W7500 Errata Sheet

Document History

Ver 1.0.0 (July.11, 2016)	First release (erratum 1) – I2C

© 2016 WIZnet Co., Inc. All Rights Reserved.
For more information, visit our website at http://www.wiznet.co.kr

Erratum 1	
Phenomenon	**Receiving repeating data in continuative data transmission causes I2C communication problem.**
Condition	 W7500 receives the first repeating data but starts to discards from the 2nd repeating data to next different data in continuative data transmission. It causes data loss.
Solution & Recommendation	To avoid this issue, W7500 uses GPIO instead of I2C. In this case, SCL has limited speed, 100MHz. Example pseudo code:

```
Function Initialize_I2C ( )
{
…
    scl_port_num = I2C_PORT(conf->scl);
    scl_pin_index = I2C_PIN_INDEX(conf->scl);

    sda_port_num = I2C_PORT(conf->sda);
    sda_pin_index = I2C_PIN_INDEX(conf->sda);

    //SCL setting
    GPIO_InitDef.GPIO_Pin = scl_pin_index;
    GPIO_InitDef.GPIO_Mode = GPIO_Mode_OUT;

    if(scl_port_num == 0)
    {
        GPIO_Init(GPIOA, &GPIO_InitDef);
        GPIO_SetBits(GPIOA, scl_pin_index);
    }
….
//SDA setting
    GPIO_InitDef.GPIO_Pin = sda_pin_index;
```

```
        GPIO_InitDef.GPIO_Mode = GPIO_Mode_IN;
        if(sda_port_num == 0)
        {
            GPIO_Init(GPIOA, &GPIO_InitDef);
            GPIO_ResetBits(GPIOA, sda_pin_index);
        }
    …..
    }


    /* SCL function */
    Function I2C_SCL
    {
    …
        if(scl_port_num == 0)
        {
            if(data == 1)
                GPIO_SetBits(GPIOA, scl_pin_index);
            else
                GPIO_ResetBits(GPIOA, scl_pin_index);
        }
    …
    }


    /* SDA function */
    Function I2C_SDA
    {
    …
         if(sda_port_num == 0)
        {
            if(data == 1)
                GPIOA->OUTENCLR = sda_pin_index;
            else
                GPIOA->OUTENSET = sda_pin_index;
        }
    …
    }
```

```
/* START function */
Function I2C_START

void I2C_Start(I2C_ConfigStruct* conf)
{
    I2C_WriteBitSCL(conf, 1);
    I2C_WriteBitSDA(conf, 1);

    I2C_WriteBitSDA(conf, 0);
    I2C_WriteBitSCL(conf, 0);
}
/* STOP function */
Function I2C_STOP

void I2C_Stop(I2C_ConfigStruct* conf)
{
    I2C_WriteBitSCL(conf, 0);
    I2C_WriteBitSDA(conf, 0);

    I2C_WriteBitSCL(conf, 1);
    I2C_WriteBitSDA(conf, 1);
}
......
```