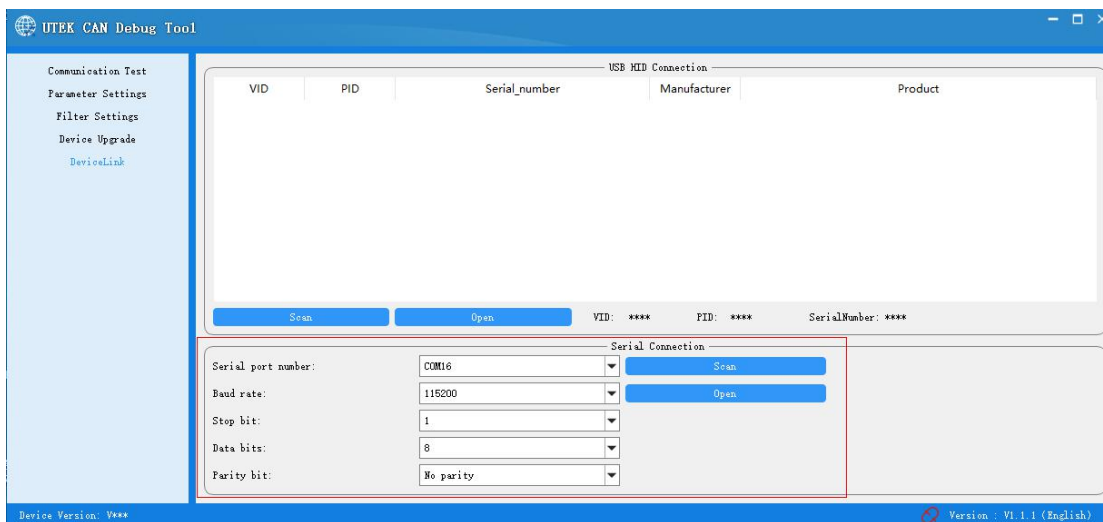


USB/RS232/RS485-to-CAN Software Instructions

1. UT-2505B and UT-2506A Parameter Setting

The UT-2505B and UT-2506A are serial-port-to-CAN conversion devices. To start parameter configuration mode for the module, short-circuit the module's SET pin with GND, so that the system is in the configuration state. Start the host computer software, and select a serial port number. Click to connect to the device and set the parameters. Then disconnect the SET pin from GND to start the normal operating state.



Note: In the configuration mode, the serial port parameters are fixed at 115200, 8, 1, none.

2. UT-8251A Parameter Setting

The UT-8251A is a USB-to-CAN conversion module, which also supports RS232-to-CAN conversion. Scan the USB device by using the host computer software, select a device (UT_8251A), and click **Open** to connect to the device. The UT-8251A allows you to directly set the serial port and conversion parameters over the USB interface. After the settings are completed, power on the device again so that the parameter settings take effect.

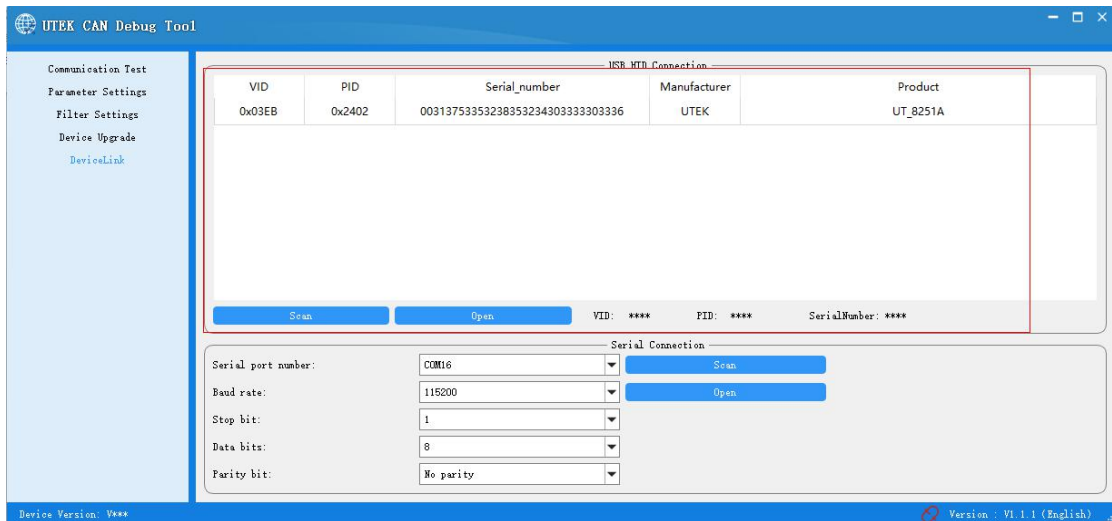


Figure 2.1

3. Software description

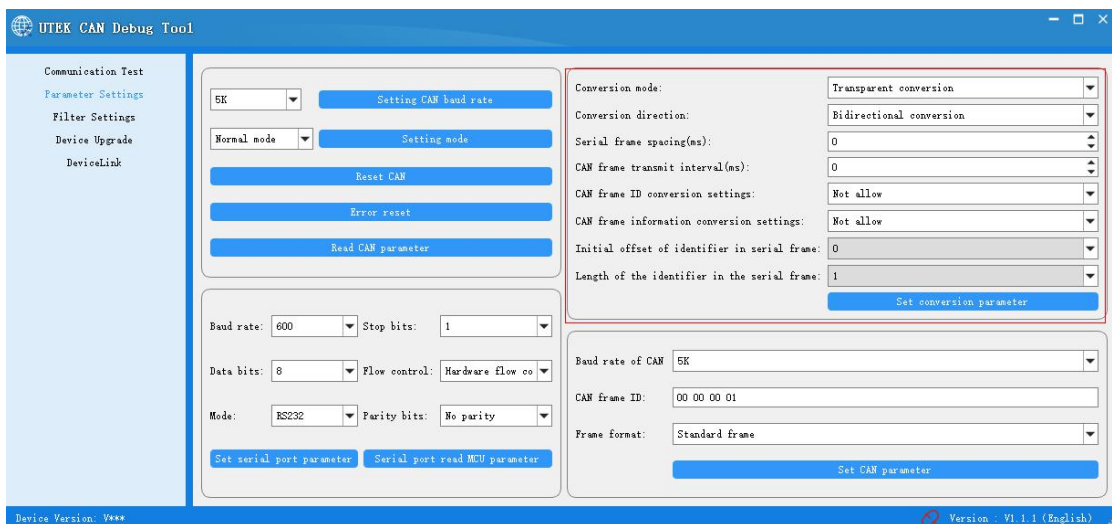


Figure 2.1 Conversion parameter interface

3.1 Parameter description:

3.1.1. Conversion modes: transparent conversion, transparent label conversion, format conversion, and Modbus mode.

3.1.2 Conversion directions:

- (1). Two-way conversion: The converter converts the data of the serial bus to the CAN bus, and converts the data of the CAN bus to the serial bus.
- (2). Serial-port-to-CAN conversion: only converts the serial bus data to the CAN bus, instead of the CAN bus data to the serial bus.
- (3). CAN-to-serial-port conversion: only converts the CAN bus data to the serial bus, instead of the serial bus data to the CAN bus.

3.1.3. Serial frame interval:

This parameter is mainly used for serial frame break. The shortest limit is 3 ms, that is, the break time must be greater than or equal to 3 ms.

3.1.4. CAN frame forwarding interval:

This parameter is mainly used for the interval time of forwarding CAN frames over the serial port. For example, in the case of frequent data transmission on the CAN bus, the module forwards CAN data to the serial bus upon receipt of the data. Then the serial bus sends out two CAN frames almost simultaneously, which is inconvenient for users to distinguish between frame breaks. When the CAN frame time is not set to 0, when the module receives CAN frame data, it is temporarily stored in the message queue, and then forwarded from the serial bus at the time interval set by the user, which helps users break the frame.

3.1.5. CAN frame ID conversion:

This parameter is only used in "transparent conversion" mode. When this option is selected, the converter will add the frame ID of the CAN message before the frame data of the serial frame and after the frame information (if frame information conversion is allowed). When this option is not selected, the CAN frame ID is not converted.

3.1.6. CAN frame information conversion settings:

This parameter is only used in "transparent conversion" mode. When this option is selected, the frame information of the CAN message is added to the first byte of the serial frame when the converter is running. When this option is not selected, the CAN frame information is not converted.

3.1.7. Starting offset of the identifier in the serial frame:

The parameter is only used in the "transparent conversion with ID" mode. When the serial port data is converted into the CAN message, the offset address of the start byte of the frame ID of the CAN message in the serial frame and the length of the frame ID (see 4.2 "Transparent Conversion with ID").

3.1.8. Length of the ID in the serial frame

In a standard frame, the frame ID can be 1 to 2 bytes, corresponding to ID1 and ID2 of the CAN message respectively. In an extended frame, the frame ID can be 1

to 4 bytes, corresponding to ID1, ID2, ID3, and ID4.

3.2. Serial Port Parameters

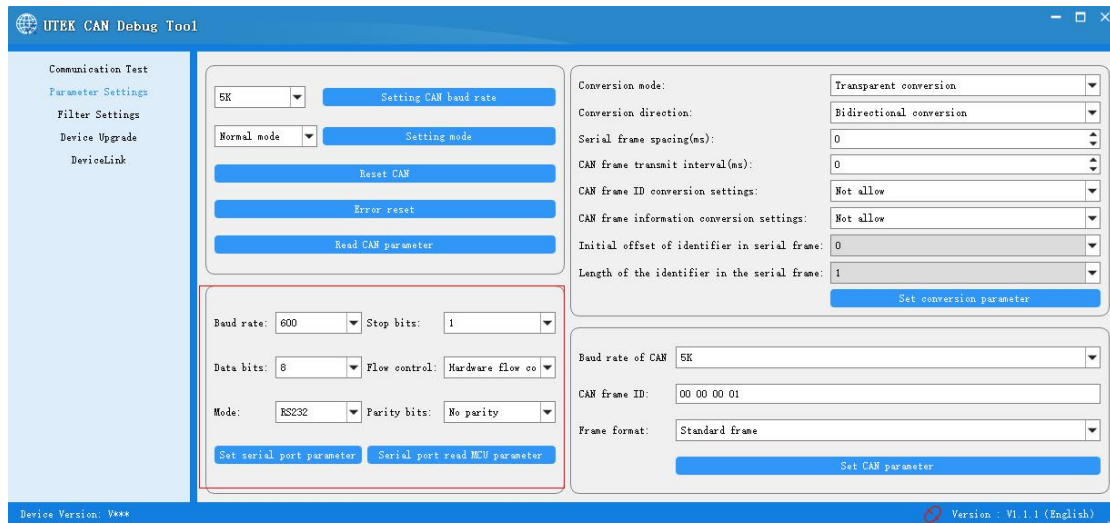


Figure 2.2 Serial port parameters

Baud rate: The serial port baud rate ranges from 600 bits/s to 230,400 bits/s.

Data bits: 5-8.

Stop bits: 1 or 2.

Parity check: no check, even check, odd check, fixed to 0, or fixed to 1.

Mode: RS232 mode, RS485 mode, and RS422 mode (not supported currently).

3.3. CAN Parameters

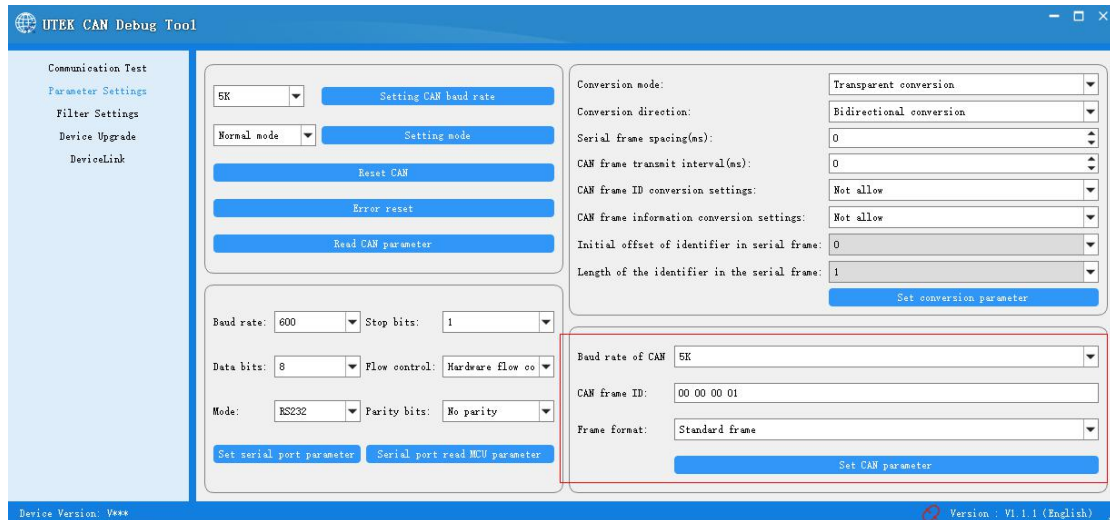


Figure 2.3 CAN parameters

Baud rate: baud rate of the CAN bus

Frame format: frame type of the CAN message during conversion. The options include the standard frame or extended frame.

CAN frame ID: ID of the CAN frame to be sent (this parameter is valid in transparent transmission mode)

3.4. Filtering Parameters

The module provides 10 sets of filtering and selective reception, which minimizes the network load of the network.

Example: Set the acceptance standard frame list ID to 0x08,0x12, and set the extended frame group ID to 0x55 to 0x66, as shown in Figure 2.4.

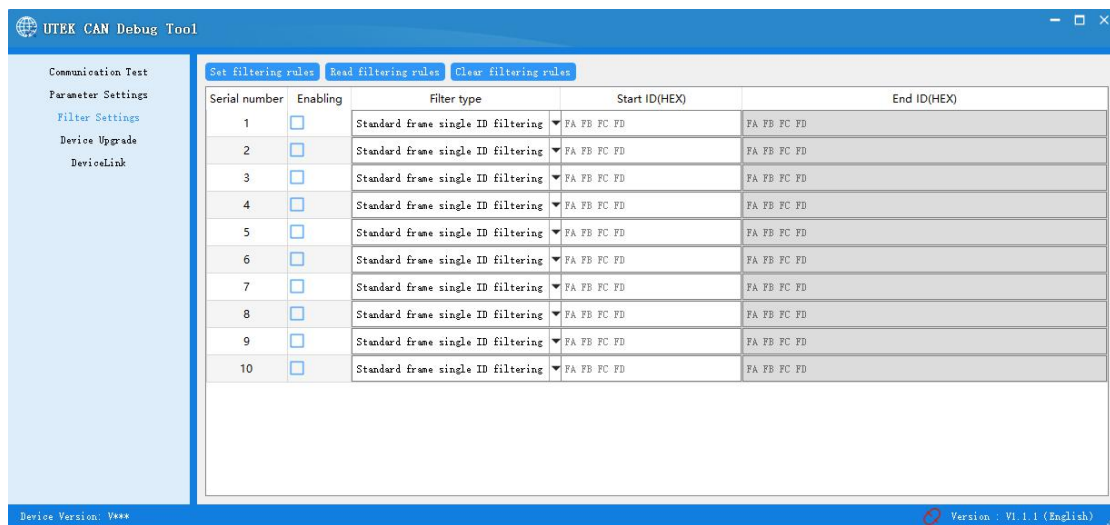
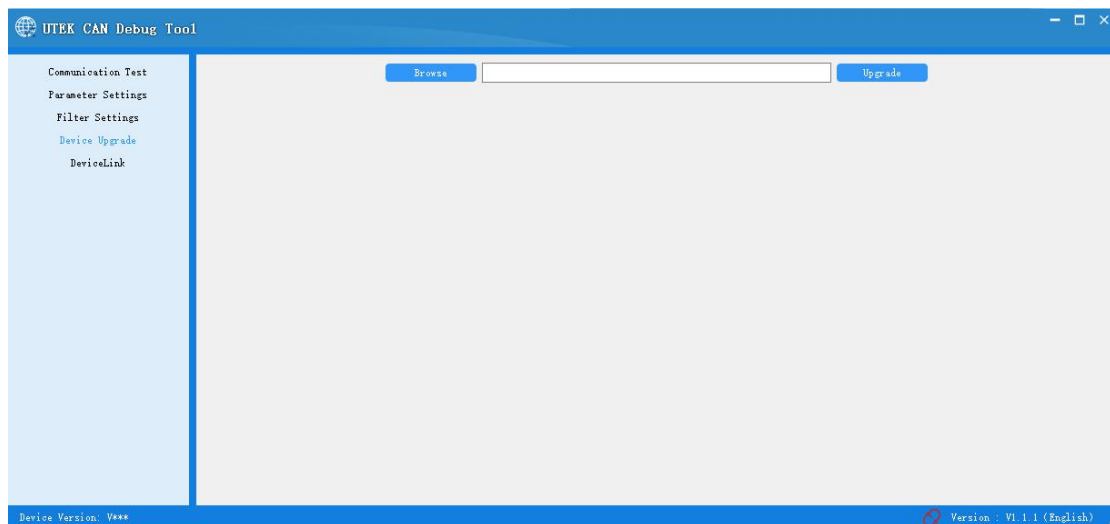


Figure 2.4 Filtering parameters

3.5. Equipment Upgrade



Click **Browse**, select a BIN file, and click **Upgrade**. The upgrade file must be an official file; otherwise, the device may not work properly.

4. Conversion Examples

4.1. Transparent Conversion

4.1.1 Serial-Frame-to-CAN conversion message

All the data of the serial frame is sequentially filled into the data field of the CAN message frame. As soon as the converter detects data on the serial bus, the converter receives and converts the data.

The converted CAN message frame information (frame type part) and frame ID come from the user's prior configuration, and the frame type and frame ID remain unchanged during the conversion. Figure 4.1 shows the data conversion format.

If the line frame length of the received string is less than or equal to eight bytes, fill 1 to n (n is the length of the serial frame) sequentially in the 1 to n bytes positions (7 in Figure 3.1n) of the data field of the CAN message.

If the serial frame contains more than eight bytes, the processor starts from the first character of the serial frame, and sequentially fills eight characters in the data field of the CAN message. After the data is sent to the CAN bus, the remaining serial frame data is converted and filled into the data fields of the CAN message until all data is converted.

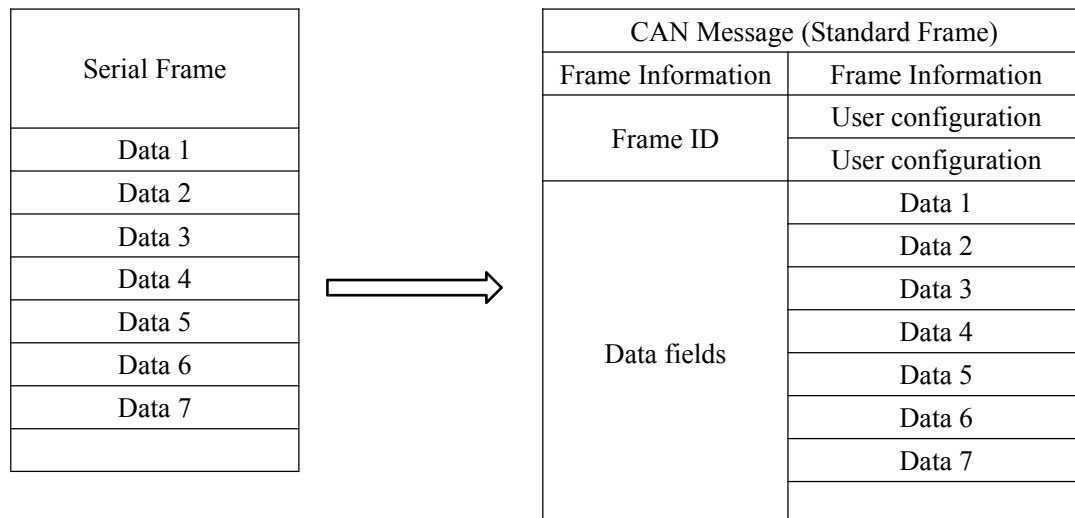
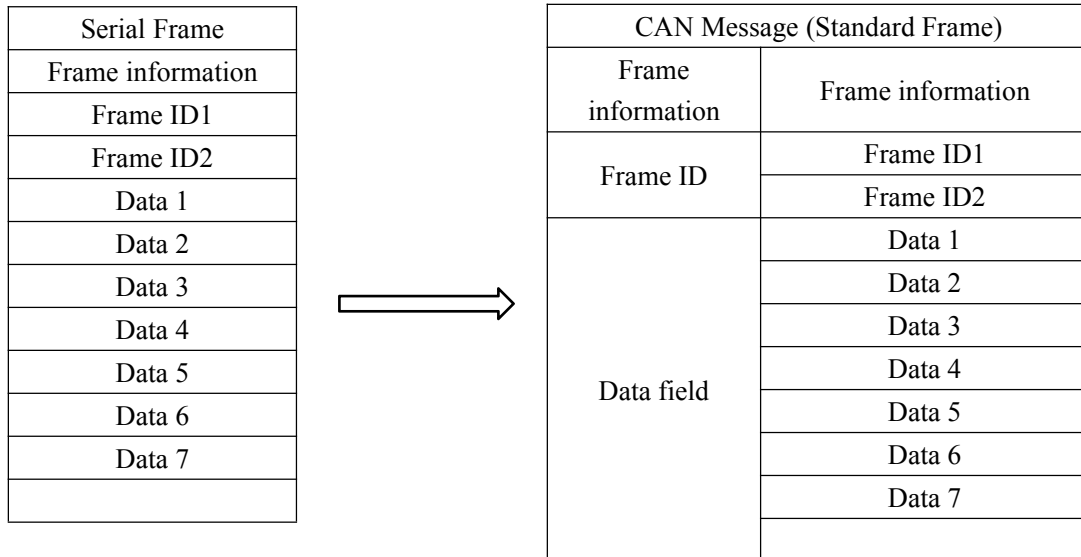


Figure 3.1 Converting a serial frame into a CAN message (transparent mode)

4.1.2. Converting a CAN Message to a Serial Frame

For CAN bus messages, a frame is immediately forwarded upon receipt. Figure 3.2 shows the data format. All the data in the CAN message data field are sequentially converted into the

serial frame. If **Frame Information Conversion** is set to **Enable**, the converter directly fills the **Frame Information** byte of the CAN message into the serial frame. If **Frame ID Conversion** is set to **Enable**, all the **Frame ID** bytes of the CAN message are also filled into



the serial frame.

Figure 3.2 Converting a CAN message to a serial frame (transparent mode)

4.2.Transparent Transmission with ID

Transparent conversion with ID is a special transparent conversion, which helps you construct your own network and use your own application protocol by using the converter. This method automatically converts the address information in the serial frame into the frame ID of the CAN bus. As long as the converter learns about the start position and length of the address in the serial frame, the converter extracts this frame ID and fills it in the frame ID field of the CAN message during conversion as the ID of the CAN message during the forwarding of the serial frame. When the CAN message is converted into a serial frame, the ID of the CAN message is converted to the corresponding position in the serial frame.

4.2.1 Serial-frame-to-CAN conversion message

The start address and length of the CAN ID in the serial frame can be set. The starting address ranges from 0 to 7, and the length ranges from 1 to 2 (standard frame) or 1 to 4 (extended frame). During conversion, the CAN frame ID in the serial frame is converted to the frame ID field of the CAN message (using the terminal storage method. If the number of frame IDs is less than that in the CAN message, 0 is added to the lower bytes of the frame ID in the CAN message) based on

the prior configuration. Other data is converted sequentially, as shown in the figure.

If a CAN message has not converted all the serial frame data, the same ID is still used as the frame ID of the CAN message until the serial frame conversion is completed.

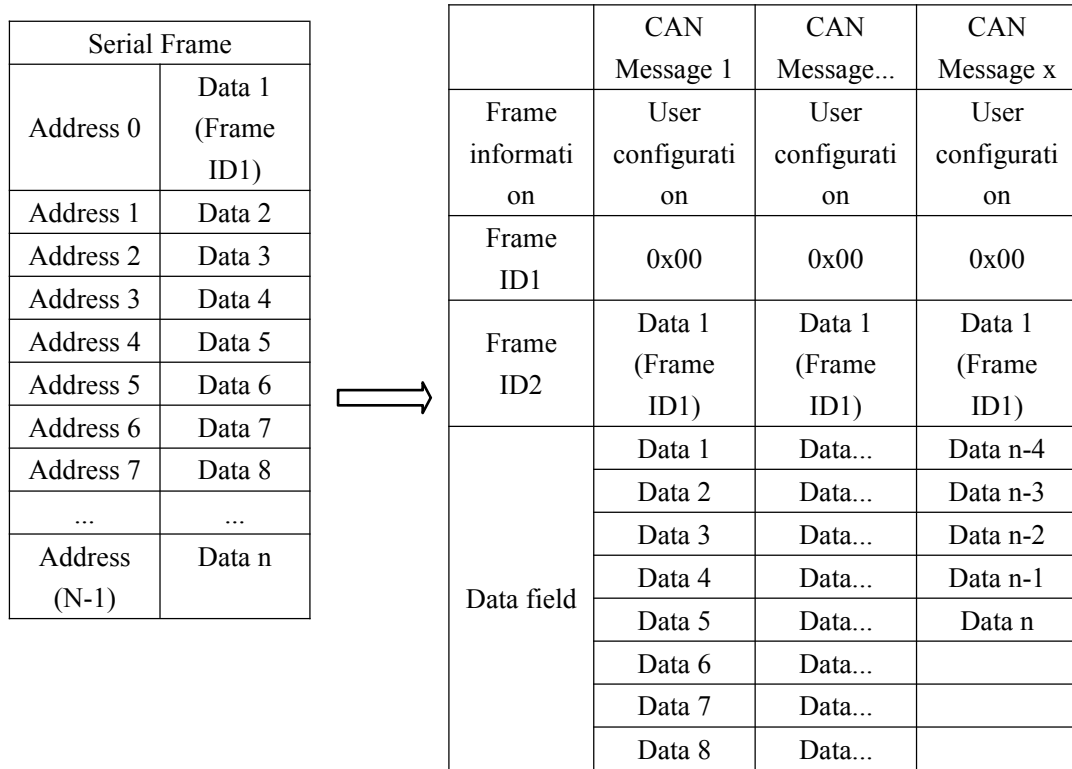


Figure 3.3 Converting a serial frame to a CAN message (transparent transmission with ID)

4.2.2 Converting a CAN message to a serial frame

For CAN messages, a frame is immediately forwarded upon receipt. Each time it is forwarded, the ID in the received CAN message is converted based on the position and length of the CAN frame ID configured in the serial frame. Other data is forwarded in order, as shown in Figure 3.4. Note that the frame format (standard frame or extended frame) of serial frames and CAN messages should meet the configured format requirements; otherwise, communication may fail.

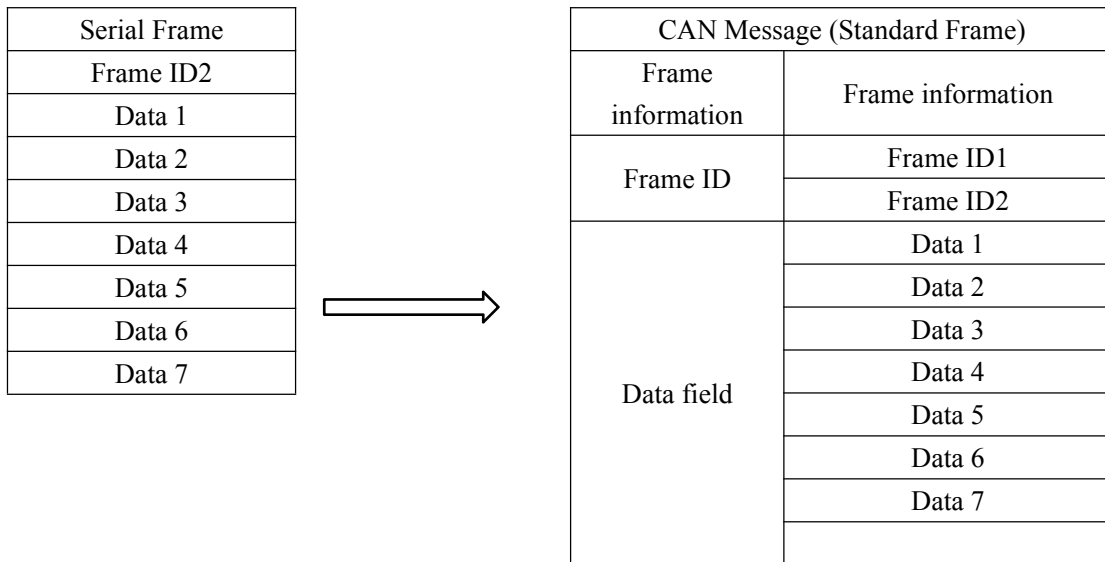
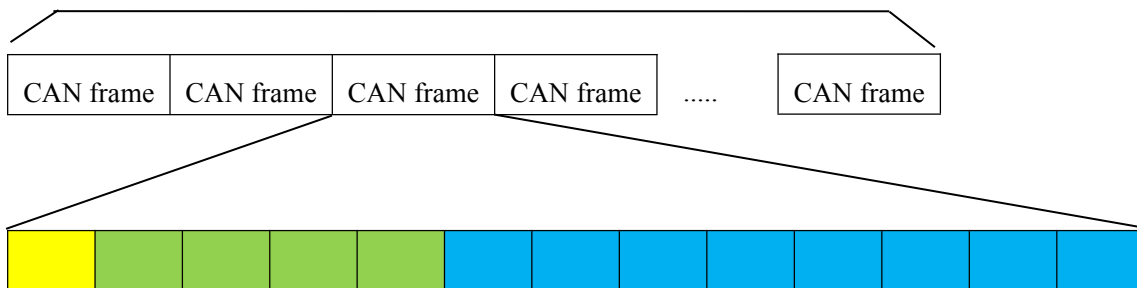
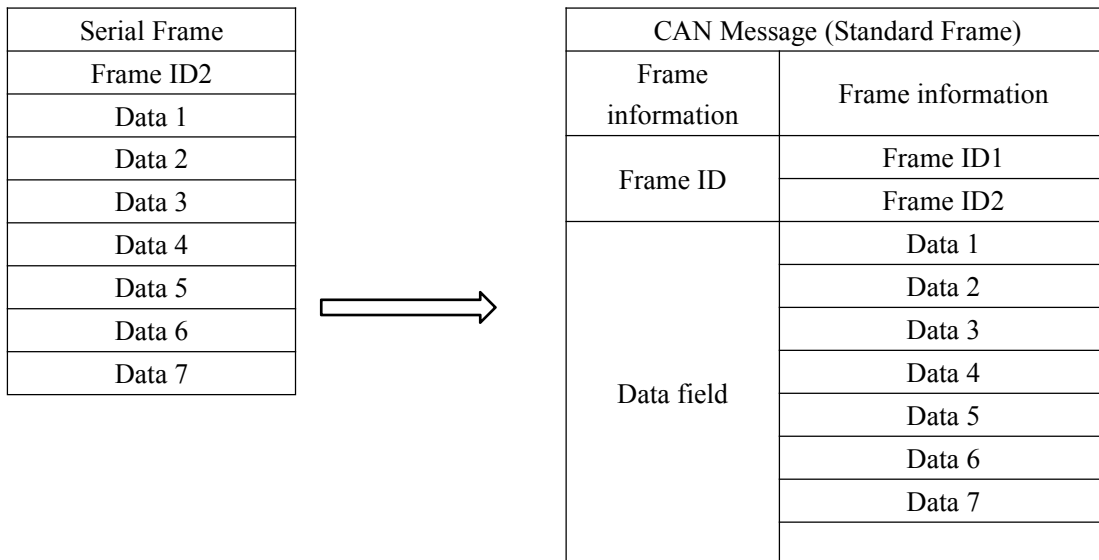


Figure 3.4 Converting a CAN message to a serial frame (transparent transmission with ID)



One CAN frame contains 13 bytes

Frame information: one byte in length, used to identify some information of the CAN frame, such as type and length.




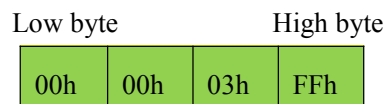
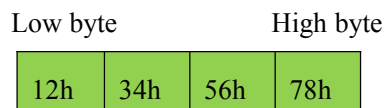
FF: identifies the standard frame and extended frame. 1 indicates the extended frame, and 0 indicates the standard frame.

RTR: identifies the remote frame and data frame. 1 indicates the remote frame, and 0 indicates the data frame.

The reserved value is 0, instead of 1.

D3-D0: identifies the data length of the CAN frame.

 Frame ID: four bytes in length. It is 11 bits for the standard frame and 29 bits for the extended frame.




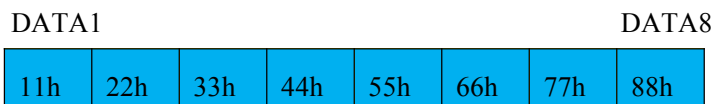
This is the extended frame ID.

This is the standard frame ID.

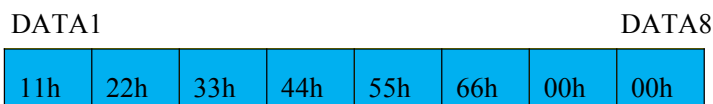
Representation of 0x12345678

Representation of 0x3FF

 Frame data: eight bytes in length. The effective length is determined by the value of D3-D0 in the frame information.



This is the representation of 8-byte valid data.



This is the representation of 6-byte valid data.

4.4 Modbus Conversion

Modbus protocol is a standard application layer protocol, which is widely used in various industrial control scenarios. The protocol is open, with strong real-time performance and good communication verification mechanism, which makes it suitable for occasions with high communication reliability requirements.

The converter uses the standard Modbus RTU protocol format on the serial port side. Therefore, the converter allows you to use the Modbus RTU protocol and can directly interface with other devices that support the Modbus RTU protocol. On the CAN side, a simple and easy-to-use segmented communication format is developed for Modbus communication. The converter is still used for protocol verification and forwarding, and supports Modbus transmission, rather than Modbus master or slave host. You need only to establish communication over Modbus.

Note that in this conversion mode, the "CAN ID" of the "CAN parameter" item of the configuration software is invalid, because the identifier (frame ID) sent at this time is filled with the address field in the Modbus RTU serial frame.

CAN 总线帧格式说明如下：

Bit Number	7	6	5	4	3	2	1	0
Frame information	FF	RTR	X	X	DLC (data length)			
Frame ID1	X	X	X	ID.28 - ID.24				
Frame ID2	ID.23 - ID.16							
Frame ID3	ID.15 - ID.8							
Frame ID4	ID.7 - ID.0							
Data 1	Segment mark	Segment type	Segment counter					
Data 2	Character 1							
Data 3	Character 2							
Data 4	Character 3							
Data 5	Character 4							
Data 6	Character 5							
Data 7	Character 6							
Data 8	Character 7							

Segmented message tag: indicates whether the message is a segmented message. If this bit is 0, it indicates a single message; if this bit is 1, it belongs to a frame in the segmented message. Note: When the CAN message is a single frame, the segment flag bit value is 0x00.

Segment type: indicate whether it is the first, middle or last. Table 4.2 lists its value definitions.

Table 4.2 Bit value of the segment type

Bit value	Meaning	Description
0	First segment	If the segment counter contains the value 0, this is the first segment in the segment series.
1	Middle segment	Indicates an intermediate segment.
2	Last segment	Indicates the last segment.

Conversion description: During the conversion from the serial port side to the CAN side, the converter works only after receiving a complete and correct Modbus RTU; otherwise, it does not work.

As shown in Figure 3.5, the address field of the Modbus RTU protocol is converted into ID4 (extended frame) or ID2 (standard frame) of the frame ID in the CAN message, and the ID remains unchanged during the frame conversion.

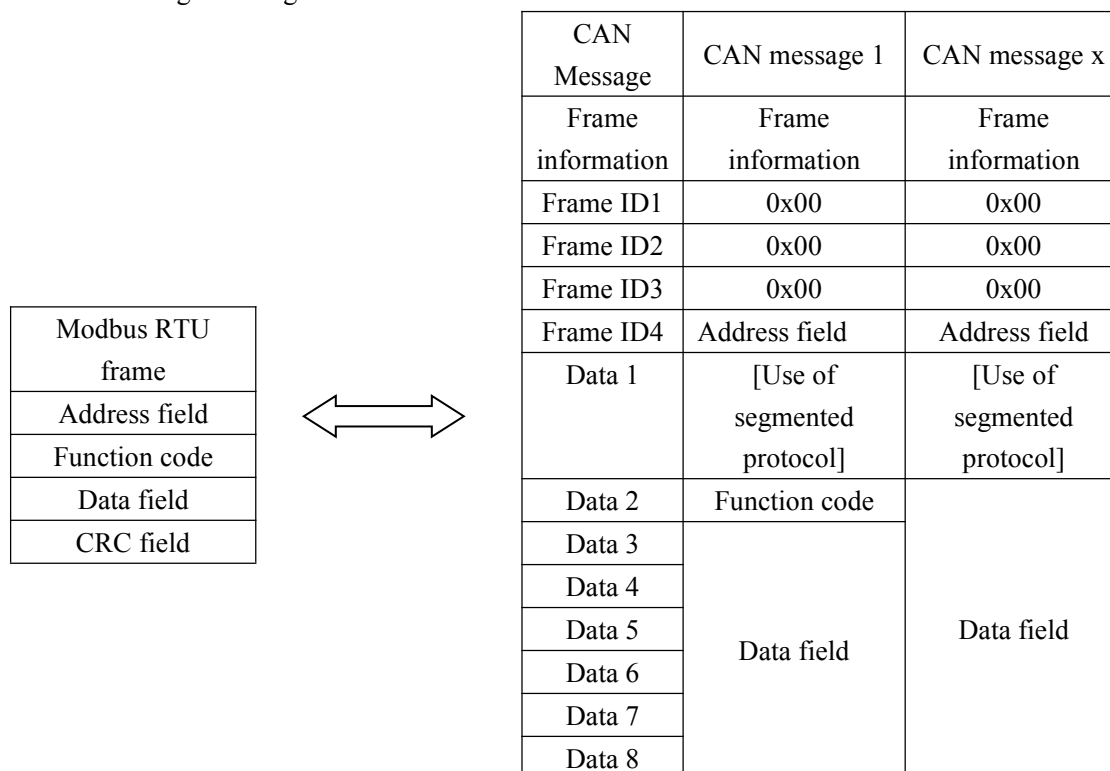


Figure 3.5 Communication frame conversion format (Modbus mode)

The CRC check byte is not converted to the CAN message, and the CAN message does not need to have the check byte of the serial frame, because the CAN bus itself has a better check mechanism.

What is converted is the Modbus RTU protocol content - function code and data field. During conversion, they are converted to the data fields of the CAN message frame in turn (starting from the second data byte. The first data byte is used for the segment protocol), because the length of the Modbus RTU frame varies according to the function code. However, the converter segments the longer Modbus RTU frame into a CAN message and sends it using the preceding CAN segmentation protocol because a CAN message can only transmit seven data items in one frame.

You need only to process the function code and data field when receiving data on the CAN node.

For the Modbus protocol data of the CAN bus, there is no need to perform cyclic redundancy check (CRC16). The converter receives data over the segmentation protocol, automatically adds cyclic redundancy check (CRC16) after receiving a frame for analysis, converts it into Modbus RTU frame, and sends it to the serial bus.

If the received data does not conform to the segmentation protocol, the group of data is discarded without conversion.