

Developing ARM Cortex-M Class Processor-based Systems

By Puru Mishra, Spencer Saunders, Mark Onions, ARM

Hardware and Software development using Cortex-M class processor requires considering a range of IP and tools options very carefully. Prototyping of a Cortex-M based system typically requires a) putting together required hardware component IP of the design in an emulation environment, b) validating the system, c) running relevant benchmark applications using application development, debug and trace tools, and d) fine-tuning or even extending the design until the product requirements are met. Each stage requires taking well-informed decisions – both from IP and tools perspective. For instance, the diversity of processors from ARM within the Cortex-M class processor family, and availability of a range of peripherals including ARM Primecell peripherals, mandates that functional IP blocks with correct performance point be chosen in order to design an optimal MCU solution for a given application. Choosing a wrong microcontroller solution can be a costly mistake. Hence, a hardware prototyping environment that makes evaluation of the most optimal MCU solution possible in a simple and cost-effective manner is important.

An emulation environment is often used for software development until the actual silicon is available. This enables software development to start in parallel with the hardware development, thus helping accelerate the time-to-market. The challenge here is to use a Cortex-M class processor-based emulation environment that is tested, pre-validated and easy to use. A reliable and easy-to-use environment with wide range of standard peripheral interfaces is desirable in this case.

Finally, if hardware IP and tools for the development of a prototyping system are pulled from separate sources, the development may not only take longer, but also result in extra investment, significantly increasing the development cost. Therefore, a single vendor that can provide a complete range of IP and tools is desirable.

Hardware System Evaluation

Before finalizing the components for a Cortex-M class processor-based MCU system, the evaluation of Cortex-M processors, ARM Primecell peripherals and third party peripherals in a hardware development environment is a requirement. There are multiple options available in the market today in order to carry out this evaluation.

A traditional approach is to use separate hardware platform boards, each based on different Cortex-M class processor, together with an on-board FPGA or structured ASIC chip to port third party or in-house peripheral IP. Once the evaluation has been done by running benchmarking programs, the most optimal system can be chosen. This approach, however, is not suitable for many reasons. The hardware development boards are not available for new Cortex-M processors. It takes at least six months to a year for a standard MCU development board to be available in the market. Also, managing separate boards requires a lot of duplicate effort, thus impacting the cost as well as the development time of the project.

Another approach would be to take all the hardware IP – including ARM Cortex-M processors, Primecell peripherals IP and third party IP, port them on an FPGA or structured ASIC to create a single system where an exhaustive evaluation can be carried out. However, access to the RTL of the IP could prove to be a huge upfront investment. Also, licensing the IP would typically require going through many standard legal processes that are often time-consuming. Therefore, this option is not suitable.

The most suitable option, however, would be a hardware development system that provides access to relevant ARM IP – Cortex-M processors and ARM PrimeCells - with simple terms and conditions, specifically for evaluation purposes.

Configuration and Validation of a Hardware Development System

A system using lots of third party IP requires a lot of effort in validation. Even when all the IP is available, integrating and validating the system will significantly add to the development cycle and delay the product development. Therefore, IP supplied in a Cortex-M hardware development system with pre-validated and tested example systems will help accelerate the development time. The pre-validated systems can also allow applications and libraries to be developed and tested in an easy and quick manner.

Today, a Cortex-M based system implemented on an FPGA can run at speeds of around 50MHz. This is similar or very close to the speed of a typical embedded microcontroller in silicon. Therefore, an FPGA based prototyping and evaluation system is ideal. Without sacrificing on the performance, it provides the flexibility of easily configuring the hardware components on the board. This enables fast development of an optimal Cortex-M class processor-based MCU solution.

Other Factors

When developing a Cortex-M based system, a developer is required to use a number of development tools for various tasks. For instance, an FPGA-based development system would require the use tools to a) design and configure the hardware system, b) download the hardware system on FPGA, c) download application code into the system, d) debug application code, e) trace execution of application code, and f) optimize application code. With the large number of tools required, it is desirable to choose a vendor that can provide the complete flow of development tools as well as providing a hardware development system.

Further, while evaluating a development system it is also important to look at the list of interfaces available on the system. It is important that a system with a complete set of interfaces be chosen. It is equally important that there is an option in the system that allows users to design their own interfaces and extend the system, either using on-board resources (FPGA) or using companion boards that can be attached to the main board.

Keeping the factors discussed above in mind, this paper evaluates the Cortex-M based Microcontroller Prototyping System (MPS)

available from ARM. The ARM Keil MPS provides a simple system to aid evaluation and prototyping of an embedded design based on ARM Cortex-M class processors. It allows early evaluation of Cortex-M class processors enabling selection of the correct processor for an application. The simple and flexible hardware development environment, combined with an integrated software environment to compile and debug software, makes this a suitable solution from concept through to final silicon.

Microcontroller Prototyping System (MPS)

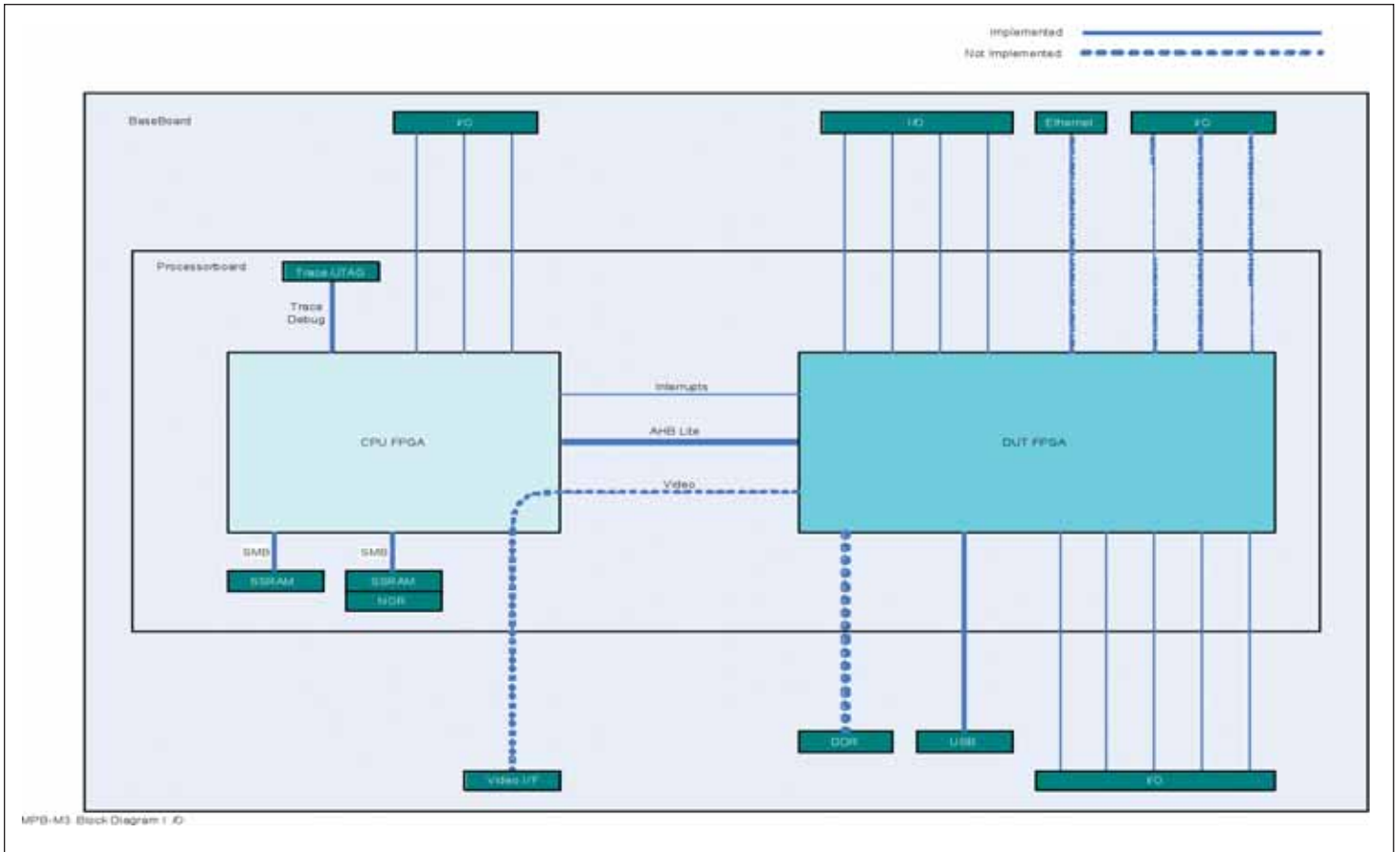
The MPS is based around two Altera Stratix III EP3SL50 FPGAs. The first holds the processor and memory subsystem (CPU), including 64MB NOR Flash and 8MB SRAM; the second FPGA is for hardware development (DUT). The processor FPGA is encrypted to allow a simple click-through license agreement (EULA) to be used for all supported processors. This means one can begin evaluating the selected processor quickly and without the need for access to the processor RTL.



The CPU FPGA can also accept a non-encrypted image, so once the processor RTL has been licensed, licensees can make modifications and update the CPU FPGA with their own image.



To aid prototyping and allow the development work to start quickly, the MPS has an array of peripheral interfaces. Some are supported via the example system while others have just the physical interfaces, to be used together with in-house IP which would be programmed into the DUT FPGA. The list below details the peripherals and level of support.



Peripheral	H/W Support	S/W Support	Note
Debug/Trace	ULINK2, RVI/RVT	MDK, RVDS	CPU FPGA
NOR Flash (32bit 64MB)	CPU FPGA	MDK, Boot Monitor	CPU FPGA
SRAM (32bit 8MB)	1 wait state CPU FPGA	MDK, Boot Monitor	CPU FPGA
UARTs	PL011 RTL	Selftest, Boot Monitor	DUT FPGA example
AC97	PL041 netlist	Selftest	DUT FPGA example
SD/MMCard	PL181 netlist	Boot Monitor	DUT FPGA example
Character LCD	RTL	Selftest, Boot Monitor	DUT FPGA example
LEDs/switches	RTL	Selftest, Boot Monitor	DUT FPGA example
USB 2.0 HOST/OTG	USB Chip	None	NXP ISP1761
Ethernet 10/100	Phy only	None	Requires MAC IP
CAN	Phy only	None	Requires IP
LIN	Phy only	None	Requires IP
FlexRay	Phy only	None	Requires IP
Video (up to XVGA)	Phy only	None	Requires IP

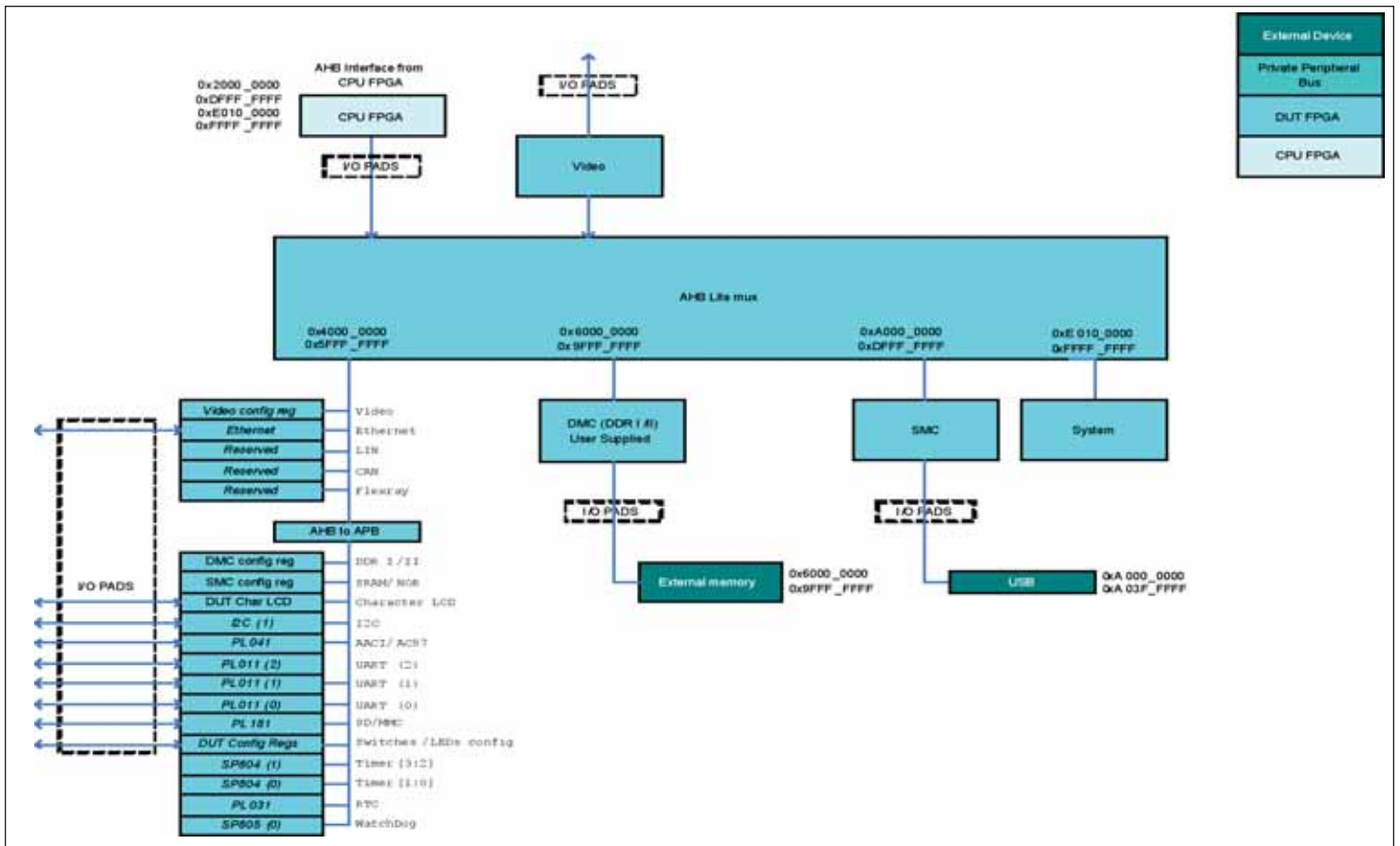
Table 1

In addition to the list of peripherals shown in Table 1, the DUT FPGA has a 'Childboard' interface to allow the expansion of the I/O features of the system. This is a simple interface of FPGA I/O pins together with power pins enabling users to use 'off the shelf' boards or develop their own for custom requirements. A selection of 'Childboards' are available from Gleichmann (Table 2) that enables addition of I/O functions to the MPS, ranging from simple I/O breakout boards with LEDs and headers to complex gigabit Ethernet Phys or DDR2 memories.

Childboard	Product Number	Features
Connector LED board	HC-COLEv2	Five 26pin 0.1" pitch IDC headers. Each signal connected to LED and these can be turned on/off with jumpers
2 Channel Gbit Ethernet	HC-ETH2v2	Two National DP83865 phy's with magnetics. MII, GMII and RGMII interface supporting 10/100/1000BASE-T
SDRAM	HC-SDRv2	256MB organized as 32M x64-bit at 120MHz
DDR2	HC-DDR	1GB of DDR2 RAM, product in development
NAND	HC-NAND4v2	Four 1GB devices with each with a separate 8-bit interface on the childboard connector
Synchronous SRAM and NOR Flash	HC-FRMEMv2	Two synchronous flow through NiRAM 1Mx32 (4MB) 8.5ns access time and two NOR Flash memories 16Mx32 (64MB) 70ns access time

Table 2

To accelerate evaluation and development, an example system is shipped with the MPS for the DUT FPGA. This FPGA image supports most of the features on the board and is supplied with the example software and hardware design files necessary to rebuild both the software and hardware image of the FPGA.



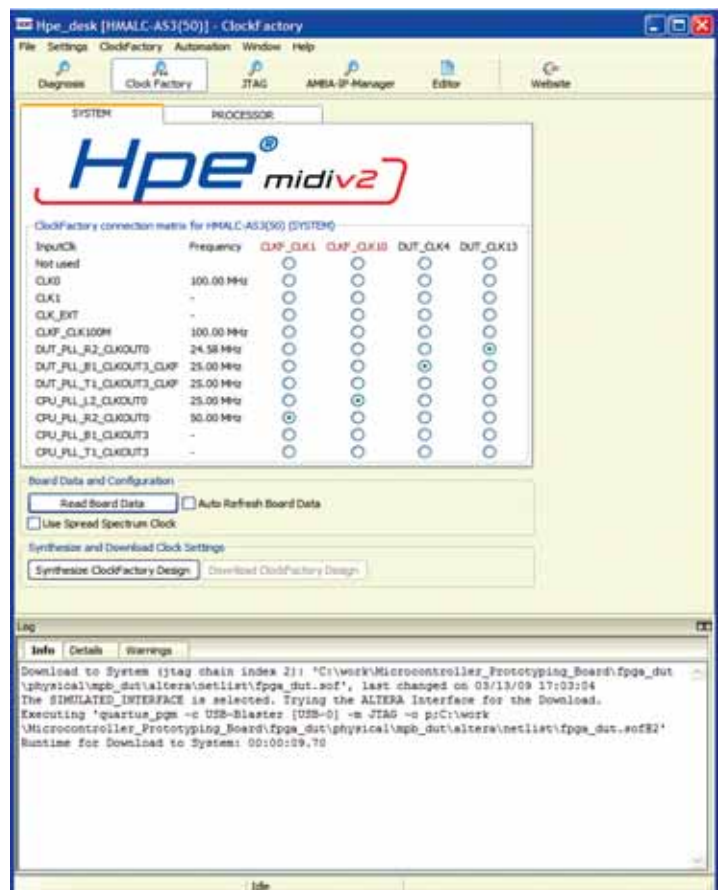
The next section of the paper will cover the design flow to rebuild, download and execute both the DUT FPGA image and software for execution on the MPS.

Hardware Design The example system is supplied as a mixture of RTL and netlists (for high value IP) allowing users to rebuild the FPGA image. The netlists for some IP blocks ensures that only a simple click-through license is required. The top level architecture above the IP blocks is RTL-based to enable easy modification.

Users need to edit and modify the RTL to add or remove features as required. They can use the free Altera Quartus® II Web Edition tools to synthesize, and then place and route the design for the FPGA. The example scripts used can easily be modified and contain the synthesis and timing constraints required to rebuild the FPGA image, and they can be used in the GUI or on the command line for automated builds.

Downloading and Configuring the MPS Users have now created the FPGA image as a .SOF file for the MPS. This can be downloaded into the MPS system via the HPE-Desk™ software supplied with the MPS. This is a Windows-based application that downloads images to the FPGA via the USB port on the front of the MPS.

The HPE-Desk offers two options for downloading the image into the DUT FPGA. The first is 'download sof file to System FPGA' which is a fast method of downloading directly into the FPGA.



However, this method is volatile meaning that when the system is rebooted, the FPGA is configured with the current system Flash image and any changes made to the FPGA image will be lost. This method is useful to save time during prototyping where users would like to quickly test small changes to the FPGA image.

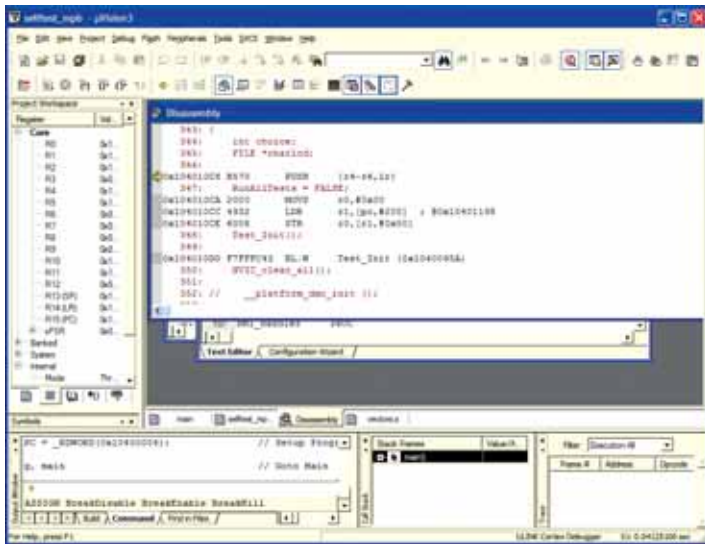
The second option takes longer but is non-volatile as the image is downloaded to the system Flash. The 'download to System Flash Memory' option is ideal for making more permanent changes to the system image and should be used later in the development phase or to make 'milestone' changes to the system.

The second feature of the HPE-Desk is the ability to route different clock signal outputs from the FPGA back into the FPGA's clock inputs. This is basically a cross switch allowing users to drive clock sources (shown as rows) to clock inputs (shown as columns) in the FPGA. The default configuration is suitable for the example system, but if users wish to change the PLL in the DUT FPGA to create different clock frequencies for the processor and AHB lite interface then they can use this to switch between clocks without having to rebuild the FPGA images, saving time.

Software Design

Once the hardware image is downloaded and working on the MPS users can begin developing and running a software application. This is a straight forward exercise as the MPS and its example system are fully supported by the Keil software development tools.

The MPS is supplied with an evaluation version of MDK-ARM (Microcontroller Development Kit) which is code size limited to 32KB, and the evaluation version can be upgraded to a full version of MDK-ARM by contacting Keil or our local distributor. The MPS also includes a ULINK2 adapter which allows the MDK-ARM to download application code to the target and debug it in JTAG or SWD via the PC's USB port.



Project Setup and Configuration

MDK-ARM contains a Device Database which allows users to easily setup and configure their project. Within the Device Database select 'ARM' and the required processor [Cortex-M3/M0] as the target device and the required tool options and customized dialogs are automatically configured. As users continue their application development, only those options that are relevant to the selected device will be displayed, thus preventing selection of incompatible directives.

General Debug and Analysis

The μ Vision4 IDE/Debugger offers a wealth of debug and analysis tools to help users analyze and optimize their application code. In the Debugger, the Watch Window displays the values of automatic variables in the current function, the Memory Window displays various memory areas, and the Serial Window provides a terminal output for the on-chip UART. The Flexible Window Management System introduced in μ Vision4 enables developers to use multiple monitors and provides complete control over window placement anywhere on the visual surface.

μ Vision4 includes a Logic Analyzer which records the values of variables and peripheral I/O signals over time and displays them. These signal values can be displayed in three different formats: bits, showing outputs as a logic level 0 or 1; State, showing stage changes of a value; and Analog, showing a graphical representation.

Cortex-M Class Processor Debug & Trace

MDK-ARM and ULINK2 have extended features for debugging applications running on Cortex-M3 and Cortex-M0 processor-based systems using the standard JTAG, or the advanced Serial-Wire debug modes. Additionally, Data Trace Windows provide information from a running Cortex-M3 processor-based system for program data, exceptions, variables and printf-style outputs. For example, one may use the Trace Records window to view all the trace records captured during the debug session, including overflows and timestamps; while the Exception Trace windows show which exceptions have taken place, including the number of times they have occurred and how long the system spent handling these exceptions. Users may also view printf-style debug or other program-specific information using the ITM Viewer window or keep a track of the events in their application using the Event Counter window.

Conclusion

This paper has evaluated the suitability of MPS for Cortex-M class processor based hardware and software development environment. MPS provides a unique development environment that allows evaluation and then selection of the correct Cortex-M processor for a given application. It allows easy and fast prototyping of Cortex-M processor-based systems. A programmable and pre-validated platform, it is tailored to the needs of designers working on Cortex-M based designs. Applications and libraries can be developed and tested before the silicon is available thus accelerating the time to market. Looking at the requirements of the industry today, MPS provides an ideal choice for both hardware and software system development based on Cortex-M class processors.

Technology and Solutions to advance your ARM SoC designs

- Start software development for ARM[®] processor-based designs 9 to 12 months prior to silicon availability using Synopsys Virtual Platforms
- Support for FPGA implementation including a free evaluation of the Cortex[™]-M1 processor
- DesignWare[®] IP solutions for the ARM AMBA[®] 2.0 and AMBA 3 AXI[™] protocols featuring automated subsystem assembly with the coreAssembler tool
- Eclipse[™] Low Power Solution, reference methodologies, LPMM and VMM-LP manuals for high-performance, low power ARM SoCs



For more information, visit
www.synopsys.com/arm