Serial Communications

Last updated 4/29/20

Serial Communications

Serial Peripheral Interface

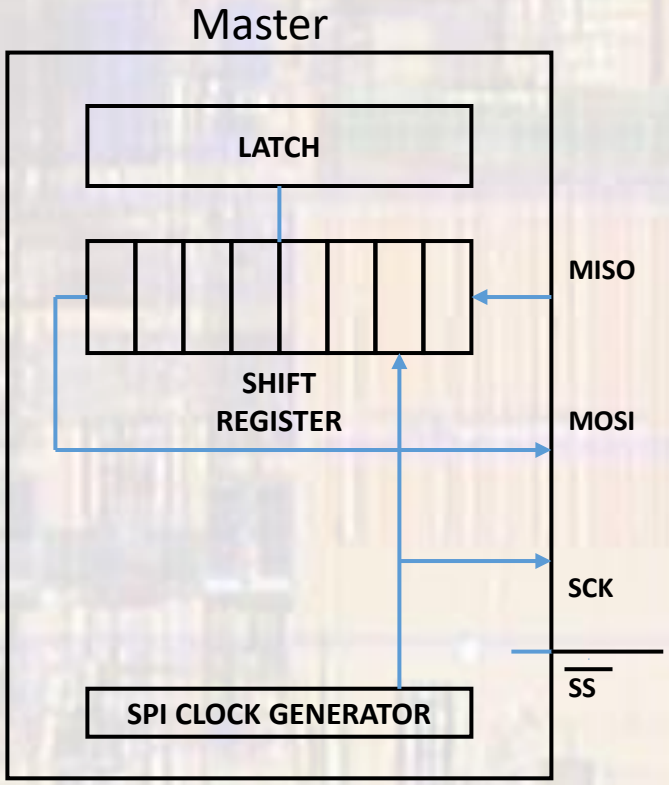
SPI

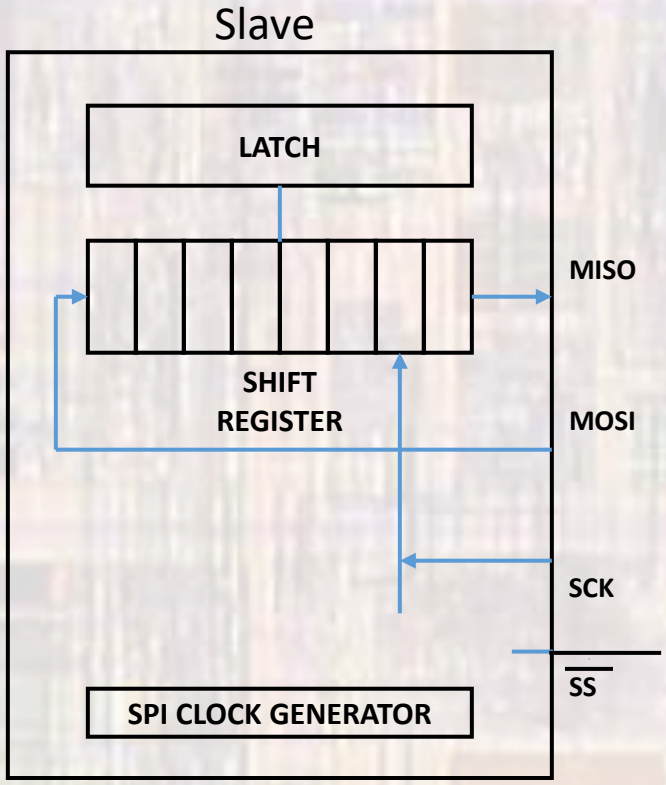
Serial Communications

- SPI - Overview
  - 8 bit synchronous shift register used to communicate externally
  - Most often used to communicate with peripherals
    - displays, sensors, converters
  - Can be used for inter-processor communication
  - Two modes of operation
    - Master – responsible for providing the clock
    - Slave – receives clock from the master

Serial Communications

- SPI Overview





MISO – Master:IN or Slave:OUT

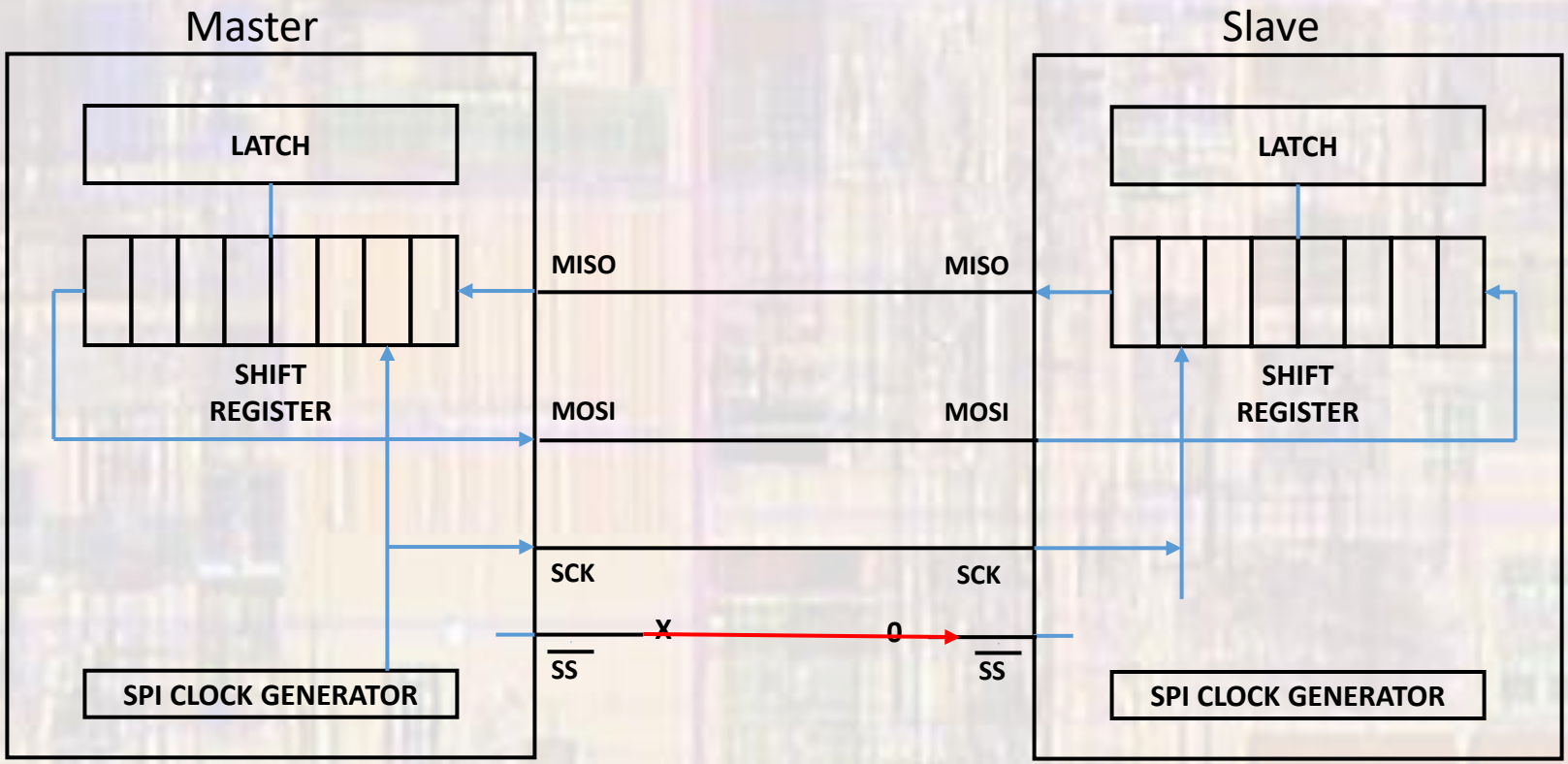
MOSI – Master:OUT or Slave:IN

SCK – SPI CLK

$\overline{SS}$ – Slave Select Bar

Serial Communications

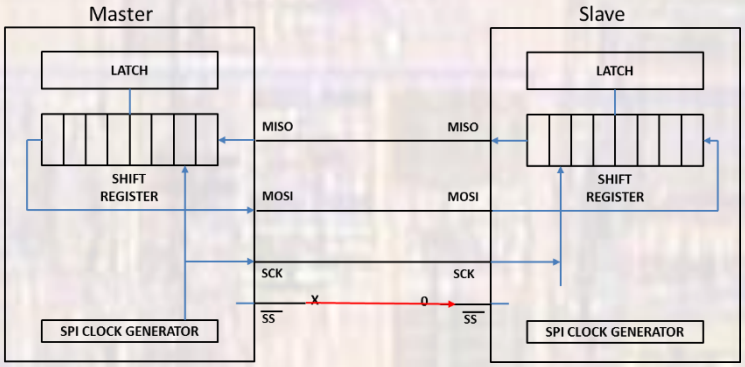
- SPI Operation



Latch → Shift Register in both master and slave
Master generates 8 clocks → shifts both registers (swaps content)
Shift Register → Latch in both master and slave

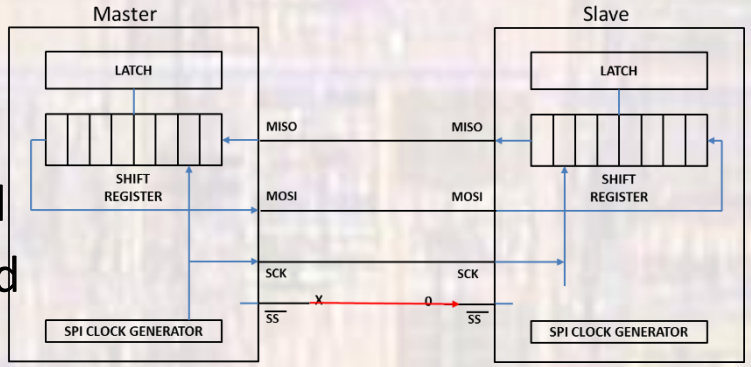
Serial Communications

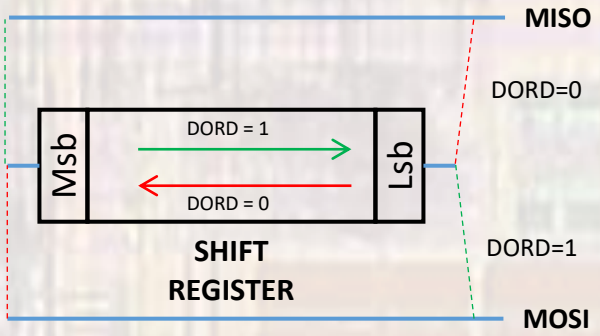
- SPI Operation
  - Configure 1 device as master
  - Configure 1 or more devices as slaves
  - Load values into register(s)
  - Pull SSbar low on the desired slave device
  - Initiate transfer by writing to the data register
    - The master will generate the appropriate clocks
  - If interrupts are enabled – an interrupt will be generated on completion



Serial Communications

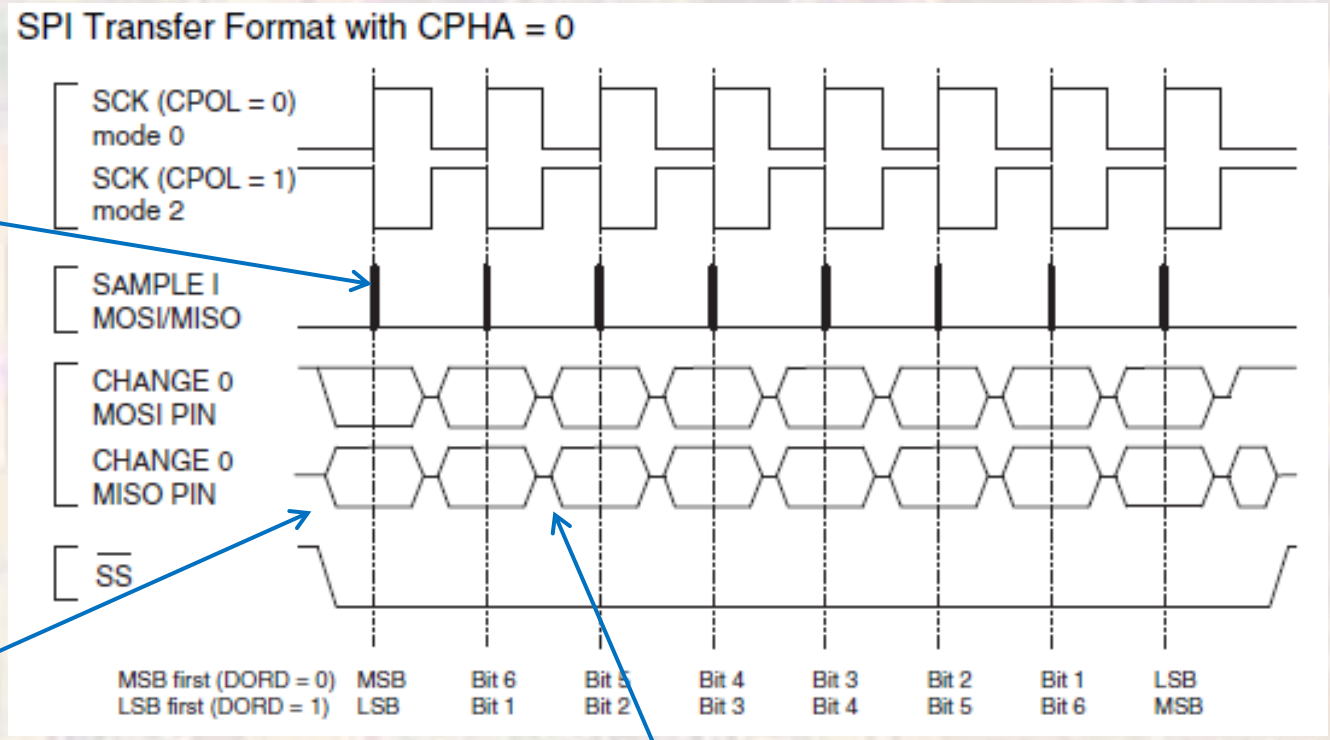
- SPI Operation
  - 2 options for clock polarity
    - CPOL = 0 → rising edge triggered
    - CPOL = 1 → falling edge triggered
  - 2 options for clock phase
    - CPHA = 0 → leading edge triggered
    - CPHA = 1 → trailing edge triggered
  - 2 options on transfer direction
    - DORD = 0 → MSB transferred first
    - DORD = 1 → LSB transferred first





Serial Communications

- SPI Operation
  - CPHA = 0



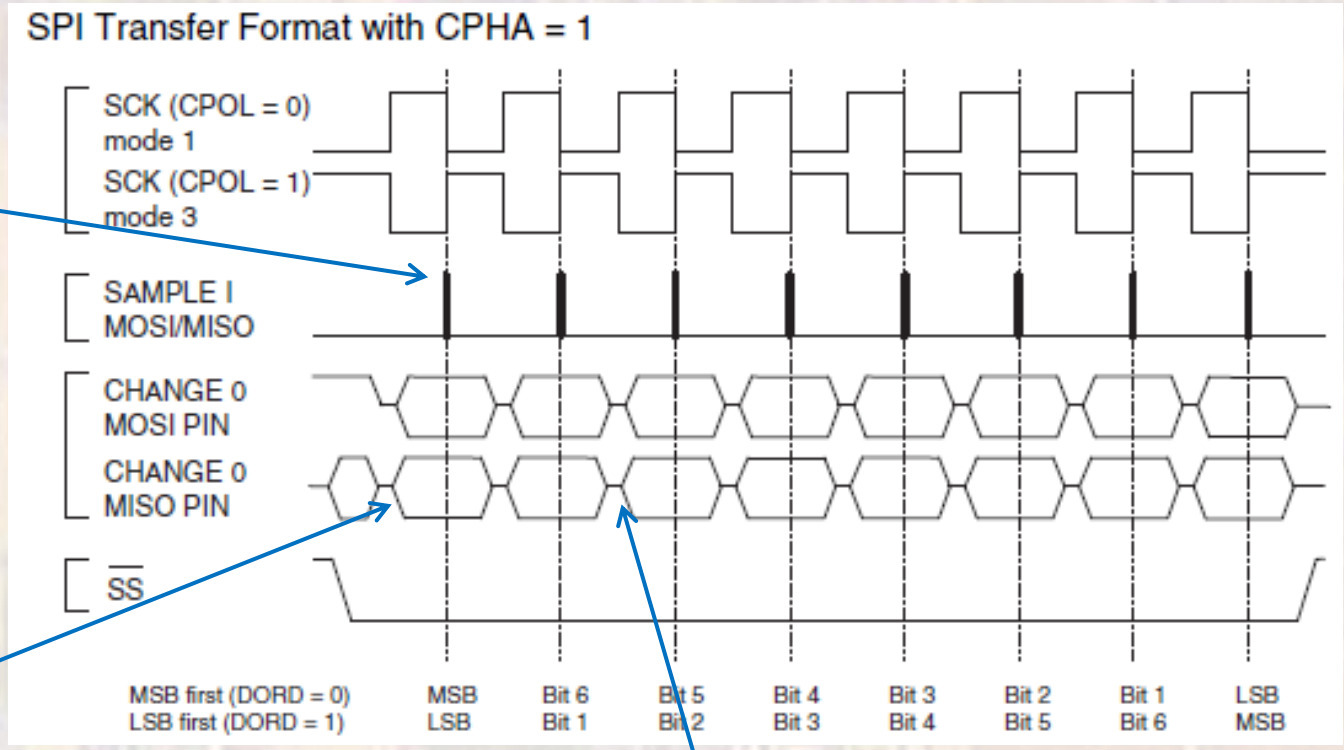
Captured in register on leading clock edge

Values active on pins as soon as SSbar goes low

New values placed on pins on trailing clock edge

Serial Communications

- SPI Operation
  - CPHA = 1



Captured in register on trailing clock edge

Values active on pins on first clock edge

New values placed on pins on leading clock edge

Serial Communications

- SPI Operation
  - Multiple Slave Configuration

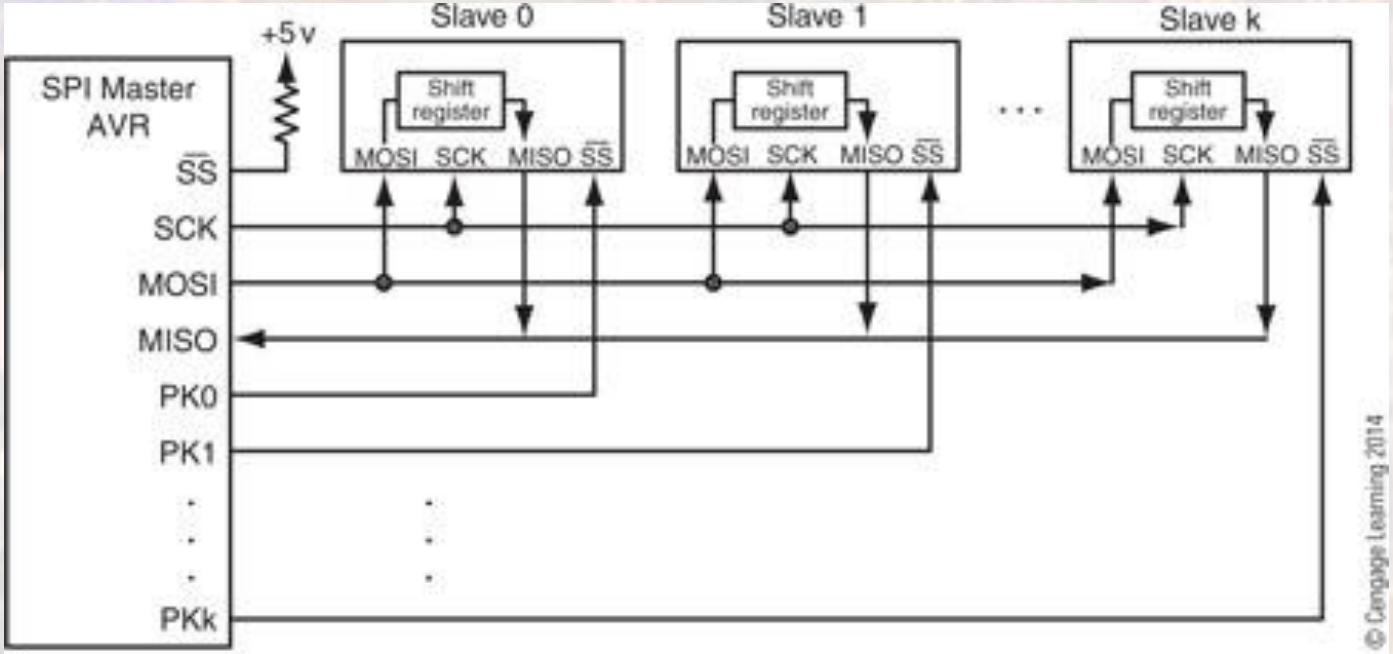


Figure 14.7 ■ Single-master and multiple-slave device connection (method 1)

Serial Communications

- SPI Operation
  - Multiple Slave – Extended Shift Configuration

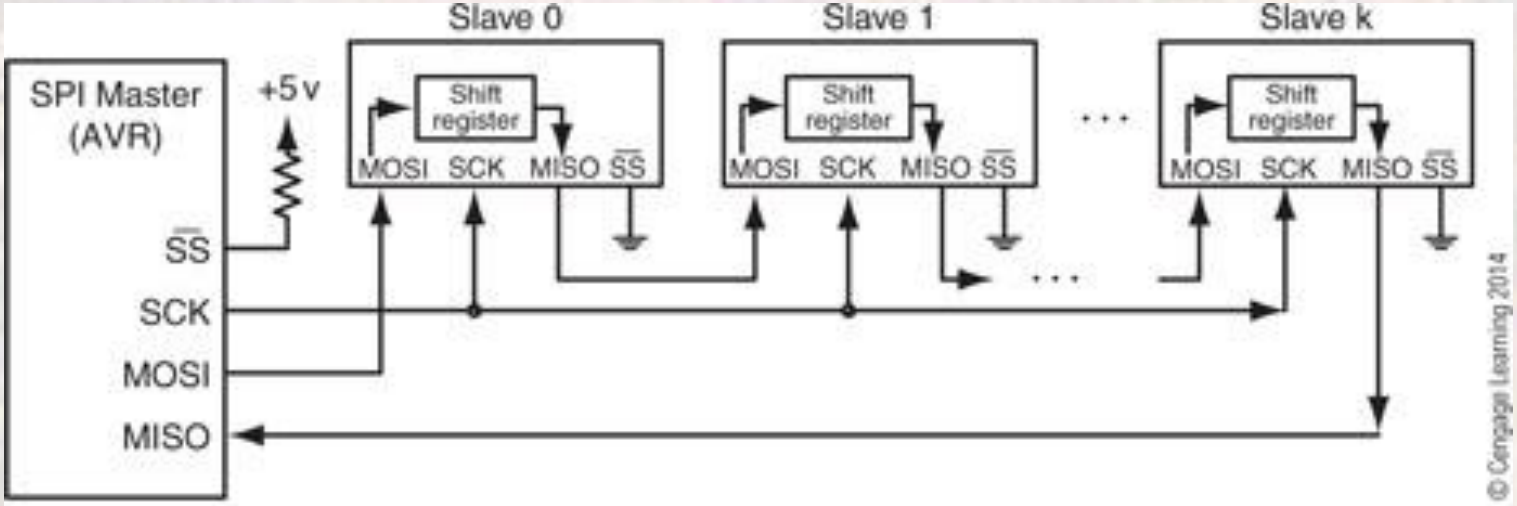


Figure 14.8 ■ Single-master and multiple-slave device connection (method 2)

Serial Communications

- SPI Implementation
  - Separate RX/TX registers
  - Separate RX/TX buffers
  - State Machine
  - Clock divider
  - Control register
  - Status register

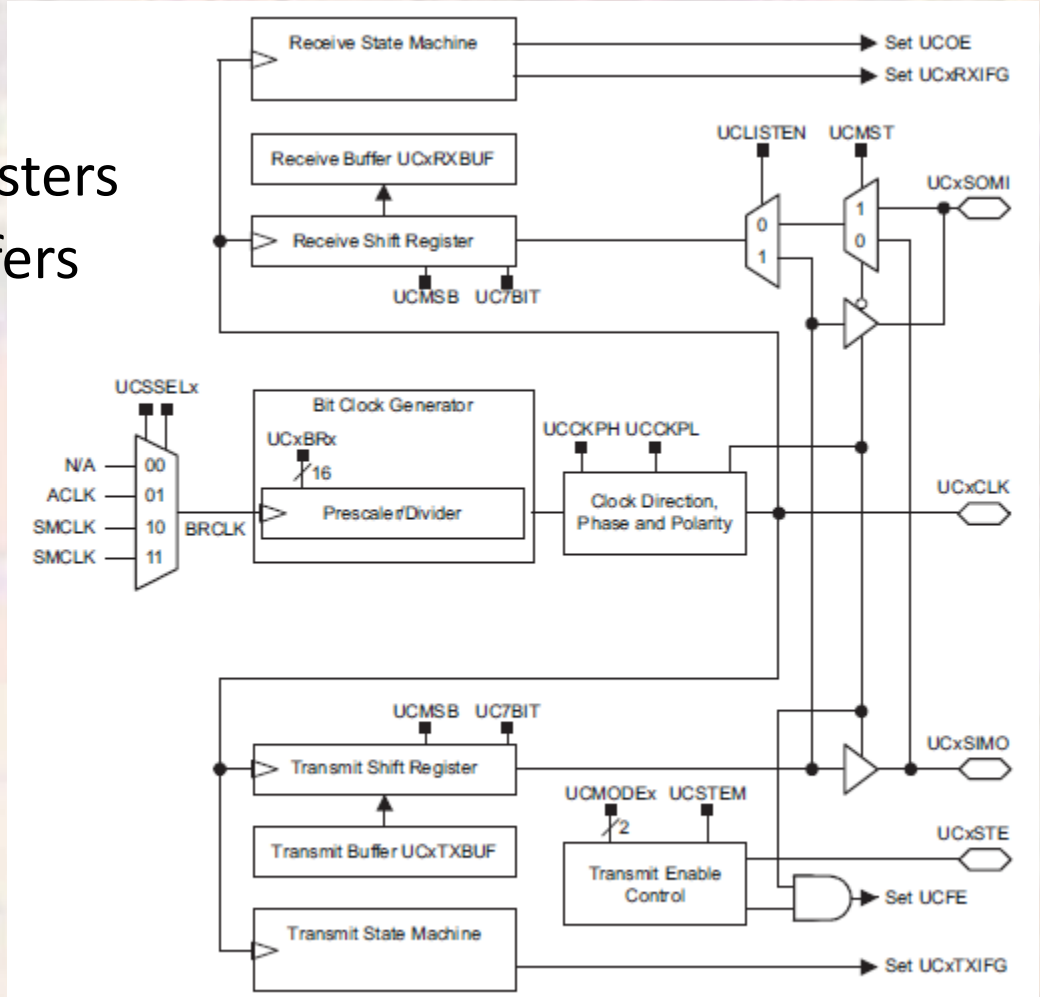


Figure 23-1. eUSCI Block Diagram – SPI Mode

Serial Communications

- SPI - Master Mode
  - Receive/Transmit initiated by writing to TX buffer
  - Data is moved to the TX shift register as soon as it is empty – UCTXIFG flag set – transfer is NOT complete
  - Transfer is controlled by the state machine
  - When complete RX data moved to the RX buffer – UCRXIFG flag is set – transfer is complete

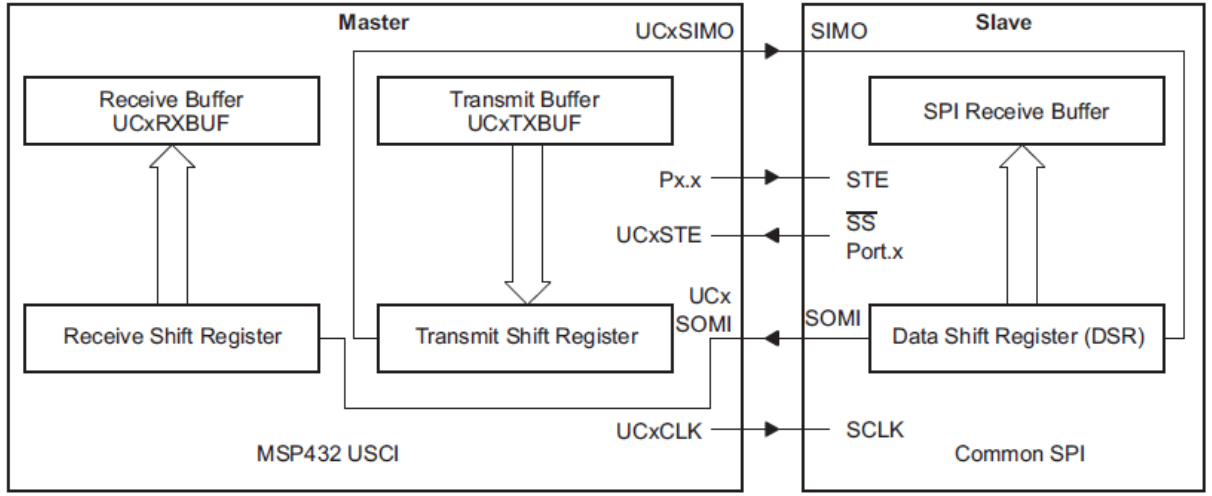


Figure 23-2. eUSCI Master and External Slave (UCSTEM = 0)

Serial Communications

- SPI - Slave Mode
  - External clock controls the transfer
  - Data is moved to the TX shift register as soon as it is empty – UCTXIFG flag set – transfer is NOT complete
  - When complete, RX data moved to the RX buffer – UCRXIFG flag is set
  - UCOE flag is set if the RX buffer has not been read prior to a new completed transfer

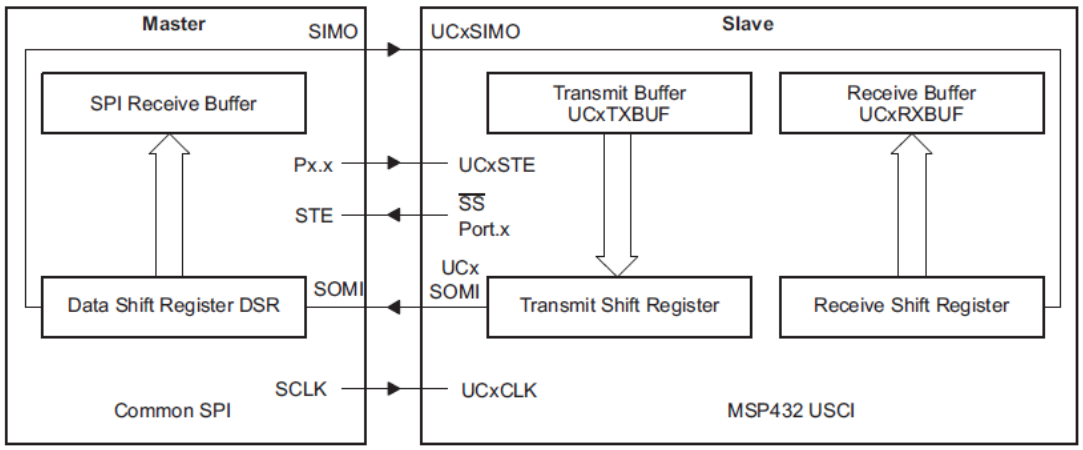


Figure 23-3. eUSCI Slave and External Master

Serial Communications

Two Wire Interface

TWI

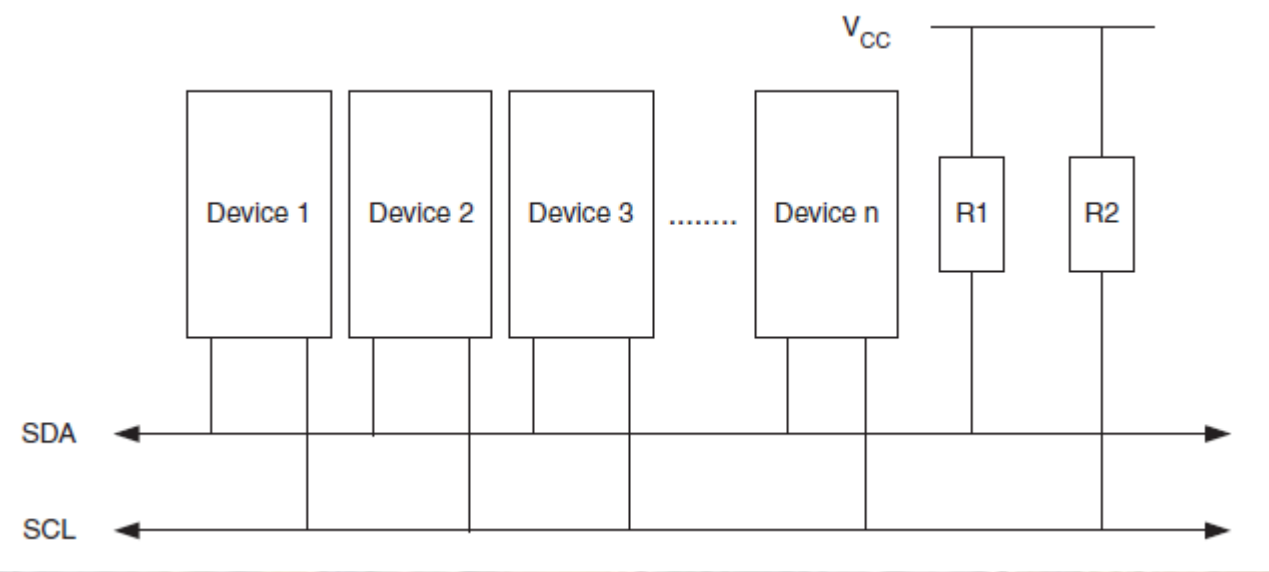
I^2C

Serial Communications

- TWI - Overview
  - 8 bit synchronous shift register used to communicate externally
    - 9 bit total communication packet
    - uni-directional
  - Most often used to communicate with peripherals
    - displays, sensors, converters
  - Supports multiple masters and multiple slaves
  - 4 modes of operation
    - Master Receive
    - Master Transmit
    - Slave Receive
    - Slave Transmit

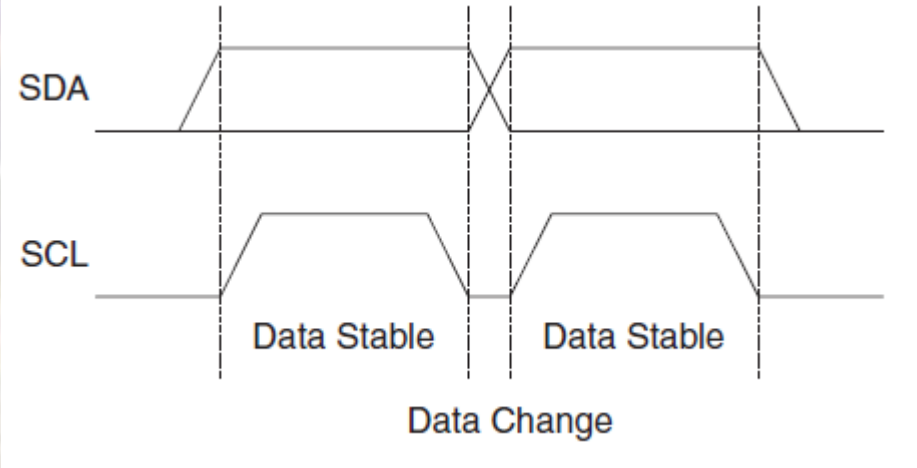
Serial Communications

- TWI Overview
  - Open drain configuration
    - outputs only pull down
    - pull up resistors or current sources pull up



Serial Communications

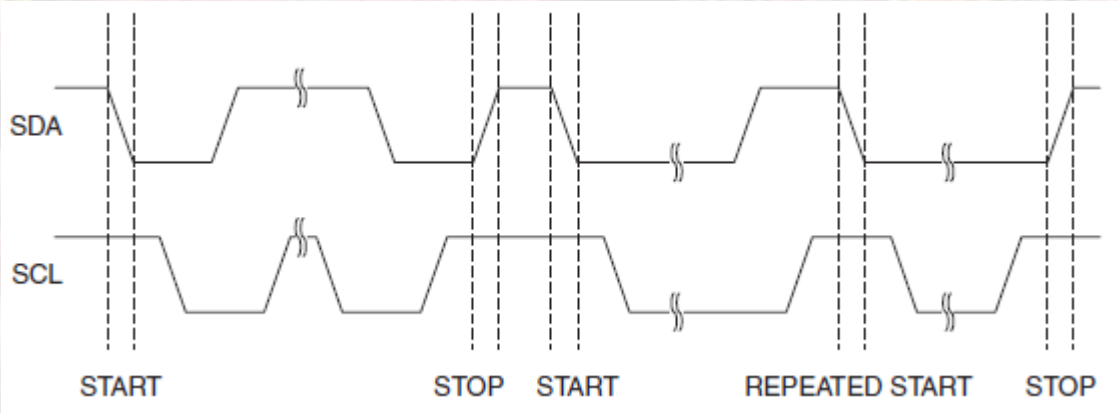
- TWI Timing
  - SDA – data line
  - SCL – clock line
  - Data must be valid during the entire positive clock cycle time



Note: data changes occur during SCL low

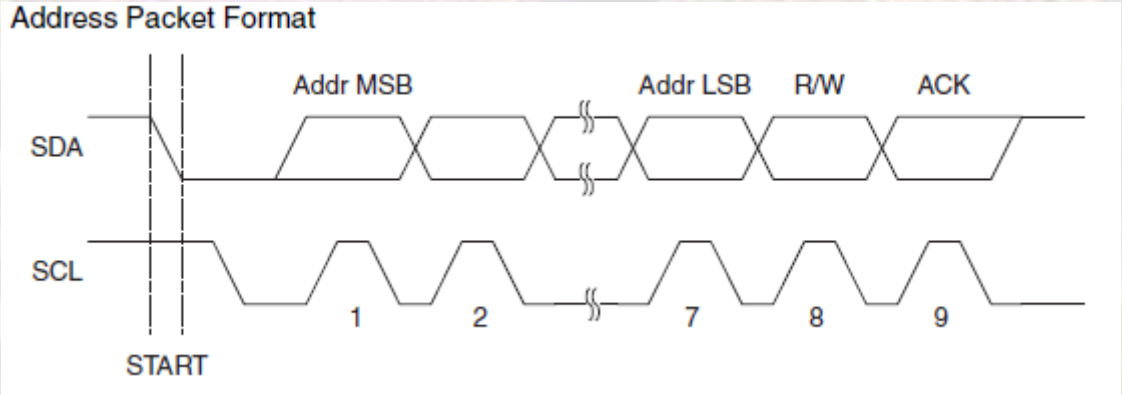
Serial Communications

- TWI Timing
  - Special timing requirements for
    - start transmission
    - stop transmission
    - repeated start transition
      - master does not relinquish the bus in this mode



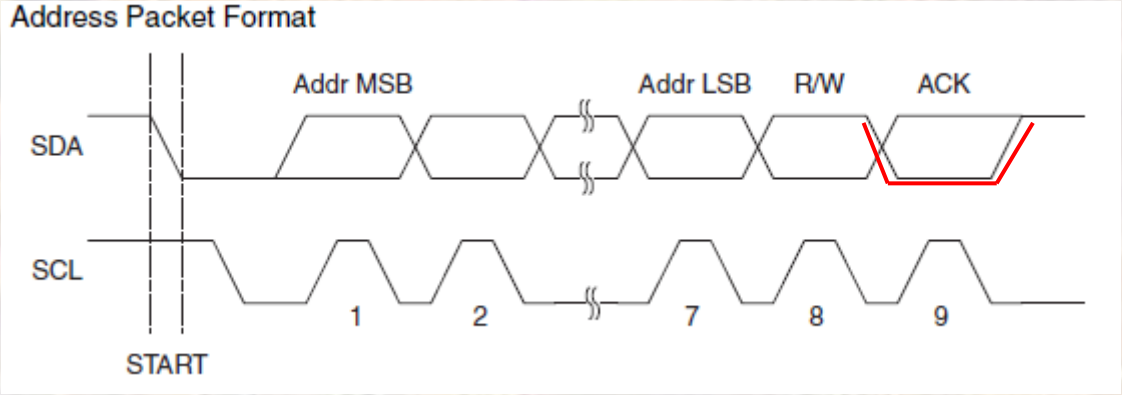
Serial Communications

- TWI Timing
  - Addressing
    - Indicate which slave to transmit to or receive from by first transmitting the “address” of the desired device
    - Often this value is hardwired via external pins on the slave device
    - 7 bits for each address



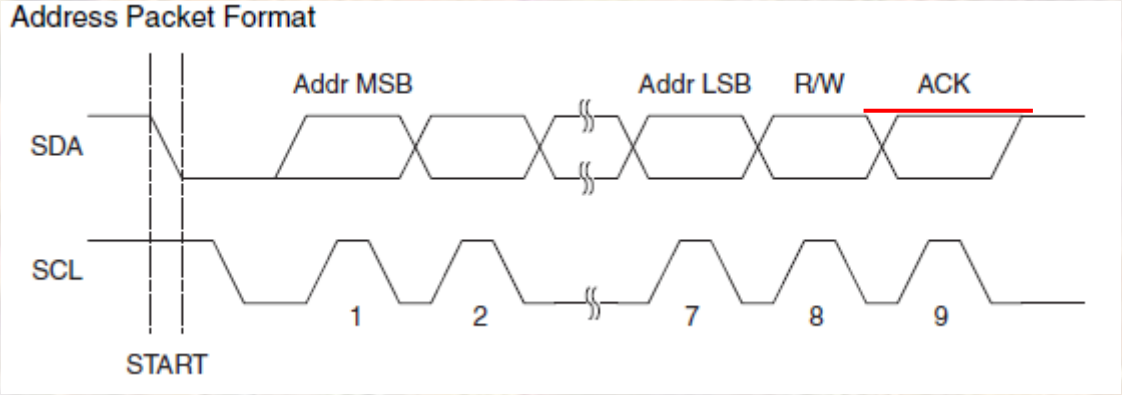
Serial Communications

- TWI Timing
  - R/W bit indicates a read or write operation is to follow
    - Read is active high
  - ACK
    - The master drives the data bus from start through the R/W bit and then releases the bus
    - The slave then pulls down the bus in the last clock cycle to indicate a completed transmission



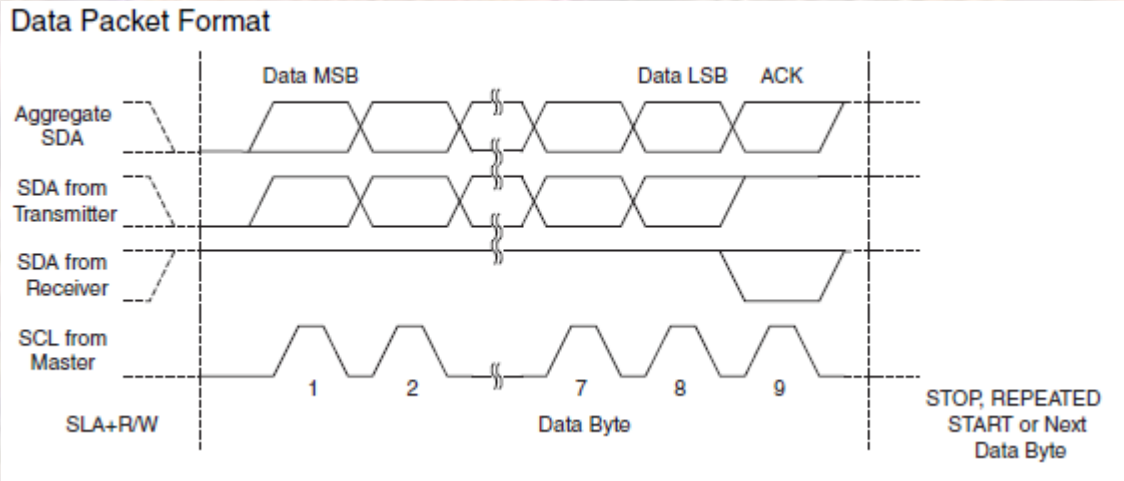
Serial Communications

- TWI Timing
  - ACK – cont'd
    - If the master fails to see the slave pull down the bus in the 9th clock cycle (NACK)
      - Transmission failed
      - Some sort of error action is required



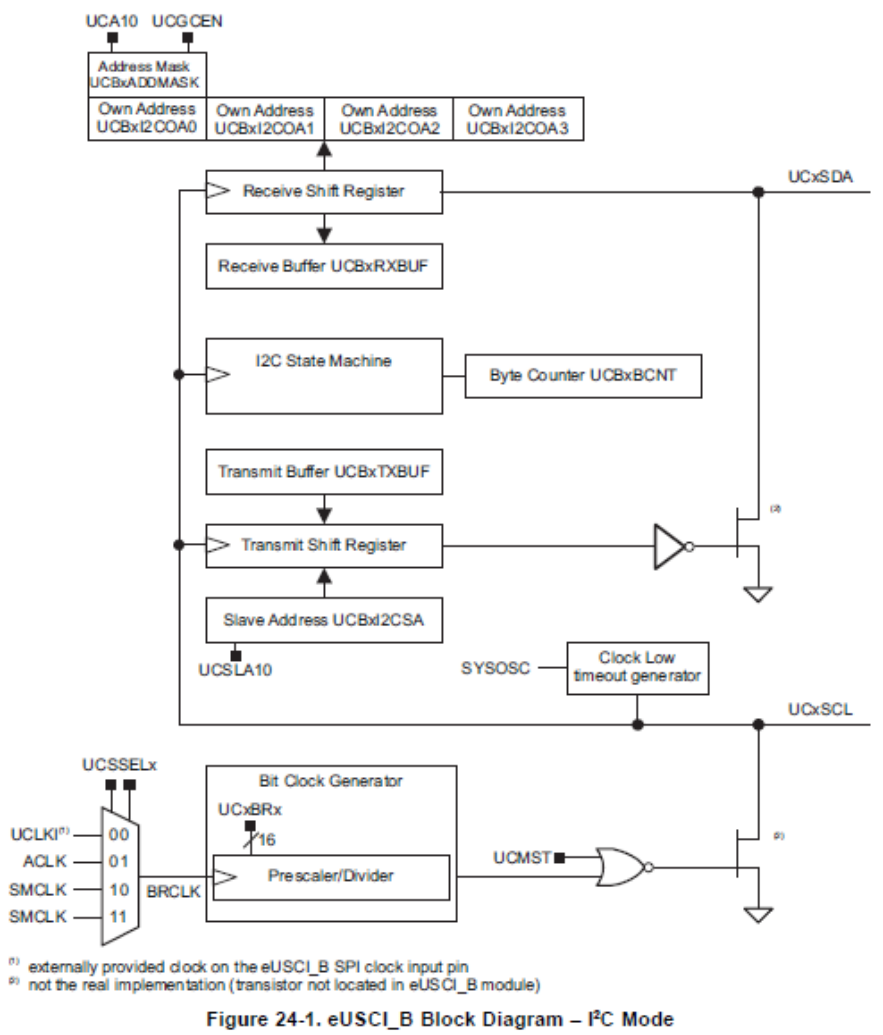
Serial Communications

- TWI Timing
  - Data packet
    - After getting an ACK on the address – data can be sent
    - 8 bits of data
    - 1 bit for a data ACK
    - This can be repeated many times



Serial Communications

- TWI Implementation
  - Separate RX/TX registers
  - Separate RX/TX buffers
  - State Machine
  - Clock divider
  - Control register
  - Status register



externally provided clock on the eUSCI_B SPI clock input pin
not the real implementation (transistor not located in eUSCI_B module)

Figure 24-1. eUSCI_B Block Diagram – I²C Mode

Serial Communications

- TWI Addressing Modes

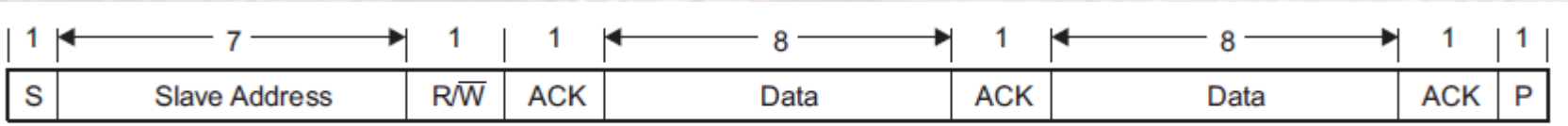


Figure 24-5. I²C Module 7-Bit Addressing Format

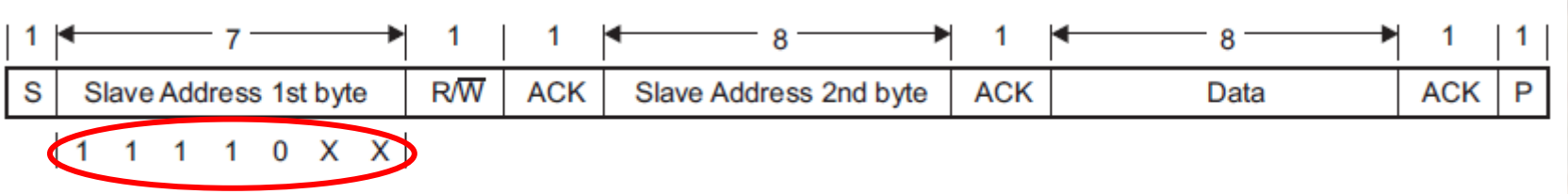


Figure 24-6. I²C Module 10-Bit Addressing Format

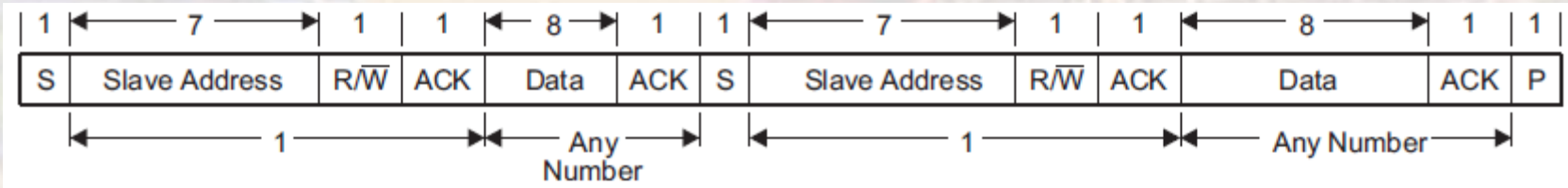


Figure 24-7. I²C Module Addressing Format With Repeated START Condition

Serial Communications

- TWI Operating Modes

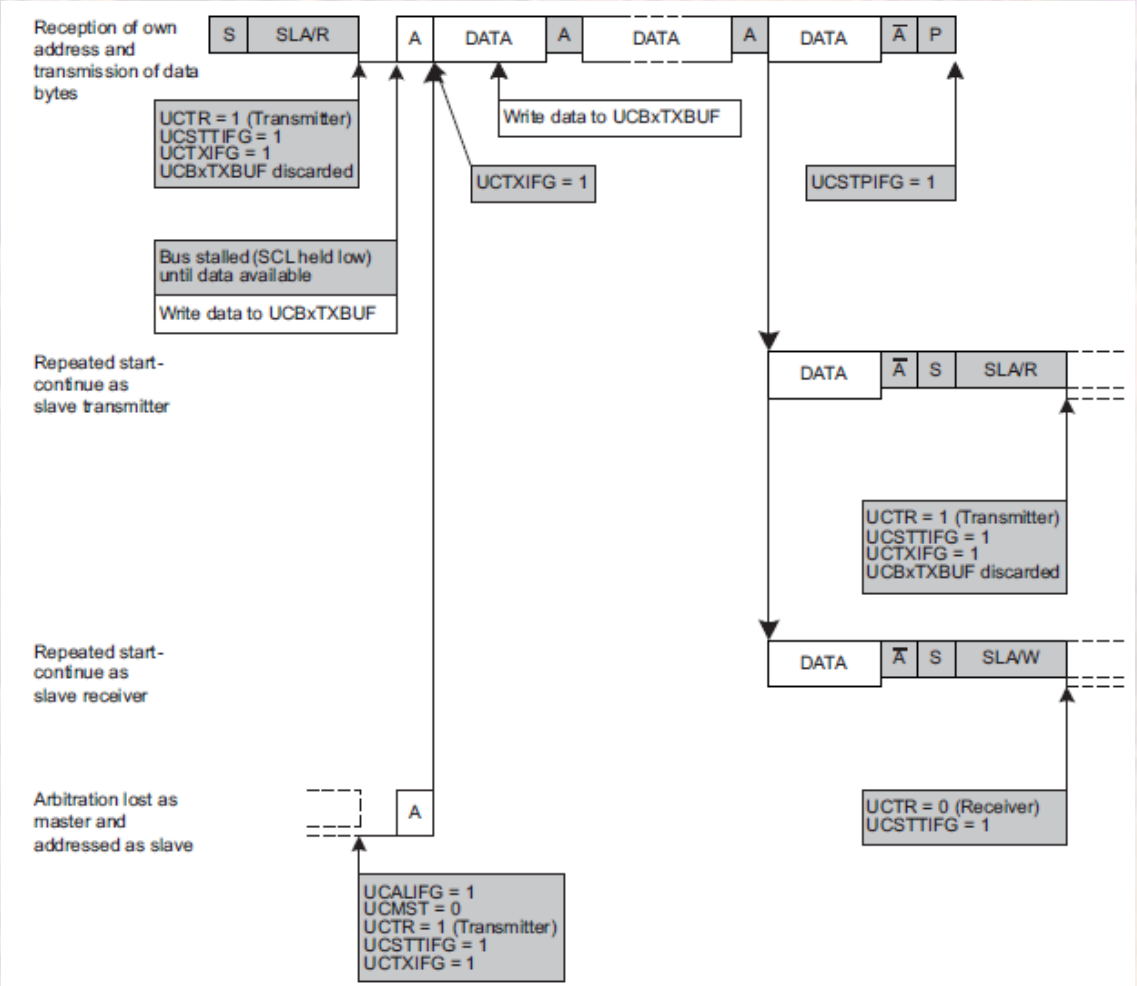


Figure 24-9. I²C Slave Transmitter Mode

Serial Communications

- TWI Operating Modes

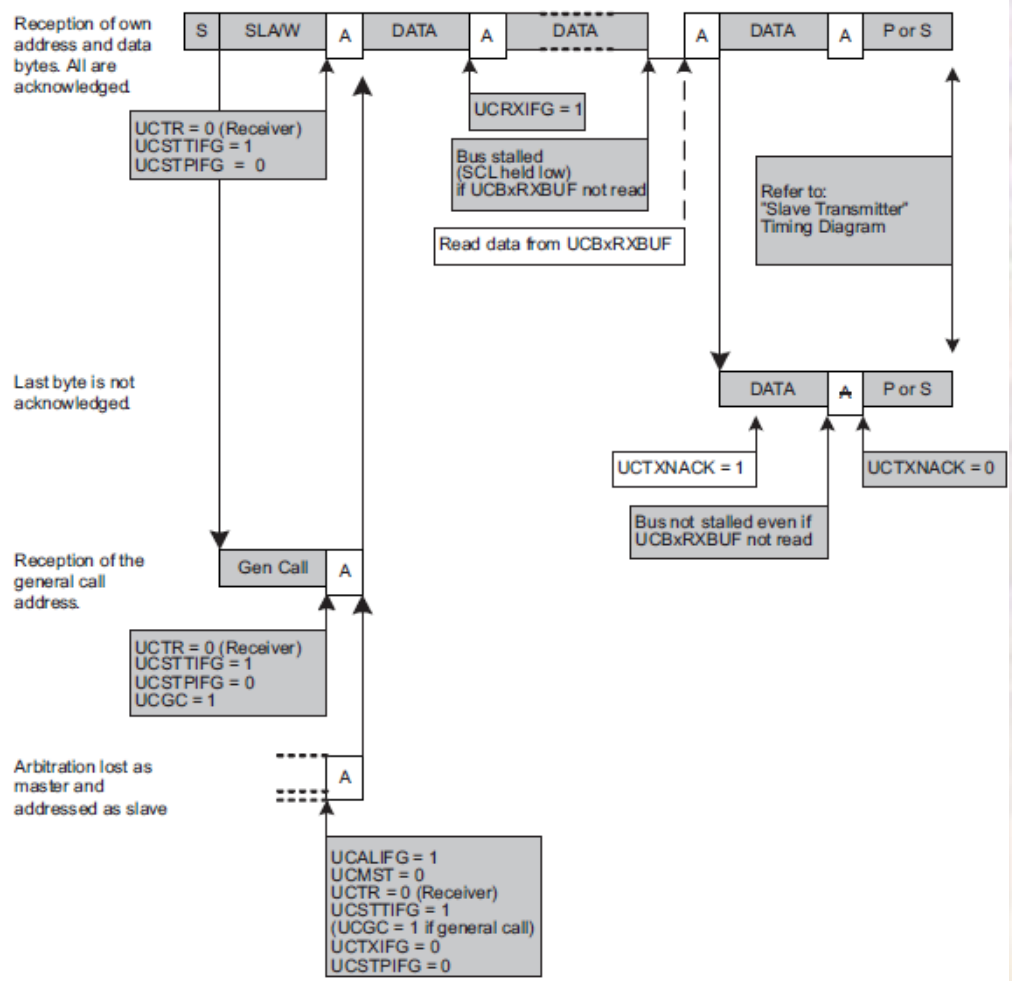


Figure 24-10. I²C Slave Receiver Mode

Serial Communications

- TWI Operating Mode

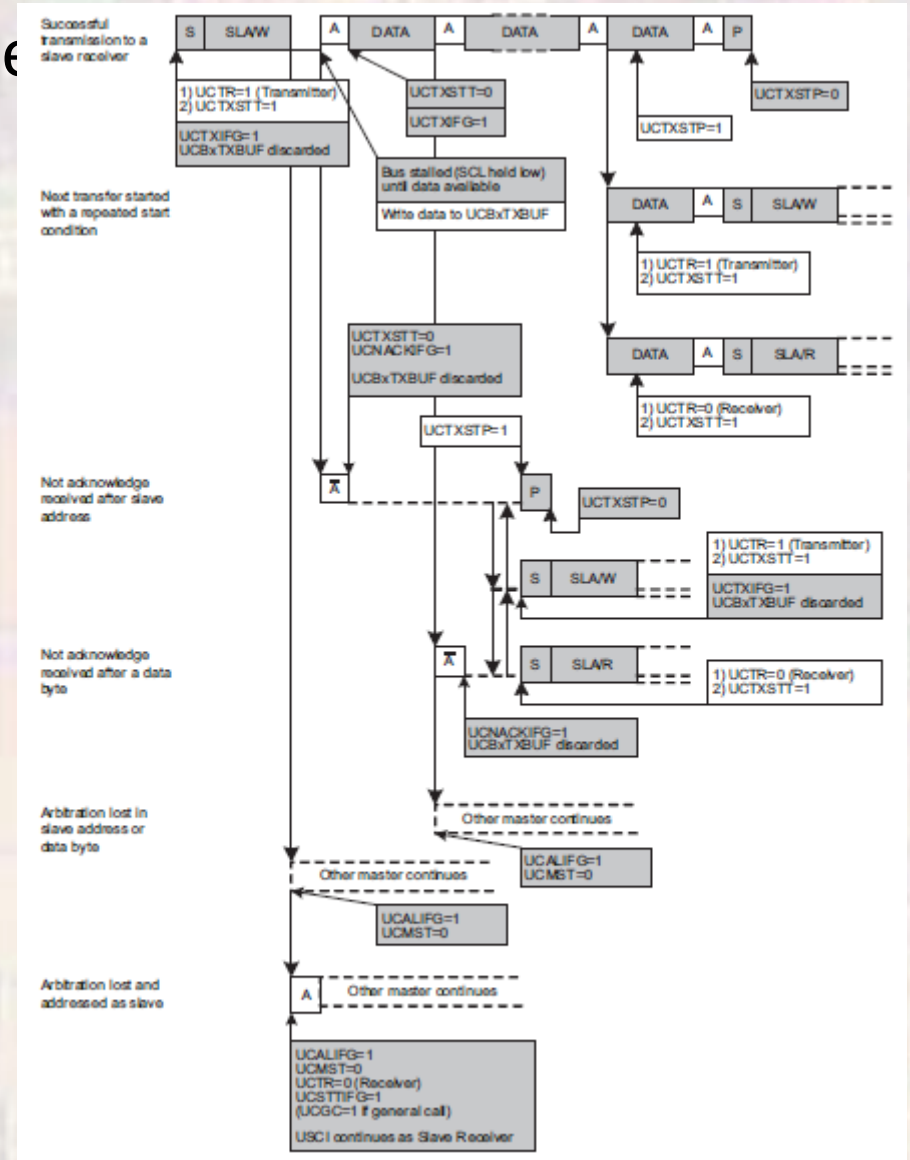


Figure 24-12. I²C Master Transmitter Mode

Serial Communications

- TWI Operating Mod

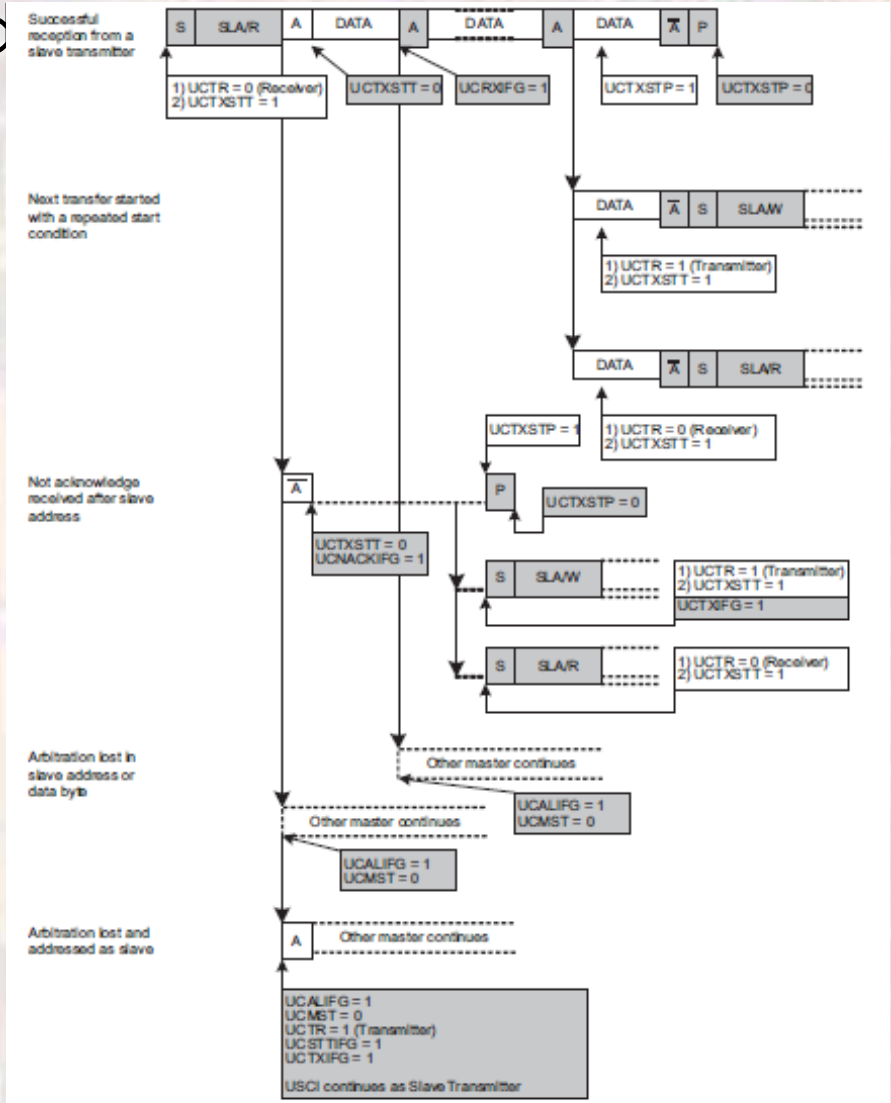
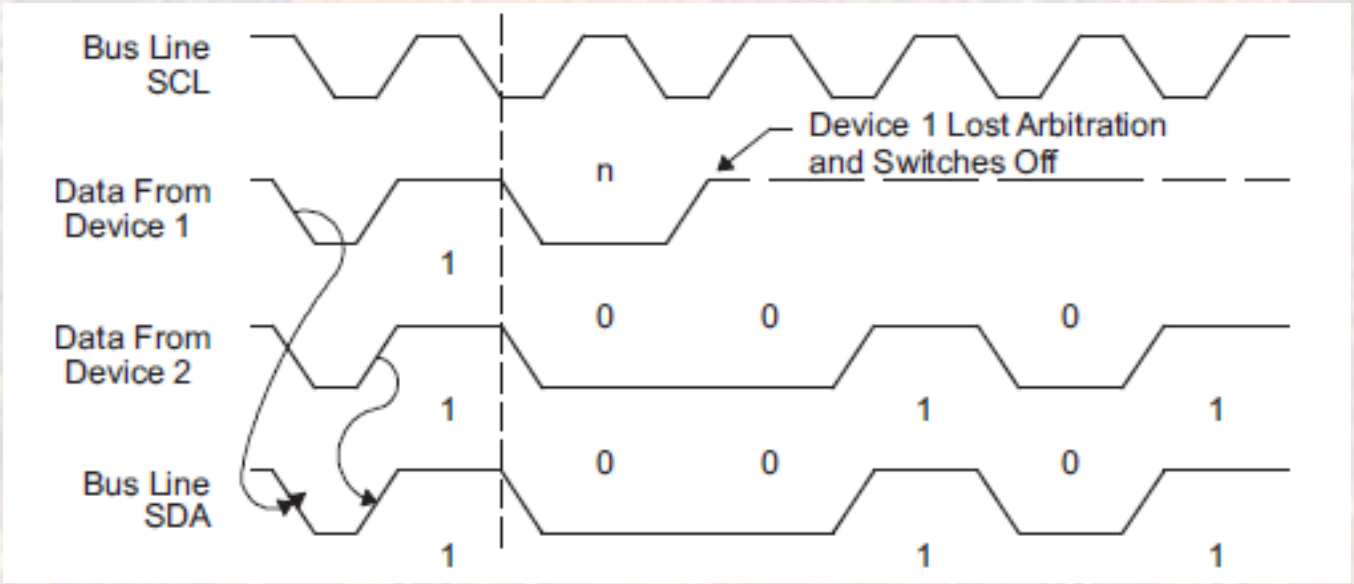


Figure 24-13. I²C Master Receiver Mode

Serial Communications

- TWI Multi-Master Arbitration
  - First master that attempts to transmit a 1 when the other transmits a 0 – loses arbitration and shuts off



Serial Communications

Universal Synchronous/Asynchronous Receiver/Transmitter

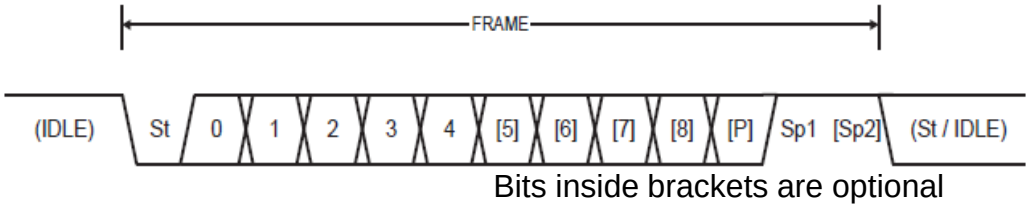
USART

Serial Communications

- UART/USART
  - Serial receiver / transmitter
  - Asynchronous and Synchronous versions
  - Asynchronous
    - 2 pin interface
    - RxD – receive data pin
    - TxD – transmit data pin
    - Clock recovery system
  - Synchronous
    - 3 pin interface
    - RxD – receive data pin
    - TxD – transmit data pin
    - Xck – Clock – PD4

Serial Communications

- UART Frame
  - 1 start bit
  - 5, 6, 7, 8, or 9 data bits – typically LSB first
  - none, even, or odd - parity bit
  - 1 or 2 stop bits,
  - Overflow, Framing, Parity – error detection

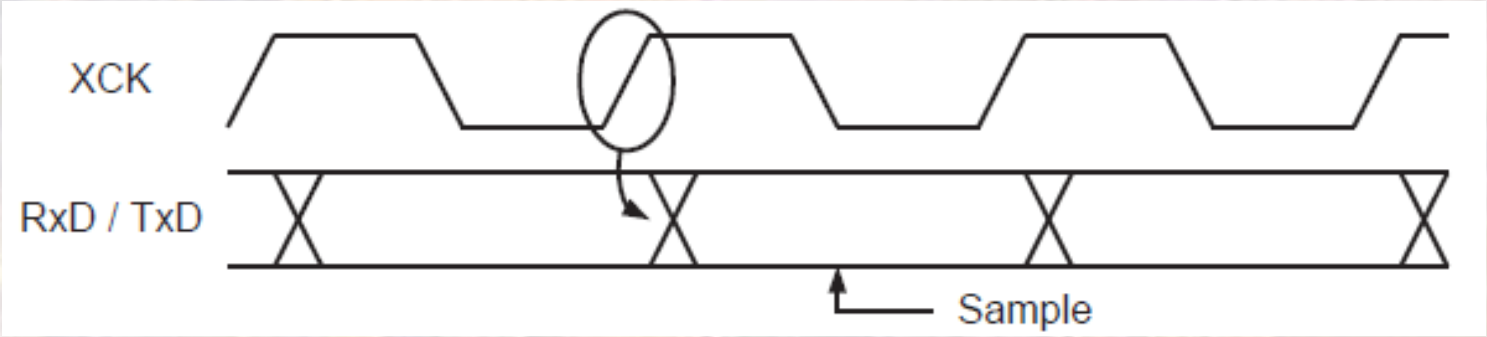


Bits inside brackets are optional

St	Start bit, always low.
(n)	Data bits (0 to 8).
P	Parity bit. Can be odd or even.
Sp	Stop bit, always high.
IDLE	No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.

Serial Communications

- UART Synchronous mode operation
  - Master creates clock signal
    - Xck max is typically $Clk_{system}/4$ for timing purposes
  - Master and Slave
    - Transmit on one clock edge
    - Receive (latch data) on the opposite clock edge

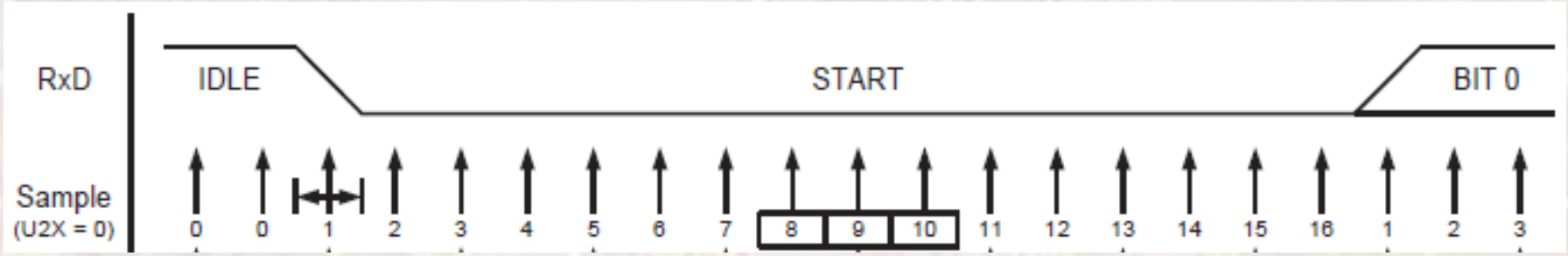


Serial Communications

- UART Asynchronous mode operation
  - Transmit is unchanged
  - Xck is disabled
    - No Master or Slave
  - Must establish an agreed BAUD rate
    - Max BAUD rate is typically $Clk_{system}/16$
    - Limited by HW – clock selection options
    - Fixed in SW
    - Start at known BAUD rate and agree to go faster/slower
  - Receiver circuitry includes:
    - Clock recovery block
    - Data recovery block

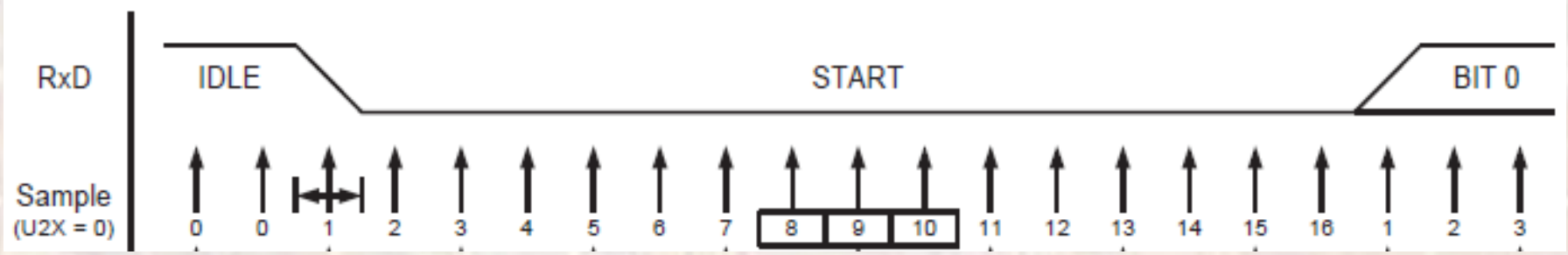
Serial Communications

- UART Asynchronous mode operation
  - Clock and Data recovery
    - Internal clock at 16x BAUD rate
    - Sample RxD signal with internal clock
    - Detect Start bit falling edge
    - Sample RxD after 8,9,10 internal clocks
    - Majority determines bit value
    - Assuming a valid start – all further bits sampled at multiples of 16 clocks



Serial Communications

- UART Multi-processor mode
  - Multiple processors sharing the same USART signals
  - Use the last bit in the data (e.g. bit 9 when using 8 bit data) to indicate an address or data value is in the frame
  - If it is an address and it is your address, collect subsequent data frames
  - If it is not your address, ignore subsequent data frames



Serial Communications

- UART Error detection
  - Parity
    - Create an error if parity is wrong
  - Overflow
    - Create an error if new data is ready to be sampled and the last data has not been read from the buffer yet
    - Double buffering is common
  - Frame error
    - Stop bit not detected when expected
    - Idle not detected when expected

Serial Communications

- UART Common BAUD rates

BAUD Rate	Bit Width (us)	Frame Length start,8bit data, parity, 1 stop (us)	Data Rate (Bits/s)	(Bytes/s)
4800	208.33	2291.67	3,491	436
9600	104.17	1145.83	6,982	873
19200	52.08	572.92	13,964	1,745
38400	26.04	286.46	27,927	3,491
57600	17.36	190.97	41,891	5,236
115200	8.68	95.49	83,782	10,473

Serial Communications

- MSP432 UART Implementation
  - Clock Selector
  - Prescaler /baud rate generator
  - RX/TX buffers
  - RX//TX state machines
  - Error detector
  - Automatic baud rate detection

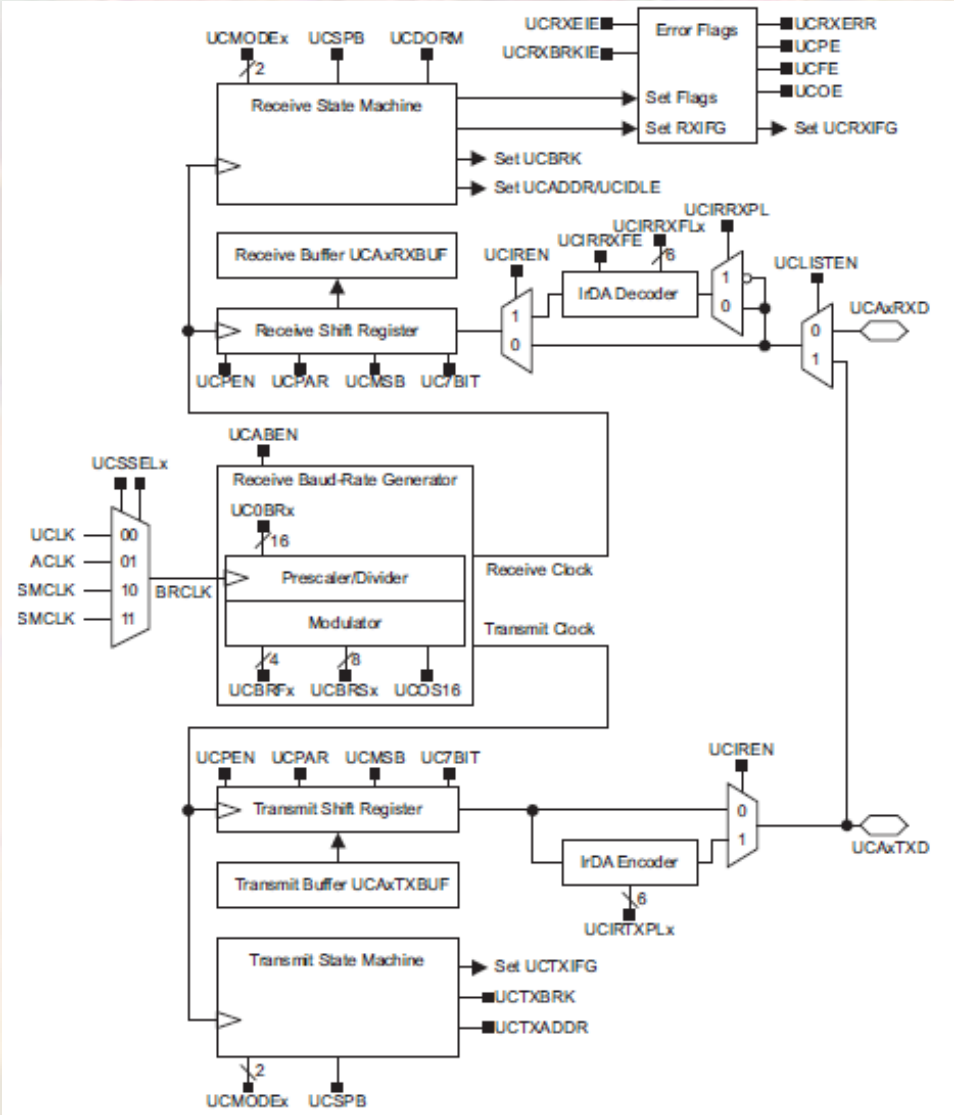


Figure 22-1. eUSCI_Ax Block Diagram – UART Mode (UCSYNC = 0)

Serial Communications

- MSP432 UART Implementation
  - Required bits
    - start
    - Data 0-7
    - Stop
  - Optional bits
    - data bit 8
    - address bit (7 bit mode)
    - parity bit
    - 2nd stop bit

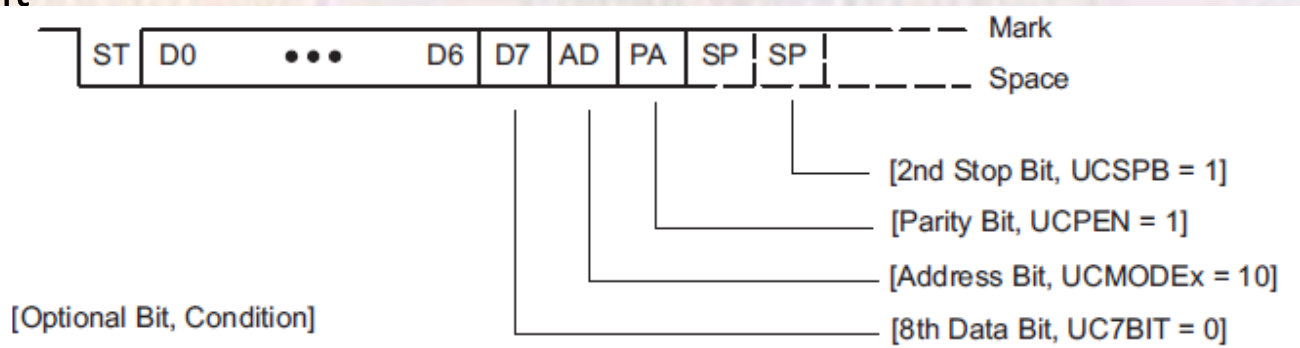


Figure 22-2. Character Format

Serial Communications

- MSP432 UART Implementation
  - Idle Line Microprocessor Format
    - Used when more than 2 devices share the signal paths
    - Blocks of data separated by at least 10 sequential 1's after a stop
    - The first block of data following an idle time is an address

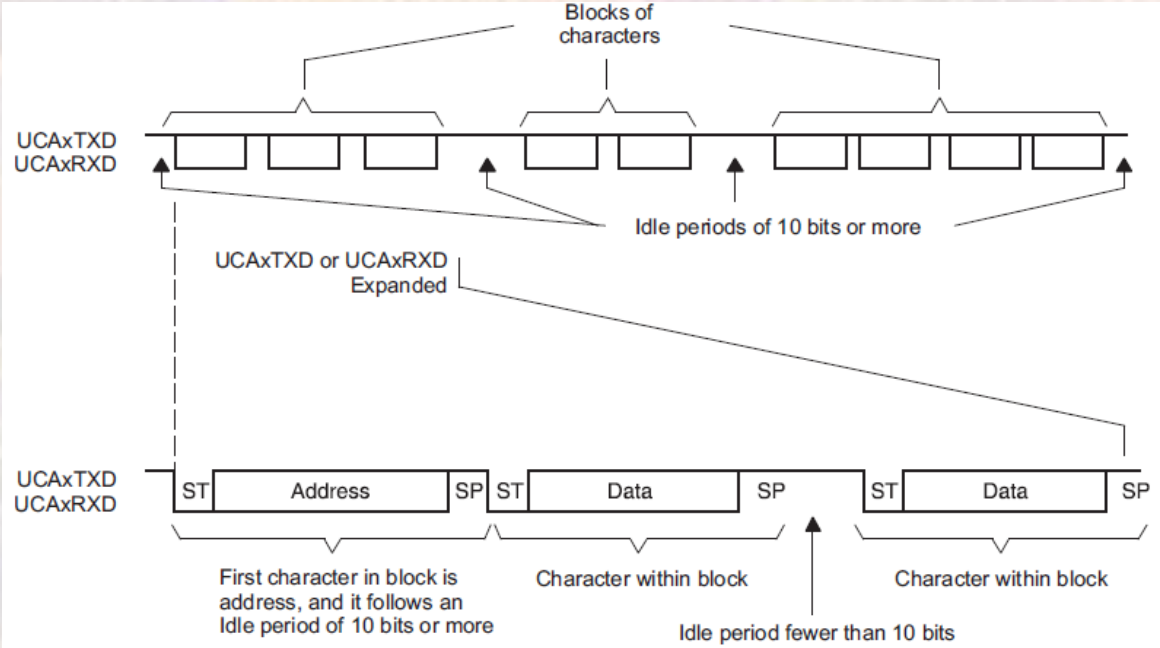


Figure 22-3. Idle-Line Format

Serial Communications

- MSP432 UART Implementation
  - Address Bit Microprocessor Format
    - Used when more than 2 devices share the signal paths
    - The first block of data after a start contains an extra bit to indicate it is an address

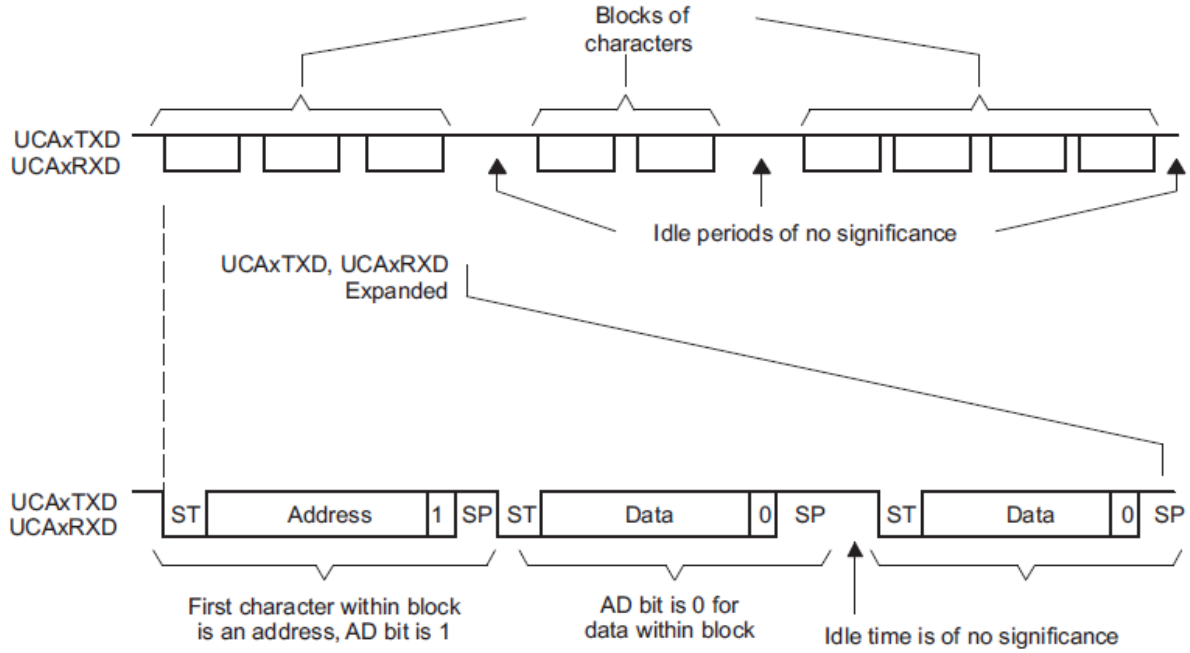


Figure 22-4. Address-Bit Multiprocessor Format

Serial Communications

- MSP432 UART Implementation
  - Automatic baud rate detection
    - Break -> delimiter -> sync pulses
    - Baud rate is calculated based on the sync pulses
      - Max = 1MBaud
      - Min – normal mode = 244 Baud
      - Min – low frequency mode = 15Baud

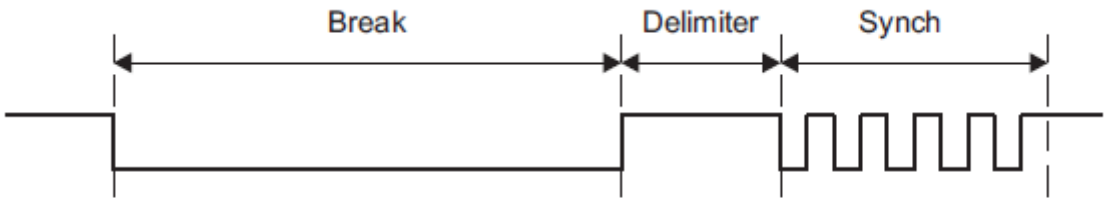


Figure 22-5. Auto Baud-Rate Detection – Break/Synch Sequence

Serial Communications

- MSP432 UART Implementation
  - Receive Error Detection

Table 22-1. Receive Error Conditions

Error Condition	Error Flag	Description
Framing error	UCFE	A framing error occurs when a low stop bit is detected. When two stop bits are used, both stop bits are checked for framing error. When a framing error is detected, the UCFE bit is set.
Parity error	UCPE	A parity error is a mismatch between the number of 1s in a character and the value of the parity bit. When an address bit is included in the character, it is included in the parity calculation. When a parity error is detected, the UCPE bit is set.
Receive overrun	UCOE	An overrun error occurs when a character is loaded into UCAxRXBUF before the prior character has been read. When an overrun occurs, the UCOE bit is set.
Break condition	UCBRK	When not using automatic baud-rate detection, a break is detected when all data, parity, and stop bits are low. When a break condition is detected, the UCBRK bit is set. A break condition can also set the interrupt flag UCRXIFG if the break interrupt enable UCBRKIE bit is set.