Fast-Lock Hybrid PLL Combining Fractional-N and Integer-N Modes of Differing Bandwidths

Kyoungho Woo, *Student Member, IEEE*, Yong Liu, *Member, IEEE*, Eunsoo Nam, *Member, IEEE*, and Donhee Ham, *Member, IEEE*

***Abstract*—We introduce a single-loop PLL that operates in a narrower-bandwidth, integer-N mode during phase lock and in a wider-bandwidth, fractional-N mode during transient. This *hybrid* PLL, as a generalization of the conventional variable-bandwidth PLL that shifts only its bandwidth, simultaneously achieves the fast-locking advantage of the fractional-N PLL and design simplicity of the integer-N PLL, and as such, brings benefits in certain important PLL applications. In addition, the frequency division mode switching, unique in the hybrid PLL, enables a new, more digital protocol to execute bandwidth switching. A CMOS IC prototype attests to the validity of the proposed approach.**

***Index Terms*—Charge-pump phase-locked loops, fractional-N frequency synthesizers, integer-N frequency synthesizers, phase-locked loops.**

I. Introduction

As IN any dynamical system, bandwidths exert key influences on the dynamics of phase-locked loop (PLL) frequency synthesizers. Important characteristics of PLL frequency synthesizers, which are critically affected by bandwidths, include lock time and output spectrum.

Consider the design implications derived from the impact of bandwidths on the PLL dynamics, especially for the widely used charge-pump PLL frequency synthesizers. On one hand, a wider loop bandwidth directly translates to a faster locking, and hence, the bandwidth must be maximized to minimize lock time: the typical upper bound set to ensure loop stability is about 10% of the reference frequency f_{REF} [1], [2]. On the other hand, too wide a bandwidth brings more spurs and component noise (except VCO noise) into the PLL's output spectrum, while too narrow a bandwidth brings more VCO phase noise into the spectrum. It is somewhere in the middle that designers find the bandwidth yielding the spectrum optimal for a given design task. This bandwidth for the optimum spectrum is often found to be smaller than $0.1f_{\mathrm{REF}}$ [3]. Overall, maximizing the bandwidth for the fastest possible locking often betrays the need for a smaller bandwidth for the optimum spectrum.

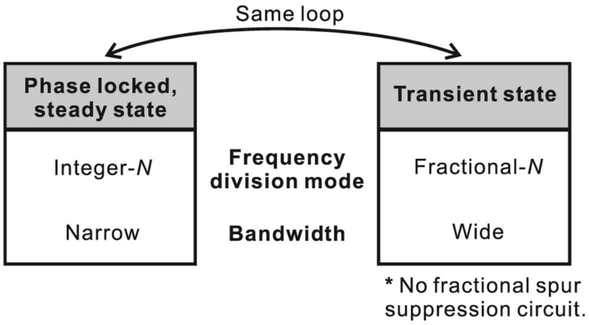

Fig. 1. Hybrid PLL operation.

To overcome this tradeoff, a variable-bandwidth scheme is frequently employed in PLL frequency synthesizers [4]. In this approach, a wider bandwidth is used during transient to accelerate phase locking, but once a PLL enters a phase-locked steady state, the bandwidth is shifted to a smaller value to attain optimum spectrum. This scheme exploits the fact that lock time matters only in transient while spectrum matters only in steady state. The efficacy of the scheme originating from Crowley's patent [4] has been proven via extensive PLL implementations, e.g., [3]–[9].

The variable-bandwidth PLL scheme, however, has so far been almost exclusively used within a fixed frequency division mode, i.e., the bandwidth switching has been executed while maintaining the same frequency division mode (integer- or fractional-N). In this paper, we generalize this conventional variable-bandwidth scheme, introducing a PLL that changes not only the bandwidth, but also the frequency division mode in transitions between transient and steady states. More concretely, the PLL reported here operates in an integer-N mode during phase lock (steady state), but in a fractional-N mode, which inherently has a wider loop bandwidth, during transient. See Fig. 1. The dual division modes and bandwidths are realized in a *single* loop. This *hybrid PLL* combining the two frequency division modes of differing bandwidths brings benefits in certain applications, as will be expounded upon shortly.

Section II describes the rationale behind our hybrid PLL. Section III presents its operation and architecture. Section IV contrasts our work with a couple of seemingly similar PLLs [10], [11]. Measurements of a CMOS IC affirming the validity of the proposed approach are found in Section V.

II. Rationale

Here we consider the rationale of our hybrid PLL, discussing its unique features and application spaces.

Manuscript received January 12, 2007; revised July 30, 2007. This work was supported by the National Science of Foundation (NSF) under Grant ECS-0313143 and Grant NSF-PHY-06-46094, and by the Electronics and Telecommunications Research Institute (ETRI).

K. Woo and D. Ham are with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138 USA (e-mail: khwoo@seas.harvard.edu; donhee@seas.harvard.edu).

Y. Liu was with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138 USA. He is now with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA.

E. Nam is with the Electronics and Telecommunications Research Institute (ETRI), Daejeon 305-706, Korea.

Digital Object Identifier 10.1109/JSSC.2007.914281

A. Feature 1—Simultaneous Achievement of Fast Locking and Design Simplicity

To elucidate this feature, let us first consider certain drawbacks of the conventional variable-bandwidth PLL whose frequency division mode is fixed at fractional- or integer-N [3]–[9].

For a given frequency resolution, the fractional-N PLL has a larger reference frequency than the integer-N PLL. Therefore, the transient-state bandwidth, limited to 10% of the reference frequency, is larger in the variable-bandwidth fractional-N PLL than in its integer-N counterpart. As a result, the former assumes a faster locking. This faster locking of the fractional-N PLL, however, comes at the price of increased design complexity. The fractional-N operation in steady state requires phase interpolators or high-order $\Sigma\Delta$ modulators to reduce fractional spurs [12], [13]. Since quantization noise of such fractional spur reduction circuits can fold into and corrupt the PLL spectrum via loop nonlinearities, more efforts are required to minimize loop nonlinearities [2], [14]–[16]. This design complexity is compounded by the fact that the negative impact of the quantization noise is hard to predict [17]. In contrast, the design of integer-N PLLs is much less complex due to the absence of fractional spurs.

Our hybrid PLL simultaneously achieves the fast-locking advantage of the fractional-N PLL and the design-simplicity benefit of the integer-N PLL. As shown in Fig. 1, the hybrid PLL operates in an integer-N mode during steady state, but operates in a fractional-N mode with a wider bandwidth (as discussed before, the fractional-N operation can accommodate a wider bandwidth) during transient. The fast locking of the hybrid PLL is the natural outcome of the wider bandwidth fractional-N operation during transient. The design simplicity of the hybrid PLL is attained because no fractional spur reduction circuit is needed in the transient fractional-N mode as spurs matter only in steady state. With no need for any fractional spur reduction circuit, the switching between the two frequency division modes is executed by a reconfiguration of only a couple of components in a *single* loop, and the overall architecture is almost a simple integer-N loop. (This will be explicitly shown in Section III.)

B. Application Spaces Enabled by Feature 1

In light of the foregoing discussion, we may view the hybrid PLL essentially as an integer-N PLL, which is made faster than the normal integer-N PLL by borrowing the speed of the fractional-N PLL during transient. Therefore, the hybrid PLL can be especially valuable when design simplicity is a major priority, and hence, an integer-N PLL is preferred, but at the same time, when the target frequency resolution is high (e.g., GSM, Bluetooth, and WLAN) so that normal integer-N PLLs with correspondingly small loop bandwidths have speed handicaps.

It is to be stressed that since integer-N PLLs in steady state have inherently worse phase noise than well-designed fractional-N PLLs in steady state, the hybrid PLL would not be an optimal design choice when phase noise is to be made as small as possible. However, in certain applications including the aforementioned GSM, Bluetooth, and WLAN, integer-N PLLs can still meet target phase noise specifications, e.g., [16] and [18]–[21]. In such applications, the hybrid PLL can serve as a valuable design choice for the reasons stated in the previous paragraph.

C. Feature 2—New Protocol for the Bandwidth Switching

Another distinctive feature of the hybrid PLL as compared to the conventional variable-bandwidth PLL is that the frequency division mode switching unique in the former enables a new protocol to execute the bandwidth switching. Here we describe the basic idea: the details are found in Section III-B.

In the conventional variable-bandwidth PLL, two building blocks are reconfigured to alter the loop bandwidth: the charge-pump (its current) and the loop filter (its component values) [22]. In our hybrid PLL, the frequency division ratio change naturally arising from the frequency division mode switching can serve as an additional parameter to change the bandwidth. Our approach thus allows for a new protocol for altering the loop bandwidth, using not only the conventional parameters (charge pump current and loop filter components), but also the frequency division ratio. This allows designers to explore a larger design space in terms of bandwidth switching. Depending on specific design goals, one can properly proportion the changes in the loop filter components, charge pump current, and the frequency division ratio to alter the bandwidth. For instance, when the bandwidth is to be changed by a large amount, this new protocol can lessen the burden of the large change in the charge-pump current, as the frequency division ratio change can also contribute to the bandwidth change.

One very interesting usage of this new bandwidth-switching protocol is to change the bandwidth with a fixed charge-pump current. This is impossible in the conventional bandwidth-switching, but is possible with the hybrid PLL because the frequency division ratio change can play the role of the charge-pump current change. This interesting case represents an execution of the bandwidth switching in a more digital fashion where the analog charge-pump is not reconfigured at all.

III. Operating Principles and Architecture

We now describe the operating principles of the hybrid PLL. We will first present the frequency division mode switching (Section III-A) and bandwidth switching (Section III-B) and then will combine them to form an overall architecture (Section III-C). Finally, we will examine the lock dynamics at around the switching moment (Section III-D).

A. Frequency Division Mode Switching

With Fig. 2, we will explain our basic scheme to switch between the fractional-N and integer-N mode in a single loop. Both modes have to produce the same identical set of output frequencies with the same frequency resolution. Fig. 2(a) shows the fractional-N mode operation of our hybrid PLL, which is used during transient. This is the standard fractional-N PLL, but with no fractional spur suppression circuits (phase interpolators or high-order $\Sigma\Delta$ modulators). As mentioned in Section II-A, the fractional spur suppression circuits are not needed in our hybrid PLL, as the fractional-N mode is used only during transient while spurs matter only in steady state. The reference frequency

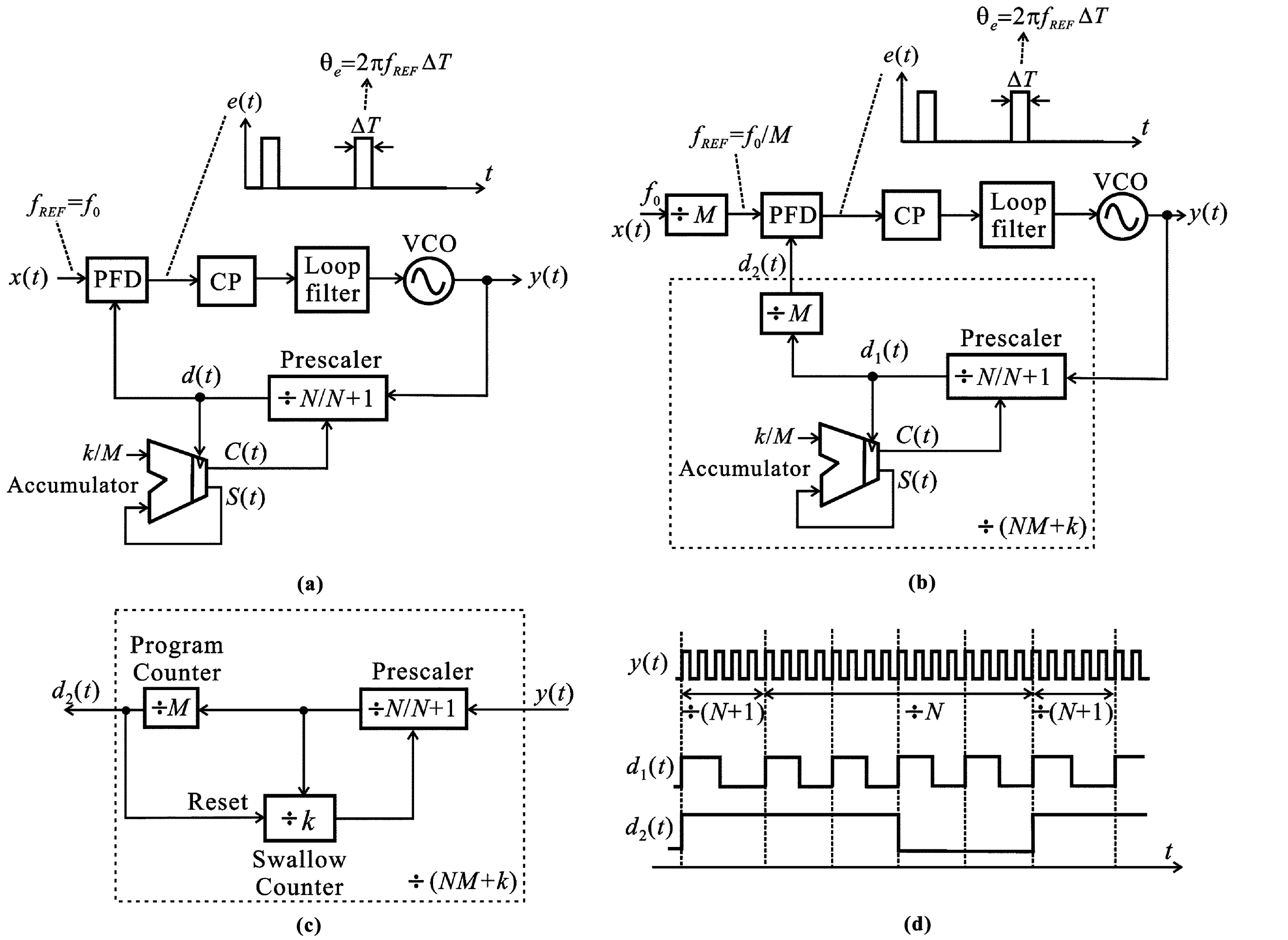

Fig. 2. (a) Fractional-N mode operation of the hybrid PLL. No fractional spur suppression circuit is used. (b) Integer-N mode operation of the hybrid PLL. The circuit within the dashed box provides the integer frequency division by $NM + k$. (c) Standard integer frequency division block [2]. (d) Timing diagram of the integer-N PLL of Fig. 2(b) ($N = 4$, $M = 5$, and $k = 1$).

f_{REF} in Fig. 2(a) is equal to the frequency f_0 of the crystal oscillator signal $x(t)$.[1] The standard accumulator–prescaler combination yields a fractional frequency division ratio of $N + k/M$. N and M are fixed integers, and $k = 0, 1, 2, \ldots, M - 1$ is a channel selection index. The set of synthesized frequencies is given by $(N + k/M)f_0$ as a function of k. The frequency resolution is f_0/M.

To enable our hybrid PLL approach, when the fractional-N loop of Fig. 2(a) reaches a phase lock, the loop should be modified to form an integer-N loop. This is done by a simple alteration that inserts a static divider (divide-by-M) in front of each of the two inputs of the phase-frequency detector (PFD), as shown in Fig. 2(b). This reconfigured loop of Fig. 2(b) is an integer-N PLL. The reference frequency is now $f_{\mathrm{REF}} = f_0/M$ and the frequency division ratio provided by the combination of the accumulator, static divider (divide-by-M), and prescaler within the dashed box of Fig. 2(b) is an integer number of $NM + k$. Therefore, the reconfigured loop produces the same set of output frequencies $(N + k/M)f_0$ with the same frequency resolution f_0/M as in the fractional-N mode of Fig. 2(a).

The $NM + k$ division block within the dashed box of the integer-N mode in Fig. 2(b) may look unfamiliar, as it is different from the conventional integer-N PLL's $NM + k$ division block consisting of a swallow counter, a static divider (program counter, divide-by-M), and a prescaler [see Fig. 2(c)] [2]. The difference arises from the use of an accumulator in our case, as opposed to the use of a swallow counter in the conventional case. To elucidate how the circuit within the dashed box in Fig. 2(b) indeed provides the integer frequency division by $NM + k$, steady-state voltage signals at the circuit's several nodes are shown in Fig. 2(d) with $N = 4$, $M = 5$, and $k = 1$ as an example: integer divide-by-21 $(NM + k = 21)$ is apparent.

Note from the foregoing discussions that the simple switching from the fractional-N to the integer-N loop is made possible by the absence of fractional spur reduction circuits in the fractional-N loop and by our subsequent recognition that the accumulator used for the simplified (no fractional spur reduction) fractional frequency division can be readily used for integer frequency division as well by replacing the standard swallow counter. This notion was also exploited in [11], but they used the division mode switching with no bandwidth switching. This contrast will be elaborated upon in Section IV.

Also note that the integer-N mode of Fig. 2(b) has just a bit more hardware complexity than the fractional-N mode of

[1] Throughout this paper, the reference frequency denoted by f_{REF} means the frequency of the signal at the *immediate input* of any PLL. The frequency of the crystal oscillator $x(t)$ is denoted as f_0. While $f_{\mathrm{REF}} = f_0$ in the fractional-N mode, it does not hold true in the integer-N mode, as seen shortly.

Fig. 2(a) (the fundamental reason for this is again the absence of any fractional spur suppression circuit in the fractional-N mode), and hence, one can expect that an overall hybrid PLL architecture incorporating both modes in a single loop would have the same level of hardware complexity as the normal integer-N PLL (the swallow counter and the accumulator have the same level of hardware complexity). We will revisit this point when we discuss the overall architecture in Section III-C.

In the hybrid PLL, the frequency division mode switching should be executed in synchronism with the bandwidth switching: the fractional-N mode used during transient can accommodate a wider bandwidth, and hence, should be given a wider bandwidth for fast locking. Section III-B describes our protocol to execute the bandwidth switching in conjunction with the division mode switching.

B. Bandwidth Switching

We start with the same bandwidth switching principle as in the conventional variable-bandwidth PLL [4]: changing the open loop bandwidth while preserving the same open loop phase margin to maintain the same degree of PLL stability. Despite the same fundamental principle, our bandwidth switching protocol will prove unconventional due to the division mode switching accompanying the bandwidth switching.

Consider a general charge-pump PLL frequency synthesizer with a second-order loop filter [see Fig. 3(a)]. The frequency division ratio N_d can be an integer or a fractional number. The open-loop transfer function of this synthesizer is

$$A_o(s) = \frac{\theta_{div}(s)}{\theta_{\mathrm{REF}}(s)} = \frac{K_{\mathrm{VCO}}}{2\pi} \cdot \frac{I_0}{N_d} \cdot \frac{F(s)}{s} \tag{1}$$

where K_{VCO} is the VCO gain, I_0 is the charge-pump current, and $F(s)$ is the input impedance of the loop filter given by

$$F(s) = \frac{1}{s(C_1 + C_2)} \cdot \frac{1 + sRC_2}{1 + sRC_{||}} \tag{2}$$

where $C_{||} \equiv C_1C_2/(C_1 + C_2)$. The magnitude and phase of $A_o(\omega)$ $(s = j\omega)$ are then given by

$$|A_o(\omega)| = \frac{K_{\mathrm{VCO}}}{2\pi(C_1 + C_2)} \cdot \frac{I_0}{N_d} \cdot \sqrt{\frac{1 + (\omega RC_2)^2}{1 + (\omega RC_{||})^2}} \cdot \frac{1}{\omega^2} \tag{3}$$

$$\phi(\omega) \approx \tan^{-1}(\omega RC_2) - \tan^{-1}(\omega RC_{||}) - 180^\circ. \tag{4}$$

The Bode plots of $|A_o(\omega)|$ and $\phi(\omega)$ are shown with solid lines in Fig. 3(b), where the unity gain frequency is denoted as ω_c and the phase margin at $\omega = \omega_c$ is signified by ϕ_M. In the Bode plots, $C_1 \ll C_2$ was assumed so that the pole $1/RC_{||} \sim 1/RC_1$ is clearly separated from the zero $1/RC_2$.

Now according to the bandwidth switching principle of [4], we seek to increase the bandwidth by a factor of α at the onset of a transient state while keeping the same phase margin of ϕ_M to preserve the same level of stability. This can be done by *simultaneously* executing the following two adjustments using three loop parameters, R (loop filter resistance), I_0, and N_d:

$$\begin{cases} \text{Adjustment A: } \underline{\text{Reduction of } R \text{ by a factor of } \alpha} \\ \text{Adjustment B: } \underline{\text{Increase of } (I_0/N_d) \text{ by a factor of } \alpha^2}. \end{cases}$$

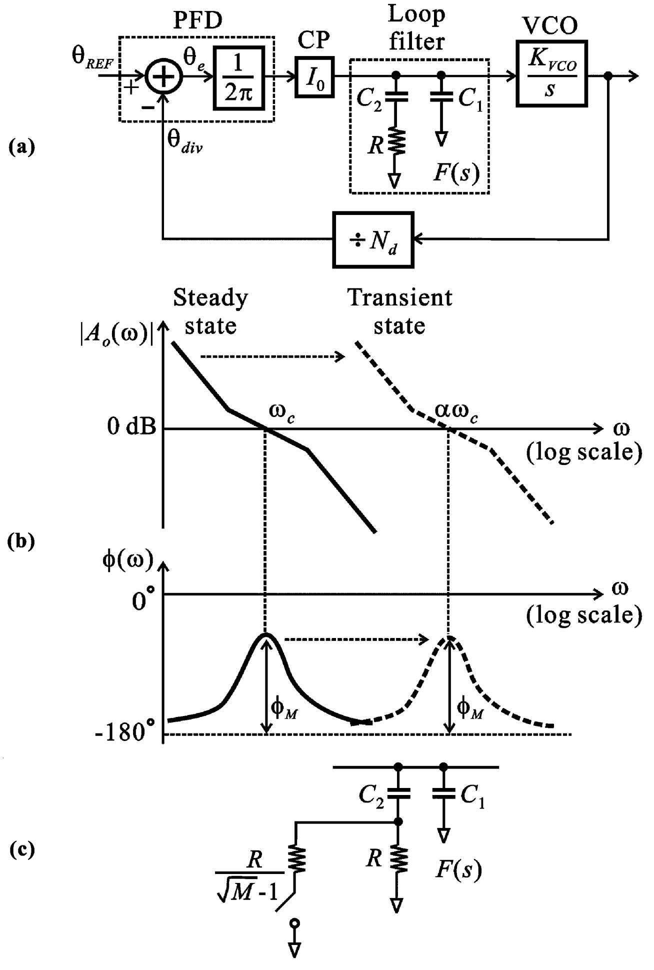

Fig. 3. (a) General charge-pump PLL model with a second-order loop filter. (b) Bode plots of the magnitude and phase of the open loop transfer function. The solid and dashed lines correspond to the smaller and larger bandwidth cases, respectively. (c) Second-order loop filter with a switch.

These adjustments replace ω in (3) and (4) with ω/α. This rescaling of ω corresponds to parallel translation of the Bode plots by $\log\alpha$ along the ω axis in the log scale, resulting in the dashed lines in Fig. 3(b). These parallel translations of the Bode plots produce the desired results: 1) the change of the unity gain frequency from ω_c to $\alpha\omega_c$ corresponds to the proportional bandwidth enhancement; 2) the phase margin at the new unity gain frequency is the same as that at the old unity gain frequency.

These parameter adjustments for bandwidth enhancement were derived quite generally, and are applicable not only to our hybrid PLL, but also to the conventional variable-bandwidth PLL with a fixed frequency division mode, as has been used numerous times [3]–[9]. In the latter, however, changing N_d is not an option, and parameters available for the adjustments are limited to I_0 and R.

In contrast, in our hybrid PLL, the increase in bandwidth at the onset of a transient is accompanied by the automatic reduction of N_d by a factor of M as the frequency division mode is also shifted from integer-N (division ratio: $NM + k$) to fractional-N (division ratio: $N + k/M$), and as a result, the hybrid PLL offers a new protocol for altering the bandwidth, which not only uses the conventional parameters (I_0 and R), but also exploits the automatic change in N_d as an additional parameter. In this protocol, N_d and I_0 are controlled *interdependently* for a given α since it is I_0/N_d that is used to execute *Adjustment B*. Below are two interesting examples of how the parameter adjustments for bandwidth switching above can be applied to our hybrid PLL.

- *Example 1*: This is an example where I_0 is kept constant and only the automatic reduction in N_d is used to execute *Adjustment B*. This is a polar opposite of the bandwidth switching in the conventional variable-bandwidth PLL where I_0 must be increased to execute Adjustment B as N_d is fixed. In this example, since I_0 is constant and N_d is automatically reduced by a factor of M, *Adjustment B* is performed with $\alpha = \sqrt{M}$. *Adjustment A* then can be done by decreasing R by a factor of $\sqrt{M}$ by designing the loop filter of Fig. 3(a) with a switch as shown in Fig. 3(c), and closing the switch when the synthesizer is switched to the fractional-N mode at the onset of a transient. With I_0 held constant and entirely relying on N_d for *Adjustment B*, this example represents a bandwidth switching protocol more digital than any other parameter adjustment possibilities.
- *Example 2*: The example above is a semidigital bandwidth switching keeping the same I_0. On the contrary, by not only exploiting the automatic reduction in N_d, but also altering I_0, one can attain the highest possible bandwidth during transient, as shown in this example. Here the transient bandwidth of the fractional-N mode is maximized to 10% of the reference frequency of the fractional-N mode, i.e., it is given by $f_{\mathrm{REF}}/10 = f_0/10$ where f_0 is the crystal oscillator frequency [Fig. 2(a)]. For the steady-state bandwidth of the integer-N mode to achieve the optimum spectrum, although it varies from design to design, let us choose 5% of the reference frequency of the integer-N mode, i.e., $f_{\mathrm{REF}}/20 = f_0/(20M)$, as a rough estimate for the sake of argument [23]. With these choices of the transient and steady-state bandwidth, we have[2] $\alpha = 2M$. Therefore, for *Adjustment A*, R should be decreased by a factor of $2M$, and for *Adjustment B*, I_0 should be increased by a factor of $4M$ since N_d is automatically reduced by a factor of M. Note that in the conventional variable-bandwidth PLL where N_d is fixed, increasing the bandwidth by the same factor of $\alpha = 2M$ requires the increase of I_0 by a much larger factor of $4M^2$.

The above two examples clearly demonstrate the increased degrees of flexibility in the bandwidth switching that are made possible by the hybrid PLL operation.

C. Overall Architecture

By combining the fractional-N mode of Fig. 2(a) and the integer-N mode of Fig. 2(b) in a single loop, and simultaneously incorporating the bandwidth switching, we form the hybrid PLL frequency synthesizer of Fig. 4. This architecture uses the semidigital bandwidth switching with a constant charge-pump current (Example 1 in Section III-B).

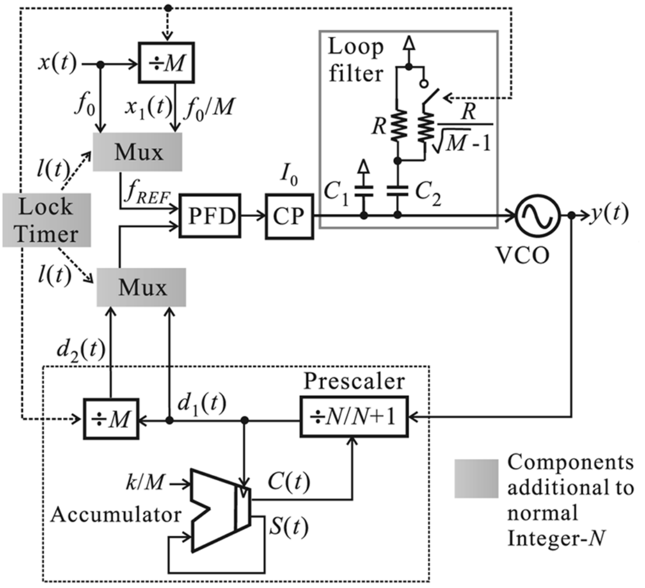

Fig. 4. Overall hybrid PLL architecture.

When the synthesizer of Fig. 4 enters a transient state, the two divide-by-M blocks are disabled and screened out by the two multiplexers shown inside shaded areas, and the crystal oscillator signal $x(t)$ and the prescaler output $d_1(t)$ are directly fed to the two inputs of the PFD. This operation mode is equivalent to the fractional-N loop of Fig. 2(a). When the synthesizer attains a phase lock (which is signaled by the lock timer shown inside a shaded area in Fig. 4), the two divide-by-M blocks are enabled and the two multiplexers are now made to choose to use the outputs of the two divide-by-M blocks as inputs of the PFD. This is equivalent to the integer-N loop of Fig. 2(b). The switch in the loop filter is open in the integer-N mode, and is closed in the fractional-N mode. Due to the constant charge-pump current, there is no switch associated with the charge pump.

In Section III-A, we argued by looking at Fig. 2(a) and (b) that the overall architecture would be almost the same as a normal integer-N PLL. Now the explicit architecture of the hybrid PLL in Fig. 4 permits us to exactly compare it to the architecture of normal integer-N PLLs. To begin with, the circuit within the dashed box of Fig. 4 consisting of the accumulator, divide-by-M, and prescaler, which performs the integer divide-by-$(NM + k)$ through the output $d_2(t)$, is similar to the more standard integer divide-by-$(NM + k)$ circuit [2] consisting of a pulse swallow counter, divide-by-M, and prescaler shown in Fig. 2(c). The only difference comes from the accumulator and the swallow counter, but they have the same level of hardware complexity. Second, the divide-by-M block in the upper portion of Fig. 4 is the same as the reference divider in standard integer-N PLLs, which is commonly used in high frequency resolution applications. Therefore, the hybrid PLL of Fig. 4 is architecturally similar to the conventional integer-N PLL, having only three additional components (the two multiplexers and the lock timer), which all consist of simple digital logic gates and programmable counters. This similarity is due fundamentally to the absence of any fractional spur suppression circuit in our hybrid PLL.

[2] This bandwidth switching factor of $2M$ is larger than the bandwidth switching factor $\sqrt{M}$ of Example 1. This is attributed to not holding I_0 at a constant value in this example.

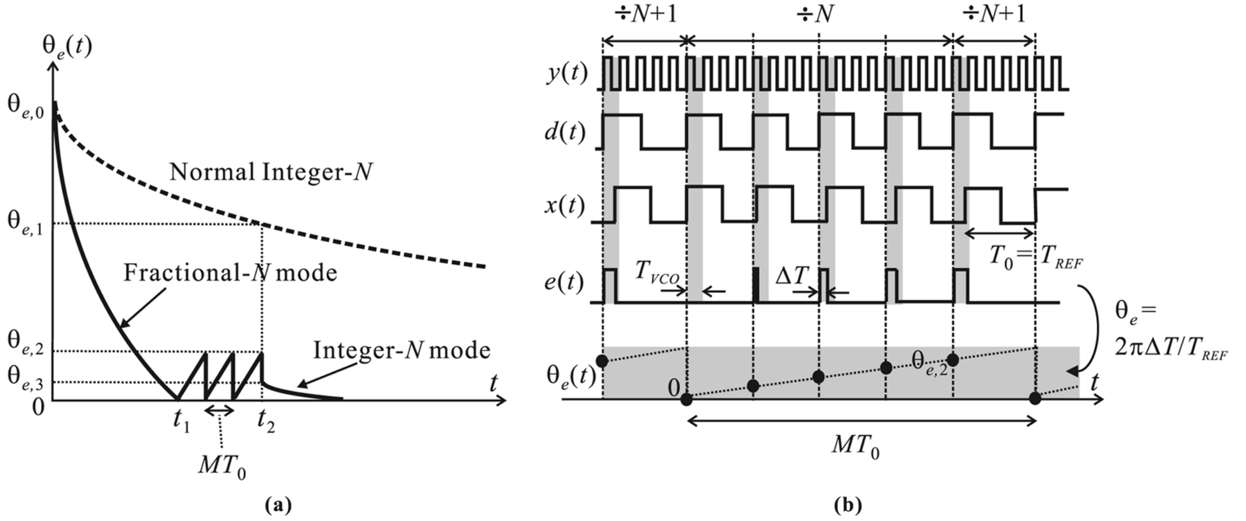

Fig. 5. (a) Hypothetical evolution of the phase error θ_e with time for the hybrid PLL (solid curve) and the normal integer-N PLL (dashed curve). (b) Voltage signals in the steady-state fractional-N mode operation [see Fig. 2(a)] for an example with $N = 4$, $k = 1$, and $M = 5$.

The lock timer can be realized in a variety of ways [9], [11], [22]. We will present our lock timer operation in Section V along with the CMOS IC measurement.

D. Locking and Switching Dynamics

As we have completed the description of the operating principle of the hybrid PLL, let us now examine the evolution of the phase error with time at the output of the PFD with special attention to its behavior at around switching moments.

The hybrid PLL is essentially an integer-N PLL, which is made faster by borrowing the speed of the fractional-N PLL during the transient. Fig. 5(a) illustrates this notion by comparing the hypothetical lock behavior of the hybrid PLL (solid curve) to that of the normal integer-N PLL (dashed curve). The loop bandwidth of the normal integer-N PLL is equal to the loop bandwidth of the hybrid PLL at its steady-state integer-N mode operation. In this figure, the vertical axis is the phase error θ_e obtained from the PFD output $e(t)$ (θ_e and $e(t)$ are with reference to Fig. 2), and the horizontal axis is the time, t. When the two synthesizers start from the same phase error $\theta_{e,0}$, the phase error of the hybrid PLL reduces towards zero faster than that of the normal integer-N PLL because of the wider loop bandwidth of the former during its transient fractional-N mode operation.

An important premise for our fast locking scheme via the hybrid approach is that at the moment when the fractional-N mode is switched to the integer-N mode at the onset of a phase lock, the phase error should not jump to a large value. More specifically in reference to Fig. 5(a), when the hybrid PLL switches from the fractional-N mode to the integer-N mode (say, at $t = t_2$) after it acquires a phase lock, the new phase error $\theta_{e,3}$ of the integer-N mode should not pop back up to exceed the phase error $\theta_{e,1}$ of the normal integer-N PLL at the same time of $t = t_2$. If this happened, the fast locking purpose of the hybrid PLL would be defeated.

We now argue that such a large phase error disturbance indeed does not occur. Since the fractional-N mode [see Fig. 2(a)] does not incorporate any fractional suppression circuit, once it enters a steady state with a phase lock at $t = t_1$, the phase error θ_e is not settled to zero, but it exhibits a small oscillation between 0 and a certain maximum value $\theta_{e,2}$, as shown with the zigzag pattern in Fig. 5(a), which corresponds to fractional spurs in the frequency domain. A bit more detail of this phase error oscillation is illustrated in Fig. 5(b) along with other relevant signals for an example with $N = 4$, $M = 5$, and $k = 1$.[3] The phase error $\theta_e = 2\pi\Delta T/T_{\text{REF}}$ of the error signal $e(t)$, which appears once in every period of the reference signal $x(t)$, continually grows up to $\theta_{e,2}$ given by

$$\begin{aligned}\theta_{e,2} &\sim 2\pi\cdot\frac{(\Delta T)_{\max}}{T_{\text{REF}}}\\ &= 2\pi\cdot\frac{M-1}{M}T_{\text{VCO}}\cdot\frac{1}{T_{\text{REF}}}\\ &\sim 2\pi\cdot\frac{M}{NM+k}\end{aligned}\tag{5}$$

until the prescaler is reset to return the phase error to zero. This growth dynamics repeats itself with the period of $M/f_{\text{REF}} = M/f_0$, corresponding to the zigzag pattern in Fig. 5(a).

Now when the fractional-N loop is reconfigured to form the integer-N mode at $t = t_2$ in Fig. 5(a) (this is the worst case scenario as the switching occurs when the phase error assumes the maximum value $\theta_{e,2}$), the new phase error $\theta_{e,3}$ will be given by

$$\theta_{e,3} = \frac{1}{M}\cdot\theta_{e,2}\tag{6}$$

because the reference frequency f_{REF} is decreased by a factor of M, while ΔT remains almost the same right after the switching. This means that there will not be a large phase error disturbance at all, but rather $\theta_{e,3}$ will be always smaller than $\theta_{e,2}$. In addition, since $\theta_{e,2} < \theta_{e,1}$ can be safely assumed because the

[3] The waveforms are based on the simplified analysis assuming that the loop is disconnected between the charge-pump and the loop filter after the fractional-N PLL reaches a phase lock [2]. While this is not the most rigorous analysis (in a rigorous picture, the phase error would oscillate not between 0 and a positive maximum, but between a negative minimum and a positive maximum), it captures the essence of the steady-state behavior of the fractional-N PLL.

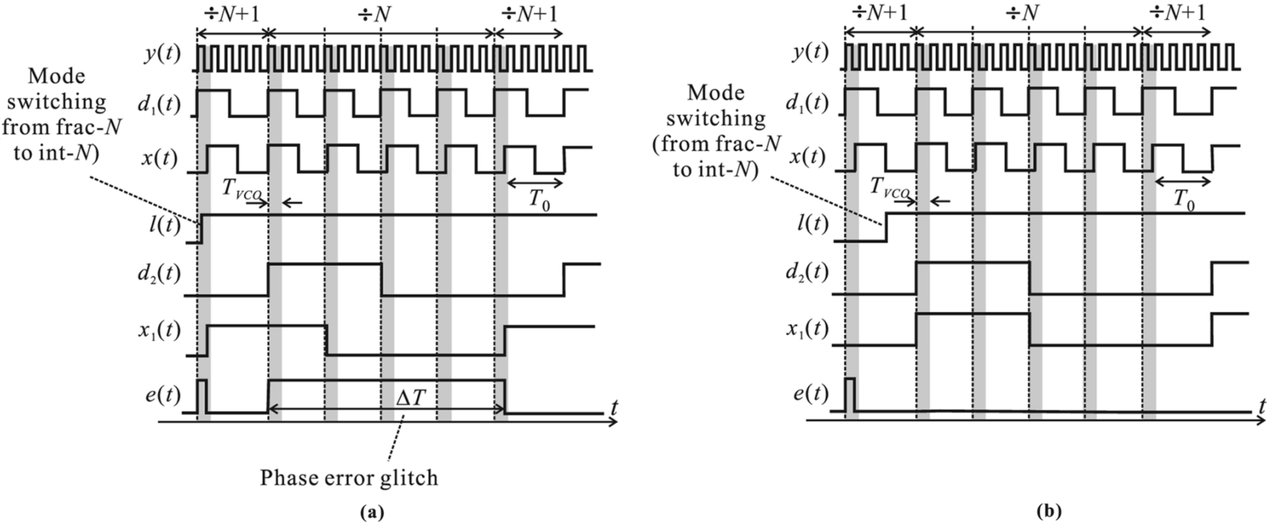

Fig. 6. Waveforms from various nodes in the hybrid PLL of Fig. 4 around switching moments. (a) If a mode switching is executed during a phase comparison [as seen, the mode switching signaled by the rising edge of the lock-timer signal $l(t)$ takes place after the rising edge of $d_1(t)$, but before the rising edge of $x(t)$], a large phase error disturbance (glitch) shown by the large ΔT can occur. (b) This glitch problem can be avoided by synchronizing the mode switching time with the falling edge of either $x(t)$ or $d_1(t)$.

fractional-N mode is faster than the normal integer-N PLL, we conclude that $\theta_{e,3} < \theta_{e,1}$ and the hybrid PLL will settle always faster than the normal integer-N PLL, as shown in Fig. 5(a). Moreover, by combining (5) and (6), we obtain

$$\theta_{e,3} = 2\pi \cdot \frac{1}{NM + k} \tag{7}$$

which means that $\theta_{e,3}$ will be not only smaller than $\theta_{e,2}$, but also significantly smaller than 2π for a practical choice of values for N, M, and k.

The foregoing discussion shows that the shift from fractional-N to integer-N makes the phase error undergo a significant reduction instead of a disturbance (glitch), and hence, in principle, the division mode switching does not compromise, but rather enhances the power of the hybrid PLL approach.

In practice, the mode switching moment should be carefully chosen to ensure that ΔT right before the mode switching remains more or less the same in the next phase comparison event immediately following the mode switching, which was the basic assumption to obtain (6). To see this clearly, let us provide an example where the mode switching time is ill chosen. If the mode switching takes place during the phase comparison, as in Fig. 6(a) (all the signal notations in this figure are with reference to Fig. 4), i.e., if the lock timer output $l(t)$ signals a mode switching after the rising edge of $d_1(t)$, but before the rising edge of $x(t)$, the PFD will miss the input rising edge of $x(t)$. As a result, the outputs of the two static dividers in Fig. 4 are misaligned, generating a large ΔT in the phase comparison period following the mode switching [see Fig. 6(a)]. This phase error glitch can be avoided by synchronizing the mode switching time (the rising edge of $l(t)$) with the falling edge of either $x(t)$ or $d_1(t)$, as shown in Fig. 6(b). With this edge alignment, ΔT will remain the same right after the switching, justifying (6) (In Fig. 6(b), we simplified the picture by assuming a lock acquisition right after the switching into the integer-N mode, and by making $\Delta T = 0$).

A parasitic capacitance bypassing the physical switch in the loop filter of Fig. 4 can also give rise to a phase error glitch [22], but a careful physical layout to minimize the parasitic capacitance easily solves the problem. As we will see in Section V, our implemented PLL does not exhibit any discernable phase error perturbation at the switching moment.

IV. Comparison to Prior Works

We now compare our work with two seemingly similar PLLs reported in [10] and [11].

In the PLL of [10], while its frequency division mode is changed from fractional-N to what is similar to integer-N at the onset of a steady state, the goal is not fast locking, but fractional spur removal, and no explicit attempt for bandwidth switching is made. In addition, although their steady-state loop yields what a standard integer-N PLL would yield in terms of output signals, the inner workings of the former are different from the latter. Overall, the PLL architecture of [10] is different from our hybrid PLL architecture.

The PLL of [11] employs the frequency division mode switching in a single loop, just as in our case, and their motivation is clearly fast locking. However, they deliberately maintained the same bandwidth between the two frequency division modes, hence not exploiting the speed advantage of the hybrid approach.[4] To preserve the same bandwidth, they used the fixed loop filter (hence, not performing *Adjustment A*, discussed in Section III-B) and at the same time they incorporated a charge pump switch in order to offset the automatic change of N_d occurring in the mode switching with the charge-pump current change (hence, not performing *Adjustment B*, discussed in Section III-B). Therefore, the PLL of [11] is topologically and operationally in contrast with our hybrid PLL, which uses

[4] For fast locking, they seem to rely on the fact that the fractional-N mode has more phase comparison events per unit time than the integer-N mode, which is naturally exploited in our design as well. However, this alone is not as powerful as the bandwidth increase in terms of enhancing locking speed.

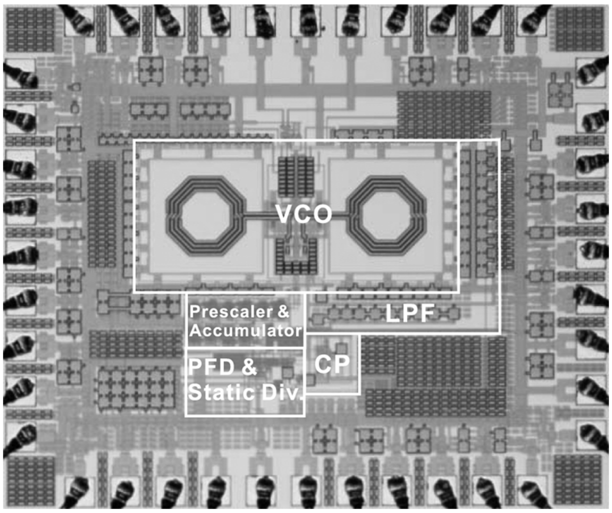

Fig. 7. Die micrograph. The chip area is 1.6 × 1.3 mm^2.

a fixed charge pump current, but incorporates a switch-bearing loop filter. Overall, the PLL of [11] may be viewed as a valuable intermediate step in the extensional development of our hybrid PLL from the conventional variable-bandwidth PLL [4].

V. Experiments

A. CMOS IC Prototype

A proof-of-concept hybrid PLL prototype was designed and fabricated in TSMC 0.18-μm mixed-signal CMOS technology. Fig. 7 shows a die micrograph for the CMOS IC. The experimental results with this IC were briefly presented in [24]. The IC was packaged inside a 10-mm LQFP 44-pin package, which is mounted on a printed circuit board (PCB) that contains auxiliary electronics. A 1.8-V power supply was used.

The CMOS IC realizes the architecture of Fig. 4 where the frequency division mode switching takes place in synchronism with the semidigital bandwidth switching (constant charge-pump current). Most of the components of Fig. 4 were implemented on chip. The only off-chip (on-PCB) components are the crystal oscillator, most of the lock timer, and most of the loop filter. The component values of the implemented loop filter of Fig. 4 are $C_1 = 150$ pF, $C_2 = 2$ nF, and $R = 11.2$ kΩ. The lock timer operation will be explained shortly.

TABLE I
Hybrid-PLL Frequency and Bandwidth Setup, and Settling-Time Measurement Results

Frequency synthesis plan (hybrid PLL)	
f_{REF} (Tracking, frac-N)	64 MHz
f_{REF} (Locked, int-N)	1 MHz
N_1, M, k	37, 64, $\{0, 1, ..., 63\}$
N_2, M, k	38, 64, $\{0, 1, ..., 63\}$
N_3, M, k	39, 64, $\{0\}$
Number of channels	129
Channel spacing	1 MHz
Output frequencies	2.368 $\sim$ 2.496 GHz
Bandwidth setup (hybrid PLL)	
Loop bandwidth (Tracking)	400 kHz
Loop bandwidth (Locked)	50 kHz
α (BW enhancement factor)	$\sqrt{M} = 8$
Settling-time measurement results	
Hybrid-PLL	$\sim$ 20 μs
Normal Integer-N PLL	$\sim$ 80 μs

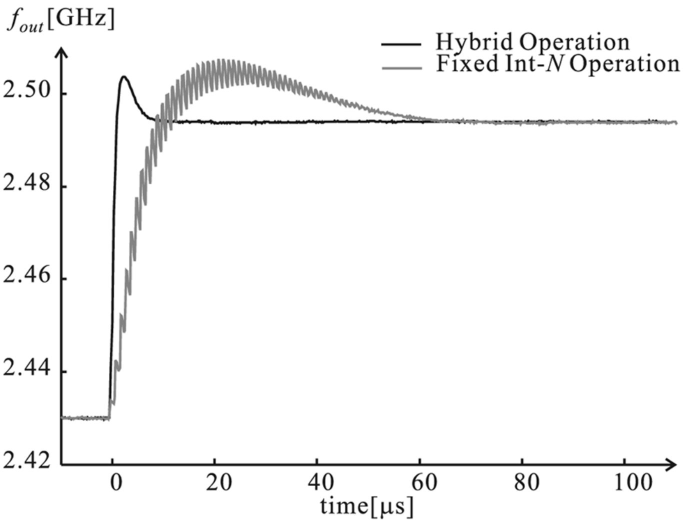

Fig. 8. Measured frequency settling transients for the hybrid PLL (black line) and the normal integer-N PLL (gray line).

The target frequency synthesis plan for the CMOS IC is summarized at the top portion of Table I. A 64-MHz crystal oscillator is used ($f_0 = 64$ MHz) and $M = 64$, and hence, the maximum number for the channel selection index k is 63. To create 129 channels, the prescaler is designed in such a way that three N values were created. The CMOS IC successfully synthesized this target set of output frequencies. Since $M = 64$ and we sought to use the semidigital bandwidth-switching protocol with a constant charge-pump current (Example 1 in Section III-B) in this implementation, the bandwidth enhancement factor α is $\sqrt{M} = 8$, and we accordingly set up the transient and steady-state loop bandwidths, as shown in the middle portion of Table I. The steady-state bandwidth of 50 kHz in the integer-N mode was set at 5% of the reference frequency of the integer-N mode (1 MHz), which is the proportion often used as an initial choice of the integer-N mode bandwidth [23]. As the main goal here lies in verifying the concept of the hybrid PLL, especially its faster locking performance in comparison to the normal integer-N PLL, we have not placed a significant effort in detailed optimization of the steady-state bandwidth of the integer-N mode for optimum output spectrum.

B. Frequency Settling Measurements

We measured the locking transient of the hybrid PLL by exciting the system with a 64-MHz frequency step (from $N = 37$, $k = 62$ (Channel-a) to $N = 38$, $k = 62$ (Channel-b), i.e., from 2.430 to 2.494 GHz) at $t = 0$. The black line of Fig. 8 shows the frequency settling transient of the hybrid PLL measured using an Agilent E5052A signal source analyzer. The division mode and bandwidth of the hybrid PLL were simultaneously switched from the fractional-N mode (400-kHz bandwidth) to the integer-N mode (50-kHz bandwidth) at $t = 10$ μs.

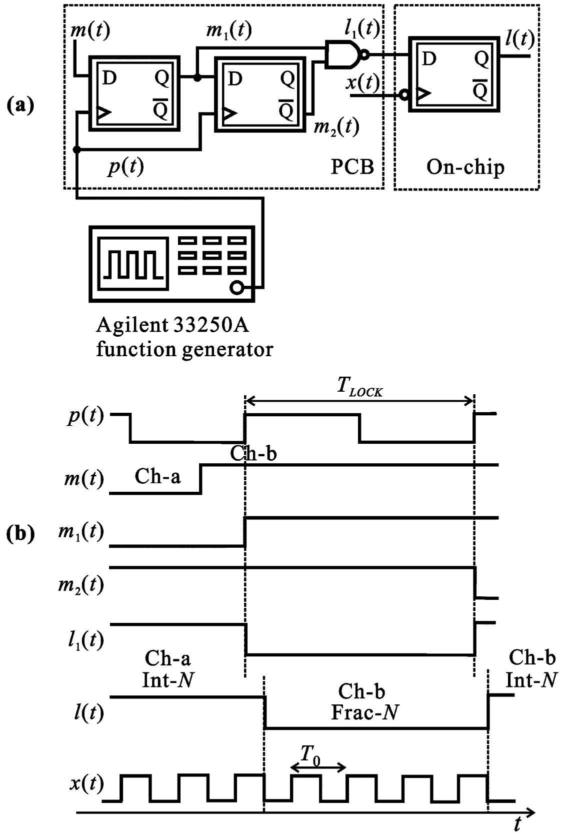

Fig. 9. (a) Schematic of our lock timer. (b) Timing diagram for various nodes in the lock timer.

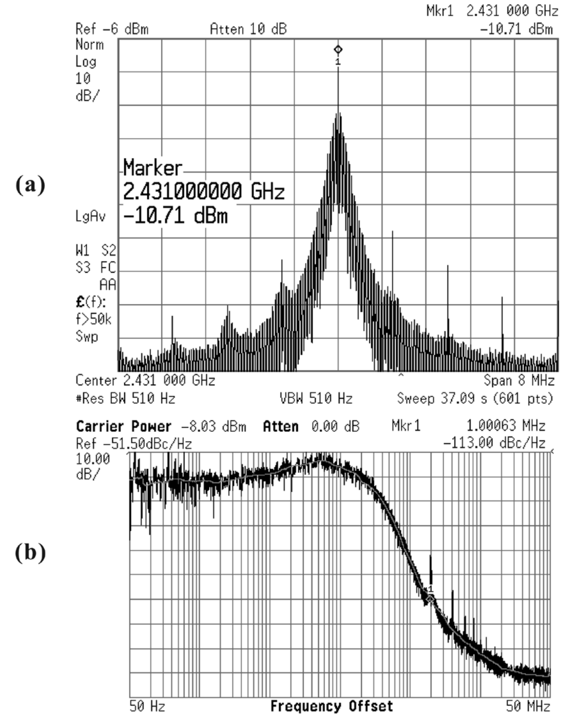

Fig. 10. (a) Measured power spectral density of the hybrid PLL in its steady-state integer-N mode. The output frequency is 2.431 GHz with $N = 37$ and $k = 63$. (b) Corresponding, measured phase noise of the hybrid PLL.

This switching time, which we call T_{LOCK}, was manually determined by first running the hybrid PLL in its fractional-N mode without a switching operation, and estimating its settling time. As seen via the black line in Fig. 8, no discernable phase disturbance is observed at the switching moment at $t = T_{\mathrm{LOCK}} = 10\ \mu$s, and the overall frequency settling time for the hybrid PLL is about 20 μs.

For comparison purposes, with the same frequency step excitation, we also measured locking transient of a normal integer-N PLL, which is obtained from the hybrid PLL by constantly operating it in its integer-N mode without employing the mode and bandwidth switching. The gray line of Fig. 8 shows the measured locking transient for the normal integer-N PLL. The frequency settling time is approximately 80 μs, which is four times larger than that of the hybrid PLL. This comparative characterization of the two PLL operations affirms the validity of our fast locking approach using the hybrid PLL.

C. Lock Timer Operation

In the experiment above, it is the lock timer that orders the PLL to switch from Channel-a to Channel-b at $t = 0$, and also orders it to switch from the fractional-N mode to the integer-N mode in Channel-b at $t = T_{\mathrm{LOCK}}$, where T_{LOCK} is the predetermined time. Our lock timer depicted in Fig. 9(a) executes these functions in the following fashion. The first two flip-flops are clocked by the clock signal $p(t)$ produced by the Agilent 33250A function generator. The period of $p(t)$ is tuned to T_{LOCK}. The step pulse $m(t)$ that signals the channel change [from Channel-a to Channel-b: see Fig. 9(b)] is fed to the first flip-flop, and gets subsequently processed by the second flip-flop and the NAND gate. Overall, the output $l_1(t)$ of the NAND gate goes from 1 to 0 and then back to 1, as shown in Fig. 9(b), where its 0-duration is T_{LOCK}. Now the third flip-flop is clocked by the crystal oscillator signal $x(t)$ and its input is $l_1(t)$. The resulting output $l(t)$ of the third flip-flop is a delayed replica of $l_1(t)$,[5] and the 0-to-1 rising edge of $l(t)$ is now aligned with the falling edge of $x(t)$.

$l(t)$ is the output of the lock timer, which signals the switching moments of the hybrid PLL. This is the same $l(t)$ of Figs. 4 and 6. When $l(t)$ goes first from 1 to 0, it leads the hybrid PLL from the steady-state integer-N mode in Channel-a to the transient fractional-N mode in Channel-b. When $l(t)$ goes from 0 back to 1 after T_{LOCK}, the PLL switches from the transient fractional-N mode in Channel-b to the steady-state integer-N mode in the same channel. The rising transition of $l(t)$ from 0 to 1 is what was shown with $l(t)$ in Fig. 6. As we aligned the falling edge of $x(t)$ and the 0-to-1 rising edge of $l(t)$ using the third flip-flop, phase error glitch does not take place at the mode switching, as explained in Section III-D.

D. Other Measurements

The main goal of this work to achieve fast locking using the hybrid PLL has been sufficiently demonstrated from the measurements above. Now for the sake of completeness, we briefly summarize other measured aspects of the hybrid PLL. Fig. 10(a) is a power spectral density of the hybrid PLL (at its steady-state integer-N mode) measured using an Agilent E4448A spectrum

[5] Since T_{LOCK} is not an exact integer multiple of the period $T_0 = 1/f_0$ of $x(t)$ in general, $l_1(t)$ and $l(t)$ cannot be exactly the same. But since $T_{\mathrm{LOCK}} \gg T_0$ in most cases, $l_1(t)$ and $l(t)$ may be regarded as practically the same.

TABLE II
OTHER MEASUREMENT RESULTS

Measured spectral data (@ 2.409 GHz)	
Reference spur @ 1MHz	-54 dBc
Phase noise @ 1MHz	-113.0 dBc/Hz
Power dissipation	
VCO	6.3 mW
VCO buffer	13.8 mW
Prescaler	8.6 mW
Charge-pump	0.4 mW
Total	29.6 mW

analyzer when $N = 37$ and $k = 63$ (2.431 GHz of output frequency). This measurement shows the desired frequency synthesis with the 1-MHz frequency resolution as planned. The fundamental reference spur of -54 dBc appears at 1 MHz offset.

Fig. 10(b) is a corresponding phase noise plot at the same output frequency of 2.431 GHz, which was measured using the built-in phase noise measurement capability of the Agilent E4448A spectrum analyzer. As mentioned earlier, the steady-state bandwidth of the integer-N mode was chosen at 5% (50 kHz) of its reference frequency without bandwidth optimization for optimum spectrum, as it was not the main goal of this work, and therefore, the rather large input referred noise was not fully suppressed by the steady-state integer-N mode, and consequently, the phase noise plot exhibits the -40-dB/dec slope of the second-order loop filter around the frequency range from 100 kHz to 1 MHz. At higher frequencies, the phase noise plot follows the -20-dB/dec slope, which corresponds to the high-pass filtered phase noise of the VCO. The phase noise can be further optimized by properly choosing the steady-state bandwidth of the integer-N mode and by reducing the component noise, without afflicting the hybrid PLL operation as the latter is independent of the phase noise optimization. Table II summarizes the measured spectral data, as well as the component-by-component breakdown of the power consumption.

VI. CONCLUSION

The conventional variable-bandwidth PLL frequency synthesizer, introduced by Crowley in [4], has gained popularity among PLL designers due to its fast locking virtue, but it has been almost exclusively operated with a fixed frequency division mode. This paper reported on an extension from this conventional technique, introducing a PLL that changes not only its bandwidth, but also its frequency division mode in transitions between transient and steady states. Achieving fast locking and design simplicity simultaneously and allowing for a more digital bandwidth control, this is an integer-N PLL that is made faster during transient by borrowing the speed of the fractional-N PLL, and could be a useful addition to the current state of PLL design.

ACKNOWLEDGMENT

The authors would like to thank Dr. H. C. Kim, ETRI, for his support for the fabrication of the CMOS IC. The authors would also like to thank W. Andress, N. Sun, Prof. P. Horowitz, and Prof. G. Wei, all with Harvard University, Prof. M. Perrott, Prof. H. S. Lee, and C. M. Hsu, all with the Massachusetts Institute of Technology, Prof. D. Ricketts, Carnegie-Mellon University, and Prof. W. Rhee, Tsinghua University, for their suggestions and help with measurements. The authors also acknowledge the donation of their electromagnetic field solvers by Sonnet Software and the Ansoft Corporation.

REFERENCES

[1] F. M. Gardner, "Charge-pump phase-lock loops," *IEEE Trans. Commun.*, vol. COM-28, no. 11, pp. 1849–1858, Nov. 1980.

[2] B. Razavi, *RF Microelectronics*. Upper Saddle River, NJ: Prentice-Hall, 1998.

[3] D. Byrd, C. Davis, and W. O. Keese, "A fast locking scheme for PLL frequency synthesizers," National Semiconductor, Santa Clara, CA, Applicat. Note 1000, 1995.

[4] Crowley, "Phase locked loop with variable gain and bandwidth," U.S. Patent 4,156,855, May 29, 1979.

[5] C. Vaucher, "An adaptive PLL tuning system architecture combining high spectral purity and fast settling time," *IEEE J. Solid-State Circuits*, vol. 35, no. 4, pp. 490–502, Apr. 2000.

[6] J. Lee and B. Kim, "A low-noise fast-lock phase-locked loop with adaptive bandwidth control," *IEEE J. Solid-State Circuits*, vol. 35, no. 8, pp. 1137–1145, Aug. 2000.

[7] C.-Y. Yang and S.-I. Liu, "Fast-switching frequency synthesizer with a discriminator-aided phase detector," *IEEE J. Solid-State Circuits*, vol. 35, no. 10, pp. 1445–1452, Oct. 2000.

[8] S. Cho and A. P. Chandrakasan, "A 6.5-GHz energy-efficient BFSK modulator for wireless sensor applications," *IEEE J. Solid-State Circuits*, vol. 39, no. 5, pp. 731–739, May 2004.

[9] M. Keaveney, P. Walsh, M. Tuthill, C. Lyden, and B. Hunt, "A 10 μs fast switching PLL synthesizer for a GSM/EDGE base-station," in *IEEE ISSCC Dig. Tech. Papers*, 2004, pp. 192–193.

[10] S. Pellerano, S. Levantino, C. Samori, and A. L. Lacaita, "A dual-band frequency synthesizer for 802.11a/b/g with fractional-spur averaging technique," in *IEEE ISSCC Dig. Tech. Papers*, 2005, pp. 104–105.

[11] B. Memmler, E. Gotz, and G. Schonleber, "New fast-lock PLL for mobile GSM GPRS applications," in *Proc. 26th ESSCIRC*, Sep. 2000, pp. 468–471.

[12] N. G. Kingsbury, "Frequency synthesizer with fractional division ratio and jitter compensation," U.S. Patent 4,179,670, Dec. 18, 1979.

[13] T. A. D. Riley, M. A. Copeland, and T. A. Kwasniewsky, "Sigma-delta modulation in fractional-N synthesis," *IEEE J. Solid-State Circuits*, vol. 28, no. 5, pp. 553–559, May 1993.

[14] T. A. D. Riley, N. M. Filiol, Q. Du, and J. Kostamovaara, "Techniques for in-band phase noise reduction in $\Sigma\Delta$ synthesizers," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 50, no. 11, pp. 794–803, Nov. 2003.

[15] B. D. Muer and M. S. J. Steyaert, "On the analysis of $\Sigma\Delta$ fractional-N frequency synthesizers for high-spectral purity," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 50, no. 11, pp. 784–793, Nov. 2003.

[16] E. Duvivier, G. Puccio, S. Cipriani, L. Carpineto, P. Cusinato, B. Bisanti, F. Galant, F. Chalet, F. Coppola, S. Cercelaru, N. Vallespin, J.-C. Jiguet, and G. Sirna, "A fully integrated zero-IF transceiver for GSM-GPRS quad-band application," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2249–2257, Dec. 2003.

[17] M. Tiebout, C. Sandner, H. Wohlmuth, N. D. Dalt, and E. Thaller, "A fully integrated 13 GHz Σ Δ fractional-N PLL in 0.13 μm CMOS," in *IEEE ISSCC Dig. Tech. Papers*, 2004, pp. 386–387.

[18] A. Ajjikuttira, C. Leung, E.-S. Khoo, M. Choke, R. Singh, T.-H. Teo, B.-C. Cheong, J.-H. See, H.-S. Yap, P.-B. Leong, C.-T. Law, M. Itoh, A. Yoshida, Y. Yoshida, A. Tamura, and H. Nakamura, "A fully integrated CMOS RFIC for Bluetooth applications," in *IEEE ISSCC Dig. Tech. Papers*, 2001, pp. 198–199.

[19] H. Komurasaki, H. Sato, M. Ono, T. Ebana, H. Takeda, K. Takahashi, Y. Hayashi, T. Iga, K. Hasegawa, and T. Miki, "A single-chip 2.4 GHz RF transceiver LSI with a wide-range FV conversion demodulator," in *IEEE ISSCC Dig. Tech. Papers*, 2001, pp. 206–207.

[20] Y.-H. Hsieh, W.-Y. Hu, S.-M. Lin, C.-L. Chen, W.-K. Li, S.-J. Chen, and D.-J. Chen, "An auto-I/Q calibrated CMOS transceiver for 802.11g," in *IEEE ISSCC Dig. Tech. Papers*, 2005, pp. 92–93.

[21] S. Mehta, D. Weber, M. Terrovitis, K. Onodera, M. P. Mack, B. J. Kaczynski, H. Samavati, H.-M. Jen, W. W. Si, M. Lee, K. Singh, S. Mendis, P. J. Husted, N. Zhang, B. McFarland, D. K. Su, T. H. Meng, and B. A. Wooley, “An 802.11g WLAN SoC,” in *IEEE ISSCC Dig. Tech. Papers*, 2005, pp. 94–95.

[22] D. Banerjee*, PLL Performance, Simulation, and Design*, 4th ed. Santa Clara, CA: National Semiconductor, 2005.

[23] I. Galton, “Delta–sigma fractional-N phase-locked loops,” in *Phase-Locking In High-Performance Systems*. New York: Wiley, 2003, pp. 23–33.

[24] K. Woo, Y. Liu, and D. Ham, “Fast-locking hybrid PLL synthesizer combining integer and fractional divisions,” in *IEEE Symp. VLSI Circuits*, Jun. 2007, pp. 260–261.

Kyoungho Woo (S’03) received the B.S. degree in electrical engineering from Seoul National University, Seoul, Korea, in 1999, the M.S. degree in electrical engineering from Stanford University, Stanford, CA, in 2004, and is currently working toward the Ph.D. degree in electrical engineering at Harvard University, Cambridge, MA.

In 2003, he was with the Timing Deign Group, Sun Microsystems, Sunnyvale, CA, where he developed timing analysis codes for the Ultraspac III microprocessor. In 2006 and 2007, he was a part-time member of an analog design team with Cavium Networks, Marlborough, MA, where he developed a multistandard high-speed serial links transmitter. His current research interest lies in analog and mixed-signal integrated circuits design, with special emphasis on PLL frequency synthesizers and high-speed serial links.

Mr. Woo is a Fellow of the Korea Foundation for Advanced Studies. He was the recipient of the 2006 Analog Devices Outstanding Student Designer Award.

Yong Liu (S’03–M’08) received the B.S.E.E. and M.S.E.E. degrees from Tsinghua University, Beijing, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering at Harvard University, Cambridge, MA, in November 2007.

In 2001, he was with the Tsinghua Tongfang Microelectronics Company, Beijing, China, where he was involved with the second-generation Chinese RF ID card. In the summers of 2005 and 2006, he was with the Mixed-Signal Communications IC Design Group, IBM T. J. Watson Research Center, Yorktown Heights, NY. He has authored or coauthored 15 publications. He holds three U.S. patents. His Ph.D. research focus at Harvard was twofold. First, he was involved in the development of CMOS ICs in conjunction with microfluidic systems to magnetically manipulate individual biological cells and to detect bio-molecular activities monitoring nuclear spin relaxation. Second, he designed RF and mixed signal ICs, especially autonomic PLLs for multidomain synchronous clocking. He is currently with the IBM T. J. Watson Research Center.

Dr. Liu was the recipient of the 1996 Seagate Scholarship for Extraordinary Freshman and 2001 Motorola Scholarship at Tsinghua University. He was also the recipient of the 2002 Second Prize in the National Graduate EDA Competition, China and the 2004 Analog Devices Outstanding Student Designer Award.

Eun-Soo Nam (M’07) received the B.Sc. degree in physics from Kyung-Pook National University, Daegu, Korea in 1983, and M.Sc. and Ph.D. degrees in physics from the State University of New York, Buffalo, in 1992 and 1994, respectively.

Since 1985, he has been with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea, where his research has examined compound semiconductor devices for microwave circuits and opto-electronics for optical fiber communications. He is currently the Head of the Photonic Wireless Transceiver Module Team, ETRI, where he leads efforts to develop microwave photonic modules operating in the millimeter-wave region to realize broadband photonic wireless communication, and laser radars that are safe to human visions. In 2006, he was a Visiting Scholar with the Division of Engineering and Applied Sciences (now the School of Engineering and Applied Sciences), Harvard University, Cambridge, MA. He holds over 50 patents in several fields, including heterojunction bipolar transistors (HBT), microwave monolithic integrated circuits (MMICs), long-wavelength InP semiconductor photonic devices, and opto-electronic integrated circuits (OEICs).

Donhee Ham (S’99–M’02) received the B.S. degree in physics from Seoul National University, Seoul, Korea, in 1996, graduating with Presidential Honor atop the Natural Science College, and the Ph.D. degree in electrical engineering from the California Institute of Technology (Caltech), Pasadena, in 2002, winning the Charles Wilts Doctoral Thesis Prize (best thesis award in electrical engineering). His doctoral research examined the statistical physics of electrical circuits.

He is currently John L. Loeb Associate Professor of the Natural Sciences with Harvard University, Cambridge, MA, where he is with the School of Engineering and Applied Sciences. His work experiences also include the Caltech-MIT Laser Interferometer Gravitational Wave Observatory (LIGO), the IBM T. J. Watson Research Center, IEEE conference technical program committees including the International Solid-State Circuits Conference (ISSCC), and industry/government technical advisory positions on subjects including ultrafast solid-state electronics and science and technology at the nanoscale. His research group at Harvard University currently works in several areas of electrical engineering and applied physics, specifically, RF, microwave and analog ICs, gigahertz-to-terahertz ultrafast quantum circuits using low-dimensional nanoscale devices, soliton and nonlinear wave electronics, and applications of CMOS ICs in biotechnology.

Dr. Ham was a Fellow of the Korea Foundation for Advanced Studies, and the recipient of the IBM Graduate Research Fellowship. He was also the recipient of the IBM Faculty Partnership Award. He was the co-recipient of the 2003 Harvard Hoopes Prize with Mr. William Andress. He is a co-editor of *CMOS Biotechnology* (Springer, 2007).