# Serial Communications I

**ENCE361 Embedded Systems 1**

Course Coordinator: Ciaran Moore (ciaran.moore@Canterbury.ac.nz)

Lecturer: Le Yang (le.yang@canterbury.ac.nz)

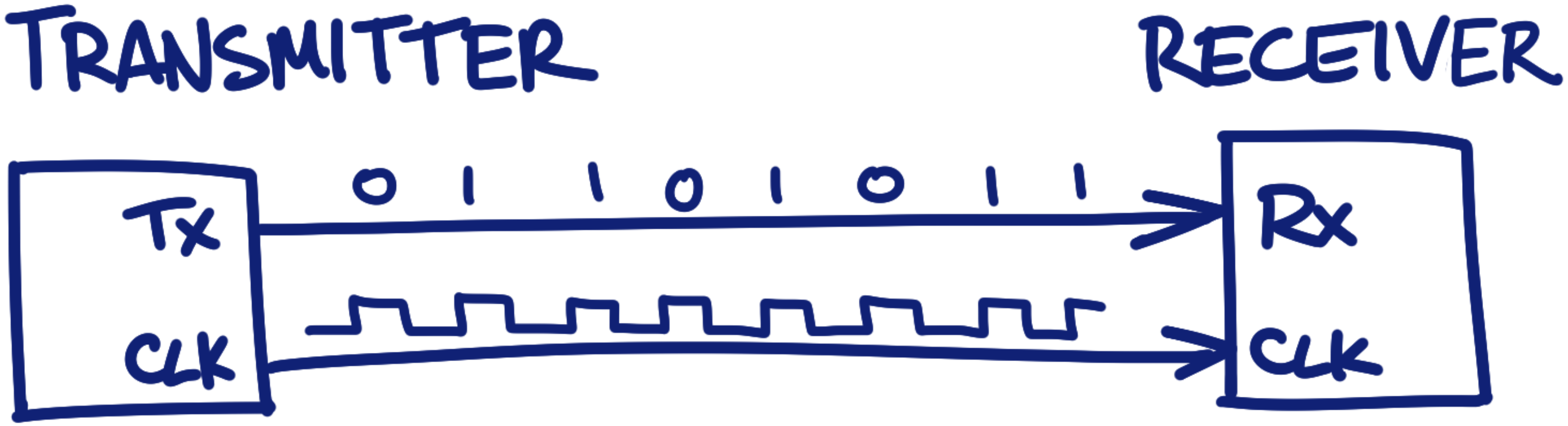
Department of Electrical and Computer Engineering

# Where we’re going today

- **Serial communication basics**
- UART on Tiva C-series launchpad
- Example code
- Homework

# Serial Communications Basics

- Serial communications
  - Send/receive data bits over a wire one bit at a time
    - Normally the signal is referenced with respect to the ground at both ends
  - Examples: USB (Universal Serial Bus), Ethernet …
  - Simplex, full duplex and half duplex



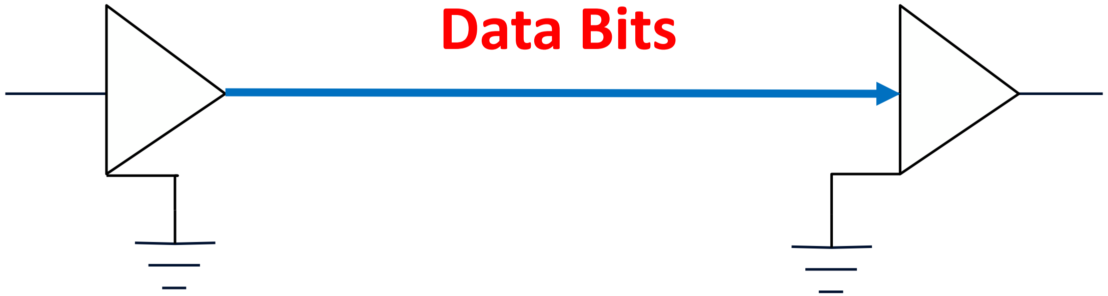
Courtesy of Dr. Allan McInnes

# Simplex Transmission

- Simplex transmission
  - One wire and data transmission is unidirectional
  - Transmit (Tx) buffer used at the transmitter
  - Receive (Rx) buffer used at the receiver
  - Data transmitted in packets

Transmitter

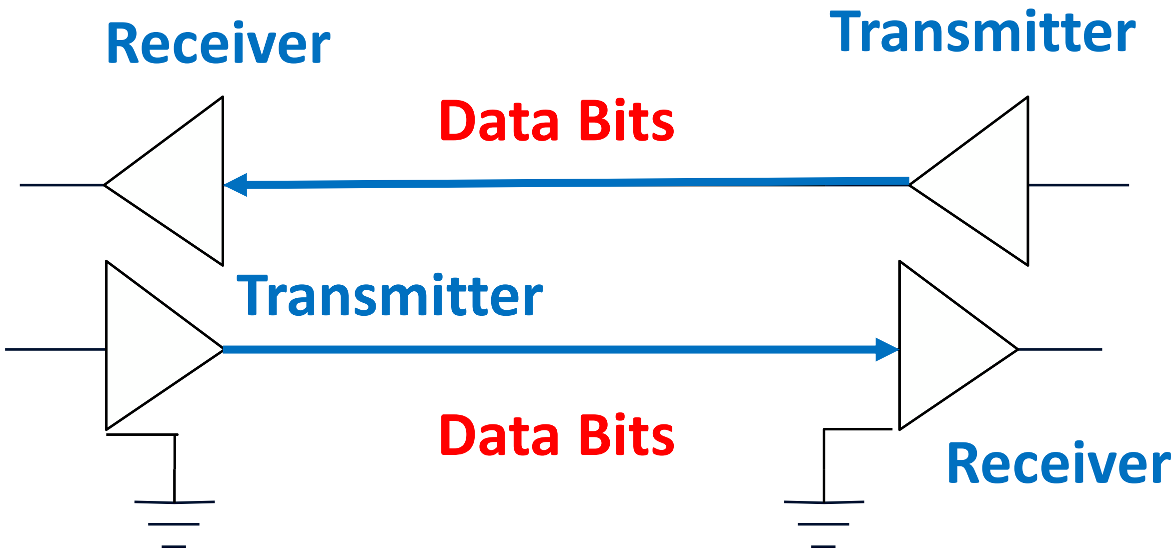
Receiver


Data Bits

Courtesy of Dr. Allan McInnes

# Full Duplex Transmission

- Full duplex transmission
  - Two wires and data transmission can be in two directions simultaneously
  - Transmit (Tx) and receive (Rx) buffers used at both ends


Receiver

Transmitter

Data Bits

Transmitter

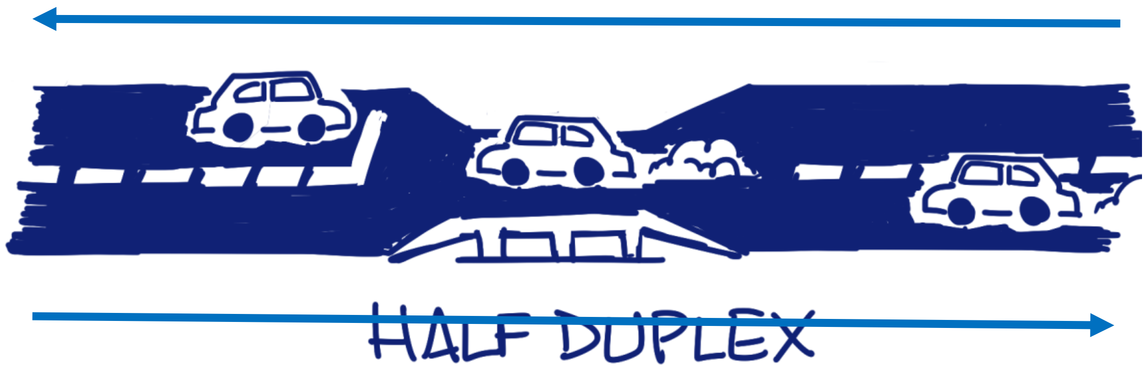
Data Bits

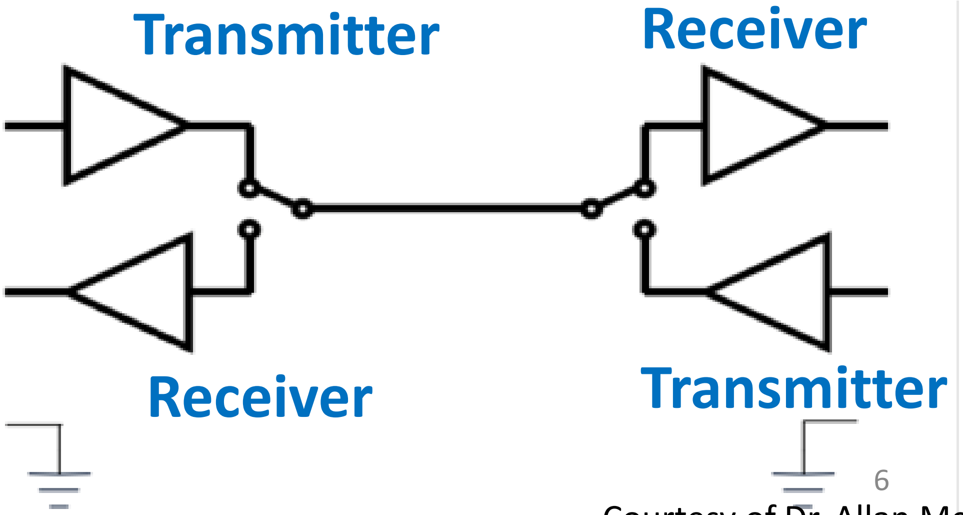
Receiver

Courtesy of Dr. Allan McInnes

# Half Duplex Transmission

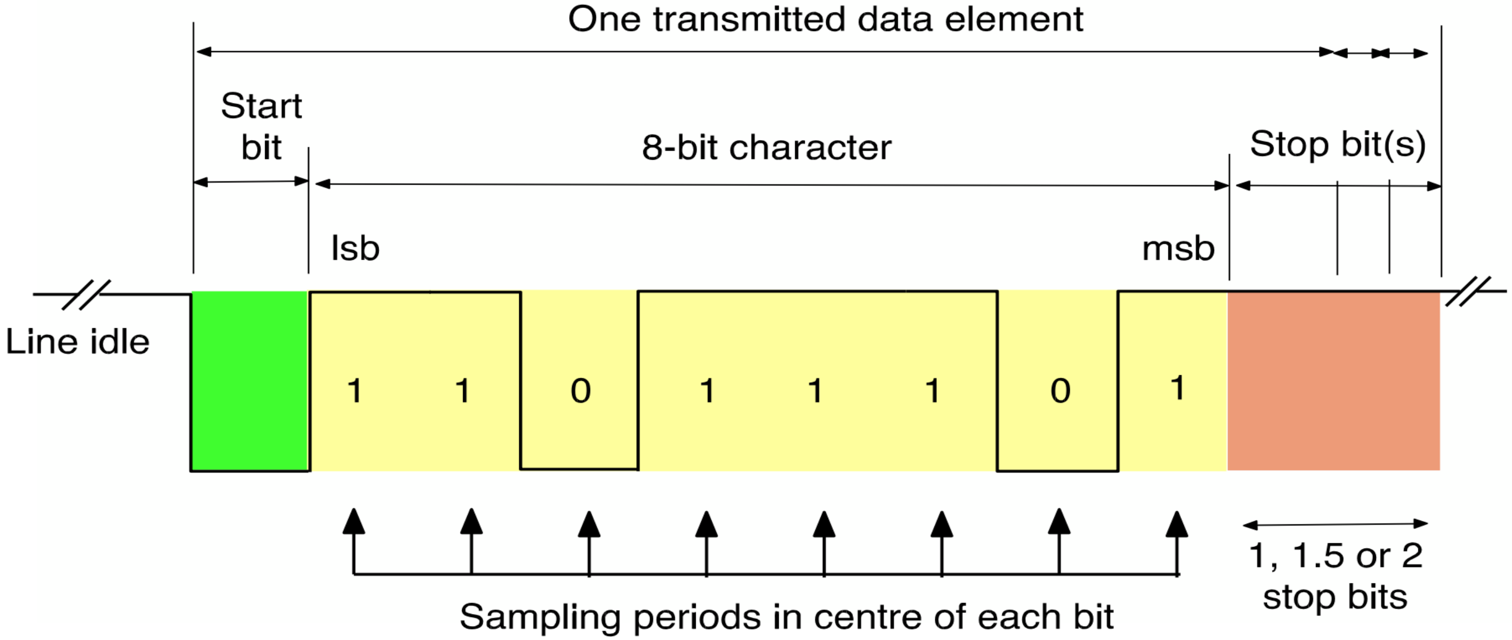
- Half duplex transmission
  - One wire and data transmission can be bidirectional
    - Time-division multiplexing (TDM) of the wire
    - Data rate is halved compared with full duplex mode
    - Switches at both ends cooperate to determine transmission direction
  - Transmit (Tx) and receive (Rx) buffers used at both ends





Courtesy of Dr. Allan McInnes

# Typical Serial Packet Structure

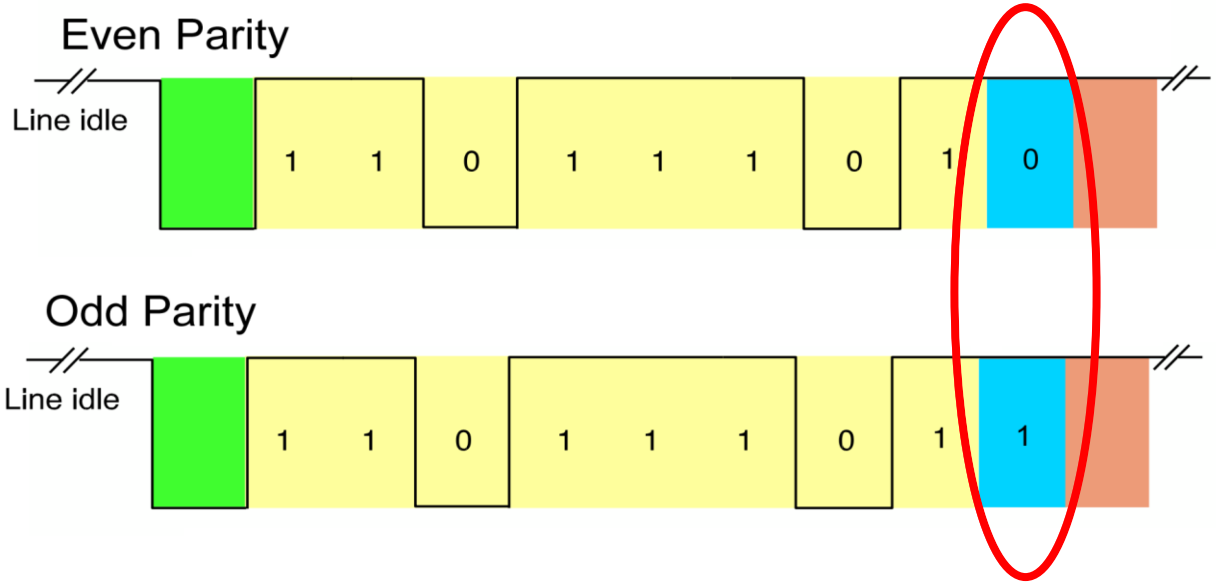


- Line idle (‘1’)
- Start bit (‘0’)
- Data byte (e.g., 8-bit character) with LSB first and MSB last ()
- Parity bit (not shown here)
- Stop bit(s)
- Line idle (‘1’)

- Non-return to zero (NRZ) coding
- **Baud rate** in bits/second

# Parity Bit

- Optional but simple way to detect error in transmission
  - Error detection coding
- Even parity
  - Number of 1’s in data + parity bit is even
- Odd parity
  - Number of 1’s in data + parity bit is odd

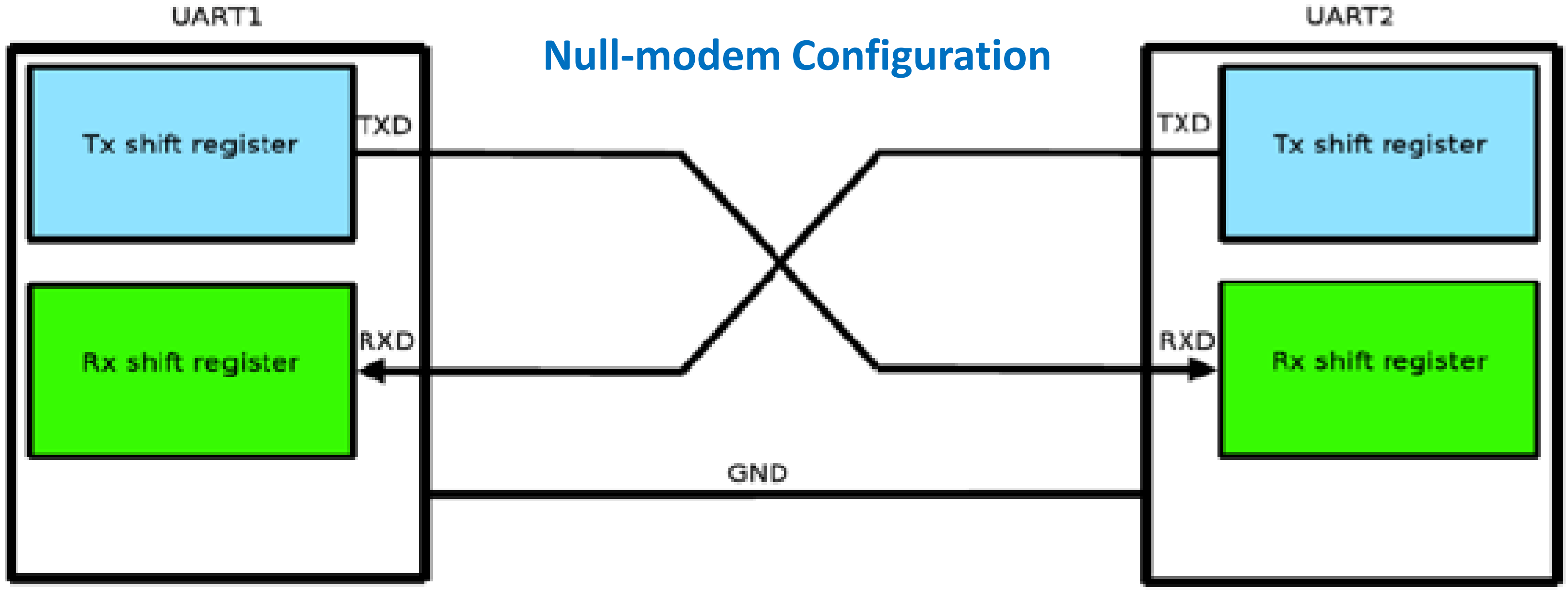


# Where we’re going today

- Serial communication basics
- **UART on Tiva C-series launchpad**
- Example code
- Homework

# UART on Tiva C-Series Launchpad (1)

- UART: Universal Asynchronous Receiver Transmitter
  - Available in many MCUs, usually full duplex
    - Null-modem configuration (i.e., crossing over Tx and Rx wires) for MCU to MCU
  - Often with extra *handshaking* pins for flow control (e.g., clear to send (CTS))



# UART on Tiva C-Series Launchpad (2)

- Eight UARTs (U0 – U7)
  - Each with 16-byte FIFO buffers for Tx and Rx

  - Programmable baud-rate generator up to 10 Mbps
  - Auto generation and stripping of start, stop and parity bits

  - Configurable number of data, stop and parity bits
    - 5, 6, 7 or 8 data bits
    - Even/odd/no parity bit
    - 1 or 2 stop bits

  - Support flow control

# Virtual COM Port

- Virtual COM port (VCP) allows Windows applications to communicate with Tiva MCU via its UART0 (U0) over a USB cable

- Windows assigns a COM port number to VCP channel

- Terminal view within CCS
  - Useful for monitoring output from the Tiva MCU

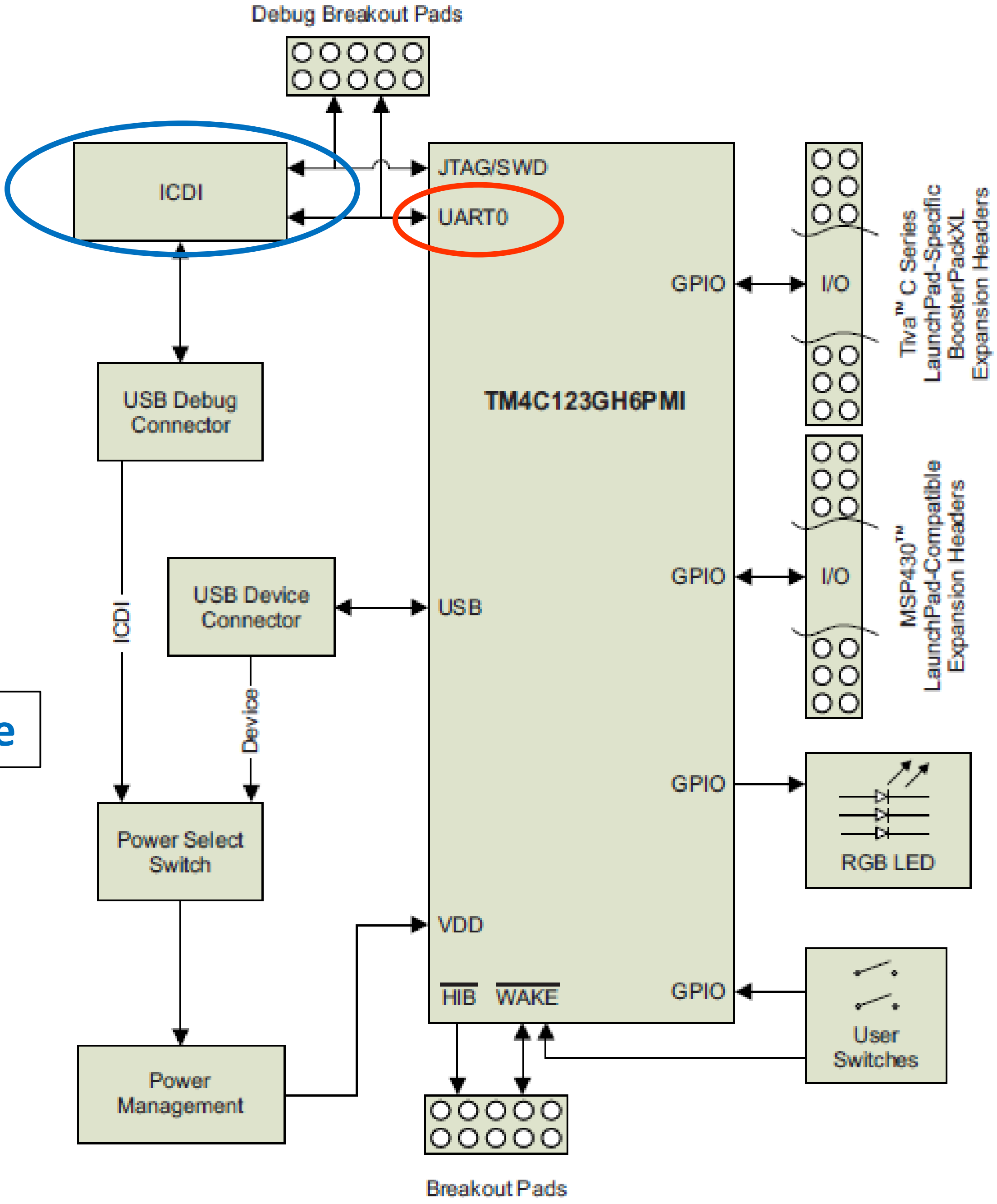
  - Baud rate: 9600 bps, 1 stop bit, no parity bits, encoding UTF-8
    - UTF-8: variable width (1-4 bytes) character encoding scheme capable of encoding 1,112,064 valid code points in Unicode

From Table 2-8 on Page 103 in TM4C123GH6PM.pdf

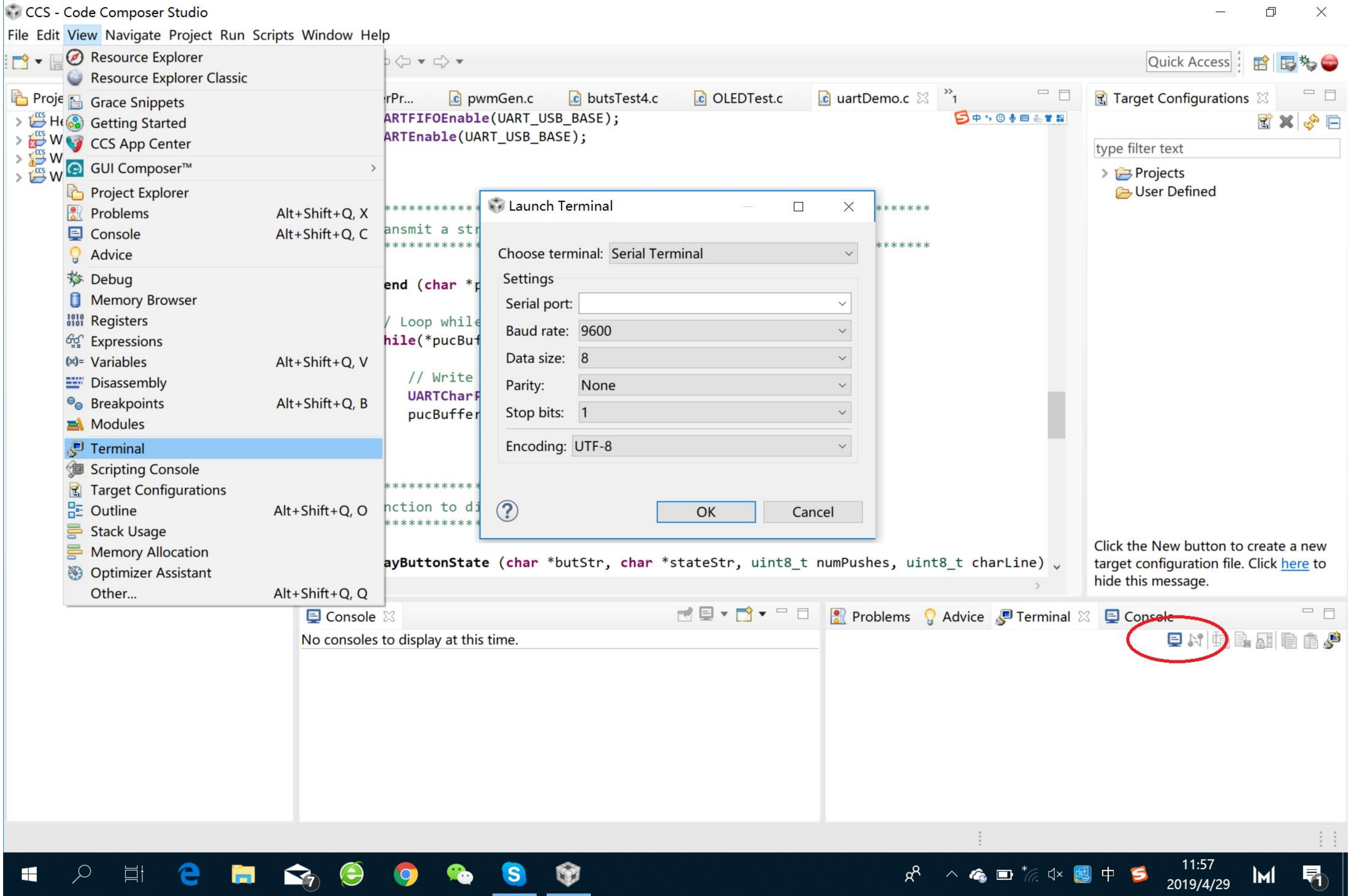
**UART0 (U0) is use**

- PA0: U0Rx
- PA1: U0Tx

**ICDI = In-Circuit Debug Interface**



From Page 7, TivaTM4C123G Launchpad Evaluation Board_Users Guide.pdf



```
ARTFIFOEnable(UART_USB_BASE);
ARTEnable(UART_USB_BASE);

***********                                    *******
ansmit a str
***********                                    *******

end (char *p

/ Loop while
hile(*pucBuf

    // Write
    UARTCharP
    pucBuffer

***********
nction to di
***********

ayButtonState (char *butStr, char *stateStr, uint8_t numPushes, uint8_t charLine)
```

Launch Terminal

Choose terminal: Serial Terminal

Settings

Serial port:

Baud rate: 9600

Data size: 8

Parity: None

Stop bits: 1

Encoding: UTF-8

No consoles to display at this time.

Click the New button to create a new target configuration file. Click here to hide this message.

# Where we’re going today

- Serial communication basics
- UART on Tiva C-series launchpad
- **Example code**
- Homework

# Example Code (1)

From UARTdemo.c

```
//************************************************************
// Initialise USB_UART - 8 bits, 1 stop bit, no parity
//************************************************************
void initialiseUSB_UART (void)
{
   // Enable GPIO port A which is used for UART0 pins (see Page 12, TivaTM4C123G Launchpad
   // Evaluation Board_Users Guide.pdf).
   SysCtlPeripheralEnable (SYSCTL_PERIPH_UART0);
   SysCtlPeripheralEnable (SYSCTL_PERIPH_GPIOA);

   // Select the alternate (UART) function for these pins.
   GPIOPinTypeUART (GPIO_PORTA_BASE, GPIO_PIN_0 | GPIO_PIN_1);
   UARTConfigSetExpClk (UART0_BASE, SysCtlClockGet(), BAUD_RATE,
                              UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE |
                              UART_CONFIG_PAR_NONE);
   UARTFIFOEnable (UART0_BASE);
   UARTEnable (UART0_BASE);
}
```

# Example Code (2)

```
//*****************************************************************************
// Transmit a string via UART0
//*****************************************************************************
void UARTSend (char *pucBuffer)
{
    // Loop while there are more characters to send.
    while(*pucBuffer)
    {
        // Write the next character to the UART Tx FIFO.
        UARTCharPut(UART0_BASE, *pucBuffer);
        pucBuffer ++;
    }
}
```

From UARTdemo.c

# Example Code (3)

From UARTdemo.c

```
while(1)
{
  // check state of each button and display if changed.
  butState = checkButton (UP);
  switch (butState)
 {
     . . . . . . .
  } // Do nothing if state is NO_CHANGE.

  // Is it time to send a message?
  if (slowTick)
  {
     slowTick = false;
     // Form and send a status message to the console.
     sprintf (statusStr, "UP=%2d DN=%2d | ", upPushes, downPushes);
     UARTSend (statusStr);
  }
}
```

# Blocking vs. Non-Blocking Transmission

- Wait to send a character from a specified port
  - void **UARTCharPut** (unsigned long ulBase, unsigned char ucData)
    - ulBase: based address of UART port
    - ucData: character to be transmitted
  - If there is no space in Tx FIFO, wait until there is a space before returning
- Try sending a character from a specified port
  - tBoolean **UARTCharPutNonBlocking** (unsigned long ulBase, unsigned char ucData)
  - Return **TRUE** if the character is successfully place in the Tx FIFO
  - Return **FALSE** if there is no space in Tx FIFO
- What are their advantages and disadvantages? (see homework)

From page 561 in TivaWare Peripheral Driver Library Users Manual.pdf

# Homework

1. A particular UART-to-UART transmission is set for 1 start bit, 8-bit data, odd parity and 2 stop bits. Which of the following 4 strings are received without detectable error and what is the data word in each of those cases?

i. 010101011111

ii. 001110001111

iii. 001001101111

iv. 011110001001

2. UART units on the Tiva Microcontroller have Tx and Rx FIFO buffers. What is the difference between a FIFO buffer and a circular buffer?
3. What are the advantages and disadvantages of using a *blocking* Tx function?
4. How might you decide between using *blocking* or *non-blocking* Tx/Rx functions in your helicopter project?