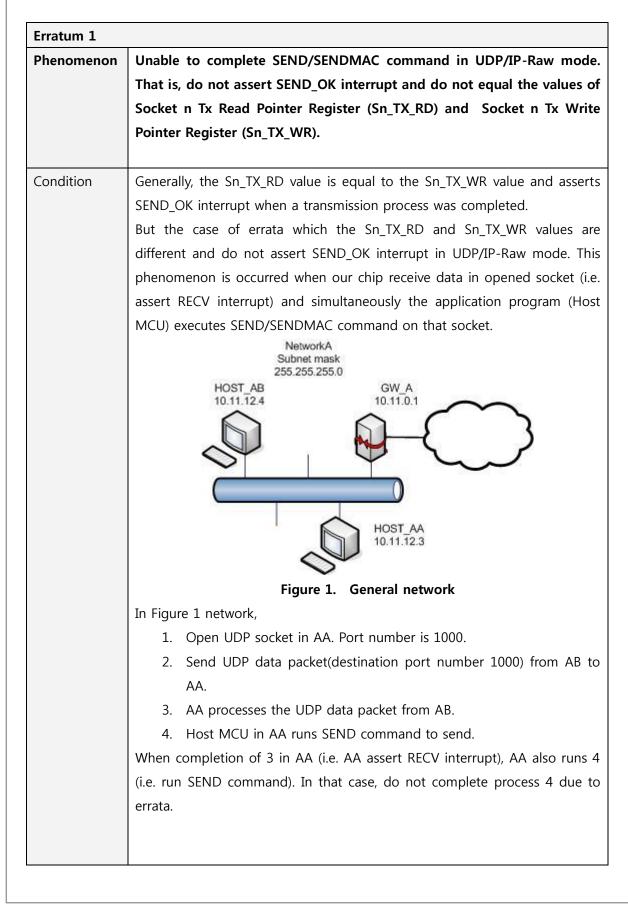# W3150A+/W5100 Errata Sheet

Document History

Ver 1.0.0 (OCT. 27, 2007)	First release (erratum 1)
Ver 2.0.0 (SEP. 10, 2008)	Add W5100 solution for erratum 1 Remove Recommendation for erratum 1
Ver 2.1 (APR. 5, 2010)	Remove the erratum solution for W5100 in v2.0 and bring back the recommendation of v1.0
Ver 2.2 (FEB. 17, 2012)	Add erratum 2, 3
Ver 2.3 (MAR. 5, 2012)	Add a solution for erratum 2, 3

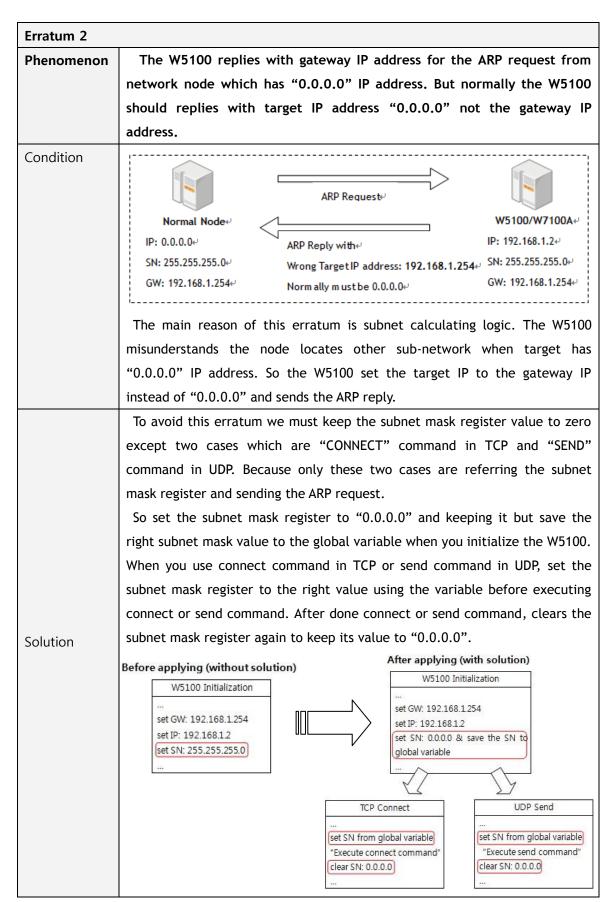| Erratum 1	
Phenomenon	**Unable to complete SEND/SENDMAC command in UDP/IP-Raw mode. That is, do not assert SEND_OK interrupt and do not equal the values of Socket n Tx Read Pointer Register (Sn_TX_RD) and Socket n Tx Write Pointer Register (Sn_TX_WR).**

Condition	Generally, the Sn_TX_RD value is equal to the Sn_TX_WR value and asserts SEND_OK interrupt when a transmission process was completed. But the case of errata which the Sn_TX_RD and Sn_TX_WR values are different and do not assert SEND_OK interrupt in UDP/IP-Raw mode. This phenomenon is occurred when our chip receive data in opened socket (i.e. assert RECV interrupt) and simultaneously the application program (Host MCU) executes SEND/SENDMAC command on that socket.
	
	Figure 1. General network
	
	In Figure 1 network,
	
	    1. Open UDP socket in AA. Port number is 1000.
	    2. Send UDP data packet(destination port number 1000) from AB to AA.
	    3. AA processes the UDP data packet from AB.
	    4. Host MCU in AA runs SEND command to send.
	
	When completion of 3 in AA (i.e. AA assert RECV interrupt), AA also runs 4 (i.e. run SEND command). In that case, do not complete process 4 due to errata.

Solution	**In case of W3150A+,**
	If you can monitor the RXDV signal (in MII interface), you can solve this problem.
	Before you execute SEND/SENDMAC command in UDP/IP-Raw mode, check the value of RXDV signal is '0'. That means there is no received data packet, so you can avoid the errata condition.
	Refer to following pseudo-code.

```
/* Change sendto() function */
Function sendto()
{
…
    While (RXDV == '1') ;
    SEND command;
/* Complete Sending */
}
```

Recommendat ion	**In case of W5100,**
	We don't have solution but we can give a recommendation.
	After complete a transmission process, check whether Sn_TX_RD and Sn_TX_WR have same values or not. If both values are still different, close the socket and reset by force.

```
/* Change sendto() function */
Function sendto()
{
…
/* Complete Sending */
    /* wait until Sn_TX_WR and Sn_TX_RD are same */
    While (Sn_TX_WR != Sn_TX_RD)
    {
        wait some time;
        loop_cnt++;
        if (loop_cnt > CONST_BLOCK_CNT) goto RESET
    }
}
```

Erratum 2	
Phenomenon	The W5100 replies with gateway IP address for the ARP request from network node which has "0.0.0.0" IP address. But normally the W5100 should replies with target IP address "0.0.0.0" not the gateway IP address.
Condition	The main reason of this erratum is subnet calculating logic. The W5100 misunderstands the node locates other sub-network when target has "0.0.0.0" IP address. So the W5100 set the target IP to the gateway IP instead of "0.0.0.0" and sends the ARP reply.
Solution	To avoid this erratum we must keep the subnet mask register value to zero except two cases which are "CONNECT" command in TCP and "SEND" command in UDP. Because only these two cases are referring the subnet mask register and sending the ARP request. So set the subnet mask register to "0.0.0.0" and keeping it but save the right subnet mask value to the global variable when you initialize the W5100. When you use connect command in TCP or send command in UDP, set the subnet mask register to the right value using the variable before executing connect or send command. After done connect or send command, clears the subnet mask register again to keep its value to "0.0.0.0".

Example pseudo code:

```
/* Global variable declaration for subnet mask value */
unsigned char subnet_val[4];
/* W5100 initialization function */
Function Initialize_W5100( )
{
...
/* Clear the subnet mask register */
    IINCHIP_WRITE(SUBR0, 0);
    IINCHIP_WRITE(SUBR1, 0);
    IINCHIP_WRITE(SUBR2, 0);
    IINCHIP_WRITE(SUBR3, 0);
/* Save the right subnet mask value if the subnet is 255.255.255.0 */
    subnet_val[0] = 255;
    subnet_val[1] = 255;
    subnet_val[2] = 255;
    subnet_val[3] = 0;
...
}


/* TCP connect function */
Function TCP_Connect( )
{
...
/* Set the subnet mask register to the right value using the variable */
    IINCHIP_WRITE(SUBR0, subnet_val[0]);
    IINCHIP_WRITE(SUBR1, subnet_val[1]);
    IINCHIP_WRITE(SUBR2, subnet_val[2]);
    IINCHIP_WRITE(SUBR3, subnet_val[3]);
/* Execute TCP connect command */
    IINCHIP_WRITE(Sn_CR(socket), Sn_CR_CONNECT);
/* Wait for command done */
    while(Sn_CR(socket));
/* Clear the subnet mask register again and keep it */
    IINCHIP_WRITE(SUBR0, 0);
    IINCHIP_WRITE(SUBR1, 0);
```
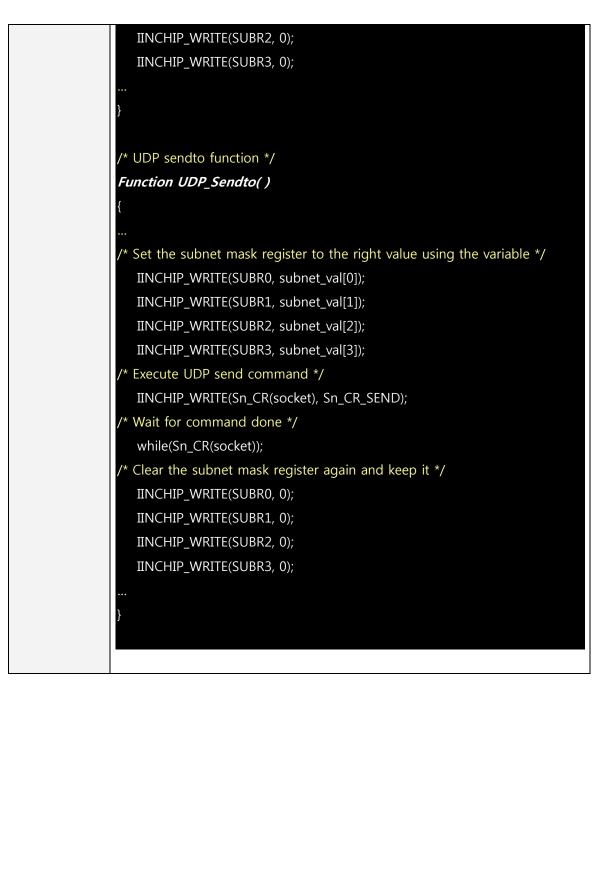
```
    IINCHIP_WRITE(SUBR2, 0);
    IINCHIP_WRITE(SUBR3, 0);
…
}


/* UDP sendto function */
Function UDP_Sendto( )
{
…
/* Set the subnet mask register to the right value using the variable */
    IINCHIP_WRITE(SUBR0, subnet_val[0]);
    IINCHIP_WRITE(SUBR1, subnet_val[1]);
    IINCHIP_WRITE(SUBR2, subnet_val[2]);
    IINCHIP_WRITE(SUBR3, subnet_val[3]);
/* Execute UDP send command */
    IINCHIP_WRITE(Sn_CR(socket), Sn_CR_SEND);
/* Wait for command done */
    while(Sn_CR(socket));
/* Clear the subnet mask register again and keep it */
    IINCHIP_WRITE(SUBR0, 0);
    IINCHIP_WRITE(SUBR1, 0);
    IINCHIP_WRITE(SUBR2, 0);
    IINCHIP_WRITE(SUBR3, 0);
…
}
```

Erratum 3	
Phenomenon	Assuming that the IP address of W5100 is "0.0.0.0" and the gateway, subnet mask is valid (not "0.0.0.0"), the W5100 set the target IP address of ARP request to the gateway IP address not the target node IP address when sends ARP request to another node. So the peer node cannot receive the ARP request from the W5100.
Condition	 The W5100 miss calculates the sub-network location when sends the ARP request if its own IP address is "0.0.0.0". In the same condition, even if the gateway IP address is "0.0.0.0", the W5100 sends ARP request to "0.0.0.0" IP address because the W5100 sends ARP request to the gateway.
Solution	The reason of this erratum3 is same as erratum2 so the solution is also same with erratum2. Please refer to the solution of erratum2.