How to cut SoC power dissipation, energy consumption and cost

Using multiple configurable processors can cut SoC power dissipation, energy consumption, and cost, says **Steve Leibson**

Rising clock rate causes dynamic power dissipation and energy consumption to grow, resulting in unacceptable energy consumption and heat in contemporary SoCs. These negative consequences create some hard limits that are halting SoC advancement.

One key way to reverse this trend is to exploit inherent parallelism, which cuts the need for high clock rates. Programmable, configurable microprocessor cores play a significant role in exploiting parallelism, in ways that cannot be achieved using conventional, fixed-ISA (instruction set architecture) processors.

Dynamic power dissipation and energy consumption clearly rise with clock frequency.

The formula is: $P(\text{dynamic}) = \frac{1}{2} CV2f$

In this formula, the C is the total switched capacitance of all on-chip nodes, V is the core operating voltage, and f is the clock frequency. It appears that the dynamic power is linearly proportional to clock frequency but SoCs must run at higher core operating voltages to attain the highest possible clock frequencies,, which also brings the V2 term into play, so the relationship between dynamic power and operating frequency is superlinear.

Battery life is inversely proportional to energy consumption. SOCs that draw a lot of energy result in products with short talk or operating times and unacceptable standby times.

At the same time, larger and more expensive batteries and power supplies are needed to power SoCs that consume a lot of energy. Such SoCs run hotter, so they need more expensive packaging (for copper heat spreaders or ceramic packaging). Larger, more costly heat sinks or noisy fans are needed to cool these SoCs and systems based on such SoCs will therefore require larger, more expensive enclosures.

64-Point, Complex 16-Bit, Decimation-In-Frequency FFT	One FFT Every 3.2 uSec		Estimated Energy (uJ)
	Required Cycle Count	Required Clock Rate	
Straight C	32187	10 GHz	3,450
Add 32-bit Multiplier	5071	1.6 GHz	575
Multiple Instruction Issue and 32-Bit Multiplier	2975	930 MHz	620
Radix-4 FFT ISA Extension	146	46 MHz	56

Figure 1: Cycle Count and Energy Reduction for Radix-4 FFT.

The added heat also reduces product reliability, which in turn increases warranty costs.

The place to start reducing system energy consumption is at the system level and the key to such cuts in digital systems is reducing clock rates. Lower clock rates immediately cut dynamic energy consumption and they reduce the need for advanced IC processes, thus reversing the concurrent upward trend in static energy consumption.

One way to cut clock rates is to increase execution parallelism. Hardware designers intuitively use parallelism but microprocessor-based designs have done the opposite for decades. In the name of saving hardware, microprocessor-based systems employ multitasking operating systems that allow the processors to execute multiple concurrent tasks. The processor's clock rate therefore becomes the aggregate clock rate needed to execute all of these multiple tasks.

When energy consumption was not a problem, when microprocessors came in individual packages, and when tasks were fairly simple, it did make sense to cut hardware costs by employing multitasking. However, processors on 90nm and 65nm SoCs consume less than 1mm sq. of silicon. To avoid high clock rates and unnecessary energy consumption, it now makes sense to use more processors running at lower clock rates through reduced multitasking.

It also makes sense to tailor each processor to the assigned task using the ISA-extension abilities inherent in configurable processors. Such tailoring makes each on-chip processor more efficient when executing a specific algorithm. Increased efficiency permits the processor to execute the algorithm during the same amount of time but with fewer instructions.

The Fast Fourier Transform (FFT) serves as a good example of an algorithm that can be greatly accelerated through ISA tailoring. The FFT decomposes signals into their constituent frequency components and is commonly used in communications and signaling applications. For example, the 802.11g wireless PHY employs 64-point, radix-4, decimation-in-frequency FFTs, executed every 3.2 microseconds or so.

Radix-4 FFTs are usually implemented with hardware butterfly blocks (the butterfly is the basic FFT operation), which require twelve 16x16-bit multipliers and more than 20 16-bit adders. Usually, it's impractical to implement an FFT in software, as illustrated by Figure 1, because of the high clock rates required. Using straight C code, a 32-bit RISC processor (with no hardware multiplier) needs 32187 cycles to execute the FFT algorithm. To execute one of these FFTs every 3.2 microseconds, the processor would need to run at 10GHz. That's clearly an impossible clock rate for any processor, much less one synthesized on an SoC.

Figure 1 shows the result of increasing amounts of ISA tailoring for the FFT. Adding one 32-bit multiplier to the Risc processor cuts the required clock rate to 1.6GHz. Although better, that number is still out of reach for synthesized processors. Creating a superscalar microprocessor that can issue multiple Risc instructions simultaneously and adding the hardware multiplier only drops the required clock rate to 930MHz, at the very edge of possibility for a synthesized processor.

However, adding a radix-4 butterfly instruction to a processor cuts the required clock rate to 46 MHz (a x217 reduction) and energy consumption by x62. Almost any algorithm can be similarly addressed.

Steve Leibson from Tensilica