DOI: 10.35598/mcfpga.2021.011

Study of the Method of Information Transfer to LED Matrix According to the ModBus Protocol

Sergiy Novoselov
ORCID 0000-0002-3190-0592
Department of Computer-Integrated Technologies, Automation and Mechatronics
Kharkiv National University of Radioelectronics
Kharkiv, Ukraine
sergiy.novoselov@nure.ua

Serhii Tesliuk
ORCID 0000-0003-0711-9250
Department of Computer-Integrated Technologies, Automation and Mechatronics
Kharkiv National University of Radioelectronics
Kharkiv, Ukraine
serhii.tesliuk@nure.ua

***Abstract*—This paper provides an example of using the ModBus industrial protocol to control an LED information board. The developed device can be used in a laboratory workshop to study the work with industrial protocols.**

Keywords—industrial networks, Modbus, Run string, Arduino, Protocol, Automated Module

I. Introduction

Industrial networks are widely used in production processes because they meet the following requirements: work in real time; lack of large arrays of transmitted information; increased reliability of data transmission in industrial conditions; work in different physical environments; coverage of large distances between network nodes; reinforced construction of network equipment; implementation of safe operation of devices in an explosive environment

Features of modern technologies of industrial networks are:

- use of multivariate field devices (they can be used together with several parallel connected circuits 4 ... 20 mA);
- no additional analog-to-digital and digital-to-analog transformations on the communication barrier and in the control system;
- exclusion of calibration of the measuring circuit (the measured value will be converted only once from the analog value to a digital value).

Modbus is one of the most common industrial protocols for data exchange between different devices (machine-to-machine, M2M). The popularity is due to many factors, including ease of implementation, no need to use additional chips, and, of course, the openness of the protocol.

The advantages of the Modbus protocol include:

- reliable error control;
- mass prevalence;
- openness;
- the ability to include in the network a large number of subordinate devices, followed by access to any of them using the address allocated to him.

II. Modbus Protocol Structure

The Modbus structure consists of queries and answers. They are based on a simple protocol package, the so-called PDU (Protocol Data Unit).

The structure of the PDU does not depend on the type of communication line and includes the function code (FCode) and the data field (Data), which is presented in fig. 1.

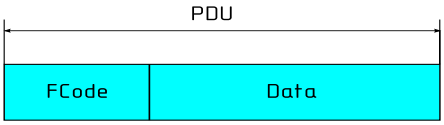

Fig. 1. PDU package structure

To transmit a packet over physical communication lines, the PDU is placed in another packet that contains additional fields. This package is called ADU (Application Data Unit). The general structure of the ADU package for the ModBus RTU is shown in fig. 2.

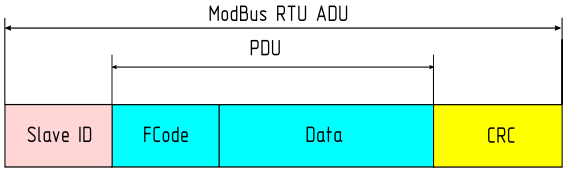

Fig. 2. General ADU package structure for Modbus RTU

The Modbus RTU ADU packet in addition to the PDU packet also includes the Slave ID – the address of the slave device and the checksum CRC16 to verify the correctness of the packet.

This implementation of the Modbus protocol uses the following data types:

- flag – one bit, the register of flags is available for both reading and writing. The flags are stored in the RAM of the microcontroller. 1 byte is allocated for flags, so you can access up to 8 flags;
- discrete register – one bit, read-only. The discrete register is the input port. Discrete registers are a microcontroller status register, so 8 bits are available;

- storage register – a 16-bit register available for reading and writing. Selected cells in the RAM of the microcontroller act as storage registers. 32 bytes are allocated for storage registers, thus access to 16 registers is possible;
- input register – a 16-bit read-only register.

The query consists of the address of the first element of the table, the value of which you want to read, and the number of read elements. The address and amount of data are specified in 16-bit numbers, the high byte of each of them is transmitted first.

The requested data is transmitted in response. The number of bytes of data depends on the number of requested items. One byte is transmitted before the data, the value of which is equal to the number of bytes of data.

The command consists of the element address (2 bytes) and the value is set (2 bytes). For the storage register, the value is just a 16-bit word.

If the command is successful, the slave returns a copy of the request.

The Modbus module interface is the interface from the Modbus server to the user program, which is determined by the program objects. The main functions of the Modbus server are to wait for a Modbus request on the TCP port 502, to process this request, and then to generate a Modbus response depending on the device context.

The principle of message exchange is shown in fig. 3.

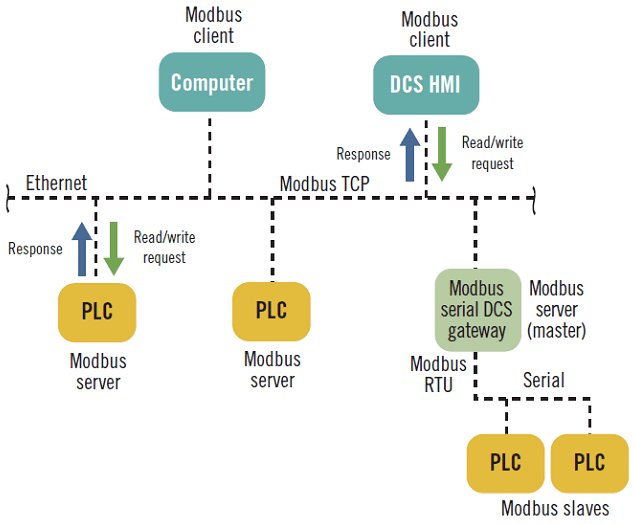

Fig. 3. The principle of messaging exchange

III. Development of the Module for Research of Methods of Transmission of Information

As a module for displaying information we will use an LED matrix of 8 × 8 points. There are various modules of this format. They differ in size, color, principle of management. To create compact devices, there are small modules of a dot-matrix LED matrix of 8 × 8 pixels with red LEDs.

The size of the matrix is 32 mm × 32 mm. The supply voltage of such modules is 1.8 V - 2.3 V. Type of modules 1088AS, connection option - common cathode. The appearance of the modules is shown in fig.4.

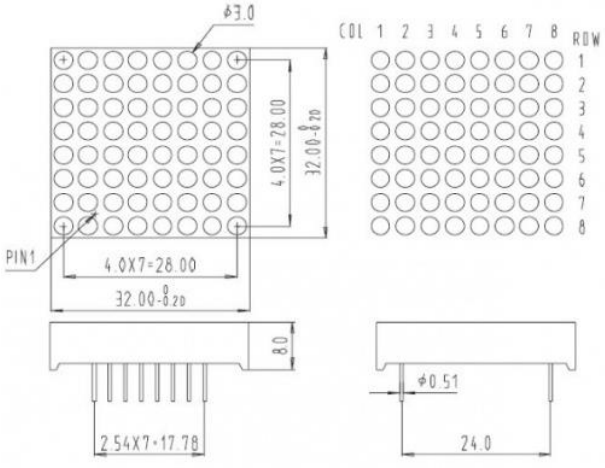

Fig. 4. Appearance of 1088AS modules

Various connection schemes are used to control LED arrays. This can be a variant with a main controller and powerful output transistors (Fig. 5.), or a variant of control via a high-speed bus based on the SPI interface (fig. 6.).

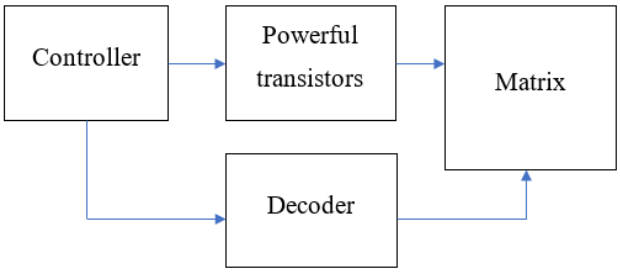

Fig. 5. Control with main controller and powerful output transistors

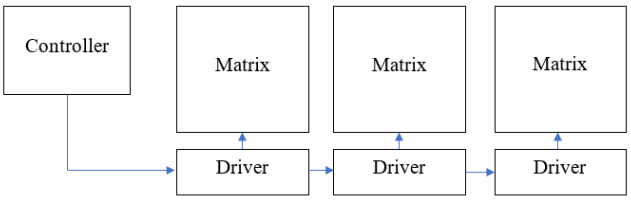

Fig. 6. Control based on the SPI interface

To display the image on the LED matrix, we will use the library Adafruit_GFX. With it, you can organize the output to the LED screen of any image or text.

The library has many features for working with images and text. For example, the "drawChar" function is used to display text.

This function uses the following parameters:

- x and y are the coordinates of the symbol;
- color – the color of the symbol (in the case of a monochrome matrix, the value HIGH, LOW is used);
- bg – background color (in the case of a monochrome matrix, the value HIGH, LOW is used);

- size – scaling factor.

Example of use:

matrix.drawChar(1, y, 'K' HIGH, LOW, 1)

In this example, the "K" symbol will be displayed from the first position of the LED screen.

An example of the application of this function in the program is shown in fig.7.

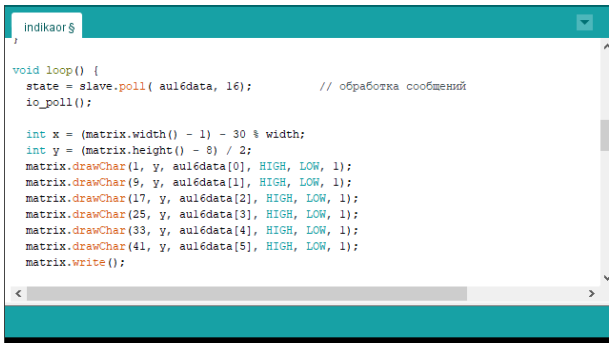

Fig. 7. Example of using the drawChar function

In this example, each character is displayed in a specific character space indicated by the X coordinate. To cycle the text and create the effects of a running tape, you must create a loop and sequentially change the coordinates of the text output each iteration of the loop.

In fig. In Fig. 8 shows an example of creating the effect of a running string.

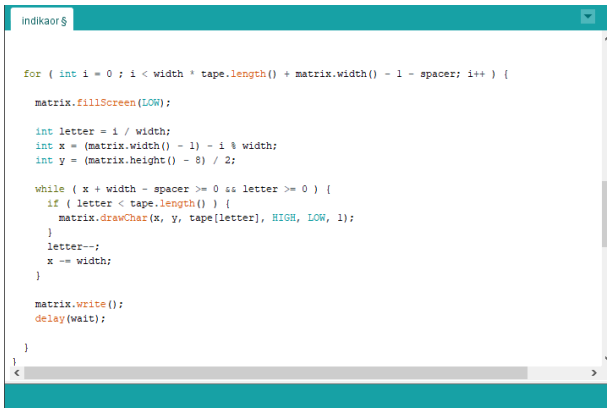

Fig. 8. Example of creating a running string effect

As you can see, the X coordinate in the loop changes with each iteration per unit.

The Max72xxPanel library is used to work with the MAX7219 drivers. This library is designed for the interaction of 8 × 8 LED arrays with a common anode on the chip MAX7219 and Arduino (Fig.9).

The Max72xxPanel library has a setRotation function that sets the orientation of the image on the matrix. For example, if we want to rotate the text 90 degrees, we need to call setRotation with the appropriate arguments immediately after calling the setIntensity function:

matrix.setRotation (0, 1)

The first parameter is the index of the matrix, in our case it is zero; the second parameter is the number of turns by 90 degrees.

The library uses the SPI interface to control the drivers, so the program must also connect the module "SPI.h".

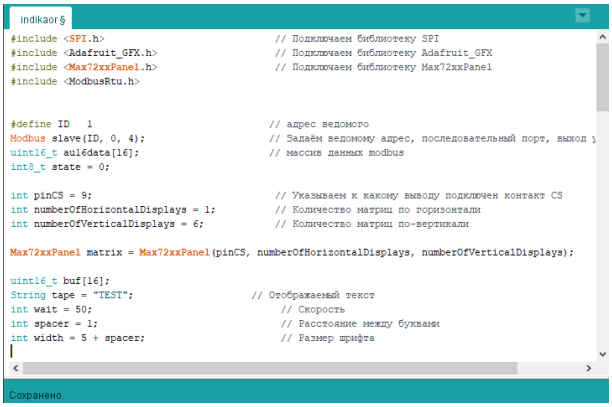

Fig. 9. An example of using this library

The ModBusRtu library was selected to work with the ModBus protocol. To interact with it, the program must create an object by specifying in its ModBus constructor address, serial port number, output number that controls the transmission (for RS485):

ModBus slave (ID, 0, 0)

Then define an array of ModBus registers:

matrix.drawChar(1, y, 'K' HIGH, LOW, 1)

After that, when starting the program, configure the serial port of the slave

slave.begin (9600)

In the main cycle of the program it is necessary to call the function of processing of ModBus messages:

state = slave.poll (au16data, 11)

You can then process the data and save the required variables in the ModBus registers.

The standard provides a separate table for each data type, but a feature of the applied library is that all registers are stored in one array in the form of overlapping tables.

The structure of the registers of the controller will look as follows, as shown in table 1.

TABLE I. STRUCTURE OF REGISTRIV CONTROLER

Register	Bit	Name	Type	ModBus Address	Access
Au16data[0]	0 1 2 3	DT0 DT1 DT2 BTN	discrete	0 1 2 3	read
Au16data[1]	0 1 2 3	CL16 CL17 CL18 LED	coil	16 17 18 19	read/write
Au16data[2]		INPT3	input	2	read
Au16data[3]		INPT4	input	3	read
Au16data[4]		INPT5	input	4	read
Au16data[5]		HOLD6	holding	5	read/write
Au16data[6]		HOLD7	holding	6	read/write
Au16data[7]		HOLD8	holding	7	read/write

During the operation of the program, the data transmitted by ModBus is written to the registers au16data []. The register number is defined in the data packet transmitted by the RS-485 interface.

According to the standard, information is written to the Holding Registers – 16-bit outputs of the device, or internal values.

Available for reading and writing. Range of addresses of registers: from 40001 to 49999. Contain functions: "03" – reading of group of registers, "06" – record of one register, "16" – record of group of registers. Each character is transferred in HEX format. The character number is taken from the ASCII table.

For example, to transmit the word "ModBus" to the screen, you need to create the following data packet:

01 10 00 00 00 07 00 00 4D 00 6F 00 64 00 42 00 75 00 73 00 72 AC

The following parameters are specified in this package:

- device number: 01;
- function number: 10h (function 16 is used for writing in several registers at once);
- starting address: 00 00;
- number of registers: 00 07 (7 registers);
- bytes "00 00 4D 00 6F 00 64 00 42 00 75 00 73 00 72 AC" form the word "ModBus" (the first two bytes are ignored by the library).

Figure 10 shows an example of the interaction of the ModBus protocol.

Fig. 11 shows an example of the transfer of the word "KITAM" on the board.

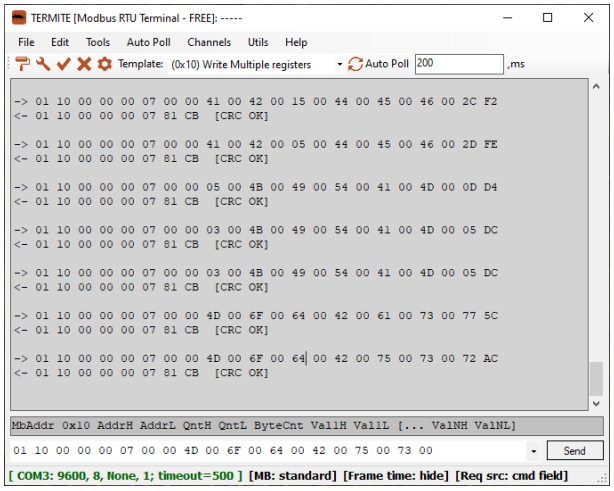

Fig. 10. Interaction of the ModBus protocol with the developed layout

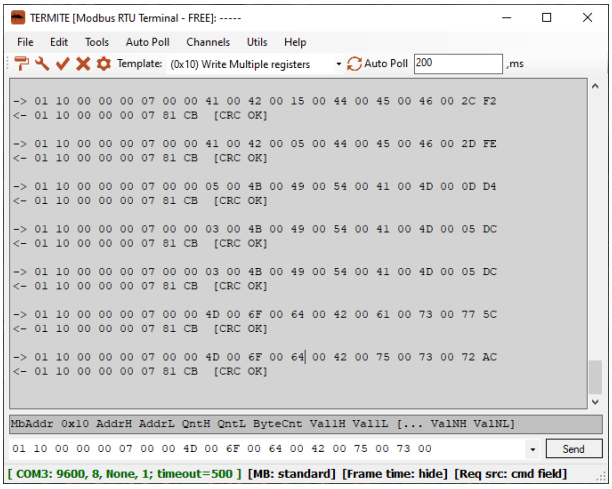

Fig. 11. Example of passing the word "KITAM" on the board

After transferring the data to the layout, the text will be displayed on the screen as shown in fig.12.

Fig. 12. Display the transmitted characters on the scoreboard

Fig. 13. The result of displaying information

CONCLUSIONS

This paper provides an example of using the ModBus industrial protocol to control an LED information board. The developed device can be used in a laboratory workshop to study the work with industrial protocols.

To control the layout via the RS-485 interface using the Modbus protocol, you need to select two modules:

- USB-to-RS-485 interface converter on the PC side;
- RS-485 to TTL converter on the layout side.

The protected USB-RS485 adapter module on the FT232 chip is selected as the USB to RS-485 interface converter on the PC side.

REFERENCES

[1] ModBus/RTU, ModBus/ASCII та ModBus/TCP [Електронний ресурс]. – Режим доступу: www / URL: http://oscada.org/wiki/Modules/ModBus/uk – 17.05.2021 р. – Загол. з екрану.

[2] ModBus, протокол: опис, сфера застосування, переваги і недоліки цього протоколу [Електронний ресурс]. – Режим доступу: www / URL: https://tostpost.com/uk/komp-yuteri/19477-ModBus-protokol-opis-sfera-zastosuvannya-perevagi-nedol-ki.html – 24.05.2021 р. – Загол. з екрану.

[3] Кодировка символов Windows-1251 [Електронний ресурс]. – Режим доступу: www / URL: https://wm-school.ru/html/html_win-1251.html – 25.05.2021 р. – Загол. з екрану.

[4] Arduino Mega2560 [Електронний ресурс]. – Режим доступу: www / URL: http://wiki.amperka.ru/продукты:arduino-mega-2560.html – 29.05.2021 р. – Загол. з екрану.

[5] Arduino IDE основи програмування [Електронний ресурс]. – Режим доступу: www / URL: http://geekmatic.in.ua/ua/arduino_osnovyi_programmirovaniya – 27.05.2021 р. – Загол. з екрану.

[6] Світлодіодна матриця 2088bs [Електронний ресурс]. – Режим доступу: www / URL: https://hcomp.ru/shop/ledmatrixld208bs/ – 16.05.2021 р. – Загол. з екрану.

[7] MAX7219 – драйвер светодиодных индикаторов [Електронний ресурс]. – Режим доступу: www / URL: https://radiolaba.ru/microcotrollers/max7219-drayver-svetodiodnyih-indikatorov.html – 19.05.2021 р. – Загол. з екрану.

[8] Как работать с драйверами индикаторов MAX7219 и MAX7221 [Електронний ресурс]. – Режим доступу: www / URL: https://radiohlam.ru/max7219_7221/ – 19.05.2021 р. – Загол. з екрану.

[9] Arduino USB UART чипы и драйвера CH340, CH340G, FTDI [Електронний ресурс]. – Режим доступу: www / URL: https://arduinomaster.ru/platy-arduino/arduino-usb-uart-chipy-i-drajvera-ch340-ch340g-ftdi/ – 01.06.2021 р. – Загол. з екрану.

[10] Конвертер USB в RS-485 [Електронний ресурс]. – Режим доступу: www / URL: https://mysku.ru/blog/aliexpress/28020.html – 19.05.2021 р. – Загол. з екрану.

[11] Промышленные роботы [Електронний ресурс]. – Режим доступу: www / URL: https://znaimo.com.ua/Промисловий_робот – 17.05.2021 р. – Загол. з екрану.

[12] Інтерфейсний модуль MAX485 UART-RS485 [Електронний ресурс]. – Режим доступу: www / URL: https://arduino.ua/prod1343-interfeisnii-modyl-max485-uart-rs485 – 27.05.2021 р. – Загол. з екрану.