Metastability-Aware Memory-Efficient Time-to-Digital Converters

Matthias Függer, Attila Kinali, Christoph Lenzen, Thomas Polzer

Metastability-Aware Memory-Efficient Time-to-Digital Converters

Matthias Függer*
*LSV, CNRS & ENS Paris-Saclay
mfuegger@lsv.fr

Attila Kinali† Christoph Lenzen†
†Max Planck Institute for Informatics, Saarbrücken
{adogan,clenzen}@mpi-inf.mpg.de

Thomas Polzer‡
‡TU Vienna
tpolzer@ecs.tuwien.ac.at

***Abstract*—We propose a novel method for transforming delay-line time-to-digital converters (TDCs) into TDCs that output Gray code without relying on synchronizers. We formally prove that the inevitable metastable memory upsets (Marino, TC'81) do not induce an additional time resolution error. Our modified design provides suitable inputs to the recent metastability-containing sorting networks by Lenzen and Medina (ASYNC'16) and Bund et al. (DATE'17). In contrast, employing existing TDCs would require using thermometer code at the TDC output (followed by conversion to Gray code) or resolving metastability inside the TDC. The former is too restrictive w.r.t. the dynamic range of the TDCs, while the latter loses the advantage of enabling (accordingly much faster) computation *without* having to first resolve metastability.**

Our all-digital designs are also of interest in their own right: they support high sample rates and large measuring ranges at nearly optimal bit-width of the output, yet maintain the original delay-line's time resolution. No previous approach unifies all these properties in a single device.

I. Introduction

Time-to-digital converters (TDCs) transform the continuous time difference between a starting and a stopping signal transition into a discrete value. Naturally, TDCs are basic building blocks in many applications, reaching from logic analyzers over experimental setups in high energy physics to high precision clock synchronization and time-of-flight for optical distance measurements. TDC designs have been extensively studied, see e.g. [7] for a recent overview on high-precision TDCs.

One popular way to build TDCs is to use a tapped delay line and bistable storage elements to measure the inter-arrival times of the starting and stopping transitions. Depending on the relative order at which the two delayed signal transitions arrive at the storage element, the element will stabilize to 0 or 1. Variants differ in whether both the starting and stopping signals are delayed (Vernier delay line) or only one signal is delayed ("regular" delay-line TDC). In the latter case, routing of the stopping signal is such that it almost simultaneously arrives at all memory elements.

Fig. 1 depicts the general structure of a delay-line TDC, see e.g. [3], [14], [16], [17]. The delay-line is tapped in between each two successive delay elements, where each tap drives the data input of an initially enabled latch. The TDC readout is then formed by the latch states. Inherently, the TDC's time resolution is in the order of the latency T_d of a delay element, e.g., an inverter. For recent delay-line architectures, see e.g. [3], [14], [17], time resolution is in the order of 10–80 ps, depending on the used process and inverter structure. With interpolation between delay elements, a bin size of 5 ps has been recently demonstrated, using a 130 nm process [14].

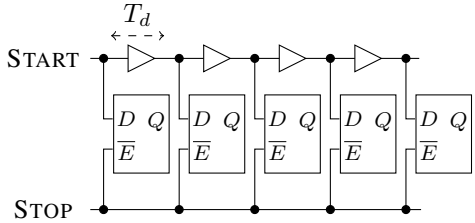

Fig. 1: Tapped delay-line TDC. Latches are initially enabled and output 0. The delayed starting signal iteratively sets latches to 1 until the stopping signal disables them.

To keep complexity low and to extend the dynamic range, often a fine-TDC/coarse-TDC approach is employed. There exist many variants of this approach and similar strategies have been invented many times independently. To the best of our knowledge, the earliest example of the fine-TDC/coarse-TDC approach is [12].

A. The Problem of Metastable Upsets

In his seminal work [10], Marino formally proved that *any* circuit with a bistable output can become *metastable*, i.e. that its output might transition to a stable 0 or 1, an arbitrarily long time after the input stimulus has been applied. In the case of TDCs, the latch input might transition exactly when being captured, violating setup/hold constraints, resulting in metastability.

However, in a TDC of the general structure depicted in Fig. 1, it is straightforward to restrict the number of metastable output bits to at most one as follows. Assume that the TDC has $n \geq 1$ latches and each delay element has latency $T_d > 0$. Let T be the time the latches are given before their outputs are further processed by a readout circuit or copied to memory. For a fixed execution of a START transition traversing the ring TDC, each latch has a *critical window* of time: if the STOP signal arrives at a time t from this window, a metastable upset of the latch may not resolve by time $t + T$; if t lies outside of this window, it is guaranteed that the latch is stable at time $t + T$. Larger T results in smaller window sizes [8], but longer readout time and thus lower sample rates. Thus, there is a general trade-off between fast readout (high sample

rate) and high reliability (low upset probability). Note that routing delay uncertainties/variations can be accounted for in the critical window sizes and positions.

Note that, for given window sizes and bounds on the unbalanced arrival of the stopping signal, one can choose the delay-element latency T_d large enough for the critical windows to be non-overlapping. In this case, the (thermometer encoded) TDC readout is guaranteed to contain at most one metastable bit: it is of the form $11..100..0$ or $11..1M00..0$, i.e. a consecutive series of 1's followed by a consecutive series of 0's with at most one metastable bit/latch M in between. No matter how M resolves, the final measurement value is off by at most one from the actual readout; we say the TDC *guarantees precision of* 1 *(in units of stage delays)*. By Marino's result this is optimal: there is no TDC implementation which guarantees only stable output bits.

We stress that reducing T_d further, such that potentially more than one bit may become metastable, may lead to better average precision, but does *not* improve worst-case precision. In other words, the critical window size is a lower bound on the single-shot precision of the TDC. In this work, we are aiming for optimal single-shot precision, which is relevant to applications that provide or rely on guaranteed precision bounds, e.g., within control loops in mission critical hardware. As in most settings the above lower bound can be matched by delay-line TDCs, extending our techniques to Vernier line TDCs is outside the scope of this paper.

B. Dealing with Metastability

As we have seen, metastable upsets cannot be avoided deterministically. Worse, when increasing the accuracy of the TDC, we increase the risk for metastability, as the relative size of the critical windows becomes larger. At the latest when reaching the limit on TDC precision implied by Marino's results, the effects of metastability on the circuit cannot be neglected anymore. Thus, the threat of metastability must be addressed by the circuit design. This can be done in the following ways.

Resolving in TDC: The TDC readout can be given sufficient time for the metastable bit to stabilize with sufficiently high probability. While this solution is the simplest, it prevents the TDC from taking a new measurement during this time, reducing its sample rate.

Resolving in Memory: The TDC readout can be written to memory; this may result in metastability of a written bit, which then is resolved by (chains of) synchronizers [8] or, again, simply waiting for sufficiently long time. Note that the TDC can "keep running," in this setting, as the "stopping" signal here is used to capture the current TDC state in memory only. This enables the TDC to be reused for additional measurements while time-stamps resolve metastability in memory.

Resolving during/after Computation: Friedrichs et al. propose a new approach [5]: they introduce the concept of metastability-containing circuits. Instead of waiting for the metastable bits to stabilize, such circuits guarantee a bounded degree of uncertainty in outputs, given that the input comprises a bounded amount of metastable bits. For example, one can immediately begin computing the maximum of potentially metastable measurement values of precision 1, yet ensure output of precision 1 [2], [9]. This permits to simultaneously use the time to resolve metastability for computations, as well as using an adaptive amount of time for resolving metastability *after* computations (depending on the time until the output needs to be sampled for further processing).

No Resolution: If all operations are metastability-containing, resolving metastability is not necessary at all. For example, this is the case when the result of the computation is used for analog control; the authors of [2], [5], [9] discuss clock synchronization as an application where this is feasible and of interest.

We emphasize that existing TDC designs have severe limitations when it comes to the last two options:

- Delay-line TDCs without coarse counters ensure that only a single bit is metastable (under the same constraints as our designs). However, such designs are practical for a very limited dynamic range due to the inefficient encoding and require subsequent conversion to Gray code for (efficient) follow-up computations.
- Most current counter-based designs directly protect their (binary) counters from metastable upsets by synchronizers, incurring a delay of several clock cycles (i.e., typically nanoseconds) before the measurement becomes available. This also necessitates to stop the TDC oscillator, disallowing multiple measurements pertaining to the same starting signal.
- An exception is the design by Mota et al. [11]. It employs two coarse counters (driven by rising and falling clock edges, respectively), one of which is guaranteed to not be metastable upon sampling the TDC state with the stop signal (i.e., latching the corresponding registers). This enables waiting for stabilization in memory. However, reading the memory word representing the TDC value correctly requires to determine which counter's sampled value to use, which in turn requires prior stabilization of the thermometer encoding of the sampled fine-TDC value. Apart from stopping the entire TDC, this requires an additional dead time of at least 1 ns to reach a moderate MTBF of 8.8 years per thermometer bit (cf. Section II).

Therefore, the techniques presented here are a big step towards an efficient implementation of the clock synchronization scheme proposed by Friedrichs et al. [5].

C. Memory Efficiency

Achieving precision 1 and the possibility to resolve metastability *outside* the TDC (and thus taking multiple measurements concurrently) are prime design goals. Unfortunately, as mentioned above, they come at the cost of a very large memory footprint for tapped delay-line TDCs: The time resolution of the TDC is given by the delay element latency T_d, and its measurement range is nT_d. Resolving metastability in memory

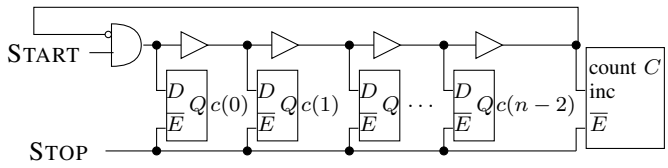

Fig. 2: Generic ring TDC architecture with coarse counter C. Latches and counter are initialized to 0. The counter increments on rising and falling transitions.

does not alleviate this problem, as the memory consumption for storing a single measurement means that we could just add another delay-line TDC at essentially the same cost!

Note that naively packing the thermometer code measurement values into binary coded values *before* metastability is resolved may result in arbitrarily bad precision (i.e., has arbitrarily large errors). For an example illustrating the problem, assume that 1111111M (with M denoting the metastable bit) is written to a FIFO with thermometer encoded stages. After metastability resolution, we end up with either 11111110 or 11111111 — decimal 7 or 8. The binary representation of these numbers are 0111 and 1000, respectively. Note that these strings differ in all bits. As shown in [5], *any* circuit computing the binary representation of unary inputs may thus return MMMM when given input 1111111M. This is to be seen as a discrete version of Marino's impossibility result [10]. The string may thus resolve to any 4-bit string, i.e., the binary representation of *any* number between 0 and 15 — losing all TDC precision at the encoding step! Although the probability for such upsets can be made small (but not zero!) by synchronizer chains, this is at the cost of increased TDC latency and synchronizer chains for *all* measurement bits. Portmann and Meng studied the same issue as it arises for flash ADCs in [15].

To reduce both the memory footprint and the size of the TDCs, both for tapped delay-lines [6] and Vernier lines [4] solutions have been proposed where the delay line has been folded into a ring oscillator-like structure with an additional (binary) coarse counter to count the number of rounds the start signal makes in the ring oscillator; see Fig. 2. The stopping signal then freezes both the latches and the counter.

However, these designs suffer from potential metastable upsets when incrementing the counter. For the reason laid out above, even waiting for the counter to stabilize after possible metastability does not fix the problem: in case the counter is binary, the resulting TDC output value has arbitrarily bad precision. Note that this analysis does not include the timing violations due to different path lengths for different bits within the adder structure, but just the metastable upsets within the output register. To address this, one can employ synchronizers more carefully [1], [8], [18], decreasing the probability of a metastable upset of the counter arbitrarily. This trades time (for the synchronizer steps) for increased reliability.

Even if this is acceptable, it requires to stop the counter for taking a measurement, restricting the TDC to taking a single measurement *and* waiting for the synchronizers to stabilize before starting a new one. While the former restriction can be lifted by duplicating the counter logic and synchronizers for each concurrent measurement, this basically amounts to duplicating the entire TDC for each concurrent measurement.

D. Our Contribution

In Section II, we present simulations demonstrating the potentially severe impact of metastable upsets on TDC precision. Given the limitations of synchronizer-based workarounds, we believe this to make a strong case for our ideas.

As our main contribution, we present a general approach of deriving memory-efficient all-digital TDCs that support (i) high sampling frequencies, (ii) and allow to resolve metastability outside the TDC. Note that this enables multisampling extensions of the presented TDCs, with multiple STOP ports *without* replicating the TDC's counter logic.

We prove correctness of three flavors of the generic design depicted in Fig. 2, showing that the approach can be flexibly adapted to different needs. They differ in how the generic "blackbox" counter is implemented and how the TDC output is encoded.

I The counter is realized by two binary counters and a single bit indicating which counter value to use (Section III-B1). While metastability cannot be avoided when reading a counter, one can make sure that the indicator bit points to a stable counter value. To the best of our knowledge, this design principle has been first described in [11], however, without a formal proof of correctness.
   + use of any *binary* counter design, despite the vulnerability of binary encoding to metastability
   − two counter values need to be stored
   − inefficient for metastability-containing operations

   We then present two highly memory-efficient alternatives.

II Using a Gray code counter (Section III-B2).
   + no memory overhead of encoding
   + suitable for metastability-containing operations
   − restrictive timing constraints

III Using a Gray code counter and an additional bit (Section III-B3). The bit indicates whether an up-count should have taken place, shifting the tight timing constraints from the counter to a single latch.
   + only a single bit memory overhead of encoding
   + same timing constraints as first option
   − inefficient for metastability-containing operations

The time resolution of all these designs is the minimum delay element latency T_d ensuring that critical windows of latches do not overlap.

II. Binary Cycle Counter Designs

When using a binary counter to keep track of the number of rounds/cycles in the ring oscillator between arrival of START and STOP, the ring is used as clock signal for the counter, and the STOP signal is used as a gate signal (connected to the enable input of the counter's D-flip flops). Such an implementation is used quite frequently in synchronous designs. The problem in the case of TDCs is that the STOP is outside of the timing closure of the counter and may occur at an

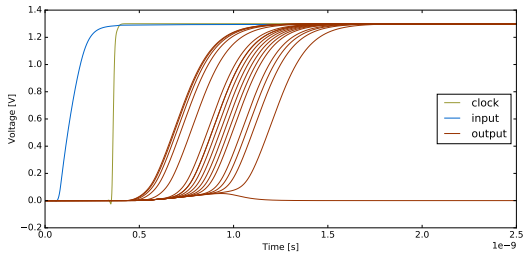

Fig. 3: Transient simulation of the capture register of the simulated TDC. Even very small variations of the applied input signal increase the time until the output value reaches its final voltage level considerably. Note that a full D-FF was used instead of a single latch.

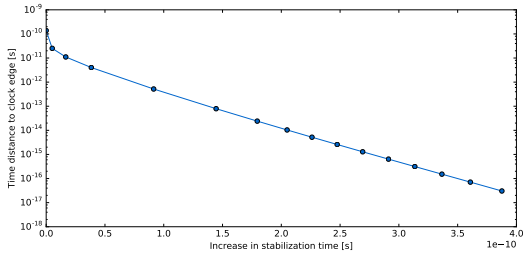

Fig. 4: Stabilization time variation due to metastable upset of the simulated TDC capture register. time between application of the input and the clock edge vs. output stabilization time.

arbitrary point in time. To mitigate this problem, synchronizers can be used to align the STOP signal with the counter clock. However, this naive solution either dramatically reduces the precision of the TDC, as an additional delay is added to the STOP signal that varies depending on the phase offset between the clocking and STOP signals, or does little to improve safety, as too little time is allocated to reliably resolve metastability. More involved designs use two fine TDCs, one for the START and one for the STOP signal (e.g. [12]) such that the counter can be stopped synchronously to its clock, but this entails having two fine TDCs and potentially duplicating the entire counter logic to allow for multisampling (due to the additional delay in the STOP signal caused by the synchronizers).

To demonstrate that the probability of metastable upsets cannot be neglected, consider a state-of-the-art ring TDC with binary counter as implemented by [13], [14]. Let us replace a ring latch in Fig. 2 with a D-FF whose data input is driven by the respective ring-segment and that is clocked by the STOP signal. For the above setup assume input transitions to occur with a rate of $f_d = 500\,\text{MHz}$ and a STOP transition occurring every 20 ns (an input rate of $f_c \geq 50\,\text{MHz}$ is typical for high-speed clock synchronization systems). For the mean time between metastable upsets in the synchronous circuit, we obtain [8] $T_{\text{MTBF}} = e^{T_{\text{res}}/\tau}/(f_c f_d T_w)$, with the remaining parameters being determined as follows. To characterize metastability of the ring state flip-flop, we implemented part of the ring (6 ring inverters) as described in [13], [14], driving the data port of the D-FF, and shaped the clock signal by 6 upfront standard inverters. Cell design of ring inverters and D-FF, including transistor sizing were taken from [13], for a 130 nm technology. We then ran Spice simulations (Fig. 3, Fig. 4) for the above circuit showing a settling time τ for the complete chain from START and STOP to the D-flip-flop outputs of 31.6 ps and a critical window size T_w of 8 ps. Assuming that metastability has $T_{\text{res}} = 1\,\text{ns}$ time to resolve before the measured value is used, we obtain an T_{MTBF} of 8.8 years *per bin and measurement channel* (i.e. assuming a TDC with 64 bins and 8 channels the MTBF reduces to roughly 6 days). Note that the situation is even worse for the coarse counter T_{MTBF}, as it comprises several flip-flops and its increment logic is not perfectly balanced. This shows that using synchronizer chains is inevitable for such designs, leading to large dead times of the TDC. In the following section, we present three alternative solutions.

III. METASTABILITY-AWARE TDCS

In this section, we present our generic approach for devising ring TDCs that do not suffer from reduced precision due to metastable upsets. We showcase our ideas at hand of three alternative designs, each of which allows for taking multiple measurements concurrently (without duplicating the TDC logic) and achieves optimal single-shot precision. Our designs differ in terms of the trade-offs between routing constraints and memory overhead; the output of one satisfies the input specification of the metastability-containing sorting networks from [2], [9], without the need for further conversion or waiting for stabilization.

For the sake of presentation, we assume throughout this section a simultaneous arrival of the stopping signal at all relevant components, following the tapped delay-line approach. In practice, one may choose other options and compensate for them or switch to a Vernier line approach; these considerations are outside the scope of this paper. Moreover, we discuss our designs in the context of a measurement with a single start signal followed by a single stop signal, where the measurement value is captured in the TDC.

A. The Ring

Consider a ring TDC architecture of length n as depicted in Fig. 2. We denote the output of latch i, $0 \leq i \leq n-2$, at arrival of the STOP signal by $c(i)$. At this point we assume latches with zero switching time. We will discuss this issue shortly. For the moment, assume that the coarse counter C is a (single-bit, binary) latch. We denote its output by $c(n-1)$.

Lemma 3.1: Denote by cnt the sum of the total number of latch output transitions of all stages between the starting and stopping signals, i.e. the total count of the TDC. Then either the ring oscillator made a number of full rounds and all latch outputs are the same, or its state was captured in between and the latch outputs are of the form 00..011..1 or 11..100..0; formally:

1) $\text{cnt} \bmod n = 0$ and $c(i) = c(j)$ for all $i, j \in [n]$ or
2) $c(\text{cnt} \bmod n) \neq c(\text{cnt}+1 \bmod n)$ and $c(i) = c(i+1)$ for all $i \in [n] \setminus \{\text{cnt} \bmod n\}$.

This lemma abstracts away the issue that the latches cannot switch from 0 to 1 or vice versa in zero time. Disabling them during their critical window may therefore result in metastability. However, as laid out in Section I-A, choosing

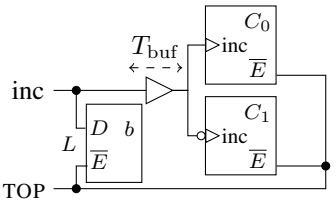

Fig. 5: Solution I: Implementing C by two binary counters and a latch. Counters C_0 and C_1 and latch L are initialized to 0. The increment inputs of C_0 and C_1 are driven via a delay buffer. C_0 increments on rising, C_1 on falling transitions.

the delay latency T_{d} large enough to separate the latches' critical windows guarantees that at most one bit is metastable. This, plus the fact that the ring stores the measurement value thermometer encoded, guarantees precision of 1:

Corollary 3.2: Assume that the critical windows of the latches are non-overlapping. Then, after potential metastability resolved, $\text{cnt} \bmod n$ can be determined with precision 1 (cf. Lemma 3.1).

B. The Coarse Counter

We now address the issue of counting the number of cycles. Doing this in a naive way incurs a loss in precision of the TDC, as has been shown in Section II. We present three coarse counter variants which do not suffer from this problem, i.e., ensure an optimal precision of 1.

1) Solution I: Redundant Binary Counters: Consider the coarse counter implementation in Fig. 5 (as described in [11]). The circuit's underlying idea is to use two redundant binary counters C_0 and C_1 and the output b of latch L capturing their common input. Initially, $b = 0$, $C_0 = 0$, and $C_1 = 0$. When the stopping signal arrives, the binary counters and latch L are disabled. Output b serves as a control bit: we use the value stored in C_0 if $b = 0$ and the value in C_1 if $b = 1$. This ensures that the counter, to which b is pointing, had a full cycle of the ring oscillator to stabilize.

In order for this approach to work, we require the following constraints to hold: The critical windows W_{C_0} and W_{C_1} of counters C_0 and C_1, as well as the critical window W_L of latch L are mutually disjoint and obey the following order in time: W_L, W_{C_0}, W_L, W_{C_1}, W_L, W_{C_0}, and so on. Assuming the same designs for both counters (i.e., $|W_{C_0}| = |W_{C_1}|$), this can be achieved by the following design constraints:

1) W_L before W_{C_0}: choose the delay latency T_{buf} sufficiently large.
2) W_{C_0} before next W_L: for a fixed T_{buf}, choose the ring size n sufficiently large for the binary counter to complete an increment before the transition traverses the ring.

Fig. 6 depicts the alignment of the critical windows with the above constraints fulfilled.

When reading the counter, the number of completed cycles cyc is computed as $2C_b + b$; see Lemma 3.3. Since $2C_b + b$ is equal to the value of the concatenated binary counter (C_b, b), we may view bit b as a shared bit of the binary counters $(C_0, 0)$ and $(C_1, 1)$ that hold the value cyc in case of $b = 0$ and $b = 1$, respectively. This allows for an efficient method for the TDC to directly return the binary encoding of cyc.

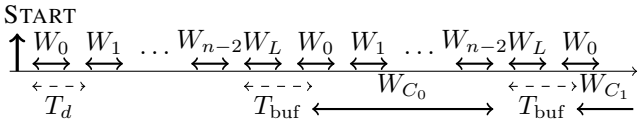

Fig. 6: Critical windows for Solution I as the START signal travels through the ring oscillator. First the first stage "handles" the START signal during the stage delay time T_{d} and can become metastable during its critical window W_0, then the second stage during it's window W_1 and so on. When the START signal traveled once around the the ring oscillator it will increment the first counter C_0 after the buffer delay T_{buf} and thus cause it to enter its critical window W_{C_0}, while the signal continues to progress through the ring oscillator. After the second round, the second counter C_1 is incremented and can potentially become metastable during its critical window W_{C_1}.

Lemma 3.3: After possible metastability of b has been resolved, it holds that $\text{cyc} = 2C_b + b$.

Proof: We only give the proof idea here. From the constraints on the critical windows, we have that the critical windows W_L, W_{C_0}, and W_{C_1} are mutually disjoint. Thus, in case b is metastable, C_0 and C_1 are not. Any resolution of b will thus result in a correct measurement value. In case, say, C_0 is metastable, $b = 1$ must hold and C_1 cannot be metastable. Again, this results in a correct measurement value. The case of C_1 being metastable is analogous. ■

The complete TDC is given by combining the methods of counting modulo n discussed in Section III-A and determining cyc given in Section III-B:

Theorem 3.4: After metastability of the latches has been resolved, $\text{cnt} = (2C_b + b)n + x_{1-b}$, where x_{1-b} is the number of latches having value $1-b$. Moreover, it is exactly the leading x_{1-b} latches that have value $1 - b$.

Recall from Section II that waiting for 1 ns before consulting the latch states resulted in a moderate MTBF of 8.8 years per bit and channel. Further note that, if the binary counters have B bits, we store $2B + n$ bits for a measurement, but the maximum cnt value $(2^{B+1} - 1)n + (n - 1)$ (in principle) requires $B + 1 + \lceil \log n \rceil$ bits only. This is of concern when taking multiple measurements concurrently. The next sections discusses two TDC variants that address this issue.

2) Solution II: Gray Code Counter: In this section, we present a circuit that can be used to read the TDC in a way requiring to store optimal $B + \lceil \log n \rceil$ bits per measurement only. As a starting point, observe that we know that no circuit can avoid metastability of an output bit if, given all other inputs, the output value depends on an input bit that may be metastable [5], [10]. Therefore, the first key insight is that we must use an encoding guaranteeing that metastability of *any* latch must not cause metastability of more than one output bit: otherwise, for any encoding without redundancy, we must lose precision, as two or more bits resolving to arbitrary values can induce a change of the encoded value larger than 1. In other words, we must use a Gray code, for which any up-count changes exactly one bit.

a) Look-ahead Gray Code Counter: Consider the ring architecture from Fig. 2 and replace the coarse counter by a single *Gray code counter with look-ahead* that increments

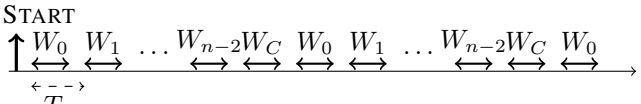

Fig. 7: Critical windows for Solution II.

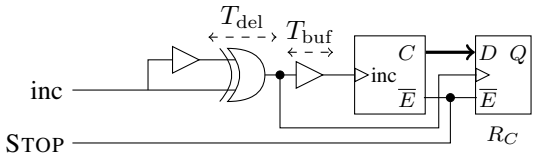

Fig. 8: Solution II: Implementation with look-ahead Gray code counter and register. The counter C is initialized with 1. Register R_C is of same width as the counter, and initialized with 0.

both on rising and falling transitions. When the counter is triggered, it is required (i) that it internally prepares an up-count within a single oscillator cycle, and (ii) that, once ready for an increment and it is triggered, its outputs react fast, such that their critical window is short and does not overlap with any other critical windows of the ring latches; thus the term look-ahead. Fig. 7 shows the resulting alignment of critical windows. Note that such a counter can be implemented by a Gray code counter that counts "ahead" by one (performing each increment within an oscillator cycle) and whose outputs are captured upon an "actual" increment. Fig. 8 shows such an implementation: the rising edge triggered counter is initialized with (the look-ahead) value 1, and the rising edge triggered register with 0. The XOR transforms falling a rising transitions on inc into pulses, triggering (a) the register to capture the actual (pre-computed) counter value, and (b) the counter to start the next increment. The buffer delay T_{buf} makes sure that (a) always happens before (b). The counter now has a complete ring oscillator cycle to finish its computation.

A TDC readout thus comprises the register states (in Gray code) and the ring latches (thermometer encoded).

We next discuss how to interpret a stabilized TDC readout, e.g., assume it was stored in an external memory with sufficient time to stabilize, and now is read by an application.

We determine whether cyc is odd or even from the coarse counter value, i.e., we compute $b = \mathrm{cyc} \bmod 2$. From b, we can infer how to correctly interpret the value stored for $\mathrm{cnt} \bmod n$, i.e., whether thermometer encoding of cnt is of the form 11..100..0 (even) or 00..011..1 (odd); making us count the leading 1s or 0s. What if the Gray code counter was metastable, and thus b was metastable, too? As before, this does not matter, as the resolution of the counter (and thus b) will induce a difference of 1 in the value of cnt at most (the counter has been triggered or has not).

However, this approach requires us to store TDC readouts as tuples of Gray code and thermometer codes, until they are guaranteed to have stabilized.

We discuss a more efficient encoding in the next section. Observe that this is non-trivial, because the interpretation of the ring latch states depends on the Gray code counter's value.

b) Encoding the Ring Latch States: Also here, we need to make use of a Gray code for safely "compressing" the

TABLE I: Encoding the ring latch states in case $n = 8$: original thermometer encoded ($n - 1$ bits $0 \dots 6$), efficient BRGC encoded ($\log_2(n)$ bits $0 \dots 2$), and the relevant decimal counts of 1s and 0s.

0	1	2	3	4	5	6	0	1	2	#1s	#0s
0	0	0	0	0	0	0	0	0	0	0	7
1	0	0	0	0	0	0	0	0	1	1	
1	1	0	0	0	0	0	0	1	1	2	
1	1	1	0	0	0	0	0	1	0	3	
1	1	1	1	0	0	0	1	1	0	4	
1	1	1	1	1	0	0	1	1	1	5	
1	1	1	1	1	1	0	1	0	1	6	
1	1	1	1	1	1	1	1	0	0	7	0
0	1	1	1	1	1	1	1	0	1		1
0	0	1	1	1	1	1	1	1	1		2
0	0	0	1	1	1	1	1	1	0		3
0	0	0	0	1	1	1	0	1	0		4
0	0	0	0	0	1	1	0	1	1		5
0	0	0	0	0	0	1	0	0	1		6

$(n-1)$-bit thermometer encoding into a $\lceil \log n \rceil$-bit string. The following lemma formalizes the corresponding positive result. It shows that given an arbitrary Gray code (e.g., the binary reflected Gray code, short BRGC), there is a circuit that efficiently translates the thermometer encoded ring state to this Gray code, inducing metastability of at most one bit that is the currently least significant bit.

Lemma 3.5: Consider an arbitrary $\lceil \log n \rceil$-bit Gray code. Suppose that we are given an n-bit ring latch state of the form 11..1M00..0 or 11..100..0. If a possible M is resolved to either 0 or 1, the resulting string is a thermometer encoding of either r or $r+1$ for some $r \in [n]$; if there is no M, the string is a thermometer encoding of $r \in [n+1]$; i.e., it has precision of 1.

There is a purely combinational circuit of at most n XOR gates and depth at most $\lceil \log n \rceil$ that transforms all such inputs into the Gray code, in the sense that in case of metastability in the input, (i) at most one output bit is metastable, and (ii) resolving it to either 0 or 1, the encoded number becomes either r or $r+1$; i.e., the circuit preserves precision of 1.

Proof: Due to lack of space, we only sketch the proof. Up-counting in a Gray code can be viewed as successively negating output bits when thermometer encoded input bits are set to 1. Lines 1–8 in Table I demonstrate this for BRGC and $n = 8$: bit 2 of the BRGC is negated upon setting thermometer bits 0, 2, 4, and 6 to 1. These negations can be implemented by XORs. The resulting circuit is presented in Fig. 9. ■

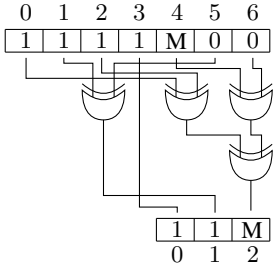

Fig. 9: XOR-tree circuit implementation counting the number of 1s as specified in Table I. Note that a metastable ring latch influences only one, the current least significant, bit (compare lines 4 and 5 in Table I): no precision is lost when encoding metastable data.

As an additional benefit of this approach, note that the same *XOR-tree circuit* can be used to encode the number of 0s in n-bit strings 00..011..1, for the following reason: Switching between encodings 11..100..0 and 00..011..1 is just by negation of the input. Propagating the negations from all inputs through the XOR-tree to the outputs yields that it suffices to negate a fixed subset of the output bits to obtain the Gray code for the complemented input thermometer encoding. In the case of BRGC, only the left-most BRGC bit has to be negated: e.g, see Table I: counting 1s in input 1000000 yields output 001, counting 0s in 0111111 yields 101.

In particular, one can "delay" the application of the respective output negations until after potential metastability of the counter (and thus how the $\lceil \log n \rceil$-bit string is to be interpreted) has been resolved.

In summary, we obtain a memory-efficient metastability-containing TDC.

Corollary 3.6: Using a B-bit Gray code coarse counter and the XOR-tree circuit from Lemma 3.5, a measurement can be stored as a tuple of two Gray codes, using $B + \lceil \log n \rceil$ bits, without losing precision.

c) Metastability-containing Computations: Corollary 3.6 presented a way to efficiently store a, potentially metastable, TDC measurement as a Gray code tuple. While this is memory-optimal, the question arises if we can store a TDC measurement as a *single* Gray code value that does not need any further transformation when being read by an application.

For operations which are metastability-containing, meaning performing the computations *before* metastability has been resolved, having a single BRGC measurement value is of specific interest as there already exist circuits requiring BRGC inputs [2], [5], [9].

Providing such inputs with our approach is not only possible, but straightforward. For n being a power of 2, a single BRGC encoded value of the measurement value can be generated in a very convenient way.

Theorem 3.7: If we use a BRGC coarse counter and n is a power of 2, then *just* the concatenation of (a) the output of the above XOR-tree circuit, *without the need to negate* any of its output bits, and (b) the output of the BRGC counter yields a BRGC encoding of the TDC measurement value.

Proof: We only sketch the proof. BRGC features the symmetry property that the sequences of bits flipped when counting up from the minimum to the maximum encoded value or counting down from the maximum to the minimum encoded value are identical. This property and the recursive structure of the code imply that, in fact, the XOR-tree from Lemma 3.5 outputs the correct bits to concatenate to the Gray counter's value regardless of its current parity. ∎

Fig. 10 depicts the resulting circuit in case of an even coarse counter value. Fig. 11 demonstrates that the same circuit works for odd and for metastable coarse counter values.

We stress the utility of this insight. A naive implementation would require to build an adder with inputs (a) BRGC encoded cyc and (b) thermometer encoded latch outputs, which is a complex circuit. Additionally, this requires both inputs to

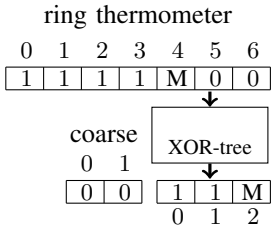

Fig. 10: Circuit encoding a TDC measurement value ($n-1$ thermometer encoded ring latch states and B bit coarse BRGC counter state) as a single BRGC encoded value without losing precision in presence of a metastable input bit. Here $n = 8$ and $B = 2$ and the coarse counter is even (decimal 0), i.e., we have to count 1s in latch states. Observe that the final value 0011M correctly encodes decimal 4 or 5, depending on how M resolves.

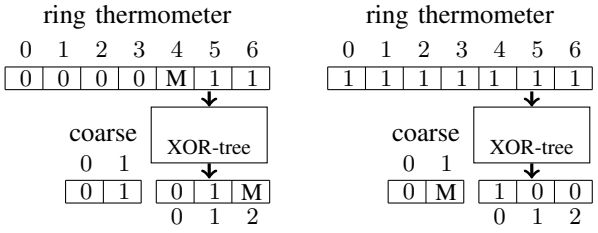

Fig. 11: Left: coarse counter is odd (decimal 1), i.e., we have to count 0s in latch states. Observe that the final value 0101M correctly encodes decimal 12 or 13, depending on how M resolves. Right: coarse counter is metastable. Observe that the final value 0M100 correctly encodes decimal 7 or 8, depending on how M resolves.

be metastability-free or to design a metastability-containing adder. By contrast, concatenation is trivially metastability-containing: no precision is lost due to possible metastability.

3) Solution III: Gray Code Counter with Latch: The Gray code counter variant of the previous section requires that the output of the counter is stable at all times outside a time window of size similar to the critical window of the latches (limited by T_{d}, cf. Fig. 7). For high accuracy TDCs with small T_{d}, this (a) either imposes a harsh constraint on the design of the counter, which is potentially difficult to meet, or (b) requires an implementation with additional registers as a workaround (see Fig. 8).

For these cases, we propose a different metastability-aware design depicted in Fig. 12: We add one extra (single-bit) latch L with output b only. The XOR gate delays the counter increment input and additionally transforms rising and falling transitions to pulses, allowing the use of standard single-edge triggered flip-flops for the Gray code counter implementation.

Analogously to the redundant binary counter solution, we only require non-overlapping critical windows of latch L and Gray code counter C; see Fig. 13.

Given a (stabilized) TDC readout, again we determine

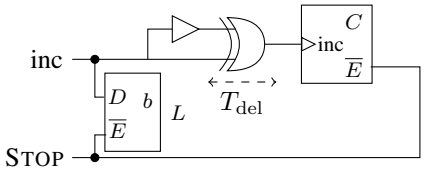

Fig. 12: Solution III: Gray code counter variant with single-bit latch.

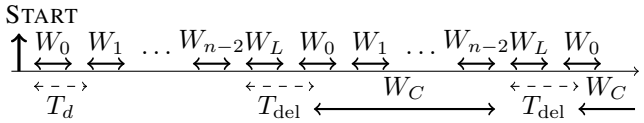

Fig. 13: Critical windows for Solution III.

TABLE II: Cell area [μ^2] as reported by Cadence RTL Compiler

variant	outputs	cell area (absolute/relative)		
		overall	ring	evaluation circuit
single bin.	71	310.69/1.00	236.47/1.00	74.21/1.00
dual bin.	71	378.52/1.22	236.47/1.00	142.04/1.91
Gray	71	320.26/1.03	236.47/1.00	83.79/1.13
Gray + tree	14	412.30/1.33	236.47/1.00	175.83/2.37

whether cyc is odd or even – this time not from the counter value, but from the explicitly stored b in latch L. This enables to correctly interpret the value stored for $\mathrm{cnt} \bmod n$ as before. Moreover, it is used to account for an incomplete up-count of the Gray code counter: if the parity of the stored counter value differs from b, we know that the counter *should* have been incremented, but has been prevented from doing so by the stopping signal. In this case, we amend this by performing an up-count on the stored value (w.r.t. the used Gray code). This results in a correct value, because metastability of the counter affects only the (unique) bit that is being changed on the respective up-count.

Naturally, it may also happen that b becomes metastable. However, in this case, the delay T_{del} ensures that the counter did not start the increment corresponding to the (incomplete) transition of L. Thus, either resolution of b results in a correct interpretation of the measurement: If L stabilizes as if it had not been triggered, all is well; if it stabilizes as if it had been triggered, we fix the counter value accordingly.

IV. Synthesis

To estimate the area requirements of our proposed TDC circuits, we synthesized the different versions using the NanGate 45nm Open Cell Library using the Cadence RTL Compiler. We decided to use a single-shot TDC with a ring size of 63 and a counter width of eight bits. As base line for our comparison a ring TDC with only one binary coarse counter was used. Please note that this circuit may well get metastable, but it is a good reference for a minimal-size implementation. Table II summarizes the number of required outputs and the pre-layout cell area as reported by the RTL compiler. The table shows that the dual counter approach requires a significant area overhead. The gray counter implementation, on the other hand, has a moderate overhead (around three percent). Using an XOR tree increases this area overhead (to approx. 33%), but enables us to compress the width of the output data from 71 to 14 bits without compromising the metastability awareness of the circuit. Due to the smaller output size and the now uniform value format (fully gray encoded instead of a combination of different encodings), the processing of the timestamps will need less area, which ultimately leads to smaller circuits.

V. Conclusion

In TDC applications that require *high-speed sampling* together with *deterministic guarantees on precision*, one must account for metastable upsets and memory-efficient storage of sampled data. Using Spice simulations, we demonstrated that a state-of-the-art ring TDC with coarse binary counter indeed suffers from significant upset rates. As a solution, we compare and formally prove correct three variants of ring TDCs that provide *high-speed sampling*, *optimal guaranteed precision* of 1 delay unit, and *memory-efficient storage* of TDC values. The proposed variants offer different trade-offs between ease of implementation and memory overhead. We stress that variant II uses an encoding of measurement values with zero memory overhead that can be computed *without* waiting for metastability to resolve. This renders it of particular interest for use in metastability-containing circuits [2], [5], [9].

Acknowledgements

Matthias Függer was with Max Planck Institute for Informatics at the time of writing. The research was partially funded by the Austrian Science Fund (FWF) project SIC (P26437).

References

[1] S. Beer and R. Ginosar. Eleven Ways to Boost Your Synchronizer. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 23(6):1040–1049, 2015.

[2] J. Bund, C. Lenzen, and M. Medina. Near-Optimal Metastability-Containing Sorting Networks. In *DATE'17*, 2017.

[3] M. A. Daigneault and J. P. David. A Novel 10 ps Resolution TDC Architecture Implemented in a 130nm Process FPGA. In *NEWCAS'10*, pages 281–284, 2010.

[4] P. Dudek, S. Szczepanski, and J. Hatfield. A high-resolution CMOS time-to-digital converter utilizing a Vernier delay line. *Solid-State Circuits, IEEE Journal of*, 35(2):240–247, 2000.

[5] S. Friedrichs, M. Függer, and C. Lenzen. Metastability-Containing Circuits. *CoRR*, abs/1606.06570, 2016.

[6] S. Henzler. *Time-to-Digital Converters*. Springer, 2010.

[7] J. Kalisz. Review of methods for time interval measurements with picosecond resolution. *Metrologia*, 41(1):17, 2004.

[8] D. J. Kinniment. *Synchronization and Arbitration in Digital Systems*. Wiley Publishing, 2008.

[9] C. Lenzen and M. Medina. Efficient Metastability-Containing Gray Code 2-Sort. In *ASYNC'16*, 2016.

[10] L. Marino. General Theory of Metastable Operation. *IEEE Transactions on Computers*, C-30(2):107–115, 1981.

[11] M. Mota, J. Christiansen, S. Debieux, V. Ryjov, P. Moreira, and A. Marchioro. A flexible multi-channel high-resolution time-to-digital converter ASIC. In *Nuclear Science Symposium Conference Record, IEEE*, volume 2, pages 9/155–9/159 vol.2, 2000.

[12] R. Nutt. Digital Time Intervalometer. *Review of scientific instruments*, 39(9):1342–1345, 1968.

[13] L. Perktold and J. Christiansen. A Flexible 5 ps Bin-Width Timing Core for Next Generation High-Energy-Physics Time-to-Digital Converter Applications. In *PRIME'12*, pages 1–4, June 2012.

[14] L. Perktold and J. Christiansen. A fine time-resolution (<3 ps-rms) time-to-digital converter for highly integrated designs. In *IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1092–1097, May 2013.

[15] C. L. Portmann and T. H. Y. Meng. Power-efficient metastability error reduction in cmos flash a/d converters. *IEEE Journal of Solid-State Circuits*, 31(8):1132–1140, Aug 1996.

[16] T. Rahkonen and J. Kostamovaara. The use of stabilized CMOS delay lines in the digitization of short time intervals. In *ISCAS'91*, pages 2252–2255, 1991.

[17] R. Staszewski, S. Vemulapalli, P. Vallur, J. Wallberg, and P. Balsara. 1.3 V 20 ps time-to-digital converter for frequency synthesis in 90-nm CMOS. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 53(3):220–224, 2006.

[18] H. J. Veendrick. The Behaviour of Flip-Flops Used as Synchronizers and Prediction of their Failure Rate. *Solid-State Circuits, IEEE Journal of*, 15(2):169–176, 1980.