IC Designs and Applications of the Matched Delay Technique

Wentai Liu, R. Cavin, G. Moyer, M. Clements, T. Schaffer, J. Kang, and J. Medero
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC 27695-7911
e-mail: wentai@eos.ncsu.edu

Abstract

The matched delay technique is a timing methodology for clock and data delay coordination that produces very high performance integrated circuits. In this methodology, control of path propgation delays, combined with careful management of clock skew and data events, is used to obtain maximum performance from digital VLSI circuits. This is a new way of comprehending fundamental performance limits of a given technology and allows the digital designer to exploit intrinsic device matching properties of integrated circuits normally used by the analog designer.

The matched delay technique has been extensively developed at North Carolina State University, and several CMOS prototype chips based on it have been designed, fabricated, and tested. In this paper, we briefly present the underlying theory of the matched delay technique and then four implementation examples: a digital sampler, a digital data generator, a clock/data recovery circuit, and a clock waveform shaper. This is followed by a discussion of technology limitations and practical issues with regard to the matched delay technique.

1 Introduction

The ability to handle very high-speed data is a significant issue in high-performance computing and communications systems. The importance of this issue has increased with the advent of powerful distributed processing systems and fiber-optic communications standards such as SONET, of which even a mid-level (OC-12) implementation requires data rates of 622 Mb/s. Currently, there is a strong trend toward implementing these high-performance systems with a network of processors rather than a single very expensive processor. A fundamental component of such a system is a high-bandwidth network interface for each processor. Such interfaces require multiplexors to combine multiple data streams at the transmitter and demultiplexers to recover the individual streams at the receiver. In addition, circuits are needed to recover the clock and data from such high-speed streams. Typical realizations of these high-speed components require high-speed, well-controlled clocks [?] [?] [?]. Due to the complexities of generating and distributing high-speed clocks, such circuits are difficult to implement. In addition, their data resolutions are limited by absolute gate delays. A new circuit technique is needed to achieve high-speed and fine-resolution operation without the need of high-speed clocks.

One innovative design methodology is the *matched delay technique* [?]. Based on the wave pipelining timing methodology [?], this technique can be used to design circuits whose speed is limited by the difference of matched delay elements rather than their absolute delays. This technique also permits high-speed operation without the need of a high-speed clock. The matched delay technique can be implemented in any technology, however, it is particularly attractive in CMOS since it allows the construction of high-speed circuits that can benefit from CMOS's advantages: low cost, low power, and high density.

This paper describes the matched delay technique and its applications. First, the concept of the matched delay is briefly defined. Then we describe four designs that use the matched delay technique and have already been completed and fabricated by MOSIS.

The matched delay technique was employed in developing a high-performance digital sampler with 1 Gb/s bandwidth and 25ps resolution in a MOSIS 1.2μm CMOS process [?]. The matched delay sampler implements the demultiplexing function of a network interface by performing a serial-to-parallel conversion on an incoming data stream. In addition, this device is a core component in two of the other matched delay designs, the clock/data recovery circuit and the clock waveform shaper.

The matched delay generator performs the inverse function of the sampler and has also been implemented in a MOSIS 1.2μm CMOS process [?]. The generator can produce an arbitrary data stream with an edge placement resolution of 100 ps. This chip includes a 512 bit data memory and extensive delay-locked loop based compensation mechanisms to maintain timing accuracy. The generator has 64 stages and is driven by a 156 MHz clock. Use of the generator and sampler together makes a high-speed CMOS network interface realizable and also forms the basis of a more general high-speed transceiver.

The third design uses the matched delay sampler in the construction of a data and clock recovery circuit for 622 Mbps (OC-12) SONET applications [?]. The design is targeted for a 1.2 μm MOSIS CMOS process. The sampler has 64 stages and is clocked at 156 MHz. Circuit simulations confirm that it reliably achieves 100 ps sampling resolution of the input data and successfully recovers the clock and data at 622 Mbps.

The fourth design uses the sampler as part of a clock waveform shaping circuit that restores the duty cycle of a distorted clock signal [?]. It also synchronizes the clock to a slower external clock. It operates at frequencies up to 450 MHz and restores the duty cycle to within 45 to 55 percent.

2 The Matched Delay Technique

The fundamental concept of the matched delay technique is that the timing resolution is determined by the *difference* between two propagation delay values rather than one absolute value. Both the sampler and generator, the designs which form the core of the others, have a structure based on a pair of tapped delay lines, with a latch at each corresponding pair of taps. The data propagates down one delay chain, and a clock propagates down the other chain, clocking the latch at each stage. The structure can be viewed as consisting of a series of stages, where each stage contains a data delay, a clock delay, and a latch. The difference between the data and clock propagation delays determines the timing interval between adjacent stages, which defines the resolution. In the sampler, the input of the latch is driven by the data delay chain tap, and in the generator, the latch output drives the data tap. described below.

3 Matched Delay Design Implementation Examples

This section describes the application of the matched delay technique to four different chips that have been designed at N.C. State University and fabricated through MOSIS. The circuits described are a digital sampler, a data pattern generator, a data recovery system, and a clock waveform shaper.

3.1 Matched Delay Sampling Technique

This section describes the application of the matched delay technique to high-speed, fine-resolution digital sampling. Fig. **??** shows a block diagram of the basic matched delay sampler. The sampler consists of a chain of stages, where each stage consists of a data latch, a data delay, Δ_D, and a clock delay, $Delta_C$. The data and clock are simultaneously propagated through their respective delay chains. The input to the latch at each stage is driven by the data delay chain tap, and the latch is clocked by the clock delay chain tap, and so each latch samples the local data with the local clock. The effective time interval, Δ_{DC}, between the samples at adjacent stages is equal to the difference between the data and clock delays:

$$\Delta_{DC} = |\Delta_D - \Delta_C| \tag{1}$$

If the sampler has N stages, then a single clock pulse sent into the clock delay chain will acquire N consecutive data samples. If a larger number of samples is required, then the sampler must be repetitively clocked, and the data from each clock stored. If the data samples acquired by a repetitive clock are to be consecutive, the clock must have period $T = N\Delta_{DC}$.

Ideally, we would like to save N consecutive data bits into an output register every T seconds. However, the sampler with a repetitive clock does not present all N consecutive bits at its latch outputs simultaneously. There are two problems that must be solved. First, the delay of the clock at each sampler stage skews the valid times of adjacent latch outputs by $Delta_C$. The skew across a number of channels prevents them from being latched simultaneously. Second, there are several clocks propagating through the sampler at once, and they divide the sampler into *clock sections*. The data samples produced by adjacent clock sections are not necessarily consecutive. In general, only the data samples within a section are consecutive.

Because the clock signal is delayed at each latch, the data valid times of adjacent latches are skewed by $Delta_C$. By delaying the earlier latch outputs by the appropriate amount, we can re-align the samples to the extent necessary to successfully latch them into the output register. This re-alignment can be achieved by adding a set of deskew latches to delay the upstream half of the channels in each clock section by $\frac{T}{2}$. These latches are driven by a clock of the same period, T, as the sampling clock, but approximately 180 degrees out of phase. Thus, the skew of the deskew latch outputs relative to the downstream channels is mostly canceled. If the phase of the deskew latch clock is optimal, then the data valid window for the output register is determined by the skew among the downstream channels of the section. Fig. **??** shows a complete continuous matched delay sampler, and the deskew latches can be seen directly below the clock delay chain.

In the typical sampler, Δ_{DC} is significantly smaller than $Delta_C$, and so there are several clock edges propagating through the sampler at once. Each clock traverses one section during each period T. The data stored in each section during a particular period is not consecutive with the data of the other sections. There is, however, a constant relationship among the sections that is determined by Δ_D and Δ_C. The samples from the various sections can be synchronized by delaying the leading sections with clocked FIFO's of the appropriate number of stages. The FIFO stages are driven by a single clock of period T. In the example sampler, there are four clock cycles present at once. As can be seen in Fig. ??, the first, second, third, and fourth sections are delayed by three, two, one, and zero clocks, respectively.

There are two issues that must be considered in the design of a particular matched delay sampler structure. First, once a desired value for the sampling interval, Δ_{DC}, has been chosen, there is a practical constraint on the nominal values of Δ_D and Δ_C. If Δ_C is not an integral multiple of Δ_{DC}, then the clocks in the sampler will be out of phase, and the synchronization FIFO for each section will have to be clocked by a separate clock. The complexity of the system will be increased significantly. Second, the input to output latency of the sampler increases with the ratio $\frac{\Delta_C}{\Delta_{DC}}$.

3.2 Matched Delay Pattern Generation Technique

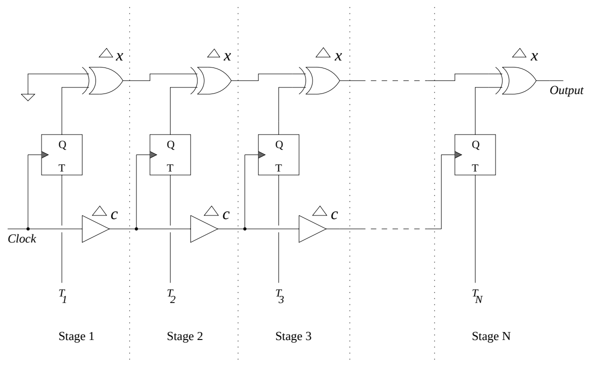

Figure 1: Matched Delay Generator

This section describes the application of the matched delay technique to the generation of high-speed data streams with precise pulse width resolutions. The basic architecture of the generator is illustrated in Fig. 1. Data is fed to the inputs of an array of T-type flip-flops. A clock pulse is then sent down a chain of delay elements, all of which have delay Δ_C. Each flip-flop (except the first one) taps this chain at the output of the corresponding delay element, so the effect is that the flip-flops are clocked in sequence $1, 2, \cdots N$ with an interval of Δ_C. Each flip-flop that has a high T input will toggle its output at the arrival of the clock pulse's rising edge. This, in turn, toggles the output of the corresponding XOR gate, which has a delay of Δ_X. This output then propagates down the XOR chain to the output. In this way edges are inserted into the serial data stream.

XOR gates are used in the data delay chain since a transition on either input switches the output. When a flip-flop output is held constant, the corresponding XOR functions simply as a delay element; the output from the previous XOR is passed through (inverted or not depending on the value of the flip-flop) after a delay Δ_X. Alternatively, when the flip-flop toggles, it inserts an edge onto the data chain as discussed above.

To find the resolution of the generator, it helps to view the generator as a cascade of stages, each consisting of a T flip-flop, XOR gate, and clock delay element. The clock pulse arrives at stage i at time $i\Delta_C$, at which point the flip-flop can toggle its output. Since the flip-flop's delay affects each stage equally, it can be ignored. This switching output causes XOR$_i$ to switch Δ_X later, at time $i\Delta_C + \Delta_X$. Similarly, the clock arrives at stage $i+1$ at time $(i+1)\Delta_C$, so XOR$_{i+1}$'s output changes state at time $(i+1)\Delta_C + \Delta_X$. However, the edge created "upstream" by XOR$_i$ arrives at XOR$_{i+1}$ at time $i\Delta_C + \Delta_X$. This causes XOR$_{i+1}$ to switch at time $i\Delta_C + 2\Delta_X$. It can be seen that the time difference between edges, Δ_{XC}, is

$$\begin{aligned} \Delta_{XC} &= [i\Delta_C + 2\Delta_X] - [(i+1)\Delta_C + \Delta_X] && (2) \\ &= \Delta_X - \Delta_C && (3) \end{aligned}$$

The above sequence is illustrated in Fig. 2, which shows the output of the first three stages superimposed on the clock seen at each stage. In this example, all T flip-flops are set to toggle on every clock pulse. Each stage is clocked Δ_C after the previous stage. An edge is generated at each stage Δ_X after the clock arrives. An edge propagating down the chain causes a transition at the following "downstream" stage Δ_X later. This is illustrated by the diagonal arrows on the right-hand of the figure. It can be seen graphically that the width of a generated pulse is $\Delta_X - \Delta_C$.

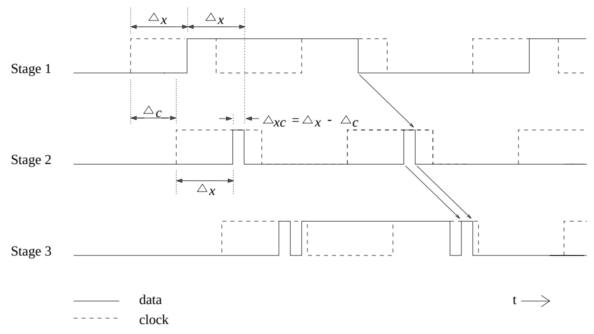

Figure 2: Pattern Creation Using Matched Delay Technique

Like the sampler described above, the generator's resolution depends on the difference $\Delta_X - \Delta_C$ of matched delay elements, which allows high-speed and fine-resolution waveforms to be generated.

3.2.1 Continuous Pattern Generation

Since each stage can create one edge per clock period, an N-stage generator can only create edges over an $N\Delta_{XC}$ interval for each clock pulse. To make data patterns larger than this interval, the generator must be repeatedly clocked. There are two major constraints to meet with regard to repetitive clocking: ensuring that the resolution remains a constant Δ_{XC} between edges created by different clock pulses, and guaranteeing that setup time requirements for the T flip-flops are met.

First, consider two clock pulses, p_1 followed by p_2, traveling down the clock chain of an N-stage generator. Assuming $\Delta_C < \Delta_X$, the edge generated at the first stage by a clock pulse is actually the last edge to reach the output, while the edge generated at stage N is the first one out. To keep a constant data stream, the time between the last edge e_1 due to p_1 and the first edge e_2 due to p_2 must be the same as the time between any two edges, which is the resolution Δ_{XC}. Since an N stage generator has $N-1$ clock delay elements and N XORs, the output times of edges e_1 and e_2 are $N\Delta_X$ and $(N-1)\Delta_C + \Delta_X + T$, respectively, where T is the time between clock pulses, that is, the clock period. This leads to the expression $T = N(\Delta_X - \Delta_C)$, which shows the close relationship between the number of stages, the resolution and the clock period. Consider an example where $\Delta_X = 500$ps and $\Delta_C = 400$ps. The resolution is 100ps. Assume also that there are $N = 64$ stages, which requires a clock period $T = 6.4$ns.

The second part of maintaining a constant Δ_{XC} resolution requires that the input data be presented to the T flip-flops in a somewhat non-intuitive fashion explained below, using the values from the previous example. First, note that $T < N\Delta_C$. This indicates that multiple clock pulses are present at any given time, which effectively wave pipelines the clock chain. The number of pulses present p is $N\Delta_C/T = 4$. There is one pulse per $N/p = 16$ stages. Each 16 stages defines a *section*.

Now consider two stages i and j in adjacent sections, where $j = i + 16$. Since i and j are 16 stages apart, and $T = 16\Delta_C$, the edges created at stages i and j are generated simultaneously by two consecutive clock pulses. The edge generated at stage j arrives at the output at $t_j = j\Delta_C + (N - j)\Delta_X$, while the edge generated at stage i arrives at $t_i = i\Delta_C + (N - i)\Delta_X + T$. The T accounts for the fact that the two clock pulses are separated in time by one clock period. The difference in arrival times, then, is $t_i - t_j = (j - i)\Delta_{XC} + T$. However, maintaining constant resolution requires that $t_i - t_j = (j - i)\Delta_{XC}$. It can be seen now that this occurs if and only if both edges are generated by the same clock pulse, which happens if the data sent to the second section is skewed by one clock period relative to the data sent to the first section. This one-cycle delay is accomplished by using simple static D-type flip-flops. Continuation of this reasoning leads to requiring one additional layer of these delays per section. Fig. 3 illustrates these delays for 16-stage sections.

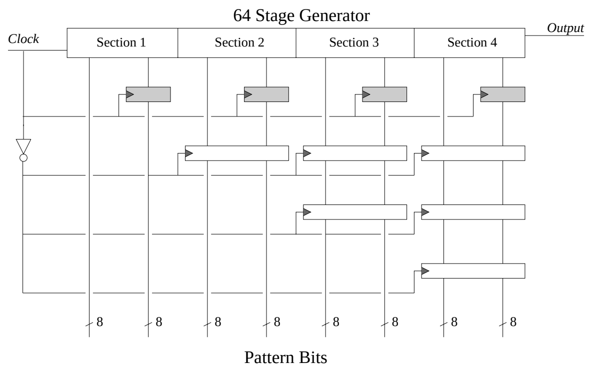

Figure 3: Latch Configuration for Continuous Pattern Generation

Setup time for the first half of each section is met by clocking the above delays on the opposite phase of the generator's clock, as shown in Fig. 3. However, the clock pulse will only be halfway through the section when the delay flip-flops are clocked again, causing the data in the second half of the section to be lost. This loss is avoided by the insertion again of one layer of delay flip-flops which are clocked in phase with the generator's clock. These delays are shaded in Fig. 3.

3.2.2 Resolution Limitations

We have shown that the resolution of the generator is determined by the difference of delays, Δ_{XC}. It is theoretically possible to get an arbitrarily small resolution by setting Δ_X and Δ_C arbitrarily close together. However, in reality, the actual values of Δ_X and Δ_C, and thus the resolution, will differ from their nominal values because of fabrication

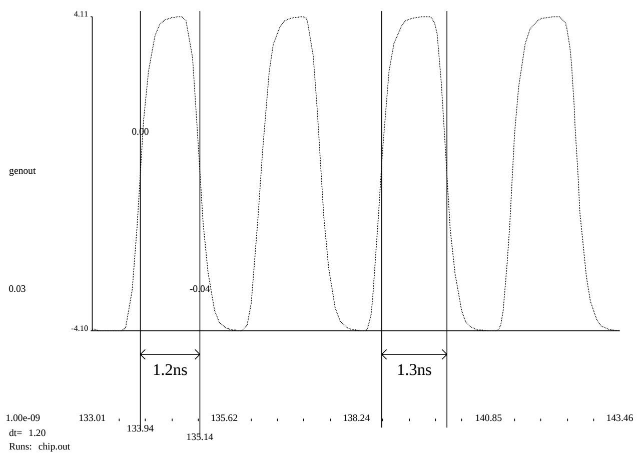

Figure 4: Precise Edge Placement in Generated Pattern

and environmental variations. Severe discrepancies between actual and nominal values will cause incorrect operation. This sets a lower limit on the resolution and suggests that Δ_X and Δ_C should be kept large relative to the possible delay changes due to process variations. Additionally, Δ_X and Δ_C should be adjustable to compensate for these variations.

The finite analog bandwidth of the XOR gates causes them to act as low-pass filters on the generated data pattern, limiting its maximum frequency. Generated data pulses that have a width below a threshold value will be attenuated completely and not make it down the data delay chain. This restricts how close edges can be placed to each other in a generated pattern, which impacts the specification of the maximum output frequency. In addition, the output drivers and the package also attenuate the output pattern, further restricting the maximum frequency.

3.2.3 Matched Delay Generator Results

The generator and supporting test structures were implemented in a full-custom layout and simulations were run on the extracted circuits. The chip has 32,071 transistors and occupies a die area of 2.71mm $\times$ 6.15mm. It was designed according to the specifications given in the example in section 3.2.1.

Fig. 4 illustrates the generator's edge-placement ability by showing two 1.2ns pulses followed by two 1.3ns pulses. These pairs differ in width by the maximum resolution, 100ps. The 1.2ns pulse is also the minimum-width pulse the generator can create, giving the generator a maximum bit rate of 833 Mb/s.

3.3 Matched Delay Data Recovery Technique

This section describes a data recovery circuit which utilizes the matched delay sampler described in section 3.1. The very fine resolution of the sampler provides an oversampling capability that allows the recovery of very high-speed data. Also, the scheme requires only a slow clock since multiple input data bits are captured in the delay lines.

In this recovery technique, the input data is oversampled and digitized. The oversampled digital data is then processed to extract the phase relationship between the the data and local clock. The "correct bit" in the oversampled digital data can be recovered by locating the NRZ input data boundary. Also, a digital phase locked loop circuit provides

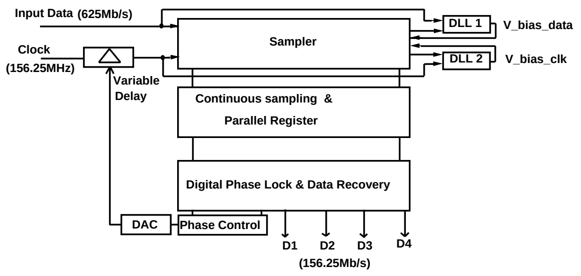

Figure 5: Block Diagram for Data Recovery

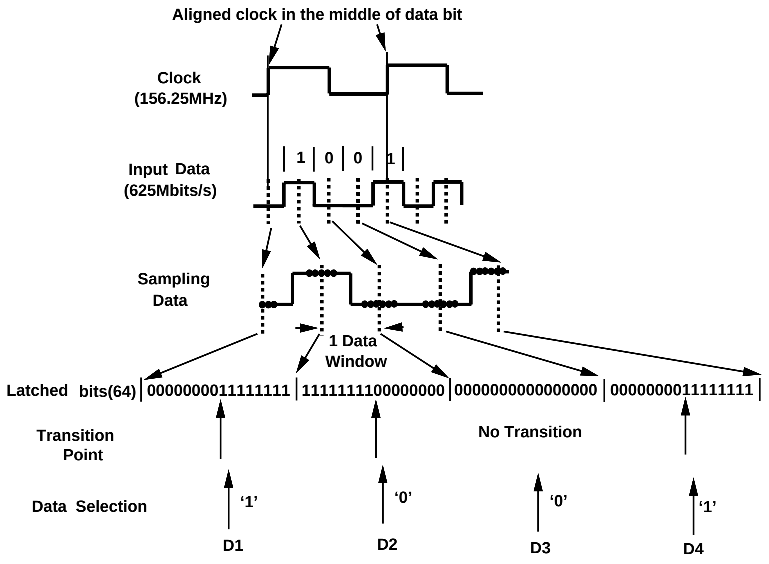

Figure 6: Waveforms for Recovery Signals

the dynamic alignment of the clock rising edge to the mid point of the incoming NRZ data bit.

A block diagram of the data recovery circuit is shown in Fig. 5. The waveforms seen at each stage in the block diagram while steady-state phase locking occurs are shown in Fig. 6. While in the steady state, the slow clock is aligned in the middle of the first data bit of the four bits to be recovered in each clock cycle. After oversampling the input data using the matched delay sampler, the sampled bits are digitally processed to recover the data and to lock the clock phase between the data and clock.

The sampler is designed to sample input data at a desired fine resolution, which is controlled by the relative skew between the clock and data signals. In principle, the sampling clock period can be much slower than the data rate, because the actual sampling resolution is determined by the difference in delays of two delay elements. However, the clock period is decided upon by the system application. Because of propagating both the data and clock through two different delay units and having a finite number of the sampling stages, there are some data points which will not be taken after the clock propagates through the sampling stages without careful design of the delay units and the clock period.

The time to traverse the sampling clock through the clock delay chain is $N\Delta_C$, where N is the number of sampling stages. In the 1:4 demultiplexer data recovery circuit, the total delay time in the clock delay chain should be $4T$, where T is the clock period. Recall

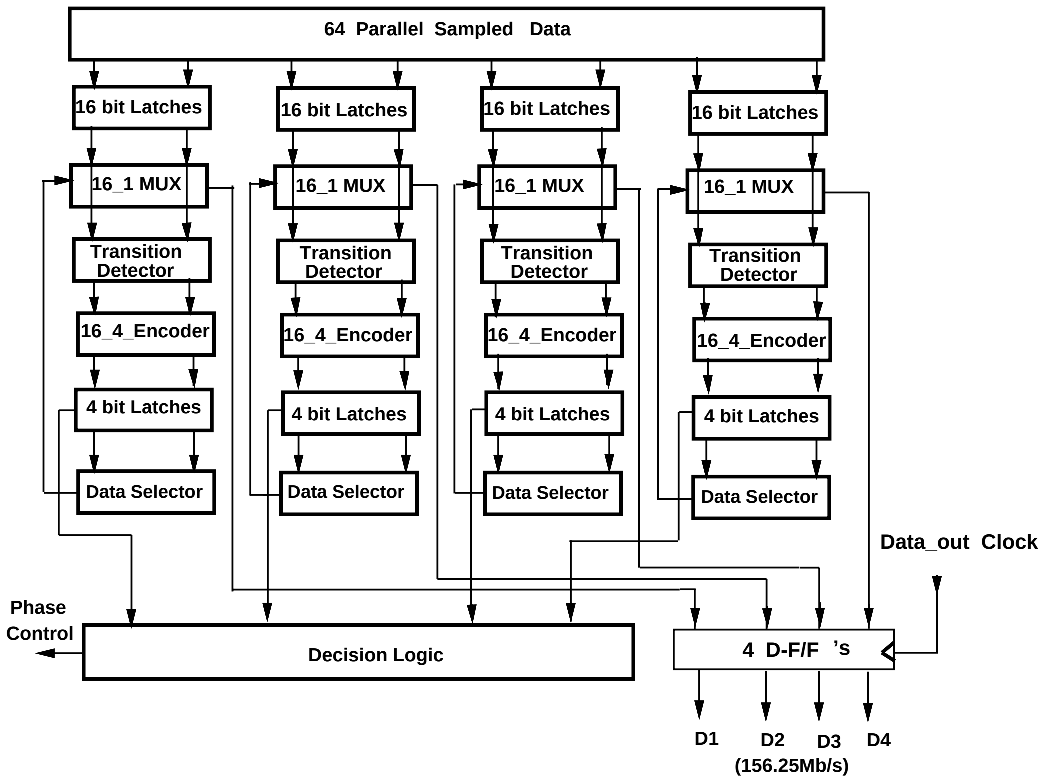

Figure 7: Decision Circuitry for Data Recovery

from section 3.1 that the resolution Δ_t of the sampler is determined by the difference of the data delay Δ_D and the clock delay Δ_C, that is, $\Delta_t = \Delta_C - Delta_D$. For continuous data sampling, it is obvious that if a sampling clock edge reaches the first latch at time t=0, its final sample at the N^{th} sampling stage is the data at time $t = (N-1)\Delta_{CD}$, and so the next clock must arrive at the first latch at $t = N\Delta_{CD}$. Thus the sampling clock period, T, and the data and clock delay values Δ_C and Δ_D are related by the equation $T = N\Delta_t$. For cyclic operation, $T = M\Delta_C$ where M is an integer number. In the 625Mb/s data recovery application, T = 6.4ns, N = 64, M=16, Δ_C = 400ps and $Delta_D$ = 300ps. Thus Δ_t is 100ps in order for 1 bit NRZ input data (pulse width=1.6ns) to be oversampled into a 16 bit digital pattern. In other words, a one bit data window is composed of 16 bits of sampled digital data. Therefore, in each 6.4ns cycle (156.25MHz) four 16–bit data windows (total 64 sampled digital pattern) are available for processing. This enables a 1:4 demultiplexing type of data recovery.

Since multiple periods of the data and clock (four clocks) are present in the sampler, latched data can be overwritten before it can be digitally processed. By shifting latched data to another latching stage, the continuous data can be safely stored. This continuous data sampling scheme [?] using a FIFO structure for the sampler is designed so that no sampled data will be lost. The continuous data sampling block, shown in Fig. ??, requires additional chip area and its own clocking scheme, but this trade-off is necessary to obtain higher speed continuous data sampling using the matched delay sampler. At the end of this continuous data sampling block the oversampled digital data is latched in parallel at the slower clocking rate for digital processing as shown in Fig. 7.

Data recovery is done by locating the transition point (input data boundary) in each data window. A noise filtering step is inserted to eliminate possible isolated noise pulses and metastability in the sampler. Selecting one of the 16 oversampled bits based on

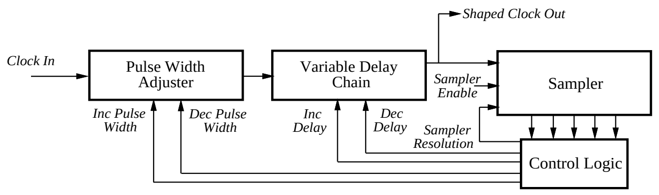

Figure 8: Block Diagram for Clock Shaping Circuit

the transition point recovers data. If there is no transition in a window, the bit derived by the previously latched transition point is taken as the recovered data. By doing this, the bit error rate will be very low. The encoded transition points also give the phase information about the clock used to synchronize with the input data. This digital phase value is converted to an analog signal and control the variable delay on the clock input line. structure, stability issues must be addressed. This system has been shown mathematically to be stable.

Two delay-locked loops as shown in the block diagram in Fig. 5 are built for maintaining the precise delay time on each delay line in the sampler.

3.4 Matched Delay Clock Waveform Shaper

This section describes a method for restoring a distorted clock signal to a 50% duty cycle. The clock shaper periodically samples a clock signal, checks the samples for the clock's current duty cycle, and then, if necessary, adjusts the clock to a 50% duty cycle. The components used in the clock shaper were wave pipelined [?] to permit precise shaping of high speed clocks using standard CMOS. The design of these components also allows the clock shaper to internally compensate for temperature and process variations. Furthermore, the design is flexible in that very simple changes to the control logic will allow shaping clocks to non-50% duty cycles.

A block diagram of the clock shaper appears in Fig. 8. This diagram shows the clock shaper to consist of four parts: a sampler, a variable delay chain, a pulse width adjuster (PWA), and control logic. The sampler periodically takes 32 samples over roughly one and a half periods of the clock to be shaped. These samples are then processed by the control logic to determine if adjustments to the clock are necessary. If the clock's duty cycle requires adjusting, the control logic uses the PWA to obtain the correct duty cycle. The control logic also adjusts the sampler's resolution and the variable delay chain to compensate for temperature and process variations experienced by the clock shaper. How the control logic decides to make these adjustments will now be examined in more detail.

The control logic continually adjusts the clock until the 32-bit sampler samples the clock as shown in Fig. 9. When the clock is correctly adjusted, only two rising edges and one falling edge are sampled and these edges occur in the positions indicated. To achieve this desired sampling, the control logic first adjusts the sampler's resolution so that only two rising edges are sampled. These two rising edges are then moved to the positions shown in Fig. 9 by tweaking both the variable delay chain and the sampler's resolution. Once the two rising edges are in place, the control logic uses the PWA to shift the clock's falling edge to its correct position. Shifting these three clock edges to the positions shown in Fig. 9 restores the clock signal to a 50% duty cycle. The control logic then only makes adjustments when the clock deviates from this expected sampling.

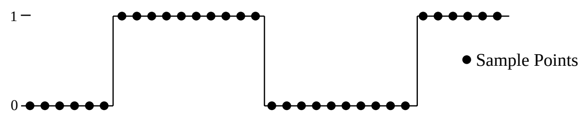

Figure 9: 32 Bit Sampling of Correctly Shaped Clock

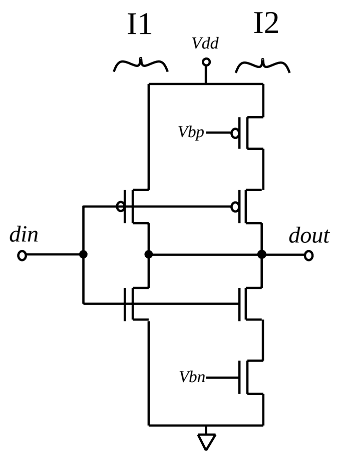

Figure 10: Variable Delay Element

The control logic could be easily modified to dynamically select an arbitrary duty cycle.

3.5 Variable Delay Element

Before discussing the logic blocks that make up the clock shaper, the variable delay element central to each of these blocks will be examined. The design goals of this element were:

- large delay range
- very precise delay adjustability
- equal rise and fall times
- low power consumption

These goals led to the variable delay element design shown in Fig. 10.

The delay element shown is two static inverters, I1 and I2, connected in parallel with inverter I2 current starved. By controlling I2's two bias voltages, V_{bn} and V_{bp}, the delay element's rise and fall times can be changed to achieve the desired delay. In addition to controlling the delay, these two bias voltages can also be used to create a symmetric output, that is, an output with equal rise and fall time. The range of the delay element and its delay precision are determined by the size of I2 relative to that of I1. A larger I2 makes the delay adjustable over a wider range of values, but this makes precision coarser and more dependent on how well the bias voltages are controlled. Making I2 smaller sacrifices delay range for finer precision and less susceptibility to fluctuations in the bias voltages. Since the variable delay element is fully static, it dissipates zero DC power.

The circuit shown in Fig. 11 makes bias voltage adjustments to create equal rise and fall times. To see this, note that the p-type transistors and the n-type transistors can be considered to be two resistors in series. When their resistances are equal, the

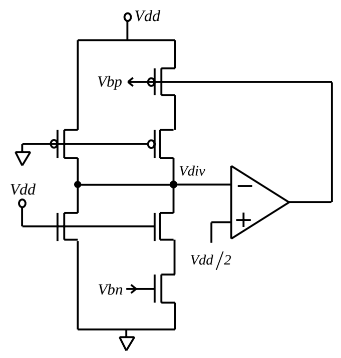

Figure 11: Bias Control Circuit for Variable Delay Element

voltage at V_{div} will be $1/2 \times$ Vdd. This implies equality of the pull-down and pull-up currents, and therefore symmetric rise and fall times. This control scheme requires only one externally controlled bias voltage input, V_{bn}, since the high-gain op-amp, functioning as a comparator between V_{div} and $1/2 \times$ Vdd, continually adjusts V_{bp} to equalize the currents. This automatic adjustment helps provide the clock shaper with immunity to process and temperature variations.

3.5.1 Functional Blocks of the Clock Shaper

The clock shaper's logic blocks are composed of the matched delay sampler, a variable delay chain, and a pulse width adjuster (PWA). The sampler is described in section 3.1.

The variable delay block is actually just a chain of variable delay elements. Each delay element can only be set to one of two bias voltages, one of which creates a relatively long delay, the other a relatively short delay. Initially, half the delay elements will be set to have a "long" delay and the other half set to have a "short" delay. As time passes, the control logic incrementally switches individual delay elements from one bias voltage to the other to obtain the desired overall delay.

The PWA is a string of variable delay elements and is very similar to the variable delay chain. The only difference is in how the PWA's bias voltages are used. Instead of setting all bias voltages so that each delay element has a symmetric output, some biases are intentionally mismatched to make rise times and fall times unequal. This inequality in rise and fall times is what makes clock shaping possible. Figure 12 illustrates how this rise and fall time inequality can be used to increase the width of a clock's 1 pulse. The first signal in Figure 12 shows a clock signal with a narrow 1 pulse. The second signal shows this clock passed through a delay element whose rise time is slower than its fall time. This signal would then pass through another delay element with symmetric output to create the signal at the bottom of Figure 12. This final clock's rise and fall times are again equal, but the clock's one pulse is wider. This same method can also be used to decrease the size of a clock's 1 pulse. The only difference for narrowing 1 pulses is that the delay element with asymmetric output has its fall time slower than its rise time.

3.5.2 Results

For the clock shaper to work properly, certain stability conditions have to be met. First, the variable delay chain and the PWA cannot shift a clock edge by more than the sampler resolution time before each sampling iteration. This is necessary so that the clock shaper

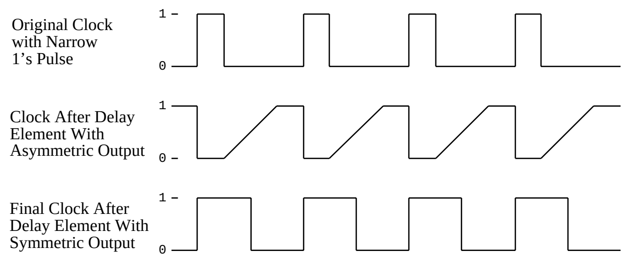

Figure 12: Increasing Width of 1's Pulse

does not continually overadjust the clock signal, which would prevent convergence on the 50/that when a clock is sampled and an adjustment is made to it, the sampler must wait for the adjustment to affect the clock before sampling again, i.e., the sampler should always work on the updated signal. For the current clock shaper design, this wait time is about 60 ns. Also, since the clock shaper is wave pipelined, the signals traverse only combinational logic. Due to the analog bandwidth limitations of the logic, the clock signal must have a minimum pulse width to be able to traverse all of the circuitry. This minimum pulse width, found to be 1.2 ns, limits both the speed of the clock to be shaped and the degree to which its duty cycle can differ from 50%. With these conditions accounted for, simulations of the extracted 1.2μ layout were done with CAzM [?] to evaluate the system's performance.

The performance of the variable delay element will be examined first. When used to control delay, the range is the difference between the minimum and maximum delay times to which the element can be set using V_{bn}. Since it was desired to never have the element's biased n-transistor in the cutoff region, V_{bn} as swept between 1V and 5V. Simulation results show that the delay range of the variable delay element is 240 – 310 ps. The change in delay caused by V_{bn} tracks linearly with change in V_{bn}, i.e.,

$$\frac{\Delta d}{\Delta V_{bn}} \approx 17.5 \frac{ps}{V}$$

The ability of the delay element to shift edges was also determined. This shaping range is the maximum time a delay element can shift a clock's rising or falling edge. To measure this, two delay elements were chained together with the first delay element having asymmetric output and the second having symmetric output. The bias voltages on the asymmetric element were then set to determine the maximum distance either signal edge could be shifted. The shifting distance for the rising edge was found by setting V_{bn} to 1V and V_{bp} to 0V. The falling edge's shifting distance was found by setting V_{bn} to 5V and V_{bp} to 4V. The shifting range was 60ps for the rising edge and 85ps for the falling edge. This yields the following relationships:

$$\frac{\Delta s_{rise}}{\Delta V_{bn}} \approx 15 \frac{ps}{V}$$

$$\frac{\Delta s_{fall}}{\Delta V_{bn}} \approx 21 \frac{ps}{V}$$

The precision at which both the delay and edge shifting can be done is determined by how well the bias voltages, V_{bn} and V_{bp}, can be maintained.

The performance of the sampler was also examined. The important parameter for the sampler is the sampling resolution. Since the delays in the clock line are constant,

Figure 13: Simulations of the Clock Shaper

the resolution is determined by the delays of the sampler enable line, which in turn are determined by V_{bn}. Sweeping V_{bn} from 1V to 5V gives sampler resolutions from 100ps – 500ps. Since it is desired for the sampler to take 20 samples per clock period (as shown in Figure 9), and the clock is limited to a 1.2ns minimum pulse width, this resolution range permits shaping of clocks with speeds of 100 – 400MHz. Also, since only 20 samples are taken of the clock period, clocks can only be shaped to within 5% of a 50% duty cycle. For more precise shaping, a larger sampler that is capable of taking more samples of a clock period is required.

Figure 13 shows simulations of the clock shaper returning a 50% duty cycle to two differant clock signals. The first signal in this figure shows a clock with narrow 1 pulses before clock shaping and the second signal shows this clock returned to a 50% duty cycle by the clock shaper. The third signal is a clock with wide 1 pulses that is changed by the clock shaper to the 50% duty cycle clock shown at the bottom of Figure 13.

4 Comments on Practical Issues

In this section, we discuss some practical considerations that affect most potential applications of the matched delay technique. There are several inherent characteristics of the technique which are the sources of its advantages, but which might also present some new problems to the system designer. These are issues that must be carefully explored and characterized during future work.

4.1 Resolution and Accuracy Considerations

A very important characteristic that has a significant influence on the ultimate usefulness of the matched delay technique is timing accuracy. In theory, the difference between the clock and data delays can be made arbitrarily small, and so the timing resolution of the sampler or generator can be increased to an arbitrary level. However, the accuracy of the resolution is limited by the accuracy of the delays of the clock and data delay elements. If the nominal delay difference is set to a very small value, then even a tiny error in the delay elements can create a very significant error in the actual difference value.

The first concern related to the accuracy of the delay elements is process and temperature variation. These variations can create a significant error in all the delay elements in a sampler or generator. A means to dynamically compensate for process and temperature in real time is necessary to maintain delay accuracy. By making the delay elements adjustable with a control voltage, and using delay-locked loop techniques, we have been able to successfully maintain delay accuracy in some of our early designs. Additional research into compensation techniques will provide further refinement in the accuracy that can be obtained from the sampler and generator.

An additional facet to the delay accuracy problem appears in the data delay chain of both the sampler and generator. If the data delay element has a data dependency, i.e., rising and falling edges are passed with unequal propagation delays, this imbalance will be magnified as the data propagates down the chain. Our primary means of eliminating this data dependency is the use of differential logic for the data delay elements. This approach has been successful but has disadvantages in power consumption and chip area. Further work is needed to develop power and area efficient delay elements with no data dependency.

Process and temperature gradients on a particular chip can cause the delay to vary from one stage to the next along the delay chain. This effect will introduce timing jitter into the data and clock signals. The only measures that we have taken against this problem are the standard VLSI layout techniques that minimize the effects of gradients. More work is needed to determine if a more advanced gradient compensation approach is practical.

The other major phenomenon that limits resolution accuracy is common to all electrical systems. Noise will cause timing jitter in the delay elements and so create a finite uncertainity in the timing interval between two matched delay stages. Obviously, the noise cannot be eliminated, but it can be minimized using standard isolation techniques. The potential of the matched delay technique to provide very accurate, very fine timing resolution justifies the cost of careful noise isolations measures. Further study and experience with the sampler and generator will allow refinement and optimization of our noise minimization approach.

4.2 Independence of Resolution and Bandwidth

As described earlier, the resolution of the matched delay devices is determined by the difference between the propagation delays of the delay elements on two parallel tapped delay lines. An individual delay or latch cycles at a rate much slower than the resolution frequency. As a result, the resolution is not limited to the minimum propagation delay of the circuit elements. In fact, to a large extent, the resolution is independent of the analog bandwidth of the circuit components, and so the sampling or generation rate can exceed the maximum switching rate of the individual components.

With conventional data sampling and generating techniques, obtaining high resolution generally requires using a high performance circuit technology. Therefore, the analog bandwidth usually places an upper bound on the maximum resolution. The relative independence of resolution and bandwidth provided by the matched delay technique creates the possibility of using CMOS circuit technologies for applications that require high timing resolution but relatively low signal bandwidth.

Most circuit families support higher bandwidth signals on a chip than between chips. The focal point of the bandwidth limitation typically is the package interface. For many applications, the on-chip bandwidth of CMOS is high enough, but the chip interface bandwidth is insufficient. In these cases, the sampler and generator would allow a high performance system to take advantage of the lower power consumption and higher integration levels of CMOS if the package interfaces could be made fast enough. Using BiCMOS interface circuits, or special CMOS circuit designs, such as small-swing differential CMOS, for the drivers and receivers, may provide enough interface bandwidth. Work is needed to develop improved interface circuits, so that CMOS implementations of the sampler and generator can be used in high performance applications.

Even when very high bandwidth circuits are used, the matched delay technique is most likely to fit best in over-sampling applications. In situations where the timing resolution is comparable to the bandwidth, simpler sampling and generation architectures can be used, Circuit technology choice is the primary consideration in these cases.

4.3 Data Parallelization

Another fundamental characteristic of the sampler and generator is an inherent serial-to-parallel or parallel-to-serial conversion, respectively. This conversion provides two important benefits. First, the data rate that must be handled by the logic that processes the data for the matched delay devices is significantly slower than the external data rate. This factor creates the possibility that an entire high performance system might consist of relatively inexpensive circuitry, because the sampler and generator provide very high timing resolution, and the parallelization allow relatively slow data processing.

The second benefit of the parallelization stems from the fact that many applications, especially in the communications area, need to operate on parallel data. In these cases, conventionally designed systems often contain multiplexors and demultiplexors. The sampler and generator can perform these functions with practically any desired parallelization factor.

The greatly increased number of internal signals that results from the parallelization is a potential problem from a system design perspective, however. A situation that requires a large number of external channels and some central logic that must simultaneously observe or drive all of the corresponding internal signals may result in a difficult interconnection problem. The most important point in preventing this problem is to achieve the correct balance between the degree of parallelization and the the speed of the central logic. Research is needed to develop guidelines for finding this balance.

Multi-chip module (MCM) technology will be a very important solution for this potential problem in many cases. An MCM based system can have a much larger number of connections between chips than a system implemented on one or more printed wiring boards. Also, the signal speeds of those connections can be significantly higher. These factors will allow a matched delay based systems with large numbers of external signals to be designed.

4.4 Clock Speed Reduction

A very important advantage of the matched delay technique is the fact that the input clock is slow relative to the sampling or generation rate. Neither the sampler nor the generator require a high speed clock to achieve high timing resolution. With conventional techniques, the clock is often the fastest signal used in a system. Matched delays might allow the use of relatively low bandwidth circuit families for some high resolution applications because it is not necessary to supply or distribute a very fast clock.

Sampling schemes in which either the data or clock are delayed also are able to sample at rates higher than the switching speed of the individual elements, and also do not require a clock with a frequency equivalent to the resolution. However, their resolution is

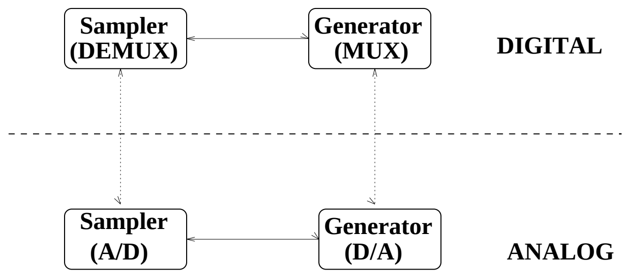

Figure 14: Matched Delay in Analog and Digital Domains

limited by the absolute delay of the delay elements. Since the matched delay resolution is determined by the *difference* between two delays, the sampler or generator can have a resolution interval that is much smaller than the minimum possible absolute delay.

There is a tradeoff between clock speed and the degree of data parallelization that must be balanced to fit each particular application of the matched delay technique. This tradeoff must also take into account the chip area issue. Further experience in building matched delay based systems will help determine the value of the reduced clock rate.

5 Matched Delay Analog Sampling and Generation

A similar approach to the digital sampler can be followed for analog signals. The digital delays used in the data path can be translated to analog gain and delay blocks and the latches can be converted to sample/hold circuits as shown in Fig. **??**. In this case, each sample and hold circuit maintains a sample of the analog data with a resolution in between each stage of relative difference delay. The problem to resolve now is to find the circuits that will delay the analog signal and provide some gain while maintaining an undistorted signal. The sample and hold circuits should store the signals without causing charge injection or any distortion to the analog data path or to any of other S/H circuits. The clock delay path is the same as the digital matched delay sampler.

This structure can be used to implement high speed A/D converters. The implementation can be serial or parallel. Serially all the samples are multiplexed into an A/D converters, while in parallel the converters are at the output of each S/H circuit.

A variation of these matched delay analog sampling structure is to replace the signal delay by a zero delay path and unity gain. In this case, a clock is applied to all the sample and hold circuits, the S/H circuits then will be in the sampling mode at the same time. As the falling edge of the clock pulse travels through the clock path the S/H circuits start opening the input switch and move into hold mode. Each S/H circuit will contain a sample that is Δ_p time units apart from the previous circuit in the chain. The sample and hold circuits do not have to be as fast as the input signals to capture the signal. The S/H chain length and the delay of digital delay line are main factors in determining the resolution of the sampled analog signal. An example implementation is found in **??**.

We have pointed out that matched delay analog sampling is the counterpart of digital sampling. This analog sampling has a direct application on high speed A/D converter. This observation is a motivation to search for the counterpart of the digital generator in the analog domain. If such a scheme exists, it will enable the implementation of a high-speed D/A converter using the matched delay technique. Fig. 14 illustrates the relationship of the matched delay technique in both the analog and digital domains.

6 Conclusion

We have presented the matched delay technique, which is a methodology for achieving very fine timing resolutions for both the sampling (demultiplexing) and generation (multiplexing) of serial digital data streams. Example implementations of the technique in the digital domain, namely high-speed data recovery and clock waveform shaping, were also presented. In addition, we have developed a matched delay analog sampling technique which serves as the counterpart of matched delay digital sampling.

We also have discussed the issues that must be examined if the matched delay technique is to be developed into a practical and useful approach for designing high-performance digital systems.