All-Digital Delay-Locked Loop/Pulsewidth-Control Loop With Adjustable Duty Cycles

You-Jen Wang, Shao-Ku Kao, and Shen-Iuan Liu, *Senior Member, IEEE*

***Abstract*—An all-digital delay-locked loop (DLL) and an all-digital pulsewidth-control loop (PWCL) with adjustable duty cycles are presented. For the DLL, by using the flash time-to-digital conversion, both the phase alignment and the duty cycle of the output clock are assured in 10 cycles. For the PWCL, the sequential time-to-digital conversion is adopted to reduce the required D-flip-flops and lock within 28 cycles. For both of the proposed circuits, the requirement of the input clock with 50% duty cycle is eliminated.**

The proposed circuits have been fabricated in a 0.35-μm CMOS process. The proposed DLL generates the output clock with the duty cycle of 25%, 50% and 75%, and the operation frequency range is from 140 to 260 MHz. For the proposed PWCL, the duty cycle is adjusted from 30% to 70% in steps of 10%. The operation frequency range is from 400 to 600 MHz.

***Index Terms*—Delay-locked loop (DLL), duty cycle and time-to-digital conversion, pulsewidth control loop.**

I. Introduction

DELAY-LOCKED loops (DLLs) have been widely used to minimize timing skews and jitters of the clock signals. The traditional analog DLLs [1]–[3] generally have better jitter and skew performances, but they need accurate analog and passive devices. It makes the analog DLLs sensitive to the process, voltage, and temperature (PVT) variations. Conversely, the digital DLLs [4]–[9] can be migrated over different processes.

With benefits from scaling CMOS technologies, the digital DLLs have the low supply voltage, the low power dissipation, and the short locked time. However, some problems existed in the conventional digital DLLs. Most of the digital DLLs [4]–[10] require differential clocks or duplicate coarse delay lines to achieve the output clocks of 50% duty cycle. It may increase the power dissipation and active area. During the coarse phase acquisition of a digital DLL, a binary searching algorithm [9] or a time-to-digital conversion [10] is adopted to shorten the locked time. In [9], it becomes an open loop when the acquisition is finished. In [10], additional coarse delay line is required to achieve a short locked time.

Furthermore, some special analog-to-digital/digital-to-analog systems need the clock generators with programmable duty cycles [11], [12]. In order to correct the duty cycle distortion of the clock in multi-stage clock buffers, several analog pulsewidth control loops (PWCLs) [13], [14] have been presented. However, these analog PWCLs require accurate matched analog devices and the capacitors that may occupy a large active area. Owing to the PVT variations, the analog PWCL [13] may become unstable if the loop gain is not properly controlled. In addition, for analog PWCLs [13], [14], the locked time will be long to have the desired duty cycle.

To deal with the above problems, a fast-locked all-digital DLL and an all-digital PWCL with adjustable duty cycles are presented. Both DLL and PWCL generate the acquired duty cycle of the output clock by adjusting delay difference between input signals of the SR latch. In addition, the short locked time and the adjustable duty cycle are achieved. For the proposed DLL, the phase is aligned and the duty cycle of the output clock is programmable (25%, 50%, and 75%). For the PWCL, the duty cycle is adjusted in 30%, 40%, 50%, 60%, and 70%, but the PWCL does not align the phase. There is a tradeoff of speed and hardware between two methods. Although the first method has more D-flip-flops than the second method, it has a faster locked time owing to the flash time-to-digital conversion. In DLL, the unselected NAND gates are turned off to save power. In PWCL, the delay line has to generate the multi-phase clocks and drive the loads. Therefore, the current starving inverters are chosen in a PWCL. The proposed DLL and PWCL will be introduced in Section II and Section III, respectively. The experimental results will be described in Section IV. Finally, conclusions are given in Section V.

II. Proposed All-Digital DLL

The simplified building block of the proposed fast-locked all-digital DLL with adjustable duty cycles is shown in Fig. 1(a). The input clock, Clkin, is propagated through the constant delay cell (CDC), the third 2-to-1 multiplexer (MUX3), and the coarse delay line (CDL). Two clocks with different phases are selected from the CDL by two 32-to-1 multiplexers, MUX1 and MUX2. Then two clocks pass through two fine delay lines (FDLs), respectively. The phase detector (PD) aligns the phase, and the duty cycle calculation circuit adjusts the duty cycle.

The detail architecture of the proposed DLL is shown in Fig. 1(b). It consists of a CDL with 32 delay cells, two 32-to-1 multiplexers, two FDLs, an SR latch, a check circuit, and a phase detector with several control logic circuits. The input period is detected by the time-to-digital conversion, which is accomplished by the CDL, a check circuit and an encoder. To coarsely align the input and output clocks, MUX1 selects a delayed clock among 32 clocks, clk(0)-clk(31). The first FDL (FDL1) reduces the phase error between input and output clocks according to the phase detecting results. The second control circuit (Control_2) receives the phase detecting result of the PD to decide the controlling values of FDL1 stored in the

Manuscript received May 23, 2005; revised February 6, 2006.

The authors are with the Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan 10617, R.O.C. (e-mail: lsi@cc.ee.ntu.edu.tw).

Digital Object Identifier 10.1109/JSSC.2006.874326

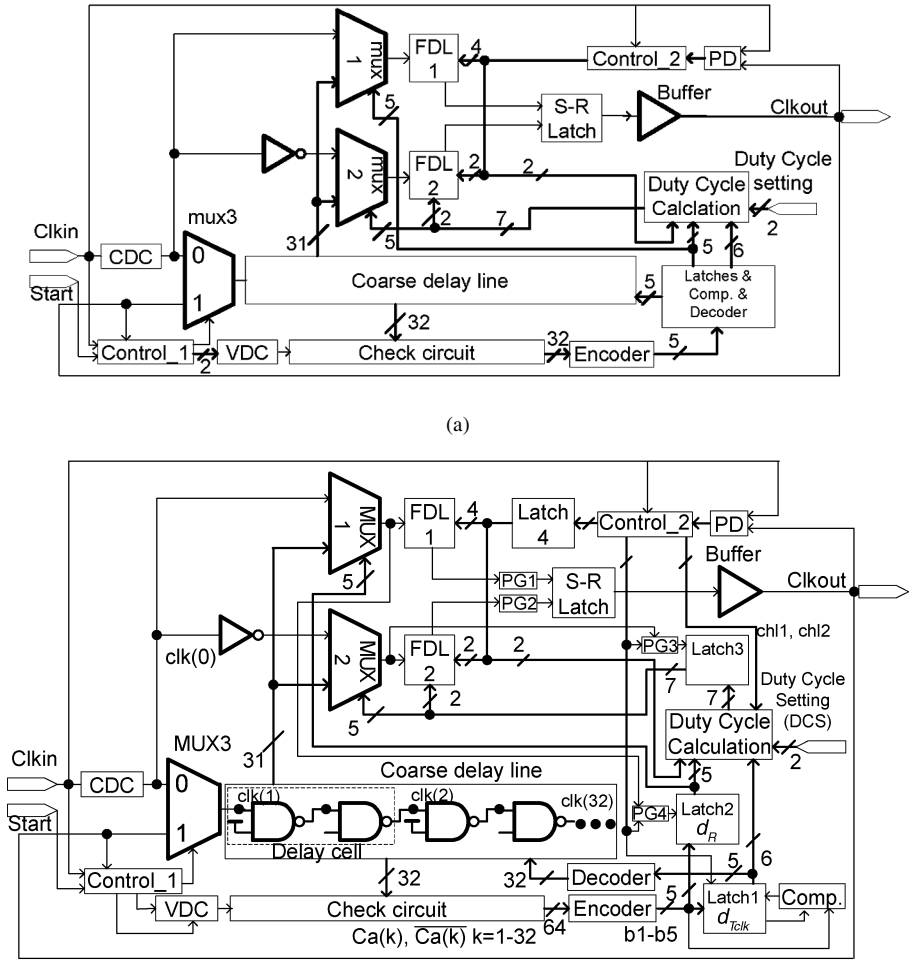


Fig. 1. (a) Simplified diagram for the proposed all-digital DLL with adjustable duty cycles. (b) Proposed all-digital DLL with adjustable duty cycles.

forth latch (latch 4). The Control_2 also controls the last two bits of the second FDL (FDL2).

In order to adjust the duty cycle, the delay difference between inputs of SR latch are set by the calculation. According to the external duty cycle setting codes, the duty cycle calculation circuit will calculate the code to control the MUX2 and two MSBs of FDL2, and the duty cycle is determined. The MUX2 selects a delayed clock among clk(0)-clk(31), and the first two bits of FDL2 reduce the duty cycle error. Then the duty cycle of the output clock will be 25%, 50%, or 75%. The operation of this proposed DLL is divided into four steps. Its timing diagram is shown in Fig. 2(a) and every operation step will be described as follows.

The first step is to convert the period of the input clock into a 6-bit binary code. In the first cycle, the first control circuit (Control_1) will allow the input clock to enter into the CDC, MUX3, and CDL. At the same time, the output of variable delay cell (VDC) is preset to low and each check cell is reset to high. Initially, the delay of VDC is set to equal to that of CDC. After the first cycle, Control_1 will allow input clock to pass through the VDC, and the check circuit captures the logic value of every delay cell in the CDL. In Fig. 2(a), the time interval t1–t0 is equal to the period of input clock, and it is converted to 5-digit code, $d1_{\mathrm{Tclk}}$.

In the second cycle, the control bit of the VDC is set to increase the delay of half a unit CDL cell, and the time-to-digital conversion is repeated. Then the time interval t2–t1 is converted, and it is equal to the input period plus the delay of half a unit CDL cell, and it is converted to 5-digit code, $d2_{\mathrm{Tclk}}$. Although the CDL is only 5 bits, both the measured results are compared to get the sixth bit for duty cycle calculation. Then the accuracy of the duty cycle is improved. If $d2_{\mathrm{Tclk}} > d1_{\mathrm{Tclk}}$, the output of comparator is set to "1". Otherwise, the bit is set to "0". The bit will be stored in the last bit of the first latch (latch 1). The last bit in latch 1 corresponds to the delay of half a unit CDL cell, and the conversion accuracy is improved. Therefore, the final 6-digit code, d_{Tclk}, which denotes the period of the input clock, is stored in latch 1.

Furthermore, the decoder will convert the first five bits of latch 1 to turn off the nonuseful CDL cells, and the power dissipation is reduced. Three cycles are needed in the first step operation, but the third cycle can be overlapped with the second step.

In the second step, the phases between the input and output clocks will be coarsely aligned. The second, third, and fourth

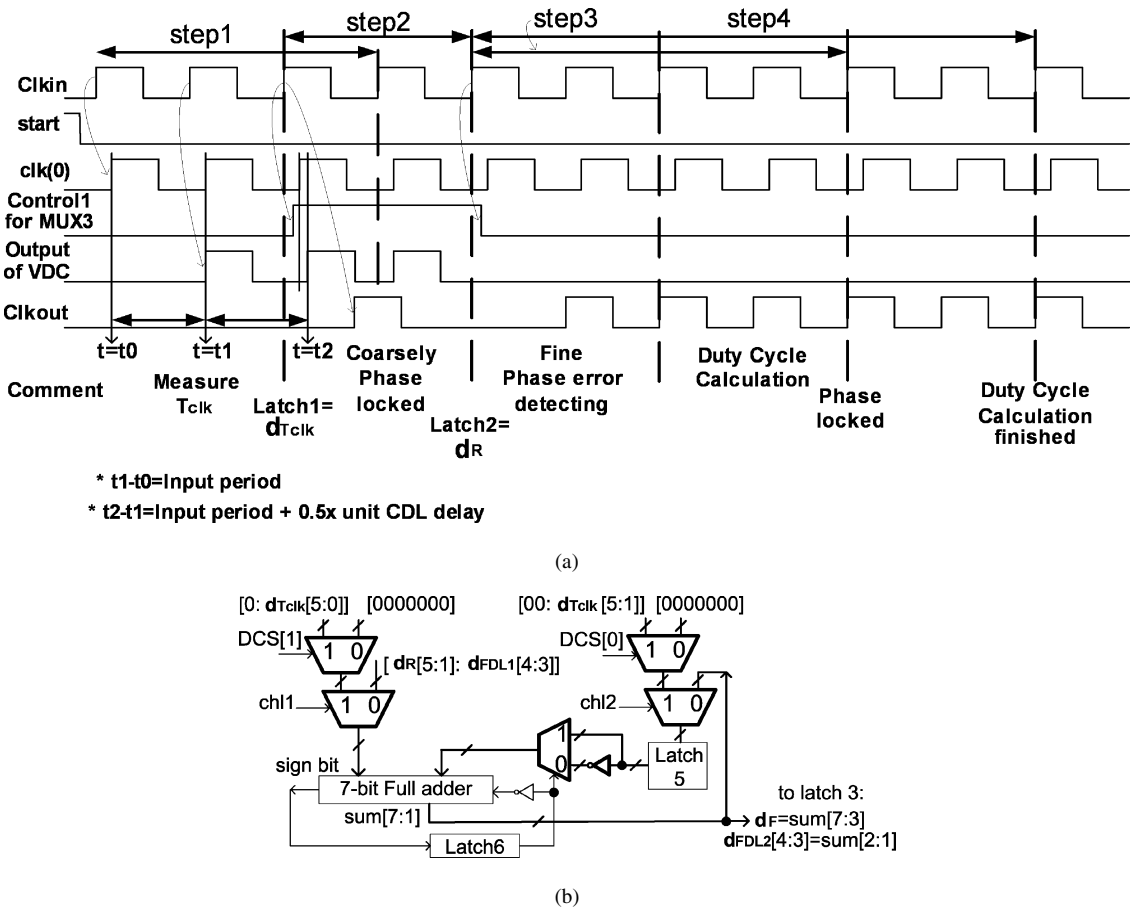


Fig. 2. (a) Timing diagram of the proposed DLL. (b) Duty cycle calculation in the proposed DLL.

latches are preset to zero and the input clock will pass through the CDC and MUX1 to generate the output clock. Control_1 will change the control bit of MUX3 and allow the output clock to enter into the CDL. Then the time-to-digital conversion is operated again, and the result is stored in the second latch (latch 2), named d_R. The delay, t_R, corresponding to the 5-digit code, d_R, is generated from CDL and MUX1 and it can be expressed as

$$t_R = T_{\rm clk} - t_{\rm CDC} - t_{\rm MUX1} - t_{\rm FDL1} - t_{\rm SR\_latch+PG} - t_{\rm buf} \quad (1)$$

where $T_{\rm clk}$ is the period of the input clock, $t_{\rm CDC}$ is the delay time of CDC, $t_{\rm MUX1}$ is the delay time of MUX1, $t_{\rm FDL1}$ is the delay time of the FDL1, $t_{\rm SR\_latch+PG}$ is the delay time of the SR latch with a pulse generator (PG) and $t_{\rm buf}$ is the delay time of the output buffer. The code, d_R, can insert a delay to coarsely align the input and output clocks. However, the phase resolution is poor. Two cycles are needed in the second step operation.

The third step operation is to reduce the static phase error between the input and output clocks. The PD will detect the phase error between the input and output clocks. The binary search algorithm is used to speed up the acquisition. From the most significant bit (MSB) to the least significant bit (LSB), each bit is set to low initially. If the PD output is low, it means the output clock lags the input one. Then the delay time of the FDL1 is decreased, and the control bit is set to high. Otherwise, the control bit is set to low. After the acquisition is completed, the final four binary-weighted bits are stored in latch 4 to control the FDL1. The phase error between input and output clocks is decreased. Two LSBs of the FDL1 are also used to control those of the FDL2. In this step, four cycles are needed.

The last step is to determine the desired duty cycle by controlling the falling edge of the output clock. The duty cyle calculation contains a 7-bit full adder and some logic gates to perform the binary calculation. To have the output clock with 50% duty cycle, the delay difference between two input signals of SR latch must be equal to half a period of the input clock ($T_{\rm clk}$). The required relation is given as

$$\left|(d_F + 0.25 \cdot d_{\rm FDL2,2\_MSB\_bits}) - (d_R + 0.25 \cdot d_{\rm FDL1,2\_MSB\_bits})\right| = 0.5 \cdot d_{\rm Tclk} \quad (2)$$

where $d_{\rm FDL1,2\_MSB\_bits}$ denotes the two MSBs of the FDL1. The generated delay time of the MSB in $d_{\rm FDL1}$ is equal to that of half a unit CDL cell, so the scaling factor is 0.25. If $d_R + 0.25 \cdot d_{\rm FDL1,2\_MSB\_bits} > 0.5 \cdot d_{\rm Tclk}$, the required delay, $d_F + 0.25 \cdot d_{\rm FDL2,2\_MSB\_bits}$, for the falling edge of the output clock is calculated as

$$d_F + 0.25 \cdot d_{\rm FDL2,2\_MSB\_bits} = d_R + 0.25 \cdot d_{\rm FDL1,2\_MSB\_bits} - 0.5 \cdot d_{\rm Tclk}. \quad (3)$$

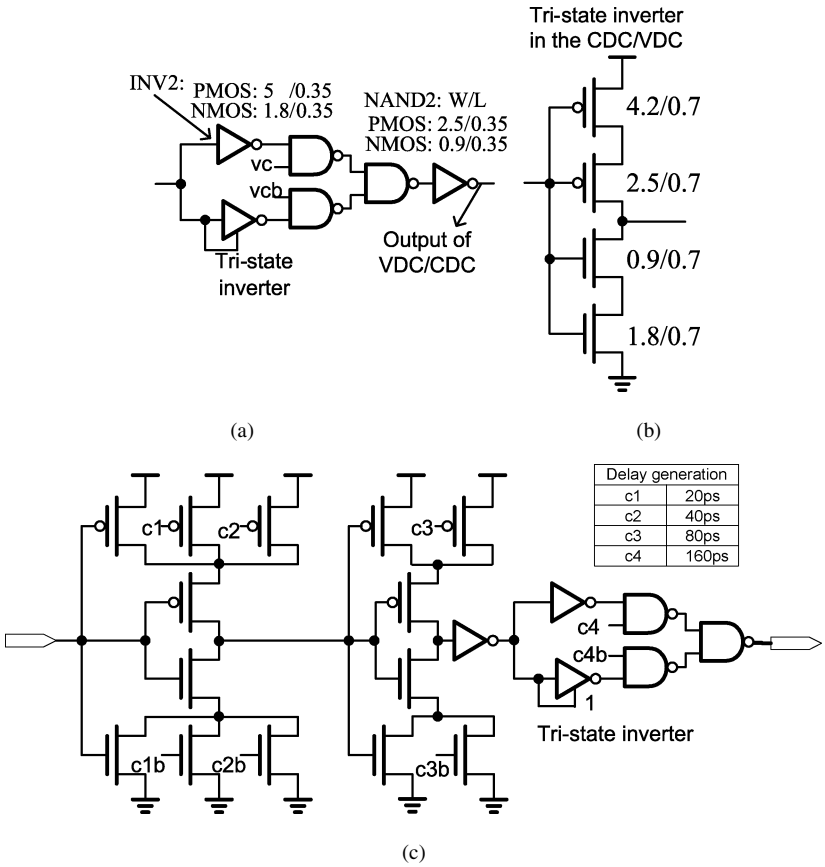


Fig. 3. (a) Constant delay cell and variable delay cell in proposed DLL. (b) Tri-state inverter in the CDC/VDC. (c) Fine delay line.

Right shifting operation actualizes these coefficients, and 2's complement is adopted to realize the subtraction. If $d_R + 0.25 \cdot d_{\mathrm{FDL1,2\_MSB\_bits}} < 0.5 \cdot d_{\mathrm{Tclk}}$, the required delay is

$$d_F + 0.25 \cdot d_{\mathrm{FDL2,2\_MSB\_bits}} = d_R + 0.25 \cdot d_{\mathrm{FDL1,2\_MSB\_bits}} + 0.5 \cdot d_{\mathrm{Tclk}}. \quad (4)$$

Eq. (3) is calculated first by the duty cycle calculation circuit. If the sign bit of this result is low, it represents $d_R + 0.25 \cdot d_{\mathrm{FDL1,2\_MSB\_bits}} > 0.5 \cdot d_{\mathrm{Tclk}}$. Then d_F and $d_{\mathrm{FDL2,2\_MSB\_bits}}$ are obtained. Otherwise, (4) is calculated. The result will be updated and stored in the third latch (latch 3) where the first five MSBs are used to control the MUX2 and the last two LSBs are the two MSBs of the FDL2.

To obtain the different duty cycles, the external 2-bit duty cycle setting (DCS) codes are used to scale d_{Tclk}, the code stored in latch 1. In the above case, the DCS is set to "10" to have the output clock with 50% duty cycle. If the DCS is "01" and "11", the duty cycle of output clock is 25% and 75%, respectively. The circuit of the duty cycle calculation is shown in Fig. 2(b). Two control signals, chl1 and chl2, are generated from Control_2. Initially, chl1, chl2, and latch 6 are set to high. After converting the period of input clock, $(0.5 \cdot \mathrm{DCS}[1] + 0.25 \cdot \mathrm{DCS}[0]) \cdot d_{\mathrm{Tclk}}$ is calculated, and it replaces the term $0.5 \cdot d_{\mathrm{Tclk}}$ in (2), (3) and (4). In Fig. 2(b), the term $[0 : d_{\mathrm{Tclk}}[5 : 0]]$ is equal to $0.5 \cdot d_{\mathrm{Tclk}}$, and $[00 : d_{\mathrm{Tclk}}[5 : 1]]$ means $0.25 \cdot d_{\mathrm{Tclk}}$. Then the signal chl2 is set to low, and the result is stored in the latch 5. When d_R and $d_{\mathrm{FDL1,2\_MSB\_bits}}$ are determined, chl1 and latch 6 are set to low, and the required delay is calculated as

$$d_F + 0.25 \cdot d_{\mathrm{FDL2,2\_MSB\_bits}} = d_R + 0.25 \cdot d_{\mathrm{FDL1,2\_MSB\_bits}} - (0.5 \cdot \mathrm{DCS}[1] + 0.25 \cdot \mathrm{DCS}[0]) \cdot d_{\mathrm{Tclk}} \quad (5)$$

where $d_R + 0.25 \cdot d_{\mathrm{FDL1,2\_MSB\_bits}}$ is equal to $[d_R[5 : 1] : d_{\mathrm{FDL1}}[4 : 3]]$ in Fig. 2(b). If the sign bit of the result is low, the latch 3 will store the sum. Otherwise, the output of latch 6 is set to high, and the result will be updated as

$$d_F + 0.25 \cdot d_{\mathrm{FDL2,2\_MSB\_bits}} = d_R + 0.25 \cdot d_{\mathrm{FDL1,2\_MSB\_bits}} + (0.5 \cdot \mathrm{DCS}[1] + 0.25 \cdot \mathrm{DCS}[0]) \cdot d_{\mathrm{Tclk}}. \quad (6)$$

The error in the duty cycle calculation may be caused by three factors. First, converting d_{Tclk} induces the quantization error. Its maximum error is the delay of half a unit CDL cell, typically 160 ps. The maximum pulsewidth error after the calculation is 160 ps multiplied by the maximum scaling factor of 0.75. It corresponds to the duty cycle error of 3% at 250 MHz. Second, to calculate $0.25 \cdot d_{\mathrm{Tclk}}$ induces the error because the LSB of d_{Tclk} is omitted. If DCS[0] is high, it leads to the duty cycle error of 1% at 250 MHz. Third, the delay mismatch between the setting and resetting paths in SR latch causes the duty cycle error. The corresponding pulsewidth error is typically 100 ps. To compensate the above error, the additional delay is added to the second pulse generator (PG2) [15] to alleviate the delay mismatch.

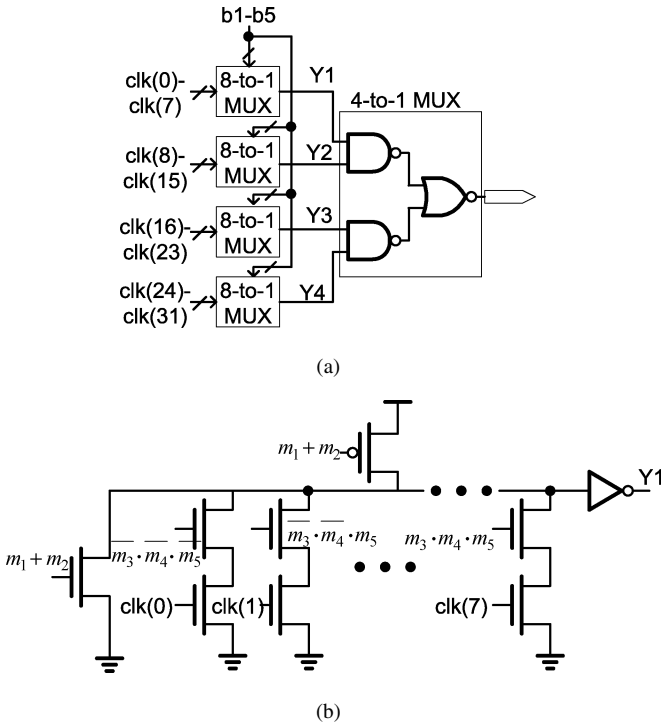


Fig. 4. (a) The 32-to-1 multiplexer. (b) The first 8-to-1 multiplexer.

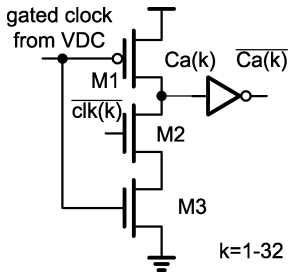
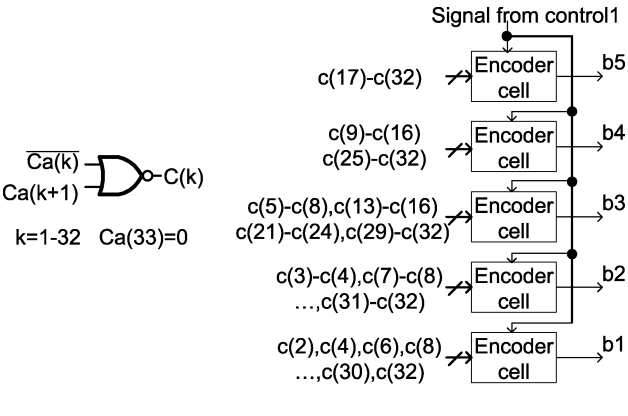


Fig. 5. Check circuit.



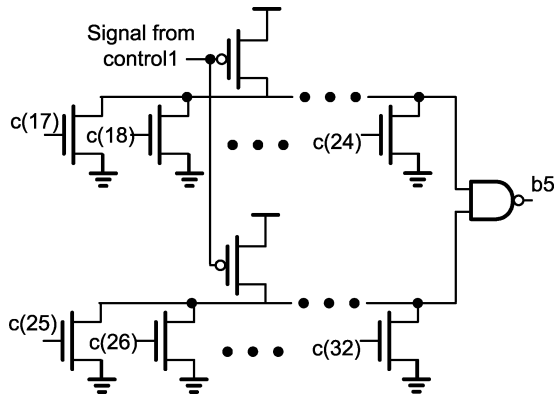


Fig. 6. (a) First stage of the encoder. (b) Second stage of the encoder. (c) Encoder cell.

In FDL1, four bits are used to align the phase difference between input and output clocks. In FDL2, two MSBs are used to adjust the duty cycle, and two LSBs have the minor effect to correct the duty cycle error, which is limited by the time-to-digital conversion. However, two LSBs in FDL2 are needed for two purposes. One is to match the intrinsic delay between FDL1 and FDL2. Additionally, if two LSBs in FDL2 are fixed, the duty cycle changes when two LSBs in FDL1 are updated. To avoid this problem, two LSBs in FDL2 are assigned by those in FDL1.

Six cycles are needed in the last step operation; however, four of them are overlapped with the third step. According to the timing diagram in Fig. 2(a), the phase of this DLL locks in 8 cycles and the correct duty cycle is available in 10 cycles. Not only the phase alignment of the input and output clocks can be achieved, but also the duty cycle of the output clock is programmable.

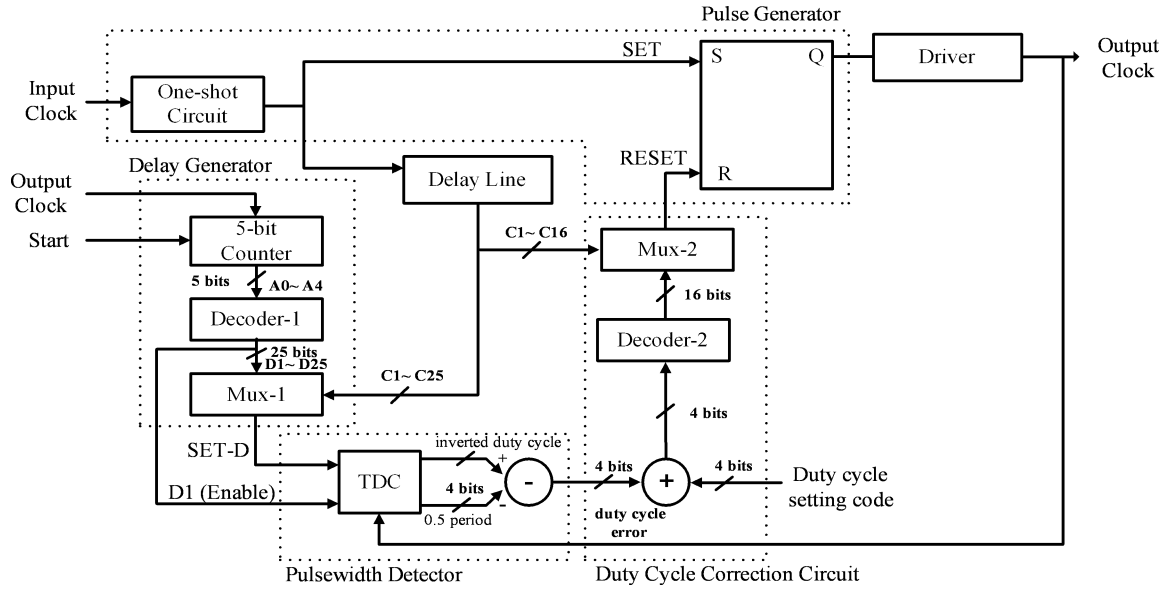
All the building blocks of the proposed DLL in Fig. 1 are described as follows.

A. Coarse Delay Line (CDL)

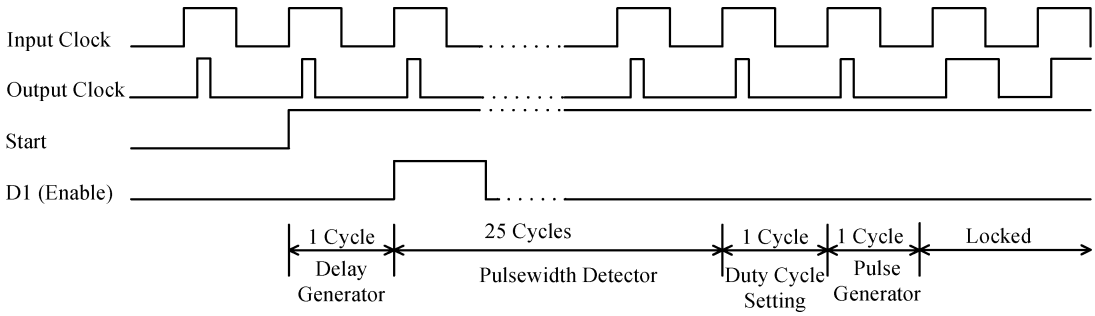
The CDL consists of 32 delay cells, and a unit delay cell is composed of two NAND gates as shown in Fig. 1(b). After the first step operation, the period of the input clock is converted to a binary code, d_{Tclk}. The PVT variations and mismatches among the delay cells affect the performance. So, the size of delay cells is enlarged to alleviate the PVT variations, and the layout of the CDL is folded to improve the matching. The decoder generates the control signals to turn off the nonuseful NAND gates. For a clock with higher frequency, fewer NAND gates are needed, and then turning off more nonuseful NAND gates saves more power consumption.

B. Constant Delay Cell (CDC) and Variable Delay Cell (VDC)

The circuit for the CDC and VDC is shown in Fig. 3(a). It is composed of three NAND gates, two inverters, and a tri-state inverter. The tri-state inverter is adopted rather than an inverter. It is because the tri-state inverter is designed to have a faster transition time than an inverter with a smaller aspect ratio of W/L. The schematic of the tri-state inverter and the size of the device are shown in Fig. 3(b). The CDC is similar to the VDC except that the control bits (vc and vcb) of the CDC are fixed; i.e., $vc = 0$ and $vcb = 1$. vc and vcb control the delay path of the cell. The delay time difference of the two paths equals to the delay time of half a CDL cell. Ideally, the delay of CDC is equal to that of VDC with Control_1 in Fig. 1(b). If the skew is smaller than the delay of one-quarter unit CDL cell, the inaccuracy has a minor effect upon the conversion result.



(a)



(b)

Fig. 7. (a) Proposed all-digital PWCL. (b) Timing diagram of the proposed PWCL.

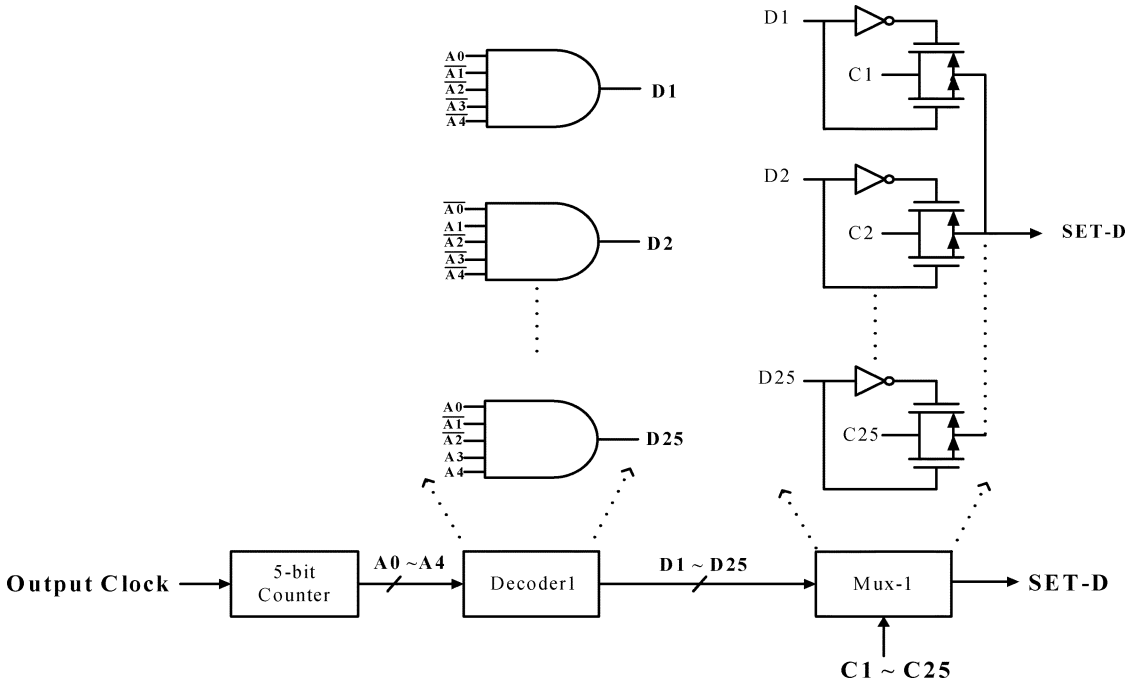
C. Fine Delay Line (FDL)

The FDL is shown in Fig. 3(c), and four binary-weighted bits, c1–c4, control the delay of FDL. The c4 is MSB and c1 is LSB. The corresponding delays generated by c1–c4 are listed in Fig. 3(c). The tuning range of an FDL is 320 ps that should be larger than the delay of a unit delay cell in the CDL. The simulated minimum timing resolution is 20 ps.

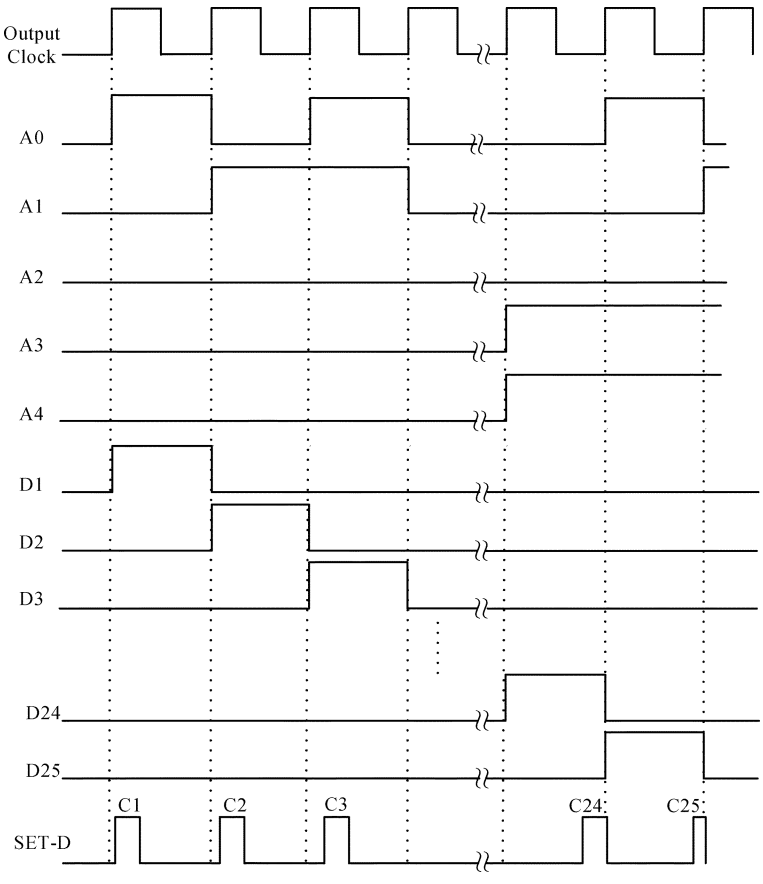
D. Multiplexer

The 32-to-1 multiplexer, as shown in Fig. 4(a), is composed of four 8-to-1 multiplexers and a 4-to-1 multiplexer. The 4-to-1 multiplexer is actualized by two NAND gates and a NOR gate. Pseudo-NMOS circuits realize the 8-to-1 multiplexer as shown in Fig. 4(b). A conventional 32-to-1 multiplexer needs 32 5-input NAND gates to realize the decoder. The decoder of the multi-stage 32-to-1 multiplexer only needs four 2-input NAND gates and eight 3-input NAND gates. So, the multi-stage multiplexer is chosen. In Fig. 4(b), the first two MSBs (m_1 and m_2) control the load PMOS. Because there is only one multiplexer needed to pass the delay clocks from the CDL, three nonuseful multiplexers are turned off to save power consumption. For example, if $(m_1, m_2) = (0, 0)$, the gate of the load pMOS in the first 8-to-1 multiplexer will be low and three other outputs, Y2, Y3, and Y4, are fixed to high. For the nMOS parts, the last three LSBs (m_3, m_4, and m_5) control the upper stage, and the delay clocks from the CDL control the lower stage. For example, if $(m_1, m_2) = (0, 0)$ and $(m_3, m_4, m_5) = (1, 1, 1)$, the outputs, Y2, Y3, and Y4 are fixed to high and the signal clk(7) will pass through the first 8-to-1 multiplexer.

When FDL runs out of the adjusting range, the binary codes used to control the MUX1 and MUX2 need to add or subtract by one. When the pre-selected clock leads the present clock for MUX2, the propagation time of MUX2 (typically 500 ps) is longer than the delay difference between the two clocks (the delay of a unit CDL cell, 320 ps). When the pre-selected clock lags the present clock for MUX2, the sum of the propagation time of MUX2 and the delay difference between the two clocks is smaller than the minimum pulsewidth of the input clock $(30\% \cdot (1/260\ \text{MHz}) = 1154\ \text{ps})$ So, when the MUX2 output is rising to high, both the pre-selected clock and the present clock are assured at high. The third pulse generator (PG3) [15]



(a)



(b)

Fig. 8. (a) Delay generator. (b) Timing diagram of the delay generator.

in Fig. 2(b) is to detect the rising edge of the MUX2 output. When the rising edge is detected, updating the value stored in the latch 3 will have no effect upon the MUX2 output, and glitch or larger jitter can be avoided.

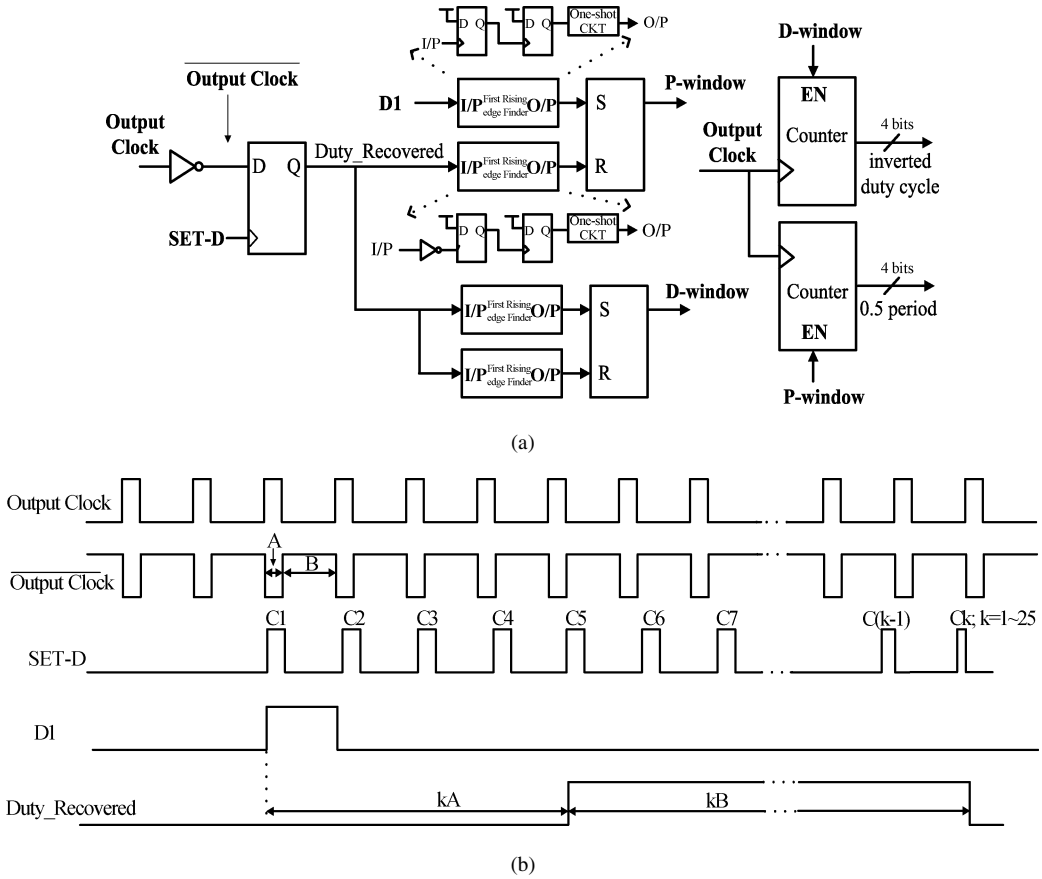


Fig. 9. (a) TDC in the pulsewidth detector. (b) Timing diagram of the pulsewidth detector.

Similarly, the fourth pulse generator (PG4) is used to detect the rising edge of the MUX1 output. The latch 2 is updated when the rising edge is detected.

E. Check Circuit and Encoder

The check circuit captures the outputs of all delay cells in the CDL. It is composed of 32 unit cells to save D-flip-flops compared to the traditional method. A unit cell is shown in Fig. 5. It is composed of three transistors and an inverter. The gated clock from the VDC output controls M1 and M3. The inverse output from the CDL, $\overline{dk(k)}$ and $\mathrm{k} = 1 \sim 32$, controls M2. In the initial state, the output of each cell denoted $Ca(k)$ is preset to high. When the next rising edge goes into each check cell, $Ca(k)$ discharges if $\overline{dk(k)}$ is high, and then $Ca(k)$ is equal to $clk(k)$.

The first stage of the encoder circuits consist of 32 NOR gates as shown in Fig. 6(a). First, the signals $Ca(k)$ from the check circuit are converted to another series of signals $C(k)$. If the $Ca(k)$ is high and $Ca(k+1)$ is low, the signal $Ca(k)$ is high. Otherwise, the signal $C(k)$ is low. Then only one of the signals $C(k)$ is high. The second stage of the encoder is shown in Fig. 6(b). It is composed of 5 encoder cells as shown in Fig. 6(c). It converts the signals, $C(k)$, to a 5-digit binary code, b1–b5. The conversion results are stored in the latches. To save the power dissipation, the decoder is required to turn off the nonuseful delay cells in the CDL.

III. Proposed All-Digital PWCL

The architecture of the proposed all-digital PWCL with adjustable duty cycles is shown in Fig. 7(a). This all-digital PWCL consists of a pulse generator, a delay line, a delay generator, a pulsewidth detector, a duty cycle correction circuit and a driver. Its timing diagram is shown in Fig. 7(b). In PWCL, the duty cycle error is detected by the pulsewidth detector, and it adds or subtracts the duty cycle setting code to controll the delay of the signal "RESET". Then a lookup table is established to obtain the desired output duty cycle. When the power is on, the LSB of the second decoder (Decoder-2) is set to high and the remaining bits are set to low. The one-shot circuit [15] generates the pulse train, "SET", to set the SR latch. The pulse train "SET" will pass through the delay line to generate the delayed pulse trains. One of the delayed pulse trains is selected by the second multiplexer (Mux-2) and the second decoder to reset the SR latch. While the input clock passes through the one-shot circuit, the SR latch and the driver, the output clock will be generated. The duty cycles of the input and output clocks may be different due to the distortion caused by the pulse generator and driver.

When the signal "Start" is active, the delay generator is enabled. The delayed pulse trains allow the pulsewidth detector to convert the inverted duty cycle and the period of the output clock into digital codes. These two converted codes are used to generate the duty cycle error. The calculated duty cycle error and the desired duty cycle setting code are combined in the duty cycle correction circuit. The calculated information is used to

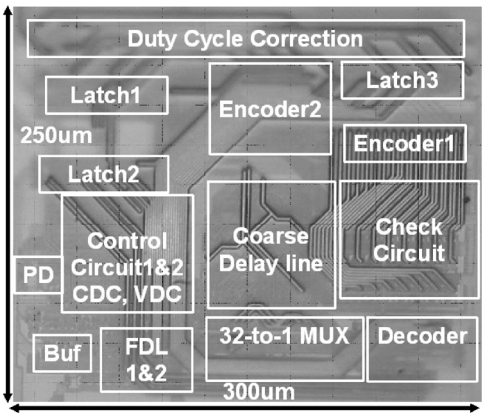


Fig. 10. Photo of the proposed DLL.

reset the falling edge of the output clock without taking care of the duty cycle of the input clock. Thus, the desired duty cycle is obtained. Based on the timing diagram in Fig. 7(b), 28 cycles are needed to generate the output clock with the desired duty cycle. The detail circuit descriptions are given as follows:

A. Pulse Generator and Driver

The pulse generator is composed of a one-shot circuit [15] and an SR latch as shown in Fig. 7(a). This one-shot circuit produces an internal pulse train, "SET", with a fixed duty cycle no matter what the duty cycle of the input clock is. The SR latch produces the output clock with different duty cycles according to the delay difference between "SET" and "RESET". The driver is realized with tapped inverters to drive the load.

B. Delay Line

To cover the operation frequency range, the number of the delay cell is around 25 based on the simulation results. So, the 5-bit counter is used. The delay line consists of 25 delay cells and every delay cell is composed of two current-starving inverters. The dummy buffer is inserted in the last stage of the delay line to match the loading of each stage. The delay time of each delay cell is 120 ps. The output of each delay cell is connected to a buffer for better driving capability. It generates 25 delayed pulse trains, C1–C25, with respect to the internal pulse train "SET".

C. Delay Generator

The delay generator and its timing diagram are shown in Fig. 8(a) and (b). This delay generator is composed of a 5-bit counter, a 5-to-25 decoder, and a 25-to-1 multiplexer. When the signal "Start" in the delay generator is enabled, the 5-bit counter will start to count. The output of the counter is a 5-bit code and the first code is "00001". According to this 5-bit code, the first decoder (Decoder-1) will generate 25 control bits (D1–D25) to select a delayed pulse train to pass through the first multiplexer (Mux-1). The first control bit, D1, enables the pulsewidth detector. The Mux-1 sequentially generates a signal, "SET-D", which is selected from the delayed pulse trains, C1–C25.

D. Pulsewidth Detector

The pulsewidth detector includes two parts: TDC and a 4-bit subtractor. The TDC in the pulsewidth detector is shown in Fig. 9(a). It converts the period and inverted duty cycle of the output clock into digital codes. Its timing diagram is shown in Fig. 9(b). The time-to-digital conversion is performed as follows. After the enable signal (D1) is active, the signal selected from the delayed pulse trains, "SET-D", is sequentially generated from the delay generator to sample the inverse output clock.

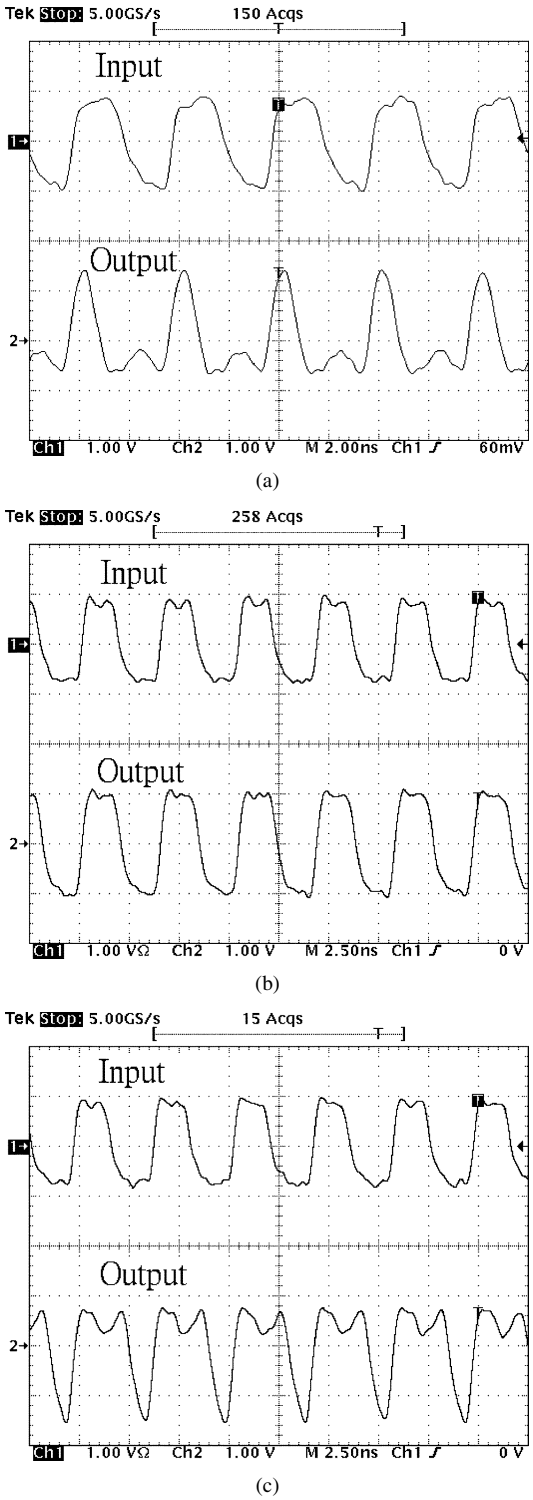


Fig. 11. (a) Input clock of 250 MHz with the duty cycle of 50%. The measured output clocks with the duty cycle of 25%. (b) Input clock of 250 MHz with the duty cycle of 50%. The measured output clocks with the duty cycle of 50%. (c) Input clock of 250 MHz with the duty cycle of 50%. The measured output clocks with the duty cycle of 75%.

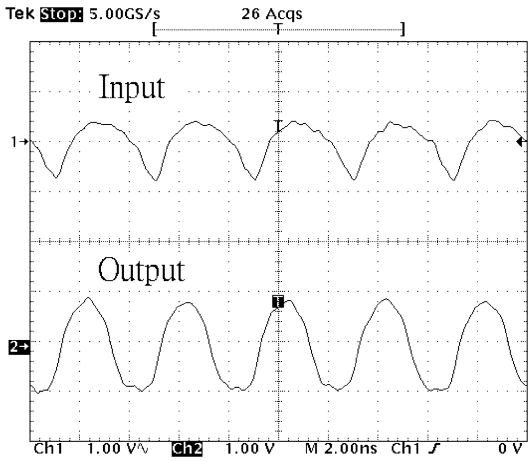


Fig. 12. Measured output clock with the duty cycle of 50% when the input duty cycle is 70%.

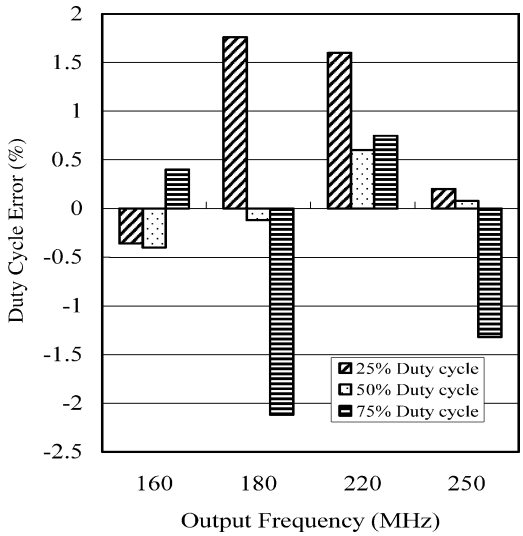


Fig. 13. Measured duty cycle error.

The sampled output signal, "Duty_Recovered", will be generated and its duty cycle is equal to that of the inverse output clock. However, the frequency of "Duty_Recovered" is much slower than that of the output clock. The counters are used to convert the period and duty cycle of "Duty_Recovered" into digital codes.

The counters are activated by two signals, "D-window" and "P-window" as shown in Fig. 9(a). The first-rising-edge finder, the first-falling-edge finder and an SR latch are used to generate "D-window" and "P-window" in Fig. 9(a). "D-window" will enable the counter to count within a time interval when "Duty_Recovered" is high, kB in Fig. 9(b). This time interval is also porportional to the duty cycle of the inverse output clock. So the converted digital code will represent the duty cycle of the inverse output clock. Similary, "P-window" opens a time interval that is porportional to the period of the inverse output clock, kA and kB in Fig. 9(b). The output of the counter with "P-window" is shifted one bit to represent as half a period of the output clock, i.e., 50% duty cycle. The output of the counter with "D-window" represents the duty cycle of the inverse output clock. Finally, the pulsewidth detector subtracts the calculated duty code and half a period code to generate a 4-bit duty cycle error code. This error code denotes how far the current duty cycle away from the duty cycle of 50%. The time-to-digital conversion not only reuses the delay line with the pulse generator, but also saves a lot of D-type flip-flops (DFFs) compared with the traditional method [16].

E. Duty Cycle Correction Circuit

The duty cycle correction circuit will calculate the desired duty cycle. It will combine the duty cycle error code from the pulsewidth detector and the external duty cycle setting code as shown in Fig. 7(a). If the MSB of the duty cycle setting code is zero, subtract the duty cycle setting code from the duty cycle error code. If the MSB of the duty cycle setting code is one, add the duty cycle setting code and the duty cycle error code. The remaining 3 bits are used to control the duty cycle. Then the final code will converted by the second decoder (Decoder-2) to control the seceond mutliplexer (Mux-2) in Fig. 7(a). Then the delayed pulse train, "RESET", will be selected from the delay line to reset the falling edge of the output clock. The final duty cycle can be adjusted from 30% to 70% in steps of 10%.

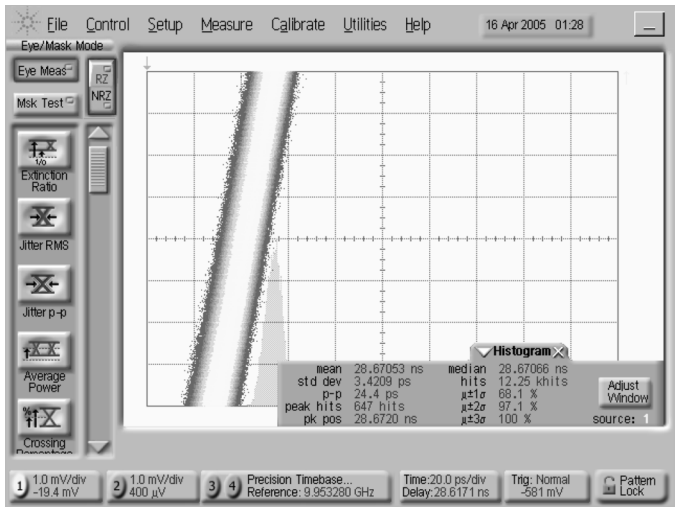


Fig. 14. Measured jitter of the output clock at 250 MHz.

In the PWCL, both the period and duty cycle of output clock are detected and the duty cycle error is calculated. In the DLL, only the period of input clock is detected. Although the PWCL needs an extra look-up table to obtain the desired duty cycle, the distortion caused by the SR latch and buffer is considered when the duty cycle of output clock is detected. In the DLL, the duty cycle calculation does not need a look-up table, but the offset exists.

IV. Experimental Results

The proposed circuits have been fabricated in a two-poly four-metal (2P4M) 0.35-μm CMOS process.

A. DLL

The photo of the DLL is shown in Fig. 10. Its core die area is 0.3×0.25 mm^2. The supply voltage is 3.3 V and the input frequency range operates within 140–260 MHz. When the input duty cycle is 50%, the measured output clocks with the duty cycles of 25%, 50%, and 75% are shown in Fig. 11(a)–(c). Fig. 12 shows the measured output clock with duty cycle of 50% when

TABLE I
PERFORMANCE SUMMARY OF THE PROPOSED DLL AND PWCL

Performance summary	Proposed DLL	Proposed PWCL
Technology	0.35μm 2P4M CMOS	
Supply Voltage	2.4V~3.3V	3.3V
Operation Frequency	140MHz~260MHz	400 ~ 600MHz
Duty cycle of input clock	30% ~ 70%	30% ~ 70%
Duty cycle of output clock	25%, 50%, 75%	30%, 40%, 50%, 60%, 70%
Locked Time	10 cycles	28 cycles
Peak-to-peak jitter	24.4ps(quiet)@250MHz	16.7 ps@500MHz
Core Size	0.3x0.25mm^2	0.65x1.05 mm^2
Power Consumption	9.9mW@3.3V, 250MHz w/o buffers	20mW @ 500MHz

TABLE II
PERFORMANCE COMPARISONS AMONG THE PROPOSED DLL AND OTHERS

	JSSC99[4]	ISSCC00[5]	ISSCC01[6]	JSSC03[7]	VLSI03[8]	JSSC05 [9]	ISSCC05[10]	Proposed DLL
Process	0.4-um CMOS	0.15-um CMOS	0.17-um CMOS	0.13-um CMOS	0.13-um CMOS	0.18-um CMOS	0.25-um CMOS	0.35-um CMOS
Timing resolution	40ps (simulated)	10ps	X	14ps	X	<28ps (simulated)	70.9ps (simulated)	<30ps (simulated)
Max. operating frequency	>667MHz	1.2GHz	250MHz	500MHz	500MHz	700MHz @1.8V	100MHz	260MHz @3.3V
Min. operating frequency	250MHz	0.8GHz	100MHz	66MHz	66MHz	2MHz @1.8V	100MHz	140MHz @3.3V
Peak-to-peak jitter	<250ps	128ps/29ps (quiet)	640ps /200ps (quiet)	X	<25ps (quiet)	17.6ps (quiet) @700MHz	30ps	24.4ps(quiet) @250MHz
Locked time	2.9us (simulated)	X	X	>100 cycles	<150 cycles	32 cycles	8 cycles	10 cycles
VDD	1.7~3.3V	1.6-2.0V	2.1V	1.6V	>1.8V	1.4~2.5V	1.0V	2.4~3.3V
50% duty cycle output	Yes	X	Yes	Yes	Yes	Yes(50 ± 2%)	X	Yes(50 ± 0.6%) (programmable)
Active area	0.96mm^2	0.32mm^2	X	X	0.4641mm^2	0.88mm^2	0.09538 mm^2	0.075mm^2
Power	340mW @400MHz	36mW @1GHz	27mW /0.84mW (standby)	29mW /<0.1mW (standby)	24mW @400MHz	23mW @700MHz	2.43mW	9.9mW @3.3V 250MHz

X: not mentioned

the duty cycle of the input clock is 70%. From Fig. 12, this DLL can generate the output clock of 50% duty cycle even the duty cycle of the input clock is not 50%. The output duty cycle can be programmable. The measured duty cycle errors with respect to different input frequencies are shown in Fig. 13. The measured peak-to-peak jitter of the output clock is 24.4 ps at 250 MHz as shown in Fig. 14. The power consumption at 250 MHz is 9.9 mW. Table I gives the performance summary of this work. Table II gives the performance comparisons among the proposed DLL and others.

B. PWCL

The die photo of PWCL is shown in Fig. 15. The active area of this circuit is 0.65 × 1.05 mm^2. The supply voltage is 3.3 V with input frequency range of 400–600 MHz. The open-drain output buffers are used in the proposed circuit. Fig. 16 shows the output clocks with different duty cycle setting codes. The measured duty cycle of the output clock can be adjusted form 30% to 70% in steps of 10%. Initially, the duty cycle is designed for 10%–90% with a step of 5% and 3 bits are used. The simulation results confirm that. After the chip was measured, the range is only from 30%–70% with a step of 10%. It is believed that the output buffer distorts the duty cycle to result in the mismatch

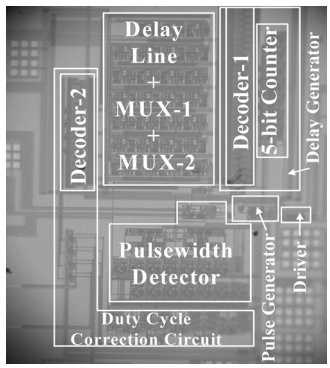


Fig. 15. Die photo of the proposed all-digital PWCL.

between the simulation and measurement results. Fig. 17 shows the simulated and measured duty cycle errors at 500 MHz with different duty cycle setting codes, and the measured duty cycle error is less than ±0.6%. When the desired duty cycle of the output clock is 50%, Fig. 18 shows the simulated and measured duty cycle errors at 500 MHz with respect to different input duty cycles. The measured duty cycle error is less than ±0.64% when the duty cycle of input clock is within 30%–70%. It is concluded that the output clock with 50% duty cycle is obtained

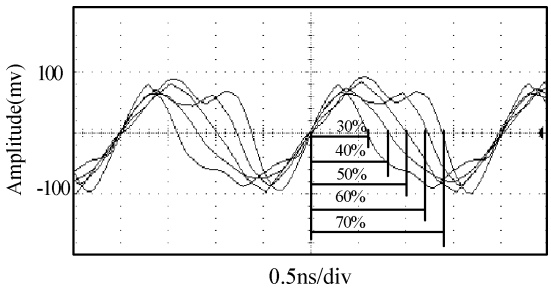


Fig. 16. The ouptut clocks with different duty cycle setting codes.

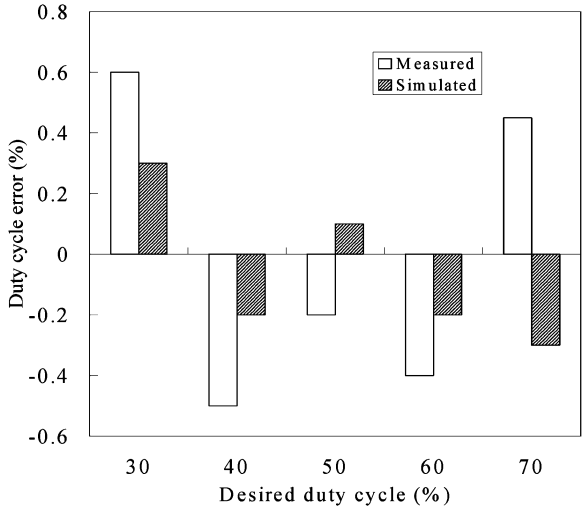


Fig. 17. The simulated and measured duty cycle errors with respect to different desired duty cycle setting codes at 500 MHz.

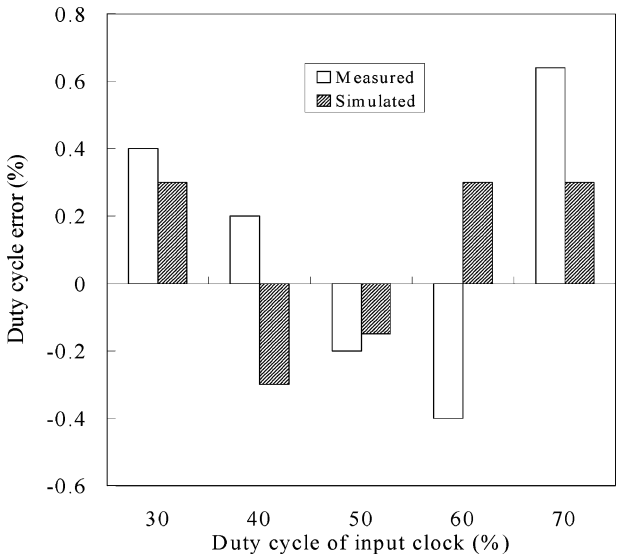


Fig. 18. The simulated and measured errors with respect to different input duty cycles when the desired duty cycle of the output clock is 50% at 500 MHz.

even the duty cycle of input clock is not 50%. The measured peak-to-peak jitter of the output clock at 500 MHz is 16.7 ps. The power consumption is 20 mW. The performance summary of this work is also listed in Table I.

V. Conclusion

In this paper, an all-digital DLL and an all-digital PWCL with adjustable duty cycles are presented. In the proposed all-digital DLL with programmable duty cycle, the design of single coarse delay line reduces the active area, and to turn off the nonuseful delay cell can save the dynamic power dissipation. The correct duty cycle is available in 10 cycles and the phase is aligned in 8 cycles. For the all-digital PWCL with adjustable duty cycles, the delay line is shared between the pulsewidth detector and the pulse generator, so the active area and power can be reduced. The proposed PWCL generates the output duty cycle from 30%–70% in steps of 10%.

Acknowledgment

The authors would like to thank National Chip Implementation Center (CIC), Taiwan, for fabricating this chip. The authors would also like to thank the anonymous reviewers for their valuable comments and suggestions.

References

[1] J. G. Maneatis, "Low-jitter process-independent DLL and PLL based on self-biased techniques," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1723–1732, Nov. 1996.

[2] Y. Moon, J. Choi, K. Lee, D. K. Jeong, and M. K. Kim, "An all-analog multiphase delay-locked loop using a replica delay line for wide-range operation and low-jitter performance," *IEEE J. Solid-State Circuits*, vol. 35, pp. 377–384, Mar. 2000.

[3] H. H. Chang, J. W. Lin, C. Y. Yang, and S. I. Liu, "A wide-range delay-locked loop with a fixed latency of one clock cycle," *IEEE J. Solid-State Circuits*, vol. 37, pp. 1021–1027, Aug. 2002.

[4] B. W. Garlepp, K. S. Donnelly, J. Kim, P. S. Chau, J. L. Zerbe, C. Huang, C. V. Tran, C. L. Portmann, D. Stark, Y.-F. Chan, T. H. Lee, and M. A. Horowitz, "A portable digital DLL for high-speed CMOS interface circuits," *IEEE J. Solid-State Circuits*, vol. 34, no. 5, pp. 632–644, May 1999.

[5] K. Minami, M. Mizuno, H. Yamaguchi, T. Nakano, Y. Matsushima, Y. Sumi, T. Sato, H. Yamashida, and M. Yamashina, "A 1 GHz portable digital delay-locked loop with infinite phase capture ranges," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2000, vol. 469, pp. 350–351.

[6] J.-B. Lee, K.-H. Kim, C. Yoo, S. Lee, O.-G. Na, C.-Y. Lee, H.-Y. Song, J.-S. Lee, Z.-H. Lee, K.-W. Yeom, H.-J. Chung, I.-W. Seo, M.-S. Chae, Y.-H. Choi, and S.-I. Cho, "Digitally-controlled DLL and I/O circuits for 500 Mb/s/pin x16 DDR SDRAM," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2001, vol. 431, pp. 68–69.

[7] T. Matano, Y. Takai, T. Takahashi, Y. Sakito, I. Fujii, Y. Takaishi, H. Fujisawa, S. Kubouchi, S. Narui, K. Arai, M. Morino, M. Nakamura, S. Miyatake, T. Sekiguchi, and K. Koyama, "A 1-Gb/s/pin 512-Mb DDRII SDRAM using a digital DLL and a slew-rate-controlled output buffer," *IEEE J. Solid-State Circuits*, vol. 38, no. 5, pp. 762–768, May 2003.

[8] J.-T. Kwak, C.-K. Kwon, K.-W. Kim, S.-H. Lee, and J.-S. Kih, "Low cost high performance register-controlled digital DLL for 1 Gbps x32 DDR SDRAM," in *Symp. VLSI Circuits Dig. Tech. Papers*, 2003, pp. 283–284.

[9] H. H. Chang and S. I. Liu, "A wide-range and fast-locking all-digital cycle-controlled delay-locked loop," *IEEE J. Solid-State Circuits*, vol. 40, no. 3, pp. 661–670, Mar. 2005.

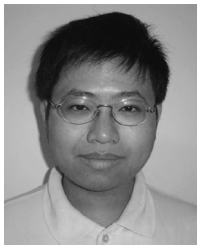
[10] J. S. Wang, Y. W. Wang, C. H. Chen, and Y. C. Liu, "An ultra-low-power fast-lock-in small-jitter all-digital DLL," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2005, pp. 422–433.

[11] S. Karthikeyan, "Clock duty cycle adjuster circuit for switched-capacitor circuits," *Electron. Lett.*, vol. 38, pp. 1008–1009, Aug. 2002.

[12] G. Manganaro, S. U. Kwak, and A. R. Bugeja, "A dual 10-b 200-MSPS pipelined D/A converter with DLL-based clock synthesizer," *IEEE J. Solid-State Circuits*, vol. 39, no. 11, pp. 1829–1838, Nov. 2004.

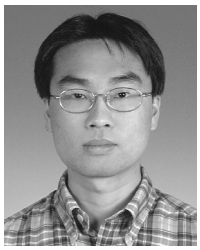
[13] F. Mu and C. Svensson, "Pulsewidth control loop in high-speed CMOS clock buffers," *IEEE J. Solid-State Circuits*, vol. 35, no. 2, pp. 134–141, Feb. 2000.

[14] S. R. Han and S. I. Liu, "A 500-MHz–1.25-GHz fast-locking pulsewidth control loop with presettable duty cycle," *IEEE J. Solid-State Circuits*, vol. 39, no. 3, pp. 463–468, Mar. 2004.
[15] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2003.
[16] M. Mota and J. Christiansen, "A four-channel self-calibrating high-resolution time to digital converter," in *Proc. IEEE Int. Conf. Electronics, Circuits, and Systems*, 1998, pp. 409–412.



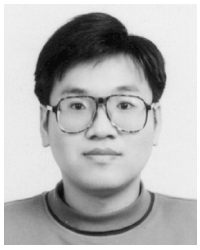
You-Jen Wang was born in Taipei County, Taiwan, R.O.C., on May 2, 1982. He received the B.S. degree in electrical engineering from National Taiwan University, Taipei, in 2004. He is currently working toward the Ph.D. degree in electrical engineering at National Taiwan University.

His research interests are PLL and DLL.



Shao-Ku Kao was born in Taipei, Taiwan, R.O.C., in 1974. He received the B.S. degree from the University of Canterbury, New Zealand, in 1997 and the M.S. degree in electrical engineering from National Taiwan University, Taipei, in 2002. He is currently working toward the Ph.D. degree at National Taiwan University.

His research interests include phase-locked loops, duty-cycle corrector circuits, and mixed-mode clock generators.



Shen-Iuan Liu (S'88–M'93–SM'03) was born in Keelung, Taiwan, R.O.C., in 1965. He received the B.S. and Ph.D. degrees in electrical engineering from National Taiwan University (NTU), Taipei, Taiwan, in 1987 and 1991, respectively.

During 1991–1993, he served as a Second Lieutenant in the Chinese Air Force. During 1991–1994, he was an Associate Professor in the Department of Electronic Engineering of the National Taiwan Institute of Technology. He joined the Department of Electrical Engineering, NTU, Taipei, in 1994, and he has been a Professor there since 1998. His research interests are in analog and digital integrated circuits and systems.

Dr. Liu has served as chair of IEEE SSCS Taipei Chapter since 2004. He has served as general chair for the 15th VLSI Design/CAD Symposium, Taiwan, 2004, and as Program Co-chair for the Fourth IEEE Asia-Pacific Conference on Advanced System Integrated Circuits, Japan, 2004. He was the recipient of the Engineering Paper Award from the Chinese Institute of Engineers, 2003, the Young Professor Teaching Award from MXIC Inc., the Research Achievement Award from NTU, and the Outstanding Research Award from National Science Council, 2004. He served as a technical program committee member for A-SSCC and ISSCC in 2005. In 2006, he became an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS. He is a member of IEICE.