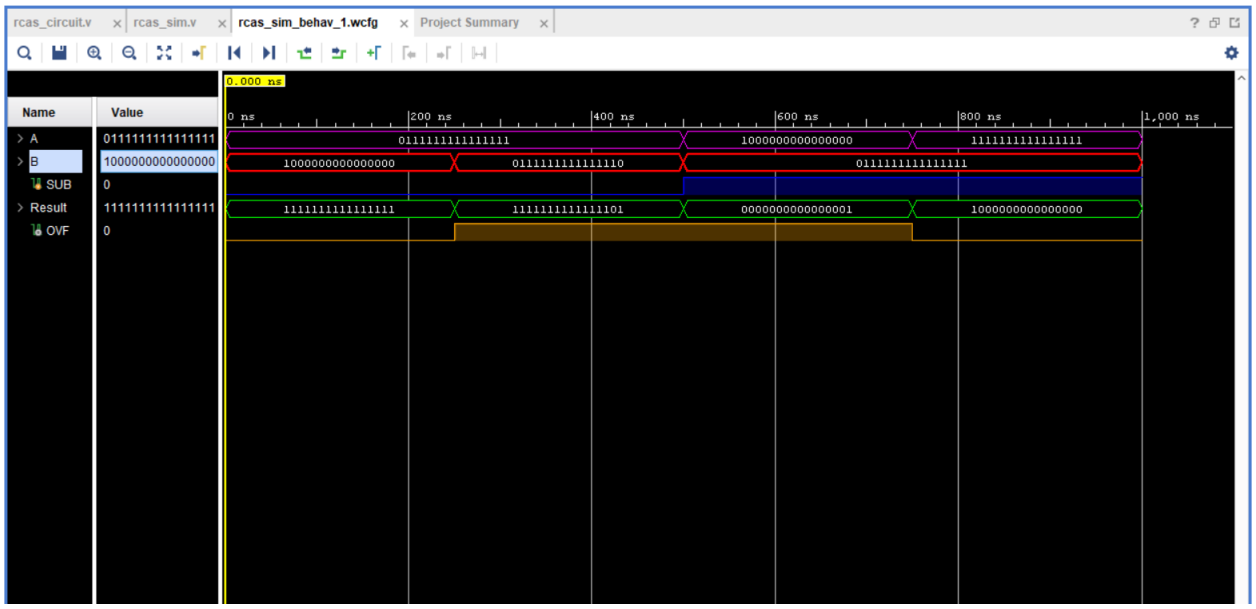
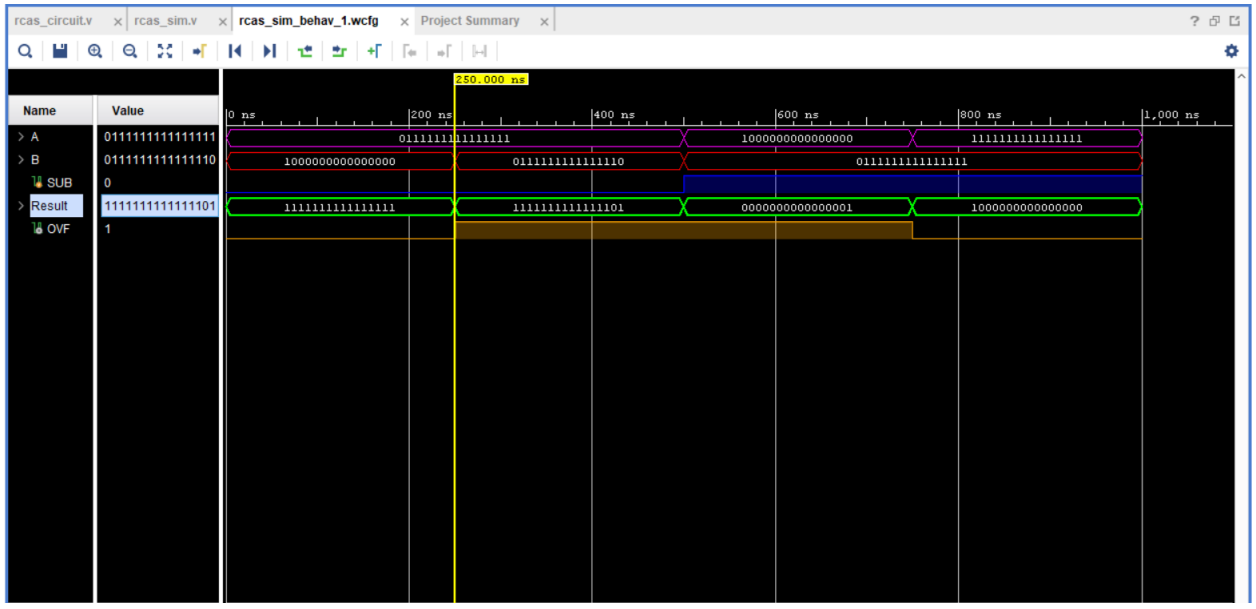
CS303 - Lab 3 Report - Deniz Cangı

1. Simulation results for 16-bit ripple-carry adder-subtractor using full adders

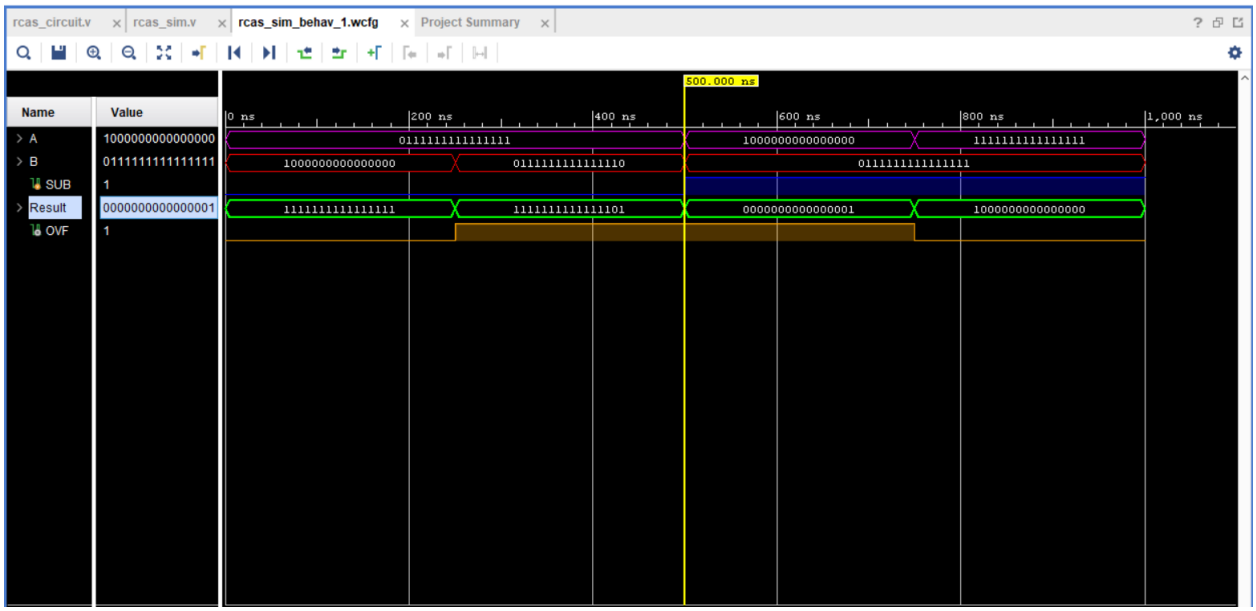
A = 0111 1111 1111 1111
B = 1000 0000 0000 0000
SUB = 0



A = 0111 1111 1111 1111
B = 0111 1111 1111 1110
SUB = 0



A = 1000 0000 0000 0000
B = 0111 1111 1111 1111
SUB = 1

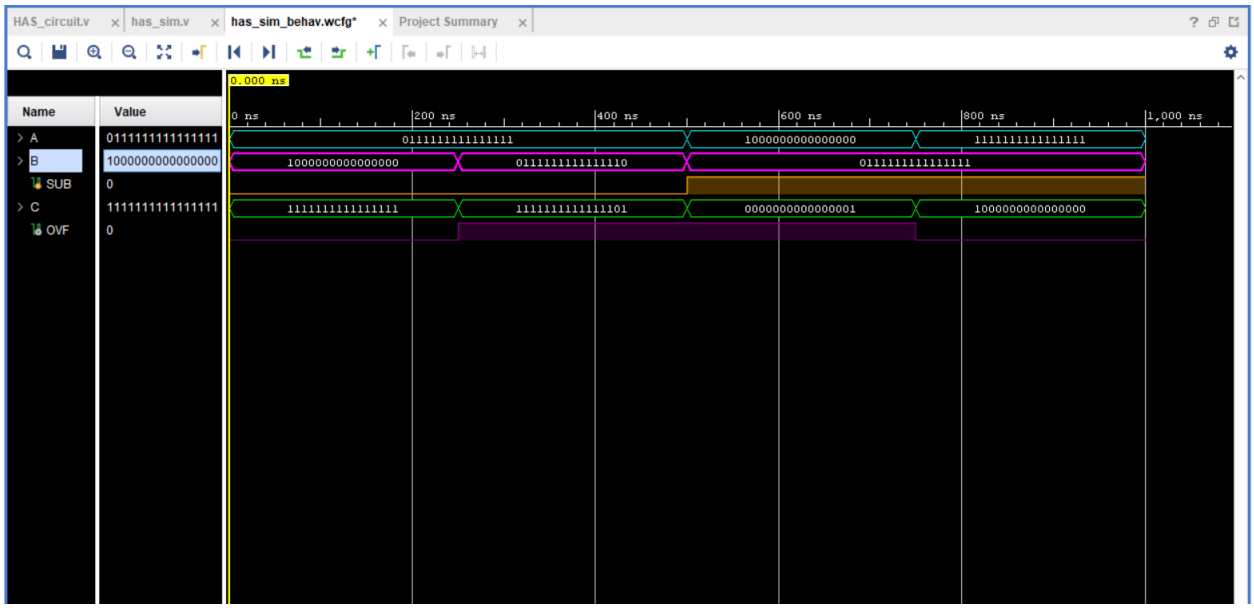


A = 1111 1111 1111 1111
B = 0111 1111 1111 1111
SUB = 1

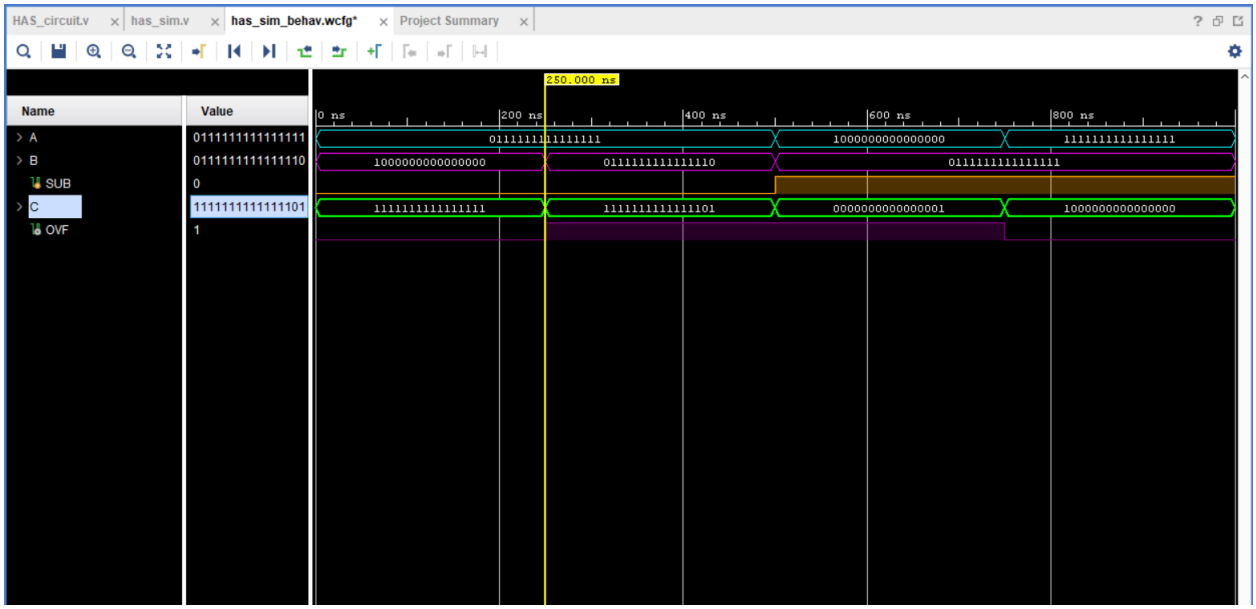
2. **Simulation results for 16-bit hybrid adder-subtractor using four 4-bit carry lookahead adders (CLAs)**

A = 0111 1111 1111 1111

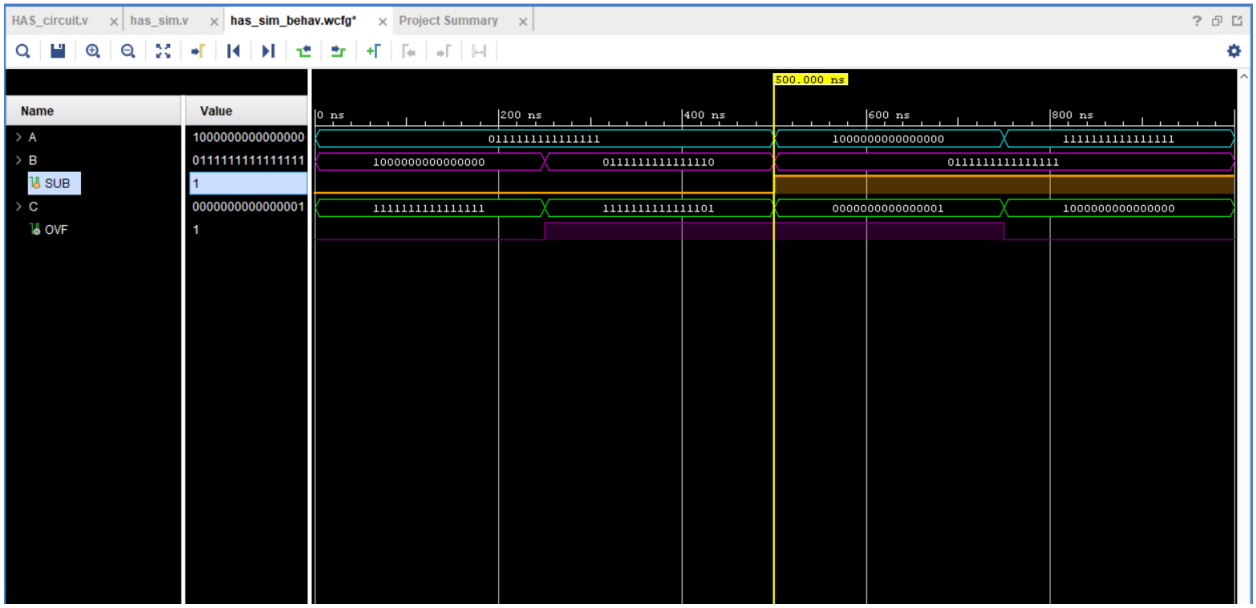
B = 1000 0000 0000 0000
SUB = 0



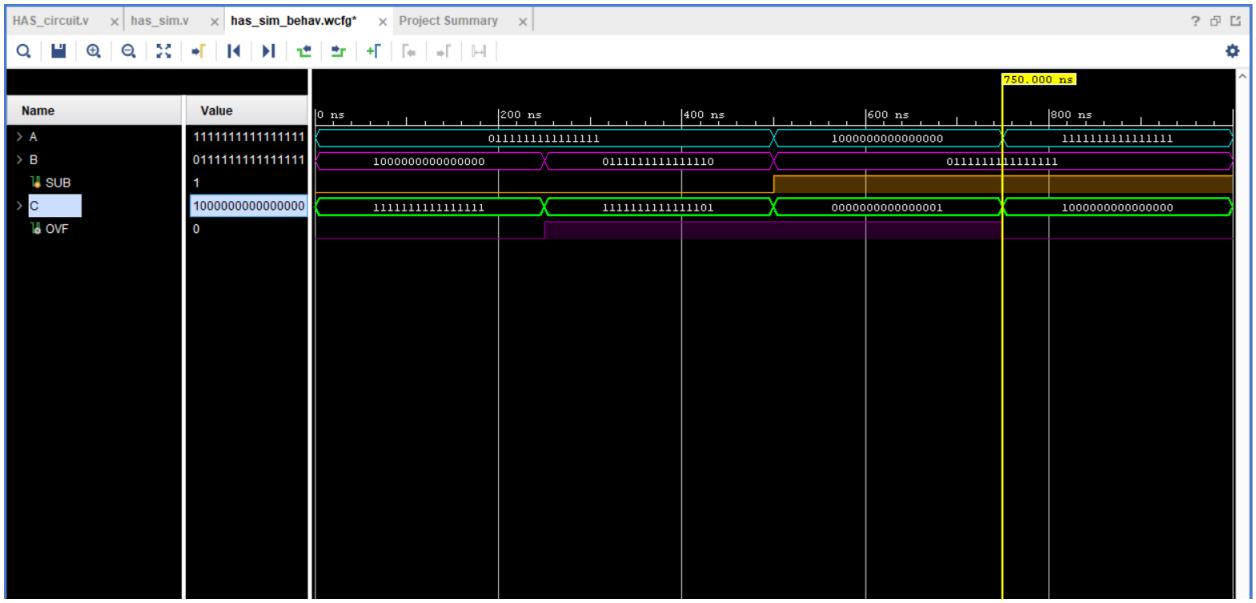
A = 0111 1111 1111 1111
B = 0111 1111 1111 1110
SUB = 0



A = 1000 0000 0000 0000
B = 0111 1111 1111 1111
SUB = 1



A = 1111 1111 1111 1111
B = 0111 1111 1111 1111
SUB = 1



3. Which one of the two is better in terms of area?

16-bit ripple-carry adder-subtractor using full adders is better in terms of area. If we check the utilization tables of 2 designs after the implementation step, the number of LUTs for the first design is 23, and the number of LUTs for the second design is 41.

16-bit ripple-carry adder-subtractor:

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	23	63400	0.04
IO	50	210	23.81

16-bit hybrid adder-subtractor using four 4-bit carry lookahead adders (CLAs)

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	41	63400	0.06
IO	50	210	23.81

4. **Which one of the two is better in terms of time?**

When we use carry lookahead adders, the complexity of the circuit increases in a way that carry delay time (critical path delay) is reduced. So the second design's time complexity is less than the first design's.
Also when we check the implemented designs' timing summary, we can see the maximum delay from one port to another in the combinational delays part.

16-bit ripple-carry adder-subtractor:

Combinational Delays

From Port	To Port	Max Delay [1]	Max Process Corner	Min Delay	Min Process Corner
SUB	OVF	15.407	SLOW	2.951	FAST
A1	OVF	15.259	SLOW	5.626	FAST
SUB	C15	15.102	SLOW	2.825	FAST
A1	C15	14.954	SLOW	5.486	FAST
A0	OVF	14.770	SLOW	5.351	FAST
SUB	C14	14.608	SLOW	2.893	FAST
A0	C15	14.465	SLOW	5.210	FAST
A1	C14	14.460	SLOW	5.336	FAST
B0	OVF	14.271	SLOW	5.132	FAST
SUB	C12	14.005	SLOW	3.043	FAST
A0	C14	13.971	SLOW	5.060	FAST
B0	C15	13.966	SLOW	4.991	FAST
B1	OVF	13.911	SLOW	5.025	FAST
A1	C12	13.857	SLOW	5.074	FAST
SUB	C11	13.719	SLOW	2.888	FAST
SUB	C10	13.706	SLOW	2.835	FAST
SUB	C13	13.607	SLOW	2.968	FAST
B1	C15	13.606	SLOW	4.884	FAST
A2	OVF	13.581	SLOW	4.923	FAST
A1	C11	13.571	SLOW	4.971	FAST
A1	C10	13.558	SLOW	4.969	FAST
B0	C14	13.472	SLOW	4.841	FAST

16-bit hybrid adder-subtractor using four 4-bit carry lookahead adders (CLAs)

Combinational Delays

From Port	To Port	Max Delay 1	Max Process Corner	Min Delay	Min Process Corner
SUB	C11	13.009	SLOW	2.929	FAST
SUB	C10	12.945	SLOW	2.887	FAST
SUB	OVF	12.822	SLOW	3.391	FAST
A5	C11	12.692	SLOW	3.665	FAST
A2	OVF	12.537	SLOW	4.339	FAST
A1	C11	12.517	SLOW	4.421	FAST
A1	C10	12.453	SLOW	4.316	FAST
A4	C11	12.359	SLOW	4.213	FAST
A1	OVF	12.330	SLOW	4.449	FAST
SUB	C15	12.311	SLOW	3.109	FAST
SUB	C14	12.260	SLOW	2.932	FAST
A5	OVF	12.214	SLOW	4.350	FAST
B5	OVF	12.085	SLOW	4.086	FAST
SUB	C8	12.043	SLOW	3.068	FAST
B4	C11	12.030	SLOW	4.024	FAST
A2	C15	12.025	SLOW	4.061	FAST
A2	C14	11.975	SLOW	4.015	FAST
A0	C11	11.902	SLOW	4.232	FAST
A4	OVF	11.881	SLOW	4.158	FAST
B5	C11	11.868	SLOW	3.662	FAST
A5	C10	11.848	SLOW	3.640	FAST
SUB	C9	11.841	SLOW	2.898	FAST
A0	C10	11.837	SLOW	4.126	FAST

As it can be seen, in the first circuit, the maximum delay from one port to another is more, where in the first circuit the maximum delay is 15.4, in the second circuit maximum delay is 13. This shows that in terms of time, the second design is more efficient.