Катупития Я.,
Бентли К.

Управление электронными устройствами на C++

ДМК ПРЕСС
ИЗДАТЕЛЬСТВО

УДК 621.3:004.438C++
ББК 31.26с
К29

Янта Катупития, **Ким Бентли**.

К29 Управление электронными устройствами на C++. Разработка практических приложений. / Перевод с англ. Бакомчев И. В. – М.: ДМК Пресс, 2016. – 442 с.

ISBN 978-5-97060-175-4

Книга предназначена всем, кому интересно изучение C++ и управление электронными устройствами на реальных и интересных примерах. Читателю предоставлена возможность научиться писать программы для выполнения конкретных задач, а не просто скучное изложение материала с картинками. Также рассказывается как создавать программы, взаимодействующие с внешними устройствами посредством специально разработанной интерфейсной платы. Книга, интерфейсная плата и предлагающееся программное обеспечение представляют собой набор простых и несложных для понимания устройств, таких как цифро-аналоговый преобразователь, аналого-цифровой преобразователь, устройство управления коллекторными и шаговыми электродвигателями, измерители температуры и напряжения, таймеры на базе компьютера и простое устройство сбора данных. Также материал книги содержит сведения из области автоматического управления, электроники и механотроники.

Издание будет полезно студентам, инженерам и научным работникам, техникам и радиолюбителям.

УДК 621.3:004.438C++
ББК 31.26с

ISBN 978-3-540-25378-5 (англ.)
ISBN 978-5-97060-175-4 (рус.)

ОГЛАВЛЕНИЕ

2 Базовые сведения о параллельном порте и работа с ним

Содержание главы:

- Устройство и функционирование параллельного порта.
- Основы логики.
- Системы счисления: десятичная, шестнадцатеричная и двоичная.
- Электроника: регистр, байт, синхронный, асинхронный, адреса.

2.1. Введение

Для эффективной работы с параллельным портом необходимо понимание основ логики и умение преобразовывать данные из одной системы счисления в другую. Здесь будут рассмотрены эти вопросы, а также описано устройство параллельного порта. Будут разъяснены такие базовые понятия, как двоичная логика, логические уровни, пространство адресов ввода/вывода и физическое подключение к порту.

Материал данной главы достаточен для использования параллельного порта в программах и подключения к нему. Эти знания понадобятся в дальнейших главах при создании программ для управления и контроля оборудования по параллельному порту.

Понимание основ цифровой схемотехники также полезно при сборке и наладке схем на интерфейсной плате.

2.2. Что такое параллельный порт?

В общих чертах порт представляет собой электронную схему, служащую интерфейсом для соединения с другим электронным устройством, чтобы обеспечить возможность обмена информацией. Такое соединение позволяет вводить информацию в порт, выводить из порта, а также осуществлять двунаправленную передачу данных через порт.

Параллельный порт обладает возможностью обмена данными между компьютером и внешним миром. Обычно он используется для передачи данных на принтер и иногда его называют портом принтера. В старых компьютерах порт принтера выполнен в виде схемы на отдельной печатной плате, которая вставляется в материнскую плату компьютера. В новых компьютерах порт принтера обычно уже встроен в материнскую плату.

Владение такими понятиями, как *семейства логических схем*, *логические уровни* и *запас помехоустойчивости* позволит лучше понимать цифровой обмен между устройствами. Эти знания окажутся полезными и при отладке цифровых схем на интерфейсной плате.

2.2.1 Цифровые логические схемы

Ранее упоминалось, что компьютерные программы выполняются электронными схемами двоичной логики, называемые также *цифровыми логическими схемами*. В двоичной логике есть только два возможных состояния: ВКЛЮЧЕНО и ВЫКЛЮЧЕНО. Обычно эти состояния двоичной логики представляются в *двоичном формате*, где 1 соответствует состоянию ВКЛЮЧЕНО, а 0 состоянию ВЫКЛЮЧЕНО.

Состояния ВКЛЮЧЕНО и ВЫКЛЮЧЕНО в параллельном порту, как и в большом количестве прочих цифровых схем, соответствуют уровням напряжения, называемыми *логическими уровнями*, обычно из диапазона 0...5 В. Заметим, что не у всех логических схем такие же логические уровни. Логические схемы могут быть в виде *интегральных схем*, в которых элементы схемы размещены на одном полупроводниковом кристалле, на так называемом *чипе*. Чип помещается внутрь пластмассового или керамического корпуса с металлическими выводами, соединёнными с чипом, которые служат для подключения внешних элементов.

Из логических схем наиболее распространены два типа (или *семейства*): ТТЛ (Транзисторно-Транзисторная Логика) и КМОП (Комплементарный Металло-Оксидный-Полупроводник). Эти семейства логики изготавливаются по разным технологиям, что приводит к различиям в эксплуатационных характеристиках. Основные различия электрических параметров семейств ТТЛ и КМОП представлены на **Рис. 2.1**. На рисунке показаны различия электрических параметров семейств логики. Следует отметить, что схемы некоторых семейств КМОП работают при напряжениях вне диапазона 0...5 В. Кроме того, выходное напряжение этих схем зависит от величины протекающего через выход тока.

Эти различия между семействами по логическим уровням имеют большое значение при взаимодействии логических схем различных семейств. В качестве примера рассмотрим варианты **а** и **б** на **Рис. 2.1**; в нашем случае микросхема ТТЛ передаёт ВЫСОКИЙ логический уровень (+2.4...+5 В) микросхеме КМОП. Микросхема ТТЛ в самом худшем случае может формировать ВЫСОКИЙ уровень не ниже +2.4 В без риска выйти из строя. Чтобы микросхема КМОП могла правильно воспринять ВЫСОКИЙ уровень, он должен составлять, по меньшей мере, +3.2 В и не превышать +5 В. Проблема заключается в том, что микросхема может ТТЛ формировать более низкие напряжения, вплоть до +2.4 В, что является недостаточным для правильного распознавания микросхемой КМОП ВЫСОКОГО уровня. Поэтому ВЫСОКИЙ логический уровень ТТЛ может быть ошибочно воспринят как НИЗКИЙ.

На **Рис. 2.1** проиллюстрирован также *запас помехоустойчивости* при передаче сигнала между логическими схемами одного и того же семейства. Рассмотрим случай, когда схема КМОП передаёт НИЗКИЙ логический уровень на вход другой схемы КМОП, **Рис. 2.1-г**. Передающая схема может формировать выходное напряжение 0...+0.2 В при соблюдении её норм эксплуатации, а принимающая схема, в свою очередь, воспринимает сигнал с уровнем 0...+1.5 В как НИЗКИЙ. Если такой выходной уровень принять равным +0.2 В, как худший случай, то помеха по напряжению для этого сигнала может достигать 1.5 – 0.2 = 1.3 В, при этом на входе принимающей схемы будет НИЗКИЙ логический уровень. В этом примере *запас помехоустойчивости* составляет 1.3 В.

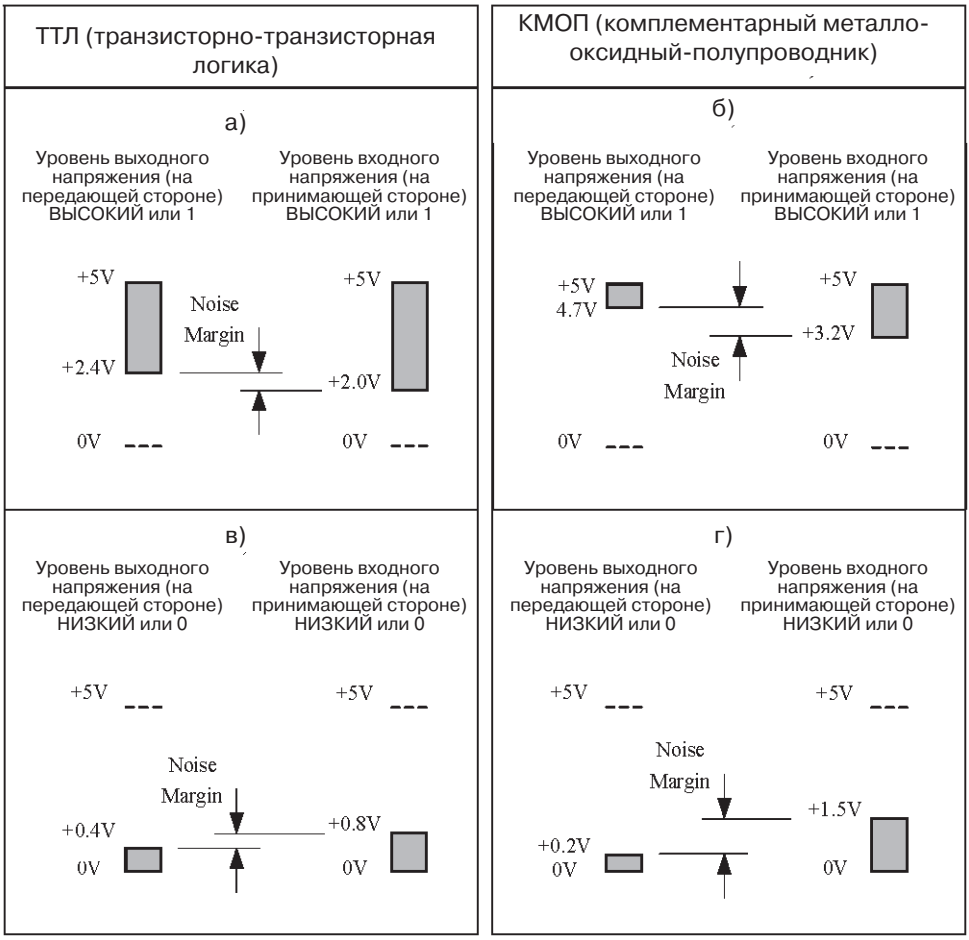

Рис. 2.1. Типичные логические уровни ТТЛ и КМОП (напряжение питания +5 В)

Если рассматривать аналогичный случай для элементов ТТЛ (**Рис. 2.1-в**), когда передаётся НИЗКИЙ логический уровень, можно увидеть, что запас помехоустойчивости составляет только 0.4 В. Обычно схемы КМОП обладают большей помехоустойчивостью, чем схемы ТТЛ. Схемы ТТЛ и КМОП также различаются по энергопотреблению, входным токам, нагрузочной способности по току и скорости переключения между состояниями. Дополнительная информация о семействах логических схем содержится в литературе, перечисленной в конце главы.

2.2.2. Устройство параллельного порта

Параллельный порт обеспечивает передачу данных для печати на принтер и приём информации о состоянии принтера. Данные от компьютера передаются по восьми линиям, что позволяет передавать в принтер *байты* информации. Байт это

группа из восьми *бит*, образующих в совокупности «порцию» данных. Каждая линия может передавать один бит данных. Бит может принимать два логических значения: 0 и 1. Остальные девять линий нужны для приёма информации о состоянии принтера и управления потоком данных. Эти девять линий разделяются на две группы: одна из пяти выходных линий, а другая из четырёх двунаправленных линий, **Рис. 2.2**. Физическое подключение к порту осуществляется посредством разъёма, обозначаемого D25F (буква D означает форму разъёма в форме буквы D). Разъём представляет собой 25-контактную розетку (буква F означает «розетка»). Кабель принтера имеет соответствующую 25-контактную вилку.

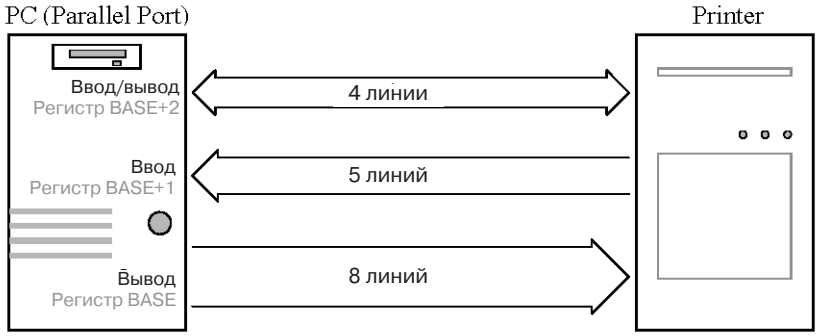

Рис. 2.2. Параллельный порт в общем виде

Три группы линий, показанные на **Рис. 2.2** изображают соединение между параллельным портом компьютера и внешним устройством, в данном случае с принтером. Каждая группа линий управляется или считывается через три регистра, последовательно расположенных в *адресном пространстве ввода/вывода*. Это адресное пространство образовано некоторым количеством ячеек памяти – *регистров*, позволяющих обмениваться данными с устройствами ввода/вывода. Это пространство памяти отличается от обычной памяти компьютера. Компьютер записывает данные в различные регистры ввода/вывода, где они сохраняются и могут быть считаны внешними устройствами. Другие регистры позволяют внешним устройствам записывать в них данные, которые затем могут быть считаны компьютером, а некоторые регистры допускают двунаправленный обмен данными.

Первый из этих трёх регистров в пространстве ввода/вывода имеет адрес BASE, **Рис. 2.3**. Этот регистр с наименьшим адресом и используется для обращения к другим регистрам параллельного порта путём прибавления констант к его значе-

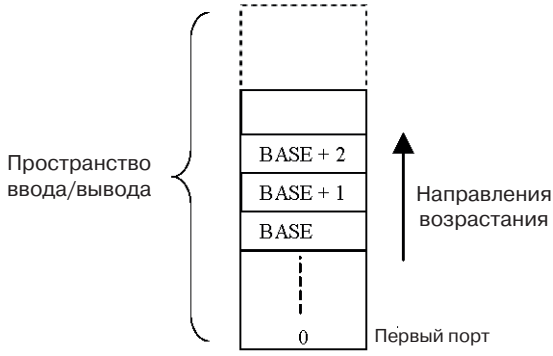

Рис. 2.3. Адресация пространства ввода/вывода

нию (т. е. адресу). Запись в регистр BASE приводит к выводу восьми бит данных (байта) в параллельный порт (**Рис. 2.2**), где каждый бит занимает отдельную линию.

Адрес следующего регистра группы на единицу больше адреса BASE и обозначается BASE+1. В регистр BASE+1 выведены пять входных линий параллельного порта. Можно только считывать состояния этих пяти сигналов.

Третий регистр обозначается BASE+2. Посредством этого регистра осуществляется управление четырьмя двунаправленными линиями параллельного порта. В этот регистр можно как записывать, так и считывать данные из него.

Примечание

Четыре двунаправленные линии регистра BASE+2 не являются непосредственными выходами логических схем. В параллельном порту к этим линиям часто подключаются резисторы и конденсаторы для уменьшения влияния помех. Это приводит к значительно более медленному переключению между состояниями и может вызвать ошибочное чтение данных при подключении логических схем различных семейств.

Также следует отметить, что разброс ёмкостей конденсаторов вызывает неодновременное (асинхронное) переключение состояний сигналов. Это асинхронное переключение выходов регистра BASE+2 может вызвать проблемы при передаче данных по синхронным интерфейсам.

В **Табл. 2.1** приведено соответствие битов данных во всех трёх адресах ввода/вывода, используемых параллельным портом, и контактов разъёма DB25. Каждый провод в соединительном кабеле между параллельным портом и внешним устройством (обычно это принтер) передаёт сигнал, соответствующий какому-либо регистру ввода/вывода. Биты в регистрах BASE и BASE+2 расположены последовательно, начиная с бита **D0**. Но в регистре BASE+1 биты располагаются с бита **D3**.

Таблица 2.1. Назначение контактов разъёма DB25 параллельного порта

Регистр BASE (8-битные выходные данные)	**Регистр BASE+1 (5-битные входные данные)**	**Регистр BASE+2 (4-битные двунаправленные данные)**
D0 – контакт 2		$\overline{D0}$ – контакт 1
D1 – контакт 3		$\overline{D1}$ – контакт 14
D2 – контакт 4		$\overline{D2}$ – контакт 16
D3 – контакт 5	D3 – контакт 15	$\overline{D3}$ – контакт 17
D4 – контакт 6	D4 – контакт 13	
D5 – контакт 7	D5 – контакт 12	
D6 – контакт 8	D6 – контакт 10	
D7 – контакт 9	$\overline{D7}$ – контакт 11	
Примечание: $\overline{D0}$ означает инверсию бита схемой параллельного порта.		

Некоторые биты по адресам BASE+1 и BASE+2 инвертируются схемой параллельного порта. Такие биты обозначаются чёрточкой над именем бита. Так же

обозначаются и сигналы на *принципиальной схеме* интерфейсной платы, на которой показаны все электрические соединения. При использовании таких битов в программе нужно учитывать их инверсию при чтении или записи данных порта.

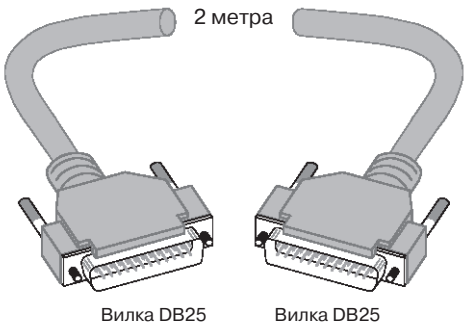

Рис. 2.4. Кабель DB25M-DB25M

Если нужно послать данные по одной из инвертированных линий порта, то в программе нужно инвертировать соответствующий бит. Такая двойная инверсия (в схеме порта и в программе) даёт в результате желаемый выходной сигнал. Аналогичным образом поступают при чтении инверсного бита порта, инвертируя нужные биты, чтобы получить их истинное значение. В программе инверсия реализуется всего одной строчкой кода, речь об этом пойдёт в разделе 3.6 следующей главы.

Контакты разъёма DB25 с 18 по 25 не показаны в **Табл. 2.1**. Они все подключены к земле компьютера и соединяются с интерфейсной платой кабелем, **Рис. 2.4**. Это кабель с вилками DB25 на концах, контакты соединяются отдельными проводами «один в один» (первый контакт одного разъёма соединён с первым контактом другого разъёма, аналогичным образом соединяются остальные контакты).

Примечание

Биты данных **D0**...**D2** по адресу BASE+1 не подключены к схеме параллельного порта. Это относится и к битам **D4**...**D7** регистра BASE+2. Чтение этих битов даст неопределённый результат.

2.3. Представление данных

Ранее говорилось, что компьютеры представляют данные в виде ***двух*** состояний ВКЛЮЧЕНО или ВЫКЛЮЧЕНО (ВЫСОКИЙ или НИЗКИЙ уровни напряжения) и поэтому называемые *двоичными*. Таким образом, данные представляются только двумя логическими состояниями (ВКЛЮЧЕНО/ВЫКЛЮЧЕНО). Числа в двоичной системе счисления представляются в виде суммы целых степеней числа 2, которые являются весом двоичного разряда. Мы можем сравнить эту систему с известной нам десятичной системой счисления. Десятичные числа образуются целыми степенями числа 10.

Например, десятичное число 25 можно представить так:

$$\begin{aligned} \mathbf{25} &= \mathbf{2}\times 10^1 + \mathbf{5}\times 10^0 \\ &= \mathbf{2}\times 10 + \mathbf{5}\times 1 \end{aligned}$$

Десятичное число 25 равно двоичному числу 11001:

$$\begin{aligned} \mathbf{11001} &= \mathbf{1}\times 2^4 + \mathbf{1}\times 2^3 + \mathbf{0}\times 2^2 + \mathbf{0}\times 2^1 + \mathbf{1}\times 2^0 \\ &= \mathbf{1}\times 16 + \mathbf{1}\times 8 + \mathbf{0}\times 4 + \mathbf{0}\times 2 + \mathbf{1}\times 1 \\ &= \mathbf{25}\ (\text{десятичное}) \end{aligned}$$

Примечание

В двоичном числе крайняя правая цифра имеет наименьший вес и называется *младший значащий бит (**L**east **S**ignificant **B**it, LSB*). Наоборот, крайняя левая цифра имеет наибольший вес и называется *старшим значащим битом (**M**ost **S**ignificant **B**it, MSB*).

Многоразрядные двоичные числа трудночитаемы. Для упрощения чтения пользуются более удобным представлением чисел, называемым *шестнадцатеричным*. В этом случае числа представляются посредством шестнадцати цифр.

В десятичной системе счисления используются десять арабских цифр от 0 до 9. В шестнадцатеричной системе нужно шестнадцать разных цифр. В качестве первых десяти используются арабские от 0 до 9, но ещё нужны цифры с одиннадцатой по пятнадцатую. В качестве этих цифр используются заглавные буквы латинского алфавита A, B, C, D, E и F.

В **Табл. 2.2** приведено соответствие десятичных, двоичных и шестнадцатеричных чисел.

Таблица 2.2. Преобразование между системами счисления

Десятичная	Двоичная	Шестнадцатеричная
$0 = 0\times 10^0$	$0 = 0\times 2^0$	$0 = 0\times 16^0$
$1 = 1\times 10^0$	$1 = 1\times 2^0$	$1 = 1\times 16^0$
$2 = 2\times 10^0$	$10 = 1\times 2^1 + 0\times 2^0$	$2 = 2\times 16^0$
$3 = 3\times 10^0$	$11 = 1\times 2^1 + 1\times 2^0$	$3 = 3\times 16^0$
$4 = 4\times 10^0$	$100 = 1\times 2^2 + 0\times 2^1 + 0\times 2^0$	$4 = 4\times 16^0$
$5 = 5\times 10^0$	$101 = 1\times 2^2 + 0\times 2^1 + 1\times 2^0$	$5 = 5\times 16^0$
$6 = 6\times 10^0$	$110 = 1\times 2^2 + 1\times 2^1 + 0\times 2^0$	$6 = 6\times 16^0$
$7 = 7\times 10^0$	$111 = 1\times 2^2 + 1\times 2^1 + 1\times 2^0$	$7 = 7\times 16^0$
$8 = 8\times 10^0$	$1000 = 1\times 2^3 + 0\times 2^2 + 0\times 2^1 + 0\times 2^0$	$8 = 8\times 16^0$
$9 = 9\times 10^0$	$1001 = 1\times 2^3 + 0\times 2^2 + 0\times 2^1 + 1\times 2^0$	$9 = 9\times 16^0$
$10 = 1\times 10^1 + 0\times 10^0$	$1010 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0$	$A = 10\times 16^0$
$11 = 1\times 10^1 + 1\times 10^0$	$1011 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0$	$B = 11\times 16^0$
$12 = 1\times 10^1 + 2\times 10^0$	$1100 = 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 0\times 2^0$	$C = 12\times 16^0$
$13 = 1\times 10^1 + 3\times 10^0$	$1101 = 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0$	$D = 13\times 16^0$
$14 = 1\times 10^1 + 4\times 10^0$	$1110 = 1\times 2^3 + 1\times 2^2 + 1\times 2^1 + 0\times 2^0$	$E = 14\times 16^0$
$15 = 1\times 10^1 + 5\times 10^0$	$1111 = 1\times 2^3 + 1\times 2^2 + 1\times 2^1 + 1\times 2^0$	$F = 15\times 16^0$
$16 = 1\times 10^1 + 6\times 10^0$	$10000 = 1\times 2^4 + 0\times 2^3 + 0\times 2^2 + 0\times 2^1 + 0\times 2^0$	$10 = 1\times 16^1 + 0\times 16^0$
$17 = 1\times 10^1 + 7\times 10^0$	$10001 = 1\times 2^4 + 0\times 2^3 + 0\times 2^2 + 0\times 2^1 + 1\times 2^0$	$11 = 1\times 16^1 + 1\times 16^0$

В программах иногда возникает необходимость выводить цифровые сигналы через параллельный порт в виде одного или нескольких байтов. Выходные сигналы являются набором логических сигналов, которые удобно представлять шестнадцатеричными цифрами. В других случаях нужно преобразовывать принятые от внешних устройств данные в шестнадцатеричные цифры. Следующие примеры демонстрируют преобразование двоичных чисел в шестнадцатеричные.

Мы можем получить шестнадцатеричное представление двоичного числа, если разобьём двоичное число на четырёхбитные группы (полубайты), начиная с младшего значащего бита. Заметим, что после разбиения байта на две группы по четыре бита мы получили две порции данных, называемых также *тетрадами*.

```
10001 = 1      0001
  = 1            1      (шестнадцатеричное)
```

Шестнадцатеричные числа часто записываются с префиксом 0x, например 0x11.

```
1010001101 = 10        1000   1101
  = 2                    8      D      (шестнадцатеричное 0x28D)
```

Шестнадцатеричные числа также обозначаются символами $ или H, например $11 или 11H.

Если нам нужно записать данные по двоичному адресу 1010001101, то мы сами должны преобразовать это число в шестнадцатеричный формат и использовать полученный адрес 0x28D в качестве адреса для записи данных. Если бы мы преобразовывали двоичное число в десятичное, то пришлось бы выполнять более сложное преобразование.

Теперь нам известно шестнадцатеричное представление чисел и в такой форме мы можем записывать адреса регистров параллельного порта. В большинстве компьютеров регистр BASE имеет адрес 0x378, но в некоторых моделях BASE может находится по адресам 0x278, 0x3BC или 0x300. В самом распространённом случае, когда BASE равно 0x378, адрес BASE+1 будет 0x379, а BASE+2 0x37A.

2.4. Программа для отображения шестнадцатеричных и десятичных чисел

Программа, приведённая в **Листинге 2.1**, преобразовывает числа из десятичной системы в шестнадцатеричную и наоборот. Шестнадцатеричные числа зачастую оказываются удобнее десятичных при работе с сигналами параллельного порта. Входные данные, считываемые из порта, наоборот, иногда удобнее выводить на экран в десятичной форме. Изучайте взаимосвязь десятичной и шестнадцатеричной систем счисления при помощи этой программы.

Листинг. 2.1. Программа для отображения чисел в десятичном и шестнадцатеричном форматах

```
/*********************************************
   Эта программа выводит на экран введённое
```

```
   десятичное число в шестнадцатеричном формате
*********************************************/

#include <iostream.h>

void main()
{
    int Number;

    cout << "Enter an integer number -> ";
    cin >> Number;
    cout << "The number is:" << endl;
    cout << dec << Number << " in decimal" << endl;
    cout << hex << Number << " in hexadecimal" << endl;
}
```

В **Листинге 2.1** заголовочный файл `iostream.h` позволяет использовать `cout` и аргумент преобразования числа `hex`. Переменная `Number` служит для хранения вводимого целого числа по запросу `"Enter an integer nember ->"` (Введите целое число). Это число затем выводится в двух следующих строках, сначала в десятичном виде, а затем в шестнадцатеричном. Шестнадцатеричный вид числа задаётся спецификатором формата `hex`.

2.5. Заключение

В этой главе было объяснены устройство параллельного порта и основные понятия цифровой логики, необходимые для использования параллельного порта с внешними схемами: двоичный формат, цифровые логические уровни, запас помехоустойчивости и семейства логических схем КМОП и ТТЛ.

Обмен данными по интерфейсу параллельного порта компьютера осуществляется посредством трёх регистров пространства ввода/вывода. Каждый регистр имеет размер один байт, при этом в двух регистрах некоторые биты не используются, а некоторые инвертируются схемой параллельного порта. Первый регистр используется только для вывода данных, второй – только для ввода, а последний – и для ввода и для вывода. Было объяснено представление чисел в десятичной, шестнадцатеричной и двоичной системах счисления. Эти знания нам понадобятся при создании программ в следующих главах.

2.6. Литература

1. Bergsman, P., *Controlling The World With Your PC*, HighText Publications, San Diego, 1994.
2. IBM, Technical Reference – Personal Computer AT, IBM Corporation, 1985.
3. NS CMOS, *CMOS Logic Databook*, National Semiconductor Corporation, 1988.
4. Wakerly, J. F., *Digital Design Principles and Practices*, Second Edition, Prentice Hall, 1994.

3 Тестирование параллельного порта

Содержание главы:

- Простое тестирование порта.
- Источник питания, интерфейс порта, буферы и управление светодиодами.
- Программирование в стиле С для полного использования параллельного порта.

3.1. Введение

Целью этой главы является создание программ, позволяющих изучить основные операции ввода/вывода параллельного порта компьютера. Контроль работы программы осуществляется при помощи простой тестовой схемы со светодиодами в качестве индикаторов. Перед тем, как приступить к работе с источником питания, параллельным портом и светодиодами, объясняется принцип их работы. Как только эти схемы будут собраны (в представленной последовательности), можно приступать к написанию и тестированию программ.

Сначала будут созданы программы, не являющиеся объектно-ориентированными, осуществляющие ввод/вывод данных через параллельный порт. Далее, в главе 5, мы познакомимся с объектно-ориентированным программированием (ООП). После прочтения этой главы вы будете знать принцип обмена данными по параллельному порту и станете лучше понимать простые программы на C++.

3.2. Источник питания интерфейсной платы

Для нормальной работы схемы на интерфейсной плате ей нужно электропитание постоянным стабилизированным напряжением. Источник питания в составе интерфейсной платы формирует все необходимые напряжения для питания её различных цепей. Эта часть платы должна быть собрана и отлажена раньше всех других схем. Другие схемы нужно собирать только когда источник питания будет давать нужные напряжения.

Схема источника питания показана на **Рис. 3.1**. Большая часть энергии поступает в интерфейсную плату из высоковольтного источника (из электросети) через трансформатор (силовой модуль). Выходы источника питания с напряжениями +5 В и +9 В могут поддерживать эти напряжения при токах нагрузки до 1 А. Для обеспечения максимальных значений тока каждый стабилизатор должен обладать соответствующим радиатором. Выход –8 В питается не от сети, а от батареи с напряжением +9 В, соответствующей требованиям этого выхода по напряжению и току. Выход –8 В предназначен для питания аналоговых цепей, работающих в

широком диапазоне питающих напряжений, вследствие этого нет необходимости в стабилизации этого напряжения.

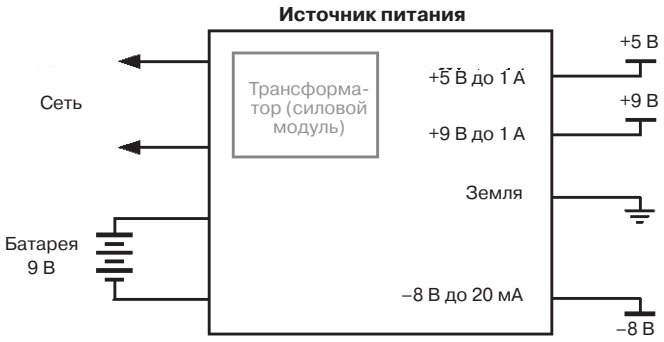

Рис. 3.1. Блок питания в обобщённом виде

На **Рис. 3.2** изображена более подробная функциональная схема источника питания. Батарея с напряжением 9 В питает схему через диод, проводящий ток только при соблюдении полярности подключения батареи. Проходящий через диод ток обуславливает падение напряжения на нём примерно на 1 В, поэтому на выходе получается –8 В.

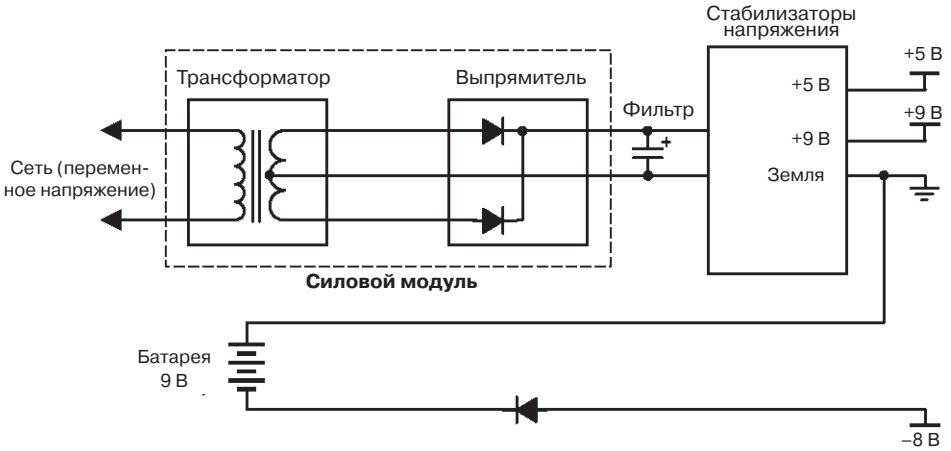

Рис. 3.2. Узлы источника питания

Сетевая часть источника питания состоит из четырёх узлов: трансформатора, выпрямителя с фильтром и двух стабилизаторов напряжения (на +5 В и +9 В). Совместная работа этих узлов приводит к выпрямлению сетевого напряжения и преобразованию его в стабилизированные напряжения +5 В и +9 В, где полностью отсутствуют колебания.

Переменное сетевое напряжение сначала должно быть уменьшено по амплитуде, затем выпрямлено и только потом уже может поступать на низковольтные схе-

мы. Эту функцию выполняет трансформатор внутри силового модуля. Там есть два диода, заставляющие протекать ток только в одном направлении, выпрямляя тем самым переменное синусоидальное напряжение трансформатора **(Рис. 3.3)**.

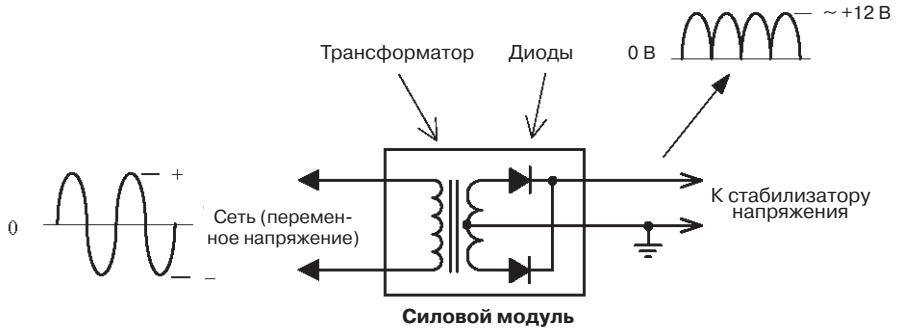

Рис. 3.3. Трансформаторный модуль источника питания (без конденсатора)

Для нормальной работы стабилизаторов напряжения необходимо подавать им на вход напряжение, превышающее выходное, по меньшей мере, на несколько вольт. К выходу силового модуля подключен конденсатор большой ёмкости, предотвращающий периодические «провалы» его выходного напряжения до нуля, обеспечивая тем самым сглаживание пульсаций напряжения до приемлемого уровня (**Рис. 3.4**).

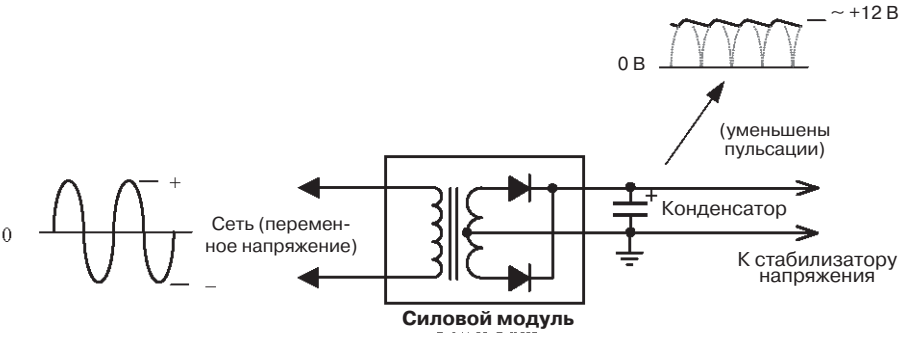

Рис. 3.4. Трансформаторный модуль с конденсатором на выходе.

На входы стабилизаторов напряжения подаётся пульсирующее напряжение **(Рис. 3.5)** и их внутренними схемами удерживается в пределах нескольких процентов от номинальных значений на их выходах. Стабилизатор +5 В, например, формирует выходное напряжение в диапазоне +4.75...+5.25 В. На интерфейсной плате ко входу и к выходу стабилизаторов подключено несколько конденсаторов, соединённых с землёй. Эти конденсаторы предотвращают появление на выходах стабилизаторов колебаний высоких частот и улучшают устойчивость стабилизаторов при быстро изменяющейся нагрузке.