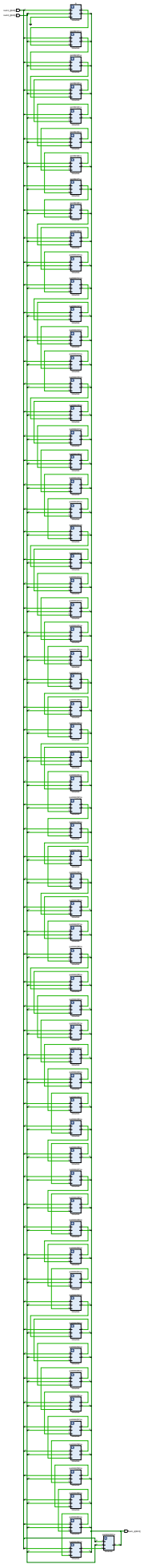
BİL 265 Project Report

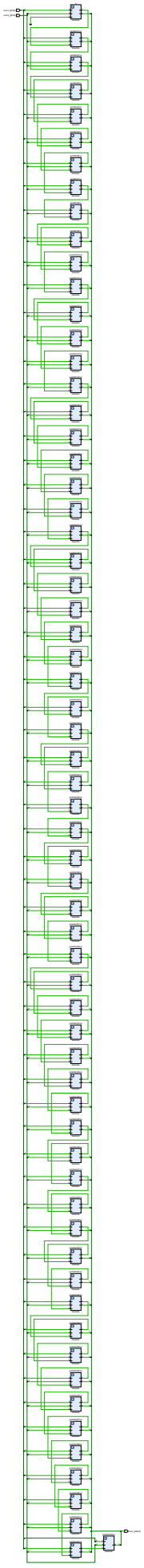
Method 1 (Ripple Carry Adder)

A 64-bit Ripple Carry Adder is built by connecting 64 full-adders so that the carry-out from each full-adder is the carry-in to the next stage. The sum and carry bits are generated sequentially, starting from the least significant bit.

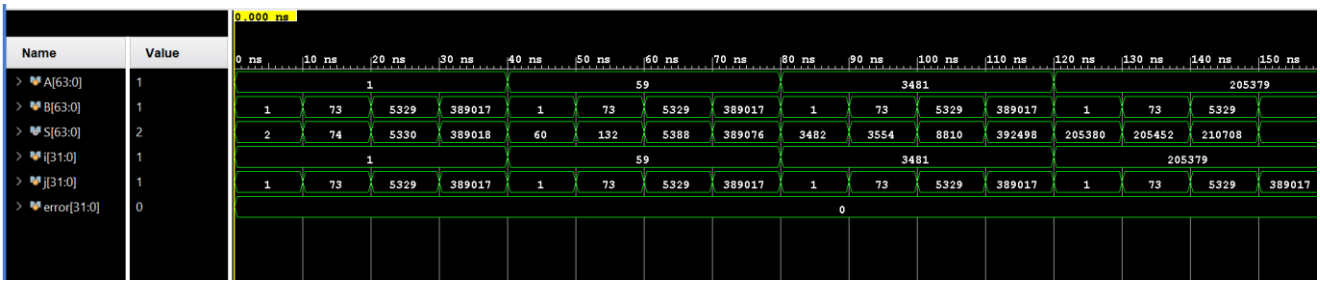
Elaborated Design:



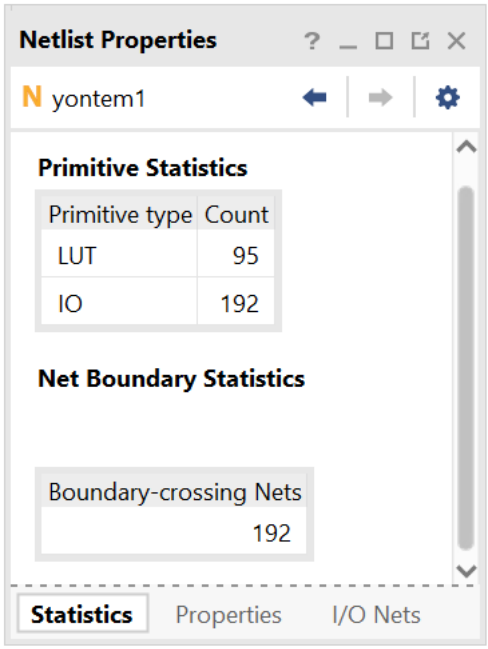
Schematic:



Test Bench Simulation:



Resource Usage:



Delay:

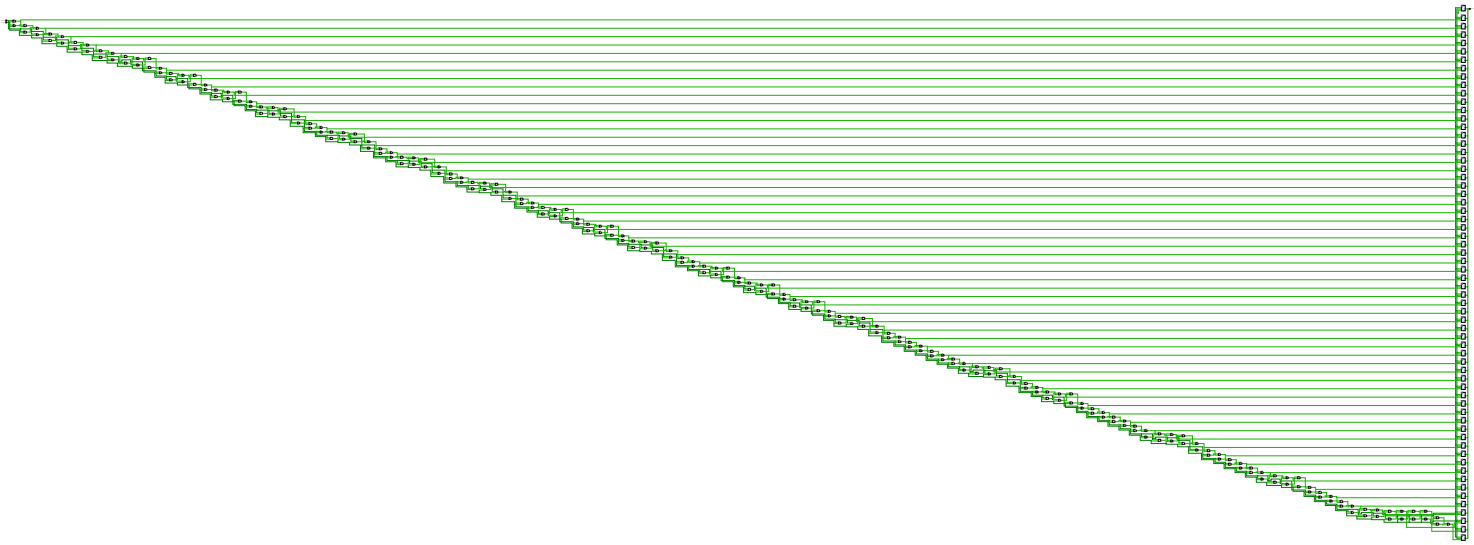
Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Except
Path 1	∞	34	35	3	num1_i[2]	sum_o[63]	23.403	7.345	16.058	∞	input port clock		
Path 2	∞	33	34	3	num1_i[2]	sum_o[61]	22.836	7.227	15.609	∞	input port clock		
Path 3	∞	33	34	3	num1_i[2]	sum_o[62]	22.836	7.227	15.609	∞	input port clock		
Path 4	∞	32	33	3	num1_i[2]	sum_o[59]	22.251	7.109	15.142	∞	input port clock		
Path 5	∞	32	33	3	num1_i[2]	sum_o[60]	22.251	7.109	15.142	∞	input port clock		
Path 6	∞	31	32	3	num1_i[2]	sum_o[57]	21.666	6.991	14.675	∞	input port clock		
Path 7	∞	31	32	3	num1_i[2]	sum_o[58]	21.666	6.991	14.675	∞	input port clock		
Path 8	∞	30	31	3	num1_i[2]	sum_o[55]	21.081	6.873	14.208	∞	input port clock		
Path 9	∞	30	31	3	num1_i[2]	sum_o[56]	21.081	6.873	14.208	∞	input port clock		
Path 10	∞	29	30	3	num1_i[2]	sum_o[53]	20.496	6.755	13.741	∞	input port clock		

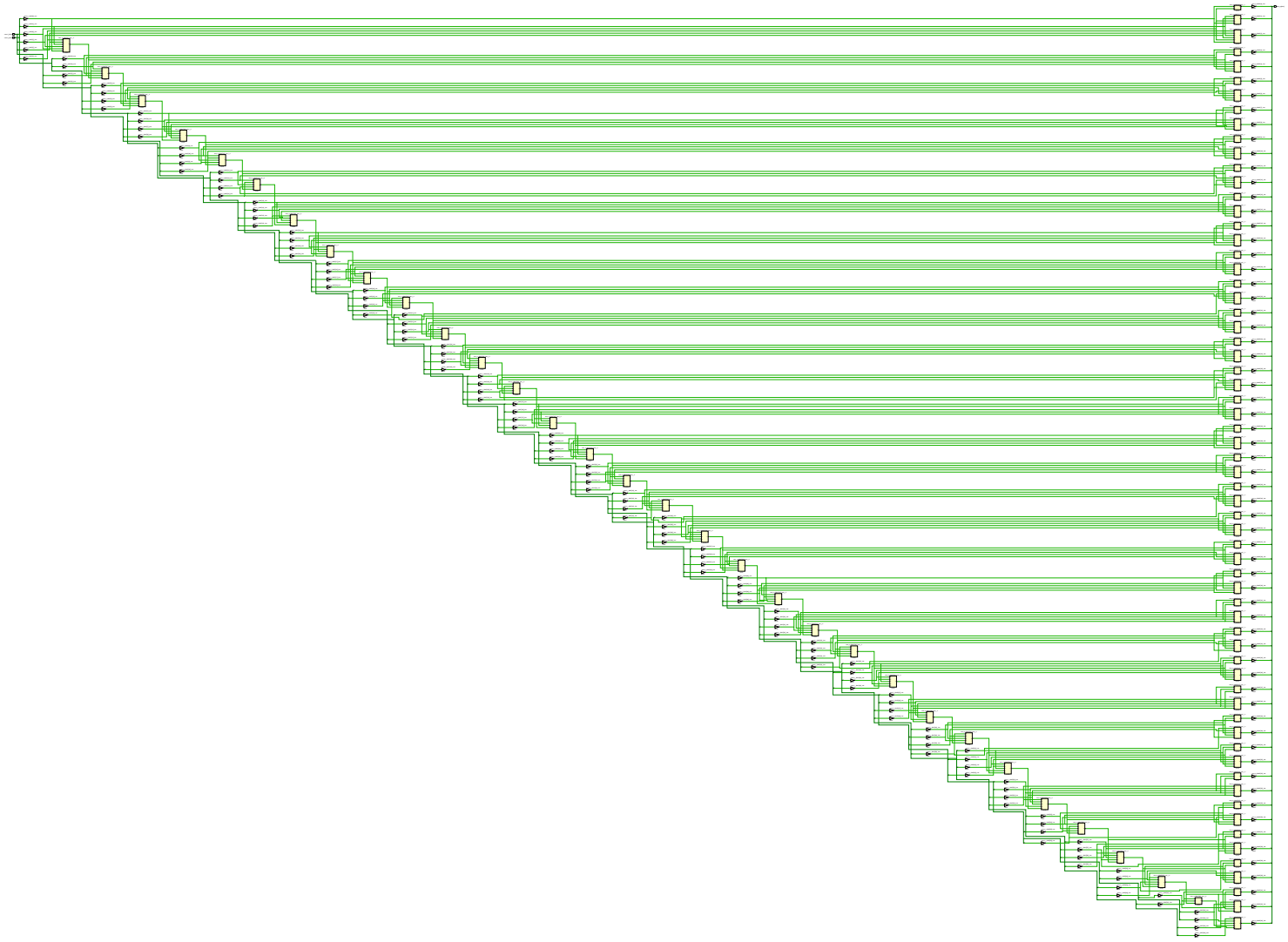
Method 2 (Carry Lookahead Adder)

For each bit in a binary sequence to be added, the carry-lookahead logic will determine whether that bit pair will generate a carry or propagate a carry. This allows the circuit to "pre-process" the two numbers being added to determine the carry ahead of time. Then, when the actual addition is performed, there is no delay from waiting for the ripple-carry effect (or time it takes for the carry from the first full adder to be passed down to the last full adder).

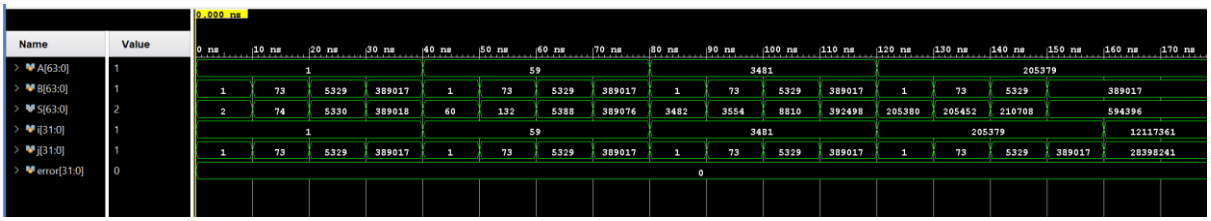
Elaborated Design:



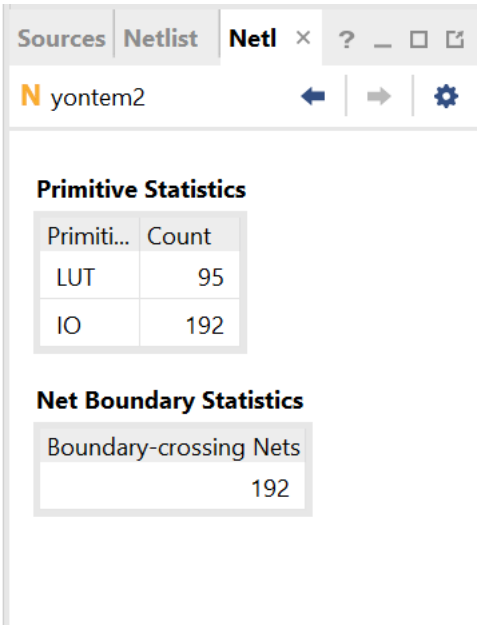
Schematic:



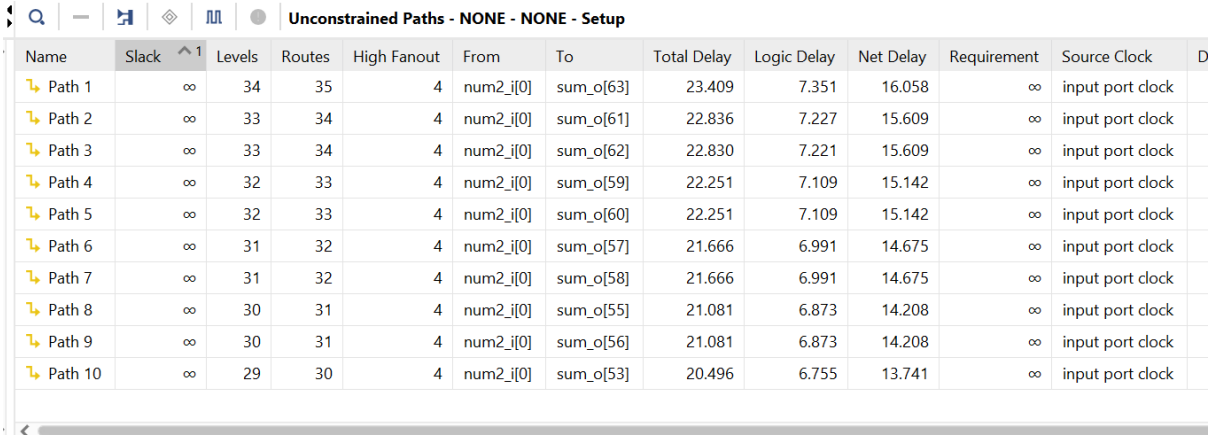
Test Bench Simulation:



Resource Usage:



Delay:

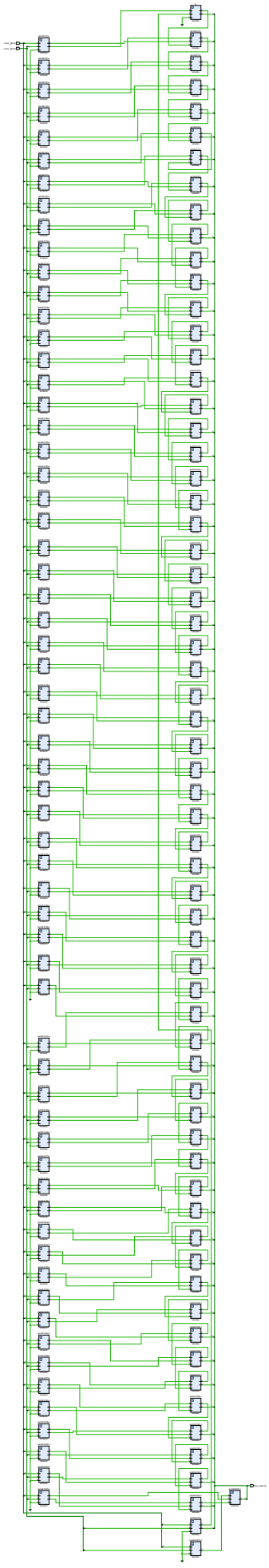


Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	D
Path 1	∞	34	35	4	num2_i[0]	sum_o[63]	23.409	7.351	16.058	∞	input port clock	
Path 2	∞	33	34	4	num2_i[0]	sum_o[61]	22.836	7.227	15.609	∞	input port clock	
Path 3	∞	33	34	4	num2_i[0]	sum_o[62]	22.830	7.221	15.609	∞	input port clock	
Path 4	∞	32	33	4	num2_i[0]	sum_o[59]	22.251	7.109	15.142	∞	input port clock	
Path 5	∞	32	33	4	num2_i[0]	sum_o[60]	22.251	7.109	15.142	∞	input port clock	
Path 6	∞	31	32	4	num2_i[0]	sum_o[57]	21.666	6.991	14.675	∞	input port clock	
Path 7	∞	31	32	4	num2_i[0]	sum_o[58]	21.666	6.991	14.675	∞	input port clock	
Path 8	∞	30	31	4	num2_i[0]	sum_o[55]	21.081	6.873	14.208	∞	input port clock	
Path 9	∞	30	31	4	num2_i[0]	sum_o[56]	21.081	6.873	14.208	∞	input port clock	
Path 10	∞	29	30	4	num2_i[0]	sum_o[53]	20.496	6.755	13.741	∞	input port clock	

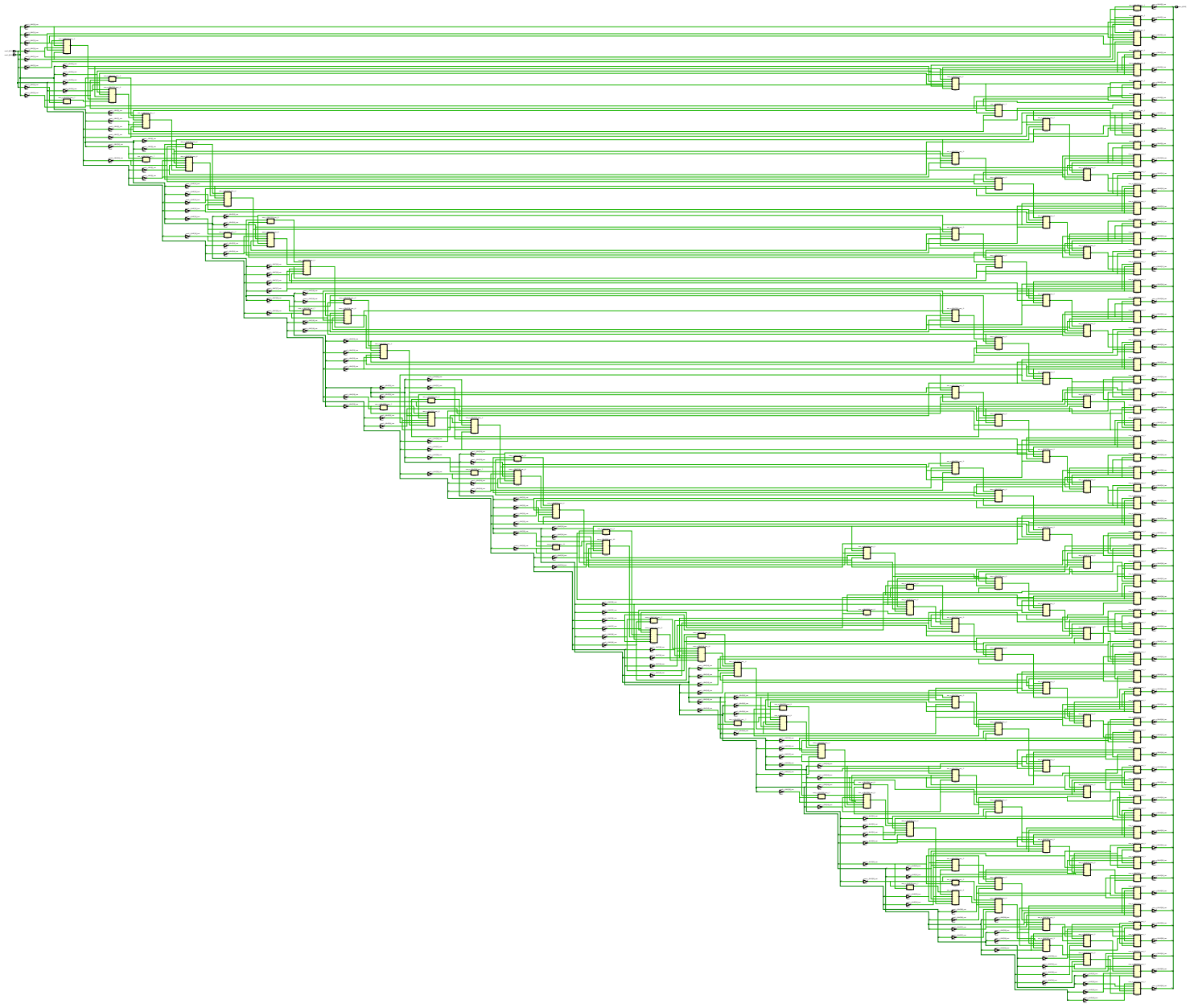
Method 3 (Carry Save Adder)

A Carry Save Adder does not transfer the intermediate carries to the next stages, but instead saves the carry and addends to the sum of next stage using another full adder. This method of adding bits is generally for 3 binary numbers, so we used 64’b0 as our third number. As a result of that our delay is a bit high for this method.

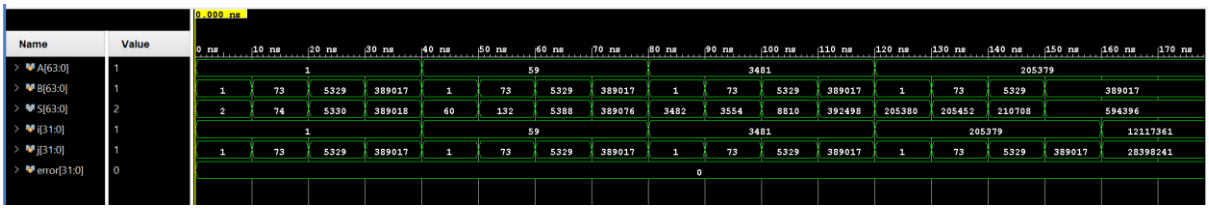
Elaborated Design:



Schematic:



Test Bench Simulation:



Resource Usage:

Primitive Statistics

Primiti...	Count
LUT	158
IO	192

Net Boundary Statistics

Boundary-crossing Nets
192

Delay:

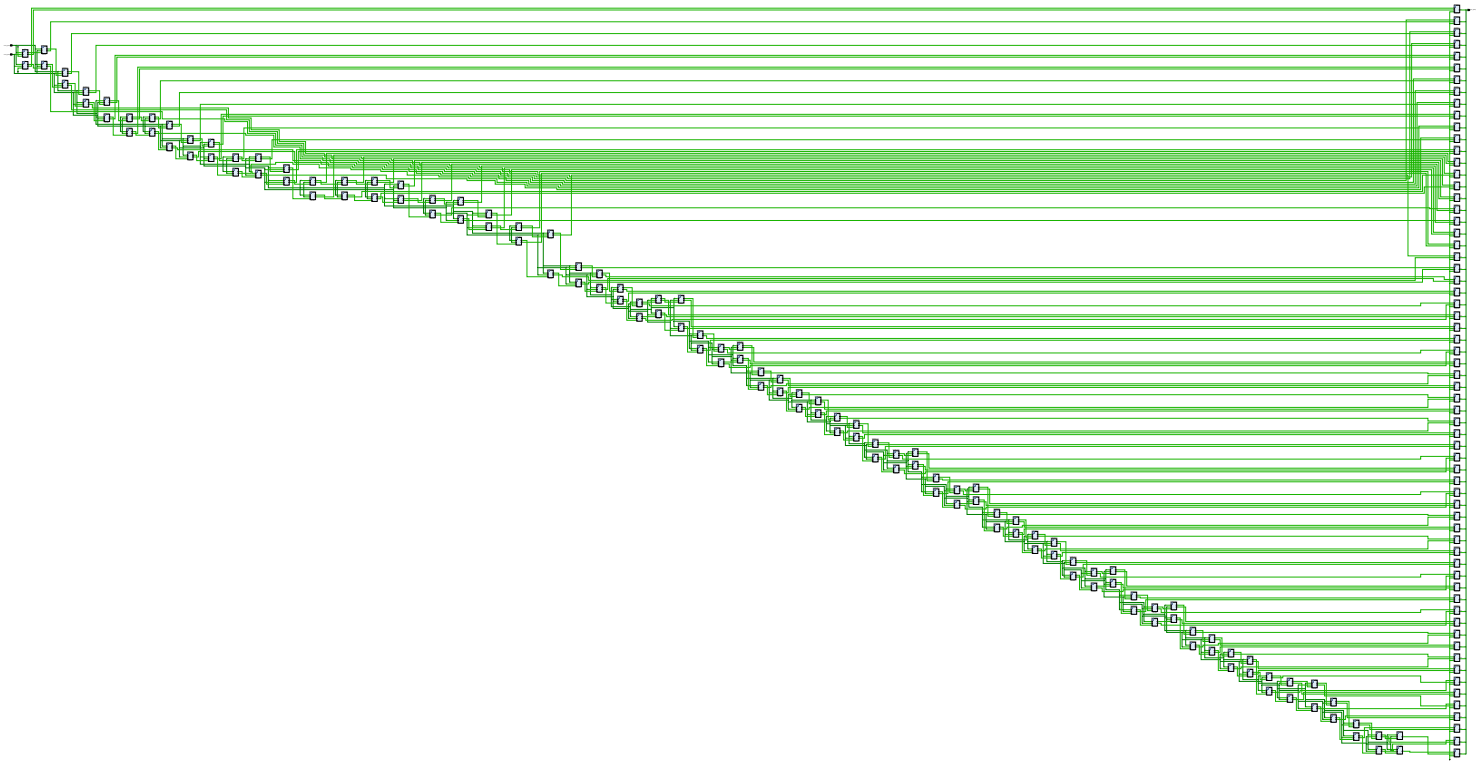
Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	28	27	5	num2_i[5]	sum_o[62]	41.416	7.689	33.727	∞	input port clock
Path 2	∞	28	27	5	num2_i[5]	sum_o[61]	41.022	7.499	33.523	∞	input port clock
Path 3	∞	28	27	5	num2_i[5]	sum_o[63]	40.991	7.040	33.951	∞	input port clock
Path 4	∞	27	26	5	num2_i[5]	sum_o[60]	40.220	7.145	33.074	∞	input port clock
Path 5	∞	27	26	5	num2_i[5]	sum_o[59]	39.667	7.151	32.516	∞	input port clock
Path 6	∞	26	25	5	num2_i[5]	sum_o[58]	39.067	6.782	32.285	∞	input port clock
Path 7	∞	26	25	5	num2_i[5]	sum_o[57]	35.244	7.433	27.811	∞	input port clock
Path 8	∞	26	25	5	num2_i[5]	sum_o[56]	34.855	7.229	27.626	∞	input port clock
Path 9	∞	25	24	5	num2_i[5]	sum_o[55]	33.706	6.884	26.821	∞	input port clock
Path 10	∞	22	21	5	num2_i[5]	sum_o[47]	33.684	7.018	26.665	∞	input port clock

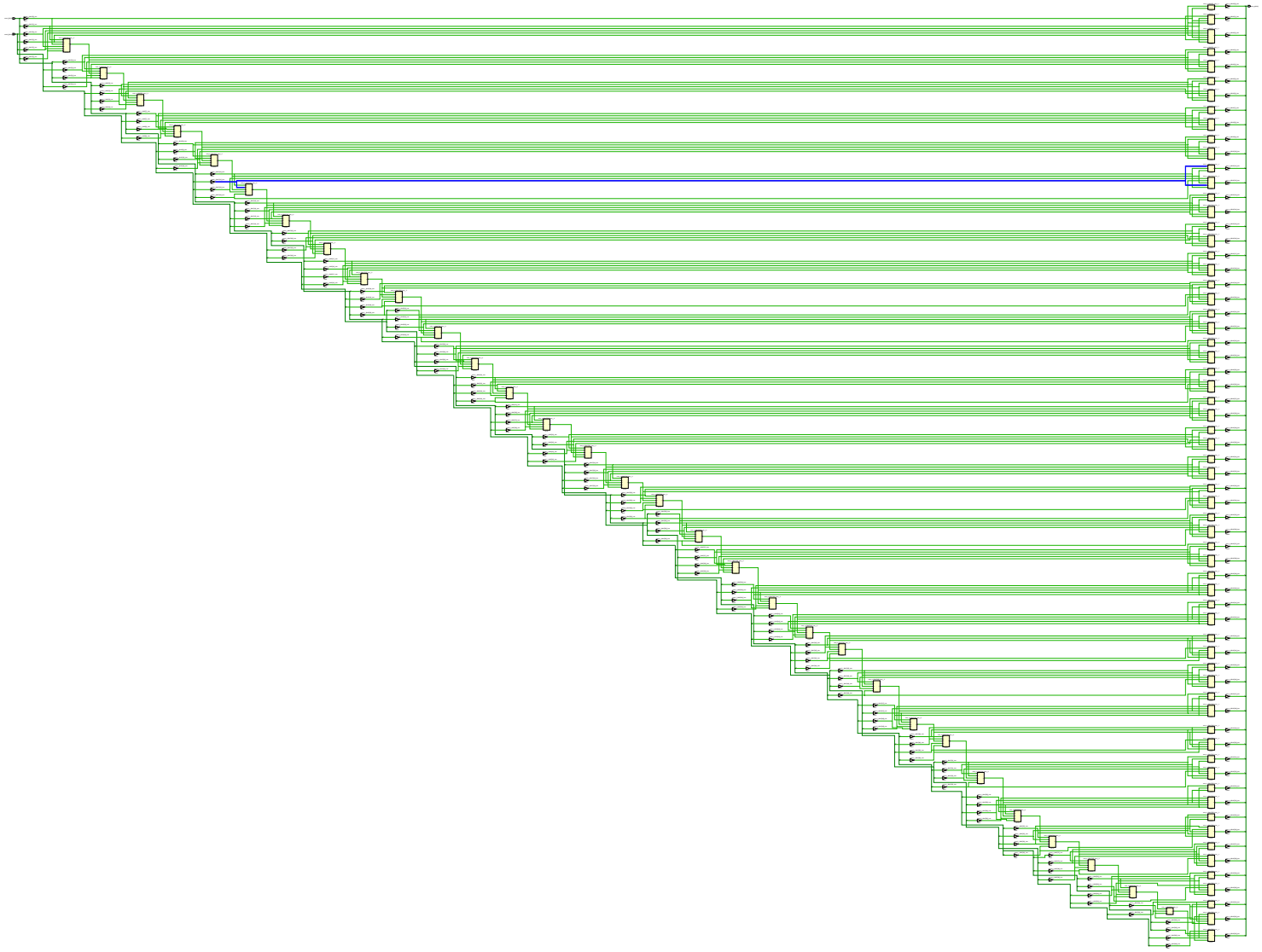
Method 4 (Carry Select Adder)

CSLA use multiple narrow adders to create fast wide adders. A CSLA breaks the addition problem into smaller groups. It is one of the fast types of adders. The adder consists of two independent units. Each unit implements the addition operation in parallel. One way to speed up the addition into several smaller groups, with each having N-bit, say 8-bit groups and then for each group four additions are performed in parallel; one assumes carry in is " 0"(CIN=0) and the other assuming the carry in is “1" (CIN=1). When the carry in is eventually known the correct sum is simply selected through an N-bit using 2-to-1 mux. The adder based on this approach is known as carry select adder (CSLA).

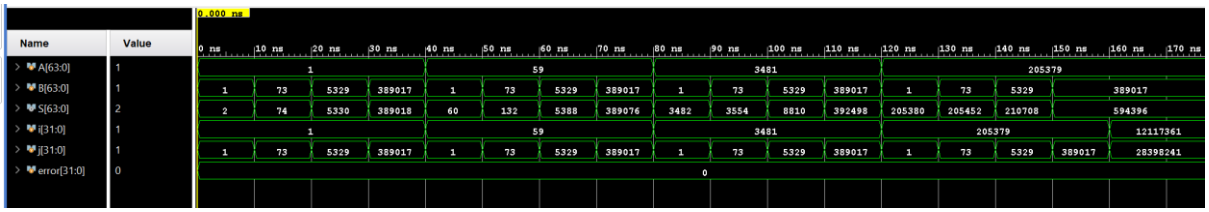
Elaborated Design:



Schematic:



Test Bench Simulation:



Resource Usage:

Primitive Statistics

Primiti...	Count
LUT	95
IO	192

Net Boundary Statistics

Boundary-crossing Nets
192

Delay:

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Dest
Path 1	∞	34	33	4	num2_i[0]	sum_o[63]	62.619	13.716	48.903	∞	input port clock	
Path 2	∞	33	32	4	num2_i[0]	sum_o[62]	62.384	13.817	48.568	∞	input port clock	
Path 3	∞	33	32	4	num2_i[0]	sum_o[61]	61.564	13.595	47.970	∞	input port clock	
Path 4	∞	32	31	4	num2_i[0]	sum_o[60]	60.947	13.239	47.708	∞	input port clock	
Path 5	∞	32	31	4	num2_i[0]	sum_o[59]	60.276	13.245	47.031	∞	input port clock	
Path 6	∞	31	30	4	num2_i[0]	sum_o[58]	59.580	12.874	46.706	∞	input port clock	
Path 7	∞	31	30	4	num2_i[0]	sum_o[57]	58.904	12.875	46.029	∞	input port clock	
Path 8	∞	30	29	4	num2_i[0]	sum_o[56]	47.113	12.547	34.566	∞	input port clock	
Path 9	∞	30	29	4	num2_i[0]	sum_o[55]	47.064	12.555	34.509	∞	input port clock	
Path 10	∞	29	28	4	num2_i[0]	sum_o[54]	45.763	12.194	33.568	∞	input port clock	