SIMULATION MODEL OF DIGITAL CLOCK AND DATA RECOVERY FOR STRONGLY DISTURBED SIGNALS

Michal Kubíček
Doctoral Degree Programme (1), FEEC BUT
E-mail: xkubic18@stud.feec.vutbr.cz

Supervised by: Jaromír Kolouch
E-mail: kolouch@feec.vutbr.cz

ABSTRACT

The paper describes VHDL-AMS simulation model of a digital link (with a signal source) together with "software" clock and data recovery module [1] and common recovery circuit incorporating a PLL. Performance of both methods is compared and discussed. Models were created to help to improve the software recovery method.

All simulations were performed in the Mentor Graphic's SystemVision 4.4 environment using VHDL-AMS [2] models of signal source, data path and recovery circuits. The software recovery algorithm is written in synthesizable subset of VHDL and can be directly used as a part of an FPGA design.

1. CLOCK RECOVERY METHODS

Data recovery is necessary step in serial data stream processing. Serial links usually have only one channel for data and no corresponding clock is transmitted. Signal on the receiver side must be sampled by clock signal that must be as close as possible to the original one in both phase and frequency. To get such clock signal, clock recovery is necessary. Once the recovered clock is available in the receiver, incoming data stream can be sampled (data recovery).

There are basically two methods of clock recovery. The first one incorporates a phase locked loop (PLL) and is usually implemented as a hardware part (part of some bigger chip or stand-alone component [3]). It is sometime called hardware recovery method. The second one is based on an algorithm that can be implemented into an FPGA. This attribute leads to calling this method software recovery.

Both of these methods were successfully modeled and are compared in the paper. Nevertheless, there are some restrictions of simulation findings. The hardware recovery method cannot be modeled exactly, only general model is simulated (there are many types of HW recovery circuits, based on different architectures and more or less sensitive to jitter and noise). For the present, no parasitic attributes of digital circuits were modeled (finite slew rate, gate delay, inaccurate comparators threshold).

2. SIMULATION MODELS

All models are described using the VHDL-AMS language, whereas software recovery method is written in pure VHDL and is ready for FPGA implementation as a part of a user design. Data rate was set to 155.52 Mb/s, nevertheless, today's FPGAs are capable to manage software recovery at data rates up to 600 Mb/s.

2.1. MODEL OF THE TEST SIGNAL SOURCE

The source of a digital test signal is a simple 23bit LFSR counter (taps 18 and 23, XNOR type). It can be easily modified using its generic parameters. Output of the LFSR counter is a pseudorandom sequence that is processed through a model of a channel (see Fig. 1:). Amplitude of the digital signal in analog domain was set to 400 mV. To simulate non-ideal signal source, jitter was added to the LFSR clock signal (1% and 10% respectively).

To get a more realistic model, noise was added to the test signal. The amplitude of noise was varied during simulation (200 mV, 300 mV and 400 mV; +/- amplitude is the maximum/minimum level of the noise signal) to determine the sensitivity of the clock recovery methods to it. The channel itself is modeled as a second order low pass filter ($Q = 0.7$, $F_P = 0.5 \cdot F_{CLK} \approx 80$ MHz).

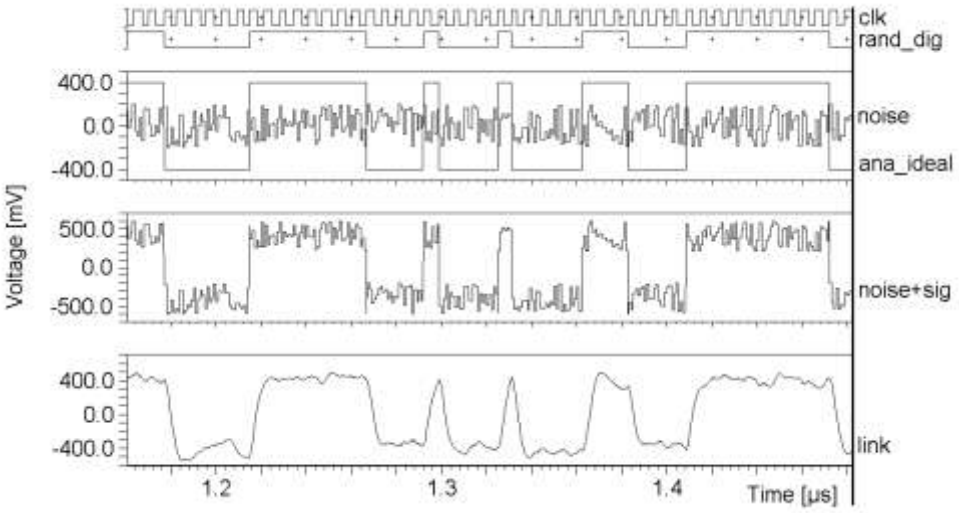

Fig. 1: The signals of the test signal source block; "clk" is a digital clock signal (with significant 10 % jitter), "rand_dig" is a pseudorandom digital (data) signal. The "noise" is an analog noise signal (amplitude 200 mV), "ana_ideal" is analog data signal. The "noise+sig" is an analog data signal with noise and the last one ("link") is signal on the channel output (processed by the low pass filter).

In Fig. 2: you can see the data eye diagrams corresponding to different settings of the testing signal source and channel. There is only a small difference between diagrams with source clock jitter of 1 % and 10 %, additive noise is dominant factor in the simulation. Both jitter and noise (and also channel bandwidth) can be changed to modify link model behavior to correspond to a particular link with specific (known) parameters. Such estimate model can help to modify the software recovery algorithm to fit best to the link.

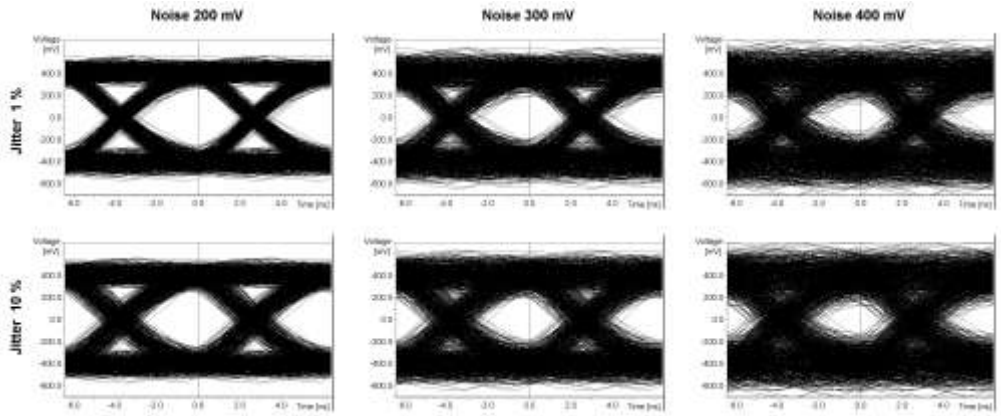

Fig. 2: Six data eye diagrams corresponding to six different test signals used to test the clock recovery models.

2.2. Model of the Hardware Recovery Circuit

To model the hardware recovery circuit, a simple model based on an edge detector and a band pass filter was used. The edge detector is made of XOR gate. The signal from the link is connected to the first gate input directly and to the second one through a delay element. Whenever an edge occurs on the link, a pulse is generated on the detector output. Its length is equal to the delay element latency and should be equal to half of the clock period. This impulse is processed by the band pass filter (its central frequency is equal to the recovered clock frequency). Thus extracted first harmonic is then processed by the PLL to reach high frequency stability.

The data signal from the link is digitalized and delayed to get in the correct phase relationship with the recovered clock signal (rising edge of the clock signal appears in the middle of the data bit; this simulates output of a real recovery circuit).

2.3. Software Recovery

A simple version of software recovery module is described in [1]. This recovery circuit was previously implemented into a Spartan-3 FPGA as a part of bit-error rate tester. Because poor performance on higher data rates was observed, the module was modeled using VHDL-AMS to find out the reason of improper functionality.

Software recovery method actually doesn't recover the clock signal. It uses clock generated by local oscillator that runs approximately on the frequency of the data rate. As the frequencies of receiver and transmitter oscillator aren't equal, there can be none, one or two bits valid per local clock period (see [1]). There are four instants (A to D) derived from local clock signal that are used to sample the received signal. A decision algorithm chooses the right domain (sampling instant) according to the position of the edge between sampling instants A to D, i.e., when transition between A and B domains is detected, samples from domain C are chosen as output in order to sample the data in the middle of the data eye diagram.

As there can be 0, 1 or 2 bits valid in a single clock period, there are two double-bit signals on the module output. The first one is data output and the second is data valid signal that validates corresponding data bits. Following block must be able to deal with such interface (for example, a FIFO structure).

3. SIMULATION

Six test signals (containing different amount of noise and jitter) were applied to both software and hardware recovery modules. Simulation time was set to 40 µs which is equal to about 6 200 transmitted bits. Results of simulation are summarized in Tab. 1. Simulation waveform of worst signal is displayed in Fig. 3:.

	jitter	
noise	1%	10%
200 mV	0 / 0	0 / 2315
300 mV	0 / 156	2 / 2007
400 mV	7 / 773	12 / 2050

Tab. 1: Number of detected error bits in dependence on jitter and noise; hardware / software recovery.

High number of errors detected on software recovery module is not a real number of incorrectly extracted bits. The software recovery module features two types of errors. The first one occurs when detected bit is opposite to the correct one. This error is typical also for the hardware recovery method. The second type of error is specific for software recovery and is caused by incorrect extraction of the *number* of valid bits.

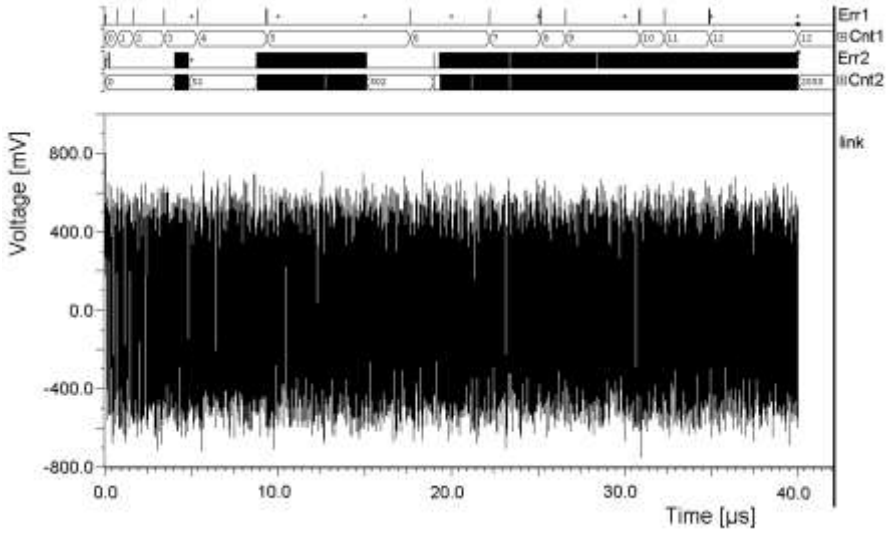

Fig. 3: Simulation results (10 % jitter, 800 mV noise): incorrect data extraction of basic SW recovery module.

Measurement / detection of wrong bits is based on LFSR counter. Counter in the receiver is synchronized on the incoming data stream at the beginning of the simulation and then it runs separately as a source of reference data. When one bit (or more) is extracted incorrectly (1 bit instead of 2 bits, etc.), synchronization of reference LFSR counter is broken and high error rate is declared (see Fig. 3:). The receiver (its part behind the recovery module) usually cannot correct such error (compared to single bit inversion when an error correction algorithm can help) hence this problem must be solved in the recovery module. In the case of the hardware recovery method, every bit was extracted. No extra (= error) bit was inserted into recovered data stream, none was omitted. Several bits were extracted incorrectly (value 1 instead of 0 or vice versa), but number of them was correct.

Error in the bit extraction of the software recovery module is caused by its portion for determining best sampling domain (A to D) according to actual edge position. Normally,

number of extracted bits changes when domain sampling interval turns from D to A (no bit is extracted, local clock is faster then transmitters one) and from A to D (vice versa, 2 bits are extracted). This algorithm works well until improper change of sampling interval is performed without touching the interval A (like directly from D to B; see Fig. 4:).

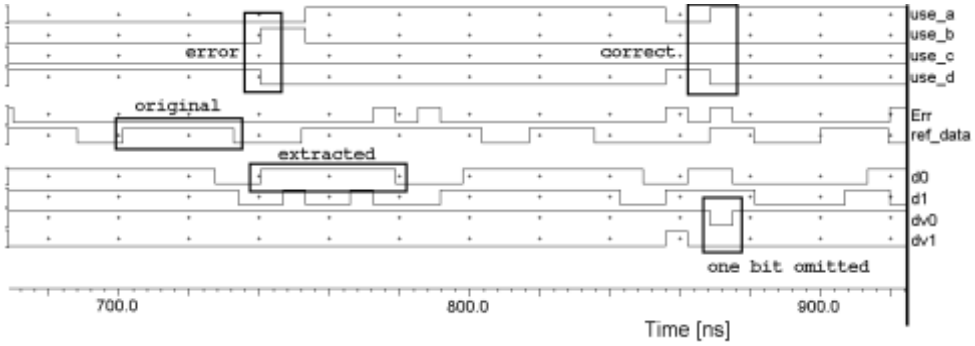

Fig. 4: Simulation results: improper and proper data extraction of basic SW recovery module; Change of domain from D to B was performed (domain is selected according to signals use_a to use_d). Consequently one invalid bit was inserted into recovered data stream d0 (signal dv0 should have disclaimed one bit).

This problem can be fixed by adding suitable restrictions to the decision algorithm. The restrictions should disable change of sampling interval to the opposite one and should force the algorithm to track long-time frequency trend except of jitter.

4. CONCLUSION

The models of the software recovery circuit and the source of realistic test signal were successfully modeled and simulated. On the basis of the simulation results, software algorithm will be improved in order to be able to handle strongly disturbed signals safely. Resource (silicon area) optimal hardware implementation is going to be secondary in this case, safe operation is primary.

ACKNOWLEDGEMENTS

Research described in the paper was financially supported by the Czech Grant Agency under grant No. GA102/05/0732 "Computer Modeling and Synthesis of Digital and Mixed Analog/Digital Systems", and grant No. GA102/05/0571 "Methods for Availability Improvement of Free Space Optics Communications", and Research Programme of Brno University of Technology MSM0021630513 "Electronic Communication Systems and New Generation Technology (ELKOM)".

REFERENCES

[1] SAWYER, N. Data Recovery. *Xilinx application note XAPP225 (v2.5),* 2005.

[2] ASHEDEN, P. J., PETERSON, G. D., TEEGARDEN, D. A. *The System Designer's Guide to VHDL-AMS*. Elsevier Science, San Francisco (California, USA), 2003. ISBN: 1-55860-749-8.

[3] SY87700V Clock and Data Recovery. *Micrel's data sheet*, 2003.