TP: **BUS**

Ocquidant Sébastien

(mail)

TP: **BUS**

Objectifs du TP:

Connaître les principales caractéristiques des bus utilisés dans l'industrie et ***Maîtriser les techniques d'interfaçage*** *(logiciel et matériel)*

- **Comprendre le schéma d'une carte dans le but de la réutiliser.**
- **Analyser les documents constructeurs pour identifier les performances d'un bus synchrone**
- **Mise en œuvre un bus SPI**
- **Faire les bonnes mesures valider un bus synchrone**

TP: **BUS**

Pré requis

- Langages: C
- Eléments de base en électronique numérique et analogique,
- Notions élémentaires sur les Processeurs et les Microcontrôleurs.
- Programmation du périphérique GPIO du STM32
- Utilisation d'un oscilloscope

Volume horaire

- **IESE4:** TP: 12H.

TP: **BUS**

Plan du TP

1. Introduction au BUS SPI

2. 3x4h de mise en pratique du BUS SPI
   1. Etude de cas
   2. Programmation du périphérique SPI sur STM32
   3. Mesure des performances du bus SPI

3. Rédaction d'un rapport de TP
   Date limite: dernier jour du TP + 2 semaines

TP: **BUS**

1. Introduction, principaux types de bus, caractéristiques

Principaux types de bus

- Parallèles synchrones: ex: PC104, PCI…
- Séries synchrones, ex: SPI, I2C
- Parallèles asynchrones, ex: VME, interface mémoire (SRAM, Nandflash, norflash)
- Séries asynchrones, ex: RS232, RS485,CAN, LIN
- Bus multiplexées synchrone et asychrone : DDRx-SDRAM, Flash parallel
- Liaison série avec clock recovery: PCIe, Ethernet

TP: **BUS**

Les bus inter-composants, le **SPI**

- Bus série synchrone développé par Motorola Freescale et destiné à faire communiquer :
  - un processeur (en général un microcontrôleur)
  - Avec des circuits périphériques (Flash, CAN, CNA, afficheurs LCD etc....)
- Interface SPI répandue sur de nombreux microcontrôleurs (HCS12 Freescale, PIC, Intel, ST, Microchip ...)

TP: **BUS**

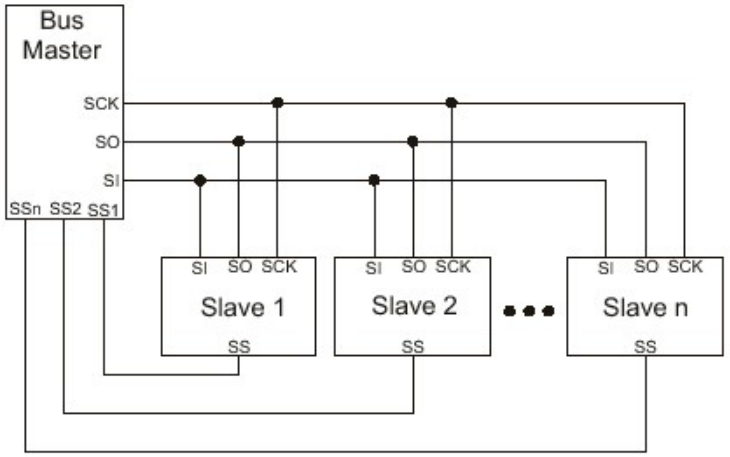
Les bus inter-composants, le **SPI**

- Architecture Maître/Esclave :
  - Le maître est responsable des activités de transfert sur le bus,
  - L'esclave répond aux commandes issues du maître,
  - 1 seul Maître par transaction,
  - Nb d'esclaves limités seulement par le nombre de signaux SS/ disponibles; dans les faits, quelques uns.
- Communication Full-duplex
- Horloge en général 1 à plusieurs dizaines MHz => Taux de transfert : 1-100 Mbit/s et parfois > 400Mbps pour des mémoires Flash Q-SPI (4 lignes MISO/MOSI half duplex)

TP:BUS

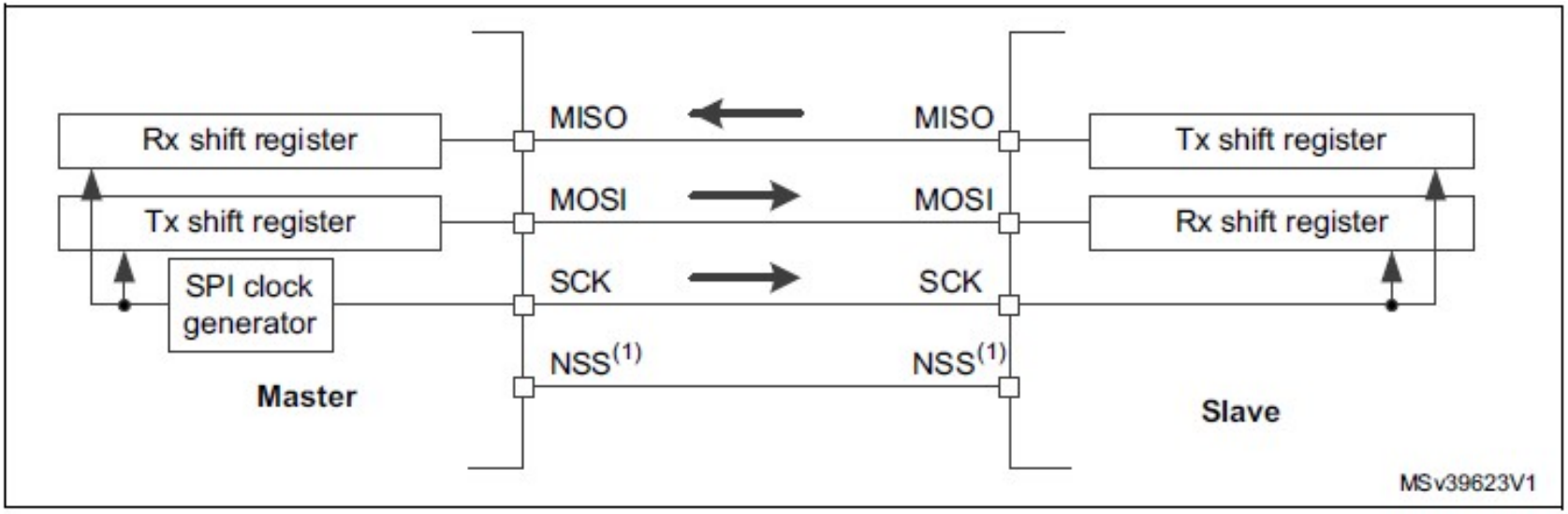
les bus inter-composants, le **SPI**

- Interface 4 fils :



SCK	Serial Clock	Maître : Sortie Esclave : Entrée	Horloge de synchronisation Les données sont décalées/latchées sur un front montant ou descendant de cette horloge
MOSI	Master OUT Slave IN	Maître : Sortie Esclave : Entrée	Données transférées en sortie du maître vers l'esclave
MISO	Master IN Slave OUT	Maître : Entrée Esclave : Sortie	Données transférées en sortie de l'esclave vers le maître
/SS ou /CS	Slave/Chip Select	Maître : Sortie Esclave : Entrée	Sélectionne l'esclave avec lequel un transfert va commencer doit rester au niveau bas pendant toute la transaction

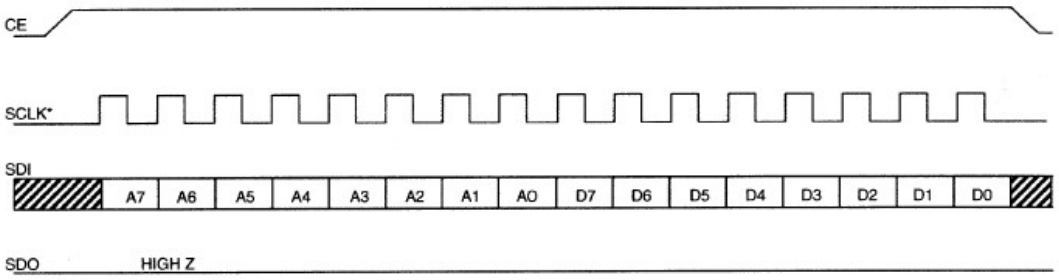
Principe de la transmission SPI



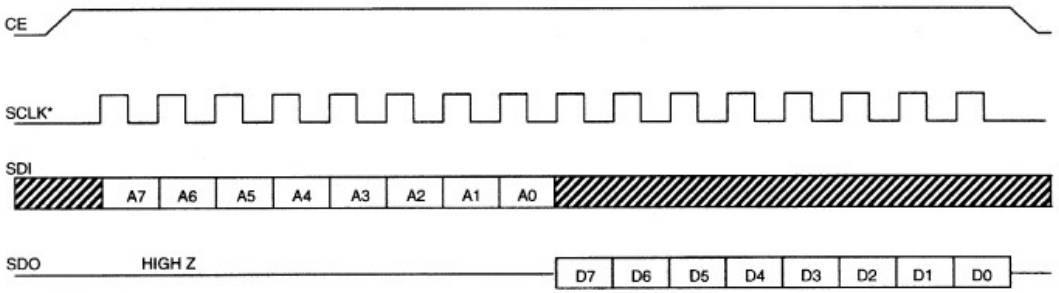
TP: **BUS**

Les bus inter-composants, le **SPI**

SPI SINGLE BYTE WRITE Figure 4



SPI SINGLE-BYTE READ Figure 5



TP: **BUS**

Timing d'un bus synchrone

- Les data sont envoyées de manière synchrone avec l'horloge SCK
- Les data sont réceptionnées en utilisant cette horloge:

  On dit que les data sont *échantillonnées*, « *latchées* » par l'horloge

Pour être correctement échantillonnées, les data doivent respecter certains timings vis à-vis de l'horloge.

La data doit être présent avant l'arrivée du front d'échantillonnage: on parle de **temps de setup** (Tsu)

La data doit être maintenu un certain temps après l'arrivée du front d'échantillonnage: on parle de **temps de hold** (Thd)

Ces temps sont du aux temps de propagation dans le composant, les charges et décharges des capacités parasites.

L'horloge pour le composant slave devra aussi respecter certains timings. Elle devra avoir
un temps haut minimum,
un temps bas minimum.
une période min. (ou fréquence max)

Dans les circuits numériques, ce sont les temps SU et HD des bascules D qui limitent la fréquence max du circuit, entre autre.

TP: **BUS**

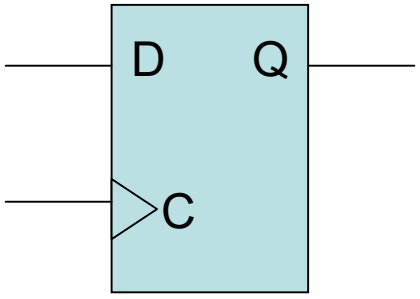
Tsu et Thd par l'exemple

- Exemple d'une bascule D

  élément de base du registre à décalage par exemple (d'une entrée SPI)

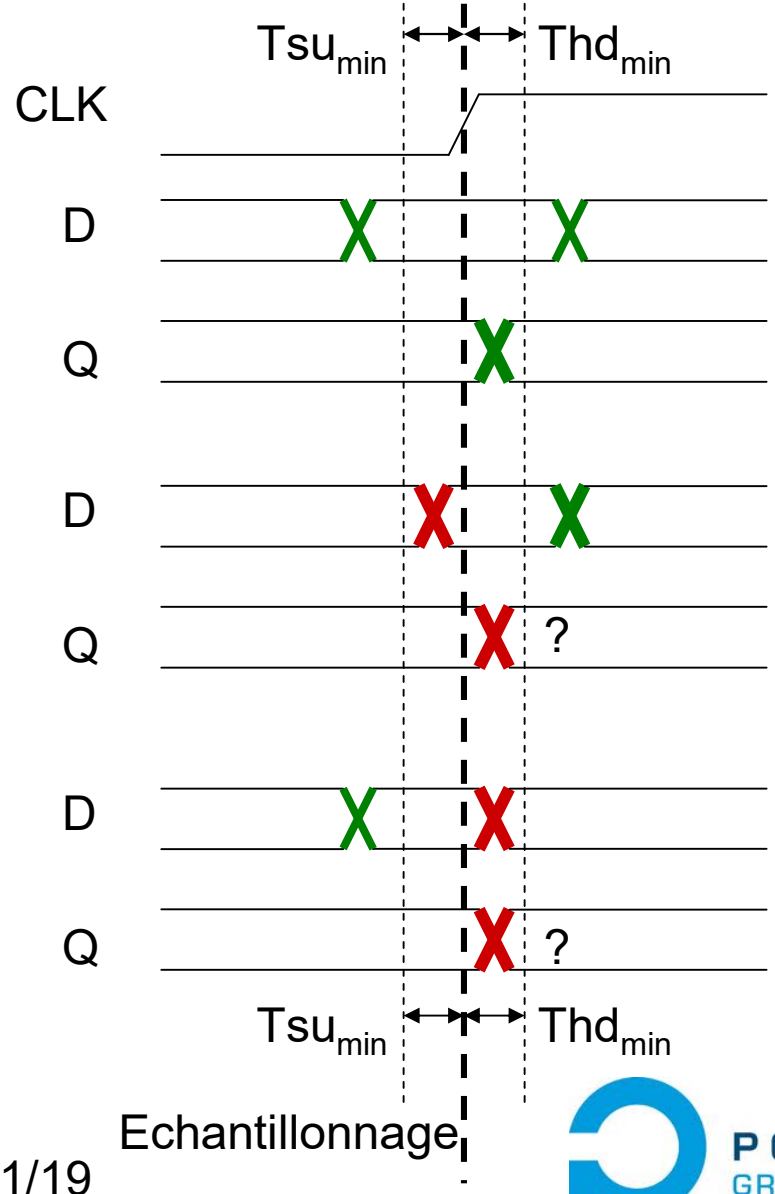
Rappel du principe de fonctionnement

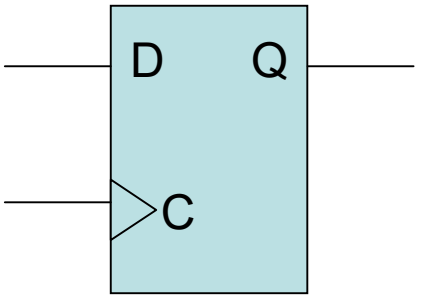
Q=D quand un front montant apparait sur l'entrée CLK



TP: **BUS**

Tsu et Thd par l'exemple





Tsu et THD respectés
La sortie Q est correct

Tsu n'est pas respecté
La sortie Q est indéterminée

Thd n'est pas respecté
La sortie Q est indéterminée

TP: **BUS**

Timing d'un bus synchrone

- Conséquence pour le master?

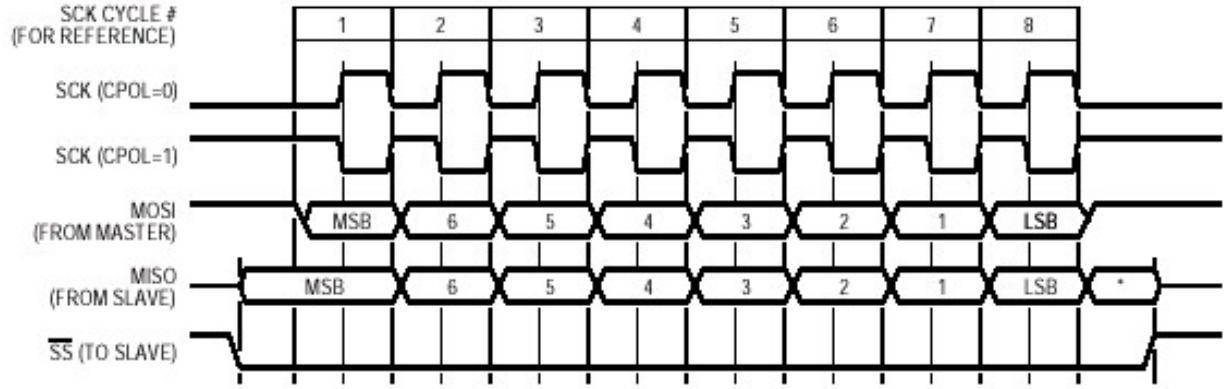
Le composant master aura des contraintes à respecter pour fournir ces signaux afin de respecter les requis du composant slave

Le composant slave impose les contraintes que le composant master devra respecter

C'est pour cela que les contrôleurs de bus sont configurables

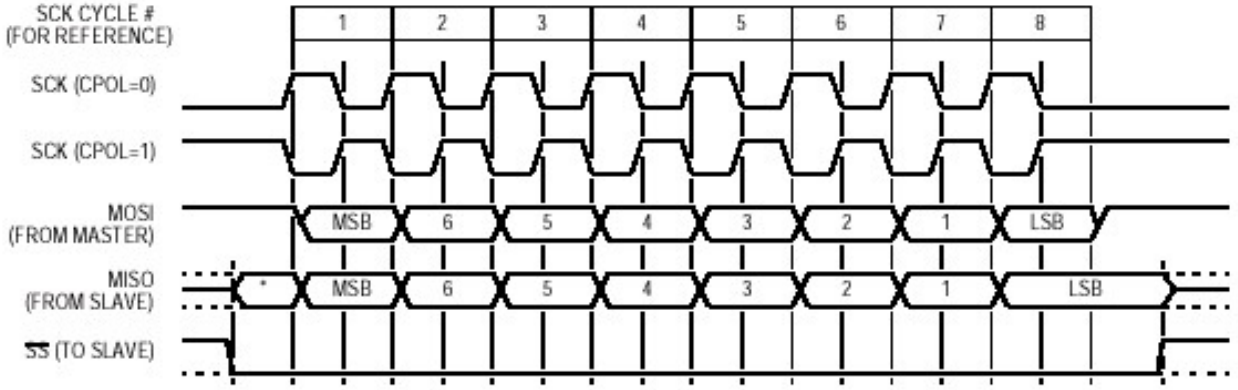
TP: **BUS**

Les bus inter-composants, le **SPI**



*Not defined but normally MSB of character just received.

Figure 8-1. CPHA Equals Zero SPI Transfer Format



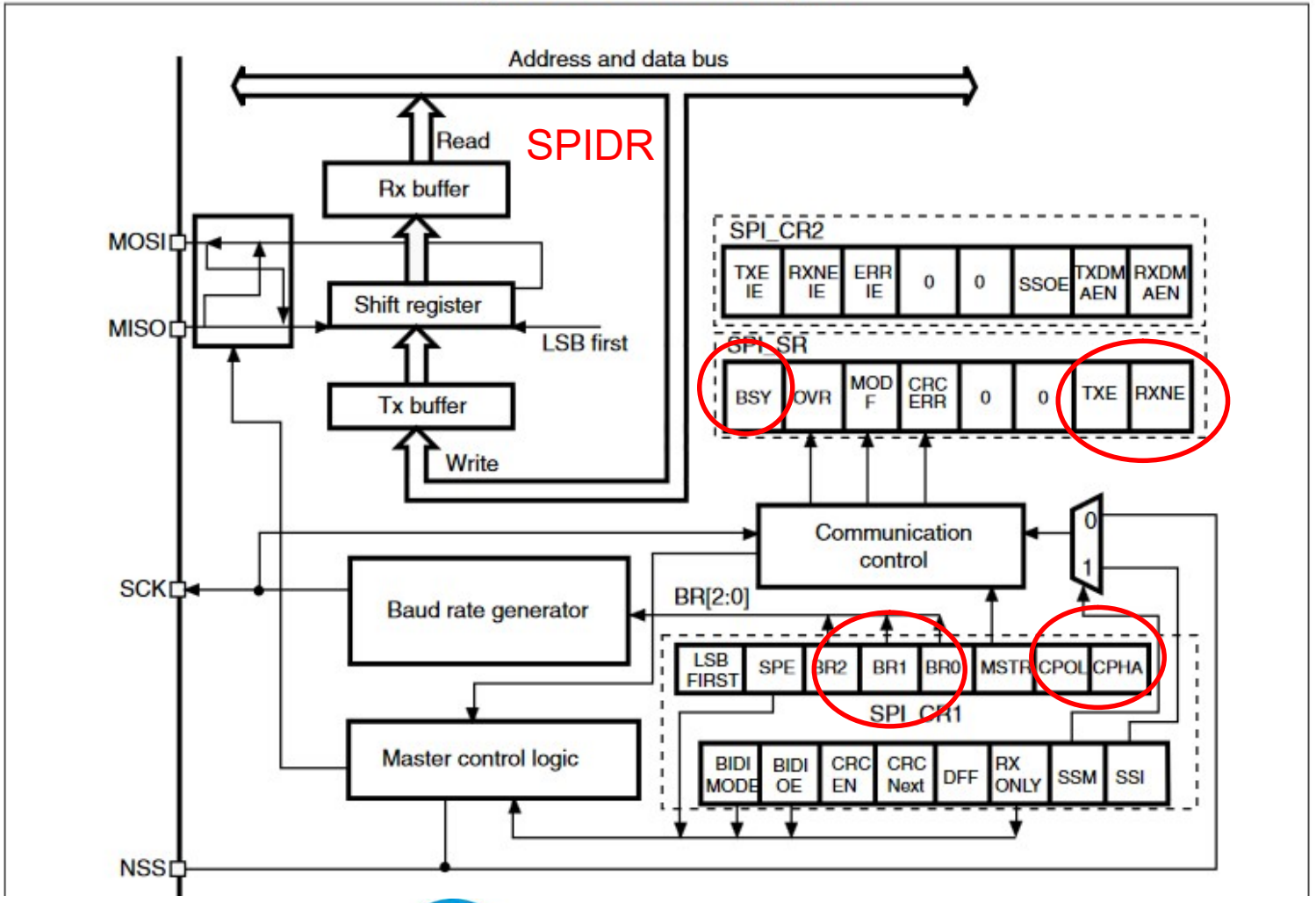
*Not defined but normally LSB of previously transmitted character.

Figure 8-2. CPHA Equals One SPI Transfer Format

TP: **BUS**

Les bus inter-composants, **le bloc SPI du STM32**

Figure 246. SPI block diagram



TP: **BUS**

Les bus inter-composants, **le bloc SPI du STM32**

Lisez bien les petites notes en bas de page des datasheets

26.3.12 SPI status flags

Three status flags are provided for the application to completely monitor the state of the SPI bus.

Tx buffer empty flag (TXE)

When it is set, the TXE flag indicates that the Tx buffer is empty and that the next data to be transmitted can be loaded into the buffer. The TXE flag is cleared by writing to the SPI_DR register.

Rx buffer not empty (RXNE)

When set, the RXNE flag indicates that there are valid received data in the Rx buffer. It is cleared by reading from the SPI_DR register.

Busy flag (BSY)

The BSY flag is set and cleared by hardware (writing to this flag has no effect).

When BSY is set, it indicates that a data transfer is in progress on the SPI (the SPI bus is busy). There is one exception in master bidirectional receive mode (MSTR=1 and BDM=1 and BDOE=0) where the BSY flag is kept low during reception.

The BSY flag can be used in certain modes to detect the end of a transfer, thus preventing corruption of the last transfer when the SPI peripheral clock is disabled before entering a low-power mode or an NSS pulse end is handled by software.

The BSY flag is also useful for preventing write collisions in a multimaster system.

The BSY flag is cleared under any one of the following conditions:

- When the SPI is correctly disabled
- When a fault is detected in Master mode (MODF bit set to 1)
- In Master mode, when it finishes a data transmission and no new data is ready to be sent
- In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

Note: It is recommended to use always the TXE and RXNE flags (instead of the BSY flags) to handle data transmission or reception operations.