SIOC嵌入式軟體實驗
實驗八：SPI

WU-YANG
Technology Co., Ltd.

support.wuyang@gmail.com

大綱

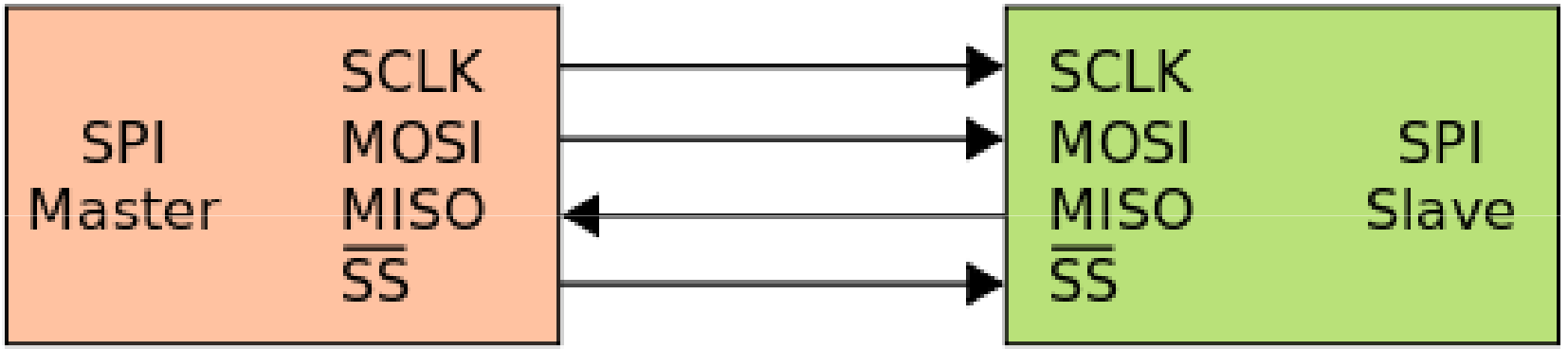
- SPI introduction
- SPI Standard Driver Library
- 實驗項目

SPI Introduction

- Serial Peripheral Interface Bus(SPI)

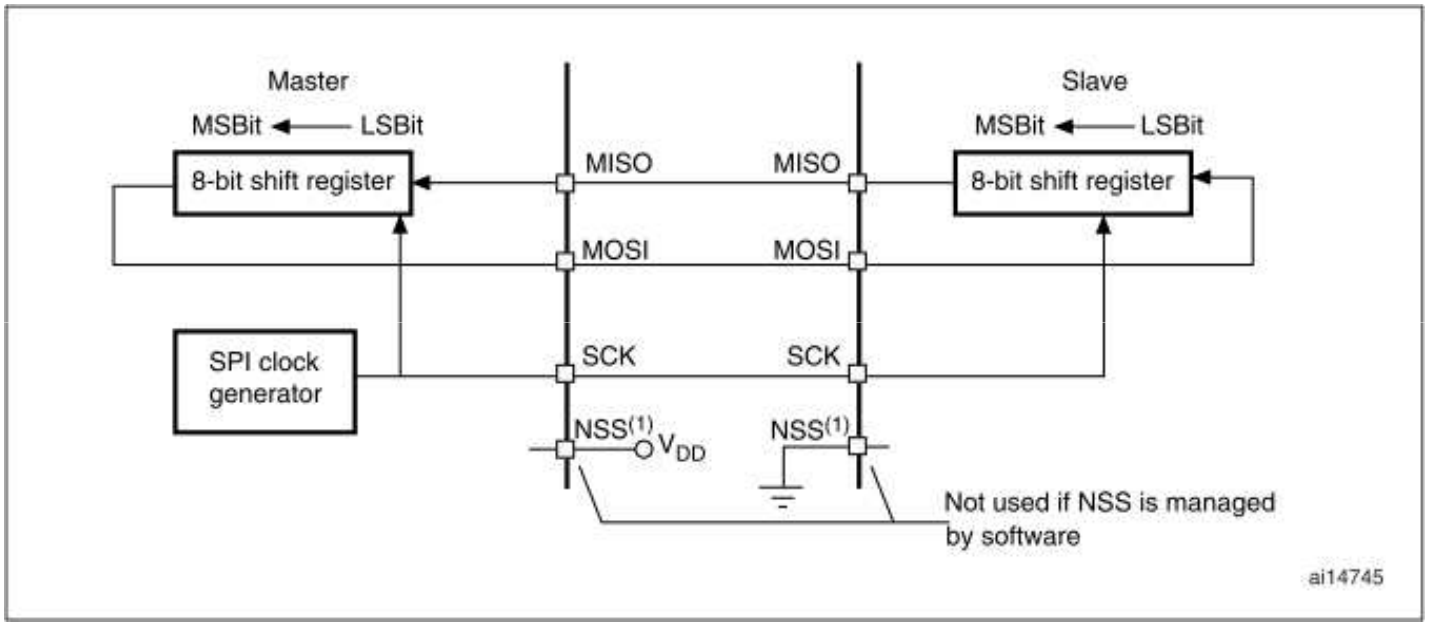
- The SPI bus specifies four logic signals:
  - SCLK: Serial Clock (output from master)
  - MOSI; SIMO: Master Output, Slave Input (output from master)
  - MISO; SOMI: Master Input, Slave Output (output from slave)
  - SS: Slave Select (active low, output from master).

單一 master 對單一 slave



單一 master 對單一 slave SPI架構

SPI Protocol



SPI Standard Driver Library

WU-YANG

Technology Co., Ltd.

SPI_InitTypeDef structure

```
typedef struct
{
	u16 SPI_Direction;
	u16 SPI_Mode;
	u16 SPI_DataSize;
	u16 SPI_CPOL;
	u16 SPI_CPHA;
	u16 SPI_NSS;
	u16 SPI_BaudRatePrescaler;
	u16 SPI_FirstBit;
	u16 SPI_CRCPolynomial;
} SPI_InitTypeDef;
```

SPI_Direction

- SPI_Direction configures the SPI unidirectional or bidirectional data mode

SPI_Direction	Description
SPI_Direction_2Lines_FullDuplex	SPI configured as 2 lines unidirectional full duplex
SPI_Direction_2Lines_Rx	Only SPI configured as 2 lines unidirectional Rx only
SPI_Direction_1Line_Rx	SPI configured as 1 line bidirectional Rx only
SPI_Direction_1Line_Tx	SPI configured as 1 line bidirectional Tx only

SPI_Mode

- SPI_Mode configures the SPI operating mode

SPI_Mode	Description
SPI_Mode_Master	**SPI configured as a master**
SPI_Mode_Slave	**SPI configured as a slave**

SPI_DataSize

- SPI_DataSize configures the SPI data size.

SPI_DataSize	Description
SPI_DataSize_16b	SPI 16-bit data frame format for transmission and reception
SPI_DataSize_8b	SPI 8-bit data frame format for transmission and reception

SPI_CPOL

- SPI_CPOL selects the serial clock steady state.

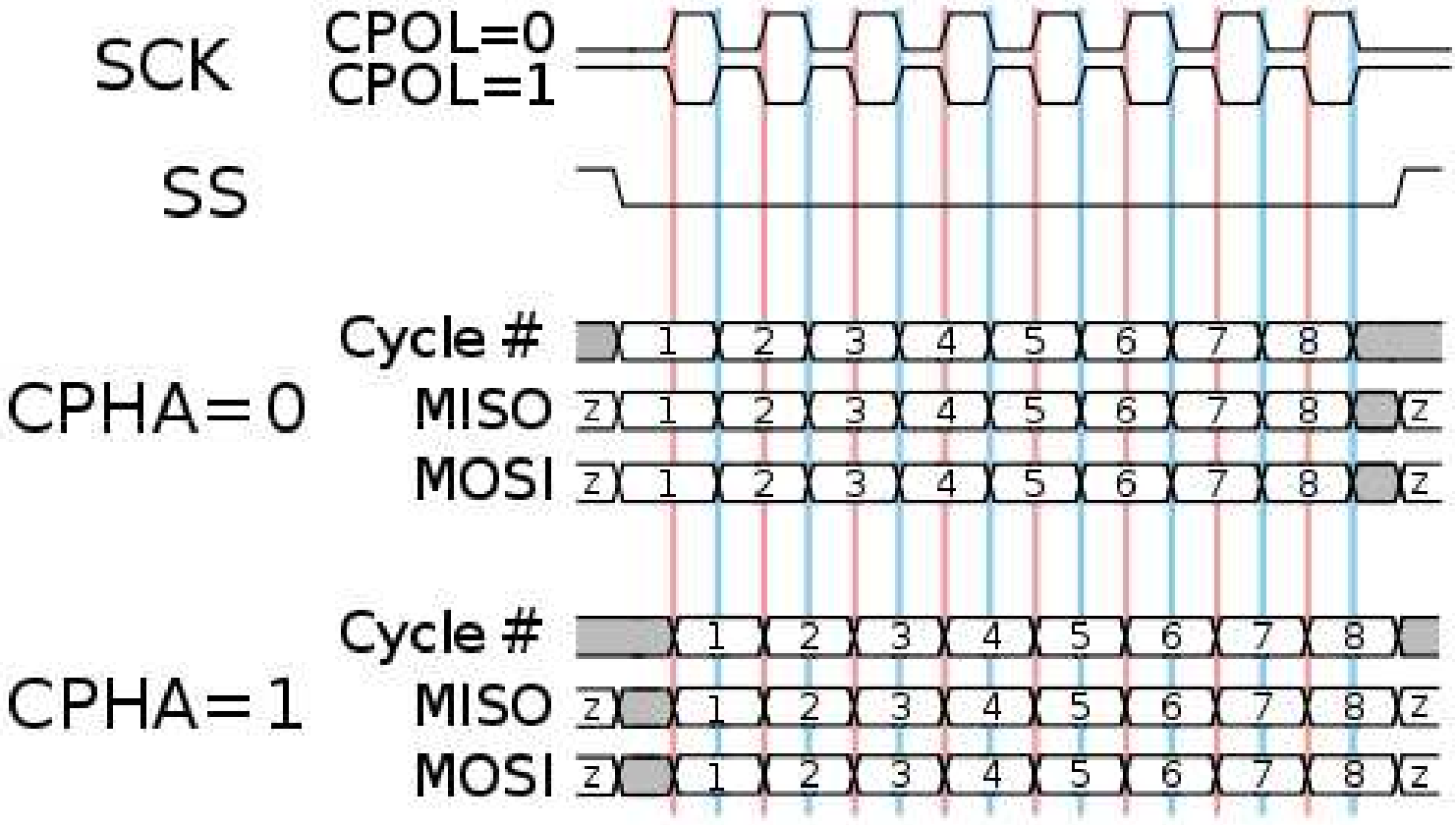
SPI_CPOL	Description
SPI_CPOL_High	Clock idle high
SPI_CPOL_Low	Clock idle low

SPI_CPHA

- SPI_CPHA configures the clock active edge for the bit capture.

SPI_DataSize	Description
SPI_CPHA_2Edge	Data is captured on the second edge
SPI_CPHA_1Edge	Data is captured on the first edge

SPI Protocol Cont.,



SPI_NSS

- SPI_NSS specifies whether the NSS signal is managed by hardware (NSS pin) or by software using the SSI bit..

SPI_NSS	Description
SPI_NSS_Hard	NSS managed by external pin
SPI_NSS_Soft	Internal NSS signal controlled by SSI bit

SPI_BaudRatePrescaler

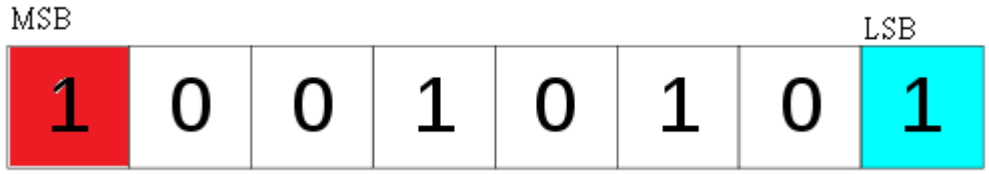
- SPI_BaudRatePrescaler is used to define the Baud Rate prescaler value which will be used to configure the transmit and receive SCK clock.

SPI_BaudratePrescaler	Description
SPI_BaudRatePrescaler2	Baud Rate Prescaler equal to 2
SPI_BaudRatePrescaler4	Baud Rate Prescaler equal to 4
SPI_BaudRatePrescaler8	Baud Rate Prescaler equal to 8
SPI_BaudRatePrescaler16	Baud Rate Prescaler equal to 16
SPI_BaudRatePrescaler32	Baud Rate Prescaler equal to 32
SPI_BaudRatePrescaler64	Baud Rate Prescaler equal to 64
SPI_BaudRatePrescaler128	Baud Rate Prescaler equal to 128
SPI_BaudRatePrescaler256	Baud Rate Prescaler equal to 256

SPI_FirstBit

- SPI_FirstBit specifies whether data transfers start from MSB or LSB bit.

SPI_FirstBit	Description
SPI_FisrtBit_MSB	First bit to transfer is the MSB
SPI_FisrtBit_LSB	First bit to transfer is the LSB



Initialize the SPI1 example

- SPI_InitTypeDef SPI_InitStructure;
- SPI_InitStructure.SPI_Direction = SPI_Direction_2Lines_FullDuplex;
- SPI_InitStructure.SPI_Mode = SPI_Mode_Master;
- SPI_InitStructure.SPI_DatSize = SPI_DatSize_16b;
- SPI_InitStructure.SPI_CPOL = SPI_CPOL_Low;
- SPI_InitStructure.SPI_CPHA = SPI_CPHA_2Edge;
- SPI_InitStructure.SPI_NSS = SPI_NSS_Soft;
- SPI_InitStructure.SPI_BaudRatePrescaler = SPI_BaudRatePrescaler_128;
- SPI_InitStructure.SPI_FirstBit = SPI_FirstBit_MSB;
- SPI_Init(SPI1, &SPI_InitStructure);

SPI_I2S_SendData function

Function name	SPI_I2S_SendData
Function prototype	SPI_I2S_SendData(SPI_TypeDef* SPIx, u16 Data)
Behavior description	Transmits data through the SPIx/I2Sx peripheral.
Input parameter1	SPIx: where x can be 1, 2 or 3 to select the SPI peripheral.
Input parameter 2	Data: Byte or half word (in SPI mode), or half word (in I2S mode) to be transmitted.

SPI_I2S_SendData function *Cont.,*

Output parameter	**None**
Return parameter	**None**
Required preconditions	**None**
Called functions	**None**

- **Example:**

```
/* Send 0xA5 through the SPI1 peripheral */
SPI_I2S_SendData(SPI1, 0xA5);
```

SPI_I2S_ReceiveData function

Function name	SPI_I2S_ReceiveData
Function prototype	**u16 SPI_I2S_ReceiveData(SPI_TypeDef* SPIx)**
Behavior description	**Returns the most recent data received through the SPIx/I2Sx peripheral.**
Input parameter1	**SPIx: where x can be 1, 2 or 3 to select the SPI/I2S? peripheral.**
Ouput parameter	**None**

SPI_I2S_ReceiveData function *Cont.,*

Output parameter	**None**
Return parameter	**The value of the received data.**
Required preconditions	**None**
Called functions	**None**

- **Example:**

```
/* Read the most recent data received by the SPI2 peripheral */
u16 ReceivedData;
ReceivedData = SPI_I2S_ReceiveData(SPI2);
```

SPI_I2S_GetFlagStatus function

Function name	SPI_I2S_GetFlagStatus
Function prototype	FlagStatus SPI_I2S_GetFlagStatus(SPI_TypeDef* SPIx, u16 SPI_I2S_FLAG)
Behavior description	Checks whether the specified SPI/I2S flag is set or not.
Input parameter1	SPIx: where x can be 1, 2 or 3 to select the SPI peripheral.
Input parameter 2	SPI_I2S_FLAG: flag to be checked. Refer to SPI_I2S_FLAG for more details on the allowed values for this parameter.

SPI_I2S_GetFlagStatus function *Cont.,*

Output parameter	**None**
Return parameter	**The new state of SPI_I2S_FLAG (SET or RESET).**
Required preconditions	**None**
Called functions	**None**

- **Example:**

```
/* Send 0xA5 through the SPI1 peripheral */
SPI_I2S_SendData(SPI1, 0xA5);
```

SPI_I2S_FLAG

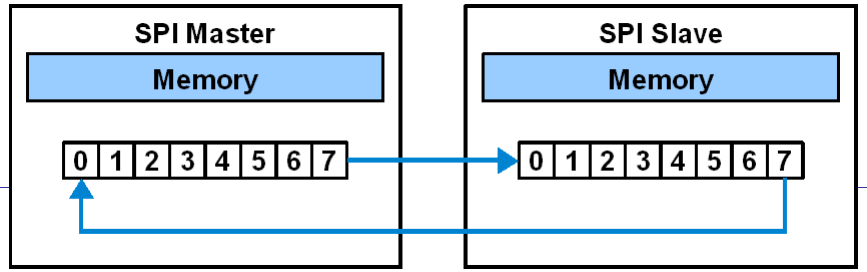
SPI_I2S_FLAG	Description
SPI_I2S_FLAG_TXE	Transmit buffer empty flag
SPI_I2S_FLAG_RXNE	Receive buffer not empty flag

- **Example:**

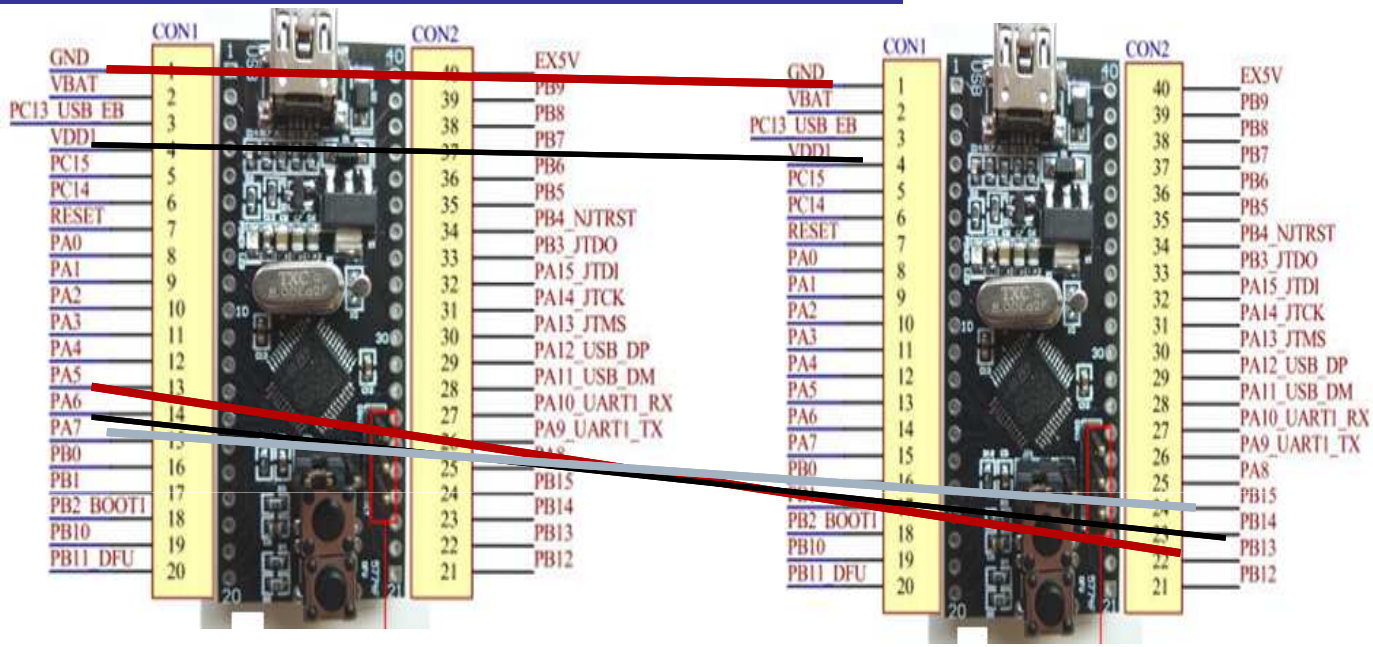
/* Test if the SPI1 transmit buffer empty flag is set or not */

FlagStatus Status;

Status = SPI_I2S_GetFlagStatus(SPI1, SPI_I2S_FLAG_TXE);



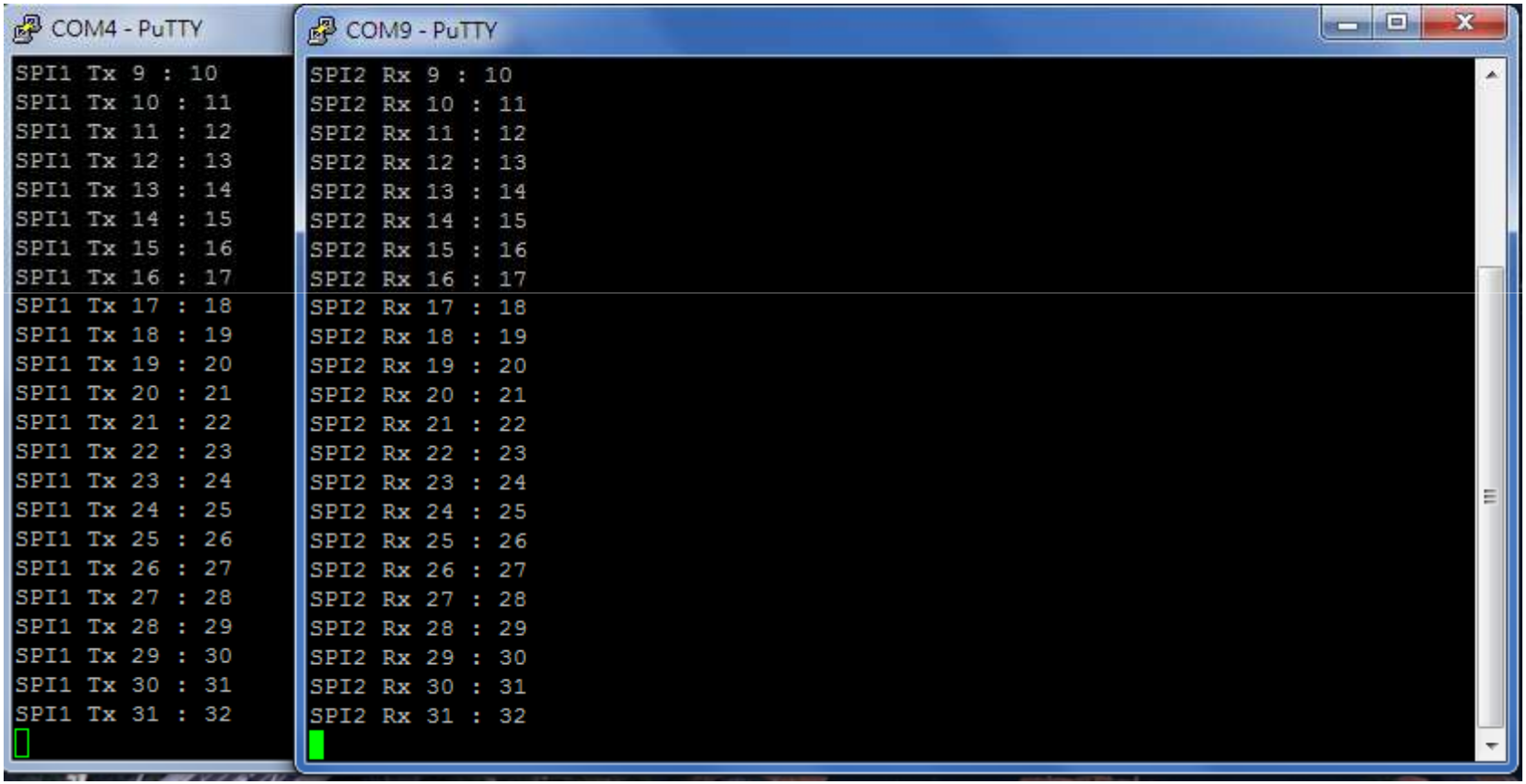
Wire Connecting Table



Pin Mapping	
Master	**Slave**
GND	GND
VDD1	VDD1
SPI1 SCK (PA.05)	SPI2 SCK (PB.13)
SPI1 MISO (PA.06)	SPI2 MISO (PB.14)
SPI1 MOSI (PA.07)	SPI2 MOSI (PB.15)

實驗一

- 將master (SPI1)32筆16位元資料傳到slave(SPI2)

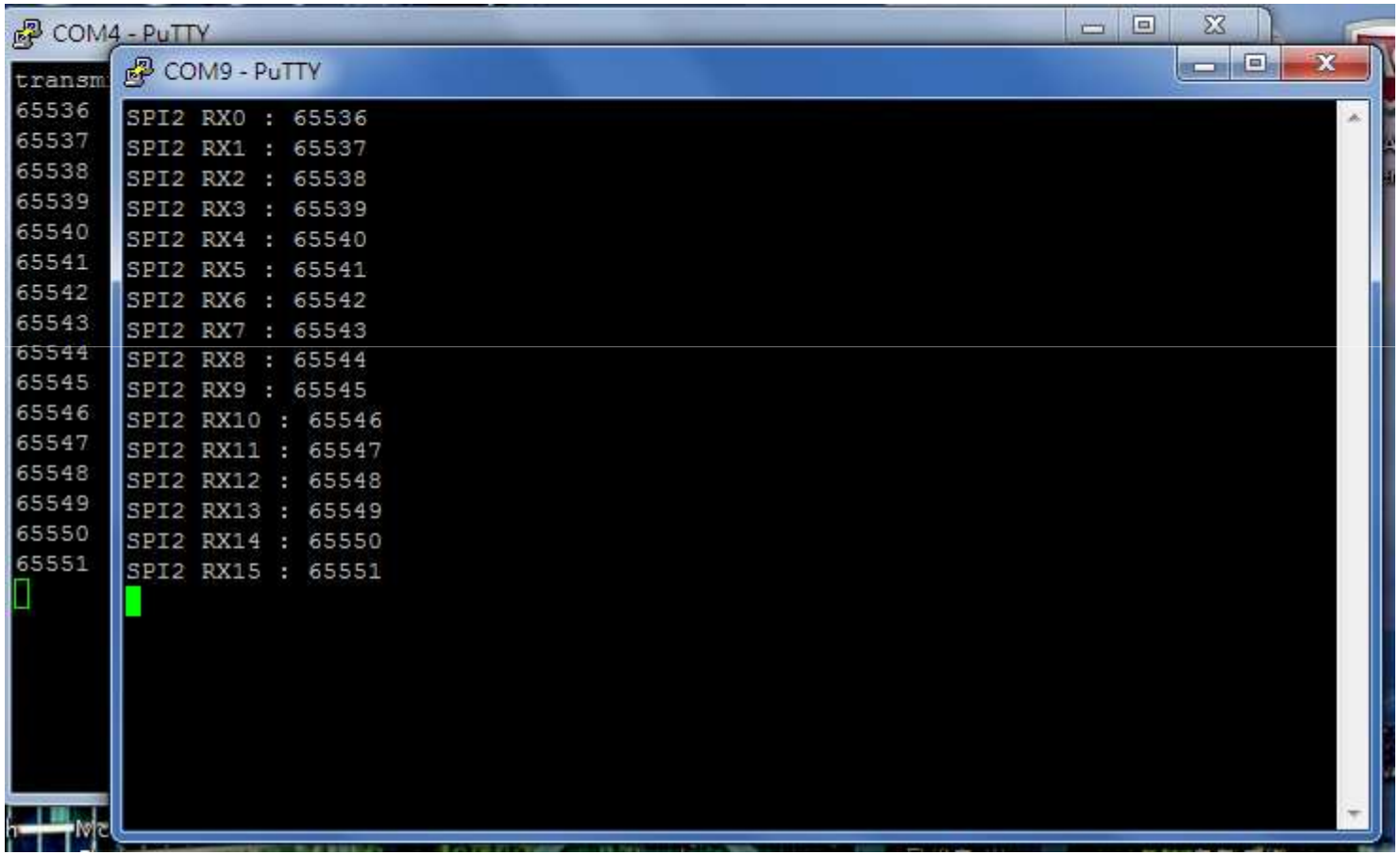


實驗一說明

- TX程式為master，RX程式為slave
- #define BufferSize 32
- uint16_t SPI1_Buffer_Tx[BufferSize]={1,2,3….,32}
- uint16_t SPI2_Buffer_Rx[BufferSize]
- uint32_t TxIdx = 0, RxIdx = 0;
- 將TX程式中SPI1_Buffer_Tx[BufferSize]，傳到RX程式中SPI2_Buffer_Rx[BufferSize]

實驗二

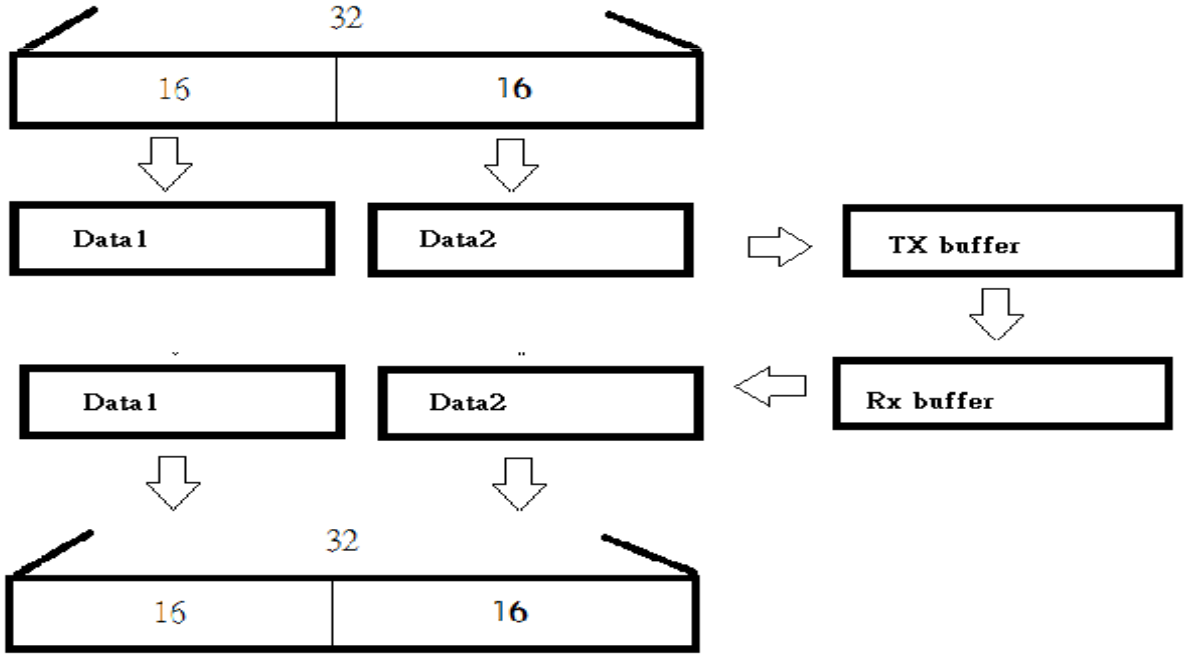
- 將master (SPI1)16筆32位元資料傳到slave(SPI2)



實驗二說明

- uint32_t SPI1_Buffer_Data[BufferSize/2] ={65536,65537,65538,… 65551};

- uint32_t SPI2_Buffer_Data[BufferSize/2]



參考資料

- 參考資料

  [1] STM32F10xxx reference manual_2011.pdf

  [2] STM32F103x8.pdf

Q & A

WU-YANG

Technology Co., Ltd.