A 400MHz - 1.6GHz Fast Lock, Jitter Filtering ADDLL based Burst Mode Memory Interface

Masum Hossain[1,2], Farrukh Aquil[1], Pak Chau[1], Brian Tsang[1], Phuong Le[1], Jason Wei[1], Teva Stone[1], Barry Daly[1], Tran Chanh[1], John Eble[1], Kurt Knorpp[1], Jared Zerbe[1]
[1]Rambus Inc, Sunnyvale, CA 94089 [2]University of Alberta, Edmonton, Canada
mhossain@rambus.com

Abstract

A fast lock DLL based 800Mb/s to 3.2 Gb/s burst mode memory interface is implemented. The DLL employs a two-step TDC during power up from 0mW to lock within 3 cycles with residual error < 33 mUI. Following initial lock, the DLL operates closed-loop to compensate for V,T drift consuming 6mW @ 1.6GHz. In addition the DLL filters high frequency input jitter and corrects 20% DCD without additional correction.

Introduction

Improving energy efficiency requires agile memory interfaces with fast wakeup to provide adjustable memory bandwidth based on real-time demand. One simple implementation is shown in Fig. 1 where a column or row access signal is used as a trigger to wake up the link from a 0mW idle mode. To avoid adding to the read latency, wake up time should be minimized. Since existing DLL and DCC solutions are not suitable for such burst mode applications, LPDDR standards avoid using DLLs. Dependence solely on matching and removal of all DLLs results in reduced timing margin and limited maximum data rate. Since in DDR specifications the DLL is 'ON' during both active and idle mode, energy efficiency can be quite poor. The fast lock DLL and DCC solution described in this work can powered up and phase locked within 12ns while idle mode power can be reduced to zero without sacrificing performance.

Fast Lock DLL Architecture

The main purpose of the DLL in a DRAM device is to compensate for clock distribution delay such that output data stream remains phase locked to the input reference clock. To accomplish this, skew provided by the phase mixer or DCDL should complement the varying clock buffer delay to complete a full clock cycle. In the conventional approach, a replica clock buffer is inserted in the feedback path of a DLL. This feedback architecture provides PVT tolerance but takes 100+ ns to settle and therefore not suitable for the burst mode applications [1]. A faster lock time can be achieved with synchronous mirror delay (SMD) [2] structure where clock distribution delay is captured as digital code using a TDC. This code is then used to select the correct DCDL output. Ideally the TDC code can be generated from single reference edge and achieve faster locking. However, this architecture is not suitable for continuous mode of operation.

To combine fast locking and lower power in active mode, we introduce a hybrid approach that allows two modes of operation: initial fast lock mode and continuous active tracking mode (Fig. 2a). After a fast bias power up sequence [3], the DLL starts in fast lock mode. First, the TDC measures the buffer delay as a 6 bit digital code within 3 reference clock cycles. The code word is then applied to the ILO, selecting the input injection point in order to provide the appropriate amount of phase shift to complement the buffer skew, $T_{ILO} = T_{cycle} - T_{Buffer}$. Once the TDC code is generated and the ILO is running, the DLL is switched form 'fast lock' to 'continuous active tracking' mode. In this mode a simple bang-bang phase-detector is used instead of a high-power TDC. The early/late outputs of the bang-bang PD are digitally accumulated and filtered to increment/decrement a 6 bit phase code similar to the original TDC code. Note that active tracking mode is initiated from the 6 bit phase code generated by the TDC, hence smooth handoff is ensured. As the phase increment is restricted to $T_{CYC}/64$ in active tracking mode a 'glitch-less' continuous clock is guaranteed. At the same time bandwidth is sufficient to compensate for any V,T drift that happens during long read or write operations. Once the TDC code is generated, only active tracking loop components – the ILO, replica buffer, BBPD and loop filter are kept on, consuming only 6 mW at 1.6 GHz. During fast-lock the system consumes ~24 mW), but is only on for 10 ns. Therefore, this architecture achieves both fast locking and low average power and compensates for process, voltage and temperature drift during long read/write operations.

Implementation and Measured Results

The coarse TDC is implemented as an eight stage delay line with resolution of 1/16 clock cycle. After sampling a reference clock edge at the replica buffer output, the coarse TDC selects two adjacent edges that bracket the reference edge. These adjacent edges are then phase mixed to achieve fine TDC resolution of 1/64 clock cycle The active tracking loop is implemented similar to conventional digital DLL except for the use of the ILO as phase shifter rather than a phase mixer. This allows for direct complementary mapping of the phase code, as well as filtering of high frequency input and dither jitter (Fig. 4b). DCD caused by the local clock distribution buffers can be calibrated and the correction code can be stored. However, as the reference clock DCD is unknown it cannot be calibrated and must be corrected. Since closed loop correction consumes time and area, we relied on the ILO to filter out DCD. The ILO was effective in removing +/-10% DCD. Although the DLL takes only 3 cycles to lock, link wakeup time is longer. The TDC code generation is gated by replica buffer delay and even after the code is applied the locked clock phase only appears at the pad the distribution buffer delay. Therefore, from DLL enable edge to a phase locked DQs is delayed by $2T_{buffer} + 3T_{ref} = \sim 10$ to 12 ns (Fig. 3). Jitter performance of the ADDLL (31ps @ 3.2Gb/s) is comparable to analog implementations. The benefit of active tracking is visible in link timing margin in the presence of supply noise as shown in Fig 5(b). Note that the sampling phase remains the same as the DLL cancels out skew caused by supply noise. Although the unfiltered part of the PSIJ and self-generated noise of the DLL reduces timing margin compared to noiseless case, there is significant improvement over an open loop solution. The proposed solution is compared to recent DDR-3&4 DLLs in Fig. 6. This solution achieves small area, low power, fast-lock and excellent jitter performance in a single design.

References

[1] H. Lee et al, "A 1.0-ns/1.0-V Delay-Locked Loop with racing mode and countered CAS latency controller for DRAM interfaces," JSSC, vol.47, no.6, pp.1436-1447, June. 2012.

[2] T. Saeki et al, "A 2.5ns clock access, 250 MHz. 256-Mb SDRAM with synchronous mirror delay," JSSC, Vol. 31, pp. 1656-1665, Nov. 1996.

[3] J. Zerbe et al., "A 5.6 Gb/s 2.4 mW/Gb/s Bidirectional Link With 8ns Power-On," Symp. on VLSI Circ., pp. 82-83, June 2011.

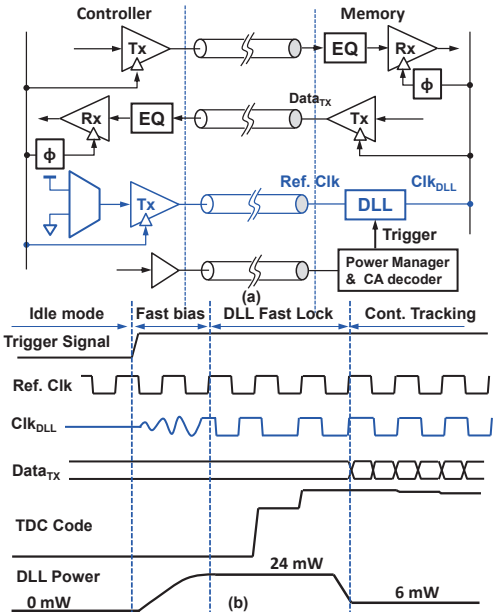


Fig.1(a) Burst mode memory interface. (b) Burst timing sequence and DLL power in different modes

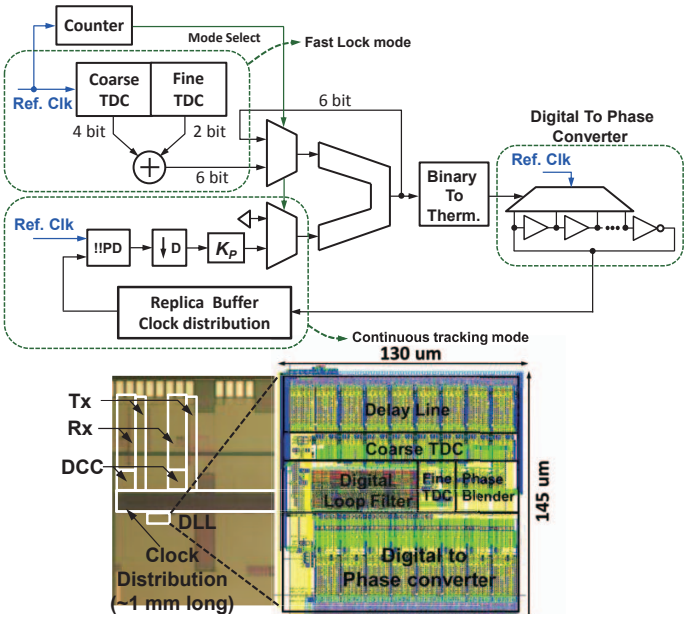


Fig.2 Fast lock DLL architecture and burst mode Link with fast lock DLL

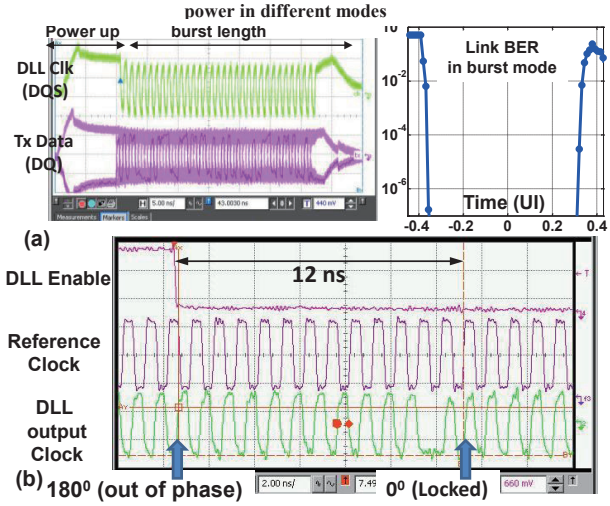


Fig. 3(a) Burst mode transient and Link BER (b) Fast Lock transient

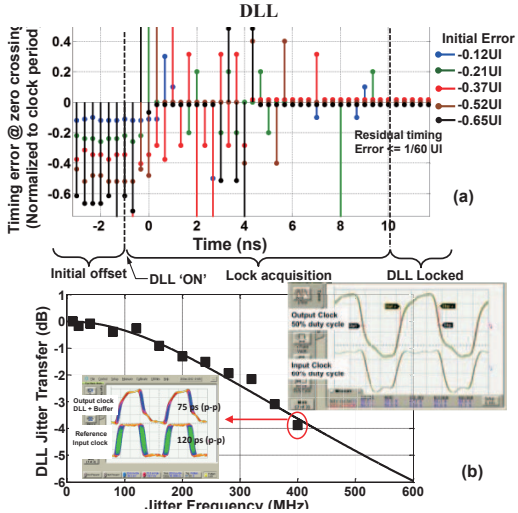
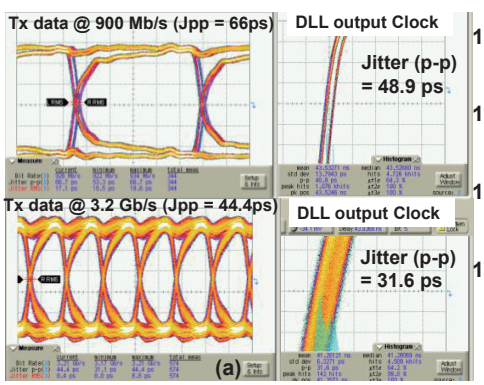


Fig.4 (a) Fast lock DLL Locking behavior for different phase error (b) DLL JTF and DCD correction



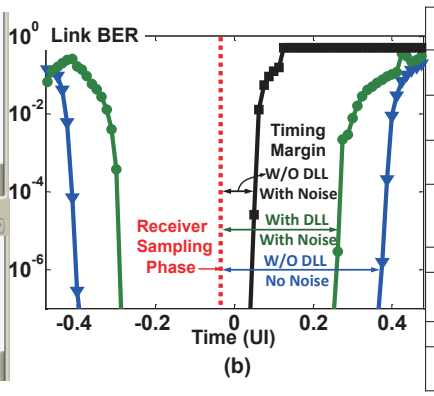


Fig.5(a) DLL Jitter performance @ 900 Mb/s, 3.2 Gb/s (b) Link BER with and w/o DLL in the presence of 20% supply variation

	K. Lee et al JSSC'07	M. Kim CICC'10	H. Lee et al JSSC'12	This Work
DLL Architecture	Feedback (Analog)	SMD (digital)	Feedback (digital)	Hybrid (digital)
Data /Clock Rate	3.2 Gb/s	100 MHz to 1 GHz	1 GHz	800Mb/s To 3.2Gb/s
Lock time (Cycle) Link Wakeup time	1500	3	512	3 12 ns
Idle Power Active Power	------ 80 pJ/Hz (DL+Dist.)	0 64 pJ/Hz (DLL Only)	6.1 pJ/Hz 6.1 pJ/Hz (DLL only)	0 3.75 pJ/Hz (DLL only)
Area	0.3 mm^2	0.21 mm^2	0.10 mm^2	0.05 mm^2
Clock Jitter Tx Data Jitter	31 ps	14 ps (w/o buffer)	75 ps	31.6 ps 44.4 ps
TDC Resolution		12 ps		4.88 ps
Technology	90 nm (DRAM)	0.18 um (SOC)	4x CMOS (DRAM)	40 nm (SOC)

Fig. 6 Performance summary and comparison of proposed DLL with existing DLL solutions