THE BENEFITS OF MODEL-BASED ENGINEERING IN PRODUCT DEVELOPMENT – FROM PCB TO SYSTEMS

MENTOR GRAPHICS

WHITEPAPER

Mentor®
A Siemens Business

PCB DESIGN

www.mentor.com

Simulation models are often used to help develop today's system designs. Models represent the system, or its parts, and show interpretations of requirements, allow exploration of design alternatives, and permit evaluation of potential solutions. Experienced design teams use models to learn more about system characteristics and capabilities, and to find solutions for the application.

Unfortunately, in many design flows, the typical system model is seldom fully leveraged throughout the development process. It often stays on the system architect's desk, not necessarily wasted, but also not fully utilized for maximum productivity. Model-based engineering – sometimes called model-driven engineering or model-driven systems engineering – improves development process productivity by using models throughout the design flow.

WHAT IS A SYSTEM?

In this paper "system" is used as a generic term to represent a design. It could represent a complex integrated circuit, an electronic circuit built on a printed circuit board, or multiple devices (think sensors, actuators, electronics, and software) connected together to do something useful. Whichever type of system you work with, the principles of model-based engineering can improve your development process.

INCOSE* (International Council on Systems Engineering) describes systems engineering as "an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem." Model-based engineering implements the goals of systems engineering and delivers the following benefits:

- Proof that design specifications are practical and realizable
- Closer association of products with customer requirements and design specifications
- Tradeoff of values, costs, etc. for more cost-effective designs
- Shortened product development cycles
- Products that better solve problems and meet customer needs

A model-based engineering process enables informed design decisions, ensuring design teams understand what is being developed (i.e. the function), where to make partitioning choices (i.e. the architecture), and how the system will be built (i.e. implementation). It supports verification of the system at each stage of design, reducing reliance on late-stage physical integration and hardware testing to validate the system.

SEQUENTIAL DEVELOPMENT VS MODEL-BASED ENGINEERING

A typical system design flow (see Figure 1) begins with system definition, which drives the architecture and implementation phases of the system's development. The problem with this flow is its sequential nature. While there may be iterations within and between the phases, in general the flow is from left to right. With this sequential flow there are numerous opportunities for issues to creep into the design that aren't discovered until very late in the process as the system comes together.

Integration of components and systems on the right side of the "V" is too expensive, inflexible, and slow to accommodate the increased design complexity and mixed-signal sophistication that characterize today's systems. Hardware prototypes are often used to evaluate systems by assembling microcontrollers, components, and mechatronic subsystems in a laboratory environment. This method creates serious configuration challenges because of the large number of devices that must be connected, configured, and tested across a wide range of environmental conditions.

While the sequential development process may be a guide for developing simple systems, the more complex the system, the less likely this approach will result in a working design, developed on schedule and within budget. The process must be improved with more concurrency, earlier validation and verification, and more efficient system integration and process collaboration.

Model-based engineering enables a concurrent and collaborative design process where users examine and define requirements, propose solution architectures, demonstrate and exchange ideas with stakeholders, and consider feature tradeoffs and costs. Using a model-based engineering process, teams work together to create development-oriented requirements that are concrete, actionable, executable, traceable, and implementable – all of which leads to a successful product design.

WHAT IS XPEDITION AMS?

Xpedition® Enterprise includes a standards-based analog and mixed-signal (AMS) modeling and simulation environment. It supports standard SPICE models, and the power and flexibility of the IEEE standard VHDL-AMS modeling language. With Xpedition, you can analyze your designs in the time and frequency domains, and run advanced analyses such as sensitivity, stress, statistical, scenario-based worst case, fault-insertion, and design experiments.

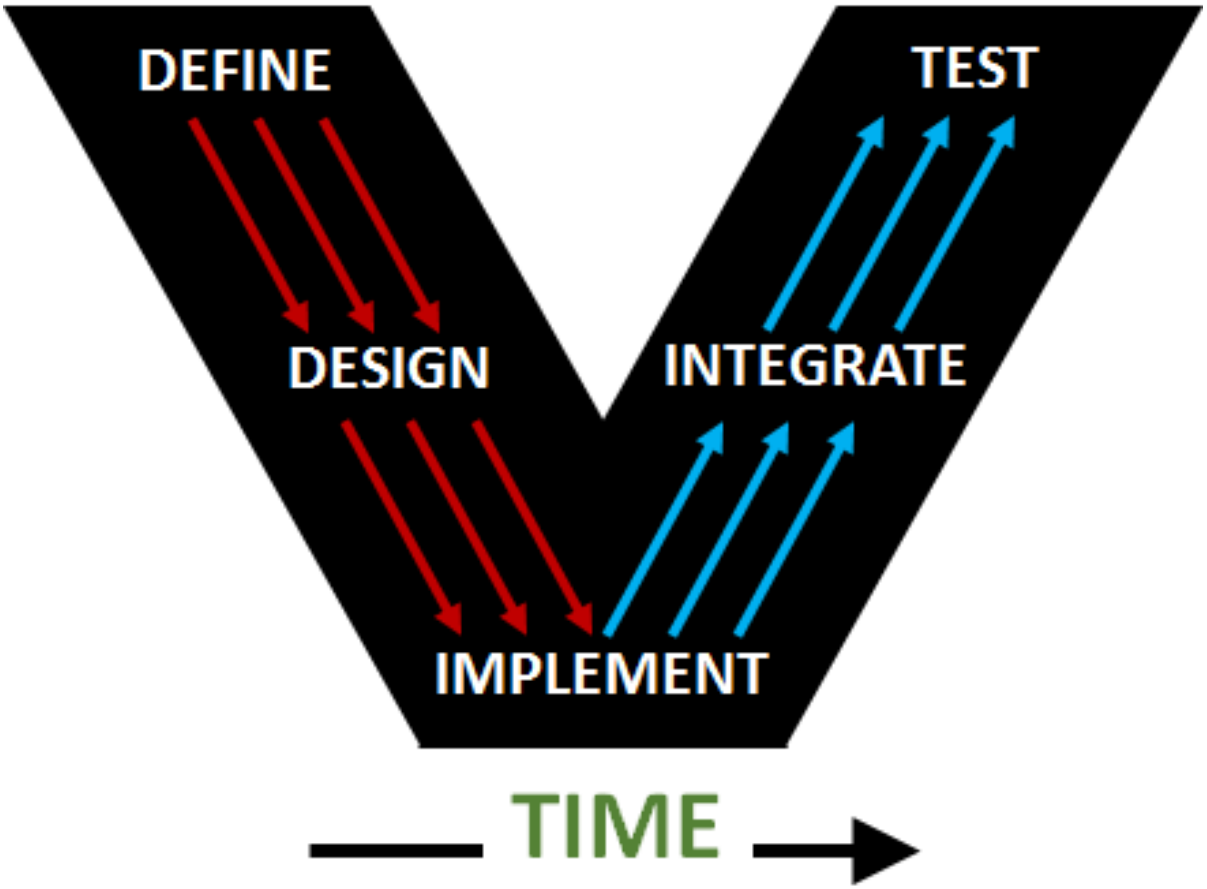

Figure 1: The V-diagram is a common example of a sequential design process.

THE MODEL-BASED ENGINEERING PROCESS

Advances in technology create opportunities and possibilities for design that are often beyond the expertise of a single design team. In general, no design team is its own island, and no system (except, perhaps, for the very simple) is designed by one team. There are too many functions, capabilities, and options to consider for a system of even moderate complexity to make single-team design practical.

Model-based engineering is a top-down development process that depends on accurate modeling and simulation at multiple levels of design abstraction. It links each level in a design flow together so that one level can be verified against another, ultimately linking the final system back to requirements to ensure the design implementation meets specification.

The following discussion focuses on three stages of design: functional design, architectural design, and implementation design. Consider this a minimum set for a reasonably complex system. Your flow may have more stages, but the concepts described here apply no matter how many you use.

As you review this discussion, keep in mind that a model-based engineering process is not a one-way trip through the different design stages. Links between design stages are a key advantage of model-based engineering, giving teams the much-needed ability to seamlessly switch back-and-forth between stages. This flexibility ultimately connects a design's implementation – the manufactured product – with the requirements and specifications. Design teams can select the design stage most suited to the level of analysis they need and, with a modeling and simulation environment such as Xpedition AMS, mix-and-match stages.

THE RIGHT ENVIRONMENT

Model-based engineering relies on two important elements: modeling and simulation. To get the most out of a model-based engineering process, it's important to select a modeling and simulation environment suitable for the complexity of your system. Although you might get by with a basic SPICE-level environment, greater system complexity requires more modeling and simulation capability. If a basic SPICE-level environment represents one end of the modeling and simulation spectrum, the opposite end is represented by an advanced simulator supporting a full-featured, general-purpose, standards-based modeling language. A language-based modeling and simulation environment, such as Xpedition AMS, is well suited to the model-based engineering process.

WHAT IS A VIRTUAL PROTOTYPE?

Before jumping into our detailed discussion, let's review an important concept: the virtual prototype. "Virtual prototype" and "virtual prototyping" are often used by simulation tool companies as buzz-phrases, but what do they really mean?

Consider a common design flow that drives towards a hardware prototype and bench testing. This flow focuses on producing a testable prototype using a sequential process, meaning the design-prototype-test flow is repeated, with test results often triggering design changes that restart the flow, until test results agree with requirements. This flow is expensive in both time and resources (engineering time, manufacturing materials, etc.). Even a small difference between measurement and requirement can trigger a complete prototype re-spin. Modeling and simulation are used sparingly, if at all, only for the design questions that do not have easy answers.

Enter the virtual prototype. With the right models and simulator, teams can develop and test their designs all in the virtual world of simulation using a model – or virtual prototype – of the real system. Why is this important? Modeling and simulation typically enable faster and more thorough evaluation of a design, including analyzing sensitivities, stresses, worst-case performance, and statistical behavior. The result is a careful vetting of design performance before building any hardware. Does this mean modeling and simulation – the virtual prototype – replaces hardware

prototyping and bench testing all together? No. But building and testing a virtual prototype, where construction and testing is more efficient and less expensive, and then verifying the system with just one or two hardware prototypes, saves significant time and resources.

FUNCTIONAL DESIGN

A model-based engineering process starts with the Functional Design stage. At this stage, the system architect(s) review the specification and turn it into a model representing the nominal functions of the system. Inputs are defined and processed by the system models to produce the desired output. Models at this stage are detailed enough to accurately represent the system's nominal function but no more; adding too much detail slows simulation performance with little added benefit. This high-level system model is sometimes called an executable specification; it gives the system architect(s) a chance to ask and answer important questions, such as "Is this specification accurate?" and "Can this system, as defined, really be built?"

Once the executable specification is running and fine-tuned, development teams use it to review the overall system function and ask their questions. Making sure all teams have the same – and correct – interpretation of system function is critical to a successful design, and is a primary purpose of this functional-level model. Once all teams have a correct understanding of how the system works and what the overall requirements are, it's time to partition the design so each team has its own piece of the system puzzle to work on.

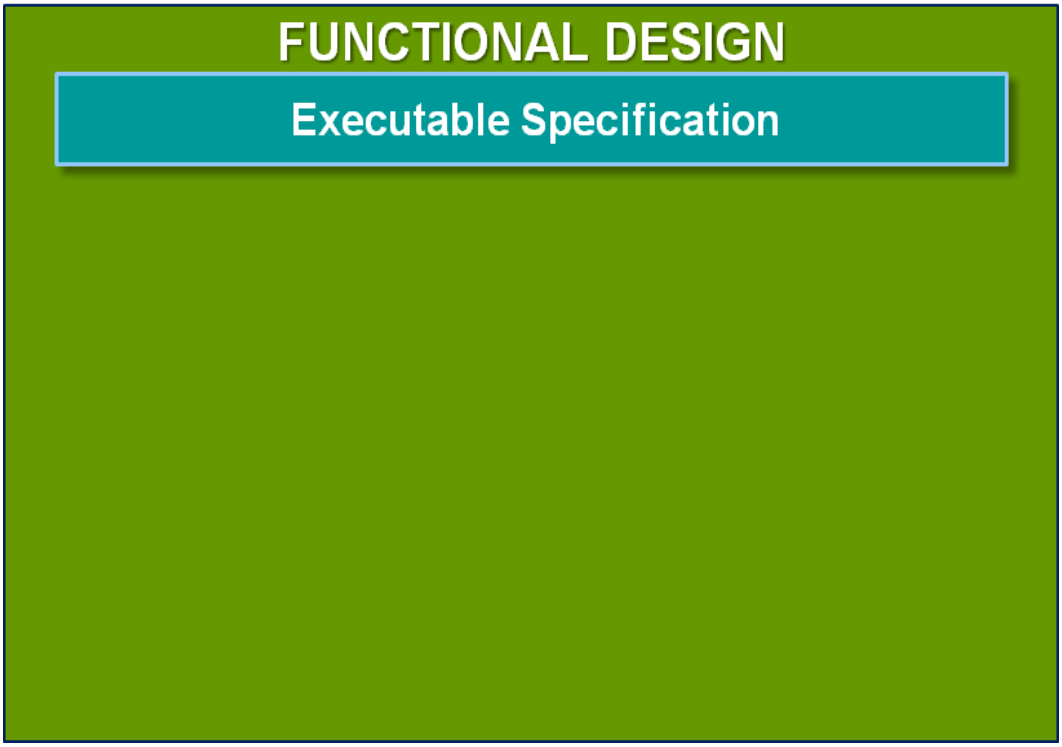

Figure 2: A model-based engineering process starts with the Functional Design Stage where design specifications are modeled as an executable specification.

ARCHITECTURAL DESIGN

The executable specification (described above) represents the high-level view of a system's nominal performance, but is typically not detailed enough for individual design teams to start their development work. To help with this, the model-based engineering process moves to the Architectural Design stage.

At this stage, the system model from the Functional Design stage – the executable specification – is partitioned into blocks with independent requirements, models, and simulation results. The system is partitioned in a way that best meets the organization's development structure. For example, if development teams are divided by technology (e.g., electronics, power, electro-mechanics, controls, embedded software), then each team's partition should match their technology expertise.

The Architectural Design stage is meant to be a system explorations phase. Design teams explore different options to find a solution that best meets the partition's requirements. When work at this stage is complete, the development team should have clear performance requirements for individual components in their design. Once component requirements are determined, component models are selected or developed, parameterized, and connected to verify design performance against the partition model. And when enough teams complete and verify their architectural partition models, these models are assembled and analyzed so the architectural design can be verified against the executable specification. Once verification is complete, the Architectural Design stage performance requirements drive work in the Implementation Design stage.

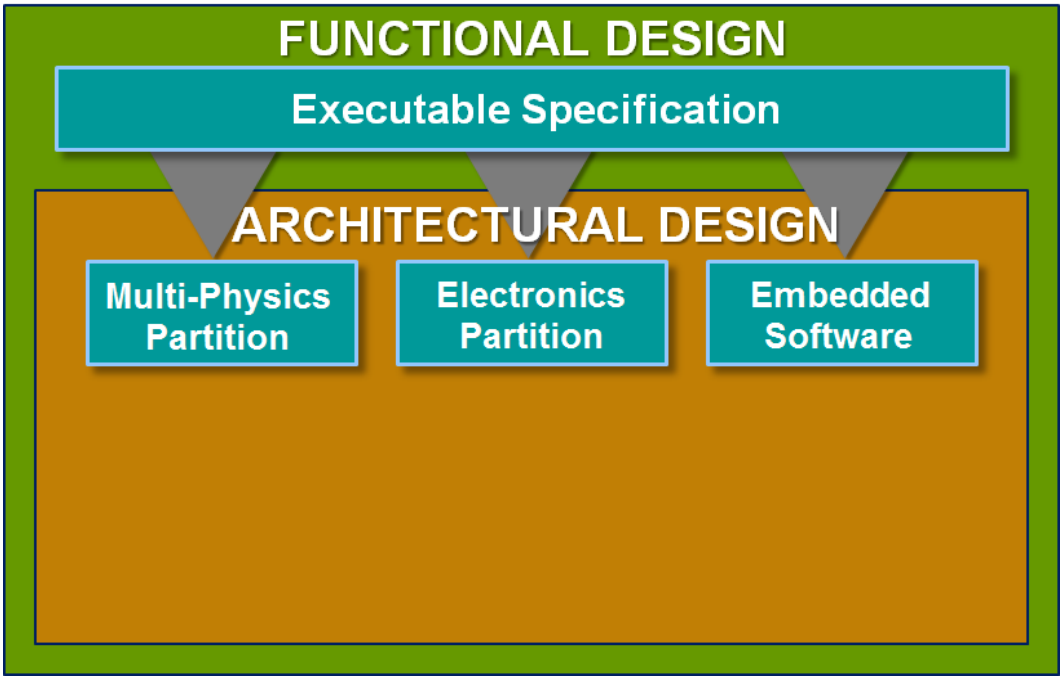

Figure 3: At the Architectural Design Stage, the executable specification is divided into partition models that design teams use to guide their design work.

IMPLEMENTATION DESIGN

At the start of the Implementation Design stage, teams have performance criteria for individual components and must find real-world components – with real part numbers – to match. The trick is finding exact matches, which often are not available. So design teams choose the closest standard part available along with its simulation model from the corporate library, a vendor library, or by creating and parameterizing a new model. These part models replace the parameterized generic models of the Architectural Design stage. Hopefully, simulation results at the Implementation Design stage will match those from the architectural stage, but there can easily be differences due to the design parameter value versus standard value dilemma. For example, the Architectural Design stage may call for a 29-microfarad capacitor, but the closest standard value is 33 microfarads.

Using models of real parts with standard values at the Implementation Design stage, design teams can run additional analyses to ensure component performance and tolerance ranges do not push system performance beyond specification limits. When the implementation design is complete, its performance is verified against the architectural and functional stage models. Once stage performance is verified against the specification, the system design is complete and ready for the next step – moving from a virtual prototype to a real prototype for further testing and verification along the right-half of the development V-diagram.

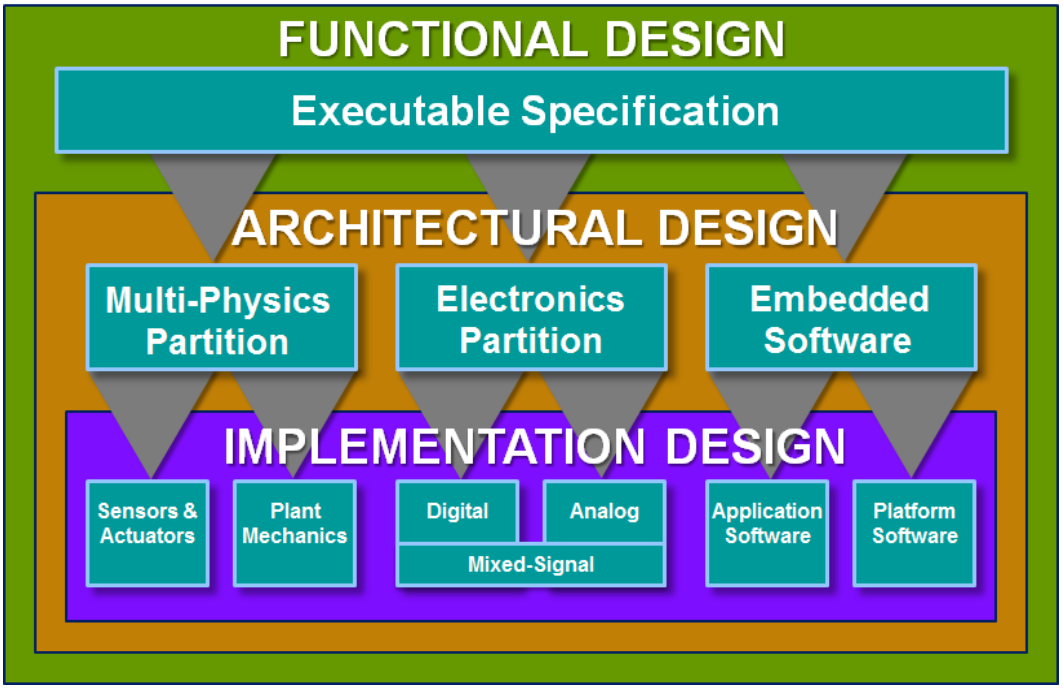

Figure 4: At the Implementation Design Stage, performance requirements for component models are translated into commercially available hardware components.

THE POWER OF MIX & MATCH

Dividing a system development process into discreet stages focused on a model-based engineering flow has power just in process organization. In other words, if development teams use the stages described above in a simple linear fashion, finishing one stage, then moving on to the next, and then the next – only backing up to verify one stage against another – there will be measurable benefits to their development programs. Beyond this, however, there is additional benefit in a modeling and simulation flow that supports the option to mix and match design abstraction levels during development – an important characteristic of a model-based engineering process.

In general, simulation performance – how fast a system simulation runs – is based on how much detail is included in the system model. Assume for this discussion that the only variable is model detail; all simulator settings and options remain the same between abstraction levels. A simulation with functional models typically runs faster than the same simulation with architectural models, which runs faster than a simulation with implementation models. But with the multi-level modeling and simulation approach of the model-based engineering process, and given the right modeling and simulation environment, it's possible to mix-and-match model abstraction levels.

What might the benefit be? Higher-level models and faster simulation times where less system detail is needed, and increased detail and focus on portions of the system as needed. For example, consider the basic audio amplifier shown in Figure 5.

This amplifier is partitioned into three blocks: pre-amplifier, controls, and power amplifier. Assume each block is designed by a separate team, and that each team has access to the entire simulation model at this partitioned level. Each team can focus on the detailed design of their own block, with the advantage of analyzing a model of their block at a detailed level within the context of the rest of the system modeled at a higher level.

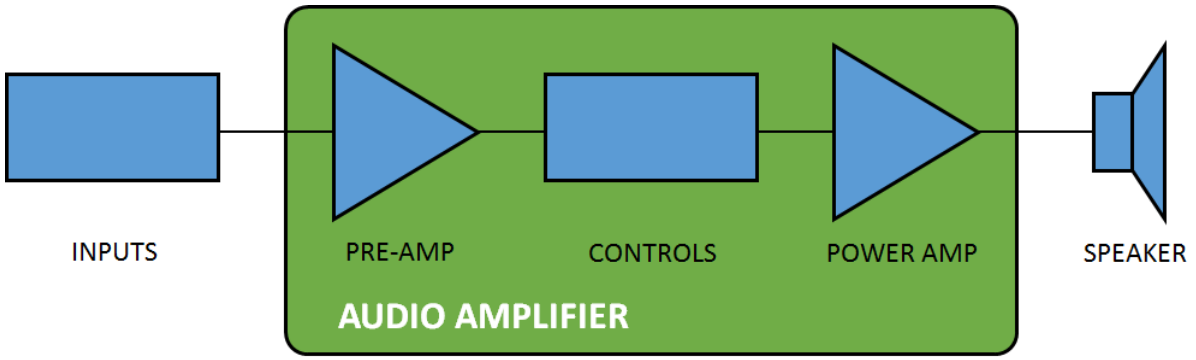

Figure 5: An audio amplifier partitioned as a model-based, multi-level design example.

SUMMARY

Model-based engineering is a practical and efficient methodology for system design. It provides a structure for managing complexity while, at each design stage, making it possible to directly link design functionality back to the system's original requirements and functional specifications. Despite its many advantages, however, it is not a cure-all for every system design issue, nor is it appropriate for every system. For example, some systems are simple enough that using model-based engineering and virtual prototypes will take more time than using the traditional design-prototype-test flow. But, as system complexity grows, a model-based engineering process becomes at first practical, then necessary, in order to ensure an efficient, cost-effective development process that leads to a successful design.

Key to implementing a model-based engineering flow is selecting the right modeling and simulation environment. Simple systems might be served by a basic SPICE-based option. More complex systems, whether implemented as a system-on-a-chip, a circuit on a board, or a mixed-technology product, will benefit from a model-based engineering flow built on advanced simulation capabilities and flexible, standards-based model development options similar to those available in the AMS environment of Xpedition Enterprise.

* http://www.incose.dk/systems-engineering/what-is-se

Corporate Headquarters
Mentor Graphics Corporation
8005 SW Boeckman Road
Wilsonville, OR 97070-7777
Phone: 503.685.7000
Fax: 503.685.1204

Sales and Product Information
Phone: 800.547.3000
sales_info@mentor.com

Silicon Valley
Mentor Graphics Corporation
46871 Bayside Parkway
Fremont, CA 94538 USA
Phone: 510.354.7400
Fax: 510.354.7467

North American Support Center
Phone: 800.547.4303

Europe
Mentor Graphics
Deutschland GmbH
Arnulfstrasse 201
80634 Munich
Germany
Phone: +49.89.57096.0
Fax: +49.89.57096.400

Pacific Rim
Mentor Graphics (Taiwan)
11F, No. 120, Section 2,
Gongdao 5th Road
HsinChu City 300,
Taiwan, ROC
Phone: 886.3.513.1000
Fax: 886.3.573.4734

Japan
Mentor Graphics Japan Co., Ltd.
Gotenyama Trust Tower
7-35, Kita-Shinagawa 4-chome
Shinagawa-Ku, Tokyo 140-0001
Japan
Phone: +81.3.5488.3033
Fax: +81.3.5488.3004

MGC 10-17 TECH-16630-w