SANDIA REPORT

SAND2014-18651
Unlimited Release
Printed Sept. 30, 2014

Yearly Update: Exascale Projections for 2014

Peter M. Kogge, David R. Resnick

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2014-18651
Unlimited Release
Printed Sept. 30, 2014
Reprinted Sept. 30, 2014

Yearly Update: Exascale Projections for 2014

Peter M. Kogge
McCourtney Prof. of CSE
Computer Science and Engr.
384 Fitzpatrick Hall
Notre Dame, IN 46556
kogge@cse.nd.edu

David R. Resnick
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1319
drresni@sandia.gov

Abstract

The HPC architectures of today are significantly different for a decade ago, with high odds that further changes will occur on the road to Exascale. This report discusses the "perfect storm" in technology that produced this change, the classes of architectures we are dealing with, and probable trends in how they will evolve. These properties and trends are then evaluated in terms of what it likely means to future Exascale systems and applications.

This page intentionally left blank.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This report was prepared as part of the XGC LDRD project at Sandia National Labs, as part of a yearly update to projections first prepared for the 2008 DARPA Exascale report[20], then updated for SC 2011[19], ISC 2012[18], and then for XGC in 2013. The goal is to predict the potential characteristics of high-end systems across a spectrum of architectures into the future, and with enough lower level characteristics to allow non-trivial extrapolations against future benchmarks.

The primary focus in this report is on the interaction of technology and architecture; other issues such as programming models and resiliency receive far less attention.

For this report the term **exascale** will take on a definition similar to that used in the Exascale study,[20] namely that the aggregate amount of resources that were needed to achieve a "petascale" system (whatever that means) could, with upgrades in technology, provide 1000X the computing capability. For the purposes of this report, the metrics to be used are those associated with several benchmarking efforts for which significant historical data is available, including TOP500,[1] GREEN500,[2], and GRAPH500.[3] Extrapolations of trends in the underlying technology as made by the ITRS[4] are used to support these projections.

Also a Stanford Univ. web-based source of data on actual microprocessors[5] was used as a comparative source throughout this report as appropriate.

1.1 Seeing the "Perfect Storm" in Metric Changes

The reason for the need for an analysis as done here was the occurrence around 2004 of a "perfect storm" of design constraints that has changed forever the framework for designing HPC systems. While the emergence of power dissipation as a first class design constraint was the major driver, it was not the only cause of the change. Fig. 1.1 outlines this confluence and its general effects. The associated explosion of architectural alternatives has made the job of selecting the "most efficient" ones to pursue as we move towards Exascale systems much more difficult.

[1] www.top500.org
[2] www.green500.org
[3] www.graph500.org
[4] www.itrs.net
[5] http://cpudb.stanford.edu/

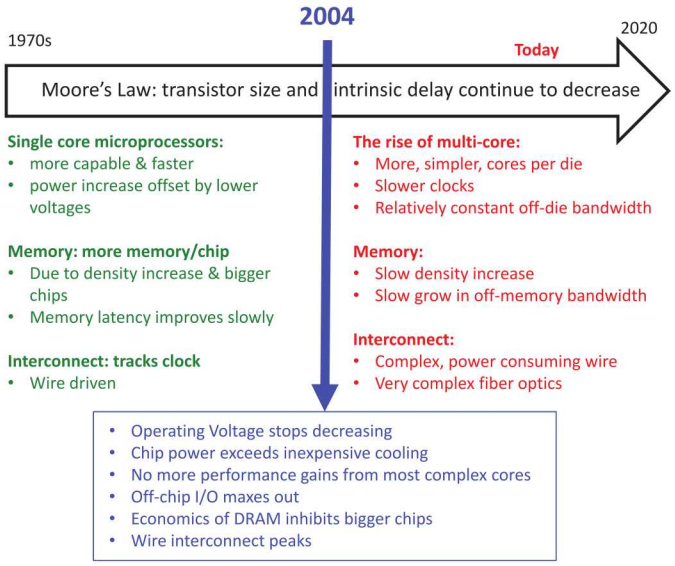


Figure 1.1. Time Line for Changes in HPC Architectures.

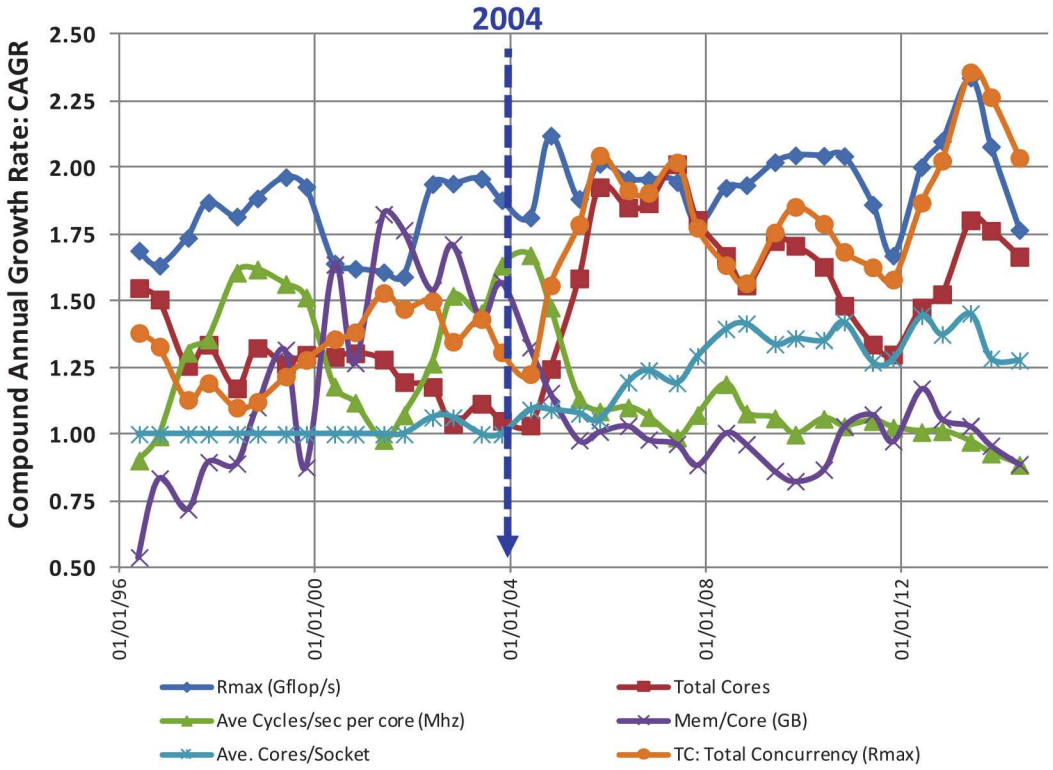


Figure 1.2. Historical CAGR Before and After the Perfect Storm.

Fig. 1.2 then demonstrates numerically some key "before" and "after" characteristics. The numbers graphed here are **Compound Annual Growth Ratios** (**CAGR**) over several years for several typical metrics. In this case a CAGR for a metric is the ratio of the value of that metric in year i+1, divided by the metric in year i. A constant CAGR over time of C years thus implies that the value of that metric in year i+k is C^k times the value in year i. A CAGR of 1 implies the metric does not change; a CAGR > 1 implies the metric increases year-over-year; a CAGR < 1 implies the metric decreases over time.

The CAGRs in Fig. 1.2 were computed by taking the performance metric R_{max}[6] values for the top 10 systems in each year from the TOP500, averaging them at each year (to get an overall technology trend), and then computing the CAGR over a 3 year period, and adjusting back to 1 year factors.[7]

The transition in the year 2004 is clearly visible in these metrics. During this entire period, the R_{max} performance metric averaged about 1.8, meaning the system performance went up by a factor of 1.8 each year. The other metrics were nowhere near so constant.

- Before 2004, the clock rate increase was comfortably above 1, with multi-year runs above 1.5. After 2004, the clock rate essentially went flat.
- Before 2004, the cores per socket was flat at 1. After 2004, the cores per socket climbed at about a 1.3X per year rate.
- Before 2004, the cores per system was declining but above 1, meaning the the number of cores in a system was growing but slowly. After 2004, the CAGR for cores per system climbed steeply, triggering a continuing explosive growth in total cores.
- Between 2000 and 2004 the memory per core was comfortable above 1.5, meaning that total system memory more than doubled every 2 years. After 2004 this growth rate went to 1 or below, meaning that available memory for each core was actually decreasing.
- Before 2004, the total concurrency as measured by the number of parallel operations that could be performed in each and every clock cycle had a growth rate of 1.25-1.5 per year. After 2004 this metric exploded.

The rest of this report dives into the details behind these numbers.

1.2 Architectures

This report is about architectures. After 2004, three classes of architectures emerged:

[6] R_{max} is the sustained number of floating point operations per second while performing some dense matrix linear algebra.

[7] A 3 year interval was chosen to provide a bit of smoothing and to match the traditional expression of "Moore's Law" as performance quadrupling every 3 years.

- **Heavyweight architectures** which are the natural progression of what are now the ubiquitous multi-core microprocessors, and are designed to work with a combination of support chips to provide the most possible performance, typically at high clock rates.
- **Lightweight architectures** which are essentially single-chip systems, other than memory, and are run at lower clock rates.
- **Hybrid accelerator architectures** that combine a multi-core chip designed with very many floating point units and a small amount of very fast external memory, to a conventional, usually heavyweight, processor.

The Exascale report[20] identified the first two; the third emerged with the announcement of Roadrunner and the later convergence of **graphics processing units** (**GPU**) and conventional processing.

In addition, a fourth class of architectures. termed here **BigLittle**, appears to be emerging from a mixture of all the above three, where there may be a mix of core designs as in the hybrids but where both the high performance heavyweight and low power lightweight cores all share the same ISA. In addition, we expect to see in a fully developed BigLittle architecture the ability of migrate threads between core types as processing warrants.

Finally, a fifth class of architecture, termed **2.5D** here, is emerging where memory and routing are integrated into a three dimensional stack of chips where transport costs between memory and logic are dramatically reduced, and where simplified protocols speed transfer of data to other stacks, both other memory stacks and/or nearby processor chips. This is in preparation for a second, and perhaps more significant, jump to true **3D** architectures, where the base chip(s) contains processing logic of sufficient capabilities to consider eliminating the need for separate conventional processing chips.

1.3 Organization

The rest of this report is organized as follows:

- Chapter 2 describes briefly the two classes of benchmarks against which the evaluations will be made, and overviews the terminology used throughout.
- Chapter 3 discusses the underlying technology driving HPC architectures, including the 2004 confluence.
- Chapter 4 discusses the architectures present both before and after 2004, and new ones that are liable to emerge in the next few years.
- Chapter 5 discusses the comparative properties of current and past systems and architectures as benchmarked against LINPACK, as in the TOP500, GREEN500 and BFS, as in the GRAPH500.

- Chapter 6 discusses the approach taken to project these architectures into the future.
- Chapter 7 summarizes the results.

It is planned that in the near future a revised model based on the 2012 environment will be prepared and added to this report.

1.4 Versions

Table 1.1 lists the versions of this document.

1.5 Known Limitations in this Document

There are several areas of updates that do not appear in this version, and hopefully will be included in future updates:

- Not all of the 2013 update to the ITRS roadmap was released, as has been normal practice, at the beginning of 2014, and had not yet been made public by the time of this document update. This includes in particular DRAM densities, power, clock, and Vdd projections. Thus Figs. 3.7 through 3.18 are the same as last year.
- No additional data was found on DDRx projections over last year.
- While the HMC Gen-2 productized memory stack has been announced, not all of its characteristics have been made available, nor much additional detail on the expected Gen-3 parts.
- Given the rapid conversion to FINFETs for the 20 nm and below technology generations, the discussion on CMOS transistors should be updated to consider the effects of FINFETS on design characteristics.
- The energy model of Chap. 6 has taken longer than expected to get right, and will be updated in the next release.

Version	Date	Changes
1.0	8/12/13	First version.
2.0	9/30/14	Update to reflect technology evolution in 2014, especially in signalling off-chip. Significantly new sets of benchmarks has been added. Discussion of 2.5D architectures begun.

Table 1.1. Version History.

- Not enough publicly available data is yet available on the first HMC stack to update the models of Chap. 6.3.1 in a meaningful way.
- The POWER7 data will be updated when enough information on products using them becomes available.

Chapter 2

Benchmarks

Two major benchmarks have become relevant to the exascale community: LINPACK to track floating point performance, and Breadth-First-Search in large graphs to track communications performance. A third suite of benchmarks, termed the HPCS suite, have become popular to measure specific aspects of a system's performance. A fourth suite of benchmarks, HPC, has been defined to address limitations of LINPACK by a more complete algorithm.

2.1 Compute Intensive Computing, LINPACK, the TOP500, and the GREEN500

The most common benchmark to date for supercomputers has been the solution of dense linear equations of the form $b = A * x$ (see [8]). Assuming A is an n by n matrix and B and x are n-element vectors, the nominally most important computational factor is the number of floating point operations performed in the solution, or **flops**. The nominal number of such flops is $2n^3/3 + 2n^2$ for an LU matrix decomposition algorithm. The LINPACK library implements this functionality, as well as other operations.

Benchmarking a system involves executing LINPACK on the system for various sizes of n, and measuring the execution time. Dividing this into the number of flops for that n gives floating point operations achieved per second, or **flops/s**. By trying different values of n, a benchmarker can then identify the value that yields the largest flops/s. The value of n which maximizes this is N_{max}, and the flops/s number is called the R_{max}. A related metric is $N_{1/2}$ which is the size of the problems that achieves $R_{max}/2$ flops/s.

When used as a benchmarking program this version of LINPACK is also called **HPL** ("high performance linpack").

The other key metric for a system is R_{peak}, which is the peak number of floating point operations that can be performed by the system, regardless of algorithm or problem size. In most cases this is the product of the number of floating point units and the clock rate of those units.

The TOP500 web site[1] has been recording such measurements for almost 20 years, and ranked

[1] www.top500.org

systems on the basis of their reported R_{max}. More recently, as power and energy efficiency has become a major concern, the **GREEN500** web site[2] has tracked the ratio of a system's R_{max} to the power it consumes, and ranks the system on the basis of the resulting **MFlops/s per Watt** (millions of flops executed per second per watt of power dissipated). An equivalent of this metric that often provides more insight is the reciprocal of this, or the energy per flop performed. This is typically expressed in units of **picoJoules/flop** (pJ/flop). where one **pJ** is 10^{-12} Joules, and where 1 pJ/flop is thus equivalent to 10^{12} flops (or 1 **teraflop**) being executed in 1 second while the system dissipates only 1 Watt.

The nominal goal for the DARPA UHPC Exascale initiative has been at least an R_{peak} of 10^{18} flops/s, with a power budget of 20MW. This corresponds to an energy goal of 20pJ/flop.

The major objection to the validity of the TOP500 benchmark is that it is too regular, and too floating point intensive. Detailed analyzes such as [27] indicate that in real, large, scientific codes the percentage of floating point operations is much less than in LINPACK, with address computations and memory referencing becoming more important to time and energy than floating-point operations. This trend is expected to grow even further as dynamic grids and multi-physics become more common.

2.2 Data Intensive Computing and the GRAPH500

The Graph500 benchmarks[3] are meant to stress parts of a system not key to LINPACK, such as handling extremely large data structures that must encompass many nodes, and high amounts of unpredictable communication between the nodes.

Several benchmarks are planned under this Graph500 umbrella, with only one of them ("Search"), currently defined and tracked through several generations of systems. Two other benchmarks ("Shortest Path" and "Maximal Independent Set") are planned in the near future, and will be added to future updates of this document as they become available.

In addition, a **GREEN GRAPH500** listing has been started which lists the most energy efficient of the submissions to the GRAPH500.

2.2.1 The Graph500 Search Benchmark

The purpose of the kernels defined in this benchmark is to build a very large graph, and then be able to rapidly start at any random vertex and identify all other vertices that are connected to it. This exploration for some starting vertex is often called **Breadth First Search** (**BFS**)

Fig. 2.1 shows a sample graph. Starting at vertex 0, a valid BFS output would be 0, 1, 2, 9, 3,

[2] www.green500.org
[3] www.graph500.org.

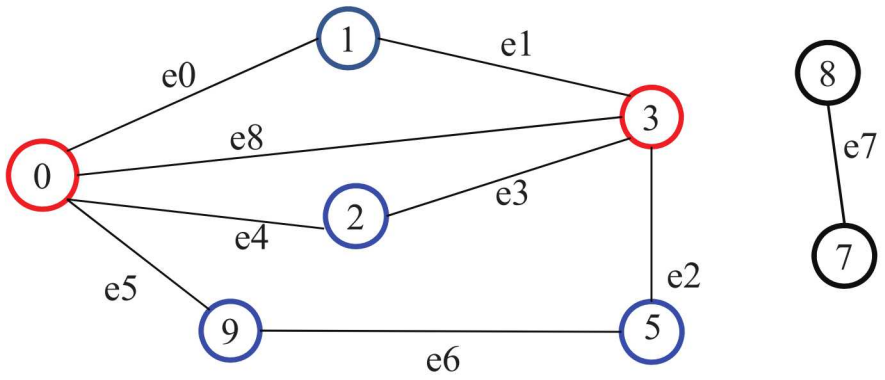


Figure 2.1. A Sample Graph.

5. Starting at 2 a valid output would be 2, 3, 0, 9, 5, 1.

There are three major steps in the benchmarking process:

1. Graph construction: to create a data structure to be used for the BFS. The two configuration numbers that go into this are:

   - Scale: base 2 log of number of vertices (N) in the graph.
   - Edgefactor: ratio of total number of edges (M) to total number of vertices (N) in the graph. The GRAPH500 uses an edgefactor of 16.

     Note that all edges are assumed bi-directional so that the average degree of a vertex is twice the edge factor, or 32.

2. Breadth-First Search: starting at a random vertex, follow all edges from that vertex and all vertices reached from that edge until as many vertices as possible have been reached. Record these vertices in level-order: the root vertex, all vertices that are 1 edge away from the root, all vertices that are 2 away from the root, ...

3. Validation: check that the answer is correct.

The key performance parameter is the time for a BFS search. If M is the total number of edges within the component traversed by a BFS search (the second step), and T is the time for doing that search, then **Traversed Edges per Second** (TEPS) for a particular search is M/T. The time for the first and third steps is not part of the benchmark.

As with the TOP500's N_{max}, there is a problem size component to the GRAPH500, although in this case it is far more important to evaluating the success of a system than it is in TOP500. Table 2.1 lists different classes of problem size and a typical size of the required data structure in bytes,

Level	Scale	Size	Vertices (Billion)	Memory (TB)
10	26	Toy	0.1	0.02
11	29	Mini	0.5	0.14
12	32	Small	4.3	1.1
13	36	Medium	69	17.6
14	39	Large	550	141
15	42	Huge	4,398	1,126

Table 2.1. GRAPH500 Problem Size Categories.

where an average node and its associated edges take about 282 bytes. The "Level" in this table is approximately the base10 log of the estimated size in bytes. To date, very few systems have reached the "Large" category (needing 140TB), let alone the "Huge" category.

Note in Table 2.1 the scale in each problem category goes up by 3 (a growth of 8X) as the level goes up by 1 (a growth by a factor of 10), except between levels 12 and 13, where there is a growth by 16X in scale. This extra step is to account for the difference between 8^3 and 10^3.

As with the TOP500, there is some criticism of the GRAPH500 in that it is too one-sided: almost only focused on inter-node communication. Consequently, the GRAPH500 organizing committee is planning on releasing two additional benchmarks: shortest path optimizations and computing maximal independent sets. Unlike the scientific-orientation of LINPACK, these benchmarks are believed to be highly related to five business areas: cybersecurity, medical informatics, data enrichment, social networks, and symbolic networks.

2.3 The HPC Benchmark Suite

The HPC Challenge benchmark suite[4] is an outgrowth of the DARPA HPCS project to focus on performance evaluations of large-scale parallel MPI implementations. Currently there have been 362 systems with self-reported results since 2003.

The benchmark programs in the suite include:

- **HPL**: the LINPACK benchmark used in TOP500.
- **DGEMM**: double precision matrix-matrix multiply.
- **STREAM**: tests for sustainable memory bandwidth for 4 long vector operations.
- **TRIAD**: vector operations of the form a[i] = b[i] + c[i]*SCALAR.

[4]http://icl.cs.utk.edu/hpcc/index.html

- **PTRANS**: parallel matrix transpose.
- **RandomAccess**: Integer random updates (**GUPS**)/indexGUPS.
- **FFT**: double precision floating point fast fourier transforms.

2.4 Sparse but Compute Intensive Computing, HPCG

To quote Jack Dongarra on the HPL benchmark using LINPACK[7]:

> HPL rankings of computer systems are no longer so strongly correlated to real application performance, especially for the broad set of HPC applications governed by differential equations, which tend to have much stronger needs for high bandwidth and low latency, and tend to access data using irregular patterns. In fact, we have reached a point where designing a system for good HPL performance can actually lead to design choices that are wrong for the real application mix, or add unnecessary components or complexity to the system.

The reason for the new focus resides in the regularity of memory references and in the ratio of memory accesses to the amount of computation. LINPACK exhibits a preponderance of **Type 1** references, where dense matrices are the predominant data structure and there is significant reuse of the data read from memory (normally with "unit stride" where a long series of adjoining memory locations are referenced).[5] This is a good match to the deep cache hierarchies and high floating point computation capabilities of current architectures.

Many of the newer high end applications, however, must sacrifice this regularity of data and computation to more accurately represent much larger problems than can be attacked with dense approaches. Such problems use adaptive or irregular meshes and mixed domain computations, leading to matrix representations that are quite sparse. By using this sparsity, orders of magnitudes reductions in both storage and computation are possible. However, compact representations of such structures, in turn, result in multiple indirect references to access the data needed to perform even singleton floating point computations, with very little reuse of the data. The resulting memory traffic, called **Type 2** references, patterns by Dongarra, put significantly different demands on architectures than do Type 1, especially in the memory subsystems.

The proposed benchmark, called **HPCG** for **High Performance Conjugate Gradient**,[12], involves solving very large problems of the form $b = A * x$. The first step is to generate a large synthetic positive definite matrix A in a compressed sparse row format (**CSR**) (similar to that used in the GRAPH500 benchmark) where for most rows there are 27 nonzero entries(corresponding to a 3D stencil on the interior of a 3D grid), with 7 to 18 values on rows corresponding to edge grid points. Also generate values for the b vector, and an initial guess for the x vector.

[5] Such patterns exhibit both good temporal and spatial locality, and thus adapt well to caching techniques.

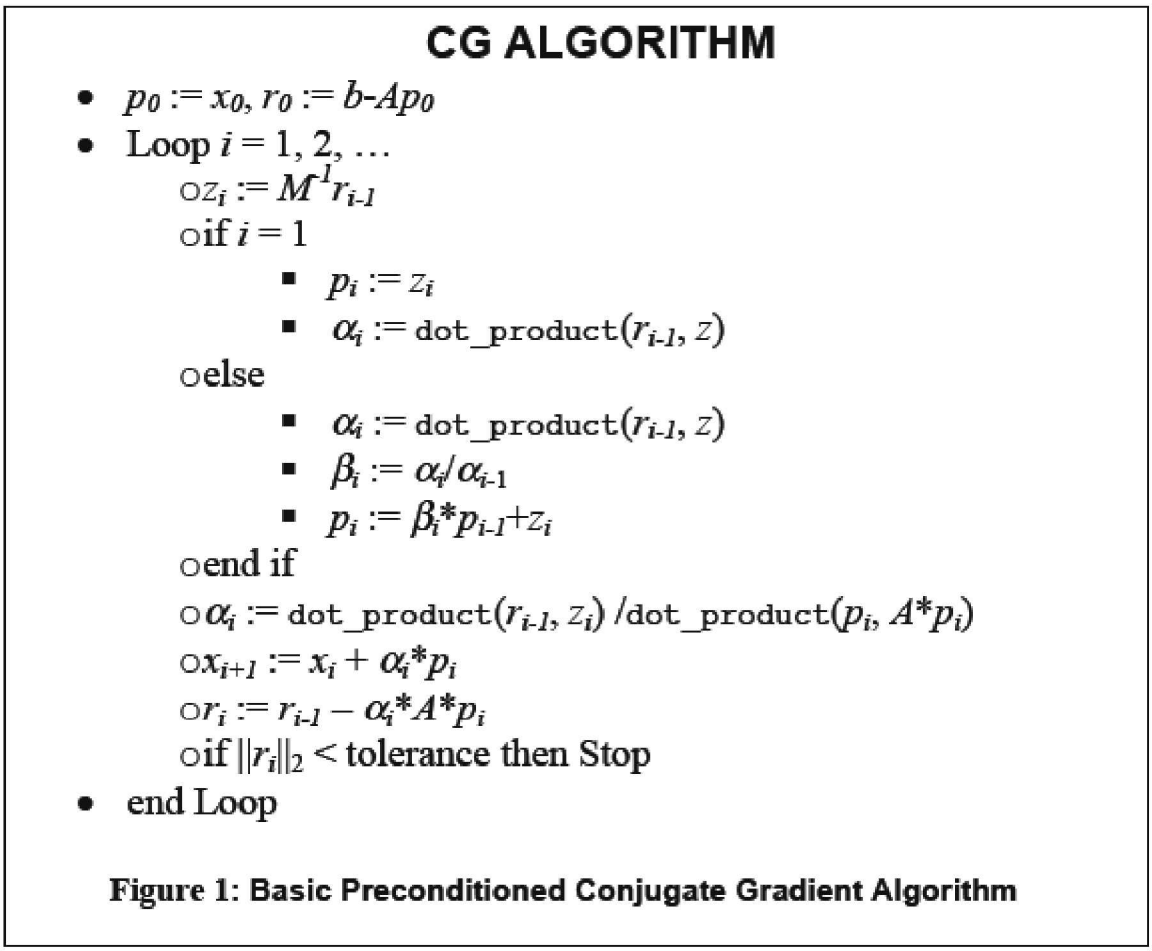


Figure 2.2. Pseudocode for HPCG, taken from [12].

The computation is a loop as pictured in Fig. 2.2 where the key operation is a large sparse matrix-dense vector product, surrounded by several inner products between dense vectors, and several dense vector-scalar operations. While the dense vector products are cache-friendly, there are only a limited number of floating point operations (2) per point. The bulk of the accesses and operations are in the sparse matrix-vector product, where there is no locality and again few operations per point.

The Sandia report [12] and the website https://software.sandia.gov/hpcg/ provides more information on the benchmark, and a single early set of measurement. It is expected that new measurements will come as frequently as the TOP500 listings with LINPACK.

This page intentionally left blank.

Chapter 3

Silicon Technology and the Perfect Storm

This chapter addresses the key properties behind the silicon technology that has driven the high performance computing environment for decades, with emphasis on what it is that caused a major disruption in architecture about 2004. Throughout this chapter, data is typically taken from the ITRS roadmaps[1], where updates to projections of the future of silicon chips have been published yearly since 1994. The data used here is an amalgam of these reports, typically starting with the earliest and overlaying each years's future projections. Thus the data for a particular year in the past was from the most recent year that included it (typically either the same year of the report or the next year's report that looked back a year). This ensured that the now "historical" data actually reflected reality on a consistent basis. The data for years in the future are likewise from the most recent year, in this version of the report from the 2013 update[2].

Much of this chapter is introductory in nature, and was included to help allow those without a semiconductor technology background to grasp what caused the changes in everything from architecture to programming.

Additional data on actual microprocessors is gleaned from personal records, the CPUDB website[3] hosted by Stanford University, and from data reported on the Intel Xeon line[4].

3.1 Moore's Law

Moore's Law[26] initially came from a plot in a 1965 paper that showed the number of transistors doubling every year, with a prediction that it would continue for a while. In [25], Moore is quoted as saying that a colleague later extended the number to a doubling every 18 months (the typically reported number), while Moore himself used 2 years starting in 1975 when he had a decade of data.

At the turn of the millennium, it was popular to interpret Moore's Law as saying that "performance" of microprocessors and "capacity" of memory chips increase exponentially, the real statement of most underlying importance is that the key linear dimensions of a transistor (its "fea-

[1] www.itrs.net

[2] as of September 2014 not all of the 2013 ITRS projections have been posted.

[3] http://cpudb.stanford.edu

[4] http://en.wikipedia.org/wiki/Xeon

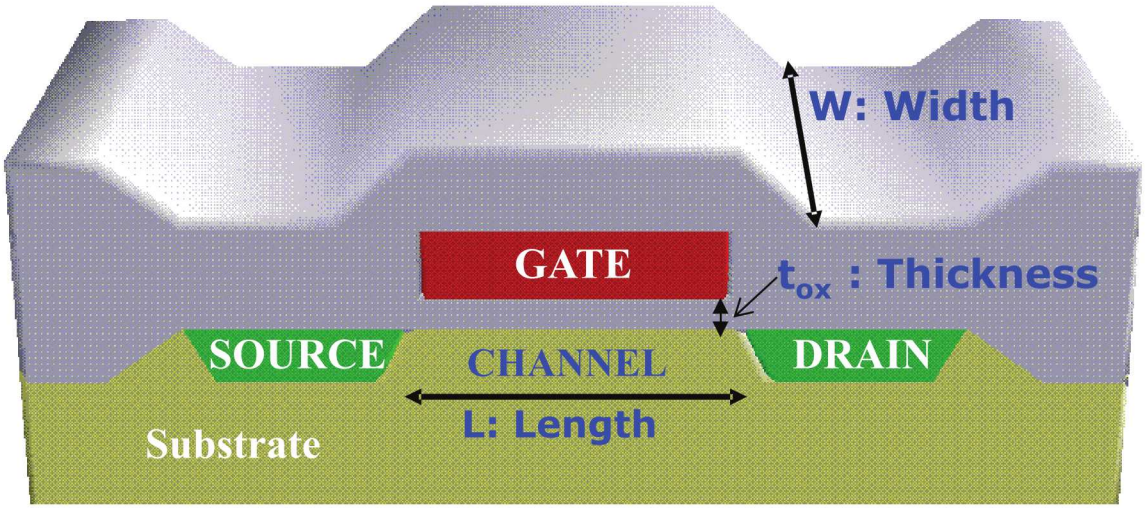


Figure 3.1. A CMOS Transistor.

ture size") shrink by a relatively constant factor every N years. This shrinkage has had two effects: the overall area of the transistor has dropped, meaning that more transistors can be placed on a die, and that the inherent delay of the transistor (due largely to the capacitance of its gate) has declined. The dimensional shrinkage has also been applied to the width of the wiring that interconnects transistors together, meaning that the area of a multi-transistor circuit, such as a processing core, has decreased likewise by approximately the square of the shrinkage in feature size (both length and width have shrunken by the same factor).

The first microprocessors used transistors with feature sizes of around 10 microns (10^{-2} mm); current transistors are around 22 to 28 nanometers (22 to 28*10^{-6} mm), with a projected "end-of-the-road" of around 5 nanometers. The end-of-the-road numbers occur when transistor dimensions are a very few atomic widths in size. All told, this represents about a 500-fold decrease in linear dimensions to date (dropping to 2,000-fold in the future) or about a quarter million-fold increase in transistors per unit area (rising to four million-fold).

It is interesting that in the original paper[26], Moore predicted that power dissipation problems "won't happen with integrated circuits. ..." In fact, shrinking dimensions on an integrated structure makes it possible to operate the structure at higher speed for the same power per unit area."

3.1.1 The CMOS Transistor and Feature Size

Fig. 3.1 diagrams the cross-section of a CMOS[5] transistor (not shown are connections to the source, drain and gate, typically from metal layers above the transistor). When a voltage of sufficient magnitude is applied between the gate and the substrate, a current can flow in the channel

[5]CMOS stands for Complementary Metal on Silicon

between the source and the drain. The key parameters of a transistor are its length, width, and thickness of the oxide between the gate and the channel. When placed in a circuit there are "layers" of metal above the transistor, with vertical "vias" descending to electrically contact each transistor's source, drain, and gate. When multiple transistors are wired in parallel and/or series using such wires, the resulting circuit can change the voltage at some output point, which in turn can affect the voltage at the gates of down-stream transistors.

In the semiconductor industry, CMOS technology is grouped into **generations** based on **feature size**, which in turn is related to the linear dimension of one of the smallest objects that can be patterned on a chip. Today, this is typically the 1/2 distance between two minimum-width wires in the layers above the transistor, which in turn is itself a small multiple of L, the channel length. Also typically a change in one linear dimension, such as L, is matched by changes in the other key dimensions, such as W and t_{ox}. While such changes used to be approximately linear in all directions (called "Dennard Scaling"), current scaling is more difficult. For example different materials are used than previously in order to change the dielectric constants of portions of a transistor, raising manufacturing complexity and cost. Another example is that metal thickness is generally not reduced by the same factor as that of line width, as metal resistance increasingly interacts with the various capacitances to slow signal propagation even though line lengths are being reduced on the average.

Assume for now that the shrinkage between one generation of transistors and the next is a ratio S. Thus if the feature size of one generation was 90nm, then the next generation would have been 90/S nm, the one after that is $90/S^2$ nm, and so on.

Clearly if all linear dimensions shrink by a factor 1/S, then the area of a circuit that has exactly the same relative layout will shrink by $1/S^2$ in the next generation. This corresponds to a growth of S^2 in the number of transistors per unit area. As an example, a reduction of a factor of 2 in feature size corresponds to a reduction of a factor of 4 in device area, and thus an approximate growth by a factor of 4 in the numbers of transistors that can be placed on the same area of silicon.

Fig. 3.2 diagrams the change in feature size for transistors as taken from the ITRS roadmaps dating from 1988 to the present[14]. Overlaid on this are red squares representing the reported feature sizes of actual microprocessors from the CPUDB database, and red diamonds are from more recent Intel Xeon data[6]. As can be seen, the downward trend has been declining exponentially for decades at a relatively constant rate, and is projected to continue to do so through 2024.

Fig. 3.3 then converts the decrease in feature size to an increase in transistor density. Given that area involves two dimensions that both improve with decreasing feature size, this curve approximates the square of the reciprocal of the feature size. The upward bumps in these graphs represent times when a change in layout for SRAM bit cells gave an extra boost to final density above and beyond the feature size decrease.

Finally, Fig. 3.4 expands the most recent data from Fig. 3.2. The points *below* the ITRS curve are from Intel microprocessor data (especially their high performance Xeons), and represent an interesting trend that has emerged over the last few years, where the best of the Intel technology

[6] http://en.wikipedia.org/wiki/Xeon

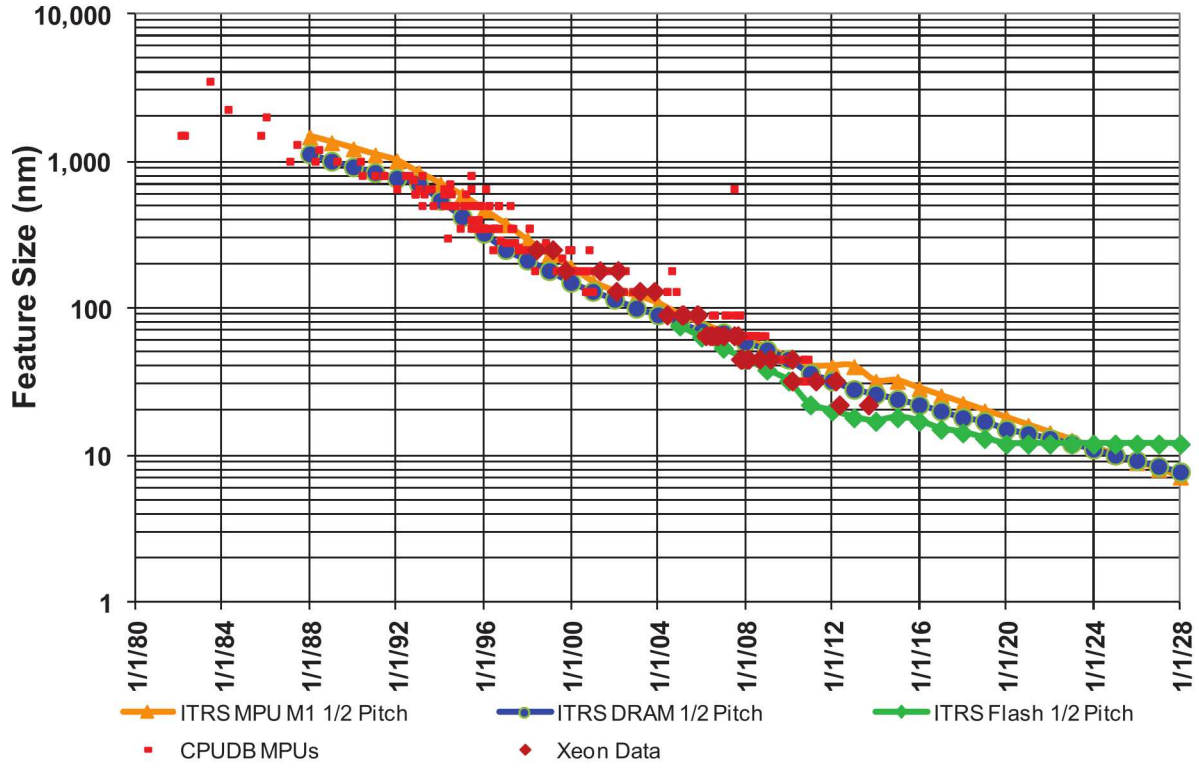


Figure 3.2. Feature Size as Reported by ITRS.

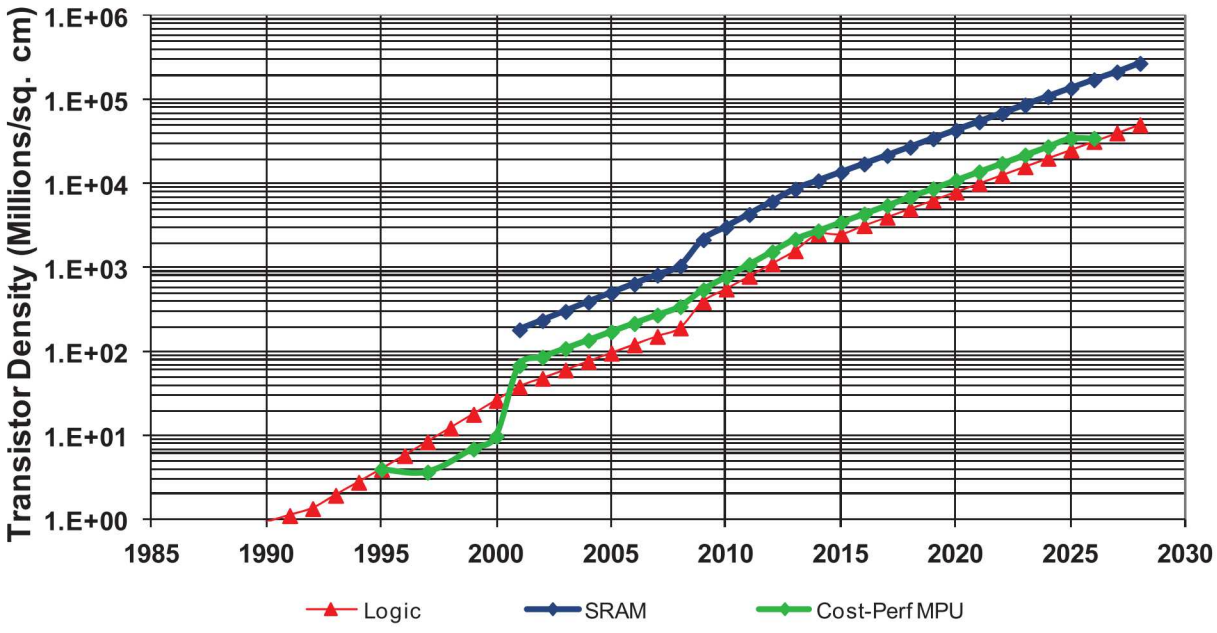


Figure 3.3. Equivalent Growth in Transistor Density.

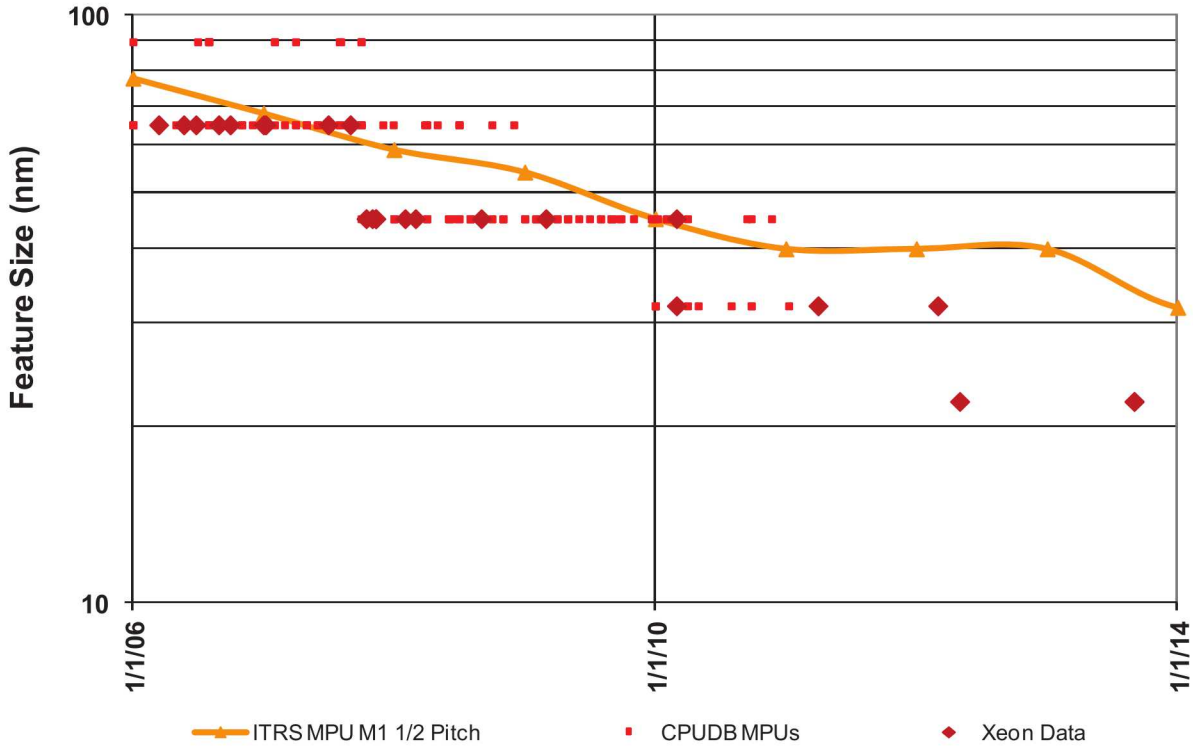


Figure 3.4. Expansion of Recent Intel Xeons.

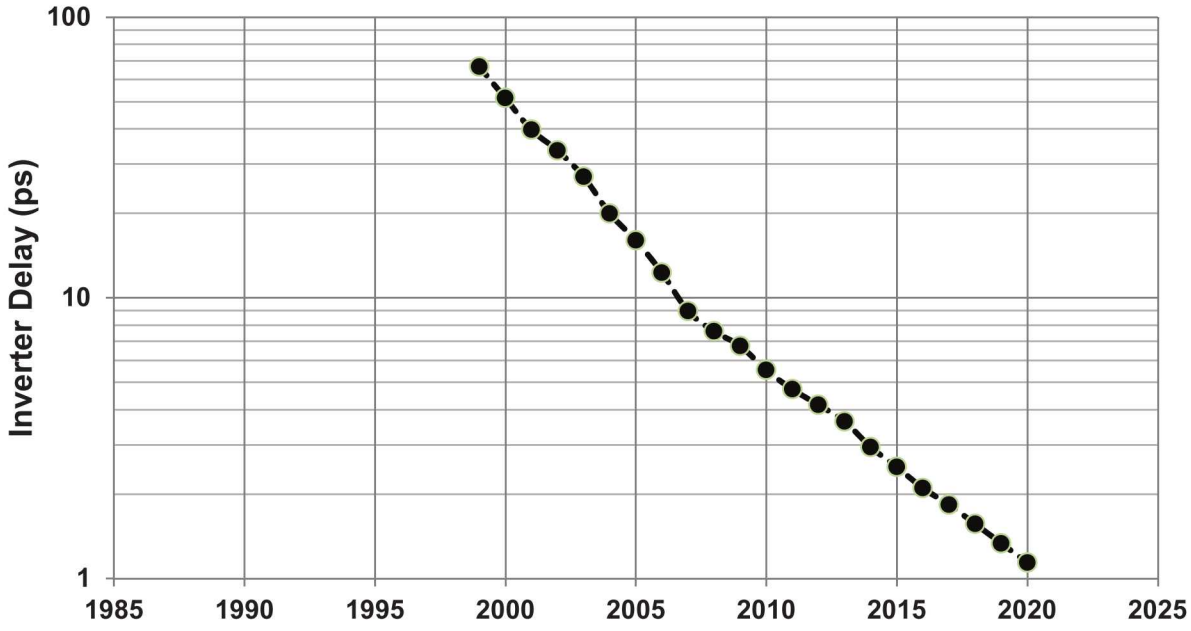


Figure 3.5. Inverter Delay as Derived from ITRS.

has about a generation lead over the ITRS "general volume production" feature size. It is likely that this trend will continue through at least 10nm, although the end of the line for both Intel and the general ITRS is the same.

3.1.2 Delay

This change in dimensions also affect circuit parameters. The big factor on circuit performance in CMOS is the time required to move a wire from a high to a low voltage, or vice versa. This is largely driven by capacitance, which is largely driven by the capacitance of the transistor's gate, which in turn can be approximated by the area of the gate over the height t_{ox}. If all these dimensions go down by a factor of 1/S, then the capacitance drops by something approximated by $(1/S^2)/(1/S) = 1/S$. which means the transistor is "faster."

Fig. 3.5 diagrams a projected decrease in the delay through an inverter (a two transistor circuit that converts a logic "1" to a logic "0" and vice versa). The ITRS has reported the "clock rate" for a chain of 12 such inverters, and the numbers here are 1/12th of the reciprocal of these numbers, taken from the ITRS 2006 roadmap. After this point, the numbers for clock rate reported by ITRS began to reflect various design constraints that will be discussed in Section 3.3, and not a projection of the actual intrinsic speed of transistors.

Fig. 3.6 graphs this ITRS delay as a function of feature size, along with a curve using computed parameters. Again we see an exponential reduction in delay, implying that besides getting smaller,

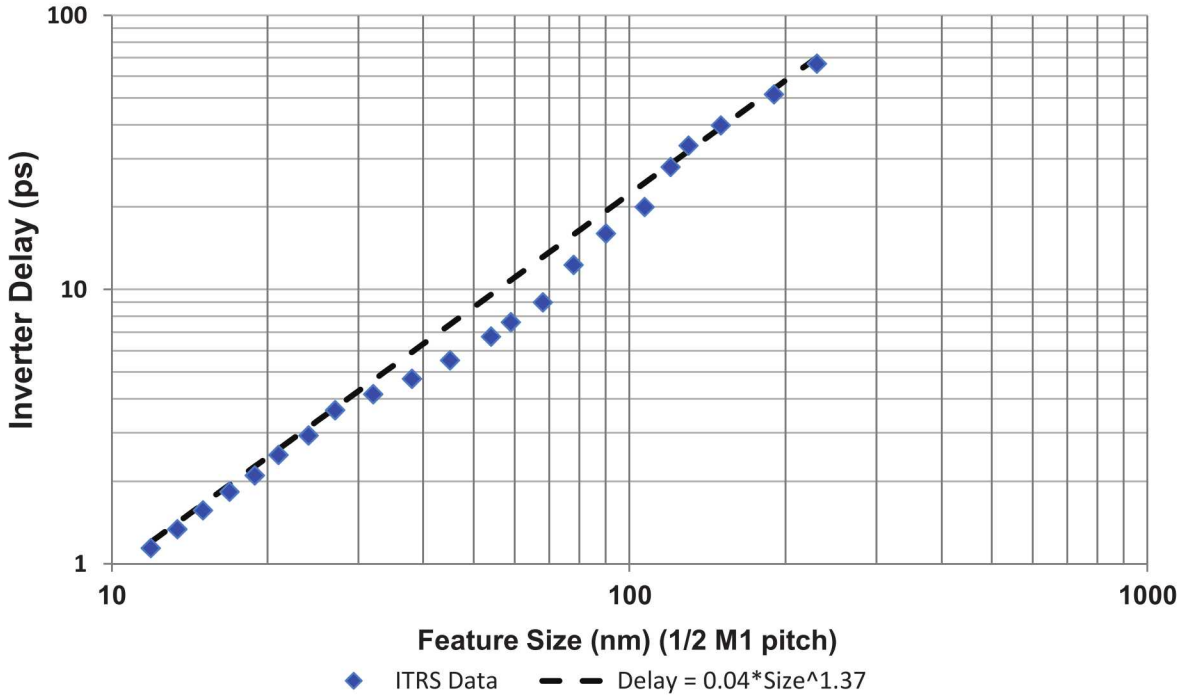


Figure 3.6. ITRS Inverter Delay as a Function of Feature Size.

transistors are in fact getting inherently faster.

3.2 The Halcyon Years

In the years before 2004, Moore's Law was used in two ways to increase the performance of single core microprocessors: use the speedup in the transistors to increase the clock frequency of the core, and use the increasing numbers of transistors to increase the work done per clock cycle. This double multiplier is what drove the normal interpretation of Moore's Law as increasing single core/single thread performance.

3.2.1 Operating Voltage

A rarely discussed by-product of this feature size decrease was the decrease in the operating voltage (typically termed V_{dd}) supplying the circuits. A major parameter to transistor operation is the electric field within a capacitor formed by the transistor's internal "gate," and as feature sizes fell, a lower and lower operating voltage was needed to keep this electric field more or less constant. Fig. 3.7 diagrams this voltage trend as a function of time for both high performance and low power

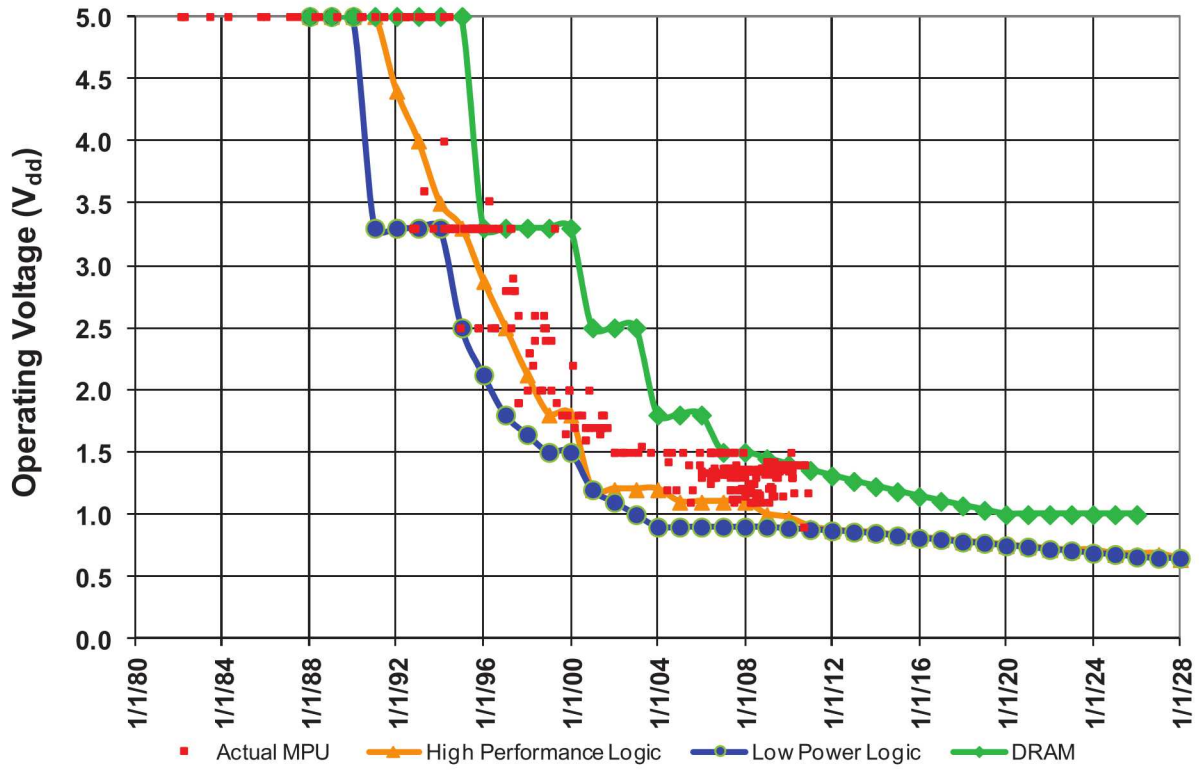


Figure 3.7. Operating Voltages.

logic chips and memory chips, again from the ITRS roadmap[14]. Again the red squares represent real data as reported by the Stanford website.

Also included in Fig. 3.7 is the operating voltage for commodity DRAM parts taken from [20] and other sources. This does not decline as rapidly or as far as logic because the internal structure of a DRAM array obeys some different constraints than logic circuits.

Lowering the voltage had a critically important side-effect. Power dissipated by a circuit is expressed by the classical equation $P = \alpha CFV^2$, where α is the percent of time during a clock cycle that capacitance is switched on average, C is the capacitance switched, F is the clock frequency, and V the voltage. Thus for a constant area of silicon, increasing the number of transistors and increasing the clock rate were at least partially balanced by the decrease in the capacitance per transistor times the square of the decrease in voltage. Thus the power dissipation per unit area was approximately constant.

At voltages below about 1 volt, transistor performance is reduced due to semiconductor physics, and so power reduction going into the future will be traded off against logic performance. This, coupled with the flattening shown in Fig. 3.7, has been one of the key factors in the transitions of 2004.

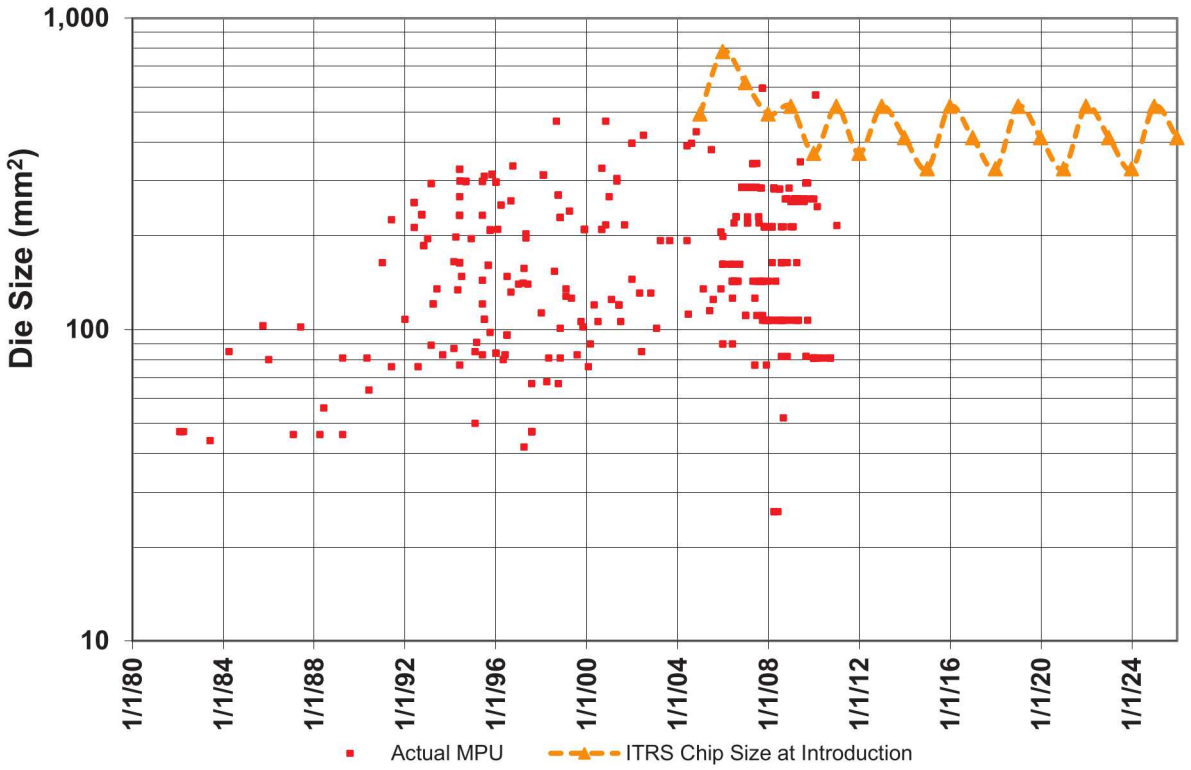


Figure 3.8. Die Area for Microprocessors at Introduction.

3.2.2 Die Size

A second phenomena of microprocessors in this time frame was that it became economically feasible to fabricate bigger chips. Thus not only did the number of transistors per unit area increase, but the total area of the chip increased, allowing even more transistors to be used in designs. Fig. 3.8 diagrams the die area for microprocessors at the time of their introduction, including real historical data that shows this increase. Fig. 3.9 diagrams production die area for all chip types, including DRAM, flash, and ASICs, again from the ITRS roadmap[14]. The production die area is typically smaller than the introduction die area, as they represent the result of technology shrinks between the first chip and production chips.

3.2.3 Basic Circuit Area Factors

A third phenomena is that as time goes on, the basic layout of key individual circuits continues to improve as engineers determine better ways of positioning the basic components (transistors, contacts, more layers of available wiring, etc.). Thus the "relative" area decreases in addition to the gains from smaller transistors. Such layout area metrics are typically given in units of "F^2," where F is the feature size of the technology. As an example for DRAM, F is typically 1/2 of the

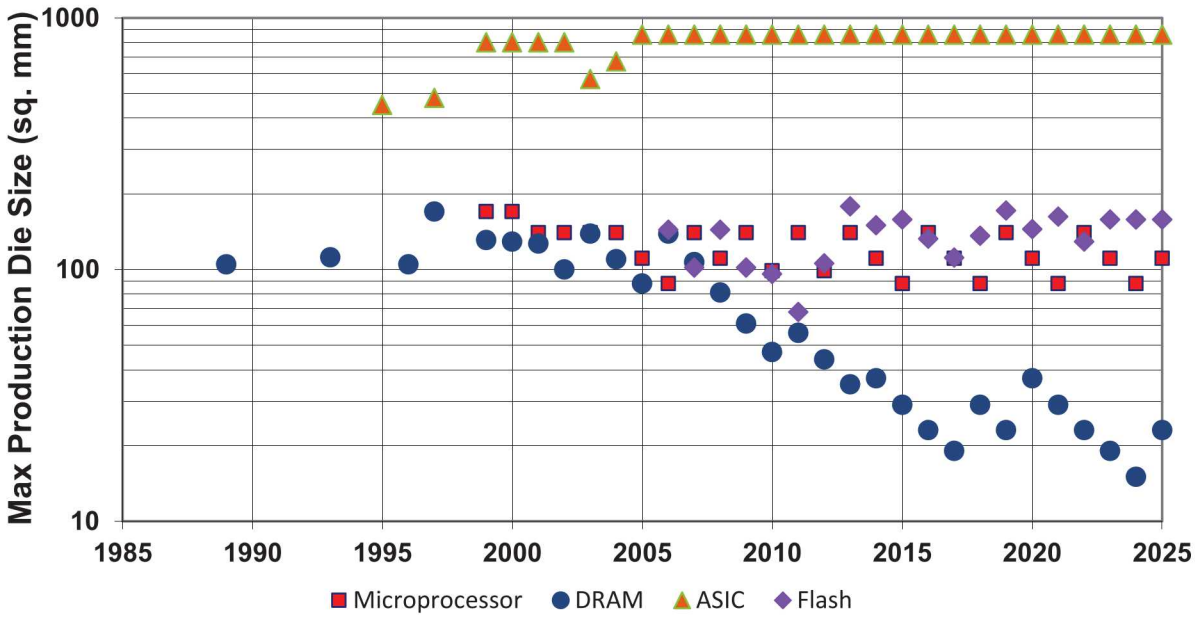


Figure 3.9. Die Areas for All Chip Types at Production.

minimum center-to-center "pitch" distance between two metal 1 wires (the layer of wiring that is closest to the surface of the die, and finest in dimension), so that the minimum pitch between two wires is 2F. Given that a DRAM cell consists of a transistor and a capacitor situated under an X-Y array of such interconnect, the minimum possible area is thus 2Fx2F, or $4F^2$. Fig. 3.10 plots these factors.

Over the time period plotted, the basic area of a logic or SRAM cell has improved by about a factor of 2, with all the projected change having already occurred. This means that the transistor density trends in Fig. 3.3 had a double gain in the early years: both smaller transistors and denser packing into basic circuits.

In contrast, there is about a 3X improvement overall from a basic DRAM cell, from about $12F^2$ down to about $4F^2$, with one final step from $6F^2$ today to the ultimate $4F^2$ left to go.

3.2.4 DRAM Memory

Fig 3.11 summarizes some of the major characteristics of the most important commodity DRAM memory over several product generations. The following sections address different aspects in more detail.

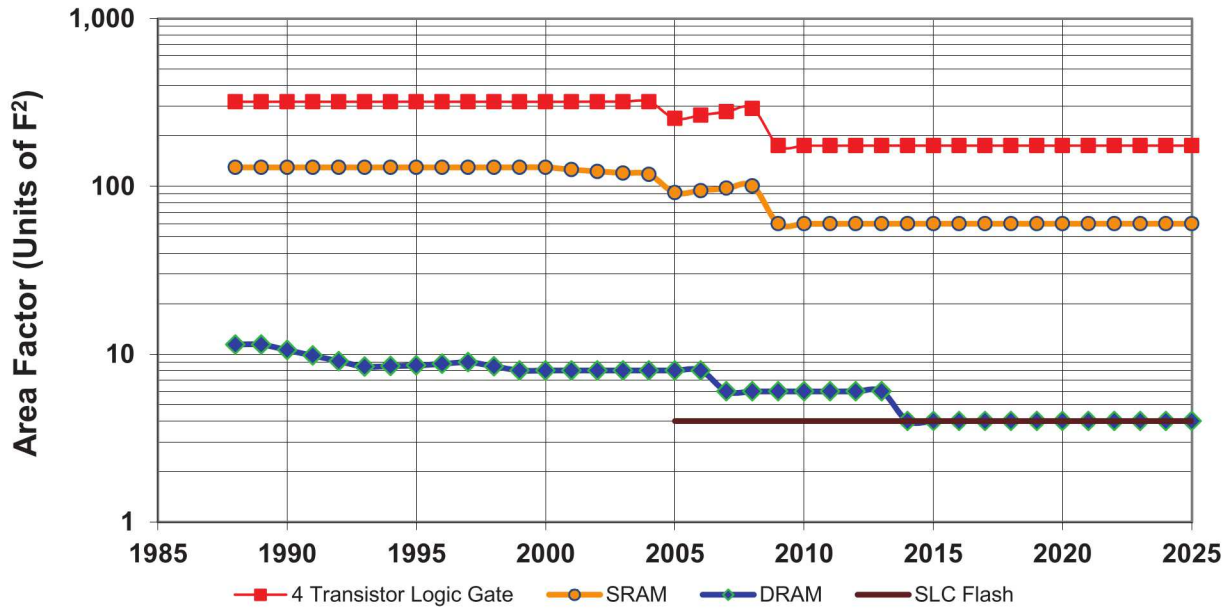


Figure 3.10. Area Factors of Basic Circuits.

Generation	Year	Type	Capacity (GB)	Banks	Bandwidth (GB/s)	Gen-Gen Bandwidth Improvement	Power (Watts)	Power/Bandwidth (pJ/bit)	Energy per access (pJ/bit)	Gen-Gen Energy Improvement	Energy-Bandwidth Factor	Gen-Gen Factor Improvement
SDRAM	1999	PC-133	1	4	1.1	1.0	5.0	583.1	762	1.0	0.01	1.00
DDR	2000	DDR-133	1	4	2.7	2.5	5.5	257.1	245	3.1	0.09	7.80
DDR2	2004	DDR2-667	2	4	5.3	2.0	5.2	121.4	139	1.8	0.31	3.54
DDR3	2007	DDR3-1333	2	4	10.7	2.0	5.5	64.7	62	2.2	1.38	4.48
DDR4	2013	DDR4-2667	4	8	21.3	2.0	6.6	38.7	39	1.6	4.38	3.18
Hybrid Stack	2011	Prototype	0.5	32	128.0	6.0	11.1	10.8	13.7	2.8	74.74	17.07
Hybrid Stack	2013	HMC	2	128	160.0	?	?	?	?	?	?	?

Figure 3.11. Typical Commodity DRAM DIMM Characteristics (abstracted from [29]).

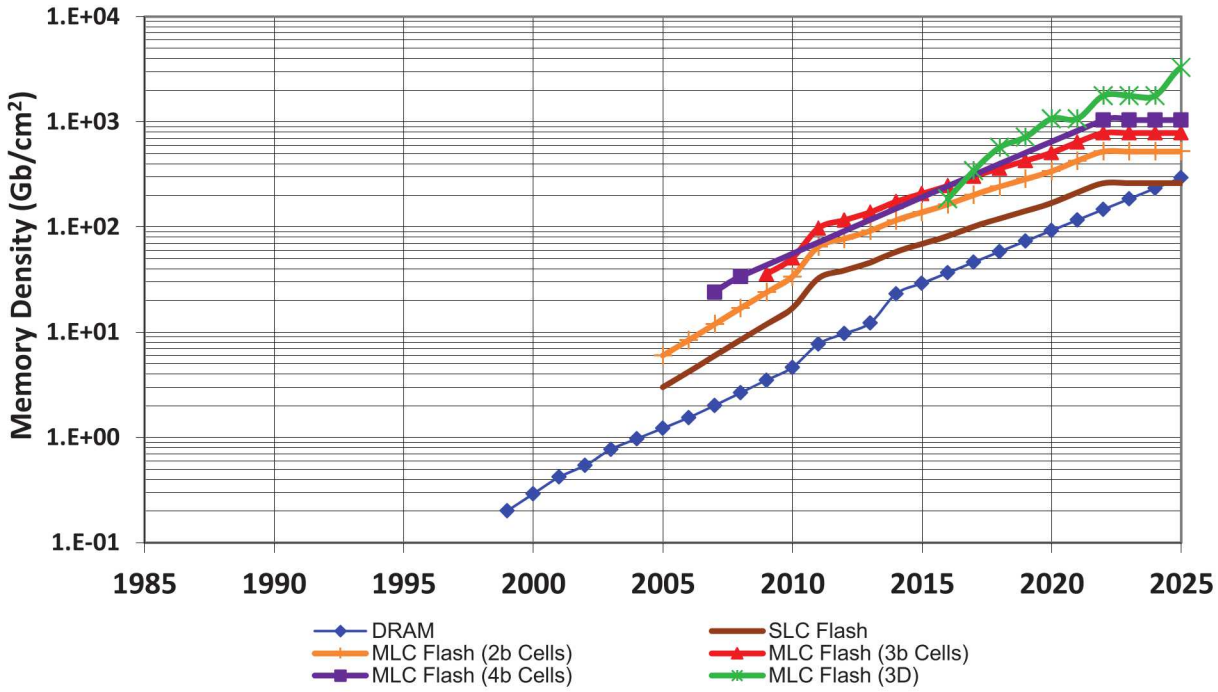


Figure 3.12. Memory Density.

Memory Density

The most obvious parameter for memory is density, how many bits are available for use. The density of memory chips has improved over time, as pictured in Fig. 3.12, again from the ITRS roadmap[14]. Memory types graphed include DRAM and NAND flash, where for NAND flash both single bit cells (**SLC**) and multi-bit cells (**MLC**) for 2, 3, and 4 bits per cell[7], along with 3D stacks of NAND flash.[8]

The density of individual memory chips of any of these kinds is driven by three factors: the basic Moore's Law shrink in transistor dimensions over time, the die area used for the chip, and the relative area of the layout of the basic memory cell that is replicated over and over in rectangular arrays. As can be seen in Fig. 3.9, a commodity chip's die area is actually shrinking over time, due to economic reasons. This is partially offset by the decrease in the area factor for DRAM (Fig. 3.10), but the overall effect is to slow the growth in "bits per chip" for commodity DRAM.

[7] 2 bits per cell is common in commodity products today; 3 bits is in production but is not being used where reliability due to wear-out is a concern; no 4-bit cells are approaching production.

[8] The term "3D" here is not the same as used later for 3D stacks of die; instead it refers to a single memory die where multiple layers of transistors are fabricated on top of the die's surface, as in [4]. Stacked NAND has just entered production, with up to 24 layers possible; see http://www.samsung.com/global/business/semiconductor/news-events/press-releases/detail?newsId=12990.

Memory Concurrency

While density has increased, the circuit constraints of a dense memory circuit are, however, considerably different that a logic circuit, so intrinsic memory access time (termed its **row latency**) has improved at best slowly during this time (a widely reported average improvement is about 7% per year for DRAM). Further, at least for DRAM, the time to transfer data off chip is so much less than most DRAM's cycle time that without the ability to overlap multiple access requests and supply multiple data transfers (data burst) for each reference, the interfaces of a standard memory part would be idle a very large percentage of the time.

To prevent this, the micro-architecture of most memory chips[31][15] divides the on-chip memory into separate **banks**, each of which has enough independent control logic that it can be performing a separate access while some other bank is transferring data through the off-chip interfaces. DDR2 chips usually supported 4 banks; DDR3 chips support 8 banks.

Accessing DRAM memory then becomes a two-step process. First a **row access** request is sent through the chip to the desired bank to perform an access. The results of the access save somewhere between 1024 and 2048 data bytes per access in what is called a **row buffer** associated with the bank. Then a second, and independent, **column access** request may select and access typically 32 or 64 bits of this row buffer for transfer through the chip's interface (at 4 or 8 bits at a time, depending on the DRAM's I/O Interface).

We note that a write to such a memory requires a row access to retrieve the entire associated row buffer, followed by a transfer from the outside into just the part of the row buffer associated with the location to be changed, followed by a restore operation to write the row buffer back into the bank.[9]

The number of such banks that may be simultaneously busy is a measure of a memory chip's potential for **concurrency**. Thus a DDR3 DRAM may support up to 8 separate and concurrent accesses, as long as each is associated with a separate bank.

In a commodity DRAM DIMM[10] today, some number of such chips are ganged together on a small card in groups that are referenced together; each such group is called a **rank**. All chips in a single rank (or all chips on the DIMM if there is only one group) receive exactly the same commands at the same time, and perform identical operations, with the only difference is that each chip contributes to a separate set of data bits on the overall interface.

Typically a **memory controller** sits between a microprocessor and such a memory. The memory controller accepts a stream of access requests from the cores within the microprocessor, and schedules them so that the concurrency within the memory, and thus the achievable memory bandwidth, is maximized. This scheduling is typically quite complicated, with multiple internal queues, especially as we have moved from single core, in-order, designs to multi-core, out-of-order, designs where requests from different cores can arrive with no address or timing relationship between

[9]The read operation in a DRAM is "destructive," reading any row erases the data within that row. At the end of a reference the row buffer must be written back to the original location in memory to restore the DRAM contents.

[10]DIMM stands for Dual Inline Memory Module.

them.

For memory channels with multiple ranks (either on the same DIMM or on different DIMMs), it typically takes some significant time to switch the memory interface from communicating with one rank to communicating with another. This is due to the usual differing clock phasing between different ranks, and and the electrical switch time to turn off one set of drivers and another set turned on along with timing uncertainties. While multiple ranks increase total memory capacity, the time to switch the interface between them can complicate the memory controller's scheduling decisions, but also improves the average sustained memory bandwidth somewhat, while not affecting the peak bandwidth.

Memory Bandwidth and Power

Memory bandwidth is the rate at which data can be transferred to and from a memory, and is driven by both circuit and micro-architectural considerations as discussed above. In cases such as DRAM DIMMs we can also isolate and discuss **memory power** as it relates to the discrete memory system. Since most power is drawn when a DRAM is busy, perhaps a more important metric is an amalgam of bandwidth and power, namely how much energy does it take to access one bit of data and transfer it in or out of the DIMM. Multiplying this by the amount of data that is desired for some operation, and dividing by the time frame over which the accesses are needed then gives an estimate of average power dissipated by a DIMM.

Fig. 3.11 includes the power and bandwidth of several generations of DRAM DIMMs (abstracted from [29]), along with numbers for a next-generation DRAM memory stack as discussed in 3.5.2. These generations include both technology and architectural improvements.

For each generation the ratio of power to bandwidth gives an approximation of the energy in pJ/bit to access and transfer a single bit of data. Fig. 3.11 includes both this and an adjusted set of numbers that reflect some additional system considerations (taken from [29]). Additional columns then give the improvements in bandwidth and energy on a generation over generation basis. The second to last column then gives a combined factor reflecting an "energy-bandwidth factor" (bandwidth over energy) that reflects the ability to access data both faster and and at lower energy, and where bigger is better. Again there is a generation over generation improvement ratio.

3.2.5 Power Dissipation

As discussed above, power dissipation (usually as heat) is roughly αCFV^2, where the aggregate capacitance C is a function of the capacitance of a transistor and the number of transistors, which is itself a function of the area of the die, and of the capacitance contributed by wires between circuits. Clearly if heat dissipation per unit area was roughly constant, then bigger chips, as indicated by Fig. 3.9, dissipated more heat. Inexpensive air cooling technology, however, was able to keep up so that power per die went from a watt to in excess of a hundred watts. Much beyond a 100 watts per chip requires alternative techniques, which in turn greatly increases the cost of a packaged

system. Examples of more exotic cooling include:

- **liquid cooling** where a fluid such as water or a fluorocarbon, is run through pipes next to the hot chips and then out through a heat exchanger.
- **micro-fluidic cooling** is a variant where the "piping" is often small grooves under the chip's surface, and where the fluid thus goes "through" the chip.
- **immersion cooling**[11], where the entire system is immersed in typically an oil-liquid, without the need for piping, and where the temperature of the liquid is monitored, and pumped as needed through a heat exchanger to cool it.
- variations of the above where the fluid undergoes a phase change (especially from liquid to gas) to remove heat via evaporative cooling. A **heat pipe** is an example of liquid cooling where the cooling pipes from a closed system with no pumps, and where the phase change to a vapor provides a pressure difference to drive the fluid back around the system once it has cooled. Variations of immersion cooling use liquids that boil at a rather low temperature (such as 3M fluid Novec 649[12] which boils at $49°C$), and where the vapor rises to a region with a cold plate where it is converted back to a liquid and drops back into the electronics.[13].

3.3 2004 - The Perfect Storm

In 2004 a series of technological barriers were hit almost simultaneously that stopped the predictable growth in capability in its tracks as discussed above:

- The ability to inexpensively extract heat from chips of any size maxed out.
- The ability to lower voltages with decreasing feature size slowed dramatically.
- The design complexity of single core microprocessors hit a point of diminishing returns where more transistors could add little to the per cycle performance of a core.
- We are approaching a limit on economically viable off-chip interconnect with the technologies in use at the time, because of electrical issues.
- The cost of increasing wire-based signalling rates also began to grow considerably, especially in power and complexity of the interface circuits.
- The economics of memory chip production stopped the growth in size of memory die.

[11] see for example http://www.grcooling.com/

[12] http://solutions.3m.com/wps/portal/3M/en_US/3MNovec/Home/ProductCatalog/?PC_Z7_RJH9U5230OOA50IEKHCMDN11H0000000_nid=F55Z1XTKWXbeQQBXSJ1LVVgl

[13] http://solutions.3m.com/wps/portal/3M/en_US/NA-DataCenters/DataCenters/AboutUs/3MInnovation/

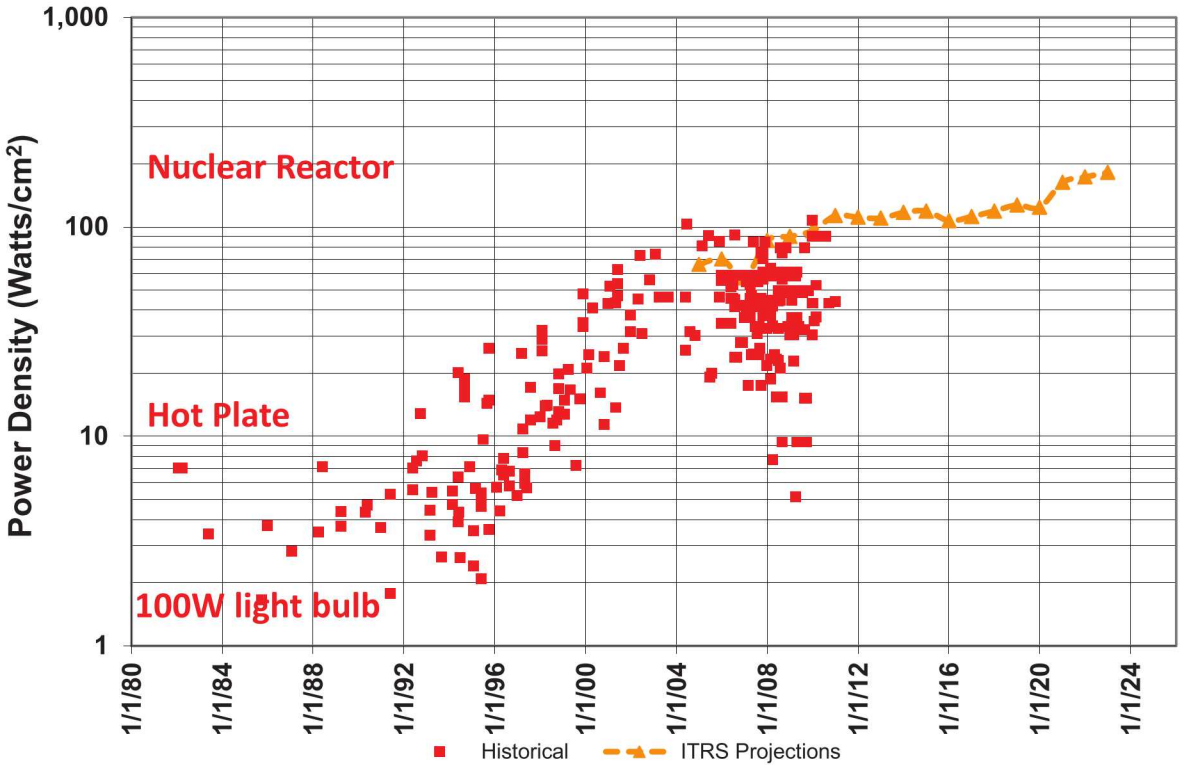


Figure 3.13. Microprocessor Power Density.

- The electrical and power issues associated with driving off of a memory chip at high rates through inexpensive commodity packaging (such as found on commodity DIMMs) to a microprocessor chip more than a few inches away reached a point where further improvements become fairly difficult, power-hungry, and/or expensive.

3.3.1 Power Density

Fig. 3.13 diagrams some historical data for the rise in power density (watts dissipated per square cm of die area), including the projections from ITRS as to maximum power density going forward. From 1975 to 2005 power density went up three orders of magnitude. For reference, the power density of several hot objects is included as reference. It is clear why heat sinks started to be needed in the late 1980s' and had to increase significantly in complexity over time. It is also clear why further increases after around 2004 were impractical for inexpensive mass production.

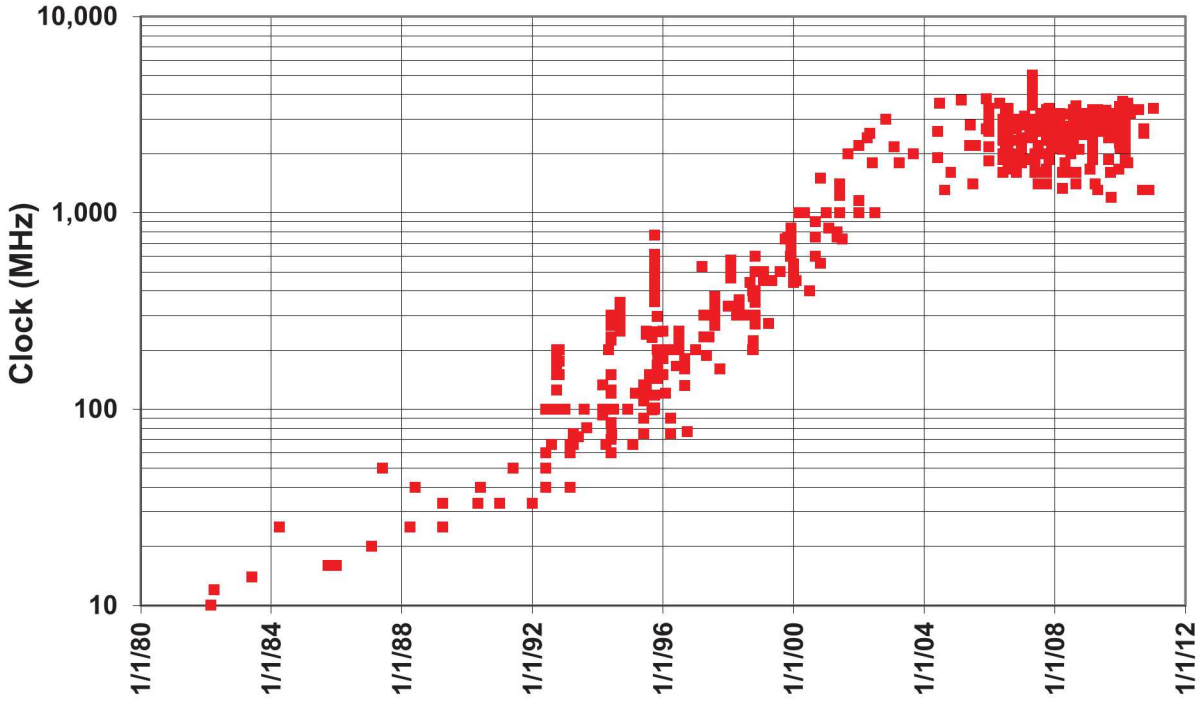


Figure 3.14. Historical Clock Rates.

3.3.2 Operating Voltage

Fig. 3.7 also demonstrates the change of slope for operating voltage after about 2001. Before that we had seen a 4:1 reduction in voltage; going forward there was at best a factor of 2, with perhaps a 33% decrease between now and the end of the roadmap. Further, the difference between "high performance" and "low power" logic is also shrinking. This largely removes the ameliorating effects that the squaring of voltage had on overall chip power.

3.3.3 Core Clocks

A vivid example of the effect of this power limit is on the reporting and projection of on-chip clocks on logic chips. Fig. 3.14 provides some historical data on microprocessor clock rates. The 3 GHz ceiling is obvious from 2004 on.

To understand how this ceiling compares with basic technology, Fig. 3.15 is an overlay of several ITRS yearly projections of on-chip clock rates from 1999 to the present, and projections into the future. Before 2006 this was based on a chain of 12 inverters as discussed in Section 3.1. Also present on this chart is the maximum clock rate of any system in the top 10 of the TOP500 list from 1999 through now.

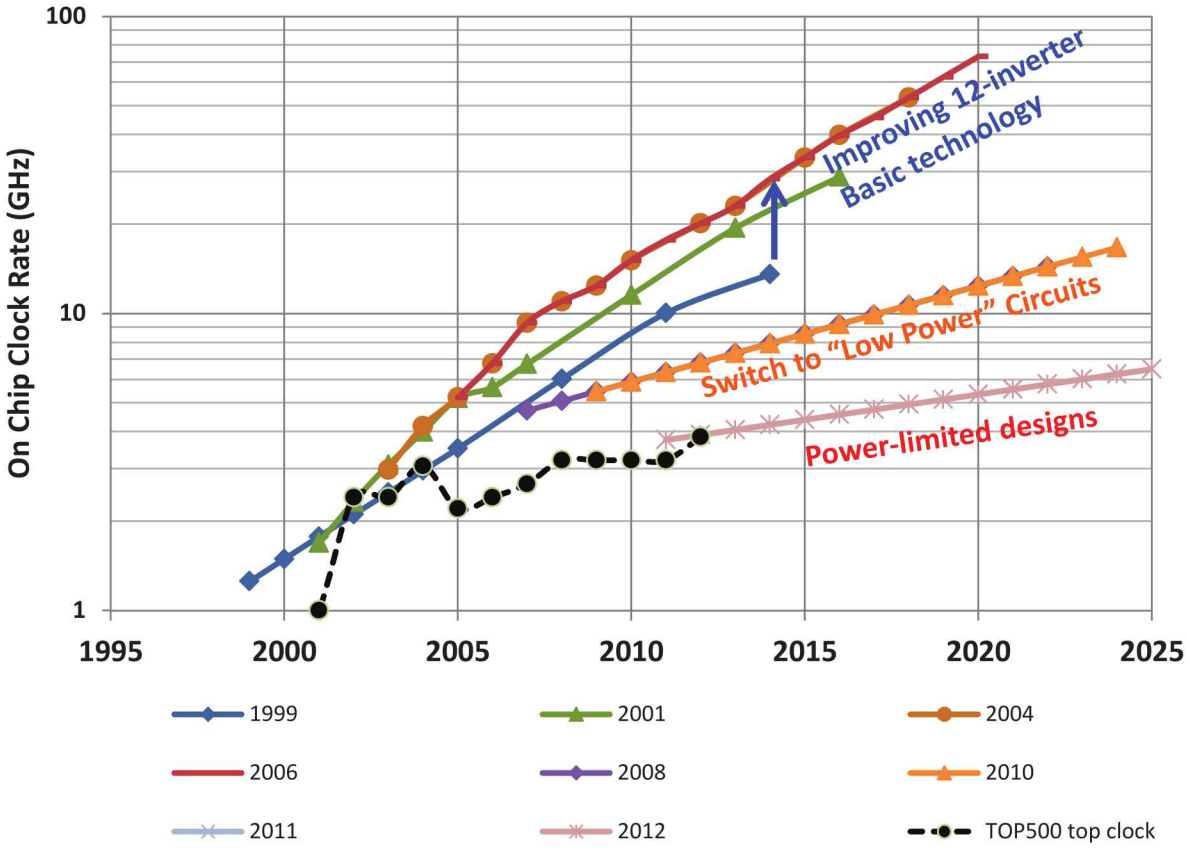


Figure 3.15. On-chip Clock from ITRS.

As can be seen, the ITRS projections through 2004 matched reasonably well the fastest microprocessors used in TOP500 systems. After that the projections from the ITRS roadmaps got considerably better, with the 2006 numbers predicting a 20GHz clock by 2012, with a CAGR of almost 18%.

These projections were right in terms of the basic transistor characteristics, but failed to take into account the design constraints introduced by the perfect storm. In 2007 through 2011 the ITRS down-graded their projected clock rates by assuming alternative low power inverter circuits that they believed reflected a "low power" design philosophy. The CAGR here was about 8%. Even these, however, proved to be too aggressive, so starting in 2012 they assumed a much more conservative projection assuming very heavily power-limited designs, with a CAGR of a mere 4.3%. At least for 2012 these estimates appear to reflect reality.

The key take-away from this is that the best of the pre 2006 curves, even though they are dated at this point, reflect relatively well the intrinsic capability of CMOS technology, but that looking forward, real microprocessors will be giving up something around an order of magnitude in on-chip clocking rate because of the design issues discussed previously.

3.3.4 Off-Chip "Per Wire" Signalling Rate

Microprocessor chips do not exist as one-chip systems, especially for HPC systems. There are off-chip memory and I/O that must connect to the microprocessor via wiring on the motherboard that must connect to contact pads at both ends. Clearly, as the complexity of the cores and their clock rates increased, there was a need to increase the bandwidth of data between chips. This bandwidth is driven by the number of wires over which data is being transferred, times the clock rate for the signalling, times the number of bits that may be transferred per signal clock. This last parameter was typically a "1" until perhaps a decade ago, when **double data rate** (**DDR**) and **quad data rate** (**QDR**) protocols permitted more bits to be transferred per cycle. DDR, for example, transferred a different bit on each of the rising and falling edges of the clock.

The product of the last two terms (clock rate and bits per clock) is a raw bit transfer rate which must be derated by bits needed for overhead, including communication management and/or error detection and correction (ECC) to data within the packet. Ignoring this overhead, the key technology characteristic is the bits of information transferred per wire per second, often called the **signalling rate**.

We will term a path for a single signalling path as a **lane**. Such a wire can be **uni-directional** where there is a distinct transmitter on one side and a receiver on the other, or **bi-directional**, where there is a combination transmitter and receiver (called a **transceiver**) on both sides. In most bus-based systems today a single lane is a single wire, and multiple such wires work together to transfer aggregates of data, with a protocol that permits the direction of transmission to change dynamically. Each such wire needs one contact on each end to function. Allowing for simultaneous signalling in both directions (called *full duplex*) typically involves 2 wires, one dedicated for signalling in each direction.

There are many cases, however, where, for electrical reasons, to increase the signalling rate, two wires are used in what is called a **differential pair** to make up a single lane. With such protocols, both wires need to connect to different contacts on both sides of the wire, effectively halving the signalling rate "per contact." The upsides of differential signalling are: much faster signal rates (2 - 5 times and more), ability to go longer distances, reduced signal levels, greatly reduced generated electrical noise, and greatly reduced sensitivity to nearby electrical noise sources.

Perhaps the most important such signalling rate is between memory and a microprocessor. Early on, the data transfers between the two were over a wide parallel bus (typically 32 to 64 data bits), often called the **front side bus**. Starting around 2000 the gating item in such transfers was not the microprocessor but the memory technology and associated economics, so memory bandwidth became dominated by commodity memory chip technology. Synchronous DRAM (SDRAM) became popular, where the data rate from a memory chip is tied directly to an explicit clock signal. This quickly led to DDR parts that transferred 2 bits per clock. Fig. 3.16 shows these single wire rates as taken from [30] for several generations of parts. Also shown for reference is a CAGR curve of about 17% from the first DDR chip through the expected end of DDR4 chips;[14] following

[14]DDR4 chips are in line for availability in 2013; their high cost, however, appears to be delaying their widespread introduction into widely available commercial systems.

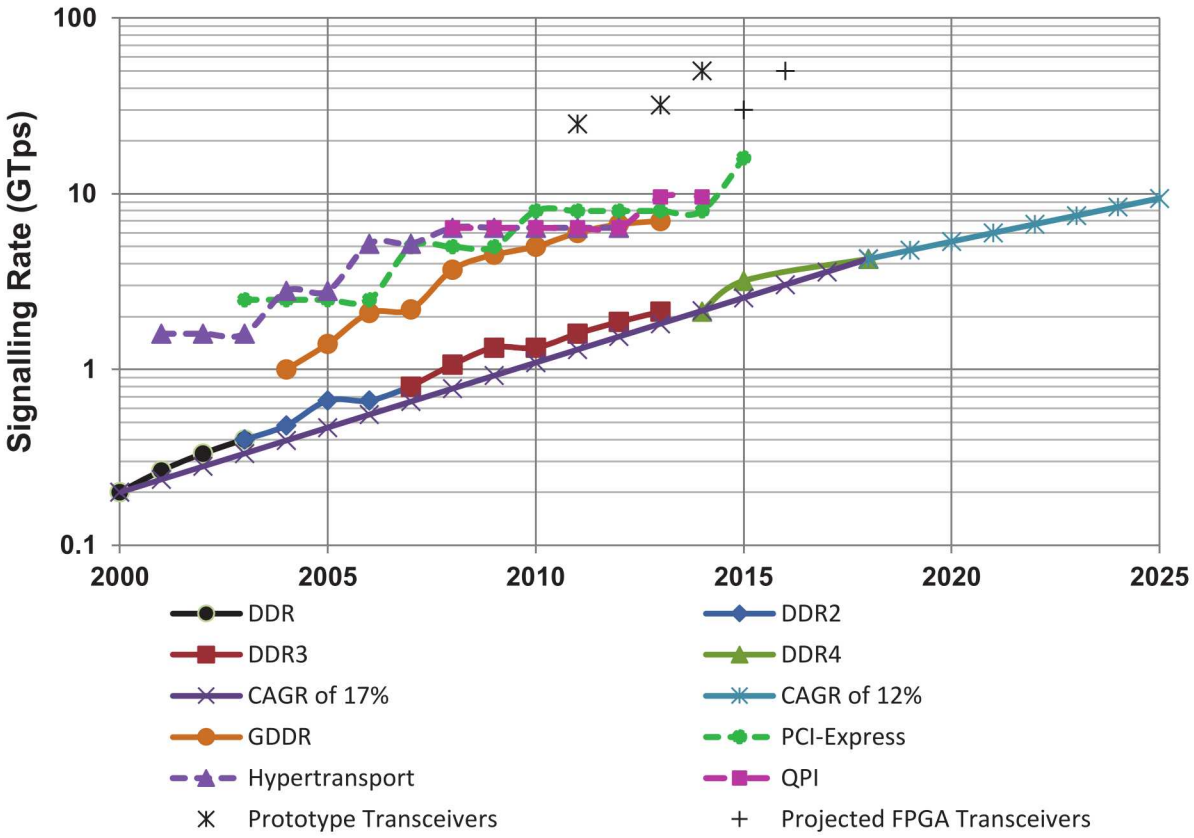


Figure 3.16. Signalling rates on Chip-Chip Wires.

that, the projections from [30] are down to a 12% CAGR.

Also shown are signalling rates for **Graphics Double Data Rate** (**GDDR**) memory chips, which are used typically by GPUs to provide higher speed access to memory (data taken from [13]). These parts use larger transistors to drive higher signalling rates and more off-chip wires per chip over which to transmit data, but have significantly higher power, lower memory capacity, and thus higher costs than the commodity types.

Next, a variety of alternative signalling protocols have been developed that allow higher signalling rates per wire, but fewer wires per link. These protocols are designed for strict point-to-point links, where, unlike DDR protocols (where multiple DRAM chips can be tied to the same physical wire), there is exactly one chip on each side. Further, wires are paired into **lanes**, and where a unique wire in a lane pair transmits data in exactly one direction. This restriction in the number of chips permits more complex equalization circuits to adjust the waveforms transmitted to overcome electrical signal distortion and loss in wires so that signal receivers can correctly determine the information content. Protocols that use such signalling include **PCI-Express**^{TM}(**PCI-E**), **Hypertransport**^{TM}, and **QuickPath Interconnect**^{TM} (**QPI**)^{TM}. Fig. 3.16 includes some recent values for transfers per second for these interfaces. As can be seen, while there was across the

board growth between 2004 and 2008, followed by a flattening around 8Gbps until recently, especially with the projection of PCIe moving in the near future to 16Gbps, and the announcement by Micron of 15Gbps rates for its Gen-2 HMC parts.

It appears that recent developments in transceivers built from 28nm and smaller technologies will fairly rapidly provide even further increases, especially in the realm of short distances. More sophisticated **SERDES** (the "serializers/deserializers" that make up the key parts of a transceiver) are applying more complex waveforms (such as **PAM-4** that will double the bits per clock from 2 to 4), and tailoring how far they can communicate to specific max distances, such as:

- backplane: a few dozen inches
- short reach: a few inches over a high quality printed circuit board
- very short reach: even short distances on a package such as an **MCM** (Multi-Chip Module)
- ultra short reach: chip to neighboring chip with a silicon interconnect as a substrate for the signals

There have been multiple demonstrations of 28 and 30 Gbps transceivers in the recent past[15] [16], and there is are early prototypes at 50+Gbps[17]. The International Solid-State Circuits Conference publishes a yearly summary of such trends[1].

Several FPGA vendors[18], [19] have announced the inclusion of such transceivers in commercial products in the next few years, with the goal of being able to directly drive backplane distances. Given this, more widespread adoption should be expected into other products around the same time.

3.3.5 Off-Chip Signalling Power

Signalling on wire is not immune to the αCFV^2 phenomena that affected logic so heavily. The capacitance of contacts and wire between chips has not changed appreciably, meaning that raising the signalling rate increases the power consumption of the logic that must drive such wires. The increase in off-chip signalling rate for DRAM is illustrative for the effects on any off-chip signalling. As shown in Fig. 3.16, the signalling rate for DRAM DIMMs using a shared DDR-x protocol to microprocessor shows a CAGR of about 17% through 2013, nicely matching the expected improvement of the basic transistors as discussed above. Part of the reason this was possible was that the average DRAM chip has only an average 4-8 off chip data wires to run at this rate.

[15] http://www.chipestimate.com/tech_talks/2011/09/20/MoSys-25-28Gbps-SerDes-Design-and-Implementation-Challenges

[16] http://newsroom.altera.com/press-releases/nr-20nm-device-altera.htm

[17] http://www.lightwaveonline.com/articles/2014/04/credo-semiconductor-offers-serdes-for-55-gbps-nrz-signaling.html

[18] http://www.altera.com/devices/fpga/stratix-fpgas/stratix10/stx10-index.jsp

[19] http://www.xilinx.com/products/silicon-devices/fpga/virtex-ultrascale.html

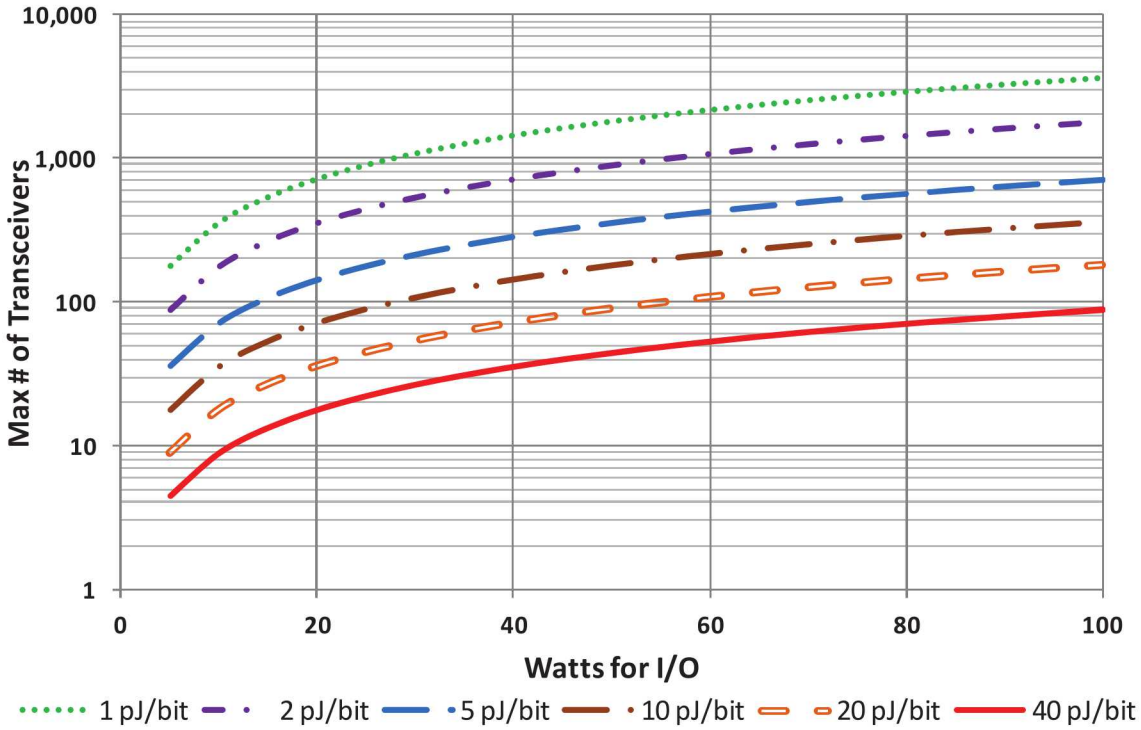


Figure 3.17. Bounding the Number of Transceivers per Chip as a Function of Power.

The specialized GDDR parts used for high speed graphics and now GPUs do show a higher clock rate but the effects of the increased power dissipation and a common interface of 16 or more data wires per chip has already dropped its actual growth rate well below that of the lower power DDR DRAM interfaces. After 2013 the projections for commodity DRAM drops to a 12% CAGR, with what appears to match the current trend in the GDDR component class, which also appears to be flattening rapidly.

Looking at the highest signalling rates discussed in the prior section, power levels become even more significant. The best reported number to date is 20pJ/bit[17] for a 28Gbps transceiver, which is equivalent to 20mW per GHz, or 560mW per transceiver.

To get a sense of why it is so important to lower this power, Fig. 3.17 diagrams how many transceivers could be supported on a chip for a given power budget (in Watts) devoted to just I/O, as a function of the energy cost (in pJ/bit) of the transceivers. The line corresponding to 20 pJ/bit corresponds to the best reported transceiver to date. Since we wish to look forward, we assume 28Gbps high-speed links.

If we wish to spend no more than say 40 Watts (perhaps 1/3 a high performance chip's total power), then with the best of today's transceivers, less than 100 transceivers can be supported. As support for this conclusion, the most leading edge FPGA chips projected in the near future in

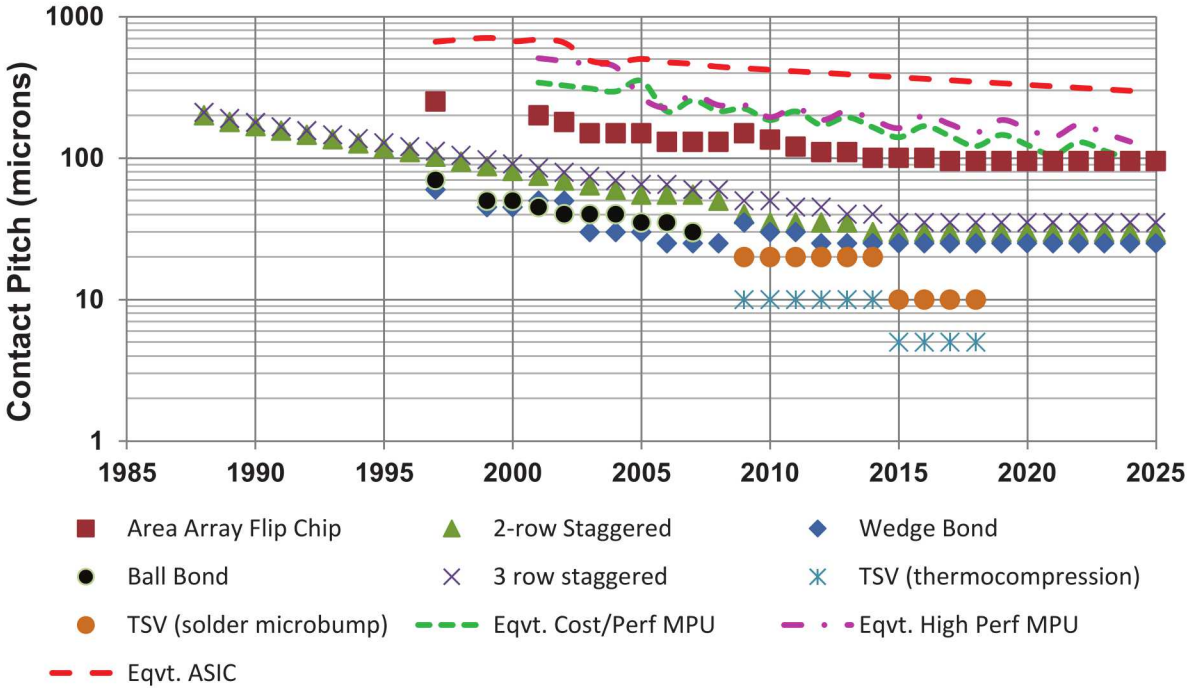


Figure 3.18. Pitch Between Chip I/Os.

fact have maximum transceiver counts in this range. Even considering that each transceiver is 2 contacts (out-going and incoming), this is perhaps 1/5th of a commodity microprocessor's signal I/O count (according to the ITRS), and far less than 1/10th the fraction of the signal contacts for an ASIC.

Board technology with better electrical properties can improve signalling power and distance, but industry has largely avoided this for wide-spread commercial use because of cost. Differential signaling, in particular, will offer lower energy per bit than single-ended, particularly at higher data rates, but at the cost of multiple traces per signal.

3.3.6 Off-Chip Contacts

Besides signalling rate, the drive to higher chip-to-chip bandwidths is also affected by the total number of signal I/Os possible from modern microprocessor chips. Fig. 3.18 diagrams the reduction in "pitch" between pads for different types of chip I/Os over time. This pitch is the minimum center-to-center distance between two contacts that are not electrically tied together.

When looking at how such contacts can be placed on the surface of a silicon die, there are two types: **peripheral** and array. Peripheral contact types can only be placed around the periphery of a chip, perhaps in 2 or 3 rows. Thus the maximum number of contact is the perimeter of the chip divided by the pitch, times the number of rows. Array contacts, however, can cover the entire area

of a die, with a total number equalling the area divided by the square of the pitch.

With current packaging, the effective number of "pads" that carry signals off a chip to a socket or motherboard is less than what might be guessed from these pitches. Fig. 3.19[20] lists these projections for current technology (non-TSV). Also, on average only between 1/3 and 1/2 of these projected contact pads can carry distinct logic signals; the rest carry power and ground.

Clearly, the projections going forwards for improvement in the pitch between contacts is that, after 2015, there is none. When coupled with the flattening of chip area as demonstrated in Fig. 3.9, the best that conventional technology projects for is a linear increase in off-chip signals of about 1,500 contacts every 5 years. This is far less than the exponential growth we have seen in most all other metrics.

The dotted lines back on Fig. 3.18 take the total counts for the MPUs from Fig. 3.19 and compute backwards to an "equivalent" pitch. As can be seen these equivalent pitches are currently 2-3 times larger than the area array, but are converging slowly towards it. Part of the difference is that not all chip area can be covered with contact pads. More importantly, however, the substrates to which the chips are flip-bonded to cannot support the maximum density. Wires on substrates must be wider than wires on a chip, and thus not so many can "escape" from a single layer under the die. Increasing the number of layers in the substrate, with different depths of vias from the top, can increase the number of such wires, but only at significant expense in packaging. The decline in effective pitch in the dotted lines of Fig. 3.18 are due largely to assumptions about more layers on the substrate and also with improved chip mounting techniques, but both incur increased cost.

3.3.7 Off-Chip Contacts versus Transistor Count: Rent's Rule

Rent's rule [32] describes an observed relationship between the number of **terminals**[21] presented by a circuit to the complexity of the circuit (number of transistors, gates, ..). Using data from IBM computers in the 1960s', Rent found that the relation $T = tG^p$ (where T is the number of terminals, G the number of gates in an identified logic function, and t and p are constants) fits real data of the time with great accuracy. The exponent p lay between 0.5 and 0.8. Later studies[5][22] found very similar relationships for much larger units than simple gates, with the "p" term in tG^p lying between 0.3 and 0.7 depending on the complexity of the functional unit.

Fig. 3.20 diagrams the projections from the ITRS for three classes of microprocessor chips. As can be seen, the data once again fits the shape of Rent's rule, but with a much smaller exponent, around 0.2. Thus the available contacts grow at a rate far less than any of the projections of the needs of traditional circuit blocks.

[20] "MPU" in this figure refers to "Microprocessor Unit."

[21] In this context a terminal is the point in a circuit where the signal (either an input or output) is made accessible to other circuits.

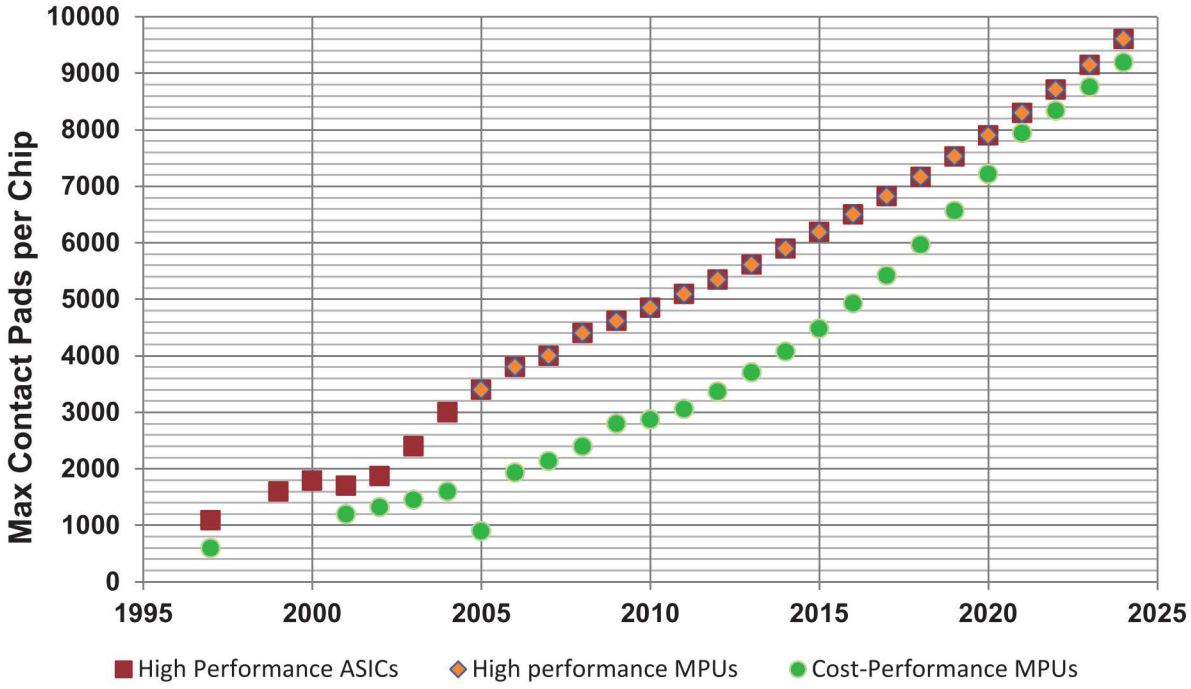


Figure 3.19. ITRS Projected Microprocessor Pad Count.

3.3.8 Maximum Off-chip Bandwidth per Unit Logic

An even more interesting metric is shown in Fig. 3.21, where the maximum available signal contacts for a high performance microprocessor are assumed all used for off-chip data transfers using either the best of that year's DDR memory, or GDDR memory, or I/O protocol (PCIe, HT, QPI, or something new based on the best available transceivers), as graphed in Fig.3.16. The product of the number of signal contacts is multiplied by the maximum rate for each type of I/O, and then divided by the maximum number of transistors (in units of a million) that can be placed on such a chip.[22] Thus if a core consumed a million transistors, regardless of feature size, this chart then gives insight into how much exclusive off-chip bandwidth might be available to just that core at each point in time.

The large jump after 2015 is assuming that the very highest speed transceivers of Section 3.3.4 are used on all contacts.

The numbers here are wildly optimistic because not all signal contacts can be data transfers (there are many control signals to account for), and, especially for GDDR and I/O, the power of running so many high speed I/Os at the same time overwhelms the power problems we already have for the cores. As discussed in Section 3.3.5 and pictured in Fig. 3.17, without massive

[22]The ripples in Figs. 3.21 through 3.23 are an artifact of the way the independent projections of each of the three terms.

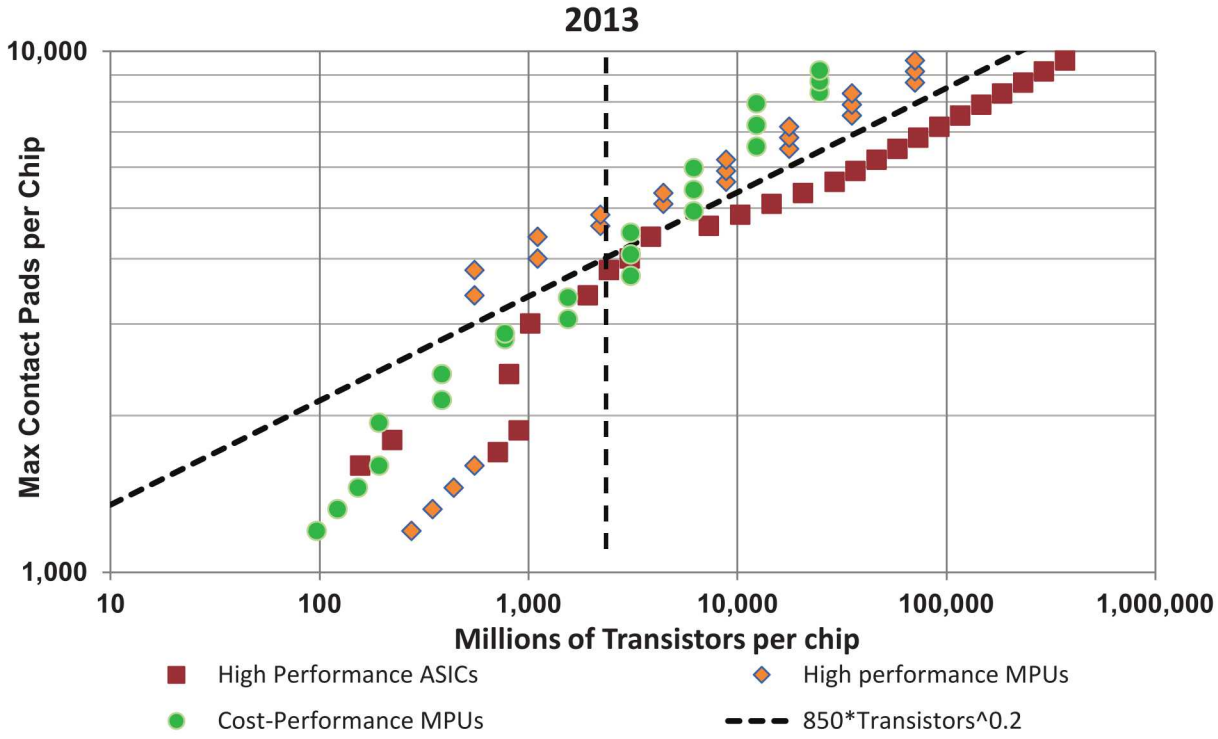


Figure 3.20. ITRS-based Chip Contacts versus Transistor Count Projections.

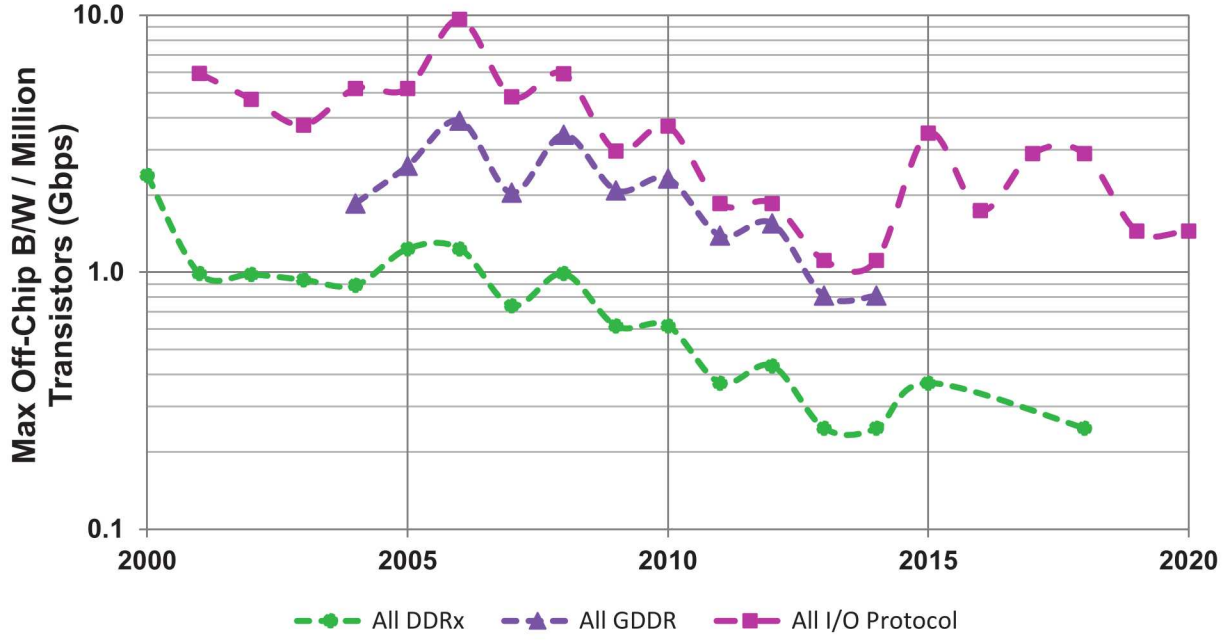


Figure 3.21. Upper Bound for Off-chip Bandwidth per Million Transistors.

improvement in transceiver power efficiency, we will be lucky to use 1/10th of the signal I/Os for high bandwidth data transfers, meaning that these curves in Fig. 3.21 should probably be reduced in the out years by a factor of 10. The one glimmer that this might be possible is a drive to component parts placed very close together, especially inside the same package, so that much shorter reach is needed to interconnect them, requiring lower power transceivers. This, however, will be expensive.

Finally, looking at the overall graph and even taking into account the power limitations, there is one striking conclusion: once again we have long since entered a period of time where the bandwidth available "per core" has declined from its peak, and will continue to decline. By the end of the decade we will have lost about an order of magnitude from the peak in the early 2000's. There is the possibility that 3D/TSV connections will ameliorate the problem somewhat (see Section 3.3.7 Rents Rule) as a result of the greater amount of logic in a block that has connections that are connected internally rather than needing external/off-chip connections. The total number of connections increases, but we can reduce the number of external connections needed.

3.3.9 Available Off-chip Bandwidth per Unit Computation

Fig. 3.21 graphs available bandwidth as a function of how many millions of transistors are on a die. This would be accurate if the clock rate for each million transistor unit were the same over all

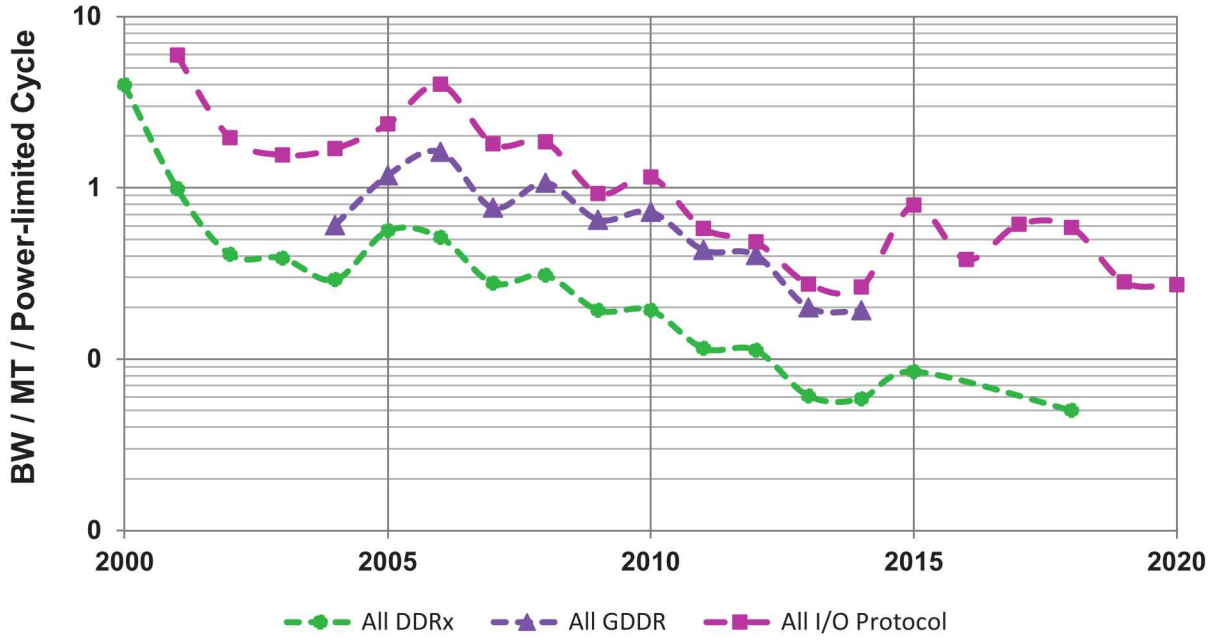


Figure 3.22. Off-chip Bandwidth per Million Transistors at Power-limited Clock Rate.

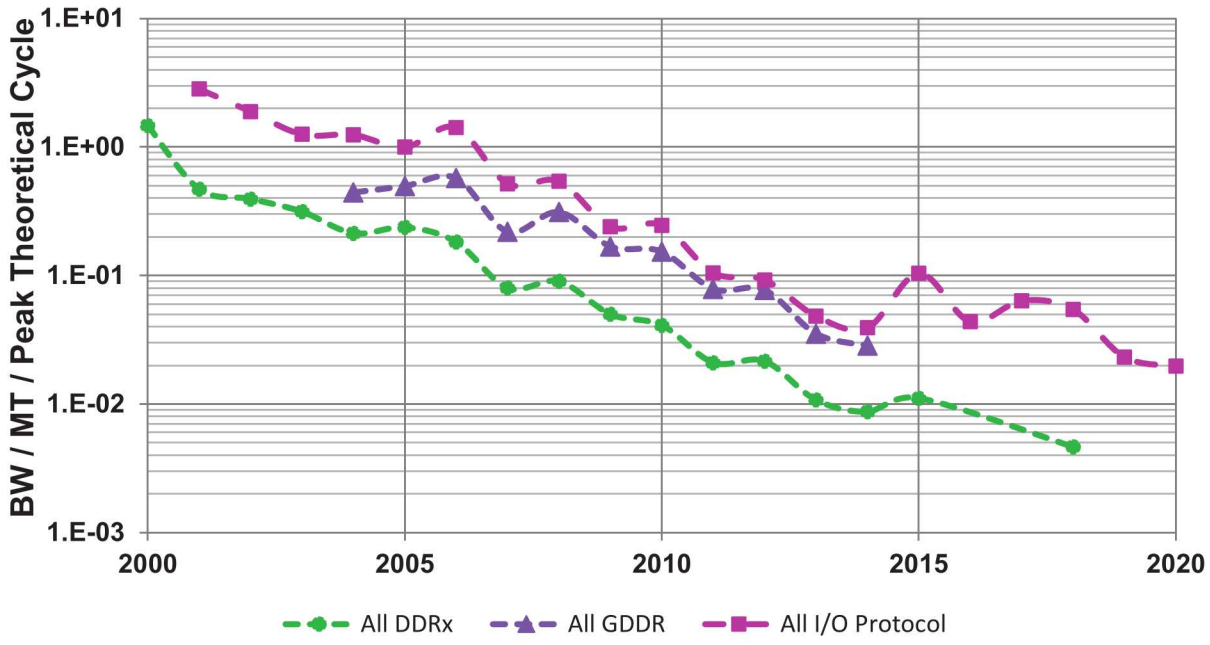


Figure 3.23. Off-chip Bandwidth per Million Transistors at Peak Theoretical Clock Rate.

time. However, clock rates have been increasing, meaning that the same 1 million transistors can do "more work" over time. Thus, it is appropriate to derate Fig. 3.21 by clock rate to get a better view of peak possible bandwidth per unit work performed inside the chip.

Fig. 3.22 does this assuming a clock rate that is the minimum shown in Fig 3.15, namely the peak clock seen in TOP500 systems up through 2012, followed by the ITRS-projected clock rate. Instead of a 10X drop, we see something approaching a 100X drop in relative bandwidth, with another 3X loss between now and the end.

Fig. 3.23 repeats this exercise using the projected 12-inverter clock, which was accurate up until around 2004. Going forward, this would be the more reasonable curve if power dissipation off the chip were no problem. Here, the derating approaches 1,000X over time, with another loss factor of 10X from today looking forward.

At least one caveat should be mentioned. Modern microprocessors are 60% or more SRAM memory, which both reduces the number of "1 million transistor blocks" that are doing computation and ups the effective internal bandwidth for heavily re-used data.

There are internal networking organizations that optimize local connectivity while supporting excellent any-to-any communication. This effectively divides a connection into multiple subsegments and thus effectively support a greater number of connections in about the same area as was done in previous internal networks. This does come at the cost of additional software overhead (better data placement requirement).

3.3.10 Memory

While the number of bits of memory per unit area on a DRAM or flash chip continues to increase as pictured in Fig. 3.12, the area of such chips is actually declining as pictured in Fig. 3.9, meaning that the number bits per chip will go through at best a slow increase over time, as pictured in Fig. 3.24. We have dropped from a roughly 4X increase in capacity per chip every 3 years to about a 2X increase every 4 years[23].

In terms of HPC systems, this means that if memory capacity is to increase with performance, more memory chips must be added, which in turn means that there is more capacitance on the wires between memory and microprocessors, which in turn means more power is dissipated in the memory subsystem.

Given the flattening of off-chip memory signalling rates, having the bandwidth performance of memory keep up with increasing on-chip microprocessor performance means that these additional memory parts need to be spread out onto more memory ports into the microprocessor sockets. However, there are no more contacts to form such ports with conventional DRAM interfaces.[24]

[23] Part of the reason is that Rent's ratio of memory size to access width/bandwidth is limiting for standard DIMMs.

[24] Differential signaling, along with getting rid of the standard interface, offers data rates high enough to reduce the total number of contacts, but requires new memory interface protocols.

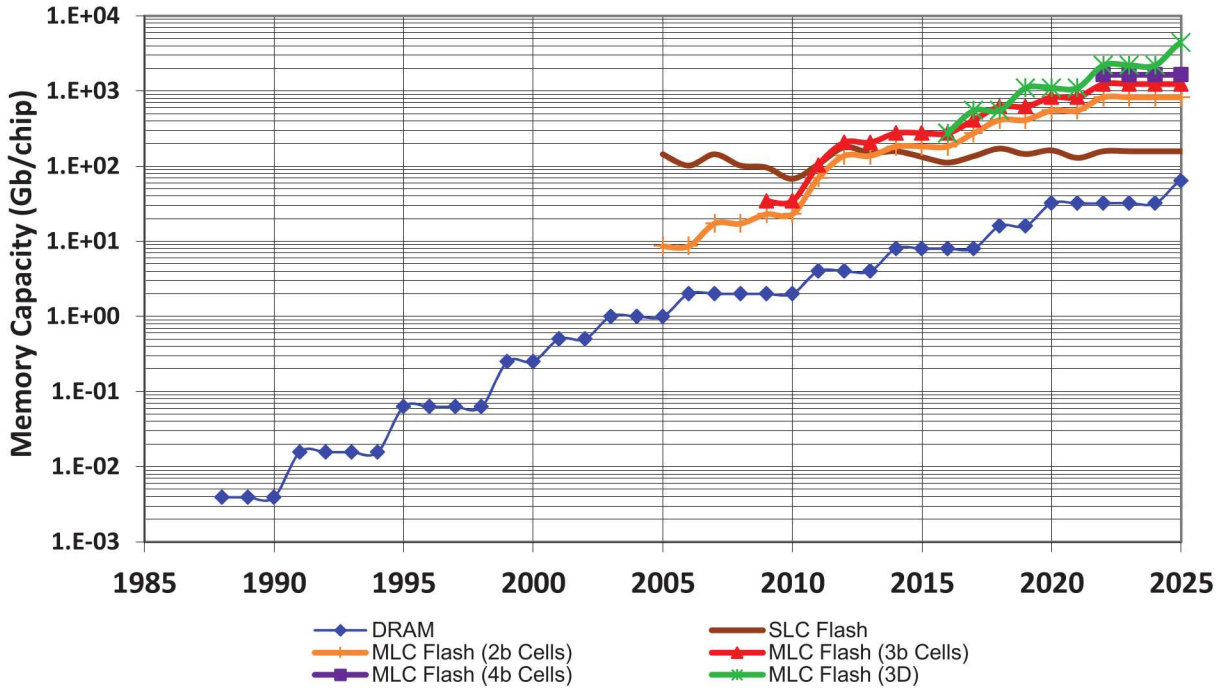


Figure 3.24. Memory Chip Capacity.

Using GDDR memory parts can help in the mid term with extra bandwidth, but their density is about 1/2 of that for commodity DRAM, meaning that even more memory parts would be needed to maintain performance. Also, as seen in Fig. 3.16, current projections are that GDDR and DDR4 eventually top off in size at about the same rate, eliminating the advantage. A different way to attack the issues of size and bandwidth is discussed in Section 3.5.

3.4 The Current Era and The Rise of Multi-core

All of the above constraints have radically affected the architecture of both commodity microprocessors and HPC systems. Microprocessor chips have gone from fast, complex, single-core designs to slower, simpler multiple cores. Many of these core designs now have a large number of FPUs (floating point units), greatly upping the aggregate number of peak flops per second possible inside the chip. At the same time the number of off-chip I/Os for memory and the I/O signalling rate have roughly flattened, making the aggregate chip bandwidth to memory relatively constant. This also means that with the number of cores per chip continuing to increase, the available off-chip bandwidth per core is actually decreasing, as pictured in Fig. 3.21.

Economic pressure on DRAM production has taken away one of the density mechanisms for DRAM, reducing the density growth. Drives to increase both the number of concurrent accesses

that can occur within a memory and the rate of off-chip signalling have added area-consuming complex control circuitry, further reducing area available for memory.

In the HPC arena, the need to link large numbers of nodes together in some sort of topology has either stolen I/Os from the memory interface, or added heavily I/O-driven router or network interface chips to the existing interfaces. Further, the need for even more bandwidth has spurred the development of fiber optic interconnect, which, at least currently doesn't integrate well with conventional silicon[25].

3.5 The Next Horizon: 3D stacks

To date, commodity technology has been primarily two dimensional: the layout of circuits on the surface of a chip, or interconnections of chips on a board. Normal off-chip contacts on a silicon chip are via pads "on the top of" a die, that is a metal pad at the uppermost layer of metal above the active side of the chip. Standard **flip-chip** packaging flips the chip upside down so these pads are on the bottom, with small solder balls placed between the pads and a pad on the substrate below it and melted to form a contact.

A new technology termed **Through Silicon Vias** (**TSV**) uses a metal-filled or doped-silicon tunnel etched all the way through the silicon substrate of a chip to allow a signal from the back side of a chip to reach the active layers and/or another surface pad. Fig. 3.25 shows a cross-section of a chip with three variants of such TSVs, depending on how high up in the layers of metal (the dark blue in the figure) the TSV interconnect goes. **Via First** TSVs go from the backside only through the substrate, and allow contacts to the first level of metal just like contacts to a transistor on the same substrate. They are fabricated before transistors or metal layers are formed. **Via Middle** go up partway into the metal layers above the substrate, and may make contact with a metal line where they meet, but do not interfere with the placement of I/O pads on the top of the chip. **Via Last** go up all the way through the chip and end at a surface pad at the top, and are formed after other processing is complete.

If the surface pads on the top of the chip line up with the backside pads of another TSV-enabled chip, a 3D stack can be formed through multiple die where communication can be not just horizontal on the surface but vertically up and down in the stack. They can be formed from via first or via middle TSVs with wire-layer interconnects to electrically connect the top and bottom bumps. In terms of ITRS notation, this is **die-to-die** (D2D) bonding of chips, with an arrangement of **back-to-face** bonding.

The expected minimum pitch between TSVs is included in the earlier Fig. 3.18. As shown, there is the potential for nearly an order of magnitude decrease in pitch over conventional contacts, meaning there may be up to 2 orders of magnitude more chip-to-chip contacts with TSVs than discussed earlier, with another factor of two if Via first or Via middle TSVs are used (top and

[25] Although there is intense work on CMOS/optic integration. Optics interconnect, starting first as a board and cabinet interconnect, can lower system cost and power.

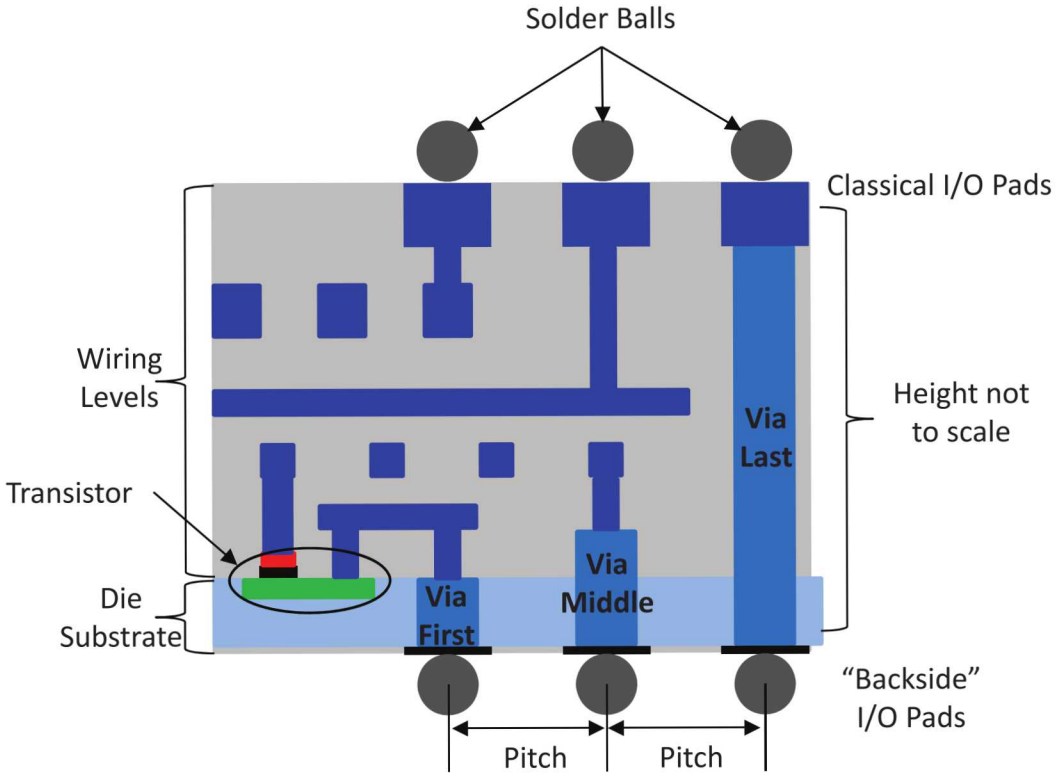


Figure 3.25. TSV Options.

bottom may then be different signals), which is useful for stacks of different die types.

While promising, there are two caveats. First, this contact density is only between chips in a stack; once an off-stack contact is needed the density reverts to the traditional numbers discussed earlier. Second, the fact that the TSV is a "through" via means that transistors cannot be placed in the region where the TSV comes up from below. Thus there is a loss of effective die area with the inclusion of TSVs, though this loss will be reduced over time with smaller via diameters along with other rule improvements.

TSV connections are much shorter (microns rather than inches) and much lower capacitance than going off-chip conventionally. Thus the complexity and energy to signal through them at high speeds is far superior to conventional I/O. In addition, as technology matures, the pads needed to allow top TSV-driven pads from one chip to mesh with bottom pads for the next chip in the stack can be placed at a far finer pitch than the pads needed for typical off-chip I/O. This means that the number of TSVs that can be placed on a unit area of silicon far exceeds the number of conventional contacts needed to go off-chip.

The great number of die-to-die connections along with the requirements and limitations of their placement, mean that, in most cases, all the die in a stacked component must be designed at the same time, in order that things like logic placement can be traded-off between the multiple die. This means that it is difficult to upgrade individual die in a stack without affecting the other stacked die. One different TSV placement in a changed die can break the whole stack.

There are ways to proceed however, for example to fix the placement of the TSVs before an upgraded or changed die design is done. The result is often less then a perfect new die implementation, but is a way forward.

3.5.1 Interposers

A technology that is intermediate between standard planar and TSV die stacks is to use a **redistribution signal layer** (**RDL**) interposer between die. The RDL is generally an insulator layer with wiring on each surface and vias through the insulation layer. A component is thus a sandwich of die (including I/O) to RDL to die to RDL to die, etc.. The insulator layer can be very thin, even using silicon in order to get very close pad spacing. The electrical result is between planar wiring and full TSV connectivity, but does not come with the manufacturing and other issues that accompany TSV packaging. There is also an easier time of upgrading a stack with a new or modified part.

RDL stacking is in current production, and its usage is likely to increase over the next few years as manufacturers are getting comfortable with TSV technology. RDL does raise cost, but also has benefits of lower power and energy and very dense packaging.

3.5.2 Hybrid Stacks

The concept of **hybrid stacks**, where dissimilar die types are stacked together, has been reduced to practice with a demonstration of a logic chip below stacks of modified memory die. The logic chip contains both memory controllers and high speed routers and signalling to go off chip to other stacks or even conventional microprocessors.

Hybrid Memory Cube

An example of this is a prototype memory stack termed the **Hybrid Memory Cube** (**HMC**) demonstrated by Micron[29] in 2011, with projected use not just with DRAM but a wide variety of alternative memory technologies in 2013[30]. A consortium of several companies, called the **Hybrid Memory Cube Consortium**[26], have recently released a specification for a first generation product[27] using this technology[6].

As pictured in Fig. 3.26, the architecture of such stacks includes a vertical "slice" of all the DRAM chips that form a **vault** that together share a TSV-implemented interface down to the **logic base chip**. On the logic chip under each of multiple such vaults is a **vault controller**, which includes at a minimum a memory controller and switching and link logic to connect each vault to interfaces that leave the bottom side of the logic chip. Fig. 3.27 is a die photo of a memory die from the 2011 HMC stack[29]; the locations of the TSVs are clearly visible as small circles.

Fig. 3.28 diagrams the architecture of the HMC stack as described in [6]. The router, link interfaces, BIST (Built-In Self-Test), and a maintenance capability all share space with the vault controllers on the base die.

Table 3.1 summarizes some reported and projected characteristics.

Dis-Integrated RAM

An interesting alternative hybrid stack has been introduced by Tezzaron[28] where there are multiple "logic layers" at the bottom of the stack. In the HMC approach the sense amps associated with the DRAM banks are on the DRAM die, with logic level signalling down the TSVs to the logic die.

In the DiRAM architecture there are two logic layers, with the one just below the memory stack containing all the sense amps and memory interface logic for the memory above. The layer at the bottom contains all the I/O and routing for off-chip communication.

[26] see http://hybridmemorycube.org/

[27] http://www.micron.com/products/hybrid-memory-cube/short-reach-hmc

[28] http://www.tezzaron.com/products/diram4-3d-memory/

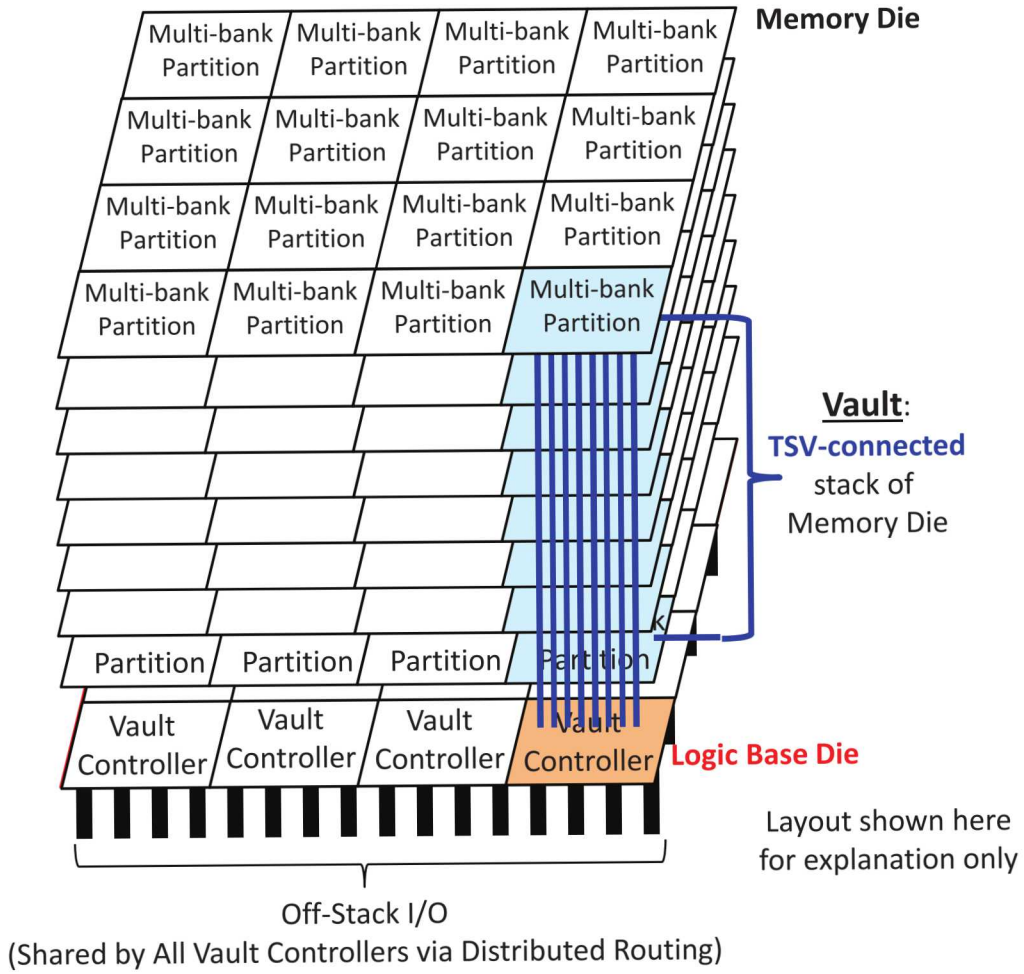


Figure 3.26. Notional Hybrid Stack.

Characteristic	2011 Prototype ([29])	2014 HMC Gen2 [6]	Projected Gen 3 ([30])
DRAM Technology	50nm		Est. 20nm
Memory die/stack	4	4-8	4-8
Memory/die	1Gb	4Gb	Est. 8Gb
Memory/stack	512MB	2-4GB	4-8GB
Logic Technology	90nm		Est. 30nm
Logic die/stack	1	1	1
Number of TSVs	Est. 2620	T.B.D	Est. 5000
Number of vaults	16-32	16	Est. 32
Memory/die/vault	8MB	16-32MB	64MB
Memory/vault	32MB	128-256MB	512MB
Memory banks/partition	2	2	2
Memory banks/vault	8-16	8-16	16-32
Down vault Bandwidth	Est. 10GB/s	Est. same	Est. same
Total Memory to Logic Bandwidth	160GB/s	Est. same	Est. 320GB/s
Off-chip links	4	2,4	8
Differential Lanes per link	16	16	16
Signal Rate/lane direction	10Gb/s	10-15Gbps	10-15Gbps
Total off-chip bandwidth	160GB/s	240GB/s	320-480GB/s
4-High Stack Power	11.02W		Est. 15W

Table 3.1. Reported and Projected Hybrid Memory Stack Characteristics.

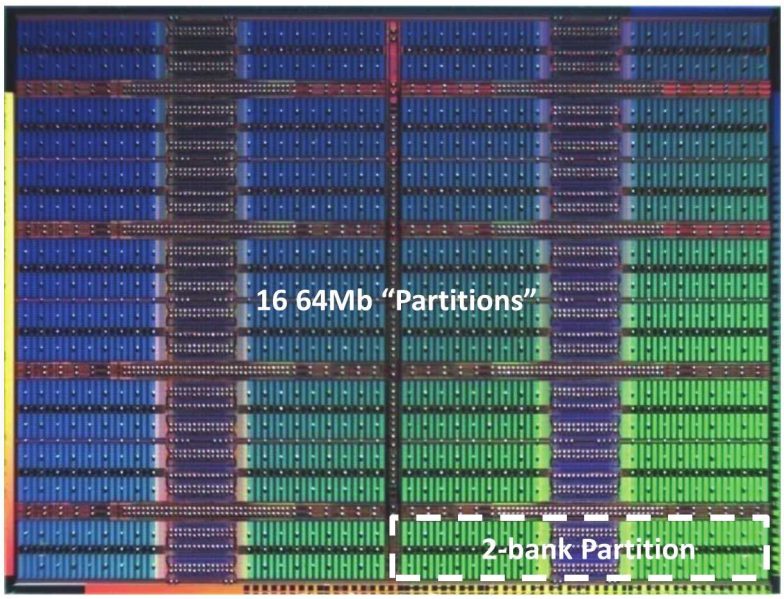


Figure 3.27. Memory Die from Micron HMC Prototype[29].

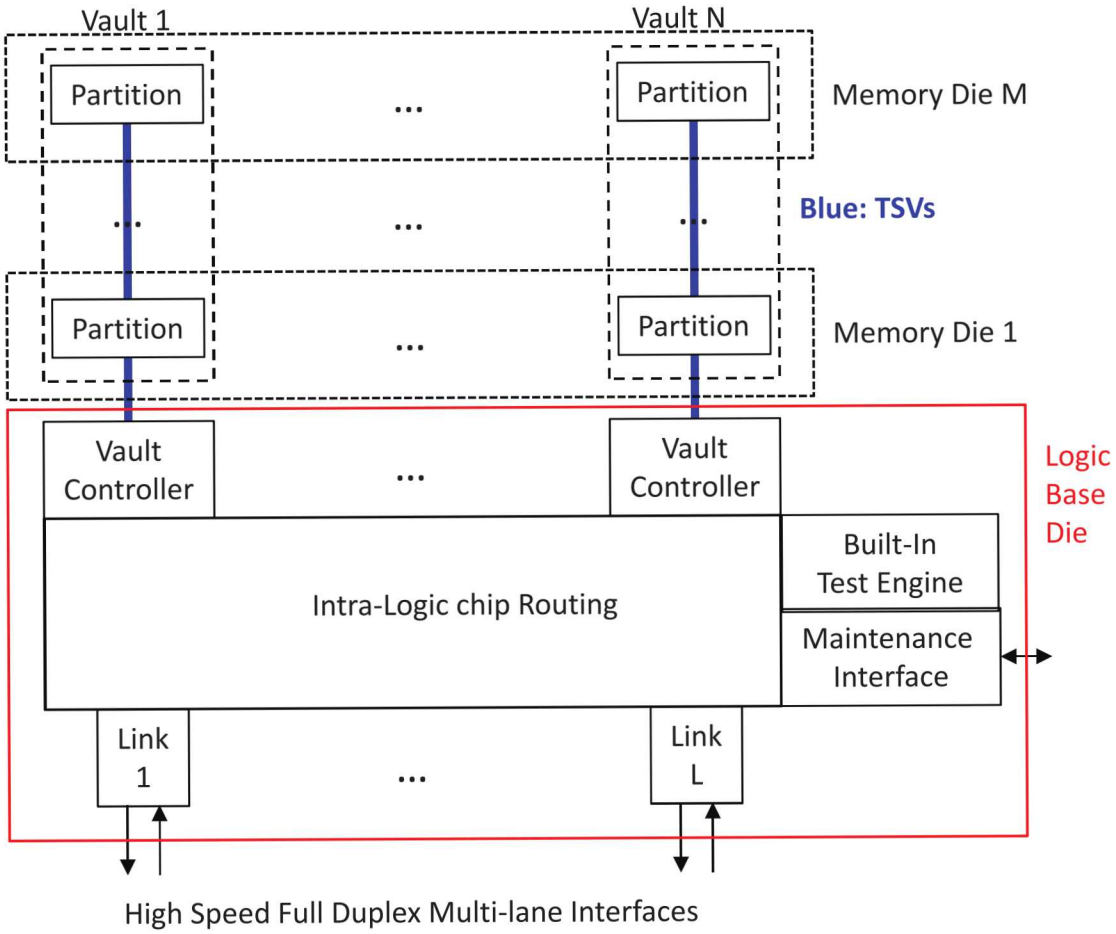


Figure 3.28. High-Level Architecture of HMC stack as defined in [6].

Chapter 4

HPC Architecture Classes

The high level architecture of virtually all HPC systems for the last 20 years has been a collection of independent **nodes**, each of which contains one or more computational processor chips, some amount of local DRAM memory, and some interface logic to permit communications between nodes or dedicated I/O units. To reflect the relatively high fraction of the node cost taken up by these compute chips, and to avoid confusion between microprocessors, cores, etc., we refer here to each compute chip with a significant number of logic transistors (and likely needing a heatsink or other cooling apparatus) as a **socket**[1].

Depending on the design, there may be separate chips for subsidiary functions such as timing generators, memory controllers, boot management, performance monitoring, and the like.

Prior to 2004 the predominate computational socket held a single core microprocessor chip derived from a commercial offering, running at the maximum clock rate of the day, and executing only one or two threads each. While the complexity of the core increased with time, the number of FPUs per core stayed relatively small. There were a few exceptions, the most notably being the Earth Simulator whose sockets held custom-designed SIMD vector units.

After 2004, there has been a tri-furcation of architectures based both on the architecture of the chips in the sockets and on the nature of the nodes themselves. Using nomenclature first introduced in the Exascale technology report[20], these include **Heavyweight**, **Lightweight**, and over the last five years **Hybrid**. In the very near future we may see the rise of BigLittle architectures that blend aspects of all of the prior three. In addition, there may be an emerging class based on moving from today's "2 dimensional" systems (in terms of how chips are designed, packaged, and interconnected on a node printed circuit card) to "3 dimensional" systems using 3D stacks of chips as discussed in Section 3.5. Each is discussed below.

For reference, Table 4.1 summarizes many of the characteristics of the different classes for representative current systems, and Fig. 4.1 pictures typical boards.

Typical Processor Socket Chip Characteristics			
Reference	POWER7	Blue Gene/Q	Nvidia GK110
Transistors (Billion)		1.47	7.1
Feature Size (nm)		45	28
Cores per Chip	8	16+2	14 SMX
FPUs per Core		4 FMA	64 FMA
Core Clock Rate (GHz)	3.836	1.6	0.732
R*peak* per chip (GF/s)	245.5	204	1312
Memory Ports per chip		2	6
Memory Type on Port		DDR3 DRAM chips	GDDR DRAM chips
Memory B/W per chip (GB/s)	128	42.7	288
Socket to Socket Interfaces		none	
Typical Node Characteristics			
Reference	POWER 775	Blue Gene/Q	Cray XK7
Heavyweight Sockets	4	0	4
Lightweight Sockets	0	1	0
Accelerator Sockets	0	0	4
R_{peak} per node (GF/s)	982	204	5,800
Main Memory per Node (GB)	128	16	32
Accelerator Memory per Node (GB)	0	0	6
I/O B/W per node (GB/s)	192	44	
Power per Node (W)		98	1,760
Nodes per Rack	96	1024	96
Cooling	Water	Air	
Typical System Characteristics			
Architecture	Heavyweight	Lightweight	Hybrid
Example System	DARPA Trial	Sequoia	Titan
Compute Nodes	1,980	98,304	4,672
Total Nodes	2,048		
Racks	24	96	200
Compute Main Memory (TB)	253	1,572	710
Accelerator Memory (TB)	0	0	112
R_{peak} (PF/s)	1.94	20.1	27.1

Table 4.1. Architecture Characteristics.

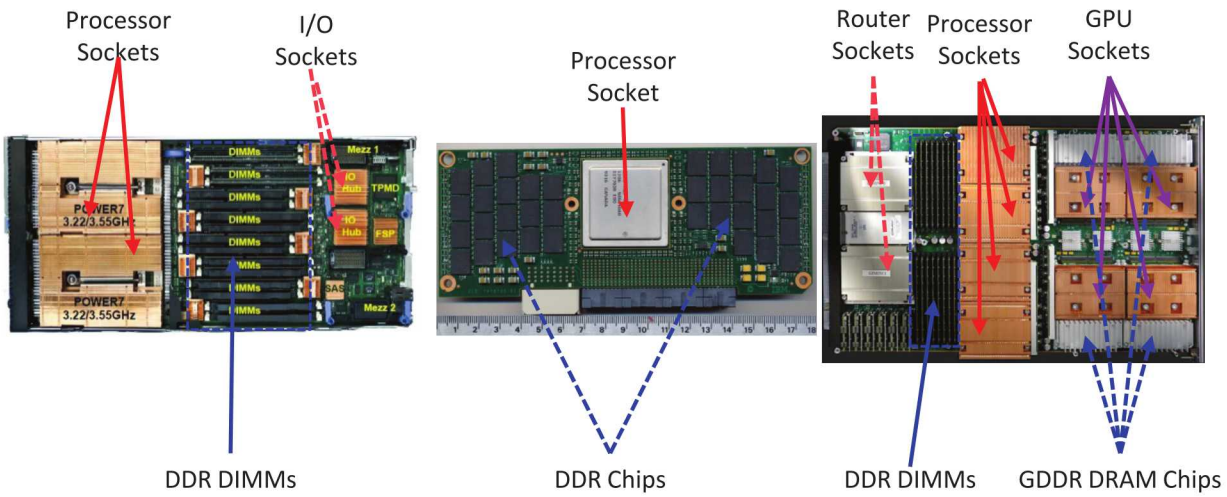


Figure 4.1. Typical Boards for different Architecture Classes.

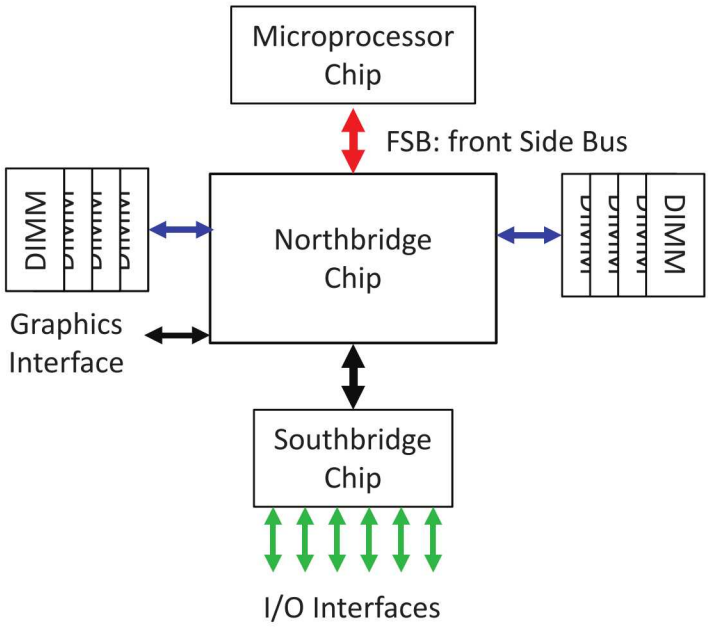


Figure 4.2. A Traditional 3-chip Chipset.

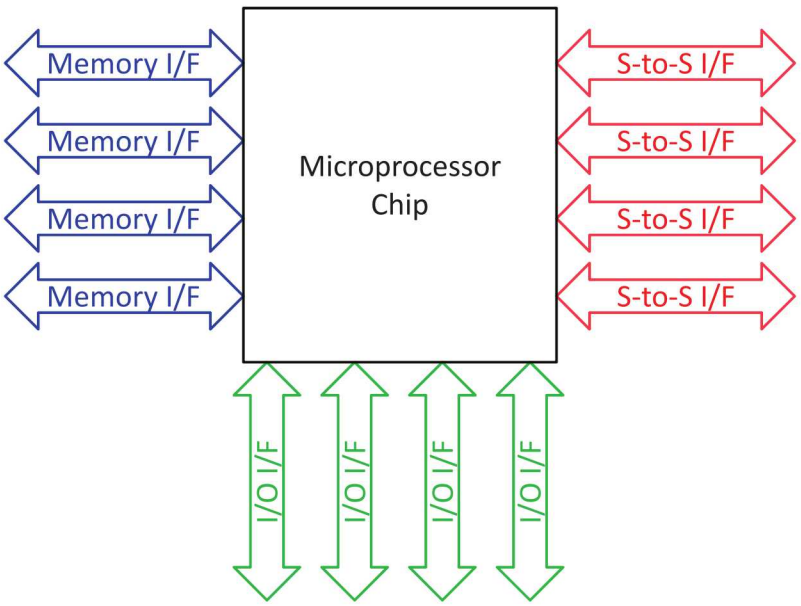


Figure 4.3. Today's Microprocessor Socket.

4.1 Traditional Chip Set Architectures

A typical computer up through the mid-2000s had upwards of three chips per node as pictured in Fig. 4.2. There was the microprocessor chip/socket, which in turn connected to a **northbridge** chip/socket over a high speed and as wide as possible **front side bus** (**FSB**). This northbridge chip then contained one or more **memory controllers** that in turn connected to standard memory parts of some sort. For use in personal computers, it often also held an either a full graphics engine and/or a high speed interface to such a controller on yet another chip. In addition, it often also connected to a **southbridge** chip/socket, which in turn had multiple interfaces (often specialized) to different classes of I/O devices. This arrangement allowed a microprocessor chip to be designed relatively independently of the type of memory and mix of I/O and graphics. It also allowed independent high speed access to memory from either the graphics interface or the I/O devices.

In a typical HPC environment, the northbridge chip needed to support only memory and perhaps some secondary I/O, with the southbridge replaced by a heavy duty router chip that allowed a compute node to be placed into a bigger topology. A common topology was a 3D toroidal mesh with six separate bidirectional links, one each for east/west, north/south, and up/down.

In contrast, Fig. 4.3 diagrams a typical modern multi-core microprocessor socket. The growth in transistors has allowed integration of much of the north and south bridge chips onto the microprocessor. In particular, there is often two to four standard memory controllers and their interfaces

[1]There are modules that contain multiple die but still package like single-chip parts. The term "socket" will refer to them also.

(typically DDR3 interfaces), allowing direct attach of memory DIMMs.

In addition there are some number of socket to socket interfaces (labeled "S-to-S") in the figure) that utilize some small number of the non-memory higher-speed protocols discussed in Section 3.3.4, such as Hypertransport™or QPI™. These interfaces provide links to other processor sockets so that a memory reference made in any core in any such connected socket can travel to the correct socket that controls the identified memory location, and fetch the data. There is often a cache-coherent protocol on top of these links, so that all memory attached to all such interconnected sockets behaves as a single shared memory system, albeit **Non-Uniform Memory Access** (**NUMA**) in access time.

The third kind of off-socket link is to provide access to I/O devices. Again some form of high-speed multi-lane protocol is typical, such as Hypertransport™again, or PCI-Express™. Memory access routing logic internal to the processor socket again allows streaming of data to or from these interfaces directly from memory, without going through a core.

Again, because of off-chip contact limitations, the number of all such interfaces is limited, and is unlikely to improve significantly with conventional technology. There will be some improvement in data rates (perhaps up to 4X for a differential pair over a single wire) but after that higher rates will need optics.

4.2 Heavyweight Architectures

The class of systems, termed **Heavyweight**, are the natural descendants of the pre-2004 systems where one or more traditional, high end, multi-core microprocessor chips have been used in the compute sockets, often with a variety of support chips connecting to relatively large numbers of conventional DRAM memory DIMMs, and chips to perform off-node communication and routing, as pictured in Fig. 4.4. Typically the compute and routing chips are run at very high clock rates and require large (and "heavy") heatsinks over them to keep them cool. Typical of such machines is a Power7+ board (Fig. 4.1(a))[2] where much of the board is taken up by the heatsinks for the processor sockets and the I/O sockets (essentially southbridge chips).

The chips in modern heavyweight processor sockets have eight to sixteen cores, each with two to four FPUs, and running at or above 3GHz. Typically up to 4 independent memory ports are supported, with each port supporting two to four high capacity, multi-rank, high bandwidth DIMMs.

All the cores in a single heavyweight processor socket share access to all the memory attached to that socket, usually including cache coherence. In many heavyweight designs this sharing of memory extends to some, usually small, number of other compute sockets in the same node. These compute sockets then typically share a node network interface that uses a non-memory oriented protocol. Thus a single node has a moderately large number of cores that share local node mem-

[2]image from http://www.theregister.co.uk/2012/11/13/ibm_power7_plus_flex_storwize/.

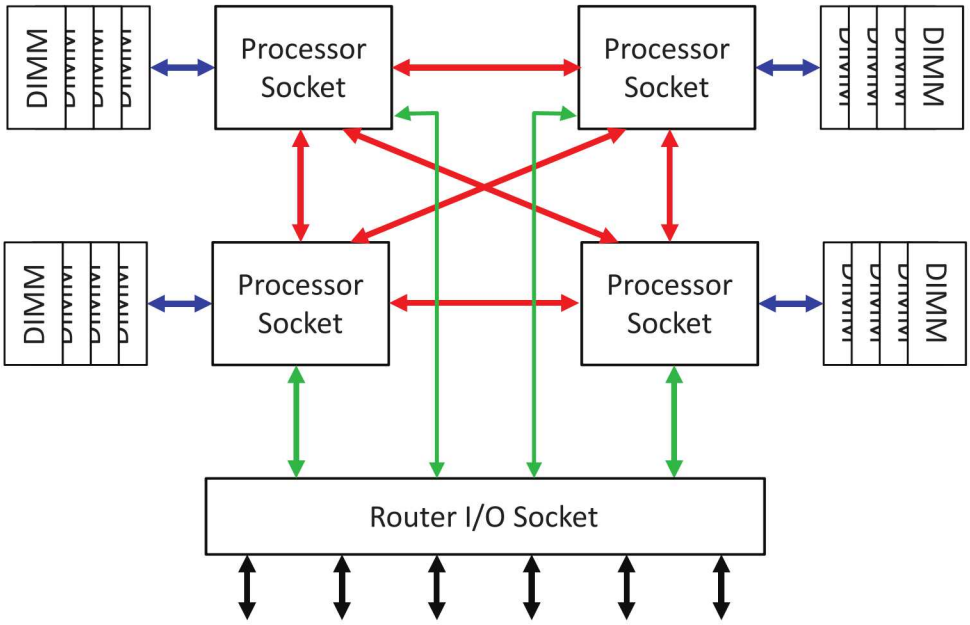


Figure 4.4. Typical Heavyweight Node Architecture.

ory with each other, but has a more distributed memory interface when dealing with any core or memory elsewhere in the system, requiring software such as MPI for communication.

Typically at most a 100 or so such nodes can fit in a single rack.

4.3 Lightweight Architectures

The introduction of the IBM Blue Gene/L[34] in 2004 used a compute socket with a dual core processor chip that included memory controller and I/O and routing functions on a single chip. The cores were much simpler than for the heavyweight machines, and ran at a much lower clock

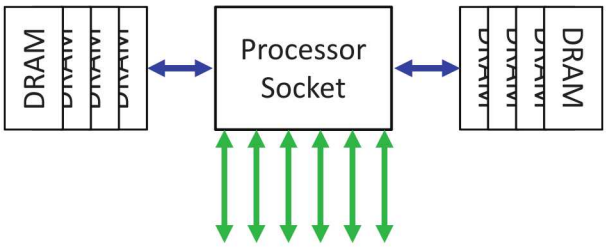


Figure 4.5. Typical Lightweight Node Architecture.

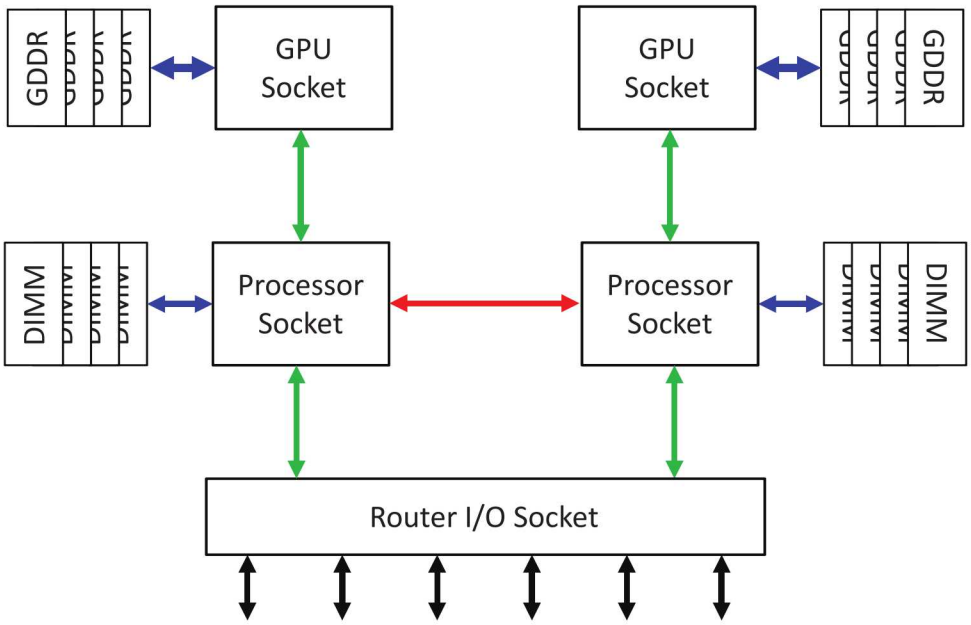


Figure 4.6. Typical Hybrid Node Architecture.

rate. Such a chip, when combined with some memory, made a complete node in the above sense, as pictured in Fig. 4.5. Two such nodes were then packaged on small cards which plugged into motherboards that inter-connected them. Because the required heatsink was so much smaller than for a heavyweight, many of these small cards could be packaged in the same space as a heavyweight node (up to 1024 of such nodes in a single rack). Subsequent versions, Blue Gene/P[3] and now Blue Gene/Q[11], as pictured in Fig. 4.1(b)[3], have continued this class of architecture. Blue Gene/Q in particular has a 16+1 multi-core processor chip, with two memory controllers connected directly to conventional DDR3 DRAM chips on the same node board.

Again all cores in a node's processor socket view the node as a single shared memory structure, but as with the heavyweight nodes, other nodes are viewed in a distributed fashion, requiring software such as MPI for communication.

4.4 Hybrid Accelerator-based Architectures

The original Exascale report[20] identified only the heavy and lightweight classes. Since then, however, a third class has surfaced, which combines with a heavyweight socket a second compute socket where the chip in it boasts a large number of simpler cores, usually with an even larger number of FPUs per core. Today such chips are derived from **Graphics Processing Unit GPUs**, such

[3]image from http://www.cpushack.com/wp-content/uploads/2013/02/IBM51Y7638_BlueGeneQ.jpg

as the Nvidia Tesla architecture[4], the Intel Xeon Phi architecture[5], or the AMD GCN architecture[6].

Fig. 4.1(c) pictures one such node from the Titan supercomputer[7].

As with the heavyweight nodes, something of on the order of a 100 such nodes could be packed into a single rack.

Memory within such nodes is today usually not as fully shared as in the other classes. Instead, the accelerator node typically can only access its own local GDDR memory during computation. Thus the heavyweight host processor must explicitly transfer between the accelerator's memory and its own memory. This staging takes both time and program complexity, and derates the usefulness of the accelerator when random accesses to larger amounts of data than can fit in the GDDR is needed.

4.5 BigLittle Architectures

Of the prior classes of architectures, both the heavyweight and lightweight had two points in common: all cores executed exactly the same ISA and had exactly the same microarchitecture. In contrast, a hybrid accelerator-class had at least two different core types, different in both ISA and microarchitectures.

An alternative architecture that is beginning to surface in discussions of high end servers is termed **BigLittle**. In such an architecture all cores have the same ISA, but the cores may have different microarchitectures, most probably even on the same chip and share the same memory. This difference reflects different optimization points. "Big" cores are optimized for highest performance; "little" cores are optimized for performance per watt. Threads can be initiated on either core type and get the same answer, but in different times and different energy expenditures. Sharing the same memory permits such a switch in execution to occur without moving data.

This class of architectures was first proposed by ARM in 2011, with core pairs Cortex-A15 (Big) and Cortex-A7 (Little)[10]. More recently, ARM has announced Cortex-A53 and Cortex-A57 cores to fulfill similar roles[8]. Recently announced sockets using this technique seem to be clustered around two different configurations as pictured in Fig. 4.7: 1-to-1 and 2-to-1 in favor of the little cores. Having a 1-to-1 ratio allows a thread to run on either the big or little as conditions require, with the "unused" core turned off for energy savings. Having more little cores than big cores allows either high-speed sequential performance from the big core or high-speed, energy-efficient, parallel performance from the multiple little cores.

[4] see http://www.nvidia.com/content/tesla/pdf/Tesla-KSeries-Overview-LR.pdf

[5] see http://www.intel.com/content/www/us/en/processors/xeon/xeon-phi-detail.html

[6] see http://www.amd.com/us/Documents/GCN_Architecture_whitepaper.pdf

[7] image from http://techreport.com/r.x/2012_10_29_Nvidia_Kepler_powers_Oak_Ridges_supercomputing_Titan/titan-blade.jpg

[8] http://www.arm.com/products/processors/technologies/biglittleprocessing.php

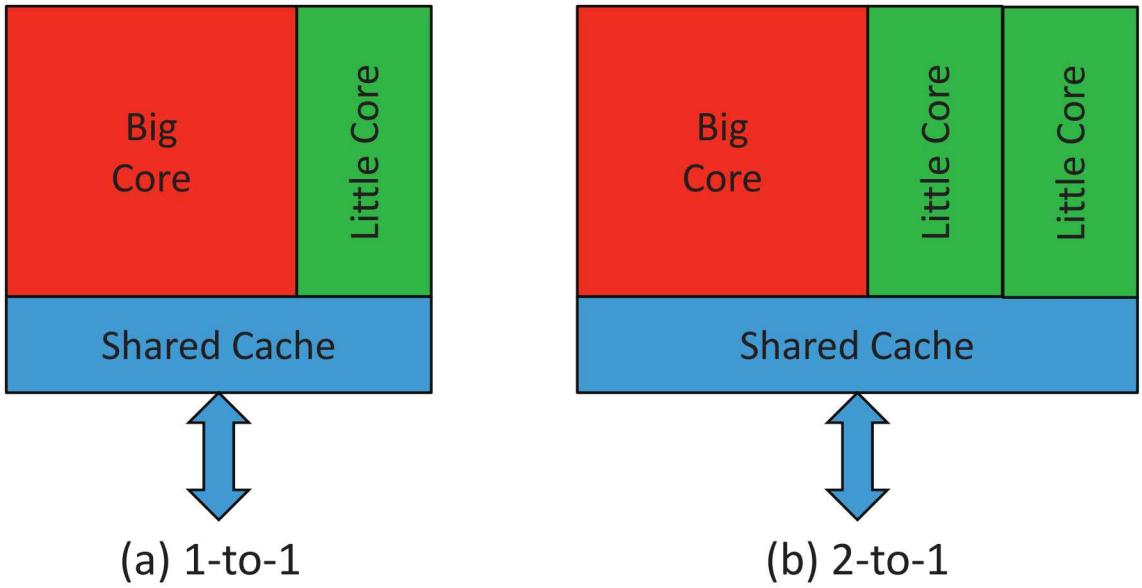


Figure 4.7. BigLittle Architectures.

In addition, a very desirable attribute of such an architecture is the ability to dynamically move a thread state from one core of one type to another *during* program execution as conditions dictate (versus just at thread spawn time). In the case of ARM, there is a cache coherent link protocol designed to connect such cores and allow such migrations[9] without having to manage cache contents of the two cores explicitly.

Arm claims that less than 2,000 instructions are needed to migrate a thread state from one core to another. This is in comparisons from estimates from the Intel Cilk Plus website that for a thread on one core to steal work from another core is about 15,000 instructions.

Some benchmark data[10] on the relative performance and energy of A7 and A15 cores are pictured in Fig. 4.8, along with some of the micro-architectural features of the cores and two trend lines. All points are at of above the equal energy-performance line (in green) indicating that increasing performance by X takes more than X increase in energy. The red line reflects the average of these energy to performance ratios of about 1.5 to 1, that is doubling performance on average triples the energy to do the computation.

Intel may also be on the road to something similar with the Intel Phi chip which contains up to 61 lightweight Xeon-compatible cores. The June 2013 top supercomputer Tianhe-2 used these chips as coprocessors to a conventional heavyweight Xeon socket. In this configuration, however, the two core types do not share the same memory; the little cores on the Phi have a memory space separate from the host Xeon, and, as in earlier hybrid accelerator architectures, explicit data transfers must be performed between the two to allow computation. There are also network

[9] http://www.arm.com/products/system-ip/interconnect/corelink-cci-400.php
[10] http://www.arm.com/files/downloads/big.LITTLE_Final.pdf

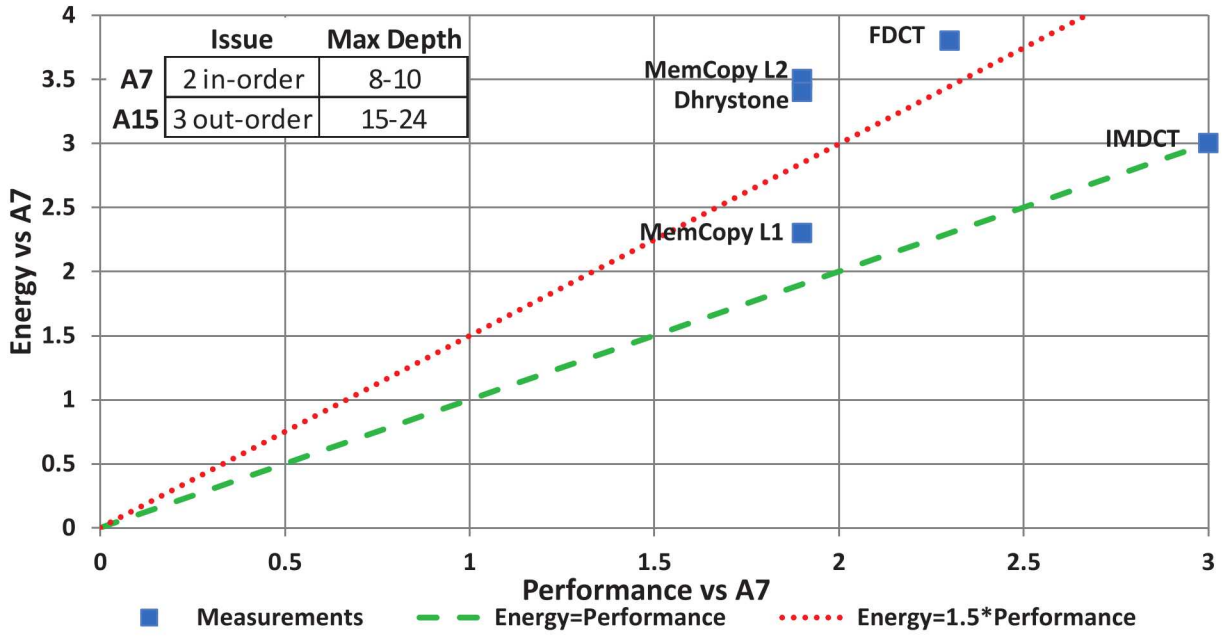


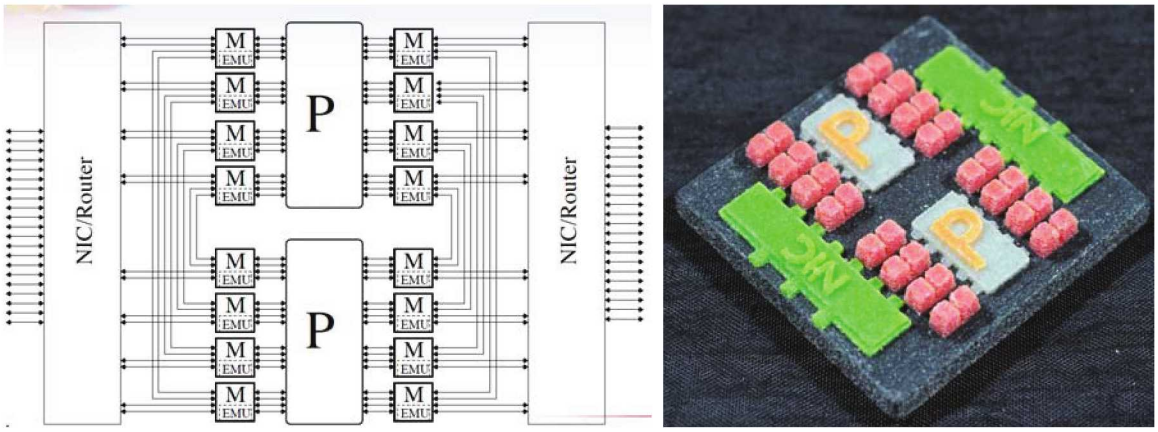
Figure 4.8. Example Energy-Performance of Two Different Cores.

differences between cores within the accelerator: Phi uses a very wide dual ring, whereas most GPUs use something akin to a cross-bar.

4.6 Emerging Stack Architectures

3D stacks have become a major study topic in computer architecture in the last few years. Black et al[2] studied two forms of die stacking: placing cache chips on top of a conventional microprocessor, and splitting such a microprocessor into two die. Ghosh and Lee[9] and Kgil et al[16] studied microarchitectures that use 3D stacking to reduce energy costs. Loh[23] discussed re-architecting DRAM die that would stack above a conventional processor core in ways that increase the bandwidth and memory level parallelism of the memory as seen by the processor. Udipi et al[35] studied 3D stacks of conventional memory die with a photonic interface base chip. Sun et al[33] discussed stacking MRAM chips on top of a multi-core processor die. Madan and Balasubramonian[24] proposed a two die processor stack where the second die performs redundant checking of the first.

With the introduction of the HMC 3D memory stack section 3.5.2, real stack-based products are now possible. Early on, however, the architecture of such systems will not be full 3D systems, but what is beginning to be called "2.5D." Such systems still rely on a large heavyweight multi-core microprocessor chip, but one that is surrounded by multiple 3D stacks. By placing both the multi-core and the memory stacks on the same package substrate, and keeping distances very short,



(a) X-caliber Node Architecture (b) X-caliber Node Mockup

P: multi-core processor socket
M: memory stack
NIC: off-node router

Figure 4.9. X-caliber Memory-Stack Centric Node.

the interconnect power issues discussed in Section 3.3.5 can be controlled and kept manageable. The following subsections discuss some of these intermediate architectures.

4.6.1 An Early 2 1/2 Architecture Study

The X-caliber project[28] at Sandia Lational Labs was an attempt to advance a 2.5D style of architecture beyond a single node, into an architecture that would scale beyond current systems. Fig. 4.9 outlines the structure of a proposed X-caliber node, and a graphic[11] of a mockup of its possible future implementation on a single substrate. Each such node would have two heavyweight processor sockets surrounded by a large number of memory stacks. The interface between the processors and the sockets would be a high-speed interface akin to the Hypertransport and QPI interfaces discussed earlier, and running at speeds in the range suggested by the HMC work from Section 3.5. Similar interfaces from the memory stacks would go to two high power router chips, which in turn would provide very high radix routing into a much more fully connected topology than a 3D torus.

For reference, Fig. 4.10 (also taken from [28]) diagrams the architecture of a single memory stack. The DRAM chips above the base logic are organized as in the HMC, with what was a "slice" (vertical partition) in the HMC labeled as a "vault" here. For each of the assumed 64 vaults per stack there is on the logic base chip a memory controller ("MC" in the figure) and a **vault atomic unit** (VAU). This VAU is a small processor core of limited capability, but capable of performing

[11] image from https://share.sandia.gov/news/resources/news_releases/images/2010/Computer.jpg.

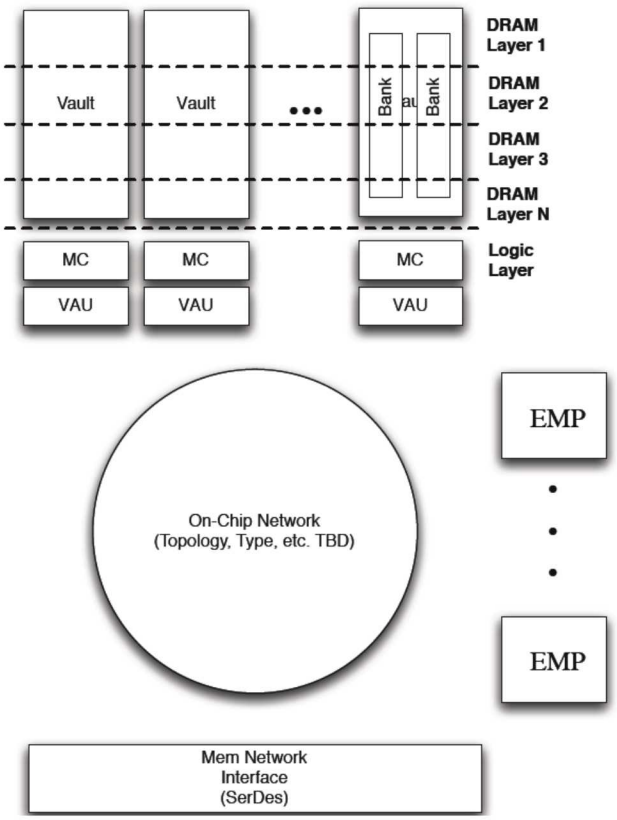


Figure 4.10. X-caliber Memory-Stack Logic Chip Architecture.

Deployment	Nodes	Topology	Compute	Mem BW	Injection BW	Bisection BW
Module	1	N/A	8 TF/s	3 TB/s	1 TB/s	N/A
Deployable Cage	22	All-to-All	176 TF/s	67.5 TB/s	22.5 TB/s	31 TB/s
Rack	128	Flat. Butterfly	1 PF/s	.4 PB/s	0.13 PB/s	0.066 PB/s
Group Cluster	512	Flat. Butterfly	4.1 PF/s	1.6 PB/s	0.52 PB/s	0.26 PB/s
National Resource	128k	Hier. All-to-All	1 EF/s	0.4 EB/s	0.13 EB/s	16.8 PB/s
Max Configuration	2048k	Hier. All-to-All	16 EF/s	6.4 EB/s	2.1 EB/s	0.26 EB/s

Figure 4.11. Performance of a Rack of 128 X-caliber nodes.

close-in "atomic" operations to memory in the vault slice above it.

Also on the logic chip are some number of more conventional cores, labeled "EMPs" here. These cores can execute conventional programs, and make memory references to any of the memory either here or on other memory stacks anywhere in the system. The key to accomplishing this is an on-chip router that couples all the EMPs, vaults, and off-chip links together, and is capable of routing memory or processing requests to any vault/VAU from any EMP or cores in the main processor sockets.

Also, each stack is assumed to have not just the DRAM stack above the base logic chip, but a second non-volatile memory stack. This non-volatile memory could be used as either a local large scratch memory, or as local disk replacements.

Sizings place up to 128 such nodes (256 processor sockets and 2048 memory stacks) for characteristics as shown in Fig. 4.11 (again taken from [28]).

4.6.2 Intel's Knights Landing™

In June of 2014 Intel announced[12] Knights Landing™, a product due out in 2015 with an architecture that moves solidly in the direction proposed in the X-caliber study. The product is a single package that includes:

- A Xeon Phi chip fabricated in a 14 nm technology, with nominally 72 cores and capable of up to 3+Tflops/sec. Each core supports up to 4 concurrent threads and a 512 bit wide SIMD capability.
- Multiple memory stacks called **MCDRAM**, each of which is akin to the HMC part but with just a single interface from each to the Phi chip. They provide an aggregate capacity of 16GB and aggregate bandwidth of 500GB/s that is 5 times the bandwidth of DDR4.
- Multiple DDR4 memory ports that can give the package more memory capacity than can be provided by the MCDRAM stacks.
- a router chip that provides an interface to a set of links that make up what Intel is terming an Omni Scale fabric.

The key differences between this architecture and the X-caliber one are that there are two distinct types of memory: MCDRAM and conventional DDR4, and there is no integrated non-volatile memory.

Also announced is an early system called "Cori[13]" that will have 9300 of these packages and will achieve upwards of 50 petaflops/sec.

[12] http://insidehpc.com/2014/06/video-intel-unveils-knights-landing-details-isc14/

[13] https://www.nersc.gov/users/computational-systems/nersc-8-system-cori/

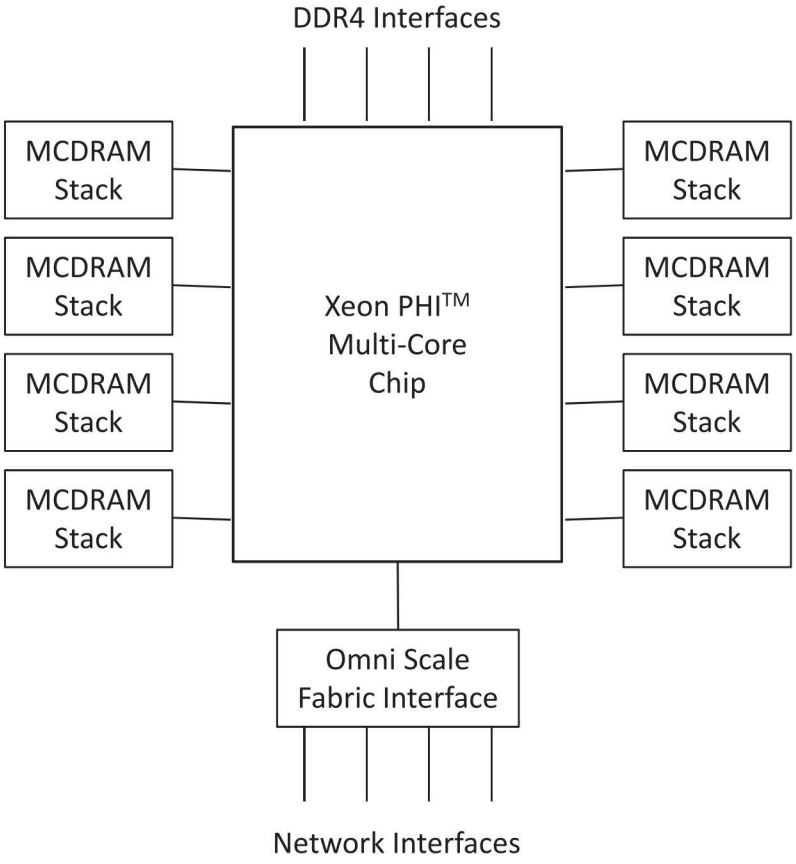
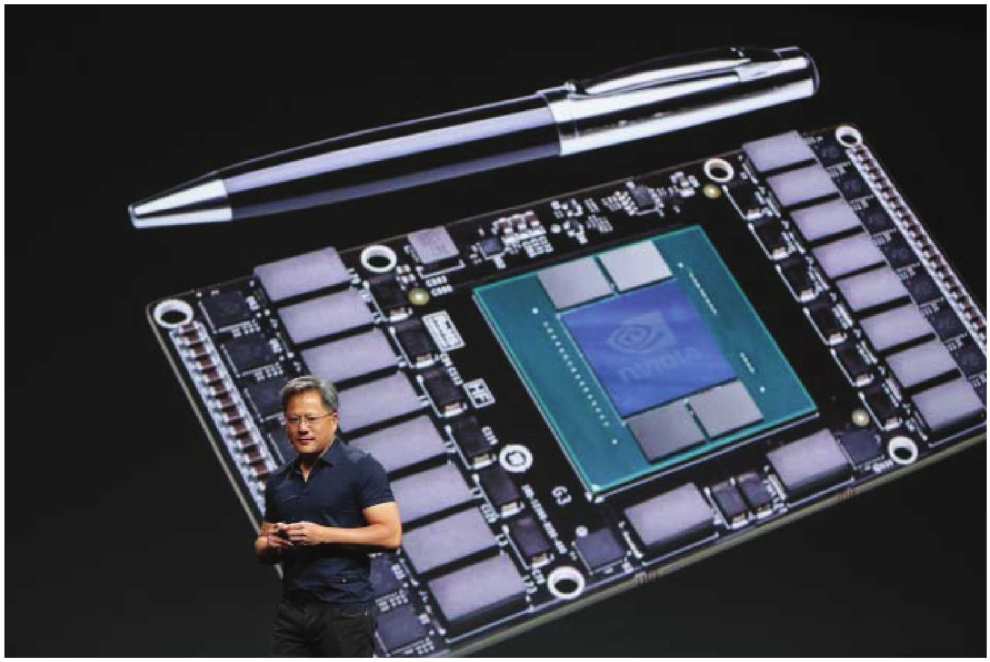


Figure 4.12. The Knights Landing Architecture.



http://blogs.nvidia.com/wp-content/uploads/2014/03/pascalmodule.png

Figure 4.13. Nvidia's Pascal Node.

4.6.3 Other 2.5D Systems

NVIDIA also recently announced[14] a GPU-based node that includes 3D memory stacks on the card, Fig. 4.13. There is no local memory other than the stacks, and there is a high speed link to a conventional microprocessor host. No details on capacity or speeds are available.

Fujitsu has also announced[15] a "Post-FX10" PTOMEHPC board, Fig 4.14, that contains 3 heavyweight microprocessor chips, each with eight HMCs as main memory. Again, however, the stacks are on a conventional printed circuit board, and not integrated into a single package.

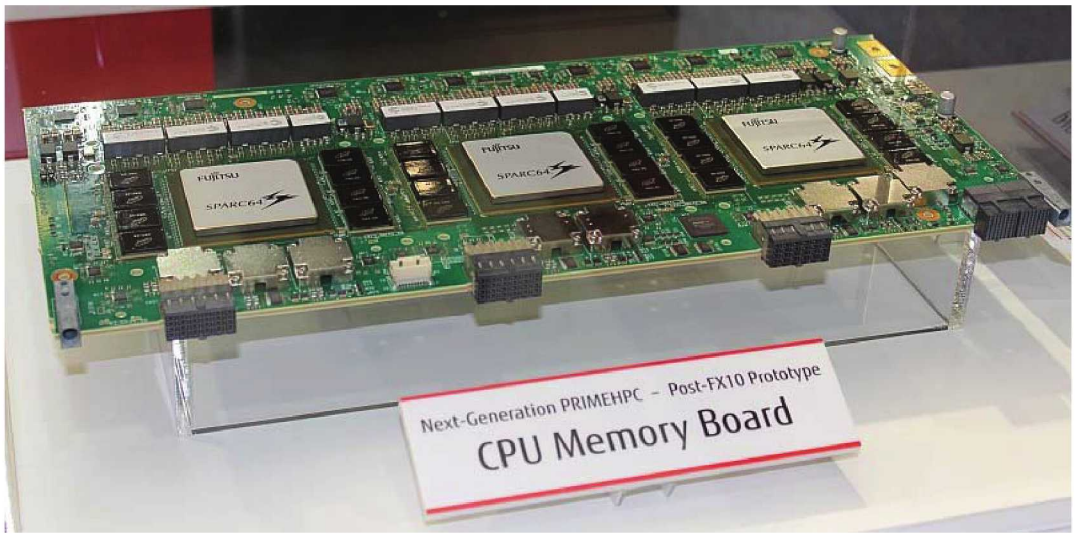
4.6.4 3D Standalone Systems

An obvious next step is the integration of full-fledged processing logic onto the logic die of a memory stack, with the use of the off-stack links to connect only to other stacks. No other processing sockets would be needed, and systems could consist of seas of such stacks.

A study [21] of a big data problem demonstrated the comparative benefit of such a system in

[14] http://blogs.nvidia.com/blog/2014/03/25/gpu-roadmap-pascal/

[15] https://www.fujitsu.com/global/Images/fujitsu-hpc-roadmap-beyond-petascale-computing.pdf



http://www.fujitsu.com/global/Images/post-fx10-cpu-memory-board_tcm100-958139.jpg

Figure 4.14. Fujitsu's Post FX10 Board.

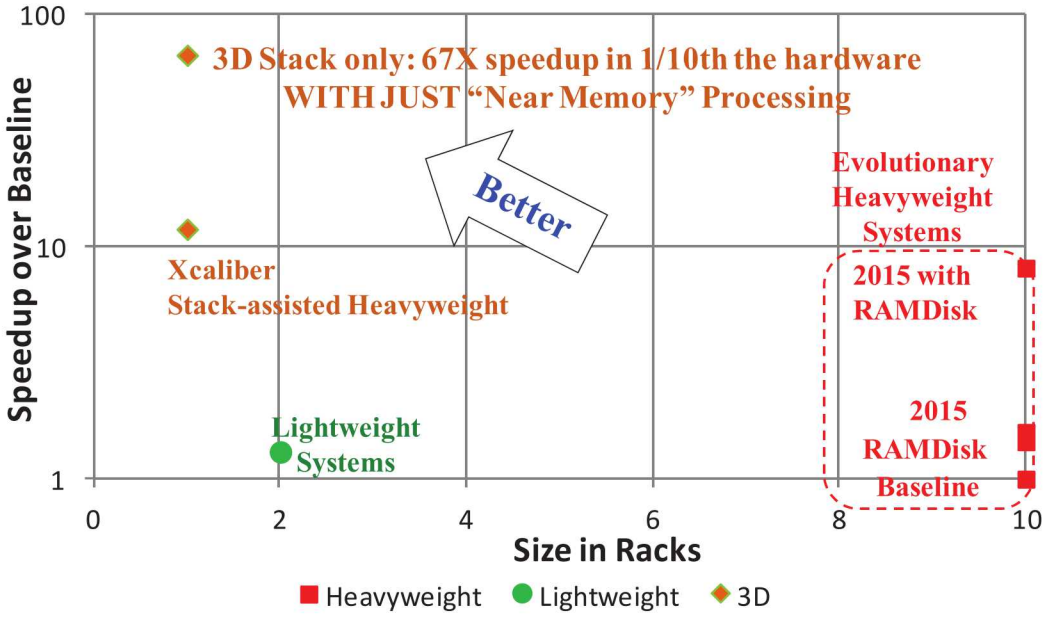


Figure 4.15. 3D versus Conventional Solutions.

comparison to a variety of other systems, including projections of future heavyweight, lightweight, and 2.5D systems such as X-caliber. The results, pictured in Fig. 4.15 suggest both an explosive increase in performance at a significant reduction is space (the baseline system running today requires 10 racks of heavyweight servers).

Going forward with such architectures do, however, introduce several considerations.

- First is density of power dissipation, since in order to be effective such stacks will want to be packaged close together. This may be mitigated by designing the off-stack transceivers for short and/or ultrashort range.
- Next is the need to integrate in non-volatile memory of some sort. X-caliber had such, positioned as a second stack of a different memory type over the bottom logic die which now had a footprint large enough to include both the DRAM stack and the non-volatile stack. In the referenced study, this distributed non-volatile memory was used for the original data sets and to hold the results. It also would be useful in holding program code, at least for bootup.
- Finally, while current stacks include a great real of redundancy logic, the dense packaging of many such stacks on the same package substrate is liable to mean that repair involving rework is liable to be difficult, meaning that such architectures must include a great deal of resiliency in their design.

This page intentionally left blank.

Chapter 5

Historical Data

The major purpose of this report is to track progress towards "exascale" by the best of current and projected technology, as a function of architecture. From its beginning, the TOP500 (Section 2.1) has focused on performance of LINPACK on the leading parallel computers of the day. More recently GRAPH500 (Section 2.2) has arisen to give insight into non-flop-intensive applications. Both have the key advantage of being tracked in a consistent way twice a year over multiple years. To allow for year-over-year comparison, the rankings have used a few simple terms to specify both the hardware architecture and the performance. While satisfactory for most of the early years, with the rise of SMPs, multi-cores, and now hybrid architectures, making "apples-to-apples" comparisons has become more difficult, especially when detailed analysis and projections are desired as was done for the original Exascale report.

In addition to the TOP500 and the GRAPH500, two other historical records are discussed here: a proposed follow-on to LINPACK called HPCG (Section 2.4), and a self-reported set of benchmarks derived from the DARPA HPCS project (Section 2.3).

In the following subsections we review the terminology used in this paper (and suggest that the use of some variants be defined and measured in future rankings to simplify future analysis), and then go through historical trend charts for each of the rankings, so as to understand what has happened in the past.

Throughout these charts an attempt is made to standardize the shape and color of data points as follows:

- Red squares relate to systems with Heavyweight nodes.
- Green circles relate to systems with Lightweight nodes.
- Purple triangles relate to systems with Hybrid nodes.
- Gold diamonds relate to systems with architectures different from any of the above (GRAPH500 only).

5.1 Basic System Parameters

Until the November 2008 TOP500 list, the key parameter used by the TOP500 to describe the architecture of a system was "Processor Count." This became quite fuzzy with the rise of multi-core, and was replaced with "core count." Even that, however, is still inadequate for the kinds of projections done here. Consequently, the following is the list of terms used in this report[1]:

- **Core**: for a CPU, a set of logic that is capable of independent execution of one or more program threads concurrently.

  For GPUs, the terms *SIMD core* and *Streaming Multiprocessor* (SM) have been used by AMD and NVIDIA respectively. Each such unit has a number (typically 16-48) of replicated simple processors (unified *shaders*) which contain the ALUs and increasingly FPUs. However, each such unit executes a common thread of instructions, and thus is counted as a single core (albeit with a great deal of internal computational concurrency).

- **Socket**: a package, typically containing a single chip, that provides a very large amount of logic for key processing functions, and reflects that when one looks at a modern HPC system, the biggest area on a board is devoted to the heatsink and socket that sandwich such a chip. The key socket is for processing, but it is not uncommon to see additional sockets for chips that perform routing and communications functions, often called **Network Interface Controllers** or **NICs**.

  In earlier rankings, a processor socket was primarily a single-core microprocessor chip, and thus counting sockets was equivalent to counting "processors." Today, a compute socket typically supports a large number of cores, and may be either a conventional "microprocessor" or an "accelerator" such as a **Graphics Processing Unit** (**GPU**), or, increasingly, a mix.

- **Node**: the set of sockets, associated memory, and NICs that represent the basic replicable unit of the system, *as seen by application software*. Several years ago the term "node" came into vogue, and was used interchangeably with "processor" and "socket." With the rise first of chip sets that couple multiple identical microprocessor chips into a single unified SMP (Symmetric Multi-Processor), and then by the rise of multi-socket machines with mixes of heterogeneous microprocessors (as in Roadrunner), it became important to make this distinction.

- **Core Clock**: the clock rate of an individual core as it executes instructions. In systems with multiple types of cores we will distinguish between the clocks of different core types.

- **Compute Cycles per Second**: With the emergence of hybrid architecture systems there are at least two major clocks, that for the cores in the main heavyweight socket, and that for the cores in the accelerator. This metric computes the sum of the product of core count and core clock rate for both types, and expresses the total number of times per second in which some core can perform a cycle of computation. In a sense it is an "aggregate clock."

[1] These definitions were first made for the Exascale report[20], amplified in the SC11 paper[19], and then refined in a more precise fashion for this document.

- **Memory Capacity**: the aggregate physical memory (typically DRAM) that holds non-transient data and is directly addressable by programs running in cores within sockets, within nodes, not including caches.

  Note that future systems will use non-volatile memory (**NVRAM**) as an additional first-level memory to support large static data sets, code, and former disk functions such as checkpoint storage.

  Most accelerator chips today, and systems that use a combination of conventional microprocessors and accelerators, use memory separate from that of the host processor. Such memory is normally much smaller and used as temporary staging areas for pieces of data in the host processor's memory. Thus, such memory, if present, should be recorded separately from the host memory, as it is the size of host memory that typically controls the biggest problems that a system can handle. Unfortunately, in many reports only an aggregate number is given.

- **Power**: total power consumed by the system. This metric today is often recorded inconsistently across system descriptions, depending on whether or not file systems (disk drives) are included, and/or whether the power required for cooling and power conditioning is counted. In the future it may be more consistent to quote perhaps **CEC power**[2] as the power actually drawn by the computing electronics, **system power** to include secondary storage and local power conditioning, and **facility power** to include cooling and power conversion and redundancy effects.

5.2 Performance Metrics

A second set of terms is important from the standpoint of comparing systems. The most obvious ones are from the prior benchmarks:

- R_{peak} and R_{max} which drive the TOP500 rankings[8] (Section 2.1), with N_{max} and $N_{1/2}$ as storage metrics. Unfortunately, very few reports include these metrics.
- **TEPS** (Traversed Edges Per Second) which measured the performance from the GRAPH500 (Section 2.2), with problem size as a storage metric. Unlike TOP500, this storage metric is an integral part of the reporting process.

The Exascale report[20] added the following metrics:

- **Thread Level Parallelism** (**TLP**): the number of distinct hardware-supported concurrent threads that makes up the execution of a program.

  None of the top systems to date have been explicitly multi-threaded, although newer chips do support at least a low level of "Hyper-Threading." Consequently, for this study each core as reported in the TOP500 list is assumed to correspond to a single thread of execution.

[2]CEC is an old term for "Central Electronics Complex"

- **Thread Level Concurrency** (**TLC**): is an attempt to measure the number of separate operations of interest that can be executed per cycle per thread.

  For the TOP500 the operation of interest has been floating point. It is computed as the performance metric (R_{max} or R_{peak}) divided by the total number of compute cycles in a second.

  TLC is meant to be similar to the Instruction Level Parallelism (ILP) term used in computer architecture to measure the number of instructions from a single thread that can either be issued per cycle within a microprocessor core (akin to a "peak" measurement), or the number of instructions that are actually completed and retired per second (akin to the sustained or "max" numbers of the current discussion).

- **Total Concurrency** (**TC**): the total number of separate operations of interest that can be computed in a system at each clock cycle.

  For the TOP500 such a measure reflects (within a factor of 2 to account for fused multiply-add) the total number of distinct hardware units capable of computing those operations. This metric can be computed as the number of cores times the peak TLC, and is important because it reflects the explicit amount of concurrency that must be expressed by, and extracted from, a program to utilize the hardware efficiently.

- **Flop/s per watt** (**FPW**): the performance of the system divided by system power.

- **Energy per flop** (**EPF**): the reciprocal of the above, in units of Joules (or more conveniently in most cases "picoJoules" (pJ), 10^{-12} Joules).

We note that many of these metrics may actually be computable in several forms depending on which performance metric (R_{max}, R_{peak}, or TEPS) is used. We also note that one interpretation for exascale is for an exaflop/s at 20MW, which is equivalent to an energy expended per flop of 20pJ.

5.3 Historical Results: TOP500

5.3.1 Basic Performance Characteristics

The most common observations about the TOP500 come from fitting simple trend lines to the key system characteristics. Figs. 5.1, 5.2, and 5.3 list R_{peak}, R_{max}, and system memory capacity as scatter plots versus time. These graphs plot just the top 10 systems from each of the bi-annual lists. The first two also are similar to the lead charts from the TOP500 website. Also included on each are simple curves based on a constant compound annual growth rate that fit key regions of the plots. This CAGR rate is listed in the legend of each figure.

These graphs also show the introduction of alternative architectures, first the lightweights in 2004 (green circles) and then the hybrids in 2008 (purple triangles).

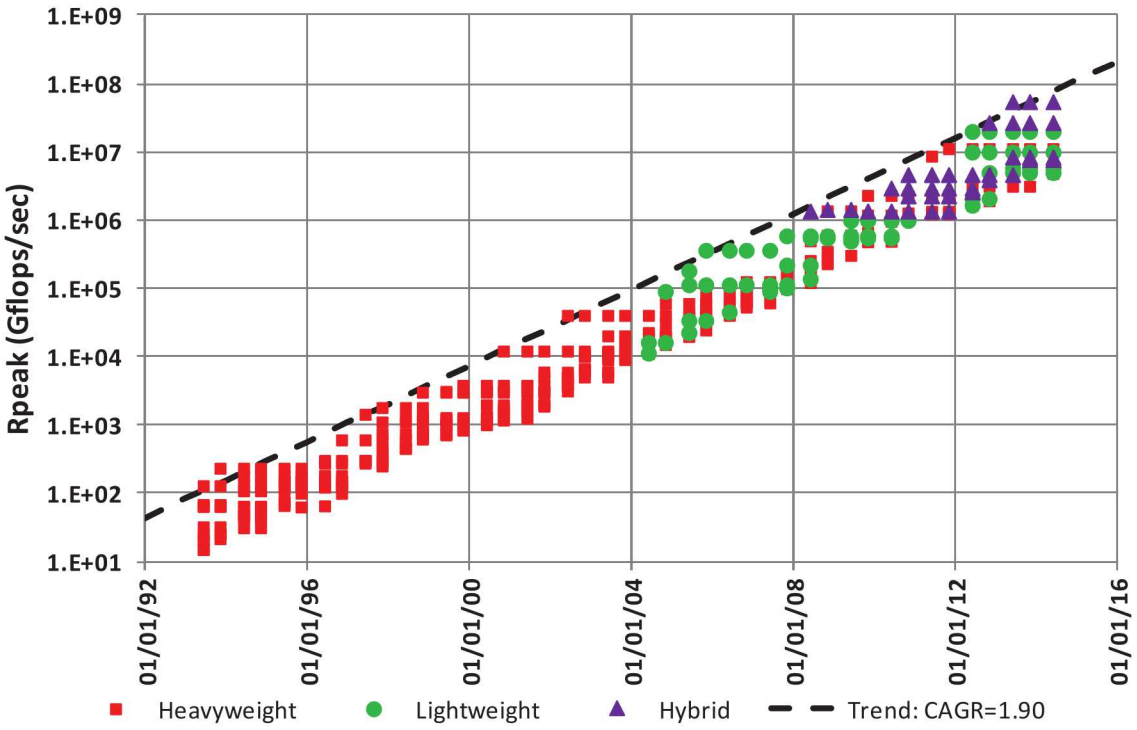


Figure 5.1. TOP500 Rpeak versus Time.

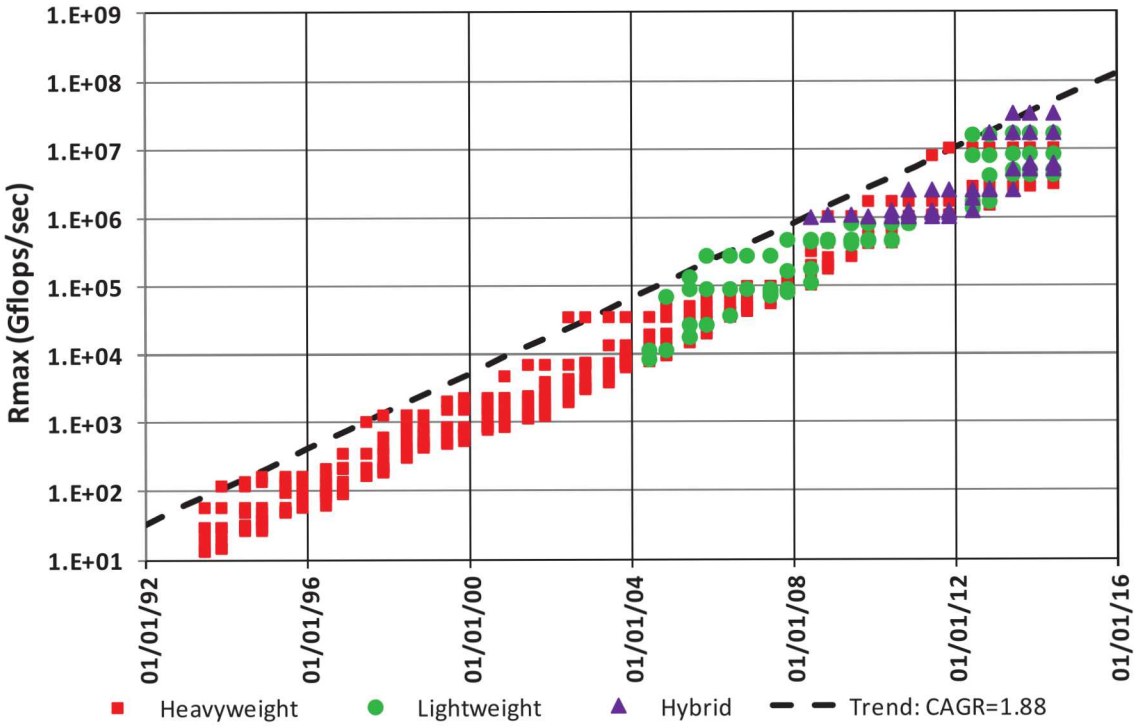


Figure 5.2. TOP500 Rmax versus Time.

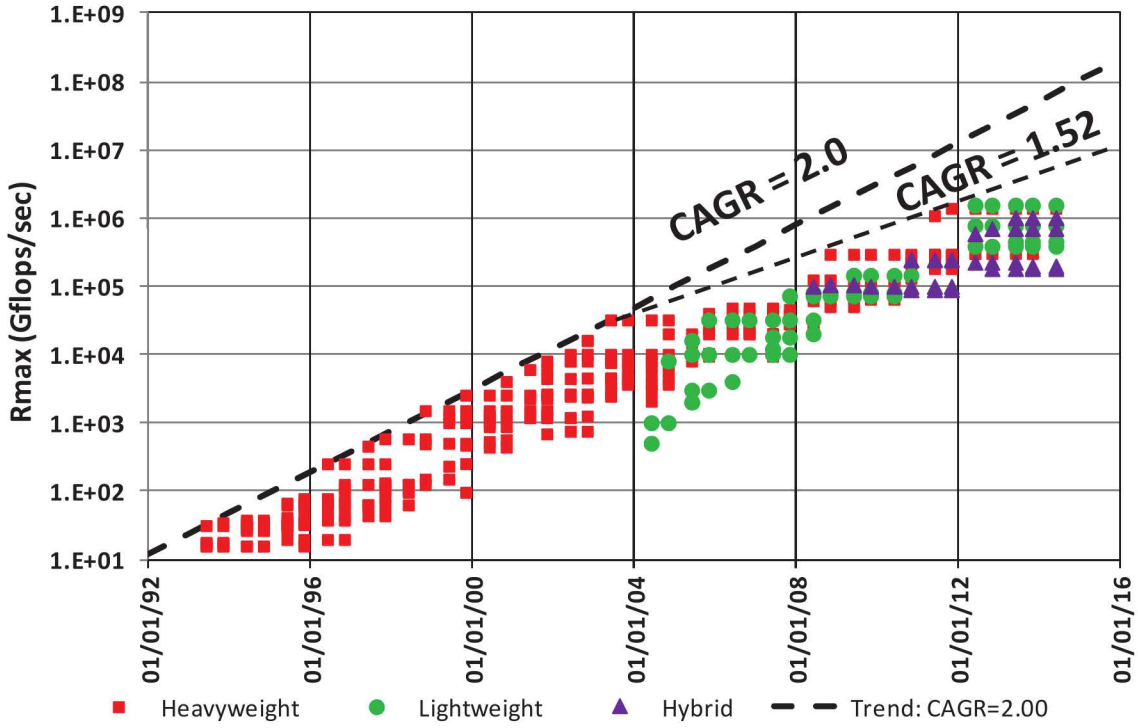


Figure 5.3. TOP500 Memory versus Time.

Both R_{peak} and R_{max} have increased at virtually the same CAGR for the last 20 years, including the 2004 transition. It is interesting that R_{peak} measurements between the top system and the number 10 system has consistently been about an order of magnitude, with the same spread for R_{max} being somewhat less.

In contrast, the total memory per system increased from 1992 through 2004 at a slightly higher rate (2X/year) than either R_{peak} (1.9X per year) or R_{max} (1.88X per year). However, after 2004 the maximum memory per system takes a distinctly stair-step pattern, with an overall CAGR for this part of the graph is more like 1.52 rather than 2.0.

Also as a note, the hybrid measurements includes both the memory for the heavyweight sockets and the accelerator memory for the accelerator sockets. Since the accelerator memory is used as transient intermediate storage, strictly speaking, this should not be included.

5.3.2 Per flops/s Metrics

Two metrics that are ratios of these metrics are **efficiency**, the ratio of R_{max} to R_{peak}, and **bytes per flop/second**. The former is a view of how well a system is in converting peak theoretical capability into sustained capability. The latter is a metric where the decades-old conventional "rule of thumb"

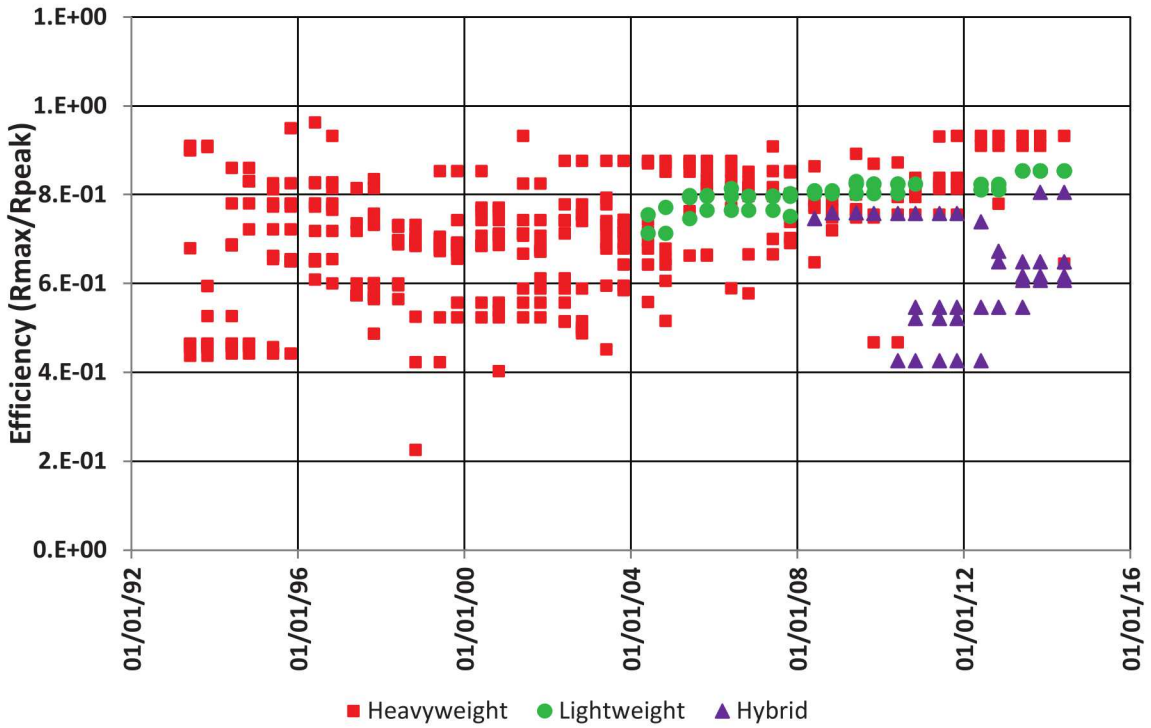


Figure 5.4. TOP500 Efficiency versus Time.

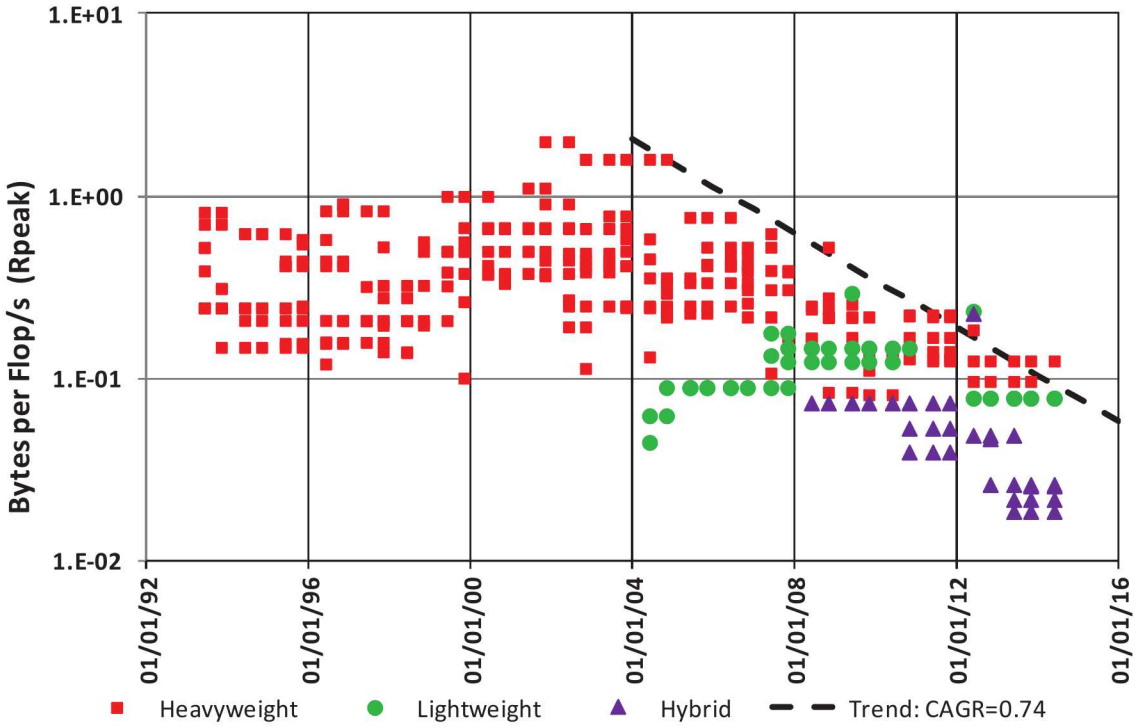


Figure 5.5. TOP500 Bytes/Rpeak versus Time.

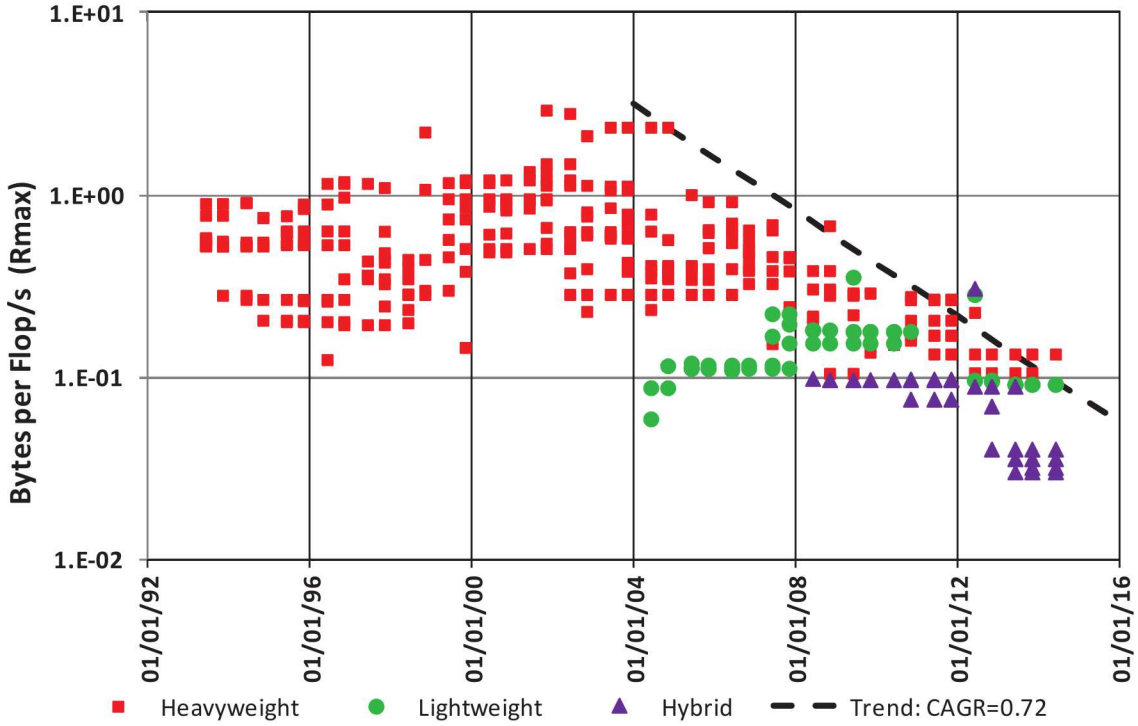


Figure 5.6. TOP500 Bytes/Rmax versus Time.

is that a byte of memory is needed for each flop executed per second.

Fig. 5.4 diagrams the efficiency versus time. This has converged in a tight band from 90% to 90+% for the heavy and lightweight systems. The early hybrid systems exhibited a precipitous drop down to the 40% range, with more recent systems converging on efficiencies in the 60 to 90% range.

An interesting consideration for the hybrids is that there are FPUs in both the main heavyweight processor and in the GPU accelerator, but most of the core floating point in LINPACK is in the accelerator FPUs. Thus this aggregate efficiency is probably too low if we look at just accelerator flops, and too high if we look at just heavyweight flops. A further study should focus on how many of each are actually used.

In terms of main memory per flop/s, Figs. 5.5 and 5.6 show similar trends using both R_{max} and R_{peak}. Before 2004, the best systems had in fact about 1 byte per flop/sec, but after 2004, this ratio has declined by a ratio of 0.75 per year (about 1/2 every 2 years). Also, as can be seen, the lightweight and hybrid systems are even worse. This tells us that flops are increasing significantly faster than bytes.

5.3.3 Socket and Core Growth

Fig. 5.7 gives a time line for the growth in sockets and cores. The solid points correspond to sockets; the hollow points correspond to cores. As before, the red, green, and purple refer to heavyweight, lightweight, and hybrid respectively.

Up until 2004 the core count overlaps the socket count; a core per socket was the norm, and the maximum count of either in a system was flat at about 10,000. Since then, however, there is a growing divergence, reflecting the growth in multi-core implementations, with a 10X increase in sockets and a 100X increase in cores. In particular, the socket count went through a rapid transition from 10,000 to about 100,000 and has been flat. The core count, however, has been growing at a CAGR of 1.67 per year. This is clear evidence of the switch from increasing clock rate to increasing physical parallelism in architectures after 2004.

Note that for single-threaded cores, the TLP metric defined earlier is the same as the core count in Fig. 5.7.

5.3.4 Clocks and Concurrency Metrics

Perhaps of even more interest are details of the way performance is obtained. Fig. 5.8 graphs the max clock rate of the cores in a system, with several obvious conclusions. First, from 1992 through 2003 the clock rate grew at a CAGR of about 1.4, but the peak clock rate did in fact flatten around 2004 for the heavyweight sockets. Second, the lightweight machines chose a significantly lower clock rate than the commodity heavyweight, and even today are perhaps 1/3rd of the heavyweight

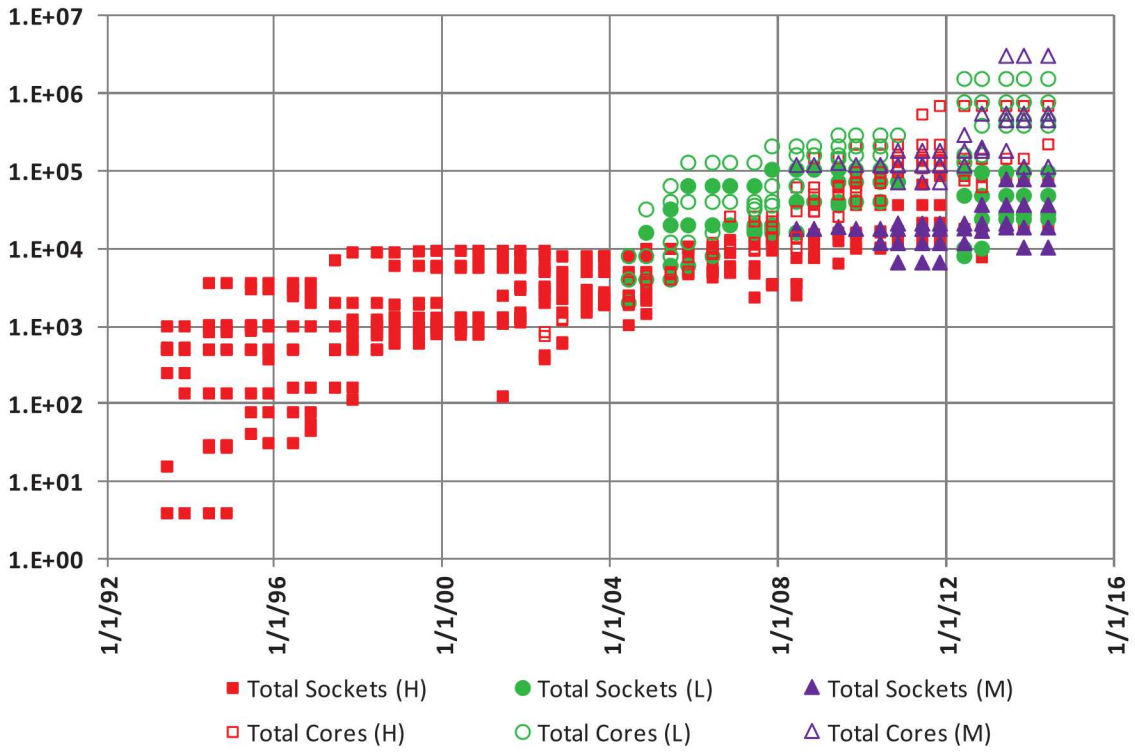


Figure 5.7. Growth in Socket and Core Count.

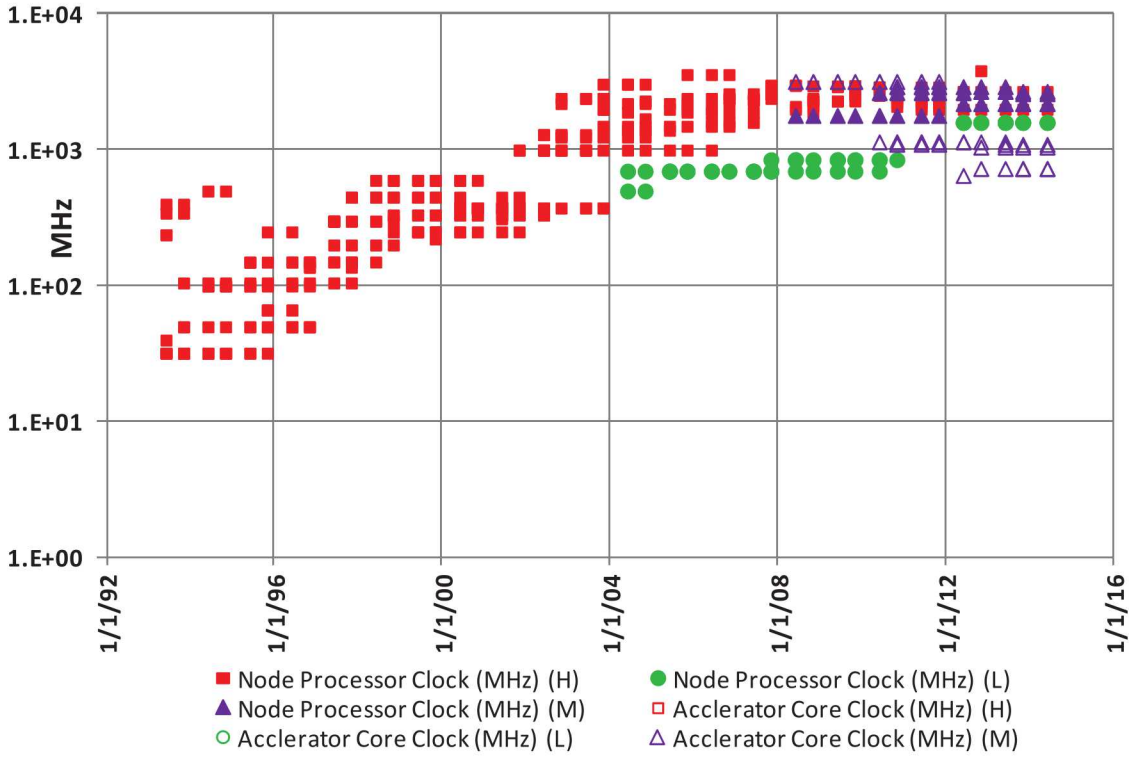


Figure 5.8. Trends in Clock Rate.

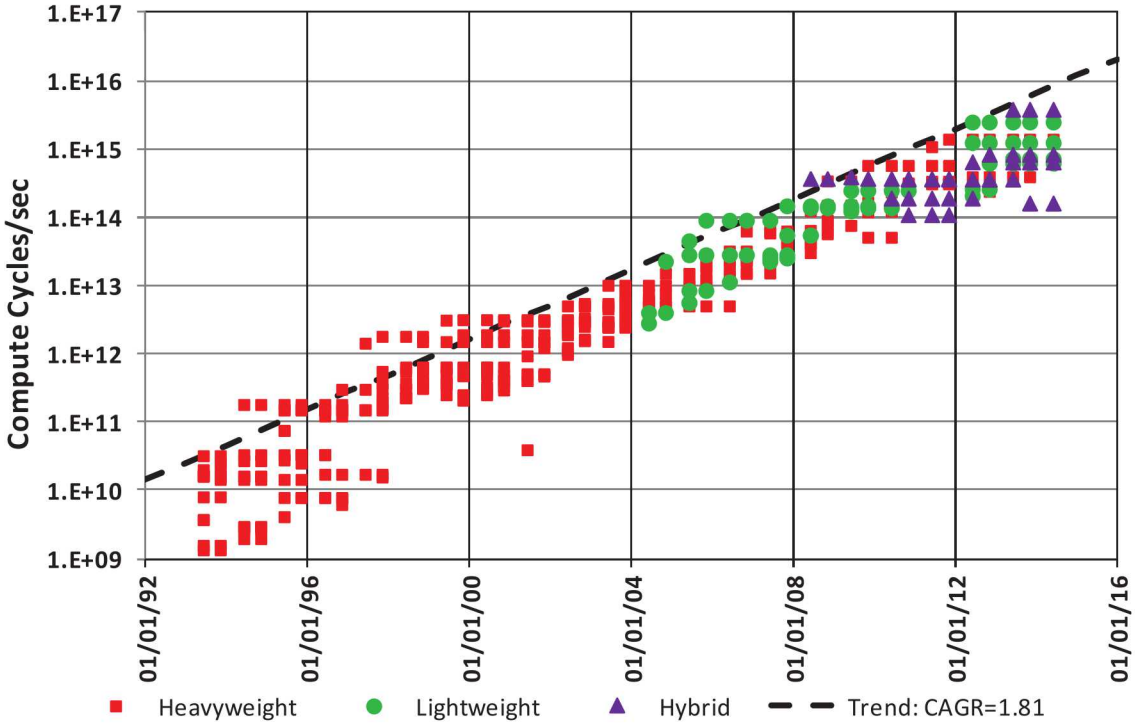


Figure 5.9. Compute Cycles per Second.

maximum. Third, the hybrid systems have two clock measurements, one for the main heavyweight socket and one for the accelerator socket. The former has followed the heavyweight trends; the latter the lightweight trends.

An additional insight is the sum total of compute cycles available on all cores per second. Fig. 5.9 diagrams this metric, and with the exception of a few years just before 2000, there was an uninterrupted CAGR of 1.8 for the entire period.

We note that this aggregate cycle count is growing slightly slower than either R_{peak} or R_{max}. Clearly the increase in performance must be due to performing more computations per cycle per thread/core (the TLC metric discussed earlier). Fig. 5.10 diagrams this count of flops per cycle for both R_{max} (solid points) and R_{peak} (hollow points). With very few exceptions (primarily the Earth Simulator in the early 2000s'), the max for this was about 4 until the rise of the hybrids and the presence of GPU cores with 10s of FPUs. Currently, the hybrids deliver about 10X both the max and the peak flops/s than the non-hybrids.

Again, there is interesting insight to be gained for the hybrid systems if we separate out host and accelerator.

Next, total concurrency (TC) is the total number of operations that must be "in the air" at each and every machine cycle. Equivalently, TC is the number of operations "retired" or completed per

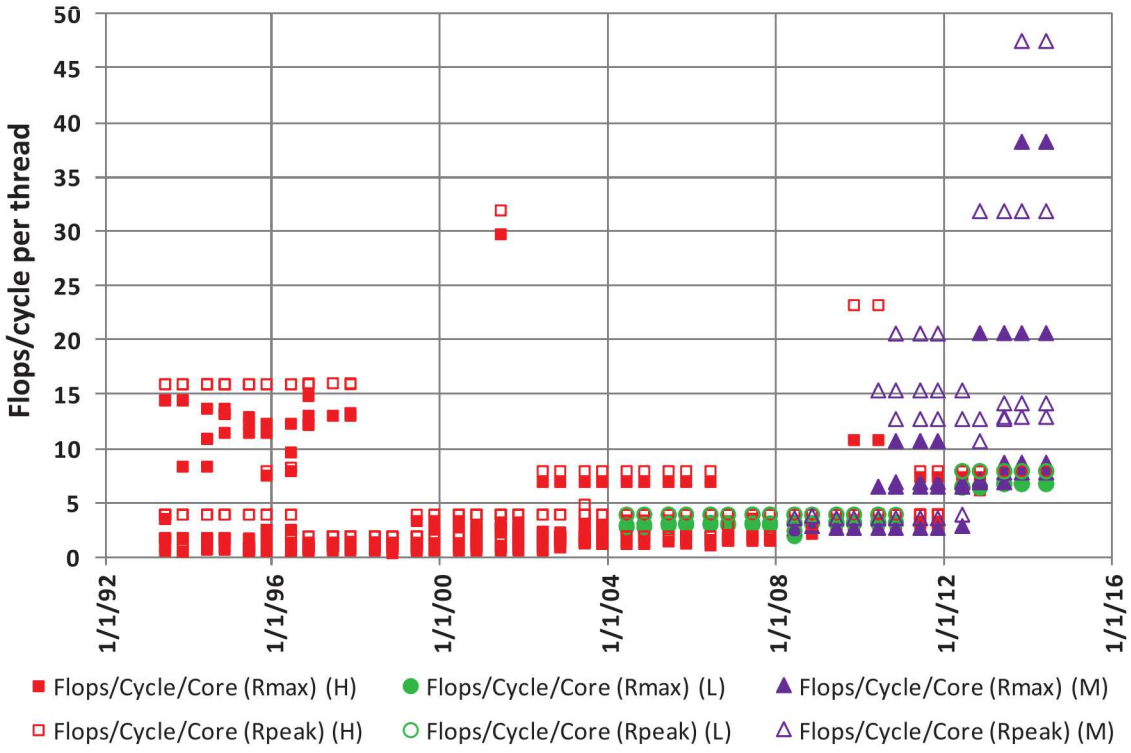


Figure 5.10. Thread Level Concurrency (TLC): Flops per cycle.

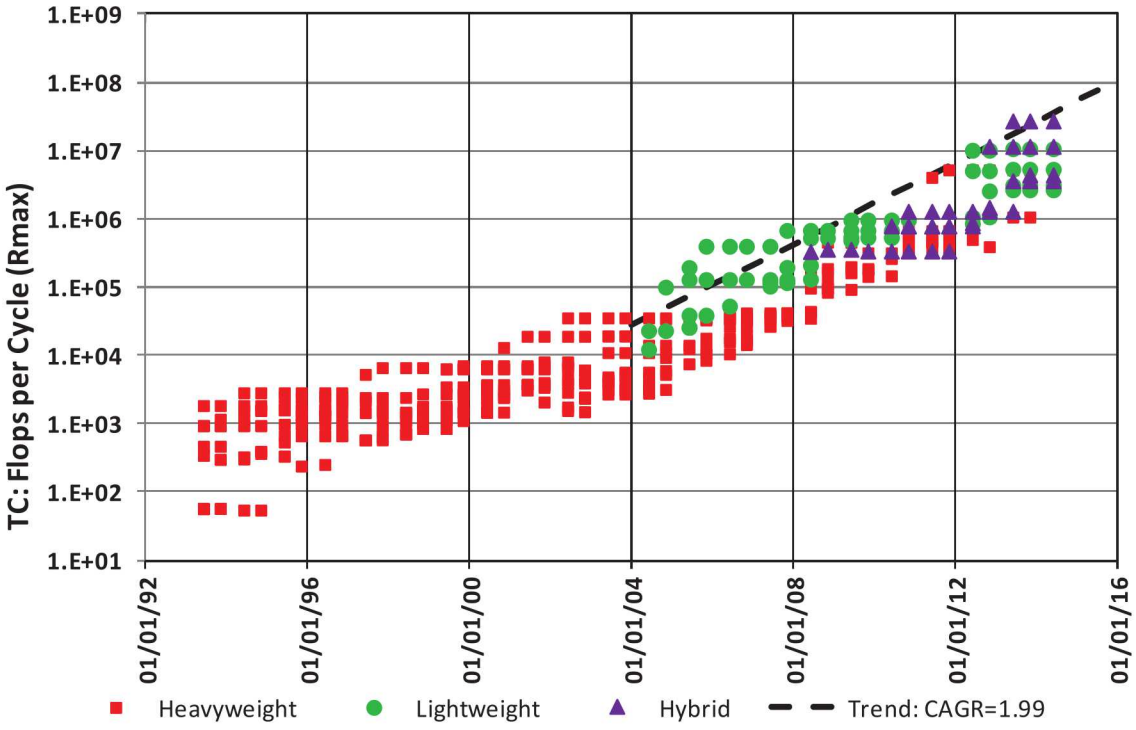


Figure 5.11. Total Concurrency (TC).

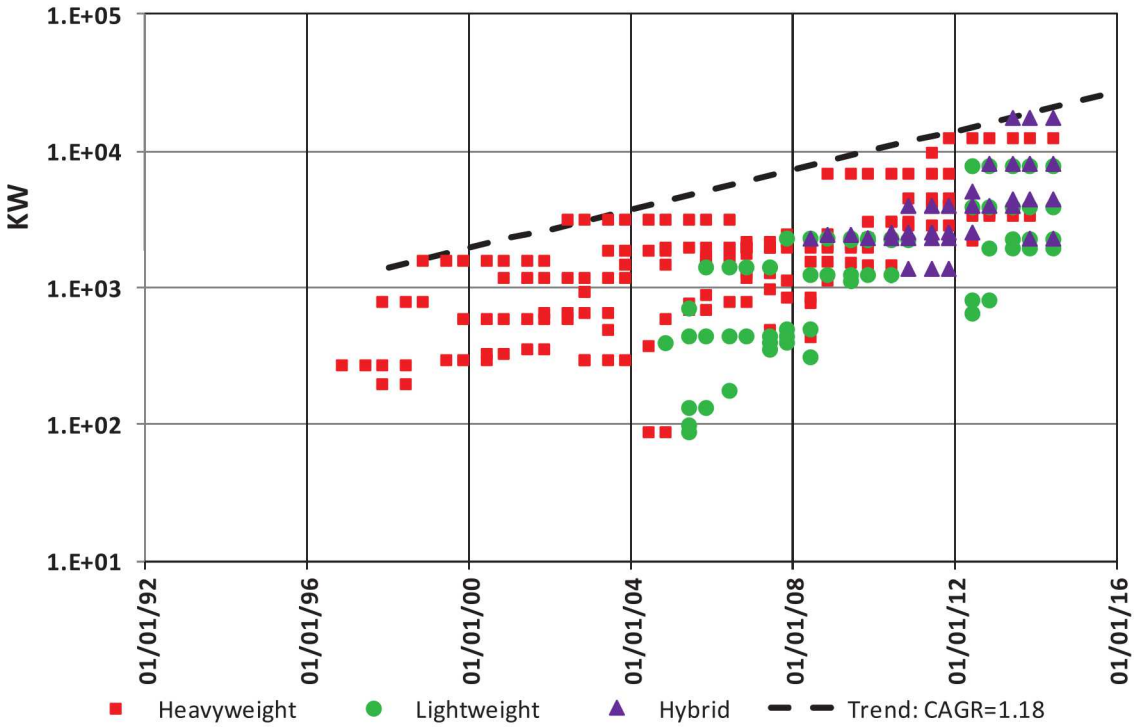


Figure 5.12. System Power.

second. This is important because it reflects the amount of parallelism that must be dragged out of the application in each and every cycle. Fig. 5.11 shows this metric by dividing R_{max} by the clock rate of a core in the system (for hybrid systems we compute an "average" rate based on the mix of heavyweight and accelerator cores). This graph shows for that the long stretch of time when technology permitted clock rates to rise, the TC value stayed between a few thousand to a few tens of thousands, at a CAGR of about 1.4. However, once the clock rate flattened (about 2004), the only way to get additional performance was through brute parallelism, and this number then began to take off, with a CAGR that grew to 2X/year.

5.3.5 Power and Energy

Fig. 5.12 diagrams total system power. The trends here are a bit harder to see, other than an average peak of 1-3 MW until 2008, when a rapid climb to 10-20 MW after that. Fig. 5.13 is perhaps more revealing by computing total system power divided by the number of compute sockets. This includes aggregate power from memory, routers, and other support circuitry. Here the heavyweights before 2004 showed an average CAGR of 1.32X/year, and then precipitously declined after reaching nearly 700W per socket. The lightweights are in the much lower 20-30W per socket, and the heterogeneous are now considerably higher per socket than the heavyweight alone. We note that

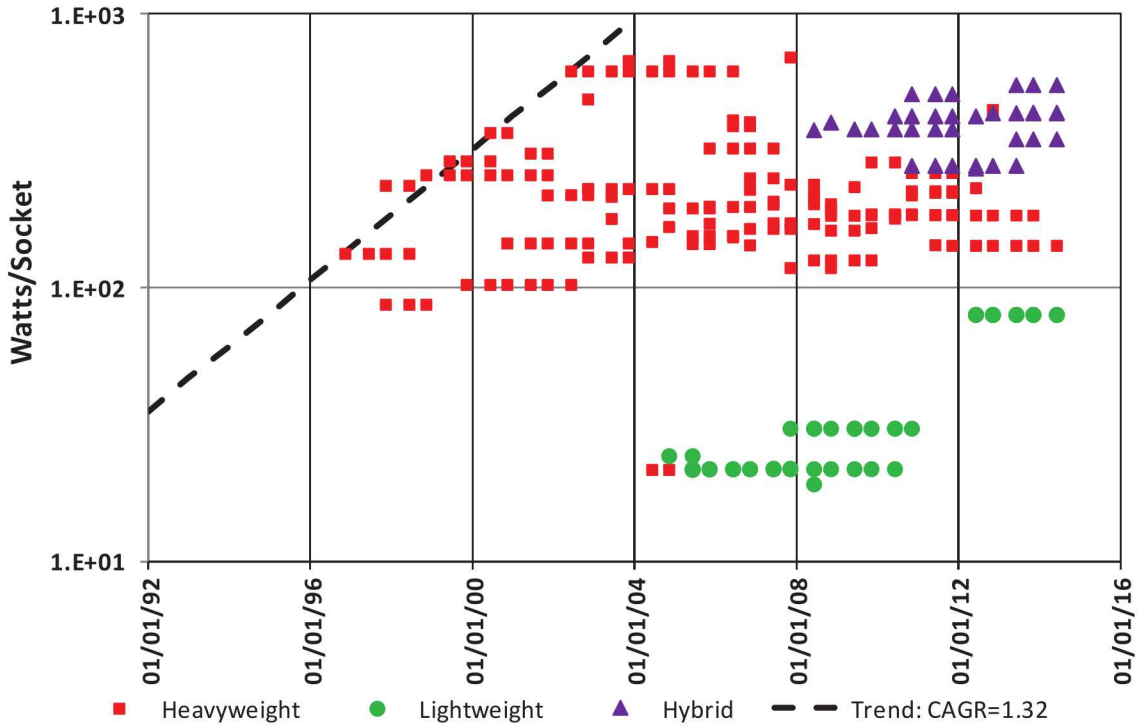


Figure 5.13. Power per Socket.

in the latter case, the socket count includes the microprocessors and the GPUs.

Finally, Fig. 5.14 divides total power by R_{max} to get an energy expended per flop executed. The best of these points, which appear at about 4 year intervals, matches up with a CAGR that is declining at about 0.6X per year, a reduction of about 8X every 4 years. However, starting in 2004, the heavyweight systems began to diverge away from this CAGR, while the lightweight architectures, especially Blue Gene/L, provided a one time improvement that moved the best points back into the CAGR 0.6X trend, and after 2008 the hybrid architectures also filled in the gap between the heavyweights and the trend line.

5.4 Historical Results: GREEN500

As discussed in Section 2.1, the GREEN500 benchmark is the same at the TOP500, but with flops/sec per watt as the metric, and R_{max} used as the point of comparison.

Fig. 5.15 inverts this metric to energy: pJ per flop, and overlays for comparison purposes the equivalent numbers for the top 10 of the TOP500 rankings from Fig. 5.14. Solid points are from the TOP500; hollow points are from the GREEN500.

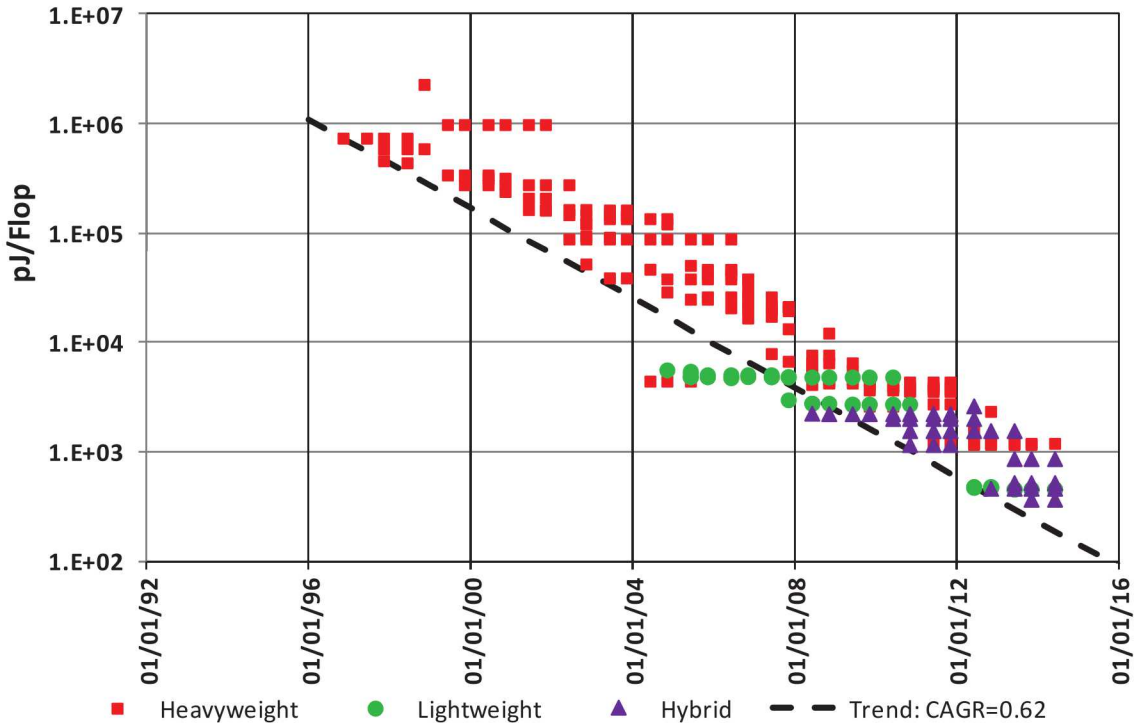


Figure 5.14. Energy per flop with Trend Line.

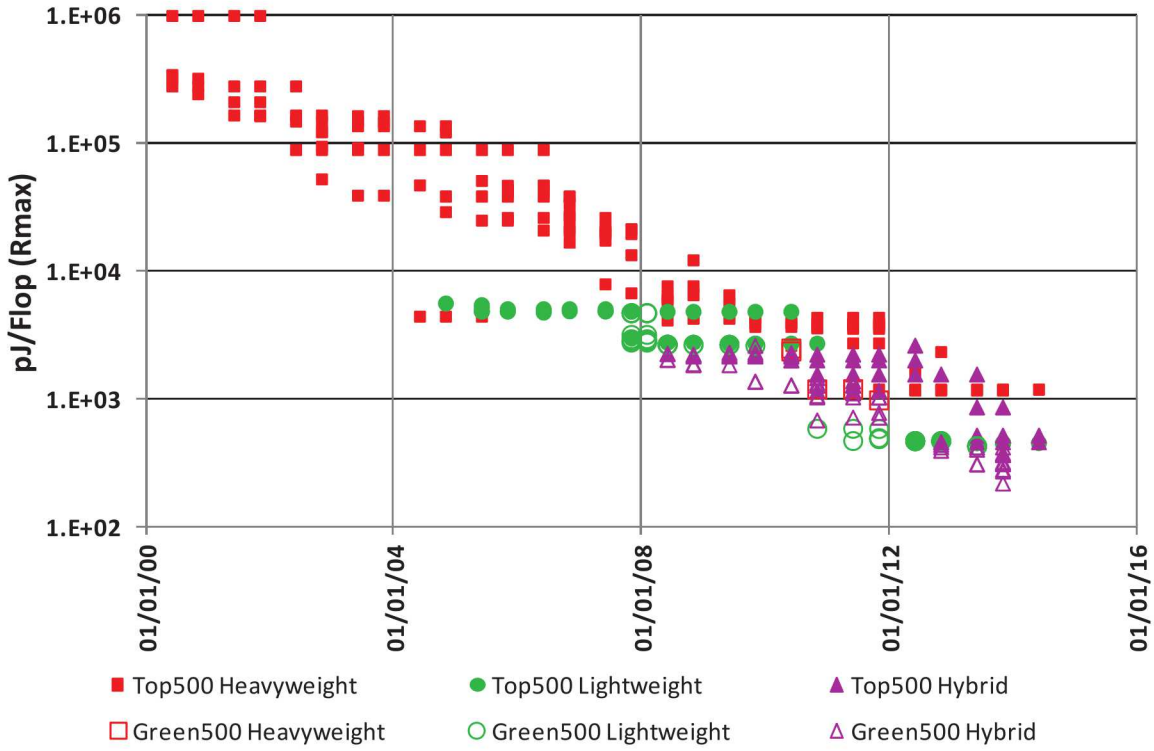


Figure 5.15. Energy per Flop vs Time: TOP500 vs Green500.

Overall, the GREEN500 systems are more or less consistently better in pJ/flop than the best of the TOP500 by perhaps a factor of 3X.

The prior figures all looked at these metrics as a function of time, so that improvements could clearly be seen. Fig. 5.16 graphs the same data points as Fig. 5.15, but measures against the R_{max} value achieved by the energy point. No distinction is made between the different time frames of the measurements. The key result is that up to about a petaflop there are GREEN500 systems at better energy than TOP500, but that stops above a petaflop where there are no GREEN500 systems.

5.5 Historical Results: GRAPH500

The GRAPH500 data, as described in Section 2.2, currently consists of a single benchmark to perform a breadth-first search through a potentially very large graph (up to 4 trillion nodes). The performance metric, TEPS, is the total time divided by the number of graph edges traversed.

There are now multiple cycles of performance data recorded, starting in November 2010, with data on the systems approximately similar in detail to that reported on the TOP500 lists. Unlike the data analysis for the TOP500, this analysis has used all the entries in all rankings, simply be-

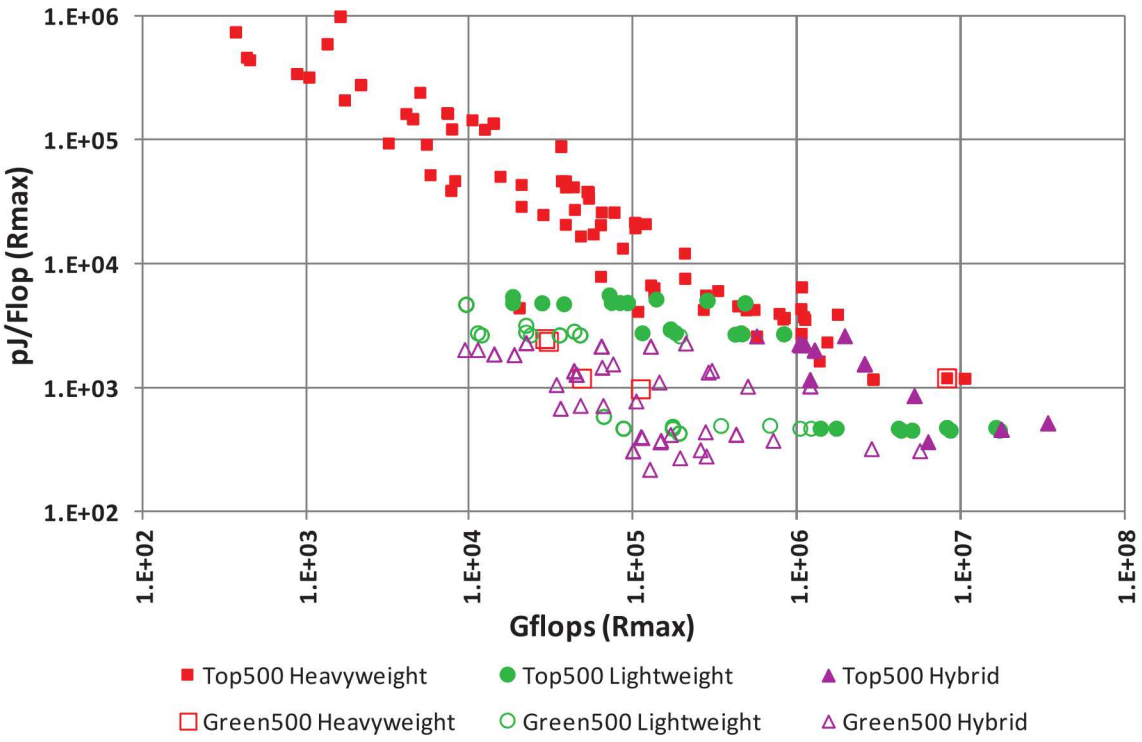


Figure 5.16. Energy per Flop vs R_{max}: TOP500 vs Green500.

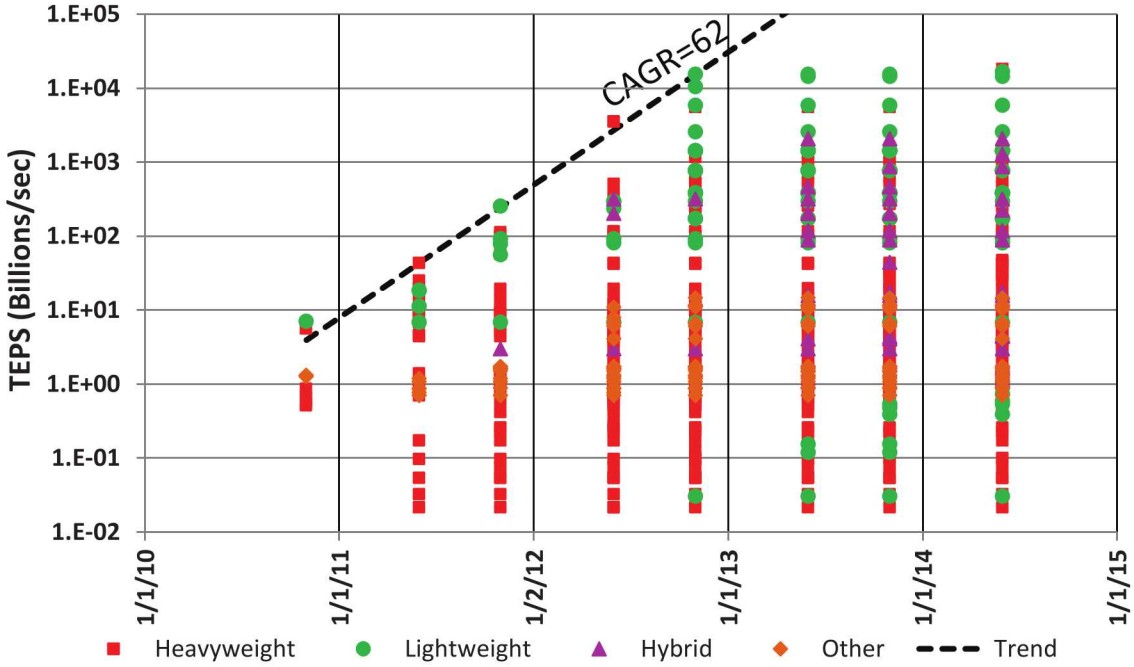


Figure 5.17. TEPS versus Time.

cause there hasn't yet been a long trail in measurements, and the software approaches to maximize performances are not as well-developed as for LINPACK.

Fig. 5.17 diagrams the growth in TEPS over these measurements. As can be seen, over the first five lists the growth in peak reported TEPS was an astonishing 62X per year, but has totally flattened since June 2013.

In Fig. 5.17 and future graphs a fourth symbol is introduced, a "diamond," for types other than those used for the TOP500 system types.

5.5.1 Architectures

The current data allowed marking data points as being related to approximately the same classes of architectures as in the TOP500, namely heavyweight, lightweight, and hybrid. In addition, however, a fourth class of architectures, marked here as **other**, has become prominent. Two system families made up this category:

- The Cray XMT-2 massively multi-threaded system. This architecture is particularly relevant to the GRAPH500 problem because it natively supports a large shared memory model, when a core anywhere in the system can directly access a memory anywhere else, without intervening software.

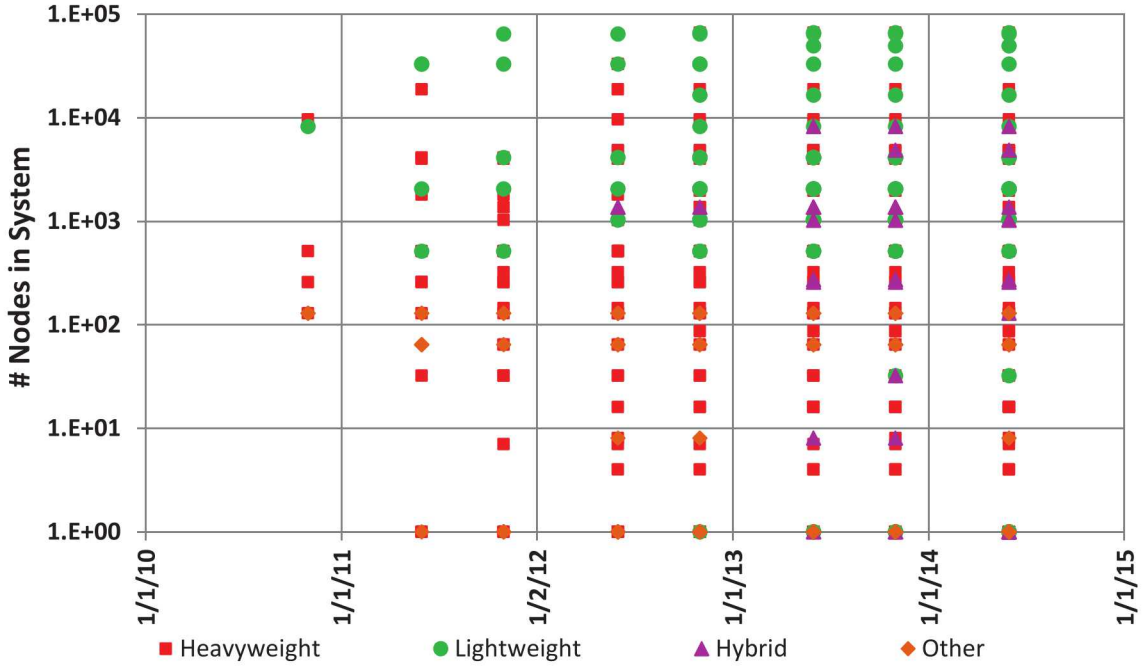


Figure 5.18. Node Count versus Time.

- Variations of the Convey FPGA-based systems[3]. This architecture is also particularly relevant because its memory system, albeit smaller than that possible with large clusters, has much more internal bandwidth, making it a good match again to the GRAPH500.

5.5.2 Performance Scalability

Fig. 5.17 shows a 62X improvement in performance per year. Although this has flattened as of June 2013, it is still important to understand where the scalability comes from within these architectures.

Fig. 5.18 graphs the number of nodes in these systems versus time, and in terms of peak numbers, there has been relatively no change over the last 18 months. All of the highest node-count systems have been Blue Gene systems.

However, if we look at TEPS versus the number of nodes in a system, Fig. 5.19, we see a very strong correlation. The trend line shown is proportional to the number of nodes to the 0.92 power. Thus doubling the number of nodes increases TEPS by 1.9X.

We posit that the less than ideal speedup projected by this factor is probably due to a combination of two factors:

[3]http://www.conveycomputer.com/

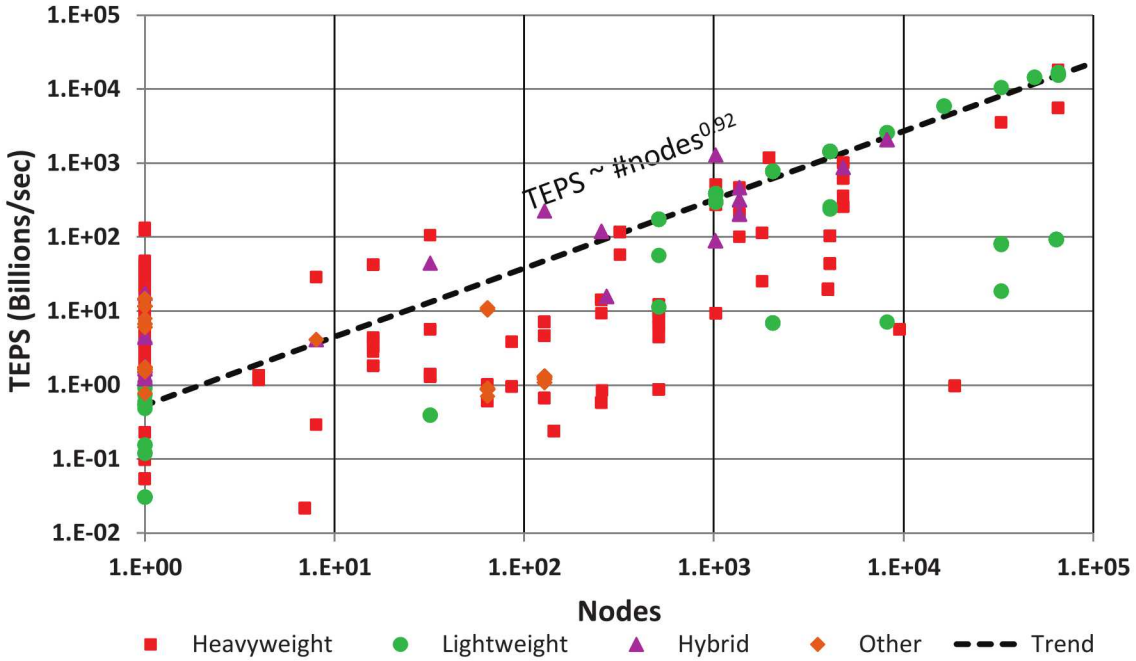


Figure 5.19. TEPS versus Node Count.

- In most implementations there are software barriers of some sort, where the wait time for getting past a barrier is often a nonlinear function of the number of nodes.
- When node-node I/O is used to carry messages about edge traversals around the system, the more nodes employed, the larger the share of bandwidth per nodes that may be consumed for traffic that is passing through, reducing the amount available for outgoing messages. In addition, the amount of management and general overhead is a non-linear function of the number of nodes.

  This is particularly appropriate for the Blue Gene architectures (the lightweight category) where there is no separate router chip and all router functions are managed within the processor socket.

A curve of TEPS versus the number of cores, Fig. 5.20, shows the same growth coefficient, although for very small systems, more cores have more effect than this trend line. This is probably due to many of these smaller machines being shared memory systems, so that node I/O does not play as great a role as it does for the larger systems.

Fig. 5.21 divides the TEPS by the number of nodes in a system, and plots by time. Fig. 5.22 graphs the same data but against the number of nodes in a system. There are several interesting observations:

- The peak values are increasing at a linear rate of 15.4 GTEPS per node per year.

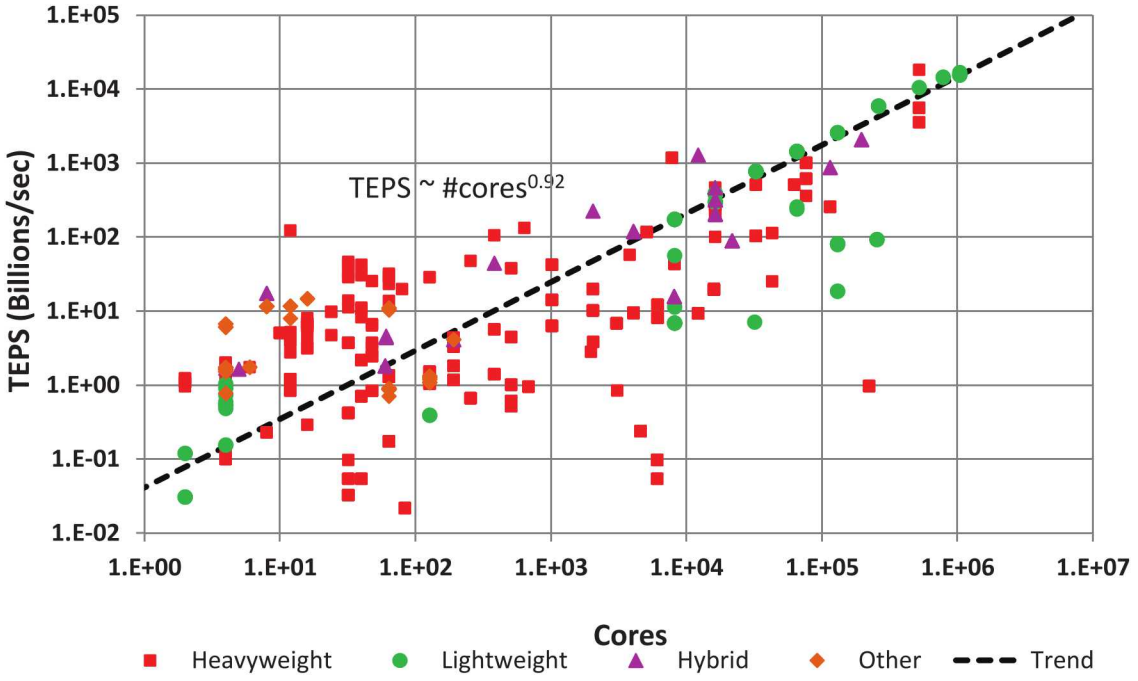


Figure 5.20. TEPS versus Core Count.

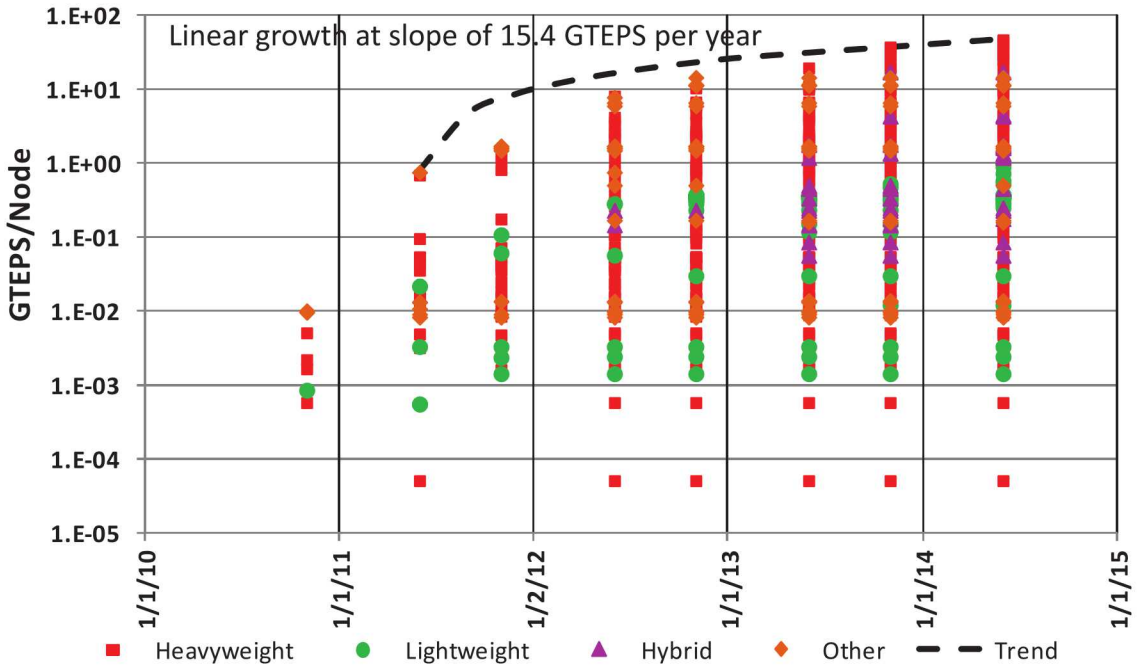


Figure 5.21. TEPS per Node versus Time.

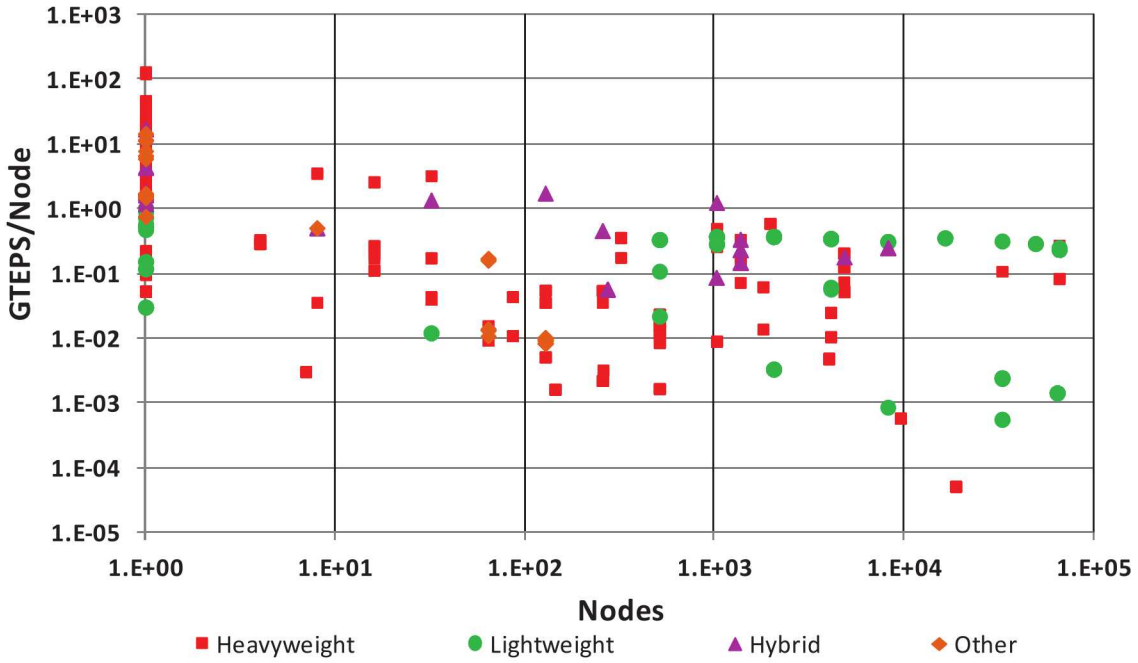


Figure 5.22. TEPS per Node versus Node Count.

- The best performance per node is from the "Other" category, mostly Convey systems which have a large number of memory ports in a single node box. From Fig. 5.22, however, the Convey system have limited scalability, and the XMT systems lose significant performance per node as they scale up.

- The best lightweight systems are perhaps a factor of 30 less in TEPS per node than the Convey numbers, but they both scale better, and something in excess of 32 more lightweight nodes can be packaged in the same cabinet volume as a Convey system today. However, once we get to systems of 512 nodes or bigger, the lightweight systems provide significantly more performance per node.

- There are multiple heavyweight systems that provide near best TEPS per node and are significantly better than the best of the lightweight, due we would guess to the higher memory bandwidth available. However, this advantage deteriorates rapidly with system size.

- Perhaps most importantly, we seem to have peaked over the last 18 months, with little gain per node. This may signal that we have hit an implementation limit where some new technology, perhaps interconnect link bandwidth, will be needed to foster more growth.

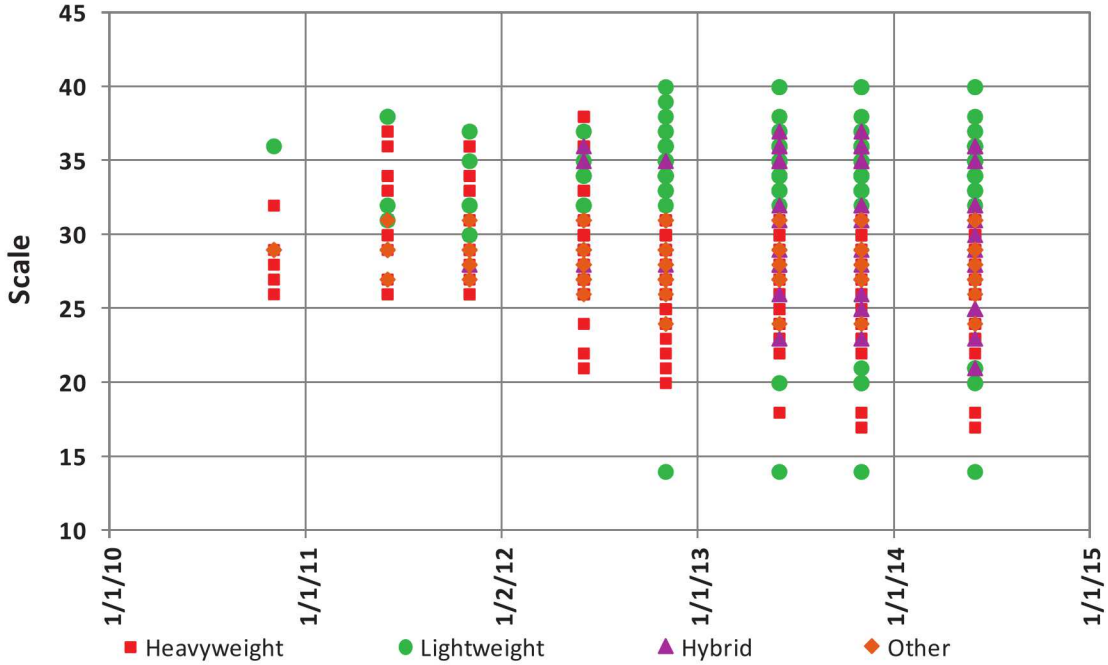


Figure 5.23. GRAPH500 Scale vs Time.

5.5.3 Problem Size and Memory Usage

The size of a problem is more important to GRAPH500 than to TOP500. Fig. 5.23 diagrams the scale of problem achieved versus time. There is no growth in these numbers since 2012, with the only real observation is that once again it appears the biggest problems, of scale 40 (1 trillion vertices), are being solved on lightweight Blue Gene systems.

As a corollary, Fig. 5.24 diagrams the scale of the problem solved versus the aggregate amount of memory reported available in the system. The trend line is when the memory size is 2 to the scale achieved, which means that problems are linear in the number of vertices, and reported implementations used virtually all available memory. This seems to be the case for all but the smallest systems.

Finally, Fig. 5.25 diagrams the performance achieved versus the scale problem attempted. Again there is almost linear peak correlation, which makes sense given that the only way to grow the problem size is to grow the number of nodes, and that as shown earlier. More nodes translates into more performance.

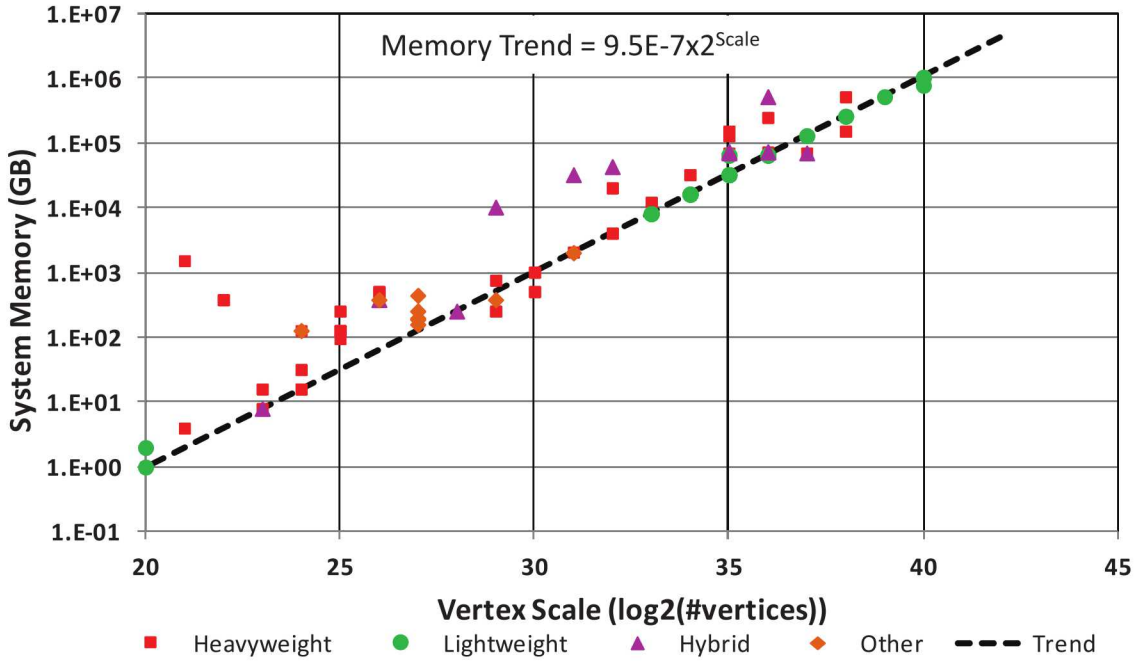


Figure 5.24. System Memory versus Problem Scale.

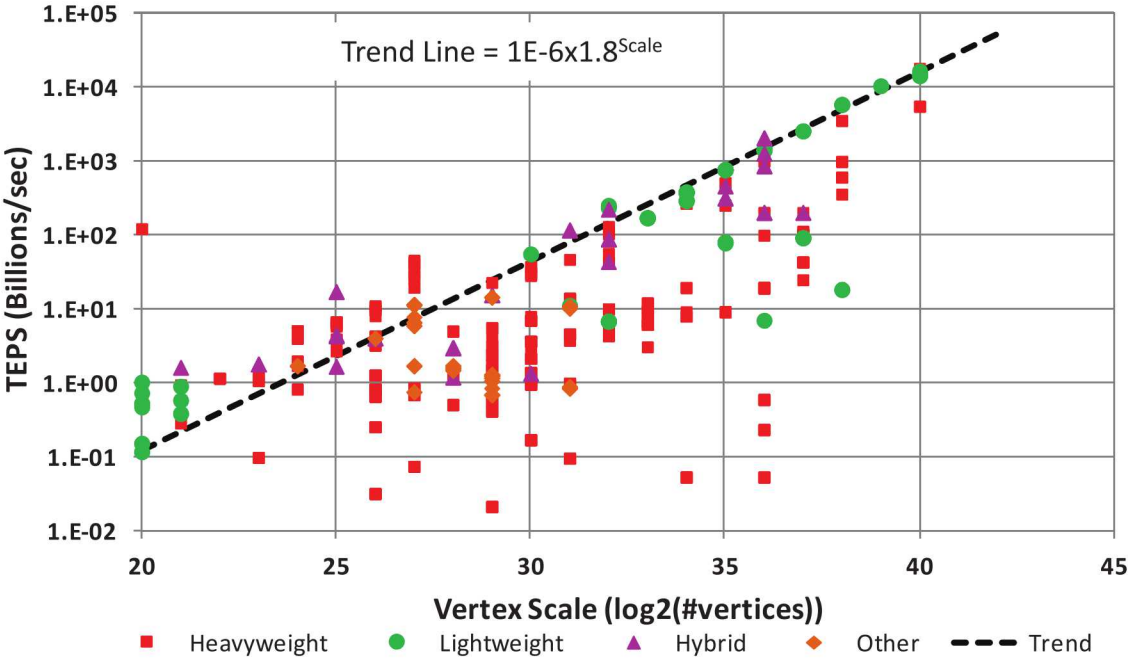


Figure 5.25. TEPS versus Problem Size.

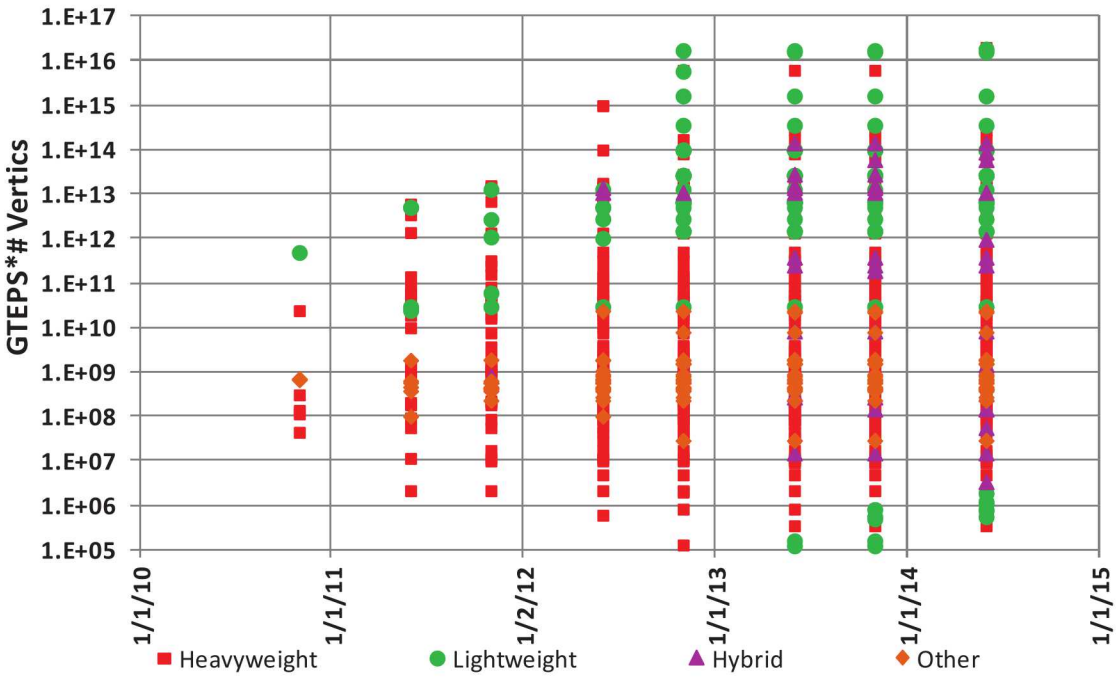


Figure 5.26. A TEPS*Size Metric.

5.5.4 An Alternative Metric

Given that GRAPH500 has two key characteristics, TEPS and problem size, rather than just the flops/s for TOPs500, metrics that combine the two are of interest. Fig. 5.26 multiplies achieved TEPS by the number of vertices (2 to the scale).

Although the curve flattened in June 2013, there has been a greater than 1000X growth in this metric before this point, with the lightweight systems dominating.

5.5.5 Improvement in Algorithms - Looking at BlueGene

The initial 62X growth per year from Fig. 5.17 cannot be explained solely by growth in the number of nodes (only a factor of 8 since the beginning). The rest must be either improvements in node design and/or algorithm improvements. To study these terms we look at the BlueGene points, since that represents two versions of the same node design (BlueGene P and Q), with many data points over time programmed by different teams. Table 5.1 summarizes the substantial differences between the two designs.

Fig. 5.27 diagrams the per node TEPS rate of both the P and Q versions over time. It appears that the P version stabilized very quickly, but the Q version improved by over an order of magnitude over its two years before it too flattened. Further, the per node performance of Q is close to 100X

Parameter	L	P	Q	Change P to Q
Cores/node	2	4	16	4X
Core Clock (GHz)	0.7	0.85	1.6	1.9X
Max Node Memory (GB)	1	4	16	4X
Memory Ports per Node	1	2	2	same
Memory Bandwidth per Port	5.6	6.8	21.35	3.1X
Total Memory Bandwidth (GB/s)	5.6	13.6	42.7	3.1X
Inter-node Topology	3D Torus	3D Torus	5D Torus	
Links per Node	12	12	22	4.7X
Bandwidth per Link (GB/s)	0.175	0.425	2	4.7X
Total Link Bandwidth(GB/s)	2.1	5.1	44	8.6

Table 5.1. BlueGene Characteristics.

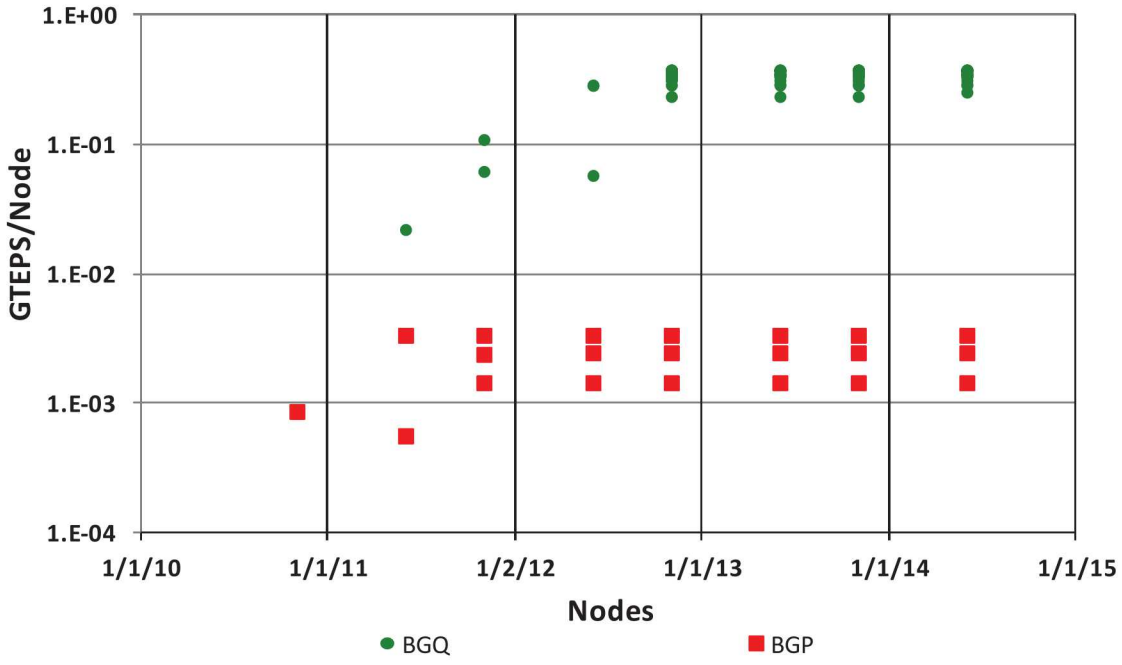


Figure 5.27. BlueGene Performance versus Time.

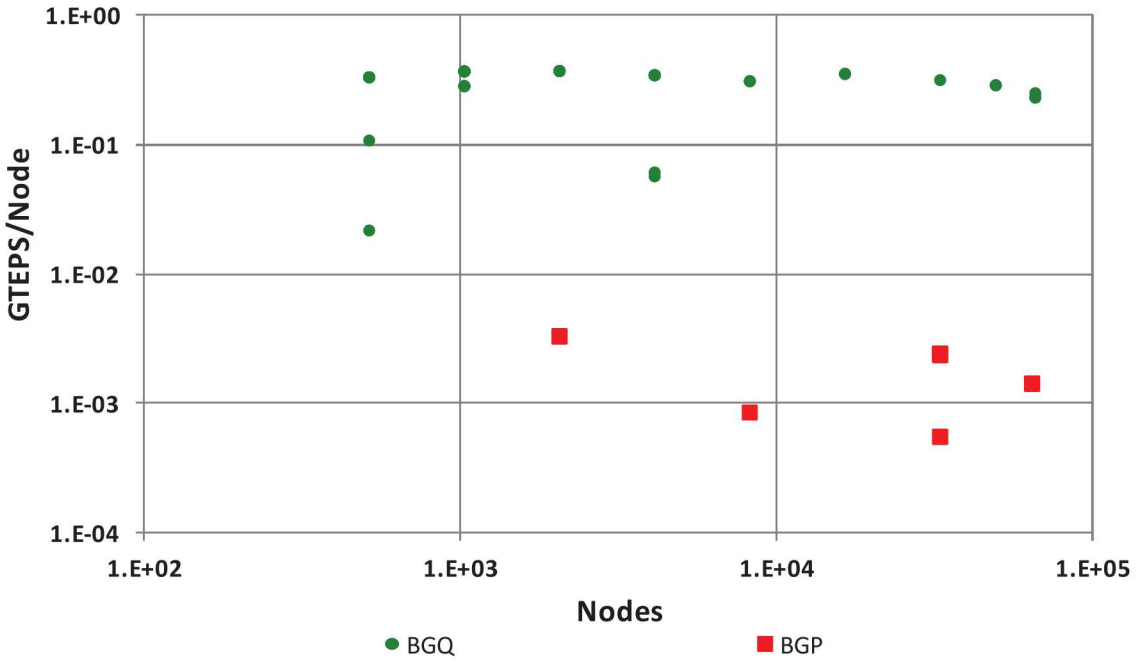


Figure 5.28. BlueGene Performance versus the Number of Nodes.

that of P. This is clearly the major contributor to performance gain over time.

Fig. 5.28 plots the same data versus the number of nodes in the system benchmarked. As discussed earlier, again there is some loss of performance per node at the larger system sizes, but Q systems seem to be somewhat less sensitive. This is most probably because BlueGene nodes employ a toroidal topology, so that as the number of nodes grow, the through node traffic increases, reducing the ability of each node to inject traffic of its own.

5.6 Historical Results: High Performance Conjugate Gradient

The new HPCG benchmark (Section 2.4) is intended to give more insight into the performance of modern compute-oriented applications than LINPACK. As of Fall 2014, only one preliminary results listing has been published[4]. While not enough to start a historical trend, and limited in detailed data about configurations, it is possible to compare the floating point efficiency of HGCG on a handful of systems that have matching numbers on the TOP500 lists. "Efficiency" here is the reported floating point count divided by R_{peak}. Fig. 5.29 diagrams this comparison for five systems that are both in the top 10 of teh TOP500 and have measurements in this one ranking.

[4]https://software.sandia.gov/hpcg/

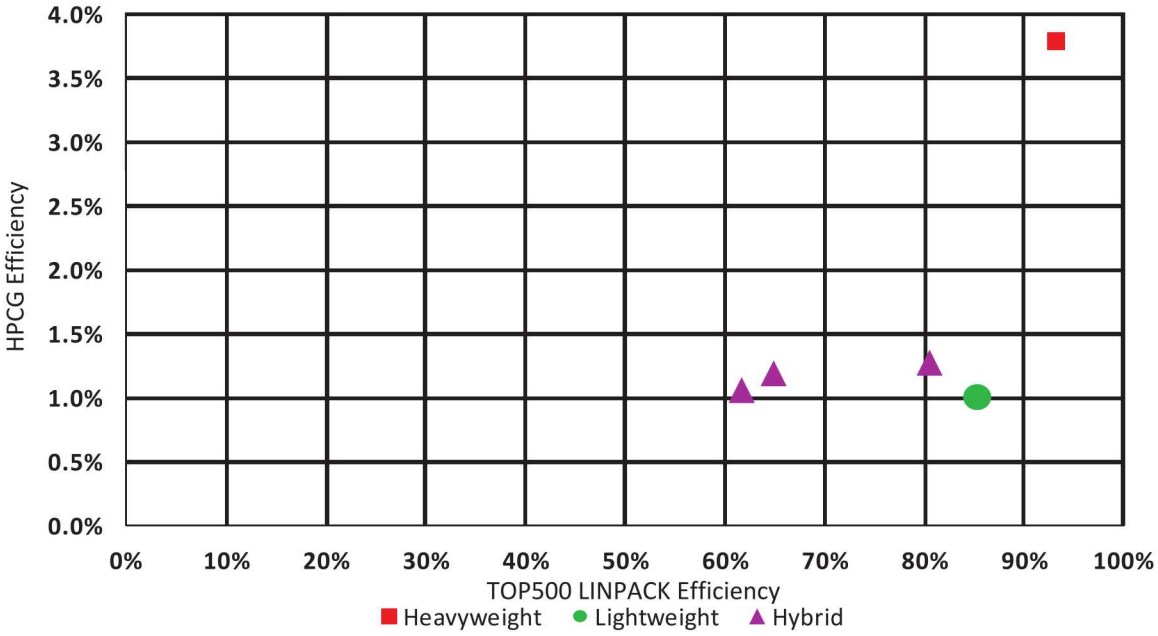


Figure 5.29. Efficiency Comparison.

The three hybrid systems and the single lightweight have efficiencies for HPCG of a measly 1%, versus 60-80% for the dense LINPACK codes. The single heavyweight reaches a whopping 4% on HPCG versus over 90% on LINPACK. The implications are significant; it is clear that the sparse nature of HPCG places so much more strain on the processors' memory system. This should explain the heavyweight results, which has far higher memory bandwidth than the others.

5.7 Historical Results: DARPA HPCS Suite

The HPCS suite (Section 2.3), not to be confused with the HPCG benchmark of the prior section, do not have the structured and repetitive temporal measurements as any of the prior benchmarks, but an analysis of their reports is still instructive.

We note that the measurements reported are in terms of the number of "processes," not "nodes," "sockets," or "cores." Looking at the data, with some exceptions, the terms "processes" and "nodes" seem close enough to use interchangeably. The cases where there were obvious mismatches in terminology were modified to fix these discrepancies.

We look first at the different benchmarks in terms of performance per process or node (in teraflops/sec) as a function of time, as pictured in Figs. 5.30 through 5.33. With the exception of the GUPS curve, there is no discernable growth trend, with only GUPS increasing significantly until about 2010, where the numbers flatten out like the others.

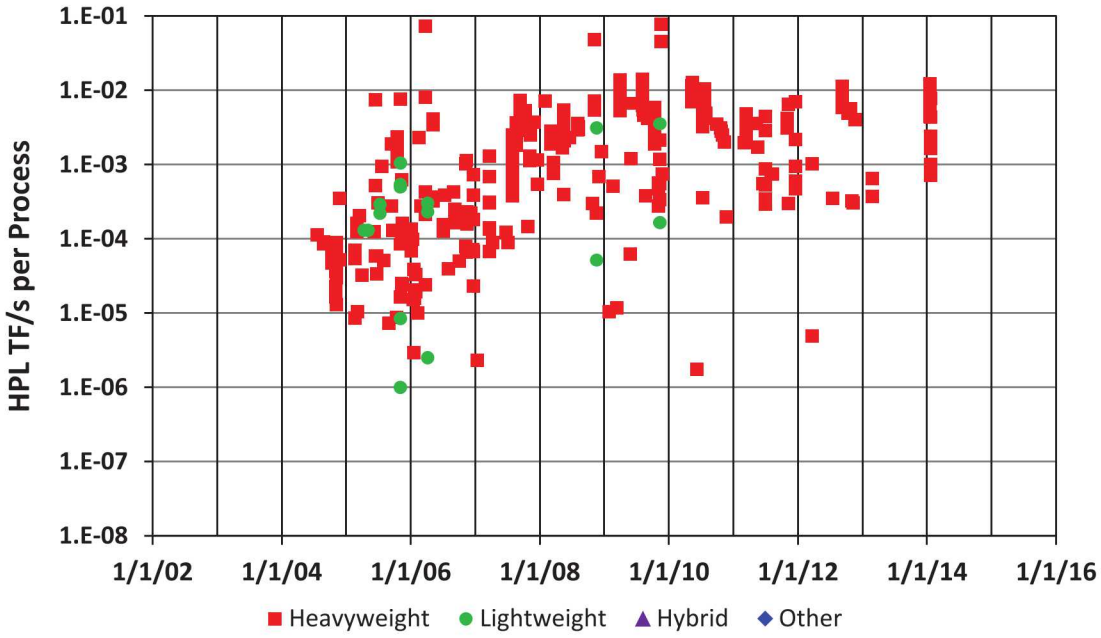


Figure 5.30. HPCS: HPL.

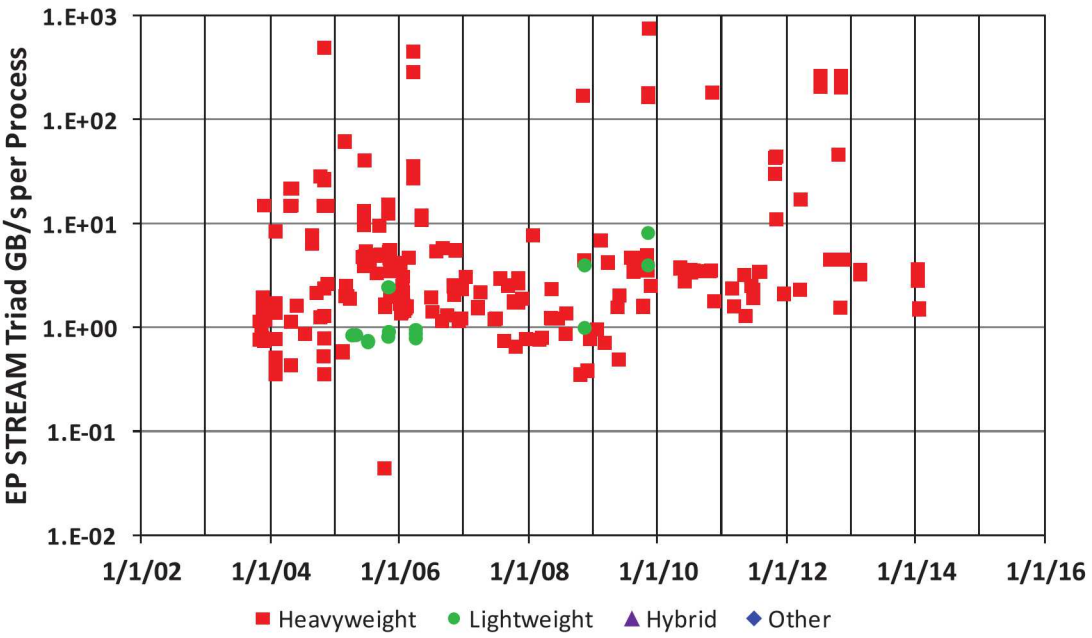


Figure 5.31. HPCS: STREAM.

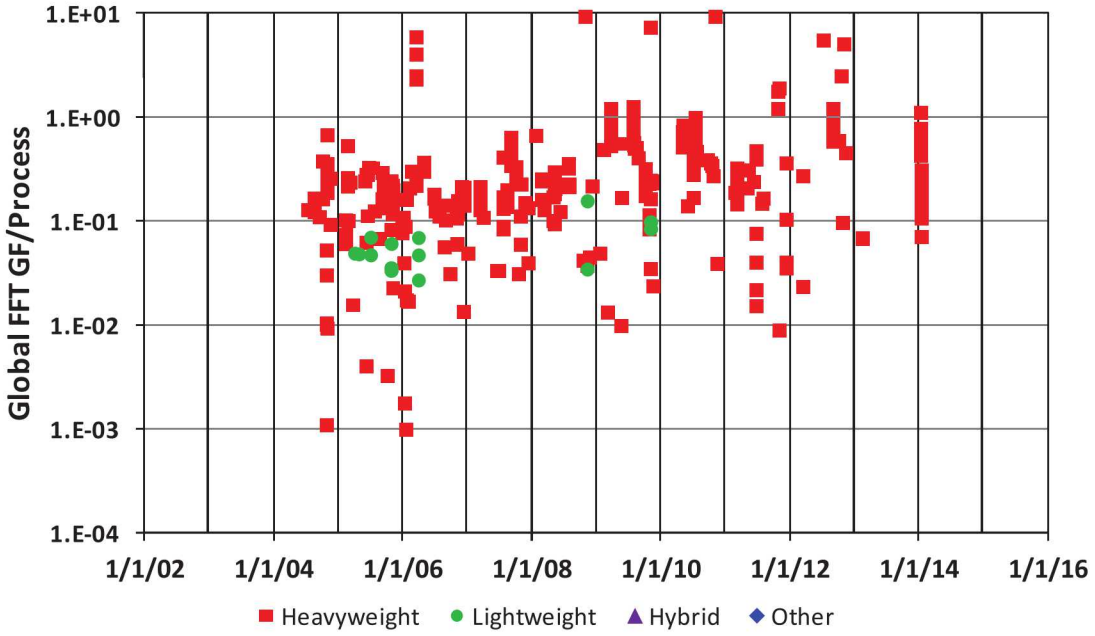


Figure 5.32. HPCS: FFT.

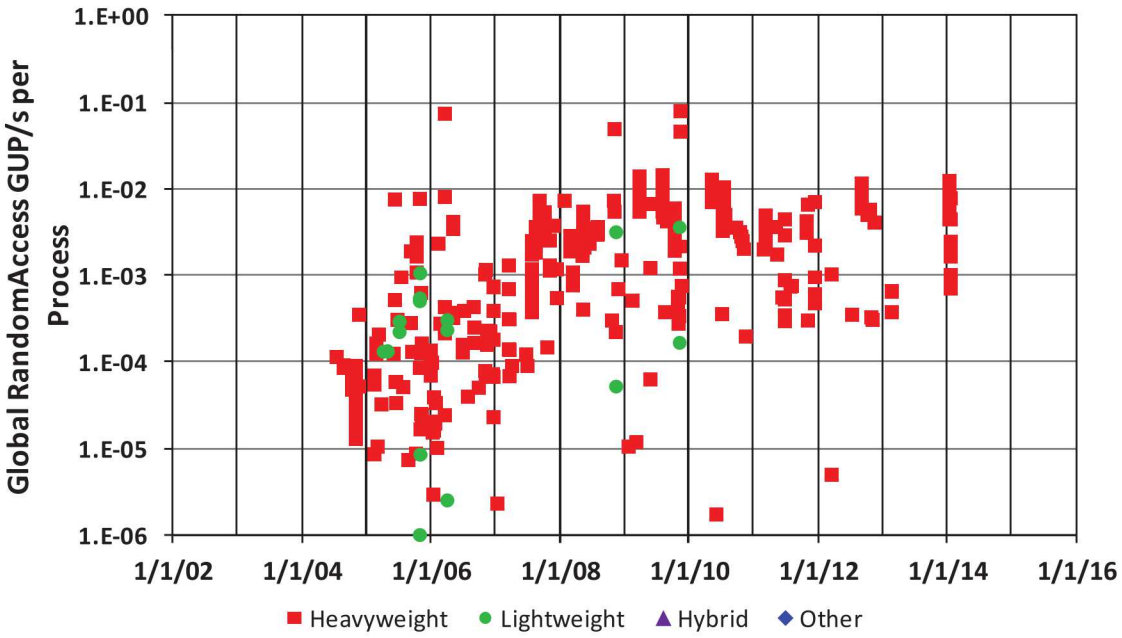


Figure 5.33. HPCS: GUPS.

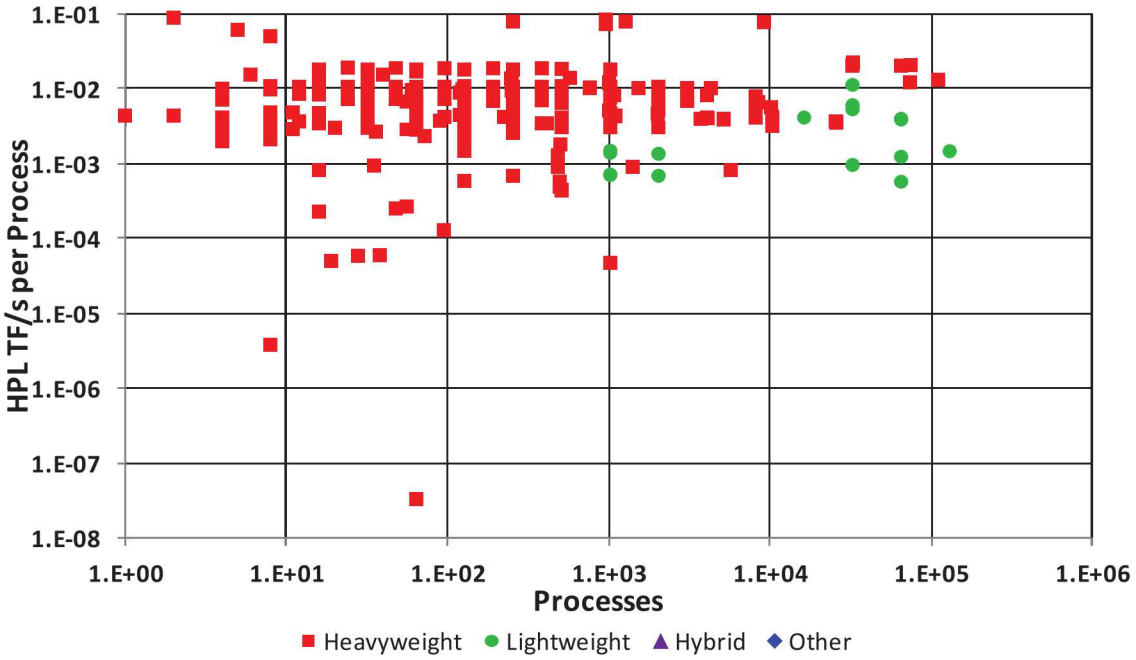


Figure 5.34. HPCS: HPL as a function of Node Count.

Figs. 5.34 through 5.37 graph the same data but as a function of the number of nodes (processes). HPL and STREAM are heavily memory-bandwidth oriented, so perhaps not surprisingly, there is little degradation in per node performance as the number of nodes increases. FFT and GUPS, however, show a definite decrease in performance as a function of node count. These applications are very communication-sensitive, and as the number of nodes increases, more of the communication bandwidth is absorbed by through-the-node routing versus traffic sourced at the node.

Fig. 5.38 graphs HPL performance versus the STREAM performance for the same system. This is an important possible relationship since STREAM is largely memory bandwidth driven, and HPL is a dense floating point driven metric. As can be seen, there is a distinct correlation between the two, meaning that systems that have more memory bandwidth tend to have better floating point performance.

Fig. 5.39 draws a similar picture, but with the floating point of HPL versus random communication bandwidth as demonstrated by GUPS. Unlike versus STREAMs, here there appears to be little correlation, showing again that HPL (and thus TOP500) is not a good benchmark to stress inter-node communication bandwidth when the pattern is essentially random.

Fig. 5.40 again repeats this process, but with the floating point of HPL versus the more regular communication bandwidth as demonstrated by PTRANS (transpose). Unlike versus GUPS, here there appears to be good correlation, showing that HPL (and thus TOP500) does seem to use regular communication.

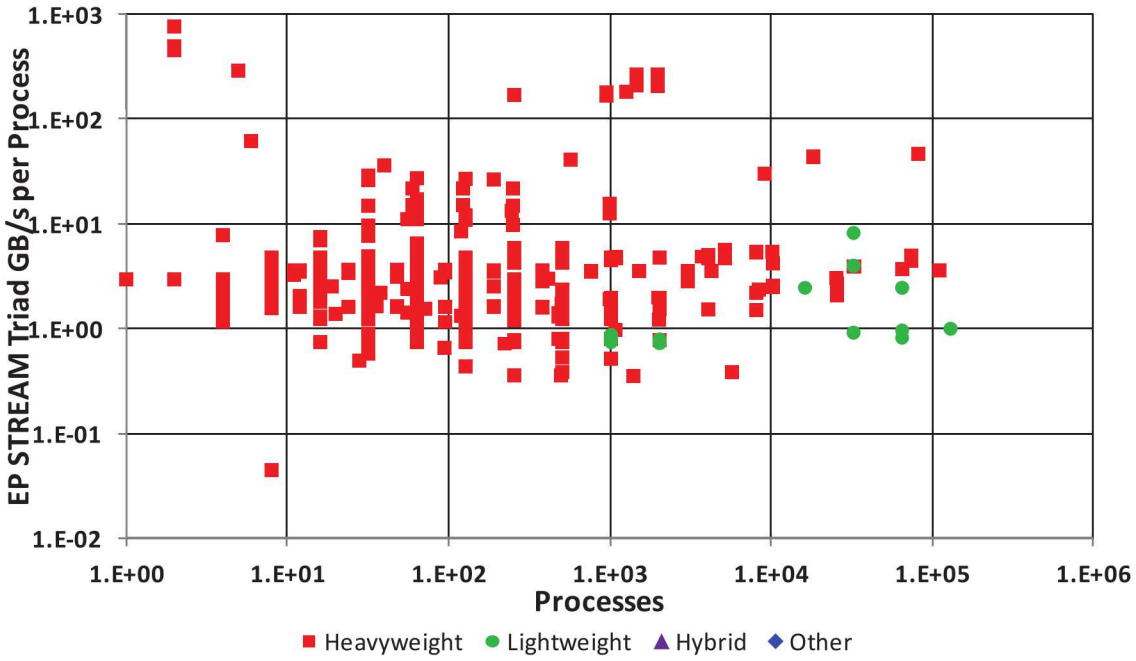


Figure 5.35. HPCS: Stream as a function of Node Count.

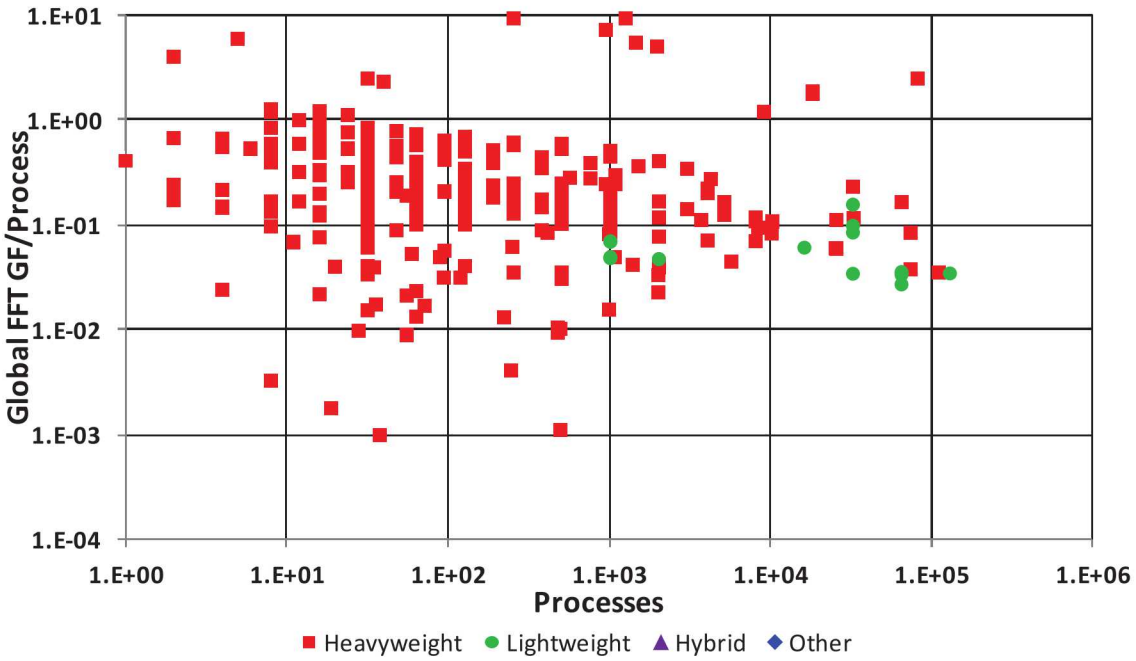


Figure 5.36. HPCS: FFT as a function of Node Count.

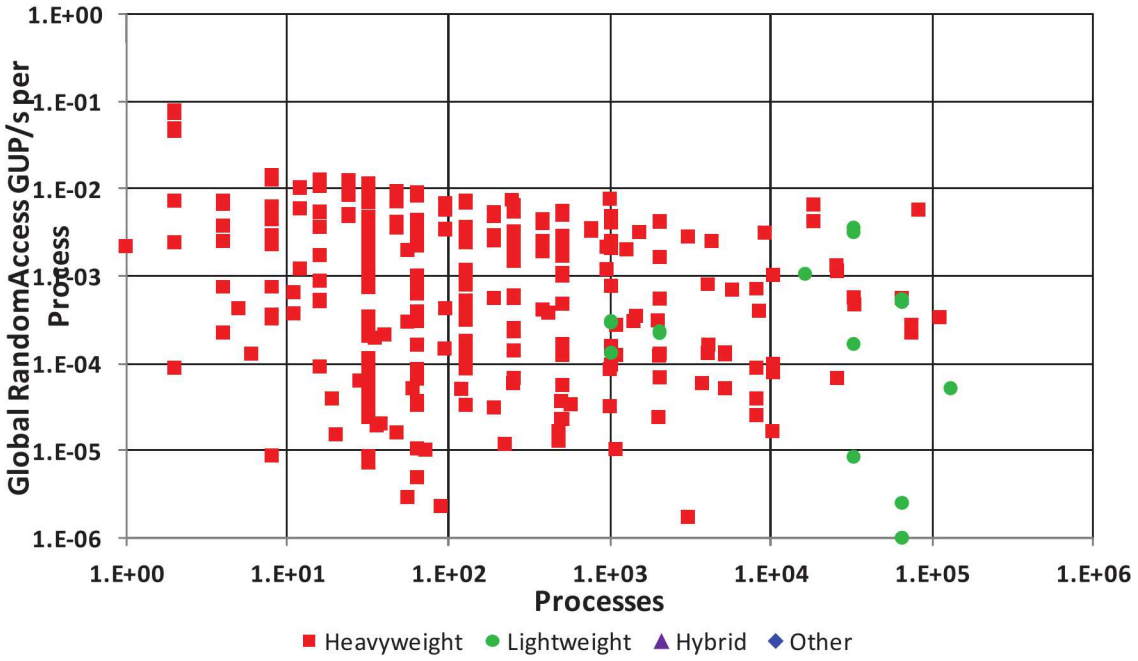


Figure 5.37. HPCS: GUPS as a function of Node Count.

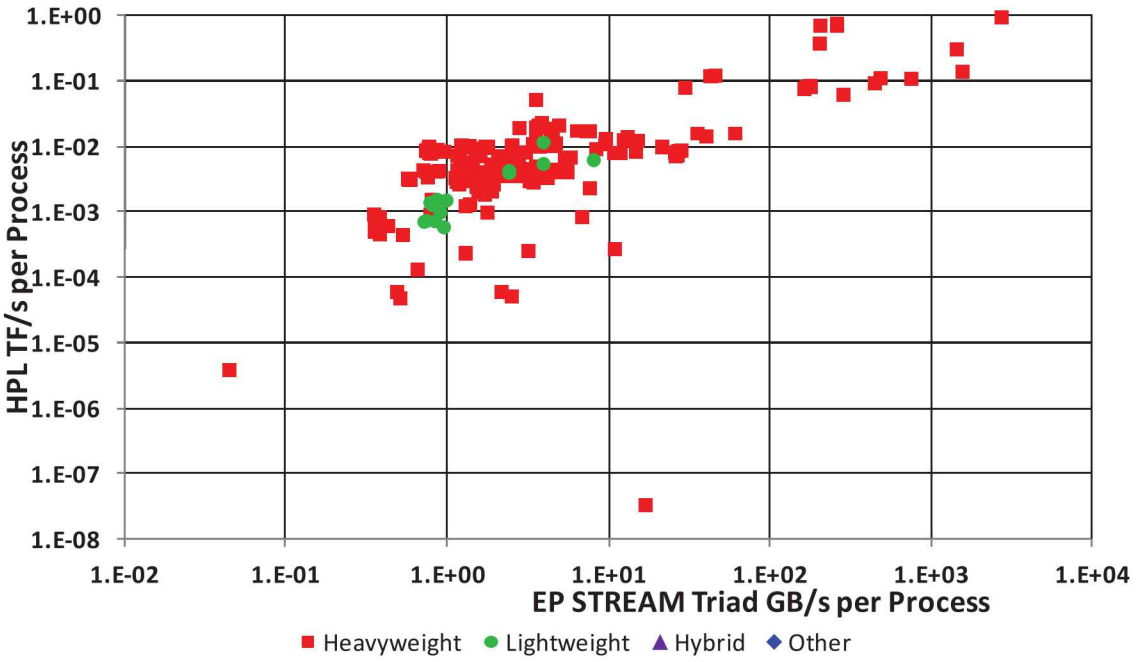


Figure 5.38. HPL versus STREAM: Flops as a Function of Memory Bandwidth.

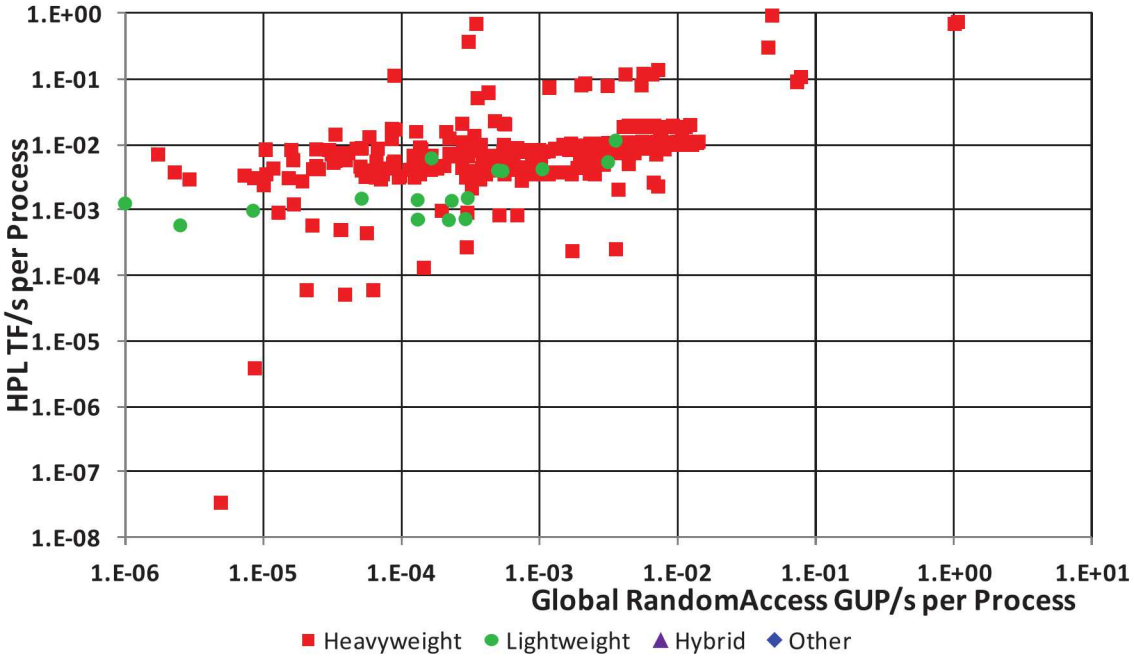


Figure 5.39. HPL versus GUPS: Flops as a Function of Random Patterns of Network Bandwidth.

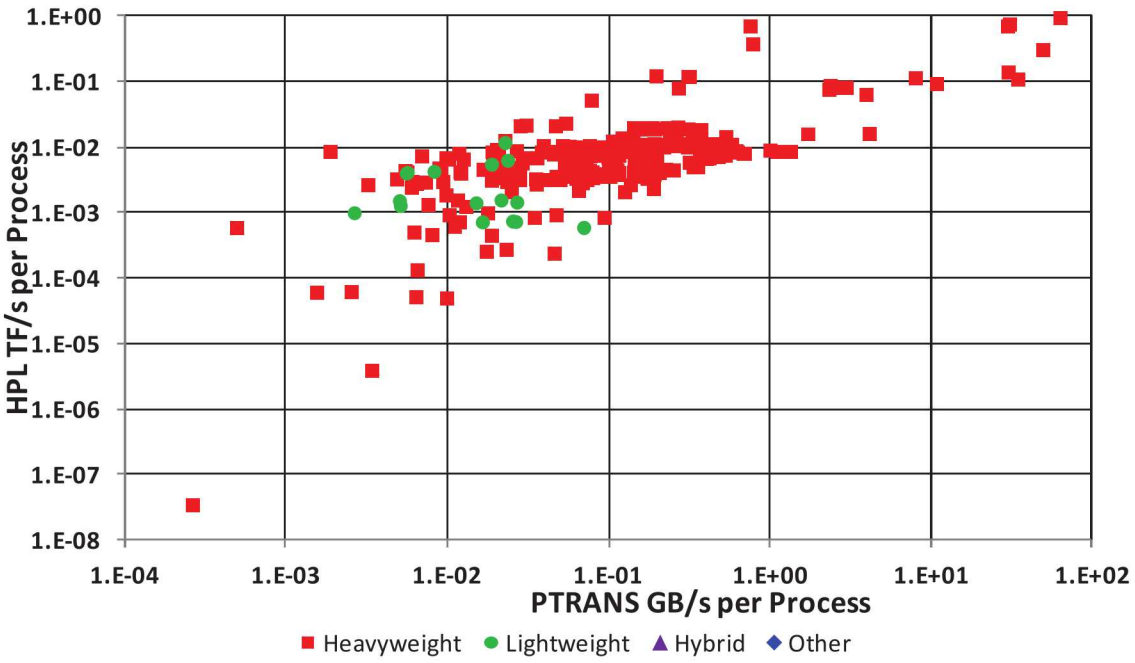


Figure 5.40. HPL versus PRTANS: Flops as a Function of Regular Patterns of Network Bandwidth..

This page intentionally left blank.

Chapter 6

Projections

The 2008 Exascale report[20] performed an analysis of the TOP500 data much as here in Chap. 5, but with 5 years less data. In addition, and more importantly, this study also made some projections about the direction of technology (again much as in Chap. 3), and then made assumptions about how TOP500 system would thus evolve in the future. These assumptions were then propagated forward and compared with what was at the time to goals of the UHPC project (an R_{peak} of 10^{18} flops/sec) at 20MW. At that time, the two architectures that were extrapolated forward were the heavyweight and lightweight. In addition, while 3D stacks were assumed as an important part of the strawman architecture suggested in the report, details and projections were at best limited, in that there was no implementation prototypes to suggest optimal design choices.

More data from both the TOP500 and the ITRS were available in 2011[19] and early 2012[18], and the hybrid architecture had emerged. These were factored into the model from 2008, and projections made going forward.

Section 6.1 chapter reviews the assumptions from 2008, and displays the results of that model updated with all the intervening data, including a partial model for hybrid architectures. This projection is to demonstrate the basic validity of the modeling process.

A near-term update to this model will reflect newer data, a revised set of assumptions, and initial projections for 3D.

6.1 2008 Model

The 2008 Exascale report[20], and then [19], made some assumptions about how systems based on the heavyweight, lightweight, and (partially) hybrid architectures would evolve through time. The following subsections review these assumptions, project what that model predicted, and then analyze how well the model did. All data up through the 2012 TOP500 lists were used.

	Units	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
Transistors/Core	MT	341	341	341	341	341	341	341	341	341	341
Die Area/Socket	mm2	220	220	220	220	220	220	220	220	220	220
Cores/Socket		2	2	3	4	5	9	13	19	27	38
Vdd	V	1.20	1.10	1.10	1.10	1.10	1.00	0.97	0.90	0.87	0.85
Flops/cycle/Core		2	2	4	4	4	4	4	4	4	8
Chip to system	Years	2	2	2	1	1	0	0	0	0	0
Growth in Mem/Socket				1.0	1.4	3.3	3.8	4.5	6.9	9.0	13.7
Compute Sockets/Board		4	4	4	4	4	4	4	8	8	8
Boards/Rack		24	24	24	24	24	24	32	32	32	32
Max Power/Rack	KW	16	16	16	16	16	32	32	32	65	65
Max Racks/System		155	155	155	200	250	300	350	400	450	500

	Units	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026
Transistors/Core	MT	341	341	341	341	341	341	341	341	341	341	341	341	341
Die Area/Socket	mm2	220	220	220	220	220	220	220	220	220	220	220	220	220
Cores/Socket		48	60	76	96	121	152	192	242	305	385	485	611	770
Vdd	V	0.82	0.8	0.77	0.75	0.73	0.71	0.68	0.66	0.64	0.62	0.61	0.59	0.57
Flops/cycle/Core		8	8	8	8	8	8	8	8	8	8	8	8	8
Chip to system	Years	0	0	0	0	0	0	0	0	0	0	0	0	0
Growth in Mem/Socket		17	40	49	56	61	72	89	103	124	148	175	211	245
Compute Sockets/Board		8	16	16	16	16	16	16	16	16	16	16	16	16
Boards/Rack		32	32	32	32	32	32	32	32	32	32	32	32	32
Max Power/Rack	KW	65	129	129	129	258	258	258	258	258	258	258	258	258
Max Racks/System		550	600	600	600	600	600	600	600	600	600	600	600	600

Figure 6.1. Assumptions from the 2008 Model.

6.1.1 Heavyweight Scaling Assumptions

The assumptions made in the Exascale report[20] about how the heavyweight architectures would mature over time drew from a 2004 baseline of the processor used in the 2006 Red Storm: the 2004 90nm dual core Opteron, 95W, 1.35V, 2.4GHz, 220mm^2, with a later upgrade that took the same chip to 2.6GHz. These assumptions, listed below, are for the most part drawn from that report. Fig. 6.1 lists these assumptions by year. Unless otherwise noted, these assumptions also apply to the non-heavyweight models as well.

We note that some of these assumptions are liable to be difficult or expensive to achieve in practice, such as a 300 KW rack, but are included here to permit bounding the possible alternatives. In particular, combinations of these assumptions, such as 600 separate 300KW racks in particular, may be theoretically possible but are liable to be jointly infeasible.

1. The microarchitecture for each core in the microprocessor will be approximately unchanged in complexity (i.e. transistor count) over the baseline, with the area consumed at any point in time approximated as proportional to the reciprocal of the transistor density as taken from

the ITRS. This count included amortized caches, memory controllers, and overhead logic, but did not consider effects of increasing threading, better efficiency, latency tolerance, or increased fault resilience. The 90nm Opteron thus had an amortized area of $110mm^2$ per core, and thus, using ITRS projections for density, have about 341M transistors per core, most of which are for caches and other memory structures.

2. The die size for the microprocessor chip will remain approximately constant at the $220mm^2$ of the Opteron, meaning that the number of cores on each microprocessor chip would grow roughly as the transistor density grows.

3. We do not account for relatively larger (in area) L3 caches to reduce pressure on off-chip contacts. This actually results in a possible overestimate of peak chip performance because we end up with more cores per die.

4. Such chips will continue to use high performance logic, with V_{dd} flattening as in Fig. 3.7.

5. The power dissipation per die has flattened, as projected in the ITRS.

6. Per core, the microarchitecture improves from a peak of 2 flops per cycle in 2004 to a peak of 4 flops per cycle in 2006, and 8 flops per cycle in 2013. This is for conventional cores, not GPUs or other accelerators

7. The delay in time from the start of production of a new chip to its inclusion in a new system was 2 years before 2006, would drop to 1 year in 2006, and then drop to the same year in 2009.

8. The memory used remains commodity memory as architected today; the effects of alternatives such as memory stacks will be addressed later.

9. The system would want to maintain the same ratio of bytes of main memory to peak flops as in Red Storm. This will be done by using whatever natural increase in density comes about from commodity DRAM, but assumes additional memory cards as necessary if that intrinsic growth is insufficient. We note that this is in contrast to the downward slope in memory per flop per second in recent TOP500 systems.

10. The maximum number of compute sockets per board will double a few times. This is assumed possible because of a possible move to liquid cooling, for example, where more power can be dissipated and allowing the white space to be used and/or the size of the heat sinks to be reduced. This projections assumes that this may happen at roughly five year intervals to a maximum of 16.

    There will also be an effect on node size (and number of nodes in a rack) because of the inclusion of NV memory as a 1st-level component. This has not, however, as yet been factored in.

11. The maximum number of boards per rack will increase by perhaps 33% once in the 2010 time frame because of assumed improvements in physical packaging and cooling, and reduction in volume for support systems. For this projection we will assume this may happen once.

12. The maximum power per rack will increase by at best a power of 16, to somewhere around 200-300KW. We assume this is a doubling every 3 years.

13. The maximum number of racks in a system was set to 155 in 2006 (to match then current systems), with an increase by 50 every year, up to a maximum of 600.

14. Secondary storage (and its growth) for scratch, file, or archival purposes is ignored. This storage must also undergo significant increases.

The above assumptions are reasonable for estimating power in the microprocessor parts of a system. There is, however, concern for how accurate such scaling rules would be for other parts of the baseline, particularly the memory system, routers, and the data center size and power level. This is because a significant amount of their power comes not from internal processing but from I/O - the transfer of data across chip boundaries. While such increases can be projected for commodity memory, the number of memory ports per socket is something that is changing. Also, both the protocols and the bandwidth of the router chips, and how they tie into the memory and microprocessors in a node may change in a complex fashion.

Scaled Model

To simplify our projections, we thus continue the approach suggested in the Exascale report by adopting two energy models: Scaled and Constant. The *Scaled* model assumes that the power per microprocessor chip grows as the ITRS roadmap has predicted, and that the power for the memory associated with each socket grows only linearly with the number of memory chips needed (i.e. the power per memory chip is "constant"). We also assume that the power associated with both the routers and other common logic remains constant. This is the same as the "Simplistically Scaled" model in the Exascale report. In a real sense, we are assuming here that both memory access *energy* and the *energy* cost of moving a bit, either across a chip boundary or between racks, will *decrease* (or "scale down") with technology advances, at least as fast as the increase in flops, with the total power constant because of a concurrent "increasing" of the flops rate of the microprocessor most probably requires a higher number of references to memory and a higher traffic through the routers. This is clearly optimistic.

Constant Model

In contrast, the *Constant* model assumes the microprocessor power grows as above, but that both the memory and router power scale linearly with the peak flops potential of the multi-core microprocessors. This naively assumes that the total energy expended in these two subsystems is used in accessing and moving data, and that the energy to handle one bit (at whatever the rate) is *constant* through time (i.e. no improvement in I/O energy protocol). This is clearly an over-estimate, and liable to be pessimistic. It is similar to the "Fully Scaled" model from the Exascale report.

Neither model takes into account the effect of only a finite number of signal I/Os available from a microprocessor die, and the power effects of trying to run the available pins at a rate consistent with the I/O demands of the multiple cores on the die.

6.1.2 Lightweight Scaling Assumptions

For consistency, the scaling assumptions used for the lightweight architecture are modified only slightly from those above for the heavyweight, and again are based on those trends used in the Exascale report. In particular, we reuse assumptions 1, 2, 3, 8, 10, and 12 in total. The list below slightly modifies assumptions 4, 6, 7, 9, and 11, in order, from the heavyweight list:

- V_{dd} flattens as for low power, not high performance, logic.
- Per core, the microarchitecture will improve from a peak of 2 flops per cycle to a peak of 8 flops per cycle in two steps.
- The bytes of main memory per flop ratio will match that of BlueGene/L.
- There is one doubling of compute cards per node board.
- The number of racks in 2005 and 2006 are set to the same as observed in real systems in order to validate the mode. After 2006, the model assumes growth in racks as for the heavyweight.

The following assumptions are specific to these systems:

- The power dissipation per chip will be allowed to increase gradually to twice what it is for BlueGene/L.
- The overhead for the rack for power and cooling will be the same percentage as in the Blue Gene/L.
- For this sizing, we will ignore what all this means in terms of system topology.

6.1.3 Hybrid Scaling Assumptions

For the time being, we will focus on heterogeneous systems utilizing GPUs for their accelerated performance. Since the leading chip for each GPU family tends to have a high transistor count, large area, and high power dissipation, the assumptions below are similar in nature to those of the heavyweight case. It is assumed that these accelerators will remain on daughter cards.

- There will be at most a 50% growth in shaders per core, and a doubling of FMAs per shader.

- The conventional microprocessors that provide the non-computational support for each node will evolve as in the heavyweight case.
- The power dissipation of the GPU will grow as for the heavyweight processor.
- Local memory for individual SMs will double every few years, and the L2 and above caches will grow so as to use the same amount of space they do now.
- The number of memory ports will be pin limited as for the heavyweight, and thus will change at best modestly.
- The directly attached memory to each GPU will grow in density in step with the ITRS predictions, and no faster.
- The ratio of GPU sockets to heavyweight sockets per node will remain the same as today.
- The density of nodes per rack will increase at the same rate as for heavyweight, and with the same power limitations.
- The number of racks in a full system starts with the number in today's biggest system (to validate model) and increases by 50 per year as in the other models, up to 600 at most.

Given that details of energy expenditure in different parts of a GPU chip are not yet available, only one model is projected here: the Scaled model.

6.2 2008-Based Projections

Fig. 6.2 diagrams the projections for R_{peak}, including more historical data from 2008 through 2012. The actual heavyweight points between 2008 and 2010 lay nicely between the scaled and constant model, and then fell very close to the scaled line after 2010. The lightweight points fell below either scaled or constant, primarily because real systems did not include as many racks as were projected until 2012. The hybrid numbers trended much as the scaled model projected.

Fig. 6.3 diagrams the projected power per system. After 2008, the projected power for the heavyweight and hybrid systems tracked reasonably with the projections, with the lightweight systems considerably less than their estimate, again because of fewer racks than in the model.

Fig. 6.4 diagrams the energy per flop (using R_{peak}. In this case the number of racks is irrelevant, and all the 2008-2012 points fits nicely within their projected bounds. Fig. 6.5 shows the same data but blown up to the same scale to focus between 2004 and 2016.

In both cases, the blue curve is a line that represents what simply reducing energy by technology advances would buy. This is CV^2 using the ITRS projections alone. As commented on in the 2008 report, this curve nicely matches the best of the historical data until about 2004, when the transition to energy-efficient designs, especially lightweight and hybrid architectures cam into play. The

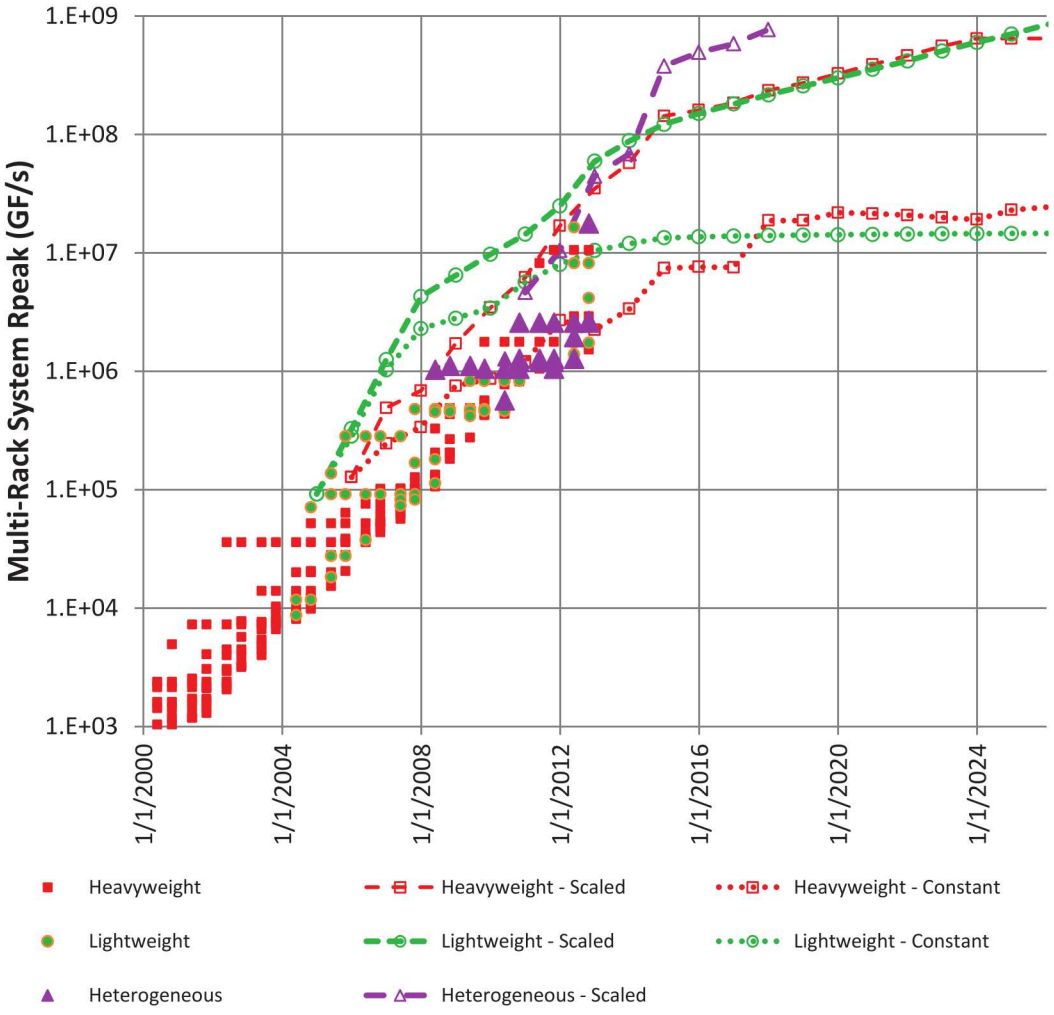


Figure 6.2. R_{peak} Projections - 2008 Model.

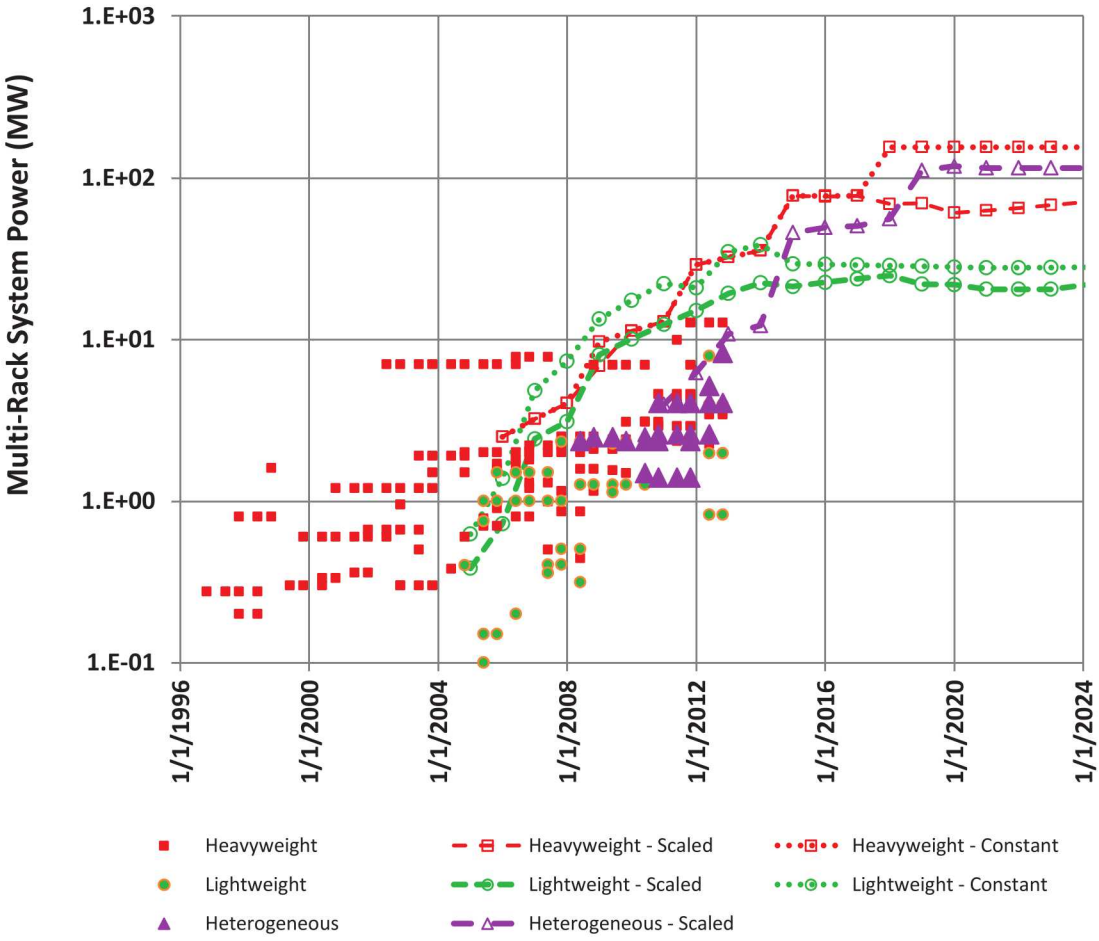


Figure 6.3. Power Projections - 2008 Model.

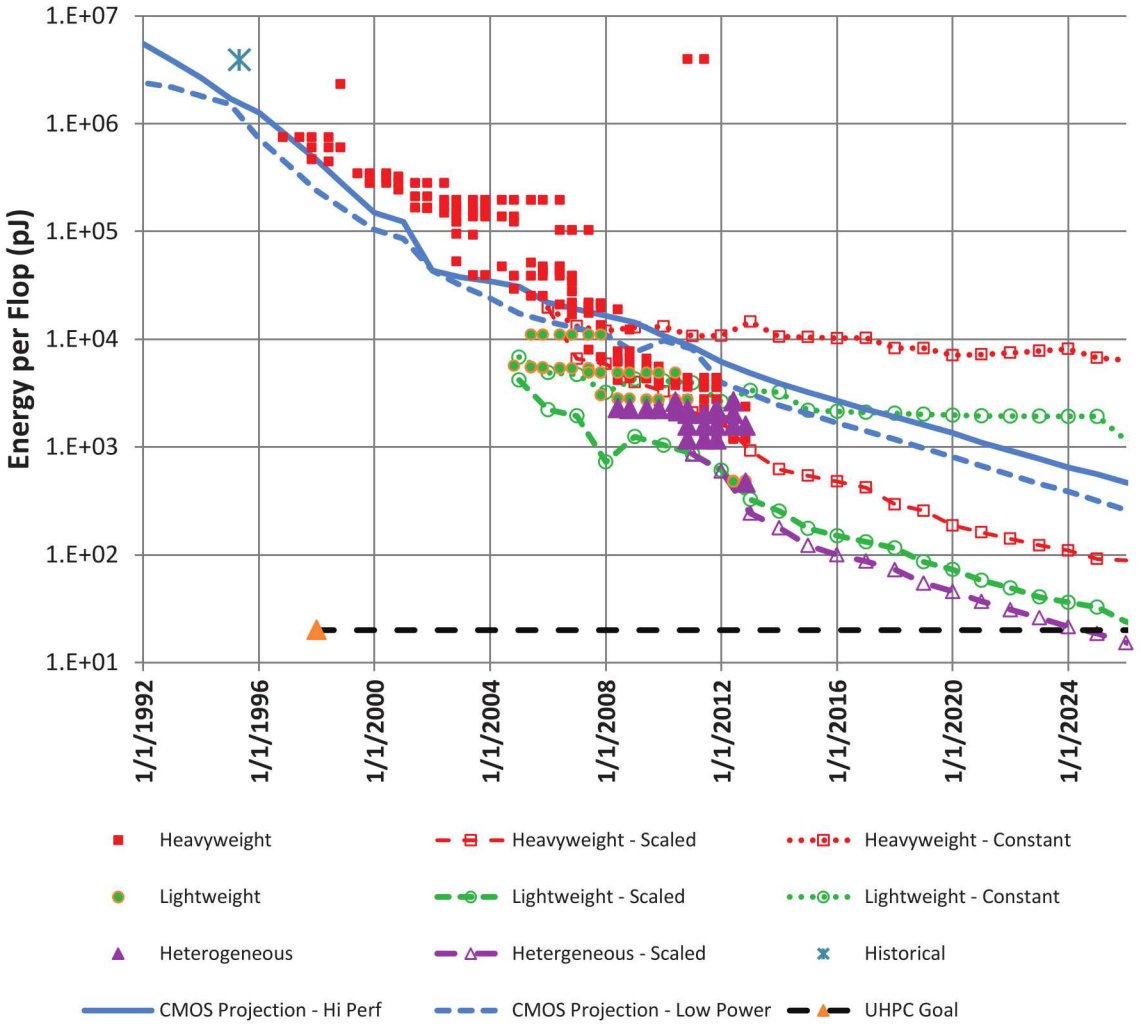


Figure 6.4. Energy per Flop - 2008 model.

take-away is that before 2004 virtually all energy efficiency improvements came from technology alone. After 2004, the introduction of lightweight and hybrid systems did lower the energy curve, but it appears that once lowered, later data suggests we are back to tracking technology. One consideration here is that all of these post 2004 system have less relative memory than 2004, meaning that some of the energy savings may come from that. This is an area worthy of further study.

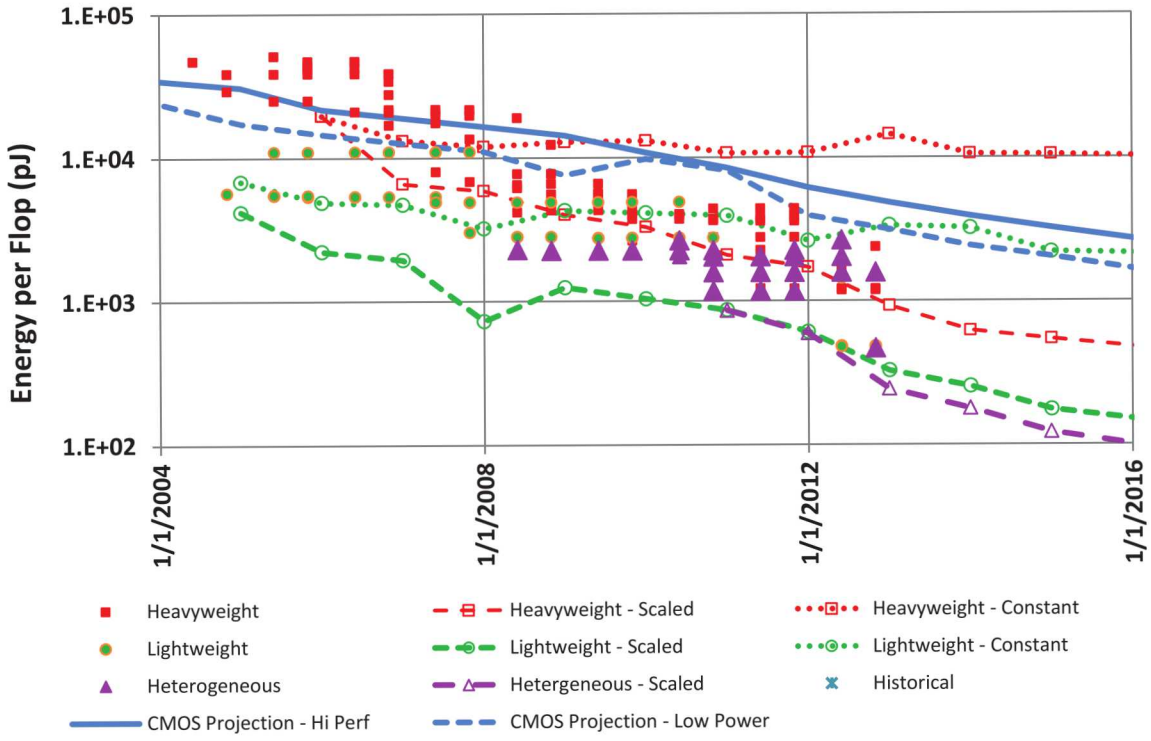


Figure 6.5. Nearterm Energy per Flop - 2008 model.

6.3 3D Stack Projections

This section projects how 3D hybrid stacks as discussed in Section 3.5 may evolve, both when used just as memory subsystems and when additional processing functionality is added to the logic die.

6.3.1 Stack Projections Through Time

Figs. 6.6 and 6.7 use the 2015 numbers suggested in [30] to develop estimates of what hybrid stacks in later years may provide in density and dissipate in power. Numbers are given for 4, 8, and 16-high stacks of DRAM die on top of a logic die. Assumptions made in these estimates include:

- Features sizes in the out years are assuming the ITRS roadmap numbers for both DRAM and logic.
- The density numbers for DRAM start with the 2015 projection from [30] (at 8Gb/die this agrees with the ITRS projections in that time frame), and go forward using the same ITRS projections.
- There is an assumed 2X increase in 2015 in the number of vaults (from [30]) and another 2X increase in 2018 to a total of 64. At best small increases are assumed after this on the basis that TSV density peaks out at about this time frame.
- Each vault requires a memory controller on the logic die, so the number of memory controllers increases linearly with the number of vaults.
- Increasing the number of memory controllers increases the complexity of the on-logic die router (each controller increases the number of ports). We assume that the logic (and thus power) for routers goes up as the square of the increase in vaults (probably a bit too pessimistic).
- The aggregate off-chip signalling rate doubles in 2015 from the prototype, and goes up again by another 50% in 2018, and again in 2020.
- [30] indicates that power is split roughly 1/3 for the memory arrays, 1/3 for non-I/O logic die functions, and 1/3 for off-chip I/O. We thus baseline the original power breakdown on the current HMC in this way. We also assume that the current logic power is split 50-50 between router and memory controllers.
- In computing logic die power, we scale by the square of the ratio of voltage decreases (using the ITRS projections for high performance chips), by the ratio of feature sizes (reflecting the decrease in capacitance as feature sizes decrease), and by the change in the number of ports to the on-chip router (whose power is assumed to grow as P*log(P) where P is the number of ports).

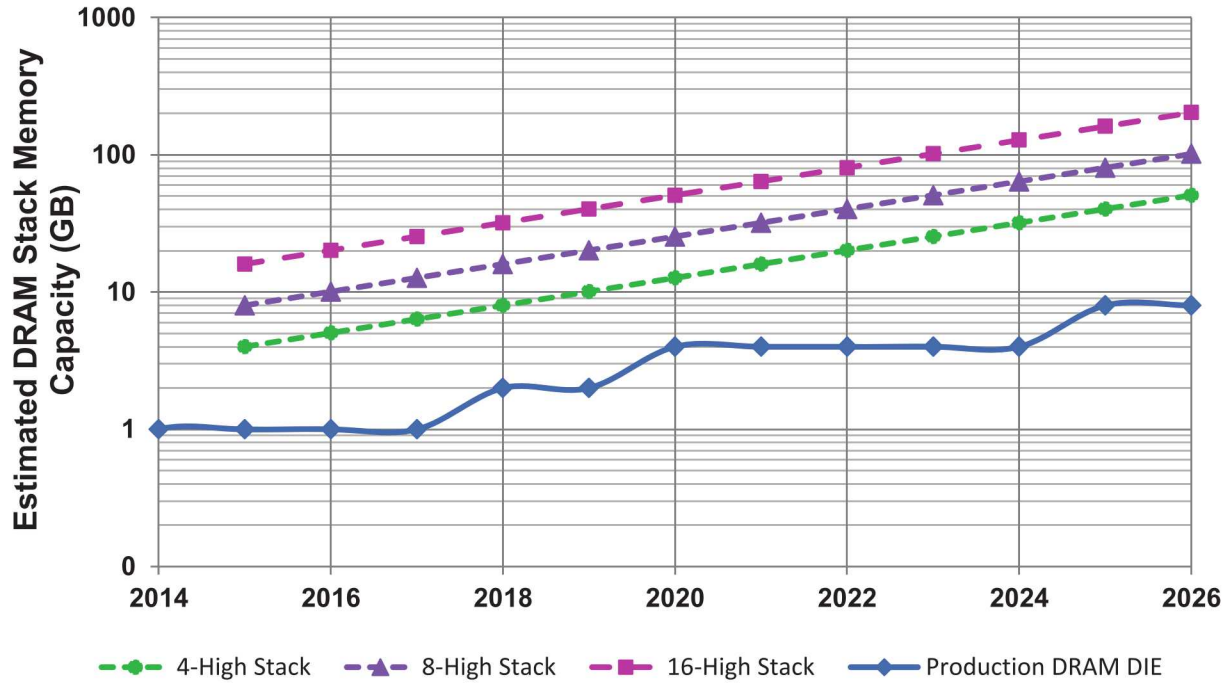


Figure 6.6. Projecting Possible Stack Density.

- In computing memory array power, we assume the power scales as the number of vaults (more banks being accessed), but that the power per bank is constant other than a decrease due to reductions in DRAM core V_{dd}, and to a 20% factor that [30] mentions is likely to happen due to DRAM cell rearchitecting.
- Growing the number of die in a stack (such as from 4 to 8 DRAM die) will not increase the number of banks busy in any one vault, and thus will not significantly increase the vault power once the vault's TSV channels have saturated. The only extra power from the other memory die is refresh power.
- I/O power scales with the total off-chip signalling rate, derated by the relative reduction in logic chip V_{dd}.

It is important to note that the power numbers here assume that all resources, vaults, routers, memory controllers, and I/O, are running at 100%.

Fig. 6.8 summarizes the projected growth of ports within a hybrid stack. Fig. 6.9 projects the aggregate bandwidths.

We also note that the logic density for a logic chip in 2015 is can be as much as 24 times that of the 90nm prototype, which should be more than adequate for the assumed increase in memory controllers and router logic. In 2018 the density is 48X that of the 90nm chip, again more than sufficient for the additional memory controllers and routers.

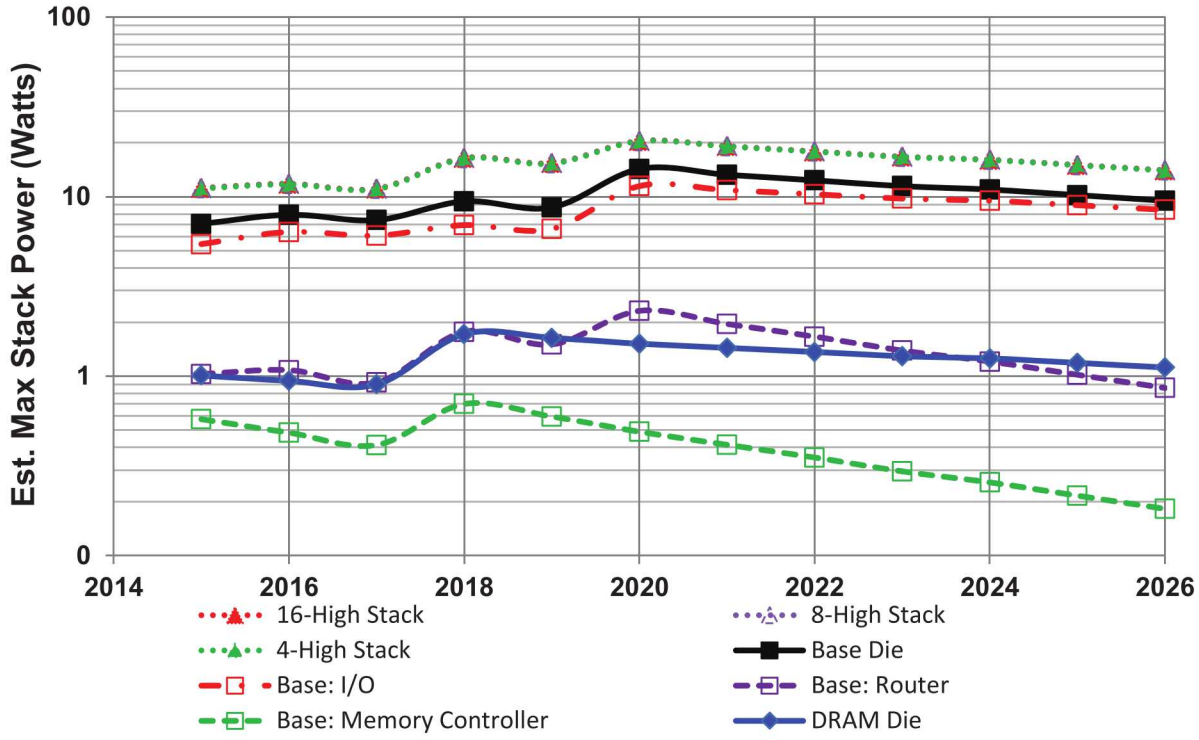


Figure 6.7. Projecting Possible Stack Power.

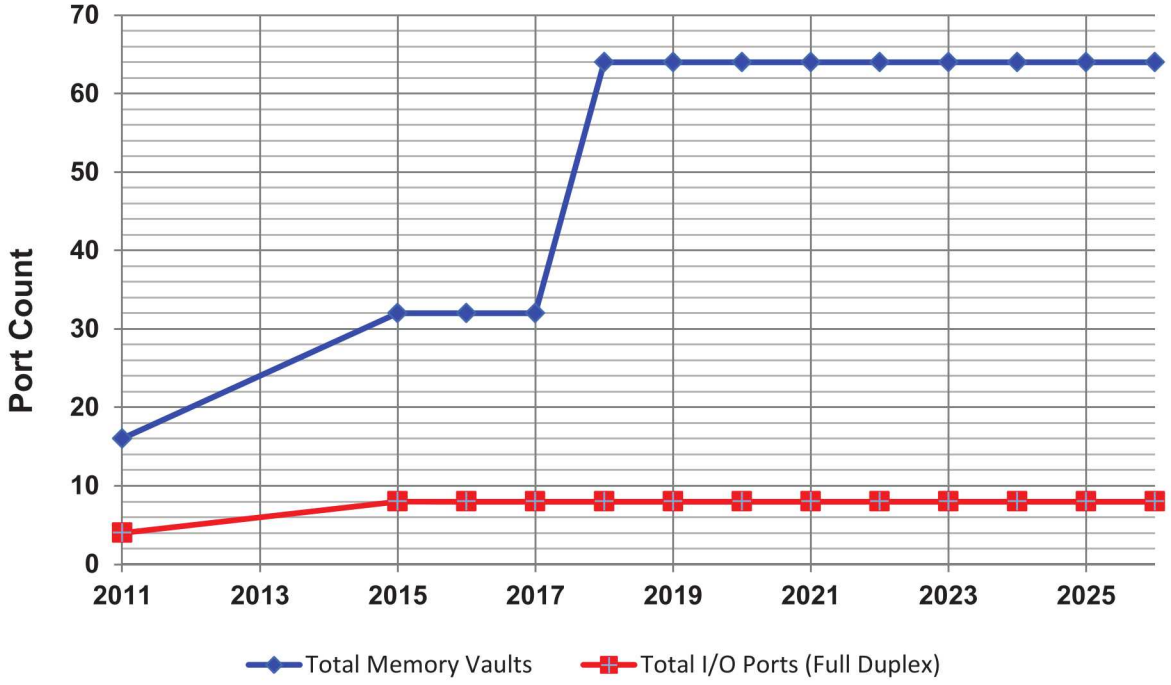


Figure 6.8. Port Counts for Hybrid Stacks.

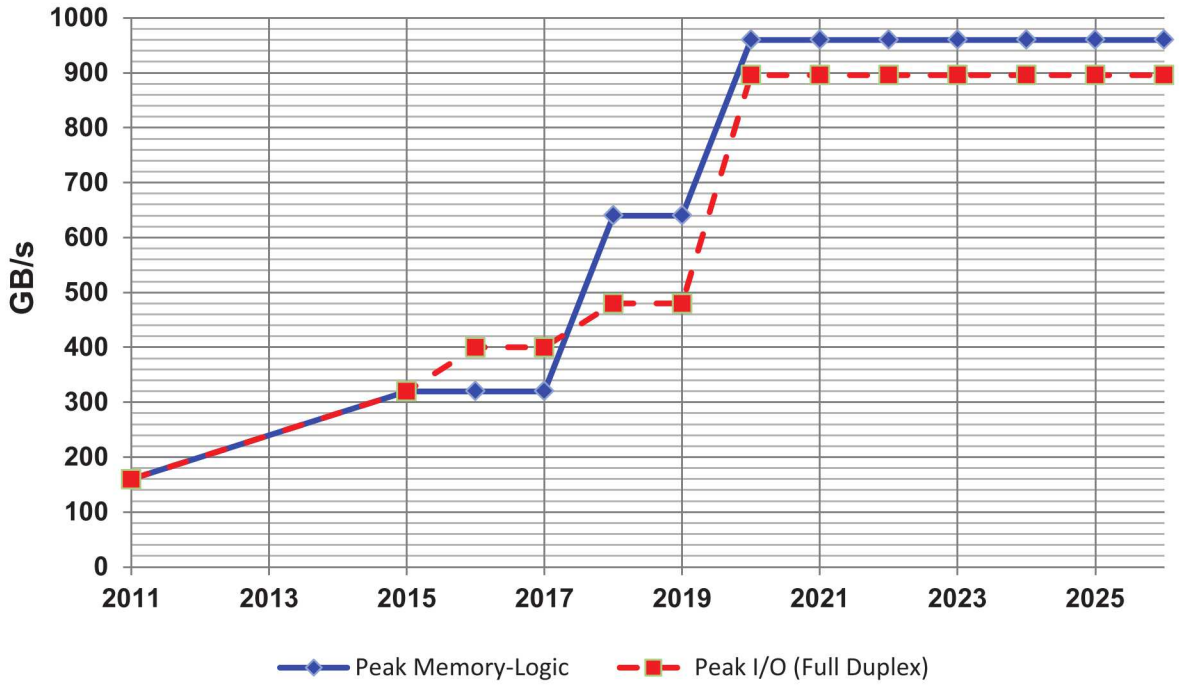


Figure 6.9. Bandwidth Projections for Hybrid Stacks.

In Fig. 6.7, the top three overlapping curves are the total stack power for 4, 8, and 16-high stacks. These three curves overlap in this figure because their differences are small as the number of die increase (only memory referesh power is counted). The two solid curves below this are the complete logic die (the solid black line) or the single DRAM die (the solid blue line). The remaining three dotted lines are power for subsystems of the logic die. The peak at 2018 is due to the simultaneous doubling of the number of vaults (and thus logic die router ports), and the increase in signalling rates.

6.3.2 Adding Cores

One interesting possibility is that of adding complete cores to the logic die, as suggested in the X-caliber project (Section 4.6). Section 7.3 of the Exascale report[20] discussed a strawman core that we will use here:

- 64-bit simple core,
- 32KB I and D L1 caches,
- 4 FPUs, each capable of a fused multiply add per cycle (equivalent to two "flops"),
- running at 1 GHz,
- when implemented in a 32 nm technology at 0.6V, the core + caches were estimated to consume 141mW, and each FPU another 30mW, for a total of 261mW.

To be a bit more realistic, raising the voltage to the ITRS standard for 32 nm of 0.9V gives a power per core of 587mW. These numbers were then scaled to match the technology expected for the logic chip through time, and added to the stack powers as discussed above. No change in clock rate was assumed. Fig. 6.10 diagrams this power for each of the three stack sizes, along with the total power assumed for the cores. Again the stack power curves overlap for the 4, 8, and 16 die stacks, reflecting that once four die is reached, the TSV's are fully used, and additional die require only refresh. The bump in core power from 2018 to 2018 reflects the rise from 32 vaults to 64. The bump in overall power is due to both the core power bump and the increase in I/O bandwidth.

Finally, as a point of comparison, Fig. 6.11 divides this power by the peak flops rate of all the cores. As a comparison point, this curve includes a line at 20pJ per flop, which was a metric of interest to the Exascale study.

6.4 Large system power transients

Power transients are an issue for current large HPC systems, and will raise increasing concerns going into the future.

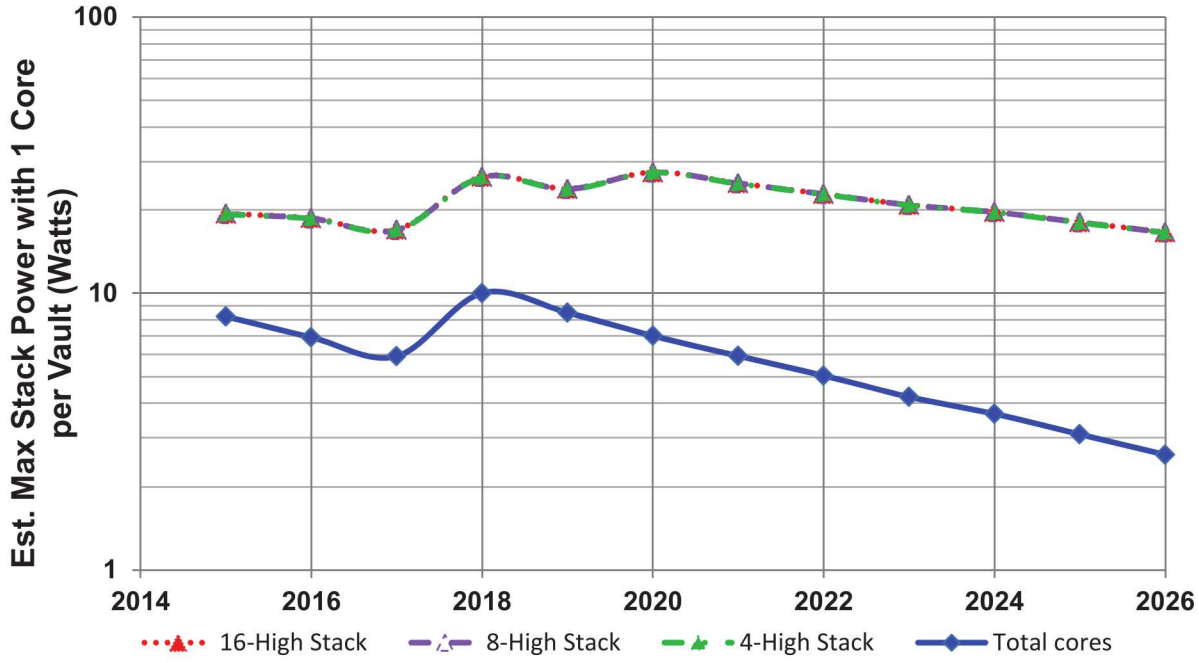


Figure 6.10. Hybrid Stack Power with One Core per Vault.

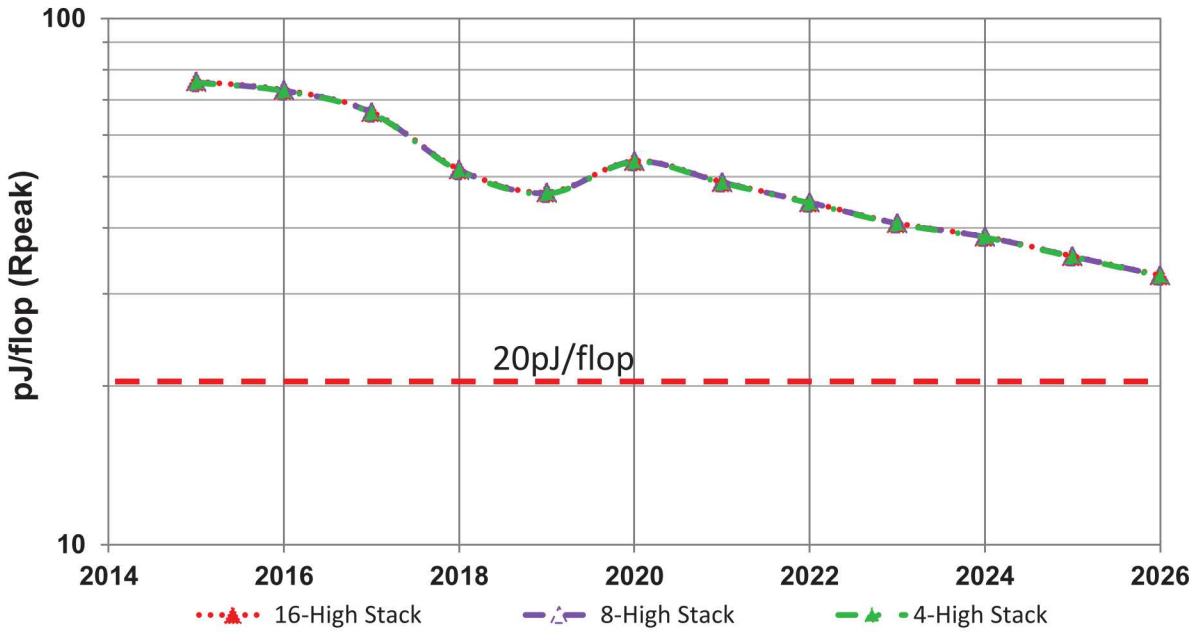


Figure 6.11. Hybrid Stack pJ per Flop.

Current large systems might have an average power of, say, 5 MW but see something like a 2 MW transient (so to 7 MW) over a time period much less than a single AC cycle. This can be the case, for example, where an application is at a sync point; when the last processor joins the sync, all the processors that were basically idle become busy with the next computation cycle. The transient puts stress on the power system at all levels, and is also an issue for the system's power vendor. The vendor must supply the transient without crossing limits on AC supply and on the power distribution system for both AC and DC supplies, and thus raise costs in the AC base supply and in the building distribution. The additional costs can be considerable and also affect cabinet power designs.

New processors are greatly increasing their voltage and clock regulation capabilities. This lowers the average power for systems, but also increases the ratio of peak power to average power: bigger transients and therefor more cost and complexity in the power system.

If large systems that currently use power of several megawatts grow to 20 MW, per the current exascale plans (or past that because things can't be pushed down enough), the transient increase will very likely exceed the average power of current systems. Eight or ten MW transients will need suppression. This is liable to be a considerable challenge.

This page intentionally left blank.

Chapter 7

Summary

The conclusions are an extension and elaboration of those from the Exascale report.

7.1 Architectural Trends

- 2004 saw commodity high performance microprocessors hit a power wall in what could be cooled economically. This was largely driven by a slowing of the rate at which operating voltage could be decreased, a factor that had been used before 2004 to offset the power increase due to more transistors on a chip running at higher clocks.

- After 2004 peak core clock rates no longer climbed at the rate that reflected the capabilities of the underlying devices, but were largely flat.

- Also in 2004, core micro-architecture techniques also hit a wall where adding more transistors could no longer significantly accelerate the single core performance of a core (that is, could no longer reduce the average cycles needed per instruction).

- This dichotomy led to a bi-furcation of HPC core architecture into two: heavyweight sockets kept clock rates at 2-3 GHz and added power cores per chip until cooling limits were reached; this was coupled with frequent off-loading of high-performance inter-node communication to separate NIC chips that themselves consumed significant energy. Lightweight sockets used much simpler cores and greatly reduced clock rates to keep the cooling infrastructure down to the point where very dense packaging was possible. In addition, all node functionality other than memory was integrated onto the same chip.

- Around 2008 the hybrid, heterogeneous, or accelerator-based class of architectures were introduced, which saw the coupling of a heavyweight socket with an equally high power-consuming multi-core accelerator socket where both the ISA and the microarchitecture of the accelerator's cores is radically different, with an emphasis on massive floating point parallelism. In addition, the accelerator socket had separate memory from that of the host socket, with a direct attach to higher speed, but lower density, GDDR-class DRAM. Also, this separation of memory spaces required explicit memory-to-memory copying before the accelerator could process data not in its memory, and before the heavyweight could access and distribute the data to other nodes as needed.

- The near term may begin to see a merger of these architectures, with variations of BigLittle architectures which combine heavyweight and lightweight cores on the same socket, all with the same ISA. Large numbers of such lightweight cores may then take on the functionality of an accelerator. In addition, the ability of threads to dynamically pass from heavyweight to lightweight cores, and back, may provide avenues to optimize power-performance on a dynamic basis.

- Also, with the imminent availability of very high bandwidth 3D stacks of DRAM, the mid term is liable to see the rise of 2.5D versions of all these architectures, with the ecent announcement of systems like Intel's Knights Landing, NVIDIA's Pascal, and Fujitsu's port-FX10 (see Section 4.6.3).

- The farther term is still likely to see the rise of stand-alone mergers of processing logic onto the bottom of 3D memory stacks, with multiple cores negating the need for other processor sockets of any other kind. That, however, will require the inclusion of non-volatile memory onto the stack and careful consideration of power dissipation.

7.2 Summary Projections

- Regardless of architectures, power will remain the number one design issue, with the power needed for off-socket communication at high bandwidth rapidly becoming a major design constraint.

- The number of cores, and thus number of independent program threads, that will be needed to grow overall application performance will continue to grow massively, greatly complicating our ability to create efficient programs.

- Memory is also constraining the shape of applications that can utilize the performance capabilities of all classes of architecture; both in available capacity, latency of access, and bandwidth to the processing logic.

- Memory capacity per node and per core compute cycle are decreasing very rapidly, especially for hybrid architectures, and are currently almost two orders of magnitude less than historical levels.

- Although not discussed in detail here, both the growth in parallelism and the need to go to extraordinary lengths to reduce power will result in systems with more logic that has less reliability than before, complicating the overall task of keeping systems up enough to perform large applications.

- The BigLittle architectures potentially combine the best of all three of the above architectures, but by sharing a common ISA may make code development simpler.

- While no 3D system has as yet been built for HPC applications, multiple 2.5D architectures are projected as near term products, with 3D memory stacks in very close proximity to processor sockets. With such systems, there is at least the potential for very significant

simultaneous improvements in virtually all the performance metrics that dominate the other architectures. Memory density, power, and resiliency will, however, still be concerns.

7.3 TOP500 Energy Model

- Although in need of a refresh to the basic model, the TOP500 energy per flop projection model made in 2008 seems to be holding relative true, with the actual data since then falling roughly in the middle of the bonds.
- Of all the architecture classes, the heavyweight class are the most limited by both power and memory bandwidth, and are likely to top out before the other architectures. They do, however, have, in the bulk synchronous message-passing model, the most mature programming environment.
- The lightweight architectures are significantly more power-efficient than the heavyweights, and permit a natural extension of today's heavyweight programming paradigms, but have limited memory and, because of the extra numbers of cores needed for a performance level, may exceed the point where these programming paradigms are sufficient for real applications.
- The hybrid architectures today are more energy efficient that the lightweights, but suffer from significantly lower memory capacity and require new programming paradigms that have as yet not proved as applicable as current ones.
- While systems at the top of the GREEN500 list were more energy efficient than those from the top of the TOP500, none of them scaled near the performance levels of the TOP500.

7.4 GRAPH500 Observations

- Unlike TOP500, GRAPH500 has both a memory capacity and performance dimension.
- In terms of just performance, there was a CAGR of 62X per year over the first two years of the benchmark, but that seems to have flattened.
- Both performance and problem size seem to be heavily driven by node count, with TEPS growing almost linearly in the number of nodes to the 0.92 power.
- The peak systems have been flat in terms of the number of nodes over the last two years.
- TEPS per node grew considerably in the first two years but has now flattened.
- A side study of just Blue Gene system performance shows that BlueGene/P performance per node flattened almost immediately, but BlueGene/Q showed a better than 10X increase per node on the same hardware over time. This can only reflect a greatly improved algorithm, but recent results seem to indicate that performance has flattened.

- The best performance per node is from the "Other" category, mostly Convey systems which have a large number of memory ports in a single node box. However, the Convey system have limited scalability, and XMT systems lose significant performance per node as they scale up.

- The best lightweight systems are perhaps a factor of 30 less in TEPS per node than the Convey numbers, but they both scale better, and something in excess of 32 more nodes can be packaged in the same size as a Convey system today. However, once we get to systems of 512 nodes or bigger, the lightweight systems provide significantly more performance per node.

- There are multiple heavyweight systems that provide near best TEPS per node and are significantly better than the best of the lightweight, due, we would guess, to the higher memory bandwidth available. However, this advantage deteriorates rapidly with system size.

- Perhaps most importantly, we seem to have peaked over the last 18 months, with little gain per node. This may signal that we have hit an implementation limit where some new technology, perhaps interconnect link bandwidth, will be needed to foster more growth.

7.5 Other Application Observations

- The emergence of early comparative measurements on a new scientific benchmark, HPCG, echoes some of the GRAPH500 observations, but with a real focus on the memory systems. This computational code relies on very sparse matrix operations, rather than the dense operations of the TOP500. This in turn requires multiple memory references per flop, with little of either the temporal or spatial locality needed by current architectures to feed multiple floating point units. Floating point efficiency numbers from the first release are literally as little as 1/80th of the efficiency of the same systems on LINPACK, and this is for system in the most recent top 10. If this continues, a major rethinking of architectures will be needed.

- The HPCS benchmark suite tests multiple aspects of architecture beyond raw computational capability. While the data doesn't support temporal studies to the extent that the others do, they do allow scaling studies of performance versus node count. Those benchmarks that were bandwidth-sensitive had distinct decreasing trends in performance per node as the number of nodes increased. Those that were not were relatively flat.

  This was reinforced by the comparison of STREAM results versus HPL for the same systems. There seems to be a strong correlation here, due probably to the fact that both are driven by memory bandwidth within the nodes. Alternatively, a network bandwidth benchmark GUPS when compared to the dense flop-intensive HPL shows minimal correlation.

7.6 3D Stack Projections

- 3D stacks of just memory have the potential to significantly affect the currently declining "memory per" curves in a positive direction, with the potential of up to a "terabyte per DIMM-like structure" in a few years.

- Power per memory stack. when used to support a conventional processing socket of any architectural class, is liable to average in the 10 watt range for a long time, with a large fraction of this in the off-chip I/O. This is sufficiently low to avoid area-consuming cooling structures, but may cause problems if dozens of such stacks are to be integrated in a small area, such as a DIMM. However, if systems do not need all this bandwidth, there is the potential for significant savings.

- One caveat to the above power number for HMCs is that sets of 2-port HMCs connected into 1D chains add capacity, but with a lower average power per HMC than simply using a spec number, as the maximal number of memory references is bounded by a single link from the processor chip, and is spread over all the HMCs in the chain. Thus the access energy in each HMC is, on average, much lower than it would be if all such references went to a single HMC. In addition, only 2 ports per HMC need be activated (and only 1 and the end), significantly reducing the power over a full-blown 4-port part.

- Adding cores to each vault has the potential to change the architecture of the overall system, with the potential of getting within a small factor of the original Exascale 20pF/flop goal. However, the power per stack may grow to the 25 watt range, assuming that all cores and all I/O are running at 100%. Further studies are needed to determine what real algorithms may demand, and thus if any scale-back on either core complexity and/or I/O is rational.

This page intentionally left blank.

References

[1] Isscc 2014 trends, Feb. 2014.

[2] Bryan Black, Murali Annavaram, Ned Brekelbaum, John DeVale, Lei Jiang, Gabriel H. Loh, Don McCaule, Pat Morrow, Donald W. Nelson, Daniel Pantuso, Paul Reed, Jeff Rupley, Sadasivan Shankar, John Shen, and Clair Webb. Die stacking (3D) microarchitecture. In *Microarchitecture, 2006. MICRO-39. 39th Annual IEEE/ACM International Symposium on*, pages 469 –479, dec. 2006.

[3] IBM Blue Gene team. Overview of the IBM Blue Gene/P project, 2008.

[4] Eun-Seok Choi, Hyun-Seung Yoo, Han-Soo Joo, Gyu-Seog Cho, Sung-Kye Park, and Seok-Kiu Lee. A novel 3d cell array architecture for terra-bit NAND flash memory. In *Memory Workshop (IMW), 2011 3rd IEEE International*, pages 1–4, 2011.

[5] P. Christie and D. Stroobandt. The interpretation and application of Rent's rule. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 8(6):639–648, 2000.

[6] HMC Consortium. Hybrid memory cube specification 1.0, April 2013.

[7] Jack Dongarra and Michael Heroux. Toward a new metric for ranking high performance computing systems. Sandia Report SAND2013 4744, Sandia National Labs, June 2013.

[8] Jack J. Dongarra. Performance of various computers using standard linear equations software. *SIGARCH Comput. Archit. News*, 20(3):22–44, June 1992.

[9] Mrinmoy Ghosh and Hsien-Hsin S. Lee. Smart refresh: An enhanced memory controller design for reducing energy in conventional and 3D die-stacked DRAMs. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 40, pages 134–145, Washington, DC, USA, 2007. IEEE Computer Society.

[10] Peter Greenhalgh. big.little processing with arm cortex-a15 and cortex-a7, 2011.

[11] R.A. Haring, M. Ohmacht, T.W. Fox, M.K. Gschwind, D.L. Satterfield, K. Sugavanam, P.W. Coteus, P. Heidelberger, M.A. Blumrich, R.W. Wisniewski, A. Gara, G.L.-T. Chiu, P.A. Boyle, N.H. Chist, and Changhoan Kim. The IBM Blue Gene/Q compute chip. *Micro, IEEE*, 32(2):48 –60, march-april 2012.

[12] Michael Heroux, Jack Dongarra, and Piotr Luszczek. Hpcg technical specification. SAndia Report SAND2013-8752, Sandia National Laboratories, October 2013.

[13] Elpida Memory Inc. Introduction to GDDR5 SDRAM, March 2010.

[14] ITRS. International technology roadmap for semiconductors: 2012 update. Technical report, Semiconductor Industry Association, Dec. 2012.

[15] Bruce Jacob, Spencer Ng, and David Wang. *Memory systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2010.

[16] Taeho Kgil, Shaun D'Souza, Ali Saidi, Nathan Binkert, Ronald Dreslinski, Trevor Mudge, Steven Reinhardt, and Krisztian Flautner. Picoserver: using 3D stacking technology to enable a compact energy efficient chip multiprocessor. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, ASPLOS-XII, pages 117–128, New York, NY, USA, 2006. ACM.

[17] H. Kimura, P. Aziz, Tai Jing, A Sinha, R. Narayan, Hairong Gao, Ping Jing, G. Hom, A Liang, E. Zhang, A Kadkol, R. Kothari, G. Chan, Yehui Sun, B. Ge, J. Zeng, K. Ling, M. Wang, A Malipatil, S. Kotagiri, Lijun Li, C. Abel, and F. Zhong. 2.1 28gb/s 560mw multi-standard serdes with single-stage analog front-end and 14-tap decision-feedback equalizer in 28nm cmos. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, pages 38–39, Feb 2014.

[18] Peter Kogge. Tracking the effects of technology and architecture on energy through the top 500, green 500, and graph 500. In *2012 International Conf. on Supercomputing*, ISC '12, 2012.

[19] Peter Kogge and Timothy Dysart. Using the top500 to trace and project technology and architecture trends. In *Proceedings of the 2011 ACM/IEEE conference on Supercomputing*, SC '11, 2011.

[20] Peter M. Kogge, Keren Bergman, Shekhar Borkar, Dan Campbell, William Carlson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Kerry Hill, Jon Hiller, Sherman Karp, Stephen Keckler, Dean Klein, Robert Lucas, Mark Richards, Al Scarpelli, Steven Scott, Allan Snavely, Thomas Sterling, R. Stanley Williams, and Katherine Yelick. Exascale computing study: Technology challenges in achieving exascale systems. Technical Report CSE 2008-13, Univ. of Notre Dame, Sept. 2008.

[21] P.M. Kogge and D.A. Bayliss. Comparative performance analysis of a big data nora problem on a variety of architectures. In *Collaboration Technologies and Systems (CTS), 2013 International Conference on*, pages 22–34, 2013.

[22] M.Y. Lanzerotti, G. Fiorenza, and R.A. Rand. Interpretation of rent's rule for ultralarge-scale integrated circuit designs, with an application to wirelength distribution models. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 12(12):1330–1347, 2004.

[23] Gabriel H. Loh. 3D-stacked memory architectures for multi-core processors. In *Proceedings of the 35th Annual International Symposium on Computer Architecture*, ISCA '08, pages 453–464, Washington, DC, USA, 2008. IEEE Computer Society.

[24] Niti Madan and Rajeev Balasubramonian. Leveraging 3D technology for improved reliability. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 40, pages 223–235, Washington, DC, USA, 2007. IEEE Computer Society.

[25] Gordon Moore. Excerpts from a conversation with Gordon Moore: Moores Law. *Video Transcript, Intel*, 2005.

[26] Gordon E Moore et al. Cramming more components onto integrated circuits, 1965.

[27] R.C. Murphy and P.M. Kogge. On the memory access patterns of supercomputer applications: Benchmark selection and its implications. *Computers, IEEE Transactions on*, 56(7):937–945, 2007.

[28] Richard Murphy. X-caliber and exascale grand challenge research summary, Sept. 2011.

[29] J. Thomas Pawlowski. Hybrid memory cube: a re-architected DRAM subsystems. In *Hot Chips 23*, Palo Alto, CA, August 2011.

[30] J. Thomas Pawlowski. 3D stacked memory architectures for multi-core processors. In *3D Architectures for Semiconductor Integration and Packaging*, San Francisco, CA, USA, 2012.

[31] Betty Prince. *High Performance Memories: New Architecture DRAMs and SRAMs–Evolution and Function*. Wiley, 1999.

[32] E. F. Rent. Microminature packaging - logic block to pin ratio. Memoranda, Dec. 1960.

[33] Guangyu Sun, Xiangyu Dong, Yuan Xie, Jian Li, and Yiran Chen. A novel architecture of the 3D stacked MRAM L2 cache for cmps. In *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*, pages 239 –249, feb. 2009.

[34] The BlueGene/L Team, T Domany, Mb Dombrowa, W Donath, M Eleftheriou, C Erway, J Esch, J Gagliano, A Gara, R Garg, R Germain, Me Giampapa, B Gopalsamy, J Gunnels, B Rubin, A Ruehli, S Rus, Rk Sahoo, A Sanomiya, E Schenfeld, M Sharma, S Singh, P Song, V Srinivasan, Bd Steinmacher-burow, K Strauss, C Surovic, Tjc Ward, J Marcella, A Muff, A Okomo, M Rouse, A Schram, M Tubbs, G Ulsh, C Wait, J Wittrup, M Bae, K Dockser, and L Kissel. An overview of the bluegene/l supercomputer, 2002.

[35] Aniruddha N. Udipi, Naveen Muralimanohar, Rajeev Balasubramonian, Al Davis, and Norman P. Jouppi. Combining memory and a controller with photonics through 3D-stacking to enable scalable and energy-efficient systems. In *Proceeding of the 38th annual international symposium on Computer architecture*, ISCA '11, pages 425–436, New York, NY, USA, 2011. ACM.

DISTRIBUTION:

1 Peter Kogge
Department Of Computer Science and Engineering
384 Fitzpatrick Hall
Notre Dame, IN 46556

1 MS 1319 David Resnick, 1422

1 MS 1319 Jim Ang, 1422

1 MS 0899 Technical Library, 9536 (electronic copy)

Sandia National Laboratories