Appendix II – Programming guide for SPI to I80 interface

(IT8951 DX only)

In IT8951 CX/DX version, we added new feature that Host can send I80 command through SPI interface, Programmer should modify the following basic functions and Implement SPI driver which depends on your host

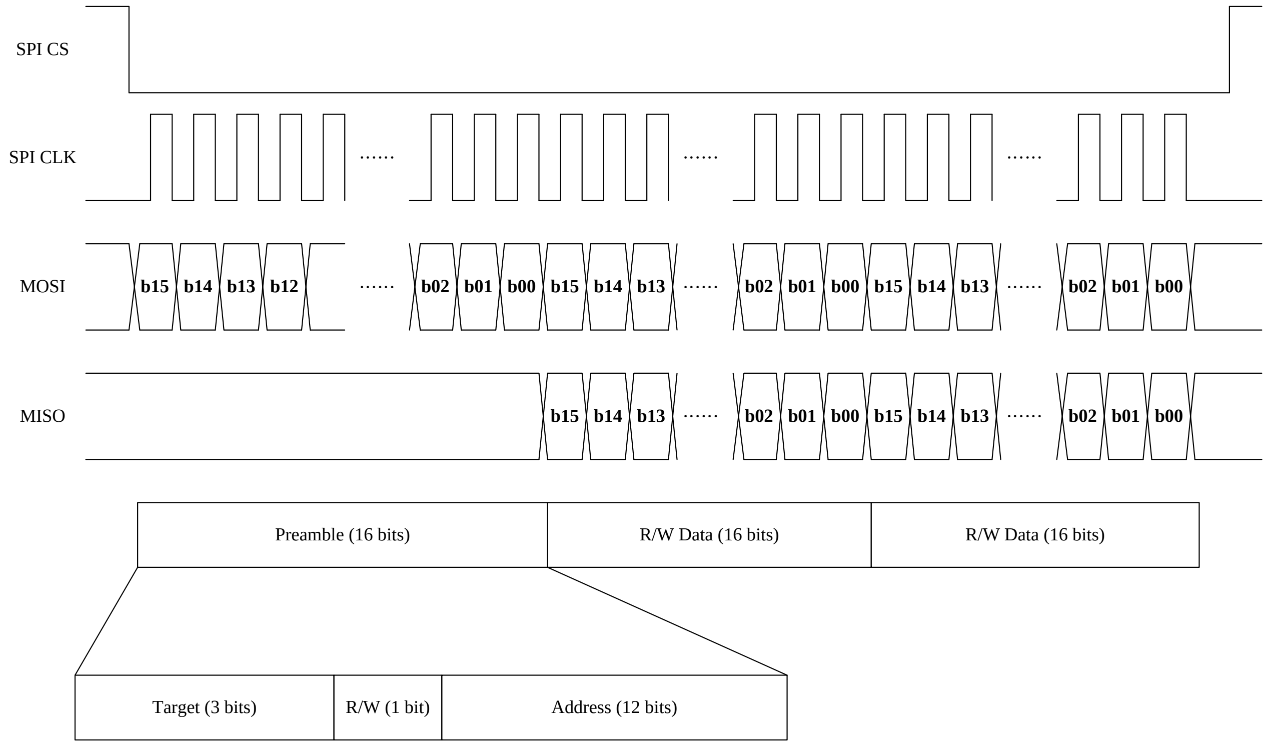
1. LCDWriteCmdCode()
2. LCDWriteData()
3. LCDWriteNData()
4. LCDReadData()
5. LCDReadNData()

Please kindly note that all of the commands/data sent via SPI interface will be converted to I80 format when IT8951 received. Therefore host programmer still have to regard some properties of I80 protocol as following:

1. Wait I80 Bus ready(HRDY) before sending command/data

   We recommend that Host may need 1 GPI pin connected to HRDY pin of IT8951.
2. 16 bits Bus width for each Write/Read transfer

SPI Format introduction:



- 16 bits base (I80 Spec)
- Send Preamble first.
- R/W bit in preamble will determine the data direction (Read or Write).
- Only the **first** 16-bits data will be preamble in each SPI cycle (SPI_CS low Period).
- Big Endian format for each Word(2 bytes) transfer
  - SPI Send Byte[0] = Word[15:8]
  - SPI Send Byte[1] = Word[7:0]

Specification of IT8951 SPI:

Suggested SPI Clock rate : 12MHZ (Max: 24MHZ)

SPI Mode : Mode 0 only (CPOL = 0, CPHA = 0)

Clock Low in idle.

Preparing Write data after falling edge, and Capture data after rising edge

Preamble for SPI to I80

Base on the SPI format, set different preamble value will convert to different I80 (M68) cycle. There are three type of cycle in I80 (M68), write command code, write data and read data. If we want to change the type of I80 (M68) cycle, you should finish this SPI cycle first, and then start another SPI cycle.

Preamble Value	I80 (M68) Cycle Type
0x6000	Write Command Code
0x0000	Write Data
0x1000	Read Data

Example

e.g.1 - Write IT8951 Register Address[0x1100] = 0x0506

1 **Send Command 0x0011 (WRITE_REG)**

1.1 Send preamble 0x6000 (command type)

1.2 Send Command 0x0011

CS to L -> MOSI Data {0x60,0x00, 0x00,0x11} -> CS to H

2 **Send Register Address 0x1100**

2.1 Send preamble 0x0000 (Data type)

2.2 Send WData 0x1100

CS to L -> MOSI Data {0x00,0x00, 0x11,0x00} -> CS to H

3 Send Write Data 0x0506

3.1 Send preamble 0x0000 (Write Data type)

3.2 Send WData 0x0506

CS to L -> MOSI Data {0x00,0x00, 0x05,0x06} -> CS to H

e.g.2 - Read IT8951 Register Address[0x1100]

1 Send Command 0x0010 (READ_REG)

1.1 Send preamble 0x6000 (command type)

1.2 Send Command 0x0010

CS to L -> MOSI Data {0x60,0x00, 0x00,0x10} -> CS to H

2 Send Register Address 0x1100

2.1 Send preamble 0x0000 (Write Data type)

2.2 Send WData 0x1100

CS to L -> MOSI Data {0x00,0x00, 0x11,0x00} -> CS to H

3 Read Data (Suppose <= Read Data = 0x0506)

3.1 Send preamble 0x1000 (Read Data type)

3.2 Read Dummy 0xXXXX

3.3 Read RData 0x0506

CS to L -> MOSI Data {0x10,0x00} -> MISO Data{0xXX,0xXX, 0x05,0x06} -> CS to H

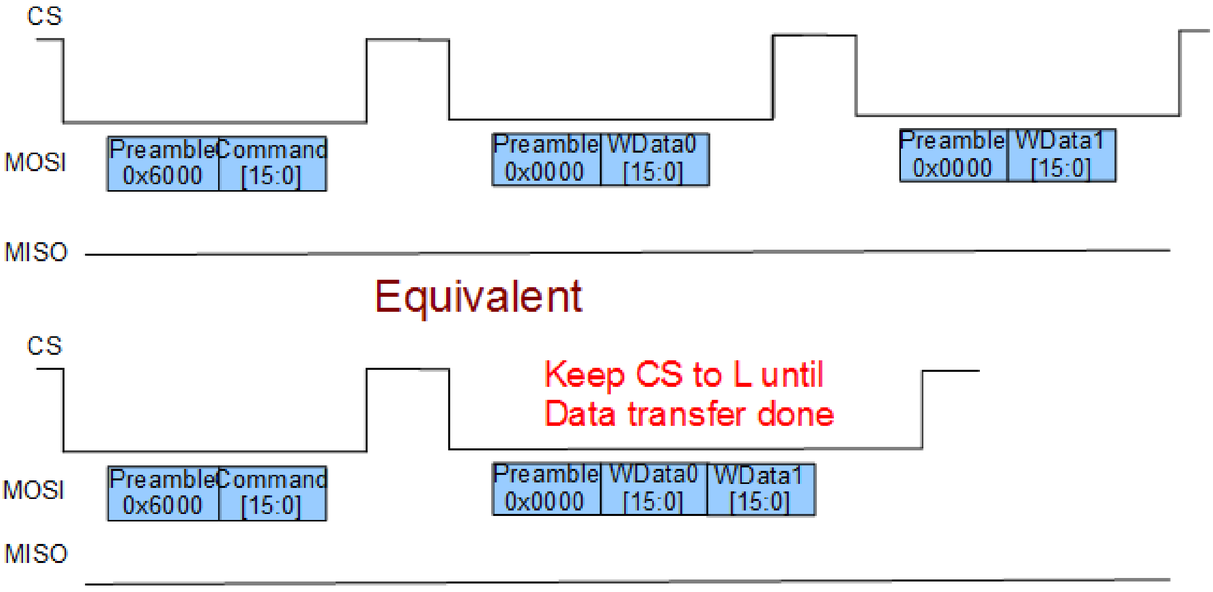
PS. In the read behavior, it needs a dummy read for each CS. The first 16-bits data are not valid data which should be ignored.

Burst Data transfer

In IT8951, it is available for Burst read/write data transfer between Host and IT8951 through interface

For example:

Write 2 words data

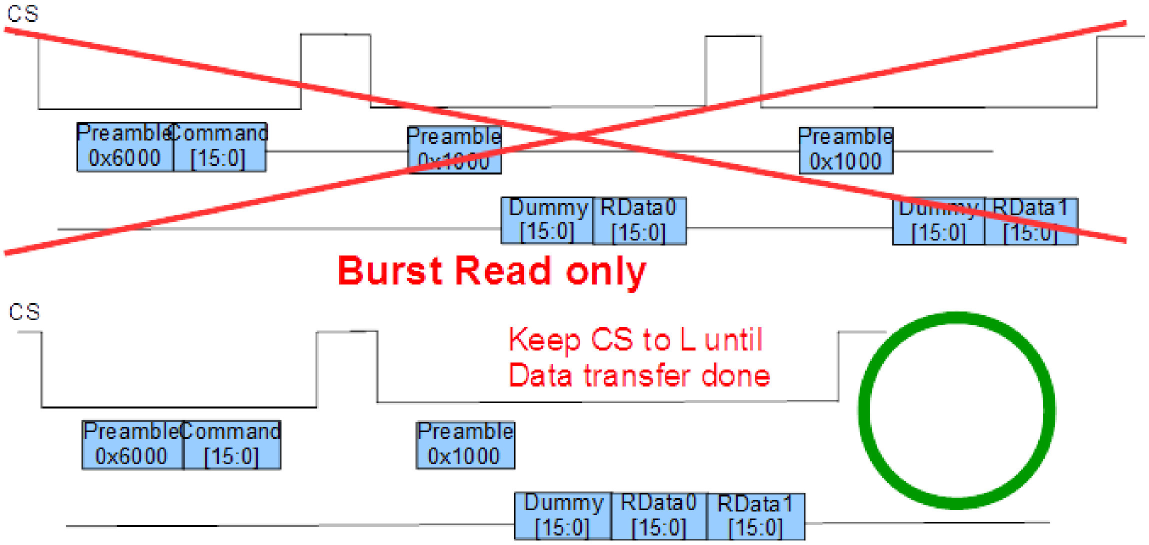


Both of above transfer type are available and equivalent in IT8951 SPI format.

Read 2 words data

However, the multiple access transfer by Single read data is not valid under IT8951 SPI specification.

Therefore, Host SPI controller must use burst read transfer only if read data >= 2;



SPI to I80 programming flow and Sample Code

If your Host MCU is little Endian type, please define the Macro as following:

```
#define __HOST_LITTLE_ENDIAN__  //Big or Little Endian for your Host platform

#ifdef __HOST_LITTLE_ENDIAN__
    //Little Endian => its needs to convert for SPI to I80
    #define  MY_WORD_SWAP(x)  ( ((x & 0xff00)>>8) | ((x & 0x00ff)<<8) )
#else
    //Big Endian => No need to convert
    #define  MY_WORD_SWAP(x)  (x)
#endif
```

Regarding to pseudo code about SPI Read/Write API Functions

They are only for reference, for more detailed description, please refer the SPI controller of your platform.

- **SPIWrite(TByte* pWBuf, TDWord Size, TByte CS)**
  - **pWBuf** – pointer of Write buffer
  - **Size** – Size of SPI Data transfer (unit: Byte)
  - **CS** –
    - 0 – CS L, it means the CS will still keep low after current data transfer
    - 1 – CS H, it means the CS will be High after current data transfer
- **SPIRead(TByte* pRBuf, TDWord Size, TByte CS)**
  - **pRBuf** – pointer of Read buffer
  - **Size** – Size of SPI Data transfer (unit: Byte)
  - **CS** –
    - 0 – CS L, it means the CS will be low (or keep low)before transferring
    - 1 – CS H, it means the CS will be High after transferring
- **In the IT8951,**

1 LCDWriteCmdCode(usCmd)

1.1 Send Preamble 0x6000 for Write command code

1.1.1 Wait for I80 Bus Ready? (IT8951 HRDY)

1.1.2 Set CS to Low

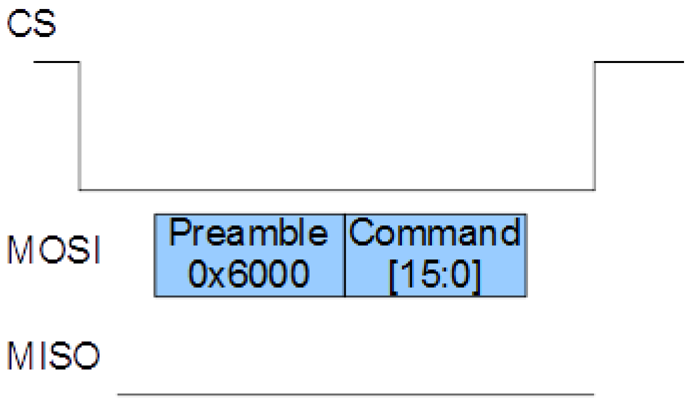
1.1.3 Send SPI Data[2] = { 0x60, 0x00 };

1.2 Send I80 Command code

1.2.1 Wait for I80 Bus Ready?

1.2.2 Send SPI Data[2] = {usCmd[15:8], usCmd[7:0]};

1.2.3 CS to High



```
void LCDWriteCmdCode (TWord wCmd)
{
    WORD wPreamble = 0;

    //Set Preamble for Write Command
    wPreamble = 0x6000;

    //Send Preamble
    wPreamble    = MY_WORD_SWAP(wPreamble);
    LCDWaitForReady();
    SPIWrite((TByte*)&wPreamble, 1*2, CS_L);

    //Send Command
    wCmd         = MY_WORD_SWAP(wCmd);
    LCDWaitForReady();
    SPIWrite((TByte*)&wCmd, 1*2, CS_H);
}
```

2 LCDWriteData(usData)

2.1 Send Preamble 0x0000 for Writing command code

2.1.1 Wait for I80 Bus Ready? (IT8951 HRDY)

2.1.2 CS to Low

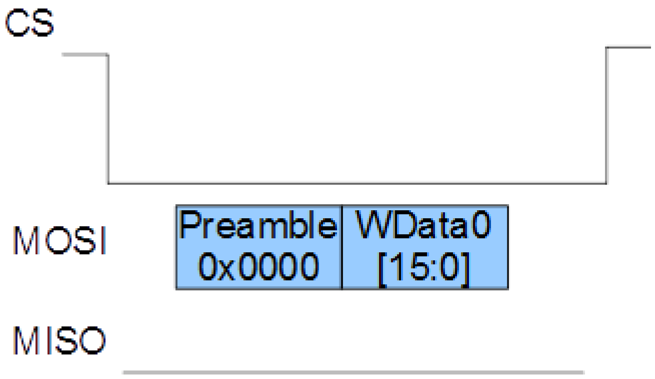
2.1.3 Send SPI Data[2] = { 0x00, 0x00 };

2.2 Send 1 Word Data

2.2.1 Wait for I80 Bus Ready?

2.2.2 Send SPI Data[2] = {usData[15:8], usData[7:0]};

2.2.3 CS to High



```
void LCDWriteData(TWord usData)
{
    WORD wPreamble  = 0;

    //set type
    wPreamble = 0x0000;

    //Send Preamble
    wPreamble = MY_WORD_SWAP(wPreamble);
    LCDWaitForReady();
    SPIWrite((TByte*)&wPreamble, 1*2, CS_L);
    //Send Data
    usData = MY_WORD_SWAP(usData);
    LCDWaitForReady();
    SPIWrite((TByte*)&usData, 1*2, CS_H);
}
```

3 LCDWriteNData(TWord* pwBuf, TDWord ulDataCnt)

This function is suitable for burst write data only. We don’t recommend

3.1 We defined Max burst Transfer Len is 2048 bytes(1024 Words) for each time

3.1.1 You have to divide into Size/1024 times to transfer if the transfer size is over 1024 words

3.1.2 or you have to polling ready for each word

3.1.3 the size of IT8951 FIFO is 2k bytes, therefore we can continued read or send data for 2kbytes(1k word)

3.2 Send Preamble 0x0000 for Write command code

In this case, the preamble 0x0000 can send once only if sending burst data.

3.2.1 Wait for I80 Bus Ready? (IT8951 HRDY)

3.2.2 Set CS to Low

3.2.3 Send SPI Data[2] = { 0x00, 0x00 };

3.3 Send N Words Data

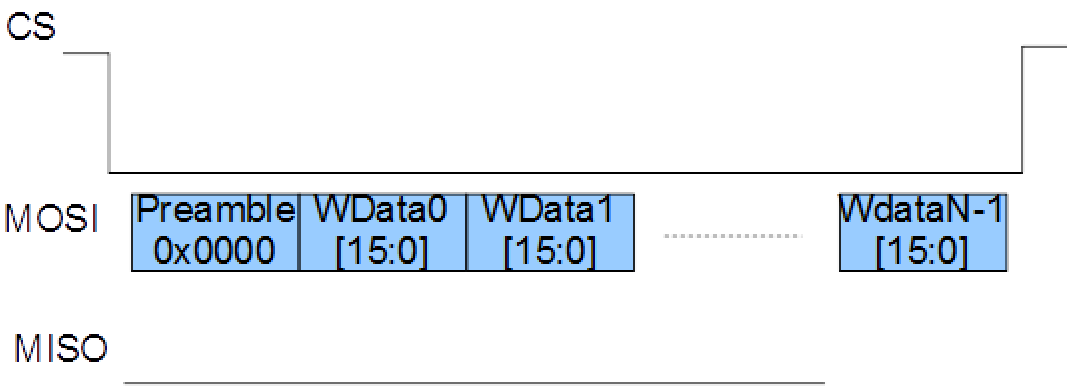
3.3.1 Wait for I80 Bus Ready?

3.3.2 Needs to Convert Data Endian? If not, go to 3.7

3.3.3 Send SPI Data for N*2 bytes = {usData[15:8], usData[7:0],…………

usDataN-1[15:8], usDataN-1[7:0]] };

3.3.4 Finished -> Set CS to High



```
void LCDWriteNData(TWord* pwBuf, TDWord ulSizeWordCnt)
{
    WORD wPreamble  = 0;
    TDWord i;
    //set type
    wPreamble = 0x0000;
    //Send Preamble
    wPreamble = MY_WORD_SWAP(wPreamble);
    LCDWaitForReady();
    SPIWrite((TByte*)&wPreamble, 1*2, CS_L);

    #ifdef __HOST_LITTLE_ENDIAN__
    //Convert Little to Big Endian for each Word
    for(i=0;i<ulSizeWordCnt;i++)
    {
        pwBuf[i] = MY_WORD_SWAP(pwBuf[i]);
    }
    #endif

    //Send Data
    LCDWaitForReady();
    SPIWrite((TByte*)pwBuf, ulSizeWordCnt*2, CS_H);
}
```

4 Word LCDReadData()

4.1 Send Preamble 0x1000 for Write command code

4.1.1 Wait for I80 Bus Ready? (IT8951 HRDY)

4.1.2 Set CS to Low

4.1.3 Send SPI Data[2] = { 0x10, 0x00 };

4.2 Read 1 Dummy Word first

4.2.1 Wait for I80 Bus Ready?

4.2.2 Get SPI Read Data[2] = {usDummy[15:8], usDummy[7:0]};

4.2.3 Read 1 Dummy Word (2bytes)

4.3 Get 1 Word Read Data

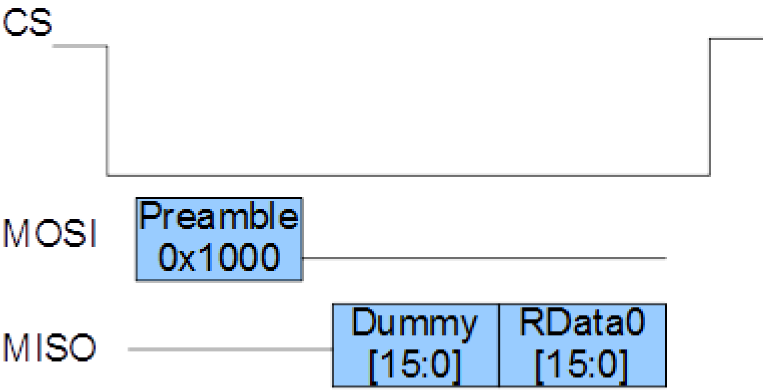
4.3.1 Wait for I80 Bus Ready?

4.3.2 Get SPI Read Data[2] = {usData[15:8], usData[7:0]};

4.3.3 CS to High

4.4 Return usData

4.4.1 Convert to Little Endian? (it depends on your host Endian type)



```
TWord LCDReadData()
{
    TWord wPreamble = 0;
   TWord wRData;
    TWord wDummy;

    //set type and direction
    wPreamble = 0x1000;

    //Send Preamble before reading data
    wPreamble = MY_WORD_SWAP(wPreamble);
    LCDWaitForReady();
    SPIWrite((TByte*)&wPreamble, 1*2, CS_L);

    //Read Dummy (under IT8951 SPI to I80 spec)
    LCDWaitForReady();
    SPIRead((TByte*)&wDummy, 1*2, CS_L ); //CS Keep L

    //Read Data
    LCDWaitForReady();
    SPIRead((TByte*)&wRData, 1*2, CS_H);

    wRData = MY_WORD_SWAP(wRData);
    return wRData;
}
```

5 Word LCDReadNData(TWord* pwBuf, TDWord ulWordDataCnt)

5.1 We define Max burst Transfer Len is 2048 bytes(1024 Words)

5.1.1 You have to divide into Size/1024 times to transfer if the transfer size is over 1024 words

5.1.2 or you have to polling ready for each word

5.1.3 the size of IT8951 FIFO is 2k bytes, therefore we can continued read or send data for 2kbytes(1k word)

5.2 Send Preamble 0x1000 for Write command code

5.2.1 Wait for I80 Bus Ready? (IT8951 HRDY)

5.2.2 Set CS to Low

5.2.3 Send SPI Data[2] = { 0x10, 0x00 };

5.3 Read 1 Dummy Word first

5.3.1 Wait for I80 Bus Ready?

5.3.2 Get SPI Read Data[2] = {usDummy[15:8], usDummy[7:0]};

5.3.3 Read 1 Dummy Word (2bytes)

5.4 Get N Words Read Data

5.4.1 Wait for I80 Bus Ready?

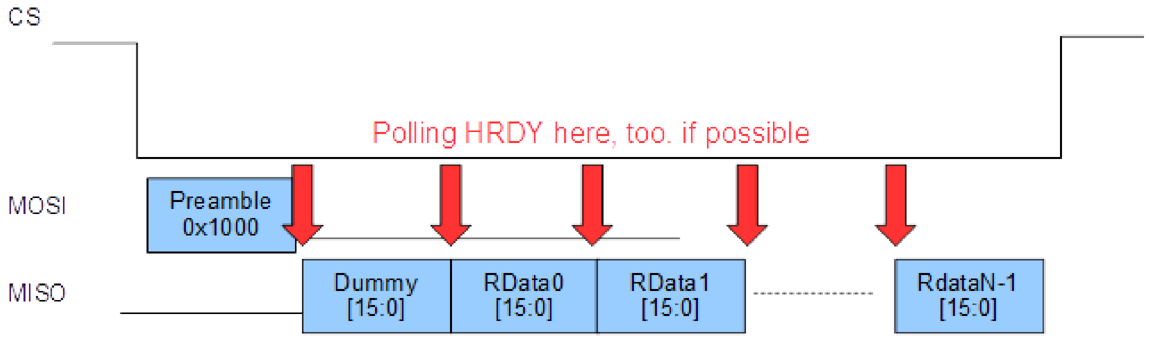
5.4.2 Get SPI Read Data[N] = {usData0[15:8], usData0[7:0],……, usDataN-1[15:8], usDataN-1[7:0]};

5.4.3 CS to High

5.5 Convert Endian from Big to Little all ReadData (Little Endian only)

5.5.1 Convert to Little Endian? (it depends on your host Endian type)

5.5.2 **Convert Endian from Big to Little all ReadData**



```
TWord LCDReadNData(TWord* pwBuf, TDWord ulSizeWordCnt)
{
    TWord wPreamble = 0;
   TWord wRData;
    TWord wDummy;
    TDWord i;

    //set type and direction
    wPreamble = 0x1000;

    //Send Preamble before reading data
    wPreamble = MY_WORD_SWAP(wPreamble);
    LCDWaitForReady();
    SPIWrite((TByte*)&wPreamble, 1*2, CS_L);

    //Read Dummy (under IT8951 SPI to I80 spec)
    LCDWaitForReady();
    SPIRead((TByte*)&wDummy, 1*2, CS_L );//CS Keep L

    //Read N Data
    //in this case, we recommend host programmer can polling LCDWaitForReady for
    //every 2 bytes(1 word)if possible to your SPI controller
    LCDWaitForReady();
    SPIRead((TByte*)pwBuf, ulSizeWordCnt *2, CS_H);

    //Convert Endian (depends on your host)
    for(i=0;i< ulSizeWordCnt ; i++)
    {
        pwBuf[i] = MY_WORD_SWAP(pwBuf[i]);
    }
}
```