

Compute Power with Energy-Efficiency

Partnerships, Standards and the ARM GPU Perspective

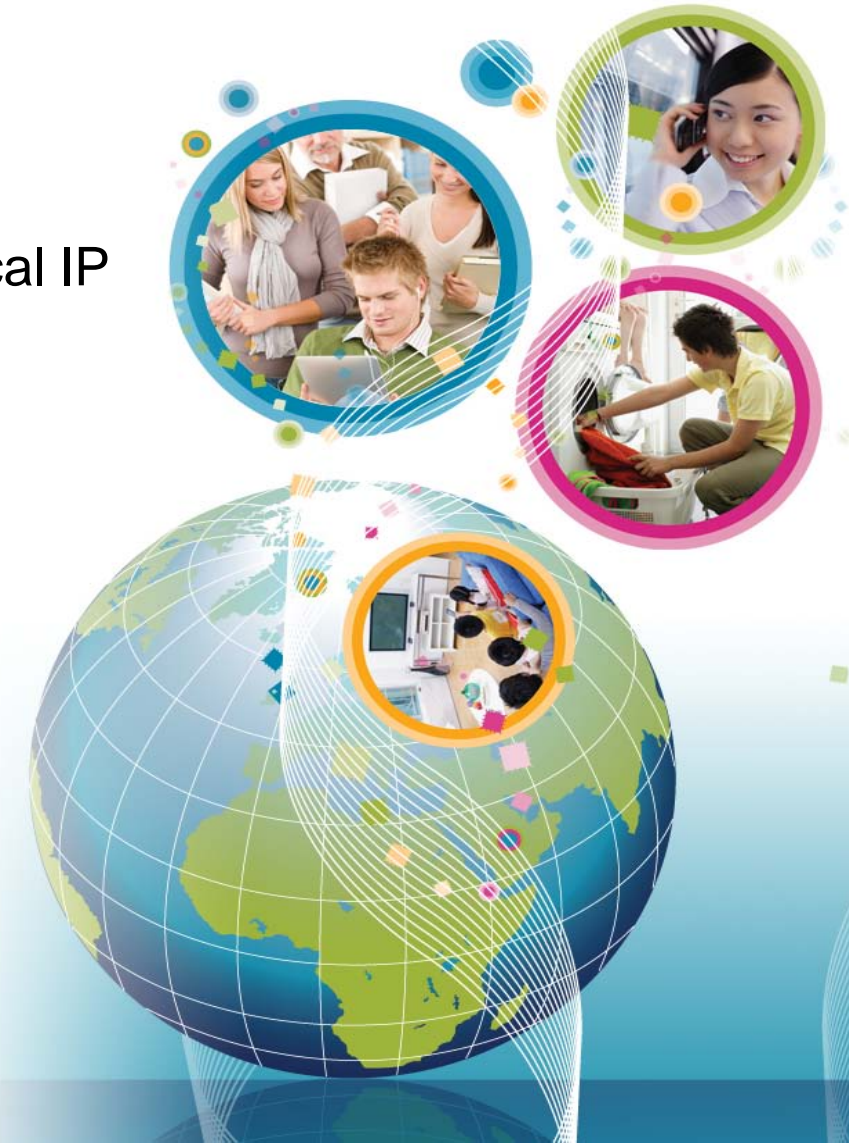
Jem Davies

ARM Fellow, VP of Technology,
Media Processing Division, ARM



What does ARM do?

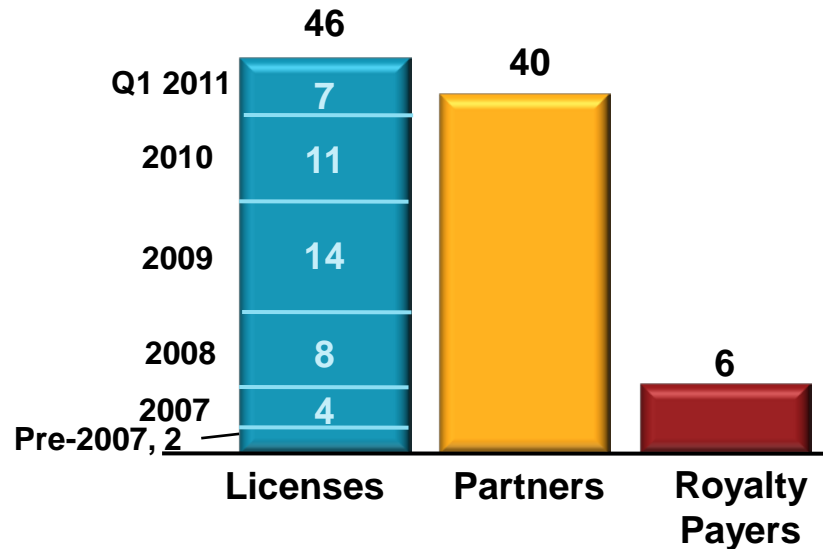
- ARM[®] was founded in 1990
 - Initially designed CPUs
 - Over twenty years, have covered CPUs, GPUs, DSPs, audio processors, video processors, tools, fabric IP and physical IP
- ARM's partners make SoCs containing many heterogeneous compute engines
- Heritage of low-power
 - Not just mobile, but across multiple segments
 - > 6 Bn ARM-based chips in 2010



ARM also does Graphics

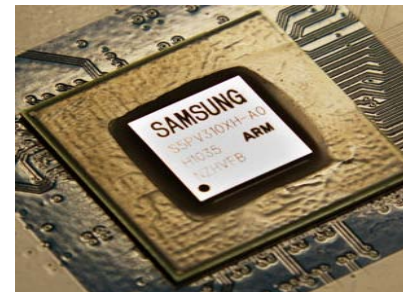
Growing the Media Processor Licensing Base

- 46 licenses for graphics and video
- 7 licenses added in Q1 2011



Growing Shipments in Mobile and Non-Mobile Applications

- More Mali™ based chips shipping into mobile and consumer electronics devices



Samsung announced that Mali-based Exynos delivered 5 times more graphics performance than their previous design

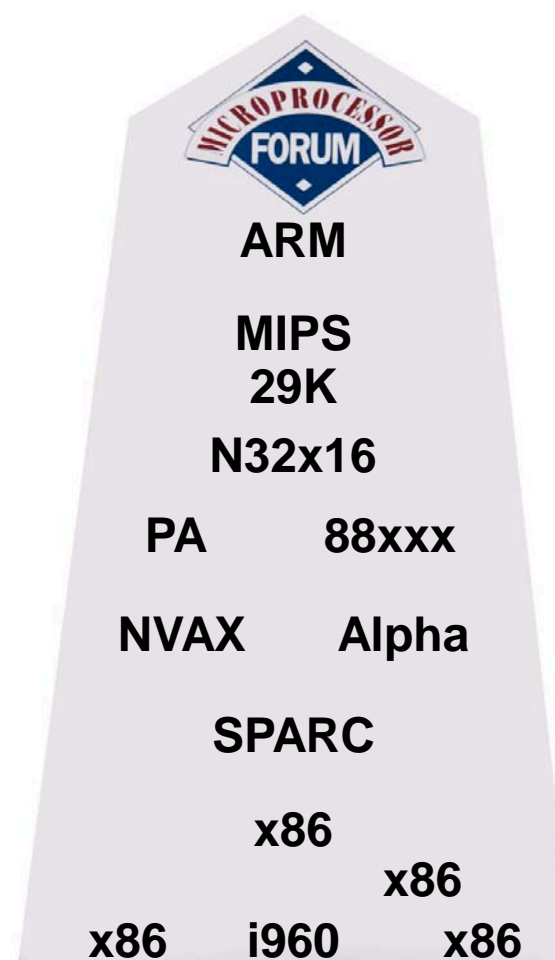
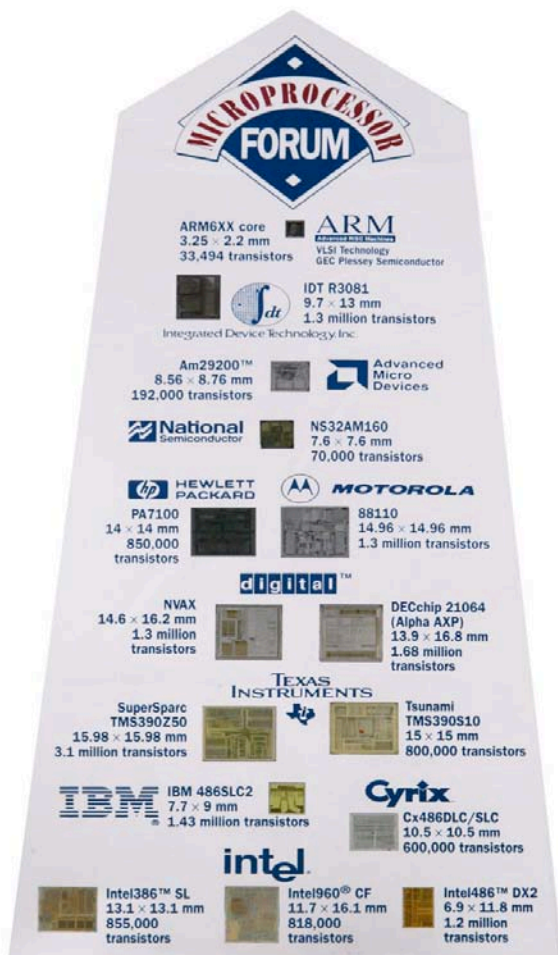
STMicroelectronics announced 10 major STB design wins for Mali-based STi7108



History - the Passage of Time

- What lessons from history can we learn?

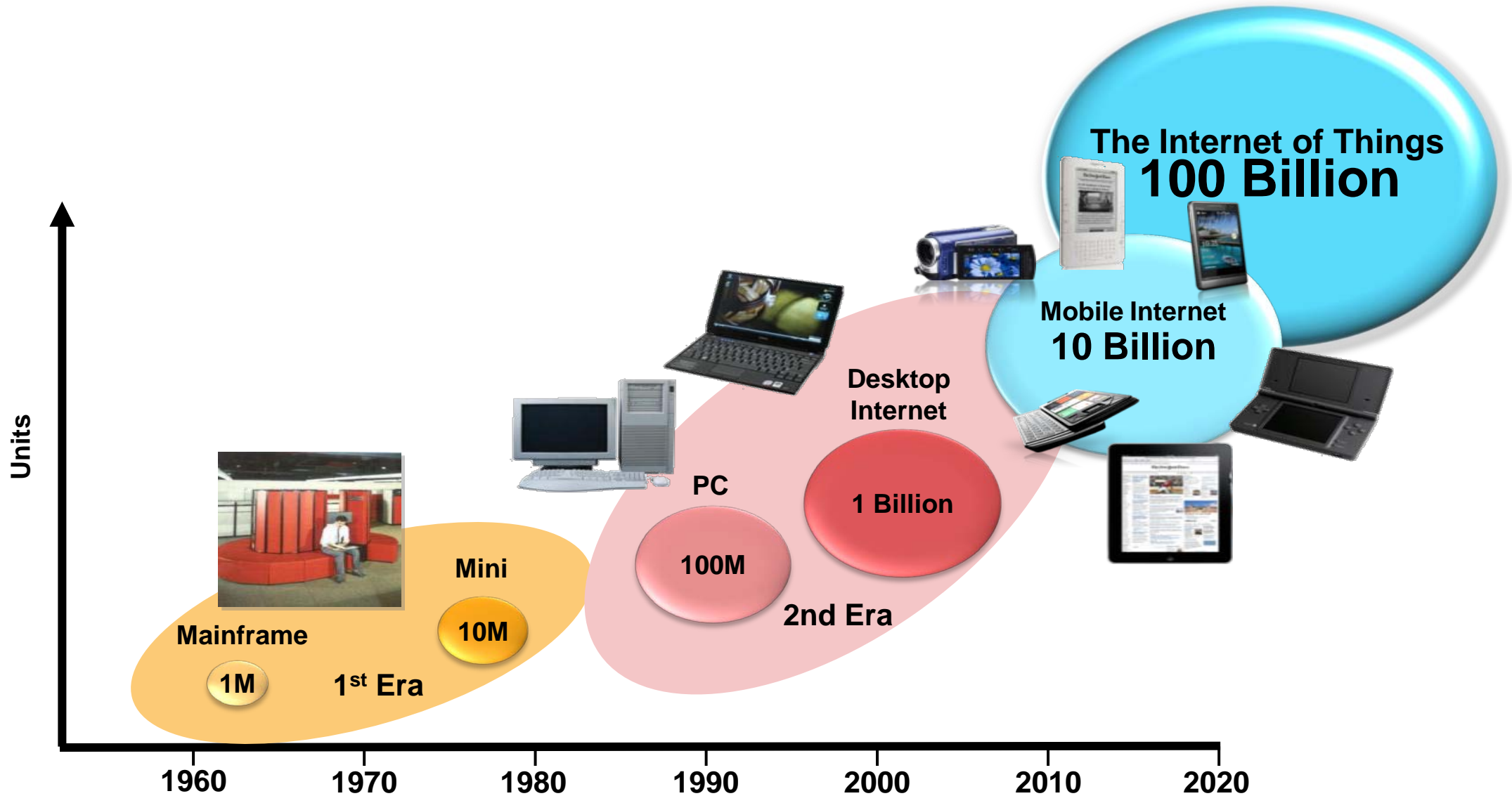
Count the Architectures (11)



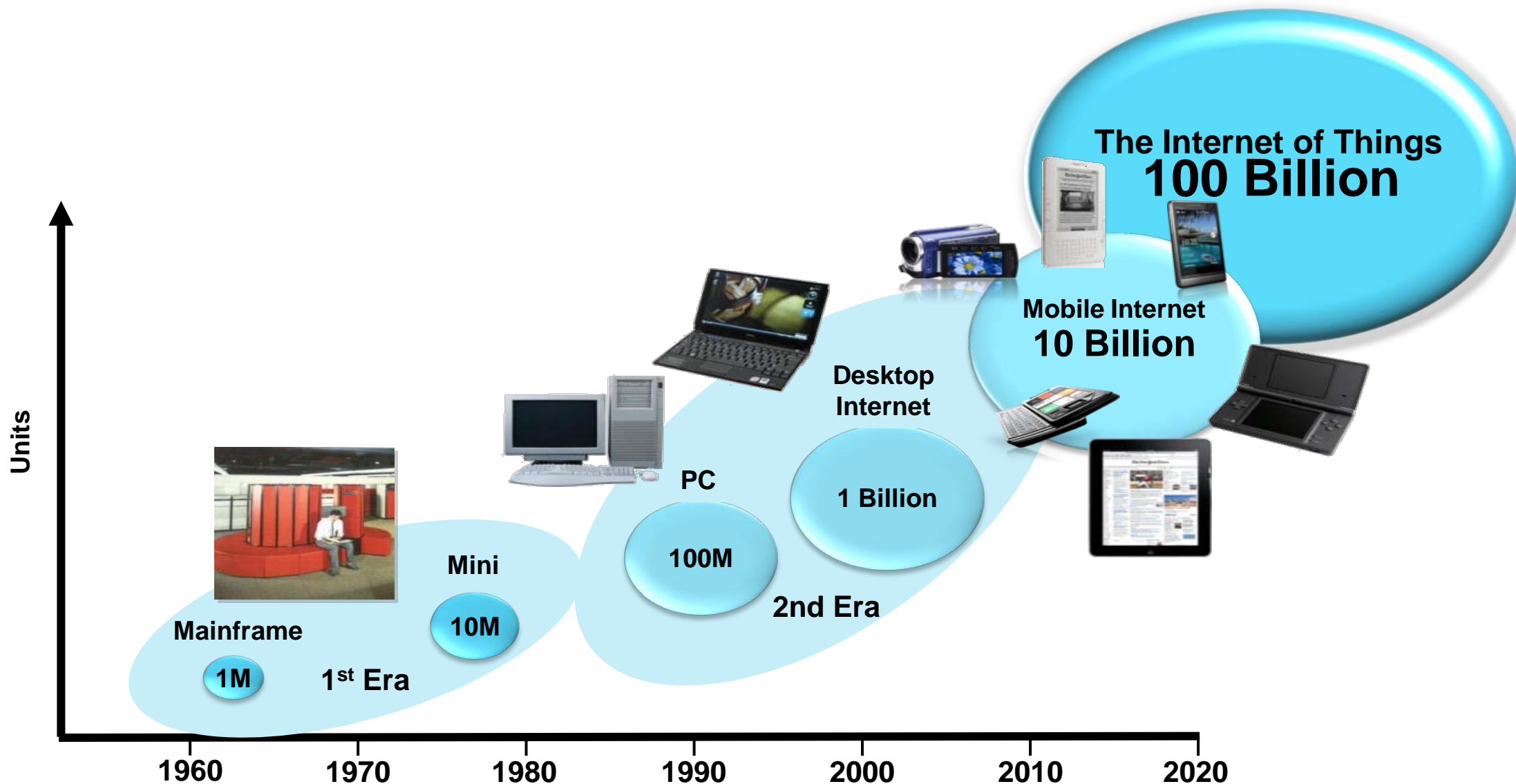
What Were the Lessons?

- Good architectures will succeed
- Power did matter
 - It has come to matter more
- Too much variety can be a bad thing
 - For developers
- The Ecosystem **Really** matters
 - Developers need to be able to support a range of platforms

The Eras of Computing



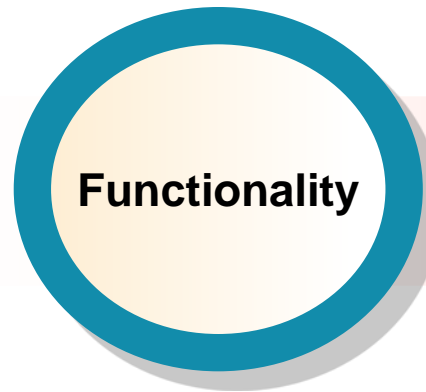
The Eras of Computing



Industry Changes in Requirements

Evolution of the industry-driving metric

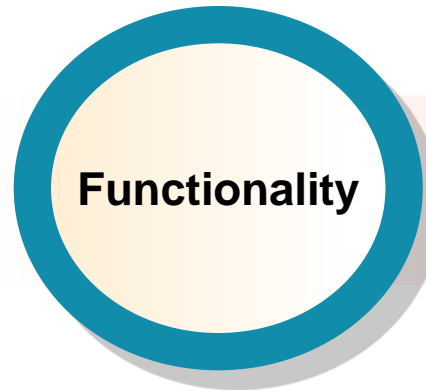
Industry Changes in Requirements



Evolution of the industry-driving metric

**Up to 1980s
Supercomputers &
mainframes**

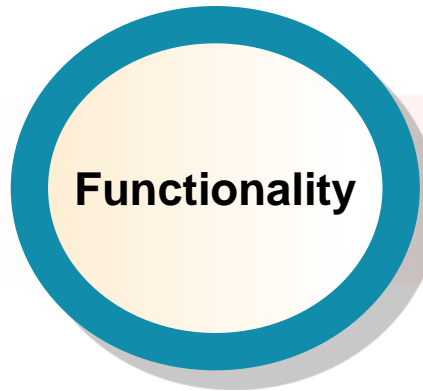
Industry Changes in Requirements



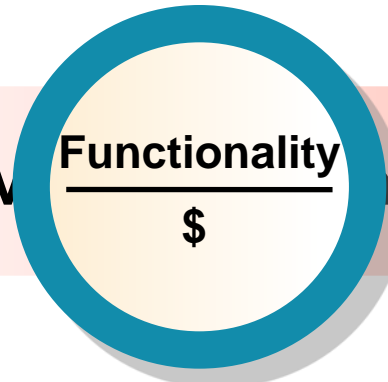
Up to 1980s
Supercomputers &
mainframes

1990s
The personal
computer

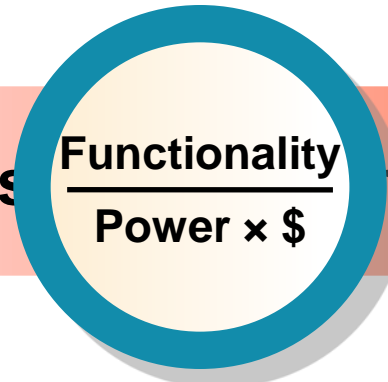
Industry Changes in Requirements



Ev



the indus



metric

Up to 1980s

Supercomputers & mainframes

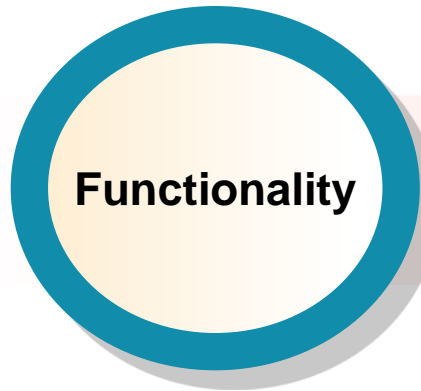
1990s

The personal computer

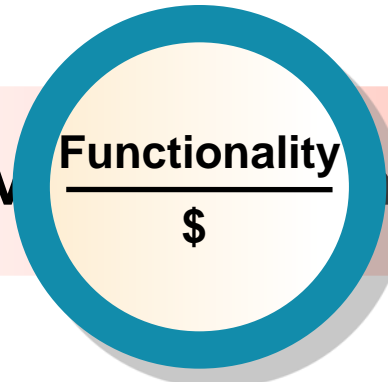
2000s

Notebooks

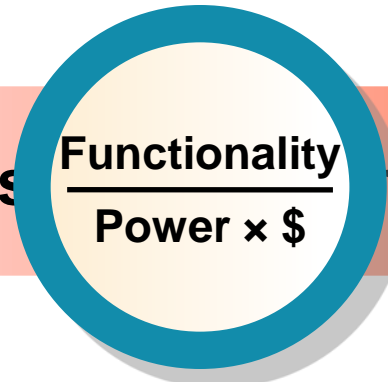
Industry Changes in Requirements



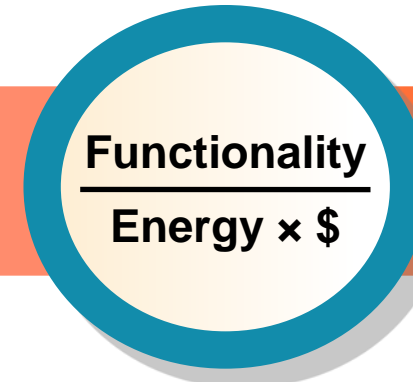
Ev



the indus



metric



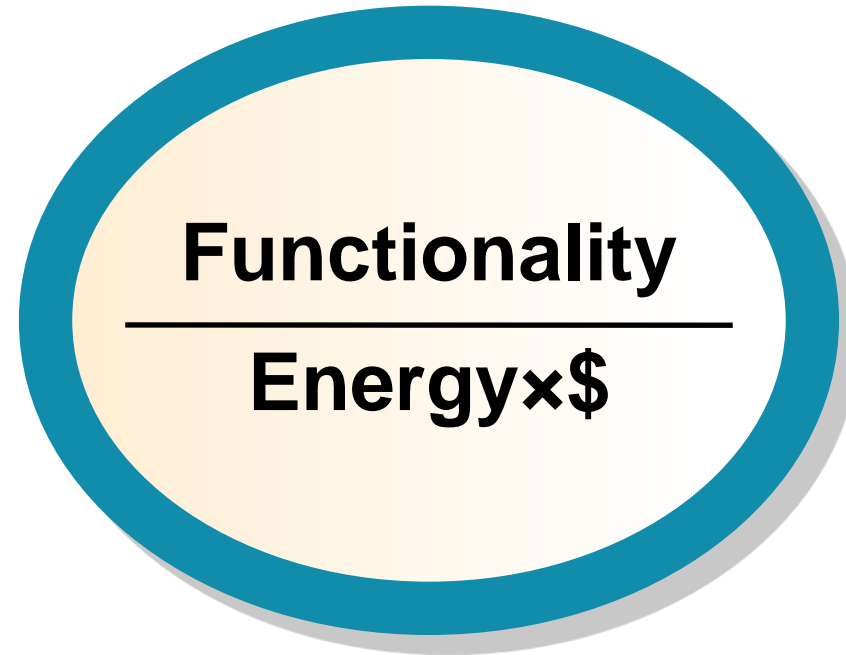
Up to 1980s
Supercomputers &
mainframes

1990s
The personal
computer

2000s
Notebooks

2010s
Mobiles &
mobility

ARM's Strategic Direction


$$\frac{\text{Functionality}}{\text{Energy} \times \$}$$

ARM's Strategic Direction

The bad news: this is a very hard metric to optimize for

$$\frac{\text{Functionality}}{\text{Energy} \times \$}$$

The good news: if you crack it, you “own” the simpler metrics as well...

Energy-efficiency Has Always Mattered

- Segment differentiation matters less than you think
- Mobile worries about warm hands/ears
 - And “*You can never be too thin*”
- Sensors want 10 years from a button cell battery
- Everybody hates fans
- Desktops struggle to:
 - Dissipate more than ~150W out of a single chip package
 - Supply more than ~300W from a PSU onto a PCI card
- Servers struggle:
 - To get more than ~10kW into a rack
 - To get more than ~500kW of heat **out of** a shipping container
 - Spending more on power and cooling than on hardware
 - To buy more power from the grid



What's Coming?

Why should developers care?



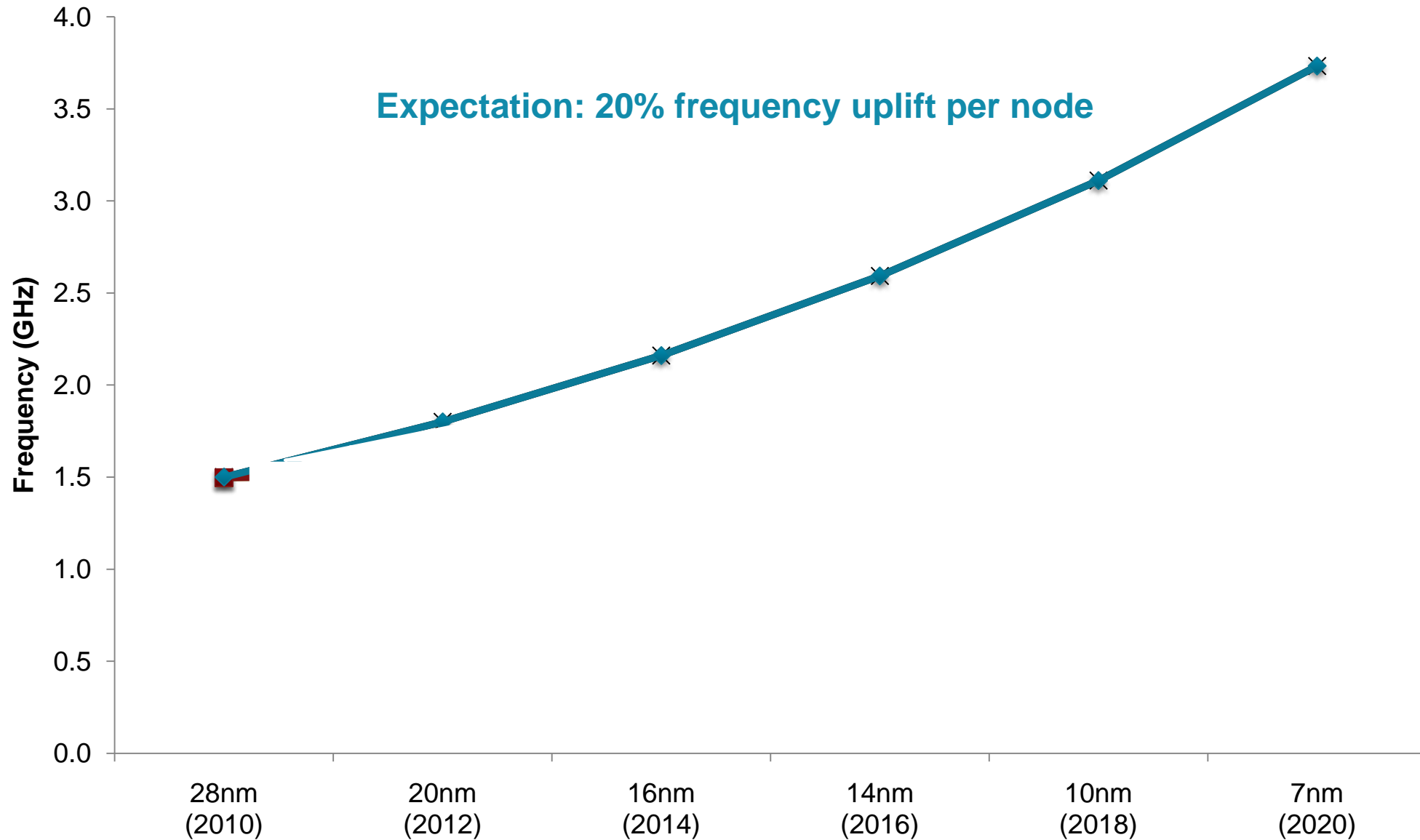
Moore's Law Is Not Dead

- Some version of Moore's law will continue to be true for this decade
- But it is getting less and less relevant

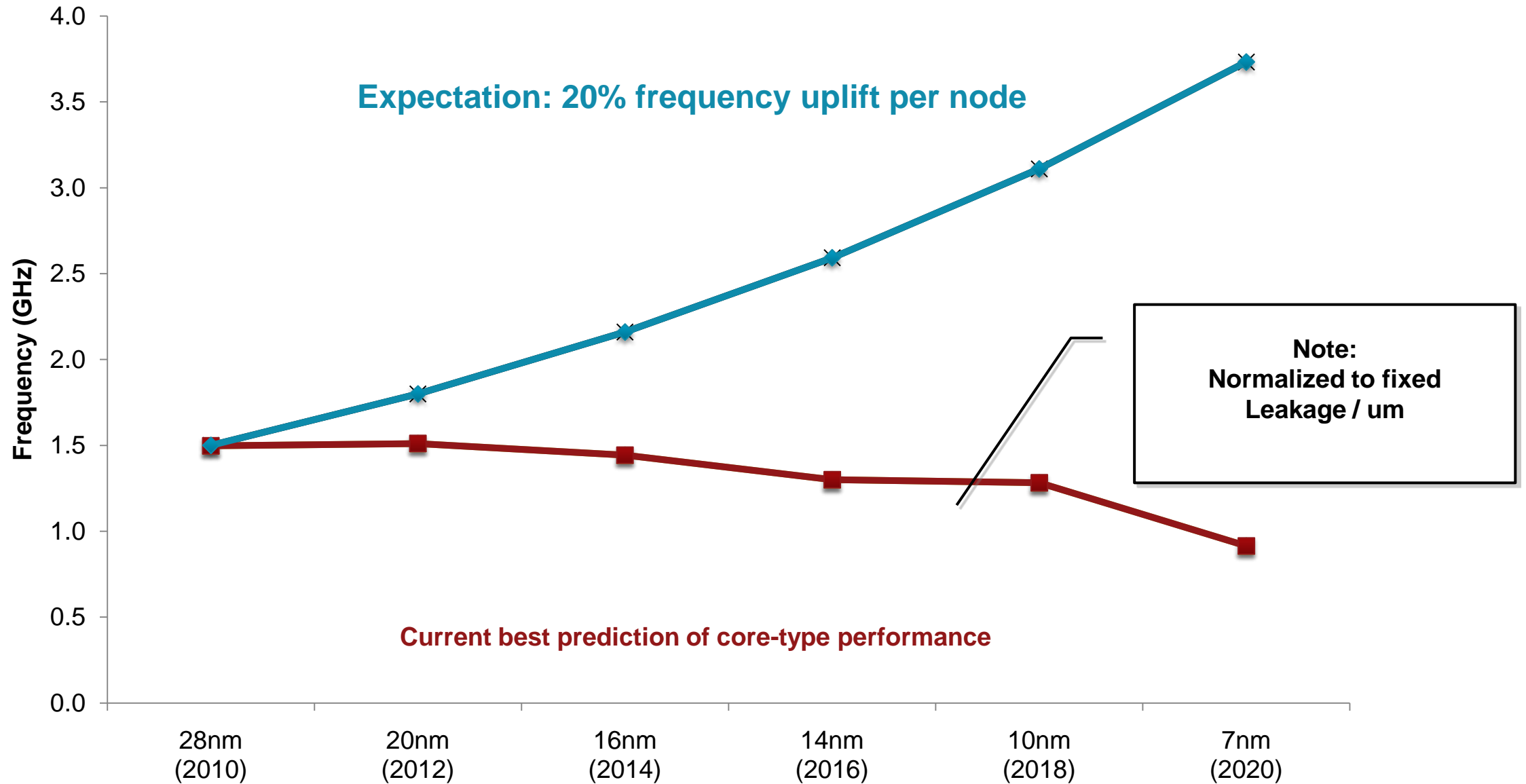


- In the past, the fabs and Moore's law gave us PPA (Power, Performance and Area) improvements for free
- But not any more...

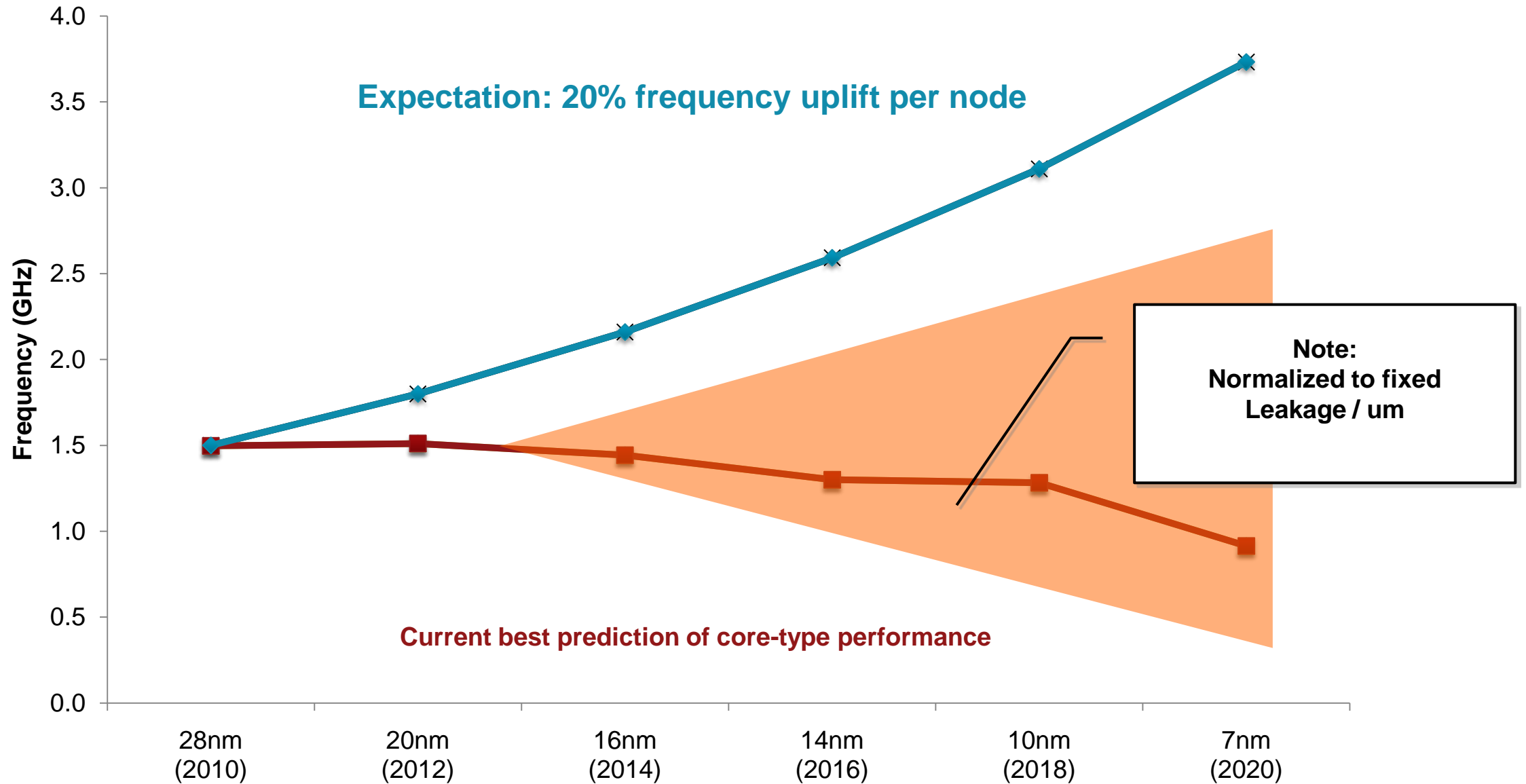
Expectations of The Future



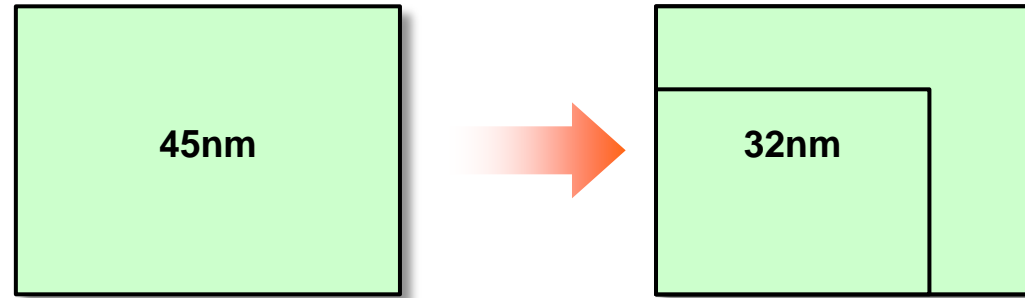
The Bitter Reality?



The Bitter Reality?

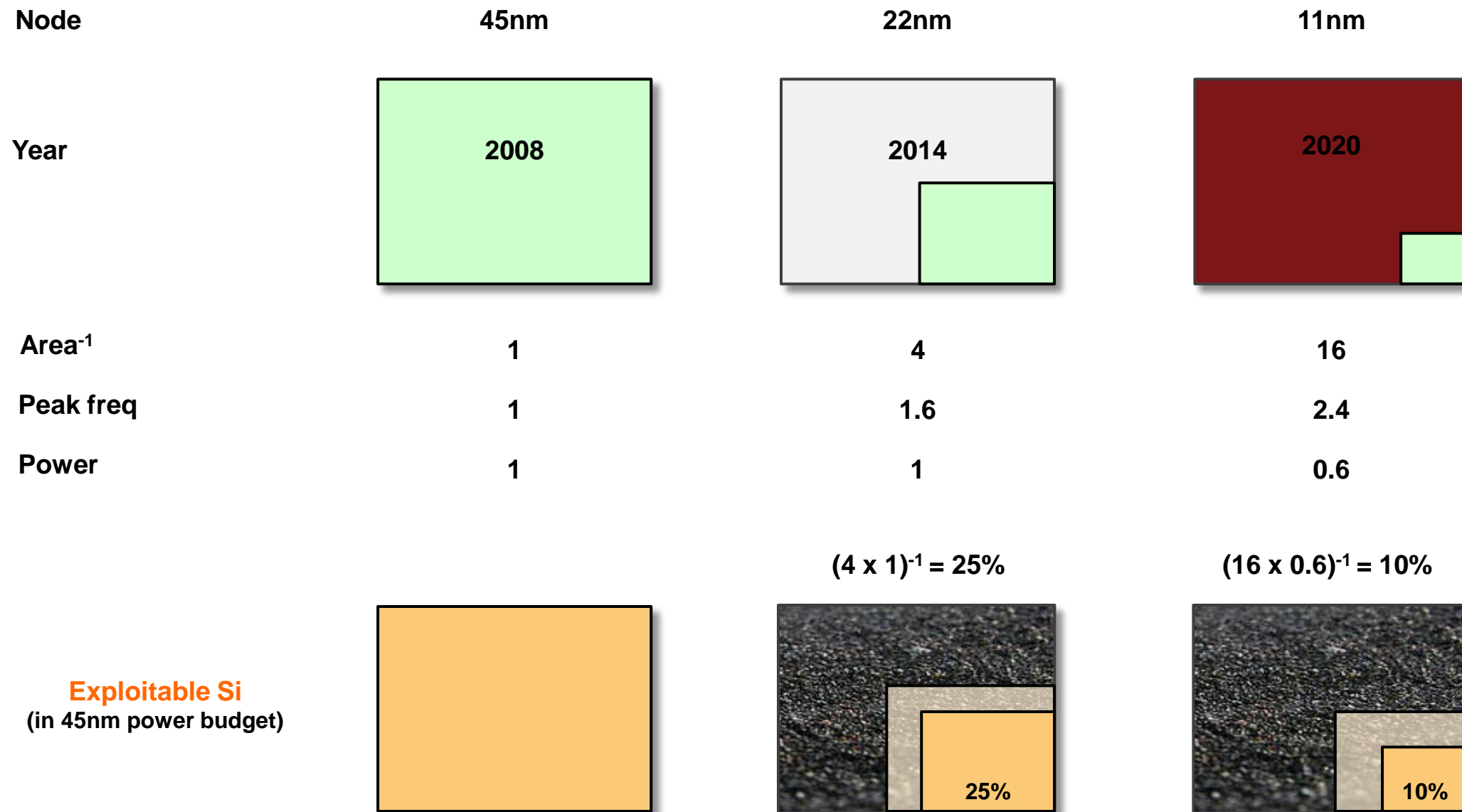


Silicon Generations: Shrink and Add



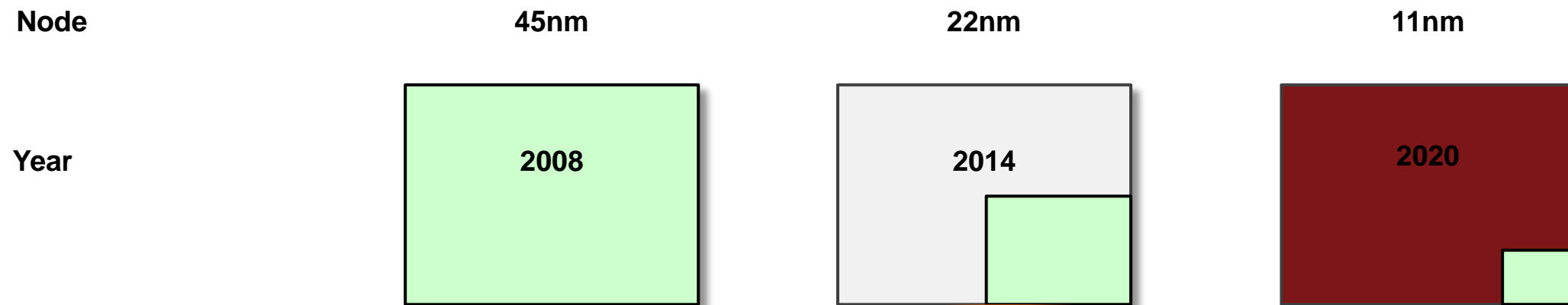
Expectation: **shrink and add**
new functionality in about same area

The Creation of Dark Silicon



Source: ITRS 2008

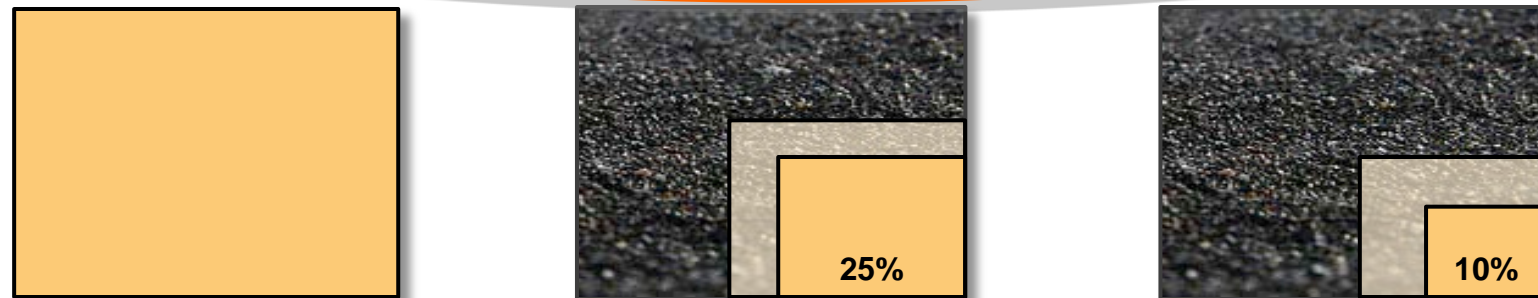
The Creation of Dark Silicon



Area⁻¹
Peak freq
Power

Lack of power scaling severely limits system options!

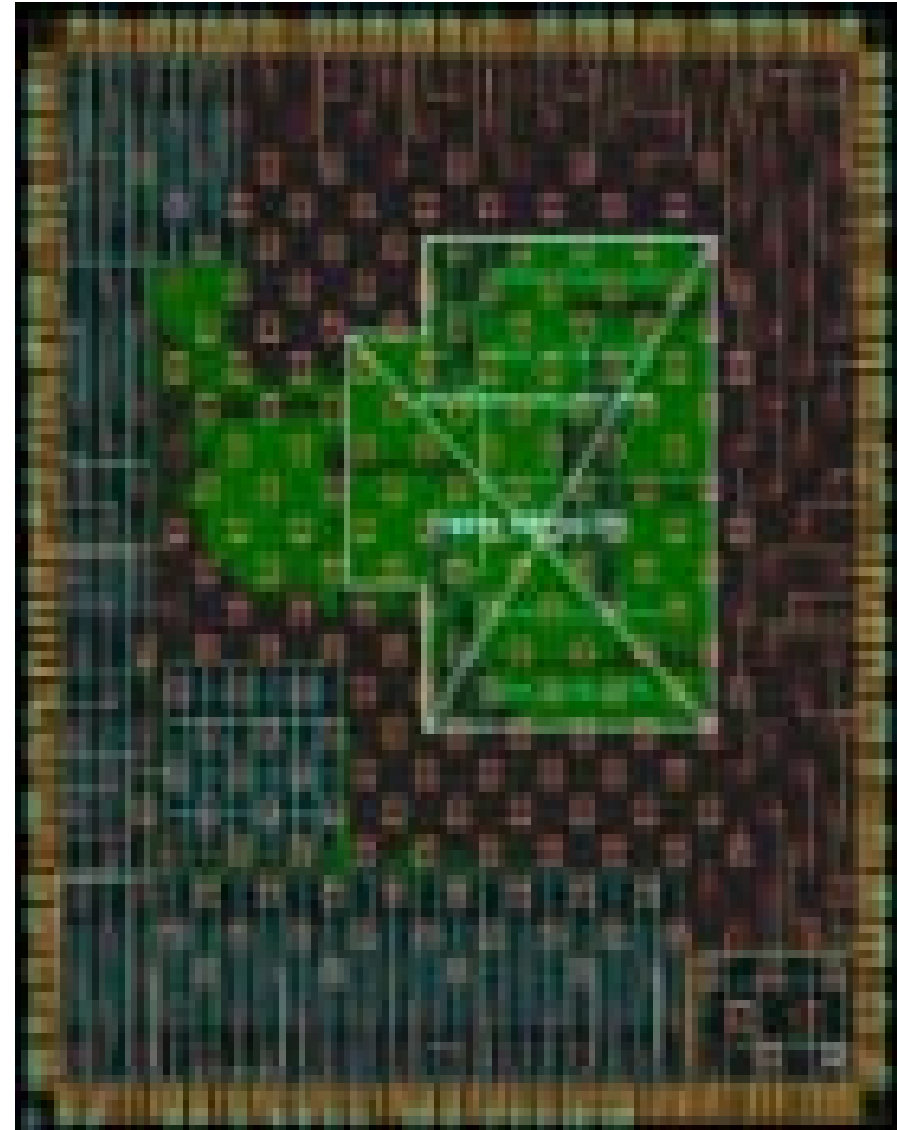
Exploitable Si
(in 45nm power budget)



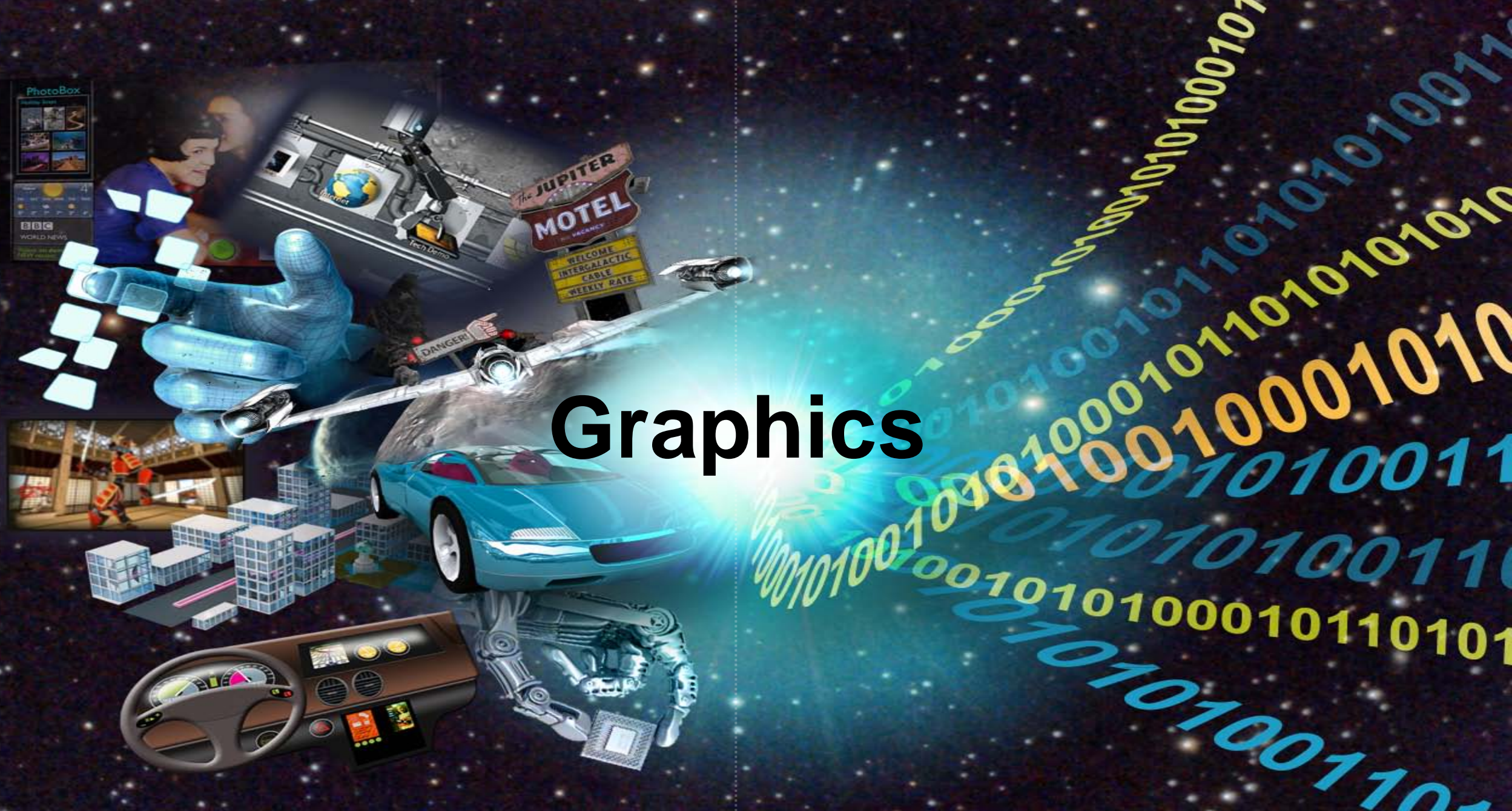
Source: ITRS 2008

So What Can We Do?

- We **can** have more transistors
- We just can't **power** them all at the same time
- We need to use these extra transistors in new ways
 - Multicores
 - Many-cores
 - Domain-specific processors
- It all points to heterogeneous processing
 - And aggressive power management
- Computing to be done in the most efficient place



Graphics



The Perfect Domain-specific Use-case?

- GPUs have traditionally been the poster-boys for domain-specific computing
 - Everyone accepts graphics on CPU makes little sense
 - From performance perspective and from energy-efficiency
- 3D computer graphics used to be very fixed-function
 - Transform, lighting, texturing, rendering...Hardware followed those fixed functions
- Modern APIs provide more flexible, programmable models
- GPUs now used for more general-purpose computing
 - So where does this all end up?

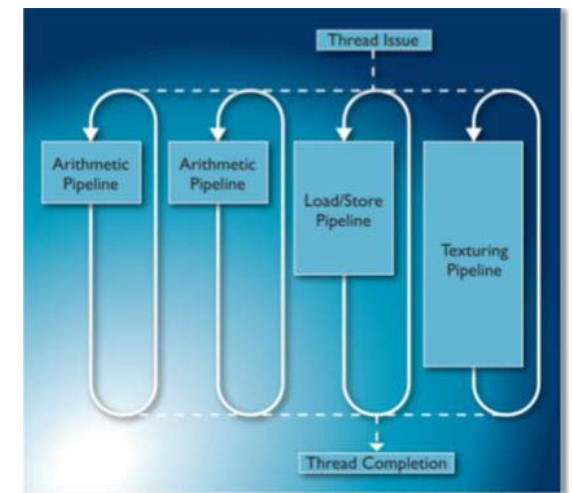
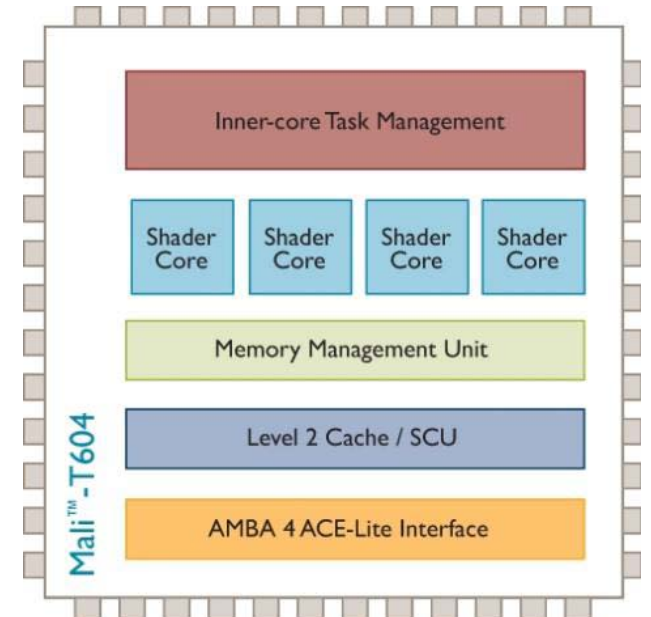


Domain-Specific vs. General Purpose

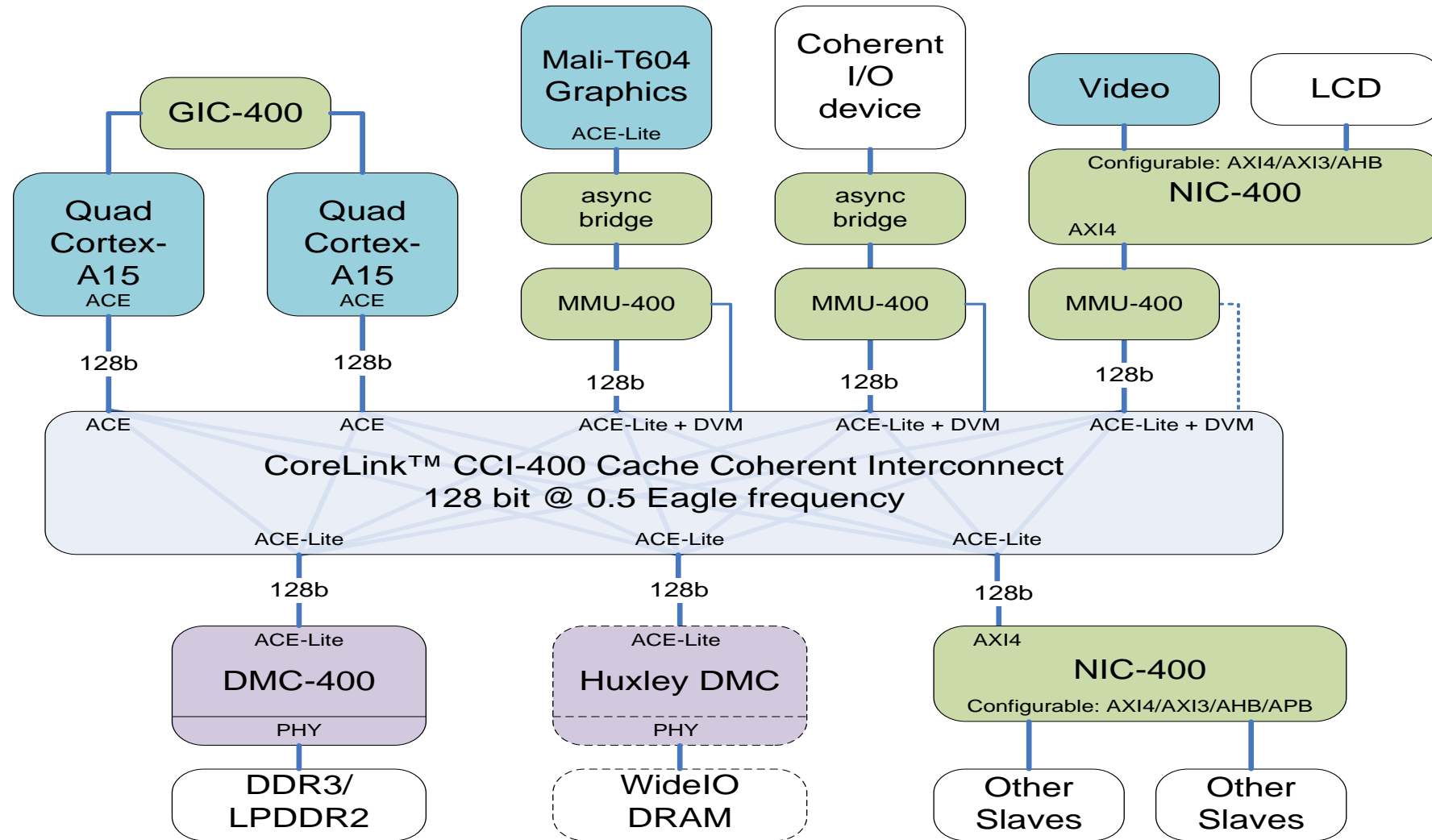
- Do GPUs merge with CPUs?
- Graphics remains one of the very few problem spaces that have **very** high levels of parallelism:
 - Do <foo> on every vertex in the frame (10s - 100s of kvertices/frame)
 - Do <bar> on every pixel in the frame (millions of pixels/frame)
- Data-level parallelism through thread-level parallelism
- Throughput-oriented architectures (and implementations) will continue to have great advantages:
 - Performance
 - **Energy**
- CPUs remain “easier” to program

Mali-T604 for Visual Computing

- Innovative 'Midgard' GPU architecture
 - Tri-pipe - for performance and flexibility
- Scalable and high-end performance
 - Superb graphics quality and performance
 - Up to 2 Gpix/s
 - Powerful general-purpose computing
 - Up to 68 GFLOPS
 - Multicore scalability up to 4 cores
- Resource efficiency
 - Frugal use of memory and bus-bandwidth
 - Dynamic power management
- Broad API and OS support
 - OpenCL™ Full Profile/RenderScript
 - OpenGL® ES and Open VG™
 - Microsoft DirectX™ to v11
- Common driver for all Midgard GPUs



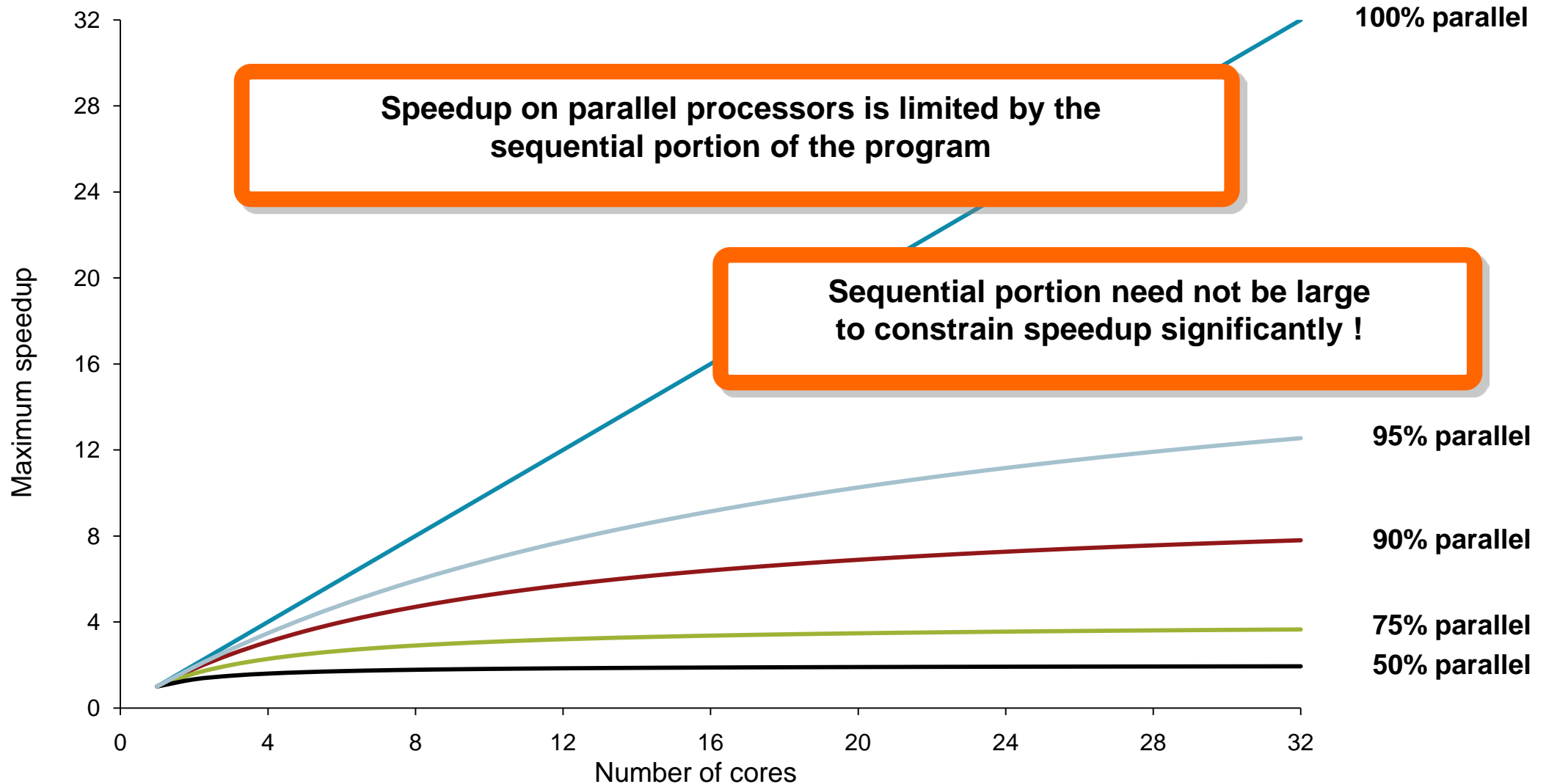
It's All About The System



GPU Computing Observations

- Beware of vested interests
 - Companies that designs CPUs, GPUs, fabric can afford a more balanced view
- Raw GFLOPS numbers tell you (nearly) nothing
- Most GPUs will only run well (efficiently, fast) with very high numbers of simultaneous threads (high parallelism)
- Most GPUs are designed for high throughput
- Most GPUs will run badly with few threads or requirements for low latency
- Remember the system (memory, energy...)
 - Understand performance vs. bandwidth vs. latency
- There is still a shortage of highly-parallelisable algorithms
 - Remember Amdahl's law

Amdahl's Law is Alive and Well

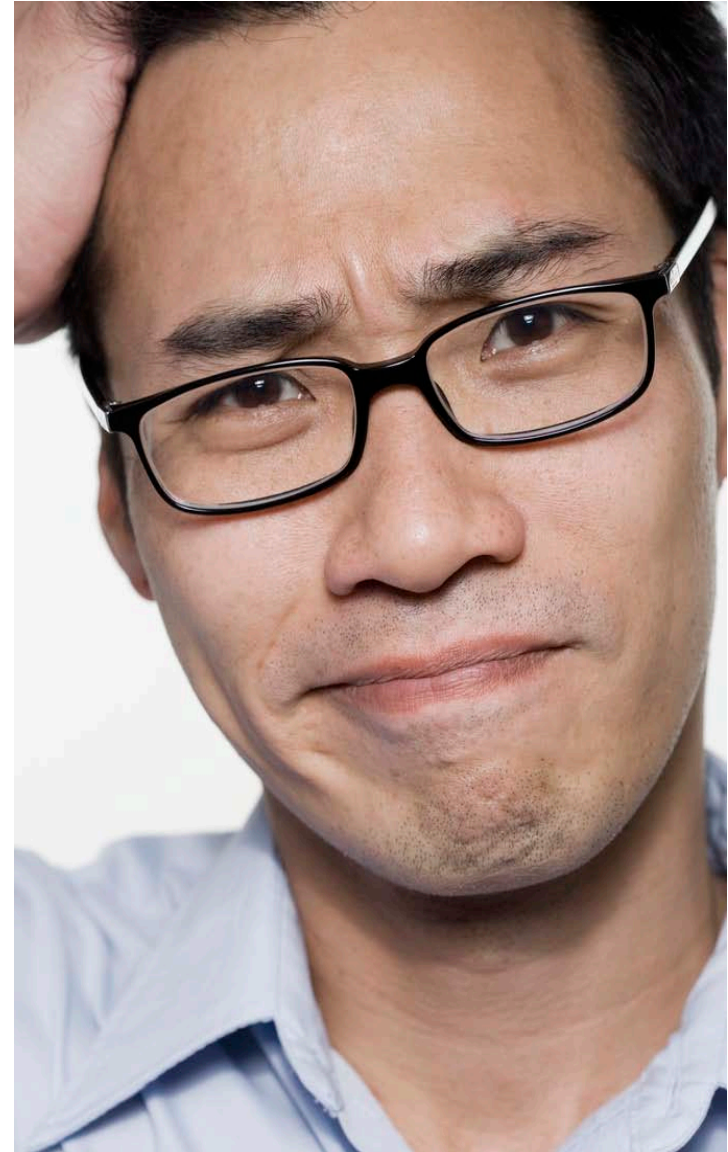


Standards



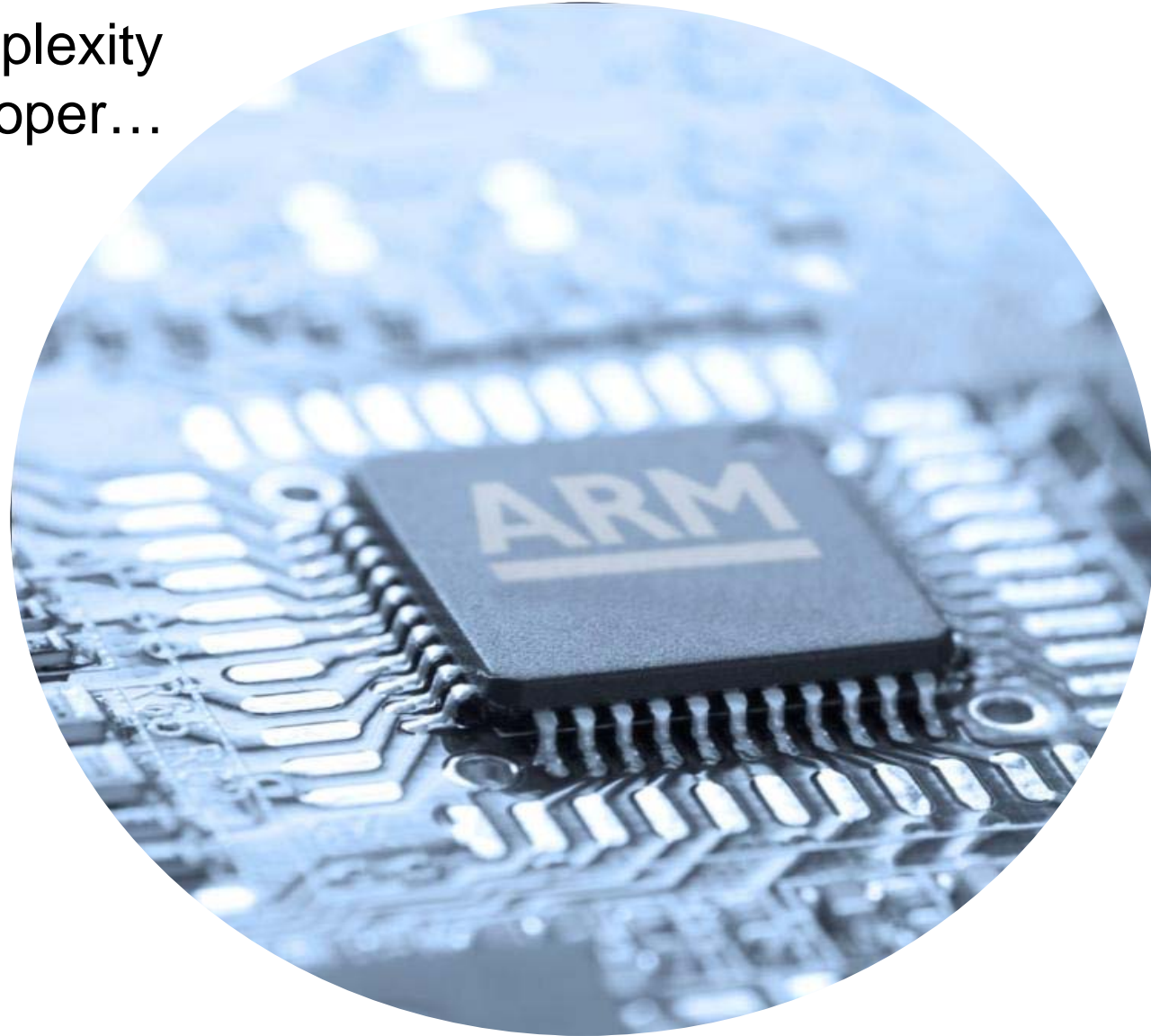
Complexity

- Out-of-order superscalar CPUs introduces some complexity
- Multicore CPUs/GPUs – more complexity
 - Memory consistency model
 - Threading models
- Heterogeneous computing - more complexity
- Some GPU compute engines are:
 - Complex
 - Badly described
 - Difficult to reason about
- Most graphics developers want to create visually stunning content, not argue about the finer points of computer science



Complexity, Abstraction, Standards

- If we do not abstract away from this complexity and present a simpler world to the developer...
 - Either they won't use these new systems
 - Or, they'll use it and get it wrong
- Either way, it will be our fault, and they will hate us for it 😞
- If we all provide different abstractions...
 - They will hate us for it
- We need **standard(s)**
 - And a very small number of them



Standards

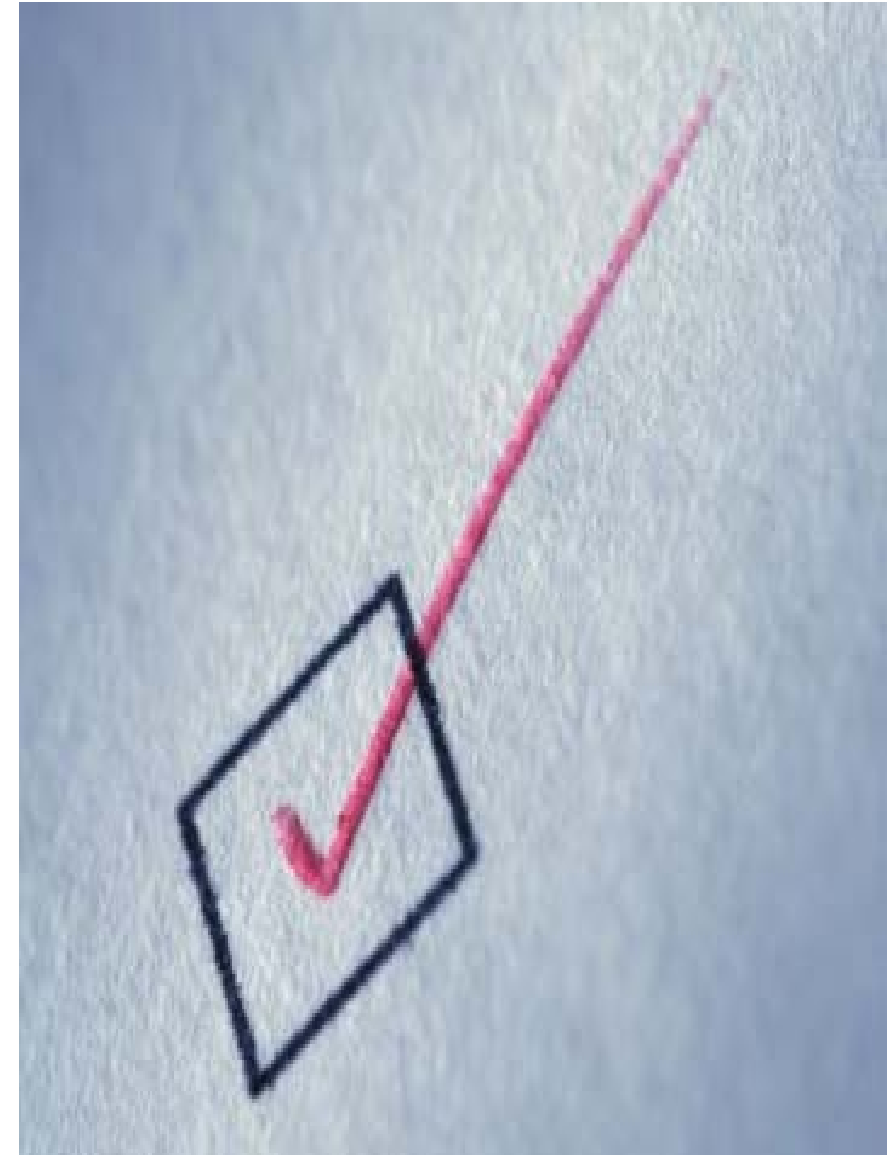
"The great thing about standards is that you have SO many to choose from."

- Andrew Tannenbaum

- And if you don't like any of the current standards, if you wait for a while, there will be new ones along...

Before Creating New Standards...

- Check that you are solving the right problem
 - The problem changes
 - Check that there isn't a "Good enough" solution already
- There are many solutions to the same problem
 - Not all have the same costs
 - Some costs are well hidden
 - Some costs are not borne by **"you"**
 - Some costs only become clear over time
- Remember the ecosystem



Developers and Developer Communities

- ARM and AMD know a bit about developer communities
- Content remains king!
- Development is really expensive and very hard
- Re-use of content is an economic imperative
 - For developers...
- But it's not enough that developers don't hate us!
 - They have to make money as well



Business Model Benefits

- Business models like ARM's means everybody makes money
 - Which encourages that developer community
 - What goes around, comes around



Different-sized Pies



Remember...

Being Right is Easy



Making Money is the Tricky Bit

Final Thoughts

- To provide increased performance we must be clever
 - Not just rely on Moore's Law
- GPU computing is here already
- Heterogeneous computing is next
- Computing in the most energy-efficient way is the real problem
 - Solve that, and everything else will be easy 😊
- When we introduce new complex stuff...
 - It must make/save money
 - It must be easily (and widely) usable
 - Or it won't get used!
 - And then we wouldn't make any money



Thank you

Questions?

