

How to Tame the Power Beast in Consumer Handheld MPUs

Using an FPGA is a popular way to expand the capabilities of an embedded system. Designers can reduce power consumption at the same time.

by Rahul V. Shah
Director of Customer Solutions
eInfochips
RahulV.Shah@eInfochips.com

Vishesh Agrawal
ASIC Design and Verification Engineer
eInfochips
Vishesh.Agrawal@eInfochips.com

The consumer handheld market is growing by leaps and bounds. With more processing power and increased support for more applications, portable products are cross-pollinating with traditional computing systems even as the product life cycle has decreased considerably in this market segment. As a result, especially in this era of economic slowdown, it is imperative that new products meet the time-to-market window to gain maximum acceptance. A decrease in product life cycles requires a reduced development cycle and an increased emphasis on reusability and reprogrammability.

The emerging handheld market is also seeing interesting trends in which each individual device in a family has lower volumes but there is more customization across the series of devices, effectively upping the total unit volumes. The key challenge then becomes how to develop a system that is widely reusable and also customizable.

These requirements have led designers increasingly to turn to the FPGA for handheld-product development. The FPGA has become more powerful and feature-rich, while gate counts, area and frequency have increased. FPGA development and turn-around cycles are considerably shorter than those of custom ASICs, and the added advantage of reprogrammability can make the FPGA a more compelling solution for handheld embedded systems.

In an ASIC- or an FPGA-based design, designers must take certain performance criteria into account. The challenges can be stated in terms of area, speed and power.

As with the ASIC, vendors are taking care of the area and speed challenges in FPGA design. With increased gate counts, the FPGA has more area and size to accommo-

date larger applications, and tools include better algorithms to utilize the area optimally. For example, technology advancements with newer standard-cell libraries have led to FPGAs achieving higher frequencies.

The newer and better FPGA technology brings with it a whole new set of challenges for the designer. Power utilization is one issue that moves to the forefront when designing an FPGA-based embedded system for a handheld or portable device.

FPGA in an Embedded System

A typical embedded system consists of a processor, memory, standard interfaces like USB, SPI, I²C and so on, along with peripherals such as liquid-crystal display, audio-out and the like. The heart of the application still resides in the processor and the proces-

sor interfaces, with various peripherals using onboard connections. The performance of the system depends on the performance of the processor, which has a very standard architecture and is not easy to customize.

The processor may also end up utilizing its activity time in processing information from a slower peripheral (for example, reading data from a slower ADC audio on an I2S interface). Although the processor usage may reach 100 percent in this case, the device is not doing microprocessor-centric activities but is working at a significantly lower performance level. Irrespective of its core frequency, the microprocessor must wait for the data from a slower clock. This also results in higher power consumption, as the processor is showing full utilization. The result is lower battery life along with larger

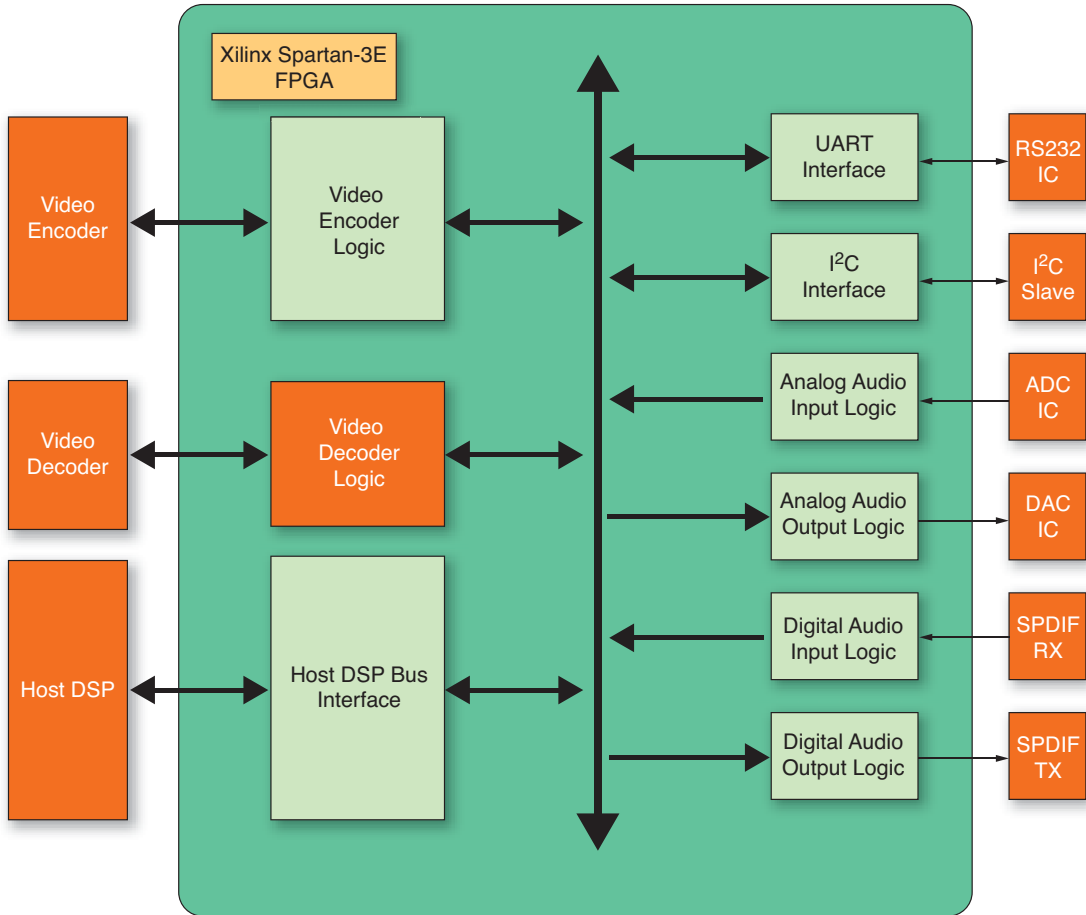


Figure 1 – FPGA architecture for audio/video distribution system

One technique to minimize static power consumption is to perform power estimations early in the design cycle and, if feasible, change the topology or use a different IP block. Xilinx's xPower Estimator tool signals very early in the design process whether you can meet the power budget.

heat sinks or fans for cooling, eventually affecting the reliability of the entire system.

FPGAs have started to play an important role on this front, since they can offload the processor of many of its peripheral interaction duties. For example, the embedded distribution system of an uncompressed audio/video stream through a standard Gigabit TCP/IP network, shown in Figure 1, has a dedicated DSP processor interfacing with a Xilinx® FPGA over a standard bus interface. The FPGA is connected to various slower peripheral devices.

For starters, it is interfacing with a 12-bit PCM audio input and a 12-bit PCM audio output, both over I2S. It also interfaces with a video encoder and decoder and communicates with I²C slaves and RS232 devices. There are few general-purpose I/Os connected to the FPGA. The standard bus operates at a high speed of 66 MHz, while the audio peripherals work at a low speed of 1.182 MHz. Serial interfaces like UART and I²C operate at 56.6 kHz and 100 kHz respectively. The data transfer takes place over multiple clock domains with only the processor configuring the data flow.

In this case, instead of the processor interacting with the slower peripheral device, the FPGA can read data from the slower PCM ADC audio and store it in its internal buffer. Either the processor can read that buffer periodically or the FPGA can send an interrupt to the processor when it has sufficient data. The processor now has more idle time in which to do processor-centric work if required; otherwise it can go into sleep mode during idle time.

Power Consumption

In a battery-operated embedded system, power saving is of the utmost importance.

Power consumption can be categorized into three areas: startup power, static power and dynamic power.

The designer cannot control startup power consumption, which plays an important role in deciding on the power supply selection. Most of the maximum current level drawn is achieved in this phase.

But static and dynamic power consumption are two areas where, with proper planning and by following correct guidelines, the embedded designer working with FPGAs can make a marked improvement in power optimization.

Static power is the flow of current through a component even when there is no activity in the system, generally due to device bias and leakage. Static power also depends upon operating voltage. Reducing the operating voltage will reduce the static power. But this decision is not always in the hands of the designer.

What the designer can do is to define the architecture in such a way that requires the least amount of resources, to use resource sharing whenever possible and to use the FPGA blocks in the most efficient manner.

Another technique to minimize static power consumption is to perform power estimations early in the design cycle and—if required and feasible—change the topology or use a different IP block. Xilinx's xPower Estimator tool is useful in knowing very early in the design whether you can meet the power budget. Early-stage power estimation may not be totally accurate, but it does serve as a guiding tool.

Dynamic Power Consumption

Dynamic power consumption is the result of some activity on the FPGA gate—namely, signal switching—when both gates are switched on briefly, causing current flow and capacitance. The speed of the signal

switching will determine how much power is consumed. Another factor that determines dynamic power consumption is the inherent capacitance created within the geometry of the circuitry.

Dynamic power is a function of the clock rate, the number of gates that are switching and the switching rate of these gates. The capacitive load on the gate fan-out and wire add to the dynamic power consumption, which is proportional to the product of capacitance, voltage and square of frequency.

The designer has maximum control over this type of power consumption, with access to many techniques that will reap the maximum benefit when it comes to dynamic power.

Reducing the signal switching frequency slashes power consumption exponentially. As in the reference design shown in Figure 1, the control logic for the UART, parity check or frame overrun error occurs in the slower clock domain. Even though there is no reduction in the gate count, the power consumption falls. Designers can also reduce dynamic power consumption by lowering the overall operating frequency, if feasible. After doing the feasibility and performance analysis, for example, the designers decided that instead of working at 133 MHz, the above design also works at 66 MHz. The DSP supported both those speeds, and reducing the voltage also helped in dissipating less power.

Another technique is to reduce the number of active gates in an operating mode. Sometimes a part of the logic, though switched on and configured at power-up, is not required to actually do anything. If, for example, the analog audio capture unit is active, the application is not performing any activity on digital SPDIF audio capture. In this case, the typical digital SPDIF audio capture still performs data

sampling, biphase decoding and so forth, which is an unnecessary power drain. If the entire digital SPDIF audio capture circuit is disabled so that no signal switching takes place in the circuit at all, the result will be a big reduction in dynamic power.

You can achieve this by disabling the clock that propagates to that portion of the design. A simple way of doing so is to AND the clock signal with an enable signal, shown in Figure 2. If the enable signal is low, the output of the AND gate will stay low. If the enable is high, the output of the AND gate follows the clock.

Other methods may also be applicable. If possible and supported by topology, multiplexing address and data lines can reduce the number of signaling lines. In our example, the output to the video encoder was 16-bit data, which we multi-

can adopt, with manifold benefits. First, it reduces the challenge of customizing the processor, as discussed earlier. Second, the interaction between the peripherals and the processor resides inside the FPGA and reduces the I/O count. Since I/Os use considerable power, this also results in some amount of power saving. Xilinx's Virtex®-5 edition supports PowerPC® 440 processors, hard processors and MicroBlaze™ soft processors, all of which designers can leverage to make any system with high-end or low-end applications in mind.

Xilinx has been at the forefront in this field and its latest FPGAs offer many power optimization features. The company's power analysis and power estimation tools, xPower Estimator and xPower Analyzer, have many features that help the designer in crafting a low-power FPGA system.

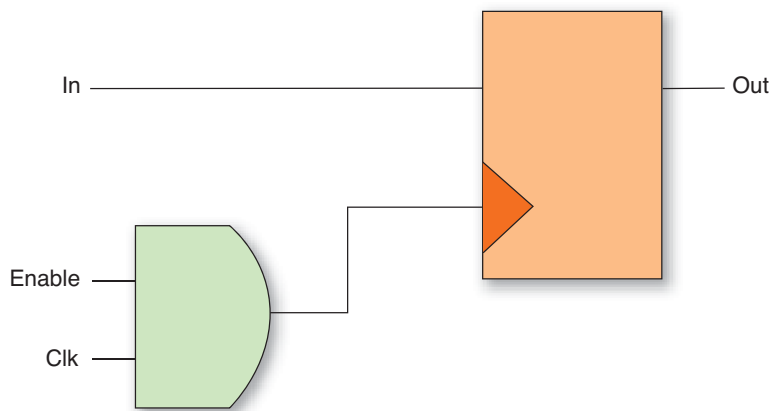


Figure 2 — A simple clock-gating mechanism

plexed over 8-bit and sent over both the edges of the clock. This also resulted in saving dynamic power. Choosing a serial rather than a parallel interface will also lower the dynamic power. Using LVTTTL or LVCMOS I/O with a lower capacitive load helps too, although the designer may not always be the one to decide on the I/O.

Embedded Processors

Embedding the processor inside the FPGA is another strategy that handheld designers

With the advent of 90- and 65-nanometer semiconductor technologies, the gates are shrinking in size, resulting in manifestation of static power consumption—a challenging phenomenon given the growing sensitivity to power metrics. There is a lot of excitement in this field as power issues gain prominence with various FPGA vendors. Low-power design will determine how much more we can pack into one system. There is a dire need to standardize a design methodology that focuses on power consumption. ●●

GET ON TARGET



IS YOUR MARKETING MESSAGE REACHING THE RIGHT PEOPLE?

Hit your target by advertising your product or service in the Xilinx *Xcell Journal*, you'll reach thousands of qualified engineers, designers, and engineering managers worldwide.

The Xilinx *Xcell Journal* is an award-winning publication, dedicated specifically to helping programmable logic users – and it works.

We offer affordable advertising rates and a variety of advertisement sizes to meet any budget!

Call today:
(800) 493-5551
or e-mail us at
xcelladsales@aol.com

