PascalABC.NET

Работа с USB-реле

Ограничения

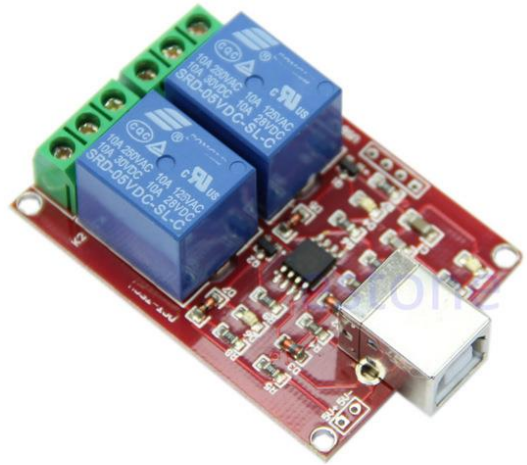
Модуль предназначен для работы с USB-реле в системе программирования PascalABC.NET.

Основан на библиотеке для .NET – проект USB-Relay, который в свою очередь основан на проекте usb-relay-hid. Вся лицензионная информация, подробная справка и данные по поддерживаемым устройствам могут быть найдены в соответствующих проектах.

Данный модуль предназначен для работы только с определённым видом устройств – эти ограничения связаны с явно реализованным протоколом управления.

Сами по себе устройства не требуют драйверов для подключения, и определяются системой как HID-устройства. В процессе работы модуль обнаруживает только поддерживаемые реле.

Предупреждение!



USB-реле – устройство для управления электрическими цепями, поэтому будьте осторожны и тщательно изучите технические характеристики, а также возможные схемы включения реле в электрические схемы!

Разработчики ПО не несут ответственности за любой ущерб здоровью или собственности, нанесённый неверным или непредусмотренным использованием таких устройств.

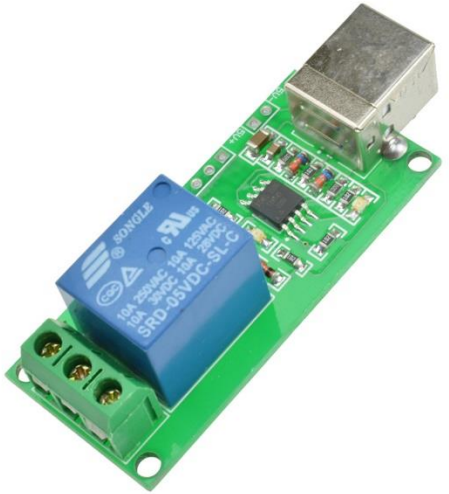


Схема подключения

Реле-устройство имеет, в зависимости от варианта исполнения, один, два или восемь портов (каналов).

Каждый порт имеет три разъёма, маркированных следующим образом:

- COM – входной разъём;
- NC – NORMALLY CLOSED, «закрытый по умолчанию»;
- NO – NORMALLY OPENED, «открытый по умолчанию».

По умолчанию контакты COM и NC замкнуты, контакты COM и NO разомкнуты.

При «открытии» порта контакты COM и NC размыкаются, контакты COM и NO замыкаются.

Рекомендуется использовать два контакта – NO и COM, при этом по умолчанию они разомкнуты, порт (канал) «закрыт». При открытии канала реле эти контакты замыкаются, на реле загорается дополнительный светодиод.

Соглашения

В данной библиотеке используются следующие соглашения:

1. Если к компьютеру подключено несколько устройств реле, то они нумеруются начиная с 0, работа с каждым конкретным реле может вестись с указанием его номера (индекса).
2. Каждый порт на устройствах реле нумеруется начиная с 1, работа с портами может вестись с указанием номера порта.

Например, если подключено только одно устройство, имеющее только один порт, то индекс устройства – 0, индекс порта – 1.

Если устройство не опознано, или не может быть использовано из-за несовместимости с данной библиотекой, то функция количества устройств в системе будет возвращать 0.

Использование библиотеки

Для работы с библиотекой необходимо, чтобы программе были доступны следующие файлы:

1. «USB_RELAY_DEVICE.dll» и «USBRelay.dll» – библиотеки. Исходный код можно найти в проектах USB-Relay и usb-relay-hid.
2. Модуль работы с библиотекой на PascalABC.NET – файл «RelayABC.pas».

Самый простой способ – сохранить три этих файла в папку с программой.

В программе на Паскале достаточно подключить соответствующий модуль:

```
uses SpeechABC;
```

Простейшие действия

Базовые функции могут быть использованы без инициализации библиотеки, работать с реле можно с помощью трёх функций:

```
function OpenChannel(deviceIndex, channelIndex : integer) : boolean;
function CloseChannel(deviceIndex, channelIndex : integer) : boolean;
function InvertChannel(deviceIndex, channelIndex : integer) : boolean;
```

Они служат для открытия порта (канала) реле, закрытия, или инвертирования соответственно. В качестве параметров используется номер (индекс) устройства, с нумерацией от 0, а также номер порта (канала) на этом устройстве (нумерация от 1).

То есть если подключено только одно устройство с одним портом, то для открытия используется команда:

```
OpenChannel(0,1);
```

Управление списком реле

Большинство функций доступно через объект RelayManager (явно создавать его не нужно!).

Для работы необходимо выполнить метод RelayManager.InitRelays(), который строит список устройств, подключенных к компьютеру (только те реле, с которыми может работать библиотека). Проверить, была ли инициализация успешной, можно с помощью функции RelayManager.Inited() логического типа, количество обнаруженных устройств возвращает функция RelayManager.DeviceCount().

В каждый момент времени одно из устройств списка является активным, выбрать активное устройство можно функцией RelayManager.OpenDevice(deviceIndex : integer).

При выборе нового устройства старое «закрывается».

Действия с активным устройством

После инициализации списка устройств и выбора активного устройства с ним доступны следующие действия:

- RelayManager.ChannelsCount – количество портов активного устройства, integer;
- RelayManager.RelaySerial – строка-идентификатор устройства, string;
- RelayManager.CurrentRelayIndex – индекс активного устройства в списке, integer;
- RelayManager.OpenChannel(channelIndex : integer) – открыть на активном устройстве канал с указанным номером, индексация с 1, boolean;
- RelayManager.CloseChannel(channelIndex : integer) – закрыть на активном устройстве канал с указанным номером, индексация с 1, boolean;
- RelayManager.OpenAllChannels – открыть все каналы (порты) активного устройства, boolean;
- RelayManager.CloseAllChannels – закрыть все каналы активного устройства, boolean;
- RelayManager.ChannelOpened(channelIndex : integer) – открыт ли канал с указанным индексом на активном устройстве, boolean;

Пример работы

```
uses RelayABC;
begin
  //  Инициализируем библиотеку
  RelayManager.InitRelays;
  //  Количество обнаруженных устройств
  WriteLn('Всего устройств обнаружено : ', RelayManager.RelaysCount);
  WriteLn('Открываем первое устройство...');
  //  Пытаемся открыть первое по порядку устройство
  if not RelayManager.OpenRelay(0) then
    begin
      WriteLn('Не могу открыть устройство для работы, завершаюсь');
      exit;
    end;
  //  Номер текущего устройства
  WriteLn('Номер текущего устройства       : ', RelayManager.CurrentRelayIndex);
  //  Строка-идентификатор текущего устройства
  WriteLn('Строка-идентификатор устройства : ', RelayManager.RelaySerial);
  //  Количество портов текущего устройства
  WriteLn('Количество каналов устройства   : ', RelayManager.ChannelsCount);
```

Работа с активным реле

```
//  Ждём указаний от пользователя
  WriteLn('1 - открыть порт (канал) 1, 2 - закрыть порт 1, q - выход :');
  while true do
    case ReadlnChar of
        '1' : begin
                WriteLn('Открываем порт ...');
                RelayManager.OpenChannel(1);
                WriteLn('Состояние порта 1 (открыт?) : ', RelayManager.ChannelOpened(1));
              end;
        '2' : begin
                WriteLn('Закрываем порт ...');
                RelayManager.CloseChannel(1);
                WriteLn('Состояние порта 1 (открыт?) : ', RelayManager.ChannelOpened(1));
              end;
        'q' : break;
      end;
//  Финализируем работу библиотеки
  RelayManager.CloseRelays;
```