Measurement of FPGA Die Temperature Using Run-time Reconfiguration

Sergio Lopez-Buedo, Javier Garrido, and Eduardo Boemo

Escuela Técnica Superior de Informática, Universidad Autónoma de Madrid,
Ctra. de Colmenar Km.15, 28049 Madrid, Spain.
{sergio.lopez-buedo,javier.garrido,eduardo.boemo}@uam.es

Abstract

In this paper a temperature sensor for FPGAs that can be dynamically inserted and eliminated from the circuit is presented. Using run-time reconfiguration, a ring-oscillator together with its auxiliary circuitry (counting and control stages) is inserted in the design. After the actual temperature of the die is captured, the value is read back and the sensor is eliminated from the FPGA. This avoids self-heating, and also releases FPGA resources. The main advantage of the technique is that the sensor is completely stand-alone: no I/O pads are required, and no permanent use of any element of the FPGA is done. The sensor is small enough for thermal mapping, allowing the designer to construct the isotherm lines of the die.

1. Introduction

Ring-oscillators can be utilized to measure die temperature [1]. Particularly, in the area of programmable logic devices they represent a simple and efficient tool for thermal management [2], [3]. They can be implemented on all FPGA architectures, and its full-digital output makes a good alternative to dedicated temperature-sensing diodes (like the ones included in the Virtex family). While diodes have the advantage of being immune to power supply variations, its analog output represents a drawback. Even if the A/D converter can be implemented in the FPGA, a few external components need to be used [4].

Previous strategy for measuring chip temperature in FPGAs was to add a ring-oscillator in the circuit at design time [3]. Two I/O pads were required: one to output the signal, and another one to disable the oscillator in order to minimize self-heating. The output could also be measured internally using a counter. But this option requires extra FPGA resources not only for counting and controlling, but also for implementing a port compatible with an external microprocessor (Fig.1). On the contrary, in the approach proposed in this paper no hardware overhead exists. Run-time reconfiguration is employed to insert a small sensor in the FPGA every time it is needed. This is accomplished using the standard configuration port. Once the measurement has been made, the same configuration port is used to read back the temperature, and finally the circuit is eliminated from the FPGA configuration.

Not only run-time reconfiguration is useful for thermal testing; the opposite is also true. FPGA based systems employing partial run-time reconfiguration, where different parts of the circuit are dynamically added or deleted, are prone to errors. The great number of possible configurations could make the a priori check computationally unfeasible. This is especially critical in evolvable systems, where the current configuration is a mutation of a previous one. Even if it is possible to perform this check, there are few tools that do it [5]. Therefore, thermal testing can demonstrate very useful to detect errors that cause excessive power dissipation, like bus contention or unconnected inputs. If the sensor utilized is small enough to allow performing a thermal map of the chip, it is possible not only to detect errors, but also their locations [6].

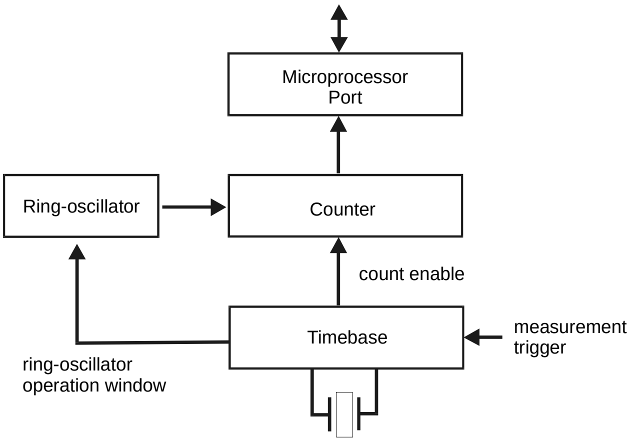

Fig.1. Minimum circuitry to sense die temperature.

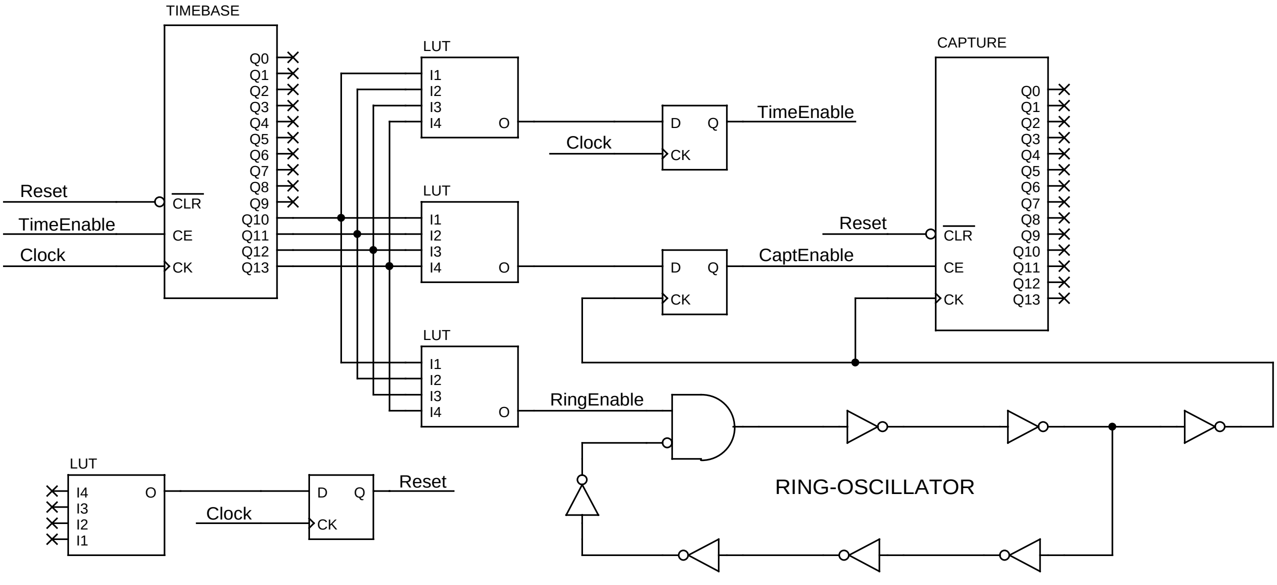

Fig. 2. Schematic diagram of the sensor.

2. Description of the sensor

The sensor is basically composed by two counters and a ring-oscillator that provides a temperature-dependent frequency output. The first counter (timebase) generates two fixed intervals during which the oscillator and the measurement will be enabled. The second counter (capture) just adds up the number of pulses generated by the oscillator during the time determined by the first counter.

2.1. Design methodology

All the hardware of the sensor was written in Java, using the JBits API (*Application Program Interface*) [7]. In short, JBits is a set of Java classes and methods that allow a low-level manipulation of the FPGA resources. JBits can be used to create a circuit using structural descriptions. Unfortunately, behavioral constructs are not supported. Only the Virtex family is supported in the current version of JBits (2.7) offered by Xilinx.

The first step in JBits is to read an existing configuration bitstream, which will be used as a base for the design. To create the logic elements for the new circuit the most common alternative is employing the library of parameterizable cores, but it is also possible to directly access to the low-level resources (look-up tables, multiplexers, etc.). Similarly, it is possible to implement the connections of the circuit using router tool provided with JBits, while the other alternative is to explicitly declare the routing resources to be used (local or global connection lines, PIPs, etc.). Once the bitstream for the new design is prepared, it can be written to the FPGA.

The main advantage of JBits is its support for partial run-time reconfiguration [8]. Once a bitstream is downloaded to the FPGA, in the following downloads only the parts that have changed are updated. This allows the user to perform very fast run-time reconfigurations, without having to interrupt the normal operation of the system. Finally, the contents of the FPGA can be read back from JBits, allowing the host processor to know the present state of the circuit.

2.2 Constructive details

The schematic for the sensor is shown in Fig.2. The ring-oscillator was constructed using 7 inverters, each implemented in a 4-input LUT (*look-up table*). The LUTs were placed separated, leaving a minimum 1-CLB (*configurable logic block*) space between them in order to increase the routing delay. Note that in this case a high frequency output is not desirable: it implies both a higher counter size and extra self-heating. In the experimental measurements, a frequency output of 45.5 MHz at 25°C was obtained using an XCV50PQ240-4 FPGA.

In order to simplify the code, the two counters included in the sensor are identical, featuring a length of 14 bits. The only external port is the clock input of the timebase counter, which should be connected to one of the main FPGA clocks. Three enable signals, generated by 4-input LUTs, are derived from the four most significant bits of this counter.

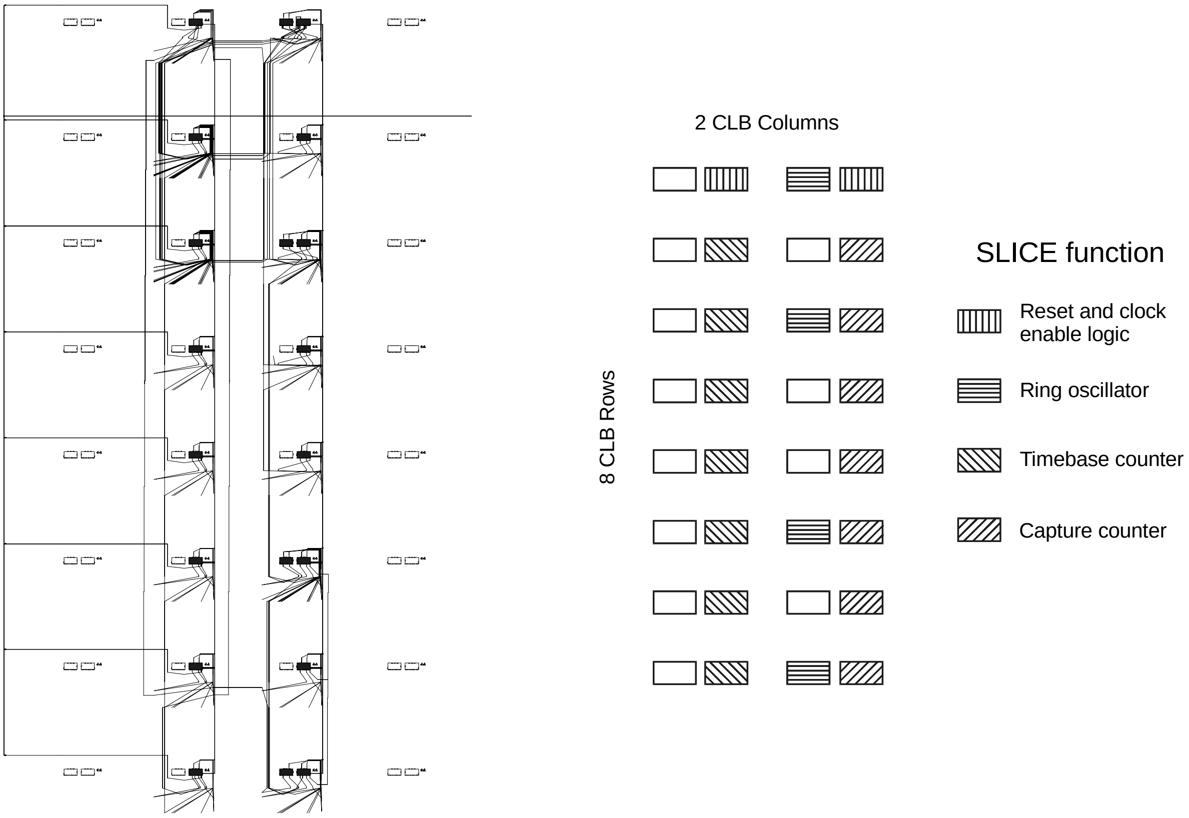

Fig. 3a. Sensor layout (left). **Fig. 3b.** Slice usage of the sensor (right).

The first, RingEnable, allows the ring-oscillator to start running. The second, CaptEnable, establishes the interval during which the ring-oscillator frequency output will be measured by the capture counter. Finally, TimeEnable is used to disable the timebase counter when it is no longer necessary. Consequently, when the measurement has finished all the circuit is disabled, thus avoiding spurious power dissipation.

This schema allows specifying any of the three enable times in steps of 1024 cycles of the main FPGA clock (10 bit count). For example, in the experiments the ring-oscillator runs in every measurement for 2048 cycles. As a 3.6864 MHz clock was used, this corresponds to 555 µs. The capture counter was enabled during the second half of that time (277 µs), leaving the first 1024 cycles as a safe margin for oscillation start-up.

Once the measurement has been made in the capture counter remains a value that is proportional to the frequency output of the ring-oscillator, and therefore, to the die temperature. This count is read back by the host processor, which calculates the actual temperature. Thus, the use of I/O pads to transfer the data is eliminated.

The sensor was created as a RTP (run-time parameterizable) core, so it can be easily instantiated in any JBits-based design. The parameters of the core are the timings of the three enable signals, expressed as the contents of the LUTs that generate them. The sensor layout is depicted in Fig. 3a, while the slice (half CLB) usage is showed in Fig. 3b. The size is 8 CLBs tall by 2 CLBs wide. That is, less than 5% of the total area of the smallest Virtex device, the XCV50. This fact makes the sensor suitable for mapping the thermal state of the FPGA in different points, especially if the biggest devices are utilized. For example, in the XCV1000 model, it occupies less than the 0.3 % of the device.

3. Operation of the sensor

Run-time reconfiguration is a powerful design technique based on dynamically altering the contents of the configuration memory of an FPGA. This is used, for example, to create adaptive or evolvable systems. But it is also possible to modify only some sections of the FPGA configuration memory, leading to changes only in some portions of the circuit, while the remainder continues its normal operation. In Virtex FPGAs the configuration bitstream is organized in frames, each corresponding approximately to a vertical portion of a column of CLBs. Partial reconfiguration is available on a frame-by-frame basis [9]. Taking into consideration that the frame size in the XCV50 is 48 bytes, and that the maximum configuration speed is 50 MB/s, the minimum configuration change will take roughly 1 µs, fast enough to permit real-time applications.

In all FPGA families a global reset signal is asserted after a full reconfiguration to initialize all flip-flops to a known state. Partial run-time reconfiguration is incompatible with this behavior, because the assertion of the global reset will initialize all FPGA registers, not only the ones in the portion of the circuit being inserted. Therefore, the part of the circuit that has not changed will not continue its normal operation, because it will be reset instead. But if the global reset is not asserted, the new logic inserted in FPGA will initialize at an unknown state, possibly causing a failure, especially if contains a counter or a state machine.

A common scenario for the use of this type of thermal strategy would be a reconfigurable board composed by one or many FPGAs, and a host microprocessor running a Java Virtual Machine (JVM). During the initialization, the host microprocessor configures the FPGAs, and the system starts working. Whenever the host requires checking the thermal status of the board, the following steps are executed:

- The current bitstream is inspected to find a free space where to insert the sensor. This operation can be made statically, providing a free space to insert the sensor at design time. But JBits also allows the designer made it dynamically, finding a place where to insert the sensor. It is useful in evolvable and adaptive systems, where the designer could ignore what circuit will be running in the FPGA at a certain time.

- Once a possible location for the sensor is found, its elements are inserted and routed. Routing is dynamic, once again to avoid being limited to static designs. The possibility of a static routing also exists, but in this case an area of the FPGA totally free of routes should be reserved.

- The new bitstream (with the sensor embedded in it) is generated by JBits, and the FPGA is partially reconfigured: only the modified frames are updated. As a consequence, the sensor is inserted in the FPGA meanwhile the system continues its normal operation.

- As the FPGA is only partially reconfigured, the state of the registers of the sensor is unknown (the global reset signal will not be asserted to permit the normal operation of the circuit that is already running in the FPGA). To solve this problem, the first circuit downloaded will have its reset signals activated, so that the count will be initialized at zero. Since the reset signal is enabled, no activity will be present. Immediately after that the sensor is reconfigured again with the reset signals de-asserted (only one LUT needs to be changed), and then the normal sensing begins.

- Finally, once the measurement has been made, the state of the sensor is read back to obtain the frequency of the ring-oscillator. The host microprocessor then translates this number into the actual temperature, using a pre-calculated table.

4. Results

In order to verify the feasibility of the above strategy, a Xilinx AFX PQ240-100 prototyping board with an XCV50PQ240-4 was utilized. The FPGA configuration data was written and read back using its SelectMAP port, which was connected via a custom interface to a host PC running JBits. This link was implemented through the parallel port of the PC, which allowed a simple, yet full connectivity to the FPGA, at a moderate configuration speed (around hundreds of KB per second). The use of a prototyping board allowed us to have a complete access to all pins of the FPGA, and permitted changing both the temperature of operation (by introducing the board in a temperature-controlled oven) and the power supply voltage. To minimize the effect of self-heating, a low frequency system clock was used, just 3.6864 MHz.

Fig. 4 shows the output of the sensor versus the die temperature, and Fig. 5 its dependence to power supply variations. The two graphics are normalized at 25°C and 2.5V. Both the experimental measurements and the ones predicted by the *prorating* option of the timing analysis tools are displayed. The graphics show that the variation of the oscillation frequency with the temperature is large enough to make this technique feasible. It could be also inferred that large increments of the chip temperature (such as those caused by serious circuit errors) can not be masked by moderate (±5%) power supply variations. Finally, it should be noted that there is a significant difference between the experimental results and the ones predicted by the tool provided by Xilinx. This makes advisable a previous calibration of the sensor if accurate measurements are needed.

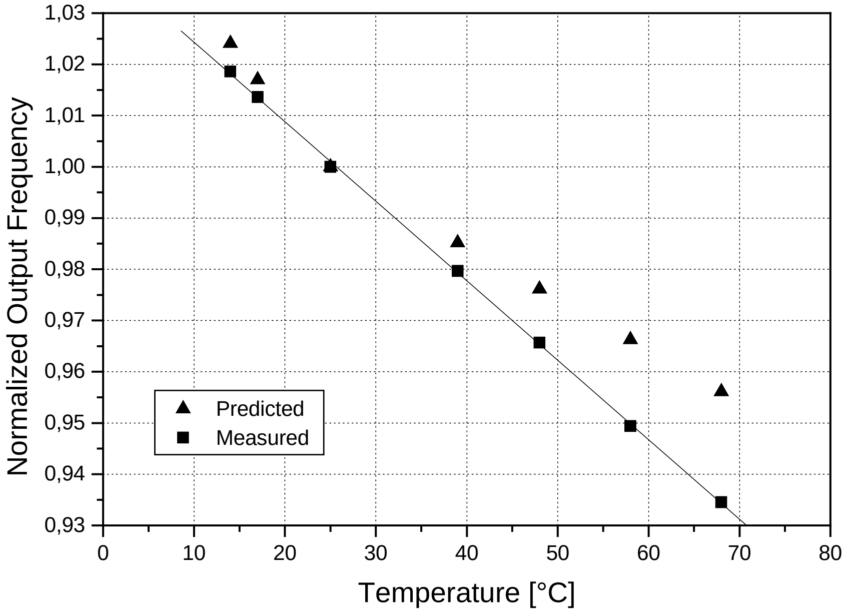

Fig. 4. Ring-oscillator frequency response versus die temperature.

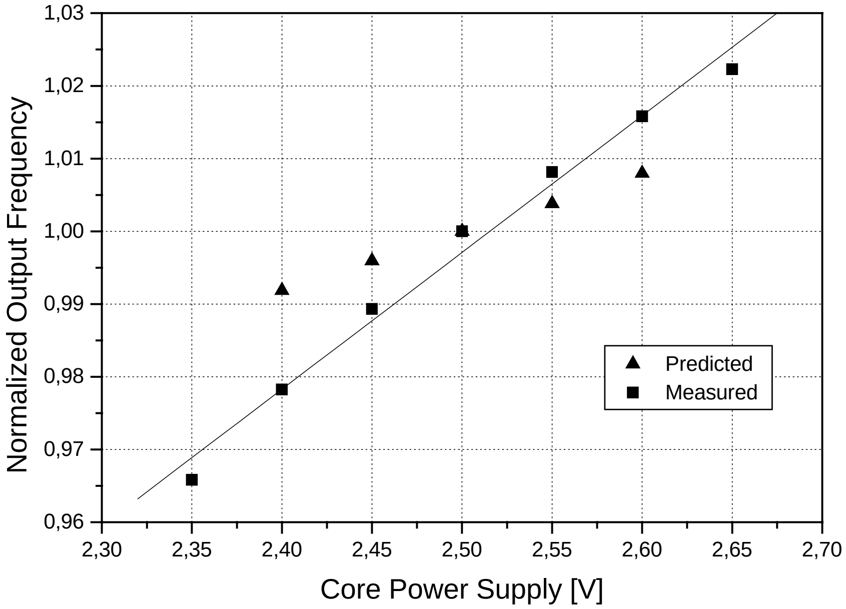

Fig. 5. Ring-oscillator frequency response versus power supply voltage.

5. Conclusions

In this paper, a solution for thermal testing of FPGAs using run-time reconfiguration has been presented. This technique allows the measurement of the die temperature of an FPGA without permanently using any resources. The temperature sensor is completely stand-alone, so no A/D converters or other external devices are needed. Moreover, it is not necessary to make any PCB modifications to implement this method (provided that run-time reconfiguration is already supported) as no additional I/O pads are used. Finally, the functionality of the JBits technology was successfully tested, proving useful for thermal testing applications. Therefore, designs already employing run-time reconfiguration can use this strategy to detect errors that cause excessive power dissipation before they could damage the device.

Acknowledges

The authors wish to thank Eduardo Ruiz and Pedro Rodríguez for their valuable help during the setup of the experiments. This work has been supported by funds provided by contract 658001 of the *Fundación General de la Universidad Autónoma de Madrid* (Spain).

References

[1] G. Quenot, N. Paris and B. Zavidovique, “A temperature and voltage measurement cell for VLSI circuits”, *Proc. 1991 EURO ASIC Conf.*, pp.334-338, IEEE Press, 1991.

[2] E. Boemo and S. Lopez-Buedo, "Thermal Monitoring on FPGAs using Ring-Oscillators", *Proc. FPL 1997 Workshop, Lecture Notes in Computer Science 1304*, pp.69-78, Berlin: Springer-Verlag, 1997.

[3] S. Lopez-Buedo, J. Garrido and E. Boemo, "Thermal Testing on Reconfigurable Computers", *IEEE Design & Test of Computers*, pp.84-90, January-March 2000.

[4] Xilinx Inc., “Virtex Analog to Digital Converter”, *Application Note XAPP155*, 1999.

[5] David Robinson and Patrick Lysaght, “Verification of Dynamically Reconfigurable Logic”, *Proc. FPL 2000 Workshop, Lectures Notes in Computer Science 1896*, Berlin: Springer-Verlag 2000.

[6] J. Altet, A. Rubio, E. Schaub, S. Dilhaire and W. Claeys, “Thermal Coupling in Integrated Circuits: Application to Thermal Testing”, *IEEE Journal of Solid-State Circuits*, Vol. 36, No. 1, January 2001.

[7] Xilinx Inc., “JBits Xilinx Reconfigurable Computing Platform”, *JBits 2.7 Tutorial*, 2000

[8] Scott McMillan and Steven A. Guccione, “Partial Run-Time Reconfiguration Using JRTR”, *Proc. FPL 2000 Workshop, Lectures Notes in Computer Science 1896*, Berlin: Springer-Verlag 2000.

[9] Xilinx Inc., “Virtex Configuration Architecture Advanced User Guide”, *Application Note XAPP151*, 2000.