Multifrequency Zero-Jitter Delay-Locked Loop

Avner Efendovich, Yachin Afek, Coby Sella, and Zeev Bikowsky

Abstract—**The approach of an all-digital phase locked loop is used in this delay-locked loop circuit. This design is designated to a system with two processing units, a master CPU and a slave system chip, that share the same bus. It allows maximum utilization of the bus, as the minimal skew between the clocks of the two components significantly reduces idle periods, and also set-up and hold times. Changes in the operating frequency are possible, without falling out of synchronization. Due to the special lead-lag phase detector, the jitter of the clock is zero, when the loop is locked, under any working conditions.**

I. Introduction

THE improved performance of computing IC's, and the growing trend to include several computing devices on the same board, present a challenge with respect to synchronizing the time frames of all the components. In order to maximize the performance of a CPU, operating together with a coprocessor or system chip, clocks must be generated inside the slave components. These clocks must be closely synchronized with the master clock.

When the operation of all components in the system should be highly synchronized, i.e., the maximum skew between the internally generated clocks of all the components should not exceed a couple of nanoseconds, it is not enough to feed the reference clock of the system to all the components. This is because different chips may have different manufacturing parameters. Therefore, there may be large differences in the phases of their generated clocks.

The most popular solution to the synchronization problem is a PLL. At first, analog PLL's were used, but with the improved performance of digital devices, there is a strong tendency to use digital PLL's. In many existing DPLL's, the only component that is digital is the phase detector (PD), while other components, such as the voltage controlled oscillator (VCO) and loop filter (LF), are analog [2], with all the problems inherent in an analog device.

This paper presents a pseudo-PLL, which eliminates the problem of frequency stabilization. Using the same crystal oscillator for both the CPU and the coprocessor, the loop is used only for aligning the phase difference between them; thus, the term delay-locked loop (DLL) is more suitable. This approach also enables the system to switch frequencies (e.g., for power save mode), without losing the phase lock. This concept has already been reported [3], where the PD is a type-3 (JK-ff) PD [2], and the phase adjustment is made by a voltage controlled delay line (VCDL). Type-3 PD's are sensitive both to frequency and phase, and in the solution presented in [3] special logic is required in order to make the circuit independent of clock frequency. In our design, however, the simpler lead-lag (LL) PD is used [1], specially designed in order to achieve zero jitter, as discussed in Section IV. The phase adjustment is done by selecting consequent taps of a high resolution tapped delay line, like in [4], with the delay of each tap constant, allowing for variations in the manufacturing process. In order to promote or retard the phase, we use a 7-b up/down counter instead of a shift register, which requires a large area. With this bang-bang control mechanism, the phase capture stage may take a very long time. This disadvantage is overcome by performing the capture during the reset period. This requires several parts, including the counter, state machines and frequency divider of the clock generator, to be designed to assure correct values, without using the reset signal.

Manuscript received January 17, 1992; revised September 10, 1993.

The authors are with National Semiconductor (I.C.) Ltd., Herzlya B, 46104, Israel.

IEEE Log Number 9213671.

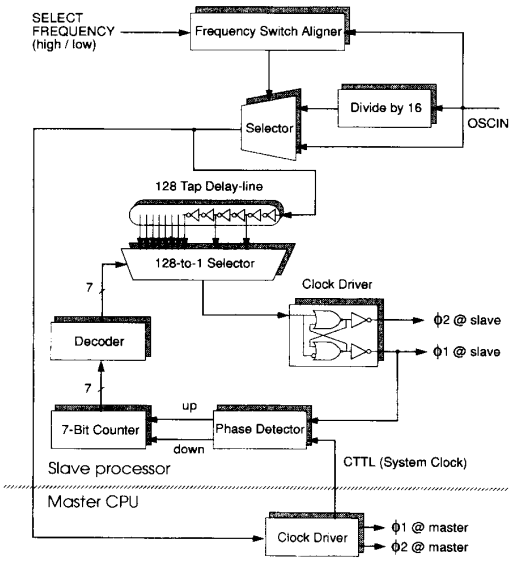

Fig. 1. Block diagram of the DLL.

The DLL described above is implemented in the National Semiconductor NS32FX200 system chip, which is used in FAX applications, to synchronize clocks with the NS32FX16 CPU.

II. Logic Description

Fig. 1 shows a general view of the DLL, located on the coprocessor. The external oscillator OSCIN feeds the divide-by-16 logic. The output of this logic is input to a 2-to-1 selector, which selects between OSCIN and the same clock after the division. The select input of the 2-to-1 selector is synchronized to enable switching between frequencies to occur

TABLE I
PHASE DETECTOR TRUTH TABLE

$\phi 1$ @ CTTL Rising Edge	Delayed $\phi 1$ @ CTTL Rising Edge	"Up"	"Down"
0	0	0	1
0	1	1	0
1	0	0	0
1	1	1	0

only after a *slow frequency* (i.e., the divided-by-16) cycle has ended. This synchronization is performed since we do not want the switching to occur before the current clock has completed a cycle. The output of the 2-to-1 selector is fed to a high resolution tapped delay line which produces 128 clock outputs, each one delayed 1ns relative to the previous one. The remaining logic selects one of these 128 outputs of the tapped delay line, buffers it, divides the frequency by two and generates a two-phase nonoverlapping clock ($\phi 1/\phi 2$).

The output of the 2-to-1 selector is also delayed by constant delay logic, and the resulting delayed signal CCLK serves as output from the coprocessor to the CPU, where the system clock (CTTL) generator is located. The system clock CTTL is directly generated from CCLK by dividing it by 2. The CTTL clock is then reinput to the phase detector of the DLL for phase comparison with the generated clock, $\phi 1$.

As a result of the PLL operation, clock $\phi 1$ is synchronized to system clock CTTL, such that the skew between them is in the range of 0–4 nanoseconds. In a number of microprocessor applications, for example the National Semiconductor microprocessor NS32FX16, clock $\phi 1$ to system clock CTTL skew is in the range of 1–4 nanoseconds. Therefore, the internal clocks of both chips are synchronized to each other accurately enough for both chips to be able to share the same bus.

The phase detector is conventional digital phase detector, that samples the system clock CTTL by clock $\phi 1$. Actually, because of implementation considerations, system clock CTTL is the signal that samples clock $\phi 1$. The phase detector then provides a 2-bit up/down digital signal, that indicate the activity to be performed; i.e., count upwards, count downwards, or remain in the same place.

The up/down output signal of the phase detector is input to a 7-b counter containing the index, which indicates which of the 128 outputs of the delay line should be selected in order to generate $\phi 1$. A 128-to-1 selection is performed, and the selected output is used to generate the two phases $\phi 1/\phi 2$, the first of which is fed back to the phase detector, as described above.

III. FREQUENCY SWITCHING

The delay line is fed by the output of a 2-to-1 selector which selects between two clock signals, OSCIN and OSCIN divided by 16. The same output is the origin of both $\phi 1$ and CTTL. Therefore, when switching to the other frequency, which is only done upon rising edges of the slow frequency clock, smooth passes are guaranteed. Furthermore, if phase lock has been achieved prior to switching, then there should be no indication of a change in phase relation between $\phi 1$ and CTTL after the actual frequency switching has been performed.

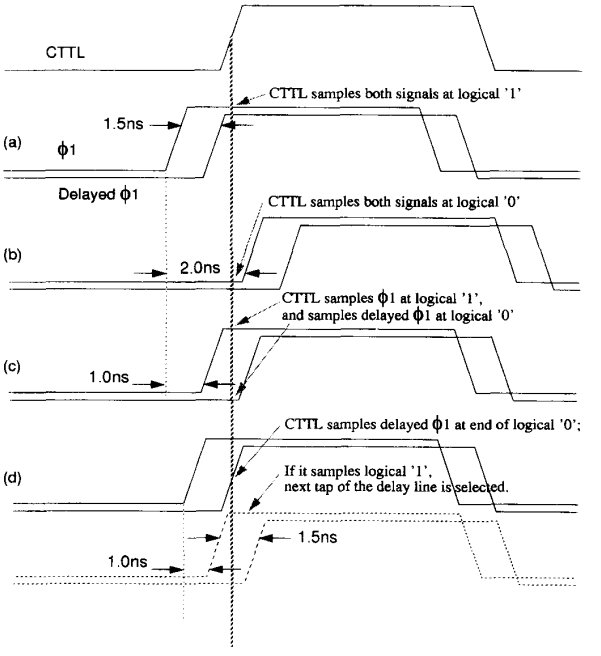

Fig. 2. Phase detector window.

IV. ZERO JITTER

The lead-lag phase detector is constructed of two sets of SR flip-flops, where the system clock CTTL samples two clock signals: $\phi 1$ and delayed $\phi 1$. Table I illustrates the truth table of the phase detector. As may be seen in Table I, the phase detection logic is simple: generally, if the CTTL rising edge samples $\phi 1$ at logic "0", it means that the $\phi 1$ rising edge is *later* than the CTTL rising edge, and the output of the phase detector is "down"; The situation in the opposite direction is similar. Only in the case where the rising edge of CTTL samples $\phi 1$ at logic "1" and the delayed $\phi 1$ at logic "0", does it mean that the $\phi 1$ rising edge is earlier but yet close enough to the CTTL rising edge, which is the desired state.

Fig. 2 illustrates the phase detection window. As may be seen in Fig. 2(d), even if synchronization is achieved at one of the edges of the window, any indication that a shift is needed will bring the synchronization point to another place inside the window; this is because the width of the window (1.5 ns) is greater than the delay of one tap of the delay line (1 ns). Therefore, there will be no jitter around this point. Notice that even without this phase detector window mechanism, if the counter counts up and down endlessly, the $\phi 1$ edge will be shifted back and forth by 1 ns, and therefore the maximal jitter will be ± 0.5 ns.

The sampling clock CTTL and $\phi 1$ are initially asynchronous; furthermore, they sample each other exactly at the

most "dangerous" moment, in terms of mean time between failures (MTBF). Therefore, 4 sampling phases were used in order to assure an MTBF that is large enough at the operating frequency. Using the resolution time constant of our SR flip-flop, $\tau = 1.5$ ns, and the MTBF formula obtained from [5], one may calculate the MTBF of 4 stages at 25 MHz to be 1.99×10^{20} seconds = 6.3×10^{12} years.

V. Implementation Issues

A. Random Initialization

In the illustrated design, the DLL must synchronize $\phi 1$ to the system clock CTTL during the reset period. This means that the reset signal cannot be used by the DLL as input to initialize either counter values, the polarity of $\phi 1$ (after division by two), or any state machine inside the DLL. Therefore, the whole module operates under the constraint of all random-initialized logic starting up at unknown values.

This is accomplished as follows. First, as a constraint, the initial frequency should be the lower one (OSCIN divided-by-16). Second, the counter value, the polarity of $\phi 1$ and state machines are initialized to random values. Third, the initial phase detection is done, yielding an up/down value. Fourth, the counter counts upwards or downwards, producing the selected output index incremented or decremented, respectively. Fifth, the initial frequency is low enough, such that the whole delay line provides a delay smaller than half a clock cycle, the delay between the $\phi 1$ and CTTL rising edges being in any case less than the span of the delay line; therefore there is *one and only one* tap in the delay line, from which the DLL is able to establish phase lock, provided that $\phi 1$ is invoked at the correct polarity. Otherwise, there is *no such tap*.

Therefore, if $\phi 1$ is correctly invoked, the counter proceeds towards phase lock in the correct direction; otherwise, it goes in the opposite direction until the edge of the delay line is reached. In the latter case, dedicated logic inverts the polarity of $\phi 1$ of necessity causing the phase detector to indicate the contrary, and the counter to count the other way. Thus, phase lock is guaranteed.

B. Multiple Clock Synthesis

The module described consists of different clock signals, which although related to each other in terms of frequency, differ a great deal in terms of phase. Therefore, the regular methods of logic synthesis, which are used for one clock system synthesis, are not completely applicable in this case. Thus, we used special implementation methodology and tools, in order to generate and verify a module consisting of signals passed from one time frame to another. Examples of such signal transfers are:

1) The output of the 2-to-1 selector, which selects between clocks with different frequencies.

2) The selected output of the delay line, which may be delayed to a large extent, and is therefore treated as a different clock.

3) The input clock CTTL from the CPU, which has the same frequency as the clock that it samples ($\phi 1$), but initially it has an unknown phase, and finally a somewhat different phase.

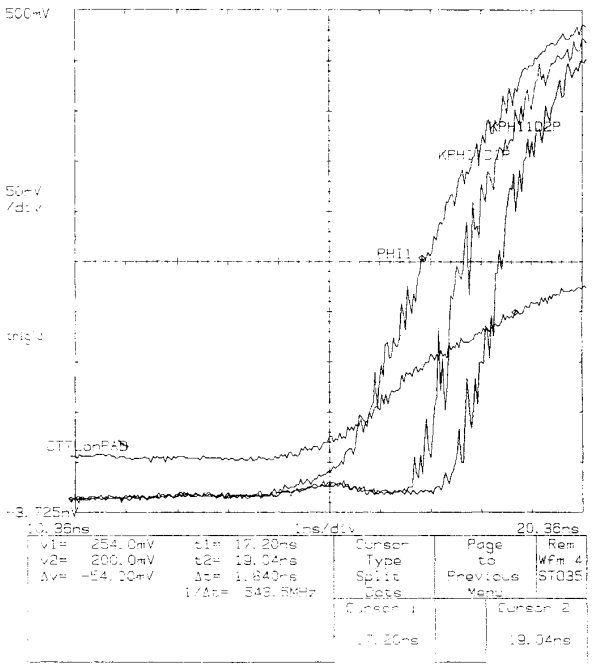

Fig. 3. Skew between 1 (PHI1) and CTTL (CTTLonPAD) clocks.

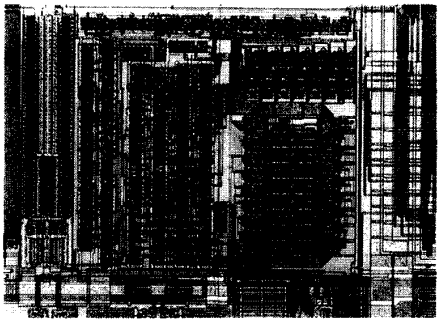

Fig. 4. Photograph of the DLL.

C. High Resolution Delay Line and Selector

The tapped delay line is constructed of 128 CMOS buffers with minimum dimensions ($W/L = 2.6\,\mu/1\,\mu$ for N-ch, $6.0\,\mu/1\,\mu$ for P-ch), dictated by the design rules, and provides a tap delay of 1ns under typical conditions. Experts will appreciate that special care was taken to assure equal delay in all 128 paths of the selector, since this parameter is most important for the stability of the synchronization point.

VI. Results

The module described was manufactured using a double-metal, single poly, twin-well $1\,\mu$ CMOS process. Fig. 3 illustrates the clock waveforms on a scope. PHI1 is the first to

rise. KPHI1D1P is delayed PHI1, and KPHI1D2P is delayed KPHI1D1P. The latter two signals resemble the phase detector window. PHI1 is delayed twice rather than once in order to assure positive skew between PHI1 and CTTL.

Two important parameters should be examined:

1) *Clock skew*—Measured skew at room temperature and VCC = 5 v is Δt = 1.84 ns.

2) *Jitter*—No jitter on PHI1 was seen on the scope. Since the expected jitter, if any, must be quantized by steps of one tap delay (1 ns), which is definitely visible on the scope, the result in Fig. 3 illustrates that the typical jitter is in actual fact zero, as explained above in detail.

Other Results:

1) Power dissipation: 100 mW @ 5v, 25 MHz.

2) Low power supply: the module is functional and gives similar results (skew and jitter) using 3.5 v power supply.

3) Total area: 550,000 μ^2(850 mil^2), including a double 110 pF clock driver.

ACKNOWLEDGMENT

The authors would like to thank A. Intrater, Y. Blecher, E. Cohen, and I. Ben-Haim of the National Semiconductor Design Center in Tel Aviv, Israel, for their great help.

REFERENCES

[1] W. C. Lindsay and C. M. Chie, "Survey of digital phase-locked loops," *Proc. IEEE*, vol. 69, no. 4, pp. 410–431, Apr. 1981.

[2] R. E. Best, *Phase-Locked Loops—Theory, Design and Applications*. New York: McGraw-Hill, 1984.

[3] M. G. Johnson and E. L. Hudson, "A variable delay line PLL for CPU-coprocessor synchronization," *IEEE J. Solid-State Circuits*, vol. 23, no. 5, pp. 1218–1223, Oct. 1988.

[4] D. Anderson *et al.*, "The 68040 32-b monolithic processor," *IEEE J. Solid-State Circuits*, vol. 25, no. 5, pp. 1178–1188, Oct. 1990.

[5] P. A. Stoll, "How to avoid synchronization problems," *VLSI Design*, pp. 56–59, Nov./Dec. 1982.

[6] A. Efendovich *et al.*, "High resolution multifrequency digital phase locked loop," National Semiconductor Corp., U.S. Patent 5 218 314, June 1992.