Laboratory 04: Software UART Transmitter and Receiver - Saleae Logic Analyzer and Tera Term Terminal Emulator

Prof. Ken Short

Overview

- Asynchronous communication is very common in embedded system design.
- Writing your own C functions to implement a serial transmitter and receiver clarifies the basic concepts of asynchronous serial transfer. It will also give you some practice in bit manipulation and software delays in C.
- You will also learn to use Saleae's 16-channel logic analyzer's protocol aware capabilities to view asynchronous serial data decoded based on a specified serial protocol.
- A terminal emulator software application running on a PC allows you to send and receive ASCII information in the asynchronous serial protocol and see this information displayed on the PC's monitor.

Laboratory 04 Laboratory Tasks

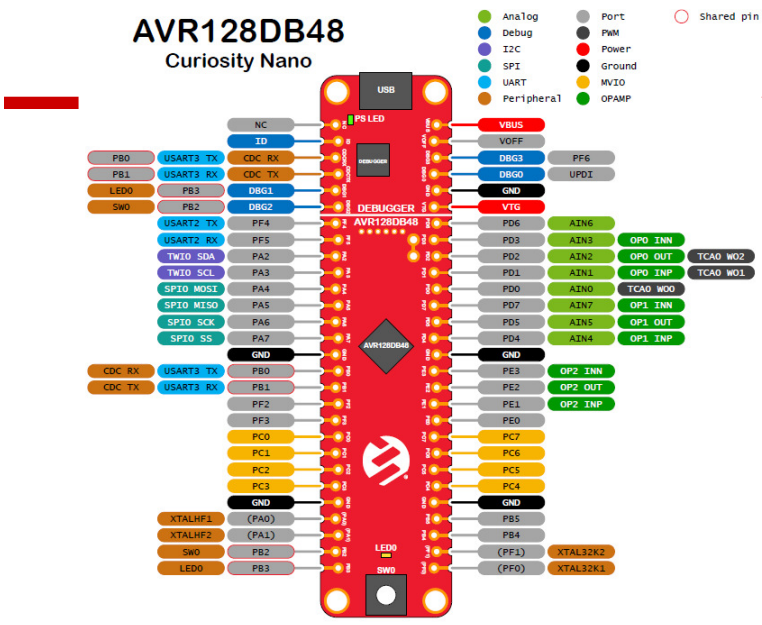
- Design Task 1: Software UART Transmitter Function
- Design Task 2: Software UART Receiver Function
- Design Task 3: Software UART Receiver Interrupt Service Routine
- Design Task 4: Interrupt Echo Program
- Design Task 5: Message Relay Program

Design Task 1: Software UART Transmitter Function

- You must write a program named asynch_sw_send that sends ASCII characters at any of the following three baud rates: 4800, 9600, or 19200 baud.
- This program must use PB0 as the TX pin for your software transmitter. In the laboratory, this program will be used to test your UART_sw_write function.
- The characters transmitted will be observed using the oscilloscope, Saleae logic analyzer, and Tera Term. You will use a loop in your program to continuously send the same character with a 1 ms delay between the characters.

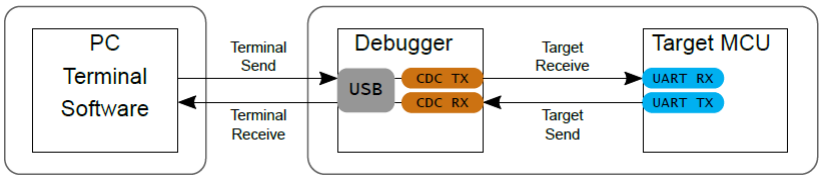
Curiosity Board

- You are to use pin PB0 for your TX signal. PB0 appears on two pins of the Curiosity Nano. It appears at pin 16 as PB0 and on pin 4 as CDC TX.
- Use pin 16 when connecting to the oscilloscope and Saleae Logic Analyzer.
- The same signal also appears on pin 4 and is connected on the Curiosity board to a virtual serial port (CDC) that provides a general purpose serial bridge between the host PC via the USB cable and the target microcontroller on the Curiosity Nano.



Curiosity Nano Virtual Serial Port (CDC)

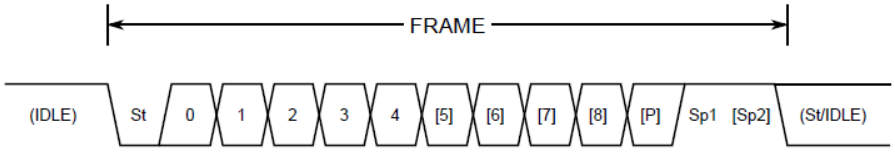
- The virtual serial port (CDC) is a general purpose serial bridge between a host PC and the target microcontroller on Curiosity Nano.
- The on-board debugger implements a composite USB device that includes a standard Communications Device Class (CDC) interface, which appears to the host as a virtual serial port. The CDC can be used to stream arbitrary data in both directions between the host computer and the target.



2/4/2022 © Copyright Kenneth Short 2022 7

Asynchronous Frame Format

1 Start bit
5, 6, 7, 8, or 9 data bits
No, even, or odd Parity bit
1 or 2 Stop bits



St	Start bit, always low
(n)	Data bits (0 to 8)
P	Parity bit, may be odd or even
Sp	Stop bit, always high
IDLE	No transfer on the communication line (RxD or TxD). The Idle state is always high.

2/4/2022 © Copyright Kenneth Short 2022 8

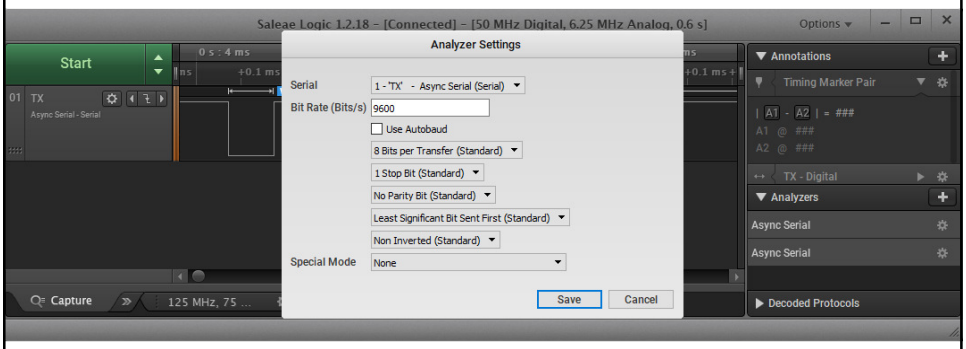
ASCII Table

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	`	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 XOFF	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	del

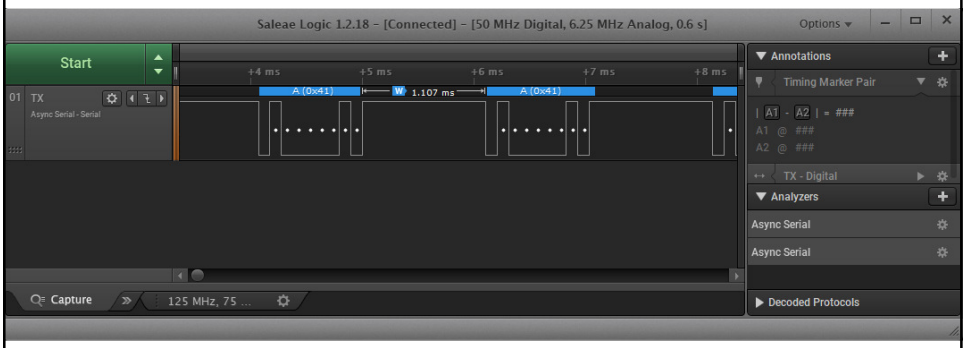
Setting Up Saleae Logic Analyzer to View Transmitted Frame

- Using the Saleae Logic Analyzer requires some setup to select the protocol to be decoded and set the format of the frames within the protocol.

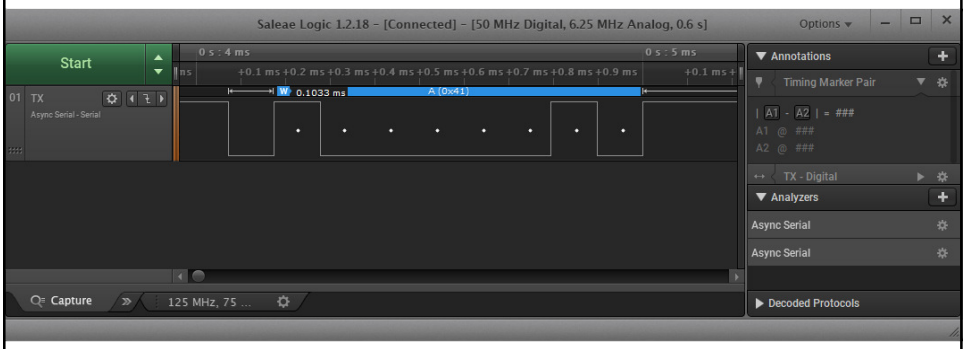
Selecting and Setting the Async Serial Protocol Analyzer – 9600 baud 8N1



Running asynch_sw_send at 9600 Baud

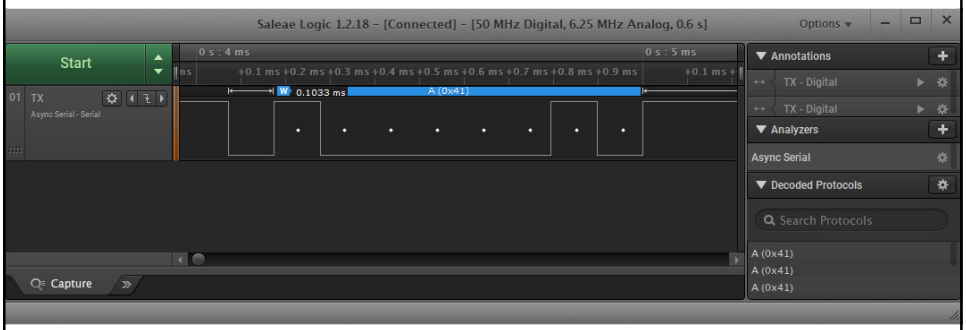


Character 'A' – Bits L to R: Start, Seven Data Bits (lsb first), and Stop Bit



White dots indicate when Saleae is sampling the transmitted data that it is receiving.

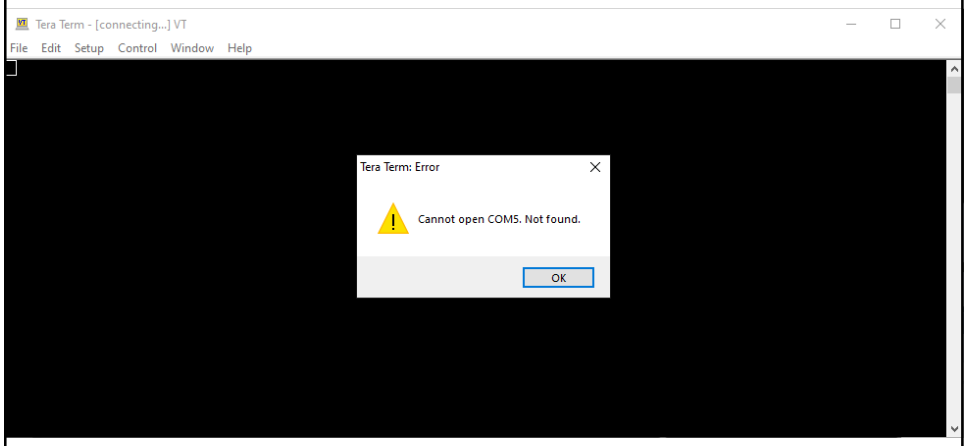
Decoded Protocols Window - Sequence of Decoded Frames (bottom right)



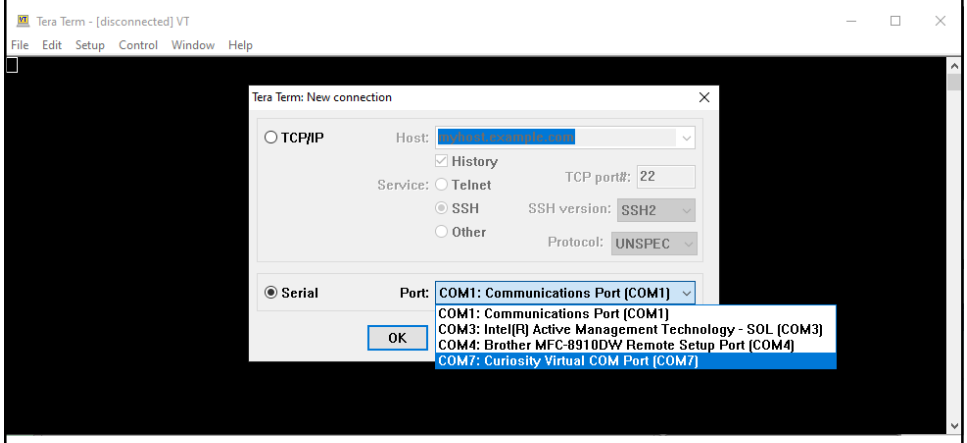
Setting Up Tera Term to View Transmitted Characters

- The Tera Term must be configured to select the COM port on the PC it will communicate through.
- It must also be configured for the asynchronous frame format.

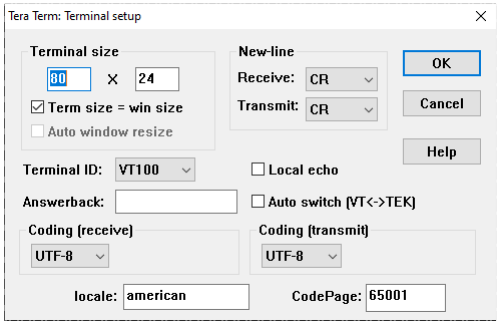
When You Click on Tera Term Icon You May Get This Message



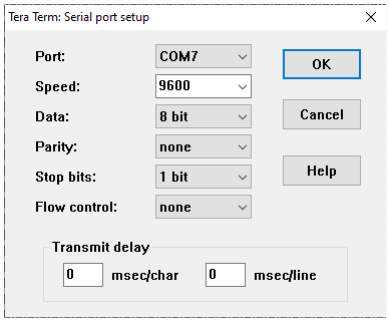
You Must Select File > New Connection to Select the Curiosity Virtual COM Port



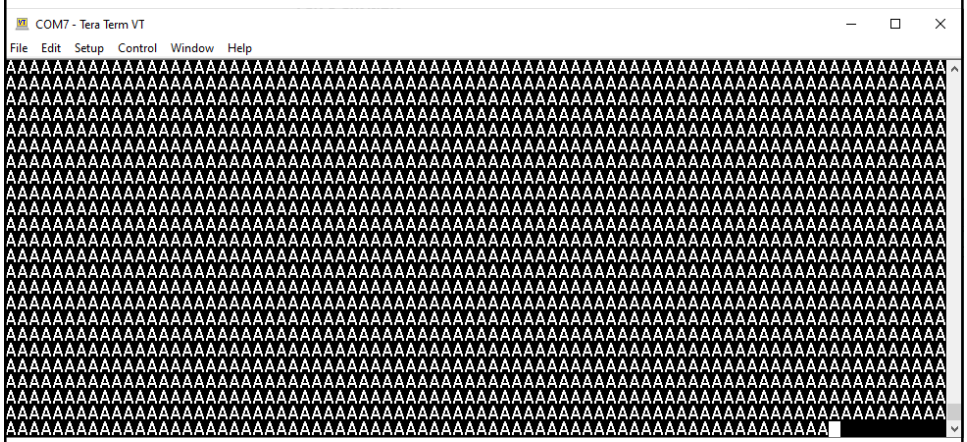
Select Setup > Terminal and Verify the Following Setting Are Selected



Select Setup > Serial Port to Specify The Baud Rate and Frame Format



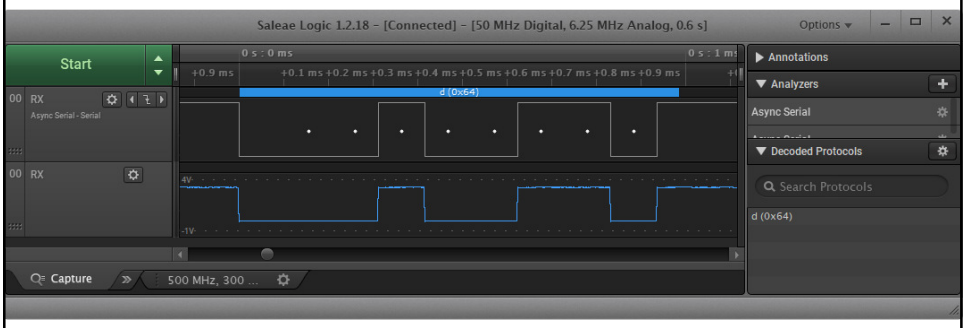
Character 'A' Being Streamed From AVR128DB48 to Tera Term



Design Task 2: Software UART Receiver Function

- You must write a function to read an asynchronous character with a format of 8N1 and baud rates of either 4800, 9600, 19200 baud.
- The RX pin is PB1 Curiosity Nano pin 17 and pin 3 (for the CDC).
- This software receiver routine must use polling to detect the start of a character frame and must implement a false start check at the middle of the start bit. This check determines if the signal is still 0. If not, a false start is assumed and the reception of the character is terminated and polling resumed to detect the next start bit.

Reading a 'd' Sent From Tera Term



Design Task 3: Software UART Receiver Interrupt Service Routine

- The polling receive function UART_sw_read must be modified to be an interrupt service routine using the pin change interrupt of pin PB1.

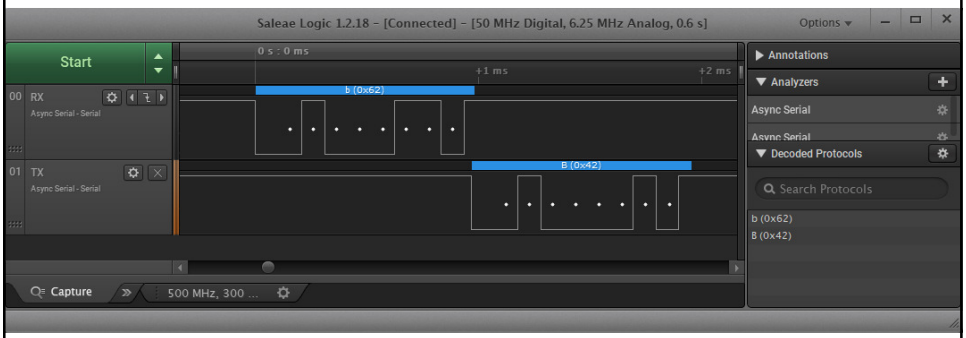
Pin Change Interrupt Code

- Include the header file <avr/interrupt.h>
- Name the interrupt service routine:
  - ISR (PORTB_PORT_vect) { }
- Enable the interrupt locally in PORTB.PIN1CTRL
- Enable the interrupts globally:
  - sei();
- In the while (1) statement in main(), put only the single instruction:
  - asm volatile ("nop");

Design Task 4: Interrupt Echo Program

- You must write a program named interrupt_echo that combines the code from Task 1 and Task 3 to create a program that uses an interrupt to receive an alphabetic character from Tera Term and echo the character back to the Tera Term with its case changed.
- So, if you send a lower case 'a' it sends back an upper case 'A'. The inputs are assumed to be limited to alphabetic characters.

Receiving a Lowercase 'b' and Transmitting and Uppercase 'B'



Design Task 5: Message Relay Program

- You must modify your program from Design Task 4 to create a program named interrupt_echo_line that uses interrupts to receive a line of ASCII characters from the Tera Term.
- The characters received must be buffered in an 80 character array until a carriage return 'CR' control character (0x0D) is received.
- Once a carriage return character is received, the program must send (echo or relay) the entire line back to the Tera Term. After a line is echoed back, the Tera Term's cursor should be moved to the beginning of the next line.