Customer Training Workshop Traveo™ II Serial Communication Block (SCB)

Q4 2020

Infineon

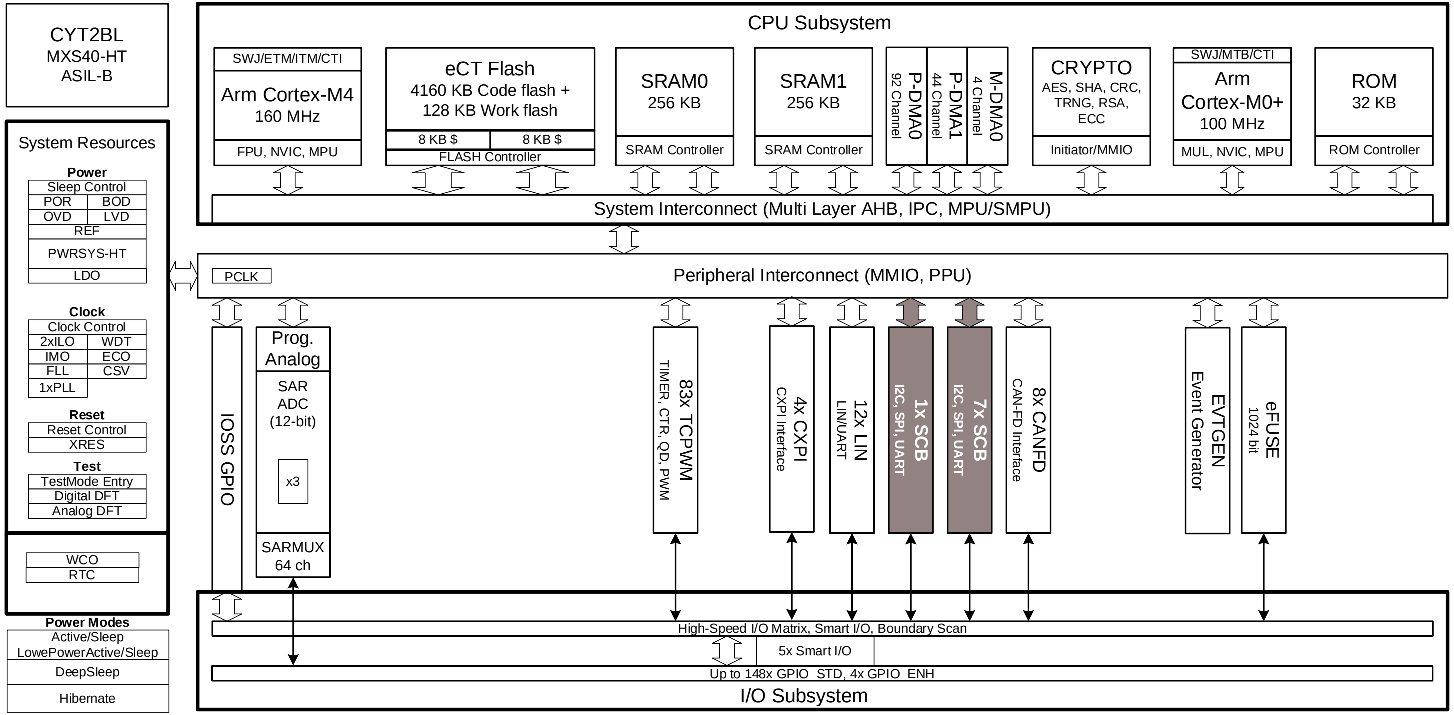
Target Products

- Target product list for this training material

Family Category	Series	Code Flash Memory Size
Traveo™ II Automotive Body Controller Entry	CYT2B6	Up to 576KB
Traveo II Automotive Body Controller Entry	CYT2B7	Up to 1088KB
Traveo II Automotive Body Controller Entry	CYT2B9	Up to 2112KB
Traveo II Automotive Body Controller Entry	CYT2BL	Up to 4160KB
Traveo II Automotive Body Controller High	CYT3BB/4BB	Up to 4160KB
Traveo II Automotive Body Controller High	CYT4BF	Up to 8384KB
Traveo II Automotive Cluster	CYT3DL	Up to 4160KB
Traveo II Automotive Cluster	CYT4DN	Up to 6336KB

Introduction to Traveo II Body Controller Entry

- SCB is part of the Peripheral blocks



CYT2BL
MXS40-HT
ASIL-B

System Resources

Power
Sleep Control
POR | BOD
OVD | LVD
REF
PWRSYS-HT
LDO

Clock
Clock Control
2xILO | WDT
IMO | ECO
FLL | CSV
1xPLL

Reset
Reset Control
XRES

Test
TestMode Entry
Digital DFT
Analog DFT

WCO
RTC

Power Modes
Active/Sleep
LowePowerActive/Sleep
DeepSleep
Hibernate

CPU Subsystem

SWJ/ETM/ITM/CTI
Arm Cortex-M4
160 MHz
FPU, NVIC, MPU

eCT Flash
4160 KB Code flash +
128 KB Work flash
8 KB $ | 8 KB $
FLASH Controller

SRAM0
256 KB
SRAM Controller

SRAM1
256 KB
SRAM Controller

P-DMA0
92 Channel

P-DMA1
44 Channel

M-DMA0
4 Channel

CRYPTO
AES, SHA, CRC,
TRNG, RSA,
ECC
Initiator/MMIO

SWJ/MTB/CTI
Arm
Cortex-M0+
100 MHz
MUL, NVIC, MPU

ROM
32 KB
ROM Controller

System Interconnect (Multi Layer AHB, IPC, MPU/SMPU)

PCLK
Peripheral Interconnect (MMIO, PPU)

IOSS GPIO

Prog.
Analog
SAR
ADC
(12-bit)
x3
SARMUX
64 ch

83x TCPWM
TIMER, CTR, QD, PWM

4x CXPI
CXPI Interface

12x LIN
LIN/UART

1x SCB
I2C, SPI, UART

7x SCB
I2C, SPI, UART

8x CANFD
CAN-FD Interface

EVTGEN
Event Generator

eFUSE
1024 bit

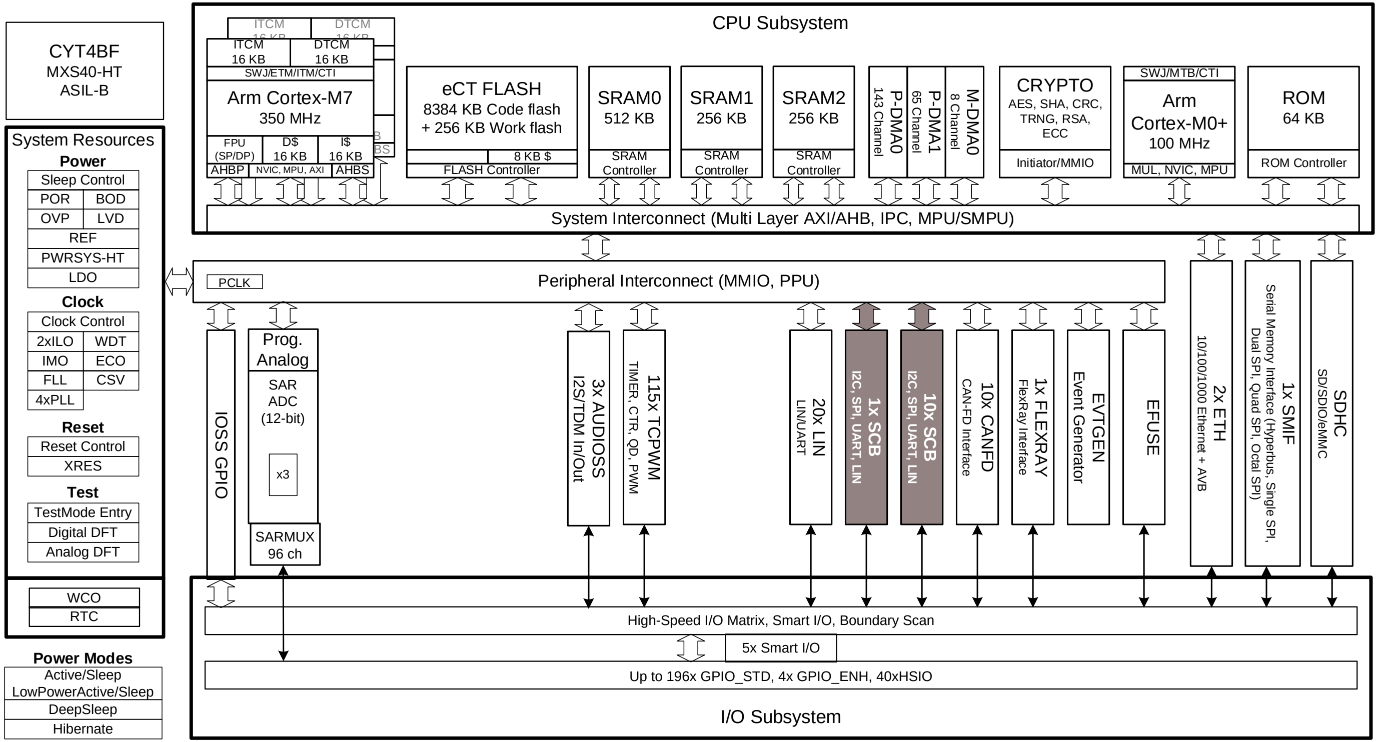
High-Speed I/O Matrix, Smart I/O, Boundary Scan
5x Smart I/O
Up to 148x GPIO_STD, 4x GPIO_ENH
I/O Subsystem

Hint Bar

Review TRM chapter 24 for additional details

Introduction to Traveo II Body Controller High

- SCB is part of the Peripheral blocks

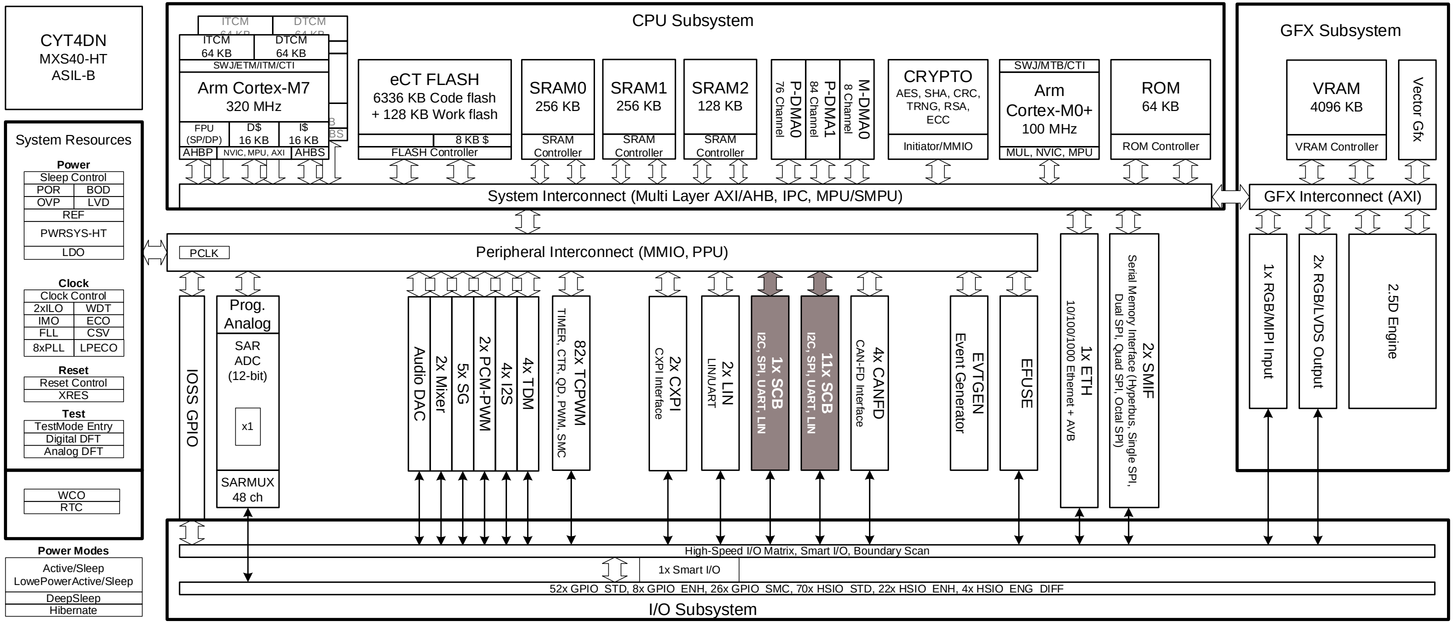


Hint Bar

Review TRM chapter 23 for additional details

Introduction to Traveo II Cluster

› SCB is part of the Peripheral blocks



Hint Bar

Review TRM chapter 23 for additional details

Serial Communication Block Overview (1/3)

- Supports three serial interface protocols
  - Serial Peripheral Interface (SPI)
  - Universal Asynchronous Receiver/Transmitter (UART)
  - Inter-Integrated Circuit (I^{2}C)
- SPI features
  - Supports master and slave functionality
  - Supports three SPI protocols (Motorola, Texas Instruments, National Semiconductor)
  - Supports up to four slave select lines
  - Programmable data frame size from 4 bits to 32 bits
  - Supports easy SPI (EZSPI) mode of operation

Hint Bar

Review TRM section 1 for additional details

Serial Communication Block Overview (2/3)

- UART Features
  - Supports two protocols
    - Standard UART
    - Multi-processor mode
  - Programmable data frame size from 4 to 16 bits
  - Programmable number of STOP bits (can set half-bit periods between 1 and 4)
  - Programmable oversampling

Hint Bar

Review TRM section 24.1 for additional details

Another dedicated LIN block can be used in Traveo II because this SCB has only standard LIN slave functionality

Serial Communication Block Overview (3/3)

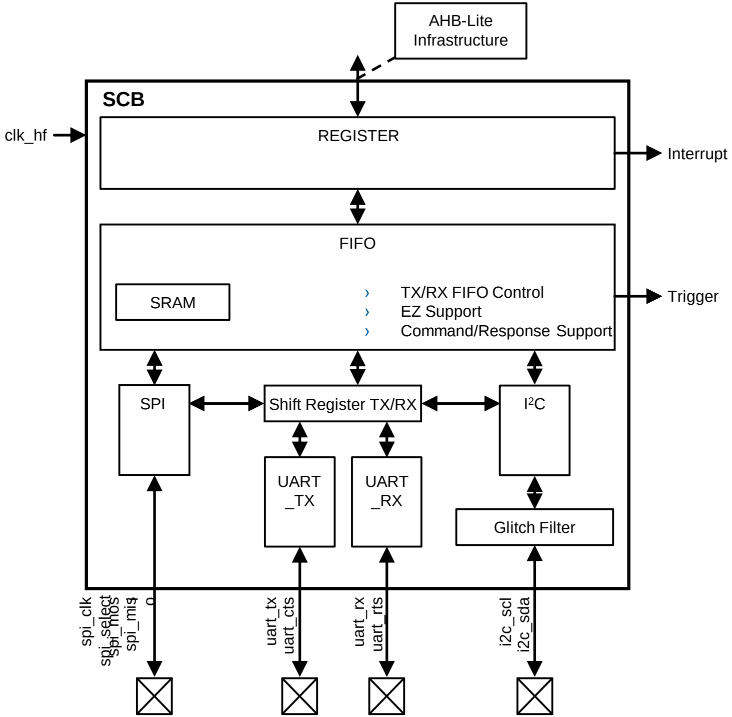
- I^{2}C Features
  - Master and slave mode
  - Four types of data-rate modes
    - Slow mode, Standard mode, Fast mode, Fast mode plus
  - 7-bit slave addressing
  - Analog and digital glitch filter – up to 50 ns

Hint Bar

Review TRM section 1 for additional details

Serial Communication Block Diagram

- Block Diagram

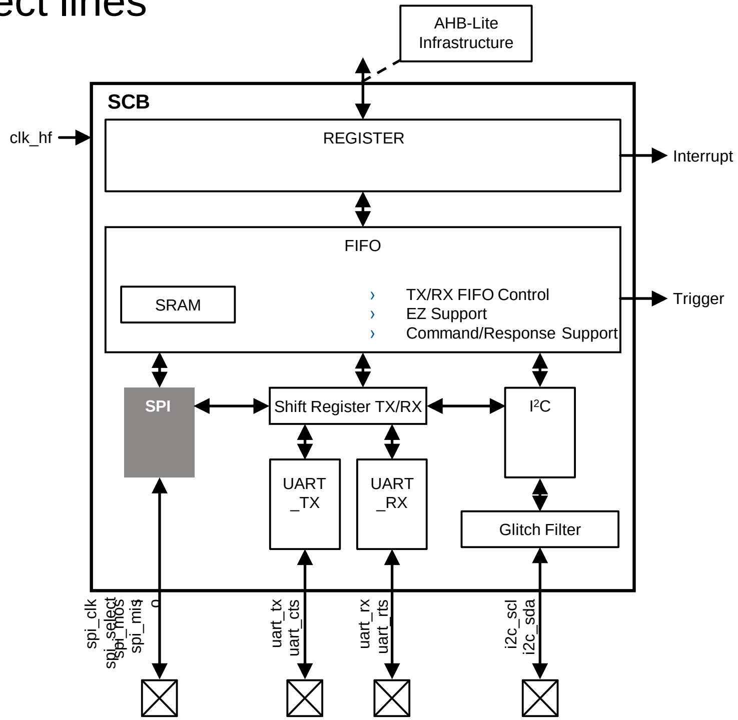


Hint Bar

Review TRM section 2 for additional details

Serial Communication Block Components

- SPI
  - Supports up to four slave select lines
  - Supports three SPI protocols
  - SPI Buffer mode
    - FIFO mode
    - EZSPI mode
    - Command-Response mode



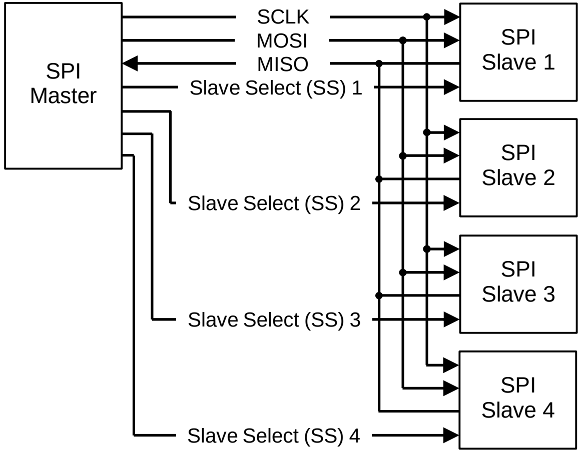
Hint Bar

Review TRM section 24.4 for additional details

SPI Slave Select Lines

- The standard SPI interface consists of four signals
  - SCLK: Serial clock (clock output from the master, input to the slave)
  - MOSI: Master-out-slave-in (data output from the master, input to the slave)
  - MISO: Master-in-slave-out (data input to the master, output from the slave)
  - Slave Select (SS): Typically an active-low signal (output from the master, input to the slave)

- Advantage
  - Can communicate with multiple external ICs using the same bus



Hint Bar

Review TRM section 4 for additional details

SPI Protocols

- Three SPI protocols[1]
  - The Motorola SPI[2] protocol has four clock modes
    - Data is driven and captured by clock polarity and clock phase
  - Texas Instruments SPI[2]
    - The Texas Instruments SPI protocol redefines the use of the SS signal to indicate the start of a data transfer
  - National Semiconductor SPI[3]
    - The National Semiconductor SPI protocol alternates transmission and reception

Hint Bar

Review TRM chapter 4 for additional details

[1] Refer to the Register TRM (SCBx_SPI_CTRL) for additional details
[2] Full Duplex
[3] Half Duplex

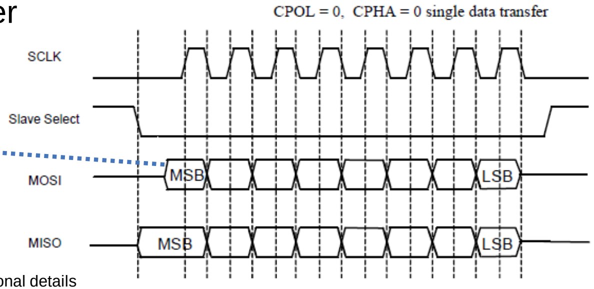
Motorola SPI Mode

- The Motorola SPI protocol has four clock modes based on how data is driven and captured on the MOSI and MISO lines
- The modes are determined by clock polarity (CPOL) and clock phase (CPHA)

Mode	CPOL	CPHA	Description
0	0	0	Data is driven on a falling edge of SCLK. Data is captured on a rising edge of SCLK.
1	0	1	Data is driven on a rising edge of SCLK. Data is captured on a falling edge of SCLK.
2	1	0	Data is driven on a rising edge of SCLK. Data is captured on a falling edge of SCLK.
3	1	1	Data is driven on a falling edge of SCLK. Data is captured on a rising edge of SCLK.

  - Motorola SPI data transfer example (Mode = 0)
    - A single 8-bit data transfer

Data Frame Size[1]
Programmable 4 to 32 bits



Legend:
CPOL: Clock Polarity
CPHA: Clock Phase
SCLK: SPI Interface Clock
MOSI: SPI Master-Out-Slave-In
MISO: SPI Master-In-Slave-Out

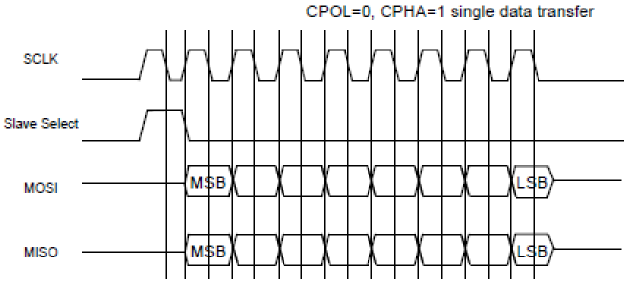
Hint Bar

Review TRM section 4.3.1 for additional details

[1] Refer to the Register TRM (SCBx_TX_CTRL, SCBx_RX_CTRL) for additional details

Texas Instruments SPI Mode

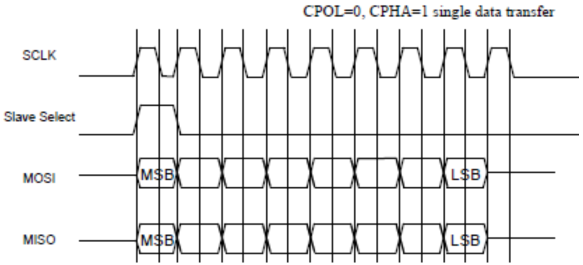
- Texas Instruments SPI data transfer[1]
  - A single 8-bit data transfer
  - The Slave Select pulse either precedes the first data bit or coincides with the first data bit of a frame
    - Slave Select pulse preceding example



CPOL=0, CPHA=1 single data transfer

Legend:
CPOL: Clock Polarity
CPHA: Clock Phase
SCLK: SPI Interface Clock
MOSI: SPI Master-Out-Slave-In
MISO: SPI Master-In-Slave-Out

    - Slave Select pulse coinciding example



CPOL=0, CPHA=1 single data transfer

Hint Bar

Review TRM section 4.3.2 for additional details

[1] Texas Instruments SPI mode uses only CPOL = 0 and CPHA = 1 settings

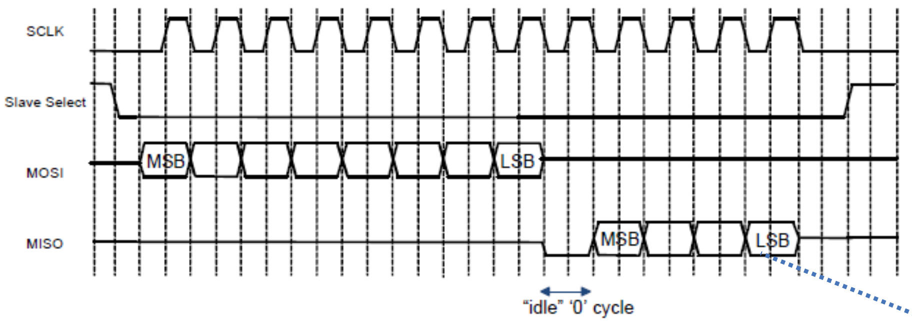
National Semiconductor SPI Mode

- National Semiconductor SPI data transfer[1]
  - The SPI protocol alternates transmission and reception
  - The transmission and reception data sizes are different
  - A single “idle” bit transfer period separates transmission from reception

Hint Bar

Review TRM section 4.3.3 for additional details

CPOL=0, CPHA=0 Transfer of one MOSI and one MISO data frame



Legend:
CPOL: Clock Polarity
CPHA: Clock Phase
SCLK: SPI Interface Clock
MOSI: SPI Master-Out-Slave-In
MISO: SPI Master-In-Slave-Out

Data Frame Size
Programmable 4 to 32 bits

[1] National Semiconductor SPI mode uses only CPOL = 0 and CPHA = 0 settings

SPI Buffer Modes (1/2)

- SPI can operate in three buffer modes[1]
  - FIFO mode
  - EZSPI mode
  - Command-Response mode
- Restrictions on using buffer mode
  - Master can only use FIFO mode
  - Slave can use all above three buffer modes. However, Command-Response mode is available only on DeepSleep-capable SCB
- SCB has an internal SRAM buffer
  - Capacity of 128 × 16 bits
  - Supports byte mode
  - Can be configured as 256 × 8 bits

Hint Bar

Review TRM section 4.4 for additional details

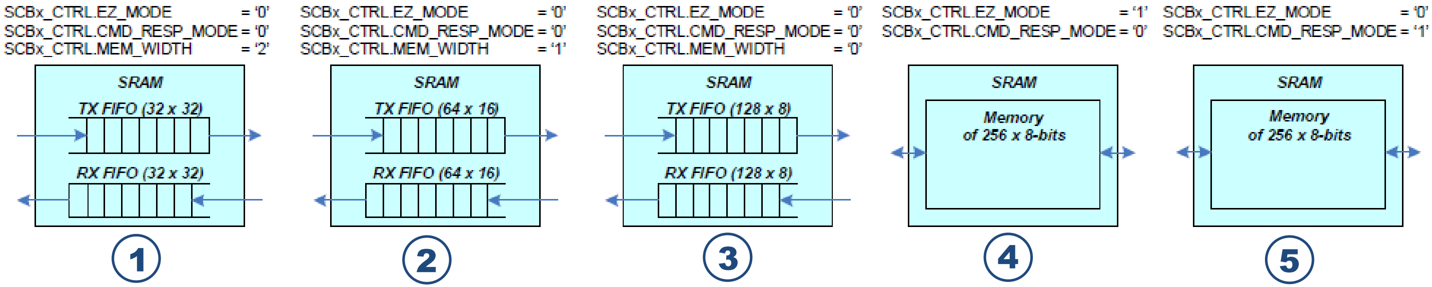
[1] Refer to the Register TRM (SCBx_CTRL.MODE) for additional details

SPI Buffer Modes (2/2)

- Buffer mode can be configured in five types in dedicated SRAM[1]
  1. Two 32 deep FIFOs for up to 32-bit data elements
  2. Two 64 deep FIFOs for up to 16-bit data elements
  3. Two 128 deep FIFOs for up to 8-bit data elements
  4. One 256 Byte EZ memory buffer
  5. One 256 Byte CMD_RESP memory buffer

Hint Bar

Review TRM section 3.1 for additional details



1:
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '0'
SCBx_CTRL.MEM_WIDTH = '2'

2:
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '0'
SCBx_CTRL.MEM_WIDTH = '1'

3:
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '0'
SCBx_CTRL.MEM_WIDTH = '0'

4:
SCBx_CTRL.EZ_MODE = '1'
SCBx_CTRL.CMD_RESP_MODE = '0'

5:
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '1'

[1] Refer to the Register TRM (SCBx_CTRL.EZ_MODE, SCBx_CTRL.CMD_RESP_MODE, SCBx_CTRL.MEM_WIDTH) for additional details

FIFO Mode

- FIFO mode includes TX and RX FIFOs
- Transmit and receive FIFO status[1] information indicates
  - Empty
  - Not empty
  - Full
  - Not full
- Provides “underflow” and “overflow” events
  - An “underflow” event is triggered by an attempt to read from an empty FIFO
  - An “overflow” event is triggered by an attempt to write to a full FIFO

Hint Bar

Review TRM section 4.4.1 for additional details

[1] Refer to the Register TRM (SCBx_INTR_TX, SCBx_INTR_RX) for additional details

EZSPI Mode

- The EZSPI[1] protocol is based on the Motorola SPI
  - The EZSPI mode is a method in which the master uses an 8-bit EZ address in the slave memory to write or read data
- It defines a single memory buffer with an 8-bit EZ address
- The EZ address indexes the buffer located on the slave device
- EZSPI has three types of transfers
  - EZ address written from the master to the slave
  - Data written from the master to an addressed slave memory location
  - Slave reads from an addressed slave memory location

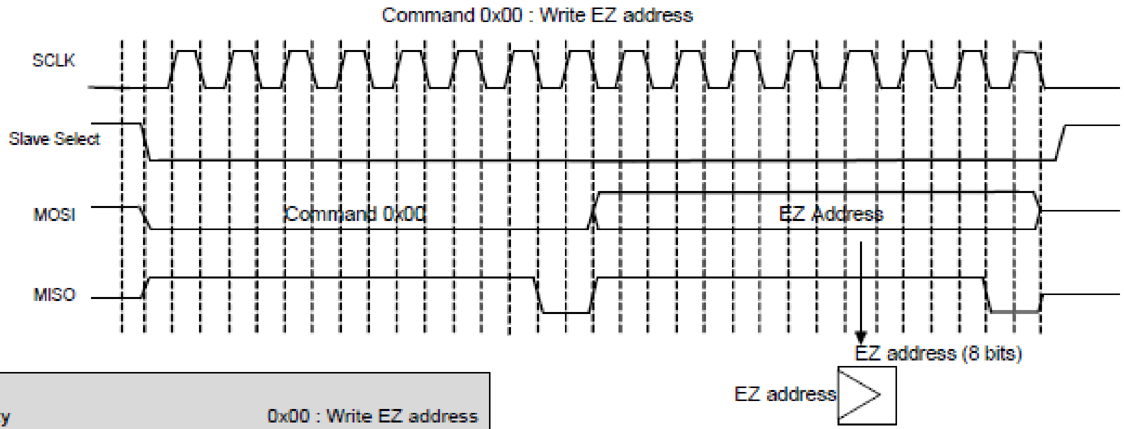
Hint Bar

Review TRM section 4.4.2 for additional details

[1] Refer to the Register TRM (SCBx_CTRL.EZ_MODE) for additional details

EZSPI Mode (Address Write)

- EZ Address Write
  - An EZ address write starts with a command byte (0x00) on the MOSI line indicating the master’s intent to write the EZ address



LEGEND :

CPOL : Clock Polarity
CPHA : Clock Phase
SCLK : SPI Interface Clock
MISO : SPI Master-In-Slave-Out
MOSI : SPI Master-Out-Slave-In

0x00 : Write EZ address
0x01 : Write DATA
0x02 : Read DATA
0xFE : "slave ready"
0xFF : "slave busy"

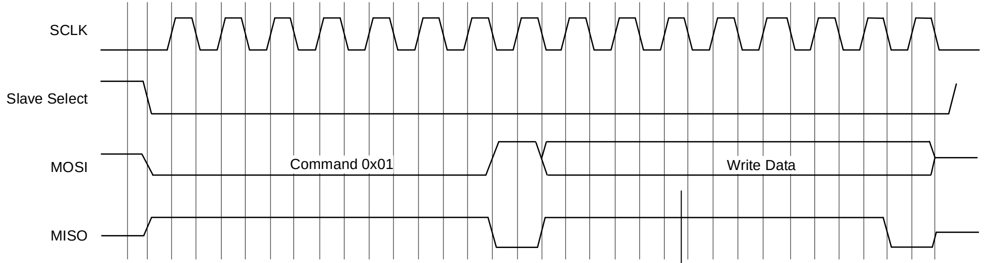
Hint Bar

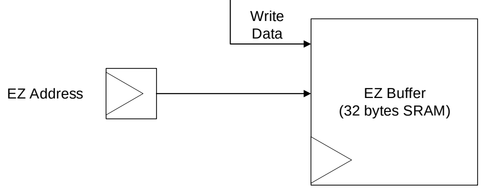
Review TRM section 4.4.2 for additional details

EZSPI Mode (Data Write)

- Memory Array Write
  - A write to a memory array index starts with a command byte (0x01) on the MOSI line indicating the master's intent to write to the memory array
  - The EZ address is automatically incremented by the slave as bytes are written into the memory array

Command 0x01: Write Data





LEGEND :
CPOL : Clock Polarity
CPHA : Clock Phase
SCLK : SPI Interface Clock
MISO : SPI Master-In-Slave-Out
MOSI : SPI Master-Out-Slave-In

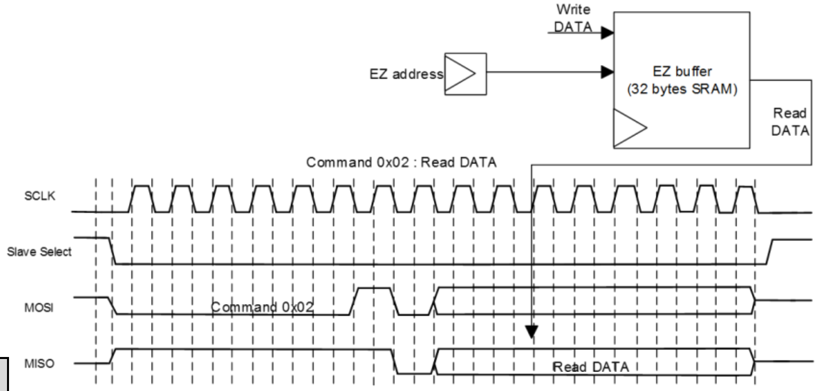
0x00 : Write EZ address
0x01 : Write DATA
0x02 : Read DATA
0xFE : "slave ready"
0xFF : "slave busy"

Hint Bar

Review TRM section 4.4.2 for additional details

EZSPI Mode (Data Read)

- Memory Array Read
  - A read from a memory array index starts with a command byte (0x02) on the MOSI line indicating the master’s intent to read from the memory array



LEGEND :
CPOL : Clock Polarity
CPHA : Clock Phase
SCLK : SPI Interface Clock
MISO : SPI Master-In-Slave-Out
MOSI : SPI Master-Out-Slave-In

0x00 : Write EZ address
0x01 : Write DATA
0x02 : Read DATA
0xFE : "slave ready"
0xFF : "slave busy"

- Advantage
  - Effective transmission and reception without using software to check the memory buffer address

Hint Bar

Review TRM section 4.4.2 for additional details

Command-Response Mode

- Only supports an SPI slave
- This mode has a single memory buffer, which is indexed using
  - Base read/write address
  - Current read/write address
- The command-response mode has two phases of operation
  - Write phase
  - Read phase
- The slave transmits either 0x62 (ready) or another value (busy)
- When disabled or reset, the slave transmits 0xFF (busy)

Hint Bar

Review TRM section 4.4.3 for additional details

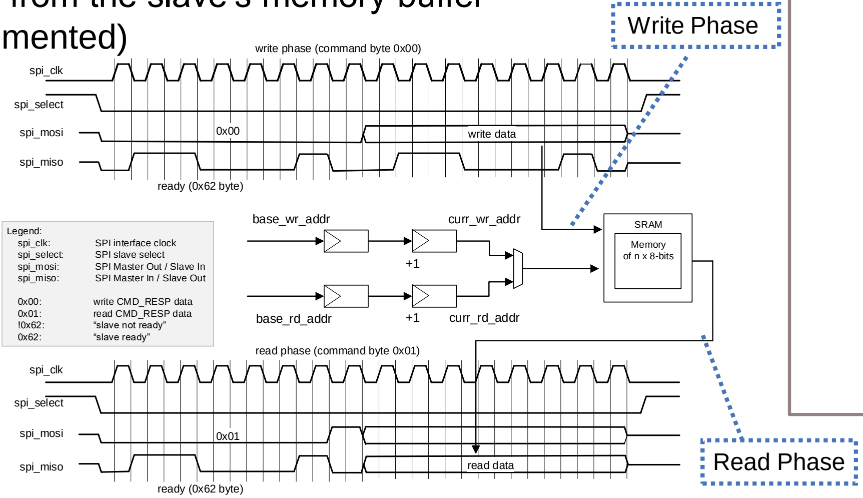
Command-Response Mode (Write/Read)

- The command-response write/read phases are
  - Write phase
    - Write phase begins with a selection byte (last bit set to ‘0’ indicating a write)
    - Master writes 8-bit data elements to the slave’s memory buffer (the current write address is incremented)
  - Read phase
    - Read phase begins with a selection byte (last bit set to ‘1’ indicating a read)
    - Master reads 8-bit data elements from the slave’s memory buffer (the current read address is incremented)

- Advantage
  - Effective transmission and reception without using software to check the memory buffer address

Hint Bar

Review TRM section 4.4.3 for additional details



Legend:

spi_clk:	SPI interface clock
spi_select:	SPI slave select
spi_mosi:	SPI Master Out / Slave In
spi_miso:	SPI Master In / Slave Out
0x00:	write CMD_RESP data
0x01:	read CMD_RESP data
!0x62:	“slave not ready”
0x62:	“slave ready”

Interrupt

› Interrupt Events in SPI Mode

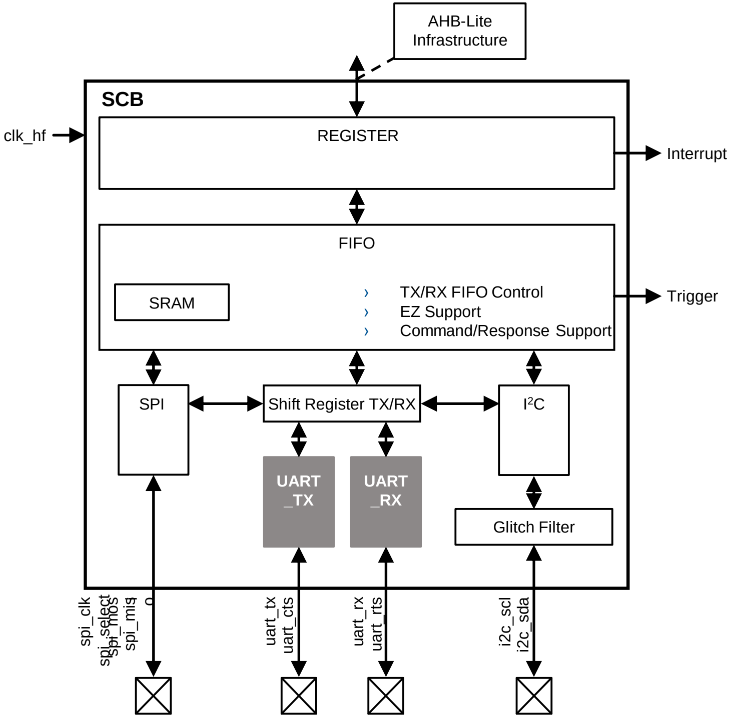
Event Type	Event
TX	SPI master transfer done
RX	SPI Bus Error
TX	TX FIFO is not full
TX	TX FIFO is empty
TX	TX FIFO overflows
TX	TX FIFO underflows
RX	RX FIFO is full
RX	RX FIFO is not empty
RX	RX FIFO overflows
RX	RX FIFO underflows

Hint Bar

Review TRM section 7.1 for additional details

UART Component Serial Communications Block

- The UART in the SCB block supports the following:
  - Standard UART
  - Multi-processor mode
  - Start skipping function
  - Oversampling

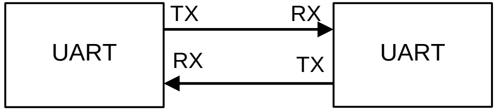


Hint Bar

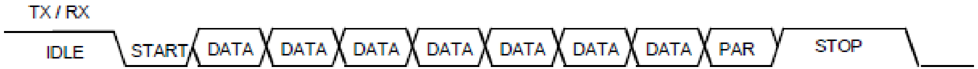
Review TRM section 24.5 for additional details

Standard UART

- Standard UART
  - TX and RX connection
  - Buffer mode only uses FIFO mode
- Typical UART transfer frame components
  - Start bit
  - Data bits (programmable from 4 to16 bits)
  - Parity bit (optional)
  - Stop bit (can be set to half-bit periods between 1 and 4)



data transfers (7data bits, 1 parity bit, 2 stop bits)



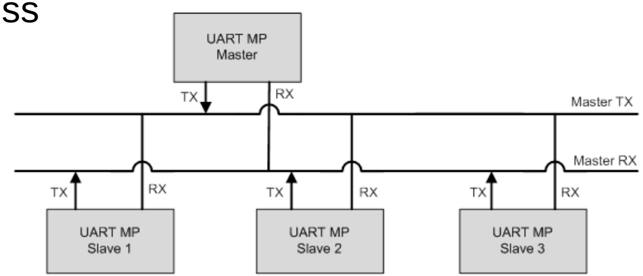
LEGEND:
TX / RX : Transmit or Receive line

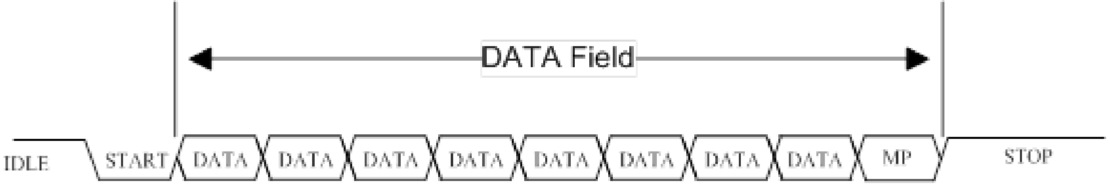
Hint Bar

Review TRM section 5.3.1 for additional details

Multi-Processor Mode

- UART_MP (Multi-Processor) mode
  - Single-master-multi-slave connection
- UART_MP mode properties
  - Each slave is identified by a unique address
  - Uses a 9-bit data field
    - Ninth bit as address/data flag (MP bit)
  - Parity bit is disabled





- Advantage
  - Can communicate with multiple external ICs using the same bus

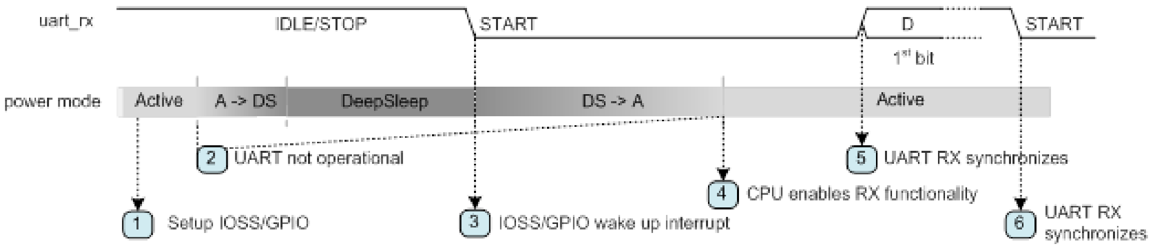
Hint Bar

Review TRM section 5.3.2 for additional details

Start Skipping Function

- Use Start Skipping to wake up from DeepSleep mode

1. The GPIO is set as an interrupt to UART RX line transition ‘1’ to ‘0’ (START bit)
2. The UART receiver is not functional during DeepSleep mode transition
3. When the GPIO interrupt is activated, the system transits from DeepSleep to Active mode
4. The CPU enables UART receive functionality
5. The UART receiver synchronizes with the receipt data frame
6. The UART receiver proceeds with normal operation



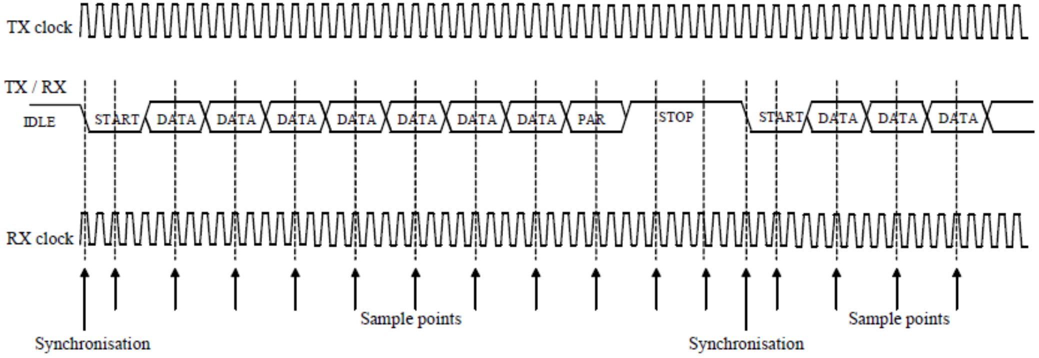
- Advantage
  - UART can receive data immediately upon MCU wakeup from DeepSleep

Hint Bar

Review TRM section 5.3.1 for additional details

Oversampling

- The receiver oversamples the incoming signal
- The sampling clock can select from 8 to 16 clock cycles by software
- The value of the sample point[1] in the middle of the bit transfer period is used for transmitter and receiver clock synchronization



LEGEND:
TX / RX : Transmit or Receive line

- Advantage
  - Can be used to synchronize the receiver with the transmitter clock

Hint Bar

Review TRM section 5.4 for additional details

[1] Alternatively, three samples around the middle of the bit transfer period are used for increased accuracy

Interrupt

› Interrupt events in UART mode

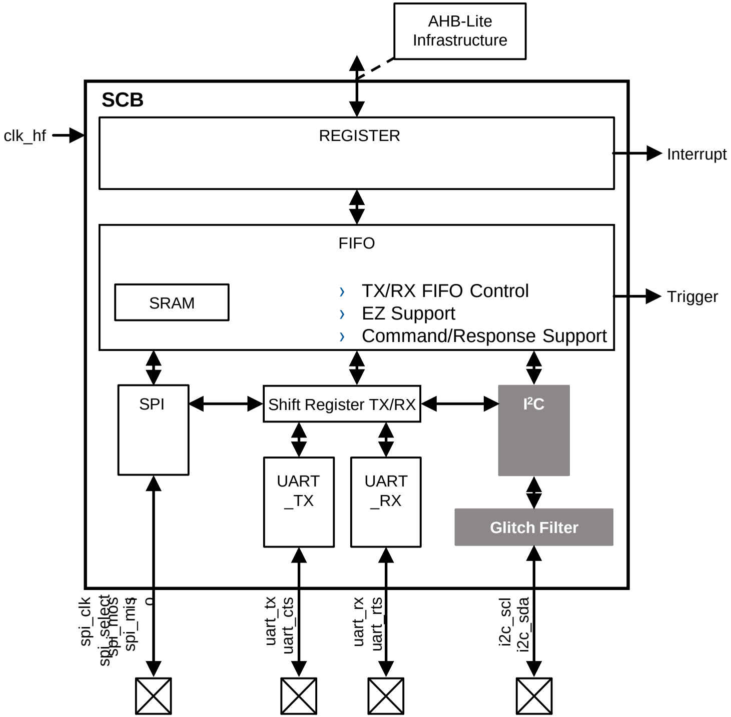
Event Type	Event
TX	TX done
RX	Frame error in received data frame
RX	Parity error in received data frame
RX	LIN baud rate detection is completed
RX	LIN break detection is successful
TX	TX FIFO is not full
TX	TX FIFO is empty
TX	TX FIFO overflows
TX	TX FIFO underflows
RX	RX FIFO is full
RX	RX FIFO is not empty
RX	RX FIFO overflows
RX	RX FIFO underflows

Hint Bar

Review TRM section 7.2 for additional details

I^{2}C Component in Serial Communications Block

- I^{2}C supports the following:
  - Standard I^{2}C
  - I^{2}C Buffer mode
    - FIFO mode[1]
    - EZI2C mode
    - Command-Response mode
  - Glitch filtering



Hint Bar

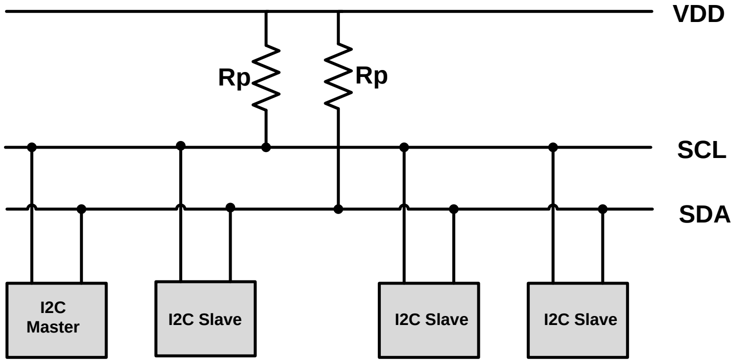
Review TRM section 6 for additional details

Refer to FIFO mode

[1] FIFO mode has the same features as SPI

Standard I^{2}C

- The standard I^{2}C bus is a two-wire interface with these lines
  - Serial Data (SDA)
  - Serial Clock (SCL)
- I^{2}C devices are connected to these lines through pull-up
- Masters and slaves can operate as either transmitter or receiver



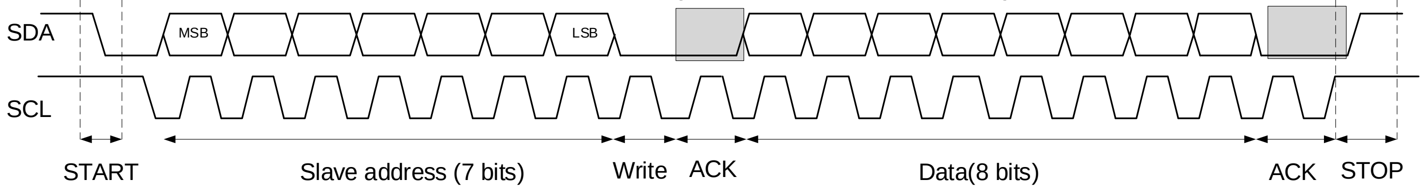
Hint Bar

Review TRM section 6 for additional details

Write Transfer

- The master sends data to the slave via a typical write transfer
  - Generate START condition
  - 7-bit I^{2}C slave address and a write indicator ‘0’ bit
  - The addressed slave transmits an acknowledgment
  - If the master receives an acknowledgment, it transmits data (8 bits) to the bus
  - When the transfer is complete, the master generates a STOP condition

Write data transfer (Master writes the data)



LEGEND :

SDA: Serial Data Line

SCL: Serial Clock Line (always driven by the master)

▒ Slave Transmit / Master Receive

Hint Bar

Review TRM section 6.4.1 for additional details

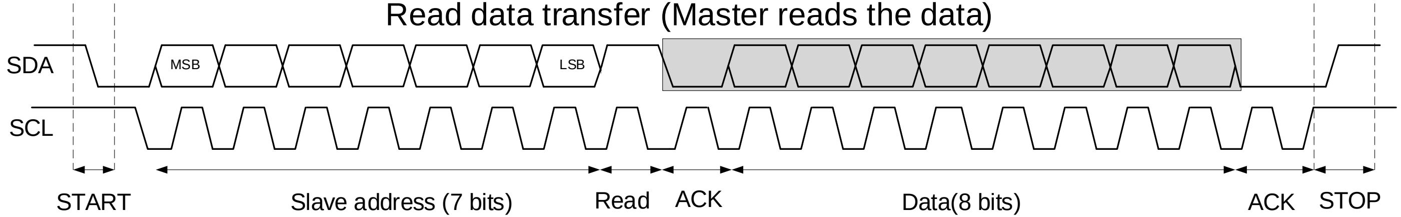
Read Transfer

- The master receives data from the slave via a typical read transfer
  - Generate START condition
  - 7-bit I^{2}C slave address and a read indicator ‘1’ bit
  - If the slave acknowledges the acknowledgment signal from the master, it starts transmitting data after the acknowledgment signal
  - The master transmits an acknowledgment to confirm
  - When the transfer is complete, the master generates a STOP condition

Hint Bar

Review TRM section 6.4.2 for additional details

Read data transfer (Master reads the data)



LEGEND :	
SDA:	Serial Data Line
SCL:	Serial Clock Line(always driven by the master)
▨	Slave Transmit / Master Receive

I^{2}C Buffer Modes (1/2)

- I^{2}C can operate in three buffer modes[1]
  - FIFO mode[2]
  - EZI2C mode
  - Command-Response mode
- Restrictions on using buffer mode[3]
  - Master can only use FIFO mode
  - Slave can use all above three buffer modes. However, Command-Response mode is available only on DeepSleep-capable SCB
- SCB has an internal SRAM buffer[3]
  - Capacity of 128 x 16 bits
  - Supports byte mode
  - Can be configured as 256 × 8 bits

Hint Bar

Review TRM section 6.5 for additional details

Refer to FIFO mode

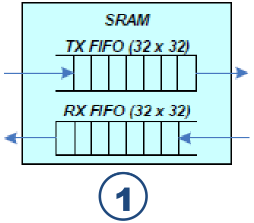
[1] Refer to the Register TRM (SCBx_CTRL.MODE) for additional details
[2] FIFO mode has the same features as SPI
[3] These features are the same as the SPI Buffer mode

I²C Buffer Modes (2/2)

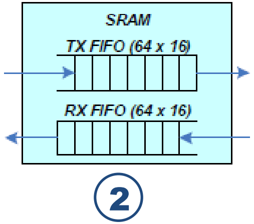
› Buffer mode can be configured in five types in dedicated SRAM[1]

1. Two 32 deep FIFOs for up to 32-bit data elements
2. Two 64 deep FIFOs for up to 16-bit data elements
3. Two 128 deep FIFOs for up to 8-bit data elements
4. One 256 Byte EZ memory buffer
5. One 256 Byte CMD_RESP memory buffer

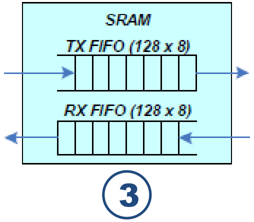
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '0'
SCBx_CTRL.MEM_WIDTH = '2'



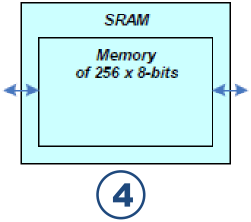
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '0'
SCBx_CTRL.MEM_WIDTH = '1'



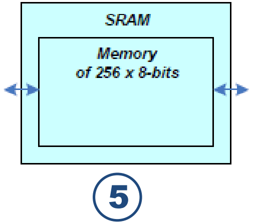
SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '0'
SCBx_CTRL.MEM_WIDTH = '0'



SCBx_CTRL.EZ_MODE = '1'
SCBx_CTRL.CMD_RESP_MODE = '0'



SCBx_CTRL.EZ_MODE = '0'
SCBx_CTRL.CMD_RESP_MODE = '1'



Hint Bar

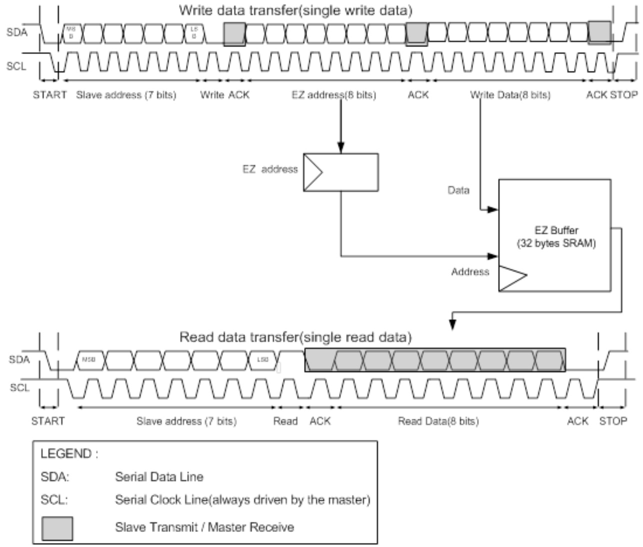
Review TRM section 3.1 for additional details

[1] Refer to the Register TRM (SCBx_CTRL.EZ_MODE, SCBx_CTRL.CMD_RESP_MODE, SCBx_CTRL.MEM_WIDTH) for additional details. These features are the same as the SPI Buffer mode.

EZI2C Mode

- EZI2C distinguishes three operation phases
  - Address phase[1]
    - The master transmits an 8-bit EZ address to the slave
  - Write phase[1]
    - The master writes 8-bit data elements to the slave’s memory buffer
  - Read phase[1]
    - The master reads 8-bit data elements from the slave’s memory buffer

- Advantage
  - Effective transmission and reception without using software to check the memory buffer address



Hint Bar

Review TRM section 6.5.2 for additional details

[1] These operations are the same as the EZ SPI mode

Command-Response Mode

- Only supports an I^{2}C slave
- This mode has a single memory buffer, which is indexed using[1]
  - Base read/write address
  - Current read/write address
- The command-response mode has two phases of operation
  - Write phase
  - Read phase

Hint Bar

Review TRM section 6.5.3 for additional details

[1] The features are same as the SPI Command response mode

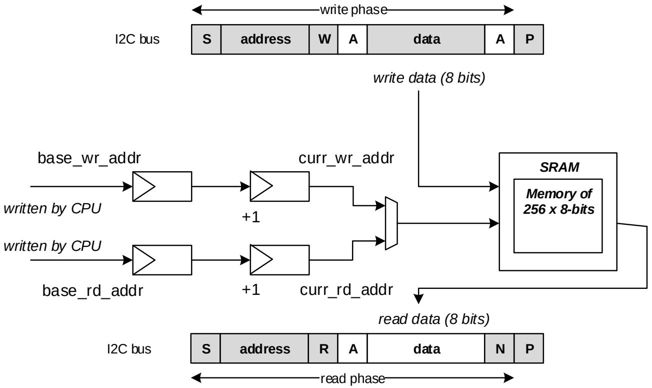
Command-Response Mode (Write/Read)

- The command-response write/read phases are
  - Write phase[1]
    - Begins with a START bit, slave address, and write bit (set to ‘0’ indicating a write)
    - The master writes 8-bit data elements to the slave’s memory buffer (the current write address is incremented)
  - Read phase[1]
    - Begins with a START bit, slave address, and read bit (set to ‘1’ indicating a read)
    - The master reads 8-bit data elements from the slave’s memory buffer (the current read address is incremented)

- Advantage
  - Effective transmission and reception without using software to check command

LEGEND:

S:	Start
RS:	Repeated start
P:	Stop
A:	ACK
N:	NACK



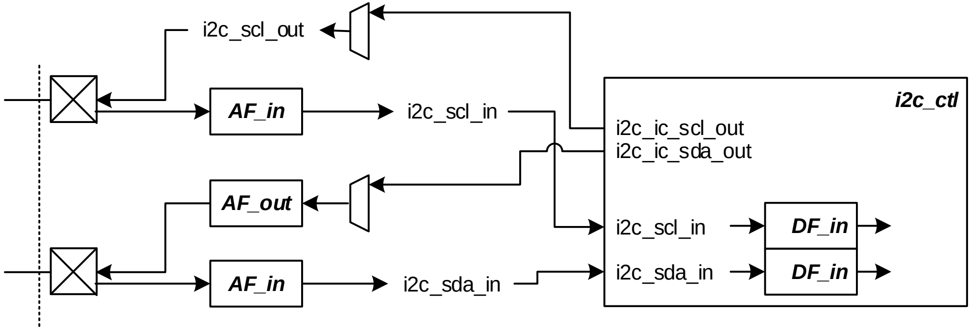
Hint Bar

Review TRM section 6.5.3 for additional details

[1] The operation is the same as the SPI Command response mode

Glitch Filtering

- The Traveo II SCB I^{2}C has analog and digital glitch filters
- Analog glitch filters (AF_in, AF_out) are applied in the following:
  - i2c_scl_in
  - i2c_sda_in
  - i2c_sda_out
- Analog glitch filters are applied to filter glitches of up to 50 ns
- Digital glitch filters (DF_in) are applied in the following:
  - i2c_scl_in
  - i2c_sda_in
- Digital glitch filters are 3-tap median filters



Hint Bar

Review TRM section 6.6.1 for additional details

Interrupt

- Interrupt events in I^{2}C mode

Event Type	Event
TX	I^{2}C master lost arbitration
RX	I^{2}C master received NACK
RX	I^{2}C master received ACK
TX	I^{2}C master sent STOP
RX	I^{2}C master bus error (unexpected stop/start condition detected)
TX	I^{2}C slave lost arbitration
RX	I^{2}C slave received NACK
RX	I^{2}C slave received ACK
RX	I^{2}C slave received STOP
RX	I^{2}C slave received START

Event Type	Event
RX	I^{2}C slave address matched
RX	I^{2}C slave bus error (unexpected stop/start condition detected)
TX	TX FIFO is not full
TX	TX FIFO is empty
TX	TX FIFO overflows
TX	TX FIFO underflows
RX	RX FIFO is full
RX	RX FIFO is not empty
RX	RX FIFO overflows
RX	RX FIFO underflows

Hint Bar

Review TRM section 7.3 for additional details

Part of your life. Part of tomorrow.

Revision History

Revision	ECN	Submission Date	Description of Change
**	6178141	05/21/2018	Initial release
*A	6415893	12/10/2018	Added slides 2, 4, 5, 8, and the note descriptions on all pages. Updated slides 3 and 7.
*B	6668243	09/04/2019	Updated slides 2, 3, 4, 6, 7, 8, 11,18 and 21. Added slide 5.
*C	6702599	10/16/2019	Corrected spec footer revision.
*D	7062561	01/08/2021	Updated slides 2, 3, 9, 10, 12, 13, 16, 18, 19, 26, 27, 32 and 36. Added slide 17 and 37.