Modbus RTU／TCP Client User Manual

LinPAC／LinCon

Version 1.0

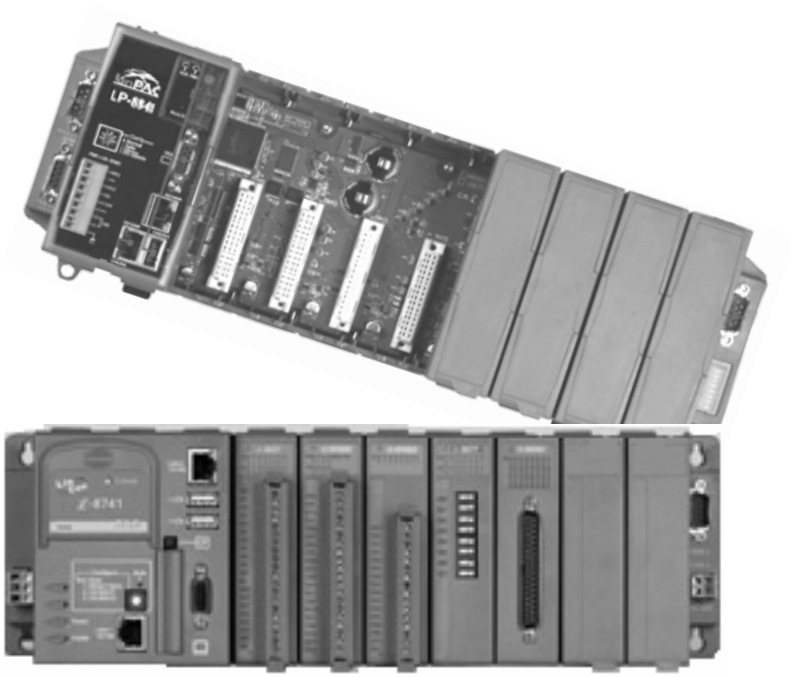
This document applied to models :

LinPAC-8000(Linux kernel 2.6.19 based)

LinPAC-8x81(Linux kernel 2.6.18 based)

LinCon-8000(Linux kernel 2.4.21 based)

.



CONTENTS

CH 1 Introduction

This manual is written for modbus users of *ICPDAS LinPAC and LinCon series*, and it only supports 8K & 87K I/O Modules.

1-1. Develop Modbus Client AP on your LinPAC／LinCon

If your LinPAC／LinCon is former released, you have to download Modbus Client Develop Tool from ICP DAS websit.
http://www.icpdas.com/download/download-list.htm
Develop Tool includes the "mbclib.a" - library file and the "mbcapi.h" - header file.

To develop your modbus client applications, the "mbcapi.h" file must be included in front of the source program,
`#include "mbcapi.h"`
and "mbclib.a" must be linked during building your applications.
`–lm mbclib.a`

CH 2 Modbus Client library

The mbclib.a is a modbus client library for LinCon／LinPAC I/O Modules.

2-1. Connection

To initialize the Modbus TCP connection :

int cMBTCPInit (char *server,int tid, int pid, int timeout)

Parameter:

server	in	Modbus server IP address. EX: “192.168.1.2”
tid	in	TCP Transaction identifier , default 0
pid	In	TCP Protocol identifier , default 0
timeout	in	Waiting for response (sec).

Return Value:

-1	connection failed
> -1	socket handler (successful connection.)

Example:

```
int socket1;
socket1 = cMBTCPInit ("10.1.115.149",0,0,4) ;
```

To initialize the Modbus RTU connection :

```
int cMBRTUInit   ( char *deviceName,
                   unsigned long ulBaudRate,
                   unsigned char parity,
                   unsigned char dataBits,
                   unsigned char stopBit )
```

Parameter:

deviceName	in	COM port device name.
ulBaudRate	in	Connection baud rate
parity	in	Parity scheme 0 : no parity bit 1 : odd 2 : even
dataBits	in	specify transmission bits per byte
stopBit	in	The number of stop bits. 1 : 1 stop bit 2 : 2 stop bits

Return Value:

<=0	connection failed
> 0	Port handler (successful connection.)

Example:

```
Int handler = 0;
handler = cMBRTUInit("/dev/ttyS0", 115200,0,8,1)
```

2-2. Close Connection

To close the Modbus TCP connection :

void cMBTCPClose(int iSocket)

Parameter:

iSocket	in	socket handler

Example:

```
int socket1;
socket1 = cMBTCPInit ("10.1.115.149",0,0,4) ;
cMBTCPClose(socket1);
```

To close the Modbus RTU connection :

void cRTUClose(int iSerialFd)

Parameter:

iSerialFd	in	port handler

Example:

```
int iHd;
iHd = cMBRTUInit("/dev/ttyS34", 115200,0,8,1) ;
cRTUClose (iHd);
```

2-3. Function Code 01：Read Coils (0xxxx)

To read back DO settings from Modbus TCP device.

```
int mbTCP_R_Coils ( int s,
                    char sID,
                    int iStartAddr,
                    int bitCount,
                    unsigned char *iRecBuf,
                    int iRecBufLen );
```

To read back DO settings from Modbus RTU device.

```
int mbRTU_R_Coils (int iSerialFd,
                    char sID,
                    int iStartAddr,
                    int bitCount,
                    unsigned char *iRecBuf,
                    int iRecBufLen );
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It’s invalid for Modbus TCP.
iStartAddr	in	specify slot index and DO address. EX : to specify slot 4 and second DO point iStartAddr = 0x0401
bitCount	in	Contiguous DO bits for query

iRecBuf	out	receive buffer
iRecBufLen	in	receive buffer size

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]～
Device ID	Function code	Data length	DO ON / OFF status

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
int socket1, iRecvLen;
unsigned char iBuff[20];
if((socket1 = cMBTCPInit ("10.1.0.49",0,0,4))<0)
        exit(0);
iRecvLen = mbTCP_R_Coils ( socket1,
                           0x09,          // Modbus Server ID = 0x09
                           0x0603,        // slot 6, DO start from point 4th
                           2,             // Contiguous 2 DO points
                           &iBuff[0],
                           sizeof(iBuff) );
cMBTCPClose(socket1);
```

2-4. Function Code 02： Read Discrete Inputs (1xxxx)

To read DI status from Modbus TCP device.

```
int mbTCP_R_Discrete ( int s,
                       char sID,
                       int iStartAddr,
                       int bitCount,
                       unsigned char *iRecBuf,
                       int iBufLen);
```

.

To read DI status from Modbus RTU device.

```
int mbRTU_R_Discrete (int iSerialFd,
                      char sID,
                      int iStartAddr,
                      int bitCount,
                      unsigned char *iRecBuf,
                      int iBufLen);
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and DI address. EX : to specify slot 4 and second DI point iStartAddr = 0x0401
bitCount	in	Contiguous DI bits for query
iRecBuf	out	receive buffer
iBufLen	in	receive buffer size

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]～
Device ID	Function code	Data length	DO ON / OFF status

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
Int iRecvLen,handler;
unsigned char iBuff[20];
if((handler =cMBRTUInit("/dev/ttyS34", 115200,0,8,1) )<0)
    exit(0);
iRecvLen = mbRTU_R_Discrete ( handler ,
                              0x07,          // Modbus Server ID = 0x07
                              0x0204,        // slot 2, DI start from point 5th
                              0xf,           // Contiguous 15 DI points
                              &iBuff[0],
                              sizeof(iBuff) );
```

2-5. Function Code 03：Read Holding Registers (4xxxx)

To read back AO settings from Modbus TCP device.

```
int mbTCP_R_Hold_Registers ( int s,
                             char sID,
                             int iStartAddr,
                             int bitCount,
                             unsigned char *iRecBuf,
                             int iBufLen   );
```

.

To read back AO settings from Modbus RTU device.

```
int mbRTU_R_Hold_Registers (int iSerialFd,
                            char sID,
                            int iStartAddr,
                            int bitCount,
                            unsigned char *iRecBuf,
                            int iBufLen );
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and AO address. EX : to specify slot 9 and first AO point iStartAddr = 0x0900
bitCount	in	Contiguous AO bits for query
iRecBuf	out	receive buffer [*]
iBufLen	in	receive buffer size

[*] *Real value on the I/O module = (Received AO value) / 1000*
Minus value (>0x7FFF) is presented in 2's Complement.

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]～
Device ID	Function code	Data length	AO values two bytes / pre AO register

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
int socket1, iRecvLen;
unsigned char iBuff[20];
if((socket1 = cMBTCPInit ("10.1.0.49",0,0,4))<0)
        exit(0);
iRecvLen = mbTCP_R_Hold_Registers (
                                socket1,
                                0x02,           // Modbus Server ID = 0x02
                                0x0102,         // slot 1, AO start from point 3rd
                                2,              // Contiguous 2 AO points
                                &iBuff[0],
                                sizeof(iBuff) );
cMBTCPClose(socket1);
```

2-6. Function Code 04：Read Input Registers (3xxxx)

To read AI values from Modbus TCP device.

```
int mbTCP_R_Registers ( int s,
                        char sID,
                        int iStartAddr,
                        int bitCount,
                        unsigned char *iRecBuf,
                        int iBufLen   );
```

.

To read AI values from Modbus RTU device.

```
int mbRTU_R_Registers ( int iSerialFd,
                        char sID,
                        int iStartAddr,
                        int bitCount,
                        unsigned char *iRecBuf,
                        int iBufLen );
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and AI address. EX : to specify slot 4 and second AI point iStartAddr = 0x0401
bitCount	in	Contiguous AI bits for query
iRecBuf	out	receive buffer [*]
iBufLen	in	receive buffer size

[*] *Real value on the I/O module = (Received AO value) / 1000*
Minus value (>0x7FFF) is presented in 2's Complement.

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]～
Device ID	Function code	Data length	AI values ,two bytes / pre AI register

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
Int iRecvLen, handler;
unsigned char iBuff[20];
if((handler =cMBRTUInit("/dev/ttyS0", 115200,0,8,1) )<0)
    exit(0);
iRecvLen = mbRTU_R_Registers (handler ,
                              0x07,          // Modbus Server ID = 0x07
                              0x0402,        // slot 4, AI start from point 3rd
                              0x08,          // Contiguous 8 AI points
                              &iBuff[0], sizeof(iBuff) );
cRTUClose (handler);
```

2-7. Function Code 05：Write Single Coil (0xxxx)

Writing the ON/OFF status to a single DO point through Modbus TCP.

```
int mbTCP_W_Coil ( int s,
                   char sID,
                   int iStartAddr,
                   char OnOff,
                   unsigned char *iRecBuf,
                   int iBufLen);
```

Writing the ON/OFF status to a single DO point through Modbus RTU.

```
int mbRTU_W_Coil ( int iSerialFd,
                   char sID,
                   int iStartAddr,
                   char OnOff,
                   unsigned char *iRecBuf,
                   int iBufLen);
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and DO address. EX : to specify slot 4 and second DO point iStartAddr = 0x0401
OnOff	in	SET_OFF(0)／SET_ON(1)
iRecBuf	out	receive buffer
iBufLen	in	receive buffer size

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]	iRecBuf[4]~[5]
Device ID	Function code	Slot index	Start Addr.	Eched value

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
int socket1, iRecvLen;
unsigned char iBuff[20];
if((socket1 = cMBTCPInit ("10.1.0.49",0,0,4))<0)
        exit(0);
iRecvLen = mbTCP_W_Coil ( socket1,
                          0x09,          // Modbus Server ID = 0x09
                          0x0603,        // slot 6, DO start from point 4th
                          SET_ON,
                          &iBuff[0],
                          sizeof(iBuff) );
cMBTCPClose(socket1);
```

2-8. Function Code 06：Write Single Registers (4xxxx)

Writing a single AO to Modbus TCP device.

```
int mbTCP_W_Register ( int s,
                       char sID,
                       int iStartAddr,
                       int regValue,
                       unsigned char *iRecBuf,
                       int iBufLen );
```

Writing a single AO to Modbus RTU device.

```
int mbRTU_W_Register ( int iSerialFd,
                       char sID,
                       int iStartAddr,
                       int regValue,
                       unsigned char *iRecBuf,
                       int iBufLen );
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and AO address. EX : to specify slot 4 and second AO point iStartAddr = 0x0401
regValue	in	regValue = real value * 1000 EX : Seting 1.234V to specified AO point, Then regValue will be 1234(d) = 0x4D2(h)
iRecBuf	out	receive buffer
iBufLen	in	receive buffer size

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]	iRecBuf[4]~[5]
Device ID	Function code	Slot index	Start Addr.	Eched value

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
int socket1, iRecvLen;
unsigned char iBuff[20];
if((socket1 = cMBTCPInit ("10.1.0.49",0,0,4))<0)
        exit(0);
iRecvLen = mbTCP_W_Coil ( socket1,
                          0x09,          // Modbus Server ID = 0x09
                          0x0603,        // slot 6, AO start from point 4th
                          0x1388,        // setting 5V to specified AO.
                          &iBuff[0],
                          sizeof(iBuff) );
cMBTCPClose(socket1);
```

2-9. Function Code 15：Write Multiple Coils (0xxxx)

Writing the ON/OFF status to contiguous DOs through Modbus TCP.

```
Int mbTCP_W_Multi_Coils ( int s,
                          char sID,
                          int iStartAddr,
                          int bitCount,
                          int byteCount,
                          unsigned char *wData,
                          int wDataLen);
```

Writing the ON/OFF status to contiguous DOs through Modbus RTU.

```
Int mbRTU_W_Multi_Coils ( int iSerialFd,
                          char sID,
                          int iStartAddr,
                          int bitCount,
                          int byteCount,
                          unsigned char *wData,
                          int wDataLen);
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and DO address. EX : to specify slot 4 and second DO point iStartAddr = 0x0401
bitCount,	in	contiguous DO numbers.
byteCount,	in	data length, = (bitCount+7) / 8
*wData	in out	(in) setting values (out) receive buffer
wDataLen	in	wData buffer size

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]	iRecBuf[4]~[5]
Device ID	Function code	Slot index	Start Addr.	DO Bit Count

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
Int iRecvLen, handler;
unsigned char ioBuff[20];
if((handler =cMBRTUInit("/dev/ttyS0", 115200,0,8,1)) <0)
    exit(0);
ioBuff[0] = 0x2A;
iRecvLen = mbRTU_W_Multi_Coils (handler,
                                0x09,        // Modbus Server ID = 0x09
                                0x0402,      // slot 4, DO start from point 3rd
                                0x06,        // Contiguous 6 DO points
                                & ioBuff [0],
                                sizeof(ioBuff) );
cRTUClose (handler);
```

Above source code indicates DO values as following :

-	-	DO#8	DO#7	DO#6	DO#5	DO#4	DO#3
-	-	ON (1)	OFF (0)	ON (1)	OFF (0)	ON (1)	0FF (0)

2-10. Function Code 16：Write Multiple Registers (4xxxx)

Writing data to contiguous AOs through Modbus TCP.

```
int mbTCP_W_Multi_Registers (   int s,
                                char sID,
                                int iStartAddr,
                                int regCount,
                                unsigned char *iRecBuf,
                                int iBufLen );
```

Writing data to contiguous AOs through Modbus RTU.

```
int mbTCP_W_Multi_Registers ( int iSerialFd,
                              char sID,
                              int iStartAddr,
                              int regCount,
                              unsigned char *iRecBuf,
                              int iBufLen );
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and AO address. EX : to specify slot 4 and second AO point iStartAddr = 0x0401
regCount	in	contiguous AO numbers.
iRecBuf	in out	(in) setting values [*1] (out) receive buffer [*2]
iBufLen	in	receive buffer size

[*1] setting value = real value * 1000, and 2 bytes data for one register.
EX : Specified 1.234V to AO #3,

Then input value will be 1234(d) = 0x04D2(h)

[*2] Real value on the I/O module = (Received AO value) / 1000
Minus value (>0x7FFF) is presented in 2's Complement.

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]	iRecBuf[4]~[5]
Device ID	Function code	Slot index	Start Addr.	Contiguous AO Count

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Example:

```
Int iRecvLen, handler;
unsigned char ioBuff[20];
if((handler =cMBRTUInit("/dev/ttyS0", 115200,0,8,1) )<0)
    exit(0);
ioBuff[0] = 0x13;
ioBuff[1] = 0x88;
ioBuff[0] = 0xEC;
ioBuff[1] = 0x78;
iRecvLen = mbRTU_W_Multi_Registers (handler ,
                                    0x09,          // Modbus Server ID = 0x09
                                    0x0402,        // slot 4, AO start from point 3rd
                                    0x02,          // Contiguous 2 AO points
                                    & ioBuff [0],
                                    sizeof(ioBuff) );
cRTUClose (handler);
```

Above source code indicates AO values as following :

assign -5V to AO #4		assign +5V to AO #3	
0xEC	0x78	0x13	0x88

2-11. Read Range Code from ICP DAS AI/O Modules

Reading range code from Modbus TCP device.

```
int mbRTU_R_RangeCode ( int s,
                        char sID,
                        int iStartAddr,
                        unsigned char *iRecBuf,
                        int iBufLen );
```

Reading range code from Modbus RTU device.

```
int mbRTU_R_RangeCode ( int iSerialFd,
                        char sID,
                        int iStartAddr,
                        unsigned char *iRecBuf,
                        int iBufLen );
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and AO address. EX : to specify slot 4 and second AO point iStartAddr = 0x0401
iRecBuf	out	receive buffer [*]
iBufLen	in	receive buffer size

[*] Analog I/O modules defined various range code (type code) for different purposes. More detail definition, please refer to following website : http://ftp.icpdas.com/pub/cd/8000cd/napdos/dcon/io_module/dcon/8k87k/modules/typetable.htm

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]	iRecBuf[4]	iRecBuf[5]
Device ID	Function code	Sub Code	Slot index	Start Addr.	Range code

※ For Modbus TCP, there are 6 bytes of prefixed in front.

2-12. Set Range Code to ICP DAS AI/O Modules

Setting range code to AI/O modules through Modbus TCP.

```
int mbTCP_W_RangeCode( int s,
                       char sID,
                       int iStartAddr,
                       char rangeCode,
                       unsigned char *iRecBuf,
                       int iBufLen ) ;
```

Setting range code to AI/O modules through Modbus RTU.

```
int mbRTU_W_RangeCode( int iSerialFd,
                       char sID,
                       int iStartAddr,
                       char rangeCode,
                       unsigned char *iRecBuf,
                       int iBufLen ) ;
```

Parameter:

s	in	socket handler
iSerialFd	in	Port handler
sID	in	device ID of Modbus server , It's invalid for Modbus TCP.
iStartAddr	in	specify slot index and AO address. EX : to specify slot 4 and second AO point iStartAddr = 0x0401
rangeCode	in	AI/O range code. [*]
iRecBuf	out	receive buffer
iBufLen	in	receive buffer size

[*] Analog I/O modules defined various range code (type code) for different purposes. More detail definition, please refer to following website : http://ftp.icpdas.com/pub/cd/8000cd/napdos/dcon/io_module/dcon/8k87k/modules/typetable.htm

Return Value:

<=0	error occurred. (refer to Appendix A)
>0	Bytes of received data.

iRecBuf[0]	iRecBuf[1]	iRecBuf[2]	iRecBuf[3]	iRecBuf[4]	iRecBuf[5]
Device ID	Function code	Sub Code	Slot index	Start Addr.	Range code

※ For Modbus TCP, there are 6 bytes of prefixed in front.

Appendix A

A1. Error Code

0	no error
-1	illegal register address
-2	illegal argument
-3	porting layer error
-4	insufficient resources
-5	I/O error
-6	protocol stack in illegal state
-7	timeout error occurred
-8	Buffer write error !