Evaluating Architectures for Application-Specific Parallel Scientific Computing Systems

Mark Murphy
Kurt Keutzer
Leonid Oliker
Chris Rowen
John Shalf

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2008-13
http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-13.html

February 12, 2008

Acknowledgement

Many thanks to Dror Maydan, Bill Huffman, and Grant Martin of Tensilica. Also thanks to Michael Wehner of Lawrence Berkeley National lab for his expertise in understanding the climate model.

Evaluating Architectures for Application-Specific Parallel Scientific Computing Systems

Mark Murphy, Kurt Keutzer, Leonid Oliker, Chris Rowen, John Shalf

***Abstract*—In this work, we examine the computational efficiency of scientific applications on three high-performance-computing systems based on processors of varying degrees of specialization: an x86 server processor, the AMD Opteron; a more specialized System-on-Chip solution, the BlueGene/L and BlueGene/P; and a configurable embedded core, the Tensilica Xtensa. We use the atmospheric component of the global Community Atmospheric Model to motivate our study by defining a problem that requires exascale-class computing performance currently beyond the capabilities of existing systems. Significant advances in power-efficiency are necessary to make such a system practical to field.**

***Index Terms*—Configurable architectures, Microprocessors, Multiprocessing, HPC.**

I. Introduction

The issue of effective system performance and power efficiency is becoming an urgent concern for all large-scale computing facilities. We are entering an era where petaflop High Performance Computing (HPC) systems are anticipated to draw enormous amounts of electrical power. For example, current leading HPC systems typically draw less than 2 MW to deliver 100 TF peak performance; while its petaflop-scale successors in 2010 are projected to draw as much as 15 MW if fully configured. More alarmingly, the DOE E3 report projects an exaflop HPC system requiring over 130 MW of power. The cost of power will exceed the procurement costs of such systems, and will thus ultimately limit the practicality of future state-of-the-art HPC platforms. These trends will lead to a crisis in HPC in the not-too-distant future, unless we work aggressively to develop more power-efficient solutions.

Many of the most cost-effective supercomputing solutions are based on high-performance commodity processors, such as high-end Intel Pentium and Itanium, AMD Opteron, and IBM Power processors. These processors were defined in an era when Moore's Law silicon scaling offered continuous exponential improvements in transistor speed and performance per unit area. Device dimensions and operating voltages decreased continuously producing the triple benefit of lower cost, higher performance, and lower power. Energy efficiency in digital circuits has improved dramatically, predictably, and continuously over the past 25 years.

In the past several years, however, this trend has slowed. While the scaling of device dimensions is continuing at almost historical rates, operating voltage is not, so dynamic and static power dissipation are no longer improving [7], [8]. In fact, we appear to have reached practical limits in IC power density, so that any further improvement in circuit density must be balanced by reduction in circuit activity, especially by reduction in clock frequency. As a result, the clock frequency of high-end processors has been essentially flat at less than 4GHz for several years. Consequently, the number of cores per chip is likely to double every 18 months henceforth.

One approach to mitigating the growing crisis of power consumption in future generations of processing elements is to leverage the enormous resources of embedded processor technology. The embedded market relies on architectural customization to meet the demanding cost and power efficiency requirements of embedded applications such as MP3 players, cell phones, and PDAs. Whereas one would expect a desktop or server processor vendor to deliver a new CPU core design every two years, a typical embedded vendor may generate up to 200 unique chip designs every year. In order to keep up with this demanding pace for semi-customized designs, leading embedded design houses such as IBM Microelectronics, Freescale, and Tensilica have evolved sophisticated tool sets to accelerate the design process through semi-automated synthesis of custom processor designs. It is of great interest to find out how to leverage the tremendous resources of this technology sector to develop a power-efficient HPC system based on application-driven, semi- custom embedded processors.

In the past, HPC systems based on semi-custom components could not compete with the price/performance of systems that primarily leveraged commodity-off-the-shelf components. However, as the price of power and cooling for such systems begins to exceed the capital procurement costs, customizations that improve energy-efficiency are increasingly cost-effective. Looking to the future, the world's major manufacturer of general-purpose microprocessors emphasizes the importance of supporting special-purpose hardware [1]. At the same time today's system developers are moving away from application-specific integrated circuit (ASIC) solutions, due to escalating non-recurring engineering costs, and are instead moving toward programmable solutions [2]. These two individual trends alone are sufficient to motivate the investigation of application-specific instruction processors (ASIP) as a building-block for application-specific parallel systems. Furthermore, parallelism appears to be our main hope for achieving high-performance within acceptable power budgets, but this migration begs the question: What is the basic building block of future parallel manycore (100's to 1000's of processing elements) multiprocessor systems? The "Berkeley View" [3] motivates this problem of defining the basic building block, and constrains the solution to be built around RISC cores of 5-9 stage pipelines; however, it leaves open the question as to the roles that fabrication-time processor configurability [4] may play in the defining the building block of future manycore multiprocessors systems. This paper further investigates that question.

The degree of micro-architectural customizability of fab-time configurable processors is defined on a continuum, so that development costs are traded against the computational efficiency of the delivered system. In this work, we examine the question of the benefits that can be derived from different amounts of investment in customization by comparing systems based on commodity x86 server building blocks such as the AMD Opteron, a more customized SoC solution that uses undifferentiated embedded cores such as the BlueGene/L and /P systems, and a custom configurable core such as the Tensilica XTensa. We use the atmospheric component of the global-climate model (CAM) to motivate our study by defining a problem that requires exascale-class computing performance that is currently beyond the capabilities of existing systems and may require significant advances in power-efficiency to make such an HPC system practical to field. For prior work in processor customization, see [13] and [14], for issues regarding configurable System-on-Chip designs, see [24], and for Network-on-Chip design for Multi-Processor-System-on-Chip systems, see [15]. For a structured design flow for ASIPs, see [2].

II. Methodology Used for this Evaluation

In this section we briefly describe our basic approach to evaluating the configurable processor as our building block. This can be described according to the methodology described in [2] which we briefly reiterate immediately below:

- Identify a high-impact application domain and common application requirements.
- Inclusively identifying the design space. Evaluate the likely architectural alternatives (AMD commodity, BG/P SoC with undifferentiated embedded core, XTensa customized core)
- Exploring the HPC design space.
- Successfully deploying the ASIP

This study extends the approach of [2] to more broadly consider multiprocessing in general, and inter-processor communication in particular.

III. Identifying a High-Impact Scientific Application

Only a modest number of large HPC systems are built and sold each year, so the economics of engineering the hardware and software has become a central concern. Supercomputing centers service extremely diverse workloads, but at the largest scale, only a handful of problems are ready to scale up to fully exploit the available computing capability. Perhaps it is more cost-effective and efficient to focus our development resources to create machines that target that handful of problems rather than attempting to scale conventional architectures to such unprecedented sizes. The most suitable applications appear to be ones that permit decomposition into a large number of parallel tasks without heavy global communication especially where the computation scales significantly more rapidly than global communication as the overall problem size increases. Structured grid computation problems appear to fit this criterion.

The application domain for our design study is atmospheric general circulation modeling at kilometer-scale resolution. The model used to drive our design exploration is an entire large scale application: the finite volume version of the Community Atmospheric Model (fvCAM). Developed at the National Center for Atmospheric research, a great deal of effort has gone into the CAM software system to ensure portability and robustness across a wide variety of high performance computing platforms. The fvCAM uses a finite volume approximation to the atmospheric equations of motion as developed by Lin and Rood [10]. It is representative in both scientific sophistication as well as computational cost to any of the leading atmospheric general circulation models in current use. The public release of version 3.1 was used without modification in this study.

Climate and weather modeling is a compelling scientific problem. The accuracy of results from climate models have the potential to influence policies with trillion-dollar ramifications. The computational requirements of climate modeling are well understood. A cloud-resolving model which can provide climate scientists meaningful data needs 1-kilometer resolution. Two key applications present real-time constraints on the completion time: simulations of anthropogenic climate change and regional scale weather forecasting.

For climate change analyses, two types of simulations are often employed. The longer of the two are control runs that simulate statistically stationary climate over periods of multiple millennia to adequately characterize internal climate variability for climate change detection studies. A shorter duration category involves transiently forced simulations of the past or current centuries. Hence, a reasonable metric is to require that the model simulate time 1000 times faster than real time. Under this constraint, the control run integration can be completed in one year and transient runs can be completed several months later depending on how many realizations and scenarios are required. We note that the runs made for the Fourth Assessment Report of the Intergovernmental Panel on Climate Change of the current NCAR Community Climate System Model, CCSM3.0, ran about 1600 times faster than real time on IBM Power4/p690 machines.

The major medium range weather prediction centers generally perform at least a one high resolution ten-day forecast each day in addition to ensembles of forecasts at lower resolutions. The associated throughput metric is significantly relaxed in this application as the model must simulate time only 10 times faster than real time. Seasonal forecasting requires an equivalent computational throughput as the integrations are for a period of about ten times longer but the turnover rate is not as constrained as for the daily forecast.

Atmospheric general circulation models (AGCM) consist of two distinct parts solving different aspects of the problem. Each has distinct scaling properties of the computational costs. The first part, often referred to as dynamics, solves the atmospheric equations of motion expressed by the Navier-Stokes equations of fluid dynamics. There are many different approaches to approximating a numerical solution to these equations. In CAM, a finite volume method [10] is one of

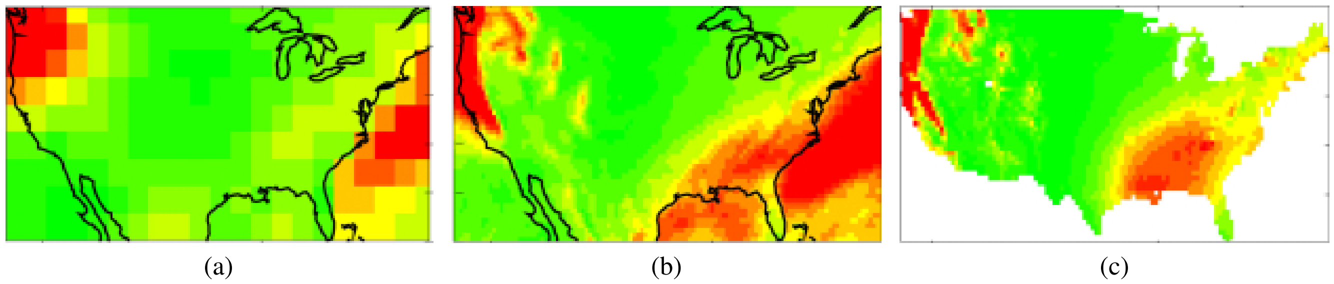
Fig. 1. Climate models at (a) 200 KM and (b) 10 KM, compared to (c) observed data. Notice the greatly improved fidelity of the Sierra-Nevada micro-climates near the western coast.

three options. In all options for CAM, the atmosphere is approximated to be in hydrostatic equilibrium. The highly stratified nature of the atmosphere at large scales makes this a very good approximation at the relatively coarse resolutions of fvCAM in current use. It will break down at the resolutions we will extrapolate to in the next section and is important to consider in that discussion.

The second part of AGCMs, often referred to as physics, approximate the unresolved or external processes relevant to the state of the atmosphere. These processes are included in the solution as source terms to the Navier-Stokes equations but must be calculated externally to the dynamics portion of the code. Relevant processes include but are not restricted to radiative transport of energy, convective transport of moisture, boundary layer effects, gravity wave drag, sub-grid scale turbulence and surface conditions.

Models such as fvCAM based on a latitude-longitude discretization of the globe require a third component, often referred to as filters. The stability of explicit solutions to the Navier-Stokes equations is governed by the Courant-Friedrichs-Levy (CFL) condition, which limits the length of the allowable time step. This limit is a function of the minimum grid spacing. Since the latitude-longitude grid is much finer near the poles, the CFL condition is overly restrictive globally. To counter this, the dynamics time step may be chosen by a mid-latitude stability criterion. fvCAM dampens the the resulting instability at the poles by careful application of latitudinally-depended Fourier filters.

We now proceed to summarize the computation requirements of fvCAM, to provide more detailed motivation for our design space exploration. We focus on three different aspects of the system design: in Section *A* we discuss the instruction-level issues which affect the design of the building-block core, in Section *B* we discuss the requirements of the memory system, and in Section *C* the inter-processor communication characteristics of climate modeling systems.

A. Operation Count: Implications for Core Design

In Table I, we show the minimum *sustained* computational performance in Teraflops required to integrate fvCAM 1000 times faster than real time. The performance required for medium range weather forecasting would be about 100 times less. We note that the relative distribution of operations changes dramatically with resolution. In the 2°x2.5° *B* mesh case, the dynamics and physics respectively consume 57% and 41% of the operations, or roughly equivalent amounts. In the 0.5°x0.675° *D* mesh case, the percentages have shifted to 81% and 14%, a dramatic change. In the kilometer scale *I* mesh, the percentage of total operations required by the physics is almost negligible at just over a half of a percent. Perhaps most surprising is that the filters never dominate the calculation, even though their computation scales as $M\ln(M)N^2$. This suggests that the need for alternatives to the simple latitude-longitude discretization of the sphere may not be as critical at high resolution as one might suppose; at least from computational considerations. We note however that some of the alternative discretizations possess other appealing properties, most notably a more uniform distribution of individual cell volumes. These will ultimately prove to be far more favorable strategies to the design of ultra-high resolution global atmospheric models.

B. Memory Requirements

The total memory required in the model might be expected to scale as the total number of computational mesh points, $M \times N$. In fact, the measured memory requirements are somewhat below this as can be seen by comparing in Table II the measured results (third column) with this upper bound (fourth column). This is likely a consequence of temporary memory requirements which do not cover the entire domain. A linear fit to the measured memory (fifth column) predicts somewhat lower extrapolated requirements. In this calculation, the global arrays required by processor 0 for initialization have been removed as a code scalable to ultra-high resolution must have some sort of parallel I/O. In any event, at the 1.5km scale *I* mesh, the total memory requirement should be of order 25 Terabytes or less.

This scaling behavior of the total memory requirements is independent of processor configuration as it is determined only by the size of the problem and not the geometry of the domain decomposition. Note that this scaling behavior is the same as for the processor count in a two dimension horizontal domain decomposition scheme. Hence, the amount of memory on any individual processor could be held constant with resolution by increasing the number of processors at the same rate as the number of grid cells. For the maximum processor case of 9 mesh cells per subdomain, the required single processor memory is just over 1 MB. In the 100 mesh cells per subdomain case considered above, this estimate increases to about 10MB. Substantial single processor memory requirements are

Mesh Name	Resolution	Horizontal Resolution Near Equator	Performance 1000x spdup (Tflop/s)	% total time in dynamics	% total time in filters	% total time in physics
B	2°x2.5°	200km	0.002	57%	2%	41%
C	1°x1.25°	100km	0.014	71%	3%	25%
D	0.5°x0.625°	50km	0.098	81%	4%	14%
E	0.25°x0.3125°	25km	0.727	87%	5%	8%
F	0.125°x0.156°	12.5km	5.608	90%	6%	4%
G	0.068°x0.078°	6.25km	44.15	92%	6%	2%
H	0.038°x0.039°	3.13km	351.3	92%	7%	1%
I	0.016°x0.020°	1.57km	2809.7	92%	7%	1%

TABLE I
MINIMUM SUSTAINED COMPUTATIONAL PERFORMANCE (IN TFLOP/S) REQUIRED TO INTEGRATE FVCAM 1000 TIMES FASTER THAN REAL TIME AND THE RELATIVE PERCENTAGES OF THE TOTAL OPERATIONS IN THREE MAIN SEGMENTS OF THE CODE.

					1D 3-rows	2D 3x3	2D 10x10	
Mesh Name	Horizontal Resolution Near Equator	Measured Total Mem (GB)	Max Memory (GB)[1]	Memory Least Sqrs Fit (GB)[2]	Required Mem/Proc (MB)	Required Mem/Proc (MB)	Required Mem/Proc (MB)	Ratio (Bytes/ /Flop)
B	200km	1.5	1.5	1.3	51	1.06	11.8	0.68
C	100km	4.9	6	5.3	102	1.06	11.8	0.44
D	50km	21.4	24	21.3	203	1.06	11.8	0.25
E	25km	—	96	85.0	406	1.06	11.8	0.13
F	12.5km	—	384	339.9	813	1.06	11.8	0.070
G	6.25km	—	1536	1359.4	1625	1.06	11.8	0.035
H	3.13km	—	6144	5436.7	3251	1.06	11.8	0.018
I	1.56km	—	24576	21745.1	6502	1.06	11.8	0.0089

TABLE II
ESTIMATES OF MEMORY REQUIREMENTS FOR FVCAM AS A FUNCTION OF HORIZONTAL RESOLUTION AND DECOMPOSITION STRATEGIES. THE LAST COLUMN SHOWS THE RATIO OF SINGLE PROCESSOR MINIMUM MEMORY CAPACITY TO MINIMUM SUSTAINED PERFORMANCE FOR ANY DOMAIN DECOMPOSITION

necessary for the one dimensional decomposition case as the number of processors can only be increased at the same rate as the number of latitude lines. Table II (column six) shows the individual processor minimum memory requirements for the one dimension horizontal domain decomposition scheme peaking at over 6GB per processor. The relationship is linear with M, the number of longitude lines.

Interestingly, the relationship between single processor performance and single processor memory consumption is independent of domain decomposition strategy. In the last column of Table II, we see that the ratio of single processor memory footprint to sustained single processor throughput (expressed in bytes per flop) decreases as horizontal resolution increases. These values are the same for the ratio of overall machine memory to sustained floating point operation rate. This decrease is a consequence of the CFL stability criterion which causes the operation count to scale at a rate greater than the number of mesh cells.

If the simulation were expanded to use 100 vertical layers, and employed a 10-way vertical domain decomposition for a 10x10x10 domain decomposition, the memory requirements per subdomain are estimated to be 5MB.

C. Interprocessor Communication Requirements

Three factors cause the sustained machine performance to be less than the advertised machine performance on any platform. The first of these is an inability to realize peak performance on the calculation within a single processor. This is generally a function of chip design as well as compiler effectiveness. The second of these is the communication of data between processors necessary to perform that calculation. Interpretation of the sustained processor speeds discussed in Section III-A relative to advertised peak computational rates must take both factors into account. Interprocessor communication in these maximal subdomain configurations is not negligible and may even dominate. A third factor, which we will not discuss in detail here, is the load imbalance caused by uneven distribution of the computational workload dictated by model physics considerations.

fvCAM lends itself to two different styles of domain decomposition. The first, least-parallel decomposition is a one dimensional division where each subdomain contains all longitude lines but only a subset of latitude lines. For low-resolution simulations with only limited parallelism, the decreased communication overhead is advantageous. The second, a two dimensional decomposition, divides subdomains by both latitude and longitude lines. Nearest-neighbor communication is doubled over the one dimensional case, but has much better scaling properties. Current models utilize between 26 and 64 vertical atmospheric layers. In a full atmospheric model, we consider that at least 100 vertical layers are required to accommodate the cloud system resolving model as well as the lower stratosphere and boundary layer. These layers could also be subdivided to provide more parallelism, although vertically contiguous domains require extremely tight communication coupling and should reside on the same SMP socket.

Speculating about the scaling behavior of the communica-

[1]Upper bound calculated as: Memory_{n+1}=4xMemory_n.
[2]Least squares linear fit ($y = Ax$) to the total measured memory.

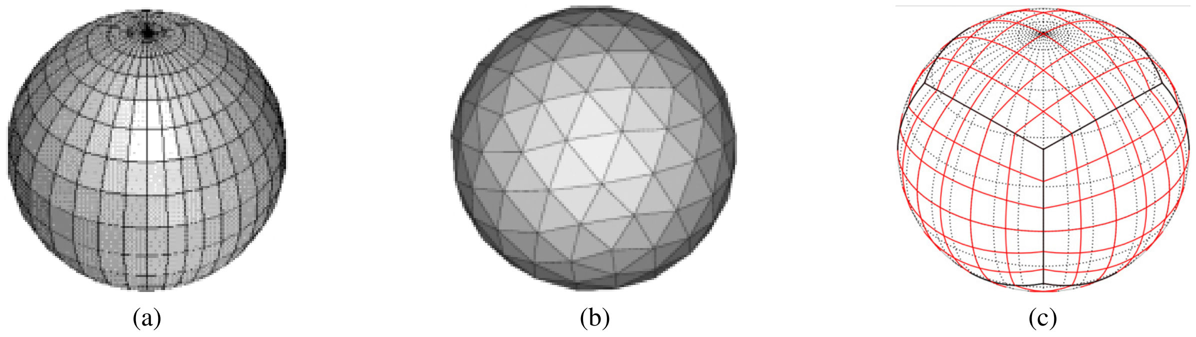

Fig. 2. Alternative domain decomposition topologies: (a) Latitude-Longitude, (b) Icosahedral, and (c) Cube-Sphere.

tions costs is made difficult by the lack of a purely horizontal two-dimensional domain decomposition for fvCAM. However, some information about nearest neighbor communication in this case may be derived by measuring point-to-point communication statistics on the one dimensional decomposition with the Integrated Performance Monitor (IPM) [12] tool. The tool reveals that in the one dimensional decomposition the largest messages contain the ghost cells values of three concatenated 3D variables sent once per dynamics time step to the north and south neighboring domains. Approximately ten messages per dynamics time step are sent containing a single 3D variable. A number of considerably smaller messages containing 2D variables are also sent. In the two dimensional domain decomposition, the number of these messages must be doubled to include the east and west neighbors but are reduced in size. For a fixed number of cells per subdomain, message sizes are independent of horizontal resolution but are dependent on the number of domains. In the one dimensional decomposition, the messages contain ghost cell values over the entire range of longitudes. Hence, message sizes are independent of the number of domains but dependent on horizontal resolution in this case.

Table III shows an estimate of the messaging requirements for fvCAM as a function of horizontal resolution and decomposition strategies. The second and third columns of Table III show the one-dimensional decomposition measured average message size and the largest measured message size in the nearest neighbor communication. Measurements were performed at the 50km *D* mesh and extrapolated to the other resolutions. In the two dimensional decomposition, the nearest neighbor message size is determined by the size and logical shape of the domain. For a fixed number of cells per subdomain, these messages are of a constant size. For the maximum processor case of 9 mesh cells per subdomain, the average nearest neighbor message size would be about 1.5KB and the maximum would be about 5.6KB. In the 100 horizontal cells per subdomain case considered above, the average would be about 5.0KB and the maximum would be about 62.4KB.

The time required to send a message is the sum of the fixed latency required to send a message between neighboring processors and the time it takes to serialize the message. For very small subdomains such as the 9-cell case, the messages may end up so small that the latency term ends up dominating the communication time, thereby wasting available communication bandwidth. Therefore, aggregating messages for the boundary exchange may be beneficial by ensuring that the interprocessor communication links remain saturated.

In the discussion above of the individual processor speed, we performed our calculations without accounting for communication costs, to obtain a lower bound on processor performance requirements. Similarly, we estimate a lower bound on the communication requirements separately from the cost of calculation. Separate treatment of these costs is a reasonable approach given the explicit message-passing communication is bulk-synchronous, where communication and computation occur in distinct phases. The total execution time of the simulation would, of course, be a mixture of these two components. Using the performance metric appropriate to climate simulation (1000 times realtime), the total rate of nearest neighbor messages per processor that must be sent and received each second must exceed the values listed in column 4 of Table III (expressed in messages/second/subdomain) for the one dimensional decomposition. As mentioned before, in the two dimensional decomposition case the number of messages per processor per dynamics time step is twice that of the one dimensional decomposition thus doubling these message rate requirements. These messages include the single and triple 3D variables mentioned above as well as numerous smaller messages. The average nearest neighbor send/receive data volume rate per processor of these messages must exceed the values in MB/s listed in fifth column of Table III for the one dimensional decomposition. For the two dimensional domain decomposition strategy the average rates are listed for maximum processor case of 9 (3x3) and 100 (10x10) mesh cells per subdomain, in columns 6 and 7 respectively. The total amount of nearest neighbor data that must communicated depends on the number of processors.

The latitude-longitude domain decomposition of fvCAM is not the only potential candidate. At least two other discretizations of the globe exist, and are preferable to the fvCAM model primarily for their lack of polar singularities. Without the polar singularities, the CFL constraint is more uniform across the globe and the operation count scales more gracefully with increased resolution. Other code bases for climate modeling use different discretizations of the sphere, and differ from fvCAM primarily in their requirements on message-passing networks; the dynamics computations at each point in the discrete grid are similar. Figure 2 illustrates two of the leading

	1D 3-rows				2D 3x3	2D 10x10
Mesh Name	Avg Msg Size (KB)	Max Msg Size (KB)	Msgs/ Second/ Domain	Msg Vol/ Domain/ Second (MB/s)	Msg Vol/ Domain/ Second (MB/s)	Msg Vol/ Domain/ Second (MB/s)
B	72.6	270	23	1.6	0.07	0.21
C	145	540	43	6.3	0.13	0.43
D	290	1078	86	25	0.26	0.87
E	581	2157	172	100	0.52	1.74
F	1161	4313	344	400	1.04	3.47
G	2323	8627	689	1600	2.08	6.94
H	4645	17252	1378	6400	4.16	13.99
I	9290	34504	2756	25600	8.33	27.78

TABLE III
ESTIMATE OF MESSAGING REQUIREMENTS FOR FVCAM AS A FUNCTION OF HORIZONTAL RESOLUTION AND DECOMPOSITIONS.

potential client applications of a Climate-Modeling domain-oriented HPC machine. An icosahedral decomposition, such as that used in the BUGS climate model [20], constructs an approximation to a sphere by iteratively subdividing the edges of an icosahedron. This creates a geodesic grid; the grid points require 6-nearest neighbor connectivity for the most heavily used communication paths. Although it is more amenable to massive parallelism due to its very uniform grid distribution, its nearest-neighbor address calculations cause costly indirection in data accesses. A cube-sphere decomposition, such as that used in the GFDL model, maps the spherical globe onto the surface of a cube. This model is attractive because its logically rectilinear structure avoids indirection and allows for fast data access, although there are still some nonuniformities at the points of the cube. Its primary drawback is surface area distortion.

D. Design Requirements Summary

Based on the analyses above, we assume the following configuration to estimate required machine characteristics for ultra-high resolution climate simulation:

- 1.5km model (*I* mesh), 100 vertical levels..
- 1000 times faster than realtime simulation performance.
- *I* mesh (18432x11521x100 cells) discretized into 2 million horizontal subdomains (10x10 cells) which may be further subdivided in a hybrid vertical/horizontal manner.

In the previous section, we derived the fvCAM application resource requirements by extrapolating from the measured resource requirements of the climate code running at increasingly higher resolutions. Given these extrapolations, we present the minimum requirements for a system that would meet the application requirements for a 1.5km climate model simulation.

- Each horizontal subdomain must be computed at 5Gflop/s sustained in order to meet the 1000 times real-time requirement. The horizontal domains can in turn be decomposed across multiple processing elements within the socket. (For instance, if 10 processing elements are available within the socket, then each element need only supply 500Mflop/s sustained in order to meet the 1000x requirement.)
- 5GB/s local memory performance per domain in order to remain compute-bound rather than limited by memory bandwidth (1 byte per flop), to ensure sufficient memory bandwidth. (The 1 byte per flop ratio is based on our previous experience with fvCAM).
- The memory footprint of each subdomain is 5 Megabytes.
- The interprocessor communication requirements are 20MB/sec to each horizontal neighbor in the north,south, east, and west dimensions (10MB/s bidirectional to each neighbor).
- We assume the subdomains in the vertical dimension are co-located on the same socket or SMP node in order to reflect the extremely tight-coupling of computation and communication in that dimension.
- For inter-socket communication, we consider only the communication requirements between horizontal domains.

IV. Inclusively Identifying the Design Space

In this section, we discuss design concerns for a climate modeling domain-oriented machine, in order to understand what a "golden-standard" building block would provide. While the dimensions we list below are not orthogonal, in that decisions made for one dimension affect the decisions for the others, they serve to guide our discussion.

- *Granularity of Parallelism*: Balancing individual processor core complexity with the number of cores per chip.
- *Memory System*: Balancing the sizes on-chip memories and the number and width of access ports with the number of on-chip cores.
- *Communication Network*: Overall communication topology, including the complexity and connectivity of both the on-chip and inter-chip network.

A. Granularity of Parallelism: Potential Core Designs

Large scale computing is often not governed by uniprocessor characteristics, but rather by computing density: how many useful aggregate GFlops can be packed into a given power envelope, physical space, or operating budget. Table IV shows double precision peak floating point performance, chip power, and efficiency for a number of recent high-performance chips. The best examples reach only about 0.6 Gflops per watt in current technology, with the exception of the embedded XTensa processor from Tensilica extended with a SIMD floating point unit.

For this study, we divide the design space into three different design points. The AMD Opteron is used to represent a mainstream commodity building block for high-end system designs. The PowerPC 450 represents a semi-custom approach employing a relatively undifferentiated embedded core packaged in an SoC. The Tensilica XTensa is used to explore the capabilities of a lean, customized power-efficient core.

1) Commodity Core, the AMD Opteron: The AMD Opteron is a popular building block for HPC systems from the commodity clusters to the tightly-integrated XT3. This solution represents the classic "commodity approach" that depends upon a processor architecture that is targeted at a diverse set of server applications that range from technical computing to web-serving. The cost-efficiencies of leveraging high-volume commodity technologies is offset by the potentially lower computational efficiency of an architecture that is not as narrowly specialized to scientific application requirements. There is another benefit that should not be ignored in any design option: cores such as the Opteron that implement general-purposem, legacy-supportive ISA's also have an enormous software infrastructure. When using an x86 core, much of the software necessary to port a realistic, complex scientific model to the machine already exists. This greatly reduces the imlementation effort, and the existence of software infrastructure is a first-class design concern.

2) Embedded Core, the PPC450: BlueGene/P (BG/P) represents a more power-efficient alternative approach that achieves its performance through higher concurrency than mainstream cluster solutions. The core of the BG/P system is a System-On-Chip (SoC) based on the low-power PowerPC 450 embedded core. The PowerPC 450 is a dual-issue in-order CPU core present in many low-power embedded applications, which is optimized for scientific applications by customizing the SOC services that are built around it on the chip. Unlike the AMD solution presented above, the BG/P SOC includes all of the logic for the interconnection network. The BG/P approach offers well-documented improvements to the power efficiency for many scientific applications; albeit the breadth of applications that are well suited for the architecture are narrower than the typical commodity cluster implementations.

3) Semi-Custom Synthesized Core, the Tensilica XTensa: Whereas the BG/P solution begins with a generic embedded core and customizes the SOC logic that surrounds it to build a power-efficient building block for a system, Tensilica takes the embedded design philosophy one step further: customizing the design of the processor core so that it is custom-tailored to the computational requirements of the application. The Tensilica performance estimates are based on existing process technology and current-generation Tensilica system design tools. The customized design retains all of the programmability of a conventional processor, but can achieve its highest efficiency when applied to the problem for which it is designed. The motivation for using a simple, shallow-pipeline, in-order, configurable core such as XTensa is that in a massively parallel application, the granularity of parallelism is fine enough that a simpler core will suffice. But how simple should this core be? A complete exploration would explore the space of tradeoffs involving pipeline depths, VLIW issue widths, SIMD instruction widths, and sharing of functional units between cores. Improvement of any of these dimensions improves performance, but increases per-core area and power. Furthermore, fvCAM and other climate modeling codes are complex, and use different algorithms for modeling different sub-problems; no one function occupies more than a few percent of the running time. Therefore the problem of finding an optimal processor core is more difficult, since potentially many different memory-access, control, and compute patterns need to be considered. Single-thread performance on this core will surely pale in comparison to the more complex cores, since cache misses, pipeline and data hazards cannot be hidden as easily. SIMD instructions or multiple-issue pipelines may ameliorate the situation, but per-core area is always traded against total core count.

B. Memory System

As detailed in table II, kilometer-resolution climate modeling requires tens of terabytes of memory. Thus the design of the memory system can dramatically affect system performance. Table II indicates that byte-to-flop ratios are approximately 0.5 at low resolutions. The apparently favorable scaling of the byte-to-flop ratio, for example 0.0089 bytes of memory traffic per floating-point operation for the *I* mesh, is due to the overly restrictive CFL condition for the latitude-longitude decomposition. As we discussed in section III-C, the alternative decompositions are more uniform and are less restricted by CFL. Therefore we can expect that byte-to-flop ratios will be between 0.5 and 1.0.

In massively parallel decompositions for climate modeling, the working set size per processor is potentially only a few megabytes. Thus increasing the size of local memories could have significant impact, as the percentage of total domain that can be kept on-chip can be improved dramatically. For example, The IBM BlueGene/L chip includes 4 Megabytes of embedded DRAM [16] in a 130 nm process, the BlueGene/P chip 8 Megabytes, and the VIRAM1[3] chip from Berkeley contains 13 Megabytes of embedded DRAM, implemented in 180 nm process on a 325 mm^2 chip. More recent IBM embedded DRAM in 65 nm PD-SOI process [22] allows a 2Mb macro to fit in 0.665 mm^2 of die area and operate at 500MHz with 1.5 ns random access time. SRAM densities are substantially less, but easier to incorporate into processor designs; the IBM eDRAM density is approximately 3x IBM's SRAM density. This large quantity of on-chip memory could be used to keep the entire subdomain on-chip for a given processor. The area occupied by this DRAM would force other subdomains onto other chips, however, and increase communication costs. The very tight coupling required in the vertical dimension places a lower-bound on the amount of parallelism that should be packed onto a single chip; it is unlikely that the off-chip communication cost incurred by vertical communication between sockets is justifiable. According to our calculations

	STI Cell	BlueGene/P (PPC450 ASIC)	AMD Opteron "Barcelona"	Intel Xeon 5100	Intel Itanium2	Tensilica XTensa with SIMD Floating Point
DP Ops per Cycle per Processor	0.6 (each SPE)	4	4	4	4	4
Cycles per Second (GHz)	3.2	0.85	2.5	3.0	1.4	0.65
Processors per IC	8	4	4	2	1	32
Aggregate DP GFlops per IC	1.5	13.6	38.6	24	5.6	83
Approx. IC Power (Watts)	30	15	92	80	130	12
IC GFLOPS/Watt	0.5	0.91	0.42	0.3	0.06	7

TABLE IV
FLOATING POINT PERFORMANCE PER WATT OF SEVERAL MICROPROCESSORS

in Section III-B, 10x10 columns at 100 vertical layers would require approximately 50 MB of memory. In the 65 nm IBM eDRAM, this occupies about 133 mm^2 of die area. Due to cost, this is larger than a die likely to be used in a climate modeling machine. For reference, the BlueGene/L die is 11.1mm x 11.1mm, or 123.2 mm^2. Therefore, we must go off-chip for data access, and must size local memories to allow for data re-use and latency hiding through prefetching.

The other extreme, having very little on-chip memory, but packing the chip with as many small cores as possible, is an unlikely solution, but illustrates a point. These small cores cores would need to be designed specifically to tolerate huge off-chip memory latencies. A highly multithreaded architecture such as Niagara2 from Sun or a vector processor such as s the Cray X1E would be well-suited to this task. However, without local memories or caches, pin-bandwidth provides an upper bound to the amount of computation that can be performed per socket: the flop-to-byte ratio multiplied by the bytes-per-second bandwidth at the pins. The extent to which algorithmic data-reuse can exploit local memories and filter off-chip accesses is also the extent to which some amount of local memory can actually *increase* the number of cores feasibly on one chip. Additionally, requiring that all memory accesses go off-chip will consume more power, since driving pins requires more energy than reading on-chip SRAMs.

C. Communication Network

The simplicity of the 1-dimensional domain decomposition is attractive not only for the software development simplicity of its implementation, but also because of the correspondingly simple chip-interconnect. As explained in section III, however, more sophisticated decompositions are necessary for scalability. Since all practical decompositions of climate modeling utilize less than three dimensions, a 3-D Torus, such as those found on BlueGene or the Cray XT3 and XT4 systems is likely a poor match for kilometer-scale climate modeling. Ideally, a climate-modeling machine would have a network with a one-to-one correspondence between communication channels in the code and physical inter-processor communication channels.

For example, if the fvCAM model is used, the primary message passing network would be a 2-dimensional mesh.

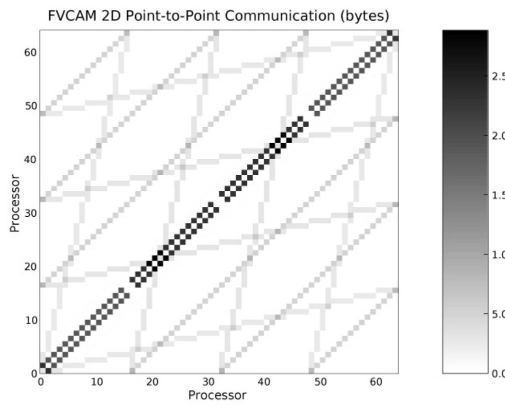

Fig. 3. Communication topology of fvCAM

We expect to have at least one complete vertical column per socket. Thus, there would be little congestion on the network for the normal bulk-synchronous communication on each timestep. However, the fvCAM model is not the only potential client of a climate-modeling domain-oriented machine, and communication patterns specific to the other potential domain decompositions should be supported at high-performance as well. This potential application diversity also makes it unclear precisely what the "ideal" communication network would be. Although the connectivity of the cube-sphere decomposition is very similar to the fvCAM decomposition, it contains points of irregularity. The implementation of the network would need to ensure that those points are supported gracefully, and do not become a bottleneck that cause global performance degradation. The icosahedral decomposition requires 6-nearest neighbor connectivity at most points, but also has irregular points. If this topology were implemented directly, it would support the other two degree-4 decompositions at the cost of unused hardware. However, if the degree-6 connectivity were folded on top of a degree-4 network, extra congestion and routing hops would adversely impact latency and bandwidth.

Previous work exploring the communication requirements of several applications [18], [19], including fvCAM, have revealed that the communication requirements of HPC applications are diverse, but commonly of low degree: a given processor communicates with a small number of the other processors in the system. Figure 3 illustrates the message-passing communication pattern of a 64×64 2-dimensional latitude-

[3]We did not include VIRAM1 in table IV because VIRAM1 does not support double-precision floating point. VIRAM1 achieves an efficiency of 1.4 GigaOps per Watt in 64b Fixed-point arithmetic.

longitude domain decomposition for fvCAM, using a 4-way SMP at each node. Notice the strong nearest-neighbor communication represented by the dark dual-diagonal bands, but also the weaker distant-neighbor communication represented by the lighter bands. No node communicates with more than 20 others, but this pattern is not isomorphic to any toroidal, mesh, or hypercube topology available on systems from Cray or IBM. Therefore the potential exists for a substantial decrease in cost and complexity necessary to meet the latency and bandwidth requirements of an application.

V. Exploring the Design Space

We have selected three candidate building blocks for our Climate Simulator system. The AMD core represents the commodity design point, the PowerPC 440 core represents a building block for an embedded SoC solution where the core is not customized, and the Tensilica XTensa represents a customized core. For the former two cases, we are able to collect benchmark data on existing systems. We can extrapolate the benchmark results to the full system scale to infer system component performance and use existing power consumption and performance figures to project the requirements of the full system. There are a very limited number of configurable parameters for these processor implementations. In the case of the Tensilica XTensa, we must evaluate the design space using Tensilica's emulation tools. Likewise, Tensilica's synthesis tools provide an accurate model of area and power requirements for the synthesized core design.

Here we discuss our approach to mapping the Climate problem onto the target system architectures. Next we discuss the computational, memory, and communication resources necessary for kilometer-scale climate modeling. Although in the 2011-timeframe in which we would expect to implement this machine, 45 nm technology would be available, in this section we compare technologies at the 65 nm node. We claim that the certainty gained in avoiding power and frequency extrapolation is more valuable than the potentially more accurate area estimate.

A. Mapping the Problem to Hardware

Exploitation of multi-core processors in an ultra-high resolution atmospheric general circulation model will require not only a two dimensional horizontal decomposition containing millions of subdomains, but also additional levels of parallelism. The tight coupling of multi-core processors to the entire socket's local memory permits a variety of options. We envision a parallelization strategy where a two dimensional horizontal decomposition is distributed among multi-core sockets. Within the horizontal subdomain, additional parallelization would be obtained by alternating between a vertical decomposition and a horizontal decomposition that might be implemented with message passing, at the loop level or a mixture of both techniques.

For a minimally sized subdomain of 10x10x10 cells, the effective compute rate that must be delivered for each subdomain is 0.5 Gigaflops in order to meet the simulation performance requirements. For simplicity, we assume that for any subdomain larger than these minimum dimensions, the processor's throughput per cell must exceed .5 Megaflops per cell. Since the overall computation rate is fixed at 10 Petaflops sustained, we can compute a lower bound on the number of processors necessary to implement our machine simply by dividing by the peak floating-point throughput of the processor by the throughput required per cell. This estimate is highly optimistic, since in our prior experiences, climate codes have run at approximately 5% peak throughput on high-performance processors, and at less than 2% peak on BlueGene/L. Once we have established the number of cells to be computed by each processor, the memory capacity and bandwidth requirements, inter-processor communication requirements, and the minimum number of processors in the system can also be computed.

B. Commodity Building Block: AMD Opteron

Here we discuss the characteristics of a peta-scale system built from our selected commodity building-block, the AMD Opteron. HPC systems built from commodity processors and interconnect have very few, if any, configurable features. Hence, here we summarize system features.

1) Core Design: The AMD Opteron is a 64-bit high-frequency, deeply pipelined, out-of-order superscalar processor. Designed to maximize single-thread performance on a wide range of applications, relatively little of the die area is devoted to floating-point or inter-processor communication hardware. The latest 65 nm revision of the Opteron core contains 2-wide double precision SIMD units, capable of executing two multiplications and two additions per cycle. Operating at 2.5 GHz, each core in a quad-core socket is capable of 9.6 GFlops/s, for a total per-socket peak floating point throughput of 38.4 GFlops/s. At this rate, each socket could potentially own 77 subdomains.

2) Memory System: Commodity memory systems for Opteron-based systems consist of DDR, DDR2, and in the future DDR3 in FBDIMM slots. Each slot can contain several gigabytes of memory, and provide approximately 10 GB/s of bandwidth per socket. The 65 nm quad-core Opteron chips include 3 levels of on-chip SRAM cache. Each core has a private 64K L1 and 512K L2 cache, and all four caches share a 2MB L3 victim cache. Cache coherence is maintained in hardware between the four cores, and between sockets in a multi-socket system. Since each socket will have at most 77 5MB subdomains, the memory required per Opteron socket 386 MB. We expect a byte-to-flop ratio of approximately 0.6 for the Opterons, so the local per-socket memory bandwidth needs to be at least 23 MB/s.

3) Interconnect: The 77 subdomains per socket decomposition would require 1,544 MB/s bandwidth from the network, which is readily achievable by commodity interconnect technologies.

C. Embedded Building Block: IBM PPC440

Although the BlueGene/P ASIC is not customizable per application, it has been designed specifically for power-efficient HPC computing. However, it attempts to be general-purpose within HPC. Again, here we cannot customize, and we just list system features.

1) Core Design: Although IBM has not disclosed the full details of the the PowerPC 450 cores in BG/P, it is expected to be similar to the PowerPC 440 core in BlueGene/L. PPC440 is a dual-issue in-order core designed for low-frequency operation. In BlueGene/P, cores operate at 850 MHz, and will feature a 2-wide SIMD floating-point unit. Implementing a fused multiply-add instruction, its peak double-precision throughput is 3.4 GFlops/s per core, or 13.6 GFlops/s per chip. This rate can support 27 subdomains per socket at peak.

2) Memory System: The PowerPC cores in the BlueGene ASIC have independent 32-KB L1 data and instruction caches. Each ASIC includes 8 MB of eDRAM. The 27 subdomains per socket will require 136 MB. With a byte-to-flop ratio of 0.9, each socket needs 12.2 GB/s of memory bandwidth.

3) Interconnect: BlueGene/P's primary message-passing network is a 3-dimensional torus, with logic on the SoC consuming slightly more area than one of the PowerPC cores. It provides 350 MB/s of bandwidth between adjacent network nodes, with a few hundred processor cycles of latency per network hop.

D. Customized Building Block: Tensilica XTensa

In this section we describe a prototype design of this processor that has been created using the Tensilica Instruction Extension language and the Xtensa Processor Generator. The processor generator translates a concise abstract description of instruction set features, and selection of interfaces and memories into a complete production-quality Verilog implementation and software development tool set. We postpone discussion of the software infrastructure until Section VI. For this application, the Tensilica tools were used to synthesize a variant of the XTensa processor that is optimized for the defined requirements of the climate problem without attempting to exploit idiosyncratic computation characteristics via application-specific instructions. A final processor configuration would reflect significant application analysis and embody optimization to omit less useful features and add instructions and interfaces discovered as particularly useful to overall throughput.

These processors assume implementation in commodity standard cell circuits generated by standard logic synthesis and cell placement and routing flows. They assume local memory systems built from multiple banks of single-port on-chip SRAM, combined with regional memory systems built from multiple channels of commodity high-performance off-chip DRAM, typically Rambus XDR.

We now walk through elements of the resulting synthesized design, which is summarized in Table V.

1) Exploration of the Core Design: A lean processor with general support of both integer and single-precision floating point data types, and a pipelined static dual-issue implementation can be built in less than 100K gates of logic (about 0.3 mm^2 of logic area in 65nm process technology) Even with double-precision floating point and multiple operations per cycle, the entire processor core consumes less than 1 mm^2 of logic. Specifically, the synthesized core contains two double-precision floating point pipelines, 128b local memory system, closely-couple block memory transfer engine and high-speed nearest neighbor communications links implemented in 65nm standard cell technology will dissipate less than 150mW at 650MHz, yielding roughly 15 GFLOPS (peak) per watt for the processor subsystem and about 7 GFLOPS (peak) per watt for the full chip. A cluster of 32 such subsystems could fit on a 200 mm^2 chip, implemented in commodity 65nm silicon technology, providing a 12W chip capable of executing more than 80 double-precision GLFOPS peak. 7 GFLOPS per watt is more than a factor of ten better than traditional high-end processors, but also a factor of 16 higher density over the traditional approach due to the small silicon area of each core. Each socket could support 172 subdomains.

2) Exporation of the Memory System: Our proposed 32-processor chip supports 4 or 8 XDR DRAM Channels (from Rambus) and Direct Buffered DRAM to meet the memory bandwidth defined by the application requirements section using modest board area, component count and power dissipation. XDR II, for example, now promises 6.4 Gbit per second per pin, and even commodity XDR implementation offers 6.4-9.6 Gbytes per second per 16-bit device. The on-chip interface circuitry consumes modest area and power so multiple DRAM channels per chip are quite feasible. Four to eight XDR channels can supply 25-75 Gbytes per second of bandwidth (sustained for optimal referenced patterns). This substantial total off-chip memory bandwidth is one key contributor to the high potential performance of each chip and the system as a whole. The DRAMs, DRAM-processor transfers, and memory controllers together are expected to consume well above half the power in the system. With 8 channels, a group of four processors share a local bus, which is then connected to an 8x8 cross-bar to the eight DRAM channels. Each processor has direct access to the 8 DRAMs and to the local data RAM space of each of the other 31 processors. With buffered writes, the transfer latency between processors is often completely hidden, though these transfers still consume shared bus bandwidth. 172 subdomains would require 864 MB, and with a byte-to-flop ratio of 0.6, would require 51.2 GB/s of bandwidth.

The processor includes a data cache but structured grid problems may also take good advantage of block data movement, especially for pre-fetching of data from off-chip DRAM into local RAM and return of data blocks previously written to local RAM. Recent studies by our group [25] shown substantial performance efficiency benefits from application-managed block data movement. Software controlled memories, such as those used in Cell allow dramatic simplification of the whole memory hierarchy, compared to multi-level cache subsystems. This is a major contributor to the energy efficiency of the chip. We propose a direct memory access (DMA) engine directly controlled by the instruction set of the processor. A single processor instruction can specify the source and destination address, the size and direction of transfer, and the number of the DMA channel to use. The state of each DMA channels is directly available the control-flow instructions of the processor for efficient block transfer completion handling.

This lightweight DMA supports 8 or 16 DMA active transfers per processor and can move data directly between the on-chip bus and the local data memory for each processor. This simple programming model and low overhead may make the necessity of programmer-managed data movement tolerable.

3) Exploration of Interconnect: The ASIP would use existing design libraries to implement a high-speed regional interconnect, on-chip and off-chip, to create a 2-D computing mesh optimized to the communication topology requirements of the climate model described in the previous section. This approach is particularly effective on problems where the pattern of memory references is not heavily data-dependent and large block data transfers between off-chip and the processors can reduce memory reference latency. Our 2-D mesh is created via four incoming and four outgoing data links for each processor and is implemented in two variants.

In the first variant, the processor's instruction set includes PUSH operations to move a value directly from the integer or floating point register directly to the interface queue, and POP operation to move data from the interface queue to the integer or floating point register. The processor stalls on PUSH if the queue is full and stalls on POP if the queue is empty. Data communication between two processors is therefore automatically synchronized and highly efficient, requiring only one instruction on both the sending and receiving processor for each data word transferred. This mechanism could be used to minimize the overhead of the single-variable messages mentioned in Section III-C.

In the second variant, the same DMA engine used for DRAM to local RAM transfers can also move data blocks between the local RAMs of different processors. This can include both movement through the cross-bar switch used to give all processors access to all DRAM attached to that, and movement of data across dedicated local links. Use of the cross-bar switch consumes some of the bandwidth used for DRAM transfers, but gives access to all the processors on the chip.

The basic memory and message transfer structure of the DMA-based variant is shown in Figure 4.

Inter-chip communications links follow the structure of on-chip links as closely as possible. The links are implemented using commodity PCI Express interfaces, routed both within boards and between boards. Each chip implements 24 bidirectional PCI channels. Figure 4 also shows one control processor implemented on each chip. This processor collects statistics and trace information on all of the memories, processors and message paths for that chip. It manages the boot process for the others, the disabling of processors with hardware faults, and coordinates debug for tasks on all the processors.

One of the most notable characteristics of this processor is low power dissipation. A single core, with its memories is projected to dissipate about 150mW at 650MHz when implemented in commodity 65nm technology. This permits a 32-processor array chip with a total power dissipation of about 12 watts, while executing about 80 GFLOPS of double-precision code. The dense packing of low-power processors and memories allows high computational density at the chip, board and rack level. A 32 chip board delivers more than

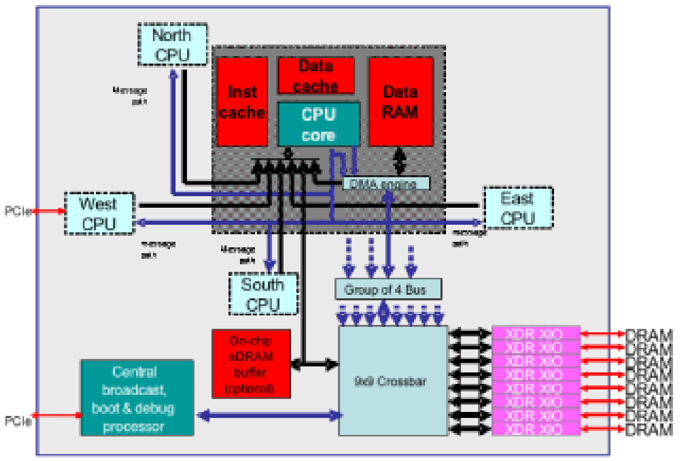

Fig. 4. Structure of Memory Bus and On-Chip Communications

2.5 Teraflop and 1-2 Terabytes per second of distributed main memory bandwidth in less than 1000W. A commodity 25KW rack will hold 24-32 such board for 60-80 Teraflops per rack using existing 65nm silicon fabrication processes.

Although we do not fully exploit these capabilities in this paper, the Tensilica environment also lends itself to rapid exploration of alternative instruction sets, pipelines and interfaces. A variant can be specified, generated and simulated in less than an hour, so a wide range of possible architectures can be quantitatively compared early in the system specification process. We believe this is a source of untapped power-efficiency optimizations that can even further improve the power-efficiency of this core design for this application.

Table V outlines the general characteristics of our prototype processor core, and Table VI compares our three design points. Figure 4 illustrates some of the larger system capabilities. Given the current depth of our design space characterization, and our understanding of the requirements of kilometer-scale climate modeling, this core provides all the necessary computational power at an order of magnitude more power efficiency than other options. As such, it is the best option of the potential candidates.

VI. Deploying the ASIP

We have mentioned previously that pre-existing software infrastructure needs to be a first-class design consideration in any new HPC machine design. Regardless of the performance potential of the machine, the software development effort required to port realistic scientific applications to the new environment can be an insurmountable barrier. The dynamics code in fvCAM is 93,407 lines of Fortran 90 code. The BUGS6 atmospheric model is 47,382 lines of fortran 90, in addition to over 100,000 lines of code in utilities, ice, ocean, and land models. Each model contains hundreds of subroutines, dozens of modules, and relies upon external libraries for communication, input/output, and other system interaction. Although optimization and tuning for the new machine may require re-writing many thousands of lines of code, it is absolutely infeasible to re-write the entire system

Instruction Set	Data Types	8,16,32 bit integer 1 bit boolean conditions 64 bit IEEE Floating Point
	Register Files	16 x 32 bit integers (in 32-entry windowed register file) 16 x 1 bit boolean 1 x 128 bit 2-way SIMD Floating Point
	Fully-Pipelined Execution Units	3-Way static superscalar instruction issue: Integer load/store (slot 0) FP 64/128b Load/Store (slot 0) FP Convert (slot 0) Integer/FP branch (slot0+slot1) Integer op (both slot 0 and slot 1) FP add/sub/mull/mul-add/mul-sub/reduction add (slot 2) FP Compare (slot 2) FP conditional move (slot 2)
Memory System	Instruction	32KB 2-way assiciative cache, with 64B line refilled over 128b bus, 64b fetch
	Data	16KB 2-way associative cache with 64B line refilled over 128b bus, 128b fetch 64KB local data RAM, 128b fetch. Implemented as 4 banks, for dual-porting with DMA local engine
Implementation Estimate (65nm)	Area	$1.0mm^2$ ($0.8mm^2$ core + $0.2mm^2$ local DMA engine). Gate count: 225K
	Typical Power	Core: 130 μW/MHz Local Memory: 100 μW/MHz Typical operating power: 150 mW @ 650 MHz
	Operating Frequency	650 MHz worst-case temperature, voltage, and processing
Software	Sofware development tools	C/C++ compiler automatically extended for full instruction set Eclipse-based integrated development Pipeline-accurate simulator Multiple processor simulation builder
	Runtime Software	Variety of real-time operating systems and Linux Communications API for message passing, DMA, shared memory, and locking Interactive MP debug and trace capture

TABLE V
EXAMPLE APPLICATION-SPECIFIC HPC PROCESSOR ARCHITECTURE AND IMPLEMENTATION SUMMARY

Name	CPU	Clock Speed (Ghz)	Flops/ Clock/ Core	Peak Core (GF/s)	Cores/ Scket	Power/ Scket (W)	Sub-domains/ Scket	Mem/ Scket (MBs)	Mem/ BW[1] (GB/s)	Peak Bytes/ Flop	Netwk BW[2] (MB/s)	Total Sockets (10^3)	Total Power (MW)	Cost (M$)
AMD	Opteron	2.5	4	9.6	4	230[6]	77.2	386	23	0.6	1,544	260	57	520[3]
BG/P	PPC450	0.85	4	2.8	4	15	27.2	56	5.5	0.9	544	740	20	722[4]
Tensilica	Custom	0.65	4	2.7	32	22	172.8	864	51.2	0.6	3,450	120	2.5	75[5]

TABLE VI
SUMMARY OF CANDIDATE SYSTEM REQUIREMENTS

from scratch. These programs have taken years to develop, and have been proven correct and relatively bug-free to the community through years of use. Not only would re-implementing an entire climate modeling system for a machine without operating system, compiler, and library infrastructure take prohibitively long, but the resulting system would need several years of probation before its results are considered credible. Moreover, in the time such an effort would require, it is likely that a system with shorter time-to-solution will have already produced the necessary scientific results. For IBM BlueGene based on PowerPC cores or Cray machines based on AMD Opterons, all the necessary software infrastructure exists. The commodity ISAs of these processors are capable of running full-featured operating systems, have well-tested suites of system software, and high-quality compilers for any commonly used programming language. Besides the hardware procurement cost benefit of commodity-based systems, this software development advantage would be a strong argument against a domain-oriented configurable machine.

Luckily, there is a solution in the case of the Tensilica XTensa cores. The same tools that produce a customized RTL description of the XTensa core also produces code necessary to compile gcc, Open64 compilers, and a complete Linux operating system and software environment. We have compiled gcc as an end-to-end Fortran90 to XTensa binary compiler, linking with runtime libraries compiled by the Tensilica-provided C compiler. The Open64 Fortran90 frontend can emit the same Whirl2 intermediate language used by the Tensilica version of the Open64 C and C++ compilers. Tensilica does not currently ship an end-to-end Fortran compiler because the embedded applications which typically utilize XTensa cores are written in C or C++. The software effort required to integrate the

[1]Minimum required local memory bandwidth per socket.

[2]Minimum required nearest neighbor interconnect bandwidth per socket.

[3]Cost of node and memory only, does not include interconnect. Estimated conservatively at $2,000 per node.

[4]Pricing for BlueGene/P estimated at $1M per rack.

[5]Cost of custom chips design and fabrication, memory, raw hardware for the system, and interconnect.

[6]Power of 230W measured on a dual-socket AMD Opteron node.

Open64 Fortran90 frontend will be minimal, however. Without the need to develop the entire software system from scratch, the task of porting a large software base to an XTensa-based system becomes tractable.

The only necessary piece of system software which does not yet exist and could not be compiled for a Tensilica-based massively parallel machine is an MPI implementation, since the interconnection network would be highly customized. In this case, the message-passing infrastructure would need to be developed. If the application required a complete MPI system for operation, this would indeed be a daunting task. Since fvCAM and other climate codes only use a small, simple subset of the MPI standard, implementing the necessary functions should be a very manageable task.

VII. Conclusion

We evaluated a number of processor alternatives for a climate modeling system. In particular, we investigated the advantages of using a customized processing element, memory, and interconnect to reduce system cost and power requirements compared to mainstream designs.

The degree of parallelism is limited at the resolutions of models currently in use. However, at horizontal resolutions of one kilometer, sufficient parallelism exists to utilize millions of processor cores. Recent technological trends indicate that the rate of processor clock speed changes is slowing. Moore's Law regarding the density of transistors is expected to hold through the next decade [7], [8] but due to power considerations chip designers are opting to increase overall per-socket performance by adding more processing cores rather than increasing clock speed [9]. These architectural trends towards increased parallelism and multi-core chip designs are driven primarily by power-efficiency considerations.

Our design study shows that further power-efficiency gains can be realized through customized processor design in line with the embedded-systems design philosophy. This portends an alternative approach to high end computing system design where efficiency can be gained through hardware designs that are customized around the application rather than the other way around. While the costs of custom hardware design may not be cost-effective for the mid-range problems, the approach may prove essential for the handful of applications that are poised to take advantage of future ultrascale systems. Without detailed attention to power efficiency concerns, the energy costs for operating such systems is likely to create a hard ceiling for practical ultra-scale computing in the future.

The combination of ultra-low power processing cores with high bandwidth communications hardware and software enables an order-of-magnitude improvement in the scale and energy efficiency of high performance computing systems for suitably structured problems. A complete hardware system can be built with commercially available configurable processors, commodity interface IP, foundry processor, board and rack technology. The software development tools are automatically generated together with the processor hardware itself, implementing a high-performance general-purpose instruction set architecture and code development system. A complete system with sustained performance to meet the needs of the scientific application can be delivered within a practical power and space envelope, whereas the power and space required from conventional approaches would be difficult to field well into the future.

These domain-oriented supercomputers exploit the growing energy-efficiency gap between embedded systems processors and traditional high end processors. This category offers ten times improvement in performance per watt, and per dollar than systems based on conventional processors. This approach runs contrary to the conventional wisdom of HPC, but reflects a simple set of observations:

- The general HPC application workload is diverse, and so appears to require either a diverse range of machine architectures or a universal general-purpose architecture to support their requirements.
- At the petaflops and exaflops scale, however, there are few candidate problems. General-purpose machines appear both too expensive and too power-hungry to efficiently support those few applications.
- For a given power or cost level, you can do a much better job by customizing the design around the requirements of the problem. This approach is not only feasible, but creates a more efficient system that delivers higher sustained performance.

Semi-custom machines built from configurable processors offer an attractive alternative to mainstream designs. As these processors lack the power-hungry complex out-of-order instruction fetch, schedule, and retire logic in high-performance commodity processors, the potential efficiency of these machines, measured in GFlops per Watt, outstrips commodity machines by more than an order of magnitude. Since the total cost of ownership of a petascale includes not only the cost of hardware, but also of supplying power, the NRE of a semi-custom design is amortizable. For these reasons, configurable processors present an attractive approach to solving petascale problems

Acknowledgment

Many thanks to Dror Maydan, Bill Huffman, and Grant Martin of Tensilica. Also thanks to Michael Wehner of Lawrence Berkeley National lab for his expertise in understanding the climate model.

References

[1] S.Y. Borkar *et al.*, *Platform 2015: Intel Processor and Platform Evolution for the Next Decade Whitepaper*, 12 pages. 2005.

[2] M. Gries and K. Keutzer Building ASIPS: The MESCAL Methodology *Springer*, 2005.

[3] K. Asanovic and R. Bodik and B.C. Catanzaro and J.J. Gebis and P. Husbands and K. Keutzer and D.A. Patterson and W.L. Plishker and J. Shalf and S.W. Williams and K.A. Yelick, The Landscape of Parallel Computing Research: A View from Berkeley, *EECS Department, University of California, Berkeley, Technical Report*, UCB:EECS-2006-183, Dec. 2006.

[4] E. Killian and C. Rowen and D. Maydan and A. Wang, *Hardware/Software Instruction set Configurability for System-on-Chip Processors, in Proceedings of the 38th Conference on Design Automation (DAC '01)*, 2001, pp. 184188.

[5] John L. Henning. PSPEC CPU2000: Measuring CPU Performance in the New Millennium. *IEEE Computer*, pages 28–35, 2000. http://www.spec.org.

[6] Borivoje Nikolic Jan M. Rabaey, Anantha Chandrakasan. Integrated circuits, a design perspective. In *Prentice Hall, Second Edition*, 2003.

[7] S. Borkar. Design challenges of technology scaling. *IEEE Micro*, 19(4):23–29, Jul-Aug 1999.

[8] P. P. Gelsinger. Microprocessors for the new millennium: Challenges, opportunities, and new frontiers. In *International Solid State Circuits Conference, ISSCC*, pages 22–25, 2001.

[9] John L. Hennessy and David A. Patterson. *Computer Architecture: A Quantitative Approach; fourth edition*. Morgan Kaufmann, San Francisco, 2006.

[10] S.-J. Lin and R. B. Rood. An explicit flux-form semi-lagrangian shallow-water model on the sphere. *Q. J. R. Meteorological society*, 123, 1997.

[11] C. Rowen. Reducing SoC Simulation and Development Time. *IEEE Computer*, December 2002.

[12] D. Skinner. Integrated Performance Monitoring: A portable profiling infrastructure for parallel applications. In *Proc. ISC2005: International Supercomputing Conference*, Heidelberg, Germany, June 21-24, 2005.

[13] P. Ienne. *Customizable Embedded Processors: Design Technologies and Applications* Second Edition, Morgan Kaufmann, 2006

[14] J. Nurmi *Processor Design: System-On-Chip Computing for ASICs and FPGAs* First Edition, Springer, 2007

[15] J. Nurmi *Interconnect-Centric Design for Advanced SOC and NOC* First Edition, Springer, 2004

[16] S. S. Iyer, J. E. Barth, Jr., *et al. Embedded DRAM: Technology platform for the Blue Gene/L chip* IBM Journal of Research and Development, Volume 49, Number 2, 2005

[17] J. Gebis, S. Williams, D. Patterson, C. Kozyrakis *VIRAM1: A Media-Oriented Vector Processor with Embedded DRAM* In *Proc. DAC2005* Design Automation Converence, 2004, San Diego, California, USA.

[18] Shalf, J., Kamil, J., Oliker, L., Skinner, D. *Analyzing Ultra-Scale Application Communication Requirements for a Reconfigurable Hybrid Interconnect* In *SC '05*, Proceedings of the 2005 ACM/IEEE conference on Supercomputing

[19] Kamil, S., Shalf, J., Oliker, L., Skinner, D. Understanding Ultra-Scale Application Communication Requirementsa In *IISWC 2005*

[20] Randall, D.A., Ringler, T.D., *et al. Climate Modeling with Spherical Geodesic grids* Computing in Science & Engineering, Vol. 4, Issue 5, Sept-Oct 2002, Pages 32-41

[21] Alex Krasnov, Andrew Schultz, *et al. RAMP Blue: A Message-Passing Manycore System in FPGAs*, In *FPL 2007*, International Conference on Field Programmable Logic and Appplications, Amsterdam, Holland, August 2007

[22] John Barth, William Reohr, *et al. A 500 MHz Random Cycle 1.5ns-latency, SOI Embedded DRAM Macro Featuring a 3T Micro Sense Amplifier* ISSCC 2007, Session 27

[23] Umesh Gajanan Nawathe, *et al. An 8-Core 64-Thread 64b Power-Efficient SPARC SoC* ISSCC 2007, Session 5

[24] Chris Rowen *Engineering the Complex SoC: Fast, Flexible Design with Configurable Processors* Prentice Hall, 2004

[25] S. Williams, J. Shalf, L. Oliker, S. Kamil, P. Husbands, K. Yelick, *Scientific Computing Kernels on the Cell Processor* International Journal of Parallel Programming (IJPP), 2007.

[26] N. R. Adiga *et al.* /it Blue Gene/L Torus Interconnection Network IBM Journal of Research and Development, Volume 49, Number 2, 2005