An Approach for Efficient Metastability Characterization of FPGAs through the Designer

Thomas Polzer and Andreas Steininger
Institute of Computer Engineering
Vienna University of Technology
{*tpolzer,steininger*}*@ecs.tuwien.ac.at*

***Abstract*—The efficient design of a synchronizer for a given MTBF limit heavily depends on the availability of an accurate metastability characterization of the bistables in the target technology. We propose a measurement approach for FPGAs that comes along without any specific measurement infrastructure and can hence be performed by the designer with relatively low efforts, but is yet very accurate. Our concept comprises the use of the FPGA-internal digital clock manager (DCM), calibration measurements for the latter, averaging over several parallel measurement runs, and separated analysis of different metastability cases. To demonstrate the power of our approach we present detailed measurement results for a Xilinx Virtex-4 FPGA that even show slave metastability. Furthermore, we discuss how diverse constraints can be considered to make the measurement more accurate and time efficient.**[1]

***Keywords*-FPGA; clock-to-output delay; metastability; measurement;**

I. Introduction

The complex designs, that the tremendous progress in VLSI technology has enabled us to fit onto a single die nowadays, are usually structured into relatively self-contained functional blocks. For various reasons those blocks are most often supplied by different independent clocks. This results in the existence of multiple timing domains within a system-on-chip (SoC), with a strong need for communication among them. Unless carefully synchronized [1], this communication may suffer from reliability problems due to sporadic metastable upsets of bistable elements at the timing domain interfaces.

Many of the techniques known to mitigate this problem rely on the use of synchronizers. Given that the complexity of SoCs in terms of their function blocks and timing domains is still growing and the communication demands are increasing, there is a trend towards a rising number of synchronizers per design [2]. Also there seems to be an upcoming need for deeper synchronizer pipelines (i.e. containing more stages) [3]. At the same time synchronizers are in general well known to incur performance penalties (unless specific properties of the clocks can be exploited, as in [4]), therefore their design should ideally be just conservative enough to guarantee safe operation, but not overly pessimistic. After all, communication is considered the predominant bottleneck in contemporary and future SoCs. This, however, requires a precise metastability characterization of the bistable elements being prone to metastability or being used for building a synchronizer. Once provided with the relevant metastability parameters, a designer can apply existing models to determine the mean time between failure (MTBF) of his synchronizer and optimize the design appropriately.

Metastability characterization, however, is a very sophisticated, mostly experimental process; it involves many sources of error, and the result heavily depends on the prevalent experimental conditions, such as routing delays, output load, voltage and temperature, that cannot always be controlled precisely enough at the very location of the element under test. This is, unfortunately, especially true for FPGAs which, due to their considerable performance and available complexity, receive increasing interest for implementing SoCs, not only for prototypes but also for products – recently even in safety-relevant embedded applications like in the automotive domain. This, of course, pronounces the need for a reliable characterization of their metastable behavior.

In the first place one would expect the FPGA vendors to provide the relevant parameters in the data sheets. However, this is true for some vendors only, and even if so they may not have satisfactory quality. In some cases the designer is provided with tools to estimate the MTBF of his circuit, however, without any possibility to verify those predictions. The designer of a critical application would expect to find a worst case / best case range or a characterization over voltage and temperature, but instead a single value is often provided without further details. Assuming this is a worst case value has allowed designers to sufficiently overdesign their system's MTBF so far, accepting an associated performance penalty. An optimization of the synchronizer, as it seems desirable for future SoCs, will definitely require a more detailed metastability characterization.

As already mentioned the situation is specifically bad for FPGAs, as the vendor cannot know in advance the relevant operating conditions of the cell in question. Also, in the nanoscale technologies used for modern FPGAs, process variations play a significant role. In this paper we therefore

[1]This work is supported by the Austrian Science Foundation (FWF, P21694).

propose an efficient approach for metastability characterization in modern FPGAs that can be performed by the designer with relatively low efforts and, most importantly, without any specific external high-resolution measurement equipment like GHz bandwidth oscilloscope or ps-jitter clock generator. After a review of related work in Section II we will present the principle of our approach as well as its detailed implementation in Sections III and IV, respectively. In Section V we will demonstrate the applicability of our approach at the example of a Xilinx Virtex-4 FPGA. Section VI will be devoted to a discussion of specific constraints that need to be considered for an efficient implementation. Finally, Section VII will conclude the paper.

II. State of the Art

A. Metastability Modeling

Since its first mentioning by I. Catt in 1966 [5], much research has been devoted to better understanding and characterizing metastability (see e.g. [6], or [7], [2]). In a nutshell, a flip flop (FF) output may become metastable, if its data input changes in close proximity to the active clock edge. The question whether the data changed before or after the clock edge becomes undecidable within bounded time then [8]. Although other behaviors of the FF output ("creeping" between HI and LO, oscillation) are possible, the most common and thus relevant manifestations of metastability on the cell level are either glitches or *late transition*. The latter involves a proper transition of the FF output, however, with a clock-to-output delay that is larger than the nominal value and essentially unbounded.

Circuit level simulators, such as SPICE have been successfully employed to observe the metastable behavior of an element [9], [10], [11]. While they allow convenient access without probing effects, they use idealized models that may not appropriately reflect the actual circuit parameters, and they tend to suffer from numerical problems and high sensitivity to even small changes in the chosen parameters and models [12].

Based on linearized models for the inverter loops forming the storage cell inside a latch, augmented with an RC-element to approximate their timing behavior, analytic models have been derived for (the creeping manifestation of) metastability [13], [14]. Under the assumption of a data input signal whose transitions exhibit a uniformly distributed phase relation to the active clock edge, the MTBF can be predicted as [14]:

$$MTBF = \frac{1}{T_W f_{clk} \lambda_{dat}} \exp\left(\frac{t_{res}}{\tau}\right) \qquad (1)$$

This equation describes the MTBF as the expected interval between two consecutive metastable upsets of a FF that samples with its clock frequency f_{clk} a data input whose logic state changes at a rate λ_{dat}. An upset resembles the case that the FF's output has become metastable and not recovered from metastability before it is sampled by a successor stage (i.e., in fact late transition). The time available for this recovery is t_{res}. T_W and τ are the relevant metastability characteristics of the flip flop and as such dependent on circuit design and technology. They are the data sheet values the designer needs for designing a synchronizer – in fact by using Equation (1). Therefore experimental measurement setups are most often targeted to the determination of T_W and τ.

B. Metastability Measurement

For an experimental parameter determination it is natural to engage the following components:

- The device under test (or short DUT).
- A mechanism to create metastable upsets within the DUT.
- A mechanism to evaluate some characteristics (length, count or shape) of the metastable upsets.

1) Metastability Generation: Metastability generation involves producing a transition of the data input inside the critical time window around the active clock edge. This can be done either randomly or deterministically.

In the *random approach* two free running, independent oscillators are employed for creating the data- and the clock signal for the DUT. Based on the independence of these two signals, their phase difference is uniformly distributed over the clock period [15]. Therefore from time to time a data transition will occur within the critical window and a metastable state of the DUT can be observed. The uniform phase distribution perfectly matches the assumption made for Equation (1), which makes this method attractive for metastability characterization. Its drawback is that, since the critical window is typically considerably smaller than the clock period, chances of producing a metastable upset are low, which increases measurement times.

In the *deterministic approach* the phase shift between clock and the data signals is carefully controlled to produce a maximum number of metastable upsets. Due to the influence of jitter, noise, voltage and parameter variations, etc. this is a very delicate task that requires special provisions like e.g. using a delay locked loop [16], [17]. The obvious advantage of this method is to generate a relatively high yield of (even deep) metastable upsets, which allows their detailed study. The correlation between clock and data transitions established by the phase control, however, rules out a characterization based on the application of Equation (1).

2) Metastability Detection: Again two basic methods exist here. For the *oscilloscope based approach* [18], [15], [16], [19] the DUT output signal is routed to a pin, using a (preferably analog) output buffer. The signal is recorded by an oscilloscope and analyzed in the analog domain.

The *late transition detection method* is illustrated in Figure 1: The output of the DUT is sampled (i) by a

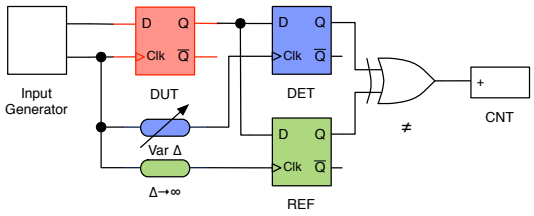

Figure 1. Digital Metastability Detection Circuit: Principle

detector flip flop (DET)[2] after a well controlled delay $var\Delta$ that determines the resolution time t_{res} and (ii) by another flip flop (REF) that samples the DUT output after (ideally) infinite delay ($\Delta \rightarrow \infty$). Since the latter sample represents a reference for the correct, stable DUT output, a comparison of the two samples indicates whether the DUT output had already changed before being sampled by DET after $var\Delta$. Of course, infinite delay for REF is not feasible in practice; a full or a half period of the clock is often used instead, which is sufficient with a carefully chosen clock frequency. Finally a counter can be used to determine the number of mismatches experienced in the measurement interval.

One benefit of the on-chip late transition detection method is that the actual threshold of a circuit element from the same technology is used to classify the DUT response as being HI or LO, and not an artificial one as in the case of an oscilloscope measurement. Another advantage is that *all* events are registered and that there is no blind time interval in the measurement period. On the downside, however, it does neither allow a detailed study of individual upsets nor the direct measurement of signal shapes or actual resolution times, as the oscilloscope method does. In fact one can only tell how often the actual resolution time has been larger than the one allowed by the choice of $var\Delta$. By varying $var\Delta$ for repeated runs of the experiment a cumulative plot of upsets over $var\Delta$ can be generated.

As this circuit is very compact and does, unlike the oscilloscope based approach, not require an analog amplifier, it is often used in ASICs and FPGAs [20], [21], [22], [23], [3]. A challenging problem, however, is the implementation of the variable delay $var\Delta$. Obviously the precision of this delay (in terms of resolution, accuracy and jitter) is crucial to the accuracy of the measured parameters.

In an FPGA analog delay lines [24], [25] and starved inverters [26], as they have been used to implement delay lines in ASICs, are simply not available. Another problem is the notorious uncertainty of routing delays within FPGAs. Not only are the delays of different paths within the FPGA subject to PVT variations, also the place and route tools use heuristics to generate the routed design. This makes the application of tapped delay lines at best problematic. Also, approaches taking advantage of differences in routing delays, like [22], [27] suffer from that problem. They require manual routing, which makes the delay generation process tedious and impossible to automate. Alternatively, one might consider using cell delays to implement $var\Delta$. In FPGAs, however, even simple inverters, are mapped to a LUT, which limits the attainable temporal resolution to the ns-range.

New FPGA families increasingly offer support for clock management, like phase-locked loops (PLL), or digital clock management (DCM). In [3] these are leveraged for implementing variable delays as well. Engineers have also proposed using the falling clock edge for DET, which allows a variation of t_{res} by virtue of the duty cycle or clock frequency [28].

III. Overall Design Concept

Our aim was to devise a setup for metastability characterization of an FPGA that does not require sophisticated external equipment (just a PC for the data processing), but still attains good accuracy for all relevant parameters and effects within reasonable measurement time. This determined the choice of the methods: For metastability generation we use random clock generation, as we do not want do depend on a two channel clock generator with precise phase alignment (to be useful for the purpose the phase resolution must be in the sub-ps range). To save the need for a high bandwidth oscilloscope along with the availability of analog output pins on the target, we decided for late transition detection (LTD). Based on these decisions it is possible to host the complete infrastructure on chip; this has been shown in the literature already. There is still a lot of conceptual work and tuning required to attain sufficient accuracy and observe all effects of interest. Before going into detail with our respective circuit design, we will briefly present the cornerstones of our approach:

A. Consideration of Slave Metastability

State of the art FPGA measurement circuits normally measure the response of the master latch only [27], [3]. In modern flip flops, however, τ of the slave latch may be significantly worse than the one of the master latch [10]. If synchronizers are designed based on the master τ only, the calculated MTBF may be too optimistic. Therefore we consider the measurement of the slave latch of vital importance. The only approach we know of, that characterizes the slave latch for FPGAs is [28]. However, in this approach, the latches are characterized using a simulation model and later it is tried to correlate these result to the real hardware using an LTD to measure the metastable response of the DUT's master latch. Our approach, in contrast, directly measures the slave's metastable response in hardware.

For an efficient measurement of slave metastability we need to vary the resolution time of the master latch towards

[2]Note that in principle DET can also get metastable if the DUT output transition coincides with its critical window, which is why it is safer to append another stage.

very low values. This makes it less likely for a metastable master output to resolve before being captured by the slave upon the falling clock edge. When using a clock with 50% duty cycle, this might be achieved by increasing the clock frequency. This, however, at the same time reduces the resolution time for the reference signal as well as requires the measurement electronics to work faster. As in most cases the speed of the measurement circuit is already at its limits, this is not an option. Instead we propose a scheme to create shorter clock pulses out of ordinary 50%-duty cycle clock signals, such that the clock period stays as long as necessary to avoid the above issues, while still attaining low and variable resolution time for the master latch. A respective pulse generation circuit that allows well controlled variation of the clock HI-time will be detailed in Section IV-B.

B. Case Separation

CMOS circuits are known to exhibit significantly different timing properties for rising and falling edges, so one can expect the same for metastability parameters. Furthermore, one possible manifestation of metastability that occurs jointly with late transitions is the occurrence of glitches. The LTD schemes proposed so far do not account for these facts – they only record the overall metastable behavior without further distinction. The characterization obtained that way is sufficient for standard synchronization problems. Note, however, that knowledge about different parameter values for rising and falling edges may be beneficial, if a data event of interest is always associated with a certain polarity (like, e.g., an arriving interrupt), while the other edge is not relevant or becomes masked. In the same way knowledge about the existence and polarity of glitches at a synchronizer output may be relevant, if the subsequent logic cannot properly handle those.

In fact it is relatively easy to implement additional bookkeeping that allows to distinguish the cases listed in Table I. In addition to detecting a mismatch between detector and reference we just need to keep track of the current and the previous reference value. This enables a much more detailed analysis of the metastability behavior.

Table I
METASTABLE CASES THE CAN BE DISTINGUISHED

case	start from (ref_last)	resolved to (ref_cur)	possible metastable effect
overall	any	any	overall # of metastable upsets
$from\_0$	LO	any	late rising edge, positive glitch
$from\_1$	HI	any	late falling edge, negative glitch
to_0	any	LO	late falling edge, negative glitch
to_1	any	HI	late rising edge, positive glitch
0_to_1	LO	HI	late rising edge
1_to_0	HI	LO	late falling edge
0_to_0	LO	LO	positive glitch
1_to_1	HI	HI	negative glitch

For a better understanding of Table I consider the case 0_to_0. If the current as well as the previous reference value indicate a LO, while the detector flip flop indicates that the DUT (at the sampling point given by the resolution time) issued a HI, this means the DUT output has exhibited a positive glitch ($0 \rightarrow 1 \rightarrow 0$). Such a glitch may well be seen in reality, if the output buffer (inverter) of the DUT has a high threshold (similarly, for low input threshold a negative glitch will be seen at the inverter output). In the conventional setup, when scanning with different resolution times, such a glitch will first be regarded as correct result and then, as resolution time is increased, as late transition, thus distorting the results (see Section V).

C. Calibration Runs

The accuracy of the measurement is, among other factors, limited by the accuracy of the variable delay elements as well as by the delays along some selected routing paths. Unfortunately interconnect delays of FPGAs are specifically pronounced and hard to control or determine. To tackle this problem we propose a calibration run in which we make the DUT behave in a well defined way (i.e. with no metastability in the game) and calibrate our on-chip measurement infrastructure so as to properly reflect this behavior. For details see Section IV-C.

D. Averaging over Results

Since the time resolution we are aiming at is right at the limit of what can be attained with the available infrastructure, we have to consider several sources of error, partly of systematic and partly of statistical nature[3]. We will take care of some systematic errors in Section VI. For statistical errors averaging is known to be an effective remedy. Since the required measurement circuit is typically much smaller than what the modern target FPGAs can accommodate, we propose to replicate circuitry, such that multiple measurements can be performed in parallel and thus without a penalty on measurement time.

IV. MEASUREMENT CIRCUIT

In the this section we will discuss the components constituting our setup in more detail. Please note that the circuits shown subsequently are simplified to the basic concepts, while details like non essential synchronizers, pipeline flip flop to increase the achievable clock frequency and special purpose buffers (like clock buffers) are omitted to improve readability.

A. Delay Generation

Before discussing the basic blocks of our setup, we will first focus on a function block that is used in several of them, namely a delay element. As already outlined in Section II it

[3]We consider an effect systematic, if the change it causes on the results can be reproduced in subsequent measurements, and statistical, if the changes caused by it significantly vary throughout multiple measurements.

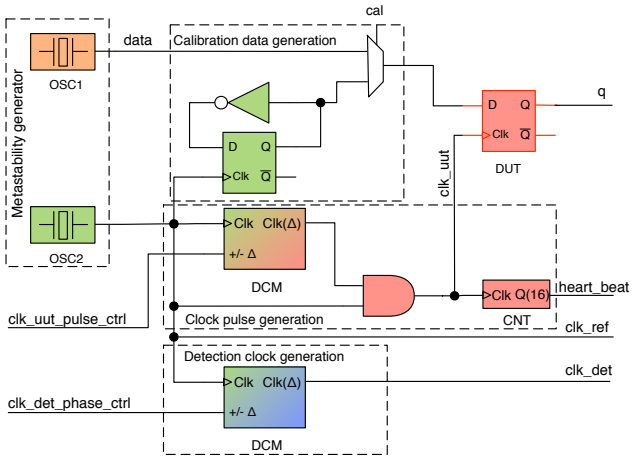

Figure 2. Metastability and clock generation circuit

is not trivial to implement a sufficiently accurate adjustable delay within an FPGA. Our solution relies on delay locked loops (DLLs), which are readily available on most modern FPGAs. They accept a reference clock at their input and supply an output signal that is delayed against that input by an adjustable phase angle. The adjustment is possible in steps, under the control of increment and decrement ports. The digital clock manager (DCM) of our Xilinx Virtex-4 target FPGA provides such a function, with a dynamic range of two clock cycles and a resolution of 256 steps per cycle. With a reference clock of 400MHz this results in a step size of 9.77 ps.

B. Metastability Generation

In accordance with our decisions from Section III we use two unrelated clock sources for generating data and clock inputs of the DUT as shown in Figure 2. The clock input is not directly connected but through a pulse generation unit that controls the duty cycle. This implements our requirement from Section III-A aiming at efficient measurement of slave metastability. As visible in Figure 2 (block *clock pulse generation*) we perform a logical AND of the original clock with a delayed version (using the DCM). This solution allows us to adjust the HI-time of the clock in steps of 9.77 ps.

For investigating slave metastability we need to determine the minimum clock pulse width that still allows safe operation of the measurement circuit. As we do not know the involved delays (interconnect, AND gate), the best option is to perform a calibration measurement. To this end we add a counter to the clock generation component that counts the clock pulses issued by the AND gate. By periodically reading the count value, a host PC can check whether the pulses are long enough to facilitate proper counting; we further call this a "heartbeat" of the counter. The calibration procedure works as follows

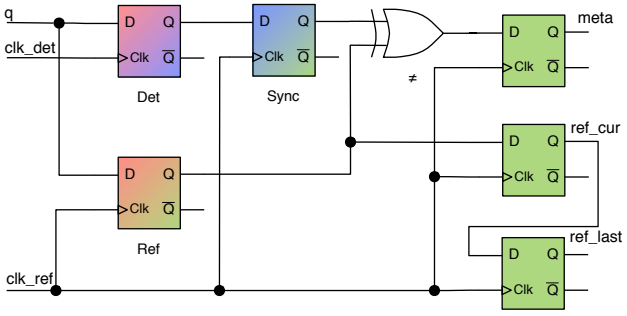

Figure 3. Adapted LTD circuit

- reset the circuit, set phase shift to zero
- decrease phase shift until heartbeat stops.
- increase phase shift until heartbeat is detected again.
- minimum pulse length for measurements is found.
- for clean measurements a safety margin of ten steps should be added (roughly 100 ps)

Note that when starting with zero delay "decrease" actually means applying *negative* delay (which the DCM can easily accommodate; it is just a phase alignment), so the delayed edges occur before the reference ones. Therefore the rising edge of the reference always determines the start of the pulse and hence the reference and the DUT clock stay in phase sync.

C. Late Transition Detection

We use the standard LTD detector found in numerous publications (e.g. [24], [27], [22]). It is shown in Figure 3.

The output of the DUT is fed into two flip flops, namely the detector flip flop and the reference flip flop. The reference flip flop has a whole clock period available for resolving metastability. The clock of the detector is shifted, such that its resolution time is variable. Please note the additional flip flop on the detection rail. It is necessary to re-synchronize the detection data to the reference clock. As the rising edge of the detection clock happens before the reference clock, the data at the xor gate would otherwise be out of sync.

The block labeled detection clock generation in Figure 2 shows how the digital clock manager (DCM) is used to generate the variable delay. This is basically the same solution as used in [3]. We have introduced a calibration here for the clock phase alignment between detection clock and reference clock, i.e. for the variable delay, since again unknown interconnect delays are involved. Our calibration algorithm works as follows:

- reset the circuit
- switch the DUT to synchronous operation, using a toggle flip flop driven by the reference clock (rather than the uncorrelated data source), as shown in Figure 2, block *calibration data generation*. In this mode

of operation we can be sure the DUT will operate metastability-free.

- decrease the variable delay until a mismatch at the XOR is seen. Since we can be sure we have no late transitions, this mismatch must be due to sampling the DUT output slightly before its nominal clock-to-output delay.
- increase the phase shift again until no more upsets are detected. Now sampling occurs right after the nominal clock-to-output delay, leaving zero resolution time for metastable upsets (corresponding to the definition of T_W from [6]).
- The calibration value for the detection clock phase shift has been found. We can now increment the resolution time, in our implementation in steps of $9.77\ ps$.

D. Detection Counters

We have slightly modified the LTD by introducing two additional flip flops (Figure 3 right). These maintain the reference values of the current and the previous cycle, such that the counter has a consistent triple available, comprising the following information: $meta$: mismatch between detector and reference detected; ref_cur: associated reference value; ref_last: previous reference value, which is the starting value of the current period. Note that it is important to have this information available in a *consistent* manner and evaluate the complete triple, rather than just counting events on all outputs. Therefore we need to provide a counter for each of the cases listed in Table I that is incremented every time a triple corresponding to its respective case is seen.

E. Controller

The on-chip controller is a combination of a simple state machine and a register bank. The register bank can be accessed by the host PC to set up measurements, calibrate the circuit and read back measurement results (counter values).

The state machine allows to precisely control the duration of a measurement: The host PC sets up the number of clock cycles the measurement shall last and sends the start signal. The controller counts the elapsed number of cycles and stops the measurement accordingly.

The controller works with a slower clock than the measurement and uses synchronizers for the control and status signals. The result counters need not to be synchronized as they are only read after the measurement was stopped.

V. Results

To validate our concept we have implemented it on a Xilinx Virtex-4 FPGA. As already mentioned we can achieve a time resolution of $9.77\ ps$ for our delay elements, using as a reference a $100\ MHz$ on-board crystal clock that we multiplied by 4 with the DCM. Figure 4 shows the results we obtained for rising edges (calibration uncertainties have been

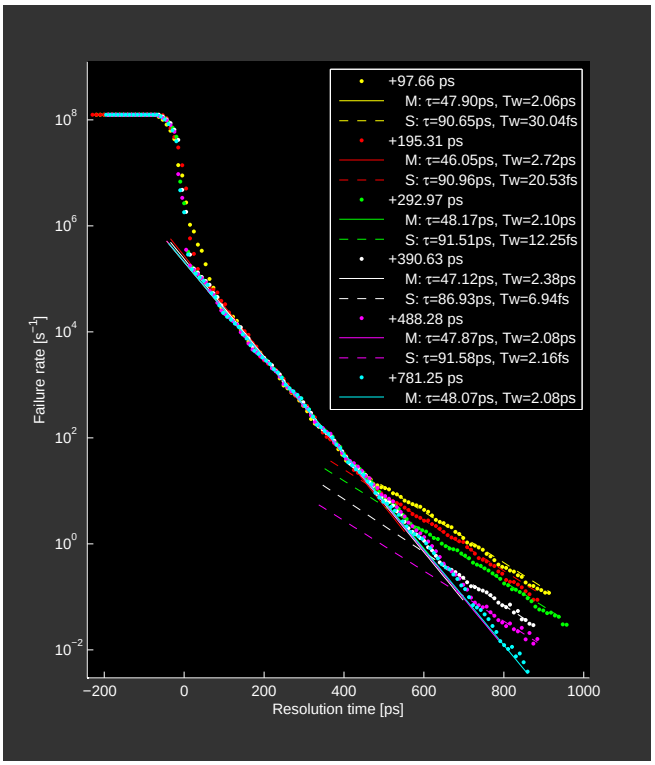

Figure 4. Measurement result for rising edges with different clock pulse widths

manually compensated). The figure represents an overlay of measurements with 6 different pulse widths – ranging from minimal pulse width $+97.66\ ps$ to $+781.25\ ps$ – for the clock (shown in different colors), therefore the slave metastability becomes effective at different points (lower right ends of the curves). The dashed lines represent the straight lines that best approximate the measured curves and hence allow to estimate the value of τ, which is in the relatively narrow range of $87 - 92\ ps$. This confirms that our approach for measuring slave metastability works.

The pronounced slope left of the slave metastability is characterized by the master metastability. Here the respective slopes (solid lines) are in the range of $46 - 48\ ps$. This matches the value published by Xilinx[29] (resulting in $\tau = 41ps$) quite well. At the same time it can be clearly seen that the τ of the master is much better than that of the slave, nearly by a factor of 2. So it is definitely optimistic to rely on the τ of the master alone (which is much easier to measure).

One can also observe that our setup allows to cover a considerable range of resolution times with very small step size, yielding a dense curve. The resolution time could be further enlarged without problems, but at the cost of considerable measurement time (see Section VI).

Plots of the raw measurement data have shown that the delay steps of the DCM are not completely uniform; there seems to be a larger step after a period of smaller ones. To

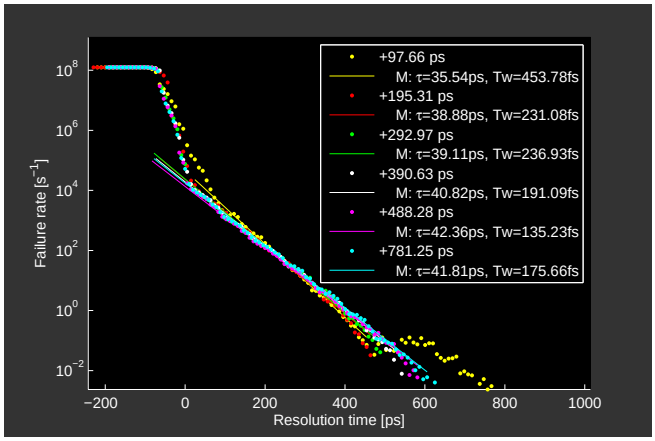

Figure 5. Measurement result for falling edges with different clock pulse widths

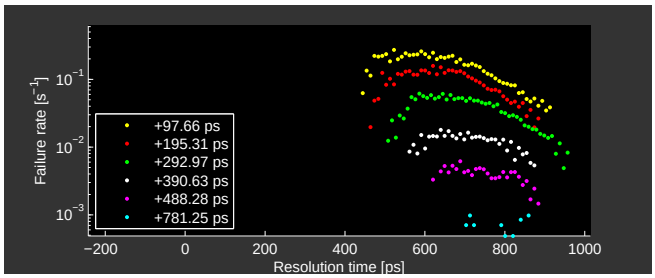

Figure 6. Measurement result for 1-0-1 pulses with different clock pulse widths

minimize the effects of this behavior, we used three parallel detectors for the same flip flop for the measurement. Since the routing delays of the detectors were slightly different, the effective resolution times ultimately differed. This delay mismatch was too small to be calibrated, so we simply aligned the measured curves. Consequently the large steps ended up at different positions on the time axis. Therefore calculating the mean of the logarithm of the data points not only removed the statistical effects (like clock jitter), but also the DCM step mismatches, yielding the relatively smooth curves shown in the figure. These curves are reproducible, even for different routing.

The equivalent results for the falling edges are shown in Figure 5. Here the τ for the master is in the range of $39 - 42\ ps$, which is clearly better than for the rising edge but in the same order. For the slave metastability we cannot make a useful observation here. This is because the output inverter has a low threshold, and so, if the slave gets metastable on a falling edge (i.e. the inverter input is rising), it will immediately cross that threshold just by going metastable, and no late transition will ever be observed. For falling edges the probability for metastable upsets is greatly decreased when the slave gets metastable. Unfortunately this is not a sign of a better resolution capability but the output flips

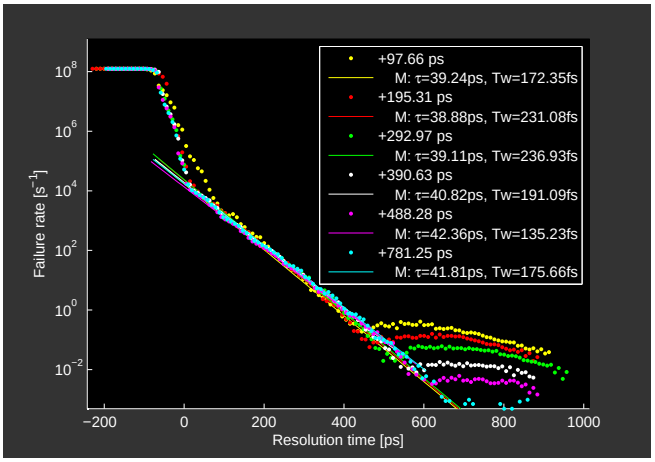

Figure 7. Measurement result for cases starting from 1

prematurely to the new value and these upsets are masked. This becomes clearly visible when comparing to the 1-0-1 pulses in Figure 6. For resolution times where we had observed slave metastability on the rising edges, now pulses start to appear. As with the slave metastability before, shorter clock pulse widths increase their probability and move their peak as well as their point of first occurrence towards higher resolution times. These graphs indicate the low threshold of the output inverter: When its input goes from LO (i.e. output HI) to the metastable state, the threshold is already crossed, causing a LO output, but then when metastability resolves back to LO, the threshold is crossed once again, bringing the output back to HI. With a similar argument we can explain why we do not see any 0-1-0 pulses for our DUT.

Figure 7 shows the plot for all metastable occurrences starting from HI, which is the sum of the occurrences from Figures 5 and 6. Without the preceding analysis one would be tempted here to misinterpret the rightmost parts of the graphs as late transitions due to slave metastability.

While for the determination of τ only differences of delays are relevant, T_W requires their absolute values. These are difficult to determine or control, and in fact we are not aware of any paper detailing an LTD scheme for accurately measuring T_W. Our calibration at least represents a step in that direction. It is, however, limited by the attainable step size which turned out way too coarse for a reliable measurement. So the values of T_W given in the figures must be seen as rough estimates.

VI. Constraints

When building a circuit that is as time critical as the late transition detection, one must adhere to numerous constraints and add annotations to make the tool chain produce a suitable implementation. The most important ones for the LTD are described in this section.

A. Critical paths

The clock domain crossings in the LTD design are of vital importance to the correctness of the circuit. Unfavorable delays decrease the available resolution time for the reference and can have a negative effect on the dynamic measurement range as well.

The available resolution time for the reference is:

$$t_{res}^{REF} = \frac{1}{f_{clk}} - \Delta_C^{REF,DUT} - \delta^{DUT,REF} - t_{co_{nom}}^{DUT} - t_{su}^{REF}$$

$\Delta_C^{REF,DUT}$ is the clock skew between detection and reference clock, $\delta^{DUT,REF}$ the data path delay between the two, $t_{co_{nom}}^{DUT}$ the nominal clock to output time of the DUT and t_{su}^{REF} the setup time of the reference. All these values shorten the available resolution time of $\frac{1}{f_{clk}}$. The only two parameters controllable by the designer are $\Delta_C^{REF,DUT}$ and $\delta^{DUT,REF}$. We tried to minimize $\Delta_C^{REF,DUT}$ by matching the paths through the clock pulse generation circuit. Additionally we manually placed the DUT and the reference to also minimize $\delta^{DUT,REF}$.

The penalty of a short resolution time for the reference is an increased probability that its value will not be resolved within the available time, yielding an incorrect reference.

For the detector (DET) the situation is somewhat more complex. As the detector must read the value from the DUT (at different times) and must also safely deliver its measured value to the reference clock domain (via the sync FF), its maximum resolution time is constrained by both paths, DUT to DET and DET to SYNC. Again we minimized the penalty of the data path delay by manually placing the flip flops. In addition, we introduced some pipeline stages to simplify the placing of the critical flip flops. To compensate for the clock skew we introduced the calibration mode. Therefore the skew between the two clocks can be bounded and is of little consequence to the measurements.

B. Nominal output delay

To be able to calculate T_W, the nominal output delay of the DUT is required. In principle, the calibration algorithm proposed in this paper can achieve this. A fundamental limitation here is, however, that the nominal output delay of the flip flop is not a single value but has a distribution, so finding its maximum is not possible within finite calibration time. A more practical issue was that the calibration granularity was too coarse.

C. Measurement considerations

There are many parameters to consider for achieving a fast and reliable measurement. The measurement time must be large enough to minimize the statistical errors, but still tractable. To achieve this we dynamically adapted the measurement duration. If for one resolution time the event count dropped below a certain limit (we used 500), we doubled the measurement period for the next (larger) resolution time. By implementing this regime, the region with high failure rates can be measured fast, while that with low failure rates still attains adequate quality. For the results presented in this paper, the measurement of the first $200ps$ of resolution time took approximately $3min$, while the last $200ps$ took $10.8h$. As one would expect, the measurement time grows exponentially.

For a good resolution the length of a time step is essential. For the DCM, the number of achievable steps per clock period and the step size itself greatly depend on the input clock frequency. For a good time resolution, the clock frequency should be as high as possible. Therefore, in the proof of concept, we used the maximum frequency supported by the Virtex-4 DCM. The non-ideal differential linearity of the step sizes has already been addressed in Section V.

Another potential source of uncertainty is the time interval measurement. We employed a counter on the target device to count the clock cycles within a measurement interval. As the time interval is now given in clock cycles, its duration in terms of time has no effect on the calculation of τ as it cancels out in the corresponding equation.

Finally, PVT variations of both, target and measurement circuits are a notorious source of uncertainty. Our approach does not solve this problem, but part of its charm is to allow a measurement of the concrete device at hand in its concrete environment rather than having to rely on abstract data sheet values that ultimately require substantial safety margins to safely cover the specific case.

VII. Conclusion

We have presented an approach to characterize the metastability parameters of an FPGA that is solely based on FPGA internal infrastructure but still attains good accuracy. This allows a designer to perform the characterization for his specific target FPGA under the anticipated operating conditions.

In our approach metastability generation is based on uncorrelated oscillators for clock and data. In order to comprise slave metastability as well, however, the conventional approach is enhanced by using a variable duty cycle for the clock oscillator that is generated by use of a DCM.

For metastability detection we rely on late transition detection. The variable delay required in this approach is again generated by means of the DCM. To handle the unknown phase of the DCM output (including routing delays) we propose a calibration measurement, while statistical errors are mitigated by averaging over multiple measurements that we run in parallel. A novel concept is the separated logging of different output scenarios (rising/falling edges) that allows a more detailed analysis of the results, in particular the identification of glitches.

Our measurements on a Xilinx Virtex-4 FPGA have demonstrated the viability and accuracy of our approach. In

a next step we will explore its suitability for a more recent FPGA technology.

We have also discussed sources of systematic errors and requirements for attaining an efficient and accurate measurement setup. Our future work will encompass the use of our setup for investigating the voltage, temperature and load dependence of the metastability parameters. Specifically the load dependence is interesting in order to perform projections from the measured values to the ones effective during field operation, as the load may be different. Furthermore we plan to investigate potential sources of uncertainty in more detail.

REFERENCES

[1] R. Ginosar, "Fourteen Ways to Fool Your Synchronizer," in *Symp. on Async. Circuits and Systems*, 2003, pp. 89–96.

[2] ——, "Metastability and Synchronizers: A Tutorial," *Design and Test of Computers*, vol. 28, no. 5, pp. 23–35, 2011.

[3] S. Beer, R. Ginosar, M. Priel, R. Dobkin, and A. Kolodny, "The Devolution of Synchronizers," in *Symposium on Asynchronous Circuits and Systems*, 2010, pp. 94–103.

[4] W. J. Dally and S. G. Tell, "The Even/Odd Synchronizer: A Fast, All-Digital, Periodic Synchronizer," in *Symposium on Asynchronous Circuits and Systems*, 2010, pp. 75–84.

[5] I. Catt, "Time Loss Through Gating of Asynchronous Logic Signal Pulses," *Transactions on Electronic Computers*, vol. EC-15, no. 1, pp. 108–111, 1966.

[6] D. J. Kinniment, *Synchronization and Arbitration in Digital Systems*. Wiley, 2007.

[7] S. Yang, I. W. Jones, and M. R. Greenstreet, "Synchronizer Performance in Deep Sub-Micron Technology," in *Symposium on Asynchronous Circuits and Systems*, 2011, pp. 33–42.

[8] L. R. Marino, "General Theory of Metastable Operation," *Trans. on Computers*, vol. C-30, no. 2, pp. 107–115, 1981.

[9] T. Kacprzak and A. Albicki, "Analysis of Metastable Operation in RS CMOS Flip-Flops," *Journal of Solid-State Circuits*, vol. 22, no. 1, pp. 57–64, 1987.

[10] I. W. Jones, S. Yang, and M. R. Greenstreet, "Synchronizer Behavior and Analysis," in *Symposium on Asynchronous Circuits and Systems*, 2009, pp. 117–126.

[11] S. Beer and R. Ginosar, "An Extended Metastability Simulation Method for Synchronizer Characterization," in *Workshop on Power and Timing Modeling, Optimization and Simulation*, 2012.

[12] S. Yang and M. R. Greenstreet, "Simulating Improbable Events," in *Design Automation Conf.*, 2007, pp. 154–157.

[13] H. J. M. Veendrick, "The Behavior of Flip-Flops Used as Synchronizers and Prediction of Their Failure Rate," *Journal of Solid-State Circuits*, vol. 15, no. 2, pp. 169–176, 1980.

[14] L. Kleeman and A. Cantoni, "Metastable Behavior in Digital Systems," *Design and Test of Computers*, vol. 4, no. 6, pp. 4–19, 1987.

[15] Y. Semiat and R. Ginosar, "Timing Measurements of Synchronization Circuits," in *Symposium on Asynchronous Circuits and Systems*. IEEE Computer Society, 2003, pp. 68–77.

[16] D. J. Kinniment, K. Heron, and G. Russell, "Measuring Deep Metastability," in *Symposium on Asynchronous Circuits and Systems*, 2006, pp. 2–11.

[17] J. Zhou, D. J. Kinniment, C. E. Dike, G. Russell, and A. Yakovlev, "On-Chip Measurement of Deep Metastability in Synchronizers," *Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 550–557, 2008.

[18] T. J. Chaney and C. E. Molnar, "Anomalous Behavior of Synchronizer and Arbiter Circuits," *Transactions on Computers*, vol. C 22, no. 4, pp. 421–422, 1973.

[19] B. M. Rogina, P. Skoda, K. Skala, I. Michieli, M. Vlah, and S. Marijan, "Metastability Testing at FPGA Circuit Design using Propagation Time Characterization," in *East-West Design and Test Symposium*, 17-20 2010, pp. 80–85.

[20] J. Kalisz and Z. Jachna, "Dual-Edge Late-Transition Detector for Testing the Metastability Effect in Flip-Flops," in *Conference on Microelectronics*, 6-8 2004, pp. 780–782.

[21] P. Alfke. (2005) Metastable Recovery in Virtex-II Pro FPGAs. [Online]. Available: http://www.xilinx.com/support/documentation/application_notes/xapp094.pdf

[22] J. Zhou, D. J. Kinniment, G. Russell, and A. Yakovlev, "Adapting Synchronizers to the Effects of on Chip Variability," in *Symposium on Asynchronous Circuits and Systems*, 2008, pp. 39–47.

[23] D. J. Kinniment and J. V. Woods, "Synchronisation and arbitration circuits in digital systems," in *Proceedings of the Institution of Electrical Engineers*, 1976, pp. 961–966.

[24] F. Rosenberger and T. J. Chaney, "Flip-Flop Resolving Time Test Circuit," *Journal of Solid-State Circuits*, vol. 17, no. 4, pp. 731–738, 1982.

[25] M. Bhushan, M. B. Ketchen, and K. K. Das, "CMOS Latch Metastability Characterization at the 65-nm-Technology Node," in *Conf. on Microel. Test Struct.*, 2008, pp. 147–151.

[26] A. Rothermal and F. Dell'ova, "Analog Phase Measuring Circuit for Digital CMOS ICs," *Journal of Solid-State Circuits*, vol. 28, no. 7, pp. 853–856, 1993.

[27] J. Wu, Y. Ma, J. Zhang, Y. Kong, H. Song, and X. Han, "Research on Metastability Based on FPGA," in *Conference on Electronic Measurement & Instruments*, 2009, p. 4.

[28] D. Chen, D. Singh, J. Chromczak, D. Lewis, R. Fung, D. Neto, and V. Betz, "A Comprehensive Approach to Modeling, Characterizing and Optimizing for Metastability in FPGAs," in *Symposium on Field Programmable Gate Arrays*. ACM, 2010, pp. 167–176.

[29] P. Alfke. (2008) Metastable Delay in Virtex FPGAs. [Online]. Available: http://forums.xilinx.com/t5/PLD-Blog/Metastable-Delay-in_Virtex-FPGAs/ba-p/7996