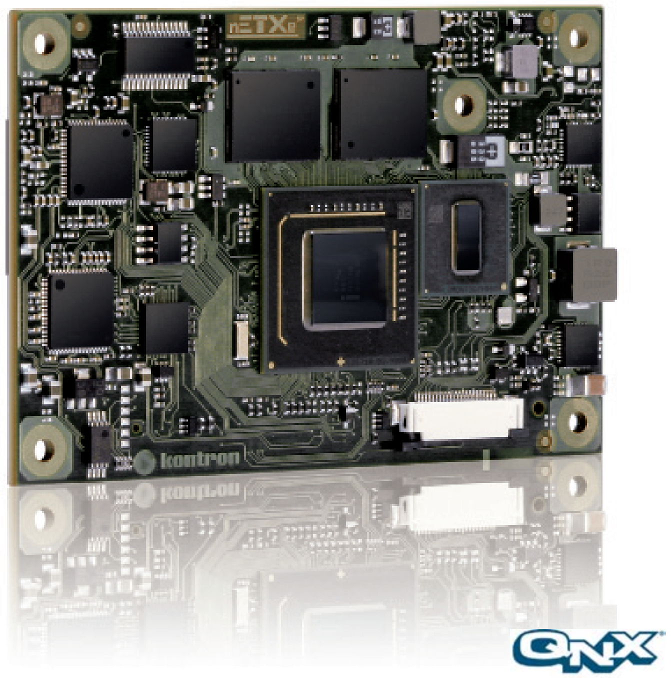
» Kontron Whitepaper «



QNX fastboot technology RTOS for X86 nanoETXexpress

nanoETXexpress saves battery power with QNX fastboot technology!

Wake up x86 devices from S5 sleep state in ms

Fastboot technology for ultra small Intel® Atom™ processor designs

» nanoETXexpress Computer-on-Modules

» Fastboot

» Instant-on

» Energy efficient

» Mobile Devices

Wake up x86 devices from S5 sleep state in ms

Fastboot technology for ultra small Intel® Atom™ processor designs

By using QNX® fastboot technology on Intel® Atom™ processor based ultra-small nanoETXexpress computer-on-modules, systems designers can achieve instant-on functionality while taking advantage of the x86 compatibility and high performance per watt of Intel® Atom™ processors. Developers can achieve dramatic boot-time and performance gains by replacing the BIOS with customized early initialization of peripherals. Furthermore, QNX® fastboot technology helps developers to profit from the low power efficiency of nanoETXexpress technology by extensively using the zero power S5 sleep state (off) instead of power consuming sleep modes (S1 – S4) at comparable wake up times.

CONTENTS

The BIOS

Traditionally, x86 systems have relied on boot firmware called the BIOS (Basic Input Output System), which provides a wide range of device compatibility and detects the system configuration at boot time. However, neither of these capabilities rank high in embedded application requirements. Embedded solutions are typically niche, purpose-specific, or mass-produced devices with the exact same configuration for each device. Some of these applications require boot-times in milliseconds. So the question is: How can developers achieve a fast, BIOS-less boot on open x86 processors, rather than on dedicated controllers? Such an approach would be ideal for:

» Real-time control and communication in automotive and security applications,
» machine controls such as soft PLCs and HMIs,
» mobile applications for measurement and diagnostics,
» mobile medical applications, and
» all other "sleeping" (Sleep state S5 = off) applications with a wake-on-x functions

BIOS boot time

The BIOS can be seen as a miniature operating system responsible for interfacing with the hardware. The portion of the BIOS that has the greatest effect on boot time is the boot code. This code, which executes on system power up, detects and initializes devices attached to the system.

Devices that require BIOS initialization include the keyboard, mouse, interrupt controllers, video card, CPU, cache, RAM, PCI devices, USB devices, sensors and disk drives. Besides having to detect and initialize each of these devices, the BIOS must also handle variations in quantity (for example, RAM) or features (for example, CPU support for 64-bit addressing). The BIOS must also execute the boot code extension for each device that has a BIOS expansion ROM. All this work is time-consuming and usually superfluous for embedded devices since in most cases they have a fixed configuration.

Booting from the BIOS allows a system design to support a wide range of hardware configurations and devices, but this flexibility comes at the cost of slower boot time. Since embedded systems typically consist of fixed hardware configurations, they should not have to bear these costs.

Fastboot time

BIOS boot takes time as the BIOS tries to support a multitude of hardware configurations. An obvious solution is to eliminate the BIOS boot portions that are not required for a particular embedded application. For the most part, this suffices to reduce boot times from a large number of seconds to under one second.

To further improve boot times, developers can hard-code interrupt assignments, boot from a linearly addressed storage device (EEPROM or flash), and defer resource allocation.

Fastboot is fast mostly because of the things that it does not do; and this approach is perfect for embedded applications, which are typically designed to carry out just one or a few things very well.

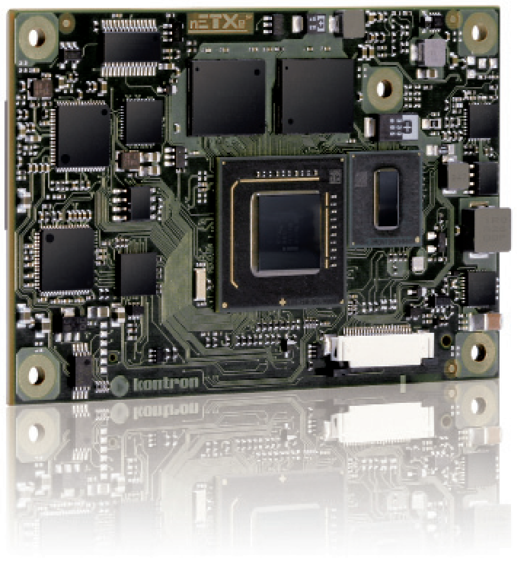


Figure 1: The Kontron nanoETXepxress-SP is one of the first platforms to support QNX® fastboot technology and ideally suited for rugged mobile applications.

Customized boot

By customizing the boot sequence, the developer can eliminate unnecessary steps that the BIOS performs at boot time. The developer can also defer steps that are not required at boot time until they become absolutely necessary.

In a BIOS-based system, the BIOS assigns and configures the interrupts; in a fastboot configuration, the developer is responsible for routing interrupts. The most cost-effective way to do this from a boot time perspective is to hard-code the interrupt assignments so that they are known at compile time. This approach offers a simple yet effective way to improve boot times and to reduce the complexity of boot code. It also allows the developer to assign interrupts in ways that can improve performance. For instance, the developer could dedicate channels to high-traffic interrupts and share interrupt lines among low-traffic interrupts. Sharing IRQs across multiple devices incurs the run-time cost of executing the interrupt service routine for all devices associated with the triggered interrupt, irrespective of which device actually triggered the interrupt.

PCI resource allocation does not have to occur at boot time and can therefore be deferred to a later stage when a driver needs to access a device on a PCI bus. Both memory and I/O resources can be allocated after the system has booted; the resource requirements are obtained by probing the PCI device. PCI interrupts, much like ISA interrupts, are configured by the developer so that the interrupt assignments are known at compile time.

Booting from a linearly mapped device such as flash or EEPROM helps to improve boot times and to reduce the complexity of the boot code by avoiding the slower BIOS calls required to read an image from disk. The final step of the fastboot loader consists of loading the OS image into the RAM and starting the execution of the OS.

As illustrated in Figure 2, the boot process consists of 2 phases:

1. Initial Program Loader (IPL) phase — Handles basic initialization, locates the OS image, copies the Startup portion of the OS image, and skips to it.
2. Startup phase — Finishes the hardware initialization, copies the rest of the OS image to the RAM, and populates the OS data structures.

The developer now has to perform steps that the BIOS previously handled. This has the following advantage; the developer is free to pick and choose what needs to be done, the disadvantage being, however, that this takes up developer time.

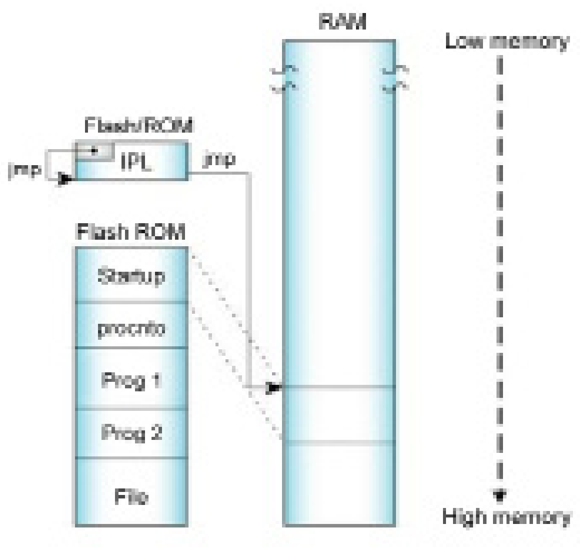


Figure 2: The boot process.

What Fastboot and BIOS have in common

Having a fully customized boot loader also means the developer must carry certain things out in the same way that the BIOS would have done. For instance, the developer needs to initialize the memory controller, CPU and cache; enable access to high memory and enter protected mode. A fastboot loader doesn't perform these steps any faster than a BIOS.

Fastboot BIOS

This paper has focused on either a full BIOS-based boot or a completely customized fastboot-based boot. The fastboot approach requires a non-negligible amount of developer time, unless it has already been implemented by the board supplier or operating system vendor. Researchers at Intel® have investigated some ways to reduce boot times of BIOS-based boots. A paper entitled "Fastboot BIOS"1 has been published by them which lists several potential boot-time optimizations. One of the recommendations stated, for example, is tailoring a BIOS to a specific embedded system, much like the recommendations in the “Customized boot” section, while other recommendations focus on optimizations such as enabling Intel® SpeedStep technology that helps reduce boot times in both BIOS and fastboot configurations.

The optimizations proposed in the Intel® “Fastboot BIOS” white paper which do not require any developer time may be sufficient for some embedded applications and offer a reasonable compromise between reduced boot times and developer effort.

Conclusion

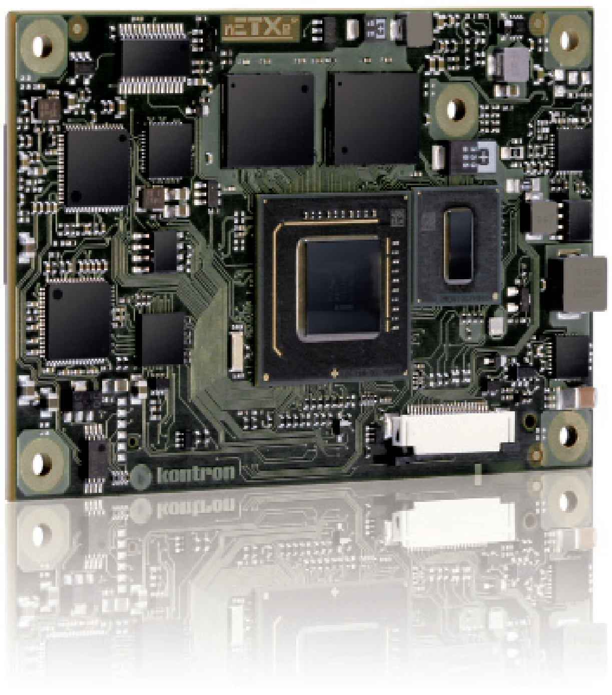
A fastboot-based system will boot significantly faster than a BIOS-based system as the designer can select or discard boot steps based on a known hardware configuration specific to the embedded application.

The versatility of BIOS-based systems is not required in the embedded applications targeted by the Intel® Atom™ based platforms, so it seems a waste to incur the boot time cost when no benefit is gained. However, discarding the BIOS isn't as simple as flipping a switch. Moreover, it requires developer effort to implement boot steps typically handled by the BIOS.

For a visual demonstration of to what extent fastboot techniques can improve boot times, refer to the YouTube video, “QNX® fast boot on Intel® Atom™ / Kontron nanoETXexpress-SP” at http://www.youtube.com/watch?v=yTUweJKAUfk

For more detailed information please visit the fastboot technology roadshow for ultra small Intel® Atom™ Processor designs: www.nanoetxexpress.org/qnx/fastboot.php

QNX fastboot technology RTOS for X86 nanoETXexpress
nanoETXexpress saves battery power with QNX fastboot technology!
kontron
QNX



First to market with nanoETXexpress

nanoETXexpress-SP Computer-on-Module is the first commercial embedded platform to support QNX® Software Systems' innovative fastboot technology for Intel® Atom™ processors. QNX® fastboot technology, which has received an "Award of Excellence" for "Most Innovative Software for the Intel® Atom™ Processor" from the Intel® Embedded and Communications Alliance, is a unique feature of the QNX® Neutrino® RTOS. It eliminates the need for a BIOS on x86 platforms, thereby reducing system costs and dramatically improving instant-on performance. Systems designers can use QNX® fastboot technology to deliver ultra-fast boot times for a wide variety of industrial, medical, military, and consumer products.

QNX® fastboot technology support for Intel® Atom™ processors also includes an HMI solution based on Adobe Flash Lite 3.1, 2D/3D hardware-accelerated graphics based on OpenGL ES, and an optimized compiler available from Intel.

Measuring just 55 mm x 84 mm, the credit card sized Kontron nanoETXexpress-SP module is compatible with the PICMG COM Express™ standard, pin-out Type 1. It is based on the highly efficient Intel® Atom™ 5xx processor series and is a perfect fit for low-power mobile applications that require excellent processor performance, high-definition graphics, PCI Express and Serial ATA, combined with longer battery life. With only one tenth of the thermal design power and one seventh the size of ULV processors of identical performance, the Kontron nanoETXexpress-SP COM offers an unprecedented performance per watt for x86-based ultra-mobile designs.

For more information, please visit the nanoETXexpress website: http://www.kontron.com/nanoETXexpress-SP

About Kontron

Kontron designs and manufactures standards-based and custom embedded and communications solutions for OEMs, systems integrators, and application providers in a variety of markets. Kontron engineering and manufacturing facilities, located throughout Europe, Americas, and Asia-Pacific, work together with streamlined global sales and support services to help customers reduce their time-to-market and gain a competitive advantage. Kontron's diverse product portfolio includes: boards and mezzanines, Computer-on-Modules, HMIs and displays, systems, and custom capabilities.

Kontron is a Premier member of the Intel® Embedded and Communications Alliance.

For half-a-decade now, Kontron has been named a VDC Platinum Embedded Board Vendor. Based entirely on user feedback, industry professionals evaluate vendors on over 45 non-product related criteria. Kontron is only one of two companies to receive the Platinum award 5-years running.

Kontron is listed on the German TecDAX stock exchange under the symbol „KBC".

For more information, please visit: **www.kontron.com**

CORPORATE OFFICES

Europe, Middle East & Africa	**North America**	**Asia Pacific**
Oskar-von-Miller-Str. 1 85386 Eching/Munich Germany	14118 Stowe Drive Poway, CA 92064-7147 USA	17 Building,Block #1,ABP. 188 Southern West 4th Ring Road Beijing 100070, P.R.China
Tel.: +49 (0)8165/ 77 777 Fax: +49 (0)8165/ 77 279	Tel.: +1 888 294 4558 Fax: +1 858 677 0898	Tel.: + 86 10 63751188 Fax: + 86 10 83682438
info@kontron.com	info@us.kontron.com	info@kontron.cn