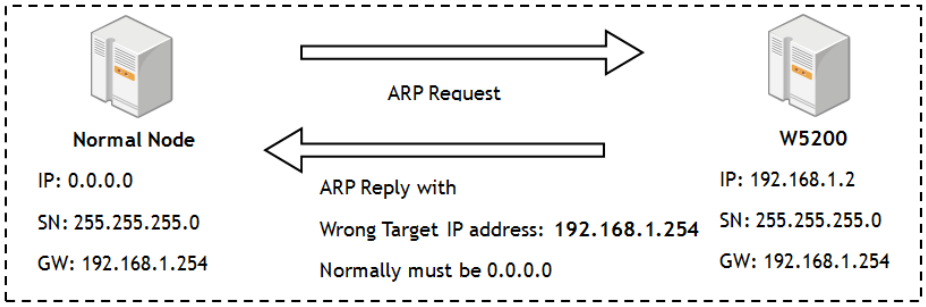
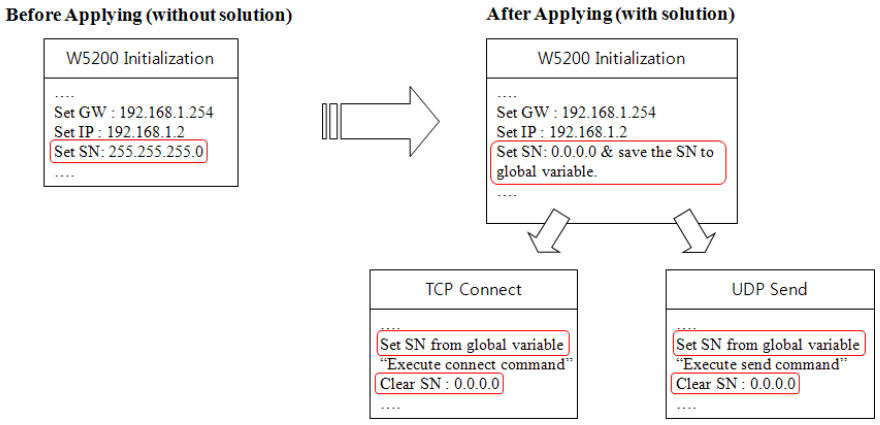
W5200 Errata Sheet

Document History

Ver 1.0.0 (Feb. 23, 2012)	First release for erratum 1
Ver 1.0.1 (Mar. 28, 2012)	Add a solution for erratum 1, 2
Ver 1.0.2 (Apr. 09, 2012)	Add a solution for erratum 3
Ver 1.0.3 (Feb. 31, 2013)	Modified phenomenon and condition for erratum 1
Ver 1.0.4 (Jun. 5, 2013)	Fixed typos in erratum 1 (W5300 -> W5200)

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

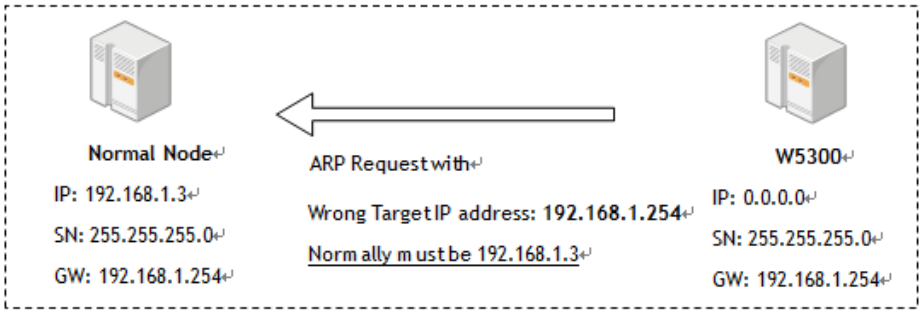
Erratum 1	
Phenomenon	**The W5200 replies with gateway IP address for the ARP request from normal node which has "0.0.0.0" IP address. But normally the W5200 should replies with target IP address "0.0.0.0" not the gateway IP address.**
Condition	 The main reason of this erratum is subnet calculating logic. The W5200 misunderstands the node locates other sub-network when target has "0.0.0.0" IP address. So the W5200 set the target IP to the gateway IP instead of "0.0.0.0" and sends the ARP reply.
Solution & Recommendation	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 W5200. 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:

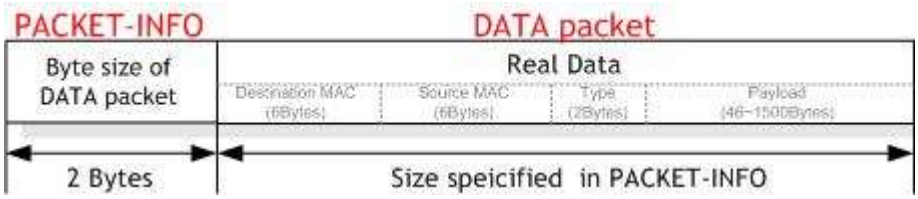
```
/* Global variable declaration for subnet mask value */
unsigned char subnet_val[4];
/* W5200 initialization function */
Function Initialize_W5200( )
{
...
/* Clear the subnet mask register */
    IINCHIP_WRITE(SUBR0, 0);
    IINCHIP_WRITE(SUBR1, 0);
    IINCHIP_WRITE(SUBR2, 0);
    IINCHIP_WRITE(SUBR3, 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 2	
Phenomenon	**Assuming that the IP address of W5200 is "0.0.0.0" and the gateway, subnet mask is valid (not "0.0.0.0"), the W5200 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 W5200.**
Condition	 The W5200 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 W5200 sends ARP request to "0.0.0.0" IP address because the W5200 sends ARP request to the gateway.
Solution & Recommendation	The reason of this erratum2 is same as erratum1 so the solution is also same with erratum1. Please refer to the solution of erratum1.

Erratum 3	
Phenomenon and Condition	When W5200 is used in MACRAW mode, sometimes the received packet size can be larger than the actual received packet size. This phenomenon occurs even if the received packet size is less than the allowed packet size; if this is the case, the received packet is thrown away because it is larger than the allowed packet size. However, when the TCP / UDP mode of the Hardwired W5200 is used, the phenomenon of the erratum3 does not occur. **- When the received packet size is larger than the actual received packet size.** The received MACRAW data format is shown in the figure below. The PACKET-INFO (the byte size of data packet) can also be larger than the actual packet size in this phenomenon. Therefore, the Dummy data is inserted from the actual packet size to the PACKET-INFO size.  **Figure The received MACRAW data Format** **- When the received packet is thrown out because its size is larger than the allowed packet size.** The allowed packet size is defined as 1514 bytes. W5200 throws out the packet which size is over the allowed packet size.
Solution and Recommendation	**- When the received packet size is larger than the actual received packet size.** To avoid this phenomenon, user should check the actual packet size by using the total length filed in IP header. IP header is assigned in the payload of the received MACRAW data, and the total length filed, which denotes the length of the datagram exists in the IP Header. Hence, the user should set the software stack to use the total length in IP Header to determine the actual received packet size when using the software stack. **- When the received packet is thrown out because its size is larger than the allowed packet size.** User should limit the use of buffer in order to avoid receiving or transmitting the maximum packet size in TCP/UDP. In the case of TCP, the Windows size (RX buffer) should be limited to the size of 1459 bytes or under. In the case of UDP, the packet size that can be sent from the Application Layer should be limited to the size of 1452 bytes or under.