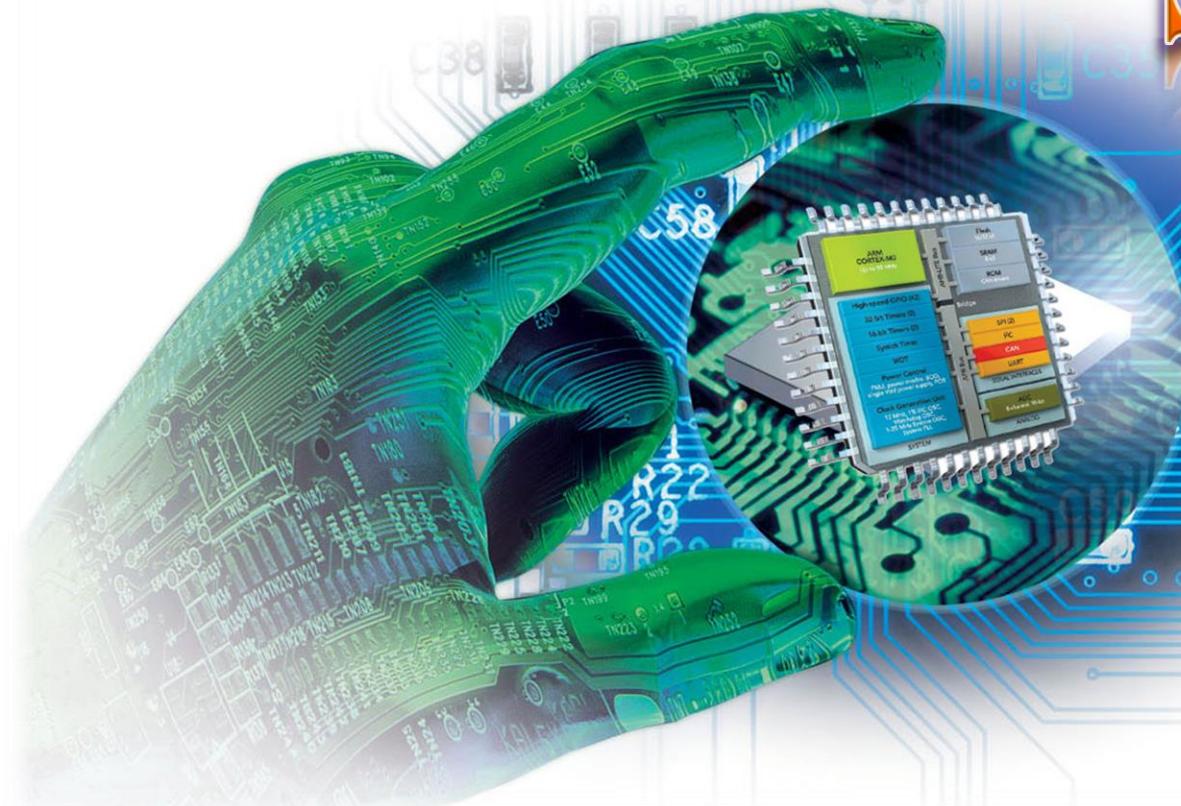
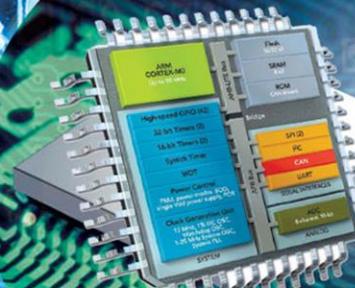


# مُتَحَكِّمَات

# STM32

## 11



# مُوضُوعات المحاضرة:

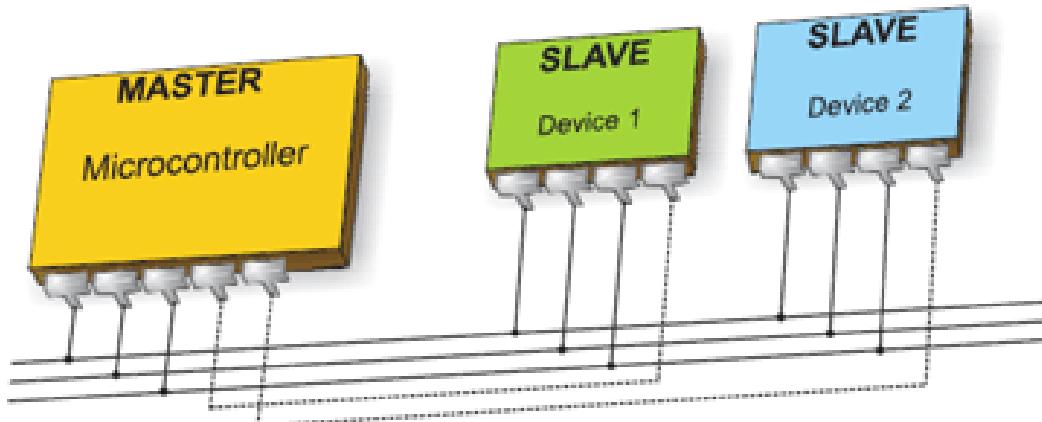
- مبدأ عمل البرتوكول SPI
- مزايا بروتوكول الاتصال التسلسلي SPI في متحكمات STM32
- المخطط الصندوقي لوحدة SPI في متحكمات STM32
- أنماط العمل المختلفة لوحدة SPI في متحكمات STM32
- دوال مكتبة HAL المستخدمة للتعامل مع وحدة SPI
- ضبط إعدادات وحدة SPI في متحكمات STM32

# مقارنة بين بروتوكولات الاتصال التسلسلي الشائعة:

	<b>RS232</b>	<b>RS485</b>	<b>I2C</b>	<b>SPI</b>	<b>M-wire</b>	<b>1-wire</b>	<b>USB</b>	<b>CAN</b>
<b>Sync/Async</b>	Async	Async.	Sync.	Sync.	Sync.	Async.	Async.	Async.
<b>Type</b>	peer-peer	master/slaves	multi-master	multi-master	master/slaves	master/slaves	host/device	multi-master
<b>Duplex</b>	full	half	half	full	full	half	half	half
<b>Signaling</b>	single-ended	Differential	single-ended	single-ended	single-ended	single-ended	Differential	Differential
<b>Max Devices No.</b>	2	32, 128, 256	40 (cap=400pf)	8 (cap, circuit)	8 (cap, circuit)	20 (cap, power)	127 per controller	2048
<b>Data Rate</b>	Up to 115Kbps	Up to 35Mbps	Std.: 100kbps Fast: 400kbps Hi: 3.4Mbps	Up to 10Mbps	Up to 1Mbps	Std.: 16.3Kbps Overdrive: 142kbps	Low: 1.5Mbps Full: 12Mbps Hi : 480Mbps	Up to 1Mbps
<b>Max. Length</b>	15m	1200m (at 100kbps)	6m	3m	3m	300m	5m	1000m (at 62kbps)
<b>Pin Count</b>	2* (Tx, Rx)	2 (A, B)	2 (SDA, SCL)	3 + SS* (SI, SO, SCK)	3 + SS* (DI, DO, SK)	1 (IO)	2 (A+, A-)	2 (CAN_H, _L)
<b>Interfacing</b>	HW	HW	SW   HW	HW   SW	HW   SW	HW & SW	protocol stack	HW & SW
<b>Flow Control</b>	HW or SW handshake	HW or SW handshake	Acknowledge from slave	None	None	CRC, Pulling	Polling by controller	CSMA / CDAMP

# SPI

## Serial Peripheral Interface



## بروتوكول الاتصال التسلسلي SPI (Serial Peripheral Interface)

- تم تطويره من قبل شركة Motorola.
- هو بروتوكول متزامن Full-duplex.
- لا يتم الانتخاب من خلال العناوين وإنما من خلال خط اختيار SS.
- مناسب أكثر للتطبيقات ذات Streaming Data مثل ADC.
- إن عدد الأجهزة على خط النقل بحدود 12~8 جهاز.
- الطول الأعظمي على ناقل SPI يمكن أن يصل إلى 3 أمتر.
- سرعة نقل البيانات على ناقل SPI تصل إلى 10Mbit.

## بروتوكول الاتصال التسلسلي SPI (Serial Peripheral Interface)

- لا يملك آلية للتأكد فيما إذا كان الجهاز الآخر على الخط أم لا.
- لا يملك آلية مصادقة للتأكد من وصول البيانات.
- التركيب الداخلي للنافذة SPI هو عبارة عن مسجل إزاحة S-to-P.
- يعتبر بروتوكول SPI أكثر استخداماً وانتشاراً من I2C.
- حزمة البيانات يمكن أن تكون بطول أكبر من 8-bit خلافاً لـ I2C.
- من أشهر التطبيقات: ADC, DAC, Data Flash Memory, MMC, Serial EEPROM, RTC, LCD, Sensors.

## بروتوكول الاتصال التسلسلي SPI (Serial Peripheral Interface)

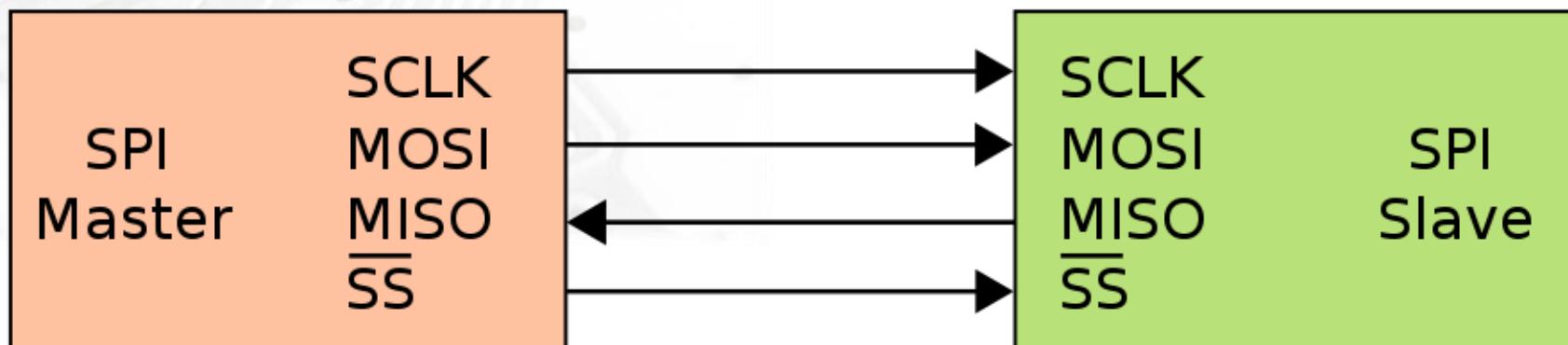
يعتمد على أربع خطوط للتalking مع الوحدات المحيطية على الناقل:

Master Output / Slave Input :**MOSI** ○

Master Input / Slave Output :**MISO** ○

Serial Clock - Synchronization :**SCK** ○

Slave Select (Optional) :**~SS** ○

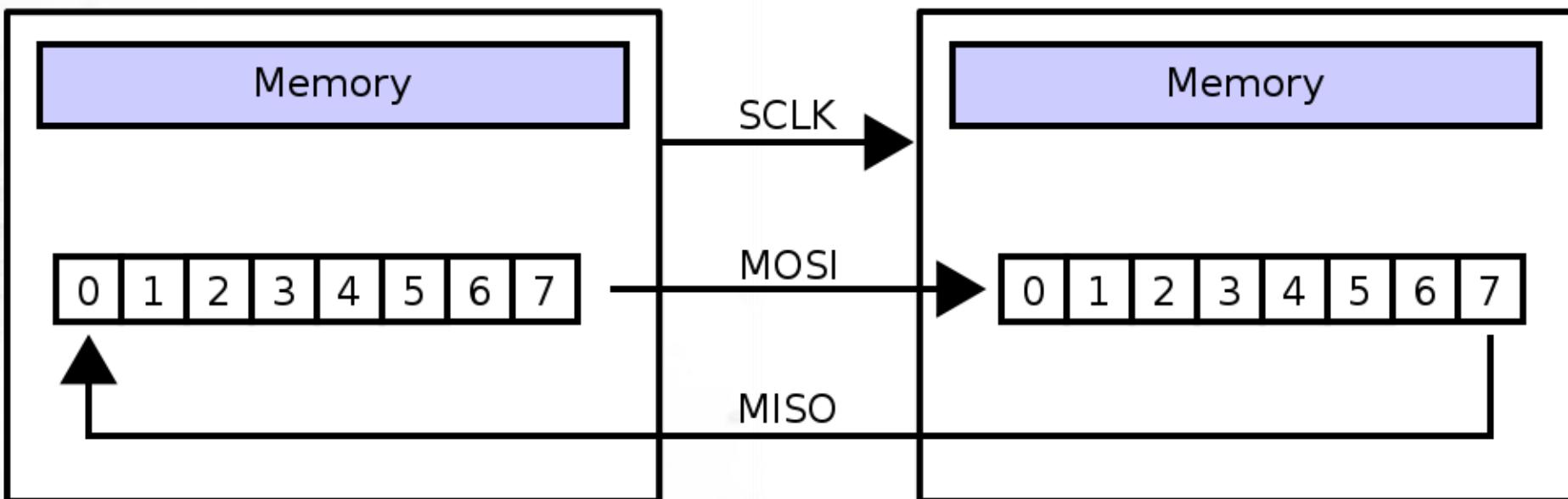


# بروتوكول الاتصال التسلسلي SPI في متحكمات STM32

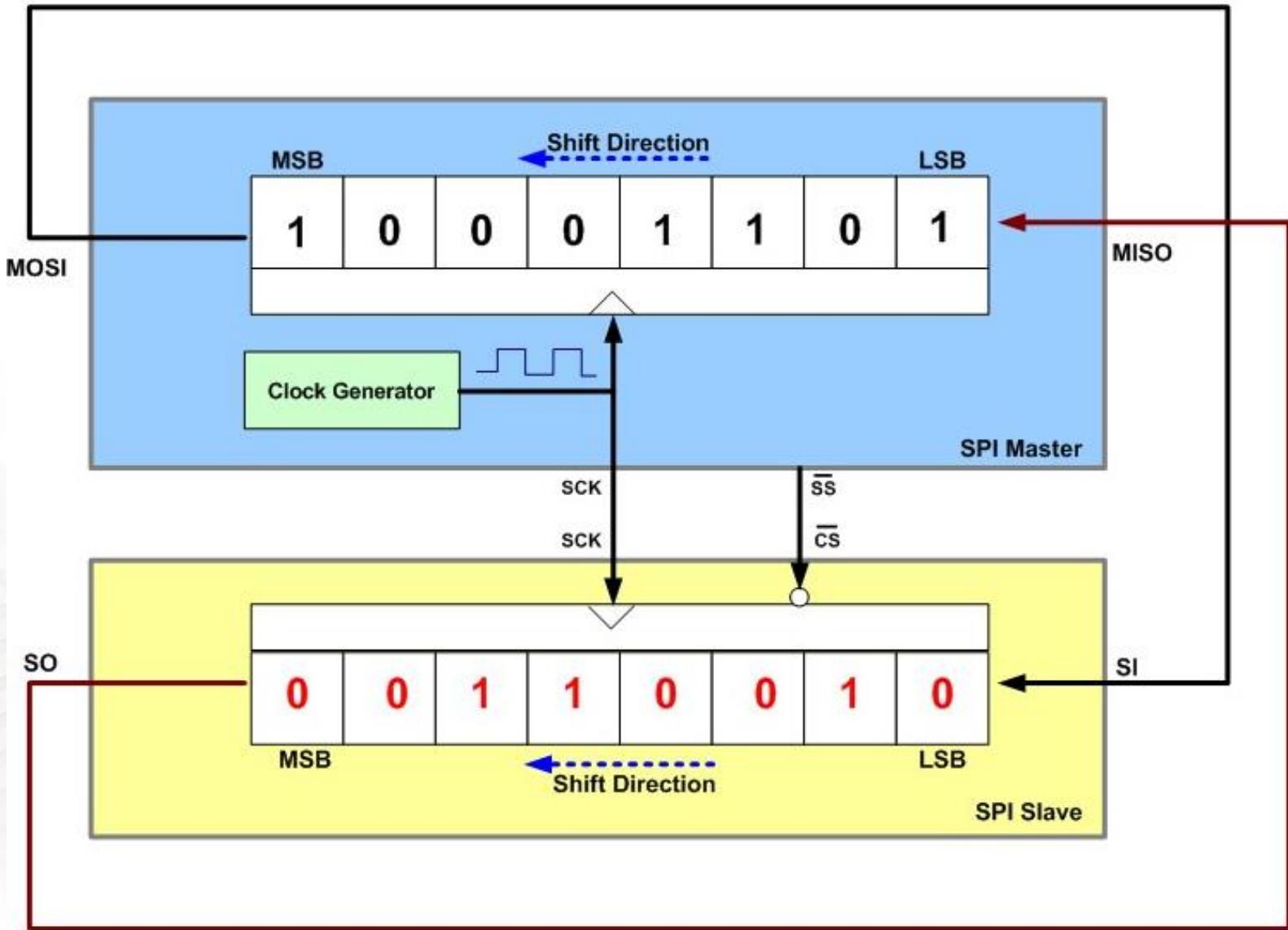
## ❖ مبدأ تراسل البيانات في ناقل الاتصال التسلسلي SPI

**Master**

**Slave**

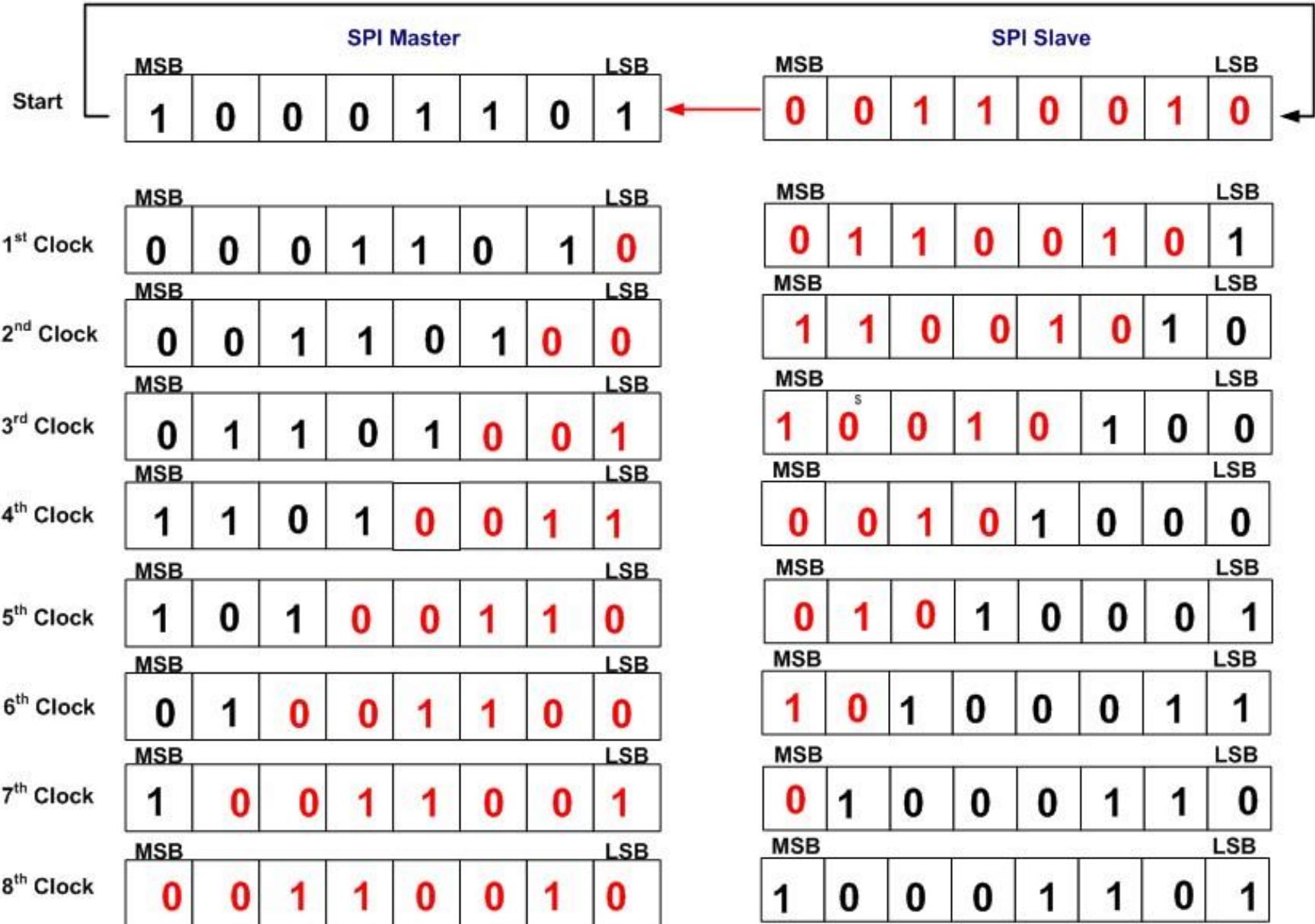


# بروتوكول الاتصال التسلسلي SPI في متحكمات STM32



Ref: <http://www.emicro.com/blog/?p=1050>

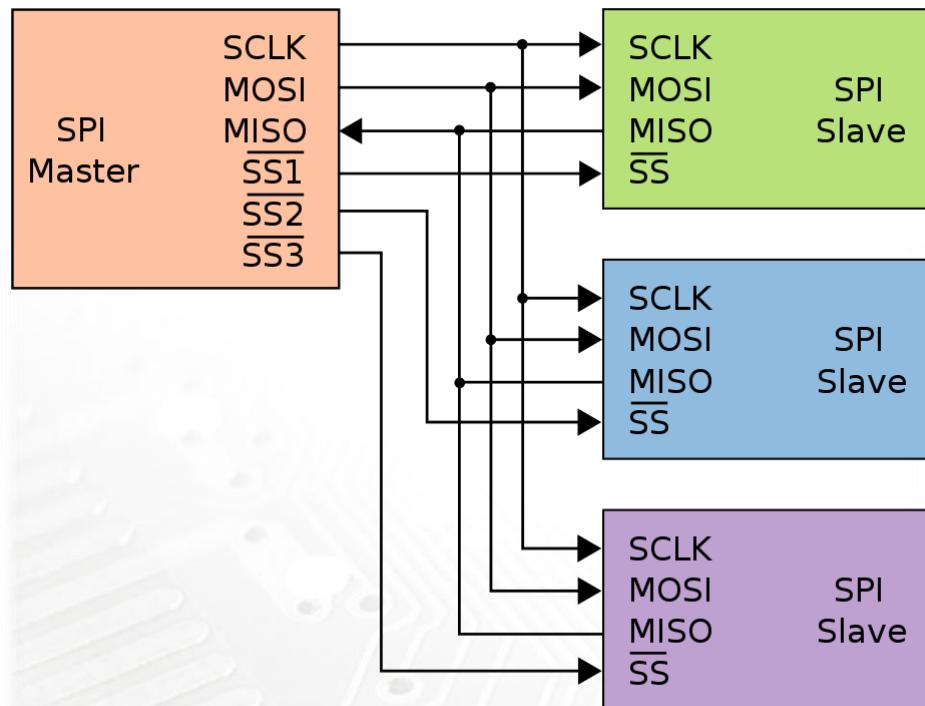
# بروتوكول الاتصال التسلسلي SPI في متحكمات STM32



SPI Master and Slave Data Transfer Diagram

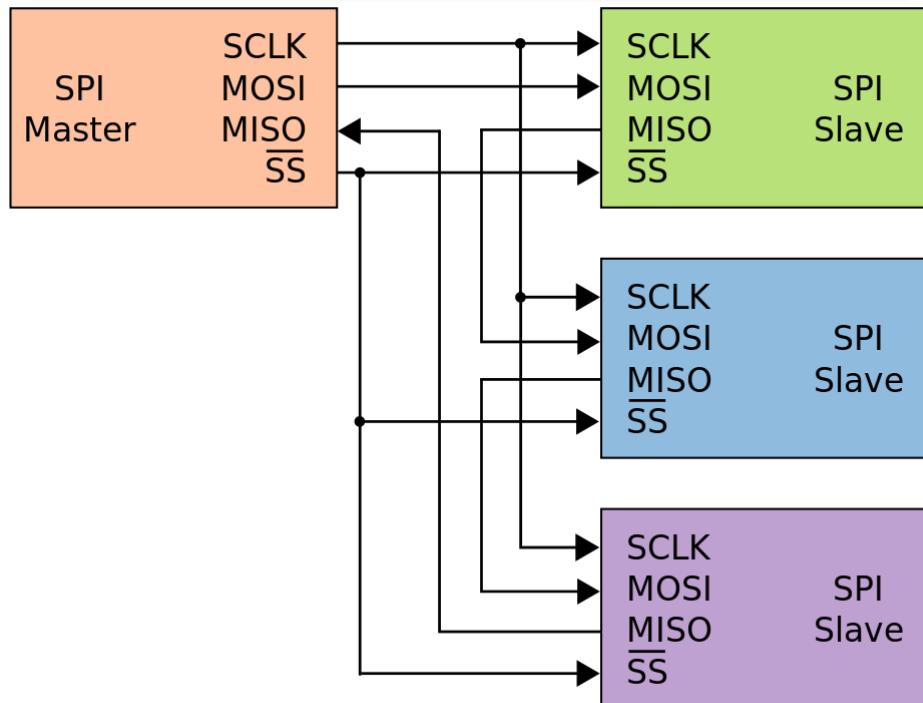
# بروتوكول الاتصال التسلسلي SPI في متحكمات STM32

## أنماط التوصيل لنقل الاتصال التسلسلي SPI



Daisy chain SPI configuration

### Independent slave SPI configuration

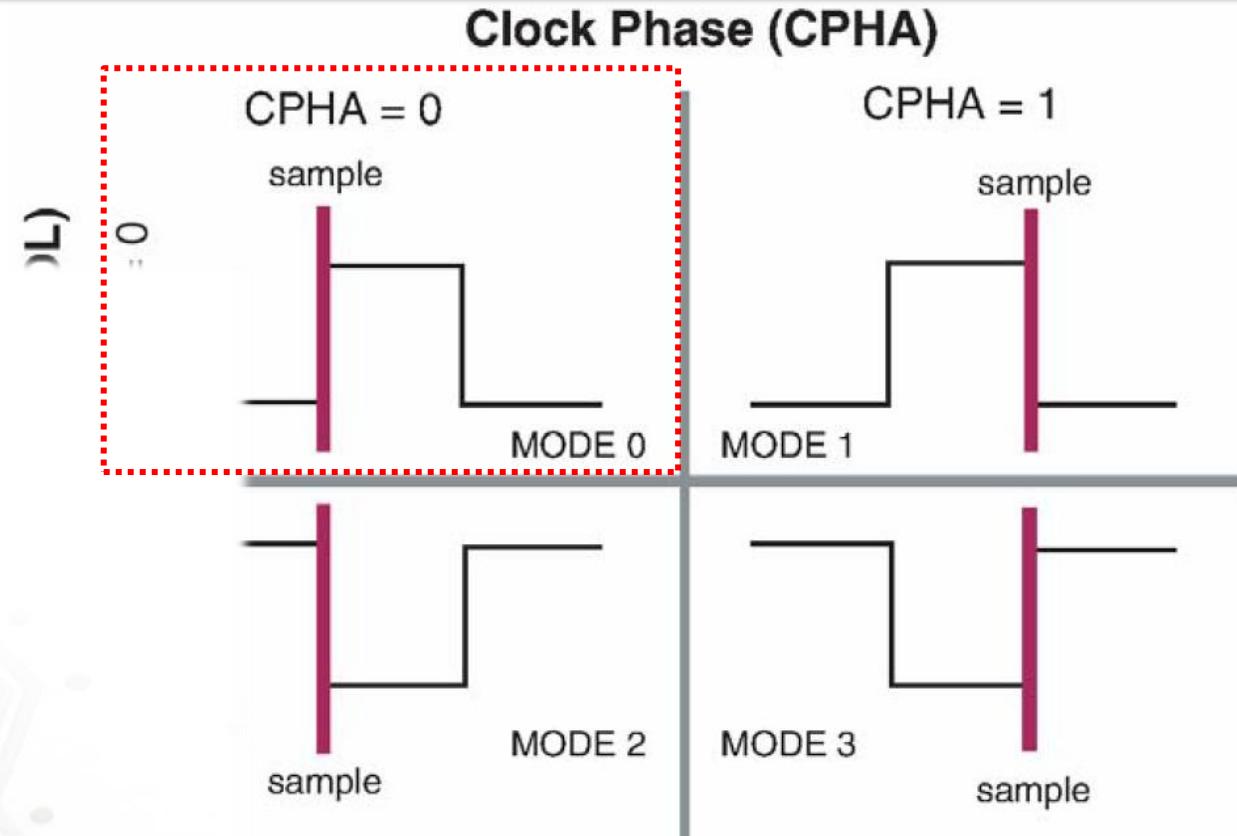


# بروتوكول الاتصال التسلسلي SPI في متحكمات STM32

## قطبية وطور إشارة التزامن في بروتوكول الاتصال التسلسلي SPI

Mode	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

data **read** on **falling edge**  
 data **change** on **rising edge**

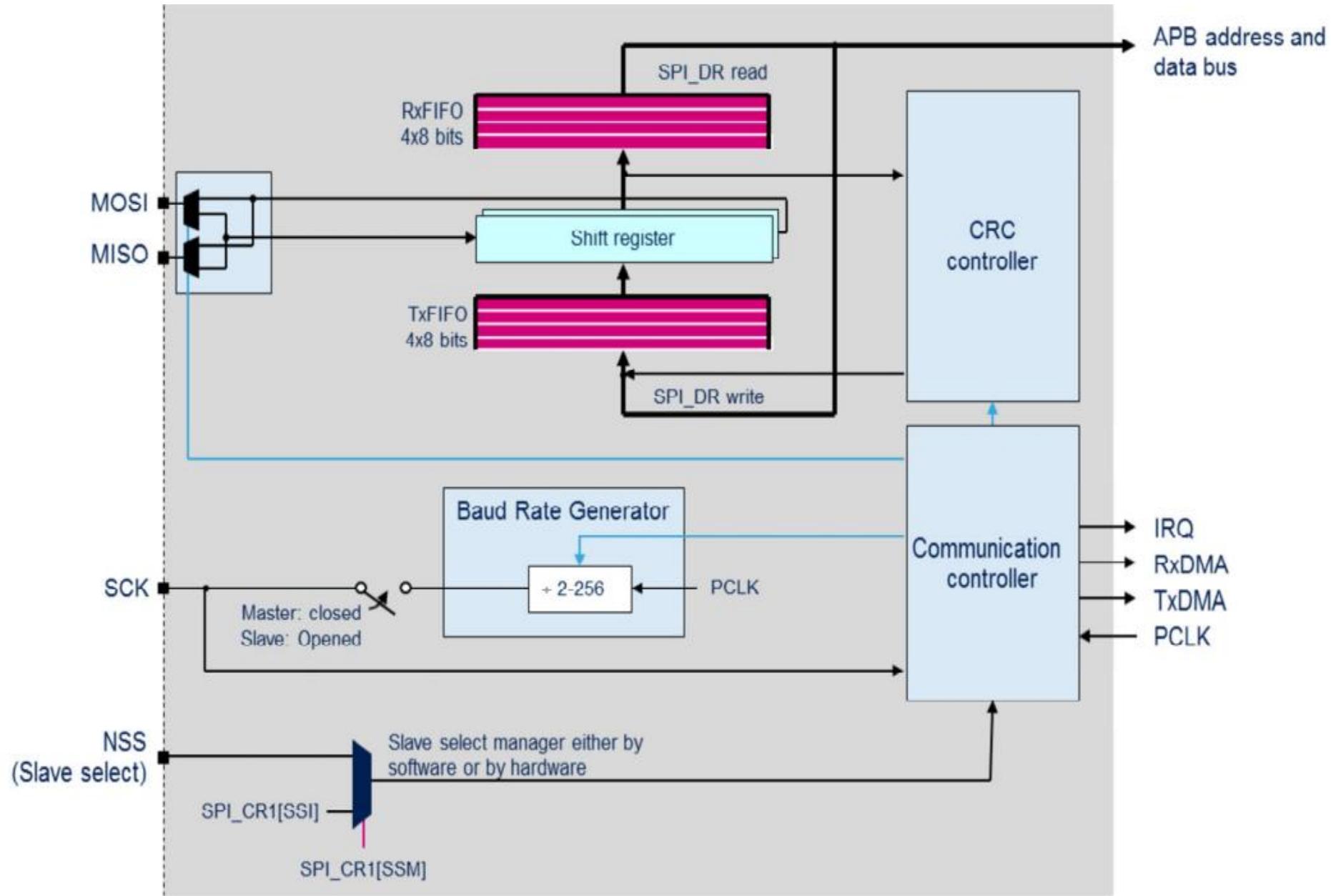


	Leading Edge	Trailing Edge	SPI Mode
CPOL = 0, CPHA = 0	Sample (Rising)	Setup (Falling)	0
CPOL = 0, CPHA = 1	Setup (Rising)	Sample (Falling)	1
CPOL = 1, CPHA = 0	Sample (Falling)	Setup (Rising)	2
CPOL = 1, CPHA = 1	Setup (Falling)	Sample (Rising)	3

- Full-duplex synchronous transfers on three lines**
- Simplex synchronous transfers on two lines with or without a bidirectional data line**
- 8- or 16-bit transfer frame format selection**
- Master or slave operation**
- Multimaster mode capability**
- 8 master mode baud rate prescalers (fPCLK/2 max.)**
- Slave mode frequency (fPCLK/2 max)**

- NSS pin management by hardware or software for both master and slave
- Programmable clock polarity and phase
- Programmable data order with MSB-first or LSB-first shifting
- Dedicated transmission and reception flags with interrupt capability
- SPI bus busy status flag

# المخطط الصندوقى لوحدة SPI في متحكمات STM32



- Master or slave (multi-master & multi slave support)
- Full-duplex, simplex or half-duplex
- Motorola and TI standards supported

وبحيث لا تتجاوز سرعة الاتصال  $f_{pclk}/2$ ، وباستخدام سلكين على الأقل

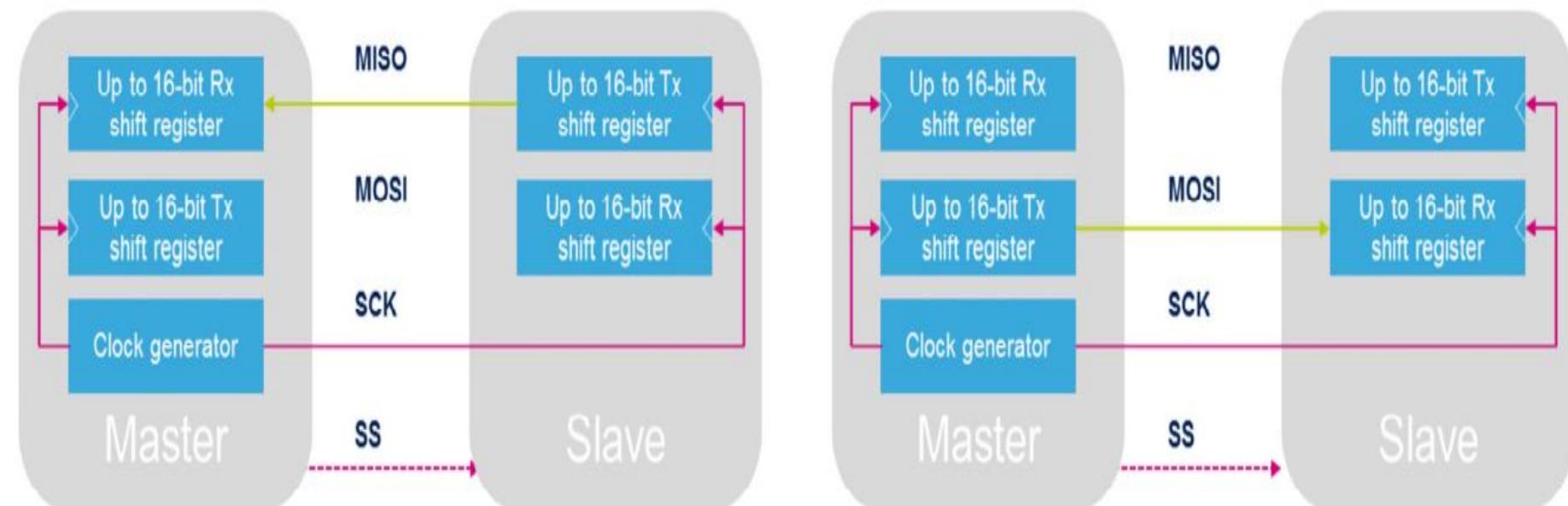
للاتصال التسلسلي وباتجاه واحد، كما يمكن أن تعمل وحدة SPI بنمط :

- Polling
- Interrupt
- DMA

# 1- Simplex Mode

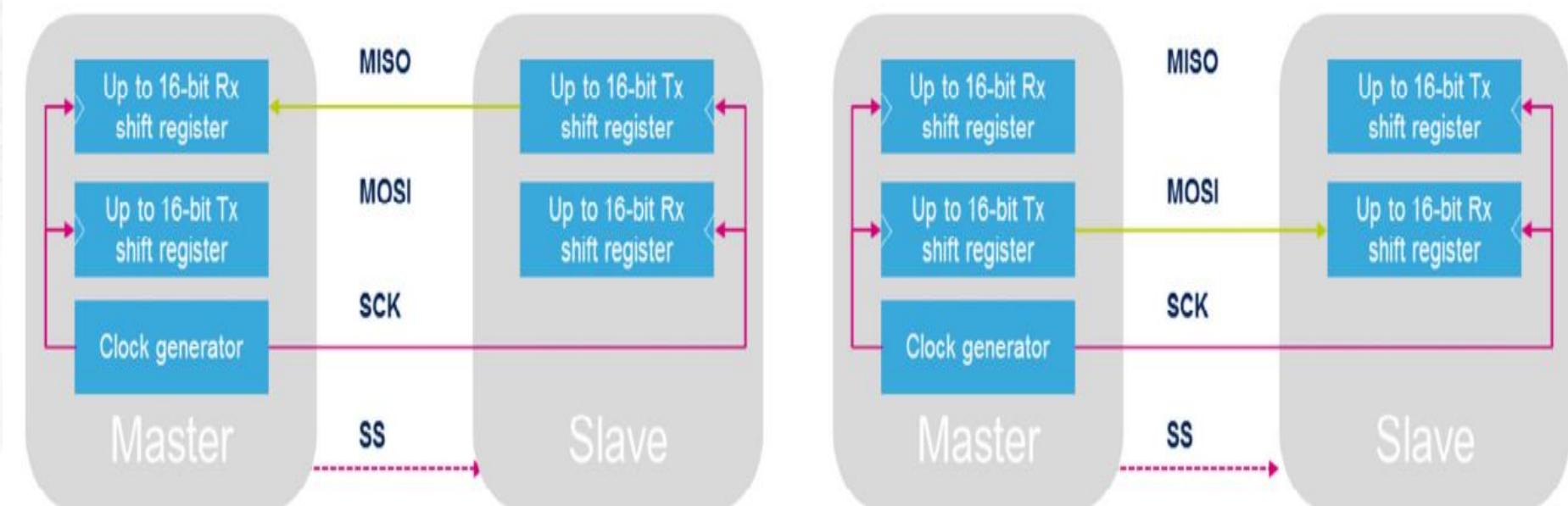
في هذا النمط من العمل يكون أحد الطرفين مرسل فقط والآخر مستقبل فقط حيث يمكن للبيانات أن تتدفق باتجاه واحد وبحسب اتجاه البيانات يتم اختيار خط البيانات حيث نحتاج إلى خط بيانات وحيد يتم اختيار هذا النمط من العمل عند اختيار نمط transmit-only أو receive-only

**receive-only**



# 1- Simplex Mode

فعلى سبيل المثال عندما يكون الـ master في نمط transmit-mode والـ slave في نمط receive-mode عندها يتم وصل قطب الـ slave من الـ master MOSI مع قطب NSS اختياري لوجود جهاز slave في هذا النمط يكون وصل قطب NSS اختياري لوجود جهاز واحد فقط



## 2- Half-duplex Mode

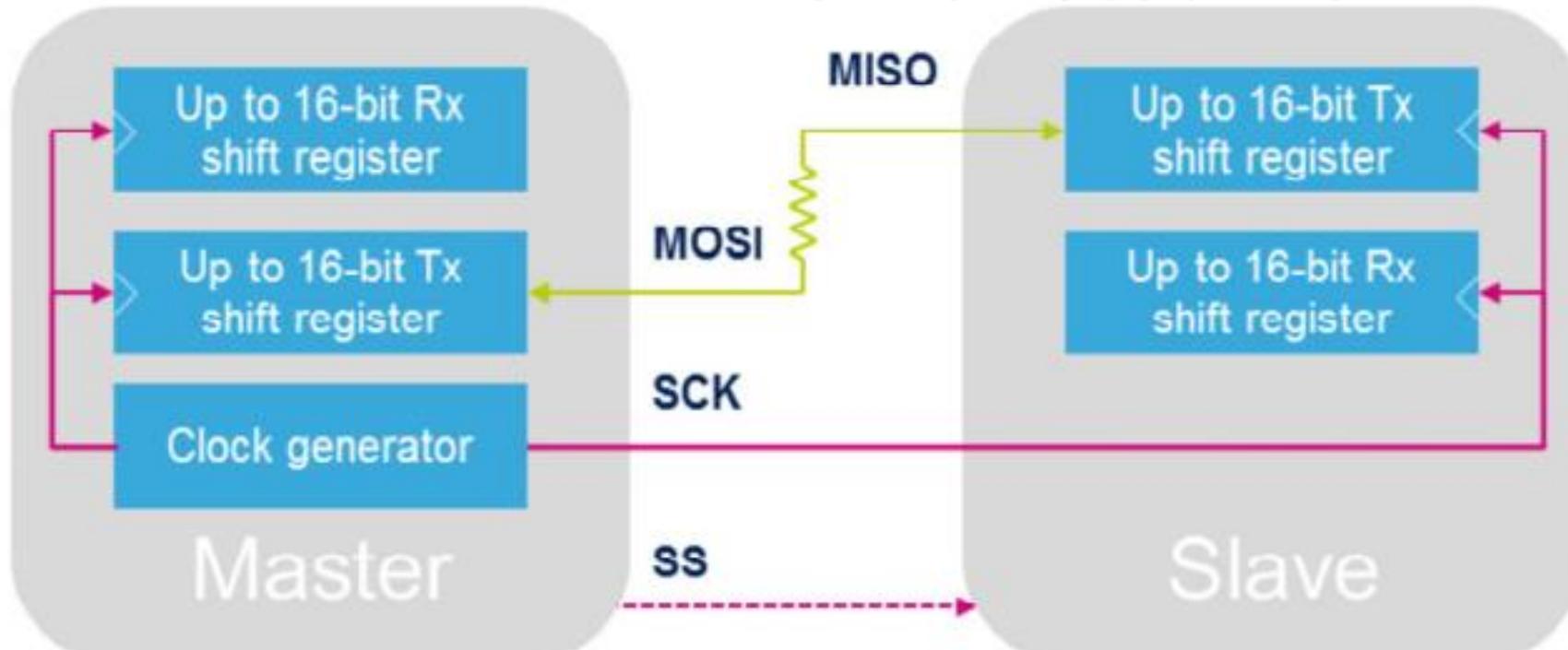
في هذا النمط من العمل يتم استخدام خط بيانات وحيد، حيث يتم وصل

قطب الـ MISO من الـ Master مع الـ MOSI من الـ Slave

خلال مقاومة 1Kohm

وبإمكان كل من الـ Master والـ Slave إرسال واستقبال البيانات ولكن

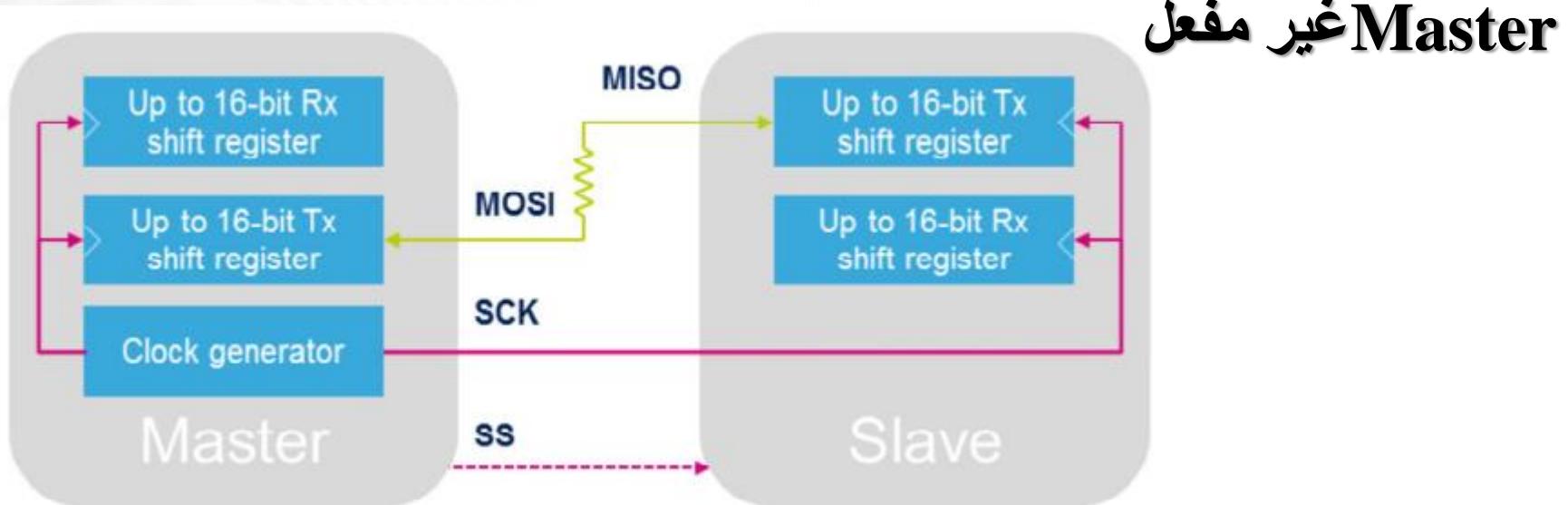
في اللحظة الواحدة يكون إما مرسلاً أو مستقبلاً



## 2- Half-duplex Mode

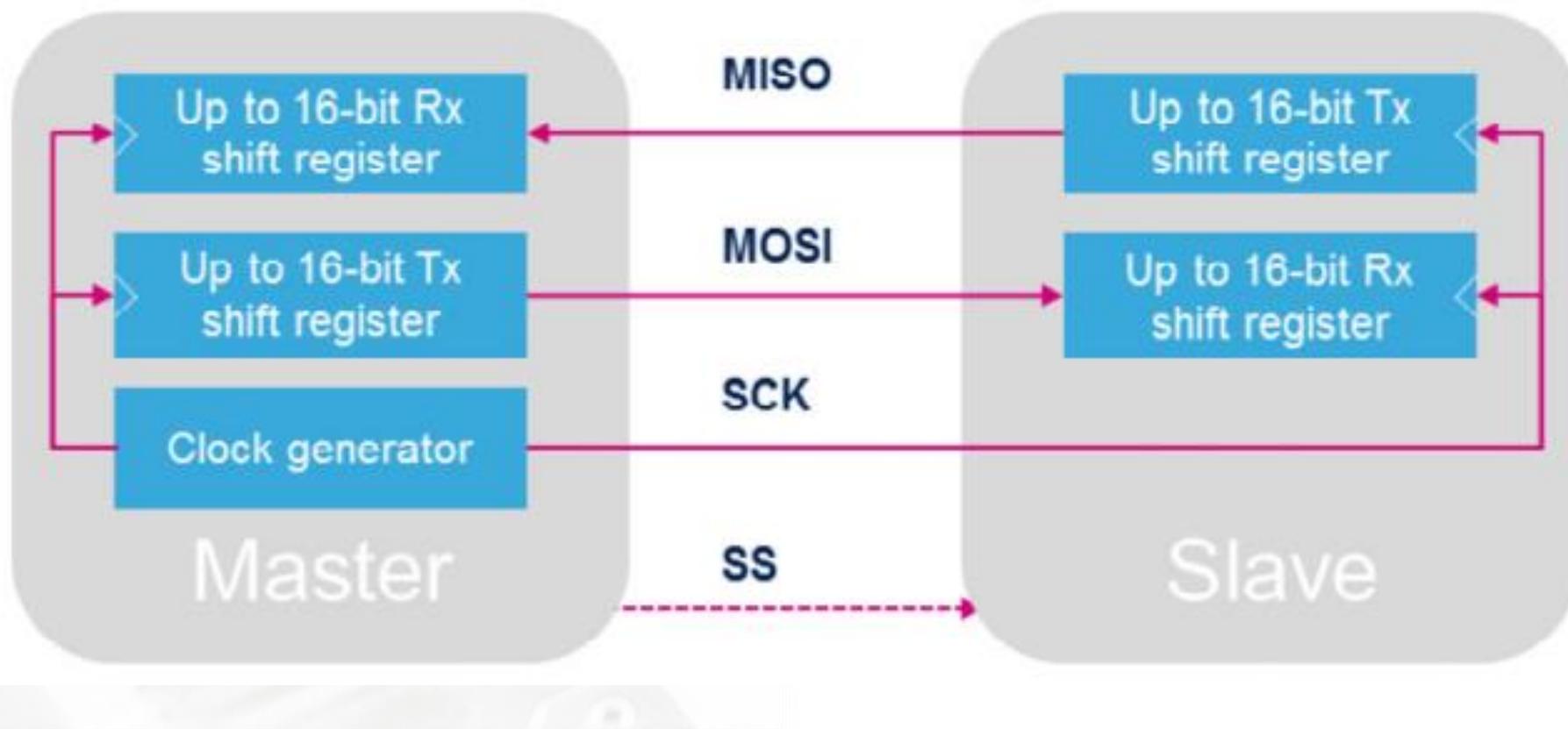
فعدما يقوم الـ slave بإرسال البيانات على خط MISO يقوم الـ Slave باستقبالها ويكون خط الـ MOSI في هذه الحالة للـ Master غير مفعل

أو يقوم الـ Master بإرسال البيانات على خط MOSI ويقوم الـ Slave باستقبالها ويكون خط الـ MISO في هذه الحالة للـ Master غير مفعل



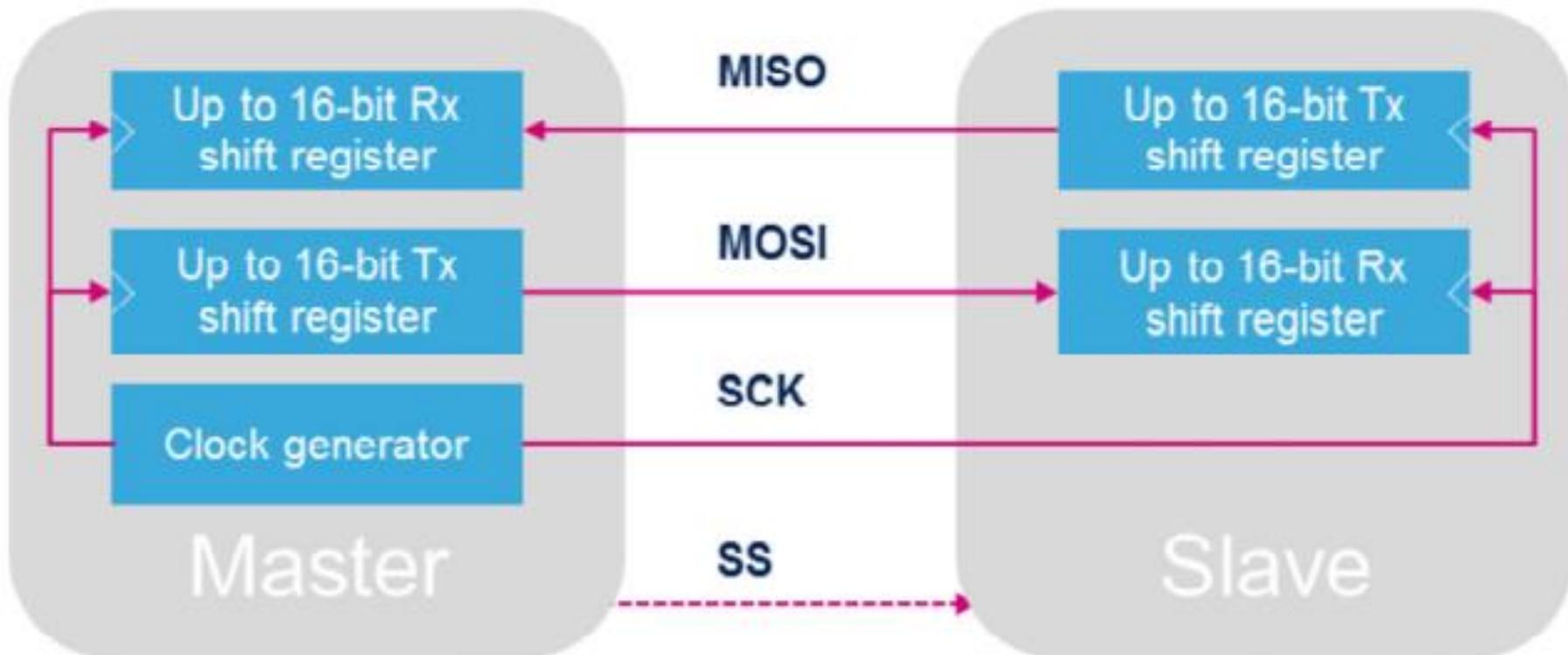
### 3- Full Duplex Mode

في هذا النمط من العمل يمكن لكلٍ من الـ Master والـ Slave أن يقوم بإرسال واستقبال البيانات في نفس الوقت

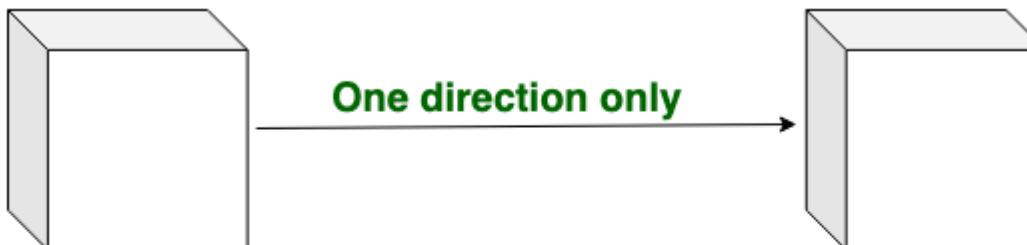


### 3- Full Duplex Mode

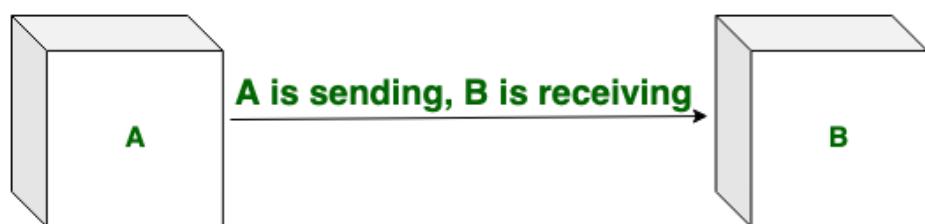
- حيث يتم ارسال البيانات من الـ Master عبر خط الـ MOSI ويقوم الـ Slave باستقبالها
- ويتم ارسال البيانات من الـ Slave عبر خط الـ MISO ويقوم الـ Master باستقبالها.



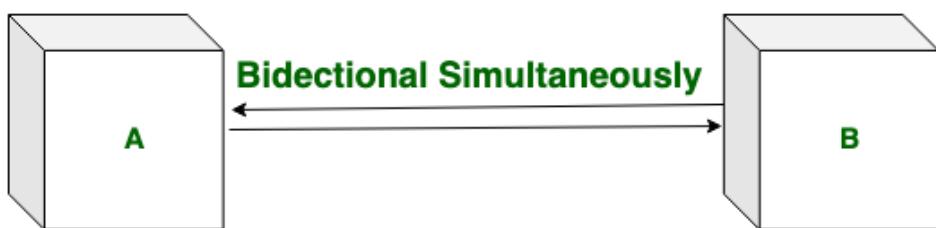
# Simplex & Half-duplex & Full duplex Mode



Simplex mode



A is receiving, B is sending



Full duplex mode

Half duplex mode

# NSS pin

نميز حالتين:

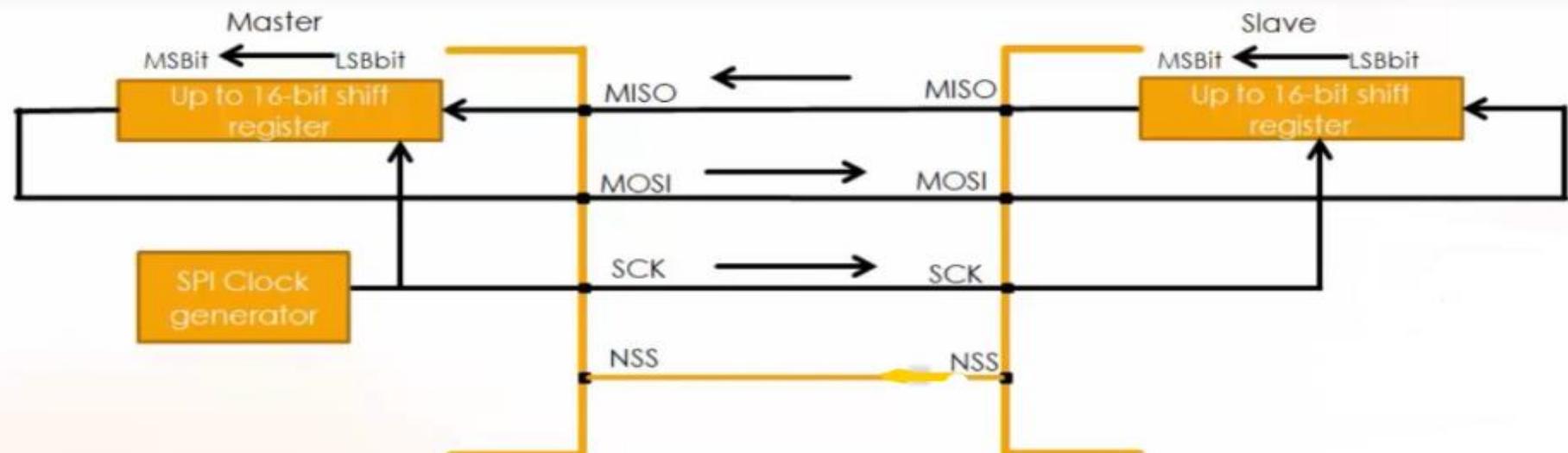
Slave mode  : في هذه الحالة يكون قطب ال NSS قطب دخل و تكون

مهمته chip select أي قطب اختيار الشريحة

Master mode  : في هذه الحالة وعندما يكون هناك جهاز

واحد ، يكون قطب ال NSS قطب خرج و تكون مهمته اختيار ال slave

المراد الاتصال به



# Motorola Mode and Texas Instrument Mode

نميز نمطي عمل لوحة SPI :

**Motorola mode**  : في هذا النمط من العمل يكون بإمكانك التحكم بقطبية وطور إشارة التزامن من خلال التحكم بقيم CPOL , CPOH

**TI mode**  : في هذا النمط من العمل لا يمكنك التحكم بقطبية وطور إشارة التزامن ، حيث يتم وضع الـ CPOL , CPOH بالمستوى

المنخفض LOW

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع الـ Polling

## Transmission

```
HAL_SPI_Transmit(SPI_HandleTypeDef * hspi, uint8_t *  
pData, uint16_t Size, uint32_t Timeout);
```

حيث: pData: مؤشر الى data buffer للبيانات المراد إرسالها، Size: حجم البيانات، Timeout: مدة المهلة قبل فشل الإرسال

## Reception

```
HAL_SPI_Receive(SPI_HandleTypeDef * hspi, uint8_t *  
pData, uint16_t Size, uint32_t Timeout);
```

## Transmit- Receive

```
HAL_SPI_TransmitReceive(SPI_HandleTypeDef * hspi,  
uint8_t * pTxData, uint8_t * pRxData, uint16_t Size, uint32_t  
Timeout);
```

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع الـ Interrupt

Transmission 

```
HAL_SPI_Transmit_IT(SPI_HandleTypeDef * hspi, uint8_t *  
pData, uint16_t Size);
```

- بعد استدعاء هذه الدالة تقوم وحدة SPI ببدء عملية الإرسال لكامل البایتات الموجودة ضمن الـ buffer وعند الانتهاء يتم استدعاء التابع التالي:

```
void HAL_SPI_TxCpltCallback(SPI_HandleTypeDef * hspi)  
{  
    // TX Done .. Do Something ...  
}
```

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع الـ Interrupt

Reception

```
HAL_SPI_Receive_IT(SPI_HandleTypeDef * hspi, uint8_t *  
pData, uint16_t Size);
```

- بعد استدعاء هذه الدالة تقوم وحدة SPI ببدء عملية الاستقبال لكامل البايتات الموجودة ضمن الـ buffer وعند الانتهاء يتم استدعاء التابع التالي:

```
void HAL_SPI_RxCpltCallback(SPI_HandleTypeDef * hspi)  
{  
    // RX Done .. Do Something ...  
}
```

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع الـ Interrupt

## Transmit- Receive

```
HAL_SPI_TransmitReceive_IT(SPI_HandleTypeDef * hspi,  
uint8_t * pTxData, uint8_t * pRxData, uint16_t Size);
```

- بعد استدعاء هذه الدالة تقوم وحدة SPI ببدء عملية الاستقبال لكامل البايتات الموجودة ضمن الـ buffer وعند الانتهاء يتم استدعاء التابع التالي:

```
void HAL_SPI_TxRxCpltCallback(SPI_HandleTypeDef * hspi)  
{  
    // TX-RX Done .. Do Something ...  
}
```

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع DMA

## Transmission

```
HAL_SPI_Transmit_DMA(SPI_HandleTypeDef * hspi, uint8_t  
* pData, uint16_t Size);
```

- بعد استدعاء هذه الدالة تقوم وحدة SPI ببدء عملية الإرسال لـ كامـل الـ باـيـات المـوجـودـة ضـمـنـ الـ buffer وعـنـدـ الـ اـنـتـهـاء يتمـ استـدـعـاءـ التـابـعـ التـالـي:

```
void HAL_SPI_TxCpltCallback(SPI_HandleTypeDef * hspi)  
{  
    // TX Done .. Do Something ...  
}
```

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع الـ DMA

## Reception

```
HAL_SPI_Receive_DMA(SPI_HandleTypeDef * hspi, uint8_t *  
pData, uint16_t Size);
```

- بعد استدعاء هذه الدالة تقوم وحدة SPI ببدء عملية الاستقبال لكامل البايتات الموجودة ضمن الـ buffer وعند الانتهاء يتم استدعاء التابع التالي:

```
void HAL_SPI_RxCpltCallback(SPI_HandleTypeDef * hspi)  
{  
    // RX Done .. Do Something ...  
}
```

# دوال مكتبة HAL المستخدمة ل التعامل مع وحدة SPI في وضع الـ DMA

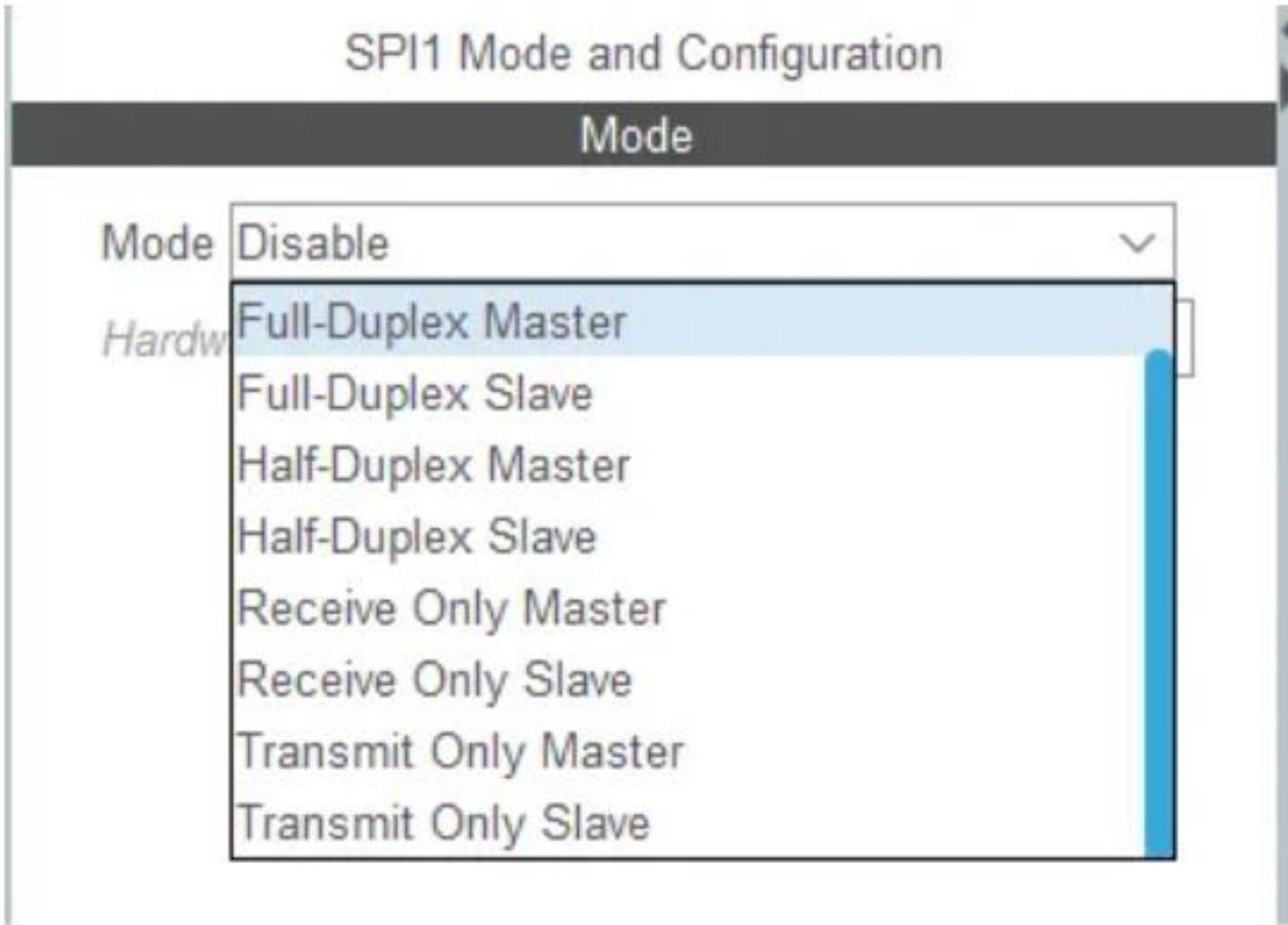
## Reception

```
HAL_SPI_TransmitReceive_DMA(SPI_HandleTypeDef * hspi,  
uint8_t * pTxData, uint8_t * pRxData, uint16_t Size);
```

- بعد استدعاء هذه الدالة تقوم وحدة SPI ببدء عملية الاستقبال ل كامل البايتات الموجودة ضمن الـ buffer و عند الانتهاء يتم استدعاء التابع التالي:

```
void HAL_SPI_TxRxCpltCallback(SPI_HandleTypeDef * hspi)  
{  
    // TX-RX Done .. Do Something ...  
}
```

## - تحديد نمط عمل وحدة SPI:



# ضبط إعدادات وحدة SPI في متحكمات STM32

ضبط بارامترات وحدة SPI وتتضمن:

ضبط إعدادات الوحدة لتعمل بنمط Motorola وضبط كل من حجم البيانات وتحديد بت البداية بالإضافة إلى تحديد قطبية وطور إشارة التزامن من خلال التحكم بقيم CPOL , CPOH ، بالإضافة إلى ضبط قيمة البت :NSS

Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Configure the below parameters :

Search (Ctrl+F)

Basic Parameters

- Frame Format: Motorola
- Data Size: 8 Bits
- \* First Bit: MSB First

Clock Parameters

- Prescaler (for Baud Rate): 2
- Baud Rate: 8.0 MBits/s
- \* Clock Polarity (CPOL): Low
- \* Clock Phase (CPHA): 1 Edge

Advanced Parameters

- CRC Calculation: Disabled
- \* NSSP Mode: Enabled

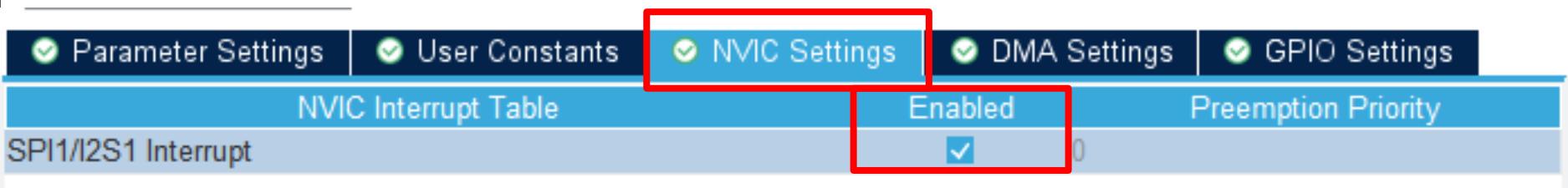
NSS Signal Type: Output Hardware

# ضبط إعدادات وحدة SPI في متحكمات STM32

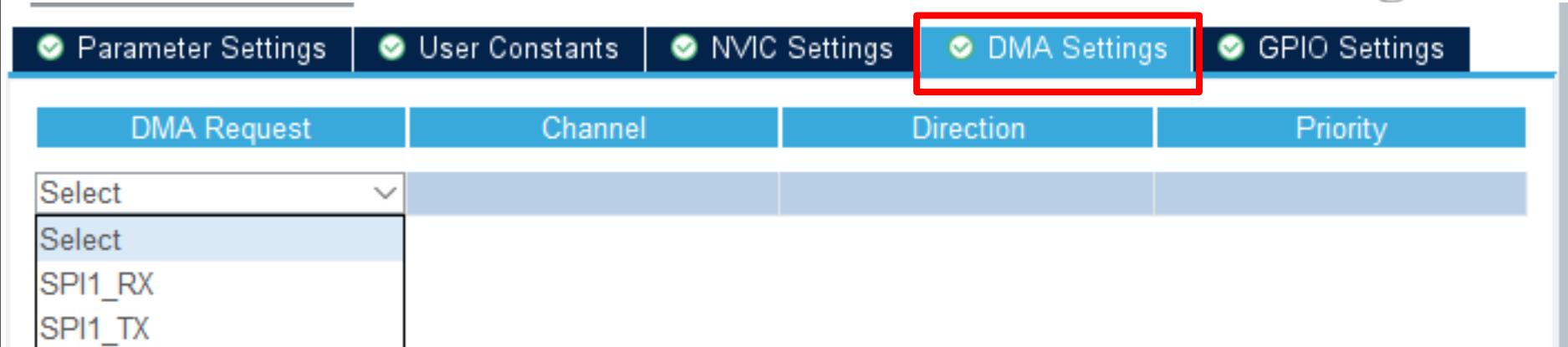
ضبط بارامترات وحدة SPI وتتضمن:

كما يمكن تفعيل المقاطعة الخاصة بوحدة الـ SPI من خلال تاب

## NVIC Settings



- كما يمكن إضافة طلب DMA لوحدة الـ SPI من خلال تاب **Settings**



**التطبيق العملي : المطلوب ربط مجموعة متحكمات Stm32G0B1CE من خلال النافذة التسلسية SPI أحدها Master والباقي Slave.**

Thank you for listening