SFU

ENSC 350

Digital Systems Design

Final Project (UART) Report

LA03: Group 4

Nicholas Mah 301427489

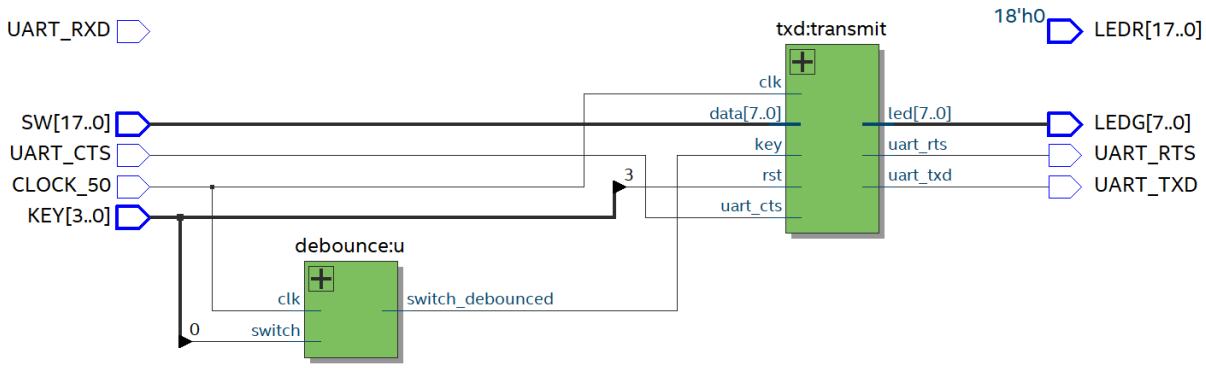
(nma66@sfu.ca)

Ritchie Kumar 301389811

(ritchiek@sfu.ca)

Implementation

For our final project, we implemented a transmitter that sends data from the FPGA to a computer with access to the open-source terminal application Tera Term (receiver). The transmitter *(txd.vhd)* consists of the following signals: clk, reset, key, data, uart_cts, uart_rts

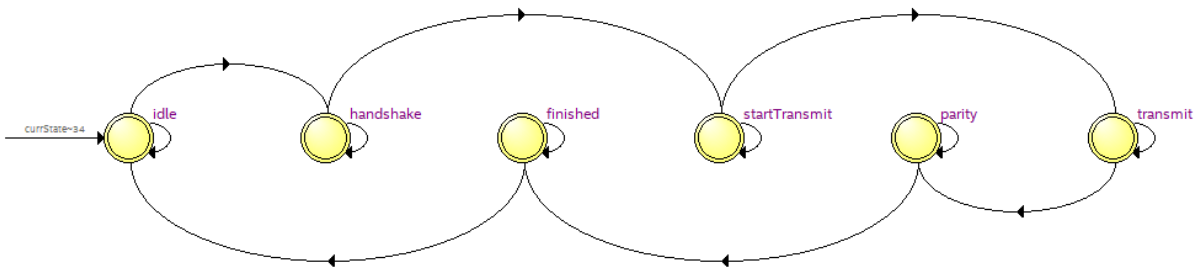


Transmitter System

The transmitter of our system works as part of the Universal Asynchronous Receiver/Transmitter (UART) communication protocol. To do so, we must first determine the Baud Rate, the rate at which bits are transferred in the communication channel. The following formula is used to determine the clock cycles per bit.

CC per bit = Clock Frequency / baud rate

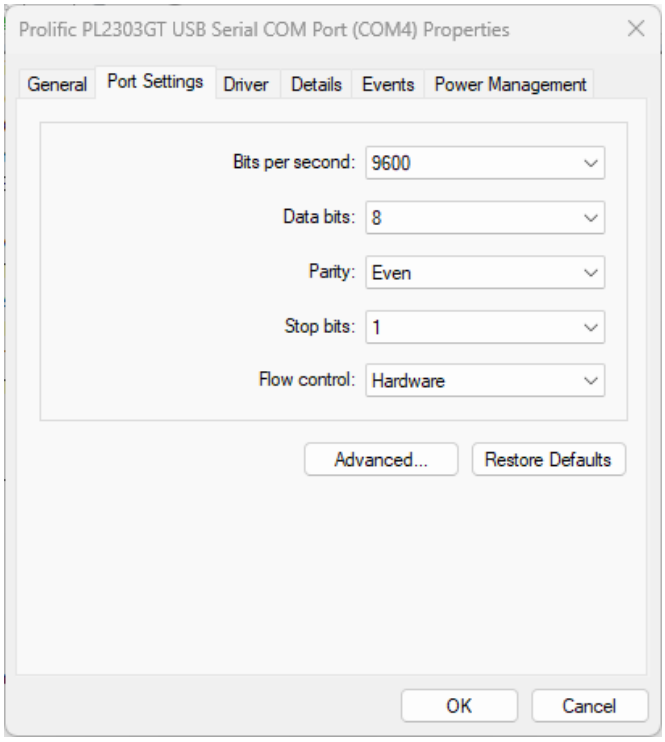
We used a standard baud rate of 9600 bits/s, and the FPGA clock frequency is 50 MHz. This approximates the clock cycles per bit to ~5208.




idle
Idle state where the transmitter checks for a key press to move to handshake
Asserts RTS if key is pressed
handshake
Checks for CTS to be asserted to move to 'startTransmit'
startTransmit
Bit changes occur according to the baudrate generator set to a baudrate of 9600 for this project
Sends a '0' to initiate the data packet
transmit
Bit changes occur according to the baudrate generator set to a baudrate of 9600 for this project
The data bits are sent one by one from LSB to MSB
Moves to 'parity' state after last bit has been sent and calculates parity bit
parity
Bit changes occur according to the baudrate generator set to a baudrate of 9600 for this project
Sends parity bit and moves to 'finished' state
FInished
Bit changes occur according to the baudrate generator set to a baudrate of 9600 for this project
Sends a '1' to end the data packet
Returns to 'idle' state after sending bit

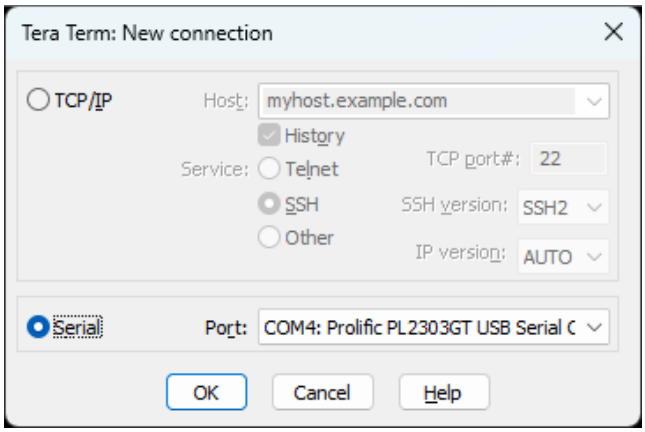
Test Instruction Manual

1. On a computer download and install Teraterm using default settings, a link to Teraterm is found below:
    a. https://github.com/TeraTermProject/teraterm/releases/download/v5.2/teraterm-5.2.exe
2. Connect a USB to Serial cable from the computer with Teraterm and the FPGA board.
3. Open ‘Device Manager’ >> ‘Ports’ >> Select your USB to Serial cable COM port and navigate to the ‘Port Settings’ tab and enter the following settings

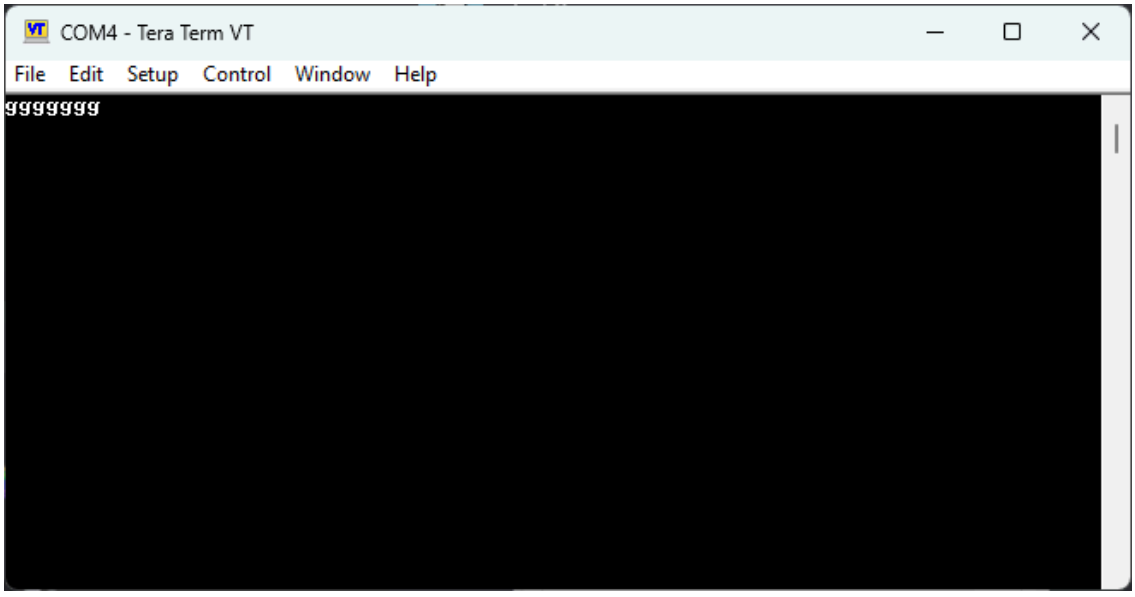


4. Unplug the cable from the computer and plug back in and check that the ‘Port Settings’ have been saved

5. Open a Teraterm window and select the ‘Serial’ radio button as shown below. Check that the COM port matches the COM port number seen in ‘Device Manager’. Click ‘OK’.



6. A blank terminal window should appear
7. Download the project zip file >> unzip the folder >> open ‘finalProject’ folder
8. Open ‘finalProject.qpf’ to open the Quartus project
9. Compile the project and load onto the FPGA board via the programmer
10. KEY(0) is used to send the 8 bit binary number currently represented by switches 7 to 0, or SW(7 downto 0). KEY(3) is an asynchronous reset. An ASCII table can be found in the Appendix of this report to verify the translation from binary to ASCII characters that are displayed in the Teraterm window. An example output is shown below.



11. The switches can be changed to represent a different character and sent to the TeraTerm terminal
12. Have fun :))

Test Bench

To run the testbench use ‘tx_tb’ and ‘txd.vhd’ in ModelSim and run the simulation for 15 ms. The transcript output should match below.

```
#    Time: 7488120 ns  Iteration: 0  Instance: /tx_tb
# ** Note: Test 7: data in = 11100011, parity bit = 1
#    Time: 7488120 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 7 Passed ================================
#    Time: 8632120 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 8 ===============================
#    Time: 8736140 ns  Iteration: 0  Instance: /tx_tb
# ** Note: Test 8: data in = 01010100, parity bit = 1
#    Time: 8736140 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 8 Passed ================================
#    Time: 9880140 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 9 ===============================
#    Time: 9984160 ns  Iteration: 0  Instance: /tx_tb
# ** Note: Test 9: data in = 10001110, parity bit = 0
#    Time: 9984160 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 9 Passed ================================
#    Time: 11128160 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 10 ===============================
#    Time: 11232180 ns  Iteration: 0  Instance: /tx_tb
# ** Note: Test 10: data in = 11110000, parity bit = 0
#    Time: 11232180 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== Test 10 Passed ================================
#    Time: 12480180 ns  Iteration: 0  Instance: /tx_tb
# ** Note: ========================== All Tests Passed ================================
#    Time: 12480180 ns  Iteration: 0  Instance: /tx_tb
```

Appendix

Note: Use the Hex column, second from the left to obtain the binary representation

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com