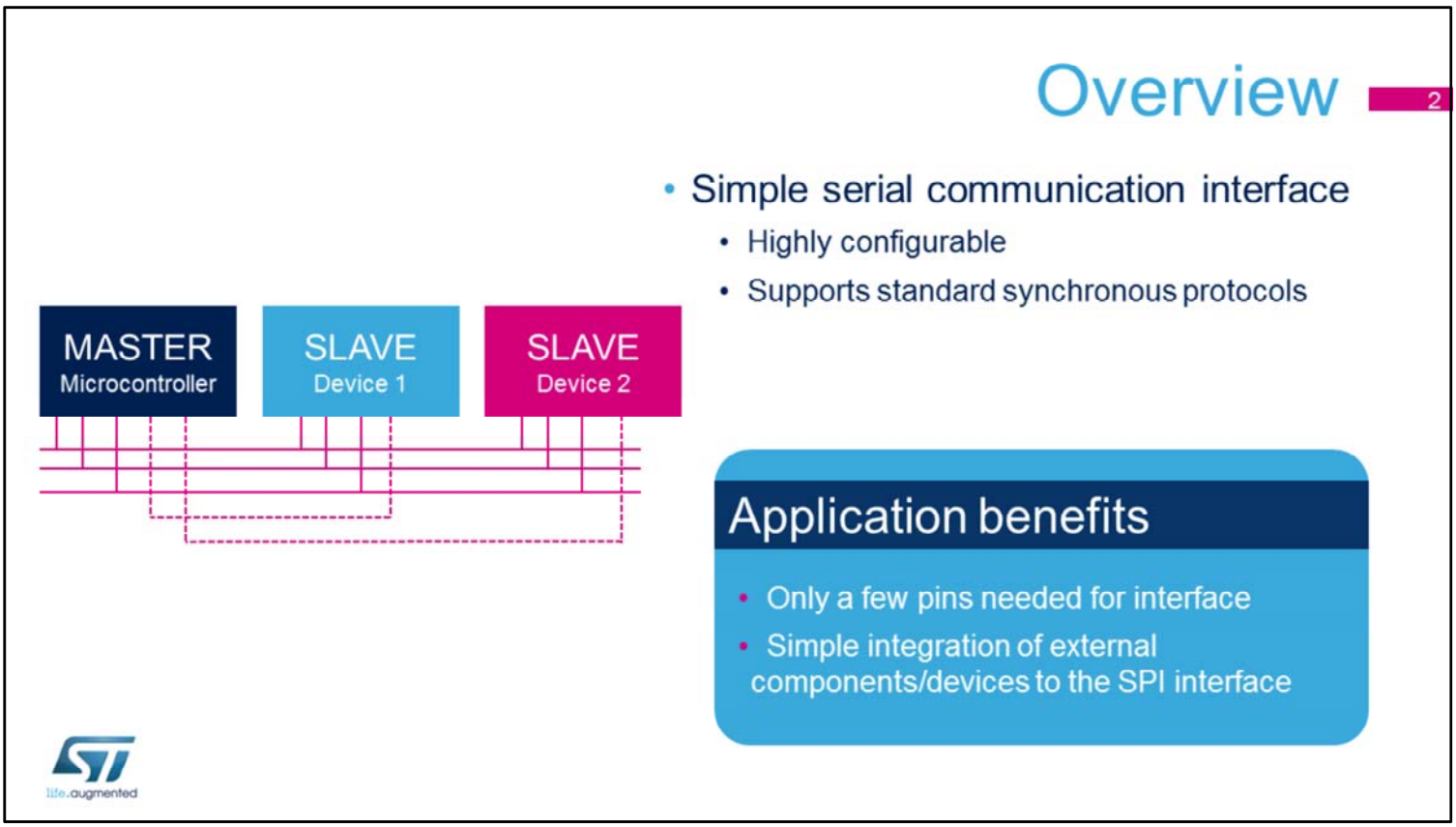
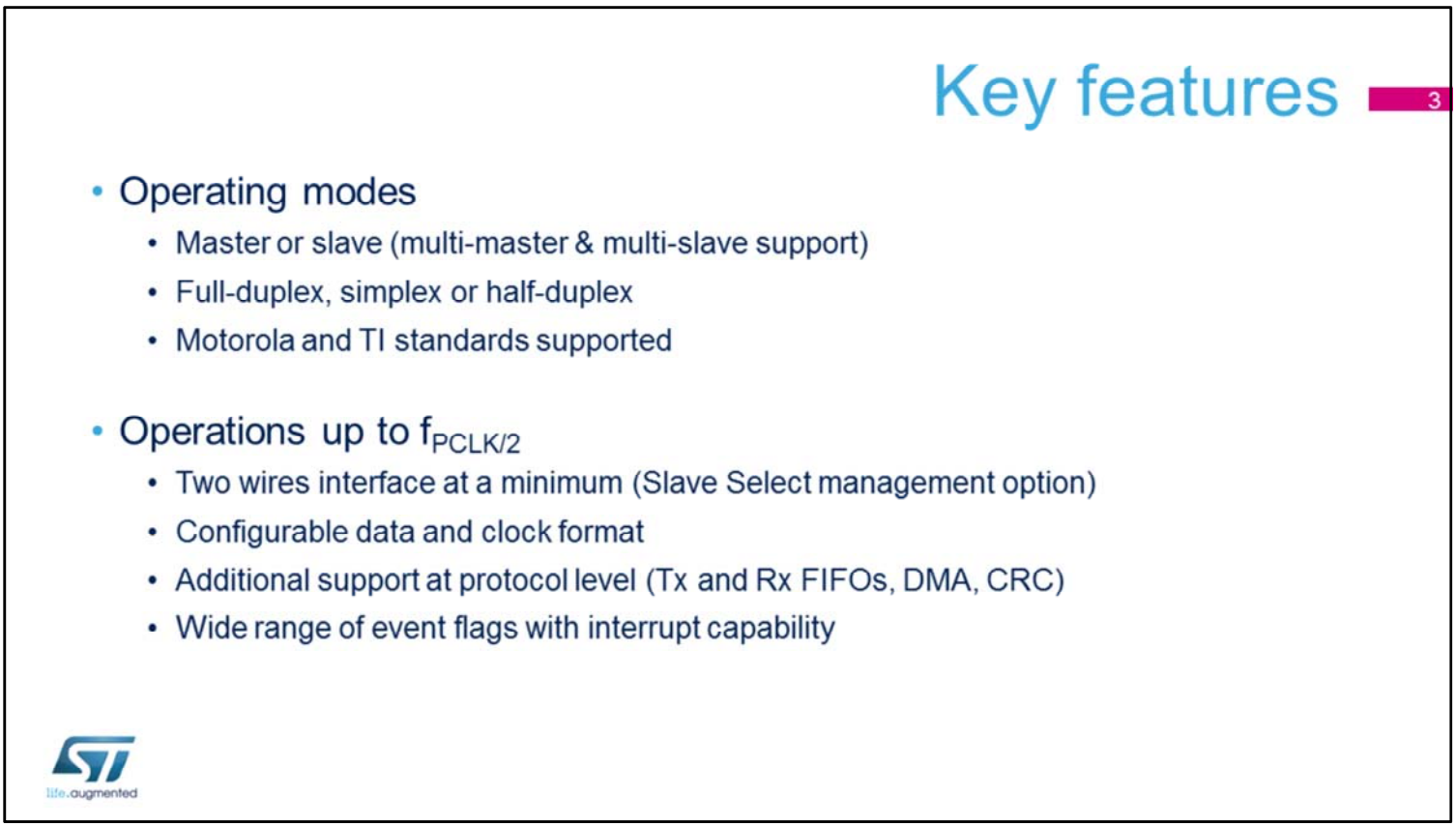
Hello, and welcome to this presentation of the STM32 Serial Peripheral Interface.

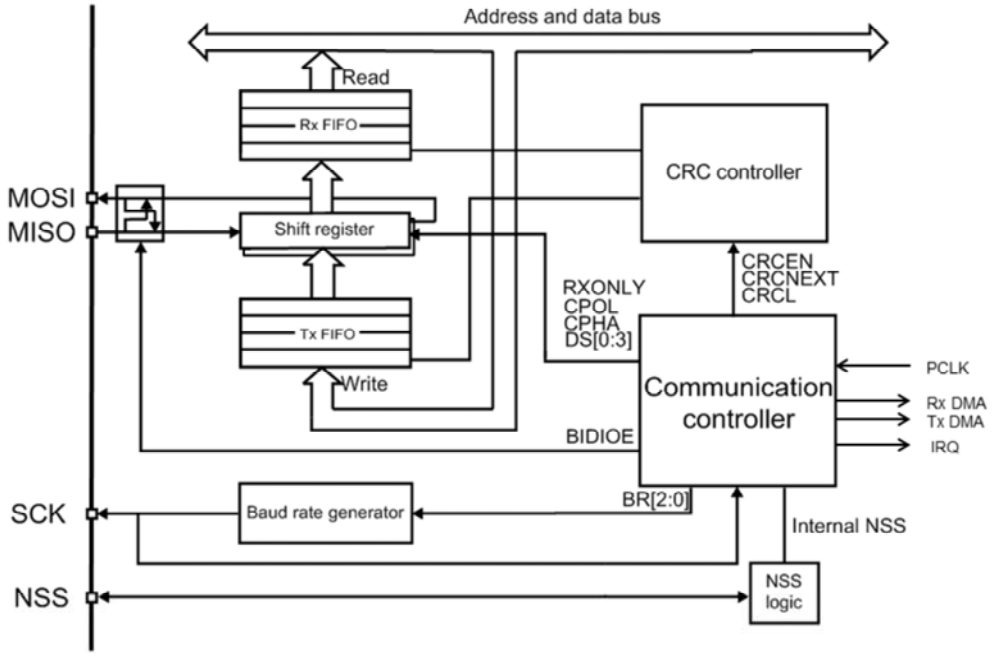


The internal Standard Peripheral Interface or SPI provides simple communication interface allowing the microcontroller to communicate with external devices. This interface is highly configurable to support many standard protocols. Applications benefit from the simple and direct connection to components which requires a few pins. Thanks to the highly configuration capabilities of the SPI, many devices can be simply accommodated in the existing project.



The STM32 SPI offers various operating modes that are explained in more detail in this presentation. The communication speed can't exceed half of the internal bus frequency, and a minimum of two wires is required to provide the serial data flow synchronized by clock signal in a single direction. An optional hardware slave select control signal can be added. The data size and transmit shift order are configurable, as well as the clock signal polarity and phase. At the protocol level, the user can use specific data buffers with an optional automatic cyclic redundancy check or CRC calculation, and transfers through the DMA controller. There are a wide range of SPI events that can generate interrupt requests.

Block diagram



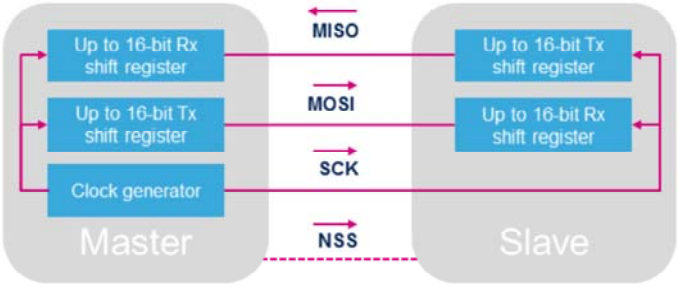
The simplified SPI block diagram shows the basic control mechanisms and functions. There are 4 I/O signals associated with the SPI peripheral. All of the data passes through receive and transmit buffers via their specific interfaces. The control block features are enabled or disabled depending on the configuration.

Interconnection of SPI nodes

5

Various master - slave interconnections are supported

- Master always provides clock and controls all the traffic (selects slave for communication)
- Data can be exchanged in both directions in parallel



- In Full-duplex mode (bidirectional), both master and slave transmit and receive data at the same time

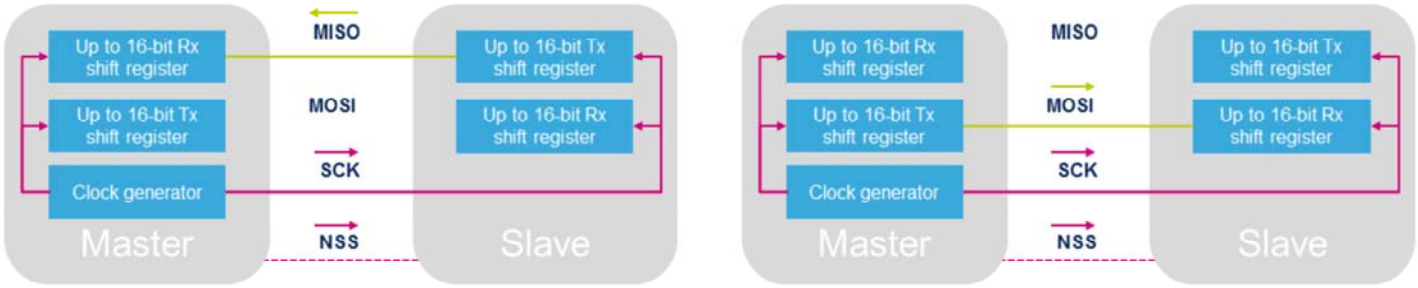
life.augmented

The SPI master always controls the bus traffic and provides the clock signal to the dedicated slave through the SCK line. The master can select the slave it wants to communicate with through the optional Slave Select or NSS signal. Data stored in the dedicated shift registers can be exchanged synchronously between the master and slave through the MOSI (Master Output, Slave Input) and the MISO (Master Input, Slave Output) data lines. In Full-duplex mode, both data lines are used and synchronous data flows in both directions.

Interconnection of SPI nodes

Various master - slave interconnections are supported

- In Simplex mode (unidirectional), one node is a transmitter while the other is the receiver



life.augmented

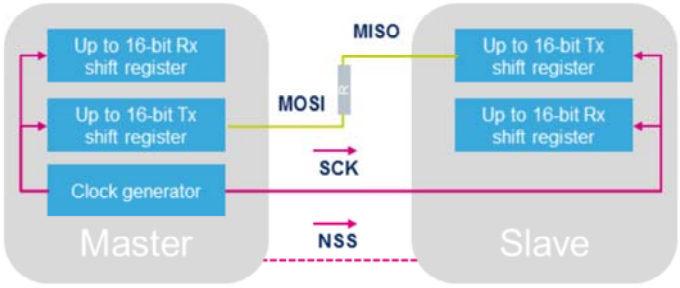
In Simplex mode, one node transmits data while the other receives the data. Data only flows in one direction. Depending on the communication direction, only one data line is used. Unused SPI pins can be used for other purposes.

Interconnection of SPI nodes

7

Various master - slave interconnections are supported

- In Half-duplex mode (quasi-bidirectional), both master and slave alternate the data transmission and reception synchronously. The nodes share the single common data line.

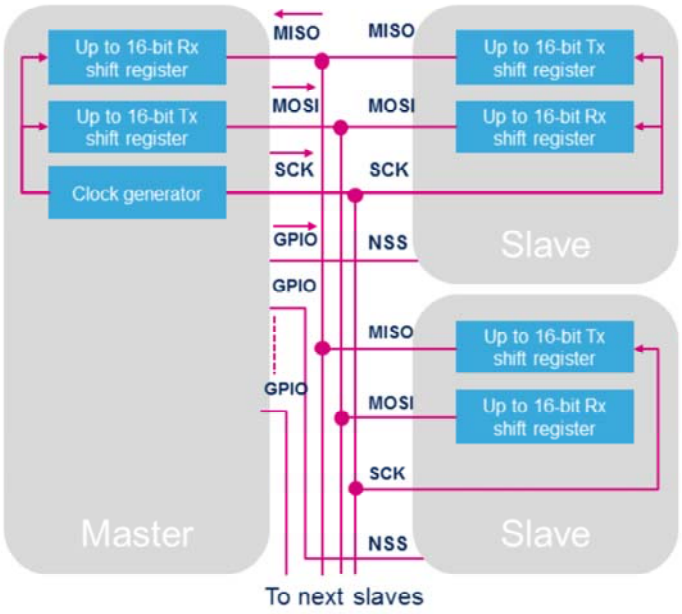


Half-duplex mode integrates the previous two modes with sharing a single line for data exchanges and data flow in a single direction at time. There is a cross connection between the master MOSI and the slave MISO pins in this mode. The master and slave have to alternate their transmitter and receiver roles synchronously when having a common data line. It is common to add a serial resistor on the half duplex data line to prevent possible temporary short-circuit connection, since master and slave nodes are not usually synchronized.

Interconnection of SPI nodes

Multi-slave net topologies support

- Multi-slave: Star topology
  - Master selects a single slave node when writing/reading data
  - Separated Slave Select signals (simulated by GPIO pins) are required
  - Slave nodes can have different clock and data formats

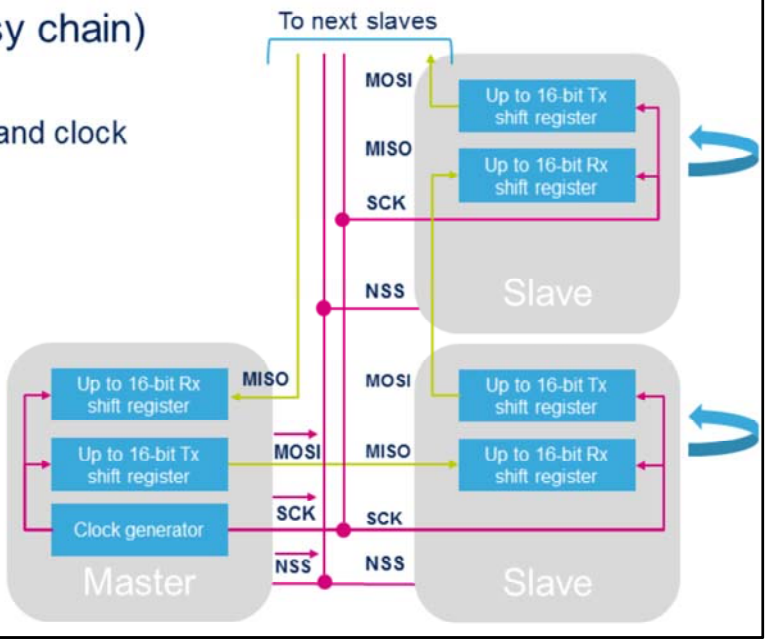


When the SPI network includes more than one slave, a star topology is commonly used. The master communicates with one slave at a time, since you can only have one slave transmit data back to the master through the common MISO pin. In this topology, a separated Slave Select signal from the master has to be provided to each slave node, so the master can select which slave to communicate with. Thanks to separate Slave Select signals (NSS), SPI data and clock format can be adapted for each slave, if the multiple slave nodes do not have a common configuration.

Interconnection of SPI nodes

Multi-slave net topologies support

- Multi-slave: Circular topology (daisy chain)
  - Data circulates through all the nodes
  - All nodes must support a common data and clock format

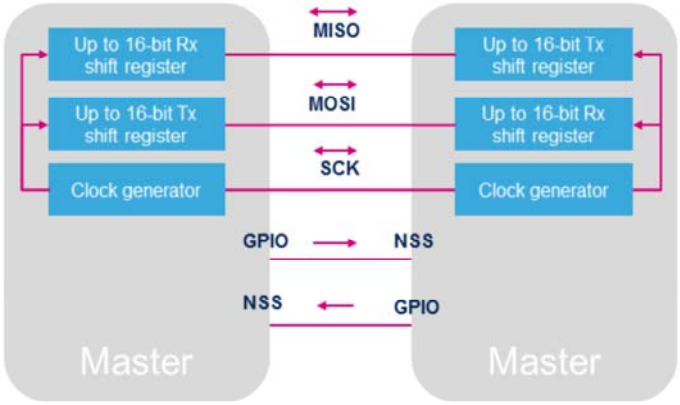


Another multi-slave configuration is the circular topology where the inputs and outputs of all the nodes are connected together in a closed serial chain. A common Slave Select signal is used for all the nodes as communication occurs at the same time. All nodes must have the same data and clock format configuration. Microcontroller SPI nodes typically use separate internal transmit and receive shift registers, so the data transferred between them has to be handled by software in a circular mode.

Interconnection of SPI nodes

Multi-master topology support

- Multi-master: Two nodes with mastering capability
  - Nodes are in Slave mode by default
  - A node switches itself to active master to take control of the bus to start a communication session
  - Slave Select pin is used as input to detect potential bus conflicts
  - The master node returns to Slave mode to end a communication session

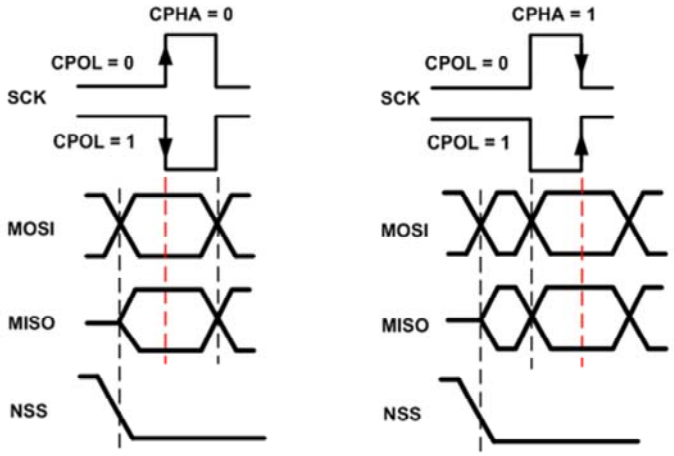


SPI networks can operate in a multi-master environment. This mode is used to connect two master nodes exclusively. When either node is not active, they are by default in a slave mode. When one node wants to take control of the bus, it switches itself into Master mode and asserts the Slave Select signal on the other node through a GPIO pin. Both Slave Select NSS pins work as a hardware input to detect potential bus collisions between nodes as only one can master the SPI bus at a single time. After the session is done, the active node master releases the Slave Select signal and returns back to passive slave mode waiting for the next session start.

Data frame format

Fully programmable and flexible format

- Size of data frame
  - From 4 up to 16 bits
- Shift order of bits
  - MSB or LSB first
- Clock setting (mode 0-3)
  - Low or high polarity at idle
  - Sampling by odd or even edges

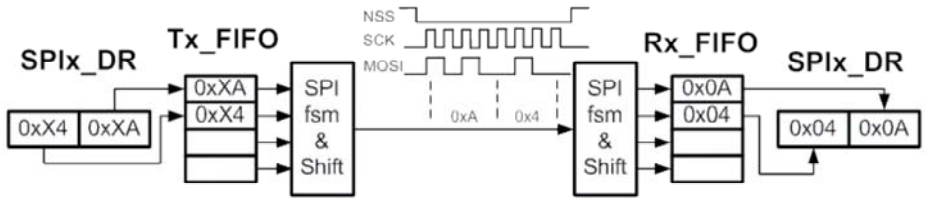


There are a few controls that are used to setup the data format. Users can define the data frame size and the transmit order of the shift register. The clock can be set to one of four basic configurations defined in the Motorola SPI specifications. The combination of two bits controls the polarity and phase of the clock signal. When the phase control bit is cleared, data bits are sampled on the odd clock edges, and the even clock edges synchronize the shifting of the next bit onto the data line. This is the opposite when the phase control bit is set. The clock polarity bit defines the idle state of the clock signal and which clock edge is used for data sampling or shifting.

Data packing, FIFO access

Advanced low demand control

- Packing mode
  - Access of FIFO registers by multiply data patterns
  - Configurable FIFO threshold levels
  - DMA access
  - Number of events and required services are decreased
  - System load is reduced



16-bit write access to data register at TxE event:

SPI_DR= 0xX4XA;

16-bit read access from data register at RxNE event based on 16-bit threshold:

uint16_t var;

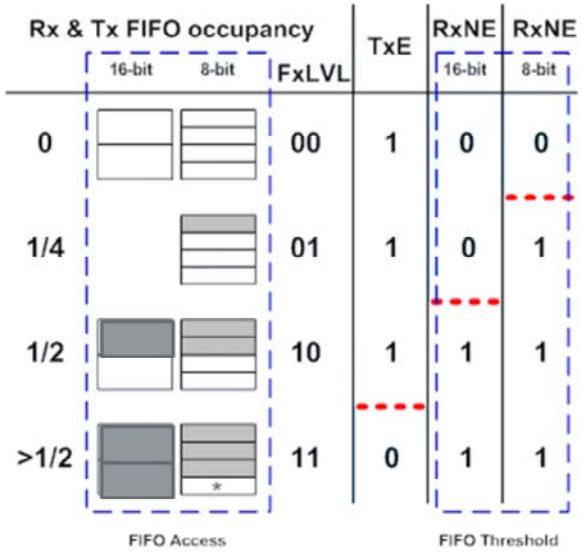
var= SPI_DR; / var= 0x040A */*

When the communication speed is fast and data frames short, it could be a demanding task to ensure a correct data flow when the clock signal becomes continuous and the Full-duplex mode is used. Slave nodes have to properly service all the transactions sent by the master properly to prevent any data overrun or underrun conditions. When the data frame size fits into a byte, packing mode can be used. Then multiple data patterns can be written or read in a single access to the FIFO registers. Together with the proper setting of the FIFO threshold event, the number of events to service will decrease to better control the data flow. When the DMA controller is also used, overall loading on the system is significantly reduced. This figure shows how two short four-bit data frames can be written and read by a single 16-bit access in the dedicated FIFO registers. The read or write data access requires just a single service event.

32-bit Rx and Tx FIFOs

Balance between threshold and access of data

- Two separate 32-bit FIFOs for transmission and reception
- 8/16-bit read/write access vs. FIFO threshold and occupancy flags
- Different capability of Tx and Rx FIFOs at 8-bit access



*) Max 3x 8-bit for TxFIFO, 4x 8-bit for RxFIFO

SPI peripheral features two 32-bit FIFOs to handle the data flow. The FIFOs can be accessed by using either 8-bit or 16-bit data instructions. During reception, the events generated from the FIFO depend on threshold setting (RxNE). The table gives an overview how the event flag behavior changes depending on the configuration. It is important to keep the FIFO access balanced with the threshold setting so that the data consistency is not lost. During transmission, the Transmit FIFO occupancy (TxLVL) depends on data access.

There is a specific event behavior when the Transmit FIFO occupancy is higher than half.
During transmission, the Transmit FIFO occupancy (FTLVL) depends on data access when the FIFO level becomes higher then half. When 8-bit access is used to store data to upper half of the FIFO, the FIFO status becomes full (FTLVL=11 and TXE=0). System doesn't

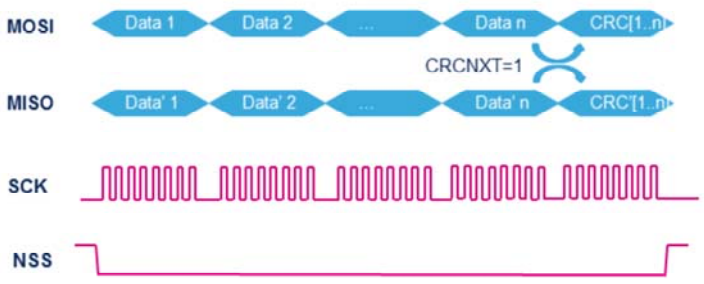
accept any new write attempt to the FIFO though there is still room available for a one more 8-bit data. This prevents possible incorrect 16-bit access and overflow of the space available for data at this situation. So the FIFO full capability can never be achieved when 8-bit data access is applied for data transmission.

Additional support at protocol level

14

Enhanced DMA and CRC management

- DMA controller automatically handles
  - Data transmit and receive events
  - CRC at the end of the transaction
  - FIFO threshold control



- CRC
  - Separated calculators for receive and transmit flows
  - CRC pattern is sent at the end of each transaction:
    - Transmitter puts the CRC result directly into the data shift register
    - Receiver stores the CRC in the Rx FIFO and compares the value with the internal calculation
  - Programmable CRC polynomial (odd values only) and CRC length (8- or 16-bit CRC frame)

life.augmented

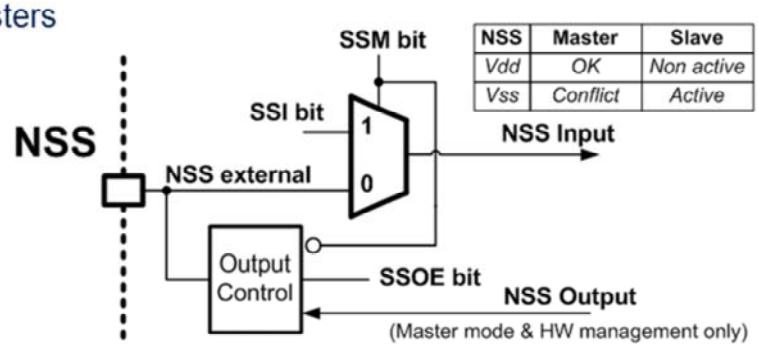
During protocol level communication, the DMA controller can be used to handle the data flow events, the CRC calculations, and the updating of the FIFO threshold automatically. In case of threshold control, the last odd data frame is correctly applied in packed mode when the number of frames is not aligned with the packet size. If the CRC is enabled, separated CRC calculators are used for the transmitter and receiver. The CRC calculation result is applied at the end of each transfer automatically by the DMA controller or by software control. Results from the transmitter CRC calculator register are loaded directly into the shift register, and the received CRC value is stored in the FIFO and compared with the receiver CRC result. The CRC polynomial used for the calculation is programmable, and the length of the CRC pattern can be set to either 8- or 16-bit frames.

Standard NSS modes

15

Enhanced management of slave select signal (NSS)

- NSS input
  - Hardware or software management
  - Slave mode – select active slave
  - Master mode – conflict between masters
- NSS output
  - Master mode
    - Select active slave
    - Specific modes



NSS	Master	Slave
Vdd	*OK*	*Non active*
Vss	*Conflict*	*Active*

Slave Select signal is commonly used by the master node to select the slave node for communication.
Slave Select signal is commonly used by the master node to select just one slave node for communication. The signal implementation is mandatory in multi-master and multi-slave topologies except for some very rare and specific cases (commonly, when any collision on MISO data line is prevented). Though it is not mandatory at a single master-slave pair, it could be helpful for data flow synchronization at whatever topology case.

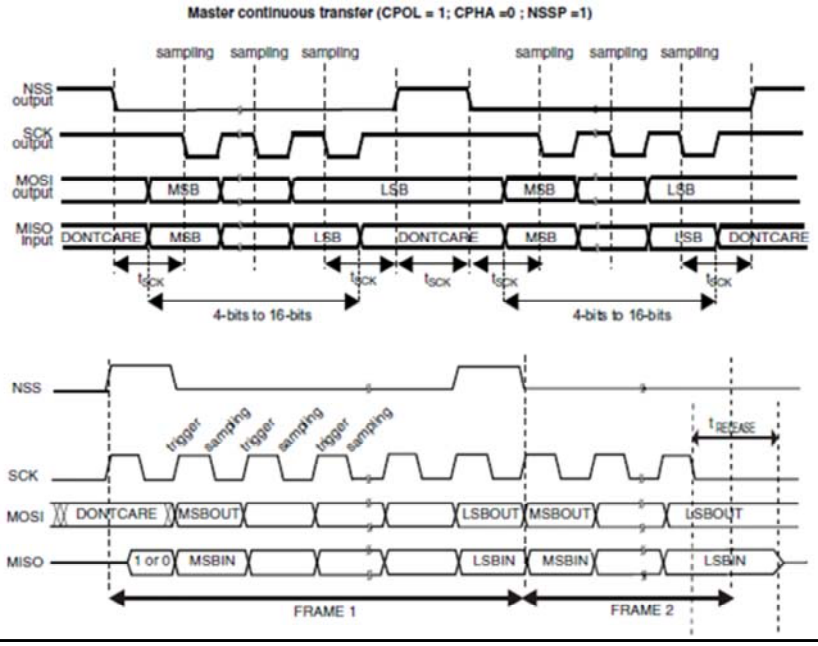
The Slave Select signal can operate as an input or as an output. The NSS input can be managed by hardware or software depending on the SSM bit, in either Master or Slave modes. As a slave input, it is used to identify itself as the active slave for communication. As a master input, it signalizes a potential conflict between masters in a multi-

master system. The NSS working as an output is only used in Master mode and managed by hardware in a standard or specific control mode. Additional slave select outputs can be provided by the GPIOs under software control.

Specific NSS modes

Enhancement modes with hardware control of Slave Select signal (NSS)

- NSS pulse mode
  - Master supported only
  - Motorola mode (CPHA 0 only)
- TI mode
  - Master and slave support
  - Fixed CPOL and CPHA setting
  - HiZ slave's MISO automatic control



There are a few enhanced modes when Slave Select signal is under specific hardware control. The Slave Select signal can operate in a pulse mode where the master generates pulses on NSS output signal between data frames for a duration of one SPI clock period when there is a continuous transfer of data. The data is then interleaved by two SPI clock periods. The clock phase is fixed in this mode. Another enhanced mode is the TI mode where the data flow is synchronized by the NSS pulses, provided by the master, on the last bit of data. The clock polarity and phase configuration is fixed and the slave data output is automatically switched into high impedance when the bus traffic stops and on a specific configurable timeout.

Interrupts and DMA

Interrupt event	Description
Transmit FIFO ready	Set when Tx FIFO is ready to accept new data.
Receive FIFO ready	Set when data is received in the Rx FIFO.
Master mode fault	Set when there is a bus conflict detected in the multi-master bus configuration.
Data overrun error	Receiver can't accept next data flow as the Rx FIFO is full.
TI frame format error	NSS signal doesn't correspond to the data format.
CRC protocol error	Checksum of the received message doesn't match the one calculated internally.

- DMA requests can be generated in Indirect mode when FIFO threshold is reached

life.augmented

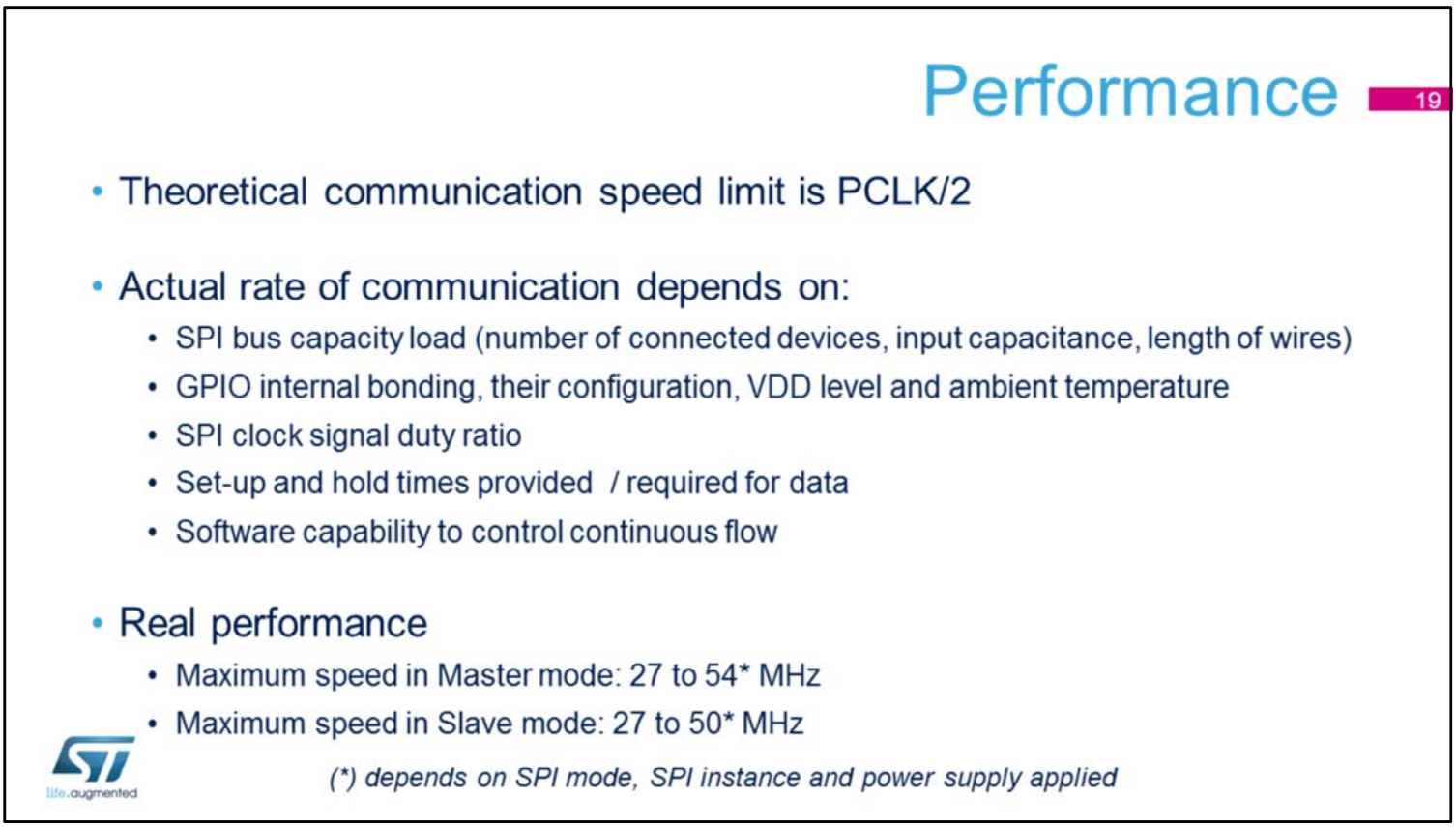
Here is an overview of the SPI interrupt events. There are FIFO and error detection events to handle data flows. DMA requests are triggered internally by FIFO threshold events.

Low-power modes

Mode	Description
Run	Active.
Sleep	Active. Peripheral interrupts cause the device to exit Sleep mode.
Stop	Frozen. Peripheral registers content is retained.
Standby	Powered-down. Peripheral must be reinitialized after exiting Standby mode.

life.augmented

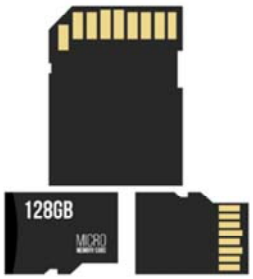
Here is an overview of the SPI status in specific low-power modes. The device is not able to perform any communication in Stop or Standby modes. It is important to ensure that all the SPI traffic is complete before the peripheral enters Stop or Powered-down modes.



The SPI performance depends mainly on the applied clock. At a minimum, the clock frequency should be twice the required communication frequency. The actual rate of communication can be decreased by application factors. The user has to consider SPI bus loads such as the number of nodes, the connection distance, the input capacitance, as well as the GPIO settings. Fast GPIO mode should be applied on the data and clock signals. Lower power supply voltage and extreme ambient temperatures slow down edges. Sometimes slower data hold or setup time requirements have to be respected between nodes. Applications can't always manage the fast data flow due to frequent servicing of exceptions.

Application examples

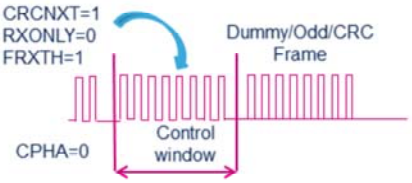
- Displays
- Smart sensors
- Memories
- MMC/SD cards
- IO expanders



The SPI can be used in a wide range of applications where a simple data transfer is required without the need for a complex communication protocol. Secured transfers are also supported when used with smart cards.

Application tips and tricks

- Common tips:
  - Before disabling the SPI (or its clock), check if it is busy and its FIFO status
  - Use DMA controller when a specific control is required (handle CRC, change of RxFIFO threshold, terminate Receive-only mode (RXONLY=1)
  - Packed mode should be used when data fits into a byte
  - Hardware management of NSS brings a benefit



- Specific Aspects:
  - CRC information is loaded into the receive FIFO, user has to flush it
  - BSY behavior is different, in Master and Slave modes, during continuous data transaction
  - Receive and Overrun flags are set during transmit operations (these should be ignored)
  - DMA tips - number of data to be configured for a channel when CRC is applied
    - For any transmission and reception in Receive-only - number of data excluding the CRC length
    - For reception in Full-duplex - number of data including the CRC length

life.augmented

There is a comprehensive summary of basic common tips, more details user can find at product reference manual:

User should follow specific procedures checking if any traffic is still running out on the bus before low power mode is applied or the peripheral clock is removed to prevent any premature terminating of such a transaction flow. It can be ongoing for a sure time after the DMA channel signalizes transaction complete status or the transmit FIFO becomes empty.

The DMA controller should be used to terminate a transaction when a specific control has to be applied which handles CRC, or receive FIFO threshold, or at Receive-only mode. Such a control have to be applied at a short time window available within the last but one data

frame transaction exclusively. This ensure transaction of correct amount of data.

When DMA and/or data packet frames are applied, the number of services required to handle the data flow decreases significantly so the system overall performance grows up. This is an effective approach at case when data frames are short, the bus communication speed is fast and the data flow is continuous especially.

Hardware management of NSS pin is not quite necessary when a master communicates with a single slave, but it can help to synchronize the data flow between them.

There are some additional specific aspects which should be taken into account when designing a SPI network.
The receiver always loads CRC information into the receive FIFO. User has to take into account an additional space needed in the FIFO and take care of the CRC information flushing from it.
The Busy flag should not be used for any data handling but to check the bus traffic stop when terminating a transaction. The BSY bit is not cleared between data frames during the master continuous data transactions. It always goes low for at least one SPI clock cycle between data frames in slave mode, no matter if the communication is continuous or not.
When the node transmits data only, the receive chain stays always active. Users should ignore all the associated receive and overrun events in this case.
When including the CRC, the number of data to be processed by the DMA controller depends on the transfer mode. The number has to be set differently for transmitter

and receiver in a Full-duplex mode, specific setting has to be applied at Receive-only mode.

STM32F7 instances features

SPI features	SPI1	SPI2	SPI3	SPI4	SPI5	SPI6
Hardware CRC calculation	Yes	Yes	Yes	Yes	Yes	Yes
Rx & Tx FIFOs	Yes	Yes	Yes	Yes	Yes	Yes
NSS pulse mode	Yes	Yes	Yes	Yes	Yes	Yes
TI mode	Yes	Yes	Yes	Yes	Yes	Yes

life.augmented

There are six SPI instances within the STM32F7, and each support all the features presented so far. SPI1, SPI2 and SPI3 are multiplexed with I2S interface.

References

- For more details, please refer to following resources
  - AN4286 - SPI protocol used in the STM32 bootloader
  - AN3364 - Migration and compatibility guidelines for STM32 microcontroller applications
  - Web (connection examples, available monitoring tools)

life.augmented

There are some dedicated SPI application notes. To learn more about general SPI connections and interface issues, there are many web pages, as well as SPI bus monitoring tools available. Many digital oscilloscopes support direct reading and analysis of data and clock signals on the SPI bus.

SPI related peripherals

- Refer to these other peripherals:
  - RCC (SPI clock enable, Clock Control in Sleep, Reset)
  - Interrupts (FIFO and Error events)
  - GPIO (Speed control, GPIO configuration)

life.augmented

Refer to these other trainings which are linked directly to the SPI. Users should be familiar with all the peripherals that can affect the behavior of the SPI.