Appendix ED3100 communication protocol

- **communication protocol**

MODBUS agreement, the host asked, in active state; the slave machine answer, passive.

(note: not all the host asked frame, the slave machine can answer. For example, radio host from machine won't response)

- **address**

set range:01 ~ 31

when systems use RS 485 serial communication interface control or monitoring, each driver must set the communication address and every connection in each net address is "only" and not repeatable.

The factory set value: 01

- **communication transfer speed**

baud rate, as shown in the table size parameters

- **communication error handling**

specific error code, as shown in the parameter table

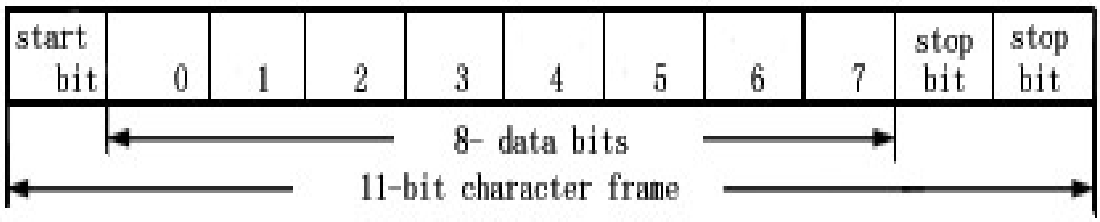
- **communication overtime detection**

this parameters set series communication timeout detection, time. During the parameters set time, without any information transmission, it indicates that the communication timeout, and the specific time, see parameter table.

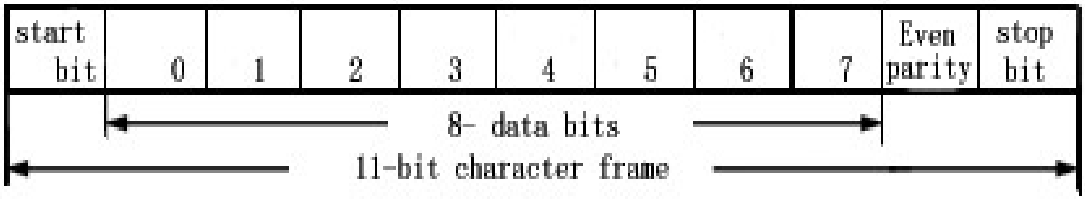
- **BIT stream format**

MODBUS RTU communication is divided into two kind of coding method and ASCII coding, here in direct transmission mode, character RTU structure: 11, can be the following three formats one of arbitrary. Specific choice way, please sees the parameter table.

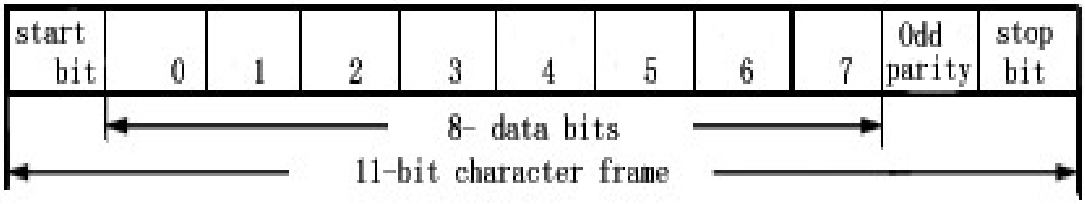
- (**Material format** 8，N，2)



- (**Material format** 8，E，1)



- (**Material format** 8，0，1)



- **communication material structure (RTU mode)**

Any of a frame of RTU mode, modbus data formats are as follows:

10 ms interval +slave machine address + function code+ specific data + CRC CHK + 10 ms interval

			|_ Length: 2 byte
			____ Length: many bytes
		_____ Length: 1 byte	
	_____ Length: 1 byte		

The following list may be more intuitive, but meaning unchanged:

STX	**Keep no input signal is equal or more than 10 ms**
Address	**The communication address: 8-bit binary address**
Function	**Function code: 8-bit binary address**
DATA(n-1) …… **DATA 0**	**Material content:** **N * 8-bit material, n < = 2 (2 pen 16 bit material)**
CRC CHK Low **CRC CHK High**	**CRC check code:** **16-bit CRC check code by two 8-bit binary combination**
END	**Keep no input signal is equal or more than 10 ms**

The specific meaning of the format is as following:

Address : communication, 0 ~ 31 (decimal)

00 H: all drive radio (be sensed), slave radio machine not respond.

01 H ~ 1 FH: a specific address to drive.

Function: Function code, also called command byte, there are four possibilities:

03 H: read register.

06 H: write a data into registers.

10 H: write many data into registers .

DATA(n-1): specific data, will have the application example .

Check code (CRC Check) for RTU model, detailed introduction in the last page.

- **Functions code of the corresponding communication frames for example :**

03 H: read register content

For example: in the drive for the internal FH address 1 set parameters for 0006 H (F006) reads the parameter value:

Ask message frame format:

Address	1FH
Function	03H
Starting data address	00H
	06H
Sizes	00H
	01H
CRC CHK Low	67H
CRC CHK High	B5H

Feedback frame format:

Address	1FH
Function	03H
data address	00H
	06H
Data content	10H
	88H
CRC CHK Low	ABH
CRC CHK High	D3H

Ask frame： 1FH+03H+00H+06H+00H+01H+67H+B5H

Specific meaning is as follows:

- **Address** ：1FH **----** This device ID is 1 FH.
- **Function** ：03H ---- Read the register contents.
- **Starting data address**：0006H ---- Register address is 0x0006, indicates read parameters from the register.
- **Sizes** ：0001H ---- Read the data of one address .
- **CRC CHK**：Reference to the last page for methods to get RTU mode check code (CRC Check).

06 H: write a data into registers

For example: the drive address 1FH, writing 5000 (1388H) to the internal drive setting parameters 0006H.

Inquiry message frame format:

Address	1FH
Function	06H
Data address	00H
	06H
Data content	13H
	88H
CRC CHK Low	67H
CRC CHK High	23H

Response message format:

Address	1FH
Function	06H
Data address	00H
	06H
Data content	13H
	88H
CRC CHK Low	67H
CRC CHK High	23H

Asking frame ：1FH+06H+00H+06H+13H+88H+67H+23H

Specific meaning as follows:

- **Address ：1FH ---- The device ID is 1FH.**
- **Function ：06H ---- Write register contents.**
- **Data address：0006H ---- Register address 0x0006, indicates write content to register.**
- **Data content ：1388H ---- Written content, 0x0006 write 1388H.**
- **CRC CHK：Reference to the last page for methods to get the RTU mode check code (CRC Check).**

Response frame ：1FH+06H+00H+06H+13H+88H+67H+23H

Specific meaning as follows:

- **Address ：1FH ----** The device ID is 1FH.
- **Function ：06H ----** Write register contents.

- **Data address：0006H** ---- Register address 0x0006, indicates read parameters from the register.
- **Data content ： 1388H** ---- That written contents in the register.
- **CRC CHK：** Reference to the last page for methods to get the RTU mode check code (CRC Check).

10H: write several data continuously (only continuously change two parameters)

For example, change the drive (Address 1FH) of the upper and lower frequency setting 00-06 = 50.00 (1388H) ,00-07 = 00.01 (0001H)

Asking frame ：

Address	1FH
Function	10H
Data start/finish address	00H
	06H
Amount of data （word）	00H 02H
Amount of data （Byte）	04H
The first data	13H 88H
The second data	00H 01H
CRC CHK Low	56H
CRC CHK High	C3H

Response frame ：

Address	1FH
Function	10H
Data start/finish address	00H 06H
Amount of data （word）	00H 02H
CRC CHK Low	A2H
CRC CHK High	77H

Asking frame **:1FH+10H+00H+06H+00H+02H+04H+13H+88H+00H+01H+56H+C3H**

Specific meaning as follows:

- **Address ： 1FH ----** The device ID is 1FH.
- **Function ： 10H ----** Write register contents.
- **Start Address: 0006H** ---- register start address is 0x0006, indicates write content to 0x0006, 0x0007.
- **Amount of data (word):** 0002H ---- the amount of words in written content
- **Amount of data (Byte) :**04 ------ the amount of bytes in written content.
- **The first data (s):** the content 1388H first writes
- **The second data(s):** the content 0001H second writes.
- **CRC CHK:** refer to the last page for the method to get the RTU mode check code (CRC Check).

Response frame ：1FH+10H+00H+06H+00H+02H+A2H+77H

Specific meaning as follows:

- **Address**: 1FH ---- The device ID is 1FH
- **Function**: 10H ---- Write registercontents
- **Start/finish Address**: 0006H ---- register start/finish address is 0x0006, indicates write content to 0x0006, 0x0007.
- **Amount of data (word)**: 0002H---- word amount of written contents.
- **CRC CHK:** refer to the last page for the method to get the RTU mode check code (CRC Check) access method.

Please note, respond frame only return the first six bytes of the asking frame, the CRC CHK is the six-byte CRC CHK.

communication protocol defines the parameters of the word address, refer the following table:

Definitions	**Parameter address**	**Function Description**
Setting parameters in the drive	GGnnH	GG indicates parameter group, nn means parameter number. For example: 04-01 represented by 0401H.

Command the drive	2000H	Bit0～1	00B: No function
			01B: Stop
			10B: Start
			11B:JOG Start
		Bit2～3	00B: A run
			11B: Continuous operation
		Bit4～5	00B: No function
			01B: FWD direction command
			10B: REV direction command
			11B: No function
		Bit6～7	Keep
		Bit8～11	Keep

Start forward command and written operating frequency for a given example (continuous operation):

Asking frame:

Address	1FH
Function	10H
Data start address	20H
	00H
Amount of data （word）	00H 02H
amount of data (Byte)	04H
Driver command	00H 1EH
The value of a given frequency	10H 88H
CRC CHK Low	67H
CRC CHK High	E6H

Response frame:

Address	1FH
Function	10H
Data start address	20H
	00H
Amount of data （word）	00H 02H
CRC CHK Low	49H
CRC CHK High	B6H

Start forward command and written operating frequency for a given instance (one time run):

Asking frame:

Address	1FH
Function	10H

Data start address	20H
	00H
Amount of data （word）	00H 02H
amount of data (Byte)	04H
Driver command	00H 12H
The value of a given frequency	10H 88H
CRC CHK Low	A7H
CRC CHK High	E5H

Response frame:

Address	1FH
Function	10H
Data start address	20H
	00H
amount of data （word）	00H 02H
CRC CHK Low	49H
CRC CHK High	B6H

Reverse run command and start to write given frequency ,for example (continuous operation):

Asking frame:

Address	1FH
Function	10H
Data start address	20H
	00H
Amount of data（word）	00H 02H
amount of data (Byte)	04H

Driver command	00H 2EH
The value of a given frequency	10H 88H
CRC CHK Low	67H
CRC CHK High	E9H

Response frame:

Address	1FH
Function	10H
Data start address	20H
	00H
amount of data （word）	00H 02H
CRC CHK Low	49H
CRC CHK High	B6H

Written stop order for example:

Asking frame:

Address	1FH
Function	06H
Data start address	20H
	00H
Driver command	00H
	01H
CRC CHK Low	40H
CRC CHK High	74H

Response frame:

Address	1FH
Function	06H
	20H
Data start address	

	00H
Driver command	00H
	01H
CRC CHK Low	40H
CRC CHK High	74H

● read monitoring parameters

For example: read control parameters from the drive's internal address 1FH set parameters 0D00H (FD00)

Inquiry message frame format:

Address	1FH
Function	03H
data address	0DH
	00H
Retained data	00H
	00H
CRC CHK Low	44H
CRC CHK High	D8H

Response message frame format

Address	1FH
Function	03H
data address	0DH
	00H
B0B1	10H
	84H
B2B3	41H
	48H
CRC CHK Low	47H
CRC CHK High	D5H

Among them, the retain data in asking frame does not affect the results, can be any design. As to B0B1, B2B3 can refer to the following table. (Note that reading invalid address more than 0D28)

B0 ~ B3 Parameters:

B0 B1		parameter value of Monitoring 1
B2 B3	0	Negative one decimal, shows the last point
	1	0 decimal places
	2	1 decimal places
	3	2 decimal places
	4	3 decimal places
	5	V
	6	Hz
	7	A
	8	Parameters normal
	9	Retained parameters
	10	Retained parameters
	11	Retained parameters
	12	Input terminals format
	13	Fault format
	14	Stopped, displays 0
	15	Retained parameters

- **Read fault code**

For example: read fault code from the drive's internal address 1FH set parameters 0E01H (FE01):

Inquiry message frame format:

Address	1FH
Function	03H
data address	0EH

	01H
Retained parameters	00H
	00H
CRC CHK Low	15H
CRC CHK High	5CH

Response message frame format::

Address	1FH
Function	03H
data address	0EH
	01H
B4B5	FFH
	FFH
B6B7	01H
	48H
CRC CHK Low	0FH
CRC CHK High	2BH

Among them, retained data in the asking frame does not affect the results, can be any design. If no error occurs, the B4B5 return FFFFH, otherwise refer to the following table for the returned B4B5B6B7

B4 ~ B7 Parameters:

B4 B5		Same as the above B2 B3, 4-10, the other bit is invalid	
B6 B7	0	Voltage normal	
	1	Electric steering 0 - Forward 1 - Reverse	
	2	Output phase sequence 0 - Forward 1 - Reverse	

	3	Command direction 0 - Forward 1 - Reverse	
	4	Operation	
	5	Failure, failure code in the B4B5 5 ~ 11	
	6	Frequency channel to 0 (digital setting)	
	7		
	8		
	9		
	10	Acceleration	
	11	Deceleration in	
	12		
	13		
	14		
	15		

- **RTU mode check code （CRC Check）**

Check the code by the end of Address to Data content.

The operation rules are as follows:

Step 1: make 16-bit register (CRC register) = FFFFH.

Step 2: Exclusive OR the first 8-bit byte message command with the low bit

16-bit CRC register, do the Exclusive OR, the result of the CRC register.

Step 3: right a CRC register, fill in the high bit at 0.

Step 4: Check the right value, if it is 0, step 3 of the new value into the CRC register, otherwise the Exclusive OR A001H and CRC register, the register of the CRC.

Step 5: Repeat step 3 and 4, the 8-bit all the operations completed.

Step 6: Repeat step 2 to 5, removed an 8-bit message command, until completion of the operation instructions of all messages. Finally, get the value of the CRC register, which is the CRC checksum. Note that the CRC check code must be placed in the message exchange check code instructions.

The following is written in C language example of CRC generation:

```
unsigned char* data    //  Message instruction pointer
unsigned char length    //  The length of the message instruction
unsigned int crc_chk(unsigned char* data, unsigned char length)
{
int j;
unsigned int reg_crc=0xffff;
while(length--)
{
   reg_crc ^= *data++;
   for(j=0;j<8;j++)
   {
     if(reg_crc & 0x01)
      { /* LSB(b0)=1 */
        reg_crc=(reg_crc>>1) ^ 0Xa001;
      }
       Else
       {
       reg_crc=reg_crc >>1;
       }
    }
}
```

return reg_crc;// **Finally, the return value of the CRC register**