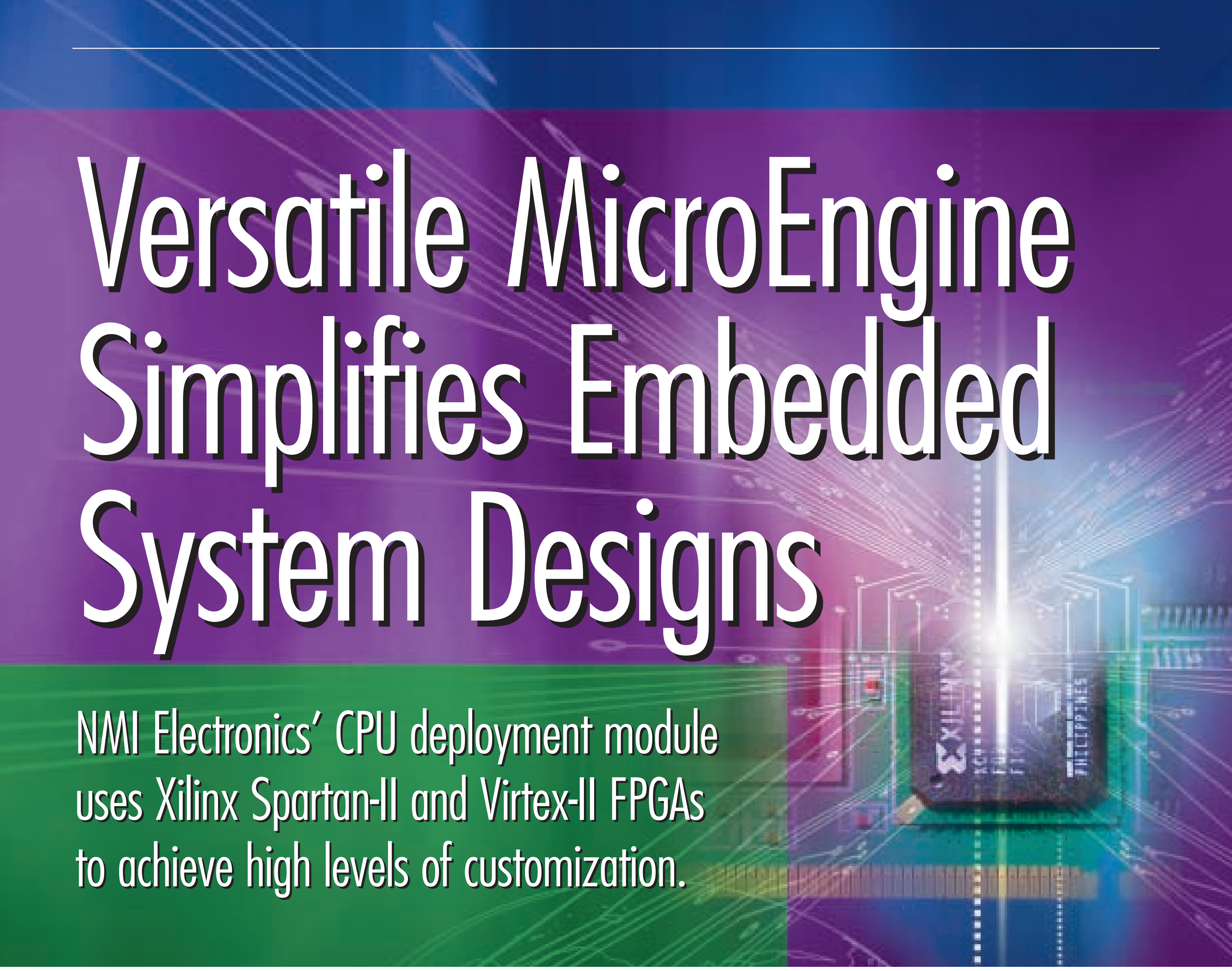

Versatile MicroEngine Simplifies Embedded System Designs

NMI Electronics' CPU deployment module uses Xilinx Spartan-II and Virtex-II FPGAs to achieve high levels of customization.

by Kevin Heawood
Vice President, Strategic Marketing
Intrinsyc Software, Inc.
kheawood@intrinsyc.com

Typically, embedded systems are designed around a specific CPU architecture that comprises a 32-bit high-performance processor unit, volatile and non-volatile memory, and a set of peripherals and interfaces specific to that system or to a particular application. With on-chip clock speeds reaching 400 MHz to 700 MHz and board-level clock speeds as fast as 133 MHz, system design has become increasingly time-consuming, complex, and risky.

Using a single-board computer (SBC) or companion chips and interface logic to reduce risk can be difficult, however, as it is not always possible to find an SBC with the correct peripheral mix and the interfaces that go with a particular CPU. Even if you do find the right SBC, the peripherals may be too expensive for your application, or otherwise inappropriate.

NMI Electronics Ltd. has developed an SBC, or more specifically, a deployment module, that greatly simplifies system design. The module contains all the main components of a 32-bit CPU system, including volatile and non-volatile memory, but it uses either a Xilinx Spartan™-II or Virtex™-II FPGA to provide a completely programmable peripheral set and an interface that can be specifically tailored to any application (Figure 1).

Programmable Interface and Companion Chip

Most SBCs have a predefined interface and set of functions. The interface is usually based on an industry standard, such as PC/104, or on a combination of a standard interface and a custom interface defined by the board manufacturer.

Peripheral functionality is normally provided by standard chipsets and is determined according to whatever the device or CPU manufacturer considers the most commonly requested functions. Beyond a few basics (for example, baud rates on UARTs or display resolutions), the mix of peripheral functions is neither flexible nor programmable, and so may be inconvenient or inappropriate for your application.

Using an FPGA effectively eliminates these restrictions. The FPGA can be placed onto the CPU local bus and closely coupled with the memory subsystem, thereby creating a highly programmable companion and interface chip.

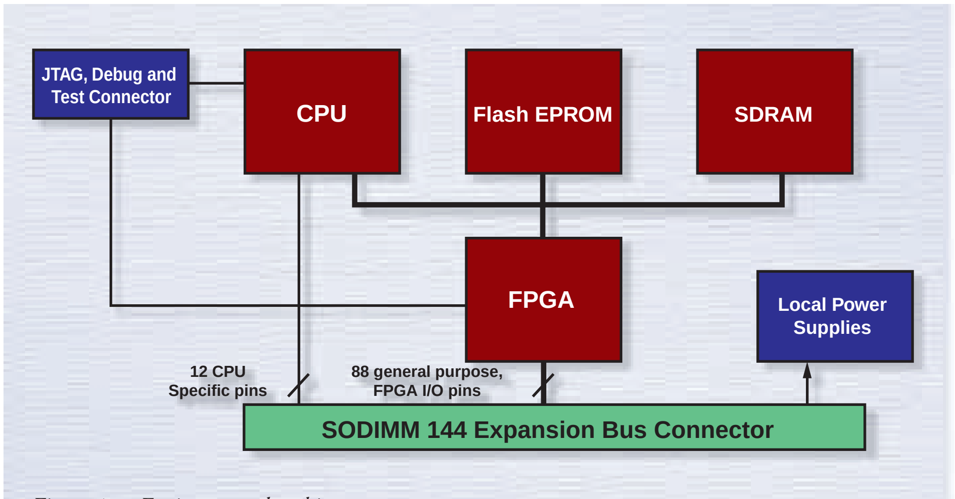
JTAG, Debug and Test Connector
CPU
Flash EPROM
SDRAM
FPGA
Local Power Supplies
12 CPU Specific pins
88 general purpose, FPGA I/O pins
SODIMM 144 Expansion Bus Connector

Figure 1 - μEngine general architecture

Interfaces

This FPGA system architecture allows you to implement the interface you require to the rest of the product hardware in a way that provides the optimum solution for your application. NMI supplies a number of standard interfaces, including the following:

- PCI
- ISA
- PCMCIA.

It is also possible to support the CPU local bus or a custom-designed interface.

Peripherals

Because almost every application has a unique set of requirements, the peripherals required for each specific system are as diverse as the applications themselves. Using the FPGA as the companion chip to fulfill these requirements, it is now relatively simple to mix and match a range of peripherals in a way that meets the exact needs of your application.

Examples of peripherals that can be included in the FPGA are:

- UARTs
- SPI, I2C, and AC97 serial interfaces
- Display controller (LCD or CRT)
- Stepper motor controller
- Camera frame grabber
- PCI slave and host interfaces
- PC/104 interface.

Accelerators

You can also place application-specific accelerators (co-processors) into the FPGA. These accelerators assist the CPU in the performance of specific functions. Examples of such accelerators include:

- 2D display assistance
- Hardware cursor support
- DSPs.

Companion Chip Application Example

Imagine you are working on an automobile navigation system that requires high-performance graphics, an interface to CAN bus, and two serial ports – one for interface to a mobile phone and one for diagnostics. It may not be possible to find a CPU and companion chip with that particular peripheral mix.

By using an FPGA-based companion chip, however, you can implement a PCI host bridge as an interface to a high-performance standard graphics chip, plus a CAN controller, and two UARTs, all within the FPGA. You could even implement a DMA controller to feed the graphics chip and service the CAN controller and UARTs, which frees up the CPU to perform more compute-intensive tasks.

NMI MicroEngine

Using the FPGA as the basis for the system interface and peripheral companion chip also makes it possible to isolate the CPU/memory/FPGA subsystem and mount this onto a small printed circuit board.

In fact, this is what we at NMI have done with our MicroEngine (μEngine). The μEngine is a small form factor deployment module that contains all the key elements of a high-performance, processor-based system (CPU, flash memory,

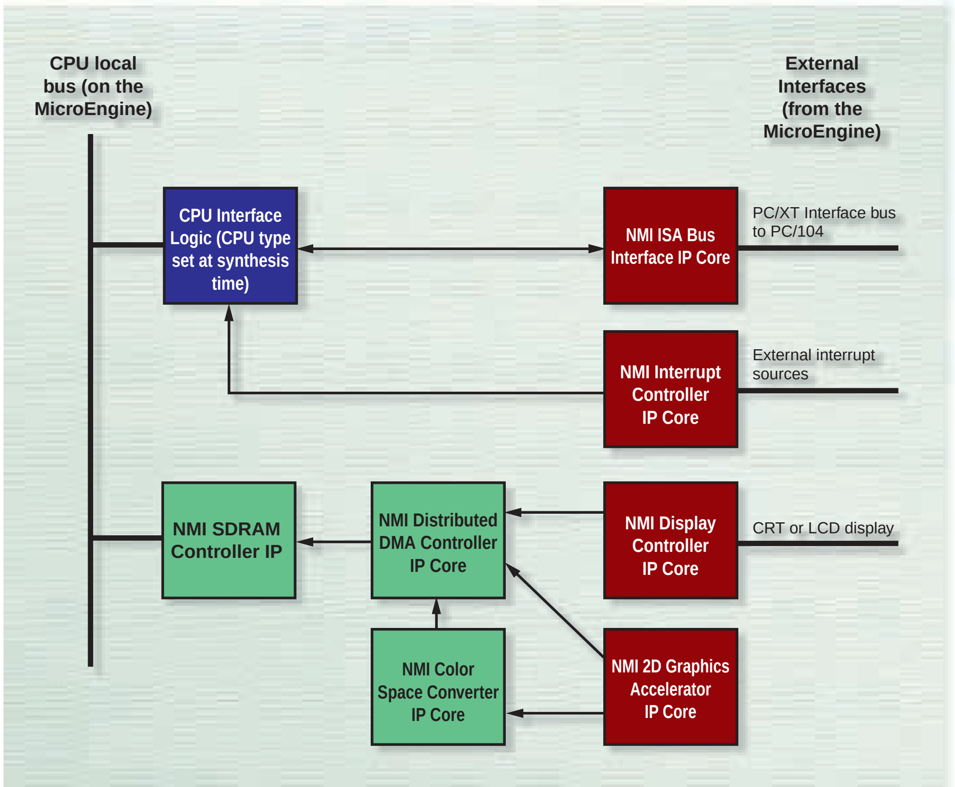
CPU local bus (on the MicroEngine)
External Interfaces (from the MicroEngine)
CPU Interface Logic (CPU type set at synthesis time)
NMI ISA Bus Interface IP Core
PC/XT Interface bus to PC/104
NMI Interrupt Controller IP Core
External interrupt sources
NMI SDRAM Controller IP
NMI Distributed DMA Controller IP Core
NMI Display Controller IP Core
CRT or LCD display
NMI Color Space Converter IP Core
NMI 2D Graphics Accelerator IP Core

Figure 2 - Example μEngine FPGA design

SDRAM), but which uses a Xilinx Spartan-II or Virtex-II FPGA to provide the system interface and peripheral functions. This arrangement provides a totally flexible core module, and it enables you to include precisely the peripherals and interfaces you need for your system. It also means that you can use the same basic board in a wide range of equipment.

In addition, the µEngine addresses the conceptually simple, yet practically more difficult, problem of designing microprocessor systems with high-speed external clocks and buses. The µEngine is, in its own right, a self-contained, pretested, high-performance microprocessor subsystem. All it needs to "run" is power.

In other words, all you need to implement your application is a baseboard containing a power supply and the specific interface logic to suit your application. In many cases, the baseboard can be relatively straightforward, and use lower technology design and less stringent manufacturing rules than the high-speed µEngine design.

The connection between the µEngine and the baseboard is made via an industry-standard, 144-pin, SODIMM connector that carries both power and logic signals. Eighty-eight pins of the interface are connected to the FPGA and are completely user-programmable. Twelve CPU-specific pins carry such dedicated functions as serial ports, ADCs, DACs, or USB, depending on the CPU deployed on the µEngine (Figure 2).

The image for the FPGA is held in the µEngine's flash memory and is completely reprogrammable. You can even place more than one FPGA image on a µEngine, enabling it to support multiple baseboards. It does this by means of a mechanism that identifies the type of baseboard it is plugged into at power-up and automatically loads the correct FPGA for the application.

In addition, the ability to isolate the CPU from the baseboard allows you to plug different CPU-based µEngines into the same baseboard. This is one of the µEngine architecture's greatest advantages. It enables upgrades of CPUs as well as the use of CPUs of varying performance levels, such as an application that requires modest graphics performance in an entry-level product and high performance in another, high-end product. For instance, the entry-level product might be based on a Hitachi SH3 (without FPU) 100 MHz µEngine (Figure 3) – and the high-end product might be based on a Hitachi SH4 (with FPU) 200 MHz µEngine (Figure 4). Both units would use the same baseboard.

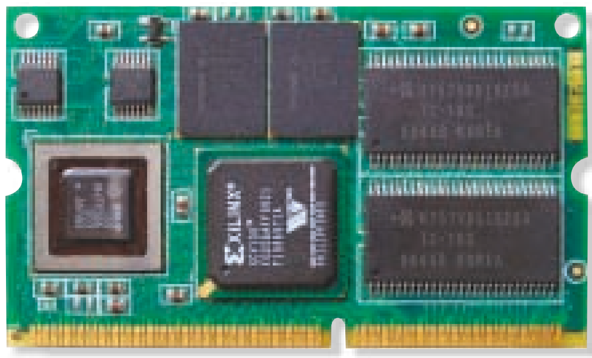
Figure 3 - Hitachi SH3 µEngine with Virtex XCV100

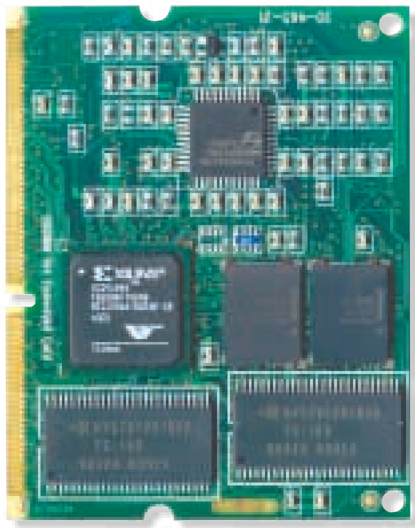
Figure 4 - Virtex XC2V1000 implemented on Hitachi SH4 µEngine

FPGA Intellectual Property

FPGA IP cores for the µEngine are available from many sources:

- NMI provides a wide range of proven IP (for example, PCI host bridge, display controllers, UART, frame grabber, 2D graphics accelerator).
- Xilinx LogiCORE™ IP
- Third-party IP
- Your own IP
- Custom-developed IP.

These elements can be freely mixed in the µEngine to produce the unique functionality required for any application. The NMI deployment module has many features that simplify integration into the final system. For instance, because most systems using high-performance CPUs are running an embedded operating system, such as Windows® CE.NET, we have provided Windows CE software drivers for all of our FPGA IP on the full range of µEngines.

What's more, you can populate a µEngine with various FPGA densities: 50K to 200K gates on Spartan-II FPGAs, 50K to 300K gates on Virtex-E FPGAs, and 250K to 1M gates on Virtex-II FPGAs. This variable gate population makes the µEngine the most cost-effective solution for almost any application, interface, peripheral, or accelerator mix.

Lastly, to ease portability from one FPGA device to another, our FPGA designs use only high-level description languages.

Conclusion

NMI developed the µEngine deployment module through imaginative use of FPGAs in CPU-based systems, creating a high-performance module that provides extraordinary hardware flexibility and upgradeability. The availability of FPGA IP and reference designs facilitates rapid and low-risk development of new products and applications, allowing companies to focus on adding value, rather than having to reinvent the core technology.

NMI offers several development platforms, enabling you to easily evaluate the µEngine and its associated IP.

Editor's note: Since this article was written, NMI Electronics was purchased by Intrinsyc Software Inc. For more information on the innovative use of Xilinx FPGAs in µEngines, visit www.intrinsyc.com/products/microengine/.