This project is done on Alinx Kit :AXU2CGB-E having FPGA XCZU2CG-SFVC784-1-e.This board has differential PL clock .So to make single clock from differential clock we used Buffer(Diff_CLk code).

Precautions: Set your board clock as per your board settings.If your board has differential clocks then buffer will not reduce that clock but only gives you a single clock with same frequency.

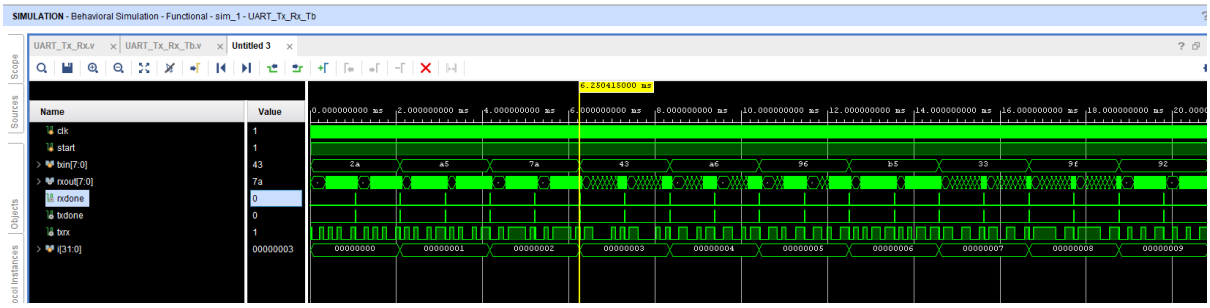
In version 1,2 and 3 We were only working with simulation and PMod verfifcations so we didn't find any error while keeping clock to 100MHz although our board clock was of 200MHz.Whiel communicating with PC we have to set baud rate according to 200MHz clock.

V1

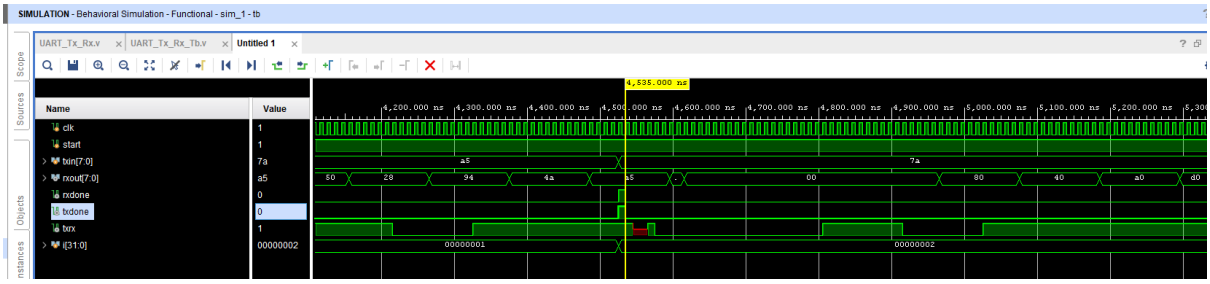
This code is same as in tutorial.

Clock Value is set to 100MHz becoz differential clock output is 100MHz.Baud Rate is 9600.Wait value will be 10416.

Simulation is done with its own testbench.Output is correct somehow



Potential Issues.

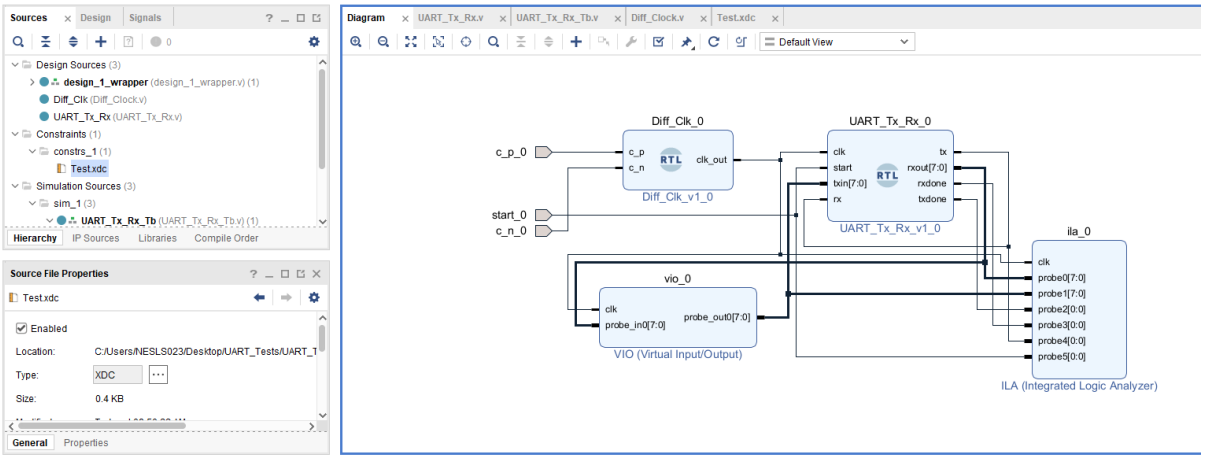


1.Txand Rx is going high at the same time.Reception is done in center of b

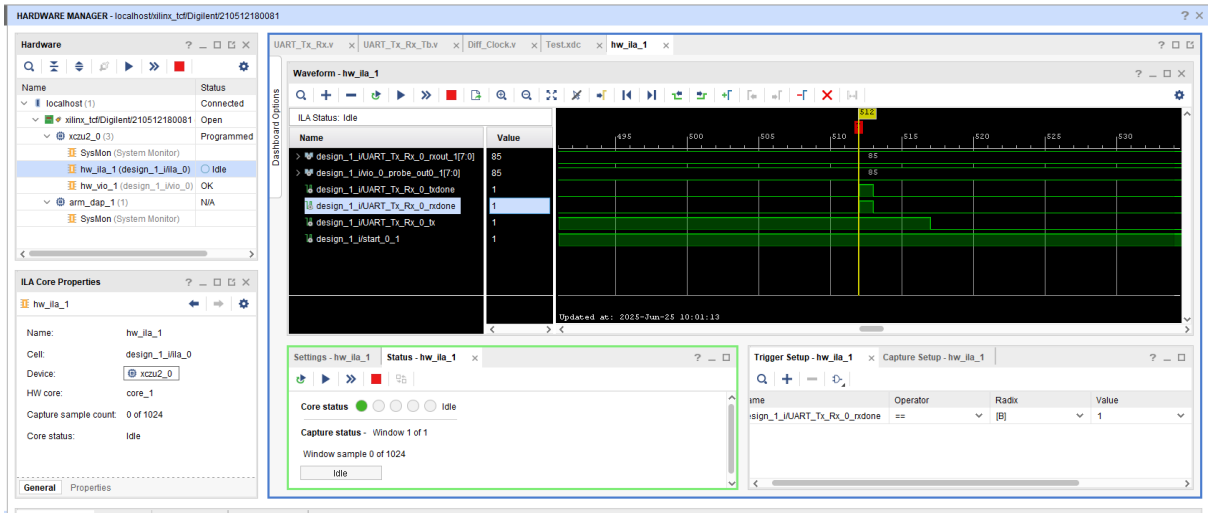
2.Rx out register become full before making Rx_Done high.

3.Continously updating Rx Register when we have only 1 transmitted data.(not an issue,but bug)

4.txrx becomes unknown at some points bcz it is wire and reg at same time.



Internal LoopBack Block Diagram

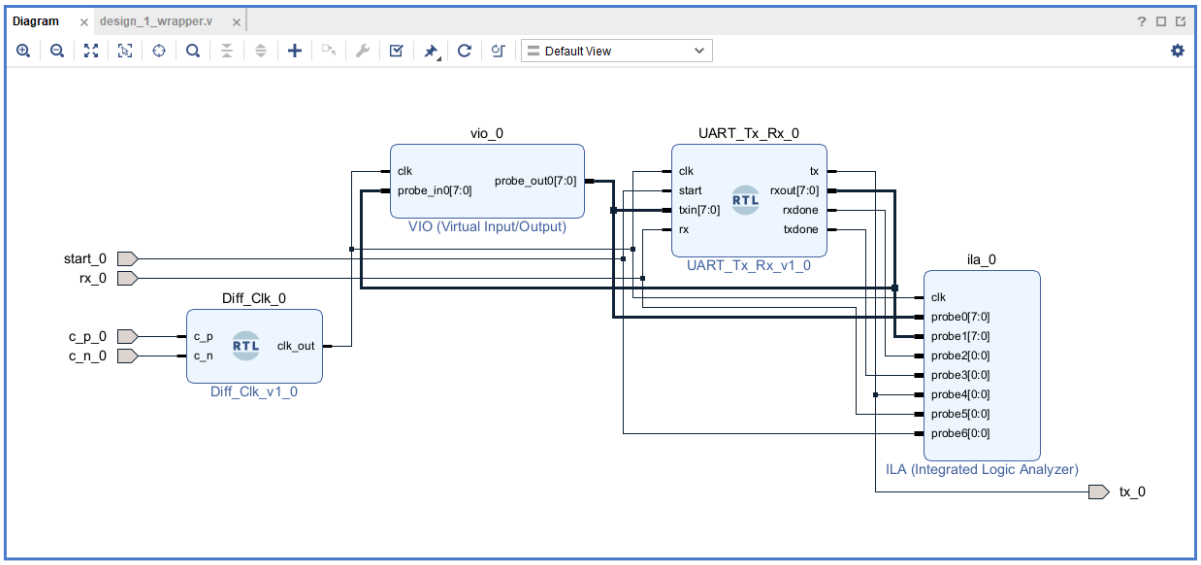


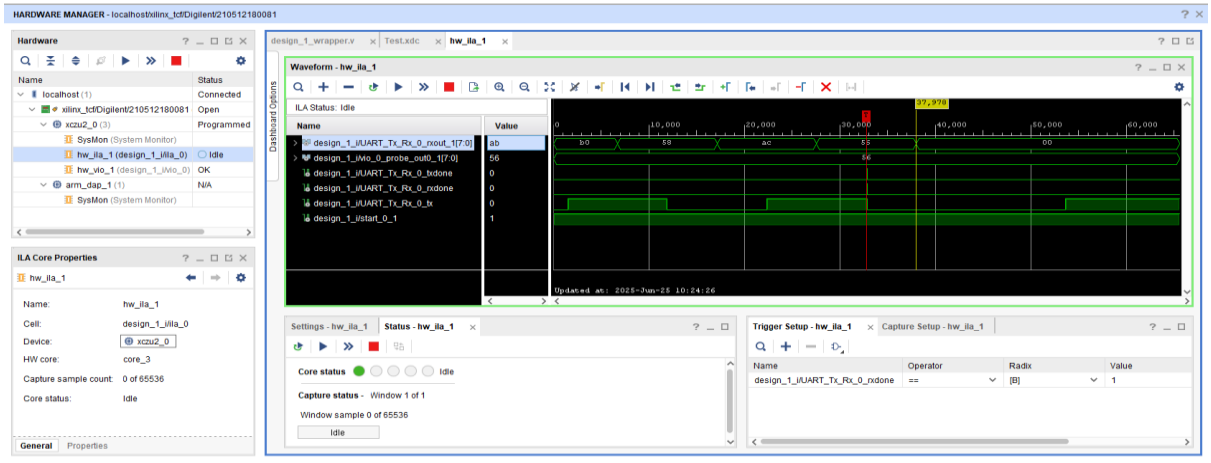
Hardware

V1 Ended

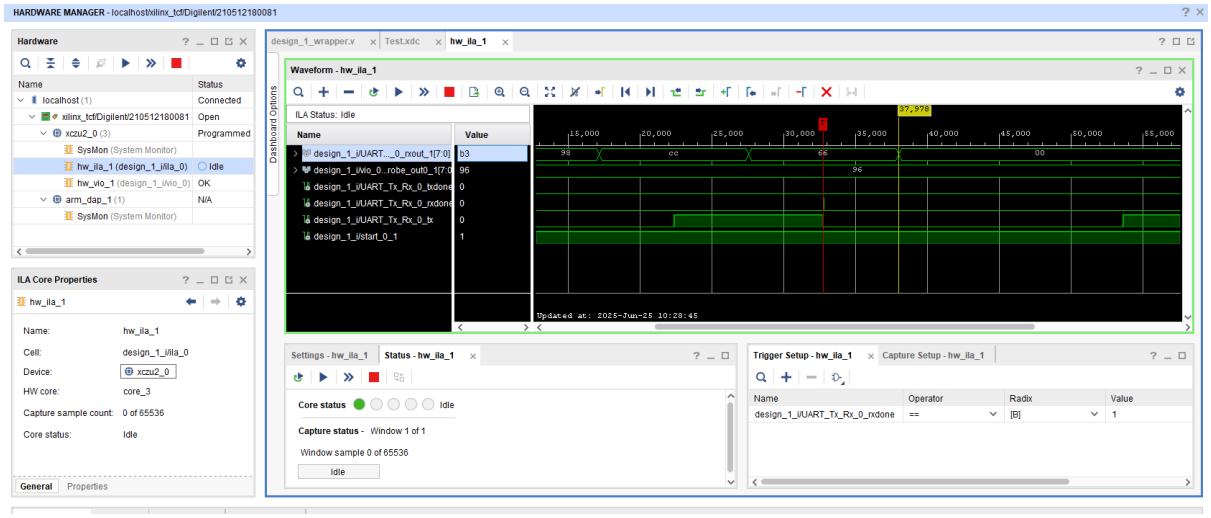
V2

Now Block Diagram will be changed to Test this code with PMOD Tx_Rx.

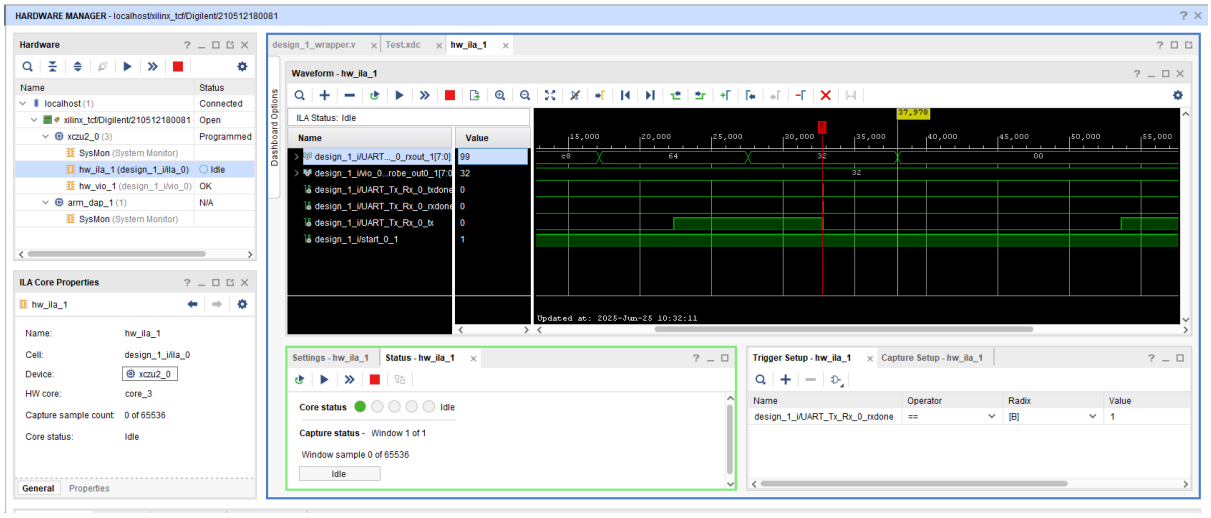




When pins are connected



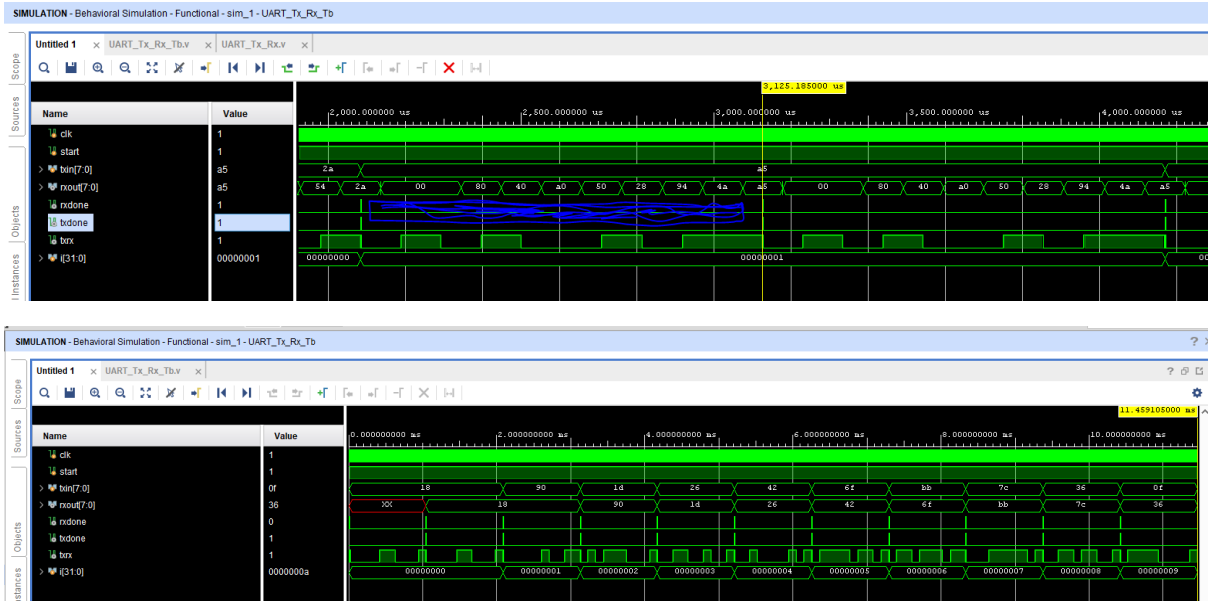
When Rx pin is disconnected,it waits for Rx_Done Trigger to come.When connected back it shows above behaviour and makes Rx_Done high when actual transmitted data was received(96).After that it shows correct transmission and reception as shown below.



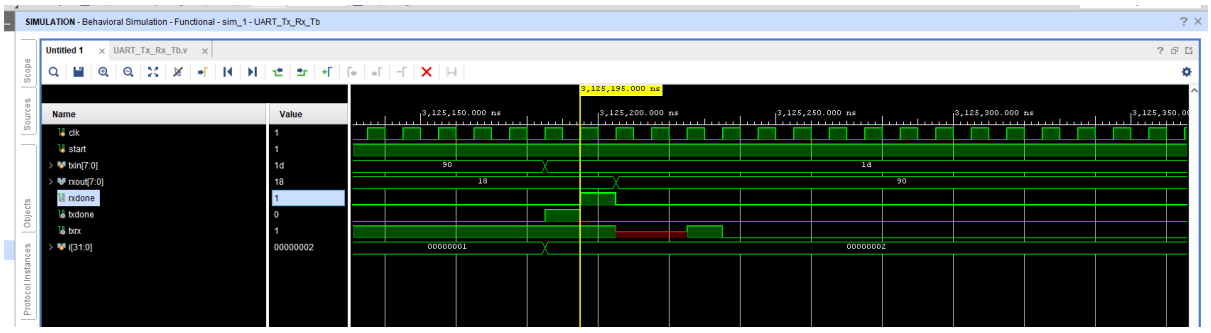
V2 Ended

V3

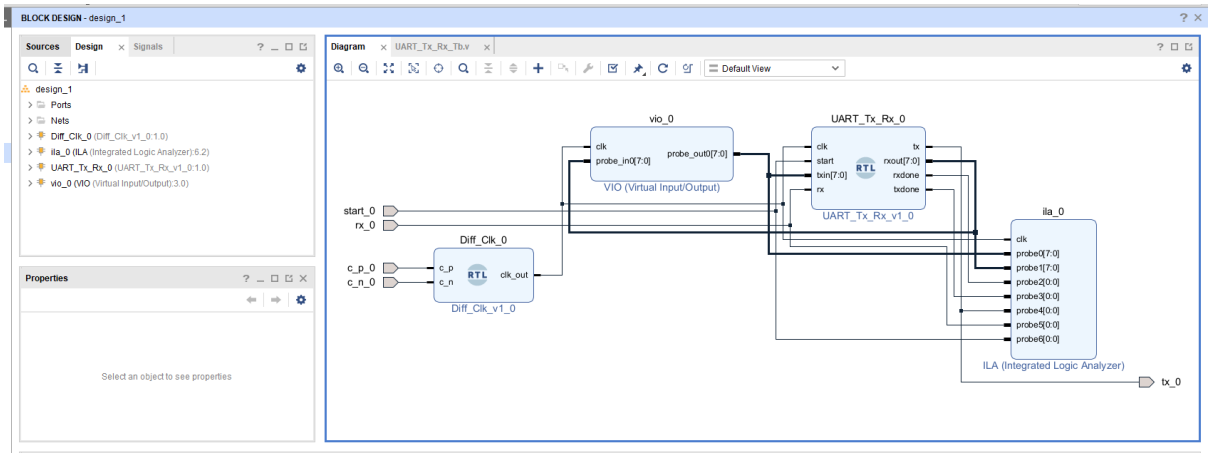
V2 is also working fine but we added some improvements in V3.As you are seeing in below image that we are receiving intermediate data also on Rx_Out Register.It may not create issue bcz when rx_done goes high then we have correct data.But to see it clearly we will remove intermediate data.And we will add one cycle delay in Rx_Done.In above both versions Tx_Done and Rx_Done was going high at same time.It is also logically incorrect.So we add one clock cycle delay in it.



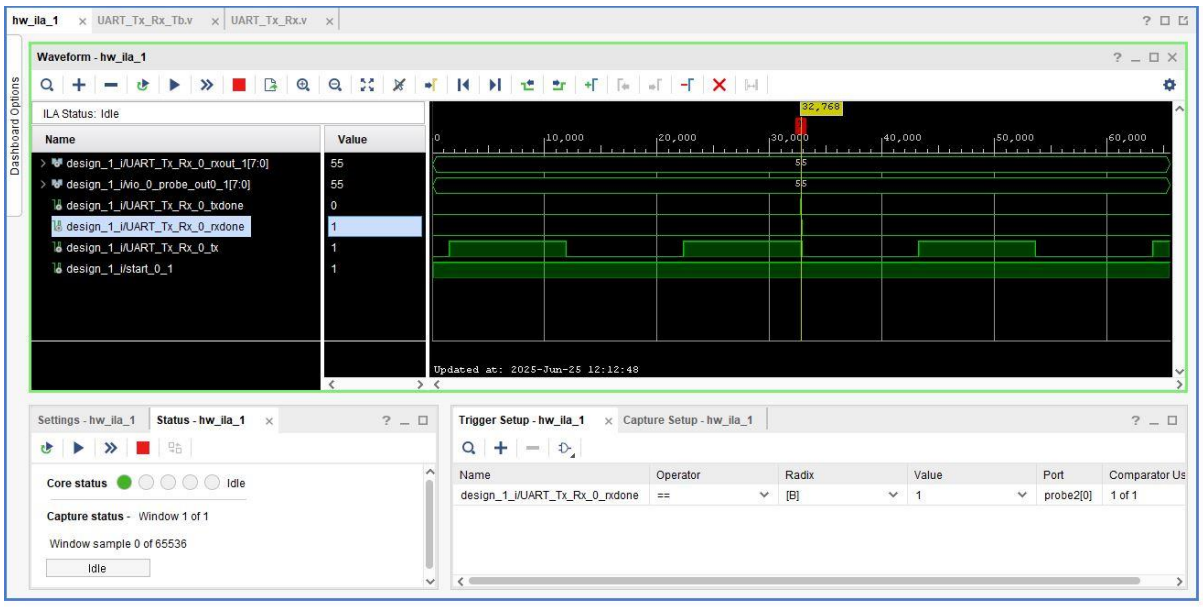
Above is the simulation results with no intermediate data



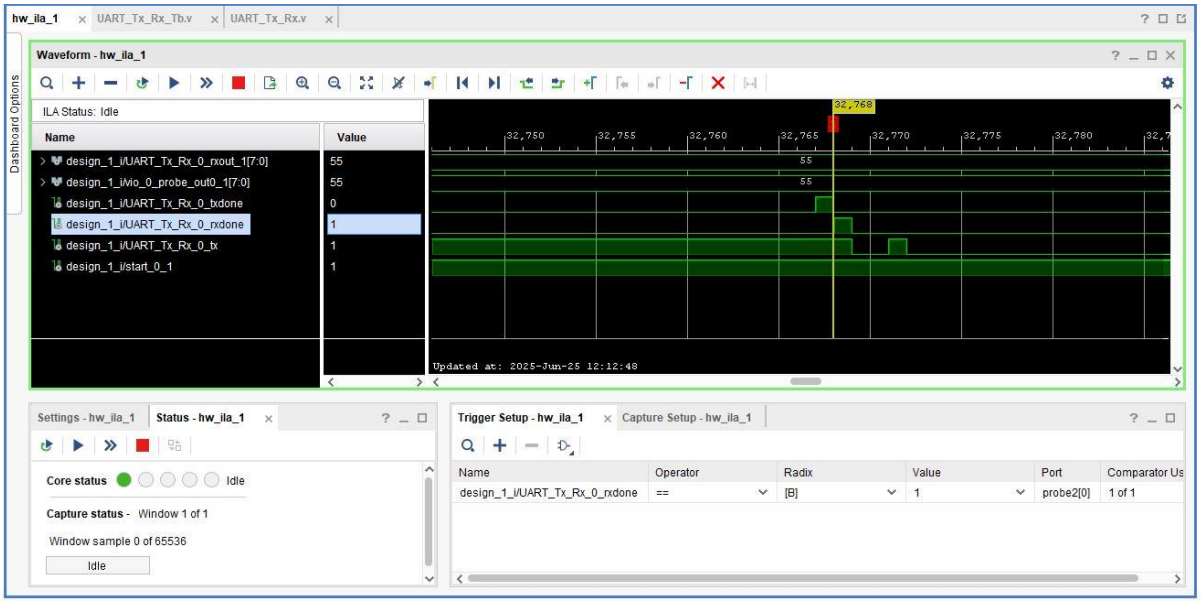
Rx_Done 1 cycle later than Tx_Done.



Block Design for Hardware testing



No intermediate data on hardware

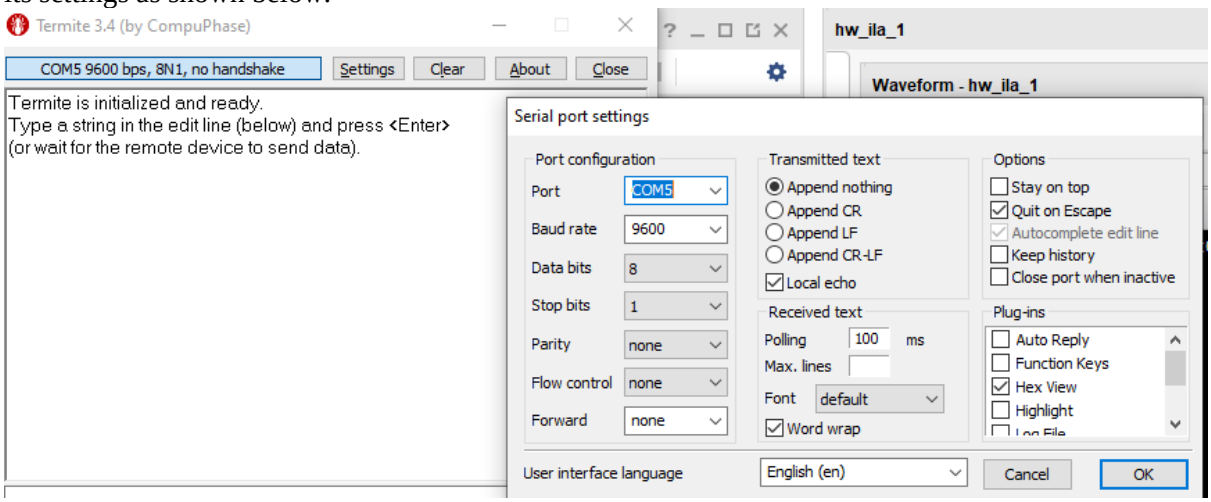


Hardware output with 1 cycle delay

V3 ended

V4

In v4 we will test our UART with PC.Code is little bit modified.
Baud rate is calculated according to actual clock of board. Some registers were combinely used by Tx and Rx ,they got separated.Connect your board either with Rs232/422/485 or through Board specific UART Cable.Before moving towards hardware,install any serial terminal .We installed Termite.Keep its settings as shown below.

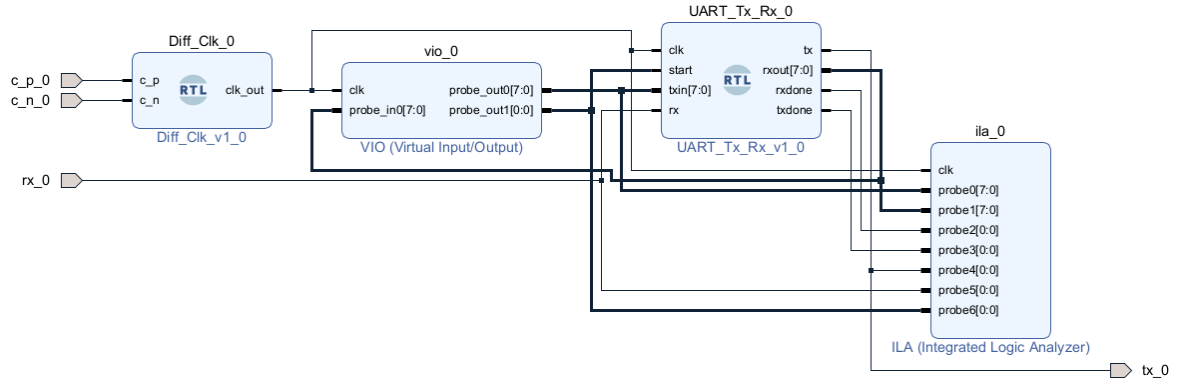


Local echo means when you send byte to Board from PC it will show that byte also on terminal.Set com port as per your pc comport.Remember you will send from Terminal to Board and board will show you the Hex of that.

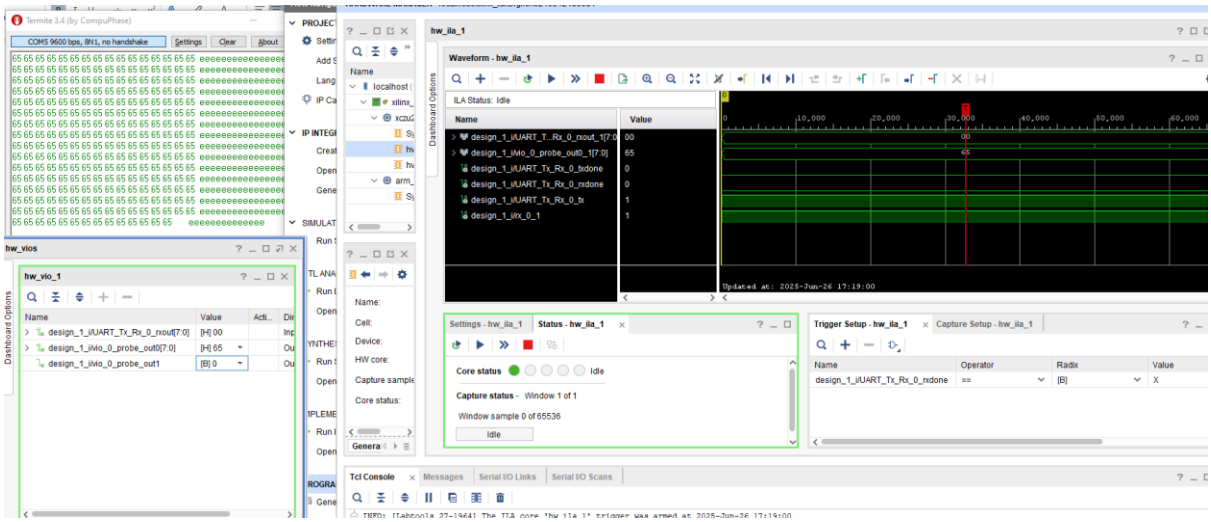
You will transmit hex from board that PC will receive on Termite and show you the hex number that you have send and its equivalent character.

ASCII TABLE

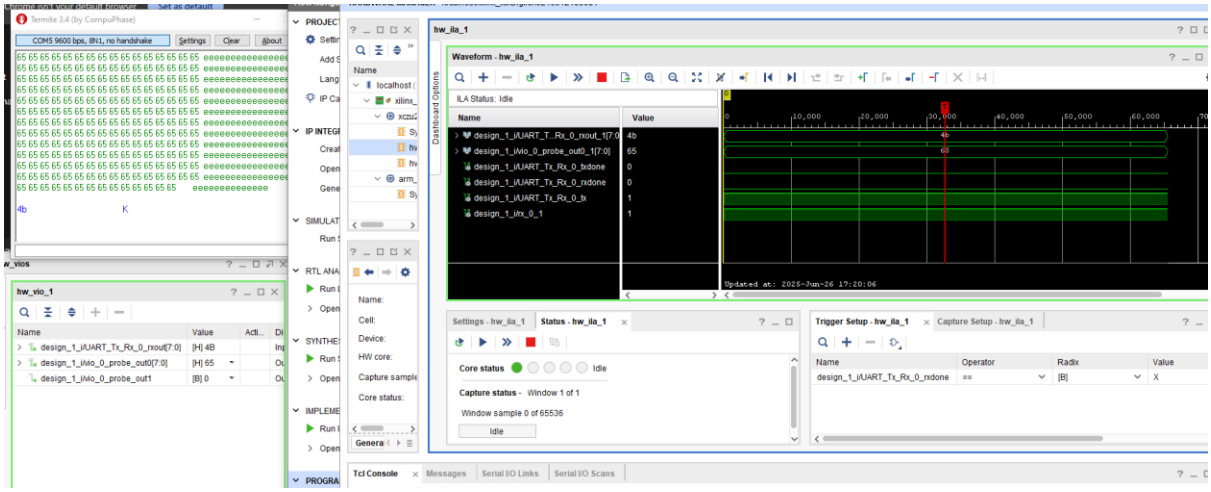
Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]



Block Design



Data sent by Board is correctly received by board



We send char “K” and board shows its hex value 4b