USB 16 Relay Board
RTU ModBus controlled

User Manual
Date: 21 Aug 2017

Content

1. Specification

- 16 x SPDT relays, selectable during purchase -10A / 250VAC, 15A / 120VAC, 10A / 28VDC;
- USB connector - type B
- Power supply requirements, selectable during purchase:
  - 12V DC / 600mA
  - 24V DC / 400mA
- Converter chipset: FT232RL
- FT232RL optical isolation against communication noises
- Communication: serial USB communication (Virtual Com Port)
- Led-s: Relay Led, Power ON Led, USB ON Led, Rx Led, Tx Led
- Operating temperature range: from 0 ºC to +80 ºC
- PCB parameters : FR4 / 1.5mm / two layers / metalized holes / HAL / white stamp / solder mask / Extra PCB openings for better voltage isolation / Doubled PCB tracks for better voltage isolation
- Dimensions: W=82mm x L=203mm x H=24mm
- SKU:
  - **DAE-RO16-12V-MODBUS-USB-PCB** for option PCB, Power Supply 12V
  - **DAE-RO16-24V-MODBUS-USB-PCB** for option PCB, Power Supply 24V
  - **DAE-RO16-12V-MODBUS-USB-BOX** for option Din Rail Box, Power Supply 12V
  - **DAE-RO16-24V-MODBUS-USB-BOX** for option Din Rail Box, Power Supply 24V

2. Applications examples

- Home automation
- Industrial automation
- Control electrical devices

3. Relay module overview

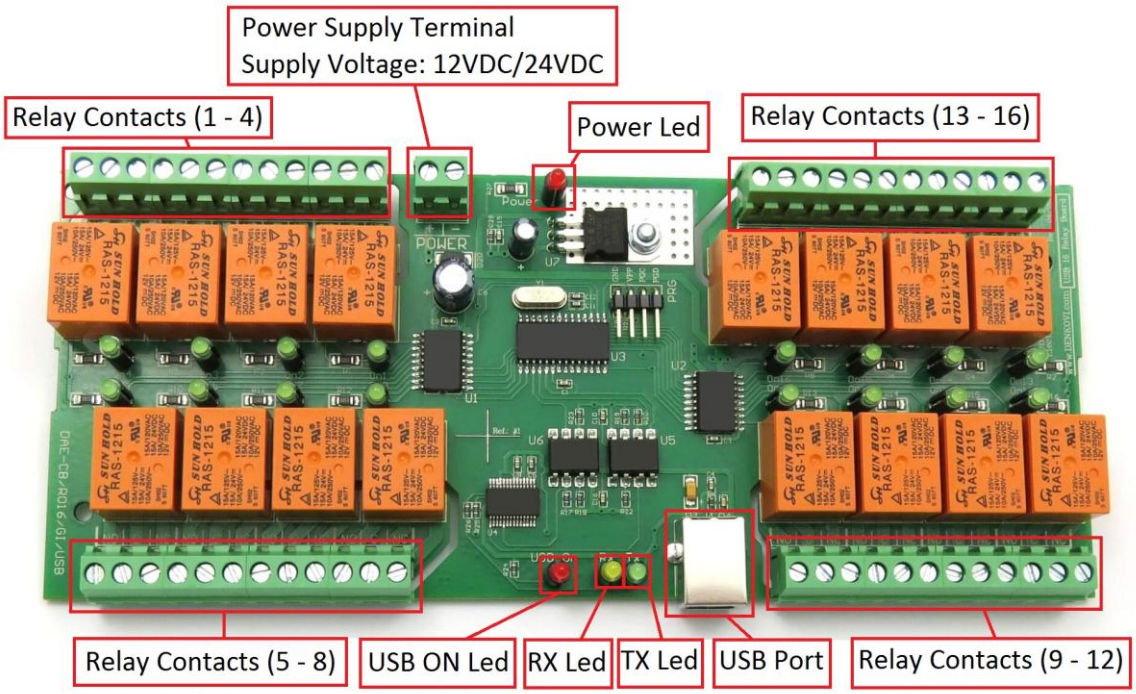


Figure 1. PCB Device Overview

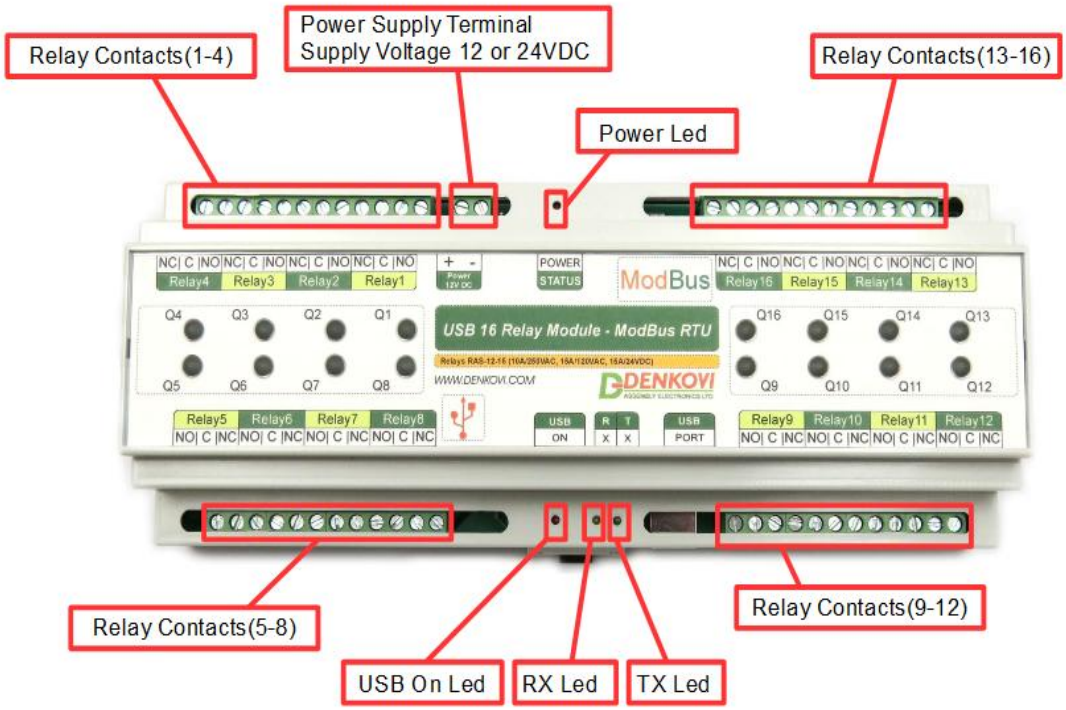


Figure 2. DIN Rail Device overview

4. USB port

The type of this USB port is shown on the image bellow. You can find suitable USB cables in our stock as well - http://denkovi.com/usb-cables

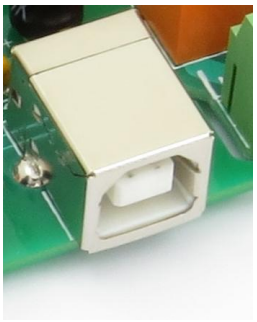


Figure 3. USB Port

5. Power connector

The power supply connector is shown on the image bellow. The supply voltage is **12VDC** or **24VDC** depending on the type selected during purchase:

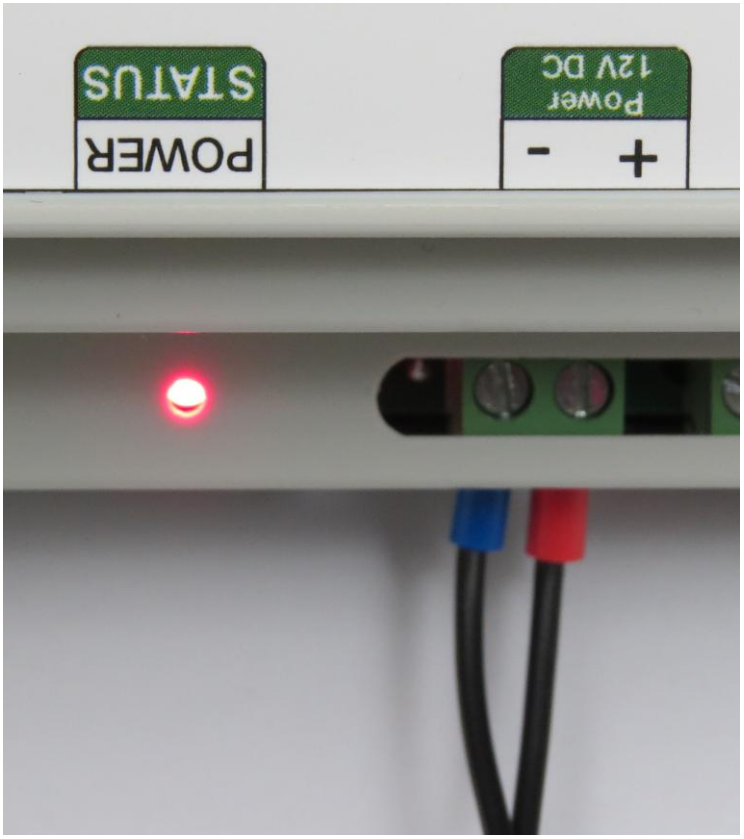


Figure 4. Relay Contacts

6. Drivers Installation

Be sure you have FTDI VCP drivers installed. In order to use the device, you have to install them first. If you don't have them installed, please go to FTDI official web page - http://www.ftdichip.com/Drivers/VCP.htm and download and install them.

Connect the USB relay board to your computer and see if it appears as virtual com port. If everything is ok with the drivers, you should see the some of the images bellow:

6.1. Windows (picture is from 7 x64)

Figure 5. Windows 7 successful installed driver

6.2. Linux (picture is from Ubuntu x64)

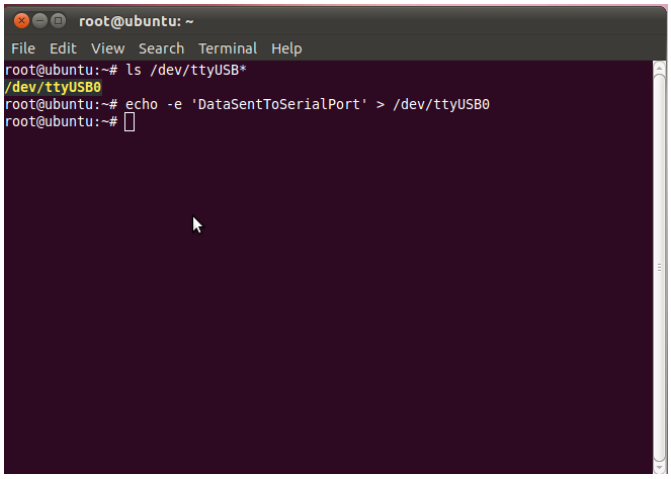


Figure 6. Linux successful installed drivers

For installation guides:
http://www.ftdichip.com/Support/Documents/InstallGuides.htm

7. Protocol Description

The **USB 16 Relay RTU ModBus** supports standart **RTU Modbus** protocol for setting/getting the relays status. The communication may be used over COM Port Communication. All relays are represented as coils so there are 16 coils supported by the device (Coil 0 to Coil 15). There are 22 holding registers for additional settings (Holding Register 0 to Holding Register 21). RTU Modbus Protocol frame is illustrated on **Figure 7**:

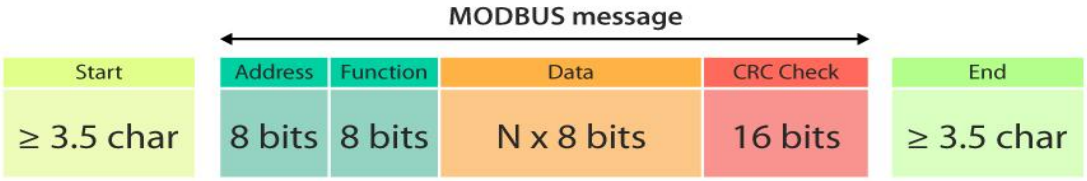


Figure 7. Linux successful installed drivers

- **Start** is at least 3.5 char time of silence between master and slave;
- **Address** is the Slave ID address;
- **Function** is the function code of Modbus protocol (several function codes will be described);
- **Data** is the appropriate data for the used function code;
- **CRC** is Cyclic Redundancy Check. Used CRC is CRC-16-ANSI (CRC-16-IBM). Polynomial of the CRC is $x^{16} + x^5 + x^2 + 1$.

Modbus frame always starts and ends with time of silence at least 3.5 characters. Address is 8 bit long and function, too. Data length could vary depend on function code. CRC is 16 bit and is transmitted least significant byte first.

Slave ID by default is 1, but can be changed.

There must be delay of 20ms minimum between every two commands.

7.1. Serial Port parameters

Table 1. Serial Port Parameters

Baud rate	9600 bps
Stop bit	1
Data bits	8
Parity	None

7.2. Coils Commands

Coils are mapped to Relays. Reading and Writing Coils reads/writes relays status. There are 16 coils for reading and writing – Coil0 to Coil15 which refer to Relay1 to Relay16.

7.2.1. Read Coils (0x01)

Read Coils (0x01) command will return current relays status. Example for reading all relays status:

Table 2. Read Coils Command Format

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x01	0x00	0x00	0x00	0x10	0x3D	0xC6

Table 3. Read Coils Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
Address	Function	Data			CRC	
0x01	0x01	0x02	0x00	0x55	0x79	0xC3

- Byte3 represents the bytes following this byte;
- Byte4 represents state of Relay 15 to Relay 8;
- Byte5 represents state of Relay 7 to Relay 0;

This example illustrate the following relays status – Relay9, Relay11, Relay13 and Relay 15 are ON. All other relays are OFF.

Please note that all bytes are in hexadecimal format!

7.2.2. Write Single Coil (0x05)

Write Single Coil (0x05) command will set relay ON or OFF. Example of how to set Relay11 ON is given in table below:

Table 4. Write Single Coil Command Format

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x05	0x00	0x0A	0xFF	0x00	0xAC	0x38

- Byte3 and Byte4 represents the data address of the coil to set ON/OFF (data address is 10 in our example which is Relay11);
- Byte5 and Byte6 represents the state to set (in our example we set relay ON);

Note: If you want to set relay OFF, Byte11 and Byte12 should be 00.

Table 5. Write Single Coil Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x05	0x00	0x0A	0xFF	0x00	0xAC	0x38

- Byte3 and Byte4 represents the data address of the coil that was set;
- Byte5 and Byte6 represents the state that was set;

7.2.3. Write Multiple Coils (0x0F)

Write Multiple Coils (0x0F) command will set multiple sequential relays ON or OFF. Example of how to set Relay1 to Relay10 with single command is given in table below:

Table 6. Write Multiple Coils Command Format

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9	Byte10	Byte11
Address	Function	Data							CRC	
0x01	0x0F	0x00	0x00	0x00	0x0A	0x02	0x55	0x01	0x1B	0xA8

- Byte3 and Byte4 represents the data address of the first coil to set ON/OFF (data address is 0 in our example which is Relay1);
- Byte5 and Byte6 represents the number of coils to set (in our example 10);
- Byte7 represents how many bytes follow this byte (in our example is 2);
- Byte8 represents coils 7 – 0 (0x55 = 0101 0101);
- Byte9 represents 6 leading zeroes and coils 9 – 8 (0x01 = 0000 0001);

This example will set Relay1, Relay3, Relay5, Relay7 and Relay 9.

Table 7. Write Multiple Coils Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x0F	0x00	0x00	0x00	0x0A	0xD5	0xCC

- Byte3 and Byte4 represents the data address of the first coil that was set;
- Byte5 and Byte6 represents the number of coils that were written;

7.3. Holding Registers Commands

There are 22 readable and 21 writable Holding Registers. The table below describes each Holding Register's function. Each Holding Register is 2 byte long. Each given numbers in the table below are in decimal unsigned type.

Table 8. Supported holding registers

Holding Register's Address	Description
0	Relay 1 Auto Off Time Units (0 ÷ 65535)
1	Relay 2 Auto Off Time Units (0 ÷ 65535)

2	Relay 3 Auto Off Time Units (0 ÷ 65535)
3	Relay 4 Auto Off Time Units (0 ÷ 65535)
4	Relay 5 Auto Off Time Units (0 ÷ 65535)
5	Relay 6 Auto Off Time Units (0 ÷ 65535)
6	Relay 7 Auto Off Time Units (0 ÷ 65535)
7	Relay 8 Auto Off Time Units (0 ÷ 65535)
8	Relay 9 Auto Off Time Units (0 ÷ 65535)
9	Relay 10 Auto Off Time Units (0 ÷ 65535)
10	Relay 11 Auto Off Time Units (0 ÷ 65535)
11	Relay 12 Auto Off Time Units (0 ÷ 65535)
12	Relay 13 Auto Off Time Units (0 ÷ 65535)
13	Relay 14 Auto Off Time Units (0 ÷ 65535)
14	Relay 15 Auto Off Time Units (0 ÷ 65535)
15	Relay 16 Auto Off Time Units (0 ÷ 65535)
16	Relays Mode Register (0 is Manual, 1 is Auto Off)*
17	Timer Units Register (0 = 100ms/unit, 1 = 1s/unit)*
18	Slave ID Register (1 ÷ 247)**
19	Relays State Register*
20	Relays Startup Mode Register (0 will turn off all relays, 1 will set relays in states written to Relays State Register, 2 will set relays in state from last change)
21	Version Register*** – Value of this register divided by 100 gives the version of the module

**Bitwise settable register – MSB is Relay16 and LSB is Relay1.*
***Slave ID out of this range (1 ÷ 247) could cause unexpected behavior. Change it carefully.*
****Read Only Register.*

7.3.1. Write Single Holding Register (0x06)

Write Single Holding Register (0x06) command will set single Holding Register. Example of how to set Relay1 Auto Off Time Units with single command is given in table below:

Table 9. Write Single Holding Register Command Format

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x06	0x00	0x00	0x00	0x64	0x88	0x21

- Byte3 and Byte4 represents the data address of the Holding Register (data address is 0 in our example which is Relay 1 Auto Off Time Units' Register);
- Byte5 and Byte6 represents the value of the Holding Register (value is 100 units in our example);

Table 10. Write Single Holding Register Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x06	0x00	0x00	0x00	0x64	0x88	0x21

- Byte3 and Byte4 represents the data address of the Holding Register that was set;
- Byte5 and Byte6 represents the value of the Holding Register that was set;

7.3.2. Read Multiple Holding Registers (0x03)

Read Multiple Holding Registers (0x03) command will read multiple sequential Holding Registers. Example of how to read 22 Holding Registers starting from data address 0 is given in table below:

Table 11. Read Multiple Holding Registers Command Format

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x03	0x00	0x00	0x00	0x16	0xC4	0x04

- Byte3 and Byte4 represents the data address of the First Holding Register to read(data address is 0 in our example which is Relay 1 Auto Off Time Units' Register);
- Byte5 and Byte6 represents the total number of Holding Registers to read (value is 22 in our example which means all registers should be read);

Table 12. Read Multiple Holding Registers Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data					
0x01	0x03	0x2C	0x00	0x64	0x00	0x01	0x00
Byte9	Byte10	Byte11	Byte12	Byte13	Byte14	Byte15	Byte16
Data							
0x01	0x00	0x01	0x00	0x01	0x00	0x01	0x00
Byte17	Byte18	Byte19	Byte20	Byte21	Byte22	Byte23	Byte24
Data							
0x01	0x00	0x01	0x00	0x01	0x00	0x01	0x00
Byte25	Byte26	Byte27	Byte28	Byte29	Byte30	Byte31	Byte32
Data							
0x01	0x00	0x01	0x00	0x01	0x00	0x01	0x00
Byte33	Byte34	Byte35	Byte36	Byte37	Byte38	Byte39	Byte40
Data							
0x01	0x00	0x01	0x00	0x53	0x80	0x21	0x00
Byte41	Byte42	Byte43	Byte44	Byte45	Byte46	Byte47	

Data						
0x01	0x03	0x58	0x00	0x00	0x00	0x64
Byte48	Byte49					
CRC						
0xB1	0xCF					

- Byte3 represents the bytes following this byte (bytes following this bytes are 44 in our example – 22 Holding Registers x 2 bytes each);
- Byte4 and Byte5 represents Relay 1 Auto Off Time Units(100 units in our example);
- Byte6 and Byte7 represents Relay 2 Auto Off Time Units (1 units in our example);
- Byte8 and Byte9 represents Relay 3 Auto Off Time Units (1 units in our example);
- Byte10 and Byte11 represents Relay 4 Auto Off Time Units (1 units in our example);
- Byte12 and Byte13 represents Relay 5 Auto Off Time Units (1 units in our example);
- Byte14 and Byte15 represents Relay 6 Auto Off Time Units (1 units in our example);
- Byte16 and Byte17 represents Relay 7 Auto Off Time Units (1 units in our example);
- Byte18 and Byte19 represents Relay 8 Auto Off Time Units (1 units in our example);
- Byte20 and Byte21 represents Relay 9 Auto Off Time Units (1 units in our example);
- Byte22 and Byte23 represents Relay 10 Auto Off Time Units (1 units in our example);
- Byte24 and Byte25 represents Relay 11 Auto Off Time Units (1 units in our example);
- Byte26 and Byte27 represents Relay 12 Auto Off Time Units (1 units in our example);
- Byte28 and Byte29 represents Relay 13 Auto Off Time Units (1 units in our example);
- Byte30 and Byte31 represents Relay 14 Auto Off Time Units (1 units in our example);
- Byte32 and Byte33 represents Relay 15 Auto Off Time Units (1 units in our example);
- Byte34 and Byte35 represents Relay 16 Auto Off Time Units (1 units in our example);
- Byte36 and Byte37 represents Relays Mode Register (0x0053 in our example which is 0000 0000 0101 0011 – Relay1, Relay2, Relay5 and Relay7 are in Auto Off Mode and all others are in Manual Mode);

- Byte38 and Byte39 represents Timer Units Register (0x8021 in our example which is 1000 0000 0010 0001 – Relay1, Relay6 and Relay 16 units are 1s/unit and all others are 100ms/unit);
- Byte40 and Byte41 represents Slave ID Register (Slave ID is 1 in our example);
- Byte42 and Byte43 represents Relays State Register (0x0358 in our example which is 0000 0011 0101 1000 – Relay4, Relay5, Relay7, Relay9 and Relay10 will be turned on all other will be turned off (if Relays Startup Mode Register is 1 or 2));
- Byte44 and Byte45 represents Relays Startup Mode Register (All relays will be turned off on startup in our example);
  Byte46 and Byte47 represents Version Register (100 in our example which means v1.00);

7.3.3. Write Multiple Holding Registers (0x10)

Write Multiple Holding Registers (0x10) command will write multiple sequential Holding Registers. Example of how to write 5 Holding Registers starting from data address 1 is given in table below:

Table 13. Write Multiple Holding Registers Command Format

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data					
0x01	0x10	0x00	0x01	0x00	0x05	0x0A	0x00
Byte9	Byte10	Byte11	Byte12	Byte13	Byte14	Byte15	Byte16
Data							
0xC8	0x01	0x2C	0x01	0x90	0x01	0xF4	0x02
Byte17	Byte18	Byte19					
Data	CRC						
0x58	0x9B	0xAA					

- Byte3 and Byte4 represents the data address of the first Holding Register (0x0001 in our example);
- Byte5 and Byte6 represents the total number of Holding Registers to be written (5 in our example);
- Byte7 represents the bytes following this byte (0x0A in our example – 5 Holding Registers x 2 bytes each);
- Byte8 and Byte9 represents the value to be written on the first data address (value is 0x00C8 in our example and will be written to data address 0x01 which is Relay 2 Auto Off Time Units Register);
- Byte10 and Byte11 represents the value to be written on the second data address (value is 0x012C in our example and will be written to data address 0x02 which is Relay 3 Auto Off Time Units Register);

- Byte12 and Byte13 represents the value to be written on the third data address (value is 0x0190 in our example and will be written to data address 0x03 which is Relay 4 Auto Off Time Units Register);
- Byte14 and Byte15 represents the value to be written on the fourth data address (value is 0x01F4 in our example and will be written to data address 0x04 which is Relay 5 Auto Off Time Units Register);
- Byte16 and Byte17 represents the value to be written on the fifth data address (value is 0x01F4 in our example and will be written to data address 0x05 which is Relay 6 Auto Off Time Units Register);

Table 14. Write Multiple Holding Registers Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x03	0x00	0x01	0x00	0x05	0x51	0xCA

- Byte3 and Byte4 represents the data address of the first Holding Register that was set;
- Byte5 and Byte6 represents the total number of Holding Registers that were written;

7.4. Report Slave ID (0x11)

Report Slave ID (0x11) command will return current Slave ID of **USB 16 Relay RTU ModBus**. Example of how to read Slave ID is given below.

This command (with unit ID 0xF8) can be always used in order to recover the slave ID in case it was forgotten.

Table 15. Report Slave ID Command Format

Byte1	Byte2	Byte3	Byte4
Address	Function	CRC	
0xF8	0x11	0xC0	0x2C

Table 16. Report Slave ID Expected Answer

Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Address	Function	Data				CRC	
0x01	0x11	0x03	0x01	0x00	0x64	0xAD	0xA6

- Byte3 represents the number of bytes following this byte (it will be always 3);
- Byte4 represents current Slave ID* (0x01 in our example);
- Byte5 and Byte6 represents the version of the board – version is formed when this value is divided by 100 (value is 100 dec (0x64) in our example which means it is v1.00);

8. Software

There is a wide variety of free and low cost software for communication through Serial COM Port on the Internet.

8.1. Modbus Poll

Modbus Poll gives you an idea of how to communicate with **USB 16 Relay RTU ModBus**. You can download and install Modbus Poll Software from this link:
http://www.modbustools.com/download.html

Run the software, a screen on the **Figure 8** should appear.

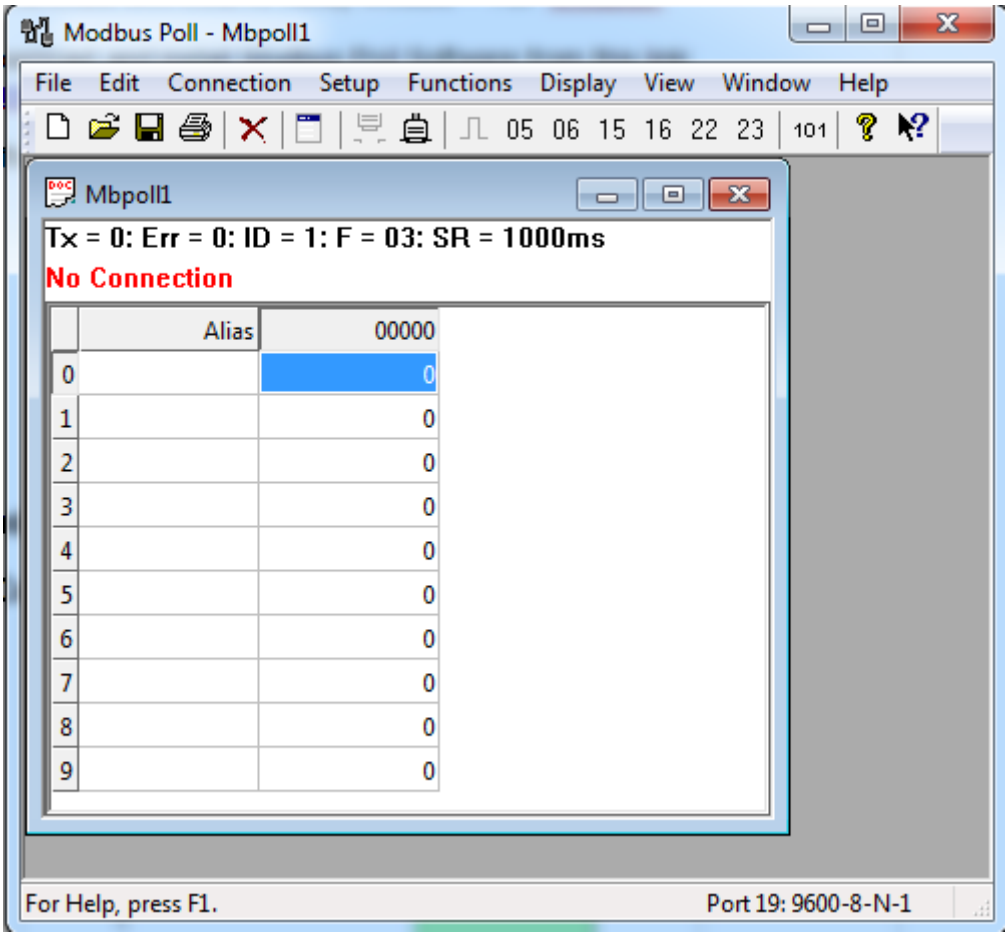


Figure 8. Modbus Poll Startup Screen

Go to Setup → Read/Write Definition and check the settings from **Figure 9**. Then press OK.

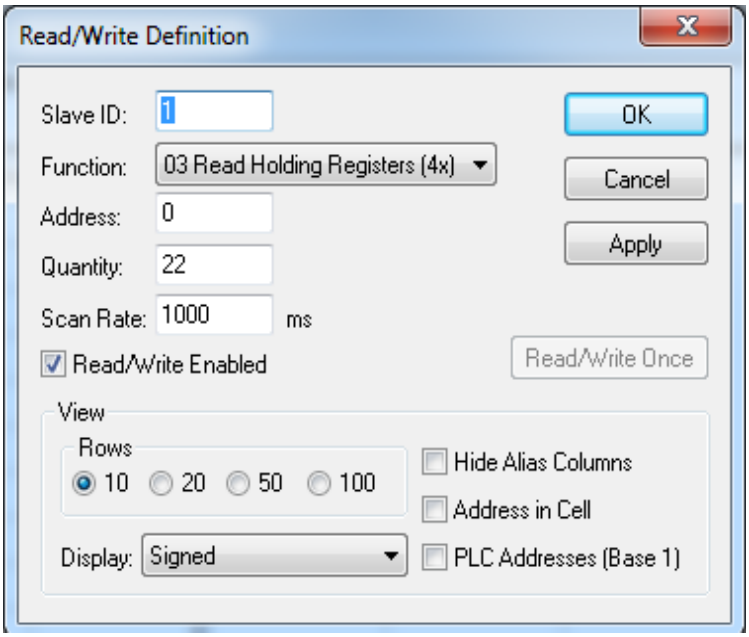


Figure 9. Modbus Poll Read/Write Definition

Next step is to create the connection. Go to Connection → Connect... **Figure 10** shows example of connection settings.

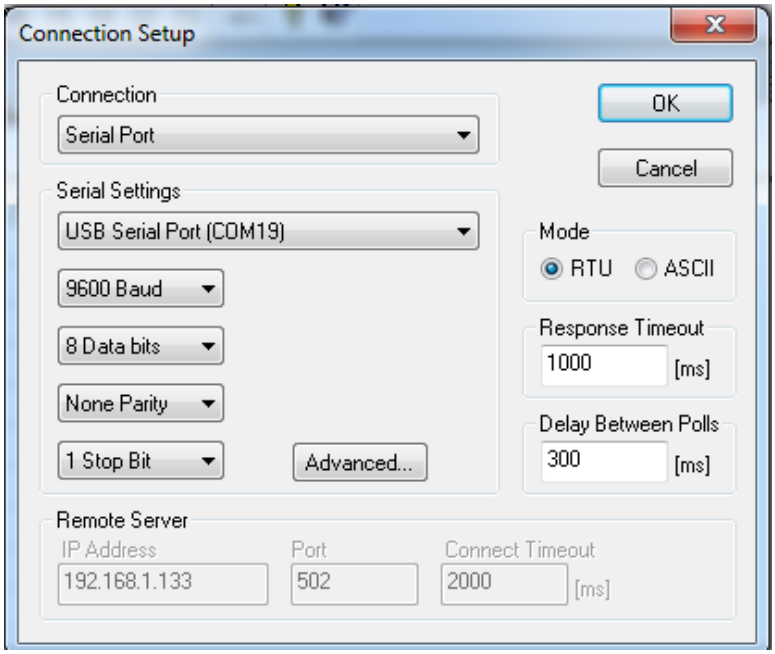


Figure 10. Modbus Poll Connection Setup

Please check the following settings:

- **Serial Port** must be the COM Port of the device;

Other parameters must be the same as in the picture (see

There must be delay of 20ms minimum between every two commands.

- Serial Port parameters);

When all is done TX counter will start incrementing itself if successful connection is made and you should view Holding Registers values (**Figure 11**).

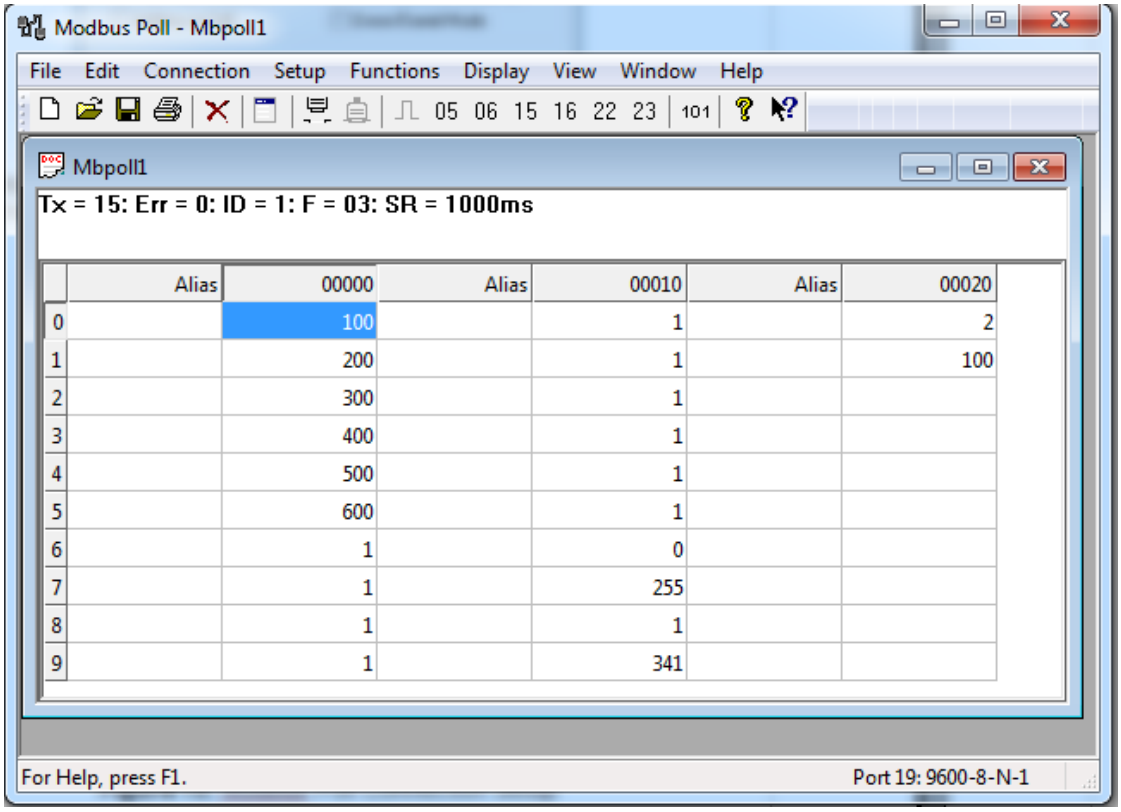


Figure 11. Modbus Poll successfully connected screen

8.2. Denkovi ModBus Tool

Denkovi ModBus Tool is our own software which can control **USB 16 Relay RTU ModBus**. It has user friendly interface and controlling is very simple. You can download it from HERE.

First select the proper COM Port of device and click Connect. Set the Slave ID (by default it is 1) and set timeout (recommended is 100). After successful connection the screen on **Figure 12** will appear.

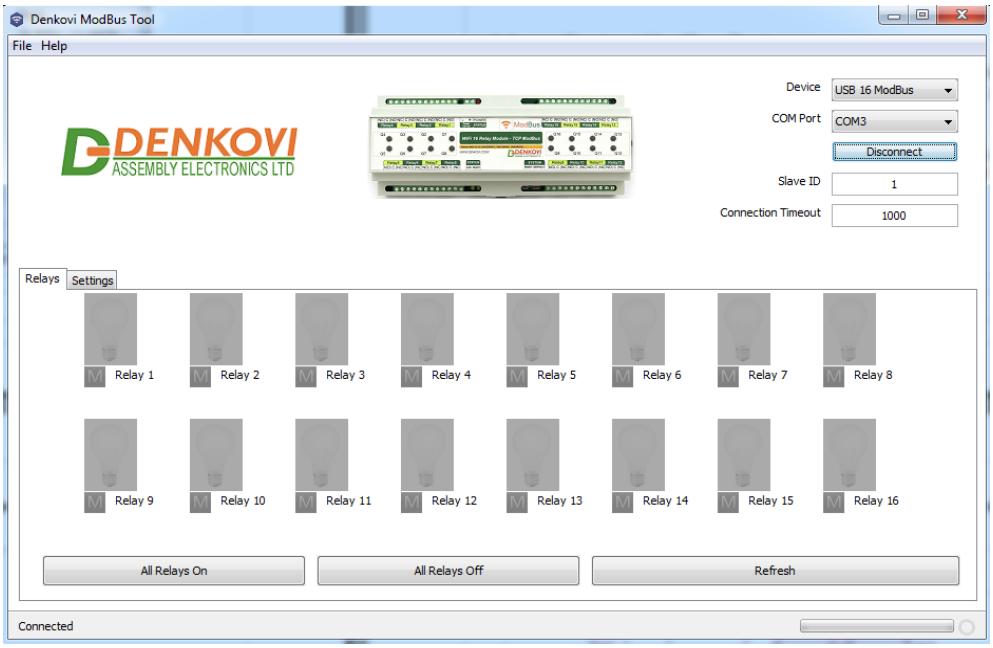


Figure 12. Denkovi ModBus Tool connected

Next click on "Refresh" button. Current states of relays will be loaded. Now you can switch on and off relays by clicking on images.

To be sure of current states of the relays every time before and after switching relays on and off click on "Refresh" button. Only this will provide you current state of relays. The software does not refresh itself!

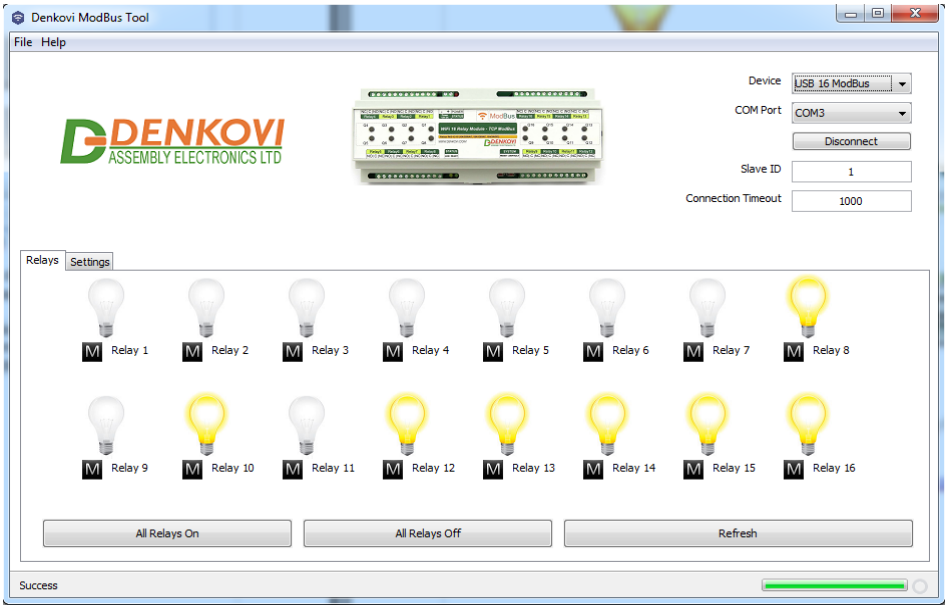


Figure 13. Getting states of relays

Another option is to get settings of the board. Just click on "Settings" tab and click "Refresh" button. Here you can see the current settings of the board.

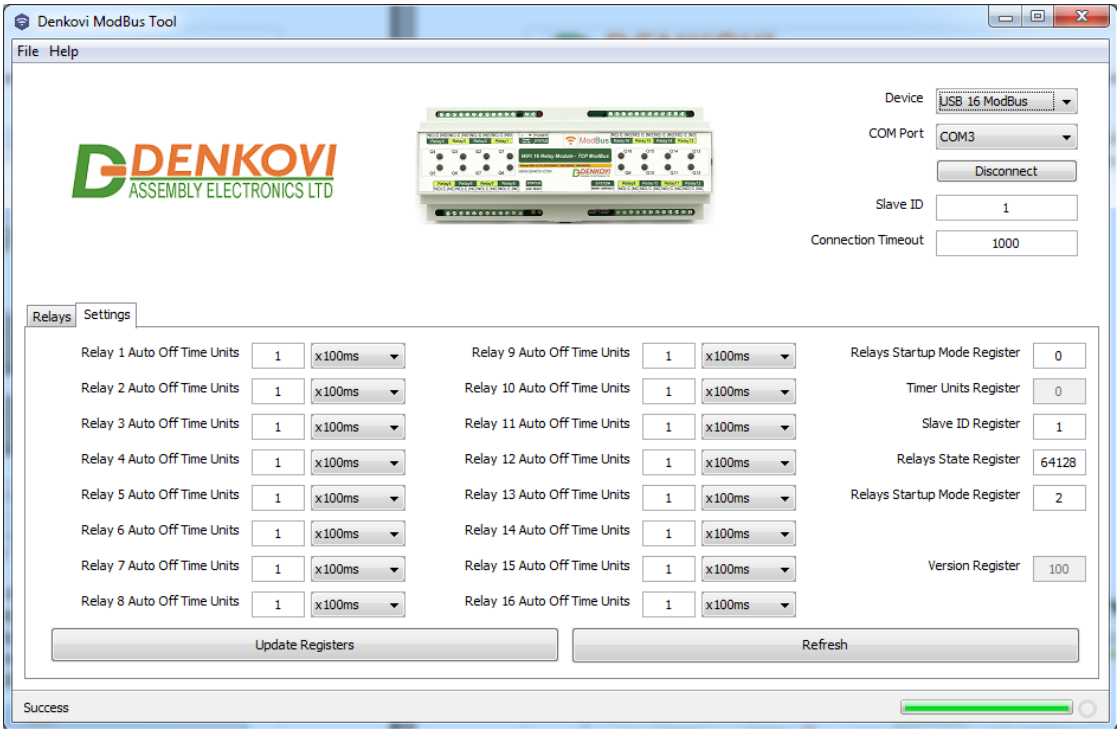


Figure 14. Getting current settings

More about settings is given in Holding Registers Commands section

To make any changes to settings of the board, fill in your preferred values and click "Update Registers" button.

9. Appendix 1. Connections

9.1. Connect lamp to relay

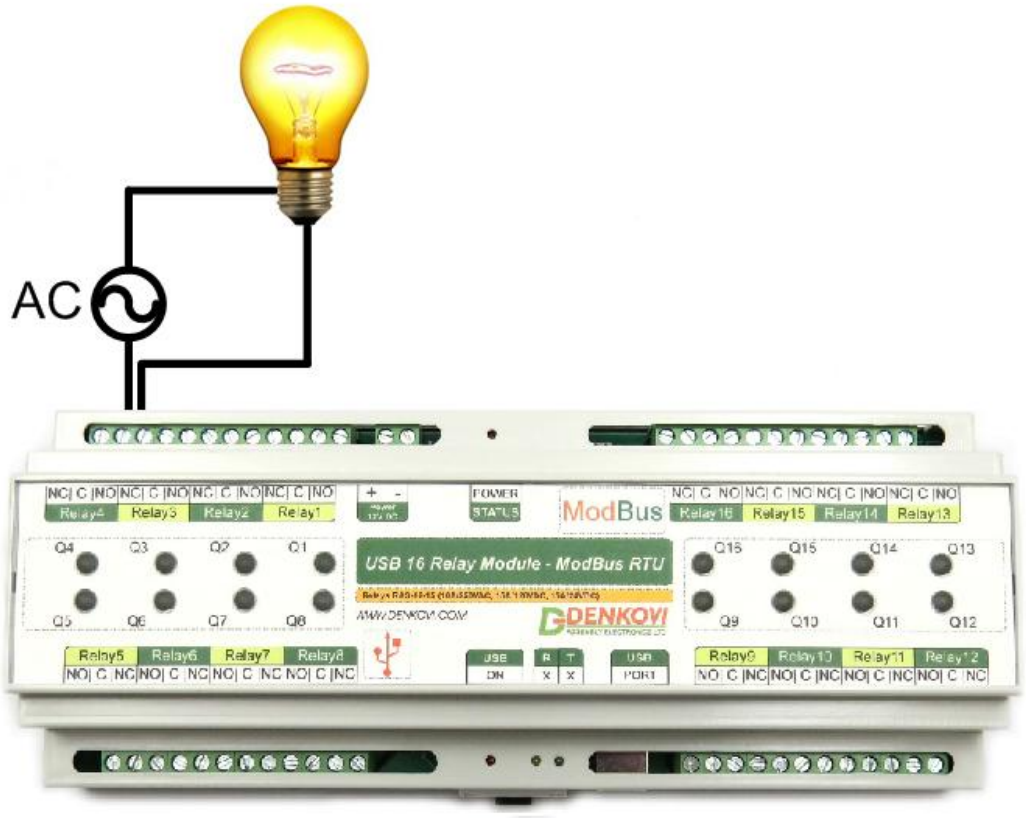
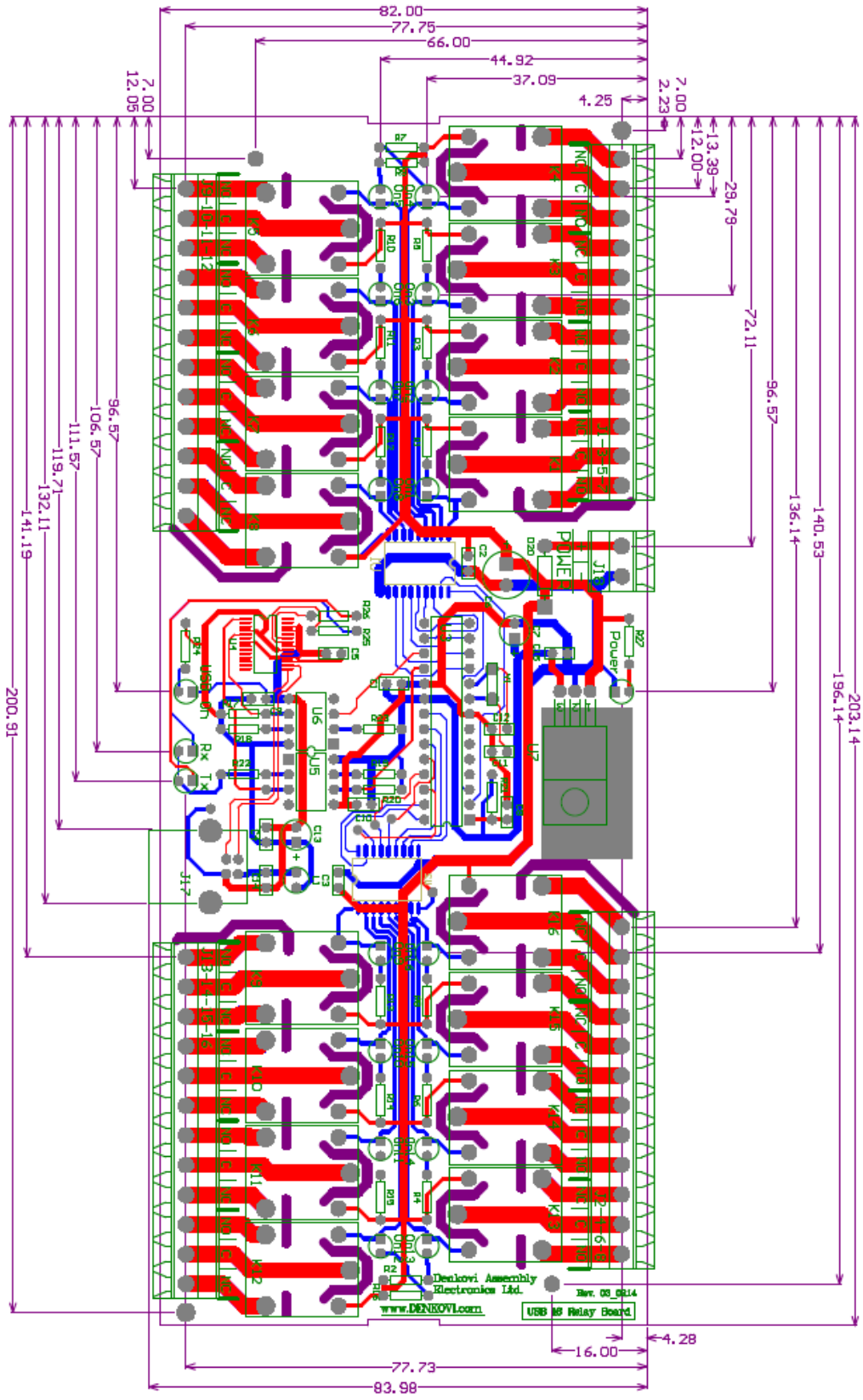


Figure 15. Connecting Lamp with **USB 16 Relay RTU ModBus**

9.2. Connect inductive load to relay

See our tutorial for connecting inductive loads – http://denkovi.com/controlling-inductive-devices.

10. Appendix 2. PCB dimensions



11. Appendix 3. BOX dimensions