i2Chip W3100A

Description

The i2Chip W3100A is an LSI of hardware protocol stack that provides an easy, low-cost solution for high-speed Internet connectivity for digital devices by allowing simple installation of TCP/IP stack in the hardware.

The W3100A offers system designers a quick, easy way to add Ethernet networking functionality to any product. Implementing this LSI into a system can completely offload Internet connectivity and processing standard protocols from the system, thereby significantly reducing the software development cost.

The W3100A contains TCP/IP Protocol Stacks such as TCP, UDP, IP, ARP and ICMP protocols, as well as Ethernet protocols such as Data Link Control and MAC protocol.

The W3100A offers a socket API (Application Programming Interface) that is similar to the windows socket API. The chip offers Intel and Motorola MCU (8051, i386, 6811 tested) bus interface and I^2C for upper-layer and supports standard MII interface for under-layer Ethernet.

The W3100A can be applied to handheld devices including Internet phones, VoIP SOC chips, Internet MP3 players, handheld medical devices, LAN cards for Web servers, cellular phones and many other non-portable electronic devices such as large consumer electronic products.

Features

- Hardware Internet protocols included:
  TCP, IP Ver.4, UDP, ICMP, ARP
- Hardware Ethernet protocols included:
  DLC, MAC
- Supports 4 independent connections simultaneously
- Internal ICMP responds to PING commands
- Protocol processing speed: full-duplex 4~5 Mbps
- Intel/Motorola MCU bus Interface
- I^2C Interface
- Standard MII Interface for under-layer physical chip
- Socket API support for easy application programming
- Supports full-duplex mode
- Internal 16Kbytes Dual-port SRAM for data buffer
- 0.35 μm CMOS technology
- Wide operating voltage:
  3.3V internal operation, 5V tolerant 3.3V IOs
- Small 64 Pin LQFP Package

Block Diagram

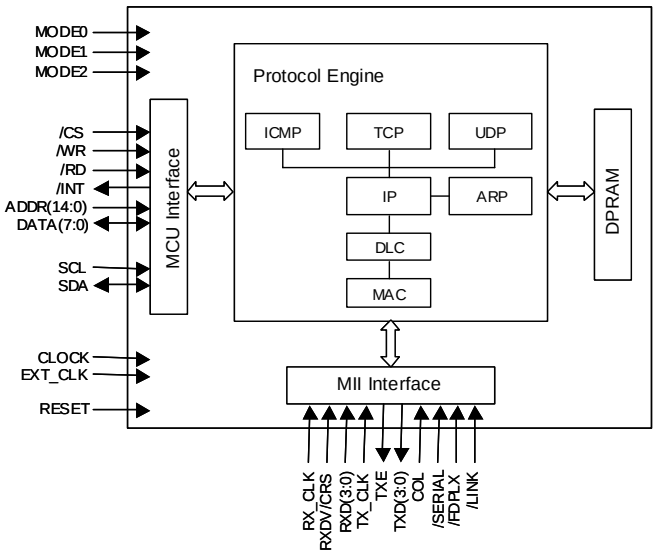
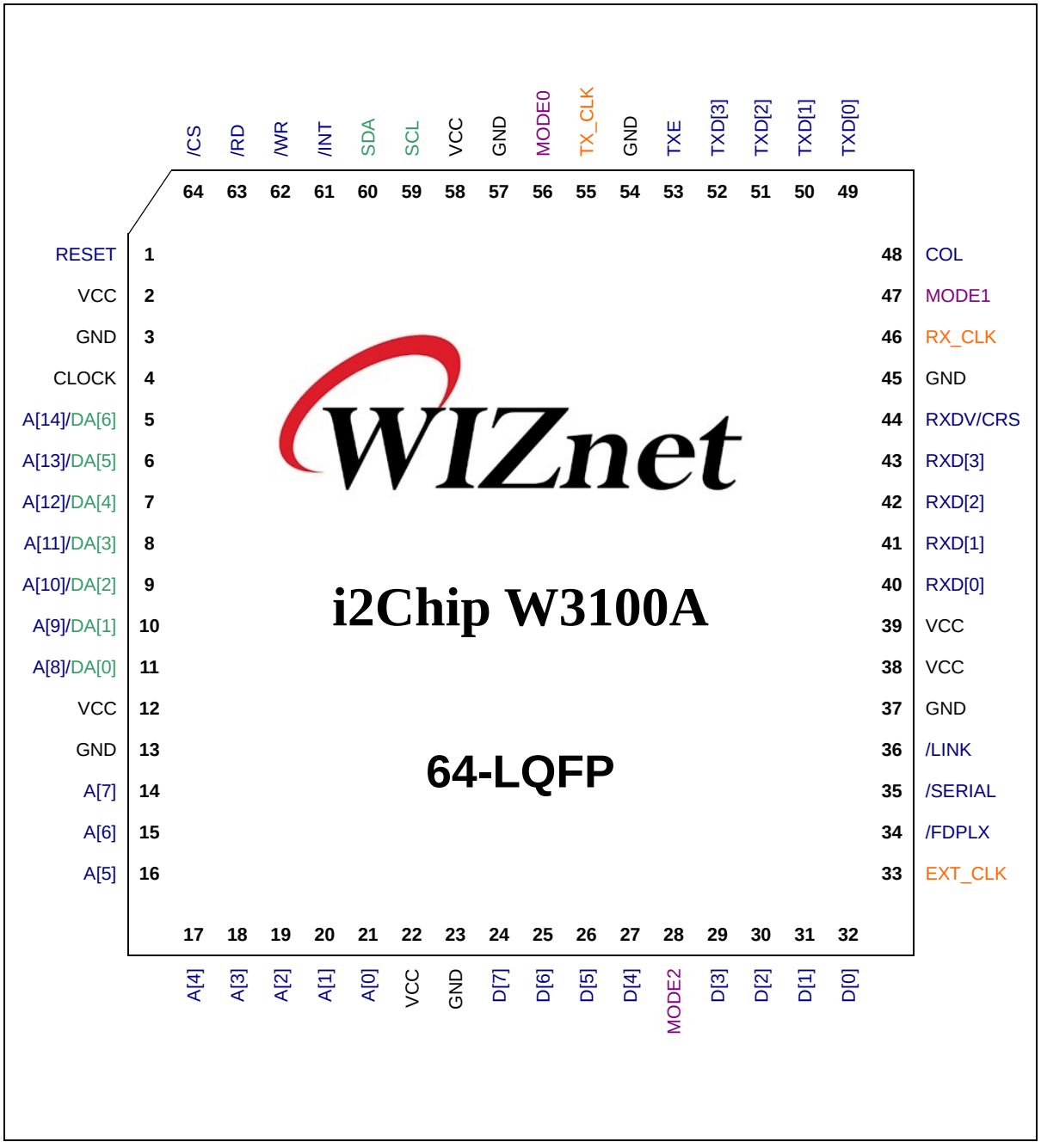


Table of Contents

■ Pin Assignment

Figure 1: 64-Pin LQFP Pin Assignments



■ Signal Description

Table 1: W3100A MII Signal Description

PIN#	Signal	I/O	Description
52 51 50 49	TXD[3] TXD[2] TXD[1] TXD[0]	O	**TRANSMIT DATA:** Nibble/Serial NRZ data output to the ENDEC that is valid on the rising edge of TX_CLK. In serial mode, the TXD[0] pin is used as the serial data pin, and TXD[3:1] are ignored.
53	TXE	O	**TRANSMIT ENABLE:** becomes active when the first nibble/serial data of the packet is valid on TXD[3:0] and goes low after the last nibble/serial data of the packet is clocked out of TXD[3:0]. This signal connects directly to the ENDEC (PHY device). This signal is active high.
55	TX_CLK	I	**TRANSMIT CLOCK:** TX_CLK is sourced by the PHY. TX_CLK is 2.5 MHz in 10BASE-T Nibble mode, and 25 MHz in 100BASE-T Nibble mode.
43 42 41 40	RXD[3] RXD[2] RXD[1] RXD[0]	I	**RECEIVE DATA:** Nibble wide receive data (synchronous to RX_CLK) that must be driven on the falling edge of RX_CLK. In serial mode, the RXD[0] pin is used as the data input pin which is also clocked in on the falling edge of RX_CLK. and RXD[3:1] pins become don't cares.
44	RXDV/CRS	I	**CARRIER SENSE:** signal provided by the ENDEC and indicates that carrier is present. This signal is active high.
46	RX_CLK	I	**RECEIVE CLOCK:** Re-synchronized clock from the ENDEC and indicates that carrier is present.
48	COL	I	**COLLISION DETECT:** becomes active when a collision has been detected in Half Duplex modes. This signal is asynchronous, active high and ignored during full-duplex operation.

Table 2: W3100A MCU Interface Signal Description

PIN#	Signal	I/O	Description
5-11	A[14-8] / DA[6-0]	I	**ADDRESS PINS / DEVICE ADDRESS PINS** Used as Address[14 – 8] pin when set in MCU Bus Interface mode. Used as Device address[6 – 0] pin for I^2C Interface when set in I^2C Interface mode.
14-21	A[7-0]	I	**ADDRESS PINS**
24-27 29-32	D[7-4] D[3-0]	I/O	**DATA PINS**
61	/INT	O	**INTERRUPT:** Indicates that the W3100A requires MCU attention after reception or transmission. The interrupt is cleared by writing to the ISR (Interrupt Status Register). All interrupts are maskable by writing IMG (Interrupt Mask Register). This signal is active low.
64	/CS	I	**CHIP SELECT:** This signal is active low.

62	/WR	I	**WRITE ENABLE:** This signal is active low.
63	/RD	I	**READ ENABLE:** This signal is active low.
59	SCL	I	**SCL:** clock used by I^2C when using I^2C Interface mode External Pull high (4.7 kΩ) is required.
60	SDA	I/O	**SDA:** data used by I^2C when using I^2C Interface mode External Pull high (4.7 kΩ) is required.

Table 3: W3100A Miscellaneous Signal Description

PIN#	Signal	I/O	Description
1	RESET	I	**RESET:** Active High input that initializes or reinitializes the W3100A. Asserting this pin will force a reset process to occur which will result in all internal registers reinitializing to their default states as specified for each bit in section x.x, and all strapping options are reinitialized. Refer to section x.x for further detail regarding reset.
4	CLOCK	I	**CLOCK:** primary clock required for internal operation of W3100A. In general, PHY driving clock is shared for saving cost. (25MHz is recommended) Note) Sharing crystal source clock with multiple devices may cause some troubles. In our reference design, we used Realtek's PHY and one crystal for both PHY and W3100A with verification. But for other kind of PHY, please confirm safety prior to decision.
33	EXT_CLK	I	**EXTERNAL CLOCK:** supplementary clock used for MCU I/F of W3100A. In external clocked mode, W3100A uses this clock to interface with MCU, and the access time of W3100A varies upon the frequency of the external clock. Refer to xx for detailed timing diagram. Frequency higher than 25MHz clock rate is granted.
36	/LINK	I	**LINK:** This is the signal generated by Ethernet PHY to indicate the PHY is connected to the Ethernet HUB device or other peer device. This is active low. W3100A can knows the status of physical line connection with this /LINK input. If /LINK is high, W3100A interprets the physical line is disconnected. It results in TCP timeout and connection close. In special PHY case, LINK signal varies in time, which can be grounded.
35	/SERIAL	I	**10BASE-T SERIAL/NIBBLE SELECT:** With the selection of this active low input, transmit and receive data are exchanged serially at a 10 MHz clock rate on the least significant bits of the nibble-wide MII data buses, pins TXD[0] and RXD[0], respectively. This mode is intended for use with the W3100A connected to a PHY using a 10 Mb/s serial interface. There is an internal pull-up resister for this pin. If this pin is left floated externally, then the device will be configured to normal mode. This pin must be externally pulled low (typically x kΩ) in

<table>
<tr><td></td><td></td><td></td><td>order to configure the W3100A for Serial MII operation.</td></tr>
<tr><td>34</td><td>/FDPLX</td><td>I</td><td>FULL/HALF DUPLEX SELECT: This input pin selects Half/Full Duplex operation.
This pin must be externally pulled low (typically x kΩ) in order to configure the W3100A for Full Duplex operation.
0 = Full Duplex
1 = Half Duplex</td></tr>
<tr><td>28, 47, 56</td><td>MODE[2–0]</td><td>I</td><td>MODE SELECT: This input pin selects MCU I/F type and operating mode of W3100A.
Since each pin is positioned as pull-down internally, clock mode – the default mode – is selected when the connection is not made.
<table>
<tr><th>M2</th><th>M1</th><th>M0</th><th></th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Clocked mode</td><td>Mode where MCU Bus signal is analyzed by W3100A by using the clock when MCU Bus I/F is in use.</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>External clocked mode</td><td>Mode where MCU Bus signal is analyzed by W3100A by using the external clock when MCU Bus I/F is in use.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Non-clocked mode</td><td>Mode where MCU bus signal is used directly by W3100A when MCU Bus I/F is in use.</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>I^2C mode</td><td>Mode using I^2C for MCU I/F.</td></tr>
<tr><td>1</td><td>X</td><td>X</td><td>Test mode</td><td>Mode used for testing at the plant. Not to be used by regular users.</td></tr>
</table>
Clocked mode, External clocked mode and Non-clocked mode are used to connect MCU and W3100A when MCU Bus I/F is in use. Choose an appropriate mode and use it by analyzing the MCU bus timing. Refer to timing diagram for each mode for more detail.</td></tr>
</table>

Table 4: W3100A Power Supply Signal Description

PIN#	Signal	I/O	Description
2, 12, 22, 38, 39, 58	VCC		**POSITIVE 3.3V SUPPLY PINS**
3, 13, 23, 37, 45, 54, 57	GND		**NEGATIVE (GROUND) SUPPLY PINS:** a decoupling capacitor is recommended to be connected between the Vcc and GND pins

Table5. W3100A Registers Map

Address	Register	Bit Definitions							
0x00	C0_CR	S/W Reset	Recv	Send	Close	Listen	Connect	Sock_Init	Sys_Init
0x01	C1_CR	Memory Test	Recv	Send	Close	Listen	Connect	Sock_Init	
0x02	C2_CR		Recv	Send	Close	Listen	Connect	Sock_Init	
0x03	C3_CR		Recv	Send	Close	Listen	Connect	Sock_Init	
0x04	C0_ISR		Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	Init_OK
0x05	C1_ISR		Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	
0x06	C2_ISR		Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	
0x07	C3_ISR		Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	
0x08	IR	C3R	C2R	C1R	C0R	C3	C2	C1	C0
0x09	IMR	IM_C3R	IM_C2R	IM_C1R	IM_C0R	IM_C3	IM_C2	IM_C1	IM_C0
0x0A – 0x0B	Reserved								
0x0C	IDM_OR	IND_EN						L/B	AUTO_INC
0x0D	IDM_AR0	Indirect bus I/F mode Address0 Register							
0x0E	IDM_AR1	Indirect bus I/F mode Address1 Register							
0x0F	IDM_DR	Indirect bus I/F mode Data Register							
0x10 – 0x13	C0_RW_PR	Channel 0 Rx Write Pointer Register							
0x14 – 0x17	C0_RR_PR	Channel 0 Rx Read Pointer Register							
0x18 – 0x1B	C0_TA_PR	Channel 0 Tx ACK Pointer Register							
0x1C – 0x1F	C1_RW_PR	Channel 1 Rx Write Pointer Register							
0x20 – 0x23	C1_RR_PR	Channel 1 Rx Read Pointer Register							
0x24 – 0x27	C1_TA_PR	Channel 1 Tx ACK Pointer Register							

0x28 – 0x2B	C2_RW_PR	Channel 2 Rx Write Pointer Register
0x2C – 0x2F	C2_RR_PR	Channel 2 Rx Read Pointer Register
0x30 – 0x33	C2_TA_PR	Channel 2 Tx ACK Pointer Register
0x34 – 0x37	C3_RW_PR	Channel 3 Rx Write Pointer Register
0x38 – 0x3B	C3_RR_PR	Channel 3 Rx Read Pointer Register
0x3C – 0x3F	C3_TA_PR	Channel 3 Tx ACK Pointer Register
0x40 – 0x43	C0_TW_PR	Channel 0 Tx Write Pointer Register
0x44 – 0x47	C0_TR_PR	Channel 0 Tx Read Pointer Register
0x48 – 0x4B	Reserved	
0x4C – 0x4F	C1_TW_PR	Channel 1 Tx Write Pointer Register
0x50 – 0x53	C1_TR_PR	Channel 1 Tx Read Pointer Register
0x54 – 0x57	Reserved	
0x58 – 0x5B	C2_TW_PR	Channel 2 Tx Write Pointer Register
0x5C – 0x5F	C2_TR_PR	Channel 2 Tx Read Pointer Register
0x60 – 0x63	Reserved	
0x64 – 0x67	C3_TW_PR	Channel 3 Tx Write Pointer Register
0x68 – 0x6B	C3_TR_PR	Channel 3 Tx Read Pointer Register
0x6C – 0x7F	Reserved	
0x80 – 0x83	GAR	Gateway Address Register
0x84 – 0x87	SMR	Subnet Mask Register
0x88 – 0x8D	SHAR	Source Hardware Address Register
0x8E – 0x91	SIPR	Source IP Address Register
0x92 – 0x93	IRTR	Initial Retry Time-value Register

0x94	RCR				TC4	TC3	TC2	TC1	TC0
0x95	RMSR	Rx data Memory Size Register							
0x96	TMSR	Tx data Memory Size Register							
0x97 – 0x9F	Reserved								
0xA0	C0_SSR	Channel 0 Socket Status Register							
0xA1	C0_SOPR	Broadcast/ERR	NDTimeout/B	NDAck	SWS/P		Protocol	Protocol	Protocol
0xA2 – 0xA7	Reserved								
0xA8 – 0xAB	C0_DIR	Channel 0 Destination IP Address Register							
0xAC – 0xAD	C0_DPR	Channel 0 Destination Port Register							
0xAE – 0xAF	C0_SPR	Channel 0 Source Port Register							
0xB0	C0_IPR	Channel 0 IP Protocol Register							
0xB1	C0_TOSR	Channel 0 TOS (type of service) Register							
0xB2 – 0xB3	C0_MSSR	Channel 0 MSS (maximum segment size) Register							
0xB4 – 0xB7	Reserved								
0xB8	C1_SSR	Channel 1 Socket Status Register							
0xB9	C1_SOPR	Broadcast	NDTimeout	NDAck	SWS		Protocol	Protocol	Protocol
0xBA – 0xBF	Reserved								
0xC0 – 0xC3	C1_DIR	Channel 1 Destination IP Address Register							
0xC4 – 0xC5	C1_DPR	Channel 1 Destination Port Register							
0xC6 – 0xC7	C1_SPR	Channel 1 Source Port Register							
0xC8	C1_IPR	Channel 1 IP Protocol Register							
0xC9	C1_TOSR	Channel 1 TOS (type of service) Register							
0xCA – 0xCB	C1_MSSR	Channel 1 MSS (maximum segment size) Register							

0xCC – 0xCF	Reserved								
0xD0	C2_SSR	Channel 2 Socket Status Register							
0xD1	C2_SOPR	Broadcast	NDTimeout	NDAck	SWS		Protocol	Protocol	Protocol
0xD2 – 0xD7	Reserved								
0xD8 – 0xDB	C2_DIR	Channel 2 Destination IP Address Register							
0xDC – 0xDD	C2_DPR	Channel 2 Destination Port Register							
0xDE – 0xDF	C2_SPR	Channel 2 Source Port Register							
0xE0	C2_IPR	Channel 2 IP Protocol Register							
0xE1	C2_TOSR	Channel 2 TOS (type of service) Register							
0xE2 – 0xE3	C2_MSSR	Channel 2 MSS (maximum segment size) Register							
0xE4 – 0xE7	Reserved								
0xE8	C3_SSR	Channel 3 Socket Status Register							
0xE9	C3_SOPR	Broadcast	NDTimeout	NDAck	SWS		Protocol	Protocol	Protocol
0xEA – 0xEF	Reserved								
0xF0 – 0xF3	C3_DIR	Channel 3 Destination IP Address Register							
0xF4 – 0xF5	C3_DPR	Channel 3 Destination Port Register							
0xF6 – 0xF7	C3_SPR	Channel 3 Source Port Register							
0xF8	C3_IPR	Channel 3 IP Protocol Register							
0xF9	C3_TOSR	Channel 3 TOS (type of service) Register							
0xFA – 0xFB	C3_MSSR	Channel 3 MSS (maximum segment size) Register							
0xFC – 0xFF	Reserved								

■ Register Definitions.

Register sets are categorized into (i) control registers related to command, status and interrupt, (ii) system registers for gateway address, subnet mask, source IP, source HA (Hardware Address) and timeout value, (iii) pointer registers for managing to send, receive data, and (iv) channel registers that control operation of each channel. R/W access to reserved register is not allowed, and also, writing on read-only register is not allowed.

1. Control Registers

C0_CR (Channel 0 Command Register) [R/W, 0x00]

This register commands Channel 0 socket to initialize, connect, close, transmit and receive data. Sys_Init command is used to set the gateway, subnet mask, source IP and source H/W Address. The same command is used to close the socket in all channels.

Sock_Init, Connect, Listen, Close, Send and Recv are used when initializing, establishing a connection, terminating a connection, sending and receiving data for Channel 0 socket. Each corresponding bit is automatically cleared after executing the command.

Sock_Init command opens the corresponding Channel in TCP, UDP, RAW mode according to the protocol value as set at C0_SOPR (Channel 0 Socket Option Protocol Register).

MCU can initialize the internal setting value of the chip by using S/W Reset.

Each bit in this register is automatically cleared after executing the command.

7	6	5	4	3	2	1	0
S/W Reset	Recv	Send	Close	Listen	Connect	Sock_Init	Sys_Init

Bit	Symbol	Description
D0	Sys_Init	Command to set Gateway IP Address, Subnet Mask, Source H/W Address, Source IP Address
D1	Sock_Init	Command to set corresponding protocol at C0_SOPR and open Channel 0 socket
D2	Connect	Command for Channel 0 socket to make a connection to the server
D3	Listen	Command to stand by for connection when Channel 0 socket acts in server mode
D4	Close	Command to terminate connection and close Channel 0 socket
D5	Send	Command to transmit Channel 0 socket data
D6	Recv	Command to receive Channel 0 socket data
D7	S/W Reset	S/W Reset command

C1_CR (Channel 1 Command Register) [R/W, 0x01]

This register commands Channel 1 Socket to initialize, connect, close, transmit and receive data.

Sock_Init, Connect, Listen, Close, Send and Recv are used when initializing, establishing a connection, terminating a connection, sending and receiving data for Channel 1 socket. Each corresponding bit is automatically cleared after executing the command.

Memory test command is used to verify transmission and reception memory where MCU reads and writes for the transmission and reception memory. Set memory test bit to '1' to become toggled as '0', '1' and W3100A acts in memory test mode when in '1'. Memory test bit needs to become set at '0' in order for W3100A to execute normal data transmission and reception.

7	6	5	4	3	2	1	0
Memory Test	Recv	Send	Close	Listen	Connect	Sock_Init	

Bit	Symbol	Description
D0		Reserved
D1	Sock_Init	Sets corresponding protocol at C1_SOPR and opens Channel 1 socket
D2	Connect	Command for Channel 1 socket to act in client mode to make a connection to the server
D3	Listen	Command to stand by for connection when Channel 1 socket acts in server mode
D4	Close	Command to terminate connection and close Channel 1 socket
D5	Send	Command to transmit Channel 1 socket data
D6	Recv	Command to receive Channel 1 socket data
D7	Memory Test	Command to set memory test mode

C2_CR, C3_CR (Channel 2, 3 Command Register) [R/W, 0x02, 0x03]

This register commands each Channel 2, 3 sockets to initialize, connect, close, transmit and receive data.

Sock_Init, Connect, Listen, Close, Send and Recv are used when initializing, establishing a connection, terminating a connection, sending and receiving data for corresponding socket. Each corresponding bit is automatically cleared after executing the command.

7	6	5	4	3	2	1	0
	Recv	Send	Close	Listen	Connect	Sock_Init	

Bit	Symbol	Description
D0		Reserved

D1	Sock_Init	Sets corresponding protocol at Cx_SOPR and opens corresponding channel socket
D2	Connect	Command for corresponding channel socket to act in client mode to make a connection to the server
D3	Listen	Command to stand by for connection when corresponding channel socket acts in server mode
D4	Close	Command to terminate connection and close corresponding channel socket
D5	Send	Command to transmit corresponding channel socket data
D6	Recv	Command to receive corresponding channel socket data
D7		Reserved

C0_ISR (Channel 0 Interrupt Status Register) [R, 0x04]

This register notifies the outcome of Channel 0 socket command.

This register becomes cleared as 0x00 by read operation.

Init_OK notifies the completion of Sys_Init command.

Established notifies the completion of a connection executed by connection set-up command (Connect, Listen).

Timeout notifies an occurrence of a time out while executing connection set-up command (Connect, Listen) or Send command.

SInit_OK, Closed, Send_OK and Recv_OK each notifies the completion of Sock_Init, Close, Send and Recv commands, respectively.

7	6	5	4	3	2	1	0
	Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	Init_OK

Bit	Symbol	Description
D0	Init_OK	Interrupt status bit for completion of Sys_Init command
D1	SInit_OK	Interrupt status bit for completion of Channel 0 socket Sock_Init command
D2	Established	Interrupt status bit for completion of Channel 0 socket connection set-up
D3	Closed	Interrupt status bit for completion of Channel 0 socket connection ending
D4	Timeout	Interrupt status bit for occurrence of time out during Channel 0 socket connection set-up or data transmission
D5	Send_OK	Interrupt status bit for completion of Channel 0 socket Send command
D6	Recv_OK	Interrupt status bit for completion of Channel 0 socket Recv command
D7		Reserved

C1_ISR, C2_ISR, C3_ISR (Channel 1, 2, 3 Interrupt Status Register) [R, 0x05, 0x06, 0x07]

This register notifies the outcome of the command of each Channel 1, 2 and 3.

This register becomes cleared as 0x00 by read operation.

Established notifies the completion of a connection executed by connection set-up command (Connect, Listen).

Timeout notifies an occurrence of a time out while executing connection set-up command (Connect, Listen) or Send command.

SInit_OK, Closed, Send_OK and Recv_OK each notifies the completion of Sock_Init, Close, Send and Recv commands, respectively.

7	6	5	4	3	2	1	0
	Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	

Bit	Symbol	Description
D0		Reserved
D1	SInit_OK	Interrupt status bit for completion of corresponding channel socket Sock_Init command
D2	Established	Interrupt status bit for completion of corresponding channel socket connection set-up
D3	Closed	Interrupt status bit for completion of corresponding channel socket connection ending
D4	Timeout	Interrupt status bit for occurrence of time out during corresponding channel socket connection set-up or data transmission
D5	Send_OK	Interrupt status bit for completion of corresponding channel socket Send command
D6	Recv_OK	Interrupt status bit for completion of corresponding channel socket Recv command
D7		Reserved

IR (Interrupt Register) [R/W, 0x08]

This register is used to sort channel with occurring interrupt.

C0, C1, C2 and C3 bit notify each of 0, 1, 2 and 3 channels that an interrupt has occurred. MCU can identify which interrupt has occurred by examining the Channel Interrupt Status Register of the corresponding channel.

C0R, C1R, C2R and C3R Bit notify that data transmission has occurred for 0, 1, 2 and 3 Channel. This register can clear the interrupt signal by writing '1' at the corresponding bit.

7	6	5	4	3	2	1	0
C3R	C2R	C1R	C0R	C3	C2	C1	C0

Bit	Symbol	Description
D0	C0	Occurrence of Channel 0 Socket Interrupt
D1	C1	Occurrence of Channel 1 Socket Interrupt
D2	C2	Occurrence of Channel 2 Socket Interrupt
D3	C3	Occurrence of Channel 3 Socket Interrupt
D4	C0R	Occurrence of Channel 0 Socket data receipt
D5	C1R	Occurrence of Channel 1 Socket data receipt
D6	C2R	Occurrence of Channel 2 Socket data receipt
D7	C3R	Occurrence of Channel 3 Socket data receipt

IMR (Interrupt Mask Register) [R/W, 0x09]

This register is used to mask an interrupt from each bit of the corresponding interrupt register. Interrupt is enabled when the corresponding bit of the interrupt register is set by setting the corresponding bit at '1'.

7	6	5	4	3	2	1	0
IM_C3R	IM_C2R	IM_C1R	IM_C0R	IM_C3	IM_C2	IM_C1	IM_C0

Bit	Symbol	Description
D0	IM_C0	Channel 0 Socket Interrupt Enable.
D1	IM_C1	Channel 1 Socket Interrupt Enable.
D2	IM_C2	Channel 2 Socket Interrupt Enable.
D3	IM_C3	Channel 3 Socket Interrupt Enable.
D4	IM_C0R	Channel 0 Socket data receipt Interrupt Enable.
D5	IM_C1R	Channel 1 Socket data receipt Interrupt Enable.
D6	IM_C2R	Channel 2 Socket data receipt Interrupt Enable.
D7	IM_C3R	Channel 3 Socket data receipt Interrupt Enable.

IDM_OR (InDirect Mode Option Register) [R/W, 0x0C]

This register facilitates indirect bus I/F mode enable and option set-up.

IND_EN (indirect mode enable) bit enables indirect bus I/F mode. H/W reset is required to convert from indirect bus I/F mode to another I/F.

L/B (Little-endian/Big-endian) bit enables the access to indirect address register as Little-endian ('1') or Big-endian ('0').

AUTO_INC (auto-increment) bit automatically increases the address during an access to indirect data register.

7	6	5	4	3	2	1	0
IND_EN						L/B	AUTO_INC

Bit	Symbol	Description
D7	IND_EN	Indirect bus I/F mode Enable.
D6		Reserved
D5		Reserved
D4		Reserved
D3		Reserved
D2		Reserved
D1	L/B	Little-endian/Big-endian ordering setting register.
D0	AUTO_INC	Address auto-increment Enable

IDM_AR0, IDM_AR1 (InDirect Mode Address Register) [R/W, 0x0D – 0x0E]

This register is for address set-up when indirect bus I/F mode is in use, and the ordering changes according to the L/B bit set-up of IM_OPT register.

When L/B bit of IDM_OPT register = '0'

MSB	LSB
IDM_ADDR0(0x0D)	IDM_ADDR1(0x0E)

When L/B bit of IDM_OPT register = '1'

LSB	MSB
IDM_ADDR0(0x0D)	IDM_ADDR1(0x0E)

IDM_DR (Indirect Mode Data Register) [R/W, 0x0F]

This register is for data when indirect bus I/F mode is in use.

2. System Registers

GAR (Gateway Address Register) [R/W, 0x80 – 0x83]

This register sets up the default gateway address to be used in the system, which is required to be set IP address before executing Sys_Init command.

SMR (Subnet Mask Register) [R/W, 0x84 – 0x87]

This register sets up the subnet mask to be used in the system, which is required to be set up before executing Sys_Init command.

SHAR (Source Hardware Address Register) [R/W, 0x88 – 0x8D]

This register sets up the HA to be used in the system, which is required to be set up before executing Sys_Init command.

SIPR (Source IP Address Register) [R/W, 0x8E – 0x91]

This register sets up the IP to be used in the system, which is required to be set up before executing Sys_Init command.

IRTR (Initial Retry Time-value Register) [R/W, 0x92 – 0x93]

This register sets up the timer value for initial re-transmission when using the TCP, and timer value 1 is equivalent to 100us.

Value	Timer (ms)
0x03E8	100
0x07D0	200
0x0FA0	400

RCR (Retry Count Register) [R/W, 0x94]

This register assigns the number of retry when re-transmission occurs, and timeout interrupt occurs when re-transmission exceeds the number of retry.

RMSR (Rx data Memory Size Register) [R/W, 0x95]

This register allocates 8KB of received memory for each channel.

CH3		CH2		CH1		CH0	
S1	S0	S1	S0	S1	S0	S1	S0

S1	S0	Memory size
0	0	1KB
0	1	2KB
1	0	4KB
1	1	8KB

2 bits of S1, S0 are allocated for each channel, and the memory for receiving is allocated according to the set-up value as shown in the table above.

TMSR (Tx data Memory Size Register) [R/W, 0x96]

This register allocates 8KB of transmitted memory for each channel.

CH3		CH2		CH1		CH0	
S1	S0	S1	S0	S1	S0	S1	S0

S1	S0	Memory size
0	0	1KB
0	1	2KB
1	0	4KB
1	1	8KB

2 bits of S1,S0 are allocated for each channel, and the memory for sending is allocated according to the set-up value as shown in the table above.

3. Pointer Registers

In order to read pointer registers, the shadow register of the corresponding pointer needs to be read and time delay of Tx_CLK * 4 is required before reading the corresponding pointer register. (Access by W3100 MCU I/F is based on 1Byte unit, but the pointer register is comprised of 4Bytes. Therefore, shadow register is used in order for MCU to properly read 4Byte pointer.)

To write, no access to the shadow register or time delay is necessary.

Shadow Registers	Address	Applicable Pointer Registers
C0_SRW_PR	0x1E0	C0_ RW_PR
C0_SRR_PR	0x1E1	C0_ RR_PR
C0_STA_PR	0x1E2	C0_ TA_PR
C1_SRW_PR	0x1E3	C1_ RW_PR
C1_SRR_PR	0x1E4	C1_ RR_PR
C1_STA_PR	0x1E5	C1_ TA_PR
C2_SRW_PR	0x1E6	C2_ RW_PR
C2_SRR_PR	0x1E7	C2_ RR_PR
C2_STA_PR	0x1E8	C2_ TA_PR
C3_SRW_PR	0x1E9	C3_ RW_PR
C3_SRR_PR	0x1EA	C3_ RR_PR
C3_STA_PR	0x1EB	C3_ TA_PR
C0_STW_PR	0x1F0	C0_ TW_PR

C0_STR_PR	0x1F1	C0_ TR_PR
C1_STW_PR	0x1F3	C1_ TW_PR
C1_STR_PR	0x1F4	C1_ TR_PR
C2_STW_PR	0x1F6	C2_ TW_PR
C2_STR_PR	0x1F7	C2_ TR_PR
C3_STW_PR	0x1F9	C3_ TW_PR
C3_STR_PR	0x1FA	C3_ TR_PR

RW_PR (Rx Write Pointer Register) [R/W, C0 : 0x10 – 0x13, C1 : 0x1C – 0x1F, C2 : 0x28 – 0x2B, C3 : 0x34 – 0x37]

Included in each channel, this register displays the data end pointer when receiving data. The register is managed internally by W3100A and increases according to the size of the data received.

MCU receives and processes the data from Rx Read Pointer to Rx Writer Pointer of the corresponding channel.

RR_PR (Rx Read Pointer Register) [R/W, C0 : 0x14 – 0x17, C1 : 0x20 – 0x23, C2 : 0x2C – 0x2F, C3 : 0x38 – 0x3B]

Included in each channel, this register displays the data start pointer when receiving data.

After processing the received data, MCU updates Rx Read Pointer as the pointer of the processed data and releases Recv Command.

TW_PR (Tx Write Pointer Register) [R/W, C0 : 0x40 – 0x43, C1 : 0x4C – 0x4F, C2 : 0x58 – 0x5B, C3 : 0x64 – 0x67]

Included in each channel, this register displays the data end pointer of the data to be transmitted when transmitting data.

For transmission, MCU writes the data to be transmitted from Tx Write Pointer of the corresponding channel, and Tx Write Pointer needs to be updated with a new value after the data is copied.

Transmission is made after executing the Send command.

TR_PR (Tx Read Pointer Register) [R/W, C0 : 0x44 – 0x47, C1 : 0x50 – 0x53, C2 : 0x5C – 0x5F, C3 : 0x68 – 0x6B]

Included in each channel, this register displays the current working pointer of the data to be transmitted when transmitting data.

The register, used internally in W3100A, displays the pointer to start transmission when transmission is made by send command.

TA_PR (Tx Ack Pointer Register) [R/W, C0 : 0x18 – 0x1B, C1 : 0x24 – 0x27, C2 : 0x30 – 0x33, C3 : 0x3C – 0x3F]

Included in each channel, this register displays the start pointer of the data to be transmitted when transmitting data.

Driver uses this register and Tx Write Pointer to calculate free size of Tx Buffer.

In other words, the difference in value of Tx Write Pointer and Tx Ack Pointer is the buffer size being used.

4. Channel Registers

SSR (Socket State Register) [R, C0 : 0xA0, C1 : 0x B8, C2 : 0x D0, C3 : 0x E8]

Displays the socket state of the corresponding channel.

Value	State	Meaning
0x00	SOCK_CLOSED	Socket is closed
0x01	SOCK_ARP	Standing by for reply after transmitting ARP Request
0x02	SOCK_LISTEN	Standing by for connection set-up to the client when acting in passive mode
0x03	SOCK_SYNSENT	Standing by for SYN,ACK after transmitting SYN for connection set-up when acting in active mode
0x04	SOCK_SYNSENT_ACK	Connection set-up is complete after SYN,ACK is received and ACK is transmitted in active mode
0x05	SOCK_SYNRECV	SYN,ACK is being transmitted after receiving SYN from the client in listen state, passive mode
0x06	SOCK_ESTABLISHED	Connection set-up is complete in active, passive mode
0x07	SOCK_CLOSE_WAIT	Connection being terminated
0X08	SOCK_LAST_ACK	Connection being terminated
0X09	SOCK_FIN_WAIT1	Connection being terminated
0X0A	SOCK_ FIN_WAIT2	Connection being terminated
0X0B	SOCK_CLOSING	Connection being terminated
0X0C	SOCK_TIME_WAIT	Connection being terminated
0X0D	SOCK_RESET	Connection is being terminated after receiving reset packet from the peer
0X0E	SOCK_INIT	Socket initializing
0X0F	SOCK_UDP	Applicable channel is initialized in UDP mode
0X10	SOCK_RAW	Applicable channel is initialized in IP layer RAW mode
0X11	SOCK_UDP_ARP	Standing by for reply after transmitting ARP request packet to the destination for UDP transmission
0X12	SOCK_UDP_DATA	Data transmission in progress in UDP or RAW mode
0X13	SOCK_RAW_INIT	W3100A initialized in MAC layer RAW mode

SOPR (Socket Option and Protocol Register) [R/W, C0 : 0xA1, C1 : 0x B9, C2 : 0x D1, C3 : 0x E9]

This register sets up socket option or protocol of the corresponding channel.

7	6	5	4	3	2	1	0
Broadcast/ ERR	NDTimeout/ B	NDAck	SWS/ P		Protocol	Protocol	Protocol

Bit	Symbol	Description
D0 D1 D2	Protocol	Sets up corresponding channel in TCP, UDP, IP Layer RAW mode or MAC Layer RAW mode **Value** / **Meaning** 000 Closed 001 SOCK_STREAM(TCP) 010 SOCK_DGRAM(UDP) 011 SOCK_IPL_RAW(IP Layer RAW Mode) 100 SOCK_MACL_RAW(MAC Layer RAW Mode)
D3		Reserved
D4	SWS/P	Silly Window Syndrome '0': prevents Silly Window Syndrome when using TCP '1': does not prevent Silly Window Syndrome when using TCP When using MAC Layer RAW mode, promiscuous packet (packet with specific MAC address) can be received by when C0_SOPR sets the bit as '1'
D5	NDAck	No Delayed ACK '0': uses delayed ACK '1': does not use delayed ACK – ACK is transmitted immediately upon receiving the data packet
D6	NDTimeout/B	No Dynamic Timeout '0': uses dynamic timeout during operation to set up timeout value regardless of the set-up value '1': activates by using the timeout value as set up in Timeout Value When using MAC Layer RAW mode, broadcast packet can be received by when C0_SOPR sets the bit as '1'
D7	Broadcast/ERR	Broadcast packet is received and transmitted in IP Layer RAW mode When using MAC Layer RAW mode, error packet can be received by when C0_SOPR sets the bit as '1'

DIR (Destination IP Address Register) [R/W, C0 : 0xA8 – 0xAB, C1 : 0xC0 – 0xC3, C2 : 0xD8 – 0xDB, C3 : 0xF0 – 0xF3]

This register sets the Destination IP Address of each channel to be used in setting the TCP connection. In active mode, IP address needs to be set before executing the Connect command. In passive mode, W3100A sets up the connection and then updates as peer IP internally.

DPR (Destination Port Register) [R/W, C0 : 0xAC – 0xAD, C1 : 0xC4 – 0xC5, C2 : 0xDC – 0xDD, C3 : 0xF4 – 0xF5]

This register sets the Destination Port number of each channel to be used in setting the TCP connection. In active mode, port number needs to be set before executing the Connect command. In passive mode, W3100A sets up the connection and then updates as peer port number internally.

SPR (Source Port Register) [R/W, C0 : 0xAE – 0xAF, C1 : 0xC6 – 0xC7, C2 : 0xDE – 0xDF, C3 : 0xF6 – 0xF7]

This register sets the Source Port number for each channel when using TCP or UDP mode, and the set-up needs to be made before executing the Sock_Init Command.

IPR (IP Protocol Register) [R/W, C0 : 0xB0, C1 : 0xC8, C2 : 0xE0, C3 : 0xF8]

This IP Protocol Register is used to be set up at the Protocol Field of IP Header when executing the IP Layer RAW Mode, and the set-up needs to be made before executing the Sock_Init Command.

TOSR (TOS Register) [R/W, C0 : 0xB1, C1 : 0xC9, C2 : 0xE1, C3 : 0xF9]

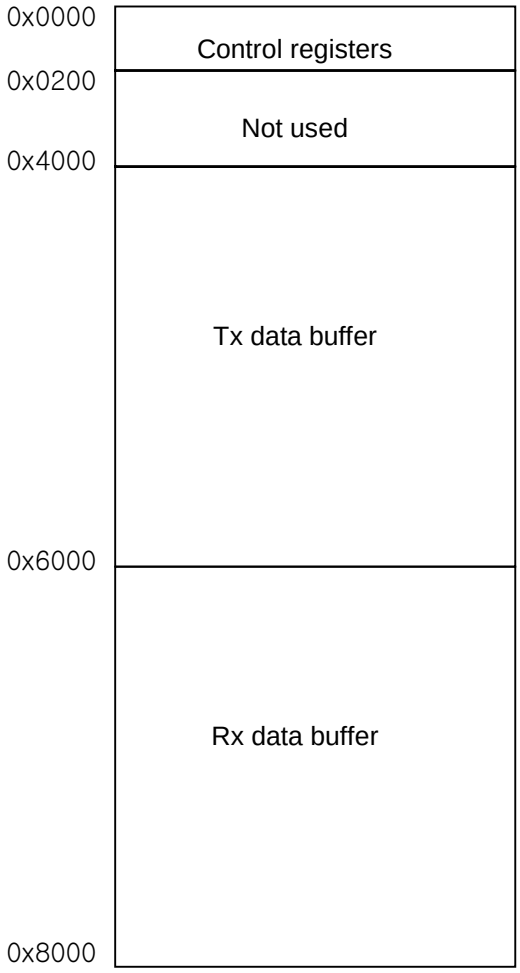
This register is used to be set up at the TOS (Type Of Service) Field of IP Header, and the set-up needs to be made before executing the Sock_Init Command.

MSSR (MSS Register) [R/W, C0 : B2 – 0xB3, C1 : 0xCA – 0xCB, C2 : 0xE2 – 0xE3, C3 : 0xFA – 0xFB]

This register is used for MSS (Maximum Segment Size) of TCP, and the register displays MSS set by the other party when TCP is activated in Passive Mode.

■ Internal Memory and Registers

W3100A Top level Memory Map



W3100A internal register and memory are comprised of 512 byte Control Registers and 16KB data buffer as displayed in the diagram above.

- 0x0000 ~ 0x00FF: Space for Control Registers
- 0x0100 ~ 0x01FF: Space for Shadow Registers
- 0x0200 ~ 0x3FFF: Not used (This space can be used by other devices)
- 0x4000 ~ 0x5FFF: Tx Data Buffer
- 0x6000 ~ 0x7FFF: Rx Data Buffer

Tx data buffer is the memory used for MCU transmission, and MCU can execute ‘write’ but cannot execute ‘read’. Rx data buffer is the memory used for MCU reception, and MCU can execute ‘read’ but cannot

execute ‘write’. In order to verify the active status of Tx data buffer and Rx data buffer, MCU can execute both write and read by setting the memory test mode (setting up of C1_CR memory test bit). In memory test mode, however, W3100A cannot execute proper transmission and reception of data. Memory test mode must be terminated for normal operation of W3100A.

■ Description of Functions

1. Initialization of W3100A

In order to use W3100A, the basic registers that are required to run W3100A need to be set up. The basic registers include GAR (Gateway Address Register), SMR (Subnet Mask Register), SHAR (Source Hardware Address Register), and SIPR (Source IP Address Register).

GAR, SMR and SIPR are the network information on which W3100A is operated, and the registers need to be set according to the operating environment. SHAR is the Hardware address to be used at the MAC layer of W3100A, and the address already provided to the manufacturer is used.

After appropriately setting up above registers, W3100A can activate in the network by executing the sys_init command. Activation can be verified by using Ping (ICMP Echo request).

2. TCP Protocol

TCP is a connection-oriented protocol. By using three-way handshaking method in executing the connection set-up and termination process, reliable data transmission and reception are assured.

TCP Initialization Process

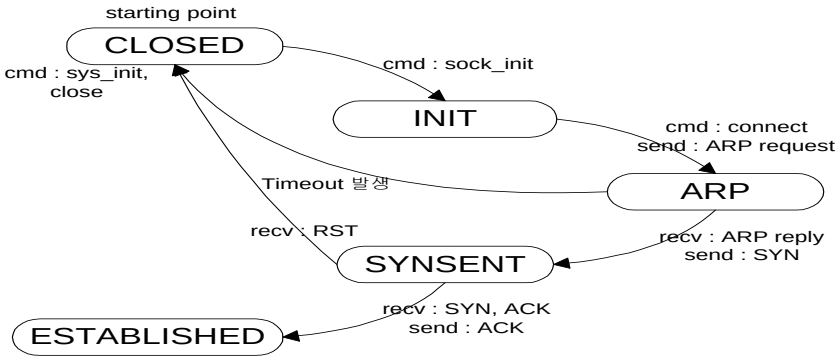
In order to use W3100A TCP, the protocol field of the corresponding channel's Cx_SOPR (Socket Option/Protocol Register of Channel x) needs to be set up as SOCK_STREAM(0x01). After the channel is activated by sock_init command, Cx_TW_PR (Tx Write Pointer Register of Channel x), Cx_TR_PR (Tx Read Pointer Register of Channel x), and Cx_TA_PR (Tx Ack Pointer Register of Channel x) need to be initialized with same value.

TCP Connection Set-up Process

In W3100A, the TCP connection process as directed by Connect or Listen command is processed internally. Sending SYN Packet as directed by Connect command is called active open, and standing by for SYN Packet from peer as directed by Listen command is called passive open.

Active open.

TCP Client mode that knows the IP address and port number of the destination, and the connection set-up is made ahead.



Above diagram illustrates the connection set-up process using active open. Each status can be verified through the socket status register of the corresponding channel.

a. CLOSED state: channel is initialized by executing sys_init or close command

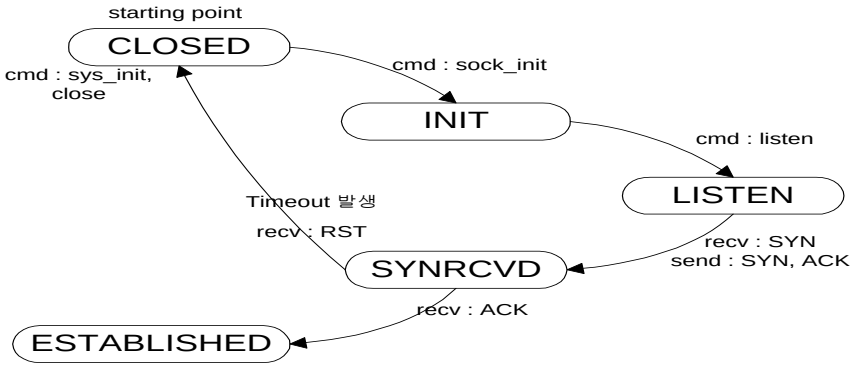
b. INIT state: sets the port number (source port register) to be used in the channel and activates the channel by executing the sock_init command

c. ARP state: In order to set up connection, MCU sets the Destination IP, Destination Port register and executes the connect command. Based on this command, W3100A changes to this state and transmits ARP request packet. When ARP reply packet is received from the peer under this state, it changes to SYNSENT state and transmits SYN packet. In case no reply is received from the peer, re-transmission is made. When no reply is received within the designated timeout duration, timeout occurs and it changes to CLOSED state.

d. SYNSENT state: In this state, W3100A transmits SYN packet and stands by to receive SYN,ACK packet from the peer. In case appropriate SYN,ACK packet is received, W3100A transmits ACK packet and completes the connection set-up to become changed to ESTABLISHED state. In case no appropriate SYN,ACK packet is received from the peer, re-transmission of SYN Packet is made. When no reply is received within the designated timeout duration, timeout occurs and it changes to CLOSED state. Also, if the peer has no application standing by in passive mode, the peer receives RST packet and changes to CLOSED state.

Passive open.

In TCP Server mode, stands by for connection set-up from the peer under the Listen command, and the connection set-up is accepted when requested.



a. CLOSED state: channel is initialized by executing sys_init or close command

b. INIT state: sets the port number (source port register) to be used in the channel and activates the channel by executing the sock_init command

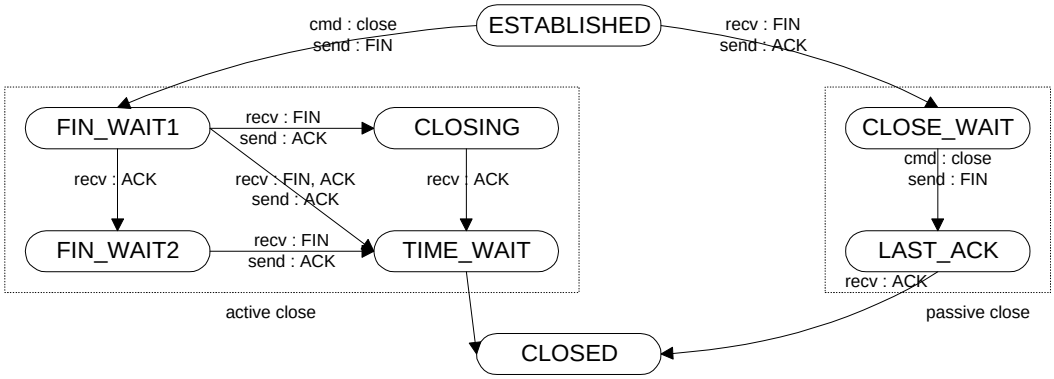
c. LISTEN state: stands by for connection set-up from the peer. When SYN packet for the corresponding port is received from the peer, SYN,ACK packet is transmitted and changes to SYNRCVD state.

d. SYNRCVD state: SYN,ACK packet is transmitted and stands by for ACK from the peer. When reply from the peer is received, it changes to ESTABLISHED state, and when no reply is received, SYN,ACK Packet is re-transmitted and changes to CLOSED state upon occurrence of timeout or receipt of RST packet.

TCP Connection Termination Process

In line with the connection set-up process, TCP connection termination process also uses three-way handshaking method.

Sending FIN after receiving Close command from the application is called active close, and closing after receiving FIN from the peer is called passive close.



Active close

After completing data transmission and reception, the application uses the close command to terminate the connection set-up. When the connection is terminated under the close command in such ESTABLISHED state, it is called active close, and the process is illustrated in the left-hand side of the diagram above.

FIN_WAIT1 state: changes from the established state under the close command and transmits FIN packet. Changes to FIN_WAIT2 when ACK for FIN is received fro the peer. Transmits ACK and changes to CLOSING state when FIN is received from the peer. Transmits ACK and changes to TIME_WAIT when FIN,ACK is received. In case of no reply, re-transmission is made, and if no reply is received until timeout occurs, changes to CLOSED state.

FIN_WAIT2 state: stands by for FIN from the peer. In this state, W3100A does not receive data from the peer, and if data is received, connection set-up is immediately terminated through RST. This is because W3100A does not process additional data in half-close state.

CLOSING state: produced when the application closes simultaneously. Changes to TIME_WAIT when ACK is received from the peer.

TIME_WAIT state: viewed as 2MSL (Maximum Segment Lifetime) WAIT State by TCP. In case FIN is resent when the peer cannot receive ACK, there is a function where TCP resends the last ACK. In case TCP connection is in 2MSL wait state, there is another function where other client, server is blocked from using this connection. In W3100A, considering the limited resource and for efficient use of the channel, it changes from this state to CLOSED state without waiting.

Passive close

In passive close, FIN is received from the peer to close in the ESTABLIHSED state as illustrated in the right-hand side of the above diagram.

CLOSE_WAIT state: changed from ESTABLISHED state by receiving FIN from the peer. Transmits ACK for FIN and creates closed interrupt at MCU. By processing the interrupt, MCU executes the close command to W3100A and completes the connection close. But, if data to be sent still are left, that is TW_PR value is not equal to TA_PR value, you should not issue the close command but wait until timeout occurs or ignore close procedure and make progress next step like sock_init command.

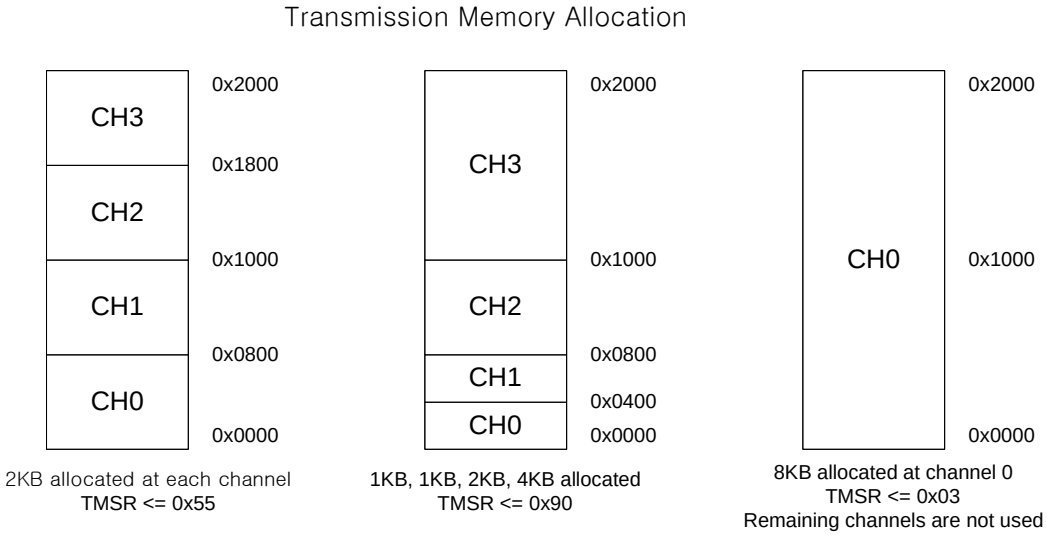
LAST_ACK state: when close command is handed down by MCU, FIN is transmitted and stands by for ACK. If no ACK is received, FIN Packet is re-transmitted. If no reply is received until timeout occurs, it changes to CLOSED state.

TCP Data Transmission and Reception

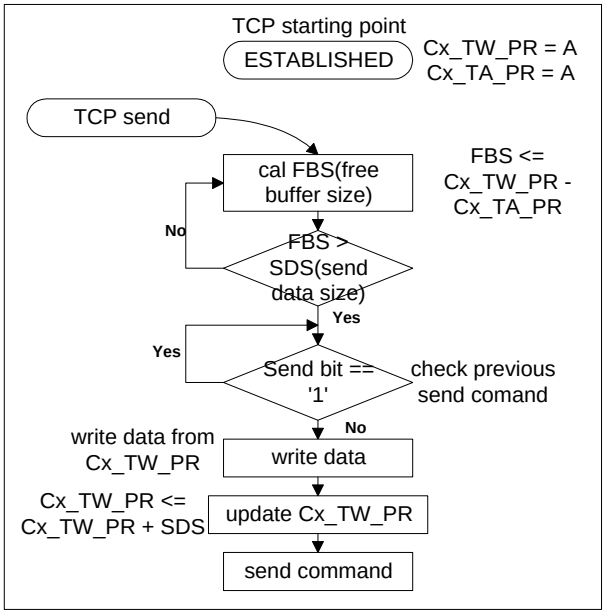
Unlike UDP, TCP data transmission and reception is possible only after the connection set-up is made. W3100A has exclusive memory for data transmission and reception, 8KB for transmission and 8KB for reception. This memory can be set up as 1KB, 2KB, 4KB and 8KB by using RMSR (Rx data Memory Size Register) and TMSR (Tx data Memory Size Register).

TCP Transmission Memory Size Set-up

W3100A transmission memory is comprised of 8KB in total, and the size can be assigned for each channel through TMSR register. An example of TMSR and each memory size is illustrated in the diagram below. When the memory size from channel 0 exceeds 8KB, all ensuing memory is ignored.



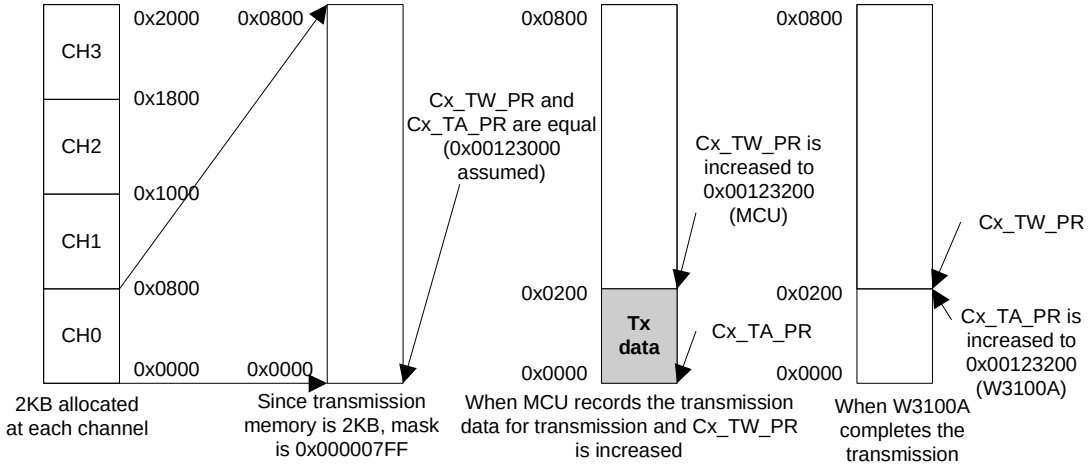
TCP Data Transmission Process



In order to execute W3100A TCP transmission, 4Byte pointer of Cx_TW_PR (Tx Write Pointer Register of Channel x) and Cx_TA_PR (Tx Ack Pointer Register of Channel x) is used. Cx_TW_PR is the pointer that writes the data to be transmitted from MCU, and Cx_TA_PR is the pointer that completed W3100A transmission. Cx_TW_PR and Cx_TA_PR become equal after connection set-up is made. In active open, they are equally initialized under the sock_init command from MCU. In passive open, one is initialized by

the other. The difference between the pointers become the actual FBS (free buffer size). Data is recorded from Cx_TW_PR according to such size, and when the data recording is complete, Cx_TW_PR is increased according to the size of the recorded data and executes the send command.

Pointer Management during TCP Transmission



Above diagram illustrates the change in Cx_TW_PR and Cx_TA_PR when actual data transmission is made after 2KB of transmission memory is set at CH0.

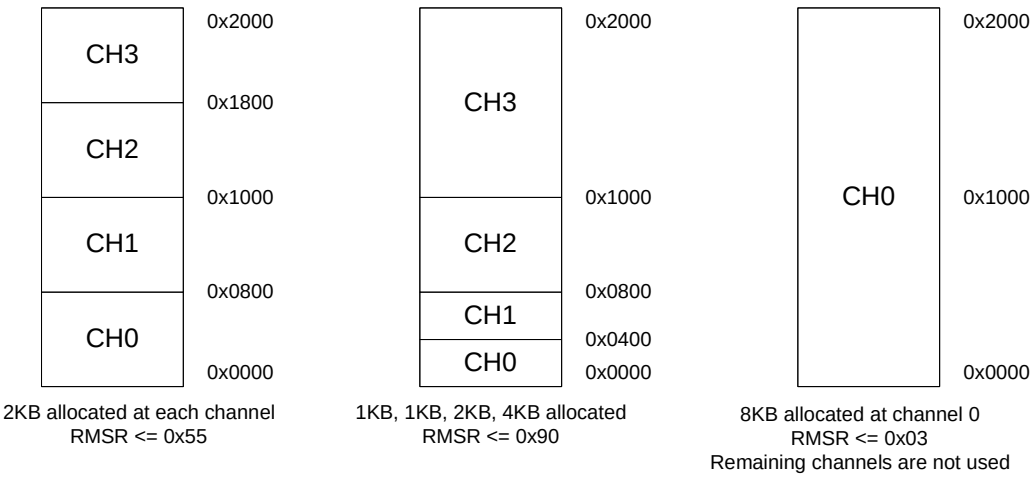
TCP Reception Memory Size Set-up

Receiving memory of W3100A has the same structure of the transmission memory and operated in same method.

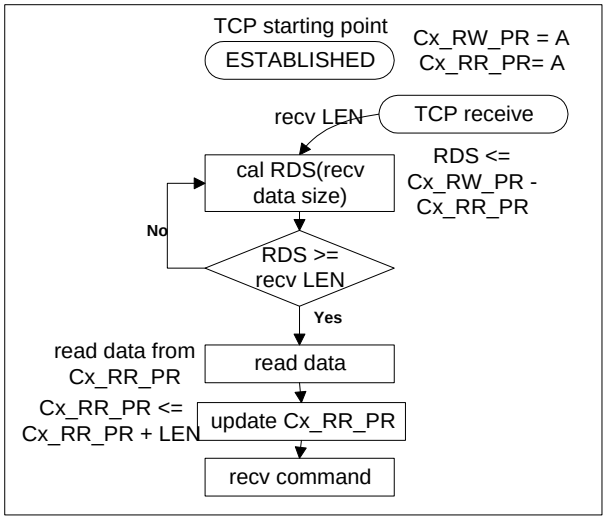
The memory is comprised of 8KB in total, and the size can be assigned for each channel through RMSR (Rx data Memory Size register). An example of RMSR and each memory size is illustrated in the diagram below.

When the memory size from channel 0 exceeds 8KB, all ensuing memory is ignored.

Reception Memory Allocation

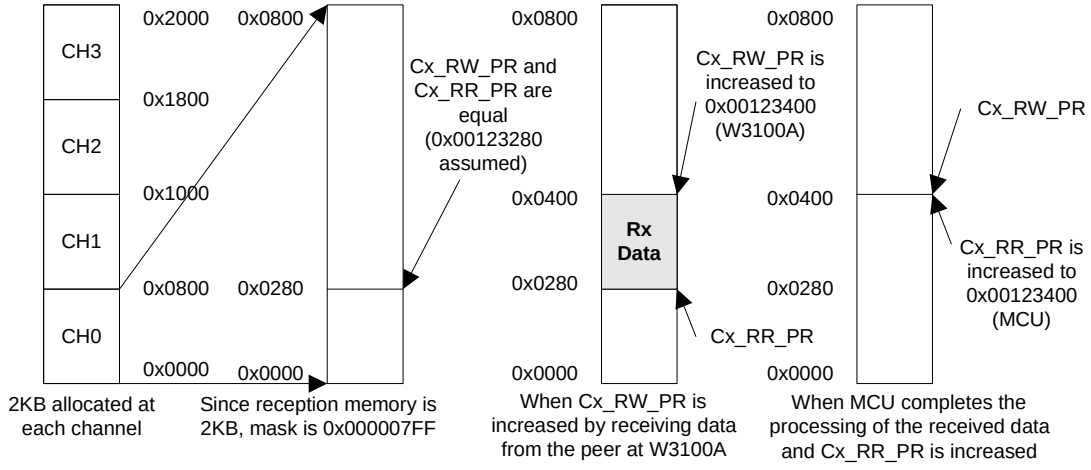


TCP Data Reception Process



TCP data reception by W3100A is illustrated in the above diagram. In W3100A, when data is received from the peer, the data is recorded as reception memory from Cx_RW_PR (Rx Write Pointer Register of Chnnel x), Cx_RW_PR is increased according to the size of the received data when the reception is complete, and then MCU is interrupted to report a data reception. Through interrupt or polling, MCU compares Cx_RW_PR and Cx_RR_PTR (Rx Read Pointer Register of Channel x), and when data reception is observed, the size of the received data is first calculated and Cx_RR_PR is increased after the data is read and processed from Cx_RR_PR. Finally, recv command is executed to report W3100A that the processing of the received data is complete.

Pointer Management during TCP Reception



Above diagram illustrates the change in Cx_RW_PR and Cx_RR_PR when actual data is received after 2KB of reception memory is set at CH0.

TCP Retry Time Adjustment

W3100A uses IRTR (Initial Retry Time-value Register) and RCR (Retry Count Register) to adjust the timer to be used in re-transmission of TCP.

TCP re-transmission is executed when the initial retry timer expires, and the retry timer is reset at the value of * 2. Such a process is repeated according to the RCR value, and in the last retry, timeout interrupt occurs and then gives up.

Formula of timeout value:

IRTR: Initial Retry Time-value Register

RCR: Retry Count Register

IRTR * 100us = start timeout second

Total timeout value until give-up = (IRTR * 100us) *(2^{RCR}-1)

Internally, the default value of IRTR is 0x07D0 and RCR is 0x06, where initial retry takes place at 200ms and the retry frequency becomes 6. Therefore, unless these registers are revised, retry is made at 200ms, 600ms, 1400ms, 3000ms, 6200ms, 12600ms each and gives up at the final 12600ms.

3. UDP Protocol

UDP is a connectionless protocol. No connection set-up or termination process is needed, thereby creating lesser load.

UDP Initialization Process

In order to use UDP of W3100A, the Cx_SOPR (Socket Option/Protocol register of Channel x) protocol field of the corresponding channel needs to be set as SOCK_DGRAM(0x02) before socket initialization. Unlike TCP, data transmission and reception is possible at UDP without any connection set-up process.

UDP Data Transmission and Reception

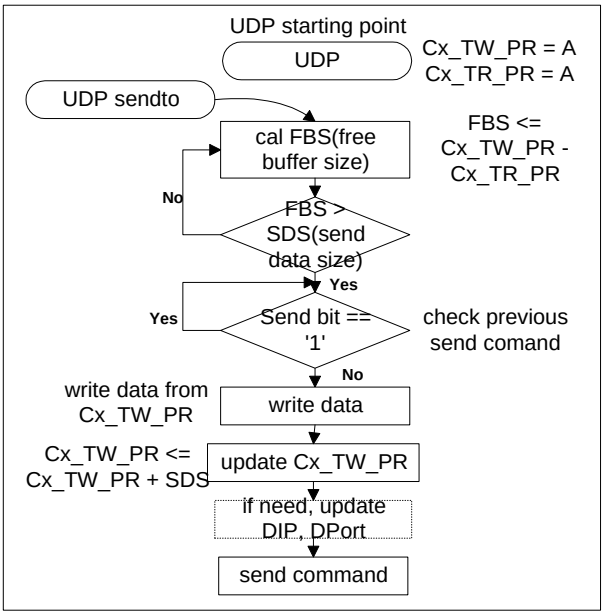
UDP transmission is activated similarly to TCP. All data received at its port can be received, and MCU needs to analyze the header information of the data to verify transmitting IP and port to confirm the corresponding data before processing.

Set-up of transmission and reception memory size is identical to TCP.

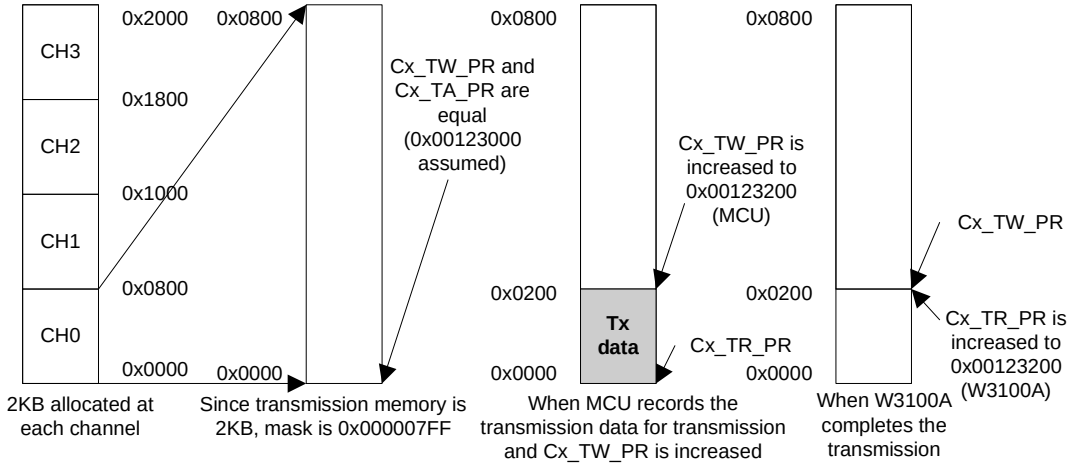
UDP Data Transmission

UDP transmission is activated similarly to TCP. Calculating the free buffer size of the memory, data copying and Cx_TW_PR are identical. The difference is the usage of Cx_TR_PR instead of Cx_TA_PR. Another difference is that destination IP and port need to be set. In other words, if the destination IP and

port set prior to this transmission are different to the destination IP and port to be used for the transmission, the values need to be updated before executing the send command.



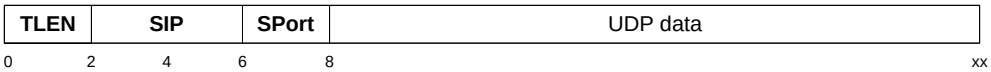
Pointer Management during UDP Transmission



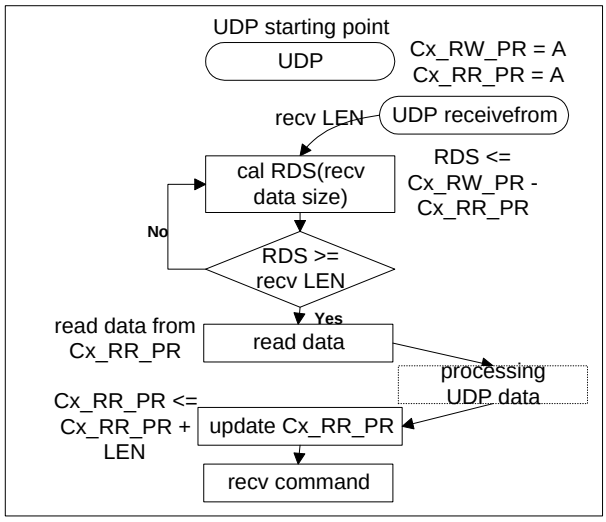
Above diagram illustrates the change in Cx_TW_PR and Cx_TA_PR when actual data transmission is made after 2KB of transmission memory is set at CH0.

UDP Data Reception

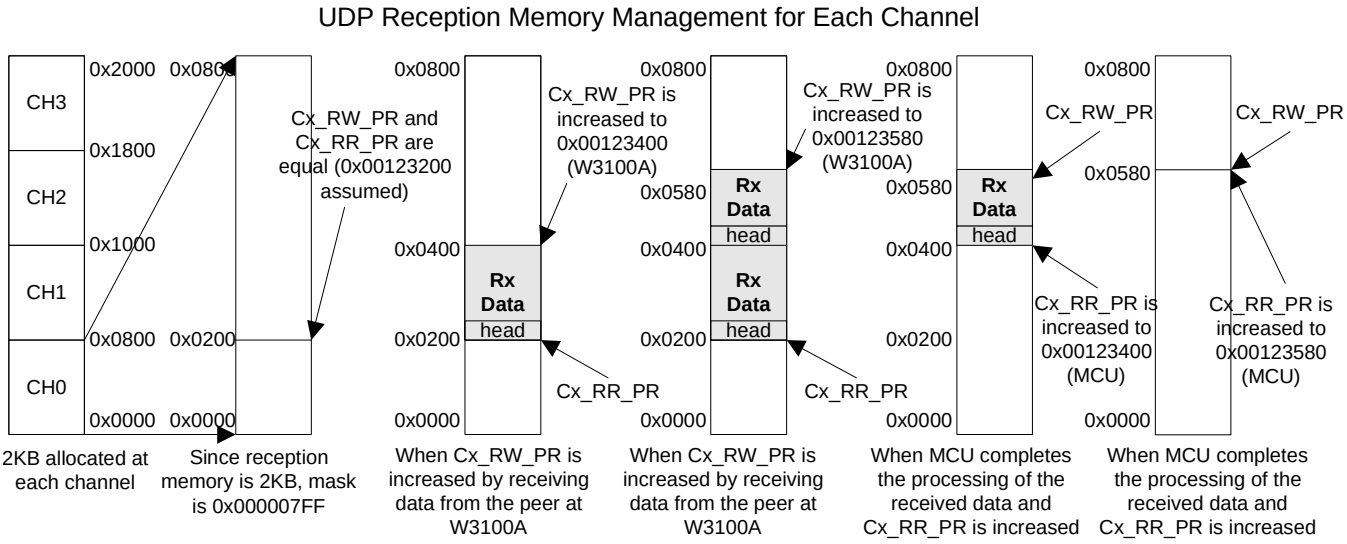
W3100A's UDP reception is similar to TCP reception. The difference is that the header information for UDP processing is included in the received data in addition to the data. The header is structured as below.



The header is comprised of (1) TLEN Field displaying the size of 2Byte Header + data, (2) 4Byte SIP displaying the sender IP that transmitted UDP data, and (3) SPort displaying the sender Port. MCU uses such information to determine whether the data needs to be processed by MCU before processing.



Above diagram illustrates the MCU processing flow for UDP data. Excluding the header processing for the UDP data, the basic flow is identical to TCP reception.



At UDP reception buffer, many different data may exist between Cx_RW_PR and Cx_RR_PR. Therefore, the header information is used to differentiate and process such data. Above diagram illustrates the process where 2 UDP's receive the data and processed by MCU.

4. IP Layer RAW Mode

W3100A's IPL_RAW(IP Layer RAW) mode is used in processing protocols (e.g., ICMP, etc.) other than TCP and UDP as provided by W3100A.

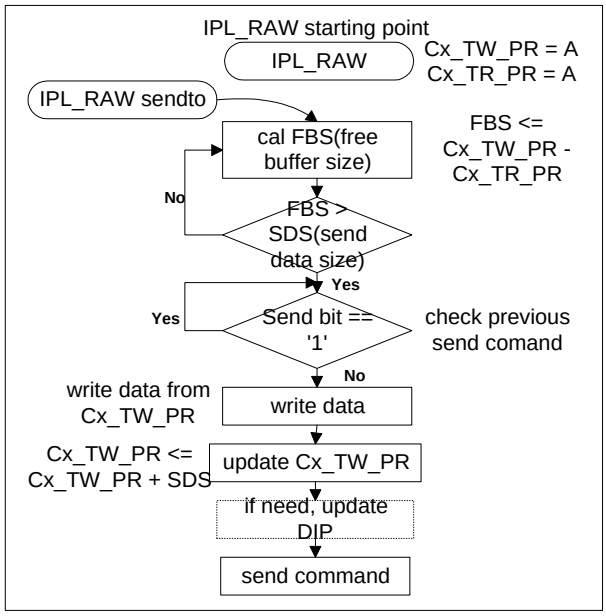
IPL_RAW Mode Initialization Process

In order to use W3100A's IPL_RAW Mode, the protocol value of the IP Layer to be used (e.g., 0x01 in case of ICMP) needs to be set as Cx_IPR (IP Protocol Register of Channel x), and the Cx_SOPR (Socket Option/Protocol register of Channel x) protocol field of the corresponding channel needs to be set as SOCK_IPL_RAW(0x03) before socket initialization (sock_init command). As in UDP, data transmission and reception is possible when the corresponding channel is initialized.

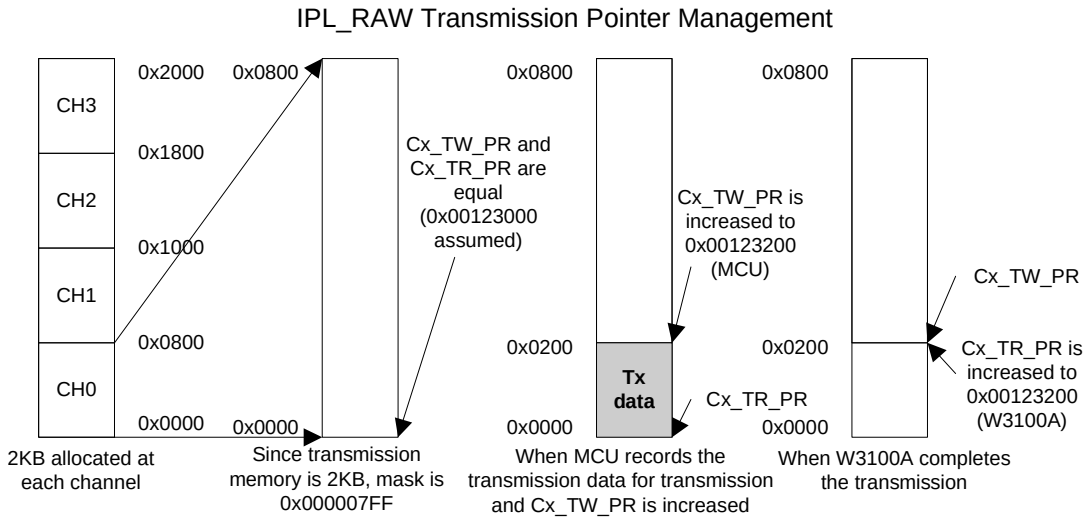
IPL_RAW Mode Data Transmission and Reception

Transmission in IPL_RAW Mode is activated similarly to UDP, and the reception is made for the same Protocol data as Cx_IPR.

IPL_RAW Mode Data Transmission



For reception in IPL_RAW Mode, calculating the free buffer size of the reception memory, data copying and usage of Cx_TW_PR, Cx_TR_PR are identical to UDP, and the destination IP needs to be set. Unlike UDP, however, no port needs to be set. As in UDP, if the destination IP set prior to this transmission is different to the destination IP to be used for the transmission, the value need to be updated before executing the send command.



Above diagram illustrates the change in Cx_TW_PR and Cx_TA_PR when actual data transmission is made after 2KB of transmission memory is set at CH0.

IPL_RAW Mode Data Reception

Reception in W3100A's IPL_RAW Mode is similar to UDP reception. As in UDP, header information is included in the received data in addition to the data, and the header is structure as below.

The header information IPL_RAW mode contains (1) TLEN identical to UDP TLEN displaying the total length of data size + header length, and (2) 4Byte SIP displaying the sender IP that transmitted UDP data. Both are used with Cx_RW_PR and Cx_RR_PR to process the received data. The process is made identically to the UDP reception data processing.

5. MAC Layer RAW Mode

In MACL_RAW (MAC Layer RAW) mode, W3100A is used as other general NIC (Network Interface Controller), and W3100A's TCP/IP module is not used in this mode.

When TCP/IP is used through W3100A, the number of channels is limited to 4.

For systems using more than 4 channels simultaneously, W3100A uses this mode and S/W TCP/IP can be processed in the higher drive.

If MAC Layer RAW Mode is used, W3100A uses Channel 0 only and other channels are ignored.

MACL_RAW Mode Initialization Process

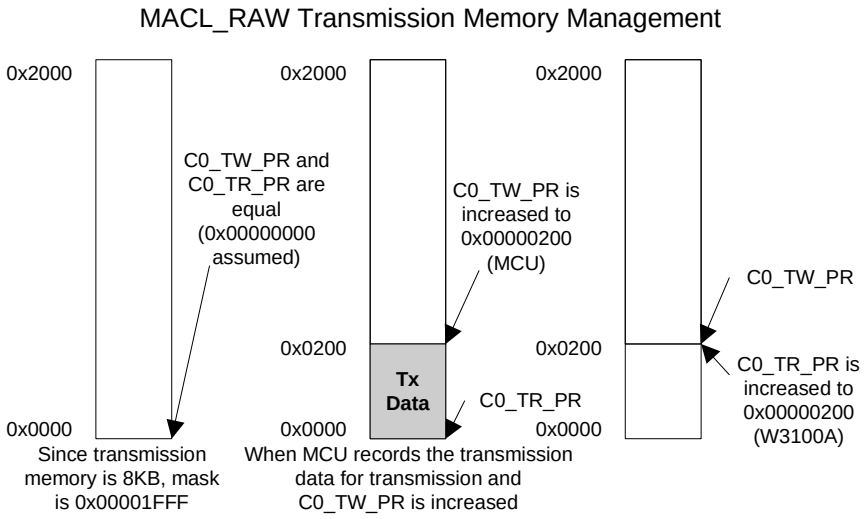
In MACL_RAW Mode, data transmission and reception is possible after the protocol field of C0_SOPR (Socket Option/Protocol register of Channel 0) is set as SOCK_MACL_RAW(0x04) and by using sock_init command of channel 0. Option values to be set at C0_SOPR include ERR, B, P bit. ERR bit allows the reception of Packet with error, B bit allows the reception of Broadcast Packet, and P bit allows the reception of Promiscuous Packet (Packet with specific MAC address).

MACL_RAW Mode Data Transmission and Reception

Transmission in MACL_RAW Mode is activated similarly to UDP, and the data reception is made according to C0_SOPR.

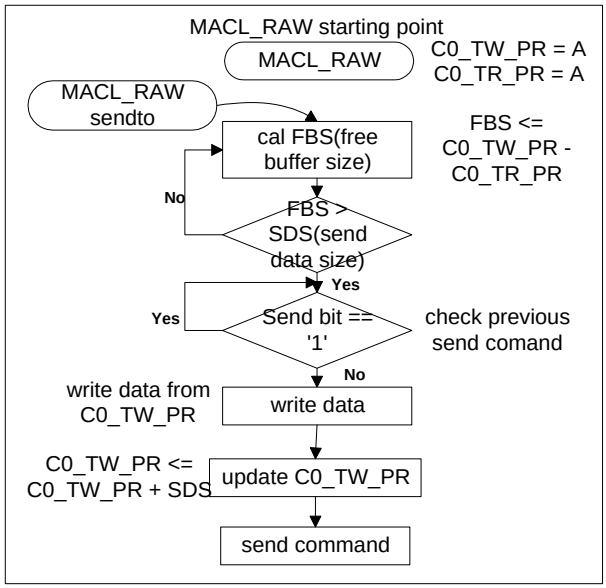
MACL_RAW Mode Data Transmission Management

MACL_RAW Mode's transmission memory management uses a single Channel only, that is Channel 0, and uses 8KB of transmission memory.

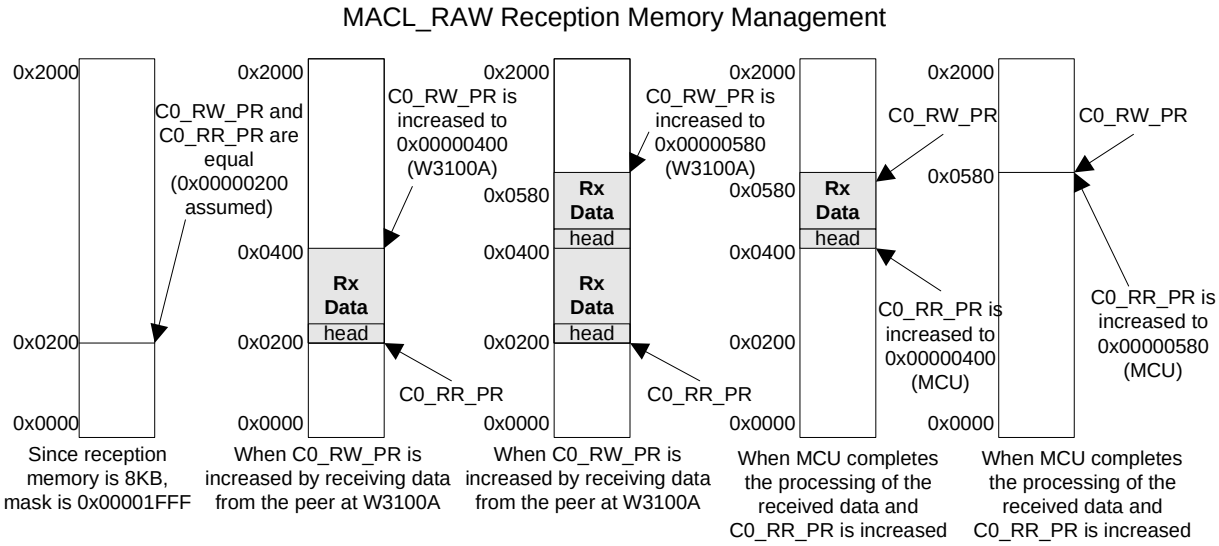


MACL_RAW Mode Data Transmission

For transmission in MACL_RAW Mode, calculating the free buffer size of the transmission memory, data copying, and usage of C0_TW_PR, C0_TR_PR are identical to UDP, but unlike UDP, no destination IP and Port set up is needed. In other words, in MACL_RAW Mode, all protocol processing is made by MCU, and such information is included in the transmission Frame.



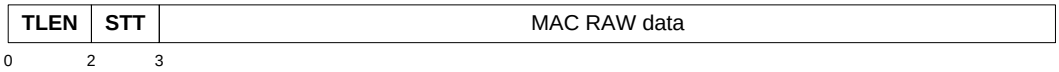
MACL_RAW Mode Data Reception Management



MACL_RAW Mode's Reception memory management uses 1 channel only, therefore all 8KB is allocated to Channel 0. Above diagram illustrates the processing of 2 data after C0_WR_PR and C0_RR_PR are equally initialized as 0x00000200.

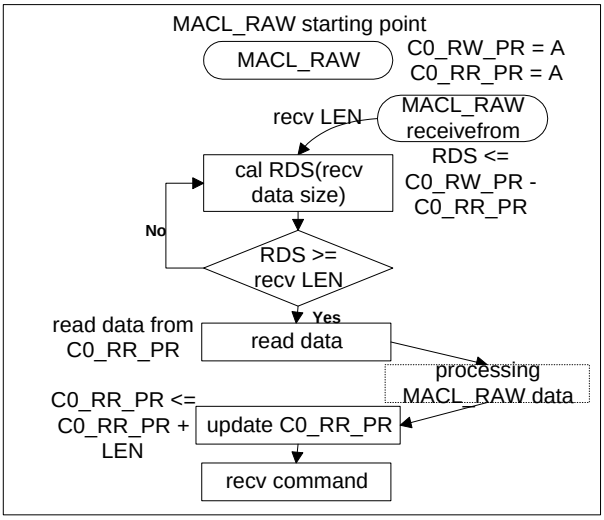
MACL_RAW Mode Data Reception

In W3100A's MACL_RAW Mode, the reception of the set packets are made according to the receive options as set at C0_SOPR. As in UDP, header information is included in the received data in addition to the data, and the header is structure as below.



As shown in the above diagram, the header information IPL_RAW mode contains (1) TLEN identical to UDP TLEN displaying the total length of data and header length, and (2) 1 Byte STT displaying the status of the received data. The description of the received data is recorded at STT Byte as below.

7	6	5	4	3	2	1	0	Meaning
1	x	x	x	x	x	x	x	Destination H/W Address of the received packet is identical to SHAR (Source H/W Address Register)
x	x	1	x	x	x	x	x	Reception of Broadcasting Packet
x	x	x	1	x	x	x	x	Reception of Packet with Error



As in UDP, the reception module uses C0_RW_PR and C0_RR_PR to process the received data. The process is identical to UDP received data process.

■ Application Information

W3100A MCU I/F is comprised of Direct Bus I/F Mode, Indirect Bus I/F Mode, and I^2C I/F Mode

1. Relationship between MCU Bus I/F Mode and Mode pin (M[2:0])

M2	M1	M0		Description
0	0	0	Clocked mode	Mode using a clock to analyze the MCU bus signal (default mode)
0	0	1	External clocked mode	Mode using an external clock to analyze the MCU bus signal
0	1	0	Non-clocked mode	Mode directly using the MCU bus signal
0	1	1	I^2C mode	Mode using I^2C
1	X	X	Test mode	Mode used at the manufacturing plant for testing Not to be used by the average user

Refer to the timing by mode diagram for detailed access timing for each mode. (Page 50 – page 58)

Considering the I/F between the MCU and W3100A, Clocked mode, External clocked mode, or Non-clocked mode can be used if Bus I/F is provided by the MCU. Timing is slightly different in each mode, and MCU bus timing also needs to be considered before selecting the mode.

The first thing to consider in selecting the mode is the relationship among /CS, /RD, /WR, and Address[14:0] granted to the W3100A.

If /CS is low and /RD, /WR, and Address[14:0] have valid signals over 100ns, use Clocked mode. This mode is the default mode and is suitable for most systems.

However, some systems require faster access (access time of less than 100ns) to the W3100A, in which case either the External clocked mode or the Non-clocked mode should be selected.

Non-clocked mode should be used in a situation where /CS granted to the W3100A drops to low as in the access timing diagram on page 54 and /RD or /WR goes low after 10ns. Otherwise, External clocked mode should be used.

In the External clocked mode, the access time differs depending on the clock granted to EXT_CLK. Therefore, caution should be taken in using External clocked mode. Refer to the access timing diagram with EXT_CLK of 50MHz (page 52).

Generally, Clocked mode is recommended after adjusting the access time for the W3100A to 100ns or higher (page 50), rather than using the External clocked mode.

Further details about each mode are as follows:

Clocked mode is used in internal functions and in the MCU I/F using the clock granted to W3100A, and the access timing may vary depending on the frequency (basic frequency: 25MHz). Clock basically uses 25MHz, but when a different frequency clock other than 25MHz is used, the access time of the MCU bus I/F and timeout occurrence time can change.

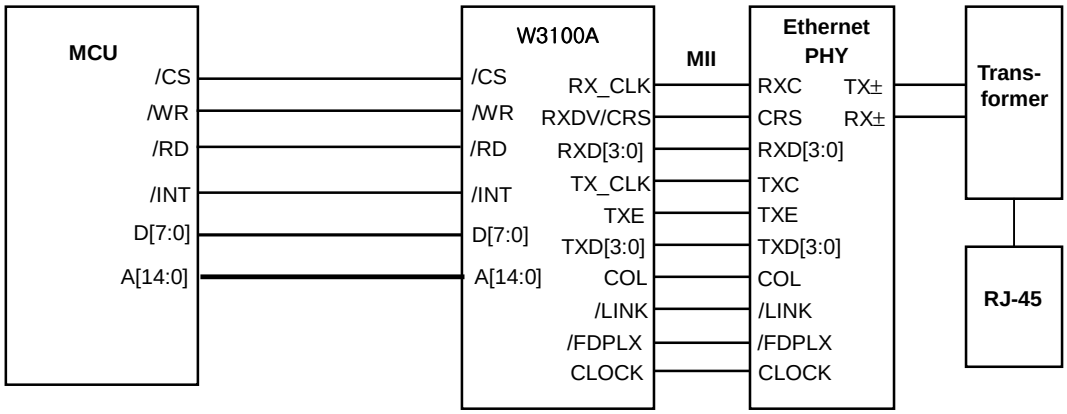
In External clocked mode, the internal functions of W3100A are executed by the clock granted to the clock

pin, and the MCU bus I/F is activated by the External clock.

In Non-clocked mode, the internal functions of W3100A are executed by the clock granted to the clock pin, and the MCU bus I/F is activated by /CS, /RD, /WR of MCU. As shown in the timing diagram, and unlike Clocked mode and External clocked mode, a timing condition exists between /CS, /RD, and /WR.

I^2C I/F mode is used when an MCU connected to W3100A supports I^2C I/F but does not support bus I/F. The basic frequency to be used for I^2C is activated by the clock granted to the clock pin and both 100KHz mode and 400KHz mode are supported.

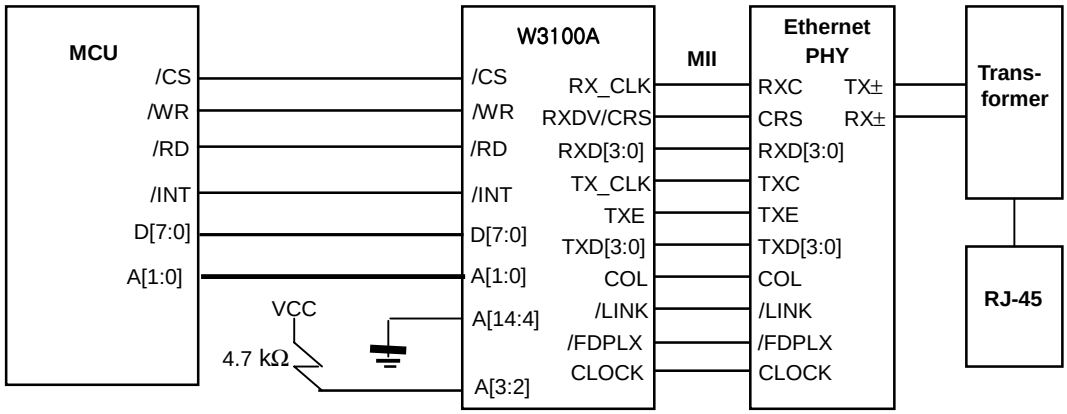
2. Direct Bus I/F Mode.



As illustrated in the above diagram, Direct Bus I/F mode uses a 15-bit address line and 8-bit data line, /CS, /RD, /WR, /INT, which is identical to the I/F of the existing W3100A.
Since the access timing changes from Clocked mode (page 50) to External clocked mode (page 52) and to Non-clocked mode (page 54), the MCU bus access timing needs to be considered.

3. Indirect Bus I/F Mode.

Indirect Bus I/F mode uses a 2-bit address line and 8-bit data line, /CS, /RD, /WR, and /INT.
Of the address lines, A[14:4] is grounded as '0' and A[3:2] is pulled up (4.7 kΩ) to '1'.



A[3:2] can be set up as '1' by software after connecting the additional 4 address lines of A[3:0] and grounding the remaining A[14:4].

Registers related to indirect bus I/F mode are listed below:

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0x0C	IDM_OR	IND_EN						L/B	AUTO_INC
0x0D	IDM_AR0	indirect bus I/F mode address0 register							
0x0E	IDM_AR1	indirect bus I/F mode address1 register							
0x0F	IDM_DR	indirect bus I/F mode data register							

IDM_OR (Indirect Mode Option Register) is comprised of IND_EN that determines the use of Indirect Bus I/F mode, the L/B bit that determines byte ordering during address set-up, and the AUTO_INC bit that automatically increases the address.

IDM_AR0 (Indirect Mode Address0 Register) designates the higher byte of the address, and IDM_AR1 (Indirect Mode Address1 Register) designates the lower byte of the address. The ordering of IDM_AR0 and IDM_AR1 varies depending upon the L/B bit set-up.

IDM_DR (Indirect Mode Data Register) displays the data to be accessed.

In order to use Indirect Bus I/F mode, IND_EN of IDM_OR needs to be setup first.

In order to read the internal register value of W3100A, the address of the internal register to be accessed needs to be written to IDM_AR0,1. Later, when IDM_DR is read, the value of the register to be accessed is read.

In order to write the internal register value of W3100A, the address of the internal register to be accessed needs to be written at IDM_AR0,1. Later, the value can be written at IDM_DR.

The L/B bit of IDM_OR that sets up the order when accessing IDM_AR0,1 functions as below:

If L/B bit of IDM_OPT register = '0'

MSB LSB

IDM_ADDR0(0x0D)	IDM_ADDR1(0x0E)

If L/B bit of IDM_OPT register = '1'

LSB MSB

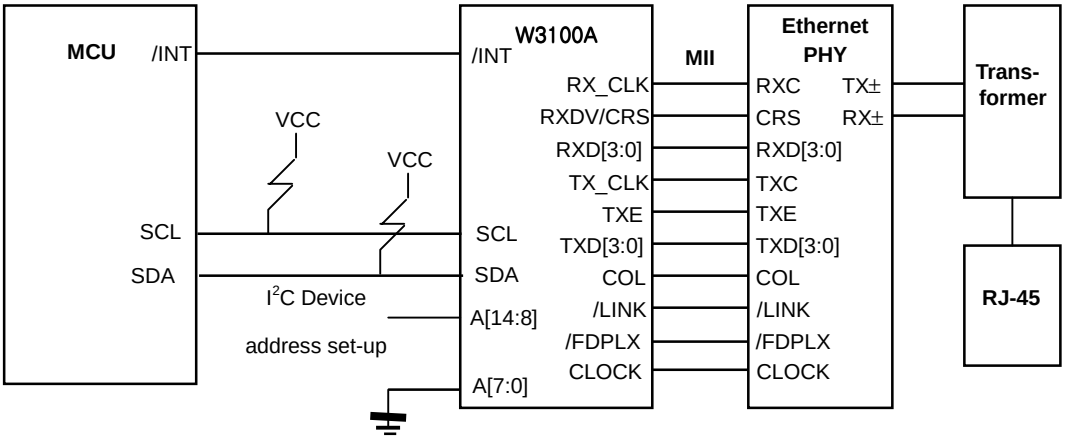
IDM_ADDR0(0x0D)	IDM_ADDR1(0x0E)

AUTO_INC Bit allows register access without setting IDM_AR0,1 for each access when continuous access is made, where IDM_AR0,1 is automatically incremented by 1 with each IDM_DR access.

4. I²C I/F Mode.

I²C I/F uses SDA, SCL to provide serial data transmission and reception in I²C mode between MCU and W3100A.

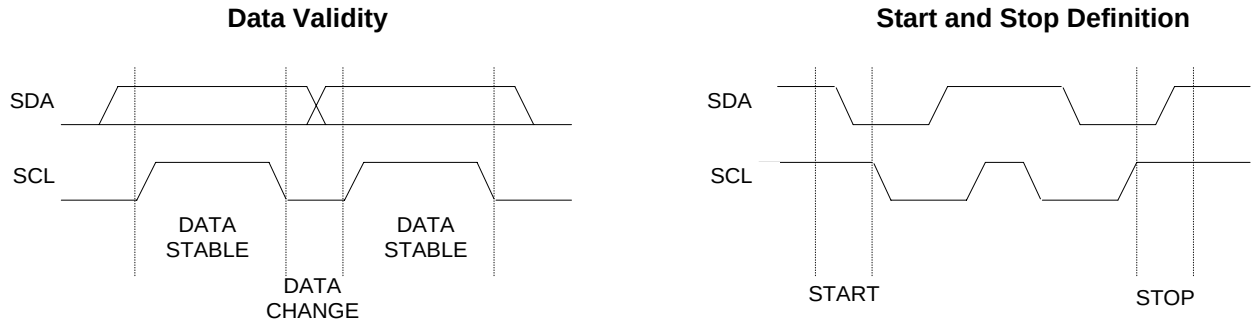
As a clock line, SCL needs to be provided by MCU, and SDA is used as the line to transfer data and address between MCU and W3100A.



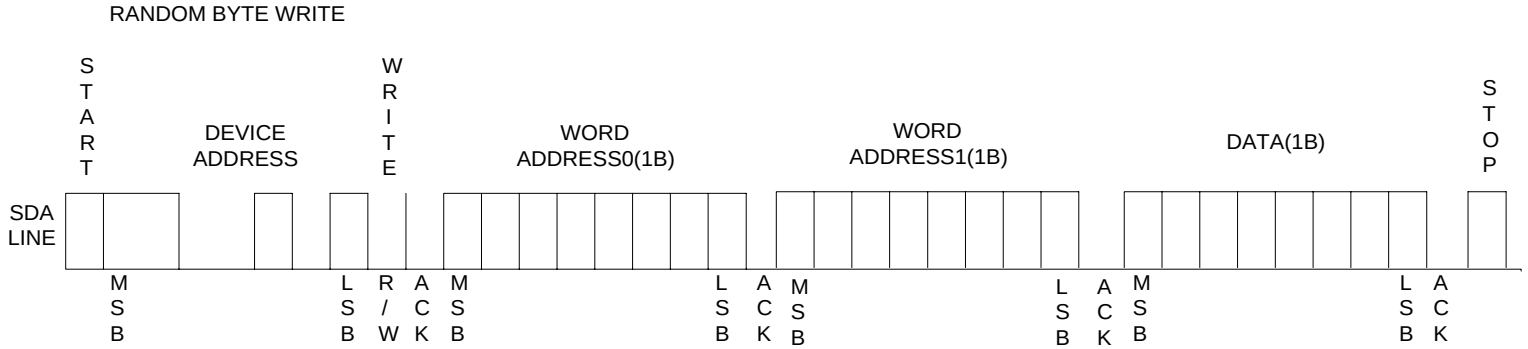
A simple block diagram as shown above illustrates the connection between MCU and W3100A using I²C.

As illustrated above, it is recommended to set the device address for I²C by using A[14:8], provide a pull-up of 4.7 kΩ for SCL and SDA lines externally, and then ground the remaining address A[7:0].

In order to synchronize MCU and W3100A, I^2C I/F creates the condition for START before transmitting and receiving data, and the condition for STOP is created after the completion of data transmission and reception. When SDA line becomes low while SCL line is high, it becomes the signal for the condition for START. When SDA line becomes high while SCL line is high, it becomes the signal for the condition for STOP.

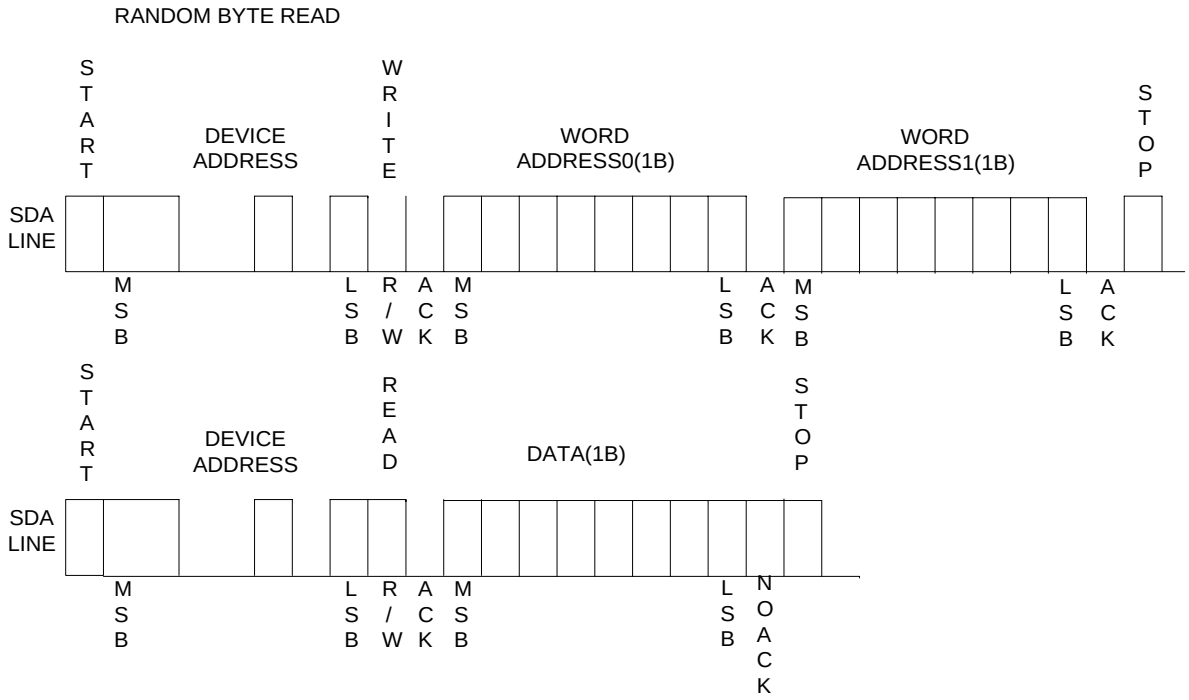


Access method includes random read/write access for 1Byte-unit access and sequential read/write access for sequential access.

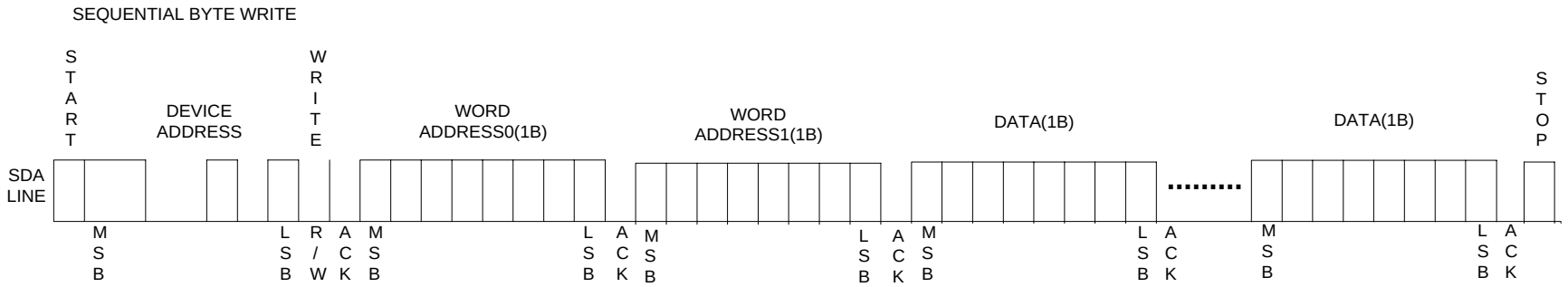


1Byte-unit access is functions as illustrated in the above diagram, and random byte write sends in the order of START, DEVICE ADDRESS, 2 Byte address of the actual register to be accessed, and the actual data before sending STOP.

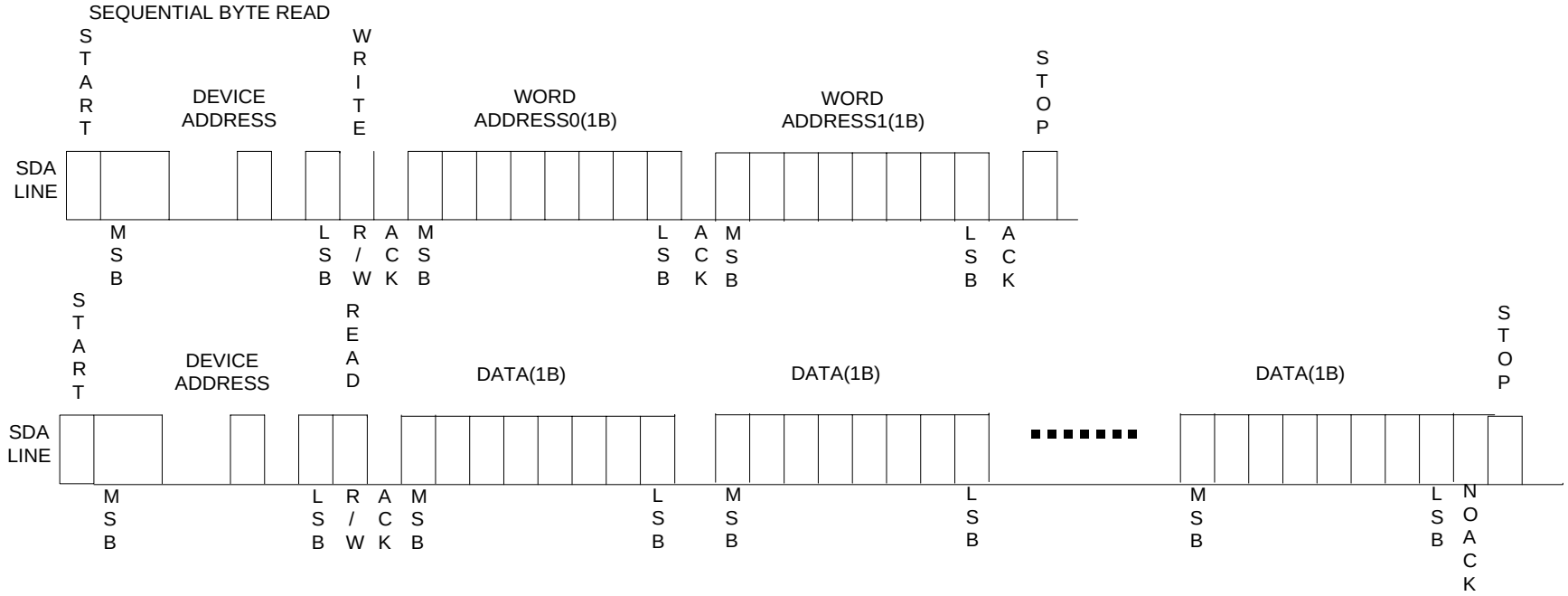
In order to set up the address of the register to be accessed, random byte read first sends START, DEVICE ADDRESS and 2 Byte register address before sending STOP. Later, once again, START, DEVICE ADDRESS is sent, actual data is read and STOP is sent.



Sequential byte write sequentially sends START, DEVICE ADDRESS, 2 Byte address of the actual register to be accessed, and data to be written before sending STOP. As a result, MCU can designate the address of the data to be written once, which allows for the data to be written sequentially and the address to automatically increase by 1.



In order to set up the address of the register to be accessed, sequential byte read first sends START, DEVICE ADDRESS and 2 Byte register address before sending STOP. Later, once again, START, DEVICE ADDRESS is sent and actual DATA is sequentially read before sending STOP. Also at this time, the address automatically increases by 1.


SEQUENTIAL BYTE READ
START
DEVICE ADDRESS
WRITE
WORD ADDRESS0(1B)
WORD ADDRESS1(1B)
STOP
SDA LINE
MSB
LSB
R/W
ACK
MSB
LSB
ACK
MSB
LSB
ACK
START
DEVICE ADDRESS
READ
DATA(1B)
DATA(1B)
DATA(1B)
STOP
SDA LINE
MSB
LSB
R/W
ACK
MSB
LSB
MSB
LSB
MSB
LSB
NOACK

5. Physical Layer Interface

W3100A is used for linking with an Ethernet Physical Layer Device (RealTek RTL8201, National DP83843, SMSC 83C180, 83C183, LevelOne LXT905, etc.). Such Physical Layer Devices generally use the Media Interface (MI) for configuration and the Media Independent Interface (MII) for data transfer when interfacing with the Ethernet MAC Layer.

5.1 Media Interface (MI)

The MI is the serial I/F for reading/writing the internal register of the Physical Layer Device to set up or check the configuration of the Physical Layer Device, and its pin is generally denoted as MDC or MDIO.

Since the W3100A does not configure or check the configuration of the Physical Layer Device through the MI, the systems equipped with the W3100A configure the Physical Layer Device using its pin (in general, SPEED, DUPLEX, Auto Negotiate, etc.), so that the W3100A can refer to the configuration status through the W3100A pins (/LINK, /SERIAL, /FDPLX). In other words, /LINK, /SERIAL, and /FDPLX pins of the W3100A must be connected to reflect the status of the Physical Layer Device. In general, the Physical Layer Device reports the network configuration status using the LED Pins, which are connected with the /LINK, /SERIAL, and /FDPLX Pin of the W3100A.

Besides, depending on the systems, it may be necessary for the MCU to configure or check the configuration of the Physical Layer Device, in which case the MCU must access the Physical Layer Device directly through the MI.

5.2 Media Independent Interface (MII)

The MII handles the data transfer between the W3100A and the Physical Layer Device.

The MII is composed of TX_CLK, TXE, and TXD[0:3] signals for sending data and RX_CLK, RXDV, RXD[0:3], and COL signals for receiving data.

When sending data from the W3100A, TXE and TXD[0:3] are output in synchronization with the falling edges of TX_CLK input from the Physical Layer Device because Physical Layer Devices generally recognize the rising edges of TX_CLK.

When receiving data, in general, the Physical Layer Devices output RXDV, RXD[0:3], and COL signals in synchronization with the falling edges of RX_CLK, so the W3100A recognizes the signals at the rising edges of RX_CLK.

There are serial MII that use 1-bit data signals and nibble MII that use 4-bit data signals.

5.2.1 serial MII

This mode is the I/F for 10Mbps Physical Layer Devices, and is composed of 1-bit TXD and RXD. TX_CLK and RX_CLK use a 10MHz cycle.

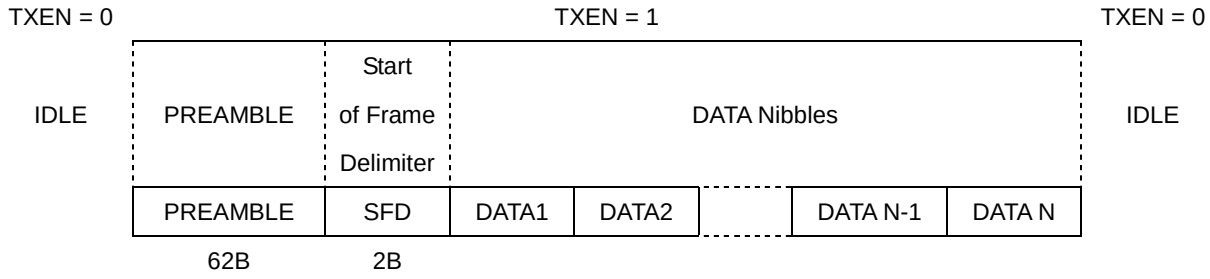
When linking the W3100A with the Physical Layer Device in this mode, RXD and TXD of the Physical Layer Device must be connected to RXD[0] and TXD[0] of the W3100A.

5.2.2 nibble MII

This mode is the I/F for 10/100Mbps Physical Layer Devices, and is composed of 4-bit TXD[0:3] and RXD[0:3]. TX_CLK and RX_CLK use a 2.5MHz cycle at 10Mbps and 25MHz at 100Mbps.
When linking the W3100A with the Physical Layer Device in this mode, RXD[0:3] and TXD[0:3] of the Physical Layer Device must be connected to RXD[0:3] and TXD[0:3] of the W3100A.

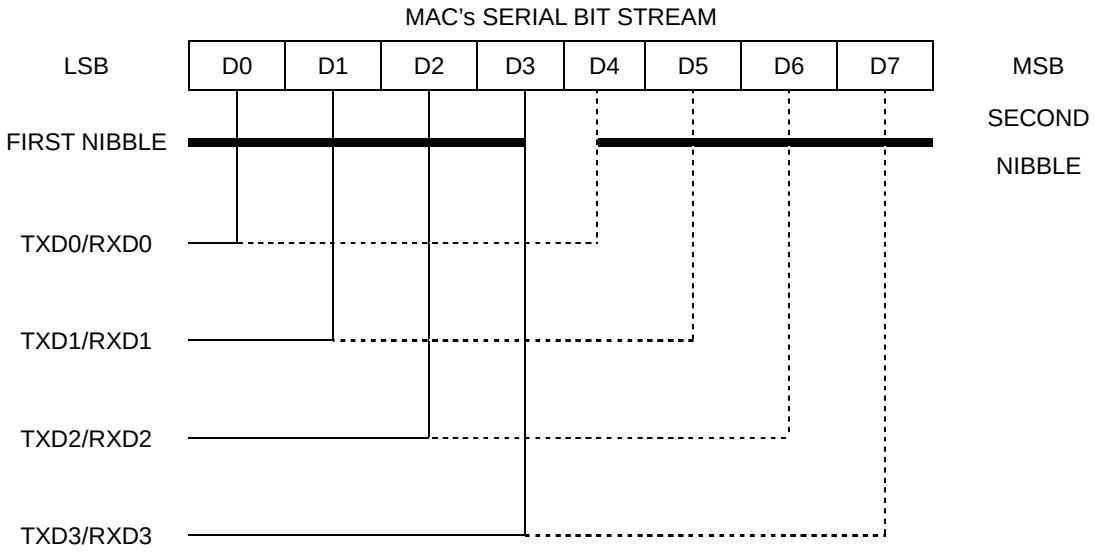
The W3100A supports both the serial mode and the nibble mode.

5.2.3 MII Frame Format



PREAMBLE	1 0 1 0 … … 62 bits long
SFD	1 1
DATA	Between 64 – 1518 data bytes
IDLE	TXEN = 0

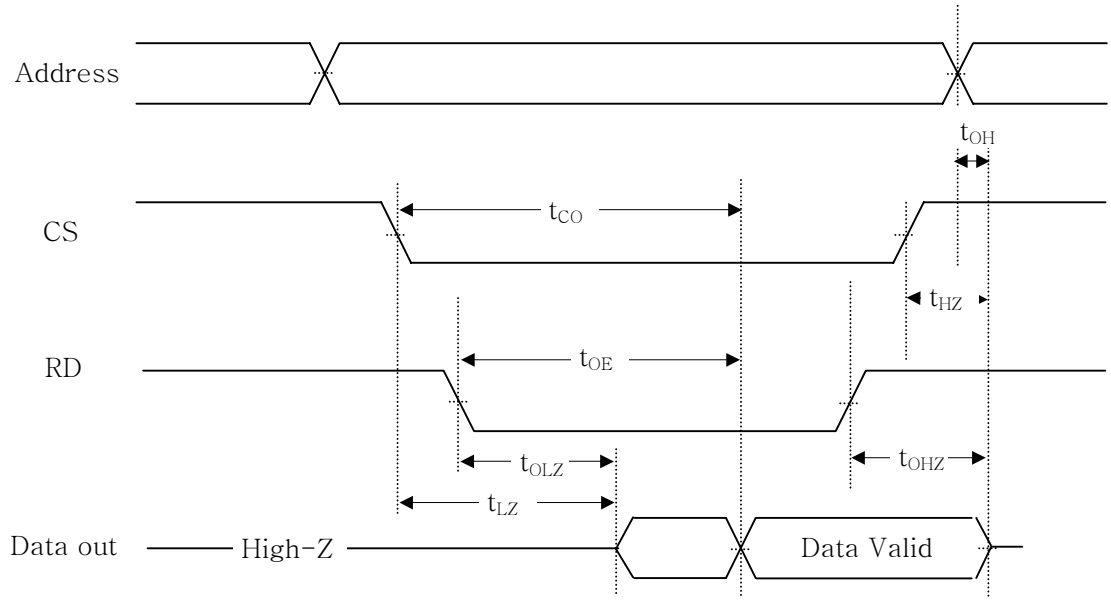
5.3.4 Nibble MII Order



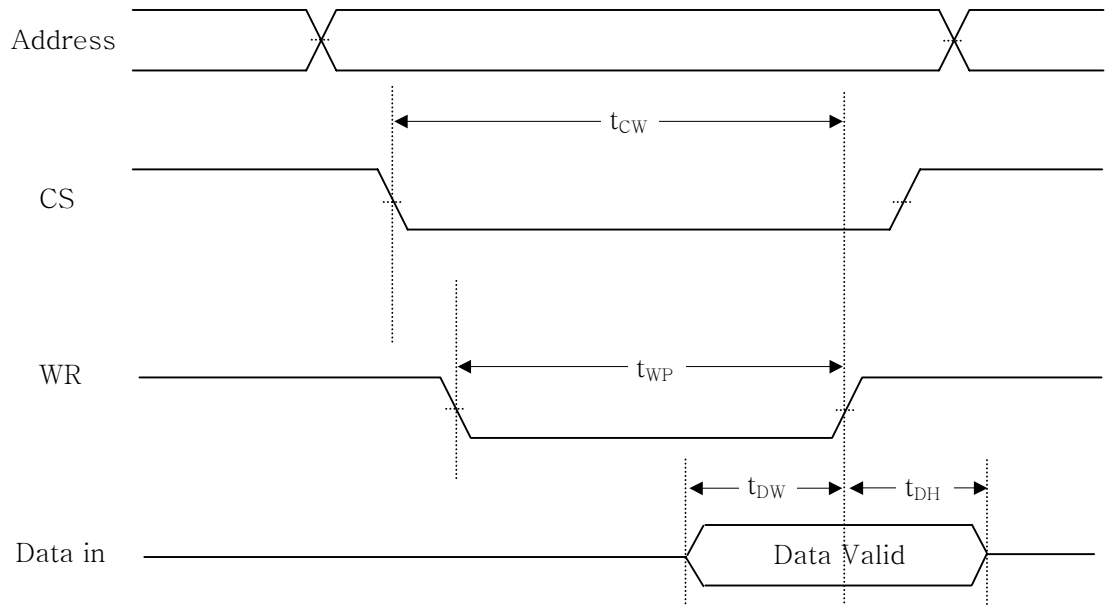
■ Timing Diagrams

1. Clocked mode(CLOCK = 25MHz)

TIMING WAVEFORM OF Register/Memory READ CYCLE



TIMING WAVEFORM OF Register/Memory WRITE CYCLE



(Note: Valid period of Address signal should be larger than or equal to assertion period of CS signal.)

AC Characteristics

Direct Mode

		Symbol	Speed		Units
			Min	Max	
Read	Chip select to output	t_{CO}		73	ns
	Output enable to valid output	t_{OE}		73	ns
	Chip select to low-Z output	t_{LZ}	13	54	ns
	Output enable to low-Z output	t_{OLZ}	13	54	ns
	Chip disable to high-Z output	t_{HZ}	2	6	ns
	Output disable to high-Z output	t_{OHZ}	2	6	ns
	Output hold from address change	t_{OH}	0		ns
Write	Chip select to end of write	t_{CW}	56		ns
	Write pulse width	t_{WP}	56		ns
	Data to write time overlap	t_{DW}	24		ns
	Data hold from write time	t_{DH}	7		ns

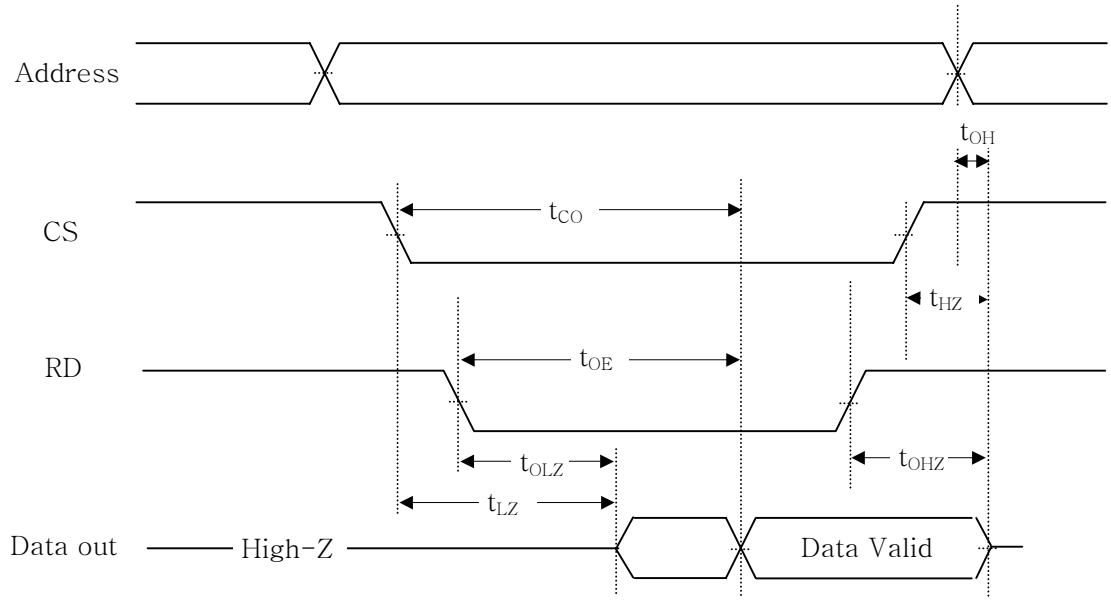
Indirect Mode

		Symbol	Speed		Units
			Min	Max	
Read	Chip select to output	t_{CO}		81	ns
	Output enable to valid output	t_{OE}		81	ns
	Chip select to low-Z output	t_{LZ}	13	54	ns
	Output enable to low-Z output	t_{OLZ}	13	54	ns
	Chip disable to high-Z output	t_{HZ}	2	6	ns
	Output disable to high-Z output	t_{OHZ}	2	6	ns
	Output hold from address change	t_{OH}	0		ns
Write	Chip select to end of write	t_{CW}	56		ns
	Write pulse width	t_{WP}	56		ns
	Data to write time overlap	t_{DW}	24		ns
	Data hold from write time	t_{DH}	7		ns

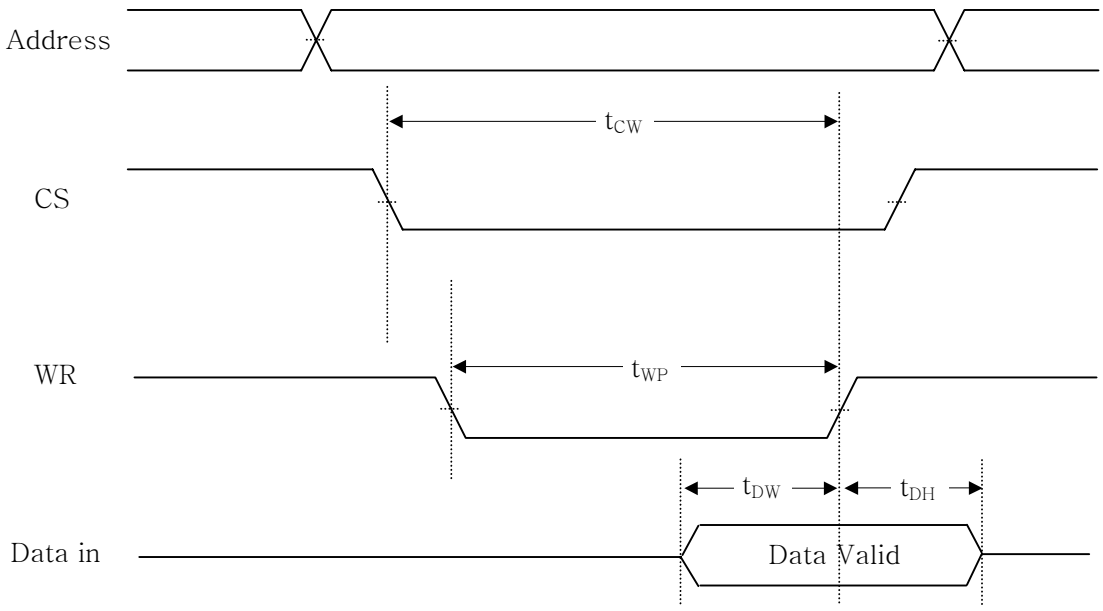
(Note: Above data is based on simulation.)

2. External clocked mode(EXT_CLK = 50MHz)

TIMING WAVEFORM OF Register/Memory READ CYCLE



TIMING WAVEFORM OF Register/Memory WRITE CYCLE



(Note: Valid period of Address signal should be larger than or equal to assertion period of CS signal.)

AC Characteristics

Direct Mode

		Symbol	Speed		Units
			Min	Max	
Read	Chip select to output	t_{CO}		53	ns
	Output enable to valid output	t_{OE}		53	ns
	Chip select to low-Z output	t_{LZ}	13	34	ns
	Output enable to low-Z output	t_{OLZ}	13	34	ns
	Chip disable to high-Z output	t_{HZ}	2	6	ns
	Output disable to high-Z output	t_{OHZ}	2	6	ns
	Output hold from address change	t_{OH}	0		ns
Write	Chip select to end of write	t_{CW}	36		ns
	Write pulse width	t_{WP}	36		ns
	Data to write time overlap	t_{DW}	24		ns
	Data hold from write time	t_{DH}	7		ns

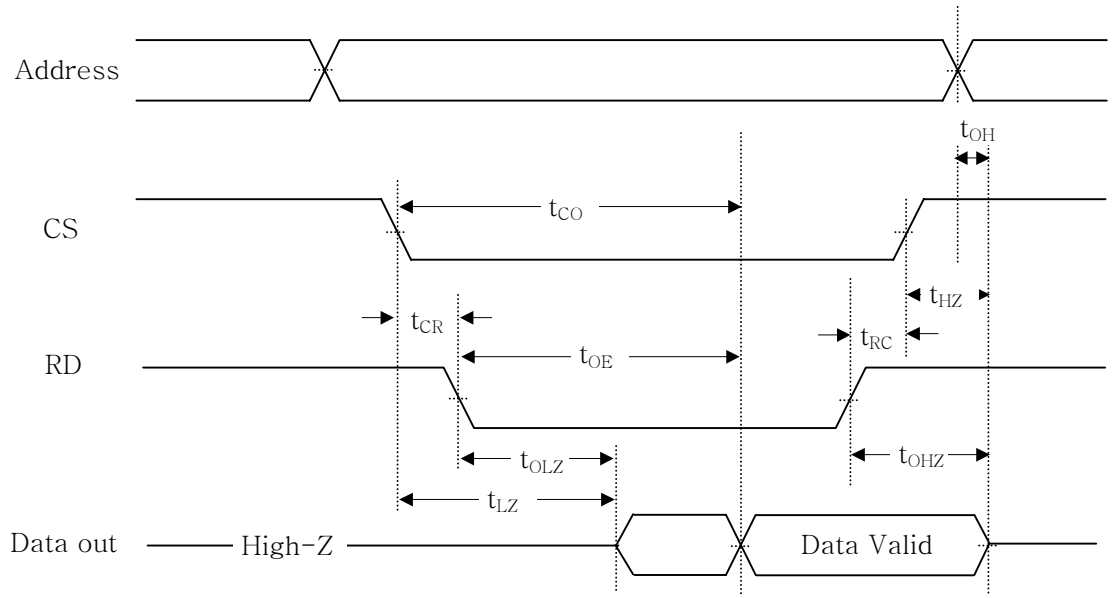
Indirect Mode

		Symbol	Speed		Units
			Min	Max	
Read	Chip select to output	t_{CO}		61	ns
	Output enable to valid output	t_{OE}		61	ns
	Chip select to low-Z output	t_{LZ}	13	34	ns
	Output enable to low-Z output	t_{OLZ}	13	34	ns
	Chip disable to high-Z output	t_{HZ}	2	6	ns
	Output disable to high-Z output	t_{OHZ}	2	6	ns
	Output hold from address change	t_{OH}	0		ns
Write	Chip select to end of write	t_{CW}	36		ns
	Write pulse width	t_{WP}	36		ns
	Data to write time overlap	t_{DW}	24		ns
	Data hold from write time	t_{DH}	7		ns

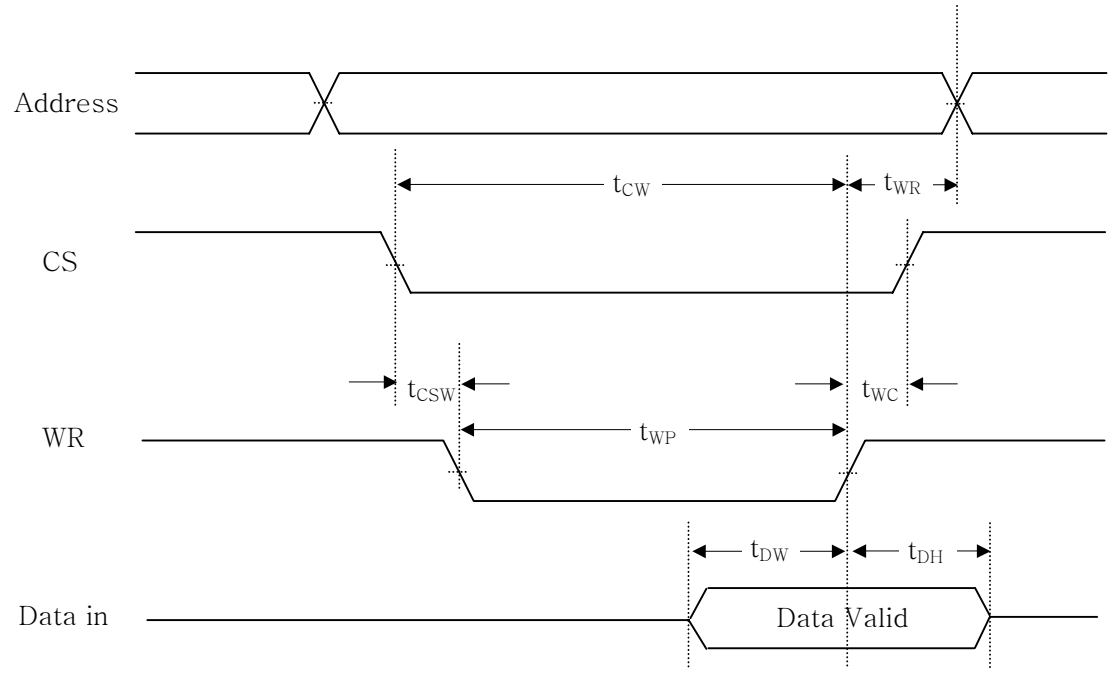
(Note: Above data is based on simulation.)

3. Non-clocked mode(CLOCK = 25MHz)

TIMING WAVEFORM OF Register/Memory READ CYCLE



TIMING WAVEFORM OF Register/Memory WRITE CYCLE



(Note: Valid period of Address signal should be larger than or equal to assertion period of CS signal.)

AC Characteristics

Direct Mode

		Symbol	Speed		Units
			Min	Max	
Read	Chip select to output	t_{CO}		11	ns
	Chip select low to Read low	t_{CR}	5		ns
	Output enable to valid output	t_{OE}		10	ns
	Chip select to low-Z output	t_{LZ}	4	8	ns
	Output enable to low-Z output	t_{OLZ}	4	8	ns
	Read high to Chip select high	t_{RC}	0		ns
	Chip disable to high-Z output	t_{HZ}	2	5	ns
	Output disable to high-Z output	t_{OHZ}	2	5	ns
	Output hold from address change	t_{OH}		5	ns
Write	Chip select to end of write	t_{CW}	5		ns
	Address set-up time	t_{AS}	0		ns
	Chip select low to write low	t_{CSW}	0		ns
	Write high to Chip select high	t_{WC}	0		ns
	Write pulse width	t_{WP}	5		ns
	Write recovery time	t_{WR}	1		ns
	Data to write time overlap	t_{DW}	4		ns
	Data hold from write time	t_{DH}	5		ns

(Note1: In Non-clocked mode, assertion period of CS signal should enclose assertion period of RD, WR signal. Especially, read signal should be asserted 5ns previously.

Note2: In case of Non-clocked mode, high speed working is possible because access time is short. But, you'd better consider MCU setup time to design in order to have bus access time more than enough compared to above data.

Note3: Above data is based on simulation.)

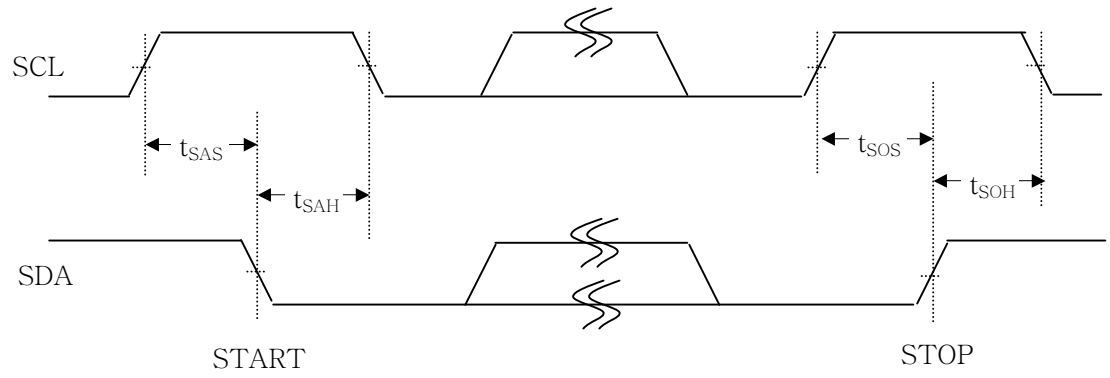
Indirect Mode

		Symbol	Speed		Units
			Min	Max	
Read	Chip select to output	t_{CO}		18	ns
	Chip select low to Read low	t_{CR}	5		ns
	Output enable to valid output	t_{OE}		18	ns
	Chip select to low-Z output	t_{LZ}	4	8	ns
	Output enable to low-Z output	t_{OLZ}	4	8	ns
	Read high to Chip select high	t_{RC}	0		ns
	Chip disable to high-Z output	t_{HZ}	2	5	ns
	Output disable to high-Z output	t_{OHZ}	2	5	ns
	Output hold from address change	t_{OH}		5	ns
Write	Chip select to end of write	t_{CW}	13		ns
	Address set-up time	t_{AS}	0		ns
	Chip select low to write low	t_{CSW}	0		ns
	Write high to Chip select high	t_{WC}	0		ns
	Write pulse width	t_{WP}	13		ns
	Write recovery time	t_{WR}	1		ns
	Data to write time overlap	t_{DW}	12		ns
	Data hold from write time	t_{DH}	5		ns

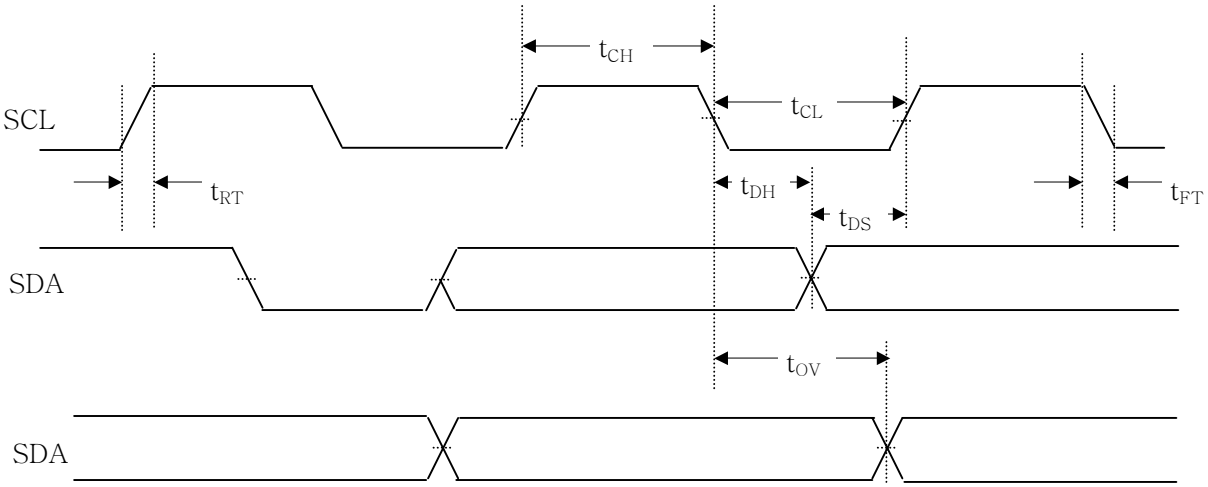
(Note: Above data is based on simulation.)

4. I²C mode(CLOCK = 25MHz)

I²C BUS START/STOP BITS TIMING



I²C BUS DATA TIMING



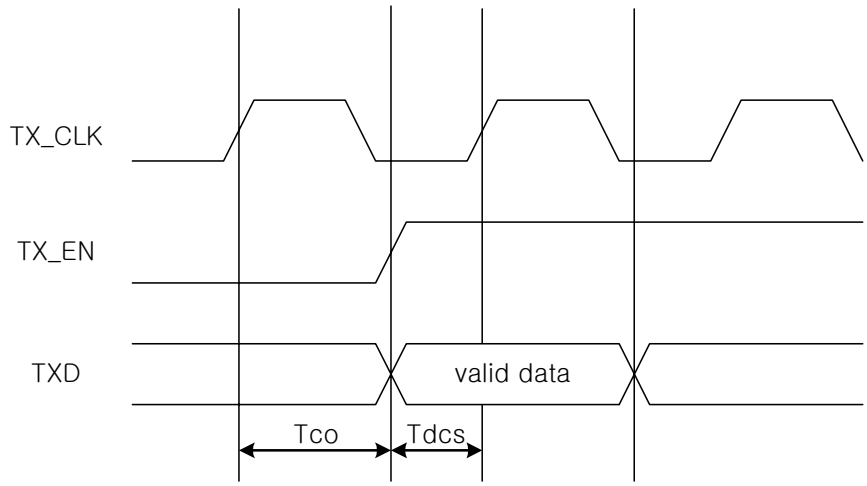
AC Characteristics

		Symbol	Speed		Units
			Min	Max	
START / STOP BITS TIMING	START condition setup time	t_{SAS}	40		ns
	START condition hold time	t_{SAH}	40		ns
	STOP condition setup time	t_{SOS}	40		ns
	STOP condition hold time	t_{SOH}	40		ns
BUS DATA TIMING	Clock Frequency, SCL	f_{SCL}		1	MHz
	SCL high time	t_{CH}	200		ns
	SCL low time	t_{CL}	300		ns
	SDA, SCL rise time	t_{RT}			ns
	SDA, SCL fall time	t_{FT}			ns
	Data input hold time	t_{DH}	0		ns
	Data input setup time	t_{DS}	0		ns
	Output valid from clock	t_{OV}	285		ns

(Note: Above data is based on simulation.)

5. Media Independent Interface (MII)

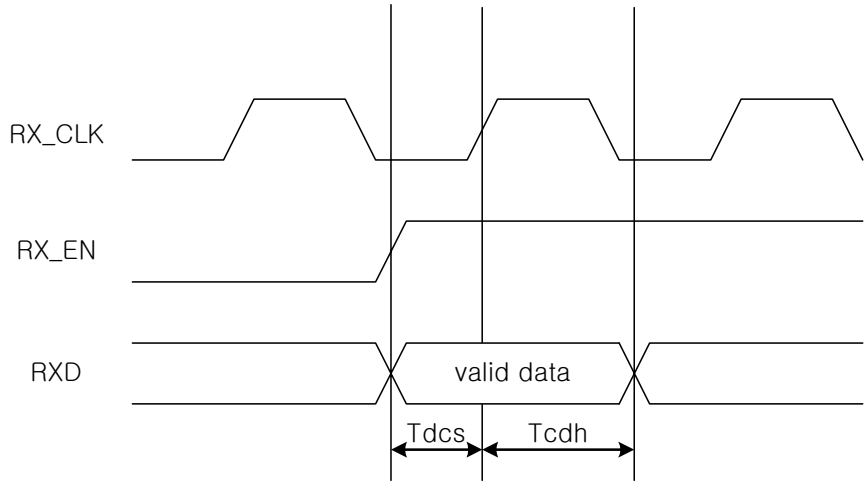
MII Tx TIMING



AC Characteristics

Parameter	Description	Notes	Min	Typ	Max	Units
Tco	TX_CLK to TXD, TX_EN	10Mbps	202	-	205	ns
Tdcs	TXD, TX_EN setup time to TX_CLK	10Mbps	195	-	198	ns
Tco	TX_CLK to TXD, TX_EN	100Mbps	22	-	25	ns
Tdcs	TXD, TX_EN setup time to TX_CLK	100Mbps	15	-	18	ns

MII Rx TIMING

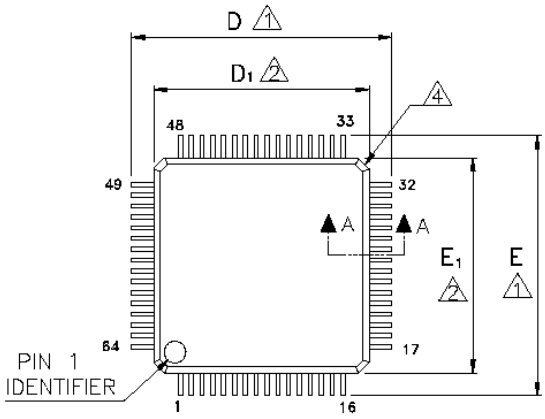


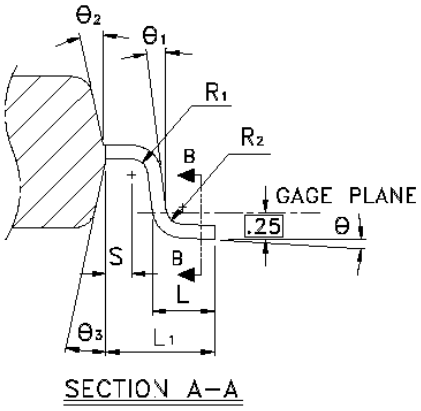
AC Characteristics

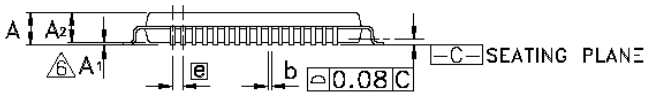
Parameter	Description	Notes	Min	Typ	Max	Units
Tdcs	Valid Data to RX_CLK (setup)	10Mbps	5	-	-	ns
Tcdh	RX_CLK to Valid Data (hold)	10Mbps	5	-	-	ns
Tdcs	Valid Data to RX_CLK (setup)	100Mbps	5	-	-	ns
Tcdh	RX_CLK to Valid Data (hold)	100Mbps	5	-	-	ns

■ Package Description

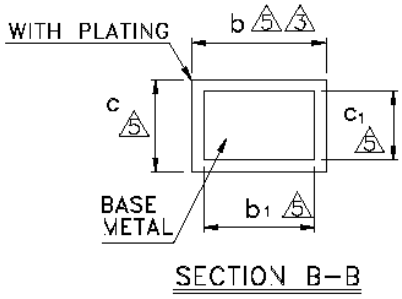
Figure 1: W3100A LQFP Package Specifications







Symbol	Dimension in mm			Dimension in inch		
	Min	Nom	Max	Min	Nom	Max
A	—	—	1.60	—	—	0.063
A1	0.05	—	0.15	0.002	—	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.17	0.22	0.27	0.012	0.015	0.018
b1	0.17	0.20	0.23	0.012	0.014	0.016
c	0.09	—	0.20	0.004	—	0.008
c1	0.09	—	0.16	0.004	—	0.006
D	12.00 BSC			0.472 BSC		
D1	10.00 BSC			0.394 BSC		
E	12.00 BSC			0.472 BSC		
E1	10.00 BSC			0.394 BSC		
e	0.50 BSC			0.020 BSC		
L	0.45	0.60	0.75	0.018	0.024	0.030
L1	1.00 REF			0.039 REF		
R1	0.08	—	—	0.003	—	—
R2	0.08	—	0.20	0.003	—	0.008
S	0.20	—	—	0.008	—	—
θ	0°	3.5°	7°	0°	3.5°	7°
θ1	0°	—	—	0°	—	—
θ2	12°TYP			12°TYP		
θ3	12°TYP			12°TYP		



■ Appendix A. Electrical Specifications

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Rating	Unit
V_{DD}	DC supply voltage	-0.3 to 3.8	V
V_{IN}	DC input voltage	-0.3 to 5.5(5V-tolerant)	V
I_{IN}	DC input current	±10	mA
T_{STG}	Storage temperature	-40 to 125	°C

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

RECOMMENDED OPERATING CONDITIONS

Parameter	Rating	Unit
Commercial temperature range	0 to 70	°C
Industrial temperature range	-10 to 70	°C

D.C. OPERATING CHARACTERISTICS $V_{DD} = 3.3 \pm 0.3V$, $V_{EXT} = 5 \pm 0.25V$

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{IH}	High level input voltage		2.0			V
V_{IL}	Low level input voltage				0.8	V
I_{IH}	High level input current	$V_{IN}=V_{DD}$	-10		10	μA
		$V_{IN}=V_{DD}$ (External Pull-down)	10	30	60	
I_{IL}	Low level input current	$V_{IN}=V_{SS}$	-10		10	μA
		$V_{IN}=V_{SS}$ (External Pull-up)	-60	-30	-10	
V_{OH}	High level output voltage	I_{OH}=-4mA	2.4			V
V_{OL}	Low level output voltage	I_{OL}=4mA			0.4	V
I_{OZ}	Tri-state output leakage current	$V_{OUT}=V_{SS}$ or V_{EXT}	-10		10	μA
I_{OS}	Output short circuit current	V_{DD}=3.6V, $V_O=V_{DD}$			55	mA
		V_{DD}=3.6V,$V_O=V_{SS}$	-55			
C_{IN}	Input capacitance[Note1]	Any input and bi-directional buffers			4	pF
C_{OUT}	Output capacitance[Note1]	Any output buffer			4	pF

*Notes: This value excludes package parasitics.

POWER DISSIPATION

Symbol	Condition	Power Consumption	Unit
P_{10BASE}	Minimum	5	mA
	Typical	9	
	Maximum	11	
$P_{100BASE}$	Minimum	10	
	Typical	30	
	Maximum	35	

■ Appendix B. Programming Guide

1. Notes when executing “Close command”

A. Recommendations before executing “Close command”

When you execute “close command” in W3100A, check whether “Tx buffer” is empty first. Only when it is empty, execute “close command”. If you execute “close command” with “Tx buffer” filled, there could be problems of flooding of FIN and ACK packet (Refer to drivers provided by WIZnet).

B. Recommendations after executing “Close command”

You can use both “interrupt method” and “polling method” to check whether right sockets are closed properly after executed “close command” at W3100A. In “interrupt method”, you can confirm normal socket close by checking “interrupt status register”, and in “polling method”, you can verify socket close by reading socket status register of currently handling channels periodically. One more thing to be cautious is that when you execute “close command” in W3100A, socket status becomes “closed (0x00)” for a certain time. But it will return to its normal status shortly. So just disregard this phenomena and proceed to your normal steps (Refer to drivers provided by WIZnet).

2. Notes when setting “MSS maximum value”

Maximum MSS value of W3100A is limited to 1460 in UDP or IP Raw mode. Therefore, do not set an MSS value greater than 1460 in UDP or IP Raw mode.

3. Notes when processing data in “IP Raw mode”

Don’t use the value as a data length. The data length value which was received in IP Raw mode should be recalculated by adding 4 bytes for use.

4. Notes when handling “Shadow register”

As internal pointer register is 4 bytes and data bus for CPU is 1 byte in W3100A, there is Shadow register in order not to update pointer register by operations of W3100A while CPU is reading pointer register. If interrupts are generated during reading Shadow register, Shadow register value can be changed. To prevent this unexpected function, we recommend you to disable external Interrupt while reading Shadow register (Refer to drivers provided by WIZnet).