IEEE Asian Solid-State Circuits Conference
November 14-16, 2011 / Jeju, Korea

Extending HyperTransport™ Technology to 8.0 Gb/s in 32-nm SOI-CMOS Processors

Bruce A. Doyle[1], Alvin L. S. Loke[1], Sanjeev K. Maheshwari[2], Charles L. Wang[2], Dennis M. Fischette[2], Jeffrey G. Cooper[1], Sanjeev K. Aggarwal[2], Tin Tin Wee[1], Chad O. Lackey[1], Harishkumar S. Kedarnath[3], Michael M. Oshima[2], Gerry R. Talbot[4], and Emerson S. Fang[2]

[1]Advanced Micro Devices, Inc., 2950 East Harmony Road, Fort Collins CO 80528-3419, USA
[2]Advanced Micro Devices, Inc., 1 AMD Place, Sunnyvale CA 94085-3905, USA
[3]Advanced Micro Devices, Inc., 7171 Southwest Parkway, Austin TX 78735-8953, USA
[4]Advanced Micro Devices, Inc., 90 Central Street, Boxborough MA 01719-1200, USA
bruce.doyle@ieee.org, alvin.loke@ieee.org

***Abstract*—We present an 8.0-Gb/s HyperTransport™ technology I/O built in a 32-nm SOI-CMOS processor for high-performance servers. Based on a 45-nm design that caps at 6.4 Gb/s, the 32-nm transceiver achieves up to 8.0 Gb/s over long-reach board channels. Key enhancements include a high-bandwidth (>200 MHz) PLL to attenuate high-frequency jitter in the received forwarded clock and redesigned power-hungry circuits to operate at 8.0 Gb/s within the existing 45-nm package thermal limit.**

I. Introduction

The performance of AMD servers critically depends on the maximum die-to-die transfer rate supported by coherent HyperTransport™ (HT) I/Os in AMD processors [1]. Unlike processor connectivity to other peripherals (such as PCI Express® to Southbridge, I/O hubs, and external GPU), the HT interface always links two AMD processors and thus need not conform to the 5.2 Gb/s HT3 protocol limit for interoperability. We have already pushed the HT rate to 6.4 Gb/s (HT3+) in the 45-nm Opteron™ 6100 Series processor [2]. Extending HT3+ in 32 nm, we designed the HT I/O to operate at 8.0 Gb/s, beyond the officially supported 6.4 Gb/s, to examine the system performance potential of AMD's new "Bulldozer" core in the new server processor codenamed Orochi [3].

This paper addresses the key enhancements beyond the 45-nm HT design [2], [4] to achieve 8.0 Gb/s across challenging channels while drawing only 5% more power. First, we introduce a wideband PLL to remove high-frequency jitter in the received forwarded clock. Jitter filtering improves receiver (RX) retiming margin as data and clock jitter become increasingly uncorrelated at higher jitter frequency due to transport delay mismatch. Second, subject to the existing 45-nm package and its thermal limit, we redesign power-hungry blocks such as the transmitter (TX) driver and RX deserializer to operate at the higher data rate without consuming more power. This also preserves power delivery to the CPU cores for uncompromised core performance. In a four-socket server configuration, a 25% boost in link speed enables processor performance to improve by 4.2% for database applications and up to 25% for I/O-intensive applications.

II. HyperTransport in AMD Processors

HyperTransport technology is the full-duplex point-to-point parallel link protocol used by AMD processors for high-bandwidth die-to-die communication [1]. Shown in Fig. 1, the data transfer is source-synchronous where the received NRZ data stream is re-timed by a forwarded TX clock for common-mode jitter rejection and low link latency. Each link direction consists of two sub-links, each comprising of nine data lanes and an accompanying half-rate forwarded-clock lane. An integrated PLL derives the TX clock from a 200-MHz spread-spectrum source that is typically shared across all sockets on a single FR-4 server board. The board channels are 100 Ω differential with worst-case lengths of 30 inches and possibly two interposing connectors to bridge sockets mounted on different daughter boards.

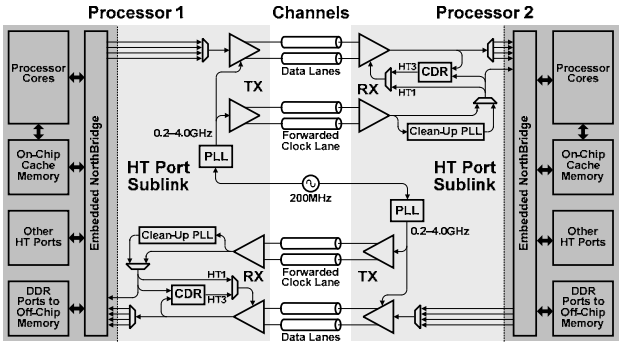


Figure 1. HyperTransport sub-link bridging two AMD processors.

Per-lane link transfer rates of 0.4 to 6.4 Gb/s (extended to 8.0 Gb/s in 32 nm) in 0.4-Gb/s increments and data lane widths can be selected based on real-time aggregate bandwidth needs and operating power constraints. At 2.0 Gb/s and below (HT1 mode), the RX simply samples received data using the received forwarded clock without clock-to-data deskewing. Beyond 2.0 Gb/s (HT3/HT3+ mode), independent per-lane DLL-based clock and data recovery (CDR) is activated to align the received clock phase to data transitions for better sampling margin. The operation of all HT ports is on-demand and managed by the embedded NorthBridge (NB) memory controller which arbitrates the flow of all data between the CPU cores, cache memories, and I/Os. When data must be transferred to another die, NB awakens the appropriate HT port, coordinates the link handshake and data transfer, and returns that port to a sleep state after use.

III. Forwarded Clock Jitter Filtering

A. *Motivation*

In a source-synchronous link, the TX data jitter will track the forwarded TX clock jitter for good RX data sampling margin if the transport delay of the data path (from TX final re-timing to RX data sampling) matches that of the forwarded

clock path. The higher link tolerance to TX clock jitter is especially important in SOI-CMOS because low-power, low-jitter TX PLLs are difficult to design due to body noise [5]. Unfortunately, even in carefully optimized practical links with matched board channels, data and clock transport delays can be mismatched by as much as several nanoseconds due to uncompensated CDR loop and clock distribution latencies. This transport delay mismatch, τ, indicates that at some jitter frequencies as low as hundreds of MegaHertz, data jitter will become completely out of phase or anti-correlated from clock jitter, resulting in doubling of sampling jitter when the clock samples data. This is of particular concern in processors in which significant supply noise exists in the 100–500 MHz band due to package resonances.

We quantify the effect by expressing the phase of the data signal, $\varphi_{data}(t)$, as a carrier of frequency f_c with sinusoidal jitter modulation of frequency f_m ($< f_c$) and amplitude A_m.

$$\varphi_{data}(t) = 2\pi f_c t + A_m \sin(2\pi f_m t) \quad (1)$$

The forwarded clock phase, $\varphi_{clock}(t) = \varphi_{data}(t-\tau)$, is simply a replica of $\varphi_{data}(t)$ shifted in time by τ as illustrated in Fig. 2. The sampling jitter can then be defined as

$$\begin{aligned}\Delta\varphi(t) &= \varphi_{data}(t) - (\varphi_{clock}(t) + 2\pi f_c\tau) \\ &= (2A_m \sin 2\pi f_m\tau)\sin(2\pi f_m t + \cot^{-1} 2\pi f_m\tau) \quad (2)\end{aligned}$$

where the static phase offset $2\pi f_c\tau$ is removed by DLL action in the CDR. The result is the following normalized jitter magnitude transfer function

$$J_{sample}(f_m, \tau) = 2 \cdot |\sin(\pi f_m \tau)|. \quad (3)$$

Eq. (3) shows that the sampling jitter magnitude transfer is periodic in jitter frequency. Sampling jitter vanishes at $f_m = N/\tau$ for $N = 0,1,2,\ldots$ when clock jitter is exactly in phase with data jitter (Fig. 3(a)). However, it exceeds unity when $(N + 1/6)/\tau < f_m < (N + 5/6)/\tau$ and even doubles at $f_m = (N + 1/2)/\tau$ when clock jitter is 180° out of phase from data jitter (Fig. 3(b)). For the example $\tau = 2$ ns, sampling jitter vanishes at 500, 1000, 1500 MHz, … but becomes amplified at 83–417, 583–917, 1083–1417 MHz, … This jitter amplification across two-thirds of the modulation spectrum can be mitigated by conditioning the received forwarded clock with a wideband PLL [6], [7]. By removing high-frequency jitter from the clock, we bound the jitter transfer to unity as shown in Fig. 4. The high modulation bandwidth is key to maintain jitter tracking at lower jitter frequencies. Otherwise, total sampling jitter across the entire jitter spectrum may worsen due to low-frequency contribution. Bandwidth (BW) selection clearly depends on τ, which is prone to vary across process variation and operating conditions.

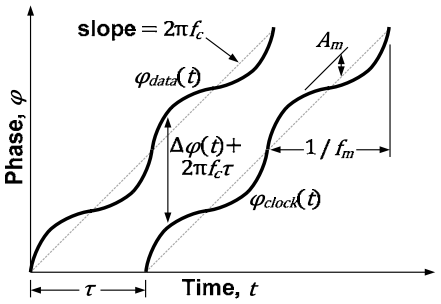
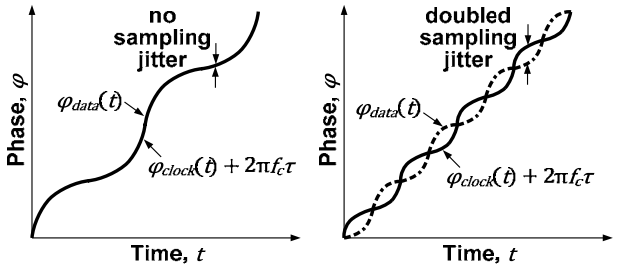
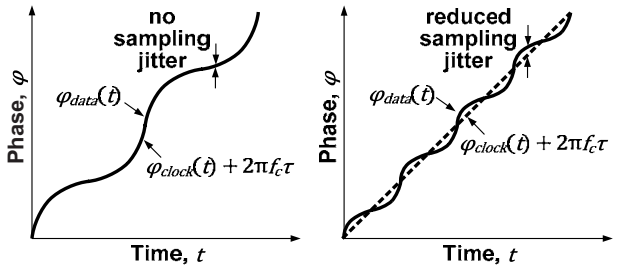


Figure 2. Phase relationship between jitter-modulated data and clock with mismatch in transport delay.



(a) (b)

Figure 3. (a) Vanishing and (b) doubling of sampling jitter.



(a) (b)

Figure 4. Clean-up PLL action at (a) $f_m \ll BW$ and (b) $f_m \gg BW$.

B. *Wideband Digital Clean-Up PLL*

Fig. 5 illustrates the wideband clean-up PLL architecture [8]. A digital solution was chosen for operation with a high input reference clock frequency (1–4 GHz), low power, and bandwidth adjustability [9]. The PLL has three control loops: phase (proportional), frequency (integral), and calibration.

The phase loop is driven by an early/late phase comparator with additional feedback clock phases for finer comparison resolution. Additional phase information improves bandwidth for better rejection of high-frequency jitter. To maximize bandwidth with minimal steady-state dithering, the VCO is driven directly by the phase comparator outputs, achieving a nominal phase loop latency of only two VCO clock cycles. The instantaneous VCO frequency bangs up or down by f_{bb1} or $f_{bb1} + f_{bb2}$ in response to early/late control from the phase comparator. The PLL bandwidth can be adjusted by controlling the magnitude of f_{bb1} and/or f_{bb2} inside the VCO. A programmable digital filter may also be inserted to average the phase comparator outputs for bandwidth reduction.

The frequency loop averages the phase comparator outputs and drives the VCO with a 14-bit ΔΣ DAC. The calibration loop compensates for process variation and the desired VCO frequency. The nominal PLL bandwidth is set to ~200 MHz, which comfortably enables the RX sampler to track 33-kHz spread-spectrum modulation while reducing sampling jitter for τ below ~2 ns.

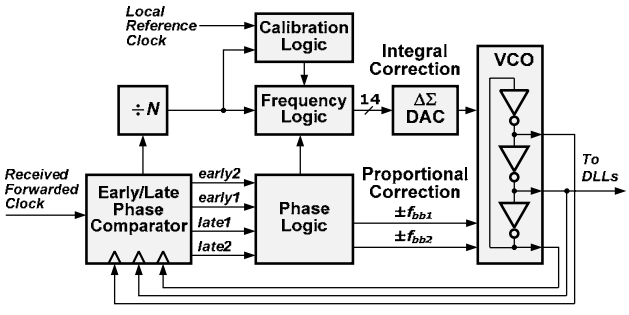


Figure 5. Block architecture of wideband digital clean-up PLL.

IV. Power Optimization

A. *Transmitter Output Driver*

The differential TX features a four-tap FIR filter with one pre- and two post-cursors for channel equalization. Power is reduced in the 32-nm design by migrating from a hybrid-mode driver with current-mode de-emphasis (Fig. 6(a)) to a pure voltage-mode driver with voltage-mode de-emphasis (Fig. 6(b)) [10], [11]. Compared to 32 nm, the 45-nm driver drew a higher quiescent current due to the need for current steering to quickly switch the TX output and to static biasing associated with the current sources. Moreover, the large parasitic capacitance of the current source legs at the TX output degraded return loss.

The 32-nm driver is implemented as a parallel bank of selectable drivelet cells with weighted pull-up and pull-down resistances. Drivelets are allocated and assigned based on both the required termination impedance and de-emphasis level. An integer-based state machine computes and selects the correct number of weighted drivelet cells to ensure that a net 50 Ω output resistance is always maintained across any de-emphasis level. The drivelets are carefully grouped to reduce power and minimize output glitching when de-emphasis updates occur. Most updates will switch from main data to one of the other cursors or vice versa. All assignments are computed and then updated on the same clock cycle. The architecture corrects for chip-mean variation in poly resistance by calibrating to an accurate off-package reference resistor [12]. FET contribution to the overall resistance was optimized to achieve good linearity but not present too much load to the pre-driver. The available de-emphasis range extends beyond the HT protocol requirement to facilitate RX eye margining tests. Since a half-rate architecture is chosen, minimizing duty cycle distortion (DCD) is important. Care was taken in structuring and implementing the clock tree and predrivers to ensure a tolerable level of DCD.

The 45-nm mixed-mode TX consumed 41 mW at 6.4 Gb/s on a 1.2-V supply. The corresponding 32-nm voltage-mode TX power was 33 mW, reflecting a 20% reduction.

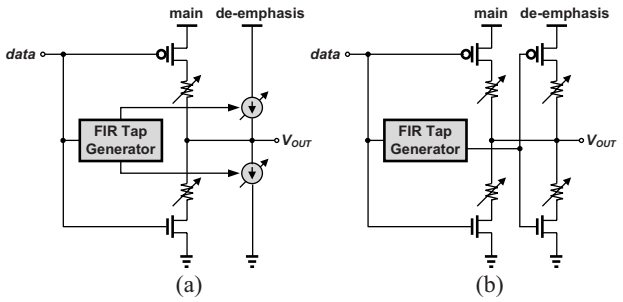


Figure 6. TX output driver half-circuit in (a) 45 nm and (b) 32 nm.

B. *Receiver Deserializer*

The 4:1 RX deserializer power was reduced by partitioning the deserializer into stages clocked at successively divided frequencies. In the 45-nm design (Fig. 7(a)), serial data are first captured into a four-stage shift register clocked at full rate. Every four cycles, the shift register outputs are transferred and subsequently split into *A* and *B* phases as a processor protocol requirement.

Power is reduced in our 32-nm design by employing the deserializer structure of Fig. 7(b). Data capture is split into two half-rate paths that latch data on both rising and falling clock edges. Each half-rate path is then recursively further deserialized into two-bit wide *A* and *B* data outputs using both edges of *div4clk*. With fewer flops and lower frequency clocking, we shaved the RX power to 12.5 mW, a 35% reduction from 45-nm power at the same 1.2-V supply.

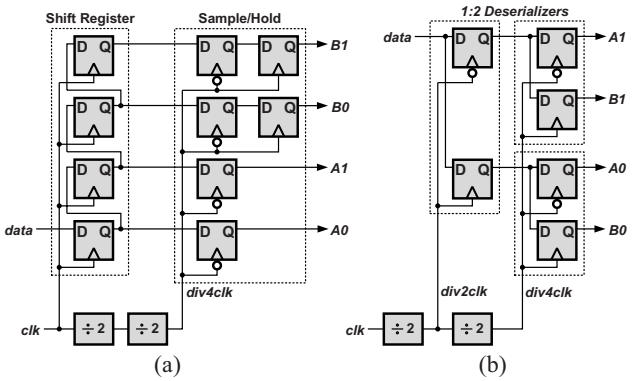


Figure 7. (a) 45- and (b) 32-nm deserializer architecture.

V. Other Transceiver Enhancements

The RX input path is AC-coupled on die to reduce TX DCD which manifests as a DC component in the clock spectrum [4]. Each picosecond of DCD translates to a higher percentage of deterministic jitter at higher data rates. We employ a one-tap speculative decision feedback equalizer to minimize the impact of pattern-based inter-symbol interference.

The RX CDR utilizes a six-stage DLL to generate 30°-spaced phases in conjunction with a 16-step phase interpolator to enable RX sampling at 1/96 UI resolution [13]. The DLL is powered by a single regulated supply, allowing for superior supply noise rejection and fewer power-hungry level shifters.

VI. Silicon Results

Four HT I/Os were integrated in the AMD server processor codenamed Orochi, shown in Fig. 8, which was fabricated in a GlobalFoundries 32-nm SOI-CMOS technology [14].

Packaged parts were tested using a ParBERT configured according to Fig. 9. The HT I/O was configured into a loopback mode that included the I/O controller in the embedded NB. This ensured sub-system test coverage of the entire I/O. Nominal supply was 1.2 V. The maximum data rate was extended to 8.0 Gb/s although Orochi officially supports up to only 6.4 Gb/s. Fig. 10 shows an 8.0 Gb/s eye opening at the TX output broadcasting a PRBS-15 pattern. Output peak-to-peak jitter was measured at approximately 0.12 UI.

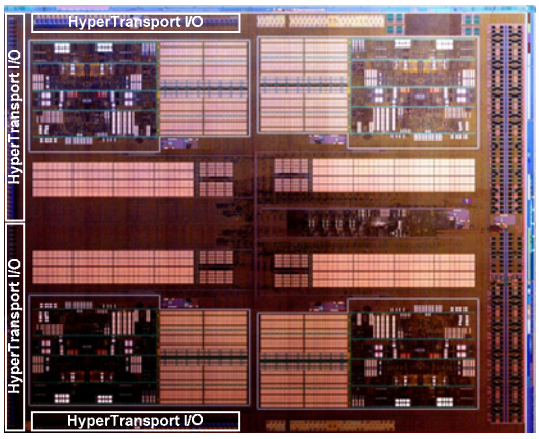


Figure 8. Photograph of 32-nm SOI-CMOS "Orochi" die.

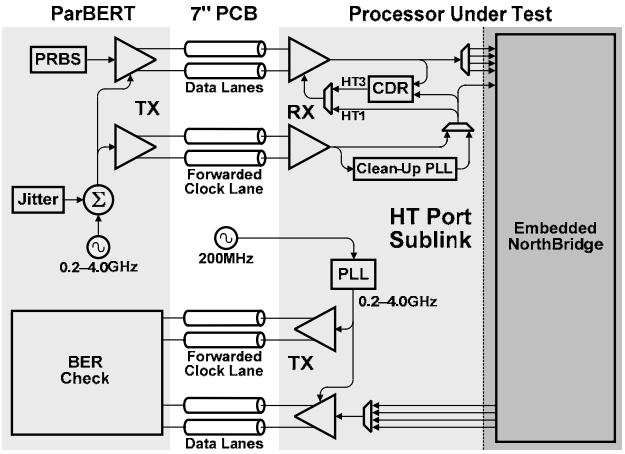


Figure 9. Test configuration to measure impact of clean-up PLL.

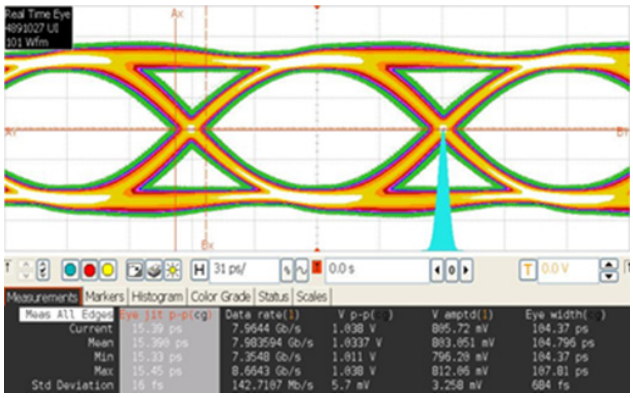


Figure 10. 8.0 Gb/s eye diagram of TX transmitting PRBS-15.

Table I compares the bit error rate (BER) performance with and without the clock clean-up PLL activated. An 8.0-Gb/s PRBS-9 pattern was used to assess the performance of the clean-up function. BERs were examined across all 18 data lanes of the HT I/O with the ParBERT introducing common clock and data jitter 0.40 UI amplitude and 10–500 MHz modulation. BER was reduced by over three orders of magnitude with the clean-up PLL enabled. The impact of the clean-up PLL vs. jitter frequency is demonstrated in Fig. 11. With the PLL enabled, a small peak can be noted near the PLL bandwidth. Beyond that frequency, the impact of the forwarded clock jitter is substantially mitigated. Table II shows that the 32-nm power optimization reduced the power consumption at 6.4 Gb/s by 14%. Consumption at 8.0 Gb/s increased a mere 5% compared to 45 nm running at 6.4 Gb/s.

Table I: Effect of Clean-Up PLL on Bit Error Rate

Jitter Frequency (MHz)	BER at 8.0 Gb/s	
	PLL Disabled	PLL Enabled
10	$< 10^{-12}$	$< 10^{-12}$
500	2.7×10^{-9}	3.3×10^{-12}

Table II: Power Consumption

Data Rate (Gb/s)	Technology	HT I/O Power (W)
6.4	45-nm SOI-CMOS	1.62
6.4	32-nm SOI-CMOS	1.40
8.0	32-nm SOI-CMOS	1.70

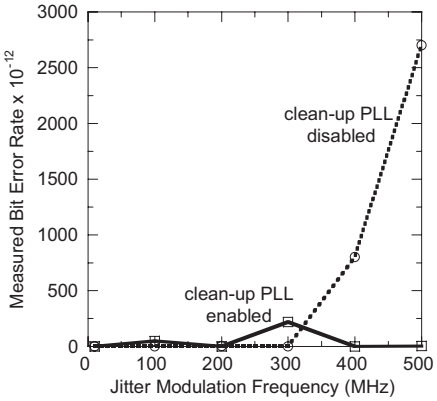


Figure 11. Bit error rate vs. jitter frequency with and without clean-up PLL enabled at data rate of 8.0 Gb/s.

ACKNOWLEDGMENT

We are indebted to the AMD and Coast-to-Coast layout teams, and to Richard DeSantis for bench measurements.

REFERENCES

[1] http://www.hypertransport.org
[2] A. Loke *et al.*, "Loopback architecture for wafer-level at-speed testing of embedded HyperTransport™ processor links," in *Proc. IEEE Custom Integrated Circuits Conf.*, pp. 605–608, Sep. 2009.
[3] T. Fischer *et al.*, "Design solutions for the Bulldozer 32nm SOI 2-core processor module in an 8-core CPU," in *IEEE Int. Solid-State Circuits Conf. Tech. Dig.*, pp. 78–79, Feb. 2011.
[4] E. Fang *et al.*, "A 5.2Gbps HyperTransport™ integrated AC coupled receiver with DFR DC restore," in *IEEE Symp. VLSI Circuits Tech. Dig.*, pp. 34–35, Jun. 2007.
[5] D. Fischette *et al.*, "A 45nm SOI-CMOS dual-PLL processor clock system for multi-protocol I/O," in *IEEE Int. Solid-State Circuits Conf. Tech. Dig.*, pp. 246–247, Feb. 2010.
[6] M. Li *et al.*, "Transfer functions for the reference clock jitter in a serial link: theory and applications," in *Proc. Int. Test. Conf.*, pp. 1158–1167, Oct. 2004.
[7] R. Reutemann *et al.*, "A 4.5 mW/Gb/s 6.4 Gb/s 22+1-lane source synchronous receiver core with optional cleanup PLL in 65 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 45, no. 12, pp. 2850–2860, Dec. 2010.
[8] S. Maheshwari, E. Fang, and S. Aggarwal, "32-nm SOI programmable, high-bandwidth 8.0-GHz digital PLL," in *Proc. IEEE Custom Integrated Circuits Conf.*, Paper M-15, Sep. 2011.
[9] R. Walker, "Designing bang-bang PLLs for clock and data recovery in serial data transmission systems," in *Phase-Locking in High-Performance Systems – From Devices to Architecture*, B. Razavi, ed., 2003.
[10] M. Kossel *et al.*, "A T-coil-enhanced 8.5 Gb/s high-swing SST transmitter in 65 nm bulk CMOS with < –16 dB return loss over 10 GHz bandwidth," *IEEE J. Solid-State Circuits*, vol. 43, no. 12, pp. 2905–2920, Dec. 2008.
[11] R. Philpott *et al.*, "A 20Gb/s SerDes transmitter with adjustable source impedance and 4-tap feed-forward equalization in 65nm bulk CMOS," in *Proc. IEEE Custom Integrated Circuits Conf.*, pp. 623–626, Sep. 2008.
[12] J. Feng *et al.*, "Bridging design and manufacture of analog/mixed-signal circuits in advanced CMOS," in *IEEE Symp. VLSI Technology Tech. Dig.*, pp. 226–227, Jun. 2011.
[13] G. Wei *et al.*, "A variable-frequency parallel I/O interface with adaptive power-supply regulation," *IEEE J. Solid-State Circuits*, vol. 35, no. 11, pp. 1600–1610, Nov. 2000.
[14] M. Horstmann *et al.*, "Advanced SOI CMOS transistor technologies for high-performance microprocessor applications," in *Proc. IEEE Custom Integrated Circuits Conf.*, pp. 149–152, Sep. 2009.