1300 Henley Court
Pullman, WA 99163
509.334.6306
www.store.digilent.com

Lab 4a: Universal Asynchronous Receiver/Transmitter

Revised April 25, 2017
This manual applies to Unit 4, Lab 4a.

1 Objectives

1. Setup a terminal monitor on a PC or Linux workstation.
2. Use the PIC32MX370 to send an ASCII encoded text string and display the characters on the workstation monitor.
3. Receive an ASCII encoded text string from the workstation monitor and display it on the Basys MX3 LCD.

2 Basic Knowledge

1. ASCII code.
2. I/O configuration for PPS Processors.
3. How to initialize special function using the PLIB functions.

3 Equipment List

3.1 Hardware

1. Basys MX3 trainer board
2. Workstation computer running Windows 10 or higher, MAC OS, or Linux
3. 2 Standard USB A to micro-B cables

In addition, we suggest the following instruments:

4. Analog Discovery 2

3.2 Software

The following programs must be installed on your development work station:

1. Microchip MPLAB X® v3.35 or higher
2. PLIB Peripheral Library
3. XC32 Cross Compiler
4. WaveForms 2015
5. PuTTY Terminal Emulation

4 Project Takeaways

1. Knowledge of a PC terminal emulation program.
2. How to develop a library of PIC32 software to provide bi-directional communications of single characters and strings of characters.
3. How to create a character LCD with a UART serial interface.

5 Fundamental Concepts

The Universal Asynchronous Receiver/Transmitter (UART) is an electronic device that converts parallel data to a serial data stream. Early microprocessors used independent integrated circuits to perform the conversion and frame the data stream with a start and stop bit. Currently, a vast majority of microprocessors have internal UART functionality, resulting in reduced system cost and complexity. The UART is capable of full-duplex operation, meaning simultaneously receiving and transmitting data. UARTs are generally constrained to a single peer-to-peer paired devices commonly called point-to-point communications.

Asynchronous communication is the transmission of data between two devices that are not synchronized with one another via a clocking mechanism or other technique. The term asynchronous implies the sender can initiate data transmission at any time, and the receiver must be ready to accept information when it arrives. The two devices must be operating at, or nearly equal to, the same clock frequency and are resynchronized by a START bit sent along with the data.

6 Problem Statement

Text messages will be sent to the UART serial port whenever a change in switch settings is detected and the action will be reported to a computer terminal. Whenever a text string is entered on the computer terminal, it will be displayed on the Basys MX3 LCD.

7 Background Information

Asynchronous communications is a serial data protocol that has been in use for many years. Normally eight bits of data are transmitted between handshaking characters to allow the clocks of transmitting and receiving devices to be synchronized. There are other less commonly used modes that can send 5, 6, or 7 bits of data. Each byte of data is framed by a start bit and a stop bit. A symbol is defined as a start, data, parity, or stop bit. It is common to define common to define communications speed as bits per second. The bit rate is defined as the inverse of the period of a unit symbol. Although the common standard bit rates are 50, 75, 110, 134, 150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, and 115200, communications is possible at any rate provided that the sender and receiver use the same rate. For most asynchronous communications, the term “baud” is commonly used interchangeably with the term “bit rate.”

8 Lab 4a

8.1 Requirements

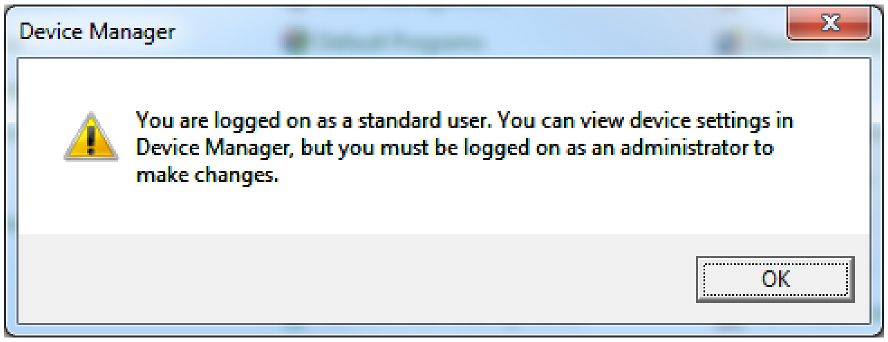
1. Communications will use the PC terminal emulation program for a bit rate of 19200, odd parity, 8 data bits, and one stop bit.
2. After receiving the line of text from the UART, you will clear the LCD before echoing the received string from the UART, starting at the leftmost character position on line 1 of the LCD.
3. Write a C program and call it lab4a.c. This program will contain the function *main* and process the serial text. Put the following tasks inside the while(1) loop:
4. Wait for line of text using the "getstr" function
5. Clear the LCD display and home cursor
6. Echo the string entered on the computer terminal to the LCD
7. Sense the state for the eight slide switches and convert the switch settings to a value, where SW7 is the most significant bit. Whenever any of the switches changes state, a text message is generated using the following format:
    a. "b b b b b b b b 0xhh ddd\n\r" where:
    b. "b" is 1 or zero representing the state of SW0 through SW7 with the SW7 bit leftmost.
    c. "hh" is the hexadecimal value of the binary encoded switches.
    d. "ddd" is the decimal equivalent of the hexadecimal number generated for part b.
    e. "\n" is the ASCII NEW LINE control character.
8. "\r" is the ASCII RETURN control character.
9. Send the text string composed in requirement 5 to the UART so it will be displayed on the terminal monitor as a single line of text.

8.2 Design Phase

1. Develop a data flow diagram for the software components needed for the requirements of Lab 4a.
2. Schematic diagrams: Provide a block diagram of the equipment used for Lab 4a.
3. Flow diagrams: Provide a complete software control flow diagram for Lab 4a.

8.3 Construction Phase

1. Connect the Basys MX3 UART USB port to one of the work station's USB ports.
2. If the workstation is running a Windows OS, open the "Control Panel" followed by opening the "Device Manager" window. If you do not have administrator privileges, you will see the window shown in Fig. 8.1. Click on the OK box to continue. Administrative privileges are not required to view the settings.



You are logged on as a standard user. You can view device settings in Device Manager, but you must be logged on as an administrator to make changes.

Figure 8.1. The device manager window showing you do not have administrator privileges.

3. Expand the tab called Ports (COM & LPT) as shown in Fig. 8.2.

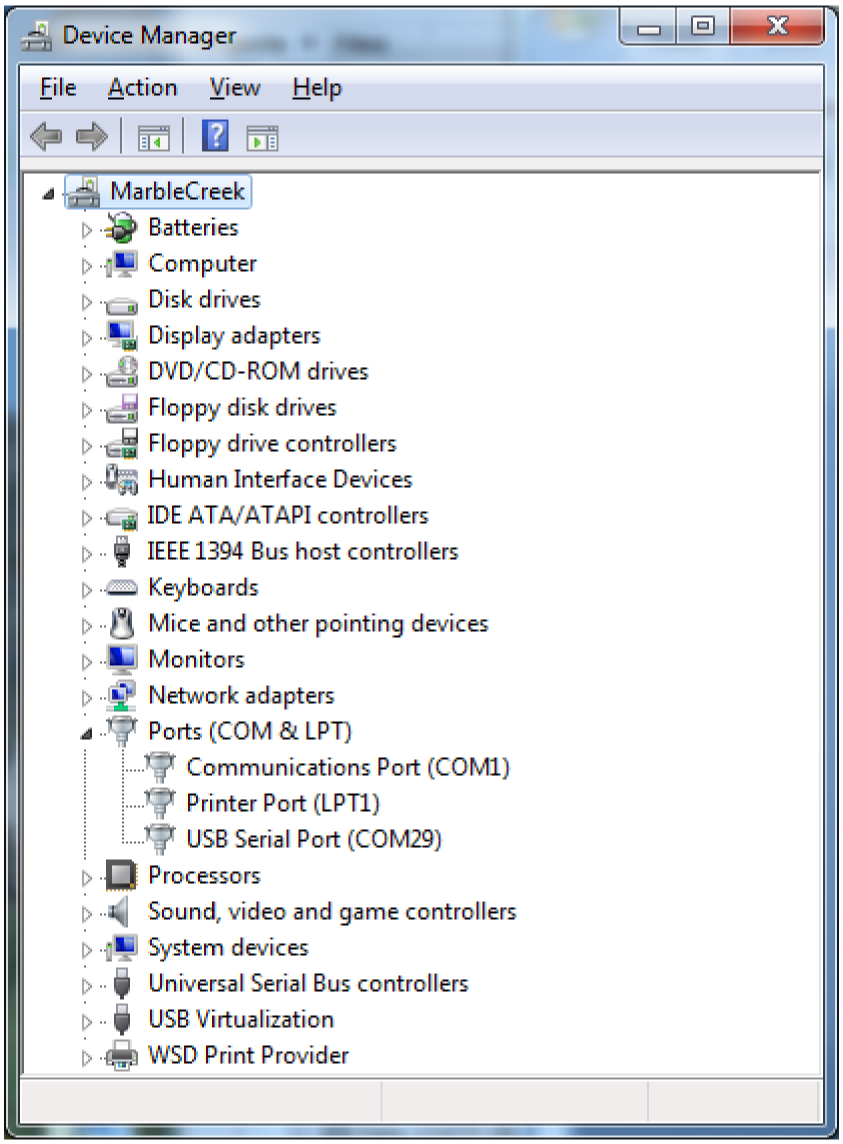


Figure 8.2. PC device manager window.

4. Note the USB Serial Port COM assignment.
5. Open the terminal emulation program on your workstation. Configure the terminal program for 19200 BAUD and ODD parity. The screen shown in Fig. A.4 of Appendix 1 is for the PuTTY terminal emulation program.
6. Launch a new Microchip MPLAB X project called LAB4b. Add the *config_bits.h* file to the project.
7. Develop the following UART interface functions: Note that the numeral "4" in the function names indicate that the Basys MX3 uses the PIC32MX370 UART 4. All receive functions are to be non-blocking.
    a. uart4Init(); // 19200 Baud, ODD parity
    b. (int) ch = uart4Getc(); // ch = -1 if no data has been received
    c. (int) len = uart4Gets(char *str); // len = -1 if no data has been received
    d. uart4Puc(char ch);
    e. uart4Puts(char *str);
8. Develop the PIC32 application that meets the requirements in section 8.1.

8.4 Testing

1. After completing the development, run the application project to verify that the LCD is initialized correctly. I generally display an initial message on the LCD for one second that declares the LCD is functional.
2. Toggle the slide switches to verify that the terminal screen displays text similar to Figure A.3.
3. Enter a series of text strings that verify the following operations:
    a. A text string containing between 1 and 16 characters displays only on the first LCD line.
    b. A text string containing between 17 and 32 characters displays on both the first and second LCD lines as shown in Fig. A.2.
    c. A text string containing between 17 and 32 characters displays on both the first and second as well as wrapping back around to the first LCD line.
4. Connect the Analog Discovery 2 to the Basys MX3 board.
    a. Configure the Logic window to display signals DIO 8 which is the UART RX pin. Measure the time from the beginning of the start character to the beginning of the stop character.
    b. Configure the Logic window to display signals DIO 0 through DIO 3. Label the signals as follows:

i.	DIO 0	DB0
ii.	DIO 1	EN
iii.	DIO 2	RW
iv.	DIO 3	RS

    c. Capture a single character display as shown in Fig. 8.3.

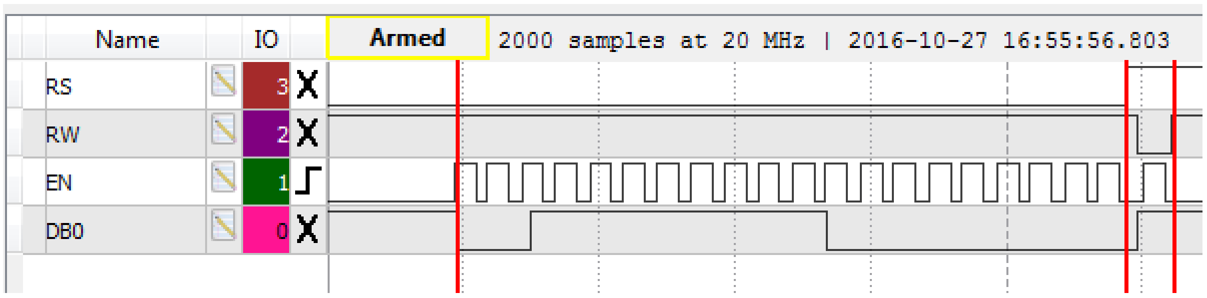
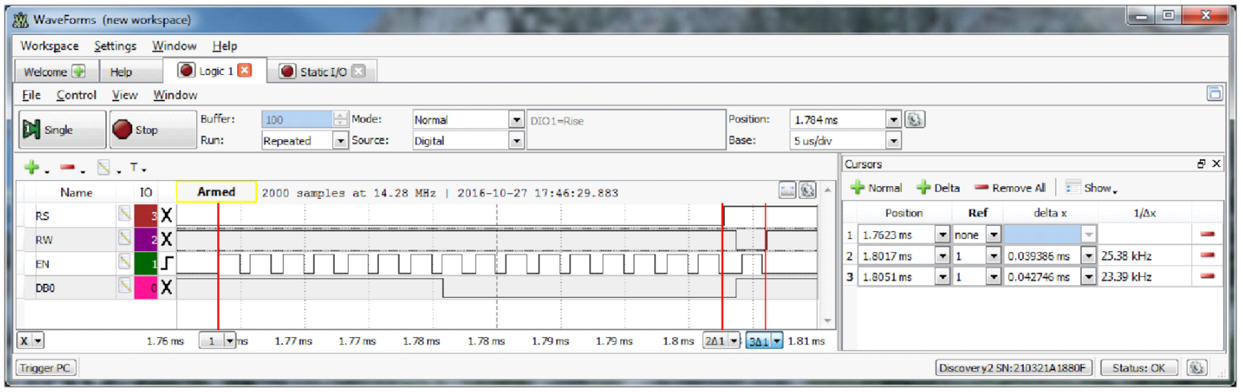


Figure 8.3. Handshaking pins and data bit zero for single character write to the LCD.

    d. Measure the time the LCD reported being busy.
    e. Measure the time required to output a single character.
    f. Measure the time required to display a string of 20 characters.

9 Questions

1. What is the effective data rate of the UART? (Remember to include the period of the stop signal that cannot be measured in testing step 4a.)
2. Based on the data collected in step 4 parts d and e of the testing procedure, what is the effective character display rate in characters per second?

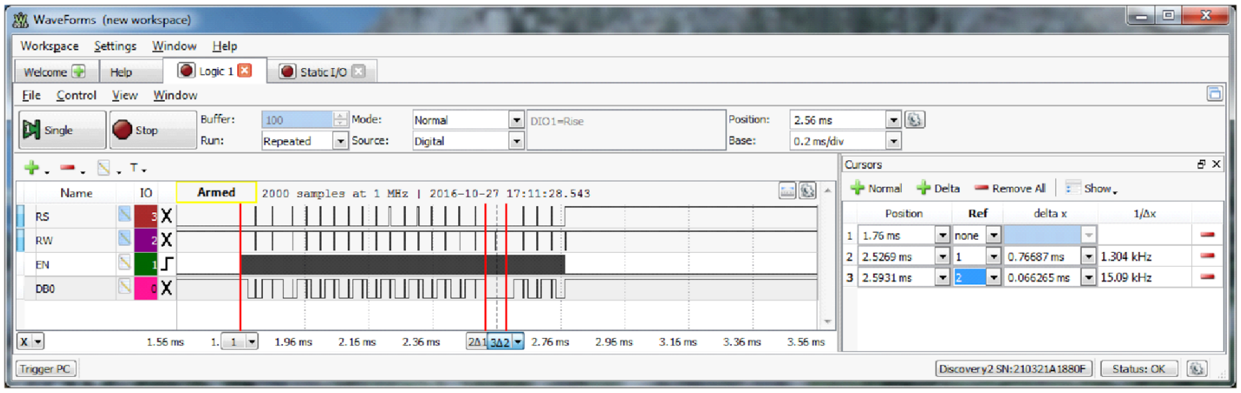


Buffer: 100 Mode: Normal Run: Repeated Source: Digital DIO1=Rise Position: 1.784 ms Base: 5 us/div

Armed 2000 samples at 14.28 MHz | 2016-10-27 17:46:29.883

	Position	Ref	delta x	1/Δx
1	1.7623 ms	none		
2	1.8017 ms	1	0.039386 ms	25.38 kHz
3	1.8051 ms	1	0.042746 ms	23.39 kHz

3. Based on the data collected in step 4, part f, how much does moving the cursor from line 1 to line 2 slow down the LCD character display rate? Justify your answer.



Buffer: 100 Mode: Normal Run: Repeated Source: Digital DIO1=Rise Position: 2.56 ms Base: 0.2 ms/div

Armed 2000 samples at 1 MHz | 2016-10-27 17:11:28.543

	Position	Ref	delta x	1/Δx
1	1.76 ms	none		
2	2.5269 ms	1	0.76687 ms	1.304 kHz
3	2.5931 ms	2	0.066265 ms	15.09 kHz

10 References

1. PIC32MX330/350/370/430/450/470 Family Data Sheet
2. “Using the USART in Asynchronous Mode”, http://ww1.microchip.com/downloads/en/DeviceDoc/usart.pdf
3. “Asynchronous Communications with the PICmicro® USART”, http://ww1.microchip.com/downloads/en/AppNotes/00774a.pdf
4. RS-232, RS-422, RS-423, RS-485 Asynchronous communications
5. HD44780U LCD data sheet, https://www.sparkfun.com/datasheets/LCD/HD44780.pdf

Appendix A: Basys MX3 Schematic Drawings

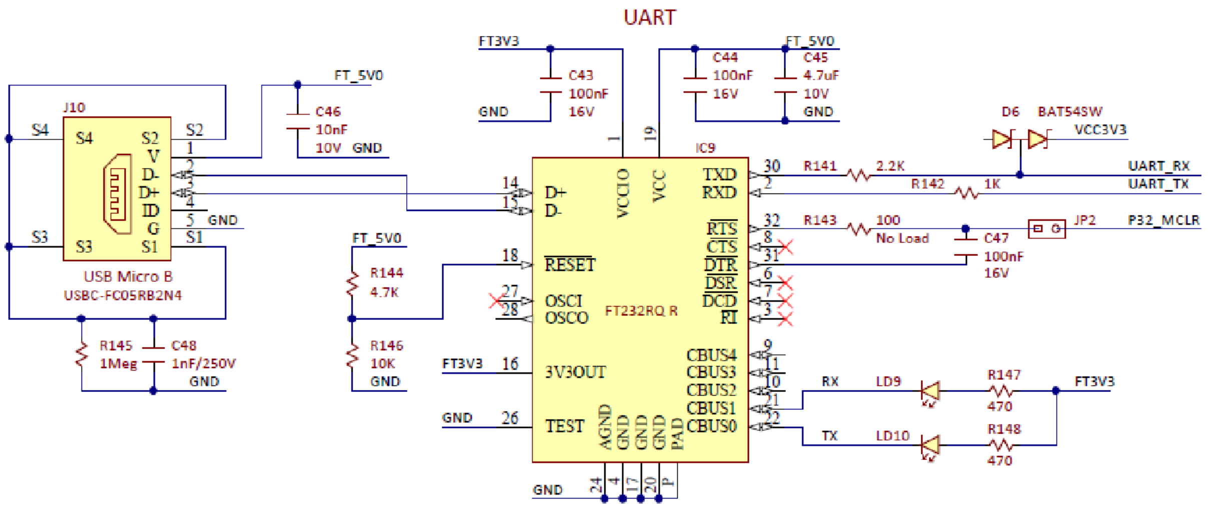


Figure A.1. PIC32MX370 to FT232RQR IC schematic diagram.

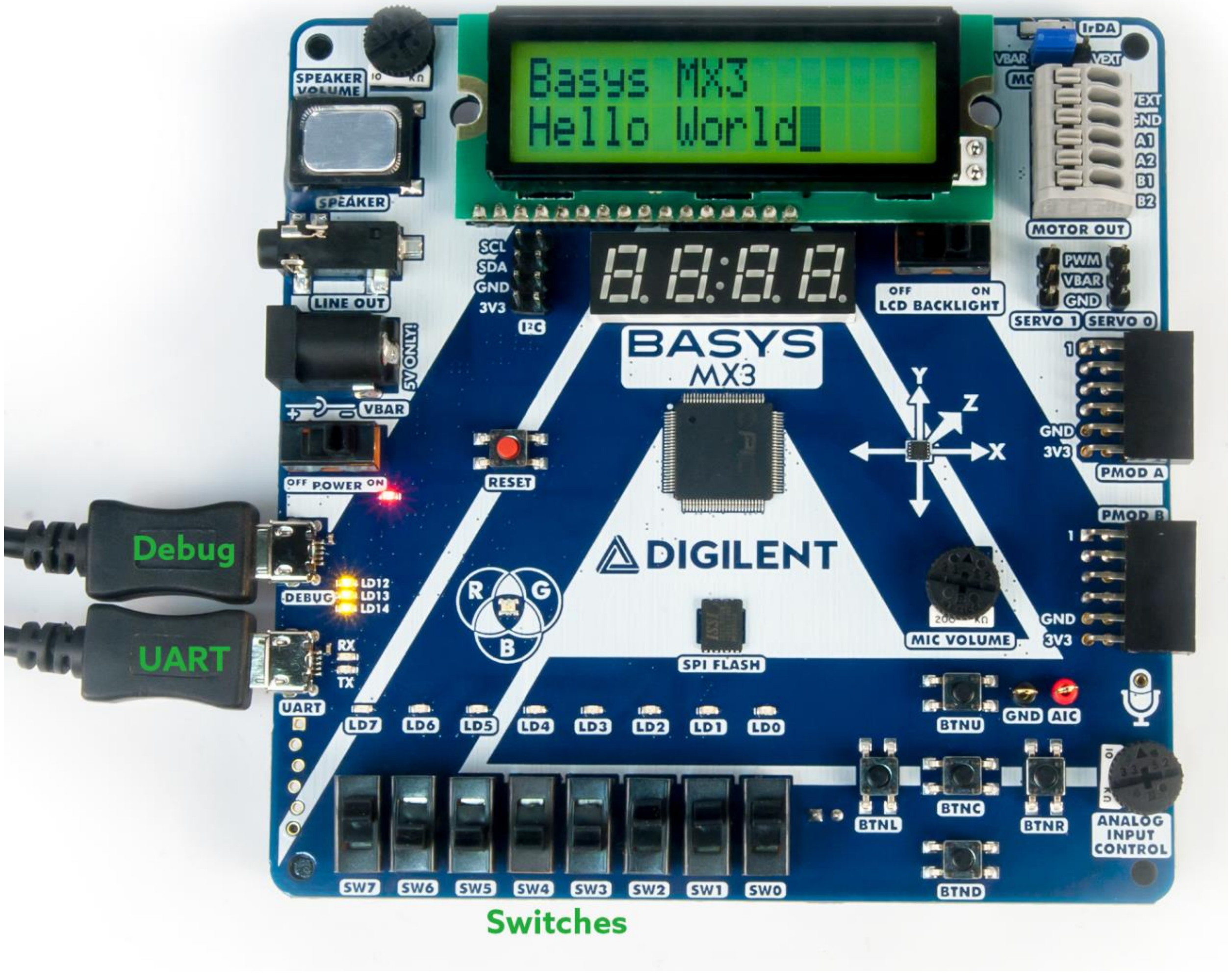
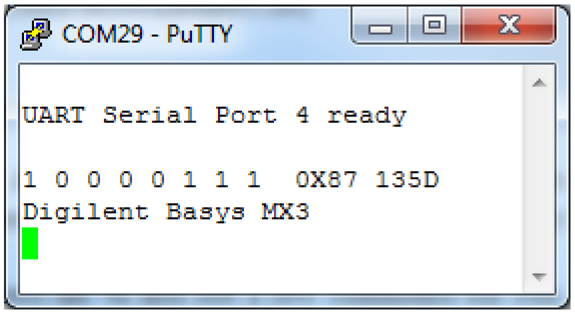


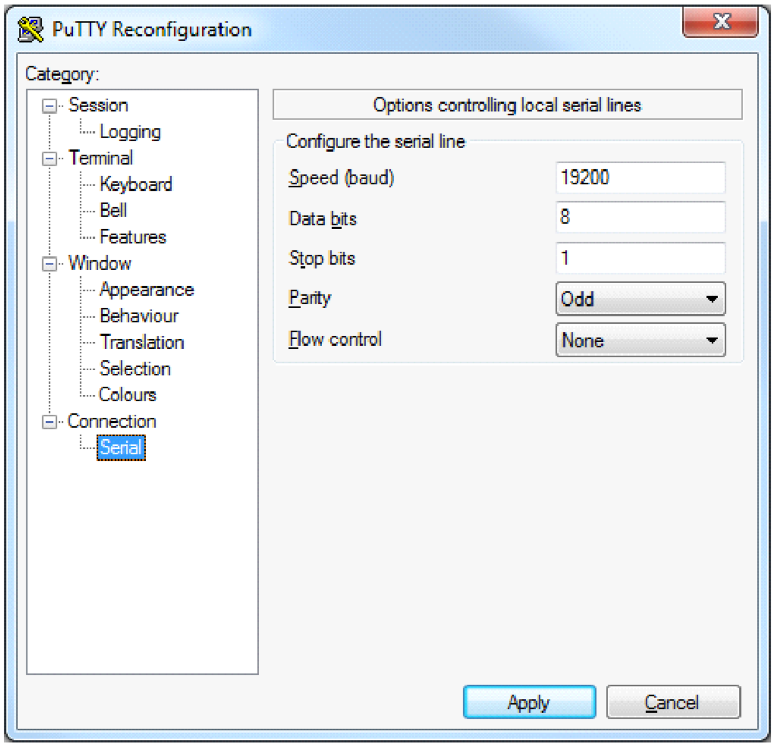
Figure A.2. UART USB connector on the Basys MX3.



```
UART Serial Port 4 ready

1 0 0 0 0 1 1 1  0X87 135D
Digilent Basys MX3
```

Figure A.3. PuTTy screen shot generating LCD display.

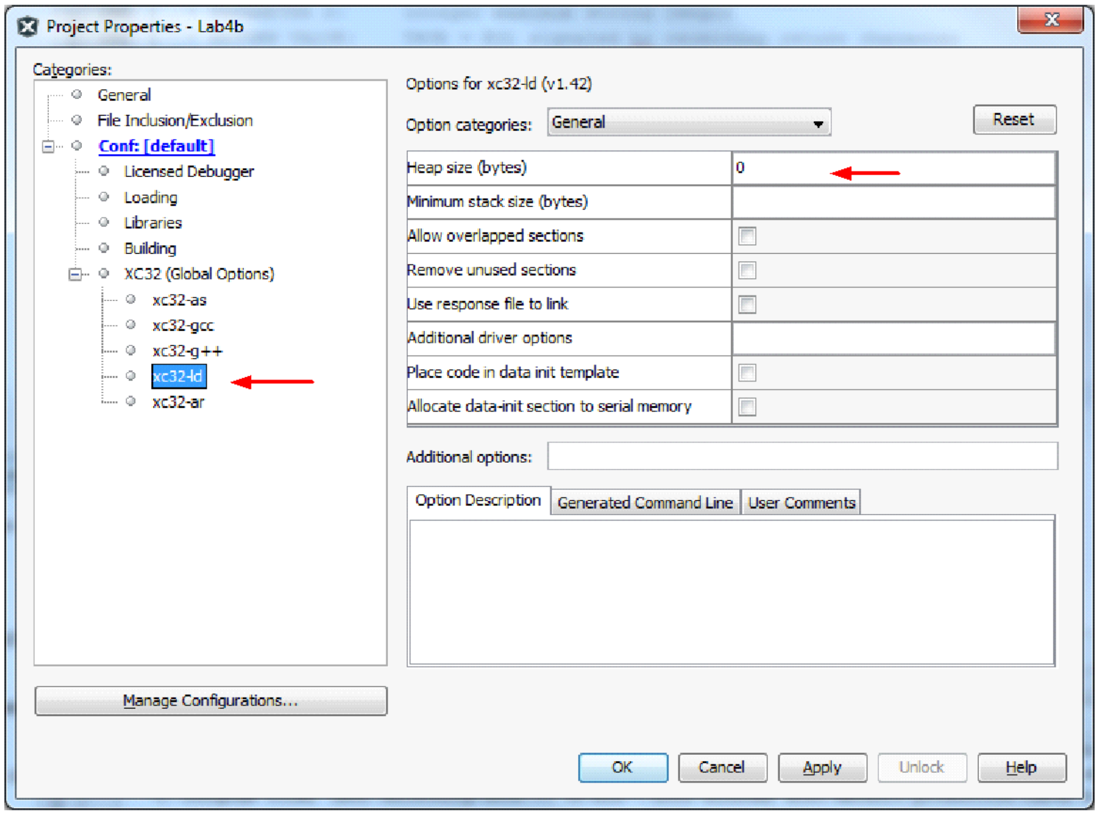


Speed (baud) 19200
Data bits 8
Stop bits 1
Parity Odd
Flow control None

Figure A.4. PuTTY screen shot of serial configuration for 19200 BAUD and ODD parity.

Appendix B: Allocating a Heap in MPLAB X

If when compiling your project you see an error like "ld.exe Error: A heap is required, but has not been specified," this is because you need to specify a heap size by setting “Run” ->“Set Project Configuration” -> “Customize…”. Go to the “xc32-ld” category (under “XC32 (Global Options)”) -> “Heap size (bytes)” to “0” The configuration window should look like Fig. B.1. Click on the “Apply” button followed by clicking on the “OK” button. See http://microchip.wikidot.com/mplabx:creating-a-heap.



Options for xc32-ld (v1.42)

Option categories: General

Heap size (bytes)	0
Minimum stack size (bytes)	
Allow overlapped sections	☐
Remove unused sections	☐
Use response file to link	☐
Additional driver options	
Place code in data init template	☐
Allocate data-init section to serial memory	☐

Additional options:

Figure B.1. Allocating Heap size.