ARCHITECTURE OF A FULLY DIGITAL CDR FOR PLESIOCHRONOUS CLOCKING SYSTEMS

Eliyah Kilada$^{1,2}$, Mohamed Dessouky$^{1,2}$, Adel Elhennawy$^1$

$^1$ Ain Shams University, $^2$ Mentor Graphics Egypt
{Eliyah_Kilada,Mohamed.Dessouky}@ieee.org

ABSTRACT
This paper describes a design of a fully digital clock and data recovery (CDR) system with plesiochronous clocking. Besides the well known advantages of digital implementations over analog ones in terms of robustness against process and temperature variations, scalability, compactness and low cost, the system also enjoys many features. It can withstand an input data cycle-to-cycle jitter up to ±37.5% UI. Data are obtained through digital correlation with the incoming symbol instead of ordinary sampling at the middle of the eye pattern, which improves BER. Furthermore, it needs only, at worst, three preamble bits to get into lock. It is insensitive to long runs of transition-free data patterns. The extracted data clock is not shifted as long as input data jitter is small (typically less than ±12.5% UI), thus, minimizing jitter in the extracted data clock. Besides, the extracted clock has a 50% duty cycle.

Index Terms— Clock and data recovery (CDR), clock multiplication, phase locking, jitter filtering.

1. INTRODUCTION
Serial links are widely used at the peripheral of many ASIC’s, specially after the failure of parallel busses at very high speeds. For this reason, effective CDR solutions must be implementable in the cheapest of digital process technologies, and easily ported across multiple technologies and speed targets [1]. Semi-digital implementations have been reported [1]-[6]. However, mixed signal blocks could not be completely avoided, specially the Digital to Phase Converter (DPC).

This paper presents a novel architecture of a Fully Digital CDR (FD-CDR) for plesiochronous clocking systems. Section 2 describes the theory of operation of the FD-CDR. The main building blocks are listed in Section 3. Some design issues are discussed in Section 4. Section 5 shows simulation results.

2. THEORY OF OPERATION
Rather than replacing each analog component in the traditional CDR with a digital one as in [1], data recovery is done based on a digital correlation rather than simple sampling at the middle of the eye pattern. Besides, the extracted clock shifting decisions are taken by a smart finite state machine (FSM).

The proposed CDR, essentially, generates 8-different phases of the data clock (referred as DataClk) with the nominal bit rate. One of these phases is elected by the FSM to be the locked one. While the system is in lock state, it monitors the input data jitter. According to the sign and magnitude of the jitter, the FSM selects the proper phase.

The same generated different phases are also used to sample the incoming data. Thus eight samples per bit period are, typically, obtained. The eight samples are used by the FSM to decide the value of the input data bit.

To maximize the operating frequency, WindowClk is introduced. Its frequency is half the bit rate. The sampling flip flops are clocked with the different phases of that clock. Mapping between WindowClk and DataClk are handled inside the system as will be shown later.

3. ARCHITECTURE
The theory of operation is best illustrated by tracing the different signals flow through the FD-CDR. Figure 1 shows the CDR block diagram. It is composed of seven main building blocks:

3.1. NPhasesGen
This block takes the MasterClk as an input. MasterClk is 4x of the bit rate. NPhasesGen is responsible for generating 16 different phases of the WindowClk (referred as NPhases) as shown in Figure 2. The WindowClk frequency is half the bit rate.

The delay between each two successive WindowClks is half the MasterClk period. NPhasesGen is implemented as 16-stage shift register with successive pairs of positive edge and negative edge flip flops that are clocked by the MasterClk. The FD-CDR is assumed to lock at one of these 16-different phases. Obviously, locking tolerance is determined by the MasterClk frequency.

3.2. ClockSelector
This block implements the Digital-to-Phase converter function. It takes the 4-bit Sel_d (i.e., Select delayed) signal and chooses the corresponding WindowClk phase as follows:

$$\text{WindowClk} \leftarrow \text{NPhases} (\text{Sel}_d)$$

Sel_d is a delayed version of the Sel (i.e., Select) signal. The Sel signal, as will be shown later, carries the actual phase information in the system.

Besides, ClockSelector also generates NPhasesSamp, which are 8 subsequent NPhases that will be used in sampling the input data.
### 3.3. Clock2XGen

It generates the DataClk (whose frequency is the same as bit rate) from the selected Windowclk (whose frequency is half bit rate). A synchronous delay block of half bit period is employed here to guarantee 50% duty cycle of the extracted DataClk.

### 3.4. Sampler

The 8-NphasesSamp clocks sample the incoming data through 8-dual edge flip flops. Effectively, this block generates 8-subsequent samples of the input data during the bit period. This is similar to the one used in the improved bang-bang phase detector in [7], [8].

### 3.5. DigCorrelator

Digital correlation is being done between the 8-data samples of each bit and the data symbol coefficients. For the case of NRZ line coding, this block reduces to a summing circuit, that generates Sum signal that carries the number of the HI samples (i.e., samples that are ONE). Besides, DigCorrelator also produces UpDown signal that indicates the location of the HI samples within this window. UpDown is HI if the Sum of the first 4-samples are greater than that of the last 4-samples, and LO otherwise. These two signals (i.e., Sum and UpDown) provide the next stage (i.e., the MotherControl) with the information required to determine if the incoming data bit is ONE or ZERO, as well as to select the appropriate clock for phase alignment.

### 3.6. MotherControl

This is the core of the FD-CDR. MotherControl is a FSM that is clocked by DataClk. It takes Sum and UpDown signals as inputs and generates Sel, DataOut and Lock signals. The operation of the MotherControl is best illustrated by the following example:

The vertical solid lines in Figure 3 correspond to occurrence of DataClk positive edges (i.e., the FSM is clocked at these times).

The first positive edge of DataClk (after the receiver reset) should bring the FSM into the reset state. Obviously, no
decision can be taken at this state since there are no available sample information. On the next positive edge of DataClk, the FSM is informed by the DigCorrelator that the current data bit contains 6-HI samples, and UpDown is HI (since all the first 4-samples are HI, while only 2 of the last 4 samples are HI). Based on this information, the FSM decides that the current data bit is ONE. It also detects that the internal extracted clock is late with respect to the transmitter clock by about 2 sample periods. As a result, the FSM reduces the current value of Sel signal by 2. The vertical dashed line corresponds to the instance when the third positive edge of DataClk would have occurred if no actions were taken. Obviously, the DataClk shift made by the FSM will result in exact alignment at the third positive edge of the DataClk. At this instance Sum is zero (which corresponds to perfect ZERO), and the extracted clock in the receiver is aligned with that of the transmitter.

The Lock signal is asserted when the receiver is confident about its relative extracted clock phase with respect to the transmitter clock.

Typically, the MotherControl has five distinct states as shown in Table 1.

### 3.6.1. RST
The FSM gets into the RST state on the first positive edge of DataClk just after the receiver resets. Obviously, no decisions can be taken here since the MotherControl doesn’t have enough sample information at that time.

### 3.6.2. WAIT_EDGE
In this state the MotherControl is waiting for a data transition to collect the required information for shifting the receiver clock. As shown in Table 1, for any value of Sum between 1 and 7 (except 4), the system can go into lock immediately and produces the proper DataOut value. However, if Sum is 4, the MotherControl can’t determine the value of this incoming bit and it increases the phase of the current DataClk by 4 sample periods.

Suppose that, initially, the system receives a stream of successive ONE’s. In this case, Sum will be greater than 7. The FSM confirms that the received data bit is ONE, however, Lock signal is LO, because there’s no enough information about the transmitter clock. In this case, the system goes into TRACK_1 state. A similar scenario for TRACK_0 state for initial successive ZERO’s.

### 3.6.3. TRACK_1
In this state the system is receiving an initial stream of successive ONE’s as described above. It does leave this state as soon as an input data transition occurs.

### 3.6.4. TRACK_0
In this state the system is receiving an initial stream of successive ZERO’s as described above. It does leave this state as soon as an input data transition occurs.

### Table 1. MotherControl states

<table>
<thead>
<tr>
<th>Current State</th>
<th>Inputs</th>
<th>Clocked Outputs</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAIT_EDGE</td>
<td>&lt;= 1</td>
<td>X +0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0 -2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1 +2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>X +4</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0 +3</td>
<td>1 1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0 +2</td>
<td>1 -2</td>
</tr>
<tr>
<td></td>
<td>&gt;= 7</td>
<td>X +0</td>
<td>0 0</td>
</tr>
<tr>
<td>TRACK_1</td>
<td>&lt;= 1</td>
<td>X +0</td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0 -2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0 -3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>X +4</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0 +3</td>
<td>1 1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0 +2</td>
<td>1 -2</td>
</tr>
<tr>
<td></td>
<td>&gt;= 7</td>
<td>X +0</td>
<td>0 0</td>
</tr>
<tr>
<td>TRACK_0</td>
<td>&lt;= 1</td>
<td>X +0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0 -2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0 -3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>X +4</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0 +3</td>
<td>1 1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0 +2</td>
<td>1 -2</td>
</tr>
<tr>
<td></td>
<td>&gt;= 7</td>
<td>X +0</td>
<td>0 0</td>
</tr>
<tr>
<td>ACQ</td>
<td>&lt;= 1</td>
<td>X +0</td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0 -2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0 -3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>X +4</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0 +3</td>
<td>1 1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0 +2</td>
<td>1 -2</td>
</tr>
<tr>
<td></td>
<td>&gt;= 7</td>
<td>X +0</td>
<td>0 0</td>
</tr>
</tbody>
</table>

### 3.6.5. ACQ
This is the acquisition state of the system where the FSM is tracking any shift in the input data transitions as depicted in Table 1.

### 3.7. NegSelBuffer
When a positive edge of DataClk occurs, the FSM is clocked, and the Sel signal is changed accordingly. The ClockSelector will change the WindowClk based on the new value of the Sel signal, which, in turn, will modify DataClk. This loop is unstable by nature and will cause glitches in DataClk as well as undesired transitions in the MotherControl.

To break up this unstable loop a buffer is inserted to delay the Sel signal, so that ClockSelector shifts the WindowClk based on a delayed version of the Sel signal (i.e., Sel_d).

How much delay is required? The maximum shift forced by the MotherControl on DataClk is 4 sample periods (i.e., half bit period). Therefore, if the Sel signal is delayed by half bit
period before it takes effect on DataClk, then no glitches can occur on DataClk. This is done by buffering Sel signal with the negative edge of the DataClk to produce Sel_d. ClockSelector, in turn, works on Sel_d.

4. DESIGN ISSUES

Obviously the bottleneck of the design is the MasterClk frequency. In details, to get a resolution of 1/8 bit period, a MasterClk of 4x of the bit rate is required. However, it should be clear, that this clock is only used in NPhasesGen to generate the different phases of WindowClk. On the other hand, all other system blocks are working on the bit rate or even half-bit rate.

5. SIMULATION RESULTS

The system has been stimulated by different test benches. Two flags are generated to indicate the relative periods of the current input data bits, namely, Long1Flag and Long0Flag. They are ±1, ±2 or ±3 if the current input data bit is ONE (or ZERO) and is longer (or shorter) than the nominal UI by ±12.5%, ±25% or ±37.5% respectively. For other signals definitions, refer to Section 3 of this paper.

5.1. Initial Acquisition

Initial acquisition operation is shown in Figure 4, and it is similar to that described in the example of the MotherControl operation in Figure 3. As stated before, the system needs, at worst, 3 preamble bits to get into acquisition. Regarding the shown situation, no input data bits are lost.

5.2. Tracking Jitter

In Figure 5, a sequence of “long ONE (+25%UI), nominal ZERO, short ZERO (-25%UI) and nominal ONE” has been applied to the system. As seen in the waveforms, the extracted DataClk expands and shrinks as required by the input data jitter amplitude and direction. Obviously, the system can recover this jittered input data pattern successfully (i.e., “1001”) without losing lock at switching times.

5.3. Tracking Frequency Drift

A test bench has been developed to test the system response in case of 100 ppm drift of the receiver (or transmitter) MasterClk frequency. With the 100 ppm drift, the extracted DataClk successfully tracks the drift. Typically, it makes one shift per 2500 bit periods. Even at the switching times, the data is correctly recognized and the system never gets out of lock under these conditions.

6. CONCLUSION

A design of a fully digital CDR system with plesiochronous clocking was presented. The FD-CDR employs a smart FSM to control the shift of the extracted clock. It can withstand an input data cycle-to-cycle jitter up to ±37.5% UI. It needs, at worst, three preamble bits to get into lock. It is insensitive to long runs of transition-free data patterns. Besides, the extracted clock has a 50% duty cycle. Furthermore, digital correlation is used to recover the data, which improves BER. Finally, the system features were confirmed by simulation.

7. REFERENCES