How to use UART in W7100A

Version 1.2

© 2012 WIZnet Co.,Ltd. All Rights Reserved.

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

Table of Contents

Ver. 1.1

1   Introduction

This document explains basic example codes of UART usage methods in W7100. All example codes are written based on C language and Keil compiler. Please refer to section '6. UART' of W7100 datasheet for more details on UART, Register, Interrupt, and etc...

The registers for setting the Baud Rate of UART are illustrated in Table 1 below. For Timer1, registers are SMOD and TH1, and for Timer 2, the registers are RLDH and RLDL.

Table1. Examples of Baud Rate Setting

| Baud Rate(bps) | Timer 1 / mode 2 | | Timer 2 |
| | TH1(0x8D) | | RLDH(0xCB), RLDL(0xCA) |
	SMOD = '0'	SMOD = '1'	
2400	160(0xA0)	64(0x40)	64384(0XFB80)
4800	208(0xD0)	160(0xA0)	64960(0xFDC0)
9600	232(0xE8)	208(0xD0)	65248(0xFEE0)
14400	240(0xF0)	224(0xE0)	65344(0XFF40)
19200	244(0xF4)	232(0xE8)	65392(0XFF70)
28800	248(0xF8)	240(0xF0)	65440(0xFFA0)
38400	250(0xFA)	244(0xF4)	65464(0XFFB8)
57600	252(0xFC)	248(0xF8)	65488(0xFFD0)
115200	254(0xFE)	252(0xFC)	65512(0xFFE8)
230400	255(0xFF)	254(0xFE)	65524(0xFFF4)

There are some examples of UART that have a fixed Baud Rate (mode0 and mode2). In this case, please refer to W7100 datasheet '6.UART' for Baud Rate calculations. All example codes are Echo-back examples that send messages received from Serial communication.

W7100 has four modes for UART, from UART mode0 to UART mode3. The example codes for each mode are as shown below.

2   mode0, 8-Bit UART, Fixed Baud Rate

```
void InitSerial(void)
{
        SCON = 0x10;                    // SERIAL mode 0, SM00 = 0, SM01 =0, REN=1
}


void PutByte(unsigned char byData)
{
        SBUF = byData;        // Write data into the serial-buffer
        while(!TI);           // Wait till data recording is finished
        TI = 0;               // Clear the transmit interrupt
}


unsigned char GetByte(void)
{
        unsigned char byData;        // Wait till data is received
        while(!RI);
        RI = 0;                      //Clear the RI
        byData = SBUF;               // Read data
        return byData;
}


void main()
{
    InitSerial();                        //Call the InitSerial() function
    while(1)   PutByte(GetByte());        //Echo-back the received data
}
```

The Baud Rate of UART mode0 is set to $f_{osc}/12$ of Internal Clock. Note that the W7100A has very fast internal clock (88.4736 MHz) so the baud rate clock will be 7.3MHz.

Due to the fixed Baud Rate and too fast Baud Rate clock, the mode0 is not actually used. Since the mode0 uses synchronous communication, there are no start bit and stop bit.

Among the example codes, set the SCON register to 0x10 in Init_iMCU()_function. The PutByte()_function write the Serial input to the Serial buffer and wait until the transmission is completed, and then clear the TI. The GetByte()_function returns the received message from the Serial and wait until the receiving is completed, and then clear the RI. The main()_function outputs the received message using all the Init_iMCU()_function, PutByte()_function, and GetByte()_function.

---

3   mode1, 8-Bit UART, Variable Baud Rate

Since mode1 uses asynchronous communication, the start bit and stop bit exist both in front and back of the data bits. The Baud Rate is created by using Timer1 and Timer2 overflow. The example codes of them are as followed.

3.1 Timer1 Clock Source

```
#define UART_MODE1_TIMER1

void InitSerial(void)
{
#ifdef UART_MODE1_TIMER1
        SCON = 0x50;                // SERIAL mode 1, SM00 = 0, SM01 =1, REN=1
        TMOD |= 0x20;           // Timer1 mode 2
        PCON |= 0x80;           // SMOD0 = 1
        TL1 = 0xFC;             // Baud Rate Setting to 115200bps, for more information
        TH1 = 0xFC;             // please refer to the W7100 Datasheet
        TR1 = 1;                // Timer1 START
#endif
}

void PutByte(unsigned char byData)
{
        SBUF = byData;      // Write data into the serial-buffer
        while(!TI);         // Wait till data recording is finished
        TI = 0;             // Clear the transmit interrupt
}

unsigned char GetByte(void)
{
        unsigned char byData;       // Wait till data is received
        while(!RI);
        RI = 0;                     //Clear the RI
        byData = SBUF;                      // Read data
        return byData;
}

void main()
```

```
{
        InitSerial();                          //Call the InitSerial() function
        while(1)   PutByte(GetByte());         //Echo-back the received data
}
```

The UART mode1 can use Timer1 or Timer2 to set the Baud Rate interchangeably. In this section, the Baud Rate is set using Timer1. Please refer to the W7100 datasheet for more details on setting the Baud Rate.

In the example code, set the SCON register to 0x50, and set Timer1 to mode2. In order to set the Baud Rate, the SMOD bit is set, and the TH1 register should be set to 0xFC. Then, the Baud Rate will be 115200bps. Other codes that output the message just the way it was received are the same as codes in section2.

3.2 Timer2 Clock Source

```
#define UART_MODE1_TIMER2


void InitSerial(void)

{
#ifdef UART_MODE1_TIMER2
        SCON = 0x50;                    // SERIAL mode 1, SM00 = 0, SM01 =1, REN=1
        T2CON = 0x30;                   // Timer2 Baud Rate Generator mode
        TH2 = 0xFF;                     // Baud Rate Setting to 115200bps, for more information about the
        TL2 = 0xE8;                     // Baud Rate please refer to the W7100 Datasheet
        RLDH = 0xFF;                    // Reload Baud Rate Setting to 115200bps
        RLDL = 0xE8;                    // Reload Baud Rate Setting to 115200bps
        TR2 = 1;                        // Timer2 START
#endif
}


void PutByte(unsigned char byData)

{
        SBUF = byData;      // Write data into the serial-buffer
        while(!TI);         // Wait till data recording is finished
        TI = 0;             // Clear the transmit interrupt
}


unsigned char GetByte(void)

{
```

Ver. 1.1

```
        unsigned char byData;        // Wait till data is received
        while(!RI);
        RI = 0;                      //Clear the RI
        byData = SBUF;               // Read data
        return byData;
}


void main()
{
        InitSerial();                        // Call the InitSerial() function
        while(1)   PutByte(GetByte());       //Echo-back the received data
}
```

The UART mode1 can use Timer1 or Timer2 to set the Baud Rate interchangeably. In this section, the Baud Rate is set using Timer2. Please refer to W7100 datasheet for more details on setting the Baud Rate.

In the example code, set the SCON register to 0x50, and set Timer2 to Baud Rate Generator mode. In order to set the Baud Rate, the value of TH2 and TL2 register should each be set to 0xFF and 0xE8. Then, the Baud Rate will be 115200bps. The Reload Values like RLDH and RLDL should also each be set to 0xFF and 0xE8. Other codes that output the message just the way it was received are the same as codes in section2.

Ver. 1.1

4   mode2, 9-Bit UART, Fixed Baud Rate

```
void InitSerial(void)
{
        SCON = 0x90;                    // SERIAL mode 2, SM00 = 1, SM01 = 0, REN=1
        PCON &= 0x7F;              // f_osc/64(SMOD = 0), f_osc/32(SMOD = 1)
}
void PutByte(unsigned char byData)
{
        SBUF = byData;        // Write data into the serial-buffer
        while(!TI);           // Wait till data recording is finished
        TI = 0;               // Clear the transmit interrupt
}


unsigned char GetByte(void)
{
        unsigned char byData;         // Wait till data is received
        while(!RI);
        RI = 0;                        //Clear the RI
        byData = SBUF;                        // Read data
        return byData;
}


void main()
{
    InitSerial();                      // Call the InitSerial() function
    while(1)   PutByte(GetByte());      //Echo-back the received data
}
```

The Baud Rated of UART mode2 are fixed by the $f_{osc}/32$ or $f_{osc}/64$ of internal clock. The $f_{osc}/32$ and $f_{osc}/64$ of internal clock are determined depending on the SMOD0 bit. Considering the internal clock of W7100, a very fast Baud Rate clock of 2.7 ~ 1.4MHz can be produced. Like the case with mode0, mode3 is not really used because of the fixed Baud Rate and too fast Baud Rate clock.

In the example codes, set the SCON register to 0x90. In order to set the Baud Rate, set the SMOD0, the highest bit of the PCON register (Baud Rate = $f_{osc}/32$). Other codes that output the message just the way it was received are the same as codes in section2.

Ver. 1.1

5   mode3, 9-Bit UART, Variable Baud Rate

5.1 Timer1 Clock Source

```c
#define UART_MODE3_TIMER1


void InitSerial(void)
{
#ifdef UART_MODE3_TIMER1
        SCON = 0xD0;                // SERIAL mode 3, SM00 = 1, SM01 =1, REN=1
        TMOD |= 0x20;           // Timer1 mode 2
        PCON |= 0x80;           // SMOD0 = 1
        TL1 = 0xFC;             // Baud Rate Setting to 115200bps, for more information about the
        TH1 = 0xFC;             //Baud Rate please refer to the W7100 Datasheet
        TR1 = 1;                // Timer1 START
#endif
}
void PutByte(unsigned char byData)
{
        SBUF = byData;      // Write data into the serial-buffer
        while(!TI);         // Wait till data recording is finished
        TI = 0;             // Clear the transmit interrupt
}


unsigned char GetByte(void)
{
        unsigned char byData;       // Wait till data is received
        while(!RI);
        RI = 0;                     //Clear the RI
        byData = SBUF;          // Read data
        return byData;
}


void main()
{
    InitSerial();                       //Call the InitSerial() function
    while(1)   PutByte(GetByte());       //Echo-back the received data
}
```

The UART mode3 can use Timer1 or Timer2 to set the Baud Rate interchangeably. In this section, the Baud Rate is set using Timer1. Please refer to W7100 datasheet for more details on setting the Baud Rate. The difference from mode1 is that a bit adds on in front of the stop bit. This added bit can be used for parity check or multiprocessor communication. Please refer to the W7100 datasheet for more details.

In the example code, set the SCON register to 0xD0, and set Timer1 to mode2. In order to set the Baud Rate, the SMOD_bit of PCON should be set, and the TH1 register should be set to 0xFC. Then, the Baud Rate will be 115200bps. Other codes that output the message just the way it was received are the same as codes in section2.

5.2 Timer2 Clock Source

```
#define UART_MODE3_TIMER2


void InitSerial(void)
{
#ifdef UART_MODE3_TIMER2
        SCON = 0xD0;            // SERIAL mode 3, SM00 = 1, SM01 =1, REN=1
        T2CON = 0x30;           // Timer2 Baud Rate Generator mode
        TH2 = 0xFF;             // Baud Rate Setting to 115200bps, for more information about the
        TL2 = 0xE8;             // Baud Rate please refer to the W7100 Datasheet
        RLDH = 0xFF;            // Reload Baud Rate Setting to 115200bps
        RLDL = 0xE8;            // Reload Baud Rate Setting to 115200bps
        TR2 = 1;                // Timer2 START
#endif
}


void PutByte(unsigned char byData)
{
        SBUF = byData;      // Write data into the serial-buffer
        while(!TI);         // Wait till data recording is finished
        TI = 0;             // Clear the transmit interrupt
}


unsigned char GetByte(void)
{
        unsigned char byData;      // Wait till data is received
        while(!RI);
        RI = 0;                    //Clear the RI
```

Ver. 1.1   10

```
        byData = SBUF;              // Read data
        return byData;
}


void main()
{
        InitSerial();                       //Call the InitSerial() function
        while(1)   PutByte(GetByte());       //Echo-back the received data
}
```

The UART mode3 can use Timer1 or Timer2 to set the Baud Rate interchangeably. In this section, the Baud Rate is set using Timer2. Please refer to W7100 datasheet for more details on setting the Baud Rate. The difference from mode1 is that a bit adds on in front of the stop bit. This added bit can be used for parity check or multiprocessor communication. Please refer to the W7100 datasheet for more details.

In the example code, set the SCON register to 0xD0, and set Timer2 to Baud Rate Generator mode. In order to set the Baud Rate, the value of TH2 and TL2 register should each be set to 0xFF and 0xE8. Then, the Baud Rate will be 115200bps. The Reload Values like RLDH and RLDL should also each be set to 0xFF and 0xE8. Other codes that output the message just the way it was received are the same as codes in section2.

6   Running example

This chapter explains how to download and run the UART example of the iMCU7100EVB application note. All example codes are written based on C language and Keil compiler. The user can use two programs to download the HEX file, which is created by compiling the Keil project on the iMCU7100EVB; either the WizISP program or the W7100 Debugger program. Please refer to 'iMCU7100EVB User's Guide,' 'WizISP Program Guide,' and 'W7100 Debugger Guide' for more details.
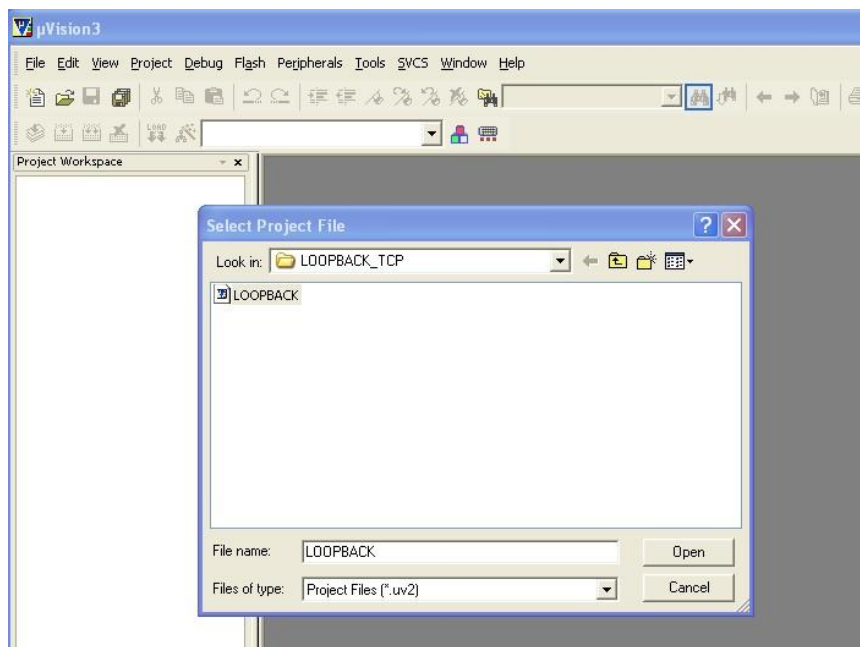
The processes for running the example code of the 'How to use UART in W7100' on the iMCU7100EVB board.

1.   Create the Keil project and write the UART example code

2.   Create the HEX file with compiler by the Keil compiler

3.   Download the created HEX file on the iMCU7100EVB board, using either the WizISP program or the W7100 Debugger

4.   Run the Serial terminal program, and set the port, bard rate, and etc.

5.   Run the UART example on the Board, and check the serial message from the terminal program.

The following sections show the results on each step.

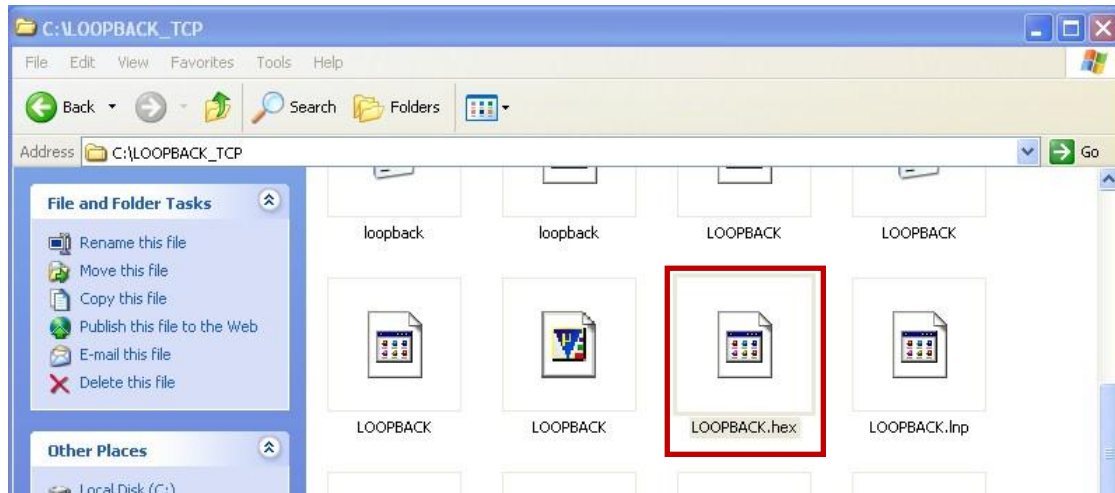## 6.1 Make a Keil project

The user can create the Keil project, or also can open the attached Keil project as shown in Fig 2.1.

<Fig.6.1> Open the Keil project of UART

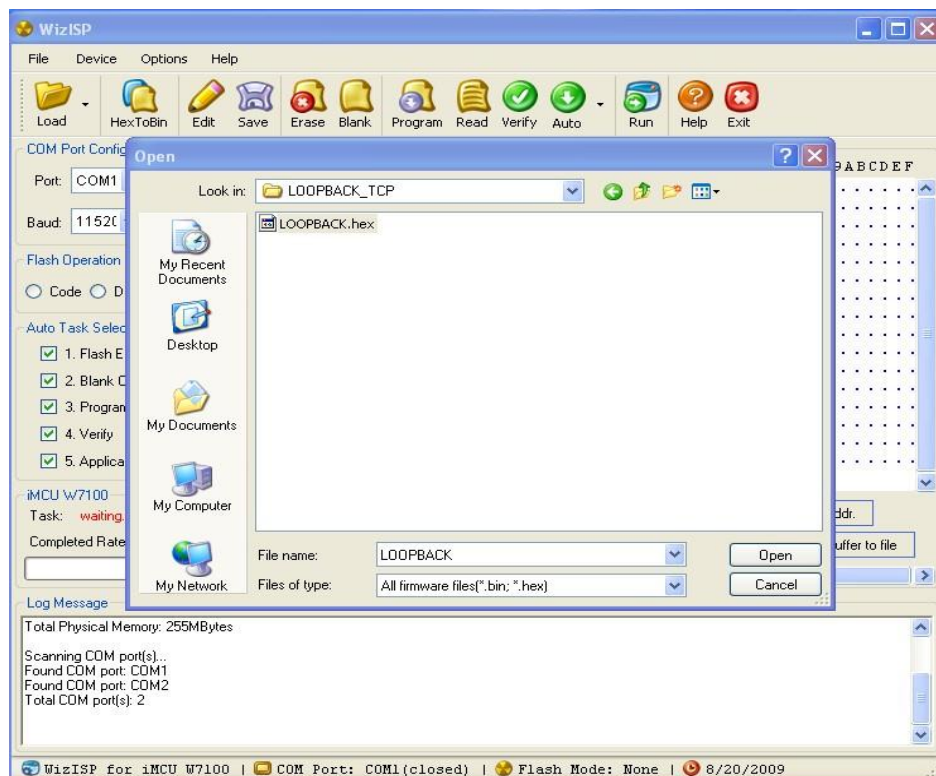## 6.2 Make a HEX file with compile

Write the example code, and compile, and then create the HEX file.

<Fig.6.2> Make HEX file using Keil compiler

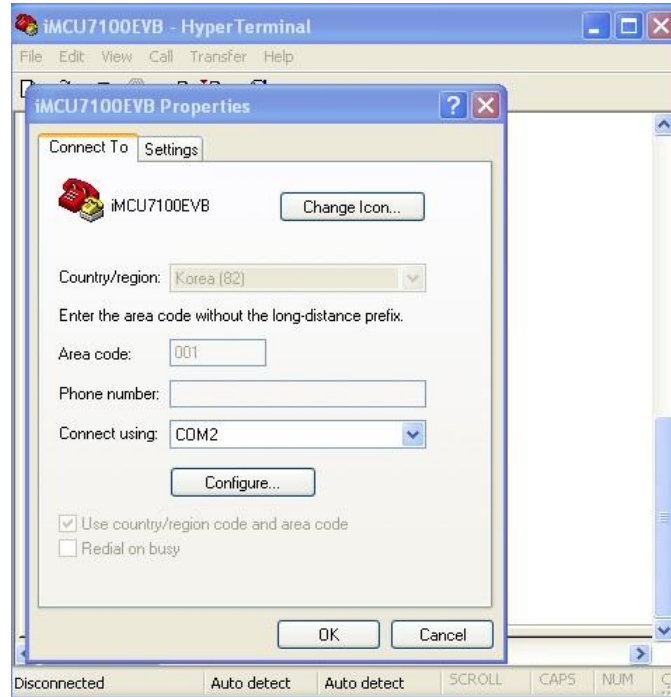## 6.3 Download the HEX file to iMCU7100EVB

Download the HEX file on the iMCU7100EVB board by using either the WizISP program or the W7100 Debugger. The figure below is using the WizISP program. Since WizISP program loads BIN file, the WizISP program has a function for changing the Hex file into BIN file.
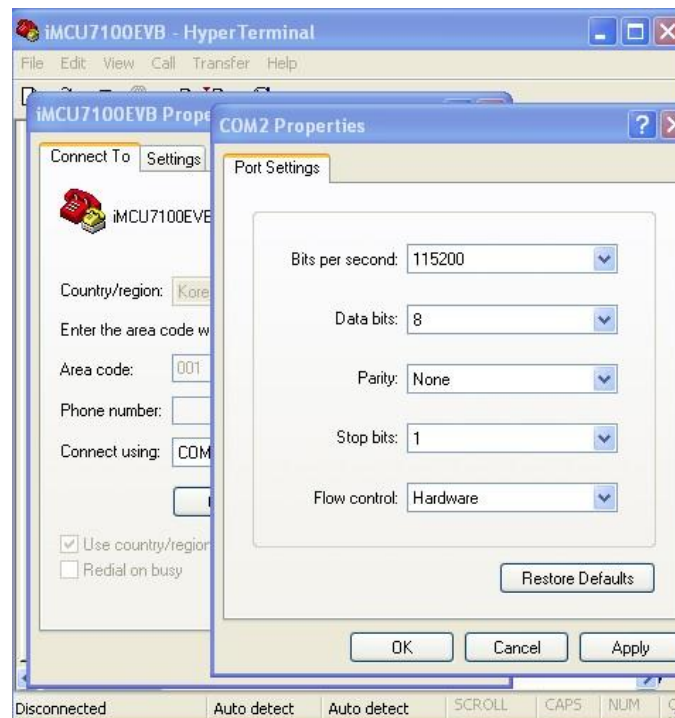

<Fig.6.3> Download the HEX file to the iMCU7100EVB

WIZnet

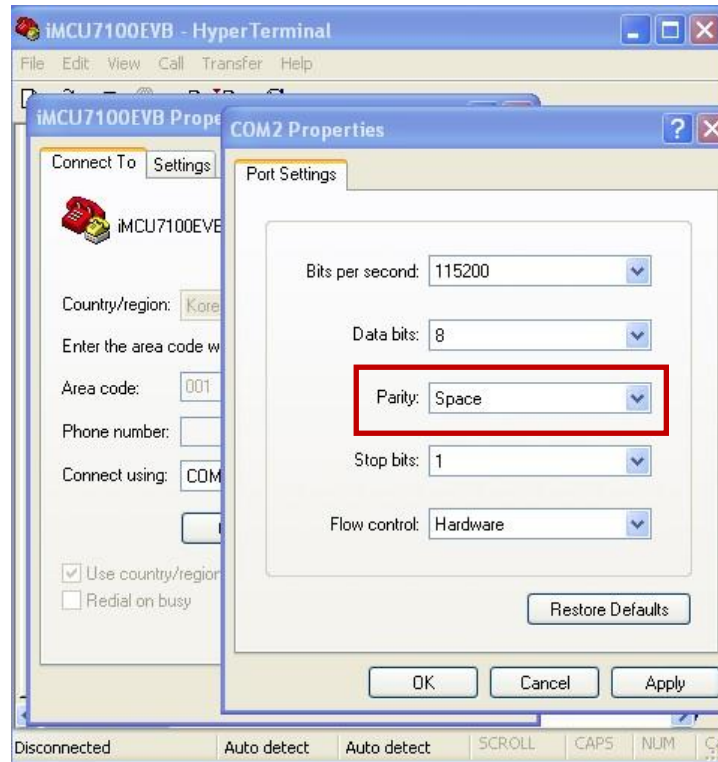## 6.4 Serial terminal program

The Serial terminal program is needed to check whether the UART example program runs well. This chapter uses the Hyperterminal program which is offered as the basic program by MS Windows. The user should set the serial port, Baud rate, and other serial settings according to their environment.

<Fig.6.4> Hyper terminal setting #1

<Fig.6.5> Hyper terminal setting #2 for mode0 or mode1 (8-Bit)

Ver. 1.1 14

<Fig.6.6> Hyper terminal setting #2 for mode2 or mode3 (9-Bit)

Set the Serial port, which is used for running the UART example code. Then set the Baud Rate according to the environment. Select 8 for data bit, none for parity, 1 for stop bit, and select the hardware control for the flow control. In this document, the Serial port is COM2, and the Baud Rate is 115200. Note that if user uses mode2 or mode3, the parity bit should be set to whatever (Not to none). Because the mode2 or mode3 has one more bit than mode0 or mode1. The added one bit is used for the parity check or the multiprocessor communication. Since the example code doesn't use the multiprocessor communication, user uses it to parity check.

6.5 Run the UART example code

After all settings are completed, run the Serial terminal program (Hyper terminal). And reset the iMCU7100EVB board, and then run the example code. The example code is echo back which shows input message with keyboard. If there is no problem with the example code and user insert the message 'Hello WIZnet!!', the message will appear as Fig 2.7 below.

<Fig.6.7> Result of the UART example code

Document History Information

Version	Date	Descriptions
Ver. 0.9βeta	Sep, 2009	Release with W7100 launching
Ver. 1.0	Mar, 2011	Modify for W7100A QFN 64pin package
Ver. 1.1	Jun, 2012	Fixed some awkward expressions of English documents. Changed the example source codes. Timer initialization part in Init_iMCU() function moved to InitSerial() function. Please refer to the 'Serial.c' and 'Serial.h' in W7100A drivers.

Copyright Notice