2011 International Conference on Computer Science and Information Technology (ICCSIT 2011)
IPCSIT vol. 51 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V51.57

Mini SerDes Based on Economic FPGA

Zhuo Bi[a+], Zhen Wang[b] and Meihua Xu[a, b]

[a]School of Mecha Engin and Automation, Shanghai Univ,Shanghai,200072, China
[b]Micro-electronic R&D Center, Shanghai Univ,Shanghai,200072, China

Abstract. SerDes has been widely used in high-speed serial interface in the past years, and it is often implemented as the form of ASIC and ASSP (application specific standard product, which is an integrated circuit that implements a specific function that appeals to a wide market). Some FPGAs contained SerDes inner have been released, but they are high-ends and expensive to the most researchers and most FPGAS have no SerDes inner. So mini SerDes based on economic FPGA has been proposed in this paper. The mini SerDes can implement verification and communication based on SerDes in economic FPGA. The synthesis and analysis results by QuartusII8.1 show that this mini SerDes has low power consumption and reaches the max frequency which is limited by the PLL in the economic FPGA.

Keywords: SerDes, high-speed serial interface, FPGA

1. INTRODUCTION

SerDes is a point to point communication technology used TDM, which is widely used in WLAN and LAN. Originally, SerDes is used in optical fiber communication as a critical part of information superhighway. With the integration of computer and communication, SerDes has a broader perspective. High-speed serial interface based on SerDes is becoming a general interface standard. In the recent years, many standards organizations have draft or are drafting high-speed serial interface standard range from 1G to 10G and the mainstream is 1~6G. The first generation is 2.5G/3.125G and the second is 5G/6.25G. These chips can implement by use the CMOS technology.

However, most SerDes is implemented as the form of ASSP and ASIC. In the past years, some FPGAs contained SerDes have appeared which can instead of independent SerDes. These FPGAs are expensive because that only high-end FPGAs contains SerDes inner. It attracted many corporations such as the Lattice semiconductor corporation, the xilinx corporation and the altera corporation and so on. Some FPGAs contained SerDes have been released such as LatticeECP2M, LatticeECP3 and cycloneIV GX and so on. However these products are expensive to the most researchers and most FPGAs have no SerDes inner.

Considering this, mini SerDes based on economic FPGA is proposed in this paper. The advantage of the proposed mini SerDes is that the the cost and the power consumption are lower and verifications and communications based on SerDes can be implemented by only with economic FPGA despite that the performance of mini SerDes cannot be campared with the SerDes in the high-end FPGAs.

In this paper, the overall of the SerDes is provided in section II. CDR is discussed in section III. 8B/10B Encoder/Decoder is proposed in section IV and then some relative contents are briefly introduced in section V. Finally, section VI is conclusion.

2. SERDES OVERALL

+ Corresponding author. Tel.: +86-21-5633-1206
E-mail address: zhuo.bi@shu.edu.cn.

2.1. Architecture of Mini SerDes

SerDes is a kind of serial transceiver. This mini one mainly contains four parts: CDR, 8B/10B Encoder/Decoder, FIFO and serial-parallel converter. Figure.1 shows the architecture of the mini SerDes.

In figure.1, the 8B/10B Encoder/Decoder has the function of detecting the error of transmission and can make sure the data transmission accurately. The CDR is a critical module for this mini SerDes. These two modules will be introduced detailed latter.

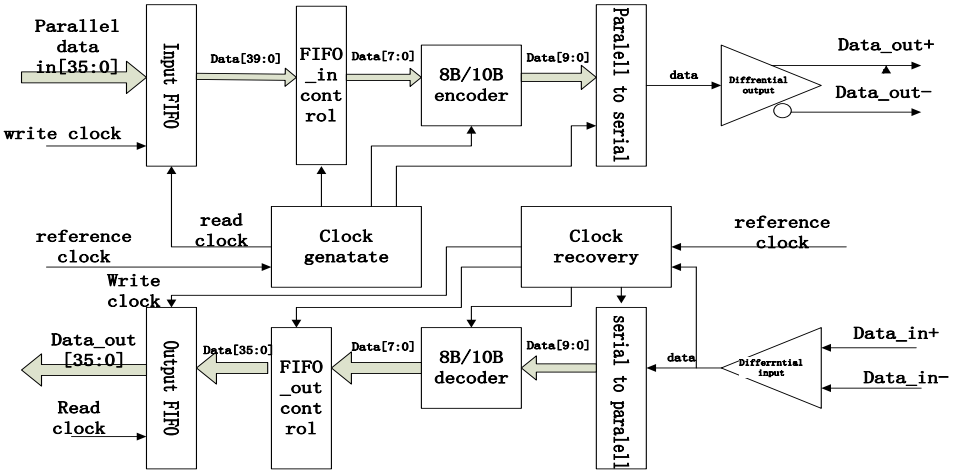

Figure.1 Architecture of SerDes

2.2. Timing Simulation

Figure.2 shows the timing simulation of Serializer while Figure.3 is Deserializer. In the Deserializer, we added the Encoder in it to make its inputs the same as the Encoder to make simulation convenient. Comparing these two figures, we can know that the SerDes works well.

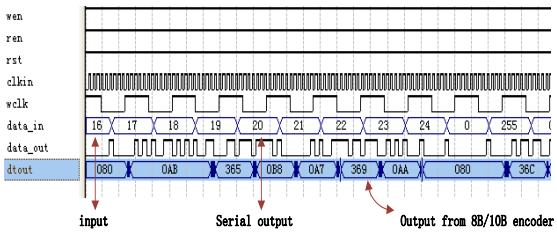

Figure.2 timing simulation of serializer;

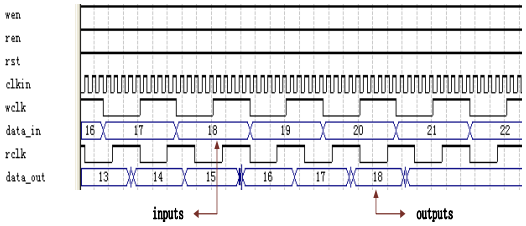

Figure.3 timing simulation of deserializer

3. CDR

3.1. Implementation Methods

The CDR is the key technology of high-speed transceiver. Generally, there are two methods to design CDR. One method is PLL and the other is over-sampling. The PLL method uses feedback loop to make the trace of clock edge of receiver alignment with the edge detected from input data bit stream. The over-sampling method uses local clock which is faster several times than the system clock to sample the data several times at a data bit width and then recover the correct clock and data according to a certain decision algorithm. The over-sampling method is mainly used by low-speed communication such as communication between computers or computer peripherals. Compared with the PLL method, the over-sampling method cannot eliminate jitter, while increase additional jitter. However, the synchronous speed of PLL method is limited by the synchronous set-up time. Compared with the PLL method, the over-sampling method satisfies some requirements of fast synchronization to receiver. Besides, the resources and frequency of the PLL in FPGA is limited. From above all, the over-sampling method is selected in this paper.

3.2. Time or Spatial Over-sampling

Two ways can be used for over-sampling. One is time over-sampling and the other is spatial over-sampling. The time over-sampling sample the data bit stream used local reference clock which frequency is several times faster than the clock in transceiver. The spatial over-sampling sample the data bit stream with

local multi-phase clocks and the factor depends on the phase. Obviously, the spatial over-sampling is more suitable for the high-speed receiver. The CDR based on spatial over-sampling is described in reference [1]. According to the limited frequency of digital PLL in this paper, the CDR is referred and improved. Six clocks are instead of the three clocks to improve accuracy. Figure.4 shows the structure of CDR. In this CDR, two PLL devices in FPGA are used to generate six clocks which have the same frequency and different phase. Then, according to the control signals, these clocks are stitched to make sure the clock sample the valid data bit. These signals are generated by phase detector and then fed into the up-down counter for some delay and finally exported to control.

3.3. Glitch free

In this design, clock switch is a critical module in this mini SerDes. An implementation method is using an AND-OR type multiplexer logic. But it can generate glitch. Figure.5 shows how a glitch is generated.

As for the glitch problem, a method is used to avoid glitch generated by switching clock in reference [2]. In the Figure.7, a negative edge triggered D flip-flop is inserted in the selection path for each of the clock sources. The selection control is registered at negative edge of the clock and enable only after other clock is de-select first. This method provides excellent protection against glitches at the output. Registering the select signal at negative edge of the clock guarantees that no changes occur at the output while either of the clocks is at high level, thus it can protect the output clock. Feedback from one clock's selection to the other enables the switch is waiting for de-selection of the Current Clock before starting the propagation of the Next Clock, avoiding any glitches. And the result is shown in Figure.6.

In this paper, the number of clocks is extended from two to six. There are total six control signals: en0, en1, en2, en3, en4, en5. In the six control signals, only one of them can be considered as high level and others as low level.

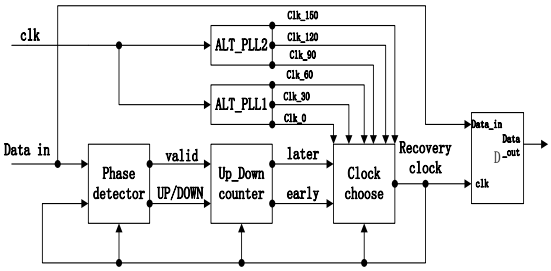

Figure.4 structure of CDR;

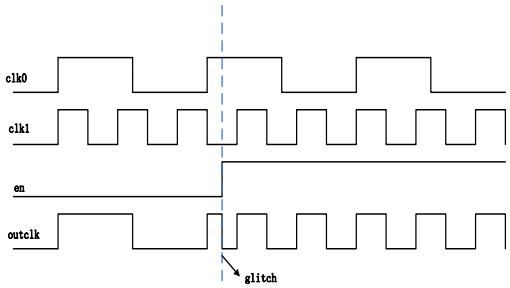

Figure.5 generation of glitch by clock switch

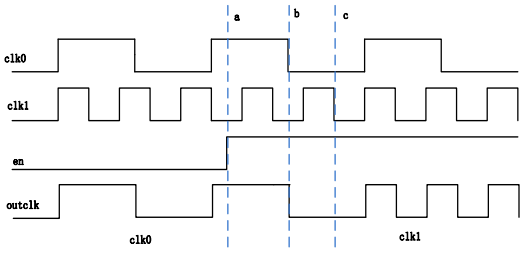

Figure.6 glitch-free;

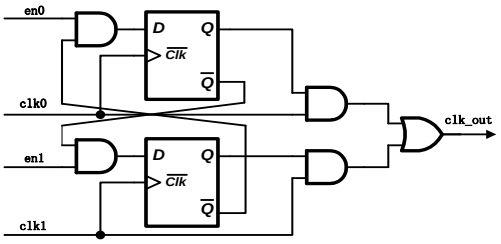

Figure.7 logic of avoiding glitch

3.4. Timing simulation

Figure.8 shows the Timing simulation of CDR. From the simulation result, we can know that the data can be recovered accurately. From the above, it shown that the result has obvious effect.

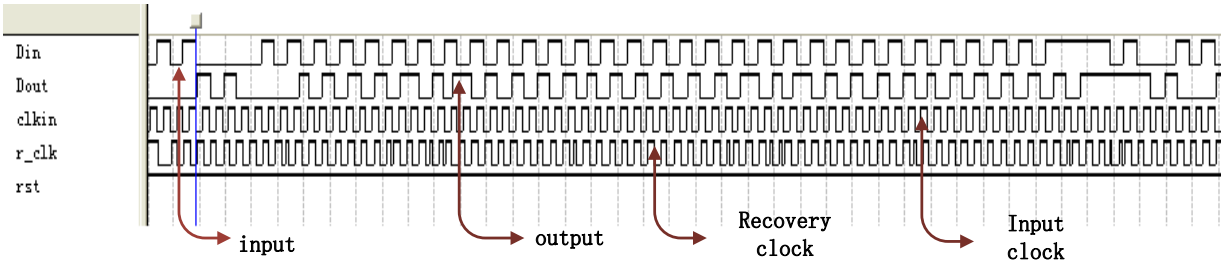

Figure.8 Timing simulation of CDR

4. 8B/10B encoder/decoder

4.1. 8B/10B Coding

8B/10B coding is a method that converts 8-bit symbols to 10-bit symbols code. There are total 256 data symbols and 12 control symbols denoted by Dx.y and Kx.y. In telecommunications, 8b/10b is a line code to achieve DC balance and bounded disparity. This coding provides enough state changes to allow reasonable clock recovery. This means that the difference between the count of '1' and '0' in a string of bits (at least 20) is no more than 2, and that there are not more than five '1' or '0' in a row. This feature helps to reduce the demand for the lower bandwidth limit of the channel necessary to transfer the signal. All of transmitted 10 bit symbols must be one of the three states which are '±2', '0' and'+2'. '+2'means two more '1' and '-2' means two more '0'. Others have equal number of '1' and '0'. The difference between the number of '1' transmitted and the number of '0' transmitted is always limited to ±2 to achieve DC balance.

DC balance which is the most important feature of 8B/10B coding can make the driver of AC-coupled load, long-cable and photovoltaic modules possible. The 8B/10B coding is usually used to data communication. So we choose the 8B/10B Encoder/Decoder in this mini SerDes.

4.2. Architecture of 8B/10B Encoder/Decoder

On one hand, the parallel pipeline architecture is used in this encoder. Figure.9 shows the architecture of this encoder. From the figure, we know that it is a three stage pipeline and divided into mainly three modules: k_enc, 5B/6B enc and 3B/4B enc. These modules are achieved by look up table and a good method is used to minimize the look up table provided in reference [3]. In this figure, the clk as the clock signal is omitted and the k is a control signal to control and instruct that the code is a data symbol or a control signal. In this Encoder, only 12 storage units can be taken to keep control symbol in k_enc . Each storage unit is 11-bit and the MSB stands perfect balance of code.

On the other hand, the decoder is used to convert 10-bit symbols to 8-bit data received from the encoder. As the same as the encoder, the decoder also uses three stage pipeline and it needs 3 cycle latency for the decoded data to transfer to the decoder output ports. The received symbols are decoded based on the running disparity process. Figure.10 shows the structure of 10B/8B decoder. The possible values of the 10 bit symbols received are 1024 kinds of binary, and 464 kinds of them are actual coding and the others are error codes. There are also three modules (3B/4B dec and 5B/6B dec and k_dec).Those decoded data are combined and cached in the second-stage and fed into multiplex or output port in the third-stage.

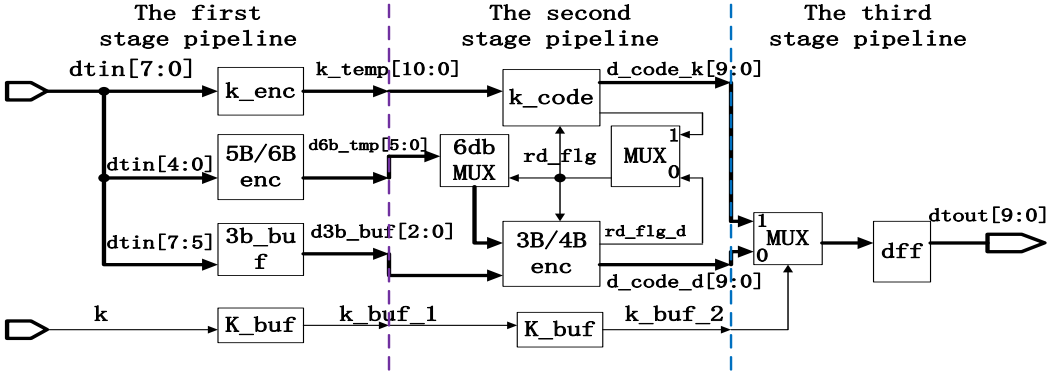

Figure.9 the structure of 8B/10B Encoder

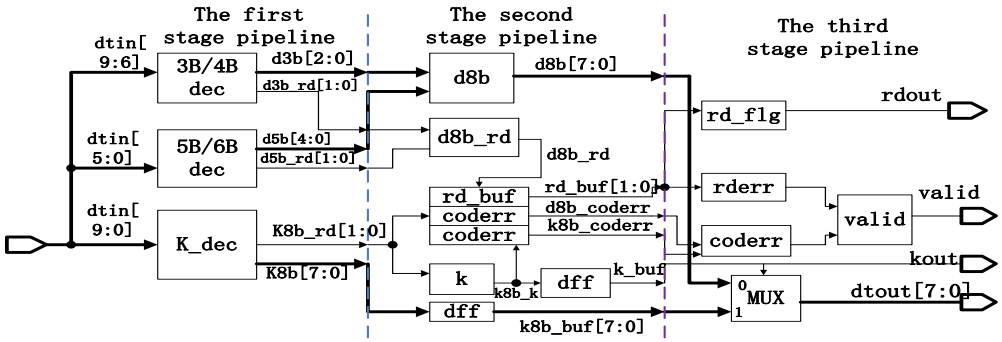

Figure.10 the structure of 10B/8B Decoder

4.3. Timing Simulation

Fig.11 shows the encoder and decoder working well by compared encoder and decoder. And the result is marked in this figure. The figure shows 3 cycle latency both encoder and decoder. And the decoder takes the outputs of the encoder as its inputs. The valid data is decoded correctly and most errors are detected.

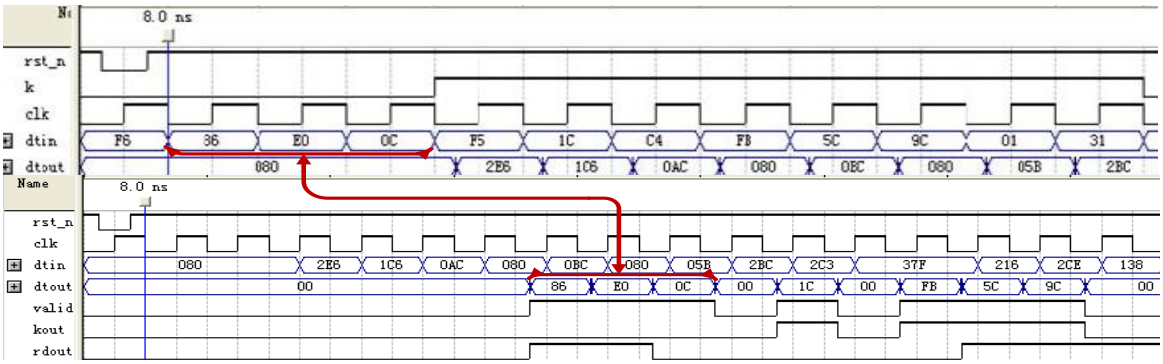

Figure .11 Timing simulation of 8B/10B Encoder/Decoder

5. RELATIVE CONTENTS

5.1. FIFO

FIFO is a first-in-first-out data buffer. In this section, an asynchronous FIFO based on ping-pong operation is proposed. It is often used to asynchronous clock domains to make the data transmission accurately. The Ping-Pong operation is an approach used to the processing of data flow. Its most important feature is that it can transfer data with no pause by switching the two rams by rule. The Ping-pong operation is very suitable for the pipeline operation and can improve the throughout capacity. Figure.12 shows the architecture of FIFO.

As the Ping-Pong operation is used, the problem of switching the two rams must be solved. In this paper, the sel signal is generated by comparing with asynchronous read and write pointer. These pointers are used Gray code to avoid metastable state.

The generating of full/empty signal is a problem. In this FIFO, the method of comparing read pointer and write pointer asynchronously is used effectively in this paper.

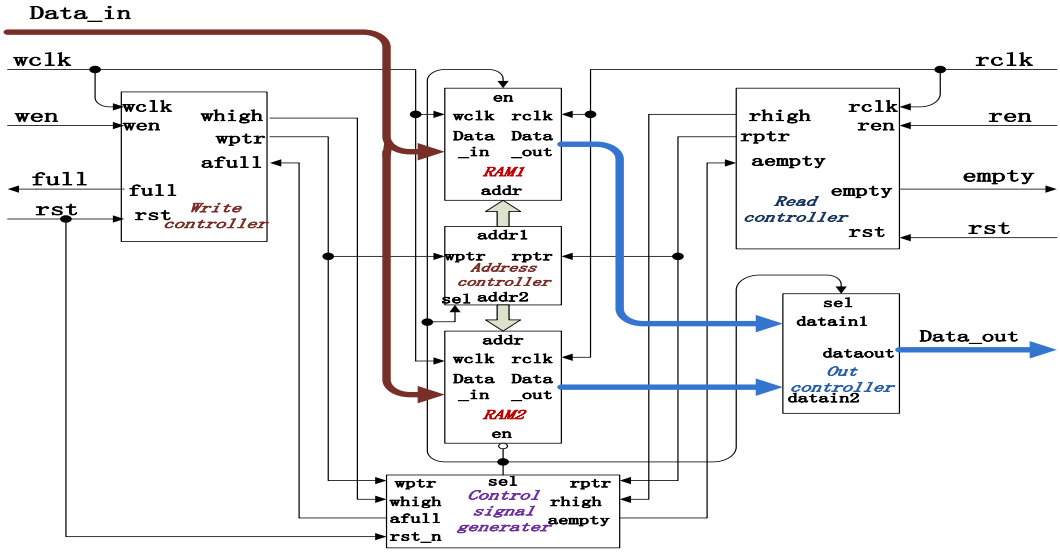

Figure.12 architecture of FIFO

5.2. Serial and Deserial

In this mini SerDes, registers and counters are used to implement the serial operation and deserial operation.

5.3. Clock Management

The clock management is also very important. There are two methods to manage clock. One is using the PLL in FPGA and the other is use special circuit such as divider and multiplier.

In this design, PLL is used to manage the relationship between these modules because of its low clock jitter and skew.

6. CONCLUSION

In this paper, a special method to develop a mini SerDes based on economic FPGA which is synthesized and simulated by Quartus II 8.1. We compared this mini SerDes with the SerDes or transceiver in reference e[5][6][7] and the results are showed in Table 1.

From the table, the speed in our work is lower as a result of the limit frequency of digital PLL in the CDR. There is no high frequency PLL in the economic FPGA, so the PLL has been the speed bottleneck in this design. The output jitter of PLL in CycloneIII is 300ps. We used six clocks so the max frequency is about 555.55MHz (1/(6*300)ps) and the max frequency of output of PLL in CycloneIII is 472.5MHz. The actual max frequency of this work is less than 472.5MHz.

Also it can be seen the obtained method can be widely applied for other projects.

TABLE1 COMPARED RESULTS

	frequency	resources (LES)	Power Consumption (mw)	Technology

This work	400 Mbps	755	144.59	CycloneIII EP3C55F484C8
references[5]	0.25~3.125Gbps	/	100 /channel	Lattice ECP2M (90nm)
references[6]	0.1~3.75Gbps	/	100 /channel	Virtex-5 (65nm)
references[7]	3.125Gbps	/	150 /channel	CycloneIV GX (60nm)

7. Acknowledgements

The authors would like to acknowledge the financial support by Shanghai – AM Fund under Grant No. 09700714000, Grant No.09530708600, and also to acknowledge the financial support by National Science and Technology Ministry under Grant No. 2009GJC00031.

8. References

[1] Razavi B. Challenges in the design of high –speed clock and data recovery circuits[J].IEEE Communications Magazine.2002,(8):94-101

[2] Humberto F Casal, Rafey Mahmud, Trong Nguyen; Mark L Shulman,Nandor G Thoma, “Symmetric clock system for a data processing system including dynamically switchable frequency divider”,USPTO,US5524035

[3] Aref and Ahmed etc,”RTL-Level Modeling of an 8B/10B Encoder-Decoder using SystemC”,Wireless and Optical Communications Networks,2008,WOCN ’ 08.5th IFIP International Conference on ,IEEE Transaction on, pp.1-4,Apr.2008.

[4] Alvin L S Loke, Robert K. Barnes, Tin Tin Wee,etal. A versatile 90-nm CMOS charge-pump PLL for SerDes transmitter clocking[J].IEEE Journal of Solid-state Circuit,2006,41(8):1 894

[5] Lattice semicoductor corporation, “LatticeECP2/M Family Handbook” HB1003 Version 04.8,July 2010

[6] Xilinx, “Virtex-5 Family Overview”,DS100(v5.0),February 6,2009

[7] Altera corporation, “CycloneIV family handbook”.