотой дискретизации (в этом режиме время между измерениями не фиксировано и составляет 112...115 мкс).

Кроме того, АЦП предоставляет прямой доступ к данным в ЗУ с помощью регистров прямого доступа к данным. *Регистры прямого доступа к данным АЦП1* находятся по субадресам 0x100...0x1FF. *Регистры прямого доступа к данным АЦП2* – по субадресам 0x200...0x2FF.

Алгоритм получения единственного отсчёта может выглядеть следующим образом:

- записать регистр управления: разрешить работу желаемого ЦАП, разрешить установку прерываний, начинать работу сразу после сигнала START
- обнулить регистры начального и конечного адреса и регистр таймера
- одновременно установить биты Start и IACK в регистре команд
- циклически опрашивать бит готовности (шину прерываний) или бит Start в регистре команд, пока измерение не закончится
- прочитать регистр данных

Практическое задание 3. Работа с магистралью Avalon и платой ЦАП-АЦП

В результате выполнения этого задания вы должны получить 2 библиотеки: для работы с магистралью Avalon и использующую её библиотеку для работы с модулем ЦАП-АЦП.

1. Напишите библиотеку для чтения и записи шины Avalon. Сначала напишите функцию инициализации шины. Затем функцию записи шины, отладить её можно с помощью индикатора магистрали. Наконец, напишите функцию чтения шины. Для демонстрации библиотеки напишите программу с полями ввода адреса платы, субадреса регистра и данных, а также кнопками для запуска записи и чтения. Поле данных сделайте знаковым 32-битным (int). Подумайте, что будет, если сначала записать, а потом прочитать регистр данных ЦАП 1. Проверьте своё предположение для разных данных: маленьких и больших, положительных и отрицательных.

2. Напишите библиотеку для генерации напряжения через ЦАП платы ЦАП-АЦП. Создайте функцию, принимающую код, и функцию, принимающую напряжение в вольтах. При подаче недопустимых значений (менее 0 или более 3,3 В или 255) функции должны корректировать их. Для

проверки модифицируйте программу из задания 2: замените ЦАП NI PXI-6251 на ЦАП Avalon-платы ЦАП-АЦП.

3. Дополните библиотеку функциями работы с АЦП. Об окончании измерения необходимо узнавать путём опроса регистра команд. Для проверки модифицируйте программу из задания 2: замените АЦП NI PXI-6251 на АЦП Avalon-платы ЦАП-АЦП.

4. Дополните библиотеки функциями для работы с битами готовности. Пользовательский интерфейс должен содержать 2 кнопки: для установки прерывания (записи бита SetI в регистре команд АЦП) и для его снятия (записи бита IACK).

5. То же, что и в части 3, но вместо опроса бита START должен происходить опрос бита готовности. После этого бит нужно сбрасывать (добавьте в программу задержку, чтобы увидеть прерывание с помощью индикатора магистрали).

Список литературы

1. IEEE 1101.1-1991, IEEE Standard for Mechanical Core Specifications for Microcomputers Using IEC 603-2 Connectors.

2. IEEE 1101.10, IEEE Standard for Additional Mechanical Specifications for Microcomputers Using IEEE 1101.1 Equipment Practice.

3. NI625*x* Specifications http://www.ni.com/pdf/manuals/371291h.pdf

4. PCI Local BusSpecification rev. 2.3.

5. CompactPCI Specification rev. 3.0.

6. Более подробно с PXI и PXI Express можно ознакомиться на сайтах: http://www.ni.com/pxi/pxie.htm, http://www.pickeringtest.com/pximate, http://www.pxisa.org/Specifications.html.