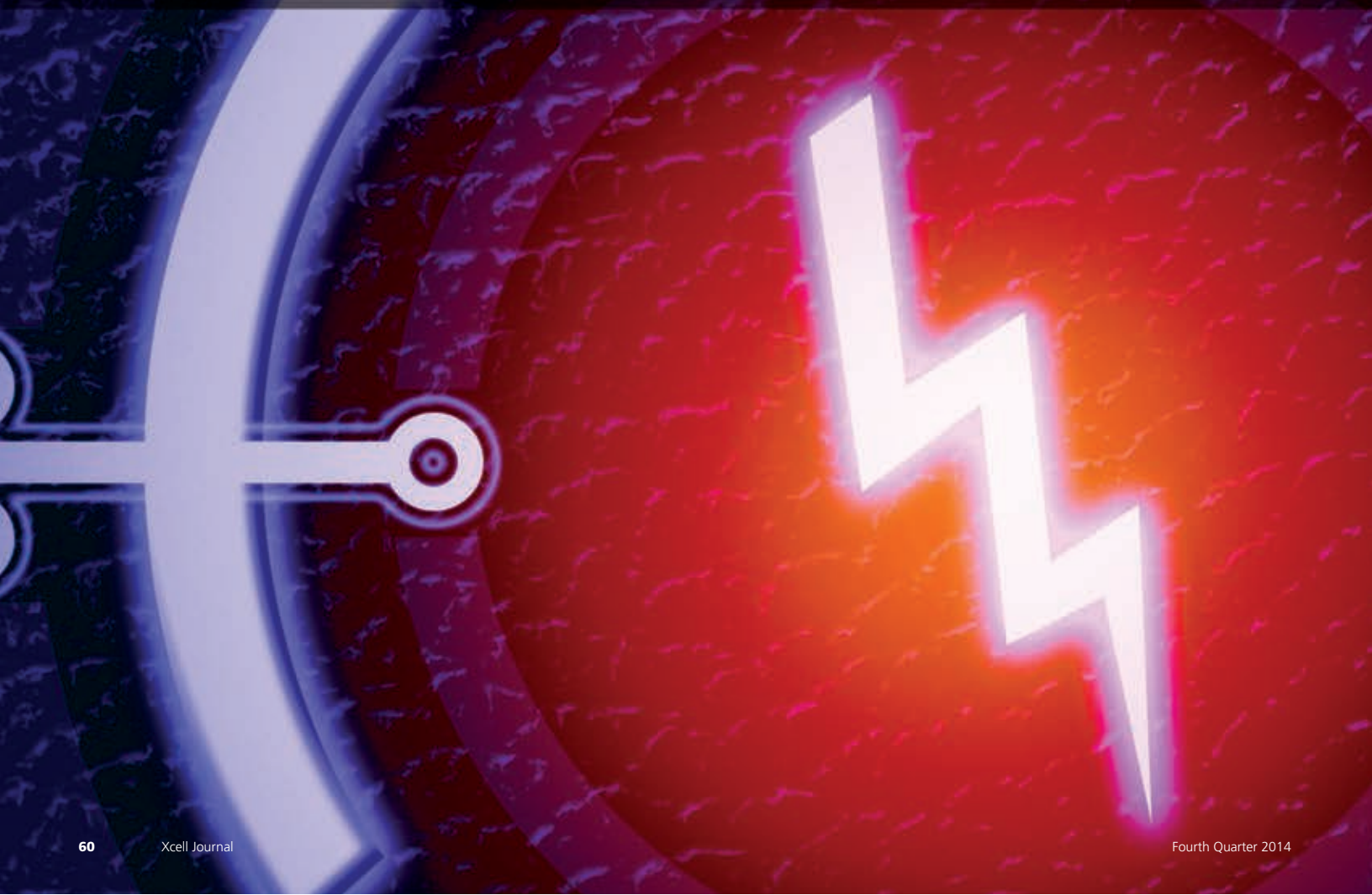
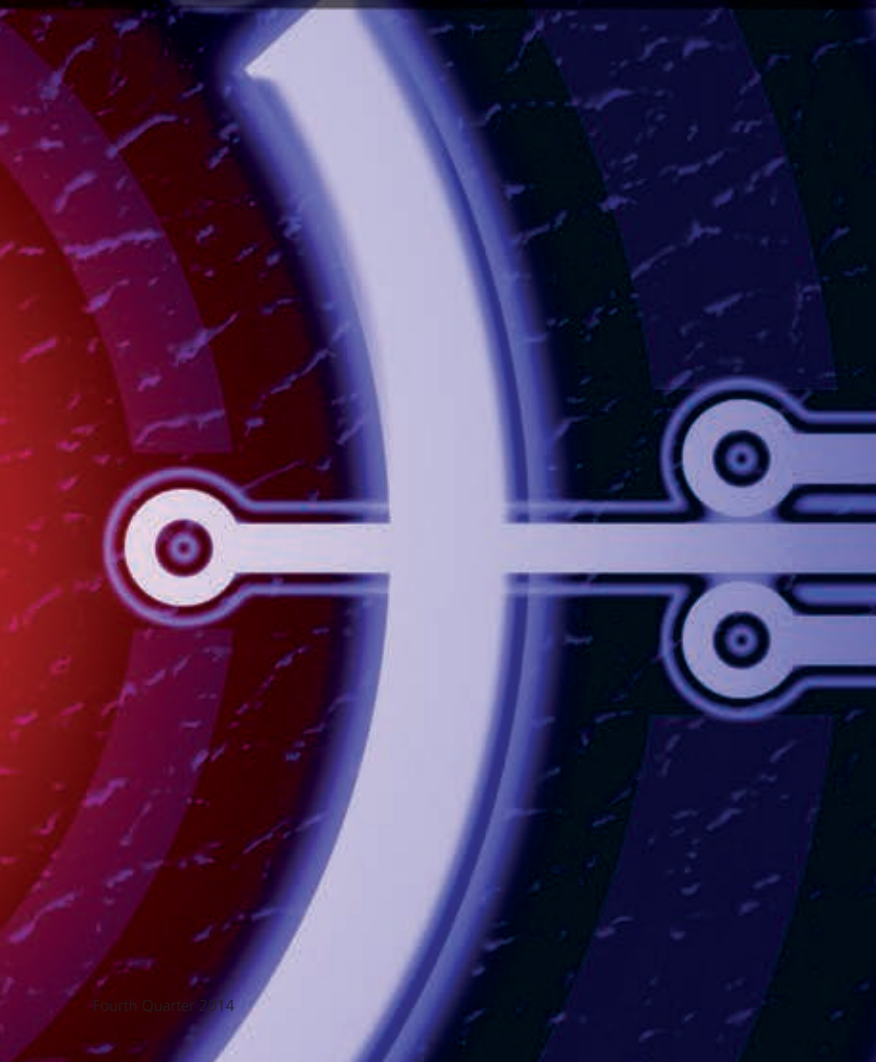
Changing Utilization Rates in Real Time to Investigate FPGA Power Behavior



Power management in FPGA designs is always a critical issue. Here's a new way to measure estimated power on a real FPGA device.

by Ahmet Caner Yüzügüler
Hardware Design Engineer
Aselsan
acyuzuguler@aselsan.com.tr

and Emre Sahin
Senior Hardware Design Engineer
Aselsan
emresahin@aselsan.com.tr



Modern FPGA chips are very capable of developing high-performance applications, but power management in these designs is usually a limiting factor. The resource usage of an FPGA device mostly determines the capacity and processing rate of the design, but adding resources increases the power consumption. Higher power dissipation brings more operating costs, larger area requirements and higher junction temperature, which the designer must address with more airflow and cooling systems.

Since the total power consumption of a board or system is so important, designers must set a power budget, juggling trade-offs between resource usage and power concerns. The ability to predict the amount of potential power consumption of the system beforehand therefore gives the designer a head start.

For preimplementation power evaluation, Xilinx offers some tools that estimate the power consumption virtually based on user entries or synthesis reports. One of them is the Xilinx® Power Estimator (XPE) spreadsheet. This Excel-based power-estimation tool lets you enter design properties such as resource usage, toggle rates and clock frequency as input and calculates the estimated power according to the device information. Another commonly used tool is Xilinx Power Analyzer (XPA). After placement and routing, XPA imports the generated NCD file and estimates the power with implementation details and simulation results instead of user inputs for more accurate estimations.

As an alternative, we have devised a new method to measure the estimated power of an FPGA design on a real device. In order to imitate different implementation scenarios, we have created a generic, device-free VHDL design within which it is possible to change the number of activated resources—namely, DSP slices, Block RAMs and slice registers—along with their operational conditions (junction temperature, clock frequency and toggle rates) through a serial channel while the FPGA is running. Our technique monitors

the current and voltage levels of the supply rail simultaneously so that we can easily observe the dynamic power characteristics of a device for different resource-usage and ambient conditions.

We have implemented the design on a Xilinx KC702 Evaluation Board. But you can implement it on any other device as well, with a few minor changes, so long as the FPGA device supports the IP cores used in the design.

Another possible way to use our technique might be as a testing VHDL design for FPGA boards. Let us imagine we have recently designed a general-purpose Kintex®-7 board. The customer may implement any design on the board with an unknown amount of resource capacity used in changing ambient conditions. To ensure the stability and robustness of the board, we need to force the operating limits of the device in the highest and lowest required ambient temperatures and verify that the FPGA can work for different resource-usage cases. However, testing every resource-utilization scenario would require a new VHDL design from scratch and take too much time. Our suggested method gives designers the flexibility to manipulate the test design as required in real time.

IMPLEMENTATION DETAILS

We have tried to keep our implementation as simple as possible to prevent generating slice LUTs used as logic, which consume power that we cannot control. The simplest way to implement slice registers is to combine them as a shift register block. Figure 1a is a block diagram of how slice registers are im-

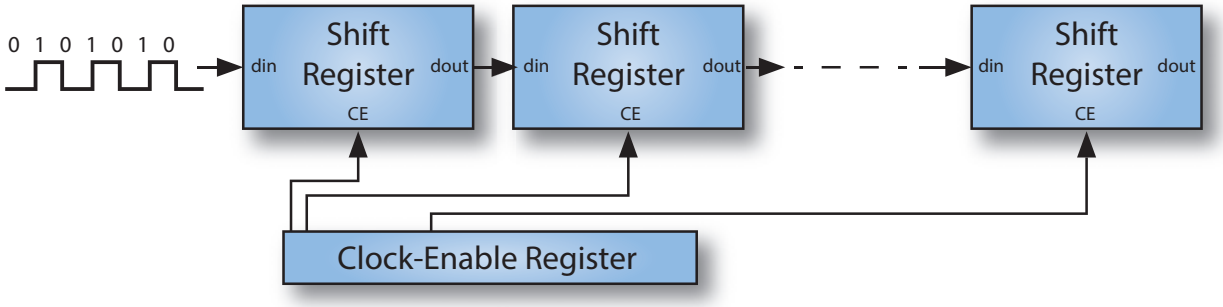


Figure 1a

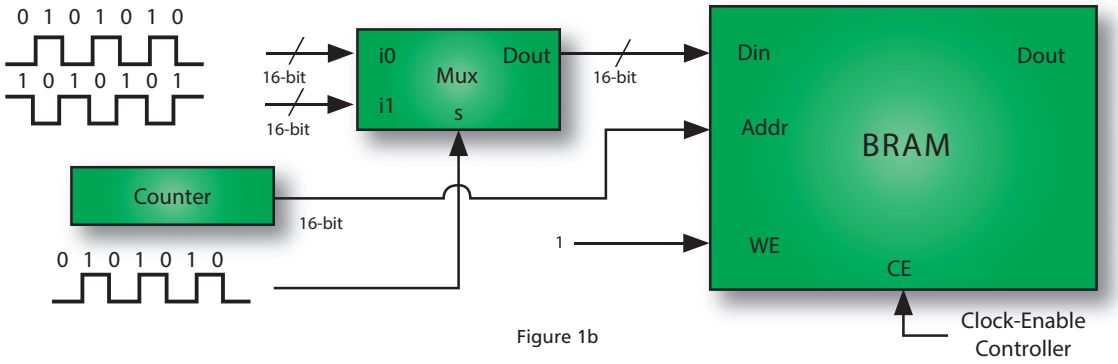


Figure 1b

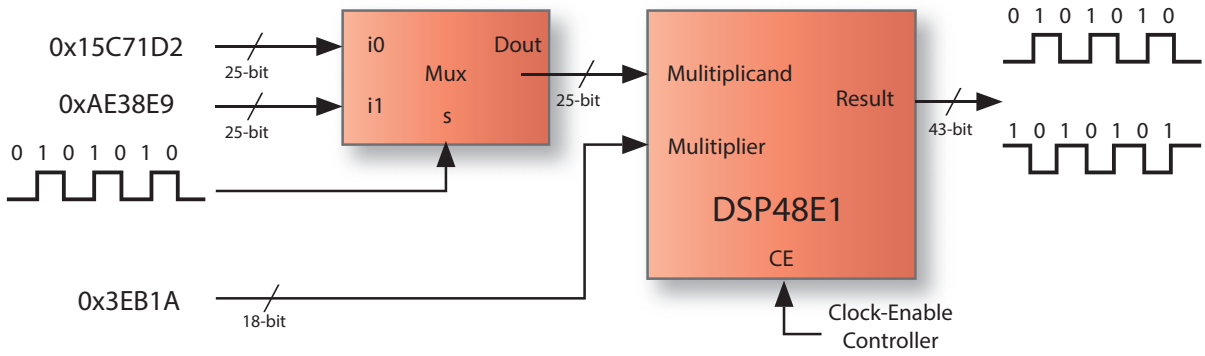


Figure 1c

Figure 1 — Implementation block diagrams: (a) slice registers, (b) Block RAMs and (c) DSP slices

plemented. One simple point to note here is that, when we attempt to create slice registers, synthesis tools tend to use slice LUTs as 32-bit shift registers (SRL32) instead of slice registers. You can use the following VHDL attribute to force the synthesis software to utilize slice registers instead:

```
attribute SHREG_EXTRACT:     string;
attribute SHREG_EXTRACT of <signal_name> : signal is "no";
```

The LogiCORE™ Block Memory Generator creates Block RAMs as single-port RAMs. A 16-bit word, whose bits are continuously toggled, is constantly written into a random address of the activated BRAMs to keep them occupied. Figure 1b is a block diagram of one BRAM component. Similarly, we use DSP slices to multiply a 25-bit multiplicand and an 18-bit multiplier that produces a 43-bit output, which is the maximum word width a single DSP48E1 can handle. The multiplicand of all DSP components is regularly altered to make the DSPs dissipate dynamic power, as shown in Figure 1c. However, synthesis tools attempt to remove the resources in the synthesis process, since the outputs of these modules are not connected to any output pin. You can use the following attribute to keep the resources untrimmed:

```
attribute KEEP : string;
attribute KEEP of <signal_name>: signal is "TRUE";
```

In this scheme, you activate resources by controlling their clock-enable signals. Clock-disabled resources consume so little power that they can be considered as not utilized in the design at all. More specifically, power consumption of 50 DSP slices with a 100 percent toggle rate is 0.112 watts, whereas it is 0.001 W when their clock-enable signals are deasserted. The result is to obtain approximately the same outcome for different utilization rates. As a consequence, a design with each resource component controlled with a clock-enable signal over serial communication simulates changing utilization rates instantaneously without the need for new design and implementation stages.

One important factor that affects power consumption considerably is the average toggle rate of the resources. The toggle rate is the number of transitions of the output for a specific resource per clock cycle. You can achieve a constant 100 percent toggle rate by alternating inputs of the resources that make the output change its state in every cycle. For a DSP slice, for example, you would adjust the toggle rate by choosing two multiplicands, which are alternated consecutively, and one multiplier, so that each bit of output changes in each cycle. The alternation rate of inputs determines the toggle rate of a resource; thus, we control the toggle rate of a resource type on the fly via serial channel.

Power consumption is also directly proportional to the clock frequency. We have utilized a Mixed-Mode Clock Manager (MMCM) to generate clocks with varying frequencies. A set of registers determines the frequency, phase and duty cycle of the MMCM output, which would normally only be initialized when the design is implemented and a bitstream file is generated. However, the MMCM's Dynamic Reconfiguration Port allows us to change such characteristics as the output clock while the FPGA is running. Clock phase and duty cycle are not of interest, since phase does not affect power consumption and duty cycle does not vary in most cases.

Frequency, on the other hand, is highly correlated with power dissipation. The internal mechanism of an MMCM works in such a way that VCO frequency is divided by the value of the CLKOUT0_DIVIDE register to obtain the desired output frequency. We assign new values to this register by following a state machine algorithm, as explained in Xilinx application note XAPP888, for a glitch-free transition. At the same time, we change the clock frequency of the design in real time according to the user input in the graphical user interface (GUI). Then, we can easily observe the power characteristics of the design for different frequencies.

Power behavior under high-temperature conditions is another important issue to observe and verify cautiously. Core temperature of the silicon depends on the design of the board, processing rate, ambient temperature, heat sink and airflow provided by the fan. In our design, we partially control the junction temperature by changing the speed of the fan located on top of the FPGA chip with a simple PWM on-off controller. You can also use an external heating tool such as a heat gun to reduce heating time.

On-chip sensors measure the core temperature and an analog-to-digital converter, XADC, is generated through the LogiCORE XADC Wizard. When you enter a reference temperature value in the GUI, it transmits to the device through the serial channel and compares it to the core temperature measured with on-chip sensors. The speed of the fan is controlled to stabilize the junction temperature at the desired level. In this way, you can observe power consumption and plot it over core temperature to confirm that it meets the power budget of your design and stays within critical limits for the required heating profile.

A graphical user interface is designed for communication with the FPGA device to easily change the parameters explained above. Figure 2 shows a screen shot of the GUI. We first connect to the device via a standard UART with one of the available ports. Then, we can open Xilinx Power Estimator to compare its estimated values with the measured ones. The GUI starts to plot a power vs. time

graph immediately for both estimated and actual power with the default values of the parameters. You can modify those parameters on the related panels and specify the number of utilized slice resources with their toggle rates at the Resource panel. The Utilization Sweep panel allows you to sweep the utilization rate of a particular resource from 0 to 100 percent with equal intervals and plot the power vs. utilization graph with the measured power values for each interval.

Similarly, you can change the output of the MMCM and sweep the clock frequency on the Clock Select panel. In the next column, the supply voltage of the FPGA chip is measured and plotted continuously on the Vccint panel. The Core Temperature panel, meanwhile, shows and plots core temperature. The user can enter a reference temperature to change junction temperature in the provided box.

Voltage and current levels of the power supply are measured with Texas Instruments' UCD9248 Digital PWM System Controller, integrated on the KC702 board. This multirail and multiphase PWM controller for power converters supports the Power Management Bus (PMBus) communication protocol. Its PWM signal drives a UCD7242 integrated circuit that regulates Vccint supply voltage. A set of PMBus commands is used to configure IC functions. The UCD7242 possesses on-chip voltage- and current-sensing circuitry and communicates with the UCD9248. Our GUI software continuously transmits the corresponding PMBus commands to the device, while the voltage and current values of the DC/DC converter are received back in a standard PMBus data format. We then convert the incoming bytes to actual values and obtain the voltage and current levels of the power supply.

Running a complex and congested design in high temperatures may disturb the supply voltage levels. Unfortunately, FPGA chips have a narrow tolerable range of input voltages. Exceeding these levels can cause loss of functionality or permanent damage on the chip. Hence, voltage-level stability during challenging operating conditions is another factor to test after you have designed a board. The Vccint panel is intended to observe and verify that the power supply system of the designed board is capable of tolerating voltage variations for overwhelming processing rates under high temperatures.

One important point designers should always keep in mind is that every device has different power dissipation due to manufacturing process variations. For example, after a Virtex-7 FPGA chip is produced, a minimum voltage supply level that is sufficient to operate the chip properly is determined in test stages and written into E-fuses. Adjusting voltage regulator output to this minimum voltage level changes static power consumption. Thus, an offset in static power may cause a difference in results between XPE and our design. For instance, static power consumption of a Kintex-7 chip estimated in XPE differs by 0.7 W between typical and worst-case calculations.

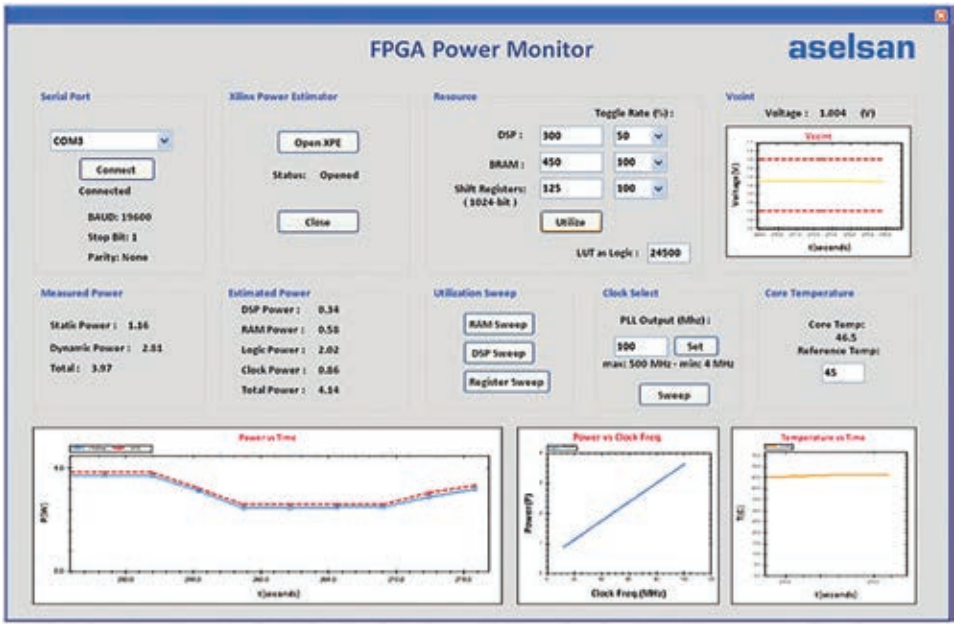


Figure 2 – Our graphical user interface

GOOD FIT WITH XPE

We have tested our design on many different design scenarios with varying factors to ensure its accuracy and reliability. In the meantime, the estimation results of XPE are also plotted on the same graphs with our measurements to compare estimated power dissipation to actual measurements.

The power behavior of the Kintex-7 device is observed by sweeping the resource usage rates starting from zero to the maximum value initially implemented on the design. Figure 3 shows the change in power dissipation as the utilization rate of a resource type is swept. Straight and dashed lines represent actual power measurements and XPE estimations respectively. We can see that the two curves follow each other very closely even when the utilization rate and power consumption are at high values.

These results show that Xilinx Power Estimator calculations are consistent with real measurements and predict the power consumption accurately. We can also conclude that our proposed method works as expected

and can be an alternative to XPE. An even better option is to use these two methods collaboratively. Testing a design with both XPE and our method enables you to double-check the results and ensures you do not make any mistake while modifying the numbers at XPE. For instance, if you do not enter the average fan-out number of a clock correctly at XPE, or simply misunderstand a concept and fill a box inaccurately, the resulting plots would not overlap, signaling that an error exists so that you can correct the entries and prevent a misleading estimation.

The rated specifications of an FPGA board, such as maximum power dissipation or allowable core temperature range, are always included in the datasheet of the board. However, a design with high resource utilization and clock frequency can exceed these rated values. Thus, it's important to verify that power regulators supply enough current to the device and that the cooling system is sufficient to keep the temperature within critical values for a possible FPGA design. Our method makes it possible to test the reliability of the board for high-performance requirements by increasing resource usage and clock frequency. Such a test helps determine the maximum signal-processing rate possible in designs implemented on this board. Alternatively, it gives us an idea about whether we need to upgrade board capabilities and the cooling mechanism of the system.

Now that test results confirm that our method is a reliable way to manipulate resource usage and evaluate power consumption, engineers have another option for power management in predesign stages. With some further improvements, such as a JTAG interface, fully device-free VHDL code or a common current-sensing system, this design would be a key tool for FPGA projects.

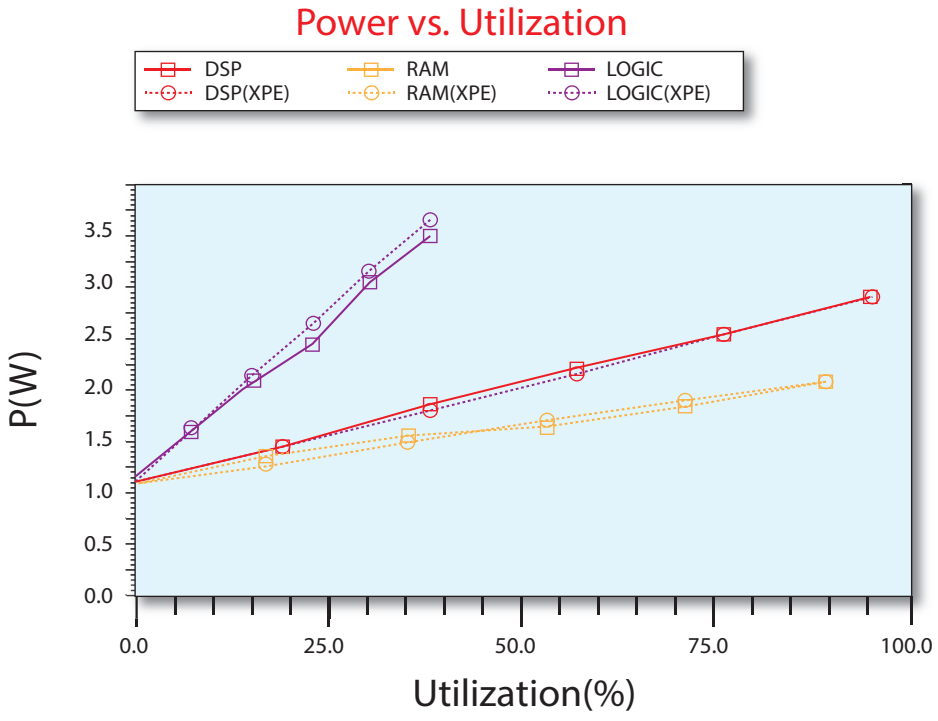


Figure 3 – In this graph plotting power vs. utilization, straight lines represent actual power measurements and dashed lines the XPE estimates.