

Ethernet開発支援ツール

ACCEL ti2k-180829 (LastUpdate Jul 31, 2019)

概要

ターミナルツール(1) TcpTermS.exe

ターミナルツール(2) TcpTermM.exe

ターミナルツール(3) UdpTermS.exe

ターミナルツール(4) UdpTermM.exe

モニターツール(1) TcpMon.exe

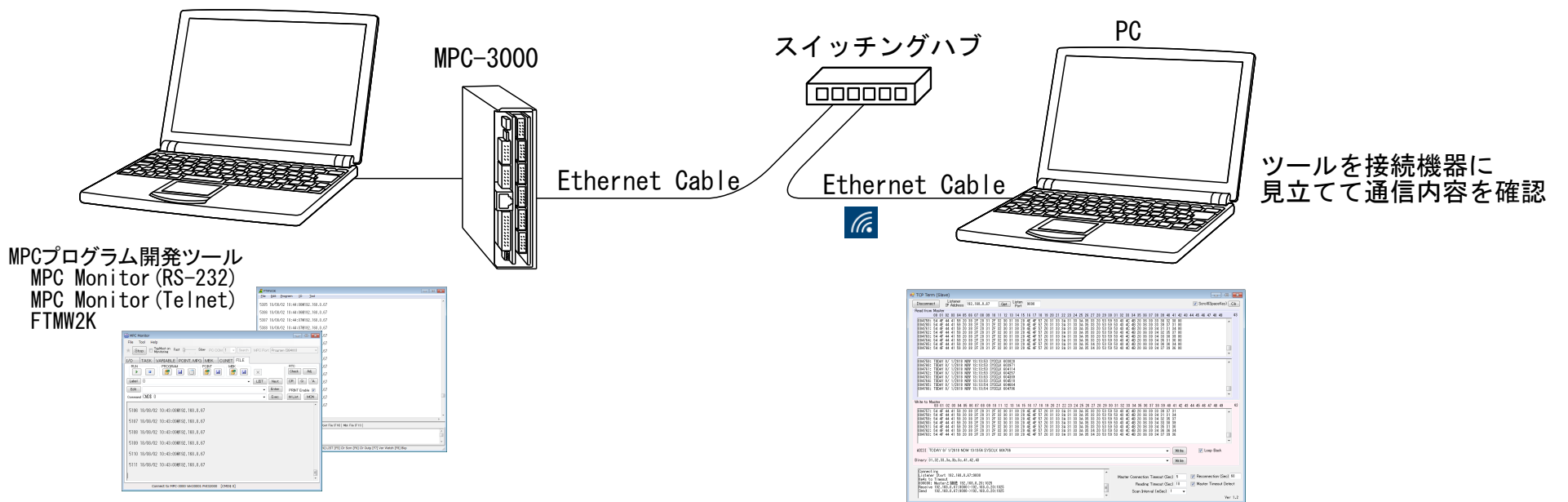
モニターツール(2) UdpMon.exe

自己IP Addressの設定例

※改版により、実際のアプリケーションと画像が異なる場合があります。

■ 概要

• ターミナルツール



ツールを接続機器に見立てて通信内容を確認

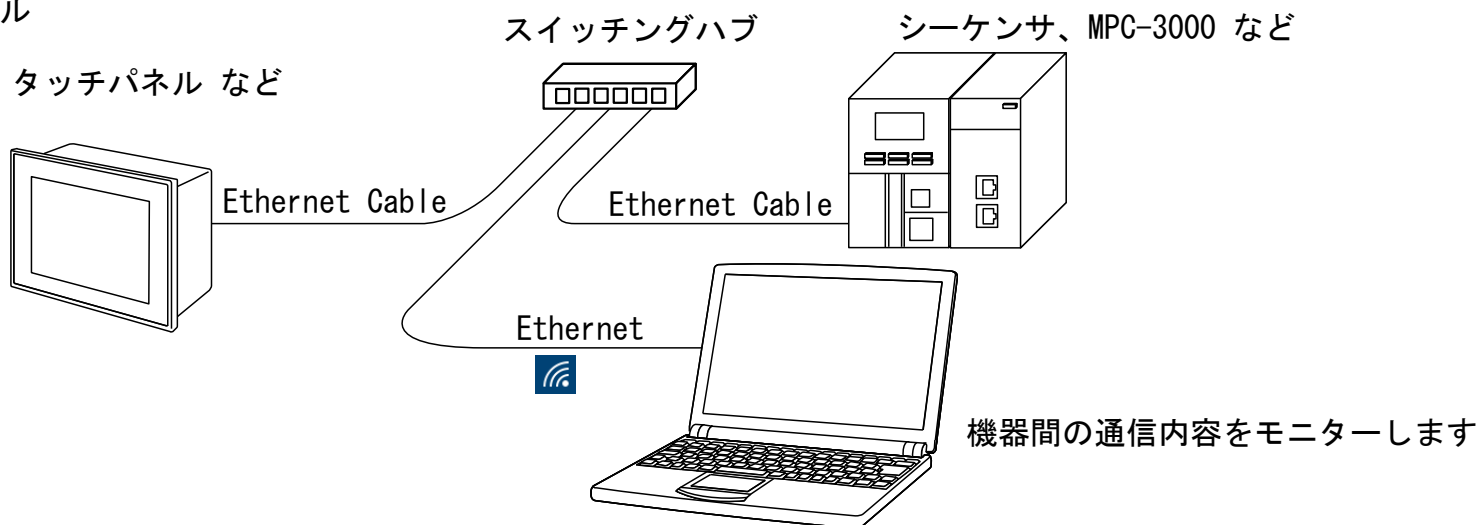
MPCの設定例

開発支援ツールの実行ファイル名

```
[Master] SET_IP TCP PACKET 9000 192 168 0 67 ← TcpTermS.exe TCPターミナル[Slave]
[Slave] SET_IP TCP PACKET 9000 → TcpTermM.exe TCPターミナル[Master]

[Master] SET_IP UDP PACKET 9000 192 168 0 67 ← UdpTermS.exe UDPターミナル[Slave]
[Slave] SET_IP UDP PACKET 9000 → UdpTermM.exe UDPターミナル[Master]
```

• モニターツール



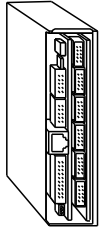
機器間の通信内容をモニターします

ツールの実行ファイル名

```
TcpMon.exe TCPモニター
UdpMon.exe UDPモニター
```

■ ターミナルツール(1) TcpTermS.exe
TCPでMPCがMaster、PCがSlave

[Master] MPC

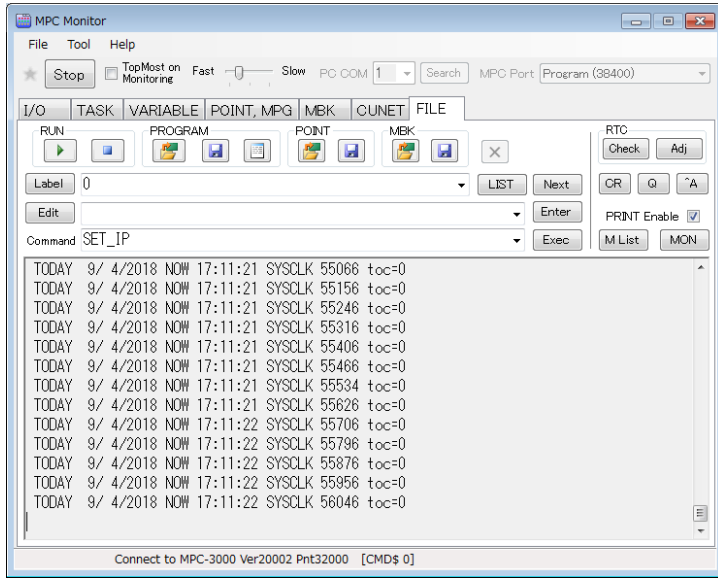


```
#SET_IP
IPAdrs = 192.168.0.20
MASK = 255.255.255.0
Gateway= 192.168.0.248
OFF Delay ACK baud=115200
MACAdrs= 001EC08DAB6E
```

← 設定内容を確認するコマンド
← MPCのIP Address

```
1 : / TCP Telnet
2 : = TCP PACKET
#
```

23 Slave connected-> 192.168.0.53
9000 <TCP_XO\$><TCP_RO\$> Master 192.168.0.53 ref_time = 50
PCのPort 送信・受信コマンド PCのIP Address



(MPC Monitor (RS-232))

■ MPCプログラム例

```
SET_IP TCP Telnet
SET_IP 192 168 0 20 255 255 255 0 192 168 0 248
SET_IP TCP PACKET 9000 192 168 0 53 /* Define as the Master
```

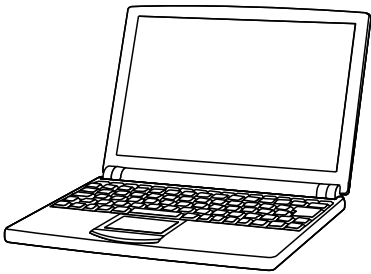
```
toc=0
SYSCLK=0
DO
TCP_RO$=""
TCP_XO$="TODAY "+DATE$(1)+" NOW "+TIME$(1)+" SYSCLK "+STR$(SYSCLK)+" toc="+STR$(toc)
timer:=20
WAIT (TCP_RO$<"") | (timer==0)
IF timer==0 THEN
TCP_XO$="Timeout"
PR "Timeout" toc
toc=toc+1
END_IF
st$=TCP_RO$
PR st$
TIME 10
LOOP
```

■ 実行例

PC→MPCループバック文字列

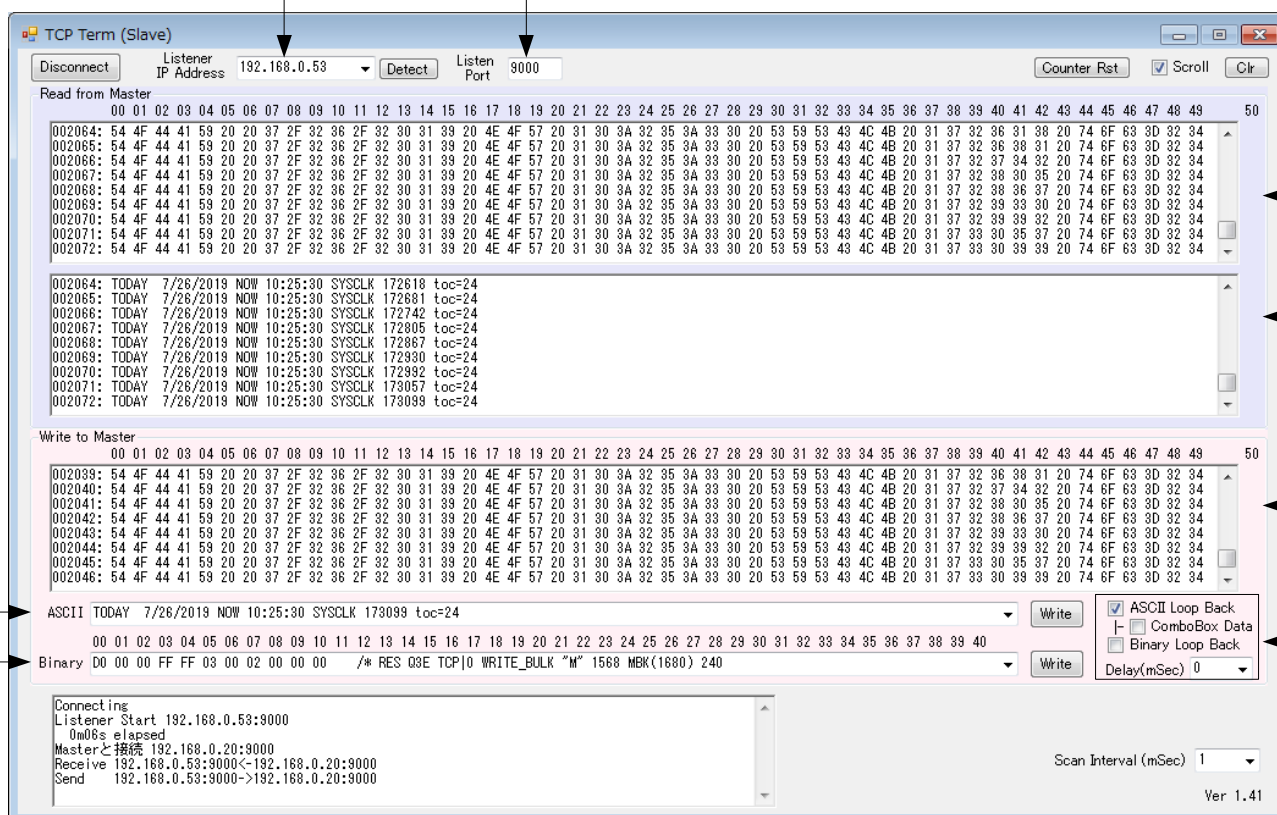
```
TODAY 9/ 4/2018 NOW 17:12:40 SYSCLK 133866 toc=0
TODAY 9/ 4/2018 NOW 17:12:40 SYSCLK 133956 toc=0
TODAY 9/ 4/2018 NOW 17:12:40 SYSCLK 134016 toc=0
TODAY 9/ 4/2018 NOW 17:12:40 SYSCLK 134090 toc=0
TODAY 9/ 4/2018 NOW 17:12:40 SYSCLK 134176 toc=0
```

[Slave] PC



TcpTermS.exe
IP Address 192.168.0.53
Port 9000

自己 IP Address 受信ポート



MPCから送信された文字列

Binary表示

ASCII表示

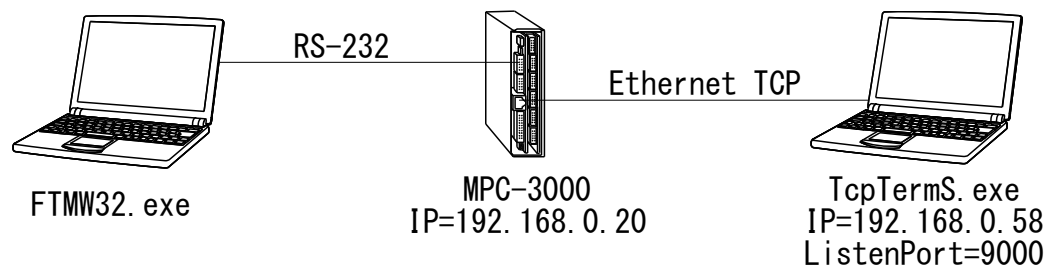
MPCへ送信した文字列

受信文字列をMPCへループバック
ComboBox Dataをチェックすると
左のComboBoxのTextを返信

送信する文字列
ASCII
Binary

• TcpTermS.exe 使用例

TcpTermSを画像処理装置に見立て、MPC-3000のINPUT_TCPコマンドの動作確認をしています。
 この装置は、MPCからMコマンド受信 → MPCにOK送信 → 計測 → MPCに結果送信 (CR区切りで複数ある) という動作をします。
 OKから結果送信までインターロックは無くタイミングも不定です。また結果が複数のパケットになることもあります。
 (確認:MPC-3000 (SH2A) BL/I 2.01_23 2019/02/28)



MPC-3000の設定
 SET_IP 192 168 0 20 255 255 255 0 192 168 0 248
 SET_IP TCP PACKET 9000 PC 192 168 0 58

FTMW2K画面

```

(1) #INPUT_TCP 0 CLR_BUF
(2) #TCP_XO$="M"
(3) #INPUT_TCP 0 EOL|13 TMOUT|10 a$
    #PR a$
    OK
(4) #INPUT_TCP 0 EOL|13 TMOUT|10 a$
    #PR a$
    12345
(5) #INPUT_TCP 0 EOL|13 TMOUT|10 a$
    #PR a$
    67890
(6) #INPUT_TCP 0 CLR_BUF
(7) #TCP_XO$="M"
(8) #INPUT_TCP 0 EOL|13 TMOUT|10 a$
    #PR a$
    OK
(9) #INPUT_TCP 0 CLR_BUF
(10) #TCP_XO$="M"
    #INPUT_TCP 0 EOL|13 TMOUT|10 a$
    #PR a$
    12345
    #INPUT_TCP 0 EOL|13 TMOUT|10 a$
    #PR a$
    67890
    #INPUT_TCP 0 CLR_BUF
(11) #TCP_XO$="M"
(12) #TCP_XO$="M" 受信バッファが空じゃないのでこの M は出ない
    #
    
```

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 1

```

000029: 4D
000030: 4D
000031: 4D
000032: 4D
    
```

000029: M
 000030: M
 000031: M
 000032: M

Write to Master

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 1

```

000023: 4F 4B 0D 31 32 33 34 35 0D 36 37 38 39 30 0D
000024: 4F 4B 0D 31 32 33 34 35 0D 36 37 38 39 30 0D
000025: 4F 4B 0D 31 32 33 34 35 0D 36 37 38 39 30 0D
000026: 4F 4B 0D 31 32 33 34 35 0D 36 37 38 39 30 0D
    
```

ASCII OK~OD12345~OD67890~OD (送信する文字列。~ODは CR)

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 1

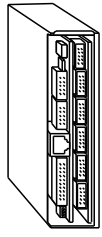
Binary D0 00 00 FF FF 03 00 08 00 00 00 02 04 2E 16 B8 22

Connecting
 Listener Start 192.168.0.58:9000
 0m02s elapsed

- (1) 受信バッファクリア
- (2) M送信。TcpTermSからMを受信後 OK<CR>12345<CR>567890<CR> と送信します。
- (3) 既に受信バッファに文字列が入っているのでINPUT_TCPは抜けてa\$に入ります (<CR>は入りません)。
- (4) 直ぐに抜けてa\$に12345が入ります (<CR>は入りません)。
- (5) 直ぐに抜けてa\$に567890が入ります (<CR>は入りません)。
- (6) (7) (8)は上と同様の手順
- (9) まだ受信バッファに2文字列残っていますがクリアします。
- (10) Mが送信されます。
- (11) (12) バッファに残留物があると送信できません。

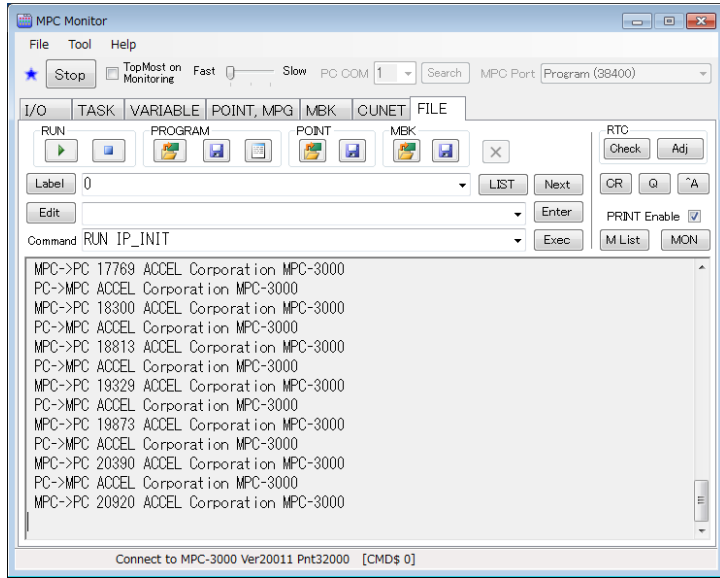
■ ターミナルツール(2) TcpTermM.exe
TCPでMPCがSlave、PCがMaster

[Slave] MPC



```
#SET_IP
IPadrs = 192.168.0.18
MASK = 255.255.255.0
Gateway= 192.168.0.248
OFF Delay ACK baud=115200
MACadrs= 001EC08DAB6E
```

```
1 : / TCP Telnet 23 Slave
2 : = TCP PACKET 9000 <TCP_XO$><TCP_RO$> Slave connected-> 192.168.0.53
# MPCのPort 送信・受信コマンド PCのIP Address
```



(MPC Monitor (RS-232))

■ MPCプログラム例

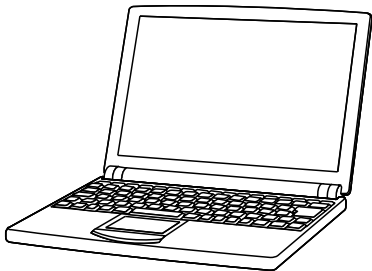
```
SET_IP 192 168 0 18 255 255 255 0 192 168 0 248
SET_IP TCP Telnet
SET_IP TCP PACKET 9000 /* Define as the Slave
```

```
TCP_XO$=""
TCP_RO$=""
DO
WAIT TCP_RO$<>""
st$=TCP_RO$
PR "PC->MPC" st$
TCP_RO$=""
st$=STR$(SYSCLK)+" "+st$
PR "MPC->PC" st$
TCP_XO$=st$ /* Reply to the Master
WAIT TCP_XO$=""
LOOP
```

■ 実行例

```
PC->MPC ACCEL Corporation MPC-3000
MPC->PC 5605667 ACCEL Corporation MPC-3000
PC->MPC ACCEL Corporation MPC-3000
MPC->PC 5606183 ACCEL Corporation MPC-3000
```

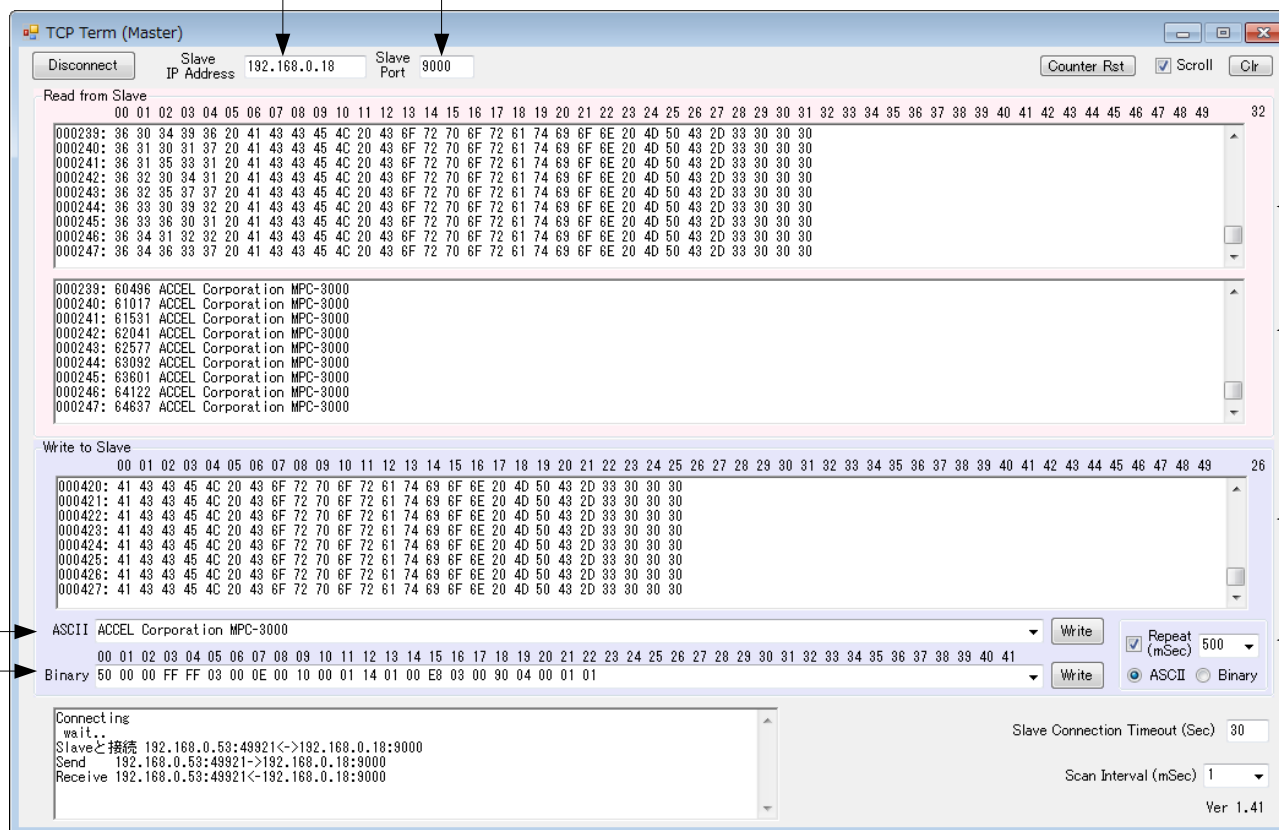
[Master] PC



TcpTermM.exe
IP Address 自動
Port 自動

MPCの IP Address MPCのポート

送信する文字列
ASCII
Binary



MPCから送信された文字列

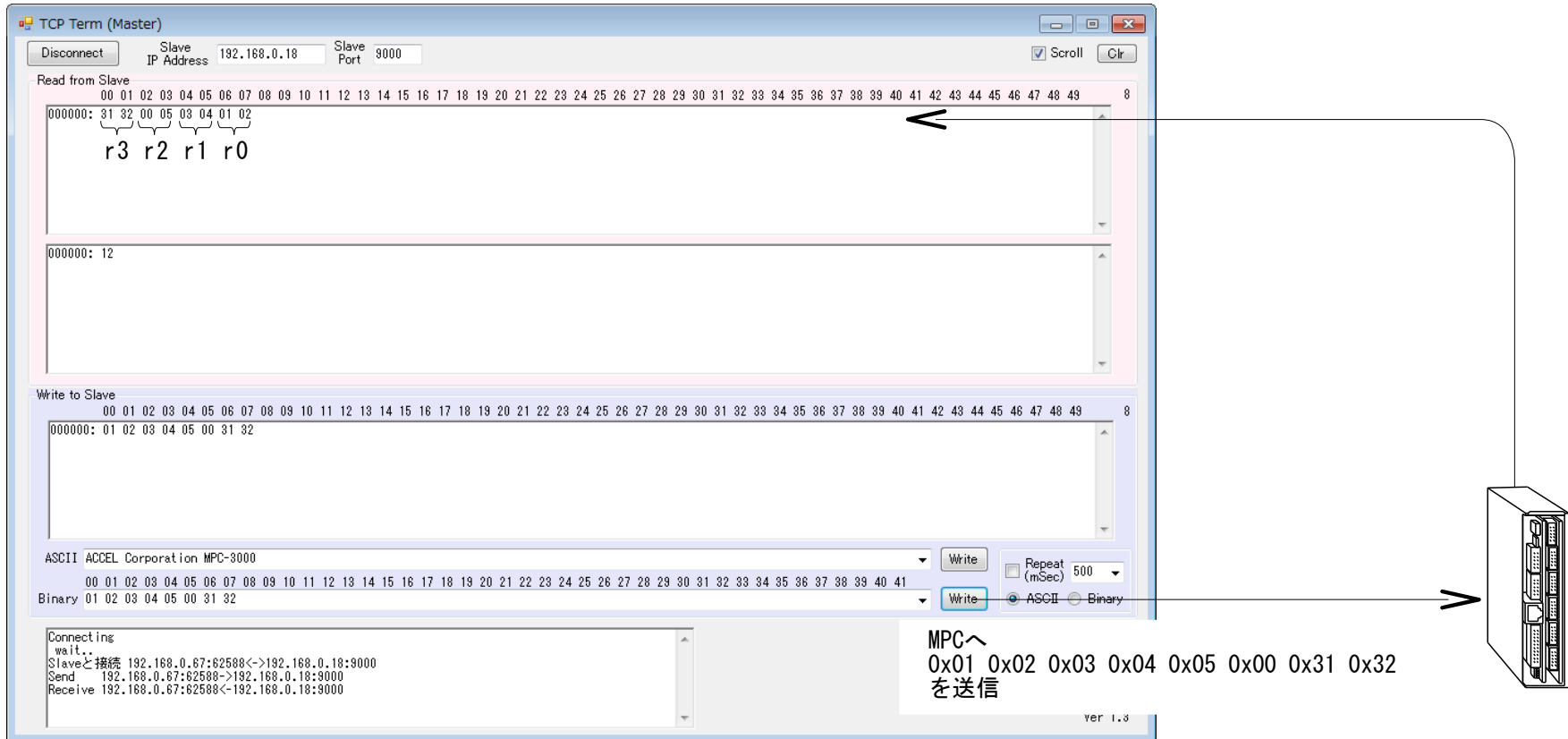
Binary表示

ASCII表示

MPCへ送信した文字列

繰り返し送信

- TcpTermM.exe 使用例1
PCからMPCへバイナリ送信 → MPCはバイナリ受信処理 → PCへPACKET送信



■ MPCプログラム

```

/* PCから文字列を受信したらバイナリデータとして変数に入れ、PCに送り返す
SET_IP 192 168 0 18 255 255 255 0 192 168 0 248
SET_IP TCP PACKET 9000 /* Master (PC) -> Slave MPC:9000

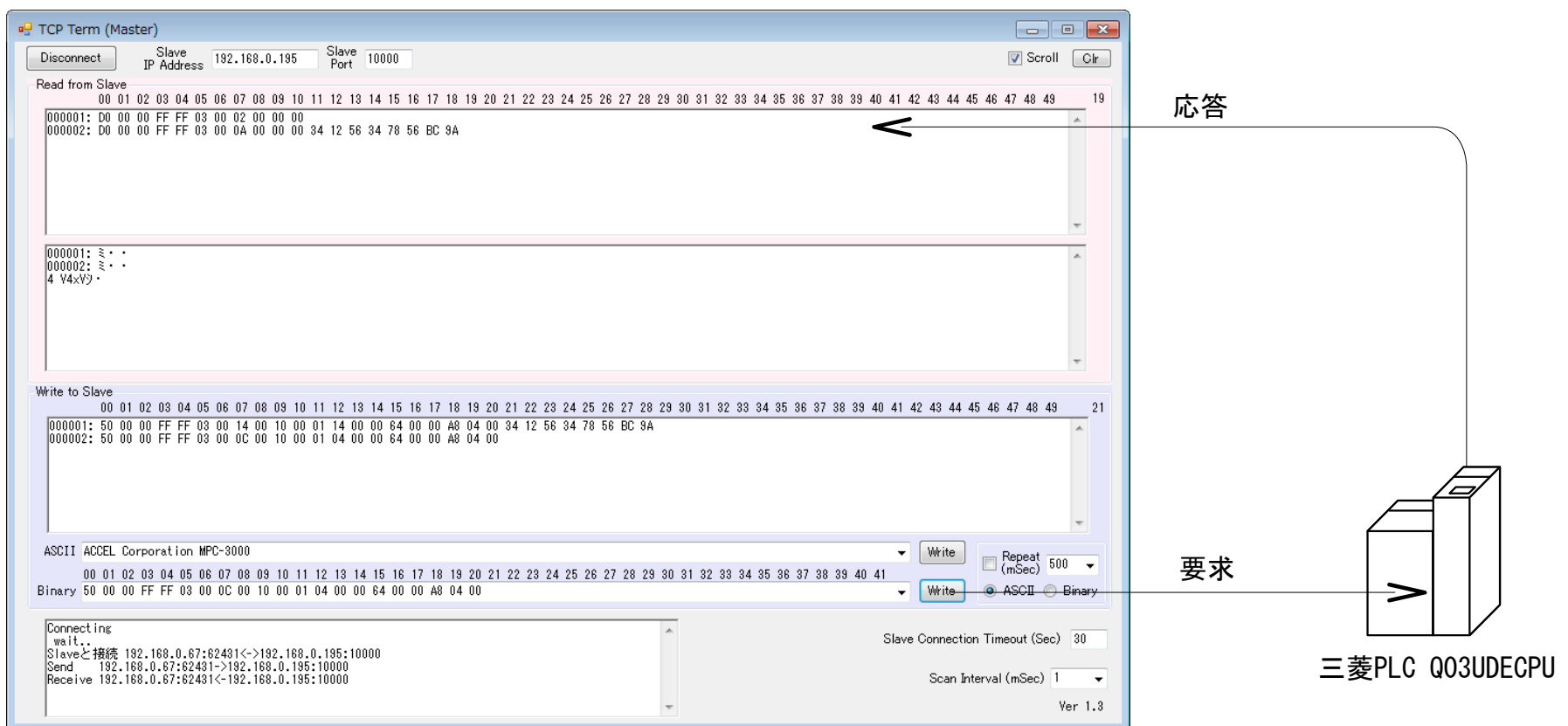
D0
TCP_R0$=""
WAIT TCP_R0$<>"" /* WAIT文:文字列先頭が0x00はNG
ptr =TCP_R0$ /* prt_は受信文字列のポインタ
r0=PTR(2,0) /* 0番目(先頭)から2バイト(ワード)取得
PR "r0=" HEX$(r0) r0
r1=PTR(4,0) /* 0番目(先頭)から4バイト(ロング)取得
PR "r1=" HEX$(r1) r1
r2=PTR(-2,4) /* 4番目から2バイト(ワード)取得、上位下位スワップ
PR "r2=" HEX$(r2) r2
r3=PTR(2,6) /* 6番目から2バイト(ワード)取得
PR "r3=" HEX$(r3) r3
PACKET TCP_XO$ r3 r2 r1 r0 /* 送信するのは下位2バイト
LOOP
  
```

■ 実行結果

```

r0= 00000102 258 ←HEX DEC
r1= 01020304 16909060
r2= 00000005 5
r3= 00003132 12594
  
```

- TcpTermM.exe 使用例2
PC→PLC MCプロトコル



■ PLCのD100~D103に 0x1234, 0x3456 0x5678 0x9ABC を書き込む

```

要求: 50 00 00 FF FF 03 00 14 00 10 00 01 14 00 00 64 00 00 A8 04 00 34 12 56 34 78 56 BC 9A
応答: D0 00 00 FF FF 03 00 02 00 00 00
  
```

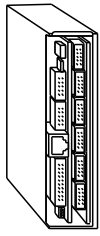
■ PLCのD100~D103を読み出す

```

要求: 50 00 00 FF FF 03 00 0C 00 10 00 01 04 00 00 64 00 00 A8 04 00
応答: D0 00 00 FF FF 03 00 0A 00 00 00 34 12 56 34 78 56 BC 9A
  
```

■ ターミナルツール(3) UdpTermS.exe
UDPでMPCがMaster、PCがSlave

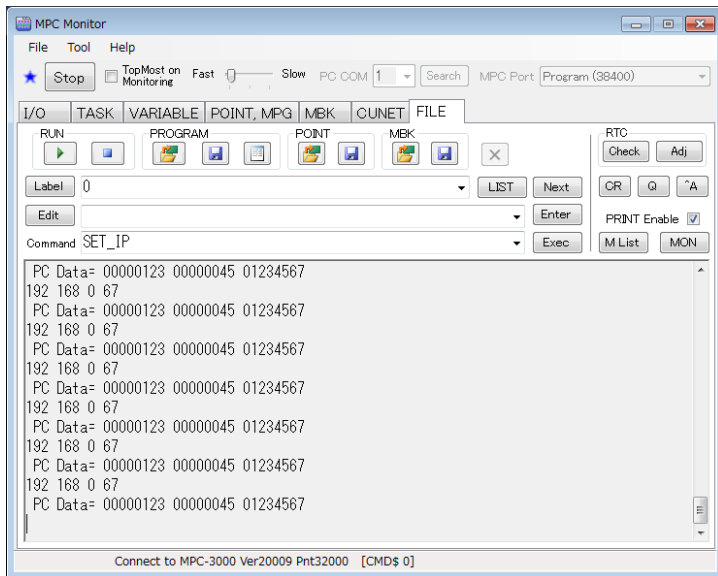
[Master] MPC



```
#SET_IP
IPAdrs = 192.168.0.20
MASK = 255.255.255.0
Gateway= 192.168.0.248
OFF Delay ACK
MACAdrs= 001EC08DAB6E
```

1 : / UDP PACKET

61440 <UDP_XO\$><UDP_RO\$>
PCのPort 送受信に用いる予約文字列変数



■ MPCプログラム例

```
SET_IP 192 168 0 20 255 255 255 0 192 168 0 248 /* 自己IP設定
SET_IP UDP PACKET 61440 /* UDP Port番号

IP_CONV 192 168 0 53 PcIp /* 変数 PcIp にPCのIP 192.168.0.53 を入れる
SET_DEST OppdIp UDP_XO$ /* 宛先のIPと送信文字列変数 UDP_XO$ を紐付け。
DO
DO
UDP_RO$=""
SET_DEST PcIp UDP_XO$ /* 宛先設定
PACKET UDP_XO$ &h0123 &h4567 /* 送信
timer_=10
WAIT (IPC(UDP_RO$) !=0) | (timer_==0) /* 応答待ち
IF timer_==0 THEN
PR "TIME OUT PC"
ELSE
BREAK
END_IF
LOOP
IP_CONV IPA(UDP_RO$) /* 送信元IP表示
ptr_=UDP_RO$ /* PC から [0]0x01 [1]0x23 [2]0x45 [3]0x67 と来る
a1=PTR(2,0) /* [0] 番目から Word 読み込み
a2=PTR(1,2) /* [2] 番目から Byte 読み込み
a3=PTR(4,0) /* [0] 番目から Long 読み込み
PRINT "PC Data=" HEX$(a1) HEX$(a2) HEX$(a3)

LOOP
```

■ 実行例

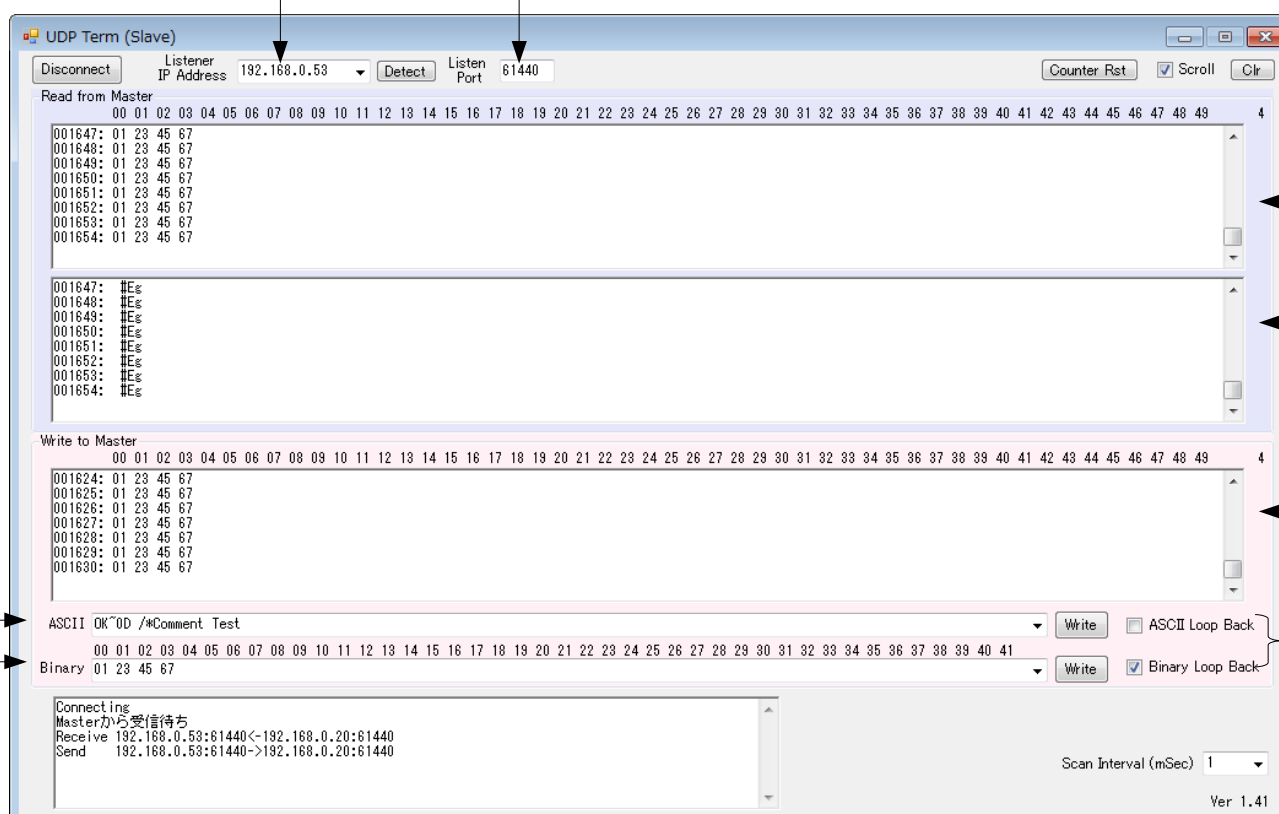
```
192 168 0 53
PC Data= 00000123 00000045 01234567
192 168 0 53
PC Data= 00000123 00000045 01234567
```

[Slave] PC



UdpTermS.exe
IP Address 192.168.0.53
Port 61440

自己 IP Address 受信ポート

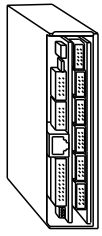


MPCから送信された文字列
Binary表示
ASCII表示
MPCへ送信した文字列
受信文字列をMPCへループバック

送信する文字列
ASCII
Binary

■ ターミナルツール(4) UdpTermM.exe
UDPでMPCがSlave、PCがMaster

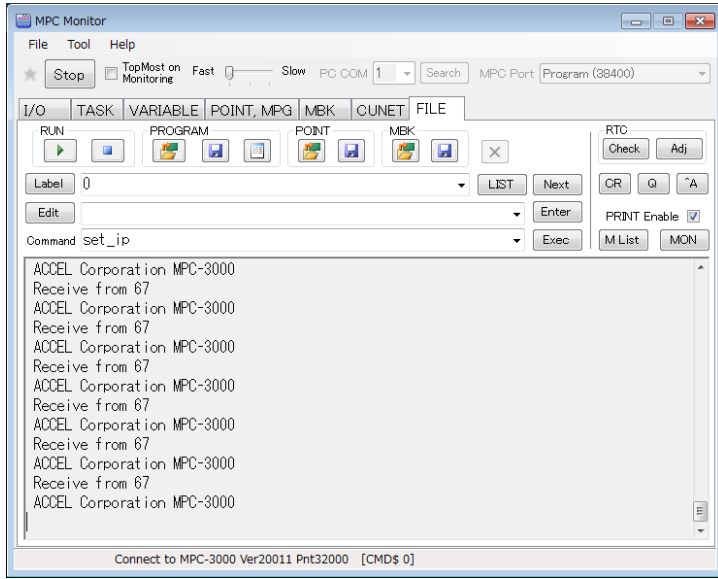
[Slave] MPC



```
#SET_IP
IPAdrs = 192.168.0.20
MASK = 255.255.255.0
Gateway= 192.168.0.248
OFF Delay ACK baud=115200
MACAdrs= 001EC08DAB6E
```

```
1 : / UDP PACKET
```

61440 <UDP_XO\$><UDP_RO\$>
MPCのPort 送受信に用いる予約文字列変数



(MPC Monitor (RS-232))

■ MPCプログラム例

/* 受信だけなら※行は不要

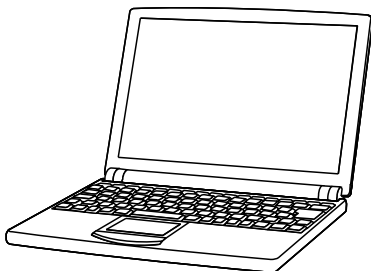
```
SET_IP 192 168 0 20 255 255 255 0 192 168 0 248 /* 自己IP設定
SET_IP UDP PACKET 61440 /* UDP Port番号
IP_CONV 192 168 0 53 PcIp /* 変数 PcIp にPCのIP 192.168.0.53 を入れる ※
SET_DEST PcIp UDP_XO$ /* 宛先設定 ※
```

```
DO
UDP_RO$=""
WAIT IPC(UDP_RO$) != 0 /* 受信待ち
ipa=IPA(UDP_RO$) &&hFF /* 送信元IPアドレスの最下位
PR "Receive from" ipa /* 受信文字列長
len=IPC(UDP_RO$)
ptr =UDP_RO$
rcv$=PTR$(len)
PR rcv$
UDP_XO$=STR$(SYSCLK)+" "+rcv$ /*SYSCLK値と受信文字列をPCに送信 ※
LOOP
```

■ 実行例

Receive from 53 ← PCのIP Address 最下位
ACCEL Corporation MPC-3000 ← 受信した文字列

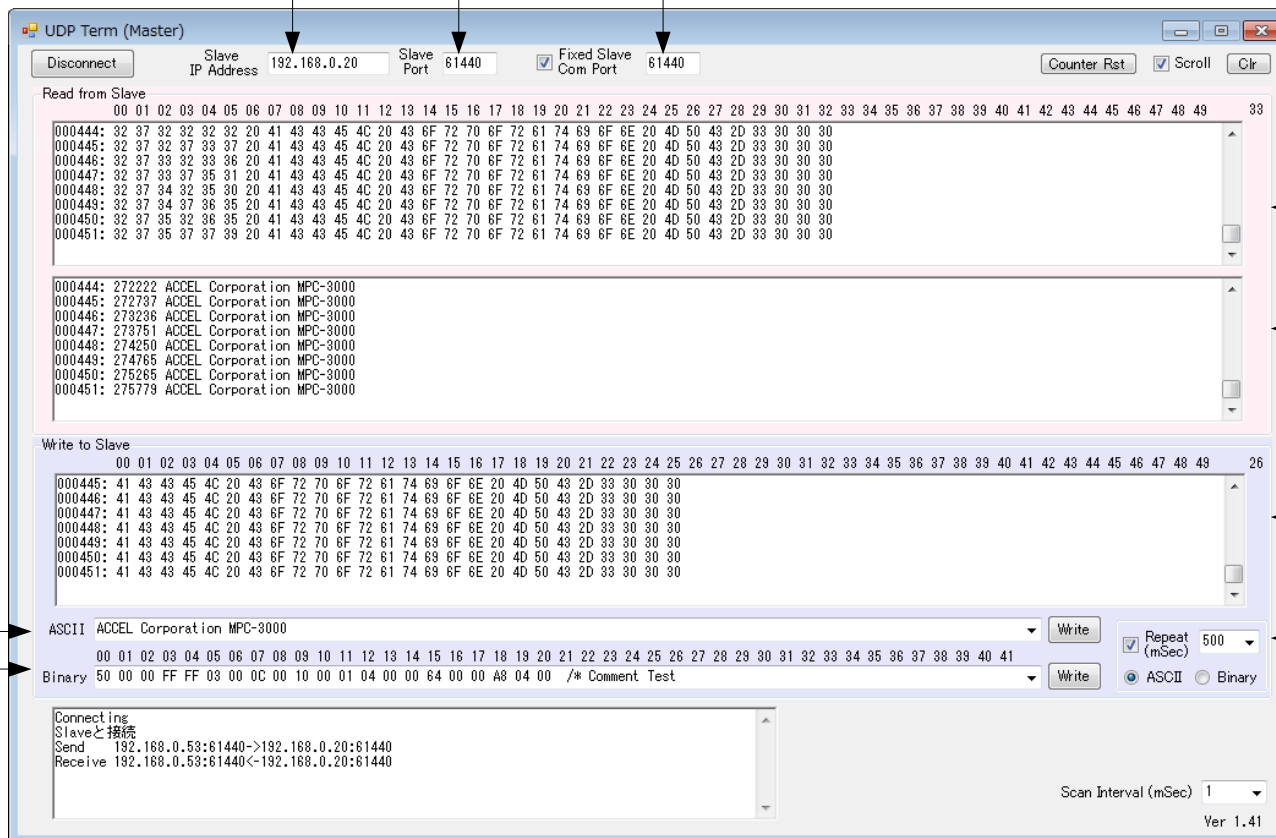
[Master] PC



UdpTermM.exe
IP Address 自動
Port 自動(固定)

MPCの IP Address MPCのポート MPCと通信するポートを固定

送信する文字列
ASCII
Binary



MPCから送信された文字列

Binary表示

ASCII表示

MPCへ送信した文字列

繰り返し送信

■ モニターツール(1) TcpMon.exe
Master・Slave TCP通信のモニタ

Proface GP4301TW

本体 イーサネット設定

IPアドレス 192.168.0.96、サブネットマスク 255.255.255.0、ポート 8000

デフォルトゲートウェイ 192.168.0.248

周辺機器 > 接続機器設定

接続機器

三菱電機(株) Q/QnAシリーズイーサネット TCP

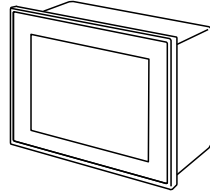
通信設定

ポート番号 固定/自動、タイムアウト 5sec、リトライ 0、送信ウェイト 0

機器設定

接続機器名 PLC1、IPアドレス 192.168.0.67、ポート番号 9000、データコード バイナリ

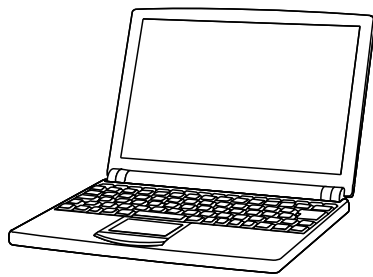
[Master]
タッチパネル



(4)
Write

(1)
Read

自己 IP Address, 受信ポート
MPCのIP Address, ポート



TcpMon.exe
IP Address 192.168.0.67
Port 9000

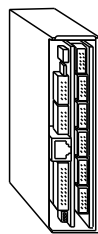
The screenshot shows the TcpMon interface with the following details:

- Listener IP Address: 192.168.0.67, Listen Port: 9000
- Slave IP Address: 192.168.0.20, Slave Port: 9000
- MPC Telnet Port: 23
- Scroll(SpaceKey): Clr
- Master section:
 - (1) Read: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 21
 - (2) Write: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 27
- Slave section:
 - (3) Read: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 27
 - (4) Write: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 21
- Status bar:
 - 000000: Masterと接続 192.168.0.96:9000
 - Slaveと接続 192.168.0.20:9000
 - Receive 192.168.0.67:9000->192.168.0.96:9000
 - Send 192.168.0.67:49681->192.168.0.20:9000
 - Receive 192.168.0.67:49681->192.168.0.20:9000
 - Send 192.168.0.67:9000->192.168.0.96:9000
- Settings:
 - Master Connection Timeout (Sec): 30
 - Slave Connection Timeout (Sec): 5
 - Reading Timeout (Sec): 3
 - Scan Interval (mSec): 1
 - Reconnection (Sec): 60
 - Master Timeout Detect:

(3)
Read

(2)
Write

[Slave] MPC

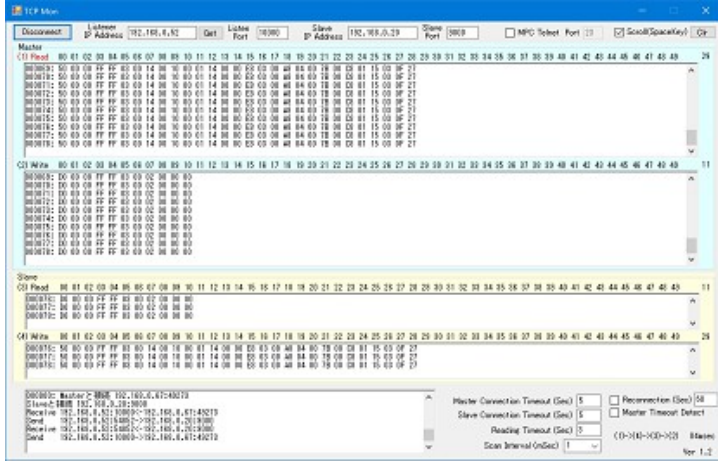
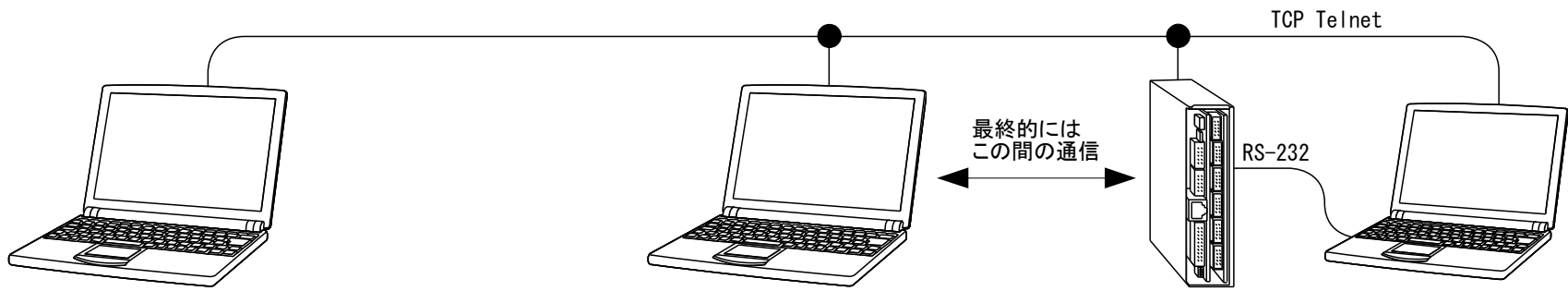


```
#SET_IP
IPadrs = 192.168.0.20
MASK = 255.255.255.0
Gateway= 192.168.0.248
OFF Delay ACK
MACadrs= 001EC08DAB6E
```

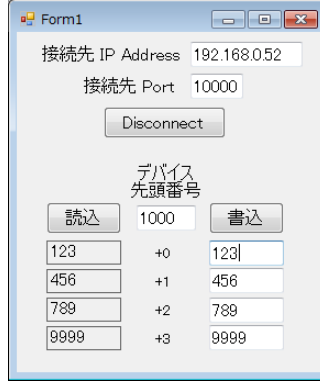
1 := TCP MC_PROTOCOL 9000 Slave connected-> 192.168.0.67
MPCのPort PCのIP Address

■ 設定コマンド
SET_IP TCP MC_PROTOCOL 9000

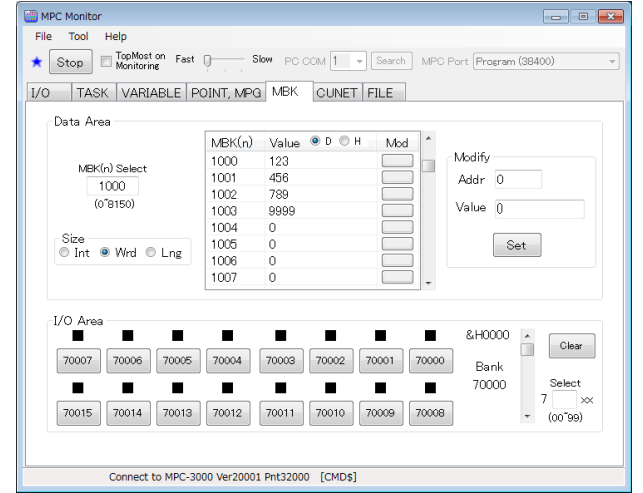
- 使用例1
通信内容を見ながらアプリケーションを開発できます。



TcpMon.exe
TCPアプリとMPCの通信内容が見えます



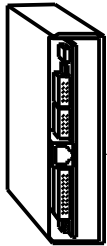
[Master]
ユーザー開発
TCPアプリケーション



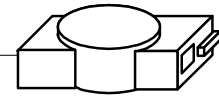
[Slave]
MPCプログラム開発

- 使用例2
MPC-3000とADAM-6052間のModbus/TCP通信をモニタしてみます。

[Master]
MPC-3000



[Slave]
Advantech I/Oモジュール
ADAM-6052

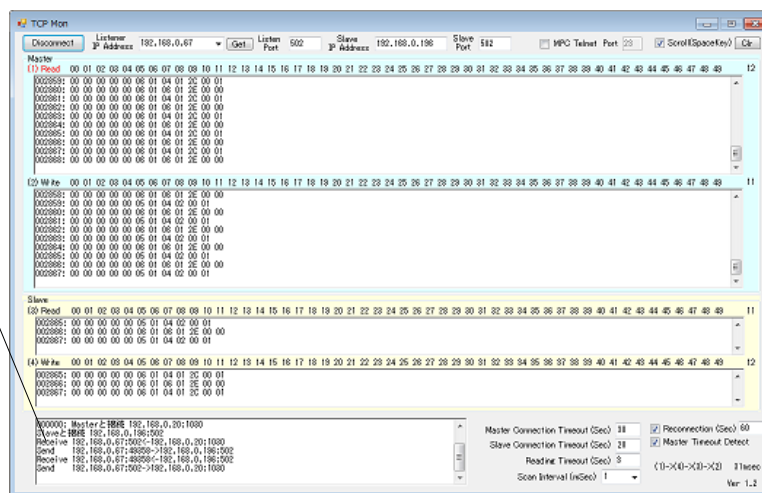


IP Address 192.168.0.196
Port Number 502

SET_IP 192 168 0 20 255 255 255 0 192 168 0 248
SET_IP TCP Modbus ADAM 192 168 0 67
(SET_IP TCP Modbus ADAM 192 168 0 196 とすればADAMと直接通信)



TcpMon.exe

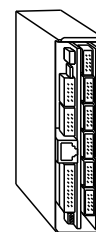


接続・通信時のエンドポイント

000000: Masterと接続 192.168.0.20:1030
Slaveと接続 192.168.0.196:502
Receive 192.168.0.67:502<-192.168.0.20:1030
Send 192.168.0.67:49358->192.168.0.196:502
Receive 192.168.0.67:49358<-192.168.0.196:502
Send 192.168.0.67:502->192.168.0.20:1030

Listener IP 192.168.0.67 Listen Port 502
Slave IP 192.168.0.196 Slave Port 502

- 使用例3
アプリ開発中のPCでもモニタができます。



MPC-3000
192.168.0.18:502

The screenshot displays the Visual Studio environment with the Modbus/TCP Master application running. The application window shows the configuration for the Master, including Slave IP (192.168.0.56) and Port (502). The TcpMon window shows the network traffic between the Master and the Slave (MPC-3000 at 192.168.0.18:502). The traffic log shows a successful read operation from the Slave to the Master.

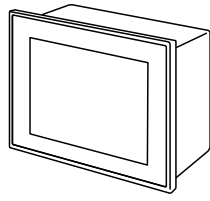
開発中のアプリ (Modbus/TCP Master)
192.168.0.56 (Portは自動)

TcpMon
192.168.0.56:502

MPC-3000
192.168.0.18:502

■ モニターツール(2) UdpMon.exe
Master・Slave UDP通信のモニタ

[Master]
タッチパネル



Proface GP4301TW

本体 イーサネット設定

IPアドレス 192.168.0.96、サブネットマスク 255.255.255.0、ポート 8000

デフォルトゲートウェイ 192.168.0.248

周辺機器 > 接続機器設定

接続機器

パナソニックデバイスSUNX(株) MEWTOCOL7-COMイーサネット UDP

通信設定

ポート番号 固定/自動、タイムアウト 5sec、リトライ 2、送信ウェイト 0

機器設定

接続機器名 PLC1、IPアドレス 192.168.0.67、ポート番号 8000

(4)
Write

(1)
Read

自己 IP Address, 通信ポート

MPCのIP Address, ポート, 通信ポート固定

GPのIP Address



UdpMon.exe
IP Address 192.168.0.67
Port 8000

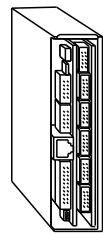
The screenshot shows the UdpMon application window with the following details:

- Listeners:**
 - Listener IP Address: 192.168.0.67, Listen Port: 8000
 - Master IP Address: 192.168.0.96
 - Slave IP Address: 192.168.0.20, Slave Port: 9000
 - Fixed Slave Com Port: 7000
- Master Log:**
 - (1) Read: Shows hex data from 000341 to 000350.
 - (2) Write: Shows hex data from 000345 to 000349.
- Slave Log:**
 - (3) Read: Shows hex data from 000348 to 000350.
 - (4) Write: Shows hex data from 000348 to 000350.
- Masterと接続:**
 - Receive 192.168.0.67:8000<-192.168.0.96:4021
 - Send 192.168.0.67:80338->192.168.0.20:9000
 - Receive 192.168.0.67:80338<-192.168.0.20:8000
 - Send 192.168.0.67:8000->192.168.0.96:4021
- Settings:**
 - Reading Timeout (Sec): 5
 - Scan Interval (mSec): 1
 - Ver 1.2

(3)
Read

(2)
Write

[Slave] MPC

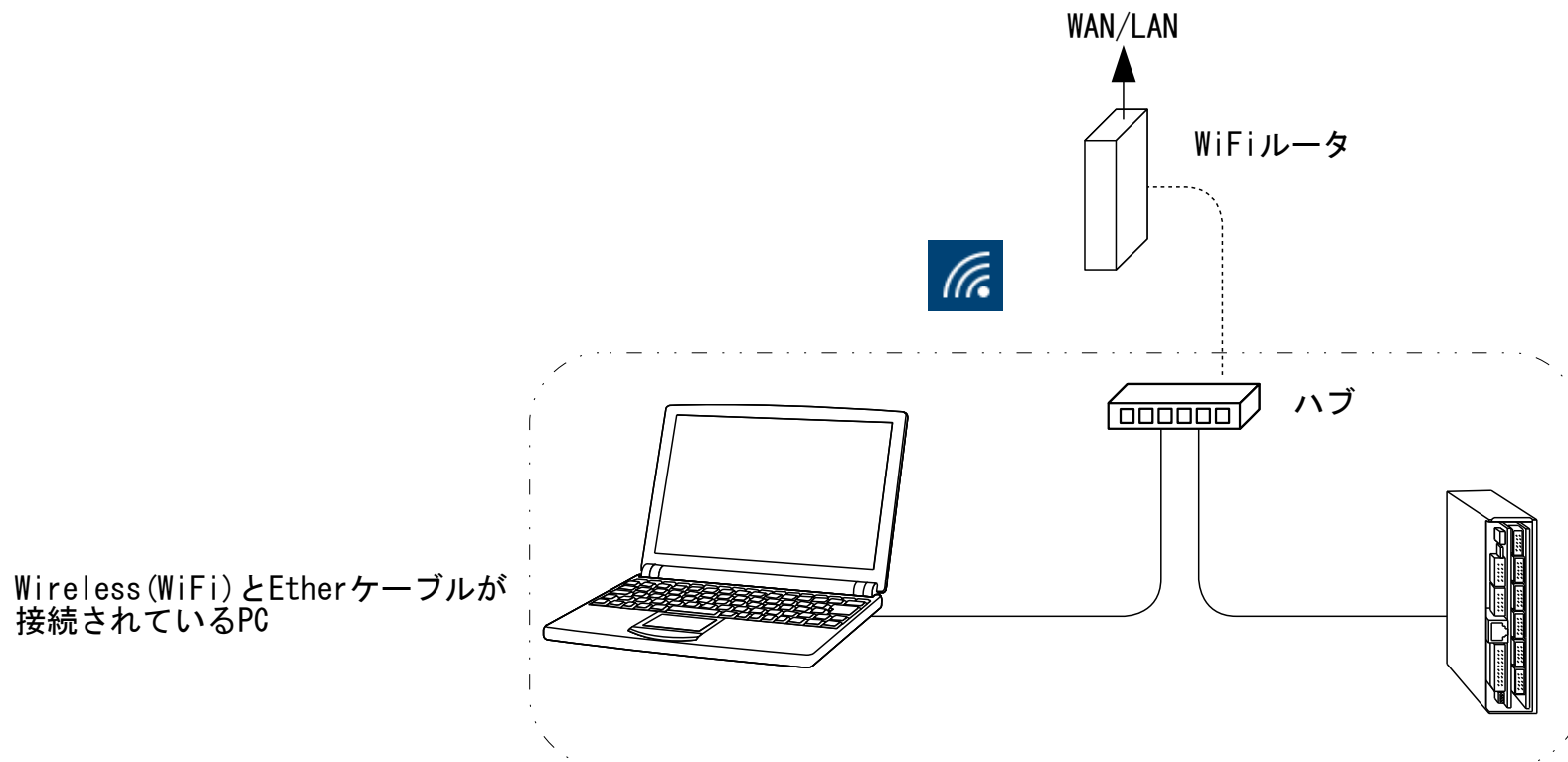


```
#SET_IP
IPAdrs = 192.168.0.20
MASK = 255.255.255.0
Gateway= 192.168.0.248
OFF Delay ACK
MACAdrs= 001EC08DAB6E
```

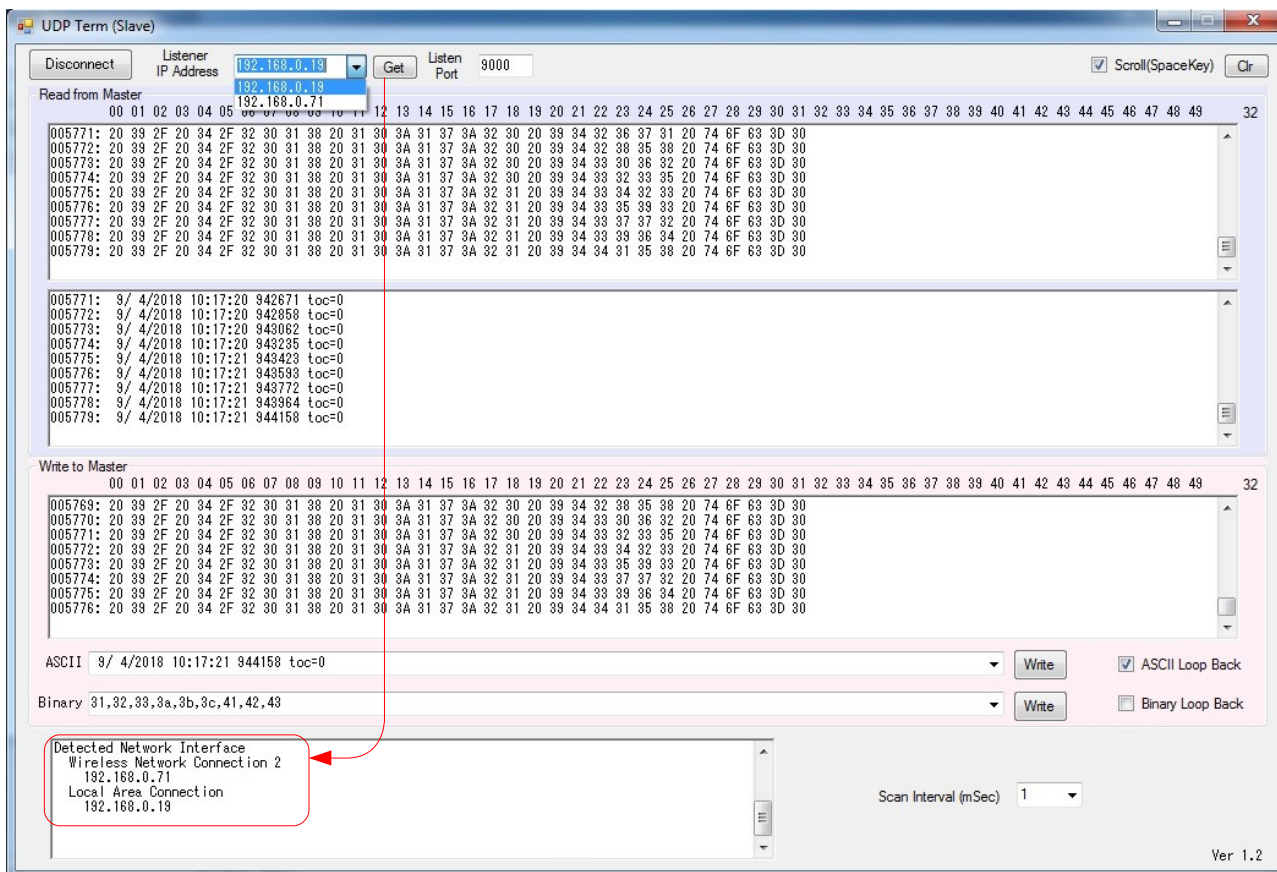
```
1 := UDP MEWTOCOL7 9000 Slave
MPCのPort
```

■ 設定コマンド
SET_IP UDP MEWTOCOL7 9000

■ 自己IP Addressの設定例
IP Addressが複数ある場合



Getボタンを押すとローカルIP Addressが2つ見つかります。
下のTextboxにインターフェースの種類が表示されるので確認して設定してください。
このPCはLocal Areaの方が具合が良いので19を設定。
通信は、MPC直結かLANと分離された状態(二点鎖線内)にするとより安定。



詳細はipconfigで

