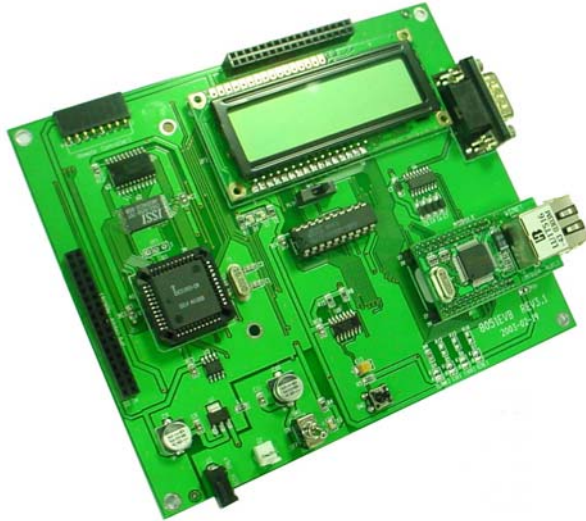


# EVB8051 User's Manual

Version 3.1



---

# COPYRIGHT NOTICE

Copyright 2002 WIZnet, Inc. All Rights Reserved.

Technical Support: [support@wiznet.co.kr](mailto:support@wiznet.co.kr)

Sales & Distribution: [sales@wiznet.co.kr](mailto:sales@wiznet.co.kr)

General Information: [info@wiznet.co.kr](mailto:info@wiznet.co.kr)

Tel: +82-2-547-9709

Fax: +82-2-547-9711

For more information, visit our website at <http://www.wiznet.co.kr>

---

# Table of Contents

|  |           |
|--|-----------|
| <b>1. Getting Started.....</b>                             | <b>1</b>  |
| 1.1 EVB8051 Package .....                                  | 1         |
| 1.1.1 Components .....                                     | 1         |
| 1.1.2 Software CD.....                                     | 7         |
| 1.2 System Configuration .....                             | 9         |
| 1.2.1 PC Setup.....  | 9         |
| 1.2.2 Evaluation Board Configuration.....                  | 13        |
| <b>2. User's Guide.....</b>                                | <b>14</b> |
| 2.1 Evaluation Board Layout.....                           | 14        |
| 2.2 Function Testing.....                                  | 15        |
| 2.2.1 Loopback Test.....                                   | 15        |
| 2.2.2 Web Server Test .....                                | 21        |
| 2.2.3 SMTP Test.....                                       | 22        |
| 2.3 Troubleshooting Guide .....                            | 23        |
| 2.3.1 Ping.....  | 23        |
| 2.3.2 Misc.....  | 24        |
| <b>3. Programmer's Guide.....</b>                          | <b>25</b> |
| 3.1 API Function .....                                     | 25        |
| 3.1.1 Type of Functions .....                              | 25        |
| 3.2 Sample Source Codes .....                              | 40        |
| 3.2.1 Loopback & TCP Server .....                          | 40        |
| 3.2.1.1 Source Codes : \Software\Firmware\DirectMode\..... | 40        |
| LB_TCP_SERVER\.....  | 40        |
| 3.2.1.2 Flow Diagram .....                                 | 41        |
| 3.2.2 Loopback & TCP Client .....                          | 42        |
| 3.2.2.1 Source Codes : \Software\Firmware\DirectMode.....  | 42        |

---

|  |           |
|--|-----------|
| \LB_TCP_CLIENT\ .....  | 42        |
| 3.2.2.2 Flow Diagram .....   | 42        |
| 3.2.3 Loopback & UDP .....   | 43        |
| 3.2.4 Web Server .....   | 44        |
| 3.2.5 SMTP_WEB .....   | 46        |
| 3.2.6 DHCP .....   | 46        |
| 3.2.6.1 Source Codes : \Software\Firmware\DirectMode\DHCP\ .....               | 47        |
| 3.2.6.2 Flow Diagram .....   | 47        |
| 3.3 Application Development Procedure .....                                    | 49        |
| 3.3.1 Program Developing Procedure (based on the KEIL compiler).....           | 49        |
| 3.3.2 Program Downloading and Running Procedure (based on Flip by<br>ATMEL) 50 |           |
| 3.3.3 Memory Map .....   | 53        |
| <b>4. Hardware Designer's Guide .....</b>                                      | <b>54</b> |
| 4.1 EVB8051 Schematic .....  | 54        |
| 4.2 PAL .....  | 54        |
| 4.3 Parts List .....   | 57        |
| <b>Appendix A. Quick Testing Procedure.....</b>                                | <b>58</b> |
| A.1 Loopback Test .....  | 58        |
| A.2 Web Server Test .....  | 58        |
| <b>Appendix B. Specification of Serial Cables .....</b>                        | <b>59</b> |
| <b>Appendix C. Specification of IIM7010A.....</b>                              | <b>60</b> |
| C.1 Advantages .....   | 60        |
| C.2 Components .....   | 60        |
| C.3 Block Diagram .....  | 60        |
| C.4 Module dimension .....   | 61        |
| C.5 Pin description .....  | 62        |
| <b>Appendix D. Specification of IIM7000A.....</b>                              | <b>67</b> |
| C.1 Advantages .....   | 67        |

---

---

|     |                       |    |
|-----|-----------------------|----|
| C.2 | Components.....       | 67 |
| C.3 | Block Diagram .....   | 67 |
| C.4 | Module dimension..... | 68 |
| C.5 | Pin description.....  | 69 |

---

# Figures

|   |    |
|---|----|
| <FIG. 1: EVB8051 PACKAGE> .....   | 1  |
| <FIG. 2: ITEMS CONTAINED IN THE EVB8051> .....                          | 2  |
| <FIG. 3: EVB8051> .....   | 3  |
| <FIG. 4: IIM7010A> .....  | 3  |
| <FIG. 5: USER'S MANUAL> .....   | 4  |
| <FIG. 6: POWER ADAPTOR (5V)> .....                                      | 4  |
| <FIG. 7: SOFTWARE CD> .....   | 5  |
| <FIG. 8: UTP CABLE> .....   | 5  |
| <FIG. 9: SERIAL CABLE> .....  | 6  |
| <FIG. 10: DIRECTORY STRUCTURE OF THE SOFTWARE CD> .....                 | 7  |
| <FIG. 11: SYSTEM CONFIGURATION BETWEEN EVB8051 AND PC> .....            | 9  |
| <FIG. 12: BOOT-UP MESSAGE OF EVB8051> .....                             | 11 |
| <FIG. 13: NETWORK CONFIGURATION MENU> .....                             | 11 |
| <FIG. 14: LAYOUT OF EVB8051> .....                                      | 14 |
| <FIG. 15: RUNNING OF AX1.EXE PROGRAM> .....                             | 16 |
| <FIG. 16: INPUT OF THE CONNECTION INFORMATION> .....                    | 16 |
| <FIG. 17: CONNECTION SETUP COMPLETED> .....                             | 16 |
| <FIG. 18: FILE TRANSFER> .....  | 17 |
| <FIG. 19: LOOPBACK TEST IN SUCCESSION> .....                            | 17 |
| <FIG. 20: INPUT OF THE LISTEN PORT > .....                              | 18 |
| <FIG. 21: STARTING SCREEN OF TCP CLIENT> .....                          | 18 |
| <FIG. 22: STARTING SCREEN OF UDP LOOPBACK> .....                        | 19 |
| <FIG. 23: OPEN UDP SOCKET> .....  | 19 |
| <FIG. 24: INPUT OF THE EVB8051 INFORMATION AND DATA FORMAT> .....       | 20 |
| <FIG. 25: UDP TEST IN SUCCESSION > .....                                | 20 |
| <FIG. 26: HOME PAGE FOR WEB PAGE TEST OF THE EVB8051> .....             | 21 |
| <FIG. 27: STARTING SCREEN FOR THE DEMO SMTP PAGE OF THE EVB8051 > ..... | 23 |
| <FIG. 28: uVISION-51> .....   | 49 |

---

|                                       |    |
|---------------------------------------|----|
| <FIG. 29: MAKING A NEW PROJECT> ..... | 50 |
| <FIG. 30: SETTING RS232>.....         | 50 |
| <FIG. 31: FLIP BY ATMEL>.....         | 52 |
| <FIG. 32: MEMORY OF EVB8051>.....     | 53 |

## Tables

|  |    |
|--|----|
| <TABLE1: LIST OF ITEMS CONTAINED IN THE EVB8051> .....               | 2  |
| <TABLE2: LIST OF COMMAND FOR NETWORK CONFIGURATION FOR EVB8051>..... | 12 |



---

# 1. Getting Started

## 1.1 EVB8051 Package

### 1.1.1 Components

The EVB8051 is packaged as shown in <Fig. 1>, and its contents are as shown in <Fig. 2>.



<Fig. 1: EVB8051 Package>

The EVB8051 contains the items described in the table below. Photographs of the items are shown in <Fig. 3> through <Fig. 11>.

**<Table 1: List of Items Contained in the EVB8051>**

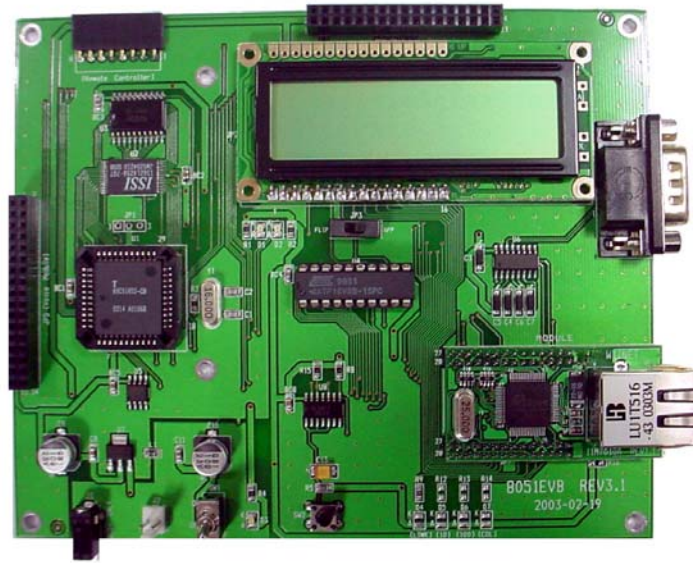
| No. | Item                       | Quantity |
|-----|----------------------------|----------|
| 1   | EVB8051 (Plugged IIM7010A) | 1        |
| 2   | User's Manual (EVB8051)    | 1        |
| 3   | Power Adaptor (5V)         | 1        |
| 4   | Software CD                | 1        |
| 5   | UTP Cable                  | 1        |
| 6   | Serial Cable               | 1        |



**<Fig. 2: Items contained in the EVB8051>**

---

<Fig. 3> shows the EVB8051 Board. It includes a LCD that is useful for testing the functions of the web server and for debugging.



**<Fig. 3: EVB8051>**

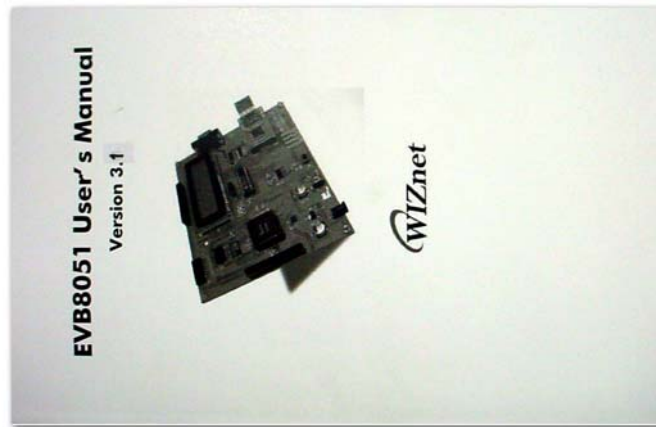
<Fig. 4> shows a IIM7010A plugged in the EVB8051. For more information, please refer to Appendix C. Specification of IIM7010A.



**<Fig. 4: IIM7010A>**

---

<Fig.5> shows the User's Manual of the EVB8051.



<Fig. 5: User's Manual>

<Fig. 6> shows the 5V Power Adaptor for supplying power to the EVB8051.



<Fig. 6: Power Adaptor (5V)>

---

<Fig. 7> shows the Software CD provided with the EVB8051. It contains Documents, Schematics, Source Code, and Software.



**<Fig. 7: Software CD>**

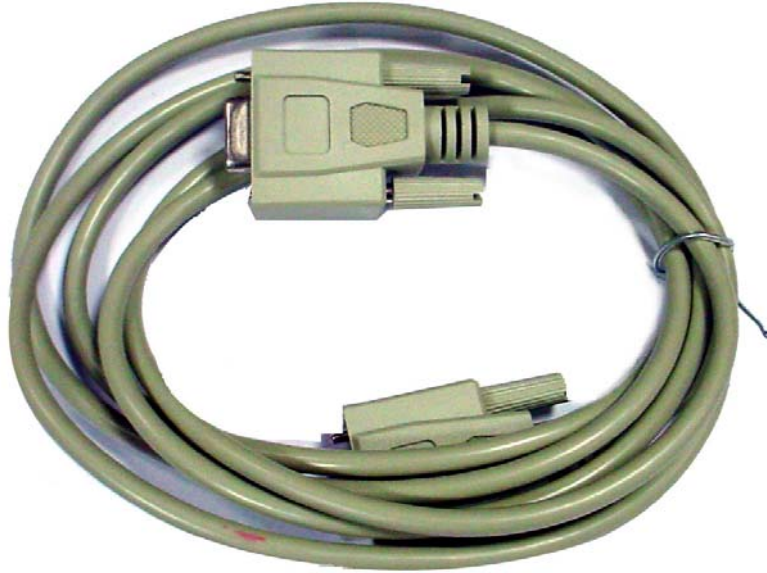
<Fig. 8> shows the UTP Cable (Crossed Cable) for connecting the EVB8051 directly to the PC.



**<Fig. 8: UTP Cable>**

---

<Fig. 9> shows the Serial Cable (Female-to-Female) for connecting the EVB8051 to the PC. It is used for monitoring and program downloading. Please refer to Appendix B. Specification of serial cable.

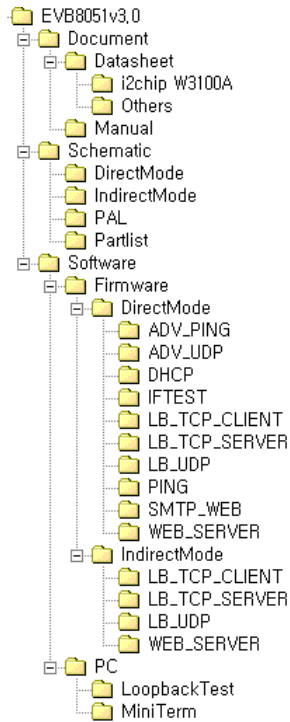


**<Fig. 9: Serial Cable>**

---

## 1.1.2 Software CD

The EVB8051 is supplied with a Software CD that contains various development tools including Documents, Schematics, Source Codes, and PC Softwares. <Fig. 10> shows the directory structure of the Software CD.



<Fig. 10: Directory Structure of the Software CD>

### 1.1.2.1 Document

Contains the data sheets of essential parts, including the W3100A. User manual is also included.

### 1.1.2.2 Schematic

Contains the circuit diagram of the EVB8051. The PAL subdirectory contains the PAL Source that is necessary for interfacing the 8051 MCU and the W3100A.

---

### **1.1.2.3 Software**

Software is provided for the Firmware and PC applications. The software for the Firmware contains the W3100A API Driver for the 8051 and some samples of application source code. The software for PC applications contains Loopback Test program and Serial Terminal program for Windows.

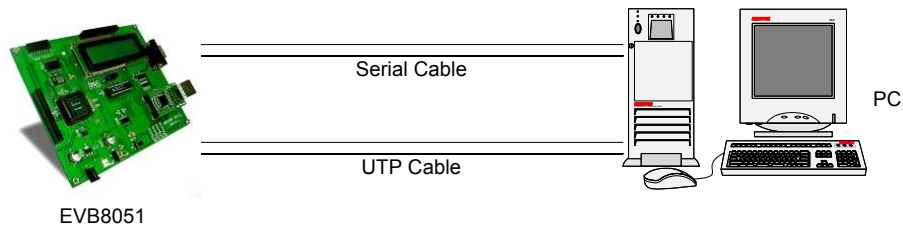
---

## 1.2 System Configuration

### 1.2.1 PC Setup

#### 1.2.1.1 Connecting Cables

For testing the functions of the EVB8051 and for application development, the system should be configured as shown in <Fig. 11>. First, the EVB8051 is connected to the PC using the crossed UTP Cable (for data transmission) and the Serial Cable (for monitoring and for program downloading).



<Fig. 11: System Configuration between EVB8051 and PC>

#### 1.2.1.2 Network Configuration

For convenience of development, the EVB8051 contains the following default network information:

- IP address: 192.168.0.2
- MAC address: 00-08-DC-00-00-00
- Gateway address: 192.168.0.1
- Subnet Mask: 255.255.255.0

The above information contained in the EVB8051 can be modified at any time to suit the developer's purpose.

First, for testing purposes, set the PC network information as follows:

- IP address: 192.168.0.5
- Gateway address: 192.168.0.1
- Subnet Mask: 255.255.255.0

After the above setup, confirm the operation of the EVB8051 on the PC using the Ping command.

```
C:\> ping 192.168.0.2 -t
```

If the connection has been set up properly, the following message will be displayed on the

---

screen:

```
Pinging 192.168.0.2 with 32 bytes of data:  
Reply from 192.168.0.2: bytes=32 time<10ms TTL=128  
Reply from 192.168.0.2: bytes=32 time<10ms TTL=128  
Reply from 192.168.0.2: bytes=32 time<10ms TTL=128  
:
```

If the connection has not been set up properly, the following message will be displayed on the screen:

```
Pinging 192.168.0.2 with 32 bytes of data:  
Request timed out.
```

In this case, please refer to Troubleshooting Guide 2.3.1.

### 1.2.1.3 Changing network configuration of EVB8051.

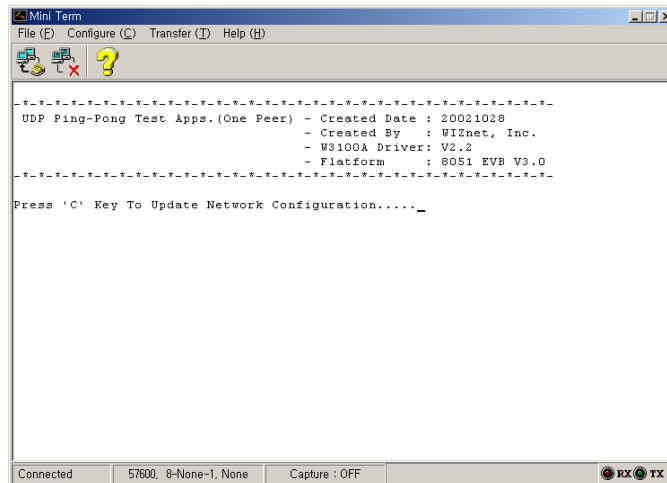
When EVB8051 boots up, you can change the network configuration of EVB8051 by serial. After you configure your PC as shown in <Fig. 11>, Run MiniTerm Program on your PC.

Set serial configuration with following values on PC.

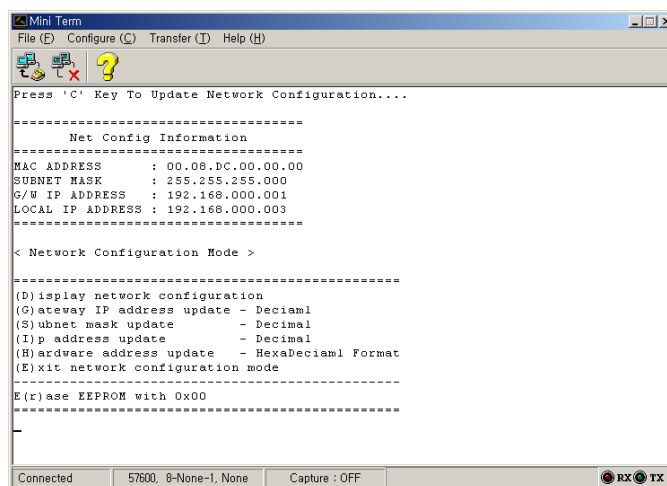
|              |       |
|--------------|-------|
| Speed        | 57600 |
| Parity       | None  |
| Data bit     | 8     |
| Stop bit     | 1     |
| Flow control | None  |

And reset EVB8051.

You can see the boot-up message as shown in <Fig. 12>.



<Fig. 12: boot-up message of EVB8051>



<Fig. 13: Network Configuration Menu>

And when you press 'C', you can see the menu as shown in <Fig. 13>. The menu consists of the Command Set as shown in <Table 2>.

---

**<Table 2: List of Command for Network Configuration for EVB8051>**

| Command | Meaning                        |
|---------|--------------------------------|
| D, d    | Display network information    |
| G, g    | Set Default Gateway IP address |
| S, s    | Set Subnet Mask                |
| I, I    | Set EVB IP address             |
| H, h    | Set EVB MAC address            |
| E, e    | Exit menu and run program      |

---

### 1.2.1.3 Program Installation

Since the EVB8051 uses an ATMEL 8051 MCU, you can use the development tools (In-System-Programmer) provided by ATMEL. To download the tools required for development, visit the ATMEL site and download the latest version of the FLIP Software for installation.

[8051 – Architecture – Software]

<http://www.atmel.com/atmel/products/prod74.htm>

**FLIP Software** ([Download dev\\_tools3bc6c0cebce3f.zip now](#). 1.9M, updated Apr 24, 2002)

FLIP (Flexible In-system Programmer) software v1.6.0. Runs Windows 9x/Me/NT/2000/XP.

Supports RS232 or CAN link.

\* Note: Above information may differ, Please refer to recent information on it from ATMEL.

## 1.2.2 Evaluation Board Configuration

### 1.2.2.1 W3100A mode setting

The W3100A's mode has been fixed to CLOCKED mode as default.

### 1.2.2.2 PHY mode setting

The PHY mode has been set as follows;

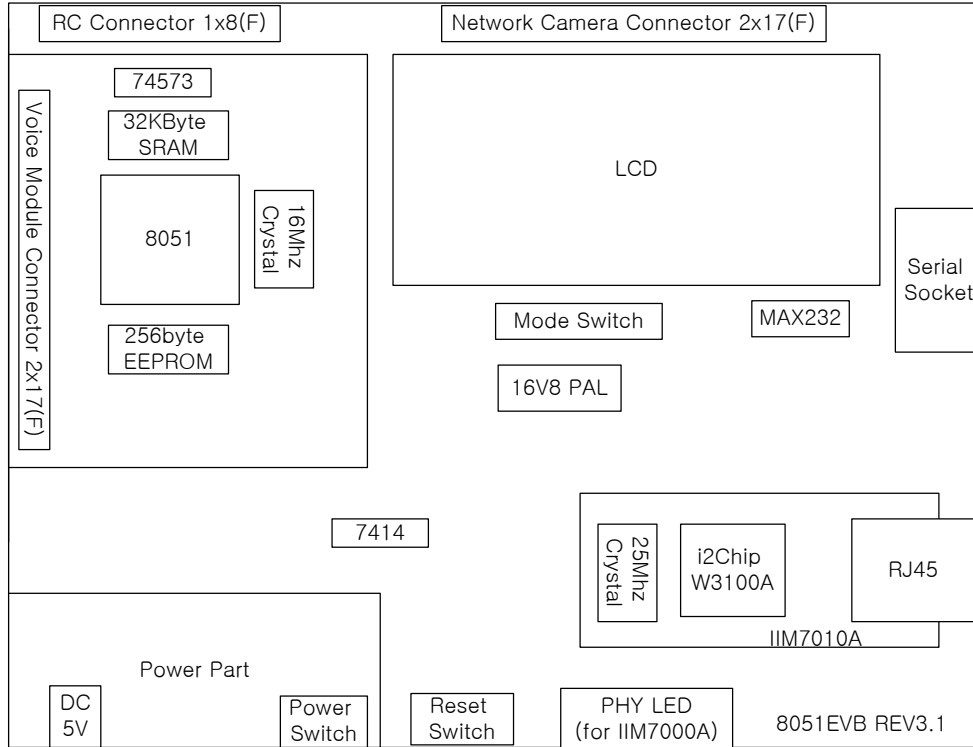
- Auto Negotiation: Yes
- Full Duplex: Yes
- Speed: 100Mbps

---

## 2. User's Guide

### 2.1 Evaluation Board Layout

<Fig. 14> illustrates the layout of the EVB8051 Board. On the upper left is the processor area that includes the 8051 MCU with 64 Kbytes of flash memory and 32 Kbytes of SRAM. On the upper right is TEXT LCD area. The power section is located on the bottom left, which accepts 5V and supplies 5V and 3.3V to the board. On the bottom right, IIM7010A, that includes W3100A, PHY and MAC jack with transformer, is located.



<Fig. 14: Layout of EVB8051>

---

## 2.2 Function Testing

### 2.2.1 Loopback Test

The Loopback is the operational mode for measuring the transmission performance of the i2Chip W3100A on the EVB8051. It is used for measuring data transfer speed when the EVB8051 board receives data from the PC and sends it back to the PC. And they are the sample code of TCP server mode and client mode.

#### 2.2.1.1 Configuration

Since the EVB8051 board is equipped with default Loopback execution code (TCP Server) in the internal flash memory, its operation can be verified immediately after a network has been set up.

First, install the Axinstall.exe program (located in the “\Software\PC\LoopbackTest” folder on the CD) on the PC. Once the AxinstallV2.1.EXE is installed, the Ax1.exe program is created and is required by the PC for Loopback testing. To run the Loopback program loaded on the EVB8051, slide the JP3 Slide Switch on the board to the right.

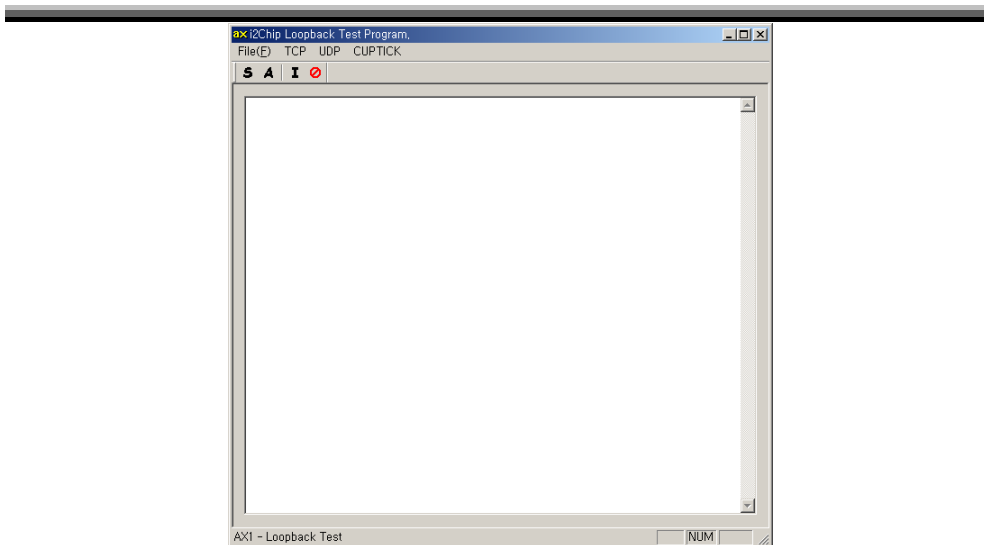
#### 2.2.1.2 Loopback TCP Server Test

Run the ping command from the PC to the EVB8051 to check network operation.

※ By default, the IP of the EVB8051 is set to 192.168.0.2.

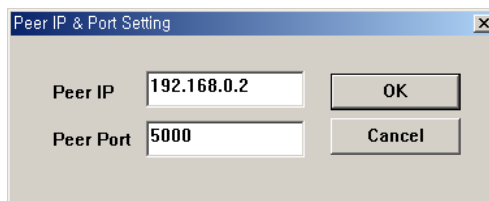
```
C:\> ping 192.168.0.2 -t
```

On the PC, run the AX1.exe program for connection setup. The screen will look like <Fig. 15>.



**<Fig. 15: Running of AX1.exe Program>**

From the 'TCP' menu of the AX1 program, select 'Connect' to display the dialog box as shown in <Fig. 16>. Enter the IP address assigned to the EVB8051 (192.168.0.2) and the Port (5000) number, and try the connection.



**<Fig. 16: Input of the Connection Information>**

Once a connection is set up between the EVB8051 and the computer, a box with the 'Connected' message appears as shown in <Fig. 17>.

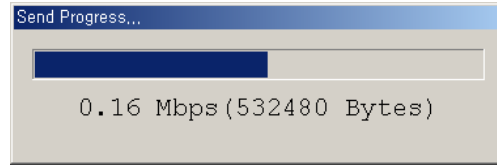


**<Fig. 17: Connection Setup Completed>**

---

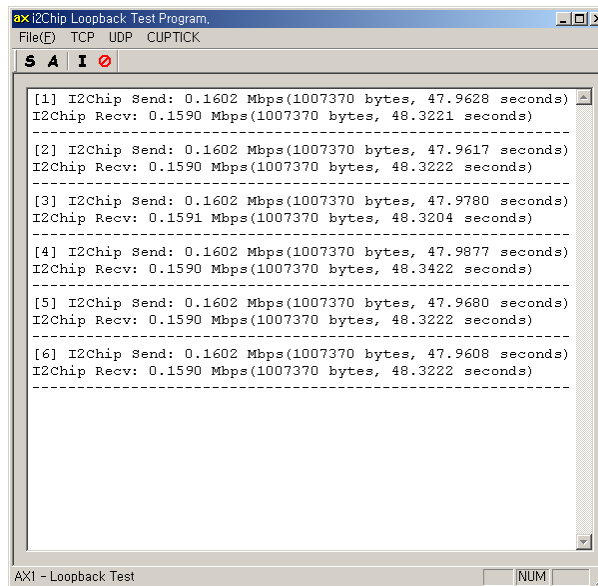
After the connection setup, select 'Send' from the 'File' menu. The dialog box for file transfer appears.

Select a file to start the loop back test. Refer to <Fig. 18>.



<Fig. 18: File Transfer>

※ You can perform the Loopback test successively using the 'A(uto)' command or the 'I(teration)' command. Make sure to perform the 'S(end)' command before the 'A(uto)' command or the 'I(teration)' command. <Fig. 19> shows the result of 'I(teration)' command execution.



<Fig. 19: Loopback Test in Succession>

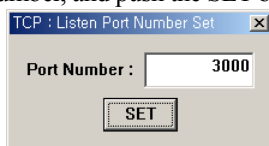
- ※ If the program does not run properly, try downloading the loopback program from the Software CD (\Software\Firmware\DirectMode\LB\_TCP\_SERVER\TCPS.HEX) into the EVB8051 again. For more information on program downloading, refer to Section 3.3.2.)

### 2.2.1.3 Loopback TCP Client Test

You can test Loopback TCP Client like Loopback Server.

On the PC, Run the AX1.exe program for connection setup.

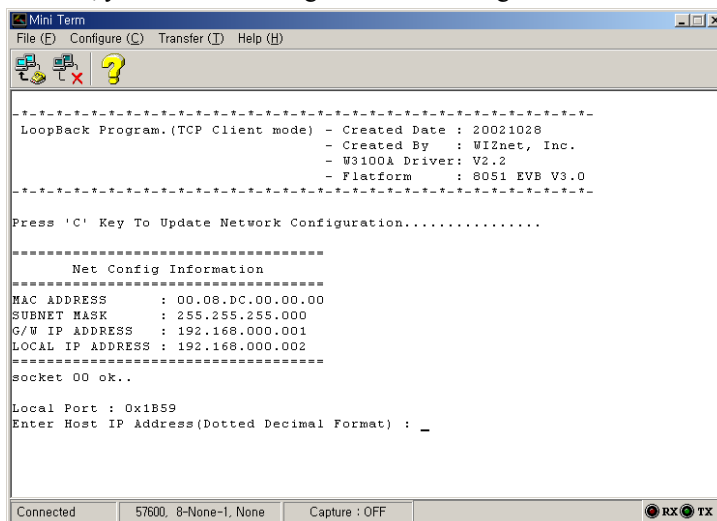
From the 'TCP' menu of the AX1 program, select 'Listen' to display the dialog box as shown in <Fig. 20>. Enter the Port (3000) number, and push the SET button.



<Fig. 20: Input of the Listen port >

Next, try downloading the loopback client program from the Software CD (\Software\Firmware\DirectMode\LB\_TCP\_CLIENT\TCPC.HEX) into the EVB8051. For more information on program downloading, refer to Section 3.3.2.)

After reset EVB8051, you can see the image as shown in <Fig 21>.



<Fig. 21: Starting Screen of TCP Client>

Input IP Address of your PC (ex : 192.168.0.5). then you can see a box with the 'Connected' message appears as shown in <Fig. 17>.

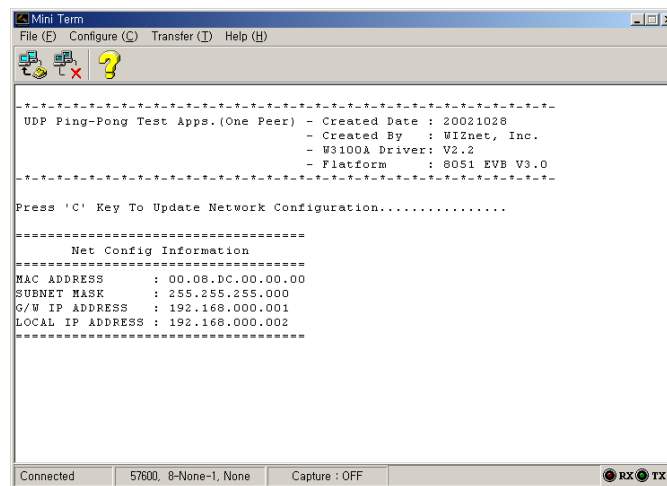
The next flow for test is same with Loopback TCP Server.

#### 2.2.1.4 Loopback UDP Test

Loopback UDP is the sample code of UDP.

Download the UDP loopback program from the Software CD (\Software\Firmware\DirectMode\LB\_UDP \UDP.HEX) into the EVB8051. For more information on program downloading, refer to Section 3.3.2.)

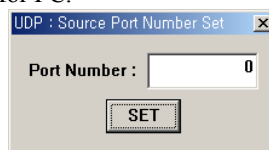
After reset EVB8051, you can see the image as shown in <Fig 22>.



<Fig. 22: Starting Screen of UDP Loopback>

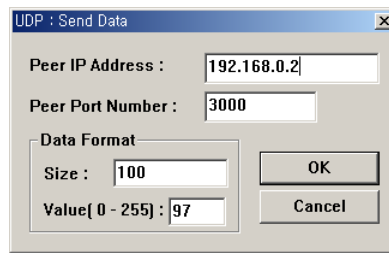
On the PC, Run the AX1.exe program.

From the 'UDP' menu of the AX1 program, select 'Open' to display the dialog box as shown in <Fig. 23>. And set port # of UDP for PC.



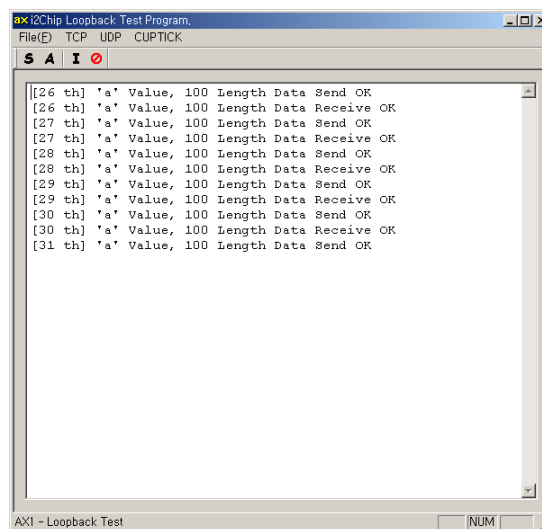
<Fig. 23: Open UDP Socket>

Select 'Send' to display the dialog box as shown in <Fig. 24>. Enter Peer IP Address, port #, data size and value for UDP loopback test of EVB8051. And push the OK button.



<Fig. 24: Input of the EVB8051 Information and Data Format>

Then, you can see the image as shown in <Fig. 25>. Now PC and EVB8051 are sending and receiving each other.



<Fig. 25: UDP Test in Succession >

---

## 2.2.2 Web Server Test

### 2.2.2.1 Outline

The EVB8051 provides the source code to control the equipment through the web and is available for developing applications that require web server functions.

### 2.2.2.2 Testing Procedure

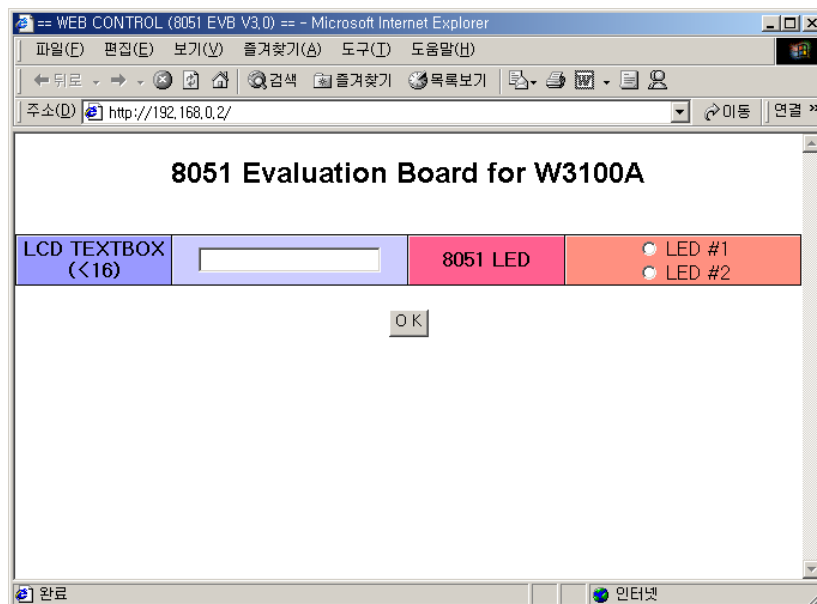
The test method for the web server is the same as for the Loopback test.

Download the web server program “\Software\Firmware\DirectMode\WEB\_SERVER\HTTPD.HEX” from the Software CD to the EVB8051 for testing. For more information on program downloading, refer to Section 3.3.2.

Run the ping command to the EVB8051 to check network operation. By default, the IP address of the EVB8051 is set to 192.168.0.2.

If the Ping command works properly, run the web browser on the PC and enter the IP address of the EVB8051 (192.168.0.2) in the URL window to attempt to access the web server of the EVB8051.

If the EVB8051 is running in web server mode, the starting screen of the web page will look like <Fig. 26>.



<Fig. 26: Home page for Web Page test of the EVB8051>

---

### 2.2.2.3 Functions of the Home page

- (1) LCD Character Display  
Entering characters in the LCD Text Box on the demo page will display the characters on the LCD of the EVB8051.
- (2) LED Remote Control  
It controls the LEDs on the board through the web. In actual applications, it can be used for controlling other devices than the LED in remote places through the web. Selecting LED#1 and LED#2 in <Fig. 26> will turn the LEDs (D1, D2) on the EVB8051 ON/OFF.

### 2.2.3 SMTP Test

#### 2.2.3.1 Outline

The EVB8051 provides the source code to send E-Mail(SMTP) through the web and is available for developing applications that require SMTP functions.

#### 2.2.3.2 Testing Procedure

The test method for the SMTP is the same as for the Web Server test.

Download the web server program “\Software\Firmware\DirectMode\SMTP\_WEB\SMTPWEB.HEX” from the Software CD to the EVB8051 for testing. For more information on program downloading, refer to Section 3.3.2.

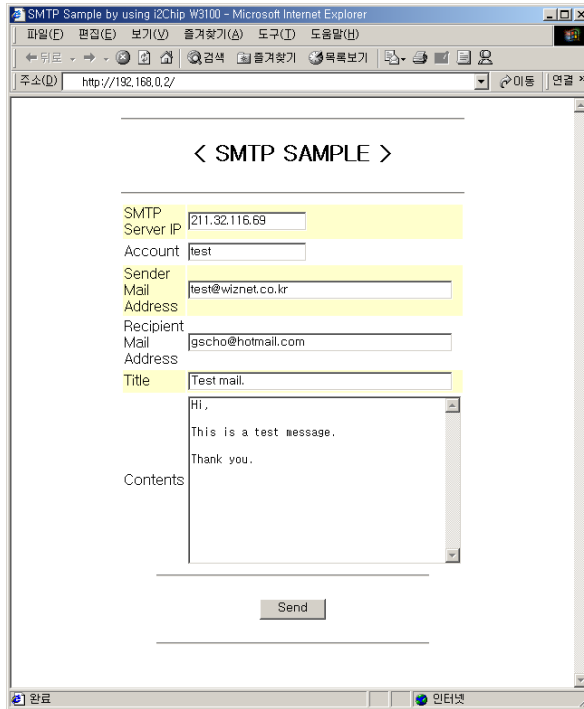
Run the ping command to the EVB8051 to check network operation. By default, the IP address of the EVB8051 is set to 192.168.0.2.

If the Ping command works properly, run the web browser on the PC and enter the IP address of the EVB8051 (192.168.0.2) in the URL window to attempt to access the web server of the EVB8051.

If the EVB8051 is running in SMTP mode, the starting screen of the web page will look like <Fig. 27>.

Enter SMTP Server IP, Account, Sender Mail Address, Recipient Mail Address, Title, Contents. And push Send button. When completed, you can see the initial screen.

\* Note: For this function test, SMTP server is reachable from EVB8051 and EVB8051 should be connected Internet.



<Fig. 27: Starting Screen for the Demo SMTP Page of the EVB8051 >

## 2.3 Troubleshooting Guide

### 2.3.1 Ping

When you cannot reach EVB8051 by Ping command,

Step 1. Did you connect correctly between test PC and EVB8051 with UTP cable?

Step 2. Did you change your test PC's network environment (IP address, Gateway, Subnet)?

If no, you should change it first as follows:

IP address: 192.168.0.5

Gateway address: 192.168.0.1

Subnet Mask: 255.255.255.0

Step 3. Whether IIM7010A's Link LED(D4) is on?

If off, you'd better check whether the UTP cable works correctly.

---

## 2.3.2 Misc.

2.3.2.1 When the screen remains blank with the power on after a connection is made

- Step 1. Check the connection condition of the serial cable.
- Step 2. Check if the COM Port numbers of the PC and terminal coincide.
- Step 3. Check the terminal configuration.

---

## 3. Programmer's Guide

### 3.1 API Function

#### 3.1.1 Type of Functions

- (1) Internal Function: Used inside the driver function
- (2) API Function: Used in applications

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>void Int0(void) interrupt 0</b>   |
| <b>Arguments</b>     | None   |
| <b>Return value</b>  | None   |
| <b>Description</b>   | Interrupt handling function of the W3100A.<br>Stores the status information that each function waits for in the global variable S_STATUS for transfer.<br>S_STATUS stores the interrupt status value for each channel. |
| <b>Category</b>      | Internal Function  |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>void ISR_ESTABLISHED(SOCKET s)</b>  |
| <b>Arguments</b>     | s: Channel number  |
| <b>Return value</b>  | None   |
| <b>Description</b>   | Established connection interrupt handling function.<br>Called upon connection establishment, and may be inserted in user code if needed by the programmer. |
| <b>Category</b>      | Internal Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void ISR_CLOSED(SOCKET s)</b>  |
| <b>Arguments</b>     | s: Channel number   |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Closed connection interrupt handling function.<br>Called upon connection closure, and may be inserted in user code if needed by the programmer. |

---

| Category | Internal Function |
|----------|-------------------|
|----------|-------------------|

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void ISR_RX(SOCKET s)</b>  |
| <b>Arguments</b>     | s: Channel number   |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Received data interrupt handling function.<br>Called upon receiving data, and may be inserted in user code if needed by the programmer. |
| <b>Category</b>      | Internal Function   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void init W3100A(void)</b>   |
| <b>Arguments</b>     | None  |
| <b>Return value</b>  | None  |
| <b>Description</b>   | W3100A initialization function.<br>Function for S/W resetting of the W3100A.<br>Sets the initial SEQ# to be used for TCP communication. |
| <b>Category</b>      | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void sysinit(u_char sbufsize, u_char rbufsize)</b>   |
| <b>Arguments</b>     | Sbufsize:tx memory size   |
| <b>Return value</b>  | rbufsize:rx memory size   |
| <b>Description</b>   | W3100A initialization function.<br>Sets the source MAC, source IP, gateway, and subnet mask to be used by the W3100A to the designated values.<br>May be called when setting the concerned register to modify network information and reflect it on the W3100A. |
| <b>Category</b>      | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void setsubmask(u_char * addr)</b>                         |
| <b>Arguments</b>     | addr: Pointer having the value for setting up the subnet mask |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Subnet mask setup function                                    |

|                 |              |
|-----------------|--------------|
| <b>Category</b> | API Function |
|-----------------|--------------|

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>void setgateway(u_char * addr)</b>                        |
| <b>Arguments</b>     | addr: Pointer having the value for setting up the gateway IP |
| <b>Return value</b>  | None   |
| <b>Description</b>   | Gateway IP setup function                                    |
| <b>Category</b>      | API Function   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void setIP(u_char * addr)</b>                                    |
| <b>Arguments</b>     | addr: Pointer having the value for setting up the source IP address |
| <b>Return value</b>  | None  |
| <b>Description</b>   | W3100A IP address setup function                                    |
| <b>Category</b>      | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void setMACAddr(u_char * addr)</b>                         |
| <b>Arguments</b>     | addr: Pointer having the value for setting up the MAC address |
| <b>Return value</b>  | None  |
| <b>Description</b>   | MAC address setup function                                    |
| <b>Category</b>      | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void settimeout(u_char * val)</b>  |
| <b>Arguments</b>     | val: Pointer having the value for setting up the timeout.<br>Upper 2 bytes have the initial timeout value, while the last 1 byte has the number of retransmissions until timeout.   |
| <b>Return value</b>  | None  |
| <b>Description</b>   | TCP timeout setup function.<br>Used for adjusting the TCP retransmission time.<br>A timeout interrupt takes place when retransmission is attempted for establishing the connection or for data transfer beyond the set value. |
| <b>Category</b>      | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void setINTMask(u_char mask)</b>   |
| <b>Arguments</b>     | mask: Value of the mask to be set ('1' refers to interrupt enable)          |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Interrupt mask setup function.<br>Enables/disables the concerned interrupt. |
| <b>Category</b>      | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void setbroadcast(SOCKET s)</b>  |
| <b>Arguments</b>     | s: Channel number   |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Broadcast data transfer enable setup function<br>Enables/disables broadcasting data transfer in UDP or IP RAW mode. |
| <b>Category</b>      | API Function  |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>void setTOS(SOCKET s, u_char tos)</b>                                     |
| <b>Arguments</b>     | s: Channel number<br>tos: Value to be set for the TOS field of the IP header |
| <b>Return value</b>  | None   |
| <b>Description</b>   | Handles protocol setup function in IP RAW mode                               |
| <b>Category</b>      | API Function   |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>char socket(SOCKET s, u_char protocol, u_int port, u_char flag)</b>   |
| <b>Arguments</b>     | s: Channel number<br>protocol: Protocol designated for the channel<br>SOCK_STREAM(0x01) -> TCP<br>SOCK_DGRAM(0x02) -> UDP<br>SOCK_IPL_RAW(0x03) -> IP Layer RAW<br>SOCK_MACL_RAW(0x04) -> MAC Layer RAW<br>port: Source port designated for the channel<br>flag: Options designated for the channel<br>SOCKOPT_BROADCAST(0x80) -> '1' refers to broadcast data transfer in UDP mode<br>SOCKOPT_NDTIMEOUT(0x40) -> '1' refers to use of only the register that designates the timeout value<br>SOCKOPT_NDACK(0x20) -> '1' refers to the delayed ACK not to be used<br>SOCKOPT_SWS(0x10) -> '1' refers to the silly window syndrome to be used |
| <b>Return value</b>  | Channel number if succeeded, or -1 if failed.  |

|                    |   |
|--------------------|---|
| <b>Description</b> | Initialization of the channel.<br>Initializes the designated channel and waits for completion of W3100A handling. |
| <b>Category</b>    | API Function  |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>char connect(SOCKET s, u_char * addr, u_int port)</b>   |
| <b>Arguments</b>     | s: Channel number<br>addr: Destination IP address<br>port: Destination port number   |
| <b>Return value</b>  | 1 if connection is established, or -1 if connection fails.   |
| <b>Description</b>   | Sets the connection to the designated peer.<br>Establishes a connection with a peer on the designated channel and waits until the connection is established. (TCP client mode) |
| <b>Category</b>      | API Function   |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>char listen(SOCKET s, u_char * addr, u_int * port)</b>  |
| <b>Arguments</b>     | s: Channel number<br>addr: Peer IP address at the time of connection establishment<br>port: Peer Port number at the time of connection establishment |
| <b>Return value</b>  | 1 if connection is established, or -1 if connection fails.   |
| <b>Description</b>   | Waits for connection with a peer. (Blocking Mode)<br>The designated channel waits for connection by a peer. (TCP Server mode)                        |
| <b>Category</b>      | API Function   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | char NListen(SOCKET s)  |
| <b>Arguments</b>     | s: Channel number   |
| <b>Return value</b>  | 1   |
| <b>Description</b>   | Waits for connection with a peer. (Non-blocking Mode)<br>The designated channel waits for connection by a peer. (TCP Server mode) |
| <b>Category</b>      | API Function  |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | void initseqnum(SOCKET s)  |
| <b>Arguments</b>     | s: Channel number  |
| <b>Return value</b>  | None   |
| <b>Description</b>   | Generates random values for the initial SEQ# to be used for establishing a TCP connection.<br>This function may be added to the code for generating random numbers for assigning a random number to initial SEQ# used in TCP.<br>In an actual internet environment, the initial SEQ# must be a random number. (A fixed number is used for EVB/DK.) |
| <b>Category</b>      | API Function   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | u_int send(SOCKET s, u_char * buf, u_int len) |
|----------------------|---|

|                     |  |
|---------------------|--|
| <b>Arguments</b>    | <p>s: Channel number</p> <p>buf: Pointer indicating the data to be sent</p> <p>len: Size of the data to be sent</p>  |
| <b>Return value</b> | Sent data size   |
| <b>Description</b>  | <p>Function for sending TCP data.</p> <p>Composed of the send() and send_in() functions.</p> <p>The send() function is an application I/F function. It continues to call the send_in() function to complete the sending of the data up to the size of the data to be sent when the application is called. The send_in() function receives the return value (the size of the data sent), calculates the size of the data to be sent, and calls the send_in() function again if there is any data left to be sent.</p> |
| <b>Category</b>     | API Function   |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>u_int send_in(SOCKET s, u_char * buf, u_int len)</b>  |
| <b>Arguments</b>     | s: Channel number<br>buf: Pointer indicating the data to be sent<br>len: Size of the data to be sent   |
| <b>Return value</b>  | Sent data size   |
| <b>Description</b>   | Internal function for sending TCP data.<br>Called by the send() function for TCP transmission.<br>It first calculates the free transmit buffer size and compares it with the size of the data to be transmitted to determine the transmission size.<br>After calculating the data size, it copies data from TX_WR_PTR.<br>It waits if there is a previous send command in process.<br>When the send command is cleared, it updates the TX_WR_PTR up to the size to be transmitted and performs the send command. |
| <b>Category</b>      | Internal Function  |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>u_int recv(SOCKET s, u_char * buf, u_int len)</b>   |
| <b>Arguments</b>     | s: Channel number<br>buf: Pointer where the data to be received is copied<br>len: Size of the data to be received  |
| <b>Return value</b>  | Received data size   |
| <b>Description</b>   | TCP data receiving function.<br>The recv() function is an application I/F function. It continues to wait for as much data as the application wants to receive. |
| <b>Category</b>      | API Function   |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>u_int sendto(SOCKET s, const u_char * buf, u_int len, u_char * addr, u_int port)</b>  |
| <b>Arguments</b>     | s: Channel number<br>buf: Pointer indicating the data to send<br>len: Size of the data to send<br>addr: Destination IP address |
| <b>Return value</b>  | Sent data size   |

|                    |   |
|--------------------|---|
| <b>Description</b> | <p>UDP data sending function.</p> <p>Composed of the sendto() and sendto_in() functions.</p> <p>The send() function is an application I/F function. It continues to call the send_in() function to complete the sending of the data up to the size of the data to be sent when the application is called.</p> <p>Unlike TCP transmission, it designates the destination address and the port.</p> |
| <b>Category</b>    | API Function  |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>u_int sendto_in(SOCKET s, const u_char * buf, u_int len)</b>   |
| <b>Arguments</b>     | s: Channel number<br>buf: Pointer indicating the data to send<br>len: Size of the data to send            |
| <b>Return value</b>  | Sent data size  |
| <b>Description</b>   | UDP data sending function.<br>An internal function that is the same as the send_in() function of the TCP. |
| <b>Category</b>      | Internal Function   |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>u_int recvfrom(SOCKET s, u_char * buf, u_int len, u_char * addr, u_int * port)</b>  |
| <b>Arguments</b>     | s: Channel number<br>buf: Pointer where the data to be received is copied<br>len: Size of the data to be received<br>addr: Peer IP address for receiving<br>port: Peer port number for sending |
| <b>Return value</b>  | Received data size   |
| <b>Description</b>   | UDP data receiving function.<br>Function for receiving UDP and IP layer RAW mode data, and handling the data header.   |
| <b>Category</b>      | API Function   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>char close(SOCKET s)</b>   |
| <b>Arguments</b>     | s: Channel number   |
| <b>Return value</b>  | 1   |
| <b>Description</b>   | Channel closing function.<br>Function for closing the connection of the designated channel. |
| <b>Category</b>      | API Function  |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>u_int select(SOCKET s, u_char func)</b>   |
| <b>Arguments</b>     | s: Channel number<br>func: SEL_CONTROL(0x00) -> return socket status<br>SEL_SEND(0x01) -> return free transmit buffer size<br>SEL_RECV(0x02) -> return data size in receive buffer |
| <b>Return value</b>  | Socket status or free transmit buffer size or received data size   |
| <b>Description</b>   | Function handling the channel socket information.  |
| <b>Category</b>      | API Function   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>u_int read_data(SOCKET s, u_char * src, u_char * dst, u_int len)</b>   |
| <b>Arguments</b>     | s: Channel number<br>src: Receive buffer pointer of the W3100A<br>dst: System buffer pointer<br>len: Data size to be copied |
| <b>Return value</b>  | Copied data size  |
| <b>Description</b>   | Copies the receive buffer data of the W3100A to the system buffer. It is called from the recv() or recvfrom() function.     |
| <b>Category</b>      | Internal Function   |

|                      |  |
|----------------------|--|
| <b>Function Name</b> | <b>u_int write_data(SOCKET s, u_char * src, u_char * dst, u_int len)</b>   |
| <b>Arguments</b>     | s: Channel number<br>src: System buffer pointer<br>dst: Transmit buffer pointer of the W3100A<br>len: Data size to be copied |
| <b>Return value</b>  | Copied data size   |
| <b>Description</b>   | Copies the system buffer data to the transmit buffer of the W3100A. It is called from the send_in() or sendto_in() function. |
| <b>Category</b>      | Internal Function  |

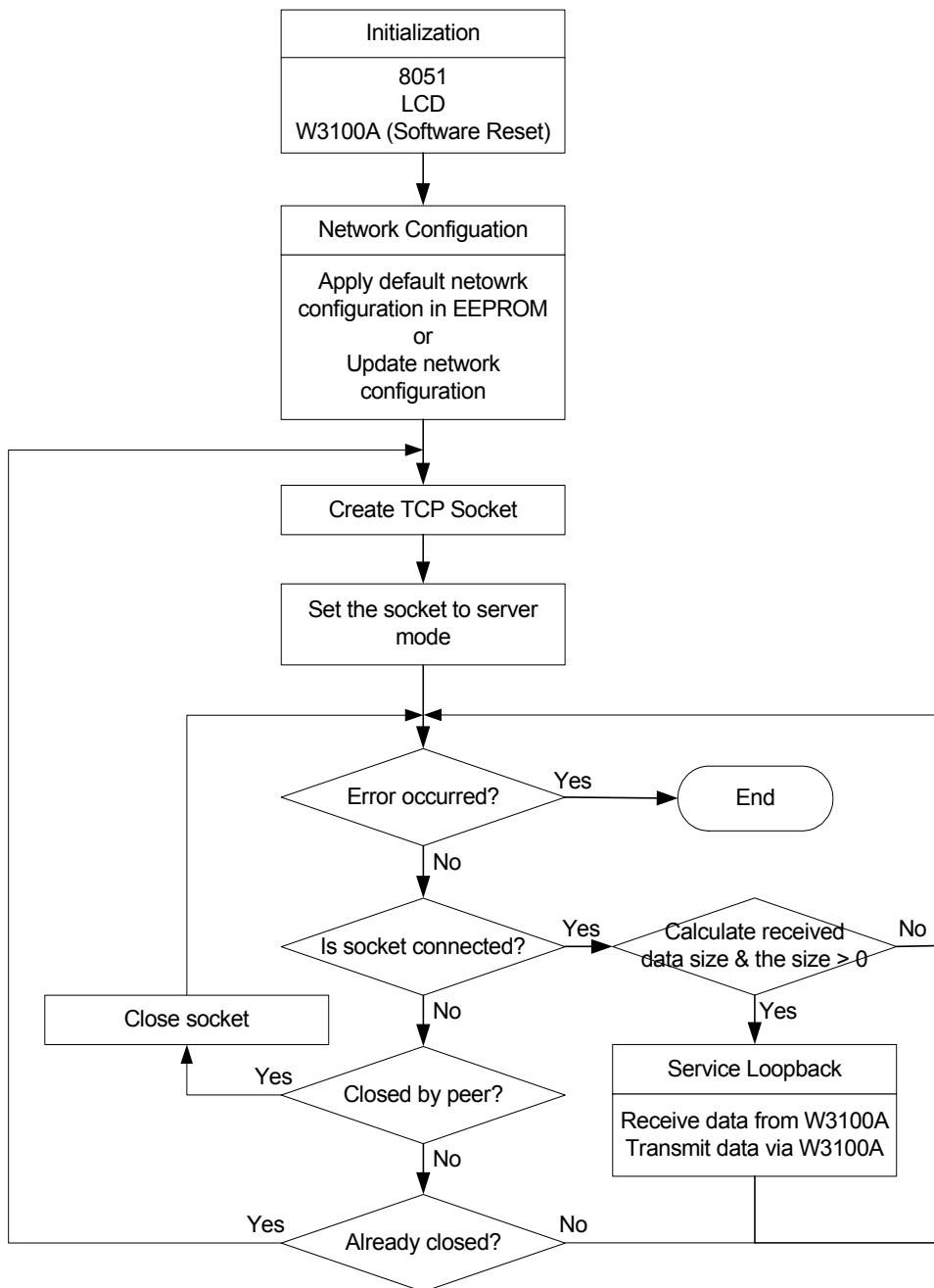
|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void wait_10ms(int cnt)</b>                      |
| <b>Arguments</b>     | cnt: count  |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Designates the delay.<br>Waits for 10 milliseconds. |
| <b>Category</b>      | Internal Function                                   |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void wait_1ms(int cnt)</b>                     |
| <b>Arguments</b>     | cnt: count  |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Designates the delay.<br>Waits for 1 millisecond. |
| <b>Category</b>      | Internal Function                                 |

|                      |   |
|----------------------|---|
| <b>Function Name</b> | <b>void wait_1us(int cnt)</b>                     |
| <b>Arguments</b>     | cnt: count  |
| <b>Return value</b>  | None  |
| <b>Description</b>   | Designates the delay.<br>Waits for 1 millisecond. |
| <b>Category</b>      | Internal Function                                 |



### 3.2.1.2 Flow Diagram



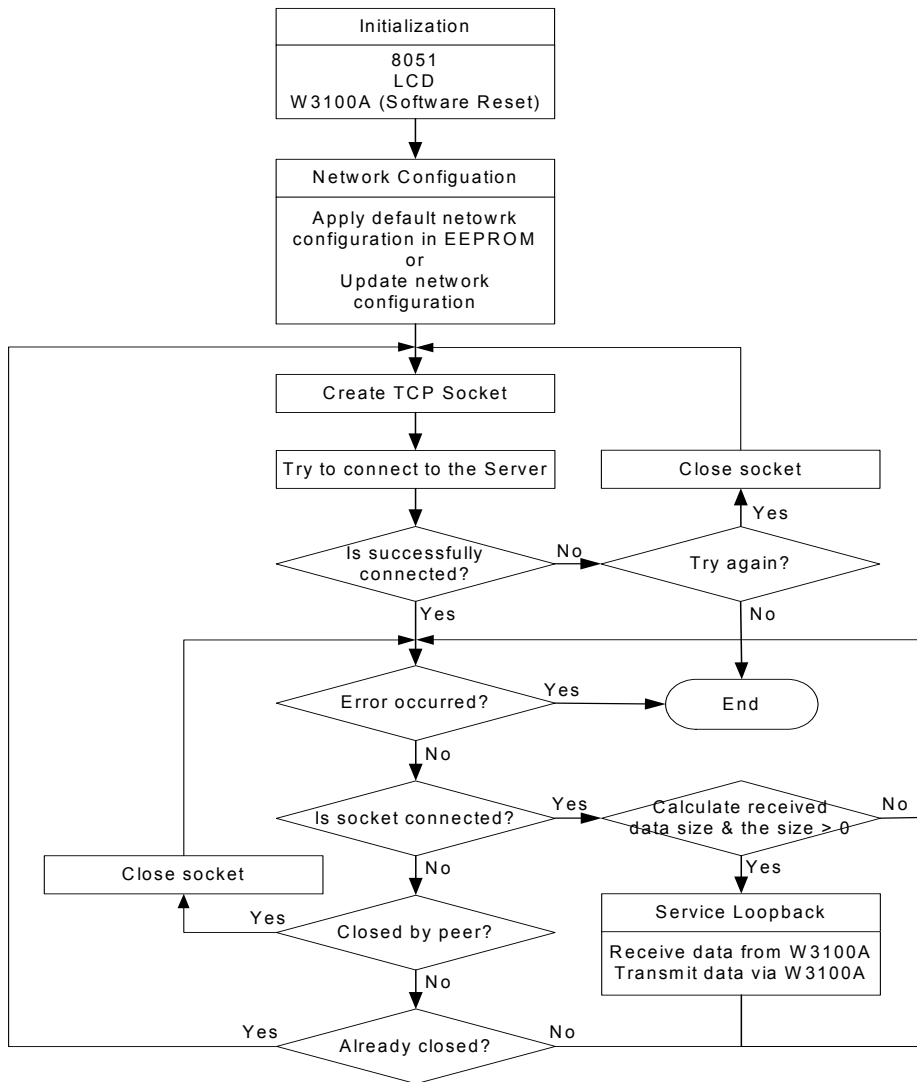
---

## 3.2.2 Loopback & TCP Client

### 3.2.2.1 Source Codes : \Software\Firmware\DirectMode

\LB\_TCP\_CLIENT\

### 3.2.2.2 Flow Diagram

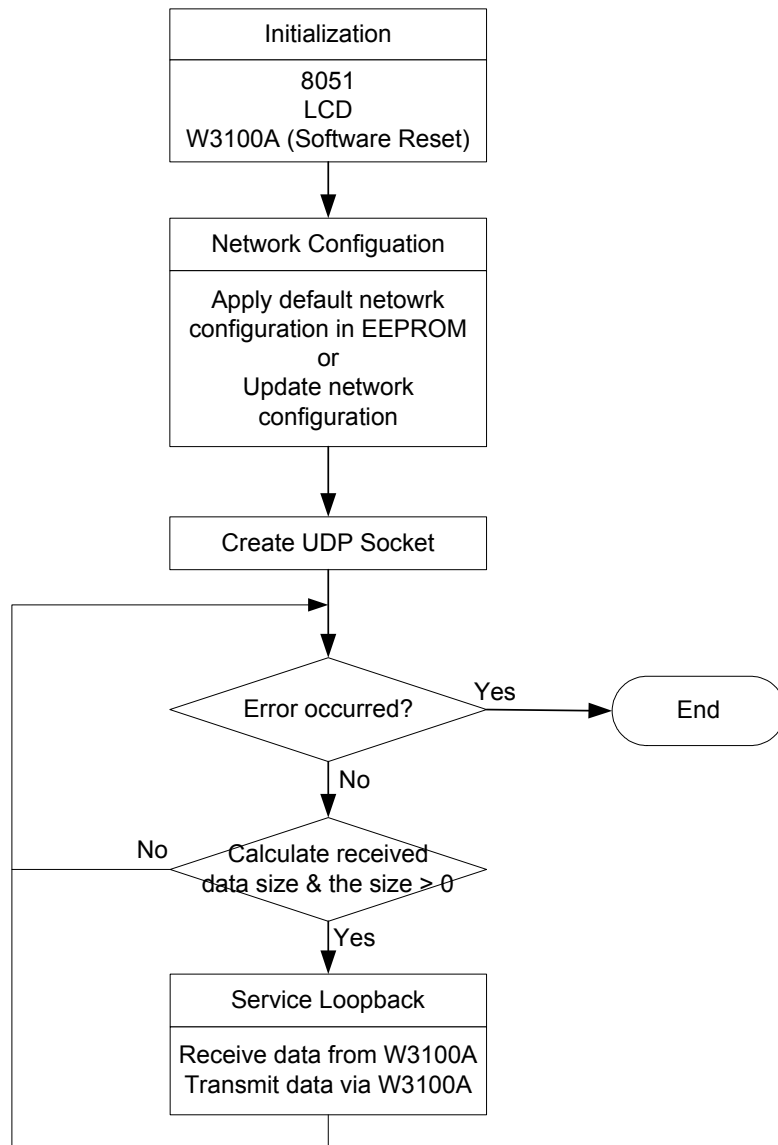


---

### 3.2.3 Loopback & UDP

3.2.3.1 Source Codes : \Software\Firmware\DirectMode\LB\_UDP\

#### 3.2.3.2 Flow Diagram

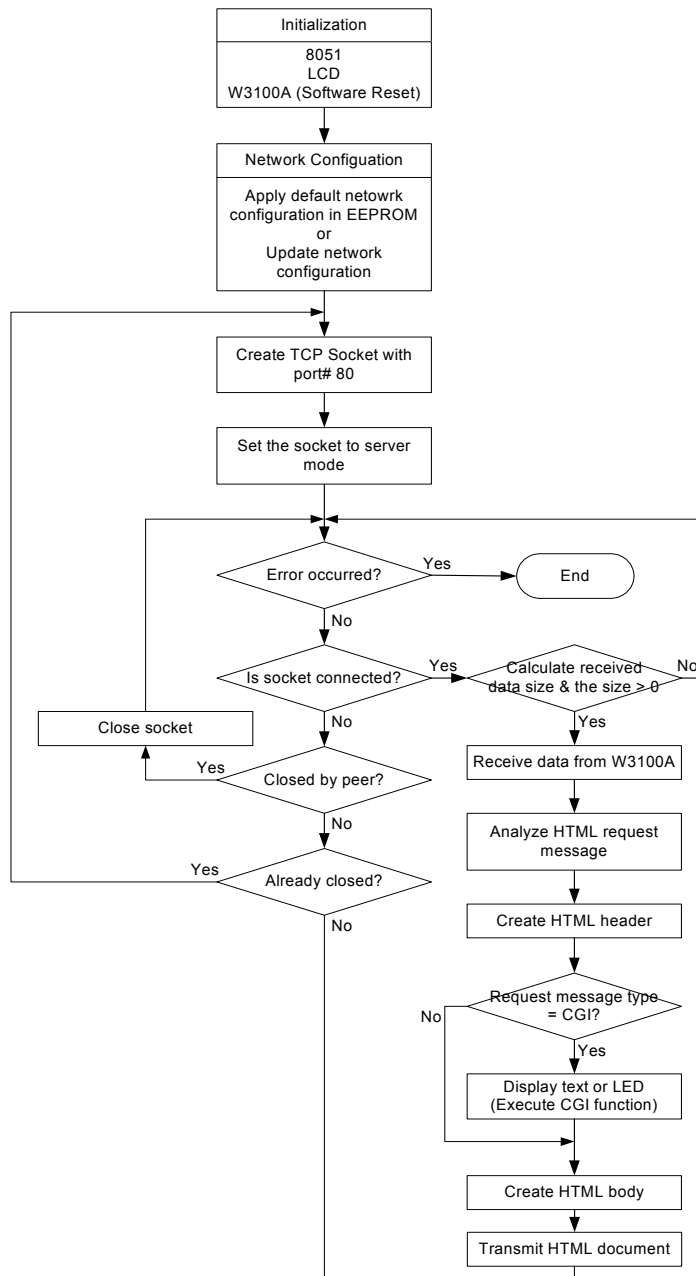


---

### 3.2.4 Web Server

3.2.4.1 Source Codes : \Software\Firmware\DirectMode\WEB\_SERVER\

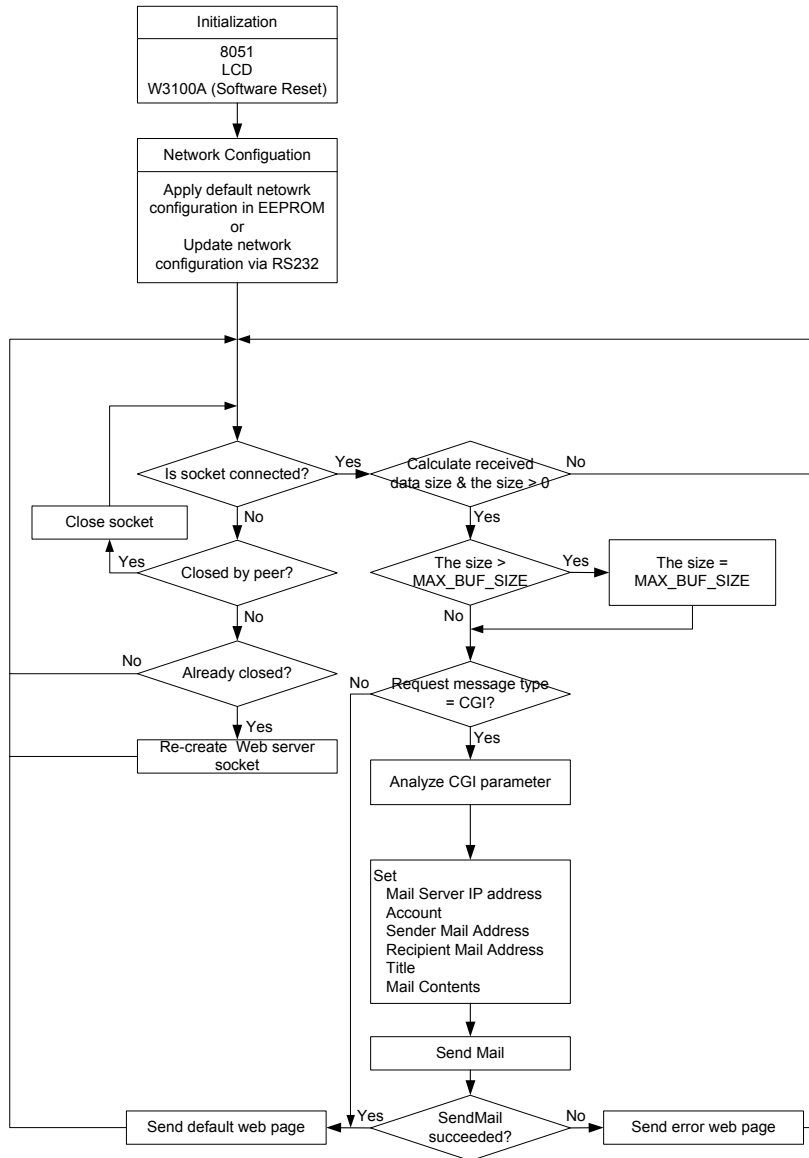
3.2.4.2 Flow Diagram



### 3.2.5 SMTP\_WEB

3.2.5.1 Source Codes : \Software\Firmware\DirectMode\SMTP\_WEB\

#### 3.2.5.2 Flow Diagram

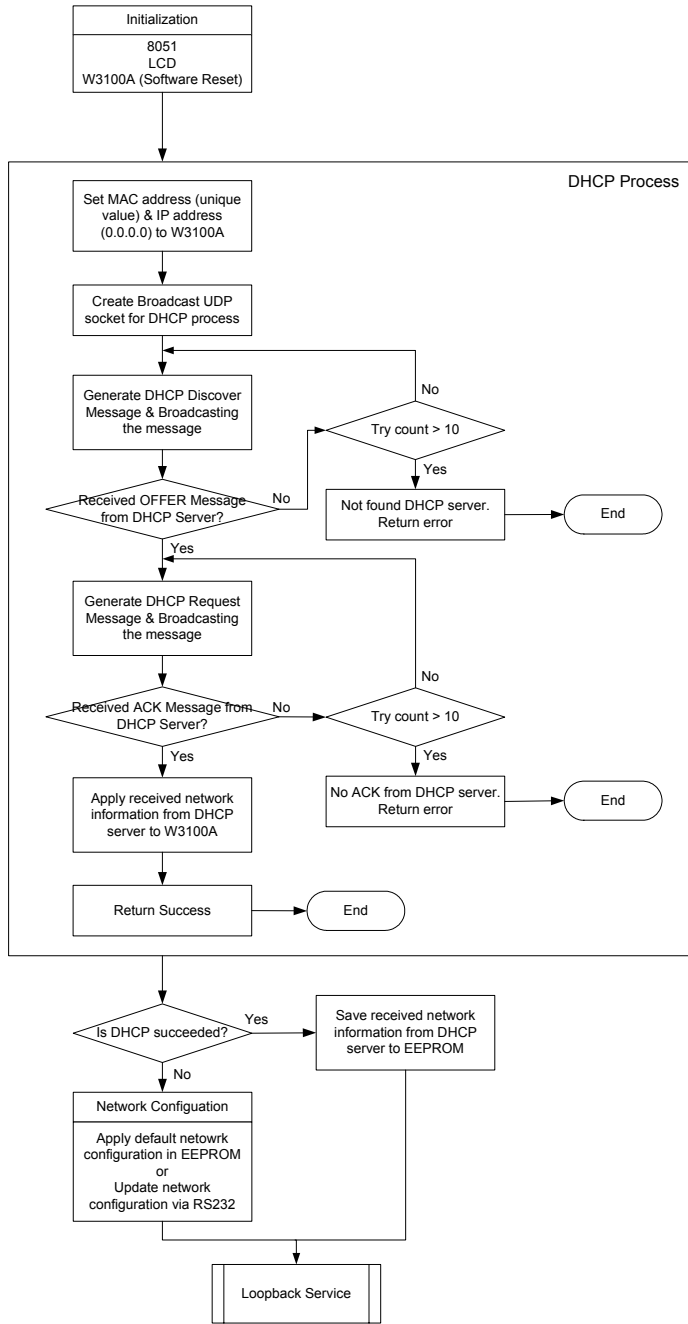


---

## 3.2.6 DHCP

3.2.6.1 Source Codes : \Software\Firmware\DirectMode\DHCP\

3.2.6.2 Flow Diagram



---

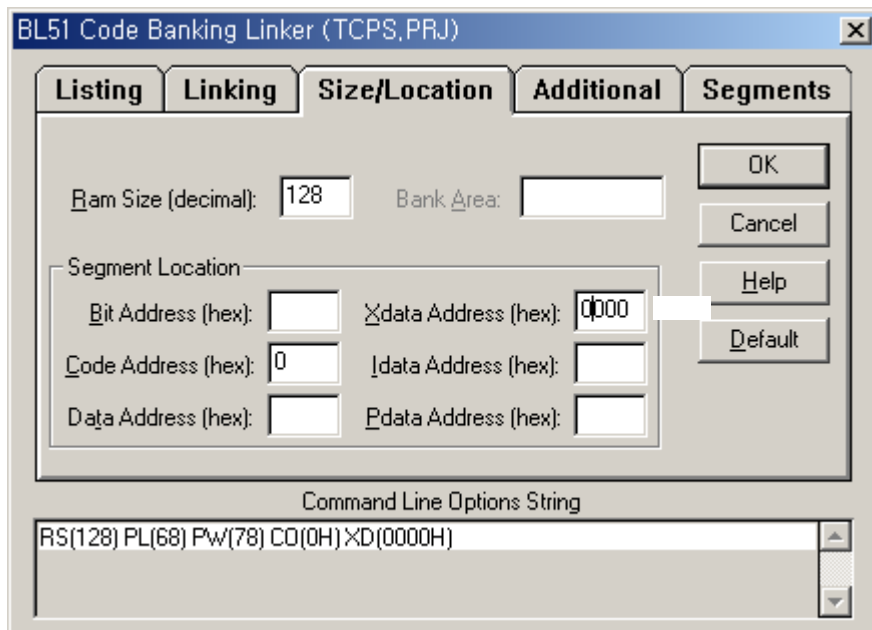
## 3.3 Application Development Procedure

### 3.3.1 Program Developing Procedure (based on the KEIL compiler)

\* Note: For this developing procedure, you need KEIL compiler and FLIP by ATMEL.

#### 3.3.1.1 Configuration

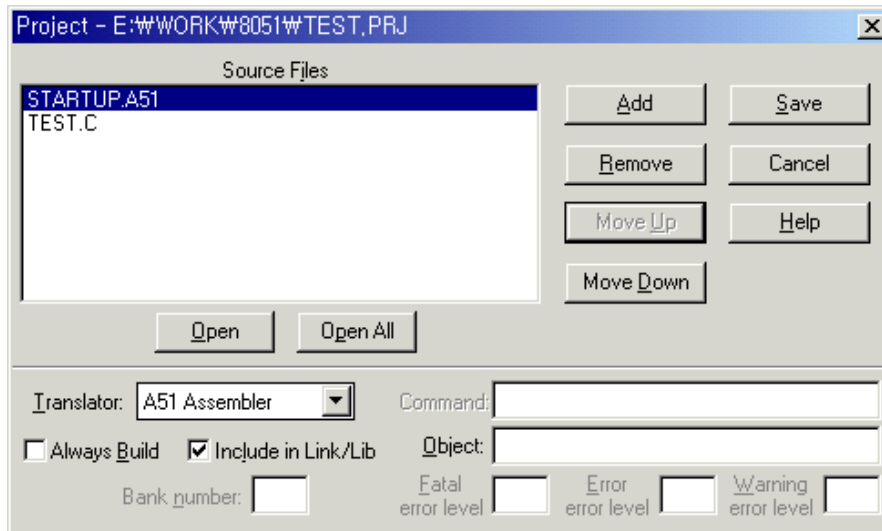
1. Run uVision-51.
2. In Options->BL51 Code Banking Linker, set the Xdata Address arbitrarily in the SRAM area and the Code Address to 0. (Refer to the section 3.3.3 Memory Map.)



<Fig. 28: uVision-51>

#### Making a New Project

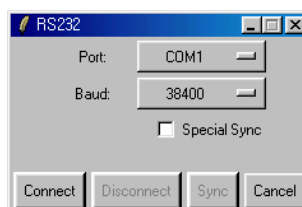
1. Make a new project and add startup.a51 and necessary sources.  
(Startup.a51 file does not need to be modified.)
2. Modify or develop the program.
3. Compile it



<Fig. 29: Making a new project>

### 3.3.2 Program Downloading and Running Procedure (based on Flip by ATMEL)

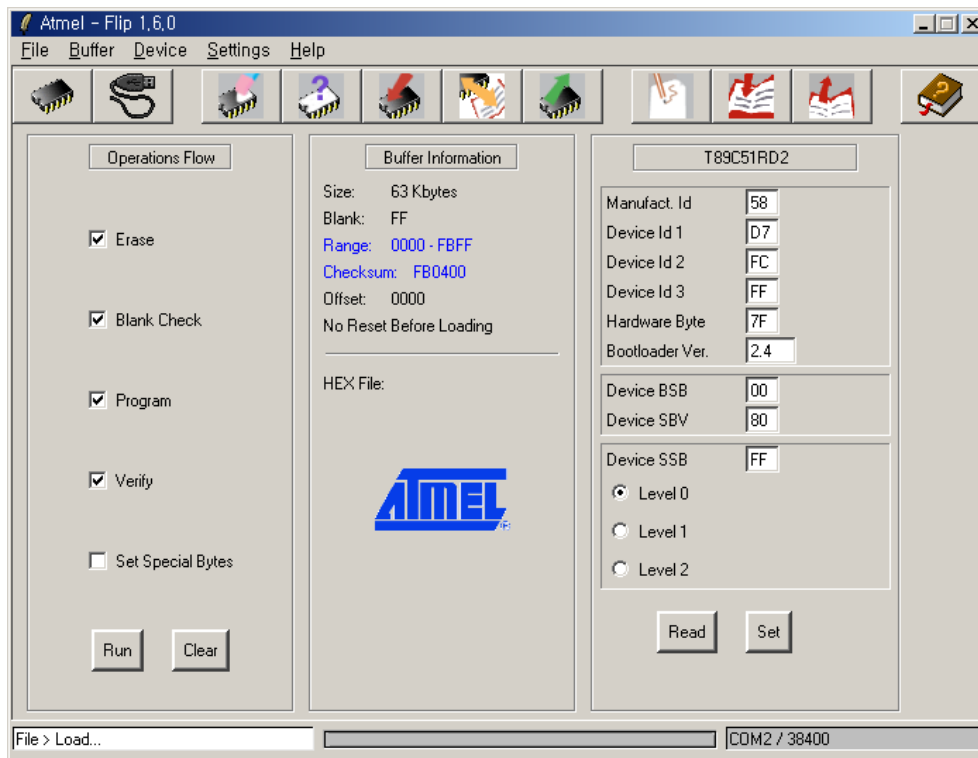
1. Connect the 8051EVB and the COM port of the PC with the serial cable.
2. Slide the switch on the JP3 to the left and turn on the power.
3. Run Flip, the ISP program of ATMEL, and select T89C51RD2 as the device.  
(Device>Select... => T89C51RD2 )
4. Select Setting>Communication>RS232 and click the Connect button.



<Fig. 30: Setting RS232>

5. Execute File > Load HEX... to load the file to be downloaded.
6. Press the Run button to reprogram the internal flash memory of the 8051 in accordance with the Operation Flow.

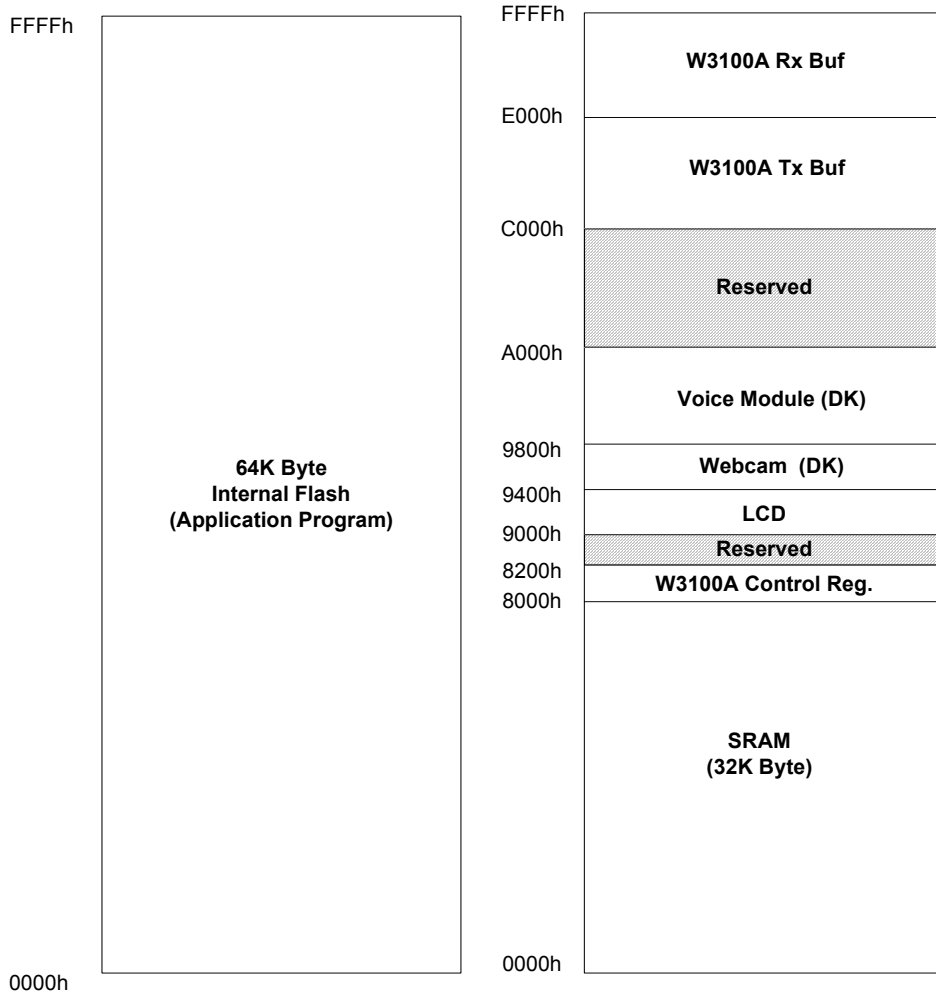
- 
7. Turn off the power, slide the switch on JP3 to the right, and turn on the power back to run the program that was downloaded in step 6.



**<Fig. 31: FLIP by ATMEL>**

\* Note: Recent version of FLIP is highly recommended.

### 3.3.3 Memory Map



<Fig. 32: Memory of EVB8051>

---

## 4. Hardware Designer's Guide

### 4.1 EVB8051 Schematic

Please refer to schematics in Software CD (\Schematic\).

### 4.2 PAL

Please refer to detailed information in Software CD (\Schematic\PAL\).

```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity test is
```

```
    port(
        Addr          : in std_logic_vector(5 downto 0);
        nPSEN         : in std_logic;
        nRD           : in std_logic;
        nWR           : in std_logic;
        nEA           : in std_logic;
        nROMCS        : out std_logic;
        nRAMCS        : out std_logic;
        nCS_I2CHIP    : out std_logic;
        LCDCS         : out std_logic;
        nCS_VM        : out std_logic;
        nCS_CAM       : out std_logic;
        nROMRD        : out std_logic;
        nRAMRD        : out std_logic
    );
```

```
ATTRIBUTE pin_numbers of test:ENTITY IS
"Addr(5):6 "
& "Addr(4):5 "
& "Addr(3):4 "
& "Addr(2):3 "
& "Addr(1):2 "
& "Addr(0):1 "
& "nPSEN:9 "
& "nRD:7 "
& "nWR:8 "
& "nEA:11 "
```

```

& "nROMCS:12 "
& "nRAMCS:13 "
& "nCS_I2CHIP:15 "
& "LCDCS:14 "
& "nCS_VM:16 "
& "nCS_CAM:19 "
& "nROMRD:17 "
& "nRAMRD:18 ";
end test;

```

architecture arch\_test of test is

```

begin
  nROMRD <= nPSEN;
  nRAMRD <= nRD;

  -- nROMCS (0x0000 - 0x7fff) : External ROM
  process(Addr, nPSEN)
  begin
    if (((Addr >= "000000") and (Addr < "100000")) and (nPSEN = '0')) then
      nROMCS <= '0';
    else
      nROMCS <= '1';
    end if;
  end process;

  --nRAMCS (0x0000 - 0x7fff) :
  process(Addr, nPSEN)
  begin
    if (((Addr >= "000000") and (Addr < "100000")) and (nPSEN = '1')) then
      nRAMCS <= '0';
    else
      nRAMCS <= '1';
    end if;
  end process;

  --LCDCS (0x9000 - 0x93ff)
  process(Addr, nRD, nWR)
  begin
    if (((Addr >= "100100") and (Addr < "100101")) and (nRD = '0' or nWR = '0')) then
      LCDCS <= '1';
    else
      LCDCS <= '0';
    end if;
  end process;

```

---

```

end process;

-- CAM (0x9400 - 0x9800)
process(Addr)
begin
  if((Addr >= "100101") and (Addr < "100110")) then
    nCS_CAM <= '0';
  else
    nCS_CAM <= '1';
  end if;
end process;

-- VM (0x9800 - 0xA000)
process(Addr)
begin
  if((Addr >= "100110") and (Addr < "101000")) then
    nCS_VM <= '0';
  else
    nCS_VM <= '1';
  end if;
end process;

-- W3100A (0x8000 - 0x9000, 0xC000 - 0x10000)
process(Addr)
begin
  if(((Addr >= "100000") and (Addr < "100100")) or (Addr >= "110000")) then
    nCS_I2CHIP <= '0';
  else
    nCS_I2CHIP <= '1';
  end if;
end process;

end arch_test;

-- W3100A (0x8000 - 0x9000, 0xC000 - 0x10000)
process(Addr, nRD, nWR)
begin
  if(((Addr >= "100000") and (Addr < "100100")) or (Addr >= "110000")) and (nRD = '0' or
nWR = '0')then
    nCS_I2CHIP <= '0';
  else
    nCS_I2CHIP <= '1';
  end if;
end process;

```

---

---

```
    end if;  
  end process;  
end arch_test;
```

### **4.3 Parts List**

Please refer to part list in Software CD (\Schematic\DirectMode\Partlist\).

---

# Appendix A. Quick Testing Procedure

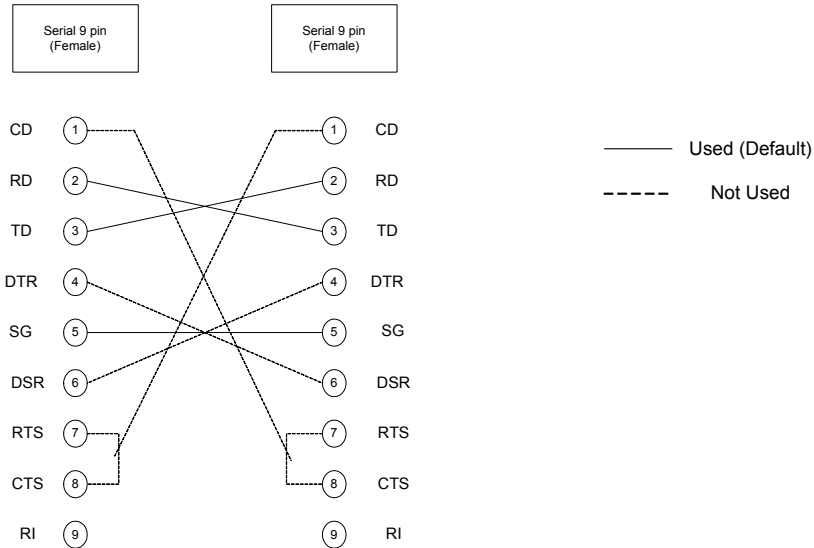
## A.1 Loopback Test

- Step 1. Check whether EVB8051 is connected to PC correctly with UTP cable.
- Step 2. Slide JP3 on EVB8051 to the right and turn on the power of EVB8051.
- Step 3. Ping to EVB8051 (192.168.0.2) on the PC whether EVB8051 is connected to the PC correctly.
- Step 4. Install Axinstall.exe (only for the first time) and run Ax1.exe on the PC.
- Step 5. Select TCP\Connect menu and enter EVB8051's IP address and Port number (5000).
- Step 6. Select File\Send menu and select a file to transfer.

## A.2 Web Server Test

- Step 1. Connect the EVB8051 and the COM port of the PC with the serial cable.
- Step 2. Slide the switch on the JP3 to the left and turn on the power.
- Step 3. Run Flip and select T89C51RD2 as the device in Device\Select ... menu.
- Step 4. Select Setting\Communication\RS232 menu and click the 'Connect' button.
- Step 5. Select File\Load HEX... to load httpd.hex to be downloaded.
- Step 6. Press the 'Run' button to reprogram the internal flash memory of the 8051 in accordance with the  
Operation Flow.
- Step 7. Turn off the power of EVB8051 and slide the switch on JP3 to the right, and turn on the power back  
to run the program that was downloaded in step 6.
- Step 8. Run web browser on the PC and enter URL, "http://192.168.0.2"

# Appendix B. Specification of Serial Cables



TD (Transmit Data) : Serial Data Output (TXD)  
 RD (Receive Data) : Serial Data Input (RXD)  
 CTS (Clear to Send) : This line indicates that the Modem is ready to exchange data.  
 DCD(Data Carrier Detect) : When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.  
 DSR (Data Set Ready) : This tells the UART that the modem is ready to establish a link.  
 DTR (Data Terminal Ready) : This is the opposite to DSR. This tells the Modem that the UART is ready to link.  
 RTS (Request To Send) : This line informs the Modem that the UART is ready to exchange data.  
 RI (Ring Indicator) : Goes active when modem detects a ringing signal from the PSTN.

---

# Appendix C. Specification of IIM7010A

IIM7010A is a module that consists of W3100A, Ethernet PHY and Mac Jack. It's used as module like a component, no effort is required to interface W3100A and PHY chip. It's the simplest and easiest solution to provide Internet connectivity.

## C.1 Advantages

- Easy design-win by reusing drop-in network module
- Users don't need to know details of network circuits
- Auto-detects 10/100 Mbps Ethernet speed



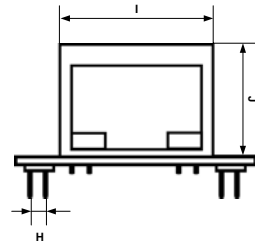
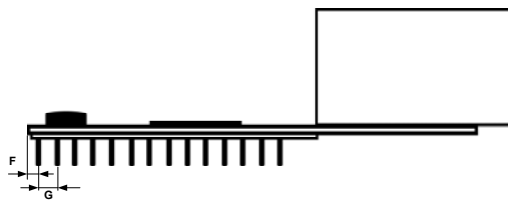
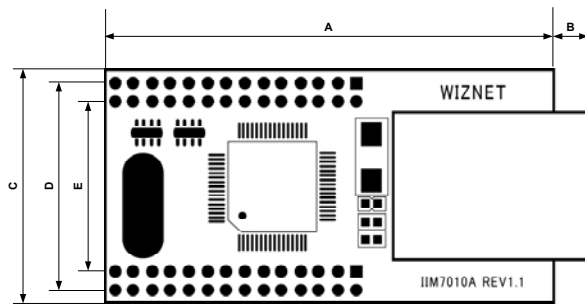
## C.2 Components

- TCP/IP:W3100A
- Ethernet physical layer:RTL8201BL
- Connector:RJ45(with transformer)

## C.3 Block Diagram



## C.4 Module dimension



| Symbol |      |
|--------|------|
| A      | 48.0 |
| B      | 4.0  |
| C      | 25.0 |
| D      | 20.4 |
| E      | 16.4 |
| F      | 1.0  |
| G      | 2.0  |
| H      | 2.0  |
| I      | 16.0 |
| J      | 13.4 |

## C.5 Pin description

I : Input

O : Output

I/O : Bi-directional Input and output

P : Power

### Power & Ground

| Symbol | Type | Pin No.  | Description                                    |
|--------|------|--|--|
| VCC    | P    | JP1 : 1 ,<br>JP2 : 24  | <b>Power</b> : 3.3 V power supply for IIM7000A |
| GND    | P    | JP1 : 8,<br>JP1 : 13,<br>JP1 : 24,<br>JP2 : 1<br>JP2 : 7,<br>JP2 : 13<br>JP2 : 14,<br>JP2 : 23 | <b>Ground</b>                                  |

### MCU Interfaces

| Symbol | Type | Pin No.   | Description  |
|--------|------|---|--|
| A14~A8 | I    | JP1 : 7,<br>JP1 : 10<br>JP1 : 9,<br>JP1 : 12<br>JP1 : 11,<br>JP1 : 14 JP1 :<br>15 | <b>Address / Device Address</b> : Used as Address[14-8] pin when set in Bus access mode.<br>Used as Device Address[6-0] pin when set in I <sup>2</sup> C interface mode. |

|       |     |  |   |
|-------|-----|--|---|
| A7~A0 | I   | JP1 : 16 ~<br>JP1 : 23   | <b>Address : Used as Address[7-0] pin when set in Bus access mode.</b><br>Not used when set in I <sup>2</sup> C interface mode. Leave them NC or ground them when they are not used.  |
| D7~D0 | I/O | JP2 : 21,<br>JP2 : 22<br>JP2 : 19,<br>JP2 : 20<br>JP2 : 17,<br>JP2 : 18<br>JP2 : 15,<br>JP2 : 16 | <b>Data</b> : Used as Data[7-0] pin.  |
| /CS   | I   | JP1 : 5  | <b>Module Select</b> : Active low. Drives /CS of the W3100A   |
| /RD   | I   | JP1 : 4  | <b>Read Enable</b> : Active low. Drives /RD of the W3100A   |
| /WR   | I   | JP1 : 3  | <b>Write Enable</b> : Active low. Drives /WR of the W3100A  |
| /INT  | O   | JP1 : 2  | <b>Interrupt</b> : Active low<br>Indicates that the W3100A requires MCU attention after reception or transmission. The interrupt is cleared by writing to the ISR of W3100A(Interrupt Status Register). All interrupts are maskable by writing IMG of W3100A(Interrupt Mask Register). This signal is active low. |
| I_SCL | I   | JP2 : 25   | <b>SCL</b> : clock used by I <sup>2</sup> C interface   |

|       |     |          |   |
|-------|-----|----------|---|
|       |     |          | mode.<br>This pin is positioned as pull-down internally.  |
| L_SDA | I/O | JP2 : 26 | <b>SDA</b> : data used by I <sup>2</sup> C interface mode.<br>This pin is positioned as pull-down internally. |

Network status Indicator LEDs

| Symbol   | Type | Pin No.  | Description   |
|----------|------|----------|---|
| L_COL    | O    | JP2 : 6  | <b>Collision LED</b> : Active low when collisions occur.  |
| L_100ACT | O    | JP2 : 8  | <b>Link 100/ACT LED</b> : Active low when linked 100 Base TX, and blinking when transmitting or receiving data. |
| L_10ACT  | O    | JP2 : 10 | <b>Link 10/ACT LED</b> : Active low when linked 10 Base T, and blinking when transmitting or receiving data.    |
| L_DUPX   | O    | JP2 : 11 | <b>Full Duplex LED</b> : Active low when in full duplex operation. Active high when in half duplex operation.   |
| L_LINK   | O    | JP2 : 12 | <b>Link LED</b> : Active low when linked  |

Miscellaneous Signals (Reset, Mode setting etc.)

| Symbol | Type | Pin No. | Description  |
|--------|------|---------|--|
| RESET  | I    | JP1 : 6 | <b>Reset</b> : Active high<br>Initializes or Reinitializes the W3100A. Asserting this pin will |

|         |    |                            | force a reset process to occur which will result in all internal registers reinitializing to their default and all strapping options are reinitialized. For complete reset function, this pin must be asserted low for at least 10us. Refer to W3100A datasheet for further detail regarding reset.   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
|---------|----|----------------------------|---|----|----|------|---|---|---------|---|---|------------------|---|---|-------------|---|---|----------------------------|
| /RESET  | I  | JP2 : 2                    | <b>Reset</b> : Active low<br>Reset RTL8201BL chip. For complete reset function, this pin must be asserted low for at least 10ms.  |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
| MODE1~0 | I  | JP1 : 25 ,<br>JP1 : 26     | <b>Mode Select</b> : This pin selects MCU interface and operating mode. Since each pin is positioned as pull-down internally, clocked mode – the default mode – is selected when the pins is not connected.<br><table border="1" data-bbox="863 1122 1240 1368"> <thead> <tr> <th>M1</th> <th>M0</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Clocked</td> </tr> <tr> <td>0</td> <td>1</td> <td>External clocked</td> </tr> <tr> <td>1</td> <td>0</td> <td>Non-clocked</td> </tr> <tr> <td>1</td> <td>1</td> <td>I<sup>2</sup>C interface</td> </tr> </tbody> </table> Refer to the W3100A datasheet for more information of setting modes. | M1 | M0 | Mode | 0 | 0 | Clocked | 0 | 1 | External clocked | 1 | 0 | Non-clocked | 1 | 1 | I <sup>2</sup> C interface |
| M1      | M0 | Mode                       |   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
| 0       | 0  | Clocked                    |   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
| 0       | 1  | External clocked           |   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
| 1       | 0  | Non-clocked                |   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
| 1       | 1  | I <sup>2</sup> C interface |   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |
| EXT_CLK | I  | JP1 : 28                   | <b>External clock</b> : supplementary clock used for external clocked mode.<br>In external clocked mode, W3100A   |    |    |      |   |   |         |   |   |                  |   |   |             |   |   |                            |

|    |   |  |  |
|----|---|--|--|
|    |   |  | use this clock to interface with MCU. Refer to the W3100A datasheet for more information.              |
| NC | - | <b>JP1 : 27,</b><br>JP2 : 3<br>JP2 : 5,<br>JP2 : 9<br>JP2 : 27 ,<br>JP2 : 28 | <b>Not Connect</b><br><b>JP1 : 27Pin</b> is used for factory test. This pin must not be used by users. |

---

# Appendix D. Specification of IIM7000A

IIM7000A is a module that consists of W3100A and Ethernet PHY.  
It's used as a module like a component, no effort is required to interface W3100A and PHY chip.  
So users can design quickly and easily and save the cost and time-to-market.

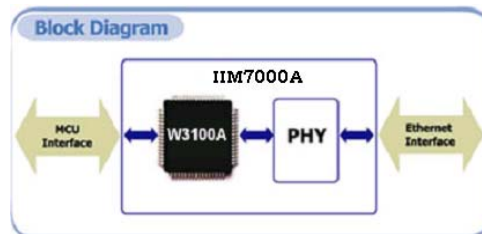
## C.1 Advantages

- Easy design by reusing drop-in network module
- No need to know details of network circuits
- Auto-detecting 10/100 Mbps Ethernet interface

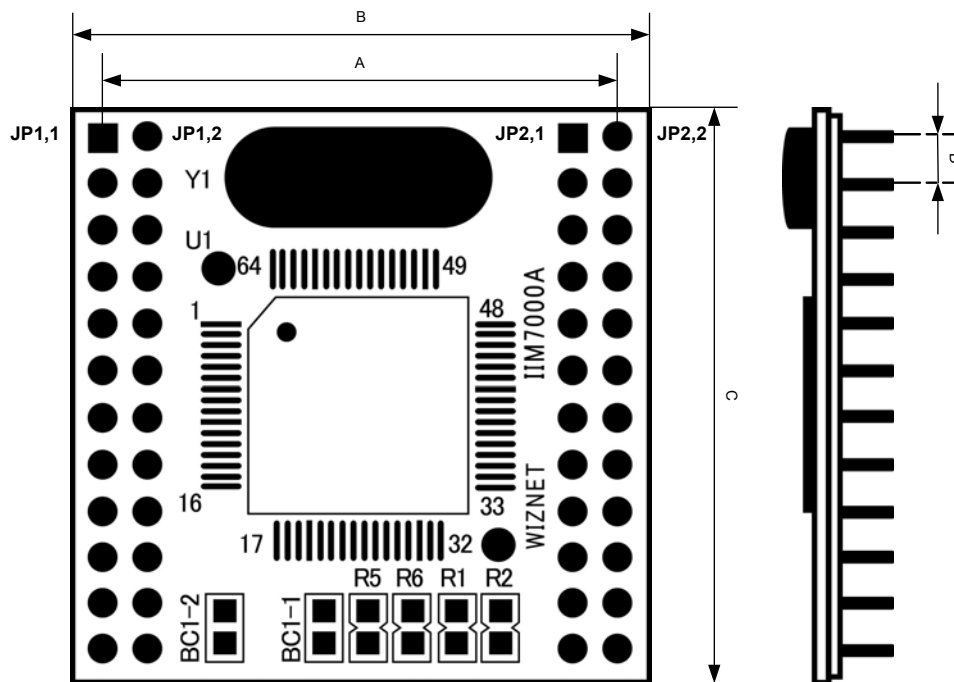
## C.2 Components

- TCP/IP:W3100A
- Ethernet physical layer:RTL8201BL

## C.3 Block Diagram



## C.4 Module dimension



| Symbol | Dimension in mm |
|--------|-----------------|
| A      | 22.4            |
| B      | 25.0            |
| C      | 25.0            |
| D      | 2.0             |

## C.5 Pin description

I : Input

O : Output

I/O : Bi-directional Input and output

P : Power

### Power & Ground

| Symbol | Type | Pin No.  | Description                                    |
|--------|------|--|--|
| VCC    | P    | JP1 : 1 ,<br>JP2 : 24  | <b>Power</b> : 3.3 V power supply for IIM7000A |
| GND    | P    | JP1 : 8,<br>JP1 : 13,<br>JP1 : 24,<br>JP2 : 1<br>JP2 : 7,<br>JP2 : 13<br>JP2 : 14,<br>JP2 : 23 | <b>Ground</b>                                  |

### MCU Interfaces

| Symbol | Type | Pin No.  | Description                              |
|--------|------|--|--|
| A14~A0 | I    | JP1 : 7,<br>JP1 : 10<br>JP1 : 9,<br>JP1 : 12<br>JP1 : 11<br>JP1 : 14 ~<br>JP1 : 23 | <b>Address : 15 bit-wide address bus</b> |

|       |     |  |  |
|-------|-----|--|--|
| D7~D0 | I/O | JP2 : 21,<br>JP2 : 22<br>JP2 : 19,<br>JP2 : 20<br>JP2 : 17,<br>JP2 : 18<br>JP2 : 15,<br>JP2 : 16 | <b>Data</b> : 8 bit-wide data bus  |
| /CS   | I   | JP1 : 5  | <b>Module Select</b> : Active low.<br>/CS of W3100A  |
| /RD   | I   | JP1 : 4  | <b>Read Enable</b> : Active low.<br>/RD of W3100A  |
| /WR   | I   | JP1 : 3  | <b>Write Enable</b> : Active low<br>/WR of W3100A  |
| /INT  | O   | JP1 : 2  | <b>Interrupt</b> : Active low<br>Indicates that the W3100A requires MCU attention after reception or transmission. The interrupt is cleared after writing values to the ISR of W3100A (Interrupt Status Register). All interrupts can be masked by writing values to the IMR of W3100A(Interrupt Mask Register).For more details refer to the W3100A Datasheet |

#### Network Interfaces & Network status Indicator LEDs

| Symbol | Type | Pin No. | Description                                       |
|--------|------|---------|---|
| TPTX+  | O    | JP2 : 3 | <b>Transmit Output</b> : Differential pair shared |

|          |   |             |   |
|----------|---|-------------|---|
| TPTX-    |   | JP2 : 5     | by 100 Base TX and 10 Base Modes.<br>When configured as 100 Base TX,  |
| TPRX+    | I | JP2 : 9     | <b>Receive Input</b> : Differential pair shared by<br>100 Base TX and 10 Base T Modes.                                |
| TPRX-    |   | JP2 :<br>11 |   |
| L_COL    | O | JP2 : 6     | <b>Collision LED</b> : Active low when collisions<br>occur.   |
| L_100ACT | O | JP2 : 8     | <b>Link 100/ACT LED</b> : Active low when<br>linked 100 Base TX, and blinking when<br>transmitting or receiving data. |
| L_10ACT  | O | JP2 :<br>10 | <b>Link 10/ACT LED</b> : Active low when<br>linked 10 Base T, and blinking when<br>transmitting or receiving data.    |
| L_LINK   | O | JP2 :<br>12 | <b>Link LED</b> : Active low when linked  |

#### Reset

| Symbol | Type | Pin No. | Description   |
|--------|------|---------|---|
| RESET  | I    | JP1 : 6 | <b>Reset</b> : Active high<br>Initializes or Reinitializes the W3100A.<br>Asserting this pin will force a reset process<br>to occur, which will result in all internal<br>registers reinitializing to their default and<br>all strapping options are reinitialized. For<br>complete reset function, this pin must be<br>asserted low for at least 10us. Refer to<br>W3100A datasheet for further detail<br>regarding reset. |
| /RESET | I    | JP2 : 2 | <b>Reset</b> : Active low   |

---

|  |  |  |   |
|--|--|--|---|
|  |  |  | Reset RTL8201BL chip. For complete reset function, this pin must be asserted low for at least 10ms. |
|--|--|--|---|