Abstract—This paper describes an FPGA implementation of a fully digital clock and data recovery system (FD-CDR) with plesiochronous clocking. The design utilizes 51 FF's only. It does require, at worst, 2 preamble bits to get into lock. The extracted clock is not shifted as long as input data jitter is small (typically less than ±12.5% UI), thus, minimizing jitter in the extracted clock. Typically, for small input data jitter, the extracted clock shows an rms jitter of 51 ps which is mainly due to the 35.6 ps jitter (rms) of the system 100MHz-master clock. Besides, it can withstand an input data cycle-to-cycle jitter up to ±37.5% UI without getting out of lock. Data are obtained through digital correlation with the incoming symbol instead of ordinary sampling at the middle of the eye pattern, which improves BER. It is insensitive to long runs of transition-free data patterns. Besides, the extracted clock has a 50% duty cycle.

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

I. INTRODUCTION

Serial links are widely used at the peripheral of many ASIC’s, specially after the failure of parallel busses at very high speeds. Semi-digital implementations have been reported [1]-[6]. However, mixed signal blocks could not be completely avoided, specially the Digital to Phase Converter (DPC). An architecture of a fully digital CDR (FD-CDR) has been suggested in [9].

This paper presents an FPGA implementation of the FD-CDR reported in [9]. Section II reviews briefly the theory of operation of the FD-CDR. Some design issues are discussed in Section III. Experimental results and performance summary are given in Section IV.

II. THEORY OF OPERATION

The theory of operation is described briefly in this section. The reader is encouraged to get a detailed description of operation in [9]. Fig. 1 shows the CDR block diagram. It is composed of seven main building blocks:

A. NPhasesGen

This block takes the MasterClk as an input. The MasterClk frequency is 4x of the bit rate. NPhasesGen is responsible for generating 16 different phases of the WindowClk (referred as NPhases). The WindowClk frequency is half the bit rate. The delay between each two successive WindowClks is half the MasterClk period. The FD-CDR is assumed to lock at one of these 16-different phases.

B. 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: WindowClk <= NPhases(Sel_d)

Sel_d is a delayed version of the Sel (i.e., Select) signal. Besides, ClockSelector also generates NPhasesSamp, which are 8 subsequent NPhases that will be used in sampling the input data.

C. Clock2XGen

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

D. 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.

E. 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. The Sum signal should carry the number of the HI samples (i.e., samples that are One). Besides, DigCorrelator also produces the 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.

F. 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 \textit{Sel}, \textit{DataOut} and \textit{Lock} signals. \textit{Sel}, which carries the actual phase information in the system, changes according to the phase misalignment between the selected phase of \textit{WindowClk} (or extracted \textit{DataClk}) and the transmitter clock. This phase misalignment information is provided by \textit{Sum} and \textit{UpDown} signals. The \textit{Lock} signal is asserted when the receiver is confident about its relative extracted clock phase with respect to the transmitter clock. \textit{DataOut} is determined based on \textit{Sum} value.

\subsection*{G. NegSelBuffer}

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

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

How much delay is required? The maximum shift forced by the MotherControl on \textit{DataClk} is 4 sample periods \cite{9} (i.e., half bit period). Therefore, if the \textit{Sel} signal is delayed by half bit period before it takes effect on \textit{DataClk}, then no glitches can occur on \textit{DataClk}. This is done by buffering \textit{Sel} signal with the negative edge of the \textit{DataClk} to produce \textit{Sel}_d. ClockSelector, in turn, works on \textit{Sel}_d.

\section*{III. DESIGN ISSUES}

\subsection*{A. MasterClk Frequency}

Obviously locking tolerance is determined by the \textit{MasterClk} frequency. In details, to get a resolution of 1/8 bit period, a \textit{MasterClk} of 4x of the bit rate is required which can be a very high frequency that causes implementation issues. However, it should be clear, that this clock is only used in NPhasesGen to generate the different phases of \textit{WindowClk} and in Clk2XGen to guarantee 50\% duty cycle of the extracted \textit{DataClk}. On the other hand, all other system blocks are working on the bit rate or even half-bit rate.

\subsection*{B. ClockSelector Glitches}

Since the intrinsic delays of the different \textit{Sel} bus elements through the ClockSelector block up to its output (i.e., \textit{WindowClk}) varies slightly, therefore, some glitches have been observed on \textit{WindowClk} during certain switching events in post place and route simulation and confirmed by measurements. To remedy this effect a buffer has been added at the ClockSelector output.
(a) MasterClk input jitter is 35.6 ps (rms).

(b) Extracted DataClk jitter is 51 ps (rms).

Fig. 2. MasterClk input and extracted DataClk jitters in case of an input data jitter of less than ±12.5% UI.

(a) Tracking a long One.

(b) Tracking a short Zero.

Fig. 3. Tracking an input data cycle-to-cycle jitter of ±12.5% UI.

(a) USB 2 standard compliance sequence spectrum.
   Center: 25 MHz, 4.95 MHz/div

(b) DataClk spectrum.
   Center: 25 MHz, 638 KHz/div.

Fig. 4. DataIn and DataClk spectrums.
IV. EXPERIMENTAL RESULTS AND PERFORMANCE

A. Area

The system has been downloaded to Virtex4-XC4VFX12. It does utilize 63 slice registers (51 FF’s and 12 latches) and 186 4-input LUTs. This is a 1% utilization of the FPGA resources.

B. Tracking Jitter

1) Jitter Less Than 12.5% UI:

As described in the MotherControl operation [9], DataClk is not shifted as long as the input data jitter is less than ±12.5% UI. In this case, DataClk jitter is mainly due to the MasterClk input jitter. This has been confirmed experimentally in Fig. 2. While MasterClk jitter is 36 ps (rms), DataClk jitter is only 51 ps (rms). The added jitter is mainly due to the inherent clock division and selection in the system.

2) Jitter More Than 12.5% UI:

The system is able to withstand cycle-to-cycle jitter up to ±37.5% UI. Fig. 3 shows a test bench that has been synthesized to verify the tracking range. Two flags are generated to indicate the relative periods of the current input data bits, namely, Long1Flag and Long0Flag. They are ±1 if the current input data bit is One (or Zero) and is longer (or shorter) than the nominal UI by ±25% respectively. For other signals definitions, refer to Section II of this paper.

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 jitted input data patterns successfully without losing lock at switching times.

C. Tracking Frequency Shift

A test bench has been developed (on the simulation level) to verify 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.

D. Extracted Clock Spectrum

Using 100MHz MasterClk (i.e., data rate of 25Mbps), the FD-CDR has been stimulated by the USB 2.0 standard compliance pattern [10]. Fig. 4 shows the spectrum of both DataIn and DataClk. Obviously, the input NRZ data pattern has no peaks at 25MHz. The extracted clock (i.e., DataClk) clearly locks on 25MHz.

E. Transition-Free Data Pattern

The proposed FD-CDR is insensitive to Transition-Free Data Pattern. Fig. 5 shows a shot of long succession of Zeros followed by One. Clearly, the system is able to recover the data correctly in this case.

F. Extracted Clock Duty Cycle

The proposed FD-CDR guarantees 50% duty cycle of DataClk in lock conditions except in Sel switching times.

V. CONCLUSION

An FPGA implementation of a fully digital CDR system with plesiochronous clocking was presented. The FD-CDR enjoys superior jitter performance especially when the input data jitter is less than ±12.5% UI. It can withstand an input data cycle-to-cycle jitter up to ±37.5% UI. It needs, at worst, two preamble bits to get into lock. It is insensitive to long runs of transition-free data patterns. Besides, the extracted clock has 50% duty cycle. No specific test has been done to measure the BER. However, the authors believe that using digital correlation instead of one mid-point sampling should enhance BER. Finally, the system features were verified by measurements.

REFERENCES