Lecture 6: Clock and Communication Modules

TNE097 Micro Computer Systems

Naveen Venkategowda

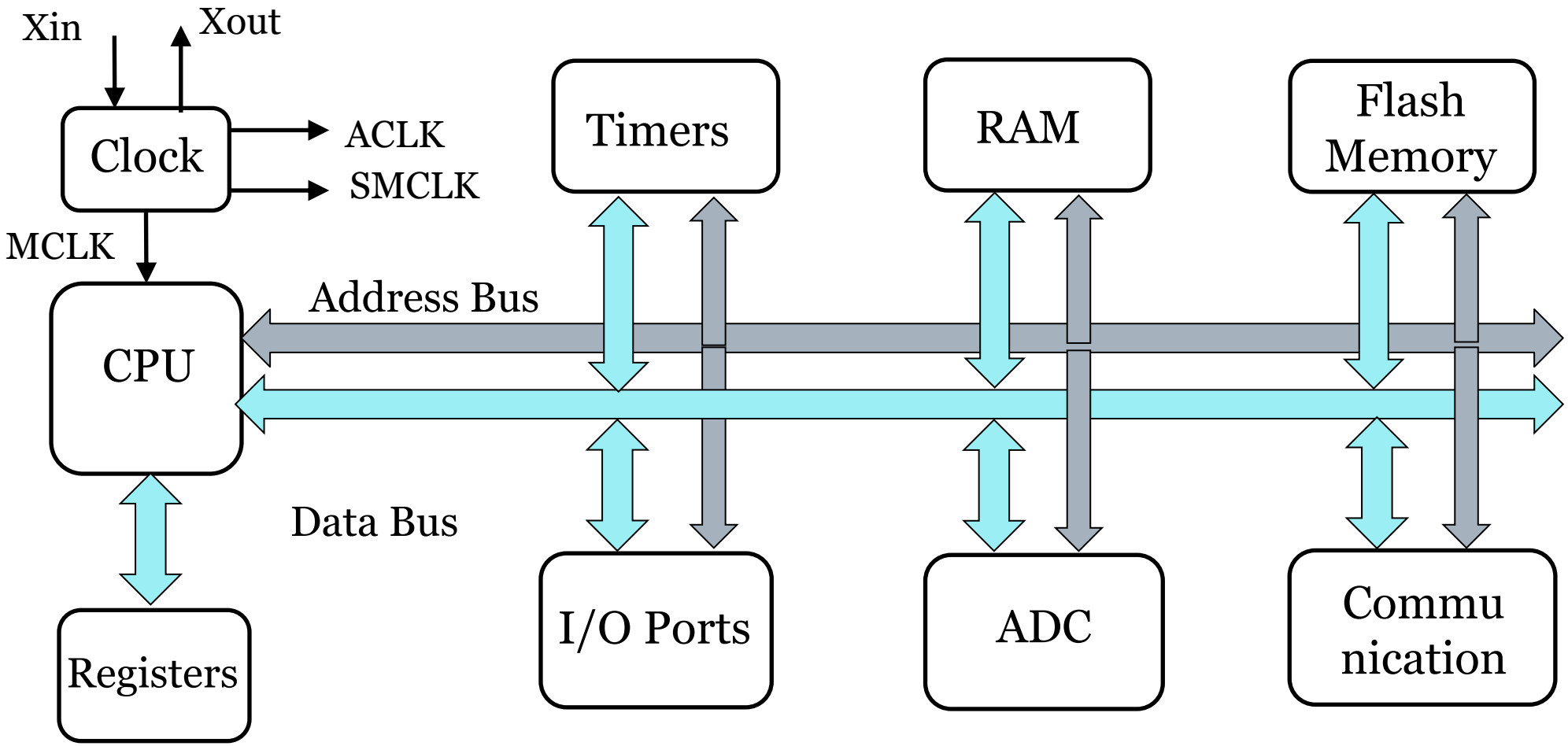
2025-11-27

LINKÖPING UNIVERSITY

Contents

Click on the section titles to directly skip to a section

Anatomy of MSP 430



Basic Clock Module

Choosing Clock Sources

- How to chose clock source (signals)?
  - Balance performance, power consumption, latency
- Conflicting requirements for clock sources:
  - Low clock frequency for energy saving and time keeping
  - High clock frequency for fast reaction times and fast burst processing capability
  - Stability over operating temperature and supply voltage

Clock Sources

- **LFXT1CLK**: Low-frequency/high-frequency oscillator
  - Can be used with low-frequency watch crystals or external clock of 32768 Hz
  - Used with standard crystals, resonators, external clock in 0.4 to 16 MHz range
- **XT2CLK**: Optional high-frequency oscillator
  - Used with standard crystals, resonators, external clock in 0.4 to 16 MHz range
- **DCOCLK**: Internal digitally controlled oscillator (DCO)
- **VLOCLK**: Internal very low power, low frequency oscillator with 12-kHz typical frequency

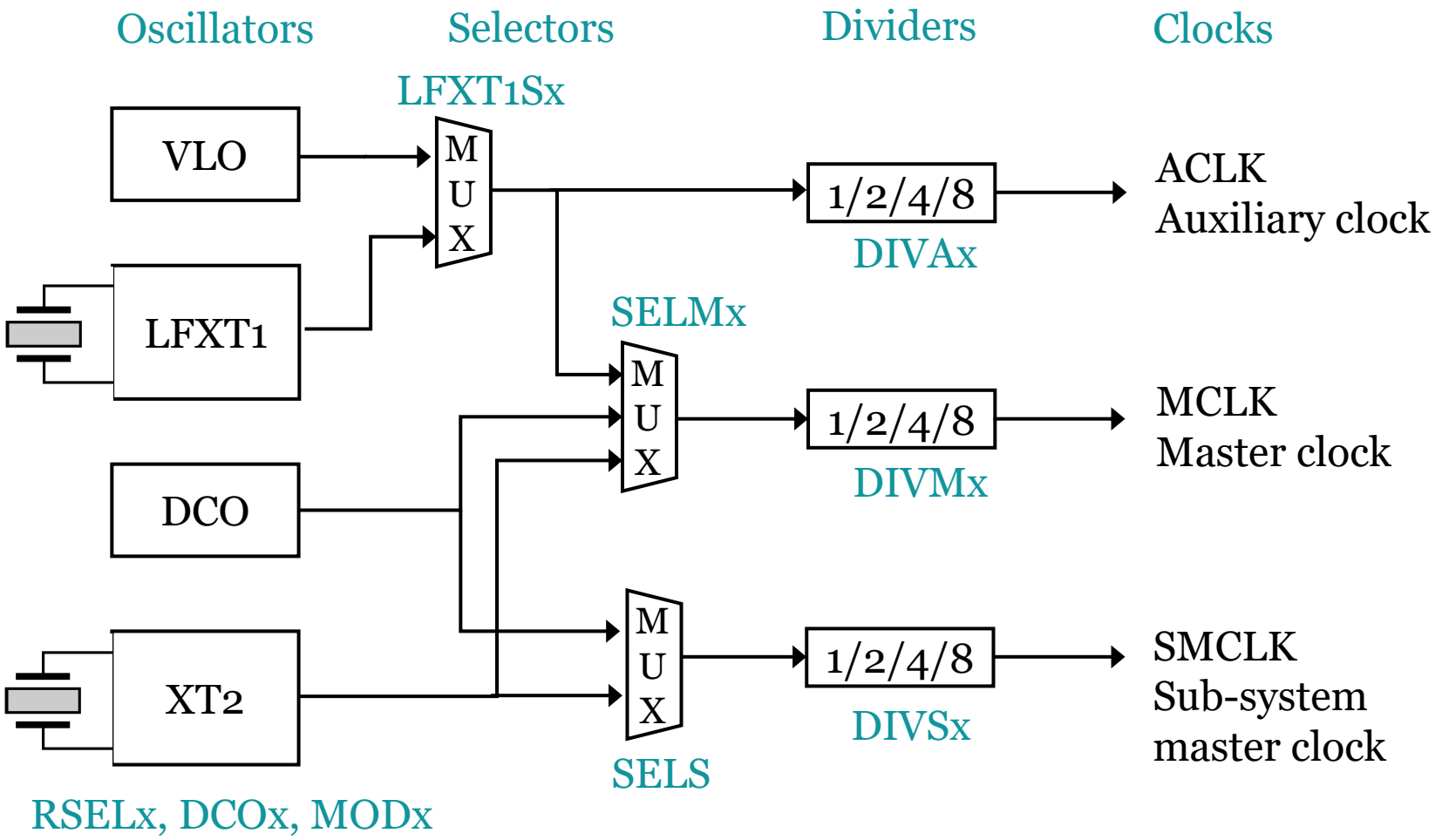
Clock Signals

- Master clock (MCLK)
  - Source: software selectable as LFXT1CLK, VLOCLK, XT2CLK (if available), or DCOCLK
  - Divider: 1, 2, 4, or 8
  - Usage: CPU and system
- Sub-main clock (SMCLK):
  - Source: software selectable as LFXT1CLK, VLOCLK, XT2CLK (if available on-chip), or DCOCLK.
  - Divider: 1, 2, 4, or 8
  - Usage: software selectable for individual peripheral modules.

Clock Signals

- Auxiliary clock (ACLK)
  - Source: software selectable as LFXT1CLK or VLOCLK
  - Divider: 1, 2, 4, or 8
  - Usage: Software selectable for individual peripheral modules

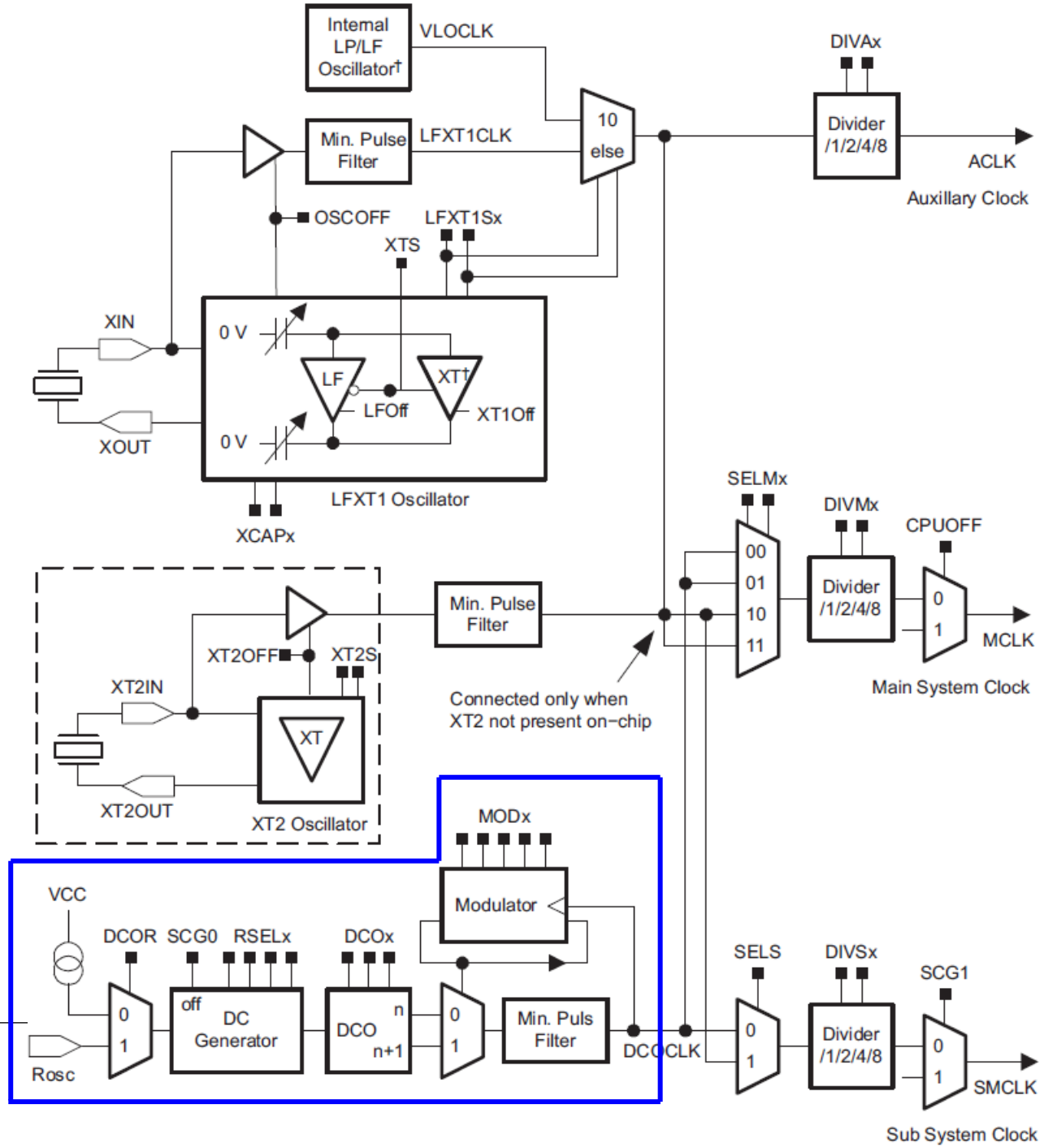
Block Diagram Clock Module



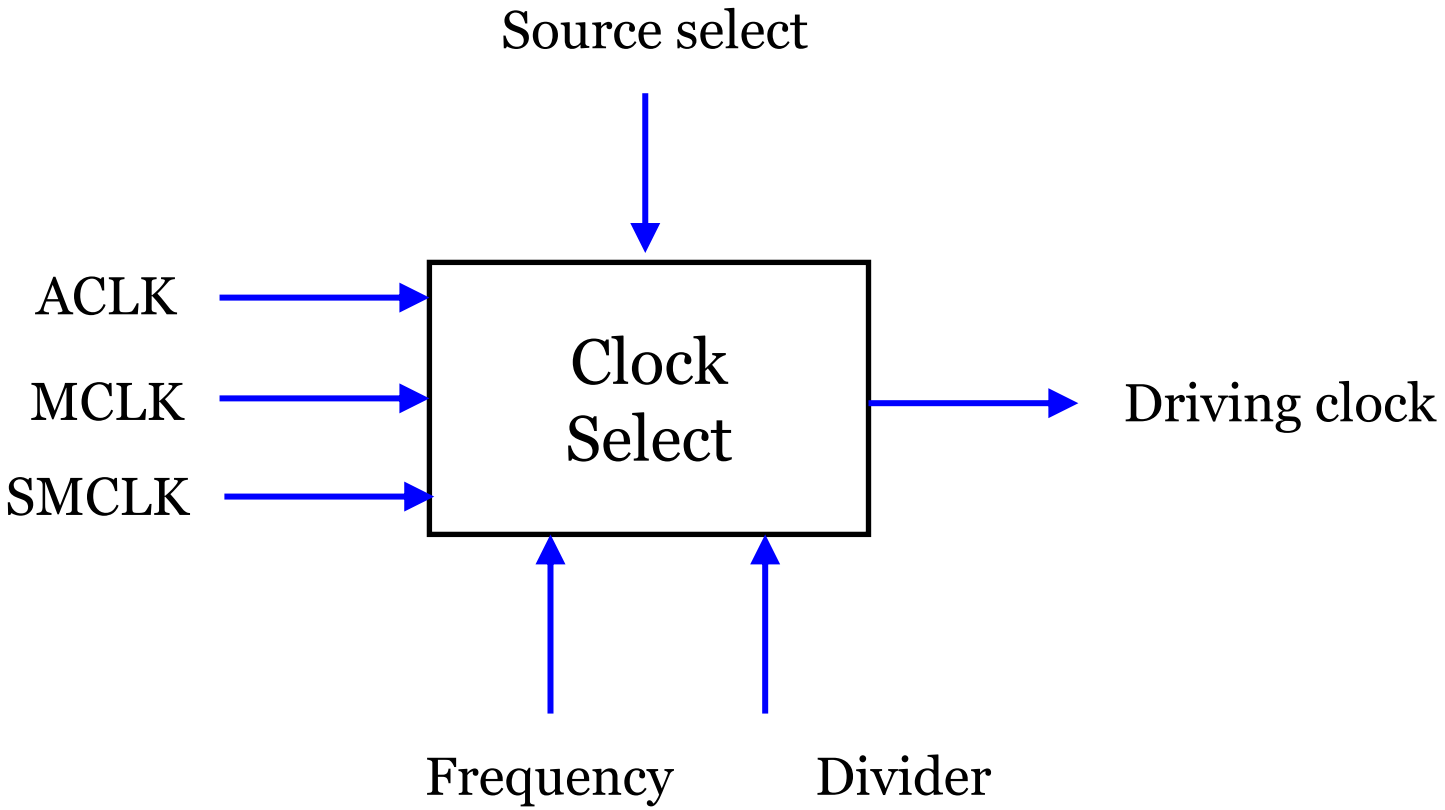
Configuration parameters

Clock Module and DCO Calibration

Digitally Controlled Oscillator (DCO)



Clock Module System



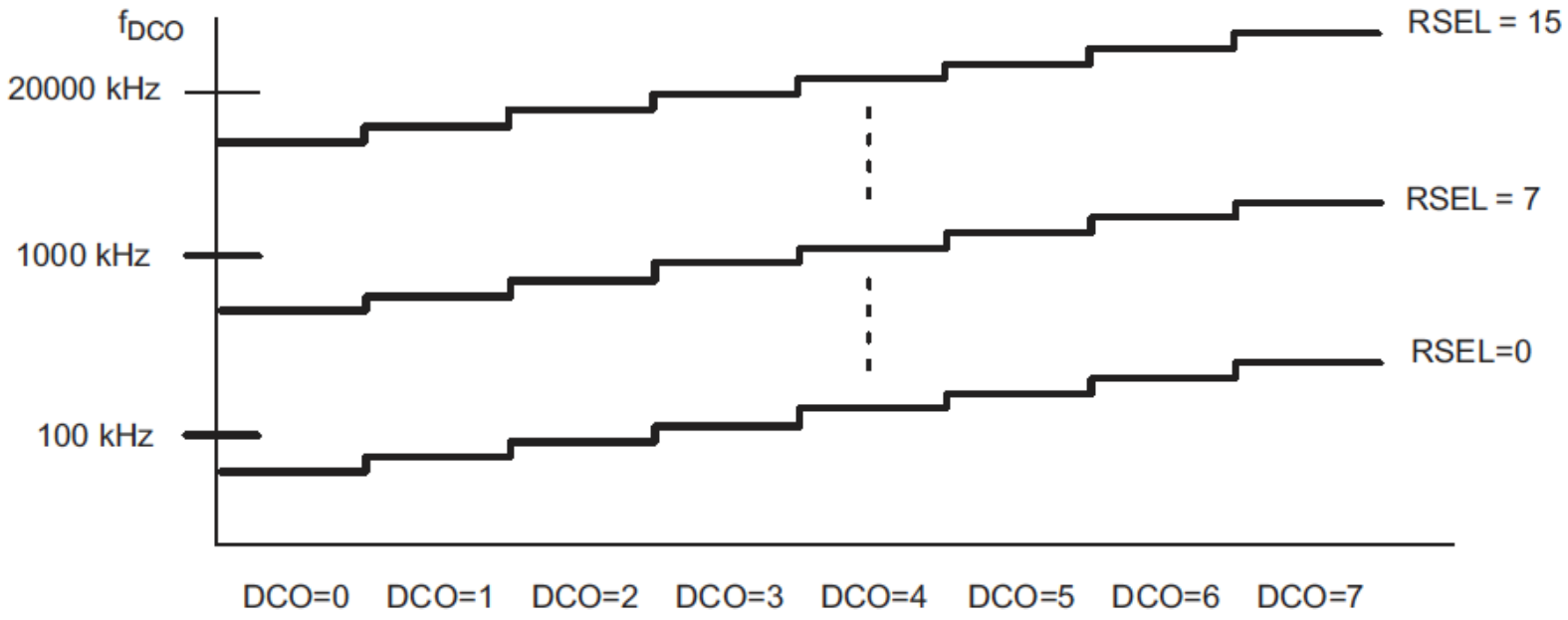
DCO Frequency Control

- 4 RSELx bits: Select one of sixteen nominal frequency ranges
- 3 DCOx bits: Divider
  - divide DCO range selected by RSELx bits into 8 frequency steps separated by approximately 10%
- 5 MODx bits: Modulation
  - Switch between freq selected by DCOx bits and next higher freq. set by DCOx+1

$$f = \frac{f_{DCO} \times f_{DCO+1}}{(\mathrm{MOD_x} \times f_{DCO}) + (32 - \mathrm{MOD_x}) \times f_{DCO+1}}$$

Digitally-Controlled Oscillator (DCO)

- After power-up clear (PUC), RSELx = 7 and DCOx = 3 and DCO starts at mid-range frequency



Clock System Control Registers

5.3.1 DCOCTL, DCO Control Register

7	6	5	4	3	2	1	0
DCOx			MODx				
rw-0	rw-1	rw-1	rw-0	rw-0	rw-0	rw-0	rw-0

DCOx	Bits 7-5	DCO frequency select. These bits select which of the eight discrete DCO frequencies within the range defined by the RSELx setting is selected.
MODx	Bits 4-0	Modulator selection. These bits define how often the f_{DCO+1} frequency is used within a period of 32 DCOCLK cycles. During the remaining clock cycles (32-MOD) the f_{DCO} frequency is used. Not useable when DCOx = 7.

Clock System Control Registers

5.3.2 BCSCTL1, Basic Clock System Control Register 1

7	6	5	4	3	2	1	0
XT2OFF	**XTS**(1)(2)	**DIVAx**		**RSELx**			
rw-(1)	rw-(0)	rw-(0)	rw-(0)	rw-0	rw-1	rw-1	rw-1

XT2OFF	Bit 7	XT2 off. This bit turns off the XT2 oscillator 0 XT2 is on 1 XT2 is off if it is not used for MCLK or SMCLK.
XTS	Bit 6	LFXT1 mode select. 0 Low-frequency mode 1 High-frequency mode
DIVAx	Bits 5-4	Divider for ACLK 00 /1 01 /2 10 /4 11 /8
RSELx	Bits 3-0	Range select. Sixteen different frequency ranges are available. The lowest frequency range is selected by setting RSELx = 0. RSEL3 is ignored when DCOR = 1.

DCO Calibration

- Information memory contains calibrated DCOCTL and BCSCTL1 register settings for specific frequencies
- To use calibrated settings, information is copied into DCOCTL and BCSCTL1 registers
- Calibration affects DCOx, MODx, and RSELx bits, and clear all other bits

Calibrated DCO Frequency Settings in C

```
// Copy the calibrated settings from flash to DCO control
registers

 if (CALBC1_1MHZ==0xFF) // If calibration constant erased
    {
     while(1);           // do not load, trap CPU!!
    }

    DCOCTL = 0;    // Select lowest DCOx and MODx settings
    BCSCTL1 = CALBC1_1MHZ;          // Set DCOCLK to 1MHz
    DCOCTL = CALDCO_1MHZ;
```

Memory Addresses of Calibration Data

- /* Calibration Data is saved in Information Memory with the following memory addresses */
- CALDCO_16MHZ = 0x10F8;
- CALBC1_16MHZ = 0x10F9;
- CALDCO_12MHZ = 0x10FA;
- CALBC1_12MHZ = 0x10FB;
- CALDCO_8MHZ = 0x10FC;
- CALBC1_8MHZ = 0x10FD;
- CALDCO_1MHZ = 0x10FE;
- CALBC1_1MHZ = 0x10FF;

Data defined in the link command file msp430g2553.cmd

Memory Addresses of Calibration Data

Table 8. Memory Organization

		MSP430G2153	MSP430G2253 MSP430G2213	MSP430G2353 MSP430G2313	MSP430G2453 MSP430G2413	MSP430G2553 MSP430G2513
Memory	Size	1kB	2kB	4kB	8kB	16kB
Main: interrupt vector	Flash	0xFFFF to 0xFFC0	0xFFFF to 0xFFC0	0xFFFF to 0xFFC0	0xFFFF to 0xFFC0	0xFFFF to 0xFFC0
Main: code memory	Flash	0xFFFF to 0xFC00	0xFFFF to 0xF800	0xFFFF to 0xF000	0xFFFF to 0xE000	0xFFFF to 0xC000
Information memory	Size	256 Byte	256 Byte	256 Byte	256 Byte	256 Byte
	Flash	010FFh to 01000h	010FFh to 01000h	010FFh to 01000h	010FFh to 01000h	010FFh to 01000h
RAM	Size	256 Byte	256 Byte	256 Byte	512 Byte	512 Byte
		0x02FF to 0x0200	0x02FF to 0x0200	0x02FF to 0x0200	0x03FF to 0x0200	0x03FF to 0x0200
Peripherals	16-bit	01FFh to 0100h	01FFh to 0100h	01FFh to 0100h	01FFh to 0100h	01FFh to 0100h
	8-bit	0FFh to 010h	0FFh to 010h	0FFh to 010h	0FFh to 010h	0FFh to 010h
	8-bit SFR	0Fh to 00h	0Fh to 00h	0Fh to 00h	0Fh to 00h	0Fh to 00h

defined in the link command file msp430g2553.cmd

CPU Active and Low-Power Modes

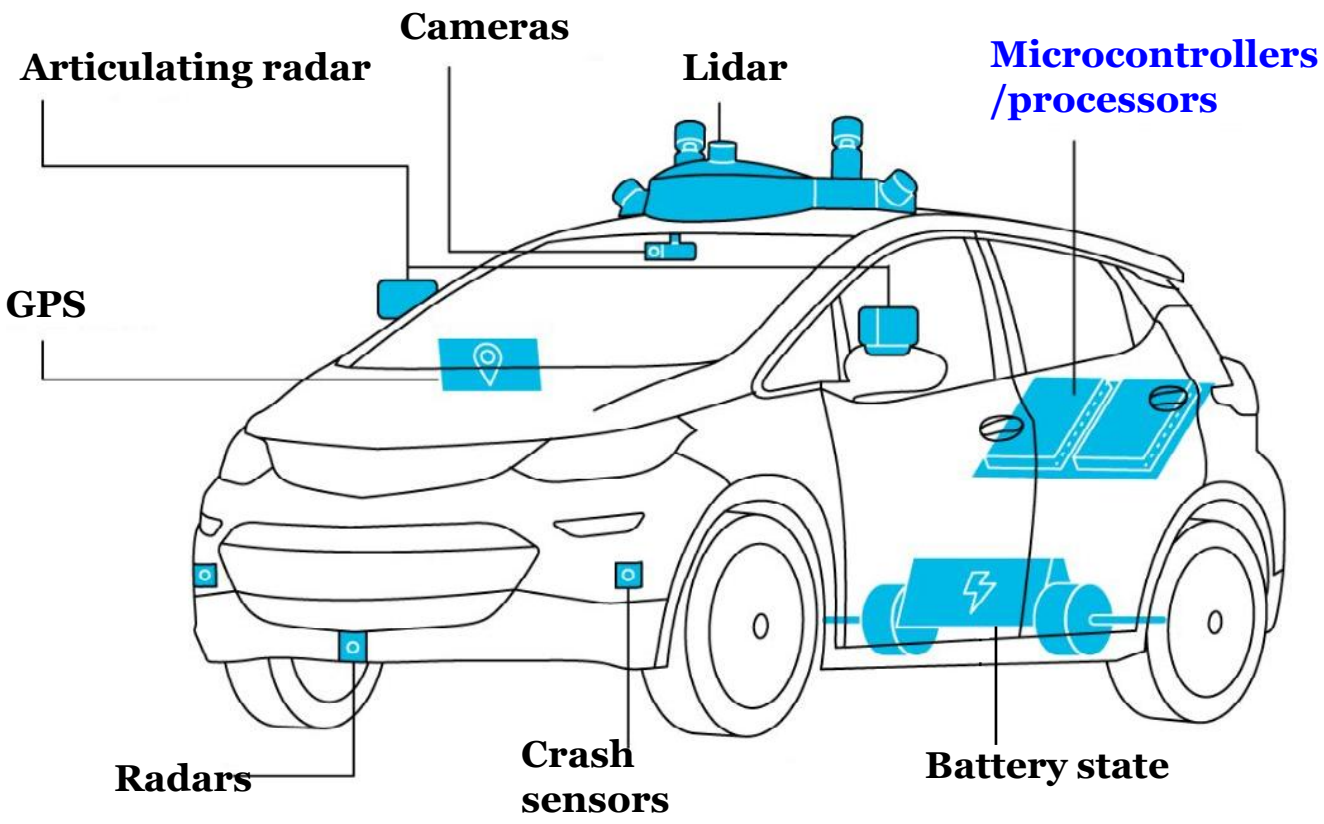
SCG1	SCG0	OSC OFF	CPU OFF	Mode	CPU and Clock Status
0	0	0	0	**Active**	CPU is active, all enabled clocks are active
0	0	0	1	**LPM0**	CPU, MCLK are disabled, SMCLK, ACLK are active
0	1	0	1	**LPM1**	CPU, MCLK are disabled. DCO and DC generator are disabled if the DCO is not used for SMCLK. ACLK is active
1	0	0	1	**LPM2**	CPU, MCLK, SMCLK, DCO are disabled. DC generator remains enabled. ACLK is active.
1	1	0	1	**LPM3**	CPU, MCLK, SMCLK, DCO are disabled. DC generator disabled. ACLK is active.
1	1	1	1	**LPM4**	CPU and all clocks disabled

CPU Active and Low-Power Modes

- Main clock (MCLK): system clock used by CPU
- Sub-Main clock (SMCLK): sub-system clock used by peripheral modules.
- Auxiliary clock (ACLK): sourced either from a 32768-Hz crystal or internal LF oscillator
- DCO: Internal digitally controlled oscillator

Serial Communication

Microprocessors and Sensors



How do these devices communicate with each other?

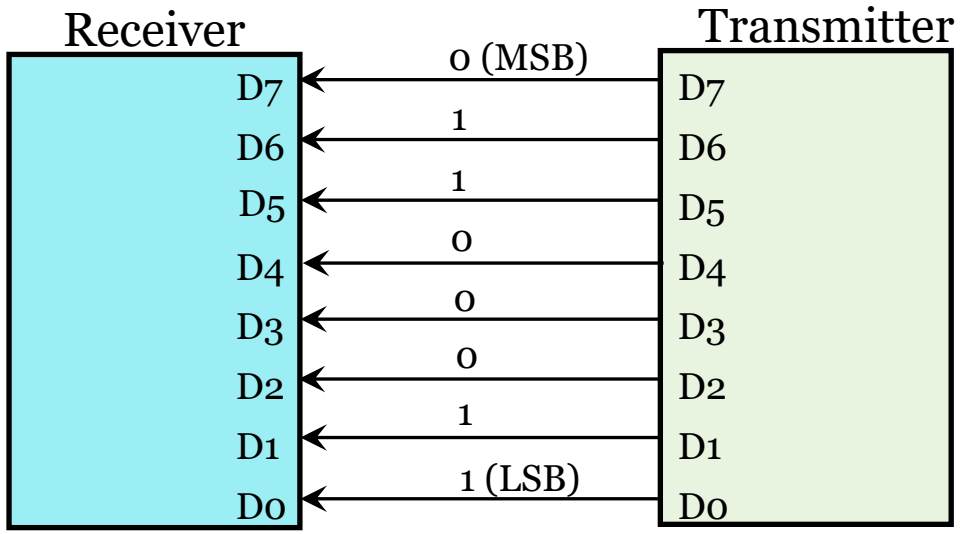
- Microcontrollers
  - acquire data from different sensors
  - process data
  - send instructions to actuators
- Sensors
  - different types
    - Cameras: image
    - Radar: distance
    - GPS: location
  - different data rates
    - Cameras: 30 fps
    - Accelerometer: 1-3kHz
    - GPS: 100-200Hz
  - different format
    - Analog: temperature
    - Digital: video/image

Key Ideas to Take-home

- Parallel vs serial communication
- Serial communication
  - Simplex, half-duplex, full-duplex
  - Synchronous vs asynchronous
- Serial communication interfaces
  - Universal Asynchronous Receiver/Transmitter (UART)
  - Serial Peripheral Interface (SPI)
  - Inter-Integrated Circuit (I2C)
- How do chose an appropriate communication method?

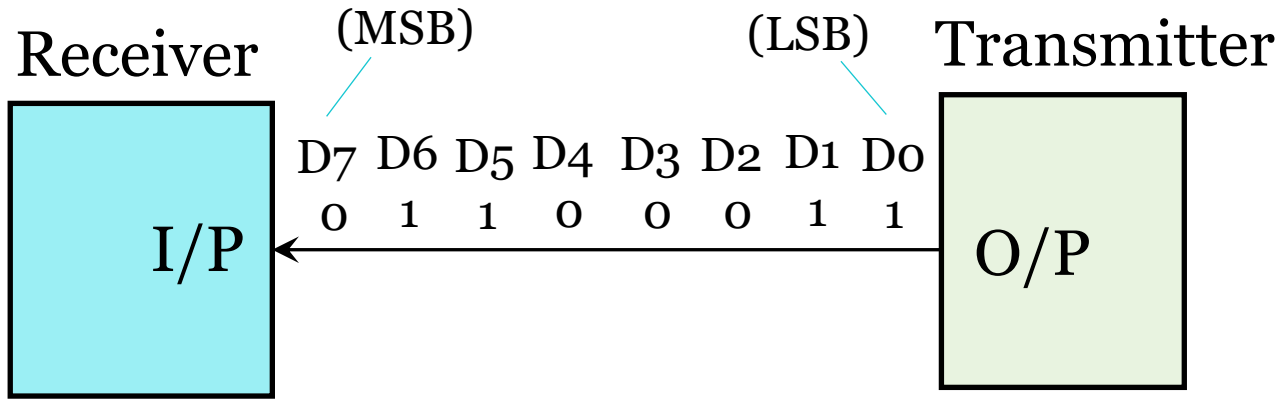
Different Forms of Communication

- Parallel communication: simultaneous transfer via multiple wires
  - all bits of a word transferred in one cycle
- Speed is measured in bits/second (bps)



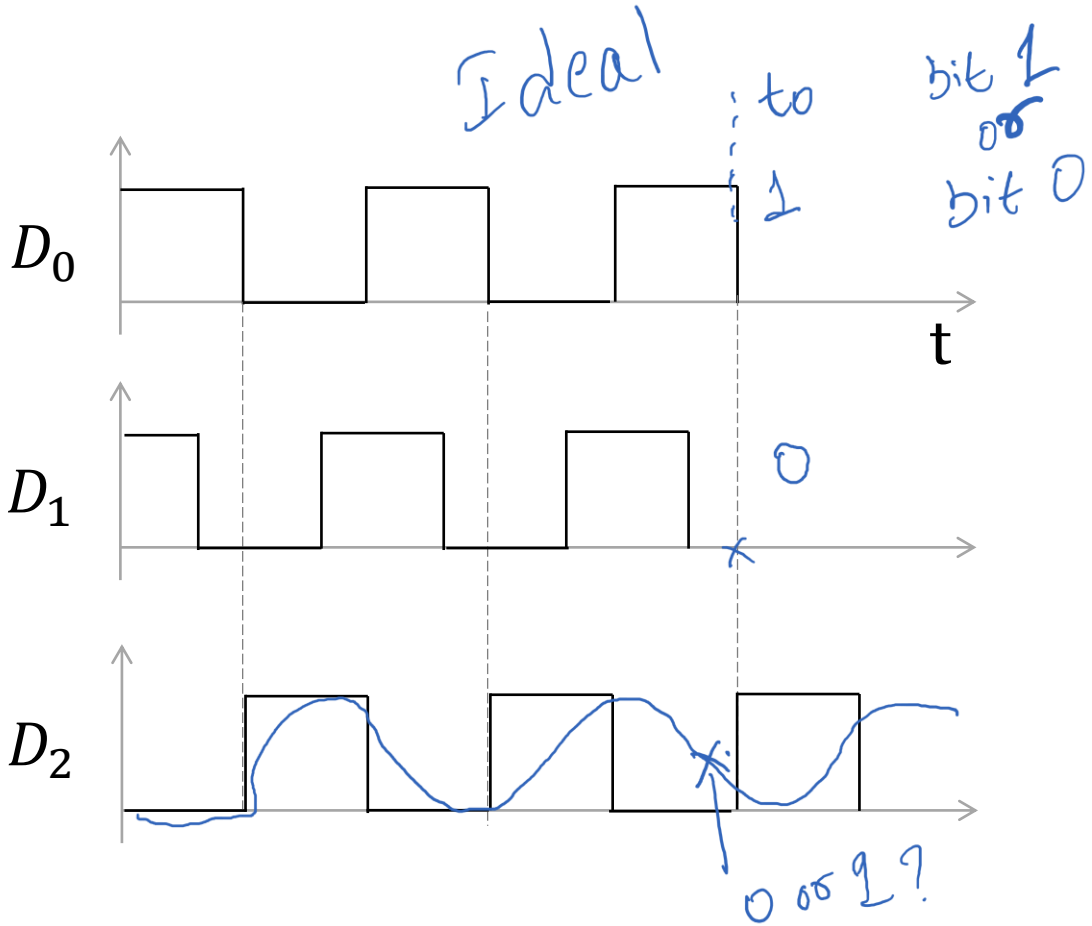
Different Forms of Communication

- Serial communication: sequential data transfer on a single wire
  - requires parallel to serial conversion
- Speed is measured in bits/second (bps)
- In principle parallel communications faster than serial



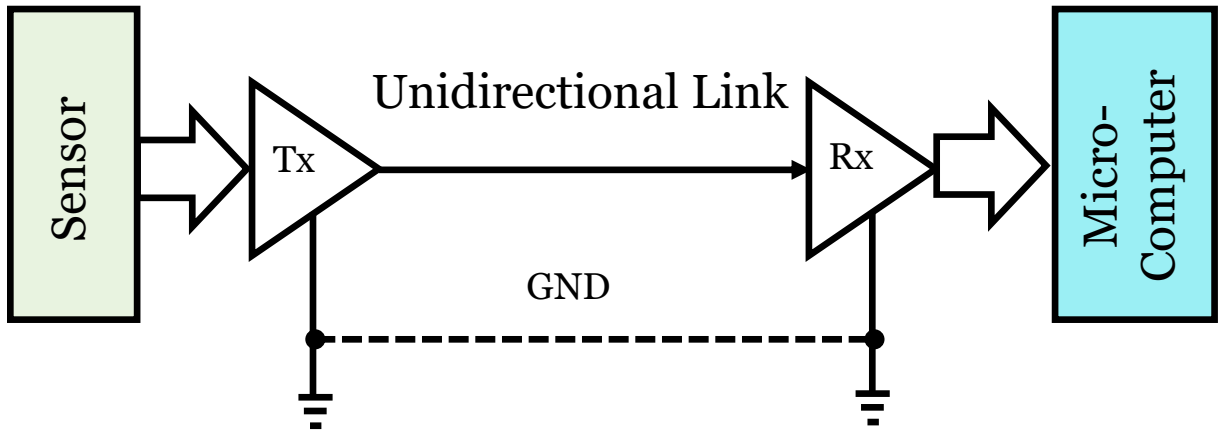
Serial vs Parallel Communication

- Harmful effects when communication speed or distance increase
  - communication delay
  - cross-talk
  - signal attenuation and noise
- Parallel communication more prone to harmful effects
  - limited maximum speeds
  - require more number of pins on IC



Types of Serial Communication

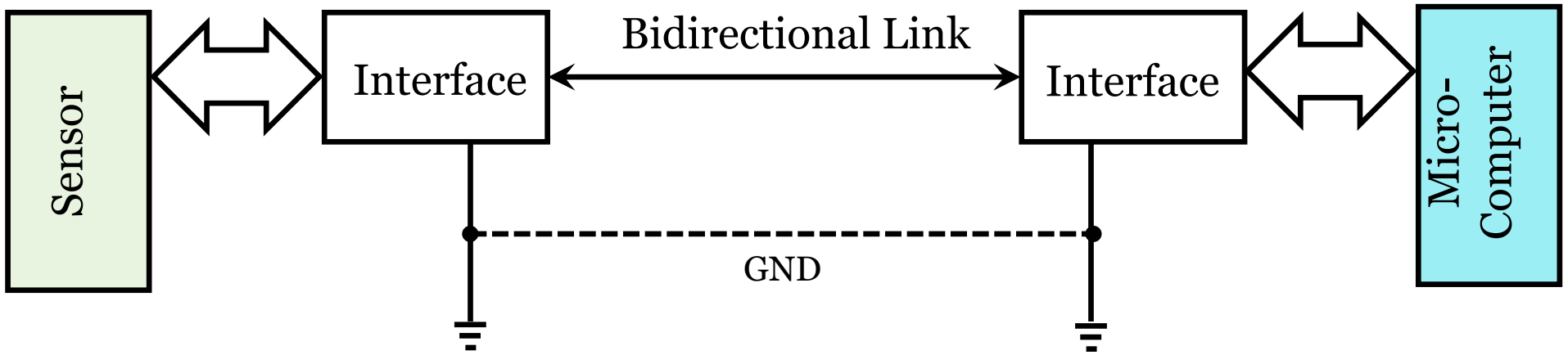
- Based on type of connectivity
  - Simplex
  - Half-duplex
  - Full-duplex



- Simplex communication
  - transmit data in only one direction
  - no means of acknowledging or verifying correct reception
  - Example: temperature sensor communicating with microcomputer

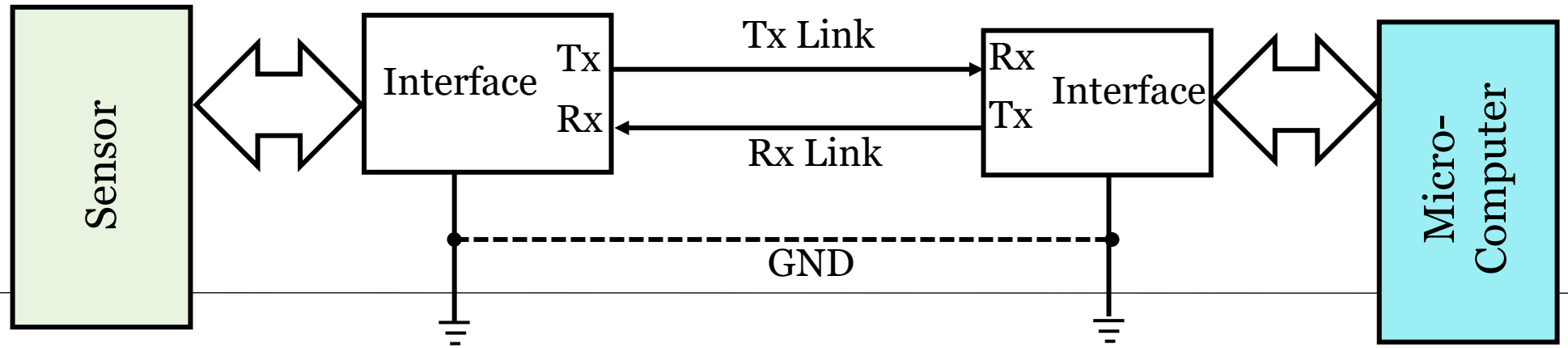
Half Duplex Communication

- Single link that allows communication in both direction
  - *but* only in one direction at a time
- Both ends of link have interfaces that work as either transmitter or receiver



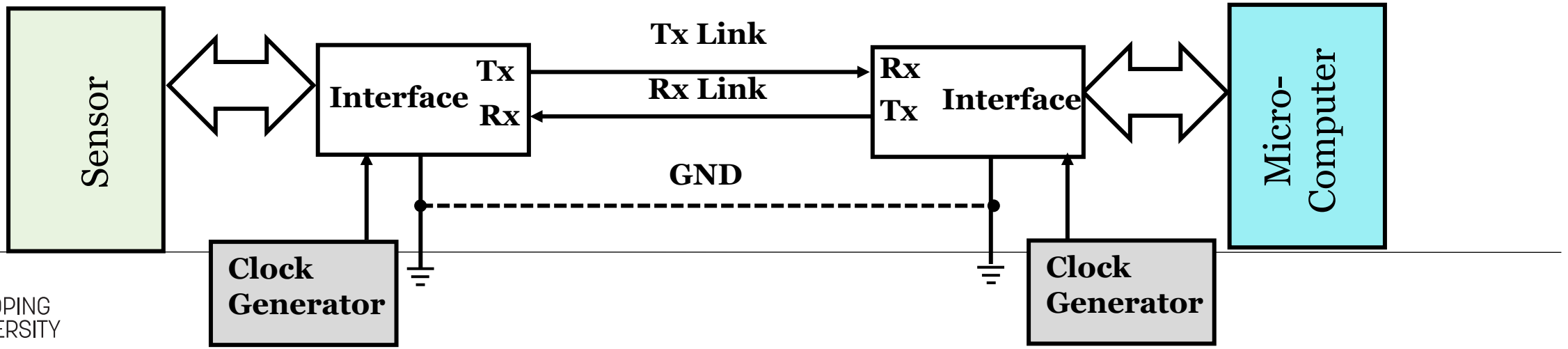
Full Duplex Communication

- Two separate links for simultaneous communication in both directions
  - one dedicated to transmit and another to receive
- Interfaces at each end of link can handle both links simultaneously
  - requires no switching
  - enables uninterrupted bidirectional communication



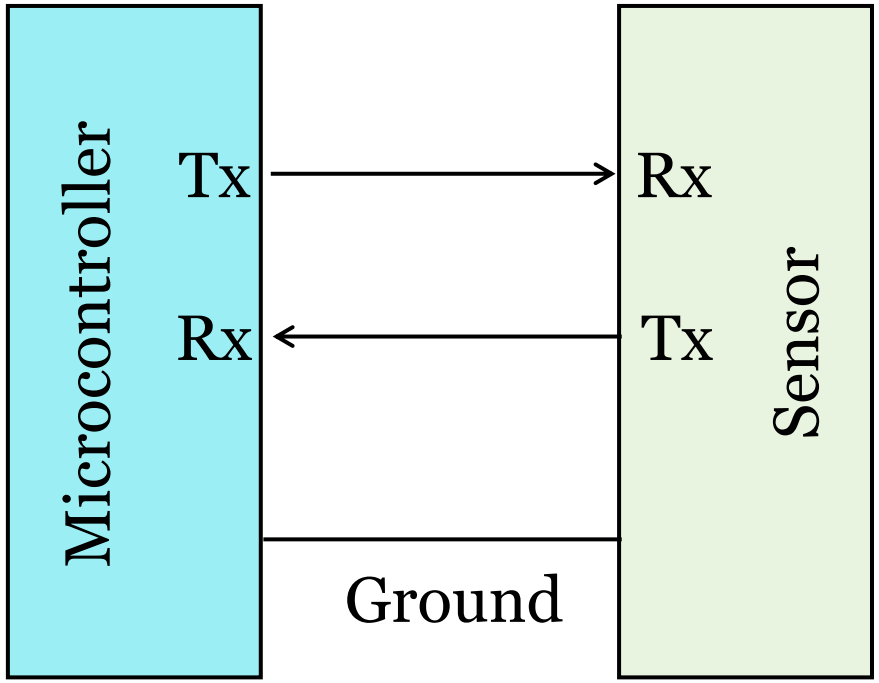
Clock Synchronization in Serial Channels

- Serial communication require stable clock signal
  - to establish transmission and reception rate
  - to synchronize operation of interfaces
- Asynchronous communication use independent clock generators at both end
- Synchronous communication transmits clock along with data



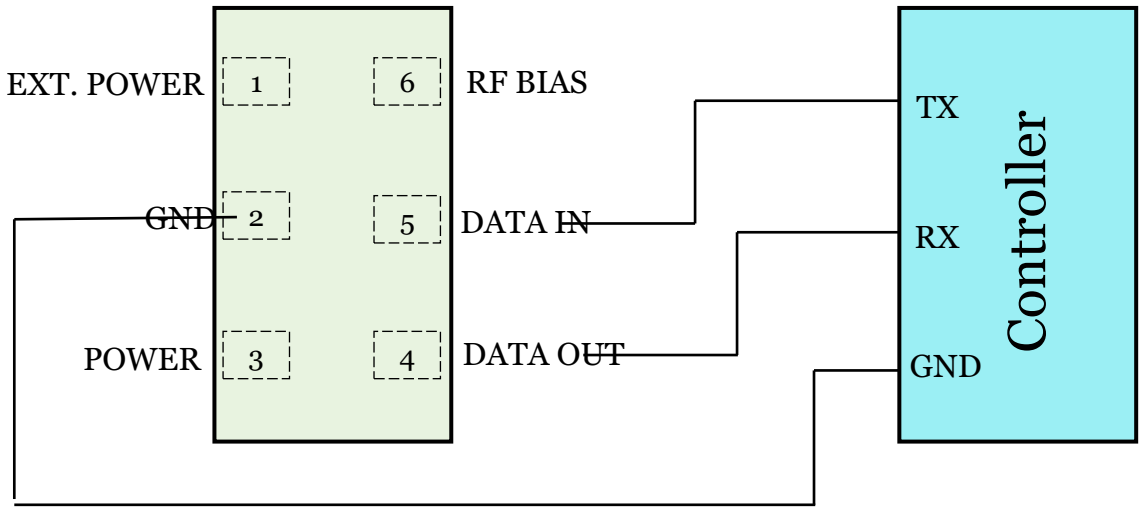
Universal Asynchronous Receiver/ Transmitter (UART)

- Serial interface
  - send and receive one bit at a time
  - does not transmit timing information
  - transmission speed and voltage levels for 1 or a 0 bit are pre-agreed
- Uses three pins to communicate
  - transmit, receive, ground



Example for UART: GPS Modules

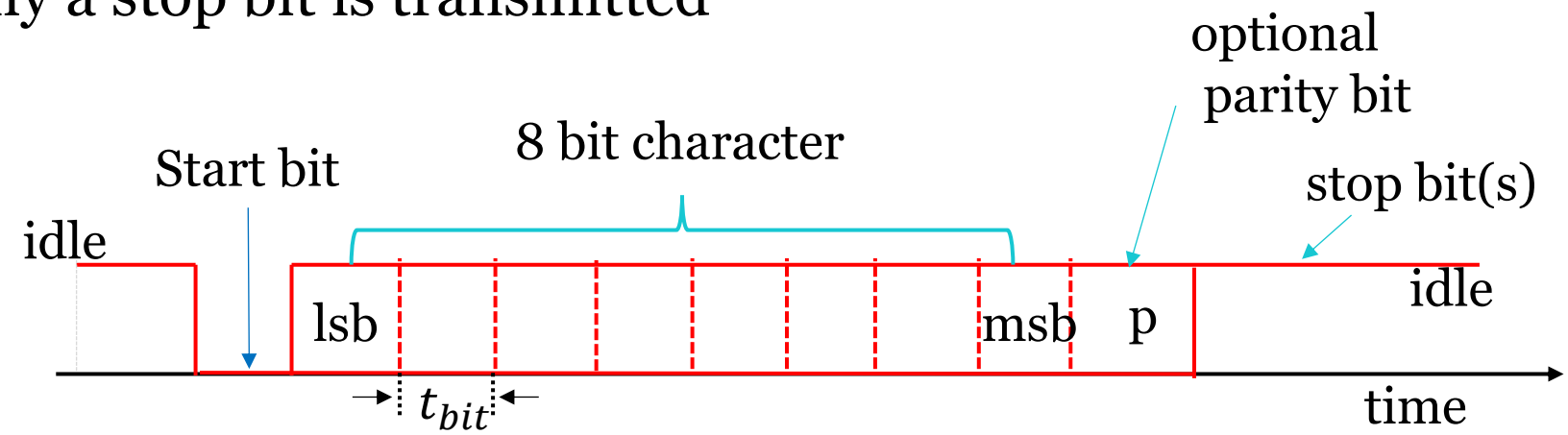
- Controller to sensor communication: Sensor Initialization Information (PGMRI) message
  - $PGMRI<1><2><3><4><5><6><7>*hh<CR><LF>
- Sensor to controller communication: Global Positioning System Fix Data (GGA) message
  - $GPGGA<1><2><3><4><5><6><7><8><9><10><11><12*hh<CR><LF>



Garmin G15X
GPS module

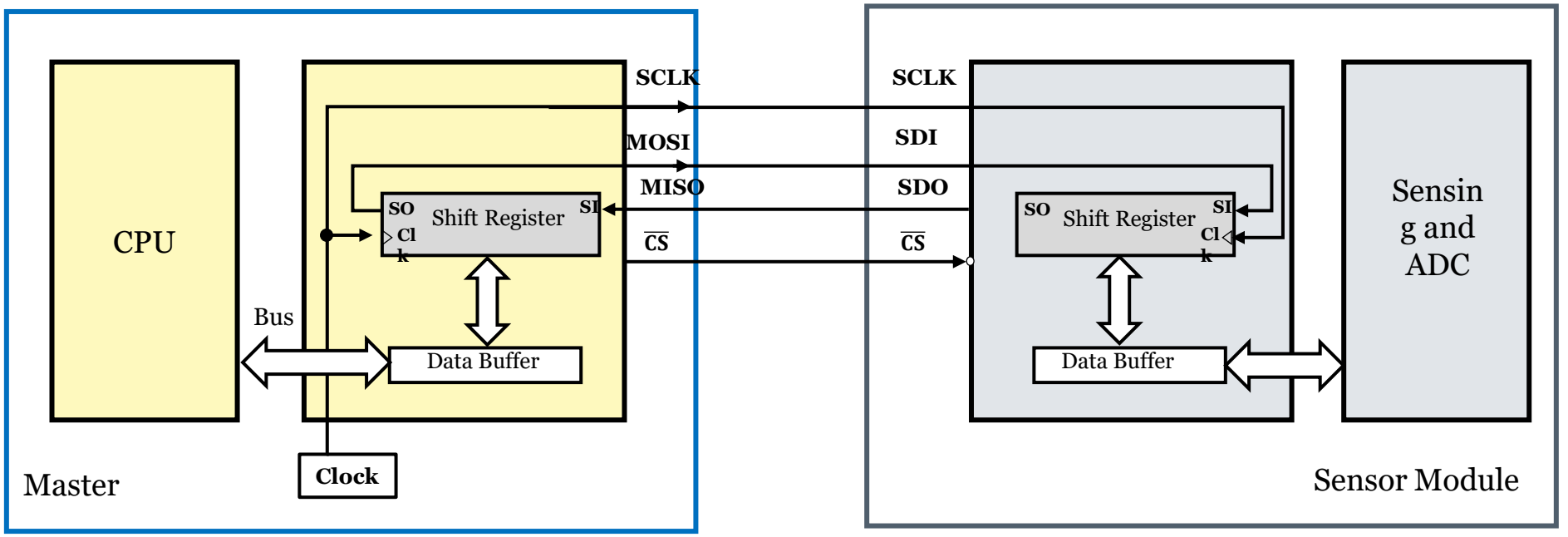
Universal Asynchronous Receiver/ Transmitter (UART)

- Sequence of operation
  - send start bit to indicate beginning of transmission
  - all data bits is transmitted
  - finally a stop bit is transmitted



Serial Peripheral Interface (SPI)

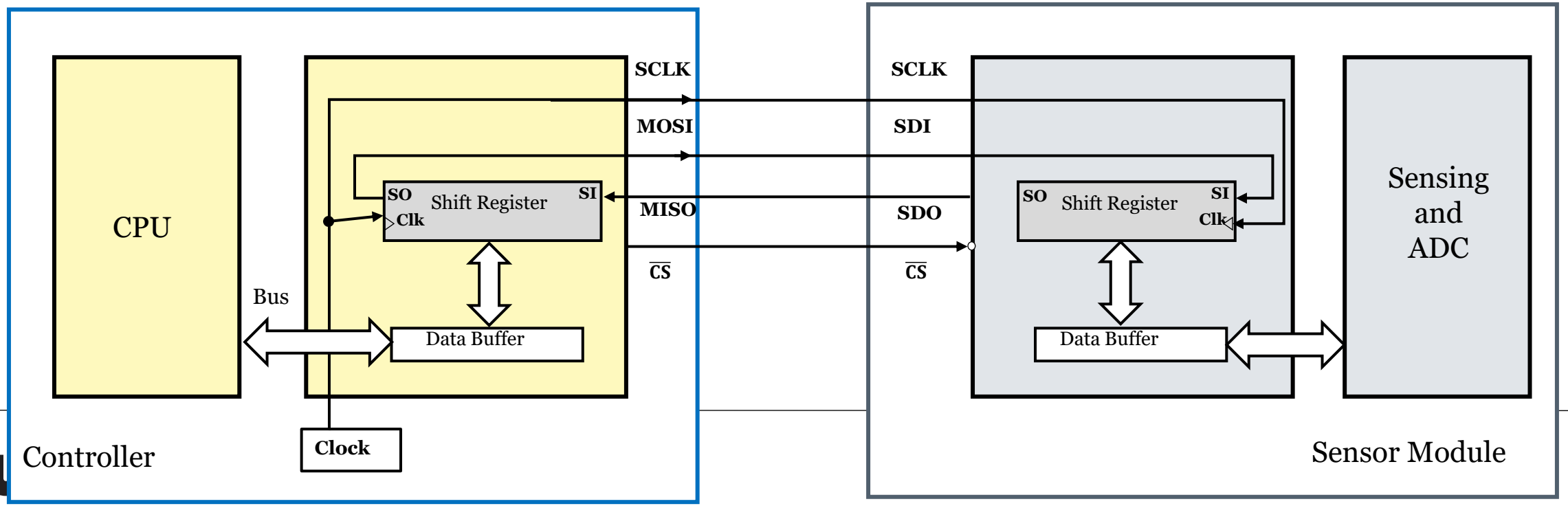
- Synchronous serial bus with full-duplex capability
  - support communications between microprocessor and multiple sensors



MOSI: Master Out - Slave In, MISO: Master In - Slave Out

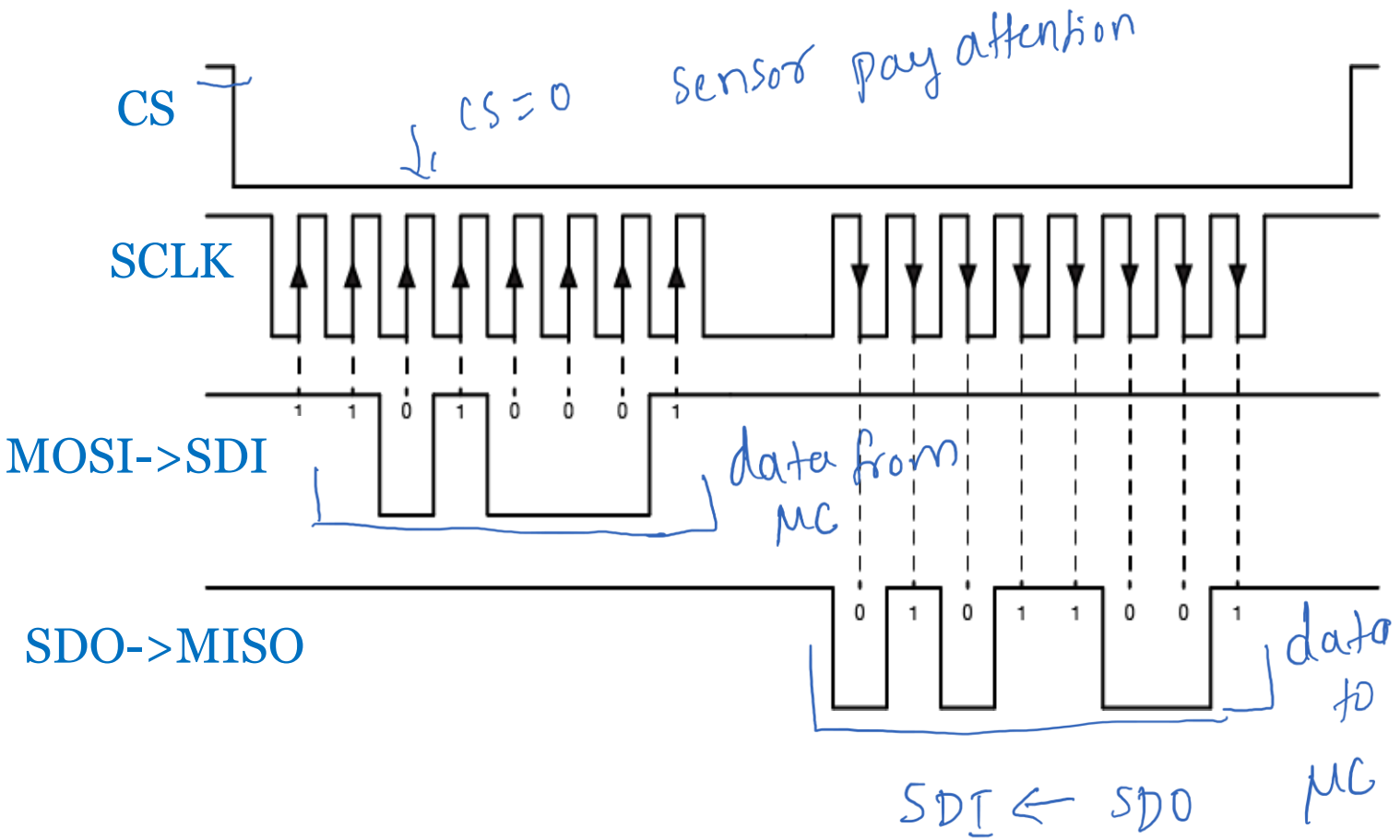
Features of SPI Controller

- Single shift register that serves as both receiver and transmitter
- Tx and Rx interfaces synchronized via clock signal
- Both interfaces can simultaneously transmit and receive



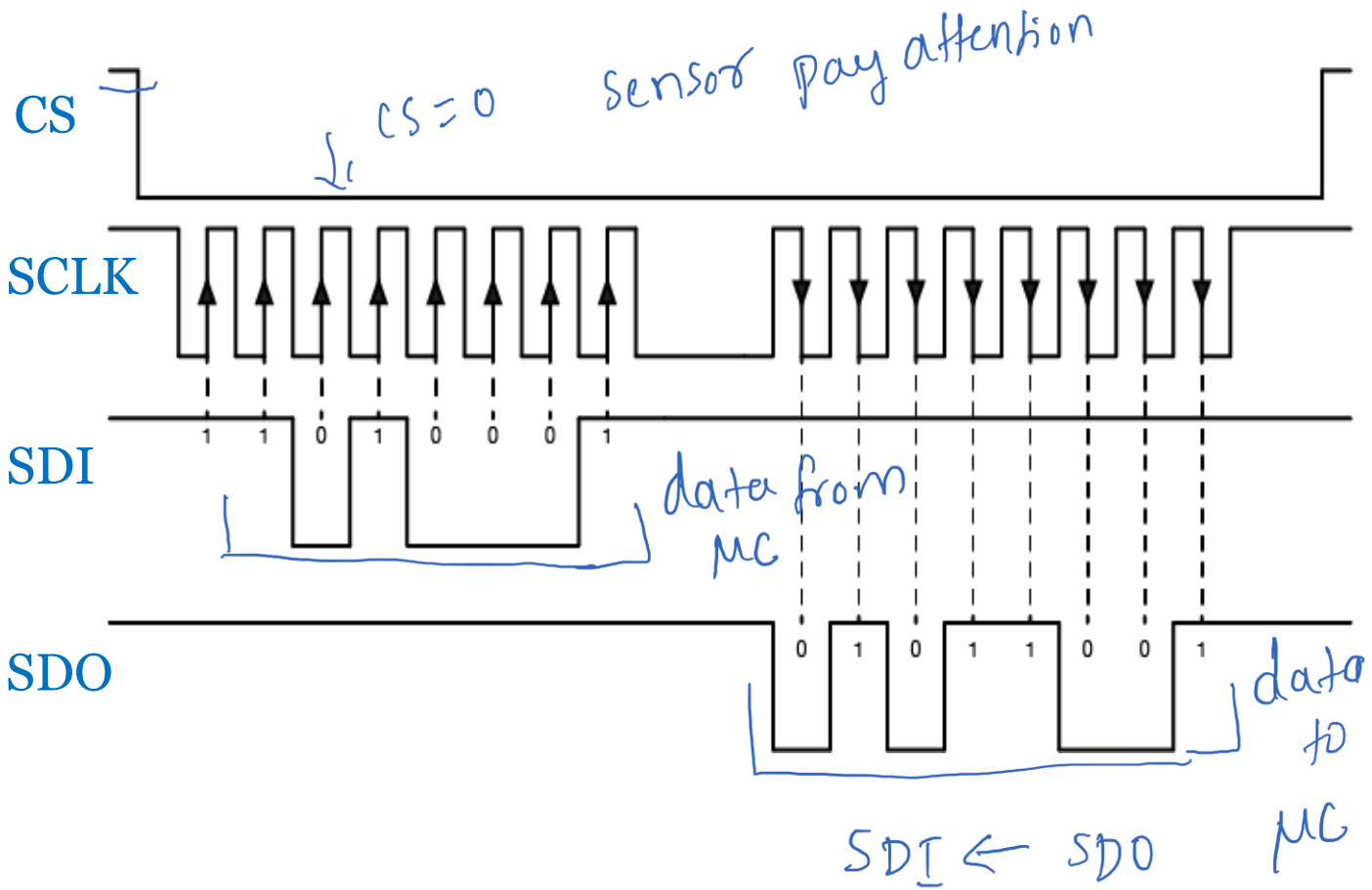
Serial Peripheral Interface Protocol

- SPI signals
  - Chip Select (CS): signal to enable device
  - Serial clock (SCLK): sent by controller
  - Serial Data-In (SDI): input data into device
  - Serial data-out (SDO): serial output data from device



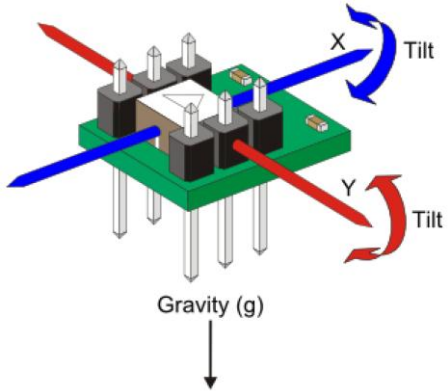
SPI Communication protocol

- Controller sets CS pin low
- When CS is high, sensor will not respond to any commands
- Sensor listens for data on its SDO line at rising edge of clock
- Sensor sends data on its SDI to controller at falling edge of clock

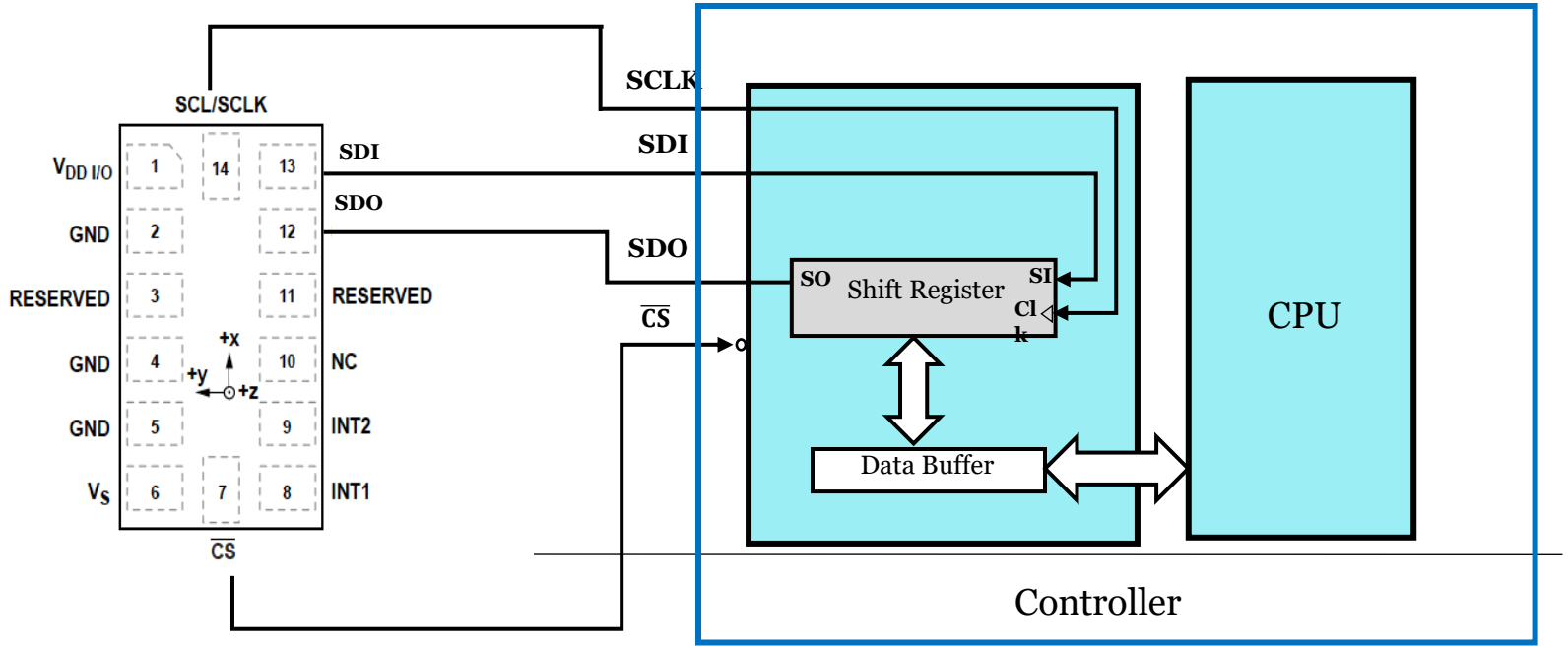


Example for SPI: Accelerometer Sensor

- Sensor to controller data
  - Acceleration along 3 axis
  - 10 to 13 bits data for each axis
- Controller to sensor data: sensor parameters
  - Output data rate: up to 3.2kHz
  - Resolution: 10 to 13 bits
  - Measurement range and sensitivity

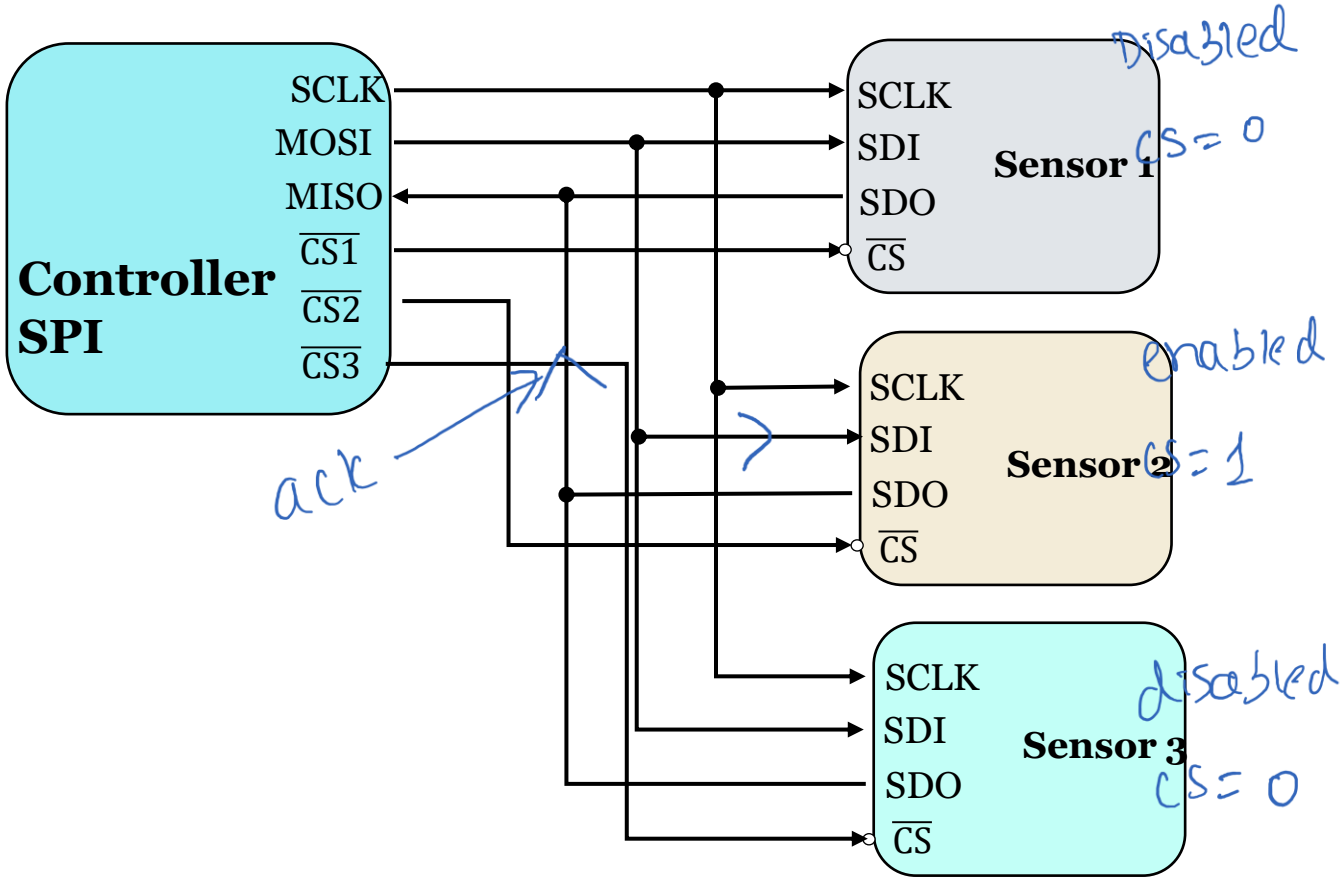


Analog Device Accelerometer ADXL345-EP



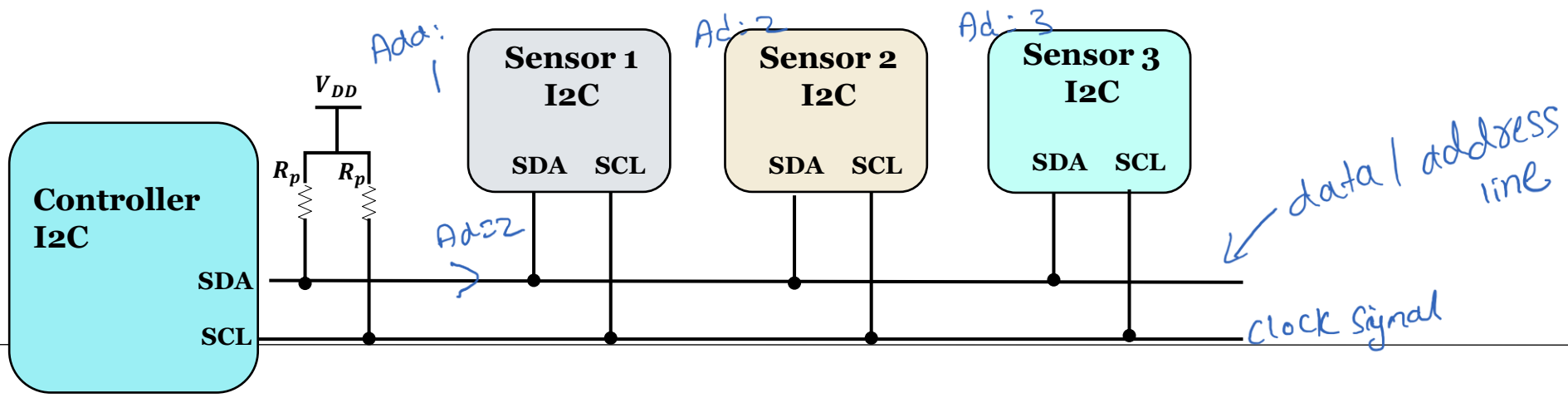
SPI for Multiple Peripherals

- Controller sets CS line of the device it wants to communication with
- Sensor listens for data on its SDO line at rising edge of clock
- Sensor sends data on its SDI to controller at falling edge of clock
- When sensor's CS pin is high, it will not respond to any commands
- Number of pins needed increase with sensors



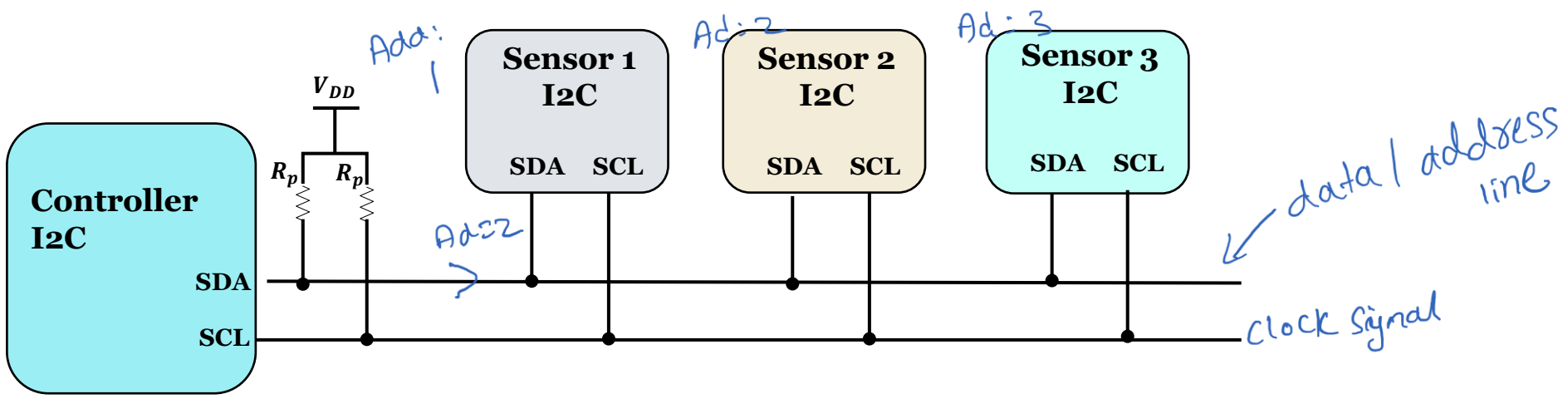
Inter-Integrated Circuit (I2C)

- Uses two lines SDA (Serial Data) and SCL (Serial Clock) for half-duplex communication
- Can handle multiple controllers and sensors
  - SCL is used to send clock signal and synchronize all bus transfers
  - SDA exchanges data in both directions



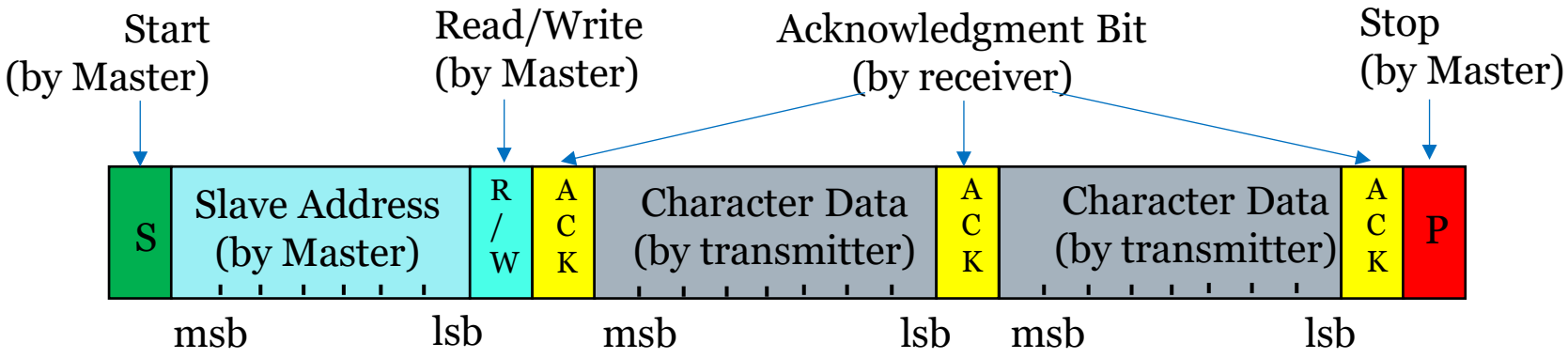
Inter-Integrated Circuit (I2C)

- Devices in I2C bus are software addressable
  - address field length: 7 or 10 bits
- Collision detection and arbitration mechanism for multi-controller mode



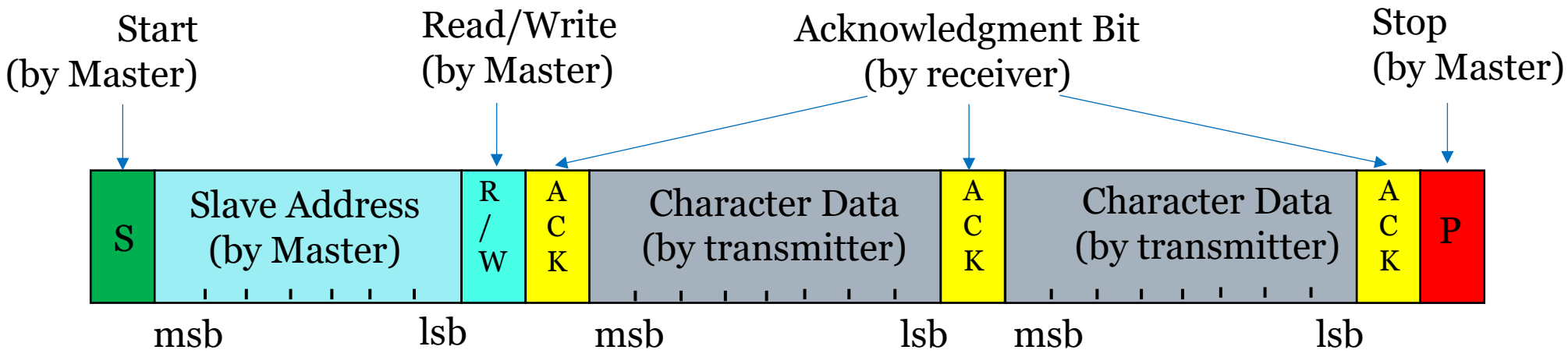
I2C Communication Protocol

- Three modes: Start (S), Transfer, Stop (P)
- Start: controller takes control of bus to initiate a transfer
  - controller listens to SDA to determine channel status
  - idle status if both SCL and SDA high for at least one clock period
  - If channel available, controller lowers SDA line to set start condition and indicate channel busy



I2C Communication Protocol

- Transfer:
  - all sensors in the bus are in listen mode for incoming data.
  - a start condition is sent in the middle of active message to change transfer direction without releasing the bus.



MSP430 Universal Serial Communication Interface (USCI)

MSP430 Universal Serial Communication Interface (USCI)

- Two USCI modules support multiple serial communication modes
- USCI_A modules support:
  - UART mode
  - Pulse shaping for infrared (IrDA) communications
  - Automatic baud rate detection for LIN communications
  - SPI mode
- USCI_B modules support:
  - I2C mode
  - SPI mode

MSP430 UART Introduction

- In asynchronous mode, USCI_A connects MSP430 to external system
  - via two external pins UCAxRXD and UCAxTXD.
- UART features:
  - 7- or 8-bit data with odd, even, or non-parity
  - Independent transmit and receive shift registers
  - Separate transmit and receive buffer registers
  - LSB-first or MSB-first data transmit and receive

MSP430 UART Introduction

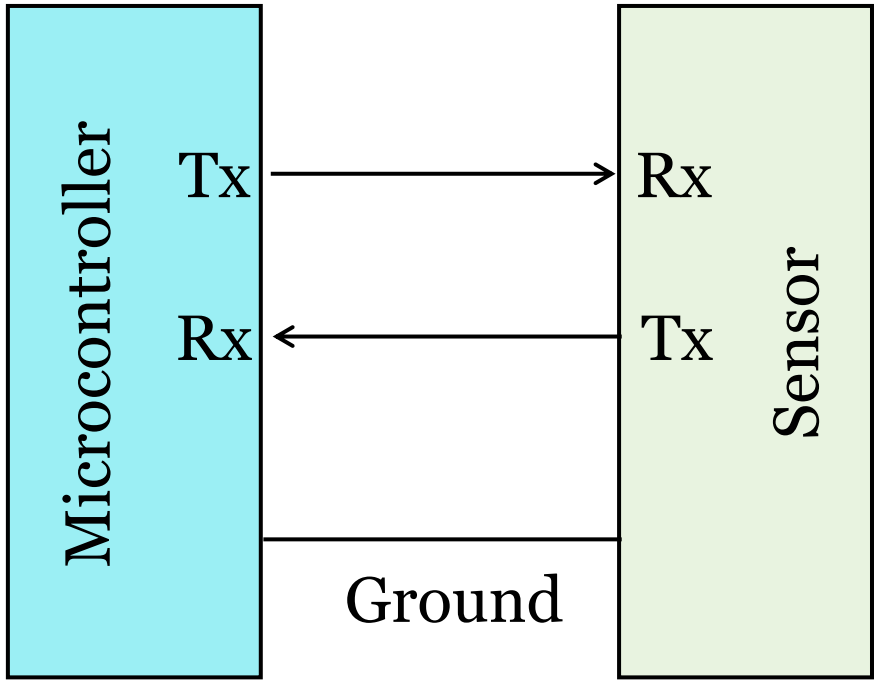
- In asynchronous mode, USCI_A connects MSP430 to external system via two external pins, UCAxRXD and UCAxTXD.
- UART features:
  - Built-in idle-line and address-bit communication protocols for multiprocessor systems
  - Programmable baud rate with modulation for fractional baud rate support

MSP430 UART Introduction

- In asynchronous mode, USCI_A connects MSP430 to external system via two external pins, UCAxRXD and UCAxTXD.
- UART status features:
  - Receiver start-edge detection for auto-wake up from LPMx modes
  - Status flags for error detection and suppression
  - Status flags for address detection
  - Independent interrupt capability for receive and transmit

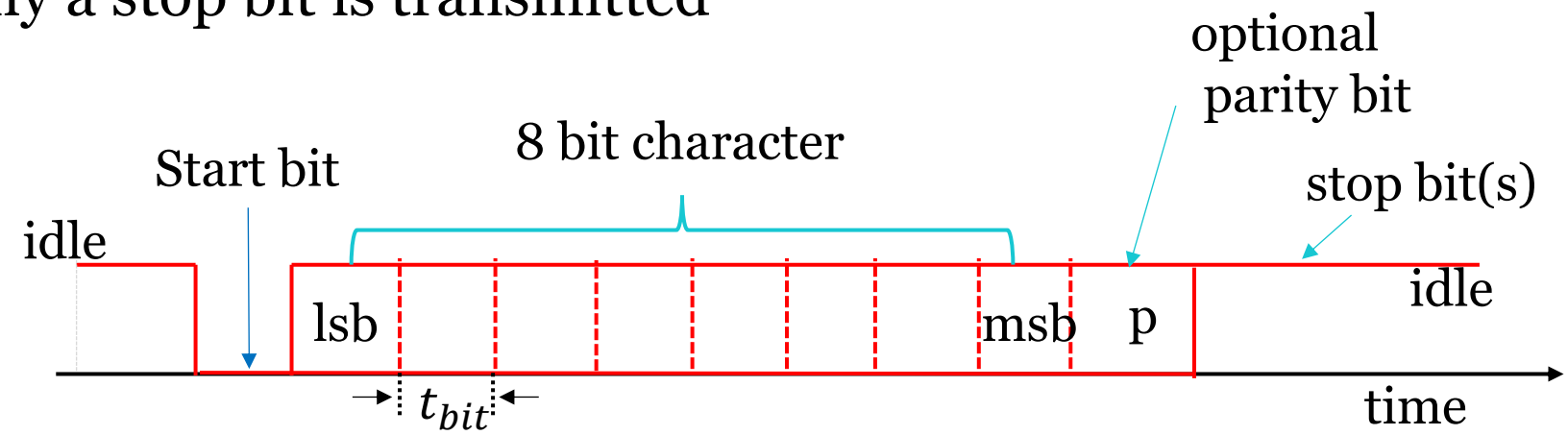
Universal Asynchronous Receiver/ Transmitter (UART)

- Serial interface
  - send and receive one bit at a time
  - does not transmit timing information
  - transmission speed and voltage levels for 1 or a 0 bit are pre-agreed
- Uses three pins to communicate
  - transmit, receive, ground



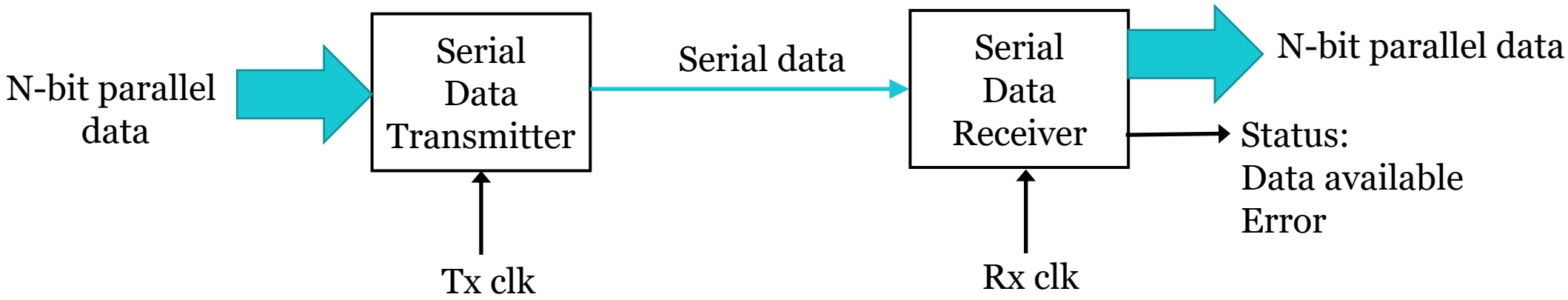
Universal Asynchronous Receiver/ Transmitter (UART)

- Sequence of operation
  - send start bit to indicate beginning of transmission
  - all data bits is transmitted
  - finally a stop bit is transmitted



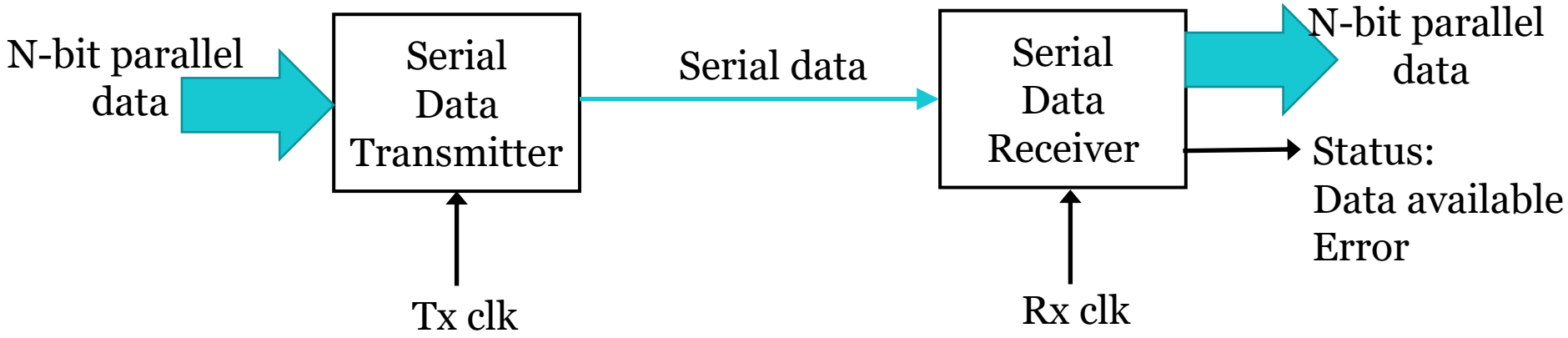
UART – Universal Asynchronous Receiver /Transmitter

- UART takes bytes of data and transmits individual bits in a sequential fashion.
- At the destination, receiver re-assembles the bits into complete bytes.
- Each UART contains a shift register for conversion between serial and

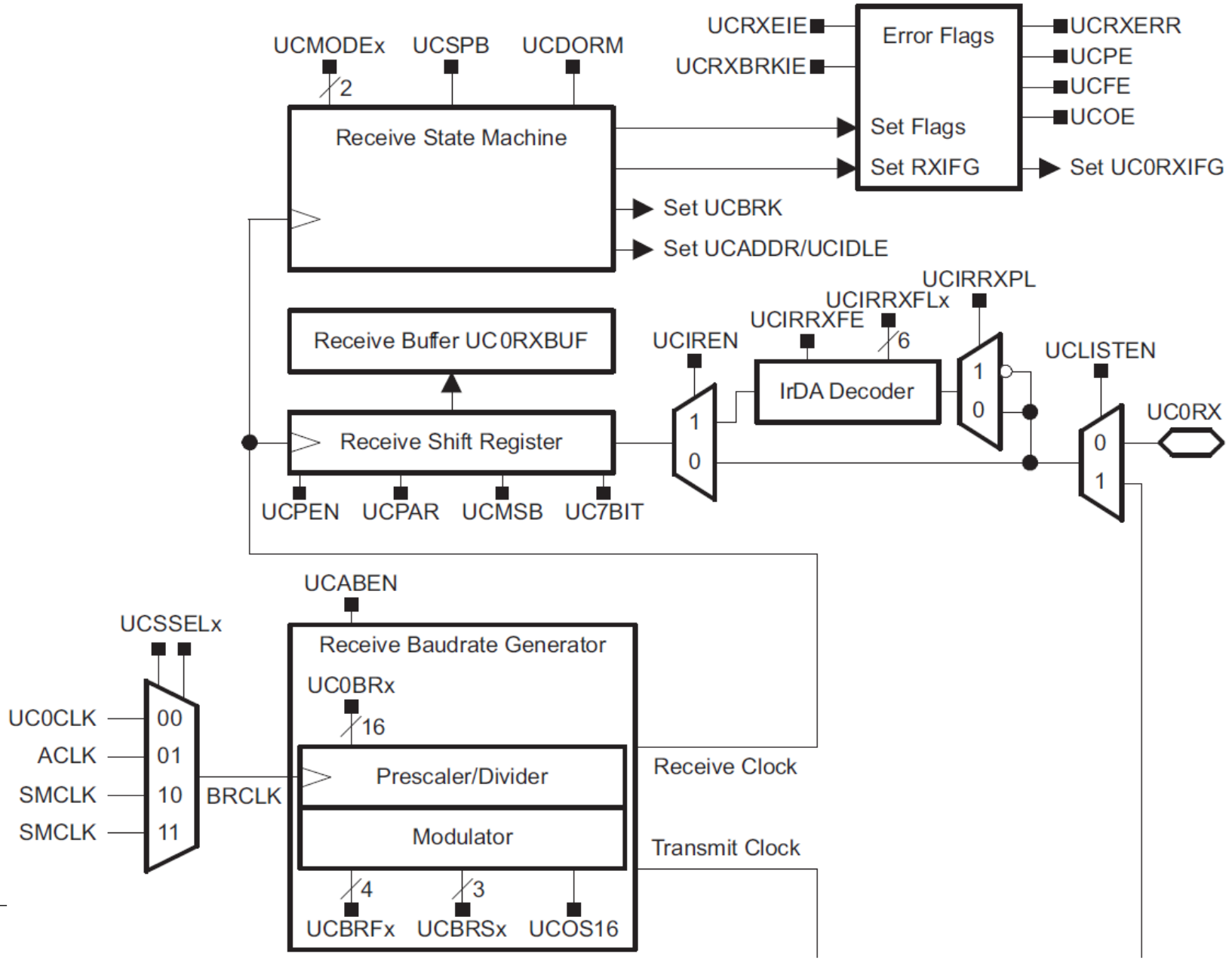


UART – Universal Asynchronous Receiver /Transmitter

- Baud rate: Symbol rate i.e the number of distinct symbol changes (signaling events) per second in a digitally modulated signal.



USCI Receiver Block



USCI Transmitter Block

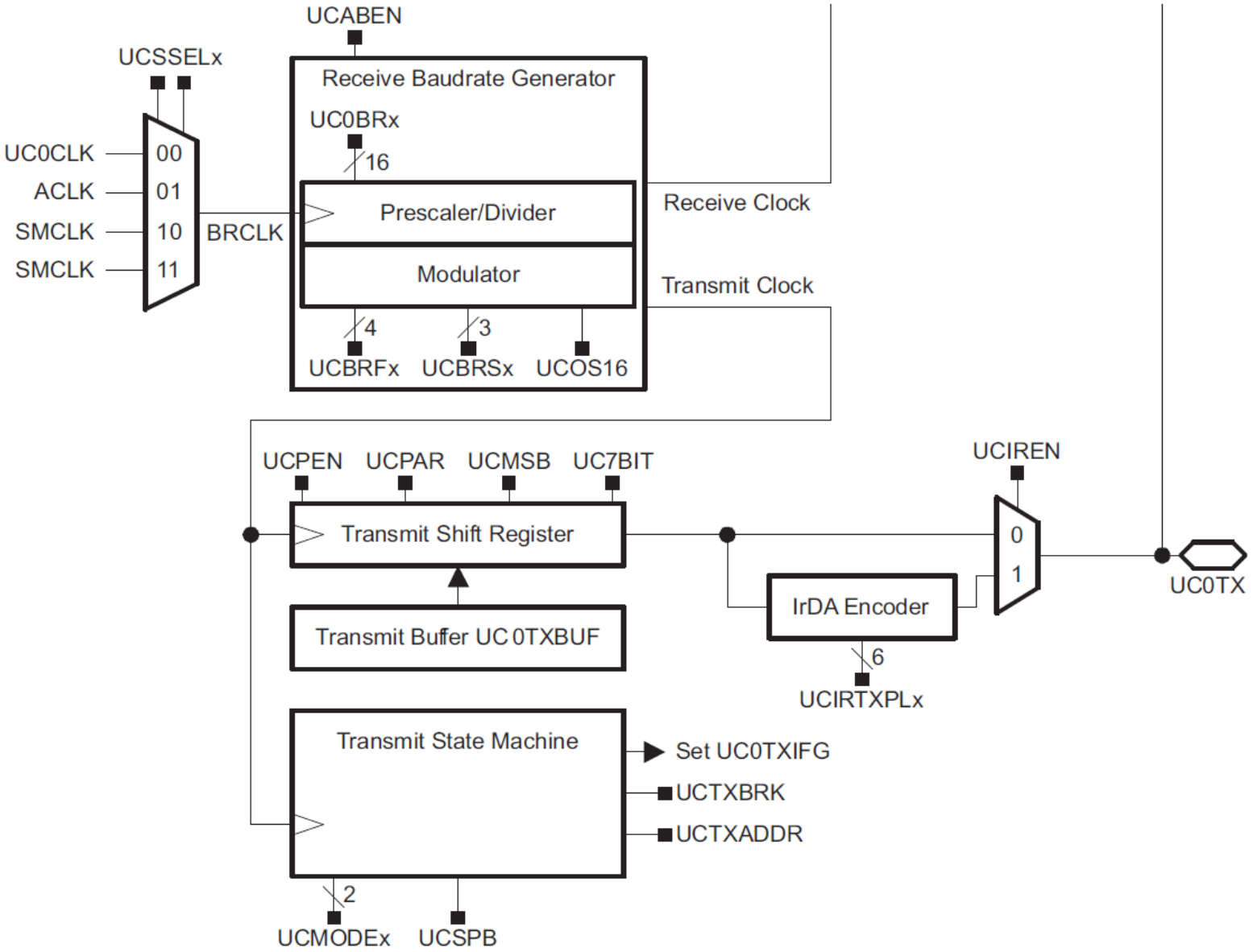


Figure 15-1. USCI_Ax Block Diagram: UART Mode (UCSYNC = 0)

Pins for UCA0RXD and UCA0TXD (Page 43, msp430g2553.pdf)

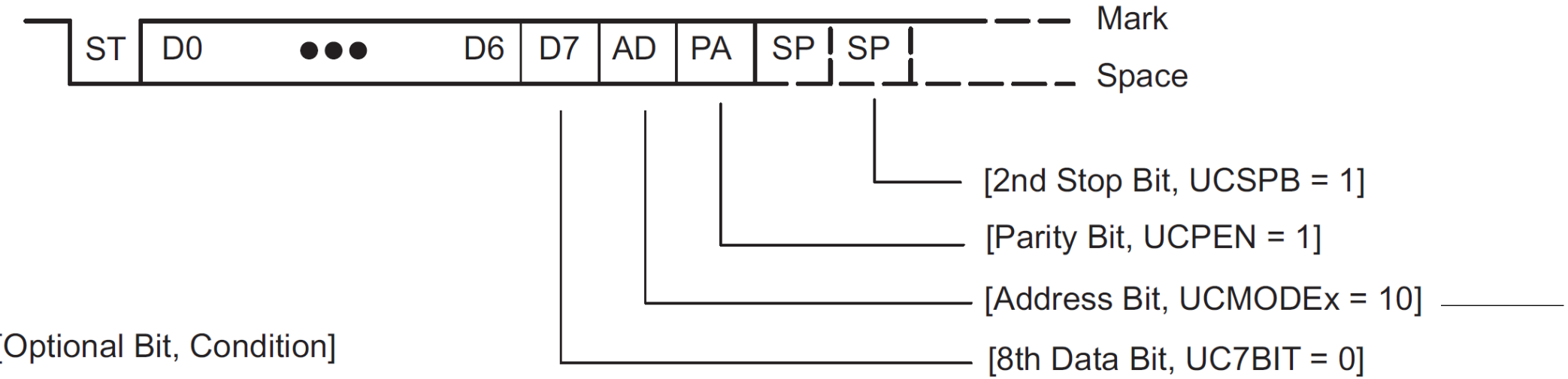
Table 16. Port P1 (P1.0 to P1.2) Pin Functions

PIN NAME (P1.x)	x	FUNCTION	CONTROL BITS AND SIGNALS[1]				
			P1DIR.x	P1SEL.x	P1SEL2.x	ADC10AE.x INCH.x=1[2]	CAPD.y
P1.0/	0	P1.x (I/O)	I: 0; O: 1	0	0	0	0
TA0CLK/		TA0.TACLK	0	1	0	0	0
ACLK/		ACLK	1	1	0	0	0
A0[2]/		A0	X	X	X	1 (y = 0)	0
CA0/		CA0	X	X	X	0	1 (y = 0)
Pin Osc		Capacitive sensing	X	0	1	0	0
P1.1/	1	P1.x (I/O)	I: 0; O: 1	0	0	0	0
TA0.0/		TA0.0	1	1	0	0	0
		TA0.CCI0A	0	1	0	0	0
UCA0RXD/		UCA0RXD	from USCI	1	1	0	0
UCA0SOMI/		UCA0SOMI	from USCI	1	1	0	0
A1[2]/		A1	X	X	X	1 (y = 1)	0
CA1/		CA1	X	X	X	0	1 (y = 1)
Pin Osc		Capacitive sensing	X	0	1	0	0
P1.2/	2	P1.x (I/O)	I: 0; O: 1	0	0	0	0
TA0.1/		TA0.1	1	1	0	0	0
		TA0.CCI1A	0	1	0	0	0
UCA0TXD/		UCA0TXD	from USCI	1	1	0	0
UCA0SIMO/		UCA0SIMO	from USCI	1	1	0	0
A2[2]/		A2	X	X	X	1 (y = 2)	0
CA2/		CA2	X	X	X	0	1 (y = 2)
Pin Osc		Capacitive sensing	X	0	1	0	0

(1) X = don't care
(2) MSP430G2x53 devices only

UART Character/ Frame Format

- UART character format composition
  - start bit
  - seven or eight data bits
  - even/odd/no parity bit
  - address bit
  - one or two stop bits.



UART Character/ Frame Format

- UART character format option control register

15.4.1 UCAxCTL0, USCI_Ax Control Register 0

7	6	5	4	3	2	1	0
UCPEN	**UCPAR**	**UCMSB**	**UC7BIT**	**UCSPB**	**UCMODEx**		**UCSYNC**
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

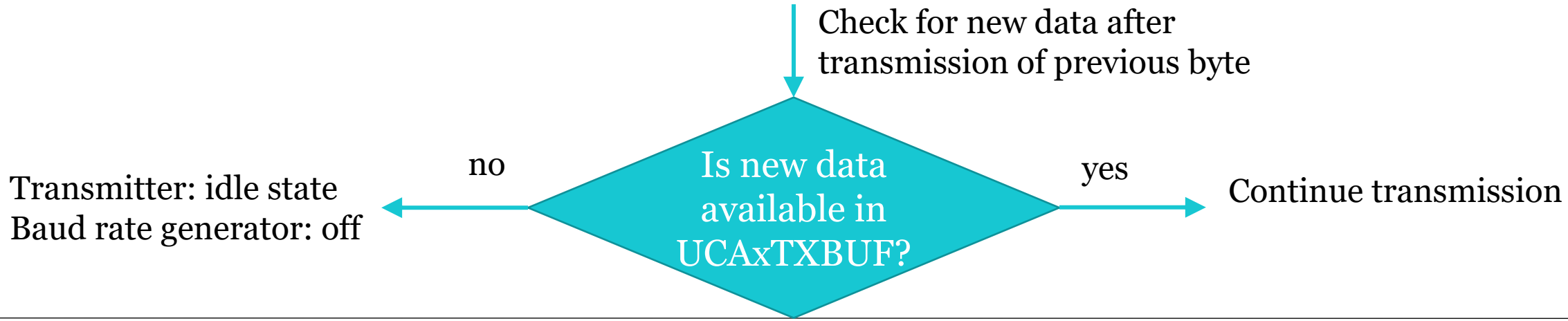
UCPEN: Parity enable
UCPAR: Parity select
UCMSB: MSB first select
UC7BIT: Selects 7-bit or 8-bit character length
UCSPB: Stop bit select. Number of stop bits.
UCMODEx: USCI mode. 00: UART mode.
UCSYNC: Synchronous mode enable. 0: Asynchronous mode.

USCI Transmit Enable

- *USCI module is enabled by clearing the UCSWRST bit*
  - Transmitter is ready and in an idle state
  - Transmit baud rate generator is ready but is not clocked or producing any clocks
- *Transmission is initiated by writing data to UCAxTXBUF*
- When is data written to UCAxTXBUF:
  - baud rate generator is enabled
  - data is moved to transmit shift register on next BITCLK after transmit shift register is empty
- *UCAxTXIFG is set when new data can be written into UCAxTXBUF*

USCI Transmit Enable

- Transmission continues if new data is available in UCAxTXBUF at the end of the previous byte transmission
- If new data is not in UCAxTXBUF when previous byte has transmitted, transmitter returns to idle state and baud rate generator is turned off.



USCI Receive Enable

- USCI module is enabled by clearing UCSWRST bit
  - Receiver is ready and in idle state
  - Receive baud rate generator is in ready state but is not clocked no producing any clocks
- Falling edge of start bit enables baud rate generator and UART state machine checks for a valid start bit.
  - If no valid start bit: UART state machine returns to idle state and baud rate generator is turned off again.
  - If a valid start bit: character will be received.

USCI Receive Enable

- When idle-line multiprocessor mode is selected
  - UART state machine checks for idle line after receiving a character.
- If a start bit is detected another character is received
- Else UCIDLE flag is set after 10 ones are received and UART returns to idle state and baud rate generator is turned off.

Receive Data Glitch Suppression

- Glitch suppression prevents USCI from being accidentally started.
  - Ignore any glitch on UCAxRXD shorter than deglitch time t_T (approx. 150 ns)
- If a glitch is longer than t_T or a valid start bit occurs on UCAxRXD:
  - USCI receive operation is started and a majority vote is taken
  - If the majority vote fails to detect a start bit USCI halts reception

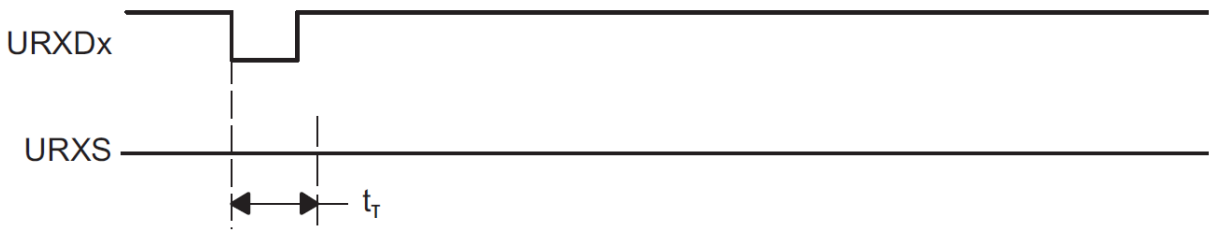
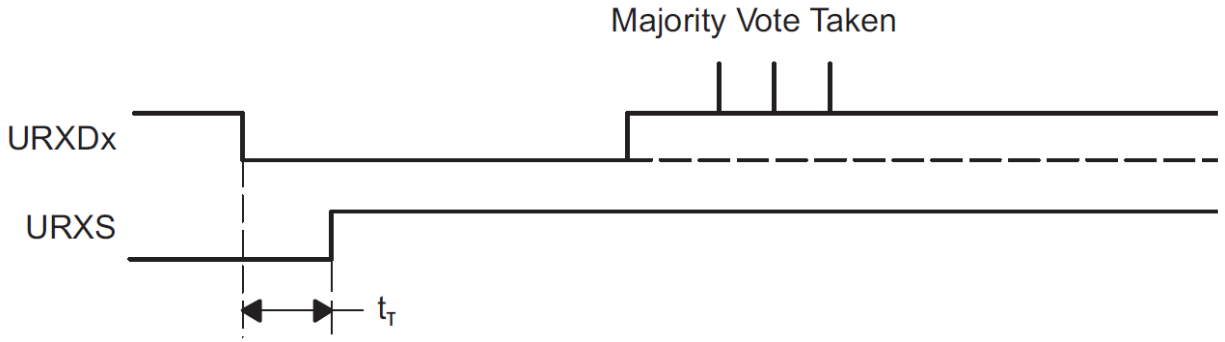


Figure 15-8. Glitch Suppression, USCI Receive Not Started



Glitch Suppression, USCI Activated

MSP430 USCI Baud Rate Generation and Interrupts

UART Baud Rate Generation

- Baud rate different than driving clock frequency
- Mismatch in baud rate due to clock divider limitation
- Additional compensation necessary to reduce the error due to mismatch

Example for the baud rate 9600 when BRCLK = 1 MHz:
N =1000000/9600 = 104.1667

Frequency divider (counter) can only use the integer portion 104:
Actual baud rate = 1000000/104 = 9615.3846

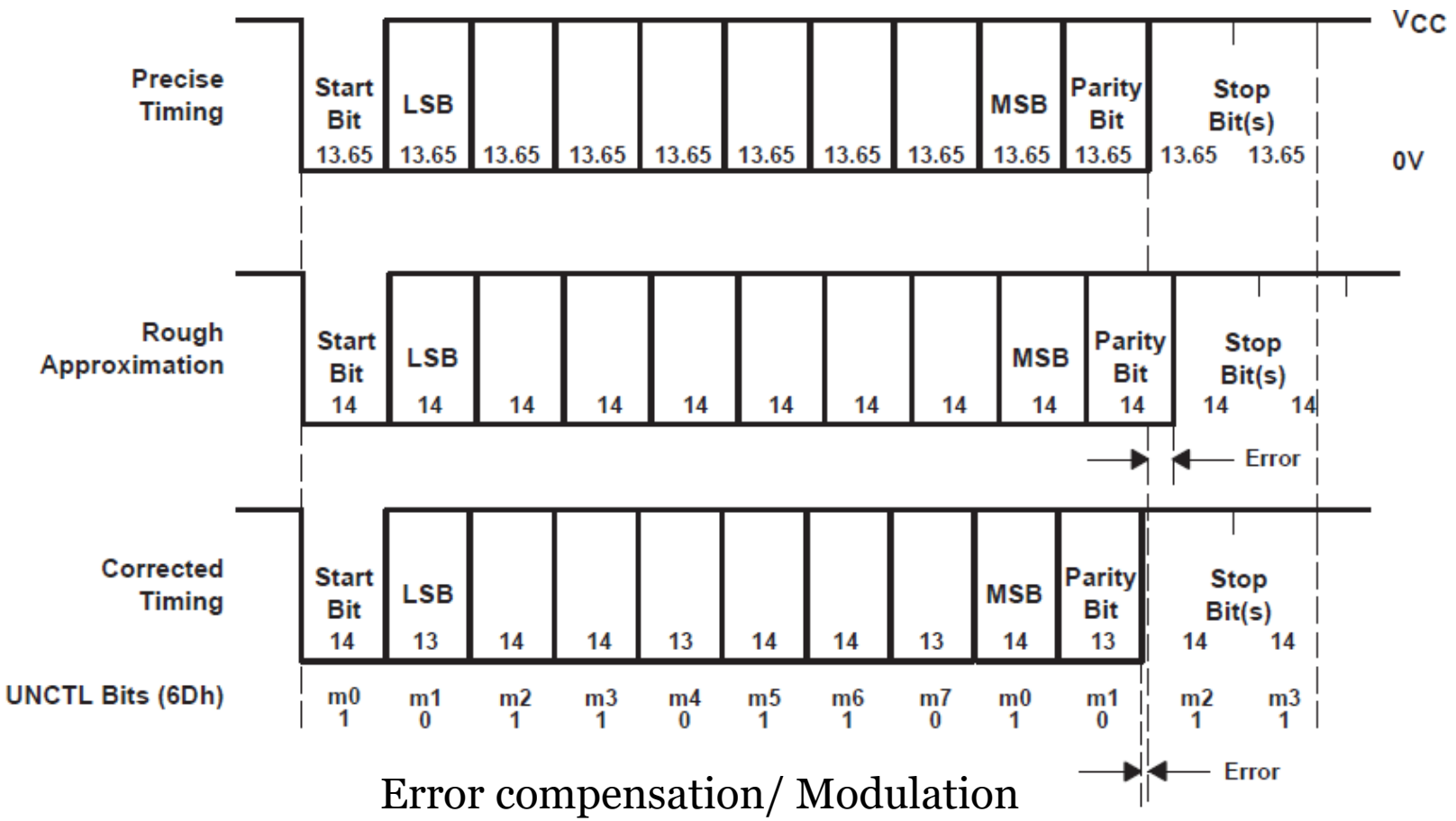
UART Baud Rate Generation

- Baud rate different than driving clock frequency
- Mismatch in baud rate due to clock divider limitation
- Additional compensation necessary to reduce the error due to mismatch

Example for the baud rate 9600 when BRCLK = 32768 Hz:
N =32768/9600 = 3.4133

Frequency divider (counter) can only use the integer portion 3:
Actual baud rate = 32768/3 = 10922.66

Baudrate Correction: Example



Example: 2400-baud rate generated with ACLK frequency (32,768 Hz)

Actual bit-length: 13.653 ACLK cycles (32,768/2400 = 13.65333)
Approximate: 14 clk cycles

(Page 8, MSP430 UART slaa049.pdf)

UART Baud Rate Generation

- USCI baud rate generator produces standard baud rates from non-standard source frequencies
- Three registers to configure baud rate generation
  - UCAxBR1/ UCAxBR0: Baud rate control registers
  - UCA_MCTL: Modulation Control Registers
- Two modes of operation selected by UCOS16 bit
  - Low-frequency mode UCOS16 = 0
  - Oversampling mode UCOS16 = 1

UART Baud Rate Generation

- Three registers to configure baud rate generation
  - UCAxBR1/ UCAxBR0: Baud rate control registers
  - UCA_MCTL: Modulation Control Registers
- UCAxBR1/ UCAxBR0: Baud rate control registers
  - UCBRx: clock prescaler values
- UCA_MCTL: Modulation Control Registers
  - UCBRSx: modulator values
  - UCOS16: sampling modes

UART Baud Rate Generation: Low Frequency Mode

- Two modes of operation selected by UCOS16 bit
  - Low-frequency mode UCOS16 = 0
  - Oversampling mode UCOS16 = 1
- Low frequency mode
  - Generates baud rates from low frequency clock sources
  - Baud rate generator uses one prescaler and one modulator to generate bit clock timing
  - maximum USCI baud rate is one-third the UART source clock frequency BRCLK

UART Baud Rate Generation: Oversampling Mode

- Two modes of operation selected by UCOS16 bit
  - Low-frequency mode UCOS16 = 0
  - Oversampling mode UCOS16 = 1
- Oversampling mode
  - maximum USCI baud rate is 1/16 the UART source clock frequency BRCLK
  - uses one prescaler and one modulator to generate BITCLK16 clock that is 16 times faster than BITCLK

Setting Baud Rates

- For a given BRCLK clock source, baud rate determines division factor N

$$N = \frac{f_{\text{BRCLK}}}{\text{Baud rate}}$$

- N is often a non-integer value
  - At least one divider and one modulator is used to match N as closely as possible.
- If N ≥16, oversampling baud rate generation mode can be chosen by setting UCOS16.

Example for the baud rate 9600 when BRCLK = 1 MHz:
N =1000000/9600 = 104.1667

Frequency divider (counter) can only use the integer portion 104:
Actual baud rate = 1000000/104 = 9615.3846

Setting Baud Rate: Low-Frequency Mode

- In low-frequency mode, integer portion of divisor is realized by prescaler

$$\text{UCBRx} = integer(N)$$

- Fractional portion is realized by the modulator using formula

$$\text{UCBRSx} = round\left(\left(N - int(N)\right) \times 8\right)$$

Example:
N = 1000000/9600 = 104.1667
UCBRx = integer(104.1667) = 104
UCBRSx = round((104.1667-104)x8) = round(1.33) = 1

Table 15-4. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 0

BRCLK Frequency [Hz]	Baud Rate [Baud]	UCBRx	UCBRSx	UCBRFx	Maximum TX Error [%]		Maximum RX Error [%]	
32,768	1200	27	2	0	-2.8	1.4	-5.9	2.0
32,768	2400	13	6	0	-4.8	6.0	-9.7	8.3
32,768	4800	6	7	0	-12.1	5.7	-13.4	19.0
32,768	9600	3	3	0	-21.1	15.2	-44.3	21.3
1,048,576	9600	109	2	0	-0.2	0.7	-1.0	0.8
1,048,576	19200	54	5	0	-1.1	1.0	-1.5	2.5
1,048,576	38400	27	2	0	-2.8	1.4	-5.9	2.0
1,048,576	56000	18	6	0	-3.9	1.1	-4.6	5.7
1,048,576	115200	9	1	0	-1.1	10.7	-11.5	11.3
1,048,576	128000	8	1	0	-8.9	7.5	-13.8	14.8
1,048,576	256000	4	1	0	-2.3	25.4	-13.4	38.8
1,000,000	9600	104	1	0	-0.5	0.6	-0.9	1.2
1,000,000	19200	52	0	0	-1.8	0	-2.6	0.9
1,000,000	38400	26	0	0	-1.8	0	-3.6	1.8
1,000,000	56000	17	7	0	-4.8	0.8	-8.0	3.2
1,000,000	115200	8	6	0	-7.8	6.4	-9.7	16.1
1,000,000	128000	7	7	0	-10.4	6.4	-18.0	11.6
1,000,000	256000	3	7	0	-29.6	0	-43.6	5.2

Setting Baud Rate: Oversampling Mode

- In the oversampling mode the prescaler is set to

$$\text{UCBRx} = integer\left(\frac{N}{16}\right)$$

- first stage modulator is set to

$$\text{UCBRFx} = round\left(\left(\frac{N}{16} - int\left(\frac{N}{16}\right)\right) \times 16\right)$$

Example:
N = 1000000/9600 = 104.1667
UCBRx = integer(104.1667/16) = integer(6.51)=6
UCBRFx = round((6.51-6)x16) = round(8.1666) = 8

Baud Rate Control Registers

15.4.3 UCAxBR0, USCI_Ax Baud Rate Control Register 0

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

15.4.4 UCAxBR1, USCI_Ax Baud Rate Control Register 1

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

UCBRx	7-0	Clock prescaler setting of the Baud rate generator. The 16-bit value of (UCAxBR0 + UCAxBR1 × 256) forms the prescaler value.

Modulation Control Registers

15.4.5 UCAxMCTL, USCI_Ax Modulation Control Register

7	6	5	4	3	2	1	0
UCBRFx				UCBRSx			UCOS16
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

UCBRFx	Bits 7-4	First modulation stage select. These bits determine the modulation pattern for BITCLK16 when UCOS16 = 1. Ignored with UCOS16 = 0. Table 15-3 shows the modulation pattern.
UCBRSx	Bits 3-1	Second modulation stage select. These bits determine the modulation pattern for BITCLK. Table 15-2 shows the modulation pattern.
UCOS16	Bit 0	Oversampling mode enabled 0 Disabled 1 Enabled

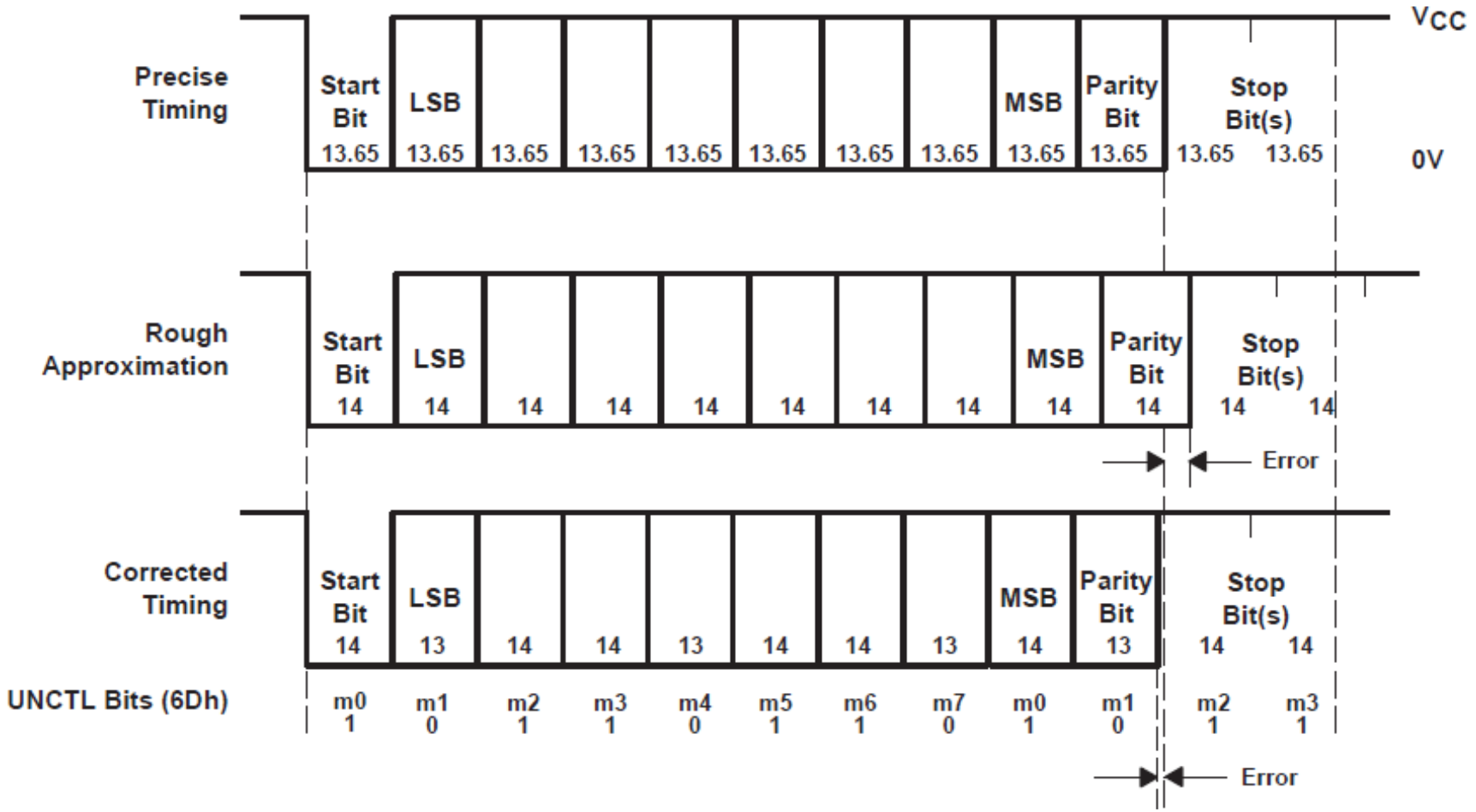
BITCLK Modulation Pattern

Table 15-2. BITCLK Modulation Pattern

UCBRSx	Bit 0 (Start Bit)	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0
3	0	1	0	1	0	1	0	0
4	0	1	0	1	0	1	0	1
5	0	1	1	1	0	1	0	1
6	0	1	1	1	0	1	1	1
7	0	1	1	1	1	1	1	1

Modulation is based on the UCBRSx setting as shown in Table 15-2. A “1" in the table indicates that m = 1 and the corresponding BITCLK period is one BRCLK period longer than a BITCLK period with m = 0. The modulation wraps around after 8 bits but restarts with each new start bit.

Baud Rate Correction: Example



(Page 8, MSP430 UART slaa049.pdf)

USCI Interrupts

- USCI has one interrupt vector for transmission and one interrupt vector for reception.
- USCI Transmit Interrupt Operation: USCIABoTX_VECTOR
- USCI Receive Interrupt Operation: USCIABoRX_VECTOR

USCI Transmit Interrupts Operation

- UCAxTXIFG interrupt flag is set by transmitter to indicate that UCAxTXBUF is ready to accept another character
- Interrupt request is generated if UCAxTXIE and GIE are also set
- UCAxTXIFG Interrupt flag
  - Automatically reset if a character is written to UCAxTXBUF
  - set after a PUC or when UCSWRST = 1.
- UCAxTXIE is reset after a PUC or when UCSWRST = 1.

USCI Receive Interrupt Operation

- UCAxRXIFG interrupt flag is set each time a character is received and loaded into UCAxRXBUF
- Interrupt request is generated if UCAxRXIE and GIE are also set
- UCAxRXIFG and UCAxRXIE are reset by a system reset PUC signal or when UCSWRST = 1
- UCAxRXIFG is automatically reset when UCAxRXBUF is read.

Configure UART step by step

- Initializing or Re-Configuring the USCI Module

1. Set UCSWRST: UCA0CTL1 |= UCSWRST;
2. Initialize all USCI registers with UCSWRST = 1 (including UCAxCTL1)
3. Configure IO ports for Tx/Rx pins
4. Configure UART registers
   - UCA0BR0 = x; // Least significant byte of divider
   - UCA0BR1 = y; // Most significant byte of divider
   - UCA0MCTL = UCBRS_1; // Modulation UCBRSx = 1
5. Clear UCSWRST via software: UCA0CTL1 &= ~UCSWRST;
6. Enable interrupts via UCAxRXIE and/or UCAxTXIE: IE2 |= UCA0RXIE;

Setting Prescaler and Modulator in C code

```
UCA0CTL1 |= UCSWRST;          // Set UCSWRST
UCA0CTL1 |= UCSSEL_2;         // SMCLK
UCA0BR0 = x;                  // Least significant byte of divider
UCA0BR1 = y;                  // Most significant byte of divider
UCA0MCTL = UCBRS_1;             // Modulation UCBRSx = 1
UCA0CTL1 &= ~UCSWRST;         // Initialize USCI state machine
IE2 |= UCA0RXIE;              // Enable USCI_A0 RX interrupt
```

```
#define UCBRS2      (0x08)  /* USCI Second Stage Modulation Select 2 */
#define UCBRS1      (0x04)  /* USCI Second Stage Modulation Select 1 */
#define UCBRS0      (0x02)  /* USCI Second Stage Modulation Select 0 */

#define UCBRS_0     (0x00)  /* USCI Second Stage Modulation: 0 */
#define UCBRS_1     (0x02)  /* USCI Second Stage Modulation: 1 */
#define UCBRS_2     (0x04)  /* USCI Second Stage Modulation: 2 */
#define UCBRS_3     (0x06)  /* USCI Second Stage Modulation: 3 */
#define UCBRS_4     (0x08)  /* USCI Second Stage Modulation: 4 */
#define UCBRS_5     (0x0A)  /* USCI Second Stage Modulation: 5 */
#define UCBRS_6     (0x0C)  /* USCI Second Stage Modulation: 6 */
#define UCBRS_7     (0x0E)  /* USCI Second Stage Modulation: 7 */
```

MSP430 Universal Serial Communication Interface (USCI) Registers

UART Control Register 0

15.4.1 UCAxCTL0, USCI_Ax Control Register 0

7	6	5	4	3	2	1	0
UCPEN	**UCPAR**	**UCMSB**	**UC7BIT**	**UCSPB**	**UCMODEx**		**UCSYNC**
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

UCPEN	Bit 7	Parity enable	
		0	Parity disabled.
		1	Parity enabled. Parity bit is generated (UCAxTXD) and expected (UCAxRXD). In address-bit multiprocessor mode, the address bit is included in the parity calculation.
UCPAR	Bit 6	Parity select. UCPAR is not used when parity is disabled.	
		0	Odd parity
		1	Even parity
UCMSB	Bit 5	MSB first select. Controls the direction of the receive and transmit shift register.	
		0	LSB first
		1	MSB first
UC7BIT	Bit 4	Character length. Selects 7-bit or 8-bit character length.	
		0	8-bit data
		1	7-bit data

UART Control Register 0

15.4.1 UCAxCTL0, USCI_Ax Control Register 0

7	6	5	4	3	2	1	0
UCPEN	UCPAR	UCMSB	UC7BIT	UCSPB	UCMODEx		UCSYNC
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

UCSPB	Bit 3	Stop bit select. Number of stop bits.
		0 One stop bit
		1 Two stop bits
UCMODEx	Bits 2-1	USCI mode. The UCMODEx bits select the asynchronous mode when UCSYNC = 0.
		00 UART mode
		01 Idle-line multiprocessor mode
		10 Address-bit multiprocessor mode
		11 UART mode with automatic baud rate detection
UCSYNC	Bit 0	Synchronous mode enable
		0 Asynchronous mode
		1 Synchronous mode

UART Control Register 1

15.4.2 UCAxCTL1, USCI_Ax Control Register 1

7	6	5	4	3	2	1	0
UCSSELx		UCRXEIE	UCBRKIE	UCDORM	UCTXADDR	UCTXBRK	UCSWRST
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-1

UCSSELx Bits 7-6 USCI clock source select. These bits select the BRCLK source clock.

- 00 UCLK
- 01 ACLK
- 10 SMCLK
- 11 SMCLK

UCRXEIE Bit 5 Receive erroneous-character interrupt-enable

- 0 Erroneous characters rejected and UCAxRXIFG is not set
- 1 Erroneous characters received will set UCAxRXIFG

UCBRKIE Bit 4 Receive break character interrupt-enable

- 0 Received break characters do not set UCAxRXIFG.
- 1 Received break characters set UCAxRXIFG.

UART Control Register 1

15.4.2 UCAxCTL1, USCI_Ax Control Register 1

7	6	5	4	3	2	1	0
UCSSELx		UCRXEIE	UCBRKIE	UCDORM	UCTXADDR	UCTXBRK	UCSWRST
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-1

UCDORM Bit 3 Dormant. Puts USCI into sleep mode.

- 0 Not dormant. All received characters will set UCAxRXIFG.
- 1 Dormant. Only characters that are preceded by an idle-line or with address bit set will set UCAxRXIFG. In UART mode with automatic baud rate detection only the combination of a break and synch field will set UCAxRXIFG.

UCTXADDR Bit 2 Transmit address. Next frame to be transmitted will be marked as address depending on the selected multiprocessor mode.

- 0 Next frame transmitted is data
- 1 Next frame transmitted is an address

UCTXBRK Bit 1 Transmit break. Transmits a break with the next write to the transmit buffer. In UART mode with automatic baud rate detection 055h must be written into UCAxTXBUF to generate the required break/synch fields. Otherwise 0h must be written into the transmit buffer.

- 0 Next frame transmitted is not a break
- 1 Next frame transmitted is a break or a break/synch

UCSWRST Bit 0 Software reset enable

- 0 Disabled. USCI reset released for operation.
- 1 Enabled. USCI logic held in reset state.

Baud Rate Control Registers

15.4.3 UCAxBR0, USCI_Ax Baud Rate Control Register 0

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

15.4.4 UCAxBR1, USCI_Ax Baud Rate Control Register 1

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

UCBRx	7-0	Clock prescaler setting of the Baud rate generator. The 16-bit value of (UCAxBR0 + UCAxBR1 × 256) forms the prescaler value.

Modulation Control Register

15.4.5 UCAxMCTL, USCI_Ax Modulation Control Register

7	6	5	4	3	2	1	0
UCBRFx				UCBRSx			UCOS16
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

UCBRFx	Bits 7-4	First modulation stage select. These bits determine the modulation pattern for BITCLK16 when UCOS16 = 1. Ignored with UCOS16 = 0. Table 15-3 shows the modulation pattern.
UCBRSx	Bits 3-1	Second modulation stage select. These bits determine the modulation pattern for BITCLK. Table 15-2 shows the modulation pattern.
UCOS16	Bit 0	Oversampling mode enabled 0 Disabled 1 Enabled

Transmit and Receive Buffers

15.4.7 UCAxRXBUF, USCI_Ax Receive Buffer Register

7	6	5	4	3	2	1	0
UCRXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

UCRXBUFx	Bits 7-0	The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCAxRXBUF resets the receive-error bits, the UCADDR or UCIDLE bit, and UCAxRXIFG. In 7-bit data mode, UCAxRXBUF is LSB justified and the MSB is always reset.

15.4.8 UCAxTXBUF, USCI_Ax Transmit Buffer Register

7	6	5	4	3	2	1	0
UCTXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

UCTXBUFx	Bits 7-0	The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted on UCAxTXD. Writing to the transmit data buffer clears UCAxTXIFG. The MSB of UCAxTXBUF is not used for 7-bit data and is reset.

Tx/Rx Interrupts

15.4.12 IE2, Interrupt Enable Register 2

7	6	5	4	3	2	1	0
						UCA0TXIE	UCA0RXIE
						rw-0	rw-0

	Bits 7-2	These bits may be used by other modules (see the device-specific data sheet).
UCA0TXIE	Bit 1	USCI_A0 transmit interrupt enable
		0 Interrupt disabled
		1 Interrupt enabled
UCA0RXIE	Bit 0	USCI_A0 receive interrupt enable
		0 Interrupt disabled
		1 Interrupt enabled

15.4.13 IFG2, Interrupt Flag Register 2

7	6	5	4	3	2	1	0
						UCA0TXIFG	UCA0RXIFG
						rw-1	rw-0

	Bits 7-2	These bits may be used by other modules (see the device-specific data sheet).
UCA0TXIFG	Bit 1	USCI_A0 transmit interrupt flag. UCA0TXIFG is set when UCA0TXBUF is empty.
		0 No interrupt pending
		1 Interrupt pending
UCA0RXIFG	Bit 0	USCI_A0 receive interrupt flag. UCA0RXIFG is set when UCA0RXBUF has received a complete character.
		0 No interrupt pending
		1 Interrupt pending

LINKÖPING UNIVERSITY