MSP430 Advanced Technical Conference 2006

In-Depth with MSP430's New Communication Interfaces

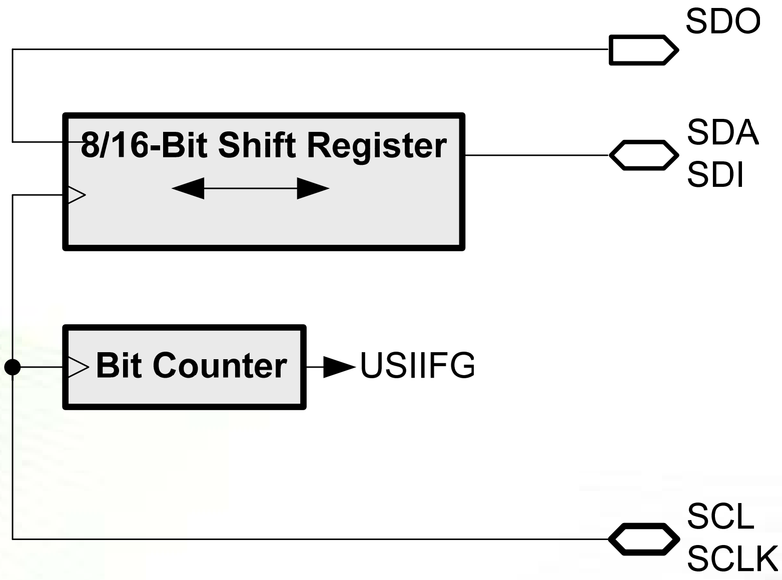
Volker Rzehak
MSP430 Systems Engineer
Texas Instruments

Agenda

- USCI vs. USI
- Introduction to USI
- USI communication modes
- Introduction to USCI
- USCI communication modes
- What module for what purpose

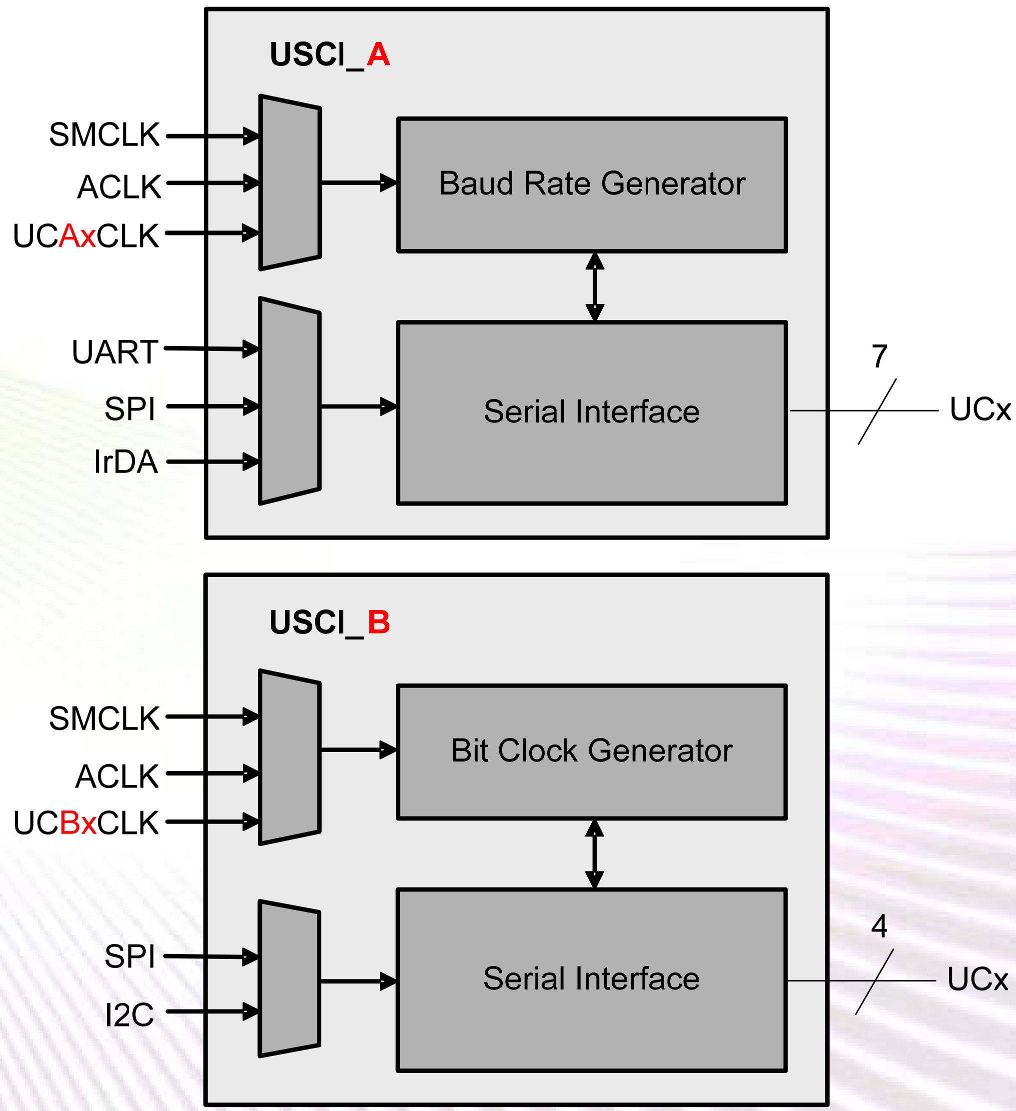
What is the USI?

- **Universal Serial Interface**
- **SPI Mode**
  - 8/16-bit shift register
  - MSB/LSB first
- **I²C Mode Support**
  - START/STOP detection
  - Arbitration lost detection
- **Interrupt Driven**
- **Reduces CPU load**
- **In low pin-count 2xx devices**



What is the USCI?

- **Universal Serial Communication Interface**
- **Two independent blocks**
- **USCI_A:**
  - UART
  - UART with automatic Baud rate detection (LIN support)
  - IrDA (SIR - Slow InfraRed)
  - SPI (Master & Slave, 3 & 4 wire)
- **USCI_B:**
  - I2C (Master & Slave modes)
  - SPI (Master & Slave, 3 & 4 wire)
- **In high-end 2xx and 4xx devices**

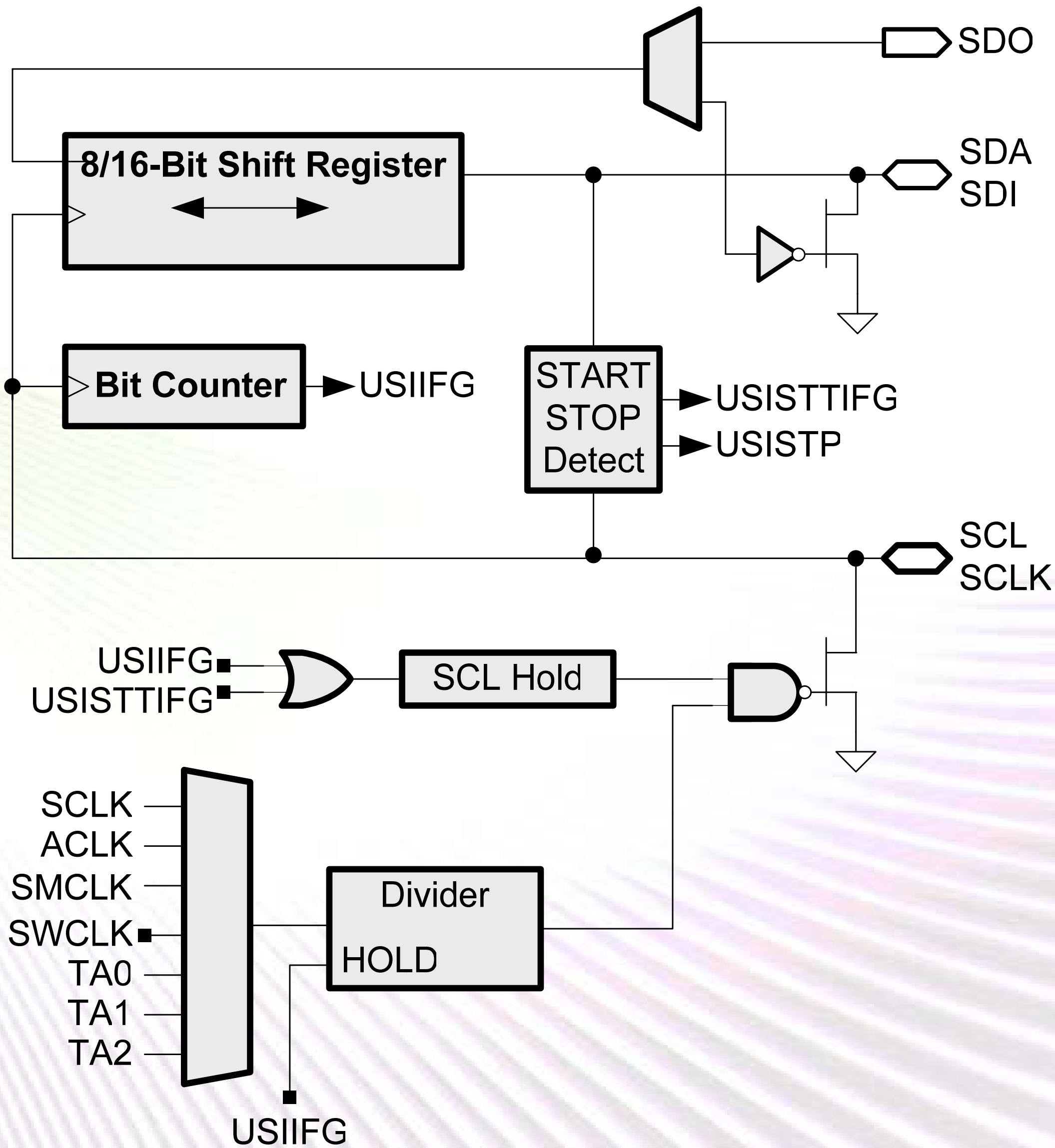


Agenda

- USCI vs. USI
- Introduction to USI
- USI communication modes
- Introduction to USCI
- USCI communication modes
- What module for what purpose

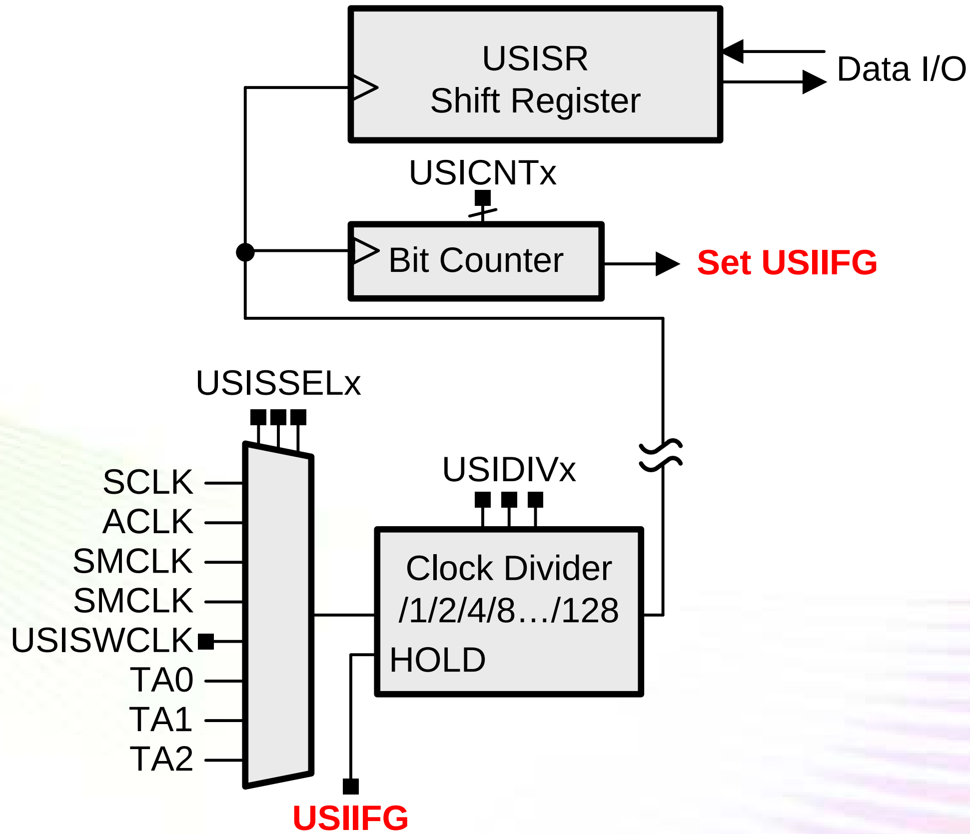
USI: Close-up

- Clock Control
- Data I/O
- USI Interrupts
- Pin Functionality



Data I/O

- Data shift register: up to 16 bits supported
- Number of bits TX'd & RX'd controlled by bit counter
- TX & RX is simultaneous
- Data I/O is user-defined: MSB or LSB first
- Selectable phase and polarity
- Bit counter automatically stops clocking after last bit & sets interrupt flag
- No data buffering!



USI Interrupts

USIIFG (used for SPI & I2C)

- Set when bit counter counts to zero
- Enabled by USIIE
- Cleared automatically when USICNTx is > 0 (USIIFGCC = 0)
- Stops clock when set

USISTTIFG (only used for I2C)

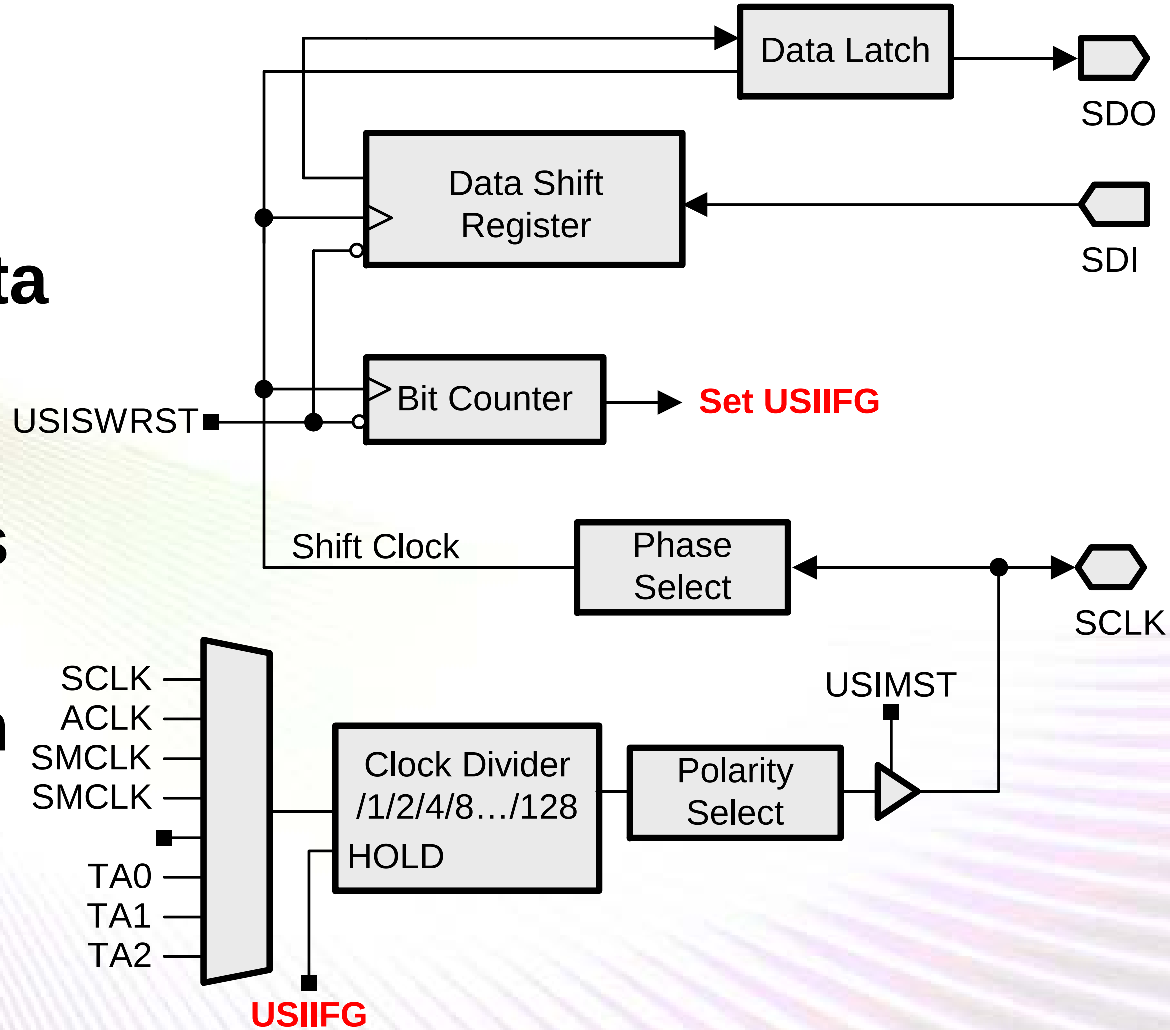
- Set when start condition detected
- Enabled by USISTTIFG
- Must be cleared in software
- Stops clock when set

Agenda

- USCI vs. USI
- Introduction to USI
- USI communication modes
- Introduction to USCI
- USCI communication modes
- What module for what purpose

USI SPI Communication

- Master or slave mode supported: USIMST
- USIPEx bits enable data & clock pins
- Port logic including interrupts functions as normal
- Data output latched on shift clock



SPI Initialization & Processing

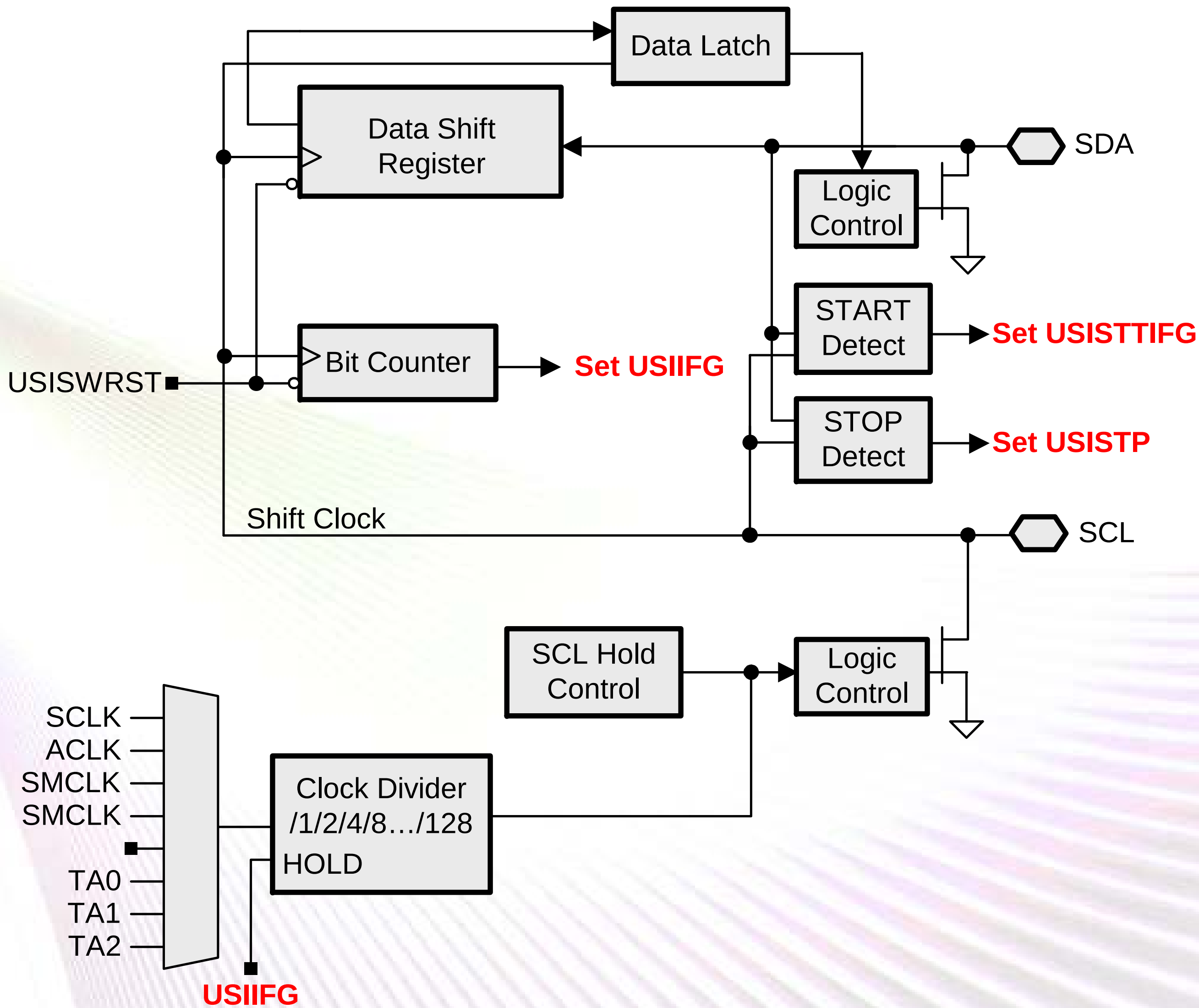
```
// Configure USI
USICTL0  = USIPE7+USIPE6+USIPE5+USIMST+USIOE;
USICTL1  = USIIE;                 // Enable USI interrupt
USICKCTL = USIDIV_4+USISSEL_2;   // Setup USI clocks: SMCLK/16
USICTL0  = USISWRST;              // Enable USI

while(1)
{_BIS_SR(LPM0_bits + GIE);}   // Enter LPM0 w/ interrupt
```

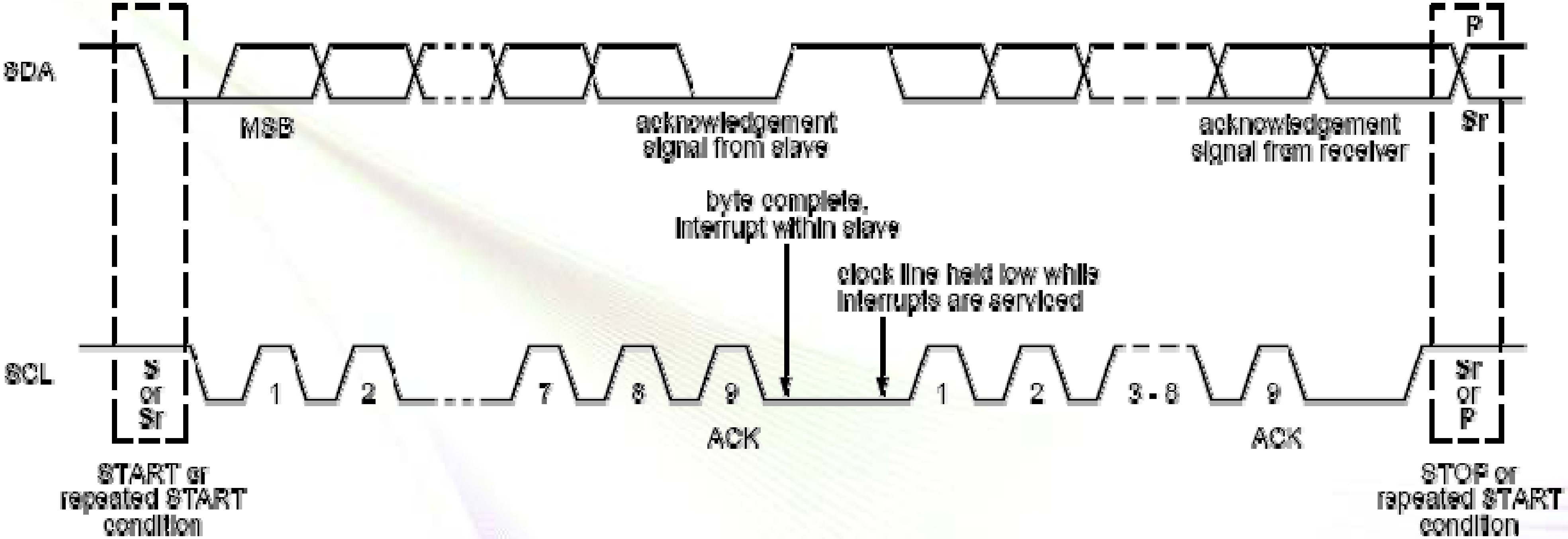
```
__interrupt void USI_TXRX (void)
{ SLV_Data = USISR;              // Store RX data
  USISR    = MST_Data;           // Load next TX data
  USICNT   = USI16B + 16;        // Load counter, clear USIIFG
}
```

USI I^2C Communication

- **Master & slave modes**
- **8-bit data MSB first**
- **Start & stop detection**
- **SCL control (divider > 1)**



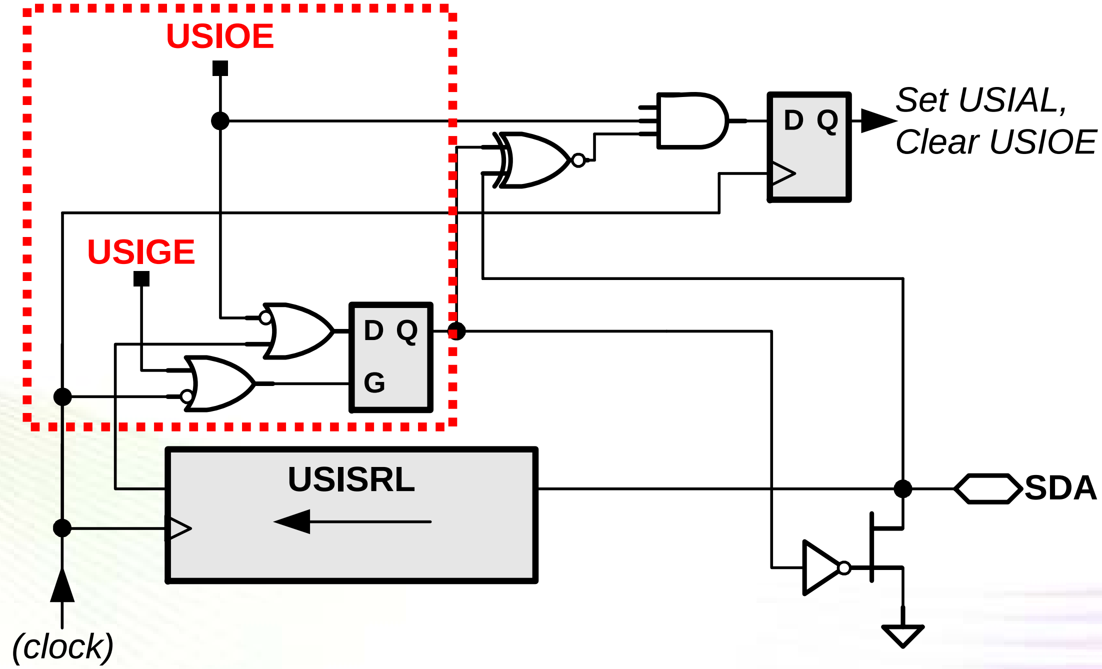
I²C Clocking and Data



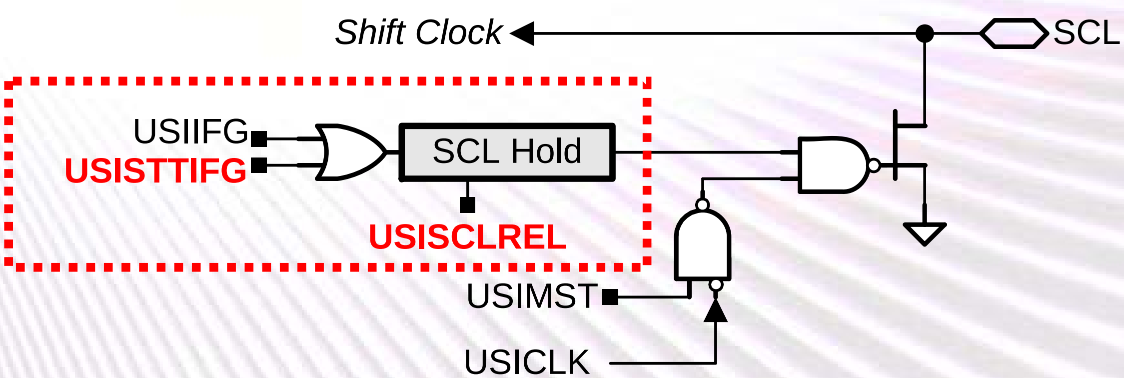
- **Start (or repeated Start)**
- **Address + R/W**
- **Slave Ack**
- **Data TX/RX + Ack for N bytes**
- **Stop (or repeated Start)**

I2C Data and Clock Handling

- **SDA Control**
  - Direction of SDA must also be controlled
  - Used for TX/RX, ACK/NACK handling and START/STOP generation
  - USIGE: Output latch control
  - USIOE: Data output enable



- **SCL Control**
  - SCL automatically held low in slave mode if USIIFG = 1 or USISTTIFG = 1 (Requires I2C compliant master supporting clock stretching)
  - SCL can be release by software with USISCLREL = 1



Start/Stop Generation: Master Only

- **Start**

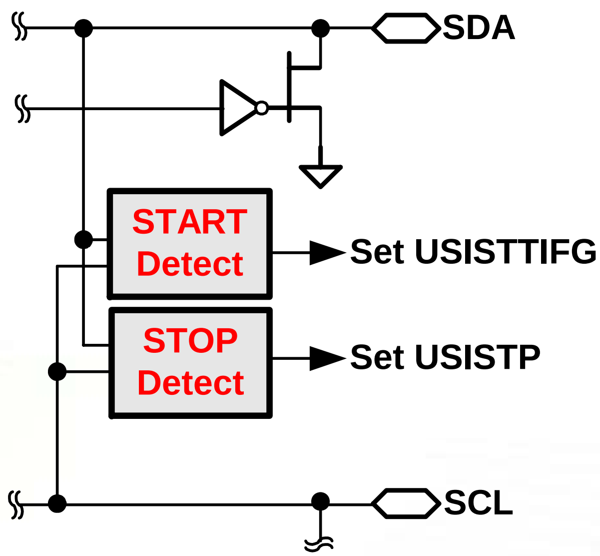
```
; Generate START
  MOV.B #000h,&USISRL             ; MSB = 0
  BIS.B #USIGE+USIOE,&USICTL0 ; Latch/SDA output enabled
  BIC.B #USIGE,&USICTL0           ; Latch disabled
  ...continue...
```

- **Stop**

```
; Generate STOP
  BIS.B #USIOE,&USICTL0           ; SDA=output
  MOV.B #000h,&USISRL             ; MSB = 0
  MOV.B #001h,&USICNT             ; USICNT = 1 for one clock
TEST_USIIFG
  BIT.B #USIIFG,&USICTL1          ; Test USIIFG
  JZ TEST_USIIFG                  ;
  MOV.B #0FFh,&USISRL             ; USISRL=1 to drive SDA high
  BIS.B #USIGE,&USICTL0           ; Transparent latch enabled
  BIC.B #USIGE+USIOE,&USICTL      ; Latch/SDA output disabled
  ...continue...
```

Start/Stop Detection

- **Slave mode START & STOP detection 100% in hardware**
- **USISTTIFG set on start**
  - Sources USI interrupt
- **USISTP set on stop**
  - CPU-accessible flag
- **SCL held low in hardware when USISTTIFG = 1 (requires an I2C compliant master supporting clock stretching)**



Receiver Ack/Nack Generation

```
;Generate (N)ACK
  BIS.B #USIOE,&USICTL0      ; SDA output

  MOV.B #00h,&USISRL         ; MSB = 0 -> Ack
              OR
  MOV.B #FFh,&USISRL         ; MSB = 1 -> NAck

  MOV.B #001H,&USICNT        ; USICNT = 1 for one clock
TEST_USIIFG
  BIT.B #USIIFG,&USICTL1     ; Test USIIFG
  JZ TEST_USIIFG
  ...continue...
```

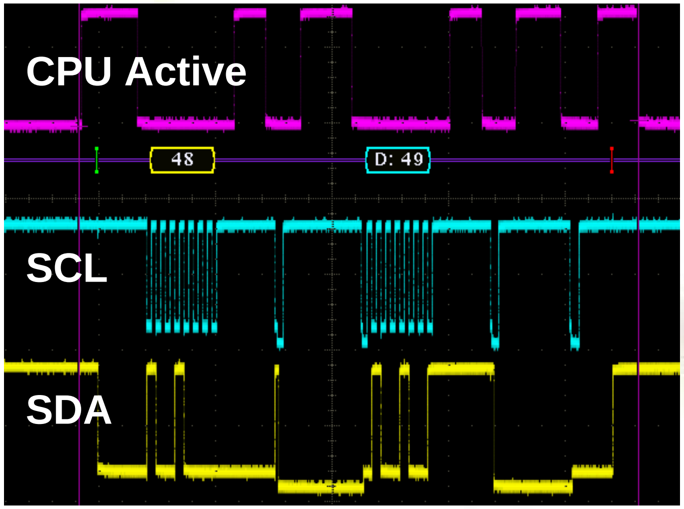
- After address/data reception
- SDA = output
- Output 1 data bit: 0 = Ack, 1 = NAck

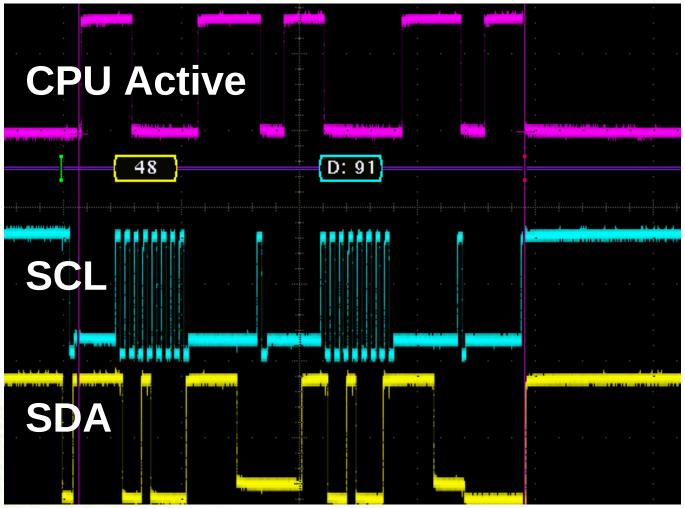
Transmitter Ack/Nack Detection

```
; Receive ACK/NACK
  BIC.B #USIOE,&USICTL0      ; SDA input
  MOV.B #01h,&USICNT         ; USICNTx = 1
TEST_USIIFG
  BIT.B #USIIFG,&USICTL1     ; Test USIIFG
  JZ TEST_USIIFG
  BIT.B #01h,&USISRL         ; Test received ACK bit
  JNZ HANDLE_NACK            ; Handle if NACK
  ...Else, handle ACK
```

- **After address/data transmission**
- **SDA = input**
- **Receive 1 data bit: 0 = Ack, 1 = NAck**

USI I^2C Master TX – Slave RX





1: Send Start, Address & R/W bit

2: Receive (N)Ack

3: Test (N)Ack & handle TX data

4: Receive (N)Ack

5: Test (N)Ack & prep Stop

6: Send Stop

1: Detect Start & receive address + R/W

2: Transmit (N)Ack

3: Data RX

4: Transmit (N)Ack

5: Reset for next Start

Typical USI I^2C CPU Loading

	SCL = 125kHz			SCL = 500kHz		
CPU Freq	1MHz	8MHz	16MHz	1MHz	8MHz	16MHz
USI Master						
CPU Loading	63%	18%	11%	87%	43%	28%
USI Slave						
CPU Loading	73%	23%	12%	90%	53%	30%

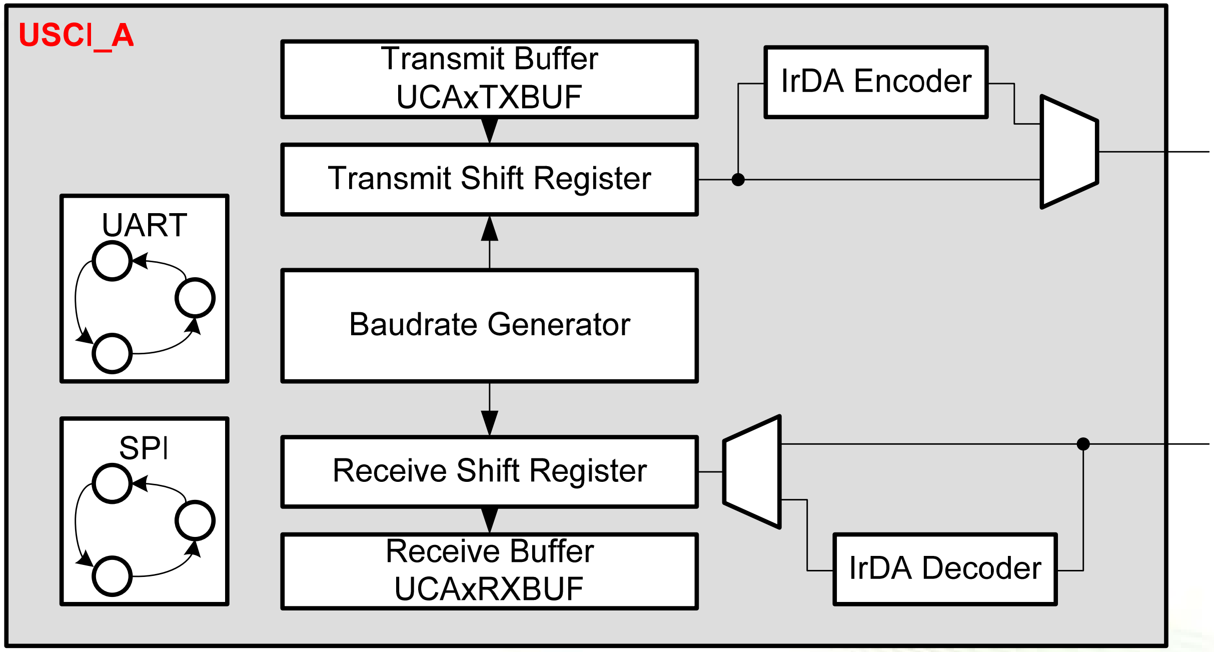
- **1 byte TX'd & 1 byte RX'd: MST to SLV**
- **Dependent on S/W protocol details**
- **125 & 500kHz used as simple example**

Agenda

- USCI vs. USI
- Introduction to USI
- USI communication modes
- Introduction to USCI
- USCI communication modes
- What module for what purpose

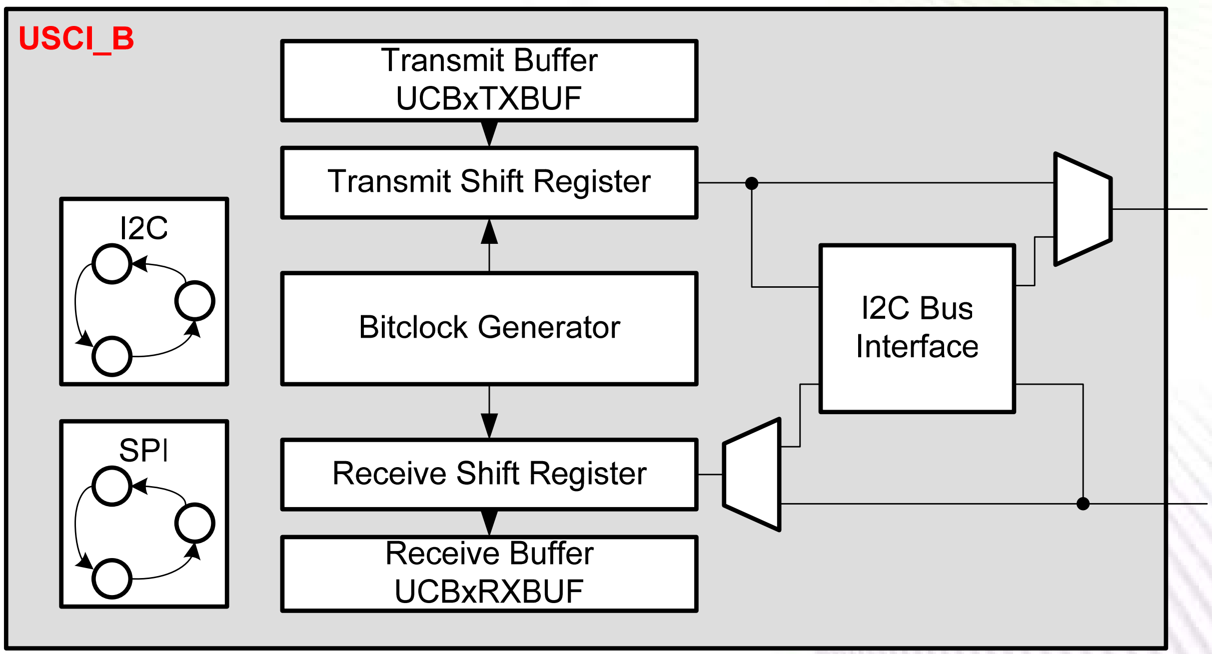
USCI – Features Overview

- **Two independent communication blocks**
- **Asynchronous communication modes**
  - UART standard and multiprocessor protocols
  - UART with automatic Baud rate detection (LIN support)
  - IrDA (SIR - slow Infrared, up to 115kBaud)
  - LPMx wake up
- **Synchronous communication modes**
  - SPI (Master & Slave modes, 3 & 4 wire)
  - I2C (Master & Slave modes)
  - LPMx operation
- **DMA enabled**
- **Interrupt driven**



USCI_A

- **UART**
  with IrDA/LIN support
  or SPI
- **Baud-rate generator with auto-baud rate detect**
- **Double buffered TX/RX**



USCI_B

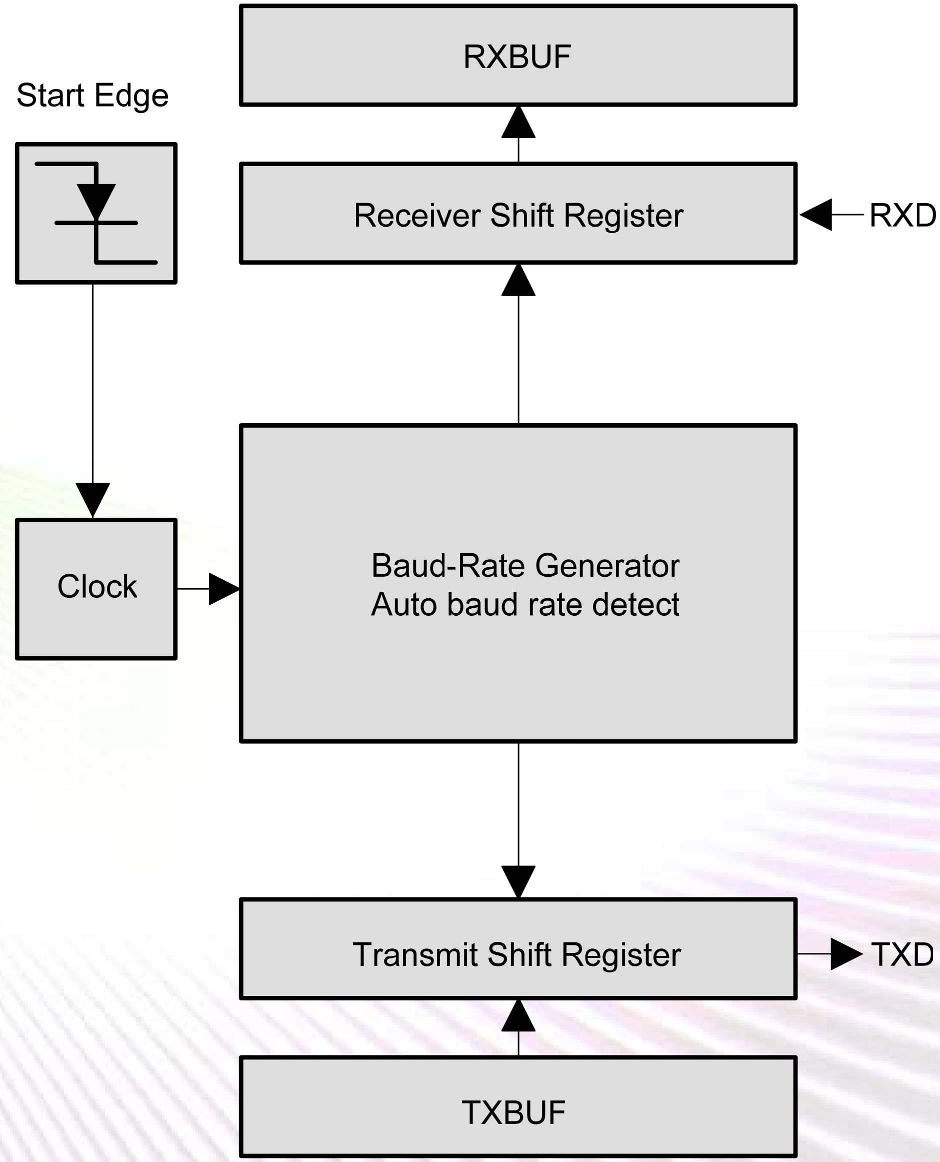
- **I2C master/slave**
  up to 400kHz
  or SPI
- **Bit clock generator**
- **Double buffered TX/RX**

Agenda

- USCI vs. USI
- Introduction to USI
- USI communication modes
- Introduction to USCI
- USCI communication modes
- What module for what purpose

USCI_A UART Mode

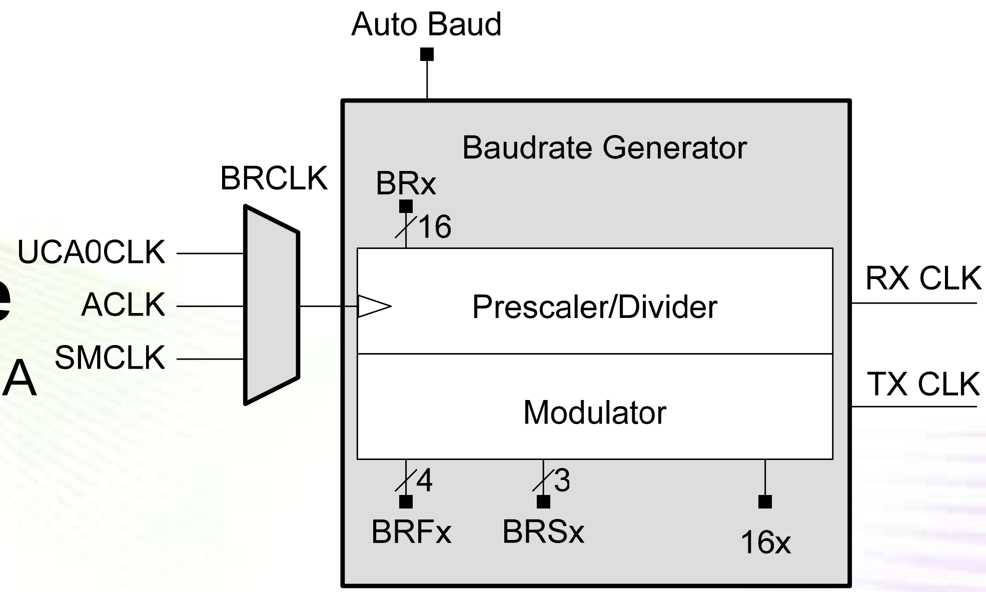
- **Ultra-low power**
- **Flexible**
  - 7- or 8-bit data
  - Odd, even or non-parity
  - LSB or MSB first
  - Glitch suppression
- **Communication Schemes**
  - IrDA
  - Idle Line Multi-processor
  - Address Bit Multi-processor
  - Auto baud (**S/W LIN**)
- **Interrupt Driven**
  - Error detection
  - TX/RX



UART Baud Rate Generator

- **Low power, low frequency mode**
  - Allows use of slower clocks
  - n/8 modulator
  - Max baud rate 1/3 BRCLK

- **16x over-sampling mode**
  - Quasi-standard for UART, LIN, IrDA
  - n/16 modulator
  - Max baud rate 1/16 BRCLK

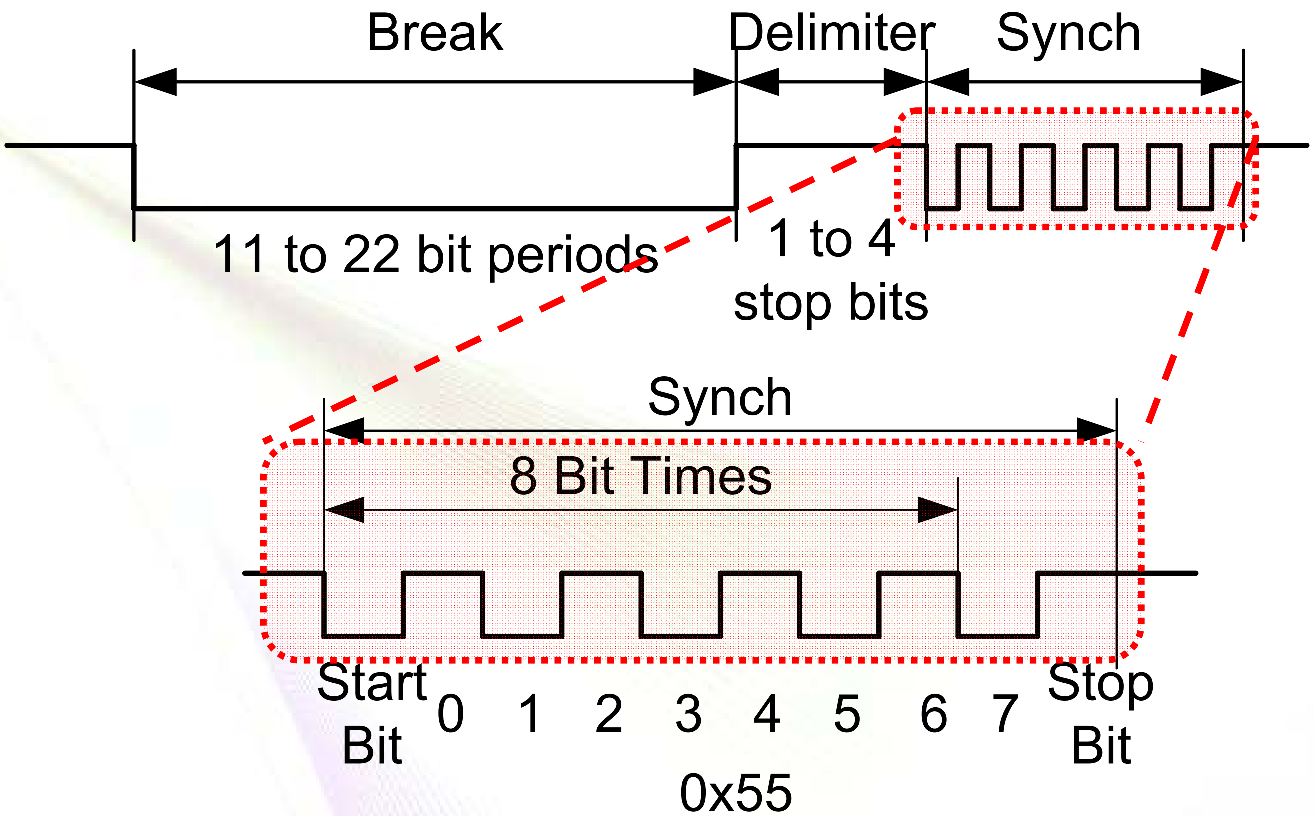


- **Flexible clock sources**
  - External UCA0CLK input
  - ACLK
  - SMCLK

UART Ultra-low Power Operation

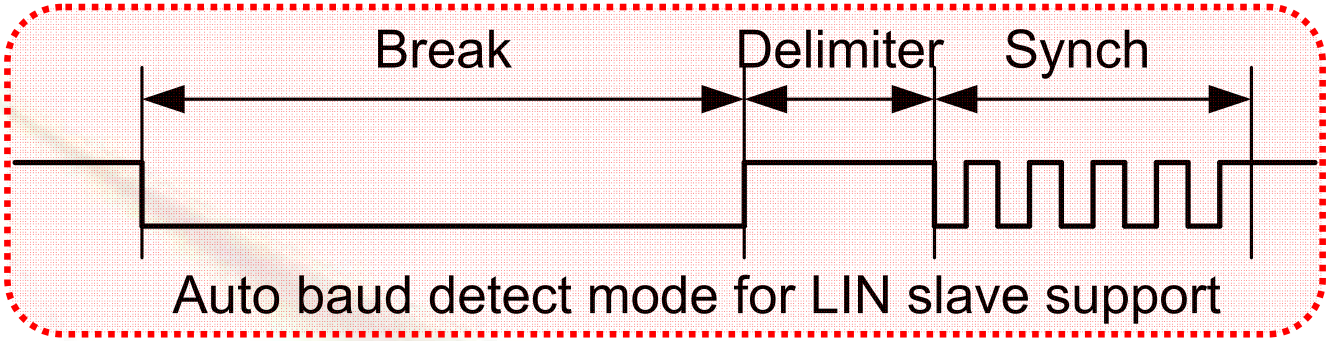
- Wake-up from LPMx
- RX start edge or writing to TXBUF automatically turns on the internal BRCLK source if turned off by LPMx
- BRCLK source off after end of TX or RX in LPMx
- No software handling required
- Instant-on DCO clock prevents loss of received char.
- SMCLK can support high baud rates in LPM3 mode!

UART Automatic Baud Rate Detect



- **Automatic baud rate detection protocol**
  - Baud rate calculated from a valid SYNC
  - Auto baud rate value stored in BR1, BR0 & modulator bits
  - BREAK time-out detect in hardware
  - Programmable delimiter time

UART LIN Support



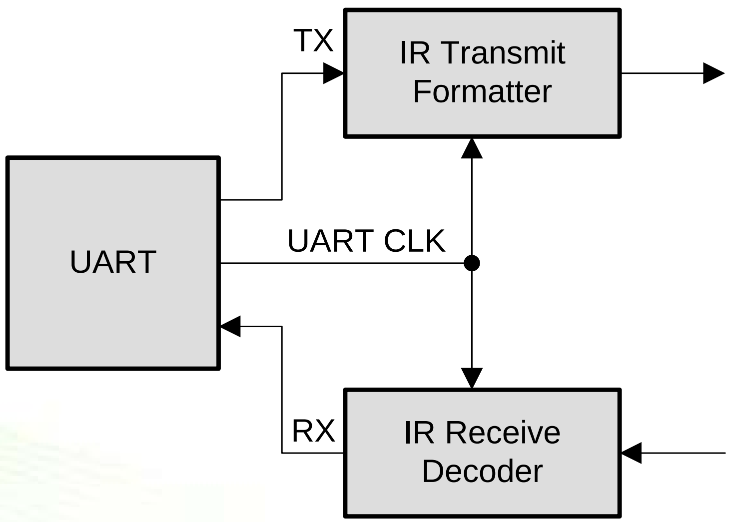
- **Local Interconnect Network is used in automotive**
- **UART automatic baud rate detect required**
- **LIN mode UART = 8-bits, LSB first, no parity, 1 stop**
- **LIN device driver is implemented in software**
- **External LIN bus line drivers required**
- **LIN slave mode support**
  - Break synch detection and automatic baud rate measurement is based on
  - received LIN break and synch field '0x55'

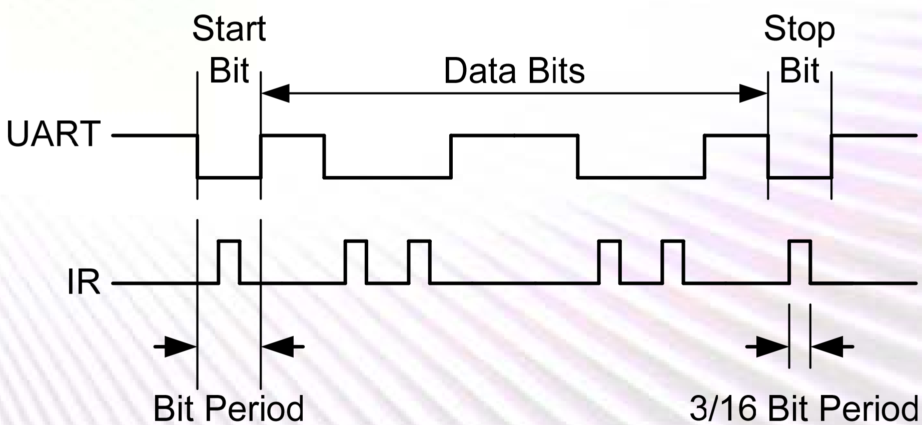
LIN Master Support

- **LIN Master mode support**
  - Automatic break-synch generation
- **S/W sequence transmits a break synch field required for LIN master mode**
  - 1. Select auto baud rate detect mode & UCTXBRK = 1
  - 2. Specify break delimiter bit length with DELIMx, default = 1 bit period
  - 3. Check if TXBUF ready & write LIN synch ‘0x55h’ to TXBUF
- **Break field of 13 bits followed by a break delimiter & the synch field are transmitted**
  - UCTXBRK is reset automatically after the synch loads into TX shift register
  - Further data written to TXBUF is transmitted normally

UART IrDA Support

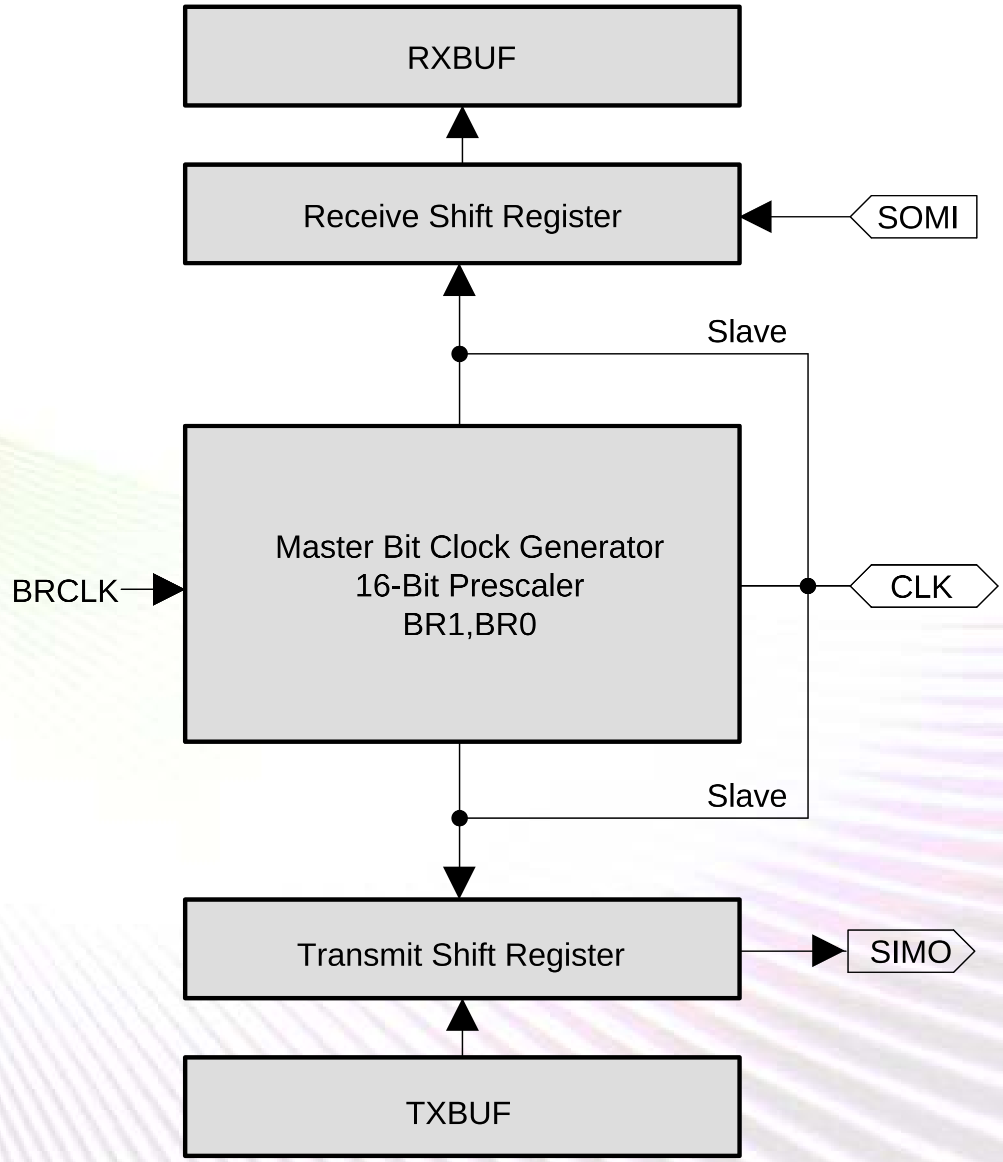
- Integrated IrDA encoder & decoder
- Directly connects to external IR analog
- Programmable digital filter stage for receive pulse filtering
- Programmable pulse length
- IrDA standard 3/16 bit period pulse supported
- IrDA protocol stack implemented in S/W



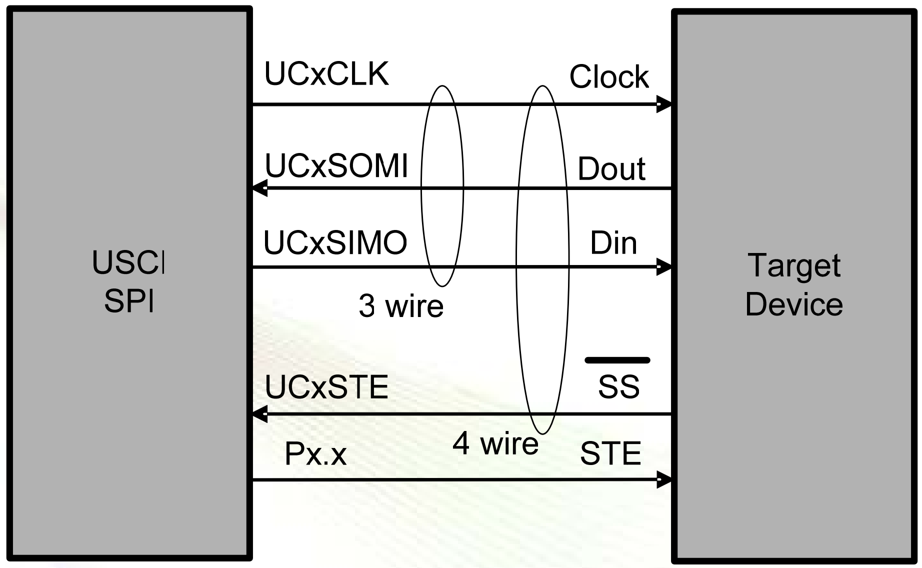


USCI_A & USCI_B SPI Mode

- **Flexible interface**
  - 3- or 4-pin SPI
  - 7- or 8-bit data length
  - Master or slave
  - LSB or MSB first
- **S/W configurable clock phase & polarity**
- **Programmable SPI master clock**
- **Double buffered TX/RX**
- **Interrupt driven TX/RX**
- **DMA enabled**
- **LPMx operation**



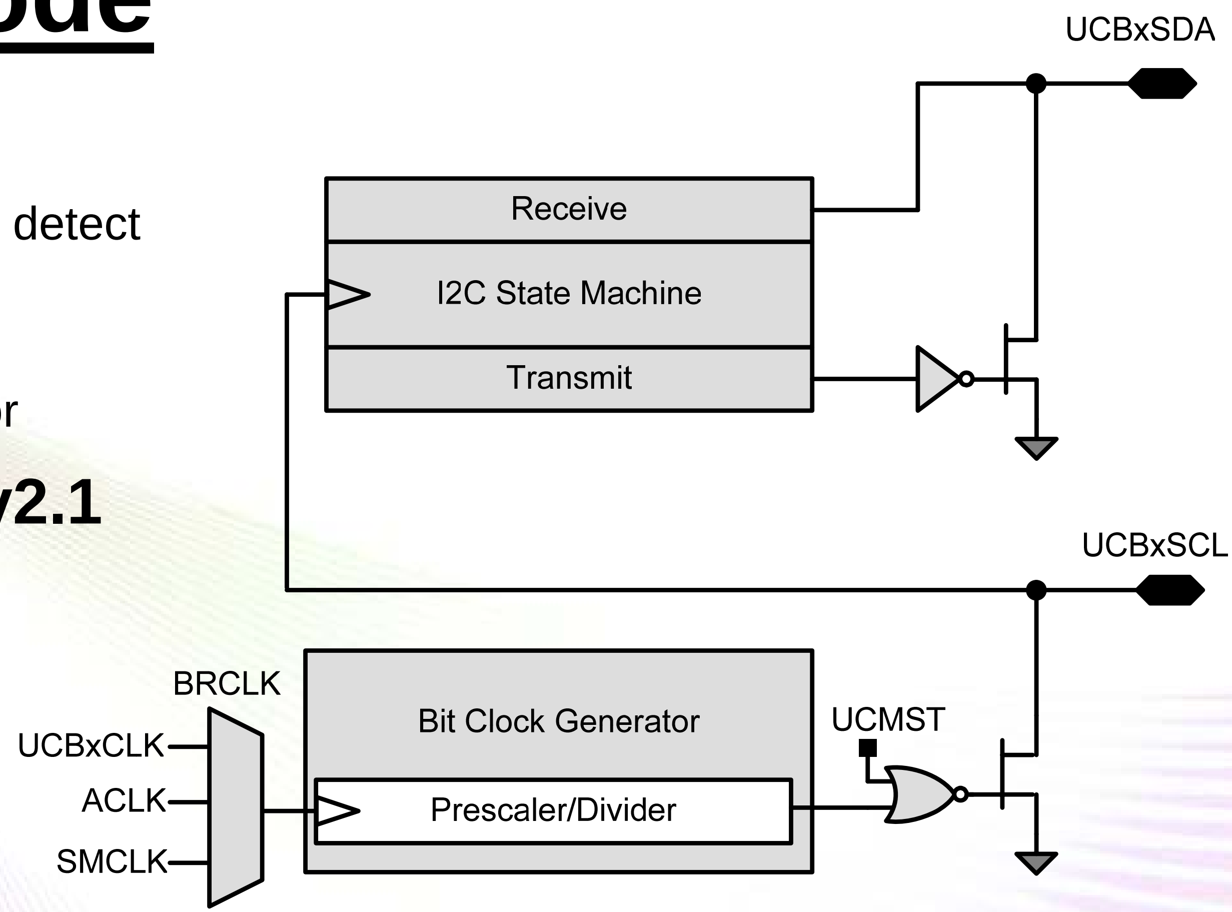
SPI Connectivity



- **3 wire mode supports master & slave modes**
- **4 wire master STE senses conflicts other master(s)**
- **In 4 wire slave, TX & RX are controlled externally using the STE**

USCI_B I²C Mode

- **Ultra-low power**
  - LPMx slave receiver START detect
  - Slave operation in LPM4
  - Flexible clock choice
  - Integrated bit clock generator
- **Philips specification v2.1**
  - 7- & 10-bit address
  - Multi-master support
  - Slave mode
  - Up to 400kbps
- **Interrupt driven**
  - No acknowledgement
  - Arbitration lost
  - Start/Stop condition
  - TX/RX



USCI I2C Master Receiver

```
UCB0CTL1 |= UCSWRST;                      // Enable SW reset
UCB0CTL0 = UCMST + UCMODE_3 + UCSYNC;     // I2C Master
UCB0CTL1 = UCSSEL_2 + UCSWRST;            // Use SMCLK, SW reset
UCB0BR0 = 12; UCB0BR1 = 0;              // fSCL=SMCLK/12=~100kHz
UCB0I2CSA = 0x4e;                         // Set slave address
UCB0CTL1 &= ~UCSWRST;                     // Clear SW reset
IE2 |= UCB0RXIE;                          // Enable RX interrupt
RxByteCtr = 2;                            // Load RX byte counter
UCB0CTL1 |= UCTXSTT;                      // I2C start condition
__bis_SR_register(CPUOFF + GIE);          // Enter LPM0 until
                                          // all data is RX'd
```

```
__interrupt void USCIAB0TX_ISR(void)
{ RxByteCtr--;                            // Decrement RX byte ctr
  if (RxByteCtr)
  { RxByte = UCB0RXBUF;                 // Get received byte
    if (RxByteCtr == 1)                   // Only one byte left?
      UCB0CTL1 |= UCTXSTP; }              // I2C stop condition
  else
  { RxByte = UCB0RXBUF;                   // Get final RX byte
    __bic_SR_register_on_exit(CPUOFF);    // Exit LPM0
  } }
```

USCI I2C Slave Transmitter

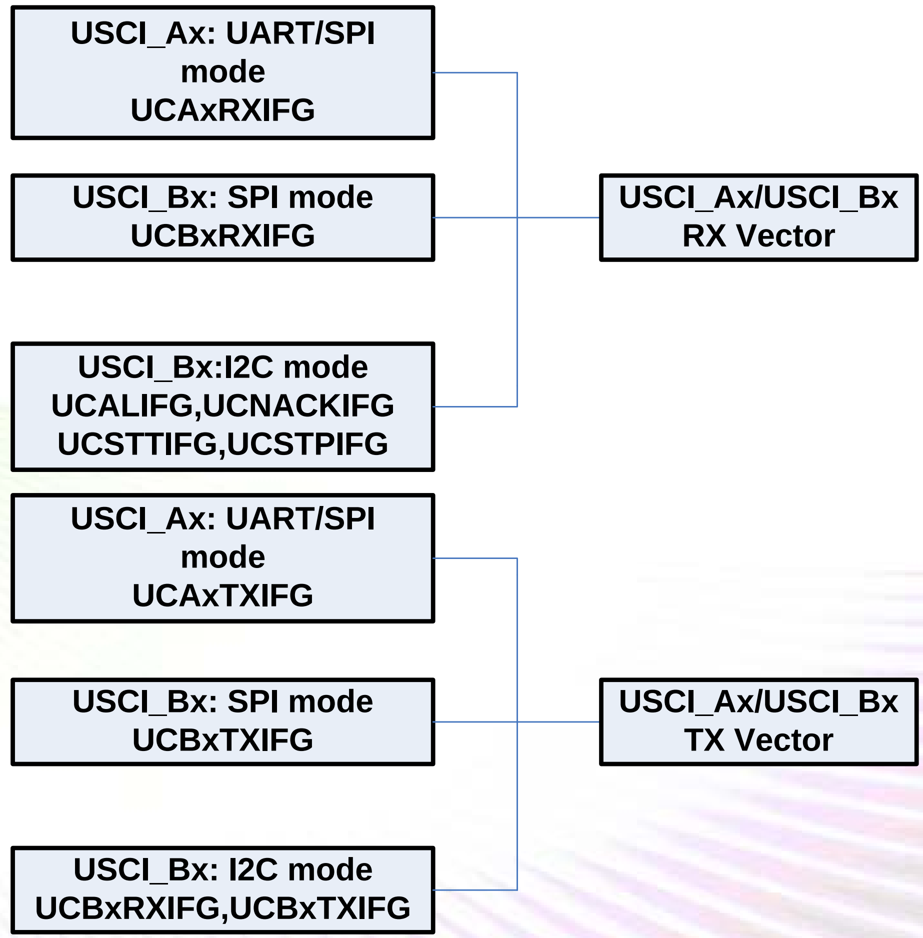
```
UCB0CTL1 |= UCSWRST;                    // Enable SW reset
UCB0CTL0 = UCMODE_3 + UCSYNC;           // I2C Slave
UCB0I2COA = 0x48;                       // Own Address is 048h
UCB0CTL1 &= ~UCSWRST;                   // Clear SW reset
UCB0I2CIE |= UCSTTIE;                   // Enable STT interrupt
IE2 |= UCB0TXIE;                        // Enable TX interrupt
TXData = 0xff;                          // Used to hold TX data
__bis_SR_register(CPUOFF + GIE);        // Enter LPM0
```

```
__interrupt void USCIAB0TX_ISR(void)
{
  UCB0TXBUF = TXData;                   // TX data
}
```

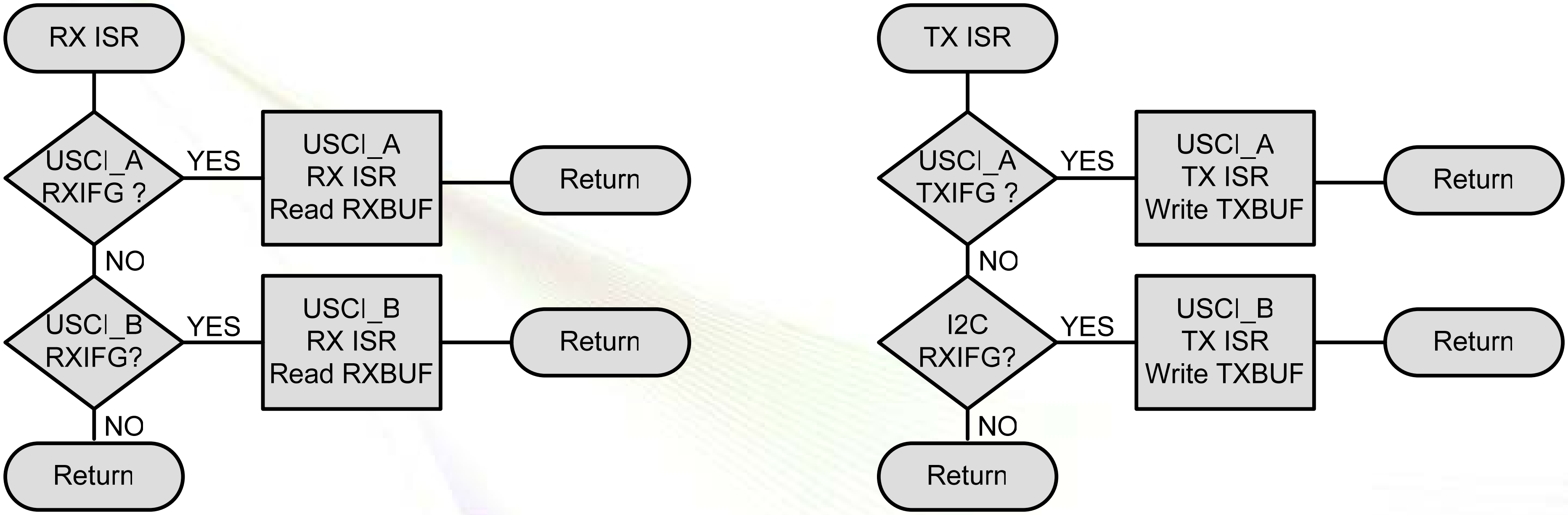
```
__interrupt void USCIAB0RX_ISR(void)
{
  UCB0STAT &= ~UCSTTIFG;                // Clear start cond. int flag
  TXData++;                             // Increment data
}
```

USCI Interrupts

- **USCI_A & USCI_B share TX & RX interrupt vectors**
- **USCI_A flags**
  - UART – TXIFG, RXIFG
  - SPI – TXIFG, RXIFG
- **USCI_B flags**
  - SPI – TXIFG, RXIFG
  - I2C State – ALIFG, NACKIFG,
    STTIFG, STPIFG
  - I2C Data – TXIFG, RXIFG

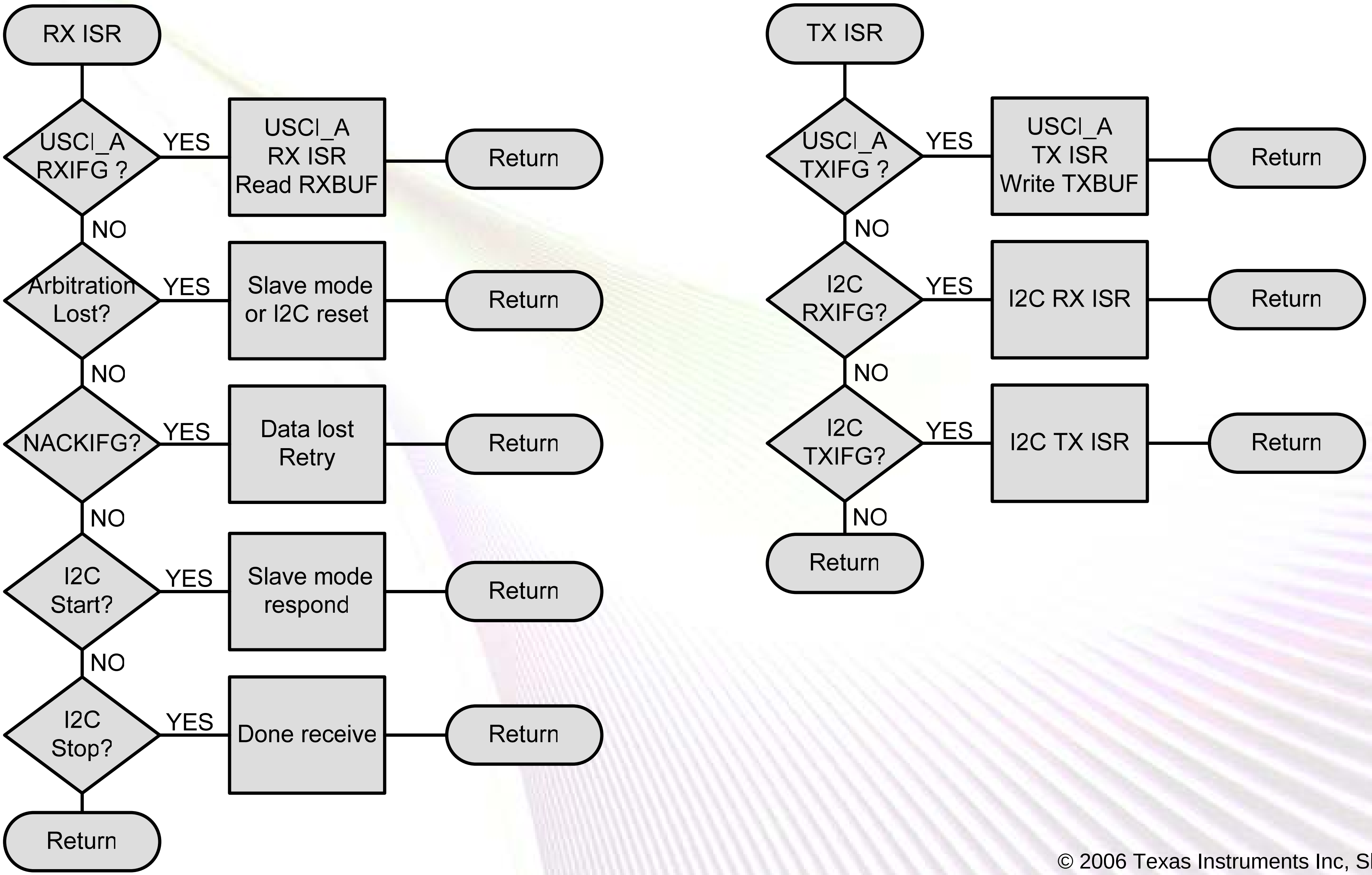


USCI SPI interrupts handling



- **TX & RX ISR recommended flow**
- **USCI_A & USCI_B share TX vector & RX vector**
- **Software check detects the correct ISR handle**

USCI I^2C Interrupt Handling



Why Doesn't the USCI Work?

```
SetupUSCIA  mov.b #UCSWRST,&UCA0CTL1    ; module reset
            bis.b #UCSSEL1,&UCA0CTL1    ; BRCLK=SMCLK
            mov.b #0x09,&UCA0BR0        ; 115k at 1048576Hz
            mov.b #0x00,&UCA0BR1        ;
            mov.b #0x08,&UCA0MCTL       ; modulation values
            bic.b #UCSWRST, &UCA0CTL1   ; clear module reset
            bis.b #UCA0RXIE, &IE2       ; enable RXinterrupt
```

- **Proper USCI_A Configuration Procedure**
  - **1. Set SWRST**
  - **2. Configure USCI registers**
  - **3. Clear SWRST**
  - **4. Enable interrupts if desired**

Agenda

- USCI vs. USI
- Introduction to USI
- USI communication modes
- Introduction to USCI
- USCI communication modes
- What bus for what purpose

USI System Benefits

- **USI provides SPI & I2C support in hardware**
  - Protocol in user S/W: Timing in USI H/W
- **SPI: up to 16MHz clocking & 16-bit data I/O**
- **I2C: interrupt-driven protocol, critical timing in H/W**
- **Provides flexibility of 100% S/W solution while maintaining timing in hardware**
- **Better than pure software implementations**
  - Faster communication speeds
  - Lower CPU loading
  - Smaller software

USCI System Benefits

- **Two independent blocks can operate simultaneously**
- **All modes capable of operating from any LPMx**
- **USCI is interrupt driven**
- **USCI is DMA enabled**
- **USCI_A supports SPI, UART, IrDA, auto-baud, LIN Bus**
- **USCI_A has integrated baud rate generator with modulator for fractional bit rate division support**
- **USCI_B supports SPI, I2C**

Summary

- **New devices typically implement either**
  - USI or
  - USCI
- **USI provides very basic communication means**
  - Timing in hardware, protocol in software
  - In low-pin 2xx devices
- **USCI provides complete feature in hardware**
  - High-end communication module
  - In high-end 2xx and 4xx devices

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
Low Power Wireless	www.ti.com/lpw	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265