

# Sustainable Embedded Systems with Virtual Prototyping

By Shabtay Matalon, Sagih Bzezinski, Russell Klein, and Colin Walls, Mentor Graphics

There are many reasons why designers of embedded systems, such as the ARM® Cortex™-A9 MP, are increasingly driven to stay within tight system power budgets. Although higher capacity batteries are becoming available, they are still falling short of expectations, and lower process geometry has failed to deliver spectacular power reductions. As a result, alternative techniques for power management have emerged.

When designing an embedded system today, embedded system designers must use strategies that pinpoint the best compromises between performance and power consumption and utilize every possible efficiency measure to meet the desired design goals. Two key ingredients must be present to accomplish this: a system-wide approach involving both hardware and software designers; and a set of tools and methodologies for model and platform creation, fast simulation, precise monitoring of power consumption, and quick optimization of both hardware and software.

The best way to blend these ingredients is through hardware aware virtual prototyping. Virtual prototyping techniques provide an alternative to hardware physical prototyping by using an abstracted representation of the hardware. Virtual prototyping opens the way to concurrent development of hardware and software and to continuous analysis and optimization of the design for performance and power.

## Controlling Power with Software

Historically, the power consumption of a device would have been regarded as the sole province of the hardware designer. In recent years, the influence of the software design on power consumption has grown. Some key areas, where software has an influence on power consumption is in displays, wireless peripherals, and CPU utilization.

Sophisticated displays are becoming increasingly common on a variety of embedded systems. Their size and resolution is steadily

increasing and touch sensitivity is widely implemented. This hardware is a major power drain. It is essential that the driving software monitors utilization very carefully and dims or shuts down displays when they are not in use.

The wireless connectivity of embedded devices has increased over the last decade. In almost all cases, the software may have the opportunity to turn off the wireless interface when it is not in use or optimize the transmission power for current circumstances.

The impact of the CPU code's execution performance may be quite significant – minimizing the number of instructions that need to be executed to perform a given task reduces the number of Watt-hours required. Low-power consumption is closely related to algorithm performance. Anything that can be done to reduce the number of clock cycles needed to execute an algorithm can be applied to reducing power consumption.

New system-on-chip (SoC) devices continue to expand the array of mechanisms available for a software developer to leverage in-power management. Numerous low-power states (sleep, doze, hibernate, etc.) for the CPU and on-chip peripherals have become increasingly sophisticated. Yet, coordinating the broad array of states across CPU and peripherals is a major task. Broadly speaking, developers have two facilities to control system power consumption via software: 1) sleep/suspend and 2) Dynamic Voltage and Frequency Scaling (DVFS). In systems that provide DVFS facilities, rather than entering a sleep or suspended mode, the program can reduce the clock frequency and voltage of the system to conserve power.

Where both facilities are available, there are trade-offs involved in deciding which approach to take, a suspended mode or a reduced voltage and clock frequency. If it is possible to meet the performance requirements of the system at a reduced clock frequency and a reduced voltage, DVFS will usually be preferred, as it will result

in the lowest power consumption. The alternative of going into a suspended mode saves less power, but higher response times and throughput can be achieved.

One of the challenges with either DVFS or power down modes comes as software complexity increases. When more than one process is running concurrently, all with differing performance requirements, it is no longer appropriate for any one process to suspend the processor, reduce the clock, or turn off a peripheral. A framework is needed to allow all processes to report their minimum performance needs, monitor those needs, and keep those minimum performance levels while turning down or turning off power where it is no longer needed.

Of course, the execution performance of code is radically affected by the processing power of the CPU. A powerful processor executes many millions of instructions per second, but runs with a very high clock frequency, which directly affects its power consumption. Multiple, lower power CPUs, running at lower frequencies, may be able to offer a similar execution throughput while consuming less power.

Thus, an increasing number of embedded systems are being implemented with multiple CPU cores. This may be a number of identical cores or cores with multiple architectures. In either case, new challenges are presented to the software developer.

Multicore systems offer the potential of significantly reducing power consumption for a processor-based system. It may not be obvious, but smaller and simpler processors are far more power efficient than larger more complex ones. This can be true even if the complex processors have sophisticated power management facilities. The challenge, of course, is to enable algorithms to take advantage of the various cores. But the benefit of going to multiple smaller cores, in terms of power consumption, is significant.

### Controlling Power in Hardware

Early validation of software on an early virtual prototype of the hardware allows design teams to readily change the hardware design topology and influence the RTL design specification before RTL specs are finalized and implemented. At this design stage, it's still easy to add or remove compute resources, add a hardware accelerator block for a performance critical function, and optimize the design for low power.

Hardware component selection includes choosing the processors and peripherals that will compose the design. A key decision is choosing the right processor. Some processors have been optimized for low power operation in portable battery operated devices and some are suitable for desktops and servers. The most advanced devices need to support numerous interfaces connecting them to the outside world, communicating video, audio, text

messages, and control signals in a variety of formats. Peripherals support data handling, computation, and communication through these interfaces and their large number may impact power.

Hardware topology defines how the design components are connected with each other usually using standard buses using standard protocols. The hardware topology impacts important attributes such as latency and throughput of the data flowing through the design and thus impacts directly the level of power consumed.

Power domains are sections in the design that can be controlled independently from others to conserve power. Such sections may not be required to operate all the time and may be shut down during idle times of certain functions in the design. Others may be controlled to run slower under given conditions, thus conserving on power. Power domains are usually associated with power states that define when to apply a specific power mode to the design and a power controller unit that can control the power domains during operation.

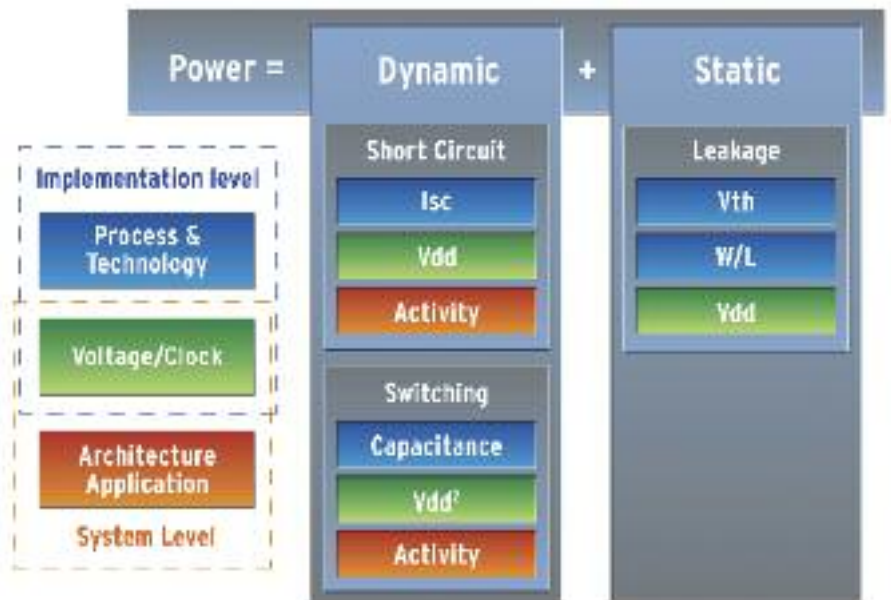


Figure 1: Factors that affect power consumption.

The power consumed by a device is composed of static and dynamic power. Both static and dynamic power can be influenced at the various design stages starting at the system level and continuing at the implementation level. System-level design entails architecture-level design and optimization of power under the control of the application software.

As static and dynamic power are additive, each must be considered independently. Static power, which is also defined as “leakage,” is consumed in the absence of any system activity. It is primarily associated with the current flowing through the transistor in idle state determined by the transistor attributes. Dynamic power is associated with the activity in the design

influenced by the volume of data that the system has to process in a given time.

The design operating frequency is usually determined by the frequency of the clocks operating in the design. The faster the clocks run, the more they increase the switching frequency of the transistors, resulting in larger dynamic power consumption. Reducing the clock frequencies to the minimal level required to meet performance will result in power savings.

The process technology has traditionally been the primary factor determining power, as moving to smaller process geometry allowed reducing Vdd voltage and reducing transistor leakage. However moving to the next process technology also resulted in increased operating frequency and increased gate count.

As simply riding the process technology curve could no longer drive the design power down and meet the design requirements, new designs combined with advanced power analysis and optimization techniques had to emerge to tackle the power problem.

### An Integrated Approach to Low Power

The best way to provide the necessary tools and methodology to achieve these goals lies in the emergence of hardware aware virtual prototyping. Virtual prototyping techniques provide an alternative to hardware prototyping by using abstracted functional models of the hardware. Virtual prototyping opens the way to concurrent development of hardware and software and continuous analysis and optimization of the design for performance and power.

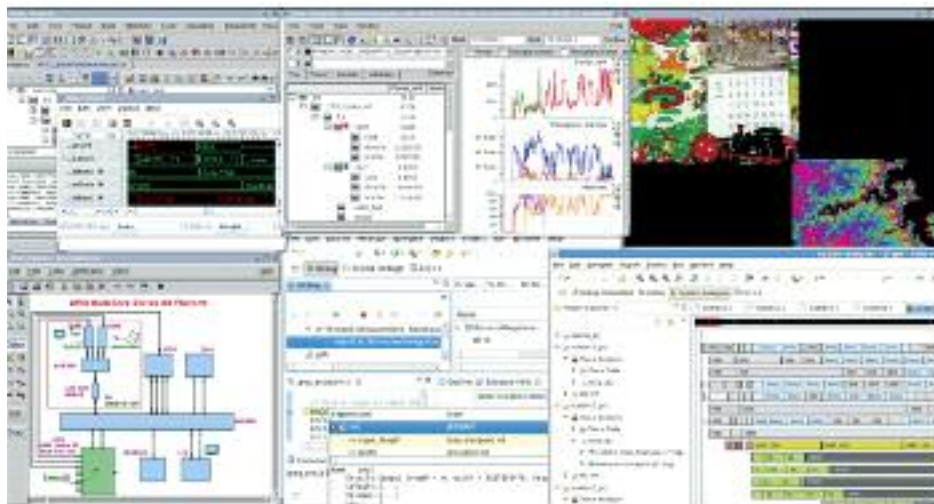


Figure 2: Virtual Prototype with Vista HW Debug and Analysis, and Sourcery CodeBench SW Analysis.

A breakthrough technology at the heart of Mentor's Vista™ virtual prototyping tool is scalable transaction-level modeling. A single, scalable SystemC TLM2.0 model handles all ESL abstraction levels and design tasks by separating communication, functionality, and the architectural aspects of timing and power into distinct but synchronized models. Scalable transaction-level models allow

timing and power details to be added, changed, or disabled as needed, while maintaining a single behavioral description throughout the design flow. Thus, designers can produce virtual prototype executable packages that can be dispatched in volume to hundreds of software engineers, enabling them to optimize power using virtual prototypes. Vista provides the capabilities to model,

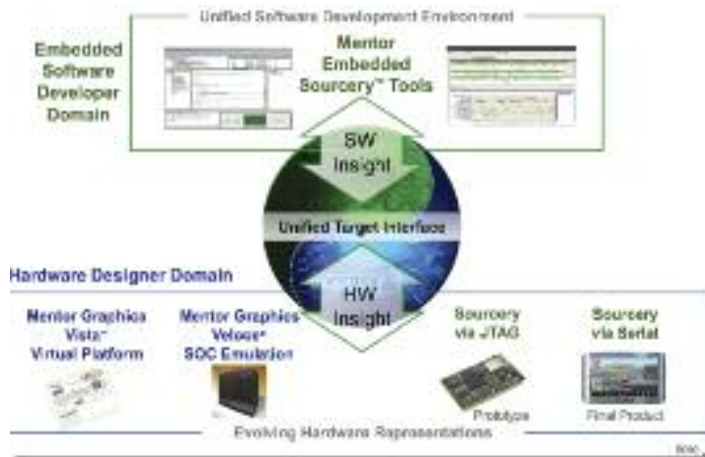


Figure 3: The Mentor Unified Software Development Environment for evolving hardware representations.

analyze, and optimize power before implementation. At this design stage, developers are able to reduce power under software control by up to 80 percent by using the sleep/suspend and the Dynamic Voltage and Frequency Scaling (DVFS) techniques. Vista provides the capabilities to apply, analyze, and optimize the impact of these techniques on the power consumed by the device.

In addition, the Sourcery™ System Analyzer tool from Mentor Graphics allows developers to understand the performance characteristics of either an application or a complete system. System Analyzer collects data from several sources (Linux, Nucleus, or other RTOS) and from the user-level application; it can run any combination of hardware representations. It can be integrated with virtual prototyping and emulation tools to establish the kind of interoperable approach to embedded system development required by today's complex embedded system designs.

With designing for low power at the forefront of many engineers' minds, an integrated hardware/software, system-wide approach to embedded system development is essential. A broad portfolio of tools for both hardware and software development are now available to help design teams achieve their design goals and get to market on time, on spec, and on budget.

END

# THIS IS NO WAY TO ENSURE SOC ARCHITECTURE SUCCESS

## *We Have a Better Way!*

At Carbon, we ensure customer success by providing a unified virtual platform to model, verify and optimize your SoC's architecture, performance and firmware before you implement your design, saving time and reducing the risk of a re-spin.

### **Only Carbon provides a complete, off-the-shelf solution that includes:**

- Cycle and instruction accurate models for all ARM® IP
- Pre-built ARM virtual reference platforms including Cortex™ A9 and A15
- Industry leading system-level debugging and profiling for HW and SW engineers

Your performance measurements are only as accurate as the platform on which they are made. The only way to achieve certainty in your performance data is to use models and systems that are accurate to your final implementation. Carbon models are 100% accurate.



**carbon**  
design systems

+1-978-264-7300  
[www.carbondesignsystems.com](http://www.carbondesignsystems.com)  
[www.carbonipexchange.com](http://www.carbonipexchange.com)