A Fast-lock Mixed-mode DLL Using a 2-b SAR Algorithm

Guang-Kaai Dehng, Student Member, Jyh-Woei Lin and Shen-Iuan Liu, Member, IEEE

Department of Electrical Engineering & Graduate Institute of Electronic Engineering, National Taiwan University
Taipei, Taiwan 10617, R. O. C.

Abstract

In this paper, a fast-lock mixed-mode DLL (MMDLL) is presented. The digital part of the MMDLL utilizes a 2-b SAR algorithm to achieve short lock time compared to the conventional RDLL, CDLL and SARDLL, while the analog part helps to reduce the static phase error and improve the output clock jitter. The measured output clock rms, peak-to-peak jitter and static phase error are 6.6ps, 47ps and 12.4ps, respectively at 100MHz and the power consumption is 15.8mW in the locked state at 2.7V supply voltage. The maximum lock time is 13.5 clock cycles when the static phase error is within 1 LSB (156ps).

Introduction

With the rapid advances in deep-submicron CMOS processes, modern digital systems operated at hundred megahertz have been successfully developed for several years, such as high-speed data links, multi-processor systems, and so forth. Since there are more and more building blocks integrated on the same chip which is guided by the concept of system-on-a-chip (SOC) and system-on-silicon (SOS), the clock-skew problem becomes one of the bottlenecks in realizing high-speed and high-performance digital systems.

Delay-locked loops (DLL's) and phase-locked loops (PLL's) have been widely adopted to solve the clock-skew problem. Generally speaking, if there is no need for frequency multiplication, DLL's are preferred since they are more stable and don't exhibit the jitter accumulation characteristic as PLL's do. DLL's can be classified into analog DLL (ADLL) and digital DLL (DDLL). In ADLL's, since the delay line is adjusted in a continuous manner, the static phase error between input and output clocks is inherently smaller than that in DDLL's. Besides, low-power operations and small chip area are more prone to be achieved in ADLL's [1]. However, due to smaller noise margins, ADLL's are more susceptible to the process variations and less immune from power-supply noise. Contrarily, DDLL's [2]-[6] can provide more robust operations over power-supply noise and process, voltage, temperature and loading effects. In addition, they can exhibit shorter lock time than ADLL's at the expense of unavoidable quantization phase error.

Among the schemes of different DDLL's, the register-controlled DLL (RDLL) [2][3] and counter-controlled DLL (CDLL) [4][5] are two common techniques. The variable delay lines are controlled by a set of digital word and no loop filter components like capacitors are required. In the RDLL, the variable delay line is composed of numerous delay cells with equal delay and the controller is a shift register. In the CDLL, however, the variable delay line is designed in a binary-weighted manner and the controller becomes an up/down counter. If the fine timing resolution is achieved through interpolation, CDLL could save a lot of chip area compared to RDLL. Another figure of merit to evaluate the performance of DDLL's is the lock time. Recently, a successive approximation register-controlled DLL (SARDLL) [6] has been proposed to reduce the lock time. It adopts the binary search algorithm to find out the optimal digital control word promptly and the lock time is effectively reduced compared to other DDLL's. Among the DDLL's addressed above, however, tight synchronization between input and output clocks is usually not provided because the lock-in process is only performed once the system starts up and stops immediately when the digital control word is found out. At this moment, the system behaves like an open loop and the output clock jitter is expected to be worse than that in a closed-loop system, such as ADLL's.

In this paper, a fast-lock mixed-mode DLL (MMDLL) designed in a 0.25-μm CMOS process is presented. It uses a 2-b SAR algorithm, which is an extension from the SAR algorithm, to achieve the fast-lock operation. Besides, the mixed-mode technique is also incorporated to make the MMDLL to be a closed-loop system after lock and improve the output clock jitter.

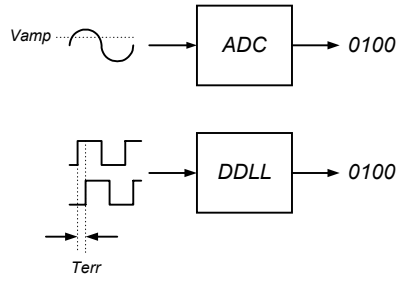

Fig. 1. Operations of ADC and DDLL.

ADC vs. DDLL and the proposed MMDLL

First, let us review the operations of analog-to-digital converters (ADC) and the DDLL. As shown in Fig. 1, the input of the ADC is in the form of analog voltage, *Vamp*, while the input of the DDLL is in the form of timing error, *Terr*. Both of the ADC and DDLL are responsible for the conversion of the analog inputs to digital outputs. Hence, their operations are quite similar. However, the requirements of the throughput in the ADC and DDLL are different. The throughput of the ADC is higher because the conversion process is repeated continuously. Contrarily, it is lower in the DDLL since the conversion process is only performed when the system starts up.

Fig. 2 shows the analogy between the successive approximation register-based ADC (SARADC) and the SARDLL. Both of the operation principles include the data comparison, the SAR control algorithm and the D/A

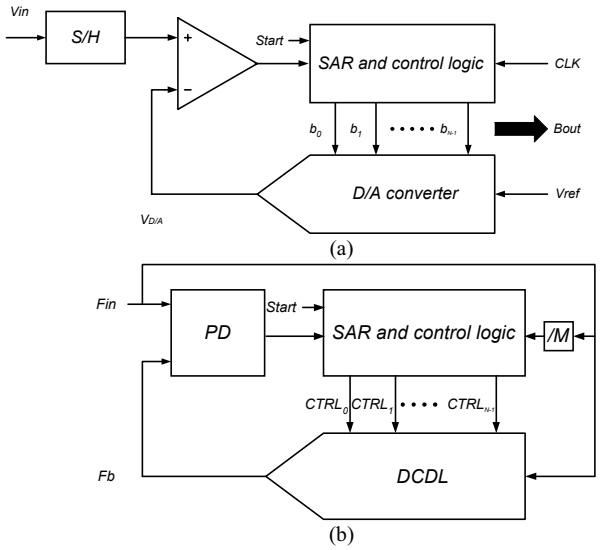

Fig. 2. Analogy between (a) SARADC and (b) SARDLL.

conversion. The role played by the phase detector in the SARDLL is just the same as that played by the comparator in the SARADC. The digital-controlled delay line (DCDL) behaves like the D/A converter in the SARADC. The difference is the sample-and-hold (S/H) circuit in Fig. 2(a) and the frequency divider, /M, in Fig. 2(b). The S/H is needed in the SARADC because it is responsible for the quality of analog input which is to be converted. What is more interesting is the additional divider in Fig. 2(b). Since the D/A conversion performed by the DCDL in the DDLL lies in the time domain, i.e. the timing delay, not the voltage domain, the system response time becomes an important issue, which affects the operation frequency of the SAR control logic in Fig. 2(b). If the SAR control logic is triggered by *Fin* directly, there will be insufficient response time and the comparison results of the phase detector will not be faithfully reflected. This will result in the DDLL's malfunction. Therefore, the additional frequency divider is necessary in the SARDLL. In the SARADC, this problem could be eased because the D/A conversion in Fig. 2(a) is performed in the voltage domain and the system response time is quite short.

Because of the analogy between the ADC and DDLL, the techniques used to implement a high-speed ADC can also be adopted to realize a fast-lock DDLL. The ADC with the highest speed is the flash type since the conversion process is only one step. Based on this concept, one can realize a flash DDLL whose conversion time or in other words, the lock time, is theoretically only one clock cycle. However, the complexity of hardware and large power consumption make it not feasible in high-resolution applications, as in the flash ADC. For example, if the timing resolution is 6-b, the number of sets of DCDL's and phase detectors to implement the flash DDLL would be 64. Although the large power consumption can be reduced by disabling the 63 sets of DCDL's and phase detectors when the system is locked, the mismatch problem between them always exists. This will results in larger and unacceptable static phase error. The SARDLL is an alternative. Its hardware is simpler and the power consumption is smaller. However, the lock-time, which involves several steps, would be larger than that of the flash DDLL. This is the trade-off which exists between the complexity of hardware and the lock time. Intuitively, one can treat the flash DDLL as a N-b SARDLL, where N is the number of bits of the timing resolution. It performs data comparison within only one step while the SARDLL performs data comparison step by step.

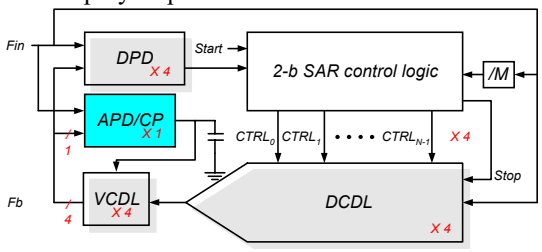

Fig. 3. The proposed MMDLL.

The proposed MMDLL is shown in Fig. 3. It consists of an analog phase detector and charge pump (APD/CP), a 2-b SAR control logic, a frequency divider, a capacitor, 4 digital phase detectors (DPD), 4 DCDL's and 4 voltage-controlled delay lines (VCDL). The system is in the initial state if *Start* is HIGH. When *Start* becomes LOW, the system begins to lock. The digital part of the MMDLL operates first. When the digital control word is found out, the 2-b SAR control logic stops and the analog part of the MMDLL is enabled. At this moment, a *Stop* signal is asserted to disable 3 sets of DPD's, DCDL's and VCDL's, leaving only 1 set active in order to reduce the power consumption. The digital part is dedicated to coarse tuning while the analog part is dedicated to fine tuning. Hence, the lock time is mainly dominated by the digital part. As far as the digital part is concerned, the proposed MMDLL is a compromise between the flash DDLL and SARDLL. It adopts a 2-b SAR algorithm. That is, every 2 bits from the most significant bit (MSB) are grouped together to perform the SAR algorithm. Hence, the complexity of hardware is about 4 times that of the original SARDLL and the lock time is only one half theoretically. The tuning range of VCDL's is about 4 least significant bits (LSB). The analog part is incorporated to reduce the quantization phase error introduced by the DCDL's. Also, it can totally compensate the mismatch between DCDL's if the mismatch is within ± 2 LSB. In addition, the jitter performance can be improved since the analog part makes the MMDLL a closed-loop system. In this work, the timing resolution is set to be 6-b and the division ratio, M, is 4. The initialization process of the MMDLL takes 1.5 clock cycles. Thus, the maximum lock time is $6 \times 4 \div 2 + 1.5 = 13.5$ clock cycles when the static phase error is within 1 LSB, which corresponds to 156ps if the input clock frequency is 100MHz.

Circuit description

A. DCDL/VCDL

Fig. 4 shows the delay cell of the DCDL and VCDL. The digital delay cell is designed in a binary-weighted manner and the interpolation achieves fine timing resolution. In Fig. 4(a), the number beside the MOS capacitors denotes the number of unit delay it provides. The overall DCDL is realized by cascading numerous digital delay cells. Fig. 4(a) is a modified version of that in [6]. The MOS capacitors are evenly divided into two groups, which is different from the original topology.

In this way, the signal path could be more symmetrical than the original one, either the upper path or the lower path. The symmetrical property helps to prevent pulse width distortion and preserve the output clock duty cycle as close as 50%. Fig. 4(b) shows the analog delay cell. Its structure is quite unsophisticated for simplicity. The analog control voltage, *Vctrl*, controls the delay cell through the back gate of the NMOS and the overall VCDL is composed of 4 identical analog delay cells.

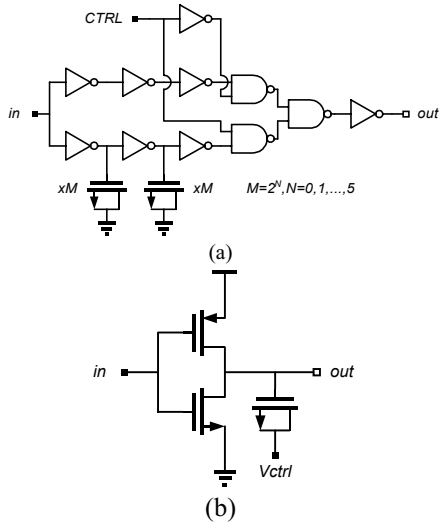

Fig. 4. (a) Digital delay cell. (b) Analog delay cell.

B. DPD/APD

The DPD adopted in this work is the same as the phase comparator in [6]. Two true-single-phase clocking (TSPC) DFF's are used to sample the input clock signal. The digital delay cells with one unit delay form a *lock detecting window*, whose width is exactly one LSB. When the input clock is located outside the window, the *comp* indicates whether the input clock leads the feedback clock or not. Once the input clock enters the window, *LD* is asserted to disable the 2-b SAR control logic immediately. At this moment, the digital part of the MMDLL is finished and the static phase error between the input and output clocks is within one LSB. The APD is a dynamic logic style phase-frequency detector (PFD), which is similar to that in [7]. When the metastable problem occurs in the DPD, the digital control word will deviate from the expected one. If the deviation is within ± 2 LSB, the analog part can always compensate the phase error, which means that the input and output clocks can coincide with each other tightly.

C. *2-b SAR control logic*

The 2-b SAR control logic in the MMDLL is a cell-based design, not a full-custom one. The algorithm is depicted in Fig. 5. It determines the digital control word two bits per step from MSB in a sequential manner. 4 sets of digital words, A_N, B_N, C_N and D_N (N=0~5) helps to set 4 different timing thresholds with equal time difference through the DCDL's. The lock detect (LD) signal is asserted when the input clock enters one of the 4 *lock detecting windows*. If it occurs, the necessary digital control word is found out and the lock-in process stops immediately.

D. Frequency Divider

In the MMDLL, a parameter called *loop delay*, T_{LOOP}, can be defined as the time delay for the input clock to propagate through the DCDL, the VCDL and the DPD. It is given as

$$T_{LOOP} = T_{DCDL} + T_{VCDL} + T_{DPD}$$

where T_{DCDL}, T_{VCDL} and T_{DPD} denote the delay time of the DCDL, VCDL and DPD, respectively. The necessary response time for the MMDLL is the maximum T_{LOOP} plus one clock cycle. Hence, the minimum integer division ratio, M, is $[T_{LOOP}/\ T_{CLK}] + 1$, where [ ] is the Gaussian operator. Typically, $T_{CLK} < T_{LOOP} < 2\ T_{CLK}$. Thus, the minimum M is 3. In this work, M is set to be 4 for convenience since the frequency divider can be realized by simply cascading two toggle flip-flops. If M is set to be 3, the maximum lock time will become 6 × 3 ÷ 2 + 1.5 = 10.5 clock cycles.

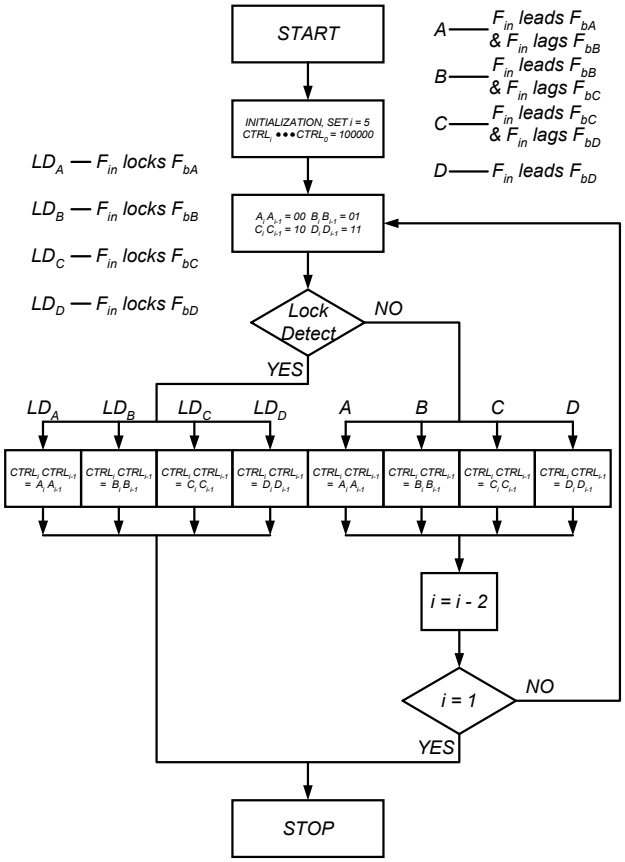

Fig. 5. 2-b SAR algorithm.

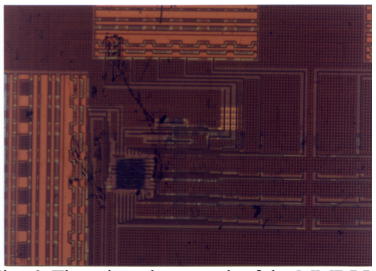

Fig. 6. The microphotograph of the MMDLL.

Experimental results

The proposed MMDLL has been fabricated in a 0.25-μm standard N-well DPQM CMOS process. Fig. 6 shows the microphotograph of the SARDLL. The active area of the chip is 1100 × 670 μm^2. In addition, a similar DDLL without the

analog part of the MMDLL is also designed in the chip for the purpose of performance comparison.

Fig. 7 shows the waveforms of the input and output clocks when the MMDLL is locked (C1 is Fin and C2 is Fb). The input clock frequency is 100MHz and the supply voltage is 2.7V. The measured static phase error of the DDLL is about 130.8ps. However, due to the operation of the analog part, it is greatly reduced to about 12.4ps in the MMDLL. The output duty cycle distortion is caused by the digital output pad provided by the foundry and simulation results confirm our guess. All the digital signals, such as *Start*, *Stop*, and the 6-b control word are recorded by HP16500B Logic Analysis System to show the transient lock-in process and to determine the lock time. Fig. 8 shows the transient lock-in process. According to the measured results, the MMDLL can always achieve lock within 13.5 clock cycles, or in other words, 135ns.

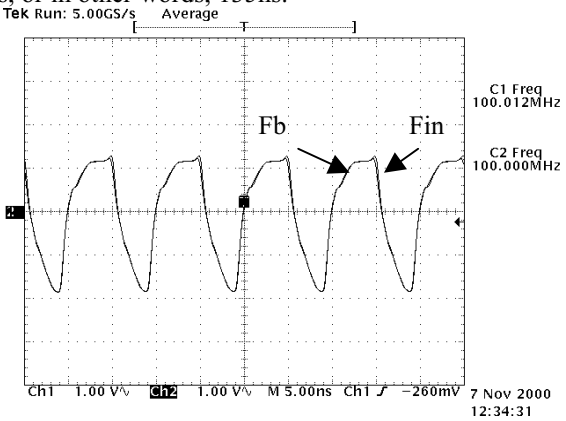

Fig. 7. Input and output clocks of the MMDLL.

The measured rms and peak-to-peak jitters of the DDLL are 9.6ps and 67ps, respectively. However, they are reduced to 6.6 ps and 47 ps in the MMDLL, as shown in Fig. 9. This confirms our expectation that the MMDLL can improve the jitter performance due to the closed-loop operation. The measured power consumption of the MMDLL is 47.2mW in the initial state and 15.8mW in the locked state. The power consumption is reduced because of the power-down operation in the locked state. Table III summaries the performance of the proposed MMDLL.

CONCLUSIONS

In this paper, a fast-lock MMDLL fabricated in a 0.25-μm CMOS process is presented. The MMDLL is composed of digital and analog parts. The digital part utilizes a 2-b SAR algorithm to promptly find out the necessary digital control word and achieve short lock time compared to the conventional RDLL, CDLL and SARDLL. The analog part is inserted to reduce the static phase error and improve the output clock jitter. When the input clock frequency is 100MHz, the measured output clock rms and peak-to-peak jitter are 6.6 ps and 47 ps, respectively. The power consumption is 47.2mW in the initial state and 15.8mW in the locked state at 2.7-V supply voltage. The maximum lock time is 13.5 clock cycles when the static phase error is within 1 LSB. This successfully demonstrates the feasibility of the proposed MMDLL.

ACKNOWLEDGMENT

The authors would like to thank SHARP Co., Japan, for the fabrication of the chip. This work was supported by the National Science Council under Grant NSC89-2215-E-002-024.

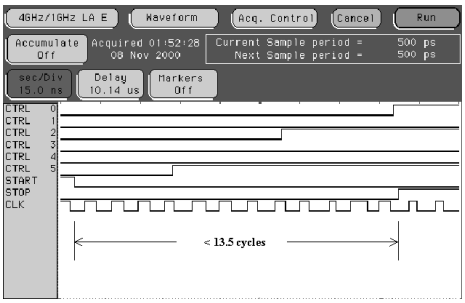

Fig. 8. Transient lock-in process.

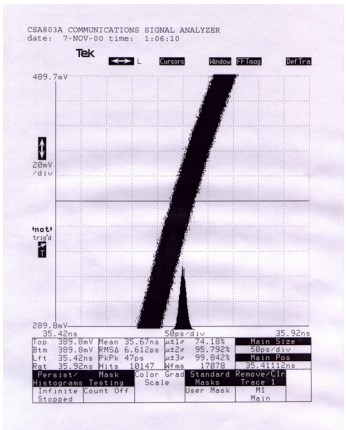

Fig. 9. Output jitter histogram at 100MHz.

Table I Performance summary

Technology	0.25μm, 2-P, 4-M
Supply Voltage	2.7V
Power @ 100MHz	47.2mW (initial), 15.8mW (locked)
Jitter @ 100MHz	47ps (peak-peak), 6.6ps (rms)
Lock Time @ 100MHz	< 135ns (digital)
Static Phase Error	12.4ps

REFERENCES

[1] S. I. Liu, J. H. Lee, and H. W. Tsao, "Low-power clock-deskew buffer for high-speed digital circuits," *IEEE J. Solid-State Circuits*, vol. 34, no. 4, pp. 554-558, Apr. 1999.

[2] A. Hatakeyama, et al., "A 256-Mb SDRAM Using a Register-Controlled Digital DLL," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1728-1734, Nov. 1997.

[3] F. Lin, J. Miller, A. Schoenfeld, M. Ma, and R. J. Baker, "A Register-Controlled Symmetrical DLL for Double-Data-Rate DRAM," *IEEE J. Solid-State Circuits*, vol. 34, no. 4, pp. 565-568, Apr. 1999.

[4] H. Sutoh, K. Yamakoshi and M. Ino, "Circuit Technique for Skew-Free Clock Distribution," *IEEE Custom Integrated Circuits Conf.*, pp. 163-166, 1995.

[5] H. Sutoh and K. Yamakoshi, "A Clock Distribution Technique with an Automatic Skew Compensation Circuit," *IEICE Trans. Electron.*, vol. E81-C, no. 2, pp. 277-283, Feb. 1998.

[6] G. K. Dehng, J. M. Hsu, C. Y. Yang, and S. I. Liu, "Clock-Deskew Buffer Using a SAR-Controlled Delay-Locked Loop," *IEEE J. Solid-State Circuits*, vol. 35, no. 8, pp. 1128-1136, Aug. 2000.

[7] S. Kim, K. Lee, Y. Moon, D. K. Jeong, Y. Choi, and H. K. Lim, "A 960-Mb/s/pin Interface for Skew-Tolerant Bus Using Low Jitter PLL," *IEEE J. Solid-State Circuits*, vol. 32, no. 5, pp. 691-700, May 1997.