SDLC-ETH

Datasheet

Email : yacer@yacer.cn
Website : www.yacer.cn

yacer 亚册
Building Blocks of Communication

Table of Contents

1 Overview

1.1 Introduction

The Yacer SDLC-ETH protocol converter, provides four high-speed RS-232/422/485 multi-protocol serial ports, supports synchronous HDLC/SDLC protocol and asynchronous UART mode. One Gigabit Ethernet port, realizes protocol conversion between serial and ethernet, supports serial Ethernet bridge function.

4~17 VDC wide voltage power supply, industrial wide temperature.

1.2 Features

- One 10/100/1000M Ethernet interface
- Four RS-232/422/485 synchronous/asynchronous serial ports, Baud rate up to 20 Mbps
- Synchronous HDLC/SDLC, Asynchronous UART, Bit stream
- NRZ, NRZI, DBPL, Manchester and Differential Manchester encoding formats
- 15 KV ESD protection
- 4 ~17 VDC wide voltage power supply
- Industrial wide temperature

1.3 Applications

- Protocol conversion between the serial port and Ethernet
- Ethernet Bridging function between Ethernet interface and serial port
- Conversion between synchronous and asynchronous serial ports
- Multiplexer of serial ports
- High-speed synchronous/asynchronous serial data communication and conversion
- Telemetry, measurement and control data acquisition and transmission
- Satellite, radio data transmission
- Air Traffic Control (ATC), Air Traffic Management (ATM)

1.4 Technical Specifications

Serial Ports	
Quantity	4
Connector	1 x D-Sub 44 (Female)
Working mode	Synchronous HDLC/SDLC, Asynchronous UART Synchronous Bit stream
Interface type	RS-232, RS-422, RS-485 (Configurable by software)
Duplex mode	Full-duplex, Half-duplex (Configurable by software)
Encoding format	NRZ, NRZI, Manchester, Differential Manchester, DBPL(Differential Bi-Phase Level)
Baud rate	Synchronous: ≤ 20 Mbps Asynchronous: ≤ 3.5 Mbps
Synchronous clock	Normal, slave and master clock mode
ESD protection	± 15 KV
Ethernet Interface	
Connector	1 x RJ-45
Speed	10/100/1000 Mbps, Auto MDI/MDI-X
Network protocol	TCP/IP
Programming interface	UDP Server, UDP Client Unicast/Multicast/Broadcast
Ethernet bridge	Serial Ethernet bridge
Configuration Management	
Configuration tool	yacer-DMS configuration management software
Configuration interface	Ethernet Interface
Power Requirements	
Power Supply	4 ~ 17 VDC
Power consumption	< 3 W
Power interface	DC Power Jack 5.5x 2.1 mm (Male) Terminal 5.0mm
Mechanical Characteristics	
Dimensions	H x W x D: 30 mm x 132 mm x 165 mm
Weight	600g
Operating Environment	
Operating temperature	-40 ~ +70℃
Storage temperature	-40 ~ +85℃
Operating humidity	5 ~ 95% RH (no condensation)

1.5 Baud rate range

Interface type	Working mode	Encoding format	Baud rate
RS-232	Asynchronous		≤ 500 Kbps
RS-232	Synchronous	NRZ	≤ 500 Kbps
RS-232	Synchronous	NRZI, DBPL, Manchester, Diff Man	≤ 250 Kbps
RS-422/485	Asynchronous		≤ 3.5 Mbps
RS-422/485	Synchronous	NRZ	≤ 20 Mbps
RS-422/485	Synchronous	NRZI, DBPL, Manchester, Diff Man	≤ 10 Mbps

1.6 Order Information

Model	RS-232/422/485 Serial Ports	Ethernet Interface
SDLC-ETH-201	2 x Asynchronous 2 x Synchronous/Asynchronous	1 x 10/100/1000M
SDLC-ETH-401	4 x Synchronous/Asynchronous	1 x 10/100/1000M

2 Hardware and Physical Interface

2.1 Appearance

D-Sub 44 female serial interfaces (S1-S4) and Ethernet interface (GE) are located on one end of the product, while DC power jack and LED indicators are located on the other end. Silk screen is located on the front.

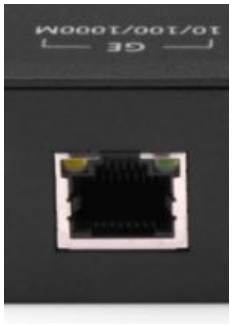
2.2 LED Indicators

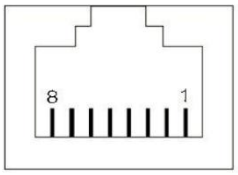
LED	Description
ALARM	Alarm indicator, on when the device is not ready to start or in case of failure, and constantly off during normal operation
RUN	Running indicator, flashing during normal operation
POWER OUT	Power OK
POWER IN	Power input

2.3 Ethernet Interface

RJ-45 interface, 10/100/1000M Ethernet, support for auto MDI/MDI-X crossover.

RJ-45	Signal
1	DA +
2	DA -
3	DB +
4	DC +
5	DC -
6	DB -
7	DD +
8	DD -





2.4 Power Interface

SDLC-ETH is powered by the 4~17V DC power supply.

The interface adopts power jack DC 5.5x2.1mm male socket.

2.5 Serial Port

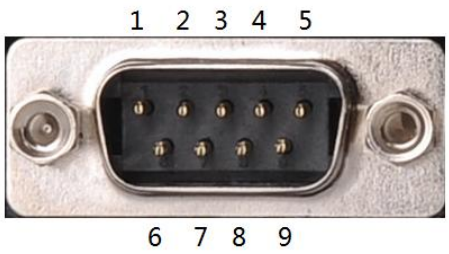
2.5.1 Functional Description

Serial port supports synchronous HDLC protocol and asynchronous UART mode of operation, and encoding format supports NRZI, Manchester, Differential Manchester, DBPL.

It supports three physical layer standards, RS-232, RS-422 and RS-485. It can be set arbitrarily by software according to requirements.

2.5.2 Pin Definition

D-sub44 female connector, providing the S1 ~ S4 serial ports. It can be converted to the quad D-sub9 male connector with the YC9-44T cable.



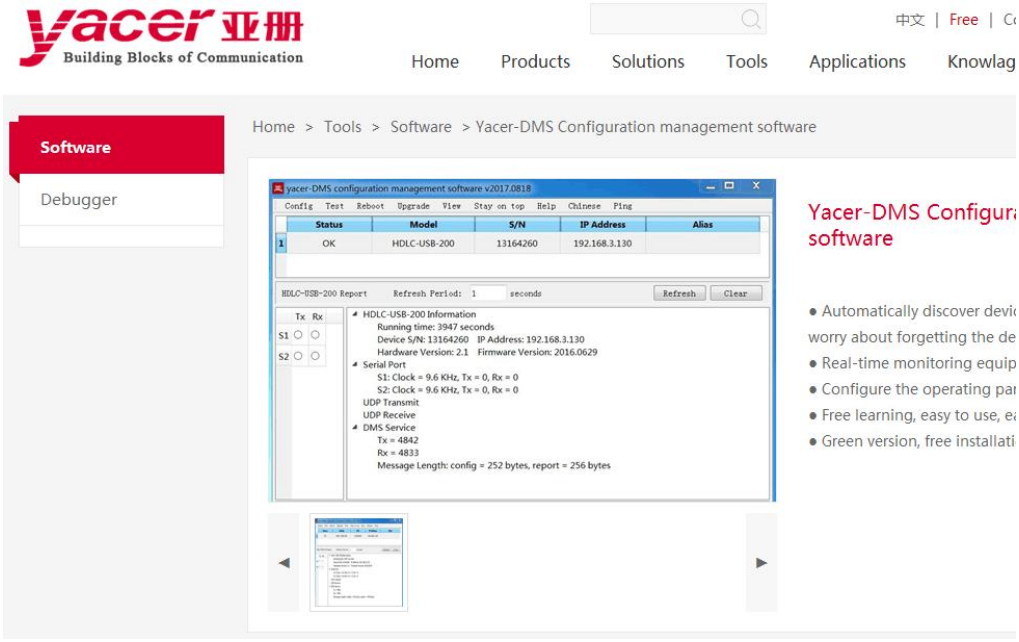
D-Sub44 Female		RS-232	RS-422	RS-485	YC9-44T Cable
Serial	PIN	Full-duplex	Full-duplex	Half-duplex	D-Sub9 Male
S1	32	TxData1	TxData1 +	Data1 +	S1-5
	31		TxData1 -	Data1 -	S1-9
	18	TxClock1	TxClock1 +	Clock1 +	S1-4
	3		TxClock1 -	Clock1 -	S1-8
	1	RxData1	RxData1 +		S1-1
	16		RxData1 -		S1-6
	17	RxClock1	RxClock1 +		S1-2
	2		RxClock1 -		S1-7
	33	GND	GND	GND	S1-3
S2	4	TxData2	TxData2 +	Data2 +	S2-5
	19		TxD2ata -	Data2 -	S2-9
	22	TxClock2	TxClock2 +	Clock2 +	S2-4
	7		TxClock2 -	Clock2 -	S2-8
	5	RxData2	RxData2 +		S2-1
	20		RxData2 -		S2-6
	21	RxClock2	RxClock2 +		S2-2
	6		RxClock2 -		S2-7
	35	GND	GND	GND	S2-3
S3	8	TxData3	TxData3 +	Data3 +	S3-5
	23		TxData3 -	Data3 -	S3-9
	26	TxClock3	TxClock3 +	Clock3 +	S3-4
	11		TxClock3 -	Clock3 -	S3-8
	9	RxData3	RxData3 +		S3-1
	24		RxData3 -		S3-6
	25	RxClock3	RxClock3 +		S3-2
	10		RxClock3 -		S3-7
	39	GND	GND	GND	S3-3
S4	12	TxData4	TxData4 +	Data4 +	S4-5
	27		TxData4 -	Data4 -	S4-9
	30	TxClock4	TxClock4 +	Clock4 +	S4-4
	15		TxClock4 -	Clock4 -	S4-8
	13	RxData4	RxData4 +		S4-1
	28		RxData4 -		S4-6
	29	RxClock4	RxClock4 +		S4-2
	14		RxClock4 -		S4-7
	43	GND	GND	GND	S4-3

3 Building Configuration Environment

3.1 Get the Configuration Management Software yacer-DMS

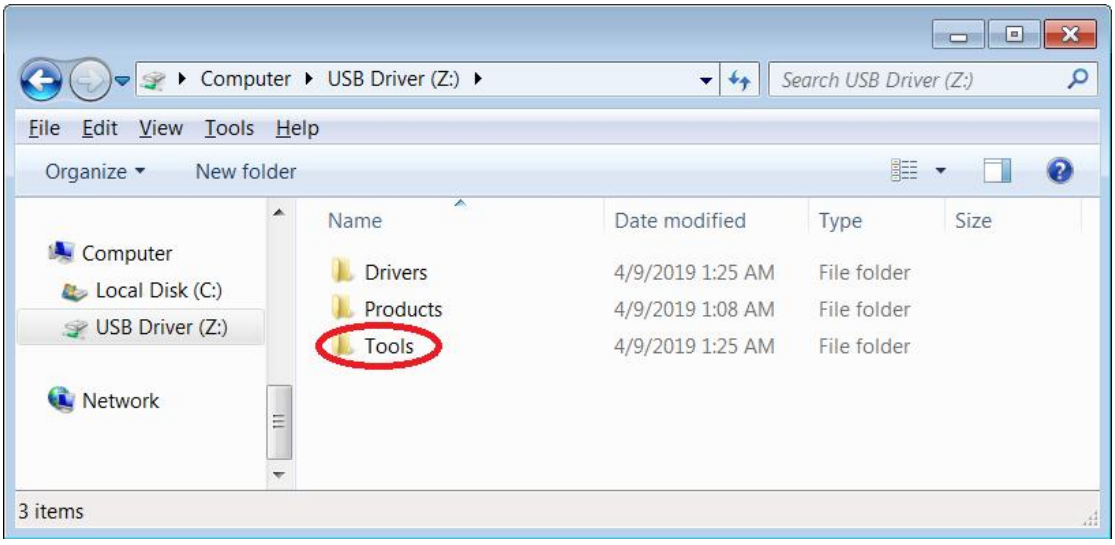
3.1.1 Official Website of Yacer

Visit the official website of yacer http://www.yacer.com.cn enter the “Tools” channel, and open the yacer-DMS software page to download the latest version of the software.



3.1.2 Accompanied USB disk

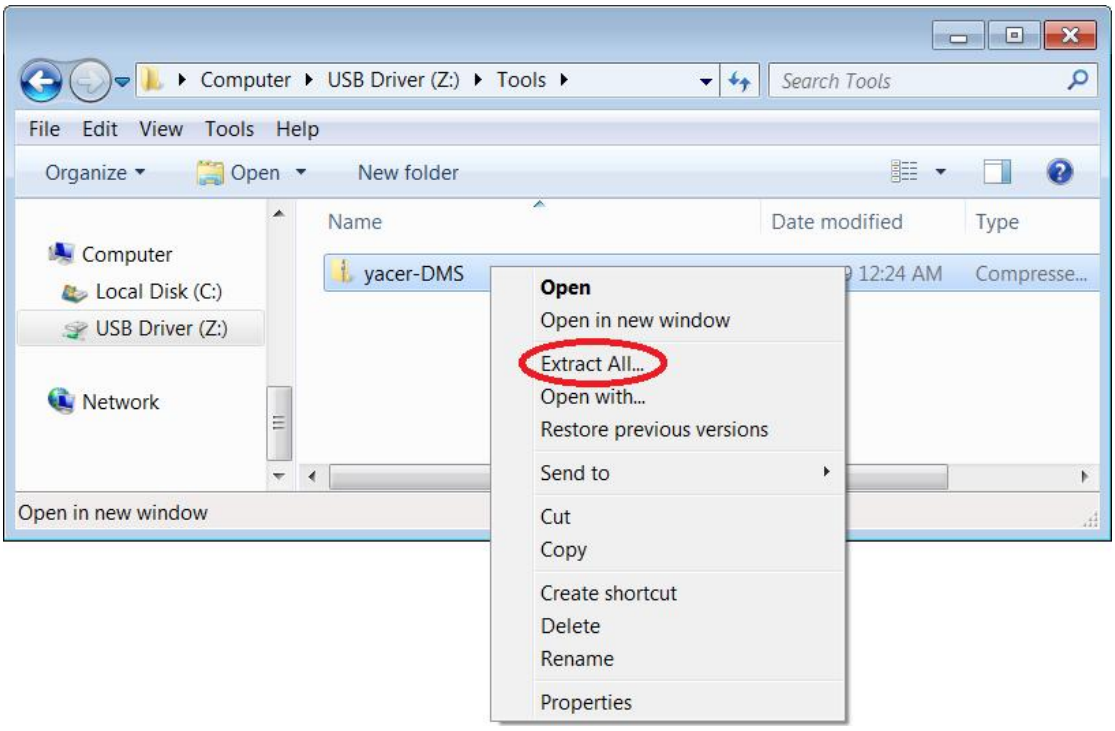
Insert the accompanied USB disk to the PC, open it and double-click to enter the “Tools” folder.



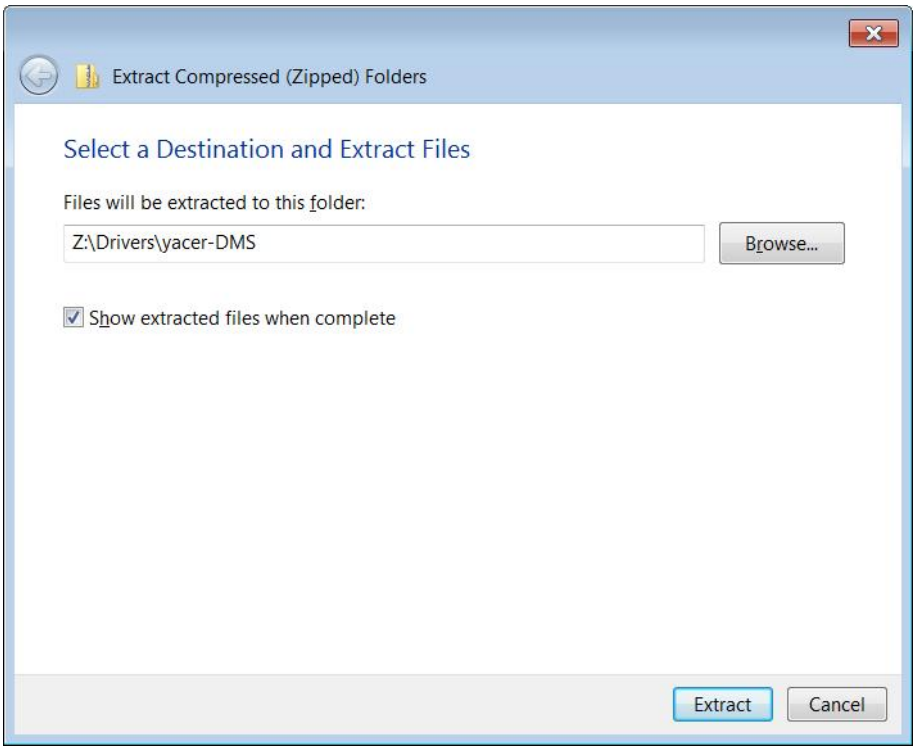
3.2 Run yacer-DMS

3.2.1 Unzip the file

- After getting the yacer-DMS.zip compressed file, first unzip it.

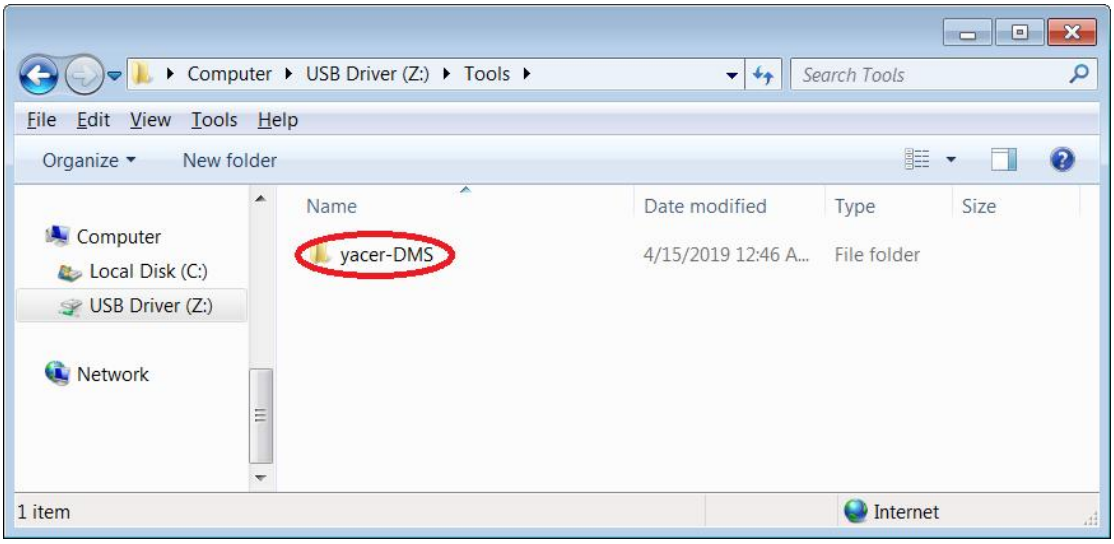


- With the Extract Compressed Folder dialog box popping up, set the path for saving extracted files in the box and click on the “Extract” button once set up.

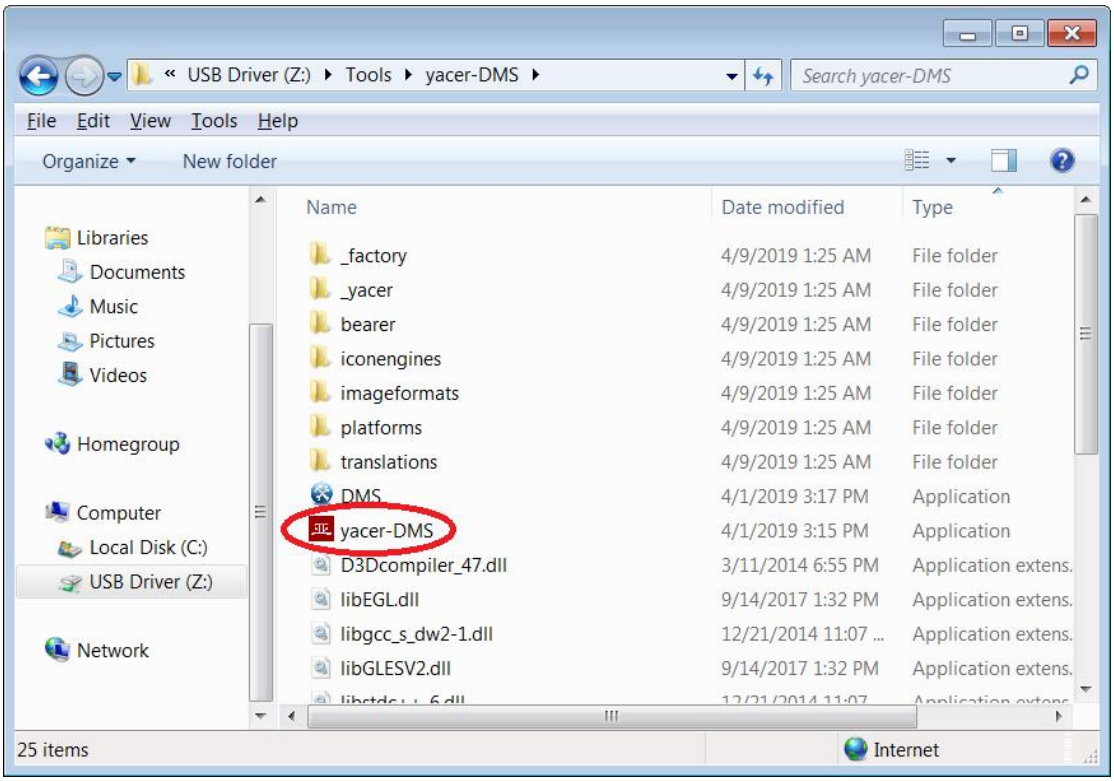


3.2.2 Run yacer-DMS

- Locate the decompressed folder according to the path for extracting the compressed files and click on it.



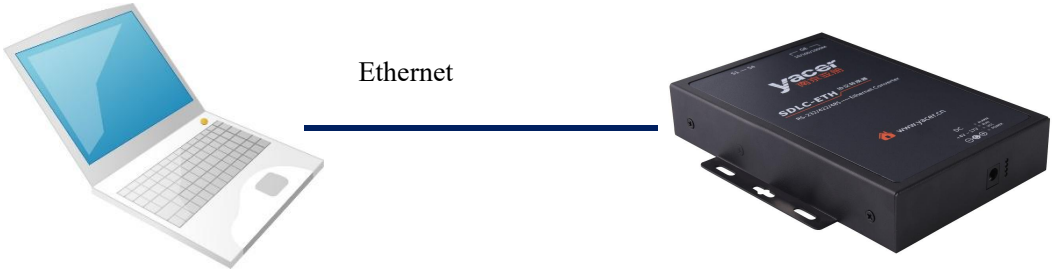
- Locate the yacer-DMS.exe file under directory and double-click on this file to run the configuration software.



3.3 Configure over the Ethernet Interface

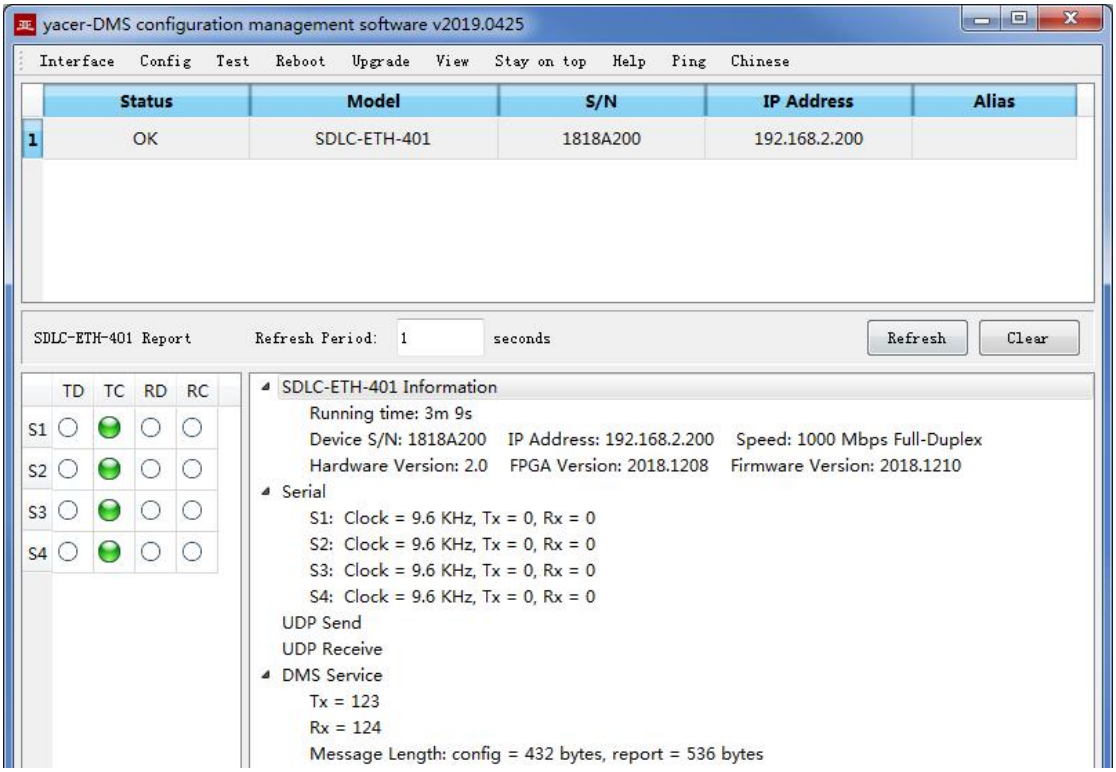
3.3.1 Connect SDLC-ETH with computer

Connect the Ethernet port of SDLC-ETH to the management computer over the ethernet cable.



3.3.2 Run yacer-DMS for configuration management

It's unnecessary knowing the SDLC-ETH's IP address or specifically configuring the IP address of computer, yacer-DMS can automatically discover the connected SDLC-ETH and perform status monitoring and parameter configuration by simply running the yacer-DMS configuration management software on the computer and keeping the SDLC-ETH powered on.

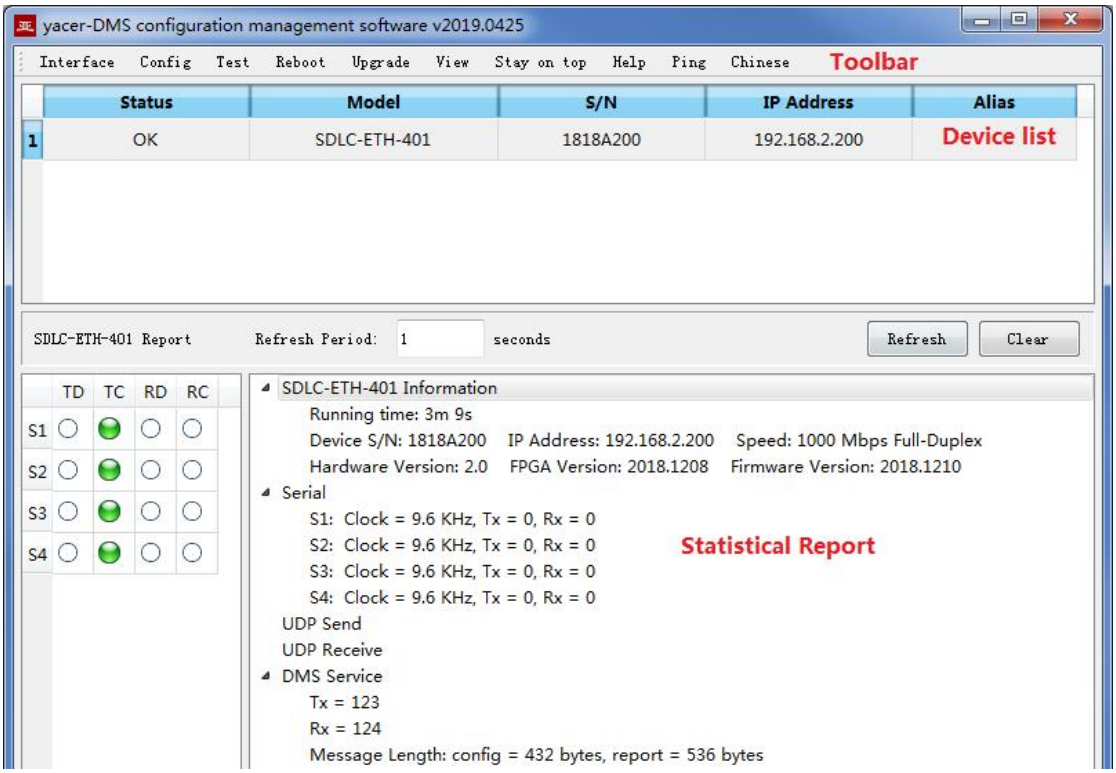


3.4 Main Window of yacer-DMS Software

3.4.1 Main window description

Below is the main window of the configuration and management software yacer-DMS, including three parts:

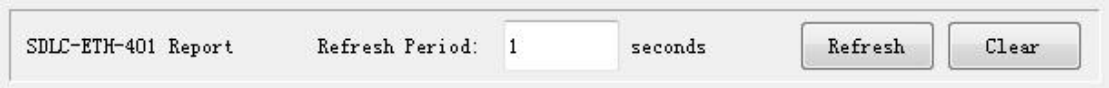
- Toolbar: Function operation buttons;
- Device List: Displaying the basic information and running status of the on-line device;
- Statistical Report: Displaying the receive/transmit indication & statistics, and device details of the specified device.



3.4.2 Statistical Report

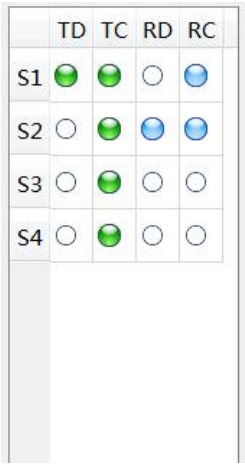
The statistical report has three panels: control panel, receive/transit indication panel, information panel.

3.4.2.1 Control Panel



Control Widget	Function
Refresh Period: 1 seconds	Set the refresh period of report
Refresh	Manually refresh the statistical report
Clear	Clear the statistical report

3.4.2.2 Receive/transmit indication

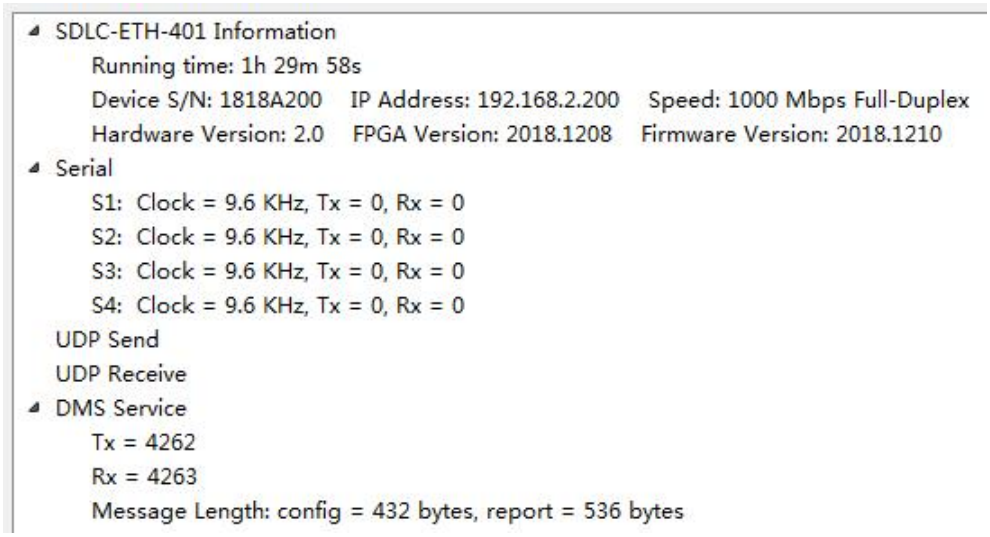


- Transmit: Each time the corresponding serial port transmits one frame data, the transmit indicator flashes once.
- Receive: Each time the corresponding serial port receives one frame data, the receive indicator flashes once.
- TD: Transmit Data Line Signal Indication
- TC: Transmit Clock Line Signal Indication
- RD: Receive Data Line Signal Indication
- RC: Receive Clock Line Signal Indication

3.4.2.3 Information Panel

The information panel is located on the right side of the statistical report, showing the following contents:

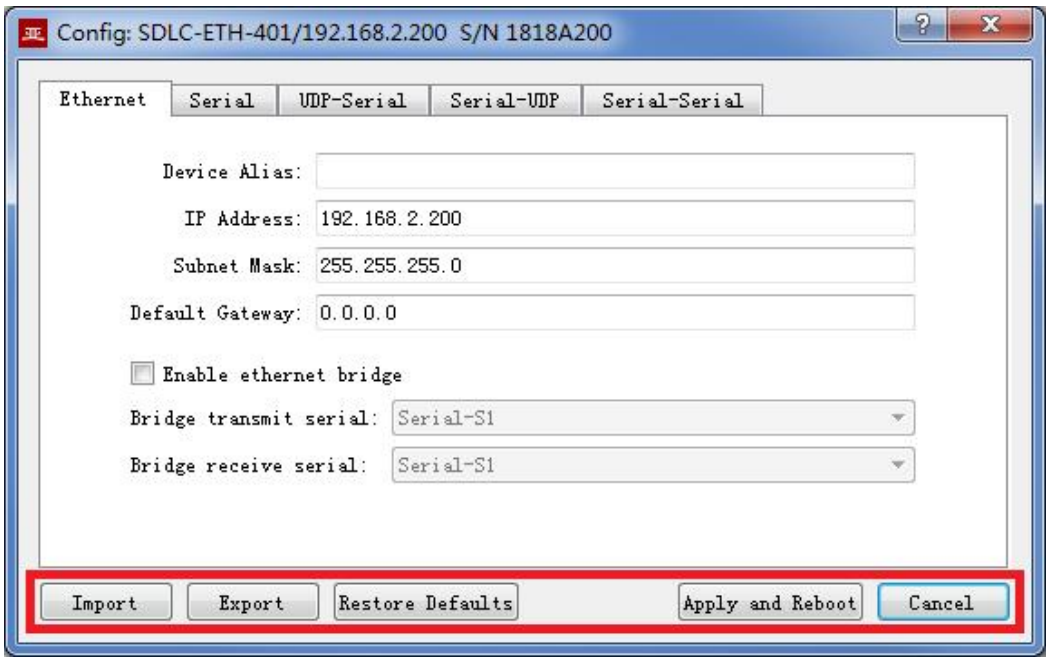
- Device Information: Running time, serial number, IP address, version number
- Serial Port: Receive/transmit statistics of all serial ports
- UDP Transmit: Displaying the relevant transmitted packets of the UDP Client for each enabled serial port to UDP entry
- UDP Receive: Displaying the relevant received packets of the UDP Server for each enabled UDP to serial port entry
- DMS Service: Displaying the information receive/transmit statistics of the configuration management between the device and the configured management computer



3.4.3 Configure Device

Click on the Config button on the toolbar or double-click on the specified device in the device list; yacer-DMS pops up the configuration dialog.

According to the interface and function, the dialog divides the configuration item into several configuration pages.



The following operation buttons are located at the bottom of the dialog:

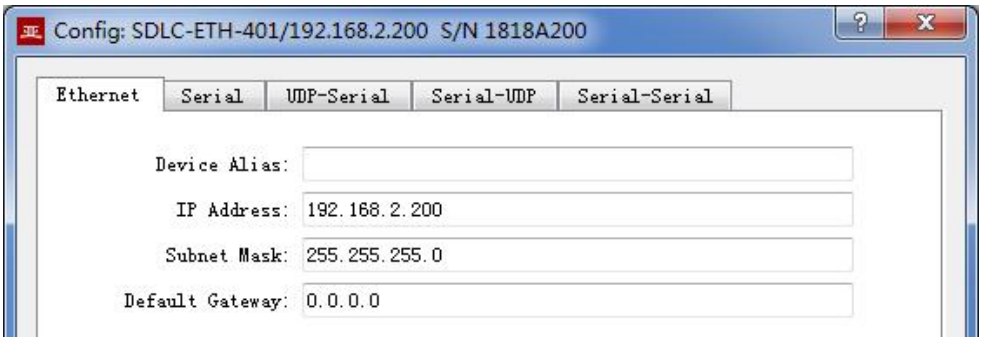
Buttons	Fuctions
Import	Open the configuration file and read the refreshed configuration dialog content
Export	Save the configuration content of the current dialog to the file
Restore Defaults	Refresh the dialog content with the device's default factory configuration
Apply and Reboot	Write the configuration content of the dialog into the device and restart the device to bring the configuration into effect
Cancel	Cancel the current configuration operation

4 Function and Configuration

4.1 Ethernet Interface

4.1.1 Device alias

It allows users to set an alias for the SDLC-ETH, thus adding description to the device or helping to remember the identification.

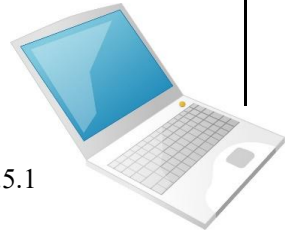


4.1.2 Default Gateway

By default, the default gateway is 0.0.0.0, representing that there is no gateway configuration.

If SDLC-ETH needs to communicate with the host on other subnet, it must rely on an external router. At this time, the SDLC-ETH's IP address must be on the same subnet with the IP address of the connected router port. Meanwhile, the IP address of router is set to the default gateway.

As shown below, the IP address of SDLC-ETH and remote PC is 192.168.2.200 and 192.168.5.100 respectively. As they do not belong to the same subnet, they must rely on the router for communication. SDLC-ETH and PC need to set the IP address of the connected router port to the default gateway of this device.



192.168.2.1

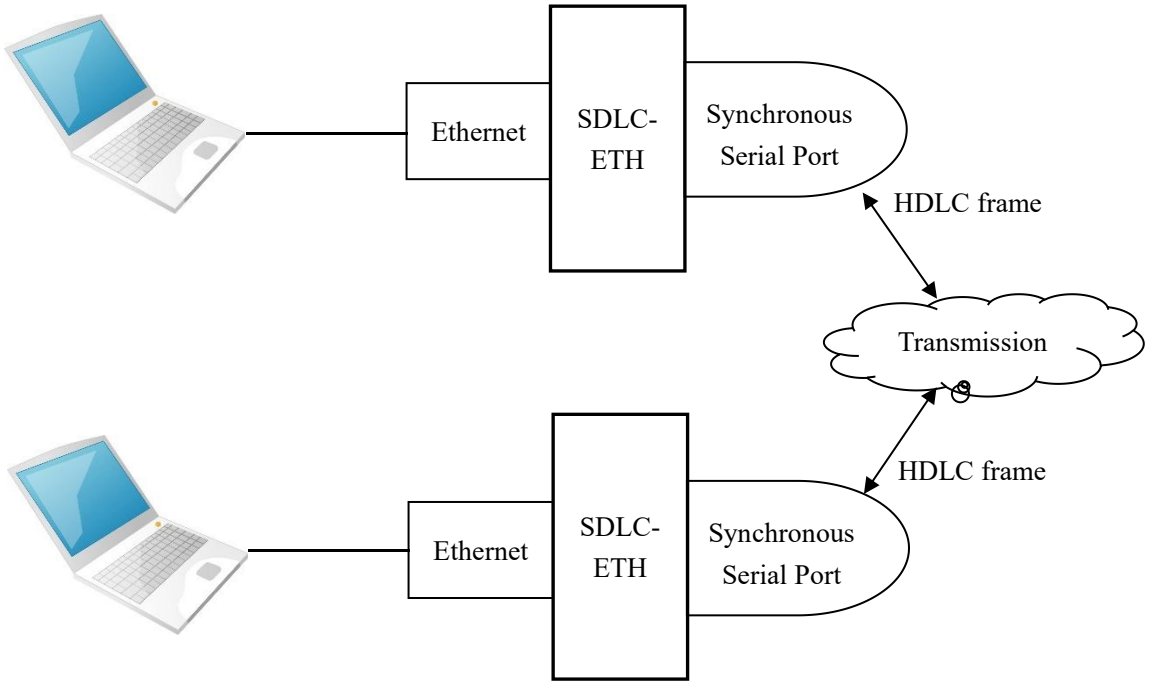
192.168.5.1

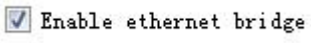
IP address: 192.168.2.200
Default gateway: 192.168.2.1

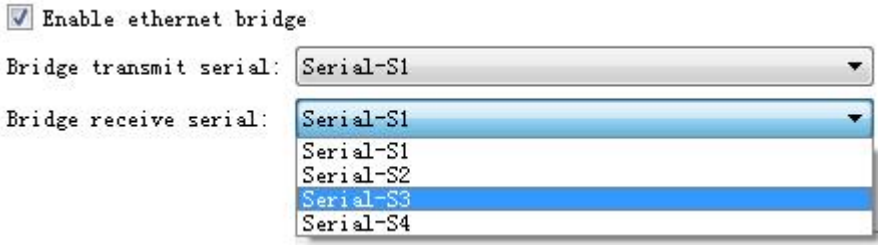
IP address: 192.168.5.100
Default gateway: 192.168.5.1

4.1.3 Ethernet Bridgeing enable

A pair of SDLC-ETH is used to build the Ethernet bridges, and the remote extension of Ethernet is realized by other transmission equipment via serial port.



By default, check the  checkbox to enable the Ethernet bridging function, providing the function to achieve the transparent Ethernet bridges by means of sending/receiving the serial port via bridging.



When the Ethernet bridging function is enabled, the following functions are disabled:

- IP address of the machine: SDLC-ETH acts as the bridging device without the IP address itself any more
- UDP to Serial Conversion
- Serial to UDP Conversion

4.2 Serial Port (S1 ~ S4)

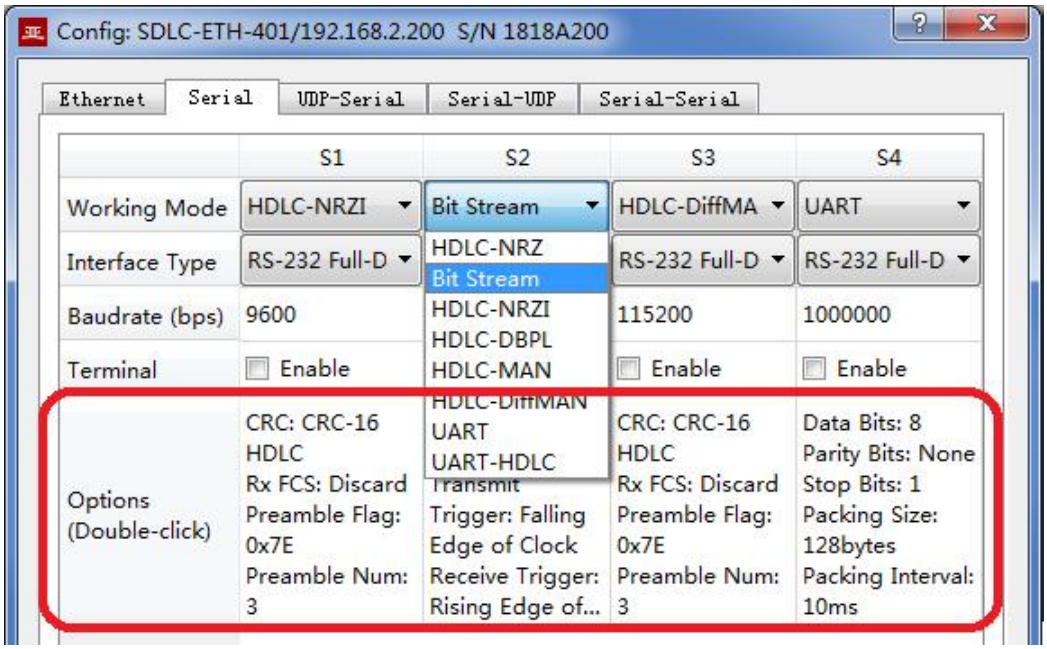
4.2.1 Working mode of the serial port

All serial ports are synchronous and asynchronous, with support for the synchronous and asynchronous working modes described in the table below.

Working Mode		Description
Synchronous	HDLC-NRZ	Synchronous HDLC protocol based on the NRZ encoding
	Bit Stream	Serial Bit data based on the receive clock sampling
	HDLC-NRZI	Synchronous HDLC protocol based on the NRZI encoding
	HDLC-DBPL	Synchronous HDLC protocol based on the DBPL (Differential Bi-Phase-Level) encoding
	HDLC-MAN	Synchronous HDLC protocol based on the Manchester encoding
	HDLC-DiffMAN	Synchronous HDLC protocol based on the differential Manchester encoding formats
Asynchronsous	UART	Universal asynchronous serial, equivalent to the serial port on the common computer
	UART-HDLC	UART-based similar HDLC communication protocol

Users can select the desired working mode from the “working mode” combobox. Due to different parameter configuration of each working mode, contents of the “Options” cell will be adjusted automatically according to the determined working mode.

If further configuration of working parameters of the selected working mode is required, mouse double-click on the “Options” cell to pop up the parameter configuration dialog.



4.2.2 Interface type selection

The physical interface type of serial port can be selected as RS-232, RS-422 or RS-485 by "interface type".

4.2.3 Baudrate configuration

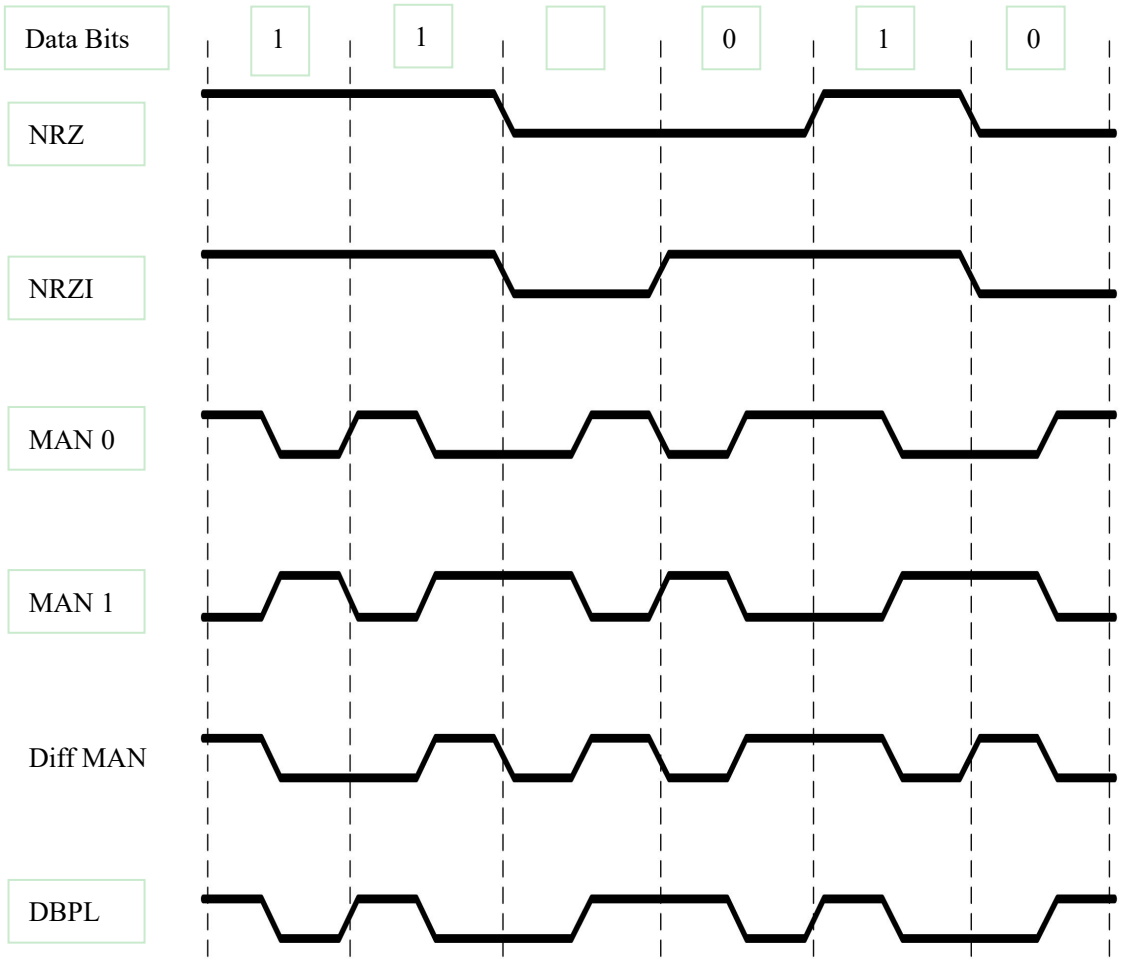
The "baud rate" configures the communication rate of the serial port. For the synchronous working modes such as HDLC-NRZI, HDLC-DBPL, HDLC-MAN, HDLC-DiffMAN and all asynchronous working modes, the baud rate of both sides of the communication must be the same in order to ensure the correct transmission of data.

4.2.4 Terminal configuration

"Terminal" is meaningful only when the interface type is RS-485 half-duplex. In generally, the terminal as a node of RS485 bus needs to enable the “terminal” function. When "terminal" is enabled, SDLC-ETH provides 120 ohms matching.

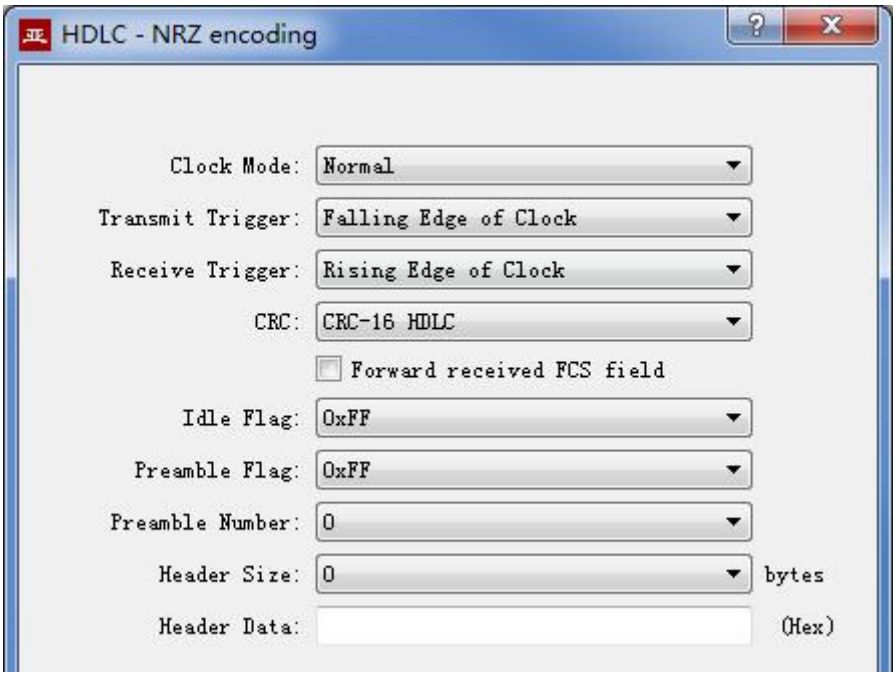
4.2.5 Encoding format of the synchronous serial port

For HDLC-NRZ, HDLC-NRZI, HDLC-DBPL, HDLC-MAN, HDLC-DiffMAN and other synchronous working modes, the link layer adopts the HDLC protocol with the encoding format difference as follows:

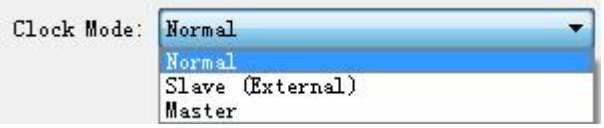


4.2.6 HDLC-NRZ Option

HDLC-NRZ is the common synchronous working mode. This encoding format relies on receiving and receiving clock signals to achieve data bit synchronization, so the configuration of clock parameters is particularly important.



4.2.6.1 Clock mode



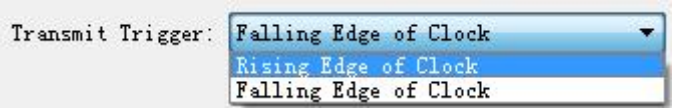
There are three clock modes for the synchronous serial port, normal, slave clock &master clock.

Clock Mode	Transmit Clock	Receive Clock
Normal	Generation from the local device, output through pin TxC.	Generation from the peer device, input through pin RxC.
Slave Clock (External)	Generation from the peer device, input through pin RxC. TxC output synchronizes with RxC automatically.	Generation from the peer device, Input through pin RxC.
Master Clock	Generation from the local device, output through pin TxC.	Generation from the local device, ignoring the clock of pin RxC.

The slave clock mode is also called as the external clock working mode. When the peer device is the DCE, SDLC-ETH is often configured as the slave clock mode and transmits data

with the clock provided by the DCE, ensuring the data transmission across the whole network based a clock and avoiding packet loss concerns caused due to different clock sources.

4.2.6.2 Transmit trigger

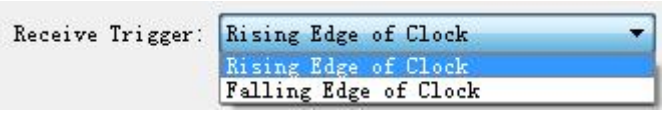


Transmit trigger defines the generation clock edge of the new data bit:

- Falling edge of clock: A new data bit is generated on the falling edge of clock
- Rising edge of clock: A new data bit is generated on the rising edge of clock

During the standard-compliant communication, falling edge of clock is generally chosen to trigger the new data transmission. In some custom development applications, sometimes users will adopt the rising edge to trigger the new data transmission.

4.2.6.3 Receive trigger



Receive trigger defines the sampling clock edge of the serial port receive data:

- Rising edge of clock: Data on the RxD line is read on the rising edge of the RxC signal
- Falling edge of clock: Data on the RxD line is read on the falling edge of the RxC signal

During the standard-compliant communication, receive trigger must be configured as the rising edge of clock to ensure correct reading of data as it takes time to stabilize the new data that is triggered with the falling edge.

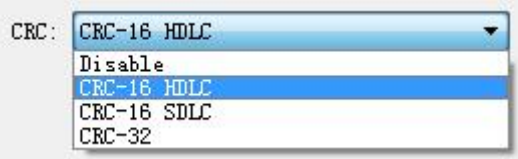
The local receive trigger configuration is determined according to the transmit trigger of the remote terminal device:

Remote Transmit Trigger	Local Receive Trigger
Falling edge of clock	Rising edge of clock
Rising edge of clock	Falling edge of clock

4.2.6.4 CRC

In order to verify the correctness of data communication, CRC function should be enabled.

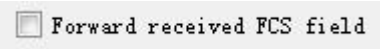
By default, configure the protocol CRC check type with CRC-16-HDLC as the most commonly used type for the HDLC protocol communication.



CRC	Description
Disable	CRC disable: ● No CRC calculation for data transmission or FCS field for HDLC frame ● No CRC check for data receiving
CRC-16 HDLC	Adopt the 16-bit IBM HDLC CRC check method
CRC-16 SDLC	Adopt the 16-bit IBM HDLC CRC check method
CRC-32	Adopt the 32-bit ISO HDLC CRC check method

4.2.6.5 Forward received FCS field

This configuration is only effective with CRC enable.



The HDLC frame structure is shown in the following table, where FCS is the frame check sequence field.

Opening Flag	Address Field	Control Field	Information Field	FCS Field	Closing Flag
0x7E	1 byte	1 byte	Variable length	CRC 2/4 bytes	0x7E
0x7E	User data			CRC 2/4 bytes	0x7E

If this option is checked, then forward the user data and FCS field.

If this option is not checked, SDLC-ETH will discard the 2/4-byte FCS field at the end of data and only forward the user data after the receive HDLC frame check is passed.

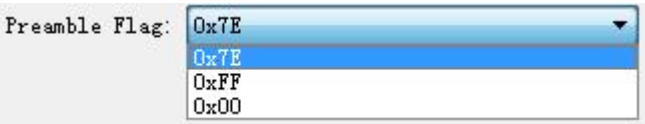
4.2.6.6 Idle Flag

The definition of HDLC inter frame filling content, the default should be 0xFF.

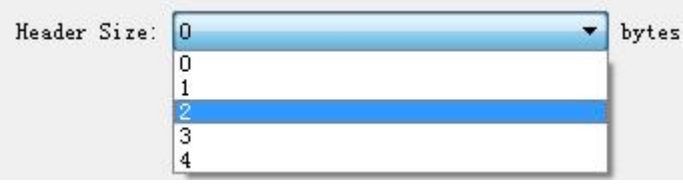
4.2.6.7 Preamble flag and number

During the half-duplex communication, a preamble flag is often required in front of the frame for receiving party synchronization, and the most commonly used method is to add 2~5 0x7E.

For full duplex applications, the Preamble number is often unrequired,set it to 0(no preamble).



4.2.6.8 Header size and data



As shown above, the header size is defined as 2, and the header data is defined as FF 03.

- While sending HDLC, the FF 03 is added before the user data, and HDLC frame data is composed with user data.
- When receiving HDLC, SDLC-ETH discards the first two bytes of HDLC frame data as the frame header, and only forwards the subsequent data to the user..

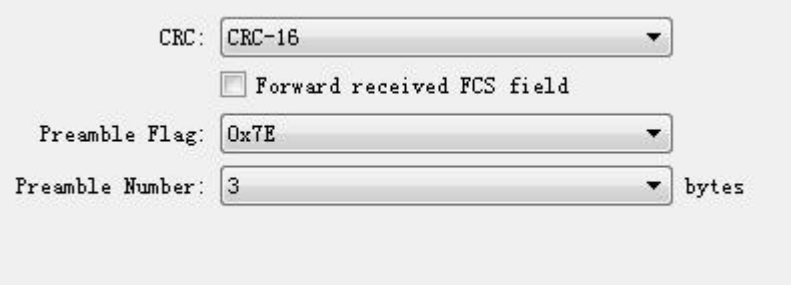
Opening Flag	Frame Header	User Data	FCS Field	Closing Flag
0x7E	0xFF 0x03	Variable length	CRC 2/4byte	0x7E

4.2.7 HDLC-NRZI/DBPL/Diff-MAN Option

Unlike the NRZ encoding format, the NRZI, DBPL, MAN and Diff-MAN encoding format data contains clock information, which only requires that the baud rate of the both communication sides should be the same, instead of the clock mode, transmit trigger, receive trigger and other parameters.

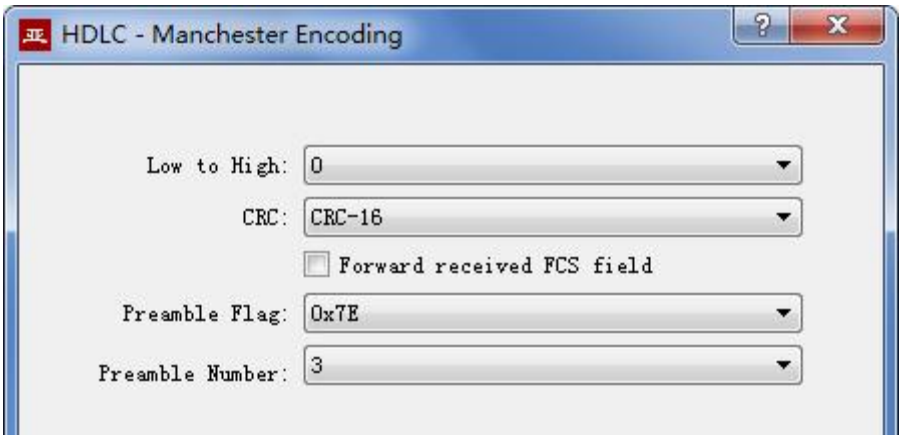
Refer to the HDLC-NRZ parameter configuration for configuration of CRC, FCS, Preamble Flag and Preamble number.

The option dialog of the HDLC-NRZI/DBPL/Diff-MAN working modes is shown as follows:



4.2.8 HDLC-MAN (Manchester) Option

The advanced option dialog of the HDLC-MAN working mode is shown as follows:



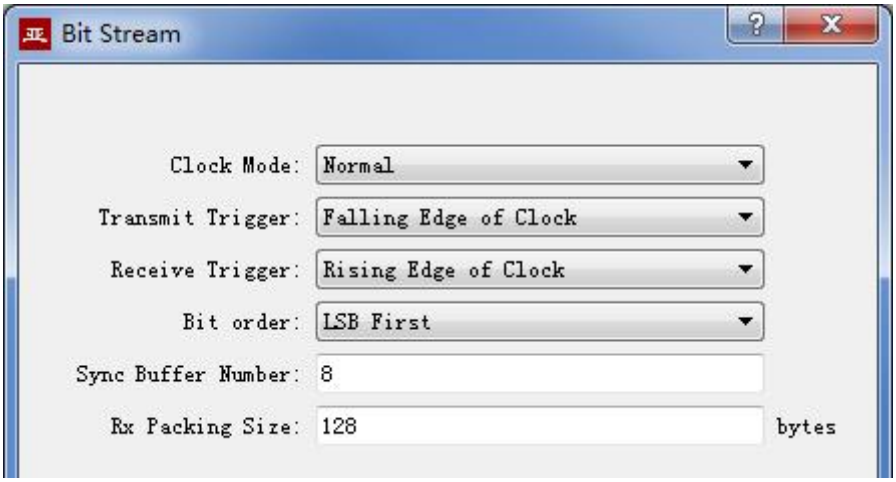
Except for the configuration parameters same as NRZI, parameters with the data line low-to-high transition definition are included for the Manchester encoding format:

- 0: Low-to-high transition indicates the logic 0;
- 1: Low-to-high transition indicates the logic 1;

Refer to the HDLC-NRZ parameter configuration for configuration of CRC, FCS, Preamble Flag and Preamble number.

4.2.9 Bit Stream Option

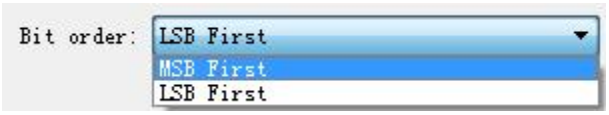
The rising or falling edge of each clock cycle samples the 1bit data on the data line, which forms a UDP message and sends to the destination IP after receiving a byte with the packet length by forming a byte with each 8bit.



Refer to the HDLC-NRZ parameter configuration for configuration of clock mode, transmit trigger and receive trigger.

4.2.9.1 Bit order

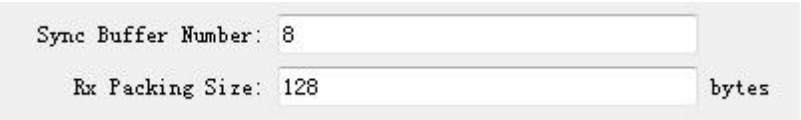
The online bit stream is stored in the computer or system memory in the form of byte.The receive/transmit sequence determines the conversion mode of byte and bit.



Receive/Transmit Sequence	Transmit Operation	Receive Operation
MSB first	First transmit the high-bit byte	Data received first is placed on the byte high bit
LSB first	First transmit the low-bit byte	Data received first is placed on the byte low bit

4.2.9.2 Sync buffer Number and Rx Packing Size

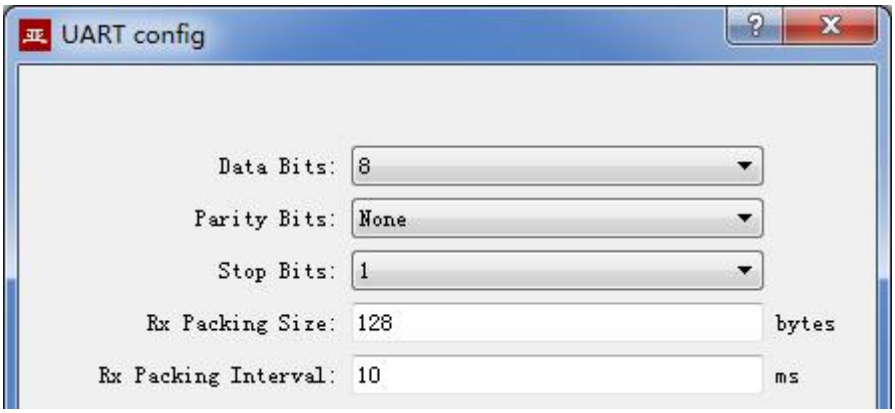
The Sync buffer number is the number of buffers sent by a synchronous Bit stream when sending data. A certain number of packets are cached in advance, and then the sending of Bit stream is started. It maintains the coherence of Bit stream data.



4.2.10 UART Option

UART is a means of character stream communication; data bits, parity bits and stop bits define the basic working parameters of the asynchronous serial port, which must be configured identically to the opposite terminal device.

Generally, data bits are defined as 8, i.e. one byte, and UART corresponds to the byte stream communication.



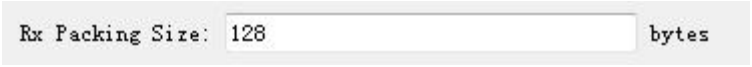
When the UART byte stream is converted into a UDP message or HDLC frame, it is too costly and inefficient if each byte is converted to a UDP message for transmission.

To improve efficiency, SDLC-ETH buffers the received byte stream and forms a number of buffered bytes into a UDP message to send, of which this process is called as packet.

Packets are controlled with two parameters, namely the packet length and the packet interval.

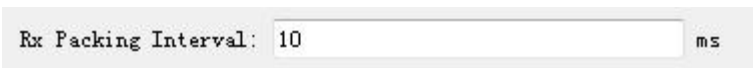
4.2.10.1 Rx packeting size

For example, if the packet length is set to 128 bytes, then it will form a UDP message to send after UART receives the full 128 bytes.



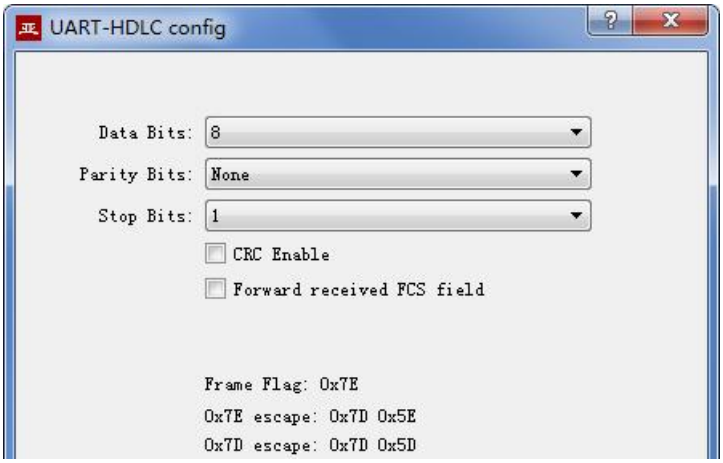
4.2.10.2 Rx Packet interval

As shown in the example above, the packet interval 10ms is set, and if no new byte data is received over 10ms, it will form the buffer data as a packet to forward no matter whether it has received the full 128 bytes.



4.2.11 UART-HDLC Option

The UART-HDLC working mode is a custom protocol by Yacer which form the asynchronous HDLC frame on the basis of the normal UART communication by packaging the byte stream. Therefore, the asynchronous serial port can perform the packet-based communication with the UDP message and synchronous HDLC frame.



The UART-HDLC frame format adds 0x7E before and after the packet as the opening flag and closing flag with the frame structure as follows:

opening Flag	Information Field	FCS Field	closing Flag
0x7E	2~1470 bytes of data	2-byte CRC data	0x7E

As the information field and FCS field may appear 0x7E, perform the character escape on such fields before transmission with the escape rules as follows:

- 0x7E: Escaped to two characters, 0x7D 0x5E
- 0x7D: Escaped to two characters, 0x7D 0x5D
- Other characters: No escape

The escape operation of data transmit is as follows:

Original Data	Actual Transmit Data
0x7E	0x7D 0x5E
0x7D	0x7D 0x5D
Others	No change

The escape operation of data receive is as follows:

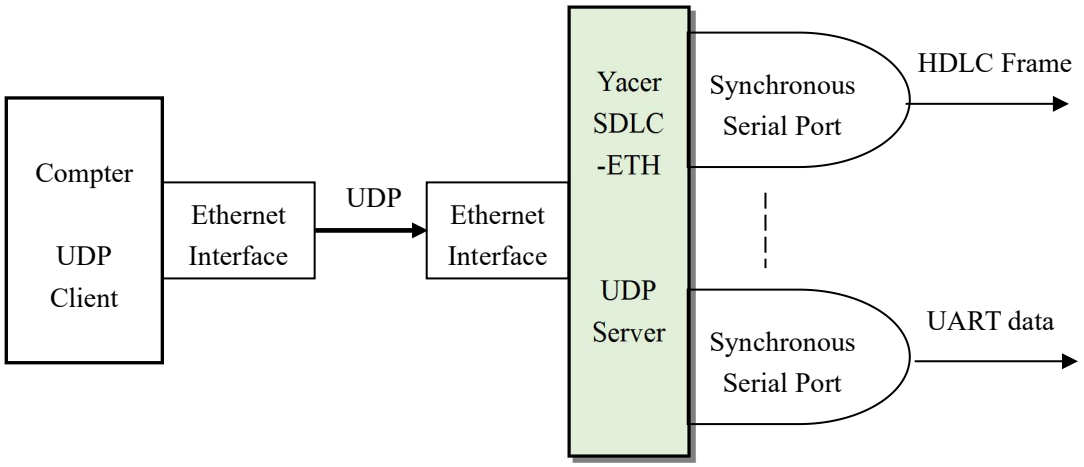
Actual Receive Data	Data
0x7D 0x5E	0x7E
0x7D 0x5D	0x7D
Others	No change

4.3 UDP to Serial Conversion

4.3.1 Application model

With SDLC-ETH, PC or server can realize the transmit function of the synchronous HDLC serial port.

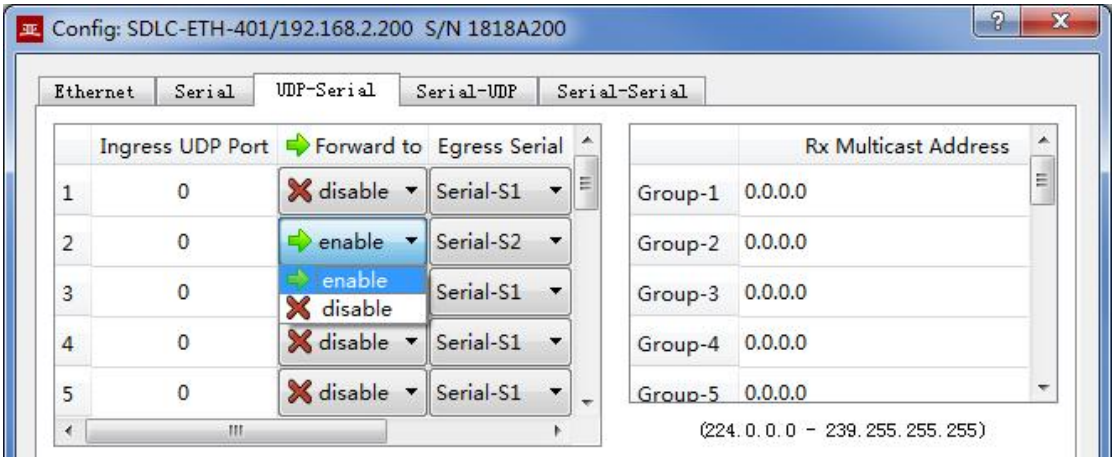
The typical application is shown as follows. PC sends a UDP message over the Ethernet interface as the UDP Client, and SDLC-ETH sends it out from the synchronous serial port after converting the received UDP message into the HDLC frame.



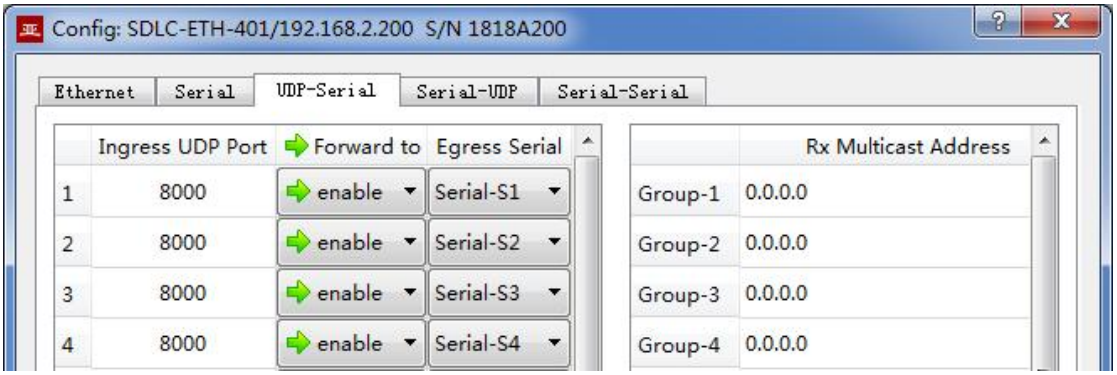
4.3.2 Forward Configuration

Set the UDP to serial port. Each row represents the forwarding entry from a UDP port to the serial port while “enable” is selected with three forwarding strategies to be achieved:

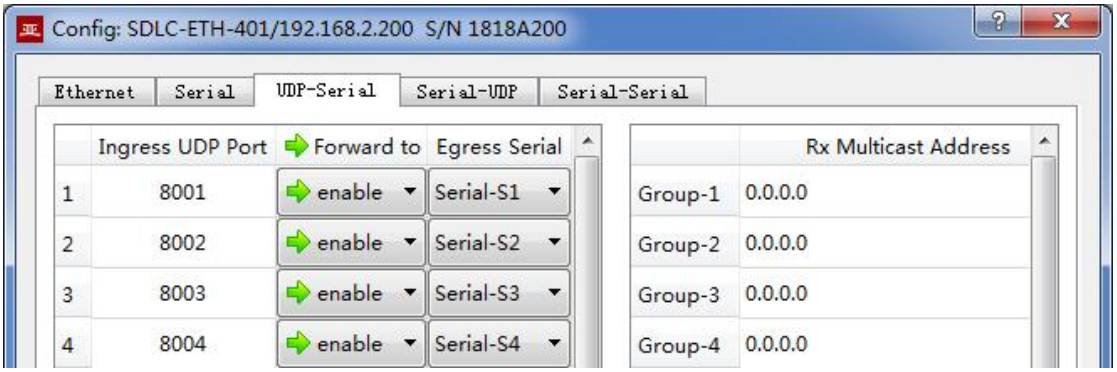
- Forwarding: Data received by the specified UDP port can be forwarded to the specified serial port.
- Multiplexer: Data received by several different UDP ports can be forwarded to the same serial port.
- Demultiplexer: Data received from the same UDP port can be forwarded to the different serial ports.



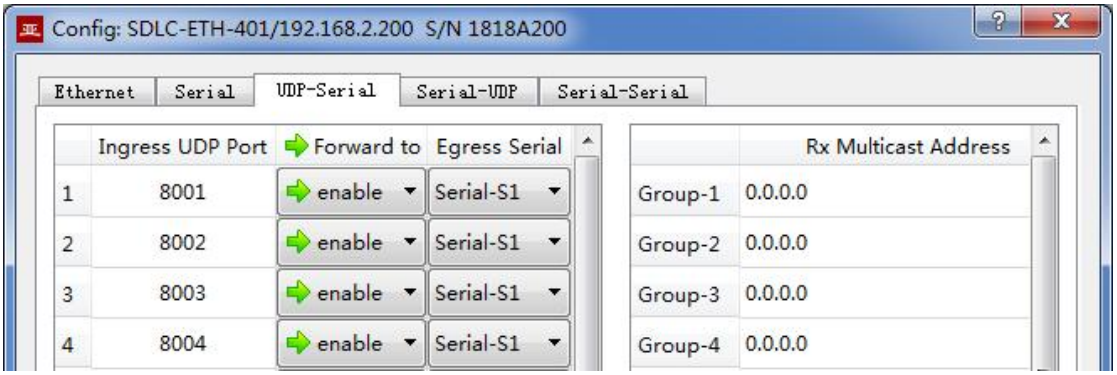
The following configuration realizes the application, where data received from 1 UDP and distributed to 4 serial ports:



The following configuration realizes the application, where data received from 4 UDP ports and distributed to 4 serial ports:



The following configuration realizes the application, where data received from 4 UDP ports and distributed to 1 serial port:



4.3.3 Receive multicast

If users need to receive the multicast UDP message, add the required multicast address from the right “Rx Multicast Address” list.

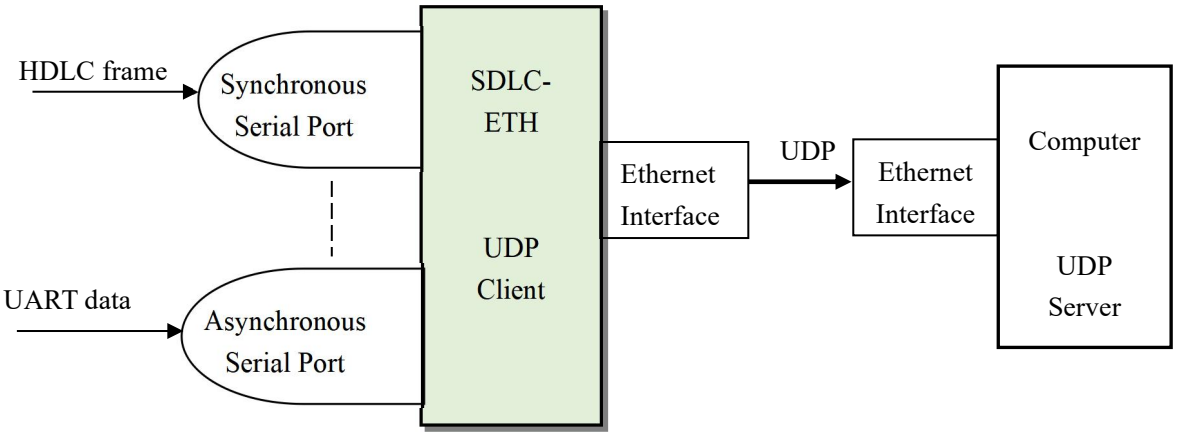
Range of the multicast address is 224.0.0.0 ~ 239.255.255.255, 224.8.8.8 is the configuration management address of the SDLC-ETH and users can’t use this address.

The multicast address configured as 0.0.0.0 indicates that the entry is not in effect.

4.4 Serial to UDP Conversion

4.4.1 Function description

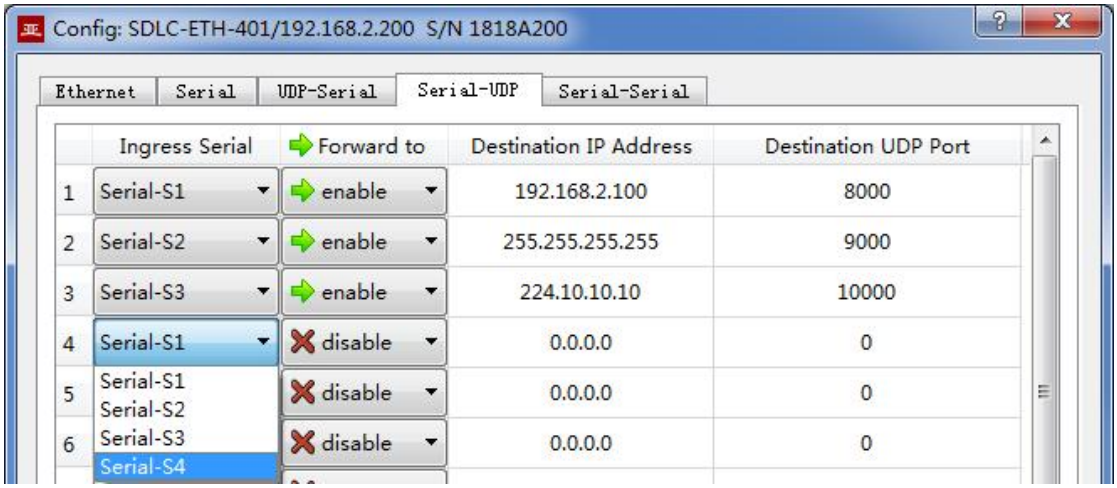
The HDLC-to-UDP function figures is shown below. SDLC-ETH receives the synchronous HDLC frame or asynchronous UART frame from other devices over the serial interface, converts it into the UDP message and sends to PC or server over Ethernet.



4.4.2 Forward Configuration

Set the serial port to UDP. Each row represents the forwarding entry from a serial port to the destination UDP port while “enable” is selected with three forwarding strategies to be achieved:

- Forwarding: Data received from the specified serial port can be forwarded to the UDP port of specifiedd estination IP.
- Multiplexer: Data received from several different serial ports can be forwarded to the same UDP port of the same destination IP.
- Demultiplexer: Data received from the same serial port can be forwarded to the different UDP ports of the different destination IP.



As shown above, three serial port to UDP entries are configured for achieving the following ems:

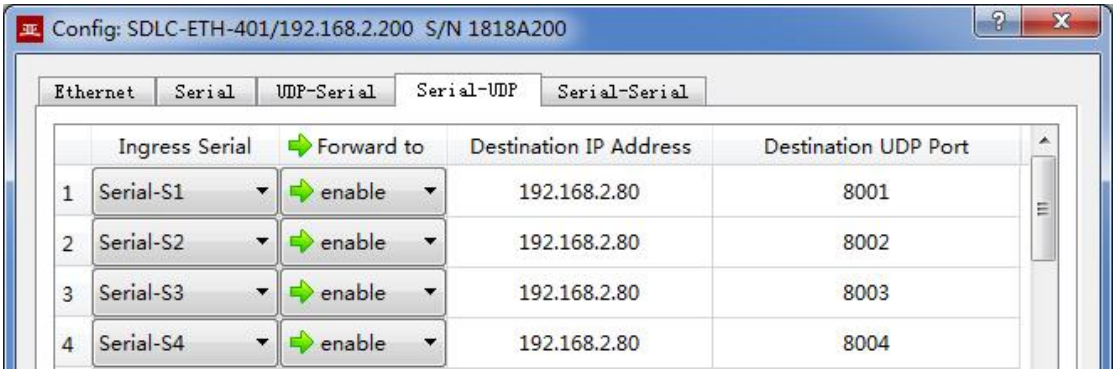
- Serial port S1 to UDP unicast, with the destination IP address as 192.168.2.80 and destination UDP port as 8000
- Serial port S2 to UDP broadcast, all hosts on the subnet can receive data from S2 at the 9000 port
- Serial port S3 to UDP multicast, only the PC joining Group 224.10.10.10 on the network can receive data from S3.

4.4.3 How does the UDP server identify the source serial

In multiplexer applications, the HDLC frame from several different serial ports needs to be forwarded to the same server or PC. In this case, a strategy enables server or PC to know which serial port the received UDP message data is from.

4.4.3.1 Distinguish the source serial port according to the source UDP port

As shown below, set different forwarding destination UDP ports for each serial port. The UDP Server PC receives data at the different UDP ports. Message received at port 8001 is from the serial port S1 while message received at port 8004 is from the serial port S4.

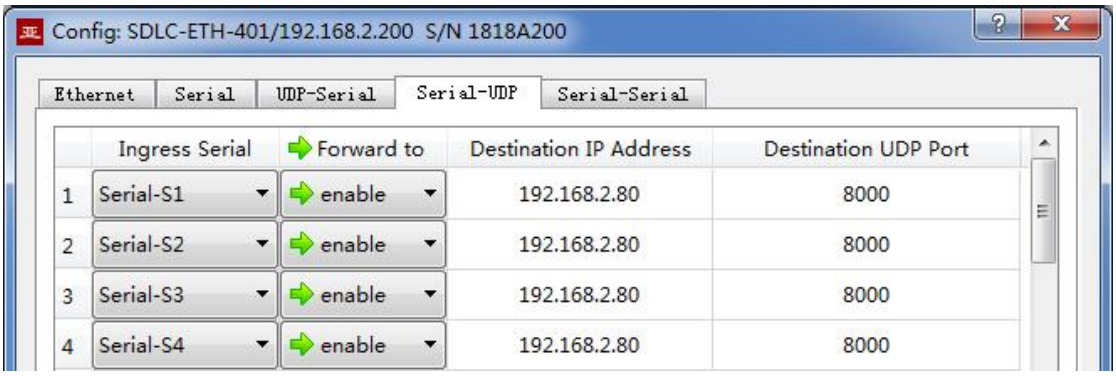


4.4.3.2 Distinguish the source serial port according to the source UDP port

When the source serial port is identified with the destination UDP port, UDP Server needs to listen and receive data on a plurality of UDP ports. In case there are many serial ports, not only the UDP Server port occupies too many resources, the configuration and programming complexity also increases significantly.

In order to simplify implementation of the UDP Server side, we can use the configuration example below, forwarding each conversion to the same port of the UDP Server. During yacer's SDLC-ETH forwarding, it will automatically adjust the source port number of the UDP message according to the source serial port. The source ports of the UDP message forwarded by the serial ports S1 and S2 are 8001 and 8002 respectively; the following is gradually increasing.

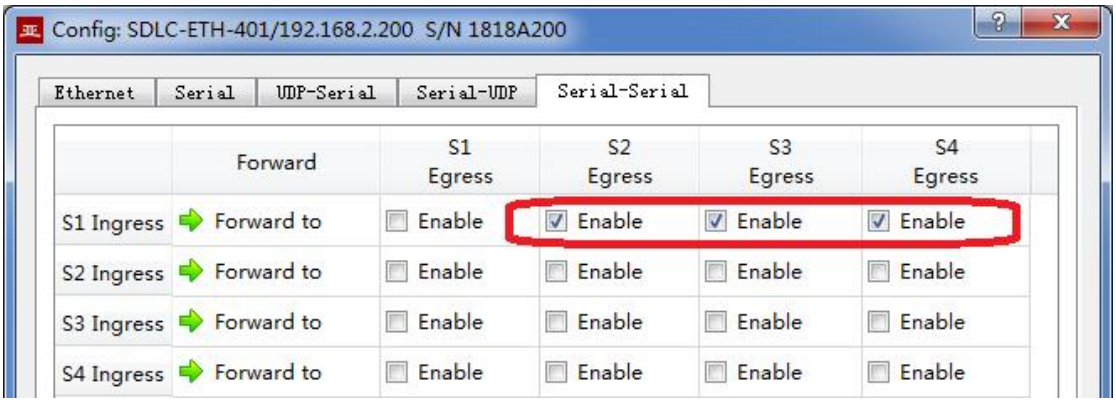
Thus, UDP Server only needs to listen and receive data at a port (8000 in the example below) and distinguishes the source serial port according to the source UDP port. If several SDLC-ETHs are provided, UDP Server can distinguish the source device via the source IP.



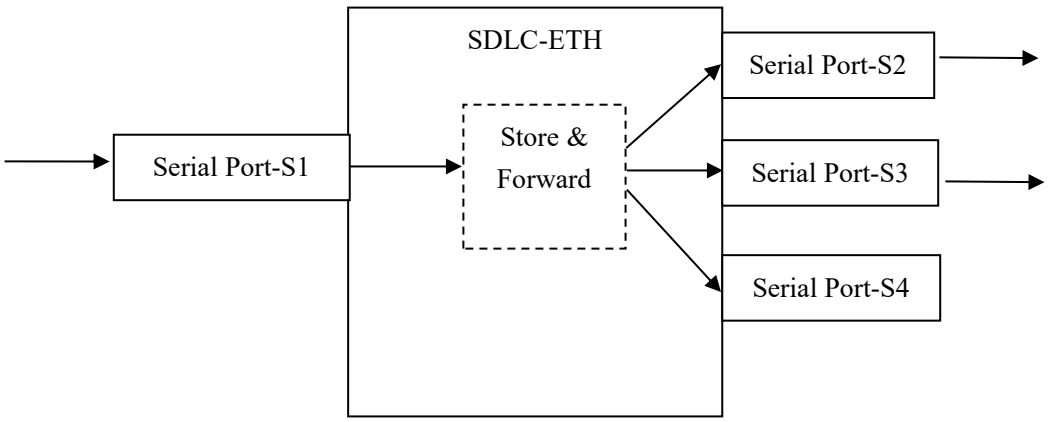
4.5 Serial Port to Serial Port

Serial-serial can forward the input data of the specified serial port to other serial port outputs, which is mainly used for:

- Conversion between synchronous and asynchronous serial ports
- Serial port Demultiplexer: Divide the single serial port data into multi-channel data. Unlike the common demultiplexer, demultiplexing is possible with SDLC-ETH, the different baud rates and clock modes can be set for each-channel serial port, avoiding the packet loss caused by clock inconsistency



Configuration shown above realizes the application of the input demultiplexing of the serial port S1 to S2, S3 and S4 outputs. SDLC-ETH performs storing and forwarding on the received data. Even if the baud rate and clock mode of S1, S2, S3 and S4 are different, packet will not loss.

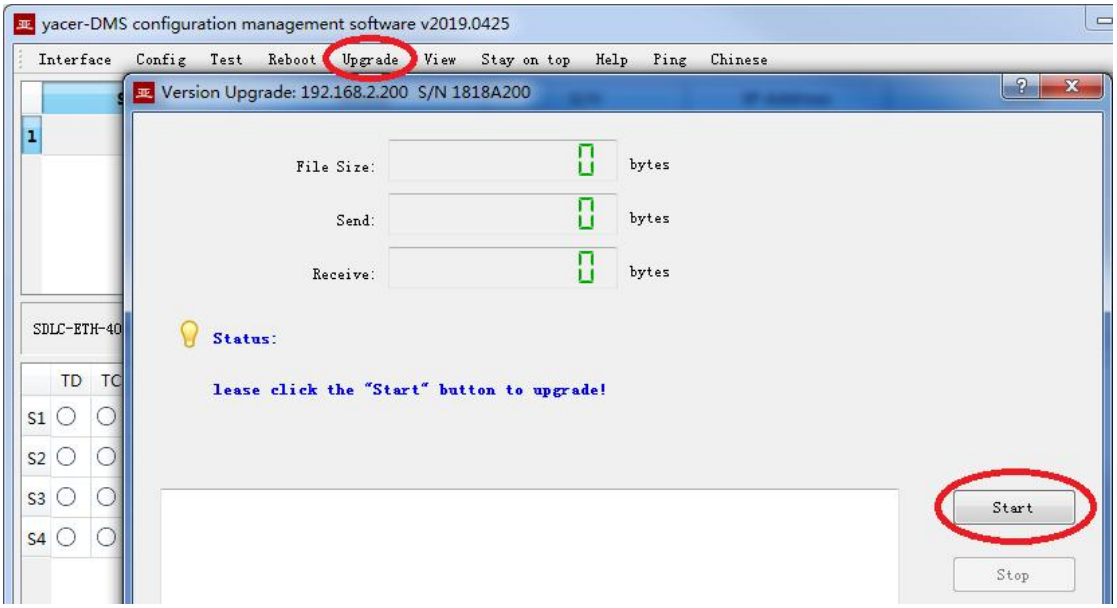


5 System Maintenance

5.1 Firmware Version Upgrade

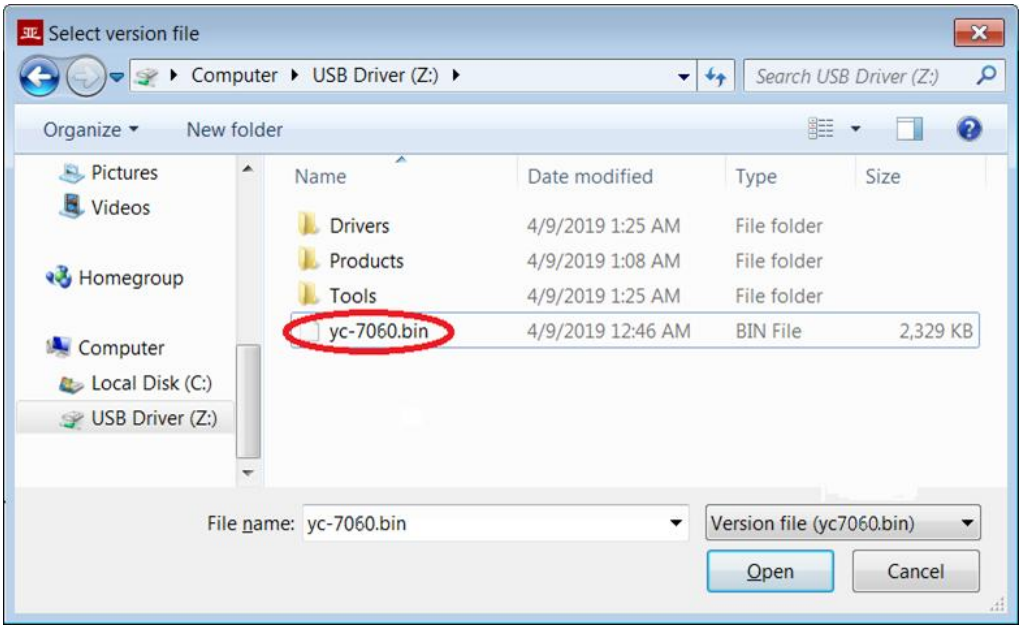
5.1.1 Start updating

Click on the Upgrade button on the toolbar to pop up the version upgrade dialog, and then click on the Start button.



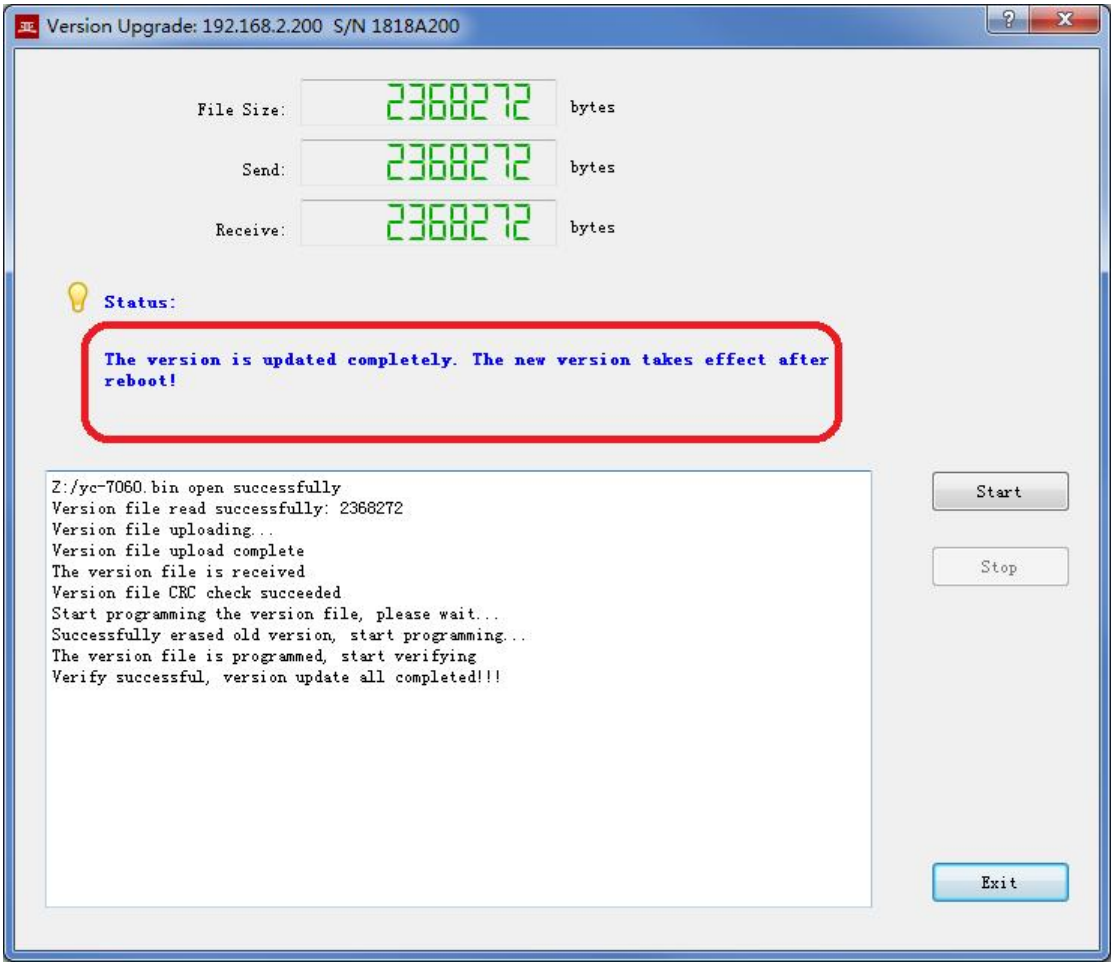
5.1.2 Locate Firmware Version

The “Selection Version File” dialog pops up. Locate the folder for storing the latest firmware version, select and click “Open” to start updating.



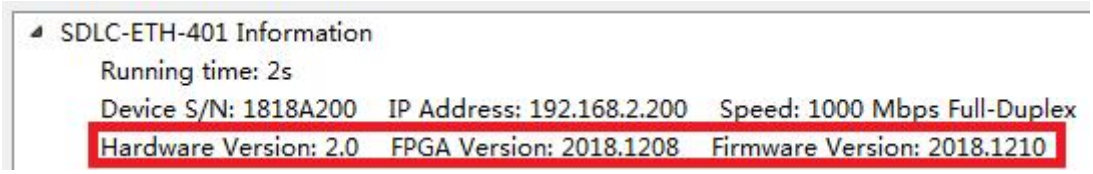
5.1.3 Upgrade Completed

After completion of upgrade, “Version Update Completed” displaying on the page indicates that the version update is completed.



5.1.4 Upgrade Confirmation

After completion of update, re power-on the device, observe the version information in the statistical report and determine whether the new version is updated successfully via the version date.



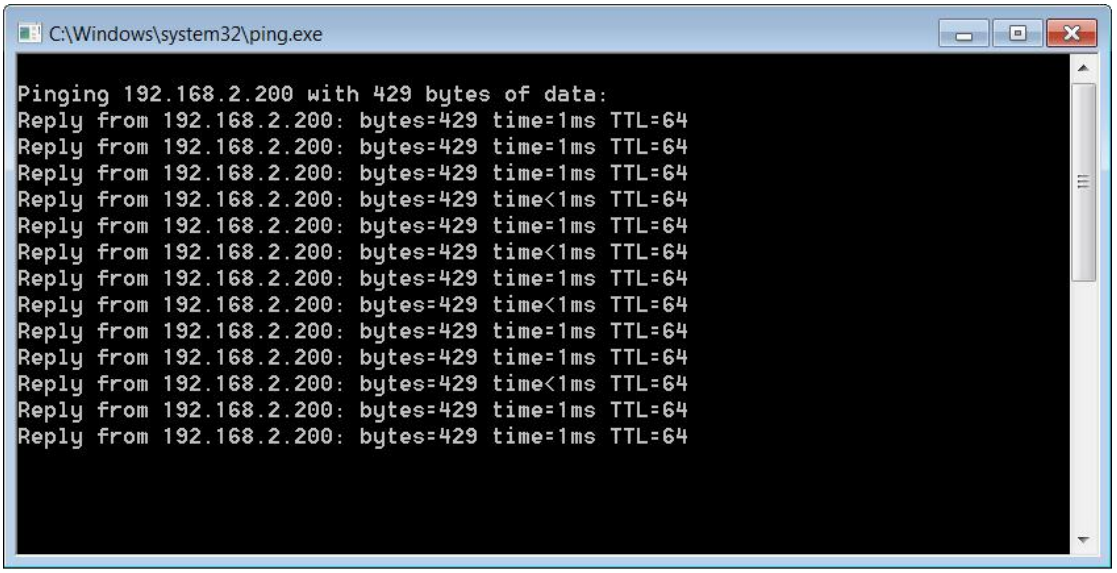
5.2 Device Reboot

Click on the Reboot button on the toolbar to pop up the device reboot dialog, and then click on the Reboot button to reboot the device.

5.3 Ping

By clicking on the Ping button on the toolbar, DMS will start the ping command automatically for the selected device so as to check whether the network connection between the configuration management computer and SDLC-ETH is normal.

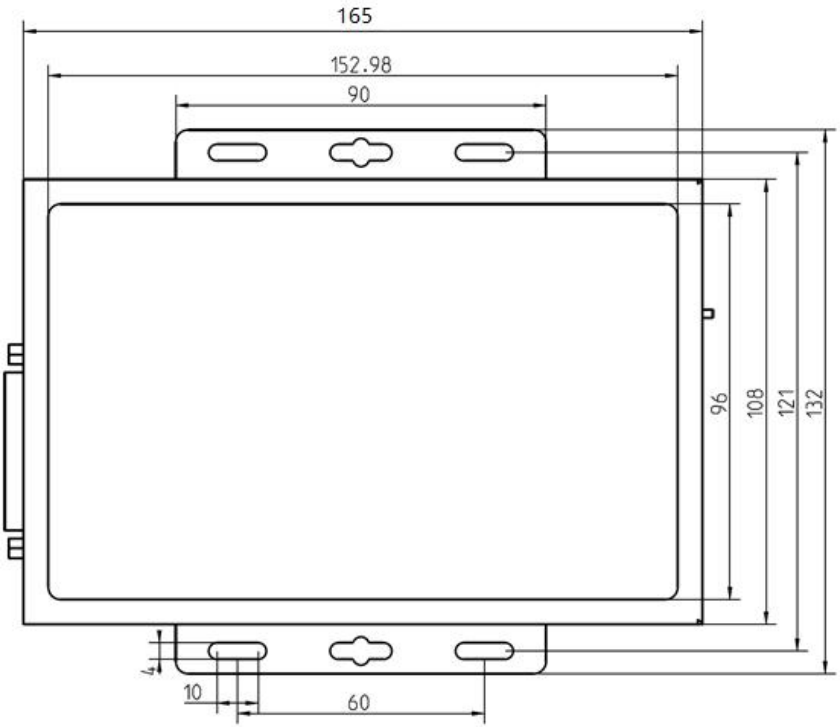
Before performing the Ping command, first ensure that the IP address of PC and SDLC-ETH is on the same subnet.



6 Mechanical characteristics and installation

6.1 Dimension

Dimension (Height x width x depth) = 30 x 132 x 165 mm.

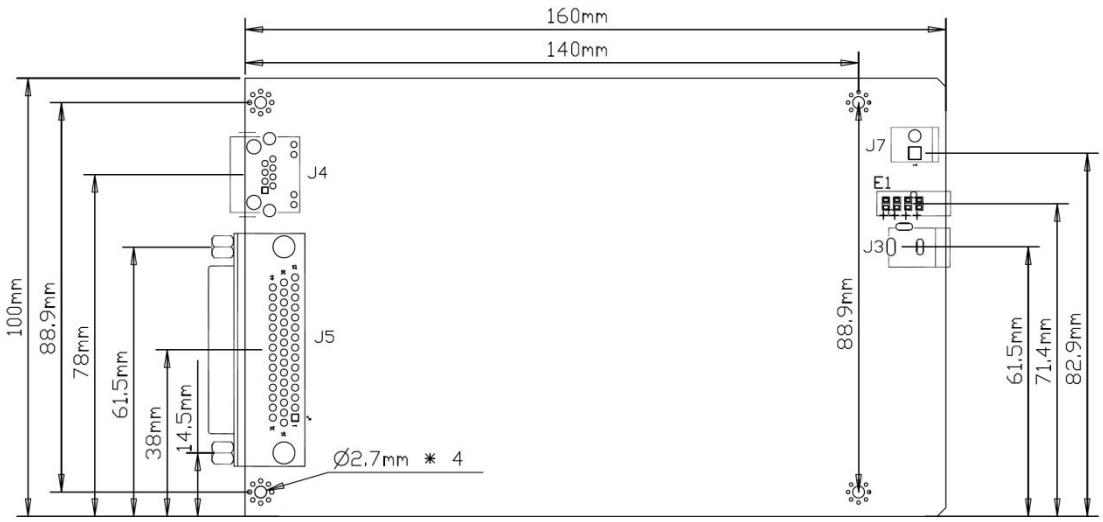


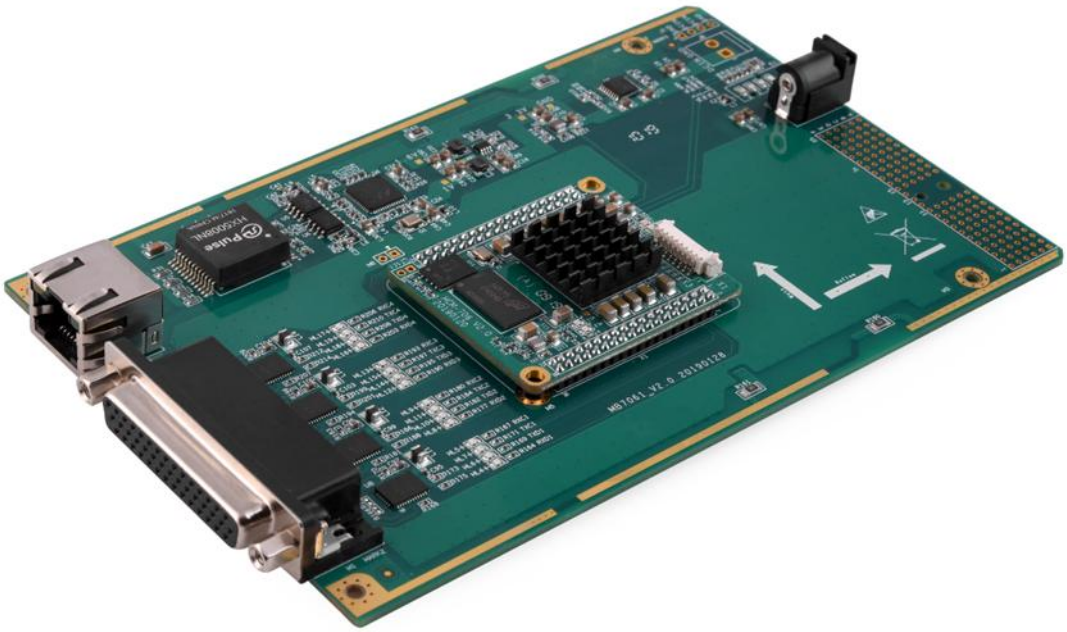
6.2 Cabling / Cable Locks

Serial Connector port use M3 thread by default for the cable locks.

Be sure not to apply the UNC4-40 thread, it will damage the thread.

6.3 Board size





7 Development and Application

7.1 Serial port data conversion

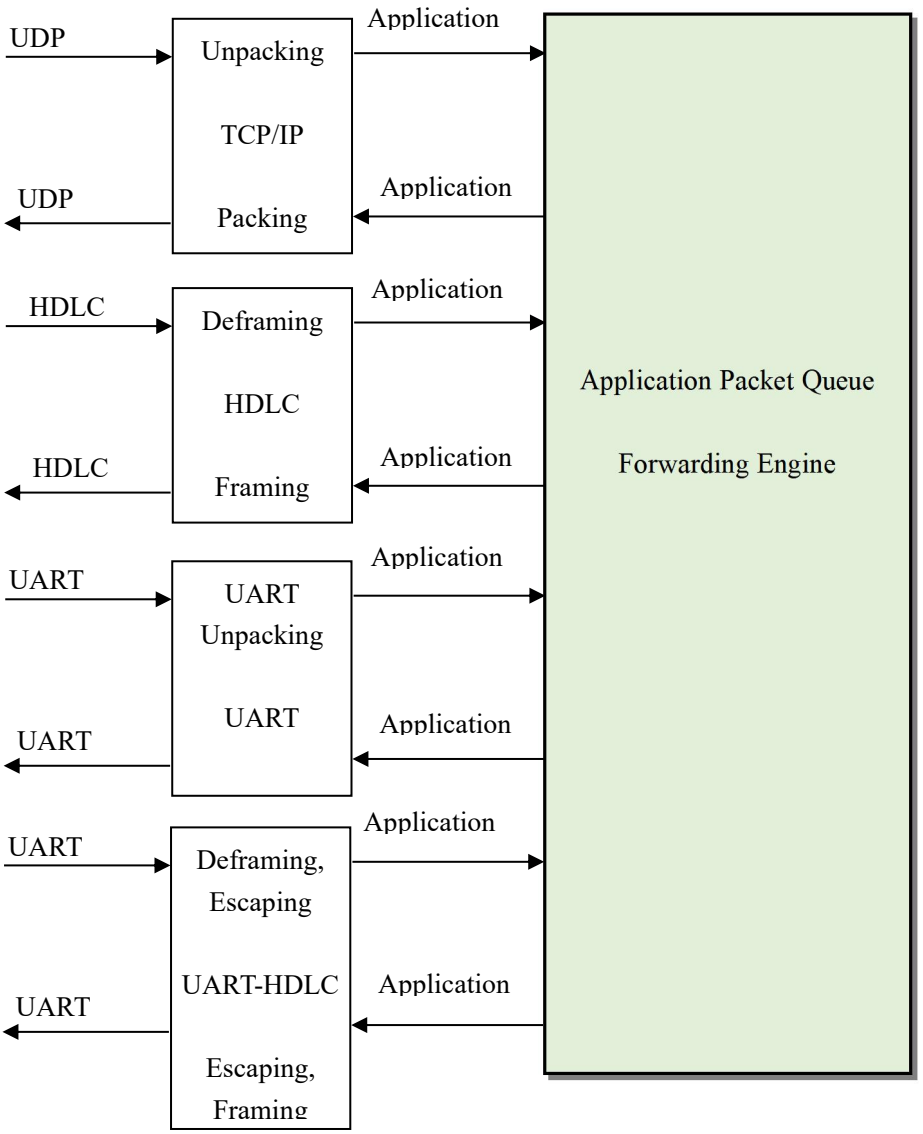
7.1.1 Application packet and conversion model

Serial port data conversion includes:

- Protocol conversion between the serial port and UDP
- Data conversion between synchronous and asynchronous serial ports

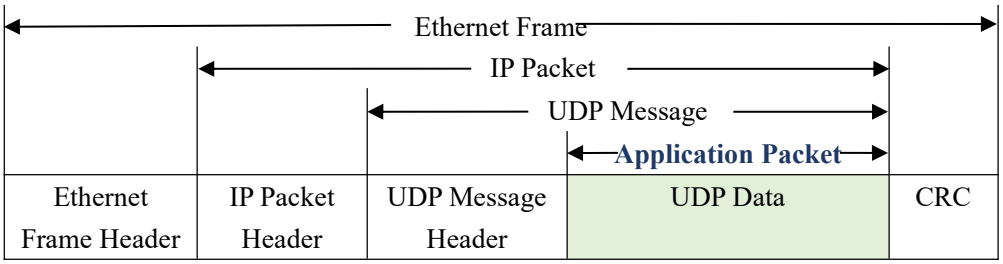
Upon receiving, the receiving and processing module of different types of interfaces unpacks or deframes the data, extracts the application packet, and sends it to the system queue.

SDLC-ETH's forwarding engine will read the application packet and send it to the transmission module of each interface according to the forwarding configuration. It sends modules for framing or packing operation on application packets to generate different types of protocol packets or data frames, which will be sent out through the physical interface.



7.1.2 UDP message format

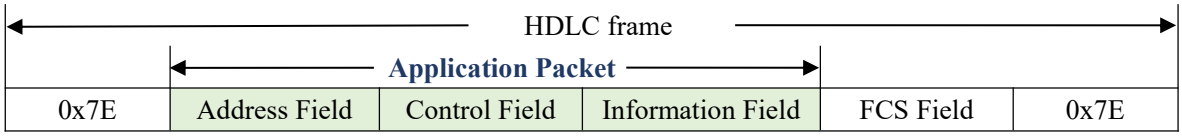
In the UDP protocol, the application packet is packaged in the data area of the UDP message. Each UDP packet contains a complete application packet.



7.1.3 HDLC frame format

A complete HDLC frame consists of several fields between the leading flag and the closing flag, including address field, control field, information field and FCS field for CRC check.

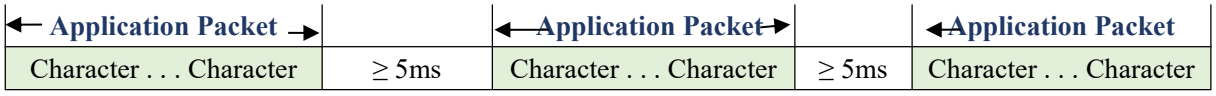
For SDLC-ETH, instead of distinguishing between address field, control field, and information field, they are uniformly presented as application packets to the upper application to fill in and process the UART packet format



7.1.4 UART data packet

When the serial port is working in the asynchronous UART mode, there is a character stream without head or tail received from the serial port, where there is no information used to perform unpacking or deframing.

SDLC-ETH adopts the time information for unpacking, allowing users to define the packet interval of UART. For example, if the packet interval is 5ms, when no new characters are received over 5ms, then the packet receiving is considered to be complete.

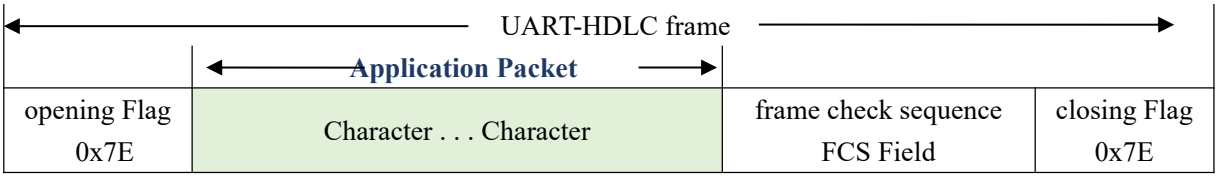


In the actual application, data transmission is not allowed during the packet interval; otherwise, it may result in a waste of communication bandwidth, and the higher the baud rate is, the more serious the waste is.

7.1.5 UART-HDLC frame format

The UART-HDLC working mode adopts another strategy to provide the unpacking capacity for UART. As shown in the following figure, the data sender calculates the application packet's CRC and adds the 0x7e to the head and tail as the leading and closing flags to form an UART-HDLC frame.

This strategy does not require increasing the additional packet interval and can make full use of the communication bandwidth, but increases the processing complexity of both communication sides.



As the application packet and FCS field may appear 0x7E, the sender and receiver shall perform the character escape on the application packet and FCS field with the escape rules as follows:

- 0x7E: Escaped to two characters, 0x7D 0x5E
- 0x7D: Escaped to two characters, 0x7D 0x5D
- Other characters: No escape

The escape operation of data transmit is as follows:

Original Data	Actual Transmit Data
0x7E	0x7D 0x5E
0x7D	0x7D 0x5D
Others	No change

The escape operation of data transmit is as follows:

Actual Receive Data	Data
0x7D 0x5E	0x7E
0x7D 0x5D	0x7D
Others	No change