EX9188END Series
Hardware & Software Manual

2005/6/3 V1.2

EXPERT DAQ

Table of Contents

1 Introduction:

Ethernet network is one of the most common infrastructures in most office buildings, factories and new homes. It allows you to remote control or access devices conveniently. Using Ethernet in industrial area is appealing because you don't have to go on site in order to gather the information.

EX9188E series are the embedded controllers designed to meet the most common requirements of Internet or Ethernet applications. An on-board 10base-T port is equipped with a RJ-45 connector, which let you connect to the network through a cable. It is powered by 80188-40 processor with 512K bytes of static RAM and 512K bytes of flash memory. RS232 and RS485 ports are provided. EX9188E is can be treated as a RS232 to Ethernet converter. The central computers can control all the RS232 ports through Ethernet with the help of EX9188E. RS232 devices can be hooked to the closest network hubs and link to the central computer. After linking all these devices together, the software application would have to develop in order to access them. In general, it is more difficult to develop a TCP/IP based application program than COM port based program. To get a software "VSP(Virtual Serial Port) or VirCom" from Web/CD of TOPSCCC or Easyunet co. that It can be used to take care of network protocol layers and let the host computer visualize COM ports of EX9188E. It makes the host computer have virtual COM ports which are actually mounted in EX9188E. Therefore, the software applications can be COM port based and the old COM port based application can still be used. The advantages can be concluded that it provides a easy interface for software development and it also keeps the old system going without program modification.

Ps: 1. VSP(Virtual Serial Port) can download from http://www.topsccc.com/ for user simple test.

2. VirCom driver can download from http://www.easyunet.com for user simple test.

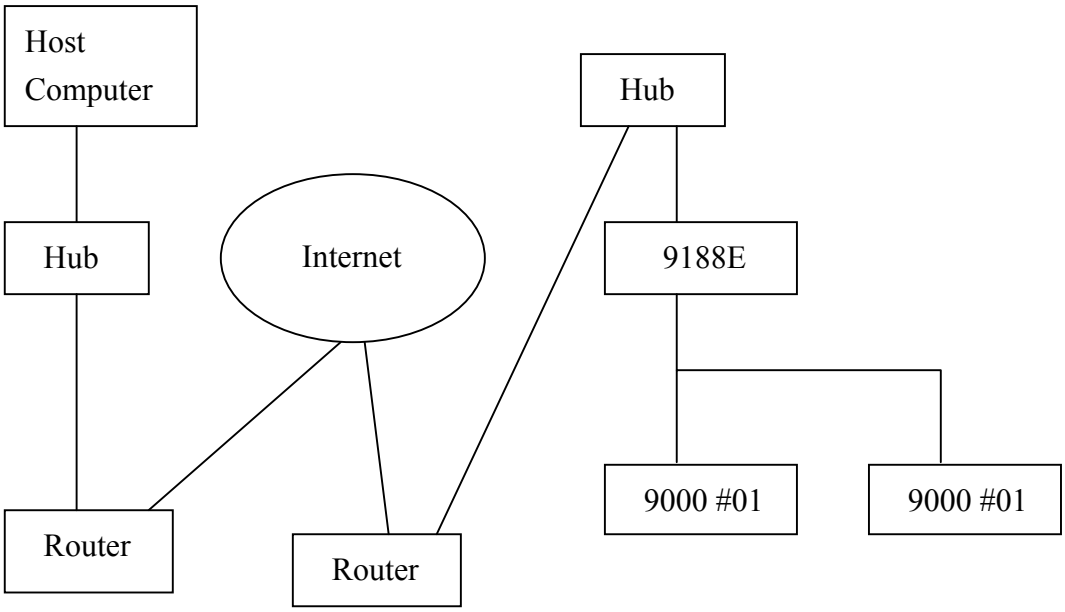


Fig1.1 Simple Structure of EX9188E Network

Default COM port configuration:				
	Baudrate	**Dtatbit**	**Parity**	**Stopbit**
COM1	57600	8	N(none)	1
COM2	9600	8	N(none)	1
COM3	9600	8	N(none)	1
COM4	9600	8	N(none)	1
COM5	9600	8	N(none)	1
COM6	9600	8	N(none)	1
COM7	9600	8	N(none)	1
COM8	9600	8	N(none)	1

2 System Description

EX9188E is network-based industrial controller. It is powered by 80188 CPU and equipped with several discrete devices in the circuit module. It supports the following functions, and all these hardware functions are packed to library. About library usage, you can refer Section 5.

1. 512K SRAM and FLASH.
2. Build-in RTC, NVRAM and serial EPROM.
   (Note: Block 7 is reserved for system).
3. Build-in 5-Digit LED display.
4. One RS-485 port with auto direction control.
5. One 5-wire hardware flow control RS-232 port.
6. Six RS-232 ports ad-on card. (Its function depends on your demand, e.g. AD/DA card, digital IO and so on.)
7. One system timer. (Interrupt no. 08)
8. One general-purpose timer.

In software view, EX9188E maps 512K SRAM in the lower 512K memory space and 512K Flash in the higher 512K memory space. Software is stacked by miniBios, ROM-DOS and Sockets. ROM-DOS and Sockets are stable products and good-supported by Datalight.

2.1 Memory mapping

Memory Address	Memory status	H/W component
0x00000~0x7FFFF	SRAM(512K)	SRAM SPACE
0x80000~0xEFFFF	Flash Memory(448K), ROM DISK initial at 0x80000	FLASH SPACE
0xF0000~0xFBFFF	ROM DOS(48K)	
0xFC000~0xFDFFF	Reserve(8K)	
0xFE000~0xFFFFF	MiniBIOS(8K)	

2.2 IO mapping

IO Address	IO status	H/W component
0x000~0x0ff	Ad-on Card	16C550s
0x100~0x2ff	Reserved	N/A
0x300~0x3ff	Ethernet Controller	RTL8019
0x400~0x6ff	Reserved	N/A

2.3 Minbios

The Datalight's miniBIOS, as the name implies, is the minimum BIOS needed to run Datalight's ROM-DOS. It is not intended to replace full BIOS but to serve those embedded situations that do not require full BIOS support.

The Datalight miniBIOS is a BIOS subset that provides support for only the BIOS features absolutely essential to the operation of ROM-DOS. The miniBIOS includes BIOS support for a remote console, timer, BIOS extensions, and hardware equipment identification. The miniBIOS does not provide support for floppy or hard disks, printers, the standard PC keyboard, or monitors.

Name	Interrupt	Sub-function
Coprocessor Esc instruction	07H	reserved
Timer 0 tick	08H	Reserved (hardware)
Keyboard Input	09H	(stubbed for APM)
Serial receive char	0BH	Reserved (hardware)
Video TTY Output	10H	0EH
Get equipment list	11H	
Get memory size in K	12H	
Disk I/O	13H	reserved
Extended Memory Support	15H	reserved
Keyboard Support	16H	00H,01H,02H
Keyboard, push Scan code into Buffer	16H	05H
Boot failure message	18H	
System DOS-boot	19H	
Time of Day	1AH	00H,01H
Timer Tick	1CH	

2.4 Romdos

EX9188 Module is equipped with DataLight ROM-DOS. ROM-DOS is a disk operating system that can be loaded in Read Only Memory (ROM) and can run entirely from within ROM and also from a hard or floppy disk, such as in a desktop system. ROM-DOS is functionally equivalent to other brands of DOS and can run

programs that are executable under a standard DOS (which executes from RAM). With ROM-DOS, the executable program resides on a disk or is placed in ROM along with ROM-DOS.[1]

2.5 Sockets

Datalight SOCKETS is a command-driven Internet protocol software package for embedded systems running DOS. Datalight SOCKETS provides a powerful data communication facility where by embedded systems and users of embedded systems can communicate with other computers (including PCs and mainframes) and their printers.

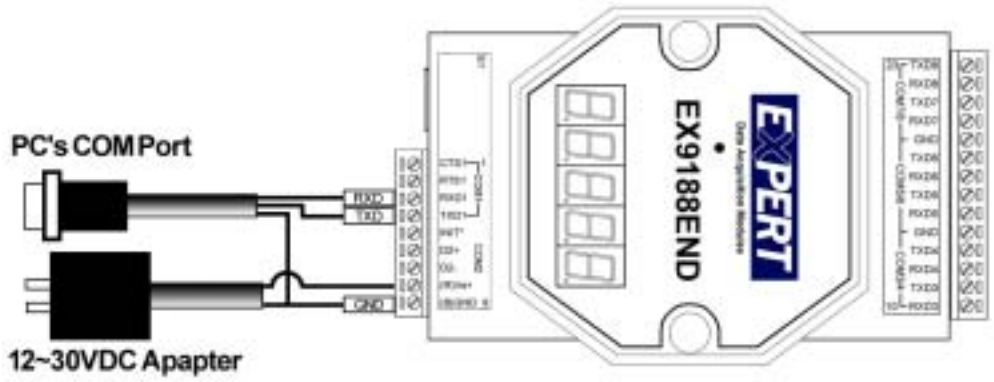
Datalight SOCKETS is a networking communications application, providing both client and server services on the network, which allows you to:

- Run applications on a TCP/IP host system from a remote embedded system.
- Transfer data between an embedded system and TCP/IP hosts.
- Run network aware applications on an embedded system.
- Print to an embedded system from TCP/IP hosts and vice versa. [2]

[1] About ROM-DOS, Please Contact Datalight (http://www.datalight.com) for more information.
[2] About Sockets, Please Contact Datalight (http://www.datalight.com) for more information.

3 Quick Start

1. Wiring connection



2. Connect COM1 of EX9188E to com1(2) of PC.
3. Under Hyperterminal of PC **(Com port setting: 57600,8.N.1, none flow control)**
4. Power On EX9188E(now EX9188E will under TCP mode).
   If need to place command from PC's COM port to EX9188E
   Please connect(touch) the INIT* pin to (B)GND and release,then EX9188e will under ROMDOS mode).
5. HyperTerminal of PC will show the boot message of EX9188E and DOS prompt (a:\>)
6. DOS command can be executed at command prompt as dir a:\; dir b:.
7. For file transfer or downloading file, please refer to manual Page.9
8. For updating the contents of ROM DISK, please refer to manual Page.10
9. Please refer to the EX9188xd S/W manual to get others function & operation as RAMDISK, VDISK

Configuration of EX9188E

Use INFOSET.EXE to read and write/set the EX9188E
******** 9188End Infomation Setting Utility *********
M : Modify Model No 9188E4
A : Modify MAC ADDR 0: 0: 0: 0: 0:f0 (Read Only)
I : Modify IP ADDR 211. 22. 83. 46
N : Modify NetMask 255.255.255. 0
G : Modify Gateway 211. 22. 83. 41
B : Modify Baudrate 1:9600 2:9600 3:9600 4:9600
5:9600 6:9600 7:9600 8:9600
D : Modify Databit 1:8 2:8 3:8 4:8 5:8 6:8 7:8 8:8
P : Modify Parity 1:0 2:0 3:0 4:0 5:0 6:0 7:0 8:0
S : Modify Stopbit 1:208 2:1 3:1 4:1 5:1 6:1 7:1 8:1
E : Enable/Disable com1 Flow Control : (Disable)
F : Factury Default
Q : quit

4 Utilitys

Several utilities run on EX9188E server will be introduced in the following sections.

You can use Hyperterminal to show the 9188E's A:\> to run this utility

Step 1. Start→Program→Accrssories→Communication→Hyperterminal.

Step 2. Select the correct Com port, Baudrate(57600), Databit(8), Parity(none), Stopbit(1)

Step 3. Power on the EX9188E, touch the 9188E's Init* to GND after XCOM.exe loaded. Then the Terminal window will show the Rom Dos prompt A:\>

4.1 Xm.exe

When you develop program on PC, you will need Xm.exe to transfer your program to EX9188E and run/test it.

Here are steps to upload file from PC to EX9188E.

1. On EX9188E DOS prompt, run this command "xm.exe /r <filename>".
2. Under HyperTerminal window, select:
   Transfer→
   Send file →
   Protocol (Xmodem/1K-Xmodem) →
   Filename(the file which will be sent)→
   OK. (within 60Sec).
3. Wait until upload file transmission completed.

Also, you can download a file from EX9188E to PC. Follow these steps:

1. On EX9188E DOS prompt, run this command "xm.exe /s <filename>".
2. Under HyperTerminal window, Select:
   Transfer →
   Receive file→
   Use 1K-Xmodem as receiving protocol→
   Place receive file in chosen folder (point to the file which is going to be downloaded) →
   OK
3. Wait until download file transmission completed.

4.2 Romdisk.exe

ROMDISK.EXE is a utility of ROM-DOS run on PC. It is for making a ROM-DOS image file. The steps to create a ROM-DISK image file are listed below:

1. Create a sub-directory called ROM.
2. Copy all the files needed for Romdisk.
3. run "ROMDISK.EXE" (romdisk.exe will put all files in the directory ROM into a file named Rom.img
4. Download file to EX9188E via TOROM.EXE (see "Using torom.exe")
5. Reboot EX9188, you will see disk A in EX9188E contains all the files of directory ROM.

4.3 Torom.exe

TOROM.EXE is used to transfer the ROMDISK image file to Flash ROM of EX9188E module and the contents of Flash ROM of ROM disk will then be updated. The steps are listed below:

1. Connect EX9188E module to PC. Use HyperTerminal as terminal.
2. run TOROM.EXE on EX9188E
3. Under HyperTerminal, select:
   Transfer→Send file→Protocal(Xmodem)→File name→OK.
   (Use Xmodem protocal & key in Rom image file name then execute file transfer function).
4. When ROM image file transmission is completed, reboot EX9188E module.
5. You will see a updated ROM disk.

Note: When execute the TOROM.EXE on EX9188E module, the process of step3 must be finished under 60 seconds otherwise timeout will occur. If timeout is the case, INIT* pin will have to be used to download the program to Flash ROM as follows:

Power on the system when EX9188E module's INIT* pin is wired to ground and COM1 is connected to PC. The disk image can be downloaded from PC to flash ROM of EX9188E module under Hyper-Terminal by clicking "transfer", "receive file", then choose Xmodem as the protocol and key in the file name and path. If the update is not successful, then repeat the process. If users want to debug the system from COM1, just power on the system with INIT* floating. **Note: INIT* must be disconnected from ground immediately before the updating disk image completes, otherwise the system will hang.**

4.4 Vdisk

Vdisk is a device driver which can create a virtual disk or RAM DISK from system memory. The contents of virtual disk will be lost when power-off. Vdisk.sys is a system file which can be configured in config.sys, the format is shown below:

Device=vdisk.sys [size[secs[dirs]]] /E
Size is for RAM DISK size, default is 64KB; Secs is for sector size, default is 512 bytes, valid values are 128, 256, 512.; Dirs is the number of sub directories or files in root of RAM DISK, default is 64.

Example: In config.sys:
device=vidsk.sys
device=vdisk.sys 32 128 16

Set 64kb RAM DISK B: ; 512 byte SECTOR (when ROM DISK A: exist)
Set 32kB RAM DISK C: ; 128byte SECTOR, 16 files in root directory.

4.5 Infoset.exe

Some EX9188E information are stored in Serial EPROM block 7 including model name, MAC address, IP address, Gateway address, and COM port data format. These information are easy to modify via infoset.exe.

4.6 N.com/Socketp.exe

N.com is standard packet driver for EX9188E module. It provides interface for socketp.exe and Ethernet controller where Socketp.exe is Sockets/DOS TCP/IP kernel provide by Datalight. These two utilities must be executed to enable EX9188E network function. They are packed to net.bat. That means you can execute run.bat to enable EX9188E network funcion.

4.7 Xping.exe

Xping.exe is the most basic test to see if connections are working as ping.exe in UNIX/Windows system. It is provide by Datalight.

4.8 VSP/VirCOM

Applications that communicated directly with a physical communication port or serial line device in the past can now be used over a TCP/IP network with VSP/VirCOM. By creating virtual COM ports on the PC, VSP/VirCOM redirects the communications from the virtual COM ports to an IP address and port number on a serial server that connects the serial line device to the network. The application can communicate with many devices anywhere on the network at the same time.

When using virtual COM applications, a PC control up to 255 COM ports. The VSP/VirCOM software can turn EX9188E into a RS232 to Ethernet converter. Sending or receiving data over a TCP/IP port with a serial communications program can be accomplished. Users can mount their systems to Ethernet/Internet with increased ease because the old COM based program requires no modifications.

A VSP/VirCOM driver must be installed first in order to use the virtual COM application. After that, the VSP/VirCOM utility can map any EX9188E COM port and let PC visualize EX9188E COM ports so that PC can get the direct control of them without taking care of network protocol. Users do not need to worry about network connections. The VSP/VirCOM driver will handle all Ethernet/Internet connections.

Here, we will show three utility to run VSP/VirCOM.

4.9 Xcom.exe

Xom.exe is the server program run in EX9188E. It accepts commands from PC via network and translates them to real action in EX9188E. Xcom.exe can be set in the autoexec.bat and then it will be started after EX9188E booting successfully. You can abort xcom.exe after it is executed via send “00quit” command from NetTest or connect INIT* pin to low.

When Xcom.exe runs, the 5-digit led will show system information sequentially as Table 4.1.

Table 4.1 Information Sequences Show on 5-digit LED

No	Format	Meaning
1	11111	Label 1
2	1XAAA	X: space, AAA:IP digit 1
3	2XAAA	X: space, AAA:IP digit 2
4	3XAAA	X: space, AAA:IP digit 3
5	4XAAA	X: space, AAA:IP digit 4
6	22222	Label 2
7	1AAAA	AAAA: baudrate of com1
8	2AAAA	AAAA: baudrate of com2
9	3AAAA	AAAA: baudrate of com3
10	4AAAA	AAAA: baudrate of com4
11	5AAAA	AAAA: baudrate of com5
12	6AAAA	AAAA: baudrate of com6
13	7AAAA	AAAA: baudrate of com7
14	8AAAA	AAAA: baudrate of com8
15	33333	Label 3
16	1XABC	X: space, A: databit, B: parity, C: stopbit of Com1
17	2XABC	X: space, A: databit, B: parity, C: stopbit of Com2
18	3XABC	X: space, A: databit, B: parity, C: stopbit of Com3
19	4XABC	X: space, A: databit, B: parity, C: stopbit of Com4
20	5XABC	X: space, A: databit, B: parity, C: stopbit of Com5
21	6XABC	X: space, A: databit, B: parity, C: stopbit of Com6
22	7XABC	X: space, A: databit, B: parity, C: stopbit of Com7
23	8XABC	X: space, A: databit, B: parity, C: stopbit of Com8
24	XXXXX	XXXXX: space

Xcom.exe support the following commands as Table 4.2, you can test them via NetTest on PC. NetTest will be introduced in the following section.

Table 4.2 Xcom.exe Support Commands

Command	**Explanations**	**Example**	
		Input	**Return**
00*[arg]*	Stop Server Xcom.exe. *Arg: quit* *Return:(none)*	00quit	(none)
01	Read version information. *Return: Vm.n.rr[mm/dd/yy]* *m: major version* *n: minor version* *rr: reversion number*	01	V2.3.B1[16/05/2005]
02*[arg]*	Set baudrate of COM ports *Arg: NBBBB* *N: COM port no (1~8)*	02257600	OK

	BBBB: Baudrate to be set *(1200,2400,4800,9600,19200,38400, 57600,115200)* *Return:* *ERROR: Fail,* *OK: Success.*		
03*[arg]*	Set data format of COM ports *Arg: NDPS* *N: COM port no (1~8)* *D:DataBit* *(7, 8 ==> COM1~ COM2)* *(5, 6, 7, 8 ==> COM3 ~ COM8)* *P:Parity* *(0 ==> none parity)* *(1 ==> even parity)* *(2 ==> odd parity)* *S:StopBit* *(1 ==> COM1 ~ COM2)* *(1, 2 ==> COM3 ~ COM8)* *Return:* *ERROR: Fail,* *OK: Success.*	032801	OK
04*[arg]*	*Read System reset status* *Arg: N* *N: Client*	0414	141 (First asking from client 14 after system reset) 140 (Not first asking from client 14 after system reset)
05*[arg]*	*RTS (COM1 only)* *Arg: 1R* *R: Set RTS* *(0 ==> Off)* *(1 ==> ON)*	0511	OK (COM1 RTS on)
06*[arg]*	Set baudrate of COM ports and write the baudrate setting to SEPROM. *Arg: NBBBB* *N: COM port no (1~8)* *BBBB: Baudrate to be set* *(1200,2400,4800,9600,19200,38400, 57600,115200)* *Return:* *ERROR: Fail,* *OK: Success.*	0629600	OK

07*[arg]*	Set data format of COM ports and write the data format setting to EEPROM. *Arg: NDPS* *N: COM port no (1~8)* *D:DataBit* *(7, 8 ==> COM1~ COM2)* *(5, 6, 7, 8 ==> COM3 ~ COM8)* *P:Parity* *(0 ==> none parity)* *(1 ==> even parity)* *(2 ==> odd parity)* *S:StopBit* *(1 ==> COM1 ~ COM2)* *(1, 2 ==> COM3 ~ COM8)* *Return:* *ERROR: Fail,* *OK: Success.*	072801	OK
08*[arg]*	Set IP address. *Arg: iiipppIIIPPP* *iii/ppp/III/PPP: 3 digits number (000~255)* *Return:* *ERROR: Fail,* *OK: Success.*	08192009040100	OK
10	Read module name. *Return:* *Module Name*	10	EX9188En
11*[arg]*	Server test. *Arg:SSS* *String to be test.* *Return:* *SSS*	11Hello!	Hello!
12*[arg]*	Set Gateway address. *Arg: iiipppIIIPPP* *iii/ppp/III/PPP: 3 digits number (000~255)* *Return:* *ERROR: Fail,* *OK: Success.*	12192009040254	OK
13	Read Gateway address. *Return:* *iii.ppp.III.PPP*	13	192.9.40.254

14	Set Mask. *Arg: iiipppIIIPPP* *iii/ppp/III/PPP: 3 digits number (000~255)* *Return:* *ERROR: Fail,* *OK: Success.*	14255255255000	OK
15	Read Mask. *Return:* *iii.ppp.III.PPP*	15	255.255.255.0
16*[arg]*	Read COM port setting. *Arg:N* *N:COM port no(1~8)* *Return:* *baudreat,data,parity,stop*	162	19200,8,0,1
20*[arg]*	Enable/Disable LED show information. *Arg:E* *E:0=>disable, 1=>enable.* *Return:* *ERROR: Fail,* *OK: Success.*	201	OK
21	Read MAC address. *Return:* *Xx:xx:xx:xx:xx:xx*	21	00:00:f1:86:97:a6
22*[arg]*	Enable/Disable flow control(com1 only) *Arg:11* Enable flow control of com1 & RTS CTS auto control by hardware. *Arg:10* Disable flow control.	2211	OK (Set COM1 flow control enable.)
231	Inquire the flow control status of COM port (com 1 only) *Return:* *0: Flow control of COM1 is disable.* *1: Flow control of COM1 is enable.*	231	0 (Flow control of COM1 is disable.)
28*[arg]*	Reset individual com port *Arg:N* *N:COM port no(1~8)*	281	OK
32*[arg]*	Set com function(Mbxcom or xcom) *Arg:N1 (*set com N to modbus mode.) *Arg:N0 (*set com N to xcom mode.) *N:COM port no(1~8)*	3211	OK (Set COM1 to modbus mode.) (For 9188E_MTCP only)
33*[arg]*	Read com function(Mbxcom or xcom) *Arg:N* *N:COM port no(1~8)*	331	1 COM1 is Modbus mode (For 9188E_MTCP only)
99	Reboot 9188E	99	

4.10 NetTest

This is a utility that lets you send and receive data to/from EX9188E through TCP/IP. Users would have to specify the IP address of EX9188E and the connection port by clicking on “connect”. Then the program should be ready for send commands to EX9188E and receive data. The main purpose of NETTEST is used for diagnosing the network connection of EX9188E.

You can install NetTest by double clicking setup.exe on NetTest directory and then the setup.exe will guide you to install the NetTest completely.

Fig4.1 shows the GUI of NetTest. Some skills to operate NetTest are as follows:

- Specify IP of EX9188E and Connection port in “Server Setting” block.
- Press “Connect” bottom. “Connect” bottom will be changed to “DisConnect” bottom if connecting successfully and server information will show on “Server Information” block.
- Send commands list as Table5.2 via “Commands” block. Executes it by press ENTER on keyboard or press “Send” bottom. EX9188E response will show on “9188E Response Window” block.
- Send string to COM port via “Send” block.
- Response will show on “Receive” block.

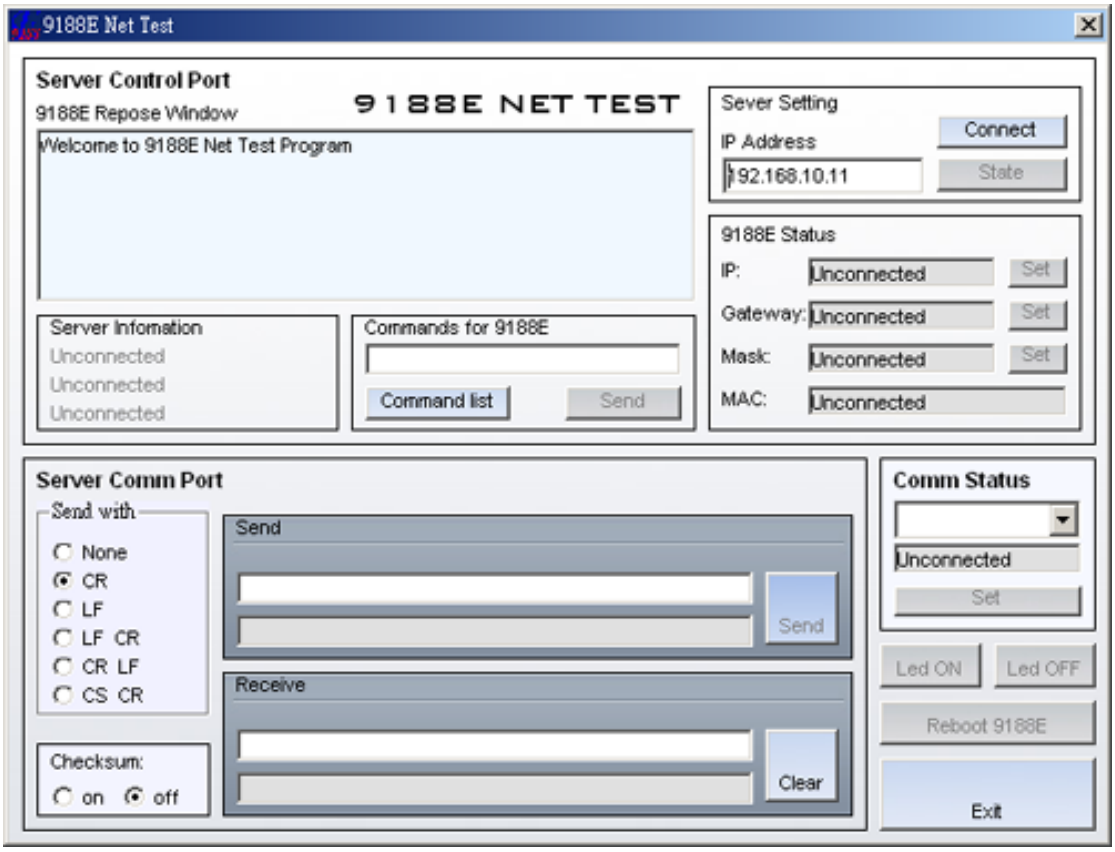


Fig 4.1 The GUI of NetTest

Use TCP/IP to connection then the TCP port 10000 is only for command and the port 10000+n are Belong to COMn for bypass data in bidirection

Example:

Please select the COM port which is connected to the RS485 Devices or EX9000 series of EX9188END to send the Command:

As $01M then received !9041D

As $012 then received !01400600

As $01F then received !VER 1.A

Port 10000 can use the command as table 4.2

5 Library

Through section 5.1 to 5.6 are functions compatible to EX9188D, please refer EX9188D library userguide for detail. Please refer Datalight Sockets 2.0 Developer's Guide for more detail about functions list in section 5.7 Socket API. Here only list the function prototypes for your reference.

5.1 Flash

int FlashReadId(void);
int FlashWrite(unsigned int seg, unsigned int offset, char data);
int FlashErase(unsigned int seg);
int FlashRead(unsigned int seg, unsigned int offset);

5.2 NVRAM/Serial EPROM

int ReadNVRAM(int addr);
int WriteNVRAM(int addr, int data);

int WriteEEP(int block, int addr, int data);
int ReadEEP(int block, int addr);
void EnableEEP(void);
void ProtectEEP(void);

Note: Block 7 is reserved for system.

5.3 LED/5-Digit 7-Segment LED

void LedOff(void);
void LedOn(void);

void Init5DigitLed(void);
int Show5DigitLed(int position, int value);
int Show5DigitLedSeg(int pos,int value);
int Show5DigitLedWithDot(int pos, int data);
void Set5DigitLedTestMode(int mode);
void Set5DigitLedIntensity(int mode);
void Disable5DigitLed(void);
void Enable5DigitLed(void);

5.4 COM port

int InstallCom(int port, unsigned long baud, int data, int parity, int stop);

```
int    RestoreCom(int port);
int    IsCom(int port);
int    ReadCom(int port);
int    ToCom(int port, int data);
int    ClearCom(int port);
int    WaitTransmitOver(int port);
int    DataSizeInCom(int port);
void SetFlowControlActive(void);
void SetFlowControlInactive(void);
int    SendCmdTo7000(int iPort, unsigned char *cCmd, int iChksum);
int    ReceiveResponseFrom7000(int iPort, unsigned char *cCmd, long lTimeout,
int iChksum);
```

5.5 Timer

```
int TimerOpen(void);
int TimerClose(void);
void TimerResetValue(void);
unsigned long TimerReadValue(void);
void DelayMs(unsigned t);
int StopWatchReset(int channel);
int StopWatchStart(int channel);
int StopWatchStop(int channel);
int StopWatchPause(int channel);
int StopWatchContinue(int channel);
int StopWatchReadValue(int channel,unsigned long *value);
int CountDownTimerStart(int channel,unsigned long count);
int CountDownTimerReadValue(int channel,unsigned long *value);
void InstallUserTimer(void (*fun)(void));
void InstallUserTimer1C(void (*fun)(void));
void Delay(unsigned ms);
void Delay_1(unsigned ms);
```

5.6 ETC

```
int getch4(void);
```

```
int kbhit4(void);
int ungetch4(int key);
void putch4(int data);
int ReadInitPin(void)
```

5.7 Socket API

```
int DisableAsyncNotification(void);
int EnableAsyncNotification(void);
DWORD GetAddress(int iSocket);
int GetPeerAddress(int iSocket, NET_ADDR *psAddr);
int GetKernelInformation(int iReserved,BYTE bCode,BYTE bDevID,
                                        void *pData,WORD *pwSize);
int GetVersion(void);
int ICMPPing(DWORD dwHost, int iLength);
int IsSocket(int iSocket);
int GetDCSocket(void);
int GetSocket(void);
int GetKernelConfig(KERNEL_CONFIG *psKC);
int ConvertDCSocket(int iSocket);
int GetNetInfo(int iSocket,NET_INFO *psNI);
int ConnectSocket(int iSocket, int iType, NET_ADDR *psAddr);
int ListenSocket(int iSocket, int iType, NET_ADDR *psAddr);
int SelectSocket(int iMaxID, long *plIflags, long *plOflags);
int ReadSocket(int iSocket, char *pcBuf, WORD wLen,
                        NET_ADDR *psFrom, WORD wFlags);
int ReadFromSocket(int iSocket, char *pcBuf, WORD wLen,
                                NET_ADDR *psFrom, WORD wFlags);
int WriteSocket(int iSocket, char *pcBuf, WORD wLen, WORD wFlags);
int WriteToSocket(int iSocket, char *pcBuf, WORD wLen,
                                NET_ADDR *psTo, WORD wFlags);
int EofSocket(int iSocket);
int FlushSocket(int iSocket);
int ReleaseSocket(int iSocket);
int ReleaseDCSockets(void);
int AbortSocket(int iSocket);
int AbortDCSockets(void);
```

```
int ShutDownNet(void);
int SetAlarm(int iSocket,DWORD dwTime,int (D_FAR *lpHandler)(),
                    DWORD dwHint);
int D_FAR *SetAsyncNotification(int iSocket,int iEvent,
int (D_FAR *lpHandler)(),DWORD dwHint);
DWORD ResolveName(char *pszName, char *pcCName, int    iCNameLen);
DWORD ParseAddress(char *pszName);
int SetSocketOption(int iSocket,int iLevel,int iOption,
                                DWORD dwOptionValue,int iLen);
int JoinGroup(DWORD dwGroupAddress, DWORD dwInterfaceAddress);
int LeaveGroup(DWORD dwGroupAddress, DWORD dwInterfaceAddress);
int IfaceIOCTL(char *pszName, WORD wFunction);
int GetSocketsVersion(void);
```