# A Fast Architecture-Level Thermal Analysis Method for Runtime Thermal Regulation

Sheldon X.-D. Tan, Lin Jiang, Pu Liu, Wei Wu, and Murli Tirumala

***Abstract*** *—As power consumption and the corresponding heat dissipated on a die grow rapidly, efficient on-chip temperature regulation becomes imperative for today's high performance microprocessors. Temperature tracking based on the on-chip thermal sensors is not sufficient as the temperature hot spots keep changing with the load. One way to mitigate this problem is by means of software sensors, where temperature of any location is computed based on realtime power information and calibrated with the physical sensors. In this paper, we present a very efficient numerical thermal analysis method, which is suitable for fast temperature tracking and runtime thermal regulation. The proposed method, called FEKIS, combines two existing numerical techniques: extended Krylov subspace reduction technique to reduce the thermal circuit complexity and large-step integration method to exploit the piecewise constant power input traces, which is typical in the power traces at the architecture level. Experimental results show that FEKIS archives 10X speedup over recently proposed thermal moment matching technique [1], [2] and the precise time-step integration method only, and three orders of magnitude faster than the traditional numerical integration method with high accuracy.*

***Keywords*** — Thermal analysis, Power, Architecture, Model reduction.

Some preliminary results of this paper appeared in *ACM Great Lakes Symposium on VLSI (GLSVLSI)*, 2008. This work is supported in part by NSF grant under CCF-0448534, NSF Grant under CCF-0541456, UC Micro Program #07-101 via Intel Corporation.

## 1 INTRODUCTION

Higher temperature has significant adverse impacts on chip performance and reliability. This problem becomes more severe as VLSI technology scales to the nanometer ranges. The exponential power density increase will in turn lead to average chip temperature to raise rapidly [3]. Excessive on-chip temperature can cause many severe problems such as reduced reliability of chips, elevated cool cost of the packaging. One way to mitigate this problem is by means of online temperature regulation or dynamic thermal management (DTM), which dynamically reduces the temperature of some hot units in a chip via a suite of techniques such as activity migration, local toggling, dynamic voltage/frequency scaling [4], [5]. Furthermore, recent studies show that architecture level thermal management at small performance degradation cost can significantly reduce the packaging costs typically designed for worst cases [5], [6], [7].

One of the most critical aspects of thermal modeling and simulation for DTM is to efficiently capture the temperature profile changes at regular execution intervals caused by the variations of the power consumption from runtime applications at the chip architecture level or the operating system level. Physical thermal diode-based sensor has problems ranging from imprecision, delay and space overhead for hardware implementation [4], [8], [9]. These sensor noises could degrade DTM performance significantly from conservative triggering of DTM [5].

One viable alternative solution to this problem is by means of fast on-chip thermal estimation technique in software form with help of physical sensors for effective DTM application. An architecture level thermal modeling and simulation tool HotSpot [5] exploited and studied different DTM techniques in regulating microprocessor operating temperature for representative benchmark programs. However, HotSpot uses Runge-Kutta (RK) method to solve the linear differential equation and it can be slow for very long power traces from modern benchmark programs which have tens to hundreds of billions of instructions. Recently more efficient thermal simulation methods based on moment matching techniques have been proposed [1], [2]. These methods compute the transient temperature changes based on frequency domain moment matching concept. But those approaches require the evaluation of exponential functions from a closed form response expressions at each simulation intervals, which can be slower at runtime simulation.

In addition to online temperature tracking, fast thermal estimation can be used for architecture level physical designs, testing during the early stages of the chip design processes. In those applications, fast thermal evaluation for long power traces are required for many thermal-aware optimization applications such as thermal floorplanning [10], thermal-safe test scheduling [11] or thermal-aware testing [12]. Typically thermal simulation is a core function, and will be called thousands of times. So the time efficiency becomes very critical.

In this paper, we propose a new, fast extended Krylov subspace integration simulation method, FEKIS, for runtime temperature monitoring and fast off-line estimation. The new method combines two existing numerical techniques, Krylov subspace reduction technique and precise time-step integration method. It exploits the fact that the typical power trace pattern shown in Figure 1, can be approximated by the average power of each step to accurately determine the temperature trend [1]. Our new contributions are as follows: (1) We apply a precise integration method (PIM)[15], which can be very fast for linear systems with a very long fixed time step and is driven by constant inputs in each fixed time step. This is typical power input patterns at the architecture level as many programs

exhibit long instruction execution intervals, which has the same power consumptions owning to loops and regular program patterns [13], to save computation cost at runtime. (2) We apply Krylov subspace method to reduce the circuit matrices into smaller ones to speed up the simulation by roughly one order of magnitude (3) We test FEKIS on the architecture thermal circuit of a Pentium 4 Northwood core under 22 SPEC2K [14] benchmarks. We show that FEKIS becomes more efficient when finer granularity thermal models are used.

The rest of the paper is organized as follows: In section 2 we introduce the PIM algorithm for thermal simulation. Section 3 describes the model order reduction technique for special pulse input. We summarize the whole FEKIS algorithm in section 4. The experimental results are presented in section 5 to validate our method. Finally, section 6 concludes this paper.

## 2 FAST THERMAL PRECISE INTEGRATION METHOD

The PIM algorithm computes the complete response of a linear system including zero-input and zero-state responses by taking second order Taylor expansion and using so called 2N algorithm to compute the exponential matrix required in the evaluation. The advantage of the PIM is that it allows a large integration step (any larger theoretically) without much error for a constant input. This is not true for the traditional integration methods like Back Euler.

### *2.1 PIM for linear circuit system*

For the equivalent thermal RC circuit, we can use modified nodal analysis (MNA) to formulate the thermal circuit ordinary differential equation in matrix form as:

$$G\mathbf{x}(t) + C\dot{\mathbf{x}}(t) = B\mathbf{u}(t), \qquad (1)$$

where $G \in R^{n\times n}$ is the circuit conductance matrix which is stamped with thermal resistors, $C \in R^{n\times n}$ is the circuit capacitance matrix which is stamped with thermal capacitors, $\mathbf{x} \in R^{n}$ is the vector of node temperature, $\mathbf{u} \in R^{p}$ is the vector of independent power sources, and $B \in R^{n\times p}$ is the input select matrix mapping the input power sources to the internal nodes. *n* and *p* are the number of nodes and power sources.

For the architecture level thermal model, matrix *C* in (1) usually satisfies with two conditions, nonsingular and diagonal. So we can easily get its inverse matrix $C^{-1}$ and transpose (1) into standard formulation:

$$\dot{\mathbf{x}}(t) = -C^{-1}G\mathbf{x}(t) + C^{-1}B\mathbf{u}(t). \qquad (2)$$

The general solution of (2) can be written as:

$$\mathbf{x}(t) = e^{-C^{-1}Gt}\ \mathbf{x}(0) + \int_0^t e^{-C^{-1}G(t-\tau)}C^{-1}B\ \mathbf{u}(\tau)d\tau. \qquad (3)$$

If we assume that the input power **u** becomes a constant during the fixed sampling interval Δt, the integration equation (3) can be represented as:

$$\mathbf{x}(i+1) = A\mathbf{x}(i) + AT\mathbf{u}(i) - T\mathbf{u}(i), \qquad (4)$$

where

$$A = e^{-C^{-1}G\Delta t} \quad and \quad T = -G^{-1}B. \tag{5}$$

Furthermore, equation (4) can be written as:

$$\mathbf{x}(i+1) = A[\mathbf{x}(i) + T\mathbf{u}(i)] - T\mathbf{u}(i). \tag{6}$$

Therefore we can calculate the transit response at any time step by using the previous time step value.

### *2.2 $2^N$ algorithm for exponential matrix approximation*

In order to get the accurate results, the key is to calculate the exponential matrix A. There are many algorithms to compute the exponential matrix. Here the so called $2^N$ algorithm [15], [16] is applied and cooperated with second order Taylor expansion.

First we rewrite the matrix A as following form.

$$A = e^{-C^{-1}G\Delta t} = (e^{-C^{-1}G\Delta t/m})^m = (e^{H\tau})^m \tag{7}$$

where

$$H = -C^{-1}G \quad and \quad \tau = \Delta t/m. \tag{8}$$

We select *m* as an integer to power of 2. If N = 20 and $m = 2^N = 1048576$, $\tau$ becomes a very small number. After second order Taylor expansion, we have

$$e^{H\tau} \approx I + H\tau + \frac{(H\tau)^2}{2!} = I + A_a, \tag{9}$$

where $A_a = H\tau + (H\tau)^2/2!$. Since $\tau$ is selected to be a very small number, second order approximation can achieve very accurate approximation.

Because the elements of matrix $\boldsymbol{A_a}$ are very small compared with the identity matrix ***I***, we can not directly compute ***A*** by computing $\boldsymbol{(I + A_a)}$ to the power of *m*. However, since we select *m* as an integer to the power of 2, we can easily derive the following relationship.

$$\begin{aligned} A &= (I + A_a)^m \\ &= (I + A_a)^{2^N} \\ &= [(I + A_a)(I + A_a)]^{2^{N-1}} \\ &= [I + (2A_a + A_aA_a)]^{2^{N-1}} \end{aligned} \tag{10}$$

Then by using a variable replacement process, $A_a = 2A_a + A_aA_a$, we can compute ***A*** recursively. We

calculate matrix ***A*** by using only *N* times of matrix multiplication as following:

$$\begin{aligned} &for \;\; i \leftarrow 1 \;\; to \;\; N \;\; do \\ &\quad A_a \leftarrow 2A_a + A_a A_a \\ &A \leftarrow I + A_a \end{aligned} \tag{11}$$

We summarize the PIM algorithm in Figure 2.

## 3 EXTENDED KRYLOV SUBSPACE FOR MODEL ORDER REDUCTION

The PIM algorithm we described above can achieve faster simulation than some transient integration algorithms, such as first order Backward Euler (BE), by taking advantage of the fixed time step. It is very suitable for architecture level thermal model simulation because of its small size (around 100 nodes). Original thermal model from HotSpot is quite basic, which only capture 1D heat flow into and within the thermal package, and 2D lateral heat flow among pipeline units. Recently more detailed thermal models are proposed in [17], [7], which are partition based grid models. The grid models can provide more accurate information, but it also leads to larger circuit size. When the matrix dimension becomes larger, the efficiency of the PIM algorithm is degraded. The reason is that it lose the sparsity of the thermal matrices. The matrix A we get is a dense matrix, where all elements are nonzero generally.

In the Pentium 4 Northwood floor-plan example, there are only 488 nonzero elements and 82 nonzero elements in thermal matrix *G* and *C* with dimension of 82. As we know, the computational complexity of sparse operations is proportional to the number of nonzero elements in the matrix. Computational complexity also depends linearly on the row size *m* and column size *n* of the matrix, but is independent of the product $m * n$, the total number of zero and nonzero elements. However, the matrix ***A*** for PIM simulation has $82^2 = 6724$ nonzero elements which may not more efficient than directly solving sparse format differential equation. Fortunately this problem can be mitigated by taking model order reduction (MOR) to the thermal model. For instance, we reduce the model dimension from 82 to 10, and reduced matrix $\hat{A}$ at most has $10^2 = 100$ nonzero elements compared with 6724 in original matrix ***A***.

### *3.1 Review of extended Krylov subspace method*

In this section, we are going to review moment matching based MOR technique [18], the extended Krylov subspace method (EKS) [19], for thermal simulation.

Formatted: Indent: First line: 0", Line spacing: single

In frequency domain, (1) can be written as

$$(G + sC)X(s) = BU(s). \tag{12}$$

If we can represent the input power signal *U(s)* as

$$U(s) = \mathbf{u}_0 + \mathbf{u}_1 s + \mathbf{u}_2 s^2 + \cdots + \mathbf{u}_i s^i + \cdots, \tag{13}$$

by taking Taylor expansion at $s = 0$ for $X(s)$, we have

$$(G + sC)(\mathbf{m}_0 + \mathbf{m}_1 s + \mathbf{m}_2 s^2 + \cdots) = B(\mathbf{u}_0 + \mathbf{u}_1 s + \mathbf{u}_2 s^2 + \cdots). \quad (14)$$

So we can derive the recursive relationship for moments generation in (15).

$$\begin{aligned} G\mathbf{m}_0 &= B\mathbf{u}_0 \\ G\mathbf{m}_1 + C\mathbf{m}_0 &= B\mathbf{u}_1 \\ &\cdots \\ G\mathbf{m}_{r-1} + C\mathbf{m}_{r-2} &= B\mathbf{u}_{r-1} \end{aligned} \quad (15)$$

Then $r$-order Krylov subspace cab be constructed as

$$K_r(G, C, B, U) \doteq span\{\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_{r-1}\} \quad (16)$$

Those moments generated in (15) are explicit moments which are easily suffering the well known numerical stability problem when higher order moments are required for better accuracy of reduced order model [18]. Fortunately EKS presented an explicit moments generation method by applying Arnoldi algorithm to orthonomalize the basis of the Krylov subspace $K_r$.

The basic idea is that we compute the first moment $\mathbf{m_0}$ by solving the first equation in (15). Then we can have the first basis $\mathbf{v_0}$ by normalizing $\mathbf{m_0}$ directly.

The second basis $\mathbf{v_1}$ can be computed as follows. First we calculate the implicit moment $\tilde{\mathbf{m}}_1$ instead of the explicit moment $\mathbf{m_1}$ based on previous moment $\mathbf{v_0}$ instead of $\mathbf{m_0}$. Next we orthogonalize $\mathbf{m_1}$ with $\mathbf{v_0}$ to get $\tilde{\mathbf{v}}_1$. Finally we normalize $\tilde{\mathbf{v}}_1$ to $\mathbf{v_1}$ as the second basis vector.

Similarly we can generate higher order basis $v_2$, $v_3$, $\cdots$ by taking the same process as we mentioned above. The new generate basis is required to orthogonal to all existed basis.

Then we use matrix $V$, the matrix representation of Krylov subspace $K_r$, to perform model order reduction by taking congruence transformation to thermal matrices $G$, $C$, and $B$.

$$\hat{G} = V^T G V \quad \hat{C} = V^T C V \quad \hat{B} = V^T B \quad (17)$$

Finally we get the reduced thermal system

$$(\hat{G} + s\hat{C})\hat{X}(s) = \hat{B}U(s). \quad (18)$$

We describe the EKS algorithm flow in Figure 4.

Now we can do the transient simulation in time domain on the order reduced model (18) instead of the original model (12) to save the computation cost. And the original simulation results can be obtained by taking projection from $r$ dimension space $\hat{\mathbf{x}}$ to n dimension space $\tilde{\mathbf{x}}$ shown in (19).

$$\tilde{\mathbf{x}}(t) = V\hat{\mathbf{x}}(t). \tag{19}$$

$\tilde{\mathbf{x}}$ is the approximation of **x**.

### *3.2 Input moments generation*

In order to perform model order reduction, we need to get the right-hand-side moments for input power traces, which means we need transform the input power signal from time domain to frequency domain.

For the thermal transient simulation algorithm described in section II, the input power trace looks like the waveform shown in Figure 5.

For this special pulse input, we can describe it in time domain as

$$\mathbf{u}(t) = \sum_{i=0}^{k} \mathbf{p}_i(E(t-t_i) - E(t-t_{i+1})), \tag{20}$$

where *E*(*t*) is a unit step function. After Laplace transformation, the s domain representation of input signal becomes

$$U(s) = \sum_{i=0}^{k} \mathbf{p}_i(\frac{e^{-t_i s}}{s} - \frac{e^{-t_{i+1} s}}{s}). \tag{21}$$

By taken Taylor expansion, (21) can be described as

$$\begin{aligned} U(s) = \sum_{i=0}^{k} \mathbf{p}_i\{[(-t_i) - (-t_{i+1})] + \frac{1}{2!}[(-t_i)^2 - (-t_{i+1})^2]s \\ + \frac{1}{3!}[(-t_i)^3 - (-t_{i+1})^3]s^2 + \cdots\}. \end{aligned} \tag{22}$$

Then we get the input moment representation as (13), where

$$\begin{aligned} \mathbf{u}_0 &= \sum_{i=0}^{k} \mathbf{p}_i\{[(-t_i) - (-t_{i+1})]\} \\ \mathbf{u}_1 &= \sum_{i=0}^{k} \mathbf{p}_i\{\frac{1}{2!}[(-t_i)^2 - (-t_{i+1})^2]\} \\ \mathbf{u}_2 &= \sum_{i=0}^{k} \mathbf{p}_i\{\frac{1}{3!}[(-t_i)^3 - (-t_{i+1})^3]\} \\ &\vdots \end{aligned} \tag{23}$$

## 4 FAST EXTENDED KRYLOV SUBSPACE INTEGRATION SIMULATION METHOD – FEKIS

In this section, we first summarize our FEKIS algorithm flow. Then we present a grid-based fine granularity thermal model for better accuracy

### *4.1 FEKIS algorithm flow*

The FEKIS algorithm flow is shown in Figure 6.

The FEKIS algorithm computes the average powers by using the sliding window based method such that a number of long fixed time steps are used. Then it computes the powers in terms of its moment forms. After the EKS method is used to reduce the original large thermal circuit matrices into small ones, those reduced circuits typically are very dense, which is suitable for the PIM method to compute the responses over a number of time internals with fixed time steps. Finally the transient responses are computed by mapping back to the original circuit nodes.

For online thermal circuits simulation, we can start with any time instance with known initial temperature and compute the temperature during the given time for the computed or predicted power traces. In this way thermal profiles can be computed in an incremental way, which is suitable for online temperature estimation.

### *4.2 Finer granularity thermal models for functional units*

Existing architecture thermal simulation assumes that each functional unit block has the uniform temperature and is treated as one node in the thermal circuit. However such model is not very accurate, since we treat the whole function block as a thermal uniform subject and ignore the gradient within the block. The other downside of block-based model is that temperature discrepancy would occur when there're blocks with radically different sizes [20].

To obtain more accurate thermal profiles within each functional unit, we need to have more accurate thermal models. One way is to generate more detailed thermal models by using grid-based method. The idea is that we further divided each FU into hundreds of sub-blocks. The cost is the larger thermal RC circuits, and longer simulation time.

## 5 EXPERIMENTAL RESULTS

We implemented the proposed FEKIS algorithm and three other algorithms, TMM (Thermal Moment Matching) [1] [2], PIM and simple integration method using Backward Euler (BE) in MATLAB 7.0. Since the integer register file is typically hottest [5], we show all experimental results at this functional unit. For the BE method, we only use the constant time step. If variable time step is used, BE can be faster. But the speedup depends on the circuits and additional overhead will be incurred.

First, we compare the experimental results by running proposed algorithm FEKIS in terms of speedup and accuracy with TMM on a Compaq Alpha 21364 shown in Figure 7, which is also used in[6]. Then we compare the new FEKIS algorithm with PIM algorithm, which is without reducing model order, on a Pentium 4 Northwood floor-plan shown in Figure 10. The reference data is obtained by running the first-order BE method that only can handle small time-step simulation. In this way, we

know the speedup contribution breakdown regarding different algorithms.

### *5.1 Comparing with the Thermal Moment Matching (TMM) method*

The floor-plan of Compaq Alpha 21364 is shown in Figure 7. First we test the FEKIS algorithm by running 9 same SPEC2K [14] benchmarks used in TMM on a 3GHz P4 processor. We sample each FU's power at 10K cycle intervals, which yields 3.3μs time interval. We set the segment window length to 1500 such that we get 5ms fixed simulation time interval. The experimental parameters are set to N = 20 for FEKIS and the reduced order is 7 for both FEKIS and TMM.

Figure 8 and 9 shows the simulation results using different algorithms under different SPEC2K benchmarks. It also gives out the error of both FEKIS and TMM compared with BE algorithm. We randomly select two simulation results, one is under *gzip* program and another is under *parser* program. Both FEKIS and TMM achieve very accurate results compared with BE algorithm, and both maximum errors are less than $0.5^{\circ}C$ that is far from $2^{\circ}C$ deviation offered by real thermal sensor in previous DTM scheme [6]. However, the speedup of FEKIS over TMM achieve to 14X and 11X for those two benchmarks.

We summarize the statistic of all testing benchmarks and results in Table I. We notice that the new proposed FEKIS algorithm achieves 12X average speedup with respect to TMM method, and 1241X average speedup over BE algorithm based on running 9 SPEC2K programs.

For accuracy evaluation, since we regard the simulation results from running the BE method as the golden values, we compare the other two set of results with them. We find that under these 9 benchmarks FEKIS achieves only $0.424^{\circ}C$ maximum error and $0.137^{\circ}C$ average error, which are very close to the performance by TMM, $0.409^{\circ}C$ maximum error and $0.144^{\circ}C$ average error. Both these two methods provide high accurate temperature simulation results.

### *5.2 Comparing with the PIM method*

In this section, the experimental parameters are set to $N = 20$ for algorithms PIM and FEKIS, and the reduced order is 20 for FEKIS as we use different architecture. First we test the proposed new algorithm on the original thermal model shown in Figure 10. Then we partition the thermal model into finer granularity grid model and test the new algorithms.

*1) Original FU block model simulation:* The Pentium 4 Northwood floor-plan is shown in Figure 10.We run 22 SPEC2K [14] benchmarks on it and sample each FU's power at $4\mu s$ time interval. For the proposed algorithms PIM and FEKIS, we set the sliding window size to 1000, which yields $4ms$ fixed simulation time interval.

In Figure 11, we show the accuracy of both proposed algorithms with or without MOR under *mcf* benchmark. Both algorithms achieve high accuracy. FEKIS has maximum error of $0.61^{\circ}C$ and average error of $0.08^{\circ}C$ for the whole execute time simulation, and the running time is 0.52 second compared with 1.61 second and 1861 second by PIM and BE algorithms.

We test our proposed algorithms under other benchmarks and summarize results in Table II. For all 22 benchmarks, the average speedup of FEKIS over BE algorithm is $3958X$, which is very significant. But FEKIS only achieves to average $2.9X$ speedup over PIM because of the small thermal model size. Although there are some maximum errors beyond $1^{\circ}C$, we observe that the maximum errors mostly

happen at the beginning of simulation and last for very short time due to the initial dramatic input power change.

*2) Finer granularity grid thermal model simulation:* The second example uses the same microprocessor as the first example. However, we do the regular partition for each FU block. In our experiment, we evenly partition each FU block into $3 \times 3$ grids. Therefore, the node number of new thermal model changes from 82 to 658. Then the FEKIS algorithm shows better efficiency than the PIM algorithm.

First we present a temperature response shown in Figure 12 under *mgrid* benchmark. The proposed method FEKIS still obtain accurate results with maximum error $0.20^{\circ}C$ and average error $0.02^{\circ}C$. The speedup over the PIM algorithm is more obvious, which is $14X$ vs. $3X$.

Similarly we run all 22 benchmarks on the $3 \times 3$ grid model. From Table II, we can see the speedup of the MOR aided precise integration method, FEKIS, is more advantageous than the PIM algorithm which is without model reduction, and achieves averagely 25X speedup. The speedup coming from the PIM method is still very impressive. As a result, we can see the proposed FEKIS method is a very efficient architecture level thermal simulation method. We believe that FEKIS will deliver more speedup for very detailed thermal circuits.

## 6 Conclusion

In this paper, we have presented a novel architecture thermal analysis algorithm, FEKIS, for fast temperature estimation of high performance microprocessors and platforms. The new efficient algorithms can be used as software sensors for dynamic thermal management and architecture level thermal estimation for early stages of the microprocessor design. The new algorithm combines the Krylov subspace for reduction and the large-step integration, which exploits the long input power traces, to speed up the simulation. Experimental results showed that FEKIS can lead to about $10X$ speedup over TMM and the precise time-step integration algorithm without model reduction and about $1000X$ faster than the numerical integration method BE with high accuracy.

## REFERENCES


[1] H. Li, P. Liu, Z. Qi, L. Jin, W. Wu, S. X.-D. Tan, and J. Yang, "Efficient thermal simulation for run-time temperature tracking and management." Proc. IEEE Int. Conf. on Computer Design (ICCD) **(2005)**, pp. 130–133.

[2] W. Wu, L. Jin, J. Yang, P. Liu, and S. X.-D. Tan, "Efficient power modeling and software thermal sensing for runtime temperature monitoring." ACM Trans. on Design Automation of Electronics Systems **(2007)**, Vol. 12, p. 26.

[3] International technology roadmap for semiconductors(itrs), 2004 update **(2004)**, http://public.itrs.net.

[4] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors." Proc. of Intl. Symp. on High-Performance Comp. Architecture **(2001)**, pp. 171–182.

[5] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature aware microarchitecture." Proc. IEEE International Symposium on Computer Architecture (ISCA) **(2003)**, pp. 2–13.

[6] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature aware microarchitecture: Extended discussion and results." University of Virginia, Dept. of Computer Science, Technical Report CS-2003-08 **(2003)**.

[7] W. Huang, M. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design." Proc. Design Automation Conf. (DAC) **(2004)**, pp. 878–883.

[8] S. Gunther, F. Binns, D. Carmean, and J. Hall, "Managing the impact of increasing microprocessor power consumption". Intel Technology Journal **(2001)**.

[9] K. Lee and K. Skadron, "Using performance counters for runtime temperature sensing in high performance processors." the Workshop on High-Performance, Power-aware Computing(HP-PAC), in conjunction with the 2005 International Parallel and Distributed Processing Symposium **(2005)**.

[10] K. Sankaranarayanan, S. Velusamy, M. R. Stan, and K. Skadron, "A case for thermal-aware floorplanning at the microarchitectural level". The Journal of Instruction-Level Parallelism **(to appear)**.

[11] P. Rosinger, B. M. Al-Hashimi, and K. Chakrabarty, "Thermal-safe test scheduling for core-based system-on-chip integrated circuits." IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems **(2006)**, Vol. 25, pp. 2502–2512.

[12] C. Liu, V. Iyengar, and D. K. Pradhan, "Thermal-aware testing of network-on-chip using multiple-frequency clocking." Proceedings of the 24th IEEE VLSI Test Symposium **(2006)**, pp. 46–51.

[13] T. Sherwood, E. Perelman, and B. Calder, "Basic block distribution analysis to find periodic behavior and simulation points in applications." the International Conference on Parallel Architectures and Compilation Techniques(PACT2001) **(2001)**, pp. 3–14.

[14] Http://www.spec.org/cpu2000/CFP2000/.

[15] W. X. Zhong and F. W. Williams, “A high precise time step integration method.” Journal of Mechanical Engineering Science **(1994)**, Vol. 208, pp. 427–430.

[16] W. X. Zhong, J. Zhu, and X. X. Zhong, “On a new time integration method for solving time dependent partial differential equations.” Computer Methods in Applied Mechanics Engineering **(1996)**, Vol. 130, pp. 163–178.

[17] L. He, W. Liao, and M. R. Stan, “System level leakage reduction considering the interdependence of temperature and leakage.” Proc. Design Automation Conf. (DAC) **(2004)**, pp. 12–17.

[18] S. X.-D. Tan and L. He, Advanced Model Order Reduction Techniques in VLSI Design, Cambridge University Press **(2007)**.

[19] J. M. Wang and T. V. Nguyen, “Extended Krylov subspace method for reduced order analysis of linear circuit with multiple sources.” Proc. Design Automation Conf. (DAC) **(2000)**, pp. 247–252.

[20] W. Huang, S. Ghosh, K. Sankaranarayanan, K. Skadron, and M.~R. Stan, ``HotSpot: A compact thermal modeling methodology for early-stage VLSI design," IEEE Trans. on Very Large Scale Integration (VLSI) Systems (**2006**), vol. 14, no. 5, pp. 501--513.

## FIGURES AND TABLES

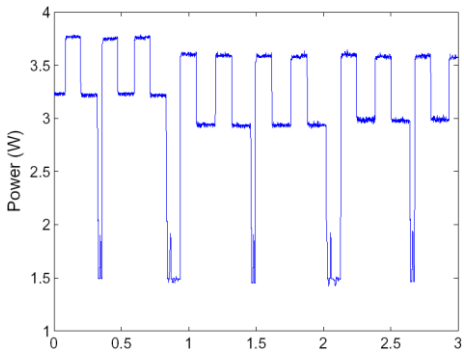



**Figure 1. The power trace under SPEC2K *wupwise* benchmark at the integer register file unit of Compaq Alpha 21364.**

PIM ALGORITHM:

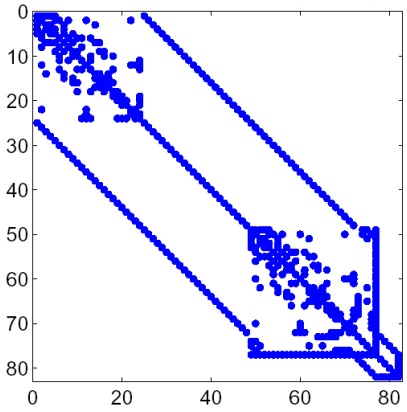
Input: $G$, $C$, $B$, $\mathbf{u}(t)$, and initial state $\mathbf{x}(0)$

Output: $\mathbf{x}(t)$

1. Solve $T$, $H$ and $\tau$ using (5) and (8);
2. Compute $A_a$ using (9);
3. Compute $A$ using (11);
4. Do transient simulation using (6).

**Figure 2. Fast precise integration method (PIM).**

a) Thermal conductance matrix(nz = 488)




b) Thermal conductance matrix(nz = 82)

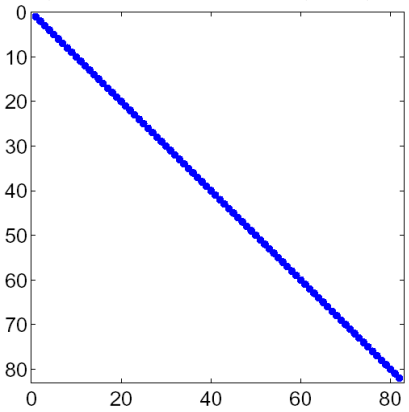



**Figure 3. The sparsity of thermal circuit.**

EKS ALGORITHM

Input: $G$, $C$, $B$, and $U$

Output: $\hat{G}$, $\hat{C}$, $\hat{B}$

1. $\tilde{\mathbf{m}}_0 = G^{-1}B\mathbf{u}_0$ and $d_0 = 1/\| \tilde{\mathbf{m}}_0 \|$
2. $\mathbf{v}_0 = d_0\tilde{\mathbf{m}}_0$
3. $\mathbf{h}_{sum} = 0$ and $d = d_0$
4. for $i = 1$ to $r - 1$
5. $\quad \tilde{\mathbf{m}}_i = G^{-1}(dB\mathbf{u}_i - C(\mathbf{v}_{i-1} + d_{i-1}\mathbf{h}_{sum}))$
6. $\quad \mathbf{h}_{sum} = 0$
7. $\quad$ for $j = 0$ to $i - 1$
8. $\quad\quad h = v_j^T \tilde{m}_i$
9. $\quad\quad \mathbf{h}_{sum} = \mathbf{h}_{sum} + h\mathbf{v}_j$
10. $\quad \tilde{\mathbf{v}}_i = \tilde{\mathbf{m}}_i - \mathbf{h}_{sum}$
11. $\quad d_i = 1/\| \tilde{\mathbf{v}}_i \|$ and $d = d * d_i$
12. $\quad \mathbf{v}_i = d_i\tilde{\mathbf{v}}_i$
13. $V = [\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{r-1}]$
14. $\hat{G} = V^T GV$, $\hat{C} = V^T CV$, $\hat{B} = V^T B$

**Figure 4. EKS algorithm for model order reduction.**

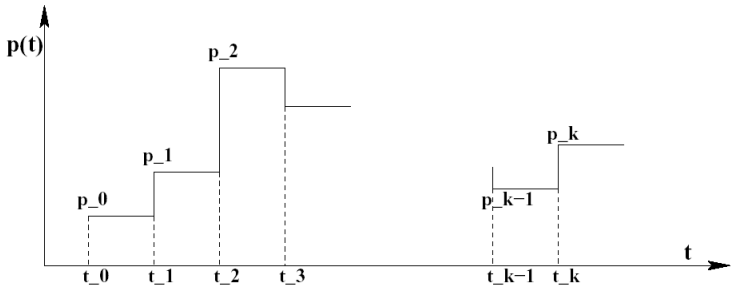



**Figure 5. The waveform of input power trace for thermal transient simulation.**

FEKIS ALGORITHM

Input: $G$, $C$, $B$, $\mathbf{u}(t)$, and reduced order $r$

Output: $\mathbf{x}(t)$

1. Compute the window-based power average for input power patterns;
2. Generate input moments $U(s)$ using (23);
3. Compute reduced order model using EKS algorithm;
4. Perform PIM for fast precise integration on reduced order model;
5. Compute original states approximation using (19).

**Figure 6. FEKIS algorithm flow.**

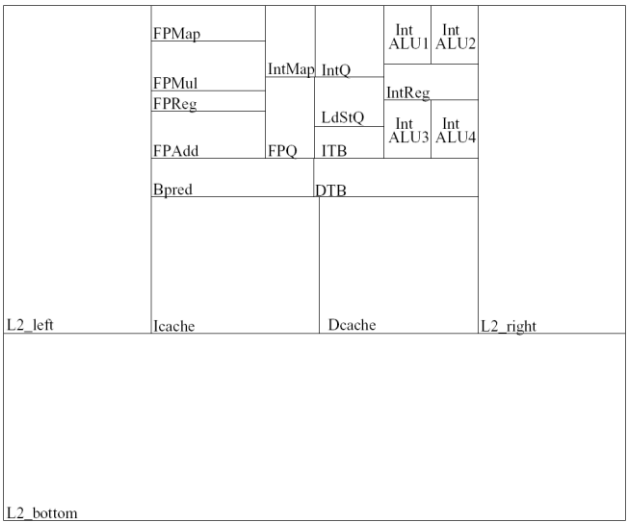



**Figure 7. The modified architecture of Compaq Alpha 21364.**

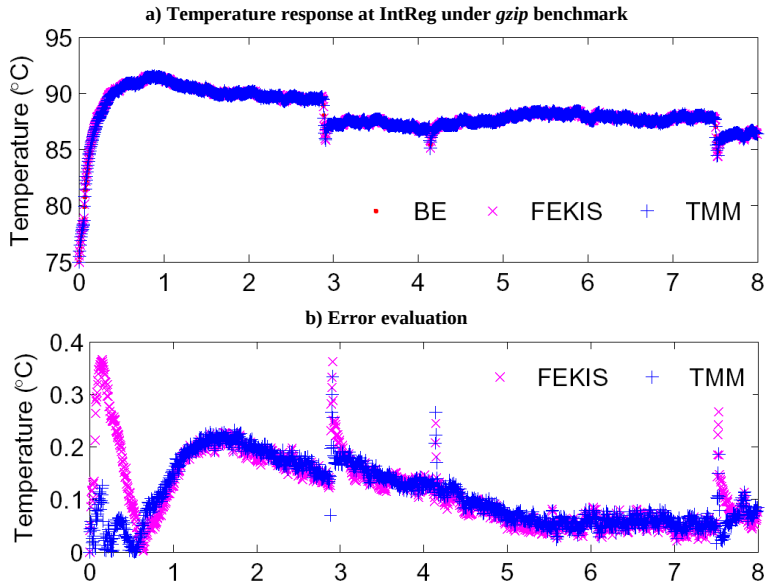



**Figure 8. Temperature comparison between FEKIS, TMM, and BE under benchmark *gzip* at the integer register file unit.**

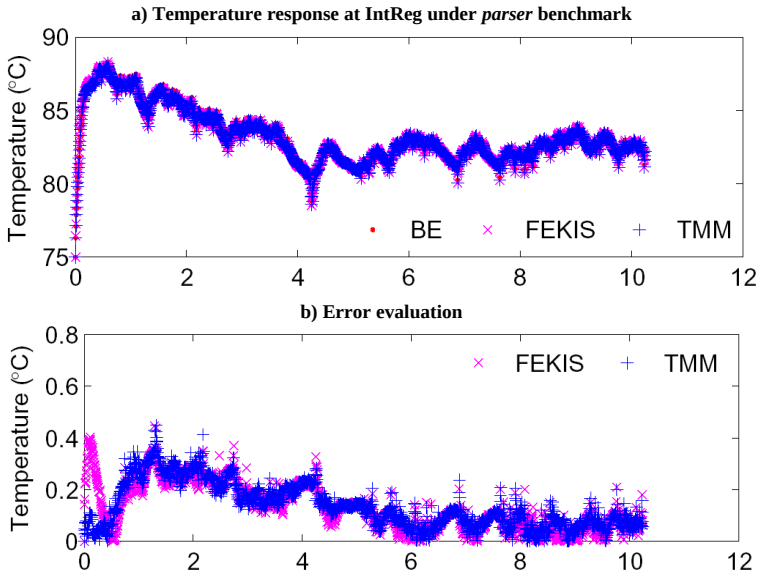



**Figure 9. Temperature comparison between FEKIS, TMM, and BE under benchmark *parser* at the integer register file unit.**

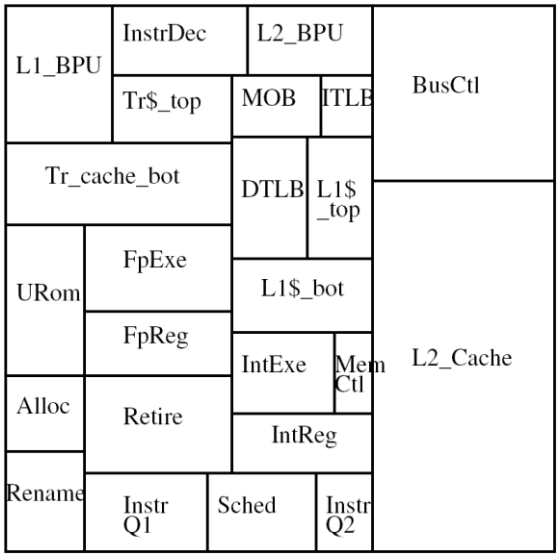



**Figure 10. Pentium 4 Northwood floor-plan.**

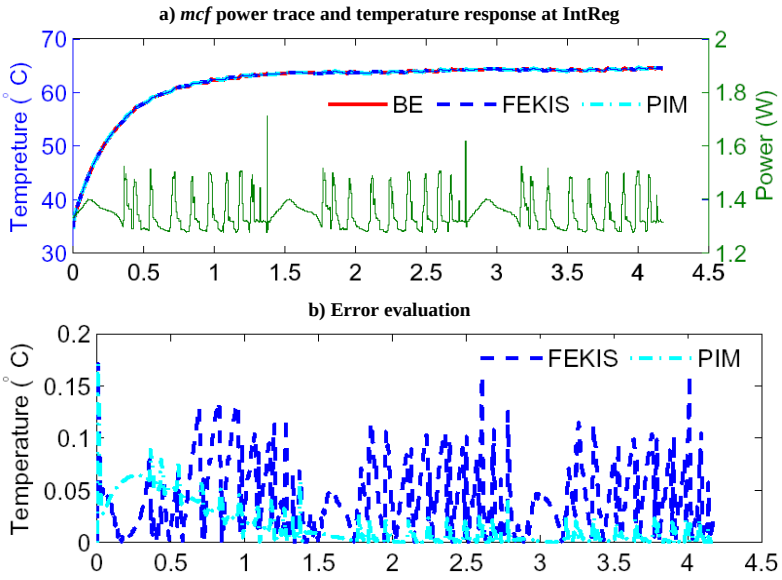



**Figure 11. Performance and error evaluation on FU block model under *mcf* benchmark.**

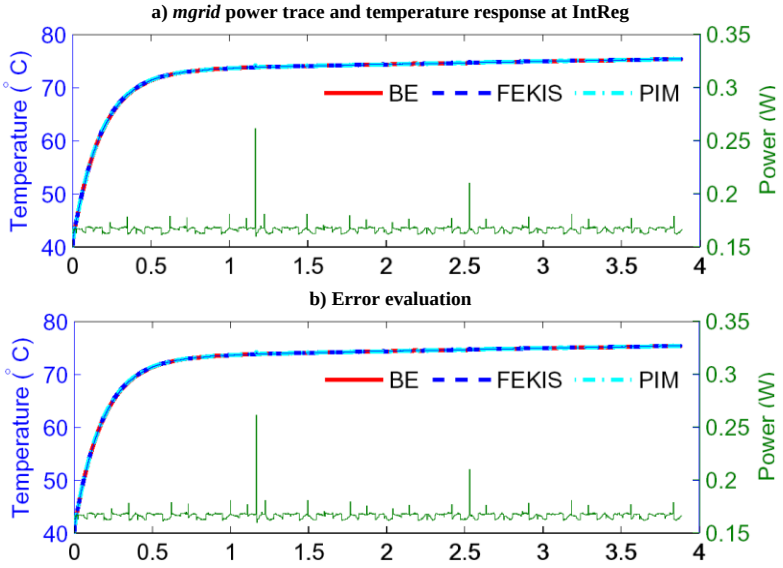



**Figure 12. Performance and error evaluation on FU grid model under *mgrid* benchmark.**

**TABLE I**
**PERFORMANCE EVALUATION OF FEKIS VS. TMM**
**UNDER SPEC2K BENCHMARKS ON COMPAQ ALPHA 21364.**

| SPEC2K | Ins. Cycles | Execute | CPU Time (sec.) | | | Speedup | | Max. Err.(°C) | | Ave. Err.(°C) | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Benchmark | (billion) | Time(s) | FEKIS | TMM | BE | TMM | BE | FEKIS | TMM | FEKIS | TMM |
| gcc | 28.5 | 9.49 | 3.90 | 42.05 | 4357 | 11 | 1117 | 0.42 | 0.38 | 0.11 | 0.11 |
| mesa | 20.0 | 6.68 | 1.86 | 28.54 | 3065 | 15 | 1648 | 0.36 | 0.37 | 0.17 | 0.18 |
| vortex | 74.3 | 24.75 | 25.23 | 127.01 | 11360 | 5 | 450 | 0.34 | 0.28 | 0.07 | 0.07 |
| swim | 11 | 3.68 | 0.56 | 14.17 | 1690 | 25 | 3018 | 0.47 | 0.55 | 0.31 | 0.35 |
| parser | 30.7 | 10.24 | 4.44 | 47.31 | 4700 | 11 | 1059 | 0.45 | 0.45 | 0.13 | 0.14 |
| lucas | 46.7 | 15.53 | 10.14 | 65.12 | 7131 | 6 | 703 | 0.50 | 0.56 | 0.16 | 0.17 |
| bzip | 55.7 | 18.55 | 14.42 | 91.52 | 8514 | 6 | 590 | 0.59 | 0.49 | 0.05 | 0.06 |
| eon | 27.3 | 9.07 | 3.52 | 41.38 | 4166 | 12 | 1184 | 0.32 | 0.27 | 0.11 | 0.11 |
| gzip | 24 | 7.99 | 2.62 | 36.49 | 3669 | 14 | 1400 | 0.37 | 0.33 | 0.12 | 0.11 |
| Average | - | - | - | - | - | **12X** | **1241X** | **0.424** | **0.409** | **0.137** | **0.144** |

**TABLE II**
**PERFORMANCE EVALUATION OF FEKIS VS. PIM**
**UNDER SPECK2K BENCHMARKS ON PENTIUM 4 NORTHWOOD.**

| SPEC2K | Execute | Original block FU model | | | | | Finer granularity grid model | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Time(s) | Sim. (s) | vs. BE | vs. PIM | Max(°C) | Ave(°C) | Sim. (s) | vs. BE | vs. PIM | Max(°C) | Ave(°C) |
| ammp | 6.26 | 1.24 | 3289 | 2.7 | 0.36 | 0.05 | 9.59 | 3450 | 7.4 | 0.24 | 0.03 |
| applu | 3.70 | 0.34 | 4732 | 3.3 | 0.18 | 0.05 | 3.52 | 4177 | 16.3 | 0.27 | 0.04 |
| apsi | 6.08 | 0.83 | 4724 | 4.0 | 1.19 | 0.15 | 10.39 | 3190 | 6.8 | 0.35 | 0.07 |
| art | 14.16 | 7.09 | 2572 | 2.4 | 0.10 | 0.02 | 49.87 | 2838 | 3.7 | 0.33 | 0.03 |
| bzip | 2.72 | 0.28 | 3125 | 2.0 | 0.56 | 0.08 | 2.08 | 4455 | 27.0 | 0.28 | 0.06 |
| crafty | 1.42 | 0.09 | 3656 | 3.0 | 0.21 | 0.06 | 0.64 | 5590 | 65.0 | 0.31 | 0.09 |
| equake | 1.64 | 0.11 | 3691 | 2.3 | 0.69 | 0.05 | 1.14 | 4578 | 38.2 | 0.24 | 0.07 |
| facerec | 4.54 | 0.48 | 4623 | 4.0 | 0.40 | 0.13 | 5.24 | 3757 | 11.0 | 0.24 | 0.05 |
| fma3d | 5.15 | 0.84 | 3423 | 2.7 | 0.91 | 0.08 | 6.77 | 3556 | 9.2 | 0.66 | 0.11 |
| gap | 1.56 | 0.09 | 3989 | 3.0 | 0.21 | 0.06 | 0.97 | 4865 | 49.0 | 0.26 | 0.06 |
| gcc | 1.64 | 0.13 | 3608 | 2.4 | 1.21 | 0.27 | 1.05 | 5626 | 63.1 | 0.42 | 0.08 |
| gzip | 2.02 | 0.13 | 4331 | 3.0 | 0.94 | 0.17 | 1.61 | 5376 | 40.9 | 0.40 | 0.05 |
| lucas | 3.77 | 0.27 | 5615 | 4.4 | 0.61 | 0.08 | 4.98 | 3517 | 14.7 | 0.40 | 0.05 |
| mcf | 4.18 | 0.52 | 3635 | 3.1 | 0.61 | 0.08 | 5.88 | 3667 | 12.2 | 0.14 | 0.03 |
| mesa | 2.44 | 0.19 | 3868 | 2.6 | 0.51 | 0.05 | 1.89 | 5514 | 34.9 | 0.34 | 0.04 |
| mgrid | 4.05 | 0.50 | 3532 | 2.9 | 0.60 | 0.03 | 5.63 | 3796 | 14.1 | 0.20 | 0.02 |
| parser | 3.14 | 0.36 | 3428 | 2.5 | 0.41 | 0.09 | 3.41 | 4719 | 20.4 | 0.26 | 0.04 |
| swim | 6.48 | 1.52 | 2827 | 2.4 | 1.18 | 0.05 | 13.70 | 3715 | 6.7 | 0.14 | 0.01 |
| twolf | 1.38 | 0.56 | 5468 | 5.0 | 0.80 | 0.02 | 9.47 | 3679 | 8.9 | 0.40 | 0.01 |
| vortex | 2.30 | 0.16 | 5469 | 2.4 | 0.64 | 0.10 | 1.87 | 5005 | 32.4 | 0.27 | 0.06 |
| vpr | 3.12 | 0.28 | 4132 | 2.6 | 0.62 | 0.07 | 3.31 | 4451 | 20.2 | 0.24 | 0.04 |
| wupwise | 1.85 | 0.14 | 3343 | 2.1 | 0.42 | 0.09 | 1.19 | 5133 | 47.9 | 0.24 | 0.04 |
| Average | - | - | **3958X** | **2.9X** | **0.61** | **0.08** | - | **4302X** | **25.0X** | **0.30** | **0.05** |

## BIOGRAPHIES

**Pu Liu** received a Ph.D. degree in Electrical Engineering from the University of California at Riverside in 2008. Now he is a Senior R&D engineer of Cadence Design System Inc, CA.

**Sheldon X.-D. Tan** received his B.S. and M.S. degrees from Fudan University, Shanghai, P. R. China, in 1992 and 1995 respectively and the Ph.D. degree from the University of Iowa, Iowa City, in 1999. He is the director of Mixed-Signal Nanometer VLSI Research Lab (MSLAB) at UC Riverside. He is also a cooperating faculty member (co-faculty) of Computer Science and Engineering department and a participating faculty member in Laboratory for Integrated Circuits and Systems (LICS) group.

**Lin Jiang** received his B.S. and Ph.D. degrees from Xi'an Jiaotong University, Shaanxi, P. R. China, in 1992 and 1996 respectively. He is the vice director of Department of Computer Science of Xi'an Institute of Post & Telecommunications. He is also the deputy director of ASIC Design Center at Xi'an Institute of Post & Telecommunications. Now he is a visiting researcher of Mixed-Signal Nanometer VLSI Research Lab (MSLAB) at UC Riverside.

**Wei Wu** received a Ph.D. degree in Computer Science from the University of California at Riverside in 2008. Now she is a Research Scientist of Intel Corporation, Portland.

**Murli Tirumala** (not available at this time)