White Paper

SYNOPSYS®
Predictable Success

Multicore and Distributed Processing With TetraMAX® ATPG

Delivering the fastest time-to-results

July 2009

Cy Hay
Product Manager,
Synopsys

Introduction

Running automatic test pattern generation (ATPG) on a single processor may take a week or longer to complete, especially for very large designs and when testing at-speed fault models. Designers and test engineers need a straightforward way to reduce ATPG runtime by many factors and deliver working test patterns in days, not weeks. Accelerating this step frees up valuable engineering resources and removes ATPG from the critical path for taping out a design to manufacturing.

Multicore processors are now widely available, but require additional tool support to maximize the compute power of these platforms. Synopsys introduces new multicore processing support in TetraMAX ATPG. This support extends the runtime benefits of distributed processing to multicore platforms, while significantly reducing the memory consumed compared to distributed processing. In this white paper, we describe how TetraMAX 2009.06 can immediately accelerate APTG runtimes by 3X or more.

ATPG and Parallelism

TetraMAX ATPG generates efficient test patterns using a widely accepted technique which is generically described below:

1. Populate the fault list.
2. Select a fault and generate a test for that fault, or prove it untestable.
3. Select another fault and generate a test for that fault if it can be merged into the current pattern.
4. Repeat step 3 until a certain limit is reached.
5. Fault simulate and store the final test pattern.
6. Drop all detected faults from the fault list.
7. Repeat steps 2 through 6 until every fault has been either selected in step 2 or detected in step 5.

There are several opportunities to utilize coarse-level parallelism in this process. The most general is to split the fault list among multiple CPUs running in parallel. Each CPU only targets 1/*N*th the number of faults compared to all *N* number of faults in a serial process. To maintain high efficiency and the fewest number of patterns, it is necessary for each CPU to immediately broadcast fault status changes to all the other CPUs.

Multicore Processing

Today's workstations and servers contain multiple CPUs. Typically these are dual-core or quad-core CPUs on a single device, and high-end servers may be configured with multiple such devices. To support these multicore architectures, the ATPG process is partitioned into a parent process (referred to as the "master") and multiple child processes (referred to as "slaves".)

- The master process spawns slave processes and establishes inter-process communication using shared memory.
- Each slave process targets different faults to not overlap with the other slave processes.
- Each slave process updates global fault status changes to all other processes.
- Each slave process transfers test patterns to the master process after fault simulation.

ATPG runtimes improve almost linearly up to the number of CPUs available on a multicore system as long as the system has available bandwidth between CPUs and the memory subsystem. Test pattern generation and fault simulation steps on large designs rapidly traverse data structures which are much bigger than the CPU cache. This behavior generates frequent access to the system's main memory. Ultimately the memory subsystem must be able to handle simultaneous operations from multiple CPUs with minimal stall cycles.

Employing parallel ATPG techniques on a shared-memory architecture makes efficient use of the available system memory. Additionally, shared memory enables zero-latency and high-bandwidth communication among slave processes. With TetraMAX multicore processing support, the additional memory consumed per additional slave is only for the private memory needed to store temporary data structures that are regularly written and modified during the ATPG process. For example, these would include pattern buffers during test generation and node values during fault simulation. The processors' shared memory can be used for static data structures such as the compiled circuit model. Shared memory can also be used for dynamic, global data structures; for example, to communicate global fault status changes. Although the memory efficiency with multicore is circuit dependent, a good empirical estimate for the total memory required is **M** + *0.3**n***M**.

Performance Results

Figure 1 shows the ATPG runtime improvement when using TetraMAX multicore processing on twelve customer designs. For quad-core CPUs, the typical runtime improvement is 3X, with a memory increase of 2X. For 8-core CPUs, the typical runtime improvement is 6X, with a memory increase of 3X. The memory consumed by multicore processing is less than half of the total memory required compared to an equivalent distributed processing run.

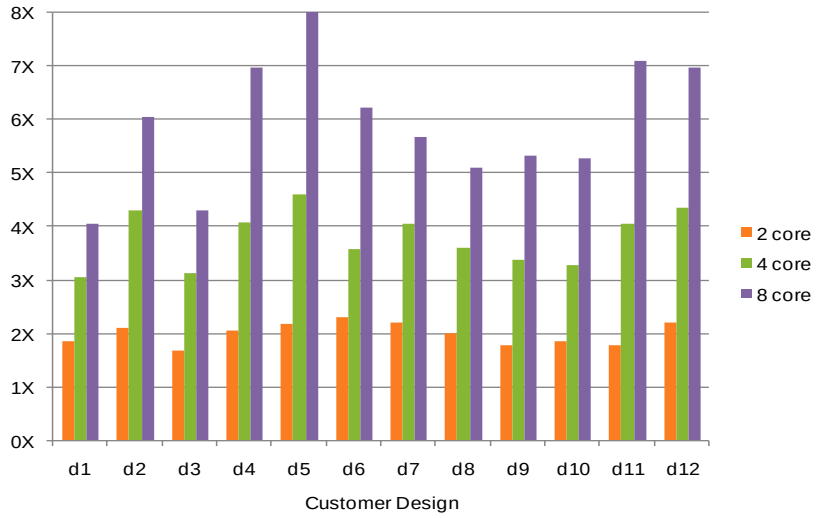

Figure 1: Runtime improvement

Distributed Processing

TetraMAX distributed processing supports a network of heterogeneous workstations and servers, as shown in Figure 2. In contrast to multicore processing, distributed processing assumes complete independence of each CPU, connected only by an Ethernet network operating at 100Mb/s.

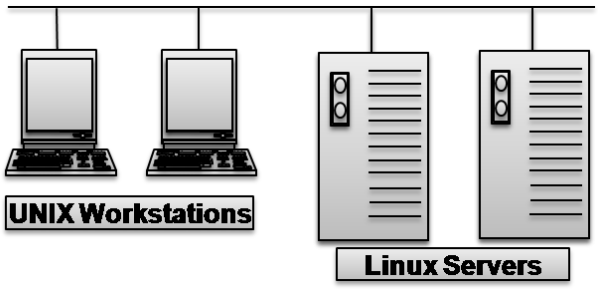

Figure 2: Distributed processing architecture

This distributed processing approach also reduces ATPG runtime and scales well to ten or more CPUs, but it has two additional considerations. First, the memory image for each slave process must be fully duplicated and separately managed on each CPU. The total memory required is approximately **M** + ***n*M**, where **M** is the single process memory and ***n*** is the number of slave processes. For a 10X reduction in ATPG runtime, the total amount of memory consumed will increase by more than 10X. Figure 3 shows a comparison of memory usage between distributed and multicore processing.

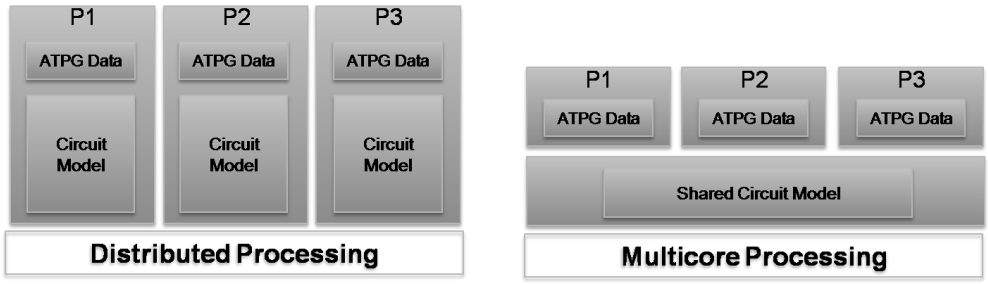

Figure 3: Memory usage comparison

Second, there are both bandwidth limitations and latency issues associated with inter-slave communications using network sockets. TetraMAX employs several proprietary strategies to maintain economical communications overhead. These avoid processing redundancies which would lead to pattern inflation, while minimizing queuing bottlenecks which can limit the scalability to only a few CPUs.

Summary

Multicore processing support in TetraMAX delivers highly optimized ATPG runtime and memory consumption on today's most common compute platforms, and is especially well-suited for applications requiring a 2X to 8X reduction in ATPG runtime. For applications requiring a 10X or more runtime reduction, TetraMAX also supports distributed processing.

SYNOPSYS®
Predictable Success
Synopsys, Inc. · 700 · East Middlefield Road · Mountain View, CA 94043 · www.synopsys.com