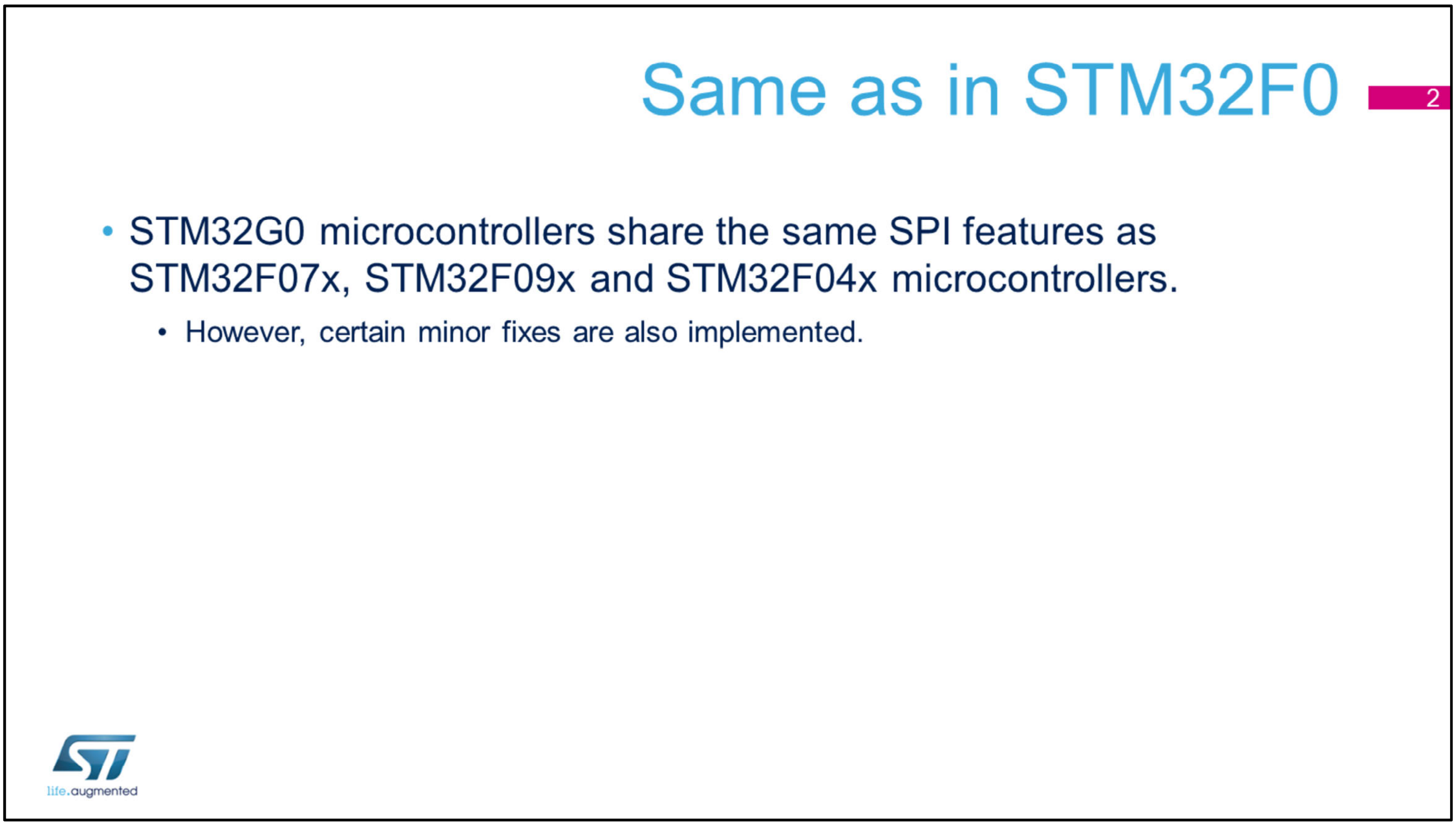
STM32G0 - SPI

Serial Peripheral Interface

Revision 1.0

life.augmented

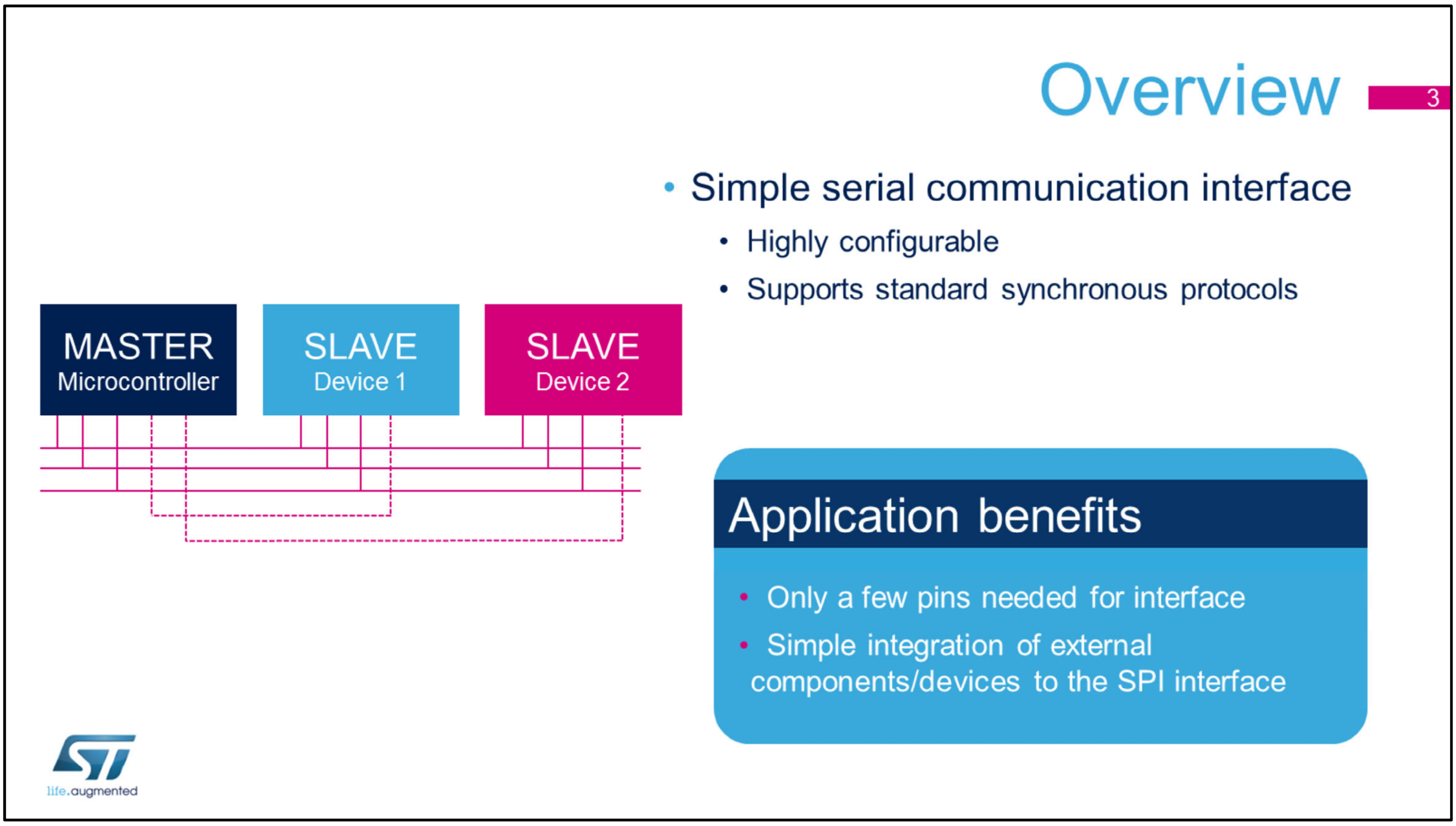
Hello, and welcome to this presentation of the STM32 Serial Peripheral Interface.



Same as in STM32F0

- STM32G0 microcontrollers share the same SPI features as STM32F07x, STM32F09x and STM32F04x microcontrollers.
  - However, certain minor fixes are also implemented.

The SPI drivers developed for STM32F0 microcontrollers work seamlessly with STM32G0 microcontrollers.



Overview

- Simple serial communication interface
  - Highly configurable
  - Supports standard synchronous protocols

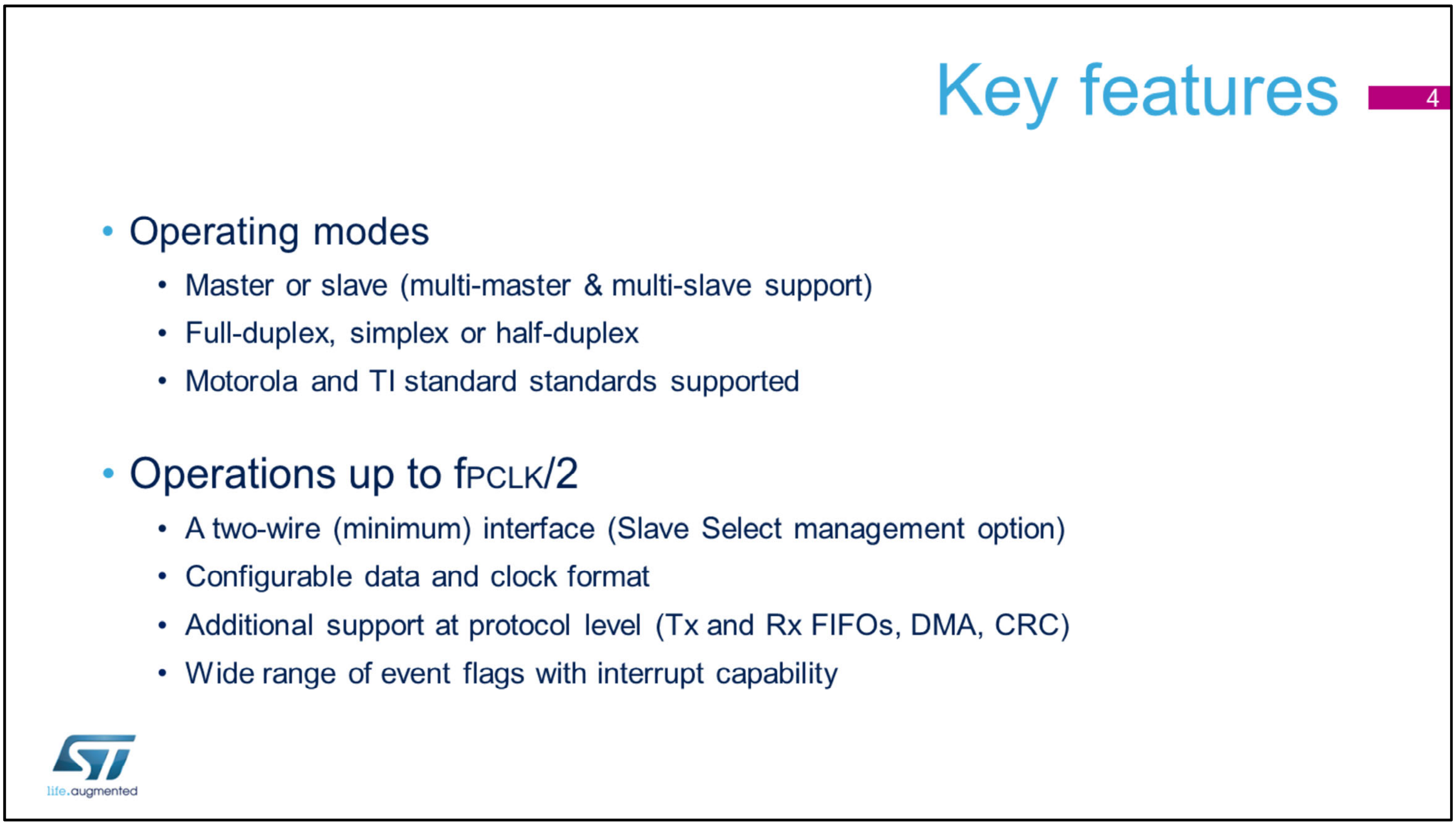
MASTER
Microcontroller
SLAVE
Device 1
SLAVE
Device 2
life.augmented

Application benefits

- Only a few pins needed for interface
- Simple integration of external components/devices to the SPI interface

The internal Serial Peripheral Interface or SPI provides a 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 only requires a few pins. Thanks to the highly configurable capabilities of the SPI, many devices can be simply accommodated in the existing project.



Key features

4

- Operating modes
  - Master or slave (multi-master & multi-slave support)
  - Full-duplex, simplex or half-duplex
  - Motorola and TI standard standards supported
- Operations up to $f_{PCLK}/2$
  - A two-wire (minimum) interface (Slave Select management option)
  - Configurable data and clock format
  - Additional support at protocol level (Tx and Rx FIFOs, DMA, CRC)
  - Wide range of event flags with interrupt capability

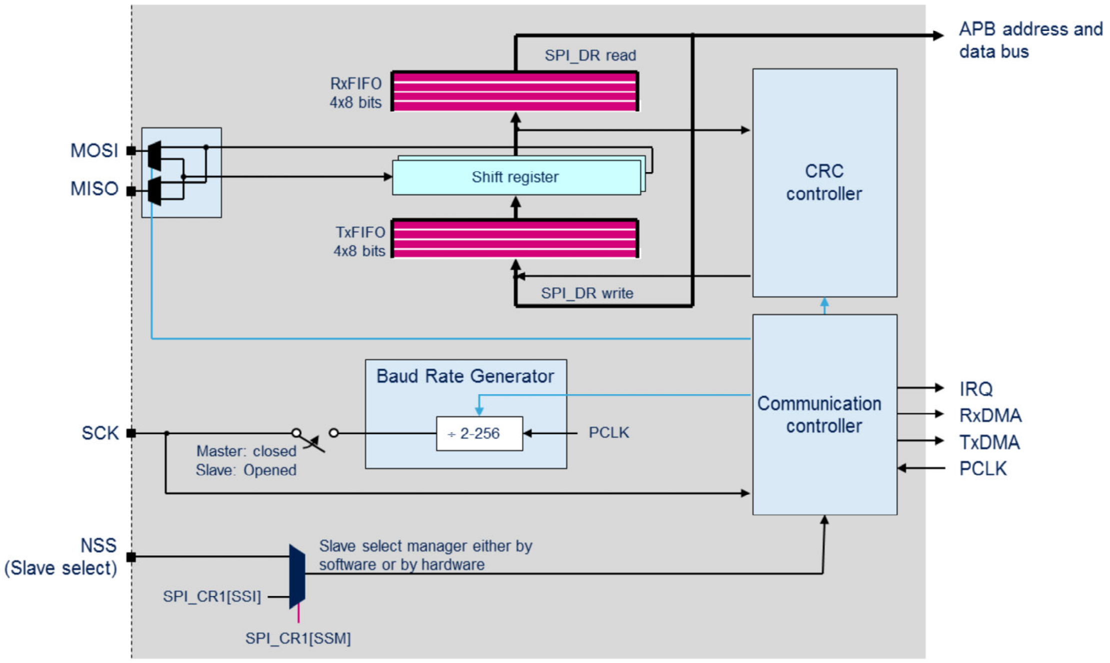
life.augmented

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 or polarity and timing adjustment of the slave select signal.

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

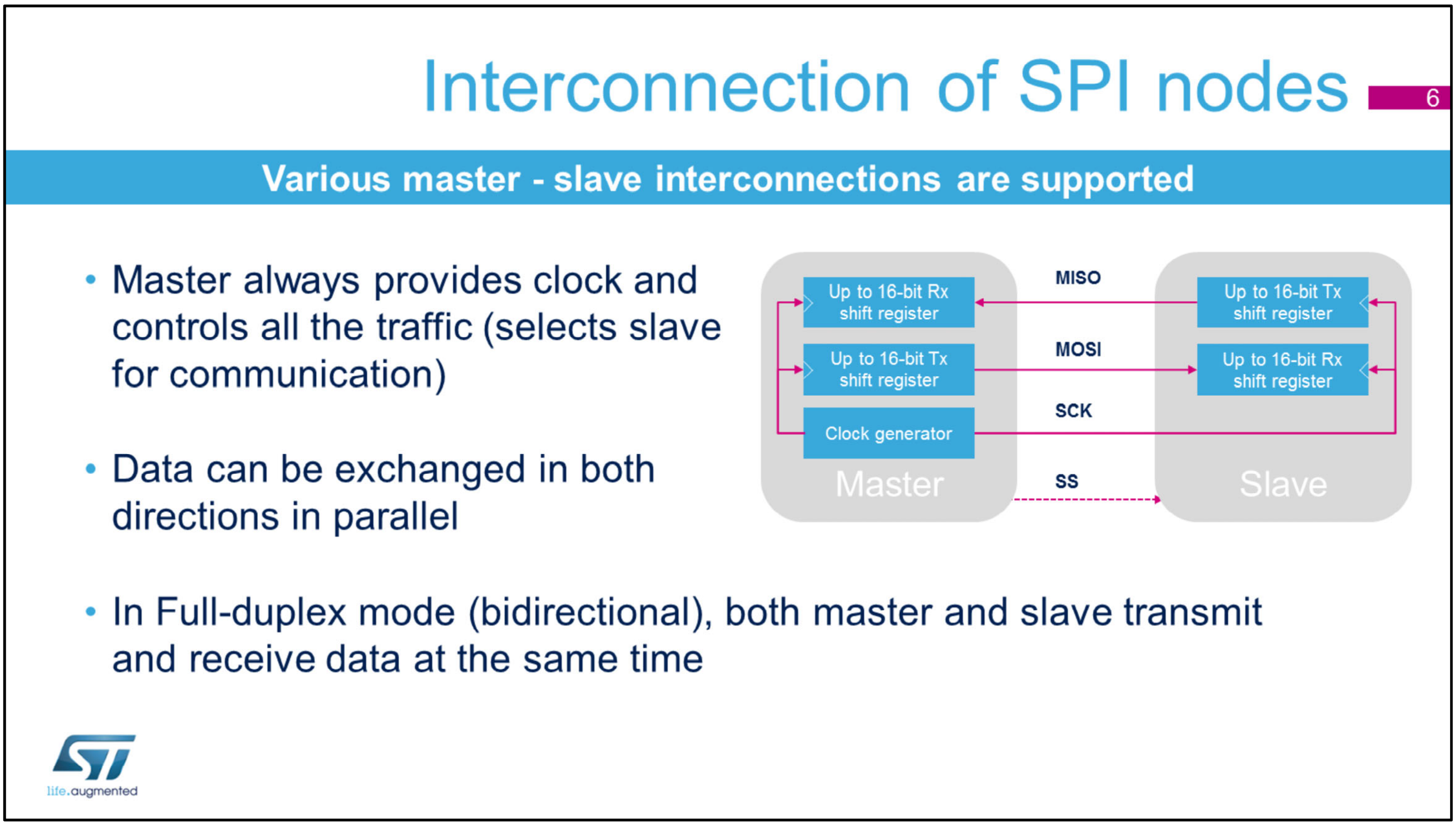


The simplified SPI block diagram shows its basic operation and functions. There are four I/O signals associated with the SPI peripheral. All data passes through the receive and transmit buffers via their specific interfaces.

Data is temporarily stored in two 32-bit embedded Rx and Tx FIFOs with DMA capability.

The NSS is managed by hardware or software for both master and slave, enabling dynamic change of master/slave operations.

The SPI controller supports a hardware cyclic redundancy check (CRC) feature for reliable communication: the CRC value can be transmitted as last bytes in transmit mode and automatic CRC error checking for last received bytes.



Interconnection of SPI nodes

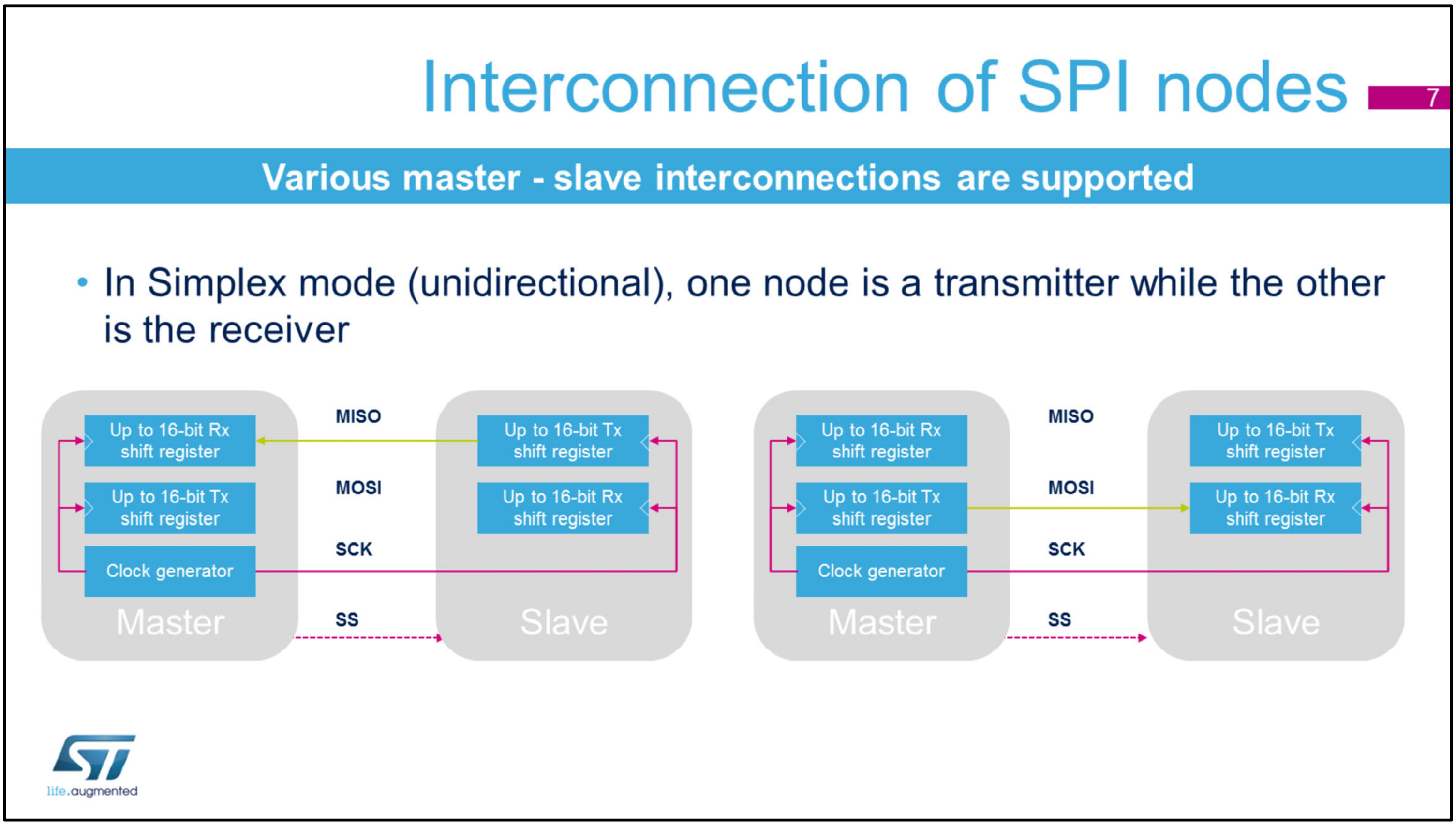
6

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

Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
Master
MISO
MOSI
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave
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 SS 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.



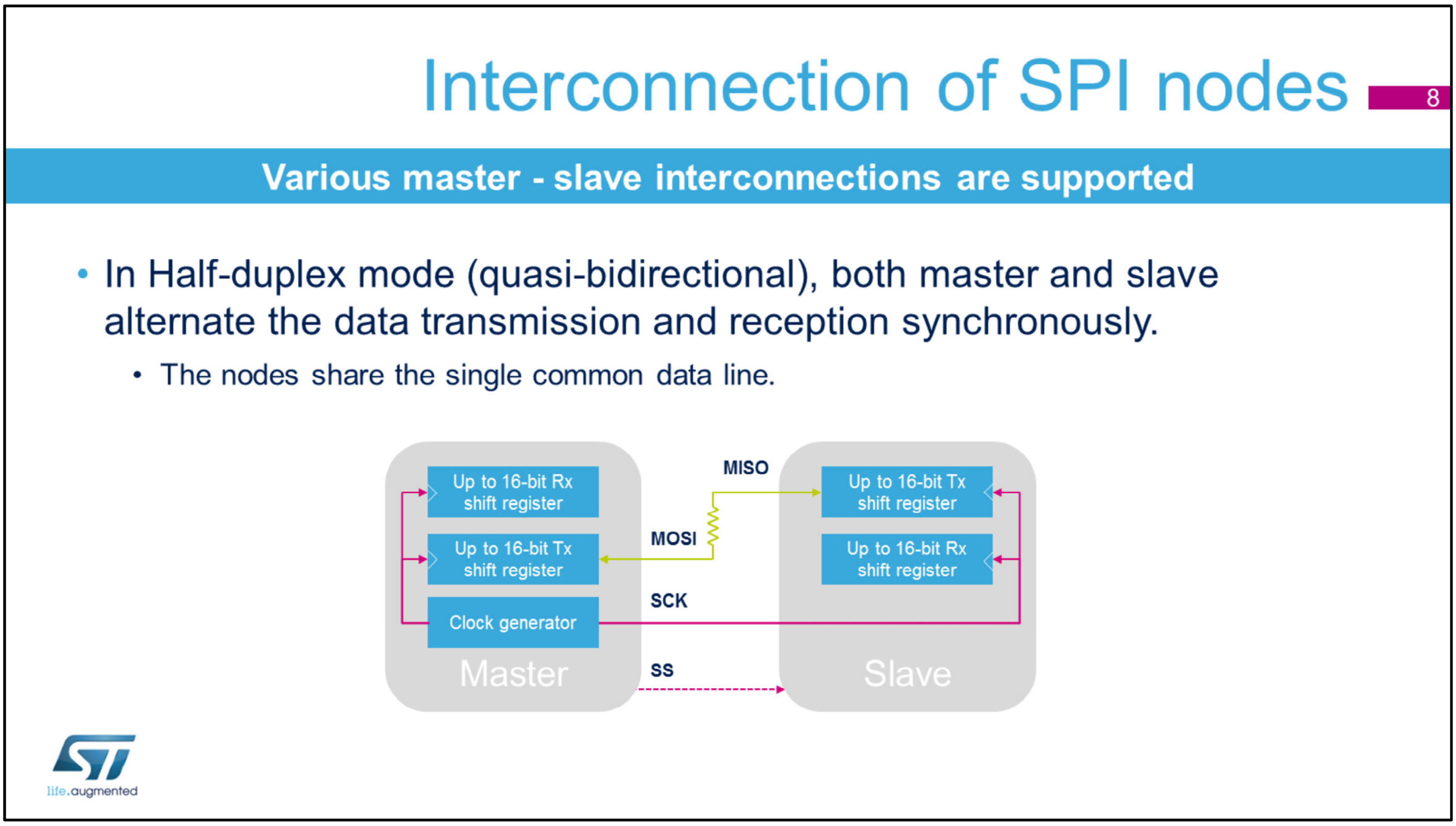
Interconnection of SPI nodes

Various master - slave interconnections are supported

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

Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
Master
MISO
MOSI
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave
Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
Master
MISO
MOSI
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave
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.



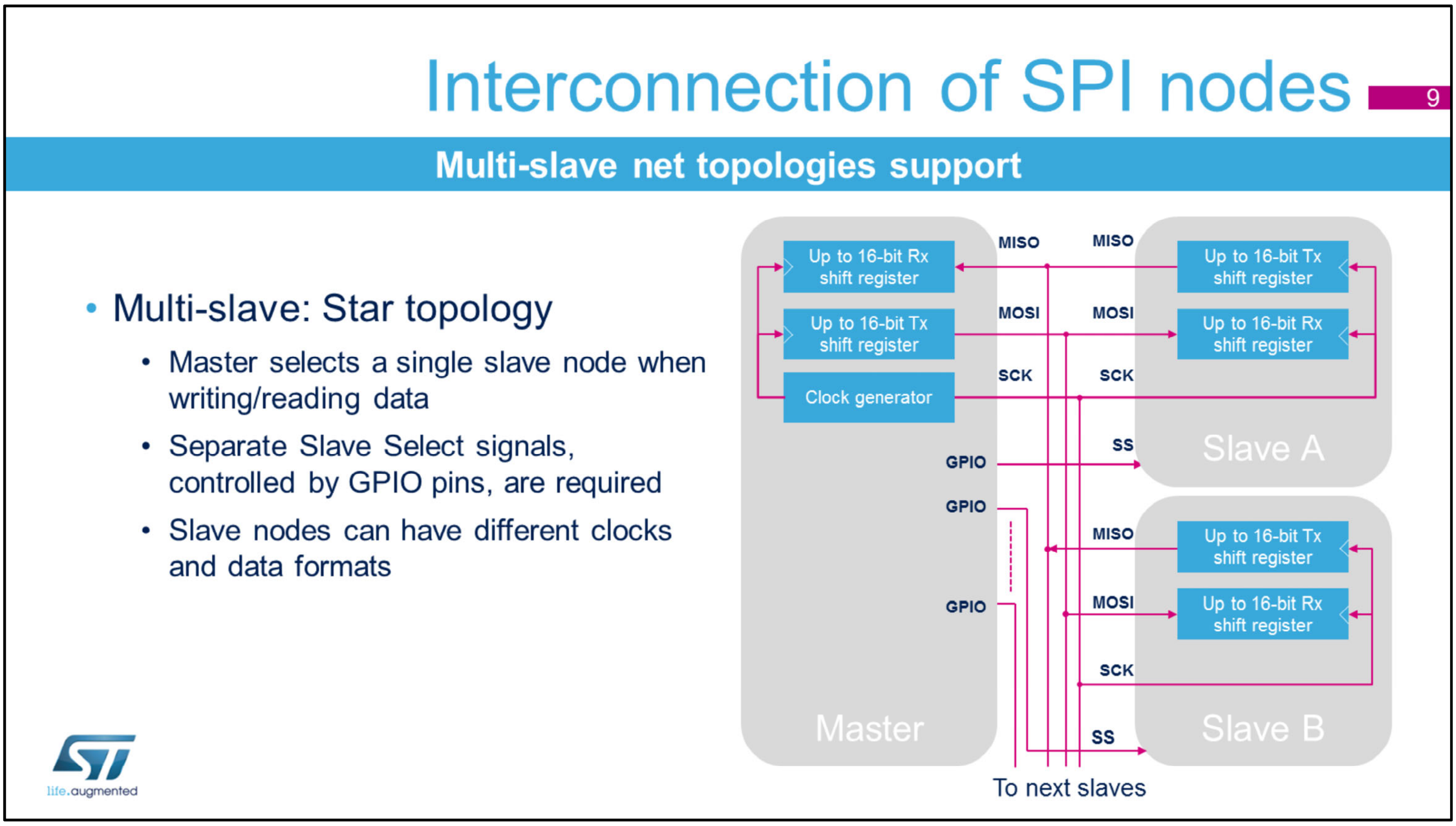
Interconnection of SPI nodes

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.

Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
Master
MISO
MOSI
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave
life.augmented

Half-duplex mode integrates the previous two modes by sharing a single line for data exchanges and data flows in a single direction at a 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 between MISO and MOSI pins to prevent possible temporary short-circuit connection, since master and slave nodes are not usually synchronized.



Interconnection of SPI nodes

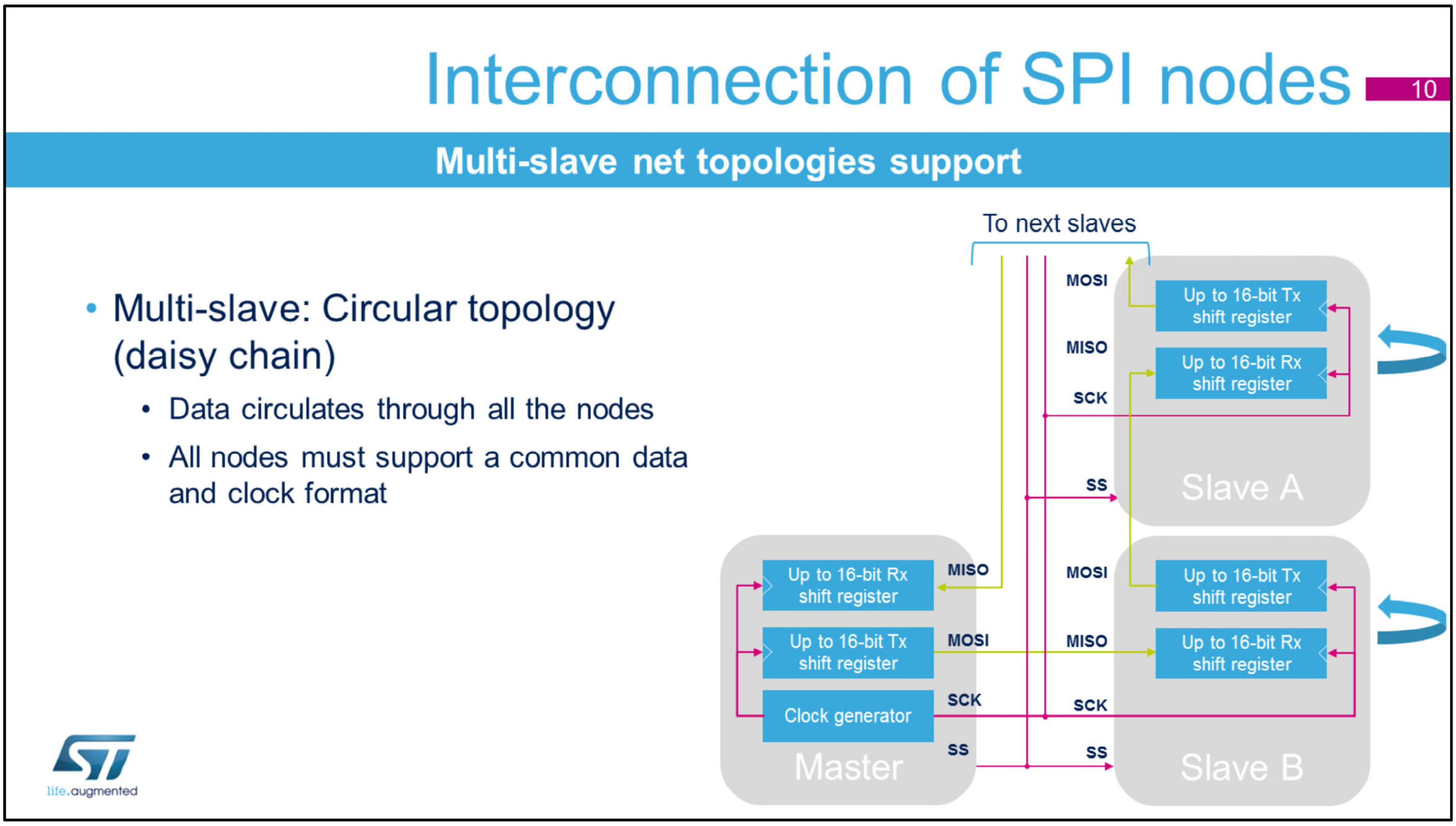
9

Multi-slave net topologies support

- Multi-slave: Star topology
  - Master selects a single slave node when writing/reading data
  - Separate Slave Select signals, controlled by GPIO pins, are required
  - Slave nodes can have different clocks and data formats

Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
Master
GPIO
GPIO
GPIO
MISO
MOSI
SCK
MISO
MOSI
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave A
MISO
MOSI
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave B
To next slaves
life.augmented

When the SPI network includes more than one slave, a star topology is commonly used. The master communicates with one slave at a time, selected by its Slave Select input. In this topology, a separate Slave Select signal from the master has to be provided to each slave node, so the master can select which slave to communicate with via dedicated GPIO pin . Thanks to separate Slave Select signals, SPI data and clock format can be adapted for each slave, if the multiple slave nodes do not have a common configuration.



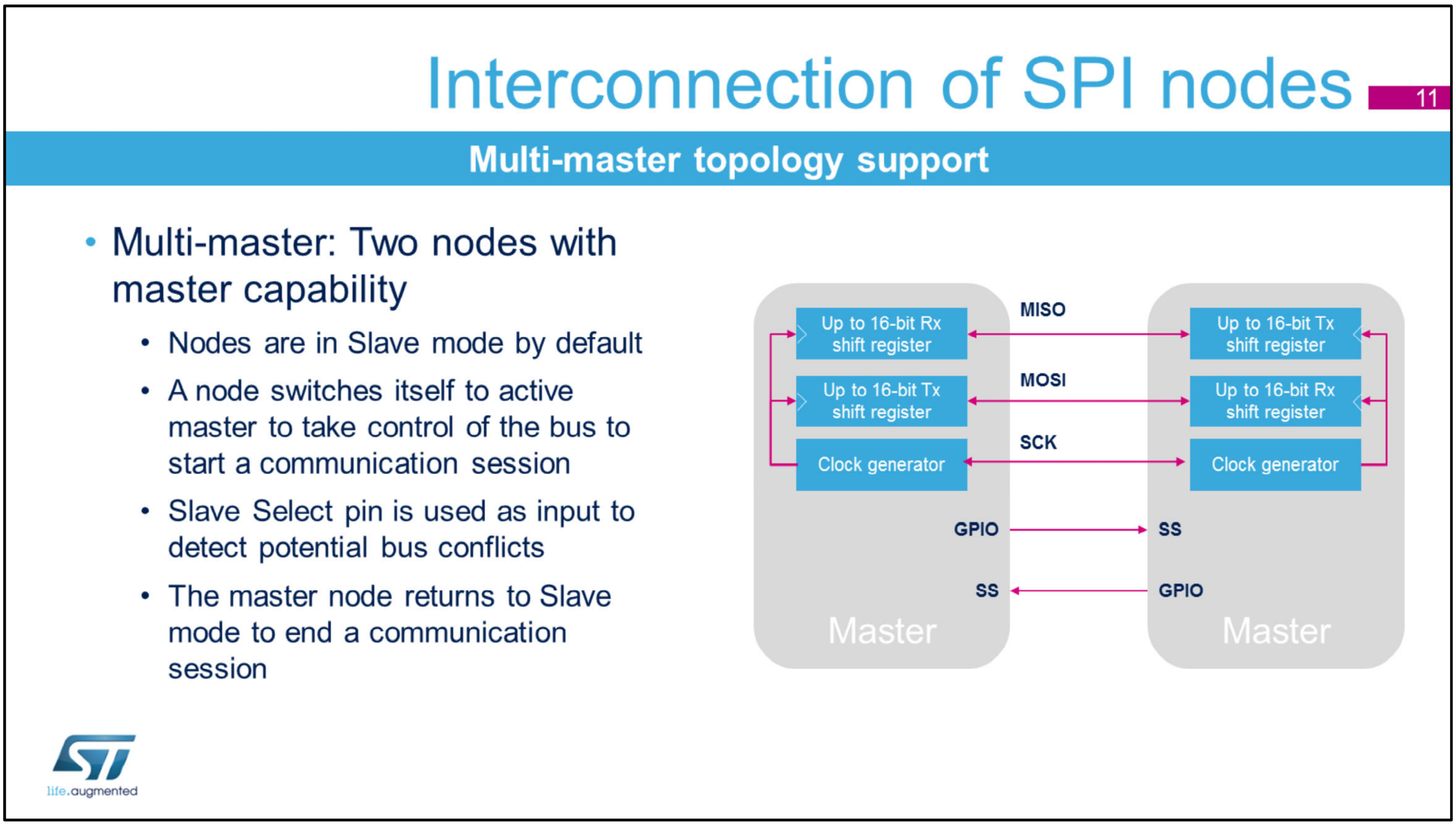
Interconnection of SPI nodes

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

To next slaves
MOSI
MISO
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave A
MISO
MOSI
SCK
SS
Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
Master
MOSI
MISO
SCK
SS
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Slave B
life.augmented

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. When the slave SPI nodes are provided by a microcontrollers, the internal transmit and receive shift registers are usually physically separated, so the data transferred between them has to be handled by software in this circular mode while master node has to provide sufficient time between data to compensate for these internal transfers.



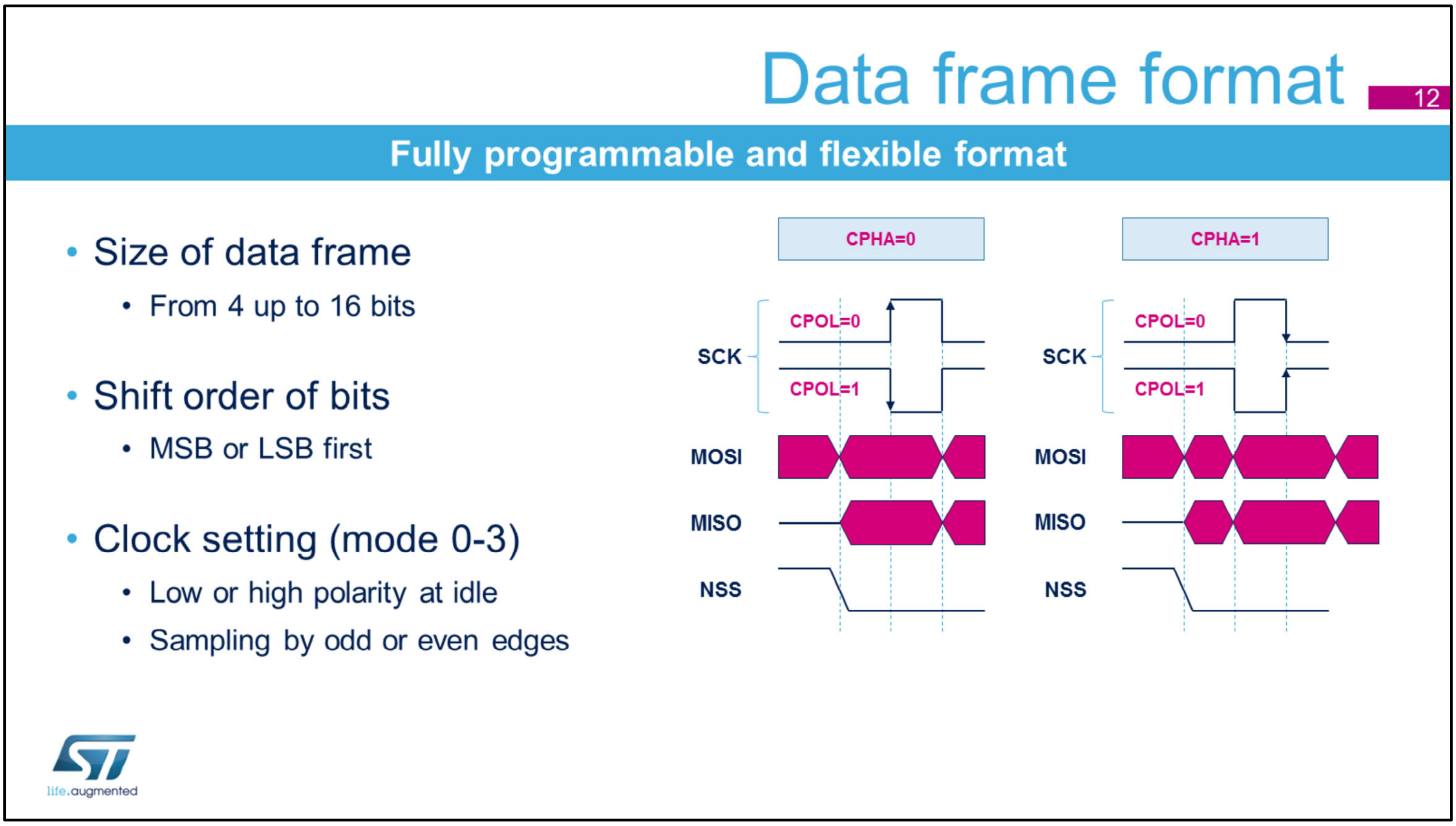
Interconnection of SPI nodes

Multi-master topology support

- Multi-master: Two nodes with master 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

Up to 16-bit Rx shift register
Up to 16-bit Tx shift register
Clock generator
MISO
MOSI
SCK
Up to 16-bit Tx shift register
Up to 16-bit Rx shift register
Clock generator
GPIO
SS
SS
GPIO
Master
Master
life.augmented

SPI networks can operate in a multi-master environment. This mode is used to connect two master nodes exclusively. When neither node is 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 (SS) 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 completed, the active node master releases the Slave Select signal and returns back to passive slave mode waiting for the next session to start.



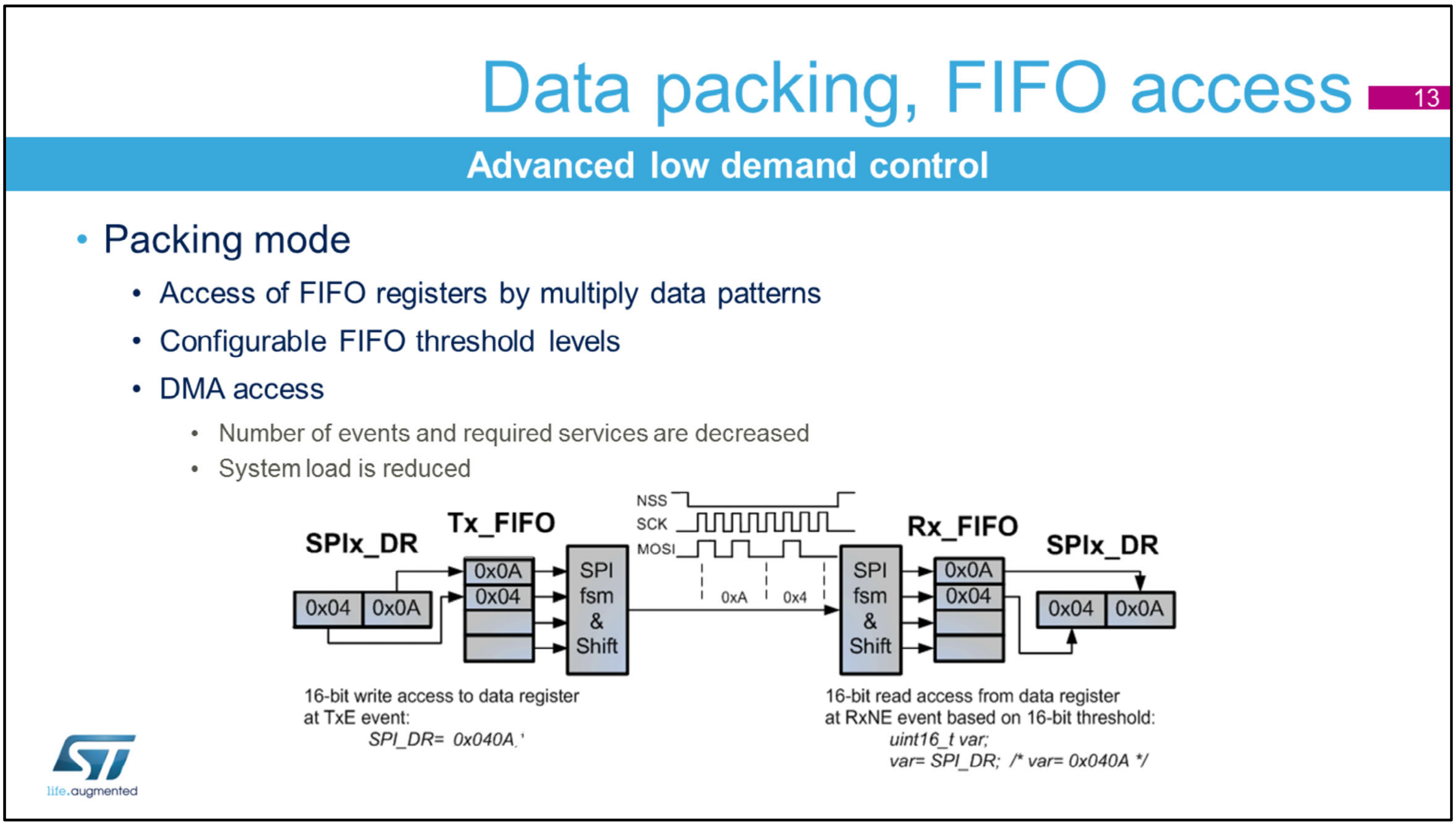
Data frame format

12

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

Several parameters 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

13

Advanced low demand control

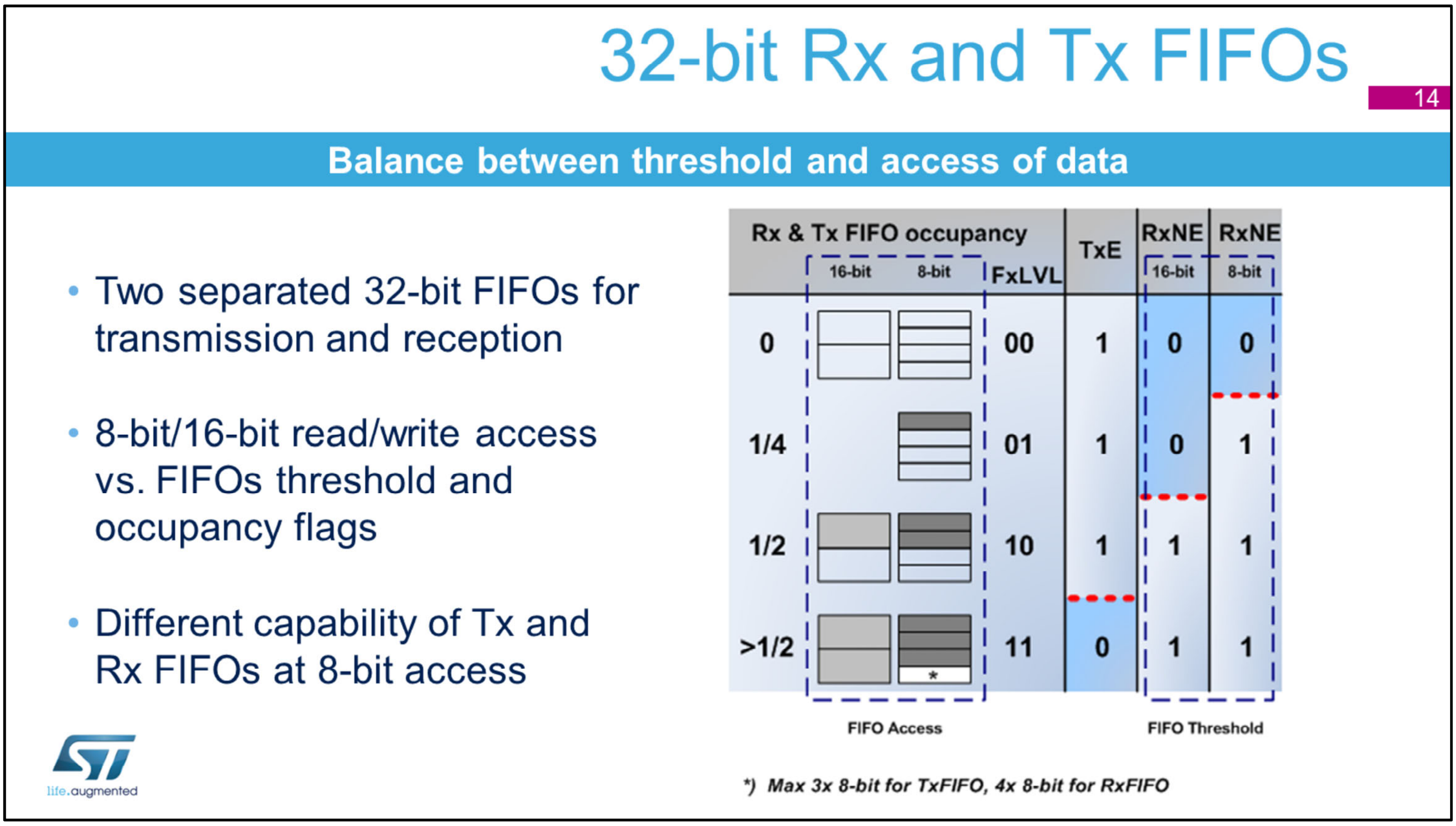
- Packing mode
  - Access of FIFO registers by multiply data patterns
  - Configurable FIFO threshold levels
  - DMA access
    - Number of events and required services are decreased
    - System load is reduced

16-bit write access to data register at TxE event:
SPI_DR= 0x040A;

16-bit read access from data register at RxNE event based on 16-bit threshold:
uint16_t var;
var= SPI_DR; / var= 0x040A */*

life.augmented

When the communication speed is fast and data frames too short, it can be a demanding task to ensure correct data flows especially when the clock signal becomes continuous and Full-duplex mode is used. Slave nodes are more critical as they have to follow properly all the transactions timing provided by the master 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 decreases to better control the data flow. When the DMA is used additionally, overall loading on the system is significantly reduced. In this figure, you can see the principle of how two short 8-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 is performed just upon a single event raised.



32-bit Rx and Tx FIFOs

Balance between threshold and access of data

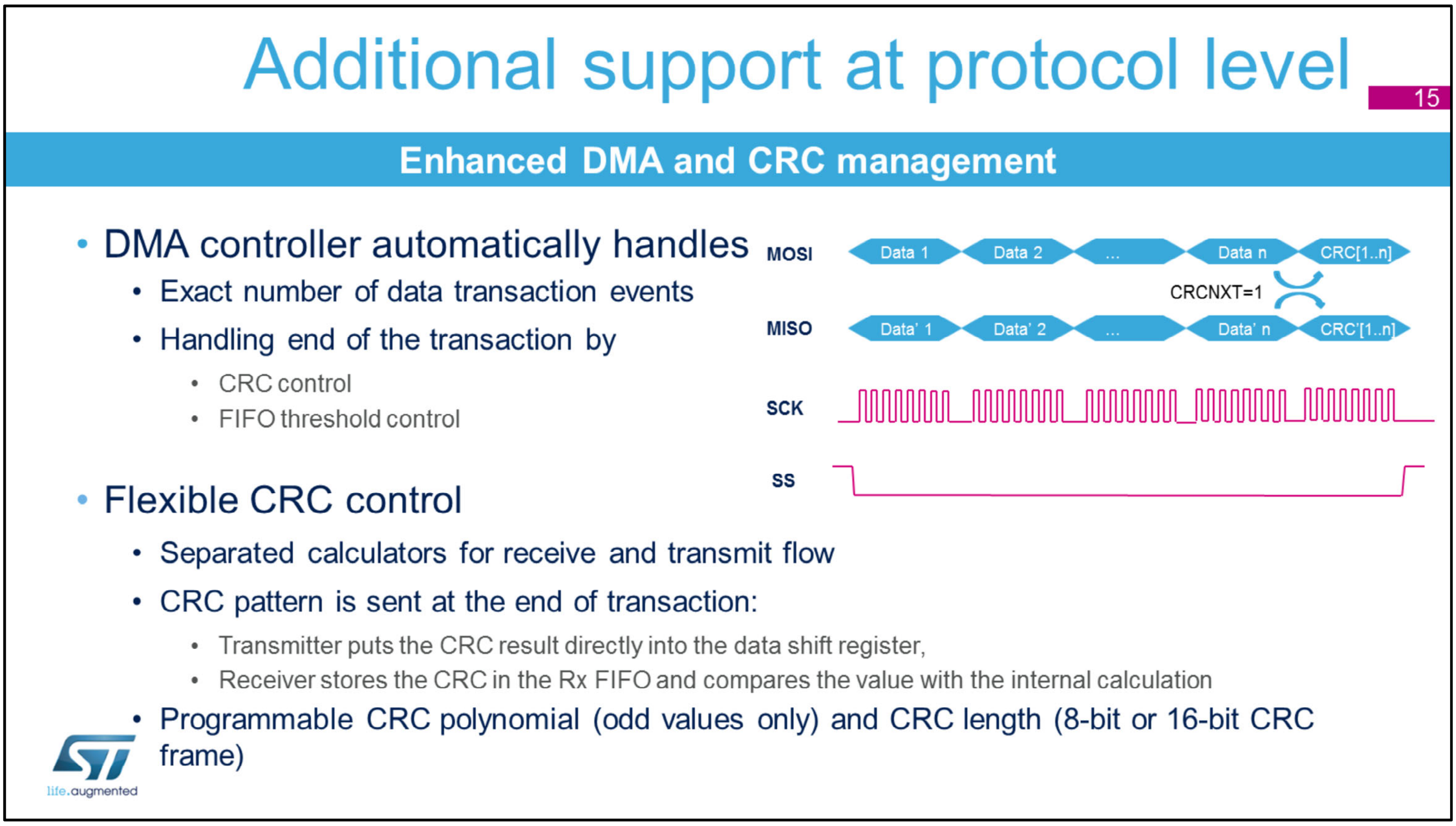
- Two separated 32-bit FIFOs for transmission and reception
- 8-bit/16-bit read/write access vs. FIFOs threshold and occupancy flags
- Different capability of Tx and Rx FIFOs at 8-bit access

Rx & Tx FIFO occupancy	16-bit	8-bit	FxLVL	TxE	RxNE 16-bit	RxNE 8-bit
0			00	1	0	0
1/4			01	1	0	1
1/2			10	1	1	1
>1/2		*	11	0	1	1

*) 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 access instructions. During reception, the event generated from the FIFO depend on threshold setting. The table gives an overview on how the event flag behavior changes depending on the configuration. It is important to keep the FIFO access balanced with the threshold setting so the data consistency is not lost. During transmission, the FIFO occupancy depends on data access.

The system can never predict next access to the transmission FIFO, so the FIFO capability is not fully used when 8-bit write access is applied to fill the second half of the FIFO. At this case, TxE flag is cleared as a consequence in spite of the Tx FIFO is not fully occupied. Look at the star in the figure.



Additional support at protocol level

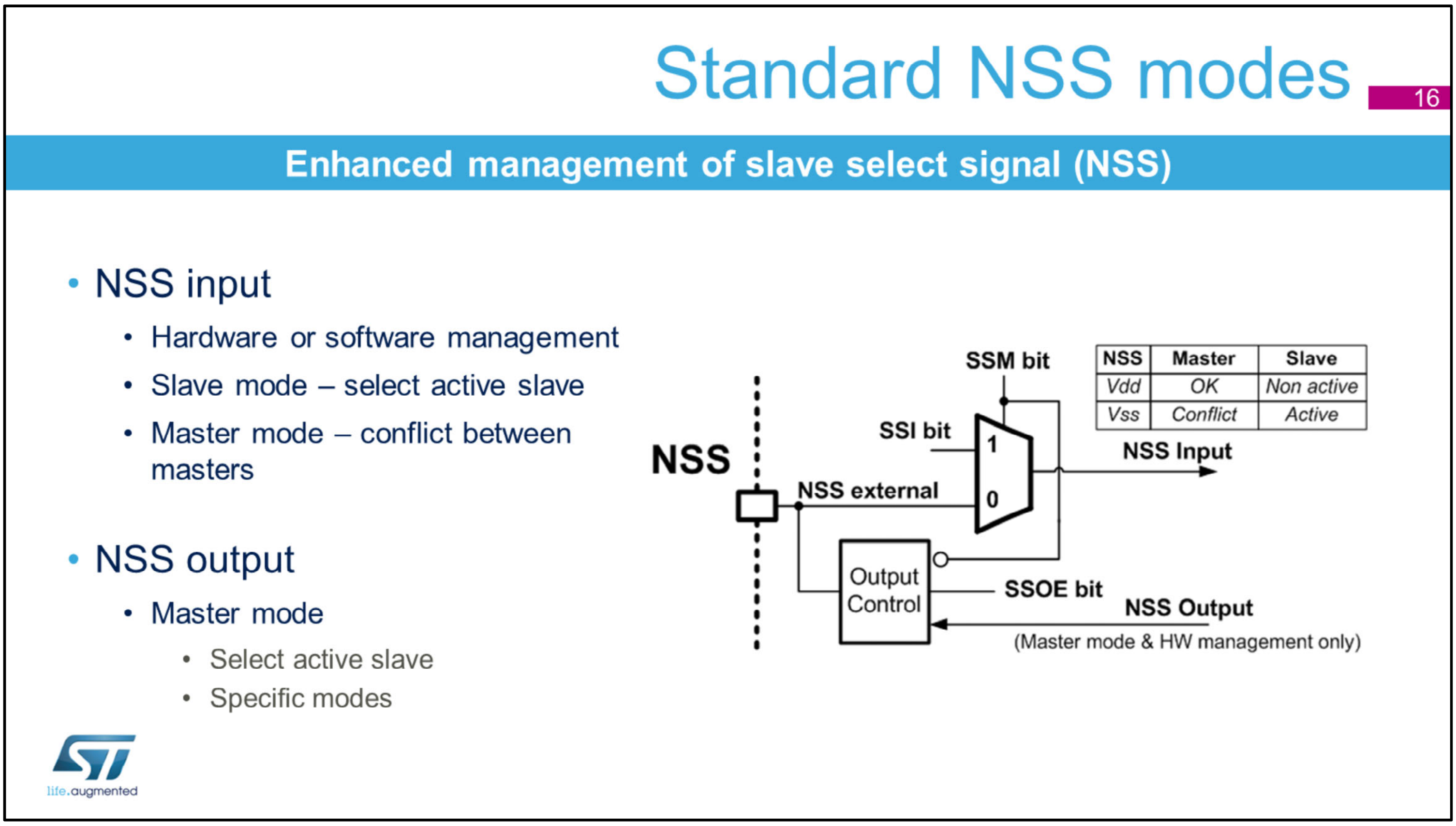
15

Enhanced DMA and CRC management

- DMA controller automatically handles
  - Exact number of data transaction events
  - Handling end of the transaction by
    - CRC control
    - FIFO threshold control
- Flexible CRC control
  - Separated calculators for receive and transmit flow
  - CRC pattern is sent at the end of 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-bit or 16-bit CRC frame)

During protocol level communications, the DMA controller can be used to automatically handle the data flow events, the CRC calculations, and the updating of the FIFO threshold. 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, separate CRC calculators are used for the transmitter and receiver. The CRC calculation result is automatically appended at the end of each transfer 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

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*

The Slave Select signal is commonly used by the master node to select the slave node for communication.

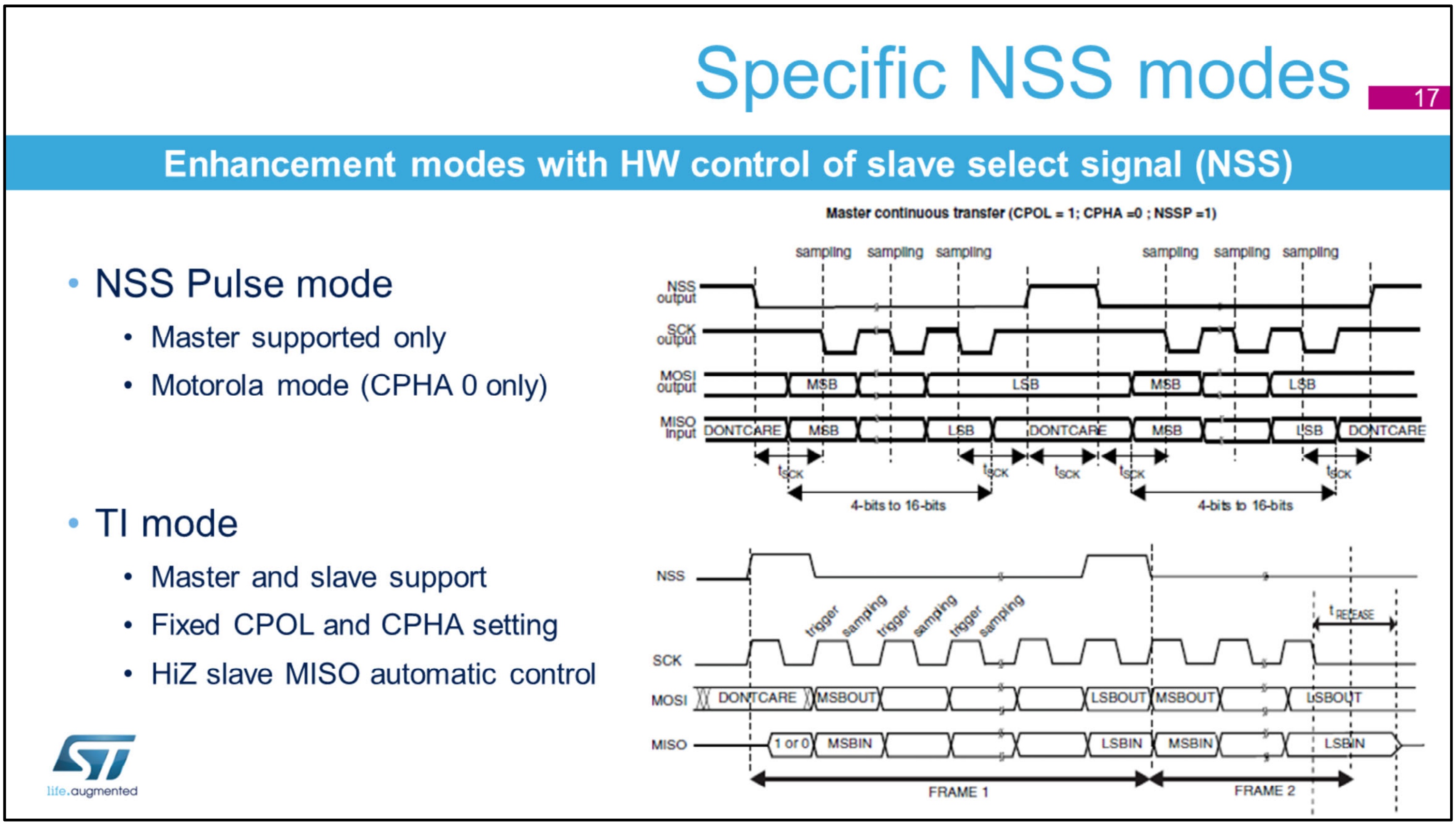
The signal implementation is mandatory in multi-master and multi-slave topologies. Though it is not mandatory at a single master-slave pair, it could be helpful for data flow synchronization, regardless of the topology case.

The Slave Select signal can operate either as an input or as an output.

The NSS input can be managed by hardware or software depending on the SSM and SSI control bits.

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 only works as an output in Master mode and is 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 HW 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 MISO automatic control

There are a few enhanced modes when the Slave Select signal is under specific hardware control.

The Slave Select signal can operate in a pulse mode where the master generates pulses on the signal between data frames.

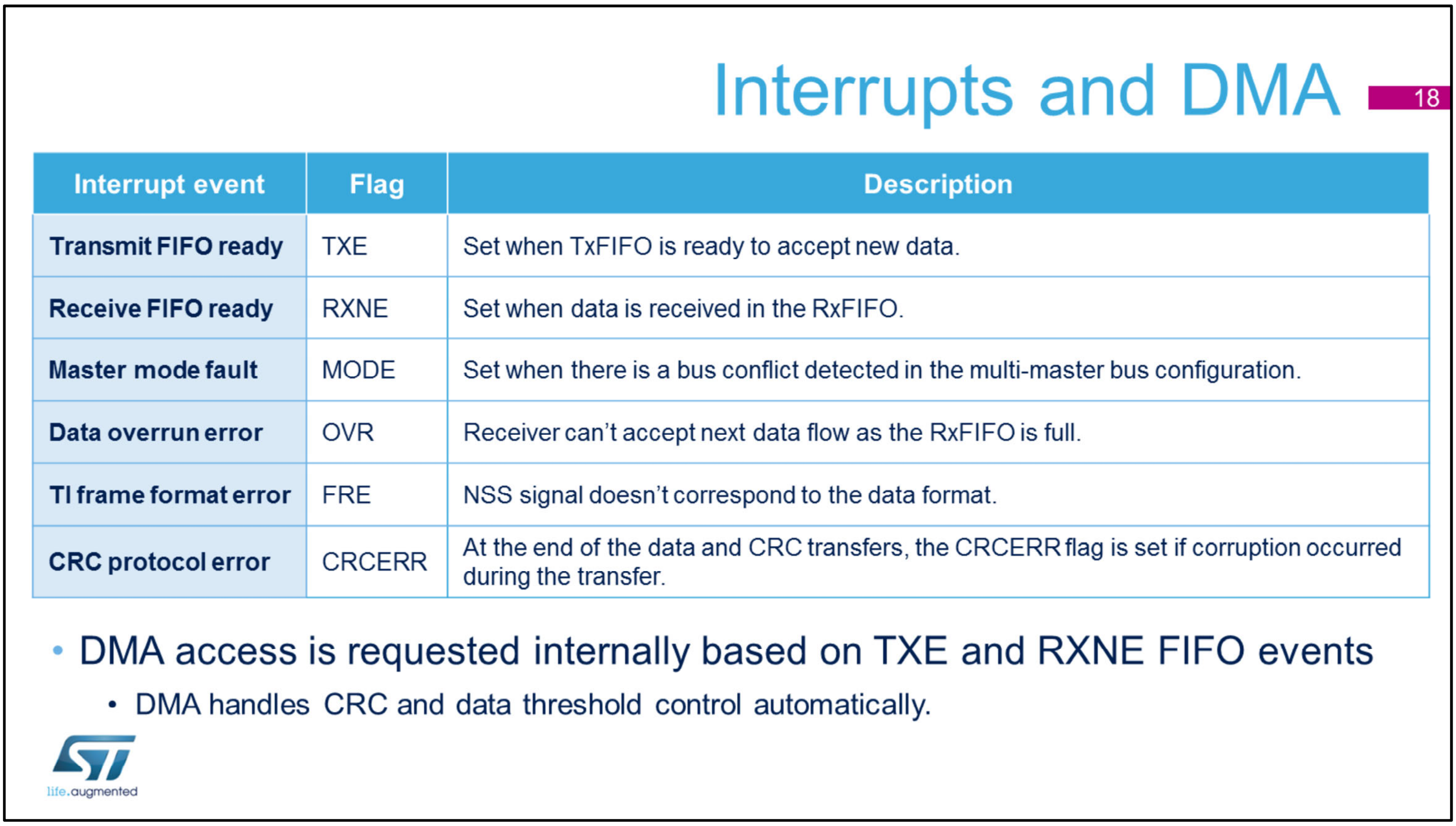
- NSS goes high between data frames for 1 SCK period for continuous transactions.
- The clock polarity and phase are fixed, the CPOL setting is ignored, and the CPHA has to be kept cleared.

Another enhanced mode is the TI mode where the data flow is synchronized by the SS 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.

- CPOL=0 CPHA=1 setting has to be kept if CRC is applied in TI mode.

A CRC cannot be applied in NSS pulse mode.



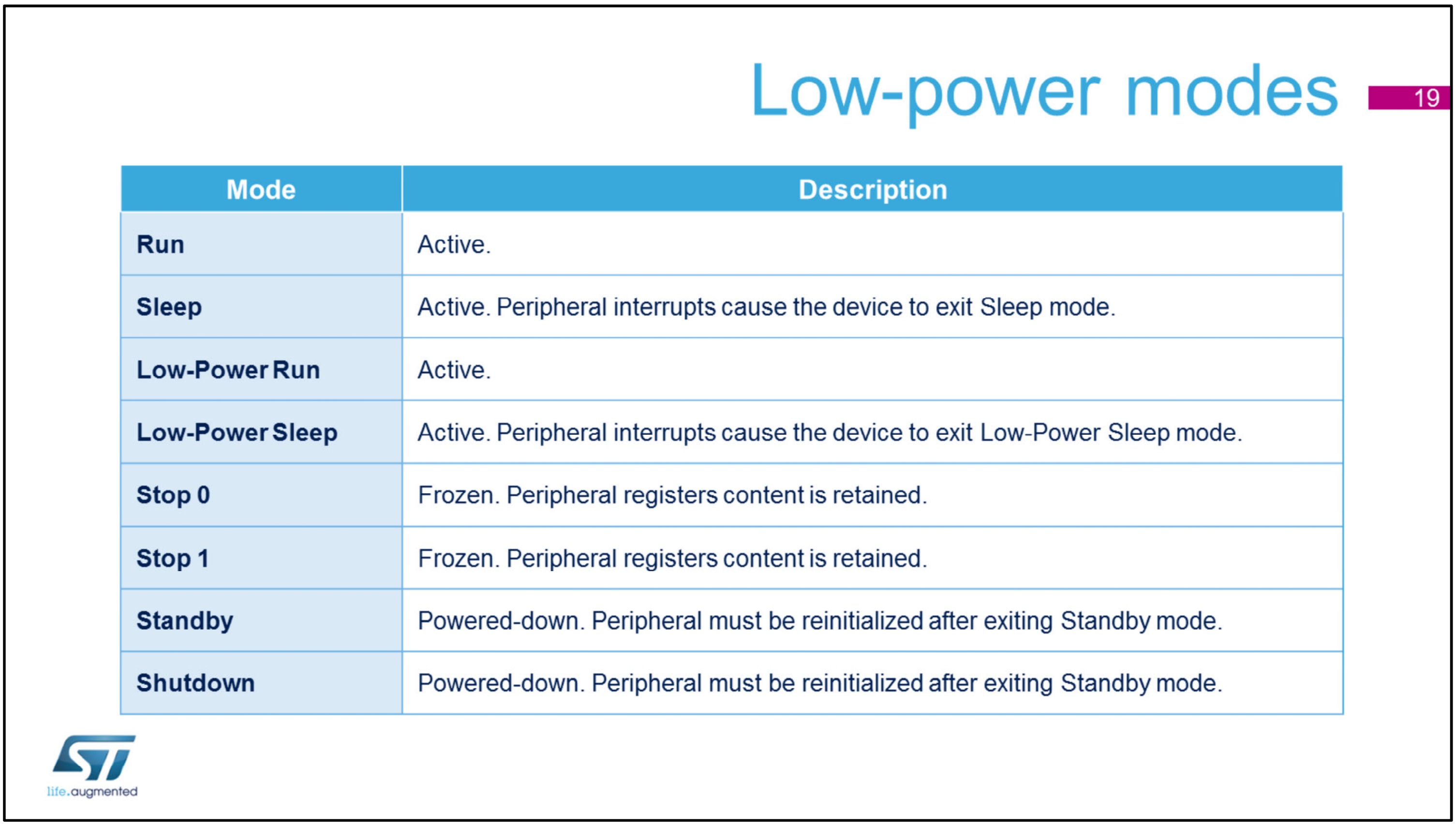
Interrupts and DMA

Interrupt event	Flag	Description
Transmit FIFO ready	TXE	Set when TxFIFO is ready to accept new data.
Receive FIFO ready	RXNE	Set when data is received in the RxFIFO.
Master mode fault	MODE	Set when there is a bus conflict detected in the multi-master bus configuration.
Data overrun error	OVR	Receiver can't accept next data flow as the RxFIFO is full.
TI frame format error	FRE	NSS signal doesn't correspond to the data format.
CRC protocol error	CRCERR	At the end of the data and CRC transfers, the CRCERR flag is set if corruption occurred during the transfer.

- DMA access is requested internally based on TXE and RXNE FIFO events
  - DMA handles CRC and data threshold control automatically.

life.augmented

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

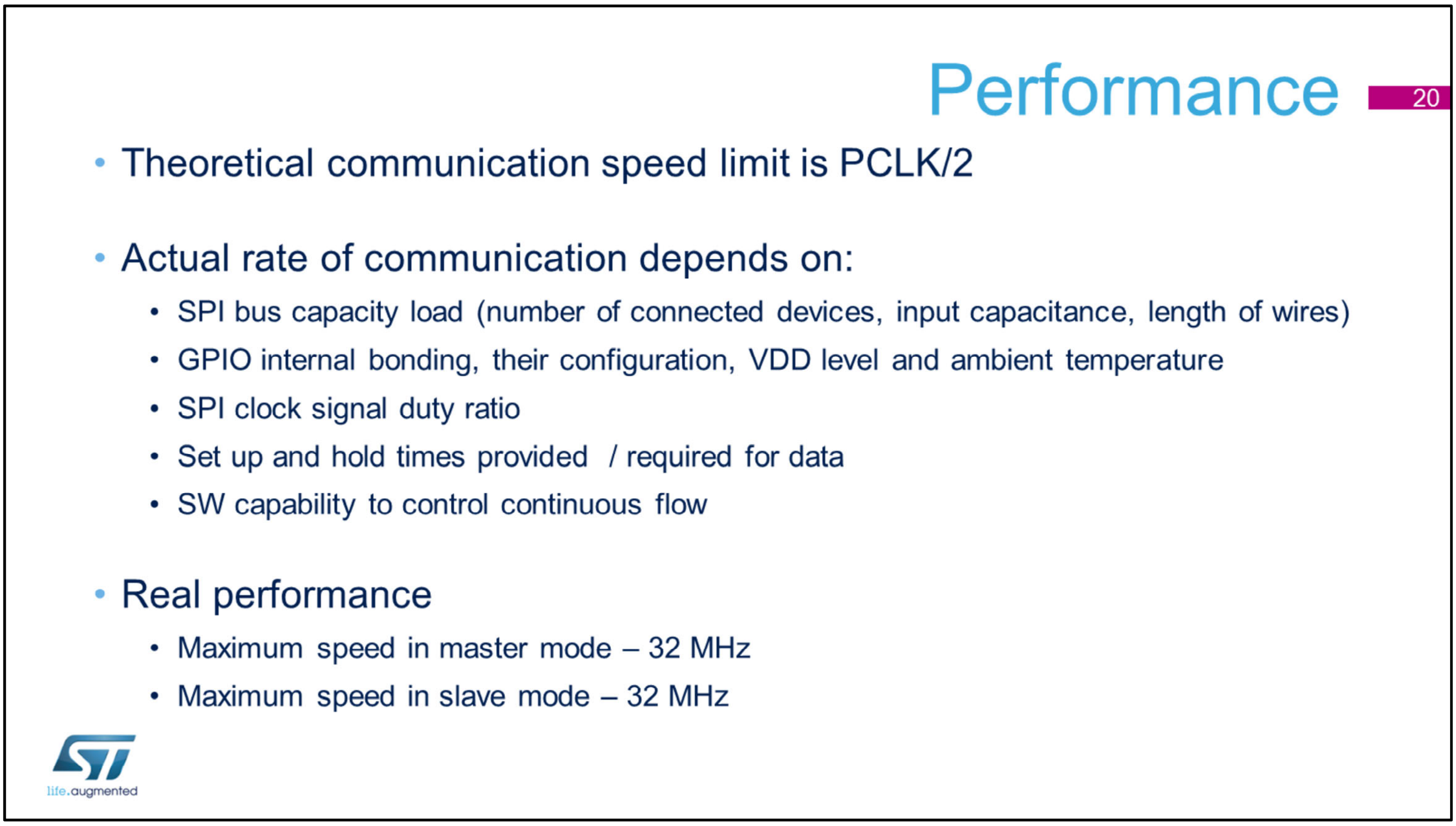


Low-power modes

Mode	Description
Run	Active.
Sleep	Active. Peripheral interrupts cause the device to exit Sleep mode.
Low-Power Run	Active.
Low-Power Sleep	Active. Peripheral interrupts cause the device to exit Low-Power Sleep mode.
Stop 0	Frozen. Peripheral registers content is retained.
Stop 1	Frozen. Peripheral registers content is retained.
Standby	Powered-down. Peripheral must be reinitialized after exiting Standby mode.
Shutdown	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, Standby or Shutdown modes. It is important to ensure that all the SPI traffic is completed before the peripheral enters Stop or Power down modes.



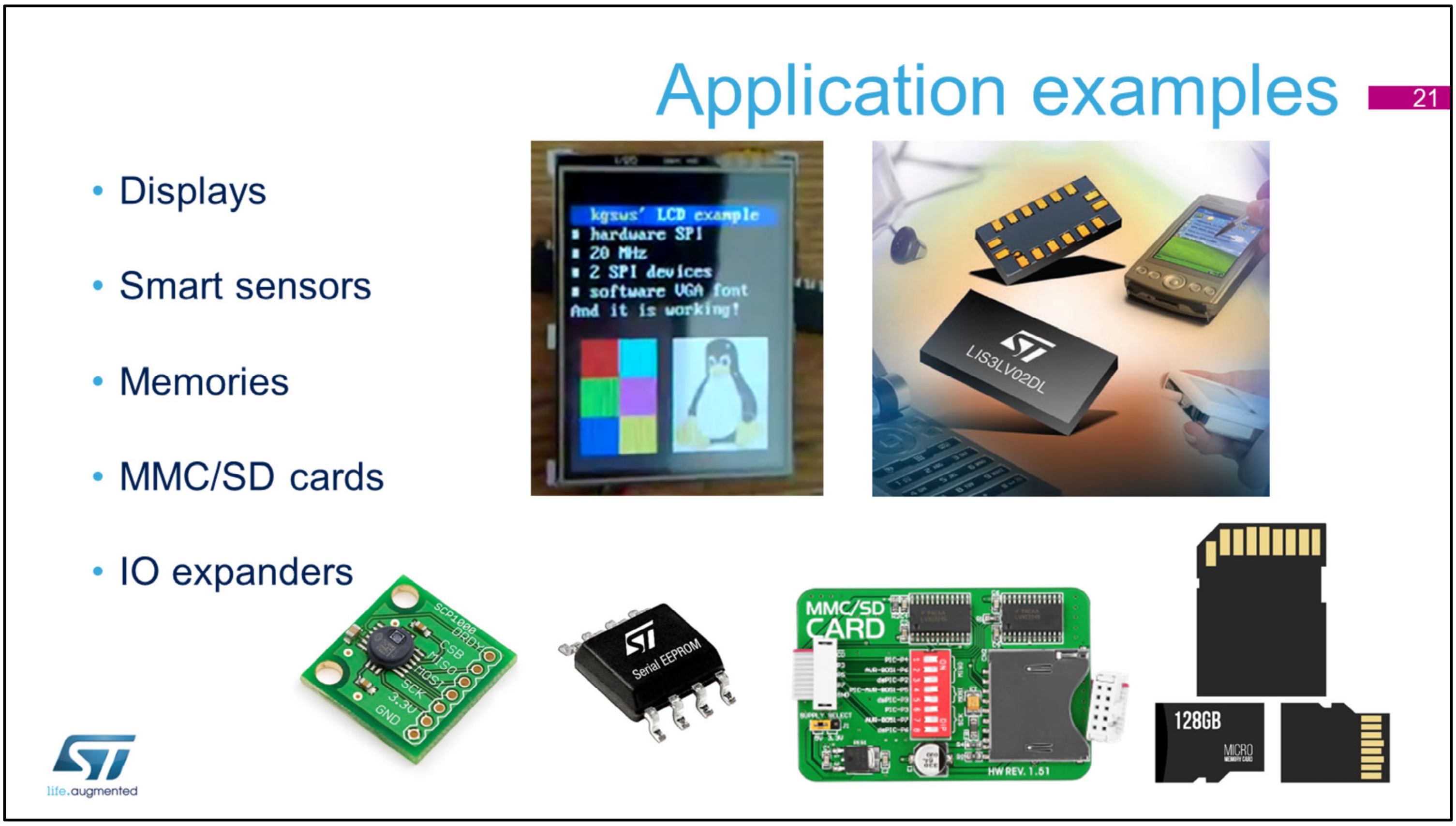
Performance

- Theoretical communication speed limit is PCLK/2
- Actual rate of communication depends on:
  - SPI bus capacity load (number of connected devices, input capacitance, length of wires)
  - GPIO internal bonding, their configuration, VDD level and ambient temperature
  - SPI clock signal duty ratio
  - Set up and hold times provided / required for data
  - SW capability to control continuous flow
- Real performance
  - Maximum speed in master mode – 32 MHz
  - Maximum speed in slave mode – 32 MHz

life.augmented

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. The DMA capacity has to be considered as well as the number of DMA channels used by the system, frequent interrupt services or execution of non-interruptible instructions.

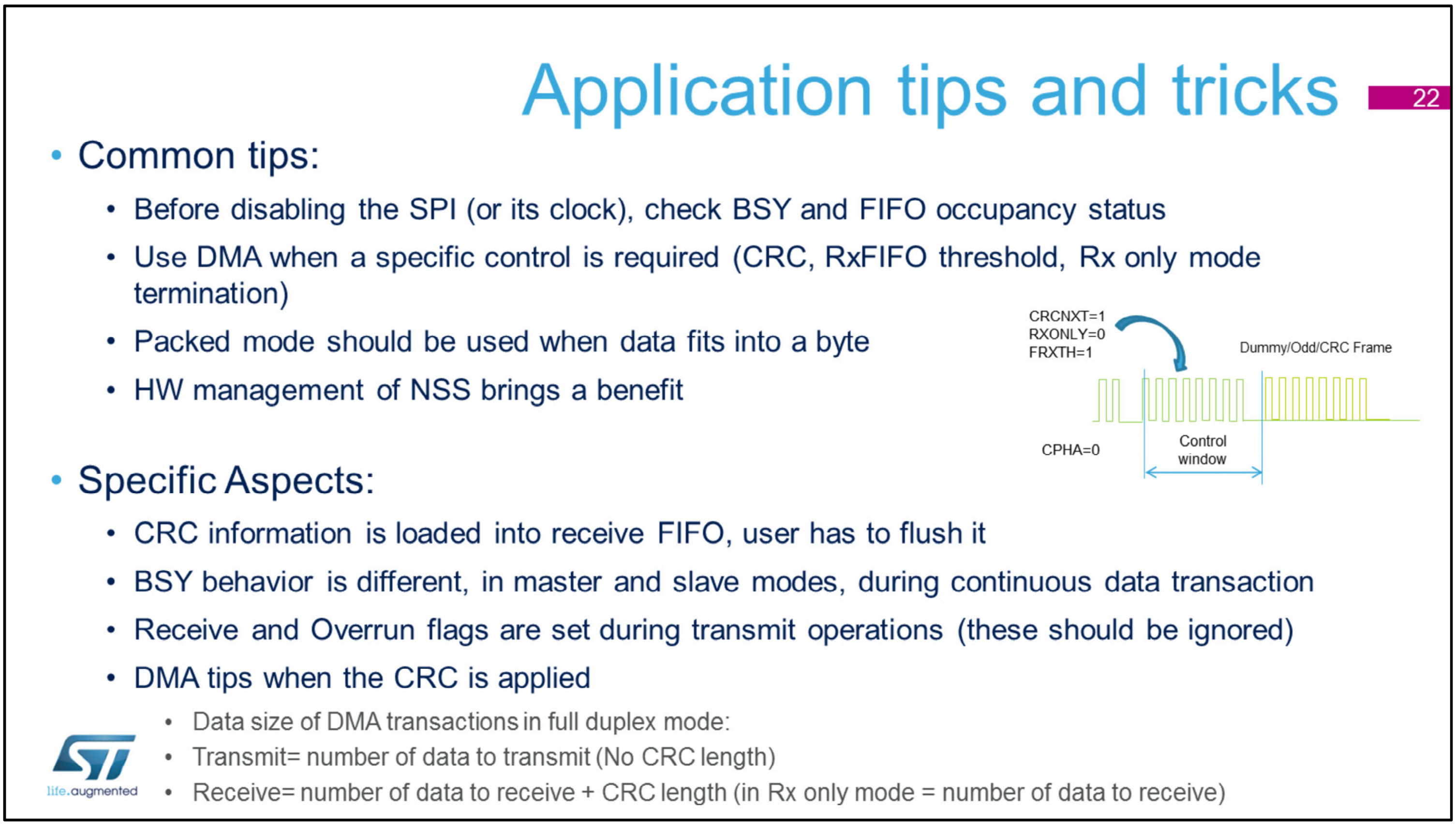


Application examples

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

kgsws' LCD example
hardware SPI
20 MHz
2 SPI devices
software VGA font
And it is working!
LIS3LV02DL
SCP1000
DRDY
CSB
MISO
MOSI
SCK
3.3V
GND
Serial EEPROM
MMC/SD CARD
HW REV. 1.61
128GB
MICRO
life.augmented

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.



Application tips and tricks

22

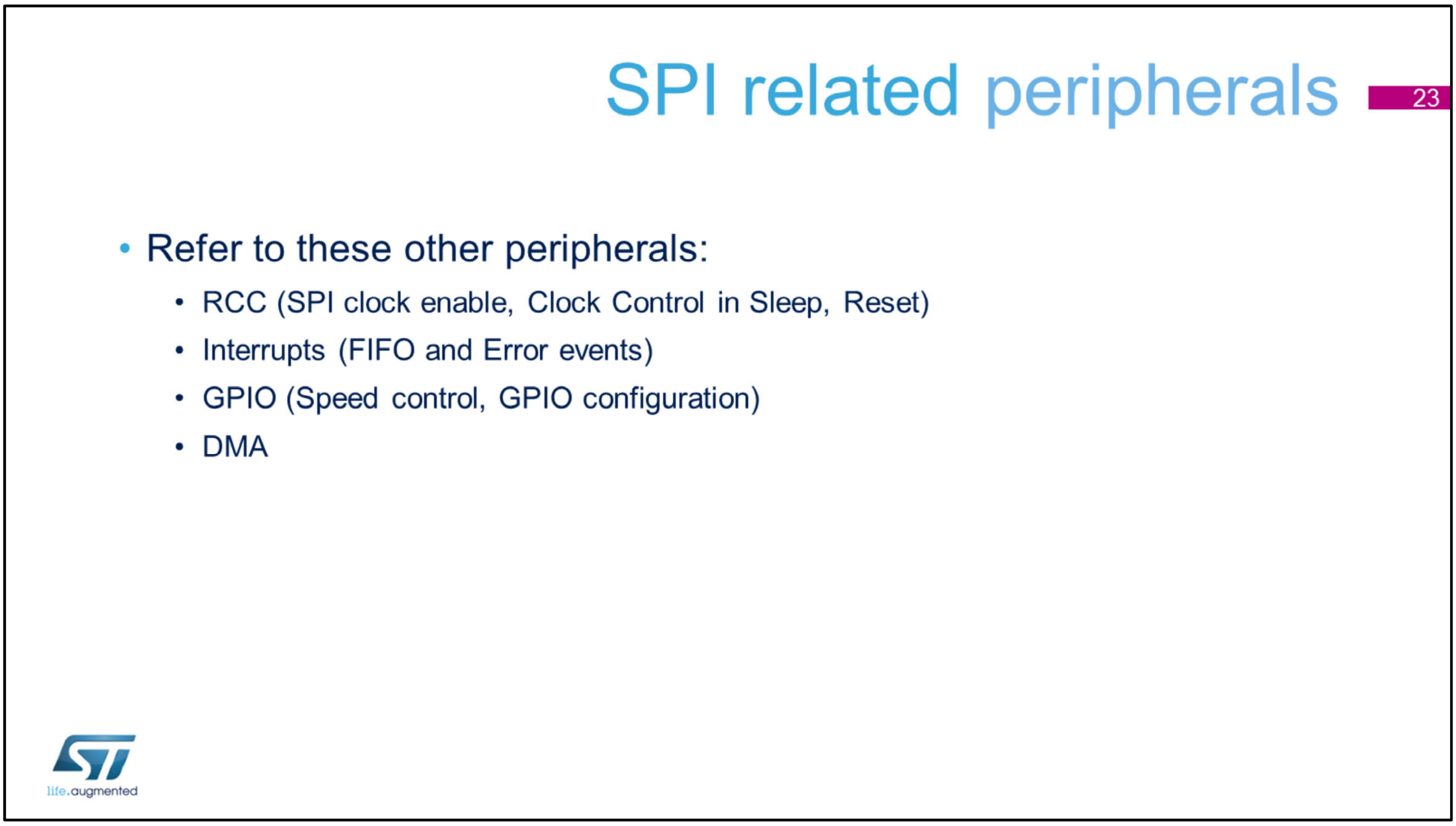
- Common tips:
  - Before disabling the SPI (or its clock), check BSY and FIFO occupancy status
  - Use DMA when a specific control is required (CRC, RxFIFO threshold, Rx only mode termination)
  - Packed mode should be used when data fits into a byte
  - HW management of NSS brings a benefit
- Specific Aspects:
  - CRC information is loaded into 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 when the CRC is applied
    - Data size of DMA transactions in full duplex mode:
    - Transmit= number of data to transmit (No CRC length)
    - Receive= number of data to receive + CRC length (in Rx only mode = number of data to receive)

Common tips are:
The user should be aware that traffic on the bus can still be ongoing even if the DMA transaction is complete or the transmit FIFO becomes empty.
This is why the user has to carefully check the peripheral’s status and follow the suggested procedures before disabling the SPI or placing it in Stop mode.
Use the DMA if you have a specific verification that requires CRC handling, or receive FIFO threshold verification (when the number of data is not aligned at packet mode), or to receive an exact amount of data in Receive-only mode. These types of verifications must be applied during a specific time window within a frame transaction exclusively so all the following transactions are handled properly.
The DMA and data packing can increase the system’s overall performance. These features can help when data

frames are short and a high-speed continuous communication flow is required.

Hardware management of NSS is not quite necessary in a single-master/single-slave pair, but it can help synchronize the data flow and prevent conflicts in a multi-master system.

- There are some additional specific aspects which should be taken into account when designing for an SPI network.
- The receiver always loads CRC information into the receive FIFO. The user has to account for this in the buffer and flush it.
- The Busy flag should not be used for any data handling but to check for ongoing traffic. The BSY bit stays set between data frames during the master continuous data transactions. It always drops 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 flow stays active. The user must ignore all receive and associated overrun events in this case.
- The data size to be processed by the DMA when including the CRC is dependent on mode. When a CRC is used, the data size has to be set differently for transmitting and receiving in Full-duplex mode as well as in Receive-only mode.

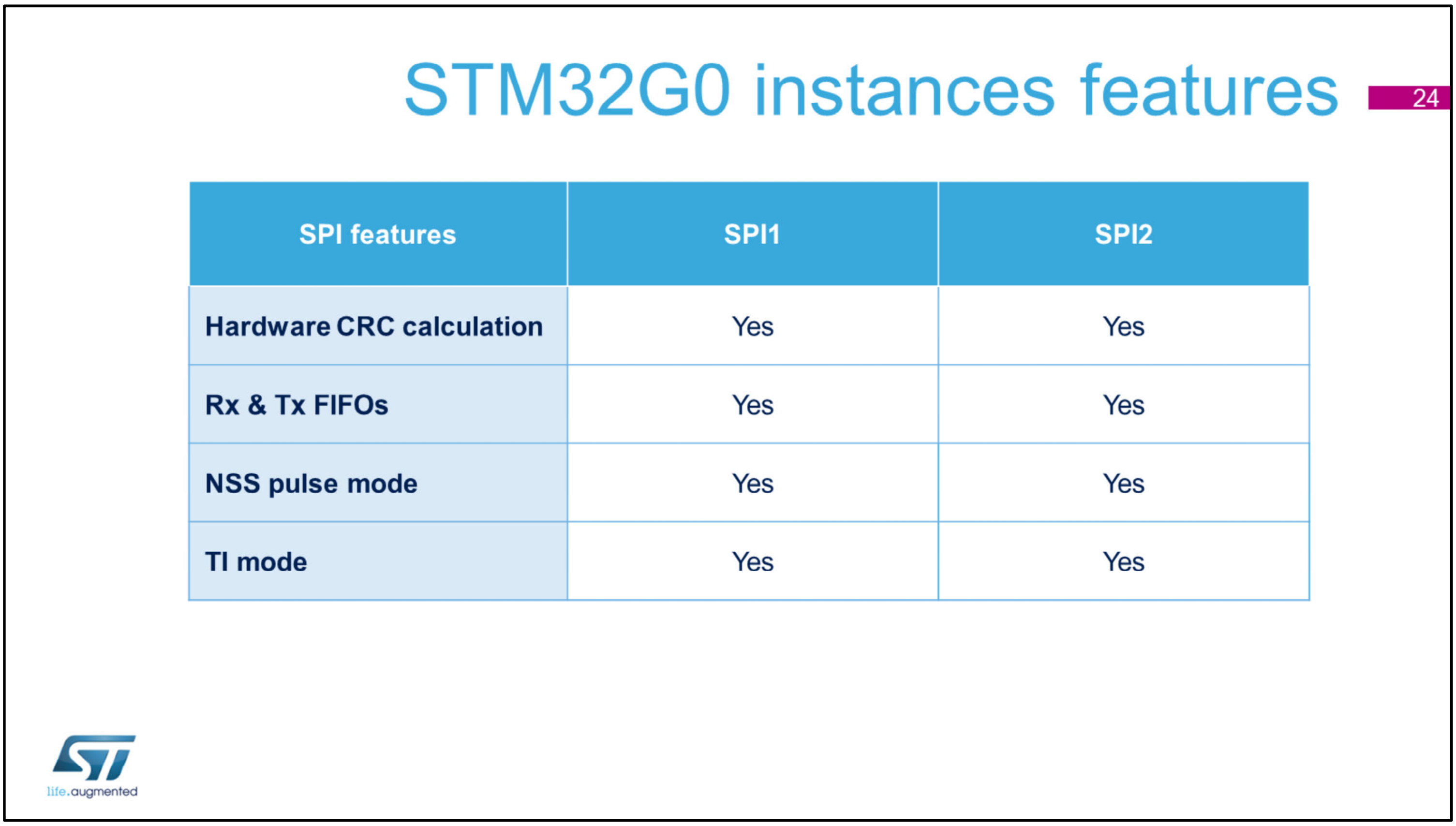


SPI related peripherals

23

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

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.

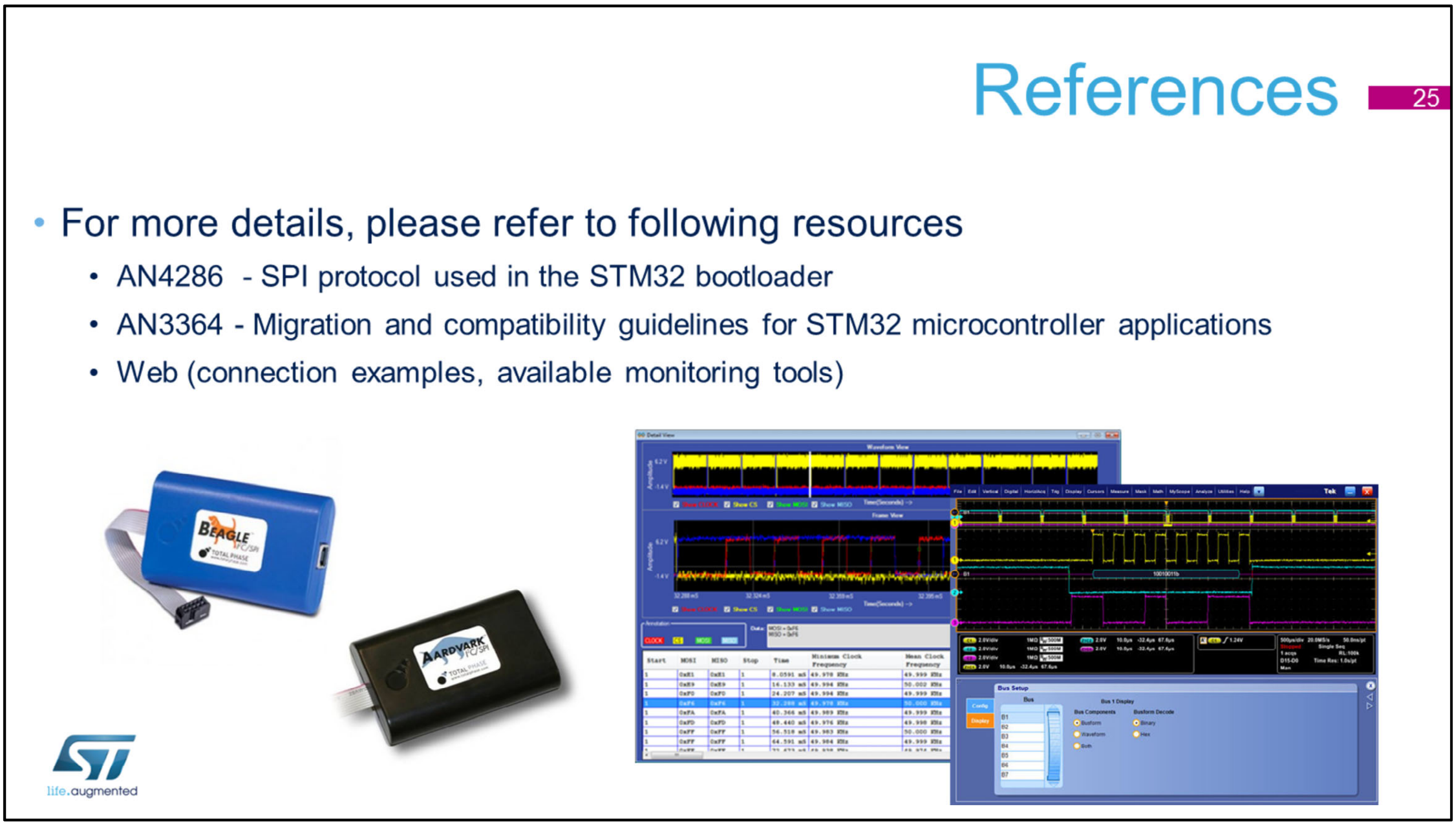


STM32G0 instances features

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

life.augmented

There are two SPI instances within the STM32G0 microcontroller, and each support all the features described in this presentation.



References

25

- 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)

BEAGLE I²C/SPI
TOTAL PHASE
AARDVARK I²C/SPI
TOTAL PHASE
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.

Slide 25

RQ1 Remi QUINTIN; 20/06/2018