Issues in Partitioning Integrated Circuits for MCM-D/Flip-Chip Technology

Sanjeev Banerjia, Alan Glaser, Christoforos Harvatis, Steve Lipa, Real Pomerleau, Toby Schaffer, Andrew Stanaski, Yusuf Tekmen, Grif Bilbro, and Paul Franzon

Department of Electrical and Computer Engineering, Box 7911
North Carolina State University, Raleigh, NC 27695

Abstract.

In order to successfully partition a high performance large monolithic chip onto MCM-D/flip-chip-solder-bump technology, a number of key issues must be addressed. These include the following: (1) Partitioning a single clock-cycle path across the chip boundary within timing; (2) Ability to use off-the-shelf memories; (3) Using the MCM for power, ground, and clock distribution; and (4) Managing test costs. This paper presents a discussion on these issues, using a CPU as an example, and speculates on some interesting possibilities arising from partitioning.

1 Introduction

It is clear that there are potential cost and performance advantages to be gained from partitioning a large chip by using multichip module technology [4, 3, 2]. However, there are a number of practical issues that need to be resolved before such a methodology becomes practical. These include the following: (1) Partitioning a single clock-cycle path across the chip boundary within timing; (2) Ability to use off-the-shelf memories; (3) Using the MCM for power, ground, and clock distribution; and (4) Managing test costs. In this paper, these issues are explored using a 6-issue superscalar CPU as an example.

A 6-issue superscalar MIPS R2000/3000 CPU has been designed to the behavioral Verilog level and is partially implemented at the layout level. The microarchitecture of the CPU is shown in Figure 1. This design is too large to fit on a monolithic die with current technologies, so partitioning it using a high performance MCM package is the only alternative. Even if the design was reduced so that it could fit in a reticle, the yield would be unacceptably low and the per-unit cost too high. In references [3, 4], we illustrate the significant cost advantages to be gained by partitioning a large die set such that each die is about 1 sq. cm. in area.

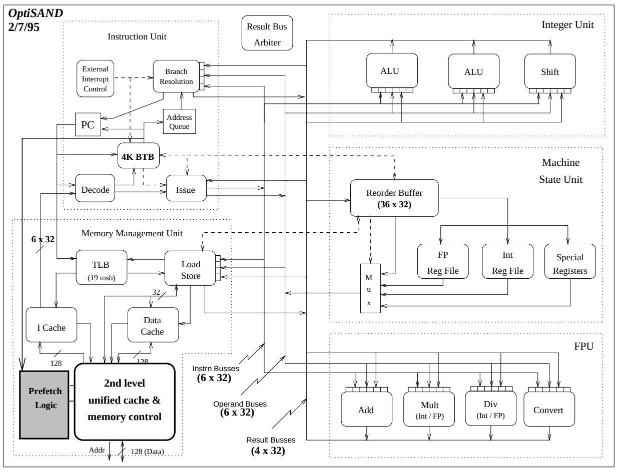

Figure 1: MicroArchitecture of the 6-issue MIPS R2000/3000 CPU.

In this paper, we investigate the application of a dense MCM technology (see Figure 2) to this problem.

2 Maintaining Timing Across Chip Boundaries

Two feasible partitions are shown in Figures 3 and 6. The less aggressive partition, shown in Figure 3, is not too dissimilar to that carried out commercially by HAL Computers.

One important item to note in both of these partitions is that the Level 1 Instruction Cache (I–Cache) and Data Cache (D–Cache) can be built in an SRAM process rather than in a modified logic process. SRAM built in an SRAM process is about twice as dense and twice as fast as SRAM built in a logic process.

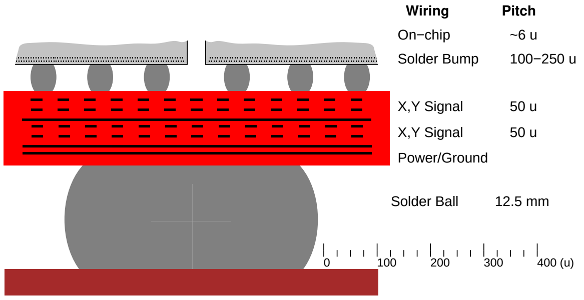

Figure 2: High Density MCM-D/flip-chip technology being targeted.

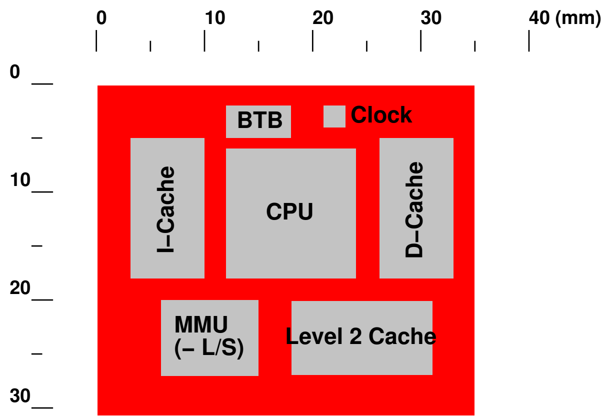

Figure 3: A feasible partitioning for the optimized processor core.

The most aggressive aspect of the first partition is that the I–Cache access must be performed in a single clock cycle. To achieve a single cycle round trip, it is necessary to floorplan the I–Cache and CPU and determine solder bump locations so that the interconnect delay from the instruction unit to the address logic, and from the sense amplifier back to the instruction unit, is only a small part of the clock cycle. The potential danger is that breakout from the on-chip cell through the solder bump to the routing channel might increase the path length, and thus the delay, by too much.

Figure 4 shows a floorplan that will minimize the effect of the breakout on clock cycle time. Figure 5 shows part of the breakout pattern from either the SRAM or the CPU chip. In this breakout pattern, the solder bumps are on a 200 μm pitch. For each end of the pattern, there are 228 signal I/Os and 228 power/ground pins. With 4 signal layers, eight rows of signal pins can break out in one direction (only four rows and the top layer of routing is shown in Figure 5). Thus an array of 29 by 16 solder bumps is needed, taking up 5.6 $\times$ 3.2 mm.

As this solder bump array is larger than the I/D unit, on-chip routing will be required for the breakout

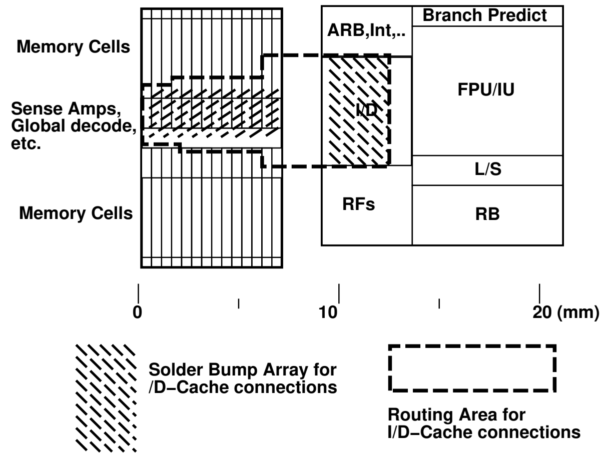

Figure 4: CPU and I–cache chip floorplans. The Instruction/Decode unit communicates with the I–cache.

on the CPU side. Early estimates indicate that up to 3 mm might be needed.

In the SRAM, a 32 by 14 solder bump array is used, consuming 6.4 $\times$ 2.8 mm of area. The longest on-SRAM connection (to a sense amplifier) is about 2 mm.

The longest on-MCM path in this connection is 9 mm. The I–Cache to Instruction/Decode unit input register path thus has a total length of 14 mm. This path is driven by a single driver, and simulation results indicate that the total delay would be about 600 ps. In comparison, the I–Cache to Instruction/Decode unit input register delay was about 100 ps before partitioning. Crosstalk, at 3.2%, is not a problem in this signal path.

Not shown in these figures are the I–Cache to Level-2-Cache refill connections. Though not a critical path, these connections need to be 196 bits wide and thus will consume considerable interconnect resources both on the I–Cache and on the MCM. These would be placed below the pads shown on the I–Cache.

In conclusion, this partitioning is feasible, though with a 500 ps latency penalty. This penalty is more than compensated by the speed-up obtainable by building the caches in an SRAM technology. In further work, we plan to build part of these circuits and more precisely determine the optimal floorplan and circuit structures.

It is worth noting that the MCM technology is a significant limiting factor in this partitioning. If a two-signal layer MCM-D technology is used, instead of the four-layer technology analyzed, then the delay of the I–cache to I/D unit path would be increased by about 200 ps to 800 ps. The routability of the two-signal layer example was actually limited more

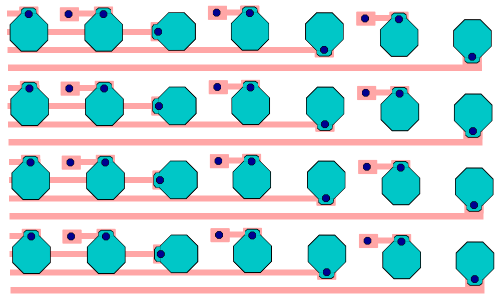

Figure 5: Details of part of the breakout pattern for the I/D to I–cache connection. Only the top signal layer is shown. Another, lower signal layer permits a similar array of bumps to the right of these to be broken out.

by the available via density than the wiring density. The vias in the target technology require 42 μm wide land pads. They only permit a single layer to be used for river-routing of this bus. If a smaller via technology was available (smaller than 34 μm), then both signal layers could have been used to route these X-direction buses (with the wires shifted by 25 μm on alternate layers in order to minimize crosstalk). If the solder bump pitch could be shrunk to 100 μm, then this delay would be decreased by over 200 ps. If the solder bump pitch could be shrunk to 50 μm, then about a further 100 ps of delay could be regained.

2.1 Using Off-The Shelf SRAMs

The above analysis presumes a custom designed memory. Using off-the-shelf memories brings additional cost advantages.

Volume SRAM chips are now available in the sub-10 ns cycle time range in sizes up to 32 K. Assuming that chips with smaller memory capacities and similar cycle times will be available in the near future, such chips can be used to build the data array for a cache for an MCM based microprocessor. The advantages are reduced cost due to elimination of full–custom design circuits for the 1st-level cache and potential design time savings, although an initial effort would be needed to integrate the SRAM chips with additional combinational "glue" logic.

We examine using SRAMs by investigating what would be needed for the cache tags and tag checking logic separately from how actual cache data could be stored. Specifically, support for a six issue machine is outlined.

SRAMs are available in a variety of sizes and data widths, so they could be used to construct virtually any type of cache, at least from the perspective of storage and bandwidth requirements. However, wide data paths are not available yet; the maximum data path is currently 36 bits, with four of those bits reserved for ECC. For an instruction cache, only one 32-bit instruction word can be fetched per cycle from a given chip. One way to organize the data array is to store each word of a cache block into a separate chip at the same address in each chip. For a six issue design, six chips are needed. A cache size that is a multiple of the cache block size should be used, so a size such as 48 kB is appropriate. Advanced designs such as a banked I–cache can be supported also but would benefit from the data path to each individual chip being wider, say, at least 64 usable bits of data. A data cache could be built similarly. Storing one word per chip actually simplifies the implementation of some features, such as sub–block replacement.

The cache tag array can be a full-custom design or perhaps can be built using off-the-shelf CAM chips. Assume that a full-custom design is used. The operation of the tag logic is straightforward and independent of the data array. There is additional logic required to index into the both the tag and data arrays, however. This (combinational) logic must be custom-designed and must be able to translate an address presented to the cache into one that can address the data array. This task is also straightforward. The same address is presented to all of the data array chips if we make the assumption that the individual words of a cache block are scattered and stored at the same address across the separate chips, one word per chip. Based on the tag compare, the words are retrieved from the data array; this is easily done as each chip is addressed independently of the others.

The translation of the physical/virtual address presented to the cache can be done very quickly, in one level of logic, and most likely will not impact the cycle time of the processor. Using SRAM chips to build caches is then both practical and economical.

2.2 More Aggressive Partitioning

A more aggressive partitioning is shown in Figure 6. This partitioning offers the interesting possibility of being able to mix and match different execution units to give different processor implementations from the same chip set. For example, one could add or delete multimedia features or add or delete extra functional units.

In our current 100 MHz, 0.8μm implementation, we allow 2 ns for the bus transit time. Simulation results (Figure 7) show that, with a proper driver choice, this delay is quite achievable with the bus dropped onto the MCM.

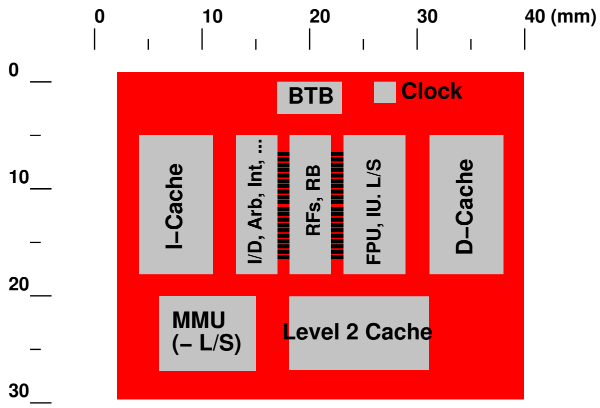

Figure 6: A more aggressive partitioning.

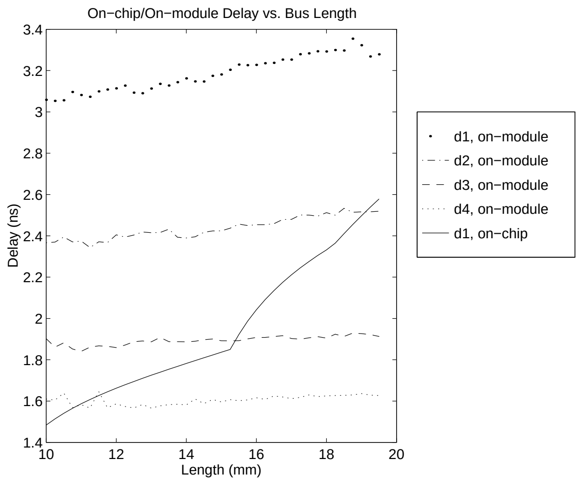

Figure 7: Simulation results for different driver strengths for on-chip and on-MCM implementations of the 10-load main CPU result and operand buses.

3 Using the MCM for Global Clock, Power and Ground Distribution

Interconnect traces on an MCM-D substrate behave like lossy transmission lines as opposed to the RC behavior of on-chip interconnect. This difference in behavior leads us to explore the distribution of global clock signals via MCM interconnect rather than IC interconnect. In addition, using an MCM substrate with area-array solder bumps provides multiple entry points distributed over the IC circuitry, many more than would be available using standard peripheral pads. This presents the opportunity not only to dramatically increase the I/O capacity but also to provide local power and ground through the bumps rather than global power and ground via on-chip rails. (Small on-chip global power and ground connections might still be required to provide a path for signal return currents.) The term "local" is loosely defined – it depends upon the fraction of bumps available for power and ground connections; this is discussed further below.

As a vehicle for these experiments, we have designed a full-custom implementation of the ANSI Data Encryption Algorithm, commonly referred to as DES, in a 0.6μm, three-metal silicon technology. The chip consists of 16 identical *rounds* as shown in Figure 9. Data flows in in a "U" shape, where each round is a pipeline stage. To minimize the interconnect length between rounds eight and nine (the bottom of the "U"), the two columns must be skewed, i.e., rounds are not aligned horizontally.

3.1 On-MCM Clock Distribution

Standard on-chip clock distribution nets, e.g., H–trees, suffer from the RC parasitics of on-chip wiring in addition to consuming large amounts of routing capacity on the metal layer used. Routing the clock net on the MCM substrate, and providing local entry points on-chip through the solder bumps at the leaves of the clock tree, effectively eliminates one stage (driver and wiring) in the on-chip clock distribution. For our design example, we simulated the entire clock distribution (including chip attachment parasitics) using an on-MCM H–tree with 16 leaf entry points (one for each round) and local on-chip distribution after each entry point. We designed the driver for the clock distribution network on the MCM substrate by appropriately sizing it, so that a good clock edge is available at the entry points (the driver giving the 'squarest' waveform in Figure 8). It was *not* necessary to size the line widths in the H–tree; we used minimal line width for all branches, greatly reducing the effect of the H–tree on the routing underneath the chip. The H–tree itself is insensitive to load mismatches at the leaves; an extreme case where one of the leaves was loaded twice as much as the others resulted only in 10 ps skew. This implies that a virtually skewless clock can be guaranteed at the entry points, if the distribution network is carefully designed. On-chip drivers are located in the vicinity of the entry points and deliver the clock signal to balanced loads. Skew is further reduced by removing one layer of on-chip clock buffering.

3.2 On-MCM Power and Ground Distribution

In a conventional peripheral pad design, large power and ground rails must reach every part of the chip. As shown in Figure 9, there is a power rail on either side of the logic and a ground rail in the middle. Horizontal metal fingers then provide power and ground to the individual rounds.

The situation is different in the case of local power distribution through area-array solder bumps. As

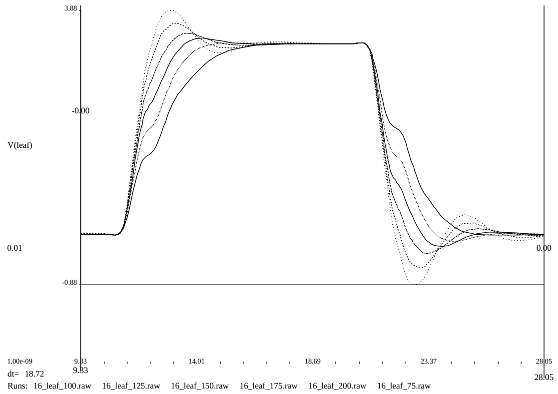

Figure 8: Clock waveforms at chip entry points for different driver sizes.

noted above, the definition of "local" power and ground distribution is design-dependent; in our case, the density of the solder bumps is sufficient to allocate one bump each for power and ground per round. (This capacity is available even after all I/O and clock signals have been accounted for.) While distributing power and ground in this manner does not change the size of the horizontal fingers, it eliminates the need for the vertical rails altogether, as shown in Figure ??. For this example, the area saved by removing these rails is approximately 4.67 sq. mm, or a 34% reduction. In both cases, the maximum peak IR drop meets the design requirement of being 5% or less.

4 Managing Test Costs and Verification

In light of the the preceding discussion, cost effective methods for testing bare, high I/O chips that have been electrically designed specifically for use on an MCM are needed. These methods must be able to prove the chips function correctly and determine the speed rating of each chip. In addition, it is desirable to use existing VLSI testers to perform the tests.

The physical interface issue is being addressed through the development of new test head technologies, such as membrane probes. Ideally the probes and chips would be designed so one probe could be used to test all the dice. However, in a high part count situation, the additional tooling cost of unique probe heads would only add a small overhead. A more significant problem is matching the chip electrical environment to the tester electrical environment. The chips, designed to run exclusively on the MCM, will not have enough drive strength to communicate a fast signal to the tester. Also the number of I/Os

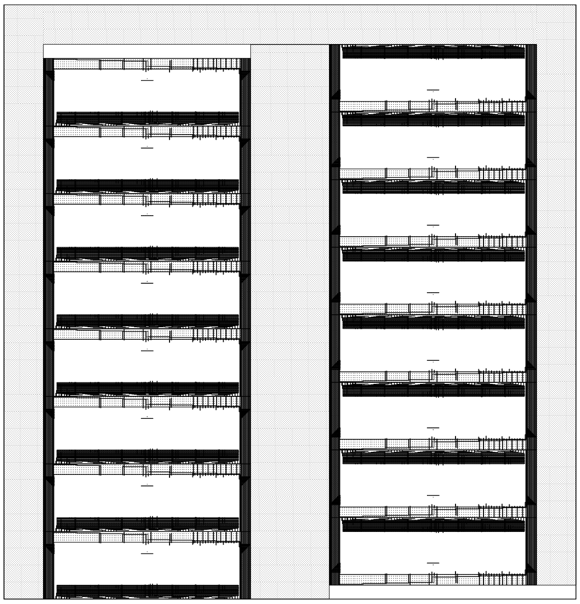

Figure 9: Floorplan using conventional (peripheral-pad) power distribution.

from the chip is likely to be larger than the tester can accommodate. Thus some active circuitry, drivers and multiplexers at least, will be needed on the test head.

Placing active circuitry on the test head presents an opportunity to extend the capability of the VLSI tester at low cost. In addition to drivers and multiplexers, pattern generation and compaction as well as IDDQ measurement can be incorporated in the test head. This allows the test head to perform some of the testing autonomously and reduces the memory requirement of the VLSI tester. Also, since the test head can be running independent of the tester for a period of time, multiple test heads can share the same tester. Thus, a chip designed for MCM implementation will require less tester memory and lower test time, at the expense of using a smart test head.

There is another issue for flip-chip MCM systems: failure diagnosis, particularly of delay faults. When a chip is flipped it is no longer accessible for voltage contrast probing. Failure diagnosis capability can be improved by the use of full scan. Sun uses this effectively on their SuperSparc 2 and UltraSparc chips [6, 5]. Also, voltage contrast probing can be accommodated in the test head by etching holes in the membrane probes. By making a set of membrane probes together with chip versions with different pad locations, it should be possible to provide 100% access. This provides the needed coverage during the development process. When the chip is mature, a single probe and chip arrangement is adequate for

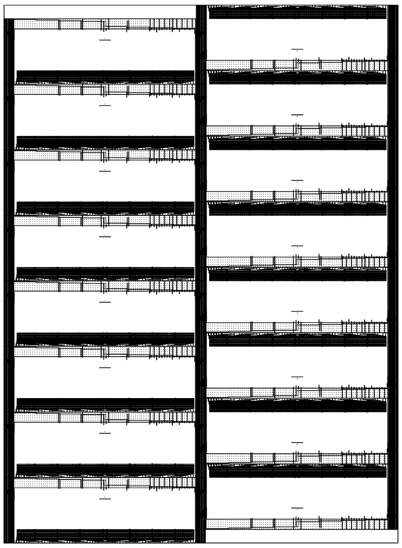

Figure 10: Floorplan using solder-bump power distribution.

functional and speed testing.

5 Discussion and Conclusions

Previous results have shown partitioning a large, low-yielding die into a number of smaller (about 1 sq. cm) dice can bring significant cost advantages [3, 4]. Furthermore, if the partition permits memory structures to be implemented in an SRAM technology, or with off-the-shelf SRAMs, then further cost savings can be incurred. Further advantages can be gained if the MCM is also used for on-chip power and ground distribution, thus releasing on-chip metal for interconnect, and for on-chip clock distribution, saving one level in the clock distribution tree. In the example given, we saved 34% of the original area by eliminating large global power rails and designed an on-MCM H–tree without adversely affecting clock skew.

However, an effective partition can not be permitted to add significant timing or area penalty. In the case study explored in this paper, the timing penalty for splitting the I–Cache, out of the CPU, was found to be 500 ps and the area penalty about 1 sq. mm. These penalties are acceptable, given that they are more than compensated for by the extra speed and area gained by building the I–Cache in an SRAM technology.

Another issue that must be resolved for partitioning to be effective is test. The dice must be speed-binned before assembly if the final module is to meet speed targets.

Finally, it is worth mentioning that the MCM technology itself does limit the advantages of partitioning. The delay and area penalty of partitioning would have been further reduced if a smaller solder ball pitch was permitted (e.g. 100 μm instead of 200 μm). Two layers of signal interconnect could have been used (instead of four) if the via size could have been halved.

Acknowledgments

The authors wish to thank the following funding and support sources: ARPA under contract DASH04-94-G-003-P2, NSF under grants MIP-901704 and DDM-9215755, NSF for Dr. Franzon's NSF Young Investigator's Award and Toby Schaffer's NSF Graduate Fellowship, and Intel Corporation. The authors also wish to thank Robert Frye, Thad Gabara, Wayne Nunn, Scott Alvarez, Steve Mozgai, Bill Schmidt, Frank Swiatoweic, Evan Davidson, and Gary Dudeck.

References

[1] ANSI. *American National Standard (X3.92-1981) Data Encryption Algorithm*, 1981.

[2] P. Dehkordi, K. Ramamurthi, D. Bouldin, H. Davidson, and P. Sandborn. Impact of packaging technology on system partitioning: A case study. In *1995 IEEE MCM Conf.*, pages 144–151, 1995.

[3] P.D. Franzon, A. Stanaski, Y. Tekmen, and S. Banerjai. System design optimzation for mcm. In *1995 IEEE MCM Conf.*, pages 138–143, 1995.

[4] P.D. Franzon, A. Stanaski, Y. Tekmen, and S. Banerjia. System design optimzation for mcm. *Trans. CPMT*, pages 620–627, December 1995.

[5] H. Hao and R. Avra. Structured design-for-debug - the supersparc ii methodology and implementation. In *Proceeings of the 1995 International Test Conference*, pages 175–183, 1995.

[6] M. Levitt, S. Nori, S. Narayanan, G. Grewal, L. Youngs, A. Jones, G. Billus, and S. Paramanandam. Testability, debuggability, and manufacturability features of the ultrasparc-i microprocessor. In *Proceeings of the 1995 International Test Conference*, pages 157–166, 1995.