W5200 Indirect Bus Interface Guide

Version 1.0.0

This guide is only for W5200.

W7200 does not support the indirect bus interface.

W7200 support High Speed SPI bus interface only.

Please do not confuse.

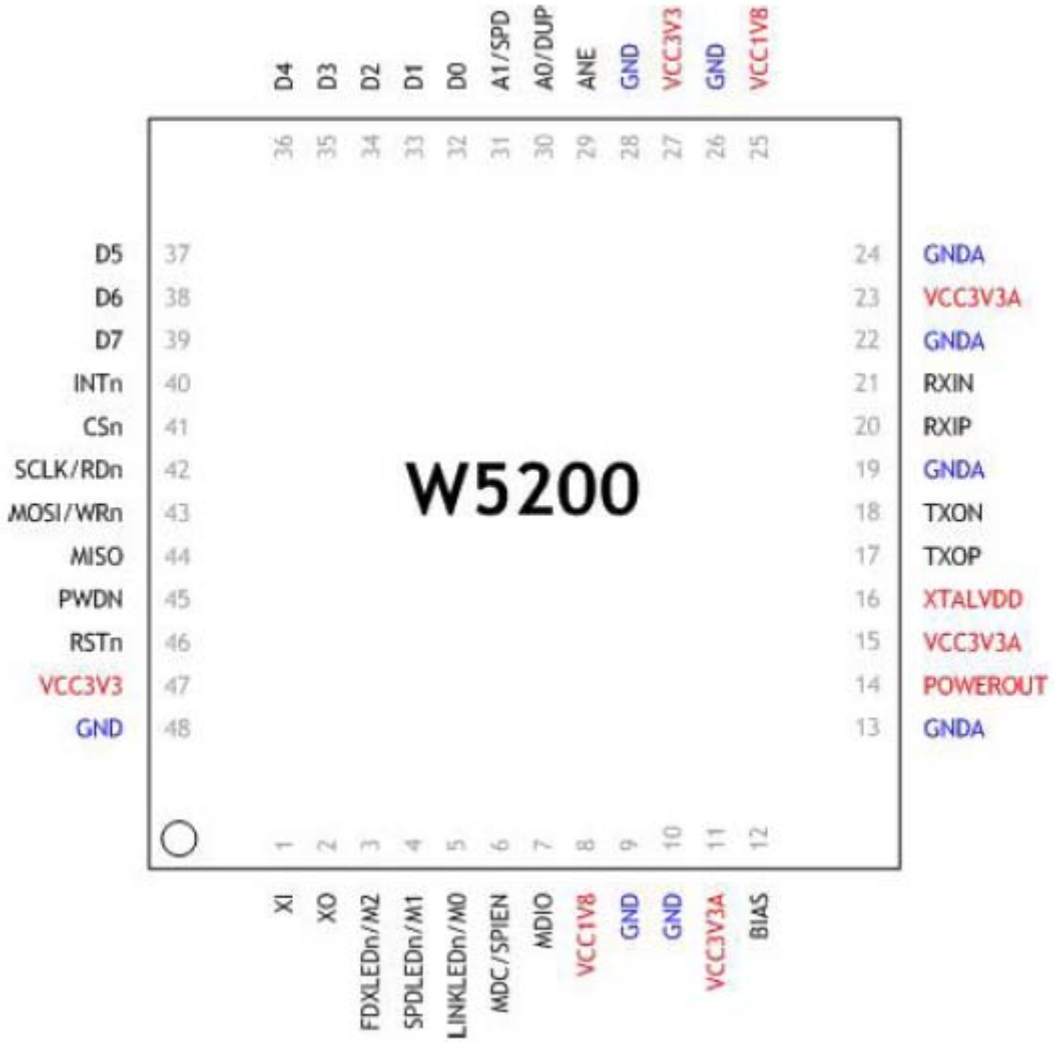
For more information, visit our website at http://www.wiznet.co.kr

Document History Information

Version	Date	Descriptions
1.0.0	2012-11-29	First Release

Table of Contents

1. Pin Assignment



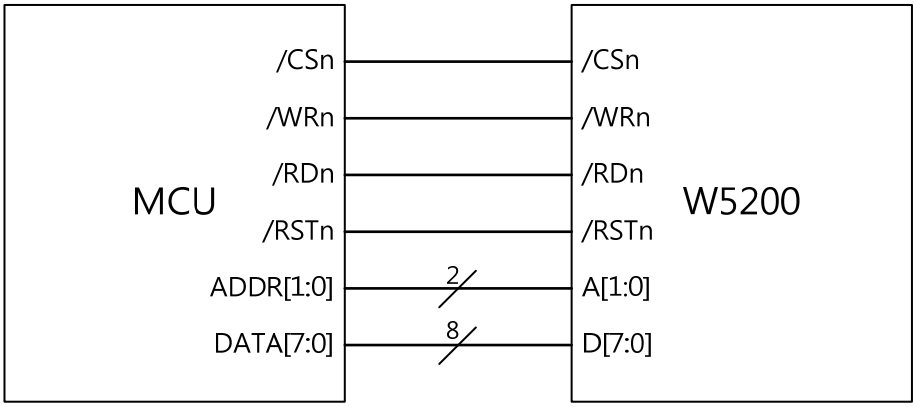
1.1 The Pins which Related with Indirect Bus Interface Mode

Symbol	Type	Pin No	Description
A0/DUP	I	30	**ADDRESS[0]** This pin is used to select a register or memory when using indirect interface. **Full Duplex Mode Enable** This pin selects Enable/Disable of Full Duplex Mode. Low = Half Duplex Mode Enable High = Full Duplex Mode Enable *This function activate only when reset period.*
A1/SPD	I	31	**ADDRESS[1]** This pin is used to select a register or memory when using indirect interface. **Speed Mode** This pin selects 100M/10M Speed Mode. Low = 10M Speed Mode High = 100M Speed Mode *This function activate only when reset period.*
D7-0	I/O	32, 33, 34, 35, 36, 37, 38, 39	**DATA** These pins are used to read and write register or memory data.
CSn	I	41	**CHIP SELECT (Active LOW)** Chip Select is for MCU to access to internal registers or memory when using indirect interface. **SPI SLAVE SELECT (Active LOW)** This pin is used to SPI Slave Select signal Pin when using SPI interface.
SCLK/RDn	I	42	**SPI CLOCK** This pin is used to SPI Clock signal Pin when using SPI interface. **READ ENABLE (Active LOW)** Strobe from MCU to read an internal register/memory selected by A[1:0] when using indirect interface.
MOSI/WRn	I	43	**SPI MASTER OUT SLAVE IN** This pin is used to SPI MOSI signal pin when using SPI

			interface. **WRITE ENABLE (Active LOW)** Strobe from MCU to write an internal register/memory selected by A[1:0] when using indirect interface. Data is latched into the W5200 on the rising edge of this input.
MDC/SPIEN	I	6	**Management Data Clock** This pin is used to MDC signal pin for access internal PHY register or debugs internal PHY. **SPI ENABLE (Active HIGH)** This pin selects Enable/disable of the SPI Mode. Low = SPI Mode Disable High = SPI Mode Enable If you don't use SPI mode, in other words, if you want to use indirect mode, then you tied this signal to '0'. This function activate only when reset period.

2. Indirect Bus Interface mode

Indirect Bus Interface mode uses 2bit address line and 8bit data line, /CSn, /RDn, /WRn.

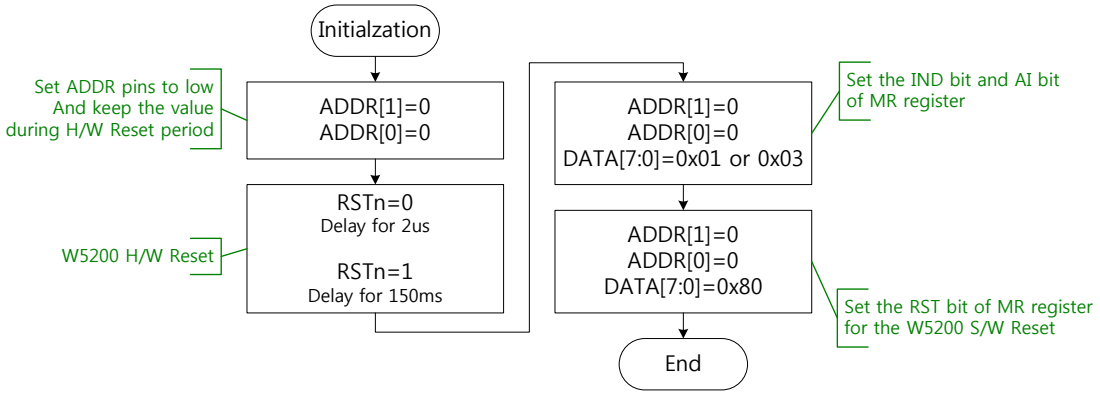


Indirect bus Interface mode related register is as below.

Value	Symbol	Description
0x00	MR	It performs the selection of Indirect bus Interface mode, address automatic increase. Refer to the Description of MR Register for more detail.
0x01 0x02	IDM_AR0 IDM_AR1	Indirect bus Interface mode address Register Big-endian use only · In case of Big-endian ordering 0x01 0x02 IDM_AR0 : MSB | IDM_AR1 : LSB Ex) In case of reading S0_CR(0x0401), 0x01(IDM_AR0) 0x02(IDM_AR1) 0x04 | 0x01
0x03	IDM_DR	Indirect bus Interface mode data Register

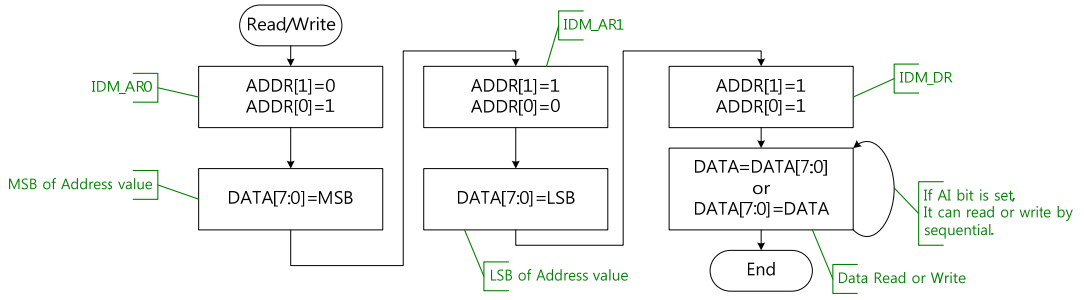
In order to initialize of W5200 Indirect Bus Interface mode,

1. Write the 0x00 on address of 0x0000 as below and keep the value during H/W Reset period.
   A. Because, ADDR0 and ADDR1 pins are shared with SPD and DUP pins.
   ((volatile uint8)(0x0000)) = 0x00;
2. Doing the H/W Reset
   A. Low assert RSTn pin for 2us (minimum)
   B. After High assert RSTn pin, delay for 150ms.
3. Write the 0x01 on MR. (IND bit will be set)
   A. If you need to using AI, You can write 0x03 on MR. (AI and IND bits will be set)
4. Write the 0x80 on MR after set the AI and IND bits on MR. (W5200 will be S/W Reset)



In order to read or write the internal register or internal TX/RX Memory,

1. Write the address to read or write on IDM_AR0, 1.
2. Read or Write IDM_DR.



In order to read or write the data on the sequential address, set AI bit of MR(Mode Register). With this, user performs above 1 only one time. Whenever read or write IDM_DR, IDM_AR, the value is automatically increased by 1. Therefore, the value can be processed on the sequential address just by continuous reading or writing of IDM_DR.

MR (Mode Register) [R/W] [0x0000] [0x00]

This register is used for S/W reset, ping block mode, PPPoE mode and indirect bus interface mode.

7	6	5	4	3	2	1	0
RST			PB	PPPoE		AI	IND

Bit	Symbol	Description
7	RST	**S/W Reset** If this bit is '1', internal register will be initialized. It will be automatically cleared after reset.
6	Reserved	Reserved
5	Reserved	Reserved
4	PB	**Ping Block Mode** 0 : Disable Ping block 1 : Enable Ping block If the bit is set as '1', there is no response to the ping request.
3	PPPoE	**PPPoE Mode** 0 : Disable PPPoE mode 1 : Enable PPPoE mode If you use ADSL without router or etc, you should set the bit as '1' to connect to ADSL Server. For more detail, refer to the application note, *"How to connect ADSL"*.
2	Not Used	Not Used
1	AI	**Address Auto-Increment in Indirect Bus Interface** 0 : Disable auto-increment 1 : Enable auto-increment At the Indirect Bus Interface mode, if this bit is set as '1', the address will be automatically increased by 1 whenever read and write are performed.
0	IND	**Indirect Bus Interface mode** 0 : Disable Indirect bus Interface mode 1 : Enable Indirect bus Interface mode If this bit is set as '1', Indirect Bus Interface mode is set.

3. Driver Porting Guide

Sample source code is implemented based on ATMega128. W5200-EVB does not compatible to sample source code and does not support indirect interface mode.

- **Base address of W5200**

  Define W5200's base address as '__DEF_IINCHIP_MAP_BASE__'.

```
#define __DEF_IINCHIP_MAP_BASE__ 0x8000
```

If its base address is more than 0xFFFF, following parts should be changed.

<Refer to types.c>

```
static u32 SBUFBASEADDRESS[MAX_SOCK_NUM];
static u32 RBUFBASEADDRESS[MAX_SOCK_NUM];
u32 getIINCHIP_RxBASE(u8 s)
u32 getIINCHIP_TxBASE(u8 s)
u8 IINCHIP_WRITE(u32 addr, u8 data)
u8 IINCHIP_READ(u32 addr)
```

- **W5200 interface**

The value of '__DEF_IINCHIP_BUS__' should be changed based on the way to connect W5200 to MCU. You should choose a mode between indirect mode and SPI mode based on implemented H/W interface.

<Refer to types.h>

```
#define __DEF_IINCHIP_INDIRECT_MODE__   1
#define __DEF_IINCHIP_SPI_MODE__        2
#define __DEF_IINCHIP_BUS__             __DEF_IINCHIP_INDIRECT_MODE__
```

- **W5200 initialization**

1) Set ADDR[0-1] to 0x0000. (Write the 0x00 on address of 0x0000)

```
*((volatile uint8*)(0x0000)) = 0x00;
```

2) Doing H/W Reset the W5200

3) Set W5200 to Indirect Interface Mode and Address Auto Increase.

```
MR [0x0000]     0x03;
```

4) Initialize W5200 with S/W reset.

```
MR [0x0000]     0x80;
```

5) Set up the network configuration.

Basic network configurations to be configured for networking are as below.

- Gateway Address Register
- Source Hardware Address Register
- Subnet Mask Register
- Source IP Address Register

ex) Gateway address set up 192.168.0.1

```
GAR [0x0001 ~ 0x0004]     [0xC0, 0xA8, 0x00, 0x01];
```

Likewise, set up Subnet Mask Register, Source IP Address Register, and Source Hardware Address Register.

6) Set up RX Memory Size Register and TX Memory Size Register. You can allocate the amount of memory to be used for each channel.

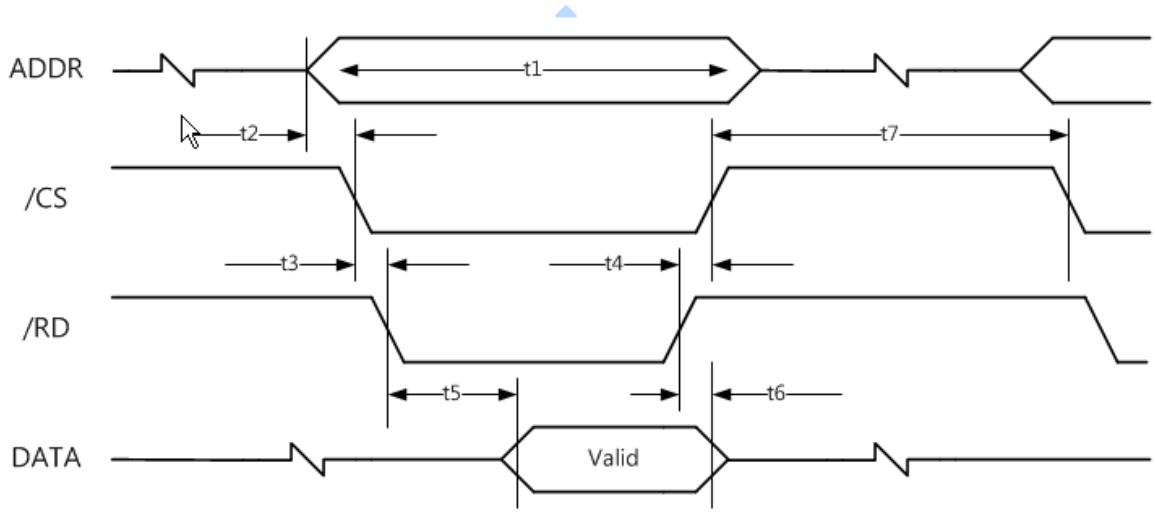
7) If you finish 6) step, you can get ICMP reply message from W5200. It means, if you execute “ping” in your PC with the destination IP address of W5200, you can get reply.

Pseudo Code for Initialization

```
// Set ADDR[0-1] to 0x0000
*((volatile uint8*)(0x0000)) = 0x00;

// H/W Reset
RSTn = 0;
Delay_us(2);
RSTn = 1;
Delay_ms(150);

// Enables Indirect Mode and Address Auto Increase
// Below is implemented in iinchip_init() function (Refer to w5200.c file)
MR = 0x03;
MR = 0x80;

// Set up the network configuration
// Below is implemented in main() function (Refer to main.c file)
SHAR[0-3] = {0x00, 0x08, 0xDC, 0x11, 0x22, 0x33};
GAR[0-3] = {192, 168, 0, 1};
SUBR[0-3] = {255, 255, 255, 0};
SIPR[0-3] = {192, 168, 0, 100};

// Set up RX/TX Memory Register
// Below is implemented in sysinit() function (Refer to w5200.c file)
S0_RXMEM_SIZE = (uint8 *) 2; // Assign 2K rx memory
S0_TXMEM_SIZE = (uint8 *) 2; // Assign 2K tx memory
S1_RXMEM_SIZE = (uint8 *) 2;
S1_TXMEM_SIZE = (uint8 *) 2;
…
S7_RXMEM_SIZE = (uint8 *) 2;
S7_TXMEM_SIZE = (uint8 *) 2;
```

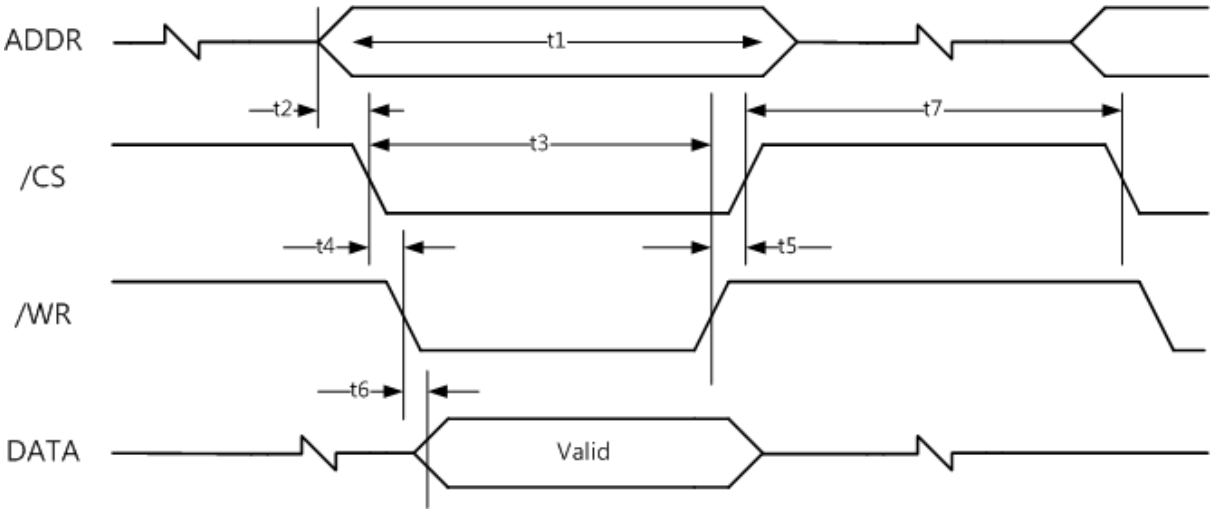
4. Signal Timing Diagram

Register/Memory READ Timing



Symbol	Description	Min	Max
t1	Read Cycle Time	80 ns	-
t2	Valid Address to /CS low time	8 ns	-
t3	/CS low to /RD low time	0 ns	
t4	/RD high to /CS high time	0 ns	-
t5	/RD low to Valid data output time	48 ns	-
t6	/RD high to Data High-Z Output time	-	1 ns
t7	/CS high to next /CS low time	32 ns	-

Register/Memory WRITE Timing



Symbol	Description	Min	Max
t1	Write Cycle Time	70 ns	-
t2	Valid Address to /CS low time	7 ns	-
t3	/CS low to /WR high time	70 ns	
t4	/CS low to /WR low time	0 ns	-
t5	/WR high to /CS high time	0 ns	-
t6	/WR low to Valid data time	-	7 ns
t7	/CS high to next /CS low time	32 ns	-