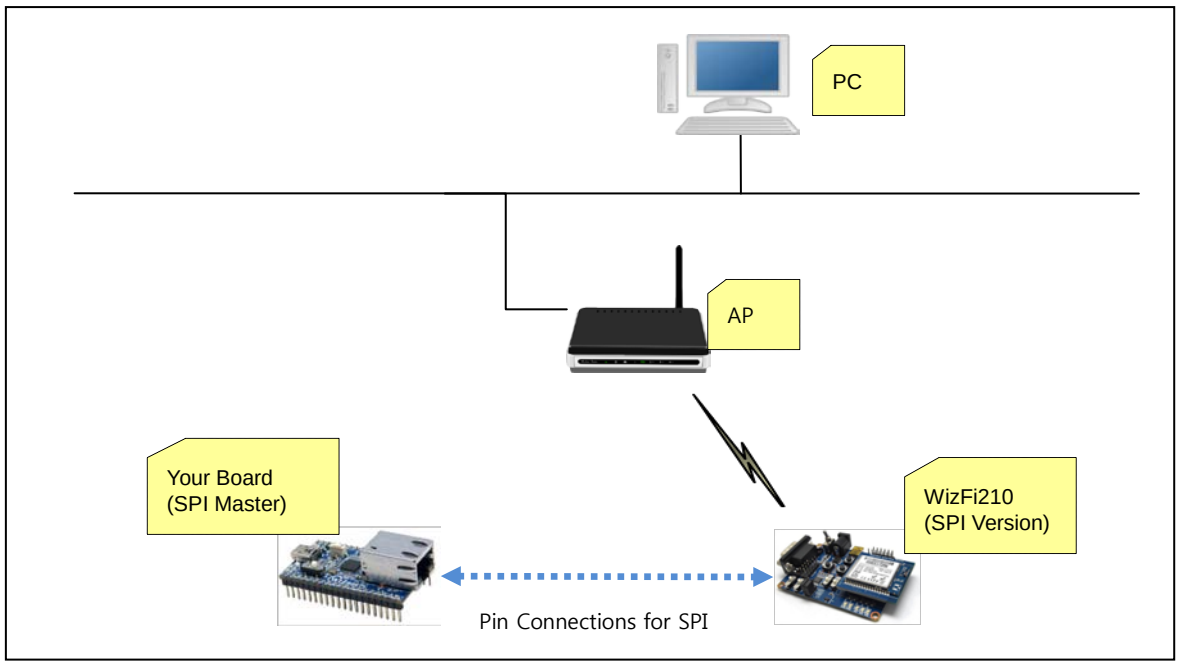
SPI-to-WiFi Guide

(WizFi 210 Application Notes)

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

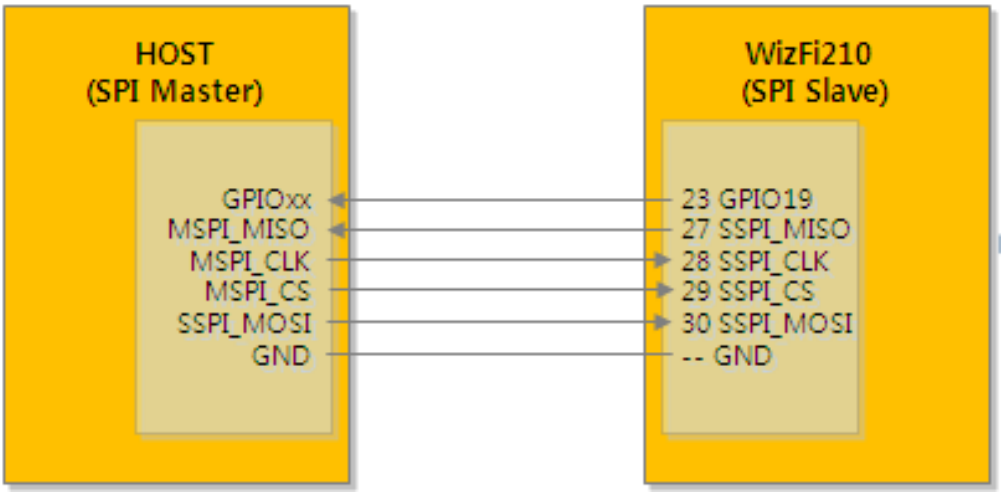
1. Diagram for example of SPI Interface



In the case of SPI interface, WizFi210 acts as slave and will communicate to master SPI controller. By default, SPI interface supports Motorola protocol with clock polarity 0 and clock phase 0.

Since SPI data transfer works in full duplex mode, it's required to make use of special octet to indicate idle data. Similarly if host MCU is sending data at higher rate flow control mechanism is required. In order differentiate these special control codes (such as idle pattern, flow control codes and other control octets) from user data, byte stuffing mechanism is incorporated.

2. Pin connections for SPI



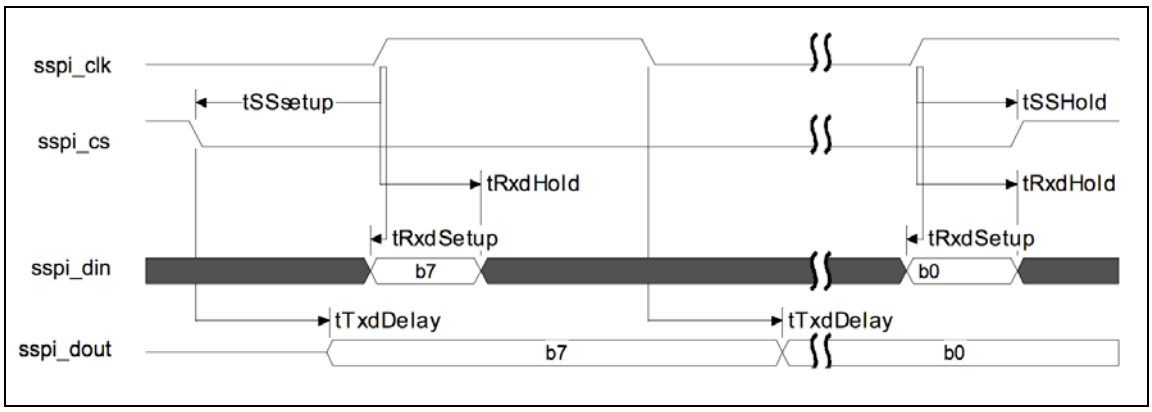
Host App (SPI Master)	WizFi210 (SPI Slave)	Remarks
MSPI_MISO	SSPI_MISO (27)	SPI Master In/Slave Out
MSPI_CLK	SSPI_CLK (28)	SPI Clock
MSPI_CS	SSPI_CS (29)	SPI Chip Select
MSPI_MOSI	SSPI_MOSI (30)	SPI Master Out/Slave In
Allocate your GPIO	GPIO#19 (23)	Host wake-up signal
GND	GND	Common ground

3. SPI interface details

In case of SPI interface additional task is required to handle SPI data transfer. SPI data transfer

- Only Motorola mode is supported
- Only 8 bit SPI data word size is supported
- By default SPI Mode#0 is selected (CPOL =0 and CPH=0)

Motorola SPI Format with CPL=0, CPH=0



Note: In case of continuous back-to-back transmissions, the Chip Select (CS) signal must be pulsed HIGH between each byte (8 bit) transfer.

4. Host Wake-Up Signal Handling

Host wake-up signal is ACTIVE HIGH signal. Host controller must give the SPI clock, as long as host wake-up signal is HIGH.

Whenever WizFi210 wants to transfer the data it asserts (HIGH) host wake-up signal. Once all the data transferred from WizFi210 it again de-asserts (LOW) the signal.

Host controller will detect the host wake-up signal transition (LOW to HIGH) as edge triggered interrupt and process the incoming data.

5. SPI Transmit data handling

The SPI data transfer layer makes use of an octet (or byte) stuffing procedure. The Control Escape octet is defined as binary ***11111011*** (hexadecimal ***0xFB***), most significant bit first.

Each special control pattern is replaced by a two octet sequence consisting of the Control Escape octet followed by the original octet exclusive-or (***XOR***) with hexadecimal ***0x20***.

Receiving implementations must correctly process all Control Escape sequences.

Escaped data is transmitted on the link as follows:

Pattern	*Encoded as*	*Description*
0xFD	0xFB 0xDD	*SPI_XON*
0xFA	0xFB 0xDA	*SPI_XOFF*
0xFB	0xFB 0xDB	Control ESCAPE
0xF5	0xFB 0xD5	*SPI _IDLE*
0xF3	0xFB 0xD3	SPI link ready indication

One dedicated GPIO signal known as host wake-up is available for data ready indication from Slave WizFi210 to Master Host controller. Master host controller must provide clock as long as host wake-up signal is active. Host controller can make use of GPIO interrupt (edge triggered low-to-high transition) to receive the data from WizFi210.

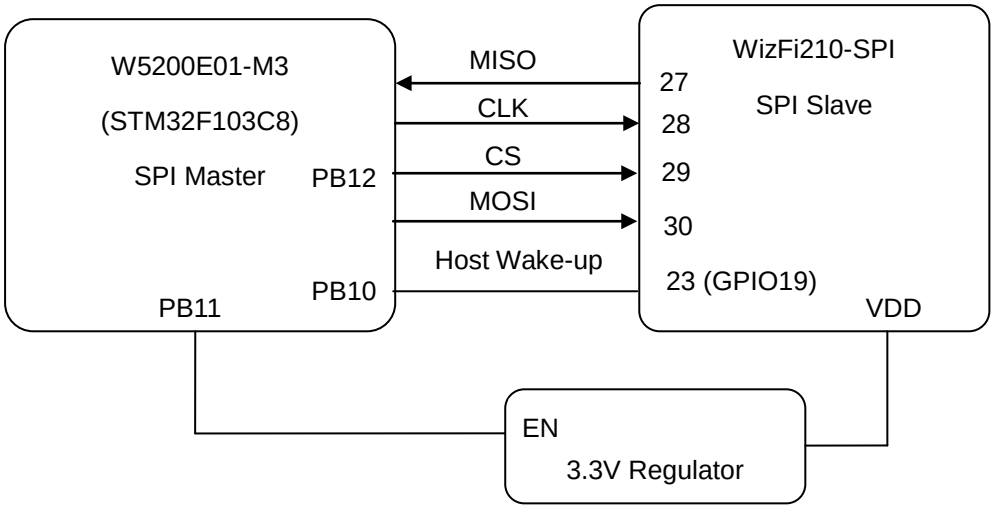
Since SPI data transfer works in full duplex mode, special fill character (***SPI _IDLE***) will be transmitted during idle period (if there is no more data to transmit). These idle fill pattern shall be dropped at receiving end.

6. SPI Receive data handling

Since byte stuffing is used, each Control Escape octet must be removed, and the next immediate octet is exclusive-or(***XOR***) with hexadecimal ***0x20***.

If receive buffer is reached upper water mark, then ***SPI_XOFF*** character will be sent out informing the host to stop transmitting actual data. After receiving ***SPI_XOFF*** character host must stop transmitting actual data. Once the application starts processing received data and enough space available for further reception (reached lower water mark), ***SPI_XON*** will be transmitted. Once host receives ***SPI_XON***, then it can resume the valid data transmission. Special control byte ***SPI _IDLE*** will be dropped at receiver.

7. Implementation Example



- SPI Master : W5200E01-M3 (STM32F103C8) See the below web pages.
  http://www.wiznet.co.kr/Sub_Modules/en/product/Product_Detail.asp?cate1=5&cate2=7&cate3=56&pid=1152
  http://www.st.com/internet/mcu/product/164476.jsp
- SPI Slave : WizFi210, Firmware Version 1.1.0.2(SPI)
- PB10 : GPIO of SPI Master, Host Wake-up signal, Active High
- PB11 : To reset WizFi210, control the Enable signal of regulator(Optional)
- SPI Mode 0 (CPOL = 0 and CPHASE = 0)
- Only 8 bit SPI data word size is supported
- SPI Clock Rate is 200 KHZ(See the SPI Master's source files)
- WizFi210 send the <IDLE code(0xF5)> periodically, when it is in idle state.
- SPI Master needs to send the <IDLE Code> to the WizFi210 after reboot to synchronize the SPI interface.
- When Host Wake-up signal is high, SPI Master needs to receive the data which WizFi210 is sending.
- SPI Master can check whether WizFi210 is the data mode or command mode by sending the dummy "AT" command.

- SPI Master's sample source is implemented in following environment.

  WizFi210 IP : 192.168.88.123 (Static IP)

  TCP Server mode : Port 5000

  SSID of AP : WIZ_RED

  WPA Passphrase : wiznet0123456

```
H/W reset or checking data mode or command mode
Send IDLE character for Synchronization
Check Host wakeup signal and read data
AT+WD
AT+WAUTO=0,WIZ_RED
AT+WAUTH=0
AT+WWPA= wiznet0123456
AT+NDHCP=0
AT+NSET=192.168.88.123,255.255.255.0,192.168.88.1
AT+NAUTO=1,1,,5000
AT+XDUM=1
ATA
```

- For the details, refer to the SPI Master's source files in WIZnet's homepage