Final team project- Design the 128-bit adder tree

Team member: 311510167 柯瑋勳 310510174 曾昱文 311510205 黃煒恩

Design an Adder Tree

Spec:

※Inputs (A1,A2, … ,A128):

→128 * 1bit

→The minimum sampling rate is 2 GHz.

→You can add any extra control signals for your design

※Output (O8- O1):

→8 bits

I. Introduction:

128-bit Adder Tree, is simply a circuit that sums up 128-bit input signal into a 8 bits output. As shown below, we came up with 2 different strategies for improving the performance of adder tree:

Type1: w/o &w/ pipeline in a sequential circuit architecture

Pipelining, an approach to optimize sequential circuits, could increase the throughput of the design

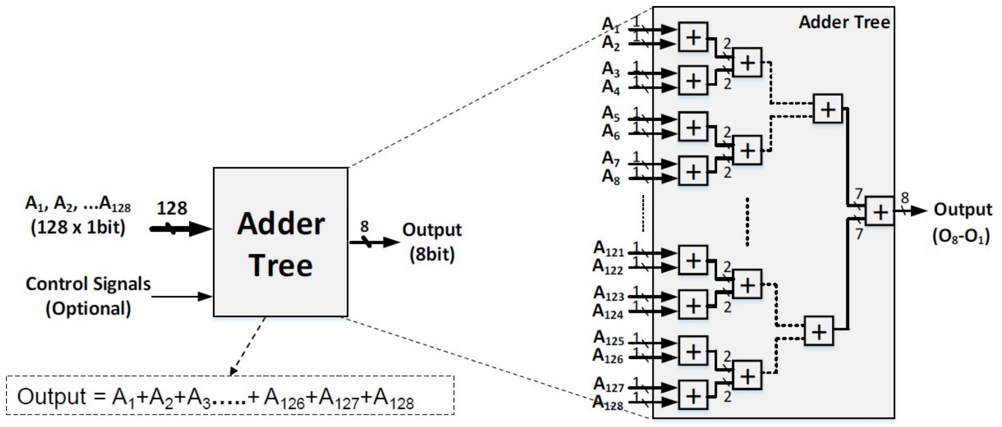
Type2: w/o &w/ Carry-Lookahead Adder (CLA)

To reduce the circuit latency, we aim to improve the design metrics for faster adder. Therefore, we replace the original Ripple-Carry Adder(RCA) with Carry-Lookahead Adder(CLA). In addition, we improved the architecture by using fewer adders, which is supposed consume less area.

This report compares different models of 128-bit Adder Tree and analysis their performance.

II. Type1: w/o &w/ pipeline analysis

2.1 Architectures of 128bit adder tree without pipeline


$A_1, A_2, \ldots A_{128}$
(128 x 1bit)
128
Control Signals
(Optional)
Adder Tree
8
Output
(8bit)
Output = $A_1+A_2+A_3\ldots\ldots+A_{126}+A_{127}+A_{128}$
Adder Tree
A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8
A_{121} A_{122} A_{123} A_{124} A_{125} A_{126} A_{127} A_{128}
1
2
7
8
Output
(O_8-O_1)

Architecture:

a. Control signals:

→We add in_valid, rst_n, out_valid for the **pipeline analysis design**

b. Design:

→The 128-bit adder tree has total **7 layer:**

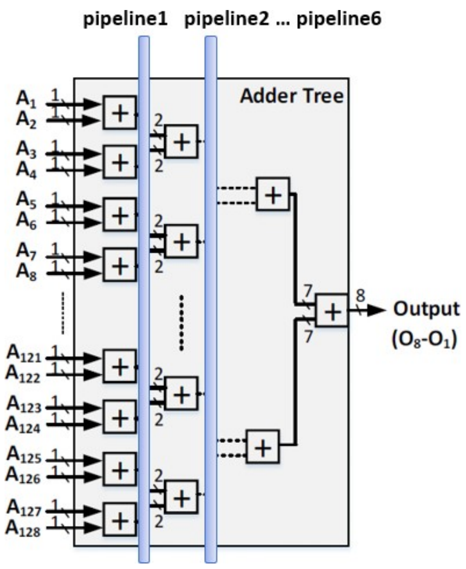
layer1 has 64* 1-bit full adders,

layer2 has 32* 2-bit full adders,

…

layer7 has 2* 7-bit full adders,   and the output can be an 8-bit data

2.2 Architectures of 128bit adder tree **with pipeline**



Architecture:

a. Design:

→We add total **6 pipelines** for the design above:

and we make the original **1 cycle** task into **6 cycles,**

for reducing the critical path timing

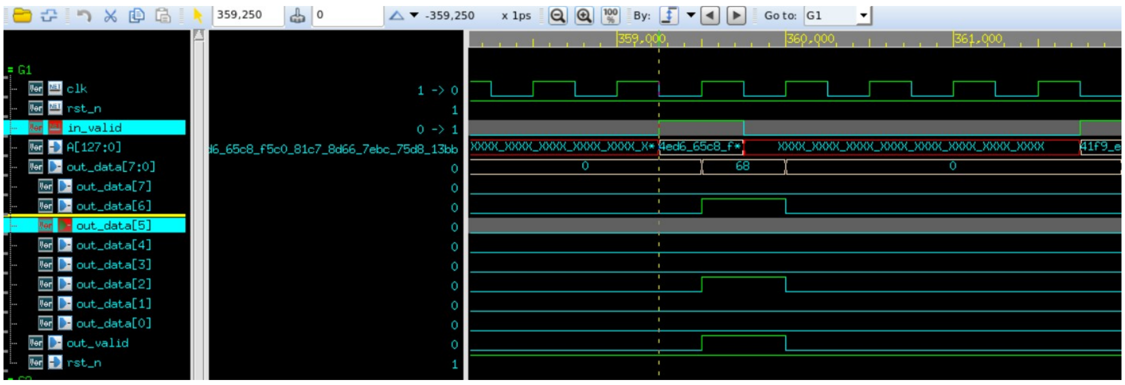
2. Critical path analysis (by timing reports of 02_SYN)
(= from Startpoint to Endpoint)

	without pipeline	using pipeline
Startpoint:	**in_valid**	**s4[5].s4bit_reg[0]**
Incr	0	0
Path	1000.00 r	0.00 r
Endpoint:	**out_data_reg[5]**	**s5[5].s5bit_reg[3]**

Incr	0	0
Path	882.84 r	181.62 r
data arrival time	882.84	181.62

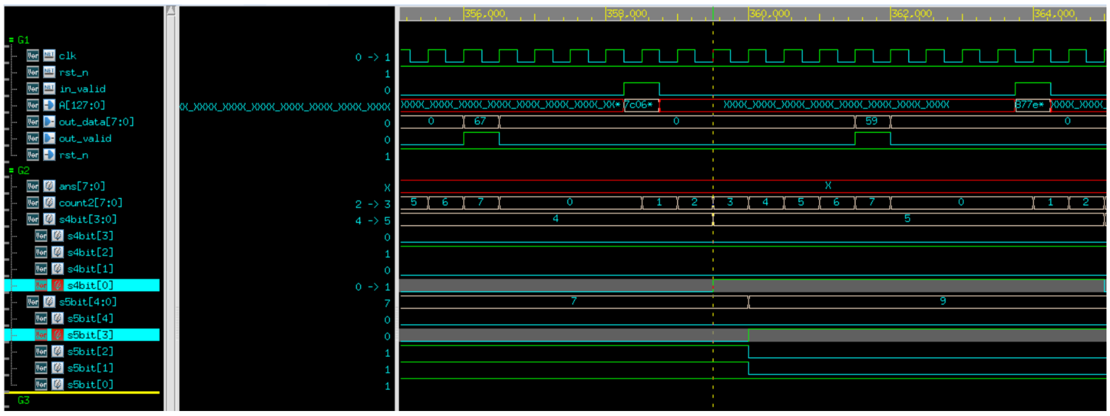
a. without pipeline:

critical path: **in_valid(rise) to out_data_reg[5](rise)**



b. using pipeline:

critical path: **s4[5].s4bit_reg[0](rise) to s5[5].s5bit_reg[3](rise)**

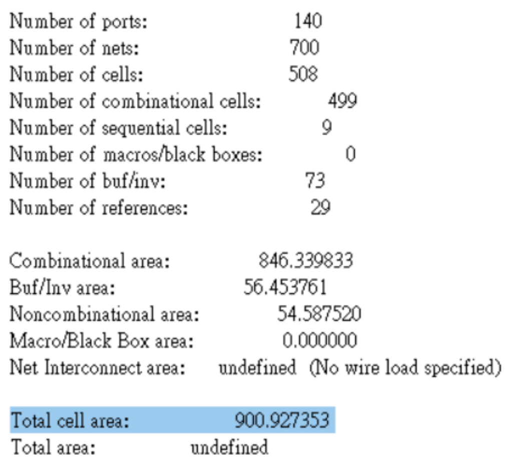


Discussion:

We can find that the critical path timing of using pipeline is smaller than without using pipeline because the 6-layer pipelines can make critical path shorter.

3. Total Area

a. without pipeline:

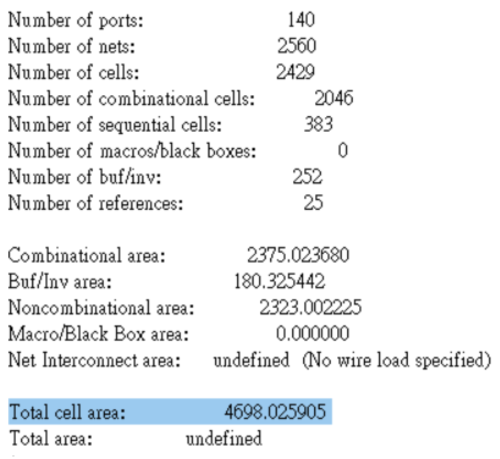


```
Number of ports:                    140
Number of nets:                     700
Number of cells:                    508
Number of combinational cells:      499
Number of sequential cells:           9
Number of macros/black boxes:         0
Number of buf/inv:                   73
Number of references:                29

Combinational area:          846.339833
Buf/Inv area:                 56.453761
Noncombinational area:        54.587520
Macro/Black Box area:          0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:             900.927353
Total area:                 undefined
```

b. using pipeline:



```
Number of ports:                    140
Number of nets:                    2560
Number of cells:                   2429
Number of combinational cells:     2046
Number of sequential cells:         383
Number of macros/black boxes:         0
Number of buf/inv:                  252
Number of references:                25

Combinational area:         2375.023680
Buf/Inv area:                180.325442
Noncombinational area:      2323.002225
Macro/Black Box area:          0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:            4698.025905
Total area:                 undefined
```

Discussion:

We can find that the **Total cell area** of using pipeline is much larger than without using pipeline because the pipelines structure needs more registers for storing the data of each pipeline.

4. Power consumption (by reports of 04_PTPX)

setting: 1000 random patterns

	without pipeline	using pipeline
Net Switching Power	8.079e-05(45.98%)	8.266e-05(7.33%)
Cell Internal Power	9.484e-05(53.98%)	1.044e-03(92.64%)
Cell Leakage Power	5.946e-08 (0.03%)	2.762e-07(0.02%)
Intrinsic Leakage	5.946e-08	2.762e-07
Gate Leakage	0.0000	0.0000
Total Power	**1.757e-04 (100.00%)**	**1.127e-03 (100.00%)**
X Transition Power	7.900e-07	5.067e-06

Glitching Power	0.0000	0.0000
Peak Power	**4.868e-04**	**1.472e-03**
Peak Time	1789	4032

Discussion:

We can find that the **Total Power** of without using pipeline is **much smaller than** using pipeline

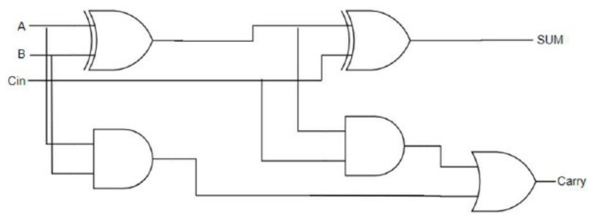
Because the 6-layer pipelines structure will **need more cycles** to finish a computation.

And we can also find that the **Peak Power** of without using pipeline is much smaller than using pipeline.

III. Type2: w/o &w/ Carry-Lookahead Adder (CLA)

3.1 The original 128-bit adder tree

The original 128-bit adder tree design, which is similar to the example adder tree in the Final Project slide, was composed by 1 bit full adders. The one-bit Full Adder shown as below:



The 128-bit adder tree has total 7 layer:

layer1 has 64* 1-bit full adders,
layer2 has 64* 1-bit full adders,
layer3 has 48* 1-bit full adders,
layer4 has 32* 1-bit full adders,
layer5 has 20* 1-bit full adders,
layer6 has 12* 1-bit full adders,
layer7 has 7* 1-bit full adders,
and the output is an 8-bit data

3.2 Proposed Work

Due to the excessive delay of our previous 128-bit adder tree design, we cannot meet the constraint of the sampling rate of 2 GHz. Thus, lowering the circuit delay would be our primary task.

A. Architectures of the proposed design

The idea of the proposed design is to use carry-lookahead adders to **reduce the propagating latency** of each layer's adder. Also, the number of layer 1 FAs can be reduced from 64 to 43 by adding three 1-bit input signals together.

The proposed 128-bit adder tree has total 7 layer:

layer1 has **43*** 1-bit full adders,

layer2 has 21* 2-bit adders,

layer3 has 11* 3-bit CLA,

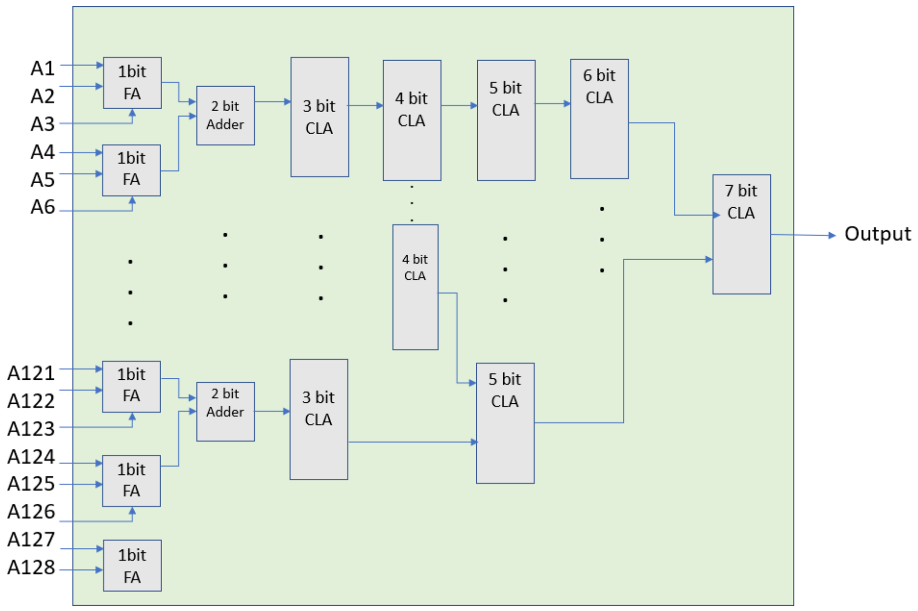
layer4 has 5* 4-bit CLA,

layer5 has 3* 5-bit CLA,

layer6 has 1* 6-bit CLA,

layer7 has 1* 7-bit CLA,

and the output is an 8-bit data



B. Synthesis & Simulation Results

The synthesis and simulation results of the proposed 128-bit adder tree design are presented in this section.

1. Area of the combinational circuits

	Area(μm^2)
The proposed 128-bit adder tree design	1304.035206
The original 128-bit adder tree design	1784.125440

2. Critical path & timing

	The proposed 128-bit adder tree design	Original 128-bit adder tree design
Startpoint:	A[49] (input port)	A[9] (input port)
Endpoint:	out_data[6]	out_data[6]
data arrival time	559.98ps	499.88ps
data required time	560.00ps	500.00ps

3. Power analysis

setting: 129 patterns (from 0 to 128)

	The proposed 128-bit adder tree design	The original 128-bit adder tree design
Net Switching Power	3.740e-06 (46.99%)	4.119e-06 (45.41%)
Cell Internal Power	4.135e-06 (51.94%)	4.856e-06 (53.54%)
Cell Leakage Power	8.541e-08 (1.07%)	9.562e-08 (1.05%)
Intrinsic Leakage	8.541e-08	9.562e-08
Gate Leakage	0.0000	0.0000
Total Power	**7.961e-06 (100.00%)**	**9.070e-06 (100.00%)**
X Transition Power	0.0000	0.0000
Glitching Power	0.0000	0.0000
Peak Power	**5.093e-05**	**4.757e-05**
Peak Time	64	128

Discussion:

1. Area:

It is observed that the **area** of the the proposed design (1304.035206 μm^2) **is smaller** than the original design (1784.125440 μm^2).

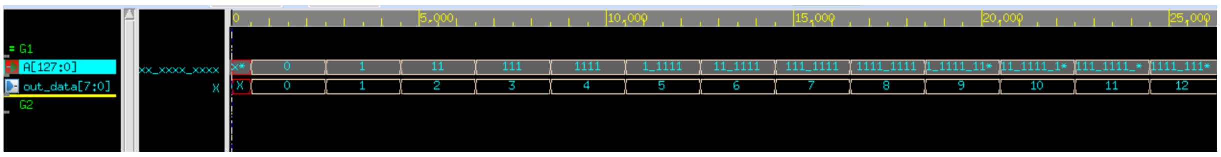
2. Critical path & timing

In terms of delay consumed by the proposed design is 499.88ps whereas the original design is 559.98ps, which **is faster** than the original design

3. Power analysis

Furthermore, the proposed design **consumed less power**. Hence in all aspects, our proposed model is efficient and performed better when compared with the original design.

4. Simulation Results of the proposed design



IV. Conclusion

In this report, we discussed two types of strategies for improving the performance of the 128-bit adder tree.

For the Type 1 strategy, using pipeline structure to implement the circuit could effectively reduce the critical path delay and increase throughput. Thus, the pipeline 128-bit adder tree could operate under 5GHz clock rate, which is about 5 times higher than the original design. The overhead of the pipelined 128-bit adder tree is that it would consume more area due to the extra storage elements and control units.

As for the Type 2 strategy, we considered how to reduce latency and simplify the architecture to save circuit area. From the simulation results, it can be observed that our proposed design shortens the circuit latency by using CLA and also successfully reduces the circuit area. In addition, the proposed design consumes less power than the original one. Therefore, our proposed design can be considered a better architecture design for 128-bit adder tree.