A Logic-Enhanced Memory for Digital Data Recovery Circuits

Kenneth J. Schultz and P. Glenn Gulak
Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON, Canada M5S 1A4

Abstract: Compact hardware solutions to the digital data recovery problem are presently limited to analog techniques. This paper describes a means of digital data recovery using digital hardware, using a digital eye diagram, where a single bit represents each pixel. A logic-enhanced Orthogonal Algorithmic Access Memory (OAAM) is proposed to facilitate this approach, and its operation is described in detail. The approach is technology-independent, and it has a large range of applications, including wired and radio data communications, approaching Gb/s data rates.

I. INTRODUCTION

The function of data recovery circuitry is to recover digital data from a demodulated (baseband) analog input which has been corrupted by channel noise, intersymbol interference, and other degradations. As the integration level and speed of digital communications systems increases, the realization of compact, fast and reliable data recovery circuits becomes more important. Single-chip implementations of such circuitry are presently limited to analog hardware or digital software. The former is compact and fast, but it offers limited accuracy. In the latter case, multi-bit quantization provides improved accuracy, but speed is compromised. A practical digital hardware solution should achieve high accuracy, area-efficiency and analysis capability, as well as high speed.

This paper proposes a means of finding the optimum slicing level and sampling instant within a baud interval, using digital hardware, at speeds comparable to those attainable using analog circuits. The central element in this scheme is a logic-enhanced — or "smart" — memory, which stores the pixels of a digital eye diagram of the incoming signal. This Orthogonal Algorithmic Access Memory (OAAM) computes the optimum slicing level and the optimum sampling instant within a baud interval. This enables excellent data throughput rates, since the internal memory data bandwidth can be exploited without memory input/output overhead. Densities are optimized, since external buses and processing circuits are not needed; rather, all required arithmetic and logic operations are performed inside the memory, where full-custom layout and integration with the memory data path minimize circuit area.

II. A DIGITAL EYE DIAGRAM

An eye diagram, such as the one shown in Fig. 1(a), is used to graphically depict the quality of a transmission channel or data receiver. Analysis of such a diagram can provide information on noise margin, distortion and timing sensitivity [2]. Moreover, an eye diagram provides a valuable paradigm for a digital hardware solution to the data recovery problem.

This solution requires an A/D converter at its front end. The converter quantizes the incoming analog signal to a baud rate $f_b$ (8 $\leq N = f_s/f_b \leq 64$). The converter is not required to fully encode its thermometer-code result, since a binary number is not needed. Rather, the A/D converter should output an $L$-bit word, where only one of the $L$ bits is high per sample, and the bit location corresponds to the quantized signal level.

The $L$-bit word is then written into the memory proposed in Section III. Generally, 1's over-write 0's, but 0's do not over-write 1's. If the ratio $N = f_s/f_b$ is an integer and the memory has $N$ words of $L$ bits each, operation can be timed so that the memory address corresponds to the sample number within a baud interval. After a few baud intervals, a digital eye pattern forms in the memory. Fig. 1(b) shows sample contents of a small $8 \times 8$ eye diagram memory. To implement a variable persistence feature, the age of each 1 should also be stored. This "aging" flushes out momentary eye closures due to such phenomena as impulse noise and signal fading. With an eye diagram residing in memory, digital processing can be used to find the widest eye opening (optimum adaptive timing) and the middle voltage level of this opening (optimum adaptive slicing level).

If additional processing is applied, further characterization and diagnostic data could be collected, including phase margin, bit error rate, and channel noise. In fact, circuitry built around such a digital eye diagram could provide, on a single IC, many functions presently confined to elaborate diagnostic equipment (see [3], for example).

Data recovery circuits based on this digital eye diagram could achieve better accuracy than analog methods at speeds much higher than those attainable by software techniques. Since embedded memory test methodology is well-known [4], the proposed technique is more testable than analog circuitry. Due to its area and power efficiency, this approach may even be preferable to software implementations for low-speed applications.

III. OAAM MEMORY ARCHITECTURE

A block diagram of the proposed Orthogonal Algorithmic Access Memory (OAAM) is shown in Fig. 2. From the perspective of port 1, we refer to data from a given sample as a word; thus, bit-parallel word-serial operations are performed on all the pixel data from a given sample, with samples processed serially. The core of the memory stores 1's and 0's corresponding to pixels in a digital eye diagram. Two basic operations are required:

(P1) Write the new A/D output word into the core, such that 0's do not over-write 1's.

(P2) Analyze the contents of the memory, looking for the widest eye opening.

We may safely assume that the characteristics of the eye are relatively slowly-varying, so that the analysis operation (P2) can be performed while the eye diagram is being updated (P1), enabling a

1. The general problem of synchronization is not addressed. It is assumed that standard techniques [1] are used to synchronize the receiver clock to the incoming data, up to the application of the techniques herein described.

2. Performance limitations of A/D converters are readily available in the literature and are not considered in this paper.

The authors would like to thank Mahshad Kooogoli of Northern Telecom for the initial suggestion to pursue this work. This work was financially supported by Bell-Northern Research and the Natural Sciences and Engineering Research Council of Canada.

0-7803-1254-6/93$00.00 © 1993 IEEE
Fig. 1: (a) An analog eye pattern; (b) sample 8 x 8 eye memory contents.

Fig. 2: Memory Block Diagram: $\phi_p$ is the sample clock at rate $f_p$; $\phi_A$ is an asynchronous clock for Ports 2 and 3.

speed-up due to parallelism. In fact, the two operations can be performed asynchronously from one another, allowing independent optimization of operating speeds.

A straightforward implementation could employ a dual-port memory core, where one port performs (P1) and the second performs (P2). However, performing (P2) on bit-parallel word-serial data would require finding the longest string of 0's, and its midpoint, in a single clock tick. Such operations are simplified when performed on bit-serial word-parallel data [5]. The proposed memory thus uses a bit-parallel word-serial port for (P1) and a bit-serial word-parallel port for (P2), and resembles an Orthogonal RAM [6].

As discussed in Section II, the memory contents should fade with age. This requires storing a count corresponding to each pixel in the memory. The count update operation depends on the new pixel data, and is thus best performed by Port 1 in conjunction with (P1). Count storage requires only single-port core cells, rather than the orthogonal dual-port cells needed for pixel storage. For a compact core layout, compatible with peripheral circuity, the two different core cells (count storage and pixel storage) should have equal pitch in the port 2 bit line direction, but unequal pitch along port 1 bit lines, as shown in Fig. 3.

In the remainder of this section, rows of the pixel memory, corresponding to different samples, are denoted by the subscript $n$ ($0 \leq n \leq N - 1$). Columns of the pixel memory, corresponding to different quantization levels, are denoted by the subscript $l$ ($0 \leq l \leq L - 1$). All cells associated with a given sample are referred to as $P[n]$ [this implies $V[n]$], and all cells associated with a given level are referred to as $P[l]$ [this implies $V[l]$].

Port 1 operation is summarized in Table 1. $Q[l]$ represents pixel data just read, $D[l]$ is new pixel data from the A/D converter, and $WE[l]$ is pixel write enable. The user is able to choose the maximum allowable age of 1's (persistence) through the variable $\max$. A block diagram of port 1 circuitry is shown in Fig. 4.

Table 1: Port 1 operation

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>none</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>increment</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$&lt;\max$</td>
<td>0</td>
<td>clear</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>$\max$</td>
<td>1</td>
<td>clear</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$X$</td>
<td>1</td>
<td>clear</td>
</tr>
</tbody>
</table>

Pseudo-code can be used to describe port 1 operation:

do in parallel for $l$ from 0 to $(L - 1)$

$\{$

count[l] = 0, P[n][l] = 0, age[n][l] = 0$

do serially for Data Samples from 1 to End_of_Time

$\{$

do from $n = 0$ to $(N - 1)$

$\{$

$Q[l] = P[n][l]$

count[l] = age[n][l] + $(Q[l] - D[l])$

$WE[l] = 1 \text{ or } \text{count}[l] = \max$

$\{$

count[l] = 0$

$P[n][l] = D[l]$

$\}$

$age[n][l] = count[l]$

$\}$

$\}$

While accesses to port 1 proceed sequentially by time (one $P[n]$ per cycle), Port 2 performs sequential accesses of the pixel memory by level (one $P[l]$ per cycle), and thus requires a separate counter, and its own row decoder (which actually selects a single column $l$, from the perspective of port 1). Port 2 also has one bit of column decoding, so that two columns $n = 2k$ and $n + 1 = 2k + 1$ share the same logic circuits. This simplifies layout of the column circuitry and allows pipelined operation, as described below.

2008
Fig. 4: Port 1 block diagram; XNOR represents bit-wise equivalence.

Port 2 logic must count the largest string of 0’s (bounded by 1’s) in a particular column P[n] of the memory, as well as keep track of the starting position of this longest string. The following pseudo-code explains the operation of port 2:

do in parallel for k from 0 to (N/2 - 1)
  
  do serially for Port 2 Pass from 1 to End_of_Time
    
    do serially for i from 0 to 1:
      
      n = 2k+i
      one[k] = 0, last[k] = 0
      zero_count_current[k] = 0, zero_start_current[k] = 0
      zero_count_high[n] = 0, zero_start_high[n] = 0
    
    do serially for i from 0 to (L-1):
      
      if (P[n] [i] == 1)
        
        one[k] = 1
      
      if (P[n] [i] * last[k] == 1)
        
        zero_start_current[k] = 1
        
        zero_count_current[k] = zero_count_current[k] + \( (P[n] [i] + one[k]) \)
        
        if (zero_count_current[k] > zero_count_high[n]

        if (zero_count_current[k] > zero_count_high[n]

        if (P[n] [i] == 1)
          
          zero_count_current[k] = 0
          
          last[k] = P[n] [i]

Fig. 5 shows the circuitry required. The counter is cleared when a 1 is read, and incremented when a 0 is read after any 1. The port 2 row address counter is sampled when there is a 1-to-0 transition. Both zero_count_high and zero_start_high are clocked by the comparator output, when the input makes a 0-to-1 transition. Note that some delay must be inserted before the clear input of zero_count_current to ensure that the data safely passes to zero_count_high, if required, before it is cleared.

The contents of zero_count_high and zero_start_high are available to a third memory port. Port 3 polls these registers and finds the time-slice with the highest zero-count (this is the optimum slicing instant). The highest zero_count_high, shifted right for a divide-by-2, is added to its associated zero_start_high to determine the optimum slicing level.

The zero_count_high registers are continually being updated based on the data read in each port 2 column, and port 3 can only produce a correct result if port 2 is frozen during its operation. The simple parallel–asynchronous method used by ports 1 and 2 cannot be employed. A more intelligent solution, making use of the port 2 column decoding, is required. Ports 2 and 3 are pipelined, and synchronized: while port 2 is analyzing even-numbered time-slices, port 3 is polling the zero_count_high registers of the odd-numbered slices; following that, port 2 looks at odd slices and port 3 polls the zero_count_high data just generated by port 2 for the even slices. Though each time-slice requires its own pair of zero_count_high and zero_start_high registers, the rest of the circuitry in Fig. 5 may be shared between adjacent time-slices, as implied in the port 2 pseudo-code.

The following pseudo-code represents port 3 operation. The k and i variables are identical to those in the port 2 pseudo-code, and the two listings may be combined (this is not shown).

do serially for Port 3_pass from 1 to End_of_Time
  
  hi = 0, st = 0, sample = 0
  do serially for i from 1 downto 0
    
    do serially for k from 0 to (N/2 - 1)
      
      n = 2k+i
      
      if (zero_count_high[n] > hi)
        
        sample = i
        
        hi = zero_count_high[n]
        
        st = zero_start_high[n]
    
  
  level = st + hi / 2

The most straightforward way ("normal-mode") to use this memory is to perform a read-modify-write cycle of port 1 for each data sample output by the A/D converter. Improved temporal resolution (higher N) can be achieved with a larger memory, running faster and dissipating more power. Improved voltage precision requires the memory to grow in its other dimension, again increasing power consumption. There is thus a continuum of trade-offs between precision, memory size, and power dissipation.

At a high enough data rate, the memory will not be able to operate at the desired N, regardless of how much circuitry and power is brought to bear on the problem. In this case, bandwidth can be further expanded by "sub-sampled–mode" operation, provided an A/D converter is available with timing resolution better than \( (N \times f_s)^{-1} \), independent of \( f_s \). A high-speed counter is required to operate in synchronism with the PLL, completing a count to N for each baud interval. This counter is sampled by the memory together with each A/D output word, to give a phase reference (address) for data input to port 1. Port 1 may then operate asynchronously of the data rate. Eventually, a complete eye pattern will form in the memory, and it may be analyzed by ports 2 and 3, as in normal–mode. Thus, it may be possible to run the memory at MHzs speeds and sample data at GHz rates. This strategy can also be used far below the maximum operating speed of the memory, purely as a power-saving measure. Sub-sampled–mode is more applicable in the case of channels whose characteristics vary slowly compared to the data rate (such as optical fibres) than in the case of rapidly-varying channels (such as in digital cellular radio).
Higher frequencies are also attainable without the use of subsampled-mode, by replicating port 1 circuits for concurrent read-modify-write accesses to multiple words. Using this "ultra-high-speed" architecture, a number of A/D samples could be added to the eye diagram in parallel for each port 1 clock cycle.

IV. PERFORMANCE ESTIMATION

In the previous sections, we have deliberately made no mention of circuit implementation details, attainable operating speeds or power dissipation. This is because the technique that we have described is technology-independent. Where speeds of only a few MHz are needed, low-power CMOS memory design methods can be used [7]. Up to 200 MHz, BiCMOS technology may be employed in conjunction with a power-efficient self-timed synchronous architecture [8]. Beyond 200 MHz, ECL or GaAs technology can be employed. Sampled or ultra-high-speed operation further increases the operating frequency. Wherever dynamic core cells are preferable to static cells, they may be used without an explicit refresh scheme, since the algorithmic access pattern provides built-in refresh.

Table 2 presents estimates for operating frequencies, power dissipation and circuit area obtainable using state-of-the-art submicron fabrication technologies for a sample 16 x 16 eye memory (N = 16, L = 16). Normal-mode operation is assumed.

<table>
<thead>
<tr>
<th>Technology</th>
<th>f0 (MHz)</th>
<th>f2 (MHz)</th>
<th>P (W)</th>
<th>Area (mm²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>low-power CMOS</td>
<td>20</td>
<td>1.25</td>
<td>0.03</td>
<td>1.3</td>
</tr>
<tr>
<td>BiCMOS</td>
<td>200</td>
<td>12.5</td>
<td>0.50</td>
<td>2.0</td>
</tr>
<tr>
<td>ECL</td>
<td>500</td>
<td>31.3</td>
<td>1.50</td>
<td>4.0</td>
</tr>
<tr>
<td>GaAs</td>
<td>600</td>
<td>37.5</td>
<td>3.00</td>
<td>2.5</td>
</tr>
</tbody>
</table>

To demonstrate the speed-up compared to a processor-and-RAM combination, we note the parallelization of approximately 10 instructions (including memory fetches) in the port 1 inner loop. The parallel outer loop also results in an N-fold speed-up. Finally, we estimate that attainable clock rate is double for the OAAM, due to the reduction in bus delays and physically distributed circuitry. Thus, for any given fabrication technology, this technique allows a multiplication by 20 x N of the supportable data rate.

V. APPLICATIONS AND POSSIBLE EXTENSIONS

We believe that Mb/s and kb/s wireless applications are the ones most likely to benefit from an OAAM-based receiver. Important requirements for cellular or wireless telephones are compactness and low power, and both of these are satisfied by the proposed circuitry. Wireless LANs are a new application needing an accurate and adaptive means of digital data recovery. The technique can also be used for 2-dimensional complex signals, such as M/4-shifted QPSK, by employing separate memories for each orthogonal component.

Noisy channels are not limited to wireless media. Intersymbol interference, symbol dispersion and electromagnetic interference corrupt data transmitted along twisted-pair copper, coaxial cable, optical fiber, or any of a number of other media. Here, high speed operation is as vital as high accuracy, and the technique of this paper is again applicable. GaAs technology and sub-sampled-mode enable operation at OC-12 rates (622 Mb/s).

An extension of this technique allows the analysis of m-level (m > 2) signals. Port 2 must then be augmented to store the highest zero-counts (and their starting levels) in m distinct bands, allowing determination of m sampling levels, one for each eye opening.

Functional enhancements are also possible. An extra adder associated with port 3 could keep track of the total number of zeros within the eye opening; this "area" metric is a figure of merit for the eye. By adding more logic to ports 2 and 3 and observing the patterns of 1's (rather than 0's), jitter and noise may be analyzed. In fact, computing first moments of jitter histograms provides an alternate means of determining the optimum sampling instant; it is especially useful if the eye diagram is asymmetric in time. If automatic gain control is implemented in the form of feedback between the eye memory and the A/D converter, L can be decreased, leading to reductions in memory size, area and power.

VI. CONCLUSIONS

This paper proposes a means of performing the data recovery function in digital hardware, allowing better accuracy than analog techniques and higher speeds than digital software methods. The Orthogonal Algorithmic Access Memory (OAAM) is the central component of this circuitry. This logic-enhanced memory structure provides high throughput due to inherent parallelism, and performs the same function as a combination of standard RAMs and external processing circuits at a much higher speed, in a smaller area, dissipating less power. The orthogonal nature of the memory enables bit-serial word-parallel operations on data which are written in standard bit-parallel word-serial fashion.

The approach is technology-independent. This, combined with a range of operating modes, allows for a continuum of trade-offs in performance versus cost. At the high end, data rates approaching 1 Gb/s are possible. The circuitry can be enhanced to allow elaborate channel diagnosis or handling of complex or multivalued data.

REFERENCES