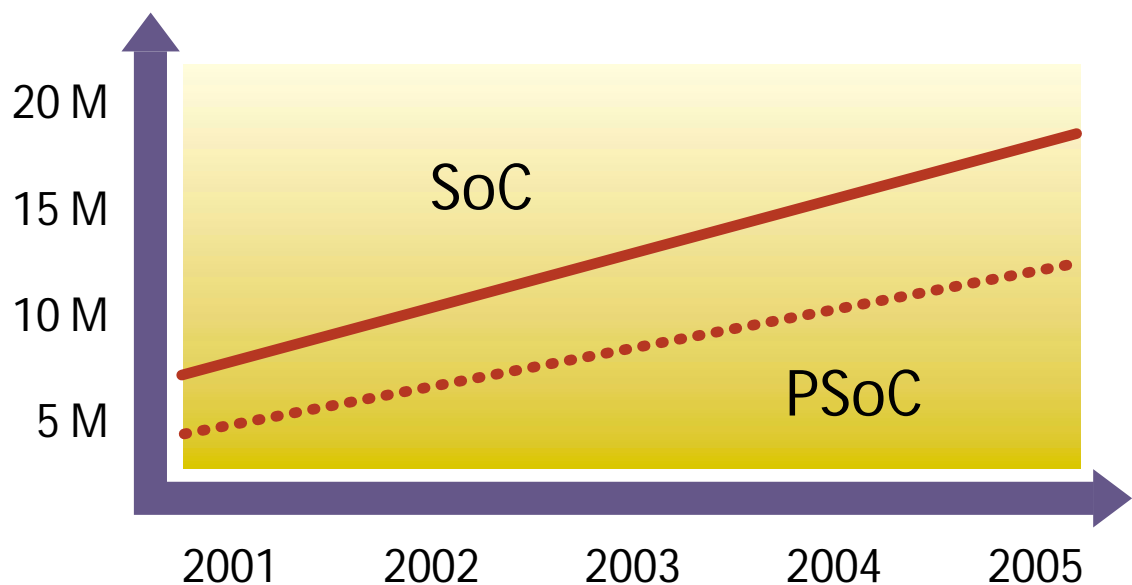


# MultiPoint™ Synthesis for ASIC and FPGA design



Moore's Law, amazingly, continues to hold. Despite predictions of unfathomable hurdles at each juncture in technological evolution, ASIC and FPGA densities continue to double approximately every 18 months as predicted by Intel's Gordon Moore years ago. In fact, by witnessing the continuation of a longstanding trend of integrating more and more functionality on a single IC, some believe that the pace of this progression may be accelerating. Systems on chip (SoCs) containing logic, memory, processors, mixed-signal elements are common today, and embedded FPGA and analog elements are on the horizon. Moreover, with underlying silicon technology on par, the SoC designer today has many choices across the spectrum of ASIC and programmable logic solutions, meaning that the issues driven by SoC complexity are being seen in both ASIC and programmable logic design flows.

As those of us in the design community have learned, the meteoric rise in silicon potential is a mixed blessing. On one hand, we have silicon technology that delivers functionality and performance capabilities to meet the demands of even the most competitive applications. On the other hand, we have frustration with design tools that limit our ability to realize the full potential of this silicon technology. Sheer design size and device complexity make the design task now the limiting factor in the types of ICs that can be realized today. This so-called "productivity gap" (Figure 1) continues to widen despite the near-myopic focus upon it by the electronic design automation (EDA) community for almost 10 years now.



**Figure 1:** Design complexity continues to double approximately every 18 months despite the fact that conventional design flows are unable to keep pace.

## The Synthesis Challenge

Nowhere is the productivity gap more evident than in the area of synthesis. Processing capacity limitations of conventional synthesis approaches demand that designs be partitioned into sub-blocks of no more than 200K gates. With current silicon capacity of up to 20 million gates, designers conceivably would need to manage 100 or more different sub-blocks in order to synthesize a design. In addition, partitioning based on gate count is non-intuitive for most designers who are more comfortable with partitioning based on functionality or timing. Another failing of current synthesis flows is that they do not separately optimize discrete functions well. This is exemplified by the demand for separate datapath synthesis today. Embedded FPGA will require both specialized synthesis mappers as well as optimization in context of the full chip.

Given the staggering increase in design complexity since synthesis technology came of age 10 years ago, it's really no surprise that the efficacy of conventional solutions is waning. Design complexity creates a number of other problems for synthesis technology. Memory utilization becomes a real challenge when a sophisticated synthesis application must actively operate on a large volume of design data. Runtime gets protracted not only by the sheer size of a design, but also by design management overhead required to handle the computational task. IP integration presents challenges as highly complex and sometimes highly constrained functions must be factored into the synthesis process. Design cycles increase dramatically on complex designs, with iterations to reconcile timing issues or interblock dependencies lengthening design time.

Another significant challenge for synthesis solutions when applied to complex designs is that of maintaining stability, a term that refers in synthesis to the extent to which incremental changes to a design have ripple effects. Incremental synthesis, whereby only portions of a design are refined, can create complications when refined sub-blocks are incorporated back into the whole. These complications occur at an increasing rate as designs, and the underlying silicon technology, become more complex. Common incremental approaches that fail to adequately account for the behavior of elements external to the sub-block being modified, or the interaction between the sub-block and the remainder of the design, can create chaos late in the design cycle. Incremental synthesis can also make formal verification difficult if changes impact multiple parts of the design.

## Alternative Approaches and Technologies

In order to avoid the shortcomings of synthesis tools, designers are deploying several alternative strategies. One is complex work-arounds such as sophisticated scripting. These elaborate and clever work-around solutions can achieve the end, but often unduly compromise quality of results (QoR) and productivity, as well as demand specialized design skills. Extensive manual scripting is often required to successfully synthesize a large ASIC design using conventional approaches, but the manual nature of such scripting makes it error prone, and demands designers be highly skilled in scripting techniques. This effort also does little to "add value" to a design, consuming valuable manpower and time that could otherwise be employed to improve design results. Undermining these scripting efforts is that the gate count of the scripted partitions must fit within the memory capacity of the synthesis tool.

Yet another strategy is to focus strictly on RTL design and hand-off RTL to an ASIC vendor for synthesis and implementation. Thus, the designer avoids the tedium associated with conventional synthesis by delegating it to the silicon vendor. This "RTL sign-off" approach is highly dependent upon intelligent, high-quality RTL and the willingness of vendors to shoulder the responsibility for design results. Designers also may be concerned about passing off a design to someone who is less knowledgeable about both design and system issues, and is less vested in the success of the design.

These strategies, by their very nature, assume that synthesis can't be improved or made less of a bottleneck. However, synthesis is an integral and essential aspect of all advanced IC design flows. To allow this technology to continue to create difficulties in implementation compromises the productivity and quality of the design effort, regardless of who performs synthesis. Thus, the synthesis problem is one that must be addressed.

## Top-Down vs. Bottom-Up

There have traditionally been two ways to attack the synthesis problem, bottom up and top-down. Each affords some benefits and has some drawbacks. The bottom-up approach refers to partitioning a design into blocks that are within the capacity of the synthesis tools. By breaking a design up into pieces, each element can be processed independently and as required by changes to that part of the design. This allows for partial recompilation and multiprocessing that will speed design compilation. A bottom-up flow also makes it possible to isolate parts of a design for incremental improvement independent of the rest of the design. If employed correctly, this can improve stability of results.

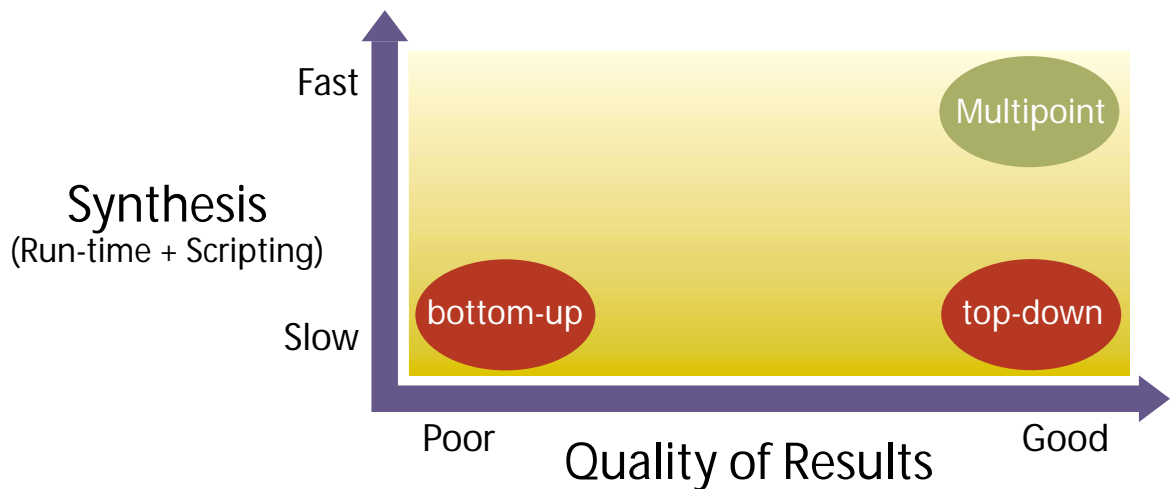
The bottom-up approach offers the best runtime on a per block basis, though overall runtime may suffer because of the project management and manual scripting overhead. The scripting effort alone can be significant, and is prone to error. There is also a QoR penalty with this approach. With a bottom-up approach, the synthesis tool can only "see" optimization opportunities within the partition, not across partition boundaries. This can eliminate the possibility for significant overall design improvements that span multiple partitions. As the number of boundaries increases, a design is further away from the best possible QoR. With two blocks you miss some optimization opportunities — with 100 blocks you miss many, many more.

The top-down approach to synthesis takes the entire system-level RTL and constraints, and allows the synthesis tool to optimize and bring the design to the gate level in one operation, without operating across partitions. When compared to a bottom-up design, this approach produces the best QoR because the synthesis tool is operating on the entire design. It is also easier to implement because the need for manual scripting is eliminated, as well as the need to manage the various partitions. However, memory and runtime requirements are prohibitive for large designs. Design iterations require that the entire design be re-synthesized for even minor changes, and are thus almost impractical except for very small designs. Likewise, replicated blocks run top-down will be individually synthesized, causing a longer runtime than the bottom-up approach of synthesizing the replicated block once, then "copying" it multiple times as the final top-level integration takes place. Finally, the place and route process can be greatly streamlined and iterations to meet timing closure avoided if a design is synthesized hierarchi-

cally with the same blocks that will be hierarchically placed-and-routed. The top-down approach is ideal in terms of delay QoR, but to accommodate other design needs or design sizes may not be the best for every design.

## MultiPoint Synthesis — Synthesis Technology for Large Designs

Neither a purely top-down nor bottom-up synthesis approach is the right answer for many designs. Instead, what is needed is an approach that combines the ease of use and QoR of top-down with the productivity and lower memory requirements of bottom-up, but which still could be used strictly top-down or bottom-up for the designs that want to and can be implemented those ways. This is particularly important in the case of large designs, where existing top-down or bottom-up only flows prevent optimal QoR and runtime. To address this and the other challenges facing designers of multi-million gate ASIC SoCs and programmable systems on chip (PSoC), Synplify has developed the MultiPoint synthesis technology (**Figure 2**).



**Figure 2:** With today's large, complex ICs, the only way to ensure both productivity and QoR is to use a different approach, combining the QoR of top-down synthesis with the stability and productivity of bottom-up synthesis.

The MultiPoint synthesis technology employs a hierarchical methodology for large designs. By contrast to traditional bottom-up hierarchical design, the MultiPoint technology automates the process of partitioning and optimizing a design. This is accomplished by performing language compilation for the entire design, which creates an intermediate format that holds all of the hierarchy information as well as key information on the design. The Synplify ASIC® solution uses the built-in HDL Analyst® tool to allow the user to visualize the RTL design hierarchy, or the user can use Tcl scripting to examine the design hierarchy. From reviewing the logical hierarchy, the designer will select the hierarchical elements which should be synthesized separately from the rest of the design — these points in the hierarchy are referred to as compile points.

Once the compile points have been selected timing constraints are then applied for each one; initially these constraints would be applied manually, but will be automated in the future. Following that step, synthesis begins at the compile point at the lowest level of the design hierarchy. As each compile point is synthesized an Interface Logic

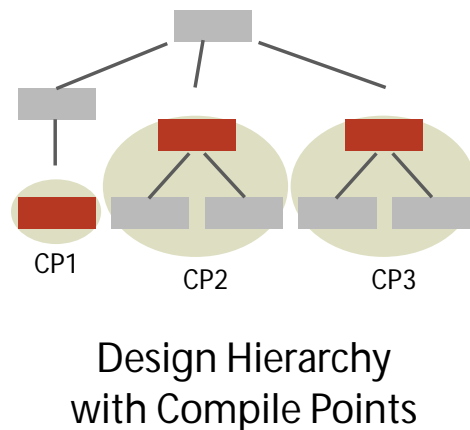
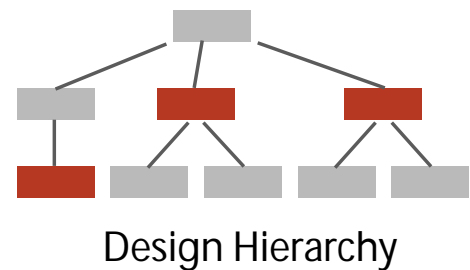
Model (ILM) is automatically created for it, and propagated forward into the next higher level of the design hierarchy. Ultimately, the top level is synthesized using the ILMs of the lower-level compilations along with any additional logic at the top level. This eliminates the need for time-consuming, error-prone scripting and enables the design team to focus on adding value to the design.

ILMs are a key technology used in MultiPoint synthesis. ILMs are partial netlists that can be written out from Synplicity® or third-party tools for any netlist or synthesized design. They are models that contain all boundary information or logic for timing analysis, therefore delivering QoR comparable to that of a top-down flow. Because ILMs only contain logic from the ports to the registers, with all other logic treated as a “black box,” they significantly reduce memory requirements and runtime for synthesis of large designs. An ILM for a large module is typically 70-80 percent smaller than its associated full design netlist.

User-defined compile points (**Table 1**) serve as the basis for creation of ILMs, and are another key element of the MultiPoint technology. Compile points are modules that are synthesized independently, and then the results of that synthesis are used to synthesize its containing block or the top-level design. One key distinction of the MultiPoint technology over other synthesis approaches is the ability to control the level of boundary optimization through three different classifications of compile points: soft, hard and locked. Soft compile points undergo full boundary optimization, meaning that port identification can change during synthesis. By contrast, port integrity is maintained for hard compile points, though boundary optimization within the containing and adjacent modules does occur. For locked compile points, logic within the module remains unchanged during optimization, though an ILM that serves as a timing model is created and may be optimized. Thus user-defined compile points serve as instructions to the synthesis tool for modeling and synthesizing a particular portion of a design. By allowing cross-boundary optimization, these compile points are key enablers for the superior QoR of a top-down flow.

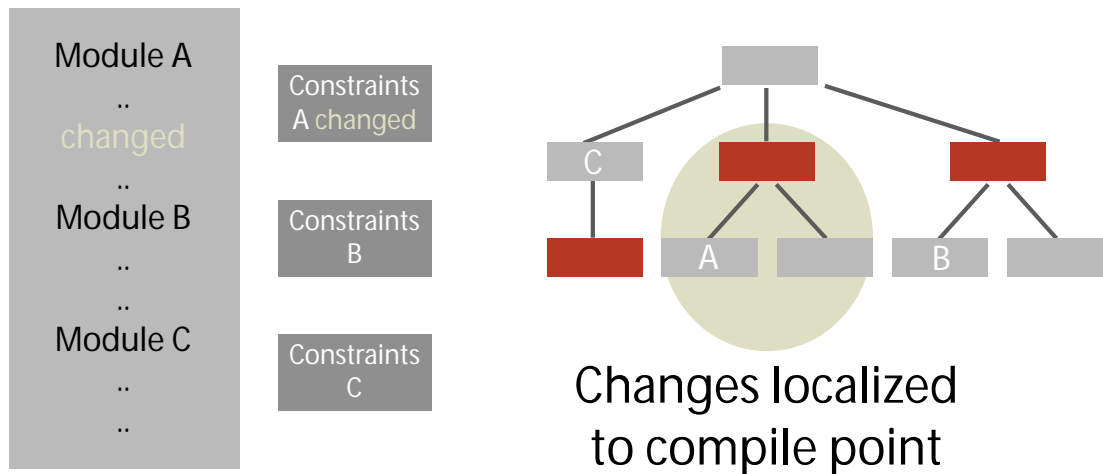
It is through user-defined compile points and ILMs that the MultiPoint technology supports IP integration. The MultiPoint technology automatically models IP and uses the timing information for synthesis. IP that is instantiated into a design can have logic optimized both within the IP module and in adjacent modules without impacting port assignments for the core itself. With multiple instances of the same IP module, boundary optimization requirements for each unique instance can be specified without requiring that each be re-synthesized.

The MultiPoint technology employs a unique difference-based incremental synthesis approach (**Figure 3 on next page**) in which only compile



**Table 1:** Another key element of MultiPoint technology are user-defined compile points that serve as the basis for creation of ILMs.

point modules that truly change are re-synthesized. Only compile points that are impacted by changes in the RTL, properties or constraints are re-synthesized. Based upon a comparison of new RTL and old, the MultiPoint technology intelligently determines which changes are substantive and therefore worthy of re-synthesis. For example, changes in the RTL time stamp, addition of comments to the RTL, and reordering of constraints will not trigger re-synthesis. This results in much more efficient synthesis, and dramatically improves productivity. Design stability is ensured via locked compile points, enabling the user to isolate changes to the module(s) of interest.



**Figure 3:** Incremental synthesis based upon comparison of actual RTL code, constraints and properties minimizes design changes and runtimes.

The MultiPoint technology is also unique in that it is equally applicable to FPGA and ASIC implementations. Indeed, with comparable gate counts, hard embedded IP, critical dimensions, performance and now cost of implementation, the distinction between the two options is becoming blurred. With the current production level of approximately 100,000 units required to break even on an ASIC design, more and more designers are witnessing the erosion of the traditional cost differential between the two options. Design technology for both ASIC and FPGA is also converging. As exemplified by Synplicity's product offerings, physical synthesis, timing-driven layout, and formal verification are no longer only at the disposal of ASIC developers. Increasingly, the designer will implement designs in either ASIC or FPGA, and even a combination of the two with embedded FPGA and embedded cores in SoCs and PSoCs, respectively. Design methodologies that employ cross-implementation EDA technology such as the MultiPoint technology provide the flexibility to implement a design in the best possible medium and are thus becoming increasingly important.

## MultiPoint Synthesis Provides a Fast Incremental Design Flow for FPGAs

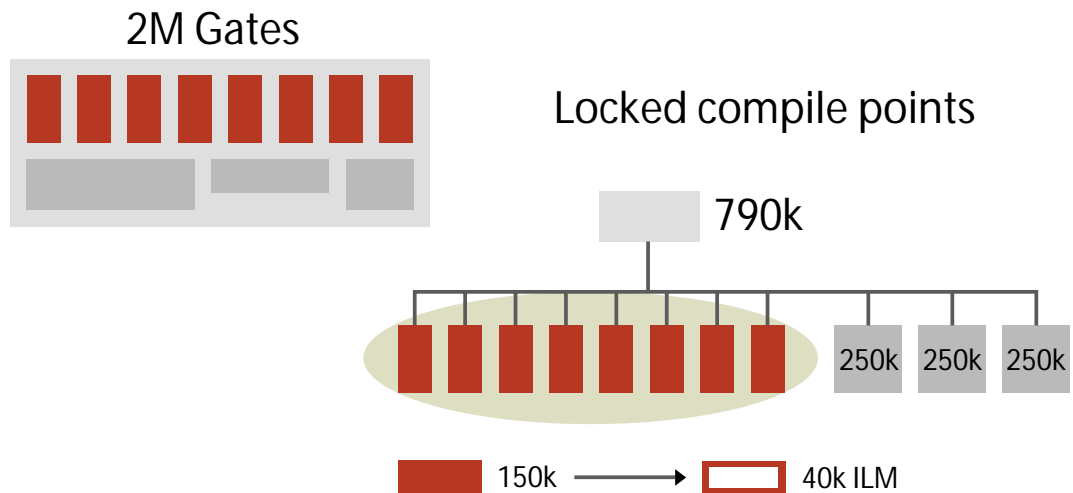
As discussed earlier the MultiPoint technology may be applied to both ASIC and FPGA designs however the current critical needs of ASIC and FPGA designers are slightly different. While ASIC designers need to efficiently manage extremely high gate count, design partitioning and the associated scripts for synthesizing and reconstructing the

design, FPGA designers generally have more need for a fast incremental design flow that does not compromise on quality of results. The MultiPoint flow when used with Altera's Logic Lock or Xilinx Modular Design Flow provides the designer with a superior incremental design methodology that allows portions of the design to be locked down and remain unchanged during synthesis. This keeps portions of the design that have already been verified to remain unchanged and stable while only the truly necessary portions of the design are modified to reflect late design changes. This approach not only provides stability of results, but also dramatically decreases both synthesis and P&R runtime per design change.

## Using MultiPoint Synthesis

The MultiPoint synthesis flow is straightforward. First, the designer compiles HDL and creates the RTL view of the entire design. Next, the designer defines compile points based upon their knowledge of the design and critical functions or paths. This is followed by automatic time budgeting performed on the entire design at once. Once these initial time budgets are established, each compile point is then synthesized using those budgets. ILMs are then created automatically, and the top-level timing analysis and optimization performed. The MultiPoint technology performs initial time budgeting by operating simultaneously on the entire design to create timing budgets for modules within the hierarchy. Thus, the need for time consuming manual constraint creation is eliminated, and time budgeting is carried out much faster than in conventional approaches.

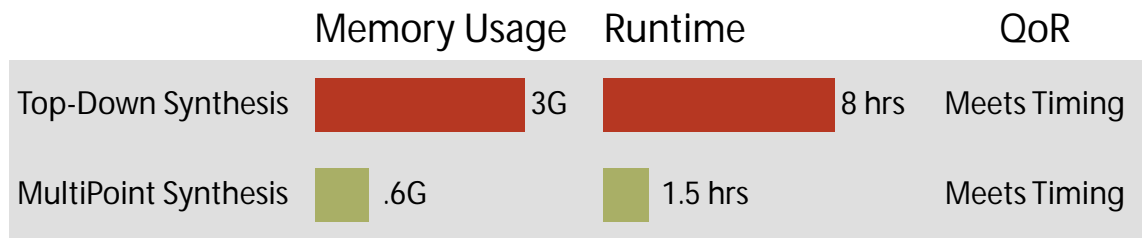
Results achievable with the MultiPoint technology are shown in **Figure 4a** on a 2M gate design implemented in 0.11 micron technology. Not unlike most advanced SoCs, this



**Figure 4a:** A 2M gate .11um design with a replicated 150K gate IP block moves its way through the MultiPoint flow, decreasing the computational overhead by making the replicated block a compile point represented by low-overhead ILMs.

particular design contains a replicated IP block. The 150K gate block is replicated nine times, and is specified as a locked compile point. The replicated block is mapped only once and then replicated at the top level. After the lower-level compile points are synthesized, top-down synthesis is performed using a single ILM of the replicated block and another of the remaining logic, reducing the computational overhead by almost 80

percent. When compared to a traditional top-down approach (**Figure 4b on next page**), memory usage and runtime are reduced by about 80 percent, and QoR is equivalent to straight top-down synthesis.



**Figure 4b:** When compared to a traditional top-down synthesis approach, the MultiPoint technology offers significant memory and runtime efficiency and produces equivalent QoR.

The advantages of the MultiPoint technology over traditional synthesis are many. Unlike conventional approaches limited to a maximum synthesizable block size of 200K gates, the MultiPoint technology has the capacity to synthesize blocks of 1.5 -2M gates at a time. This means designers do not have to partition their designs based on tool-imposed memory limits, but more intuitively based on functionality or timing of the design. Through its unique difference-based incremental approach and automation of ILM creation and time budgeting, the MultiPoint technology affords a very high productivity automated solution. Its top-down, hierarchical approach provides QoR equivalent to top-down synthesis, and incremental synthesis with locked compile points ensures bottom-up stability. A MultiPoint flow also streamlines vendor hand-off, as ILMs form the exchange media for IP and other large blocks, and the comparatively small project file of compile points is much easier to transfer than a large file of scripts.

The versatility to apply the MultiPoint technology to FPGA or ASIC implementations eliminates the need to master separate tool suites, and enables developers to select the best possible implementation choice, whether ASIC, FPGA or a hybrid of the two, for each project. Likewise, the MultiPoint technology's scalability eliminates the need to change methodology for each new generation of process technology. Finally, is the fact that the MultiPoint technology comes from a recognized leader in synthesis solutions for complex ICs provides the assurance that the technology is viable and will be supported over the long term.

## Future Design Needs Addressed By MultiPoint Technology

The MultiPoint technology is an extendable and scalable platform for future design needs, as well. Many designers rightfully are concerned that the synthesis methodologies currently used are not extendible into the next generation of designs. With this in mind, Synplicity developed the MultiPoint technology to adapt to future design requirements without having to change-out the synthesis methodology used. The basic architecture of MultiPoint synthesis allows integration with new capabilities as they become design requirements, and to support future hardware, operating system, and memory structures. MultiPoint synthesis is also amenable to multiprocessing computer architectures, in that each compile point could be synthesized on separate processors, then the whole netlist can be integrated together.

## Summary

Economics and market forces compel us to deploy the best that silicon technology has to offer. With the MultiPoint synthesis technology, Synplicity offers designers an advanced, high capacity, highly automated design flow for both FPGA and ASIC designs. Capabilities such as difference-based incremental synthesis, automatic ILM creation and use, and user-definable compile points make this flow more attractive for designing large ASICs and FPGAs than any other approach. The MultiPoint technology embodies a unique mix of traditional top-down and bottom-up synthesis approaches and advanced synthesis techniques that enable high QoR and productivity when developing today's highly complex SoCs and PSoCs.