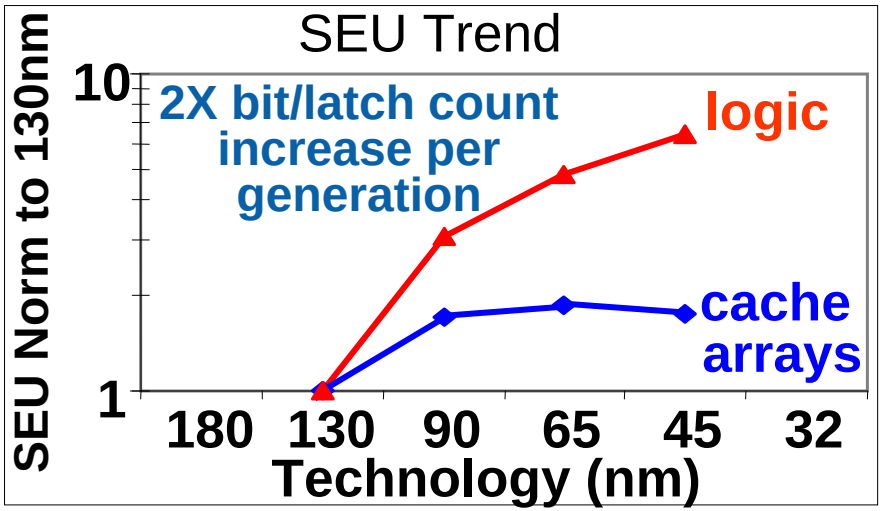
# Commercial/Consumer Systems Brief-in

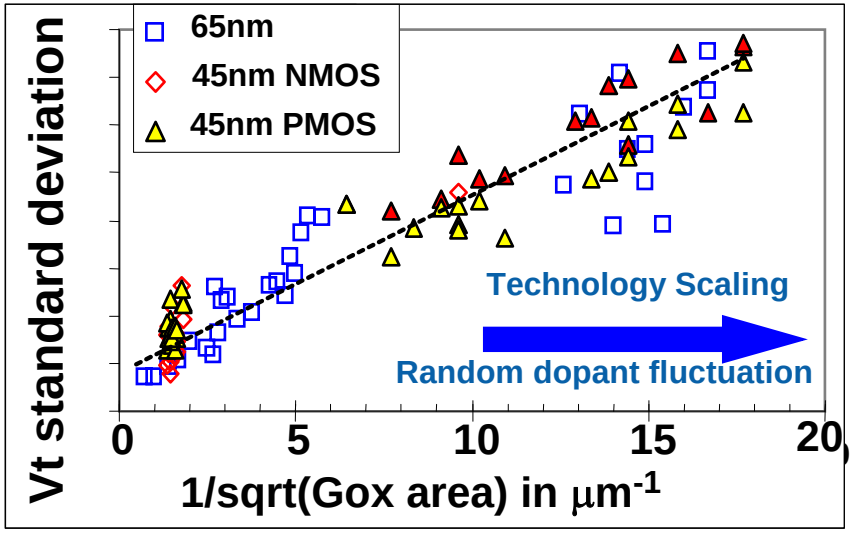
# The Future is:

- Continued feature size scaling
- High variation
- Significant fault rates
- Slower $V_{dd}$ scaling
- Slower clock rate scaling
- Burn-in less of an option
- Power becoming primary limit on integration

# Trends

- Fabrication variation, aging, soft error rate *per chip* all getting worse with scaling
- Commodity processors incorporating more "point solution" reliability mechanisms with each generation
- Argument: time to stop applying band-aids and rethink the problem from scratch




**T=0 Variation Increasing**

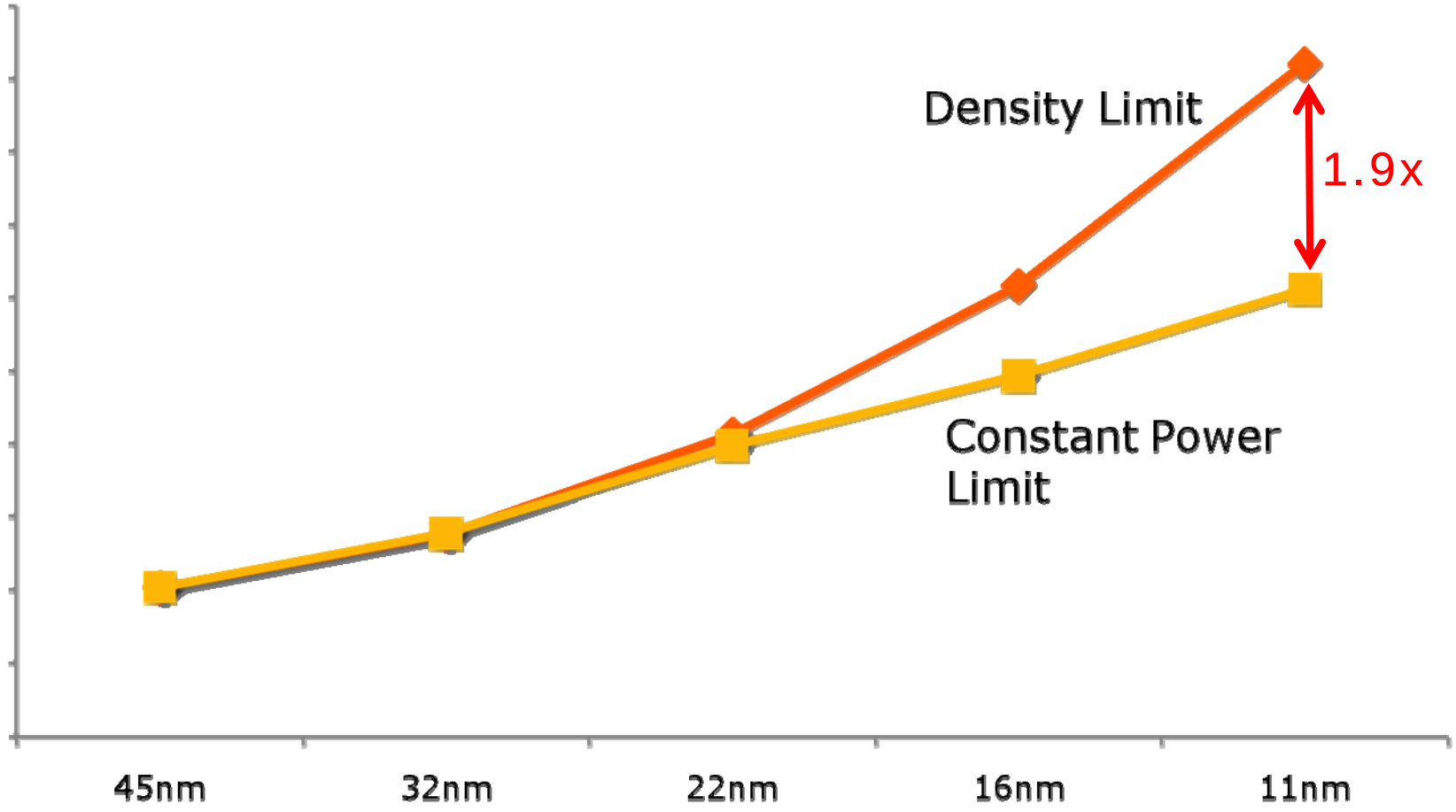
# Technology Outlook

| High Volume Manufacturing | 2008 | 2010 | 2012 | 2014 | 2016 | 2018 | 2020 | 2022 |
|---|---|---|---|---|---|---|---|---|
| Technology Node (nm) | 45 | 32 | 22 | 16 | 11 | 8 | 6 | 4 |
| Integration Capacity (BT) | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| Delay Scaling | >0.7 | ~1? | | | | | | |
| Energy Scaling | ~0.5 | >0.5 | | | | | | |
| Transistors | Planar | 3G, FinFET | | | | | | |
| Variability | High | Extreme | | | | | | |
| ILD | ~3 | towards 2 | | | | | | |
| RC Delay | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Metal Layers | 8-9 | 0.5 to 1 Layer per generation | | | | | | |



4

# How Many Transistors Can I Build?



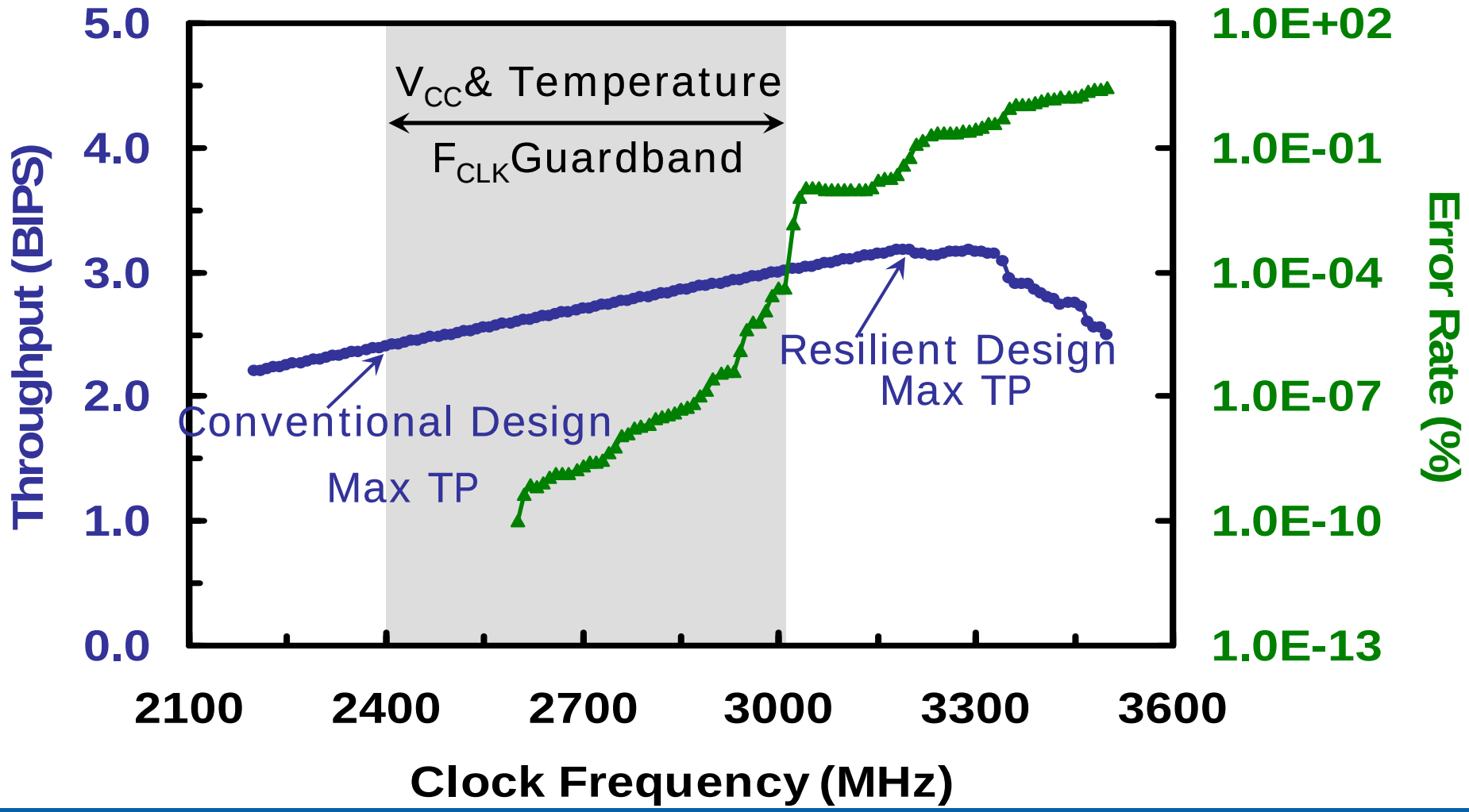
# Thesis

- For commercial/consumer systems, resilience/reliability is the tool that will let us take advantage of smaller feature sizes
  - Tolerate increasing variation/errors without decreasing end-user reliability, product lifetime, confidence
  - Transform energy margins into information margins to reduce power consumption
  - Adapt redundancy and checks to needs of system, application, and user
    - Single-pixel errors in DVD playback better than running out of battery before end of movie
    - Single-digit errors in credit card payment worse than being told you don't have enough battery left



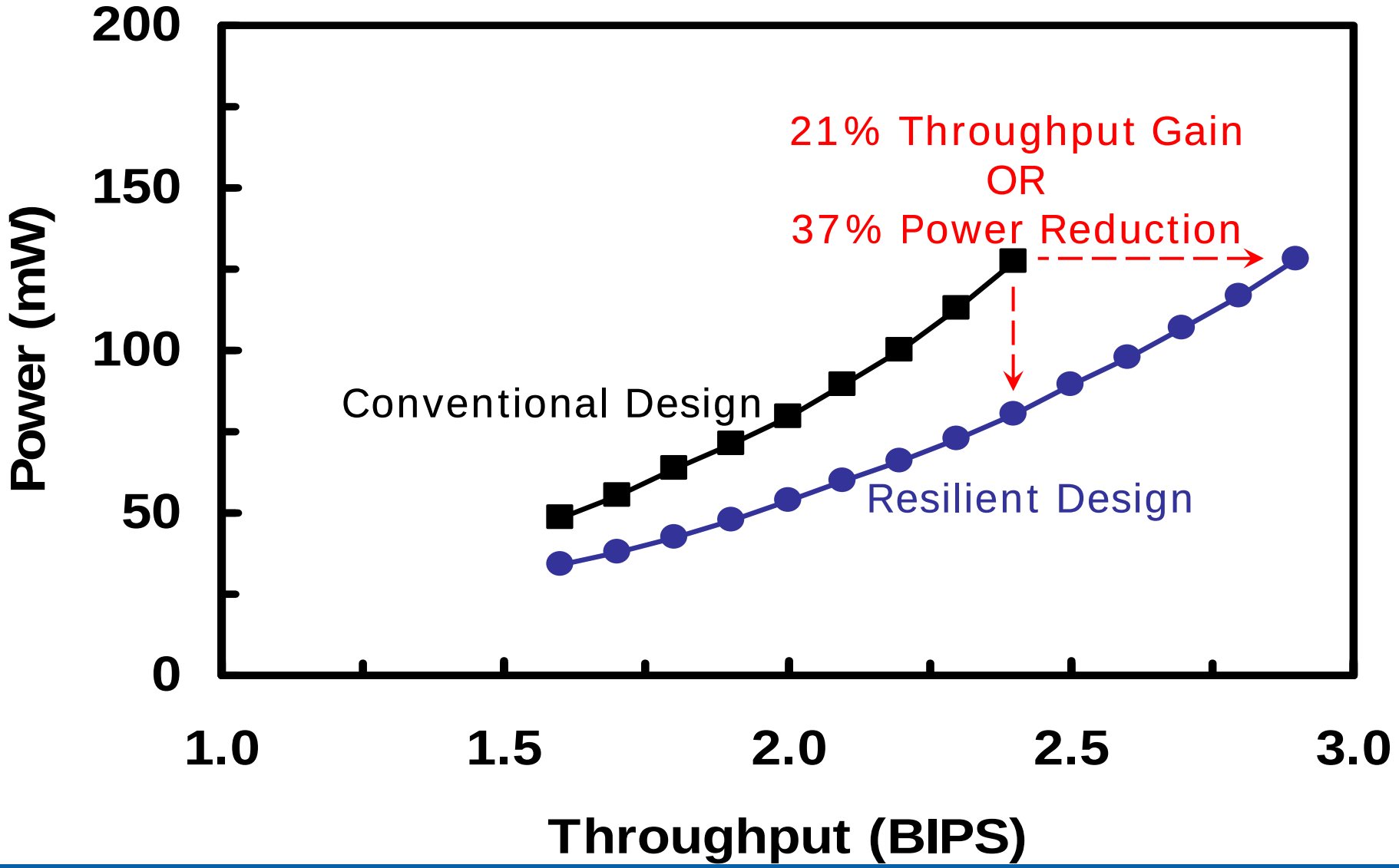
intel

# Example: Operate at Average-Case via Resilience (not Worst-Case)




Source: K. Bowman, et al., *ISSCC*, 2008.

# Power vs. Throughput




Source: K. Bowman, et al.,*JSSC*, 2009.

# Strawman Resilient System

- Adaptive software diagnostics determine system state and aging
- Small set of hardware mechanisms to detect errors
- Reconfigure at coarse grain (ex: core)
  - Finer-grained reconfig. only where it's very cheap (e.g., cache line disable)
- Divide error analysis/recovery/reconfiguration between HW and SW
  - Tune system for error rate that maximizes power efficiency
  - Build cheapest set of HW mechanisms that allow that error rate
  - Push everything else into SW



intel

# Questions for the Group

- What is an appropriate overhead "goal" for each fab. generation? Can we hit 10-20%?
- What information can we provide to the OS, and how much will it cost?
- What can we do to make reliability "tunable" and to make commodity parts better components of systems in other domains?
- When does it not make sense to involve the entire system stack in handling a fault?
  - E.g., power/performance/error rate trade-offs
- How much help can we get/expect from applications?



intel