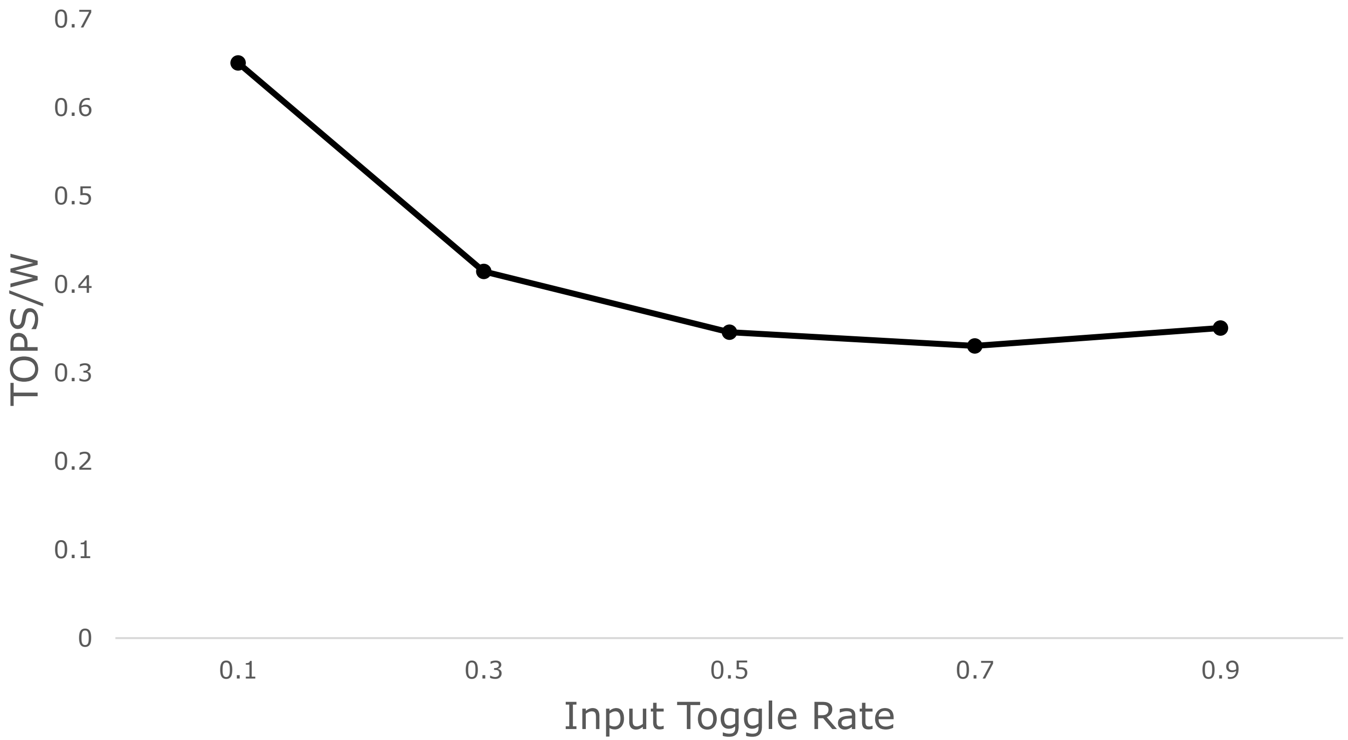
Adder Tree / MAC

B11901027 王仁軒

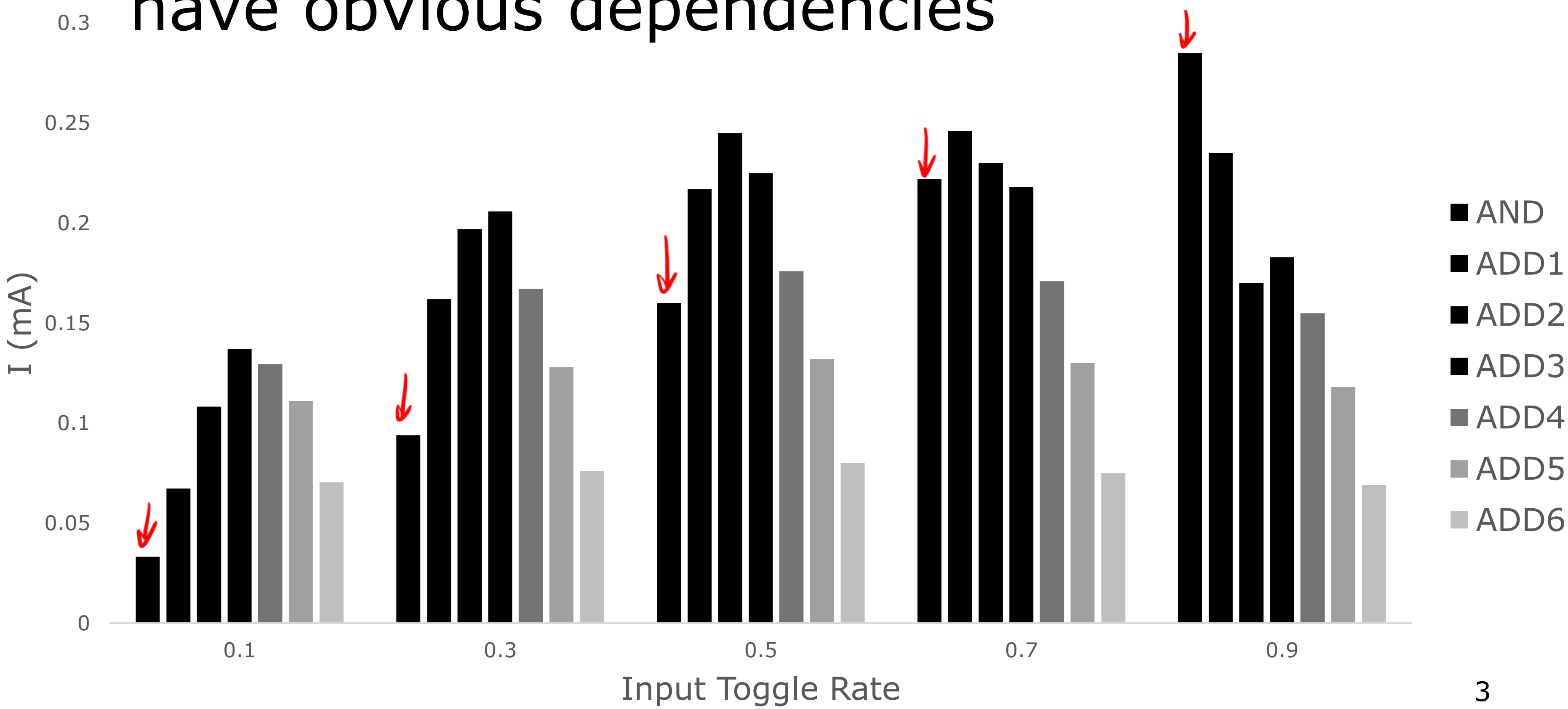
Energy Efficiency

- TOPS/W drops from 0.65 to 0.33 with the increase of input toggle rate
  - T_{cycle} = 2.6ns, 2.30 $TOPS/mm^2$



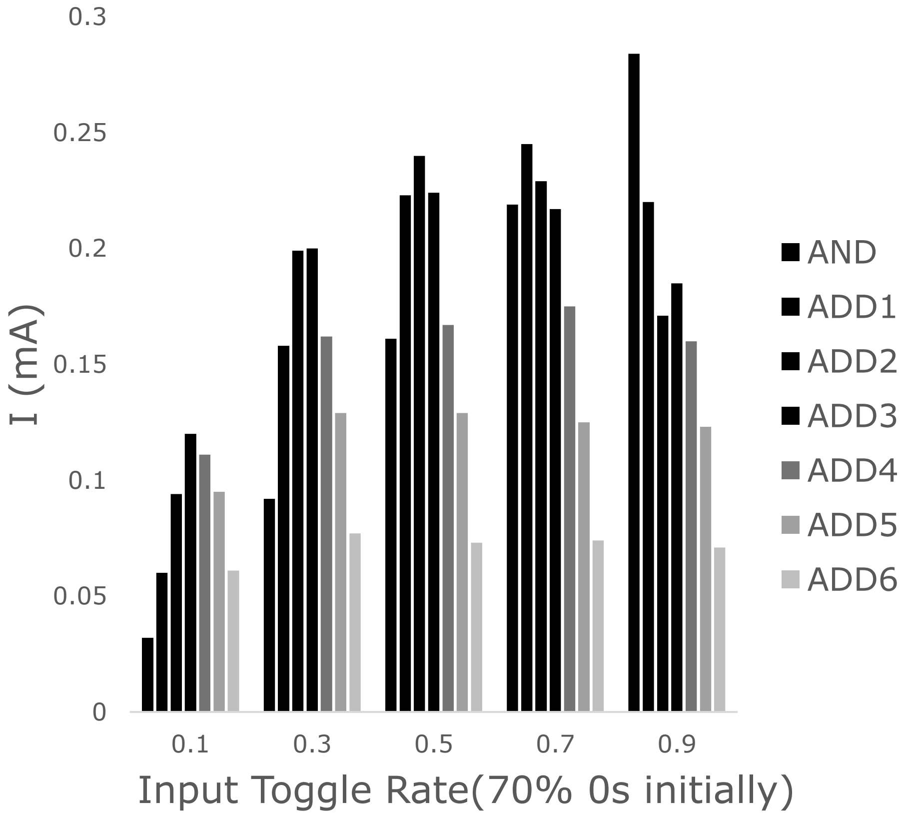
Power of All Adder Tree Stages

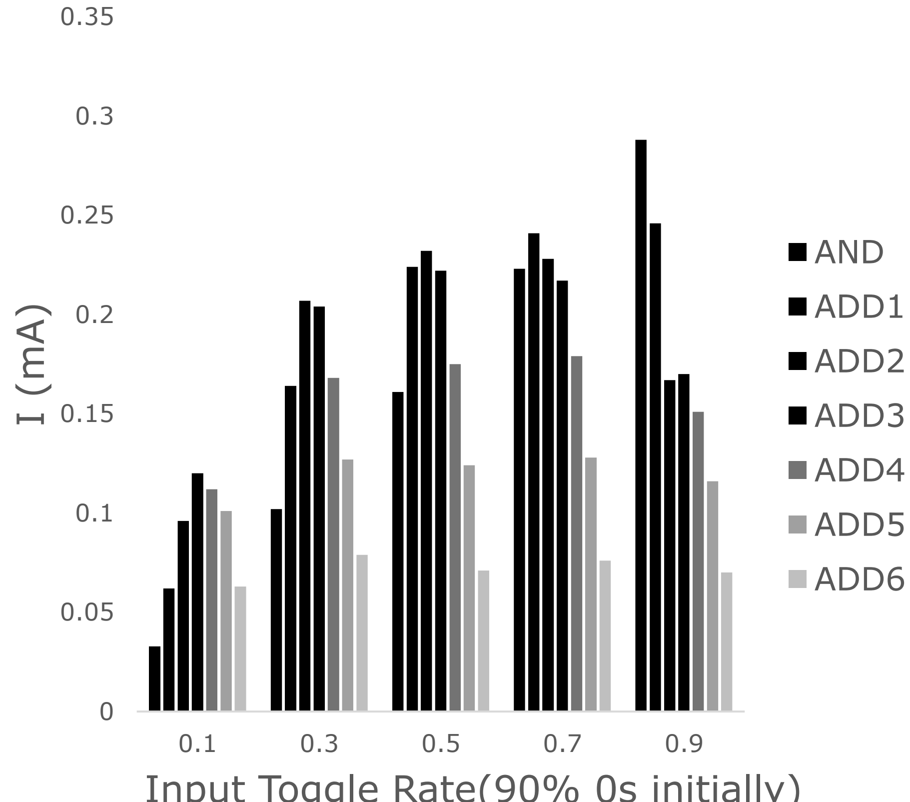
- Power of first few stages greatly depend on input toggle rate, while other stages do not have obvious dependencies



Input Initial condition

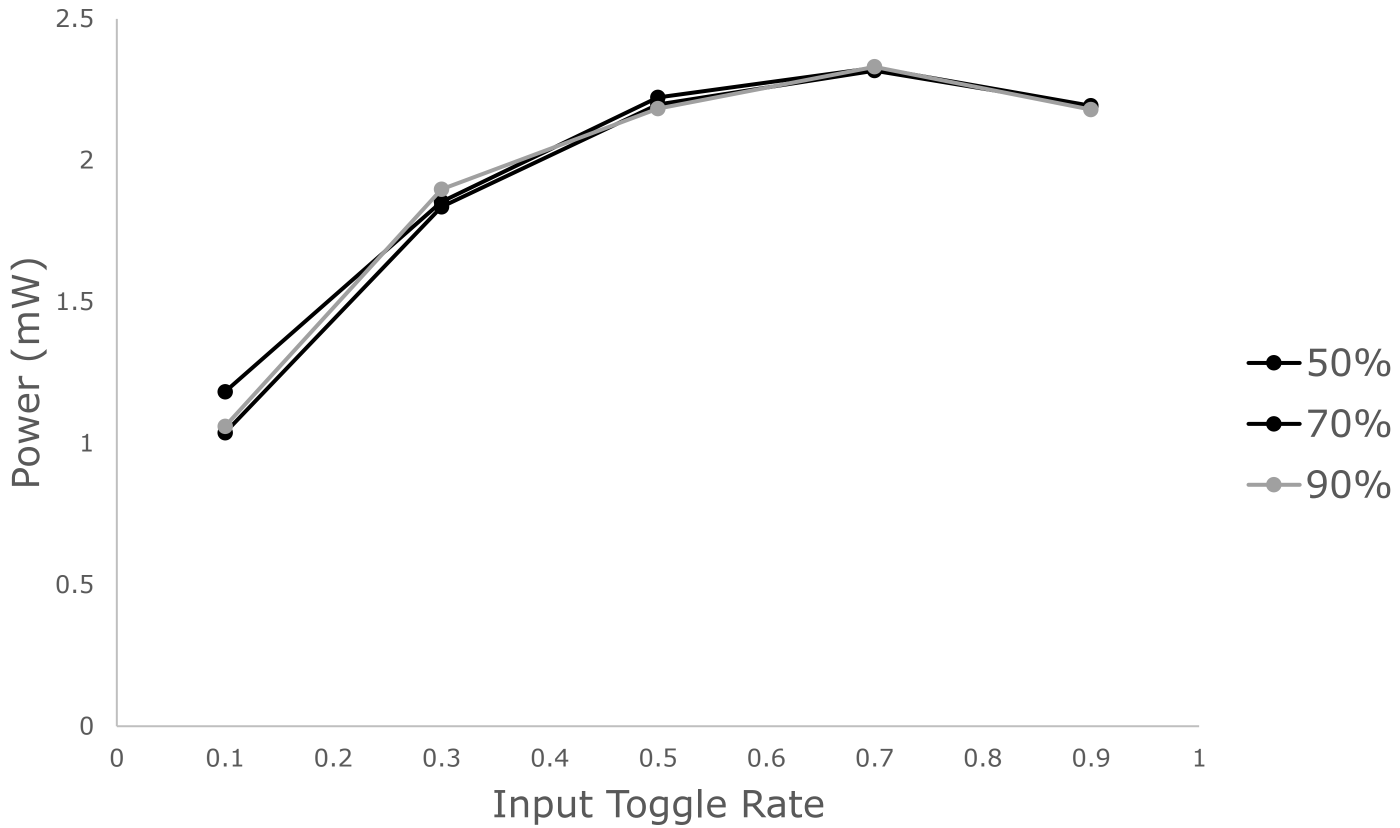
- The power distribution in all stages are same as 50% 0's input pattern's result





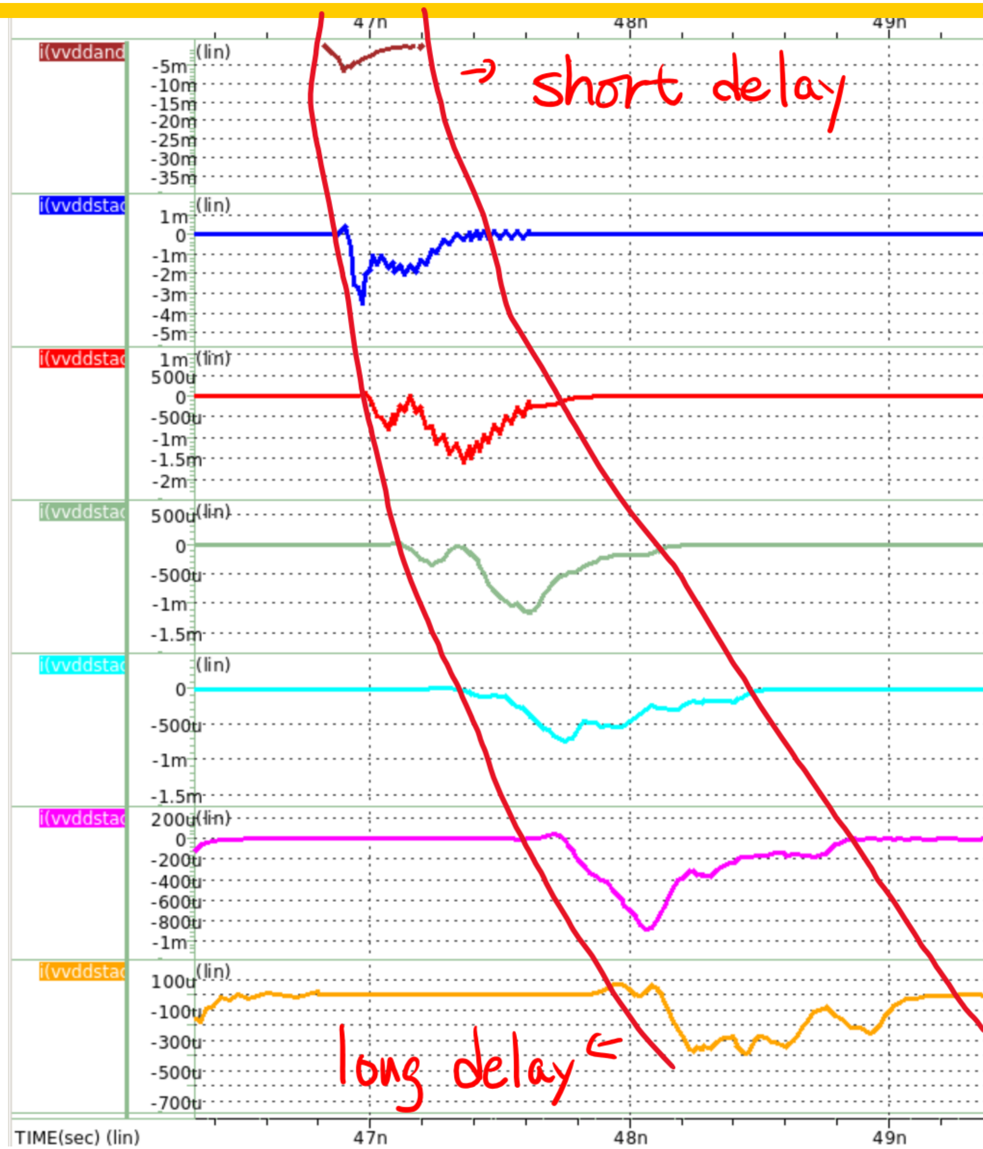
Input Initial condition

- The proportion of 0's in first input pattern does not affect the power of MAC



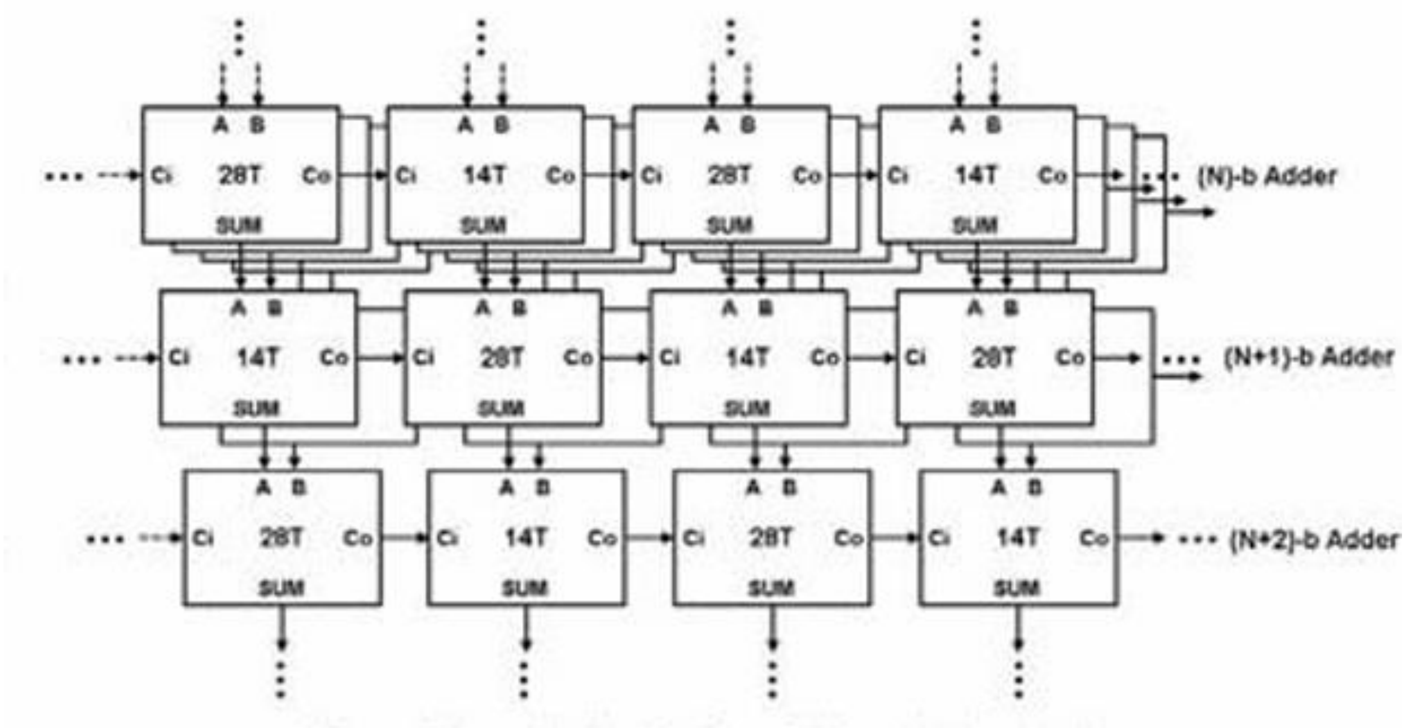
Timing diagram?

- AND 7 mA
- VDD1 3.5 mA
- VDD2 1.6 mA
- VDD3 1.2 mA
- VDD4 0.7 mA
- VDD5 0.9 mA
- VDD6 0.3 mA



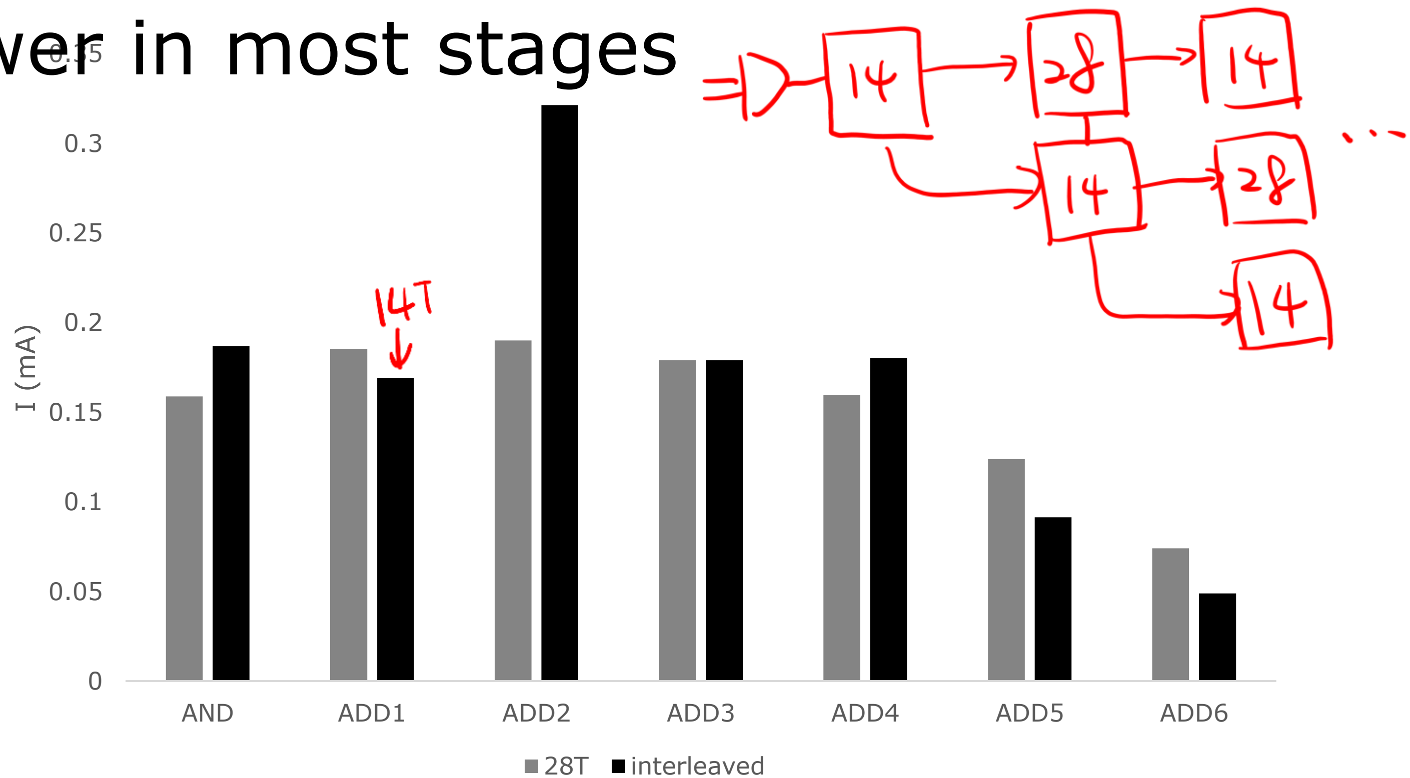
Interleaved FA

- Interleave 28T FAs along with 14T Fas
  - 30% smaller than the original design
  - But require longer cycle time (2.9ns)
  - Consumes 18% more power in average



Interleaved FA

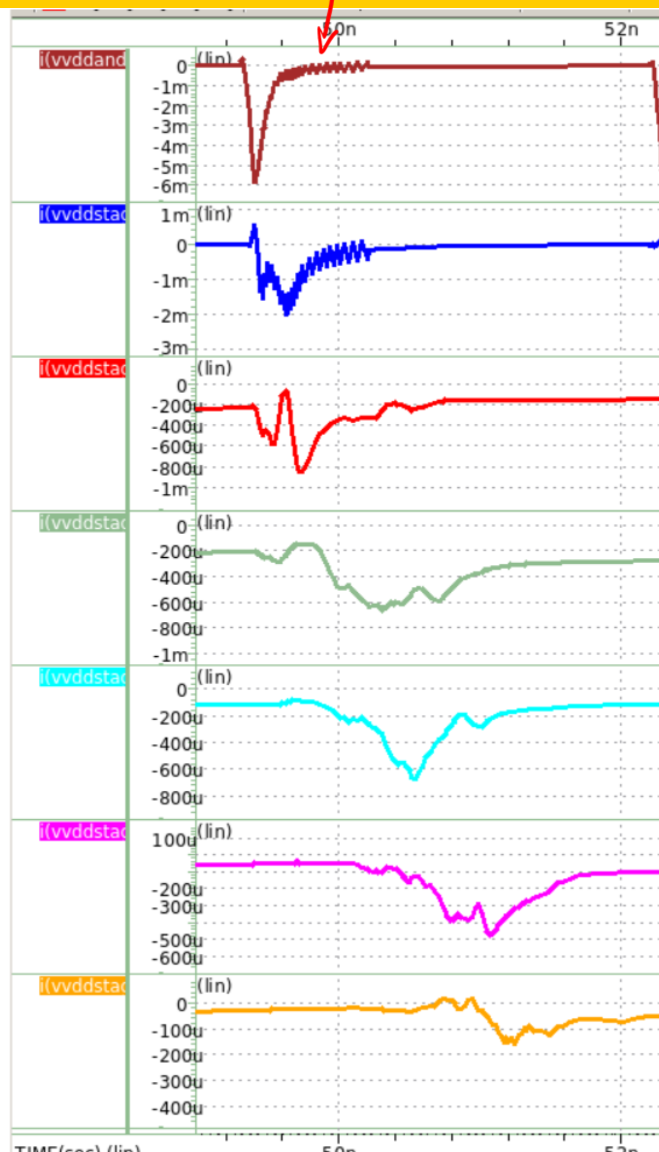
- Energy Efficiency and Area Efficiency
  - 0.26 TOPS/W, 2.97 TOPS/mm^2
- Interleaved FA structure consumes more power in most stages



Interleaved FA

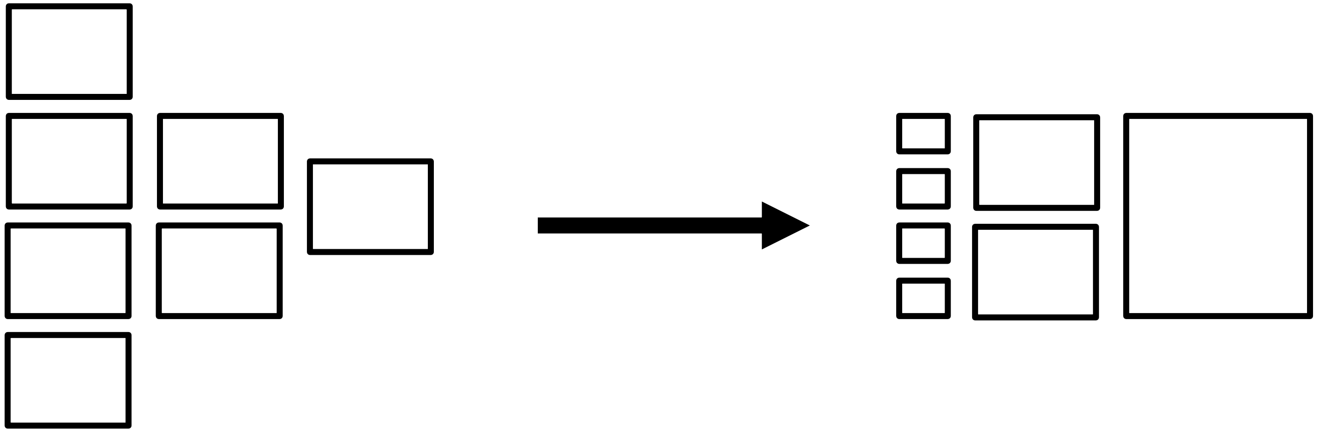
$I_{peak} \downarrow$, but $I_{avg} \uparrow$

- AND 6(7) mA
- VDD1 2(3.5) mA
- VDD2 0.8(1.6) mA
- VDD3 0.6(1.2) mA
- VDD4 0.7(0.7) mA
- VDD5 0.5(0.9) mA
- VDD6 0.15(0.3) mA



FA Sizing

- High Vth for first few stages
  - But I don't know how to modify Vth
- Design first few stages smaller and last 2 stages larger faster less power
  - Alternative approach
  - May require more area

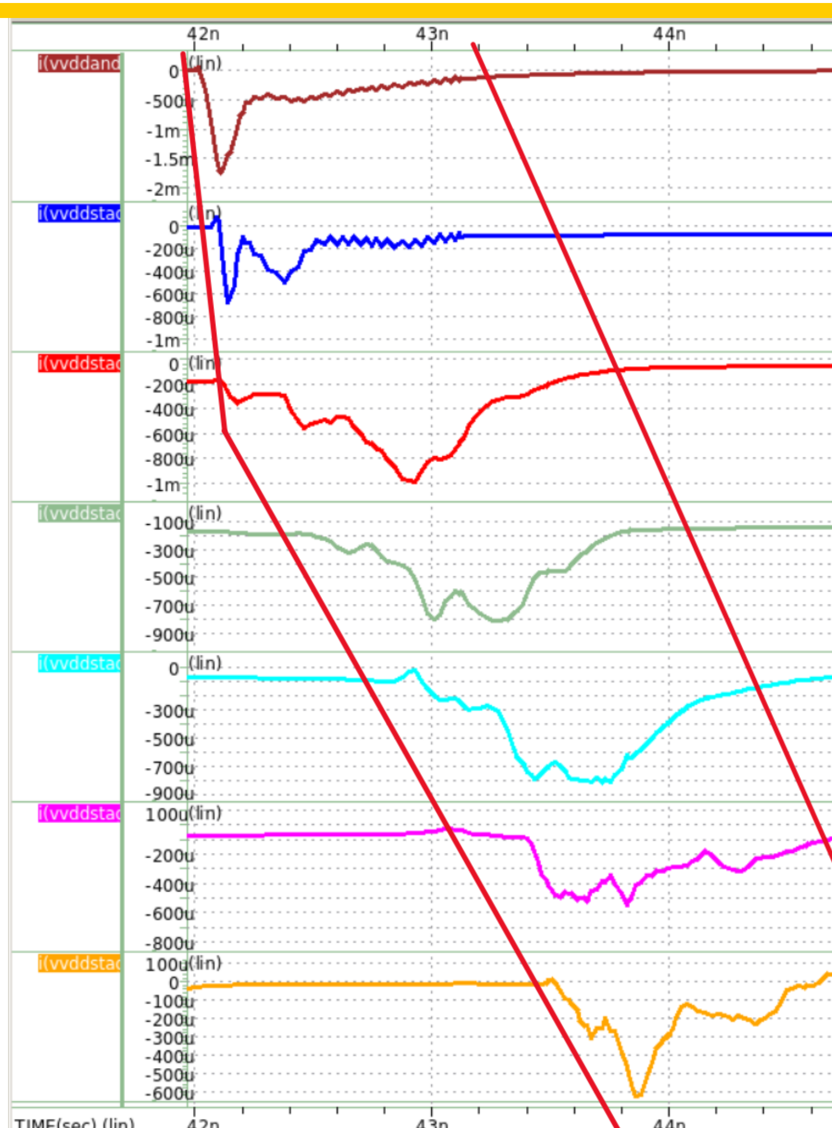


FA Sizing

- Applied on the interleaved design
  - Energy efficiency: 0.26 -> 0.29 TOPS/W
  - Area efficiency: 2.97 -> 2.81 $TOPS/mm^2$
- Slight improvement on energy efficiency
  - Area efficiency dropped, but not much

FA Sizing

- AND 1.8(6) mA
- VDD1 0.7(2) mA
- VDD2 1(0.8) mA
- VDD3 0.8(0.6) mA
- VDD4 0.8(0.7) mA
- VDD5 0.6(0.5) mA
- VDD6 0.6(0.15) mA

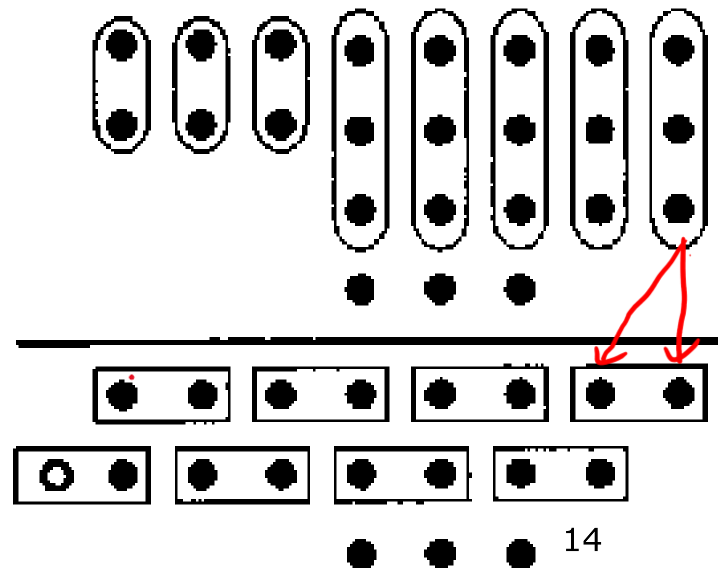


FA Sizing

- Apply on the original design
- Energy efficiency
  - 0.33 -> 0.36 TOPS/W
- Area efficiency
  - 2.30 -> 1.27 TOPS/mm^2 cannot furthur lower width
  - Since minimum width is 250nm, increase L to lower drain current -> more area -> bad result

Adder Tree

- Calculate 64 1-bit input with groups of 2 input adders
  - FA has 3 degree of freedom at input but we only utilize 2 of them
  - We should use the FA fully
- Carry save adder design
  - Multiple input



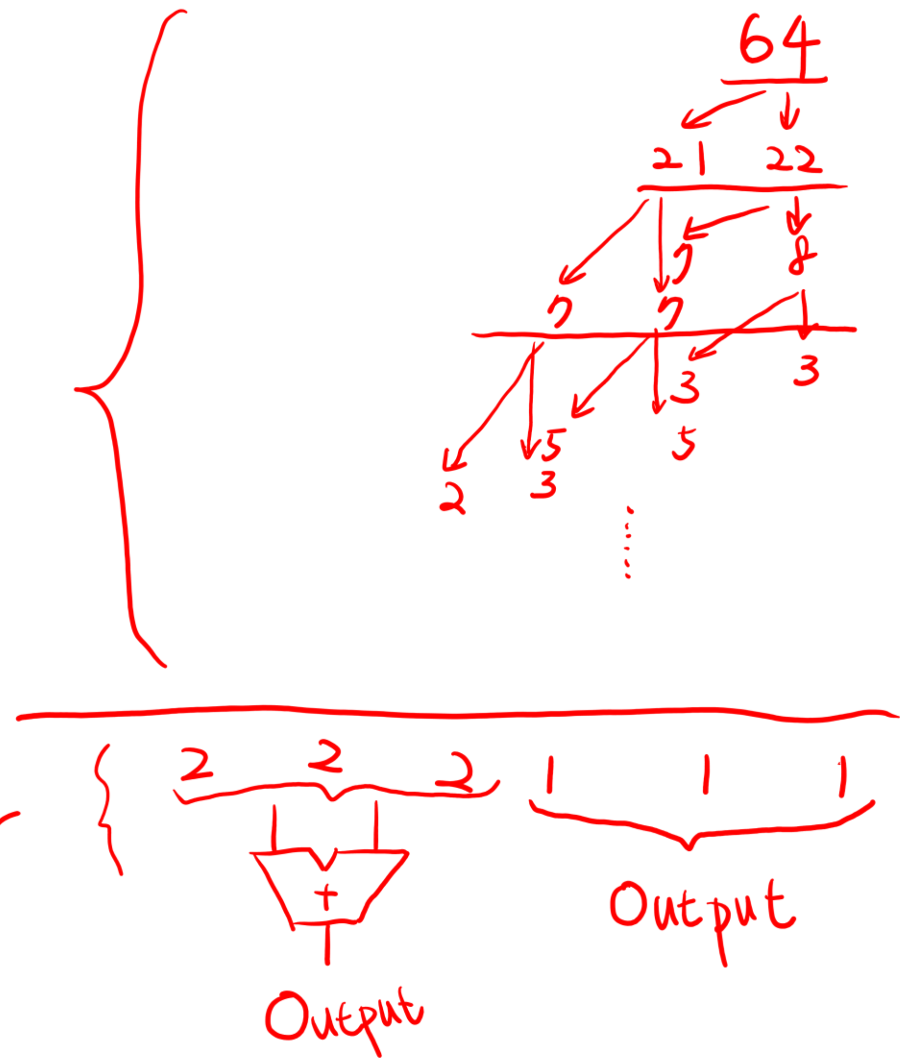
Inputs to the second stage

Result of the second stage

Adder Tree

Carry-save adder

Ripple-carry adder

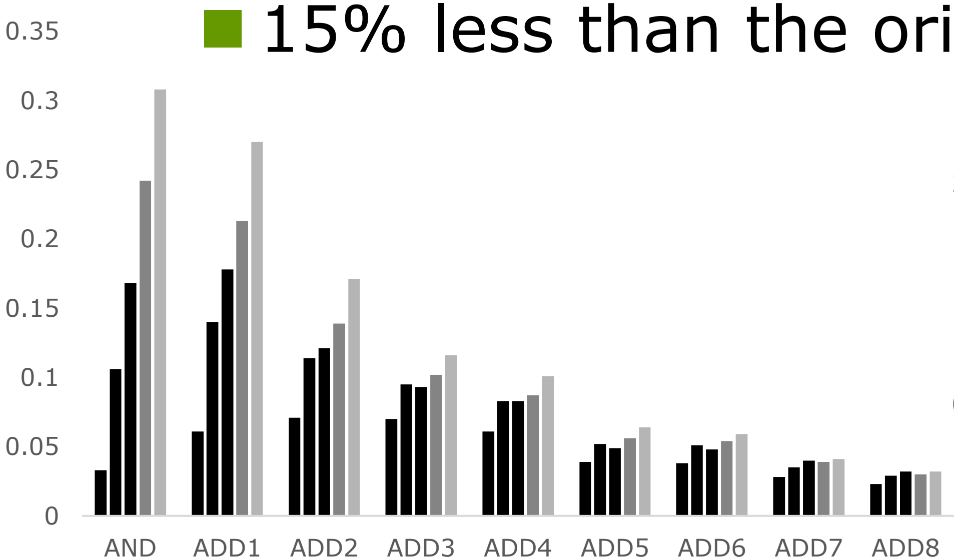


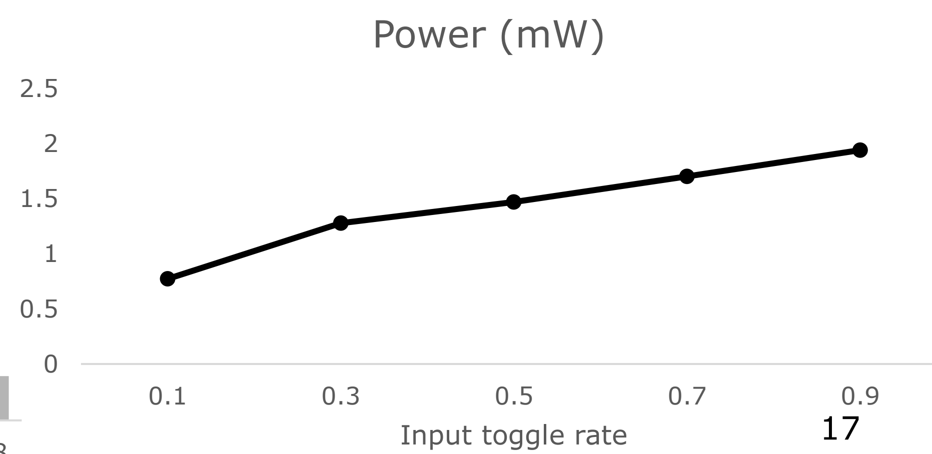
Carry Save Adder Tree

- For regular 64b to 6b adder tree
  - Require 120 Full adders
  - Ripple carry adders propagate carries in every stage
  - Long critical path
- Carry save adder design
  - Only 67 full adders is used → 64 as CSA
  - Only need to propagate carry in the last stage
  - Relatively short critical path

CSA Tree Performance

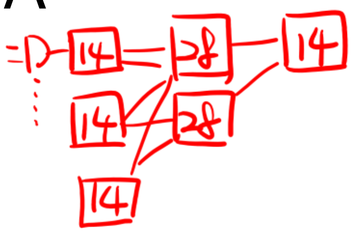
- 0.45 TOPS/W
  - 36% improvement w.r.t. the original design
- 4.33 TOPS/mm2
  - 88% improvement w.r.t. the original design
- 2.3 ns cycle time
  - 15% less than the original design



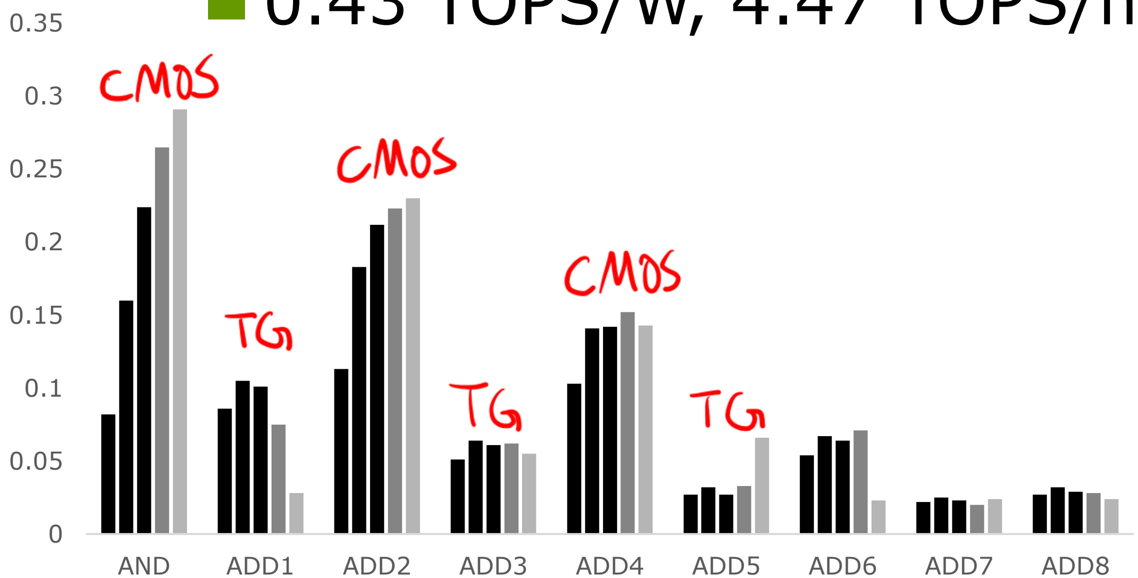


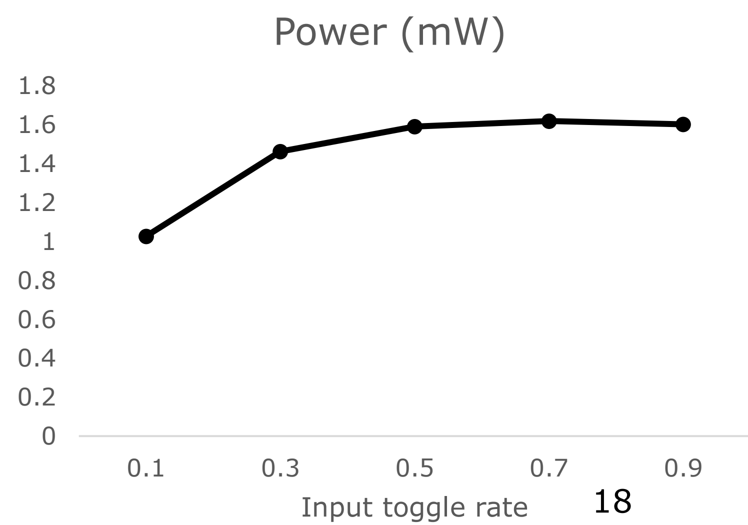
CSA Tree with interleaving

- 1^{st} stage: 14T, 2^{nd} stage: 28T ……
  - But ensure output are driven by 28T FA
  - 37/67 FAs are replaced by 14T FA



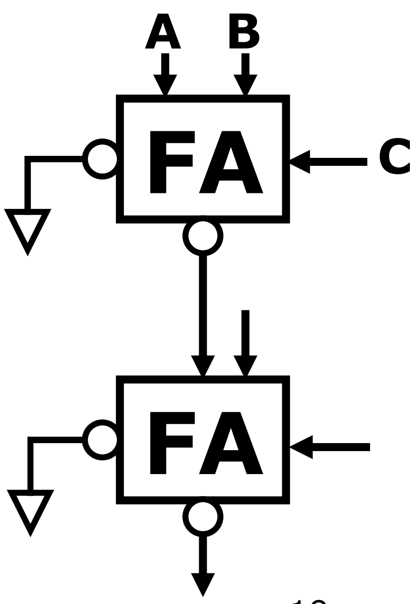
- Performance
  - 0.43 TOPS/W, 4.47 TOPS/mm2





CSA Tree with inverting FA

- More than better
- Use inverting full adders to remove the redundant inverters in FA's output stage
  - 28T -> 24T but 14T remains 14 T
  - May require extra inverters in some stages and at output
  - Driving capability of FA may drop due to no buffer at output
  - Saves some time consumed by INVs but smaller drain currents require more time to charge nodes



Performance of CSA Tree w/ invFA

- 0.55 TOPS/W, 5.14 TOPS/mm2
  - 22% improvement in energy efficiency
  - 15% improvement in area efficiency
  - Cycle time = 2.3 ns
- What about the design with interleaving
  - 0.51 TOPS/W, 5.35 TOPS/mm2
  - 19% improvement in energy efficiency
  - 19% improvement in area efficiency
  - Cycle time = 2.9ns

Summary of All Designs

Adder Type	Additional Feature	Energy Efficiency	Area Efficiency	Cycle Time
Ripple Carry	No	0.33	2.30	2.6
Ripple Carry	Interleaving	0.26	2.97	2.9
Ripple Carry	Sizing	0.36	1.27	2.7
Ripple Carry	Interleaving Sizing	0.29	2.81	3.0
Carry Save	No	0.45	4.33	2.3
Carry Save	Interleaving	0.43	4.47	2.9
Carry Save	Inverting	0.55 +66%	5.14 +123%	2.3
Carry Save	Interleaving Inverting	0.51	5.35 +133%	2.9

Summary

- Ripple carry adder tree
  - Longer delay, large area, and higher power
  - Easy to implement and debug
  - Compatible with input size scaling
- Carry save adder tree
  - Shorter delay, small area, and lower power
  - Hard to implement and debug
  - Cannot be made to be compatible with input size scaling easily (I don’t know how to)
- Maybe even better? I tried but performance are poor
  - Use abc+yosys to further optimize adder tree

Summary

- Standard (do nothing)
  - Old friends are always reliable
- Interleaving
  - Energy efficiency + Speed <-> Area efficiency
- ~~Vth scaling~~ Transistor Sizing
  - As pathetic as my electric circuit design grade
- Inverting Full Adders
  - Increasing both energy and area efficiency
  - Very very good idea