Verifying Global Convergence for a Digital Phase-Locked Loop

Jijie Wei Yan Peng Ge Yu Mark Greenstreet
University of British Columbia

***Abstract*—We present a verification of a digital phase-locked loop (PLL) using the SpaceEx hybrid-systems tool. In particular, we establish global convergence – from any initial state the PLL eventually reaches a state of phase and frequency lock. Having shown that the PLL converges to a small region, traditional methods of circuit analysis based on linear-systems theory can be used to characterize the response of the PLL when in lock. The majority of the verification involves modeling each component of the PLL with piece-wise linear differential inclusions. We show how non-linear transfer functions, quantization error, and other non-idealities can be included in such a model. A limitation of piece-wise linear inclusions is that the linear coefficients for each component must take on fixed values. For real designs, ranges will be specified for these components. We show how a key step of the verification can be generalized to handle interval values for the linear coefficients by using an SMT solver.**

***Index Terms*—analog/mixed-signal verification, digital PLL, global stability, hybrid systems, linear differential inclusions, non-linear systems, SMT.**

I. Introduction

Phase-locked loops (PLLs) are ubiquitous in analog and mixed-signal designs. Their uses include frequency multiplication to generate a high-frequency clock from a lower frequency reference, for clock-acquisition in high-speed links, and as modulators and demodulators for wireless communication. Recently, PLL design has shifted from traditional, analog, charge-pump based designs to various "all-digital" architectures. Several consequences of device scaling to smaller feature sizes have motivated this transition including: greater device-to-device parameter variation impairs designs that depend on matched circuits; lower power supply voltages removes the "voltage headroom" needed for high-quality, on-chip current sources; and the scaling of passive components such as inductors and capacitors lags that for transistors. A failed PLL can block further test of an entire chip or major subsystem; thus, there is a high value in verifying correctness of PLL designs. This paper presents the formal verification of global convergence for the digital PLL published in [1].

Functional verification of analog blocks such as PLLs can be divided into two parts: global convergence to an intended operating point, and small-signal analysis at the operating point. The key insight here is that nearly all analog blocks are *intended* to have some kind of linear response when at or near their intended operating point [2]. Existing analysis techniques such as periodic AC analysis (PAC) [3] are available in standard commercial CAD tools such as Spectre® from Cadence. These techniques allow designers to characterize key performance properties of analog blocks such as the jitter, power-supply sensitivity, and tracking bandwidth of a PLL. A designer can have high confidence in the correct functioning of their block *assuming* that it reaches its intended operating point.

To show that an analog block will reach its intended operating point from any initial state is the *global convergence* problem. Here, the non-linearities of the circuit must be taken into account. Simulation based methods are impractical both because of the impossibility of covering all possible inputs, initial states, and operating conditions, and because individual simulations may need to cover thousands of cycles of the PLL's oscillator to show the locking behaviour. To support power-management techniques such as dynamic voltage-frequency scaling and power down modes, PLLs may need to start-up or change lock frequency tens to hundreds of times per second. If a PLL occasionally fails to lock, then the chip is useless. Tracking down such bugs on real silicon can be extremely difficult. Thus, proving global convergence is of great value for real designs.

In this paper, a digital PLL is modeled as a piecewise-linear hybrid-automaton, and global convergence of the automaton is shown. The first step of this approach is to formulate hybrid-automaton models for each component of the PLL. We note that the basic components of oscillators, dividers, phase comparators, and integrators are common to all PLL designs; thus, we expect this work to be re-usable for verifying other PLL structures. We use the SpaceEx [4] tool to show that all initial states converge to a small region around the intended operating point. To obtain practical verification times, we found it necessary to identify "phases" that the PLL passes through when converging to its operating point. By structuring our hybrid automaton model to reflect these phases, we avoid SpaceEx needing to perform costly fix point computations.

A limitation of the SpaceEx based approach is that the model parameters of the piece-wise linear inclusions are fixed. This requires giving specific values to some analog quantitities that a designer can only guarantee to be in some range. We show that for the PLL from [1], a global Lyapunov function (i.e. progress function) can be constructed using basic methods from linear systems theory. We then use the Z3 SMT solver [5] to show global convergence for a simplified model of the DPLL.

The key contributions of this paper are:

- We verify global convergence for a digital PLL. Another approach to digital PLL verification that was developed independently was recently reported in [6]. We believe

ISBN 978-0-9835678-3-7/13.

that these are the first such verifications for digital PLLs.
- We show how each component of the digital PLL can be modeled as a hybrid automaton. Our models account for non-linearities of the components, quantization, and other non-idealities.
- We demonstrate how convergence can be shown by reachability analysis (using SpaceEx) and by solving systems on non-linear inequalities (using Z3).

II. Related Work

There have been several previously published reports of PLL verification using formal methods. The earliest verification that we know of was by [7, chap. 6]. Dhingra's design uses a fixed-frequency oscillator that is intended to operate at N times the frequency of a reference. The PLL adaptively chooses edges of its internal oscillator to approximate the edges of the lower frequency reference. The time resolution is limited by the period of the internal oscillator. While this may be useful for low frequency applications such as audio frequency modems, we are not aware of any such PLLs in use for the more standard PLL applications of clock generation, clock-data-recovery, and wireless communication. Dhingra verified the tracking behaviour of his design using the HOL theorem prover.

More recently, Dong *et al.* [8], [9] proposed using property checking for AMS verification, including PLLs. They used "symbolic recurrence equations" as a property specification language, and show how this can be used to automatically construct a monitor to check simulation runs to see if a PLL locks in the required time for that run. This does not address the problems of long-simulation times to show that a PLL locks or the incompleteness of simulation based approaches to show convergence.

Shortly after the work by Dong *et al.*, Jesser and Hedrich [10] described a model-checking result for an analog PLL with an XOR-gate phase detector. They performed symbolic model checking using MTBDDs to represent both the discrete and analog parts. They state that the four-dimensional analog state space is partitioned into 2^{11} hyperboxes, and that next-box relations are determined by random simulation. It is not clear how they guarantee the complete coverage with this approach.

Two years ago, Althoff *et al.* [11] presented the verification of a charge-pump PLL using an approach that they refer to as "continuization." They use a purely linear model for the components of their PLL, and their focus is on the switching activities of the phase-frequency detector, in particular, uncertainties in switching delays. Their approach also verifies the PLL for ranges of component parameters. We present an SMT-based technique for handling interval parameters in Section V. Althoff *et al.* is the only other work that we are aware of that accounts for such variation.

More recently, Lin et al. [6] independently developed an approach for verifying a digital PLL using SMT techniques. To the best of our knowledge, they are the first to claim formal verification of a digital PLL. Similar to the approach presented in this paper, they consider a purely linear, analog model and then reason about the discrepancies between this idealized model and a digital implementation. They use the KRR SMT solver to verify bounds on this discrepancy. They verify bounds on the lock time of a digitally intensive PLL assuming that most of the digital variables are initialized to fixed values, and that only the oscillator phase is unknown. Our work shows initialization for a different PLL design over the complete state space.

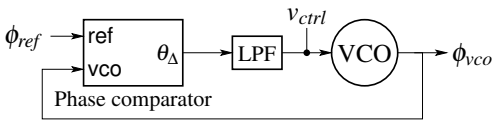


Fig. 1. A Simple PLL

III. The Digital PLL

A. A PLL Primer

The function of a phase-locked loop (PLL) is to adjust the PLL's oscillator so that it tracks the frequency and phase of a reference signal. Figure 1 shows a simple PLL. The PLL sets the control voltage, v_{ctrl} of the VCO according to the phase difference between the VCO and the reference and the integral of this difference to match the VCO's frequency to that of the reference and align their phases. Simply, if the PLL's oscillator lags the reference, then v_{ctrl} increases; and the VCO frequency increases so that the VCO will catch up with the reference. Conversely, if the PLL's oscillator is ahead of the reference, then v_{ctrl} will decrease causing the PLL's oscillator frequency to decrease.

In more detail, the voltage-controlled oscillator (VCO) can be understood as a voltage-to-frequency converter. Phase is the integral of frequency; so we can express the phase of the VCO output as $\theta_{vco} = (\int v_{ctrl} dt) \bmod [-\pi, +\pi)$. Phase is modular, and we write $\theta \bmod [-\pi, +\pi)$ to indicate the value in $[-\pi, +\pi)$ that is congruent with θ modulo 2π radians. The phase comparator generates an output voltage that is proportional to the phase difference of its inputs: $\theta_\Delta = (\theta_{vco} - \theta_{ref}) \bmod [-\pi, +\pi]$. The reference is assumed to have a constant frequency, ω_{ref}; thus $\theta_{ref} = \omega_{ref} t$, where t is time. The low pass filter implements the integral and proportional correction terms with $v_{ctrl} = a_0 \theta_\Delta + a_1 \int \theta_\Delta dt$. Combining these equations and differentiating twice, we get:

$$\theta_{vco} \quad = \quad a_0 \int \theta_\Delta dt + a_1 \int\int \theta_\Delta dt\, dt \qquad (1)$$

In the simple case where $a_0 = 0$ and $a_1 = 0$, the PLL becomes a simple harmonic oscillator. The frequency of the PLL oscillates with mean value of ω_{ref} but never converges. If both a_0 and a_1 are negative, then there is a unique solution where the PLL's oscillator converges to the frequency and phase of the reference. Note that if all of the components are linear, then simple algebraic techniques suffice to show (or refute) global convergence.

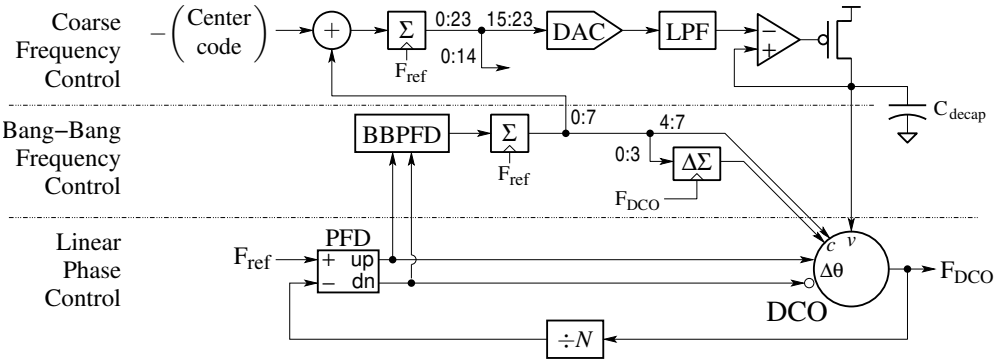
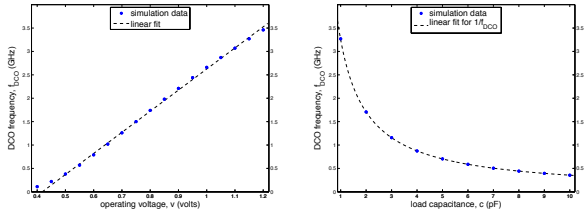


Fig. 2. The digital PLL from [1]

For real designs, the components are not perfectly linear. The component may very closely match their linear idealizations when the PLL has locked to the reference, but significant non-linearities may occur when out of lock. Furthermore, the analog components of the simple PLL are difficult to implement in advanced fabrication technologies. For example, large capacitors are needed to implement the integrator part of the low pass filter. For these and other reasons, designers are making more and more extensive use of digital and mixed signal designs for PLLs.



(a) frequency vs. operating voltage (b) frequency vs. load capacitance

Fig. 3. Ring-oscillator response

B. The Digital PLL

Figure 2 shows the digital PLL architecture from [1]. While this real-world PLL has many more components than the simple PLL from Fig. 1, its operation is similar. The digitally controlled oscillator (DCO) performs the role of the VCO from the simple PLL. The divider, $\div N$ is added to make the lock-point for the DCO a multiple of the reference frequency. The phase comparator of the simple PLL is replaced by two phase-frequency detectors – a linear PFD that reports the phase difference of the reference and the frequency divided DCO, and a "bang-bang" PFD that only reports the sign of this phase difference. The remaining components implement the low-pass filter of the simple PLL with the accumulators functioning as integrators.

The three control paths of the PLL (linear phase, bang-bang frequency, and coarse frequency) work together to set the frequency, f_{dco}, of the digitally controlled oscillator (DCO) to N times the reference frequency, f_{ref}, and to align the phase of the DCO and the reference (i.e., rising edges of the reference clock should coincide with rising edges of the DCO).

The digitally-controlled oscillator in [1] is a three-stage ring-oscillator with three control inputs: v, c, and θ_Δ. The v input sets the operating voltage of the DCO. First-order transistor models suggest that f_{dco} should be roughly proportional to v. Figure 3(a) shows the results of Spectre® simulations of a ring-oscillator in a 65nm CMOS process with a 1.2V nominal V_{dd}. For an operating voltage v with $0.5 \leq v \leq 1.2$, a linear fit provides a good approximation of the DCO frequency.

The c input controls switches that add capacitors to increase the load for each stage of the ring oscillator. As seen in Fig. 3(b), the oscillator period (inverse frequency) is very accurately modeled as a linear function of load capacitance. In addition to the four, binary weighted, values from c, a fifth bit with weight equal to the least significant bit, is provided from the Delta-Sigma modulator. This provides a period averaging to improve the frequency resolution of the bang-bang control path.

Note that the DCO frequency, f_{dco} is proportional to a linear function of v and *inversely proportional* to a linear function of c. We can write this as

$$f_{dco} = \frac{1+\alpha v}{1+\beta c} f_0 \qquad (2)$$

where f_0 is the frequency of the DCO (extrapolated) to where v and c are at zero, and α and β are the sensitivities to v and c respectively. Most importantly, when modeling the response to both v and c, either their ranges must be quite small, or the model is inherently non-linear. To show global convergence, the non-linearity of the DCO must be included in the model.

The θ_Δ input controls a linear phase path. Each stage of the ring-oscillator consists of an inverter in parallel with two tri-state inverters. One of the tri-state inverters is *enabled* when up is asserted, and the other is *disabled* when dn is asserted. This causes the oscillator to run faster when up is asserted without dn, and slower when dn is asserted without up. This produces a phase shift advance (resp. delay) to the length of time that the oscillator is in its fast (resp. slow) mode.

The frequency divider, labeled $\div N$ in figure 2 is a modulo-N counter. Its output has a period N times that of the DCO.

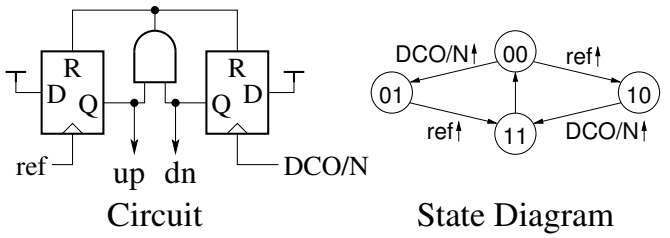


Fig. 4. The PFD

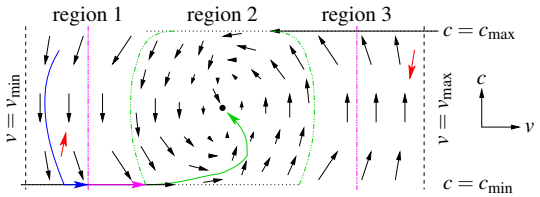


Fig. 5. Operation with saturating integrators based on linear model from Eq. 3, but not to scale.

The two phase-frequency detectors (PFD and BBPFD) in the diagram determine if the DCO should increase or decrease its frequency. Figure 4 shows a typical circuit for the linear phase-frequency detector, labeled PFD in Fig. 2, and its state transition diagram. Each state is labeled with the value of the up and dn (down) outputs of the PFD – for example, in state 01 the up signal is low and the dn signal is high. The PFD is reset to state 00 after it has received rising edges from both the reference clock, ref, and the frequency divided DCO, DCO/N. If the next event is a rising edge from the divided DCO, this indicates that the DCO is ahead of the reference in phase or running at a higher frequency. Accordingly, the PFD enters state 01 (asserting dn) indicating that the DCO should decrease in frequency (equivalently, drop back in phase). Likewise, if the first event after resetting is a rising edge of the reference, then up is asserted. Note that the *duration* of asserting dn without up (or vice-versa) indicates the phase difference between the reference and DCO. This is the principle behind the linear-phase control path of the PLL.

The BBPFD is a so-called "bang-bang" phase frequency detector. It simply detects whether the linear PFD asserts up before dn or dn before up and then outputs $+1$ or -1 respectively. The output of the BBPFD indicates the direction that the DCO should move, but does not contain the magnitude information of the linear PFD.

The accumulators implement integrators. The integrator in the bang-bang path directly controls the c input of the DCO. The Delta-Sigma modulator provides an averaging: if the lower four bits of the accumulator for c encode a value of k, then the output of the Delta-Sigma modulator will be asserted for k out of 16 cycles of the DCO. The integrator in the coarse frequency control path is designed to have low-bandwidth to ensure low jitter (cycle-to-cycle variation of the DCO period). The output of the integrator is converted to a voltage with the digital-to-analog converter (DAC). To suppress the coupling of power-supply noise into the DCO, an additional low-pass filter and linear regulator are included in the loop.

To understand the operation of the digital PLL when $\frac{1}{N} f_{dco} \neq f_{ref}$, the difference will be noted by the PFD and the BBPFD. The BBPFD will output a value to drive c in the direction to correct for the frequency difference. For small frequency differences, this bang-bang loop provides fast tracking. For larger differences, the accumulator for c will saturate at its minimum or maximum value. Any of these situations lead to c being different from the center code. This drives the coarse frequency loop to change v in the direction needed to bring $\frac{1}{N} f_{dco}$ to f_{ref} and to return c to the center code value. Because the DCO has only a fixed set of oscillation frequencies, for most values of f_{ref} there is no choice of c and v that causes $\frac{1}{N} f_{dco}$ to be *exactly* f_{ref}. This leads to limit-cycle oscillations. The linear phase control path suppresses these oscillations.

IV. Verifying Convergence with SpaceEx

We divided our verification effort into two main tasks: designer and verifier. One of us (the "designer") wrote Matlab models of the components (such as the DCO, phase-frequency detectors, etc.) and used these for simulations. The other person was the "verifier" who translated these models into hybrid automata and used SpaceEx to verify global convergence. We found it very helpful to start with a simple, completely linear model that reflected the structure of digital PLL and gradually refine it and add details to faithfully model the actual PLL.

A. Modeling the digital PLL

A simple linear model. Let

$$\begin{array}{rcl} f_{dco} & = & \alpha v - \beta c \\ \frac{d}{dt}\theta_\Delta & = & \frac{1}{N} f_{dco} - f_{ref} - g_\theta \theta_\Delta \\ \frac{d}{dt} c & = & -g_1 \theta_\Delta \\ \frac{d}{dt} v & = & g_2 (c - c_{center}) \end{array} \tag{3}$$

where α, β, N, and θ_Δ are as described in Section III; g_θ is the "time gain' for the linear phase path; g_1 is the integrator gain for c; g_2 is the integrator gain of the v path. Because the model is linear, global convergence can be shown (or refuted) by simple, analytical methods. If $g_\theta > 0$, $g_1 < 0$, and $g_2 < 0$, then this linear PLL converges globally.

Modeling the accumulators. The accumulators of the digital PLL approximate the integrators of the linear model by computing a Reimann sum on rounded values of the integrand. The accumulators add perturbations due to quantization and saturation. Saturation of the accumulators is modeled by providing bounds for c and v: $c_{\min} \leq c \leq c_{\max}$ and $v_{\min} \leq v \leq v_{\max}$. The model is based on Eq. 3 with the change that if c reaches $c_{\min}$ and $\dot{c} < 0$ by Eq. 3, then $\dot{c} = 0$ in this "saturating" model. Likewise for the cases when $c = c_{\max}$ or when v reaches one of its bounds.

Simulations of the Matlab model with saturating integrators showed the behaviour depicted in Fig. 5. The colored path shows a typical trajectory, and the red arrows show an artifact that is can be caused by the internal delays of the PFD that will be discussed later in this section. In region 1, v is too low to achieve $\frac{1}{N} f_{dco} = f_{ref}$. In this case, c reaches its saturation value of $c_{\min}$ (the blue curved path), and then v increases (the blue and magenta arrow)until $\frac{1}{N} f_{dco} \approx f_{ref}$. At this point $\dot{c} > 0$ and the trajectory enters region 2. Trajectories in region 2 asymptotically approach the equilibrium point (the curved green path) without further saturation of c or v. In region 3, v is too high, and c saturates at $c_{\min}$ until the trajectory enters region 2. The corresponding hybrid automaton has seven modes: four for saturated values of c or v, and one for each of the regions described above. Again, SpaceEx readily showed global convergence.

The observation that the phase locked loop first saturates c, then gets v close to its final value, and the converges in both c and v applies to the actual PLL as well as to this simplified model. This observation allowed us to describe the PLL in a way where SpaceEx shows the convergence of one variable at a time. In the course of verifying the PLL, cycles in the mode-transition graph would cause time-outs while SpaceEx tried to compute fix points. The "one variable at a time" approach eliminated the most egregious of these cycles from the model and achieved very practical execution times.

The SpaceEx model approximates the values of the accumulators of the digital PLL with integrators. Thus, the error analysis is basically that for a Riemann sum approximation of an integral, but in this case the integral is approximating the sum rather than the other way around. Let $f : \mathbb{R}^+ \to \mathbb{R}$ be an integrable function such that for all $t \geq 0$. $|f(t)| \leq F$ and $\left|\frac{d}{dt} f(t)\right| \leq D$ for some $F, D \in \mathbb{R}^+$. Let $\text{round}(x, \gamma)$ denote the rounding of x to the nearest integer multiple of γ, for any $\gamma \geq 0$. Then for $\Delta T > 0$

$$\Delta T \sum_{k=0}^{N-1} \text{round}(f(k\Delta T), \gamma) \;=\; \int_0^{N\Delta T} f(u) + \mu(u)\, du + \rho(t) \qquad (4)$$

for some $\mu, \rho : \mathbb{R}^+ \to \mathbb{R}$ with $|\mu(t)| \leq \frac{\gamma}{2} + D\Delta T$ and $|\rho(t)| \leq F\Delta T + \frac{\gamma}{2}$ for all $t \geq 0$. Equation 4 provides error bounds for approximating the values output by the digital accumulators with continuous integrators. For the digital PLL design, the integrand for the c-integrator is either $+1$ or -1, and its discretization is exact. Furthermore, the discretization for the values of c and v are accounted for by the ρ terms of their integrators. Hence, we can simplify Equation 4 and let $\mu = 0$ for the digital PLL model. SpaceEx supports linear differential inclusions, so, the ρ functions are easily added to the model for computing c and v. Once again, SpaceEx quickly establishes global convergence.

Non-linearities of the DCO. As described in Section III, the DCO is fundamentally non-linear in its response to its control inputs, c and v. For our model, we considered c in a range of 0.9 to 1.1 and v in a range of 0.1 to 2.5 (arbitrary units). The range of c matches what we could infer from [1]. The range

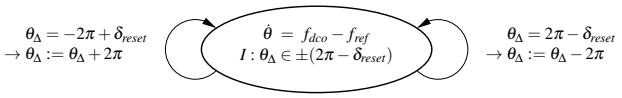


Fig. 6. A hybrid automaton for the linear PFD

of v is definitely wide; we chose it to show the flexibility of our approach. We divided v into seven overlapping intervals: when a trajectory leaves one region it arrives in the *center* of next interval – this prevents chattering mode-transitions that would cause SpaceEx to time out. For each interval, we computed a linear approximation for f_{dco} as a function of c and v, bounded the error, and incorporated the error terms into the linear differential inclusions for $\dot{c}$.

The bang-bang PFD. First, consider the linear PFD. It can detect phase differences of up to nearly a full clock period in either direction. For example, if the up signal goes high nearly a clock period before the dn signal, that indicates that the DCO is nearly 2π radians behind the reference. At the other extreme, the dn signal goes high nearly a clock period before the up signal to indicate that the DCO is nearly 2π radians ahead. Figure 6 shows a hybrid automaton model for the linear PFD.

Note that during the time that the reset signals are asserted (see Fig. 4), the PFD may fail to acquire an edge of one of the clocks.

A simple model of the digital PLL could be obtained by creating the product automaton from each of the component automata described above. To verify global convergence, it suffices to show that this product automaton converges to correct phase and frequency lock from all initial states. However, this results in a *huge* number of mode transitions. The key issue is that while the output of the PFD could have a mode transition for nearly every cycle, the value of v changes quite slowly. Thus, SpaceEx would need to analyse a large number of mode changes before v settles; in practice, this results in a time out. Furthermore, SpaceEx adds an error-term in all directions to the reachable space with each mode transition. Because v changes very little between mode transitions of the PFD, these error-terms overwhelm the convergence of v. To avoid these limitations, we created a model for the digital PLL that consisted of three sub-models according to how "far" the PLL is from its lock state.

Model 1: (the blue path segment in Fig. 5). This model is for the region where v is much lower than the equilibrium value. We construct a model that has c and θ_Δ as state variable and models v as a static interval. A complication to this argument is that due to delays in the reset loop of the PFD, the PFD may occasionally report that the DCO leads the reference, causing c to temporarily increase – these anomalous flows are depicted by the red arrows in Fig. 5. These anomalies are captured by our model, and SpaceEx shows that c moves to a small, invariant interval containing $c_{\min}$. Because $c < c_{center}$, $\dot{v} > 0$, from which we conclude that v increases, and the entry conditions for model 2 will eventually be satisfied. We use

an equivalent construction when v is much greater than the equilibrium value. Note that v must be analysed separately from c and θ_Δ to avoid the issues with different time scales for v and θ_Δ.

Model 2: (the magenta path segment in Fig. 5). For v in these bounds, the linear phase path bounds θ_Δ and ensures that there are no anomalies like those described for Model 1. Here we use the product-automata construction described earlier; c, v, and θ_Δ are all included as state variables. SpaceEx shows that v continues to progress towards its equilibrium value, and in so doing verifies our choice of bounds for v. In other words, our manual calculation was helpful to obtain a successful verification, but the soundness of the verification does not depend on the correctness of these calculations.

Model 3: (the green path segment in Fig. 5). Here, $\frac{1}{N} f_{dco} = f_{ref}$, and c and v follow a zig-zag path to settle at their equilibrium values. Along this path, θ_Δ frequently changes sign, causing a large number of mode transitions that would obscure the progress of v in the SpaceEx analysis. Our linearized model for the DCO frequency is

$$f_{dco} \quad \in \quad a_v v + a_c c \oplus Err \tag{5}$$

where $\oplus$ denotes Minkowski sum[1], and Err is an error-bound interval for the linear approximation.

$$w \quad = \quad \frac{a_v v + a_c c}{N} - f_{ref} \tag{6}$$

and construct a model whose state variables are w and θ_Δ. SpaceEx readily shows that w and θ_Δ both converge to intervals around 0.

Now, note that if $|w|$ is small, then given v, we can derive tight bounds for c. This allows us to construct a model, using a small interval for w, that shows that v (and therefore c) converges to its equilibrium value.

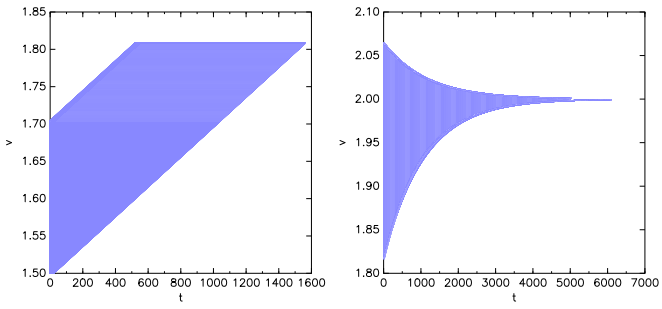
Together, these results from SpaceEx show that all trajectories that start in the valid region for model 1 eventually enter the valid region for model 2. All trajectories that start in the valid region for model 2, eventually enter the valid region for model 3. All trajectories that start in the valid region for model 3, converge to the desired equilibrium point. Because the union of the valid regions for models 1, 2, and 3 covers the entire state space for c, v, and θ_Δ, global convergence of the digital PLL is verified. As an example of the process, Figure 7 shows how v converges to its equilibrium value in models 2 and 3.

B. Limitations of the model

Our verification is relative to the model, and the model makes several simplifications relative to the real circuit. This section summarizes the most important of these simplifications.

[1] The Minkowski sum of two sets, A and B is the set of elements that can be obtained as the sum of an element from A and an element from B:

$$A \oplus B \quad = \quad \{z \mid \exists a \in A.\ \exists b \in B.\ z = a + b\}$$



v vs. t in Model 2 v vs. t in Model 3

Fig. 7. SpaceEx plots showing convergence of v for Models 2 and 3

Our model omits the Delta-Sigma modulator, the direct phase control path and the low-pass filter of the complete design. **Modeling the Delta-Sigma modulator** should be straightforward using the quantization model from Eq. 4.

The linear phase control path. We included a simple, linear model of the "time gain" of the linear phase control in our model. The phase shift of this path is applied once for every cycle of the PFD. This means that the phase shift is proportional to both the phase offset and the minimum of f_{ref} and $\frac{1}{N} f_{dco}$. Our linear model is valid when the PLL is close to lock, and we plan to model the non-linearity of this path in future work.

The low-pass filter seemed like an obvious candidate to include in our model for SpaceEx: we can model it as a purely linear system with three state variables and no mode transitions. When we included the filter in the PLL model, SpaceEx failed to show convergence. We suspect that the filter's low cut-off frequency results in a stiff model. We hope to explore this further and possibly along the lines of those presented in [12] to model the PLL with the low-pass filter.. An interesting opportunity here is that if the cut-off frequency of the low pass filter is close to that of the v-integrator, the PLL will be unstable. We intend to use this as a test case to show how our methods can identify a faulty design.

The PFD has a metastability issue that is hidden by the abstraction that we used. Basically, there are situations where the PFD must "decide" between reseting θ_Δ to 0 or continuing with a value that is close to $\pm 2\pi$. While the issue of metastability occurs in any PLL design, we have not seen it mentioned or addressed in the verification literature.

Finally, our model has **fixed coefficients** for the linear differential inclusions that model the PLL components. A real design will not exactly match any pre-specified value for these parameters, and they will be specified as intervals instead. SpaceEx only supports models where the coefficients are fixed, real numbers. In the next section, we introduce an approach that allows verifying global convergence under the more general realistic condition of having interval bounds for key model parameters.

V. Parameterized Verification with Z3

Consider the problem of showing that all trajectories in an invariant region Q_0 eventually reach a target region Q_T with $Q_T \subseteq Q_0$. This can be proven by exhibiting a *Lyapunov function*, Φ, that satisfies the two conditions below:

1) $\forall x \in Q_0 - Q_T.\ \Phi(x) > 0$; and
2) $\forall x \in Q_0 - Q_T.\ \frac{d}{dt}\Phi(x) < 0$.

Because Q_0 is invariant, all trajectories that start in Q_0 will remain in Q_0 forever. Furthermore, if the trajectory has not entered Q_T, $\Phi(x)$ strictly decreases with time along any trajectory outside of Q_T. Therefore, $\Phi(x)$ must eventually be less than or equal to zero. Because $\Phi(x)$ is strictly positive outside of Q_T, the trajectory must eventually enter Q_T.

The soundness of the Lyapunov argument does not depend on how Φ was obtained; it only requires that Φ satisfy the Lyapunov conditions. In this section, we borrow an approach from linear systems theory to construct a Lyapunov function, and use the Z3 SMT solver to show that the Lyapunov conditions stated above hold for this function when used with the non-linear model for the digital PLL. By using this approach with interval bounds for key model parameters, we show that the verification holds for *any* digital PLL whose components implement the model within the given bounds. We observed that a direct application of this method produced a system of non-linear relations where the SMT solver did not terminate in a practical amount of time. However, we can modify the original function to produce a new function that Z3 can show satisfies the Lyapunov conditions above. This verifies global convergence for any implementation of the PLL whose parameters are within the interval bounds.

As a preliminary experiment, we considered showing convergece from states where $\frac{1}{N}f_{dco}$ is much different than f_{ref}. In this case, the PFD acts like a frequency comparator, and we considered a simplification of Eq. 3 without θ and where $\dot{c} = g_1(f_{ref} - \frac{1}{N}f_{dco})$. Here, we use a non-linear DCO from Eq. 2.

To construct Φ, first consider a linear system, $\dot{x} = Ax$. Let P be the solution of

$$A^T P + PA \quad = \quad -I \tag{7}$$

By construction, P is symmetric. If P is positive definite, then the system $\dot{x} = Ax$ globally converges to $x = 0$ [13, p. 154]. To prove this, observe that $\Phi(x) = x^T Px$ satisfies the Lyapunov conditions.

Next, consider the PLL model with a non-linear DCO model and saturating accumulators as described in the previous section. Let h be the time-derivative function for this model, in other words, $\dot{x} = h(x)$ where $x = [c\ \ v]^T$. Let x_0 be the desired operating point of the PLL; in particular $h(x_0) = 0$. To show global convergence, let $A = Jac(h, x_0)$ be the Jacobian of h when evaluated at x_0; let P be defined as in Eq. 7; and let $\Phi(x) = x^T Px$. The PLL globally converges to x_0 if P is positive definite and for all initial points $x \in Q_0 - \{x_0\}$, $\frac{d}{dt}\Phi(x) < 0$. Positive-definiteness can be tested by adding a conjunct with that constraint to the solver and showing that a suitable P exists. The second part is equivalent to showing $\forall x \in Q_0 - \{x_0\}.\ h(x)^T Px < 0$. Z3 solves this problem in a few seconds, including the multiple cases in the definition of h to account for saturation of the accumulators.

We attempted to repeat this analysis using interval bounds for the parameters α, β, and f_0 for the DCO from Eq. 2, allowing each parameter to vary $\pm 20\%$ from its nominal value. With our initial attempt, Z3's solver failed to complete. While modern SMT solvers can handle non-linear relations, the computational cost grows extremely rapidly with the number of non-linear terms. Accordingly, we sought to simplify our Lyapunov function. Using the Jacobian based approach defined above:

$$A_0 \quad = \quad f_0 \begin{bmatrix} \frac{g_1 f_{ref}\beta}{1+\beta c_{center}} & -\frac{g_1 \alpha}{1+\beta c_{center}} \\ g_2 & 0 \end{bmatrix} \tag{8}$$

As described above, one can propose any matrix for A, and if function obtained by solving for P satisfies the Lyapunov conditions, global convergence is established. Thus, we looked for ways to "simplify" A_0 to obtain a system of inequalities that would show convergence and be tractable in Z3. Noting that the nominal values for α and β are both one, we factored them out from the numerators in the elements for the first row of A_0 to get

$$A_1 \quad = \quad f_0 \begin{bmatrix} \frac{g_1 f_{ref}}{1+\beta c_{center}} & -\frac{g_1}{1+\beta c_{center}} \\ g_2 & 0 \end{bmatrix} \tag{9}$$

With this change, Z3 verified the Lyapunov conditions, again in just a few seconds.

Now, consider adding error terms as described in Section IV to the derivatives to obtain an inclusion. These error terms perturbed the effective values of c and v in the calculation of the derivative. Let Err denote these error terms, and assume that Err is symmetric about 0: if $\eta \in Err$ then $-\eta \in Err$ as well. We now want to show:

$$\forall X \in Q_0 - Q_T.\ \forall \eta \in Err.\ h(x+\eta)^T Px < 0$$

From the symmetry of Err, this is equivalent to showing

$$\begin{array}{l} \forall X \in (Q_0 - Q_T) \oplus Err.\ \forall \eta \in Err. \\ \quad (x + \eta \in Q_0 - Q_T) \Rightarrow (h(x)^T P(x+\eta) < 0) \end{array} \tag{10}$$

The last form is easier for the SMT solver because it moves the η term out of the non-linear function, h. With the earlier models, Z3 showed convergence to the point x_0. With this model, Z3 shows convergence into a small rectangle that contains x_0. This rectangle is larger than $x_0 \oplus E$ because the disturbance can be time-varying.

Finally, we combined using interval bounds for the model parameters and including error terms in the derivative function. Again, Z3's solver failed to converge. This time we noted that the denominator of the elements in the first row of A_0, $1 + \beta c_{center}$ is always positive. Thus, we can multiply the second inequality of the Lyapunov conditions by $1 + \beta c_{center}$ to obtain the equivalent condition:

$$\begin{array}{l} \forall x \in (Q_0 - Q_T) \oplus Err.\ \forall \eta \in Err.\ (x + \eta \in Q_0 - Q_T) \\ \quad \Rightarrow ((1 + \beta c_{center}) h(x)^T P(x+\eta) < 0) \end{array} \tag{11}$$

Using this formulation, Z3 quickly verified global convergence with interval bounds for model parameters and error terms in the derivative function.

VI. Conclusions and Future Work

We have shown global convergence for a digital phase locked loop (PLL). We modeled the components of the PLL using piecewise linear differential inclusions, and then showed that all initial states converge to a small region near the intended operating point. These component models included non-linearities in the digitally controlled oscillator, saturation and quantization effects in the accumulators, and modeling of both a linear and a bang-bang phase-frequency detectors. Using a simplified model, we showed how the convergence results can be extended to the case where the specifications for components are given as interval bounds rather than exact values.

We chose to use SpaceEx [4] for the reachability computations because it was designed from the outset to handle large linear hybrid automaton models. The piecewise linear inclusions model the PLL components quite well. On the other hand, we encountered problems with long compute times and large over approximations when SpaceEx computed non-trivial fix points for cycles of modes. The solution we found was to organize the modes of the automaton according to the typical behaviour of the PLL during lock to avoid cycles of modes. With thise changes SpaceEx could verify global convergence in a few minutes.

SpaceEx requires fixed values for the model coefficients. We also showed the convergence can be established using Lyapunov functions, and the correctness of these functions can be shown with an SMT solver. For the work reported here, we used Z3 [5]. Here too, we encountered issues of time-outs: the solver would either complete in a second or two, or they would go on for hours without terminating. In this case, the solution was to manually simplify the function. This works particularly well with the Lyapunov approach; there's no way to introduce an error by simplifying a proposed Lyapunov function. If an inappropriate change is made, the proposed function will be refuted. Our SMT-based method is at a relatively preliminary stage and we are interested in seeing if we can apply it to a model that is as detailed as the one that we used with the hybrid-automata appproach.

There are many areas for future work. We would like to provide bounds on lock time (excluding metastability). Then we plan to complete models for the low-pass filter and the Delta-Sigma modulator. We plan to examine other digital PLL architectures to assess how much of the effort from this verification can be re-used for other designs. We expect that the re-use will be large, but we do not expect full automation given the need to guide the tools away from problems of time-outs and over approximations.

A very promising follow-on is to formalize the connection between the models used here and those used in other phases of the analog and mixed-signal design process. For example, we used "designer" provided models of the main components of the PLL. How do we know that these handwritten models correspond to the actual circuit? Thus, we want to connect this work with component validation.

Acknowledgments

We appreciate feedback from designers who have given us feedback on PLL design and verification, especially Elad Alon, Brian Casper, Frankie Liu, Frank O'Mahony, and Suwen Yang. This work has been supported by grants from Intel and from NSERC Canada.

References

[1] J. Crossley, E. Naviasky, and E. Alon, "An energy-efficient ring-oscillator digital PLL," in *Proceedings of the Custom Integrated Circuits Conference (CICC'2010)*, Sep. 2010. [Online]. Available: http://dx.doi.org/10.1109/CICC.2010.5617417

[2] J. Kim, M. Jeeradit, B. Lim, and M. A. Horowitz, "Leveraging designer's intent: a path toward simpler analog CAD tools," in *Proceedings of the Custom Integrated Circuits Conference (CICC'2009)*, Sep. 2009, pp. 613–620. [Online]. Available: http://dx.doi.org/10.1109/CICC.2009.5280741

[3] K. S. Kundert, "Introduction to RF simulation and its application," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 9, pp. 1298–1319, 1999. [Online]. Available: http://dx.doi.org/10.1109/4.782091

[4] G. Frehse, C. L. Guernic *et al.*, "SpaceEx: Scalable verification of hybrid systems," in *Proceedings of the 23rd Conference on Computer Aided Verification*, 2011. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-22110-1_30

[5] L. Moura and N. Bjørner, "Z3: An efficient SMT solver," in *Tools and Algorithms for the Construction and Analysis of Systems*, ser. Lecture Notes in Computer Science, C. Ramakrishnan and J. Rehof, Eds., vol. 4963. Springer Berlin Heidelberg, 2008, pp. 337–340. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-78800-3_24

[6] H. Lin, P. Li, and C. J. Myers, "Verification of digitally-intensive analog circuits via kernel ridge regression and hybrid reachability analysis," in *Proceedings of the 50th Annual Design Automation Conference*, ser. DAC '13. New York, NY, USA: ACM, 2013, pp. 66:1–66:6. [Online]. Available: http://doi.acm.org/10.1145/2463209.2488814

[7] I.-S. Dhingra, "Formalising an integrated circuit design style in higher order logic," Ph.D. dissertation, Computer Laboratory, Cambridge University, Nov. 1988. [Online]. Available: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-151.html

[8] Z. J. Dong, M. H. Zaki, G. Al-Sammane, S. Tahar, and G. Bois, "Checking properties of PLL designs using run-time verification," in *International Conference on Microelectronics (ICM'2007)*, 2007, pp. 125–128. [Online]. Available: http://dx.doi.org/10.1109/ICM.2007.4497676

[9] Z. Wang, N. Abbasi, R. Narayanan, M. Zaki, G. Al-Sammane, and S. Tahar, "Verification of analog and mixed signal designs using online monitoring," in *Mixed-Signals, Sensors, and Systems Test Workshop, 2009. IMS3TW '09. IEEE 15th International*, 2009, pp. 1–8. [Online]. Available: http://dx.doi.org/10.1109/IMS3TW.2009.5158695

[10] A. Jesser and L. Hedrich, "A symbolic approach for mixed-signal model checking," in *Proceedings of the 2008 Asia and South Pacific design automation conference (ASPDAC'08)*. Los Alamitos, CA, USA: IEEE Computer Society Press, 2008, pp. 404–409. [Online]. Available: http://dl.acm.org/citation.cfm?id=1356802.1356903

[11] M. Althoff, A. Rajhans *et al.*, "Formal verification of phase-locked loops using reachability analysis and continuization," in *Proceedings of the 2011 International Conference on Computer Aided Design*, Nov. 2011, pp. 659–666.

[12] C. Yan, M. R. Greenstreet, and J. Eisinger, "Formal verification of an arbiter circuit," in *Proceedings of the 16th International Symposium on Asynchronous Circuits and Systems*, 2010, pp. 165–175. [Online]. Available: http://dx.doi.org/10.1109/ASYNC.2010.25

[13] P. J. Antsaklis and A. N. Michel, *A Linear Systems Primer*, 1st ed. Birkhauser Basel, 2007. [Online]. Available: http://dx.doi.org/10.1109/MCS.2009.932913