SATTA: a Self-Adaptive Temperature-based TDF awareness methodology for dynamically reconfigurable FPGAs

A

SATTA: a Self-Adaptive Temperature-based TDF awareness methodology for dynamically reconfigurable FPGAs

STEFANO DI CARLO, GIULIO GAMBARDELLA, PAOLO PRINETTO, DANIELE ROLFO, and PASCAL TROTTA, Politecnico di Torino

Dependability issues due to non functional properties are emerging as major cause of faults in modern digital systems. Effective countermeasures have to be presented to properly manage their critical timing effects. This paper presents a methodology to avoid transition delay faults in FPGA-based systems, with low area overhead. The approach is able to exploit temperature information and aging characteristics to minimize the cost in terms of performances degradation and power consumption. The architecture of a hardware manager able to avoid delay faults is presented and deeply analyzed, as well as its integration in the standard implementation design flow.

Categories and Subject Descriptors: B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance

General Terms: Design, Performance, Reliabiity

Additional Key Words and Phrases: FPGA, Aging, Partial Reconfiguration, Transition Delay Faults

ACM Reference Format:
Stefano Di Carlo, Giulio Gambardella, Paolo Prinetto, Daniele Rolfo, and Pascal Trotta, 2013. SATTA: a Self-Adaptive Temperature-based TDF awareness methodology for dynamically reconfigurable FPGAs. *ACM Trans. Reconfig. Technol. Syst.* V, N, Article A (January YYYY), 23 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Field Programmable Gate Arrays (FPGAs) are among the fastest devices moving towards low feature size and innovative technologies such as 16nm FinFET [Xilinx Corporation 2013c] and 14nm Tri-Gate [Altera Corporation 2013]. This has been pushed by FPGA low non-recurring engineering cost and massive parallel architecture that suit the requirements of high-performance Systems-on-Programmable-Chips (SoPCs). However, aggressive technology shrinking boosts the effect that non-functional properties have on device reliability [Srinivasan et al. 2008].

Accelerated device aging increases the occurrence of disruptive effects, such as Electro-Migration (EM) and Time Dependent Dielectric Breakdown (TDDB). Moreover, it aggravates non-disruptive phenomena such as Negative Bias Temperature Instability (NBTI) and Hot Carrier Injection (HCI) that lead to impaired timing performance of the device [Stott et al. 2010]. The rate of the degradation depends on several parameters, including supply voltage, temperature, switching activity, state of the gates, and leakage current.

Among the others, high temperature is one of the main parameters responsible for the degradation of the connections' delay and system's timing [Choi et al. 2007]. This degradation may have severe effects leading to the generation of transient faults, such as Transition Delay Faults (TDFs) [Park et al. 2009]. TDFs are caused by incorrect data sampling due to timing variations in specific paths of a circuit. If a register samples a signal close to the clock rising edge, metastability may occur due to the violation of the set-up and hold times. Furthermore, if the data has a late transition and changes its value after the clock rising edge, wrong information may be sampled.

TDFs have been intensively studied for Application Specific Integrated Circuits (ASICs). Several efficient solutions spanning from Dynamic Frequency Scaling (DFS) [Fukazawa et al. 2008] to Dynamic Voltage Scaling (DVS) [Das et al. 2006] have been proposed in the scientific literature and are applied to real designs. However, when considering SoPCs, the use of FPGA specific facilities to countermeasure TDFs is a topic that has not been yet fully investigated.

Previous publications, addressing aging phenomena and late transitions effects in FPGA-based systems, mainly focus on TDFs detection [Li and Lach 2007; Amouri and Tahoori 2011; Valdes-Pena et al. 2013], requiring significant manual design efforts to obtain aging-aware designs [Bexiga et al. 2011]. Only a few attempts to exploit Dynamic Partial Reconfiguration available in several FPGA architectures to design ad-hoc countermeasures against TDFs have been proposed [Di Carlo et al. 2012; Kameda et al. 2012].

This paper recognizes Dynamic Partial Reconfiguration as a powerful instrument to mitigate TDFs effects in complex SoPCs and proposes a Self-Adaptive Temperature-based TDF Awareness (SATTA) methodology exploiting FPGA Dynamic Partial Reconfiguration to prevent TDFs in complex SoPCs. One of the main contribution of SATTA is a self-adaptive hardware manager able to tune the system's working frequency, at run-time, counteracting timing variations introduced by aging effects. This manager exploits temperature information to smartly activate itself, forecasting the aging effects on the system, and self-adapting to the timing-degradation figure encountered during its operational conditions. In addition, it requires minimum design modifications to achieve TDFs detection and avoidance and features easy integration in state of the art FPGA design flows.

The rest of the paper is organized as follows: Section 2 overviews TDFs causes and effects, exploring existing solutions already developed for ASICs and FPGAs. Section 3 introduces the sensors employed in SATTA and details the hardware architecture of the SATTA manager. Section 4 focuses on the integration of the proposed methodology in state of the art FPGA design flows. Section 5 reports experimental results collected while applying SATTA design methodology on several case studies. Eventually, Section 6 concludes the paper.

2. BACKGROUND AND RELATED WORKS

In a complex FPGA-based SoPC, high clock frequency and precise timing optimization are crucial to guarantee high performance and throughput. Designers try to push systems' working frequency to its limit, which is mostly settled by the maximum path delay of the design. Proper *set-up* ($\mathrm{T_{setup}}$) and *hold* ($\mathrm{T_{hold}}$) times at the input of each flip-flop must be guaranteed in order to sample stable signals, thus avoiding metastable states and sampling of wrong values belonging to previous or following clock cycles.
Faults due to incorrect sampling caused by timing issues are in general referred to as *Transition Delay Faults* (TDFs). Fig. 1 shows a typical TDF example in which the signal fed to a sampling register violates the set-up time.
TDFs are often caused by non-functional parameters such as process variations, voltage fluctuation, high temperature and aging. Aging effects, such as NBTI and HCI

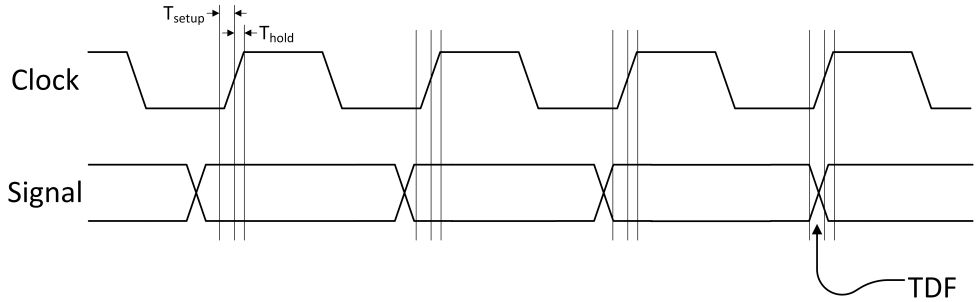


Fig. 1: Example of Transition Delay Fault. A signal connected to the input of a register or flip-flop violates the set-up time changing its value in correspondence to the clock rising edge and thus increasing the probability of sampling a wrong value.

may deteriorate the system's timing during its lifetime. NBTI and HCI affect PMOS and NMOS transistors by respectively decreasing their capability of charging and discharging capacitors. Therefore, both *low-to-high* and *high-to-low* transitions are slowed down. If the system's clock frequency is set at design time, unless worst-case safety margins too detrimental for the overall system's performance are settled, the delay effects induced by NBTI and HCI may be far from being negligible, introducing an undesired high rate of TDFs. Aging due to NBTI and HCI strongly depends on the static probability of signal values. NBTI ages a PMOS transistor when it is negatively biased (i.e., a logic '0' is applied to the gate), during the so called *stress phase*. Whenever the driven signal is set to logic '1' (*recovery phase*), the PMOS aging effect is partially mitigated. Similarly, HCI ages a NMOS transistor when it is positively biased and recovers the aging effect when the input signal is set to logic '0'.

High temperature is another significant TDF cause. It magnifies aging effects such as NBTI and HCI and, furthermore, it directly impacts the transistors' delay by increasing their threshold voltage. Park et al. [Park et al. 2009] experimentally demonstrated that working frequency penalty triplicates when changing the operating temperature from 25 °C to 125 °C.

Solutions targeting TDFs avoidance and detection have been deeply studied in ASICs. Dynamic Frequency Scaling (DFS) [Fukazawa et al. 2008] consists in dynamically decreasing the system's working frequency when the path delay increases. TDFs are avoided by keeping a guard-band on the set-up time. On the other hand, Dynamic Voltage Scaling (DVS) [Das et al. 2006; Das et al. 2009] relies on increasing the voltage source to increase the drain current, thus compensating the performances losses. Another approach to increase drain current capability consists in modifying the body bias voltage, as in [Fuketa et al. 2012]. DVS and DFS have also been merged to maximize their mitigation efficiency [Sato and Kunitake 2007].

Adapting ASIC approaches to FPGA-based systems is non-trivial, due to the fixed FPGA internal architecture that restricts the set of available design solutions. A set of publications propose the use of sensors based on shadow registers with clock skewed with respect to the primary one in order to detect delay variations [Amouri and Tahoori 2011; Valdes-Pena et al. 2013; Li and Lach 2007]. [Amouri and Tahoori 2011] proposes a sensor for detecting late transitions due to aging effects in FPGAs. The sensor is designed in order to be inserted in parallel at the endpoint of the critical path and exploits two flip-flops wit a positive clock phase to detect late transitions. One of the main drawbacks of this sensor is that it enables to detect TDFs when they arise and

therefore it requires complex recovery strategies that usually imply stalling the system. Differently from [Amouri and Tahoori 2011], [Li and Lach 2007] proposes to use negative clock skew to detect delay variations thus enabling to identify paths that are reaching a critical delay level before actual faults arise. The goal of the proposed technique is to accurately and precisely characterize the register-to-register delays of a large number of otherwise unobservable combinational paths at test-time. Even if not explicitly defined for FPGA designs, the method has been applied for the characterization of a FPGA based test case. A similar idea has been presented in [Valdes-Pena et al. 2013], where a similar sensor is employed to detect the likelihood of TDFs occurrence in selected critical paths. Despite we recognize that the use of sensors based on shadow registers with clock negatively skewed with respect to the primary one is a very efficient method to detect delay variations, in these publications, no counteractions at system level are presented to exploit the gathered information at run-time to avoid TDFs occurrence in FPGA designs. Furthermore, these solutions are characterized by a constant monitoring of the observed paths, thus incurring in sensor self-aging and power waste due to clock duplication, if used for run-time monitoring of FPDA aging.

Being aware of the TDF main causes and considering the models beyond their occurrences are crucial to efficiently tackle the problem of TDF avoidance. Nonetheless, static probability of signal values and temperature profile of a system depend on the actual workload and on the environmental conditions [Zick and Hayes 2010]. They are therefore hard to estimate at design time [Amouri and Tahoori 2012]. This forces designers to take worst-case decisions when selecting the system's working frequency.
This paper tries to overcome these limitations presenting SATTA, a design solution able to exploit run-time device temperature information to predict delay variations and to smartly activate temperature sensors, thus reducing power overhead. Furthermore, SATTA manages the dynamic reconfiguration of the system's working frequency, taking into account the delay dependency on the temperature and the transitory nature of aging phenomena (due to signal probabilities and temperature profiles). The proposed methodology has the potential to efficiently avoid TDFs, guaranteeing a graceful degradation of system's performance when required, but restoring or boosting it in absence of critical conditions. A manager, implementing the proposed methodology, is introduced in this paper, in order to easily apply the methodology to FPGA-based SoPCs.

3. SATTA SENSORS ORGANIZATION AND ARCHITECTURE

SATTA implements a TDFs avoidance strategy based on the monitoring of the delay of those signals routed through the critical paths of the FPGA design. Two main events may arise:

(1) whenever, due to increased temperature and/or aging effects, the delay of a critical path approaches the set-up sampling time limit, the system's clock frequency is lowered;
(2) whenever, due to decreased temperature and/or recovery from aging effects (see Section 2), the critical path delay decreases, the system's clock frequency is slowly increased again.

A set of hardware modules are required to efficiently implement this monitoring activity and the related reaction policies. These modules are designed exploiting a set of features commonly available in modern reconfigurable devices, thus minimizing area, power and performance overheads. Resorting to widely spread FPGA functionalities enables easy implementation on all modern dynamically reconfigurable FPGAs, such as the newest Xilinx Virtex [Xilinx Corporation 2012] or Altera Stratix [Altera Corporation 2012] FPGAs.

Figure 2 graphically summarizes the overall SATTA architecture. It includes:

— One or more *Temperature sensors*. They measure the temperature of the device and provide this information to the manager that exploits it to estimate the system's path delay status.
— One or more *TDF sensors*. A TDF sensor is able to monitor the delay of a given path asserting a warning signal whenever it increases to a level that may compromise the correct sampling of the information.
— A *CLK Generator*. It generates the set of clock signals (i.e., CLK1 and CLK2) required by the TDF sensors to properly operate. It enables to dynamically change the phase relationship between CLK1 and CLK2. This feature is exploited by the manager to measure the actual delay in the observed paths. Moreover, it can dynamically modify the frequency of the main system's clock (CLK1) at run-time, exploiting FPGA Dynamic Partial Reconfiguration (DPR). This enables to adapt the system to the temperature and aging conditions, avoiding TDFs while maximizing the performance.
— A *Manager*, that collects information acquired from the sensors (i.e., *TDF sensors* and *Temperature sensors*) and issues proper reconfiguration commands for the *CLK Generator* without interrupting the system's tasks.

Details about the implementation of each block are provided in the next sections.

3.1. Temperature Sensor

Temperature has a strong impact on aging effects and on TDFs in general. An increase of the FPGA temperature would, on the one hand, heighten the path delay and, on the other hand, raise the transistor's threshold voltage due to NBTI.
Fig. 3 plots the trend of a path delay in a Virtex 4 device, depending on the die temperature, using the temperature model provided by Xilinx, extracted with the aid of the timing analysis tool. Fig. 3 shows an increase of more than 300 ps when the operating temperature raises from 0°C to 85°C. Similar effects are also experimentally measured on different devices such as 45 nm Spartan 6 and 28 nm Artix 7 devices [Pfeifer and Pliva 2012; 2013]. Device temperature must therefore be monitored, and gathered information must be exploited to predict the delay behavior and to take proper countermeasures.

Both Altera and Xilinx FPGAs include a built-in temperature sensor diode. However, this diode is located in a fixed position of the die and is unable to provide measures related to a specific path. Following [Happe et al. 2011], SATTA exploits the built-in temperature diode as a reference sensor and a ring oscillator with odd number of inverters (i.e., 11 inverters mapped on different FPGA slices) as an approximated temperature sensor. This structure has several advantages: (i) it does not require device dependent hardware features, (ii) it enables to measure temperature with a reasonable area occupation and accuracy, and (iii) it enables to instantiate several sensors in the device, monitoring the temperature in the area in which the controlled critical path is routed. Looking at Fig. 3, it is worth to highlight here that the temperature can be measured with an uncertainty in the measure of some degrees without introducing significant errors in the corresponding delay estimation. This enables us to strongly reduce the number of cycles required to obtain a measurement from the ring oscillator. Therefore, differently from [Happe et al. 2011] that uses 2^{17} measurement cycles, SATTA can perform measurements with a drastically reduced number of cycles (i.e., few tens of cycles are sufficient to have a measure suitable for the SATTA purposes).
If the available hardware resources are very limited, the reference diode alone can still be used to evaluate the temperature in the device. In this case, temperature information provided by the diode refers to the die temperature. Even if it is less specific,

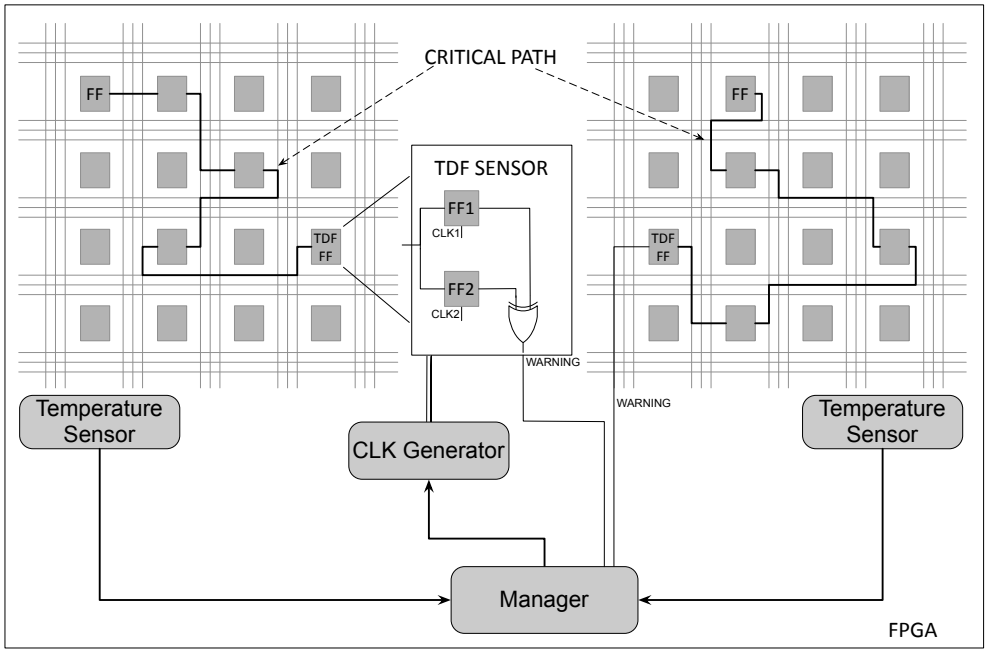


Fig. 2: SATTA overall architecture. TDF detection and mitigation is implemented inserting in the design: (i) a set of temperature sensors, (ii) a set of TDF sensors, (iii) a clock generator, (iv) and a global manager.

this information can still be used to evaluate the overall temperature at which the device works. It is important to highlight here that sensors are subject to aging as well. Therefore, the temperature should not be constantly measured to avoid both sensor self-aging and to decrease power consumption.

If more built-in temperature sensors spread on the FPGA will be available in next generation devices, they can be exploited to obtain fast and accurate temperature measurements and to save area dedicated to ring oscillators used as temperature sensors. Nevertheless, these sensors require the use of built-in analog-to-digital converters available on the FPGA. These converters represent limited resources often required to interface the FPGA with external devices. Ring oscillators remain a viable general solution whenever these resources must be allocated to different tasks related to the system's mission.

3.2. TDF Sensor

Once the most critical paths of the design have been identified, a TDF sensor for each critical path is inserted. The TDF sensor is a special block able to monitor the delay of a path. Each critical path includes several levels of combinational logic and is delimited

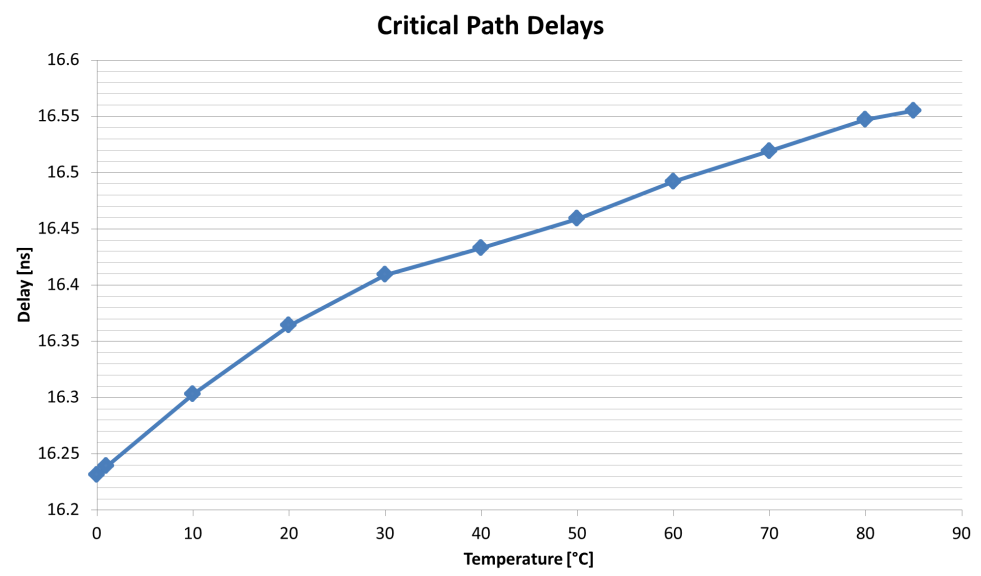


Fig. 3: Example of a critical path delay variation w.r.t. temperature in Virtex 4 Devices

by two memory elements (e.g., flip-flops) working at the system's clock frequency. The TDF sensor replaces the flip-flop placed at the end of the selected critical path.

Following previous works [Li and Lach 2007; Valdes-Pena et al. 2013] we exploit a sensor based on a shadow flip-flop clocked with a negative clock skew with respect to the main system's clock. The sensor is composed of two flip-flops (FF1 and FF2) and a 1-bit XOR gate (Figure 2). It is connected to the selected path and clocked with two different clock signals. Since the two flip-flops are clocked with two different clock signals, they sample the critical path output at two different times (see Figure 4). FF1 is clocked at the system's clock *CLK1* and its output is connected to the rest of the system. Instead, FF2, clocked at *CLK2*, acts as a warning sensor. *CLK2* is a replica of *CLK1* shifted with a negative phase, in order to sample the data before *CLK1*.

The output of the two flip-flops is compared using the XOR gate and a warning signal is asserted if the two sampled values differ. This happens if the sampled signal stabilizes too close to the rising edge of the system's clock, indicating that the path delay is increasing due to temperature variations, aging effects, or presence of glitches.

Setting a negative phase between *CLK1* and *CLK2* is pivotal to obtain fault avoidance. When the path delay increases, metastability conditions arise in FF2 first. If timely detected, the SATTA *Manager* can take the proper countermeasures to prevent that the path timing degrades at a level in which also FF1 is compromised generating an error in the system. The way this countermeasures are implemented will be better detailed in the next sections.

To increase the accuracy of the TDF sensor, FF1 and FF2 must be placed close to each other in the design, thus avoiding timing differences that may cause erroneous TDFs warnings. Differently from previous approaches [Valdes-Pena et al. 2013], that rely on post-placement manipulation to add the desired register by hand, we designed a procedure to easily add TDF sensors at RTL level while respecting this constraint (see Section 4 for details on the sensors insertion methodology).

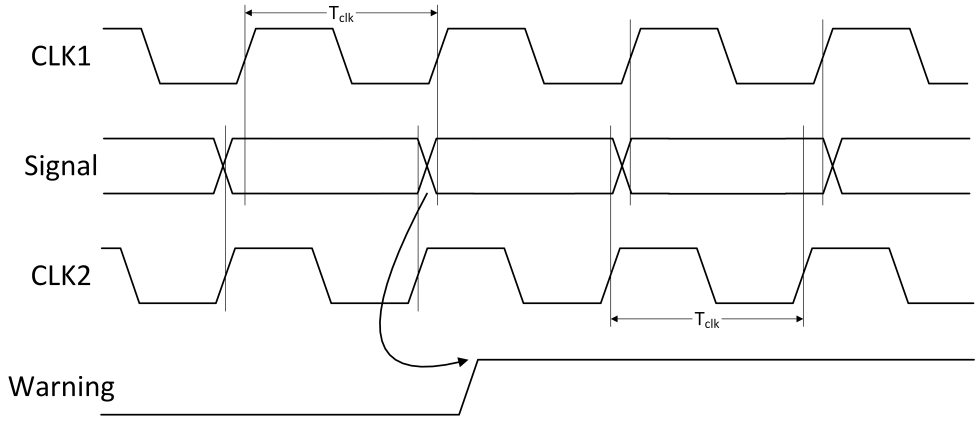


Fig. 4: Behavior of the TDF sensor

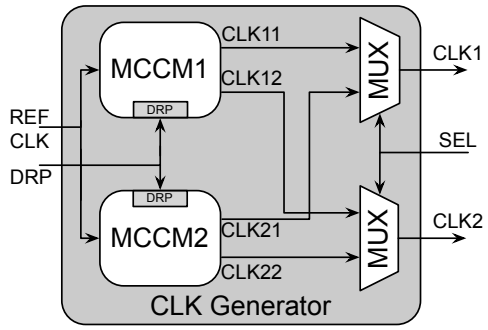


Fig. 5: Clock Generator internal architecture

3.3. CLK Generator

The *CLK Generator* is responsible for generating both the main system's clock *CLK1*, and the negative shifted clock *CLK2*. These two signals are generated using reconfigurable *Phase Locked Loop* (PLL) hard-macros available in modern FPGAs [Xilinx Corporation 2012; Altera Corporation 2012].

Reconfigurable PLLs have two main characteristics: (i) they can be configured at run-time, thus enabling to adapt the system's clock frequency and the clock phase difference according to the *Manager* requests, and (ii) they enable the generation of multiple clock signals. Nevertheless, the main problem of these PLL hard-macros is that, in general, they must be reset after each reconfiguration. To avoid any interruption of the system's operation, two PLLs are therefore used. Figure 5 shows the internal architecture of the *CLK Manager* implemented in a Xilinx 7 Series FPGA [Xilinx Corporation 2012] (similar considerations and architectures also apply for Altera FPGAs). It includes two Xilinx Mixed Mode Clock Managers (*MMCM*) hard-macros and two clock multiplexers [Xilinx Corporation 2013a]. Each *MMCM* can synthesize one or more clock signals starting from a reference input clock. Each output clock signal can have independent frequency and phase relationship with respect to other outputs.

In the proposed *CLK Manager* architecture, each *MMCM* generates two clock signals (*CLK11*/*CLK12* for *MMCM1* and *CLK21*/*CLK22* for *MMCM2*).

At system's start-up, *MMCM1* is selected as main clock source and therefore *CLK11* is used as main system's clock while *CLK12* represents the negative shifted secondary clock, required by the TDF sensors.

Whenever the *Manager* detects a warning from one of the TDF sensors, it reconfigures *MMCM2* through its Dynamic Reconfiguration Port (*DRP*) [Xilinx Corporation 2012]. The reconfiguration process sets new clock frequency parameters, while the first *MMCM* is still operating with the current frequency.

When the dynamic reconfiguration of *MMCM2* is completed, it is activated and the clock multiplexers are both switched to modify the system's clock frequency, therefore setting *MMCM2* as clock source. Similar operations are performed, alternatively switching the role of *MMCM1* and *MMCM2*, whenever a new reconfiguration is required. The inactive *MMCM* is powered down to reduce the power consumption of the clocking network.

It is important to highlight that the switching between the two clock sources exploits the hard macro *BUFGMUX* of Xilinx FPGAs [Xilinx Corporation 2013a], that is able to avoid metastability problems. It eliminates the need to stall the system clock, thus interrupting the system operations. The reader may refer to [Xilinx Corporation 2013a] for more information and detailed time diagrams of the *BUFGMUX* behavior.

The proposed *CLK Generator* architecture represents an improvement with respect to the scheme presented in [Valdes-Pena et al. 2013], in which two clock generators are used, as well. The main difference between the two architectures is that in [Valdes-Pena et al. 2013] the system clock and the auxiliary clock are generated by two different clock managers, thus introducing uncertainty in the clock phase difference. Since the two clock managers are placed in different portions of the FPGA, process variations and temperature gradients can cause different skew and jitter effects between the two clock signals. The *CLK Generator* architecture presented in this paper does not incur in power penalty with respect to [Valdes-Pena et al. 2013]. In fact, only a single *MMCM* is active during the normal operation, while both *MMCMs* are turned-on only during a short transient phase (i.e., few tens of clock cycles for reconfiguration).

3.4. Manager

The Manager coordinates the activities of all the described blocks (i.e., TDF sensors, temperature sensors, and CLK generator) according to the Algorithmic State Machine (ASM) reported in Figure 6.

At the beginning, in the RESET state the system is initialized. The most important parameters managed by this block are:

— CLK_FREQ: represents the actual working frequency of the system;
— CLK_PHASE: is the actual phase relationship between the two clocks in output from the *Clock Generator*, evaluated as the ratio between the skew of *CLK1* and *CLK2* and their period;
— TEMP: represents the actual measured device temperature;
— TIMER_VALUE: represents the timeout period of an internal timer used to switch on the monitoring procedure.

The value of the different constants and thresholds (i.e., INIT_FREQ, INIT_PHASE, ΔCLK_DWFREQ) depends on the specific design and user specifications, and must be customized taking into account synthesis parameters (see Section 5).

Starting from its IDLE state, the Manager is activated at regular intervals by an internal timer (TIMER_VALUE and TIMER_END represent the timeout period and the flag, which is set at the end of this period). Each time the manager is switched-on (i.e.,

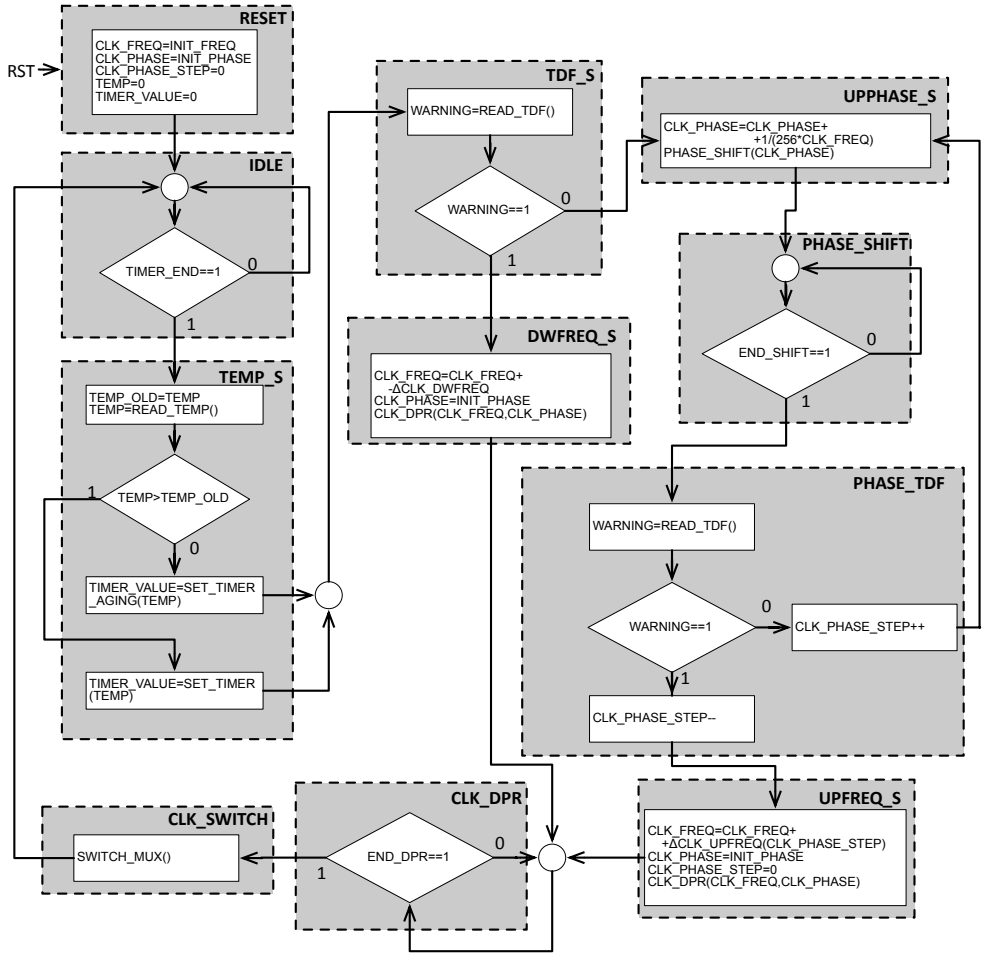


Fig. 6: ASM describing the behavior of the Manager

each time the timer reaches its time-out value) all temperature sensors are activated and, based on their readings, a new value of TIMER_VALUE is computed (TEMP_S state). If multiple temperature sensors are instantiated, only the highest temperature value is taken into account for TIMER_VALUE computation.
Basically, the temperature and TDF sensors activation period depends on: (i) the actual operating frequency of the circuit, (ii) the operating temperature, and (iii) the temperature variation speed. The sampling period must always respect the condition that the maximum increment of the path delay between two consecutive readings must not exceed the phase relationship delay between the two clocks sourcing TDF sensors (i.e., *CLK1* and *CLK2*) according to the following equation:

$$TIMER\_VALUE : \Delta\tau = \tau(t + TIMER\_VALUE) - \tau(t) < \overbrace{CLK\_PHASE \cdot T_{CLK1}}^{\text{Clock skew}} \quad (1)$$

where $\Delta\tau$ is the maximum increment in the path delay between two sensors activations, T_{CLK1} is the actual clock period of the system, and *CLK_PHASE* is the actual phase relationship, evaluated as the ratio between the skew of *CLK1* and *CLK2* and their period. This condition ensures that, between two sensor readings, the increment of path delay will not cause an actual TDF.

However, before computing the new TIMER_VALUE (TEMP_S state), the new value of temperature is compared with the last measured temperature (TEMP_OLD). The case TEMP>TEMP_OLD represents the worst condition for the system, since an increase of temperature has immediate impact on the delay variation. The temperature sensor must therefore be activated at short intervals to guarantee proper adaptation of the system without incurring in transition delay faults. Differently, if TEMP<=TEMP_OLD, the delay increment can be only generated by aging effects, whose impact is in general much slower than temperature variation effects. Larger activation intervals for the temperature sensors can thus be setup. The SET_TIMER and SET_TIMER_AGING functions compute the new value of the timer based on the sign of ΔTemp=TEMP-TEMP_OLD and on the actual temperature value (i.e., TEMP in Fig. 6).

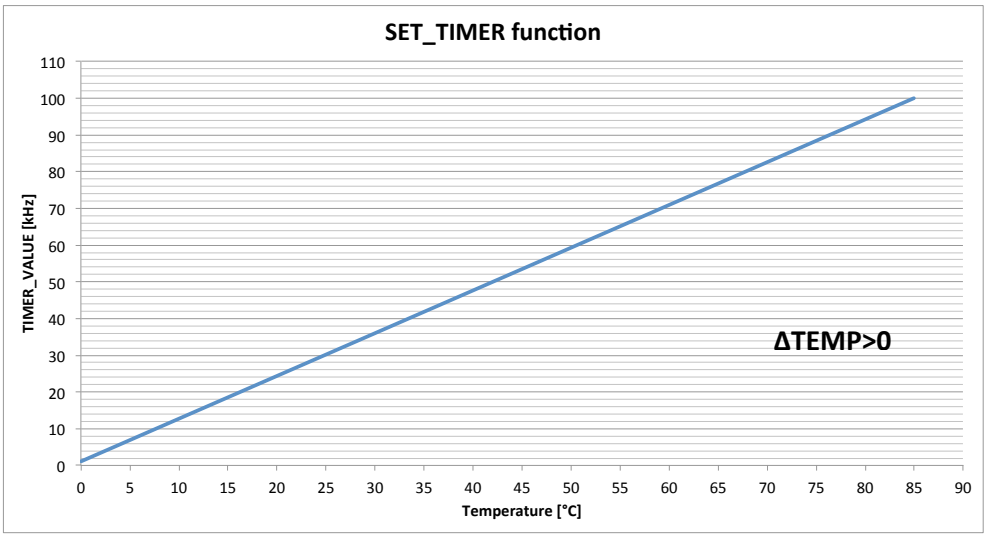
As aforementioned, the circuit temperature variation speed directly impacts on the timer sampling period. However, this parameter strongly depends on the device technology, on the circuit implemented in the FPGA device, and on the actual workload. To extract this parameter, the user should measure how the temperature profile of the circuit changes during a representative execution time interval [Happe et al. 2011]. If this information is not known, the designer can choose to adopt sampling periods that are orders of magnitude higher than the time constant with which the temperature changes (tenths of a second, or seconds, depending on the aforementioned parameters).

In particular, we adopt a linear function for both positive and negative ΔTemp (i.e., SET_TIMER and SET_TIMER_AGING functions), according to Fig. 7a and Fig. 7b.

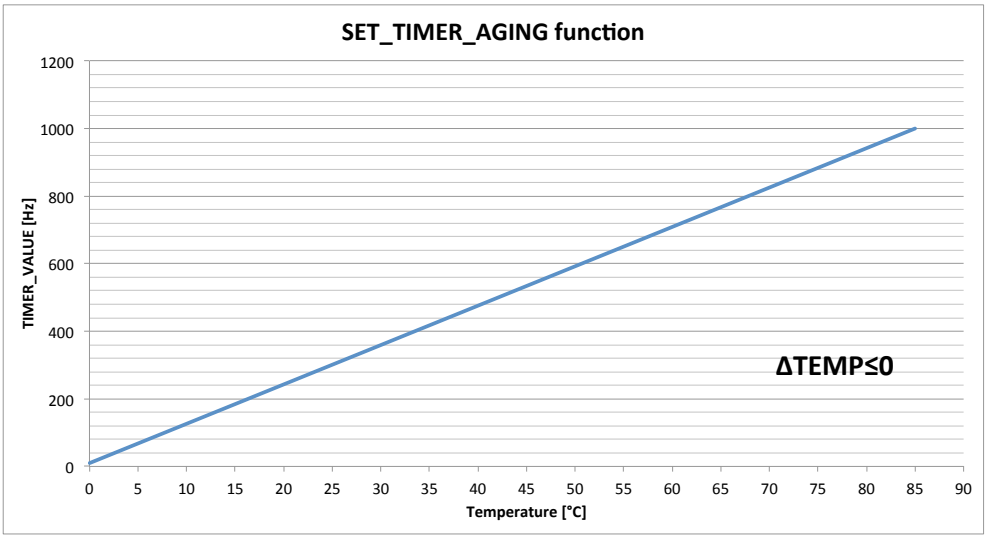
The difference between the two functions is the value of the sampling period (TIMER_VALUE), which spans, in the operating temperature range (0-85°C), from 1 kHz to 100 kHz and from 10 Hz to 1 kHz for positive and negative ΔTemp, respectively. In both cases, the higher is the temperature, the higher will be the sampling frequency (i.e., TIMER_VALUE). Moreover, the lower sampling frequency in the case ΔTemp<=0 leads to a saving in SATTA modules power consumption, with respect to the case ΔTemp>0.

After TIMER_VALUE computation, the manager enters the TDF_S state in which all TDF sensors are enabled. The outputs of all TDF sensors are OR-ed, and if at least one sensor issues a warning condition (DWFREQ_S state) the system's clock frequency (CLK_FREQ) is lowered by a given step (ΔCLK_DWFREQ) and the phase difference between the two clocks of the CLK generator (CLK_PHASE) is set to a predefined value (INIT_PHASE). A DPR command is therefore issued to the CLK Generator (CLK_DPR state) to change the operating frequency. When the configuration is complete, the CLK Generator multiplexers are switched (CLK_SWITCH state) and the *Manager* returns to the IDLE state, waiting for a new sampling cycle.

If no warning signal is raised, the *Manager* investigates the possibility of increasing the phase difference between the two clocks (i.e., the TDF sensor one and the standard one) in order to understand which is the maximum frequency the system can actually support. In this procedure, composed of the UPPHASE_S, the PHASE_SHIFT, and the PHASE_TDF states, the *Manager* tries to increase the phase difference between the two clocks generated by the *CLK Manager* up to a limit in which a warning is issued. This is obtained by iteratively increasing CLK_PHASE. In our specific Xilinx implementation of the *Manager*, the phase is increased of 1/256 of the actual system's



(a) SET_TIMER function



(b) SET_TIMER_AGING function

Fig. 7: Sampling periods with respect to temperature

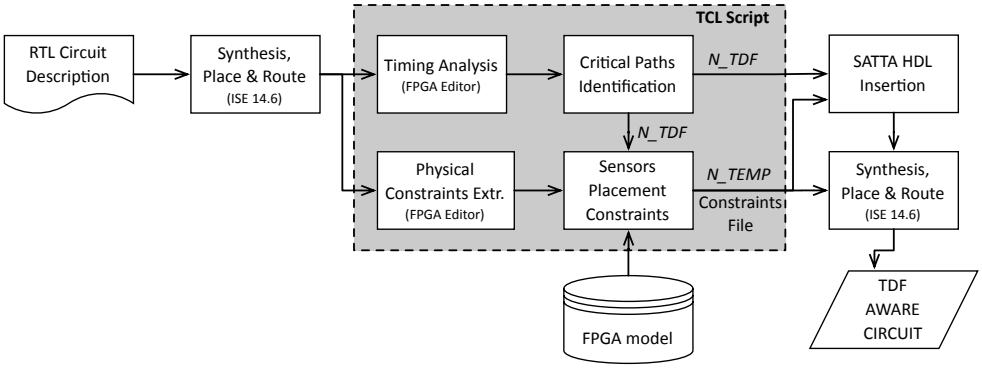


Fig. 8: SATTA integration flow. It enables to automate the insertion of all sensors and structures required to implement the TDF detection and compensation.

clock period at every iteration (it is the phase resolution limit of the *MMCM*). As soon as a warning signal is detected, in the PHASE_TDF state, the limit has been already exceeded, so the previous value of CLK_PHASE_STEP is used to modify the working frequency (UPFREQ_S state). The new boosted clock frequency is therefore computed according to the following equation:

$$CLK\_FREQ = \frac{CLK\_FREQ}{1 - \frac{CLK\_PHASE\_STEP}{256}} \tag{2}$$

The CLK_PHASE_STEP decrement (before leaving the PHASE_TDF state) is necessary in order to increase the working frequency in a dependable way, leaving a margin with respect to the actual maximum working frequency (Eq. 2).

4. SATTA INTEGRATION METHODOLOGY

In order to gather information and manage reconfigurations, SATTA needs to define a set of design rules to properly automate both sensors insertion and system integration. Fig. 8 highlights the overall SATTA integration flow.

SATTA integration starts from the synthesis and implementation of the RTL model of the target system given a target FPGA. After the placement and routing phase, each design path is analyzed in order to calculate the slack (i.e., the difference between the clock period and the actual path delay) using a timing analysis tool (e.g., *PlanAhead* or *FPGA Editor* tools for Xilinx FPGAs). For instance, the timing analyzer tool embedded in Xilinx PlanAhead can generate a text *timing report (.twr)* containing the slack information for each path with respect to the maximum allowable delay (i.e., the clock period). In particular, for each path are also listed the source and destination Flip-Flops (FFs). Fig. 9 shows a snippet of the timing analysis output file extracted using the Xilinx Timing Analyzer tool, highlighting the lines containing the slack values for the analyzed paths. It is worth to note that each destination FF (e.g., S_30 and S_reg29 in Fig. 9) can be associated to several design paths. In this case, paths with the same destination FF present different source-destination delays since they are sourced by different FFs (i.e., B_reg_4, B_reg_10 and A_reg_7 in Fig. 9) spread on the FPGA physical area (thus leading to different logic and routing delays).

A script can automatically parse the .twr file to extract all values contained in the *Slack (setup path)* lines, annotating also the source and the destination points, to iden-

```
Slack (setup path):     0.493ns (requirement - (data path - clock
path skew + uncertainty))
  Source:               B_reg_4 (FF)
  Destination:          S_30 (FF)
  Requirement:          5.500ns
  Data Path Delay:      4.912ns (Levels of Logic = 4)
  Clock Path Skew:      -0.060ns (0.831 - 0.891)
  Source Clock:         clk_BUFGP rising at 0.000ns
  Destination Clock:    clk_BUFGP rising at 5.500ns
  Clock Uncertainty:    0.035ns

Slack (setup path):     0.496ns (requirement - (data path - clock
path skew + uncertainty))
  Source:               B_reg_10 (FF)
  Destination:          S_30 (FF)
  Requirement:          5.500ns
  Data Path Delay:      4.909ns (Levels of Logic = 4)
  Clock Path Skew:      -0.060ns (0.831 - 0.891)
  Source Clock:         clk_BUFGP rising at 0.000ns
  Destination Clock:    clk_BUFGP rising at 5.500ns
  Clock Uncertainty:    0.035ns

Slack (setup path):     0.497ns (requirement - (data path - clock
path skew + uncertainty))
  Source:               A_reg_7 (FF)
  Destination:          S_30 (FF)
  Requirement:          5.500ns
  Data Path Delay:      4.903ns (Levels of Logic = 3)
  Clock Path Skew:      -0.065ns (0.831 - 0.896)
  Source Clock:         clk_BUFGP rising at 0.000ns
  Destination Clock:    clk_BUFGP rising at 5.500ns
  Clock Uncertainty:    0.035ns

                        ......
                        ......
                        ......

Slack (hold path):      0.484ns (requirement - (clock path skew +
uncertainty - data path))
  Source:               B_reg_25 (FF)
  Destination:          S_reg29 (FF)
  Requirement:          0.100ns
  Data Path Delay:      0.915ns (Levels of Logic = 2)
  Clock Path Skew:      0.496ns (2.000 - 1.504)
  Source Clock:         clk_BUFGP rising at 5.500ns
  Destination Clock:    clk1_BUFGP rising at 5.400ns
  Clock Uncertainty:    0.035ns
```

Fig. 9: Example of a timing analysis report generated using the Xilinx Timing Analyzer tool.

tify the related paths in the design. The lower is the slack, the more a path is critical. The selection of the most critical paths can be done by analyzing the distribution of the slack values with respect to the clock period (see Section 5 for some distribution examples and how critical paths can be selected). The number of identified critical paths (presenting different destination FFs) defines the number of TDF sensors needed to protect the circuit against transition delay faults (*N_TDF* value in Fig. 8).

Afterwards, physical placement and routing information of the implemented original design are extracted and stored as a user physical constraint file (.ucf). This task can be performed exploiting, for instance, the *Xilinx FPGA Editor* tool. This tool is able to read the implemented design netlist and to extract the placement information for each LUT and FF used in the FPGA device. Fig. 10 shows a snippet of the .ucf file, where each design resource is allocated to a LUT or a FF in a slice.

Xilinx FPGAs are arranged in columns and rows of Configurable Logic Blocks (CLBs), in which several slices are available. Each slice is numbered and uniquely identified by a couple of coordinates (X-Y) [Xilinx Corporation 2013b], and includes several LUTs and FFs. As an example, in Fig. 10, the A_reg_13 register is allocated in a FF inside the Slice X80Y91.

```
INST "ADDER1_gen[13].FAI/Co1_SW8" LOC=SLICE_X66Y80;
INST "ADDER1_gen[13].FAI/Co1_SW8" BEL="A5LUT";
INST "ADDER1_gen[13].FAI/Co1_SW2" LOC=SLICE_X66Y80;
INST "ADDER1_gen[13].FAI/Co1_SW2" BEL="A6LUT";
INST "ADDER1_gen[13].FAI/Co1_SW1" LOC=SLICE_X66Y80;
INST "ADDER1_gen[13].FAI/Co1_SW1" BEL="B6LUT";
INST "ADDER1_gen[13].FAI/Co1_SW7" LOC=SLICE_X67Y80;
INST "ADDER1_gen[13].FAI/Co1_SW7" BEL="B6LUT";
INST "A_reg_13" LOC=SLICE_X80Y91;
INST "A_reg_13" BEL="DFF";
INST "ADDER1_gen[13].FAI/Co1_SW5" LOC=SLICE_X81Y79;
INST "ADDER1_gen[13].FAI/Co1_SW5" BEL="B5LUT";
INST "ADDER1_gen[13].FAI/Co1_SW3" LOC=SLICE_X81Y79;
INST "ADDER1_gen[13].FAI/Co1_SW3" BEL="B6LUT";
INST "ADDER1_gen[13].FAI/Co1_SW6" LOC=SLICE_X81Y80;
INST "ADDER1_gen[13].FAI/Co1_SW6" BEL="B5LUT";
INST "ADDER1_gen[13].FAI/Co1_SW4" LOC=SLICE_X81Y80;
INST "ADDER1_gen[13].FAI/Co1_SW4" BEL="B6LUT";
INST "ADDER1_gen[13].FAI/Co1_SW0" LOC=SLICE_X81Y80;
INST "ADDER1_gen[13].FAI/Co1_SW0" BEL="C6LUT";

                                  ......
                                  ......
                                  ......
```

Fig. 10: Example of User Constraints File (.ucf) extracted running Xilinx FPGA Editor tool after design implementation.

This placement information is used in the last implementation phase to force the synthesis and implementation tools to keep the circuit implementation constant, thus avoiding changes in the path delays. The physical constraint file is also used, along with the *FPGA model* file, to extract (i) the number of temperature sensors (*N_TEMP* in Fig. 8), and (ii) placement constraints for both TDF and temperature sensors.

The *FPGA model* file contains information about the physical organization of Configurable Logic Blocks (CLBs) and slices inside a specific FPGA device. By merging the information of the FPGA slices organization and the one contained in the physical constraint file, it is possible to automatically search for free slices in which sensors can be placed. More specifically, to ensure good path delay monitoring capabilities, the *FF2* of a TDF sensor must be placed in a slice as near as possible to the original circuit FF (i.e., *FF1)* associated with the selected path.

Although temperature sensors should be placed as near as possible to those FFs, some considerations and optimizations can be done. In fact, the number of temperature sensors to insert in the circuit depends on relative distances between the TDF sensors inserted in the design, and on the target measure accuracy. However, as already mentioned in Section 3.1, SATTA does not need a highly accurate temperature measure. Thus, if two or more TDF sensors are physically located close to each other, they can be grouped together. A single temperature sensor can be employed to measure their temperature value, thus reducing the total number of required temperature sensors. This leads to a significant decrease of both power consumption and hardware overhead, wasted by the ring oscillators. As in [Happe et al. 2011], the maximum zone that a temperature sensor is charged to monitor is 10x10 FPGA CLBs. This means that, if one or more critical paths are enclosed in a 10x10 CLBs square, a single temperature sensor can be associated to them.

Once the number of required sensors and their associated placement constraints are computed, the HDL description of the circuit is modified to include SATTA modules and sensors. The different thresholds exploited by the *Manager* to perform its activities are selected by resorting to the knowledge of the circuit functionality (see Section 5). Finally, the modified circuit is re-synthesized, placed, and routed using the *Constraints file*, in order to obtain the final implementation of the TDF aware circuit. Obviously, the *Constraints file*, includes both sensors and original circuit implementation constraints.

In each step of Fig. 8, tools used when dealing with Xilinx FPGAs are listed in brackets. It is worth to note that all steps enclosed by the dashed line can be automated using a *tcl* script.

5. EXPERIMENTAL RESULTS

In order to evaluate the effectiveness of the proposed methodology, in both highly and lowly pipelined FPGA-based digital systems, SATTA has been applied on several circuit designs: (i) a LEON3-based SoC, including the LEON3 processor [Gaisler 2002] and several peripherals, such as memory controllers, UARTs, and debug units, (ii) an AES decoder core [AES 2009], and (iii) an AMBER a25 ARM-compatible processor [AMB 2013].

The LEON3-based SoC has been implemented on a Xilinx Virtex 6 device (*XC6VLX240T-1FF1156*), while the other circuits on a Virtex 7 FPGA device (*XC7VX485T-2FFG1761C*). Table I shows synthesis results of the stand-alone original designs, obtained using Xilinx Vivado™ Design Suite v2013.3.

Table I: Synthesis Results

	Look-Up Tables	*Registers*	*Crit. path delay*	*Max. frequency*
Leon3-based SoC	12,225	10,834	12.54 ns	79.74 MHz
AES	1,335	429	4.99 ns	200.40 MHz
AMBER	8,230	3,002	17.39 ns	57.50 MHz

Following the SATTA insertion methodology presented in Section 4, for each implemented circuit path delays and slack distributions have been extracted. Fig. 11 shows the slack distribution for each circuit. The figure reports, for each circuit, the number of paths (y axis) with a delay comprised within a given range with respect to the most critical path that sets the clock period (x axis). One can notice that few paths are very close to the clock period (range of 2% or 5%). Looking to these distributions, the user can choose how many paths can be considered as "critical", thus requiring the TDF sensor insertion. As an example, if a 5% range is considered, the designer assumes that, during the whole lifetime of the device, the maximum increase of delay, caused by both aging effects and temperature variations, on the unmonitored paths (i.e., paths with slack higher than 5%), never exceeds the delay of the most critical path of the circuit.

In the sequel, for the sake of brevity, experimental setup and simulation results will be given for the Leon3-based SoC design, only, while area overhead results, after applying the proposed methodology, will be provided for all the considered test cases.

By analyzing the slack distribution of the Leon3-based SoC design (Fig. 11), assuming that the circuit will operate for one year, aging phenomena will cause an increment of paths delay of no more than 2% [Srinivasan et al. 2008], and the device temperature variations will cause a maximum delay increment of 3% (e.g., from 25 °C to 75 °C following the model reported in [Pfeifer and Pliva 2013]). All paths whose slack is less or equal than 5% of the clock period are therefore good candidates to be monitored and protected against TDFs. In this case study, the number of chosen paths is equal to 15, while all the other paths are not monitored since TDFs are not likely to appear in such paths.

Afterwards, the original design is modified and the 15 TDF sensors are automatically inserted, replacing the final FFs of the selected critical paths. Fig. 12 shows that the optimal number of temperature sensors that should be inserted is equal to 3, since the 15 critical paths can be grouped into three proximity sub-sets (i.e., 10x10 CLBs square, as explained in Sec. 4). In Fig. 12, white arrows show critical paths logical connections, from the starting point to the endpoint, while yellow circles represent

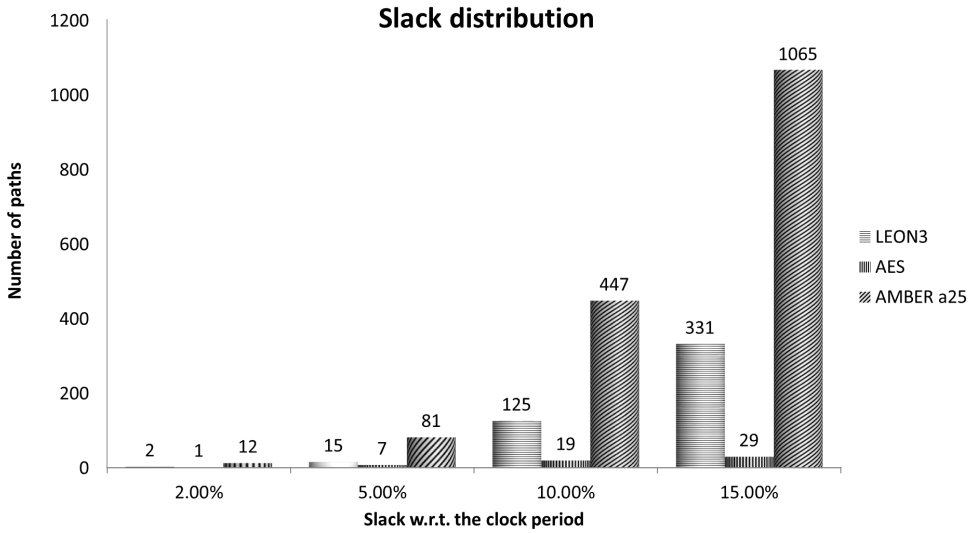


Fig. 11: Paths slack distribution. The y axis represents how may paths have a delay included in a range of the nominal clock period identified on the x axis.

the area in which each temperature sensor must be placed. In this case study the maximum group dimension (i.e., maximum distance between endpoints in terms of FPGA CLBs) is equal to 9 CLBs.

The initial system's frequency (INIT_FREQ) has been set at the reference value provided by the synthesis tool considering worst-case temperature and voltage conditions (i.e., 79.74 MHz). The negative phase relationship between CLK1 and CLK2 (i.e., INIT_PHASE), has been chosen as the minimum possible (i.e., 1/256) of the initial system clock period (i.e., 311 ns), while CLK_DWFREQ has been set equal to 1/256 of the synthesis frequency (i.e., 0.3 MHz), minimizing the performance loss when scaling down frequency.

Afterwards, the modified system has been synthesized and implemented, including all sensors and SATTA modules. Table II shows the area occupancy of the single blocks involved in the SATTA methodology.

Table II: SATTA synthesis results

	LUTs	*Registers*	*Block-RAMs*	*XADC*	*MMCM*	*BUFGMUX*
TDF sensor	2	2	-	-	-	-
CLK Generator	-	-	-	-	2	2
Temperature sensor	53	35	-	1	-	-
Manager	488	63	1	-	-	-

Fig. 13a to Fig. 13c show the resources overhead (in terms of LUTs and FFs) when applying SATTA to the five presented case studies, varying the selected slack threshold (i.e., 0%, 2%, and 5%). The x-axis also reports the number of required TDFs and temperature sensors. The number of TDF sensors is extracted by looking at Fig. 11,

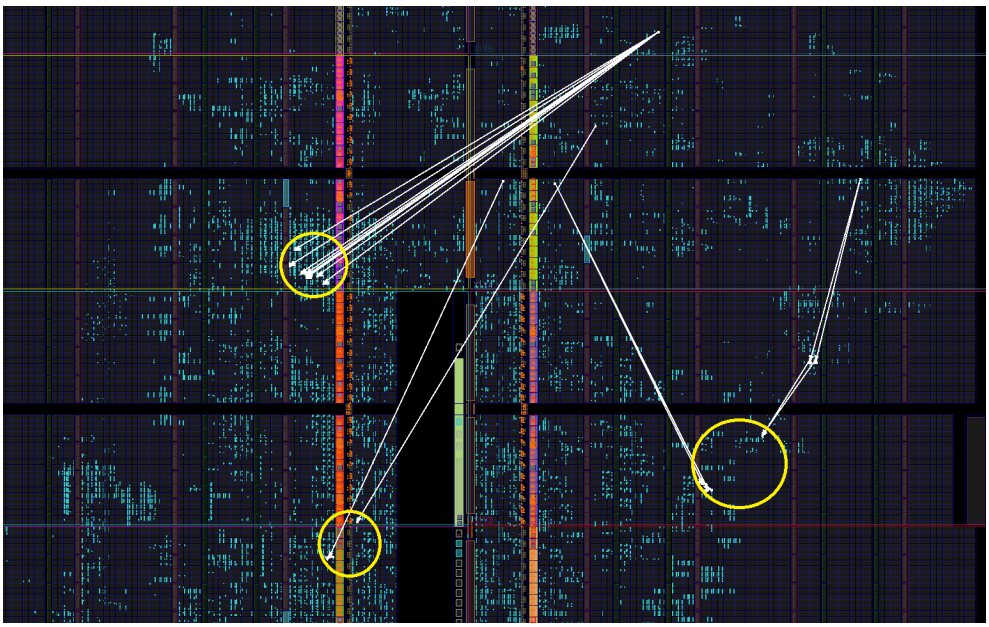


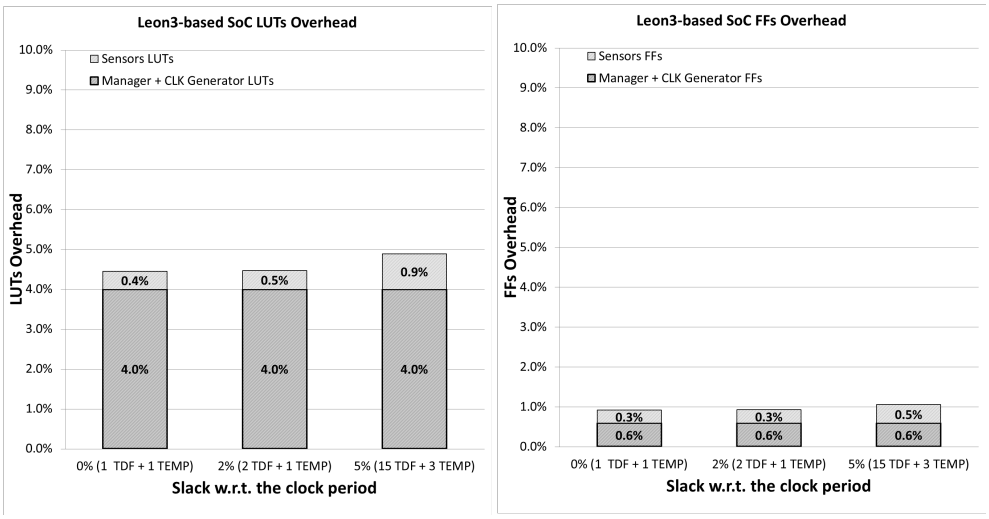
Fig. 12: Leon3-based SoC critical paths and temperature sensors floorplan

while te number of temperature sensors is extracted looking to the physical layout of the implemented circuit (as done in the Leon3-based SoC case).

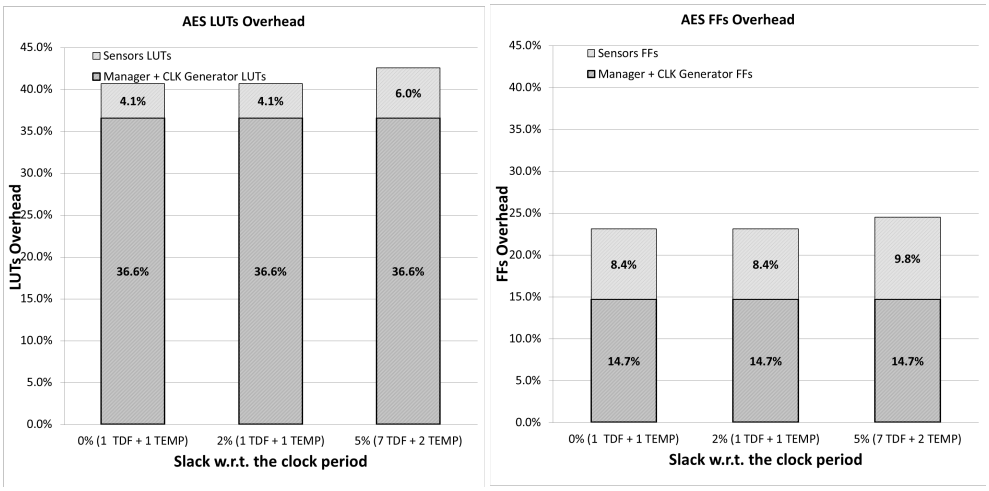
Looking at the results, one can notice that the area overhead introduced by the SATTA is in the range of 4-7% for large designs such as the Leon3-based SoC, while it becomes very large when applied to small modules such as the AES design. SATTA is therefore better suited for complex designs, like SoCs or processors.

The area overhead introduce by SATTA is due to two contributions highlighted with different textures in Fig. 13a to Fig. 13c: (i) a constant part represented by the *Manager* and *CLK Generator* and (ii) a design-dependent part, which strongly depends on the number of sensors to be inserted. As depicted in Fig. 13a to Fig. 13c, the variable hardware overhead can be reduced by selecting a lower threshold when looking to the paths slack distribution. It is worth noting that selecting a 0% slack threshold means that only the most critical path of the design is monitored.

Looking to the Leon3-based SoC case study, the insertion of SATTA modules causes an increase of resources usage of 4.9%, in terms of LUTs, and 1.2%, in terms of FFs, with a slack threshold equal to 5%. A single FPGA internal Block-RAM memory is used to store the functions implemented by the *Manager* as look-up tables. An Analog-to-Digital Converter (i.e., *XADC* FPGA internal hard macro) is also required to measure the temperature from the internal reference diode. Nonetheless, just one input channel is employed, leaving available other channels to be used for other purposes. The CLK Generator employs 2 MMCMs and 2 clock multiplexers (i.e., *BUFGMUX*), but the actual overhead is of 1 MMCM, since one is already present in the original design in order to provide the system clock signal. A small timing overhead on the monitored path delay is present as well. This overhead is due to the insertion of the TDF sensor (composed of two FFs, instead of one FF present in the original design) in the path. In this case-study, the insertion of the TDF sensor leads



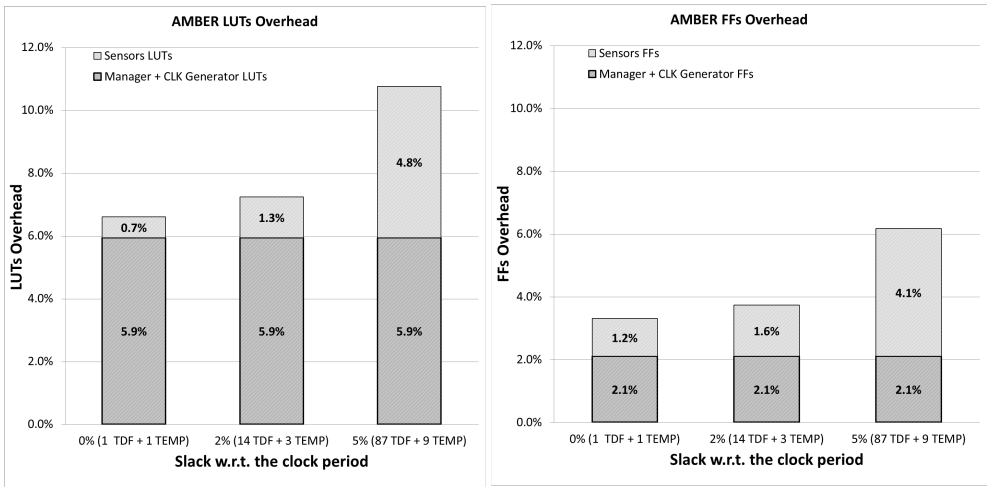
(a) SATTA LUTs and FFs overhead when applied to the Leon3-based SoC case study.



(b) SATTA LUTs and FFs overhead when applied to the AES case study.

Fig. 13: FPGA hardware overhead when applying SATTA to the five case studies.

to a 45 ps increase of the path delay, resulting in a minimal decrease of the maximum working frequency. However, this increase on the critical path delay is design and technology-dependent. Looking to the other case-studies, this increase attests in the range of 32-65 ps. Moreover, it is important to highlight that in all considered case studies, it was possible to insert the *FF2* of each TDF sensor into a CLB adjacent to the one containing the related *FF1* (as mentioned in Sec. 4).
Regarding power consumption, the initial unmodified design consumes 5.47 W. After the insertion of the SATTA modules, there is a power increase of 124 mW, mainly due to the additional *MMCM* hard macro and to the additional clock net, while the



(c) SATTA LUTs and FFs overhead when applied to the AMBER case study.

Fig. 13: FPGA hardware overhead when applying SATTA to the five case studies.

additional logic's power waste is negligible. This leads to a power overhead of 2.2% with respect to the unmodified design. It is worth noting that a power comparison with [Valdes-Pena et al. 2013], which implemented a TDF sensor along with the clock generation network, is not possible since the used technologies are quite different. The Xilinx Virtex 6 and Virtex 7 FPGA devices used in this work are built on a 40 nm and 28 nm process technology, respectively, while the Xilinx Spartan-3AN device used in [Valdes-Pena et al. 2013] is built on a 90 nm process technology.

To verify the proper working of the aging manager in the system, the post place and route design (generated by the synthesis tool) has been simulated using Modelsim® SE 10.0c. The effects of temperature and aging on path delays have been emulated inserting parametric delays in the post place and route HDL simulation model. Delays are automatically read from a text file at different times. These delays increase during the execution time, thus emulating the effects of temperature variations. The higher the temperature is, the greater is the delay, following the models reported in [Pfeifer and Pliva 2012; 2013], in which the path delay increases linearly of 5% when spanning the full operating temperature range (0-85 °C). Fig. 14 plots the operating frequency profile when the temperature profile changes over time.

In Fig. 14 the temperature increases linearly from 25 °C to 85 °C in 45 seconds. Then, after a constant period, it decreases linearly and, finally, it remains stable around 50 °C. At power-up, the working frequency quickly grows compared to its initial value (*Synthesis Frequency* in Fig. 14) since the real temperature profile decreases the path delay, that was calculated in worst-case conditions during the synthesis process. This therefore improves the system's performance. When the temperature starts to increase, the working frequency is adapted by the *Manager*, decreasing its value by CLK_DWFREQ when a warning signal is raised. Obviously, when the temperature reaches the worst case condition (i.e., 85 °C), the operating frequency approaches the synthesis frequency. A soon as the worst case is reached, the working frequency remains stable since, even if no TDF warnings are detected, the *Manager* is not able to increase the frequency. This is due to the fact that the *Manager* recognizes a warning signal after a single CLK_PHASE increment (see Fig. 6).

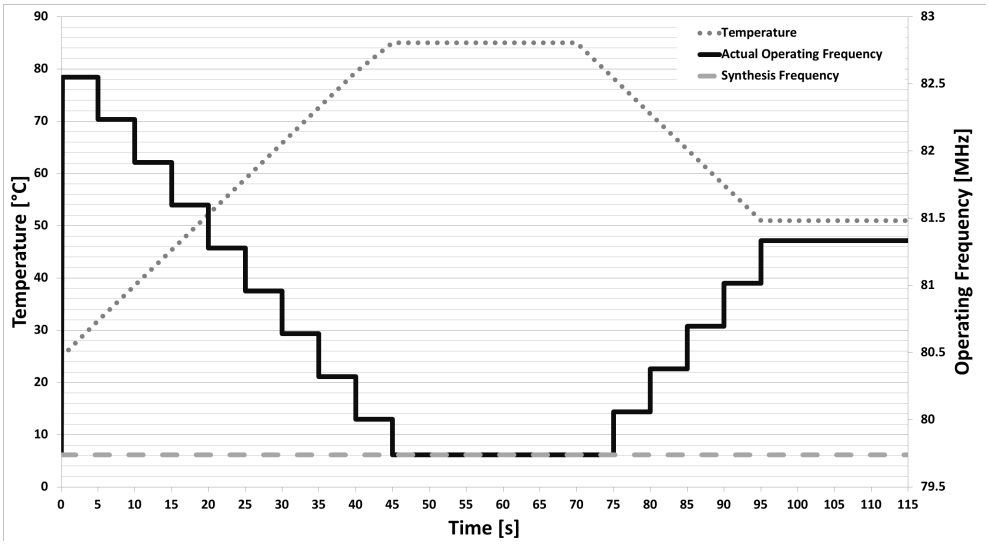


Fig. 14: Simulated operating frequency trend with temperature variations

Whenever the temperature decreases (from 70 seconds in Fig. 14), the working frequency is then adapted following the behaviour explained in Sec. 3.4.

6. CONCLUSION

This paper presented SATTA, a Self-Adaptive Temperature-based TDF Awareness methodology to compensate for path delay variations due to both variation of temperature and aging effects in comps SoPCs. SATTA main contribution is the SATTA manager that, supported by a reconfigurable and efficient clock manager module, is able at run-time to optimize the system frequency avoiding TDFs while optimizing the performance. Moreover, the manager optimizes the use of all SATTA sensors in order to minimize their aging.

The application of SATTA on a set of real designs clearly proved the effectiveness of this approach especially when applied to large real designs that incur in very low overheads.

Eventually, the full SATTA design pipeline has been easily automated by means of a set of TCL scripts. This represents an additional and valuable result for the easy exploitation of this methodology in real design flows.

REFERENCES

2009. AES (Rijndael). (2009). http://opencores.org/project,rijndael

2013. Amber ARM-compatible core. (2013). http://opencores.org/project,amber

Altera Corporation. 2012. *Stratix V Device Overview* (sv51001 ed.).

Altera Corporation. 2013. *The Breakthrough Advantage for FPGAs with Tri-Gate Technology* (wp435wp-01201-1.0 ed.).

Abdulazim Amouri and Mehdi Tahoori. 2011. A Low-Cost Sensor for Aging and Late Transitions Detection in Modern FPGAs. In *Field Programmable Logic and Applications (FPL), 2011 International Conference on*. 329–335. DOI:http://dx.doi.org/10.1109/FPL.2011.66

A. Amouri and M. Tahoori. 2012. High-level aging estimation for FPGA-mapped designs. In *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*. 284–291. DOI:http://dx.doi.org/10.1109/FPL.2012.6339194

V. Bexiga, C. Leong, J. Semiao, I.C. Teixeira, J.P. Teixeira, M. Valdes, J. Freijedo, J.J. Rodriguez-Andina, and F. Vargas. 2011. Performance Failure Prediction Using Built-In Delay Sensors in FPGAs. In *Field Programmable Logic and Applications (FPL), 2011 International Conference on*. 301–304. DOI:http://dx.doi.org/10.1109/FPL.2011.61

Jung-Hwan Choi, Jayathi Murthy, and K. Roy. 2007. The effect of process variation on device temperature in finFET circuits. In *Computer-Aided Design, 2007. ICCAD 2007. IEEE/ACM International Conference on*. 747–751. DOI:http://dx.doi.org/10.1109/ICCAD.2007.4397355

S. Das, D. Roberts, Seokwoo Lee, S. Pant, D Blaauw, T. Austin, K. Flautner, and T. Mudge. 2006. A self-tuning DVS processor using delay-error detection and correction. *Solid-State Circuits, IEEE Journal of* 41, 4 (2006), 792–804. DOI:http://dx.doi.org/10.1109/JSSC.2006.870912

Shidhartha Das, Carlos Tokunaga, Sanjay Pant, Wei-Hsiang Ma, Sudherssen Kalaiselvan, Kevin Lai, David M. Bull, and David T. Blaauw. 2009. RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance. *Solid-State Circuits, IEEE Journal of* 44, 1 (January 2009), 32–48. DOI:http://dx.doi.org/10.1109/JSSC.2008.2007145

Stefano Di Carlo, Salvatore Galfano, Giulio Gambardella, Daniele Rolfo, Paolo Prinetto, and Pascal Trotta. 2012. NBTI Mitigation by Dynamic Partial Reconfiguration. In *Electronics Conference (BEC), 2012 13th Biennial Baltic*. 93–96. DOI:http://dx.doi.org/10.1109/BEC.2012.6376823

M. Fukazawa, M. Kurimoto, R. Akiyama, H. Takata, and Makoto Nagata. 2008. Experimental evaluation of digital-circuit susceptibility to voltage variation in dynamic frequency scaling. In *VLSI Circuits, 2008 IEEE Symposium on*. 150–151. DOI:http://dx.doi.org/10.1109/VLSIC.2008.4585986

H. Fuketa, M. Hashimoto, Y. Mitsuyama, and T. Onoye. 2012. Adaptive Performance Compensation With In-Situ Timing Error Predictive Sensors for Subthreshold Circuits. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 20, 2 (2012), 333–343. DOI:http://dx.doi.org/10.1109/TVLSI.2010.2101089

J. Gaisler. 2002. A portable and fault-tolerant microprocessor based on the SPARC v8 architecture. In *Dependable Systems and Networks (DSN), 2002. International Conference on*. 409–415. DOI:http://dx.doi.org/10.1109/DSN.2002.1028926

Markus Happe, Andreas Agne, and Christian Plessl. 2011. Measuring and Predicting Temperature Distributions on FPGAs at Run-Time. In *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*. 55–60. DOI:http://dx.doi.org/10.1109/ReConFig.2011.59

Toshihiro Kameda, Hiroaki Konoura, Dawood Alnajjar, Yukio Mitsuyama, Masanori Hashimoto, and Takao Onoye. 2012. A predictive delay fault avoidance scheme for coarse-grained reconfigurable architecture. In *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*. 615–618. DOI:http://dx.doi.org/10.1109/FPL.2012.6339220

Jie Li and J. Lach. 2007. Negative-skewed shadow registers for at-speed delay variation characterization. In *Computer Design, 2007. ICCD 2007. 25th International Conference on*. 354–359. DOI:http://dx.doi.org/10.1109/ICCD.2007.4601924

Sang Phill Park, Kunhyuk Kang, and Kaushik Roy. 2009. Reliability Implications of Bias-Temperature Instability in Digital ICs. *Design & Test of Computers, IEEE* 26, 6 (November 2009), 8–17. DOI:http://dx.doi.org/10.1109/MDT.2009.154

P. Pfeifer and Z. Pliva. 2012. On measurement of impact of the metallization and FPGA design to the changes of slice parameters and generation of delay faults. In *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*. 743–746. DOI:http://dx.doi.org/10.1109/FPL.2012.6339167

Petr Pfeifer and Zdenek Pliva. 2013. On measurement of parameters of programmable microelectronic nanostructures under accelerating extreme conditions (Xilinx 28nm XC7Z020 Zynq FPGA). In *Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on*. 1–4. DOI:http://dx.doi.org/10.1109/FPL.2013.6645584

T. Sato and Y. Kunitake. 2007. A Simple Flip-Flop Circuit for Typical-Case Designs for DFM. In *Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on*. 539–544. DOI:http://dx.doi.org/10.1109/ISQED.2007.23

Suresh Srinivasan, Ramakrishnan Krishnan, Prasanth Mangalagiri, Yuan Xie, Vijaykrishnan Narayanan, Mary Jane Irwin, and Karthik Sarpatwari. 2008. Toward Increasing FPGA Lifetime. *Dependable and Secure Computing, IEEE Transactions on* 5, 2 (April 2008), 115–127. DOI:http://dx.doi.org/10.1109/TDSC.2007.70235

Edward A. Stott, Justin S.J. Wong, Pete Sedcole, and Peter Y.K. Cheung. 2010. Degradation in FPGAs: measurement and modelling. In *Proceedings of the 18th annual ACM/SIGDA international*

symposium on Field programmable gate arrays (FPGA '10). ACM, New York, NY, USA, 229–238. DOI:http://dx.doi.org/10.1145/1723112.1723152

M.D. Valdes-Pena, J. Fernandez Freijedo, M.J.M. Rodriguez, J.J. Rodriguez-Andina, J. Semiao, I.M. Cacho Teixeira, J.P. Cacho Teixeira, and F. Vargas. 2013. Design and Validation of Configurable Online Aging Sensors in Nanometer-Scale FPGAs. *Nanotechnology, IEEE Transactions on* 12, 4 (2013), 508–517. DOI:http://dx.doi.org/10.1109/TNANO.2013.2253795

Xilinx Corporation. 2012. *7 Series FPGAs Overview* (ds180 (v1.14) ed.).

Xilinx Corporation. 2013a. *7 Series FPGAs Clocking Resources* (ug472 (v1.8) ed.).

Xilinx Corporation. 2013b. *7 Series FPGAs Configurable Logic Block* (ug474 (v1.5) ed.).

Xilinx Corporation. 2013c. *Xilinx UltraScale:The Next-Generation Architecture for Your Next-Generation Architecture* (wp435 ed.).

Kenneth M. Zick and John P. Hayes. 2010. On-line sensing for healthier FPGA systems. In *Proceedings of the 18th annual ACM/SIGDA international symposium on Field programmable gate arrays (FPGA '10)*. ACM, New York, NY, USA, 239–248. DOI:http://dx.doi.org/10.1145/1723112.1723153