



WP352 (1.0) July 29, 2008

CoolRunner-II CPLDs in Portable Navigation Devices

By: Nick Mehta and Arthur Yang

Asking for directions is fast becoming a foreign concept, much like the art of folding a map or a newspaper. This impending extinction is attributed to the increasing popularity of the Global Positioning System (GPS). GPS is present in a growing number of products: automobiles, cell phones, personal data assistants, even wristwatches. Despite this ever growing presence of GPS equipped devices, dedicated Portable Navigation Devices (PND) are enjoying an all time boom, feeling untroubled by the smaller screen size and inferior performance of non-dedicated equipment.

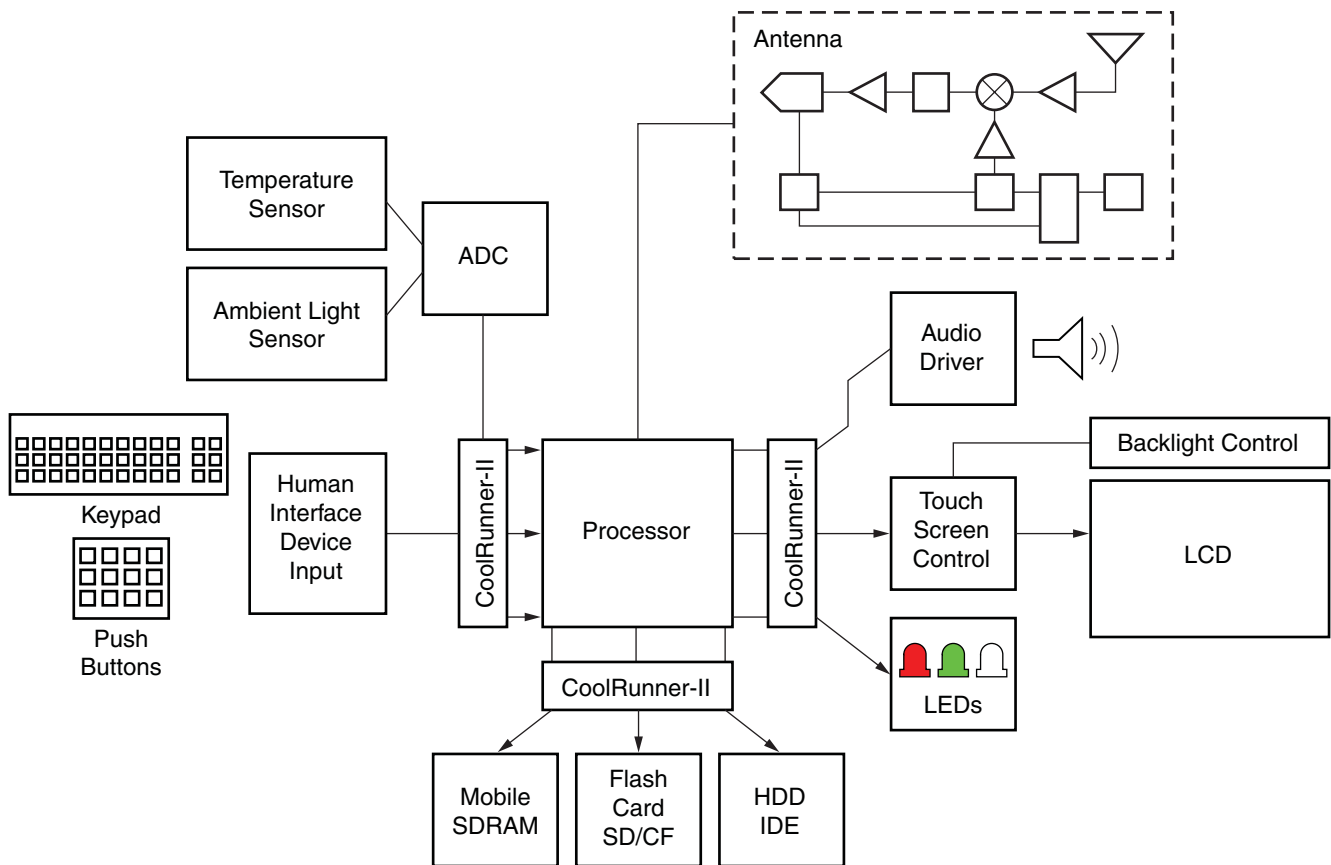
Global shipments of GPS equipped devices are predicted to reach 350 million units by 2010. The relative ease of entry into the PND market makes navigation tools an attractive market for new manufacturers. With the average unit cost of PNDs decreasing, and each manufacturer offering a wealth of differing models, new players in the market are

under great pressure to offer product differentiation to gain valuable market share.

This white paper discusses using a CoolRunner™-II CPLD to augment the processor feature set to aid a PND manufacturer to quickly produce a differentiated product.

Typical PND System

PNDs are typically built around a CPU containing an ARM processor such as the Samsung 244x. These processors are great at performing specific functions, but have a couple of drawbacks. Since the processors are only redesigned every 12-18 months, it can be hard for them to incorporate new features as quickly as they are required by the PND manufacturers. The more tasks performed by the CPU, the less bandwidth available for the primary functions of receiving data, calculating directions, and rendering images. It is a significant benefit to offload tasks from the CPU into an auxiliary device as shown in Figure 1.



WP352_01_071008

Figure 1: Typical PND Block Diagram

Achieving Product Differentiation in GPS Units

Original consumer PNDs were straightforward. They simply gave location information in the form of latitude and longitude. Today's units not only offer real-time maps and directions, but also MP3 playback and integration with cell phones using Bluetooth. There are also market-specific GPS systems with features such as traffic updates for in-car navigation, ultra-small units for runners or cyclists to measure their pace, units with sonar for fishermen, and even GPS-enabled collars for keeping tabs on the family pet. Each product requires an interface to something different—making the CoolRunner-II CPLD a perfect design fit. The following topics demonstrate some uses for the CoolRunner-II CPLDs illustrated in [Figure 1](#).

SD Card Interface

SD memory (in its various physical formats such as mini-SD and micro-SD) is now the leading memory interface for this market. Although other interfaces such as MMC and Compact Flash are still around, they represent only a small percentage of the market. Flash cards are essential for any product with picture or MP3 playback; both features are becoming more prevalent in GPS handheld units. Some high-end processors already incorporate a single SD port, but it is becoming increasingly common for multiple peripherals to talk to the processor through the SD standard. In such eventualities, the CoolRunner-II CPLD can switch seamlessly between two SD devices as shown in application note [XAPP906](#), *Supporting Multiple SD Devices with CoolRunner-II CPLD*.

Mobile SDRAM Interface

Simplify microprocessor code by allowing the CPLD to act as the memory interface. Some high-end models require multiple memory modules but a low-end product does not; let the CPLD code change, not the microprocessor. An example design where a CoolRunner-II CPLD interfaces to a Mobile SDRAM device is provided with application note [XAPP394](#), *Interfacing to Mobile SDRAM with CoolRunner-II CPLDs*.

Level Translation

As PNDs move into newer consumer areas, the connection to other components is not yet optimized for low-voltage operation. CoolRunner-II CPLDs have a minimum of two I/O banks (increasing to four I/O banks in the largest device), allowing multi-voltage interfaces to be addressed easily. Supported voltage standards include 1.5V, 1.8V, 2.5V, 3.3V, SSTL2-1, SSTL3-1, and HSTL1. 5V interfaces are supported with some external circuitry. Using CoolRunner-II CPLDs as voltage level translators is effectively free since it may not be the primary purpose of using the device in the system, but it does not burn any extra resources. The methodology of using CoolRunner-II CPLD to translate between two or more voltages is shown in application note [XAPP785](#), *Level Translation Using Xilinx CoolRunner-II CPLDs*.

Keypad Scanner

Many of the lower-end PND models cannot use a touch-screen interface because of cost constraints or size limitations. Even in the expensive models, a few buttons are designated for certain features such as power or volume control. Some form of keypad or button interface is used in most models. This is an ideal use for CoolRunner-II CPLDs because when there is inaction from users, the CPLD remains in a quiescent state and can immediately respond to a user key press without having to wake up

from sleep mode. Furthermore, the CPLD can be designed to verify user data before waking up the rest of the system. For example, many cell phones require pressing two keys in sequence before waking up to ensure that the buttons were not accidentally pressed. Application note [XAPP512](#), *Implementing Keypad Scanners with CoolRunner-II CPLDs*, illustrates how a CoolRunner-II CPLD can scan and decode an 8x8 keypad. A PND could rely on the touchscreen and have limited key access. The size of the key array is completely customizable. That is the beauty of programmable logic!

System Management

Several PND system peripherals will use a common interface to communicate with the main processor. The Inter-Integrated Circuit (I²C) bus is perfect for connecting low speed peripherals including memories, ADCs, simple displays, and other components to a system. I²C can operate at a variety of different speeds. Application note [XAPP385](#), *CoolRunner-II CPLD I2C Bus Controller Implementation* provides a working I²C controller implemented in a CoolRunner-II CPLD.

System Management Bus (SMBus), a subset of I²C, was introduced primarily to control processor based systems, mostly for system and power management, such as the monitoring of batteries and temperature and voltage sensors. Application note [XAPP353](#), *CoolRunner XPLA3 SMBus Controller Implementation* provides a fully functional SMBus controller that can fit into a Xilinx CoolRunner-II or CoolRunner XPLA3 CPLD.

Alternatively, if the I²C or SMBus controller are already present and, therefore, the CPLD does not need to act as a full controller, it can be used to incorporate extra I/O ports for the system to communicate with more I²C or SMBus compatible peripherals. Application note [XAPP799](#), *An SMBus/I2C-Compatible Port Expander* shows how easy it is for a CoolRunner-II CPLD to add extra ports into a system.

The Power Advantage

Both PNDs and in-dash GPS devices must adhere to a strict power budget. CoolRunner-II CPLDs have the advantage of minimal power consumption without the need for sleep mode states. The smallest CoolRunner-II device has a quiescent power of 13 μ A. In some portable applications, a sleep mode is sufficient and a wake-up time in the hundreds of milliseconds is acceptable. In some cases, however, the current design states are lost. When using sleep mode, the entire device shuts down. A secondary device must be used to poll for interrupts and initiate a wake-up sequence.

CoolRunner-II devices offer the unique DataGATE power-saving feature, reducing the design headaches associated with a sleep mode. DataGATE, a self-contained and user-configurable circuit allows you to disable as much of the device as you desire. Enabling DataGATE turns off any inputs in several nanoseconds, thereby shifting the power consumed by the CPLD closer to a quiescent state. This can be done periodically, such as when polling for an interrupt, or it can be dependant on a particular state of operation. Thus, portions of the CoolRunner-II device can remain active while others are in standby.

Consider the following example. The CPLD sits on an address and data bus between a microprocessor, mobile SDRAM, and SD Flash card. Data moves between each of these devices. The CPLD monitors the data activity and blocks traffic based on the particular function. If the current task is to move data from the microprocessor to the SD card, then the CPLD blocks any data to and from the SDRAM using DataGATE. Within the CPLD, a write code simply decodes the function being performed (either

from watching the address lines or by parsing frame data) and then enables/disables whichever data path is required. This is a trivial example that could be accomplished with other programmable logic devices (PLDs) by using 3-state outputs. The advantage of DataGATE is that it effectively 3-states the input of the CPLD. Power savings increase because the CPLD I/O as well as core circuitry are placed in a quiescent state, whereas alternate solutions require the entire PLD to stay in active mode. Application note [XAPP436](#), *Managing Power in FPGAs and Other Devices Using CoolRunner-II CPLDs*, gives great insight into the power of using DataGATE to aid system power management. Application note [XAPP347](#), *Decrease Processor Power Consumption Using a CoolRunner CPLD*, gives some pointers on how to lower system power by letting a CoolRunner-II CPLD handle interrupts, resets, and wake up signals.

The Security Advantage

PND system pricing makes it an attractive target for product cloning and overbuilding. To help prevent this, utilize CoolRunner-II CPLD's read/write protect security to implement a security system that prevents cloning and overbuilding. Overbuilding occurs when a contract manufacturer orders extra components for a given production run, and then builds extra products that are identical to the authentic versions.

To prevent overbuilding, design a system that requires interaction with the CPLD to perform any desired function. This security can be implemented in any number of ways. The most straightforward solution is to have a CPLD act as a data-traffic cop, directing data between the individual devices on the board. A more complex solution involves using the CPLD to implement a block cipher. The microprocessor submits a random stream of data for encryption by the CoolRunner-II device and return. This data is decrypted and verified against the original data. The CoolRunner-II CPLD's security is implemented using multiple programming bits, so any attempt to determine relevant security bits must find and disable several bits out of tens of thousands in the nonvolatile array.

The CoolRunner-II CPLD read/write security prevents readback and programming on top of the existing pattern. Thus, the device will not permit extraction of the programmed JEDEC file, nor will it allow an overlay of a modified code version on top of an existing pattern. To reprogram the device, the entire device must first be erased, in which case the design information will be lost. The key factor in any of these flows is that pre-programmed CPLDs must be provided to the contract manufacturer from a trusted source (Xilinx or an authorized Xilinx distributor). This prevents overbuilding because simply ordering more encrypted CPLDs is not useful without access to the programming file.

Designing in CoolRunner-II CPLDs

Once the system challenge has been identified, it is extremely quick and easy to get designing. Download the free WebPACK software at www.xilinx.com/webpack. There are several tutorials available, including the [Programmable Logic Quick Start Handbook](#). Alternatively, a low cost CoolRunner-II Design Kit contains everything needed to get a design running on a board at the lowest possible cost.

Conclusion

CoolRunner-II CPLDs demonstrate their usefulness in lowering power, increasing security, and providing connectivity solutions for today's Portable Navigation Devices. With a portfolio of ultra small form-factor packages, these devices can fit into even the smallest portable units. Finally, the easy-to-use software design flow helps a manufacturer introduce new products with the latest features.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
07/29/08	1.0	Initial Xilinx release. (for v1.0 only)

Notice of Disclaimer

The information disclosed to you hereunder (the "Information") is provided "AS-IS" with no warranty of any kind, express or implied. Xilinx does not assume any liability arising from your use of the Information. You are responsible for obtaining any rights you may require for your use of this Information. Xilinx reserves the right to make changes, at any time, to the Information without notice and at its sole discretion. Xilinx assumes no obligation to correct any errors contained in the Information or to advise you of any corrections or updates. Xilinx expressly disclaims any liability in connection with technical support or assistance that may be provided to you in connection with the Information. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE INFORMATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT OF THIRD-PARTY RIGHTS.