High-Performance Internet Connectivity Solution

W5300

Version 1.2.4

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

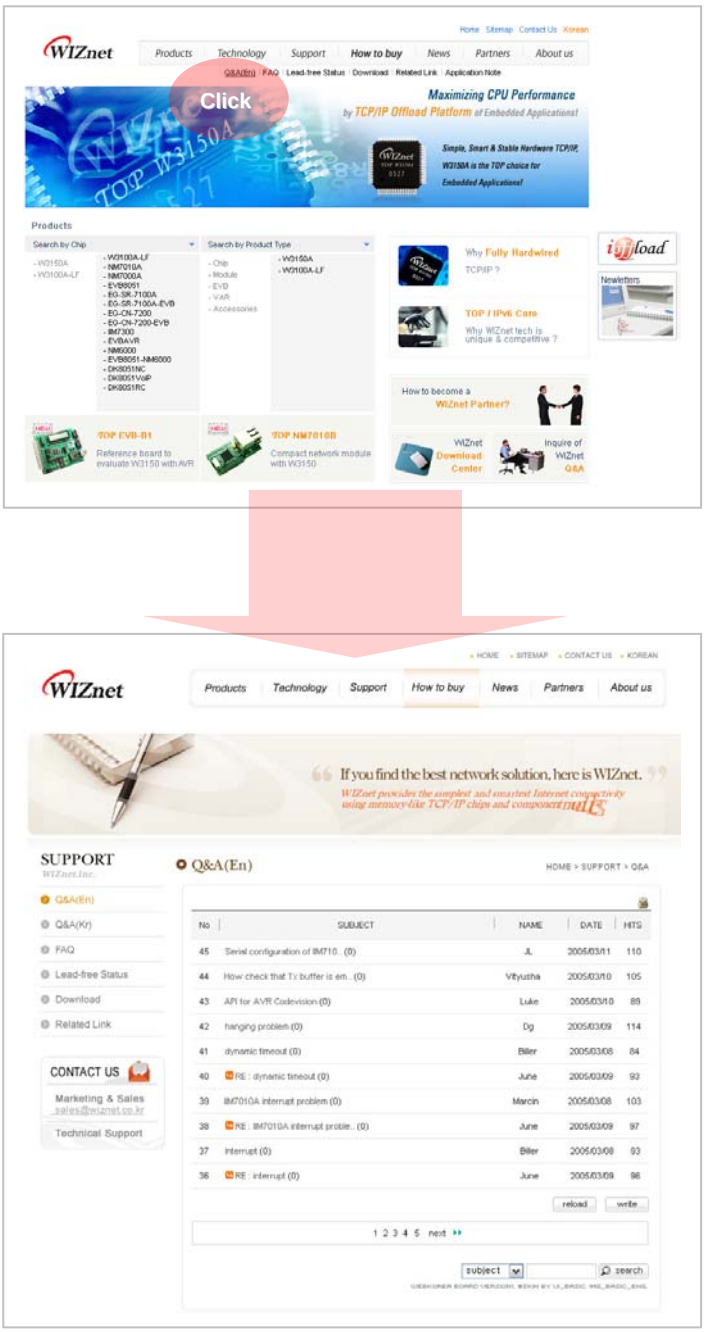
Document History Information

Version	Date	Descriptions
Ver. 1.0.0	Mar. 11, 2008	Release with W5300 launching
Ver. 1.1.0	May. 15, 2008	◦ 오타 수정 ◦ 4.4 SOCKET Register >> Sn_DPORTR R/W → WO, 설명 수정, P.77 참조 ◦ 4.4 SOCKET Register >> Sn_MSSR MSS표에서 MACRAW의 PPPoE MSS값 수정 (1502 → 1514) P.79 참조 ◦ 5.2.1.1 TCP SERVER >> ▪ ESTABLISHED : Receiving process <Notice>의 example code 수정 Sn_CR_SEND 대신 Sn_CR_SEND_KEEP 사용. P.93 참조 ◦ 5.2.4 MACRAW >> ▪ Receiving process <NOTICE>에서 Free Size 1526→1528, CRC(2) → CRC(4) P.110 참조
Ver. 1.1.1	July 4, 2008	◦ 오타 수정 ◦ 5.2.1.1 TCP SERVER >>ESTABLISHED : Receiving process <Notice>의 example code 수정 Sn_CR_SEND_KEEP 대신 Sn_CR_SEND 사용. P.93 참조
Ver. 1.2	Dec. 30, 2008	◦ 1. PIN Description '8' Symbol 추가 ◦ 1.2 Configuration Signals ADDR type 변경 (ID → I), No Internal Pulled down DATA[15:0] type 변경 (IO → IO8) ◦ 6.2. Indirect Address Mode ADDR[9:3]은 Internal Pulled-down되어 있지 않아, Indirect Address Mode로 사용할 경우 반드시 Ground 처리해야 한다. 이에 따른 설명과 그림 수정.

V1.2.1	Jan. 22, 2009	◦ Figure 2 수정 – Ferrite Bead 0.1uF → 1uH
V1.2.2	Feb. 16, 2009	◦ 1.7 Clock Signals – XTLP/XTLN PIN Type 삭제 ◦ 7. Electrical Specifications – DC Characteristics : V_{OH}, V_{OL} Test Condition 수정 : V_{OH} – Min (2.0→2.4), Typical 삭제, Max 삭제 : V_{OL} – Min 삭제 , Typical 삭제
V1.2.3	Feb.11, 2010	◦Figure 2 수정 –W5300 Power Supply Signal schematic 변경
V1.2.4	Aug. 19, 2010	- Change Temperature condition

WIZnet's online Technical Support

If you have something to ask about WIZnet Products, write down your question on Q&A Board of 'Support' menu in WIZnet website (www.wiznet.co.kr). WIZnet Engineer will give an answer as soon as possible.



W5300

W5300 은 위즈네트의 Hardware TCP/IP 기술을 이용한 임베디드 시스템을 위한 인터넷 솔루션 중 멀티미디어 서비스에 적합한 고성능에 목적을 둔 제품이다. 기존의 위즈네트 칩에 비해서 메모리 및 데이터 처리 부분을 개선하여 성능을 향상시켰으며, 최근 각광을 받고 있는 IPTV, IP-STB 등의 대용량 멀티미디어 데이터 전송에 대응할 수 있도록 개발된 제품이다. W5300 하나의 칩으로 TCP/IP 프로토콜 처리 및 10/100 Ethernet PHY 와 MAC 을 구현하여 개발하고자 하는 Application 에 Internet Connectivity 를 손쉽게 구현할 수 있도록 지원한다.

High-Performance Hardware TCP/IP single chip solutions

위즈네트에서는 TCP, UDP, IPv4, ICMP, IGMP, ARP, PPPoE 등의 통신 프로토콜을 Full hardware logic으로 개발하여 여러 제품에서 사용하고 있다. W5300에서는 보다 고성능의 데이터 통신을 제공하기 위해서 data communication memory를 128Kbyte로 확장하고, 16bit bus interface를 지원한다. 이에 사용자는 W5300의 하드웨어로 처리되는 8개의 독립적인 고속의 하드웨어 SOCKET을 사용할 수 있다.

More flexible memory allocation for various applications

W5300의 data communication memory는 사용자의 설정에 따라 각 SOCKET별로 0~64Kbytes 범위에서 조절할 수 있으므로 사용하고자 하는 application에 맞춰 자유롭게 사용할 수 있다. 따라서 사용자는 고성능이 요구되는 application에 자원을 집중하여 보다 효율적으로 시스템을 구성할 수 있도록 유연한 메모리 사이즈 할당 기능을 제공한다.

Easy to implements for beginners

W5300의 Host Interface 방식은 SRAM 메모리등과 같은 System bus interface를 제공하며 Direct address access 방식과 Indirect address access방식을 지원하여 메모리를 사용하듯 쉽게 사용할 수 있도록 한다. 또한 W5300의 Data communication memory는 각 SOCKET 별로 존재하는 송신 FIFO Register와 수신 FIFO Register를 통해서 간단히 Access 할 수 있도록 하여 보다 쉽고 간단히 W5300을 사용할 수 있도록 하여 네트워크를 처음 접하는 엔지니어도 쉽게 Internet connectivity를 구현할 수 있도록 지원한다.

Target Applications

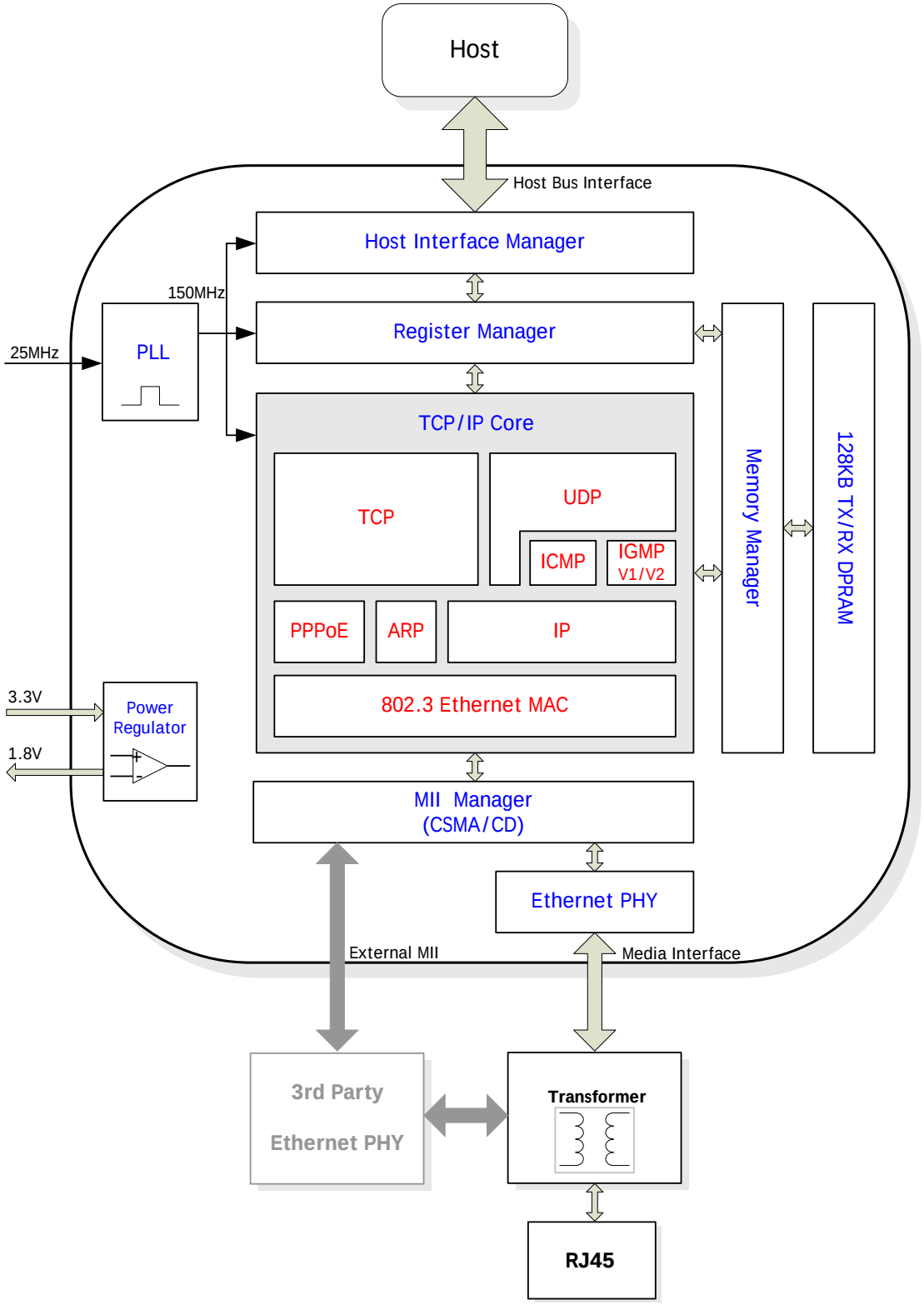
W5300은 다음과 같은 Embedded application에 적합하다.

- Home Network Devices: Set-Top Boxes, PVRs, Digital Media Adapters
- Serial-to-Ethernet: Access Controls, LED displays, etc.
- Parallel-to-Ethernet: POS / Mini Printers, Copiers
- USB-to-Ethernet: Storage Devices, Network Printers
- GPIO-to-Ethernet: Home Network Sensors
- Security Systems: DVRs, Network Cameras, Kiosks
- Factory and Building Automation
- Medical Monitoring Equipment
- Embedded Servers

Features

- Supports hardwired TCP/IP protocols : TCP,UDP,ICMP,IPv4,ARP,IGMPv2,PPPoE, Ethernet
- Supports 8 independent SOCKETs simultaneously
- High network performance : Up to 50Mbps
- Supports hybrid TCP/IP stack(software and hardware TCP/IP stack)
- Supports PPPoE connection (with PAP/CHAP Authentication mode)
- IP Fragmentation is not supported
- Internal 128Kbytes memory for data communication(Internal TX/RX memory)
- More flexible allocation internal TX/RX memory according to application throughput
- Supports memory-to-memory DMA (only 16bit Data bus width & slave mode)
- Embedded 10BaseT/100BaseTX Ethernet PHY
- Supports auto negotiation (Full-duplex and half duplex)
- Supports auto MDI/MDIX(Crossover)
- Supports network Indicator LEDs (TX, RX, Full/Half duplex, Collision, Link, Speed)
- Supports a external PHY instead of the internal PHY
- Supports 16/8 bit data bus width
- Supports 2 host interface mode(Direct address mode & Indirect address mode)
- External 25MHz operation frequency (For internal PLL logic, period=40ns)
- Internal 150MHz core operation frequency (PLL_CLK, period=about 6.67ns)
- Network operation frequency (NIC_CLK : 25MHz(100BaseTX) or 2.5MHz(10BaseT))
- 3.3V operation with 5V I/O signal tolerance
- Embedded power regulator for 1.8V core operation
- 0.18 μm CMOS technology
- 100LQFP 14X14 Lead-Free Package

Block Diagram



PLL(Phase-Locked Loop)

25MHz의 Clock source를 6배 빠른 150MHz clock으로 생성한다. 생성된 150MHz clock은 TCP/IP core block, Host Interface Manager, Register Manager와 같은 내부 Block의 Operation clock으로 사용된다. PLL은 Reset 후 Lock-in되어 안정된 clock을 공급한다.

Power Regulator

3.3V Power를 입력 받아 1.8V/150mA의 Power를 생성한다. 이 Power Regulator는 W5300의 Core operation power를 공급한다. 외부에 다른 Power regulator 장착이 필요 없다. 보다 안정적인 1.8V Power 공급을 위해 외부에 Recommended capacitor를 장착한다.

Host Interface Manager

Host bus signal을 감지하고, Data bus width나 Host interface mode 설정에 따라 Host의 Read/Write operation을 관리한다.

Register Manager

Mode register, COMMON register, SOCKET register등의 주요 Register들을 관리한다.

Memory Manager

128Kbytes의 Internal data memory를 관리하고, Host에 의해 할당된 각 SOCKET의 TX/RX memory를 관리한다. Host는 각 SOCKET의 TX/RX FIFO Register을 통해서만 이 Memory를 Access할 수 있다.

128KB TX/RX DPRAM

128Kbytes의 Data memory로, 8Kbytes DPRAM(Dual-Port RAM) 16개로 구성된다. Host에 의해 각 SOCKET 별로 flexible하게 할당된다.

MII(Media Independent Interface) Manager

MII interface를 관리한다. MII interface은 TEST_MODE[3:0]의 설정에 따라 Internal PHY나 External PHY(3rd Party PHY)로 Switching된다.

Internal Ethernet PHY

10BaseT/100BaseTX Ethernet PHY로, Half-duplex/Full-duplex, Auto-negotiation, Auto MDI/MDIX를 지원한다. 또한 Link status, Speed, Duplex와 같은 6개의 Network Indicator LED output signal을 지원한다.

TCP/IP Core

WIZnet의 Network protocol processing 기술로, TCP/IP stack이 Fully hardwired logic으로 구현

된다.

- 802.3 Ethernet MAC(Media Access Control)

  CSMA/CD(Carrier Sense Multiple Access with Collision Detect) 방식의 Ethernet 접근을 제어한다. 48bits의 Source/Destination MAC address를 기반으로 하는 Protocol 기술이다. Hardware TCP/IP stack뿐만 아니라, 0번째 SOCKET을 이용해 Host가 직접적으로 MAC layer를 Control할 수 있게 하여 Software TCP/IP stack을 구현할 수 있다.

- PPPoE(Point-To-Point Protocol over Ethernet)

  Ethernet상에서 PPP service를 이용하게 하는 Protocol 기술이다. 이는 Ethernet frame의 Payload(Data)부분을 PPP frame으로 Encapsulation하여 전송하며, PPP frame을 De-capsulation하여 수신한다. PPPoE는 PPPoE server와의 PPP 통신을 지원하며 PAP/CHAP 방식의 Authentication만을 지원한다.

- ARP(Address Resolution Protocol)

  IP address를 이용한 MAC address를 Resolution하는 Protocol 기술이다. Peer로부터 수신한 ARP-request에 대한 ARP-reply를 전송하며, Peer의 MAC address를 찾는 ARP-request를 전송하며, 그에 대한 ARP-reply를 수신하여 처리한다.

- IP(Internet Protocol)

  IP layer의 Data 통신을 지원하는 Protocol 기술이다. IP fragmentation은 지원하지 않는다. Fragmentation이 발생한 모든 Packet은 수신할 수 없다.

  TCP나 UDP를 제외한 모든 protocol number를 지원한다. TCP나 UDP는 이미 구현되어 있는 Hardwired Stack을 이용한다.

- ICMP(Internet Control Message Protocol)

  Ethernet상의 Fragment MTU와 Unreachable destination의 ICMP packet들을 수신하고 이를 Host에게 알리며, Ping-request ICMP packet을 수신하여 Ping-reply ICMP packet을 전송한다. Ping-request size는 119 Bytes 이상 지원하지 않는다.

- IGMPv1/v2(Internet Group Management Protocol version 1/2)

  UDP에서 Multicasting 통신을 할 경우 IGMP Join/Leave, Report와 같은 IGMP를 처리한다. IGMP logic은 Version 1과 2만을 지원한다. 상위 버전의 IGMP를 사용하고자 할 경우 IP layer를 이용하여 직접 구현한다.

- UDP(User Datagram Protocol)

  UDP layer의 Data 통신을 지원하는 Protocol 기술이다. Unicast, Multicast, Broadcast 방식의 User datagram을 지원한다.

- TCP(Transmission Control Protocol)

  TCP layer의 Data 통신을 지원하는 Protocol 기술이다. “TCP SERVER”와 “TCP CLIENT” 통신을 지원한다.

W5300은 모든 Protocol 처리를 Host 개입 없이 순수 Hardware logic으로만 처리하여, TCP/IP stack 처리에 대한 Host overhead를 줄여, Host의 자원을 보다 효율적으로 활용할 수 있도록 해주는 TOE(TCP/IP Offload Engine) 기술을 기반으로 하고 있다.

Table of Contents

Table of Contents 10
List of Figures 11
1. PIN Description 12
1.1 PIN Layout 12
1.2 Configuration Signals 13
1.3 Host Interface Signals 14
1.4 Media Interface Signals 16
1.5 MII Interface signal for external PHY 17
1.6 Network Indicator LED Signals 19
1.7 Clock Signals 20
1.8 Power Supply Signals 20
2. System Memory Map 23
3. W5300 Registers 25
3.1 Mode Register 26
3.2 Indirect Mode Registers 26
3.3 COMMON registers 26
3.4 SOCKET registers 30
4. Register Description 46
4.1 Mode Register 47
4.2 Indirect Mode Registers 49
4.3 COMMON Registers 51
4.4 SOCKET Registers 67
5. Functional Description 89
5.1 Initialization 89
5.2 Data Communication 91
5.2.1 TCP 91
5.2.2 UDP 100
5.2.3 IPRAW 107
5.2.4 MACRAW 109
6. External Interface 115
6.1 Direct Address Mode 115
6.1.1 16 Bit Data Bus Width 115
6.1.2 8 Bit Data Bus Width 115
6.2 Indirect Address Mode 116
6.2.1 16 Bit Data Bus Width 116

6.2.2 8 Bit Data Bus Width 116
6.3 Internal PHY Mode 117
6.4 External PHY Mode 118
7. Electrical Specifications 119
8. IR Reflow Temperature Profile (Lead-Free) 123
9. Package Descriptions 124

List of Figures

Fig 1. PIN Layout 12
Fig 2. Power Design 22
Fig 3. Memory Map 24
Fig 4. 'BRDYn' Timing 66
Fig 5. SOCKETn Status Transition 78
Fig 6. Access to Internal TX Memory 87
Fig 7. Access to Internal RX Memory 88
Fig 8. Allocation Internal TX/RX memory of SOCKETn 90
Fig 9. "TCP SERVER" & "TCP CLIENT" 91
Fig 10. "TCP SERVER" Operation Flow 92
Fig 11. The received TCP data format 94
Fig 12. "TCP CLIENT" Operation Flow 99
Fig 13. UDP Operation Flow 100
Fig 14. The received UDP data format 102
Fig 15. IPRAW Operation Flow 108
Fig 16. The received IPRAW data format 109
Fig 17. MACRAW Operation Flow 110
Fig 18. The received MACRAW data format 111
Fig 19. Internal PHY & LED Signals 117
Fig 20. External PHY Interface with MII 118

1. PIN Description

Type	Description	Type	Description
I	Input	D	Internal pulled-down with 75KΩ resistor
O	Output with driving current 2mA	M	Multi-function
IO	Input/Output (Bidirectional)	H	Active high
U	Internal pulled-up with 75KΩ resistor	L	Active low
O8	Output driving current 8mA		

<Notation> IUL : Input PIN with 75KΩ pull-up resistor. Active low

OM : Multi-functional output PIN

1.1 PIN Layout

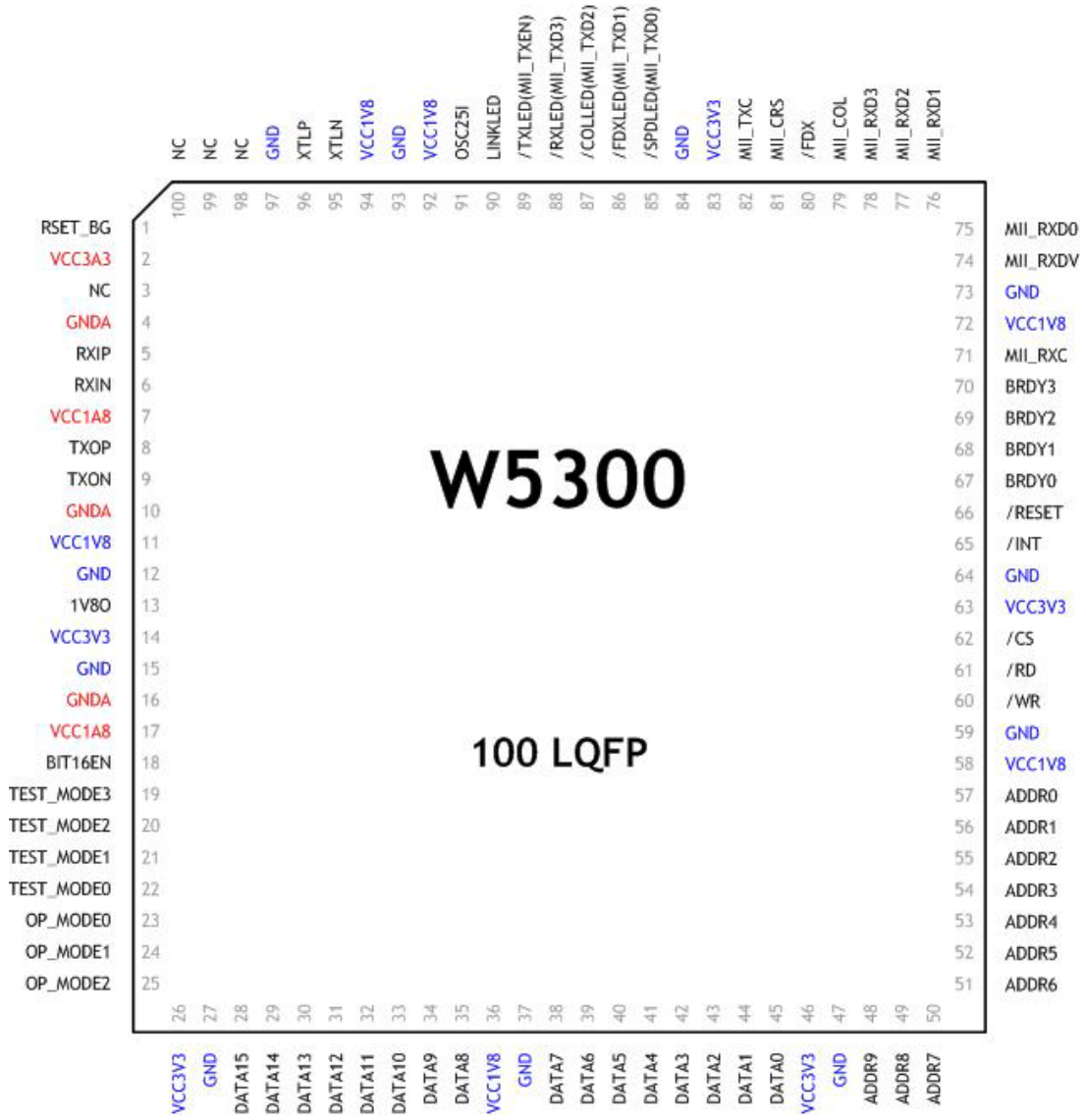


Fig 1. PIN Layout

1.2 Configuration Signals

<table>
<tr><th>Symbol</th><th>Type</th><th>Description</th></tr>
<tr><td>TEST_MODE[3:0]</td><td>ID</td><td>W5300 MODE SELECT

W5300의 PHY mode 및 Factory test mode를 설정한다.
<table>
<tr><th colspan="4">TEST_MODE</th><th rowspan="2">Description</th></tr>
<tr><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Internal PHY mode
Normal operation mode,
Auto-negotiation enable with all capabilities</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>External PHY mode with crystal clock</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>External PHY mode with oscillator clock</td></tr>
<tr><td colspan="4">Others</td><td>Reserved (Factory test mode)</td></tr>
</table>
External PHY mode에서 Clock source에 따라 사용되는 Clock input pin이 달라진다. “1.7 Clock Signals” 참조.</td></tr>
<tr><td>OP_MODE[2:0]</td><td>ID</td><td>Internal PHY Operation Control Mode

Internal PHY의 여러 가지 동작 Mode를 설정한다.
<table>
<tr><th colspan="3">OP_MODE</th><th rowspan="2">Description</th></tr>
<tr><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Normal operation mode, 권장
Auto-negotiation enable with all capabilities</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Auto-negotiation with 100 BASE-TX FDX/HDX ability</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Auto-negotiation with 10 BASE-T FDX/HDX ability</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>Manual selection of 100 BASE-TX FDX</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>Manual selection of 100 BASE-TX HDX</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Manual selection of 10 BASE-T FDX</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Manual selection of 10 BASE-T HDX</td></tr>
</table>
cf> FDX : Full-duplex, HDX : Half-duplex

설정 값은 Hardware reset 이후 Latch된다.</td></tr>
</table>

1.3 Host Interface Signals

Symbol	Type	Description
/RESET	IL	**RESET** Hardware reset signal. W5300을 초기화한다. RESET은 Low assert 이후 최소 2us 이상 유지해야 하고, High de-assert 이후 내부 PLL logic이 안정화될 때까지 최소 10ms 이상 대기해야 한다. "7 Electrical Specification"의 RESET Timing 참조. W5300은 Power-On-Reset을 지원하지 않는다. 따라서 Power-On-Reset Circuit를 Target system에 설계해야 한다.
BIT16EN	IU	16/8 BIT DATA BUS SELECT High : 16 bit data bus Low : 8 bit data bus Data bus width를 결정한다. 이 Signal은 Reset 시 내부적으로 Mode register(MR)의 15번째 Bit('BW')로 Latch되며, Reset 이후의 Signal 변화는 무시된다. 즉 Reset 이후 Data bus width를 변경할 수 없다. 8bit data bus를 사용할 경우 반드시 Ground 처리한다.
ADDR9-0	I	ADDRESS System address bus. W5300의 Host interface mode와 Data bus width에 따라 선택적으로 사용될 수 있다. 16 Bit Data bus를 사용할 경우 ADDR0은 내부적으로 무시된다. "6. External Interface" 참조.
DATA[15:8]	IO8	DATA System high data bus. W5300 register의 Read/Write에 사용된다. 8bit data bus를 사용할 경우 High-Z 상태가 된다.(High-Z driven)
DATA[7:0]	IO8	DATA System low data bus. W5300 register의 Read/Write에 사용된다.

/CS	IL	CHIP SELECT Chip Select Signal. Host는 W5300 Read/Write operation시 W5300을 선택한다. '/CS' signal이 High de-assert된 경우 DATA[15:0]은 High-Z가 된다.
/WR	IL	**WRITE ENABLE** Write Enable Signal. Host에서 ADDR[9:0]으로 선택한 W5300 register에 DATA[15:0] 값을 Write하도록 한다. DATA[15:0]은 W5300 Write data fetch timing 설정에 따라 W5300으로 Latch 된다. (MR의 13-11번째 bit(WDF[2:0]) 참조)
/RD	IL	READ ENABLE Read Enable Signal. Host에서 ADDR[9:0]으로 선택한 W5300 register를 DATA[15:0]를 통해 Read하도록 한다.
/INT	OL	INTERRUPT Interrupt Request Signal. W5300이 동작 중 Interrupt (Connected, Disconnected, Data Received, Data Sent, or Timeout)가 발생할 경우 Low assert되며, Host의 Interrupt service가 완료되고 Interrupt register(IR)가 Clear될 때 High de-assert된다. IR, Interrupt Mask Register(IMR), SOCKETn Interrupt Register(Sn_IR), SOCKETn Interrupt Mask Register(Sn_IMR) 참조.
BRDY[3:0]	O	Buffer Ready Indicator 각 PIN들은 사용자에 의해 SOCKET Number, Memory Type, Buffer Depth등을 설정되고, 설정된 SOCKET의 Memory가 Buffer Depth보다 크거나 같을 경우 High나 Low로 Signal된다. "4.3 COMMON Registers"의 Pn_BRDYR과 Pn_DPTHR 참조.

1.4 Media Interface Signals

W5300의 Internal PHY mode(TEST_MODE[3:0] = "0000", "1.2 Configuration Signals" 참조)를 사용할 경우, Network media(10Mbps/10Mbps)와 Interface하기 위한 Signals이다.

Symbol	Type	Description
RXIP	I	RXIP/RXIN Signal Pair Differential receive Input signal pair. Media로부터 Data을 수신한다. 이 Signal pair는 더 좋은 Impedance matching을 위해 2개의 Termination resistor 50Ω(±1%)과 1개의 Capacitor 0.1uF이 필요하며, 이 Resistor/Capacitor pair는 Magnetic(Transformer)에 보다 가깝게 위치시킨다. 사용하지 않을 경우 Ground 처리한다.
RXIN	I	
TXOP	O	TXOP/TXON Signal Pair Differential transmit output signal pair. Media로 Data를 송신한다. 이 Signal pair는 더 좋은 Impedance matching을 위해 2개의 Termination resistor 50Ω(±1%)과 1개의 Capacitor 0.1uF이 필요하며, 이 Resistor/Capacitor pair는 W5300에 보다 가깝게 위치시킨다. 사용하지 않을 경우 Float시킨다.
TXON	O	
RSET_BG	O	Off-chip Resistor 이 PIN은 12.3 kΩ(±1%) Resistor을 통해 Ground로 반드시 연결한다.

안정적인 동작을 위한 권장사항이다.

1. RXIP/RXIN signal pair(RX)의 길이를 가능한 같게 한다.
2. TXOP/TXON signal pair(TX) 의 길이를 가능한 같게 한다.
3. RXIP와 RXIN signal은 최대한 가깝게 위치시킨다.
4. TXOP와 TXON signal은 최대한 가깝게 위치시킨다.
5. RX와 TX signal pair는 bias resistor나 crystal 같은 noisy signals과는 최대한 멀리 떨어지도록 한다.

자세한 내용은 "W5100 Layout Guide.pdf"를 참조하라.

1.5 MII Interface signal for external PHY

MII interface signal들은 W5300의 Internal PHY를 사용하지 않고, External PHY를 사용할 경우 External PHY와 Interface를 위한 Signal들이다.

External PHY Mode(TEST_MODE[3:0] = "0001" or "0010")일 때 사용된다. "1.2 Configuration Signals" 참조. Internal PHY mode를 사용할 경우 Multi-function PIN들을 제외한 나머지 PIN들은 Internal pulled-down 되어있으므로 float 시켜도 무방하다.

Symbol	Type	Description
/TXLED(**MII_TXEN**)	OMH	Transmit Act LED / **Transmit Enable** MII_TXD[3:0]으로 출력되는 Transmit packet이 Valid 함을 알리는 signal이다. Transmit packet이 MII_TXD[3:0]를 통해 MII_TXC clock에 동기화되어 Nibble 단위로 출력될 때 High assert되며, Transmit packet의 마지막 Nibble data가 출력된 후 Low de-assert된다.
/RXLED(**MII_TXD3**)	OM	/RXLED,/COLLED,/FDXLED,/SPDLED / **Transmit data output** MII_TXEN이 High일 때, Transmit packet이 Nibble 단위로 MII_TXC clock에 동기화되어 출력된다. MII_TXD3가 Most Significant Bit(MSB)이다.
/COLLED(**MII_TXD2**)		
/FDXLED(**MII_TXD1**)		
/SPDLED(**MII_TXD0**)		
MII_TXC	ID	**Transmit Clock Input** External PHY로부터 입력되는 연속적인 Transmit clock으로 100BaseTX일 때 25MHz, 10BaseT일 때 2.5MHz이다. Transmit clock은 MII_TXD[3:0]의 Timing reference로 사용되며, W5300의 Network operation을 위한 Clock(NIC_CLK)으로 사용된다. Rising edge sensitive.
MII_CRS	IDH	**Carrier Sense** Media의 Link traffic을 알려주는 Signal로 Media의 Carrier가 Idle이 아닐 경우(Carrier present) High assert된다.
MII_COL	IDH	**Collision Detect** Media 상에서 Collision이 검출되면 High assert된다. Half-duplex에서만 유효하며, Full-duplex는 무시된다. Asynchronous signal.

MII_RXD3	ID	**Receive Data Input** MII_RXDV가 High일 때, Receive packet이 Nibble 단위로 MII_RXC에 동기화되어 입력된다. MII_RXD3가 MSB이다.
MII_RXD2		
MII_RXD1		
MII_RXD0		
MII_RXDV	IDH	**Receive Data Valid** MII_RXD[3:0]으로부터 입력되는 Receive packet이 Valid 함을 알리는 signal이다. Receive packet이 MII_RXD[3:0]으로부터 MII_RXC clock에 동기화되어 Nibble 단위로 입력될 때 High assert되며, Receive packet의 마지막 Nibble data가 입력된 후 Low de-assert된다. MII_RXC가 Rising edge일 때 Valid하다.
MII_RXC	ID	**Receive Clock Input** External PHY로부터 입력되는 Continuous receive clock으로 100BaseTX일 때 25MHz, 10BaseT일 때 2.5MHz이다. Receive clock은 MII_RXD[3:0]와 MII_RXDV의 Timing reference로 사용. Rising edge sensitive.
/FDX	IDL	**Full-Duplex Select** 0 : Full-duplex 1 : Half-duplex External PHY의 현재 Link된 상태(Full/Half-duplex)를 입력받는 Signal이다. 대부분의 External PHY는 Auto-negotiation을 지원하고 그 결과를 Network Indicator LED나 그 외의 Signals을 통해 알려준다. /FDX는 이런 Signal과 연결될 수 있으며, 또한 High나 Low를 직접 입력하여 Manually 설정이 가능하다.

안정적인 동작을 위한 권장사항이다.

1. MII Interface Signal의 길이는 25cm를 가능한 넘지 않도록 한다.
2. MII_TXD[3:0] Signal의 길이는 가능한 같게 한다.
3. MII_RXD[3:0] Signal의 길이는 가능한 같게 한다.
4. MII_TXD[3:0]와 MII_TXC에서, MII_TXC의 길이는 MII_TXD[3:0]의 길이보다 2.5cm를 초과하지 않도록 한다.
5. MII_RXD[3:0]와 MII_RXC에서, MII_RXC의 길이는 MII_RXD[3:0]의 길이보다 2.5cm를 초과하지 않도록 한다.

1.6 Network Indicator LED Signals

LINKLED를 제외한 나머지 Signal은 TEST_MODE[3:0]의 설정에 따라 Multi-function PIN으로 사용된다. 이 Signal을 Network indicator signal로 사용할 경우 Internal PHY mode (TEST_MODE[3:0]="0000")로 설정해야 된다.

Symbol	Type	Description
LINKLED	OML	**Link LED** Media(10/100M)의 연결 상태를 알려준다.
/TXLED(MII_TXEN)	OML	**Transmit activity LED**/Transmit Enable TXOP/TXON을 통한 Transmit Data의 출력(Transmit Activity)을 알린다.
/RXLED(MII_TXD3)	OML	**Receive activity LED**/Transmit Data RXIP/RXIN로부터의 Receive Data의 입력(Receive activity)을 알린다. cf> /TXLED와 /RXLED signal을 'AND' gate로 연결하여 Network activity LED로 사용할 수 있다.
/COLLED(MII_TXD2)	OML	**Collision LED**/Transmit Data Collision 발생을 알린다. Half-duplex에서만 유효하며, Full-duplex이면 High를 유지한다.
/FDXLED(MII_TXD1)	OML	**Full duplex LED**/Transmit Data Auto-negotiation이나 OP_MODE[2:0]의 Manual 설정에 따라, Full-duplex이면 Low, Half-duplex이면 High.
/SPDLED(MII_TXD0)	OML	**Link speed LED**/Transmit Data Auto-negotiation이나 OP_MODE[2:0]의 Manual 설정에 따라, 100Mbps 이면 Low, 10Mbps 이면 High.

1.7 Clock Signals

W5300은 Crystal과 Oscillator를 Clock source로 사용할 수 있으나, 보다 안정적인 동작을 위해 Crystal clock 사용을 권장한다.

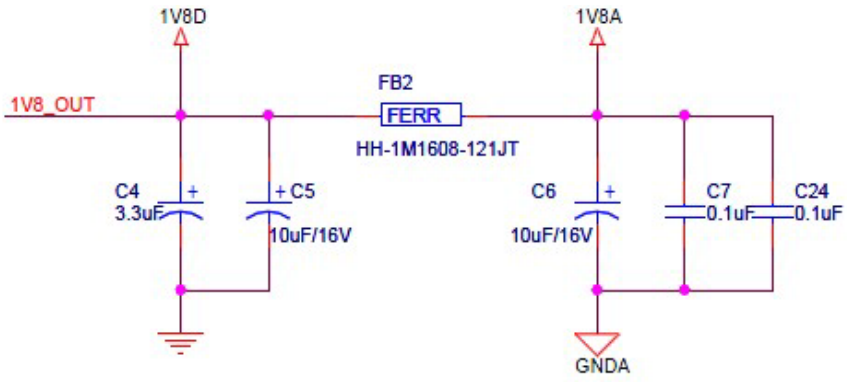
Clock source로부터 입력되는 25MHz frequency는 W5300의 PLL logic을 거쳐 생성된 150MHz frequency로 생성된다. PLL logic을 거쳐 생성된 150MHz frequency는 PLL_CLK(Period 6.67ns)으로 W5300 core operation clock으로 사용된다.

Symbol	Type	Description
XTLP		25MHz crystal input/output 25MHz parallel-resonant crystal은 Internal oscillator stabilization을 위해 Matching capacitor와 함께 연결되어 사용된다. "7.Electrical Specifications"의 Clock Characteristics 참조.
XTLN		이 Signal은 Internal PHY mode(TEST_MODE[3:0]="0000")나 External PHY mode with crystal clock (TEST_MODE[3:0] = "0001")일 때만 사용된다. Internal PHY mode에서 Oscillator를 사용할 경우, 반드시 1.8V Level의 Oscillator를 사용하며, Clock source를 XTLP에만 연결하고 XTLN은 float 시킨다.
OSC25I	I	25MHz Oscillator input External PHY mode with oscillator clock(TEST_MODE[3:0]="0010")일 때만 사용된다. 이 때 XTLP는 leakage current를 방지하기 위해 반드시 High로 유지하고 XPLN을 Float 시키며, 1.8V Level의 oscillator를 사용해야 한다.

1.8 Power Supply Signals

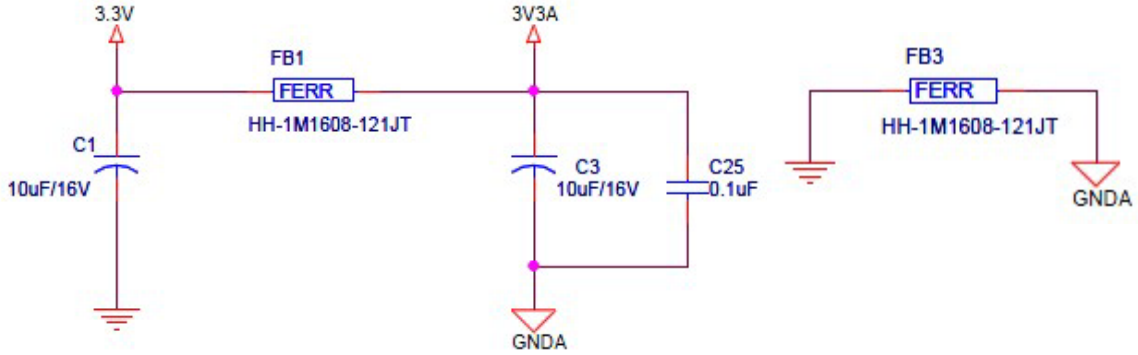
Symbol	Type	Description
VCC3A3	Power	**3.3V power supply for Analog part** VCC3A3과 GNDA사이에는 Power compensation을 위한 10uF Tantalum capacitor을 반드시 연결한다.
VCC3V3	Power	3.3V power supply for Digital part 각각의 VCC3V3과 GND사이에는 0.1uF Decoupling capacitor를 선택적으로 연결한다. VCC3V3는 1uH Ferrite bead로 분리되어 VCC3A3으로 연결할 수 있다.
VCC1A8	Power	**1.8V power supply for Analog part**

		VCC1A8과 GNDA사이에는 Core power noise filtering을 위한 10uF Tantalum capacitor와 0.1uF Capacitor을 반드시 연결한다.
VCC1V8	Power	**1.8V power supply for Digital part** 각각의 VCC1V8과 GND사이에는 0.1uF Decoupling capacitor를 선택적으로 연결한다.
GNDA	Ground	**Analog ground** PCB layout 시 Analog ground plane을 가능한 넓게 한다.
GND	Ground	**Digital ground** PCB layout 시 Digital ground plane을 가능한 넓게 한다.
1V8O	O	**1.8V regulator output voltage** Internal Power regulator에서 생성되는 1.8V/150mA Power로 Core operation power(VCC1A8, VCC1V8)로 사용된다. 1V8O와 GND사이에 Output frequency compensation을 위한 3.3uF Tantalum capacitor는 반드시 연결하고, High frequency noise decoupling을 위한 0.1uF Capacitor는 선택적으로 연결한다. 1V8O은 VCC1V8와 연결되고 1uH ferrite bead로 분리되어 VCC1A8로 연결된다. <Notice> 1V8O는 W5300의 Core operation을 위한 Power이므로 다른 Device의 Power로 연결해서는 안 된다.



Place C4, C5, FB2 close to 1V8_OUT and place C6, C7, C24 close to 1V8A pin.

Place C8, C9, C10, C11, C12, C13, C14, C15, C16, C17, C18 as close to each power pin as possible.



Place FB1, C1, C3, C25 as close to each power pin as possible.

Fig 2. Power Design

Power 설계를 위한 권장사항이다.

1. Decoupling capacitor는 가능한 W5300에 가깝게 위치.
2. Ground plane은 가능한 넓게 사용.
3. Ground plane이 충분히 넓다면 Analog ground plane과 Digital ground plane를 분리.

Ground plane이 충분히 넓지 않다면, Analog ground plane과 Digital ground plane을 분리하는 것보다 Single ground plane으로 설계하는 것이 더 좋다.

2. System Memory Map

W5300은 Host interface 방식으로 Direct address mode와 Indirect address mode를 지원한다.

Direct address mode란, Target host system이 W5300 register들을 T.M.S(Target host system의 Memory-mapped I/O Space)에 Mapping한 후, Mapping된 W5300 register들을 직접적으로 Access하는 Mode를 말한다.

W5300에서의 Direct address mode memory map은 Mode register(MR), COMMON registers, SOCKET registers들로 구성되며, 이 Register들은 T.M.S의 BA(Base Address)에서부터 2Byte씩 증가하며 순차적으로 Mapping된다. Target host system은 Mapping된 Address로 W5300의 MR, COMMON registers, SOCKET registers을 직접적으로 Access할 수 있다. 따라서, Target host system이 W5300을 Direct address mode로 사용할 경우 총 0x400 Bytes의 Memory space가 필요하게 된다.

Indirect address mode란, Target host system이 W5300의 MR, Indirect mode address register(IDM_AR), Indirect mode data register(IDM_DR)만을 T.M.S에 Mapping한 후, Mapping된 IDM_AR과 IDM_DR register만을 직접적으로 Access하여, COMMON registers와 SOCKET registers를 간접적으로 Access하는 Mode를 말한다.

W5300에서의 Indirect address mode memory map은 Host가 직접적으로 Access할 수 있는 MR, IDM_AR, IDM_DR과, 간접적으로 Access할 수 있는 COMMON registers, SOCKET registers 들로 구성된다. MR, IDM_AR, IDM_DR만 T.M.S의 BA(Base Address)에서부터 2Byte씩 증가하며 순차적으로 Mapping되고, COMMON & SOCKET registers들은 IDM_AR & IDM_DR을 이용하여 간접적으로 Access되기 때문에 T.M.S에 Mapping되지 않는다. 따라서, Target host system이 Indirect address mode로 사용할 경우 0x06 Bytes의 Memory space만 필요하게 된다.

예로, Target host system이 Indirect mode로 COMMON registers의 Interrupt register(IR)를 Access할 경우

Host Write : IDM_AR를 IR의 주소 0x0002 설정 (IDM_AR = 0x0002)
IDM_DR에 0xFFFF로 설정 (IDM_DR = 0xFFFF)
Host Read : IDM_AR를 IR의 주소 0x0002 설정 (IDM_AR = 0x0002)
IDM_DR을 Read하여 Value로 저장 (Value = IDM_DR)

W5300의 Host interface mode는 MR의 'IND' bit (0번째 Bit)의 값에 따라 결정된다.

MR(0) = '0' 이면, Direct address mode
MR(0) = '1' 이면, Indirect address mode

각 Host interface 방식에 따른 Target host system memory Map은 다음과 같다.

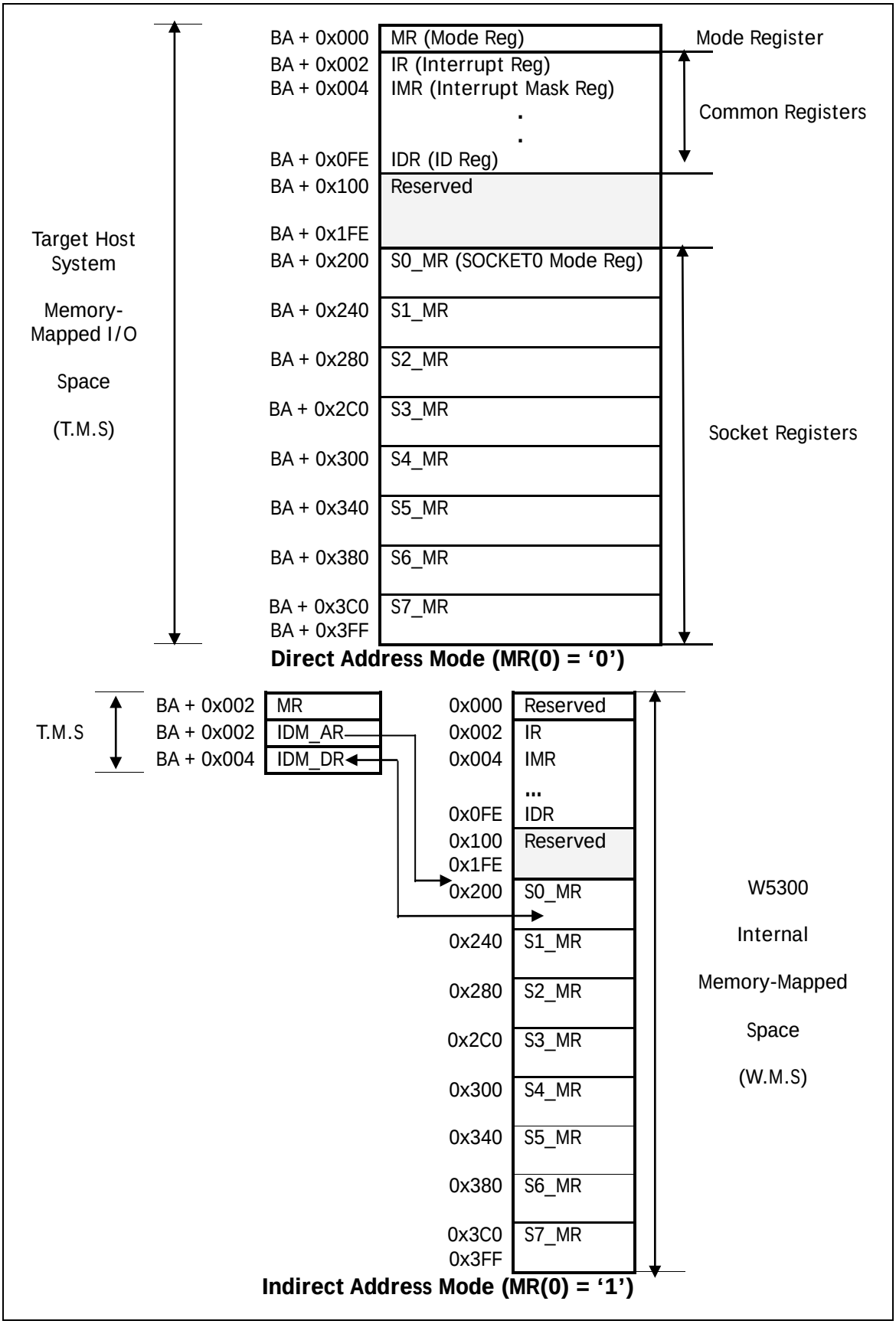


Fig 3. Memory Map

3. W5300 Registers

W5300 register은 Direct/Indirect address mode를 결정하는 MR와, Indirect address mode로 설정 시 사용되는 IDM_AR & IDM_DR와, Address mode에 상관없이 사용되는 COMMON registers, SOCKET registers로 구성된다.

MR, IDM_AR, IDM_DR은 Target host system의 T.M.S에 Mapping되며, COMMON registers와 SOCKET registers는 Address mode에 따라 T.M.S 혹은 W.M.S(W5300 Internal Memory Space)에 Mapping된다.

모든 W5300 register는 1Byte, 2Bytes, 4Bytes, 6Bytes로 구성되어 있으며, Target host system의 Data bus width에 따라 16Bit data bus인 경우 2 bytes address offset으로, 8Bit data bus인 경우 1 byte address offset으로 Access 할 수 있다.
W5300의 register가 T.M.S에 Mapping될 경우, W5300 register의 Physical T.M.S address는

Physical Address of W5300 Reg **= Base Address** of T.M.S **+ Address offset** of W5300 Reg

로 구성된다.
또한, 모든 W5300 register의 Byte ordering은 Low address byte가 Most significant byte로 사용되는 Big-endian을 사용한다.

[Register Notation]

MR : MR register
MR0 : Low address register of MR (Address offset – 0x000), Most significant byte
MR1 : High address register of MR (Address offset – 0x001), Least significant byte
MR(15:5) : MR register의 15번 bit부터 5번 bit까지 11 bit
MR(0) : MR register의 0번째 bit, MR1의 0번째 bit
MR(13) : MR register의 13번째 bit, MR0의 5번째 Bit
MR0(7) : MR register의 15번째 bit, Most significant bit of MR0
MR(DWB) : MR의 DWB bit (DWB : Bit Symbol)

SHAR : Source Hardware Address Register
SHAR0 : 1ST address register of SHAR (Address offset – 0x008)
SHAR1 : 2nd address register of SHAR (Address offset – 0x009)
SHAR2 : 3rd address register of SHAR (Address offset – 0x00A)
SHAR3 : 4th address register of SHAR (Address offset – 0x00B)
SHAR4 : 5th address register of SHAR (Address offset – 0x00C)
SHAR5 : 6th address register of SHAR (Address offset – 0x00D)

3.1 Mode Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x000	0x000	MR	MR0	Mode Register
	0x001		MR1	

3.2 Indirect Mode Registers

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x002	0x002	IDM_AR	IDM_AR0	Indirect Mode Address Register
	0x003		IDM_AR1	
0x004	0x004	IDM_DR	IDM_DR0	Indirect Mode Data Register
	0x005		IDM_DR1	

3.3 COMMON registers

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x002	0x002	IR	IR0	Interrupt Register
	0x003		IR1	
0x004	0x004	IMR	IMR0	Interrupt Mask Register
	0x005		IRM1	
0x006	0x006			Reserved
	0x007			
0x008	0x008	SHAR	SHAR0	Source Hardware Address Register
	0x009		SHAR1	
0x00A	0x00A	SHAR2	SHAR2	
	0x00B		SHAR3	
0x00C	0x00C	SHAR4	SHAR4	
	0x00D		SHAR5	
0x00E	0x00E			Reserved
	0x00F			
0x010	0x010	GAR	GAR0	Gateway Address Register
	0x011		GAR1	
0x12	0x012	GAR2	GAR2	
	0x013		GAR3	

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x014	0x014	SUBR	SUBR0	Subnet Mask Register
	0x015		SUBR1	
0x016	0x016	SUBR2	SUBR2	
	0x017		SUBR3	
0x018	0x018	SIPR	SIPR0	Source IP Address Register
	0x019		SIPR1	
0x01A	0x01A	SIPR2	SIPR2	
	0x01B		SIPR3	
0x01C	0x01C	RTR	RTR0	Retransmission Timeout-value Register
	0x01D		RTR1	
0x01E	0x01E	RCR	RCR0	Reserved
	0x01F		RCR1	Retransmission Retry-count Register
0x020	0x020	TMS01R	TMSR0	Transmit Memory Size Register of SOCKET0
	0x021		TMSR1	Transmit Memory Size Register of SOCKET1
0x022	0x022	TMS23R	TMSR2	Transmit Memory Size Register of SOCKET2
	0x023		TMSR3	Transmit Memory Size Register of SOCKET3
0x24	0x024	TMS45R	TMSR4	Transmit Memory Size Register of SOCKET4
	0x025		TMSR5	Transmit Memory Size Register of SOCKET5
0x26	0x026	TMS67R	TMSR6	Transmit Memory Size Register of SOCKET7
	0x027		TMSR7	Transmit Memory Size Register of SOCKET 8
0x028	0x028	RMS01R	RMSR0	Receive Memory Size Register of SOCKET0
	0x029		RMSR1	Receive Memory Size Register of SOCKET1
0x02A	0x02A	RMS23R	RMSR2	Receive Memory Size Register of SOCKET2
	0x02B		RMSR3	Receive Memory Size Register of SOCKET3
0x02C	0x02C	RMS45R	RMSR4	Receive Memory Size Register of SOCKET4
	0x02D		RMSR5	Receive Memory Size Register of SOCKET5
0x02E	0x02E	RMS67R	RMSR6	Receive Memory Size Register of SOCKET6
	0x02F		RMSR7	Receive Memory Size Register of SOCKET7
0x030	0x030	MTYPER	MTYPER0	Memory Block Type Register
	0x031		MTYPER1	
0x032	0x032	PATR	PATR0	PPPoE Authentication Register
	0x033		PATR1	

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x034	0x034			Reserved
	0x035			
0x036	0x036	PTIMER	PTIMER0	Reserved
	0x037		PTIMER1	PPP LCP Request Time Register
0x038	0x038	PMAGICR	PMAGICR0	
	0x039		PMAGICR1	PPP LCP Magic Number Register
0x03A	0x03A			Reserved
	0x03B			
0x03C	0x03C	PSIDR	PSIDR0	PPP Session ID Register
	0x03D		PSIDR1	
0x03E	0x03E			Reserved
	0x03F			
0x040	0x040	PDHAR	PDHAR0	PPP Destination Hardware Address Register
	0x041		PDHAR1	
0x042	0x042	PDHAR2	PDHAR2	
	0x043		PDHAR3	
0x044	0x044	PDHAR4	PDHAR4	
	0x045		PDHAR5	
0x046	0x046			Reserved
	0x047			
0x048	0x048	UIPR	UIPR0	Unreachable IP Address Register
	0x049		UIPR1	
0x04A	0x04A	UIPR2	UIPR2	
	0x04B		UIPR3	
0x04C	0x04C	UPORTR	UPORT0	Unreachable Port Number Register
	0x04D		UPORT1	
0x04E	0x04E	FMTUR	FMTUR0	Fragment MTU Register
	0x04F		FMTUR1	
0x050	0x050			Reserved
	0x051			
:				:
:				:
0x5E	0x05E			Reserved
	0x060			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x060	0x060	P0_BRDYR	P0_BRDYR0	Reserved
	0x061		P0_BRDYR1	PIN "BRDY0" Configure Register
0x062	0x062	P0_BDPTHR	P0_BDPTHR0	PIN "BRDY0" Buffer Depth Register
	0x063		P0_BDPTHR1	
0x064	0x064	P1_BRDYR	P1_BRDYR0	Reserved
	0x065		P1_BRDYR1	PIN "BRDY1" Configure Register
0x066	0x066	P1_BDPTHR	P1_BDPTHR0	PIN "BRDY1" Buffer Depth Register
	0x067		P1_BDPTHR1	
0x068	0x068	P2_BRDYR	P1_BRDYR0	Reserved
	0x069		P2_BRDYR1	PIN "BRDY2" Configure Register
0x06A	0x06A	P2_BDPTHR	P2_BDPTHR0	PIN "BRDY2" Buffer Depth Register
	0x06B		P2_BDPTHR1	
0x06C	0x06C	P3_BRDYR	P3_BRDYR0	Reserved
	0x06D		P3_BRDYR1	PIN "BRDY3" Configure Register
0x06E	0x06E	P3_BDPTHR	P3_BDPTHR0	PIN "BRDY3" Buffer Depth Register
	0x06F		P3_BDPTHR1	
0x070	0x070			Reserved
	0x071			
:				:
:				:
0xFC	0x0FC			Reserved
	0x0FD			
0xFE	0x0FE	IDR	IDR0	W5300 ID Register
	0x0FF		IDR1	

3.4 SOCKET registers

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x200	0x200	S0_MR	S0_MR0	SOCKET0 Mode Register
	0x201		S0_MR1	
0x202	0x202	S0_CR	S0_CR0	Reserved
	0x203		S0_CR1	SOCKET0 Command Register
0x204	0x204	S0_IMR	S0_IMR0	Reserved
	0x205		S0_IMR1	SOCKET0 Interrupt Mask Register
0x206	0x206	S0_IR	S0_IR0	Reserved
	0x207		S0_IR1	SOCKET0 Interrupt Register
0x208	0x208	S0_SSR	S0_SSR0	Reserved
	0x209		S0_SSR1	SOCKET0 Socket Status Register
0x20A	0x20A	S0_PORTR	S0_PORTR0	SOCKET0 Source Port Register
	0x20B		S0_PORTR1	
0x20C	0x20C	S0_DHAR	S0_DHAR0	SOCKET0 Destination Hardware
	0x20D		S0_DHAR1	Address Register
0x20E	0x20E	S0_DHAR2	S0_DHAR2	
	0x20F		S0_DHAR3	
0x210	0x210	S0_DHAR4	S0_DHAR4	
	0x211		S0_DHAR5	
0x212	0x212	S0_DPORTR	S0_DPORTR0	SOCKET0 Destination Port Register
	0x213		S0_DPORTR1	
0x214	0x214	S0_DIPR	S0_DIPR0	SOCKET0 Destination IP Address
	0x215		S0_DIPR1	Register
0x216	0x216	S0_DIPR2	S0_DIPR2	
	0x217		S0_DIPR3	
0x218	0x218	S0_MSSR	S0_MSSR0	SOCKET0 Maximum Segment Size
	0x219		S0_MSSR1	Register
0x21A	0x21A	S0_PORTOR	S0_KPALVTR	SOCKET0 Keep Alive Time Register
	0x21B		S0_PROTOR	SOCKET0 Protocol Number Register
0x21C	0x21C	S0_TOSR	S0_TOSR0	Reserved
	0x21D		S0_TOSR1	SOCKET0 TOS Register
0x21E	0x21E	S0_TTLR	S0_TTLR0	Reserved
	0x21F		S0_TTLR1	SOCKET0 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x220	0x220	S0_TX_WRSR	S0_TX_WRSR0	Reserved
	0x221		S0_TX_WRSR1	SOCKET0 TX Write Size Register
0x222	0x222	S0_TX_WRSR2	S0_TX_WRSR2	
	0x223		S0_TX_WRSR3	
0x224	0x224	S0_TX_FSR	S0_TX_FSR0	Reserved
	0x225		S0_TX_FSR1	SOCKET0 TX Free Size Register
0x226	0x226	S0_TX_FSR2	S0_TX_FSR2	
	0x227		S0_TX_FSR3	
0x228	0x228	S0_RX_RSR	S0_RX_RSR0	Reserved
	0x229		S0_RX_RSR1	SOCKET0 RX Receive Size Register
0x22A	0x22A	S0_RX_RSR2	S0_RX_RSR2	
	0x22B		S0_RX_RSR3	
0x22C	0x22C	S0_FRAGR	S0_FRAGR0	Reserved
	0x22D		S0_FRAGR1	SOCKET0 FLAG Register
0x22E	0x22E	S0_TX_FIFOR	S0_TX_FIFOR0	SOCKET0 TX FIFO Register
	0x22F		S0_TX_FIFOR1	
0x230	0x230	S0_RX_FIFOR	S0_RX_FIFOR0	SOCKET0 RX FIFO Register
	0x231		S0_RX_FIFOR1	
0x232	0x232			Reserved
	0x233			
:				:
:				:
0x23E	0x23E			Reserved
	0x23F			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x240	0x240	S1_MR	S1_MR0	SOCKET1 Mode Register
	0x241		S1_MR1	
0x242	0x242	S1_CR	S1_CR0	Reserved
	0x243		S1_CR1	SOCKET1 Command Register
0x244	0x244	S1_IMR	S1_IMR0	Reserved
	0x245		S1_IMR1	SOCKET1 Interrupt Mask Register
0x246	0x246	S1_IR	S1_IR0	Reserved
	0x247		S1_IR1	SOCKET1 Interrupt Register
0x248	0x248	S1_SSR	S1_SSR0	Reserved
	0x249		S1_SSR1	SOCKET1 Socket Status Register
0x24A	0x24A	S1_PORTR	S1_PORTR0	SOCKET1 Source Port Register
	0x24B		S1_PORTR1	
0x24C	0x24C	S1_DHAR	S1_DHAR0	SOCKET1 Destination Hardware Address Register
	0x24D		S1_DHAR1	
0x24E	0x24E	S1_DHAR2	S1_DHAR2	
	0x24F		S1_DHAR3	
0x250	0x250	S1_DHAR4	S1_DHAR4	
	0x251		S1_DHAR5	
0x252	0x252	S1_DPORTR	S1_DPORTR0	SOCKET1 Destination Port Register
	0x253		S1_DPORTR1	
0x254	0x254	S1_DIPR	S1_DIPR0	SOCKET1 Destination IP Address Register
	0x255		S1_DIPR1	
0x256	0x256	S1_DIPR2	S1_DIPR2	
	0x257		S1_DIPR3	
0x258	0x258	S1_MSSR	S1_MSSR0	SOCKET1 Maximum Segment Size Register
	0x259		S1_MSSR1	
0x25A	0x25A	S1_PORTOR	S1_KPALVTR	SOCKET1 Keep Alive Time Register
	0x25B		S1_PROTOR	SOCKET1 Protocol Number Register
0x25C	0x25C	S1_TOSR	S1_TOSR0	Reserved
	0x25D		S1_TOSR1	SOCKET1 TOS Register
0x25E	0x25E	S1_TTLR	S1_TTLR0	Reserved
	0x25F		S1_TTLR1	SOCKET1 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x260	0x260	S1_TX_WRSR	S1_TX_WRSR0	Reserved
	0x261		S1_TX_WRSR1	SOCKET1 TX Write Size Register
0x262	0x262	S1_TX_WRSR2	S1_TX_WRSR2	
	0x263		S1_TX_WRSR3	
0x264	0x264	S1_TX_FSR	S1_TX_FSR0	Reserved
	0x265		S1_TX_FSR1	SOCKET1 TX Free Size Register
0x266	0x266	S1_TX_FSR2	S1_TX_FSR2	
	0x267		S1_TX_FSR3	
0x268	0x268	S1_RX_RSR	S1_RX_RSR0	Reserved
	0x269		S1_RX_RSR1	SOCKET1 RX Receive Size Register
0x26A	0x26A	S1_RX_RSR2	S1_RX_RSR2	
	0x26B		S1_RX_RSR3	
0x26C	0x26C	S1_FRAGR	S1_FRAGR0	Reserved
	0x26D		S1_FRAGR1	SOCKET1 IP FLAG Field Register
0x26E	0x26E	S1_TX_FIFOR	S1_TX_FIFOR0	SOCKET1 TX FIFO Register
	0x26F		S1_TX_FIFOR1	
0x270	0x270	S1_RX_FIFOR	S1_RX_FIFOR0	SOCKET1 RX FIFO Register
	0x271		S1_RX_FIFOR1	
0x272	0x272			Reserved
	0x273			
:				:
:				:
0x27E	0x27E			Reserved
	0x27F			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x280	0x280	S2_MR	S2_MR0	SOCKET2 Mode Register
	0x281		S2_MR1	
0x282	0x282	S2_CR	S2_CR0	Reserved
	0x283		S2_CR1	SOCKET2 Command Register
0x284	0x284	S2_IMR	S2_IMR0	Reserved
	0x285		S2_IMR1	SOCKET2 Interrupt Mask Register
0x286	0x286	S2_IR	S2_IR0	Reserved
	0x287		S2_IR1	SOCKET2 Interrupt Register
0x288	0x288	S2_SSR	S2_SSR0	Reserved
	0x289		S2_SSR1	SOCKET2 Socket Status Register
0x28A	0x28A	S2_PORTR	S2_PORTR0	SOCKET2 Source Port Register
	0x28B		S2_PORTR1	
0x28C	0x28C	S2_DHAR	S2_DHAR0	SOCKET2 Destination Hardware Address Register
	0x28D		S2_DHAR1	
0x28E	0x28E	S2_DHAR2	S2_DHAR2	
	0x28F		S2_DHAR3	
0x290	0x290	S2_DHAR4	S2_DHAR4	
	0x291		S2_DHAR5	
0x292	0x292	S2_DPORTR	S2_DPORTR0	SOCKET2 Destination Port Register
	0x293		S2_DPORTR1	
0x294	0x294	S2_DIPR	S2_DIPR0	SOCKET2 Destination IP Address Register
	0x295		S2_DIPR1	
0x296	0x296	S2_DIPR2	S2_DIPR2	
	0x297		S2_DIPR3	
0x298	0x298	S2_MSSR	S2_MSSR0	SOCKET2 Maximum Segment Size Register
	0x299		S2_MSSR1	
0x29A	0x29A	S2_PORTOR	S2_KPALVTR	SOCKET2 Keep Alive Time Register
	0x29B		S2_PROTOR	SOCKET2 Protocol Number Register
0x29C	0x29C	S2_TOSR	S2_TOSR0	Reserved
	0x29D		S2_TOSR1	SOCKET2 TOS Register
0x29E	0x29E	S2_TTLR	S2_TTLR0	Reserved
	0x29F		S2_TTLR1	SOCKET2 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x2A0	0x2A0	S2_TX_WRSR	S2_TX_WRSR0	Reserved
	0x2A1		S2_TX_WRSR1	SOCKET2 TX Write Size Register
0x2A2	0x2A2	S2_TX_WRSR2	S2_TX_WRSR2	
	0x2A3		S2_TX_WRSR3	
0x2A4	0x2A4	S2_TX_FSR	S2_TX_FSR0	Reserved
	0x2A5		S2_TX_FSR1	SOCKET2 TX Free Size Register
0x2A6	0x2A6	S2_TX_FSR2	S2_TX_FSR2	
	0x2A7		S2_TX_FSR3	
0x2A8	0x2A8	S2_RX_RSR	S2_RX_RSR0	Reserved
	0x2A9		S2_RX_RSR1	SOCKET2 RX Receive Size Register
0x2AA	0x2AA	S2_RX_RSR2	S2_RX_RSR2	
	0x2AB		S2_RX_RSR3	
0x2AC	0x2AC	S2_FRAGR	S2_FRAGR0	Reserved
	0x2AD		S2_FRAGR1	SOCKET2 IP FLAG Field Register
0x2AE	0x2AE	S2_TX_FIFOR	S2_TX_FIFOR0	SOCKET2 TX FIFO Register
	0x2AF		S2_TX_FIFOR1	
0x2B0	0x2B0	S2_RX_FIFOR	S2_RX_FIFOR0	SOCKET2 RX FIFO Register
	0x2B1		S2_RX_FIFOR1	
0x2B2	0x2B2			Reserved
	0x2B3			
:				:
:				:
0x2BE	0x2BE			Reserved
	0x2BF			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x2C0	0x2C0	S3_MR	S3_MR0	SOCKET3 Mode Register
	0x2C1		S3_MR1	
0x2C2	0x2C2	S3_CR	S3_CR0	Reserved
	0x2C3		S3_CR1	SOCKET3 Command Register
0x2C4	0x2C4	S3_IMR	S3_IMR0	Reserved
	0x2C5		S3_IMR1	SOCKET3 Interrupt Mask Register
0x2C6	0x2C6	S3_IR	S3_IR0	Reserved
	0x2C7		S3_IR1	SOCKET3 Interrupt Register
0x2C8	0x2C8	S3_SSR	S3_SSR0	Reserved
	0x2C9		S3_SSR1	SOCKET3 Socket Status Register
0x2CA	0x2CA	S3_PORTR	S3_PORTR0	SOCKET3 Source Port Register
	0x2CB		S3_PORTR1	
0x2CC	0x2CC	S3_DHAR	S3_DHAR0	SOCKET3 Destination Hardware
	0x2CD		S3_DHAR1	Address Register
0x2CE	0x2CE	S3_DHAR2	S3_DHAR2	
	0x2CF		S3_DHAR3	
0x2D0	0x2D0	S3_DHAR4	S3_DHAR4	
	0x2D1		S3_DHAR5	
0x2D2	0x2D2	S3_DPORTR	S3_DPORTR0	SOCKET3 Destination Port Register
	0x2D3		S3_DPORTR1	
0x2D4	0x2D4	S3_DIPR	S3_DIPR0	SOCKET3 Destination IP Address
	0x2D5		S3_DIPR1	Register
0x2D6	0x2D6	S3_DIPR2	S3_DIPR2	
	0x2D7		S3_DIPR3	
0x2D8	0x2D8	S3_MSSR	S3_MSSR0	SOCKET3 Maximum Segment Size
	0x2D9		S3_MSSR1	Register
0x2DA	0x2DA	S3_PORTOR	S3_KPALVTR	SOCKET3 Keep Alive Time Register
	0x2DB		S3_PROTOR	SOCKET3 Protocol Number Register
0x2DC	0x2DC	S3_TOSR	S3_TOSR0	Reserved
	0x2DD		S3_TOSR1	SOCKET3 TOS Register
0x2DE	0x2DE	S3_TTLR	S3_TTLR0	Reserved
	0x2DF		S3_TTLR1	SOCKET3 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x2E0	0x2E0	S3_TX_WRSR	S3_TX_WRSR0	Reserved
	0x2E1		S3_TX_WRSR1	SOCKET3 TX Write Size Register
0x2E2	0x2E2	S3_TX_WRSR2	S3_TX_WRSR2	
	0x2E3		S3_TX_WRSR3	
0x2E4	0x2E4	S3_TX_FSR	S3_TX_FSR0	Reserved
	0x2E5		S3_TX_FSR1	SOCKET3 TX Free Size Register
0x2E6	0x2E6	S3_TX_FSR2	S3_TX_FSR2	
	0x2E7		S3_TX_FSR3	
0x2E8	0x2E8	S3_RX_RSR	S3_RX_RSR0	Reserved
	0x2E9		S3_RX_RSR1	SOCKET3 RX Receive Size Register
0x2EA	0x2EA	S3_RX_RSR2	S3_RX_RSR2	
	0x2EB		S3_RX_RSR3	
0x2EC	0x2EC	S3_FRAGR	S3_FRAGR0	Reserved
	0x2ED		S3_FRAGR1	SOCKET3 IP FLAG Field Register
0x2EE	0x2EE	S3_TX_FIFOR	S3_TX_FIFOR0	SOCKET3 TX FIFO Register
	0x2EF		S3_TX_FIFOR1	
0x2F0	0x2F0	S3_RX_FIFOR	S3_RX_FIFOR0	SOCKET3 RX FIFO Register
	0x2F1		S3_RX_FIFOR1	
0x2F2	0x2F2			Reserved
	0x2F3			
:				:
:				:
0x2FE	0x2FE			Reserved
	0x2FF			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x300	0x300	S4_MR	S4_MR0	SOCKET4 Mode Register
	0x301		S4_MR1	
0x302	0x302	S4_CR	S4_CR0	Reserved
	0x303		S4_CR1	SOCKET4 Command Register
0x304	0x304	S4_IMR	S4_IMR0	Reserved
	0x305		S4_IMR1	SOCKET4 Interrupt Mask Register
0x306	0x306	S4_IR	S4_IR0	Reserved
	0x307		S4_IR1	SOCKET4 Interrupt Register
0x308	0x308	S4_SSR	S4_SSR0	Reserved
	0x309		S4_SSR1	SOCKET4 Socket Status Register
0x30A	0x30A	S4_PORTR	S4_PORTR0	SOCKET4 Source Port Register
	0x30B		S4_PORTR1	
0x30C	0x30C	S4_DHAR	S4_DHAR0	SOCKET4 Destination Hardware Address Register
	0x30D		S4_DHAR1	
0x30E	0x30E	S4_DHAR2	S4_DHAR2	
	0x30F		S4_DHAR3	
0x310	0x310	S4_DHAR4	S4_DHAR4	
	0x311		S4_DHAR5	
0x312	0x312	S4_DPORTR	S4_DPORTR0	SOCKET4 Destination Port Register
	0x313		S4_DPORTR1	
0x314	0x314	S4_DIPR	S4_DIPR0	SOCKET4 Destination IP Address Register
	0x315		S4_DIPR1	
0x316	0x316	S4_DIPR2	S4_DIPR2	
	0x317		S4_DIPR3	
0x318	0x318	S4_MSSR	S4_MSSR0	SOCKET4 Maximum Segment Size Register
	0x319		S4_MSSR1	
0x31A	0x31A	S4_PORTOR	S4_KPALVTR	SOCKET4 Keep Alive Time Register
	0x31B		S4_PROTOR	SOCKET4 Protocol Number Register
0x31C	0x31C	S4_TOSR	S4_TOSR0	Reserved
	0x31D		S4_TOSR1	SOCKET4 TOS Register
0x31E	0x31E	S4_TTLR	S4_TTLR0	Reserved
	0x31F		S4_TTLR1	SOCKET4 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x320	0x320	S4_TX_WRSR	S4_TX_WRSR0	Reserved
	0x321		S4_TX_WRSR1	SOCKET4 TX Write Size Register
0x322	0x322	S4_TX_WRSR2	S4_TX_WRSR2	
	0x323		S4_TX_WRSR3	
0x324	0x324	S4_TX_FSR	S4_TX_FSR0	Reserved
	0x325		S4_TX_FSR1	SOCKET4 TX Free Size Register
0x326	0x326	S4_TX_FSR2	S4_TX_FSR2	
	0x327		S4_TX_FSR3	
0x328	0x328	S4_RX_RSR	S4_RX_RSR0	Reserved
	0x329		S4_RX_RSR1	SOCKET4 RX Receive Size Register
0x32A	0x32A	S4_RX_RSR2	S4_RX_RSR2	
	0x32B		S4_RX_RSR3	
0x32C	0x32C	S4_FRAGR	S4_FRAGR0	Reserved
	0x32D		S4_FRAGR1	SOCKET4 IP FLAG Field Register
0x32E	0x32E	S4_TX_FIFOR	S4_TX_FIFOR0	SOCKET4 TX FIFO Register
	0x32F		S4_TX_FIFOR1	
0x330	0x330	S4_RX_FIFOR	S4_RX_FIFOR0	SOCKET4 RX FIFO Register
	0x331		S4_RX_FIFOR1	
0x332	0x332			Reserved
	0x333			
:				:
:				:
0x33E	0x33E			Reserved
	0x33F			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x340	0x340	S5_MR	S5_MR0	SOCKET5 Mode Register
	0x341		S5_MR1	
0x342	0x342	S5_CR	S5_CR0	Reserved
	0x343		S5_CR1	SOCKET5 Command Register
0x344	0x344	S5_IMR	S5_IMR0	Reserved
	0x345		S5_IMR1	SOCKET5 Interrupt Mask Register
0x346	0x346	S5_IR	S5_IR0	Reserved
	0x347		S5_IR1	SOCKET5 Interrupt Register
0x348	0x348	S5_SSR	S5_SSR0	Reserved
	0x349		S5_SSR1	SOCKET5 Socket Status Register
0x34A	0x34A	S5_PORTR	S5_PORTR0	SOCKET5 Source Port Register
	0x34B		S5_PORTR1	
0x34C	0x34C	S5_DHAR	S5_DHAR0	SOCKET5 Destination Hardware Address Register
	0x34D		S5_DHAR1	
0x34E	0x34E	S5_DHAR2	S5_DHAR2	
	0x34F		S5_DHAR3	
0x350	0x350	S5_DHAR4	S5_DHAR4	
	0x351		S5_DHAR5	
0x352	0x352	S5_DPORTR	S5_DPORTR0	SOCKET5 Destination Port Register
	0x353		S5_DPORTR1	
0x354	0x354	S5_DIPR	S5_DIPR0	SOCKET5 Destination IP Address Register
	0x355		S5_DIPR1	
0x356	0x356	S5_DIPR2	S5_DIPR2	
	0x357		S5_DIPR3	
0x358	0x358	S5_MSSR	S5_MSSR0	SOCKET5 Maximum Segment Size Register
	0x359		S5_MSSR1	
0x35A	0x35A	S5_PORTOR	S5_KPALVTR	SOCKET5 Keep Alive Time Register
	0x35B		S5_PROTOR	SOCKET5 Protocol Number Register
0x35C	0x35C	S5_TOSR	S5_TOSR0	Reserved
	0x35D		S5_TOSR1	SOCKET5 TOS Register
0x35E	0x35E	S5_TTLR	S5_TTLR0	Reserved
	0x35F		S5_TTLR1	SOCKET5 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x360	0x360	S5_TX_WRSR	S5_TX_WRSR0	Reserved
	0x361		S5_TX_WRSR1	SOCKET5 TX Write Size Register
0x362	0x362	S5_TX_WRSR2	S5_TX_WRSR2	
	0x363		S5_TX_WRSR3	
0x364	0x364	S5_TX_FSR	S5_TX_FSR0	Reserved
	0x365		S5_TX_FSR1	SOCKET5 TX Free Size Register
0x366	0x366	S5_TX_FSR2	S5_TX_FSR2	
	0x367		S5_TX_FSR3	
0x368	0x368	S5_RX_RSR	S5_RX_RSR0	Reserved
	0x369		S5_RX_RSR1	SOCKET5 RX Receive Size Register
0x36A	0x36A	S5_RX_RSR2	S5_RX_RSR2	
	0x36B		S5_RX_RSR3	
0x36C	0x36C	S5_FRAGR	S5_FRAGR0	Reserved
	0x36D		S5_FRAGR1	SOCKET5 IP FLAG Field Register
0x36E	0x36E	S5_TX_FIFOR	S5_TX_FIFOR0	SOCKET5 TX FIFO Register
	0x36F		S5_TX_FIFOR1	
0x370	0x370	S5_RX_FIFOR	S5_RX_FIFOR0	SOCKET5 RX FIFO Register
	0x371		S5_RX_FIFOR1	
0x372	0x372			Reserved
	0x373			
:				:
:				:
0x37E	0x37E			Reserved
	0x37F			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x380	0x380	S6_MR	S6_MR0	SOCKET6 Mode Register
	0x381		S6_MR1	
0x382	0x382	S6_CR	S6_CR0	Reserved
	0x383		S6_CR1	SOCKET6 Command Register
0x384	0x384	S6_IMR	S6_IMR0	Reserved
	0x385		S6_IMR1	SOCKET6 Interrupt Mask Register
0x386	0x386	S6_IR	S6_IR0	Reserved
	0x387		S6_IR1	SOCKET6 Interrupt Register
0x388	0x388	S6_SSR	S6_SSR0	Reserved
	0x389		S6_SSR1	SOCKET6 Socket Status Register
0x38A	0x38A	S6_PORTR	S6_PORTR0	SOCKET6 Source Port Register
	0x38B		S6_PORTR1	
0x38C	0x38C	S6_DHAR	S6_DHAR0	SOCKET6 Destination Hardware Address Register
	0x38D		S6_DHAR1	
0x38E	0x38E	S6_DHAR2	S6_DHAR2	
	0x38F		S6_DHAR3	
0x390	0x390	S6_DHAR4	S6_DHAR4	
	0x391		S6_DHAR5	
0x392	0x392	S6_DPORTR	S6_DPORTR0	SOCKET6 Destination Port Register
	0x393		S6_DPORTR1	
0x394	0x394	S6_DIPR	S6_DIPR0	SOCKET6 Destination IP Address Register
	0x395		S6_DIPR1	
0x396	0x396	S6_DIPR2	S6_DIPR2	
	0x397		S6_DIPR3	
0x398	0x398	S6_MSSR	S6_MSSR0	SOCKET6 Maximum Segment Size Register
	0x399		S6_MSSR1	
0x39A	0x39A	S6_PORTOR	S6_KPALVTR	SOCKET6 Keep Alive Time Register
	0x39B		S6_PROTOR	SOCKET6 Protocol Number Register
0x39C	0x39C	S6_TOSR	S6_TOSR0	Reserved
	0x39D		S6_TOSR1	SOCKET6 TOS Register
0x39E	0x39E	S6_TTLR	S6_TTLR0	Reserved
	0x39F		S6_TTLR1	SOCKET6 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x3A0	0x3A0	S6_TX_WRSR	S6_TX_WRSR0	Reserved
	0x3A1		S6_TX_WRSR1	SOCKET6 TX Write Size Register
0x3A2	0x3A2	S6_TX_WRSR2	S6_TX_WRSR2	
	0x3A3		S6_TX_WRSR3	
0x3A4	0x3A4	S6_TX_FSR	S6_TX_FSR0	Reserved
	0x3A5		S6_TX_FSR1	SOCKET6 TX Free Size Register
0x3A6	0x3A6	S6_TX_FSR2	S6_TX_FSR2	
	0x3A7		S6_TX_FSR3	
0x3A8	0x3A8	S6_RX_RSR	S6_RX_RSR0	Reserved
	0x3A9		S6_RX_RSR1	SOCKET6 RX Receive Size Register
0x3AA	0x3AA	S6_RX_RSR2	S6_RX_RSR2	
	0x3AB		S6_RX_RSR3	
0x3AC	0x3AC	S6_FRAGR	S6_FRAGR0	Reserved
	0x3AD		S6_FRAGR1	SOCKET6 IP FLAG Field Register
0x3AE	0x3AE	S6_TX_FIFOR	S6_TX_FIFOR0	SOCKET6 TX FIFO Register
	0x3AF		S6_TX_FIFOR1	
0x3B0	0x3B0	S6_RX_FIFOR	S6_RX_FIFOR0	SOCKET6 RX FIFO Register
	0x3B1		S6_RX_FIFOR1	
0x3B2	0x3B2			Reserved
	0x3B3			
:				:
:				:
0x3BE	0x3BE			Reserved
	0x3BF			

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x3C0	0x3C0	S7_MR	S7_MR0	SOCKET7 Mode Register
	0x3C1		S7_MR1	
0x3C2	0x3C2	S7_CR	S7_CR0	Reserved
	0x3C3		S7_CR1	SOCKET7 Command Register
0x3C4	0x3C4	S7_IMR	S7_IMR0	Reserved
	0x3C5		S7_IMR1	SOCKET7 Interrupt Mask Register
0x3C6	0x3C6	S7_IR	S7_IR0	Reserved
	0x3C7		S7_IR1	SOCKET7 Interrupt Register
0x3C8	0x3C8	S7_SSR	S7_SSR0	Reserved
	0x3C9		S7_SSR1	SOCKET7 Socket Status Register
0x3CA	0x3CA	S7_PORTR	S7_PORTR0	SOCKET7 Source Port Register
	0x3CB		S7_PORTR1	
0x3CC	0x3CC	S7_DHAR	S7_DHAR0	SOCKET7 Destination Hardware Address Register
	0x3CD		S7_DHAR1	
0x3CE	0x3CE	S7_DHAR2	S7_DHAR2	
	0x3CF		S7_DHAR3	
0x3D0	0x3D0	S7_DHAR4	S7_DHAR4	
	0x3D1		S7_DHAR5	
0x3D2	0x3D2	S7_DPORTR	S7_DPORTR0	SOCKET7 Destination Port Register
	0x3D3		S7_DPORTR1	
0x3D4	0x3D4	S7_DIPR	S7_DIPR0	SOCKET7 Destination IP Address Register
	0x3D5		S7_DIPR1	
0x3D6	0x3D6	S7_DIPR2	S7_DIPR2	
	0x3D7		S7_DIPR3	
0x3D8	0x3D8	S7_MSSR	S7_MSSR0	SOCKET7 Maximum Segment Size Register
	0x3D9		S7_MSSR1	
0x3DA	0x3DA	S7_PORTOR	S7_KPALVTR	SOCKET7 Keep Alive Time Register
	0x3DB		S7_PROTOR	SOCKET7 Protocol Number Register
0x3DC	0x3DC	S7_TOSR	S7_TOSR0	Reserved
	0x3DD		S7_TOSR1	SOCKET7 TOS Register
0x3DE	0x3DE	S7_TTLR	S7_TTLR0	Reserved
	0x3DF		S7_TTLR1	SOCKET7 TTL Register

Address offset		Symbol		Description
16Bit	8Bit	16Bit	8Bit	
0x3E0	0x3E0	S7_TX_WRSR	S7_TX_WRSR0	Reserved
	0x3E1		S7_TX_WRSR1	SOCKET7 TX Write Size Register
0x3E2	0x3E2	S7_TX_WRSR2	S7_TX_WRSR2	
	0x3E3		S7_TX_WRSR3	
0x3E4	0x3E4	S7_TX_FSR	S7_TX_FSR0	Reserved
	0x3E5		S7_TX_FSR1	SOCKET7 TX Free Size Register
0x3E6	0x3E6	S7_TX_FSR2	S7_TX_FSR2	
	0x3E7		S7_TX_FSR3	
0x3E8	0x3E8	S7_RX_RSR	S7_RX_RSR0	Reserved
	0x3E9		S7_RX_RSR1	SOCKET7 RX Receive Size Register
0x3EA	0x3EA	S7_RX_RSR2	S7_RX_RSR2	
	0x3EB		S7_RX_RSR3	
0x3EC	0x3EC	S7_FRAGR	S7_FRAGR0	Reserved
	0x3ED		S7_FRAGR1	SOCKET7 IP FLAG Field Register
0x3EE	0x3EE	S7_TX_FIFOR	S7_TX_FIFOR0	SOCKET7 TX FIFO Register
	0x3EF		S7_TX_FIFOR1	
0x3F0	0x3F0	S7_RX_FIFOR	S7_RX_FIFOR0	SOCKET7 RX FIFO Register
	0x3F1		S7_RX_FIFOR1	
0x3F2	0x3F2			Reserved
	0x3F3			
:				:
:				:
0x3FE	0x3FE			Reserved
	0x3FF			

4. Register Description

[Notation]

1. Symbol(Name)[R/W,RO,WO][AO1/AO2][Reset]

Symbol : Register symbol

Name : Register name

R/W : Read/Write

RO : Read Only

WO : Write Only

AO1 : Physical address of W5300 reg. in T.M.S (For Direct address mode)

AO2 : Address Offset of W5300 reg. in W.M.S (For Indirect address mode)

Reset : Reset value

편의상 T.M.S의 Base Address(BA)는 0x08000라 가정하고, 앞으로 설명될 W5300 register 의 Physical address는 0x08000을 BA로 한다.

2. Pn_ : Buffer ready PIN n("BRDYn") register prefix

Pn_BRDYR(BRDYn configure register, 0 <= n <= 3)

3. Sn_ : SOCKETn register prefix

Sn_MR (SOCKETn Mode Register, 0 <= n <= 7)

4.

Symbol of low address Reg.	Bit 15	14	13	12	11	10	9	8
Physical Address	Symbol	–	–	–	–	–	–	–
Address offset	Reset Value	1	0	0	X	U(R)	0	0
Symbol of high address Reg.	Bit 7	6	5	4	3	2	1	0
Physical Address	Symbol	–	–	–	–	–	–	–
Address offset	Reset Value	0	0	0	0	0	0	0

– : Reserved Bit 1 : Logical High 0 : Logical Low

X : Don't Care U : 1 or 0 (R) : Read Only Bit

16 bit Register Symbol(AO1/AO2)	
8bit Register Symbol (AO1/AO2)	8bit Register Symbol (AO1/AO2)
MSB(Value)	LSB(Value)

4.1 Mode Register

MR (Mode Register) [R/W] [0x08000/----][0x3800 or 0xB800]

MR은 전반적인 W5300 mode(Host I/F, Sn_TX_FIFOR & Sn_RX_FIFOR의 MSB/LSB swap, S/W reset, Internal TX/RX memory test, Data bus의 MSB/LSB swap, address mode 등)을 설정한다.

MR0	15	14	13	12	11	10	9	8
0x08000	DBW	MPF	WDF2	WDF1	WDF0	RDH	-	FS
----	U(R)	0(R)	1	1	1	0	0	0
MR1	7	6	5	4	3	2	1	0
0x08001	RST	-	MT	PB	PPPoE	DBS	-	IND
----	0	0	0	0	0	0	0	0

MR(15:8)/MR0(7:0)

Bit	Symbol	Description
15	DBW	**Data Bus Width** 0 : 8 bit Data Bus 1 : 16 bit Data Bus W5300의 Reset시, 이 bit는 PIN "BIT16EN"의 Logic level에 따라 결정되며, Reset 이후 변경되지 않는다. "1.1 PIN Layout"의 PIN "BIT16EN" 참조.
14	MPF	**MAC Layer Pause Frame** 0 : Normal frame 1 : Pause frame Router나 Switch 장비로부터 Pause frame을 수신할 경우 '1'로 설정된다. '1'로 설정되었을 경우, '0'으로 바뀔 때까지 모든 Data전송은 Pause 된다.
13	WDF2	**Write Data Fetch Time** Host-Write 동작 시, W5300은 '/CS'가 Low assert된 시점부터 WRF X PLL_CLK 후에 Write-Data를 Fetch한다. WRF X PLL_CLK 이전에 Host-Write operation이 끝날 경우('/CS'이 High로 de-assert된 경우)는, '/CS'가 High로 de-assert된 그 시점에서 Write-Data를 Fetch한다.
12	WDF1	
11	WDF0	

10	RDH	**Read Data Hold Time** 0 : No use data hold time 1 : Use data hold time (2 X PLL_CLK) Host-Read 동작 시, W5300은 Host-Read operation이 끝난 후('/CS'가 High로 de-assert 된 후)에도 2 X PLL_CLK 동안 Read-Data를 Hold한다. 이 경우 Data bus 충돌이 발생할 수 있기 때문에 주의해서 사용한다.
9	-	Reserved
8	FS	**FIFO Swap Bit** 0 : Disable swap 1 : Enable swap Sn_TX_FIFOR/Sn_RX_FIFOR의 Most significant byte(MSB)와 Least significant byte(LSB)를 서로 Swap한다. 기본적으로 W5300의 Byte ordering은 Big-endian이다. 만약 Target host system이 Little-endian이라면, 이 Bit을 '1'로 설정하여 Sn_TX_FIFOR/Sn_RX_FIFOR의 Byte ordering을 바꾸어 마치 Little-endian처럼 사용할 수 있다.

MR(7:0)/MR1(7:0)

Bit	Symbol	Description
7	RST	**S/W Reset** '1'이면, W5300을 Reset시킨다. 이 Bit는 Reset이후 자동으로 Clear된다.
6	-	Reserved
5	MT	**Memory Test Bit** 0 : Disable internal TX/RX memory test 1 : Enable internal TX/RX memory test 기본적으로 W5300의 Internal TX memory는 Sn_TX_FIFOR을 통한 Host-Write operation만 지원하고, Internal RX memory는 Sn_RX_FIFOR을 통한 Host-Read operation만 지원한다. 그러나, 이 Bit를 '1'로 설정할 경우, Internal TX/RX memory는 Sn_TX_FIFOR/Sn_RX_FIFOR을 통해 Host Read/Write operation을 모두 지원하여, Internal TX/RX memory을 검증할 수 있다. W5300의 Internal TX/RX memory test 이후, 반드시 Reset이나 해당 SOCKET을 Close해야 한다. 자세한 내용은 *"How to test internal TX/RX memory"*를 참조하라.

4	PB	**Ping Block Mode** 0 : Disable Ping Block 1 : Enable Ping Block 이 Bit이 '1'로 설정될 경우, W5300 ICMP logic block의 Auto-ping-reply-process가 Disable되어 상대방의 Ping-request(ICMP echo request)에 대한 Ping-reply(ICMP echo reply)를 하지 않는다. cf> Ping block mode가 '0'이라 할지라도, User가 ICMP SOCKET (Sn_MR (P3:P0)=Sn_MR_IPRAW and Sn_PROTOR1=0x01)을 사용(User가 ICMP Packet을 직접 처리)하고자 할 경우, Auto Ping-reply를 하지 않는다. Auto-ping-reply는 119Bytes까지만 지원한다.
3	PPPoE	**PPPoE Mode** 0 : Disable PPPoE mode 1 : Enable PPPoE mode Router나 기타 장비 없이 PPPoE server에 접속할 경우, 이 bit를 '1'로 설정한다. 자세한 내용은 *"How to use PPPoE in W5300"* 를 참조하라.
2	DBS	**Data Bus Swap** 앞서 설명한 FS bit가 Sn_TX_FIFOR/Sn_RX_FIFOR만 MSB와 LSB를 Swap하는 반면, 이 비트는 Sn_TX_FIFOR/Sn_RX_FIFOR를 포함한 모든 Register의 MSB와 LSB를 Swap한다. 단, 이 bit는 DBW bit가 '1'일 때만 적용된다.
1	-	Reserved
0	IND	**Indirect Bus I/F mode** 0 : Direct address mode 1 : Indirect address mode W5300의 Host interface mode를 설정한다.

4.2 Indirect Mode Registers

MR(IND) = '1' 일 경우 W5300은 Indirect address mode로 동작하게 된다. 이때 Target host system은 MR, IDM_AR, IDM_DR만을 사용하여(즉 MR, IDM_AR, IDM_DR만 T.M.S에 Mapping하여 이들 register만 Direct로 Access하여), COMMON registers와 SOCKET registers를 Access하게 된다.

IDM_AR(Indirect Mode Address Register) [R/W] [0x08002/----][0x0000]

Indirect로 Access할 COMMON registers나 SOCKET registers의 Address offset을 설정한다.
IDM_AR의 최하위 Bit인 IDM_AR(0) 혹은 IDM_AR1(0)은 무시된다.

Ex) S4_RX_FIFOR(0x330)를 Access할 경우 다음과 같다.

IDM_AR0 = S4_RX_FIFOR의 Address offset의 MSB (0x03)

IDM_AR1 = S4_RX_FIFOR의 Address offset의 LSB (0x30)

IDM_AR(0x08002/----)	
IDM_AR0(0x08002/----)	IDM_AR1(0x08003/----)
0x03	0x30

IDM_DR(Indirect Mode Data Register) [R/W] [0x08004/-----][0x0000]

IDM_AR로 지정된 COMMON registers나 SOCKET registers의 실제 Data 값을 Access한다.
IDM_AR에 지정된 Register의 MSB와 LSB값은 DM_DR0와 IDM_DR1으로 각각 대응된다.
8 bit data bus width를 사용하는 경우, 지정한 Register의 LSB값을 Access하고자 한다면, IDM_DR1을 사용하고, MSB값을 Access하고자 한다면 IDM_DR0을 사용한다.

Ex1) IR(0x002)에 0x80F0 값을 Host-Write하는 동작

16 bit data bus width (MR(DBW) = '1')	8 bit data bus width (MR(DBW) = '0')
IDM_AR = 0x0002 IDM_DR = 0x80F0	IDM_AR0 = 0x00 IDM_AR1 = 0x02 IDM_DR0 = 0x80 IDM_DR1 = 0xF0

Ex2) IR(0x0FE)의 값을 Host-Read해서 변수 'val'에 저장하는 동작

16 bit data bus width (MR(DBW) = '1')	8 bit data bus width (MR(DBW) = '0')
IDM_AR = 0x0002 val = IDM_DR	IDM_AR0 = 0x00 IDM_AR1 = 0x02 val = IDM_DR0 val = (val << 8) + IDM_DR1

IDM_AR(0x08002/----)	
IDM_AR0(0x08002/----)	IDM_AR1(0x08003/----)
0x00	0x02

IDM_DR(0x08004/----)	
IDM_DR0(0x08004/----)	IDM_DR1(0x08005/----)
IR의 MSB(IR0)	IR의 LSB(IR1)

4.3 COMMON Registers

IR (Interrupt Register) [R/W] [0x08002/0x002] [0x0000]

IR은 Host에게 W5300에서 발생한 Interrupt 종류를 알려주기 위한 Register이다.

Interrupt 발생시, IR의 해당 Interrupt bit가 '1'로 설정되고 IMR의 해당 Interrupt mask bit이 '1'일 경우, '/INT' signal은 Low로 Assert된다.

'/INT' signal은 IR의 모든 Bit가 '0'이 될 때까지 Low를 유지하며, IR의 모든 Bit가 '0'이 되었다면 High로 De-assert된다. '1'로 설정된 IR0의 Bit를 Clear하기 위해서는 그 Bit를 '1'로 Host-Write 해야 한다. '1'로 설정된 IR1의 Bit는 그 Bit에 해당하는 Sn_IR를 Clear할 경우 자동으로 Clear된다.

IR0	15	14	13	12	11	10	9	8
0x08002	IPCF	DPUR	PPPT	FMTU	–	–	–	–
0x002	0	0	0	0	0	0	0	0
IR1	7	6	5	4	3	2	1	0
0x08003	S7_INT	S6_INT	S5_INT	S4_INT	S3_INT	S2_INT	S1_INT	S0_INT
0x003	0	0	0	0	0	0	0	0

IR(15:8)/IR0(7:0)

Bit	Symbol	Description
15	IPCF	**IP Conflict** W5300의 IP address가 충돌할 경우(Source IP address와 동일한 IP address를 갖는 ARP-request packet을 수신할 경우) '1'로 설정된다. '1'로 설정된 경우, Network상에 동일 IP address를 사용하는 Network 장비가 있음을 의미하고, 이는 통신장애의 원인이 되므로 이를 해결해야 한다.
14	DPUR	**Destination Port unreachable** W5300은 ICMP(Destination port unreachable) packet을 수신 할 경우 '1'로 설정된다. UIPR과 UPORTR를 참조하라.
13	PPPT	**PPPoE Terminate** PPPoE mode에서, PPPoE server와 Connection이 Close되었을 때 '1'로 설정된다.
12	FMTU	**Fragment MTU**

		ICMP(Fragment MTU) packet을 수신할 경우 '1'로 설정된다. FMTUR 참조를 참조하라.
11	-	Reserved
10	-	Reserved
9	-	Reserved
8	-	Reserved

IR(7:0)/IR1(7:0)

Bit	Symbol	Description
7	S7_INT	**Occurrence of SOCKET7 Interrupt** SOCKET7에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S7_IR1에 반영되며, S7_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
6	S6_INT	**Occurrence of SOCKET6 Interrupt** SOCKET6에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S6_IR1에 반영되며, S6_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
5	S5_INT	**Occurrence of SOCKET5 Interrupt** SOCKET5에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S5_IR1에 반영되며, S5_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
4	S4_INT	**Occurrence of SOCKET4 Interrupt** SOCKET4에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S4_IR1에 반영되며, S4_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
3	S3_INT	**Occurrence of SOCKET3 Interrupt** SOCKET3에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S3_IR1에 반영되며, S3_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
2	S2_INT	**Occurrence of SOCKET2 Interrupt**

		SOCKET2에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S2_IR1에 반영되며, S2_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
1	S1_INT	Occurrence of SOCKET1 Interrupt SOCKET1에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S1_IR1에 반영되며, S1_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.
0	S0_INT	Occurrence of SOCKET0 Interrupt SOCKET0에서 Interrupt가 발생할 경우 '1'로 설정된다. 이때 발생한 Interrupt 정보는 S0_IR1에 반영되며, S0_IR1이 Host에 의해 0x00으로 Clear될 경우 이 Bit는 자동으로 Clear된다.

IMR (Interrupt Mask Register) [R/W] [0x08004/0x004] [0x0000]

Host로 알려줄 W5300의 Interrupt를 설정한다. IMR의 Interrupt mask bit들은 IR의 Interrupt bit들과 각각 대응되며, IR의 임의의 bit가 '1'로 설정되고 IMR의 대응 Bit가 '1'로 설정되었을 경우, Host에게 Interrupt가 Issue('/INT' signal은 High에서 Low로 Assert)된다.

만약 IMR의 대응 bit가 '0'으로 설정되었다면, IR의 그 bit가 '1'로 설정되었다 할지라도, Host에게 Interrupt는 Issue('/INT' pin이 High를 계속 유지)되지 않는다.

IMR0	15	14	13	12	11	10	9	8
0x08004	IPCF	DPUR	PPPT	FMTU	–	–	–	–
0x004	0	0	0	0	0	0	0	0
IMR1	7	6	5	4	3	2	1	0
0x08005	S7_INT	S6_INT	S5_INT	S4_INT	S3_INT	S2_INT	S1_INT	S0_INT
0x005	0	0	0	0	0	0	0	0

IMR(15:8)/IMR0(7:0)

Bit	Symbol	Description
15	IPCF	IR(IPCF) Interrupt Mask
14	DPUR	IR(DPUR) Interrupt Mask
13	PPPT	IR(PPPT) Interrupt Mask
12	FMTU	IR(FMTU) Interrupt Mask
11	–	Reserved

10	-	Reserved
9	-	Reserved
8	-	Reserved

IMR(7:0)/IMR1(7:0)

Bit	Symbol	Description
7	S7_INT	IR(S7_INT) Interrupt Mask
6	S6_INT	IR(S6_INT) Interrupt Mask
5	S5_INT	IR(S5_INT) Interrupt Mask
4	S4_INT	IR(S4_INT) Interrupt Mask
3	S3_INT	IR(S3_INT) Interrupt Mask
2	S2_INT	IR(S2_INT) Interrupt Mask
1	S1_INT	IR(S1_INT) Interrupt Mask
0	S0_INT	IR(S0_INT) Interrupt Mask

SHAR (Source Hardware Address Register) [R/W] [0x08008/0x008] [00.00.00.00.00.00]

Source hardware address(MAC address)를 설정한다.

Ex) SHAR = “00.08.DC.01.02.03”

SHAR(0x08008/0x008)	
SHAR0(0x08008/0x008)	SHAR1(0x08009/0x009)
0x00	0x08
SHAR2(0x0800A/0x00A)	
SHAR2(0x0800A/0x00A)	SHAR3(0x0800B/0x00B)
0xDC	0x01
SHAR4(0x0800C/0x00C)	
SHAR4(0x0800C/0x00C)	SHAR5(0x0800D/0x00D)
0x02	0x03

GAR (Gateway IP Address Register) [R/W] [0x08010/0x010] [00.00.00.00]

Gateway IP address를 설정한다.

Ex) GAR = “192.168.0.1”

GAR(0x08010/0x010)		GAR2(0x08012/0x012)	
GAR0(0x08010/0x010)	GAR1(0x08011/0x011)	GAR2(0x08012/0x012)	GAR3(0x08013/0x013)
192(0xC0)	168(0xA8)	0(0x00)	1(0x01)

SUBR (Subnet Mask Register) [R/W] [0x08014/0x014] [00.00.00.00]

Subnet mask address를 설정한다.

Ex) SUBR = “255.255.255.0”

SUBR(0x08014/0x014)		SUBR2(0x08016/0x016)	
SUBR0(0x08014/0x014)	SUBR1(0x08015/0x015)	SUBR2(0x08016/0x016)	SUBR3(0x08017/0x017)
255 (0xFF)	255 (0xFF)	255 (0xFF)	0 (0x00)

SIPR (Source IP Address Register) [R/W] [0x08018/0x018] [00.00.00.00]

Source IP address를 설정하거나, W5300 내의 PPPoE-process를 통해 설정된 Source IP address를 알려준다.

Ex) SIPR = “192.168.0.3”

SIPR(0x08018/0x018)		SIPR2(0x0801A/0x01A)	
SIPR0(0x08018/0x018)	SIPR1(0x08019/0x019)	SIPR2(0x0801A/0x01A)	SIPR3(0x0801B/0x01B)
192(0xC0)	168(0xA8)	0(0x00)	3(0x03)

RTR (Retransmission Timeout-period Register) [R/W] [0x0801C/0x01C] [0x07D0]

Retransmission timeout-period(Data 재전송 시간)를 설정한다. RTR의 기본의 단위는 100us이고, Reset시 2000(0x07D0)으로 초기화되어 200ms의 Timeout-period를 갖는다.

Timeout-period = RTR X 0.1ms

RTR = (Timeout-period / 1ms) X 10

Ex) Timeout-period 400ms 설정, RTR = (400ms / 1ms) X 10 = 4000(0x0FA0)

RTR(0x0801C/0x01C)	
RTR0(0x0801C/0x01C)	RTR1(0x0801D/0x01D)
0x0F	0xA0

RCR (Retransmission Retry-Count Register) [R/W] [0x0801E/0x001E] [0x--08]

Retransmission count(Data 재전송 회수)를 설정한다. ‘RCR + 1’ 개의 Retransmission이 발생할 경우, Timeout interrupt(Sn_IR의 ‘TO’ bit가 ‘1’로 설정)된다.

TCP 통신인 경우, Sn_IR(TIMEOUT)= ‘1’과 동시에 Sn_SSR의 값이 ‘SOCK_CLOSED’로 변경된다.

TCP 통신이 아닌 경우, Sn_IR(TIMEOUT) = '1'만 된다.

Ex) RCR = 0x0007

RCR(0x0801E/0x01E)	
RCR0(0x0801E/0x01C)	RCR1(0x0801F/0x01F)
Reserved	0x07

W5300에서의 Timeout은 RTR과 RCR로 Data 재전송의 시간과 횟수를 설정할 수 있다.

W5300의 Timeout에 대해 좀더 살펴 보면,

ARP retransmission timeout과 TCP retransmission timeout 2가지가 있다.

먼저 ARP("RFC 826" 참조, http://www.ietf.org/rfc.html) retransmission timeout 살펴보면, W5300은 IP, UDP, TCP를 이용한 통신시 상대방의 IP address로 MAC address를 알기 위해 자동으로 ARP-request를 전송한다. 이때 상대방의 ARP-response 수신을 기다리는데, RTR의 설정 대기 시간 동안 ARP-response의 수신이 없으면, Timeout이 발생하고 ARP-request를 Retransmission한다. 이와 같은 작업은 'RCR + 1'만큼 반복하게 된다.

'RCR + 1'개의 ARP-request retransmission이 일어나고, 그에 대한 ARP-response가 없다면, Final timeout이 발생하게 되고, Sn_IR(TIMEOUT) = '1' 된다.

ARP-request의 Final timeout(ARP_{TO}) 값은 다음과 같다.

$$ARP_{TO} = (RTR \times 0.1ms) \times (RCR + 1)$$

TCP packet retransmission timeout을 살펴보면, W5300은 TCP packet (SYN, FIN, RST, DATA packet)을 전송하고 그에 대한 Acknowledgment(ACK)을 RTR과 RCR에 의해 설정된 대기 시간 동안 기다리게 된다. 이때 상대방으로부터 ACK가 없으면 Timeout이 발생하고 이전에 보냈던 TCP packet을 Retransmission한다. 이와 같은 작업은 'RCR + 1'만큼 반복하게 된다. 'RCR + 1'개의 TCP packet retransmission이 일어나고, 그에 대한 ACK 수신이 없다면, Final timeout이 발생하게 되고, Sn_IR(TIMEOUT) = '1'과 동시에 Sn_SSR이 'SOCK_CLOSED"로 변경된다. TCP packet retransmission의 Final timeout(TCP_{TO}) 값은 다음과 같다.

$$TCP_{TO} = \left(\sum_{N=0}^{M} (RTR \times 2^N) + ((RCR-M) \times RTR_{MAX}) \right) \times 0.1ms$$

N : Retransmission count, 0 <= N <= M

M : RTR X $2^{(M+1)}$ > 65535 and 0 <= M <= RCR를 만족하는 최소값

RTR_{MAX} : RTR X 2^M

Ex) RTR = 2000(0x07D0), RCR = 8(0x0008)일 때,

ARP_{TO} = 2000 X 0.1ms X 9 = 1800ms = 1.8s

TCP_{TO} = (0x07D0 + 0x0FA0 + 0x1F40 + 0x3E80 + 0x7D00 + 0xFA00 + 0xFA00 + 0xFA00 + 0xFA00) X 0.1ms

= (2000 + 4000 + 8000 + 16000 + 32000 + ((8 – 4) X 64000)) X 0.1ms

= 318000 X 0.1ms = 31.8s

TMSR(TX Memory Size Register) [R/W] [0x08020/0x020] [08.08.08.08.08.08.08.08]

각 SOCKET의 Internal TX memory size를 1Kbytes 단위로 설정한다.

각 SOCKET의 TX memory size는 0Kbyte에서 64Kbytes내에서 설정이 가능하며, Reset시 8Kbytes로 설정된다. 각 SOCKET의 TX memory size의 총합(TMS_{SUM})은 반드시 8의 배수가 되도록 설정하며, 또한 TMS_{SUM}과 각 SOCKET의 RX memory size의 총합(RMS_{SUM})의 합이 128Kbytes가 되도록 설정한다.

TMS01R(TX Memory Size of SOCKET0/1 Register) [R/W] [0x08020/0x020] [0x0808]

SOCKET0과 SOCKET1의 Internal TX memory size를 결정한다.

Ex1) SOCKET0 : 4KB, SOCKET1 : 16KB

TMS01R(0x08020/0x020)	
TMSR0(0x08020/0x020)	TMSR1(0x08021/0x021)
4 (0x04)	16 (0x10)

TMS23R(TX Memory Size of SOCKET2/3 Register) [R/W] [0x08022/0x022] [0x0808]

SOCKET2과 SOCKET3의 Internal TX memory size를 결정한다.

Ex2) SOCKET2 : 1KB, SOCKET3 : 20KB

TMS23R(0x08020/0x020)	
TMSR2(0x08022/0x022)	TMSR3(0x08023/0x023)
1 (0x01)	20 (0x14)

TMS45R(TX Memory Size of SOCKET4/5 Register) [R/W] [0x08024/0x024] [0x0808]

SOCKET4과 SOCKET5의 Internal TX memory size를 결정한다.

Ex3) SOCKET4 : 0KB, SOCKET5 : 7KB

TMS45R(0x08024/0x024)	
TMSR4(0x08024/0x024)	TMSR5(0x08025/0x025)
0 (0x00)	7 (0x07)

TMS67R(TX Memory Size of SOCKET6/7 Register) [R/W] [0x08024/0x024] [0x0808]

SOCKET6과 SOCKET7의 Internal TX memory size를 결정한다.

Ex4) SOCKET6 : 12KB, SOCKET7 : 12KB

TMS67R(0x08026/0x026)	
TMSR6(0x08026/0x026)	TMSR7(0x08027/0x027)
12 (0x0C)	12 (0x0C)

상기 Ex1)~Ex4) 에서, TMS_{SUM}(TMSR0 + TMSR1 + TMSR2 + TMSR3 + TMSR4 + TMSR5 + TMSR6 + TMSR7)은 72로, 8의 배수로 설정되었다(72 % 8 = 0).

RMSR(RX Memory Size Register) [R/W] [0x08028/0x028] [08.08.08.08.08.08.08.08]

각 SOCKET의 Internal RX memory size를 1Kbytes 단위로 설정한다.

각 SOCKET의 RX memory size는 0Kbyte에서 64Kbytes내에서 설정이 가능하며, Reset시 8Kbytes로 설정된다. 각 SOCKET의 RMS_{SUM}은 TMS_{SUM}과 각 RMS_{SUM}의 합이 128KB가 되도록 설정하여야 한다.

RMS01R(RX Memory Size of SOCKET0/1 Register) [R/W] [0x08028/0x028] [0x0808]

SOCKET0과 SOCKET1의 Internal RX memory size를 결정한다.

Ex5) SOCKET0 : 17KB, SOCKET1 : 3KB

RMS01R(0x08028/0x028)	
RMSR0(0x08028/0x028)	RMSR1(0x08029/0x029)
17 (0x11)	3 (0x03)

RMS23R(RX Memory Size of SOCKET2/3 Register) [R/W] [0x0802A/0x02A] [0x0808]

SOCKET2과 SOCKET3의 Internal RX memory size를 결정한다.

Ex6) SOCKET2 : 5KB, SOCKET3 : 16KB

RMS23R(0x0802A/0x02A)	
RMSR2(0x0802A/0x02A)	RMSR3(0x0802B/0x02B)
5 (0x05)	16 (0x10)

RMS45R(RX Memory Size of SOCKET4/5 Register) [R/W] [0x0802C/0x02C] [0x0808]

SOCKET4과 SOCKET5의 Internal RX memory size를 결정한다.

Ex7) SOCKET4 : 3KB, SOCKET5 : 4KB

RMS45R(0x0802C/0x02C)	
RMSR4(0x0802C/0x02C)	RMSR5(0x0802D/0x02D)
3 (0x03)	4 (0x04)

RMS67R(TX Memory Size of SOCKET6/7 Register) [R/W] [0x0802E/0x02F] [0x0808]

SOCKET6과 SOCKET7의 Internal RX memory size를 결정한다.

Ex8) SOCKET6 : 4KB, SOCKET7 : 4KB

RMS67R(0x0802E/0x02E)	
RMSR6(0x0802E/0x02E)	RMSR7(0x0802F/0x02F)
4 (0x04)	4 (0x04)

상기 Ex1)~Ex8) 에서, RMS_{SUM}(RMSR0 + RMSR1 + RMSR2 + RMSR3 + RMSR4 + RMSR5 + RMSR6 + RMSR7)은 56로 설정되었다. 또한 TMS_{SUM} 와 RMS_{SUM}의 합은 128이다.

MTYPER(Memory Type Register) [R/W] [0x08030/0x030] [0x00FF]

W5300의 128Kbytes data memory(Internal TX/RX memory)는 8Kbytes의 Memory block 16개로 구성된다. MTYPER은 8KB의 Memory block들이 TX memory로 사용될지, RX memory로 사용될지를 설정한다. 8KB memory block의 Type은 MTYPER의 각 Bit로 대응되며 그 값이 '1'인 경우 TX memory, '0'인 경우 RX memory로 사용된다. MTYPER는 반드시 하위 Bit부터 TX memory type으로 설정하며, TX memory로 설정하지 않은 나머지 Bit는 '0'으로 설정한다.

MTYPER0	15	14	13	12	11	10	9	8
0x08030	MB15	MB14	MB13	MB12	MB11	MB10	MB9	MB8
0x030	0	0	0	0	0	0	0	0
MTYPER1	7	6	5	4	3	2	1	0
0x08031	MB7	MB6	MB5	MB4	MB3	MB2	MB1	MB0
0x031	1	1	1	1	1	1	1	1

MTYPER(15:8)/MTYPER0(7:0)

Bit	Symbol	Description
15	MB15	16th Memory Block Type
14	MB14	15th Memory Block Type
13	MB13	14th Memory Block Type
12	MB12	13th Memory Block Type
11	MB11	12th Memory Block Type
10	MB10	11th Memory Block Type
9	MB9	10th Memory Block Type
8	MB8	9th Memory Block Type

MTYPER(7:0)/MTYPER1(7:0)

Bit	Symbol	Description
7	MB7	8th Memory Block Type
6	MB6	7th Memory Block Type
5	MB5	6th Memory Block Type
4	MB4	5th Memory Block Type
3	MB3	4th Memory Block Type
2	MB2	3rd Memory Block Type
1	MB1	2nd Memory Block Type
0	MB0	1st Memory Block Type

Ex1) TMS_{SUM} = 72, RMS_{SUM} = 56

72 / 8 = 9 이므로 MB0부터 MB8까지 TX Memory로 설정한다.

MTYPER(0x08030/0x030)	
MTYPER0(0x08030/0x030)	MTYPER1(0x08031/0x031)
0x01	0xFF

Ex2) TMS_{SUM} = 128, RMS_{SUM} = 0

MTYPER(0x08030/0x030)	
MTYPER0(0x08030/0x030)	MTYPER1(0x08031/0x031)
0xFF	0xFF

Ex3) TMS_{SUM} = 0, RMS_{SUM} = 128

MTYPER(0x08030/0x030)	
MTYPER0(0x08030/0x030)	MTYPER1(0x08031/0x031)
0x00	0x00

PATR (PPPoE Authentication Type Register) [R] [0x08032/0x032] [0x0000]

PPPoE server와의 통신에서 협의된 Authentication method을 알려준다.

W5300 2가지의 Authentication method를 지원한다.

Value	Authentication method
0xC023	PAP
0xC223	CHAP

Ex) PATR = 'CHAP'

PATR(0x08032/0x032)	
PATR0(0x08032/0x032)	PATR1(0x08033/0x033)
0xC2	0x23

PTIMER(PPP Link Control Protocol Request Timer Register)[R/W][0x08036/0x036][0x--28]

Link control protocol(LCP) echo request의 전송 Timer를 설정한다.

Value 1은 약 25ms에 해당한다.

Ex) PTIMER = 200 (200 * 25ms = 5000ms = 5s)

PTIMER(0x08036/0x037)	
PTIMER0(0x08036/0x036)	PTIMER1(0x08037/0x037)
Reserved	200 (0xC8)

PMAGICR(PPP LCP Magic number Register)[R/W][0x08038/0x038][0x--00]

PPPoE server와 LCP negotiation 동안 사용하게 될 4bytes "Magic number"로 사용될 Byte 값을 설정한다. *"How to use PPPoE in W5300"* 문서를 참조하라.

Ex) PMAGICR = 0x01

PMAGICR(0x08036/0x037)	
PMAGICR0(0x08038/0x038)	PMAGICR1(0x08039/0x039)
Reserved	0x01

Magic number = 0x01010101

PSIDR(PPPoE Session ID Register)[R][0x0803C/0x03C][0x0000]

W5300 내의 PPPoE-process를 통해 획득한 PPPoE server와의 통신에서 사용하게 될 PPP session ID를 알려준다.

Ex) PSIDR = 0x0017

PSIDR(0x0803C/0x03C)	
PSIDR0(0x0803C/0x03C)	PSIDR1(0x0803D/0x03D)
0x00	0x17

PDHAR(PPPoE Destination Hardware Address Register)[R][0x08040/0x040] [00.00.00.00.00.00]

W5300 내의 PPPoE-process를 통해 획득한 PPPoE server의 Hardware address를 알려준다.

Ex) PDHAR = 00.01.02.03.04.05

PDHAR(0x08040/0x040)	
PDHAR0(0x08040/0x040)	PDHAR1(0x08041/0x041)
0x00	0x01
PDHAR2(0x08042/0x042)	
PDHAR2(0x08042/0x042)	PDHAR3(0x08043/0x043)
0x02	0x03
PDHAR4(0x08044/0x044)	
PDHAR4(0x08044/0x044)	PDHAR5(0x08045/0x045)
0x04	0x05

UIPR (Unreachable IP Address Register) [R] [0x08048/0x048] [00.00.00.00]

UPORTR (Unreachable Port Register) [R] [0x0804C/0x04C] [0x0000]

열려있지 않는 Destination port number로 UDP data 전송을 시도할 때, W5300은 ICMP(Destination port unreachable) packet를 수신할 수 있다.

이 경우 IR(DPUR) = '1'이 되고, 수신된 ICMP packet의 Destinaton IP address와 Unreachable port number는 각각 UIPR와 UPORTR을 통해 알 수 있다.

Ex1) UIPR = 192.168.0.11

UIPR(0x08048/0x048)		UIPR2(0x0804A/0x04A)	
UIPR0(0x08048/0x048)	UIPR1(0x08049/0x049)	UIPR2(0x0804A/0x04A)	UIPR3(0x0804B/0x04B)
192 (0xC0)	168 (0xA8)	0 (0x00)	11 (0x0B)

Ex2) UPORTR = 5000(0x1388)

UPORTR(0x0804C/0x04C)	
UPORTR0(0x0804C/0x04C)	UPORTR1(0x0804D/0x04D)
0x13	0x18

FMTUR (Fragment MTU Register) [R] [0x0804E/0x04E] [0x0000]

MTU가 서로 다른 상대방과 통신을 시도할 경우, W5300은 ICMP(Fragment MTU) packet을 수신할 수 있다. 이 경우 IR(FMTU)='1'이 되며, 수신된 ICMP packet의 Destination IP address와 Fragment MTU값은 각각 UIPR과 FMTUR을 통해 알 수 있다. Fragment MTU가 발생한 상대방과 통신을 계속 시도할 경우, 그 FMTUR 값을 해당 통신 SOCKET의 Sn_MSSR에 설정한 후 다시 통신을 시도해야 한다.

Ex) FMTUR = 512(0x200)

FMTUR(0x0804E/0x04E)	
FMTUR0(0x0804E/0x04E)	FMTUR1(0x0804F/0x04F)
0x02	0x00

Pn_BRDYR (PIN "BRDYn" Configure Register) [R/W] [0x08060+4n/0x060+4n] [0x--00]

SOCKET의 TX/RX memory 상태를 Monitoring하는 PIN "BRDYn"을 설정한다.

Pn_BRDYR의 설정에 따라, SOCKET의 TX memory의 Free buffer size가 Pn_BDPTHR에 설정된 Buffer depth보다 같거나 클 경우, 혹은 RX memory의 Received buffer size가 Pn_BDPTHR보다 같거나 클 경우에 PIN "BRDYn"은 signal된다.

Pn_BRDYR0	15	14	13	12	11	10	9	8
0x08060 + 4n	–	–	–	–	–	–	–	–
0x060 + 4n	0	0	0	0	0	0	0	0
Pn_BRDYR1	7	6	5	4	3	2	1	0
0x08061	PEN	PMT	PPL	–	–	SN2	SN1	SN0
0x061	0	0	1	0	0	0	0	0

Pn_BRDYR(7:0)/Pn_BRDYR1(7:0)

Bit	Symbol	Description
7	PEN	**PIN "BRDYn" Enable** 0 : Disable BRDYn 1 : Enable BRDYn PIN "BRDYn"을 사용하고자 할 경우 '1'로 설정한다.
6	PMT	**PIN Memory Type** 0 : RX memory 1 : TX memory Monitoring할 SOCKET의 Memory를 설정한다.
5	PPL	**PIN Polarity** 0 : Low sensitive 1 : High sensitive TX/RX memory의 Free/Received buffer size가 Pn_DPTHR보다 같거나 클 경우, Host에게 Signal 될 PIN "BRDYn"의 Logic level을 설정한다.
4	–	Reserved
3	–	Reserved
2	SN2	**SOCKET Number** PIN "BRDYn"으로 Monitoring할 SOCKET Number를 설정한다.
1	SN1	
0	SN0	

	SN2	SN1	SN0		SN2	SN1	SN0
7	1	1	1	3	0	1	1
6	1	1	0	2	0	1	0
5	1	0	1	1	0	0	1
4	1	0	0	0	0	0	0

P0_BRDYR (PIN "BRDY0" Configure Register) [R/W] [0x08060/0x060] [0x--00]
PIN "BRDY0"을 설정한다.

P1_BRDYR (PIN "BRDY1" Configure Register) [R/W] [0x08064/0x064] [0x--00]
PIN "BRDY1"을 설정한다.

P2_BRDYR (PIN "BRDY2" Configure Register) [R/W] [0x08068/0x068] [0x--00]
PIN "BRDY2"을 설정한다.

P3_BRDYR (PIN "BRDY3" Configure Register) [R/W] [0x0806C/0x06C] [0x--00]
PIN "BRDY3"을 설정한다.

Pn_BDPTHR (PIN "BRDYn" Buffer Depth Register) [R/W] [0x08062/0x062] [0xUUUU]

PIN "BRDYn"의 Buffer depth를 설정한다. TX memory를 Monitoring할 경우, Sn_TX_FSR이 Pn_DPTHR보다 같거나 클 경우 PIN "BRDYn"은 signal된다. RX memory를 Monitoring할 경우, Sn_RX_RSR이 Pn_DPTHR보다 같거나 클 경우에 PIN "BRDYn"은 signal된다. Pn_BDPTHR은 TMSR이나 RMSR에 의해 설정된 SOCKET의 TX/RX memory 할당 크기를 초과하여 설정할 수 없다.

P0_BDPTHR (PIN "BRDY0" Buffer Depth Register) [R/W] [0x08062/0x062] [0xUUUU]
PIN "BRDY0"의 Buffer depth를 설정한다.

P1_BDPTHR (PIN "BRDY1" Buffer Depth Register) [R/W] [0x08066/0x066] [0xUUUU]
PIN "BRDY1"의 Buffer depth를 설정한다.

P2_BDPTHR (PIN "BRDY2" Buffer Depth Register) [R/W] [0x0806A/0x06A] [0xUUUU]
PIN "BRDY2"의 Buffer depth를 설정한다.

P3_BDPTHR (PIN "BRDY3" Buffer Depth Register) [R/W] [0x0806E/0x06E] [0xUUUU]
PIN "BRDY3"의 Buffer depth를 설정한다.

Ex) PIN "BRDY3"로 SOCKET5의 TX memory Free size가 2048 이상인지를 High sensitive로 Monitoring 할 경우,

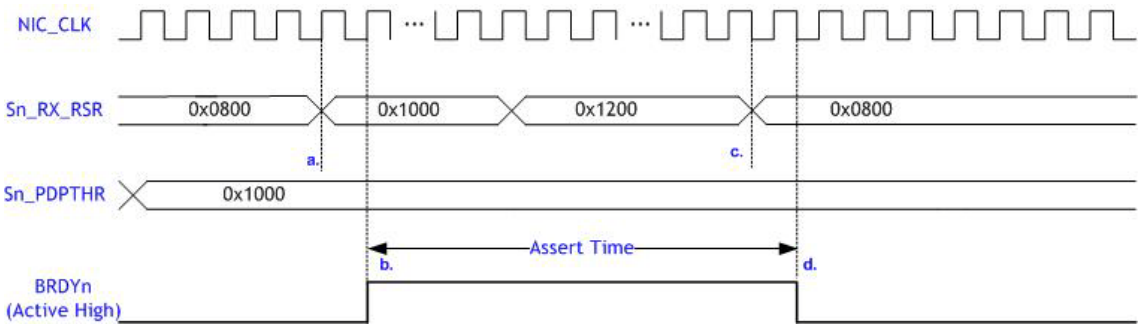
P3_BRDYR = 0x00E5

P3_BRDYR(0x0806C/0x06C)	
P3_BRDYR0(0x0806C/0x06C)	P3_BRDYR1(0x0806D/0x06D)
Reserved	0xE5

P3_BDPTHR = 2048(0x0800)

P3_BDPTHR(0x0806E/0x06E)	
P3_BDPTHR0(0x0806E/0x06E)	P3_BDPTHR1(0x0806F/0x06F)
0x08	0x00

다음은 SOCKETn의 RX memory를 Monitoring할 때의 PIN 'BRDYRn'의 Signal 변화를 보여 준다.



a. Sn_RX_RSR > Sn_BDPTHR 감지

b. 1 NIC_CLK 후, PIN 'BRDYn' High Assert

c. Host의 RX memory Read에 의해 Sn_RX_RSR가 감소, Sn_RX_RSR < Sn_BDPTHR 감지

d. 1 NIC_CLK 후, PIN 'BRDYn' Low De-assert

Assert Time : BRDYRn의 Active Time. 최소 80ns 이상 Sn_RX_RSR > Sn_BDPTHR인 동안 유지.

Fig 4. 'BRDYn' Timing

IDR (Identification Register) [R] [0x080FE/0x0FF] [0x5300]

W5300의 ID 값을 알려 준다.

IDR(0x080FE/0x0FE)	
IDR0(0x080FE/0x0FE)	IDR1(0x080FF/0x0FF)
0x53	0x00

4.4 SOCKET Registers

Sn_MR (SOCKETn Mode Register) [R/W] [0x08200+0x40n/0x200+0x40n] [0x0000]

SOCKETn의 Protocol type이나 Option을 설정한다.

Sn_MR0	15	14	13	12	11	10	9	8
0x08200 + 0x40n	–	–	–	–	–	–	–	ALIGN
0x200 + 0x40n	0	0	0	0	0	0	0	0
Sn_MR1	7	6	5	4	3	2	1	0
0x08201 + 0x40n	MULTI	–	ND/MC	–	P3	P2	P1	P0
0x201 + 0x40n	0	0	1	0	0	0	0	0

Sn_MR(15:8)/Sn_MR0(7:0)

Bit	Symbol	Description
15	–	Reserved
14	–	Reserved
13	–	Reserved
12	–	Reserved
11	–	Reserved
10	–	Reserved
9	–	Reserved
8	ALIGN	**Alignment** 0 : No use alignment 1 : Use alignment 이 Bit는 TCP(P3~P0 : “0001”)일 때만 유효하다. TCP 통신에 있어서, 모든 수신 Data의 크기가 짝수(Even)일 때 ‘1’로 설정하면, 매 수신 Data마다 붙는 PACKET-INFO(Data size)를 제거하여, Data 수신 성능을 향상시킬 수 있다. “5.2.1.1 TCP SERVER” 참조.

Sn_MR(7:0)/Sn_MR1(7:0)

Bit	Symbol	Description
7	MULTI	**Multicasting** 0 : Disable multicasting 1 : Enable multicastting

		이 Bit는 UDP(P3~03 : “0010”)일 경우에만 유효하다. Multicasting을 위해, Sn_CR의 “OPEN” command 이전에 Multicast-group의 IP address와 Port number를 Sn_DIPR과 Sn_DPORTR에 각각 설정한다.
6	MF	MAC Filter 0 : Disable MAC filter 1 : Enable MAC filter 이 Bit는 MACRAW(P3~P0 : “0100”)일 경우에만 유효하다. ‘1’로 설정될 경우, W5300은 Broadcasting packet이나 자신에게 전송되는 Packet만을 수신하게 된다. ‘0’으로 설정될 경우, W5300은 Ethernet 상의 모든 Packet을 수신하게 된다. Hybrid TCP/IP stack을 구현하고자 하는 경우, Host의 수신 Overhead를 감소시키기 위해 이 Bit를 ‘1’로 설정할 것을 권장한다.
5	ND/MC	Use No Delayed ACK 0 : Disable no delayed ACK option 1 : Enable no delayed ACK option 이 Bit는 TCP(P3~P0 : “0001”)일 때만 유효하다. ‘1’로 설정된 경우, ACK packet은 상대방으로부터 DATA packet을 수신할 때마다 즉시 ACK packet을 전송한다. 이 Bit는 TCP의 성능향상을 위해 ‘1’로 설정하는 것을 권장한다. ‘0’으로 설정된 경우, ACK packet은 상대방의 DATA packet 수신에 상관없이 RTR에 설정된 시간 이후 전송된다. Multicast 0 : using IGMP version 2 1 : using IGMP version 1 이 Bit는 MULTI=‘1’ 이고, UDP(P3~P0 : “0010”)일 때만 유효하다. Multicast-group에 Join/Leave/Report와 같은 IGMP message를 전송할 IGMP version을 설정한다.
4	-	Reserved

3	P3	Protocol 각 SOCKET별로 사용할 통신 Protocol(TCP, UDP, IP RAW, MAC RAW)을 설정하거나, PPPoE server와의 연동에 사용할 PPPoE SOCKET을 설정한다.
2	P2	
1	P1	
0	P0	S0_MR_MACRAW와 S0_MR_PPPoE는 오직 SOCKET0에서만 유효하다. S0_MR_PPPoE는 PPPoE server connection/termination을 위해 일시적으로 사용되는 것으로 연결 후 다른 Protocol로 활용될 수 있다.

Symbol	P3	P2	P1	P0	Meaning
Sn_MR_CLOSE	0	0	0	0	Closed
Sn_MR_TCP	0	0	0	1	TCP
Sn_MR_UDP	0	0	1	0	UDP
Sn_MR_IPRAW	0	0	1	1	IP RAW
S0_MR_MACRAW	0	1	0	0	MAC RAW
S0_MR_PPPoE	0	1	0	1	

Sn_CR (SOCKETn Command Register) [R/W] [0x08202+0x40n/0x202+0x40n] [0x--00]

SOCKETn에 대한 Open, Close, Connect, Listen, Send, Recv와 같은 Command를 설정한다. W5300이 그 Command를 인지하게 되면 Sn_CR은 자동으로 Clear된다. Sn_CR이 0x00으로 Clear되었다 할지라도, 해당 Command는 수행 중 일 수 있으며, Command의 완료는 Sn_IR이나 Sn_SSR 등을 통해 Check할 수 있다.

Sn_CR(0x08202+0x40n/0x202+0x40n)	
Sn_CR0(0x08202+0x40n/0x202+0x40n)	Sn_CR1(0x08203+0x40n/0x203+0x40n)
Reserved	Command

Sn_CR(7:0)/Sn_CR1(7:0)

<table>
<tr><th>Value</th><th>Command</th><th>Description</th></tr>
<tr><td>0x01</td><td>OPEN</td><td>SOCKETn을 초기화하고, Sn_MR(P3:P0)에서 설정한 Protocol에 따라 Open한다.
다음은 Sn_MR(P3:P0)에 따른 Sn_SSR 값의 변화이다.
<table>
<tr><th>Sn_MR(P3:P0)</th><th>Sn_SSR</th></tr>
<tr><td>Sn_MR_CLOSE</td><td>-</td></tr>
<tr><td>Sn_MR_TCP</td><td>SOCK_INIT</td></tr>
<tr><td>Sn_MR_UDP</td><td>SOCK_UDP</td></tr>
<tr><td>Sn_MR_IPRAW</td><td>SOCK_IPRAW</td></tr>
<tr><td>S0_MR_MACRAW</td><td>SOCK_MACRAW</td></tr>
<tr><td>S0_MR_PPPoE</td><td>SOCK_PPPoE</td></tr>
</table></td></tr>
<tr><td>0x02</td><td>LISTEN</td><td>TCP mode(Sn_MR(P3:P0)=Sn_MR_TCP)일 때만 유효하다.

SOCKETn을 “TCP SERVER”로 동작시킨다. 이것은 임의의 “TCP CLIENT”의 Connect-request(SYN packet)을 기다리기 위해 Sn_SSR을 SOCK_INIT에서 SOCK_LISTEN으로 변경시킨다.

Sn_SSR = SOCK_LISTEN이고 “TCP CLIENT”의 Connect-request를 성공적으로 처리했을 경우, Sn_IR(0)=‘1’로 되고 Sn_SSR은 SOCK_ESTABLISHED로 변경된다. Connect-request 처리를 실패했을 경우(SYN/ACK 전송 실패), TCP_{TO}가 발생하고(Sn_IR(3)= ‘1’), Sn_SSR은 SOCK_CLOSED로 변경된다.

cf> “TCP CLIENT”의 TCP connect-request port number가 존재하지 않을 경우, W5300은 RST packet을 전송하며, Sn_SSR은 변경되지 않는다.</td></tr>
<tr><td>0x04</td><td>CONNECT</td><td>TCP mode일 때만 유효하다.

SOCKETn을 “TCP CLIENT”로 동작시킨다.
이것은 Sn_DIPR와 Sn_DPORTR로 설정된 “TCP SERVER”에게 Connect-request(SYN packet)를 전송한다.

Connect-request가 성공했을 경우(SYN/ACK packet을 수신했을 경우), Sn_IR(0)=‘1’로 되고 Sn_SSR은 SOCK_ESTABLISHED로 변경된다.</td></tr>
</table>

		Connect-request가 실패했을 경우는 3가지가 있다. - ARP-process를 통해 Destination hardware address를 얻지 못하여 ARP_{TO}가 발생(Sn_IR(3)='1')한 경우 - SYN/ACK packet를 수신 못하고 TCP_{TO}가 발생(Sn_IR(3)= '1')한 경우 - SYN/ACK packet 대신 RST packet을 수신했을 경우. 이런 경우 Sn_SSR은 SOCK_CLOSED로 변경된다.
0x08	DISCON	TCP mode일 때만 유효하다. "TCP SERVER"와 "TCP CLIENT"에 상관없이, 접속중인 상대방에게 Disconnect-request(FIN packet)를 전송하거나(Active close), 상대방으로부터 Disconnect-request(FIN packet)을 수신했을 때(Passive close), FIN packet을 전송한다(Disconnect-process). Disconnect-request가 성공했다면(FIN/ACK packet을 수신했을 경우), Sn_SSR은 SOCK_CLOSED로 변경된다. Disconnect-request가 실패했다면, TCP_{TO}가 발생(Sn_IR(3)= '1')하고 Sn_SSR은 SOCK_CLOSED로 변경된다. cf> DISCON 대신 CLOSE를 사용할 경우, Disconnect-process(disconnect-request 전송)없이, 단지 Sn_SSR만 SOCK_CLOSED로 변경된다. 그리고 통신 중 상대방으로부터 RST packet을 수신할 경우, 무조건 Sn_SSR은 SOCK_CLOSED로 변경된다.
0x10	CLOSE	SOCKETn을 close한다. Sn_SSR은 SOCK_CLOSED로 변경된다.
0x20	SEND	상대방에게 Sn_TX_WRSR으로 설정된 크기의 Data를 전송한다. TCP나 UDP mode에서, Sn_TX_WRSR이 Maximum segment size(MSS)보다 클 경우 W5300은 자동으로 Data를 MSS 단위로 나누고, 나누어진 Data(DATA packet)을 전송하게 된다. 그러나 IPRAW나 MACRAW Mode에서는 이와 같은 기능은 지원되지 않고 Host가 전송 Data를 직접 MSS 단위로 나누어 전송해야 한다. SEND에 대한 처리가 완료되었을 경우 Sn_IR (SENDOK)='1'로 된다. Host는 Sn_IR(SENDOK)='1'를 확인 후 그 다음 Data에 대한 SEND command를 내릴 수 있다.

		SEND에 의해 DATA packet를 상대방에게 성공적으로 전송한 경우 (상대방으로부터 DATA/ACK packet을 수신한 경우) Sn_TX_FSR은 전송 DATA packet size만큼 증가한다. 그렇지 못한 경우(DATA/ACK packet을 수신하지 못했을 경우), TCP_{TO}가 발생(Sn_IR(3)= '1')하고 Sn_SSR은 SOCK_CLOSED로 변경된다. cf> SEND 이전에, Host는 전송할 Data를 Sn_TX_FIFOR을 통해 SOCKETn의 Internal TX memory로 copy하고, Data size를 Sn_TX_WRSR에 설정해야 한다.
0x21	SEND_MAC	UDP(Sn_MR(P3:P0)=Sn_MR_UDP)나 IPRAW((Sn_MR(P3:P0)=Sn_MR_IPRAW) mode일 때만 유효하다. 기본동작은 SEND와 같다. SEND는 자동으로 ARP-process를 통해 Destination hardware address를 얻은 후 Data를 전송하는 반면, SEND_MAC은 Host가 설정한 Sn_DHAR을 Destination hardware address로 하여 Data를 전송한다. SEND_MAC은 Hardware address를 이미 알고 있는 Destination으로 UDP나 IPRAW data를 전송할 때 불필요한 ARP-process를 없애 Network traffic을 감소시킬 수 있다.
0x22	SEND_KEEP	TCP mode일 때만 유효하다. 상대방의 TCP 접속 상태를 Check하기 위해 KEEP ALIVE(KA) packet을 전송한다. SEND_KEEP은 Sn_KPALVTR = 0 일 때만 동작하며, Sn_KPALVTR > 0 일 경우 무시된다. Sn_KPALVTR > 0 인 경우, Sn_KPALVTR의 설정 시간 동안 Data 송수신이 없을 때 자동으로 상대방에게 KA packet을 전송한다. KA packet을 성공적으로 전송했다면(KA/ACK packet을 상대방으로부터 수신했다면), Sn_SSR은 SOCK_ESTABLISHED를 계속 유지한다. 실패 했을 경우(상대방이 이미 접속을 종료했거나, KA/ACK를 전송하지 않을 때)는 TCP_{TO}가 발생(Sn_IR(3)= '1')하고 Sn_SSR은 SOCK_CLOSED로 변경된다. cf> KA packet은 접속 이후 한번 이상의 Data 송신이나 수신 이후에

		전송될 수 있다.
0x40	RECV	Host가 SOCKETn의 수신 DATA packet을 수신했음을 알린다. cf> RECV 이전에, Host는 SOCKETn의 Internal RX memory에서 수신 DATA packet를 Sn_RX_FIFOR을 통해 Host memory로 Copy해야 한다.
아래 command들은 SOCKET0이고 S0_MR(P3:P0)=S0_MR_PPPoE일 때만 유효하다. *"How to use PPPOE"*를 참조하라.		
0x23	PCON	PPPoE discovery packet 전송을 시작으로 PPPoE connection을 시작한다.
0x24	PDISCON	PPPOE connection을 종료한다.
0x25	PCR	각 Phase에서, REQ message를 전송한다.
0x26	PCN	각 Phase에서, NAK message를 전송한다.
0x27	PCJ	각 Phase에서 REJECT message를 전송한다.

Sn_IMR (SOCKETn Interrupt Mask Register)[R/W] [0x08204+0x40n/0x204+0x40n] [0x--FF]

Host로 알려줄 SOCKETn의 Interrupt를 설정한다.

Sn_IMR의 Interrupt mask bit들은 Sn_IR의 Interrupt bit들과 각각 대응된다. 임의의 SOCKET interrupt가 발생하고 Sn_IMR의 그 bit가 '1'로 설정되어있을 경우 Sn_IR의 대응 Bit가 '1'로 설정된다. Sn_IMR과 Sn_IR의 임의 bit가 모두 '1'일 때 IR(n)='1'된다. 이때 IMR(n)='1'이라면 Host에게 Interrupt가 Issue('/INT' signal low assert)된다.

Sn_IMR0	15	14	13	12	11	10	9	8
0x08204 + 0x40n	–	–	–	–	–	–	–	–
0x204 + 0x40n	0	0	0	0	0	0	0	0
Sn_IMR1	7	6	5	4	3	2	1	0
0x08205 + 0x40n	PRECV	PFAIL	PNEXT	SENDOK	TIMEOUT	RECV	DISCON	CON
0x205 + 0x40n	1	1	1	1	1	1	1	1

Sn_IMR(15:8)/Sn_IMR0(7:0) : All Reserved

Sn_IMR(7:0)/Sn_IMR1(7:0)

Bit	Symbol	Description
7	PRECV	Sn_IR(PRECV) Interrupt Mask SOCKET=0 이고 S0_MR(P3:P0)=S0_MR_PPPoE일 때만 유효하다.
6	PFAIL	Sn_IR(PFAIL) Interrupt Mask

		SOCKET=0 이고 S0_MR(P3:P0)=S0_MR_PPPoE일 때만 유효하다.
5	PNEXT	Sn_IR(PNEXT) Interrupt Mask SOCKET=0 이고 S0_MR(P3:P0)=S0_MR_PPPoE일 때만 유효하다.
4	SENDOK	Sn_IR(SENDOK) Interrupt Mask
3	TIMEOUT	Sn_IR(TIMEOUT) Interrupt Mask
2	RECV	Sn_IR(RECV) Interrupt Mask
1	DISCON	Sn_IR(DISCON) Interrupt Mask
0	CON	Sn_IR(CON) Interrupt Mask

Sn_IR (SOCKETn Interrupt Register) [R/W] [0x08206+0x40n/0x206+0x40n] [0x--00]

Sn_IR은 Host에게 Establishment, Termination, Receiving data, Timeout과 같은 SOCKETn의 Interrupt 종류를 알려주기 위한 Register이다.

임의의 Interrupt가 발생하고 Sn_IMR의 해당 Mask bit이 '1'인 경우 Sn_IR의 그 Interrupt bit가 '1'이 된다.

'1'로 설정된 Sn_IR의 Bit를 Clear하기 위해서는 그 Bit를 '1'로 Host-Write한다. Sn_IR의 모든 Bit이 '0'으로 Clear될 때, IR(n)은 자동으로 Clear된다.

Sn_IR0	15	14	13	12	11	10	9	8
0x08206 + 0x40n	-	-	-	-	-	-	-	-
0x206 + 0x40n	0	0	0	0	0	0	0	0
Sn_IR1	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0x08207 + 0x40n	PRECV	PFAIL	PNEXT	SENDOK	TIMEOUT	RECV	DISCON	CON
0x207 + 0x40n	0	0	0	0	0	0	0	0

Sn_IR(15:8)/Sn_IR0(7:0) : All Reserved

Sn_IR(7:0)/Sn_IR1(7:0)

Bit	Symbol	Description
7	PRECV	PPP Receive Interrupt 지원하지 않는 Option data를 수신하였을 경우 설정
6	PFAIL	PPP Fail Interrupt PAP authentication이 실패했을 경우 설정
5	PNEXT	PPP Next Phase Interrupt PPPoE connection 처리과정에서 Phase 변경 시 설정
4	SENDOK	SEND OK Interrupt

		SEND command가 완료되었을 경우 설정
3	TIMEOUT	TIMEOUT Interrupt ARP_{TO}나 TCP_{TO}가 발생했을 경우 설정
2	RECV	Receive Interrupt 상대방으로부터 DATA packet을 수신할 때 마다 설정
1	DISCON	Disconnect Interrupt 상대방으로부터 FIN packet이나 FIN/ACK Packet을 수신하였을 경우 설정
0	CON	Connect Interrupt 상대방과의 접속이 성공적으로 이루어졌을 경우 설정

Sn_SSR (SOCKETn SOCKET Status Register) [R] [0x08208+0x40n/0x208+0x40n] [0x--00]

SOCKETn의 SOCKET status를 알려준다. SOCKET status는 Sn_CR의 Command나, Packet 송수신에 의해 변경될 수 있다.

Sn_SSR(0x08208+0x40n/0x208+0x40n)	
Sn_SSR0(0x08208+0x40n/0x208+0x40n)	Sn_SSR1(0x08209+0x40n/0x209+0x40n)
Reserved	Socket status

Sn_SSR(15:8)/Sn_SSR0(7:0) : All Reserved

Sn_SSR(7:0)/Sn_SSR1(7:0)

Value	Symbol	Description
0x00	SOCK_CLOSED	SOCKETn의 Resource가 Release된 상태. DISCON, CLOSE command가 수행되거나 ARP_{TO},TCP_{TO}가 발생했을 경우 이전 값에 관계없이 전이된다. 이 상태에서는 오직 OPEN command만 수행 가능하다.
0x13	SOCK_INIT	SOCKETn이 TCP mode로 Open된 상태. Sn_MR(P3:P0)=Sn_MR_TCP이고, OPEN command가 수행되었을 때 전이되며, TCP connection establishment의 초기 단계이다. “TCP SERVER”로 동작할 경우 LISTEN, “TCP CLIENT”로 동작할 경우 CONNECT command가 수행 가능하다.
0x14	SOCK_LISTEN	SOCKETn이 “TCP SERVER”로 동작하며, “TCP CLIENT”의

		connection-request(SYN packet)를 기다리는 상태. LISTEN command가 수행되었을 때 전이된다. SOCK_LISTEN에서 "TCP CLIENT"의 Connect-request (SYN packet) 처리를 성공했을 경우 SOCK_ESTABLISHED로 전이된다. 실패했을 경우 TCP_{TO}가 발생(Sn_IR(TIME OUT)='1')하고 SOCK_CLOSED로 전이된다.
0x17	SOCK_ESTABLISHED	TCP connection이 established 상태. SOCK_LISTEN에서 "TCP CLIENT"의 SYN packet 처리를 성공했을 경우나 CONNECT command에 수행이 성공했을 경우 전이된다. 이 상태에서 DATA packet 송수신이 가능하다. 즉 SEND나 RECV command를 수행할 수 있다.
0x1C	SOCK_CLOSE_WAIT	상대로부터 Disconnect-request(FIN packet)를 수신한 상태. TCP connection이 완전히 Disconnect된 것이 아닌 Half close 상태이므로 DATA packet 송수신이 가능하다. TCP connection을 완전히 Disconnect 하기 위해선 DISCON command를 수행한다. 하지만 단순히 SOCKET을 Close하기 원한다면 CLOSE command를 수행한다.
0x22	SOCK_UDP	SOCKETn이 UDP mode로 Open된 상태. Sn_MR(P3:P0) = Sn_MR_UDP이고, OPEN command가 수행되었을 때 전이되며, TCP mode SOCKET과 같은 Connection-process없이 DATA packet을 직접 송수신할 수 있다.
0x32	SOCK_IPRAW	SOCKETn이 IPRAW mode로 Open된 상태. Sn_MR(P3:P0) = Sn_MR_IPRAW이고, OPEN command가 수행되었을 때 전이되며, UDP mode SOCKET 처럼 Connection-process없이 IP packet을 packet을 직접 송수신할 수 있다.
0x42	SOCK_MACRAW	SOCKET0이 MACRAW mode로 Open된 상태. S0_MR(P3:P0)=S0_MR_MACRAW이고, S0_CR=OPEN 일 때 전이되며, UDP mode SOCKET처럼 Connection-process없

		이 MAC packet(Ethernet frame)을 직접 송수신할 수 있다.
0x5F	SOCK_PPPoE	SOCKET0이 PPPoE mode로 Open된 상태 S0_MR(P3:P0)=S0_MR_PPPoE이고, S0_CR=OPEN 일 때 전이되며, PPPoE connection에서 일시적으로 사용된다. "*How to use PPPoE in W5300*"을 참조.

아래 SOCKET status은 Sn_SSR의 전이 과정에서 관찰될 수 있는 temporary Status들이다.

0x15	SOCK_SYNSENT	"TCP SERVER"에게 Connect-request(SYN packet)를 전송한 상태. 이 Status는 CONNECT command에 의한 SOCK_INIT에서 SOCK_ESTABLISEHD로의 전이과정에서 나타난다. 이 Status에서 "TCP SEVER"로부터 Connect-accept (SYN/ACK packet)을 수신할 경우 자동으로 SOCK_ESTABLISHED로 전이된다. "TCP SEVER"로부터 TCP_{TO} 발생(Sn_IR(TIMEOUT)='1') 이전까지 SYN/ACK packet을 수신하지 못할 경우 SOCK_CLOSED로 전이한다.
0x16	SOCK_SYNRECV	"TCP CLIENT"로부터 Connect-request(SYN packet)를 수신한 상태. W5300이 Connect-request에 대한 응답으로 Connect-accept (SYN/ACK packet)을 "TCP CLIENT"에게 성공적으로 전송하였을 경우 자동으로 SOCK_ESTABLISHED로 전이한다. 전송에 실패하였을 경우 TCP_{TO}가 발생 (Sn_IR(TIMEOUT)='1')하고 SOCK_CLOSED로 전이된다.
0x18	SOCK_FIN_WAIT	SOCKETn이 Closing되는 상태 Active close나 Passive close시, Disconnect-process에서 관찰된다. Disconnect-process 과정이 성공적으로 완료되거나, TCP_{TO}가 발생(Sn_IR(TIMEOUT)='1')하면 SOCK_CLOSED로 전이된다.
0X1B	SOCK_TIME_WAIT	
0X1D	SOCK_LAST_ACK	
0x01	SOCK_ARP	Destination hardware address를 찾기 위해 ARP-request를 전송하는 상태 이 상태는 SOCK_UDP나 SOCK_IPRAW에서 SEND command를 수행 할 경우 관찰되거나, SOCK_INIT에서 CONNECT command를 수행할 경우 관찰되는 Status이다.

		Destination으로부터 Hardware address를 성공적으로 얻은 경우(ARP-response을 수신한 경우), SOCK_UDP, SOCK_IPRAW, SOCK_SYNSENT로 각각 전이된다. 실패할 경우 ARP_{TO}가 발생(Sn_IR(TIMEOUT)='1')하고, UDP나 IPRAW mode일 경우 이전 Status인 SOCK_UDP나 SOCK_IPRAW로 되돌아 가며, TCP인 경우 SOCK_CLOSED로 전이된다. cf> SOCK_UDP나 SOCK_IPRAW에서, 이전 SEND command에 대한 Sn_DIPR와 현재 SEND command의 Sn_DIPR이 다를 경우에만 ARP-process가 동작한다. Sn_DIPR이 같을 경우 ARP-process 없이 이전에 획득한 Destination hardware address를 그대로 사용한다.

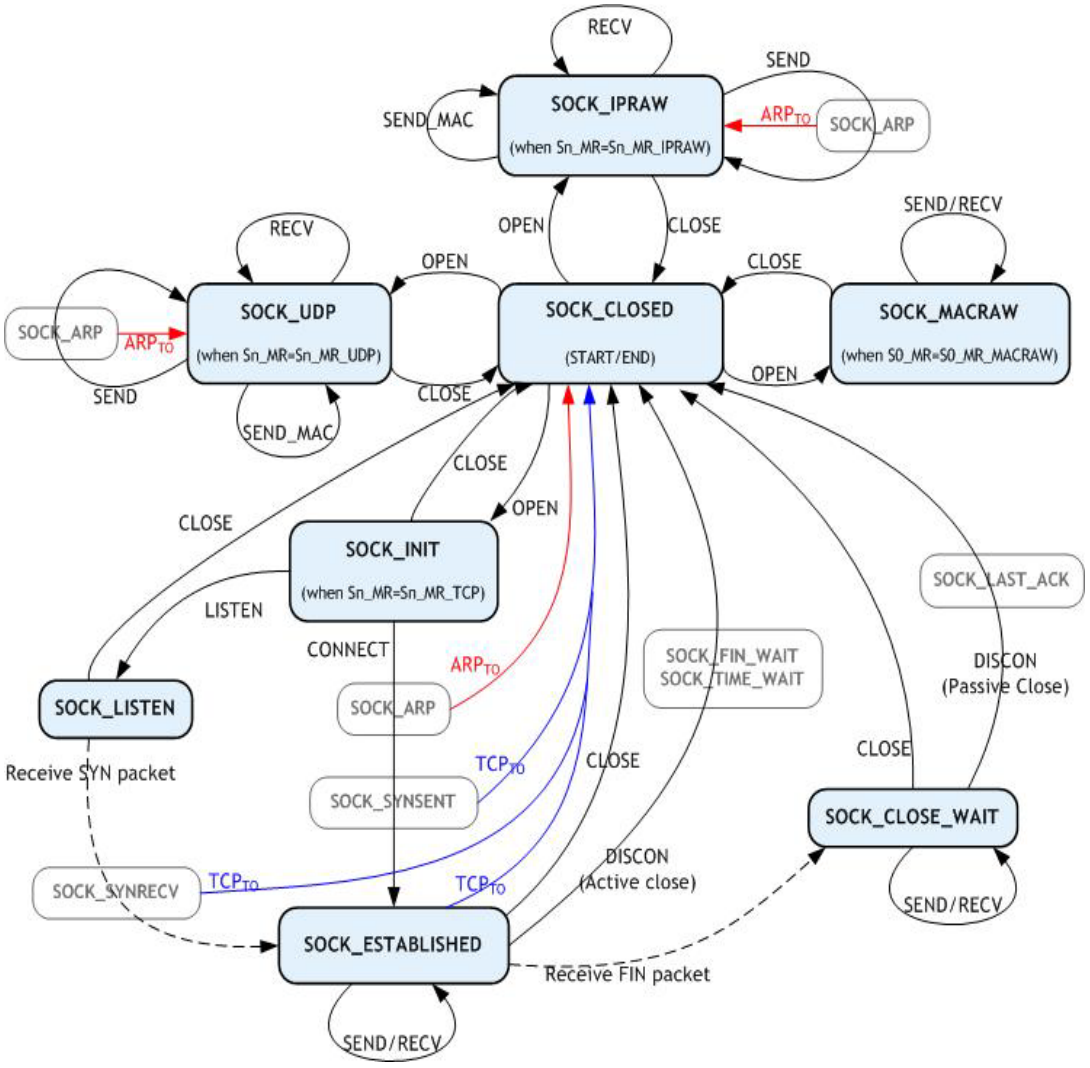


Fig 5. SOCKETn Status Transition

Sn_PORTR(SOCKETn Source Port Register)[R/W][0x0820A+0x40n/0x20A+0x40A] [0x0000]

Source port number를 설정한다.

SOCKETn을 TCP나 UDP mode로 사용할 때만 유효하며, 그 외 mode에서는 무시된다.

OPEN Command 이전에 반드시 설정한다.

Ex) Sn_PORTR = 5000(0x1388)

Sn_PORTR(0x0820A+0x40n/0x20A+0x40n)	
Sn_PORTR0(0x0820A+0x40n/0x20A+0x40n)	Sn_PORTR1(0x0820B+0x40n/0x20B+0x40n)
0x13	0x88

Sn_DHAR (SOCKETn Destination Hardware Address Register) [R/W]

[0x0820C+0x40n/0x20C+0x40n] [FF.FF.FF.FF.FF.FF]

SOCKETn의 Destination hardware address를 설정하거나 설정된다. 또한 SOCKET0이 PPPoE mode로 사용될 경우 S0_DHAR은 이미 알고 있는 PPPoE server hardware address로 설정한다.

UDP나 IPRAW mode에서 SEND_MAC command를 사용할 경우 SOCKETn의 Destination hardware address를 설정한다. 또한 TCP, UDP, IPRAW mode에서 Sn_DHAR은 CONNECT나 SEND command에 의한 ARP-process를 통해 획득한 Destination hardware address로 설정된다. Host는 CONNECT나 SEND command 성공 이후 Sn_DHAR을 통해 Destination hardware address를 알 수 있다.

PPPoE mode에서, W5300의 PPPoE-process를 이용할 경우 PPPoE server hardware address를 따로 설정할 필요는 없다. 하지만 W5300의 PPPoE-process를 이용하지 못하고 MACRAW mode로 PPPoE-process를 직접 구현하여 처리한 경우라 할지라도, PPPoE packet을 송수신하기 위해서는, 직접 구현한 PPPoE-process를 통해 획득한 PPPoE server hardware address, PPPoE server IP address, PPP session ID를 설정하고 MR(PPPoE)를 '1'로 반드시 설정한다. S0_DHAR는 이미 알고 있는 PPPoE server hardware address를 설정하며, OPEN command 이전에 설정한다. S0_DHAR을 통해 설정된 PPPoE server hardware address는 OPEN command 이후 PDHAR에 반영된다.

설정된 PPPoE information은 CLOSE command 이후에도 내부적으로 계속 유효하다.

Ex) Sn_DHAR = 00.08.DC.01.02.10

Sn_DHAR(0x0820C+0x40n/0x20C+0x040n)	
Sn_DHAR0(0x0820C+0x40n/0x20C+0x040n)	Sn_DHAR1(0x0820D+0x40n/0x20D+0x040n)
0x00	0x08
Sn_DHAR2(0x0820E+0x40n/0x20E+0x040n)	
Sn_DHAR2(0x0820E+0x40n/0x20E+0x040n)	Sn_DHAR3(0x0820F+0x40n/0x20F+0x040n)
0xDC	0x01
Sn_DHAR4(0x08210+0x40n/0x210+0x040n)	

Sn_DHAR4(0x08210+0x40n/0x210+0x040n)	Sn_DHAR5(0x08211+0x40n/0x211+0x040n)
0x02	0x10

Sn_DPORTR (SOCKETn Destination Port Register) [WO]

[0x08212+0x40n/0x212+0x40n] [0x0000]

SOCKETn의 Destination port number를 설정하거나, SOCKET0이 PPPoE mode로 사용될 경우 S0_DPORTR은 이미 알고 있는 PPP Session ID로 설정한다.

TCP, UDP, PPPoE mode에서만 유효하고, 그 외의 mode에서는 무시된다.

TCP mode에서, "TCP CLIENT"로 동작할 경우 접속하기 위한 "TCP SERVER"의 Listen port number로 설정하고, CONNECT command 이전에 설정한다.

UDP mode에서, Sn_DPORTR은 UDP DATA packet 전송에 사용될 Port number로 SEND나 SEND_MAC command 이전에 설정한다.

PPPoE mode에서, S0_PDHAR과 같은 경우로 S0_DPORTR는 이미 알고 있는 PPP Session ID를 설정한다. S0_DPORTR을 통해 설정된 PPP Session ID는 OPEN command 이후 PSIDR에 반영된다.

Ex) Sn_DPORTR = 5000(0x1388)

Sn_DPORTR(0x08212+0x40n/0x212+0x40n)	
Sn_DPORTR0(0x08212+0x40n/0x212+0x40n)	Sn_DPORTR1(0x08213+0x40n/0x213+0x40n)
0x13	0x88

Sn_DIPR (SOCKETn Destination IP Address Register) [R/W]

[0x08214+0x40n/0x214+0x40n] [00.00.00.00]

SOCKETn의 Destination IP address를 설정하거나 설정되며, SOCKET0이 PPPoE mode로 사용될 경우 S0_DIPR은 이미 알고 있는 PPPoE server IP address로 설정한다.

TCP, UDP, IPRAW, PPPoE mode에서만 유효하고, MACRAW mode에서는 무시된다.

TCP mode에서, "TCP CLIENT"로 동작할 경우 접속하기 위한 "TCP SERVER"의 IP address로 설정하고, CONNECT command 이전에 설정한다. "TCP SERVER"로 동작할 경우 "TCP CLIENT"와 접속 성공 이후 내부적으로 "TCP CLIENT"의 IP address로 설정된다.

UDP나 IPRAW mode에서, Sn_DIPR은 UDP나 IP DATA packet 전송에 사용될 Destination IP address로 SEND나 SEND_MAC command 이전에 설정한다.

PPPoE mode에서, S0_DHAR과 같은 경우로 S0_DIPR는 이미 알고 있는 PPPoE server IP address를 설정한다.

Ex) Sn_DIPR = 192.168.0.11

Sn_DIPR(0x08214+0x40n/0x214+0x040n)	
Sn_DIPR0(0x08214+0x40n/0x214+0x040n)	Sn_DIPR1(0x08215+0x40n/0x215+0x040n)
192 (0xC0)	168 (0xA8)
Sn_DHAR2(0x08216+0x40n/0x216+0x040n)	
Sn_DIPR2(0x08216+0x40n/0x216+0x040n)	Sn_DIPR3(0x08217+0x40n/0x217+0x040n)
0 (0x00)	11 (0x0B)

Sn_MSSR (SOCKETn Maximum Segment Size Register) [R/W]

[0x08218+0x40n/0x218+0x40n] [0x0000]

SOCKETn의 MTU(Maximum Transfer Unit)를 설정하거나, 설정된 MTU를 알려준다.
Host가 Sn_MSSR를 설정하지 않을 경우는 Default MTU로 설정된다.

TCP나 UDP mode만 지원하며, PPPoE를 사용할 경우(MR(PPPoE)='1') PPPoE의 MTU내에서 TCP나 UDP mode의 MTU가 결정된다. IPRAW나 MACRAW는 내부적으로 MTU를 처리하지 않고 Default MTU가 적용되므로, Host는 Default MTU보다 큰 Data를 전송할 때 Data를 Default MTU 단위로 직접(Manually) 나누어 전송해야 한다.
TCP나 UDP mode에서는 Host가 전송할 Data가 설정된 MTU보다 클 경우, W5300은 설정된 MTU 단위로 Data를 내부적으로(Automatically) 나누어 전송한다.
MTU는 TCP mode에서 MSS라 불리며, MSS는 TCP connection 과정을 통해 Host-Written-Value(Host 설정 값)와 상대방의 MSS 값 중 작은 값으로 자동으로 설정된다.
UDP mode에서는 TCP mode와 같은 Connection-process가 없고 Host-Written-Value를 그대로 사용한다. MTU가 서로 다른 상대방과 통신 할 경우, W5300은 ICMP(Fragment MTU) packet을 수신할 수 있다. 이 경우 IR(FMTU)='1'가 되고 Host는 FMTUR과 UIPR을 통해 Fragment MTU와 Destination IP address를 알 수 있다. IR(FMTU)='1'일 경우 그 상대방과는 UDP 통신이 불가능하므로, 해당 SOCKET을 close하고 알아낸 FMTU를 Sn_MSSR로 설정한 후 OPEN command로 open하여 다시 통신을 시도한다.

Mode	Normal (MR(PPPoE)='0')		PPPoE (MR(PPPoE)='1')	
	Default MTU	Range	Default MTU	Range
TCP	1460	1 ~ 1460	1452	1 ~ 1452
UDP	1472	1 ~ 1472	1464	1 ~ 1464
IPRAW	1480		1472	
MACRAW	1514			

Ex) Sn_MSSR = 1460 (0x05B4)

Sn_MSSR(0x08218+0x40n/0x218+0x040n)	
Sn_MSSR0(0x08218+0x40n/0x218+0x040n)	Sn_MSSR1(0x08219+0x40n/0x219+0x040n)
0x05	0xB4

Sn_KPALVTR(SOCKETn Keep Alive Time Register)[R/W]
[0x0821A+40n/0x21A+0x40n][0x00]

1 byte register로 SOCKETn의 KEEP ALIVE(KA) packet의 전송 Time을 설정한다. TCP mode만 유효하며, 그 외 mode는 무시된다. 단위는 5s이다.

KA packet은 Sn_SSR이 SOCK_ESTABLISHED로 전이되고 한번 이상의 DATA packet 송신이나 수신 이후 전송이 가능하다. Sn_KPALVTR > 0일 경우, 설정된 Time-period가 지나게 되면 W5300은 내부적으로(automatically) KA packet을 전송하여 TCP connection을 Check 한다(Auto-Keep-Alive-process). Sn_KPALVTR = 0 일 경우는 Auto-Keep-Alive-process는 동작하지 않으며, Host의 SEND_KEEP command에 의해 KA packet이 전송될 수 있다 (Manual-Keep-Alive-process). Manual-Keep-Alive-process는 Sn_KPALVTR > 0 일 경우 무시된다.

Ex) Sn_KPALVTR = 10, 매 50s마다 KA packet을 전송

Sn_PROTOR(0x0821A+0x40n/0x21A+0x040n)	
Sn_KPALVTR(0x0821A+0x40n/0x21A+0x040n)	Sn_PROTOR (0x0821B+0x40n/0x21B+0x040n)
10 (0x0A)	Sn_PROTOR

Sn_PROTOR (SOCKETn Protocol Number Register)[R/W]
[0x0821B+40n/0x21B+0x40n] [0x00]

1 byte register로 IP layer에서 IP header의 Protocol number field를 설정한다.

IPRAW mode에서만 유효하며, 그 외 mode는 무시된다. Sn_PROTOR은 OPEN command 이전에 설정한다. IPRAW mode로 Open된 SOCKETn은 Sn_PROTOR에 설정된 Protocol number의 Data만을 송수신한다. Sn_PROTOR은 0x00 ~ 0xFF 의 범위 내에서 설정 가능하나, W5300은 TCP(0x06), UDP(0x11) protocol number은 지원하지 않는다.

Protocol number는 IANA(Internet Assigned Numbers Authority)에서 정의하고 있으며, IANA의 online document(http://www.iana.org/assignments/protocol-numbers)를 참조하라.

Ex) Sn_PROTOR = 0x01 (ICMP)

Sn_PROTOR(0x0821A+0x40n/0x21A+0x040n)	
Sn_KPALVTR(0x0821A+0x40n/0x21A+0x040n)	Sn_PROTOR (0x0821B+0x40n/0x21B+0x040n)
Sn_KPALVTR	0x01

Sn_TOSR (SOCKETn TOS Register) [R/W] [0x0821C+40n/0x21C+40n] [0x00]

IP layer에서 IP header의 TOS(Type of service) field를 설정한다. OPEN command 이전에 설정한다. http://www.iana.org/assignments/ip-parameters 참조.

Ex) Sn_TOSR = 0x00

Sn_TOSR(0x0821C+0x40n/0x21C+0x040n)	
Sn_TOSR0(0x0821C+0x40n/0x21C+0x040n)	Sn_TOSR1(0x0821D+0x40n/0x21D+0x040n)
Reserved	0x00

Sn_TTLR (SOCKETn TTL Register) [R/W] [0x0821E+40n/0x21E+40n] [0x80]

IP layer에서 IP header의 TTL(Time to live) field를 설정한다. OPEN command 이전에 설정한다. http://www.iana.org/assignments/ip-parameters 참조.

Ex) Sn_TTLR = 128 (0x80)

Sn_TTLR(0x0821E+0x40n/0x21E+0x040n)	
Sn_TTLR0(0x0821E+0x40n/0x21E+0x040n)	Sn_TTLR1(0x0821F+0x40n/0x21F+0x040n)
Reserved	0x80

Sn_TX_WRSR (SOCKETn TX Write Size Register) [R/W] [0x08220+40n/0x220+40n] [0x00000000]

Sn_TX_FIFOR을 통해 Internal TX memory에 Write한 Data의 Byte size를 설정한다. SEND나 SEND_MAC command 이전에 설정하며, TMSRn에 의해 설정된 Internal TX memory size보다 크게 설정할 수 없다.

TCP나 UDP mode이고 Sn_TX_WRSR > Sn_MSSR 인 경우, W5300은 내부적으로 (Automatically) Sn_MSSR 단위로 Data를 나누어 전송한다. 그 외 Mode에서는 Sn_TX_WRSR을 Sn_MSSR보다 크게 설정해선 안 된다.

Ex1) Sn_TX_WRSR = 64KB = 65536 = 0x00010000

Sn_TX_WRSR(0x08220+0x40n/0x220+0x040n)								
Sn_TX_WRSR0(0x08220+0x40n/0x220+0x040n)	Sn_TX_WRSR1(0x08221+0x40n/0x221+0x040n)							
Reserved	-	-	-	-	-	-	-	'1'
Sn_TX_WRSR2(0x08222+0x40n/0x222+0x040n)								
Sn_TX_WRSR2(0x08222+0x40n/0x222+0x040n)	Sn_TX_WRSR3(0x08223+0x40n/0x21D+0x040n)							
0x00	0x00							

Ex2) Sn_TX_WRSR = 2017 = 0x000007E1

Sn_TX_WRSR(0x08220+0x40n/0x220+0x040n)								
Sn_TX_WRSR0(0x08220+0x40n/0x220+0x040n)	Sn_TX_WRSR1(0x08221+0x40n/0x221+0x040n)							
Reserved	-	-	-	-	-	-	-	'0'
Sn_TX_WRSR2(0x08222+0x40n/0x222+0x040n)								
Sn_TX_WRSR2(0x08222+0x40n/0x222+0x040n)	Sn_TX_WRSR3(0x08223+0x40n/0x223+0x040n)							
0x07	0xE1							

Sn_TX_FSR (SOCKETn TX Free Size Register) [R]

[0x08224+40n/0x224+40n] [0x00002000]

SOCKETn의 Internal TX memory의 Free size(전송 가능한 Data의 Byte size)를 알려준다. Sn_TX_FSR보다 크게 Sn_TX_FIFOR을 Host-Write하면 안 된다. 따라서 Data 전송 전에 Sn_TX_FSR를 반드시 확인하고, 전송할 Data의 크기가 Sn_TX_FSR보다 작거나 같으면 SEND나 SEND_MAC command로 Data를 전송한다.

TCP mode에서는 상대방으로부터 Data 수신이 확인(DATA/ACK packet 수신)되면, Sn_TX_FSR은 상대방이 수신한 DATA packet 크기만큼 내부적으로 증가하게 된다. 그 외 mode에서는 Sn_IR(SENDOK) = '1'인 경우 Sn_TX_FSR은 전송한 Data size만큼 내부적으로 증가하게 된다.

Ex1) Sn_TX_FSR = 64KB = 65536 = 0x00010000

Sn_TX_FSR(0x08224+0x40n/0x224+0x040n)								
Sn_TX_FSR0(0x08224+0x40n/0x214+0x040n)	Sn_TX_FSR1(0x08225+0x40n/0x225+0x040n)							
Reserved	-	-	-	-	-	-	-	'1'
Sn_TX_FSR2(0x08226+0x40n/0x226+0x040n)								
Sn_TX_FSR2(0x08226+0x40n/0x226+0x040n)	Sn_TX_FSR3(0x08227+0x40n/0x227+0x040n)							
0x00	0x00							

Ex2) Sn_TX_FSR = 33332 = 0x00008234

Sn_TX_FSR(0x08224+0x40n/0x224+0x040n)								
Sn_TX_FSR0(0x08224+0x40n/0x224+0x040n)	Sn_TX_FSR1(0x08225+0x40n/0x225+0x040n)							
Reserved	-	-	-	-	-	-	-	'0'
Sn_TX_FSR2(0x08226+0x40n/0x226+0x040n)								
Sn_TX_FSR2(0x08226+0x40n/0x226+0x040n)	Sn_TX_FSR3(0x08227+0x40n/0x227+0x040n)							
0x82	0x34							

Sn_RX_RSR (SOCKETn RX Received Size Register) [R]

[0x08228+40n/0x228+40n] [0x00000000]

SOCKETn의 Internal RX memory의 Received data의 Byte size를 알려준다.

Sn_RX_RSR보다 크게 Sn_RX_FIFOR을 Host-Read하면 안 된다. 따라서 Data 수신 전에 Sn_RX_RSR를 반드시 확인하고, Sn_RX_RSR보다 같거나 작게 Sn_RX_FIFOR을 Host-Read 하여 Host system memory로 Copy한다. Memory copy 후에는 RECV command를 수행하여 수신 Data copy를 완료했음을 W5300에게 알린다. Sn_RX_RSR은 Sn_RX_FIFOR를 Host-Read 할 때마다, 2 bytes씩 내부적으로 감소한다. Sn_RX_RSR > 0 일 경우 하나 이상의 DATA packet이 존재할 수 있고 DATA packet 단위로 처리되어야 한다. Sn_RX_FIFOR을 참조하라.

Ex1) Sn_RX_RSR = 64KB = 65536 = 0x00010000

Sn_RX_RSR(0x08228+0x40n/0x228+0x040n)								
Sn_RX_RSR0(0x08228+0x40n/0x21C+0x040n)	Sn_RX_RSR1(0x08229+0x40n/0x229+0x040n)							
Reserved	–	–	–	–	–	–	–	'1'
Sn_RX_RSR2(0x0822A+0x40n/0x22A+0x040n)								
Sn_RX_RSR2(0x0822A+0x40n/0x22A+0x040n)	Sn_RX_RSR3(0x0822B+0x40n/0x22B+0x040n)							
0x00	0x00							

Ex2) Sn_RX_RSR = 3800 = 0x00000ED8

Sn_RX_RSR(0x08228+0x40n/0x228+0x040n)								
Sn_RX_RSR0(0x08228+0x40n/0x21C+0x040n)	Sn_RX_RSR1(0x08229+0x40n/0x229+0x040n)							
Reserved	–	–	–	–	–	–	–	'0'
Sn_RX_RSR2(0x0822A+0x40n/0x22A+0x040n)								
Sn_RX_RSR2(0x0822A+0x40n/0x22A+0x040n)	Sn_RX_RSR3(0x0822B+0x40n/0x22B+0x040n)							
0x0E	0xD8							

Sn_FRAGR (SOCKETn Fragment Register) [R/W] [0x0822C+40n/0x22C+40n] [0x40]

IP layer에서 IP header의 Fragment field를 설정한다. W5300은 IP layer의 Packet fragment는 지원하지 않는다. Sn_FRAGR를 설정하더라도 IP data는 Fragment되지 않으며 이를 설정하는 것은 권장하지 않는다. OPEN command 이전에 설정한다.

Ex) Sn_FRAGR = 0x40 (Don't Fragment)

Sn_FRAGR(0x0822C+0x40n/0x22C+0x040n)	
Sn_FRAGR0(0x0822C+0x40n/0x22C+0x040n)	Sn_FRAGR1(0x0822D+0x40n/0x22D+0x040n)
Reserved	0x40

Sn_TX_FIFOR (SOCKETn TX FIFO Register) [R/W] [0x0822E+40n/0x22E+40n] [0xUUUU]

SOCKETn의 Internal TX memory를 간접적으로 접근한다.

SOCKETn의 Internal TX memory는 Host에 의해 직접적으로 접근될 수 없으며, Sn_TX_FIFOR을 통해서만 접근이 허용된다. MR(MT) = '0' 인 경우 Internal TX memory는 Sn_TX_FIFOR을 통해 Host-Write만 허용된다. MR(MT) = '1' 인 경우 Host-Write와 Host-Read 모두 허용되며, Target host system과 W5300간의 Interface 검증 후에는 반드시 '0'으로 설정한다.("*How to Test Internal TX/RX Memory*" 참조).

Target host system이 8bit data bus width를 사용한다면 반드시 Sn_TX_FIFOR0와 Sn_TX_FIFOR1를 한 쌍(pair)으로 접근해야 한다. 1byte 크기의 Data를 Internal TX memory로 Copy할지라도, 그 1byte data는 Sn_TX_FIFOR0에 Host-Write하고, Sn_TX_FIFOR1은 dummy data로 Host-Write해야 한다. Sn_TX_FIFOR은 반드시 2bytes 크기로 접근해야 하며, Low address register인 Sn_TX_FIFOR0를 먼저 접근한 후 high address register인 Sn_TX_FIFOR1을 접근해야 한다. Sn_TX_FIFOR0을 접근 후 Sn_TX_FIFOR1 이외에 다른 W5300 Register의 접근은 허용되지 않는다.

임의의 Data를 2bytes씩 Sn_TX_FIFOR을 통해 Host-Write할 경우 그 Data는 Internal TX memory로 순차적으로 Copy된다. Sn_TX_FIFOR0과 Sn_TX_FIFOR1의 값들은 Internal TX memory의 low address와 high address로 각각 저장된다. Internal TX memory에 저장된 Data들은 SEND나 SEND_MAC command에 의해 Low address부터 순서대로 전송된다.

Ex1) Sn_TX_FIFOR = 0x1122

Sn_TX_FIFOR(0x0822E+0x40n/0x22E+0x040n)	
Sn_TX_FIFOR0(0x0822E+0x40n/0x22E+0x040n)	Sn_TX_FIFOR1(0x0822F+0x40n/0x22F+0x040n)
0x11	0x22

Ex2) 5bytes의 String data "abcde"를 전송할 경우 (abcde – 0x61 0x62 0x63 0x64 0x65)

16 Bit Data Bus Width (MR(DBW) = '1')	8 Bit Data Bus Width (MR(DBW) = '0')
Sn_TX_FIFOR = 0x6162 Sn_TX_FIFOR = 0x6364 Sn_TX_FIFOR = 0x6500 Sn_TX_WRSR0 = 0x0000 Sn_TX_WRSR1 = 0x0005 Sn_CR = 0x0020 (SEND command)	Sn_TX_FIFOR0 = 0x61 Sn_TX_FIFOR1 = 0x62 Sn_TX_FIFOR0 = 0x63 Sn_TX_FIFOR1 = 0x64 Sn_TX_FIFOR0 = 0x65 Sn_TX_FIFOR1 = 0x00 Sn_TX_WRSR0 = 0x00 Sn_TX_WRSR1 = 0x00 Sn_TX_WRSR2 = 0x00 Sn_TX_WRSR2 = 0x05 Sn_CR1 = 0x20 (SEND command)

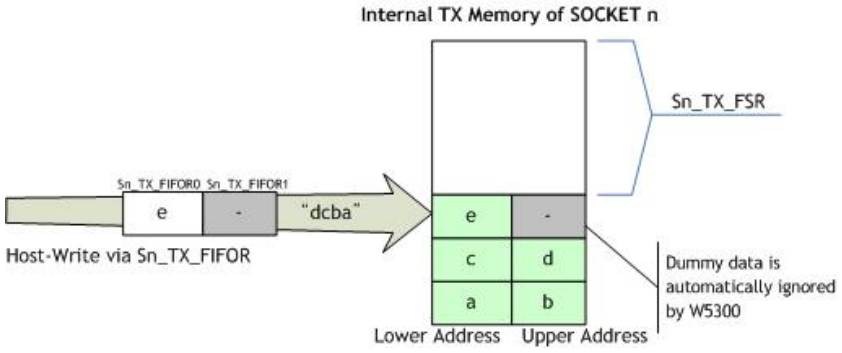


Fig 6. Access to Internal TX Memory

Sn_RX_FIFOR (SOCKETn RX FIFO Register) [R/W] [0x08230+40n/0x230+40n] [0xUUUU]

SOCKETn의 Internal RX memory를 간접적으로 접근한다.

SOCKETn의 Internal RX memory는 Host에 의해 직접적으로 접근될 수 없으며, Sn_RX_FIFOR을 통해서만 접근이 허용된다. MR(MT) = '0' 인 경우 Internal RX memory는 Sn_RX_FIFOR을 통해 Host-Read만 허용된다. MR(MT) = '1' 인 경우 Host-Read와 Host-Write 모두 허용되며, Target host system과 W5300간의 Interface 검증 후에는 반드시 '0'으로 설정한다.("*How to Test Internal TX/RX Memory*" 참조).

Target host system이 8bit data bus width를 사용한다면 Sn_TX_FIFOR과 마찬가지로 Sn_RX_FIFOR0와 Sn_RX_FIFOR1를 한 쌍(pair)으로 접근해야 하고, 또한 Sn_TX_FIFOR0과 Sn_TX_FIFOR1을 접근한 바로 직후 Sn_RX_FIFOR0과 Sn_RX_FIFOR0을 접근할 수 없다. 이럴 경우 Sn_RX_FIFOR0과 Sn_RX_FIFOR1의 값을 제대로 읽을 수가 없다. 이를 방지하기 위해서 Sn_TX_FIFOR0과 Sn_TX_FIFOR1을 읽은 후 Sn_MR와 같은 임의의 Register를 먼저 Host-Read한 다음 Sn_RX_FIFOR을 접근한다.

Internal RX memory에 수신된 DATA packet을 2bytes씩 Sn_RX_FIFOR을 통해 Host-Read 할 경우 Internal RX memory의 Low address와 High address에 위치한 Data는 각각 Sn_RX_FIFOR0와 Sn_RX_FIFOR1를 통해 알 수 있다. Host는 Internal RX memory의 수신된 DATA packet 처리를 완료했을 경우 RECV command를 수행한다.

Internal RX memory에 수신된 Data들은 Sn_MR(P3:P0)에 따라 DATA packet에 대한 PACKET-INFO가 앞에 추가된다. 추가된 PACKET-INFO는 그 DATA packet에 대한 Size를 포함한 기타 정보를 가지고 있으며, Host는 PACKET-INFO를 먼저 처리 한 후 DATA packet을 처리해야 한다. 수신된 DATA packet이 홀수 크기일 경우 1byte dummy data가 추가되며, Host는 이 Dummy data를 읽은 후 무시해야 한다. DATA packet의 마지막 Byte가 Dummy data인지 아닌지는, PACKET-INFO의 Size 정보로 판단할 수 있다.

Host는 Internal RX memory에 수신된 PACKET-INFO와 DATA packet 쌍들을 Sn_RX_FIFOR을 통해 2bytes씩 Host-Read하여 순차적으로 처리한다.

PACKET-INFO는 TCP나 MACRAW mode인 경우 2bytes, UDP mode인 경우 8bytes, IPRAW mode인 6bytes의 고정길이를 갖는다. PACKET-INFO 처리에 대한 자세한 설명은

*"Chapter 5. Functional Description"*의 각 mode 별 설명을 참조하라.

Ex1) Sn_RX_FIFOR = 0x3344

Sn_RX_FIFOR(0x08230+0x40n/0x230+0x040n)	
Sn_RX_FIFOR0(0x08230+0x40n/0x230+0x040n)	Sn_RX_FIFOR1(0x08231+0x40n/0x231+0x040n)
0x33	0x44

Ex2) TCP mode에서 5bytes의 String data "abcde"를 수신하여 "str" 변수에 저장할 경우

16 Bit Data Bus Width (MR(DBW) = '1')	8 Bit Data Bus Width (MR(DBW) = '0')

INT16 pack_size, idx,temp
INT8 str[5]
pack_size = Sn_RX_FIFOR
idx = 0
LOOP pack_size/2
  temp = Sn_RX_FIFOR
  str[idx] = (INT8)(temp >> 8)
  idx = idx + 1
  str[idx] = (INT8)(temp & 0x00FF)
  idx = idx + 1
END LOOP
IF pack_size is odd ? THEN
  temp = Sn_RX_FIFOR
  str[idx] = (INT8)(temp >> 8)
END IF
Sn_CR = 0x0040 (RECV command)
	INT16 pack_size, idx,temp
	INT8 str[5], dummy
	pack_size = Sn_RX_FIFOR0
	pack_size = (pack_size << 8)
	pack_size = pack_size + Sn_RX_FIFOR1
	idx = 0
	LOOP pack_size/2
	  str[idx] = Sn_RX_FIFOR0
	  idx = idx + 1
	  str[idx] = Sn_RX_FIFOR1
	  idx = idx + 1
	END LOOP
	IF pack_size is odd ? THEN
	  str[idx] = Sn_RX_FIFOR0
	  dummy = Sn_RX_FIFOR1
	END IF
	Sn_CR1 = 0x40 (RECV command)

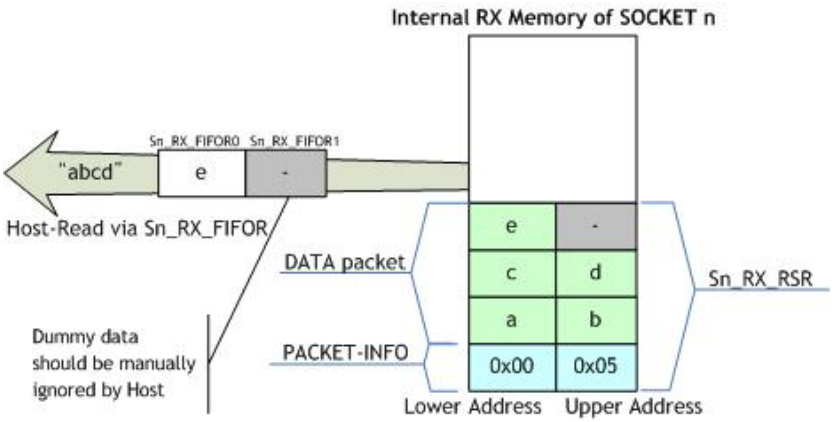


Fig 7. Access to Internal RX Memory

5. Functional Description

W5300은 간단한 Register 조작만으로 Internet connectivity를 제공한다. 이 Chapter에서는 W5300의 초기화와 각 Protocol(TCP, UDP, IPRAW, MACRAW)에 따른 Data 통신방법에 대하여각 단계별로 Pseudo code를 기반으로 살펴 본다.

5.1 Initialization

W5300의 초기화는 Host interface 설정, Network 정보 설정, Internal TX/RX memory 설정과 같이 3단계로 이루어진다.

■ STEP 1 : Setting host interface

1. Data bus width, Host interface mode & timing 설정 (MR 참조)
2. Host interrupt 설정 (IMR 참조)

■ STEP 2 : Setting network information

1. 통신을 위한 기본 Network 정보 설정(SHAR, GAR, SUBR, SIPR 참조)
2. Packet 전송을 실패 시 사용하게 될 재전송 time & count 설정 (RTR, RCR 참조)

SHAR에 의해 설정되는 Source hardware address는 모든 Device에 대해 유일한 Hardware address(Ethernet MAC address)값을 Ethernet MAC layer에서 사용하도록 정해져 있다. 이 MAC address의 할당은 IEEE에서 관장하고 있으며, Network device를 생산하는 Manufacture는 생산된 Network device에 IEEE로부터 할당 받은 MAC address를 부여하여야 한다.

http://www.ieee.org/, http://standards.ieee.org/regauth/oui/index.shtml를 참조하라.

■ STEP 3 : Allocation internal TX/RX memory for SOCKETn

1. Internal TX/RX memory 크기를 각각 결정 (MTYPER 참조)
2. SOCKETn의 TX/RX memory를 각각 결정 (TMSR, RMSR 참조)

W5300은 8Kbytes의 Memory Block 16개를 내부적으로 포함하고 있다. 16개의 Memory Block은 128Kbytes의 Address space에 순차적으로 Mapping되어 있다. 128Kbytes의 Memory는 크게 Transmission(TX) memory, Reception(RX) memory로 구분된다. Internal TX memory와 Internal RX memory는 128Kbytes 범위 내에서 8Kbytes 단위로 할당될 수 있다. Internal TX/RX memory는 각 할당된 크기 내에서 또다시 SOCKET 별로 최소 0Kbyte에서 최대 64Kbytes 내에서 1Kbytes 단위로 각각 할당 될 수 있다. 다음은 예로 Internal TX memory로 72Kbytes, Internal RX memory로 56Kbytes를 각각 할당한 것이다. SOCKET0에서 SOCKET7까지의 TX memory는 72Kbytes 범위 내에서 각각 4, 16, 1, 20, 0, 7, 12, 12KBytes로 할당되고, RX Memory는 56Kbytes 범위 내에서 각각 17, 3, 5, 16, 3, 4, 4, 4Kbytes로 된다. 이때 0Kbyte로 설정된 SOCKET4은 Data 전송이 불가능함에 유의하라.

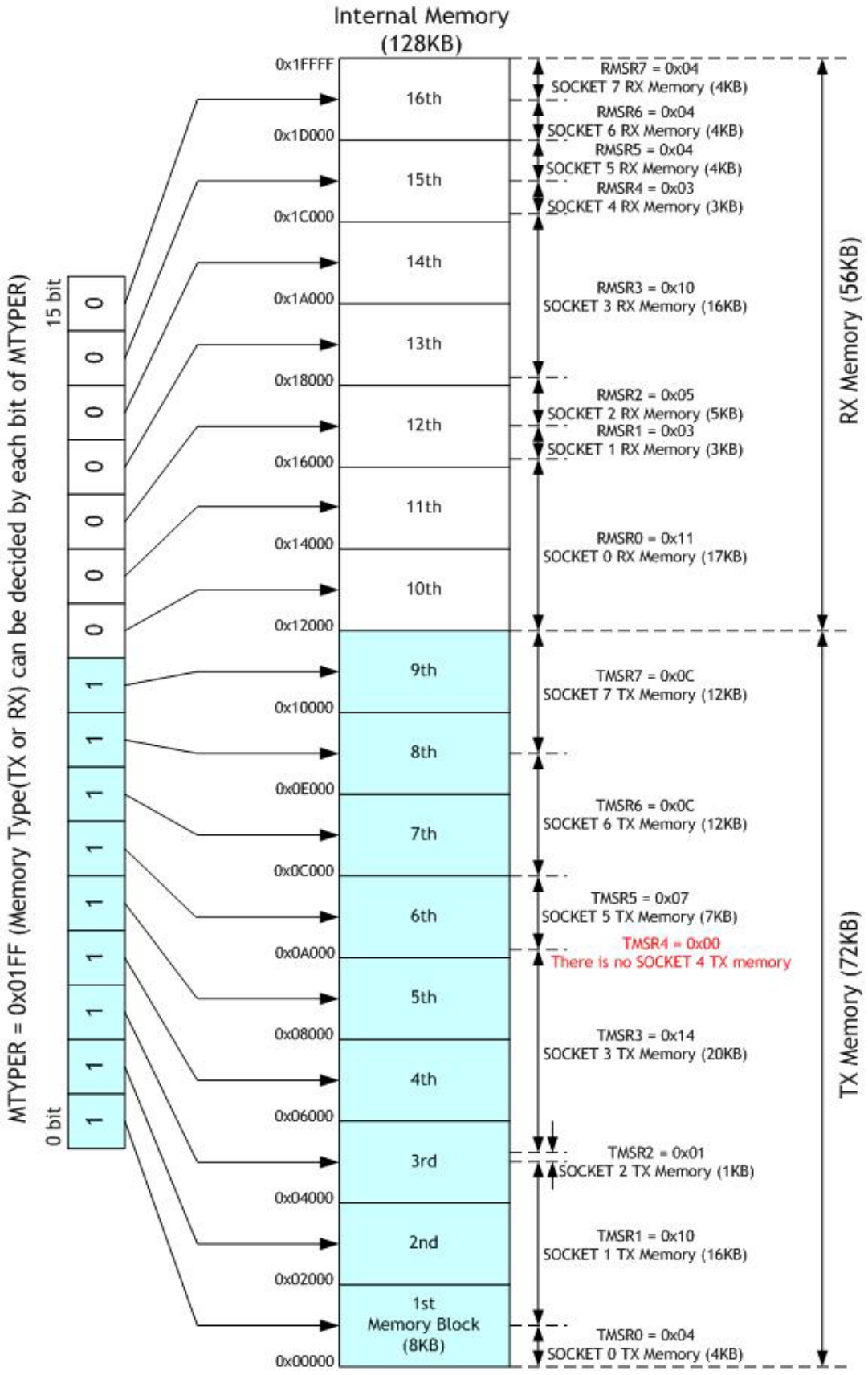


Fig 8. Allocation Internal TX/RX memory of SOCKETn

3단계에 걸친 W5300 initialization 과정을 성공적으로 마쳤다면, W5300은 Ethernet을 통해 Data communication이 가능하다. 이 시점부터 W5300은 Network으로부터 수신한 Ping-request packet에 대한 Ping-reply를 전송할 수 있게 된다(Auto-ping-reply).

5.2 Data Communication

Initialization 과정 후, W5300은 TCP, UDP, IPRAW, MACRAW mode의 SOCKET을 open하여 상대방과 Data를 송수신할 수 있게 된다. W5300은 독립적으로 동시에 사용 가능한 SOCKET을 총 8개까지 지원한다. 이 Chapter에서는 각 Mode에 따른 통신 방법에 대해서 설명한다.

5.2.1 TCP

TCP는 Connection-oriented protocol이다. TCP는 자신의 IP address와 Port number 그리고 상대방의 IP address와 Port number를 한 쌍으로 Connection SOCKET을 형성하게 되고, 형성된 Connection SOCKET을 통해 Data를 송수신한다.

Connection SOCKET의 형성 방법에는 “TCP SERVER”와 “TCP CLIENT” 2가지가 있다. 이는 누가 connect-request(SYN packet)을 전송하느냐에 따라 구분된다.

“TCP SERVER”은 상대방의 connect-request 전송을 대기하며, 전송된 connect-request을 accept하여 Connection SOCKET을 형성하게 된다(Passive-open).

“TCP CLIENT”는 자신이 connect-request를 상대방에게 전송하여 Connection SOCKET 형성을 먼저 요구하게 된다(Active-open).

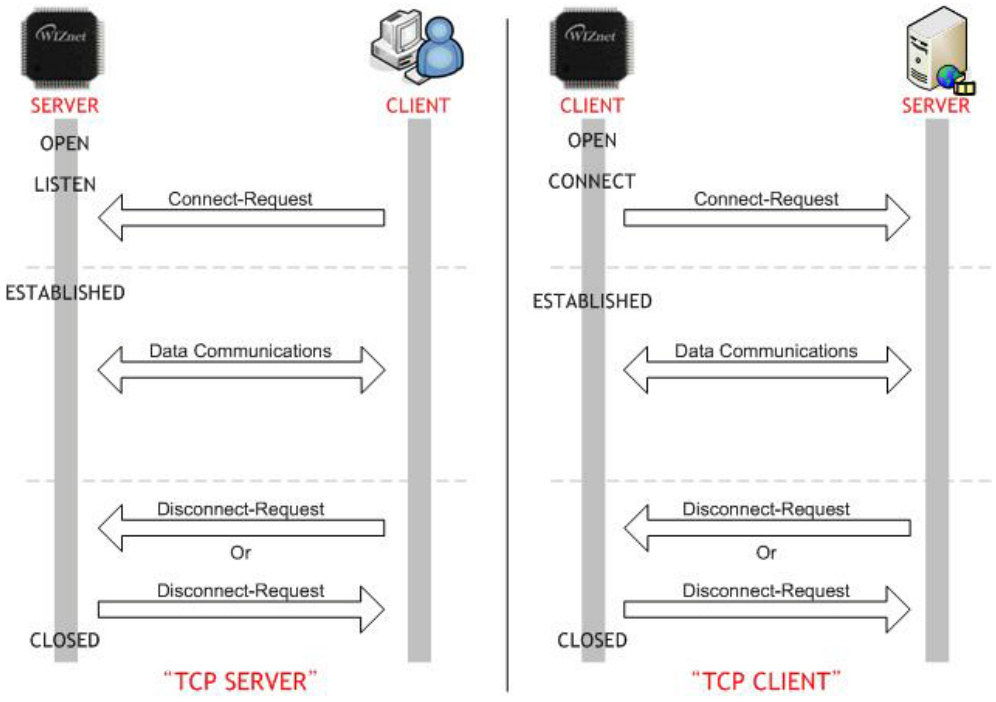


Fig 9. “TCP SERVER” & “TCP CLIENT”

5.2.1.1 TCP SERVER

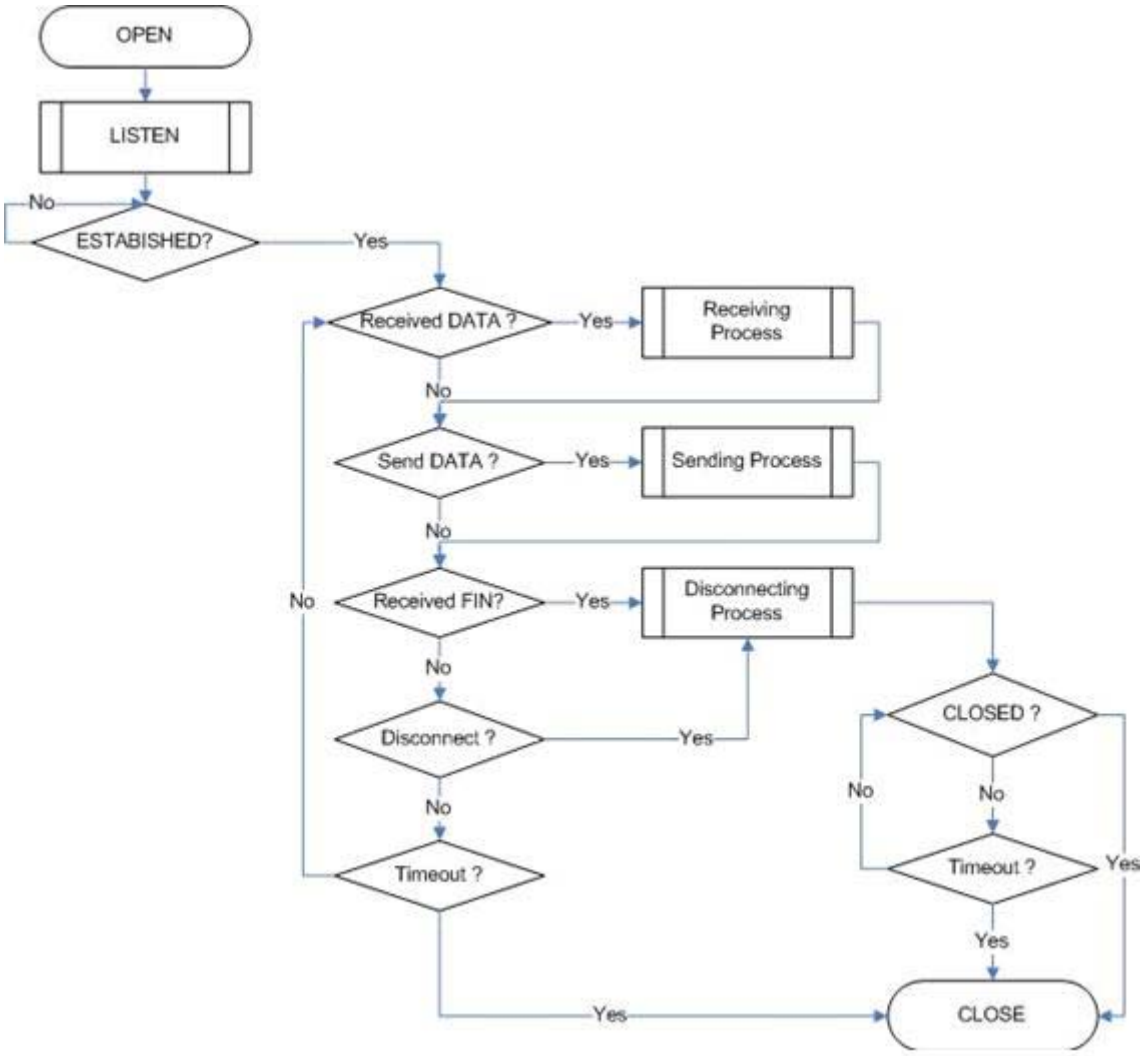


Fig 10. "TCP SERVER" Operation Flow

■ SOCKET Initialization

TCP Data communication을 위해 SOCKET Initialization 과정이 필요하다. 이는 SOCKET을 open하는 일이다. SOCKET open 과정은 W5300의 8개의 SOCKET 중 하나를 선택하여 선택된 SOCKET의 Protocol mode(Sn_MR(P3:P0))와 Source port number("TCP SERVER"에서는 Listen port number라고 함)인 Sn_PORTR을 설정한 후, OPEN command를 수행함으로써 이루어진다. OPEN command 이후 Sn_SSR이 SOCK_INIT으로 변경되면 SOCKET initialization 과정은 완료된다.

SOCKET initialization 과정은 "TCP SEVER"와 "TCP CLIENT"의 구분 없이 동일하게 적용된다. 다음은 SOCKETn을 TCP mode의 Initialization 과정을 보여준다.

```
{
START:
```

```
  Sn_MR = 0x0001;                    /* sets TCP mode */
  Sn_PORTR = source_port;                    /* sets source port number */
  Sn_CR = OPEN;                      /* sets OPEN command */
  /* wait until Sn_SSR is changed to SOCK_INIT */
  if (Sn_SSR != SOCK_INIT) Sn_CR = CLOSE; goto START;
}
```

만약 상대방으로부터 수신한 Data size가 모두 짝수크기로 이루어진다면, Sn_MR(ALIGN)을 '1'로 설정 할 수 있다. Sn_MR(ALIGN) = '1' 인 경우 W5300은 TCP mode의 PACKET-INFO을 추가하지 않고, DATA packet만을 SOCKETn의 Internal RX Memory로 저장하게 된다. 이 방법은 Host의 PACKET-INFO 처리 Overhead를 줄여 수신성능을 향상시킬 수 있다. (위의 Code에서 Sn_MR = 0x0001 대신 Sn_MR = 0x0101를 사용)

■ LISTEN

LISTEN command를 수행하여 "TCP SERVER"로 동작시킨다.

```
{
    /* listen socket */
    Sn_CR = LISTEN;
    /* wait until   Sn_SSR is changed to SOCK_LISTEN */
    if (Sn_SSR != SOCK_LISTEN) Sn_CR = CLOSE; goto START;
}
```

■ ESTABLISHED ?

Sn_SSR이 SOCK_LISTEN일 때 상대방으로부터 SYN packet을 수신하게 되면 Sn_SSR은 SOCK_SYNRECV로 변경되고 SYN/ACK packet을 전송 후 SOCKETn은 Connection을 형성하게 된다. SOCKETn의 Connection이 형성된 이후부터 Data communication은 가능해진다. SOCKETn의 Connection 형성을 확인하는 방법은 2가지가 있다.

```
First method :
{
    if (Sn_IR(CON) == '1')   Sn_IR(CON) = '1'; goto ESTABLISHED stage;
    /* In this case, if the interrupt of SOCKETn is activated, interrupt occurs. Refer to IR, IMR
       Sn_IMR and Sn_IR. */
}
```

```
Second method :
{
    if (Sn_SSR == SOCK_ESTABLISHED) goto ESTABLISHED stage;
}
```

■ ESTABLISHED : Received Data ?

상대방으로부터의 TCP data 수신을 확인한다.

```
First method :
{
    if (Sn_IR(RECV) == '1') Sn_IR(RECV) = '1'; goto Receiving Process stage;
    /* In this case, if the interrupt of SOCKETn is activated, interrupt occurs. Refer to IR, IMR
       Sn_IMR and Sn_IR. */
}
```

```
Second Method :
{
    if (Sn_RX_RSR != 0x00000000) goto Receiving Process stage;
}
```

First method는 매 수신 DATA packet 마다 Sn_IR(RECV)이 '1'로 설정된다. Host가 이전에 수신한 DATA packet의 Sn_IR(RECV)를 미처 처리 못하고 W5300이 다음 DATA packet을 수신할 경우, 이전 Sn_IR(RECV)에 중복 설정되어 Host는 그 다음의 수신 DATA packet에 대한 Sn_IR(RECV)를 인지할 수가 없게 된다. 따라서 Host가 각 Sn_IR(RECV)에 대한 DATA packet을 완벽하게 처리하지 못한다면 이 방법은 권장되지 않는다.

■ ESTABLISHED : Receiving Process

Internal RX memory에 수신된 TCP data를 처리한다. 수신된 TCP data의 구조는 아래와 같다.

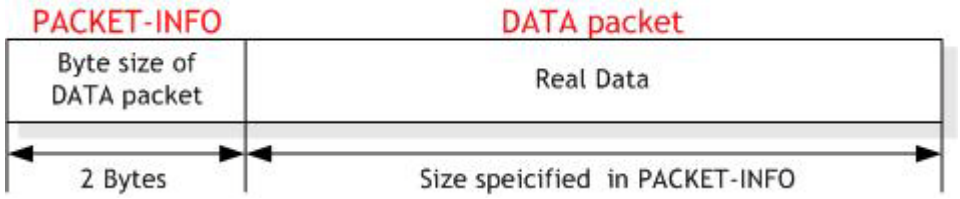


Fig 11. The received TCP data format

수신된 TCP data는 Sn_MR(ALIGN)='0'일 경우 PACKET-INFO와 DATA packet로 이루어지며, Sn_MR(ALIGN)='1'일 경우 PACKET-INFO는 제거되어 DATA packet으로만 이루어진다.

TCP mode에서 상대방이 전송한 Data 크기가 SOCKETn의 RX memory free size보다 클 경우 W5300은 그 Data를 수신할 수 없으며, RX memory free size가 클 때까지 Connection을 유지한 체 기다린다.

```
{
```

```
    /* first, check Sn_MR(ALIGN) */
    if (Sn_MR(ALIGN) == '0')
    {
        pack_size = Sn_RX_FIFOR; /* extract size of DATA packet from internal RX memory */
    }
    else
    {
        pack_size = Sn_RX_RSR; /* check the total received data size */
    }

    /* calculate the read count of Sn_RX_FIFOR */
    if (pack_size is odd ?) read_cnt = (pack_size + 1) / 2;
    read_cnt = pack_size / 2;

    /* extract DATA packet from internal RX memory */
    for( i = 0; i < read_cnt; i++)
    {
        data_buf[i] = Sn_RX_FIFOR; /* data_buf is array of 16bit */
    }
    /* set RECV command */
    Sn_CR = RECV;
}
```

<Notice> SOCKETn을 송신 없이 수신 전용으로 사용하고자 할 경우

Host의 Internal RX memory 처리가 늦어져 Internal RX memory가 Full에 도달할 수 있다. 이럴 경우, W5300의 Window size(TCP에서 수신 가능한 최대 Data 크기)가 0이 아님에도 불구하고 상대방은 W5300의 Window size를 0으로 오인하여, 더 이상 Data를 전송하지 않고 Window size가 증가할 때까지 기다리게 된다. 이런 현상은 W5300의 Data 수신 성능을 급감시키는 원인이 된다. 이를 해결하기 위해서 Internal RX memory에 수신된 Data를 처리한 후, 처리한 수신 Data size만큼 W5300의 Window size가 증가했음을 상대방에게 알려야 한다. 상기 code에서 RECV command 이후 다음과 같은 Code를 추가함으로써 간단히 해결할 수 있다.

```
    /* set RECV command */
    Sn_CR = RECV;
    /* Add the code that notifies the update of window size to the peer */
    /* check the received data process to finish or not */
    if(Sn_RX_RSR == 0) /* send the window-update packet when the window size is full */
```

```
{   /* Sn_RX_RSR can be compared with another value instead of '0',
       according to the host performance of receiving data */
    Sn_TX_WRSR = 0x00000001;      /* set Dummy Data size to Sn_TX_WRSR */
    Sn_TX_FIFOR = 0x0000;         /* Write Dummy Data into TX memory */
    Sn_CR = SEND;                 /* set SEND command */
    while(Sn_CR != 0x00);         /* check SEND command completion */
    while(Sn_IR(SENDOK) == '0');  /* wait for SEND OK */
    Sn_IR(SENDOK) = '1';          /* Clear SENDOK bit */
}
```

■ ESTABLISHED : Send DATA ? / Sending Process

전송할 Data를 Sn_TX_FIFOR을 통해 Internal TX memory에 저장한 후 상대방에게 전송을 시도한다. 전송할 Data 크기는 할당된 SOCKETn의 Internal TX memory보다 클 수 없으며, 전송할 Data 크기가 설정된 MSS보다 클 경우 MSS 단위로 나뉘어져 전송된다. 다음 Data를 전송하기 위해선 반드시 이전의 SEND command가 완료되었는지 확인해야 한다. 이전 SEND command 완료 전에 다시 SEND command를 수행할 경우 오류가 발생할 수 있다. Data의 크기가 클수록 SEND command 완료 시간도 길어지므로, 전송 Data를 적정한 크기로 나누어 전송하는 것이 유리하다.

```
{
    /* first, get the free TX memory size */
FREESIZE:
    get_free_size = Sn_TX_FSR;
    if (Sn_SSR != SOCK_ESTABLISHED && Sn_SSR != SOCK_CLOSE_WAIT) goto CLOSED state;
    if (get_free_size < send_size) goto FREESIZE;

    /* calculate the write count of Sn_TX_FIFOR */
    if (send_size is odd ?) write_cnt = (send_size + 1) / 2;
    else write_cnt = send_size / 2;

    /* copy data to internal TX memory */
    for (i = 0; i < write_cnt; i++)
    {
       Sn_TX_FIFOR = data_buf[i]; /* data_buf is array of 16bit */
    }

    /* check previous SEND command completion */
    if (is first send ?) ; /* skip check Sn_IR(SENDOK) */
```

```
    else
    {
        while(Sn_IR(SENDOK)=='0')
        {
          if(Sn_SSR == SOCK_CLOSED) goto CLOSED state; /* check connection establishment */
        }
        Sn_IR(SENDOK) = '1'; /* clear previous interrupt of SEND completion */
    }

    /* sets transmission data size to Sn_TX_WRSR */
    Sn_TX_WRSR = send_size;

    /* set SEND command */
    Sn_CR = SEND;
}
```

■ ESTABLISHED : Received FIN?

상대방으로부터 Disconnect-request(FIN packet)를 수신했는지 확인한다. FIN packet 수신은 다음과 같이 확인할 수 있다.

```
First method :
{
    if (Sn_IR(DISCON) == '1') Sn_IR(DISCON)='1'; goto CLOSED stage;
    /* In this case, if the interrupt of SOCKETn is activated, interrupt occurs. Refer to IR, IMR
      Sn_IMR and Sn_IR. */
}
```

```
Second method :
{
    if (Sn_SSR == SOCK_CLOSE_WAIT) goto CLOSED stage;
}
```

■ ESTABLISHED : Disconnect ? / Disconnecting Process

더 이상 상대방과의 Data communication이 필요가 없는 경우나 상대방으로부터 FIN packet을 수신했을 경우는 Connection SOCKET을 Disconnect한다.

```
{
    /* set DISCON command */
    Sn_CR = DISCON;
}
```

■ ESTABLISHED : CLOSED ?

DISCON이나 CLOSE command에 의해 SOCKETn이 Disconnect 혹은 Close 되었는지 확인한다.

```
First method :
{
    if (Sn_IR(DISCON) == '1') goto CLOSED stage;
    /* In this case, if the interrupt of SOCKETn is activated, interrupt occurs. Refer to IR, IMR
       Sn_IMR and Sn_IR. */
}
```

```
Second method :
{
    if (Sn_SSR == SOCK_CLOSED) goto CLOSED stage;
}
```

■ ESTABLISHED : Timeout

Timeout은 Connect-request(SYN packet)나 그것에 대한 응답(SYN/ACK packet), DATA packet이나 그것의 응답(DATA/ACK packet), Disconnect-request(FIN packet)나 그것의 응답(FIN/ACK packet)등, 모든 TCP packet을 전송할 때 발생할 수 있다. RTR과 RCR에 설정된 Timeout 시간 동안 상기 packet들을 전송하지 못하면 TCP final timeout(TCP_{TO})이 발생하게 되고 Sn_SSR은 SOCK_CLOSED로 전이한다. TCP_{TO}의 확인은 다음과 같이 할 수 있다.

```
First method :
{
    if (Sn_IR(TIMEOUT bit) == '1') Sn_IR(TIMEOUT)='1'; goto CLOSED stage;
    /* In this case, if the interrupt of SOCKETn is activated, interrupt occurs. Refer to IR, IMR
       Sn_IMR and Sn_IR. */
}
```

```
Second method :
{
    if (Sn_SSR == SOCK_CLOSED) goto CLOSED stage;
}
```

■ SOCKET Close

Disconnect-process에 의해 이미 Disconnection된 SOCKETn이나 TCP_{TO}에 의해 Close된 SOCKETn을 완전히 Close하거나, Host가 Disconnect-process없이 필요에 의해 SOCKETn을 Close할 경우 사용될 수 있다.

```
{
    /* clear the remained interrupts of SOCKETn*/
    Sn_IR = 0x00FF;
    IR(n) = '1';
    /* set CLOSE command */
    Sn_CR = CLOSE;
}
```

5.2.1.2 TCP CLIENT

CONNECT state를 제외한 모든 state는 TCP SEVER와 동일하다. "5.2.1.1 TCP SERVER"를 참조.

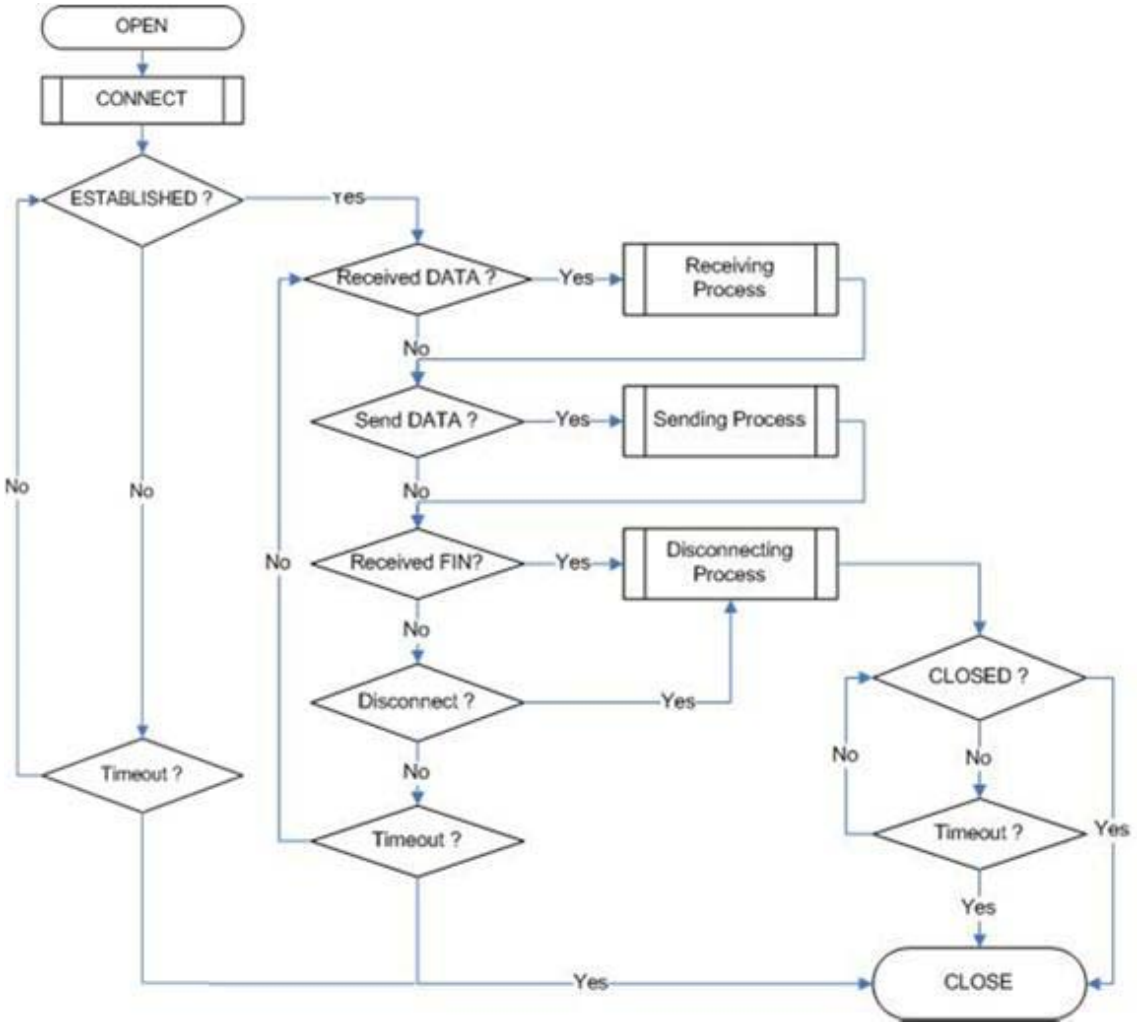


Fig 12. "TCP CLIENT" Operation Flow

■ CONNECT

"TCP SERVER"에게 connect-request(SYN packet)를 전송한다. "TCP SERVER"와의 Connection SOCKET 형성 과정에서 ARP_{TO}, TCP_{TO}와 같은 Timeout이 발생할 수 있다.

```
{
    Sn_DIPR = server_ip;            /* set TCP SERVER IP address*/
    Sn_DPORTR = server_port;        /* set TCP SERVER listen port number*/
    Sn_CR = CONNECT;                /* set CONNECT command */
}
```

5.2.2 UDP

UDP는 Connection-less protocol이다. UDP는 TCP와 달리 Connection SOCKET을 형성하지 않고 Data를 송수신한다. TCP는 신뢰성 있는 Data 통신을 보장하는 반면, UDP는 Data 통신의 신뢰성을 보장하지 않는 Datagram 통신을 하는 protocol이다. UDP는 Connection SOCKET을 사용하지 않기 때문에 자신의 IP address와 Port number를 알고 있는 많은 상대방과의 통신이 허락된다. 이와 같은 Datagram 통신은 하나의 SOCKET을 이용하여 많은 상대방과 통신을 할 수 있는 이점이 있는 반면, 전송된 Data의 손실이나 원치 않는 상대방으로부터의 Data 수신과 같은 여러 문제가 발생할 수 있다. 이와 같은 문제를 해결하고 신뢰성을 보장하기 위해서, Host가 직접 손실된 Data를 재전송하거나, 원치 않는 상대방으로부터의 수신 Data를 무시해야 한다. UDP 통신은 unicast, broadcast, multicast 통신을 지원하며, 다음과 같은 통신 flow를 따른다.

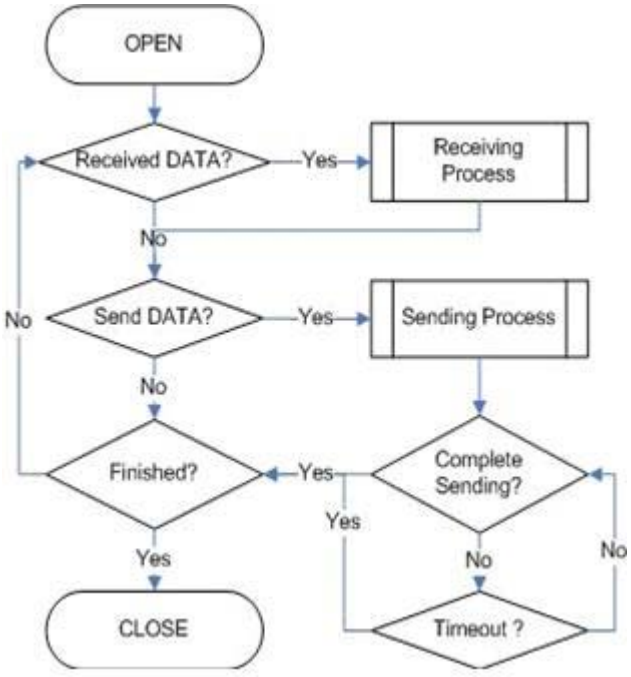


Fig 13. UDP Operation Flow

5.2.2.1 Unicast & Broadcast

Unicast 통신은 가장 일반적인 UDP 통신으로, 한번에 하나의 상대방에게 Data를 전송한다. 반면, Broadcast 통신은 Broadcast IP address(255.255.255.255)을 이용하여 한번의 통신으로 수신 가능한 모든 상대방에게 Data를 전달한다. 예로 A,B,C 에게 Data를 전송할 경우, Unicast 통신은 A, B, C 각각에 대해서 한번씩 Data를 전송을 한다. 이 때 A, B, C에 대한 Destination hardware address를 획득하는 과정(ARP-process)에서 ARP_{TO}가 발생할 수 있으며, ARP_{TO}가 발생한 상대방에게는 Data를 전송할 수가 없다.

Broadcast 통신은 "255.255.255.255" IP address로 한번의 Data 전송을 통하여 A, B, C 모두에게 동시에 Data를 전달한다. 이 때 A, B, C에 대한 Destination hardware address를 획득할 필요가 없으며, ARP_{TO} 역시 발생하지 않는다.

■ SOCKET Initialization

UDP data communication을 위해 SOCKET initialization 과정이 필요하다. 이는 SOCKET을 open하는 일이다. SOCKET open 과정은 W5300의 8개의 SOCKET 중 하나를 선택하고, 선택된 SOCKET의 Protocol mode(Sn_MR(P3:P0))와 상대방과의 통신에 사용할 Source port number인 Sn_PORTR을 설정한 후, OPEN command를 수행한다. OPEN command 이후 Sn_SSR이 SOCK_UDP으로 변경되면 SOCKET initialization 과정은 완료된다.

```
{
START:
  Sn_MR = 0x02;                  /* sets UDP mode */
  Sn_PORTR = source_port;        /* sets source port number */
  Sn_CR = OPEN;                  /* sets OPEN command */
  /* wait until Sn_SSR is changed to SOCK_UDP */
  if (Sn_SSR != SOCK_UDP) Sn_CR = CLOSE; goto START;
}
```

■ Received DATA?

상대방으로부터의 UDP data 수신을 확인한다. TCP 통신과 동일한 방법으로 확인이 가능하다. 물론 TCP와 같은 이유로 Second method를 권장한다. "5.2.1.1 TCP SERVER"의 해당 절을 참조하라.

```
First method :
{
    if (Sn_IR(RECV) == '1') Sn_IR(RECV) = '1'; goto Receiving Process stage;
    /* In this case, if the interrupt of SOCKETn is activated, interrupt occurs. Refer to IR, IMR
       Sn_IMR and Sn_IR. */
}
```

```
Second Method :
{
    if (Sn_RX_RSR != 0x00000000) goto Receiving Process stage;
}
```

■ Receiving Process

Internal RX memory에 수신된 UDP Data를 처리한다. 수신된 UDP data의 구조는 아래와 같다.

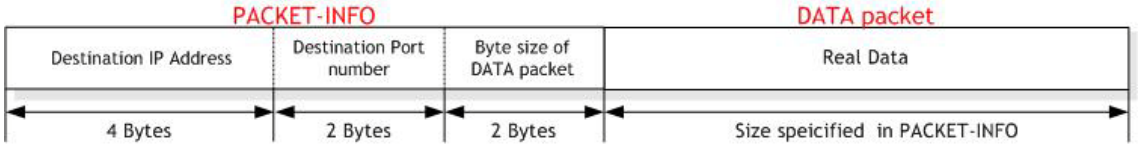


Fig 14. The received UDP data format

수신된 UDP data는 8bytes의 PACKET-INFO와 DATA packet으로 이루어지며, PACKET-INFO는 송신자의 정보(IP address, Port number)와 DATA packet의 길이가 포함된다. UDP는 많은 송신자로부터 UDP data를 수신할 수 가 있다. 송신자의 구분은 PACKET-INFO의 송신자 정보를 통해 확인할 수 있다. 송신자가 “255.255.255.255” IP address를 이용하여 broadcast한 경우도 수신된다. Host는 PACKET-INFO의 송신자 정보를 분석하여 필요 없는 수신 DATA packet은 무시해야 한다.

송신자의 Data 크기가 SOCKETn의 Internal RX memory free size보다 클 경우 그 Data는 수신할 수 없으며, 또한 Fragment된 Data 역시 수신할 수 없다.

```
{
    /* process PACKET-INFO read from internal RX memory */
    temp = Sn_RX_FIFOR;   /* extract destination IP address from internal RX memory */
    dest_ip[0] = ((temp & 0xFF00) >> 8);
    dest_ip[1] = (temp & 0x00FF);
    temp = Sn_RX_FIFOR;
    dest_ip[2] = ((temp & 0xFF00) >> 8);
    dest_ip[3] = (temp & 0x00FF);
    dest_port = Sn_RX_FIFOR; /* extract destination port number from internal RX memory */
    pack_size = Sn_RX_FIFOR; /* extract length of DAT packet from internal RX memory */

    /* calculate the read count of Sn_RX_FIFOR */
    if (pack_size is odd ?) read_cnt = (pack_size + 1) / 2;
    read_cnt = pack_size / 2;
```

```
    for ( i = 0 ; i < read_cnt ; i++ )
    {
        data_buf[i] = Sn_RX_FIFOR;    /* data_buf is array of 16bit */
    }

    /* set RECV command */
    Sn_CR = RECV;
}
```

■ Send Data? / Sending Process

상대방의 IP address와 Port number를 설정하고 전송할 Data를 Sn_TX_FIFOR을 통해 Internal TX memory에 저장한 후 상대방에게 전송을 시도한다. 전송할 Data크기는 할당된 SOCKETn의 Internal TX memory보다 클 수 없으며, 전송할 Data 크기가 MTU보다 클 경우 MTU 단위로 자동으로 나누어져 전송된다.

Broadcast할 경우 Sn_DIPR을 "255.255.255.255"로 설정한다.

```
{
    /* first, get the free TX memory size */
FREESIZE:
    get_free_size = Sn_TX_FSR;
    if (get_free_size < send_size) goto FREESIZE;

    /* Set the destination information */
    Sn_DIPR0 = dest_ip[0]; //or 255; /* Set the 4 bytes destination IP address to Sn_DIPR */
    Sn_DIPR1 = dest_ip[1]; //or 255;
    Sn_DIPR2 = dest_ip[2]; //or 255;
    Sn_DIPR3 = dest_ip[3]; //or 255;
    Sn_DPORTR = dest_port; /* Set the 2 bytes destination port number to Sn_DPORTR */

    /* calculate the write count of Sn_TX_FIFOR */
    if (send_size is odd ?) write_cnt = (send_size + 1) / 2;
    else write_cnt = send_size / 2;

    /* copy data to internal TX memory */
    for (i = 0; i < write_cnt; i++)
    {
       Sn_TX_FIFOR = data_buf[i]; /* data_buf is array of 16bit */
    }
```

```
    /* sets transmission data size to Sn_TX_WRSR */
    Sn_TX_WRSR = send_size;

    /* set SEND command */
    Sn_CR = SEND;
}
```

■ Complete Sending? & Timeout

다음 Data를 전송하기 위해선 반드시 이전 SEND command가 완료되었는지 확인해야 한다. Data의 크기가 클수록 SEND command 완료 시간도 길어지므로, 전송 Data를 적정한 크기로 나누어 전송하는 것이 유리하다. UDP data 전송 시 ARP_{TO}가 발생할 수 있고, ARP_{TO}가 발생할 경우 UDP data 전송은 실패한다.

```
{
  /* check SEND command completion */
  while(Sn_IR(SENDOK)=='0')   /* wait interrupt of SEND completion */
  {
    /* check ARPTO */
    if (Sn_IR(TIMEOUT)=='1') Sn_IR(TIMEOUT)='1'; goto Next stage;
  }
  Sn_IR(SENDOK) = '1'; /* clear previous interrupt of SEND completion */
}
```

■ Finished? / Socket Close

더 이상 통신이 필요 없을 경우 SOCKETn을 close한다.

```
{
    /* clear remained interrupts */
    Sn_IR = 0x00FF;
    IR(n) = '1';
    /* set CLOSE command */
    Sn_CR = CLOSE;
}
```

5.2.2.2 Multicast

Broadcast 통신이 불특정 다수와 통신하는 반면, Multicast 통신은 특정 Multicast-group에 등록된 다수와 통신을 한다. A, B, C가 특정 Multicast-group에 등록되어 있고, A가 등록된

Mutlicast-group으로 Data를 전송할 경우 B, C 역시 A의 전송 Data를 수신할 수 있다. Multicast 통신을 하기 위해선 IGMP protocol을 이용하여 Multicast-group에 등록하여야 한다. Multicast-group은 Group hardware address, Group IP address, Group port number로 구분된다. Group hardware address와 IP address는 이미 지정되어 있는 Address를 사용하고, Group port number는 임의로 사용할 수 있다.

Group hardware address는 지정 범위("01:00:5e:00:00:00"에서부터 "01:00:5e:7f:ff:ff") 내에서 선택되며, Group IP address는 D-class IP address 범위("224.0.0.0"에서 "239.255.255.255"까지, http://www.iana.org/assignments/multicast-addresses참조)내에서 선택된다. 이때 6bytes의 Group hardware address와 4bytes의 IP address의 하위 23bit는 같도록 선택해야 한다. 예로, Group IP address를 "224.1.1.11"로 선택할 경우 Group hardware address는 "01:00:5e:01:01:0b"로 선택된다.

"RFC1112" 참조(http://www.ietf.org/rfc.html).

W5300에서는 Multicast-group 등록에 필요한 IGMP 처리는 내부적으로(Automatically) 이루어진다. SOCKETn을 Multicast mode로 Open할 경우 IGMP의 "Join" message, Close할 경우 "Leave" message가 내부적으로 전송된다. SOCKET open 이후 통신 시 주기적으로 "Report" message가 내부적으로 전송된다.

W5300은 IGMP version 1과 version 2만을 지원하며 상위 version을 사용하고자 한다면, IPRAW mode SOCKET을 이용하여 Host가 직접 IGMP를 처리해야 한다.

■ SOCKET Initialization

Multicast 통신을 위해 8개의 SOCKET 중 하나를 선택하고, Sn_DHAR을 Multicast-group hardware address로 Sn_DIPR을 Multicast-group IP address로 설정한다. Sn_PORTR과 Sn_DPORTR을 Multicast-group port number로 설정한다. Sn_MR(P3:P0)를 UDP로 Sn_MR (MULTI)를 '1'로 설정한 후 OPEN command를 수행한다. OPEN command 이후 Sn_SSR이 SOCK_UDP으로 변경되면 SOCKET initialization 과정은 완료된다.

```
{
START:
    /* set Multicast-Group information */
    Sn_DHAR0 = 0x01;        /* set Multicast-Group H/W address(01:00:5e:01:01:0b) */
    Sn_DHAR1 = 0x00;
    Sn_DHAR2 = 0x5E;
    Sn_DHAR3 = 0x01;
    Sn_DHAR4 = 0x01;
    Sn_DHAR5 = 0x0B;
    Sn_DIPR0 = 211;         /* set Multicast-Group IP address(211.1.1.11) */
```

```
    Sn_DIPR1 = 1;
    Sn_DIPR2 = 1;
    Sn_DIRP3 = 11;
    Sn_DPORTR = 0x0BB8;   /* set Multicast-Group Port number(3000) */
    Sn_PORTR = 0x0BB8;    /* set Source Port number(3000) */
    Sn_MR = 0x0002 | 0x0080; /* set UDP mode & Multicast on SOCKETn Mode Register */

    Sn_CR = OPEN;         /* set OPEN command */

    /* wait until Sn_SSR is changed to SOCK_UDP */
    if (Sn_SSR != SOCK_UDP) Sn_CR = CLOSE; goto START;
}
```

- Received DATA?
- Receiving Process

"5.2.2.1 Unicast & Broadcast" 참조.

- Send Data? / Sending Process

SOCKET initialization에서 이미 Multicast-group에 대한 정보를 설정하였으므로, Unicast통신처럼 상대방의 IP address와 Port number를 설정할 필요가 없다. 따라서 전송할 Data를 internal TX memory로 copy한 후 SEND command를 수행한다.

```
{
    /* first, get the free TX memory size */
FREESIZE:
    get_free_size = Sn_TX_FSR;
    if (get_free_size < send_size) goto FREESIZE;

    /* calculate the write count of Sn_TX_FIFOR */
    if (send_size is odd ?) write_cnt = (send_size + 1) / 2;
    else write_cnt = send_size / 2;

    /* copy data to internal TX memory */
    for (i = 0; i < write_cnt; i++)
    {
        Sn_TX_FIFOR = data_buf[i]; /* data_buf is array of 16bit */
    }
```

```
    /* sets transmission data size to Sn_TX_WRSR */
    Sn_TX_WRSR = send_size;

    /* set SEND command */
    Sn_CR = SEND;
}
```

■ Complete Sending? & Timeout

이미 설정된 Multicast-group과의 통신이므로, ARP-process가 필요 없고 ARP_{TO}는 발생하지 않지 않는다.

```
{
  /* check SEND command completion */
  while(Sn_IR(SENDOK)=='0');    /* wait interrupt of SEND completion */
  Sn_IR(SENDOK) = '1';          /* clear previous interrupt of SEND completion */
}
```

■ Finished? / Socket Close

"5.2.2.1 Unicast & Broadcast" 참조.

5.2.3 IPRAW

IPRAW는 TCP와 UDP의 하위 protocol 계층인 IP layer를 이용한 Data 통신이다. IPRAW는 protocol number에 따라 ICMP(0x01), IGMP(0x02)와 같은 IP layer의 protocol을 지원한다. ICMP의 ping이나 IGMP v1/v2는 W5300에서 Hardware logic으로 이미 구현되어있다. 하지만 필요에 따라 Host는 SOCKETn을 IPRAW mode로 open하여 이를 직접 구현하여 처리할 수 있다. IPRAW mode SOCKET을 사용할 경우, 어떤 protocol을 사용할지 반드시 IP header의 protocol number field를 설정하여야 한다. Protocol number는 IANA에 의해 이미 정의되어 있다. http://www.iana.org/assignments/protocol-numbers 참조. Protocol number는 SOCKET Open 이전에 Sn_PROTOR에 반드시 설정한다. W5300은 IPRAW mode에서 TCP(0x06)나 UDP(0x11) protocol number는 지원하지 않는다. IPRAW mode SOCKET의 통신은 지정된 protocol number만의 통신을 허용한다. ICMP로 설정된 SOCKET은 IGMP와 같이 설정되지 않은 그 외의 Protocol Data를 수신할 수 없다.

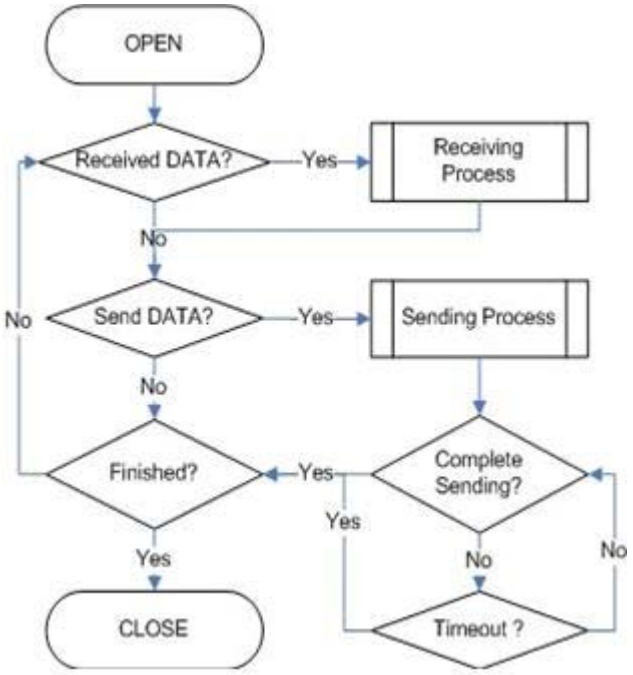


Fig 15. IPRAW Operation Flow

■ SOCKET Initialization

SOCKET을 선택하고 Protocol number를 설정한다. Sn_MR(P3:P0)를 IPRAW mode로 설정하고 OPEN command를 수행한다. OPEN command 이후 Sn_SSR이 SOCK_IPRAW로 변경되면 SOCKET initialization 과정은 완료된다.

```
{
START:
    /* sets Protocol number */
    /* The protocol number is used in Protocol Field of IP Header. */
    Sn_PROTOR = protocol_num;
    /* sets IP raw mode */
    Sn_MR = 0x03;
    /* sets OPEN command */
    Sn_CR = OPEN;
    /* wait until Sn_SSR is changed to SOCK_IPRAW */
    if (Sn_SSR != SOCK_IPRAW) Sn_CR = CLOSE; goto START;
}
```

■ Received DATA?

"5.2.2.1 Unicast & Broadcast" 참조.

■ Receiving Process

Internal RX Memory에 수신된 IPRAW Data를 처리한다. 수신된 IPRAW Data의 구조는 아래와 같다.

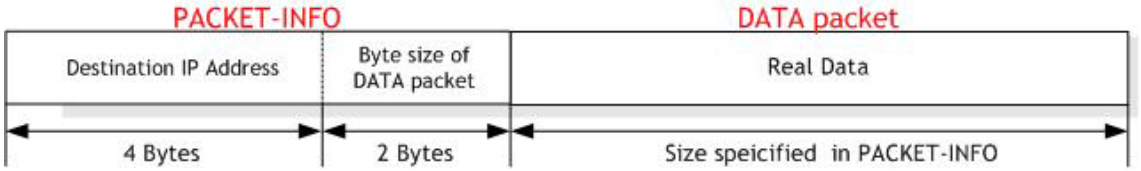


Fig 16. The received IPRAW data format

IPRAW Data는 6 bytes의 PACKET-INFO와 DATA packet으로 이루어지며, PACKET-INFO는 송신자의 정보(IP address)와 DATA packet의 길이가 포함된다. IPRAW mode의 Data 수신은 UDP의 PACKET-INFO에서 송신자의 Port number 처리를 제외하고는 UDP data 수신과 모두 동일하다. "5.2.2.1 Unicast & Broadcast" 참조.

송신자의 Data 크기가 SOCKETn의 RX memory free size보다 클 경우 그 Data는 수신할 수 없으며, 또한 Fragmented data 역시 수신할 없다.

- Send DATA? / Sending Process

전송할 Data 크기는 할당된 SOCKETn의 Internal TX memory보다 클 수 없고, Default MTU보다 클 수 없다. IPRAW data 전송은 UDP data 전송에서 Destination port number를 설정하는 것을 제외하고 모두 동일하다. "5.2.2.1 Unicast & Broadcast" 참조.

- Complete Sending & Timeout
- Finished? / Socket Closed

UDP 통신과 동일하다. "5.2.2 UDP" 참조.

5.2.4 MACRAW

MACRAW 통신은 Ethernet MAC을 기반으로 그 상위 Protocol을 Host가 목적에 맞게 유연하게 사용할 수 있도록 하는 통신방법이다.

MACRAW mode는 오직 SOCKET0만 사용 가능하다. SOCKET0을 MACRAW로 사용할 경우 SOCKET1에서 7까지는 Hardwired TCP/IP stack을 그대로 사용할 뿐만 아니라, SOCKET0을 마치 NIC(Network Interface Controller)처럼 사용할 수 있어 Software TCP/IP stack을 구현할 수 있다. 이와 같이 W5300은 Hardwired TCP/IP와 Software TCP/IP를 모두 구현할 수 있는 Hybrid TCP/IP stack을 지원한다. W5300이 지원하는 8개의 SOCKET보다 더 많은 SOCKET들이 요구될 경우, 높은 성능을 요구하는 SOCKET들은 Hardwired TCP/IP Stack으로 구현하고, 그 외는 MACRAW mode를 이용하여 Software TCP/IP로 구현하여 SOCKET 수의 한계를 극복할 수 있다. MACRAW mode의 SOCKET0은 SOCKET1에서 7까지 사용되고 있는 protocol들을 제외한 모든 protocol를 처리할 수 있다. MACRAW 통신은 아무런 처리 없이 순수 Ethernet packet만의 통신이므로 MACRAW 설계자는 이러한 protocol을 분석하고 처리할 있는 Software TCP/IP stack를 직접 구현해야 한다. MACRAW data는 Ethernet

MAC을 기반으로 하기 때문에 6bytes의 Source hardware address, 6bytes의 destination hardware address, 2 bytes의 Ethernet type 총 14bytes을 기본적으로 포함해야 한다.

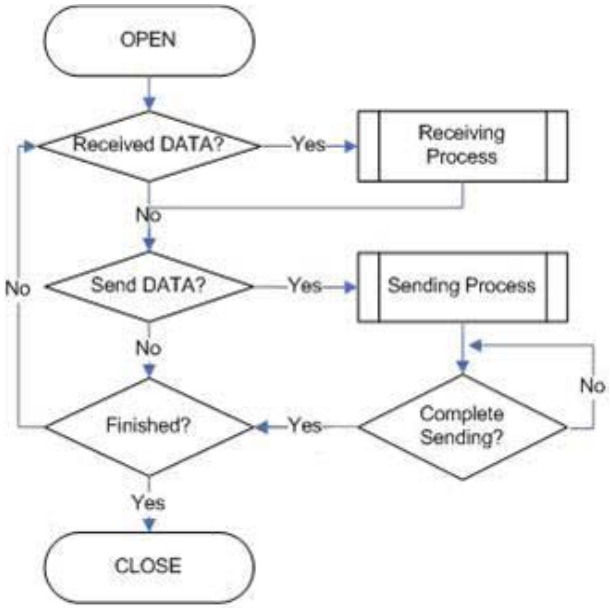


Fig 17. MACRAW Operation Flow

■ SOCKET Initialization

SOCKET을 선택하고 Sn_MR(P3:P0)를 MACRAW mode로 설정한 후 OPEN command를 수행한다. OPEN command 이후 Sn_SSR이 SOCK_MACRAW로 변경되면 SOCKET initialization 과정은 완료된다. 이때 통신에 필요한 모든 정보(Source hardware address, Source IP address, Source port number, Destination hardware address, Destination IP address, Destination port number, 각종 Protocol header, ETC)는 MACRAW Data의 일부분이므로 이와 관련된 Register 설정은 필요 없다.

```
{
START:
    /* sets MAC raw mode */
    S0_MR = 0x04;
    /* sets OPEN command */
    S0_CR = OPEN;
    /* wait until Sn_SSR is changed to SOCK_MACRAW */
    if (Sn_SSR != SOCK_MACRAW) S0_CR = CLOSE; goto START;
}
```

■ Received DATA?

“5.2.2.1 Unicast & Broadcast” 참조.

■ Receiving Process

SOCKET0의 Internal RX Memory에 수신된 MACRAW Data를 처리한다. MACRAW Data의 구조는 아래와 같다.

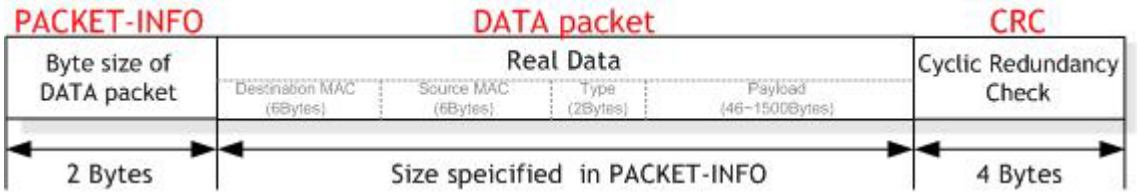


Fig 18. The received MACRAW data format

MACRAW data는 2 bytes의 PACKET-INFO, DATA packet, 4bytes의 CRC로 이루어진다. PACKET-INFO는 DATA packet의 길이이며, DATA packet은 6bytes destination MAC address, 6bytes source MAC address, 2bytes type, 46~1500 bytes payload로 이루어진다. DATA packet의 Payload는 Type에 따라 ARP, IP와 같은 Internet protocol로 이루어진다. Type에 관한 정보는 http://www.iana.org/assignments/ethernet-numbers 를 참조하라.

MACRAW data의 CRC는 반드시 S0_RX_FIFOR을 통해 Host-Read한 후 무시해야 한다.

```
{
  /* extract size of DATA packet from internal RX memory */
  pack_size = S0_RX_FIFOR;

  /* calculate the read count of Sn_RX_FIFOR */
  if (pack_size is odd ?) read_cnt = (pack_size + 1) / 2;
  read_cnt = pack_size / 2;

  /* extract DATA packet from internal RX memory */
  for( i = 0; i < read_cnt; i++)
  {
    data_buf[i] = S0_RX_FIFOR; /* data_buf is array of 16bit */
  }
  /* extract 4 bytes CRC from internal RX memory and then ignore it */
  dummy = S0_RX_FIFOR;
  dummy = S0_RX_FIFOR;
  /* set RECV command */
  S0_CR = RECV;
}
```

<Notice>

Internal RX memory의 Free size가 W5300이 수신해야 할 MACRAW data의 크기보다 작을 경우, 실제 수신되어선 안될 그 MACRAW data의 PACKET-INFO와 DATA packet의 일부가 Internal RX memory에 저장되는 문제가 간혹 발생할 수 있다. 이는 상기 Sample code에서 PACKET-INFO 분석의 오류를 야기시켜 올바른 MACRAW data 수신 처리를 할 수 없게 된다. 이 문제는 Internal RX memory가 Full에 가까울수록 발생할 확률이 높아진다. 이 문제는 MACRAW data의 소실을 어느 정도 감안한다면 해결될 수 있다.

해결 방법은,

- Internal RX memory의 처리를 최대한 빨리 하여 Full에 도달하는 것을 방지한다.
- 자신에 해당하는 MACRAW data만을 수신하도록 하여 수신부하를 줄인다.

  SOCKET Initialization 과정의 Sample code에서 S0_MR의 MF bit를 설정한다.

```
{
START:
    /* sets MAC raw mode with enabling MAC filter */
    S0_MR = 0x44;
    /* sets OPEN command */
    S0_CR = OPEN;
    /* wait until Sn_SSR is changed to SOCK_MACRAW */
    if (Sn_SSR != SOCK_MACRAW) S0_CR = CLOSE; goto START;
}
```

- Internal RX memory의 Free size가 1528 – Default MTU(1514)+PACKET-INFO(2) + DATA packet(8) + CRC(4) – 보다 작을 경우 SOCKET0을 Close한 후 지금까지 수신한 모든 MACRAW data를 처리하고 다시 SOCKET0을 Open하여 정상 처리한다. 이때 SOCKET0 Close이후 수신되는 MACRAW data는 소실될 수 있다.

```
{
  /* check the free size of internal RX memory */
  if((RMSR0 * 1024) - Sn_RX_RSR < 1528)
  {
      recved_size = Sn_RX_RSR;      /* backup Sn_RX_RSR */
      Sn_CR = CLOSE;                /* SOCKET0 Closed */
      while(Sn_SSR != SOCK_CLOSED);         /* wait until SOCKET0 is closed */
      /* process all data remained in internal RX memory */
      while(recved_size > 0)
      {
          /* extract size of DATA packet from internal RX memory */
          pack_size = S0_RX_FIFOR;
```

```
            /* calculate the read count of Sn_RX_FIFOR */
            if (pack_size is odd ?) read_cnt = (pack_size + 1) / 2;
            read_cnt = pack_size / 2;
            /* extract DATA packet from internal RX memory */
            for( i = 0; i < read_cnt; i++)
            {
                data_buf[i] = S0_RX_FIFOR; /* data_buf is array of 16bit */
            }
            /* extract 4 bytes CRC from internal RX memory and then ignore it */
            dummy = S0_RX_FIFOR;
            dummy = S0_RX_FIFOR;
            /* calculate the size of remained data in internal RX memory*/
            recved_size = recved_size - 2 - pack_size - 4;
        }
        /* Reopen the SOCKET0 */
        /* sets MAC raw mode with enabling MAC filter */
        S0_MR = 0x44; /* or S0_MR = 0x04 */
        /* sets OPEN command */
        S0_CR = OPEN;
        /* wait until Sn_SSR is changed to SOCK_MACRAW */
        while (Sn_SSR != SOCK_MACRAW);
    }
    else /* process normally the DATA packet from internal RX memory */
    {
        /* This block is same as the code of "Receiving process" stage*/
    }
}
```

■ Send DATA? / Sending Process

전송할 Data 크기는 할당된 SOCKET0의 Internal TX memory보다 클 수 없고, 또한 Default MTU보다 클 수 없다. Host는 "Receiving process" 절의 DATA packet 형식과 동일한 MACRAW data를 생성하고 그 Data를 전송한다. 이 때 생성된 Data의 크기가 60 bytes 미만일 경우 실제 Ethernet으로 전송되는 Packet은 내부적으로 60bytes가 되도록 "Zero padding"되어 전송된다.

```
{
    /* first, get the free TX memory size */
FREESIZE:
```

```
    get_free_size = S0_TX_FSR;
    if (get_free_size < send_size) goto FREESIZE;

    /* calculate the write count of Sn_TX_FIFOR */
    if (send_size is odd ?) write_cnt = (send_size + 1) / 2;
    else write_cnt = send_size / 2;

    /* copy data to internal TX memory */
    for (i = 0; i < write_cnt; i++)
    {
        S0_TX_FIFOR = data_buf[i]; /* data_buf is array of 16bit */
    }

    /* sets transmission data size to Sn_TX_WRSR */
    S0_TX_WRSR = send_size;

    /* set SEND command */
    S0_CR = SEND;
}
```

■ Complete Sending?

Data 통신에 필요한 모든 Protocol 처리는 Host가 관장하므로 Timeout은 발생하지 않지 않는다.

```
{
  /* check SEND command completion */
  while(S0_IR(SENDOK)=='0');    /* wait interrupt of SEND completion */
  S0_IR(SENDOK) = '1';          /* clear previous interrupt of SEND completion */
}
```

■ Finished? / Socket Close

"5.2.2.1 Unicast & Broadcast" 참조.

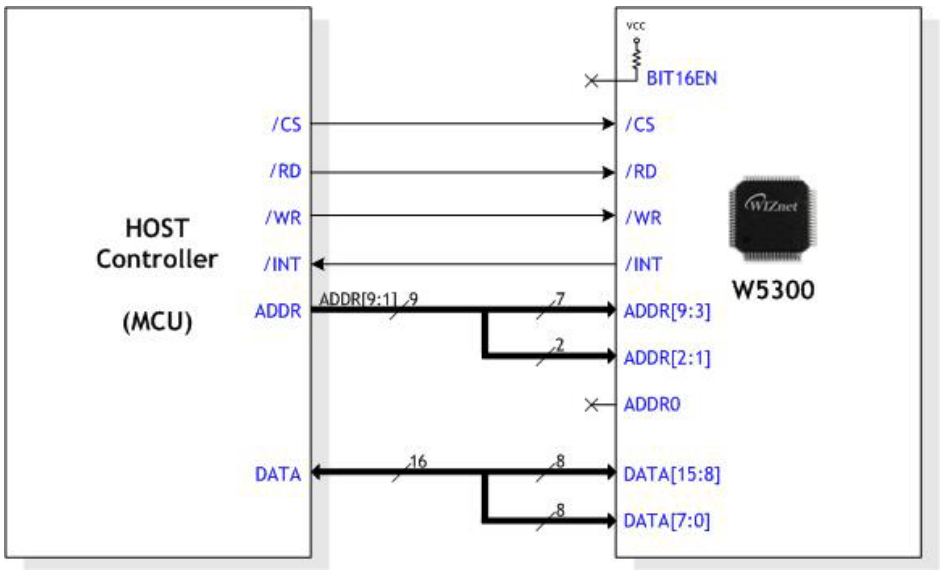
6. External Interface

W5300의 Host interface는 Direct/Indirect address mode와 16/8 bit data bus width에 따라 결정된다. 또한 PIN TEST_MODE[3:0] 설정에 따라 W5300은 Internal PHY 혹은 External PHY와 Interface 된다.

6.1 Direct Address Mode

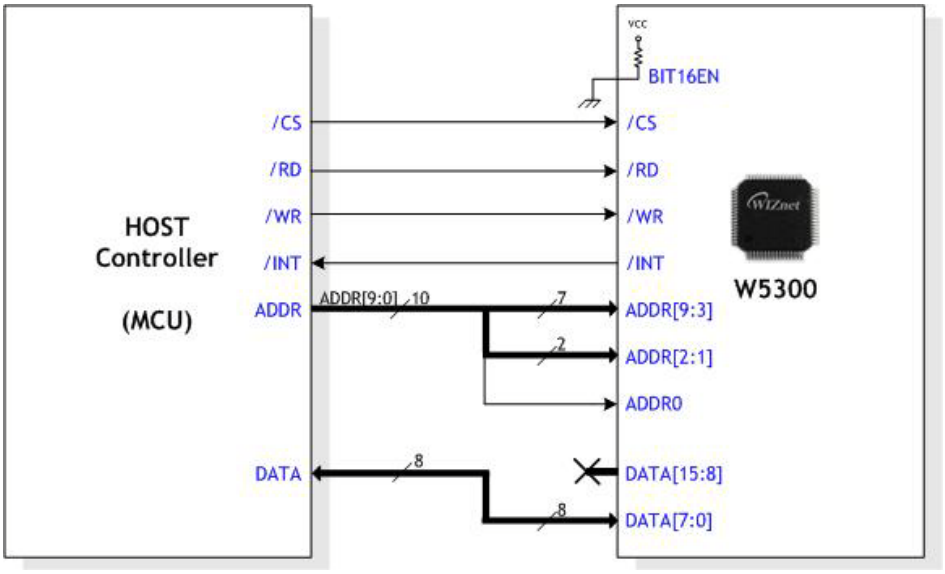
6.1.1 16 Bit Data Bus Width

16bit data bus width를 사용할 경우, ADDR[9:1]만 사용되며, ADDR0은 Float나 Ground 처리한다. 'BIT16EN'은 내부적으로 Pull-up 처리되어 있어 Float 시켜도 무방하다.



6.1.2 8 Bit Data Bus Width

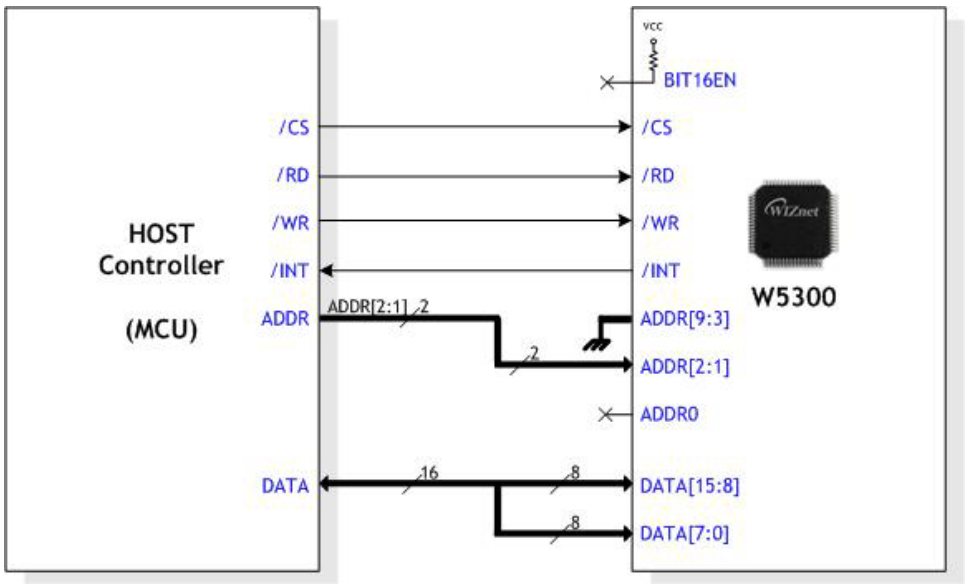
8bit data bus width를 사용할 경우, ADDR[9:0] 모두 사용되며, 'BIT16EN'은 반드시 Logical LOW(Ground) 처리한다. 사용하지 않는 DATA[15:8]은 모두 Float 시킨다.



6.2 Indirect Address Mode

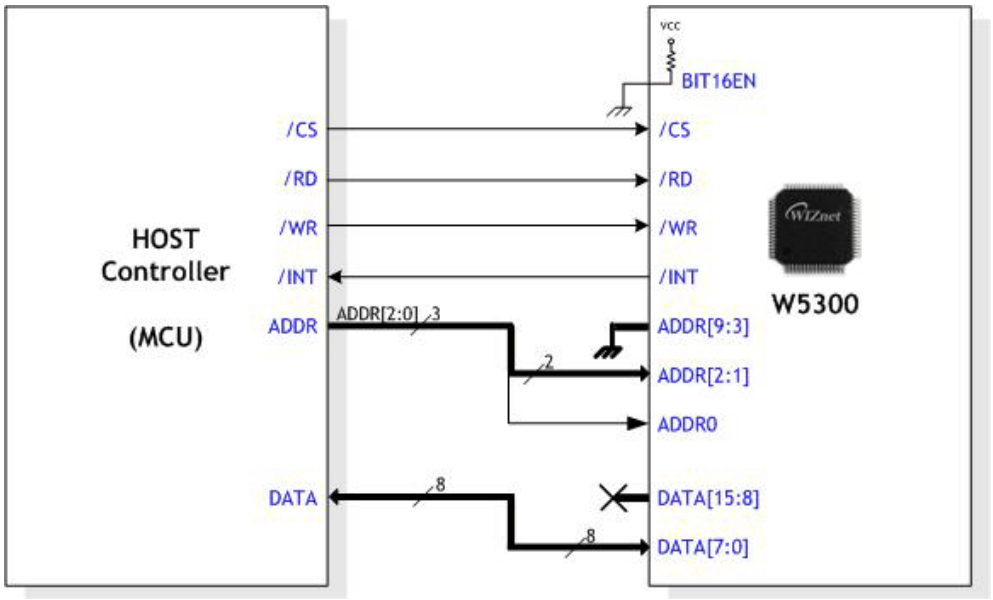
6.2.1 16 Bit Data Bus Width

16bit data bus width를 사용할 경우, ADDR[2:1]만 사용되며, ADDR[9:3]은 반드시 Ground 처리하며, ADDR0은 float시켜도 무방하다. 'BIT16EN'은 내부적으로 Pull-up 처리되어 있어 Float 시켜도 무방하다.



6.2.2 8 Bit Data Bus Width

8bit data bus width를 사용할 경우, ADDR[2:0]만 사용되며, ADDR[9:3]은 반드시 Ground 처리한다. 'BIT16EN'은 반드시 Ground 처리한다. 사용하지 않는 DATA[15:8]은 모두 Float 시킨다.



6.3 Internal PHY Mode

W5300의 Internal PHY를 사용하는 Mode로 TEST_MODE[3:0]은 Float 시키거나, Ground 처리한다. OP_MODE[2:0]은 Internal PHY의 동작을 설정하는 PIN으로, 각 PIN의 Logical value에 따라 Internal PHY의 동작 Mode가 결정된다. "1.1 Configuration Signals" 참조.

Internal PHY와 Transformer간의 Interface에서 보다 좋은 Impedance matching을 위한 Termination resistor와 Capacitor가 필요하다. Resister는 50ohm(오차1%), Capacitor는 0.1uF을 사용한다.

Internal PHY는 LINK, SPEED와 같은 6가지의 Network indicator LED를 지원한다. 사용하지 않는 Network indicator LED는 float 시킨다. /RXLED와 /TXLED를 Logical AND로 묶어 사용하면 ACT LED(Active LED)를 구현할 수 있다. "1.6 Network Indicator LED Signals" 참조.

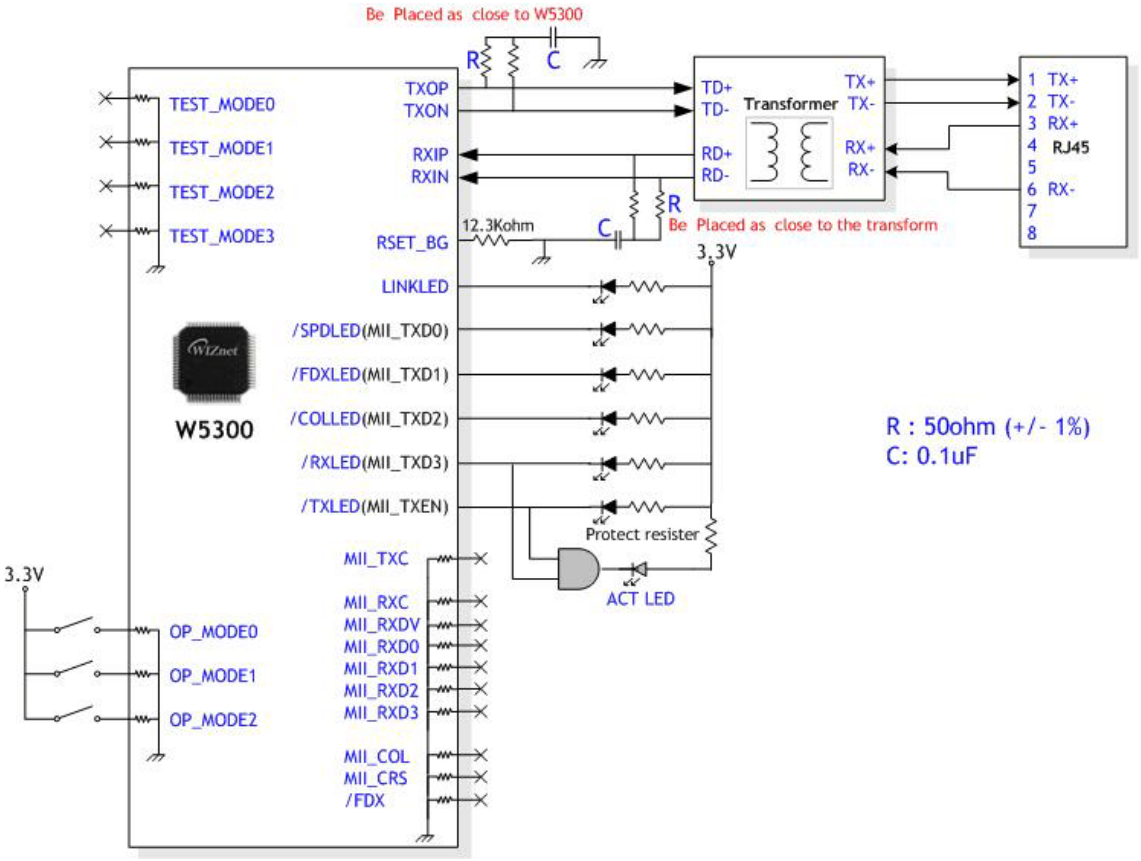


Fig 19. Internal PHY & LED Signals

6.4 External PHY Mode

W5300의 Internal PHY 특성이 맞지 않을 경우 External PHY를 사용할 수 있다. External PHY를 사용할 경우, W5300의 Clock source를 결정해주어야 한다. TEST_MODE0이 Logical HIGH일 경우 Crystal을, TEST_MODE1이 Logical HIGH일 경우 Oscillator를 사용한다.

"1.1 Configuration Signals"와 "1.7 Clock Signals"를 참조하라.

External PHY와 Transformer간의 Impedance matching 회로는 각 제조사의 문서를 참조하라.

W5300의 '/FDX' Pin은 External PHY의 Duplex indicator signal과 연결할 수 있다.

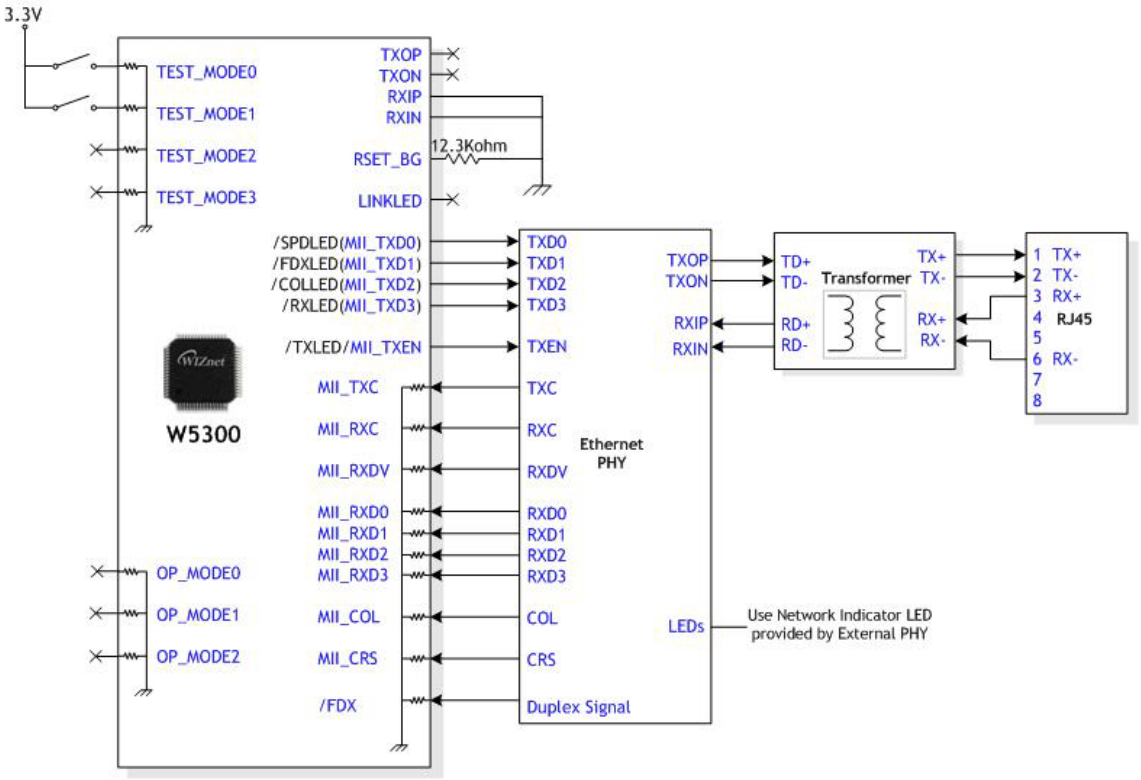


Fig 20. External PHY Interface with MII

7. Electrical Specifications

Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
V_{DD}	DC Supply voltage	-0.5 to 3.6	V
V_{IN}	DC input voltage	-0.5 to 5.5 (5V tolerant)	V
V_{OUT}	DC output voltage	-0.5 to 3.6	V
I_{IN}	DC input current	±5	mA
I_{OUT}	DC output current	2 to 8	mA
T_{OP}	Operating temperature	-40 to 80 [1]	°C
T_{STG}	Storage temperature	-55 to 125	°C

***COMMENT**: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage.

[1] : Please refer our Qualification Report in our website(search in http://www.wiznet.co.kr or http://www.wiznet.co.kr/UpLoad_Files/ReferenceFiles/KOLAS_Test_Report_QRTC-D-0808-169_W5300[0].pdf)

DC Characteristics

Symbol	Parameter	Test Condition	Min	Typ	Max	Unit
V_{DD}	DC Supply voltage	Junction temperature is from -55°C to 125°C	3.0	3.3	3.6	V
V_{IH}	High level input voltage		2.0		5.5	V
V_{IL}	Low level input voltage		- 0.5		0.8	V
V_{OH}	High level output voltage	I_{OH} = 2 ~ 16 mA	2.4			V
V_{OL}	Low level output voltage	I_{OL} = -2 ~ -16 mA			0.4	V
I_I	Input Current	$V_{IN} = V_{DD}$			±5	μA
I_O	Output Current	$V_{OUT} = V_{DD}$	2		8	mA

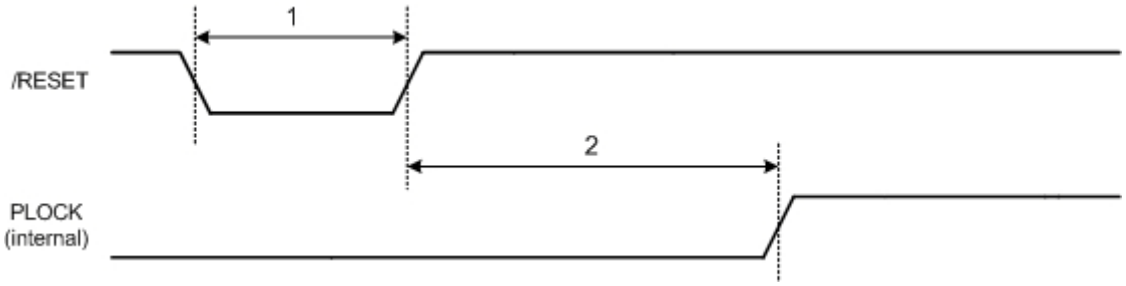
POWER DISSIPATION

Symbol	Parameter	Test Condition	Min	Typ	Max	Unit
P_{IA}	Power consumption when using the auto-negotiation	Vcc 3.3V Temperature 25°C	-	180	250	mA

	of internal PHY mode					
P_{IM}	Power consumption when using manual configuration of internal PHY mode	Vcc 3.3V Temperature 25°C	-	175	210	mA
P_E	Power consumption when using external PHY mode	Vcc 3.3V Temperature 25°C		65	150	mA

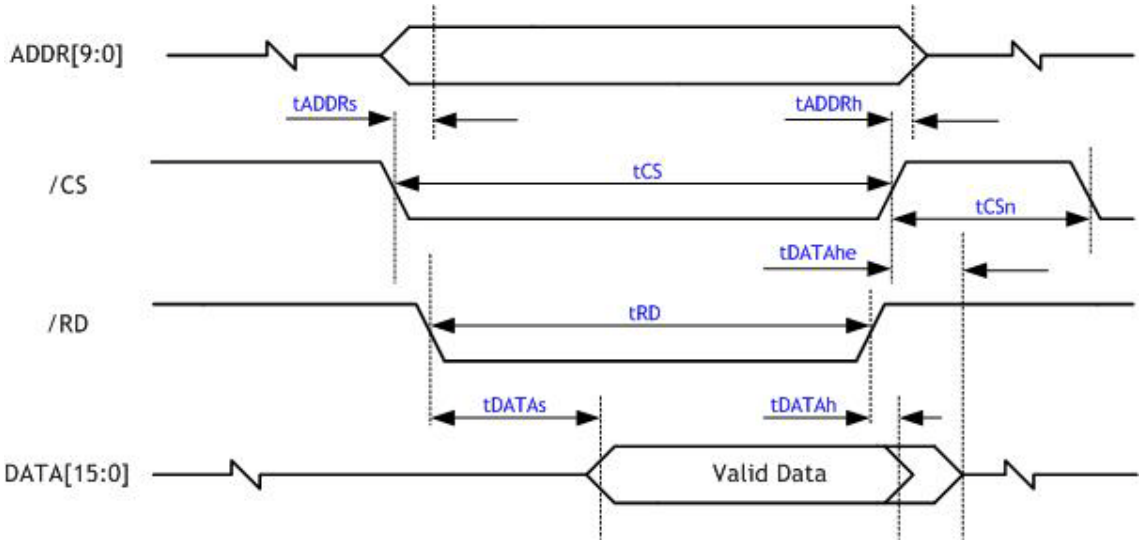
AC Characteristics

Reset Timing



	Description	Min	Max
1	Reset Cycle Time	2 us	-
2	PLL Lock-in Time	50 us	10 ms

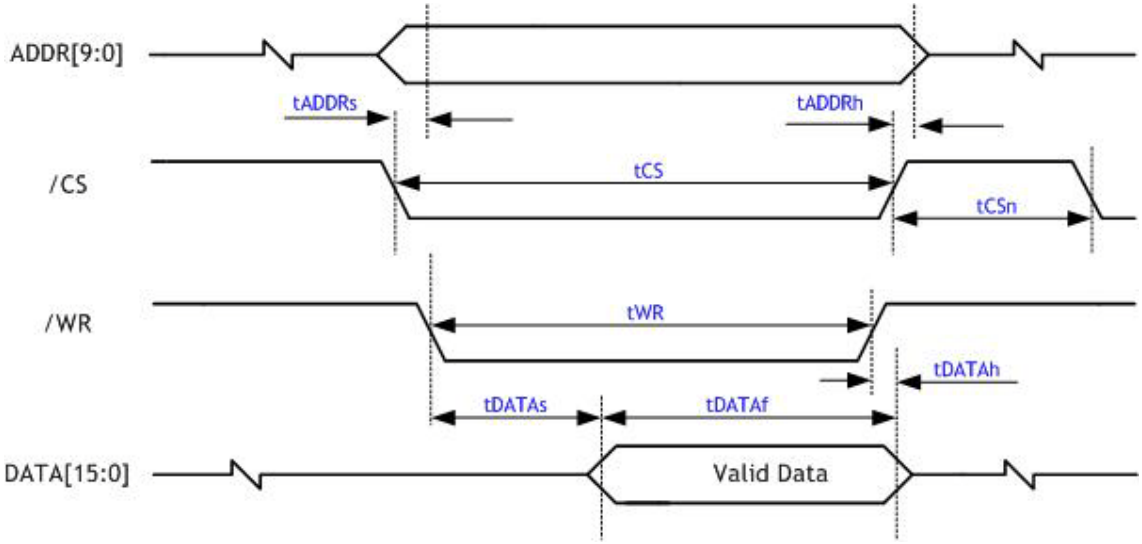
Register READ Timing



Description		Min	Max
tADDRs	Address Setup Time after /CS and /RD low	-	7 ns
tADDRh	Address Hold Time after /CS or /RD high	-	-
tCS	/CS Low Time	65 ns	-
tCSn	/CS Next Assert Time	28 ns	-
tRD	/RD Low Time	65 ns	-
tDATAs	DATA Setup Time after /RD low	42 ns	-
tDATAh	DATA Hold Time after /RD and /CS high	-	7 ns
tDATAhe	DATA Hold Extension Time after /CS high	-	2XPLL_CLK

<Note> 'tDATAhe'는 MR(RDH)='1'일 때만 적용되는 Data Hold Time이다. MR(RDH) = '1'은 '/CS'가 High로 De-assert된 후에도 2XPLL_CLK 동안 Data bus가 Driven되기 때문에 Data bus collision이 발생할 수 있다. 이는 사용에 있어 주의를 요한다.

Register WRITE Timing



Description		Min	Max
tADDRs	Address Setup Time after /CS and /WR low	-	7 ns
tADDRh	Address Hold Time after /CS or /RD high	-	-
tCS	/CS low Time	50 ns	-
tCSn	/CS next Assert Time	28 ns	
tWR	/WR low time	50 ns	
tDATAs	Data Setup Time after /WR low	7 ns	7ns + 7XPLL_CLK
tDATAf	Data Fetch Time	14 ns	tWR-tDATAs
tDATAh	Data Hold Time after /WR high	7 ns	-

<Note> 'tDATAs'는 MR(WDF2-WDF0)의 설정 값에 따라 최대 7 PLL_CLK동안 Host-Write data의 Fetch를 지연시키는 시간이다.

'tDATAf'는 Host-Write data를 Fetch할 수 있는 시간으로, 이 시간보다 먼저 /WR가 High로 De-assert될 경우 'tDATAf'와 상관없이 /WR High-De-assert시점에 Host-Write data를 Fetch한다.

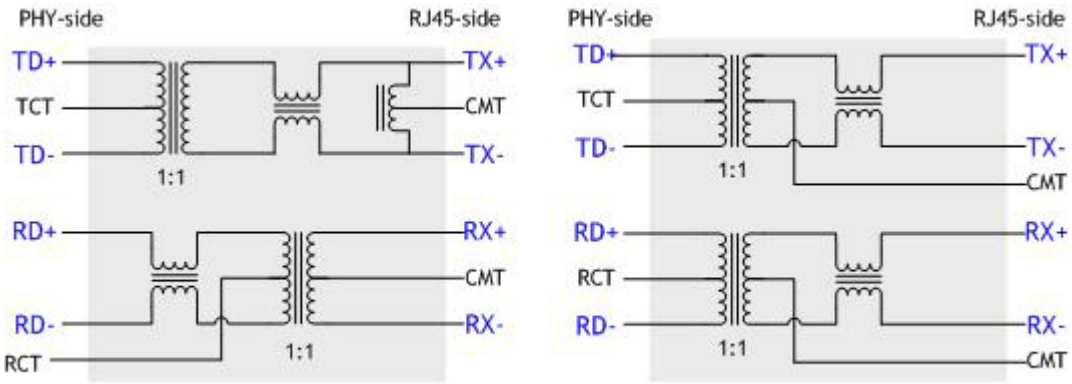
이 때 유효한 data를 Fetch하기 위해 Host는 'tDATAh'를 보장해야 한다.

Crystal Characteristics

Parameter	Range
Frequency	25 MHz
Frequency Tolerance (at 25℃)	±30 ppm
Shunt Capacitance	7pF Max
Drive Level	1 ~ 500uW (100uW typical)
Load Capacitance	27pF
Aging (at 25℃)	±3ppm / year Max

Transformer Characteristics

Parameter	Transmit End	Receive End
Turn Ratio	1:1	1:1
Inductance	350 uH	350 uH



Internal PHY mode에서 Auto MDI/MDIX(Crossover)을 지원하기 위해서는 반드시 Symmetric transformer를 사용해야 한다.

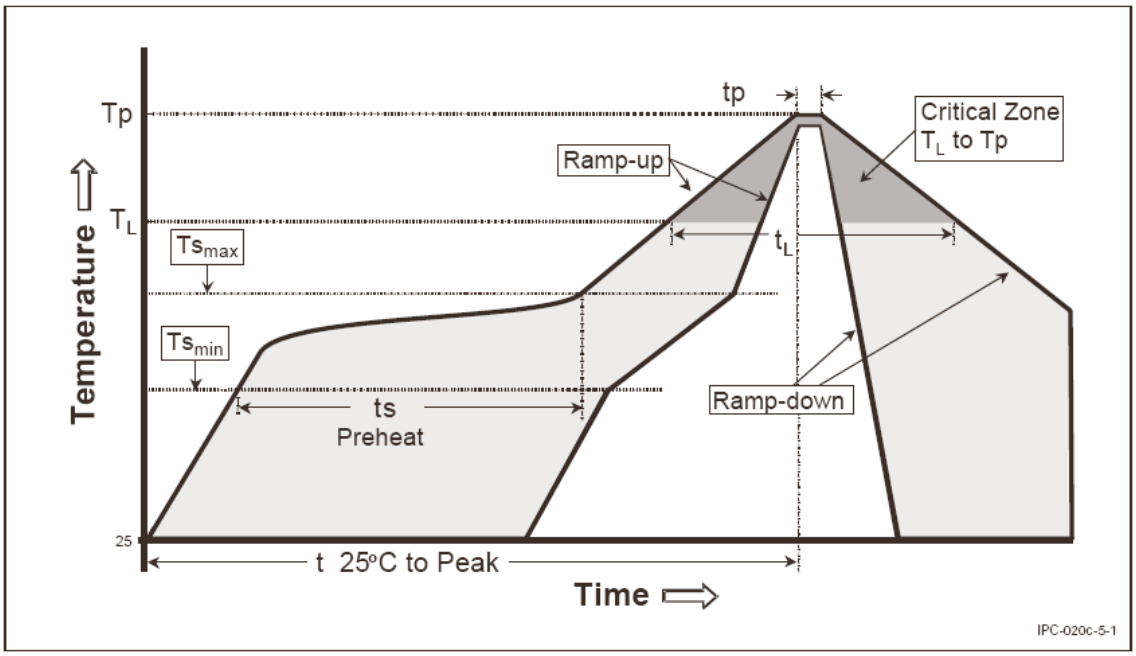
External PHY mode에서는 External PHY의 Spec에 맞는 Transformer를 선택하여 사용한다.

8. IR Reflow Temperature Profile (Lead-Free)

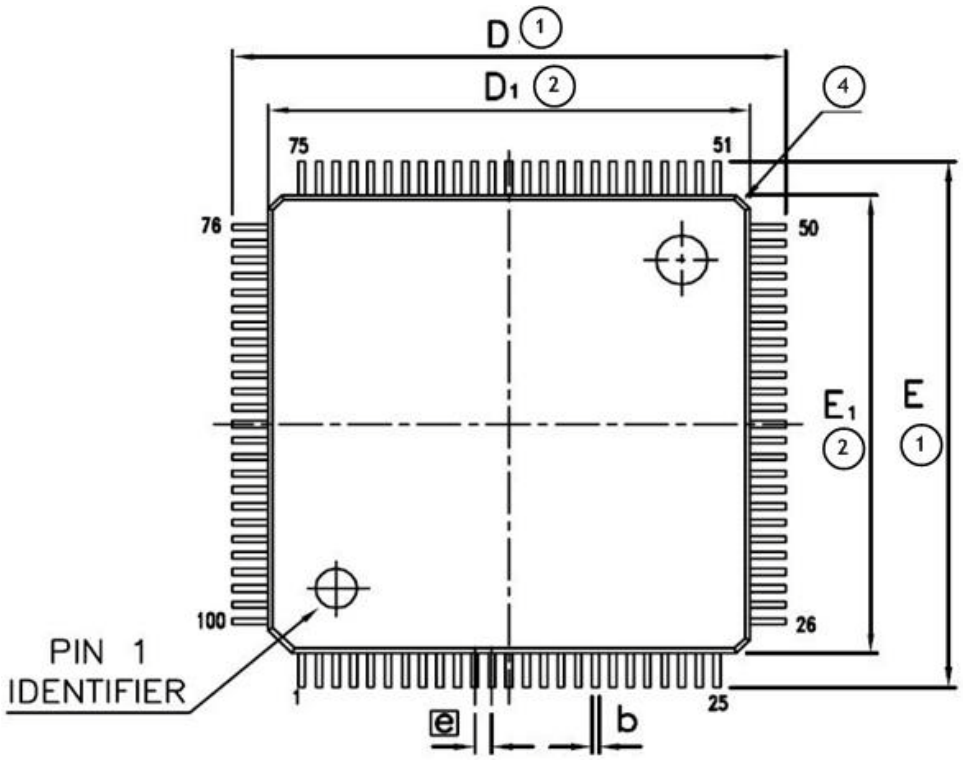
Moisture Sensitivity Level : 3

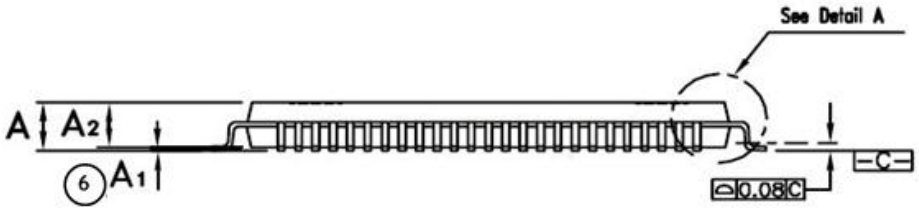
Dry Pack Required : Yes

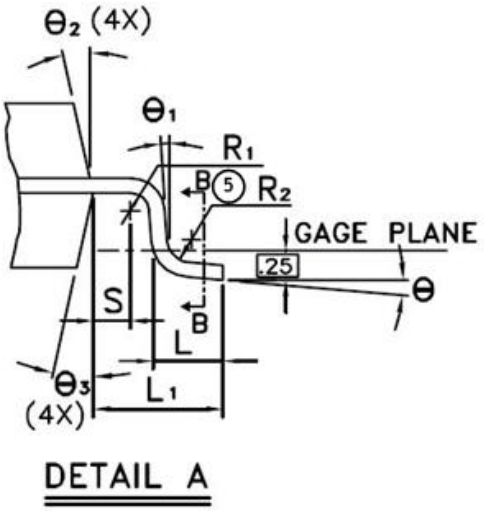
Average Ramp-Up Rate (Ts_{max} to Tp)	3° C/second max.
Preheat – Temperature Min (Ts_{min}) – Temperature Max (Ts_{max}) – Time (ts_{min} to ts_{max})	 150 °C 200 °C 60-180 seconds
Time maintained above: – Temperature (TL) – Time (tL)	 217 °C 60-150 seconds
Peak/Classification Temperature (Tp)	260 + 0 °C
Time within 5 °C of actual Peak Temperature (tp)	20-40 seconds
Ramp-Down Rate	6 °C/second max.
Time 25 °C to Peak Temperature	8 minutes max.



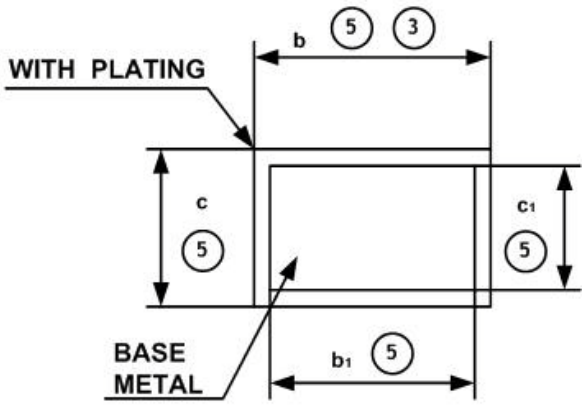
9. Package Descriptions







DETAIL A



SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	-	-	1.60	-	-	0.063
A_1	0.05	-	0.15	0.002	-	0.006
A_2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.17	0.22	0.27	0.007	0.009	0.011
b_1	0.17	0.20	0.23	0.007	0.008	0.009
c	0.09	-	0.20	0.004	-	0.008
c_1	0.09	-	0.16	0.004	-	0.006
D	15.85	16.00	16.15	0.624	0.630	0.636
D_1	13.90	14.00	14.10	0.547	0.551	0.555
E	15.85	16.00	16.15	0.624	0.630	0.636
E_1	13.90	14.00	14.10	0.547	0.551	0.555
e		0.50 BSC			0.020 BSC	
L	0.45	0.60	0.75	0.018	0.024	0.030
L_1		1.00 REF			0.039 REF	
R_1	0.08	-	-	0.003	-	-
R_2	0.08	-	0.20	0.003	-	0.008
S	0.20	-	-	0.008	-	-
θ	0°	3.5°	7°	0°	3.5°	7°
$θ_1$	0°	-	-	0°	-	-
$θ_2$		12° TYP			12° TYP	
$θ_3$		12° TYP			12° TYP	

<NOTE>

① To be determined at seating plane -C-.

② Dimensions 'D_1' and 'E_1' do not include mold protrusion.
D_1' and 'E_1' are maxium plastic body size dimensions including mold mismatch.

③ Dimension 'b' does not include dambar protrusion.
Dambar can not be located on the lower radius or the foot.

④ Exact shape of each corner is optional

⑤ These Dimensions apply to the flat section of the lead between 0.10mm and 0.25mm from the lead tip.

⑥ A_1 is defined as the distance from the seating plane to the lowest point of the package body.

7 Controlling dimension : Millimeter

8 Reference Document : JEDEC MS-026 , BED.