Глава 2

Порт LPT в нестандартном режиме

Порт параллельного вывода данных LPT тоже может работать в нестандартном (необычном) режиме. Один из возможных нестандартных режимов работы LPT можно задействовать манипулируя битами вводимых (выводимых) данных. Подробности смотрите в *главе 3*.

Многочисленные опыты показали, что вывод команд со стороны компьютера очень хорошо выполняется посредством сигналов Data1…Data8 (выводы 2…8) порта, а ввод сигналов от внешнего аппаратного устройства на компьютер выполняется через выводы 10, 11, 12 и 15.

Мною разработан в учебных целях проект, исполняемый файл которого **lpt_port2.exe** и исходные коды проекта можете найти на прилагаемом к книге компакт-диске.

На рис. 2.1 показана рабочая форма проекта с установленными на ней компонентами.

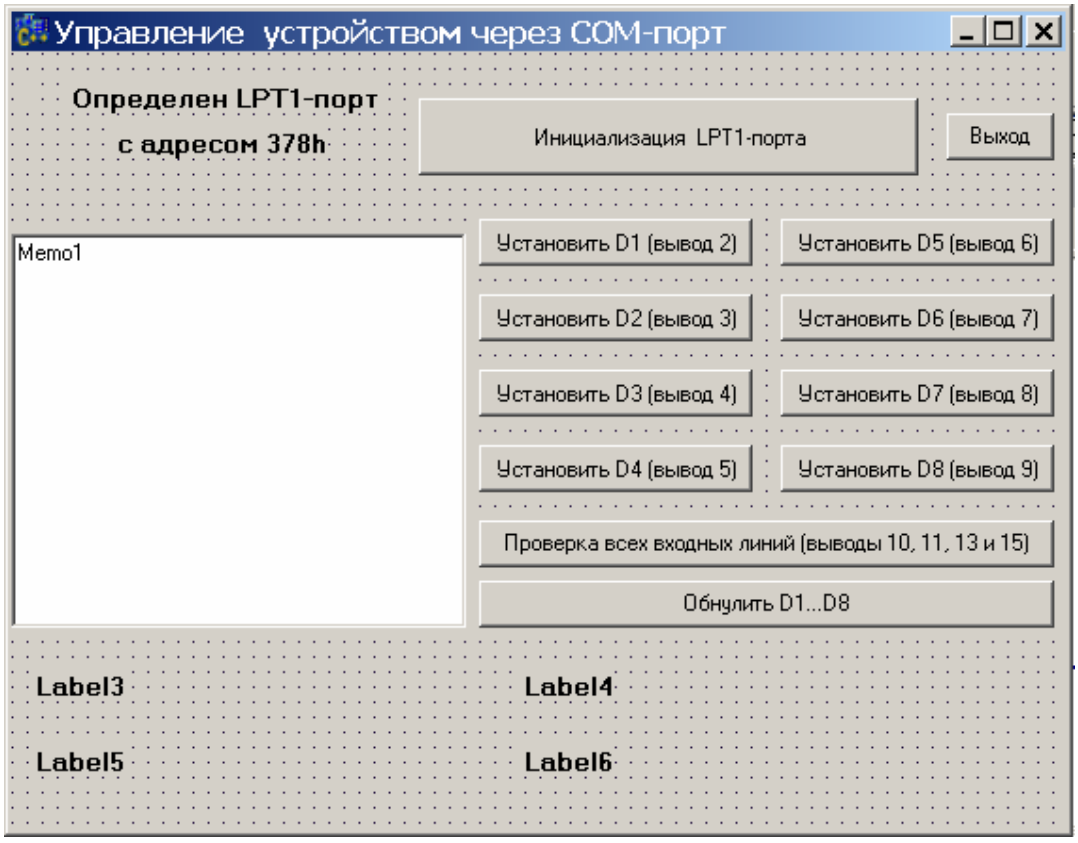

Рис. 2.1. Рабочая форма проекта

В листинге представлены исходные коды заголовочного файла **sport_a.h**. В этом файле объявлены все задействованные в проекте компоненты и функции.

Листинг 2.1. Файл *sport_a.h*

```
//---------------------------------------------------------------------
#ifndef sport_aH
#define sport_aH
//---------------------------------------------------------------------
```

```
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Buttons.hpp>
//---------------------------------------------------------------------------
class TForm1 : public TForm
{
__published:     // IDE-managed Components
        TButton *Button1;
     TLabel *Label1;
        TLabel *Label2;
        TSpeedButton *SpeedButton1;
        TMemo *Memo1;
        TSpeedButton *SpeedButton2;
        TSpeedButton *SpeedButton3;
        TSpeedButton *SpeedButton4;
        TLabel *Label3;
        TLabel *Label4;
        TLabel *Label5;
        TSpeedButton *SpeedButton5;
        TButton *Button3;
        TSpeedButton *SpeedButton6;
        TSpeedButton *SpeedButton7;
        TSpeedButton *SpeedButton8;
        TSpeedButton *SpeedButton9;
        TLabel *Label6;
        TButton *Button2;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall SpeedButton1Click(TObject *Sender);
        void __fastcall SpeedButton2Click(TObject *Sender);
        void __fastcall SpeedButton3Click(TObject *Sender);
        void __fastcall SpeedButton4Click(TObject *Sender);
        void __fastcall SpeedButton5Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
        void __fastcall SpeedButton6Click(TObject *Sender);
        void __fastcall SpeedButton7Click(TObject *Sender);
        void __fastcall SpeedButton8Click(TObject *Sender);
        void __fastcall SpeedButton9Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
private:    // User declarations
public:          // User declarations
        __fastcall TForm1(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TForm1 *Form1;
//---------------------------------------------------------------------------
#endif
//---------------------------------------------------------------------------
```

Файл создается автоматически в прочесе работы над проектом и никаким изменениям, без веских на то обстоятельств, подвергаться не должен.

На рис. 2.2 показано рабочее окно программы **lpt_port2.exe** в процессе работы. Программа предназначена только для работы с портом LPT1.

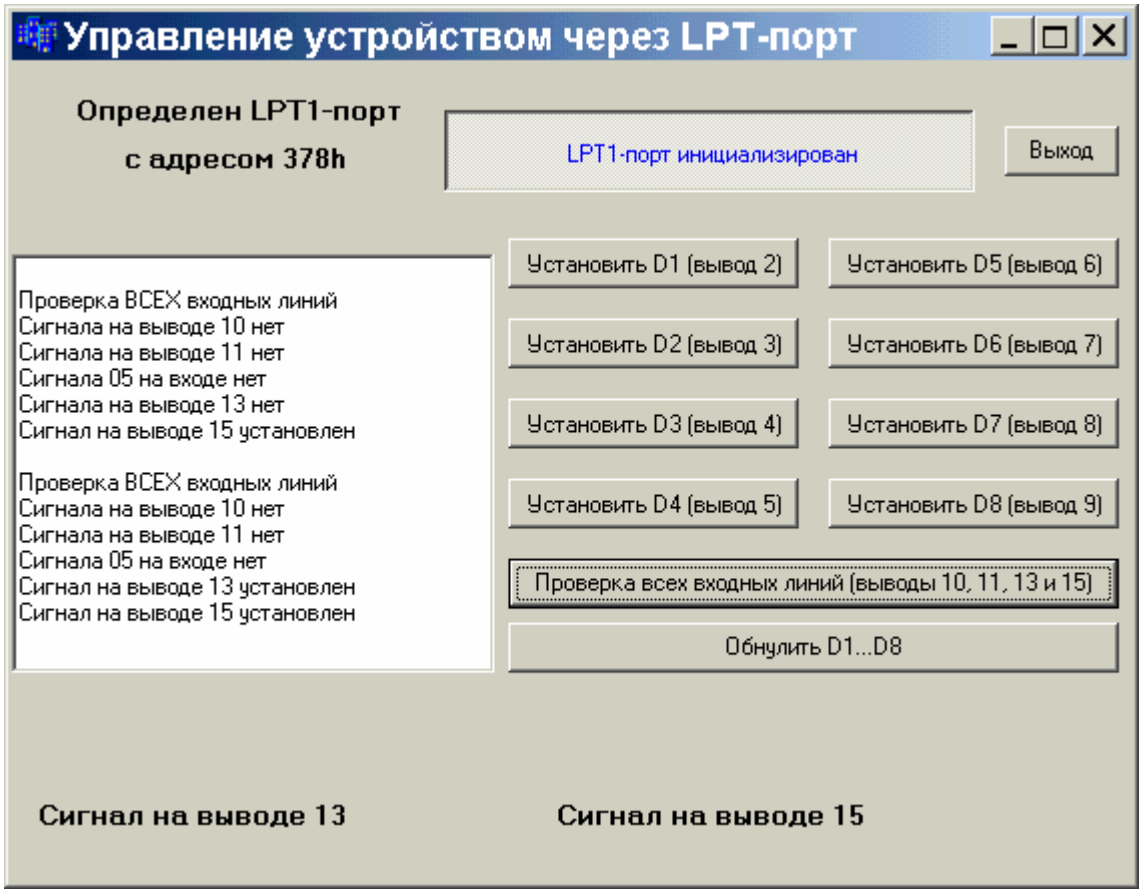

Рис. 2.2. Рабочее окно экспериментальной программы

Вначале следует провести инициализацию порта, в процессе которой в рабочие биты D1…D8 будут установлены положительные напряжения – единицы. Ниже располагаются клавиши, посредством которых в тот или иной бит устанавливается значение NULL. Получается так, что работа выполняется с инверсными величинами.

> Это может потребовать установки на вводную линию каждого байта отдельное инвертирующее устройство в виде микросхемы обычной ТТЛ логики. Следует помнить, что LPT-порт работает с сигналами напряжением не более +5В.

В листинге 2.2 привожу исходные коды файла ***sport_a.cpp***. В первой секции листинга 2.2 выполнено подключение заголовочных файлов, в следующей секции – объявление задействованных компонентов и глобальных переменных величин.

Листинг 2.2. Файл *sport_a.cpp*

```
//---------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#pragma inline              //разрешение работать с Ассемблером
#include "sport_a.h"
#include <conio.h>
#include <stdio.h>
```

```
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
 TMemo *Memo1;
 TLabel *Label3;
 TLabel *Label4;
 TLabel *Label5;
 TLabel *Label6;
 int prt1, prt2, prt3,prt4,prt5,prt6,prt7,prt8; //глобальные переменные
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
 Caption = "Управление устройством через LPT-порт"; // заголовок
  Memo1->Clear();               //очищается многострочное окно
 Label3->Caption = "";         //очищаются метки Label
 Label4->Caption = "";
 Label5->Caption = "";
 Label6->Caption = "";
 SpeedButton1->AllowAllUp = true; //показывает утопленную клавишу
 SpeedButton1->GroupIndex = 1;    //принадлежность к группе клавиш
 SpeedButton2->AllowAllUp = true;  // то же
 SpeedButton2->GroupIndex = 2;     // то же
 SpeedButton3->AllowAllUp = true;
 SpeedButton3->GroupIndex = 2;
 SpeedButton4->AllowAllUp = true;
 SpeedButton4->GroupIndex = 2;
 SpeedButton5->AllowAllUp = true;
 SpeedButton5->GroupIndex = 2;
 SpeedButton6->AllowAllUp = true;
 SpeedButton6->GroupIndex = 2;
 SpeedButton7->AllowAllUp = true;
 SpeedButton7->GroupIndex = 2;
 SpeedButton8->AllowAllUp = true;
 SpeedButton8->GroupIndex = 2;
 SpeedButton9->AllowAllUp = true;
 SpeedButton9->GroupIndex = 2;
}
//---------------------------------------------------------------------------
// Реакция на нажатие клавиши <Выход>, прекращается работа программы
void __fastcall TForm1::Button1Click(TObject *Sender)
{
 Close();
}
//---------------------------------------------------------------------------
// Инициализация порта
void __fastcall TForm1::SpeedButton1Click(TObject *Sender)
{
   if(SpeedButton1->Down)
```

```
        {asm mov dx, 0x379
         asm mov al, 10011000b
         asm out dx, al
         asm mov dx, 0x37A
         asm mov al, 00001100b
         asm out dx, al
         asm mov dx, 0x378
         asm mov al, 255    //1111'1111b - во всех битах - 1
         asm out dx, al
         }
       Sleep(100);
      SpeedButton1->Font->Color = clBlue;
      SpeedButton1->Caption = "LPT1-порт инициализирован";
      Memo1->Lines->Add("Порт LPT1 инициализирован");
}
//---------------------------------------------------------------------------
// Обнуляем БИТ0
void __fastcall TForm1::SpeedButton2Click(TObject *Sender)
{
 if(SpeedButton2->Down)
      {
         asm mov dx, 0x378
         asm mov al, 254  // 1111'1110b
         asm out dx, al
   Memo1->Lines->Add("Вывод 2 (D1) установлен"); //это - сигнал
      }
   else
      {
      asm mov dx, 0x378
      asm mov al, 255  //возвращаем 1 в БИТ0
      asm out dx, al
   Memo1->Lines->Add("(Вывод 2 (D1) очищен"); // т.е. возвращена 1
   Memo1->Lines->Add("");
     }
 }
//---------------------------------------------------------------------------
// Обнуляем БИТ1
void __fastcall TForm1::SpeedButton3Click(TObject *Sender)
{
  if(SpeedButton3->Down)
    {
        asm mov dx, 0x378
         asm mov al, 253  //1111'1101b
         asm out dx, al
   Memo1->Lines->Add("Вывод 3 (D2) установлен ");
       }
   else
      {
      asm mov dx, 0x378
      asm mov al, 255 // 1111'1111b
```

```
      asm out dx, al
   Memo1->Lines->Add("Вывод 3 (D2)  очищен");
   Memo1->Lines->Add("");
     }
 }
//---------------------------------------------------------------------
// Обнуляем БИТ2
void __fastcall TForm1::SpeedButton4Click(TObject *Sender)
{
  if(SpeedButton4->Down)
      {
         asm mov dx, 0x378
         asm mov al, 251 //1111'1011b
         asm out dx, al
   Memo1->Lines->Add("Вывод 4 (D3) установлен ");
       }
   else
      {
      asm mov dx, 0x378    //+2
      asm mov al, 255  // FF или 1111'1111
      asm out dx, al
   Memo1->Lines->Add("Вывод 4 (D3) очищен");
   Memo1->Lines->Add("");
     }
 }
//---------------------------------------------------------------------

void __fastcall TForm1::SpeedButton5Click(TObject *Sender)
{
  if(SpeedButton5->Down)
       {
      asm mov dx, 0x378
      asm mov al, 223 //1101'1111 -
      asm out dx, al
   Memo1->Lines->Add("Вывод 6 (D5) установлен");
      }
        else
      {
      asm mov dx, 0x378
      asm mov al, 255
      asm out dx, al
   Memo1->Lines->Add("Вывод 6 (D5) очищен");
   Memo1->Lines->Add("");
     }
 }
//---------------------------------------------------------------------
```

В листинге 2.3 продолжается описание функций, которые являются реакциями на нажатие клавиш, устанавливающих сигналы NULL в битах 3…7. Практически все они имеют одинаковую структуру.

Следует помнить, что эта программа написана только для учебных целей, чтобы показать один из возможных вариантов нетрадиционного программирования LPT-порта.

По образцу этой программы вы можете сделать свою программу, предназначенную для управления каким-то аппаратом.

Листинг 2.3. Продолжение 1 файла *sport_a.cpp*

```
//---------------------------------------------------------------------------

void __fastcall TForm1::SpeedButton6Click(TObject *Sender)
{
 if(SpeedButton6->Down)
      {
         asm mov dx, 0x378
         asm mov al, 247 //1111'0111
         asm out dx, al
   Memo1->Lines->Add("Вывод 4 (D3) установлен ");
       }
   else
      {
      asm mov dx, 0x378
      asm mov al, 255
      asm out dx, al
   Memo1->Lines->Add("Вывод 4 (D3) очищен");
   Memo1->Lines->Add("");
     }
}
//---------------------------------------------------------------------------
void __fastcall TForm1::SpeedButton7Click(TObject *Sender)
{
 if(SpeedButton7->Down)
      {
         asm mov dx, 0x378
         asm mov al, 239 //1110'1111  - DF
         asm out dx, al
   Memo1->Lines->Add("Вывод 7 (D6) установлен ");
       }
   else
      {
      asm mov dx, 0x378
      asm mov al, 255
      asm out dx, al
   Memo1->Lines->Add("Вывод 7 (D6) очищен");
   Memo1->Lines->Add("");
     }
}
//---------------------------------------------------------------------------
void __fastcall TForm1::SpeedButton8Click(TObject *Sender)
{
```

```
  if(SpeedButton8->Down)
       {
       asm mov dx, 0x378
       asm mov al, 191 //1011'1111 -
       asm out dx, al
   Memo1->Lines->Add("Вывод 8 (D7) установлен");
        }
      else
       {
       asm mov dx, 0x378
       asm mov al, 255
       asm out dx, al
   Memo1->Lines->Add("Вывод 8 (D7) очищен");
   Memo1->Lines->Add("");
     }
}
//---------------------------------------------------------------------------
void __fastcall TForm1::SpeedButton9Click(TObject *Sender)
{
 if(SpeedButton9->Down)
       {
       asm mov dx, 0x378
       asm mov al, 127 //0111'1111 -
       asm out dx, al
   Memo1->Lines->Add("Вывод 9 (D8) установлен");
        }
      else
       {
       asm mov dx, 0x378
       asm mov al, 255
       asm out dx, al
   Memo1->Lines->Add("Вывод 9 (D8) очищен");
   Memo1->Lines->Add("");
     }
}
//---------------------------------------------------------------------------
```

В листинге 2.4 приведены исходные коды функции, предназначенной для проверки всех битов регистра 379h на наличие в них сигналов типа NULL. Результаты проверки выводятся в окне многострочного редактирования Memo1, а в случае появления сигнала – соответствующая запись появляется в нижней части рабочего окна программы.

Листинг 2.4. Продолжение 2 файла *sport_a.cpp*

```
//---------------------------------------------------------------------------
// Выполняется поиск битов со значением 0
void __fastcall TForm1::Button3Click(TObject *Sender)
{
Memo1->Lines->Add(" ");
Memo1->Lines->Add("Проверка ВСЕХ входных выводов LPT-порта");
// проверка вывода 15 порта
```

```
 asm {mov dx, 0x379
      in al, dx
      and al, 00001000b
      jz wyh14
      jnz wyh24
      }
 wyh14: prt4 = 0;
  asm jmp wyh34
 wyh24: prt4 = 1;
 wyh34:
    if(prt4 == 0){ Memo1->Lines->Add("Сигнал на выводе 15 установлен");
                   Label3->Caption = "Сигнал на выводе 15";
                   }
    else  {
            Memo1->Lines->Add("Сигнала на выводе 15 нет");
            Label3->Caption = "";
            }
// проверка вывода 13 порта
 asm {mov dx, 0x379
      in al, dx
      and al, 00010000b
      jz wyh15
      jnz wyh25
      }
 wyh15: prt5 = 0;
  asm jmp wyh35
 wyh25: prt5 = 1;
 wyh35:
    if(prt5 == 0){ Memo1->Lines->Add("Сигнал на выводе 13 установлен");
                   Label4->Caption = "Сигнал на выводе 13";
                   }
    else  {
            Memo1->Lines->Add("Сигнала на выводе 13 нет");
            Label4->Caption = "";
            }
// проверка вывода  порта
/*
 asm {mov dx, 0x379    //+1
      in al, dx
      and al, 00100000b
      jz wyh16
      jnz wyh26
      }
 wyh16: prt6 = 0;
  asm jmp wyh36
 wyh26: prt6 = 1;
 wyh36:
    if(prt6 == 0){ Memo1->Lines->Add("Сигнал 05 установлен");
                   Label6->Caption = "Сигнал на линии 05";
                   }
```

```
    else  {
            Memo1->Lines->Add("Сигнала 05 на входе нет");
            Label6->Caption = "";
            }
*/
// проверка вывода 10 порта
 asm {mov dx, 0x379    //+1
      in al, dx
      and al, 01000000b
      jz wyh17
      jnz wyh27
      }
 wyh17: prt7 = 0;
  asm jmp wyh37
 wyh27: prt7 = 1;
 wyh37:
    if(prt7 == 0){ Memo1->Lines->Add("Сигнал на выводе 10 установлен");
                   Label5->Caption = "Сигнал на выводе 10";
                   }
    else  {
            Memo1->Lines->Add("Сигнала на выводе 10 нет");
            Label5->Caption = "";
            }
// проверка вывода 11 порта
 asm {mov dx, 0x379    //+1
      in al, dx
      and al, 10000000b
      jz wyh18
      jnz wyh28
      }
 wyh18: prt8 = 0;
  asm jmp wyh38
 wyh28: prt8 = 1;
 wyh38:
    if(prt8 == 1){ Memo1->Lines->Add("Сигнал на выводе 11 установлен");
                   Label6->Caption = "Сигнал на выводе 11";
                   }
    else  {
            Memo1->Lines->Add("Сигнала на выводе 11 нет");
            Label6->Caption = "";
            }
}
//---------------------------------------------------------------------------
// обнулить D1...D8
void __fastcall TForm1::Button2Click(TObject *Sender)
{
      asm mov dx, 0x378
      asm mov al, 0        //255
      asm out dx, al
   Memo1->Lines->Add("");
```

```
   Memo1->Lines->Add("Все выводы (10, 11, 13 и 15) очищены");
}
//---------------------------------------------------------------------------
```

Проверка всех битов на наличие в них сигнала закончена. Вместо установки в битах сигналов программным путем можно применить способ электрический, т.е. вывод заданного бита подсоединять на землю (экран) через резистор 10…75 Ом. В реальных конструкциях к выводу заданного бита следует подключать инвертор с ТТЛ уровнями напряжений. При этом входные сигналы следует подавать через этот инвертор.

Проверяем работоспособность LPT-порта

Для проверки работоспособности LPT-порта при нестандартном его программировании, мною предлагается простейшее устройство, схема которого показана на рис. 2.3.

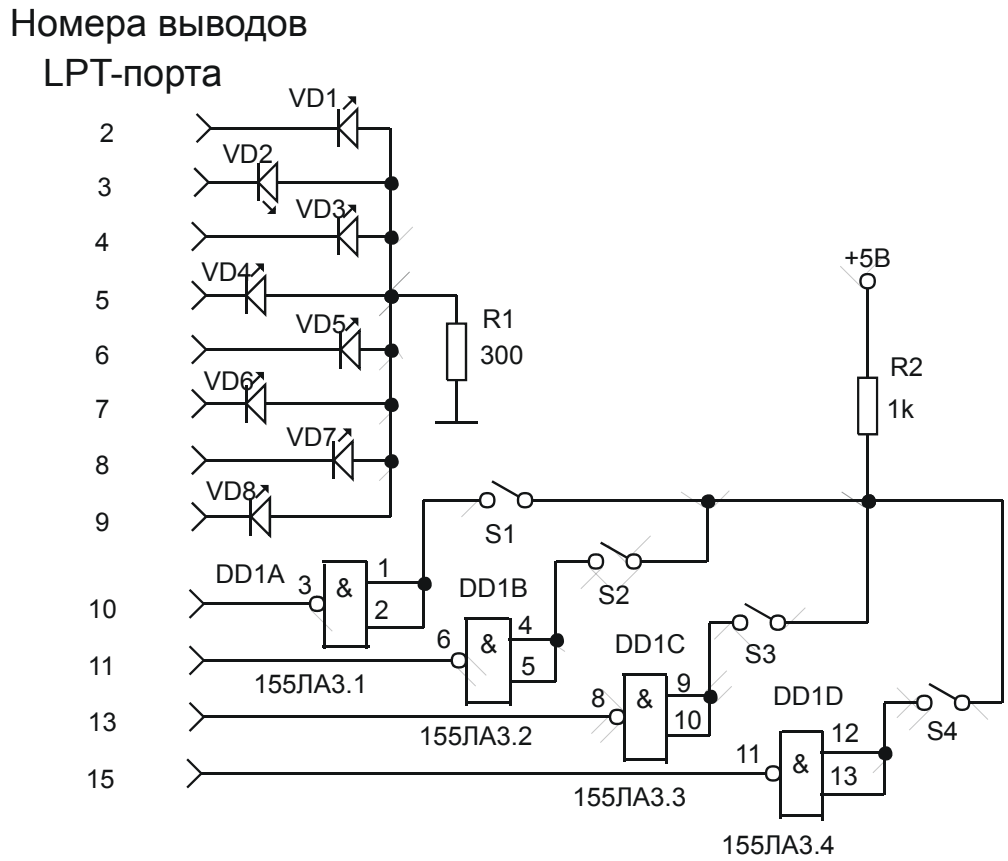

Рис. 2.3. Схема устройства для проверки LPR-порта

Устройство подключается к выводам 2…9, 10, 11,13 и 15 LPT-порта. При этом с выводов 2…9 выдаются сигналы непосредственно на светодиоды VD1…VD8, а на выводы 10, 11, 13 и 15 посредством ключей S1…S4 подаются сигналы с внешнего устройства на компьютер через инверторы DD1A…DD1D.

Предлагаемое устройство создано только для учебных целей и практического применения не имеет.

Вперед =>