Optimal Chip-Package Codesign for High-Performance DSP

Pronita Mehrotra, *Member, IEEE*, Vikram Rao, Thomas M. Conte, *Senior Member, IEEE*, and Paul D. Franzon, *Senior Member, IEEE*

***Abstract*—In high-performance DSP systems, the memory bandwidth can be improved using high-density interconnect technology and appropriate memory mapping. High-density MCM and flip-chip solder bump technology is used to achieve a system with an I/O bandwidth of 100 Gb/s/cm^2 die. The use of DRAMs in these systems usually make the performance of these systems poor, and some algorithms make it difficult to fully utilize the available memory bandwidth. This paper presents the design of an fast Fourier transform (FFT) engine that gives SRAM-like performance in a DRAM-based system. It uses almost 100% of the available burst-mode memory bandwidth. This FFT engine can compute a million-point FFT in 1.31 ms at a sustained computation rate of 8.64 $\times$ 10^{10} floating-point operations per second (FLOPS). This is at least an order of magnitude better than conventional systems.**

***Index Terms*—Chip-package codesign, fast Fourier transform (FFT), seamless high off-chip connectivity (SHOCC).**

I. Introduction

HIGH-PERFORMANCE DSP applications, like synthetic aperture radar (SAR), require extremely large computation rates and have large working data sets. By using a high-density interconnect technology like seamless high off-chip connectivity (SHOCC), memory bandwidth can be improved. As the demand for higher resolution DSP systems increases, the computation rate is expected to reach tera floating-point operations per second (TFLOP) rates. These applications involve manipulations to large data volumes (1 GB or more), which makes it necessary from a cost point of view to use DRAMs in these systems. DRAMs, due to their refresh and row access cycles, would be expected to perform much worse than SRAMs in most cases. In addition, signal processing algorithms are mostly memory starved, and one expects that increasing the memory bandwidth would improve performance. Some algorithms make it difficult to fully utilize the available memory bandwidth. The work reported in this paper focuses on how large DRAM-based DSP systems that are capable of high sustained memory performance can be built using SHOCC technology. Depending on the application, an efficient memory management scheme can lead to better memory bandwidth utilization and give almost SRAM-like performance.

Manuscript received September 25, 2003; revised March 22, 2004. This work was supported by ARDA under Contract MDA904-00-C-2133 and the NSF under Contract EIA-9703090.

P. Mehrotra was with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh NC 27695 USA. She is now in Allentown, PA, 18104 USA. (e-mail: pronita@ieee.org).

V. Rao was with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh NC 27695 USA. He is now with Sun Microsystems, Mountain View, CA 94043 USA.

T. M. Conte and P. D. Franzon are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh NC 27695 USA (e-mail: conte@eos.ncsu.edu; paulf@eos.ncsu.edu).

Digital Object Identifier 10.1109/TADVP.2005.846937

SHOCC is a combined packaging, interconnect, and IC design technology aimed at providing system-level integration by using very high density solder bump and thin-film technologies [1], [2]. The purpose of this paper is to show how the use of such a technology can result in a radical performance improvement in a specific application. We show that a five-layer thin-film multichip module (MCM), together with a 140-μm flip-chip solder bump technology can be used to achieve a peak I/O bandwidth of 100 Gb/s/cm^2 die without compromising noise performance. The key to achieving this performance is the use of a redistribution layer with a local ground. There are several applications that can benefit from such high I/O bandwidth including networking and graphics.

In this paper, we take a large fast Fourier transform (FFT) system to demonstrate the performance gains that can be achieved when combined with an efficient memory mapping scheme and a better utilization of available resources. We chose an FFT system to demonstrate this, since the FFT is the most challenging and time consuming part in many signal processing algorithms, and is difficult to map onto DRAMs.

The rest of the paper is organized as follows. Section II discusses the signal integrity related issues which determine the maximum bandwidth that can be obtained from the SHOCC technology. Section III presents the results obtained after simulating the SHOCC transmission line. Section IV summarizes the results of the simulation in terms of the overall bandwidth available from the SHOCC technology. Section V discusses the physical architecture of the FFT system, i.e., the layout of the memory and the microaccelerators on the SHOCC substrate, the details of the microaccelerator chip and the sequence of operations of the FFT system. Section VI describes the logical architecture of the design. The focus in this section is on two schemes that help improve the performance of the FFT system: 1) the memory management scheme which extracts the maximum possible performance out of DRAMs and 2) the twiddle factor generation scheme, which is crucial for the successful operation of the FFT. We finally conclude by analyzing the performance of the FFT engine in Section VII.

II. Signal Integrity in Densely Routed Substrates

This section looks at various circuit-related aspects like the substrate cross section, number of routing layers, routing

1521-3323/$20.00 © 2005 IEEE

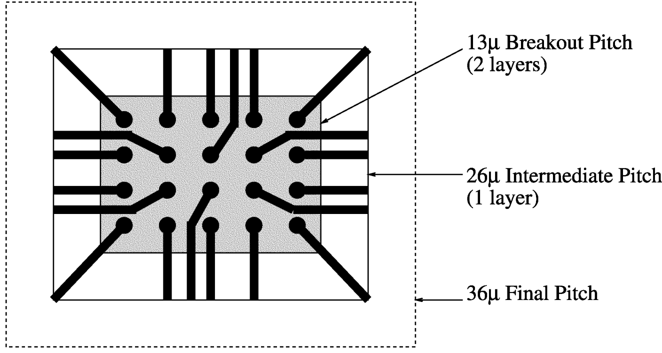


Fig. 1. Two-stage breakout routing approach.

pitches, etc., that maximize the bandwidth between two die interconnected via a SHOCC substrate. These issues are governed by the number of signals that need to be routed from each chip and the routing strategy used. In our design, we have limited our alternatives to those that use only two layers for routing. This reduces the cost of the substrate processing. Other constraints used in the design, to reduce processing costs, are to use a minimum trace width of 10 μm and not more than 50 μm of the total substrate stackup. The trace thickness is 2 μm in all cases considered.

The number of signal I/Os that need to be routed out per chip (the architecture is presented in Section IV), including control signals, in the design is around 2000. To add more flexibility to the design, we have used a stricter estimate of around 2500 signal I/Os to be routed out per chip. In addition, the ratio of power and ground bumps to I/O bumps is assumed to be 1:1. This means that a total of 5000 bumps need to be placed on each chip. For a 1 cm $\times$ 1 cm chip, the bump pitch required to support these many bumps is 140 μm. A two-stage breakout approach has been assumed as shown in Fig. 1. In the initial phase of the breakout, the routing pitch between the traces in the two layers is small. This makes the crosstalk noise in this phase very high. In the intermediate phase, the routing becomes *XY* in nature, and mutual coupling (and, hence, crosstalk) between traces on the two layers becomes very small. In the final phase, the routing pitch is determined by the routing under the DRAMs and is larger than the pitches in the first two phases. This phase contributes least to the total crosstalk noise. To determine the routing pitches as shown in Fig. 1, we have also assumed an overhead of 15%–20% to account for the loss of area due to power and ground vias.

A. Modeling and Simulation of SHOCC Lines

To model the SHOCC transmission line, *R*, *L*, and *C* parameters were first extracted using Maxwell's Q-3D parameter extractor [3]. For the crosstalk noise analysis, we extended the model used in [4], to include the couplings from all the nearest neighbors. A SHOCC line, driven by a five-stage driver (with a stage ratio of 3), was then simulated in SPICE. The signal starts from the output of driver and goes through ≈0.2-mm on-chip line. It then travels through a via to the solder bump on the long off-chip SHOCC line. At the other end of the SHOCC line, it goes through another solder bump, a second 0.2 mm of an on-chip segment, and finally to the receiver. The on-chip segments account for the distance that the signal has to travel from the driver/receiver to the nearest solder bump. This value of the on-chip segment length and the *R*, *L*, *C* parameters for the on-chip segment and the solder bump has been kept the same as in [4], [5]. Fig. 2 shows a segment of the SHOCC transmission line which includes the coupling effects of the nearest neighbors. The various parameters in the figure are explained below:

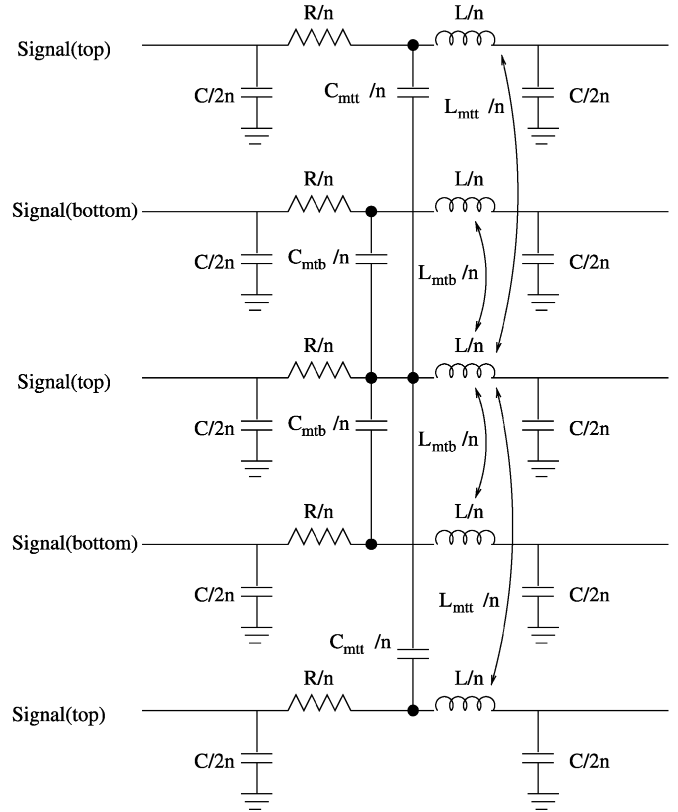


Fig. 2. Equivalent circuit model used for crosstalk simulations.

1) R = resistance per unit length of a trace;
2) C = self capacitance per unit length of a trace;
3) C_{mtt} = mutual capacitance per unit length between two neighbors on the top layer;
4) C_{mtb} = mutual capacitance per unit length between a trace on the top layer with its neighbor on the bottom layer;
5) C_{mbb} = mutual capacitance per unit length between two neighbors on the bottom layer;
6) L = self inductance per unit length of a trace loop;
7) L_{mtt} = mutual inductance per unit length between two neighbors on the top layer;
8) L_{mtb} = mutual inductance per unit length between a trace loop on the top layer with its neighboring loop on the bottom layer;
9) L_{mbb} = mutual inductance per unit length between two neighboring loops on the bottom layer.

The minimum number of segments required to model a transmission line are governed by the following two equations:[6]

$$T_r \geq 3.5\pi T_0 l \quad (1)$$

$$R_0 l \leq 0.1 Z_0. \tag{2}$$

Our simulations use a rise time of 80 ps and for Z_0 in the range of 50–100 Ω, 15–16 segments are sufficient to model long transmission lines. In our simulations we have taken 20 segments to model all transmission lines, where each segment is represented using the π model. The on-chip interconnects and the solder bump was modeled using an L model [7] and the complete SHOCC line was then driven by cascaded CMOS drivers.

B. Substrate Alternatives

In order to accommodate all the signal I/Os with the pitches dictated by physical limits while maintaining the noise and timing constraints, various substrate stackups were explored. The first choice is between an interconnected mesh power system (IMPS)-like substrate and a more conventional signal/power stackups. In IMPS configuration, the power and ground lines alternate between the signal lines. For instance, for a two-layer IMPS, all the traces in one layer run in the X direction and all the traces in the other layer run in the Y direction. In each layer, the power and ground traces alternate with the signal traces. The mutual couplings between the traces in the two-layer is quite small due to the traces being perpendicular to each other. However, the problem with this topology is that the routing densities become half that of conventional approaches due to the presence of the power and ground traces. For our design, this would mean that the pitch in the initial phase would be 6.5 μm in two layers or 13 μm in each layer. This is not possible with a minimum trace width of 10 μm, as the minimum pitch that the IMPS can handle is 20 μm (with a trace separation of 10 μm). The IMPS topology has, therefore, not been used in our design. From a routing density point of view, conventional signal and power/ground layers are more attractive, and these are discussed next.

Fig. 3 shows two possible stackups with two signal layers each that can be used to route the signal I/Os. In the first case, the signal layers are sandwiched between the power and ground planes. The advantage of this stackup is that the delay and noise on both the signal layers is close to identical. However, signal vias have to cut through the first plane, making routing more difficult. In the second case, the signal layers are on top of the power and ground planes, both of which are placed very close to each other. This maximizes the amount of decoupling capacitance between the power and ground planes. However, the signal characteristics in the two layers is not identical since they are at different distances from the power and ground planes. In our design, we have chosen the second substrate stackup to exploit the inherent decoupling capacitance.

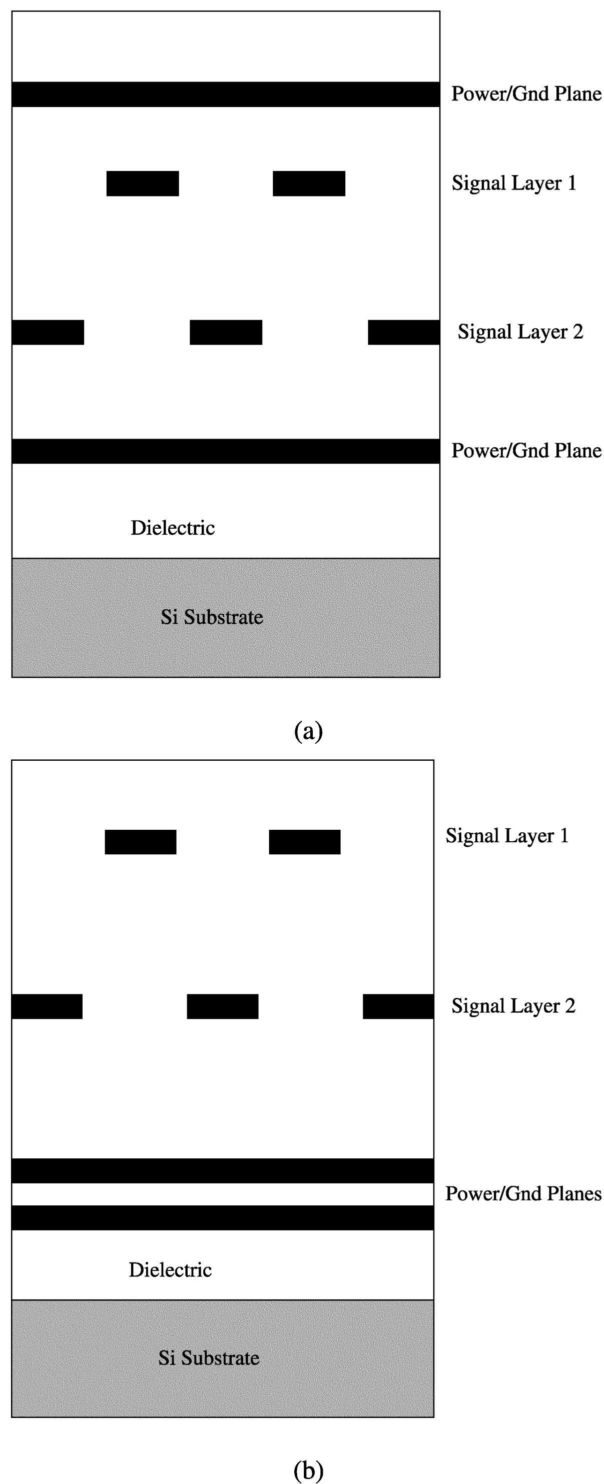


Fig. 3. SHOCC substrate stackups.

For the stackup of above, a noise and timing analysis was performed using the approach outlined in the next section. The crosstalk noise in the initial phase of routing only comes out to be 0.5 V, leaving very little margin for other noise components. This was clearly not acceptable. We, therefore, modified the substrate stackup to include an additional layer between the two signal layers which acts as a local ground (LG) in the initial routing phase as shown in Fig. 4. The addition of the local ground shields the two signal layers from each other and reduces the coupling between the two layers. In the analysis performed in this paper, we assumed this local ground plane was firmly tied to the other package ground planes by an array of vias. The crosstalk noise then comes down to acceptable levels as shown in Fig. 5, where the two curves correspond to traces on the top layer with a driver size of 81x. As can be seen from the figure, considerable reduction in noise can be obtained by adding the local ground. More than two and half times reduction in noise was obtained for a trace length of 1 cm.

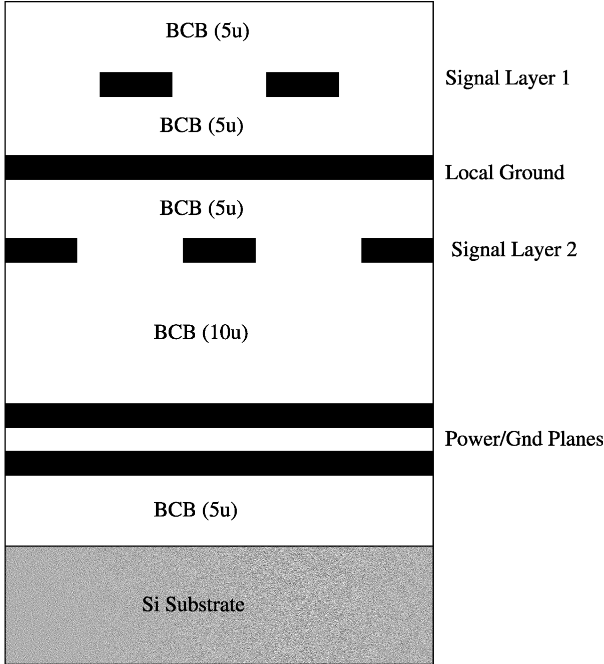


Fig. 4. SHOCC substrate with a local ground.

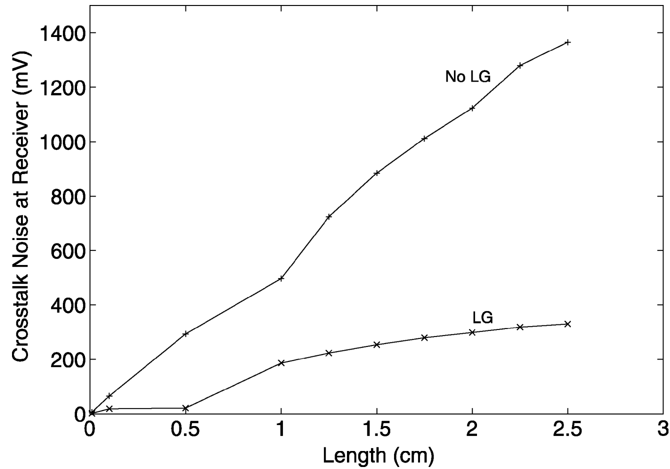


Fig. 5. Crosstalk noise with and without local ground (driver size $= 81\text{x}$) for a trace on the top layer (width $= 10\ \mu$m and pitch $= 26\ \mu$m).

III. Simulation Results

A. Impedance and Propagation Time

The extracted values of *R*, *L*, and *C* and the calculated values of the characteristic impedance and propagation delay for various dimensional parameters is shown in Table I. For all cases, the characteristic impedance of the bottom layer is smaller than that of the top layer while the propagation delay is larger. A few points can be noted from the table. The proximity of the bottom layer to the power/ground planes makes the capacitance parameter larger. It also reduces the loop area, making the inductance smaller. The effect on the capacitance increase dominates over the effect of the decrease in inductance in this case, leading to the above results. The presence of the local ground makes the behavior of the top layer approach that of the bottom layer. The capacitance for both the bottom and the top layer increases, though the increase is more dramatic for the top layer. The difference in the inductances also decreases. A wider trace shows larger capacitance values (due to a larger area).

B. Crosstalk Noise

The extracted mutual coupling parameters are listed in Table II. Without a local ground, the mutual capacitance between two traces on the top layer (C_{mtt}) is always greater than the mutual capacitance between two traces on the bottom layer (C_{mbb}). This is expected due to the proximity of the bottom layer to the ground/power planes and, therefore, a larger capacitance to ground (see Table I). The local ground, as before, tends to make things more even for the bottom and top traces. The mutual capacitance between the top and bottom traces is larger than the mutual capacitance between traces on the same layer. This is again expected, since edge side capacitance is smaller than broadside or diagonally placed traces. The mutual capacitance between the top and bottom layer trace vanishes when the local ground is introduced. Also, from the last two rows in the table, it can be seen that for the same pitch width, a wider trace has higher mutual capacitance than a narrower trace. Mutual inductance between two traces on the top layer is larger than the mutual inductance between two traces on the bottom layer due the larger area of the top layer loops. As before, the presence of the local ground tends to reduce the mutual couplings between traces.

The crosstalk noise for the different cases is shown in Table III. The effect of the stronger couplings for the top layer can be seen in these results, where the top traces show much larger crosstalk noise than the bottom traces. The local ground is very effective in reducing the crosstalk noise in the initial breakout phase. Noise is reduced by 5–6 times with the addition of the local ground.

C. Delay

The worst-case delays for the different cases is shown in Table IV. The delays for a bottom trace are larger than the delays for a trace on the top layer. This verifies the difference in the propagation times listed in Table I.

D. Reflection Noise

The reflection noise for a trace in the final routing phase is shown in Fig. 6. The reflection noise is much smaller than the corresponding crosstalk noise components and constitutes a fairly small percentage of the total noise. The total reflection noise, when all sections are considered together, is 0.006 V for the bottom trace and 0.028 V for the top trace.

IV. Overall Bandwidth Analysis

The escape length for the first breakout stage is 1 cm. The maximum length that a signal has to traverse under the DRAMs, is 2.2 cm (for 11 DRAMs in a row at a pitch of 2 mm). The second escape length for a total routing length of 4 cm is 0.8 cm. The total crosstalk noise can, therefore, be determined by summing the individual components. The maximum crosstalk

TABLE I
IMPEDANCE AND PROPAGATION DELAY FOR SHOCC LINES

Substrate Dimensions (μ)	Signal Layer	Resistance (Ω/cm)	Capacitance (pF/cm)	Inductance (nH/cm)	Characteristic Impedance $Z_0 = (L/C)^{1/2}$	Propagation Delay $\tau = (LC)^{1/2}$
w=10,p=26	Bottom	8.62	0.63	3.65	76.12	47.95
(No LG)	Top	”	0.121	4.64	195.8	23.7
w=10,p=26	Bottom	8.62	0.94	3.06	57.06	53.63
(LG)	Top	”	0.546	3.75	82.87	45.25
w=15,p=36	Bottom	5.75	0.687	4.2	78.2	53.72
(No LG)	Top	”	0.135	4.85	189.5	25.59
w=10,p=36	Bottom	8.62	0.59	3.7	79.2	46.72
(No LG)	Top	”	0.157	4.6	171.17	26.87

TABLE II
MUTUAL L AND C PARAMETERS FOR SHOCC LINES

Substrate Dimensions (μ)	C_{mtt} (pF/cm)	C_{mbb} (pF/cm)	C_{mtb} (pF/cm)	L_{mtt} (nH/cm)	L_{mbb} (nH/cm)	L_{mtb} (nH/cm)
w=10,p=26 (No LG)	0.116	0.042	0.186	1.298	0.63	1.33
w=10,p=26 (LG)	0.0093	0.00317	-	0.565	0.186	0.176
w=15,p=36 (No LG)	0.106	0.012	0.216	0.975	0.315	1.4
w=10,p=36 (No LG)	0.068	0.0068	0.16	0.87	0.34	1.1

TABLE III
CROSSTALK NOISE AT RECEIVER FOR DIFFERENT LAYERS AND DRIVER SIZES (W=WIDTH; P=PITCH, LG=LOCAL GROUND). SEE FIG. 4 FOR THE MEANING OF LOCAL GROUND. THE “3x” ETC., REFERS TO THE SIZE OF THE FINAL STAGE OF THE DRIVER, AS EXPLAINED IN THE TEXT

Trace & driver size	0.25 cm (mv)	0.5 cm (mV)	1 cm (mv)	2 cm (mV)
Bottom trace, LG				
W=10 μm; P=26 μM				
Driver = 3x	24.3			
Driver = 9x	27.8	35.2		
Driver = 27x	21.73	32.1	44.8	64.46
Driver = 81x		34.7	67.6	112.1
Top trace, LG				
W=10 μm; P=26 μM				
Driver = 3x	97.48			
Driver = 9x	100.6	135.3		
Driver = 27x	74.2	113.94	156.89	201.91
Driver = 81x		21.6	186.6	299
Bottom trace, no LG				
W=10 μm; P=26 μM				
Driver = 3x	202.5			
Driver = 9x	211.9	272.3		
Driver = 27x	170.08	249.5	340.48	408.5
Driver = 81x		213.1	433.9	654.3
Top trace, no LG				
W=10 μm; P=26 μM				
Driver = 3x	464.71			
Driver = 9x	401	595.9		
Driver = 27x	297.95	505.64	781.6	1100
Driver = 81x		293.7	497.3	1123.4
Bottom trace, XY routing mode, no LG				
W=10 μm; P=36 μM				
Driver = 3x	7.41			
Driver = 9x	7.9	11		
Driver = 27x	7.09	11.06	18.39	32.19
Driver = 81x		28.1	56.7	105.8
hline Top trace, XY routing mode, no LG				
W=10 μm; P=36 μM				
Driver = 3x	115.75			
Driver = 9x	105.4	161.2		
Driver = 27x	67.95	123.33	202.96	313.14
Driver = 81x		58.2	101.4	160.9

values (for 81x driver) which correspond to a trace on the top layer, have been used.

1) Crosstalk noise for initial breakout pitch $= 0.19$ V,
2) Crosstalk noise for intermediate pitch $= 0.2$ V,
3) Crosstalk noise for final pitch $= 0.17$ V

which gives a total crosstalk noise of 0.56 V.

The reflection noise component comes to 0.03 V, and with an estimated simultaneous switching noise (SSN) of 0.2 V, the total noise can be obtained by computing the root sum squares of the individual components. The total noise is, therefore

$$V_{n,\text{total}} = \sqrt{V_{\text{crosstalk}}^2 + V_{\text{reflection}}^2 + V_{\text{SSN}}^2} = 0.6 \text{ V}. \quad (3)$$

The total noise is well within the noise budget of 0.7 V. The worst case off-chip skew on an 8 cm × 8 cm substrate is around 0.2 ns. After adding estimated on-chip skew component of 100 ps and a jitter component of another 0.2 ns, we can expect a cycle time of at least 2 ns. The total I/O bandwidth is then

$$\text{I/O Bandwidth} = 2000 \times 500 \times 10^6 > 100 \text{ GB/s}. \quad (4)$$

V. ARCHITECTURE

The physical architecture of the FFT system is shown in Fig. 7. The chip set contains 1 GB of memory distributed among 128 64 Mb DDR-2 DRAM chips, and four custom 1 cm^2 microaccelerator chips. The FFT is designed as a radix-64 engine, with two microaccelerator chips working together in

TABLE IV
DELAY FOR DIFFERENT LAYERS AND DRIVER SIZES (W=WIDTH; P=PITCH, LG=LOCAL GROUND). SEE FIG. 4 FOR THE MEANING OF LOCAL GROUND. THE "3x" ETC., REFERS TO THE SIZE OF THE FINAL STAGE OF THE DRIVER, AS EXPLAINED IN THE TEXT. DELAY INCLUDES INTERNAL DRIVER DELAY

Trace & driver size	0.25 cm (ns)	0.5 cm (ns)	1 cm (ns)	2 cm (ns)
Bottom trace, LG				
W=10 μm; P=26 μM				
Driver = 3x	0.84			
Driver = 9x	0.55	0.72		
Driver = 27x	0.55	0.61	0.73	0.96
Driver = 81x		0.65	0.69	0.75
Top trace, LG				
W=10 μm; P=26 μM				
Driver = 3x	0.71			
Driver = 9x	0.49	0.60		
Driver = 27x	0.53	0.57	0.65	0.81
Driver = 81x		0.64	0.67	0.72
Bottom trace, No LG				
W=10 μm; P=26 μM				
Driver = 3x	0.72			
Driver = 9x	0.50	0.61		
Driver = 27x	0.53	0.58	0.66	0.82
Driver = 81x		0.64	0.67	0.73
Top trace, No LG				
W=10 μm; P=26 μM				
Driver = 3x	0.50			
Driver = 9x	0.42	0.45		
Driver = 27x	0.50	0.51	0.54	0.58
Driver = 81x		0.62	0.63	0.66
Bottom trace, XY-mode, No LG				
W=10 μm; P=36 μM				
Driver = 3x	0.70			
Driver = 9x	0.49	0.59		
Driver = 27x	0.53	0.57	0.64	0.79
Driver = 81x		0.64	0.66	0.71
Top trace, XY-mode, No LG				
W=10 μm; P=36 μM				
Driver = 3x	0.51			
Driver = 9x	0.41	0.46		
Driver = 27x	0.50	0.52	0.55	0.6
Driver = 81x		0.62	0.63	0.65

each stage. Each of the chips, therefore, reads 32 complex numbers during each FFT computation.

For a 0.25-μm technology, it is possible to design a 32-bit multiply/accumulate unit in less than 1 mm^2. For a 0.18-μm technology, the area requirement would be even less. Therefore, it is possible to accommodate a large number of floating-point units on each chip. Each microaccelerator chip, in our design, contains an array of 64 32-bit multiply-accumulate units and 32 memory interface channels. Each of the 60-pin DRAMs are edge-mounted onto a high-density 8 cm $\times$ 8 cm SHOCC interposer substrate using a previously developed solder-bump edge mounting technique [8]. This edge-mounting technique permits chips to be mounted at low cost as no special processing steps are involved in the assembly. The only limitations are pin count (50 per cm of edge) and power dissipation (around 0.3–0.4 W due to the limited cooling path). Each memory chip is wired directly to one (and only one) memory port on a microaccelerometer chip, i.e., there is no shared memory bus, each DRAM has its own bus. The high-density substrate, thus, contains 128 independent 16-bit memory buses, which together with the control and interaccelerator bus, make up approximately 8000 total nets to be accommodated. This large amount of wiring is possible in the SHOCC technology, as it contains three signal, a power and ground layer in the substrate, with the signal layer being routed down to 20-μm pitch. Each microaccelerator has 2500 signal pins, requiring a solder bump pitch of 140 μm, again, made possible with the SHOCC technology.

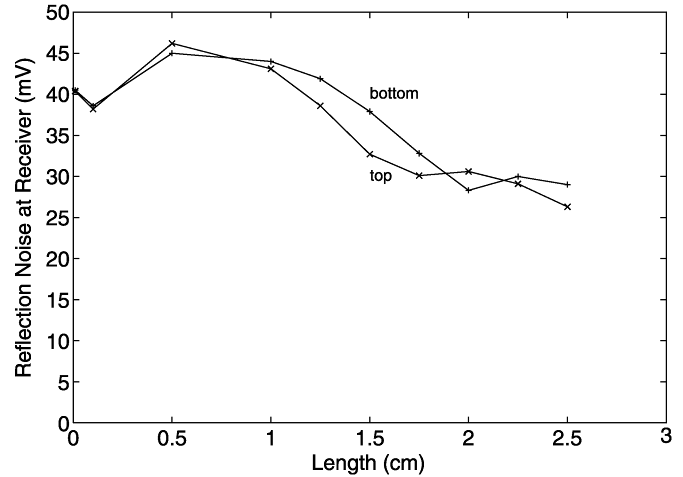


Fig. 6. Reflection noise for the final phase routing ($Aw = 10$, $p = 36$).

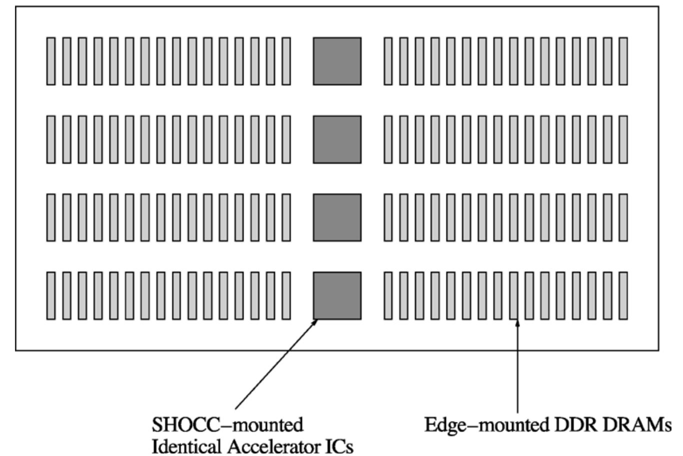


Fig. 7. Physical architecture of the FFT processor.

Each microaccelerator chip performs floating-point operations to contribute to the 64-point FFT. A million-point FFT performed in Radix-64 requires four stages to complete. To avoid bus contention for memory reads and writes, we split the memory into two sets. In the first stage of the FFT, the data is read from the first set of memory and written to the second set. In the next stage, data is read from the second set and written to the first set. This "ping-pong" action cuts the memory conflicts that would arise if data were read and written to the same set of memory. The 64-point FFT is further broken down in Radix-8 and, hence, requires two stages to compute. The first two chips perform the first stage of the FFT and the results of this stage

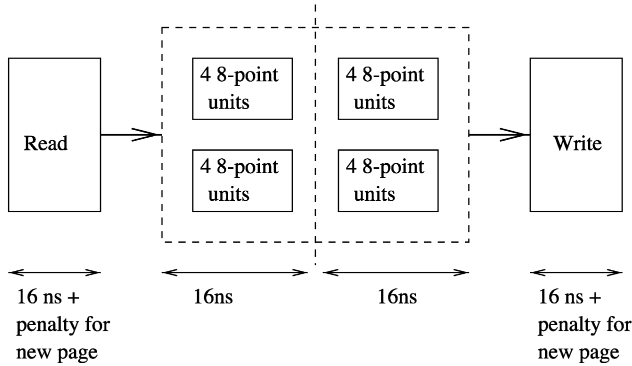


Fig. 8. Schedule of operations for the entire 64-point FFT.

are passed to the next two chips, through the high speed bus. A 64-point FFT in Radix 8 would require 32 8-point units, and these are split evenly between all the chips. Therefore, each of the two microaccelerator chips in the first stage read in 32 numbers each and perform four 8-point FFTs. After twiddling the results, the output is passed onto the next two microaccelerator chips where the sequence of operation is similar. The only difference is that the twiddle factors in the final twiddling stage are different for the chips in the first stage and those in the second. (The twiddle factors are the $T_n = e^{-j2\pi n/N}$ terms in the FFT algorithm, where $n = 0 \ldots (N/2) - 1$ [9].) The breakdown of these operations is shown in Fig. 8. By pipelining the operations as shown, a 64-point FFT can be computed every 20 ns. This is assuming that all the 64 numbers required for the FFT can be read in 20 ns.

The computation block uses data read from the DRAMs and the on-chip SRAM to compute the FFT. Details of these operations are described later in this paper. Apart from the floating-point multipliers and adders, the arithmetic block also contains units for swapping and negating data. These are used in the first two stages of the 8-point FFT, which involve twiddling with $\pm$j.

VI. Optimal FFT Algorithm

This section addresses the two main bottlenecks in the design of the high-performance million-point FFT system. In the first part, we outline a scheme for the storage and retrieval of data in DRAMs that minimizes the timing overheads of DRAMs and makes full use of all the memory bandwidth available in the design. In the second part, we address an issue specific to an FFT, that of handling the twiddle factors. By making full use of all available arithmetic units, the memory bandwidth requirement can actually be reduced.

A. Memory Addressing Scheme

The key to a successful high-volume, high-performance FFT system lies in efficient memory management. In our scheme, presented next, we stagger the results obtained from the previous stage before storing them in memory. The amount of stagger is not constant between the FFTs. This places the data in the correct memories for the next stage. The algorithm is not in-place, i.e., data is not stored in the same location from which it was read, but in a different order in another set of memory. This allows us to efficiently pipeline the READs and WRITEs

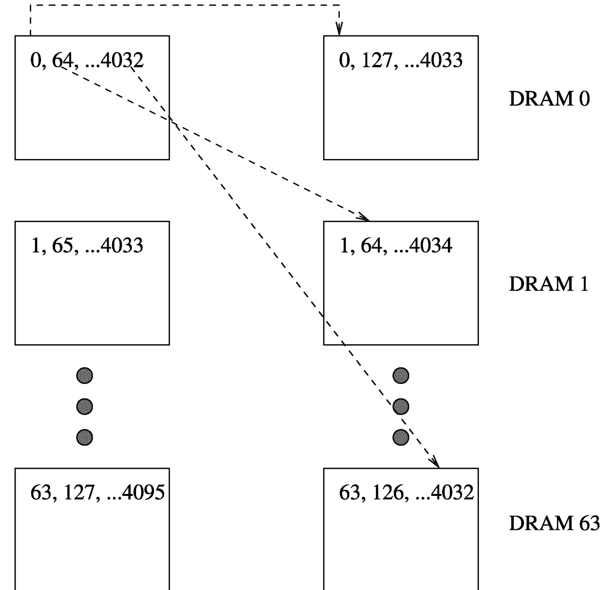


Fig. 9. Memory layout of data after the first stage.

to memory. Our scheme is closest in nature to the scheme in [10], [11], where data in memory is mapped according to the stride of the algorithm. The main difference is that since the stride is not constant between different stages of the FFT, an additional rotation is provided before data is stored in order to maintain the same stride. In addition, our memory mapping scheme also takes care of the DRAM-related issues like the precharge/refresh times.

For the Radix-64 case, the memory mapping scheme can be extended, and is given by the following relation:

$$\text{DRAM_No} = (\text{FFT_No} + \text{index})\%\ 64 \tag{5}$$

where $\text{FFT_No} = \lfloor(\text{index}/64)\rfloor$ and index refers to the index number of the data $(0 < \text{index} < 1\,048\,575)$.

The memory allocation for any stage is given in Fig. 9. This scheme introduces a stagger of 64 required in each stage and can be easily implemented using shift-registers at the input and output.

1) Memory Mapping of Data Into Different DRAMs: The above scheme would work perfectly if an ideal memory was used, i.e., if a particular data from any row/column address could be read in the given time. This is not the case with conventional DRAMS where the row precharge times can add significant timing overheads. For our design, we have used DDR SDRAM (MT46V4M16) from Micron Semiconductor Products Inc.[1] The DDR SDRAM can read or write data at both the rising and falling edge of the clock, which makes it faster than other SDRAMs. In the chosen DDR SDRAMs, a random access request takes 60 ns to serve (compared with less than 5 ns for modern SRAMs). Thus, if the data accesses were all truly random the total memory bandwidth of our system would be 15 Gb/s and the FFT performance unacceptably low. The total memory is divided into four banks, and each row is of 4096 bits. Once a row in a particular bank has been activated,

[1] 128-Mb DDR SDRAM Datasheet. http://www.micron.com/products/datasheets/ddrsdramds.html

TABLE V
SOME OF THE TIMING PARAMETERS ASSOCIATED WITH DDR SDRAM

Parameter	Value
ACTIVE to ACTIVE/ AUTO REFRESH Command Period	60ns
ACTIVE to READ or WRITE Delay	15ns
ACTIVE bank A to ACTIVE bank B command	15ns
ACTIVE to PRECHARGE command	45ns
PRECHARGE Command Period	15ns

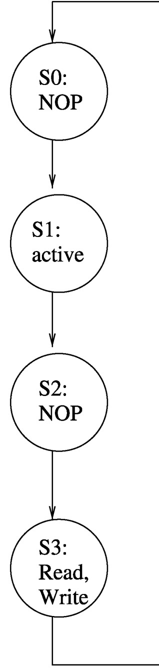


Fig. 10. State diagram showing the timing of different commands.

data from that row can be read or written every 5 ns. However, a minimum burst size of two is required for this. Another bank in the same memory can be activated while the first bank is still active after a time of 15 ns. The various relevant timing parameters have been summarized in Table V.

To minimize the overheads due to precharge times, bank, row, and column addresses have to be generated in a fashion that hides the row activation time. The state diagram in Fig. 10 shows the relative timing of the active and read/write commands. Each state takes 5 ns, and one FFT can be completed in 20 ns. The reads and writes are always in a burst of four and once the read/write command is issued data appears on the data bus every 5 ns after an initial latency of two clock cycles. The key point to note here is that an active command can be given to another bank while data is being read/written to the previous bank. In our scheme, a 64-bit complex data is written in four adjacent columns and, hence, data is always accessed in a burst of four. An extra state of no operation (NOP) is required between the active command and the read/write command. This is because commands can be registered only at the positive clock edge even though data could appear on the data bus on both edges of the clock.

The scheme for generating bank and row addresses is different for reads and writes and is summarized below.

Reads:

$$\text{Row_No} = \lfloor \text{FFT_No}/4 \rfloor \tag{6}$$
$$\text{Bank_No} = \text{FFT_No} \,\%\, 4. \tag{7}$$

Writes:

$$\text{Row_No} = \lfloor \text{FFT_No}/256 \rfloor \tag{8}$$
$$\text{Bank_No} = (\lfloor \text{FFT_No}/64 \rfloor) \,\%\, 4 \tag{9}$$

where FFT_No $= (\text{ index}/64)$ as before. A row hop takes place after every four FFTs while reading and after every 256 FFTs while writing. A bank hop takes place after every FFT while reading and after every 64 FFTs while writing.

B. Twiddle Factor Generation Scheme

The other point of consideration in the design of the FFT system is the handling of the twiddle factors. For most small DSP systems, the twiddle factors can be stored in on-chip memory. This works fine for smaller systems but would require huge amounts of memory for large systems, like the one under consideration. For a million-point FFT under consideration, a million twiddle factors would have to be stored in memory. This would not only double the physical memory requirement but also the number of memory channels. The other alternative, which we discuss next, is to generate the twiddle factors needed for the FFT on the chip itself. As discussed before, computing an FFT in Radix 64 is equivalent to breaking down one row of numbers into a two-dimensional (2-D) array of 64 rows. This can be expressed as [9]

$$X(s,r) = \sum_{m=0}^{M-1} W^{Lmr} W^{ms} \sum_{l=0}^{L-1} x(l,m) W^{Msl} \tag{10}$$

where W^{ms} is the twiddle factor required to twiddle the 64-point FFT results for the next stage computations and is given by

$$W^{ms} = e^{-j\left(\frac{2\pi}{N}\right)ms} \tag{11}$$

For a Radix-64 FFT, *L* is 64 in the equation for the 2-D representation. The sequence of operations is then:

1) compute the 64-point FFT of each column;
2) twiddle the results with W^{ms};
3) compute the FFT of each row.

For the twiddle factors, s varies from 0 to 63 in all stages, while W^{ms} takes on values depending on the stage. For our twiddle factor generation scheme, we store an initial 64 twiddle factors (corresponding to $m = 1$ in the previous equation) in on-chip SRAM. Twiddle factors for the any FFT can be generated from the twiddle factors of the previous FFT and the initial twiddle factor set. This can be seen from the following equation:

$$W_N^{ms} = W_N^{(m-1)s} W_N^{1,s}. \tag{12}$$

The initial twiddle factor set for the next stage can be generated in the previous stage by storing the twiddle factor values

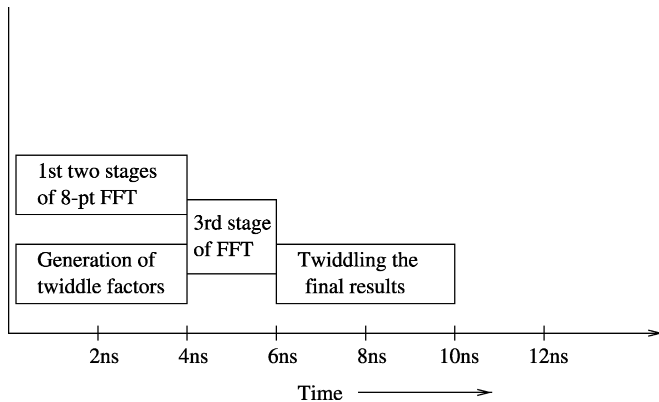


Fig. 11. Sequence of operations for the FFT engine.

corresponding to $m = 64$. This can be seen from the equation given below

$$W_N^{64.s} = W_{\frac{N}{64}}^{1.s}. \quad (13)$$

In this way, with a relatively small amount of on-chip memory, all the twiddle factors can be generated on-chip, and no extra memory bandwidth is required. This assumes that the twiddle factors for the next stage can be computed in the previous stage, along with the computations of the FFT. For an FFT systems, this is possible because the first two stages of the eight-point FFT computation does not require any multiplications and the 64 32-bit floating point multipliers can be used in generating the twiddle factors for the next stage. The sequence of operations is shown in Fig. 11. All the computations required for the FFT can be completed well within the budget of 20 ns.

VII. Conclusion

By utilizing the SHOCC thin-film MCM, flip-chip solder bump technology, we were able to provide a high memory bandwidth for the FFT engine. With five metal layers and a solder bump pitch of 140 μm, an I/O bandwidth of over 100 Gb/s/cm^2 chip could be achieved with sufficiently good noise control. A key element to such high I/O bandwidth was the use of a redistribution layer with a local ground. Further improvements in system performance were obtained by using an efficient memory management scheme that allowed DRAMs to be used in place of expensive SRAMs. The FFT engine computes a 64 point FFT in 20 ns. A million-point FFT, can be done in four stages, with 16 384 64-point FFTs in each stage. The total time to compute the million-point FFT is, therefore, 1.31 ms. The performance capability of our system is as follows.

1) Peak performance of 2.56×10^{11} (if all arithmetic units were 100% occupied).
2) Peak memory bandwidth of 47.6 GB/s (i.e., if all DRAMs were operating continuously in burst mode).
3) Sustained floating point performance of 8.64×10^{10} FLOPS, corresponding to 763 million-point FFTs/s.
4) Sustained memory performance of 47.6 GB/s.

We compared the performance of our FFT engine with two other systems. The first one uses four BOPS Inc. DSP chips.[2] The system has four 32-bit memory channels. Each chip has four procesing elements (PEs), and each of the PEs has five floating point (FP) units. This system would take up approximately 80 cm^2 of PCB and would perform a million-point FFT in 21.5 ms, more than an order of magnitude slower than ours. We also compared our engine with a G4 Velocity Engine implementation of the FFT, using the Motorola's Altivec technology. Computing a million-point FFT on their system takes 511 ms, almost 400X slower than ours [12].

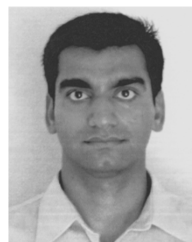
[2]http://bopsnet.com/cores

References

[1] M. Dibbs, P. Garrou, C. Chau, Y. So, D. Frye, J. Wagner, J. Ousley, G. Baugher, J. Santandrea, G. Connor, J. MacPherson, G. Adema, P. Dean, L. Schaper, R. Eden, and R. Sands, "Development of seamless high off-chip connectivity," in *Proc. SPIE Int. Symp. Microelectron.*, vol. 3235, Oct. 1997, pp. 138–143.
[2] L. Schaper, M. Dibbs, P. Garrou, C. Chau, Y. So, D. Frye, J. Wagner, J. Ousley, G. Baugher, R. Pickard, G. Connor, D. Winn, and P. Deane, "Seamless high off-chip connectivity," in *Proc. IC/Package Design Integration Symp.*, 1998, pp. 39–44.
[3] *Maxwell Q-3D User Guide*, Ansoft Corporation, Pittsburgh, PA.
[4] S. Afonso, W. D. Brown, L. W. Schaper, and J. P. Parkerson, "Signal propagation on seamless high off-chip connectivity," in *Proc. IEEE 7th Topical Meeting Electrical Performance Electronic Packaging*, 1998, pp. 31–34.
[5] S. Afonso, L. W. Schaper, J. P. Parkerson, W. D. Brown, S. Ang, and H. A. Naseem, "Modeling and electrical analysis of Seamless High Off-Chip Connectivity (SHOCC) interconnects," *IEEE Trans. Adv. Packag.*, vol. 22, no. 3, pp. 309–320, Aug. 1999.
[6] R. K. Poon, *Computer Circuits Electrical Design*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
[7] A. Deutsch, G. V. Kopcsay, C. W. Surovic, B. J. Rubin, L. M. Terman, T. A. Gallo, and R. H. Dennard, "Modeling and characterization of long on-chip interconnections for high-performance microprocessors," *IBM J. Res. Develop.*, vol. 39, pp. 547–566, Sep. 1995.
[8] M. Nakkar, A. Glaser, P. Franzon, K. Williams, M. Roberson, and G. Rinne, "Three dimensional MCM package assembly and analysis," in *Proc. IEEE/IMAPS Conf. High Density Packaging and MCMs*, 1999, pp. 188–192.
[9] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975, ch. 6.
[10] D. T. Harper, "Block, multistride vector, and FFT accesses in parallel memory systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 2, no. 1, pp. 43–51, Jan. 1991.
[11] ——, "Conflct-free vector access using a dynamic storage scheme," *IEEE Trans. Parallel Distrib. Syst.*, vol. 40, no. 3, pp. 276–283, Mar. 1991.
[12] R. Crandall and J. Klivington. (1999) Supercomputer-Style FFT Library for Apple G4. [Online] Available: http://www.motorola.com/

Pronita Mehrotra (M'98) received the Master's degree in electrical engineering from the Indian Institute of Technology, Bombay, India, in 1997 and the Ph.D degree from North Carolina State University, Raleigh, in 2002.

She was a Staff Research Engineer in the Advanced Networking Research group at MCNC-RDI, Research Triangle Park, NC, where her focus was on Optical Burst Switched Networks. Her current research interests are in VLSI design, advanced packaging, and algorithms.



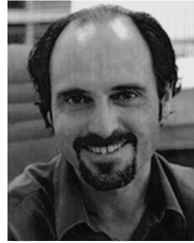
Vikram Rao received the B.S. degree in electrical engineering from the India Institute of Technology, Bombay, India, in 1998 and the M.S. degree in computer engineering from North Carolina State University, Raleigh, in 2000.

Since then, he has been a Staff Engineer in the Compiler Group at Sun Microsystems, Mountainview, CA.

Thomas M. Conte (S'84–M'92–SM'99) received the B.S. degree in electrical engineering from the University of Delaware, Newark, in 1986 and the M.S. and Ph.D. degrees in electrical engineering from the University of Illinois at Urbana-Champaign in 1988 and 1992, respectively.

He is currently a Professor of Electrical and Computer Engineering and Director of the Center for Embedded Systems Research at North Carolina State University, Raleigh. While on leave from NC State in 2000–2001, he served as the Chief Microarchitect and Manager of Back End Compiler Development for DSP vendor BOPS, inc. Mouintain View, CA. He is Editor-in-Chief of the *Journal of Instruction-Level Parallelism*, and an Associate Editor of both the *ACM Transactions on Architecture and Compiler Optimization* and the *ACM Transactions on Embedded Computer Systems*. He is also the chair of the IEEE CS Technical Committee on Microprogramming and Microarchitecture (TC-uARCH). He currently directs several Ph.D. students on the TINKER project in topics spanning advanced microarchitecture, compiler optimization, and performance analysis. His research is supported by Compaq, DARPA, HP, IBM, Intel, Sun Microsystems, TI, and the National Science Foundation.

Dr. Conte is the recipient of an NSF CAREER Award and the IBM Thomas J. Watson Partnership Award for Faculty Development.



Paul D. Franzon (SM'99) received the Ph.D. degree from the University of Adelaide, Adelaide, Australia, in 1988.

He is currently a Distinguished Alumni Professor of Electrical and Computer Engineering at North Carolina State University, Raleigh. He has also worked at AT&T Bell Laboratories, DSTO Australia, Australia Telecom, and two companies he cofounded, Communica and LightSpin Technologies. His current interests center on the technology and design of complex systems incorporating VLSI, MEMS, advanced packaging, and molecular electronics. Application areas currently being explored include novel advanced packaging structures, network processors, SOI baseband radio circuit design for deep space, on-chip inductor and inductance issues, RF MEMS, and moleware circuits and characterization. He has lead several major efforts and published over 120 papers in these areas. In 1993, he received an NSF Young Investigators Award, in 2001, he was selected to join the NCSU Academy of Outstanding Teachers, and in 2003, he was selected as a Distinguished Alumni Professor.