RS232/484/422 Communication Module AN3485 User Manual

ALINX

Version Record

Version	Date	Release By	Description
Rev 1.0	2022-04-30	Rachel Zhou	First Release

The ALINX official Documents are in Chinese, and the English version was translated by **Shanghai Tianhui** Trading Company. They has **not been officially Review by ALINX** and are for reference only. If there are any errors, please send email feedback to support@aithtech.com for correction.

Amazon Store: https://www.amazon.com/alinx

Aliexpress Store:

https://alinxfpga.aliexpress.com/store/911112202?spm=a2g0o.detail.1000007.1.704e2bedqLBW90

Ebay Store: https://www.ebay.com/str/alinxfpga

Customer Service Information

Skype: rachelhust@163.com

Wechat: rachelhust

Email: support@aithtech.com and rachehust@163.com

Table of Contents

Part 1: RS232/485/422 Communication Module

The ALINX AN3845 module, RS232/485/422 communication module designed for industrial field applications. It includes one RS232 interface, two RS485 and two RS422 communication interfaces. Cooperate with the FPGA development board to realize remote data transmission and communication of RS232, 485 and 422.

The RS232, 485, and 422 interfaces use the MAX3232, MAX3485, and MAX3490 chips as level-shifting chips. The AN3485 module has a 40-pin female header for connecting to the FPGA development board. The RS232 interface is a standard DB9 serial port, which can be directly connected to a computer or other device through a serial cable. The RS485 and RS422 interfaces are connected to the external terminals by terminals. The ultra-long-distance transmission can reach up to several kilometers. In addition, the RS485 and RS422 interface parts are equipped with ESD protection function of plus or minus 15KV

Figure 1-1 is the AN3485 module product photo as below:

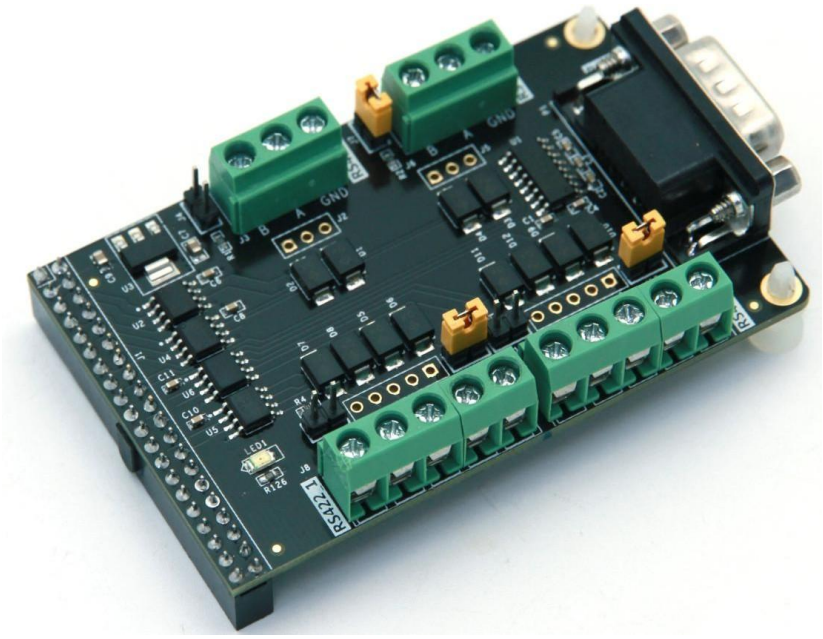


Figure 1-1: AN3485 Communication Module Photo (Front side)

Part 1.1: AN3485 Communication Module Detail Parameter

➢ **RS232 Interface**

- One-channel standard DB9 serial interface
- Max3232 to convert between RS232 and TTL (Transistor-Transistor Logic) level signals
- Transfer speed up to 120kbps

➢ **RS485 Interface**

- 2-channel RS484, connected with 3 wire terminals
- Max3485 to convert between RS485 and TTL (Transistor-Transistor Logic) level signals
- Industrial design, Strong anti-interference ability, and effective lightning protection
- With a 120 ohm matching resistor, a jumper cap can be inserted to enable matching resistors, and short-circuiting is recommended for long-distance transmission.
- Support multi machine communication, Max. allowed 128 devices in bus
- The transmission rate is up to 500Kbps data communication rate.

➢ **RS422 Interface**

- 2-channel RS422, connected with 5 wire terminals
- Max3490 to convert between RS422 and TTL (Transistor-Transistor Logic) level signals
- Industrial design, Strong anti-interference ability, and effective lightning protection
- With a 120 ohm matching resistor, a jumper cap can be inserted to enable matching resistors, and short-circuiting is recommended for long-distance

transmission.

- Support multi machine communication, Max. allowed 128 devices in bus
- The transmission rate is up to 500Kbps data communication rate.

Part 1.2: AN3485 Communication Module Form Factors

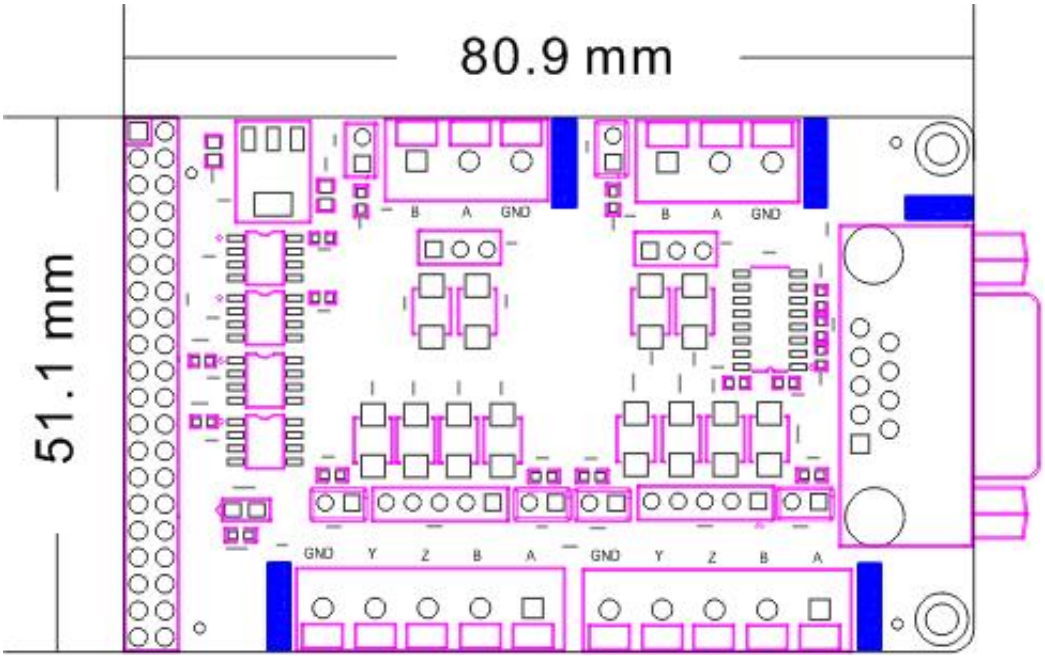


Figure 1-2: AN3485 Module Form Factors

Part 2: AN3485 Module Function Description

Part 2.1: RS232 Circuit Design

The RS232 interface of the AN3485 module uses the MAX3232 chip to convert RS232 and +3.3V TTL levels. The TTL level serial port receive and transmit signals (RXD, TXD) are connected to the 40-pin connector for serial communication with the external FPGA chip or ARM chip. The maximum speed of RS232 serial communication is 120kbps. The schematic diagram of the RS232 interface is shown in Figure 2-1

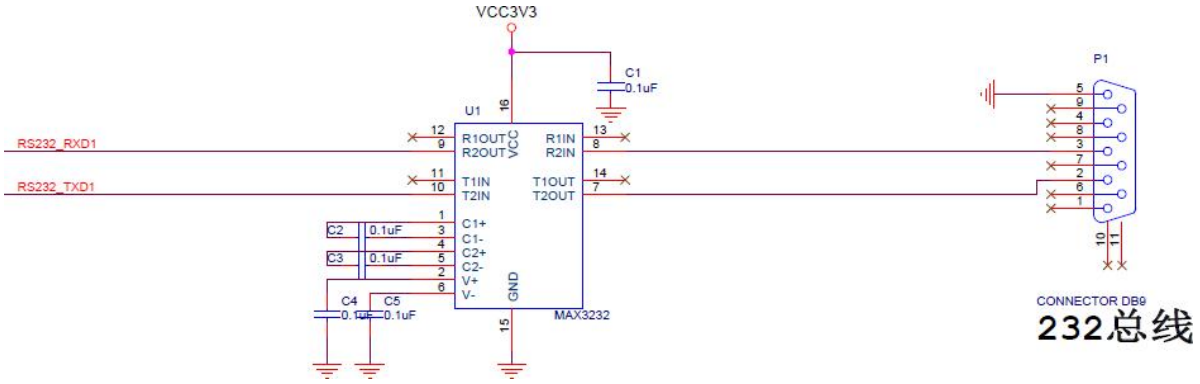


Figure 2-1: The RS232 Circuit Design

Part 2.2: RS485 Circuit Design

The RS485 interface of the AN3485 module uses the MAX3485 chip to convert RS485 and +3.3V TTL levels. The TTL level 485 signal (RXD, DE, TXD) is connected to the 40-pin connector to communication with the external FPGA chip or ARM chip for 485 data communication, where the DE signal is used to control the 485 transmission direction.

The RS485 bus has a 120 ohm termination resistor. When the 485 interface is used as a slave, the jumper cap needs to be connected to enable a 120 ohm termination resistor. In addition, TVS transient suppression diodes are connected to the RS485 A and B buses to provide positive and negative 15KV ESD protection. The maximum speed of RS485 communication is 500kbps. The schematic diagram of the RS232 interface is shown in Figure 2-2.

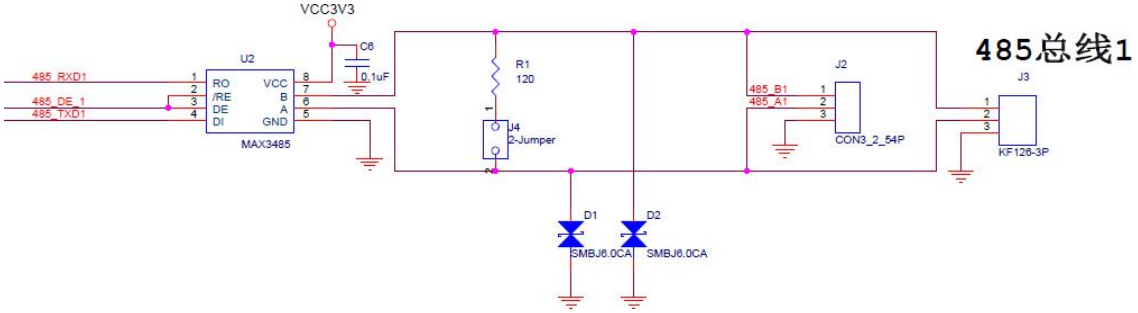


Figure 2-2: The RS485 Circuit Design

Part 2.3: RS422 Circuit Design

The RS422 interface of the AN3485 module uses the MAX3490 chip to convert RS422 and +3.3V TTL levels. The TTL level 422 signal (RXD, TXD) is connected to the 40-pin connector to communicate with the external FPGA chip or ARM chip for 422 data communication. Because the RS422 transmission is full-duplex, there is no DE signal.

A 120 ohm termination resistor is reserved between the RS422 transmit bus Z-Y and the receive bus A-B. The user only needs to connect the jumper cap of the receiving bus (A-B) to enable 120 ohm terminal matching. In addition, TVS transient suppression diodes are connected to the RS422's receive and transmit buses to provide positive and negative 15KV ESD protection. The maximum speed of RS422 communication is 500kbps. The schematic diagram of the RS422 interface is shown in the figure 2-3.

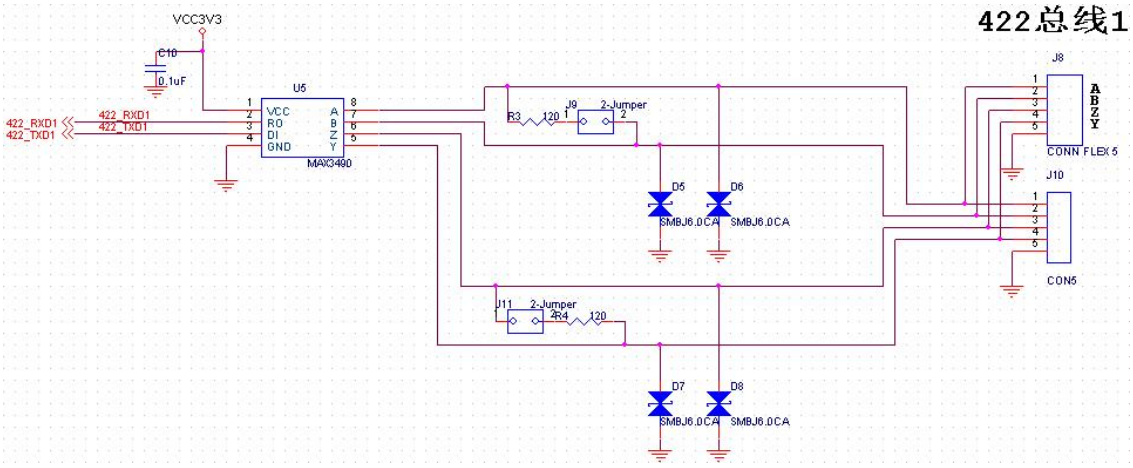


Figure 2-3: The RS422 Circuit Design

Part 2.4: 40-pin female pin assignment

Pin	Pin Name	Description
1	GND	Ground
2	+5V	5V Power Input
3	-	-
4	-	-
5	485_RXD1	First RS485 data receive
6	485_DE_1	First RS485 Send Enable
7	485_TXD1	First RS485 data transmit
8	-	-
9	-	-
10	-	-
11	485_RXD2	Second RS485 data receive
12	485_DE_2	Second RS485 Send Enable
13	485_TXD2	Second RS485 data transmit
14	-	-
15	-	-
16	-	-
17	RS232_RXD1	RS232 data receive
18	RS232_TXD1	RS232 data transmit
19	-	
20	-	
21	422_RXD2	Second RS422 data receive
22	422_TXD2	Second RS422 data transmit
23	-	-
24	-	-
25	422_RXD1	First RS422 data receive

Pin Assignment of 40-Pin Female Connector

Part 3: RS232 Communication Program

Part 3.1: RS232 Communication Program Introduction

The RS232 communication routine (rs232_test) mainly demonstrates the serial port receiving and transmitting functions of the FPGA development board and the AN3485 communication module. The user needs to prepare an RS232 serial cable (straight connection) to connect the FPGA development board to the PC for serial port data communication.

In the routine, when the PC sends no information, the FPGA program will continuously send the message "Hello ALINX AN3485" to the PC through the serial port of AN3485. When the user sends data from the serial port tool of the PC to the AN3485 communication module to the development board, the FPGA program sends the data back to the PC after receiving the data, thereby implementing the loopback function.

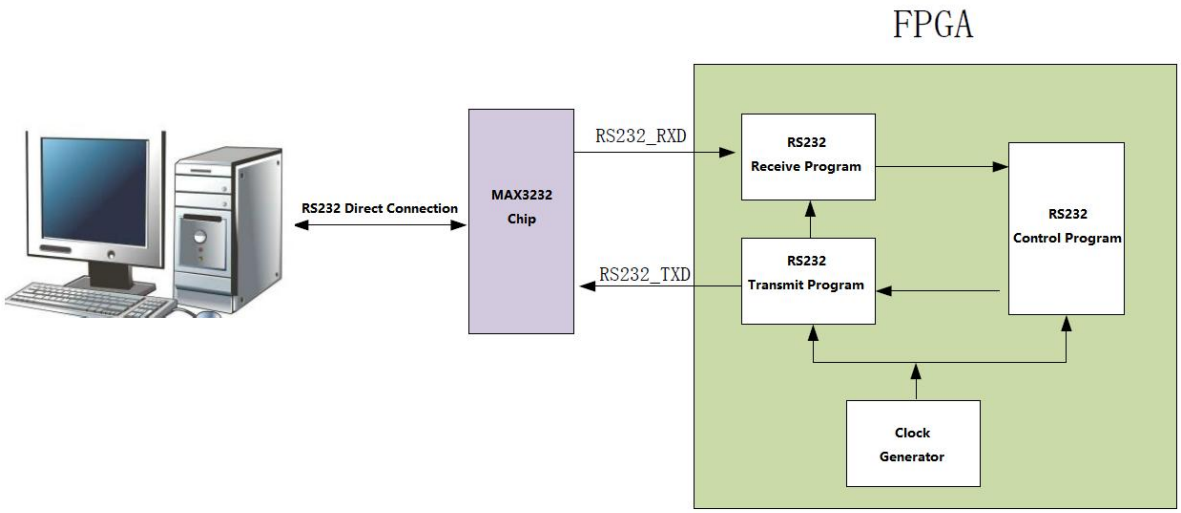


Figure 3-1: The Design Block Diagram of the FPGA Program

The serial communication routine consists of a TOP program rs232_test.v

and four subroutines. The four subroutines are RS232 transmit program uarttx.v, RS232 receive program uartrx.v, clock generator program clkdiv.v and RS232 send control program uartctrl.v. The design block diagram of the FPGA program is shown in Figure 3-1

Let's take a look at each program separately

1. Clock Generator Program clkdiv.v

It is used to generate the uart receiving and transmitting clock signal with a baud rate of 9600bps. The program divides the system clock of 50Mhz. The frequency dividing parameter 326 is calculated as follows:

The clock clkout generated here is 16 times the baud rate. Assuming the baud rate of the data is p, the frequency of the clock clkout here is 16*p. Taking the baud rate p of 9600 as an example, if the system clock is 50MHz, the division factor is 50000000/(16*9600) = 325.5, which is rounded to 326.

The purpose of the clkout clock is 16 times the baud rate. In order to receive the data received by each bit, there are 16 clock samples in the uart reception, and the intermediate sample values are taken to ensure that the samples do not slip or error.

2. RS232 Transmit Program uarttx.v

Control one byte of data transmission process of serial port. When in the idle state, the transmission line (TX) of the serial port is at a high potential; when the transmission data command is received, the time T of one data bit of the line is pulled down, and then the data is sequentially transmitted from the low bit to the high bit. After the data transmission is completed, the parity bit and the stop bit are transmitted (the stop bit is high), and the transmission of one frame of data (one byte) ends.

3. RS232 Receive Program uartrx.v

Control one byte of data receiving process of serial port. When in the idle state, the receiving line (RX) of the serial port is at a high potential; when the falling edge of the line RX is detected (the line potential changes from high to low), the line has data transmission, according to the agreed baud rate from the low point to the upper bit receives the data. After the data is received, it then receives and compares the parity bit. If it is correct, it notifies the subsequent device to receive the data or store it in the cache.

Since the UART is asynchronously transmitted, no synchronous clock is transmitted. In order to ensure the correctness of data transmission, the UART uses a clock of 16 times the data baud rate for sampling. Each data has 16 clock samples, taking the middle (8th clock) sample value to ensure that the sample will not slip or bit error. Generally, the number of data bits per frame of a UART is 8, so that even if each data has a clock error, the receiver can correctly sample the data.

4. RS232 Send Control Program uartctrl.v

After power-on, the program continuously sends the "Hello ALINX AN3485" string to the serial port. If the serial port receives the data sent from the PC, it will send the received data back to the PC.

Part 3.2: RS232 Communication Hardware Connection

The hardware connection between the AN3485 module and the FPGA development board is very simple. Just plug the 40-pin female J1 of the AN3485 module into the expansion port of the FPGA development board, and the pin 1 of the connector is aligned. Figure 3-2 detailed the hardware connection between the J1 expansion port of the ALINX AX301 FPGA development board and the AN3485 communication module (If connect to the J2 expansion port of

the FPGA development board, the pins need to be reassigned). If you use other FPGA development boards, users need to assign pins themselves.

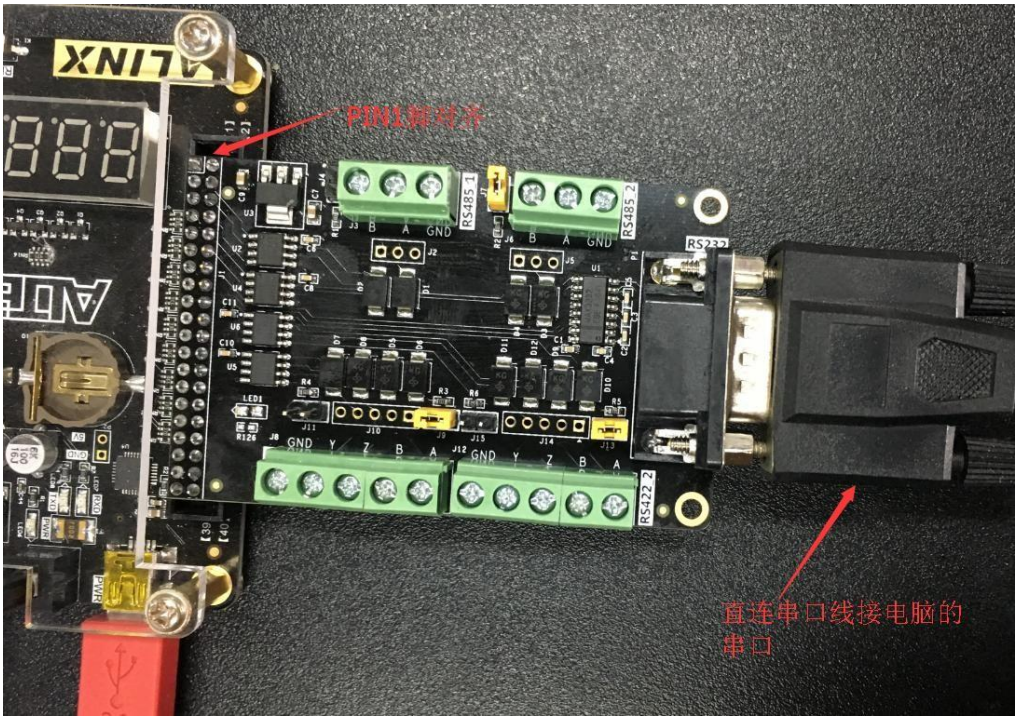


Figure 3-2: Hardware Connection

Part 3.3: RS232 communication experiment

After the FPGA development board is powered on, turn on the serial debugging assistant, select the correct COM1 or COM2, set the baud rate to 9600, set the parity bit to odd or even parity, and stop bit to 1. The specific configuration is as Figure 3-3:

Figure 3-3: The Specific Configuration of Serial Debugging Assistant

After downloading the program, we can see the "Hello ALINXAN3845" information from the display window, that sent by FPGA development board continuously.

Figure 3-4: After Download the Program

Send "Hello world" from the serial debugging assistant, you can see that the FPGA will immediately return the same information to the PC.

Part 4: RS485 communication program

Part 4.1: RS485 communication program introduction

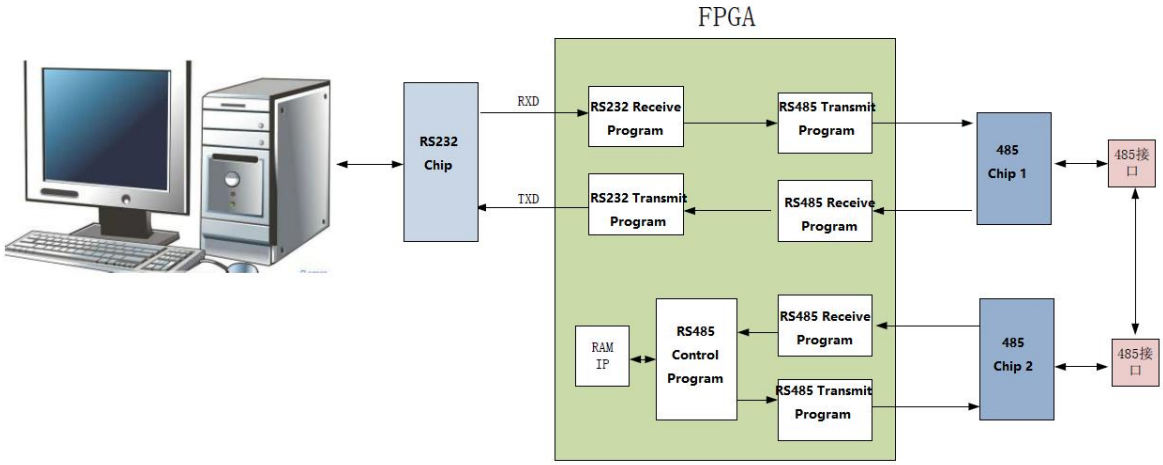
The function of RS485 communication routine (rs485_test) mainly demonstrates the function of receiving and transmitting RS485 on AN3485 communication module, because there are two RS485 interfaces on AN3485 module. This experiment is 485 data communication by connecting to these two 485 buses.

RS485 data communication is one-way. Only one interface is used as the data transmitter at the same time. Other interfaces can only be used as data receivers. Therefore, RS485 data communication requires protocol

requirements. The receipt of RS485 data in our history will judge the first word of the instruction. If the first word of the instruction is judged to be 'S', it indicates that it is an instruction sent to itself, and it needs to receive and return the instruction. If the first word of the instruction is not 'S', the instruction is ignored and the instruction ends with a carriage return or line feed.

Before RS485 data communication, the PC needs to send the command to the RS232 serial port. The first word of the command is 'S', followed by a piece of data, preferably with a carriage return or line feed. After receiving the serial port command, the FPGA program is sent out from the first RS485 interface, and then received through the second RS485. The FPGA program determines the RS485 receive command. If the first word of the command is 'S', the instruction is stored in the RAM. Until the carriage return is received or the line feed ends. The received command is also sent from the second RS485 interface and then received via the first RS485. The data received by the first RS485 is sent directly to the RS232 serial port and returned to the computer. Implement the functionality of Loopback.

The RS485 communication routine consists of a TOP program rs485_test.v and 5 subroutines. The 5 subroutines are RS232 transmitter uarttx.v, RS232 receiver uartrx.v, RS485 transmitter rs485_tx.v, RS485 receiver rs485_rx.v, and RS485 control program rs485_ctrl.v. The design block diagram of the FPGA program is shown below:



Let's introduce each program separately. RS232 sending program and RS232 receiving program have been explained in the RS232 communication program, no longer introduced.

1. RS485 transmitting program rs485_tx.v

The RS485 transmission program is basically the same as the RS232 transmission program, but the RS485 transmission program has one more transmission enable control signal DE. This signal needs to be set high when RS485 data is transmitted, other times is set to low.

2. RS485 receiving program rs485_rx.v

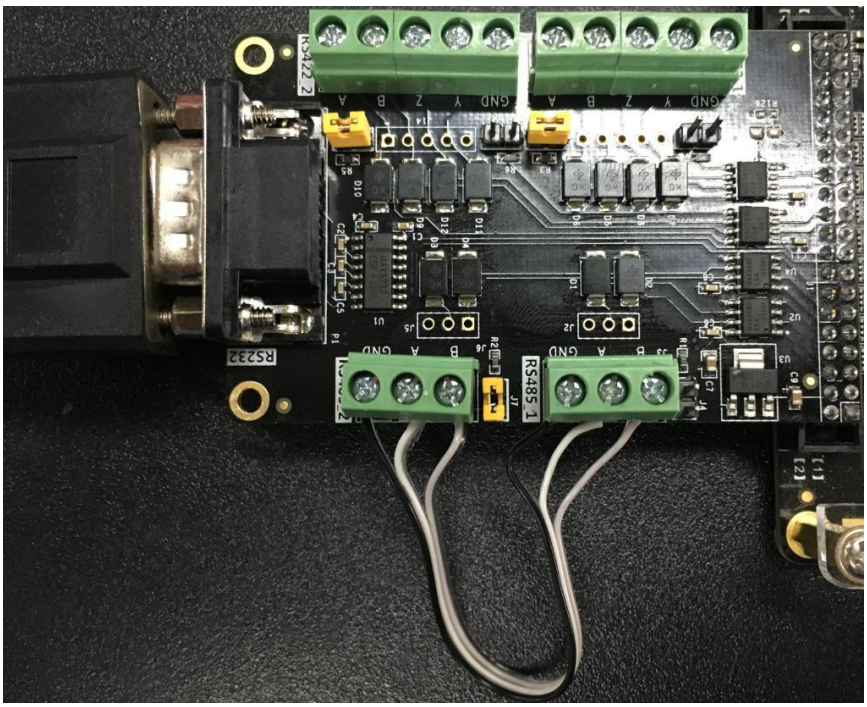
The RS485 receiver is the same as the RS232 receiver.

3. RS485 control program rs485_ctrl.v

The RS485 control program determines the receive command of RS485. If the first word of the receive command is 'S', the instruction is stored in RAM until a carriage return is received or a newline character ends the receipt of an instruction. If an instruction is received, the RS485 transmission program is started, and the received command is sent out. The length of the transmission is the length of the received command.

Part 4.2: RS485 communication hardware connection

First, you need to connect the RS485 first channel and the second channel 485 bus (A and A are docked, B and B are connected, GND is connected to GND), otherwise data communication is impossible. Also plug the AN3485 module into the expansion port (J1) of the AX301 FPGA development board. The RS232 interface is connected to the serial port of the computer with a serial cable. The hardware connection is shown below:



Part 4.3: RS485 communication experiment

Power on the FPGA Development board, and download the rs485_test.sof file to the FPGA, open the serial debugging assistant, select the correct COM1 or COM2, set the baud rate to 9600, set the parity bit to odd or even parity, and stop bit to 1.Then the serial port sends the command with the uppercase “S” as the start character, followed by a string of data, and the carriage return as the

terminator. For example, after we send the string "S0123456789ABCDEFGHIJK Enter" here, the serial port can receive the same string, indicating that RS485 data communication is normal.

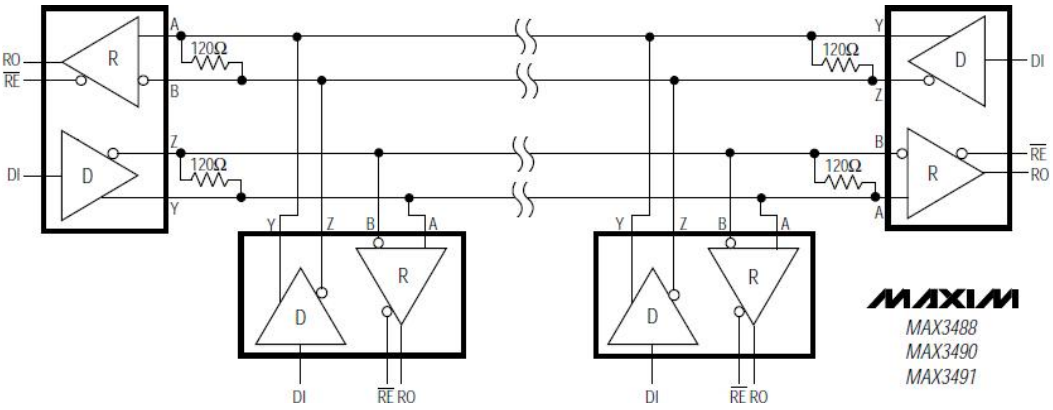
It must be noted here that the command needs to start with the uppercase 'S', the carriage return or the newline character ends, otherwise the RS485 communication is unsuccessful.

Part 5: RS422 communication program

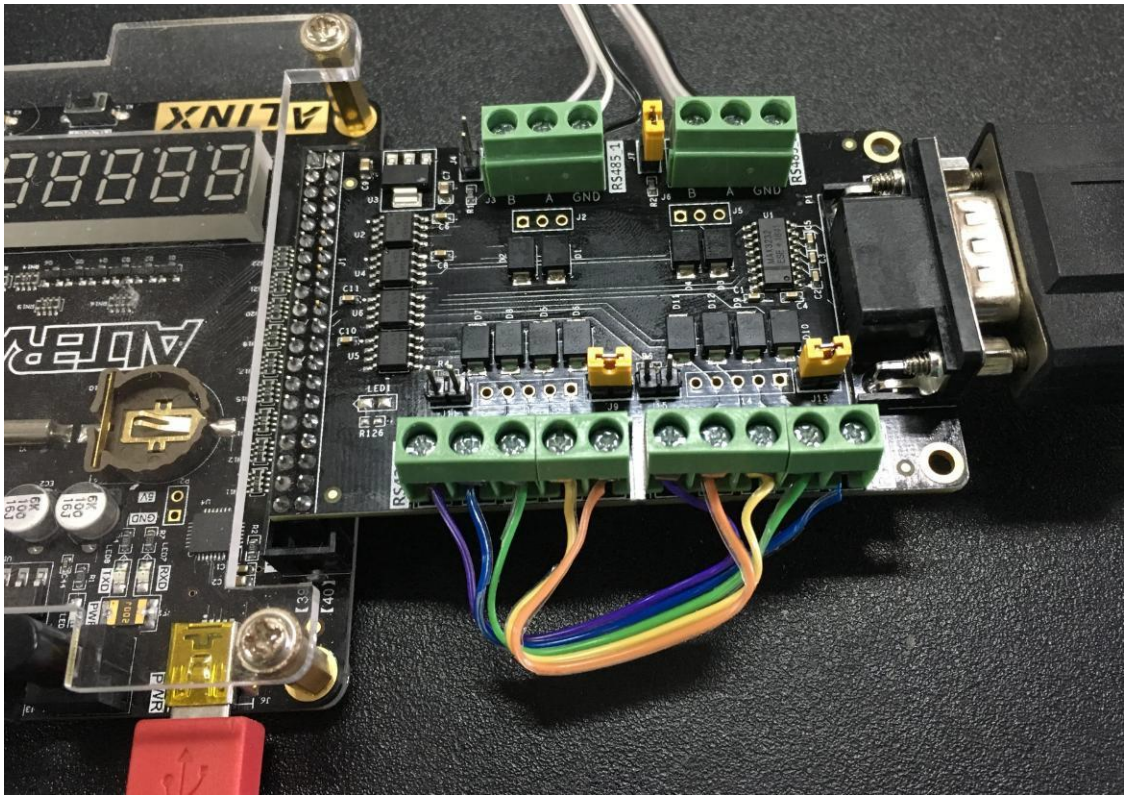
The RS422 communication program is basically the same as the RS485 routine, except that RS422 does not require a DE signal. We will not describe the program here. In addition, the experimental method is the same as RS485 communication, sending instructions from the computer to the RS232 serial port, sending through the first RS422, the second RS422 receiving the command,

and then sending the second RS422 through the second RS422, the first After receiving the post command, the RS422 returns to the RS232 serial port to the computer.

RS422 bus connection between the first and second channels requires special attention. A and B are the data reception of RS422, and Y and Z are the data transmission of RS422. Therefore, the first channel A and B need to be connected with the second channel Y and Z. The first channel Y and Z need to be connected with the second channel A and B. The following figure shows the connection diagram of the MAX3490 chip manual.



AN3485 module hardware connection diagram of the two-way RS422 data communication is as follows (A and Y are connected, B and Z are connected, GND is connected to GND), and the line should not be reversed.



Experimental methods and result refer to the chapter on RS485 communication.