Think Blue…

A 5GHz+ 128-bit Binary Floating-Point Adder for the POWER6 Processor

Xiao Yan Yu[1], Yiu-Hing Chan[1],
Brian Curran[1], Eric Schwarz[1],
Bruce Fleischer[2], Michael Kelly[1]

[1] IBM Systems and Technology Group, Poughkeepsie, USA
[2] IBM T. J. Watson Research Center, Yorktown Heights, USA

Presentation Outline

- Motivation
- Background
- Our Adder Implementation
- Comparisons with Conventional Designs
- Conclusion
- Questions?

Motivation

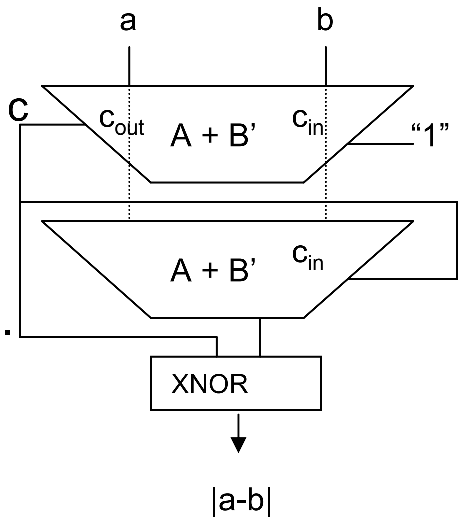
- **Current state-of-the-art high-performance adder designs use dynamic circuits to achieve the performance. They result in significant power consumption in 65nm technology. Design structure space must be fully explored before a design choice is made.**
- **Wire delays contribute large percentage of cycle time in 65nm technology. Designs with long wires will suffer from significant increase in area, delay and power. Performance impact due to physical implementation must be analyzed at design time in order to guarantee the optimality of a design.**

Background

- **Key Features of our adder:**
  - Fabricated in IBM's 65nm SOI technology
  - It is part of a 7-cycle multiply-add pipeline using end-around carry technique
  - It provides the best overall floating point performance
  - Implementation uses all static circuits with nominal V_t devices

Why do we need end-around carry computation?

- **The magnitude of the operands is not known prior to this stage.**
- **The adder needs to produce positive magnitude result during subtraction.**
  - If operand a > b, | a – b | = a – b = (a + b’ + 1)
  - If operand b > a, | a – b | = b – a = -(a – b) = -(a + b’) – 1 = (a + b’)’
  - During subtraction of a – b (2’s complement), carry-out c is 1 when a > b and 0 when a < b
    When a > b, a – b = (a + b’ + c) XNOR c
    When b > a, b – a = (a + b’ + c) XNOR c
  - Theoretically, the adder can be constructed as:
  - Practically, much better implementation is feasible.



End-Around Carry Computation

- **Assume the adder is divided into four groups. During subtraction, the carry for each group can be expressed as where 0th bit is MSB group and 3rd bit is LSB group:**

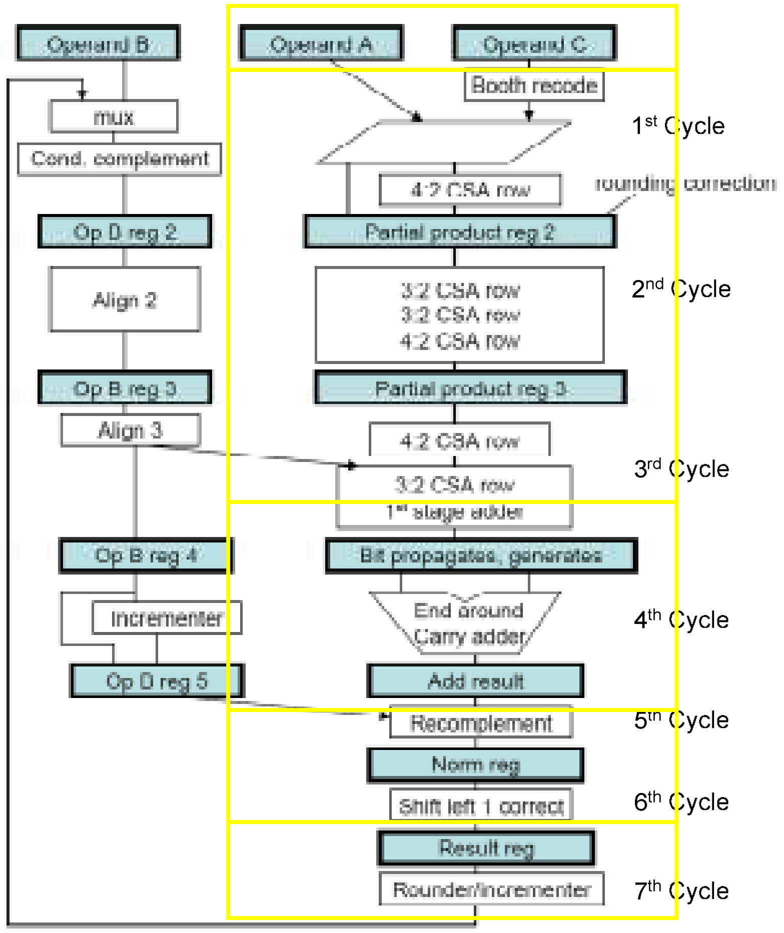
$$\boxed{\begin{aligned}
C_0 &= G_0 + P_0G_1 + P_0P_1G_2 + P_0P_1P_2G_3 + P_0P_1P_2P_3 \\
C_1 &= G_1 + P_1G_2 + P_1P_2G_3 + P_1P_2P_3G_0 + P_0P_1P_2P_3 \\
C_2 &= G_2 + P_2G_3 + P_2P_3G_0 + P_2P_3P_0G_1 + P_0P_1P_2P_3 \\
C_3 &= G_3 + P_3G_0 + P_3P_0G_1 + P_3P_0P_1G_2 + P_0P_1P_2P_3
\end{aligned}}$$

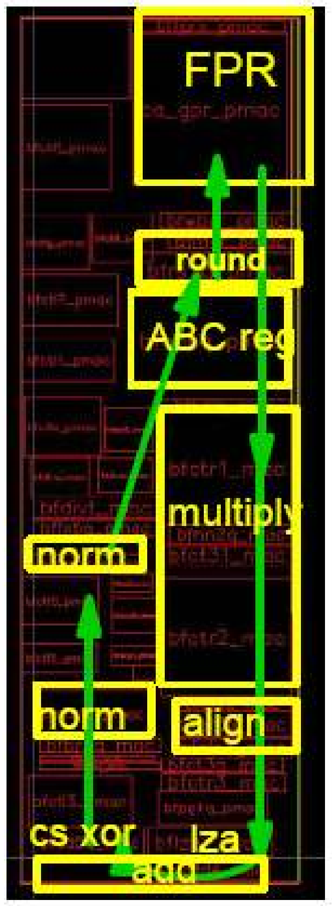
- **During addition P_3 is set to 0 and conventional carries are computed.**

Adder Architecture

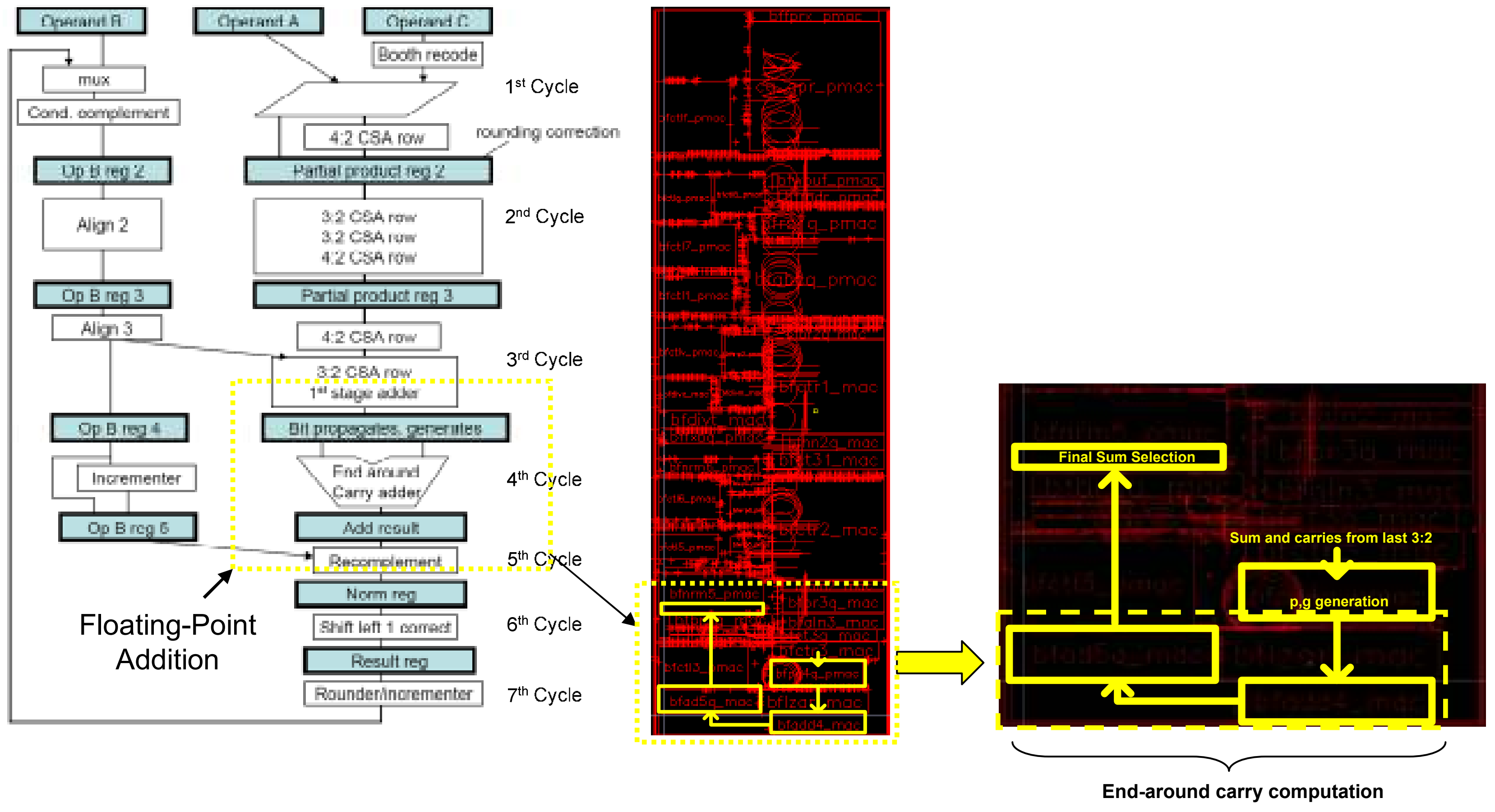
- **In order to minimize communication overhead within the floating-point unit, we created an "O" shaped floorplan. This limits the internal wire resources within the adder.**
  - Our goal is to create an adder with best floating-point performance, <u>not stand-alone performance!</u>
- **An non-uniformly sparse adder scheme was used based on the given wire resource to optimize performance.**
  - This allows us to route critical signals with better wire width and space without allocating dedicated routing areas

Organization of POWER6 Floating-Point Dataflow





Organization of POWER6 Floating-Point Dataflow cont.



Adder Block Diagram

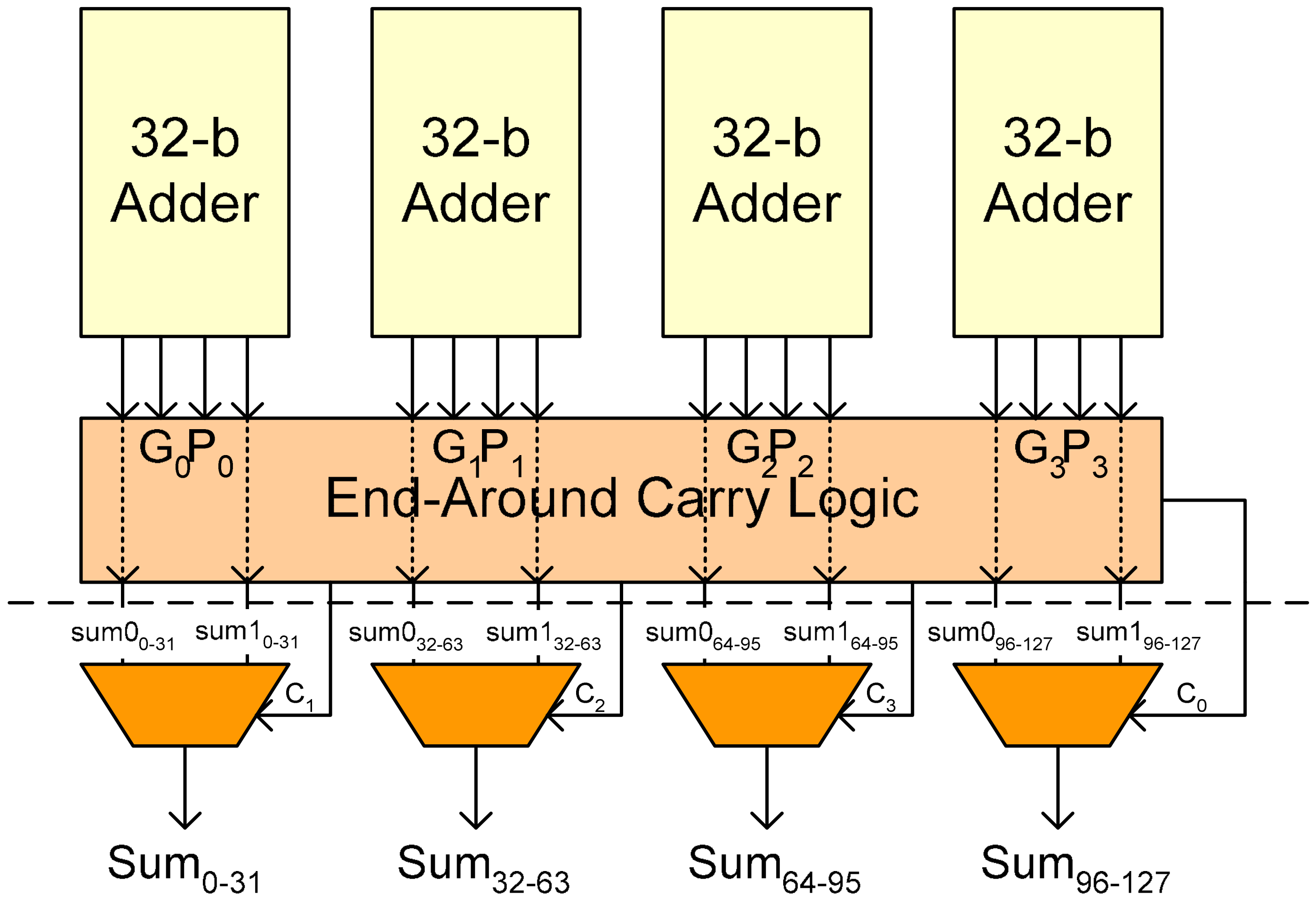
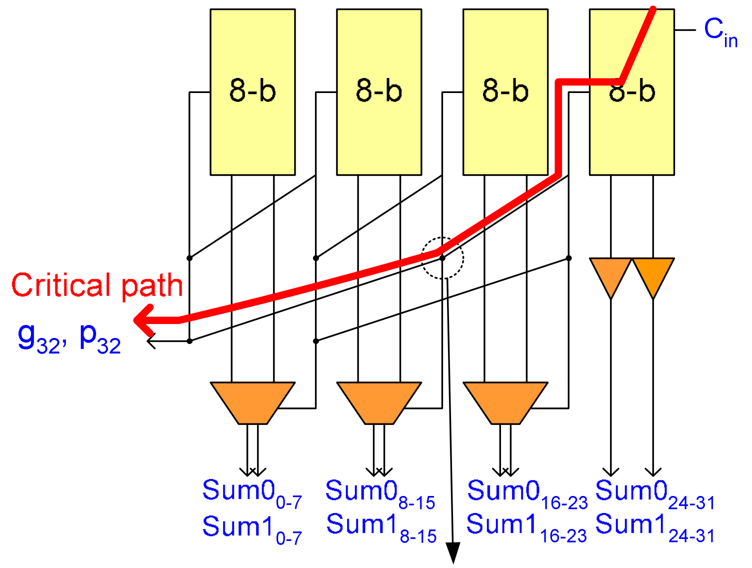


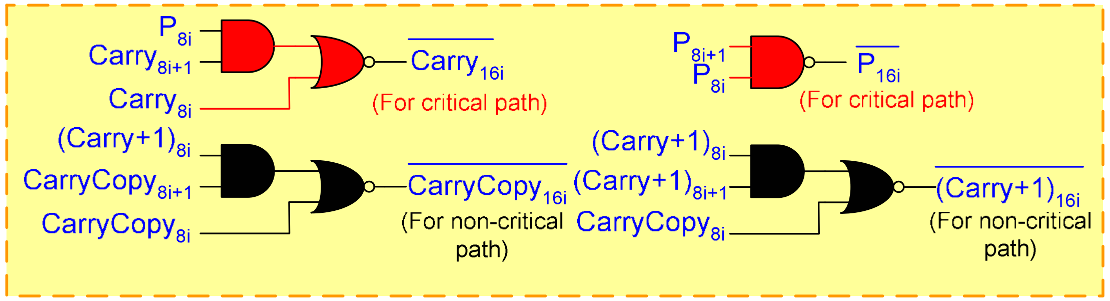
Diagram of the 32-b block



8-b
8-b
8-b
8-b
C_{in}
Critical path
g_{32}, p_{32}
$Sum0_{0-7}$ $Sum1_{0-7}$
$Sum0_{8-15}$ $Sum1_{8-15}$
$Sum0_{16-23}$ $Sum1_{16-23}$
$Sum0_{24-31}$ $Sum1_{24-31}$

$(Carry+1)_i$ = $Carry_i$ when $C_{in}=1$
Sum0 = sum when $C_{in}=0$
Sum1 = sum when $C_{in}=1$

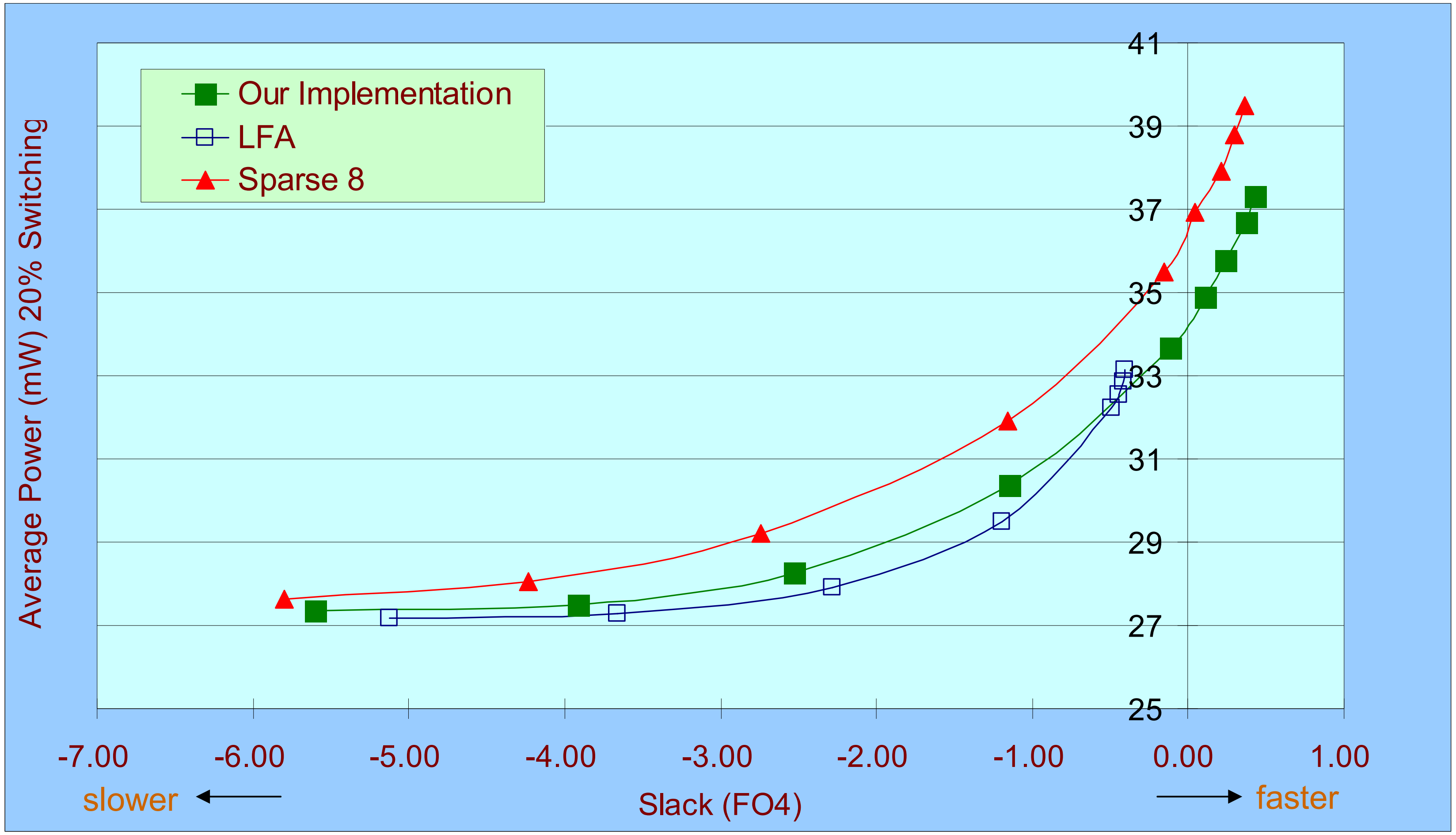
Since $(Carry+1)_i = Carry_i$ or P_i
Therefore,
$P_i \subseteq (Carry+1)_i$
and $(Carry+1)_i$ can be used instead of P_i on the non-critical paths.



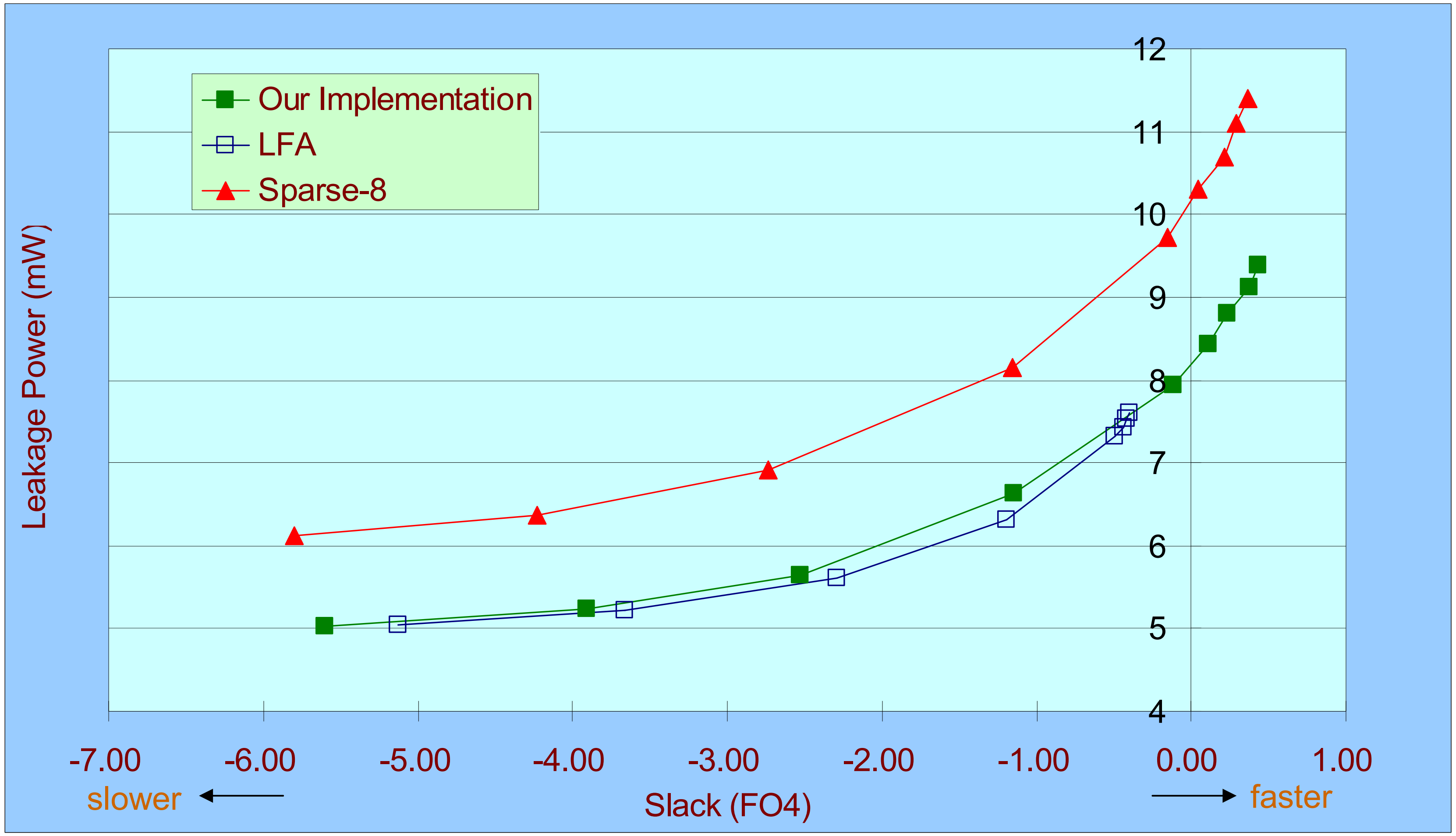
Comparison Setup

- **We have compared our design against the Ladner-Fischer (LFA) design and a prefix-2 Kogge-Stone adder with sparseness of 8 (Sparse 8).**
- **All designs use only nominal V_t transistors.**
- **The optimization points of each design are obtained by varying power performance tradeoff factor using our in-house tuner, Einstuner with constrained input size .**
- **The performance of each point is simulated using our in-house transistor level static timer, EinsTLT and is presented as a slack number.**
- **The average power dissipation of a design at each performance point is simulated using our in-house power simulator, CPAM.**
- **Each output is loaded with equivalent capacitive load calculated at the unit level of our POWER6 floating-point unit.**

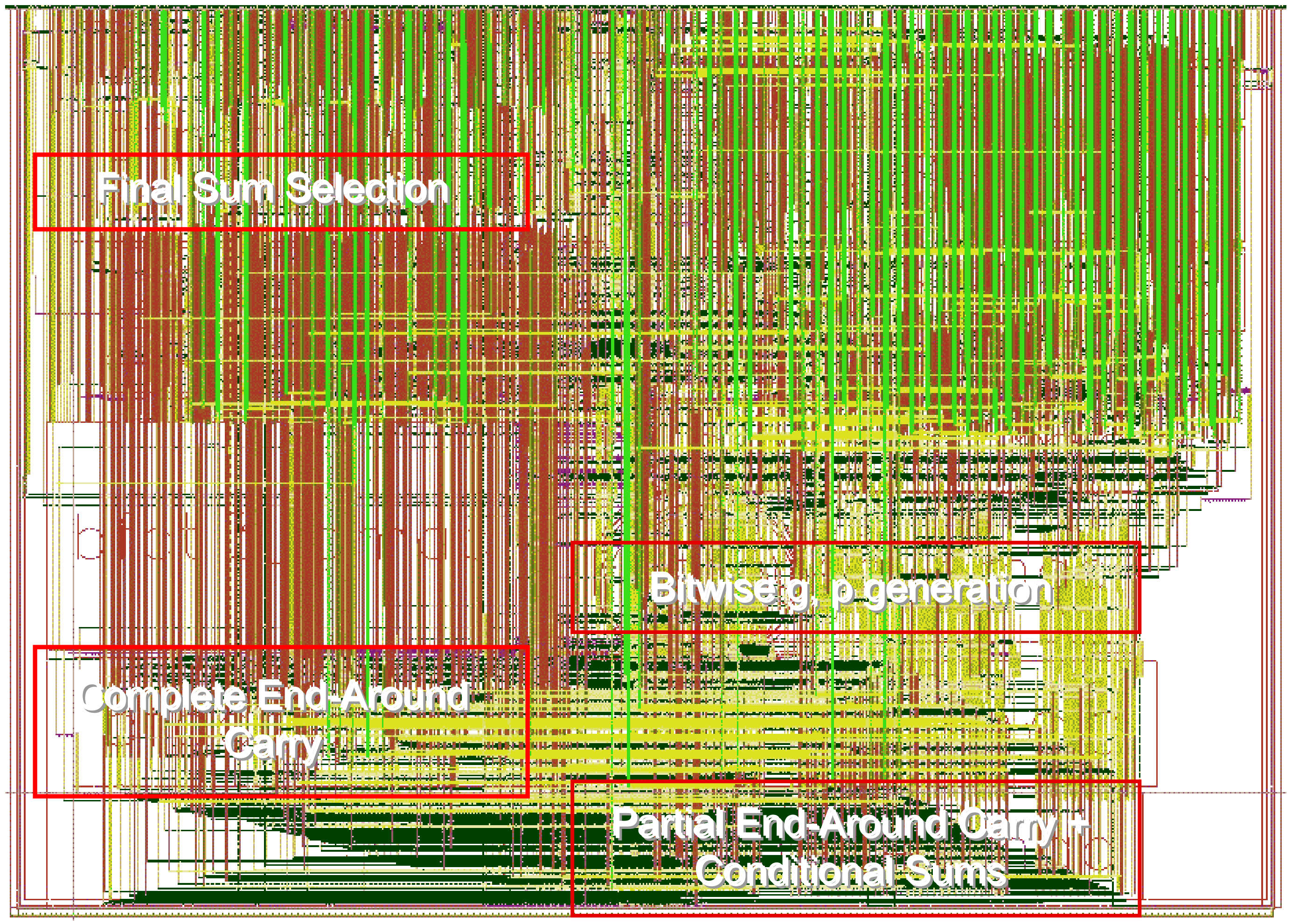
Power-Performance Result
Average Power vs. Performance



Power-Performance Result
Leakage Power vs. Performance



Adder Layout


Final Sum Selection
Bitwise g, p generation
Complete End-Around Carry
Partial End-Around Carry + Conditional Sums

Conclusion

- **A fast 128-bit floating-point adder is implemented and fabricated as part of the POWER6 processor in IBM's 65nm SOI technology.**
- **We used non-uniform sparse Kogge-Stone tree and carefully balanced the prefix tree according to its critical path.**
- **Compared to Ladner Fischer design, our design is 1 FO4 faster and satisfies the stringent performance requirement. Final design consumes 6% more area and 5% more power compared to Ladner Fischer design.**
- **The chip measurements demonstrate operation of this adder beyond 5GHz with 1.1V supply.**

Thank you for listening to my talk. Questions?