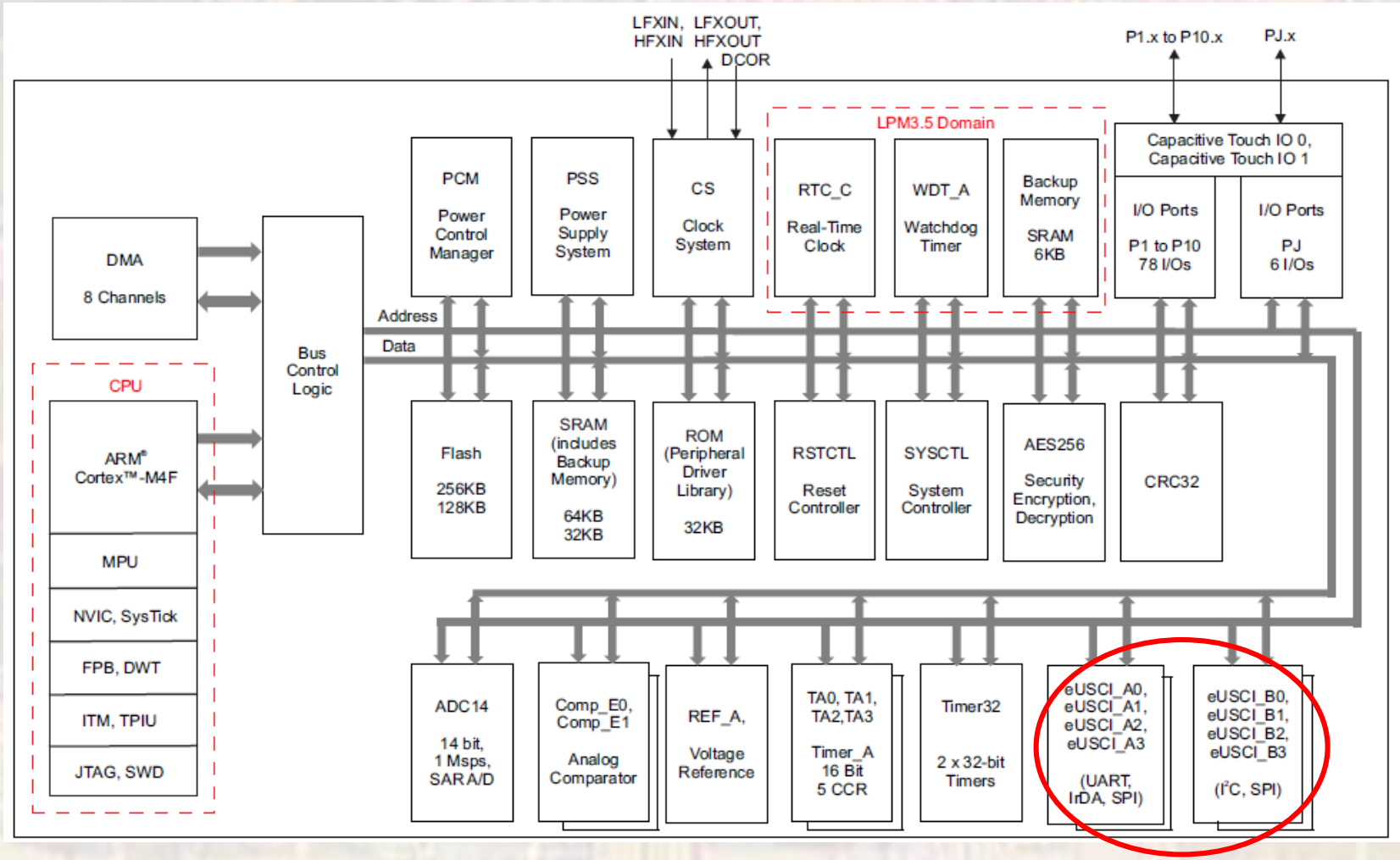
Serial Peripheral Interface (SPI)

Last updated 10/21/19

SPI

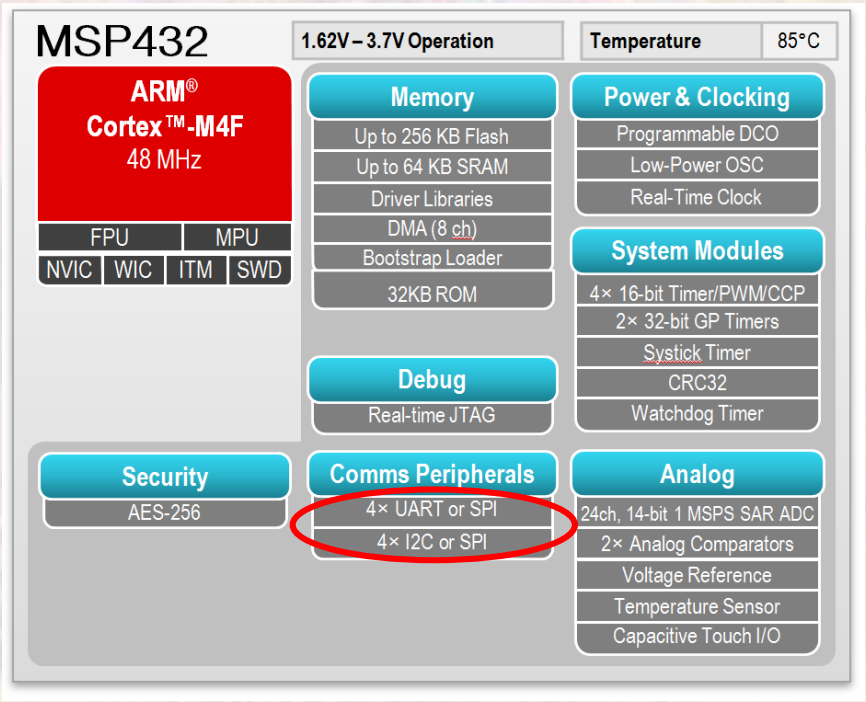
- MSP432 SPI



eUSCI = enhanced Universal Serial Communications Interface

SPI

- MSP432 SPI
  - ARM (AMBA Compliant)
  - 7/8 bit transmission
  - Master/Slave
  - LSB/MSB first
  - Separate RX/TX registers
  - 4 A modules and 4 B modules

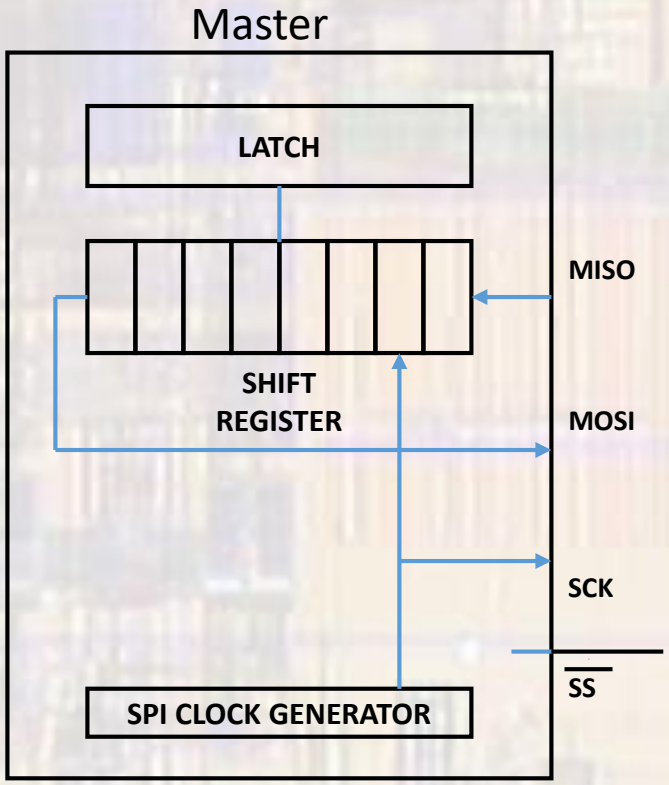


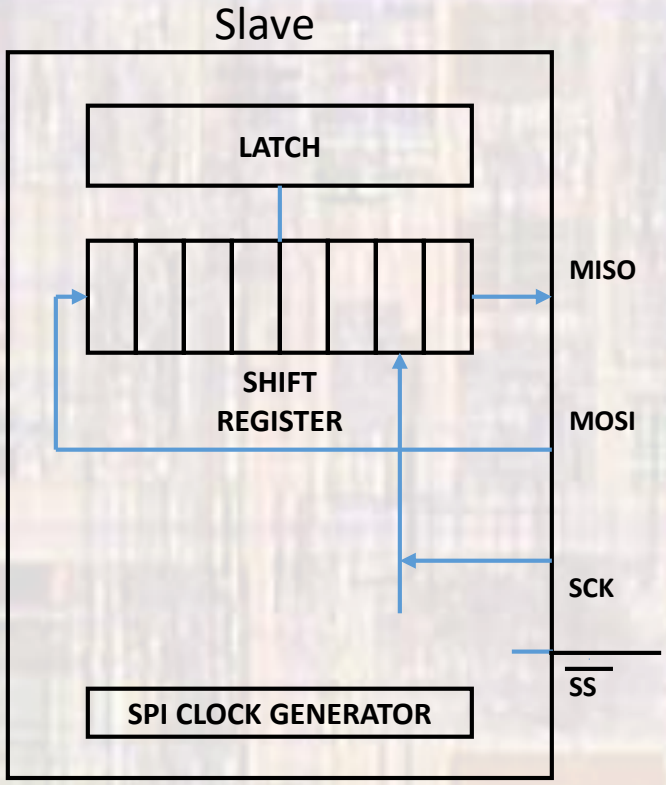
SPI

- 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

SPI

- Overview





MISO – Master:IN or Slave:OUT

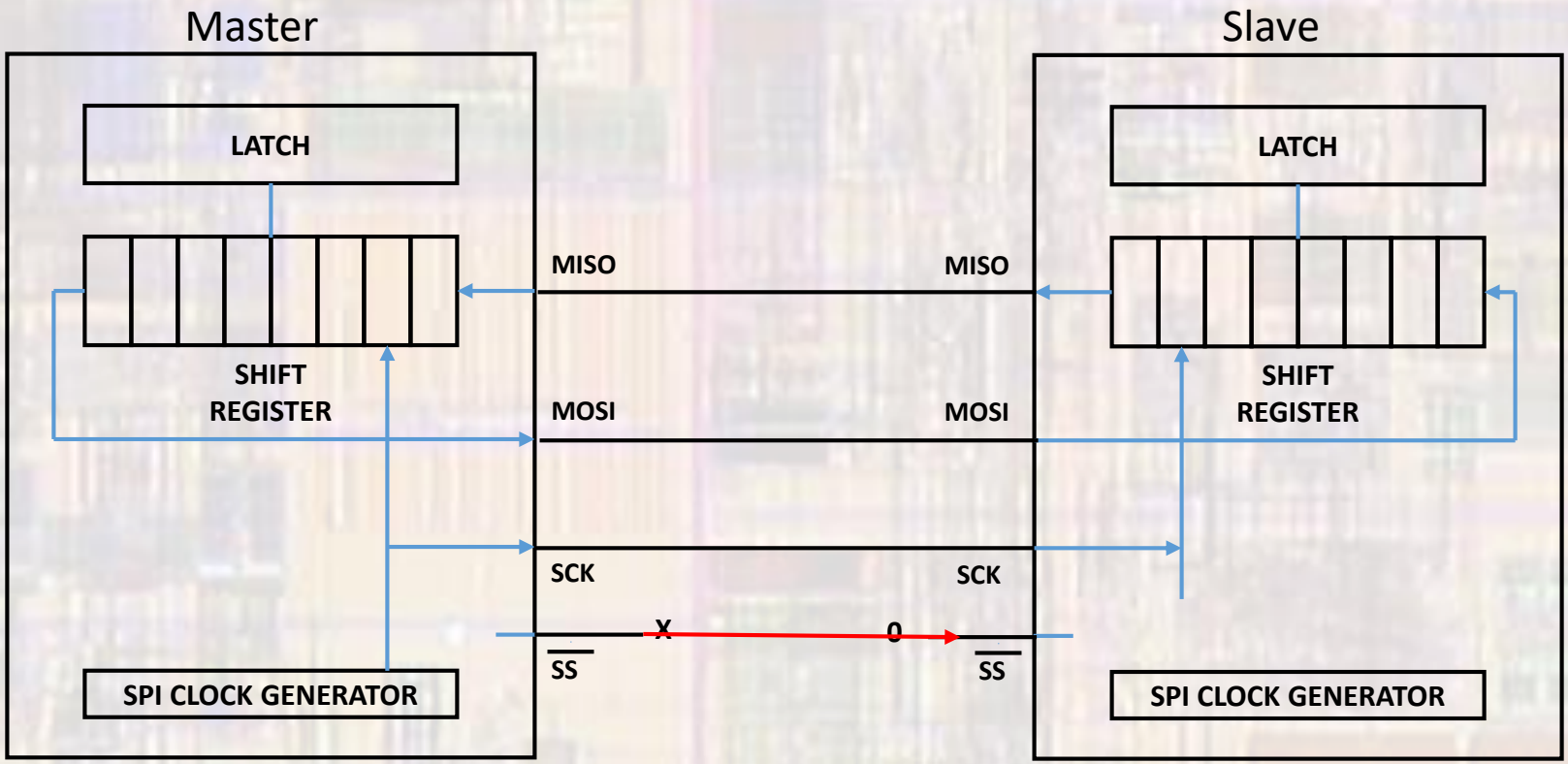
MOSI – Master:OUT or Slave:IN

SCK – SPI CLK

$\overline{SS}$ – Slave Select Bar

SPI

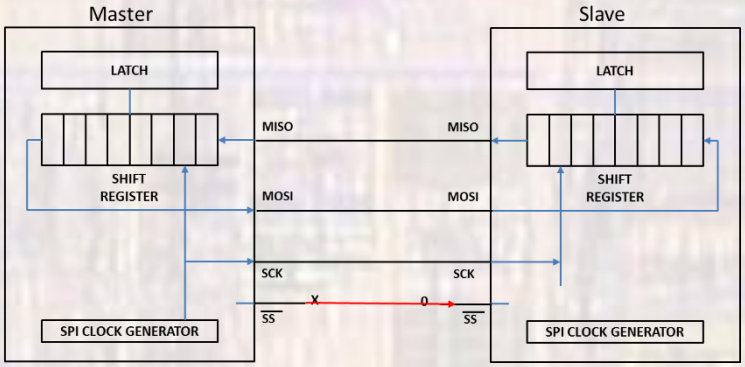
- Operation



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

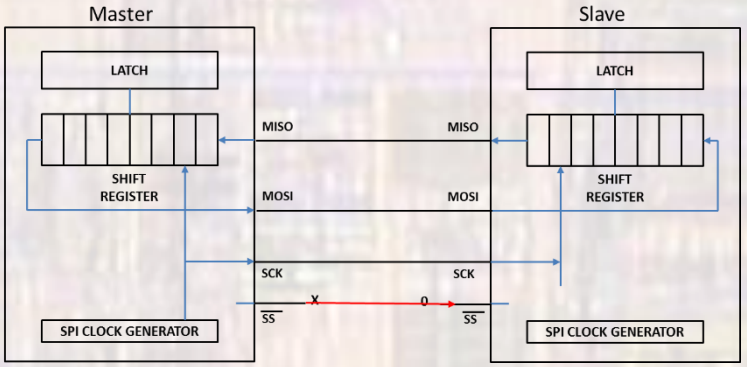
SPI

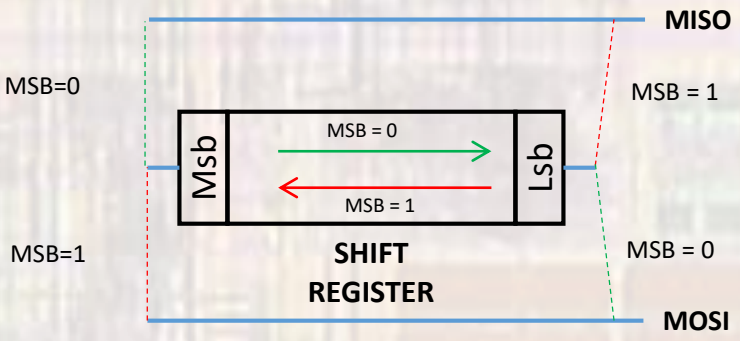
- Operation
  - Configure 1 device as master
  - Configure 1 or more devices as slaves
  - Load values into register(s)
  - Pull SSbar low on the desired slave device
  - Initiate transfer by writing to the data register
    - The master will generate the appropriate clocks
  - If interrupts are enabled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



SPI

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

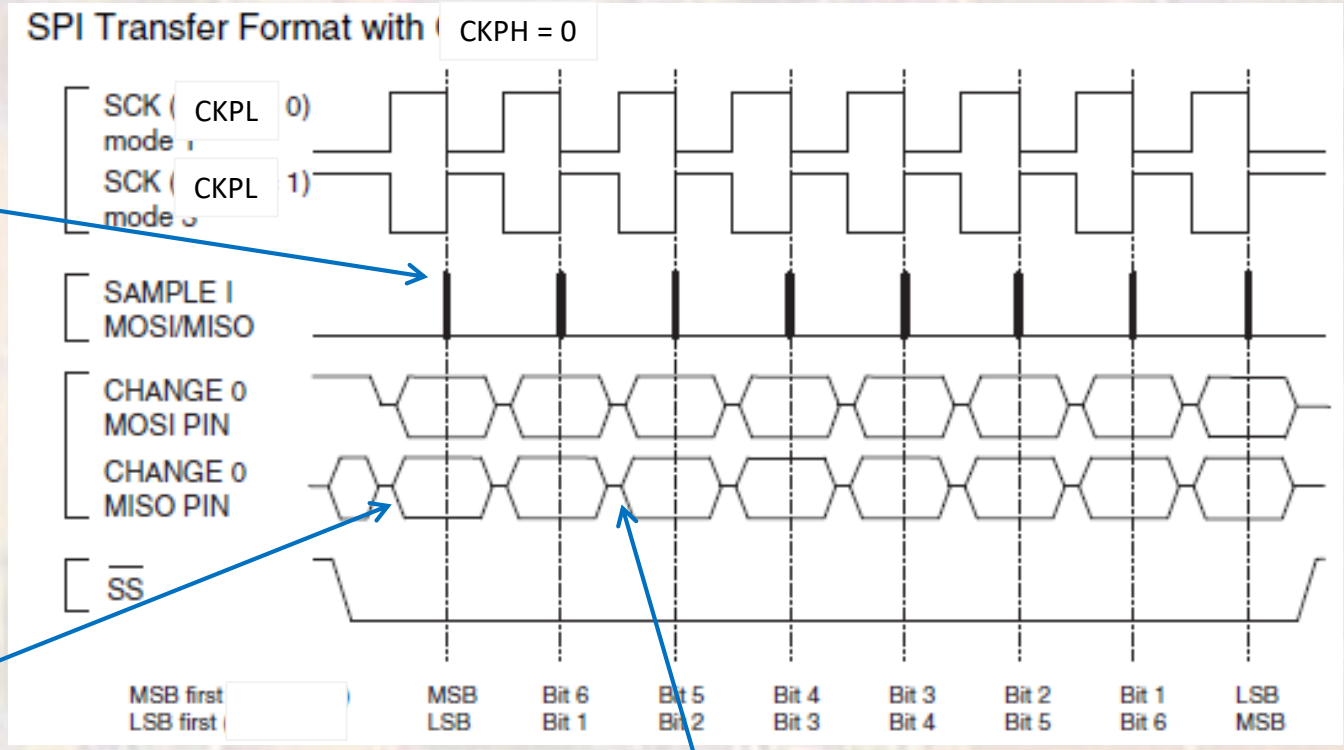




SPI

- Operation
  - CKPH = 0

Captured in register on trailing clock edge

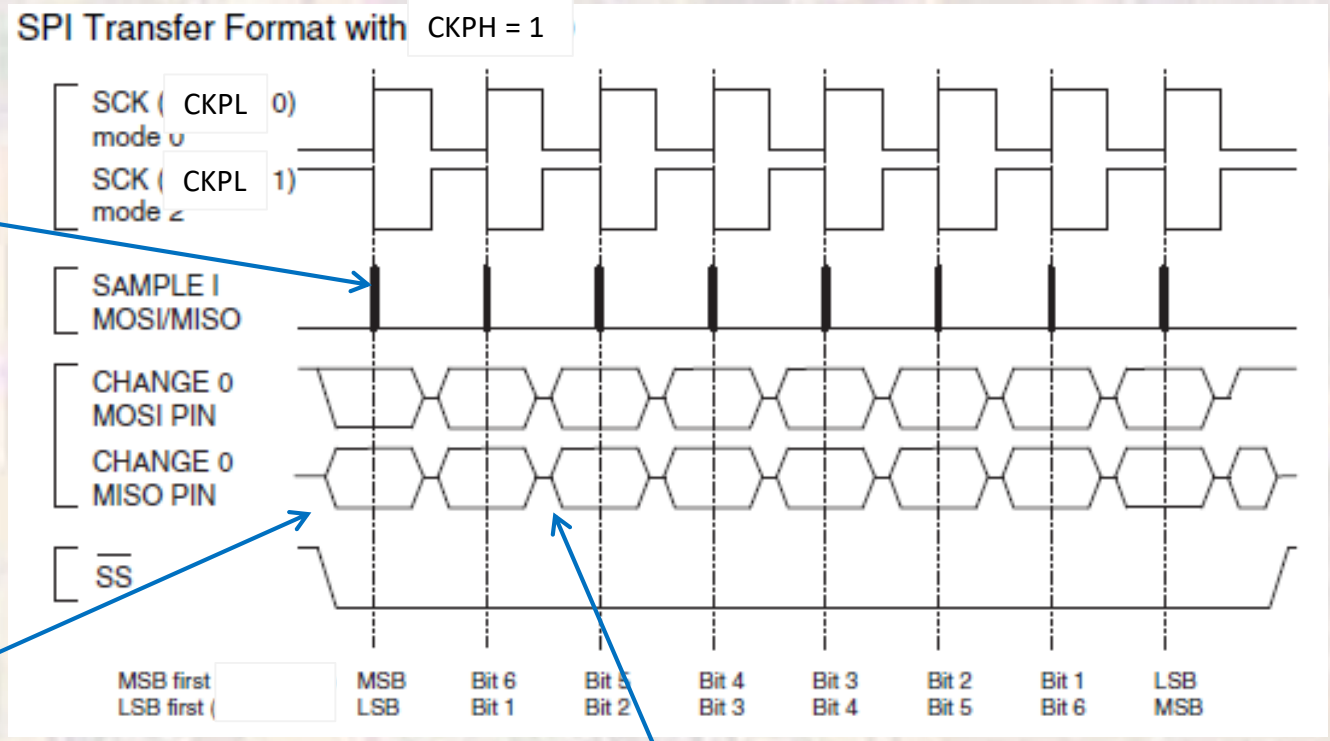


Values active on pins on first clock edge

New values placed on pins on leading clock edge

SPI

- Operation
  - CKPH = 1



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

SPI

- Operation
  - Multiple Slave Configuration

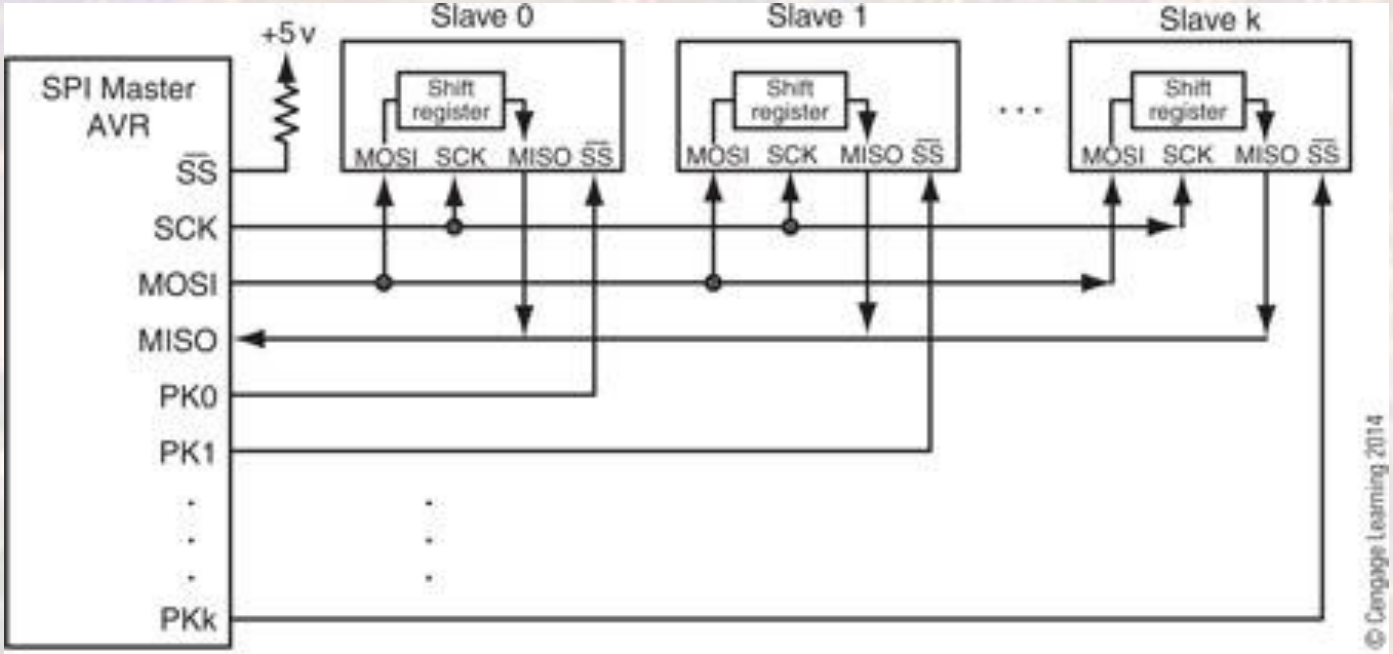


Figure 14.7 ■ Single-master and multiple-slave device connection (method 1)

SPI

- Operation
  - Multiple Slave – Extended Shift Configuration

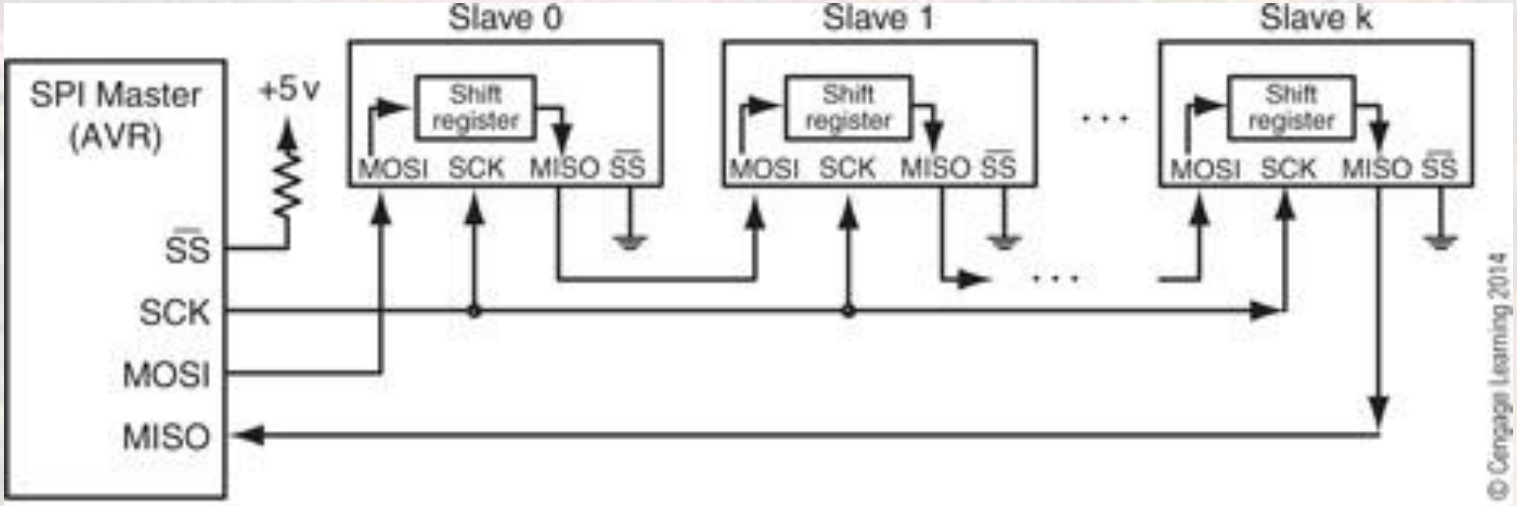


Figure 14.8 ■ Single-master and multiple-slave device connection (method 2)

SPI

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

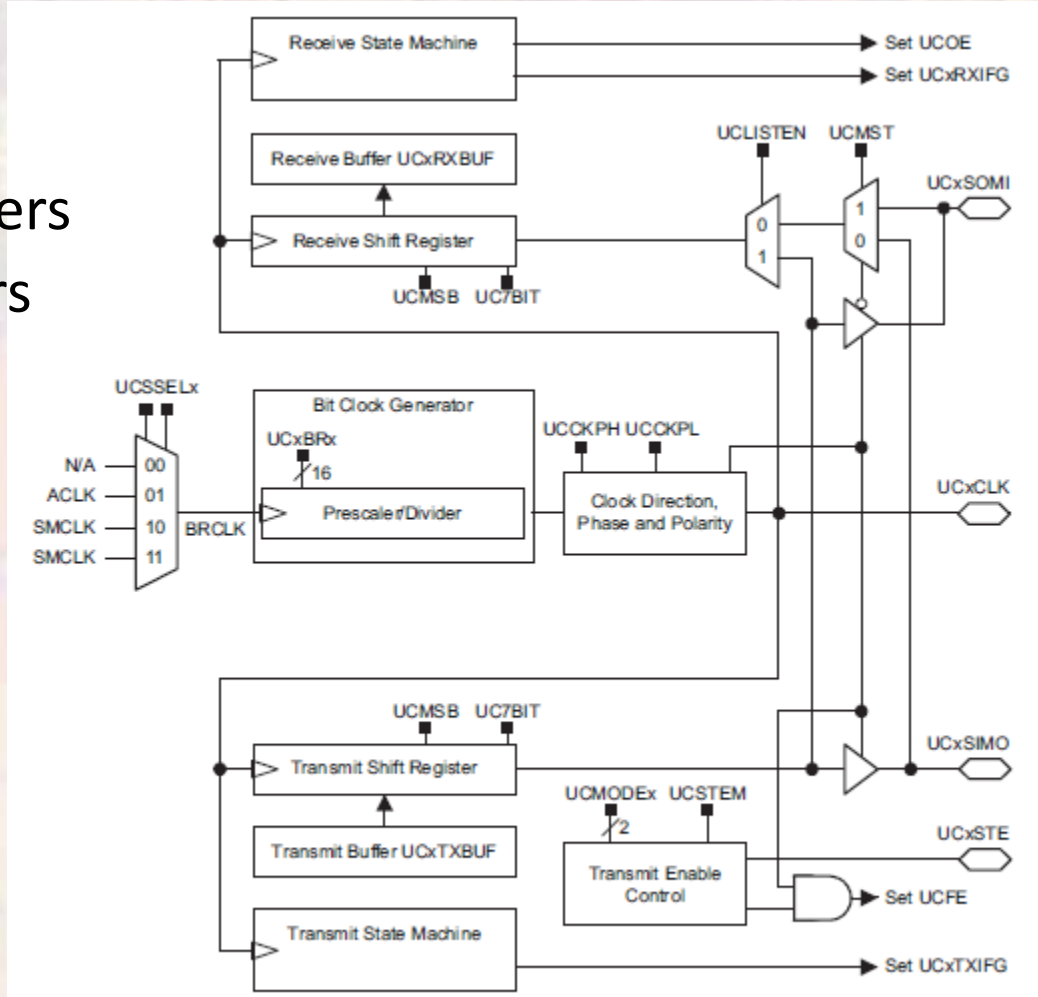


Figure 23-1. eUSCI Block Diagram – SPI Mode

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

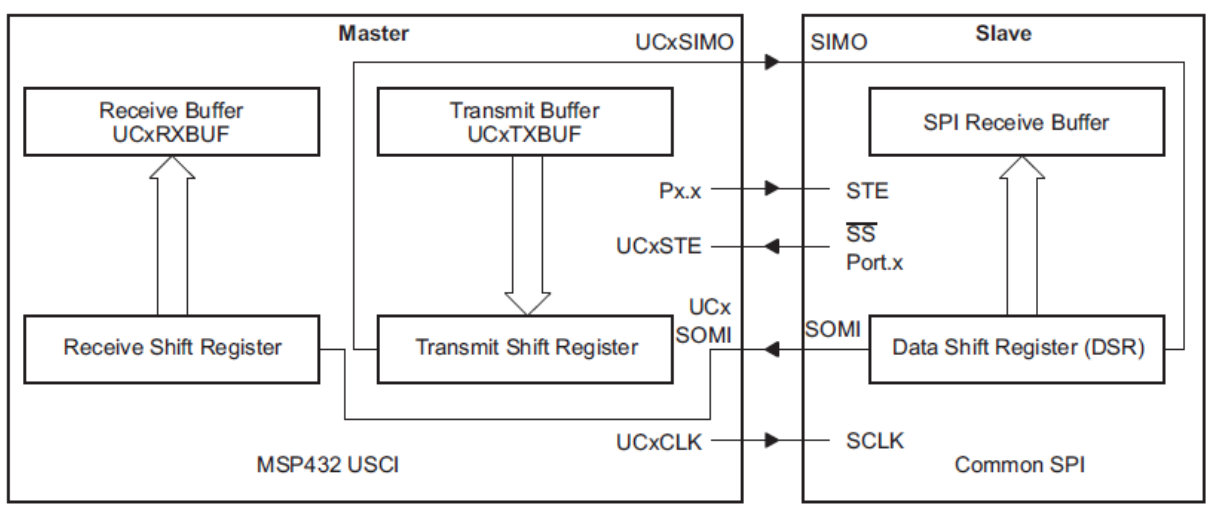


Figure 23-2. eUSCI Master and External Slave (UCSTEM = 0)

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 (no interrupt)

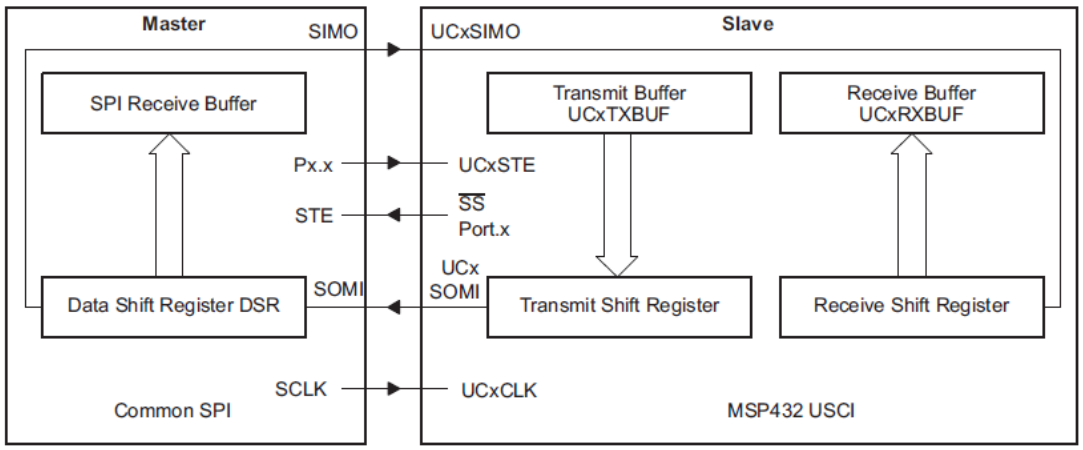


Figure 23-3. eUSCI Slave and External Master

SPI

- MSP432 SPI Registers

Table 23-2. eUSCI_A SPI Registers

Offset	Acronym	Register Name	Section
00h	UCAxCTLW0	eUSCI_Ax Control Word 0	Section 23.4.1
00h	UCAxCTL1	eUSCI_Ax Control 1	
01h	UCAxCTL0	eUSCI_Ax Control 0	
06h	UCAxBRW	eUSCI_Ax Bit Rate Control Word	Section 23.4.2
06h	UCAxBR0	eUSCI_Ax Bit Rate Control 0	
07h	UCAxBR1	eUSCI_Ax Bit Rate Control 1	
0Ah	UCAxSTATW	eUSCI_Ax Status	Section 23.4.3
0Ch	UCAxRXBUF	eUSCI_Ax Receive Buffer	Section 23.4.4
0Eh	UCAxTXBUF	eUSCI_Ax Transmit Buffer	Section 23.4.5
1Ah	UCAxIE	eUSCI_Ax Interrupt Enable	Section 23.4.6
1Ch	UCAxIFG	eUSCI_Ax Interrupt Flag	Section 23.4.7
1Eh	UCAxIV	eUSCI_Ax Interrupt Vector	Section 23.4.8

SPI

- MSP432 SPI Control Register 0

EUCSI_nx_SPI->CTLW0
n = A or B
x = 0, 1, 2, 3

Figure 23-5. UCAxCTLW0 Register

15	14	13	12	11	10	9	8
UCCKPH	UCCKPL	UCMSB	UC7BIT	UCMST	UCMODEx		UCSYNC
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
UCSSELx		Reserved				UCSTEM	UCSWRST
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-1
	Modify only when UCSWRST = 1.						

Table 23-3. UCAxCTLW0 Register Description

Bit	Field	Type	Reset	Description
15	UCCKPH	RW	0h	Clock phase select 0b = Data is changed on the first UCLK edge and captured on the following edge. 1b = Data is captured on the first UCLK edge and changed on the following edge.
14	UCCKPL	RW	0h	Clock polarity select 0b = The inactive state is low. 1b = The inactive state is high.
13	UCMSB	RW	0h	MSB first select. Controls the direction of the receive and transmit shift register. 0b = LSB first 1b = MSB first

SPI

- MSP432 SPI Control Register 0

Bit	Field	Type	Reset	Description
12	UC7BIT	RW	0h	Character length. Selects 7-bit or 8-bit character length. 0b = 8-bit data 1b = 7-bit data
11	UCMST	RW	0h	Master mode select 0b = Slave mode 1b = Master mode
10-9	UCMODEx	RW	0h	eUSCI mode. The UCMODEx bits select the synchronous mode when UCSYNC = 1. 00b = 3-pin SPI 01b = 4-pin SPI with UCxSTE active high: Slave enabled when UCxSTE = 1 10b = 4-pin SPI with UCxSTE active low: Slave enabled when UCxSTE = 0 11b = I2C mode
8	UCSYNC	RW	0h	Synchronous mode enable 0b = Asynchronous mode 1b = Synchronous mode
7-6	UCSSELx	RW	0h	eUSCI clock source select. These bits select the BRCLK source clock in master mode. UCxCLK is always used in slave mode. 00b = Reserved 01b = ACLK 10b = SMCLK 11b = SMCLK
5-2	Reserved	R	0h	Reserved
1	UCSTEM	RW	0h	STE mode select in master mode. This byte is ignored in slave or 3-wire mode. 0b = STE pin is used to prevent conflicts with other masters 1b = STE pin is used to generate the enable signal for a 4-wire slave
0	UCSWRST	RW	1h	Software reset enable 0b = Disabled. eUSCI reset released for operation. 1b = Enabled. eUSCI logic held in reset state.

SPI

- MSP432 SPI Bit Rate Control Register

EUCSI_nx_SPI->BRW
n = A or B
x = 0, 1, 2, 3

  - SPI transfer bit rate: $f_{BitClock} = f_{BRCLK}/UCBRx$

    - BRCLK: ACLK or SMCLK

Figure 23-6. UCAxBRW Register

15	14	13	12	11	10	9	8
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw
Modify only when UCSWRST = 1.							

Table 23-4. UCAxBRW Register Description

Bit	Field	Type	Reset	Description
15-0	UCBRx	RW	0h	Bit clock prescaler setting

SPI

- MSP432 SPI Status Register

EUCSI_nx_SPI->STATW
n = A or B
x = 0, 1, 2, 3

Figure 23-7. UCAxSTATW Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCLISTEN	UCFE	UCOE	Reserved				UCBUSY
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	r-0
	Modify only when UCSWRST = 1.						

Table 23-5. UCAxSTATW Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved
7	UCLISTEN	RW	0h	Listen enable. The UCLISTEN bit selects loopback mode. 0b = Disabled 1b = Enabled. The transmitter output is internally fed back to the receiver.
6	UCFE	RW	0h	Framing error flag. This bit indicates a bus conflict in 4-wire master mode. UCFE is not used in 3-wire master or any slave mode. 0b = No error 1b = Bus conflict occurred
5	UCOE	RW	0h	Overrun error flag. This bit is set when a character is transferred into UCxRXBUF before the previous character was read. UCOE is cleared automatically when UCxRXBUF is read, and must not be cleared by software. Otherwise, it does not function correctly. 0b = No error 1b = Overrun error occurred
4-1	Reserved	RW	0h	Reserved
0	UCBUSY	R	0h	eUSCI busy. This bit indicates if a transmit or receive operation is in progress. 0b = eUSCI inactive 1b = eUSCI transmitting or receiving

SPI

- MSP432 SPI Receive Buffer

EUCSI_nx_SPI->RXBUF
n = A or B
x = 0, 1, 2, 3

Figure 23-8. UCAxRXBUF Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCRXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

Table 23-6. UCAxRXBUF Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved
7-0	UCRXBUFx	R	0h	The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCxRXBUF resets the receive-error bits and UCRXIFG. In 7-bit data mode, UCxRXBUF is LSB justified and the MSB is always reset.

SPI

- MSP432 SPI Transmit Buffer

EUCSI_nx_SPI->TXBUF
n = A or B
x = 0, 1, 2, 3

Figure 23-9. UCAxTXBUF Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCTXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

Table 23-7. UCAxTXBUF Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved
7-0	UCTXBUFx	RW	0h	The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted. Writing to the transmit data buffer clears UCTXIFG. The MSB of UCxTXBUF is not used for 7-bit data and is reset.

SPI

- MSP432 SPI Interrupt Enable Register

EUCSI_nx_SPI->IE
n = A or B
x = 0, 1, 2, 3

Figure 23-10. UCAxIE Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved						UCTXIE	UCRXIE
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0

Table 23-8. UCAxIE Register Description

Bit	Field	Type	Reset	Description
15-2	Reserved	R	0h	Reserved
1	UCTXIE	RW	0h	Transmit interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
0	UCRXIE	RW	0h	Receive interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

SPI

- MSP432 SPI Interrupt Flag Register

EUCSI_nx_SPI->IFG
n = A or B
x = 0, 1, 2, 3

Figure 23-11. UCAxIFG Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved						UCTXIFG	UCRXIFG
r-0	r-0	r-0	r-0	r-0	r-0	rw-1	rw-0

Table 23-9. UCAxIFG Register Description

Bit	Field	Type	Reset	Description
15-2	Reserved	R	0h	Reserved
1	UCTXIFG	RW	1h	Transmit interrupt flag. UCTXIFG is set when UCxxTXBUF empty. 0b = No interrupt pending 1b = Interrupt pending
0	UCRXIFG	RW	0h	Receive interrupt flag. UCRXIFG is set when UCxxRXBUF has received a complete character. 0b = No interrupt pending 1b = Interrupt pending

SPI

- MSP432 SPI Interrupt Vector Register

EUCSI_nx_SPI->IV
n = A or B
x = 0, 1, 2, 3

Figure 23-12. UCAxIV Register

15	14	13	12	11	10	9	8
UCIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCIVx							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 23-10. UCAxIV Register Description

Bit	Field	Type	Reset	Description
15-0	UCIVx	R	0h	eUSCI interrupt vector value 000h = No interrupt pending 002h = Interrupt Source: Data received; Interrupt Flag: UCRXIFG; Interrupt Priority: Highest 004h = Interrupt Source: Transmit buffer empty; Interrupt Flag: UCTXIFG; Interrupt Priority: Lowest

SPI

- MSP432 SPI Registers
  - Module B similar to module A

Table 23-11. eUSCI_B SPI Registers

Offset	Acronym	Register Name	Section
00h	UCBxCTLW0	eUSCI_Bx Control Word 0	Section 23.5.1
00h	UCBxCTL1	eUSCI_Bx Control 1	
01h	UCBxCTL0	eUSCI_Bx Control 0	
06h	UCBxBRW	eUSCI_Bx Bit Rate Control Word	Section 23.5.2
06h	UCBxBR0	eUSCI_Bx Bit Rate Control 0	
07h	UCBxBR1	eUSCI_Bx Bit Rate Control 1	
08h	UCBxSTATW	eUSCI_Bx Status	Section 23.5.3
0Ch	UCBxRXBUF	eUSCI_Bx Receive Buffer	Section 23.5.4
0Eh	UCBxTXBUF	eUSCI_Bx Transmit Buffer	Section 23.5.5
2Ah	UCBxIE	eUSCI_Bx Interrupt Enable	Section 23.5.6
2Ch	UCBxIFG	eUSCI_Bx Interrupt Flag	Section 23.5.7
2Eh	UCBxIV	eUSCI_Bx Interrupt Vector	Section 23.5.8

SPI

- MSP432 SPI Config

PORT						PSEL[1:0]	PORT						DIR	PSEL[1:0]
P1.0	UC	A	0	STE		01	P1.4	UC	B	0	STE			01
P1.1				CLK			P1.5				CLK			
P1.2				SOMI			P1.7				SOMI			
P1.3				SIMO			P1.6				SIMO			
P2.0	UC	A	1	STE		01	P6.2	UC	B	1	STE			01
P2.1				CLK			P6.3				CLK			
P2.2				SOMI			P6.5				SOMI			
P2.3				SIMO			P6.4				SIMO			
P3.0	UC	A	2	STE		01	P3.4	UC	B	2	STE			01
P3.1				CLK			P3.5				CLK			
P3.2				SOMI			P3.7				SOMI			
P3.3				SIMO			P3.6				SIMO			
P9.4	UC	A	3	STE		01	P10.0	UC	B	3	STE			01
P9.5				CLK			P10.1				CLK			
P9.6				SOMI			P10.3				SOMI			
P9.7				SIMO			P10.2				SIMO			
							P8.0	UC	B	3	STE			01
							P8.1				CLK			01
							P6.7				SOMI			10
							P6.6				SIMO			10

eUSCI_SPI Info Sheet		
n=3:0	x=A or B	
EUSCI_xn_SPI ->	CTLW0	Control Word
	BRW	Bit rate divider
	STATW	Status
	RXBUF	RX Buffer
	TXBUF	TX Buffer
	IE	Interrupt Enable
	IFG	Interrupt Flags
	IV	Interrupt Vector

EUSCIA0_IRQHandler	INTISR[16]	eUSCI_A0	UART or SPI mode TX, RX, and Status Flags
EUSCIA1_IRQHandler	INTISR[17]	eUSCI_A1	UART or SPI mode TX, RX, and Status Flags
EUSCIA2_IRQHandler	INTISR[18]	eUSCI_A2	UART or SPI mode TX, RX, and Status Flags
EUSCIA3_IRQHandler	INTISR[19]	eUSCI_A3	UART or SPI mode TX, RX, and Status Flags
EUSCIB0_IRQHandler	INTISR[20]	eUSCI_B0	SPI or I2C mode TX, RX, and Status Flags (I2C in multiple-slave mode)
EUSCIB1_IRQHandler	INTISR[21]	eUSCI_B1	SPI or I2C mode TX, RX, and Status Flags (I2C in multiple-slave mode)
EUSCIB2_IRQHandler	INTISR[22]	eUSCI_B2	SPI or I2C mode TX, RX, and Status Flags (I2C in multiple-slave mode)
EUSCIB3_IRQHandler	INTISR[23]	eUSCI_B3	SPI or I2C mode TX, RX, and Status Flags (I2C in multiple-slave mode)

SPI

- MSP432 SPI Config

SPI

- MSP432 SPI
  - Loopback example
  - Note – EUSCI_B3 pins are wired up to the onboard RGB LEDs

```
/*
 * spi_example.c
 *
 *  Created on: Aug 7, 2018
 *      Author: johnsontimoj
 */
//////////////////////////////////
//
// use 2 spi interfaces to loopback
//
// input none
//
// output - printf of spi register values
//
// Transfer a count value back and forth
//
//////////////////////////////////

// includes
#include <stdio.h>
#include "msp432.h"
#include "msoe_lib_all.h"

void pin_setup(void);
void spi_master_setup(void);
void spi_slave_setup(void);
```

SPI

- MSP432 SPI
  - Loopback example

```
int main(void){
    //
    // setup counter
    uint8_t count = 0;

    // Configure pins
    pin_setup();

    // Setup SPIs
    spi_master_setup();
    spi_slave_setup();

    while(1){
        // Load slave with a value to return
        EUSCI_B3_SPI->TXBUF = 0xFF - count;
        // Write to master buffer to initiate transfer
        EUSCI_A1_SPI->TXBUF = count;
        // wait for transfer to complete
        Delay_3MHz_ms(10);
        // Print rcvd values
        printf("Master sent %i - Slave sent %i\n", EUSCI_B3_SPI->RXBUF, EUSCI_A1_SPI->RXBUF);
        // Update count and wait
        count ++;
        Delay_3MHz_ms(1000);
    }
    return 0;
} // end main
```

SPI

- MSP432 SPI
  - Loopback example

```
void pin_setup(void){
    // Setup USCIA1 as master
    // Setup USCIB0 as slave
    // Transfer data from Master to slave in a while loop
    //
    // USCIA1 --> P2.0-STE, P2.1- CLK, P2.2-SOMI, P2.3-SIMO
    // USCIB3 --> P10.0-STE, P10.1-CLK, P10.3-SOMI, P10.2-SIMO
    //
    P2->SEL0 |= 0x0F;   // Set to USC mode
    P2->SEL1 &= ~0x0F;  // 0-1 mode
    P10->SEL0 |= 0x0F;
    P10->SEL1 &= ~0x0F;

    // Nothing else needed - the module sets directions
    return;
} // end pin_setup
```

SPI

- MSP432 SPI
  - Loopback exampl

```
void spi_master_setup(void){
    // USCIA1 - master
    // CKPL-0, CKPH-0(trailing), 8 bit, LSB first
    //
    // CTLW0
    // PH(trailing)  PL  LSB   8b   master  4/ssb   sync  Aclk     ss    rst
    // 0              0   0    0       1      10     1    01  xxxx  1     1
    // 0D42
    EUSCI_A1_SPI->CTLW0 = 0x0D43;
    //
    // BRW
    // 0000 0000 0100 0000
    // 0040
    EUSCI_A1_SPI->BRW = 0x0040;
    //
    // No interrupts for now
    //
    // Release reset
    EUSCI_A1_SPI->CTLW0 &= ~0x0001;

    return;
}   // end spi_master_setup

void spi_slave_setup(void){
    // USCIB3 - master
    // CKPL-0, CKPH-0(trailing), 8 bit, LSB first
    //
    // CTLW0
    // PH(trailing)  PK  LSB   8b   slave  4/ssb   sync  Aclk      ss  no rst
    // 0             0    0    0       0      10     1    01  xxxx  1     0
    // 0542
    EUSCI_B3_SPI->CTLW0 = 0x0542;
    //
    // No interrupts for now

    return;
}   // end spi_slave_setup
```

SPI

- MSP432 SPI
  - Loopback example

```
Class_Project:CIO
Master sent 44 - Slave sent 211
Master sent 45 - Slave sent 210
Master sent 46 - Slave sent 209
Master sent 47 - Slave sent 208
Master sent 48 - Slave sent 207
Master sent 49 - Slave sent 206
Master sent 50 - Slave sent 205
Master sent 51 - Slave sent 204
```