PHASE-LOCKED LOOP ARCHITECTURE FOR ADAPTIVE JITTER OPTIMIZATION

Socrates D. Vamvakos[1], *Carl Werner*[2] *and Borivoje Nikolić*[1]

[1]University of California, Berkeley, CA 94720 – [2]Rambus, Inc.

ABSTRACT

A phase-locked loop (PLL) architecture is presented that allows adaptive optimization of tracking jitter by using an on-chip jitter estimation block. The jitter estimation circuit operates at the PLL reference clock frequency and is composed of digital blocks, improving the robustness of the overall architecture. The jitter estimates may be used to adaptively tune the PLL loop parameters to achieve minimum jitter operation. System design considerations are discussed and simulation results are reported for a PLL in 0.13 µm CMOS technology.

1. INTRODUCTION

Increasing demand for higher microprocessor speeds, as well as data rates in communication systems has resulted in stringent performance requirements on clock synthesis and clock and data recovery (CDR) circuits used in these applications. The main building block of these systems is the phase-locked loop (PLL), whose timing jitter performance in most applications must be limited to a small fraction of the clock cycle.

Figure 1 shows a basic PLL architecture along with some of the noise sources that contribute to the timing jitter at the output of the PLL. The amount of contribution of the various noise sources to the output jitter depends on the noise power of the source and on the transfer function from the noise node to the output. It is known [1,2] that by adjusting the PLL loop parameters such as loop bandwidth and damping factor, the amount of output jitter can be modulated to achieve a minimum. The operating point of the PLL for which this minimum is achieved depends on the noise source characteristics, which may be difficult to determine a priori. Furthermore, process, voltage and temperature (PVT) variations, as well as changes in the operating frequency, may shift the optimal operating point of the PLL, causing suboptimal jitter performance. Therefore, an adaptive scheme, as in Figure 2(a) that would converge to a minimum jitter operating point, as shown in Figure 2(b), would be desirable.

In this paper we present a PLL architecture that allows adaptive optimization of the output tracking jitter by using an on-chip jitter estimation block. It is assumed that the reference clock phase noise is negligible compared to the rest of the noise sources in Figure 1 (supply, control node, thermal noise etc.).

The main issue with jitter estimation is that it cannot be carried out directly, unless the circuits used for the measurements can operate at speeds that are of the order of the jitter values to be measured, which can be as low as a few picoseconds. In this paper an indirect way of estimating jitter is used that can measure jitter values of the order of picoseconds.

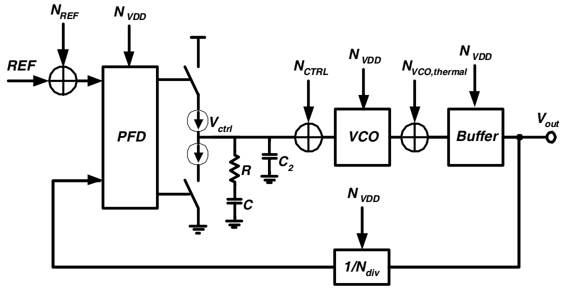
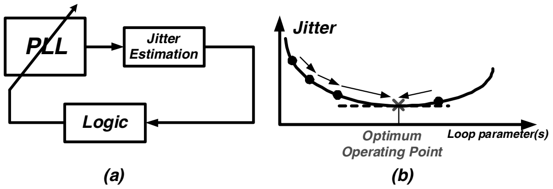


Figure 1: PLL with noise sources



(a) (b)

Figure 2: (a) Adaptive PLL architecture, (b) jitter optimization

The estimated jitter values can be used to adaptively tune the PLL loop parameters using a LUT or gradient-descent algorithm to achieve minimum jitter operation, as described in [3].

The paper is organized as follows: Section 2 describes the jitter measurement circuit along with a discussion of the estimation algorithm and choice of parameters. In Section 3, the overall system architecture is shown along with the various components. In Section 4 simulation results are presented.

3. JITTER ESTIMATION BLOCK

The block diagram of the jitter estimation circuit is shown in Figure 3. It consists of two voltage-controlled delay lines (VCDLs) whose output is a delayed version of the PLL reference clock. Each of the VCDL outputs is fed into an edge comparison circuit along with the PLL output clock whose jitter is to be measured. The top (bottom) edge comparator produces a '1', if the PLL edge occurs before V_{REF1} (after V_{REF2}). The number of hits H is counted over a time interval equal to N reference clock periods and compared to a target value M. The difference is used to adjust the VCDL control voltages in such a manner as to decrease the difference between H and M. The procedure is repeated until a convergence criterion is met. The end result is the creation of a dead-zone [4,5], the width of which gives an estimate of the PLL output jitter at the current operating conditions, as illustrated in Figure 4. It should be mentioned here that by using two independently controlled VCDLs instead of one as in [4,5], the number of iterations per

0-7803-8251-X/04/$17.00 ©2004 IEEE

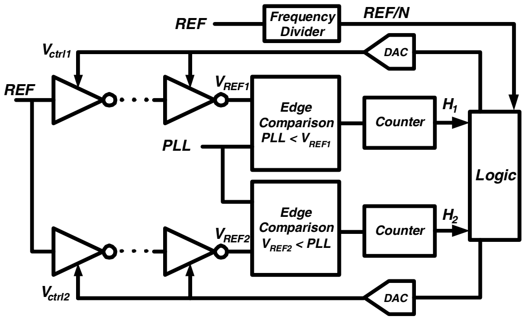


Figure 3: Jitter estimation circuit.

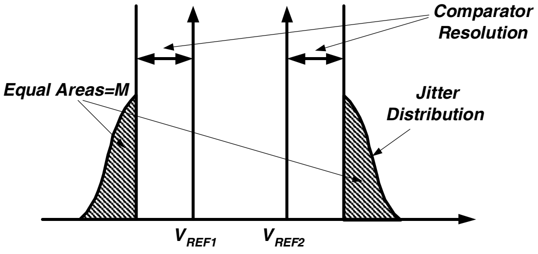


Figure 4: Tracking of jitter distribution edges by the VCDL outputs.

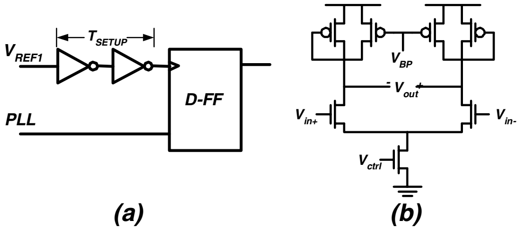


Figure 5: (a) Edge comparison circuit; (b) VCDL stage

cycle (*N*) which is required to obtain a statistically significant number of hits becomes independent of the jitter value. This is important in the case of very small jitter.

The jitter estimation circuitry introduces noise to the jitter measurement which is approximately uncorrelated to the PLL jitter, assuming that the common supply noise is approximately white with high frequency deterministic components. This measurement noise is also independent of the PLL operating point. Therefore, when comparing the jitter estimates at various operating points, the noise introduced by the measurement circuit constitutes a systematic error, and as such it is cancelled out. The PLL jitter component that is the result of low frequency deterministic supply noise may be partially masked during the measurement.

The edge comparison circuit is shown in Figure 5a. It uses the metastability property of a flip-flop to achieve a resolution of a few picoseconds over all process corners. The finite resolution of the edge comparator is again a systematic error and will be cancelled out. The VCDL stage is shown in Figure 5b. It uses symmetric loads for improved PSRR [6] and achieves an average resolution of 1 ps.

Figure 6 describes the algorithm used for updating the VCDL control voltage V_{ctrl1}. *DAC* denotes the DAC codeword, *Δmax* is the maximum allowable change in the DAC codeword at any given step, *H* is the counter value at the end of the current iteration and *M* is the target value. When the difference |*H-M*| becomes less than some predetermined value *K*, then the update algorithm enters a fine search mode, where *Δmax*=1 until the algorithm converges.

```
DAC = DAC0; $\Delta$max=$\Delta$0; $\Delta$sign = 1 ;
converge=0;

while ( converge == 0)
        $\Delta$sign_old = $\Delta$sign; $\Delta$sign = sign{H-M} ;
        if ( $|H-M| < \Delta max$)
                $|\Delta DAC| = |H-M|$ ;
        elseif ($\Delta sign \times \Delta sign\_old > 0$ )
                $|\Delta DAC| = \Delta max$;
        elseif ($\Delta sign \times \Delta sign\_old < 0$ )
                $|\Delta DAC| = \Delta max-1$;
        else
                $|\Delta DAC| = 0$;
        end

        if ( $|H-M| > K$ )
                $\Delta DAC = \Delta sign \times |\Delta DAC|$;
                $\Delta max = |\Delta DAC|$;
        elseif ( $|H-M| \leq K$ )
                if ( $|\Delta DAC| > 0$ )
                    % Enter fine search mode:
                    $\Delta DAC = \Delta sign$;
                    $\Delta max = 1$;
                else
                    $\Delta DAC = 0$;
                    $\Delta max = 0$;
                end
        end
        if ($\Delta DAC == 0$)
                converge = 1;
        end
end
```

Figure 6: Update algorithm for V_{ctrl1}. We assume negative delay coefficient for the VCDL.

The jitter estimation algorithm described in Figure 6 is guaranteed to converge. However, it is obvious that its performance depends on the choice of the parameters *N*, *M*, *K*, as well as the stochastic characteristics of the PLL jitter. Figure 7 shows the standard deviation of the jitter estimate produced by the algorithm as a function of the number of cycles per iteration (*N*) and the PLL RMS jitter (the VCDL resolution is assumed to be 1 ps). As expected, the variance of the estimate increases for smaller *N* and larger PLL jitter values. Having $N \geq 2^{14}$ essentially eliminates the uncertainty in the jitter estimate.

Figure 8 shows the dependence of the average number of iterations required for the algorithm to converge vs. the ratio *K/N*, where *K* is the parameter that shifts the jitter estimation algorithm into fine search mode, as described in Figure 6. For large values of *K*, the algorithm enters the fine search mode prematurely, while for small values of *K* the algorithm oscillates around the correct estimate. This is illustrated in Figure 8, where it is seen that a value *K/N*≈1% results in a minimum number of expected iterations.

3. SYSTEM ARCHITECTURE

The overall system architecture is shown in Figure 9. Before the operation of the jitter estimation block begins, the DAC codewords are initialized so that both VCDL delays are equal to half of the delay range. The delay line control voltage V_{DL} is subsequently adjusted so that the edges of the PLL output clock

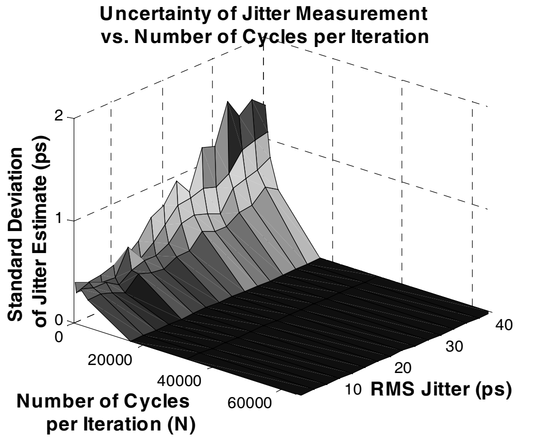


Figure 7: Uncertainty of final jitter estimate vs. number of cycles per iteration.

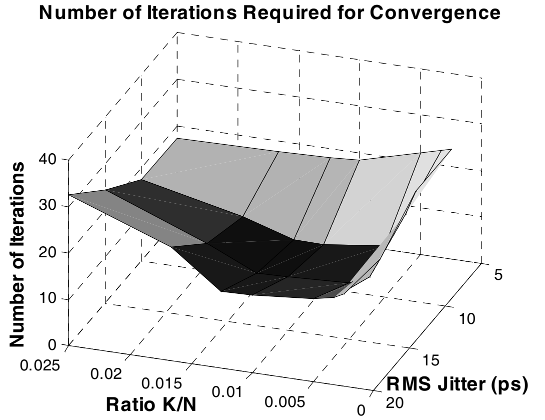


Figure 8: Average number of algorithm iterations required for convergence

and the VCDL outputs are aligned. This procedure provides the maximum dynamic range for the jitter measurement.

The PLL uses a dual charge-pump self-biased architecture [6] as shown in Fig. 10. The charge pump currents are controlled by 8-bit DACs, which allow adjustment of the PLL loop parameters such as loop bandwidth ω_N and damping factor ζ.

4. SIMULATION

Transistor-level simulations of the system in figure 9 were performed using NanoSim^{TM} [7] to verify correct operation. Gaussian noise was superimposed on the PLL supply for various settings of the PLL DACs, and the final values of the jitter block DACs were read after the algorithm converged. The corresponding delay difference $\Delta\tau$ of the VCDLs was determined through simulation and the RMS jitter of the PLL output was estimated using the equation

$$\sigma_{EST} = \frac{\Delta\tau}{2\sqrt{2}\times erfinv\left(1-\frac{2\times M}{N}\right)} \qquad (1)$$

where $erf(x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ is the error function and *erfinv* the inverse error function. In addition, the RMS jitter was calculated directly from the PLL output waveform for comparison purposes. The system parameters used in the simulations were N=2^{10}, M=50 and K=10. The PLL can operate from 0.8 to 2.5 GHz.

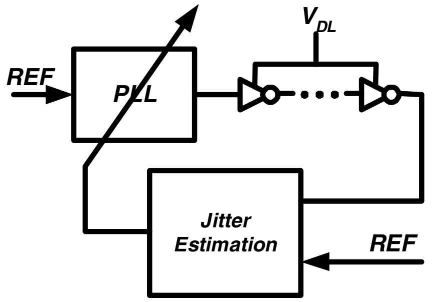


Figure 9: System architecture.

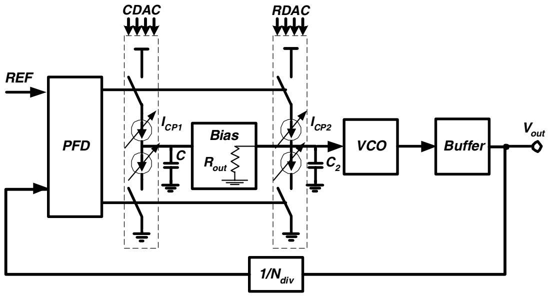


Figure 10: Self-biased PLL with adjustable charge pumps.

Figure 11 shows the sequence of jitter estimates during two simulation runs with different supply noise powers. Figure 12 shows the PLL output jitter estimates for different supply noise powers. The curve denoted "Clean Estimate" is obtained in the case where the jitter block supply is clean. The curve denoted "Noisy Estimate" is obtained in the case where the jitter block supply is the same as the noisy supply of the PLL. The estimates are compared to the jitter values obtained through direct observation of the PLL output waveform. It can be seen that the relative error of both the clean and noisy estimates is less than 20% for jitter values as low as few picoseconds.

The feasibility of modulating the output jitter by adjusting the ratio of the charge pump currents I_{CP2}/I_{CP1} for a PLL architecture similar to the one in Figure 10, has been examined in [2]. Increasing I_{CP2}, while I_{CP1} is kept constant, increases the equivalent damping factor ζ, while keeping the loop bandwidth ω_N constant. This initially reduces jitter peaking and decreases the output jitter. Further increase of I_{CP2} increases the unity gain frequency ω_0 of the loop transfer function, bringing it closer to the third-order pole of the system. This causes phase margin degradation which in turn increases the output jitter, as shown in Figure 2(b).

Figure 13 shows the estimated and measured jitter at various PLL operating points for two different output frequencies. The curves were obtained by varying the current I_{CP2} while keeping I_{CP1} constant, as described above. Again, the jitter estimates are close to the measured values. More importantly, the minimum jitter operating point is correctly identified in both cases. It can be seen that the minimum jitter value is obtained for current ratios of approximately 1.5 and 2.5 when the operating frequency is 2.5 GHz and 1.25 GHz,

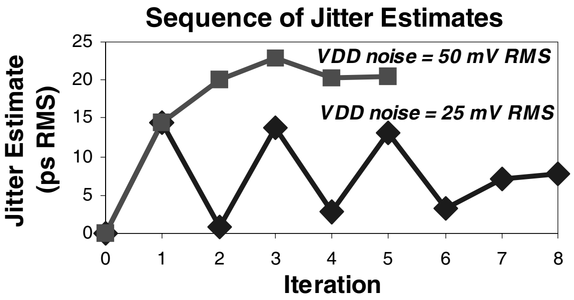


Figure 11: Sequence of jitter estimates during simulation run.

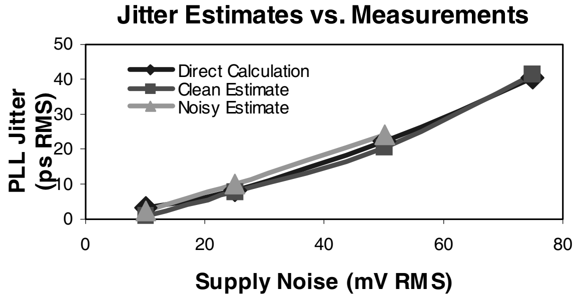


Figure 12: PLL jitter estimated values vs. direct calculation.

respectively. Thus, a change in the PLL operating frequency may result in suboptimal jitter performance up to 25%, unless the loop parameters are adjusted correspondingly. It should be noted here that the non-convexity of the curves at 2.5 GHz can be attributed to statistical variations, since the number of cycles per iteration (*N*) is rather small. Figure 14 shows the estimated and measured jitter at various PLL operating points for two different process corners at 2.5 GHz. Again, the minimum jitter operating condition is correctly identified in both cases. It can be seen that process variations can cause suboptimal jitter performance of up to 20%.

5. CONCLUSION

An adaptive PLL architecture is proposed that uses an on-chip jitter estimation circuit to determine the PLL jitter at various operating conditions. The jitter estimation algorithm was presented and analyzed. Simulation results show that the system correctly identifies the minimum jitter operating condition of the PLL.

6. ACKNOWLEDGMENTS

The first author would like to thank the LID group at Rambus Inc., especially Vladimir Stojanović and Jared Zerbe for valuable technical help and support. This work was supported by MARCO C2S2.

7. REFERENCES

[1] K. Lim et al., “A Low-Noise Phase-Locked Loop Design by Loop Bandwidth Optimization”, *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 807-815, June 2000.

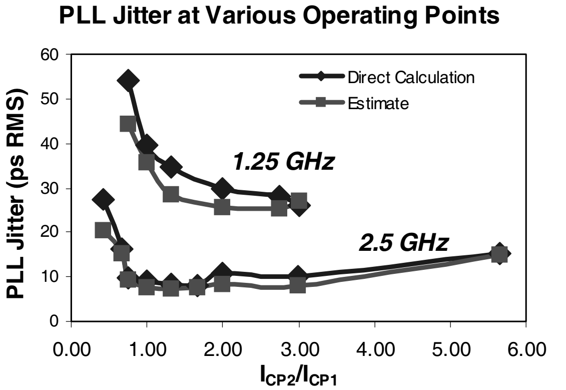


Figure 13: PLL jitter at various operating points for two different operating frequencies.

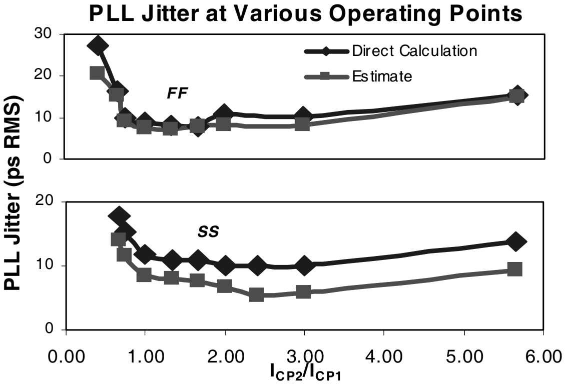


Figure 14: PLL jitter at various operating points for two different process corners at 2.5 GHz.

[2] M. Mansuri and C.-K. K. Yang, “Jitter Optimization Based on Phase-Locked Loop Design Parameters”, *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1375-1382, Nov. 2002.

[3] M. Mansuri et al., “Methodology for On-Chip Adaptive Jitter Minimization in Phase-Locked Loops”, *IEEE Trans. On Circuits and Systems – II*, vol. 50, No. 11, pp. 870-878, Nov. 2003.

[4] Y. Moon et al., “A 0.6-2.5-GBaud CMOS Tracked 3x Oversampling Transceiver with Dead-Zone Phase Detection for Robust Clock/Data Recovery”, *IEEE Journal of Solid-State Circuits*, vol. 36, pp. 1974-1983, Dec. 2001.

[5] B.-J. Lee et al., “A 2.5-10 Gb/s CMOS Transceiver with Alternating Edge Sampling Phase Detection for Loop Characteristic Stabilization”, *ISSCC Digest of Technical Papers*, pp. 76-77, Feb. 2003.

[6] J. G. Maneatis, “Low-Jitter Process-Independent DLL and PLL Based on Self-Biased Techniques”, *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 1723-1732, Nov. 1996.

[7] Synopsys, “NanoSim Reference Guide”, TLD 2001.06, June 2001.