Managing Power with CoolRunner-II CPLDs

XAPP436 (v1.2) September 28, 2005

Summary

This application note demonstrates how multiple devices, including Virtex^{TM}-II, Spartan^{TM}-3, and Spartan^{TM}-3L FPGAs, can be effectively power managed by a single CoolRunner^{TM}-II CPLD. It is written with battery powered applications in mind.

Introduction

All chips draw power, but some applications are more sensitive than others to the amount drawn. Portable applications are sensitive simply because they draw from a battery. Most digital chips are designed to operate at 5V, 3.3V, 2.5V, 1.8V and so forth. This does not match well with today's battery voltages. Hence, there will be a regulator or two on most boards. Managing power will involve managing those regulators.

CoolRunner-II CPLDs were designed to operate with a core voltage of 1.8V, well suited to its 0.18 micron core, but its I/O structure supports 3.3V, 2.5V, 1.8V and 1.5V operation. Being standard low power CMOS, the I/Os also operate within that range, but are only speed specified at those voltages.

CoolRunner-II CPLDs have been successfully used as voltage translators, logic collection sites and even power management solutions. Their "early on" behavior makes them ideal for managing other chips' power. This application note focuses on using CoolRunner-II to manage power for Xilinx FPGAs, providing greater FPGA utility in a portable products. Additional CoolRunner-II qualities are shown to add value to reducing the portable product power budget in many cases.

The Power Equation

As expected, the starting point is physics, but CMOS power consumption can be deceptive. Simplistically, it follows an equation relating the switching speed of a logic gate output, times the voltage range over which it swings, times the load capacitance being driven. Adding in the static leakage component says it all.

Equation 1: $Power = V(VCF) + static\ power$

where:

V = output voltage swing

C = load capacitance

F = gate output switching frequency

$Static\ power$ = somewhat constant value for many parts (often negligible)

This equation is a guideline. Originally developed for a simple CMOS inverter, it shows a trend, but only suggests how to reduce power. Equation 1 drives most CMOS power estimation tools. Xilinx provides several ways to estimate power, including spreadsheets, application notes and the XPower software, available with the ISE design suite.

www.BDTIC.com/XILINX

The CoolRunner-II Solution

The CoolRunner-II approach to power management attacks both static and dynamic current. A battery is a charge reservoir with varying capabilities on voltage regulation, and in some cases an ability to be recharged. Each battery has its own qualities for delivering charge. Many batteries can deliver high currents for brief time periods, or smaller currents for longer times. Trade-offs can and must be made.

To get a feeling for power activity, let's look at a real world example, that was documented by Portelligent in their Report #116.02- 031023-1d. Figure 1 is a reconstruction of their measurements made on an LG cell phone that was able to record video or take a picture. Figure 1 shows power use while taking a photograph, with the cell phone camera.
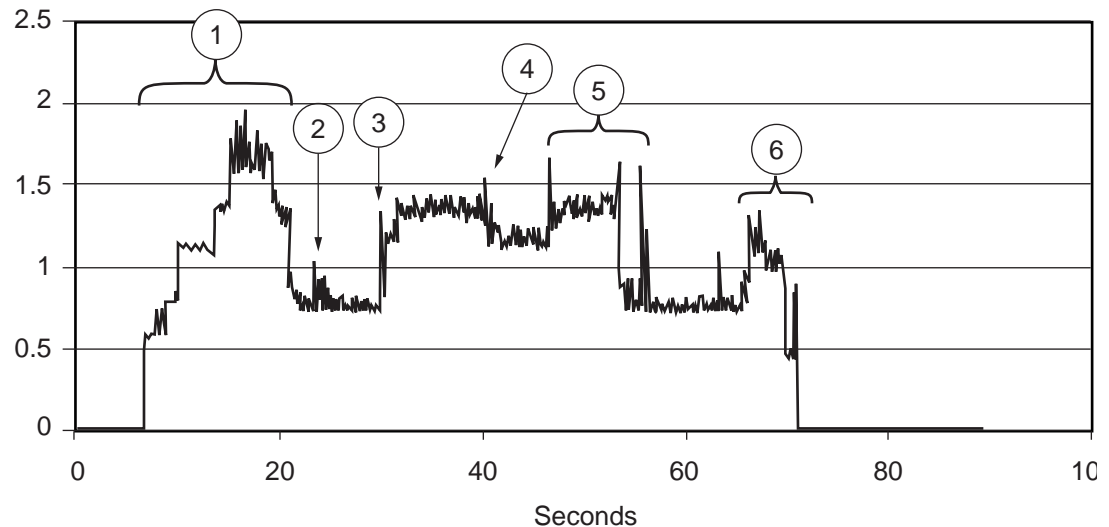
Figure 1: **Power Profile of LG Cell Phone Taking Still Photograph**

The steps in taking a still photo are summarized as:

1. Power up and turn on the LCD backlight
2. Select the camera from menu
3. Select taking a still photo
4. Take the photo
5. Return to the main menu
6. Power down

The numbers in the diagram correspond to the activity list above. The Portelligent report explains the activity of the various chips within the camera, but Figure 1 shows the composite power. There are many variables at play. Several chips within the camera receive power at different times. The display, camera chip and memories are involved here, but none of the standard cell phone capabilities - like making a call. That requires a different profile. Nonetheless, these actions are typical of a large number of today's portable systems ranging from, walkie-talkies to PDAs, to video cameras, to the emerging software defined radios (SDR). 3G cell phones, in particular, with their added on applications and multi-band operation take this type of power management to new heights. In fact, these systems may rival some of the very power conscious projects like the Mars rover or other deep space probes in complexity. Power management is vital to all of these products.

Now, let's shift our attention to Figure 2 which shows a small system using two FPGAs, an ASSP and a CoolRunner-II CPLD. Note, there are also multiple regulator chips shown (LDO), deriving their voltages by regulating down 5 volts. Another approach would be to regulate down from a battery, which most regulators can manage.

This particular example has the FPGAs being powered through the LDOs, but the ASSP simply connects to the distributed supply through a power FET. The particular FET will depend on the channel drop that can be tolerated, the turn on voltage and the power requirements of the ASSP. All power sources are controlled from logic signals coming off CoolRunner-II pins. The idea here is simple. When CoolRunner-II remains turned on, its power consumption is only in microamps (standby), whereas the FPGA current ranges into milliamps, depending on family and density.

In this example, we have combined a 150 nanometer Virtex-II FPGA and a 90 nanometer Spartan-3 FPGA. Each has different power needs, and illustrates choices and trade-offs to be made. As we will see later, the CoolRunner-II will be able to introduce an extra power down mode to the FPGAs, to reduce dynamic power. That will give a choice on whether to turn the part off, and pay for reconfiguration again, or just reduce dynamic power during standard operation. A reasonable regulator for the LDO 2 module is the TPS75003 from Texas Instruments[TM].
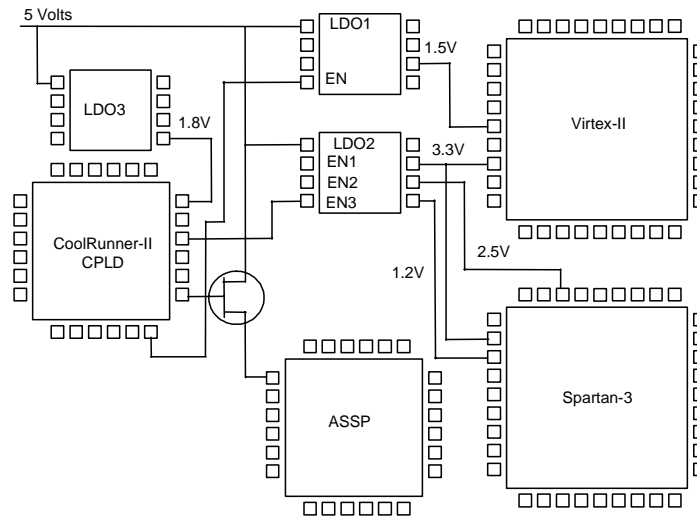

Figure 2: **Multiple Chips "Power Managed" by a CoolRunner-II CPLD**

The Virtex-II FPGA has reduced power up surge, but larger density parts may draw more current than a portable budget allows. Each FPGA might be a candidate for shutdown, except when needed. Table 1 summarizes quiescent internal current drawn by various Virtex-II family members. The dynamic current is a function of signal switching rates within the parts, and is design dependent. To estimate the power, you can use any of the methods mentioned earlier,

or simply build up the design and measure the requirement. Additional estimation resources are listed at the end. Table 2 gives similar data for Spartan-3 FPGAs.

Table 1: **Virtex-II FPGA Typical Internal Quiescent Currents**

Virtex-II Device	Typical Internal Quiescent Supply Current[1]
XC2V40	3 mA
XC2V80	5 mA
XC2V250	8 mA
XC2V500	10 mA
XC2V1000	12 mA

1. Refer to the data sheet for the most accurate and up-to-date information

Table 2: **Spartan-3/L FPGA Typical Internal Quiescent Currents**

Spartan-3/L Device	Spartan-3 Typical Internal Quiescent Supply Current[1]	Spartan-3L Typical Internal Quiescent Supply Current[1]	Spartan-3L Quiescent[1] Max (Hibernate Mode)[2]
XC3S50	10 mA		
XC3S200	20 mA		
XC3S400	35 mA		
XC3S1000/L	65 mA	30 mA	6 mA
XC3S1500/L	65 mA	50 mA	8 mA

1. Refer to the data sheet for the most accurate and up-to-date information
2. Hibernate Mode available for Spartan-3L. **See DS313**.

Figure 3 shows how a current profile might look (averaged) over time. Usually, there is an initial surge as all on board capacitance becomes charged, various parts undergo configuration, initialization, bootstrapping, and so forth. Then things settle down, and various chips can be turned off, placed into low power, or whatever, as dictated by the application. The average current draw of this profile is substantially less than the initial surge value, so there may be a

payoff for turning chips off all together, or placing them in a low power mode. Table 3 and Table 4 give configuration times in byte wide mode at maximum speed.

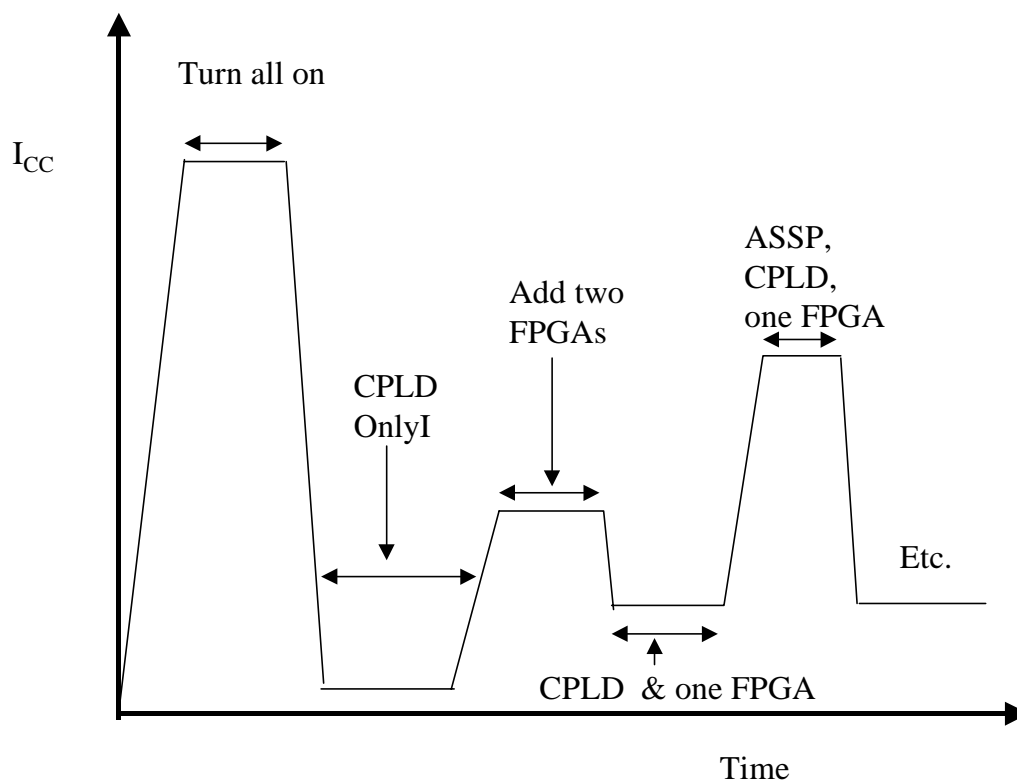*Figure 3:* **Current Profile of a System with Various Active Devices Over Time**

Table 3: **Virtex-II Configuration Parameters**

Virtex-II Device	Configuration Bits	Configuration Time in Microseconds (at 50 MHz)
XC2V40	338,976	42.372
XC2V80	598,816	74.852
XC2V250	1,593,632	199.204
XC2V500	2,560,544	320.068
XC2V1000	4,082,592	510.324

Table 4: **Spartan-3 Configuration Parameters**

Spartan-3 Device	Configuration Bits	Configuration Time in Microseconds (at 50 MHz)
XC3S50	439,264	54.9
XC3S200	1,047,616	130.95
XC3S400	1,699,136	212.392
XC3S1000	3,223,488	402.936
XC3S1500	5,214,784	651.848

By lowering the average power, CoolRunner-II CPLDs can dramatically extend the battery life of a system in a way that brings the high flexibility and value of FPGAs into the portable world. Let's show how more value is gained using CoolRunner-II DataGATE.

DataGATE

DataGATE was designed to stop unwanted input switching from continuously draining power in CoolRunner-II CPLDs. Additional applications evolved from testing to security, and are documented in the Advanced Features and DataGATE application notes. However, one additional application is simply to "DataGATE" other chips.

Figure 4 shows how the DataGATE feature works. A metal rail (DataGATE Assertion Rail) circles the whole chip inside, near the pins. Each input site provides a place where the received signal can be blocked by a pass transistor, depending on two conditions. The first condition is an enable bit, selecting that pin to participate in the DataGATE decision. The second condition is simply whether the DataGATE Rail is asserted. If the rail is asserted and that input's participation selected, the input signal is blocked from penetrating the chip, until the rail releases assertion. It's that simple. When the rail asserts, blocking follows immediately. The previous input level automatically latches, so static CMOS logic signals forward into the CPLD core. The signal freezes until released. When the rail releases, switching action resumes.
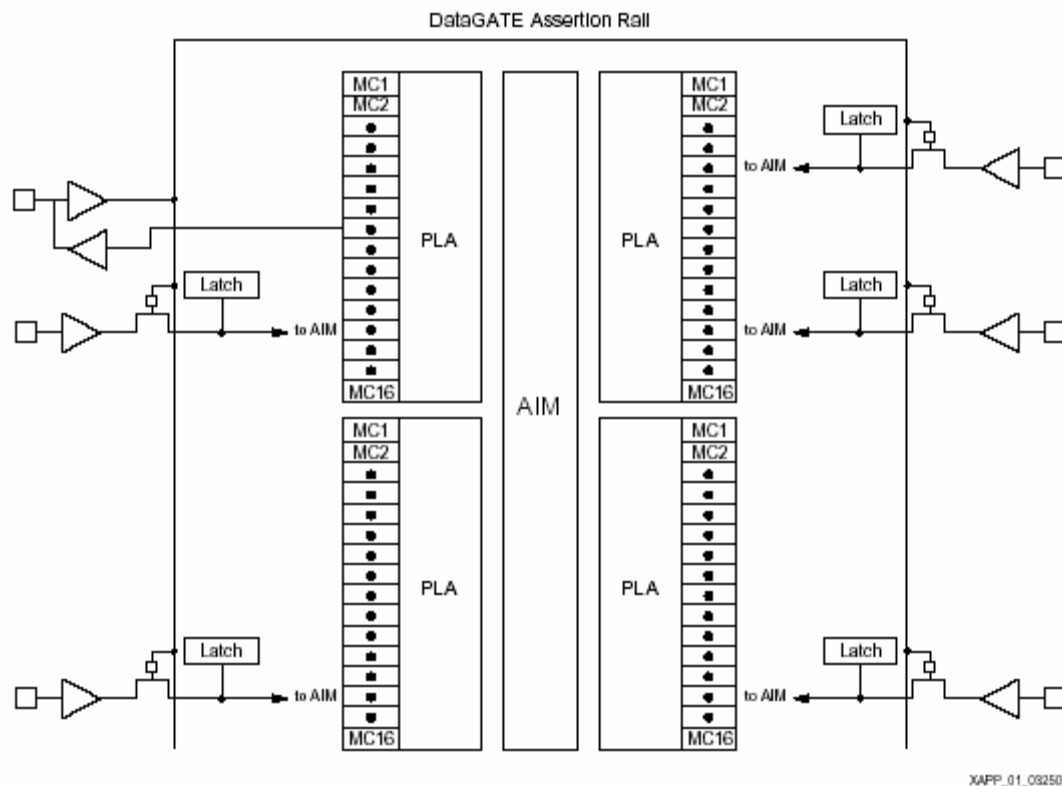
Figure 4: **DataGATE Architecture**

Figure 5 shows a close-up of how the pass transistor, enable cell and latch all connect to automatically block and freeze input signals that forward through the CoolRunner-II core.
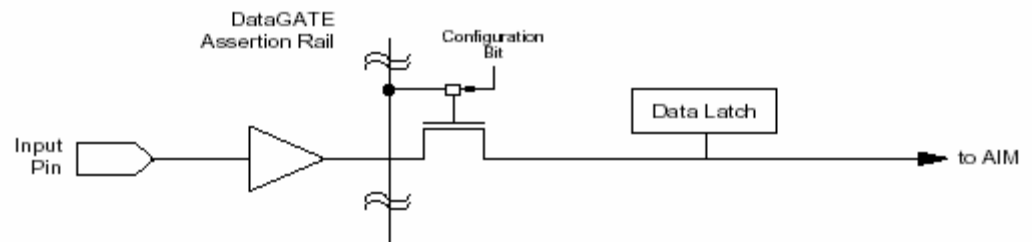

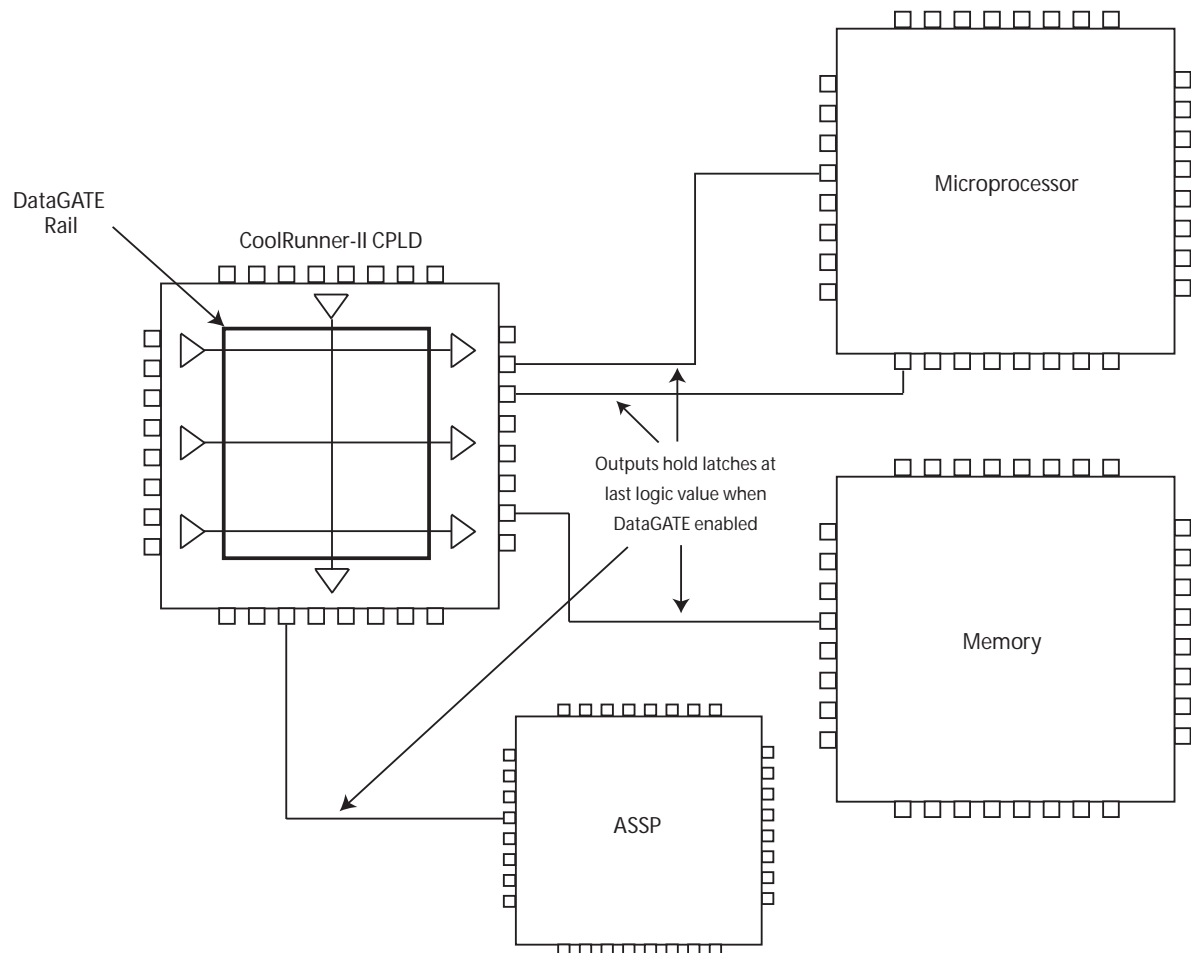*Figure 5:* **Close-up of DataGATE Switch Mechanism**

Figure 6: **DataGATE Blocking Switching Activity to Other Chips**

Figure 6 shows how signals passing through the CoolRunner-II DataGATE freeze signals to other chips. In this situation, we passed signals through the CoolRunner-II part, directly to the outside, where they drive to other chips. These will be held at the logic level that was last on the input pad, when DataGATE asserts. Should a 3-state signal be forwarded through the CoolRunner-II output pins, it will naturally be pulled to a high or low value, by the weak keeper that is on them, thereby covering the case when a "frozen" output gets 3-stated by another condition. An interesting proposition of passing signals through the CoolRunner-II is that it might be done as a natural side product of simply translating voltage levels on signals as they

 www.xilinx.com

interface through the CPLD. The DataGATE action, is really for free under those conditions, which occur frequently.

DataGATE is offered on CoolRunner-II CPLDS that have 128, 256, 384 and 512 macrocells. This extends into the hundreds of I/O pins that can be blocked as needed by the DataGATE facility. If multiple sets of pins need to be blocked, under different circumstances, then multiples of the smaller parts can be used to cover the number and condition needs of any given design.

At this point, we have not described what drives the DataGATE rail. It is very simple. One macrocell within a particular CoolRunner-II CPLD is designated as the "DataGATE" macrocell. It is identical to all other macrocells, which means any logic situation a designer sets up to drive that macrocell, asserts the DataGATE signal, if enabled in the design software. The DataGATE signal releases whenever the logic driving that macrocell dictates. An event as simple as a switching input can trigger the DataGATE macrocell, and an event as complex as a conditional state machine driving a timer can trigger the macrocell. Designers are free to dream up whatever they want. It is possible to block any input chosen, including clocks, but extreme care must be used when designing that way! All, none or any subset of the input pins can be blocked. Being a reprogrammable CPLD, this facility can be used experimentally to determine the best set of signals to "freeze" and the best set of circumstances to assert and release the rail.

So, how do you know what to "freeze?" That will vary, from system to system. Here's an example. When a microprocessor first bootstraps, it frequently sends values from its databus into a CPLD. These might be address values being loaded into comparators to select ranges of memory and I/O devices. Once those registers are initialized they need only compare against the address lines to operate. The databus connections are never needed again, but are still attached. DataGATE lets designers identify the time when the connection is no longer needed, and eliminate the extraneous switching that draws unneeded current. To learn more about other things you can do with DataGATE, check the references at the end.

Conclusion

We have omitted some details. What is the power impact of unpowered I/O pins attached to powered up termination resistors? How much leakage per pin occurs if an unpowered pin is driven by a powered one? Many questions can only be answered by assessing the specific situations with the particular device's data sheet in hand. We hope the methods described here will have some value by simply increasing your choices on power reduction methods.

CoolRunner-II CPLDs are designed to be inherently low power devices. Additional features within them – including DataGATE, can help other chips also reduce their overall power, when properly applied.

References

Portelligent Report #116.02-031023-1d

Estimation equation: http://www.xilinx.com/bvdocs/appnotes/xapp317.pdf

Low power design methods: http://www.xilinx.com/bvdocs/appnotes/xapp346.pdf

Decreasing power: http://www.xilinx.com/bvdocs/appnotes/xapp347.pdf

Accurate XPLA3 estimation: http://www.xilinx.com/bvdocs/appnotes/xapp360.pdf

Accurate CoolRunner-II Design estimation: http://www.xilinx.com/bvdocs/appnotes/xapp377.pdf

Powering CoolRunner-II: http://www.xilinx.com/bvdocs/appnotes/xapp389.pdf

DataGATE: http://www.xilinx.com/bvdocs/appnotes/xapp395.pdf

XPower http://www.xilinx.com/xlnx/xebiz/designResources/

Power Estimator (Spartan) http://www.xilinx.com/cgi-bin/power_tool/power_Spartan3

Power Estimator (Virtex-II) http://www.xilinx.com/cgi-bin/power_tool/power_Virtex2

Additional Information

CoolRunner-II Data Sheets, Application Notes, and White Papers
Device Packages

Spartan-3 Data Sheets, Application Notes, and White Papers

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/29/04	1.0	Initial Xilinx release.
03/30/05	1.1	Removed incorrect reference to power surge. Added Spartan-3L.
09/28/05	1.2	Corrected part number on TI regulator, page 3.

www.xilinx.com XAPP436 (v1.2) September 28, 2005