BER Testing of Communication Interfaces

Yongquan Fan, *Student Member, IEEE*, and Zeljko Zilic, *Senior Member, IEEE*

***Abstract*—This paper presents a versatile bit-error-rate (BER) testing scheme to characterize the quality of communication interfaces. Traditionally, the presilicon BER is evaluated using time-consuming software simulations. The stand-alone BER test products for postsilicon evaluation are expensive and do not include channel emulators, which are essential to testing the BER under the presence of noise. For both the design and evaluation phases, we present a scheme for BER testing in field-programmable gate arrays (FPGAs) that consists of a BER tester (BERT) core and a novel additive white Gaussian noise (AWGN) generator core. The maximum output value of our AWGN generator is 53, whereas that of the existing solutions is less than 7. Therefore, our generator can better emulate the tail of a Gaussian distribution, which is suitable for exploring applications at very low BERs. We also present a pipelined structure that exploits the central limit theorem for speedups of four or more. Combining a BERT and an AWGN in FPGAs is orders of magnitude more efficient in cost, volume, and energy over the existing similar-speed stand-alone solutions and has a huge speed advantage over software simulations. We demonstrate the applications of our solution through two case studies.**

***Index Terms*—Additive white Gaussian noise (AWGN), bit error rate (BER), clock/data recovery (CDR), polar method, signal-to-noise ratio (SNR).**

I. Introduction

FOR A communication system, the *channel* serves as the physical medium used to send a signal from the transmitter to the receiver. One problem associated with the channel is that it corrupts the transmitted signal in a random manner. The additive white Gaussian noise (AWGN) model is predominantly used to analyze this problem. As a measure of how well the overall communication system performs, a bit error rate (BER) is the probability of a bit error at the output of the receiver, whose importance has been widely recognized [1].

Traditionally, software simulations have been used for presilicon BER evaluation, where the real communication system is emulated by its software model. Although software simulations are easy to set up, they are time consuming. A hardware-based solution is commonly 100 000 to 1 million times faster than the best simulation software at the same abstraction level [2]. Although there are some products available for BER testing [3], [4], they do not integrate an AWGN channel emulator; therefore, such testers are difficult to set up for BER testing in the presence of noise.

Theoretically, the tail of an AWGN distribution should extend toward infinity. For an AWGN emulator, the tail is bounded by its maximum output value m, which determines the maximum signal-to-noise ratio (SNR) that the AWGN system can generate. Although there exist stand-alone AWGN generators, their m value is less than 7 [5]–[7]. Their tail distribution accuracy needs to be improved for very low BER applications. In addition, the existing methods of AWGN generation are complicated to implement for high accuracy. Moreover, the cost of the existing stand-alone BER tester (BERT) and AWGN generation solutions is high, ranging from a few thousand dollars to tens of thousands dollars.

This paper proposes a versatile and low-cost scheme for BER testing in field-programmable gate arrays (FPGAs), which is suitable for both presilicon and postsilicon evaluation. The scheme incorporates a BERT and a novel AWGN generator in a single FPGA with a total cost up to a few hundred dollars. The AWGN generator exhibits a better tail distribution, whose m value reaches 53. We also propose a new architecture to implement the central limit theorem (CLT), which can run four times or more over the existing solution. The whole BER testing scheme can be used to test and characterize the performance of a wide range of communication devices, including native clock/data recovery (CDR) interfaces, as well as various user-defined modulation, spread spectrum, and error-correcting codes. Using our solution, we successfully conduct two challenging testing cases: one is testing a high-speed serial interface and the other is testing an AWGN baseband transmission system.

This paper is organized as follows: Section II outlines the background of BER testing. Our scheme for BER testing is presented in Section III. Section IV covers the detailed implementation and performance of our AWGN core and its advantages over the existing solutions. We present the BERT core implementation in Section V. Section VI demonstrates the advantages of our scheme through case studies. Conclusions are presented in Section VII.

II. BER Background

The basic components of a digital communication system include a transmitter, a communication channel, and a receiver. The physical channel may be a pair of wires, an optical fiber, or any other communication medium. The AWGN communication channel model applies to a broad class of physical communication channels. Its mathematical model is shown in Fig. 1 [8].

Manuscript received August 10, 2007; revised November 12, 2007. This work was supported by Altera Corporation.

Y. Fan is with the Integrated Microsystems Laboratory, McGill University, Montreal, QC H3A 2A7, Canada, and also with the Test Engineering of Storage Peripheral Group, LSI Corporation (e-mail: yongquan.fan@mail.mcgill.ca).

Z. Zilic is with the Integrated Microsystems Laboratory, McGill University, Montreal, QC H3A 2A7, Canada (e-mail: zeljko.zilic@mcgill.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2007.913760

0018-9456/$25.00 © 2008 IEEE

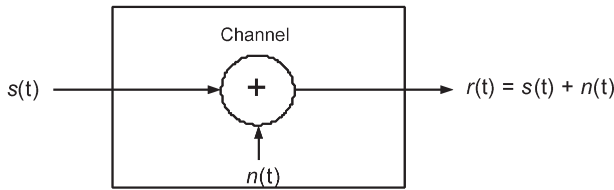

Fig. 1. AWGN channel model.

In the AWGN model, the transmitted signal $s(t)$ is corrupted by noise $n(t)$. The model can be expressed by

$$r(t) = s(t) + n(t).$$

The noise is introduced by the channel, as well as by electronic components, including amplifiers at the receiver. This type of noise is most often characterized as thermal noise or, statistically, as Gaussian noise. Its probability density function (pdf) is expressed by

$$p(x) = \frac{1}{\delta\sqrt{2\pi}} e^{-(x-m_x)^2/2\delta^2}$$

where m_x is the mean, and σ^2 is the variance of the Gaussian random variable.

An important function used to characterize the Gaussian distribution is the Q function, which represents the area under the tail of the Gaussian pdf. $Q(x)$ is used to compute the probability of error in communication systems. Normalized to a zero mean and a unit variance, $Q(x)$ is defined as [8]

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt. \quad (1)$$

A. BER and SNR

A BER is the ratio of the number of incorrect bits and the total number of received bits. In general, the BER is a function of the characteristics of the channel (i.e., amount of noise), the type of waveforms used to transmit information over the channel, the transmitter power, the timing jitter, and the method of demodulation and decoding.

An SNR is the fundamental input quantity that determines the channel capacity C for a given bandwidth B, according to the fundamental Shannon law, i.e.,

$$C = B \log_2(1 + \text{SNR}).$$

In practice, communication system designers balance the bandwidth and SNR to maximize the channel capacity for an acceptable BER performance. The BER is related to the SNR in a way that is impossible to analytically describe in realistic systems, and the key role of the tools, such as ours, is to discover the relation between the two.

There are several types of communication systems in which this balancing act is played in different ways. In baseband transmission, the data and clock are transmitted as digital waveforms. Baseband schemes, such as the commonly used nonreturn-to-zero CDR encoding, combine clock and data signals at the transmitting side and decouple them at the receiver. Careful timing extraction leads to a reduction in the number of transmission errors, which is equivalent to an increase in the system SNR. The theoretical analysis indicates that in a digital baseband, the relationship between the BER and SNR can be expressed by

$$\text{BER} = Q(\sqrt{\text{SNR}}) \quad (2)$$

where $Q()$ is defined in (1).

Another class of communication systems employs *modulation* schemes for communication over a given portion of spectrum. The modulator at the transmitter performs the function of mapping the digital sequence into sinusoidal signal waveforms. The BER performance of the receivers varies widely, depending on the modulation scheme. For example, assuming that a Gray code is used [8], the relationship between the BER and SNR for the quadrature phase-shift keying (QPSK) modulation can be theoretically characterized by

$$\text{BER} \approx Q(\sqrt{2\text{SNR}}) \left[1 - 0.5Q(\sqrt{2\text{SNR}})\right]. \quad (3)$$

In such modulation schemes, as the baseband digital signal is modulated by a complex exponential (sine and cosine waves), two real-valued data streams appear and have to be separately processed. They are referred to as the I (in-phase) channel and the Q (quadrature) channel.

The *spread-spectrum* technique is yet another implementation of the Shannon law by which the transmitted signal bandwidth B is much greater than the information bandwidth C. This excess bandwidth is used as a "coding gain" to protect the signal from the interference caused by multiple users in the same channel, as well as from the intentional jamming.

In all such implementations, the theoretical and practically achieved BER versus SNR curves serve to evaluate the overall capacity and coding gain that is equivalent to the increase in the system SNR. It is desirable to quickly obtain the BER performance of a manufactured device in either such case.

As indicated by (2) and (3), a low BER requires a high SNR. In an AWGN communication system, the SNR is determined by the variance of the AWGN generator. To emulate a low BER system, we usually shrink the distribution of the Gaussian generator to achieve a low variance. However, some outputs of the generator must be large enough to produce bit errors. The lowest BER that an AWGN communication system can achieve is determined by the tail distribution of the Gaussian noise or, more specifically, by the maximum output value m of the noise variable. For a digital system where data "0" and "1" are transmitted, the maximum output of the noise generator is generally required to be bigger than 0.5 to generate bit errors. The existing hardware AWGN generators (e.g., [5]–[7]) have a maximum output value of 7. If we use such generators in baseband transmission, they can only generate a theoretical maximum SNR of 16.9 dB by being scaled by a factor of 14 (see Section VI-B). Although the 16.9-dB SNR can translate into a BER value around 10^{-12} according to (2), such generators are only suitable for exploring the channel behavior at BERs down to the range of 10^{-9} to 10^{-10} [7]. The distribution near its maximum output value of any AWGN generator with bounded

output is no longer Gaussian, as the ideal Gaussian distribution is unbounded (the maximum output value is infinity). Hence, the tail distribution of the existing AWGN generators needs to be improved for very low BER applications, such as 10^{-12}.

B. BER Testing

All BERTs use the same basic principle: known test patterns are sent to a design under test (DUT), and the patterns are compared bit by bit with the output of the DUT after a certain period of time.

1) Software Simulation: In the development of a digital communication system, an initial evaluation of its presilicon performance is usually performed based on a simplified mathematical analysis. Simulation tools like MATLAB and Simulink [9] are mostly used for this purpose. Since a digital communication system suffers from a wide variety of effects that are difficult to accurately analyze, gaining confidence by software simulation is an essential part of the early development stage.

In software simulations, each component of the communication system is built using software models. Each model exhibits the characteristics of its represented component. For instance, in MATLAB and Simulink, the DSP Blockset's Random Source block [10] represents the AWGN noise source. The initial seed and the variance of the noise generated by the block can be specified by users to simulate the AWGN channel in different SNR conditions.

2) Hardware Emulation: Although software simulations are easy to set up for BER evaluation, they are very time consuming. Execution speed depends on the level of abstraction of the simulation models. Due to vast amounts of data and run-time overhead, simulations are generally not suitable for the evaluation of communication systems with a low BER. For example, if we run a simulation of BER $= 10^{-12}$ with ten errors using an average communication model, it would take years on a personal computer equipped with a 1-GHz Pentium-IV processor. Moreover, many design variables, such as sampling frequency, digital format, carrier resolution, rounding, and quantization, have to be optimized while satisfying the best tradeoff between performance and complexity. This would further lengthen the simulation process.

To speed up the BER evaluation process and final parameter optimization, one can perform direct hardware simulation, i.e., *emulation*. As an alternative to simulation, emulation most commonly utilizes programmable logic devices, such as FPGAs, to map all or part of a design. With emulation, performance evaluation can also take place in hardware. To evaluate the BER in hardware, a high-speed channel emulator and a BERT are essential. In [11], a BER testing solution is presented based on Xilinx RocketIO FPGAs, but it does not include a channel emulator. Although a hardware-based emulator combining a BERT and an AWGN can be found in [12], it needs software involvement, and the cost is still high. As a result, there is an urgent need to develop a low-cost hardware-based BER testing scheme that combines an AWGN generator and a BERT.

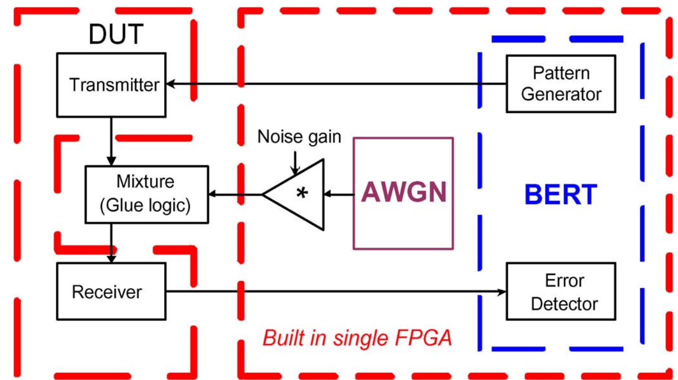

Fig. 2. Block diagram of our BER testing scheme.

III. Overview of Our BER Testing Scheme

To conveniently and cost efficiently test BERs, a new testing scheme is proposed and shown in Fig. 2 [13]. This method can facilitate the BER testing and characterization of various communication interfaces.

The proposed solution combines a BERT and an AWGN generator in a single FPGA device. The AWGN generator is used to emulate an AWGN communication channel by adding noise to transmitted streams. The amplitude of the noise is programmed according to emulated real noise conditions. Hence, we can emulate an AWGN channel in which signals are transmitted with different SNRs. The proposed scheme can easily be set up to test the BER performance of a real DUT in real operations under different SNR conditions.

The DUT can be any communication interface or system that receives bit or word sequences and then restores the sequences after some signal processing or format changes. Some DUT examples include a transceiver (a transmitter and a receiver), the combination of a modulator and a demodulator, and the integration of an encoder and a decoder. Parameterized design enables the tester to interface a DUT either in serial, parallel, or CDR format. The detailed implementation of the AWGN core and the BERT core is discussed in Sections IV and V.

IV. AWGN Core Design

AWGN generation methods utilize a variety of statistical techniques. However, they are almost always based on transformations or operations performed on uniform random variables [14].

A. Existing Methods of AWGN Generation

There are few publications on generating AWGN in digital hardware. The most relevant publications in this area are [5]–[7], which implement AWGN generators in FPGAs. These existing solutions are mainly based on the Box–Muller method.

1) CLT Method: According to the CLT, if X is a random variable of mean m_x and standard deviation (SD) σ^x, the random variable X_N, which is defined as

$$X_N = \frac{1}{\delta_x \sqrt{N}} \sum_{i=0}^{N-1} (x_i - m_x)$$

tends toward the Gaussian distribution of a zero mean and a unit SD when N tends toward infinity, where x_i is N independent instances of X.

The CLT method usually employs an accumulator, which greatly slows down the output rate. In addition, implementing a high-accuracy AWGN generator using this method needs a large number of samples. Hence, it is not suitable for high-speed and high-accuracy applications. The AWGN generator in [15] uses this method. It produces one output every 12 clock cycles by adding 48 random numbers. Its output rate is 1 MHz.

2) Box–Muller Method: The Box–Muller method [17], as shown in Algorithm 1, is the most widely known method for AWGN generation.

Algorithm 1: Box–Muller method

1. Generate two independent random values x_1 and x_2, uniformly distributed over [0, 1].
2. Obtain:
   $f(x_1) = \sqrt{-\ln(x_1)}$
   $g(x_2) = \sqrt{2}\,\cos(2\pi x_2)$.
3. Generate a Gaussian variable
   $n = f(x_1)g(x_2)$.

An FPGA implementation of the Box–Muller method is proposed in [5], where implementing $\ln$ and $\cos$ functions requires careful consideration with regard to the number of recursions and relative position of points, as well as the precision of implementation. The efficient implementation is therefore not straightforward.

Another disadvantage of the Box–Muller method is that it is not suitable for high m value applications. As indicated in Algorithm 1, the maximum output value of n is determined by f, as g is bounded by $[-\sqrt{2}, +\sqrt{2}]$. Since f approaches infinity when values of x_1 are close to zero, the maximum output value of n is determined by the smallest value of x_1. We express x_1 as 2^{-l}, where l is the number of bits used to represent x_1. When $l = 32$, the maximum output value of the generator is around 6.7; while l increases to 64, the maximum value can only increase to 9.4. Obviously, the hardware cost is high, and the output speed is limited if we need to achieve a good tail distribution using the Box–Muller method.

Based on the Box–Muller method presented in [5], a commercial AWGN core has been developed [6]. This core is only capable of a maximum m value of 4.7. More recently, Lee *et al.* have advanced the Box–Muller method [7] that increases the maximum m value to 6.7. This improves the tail distribution of the AWGN generator. However, the price paid for this improvement is the quadrupled hardware resources, whereas the speed is halved. While [7] utilizes sophisticated statistical test tools to evaluate the AWGN generator, in reality, the evaluation tool is determined by applications. In most cases, the Q function is good enough for the performance evaluation as we do in Section IV-D.

3) Mixed Method: As presented in [5], the mixed method combines the CLT method and the Box–Muller method for high accuracy. However, the CLT method slows down the output rate by a factor of N, where N is the number of iterations; therefore, the mixed method decreases the AWGN output rate. For the generator proposed in [5], when $N = 4$, the output rate is only 24.5 MHz, whereas its clock rate reaches 98 MHz.

B. Our Method

Our method combines the polar method, i.e., Algorithm 2, with a version of the CLT method [16] in a way that is suitable for hardware.

Algorithm 2: Polar method

1. **Do**
2. Generate two independent random values U_1 and U_2, uniformly distributed over [0, 1].
3. Set: $V_1 = (2^* U_1) - 1$ and $V_2 = (2^* U_2) - 1$.
4. Set: $S = V_1^2 + V_2^2$.
5. If $S >= 1$, go back to line 2.
6. **Loop** until $S < 1$.
7. Set: $W = \sqrt{-2\,\ln(s)/s}$.
8. Generate two Gaussian variables
   $X_1 = V_1^* W$
   $X_2 = V_2^* W$.

As an improvement to the Box–Muller method, the polar method uses rejection techniques to eliminate the trigonometric calculations that are usually rather slow [14]. It generates two independently distributed Gaussian variables at the same time, which are additionally convenient for applications like QPSK transmission, where two communication channels are needed. The polar algorithm is faster than the Box–Muller algorithm because it uses fewer transcendental functions, although it throws away, on average, 21% of the numbers generated in the Do loop. The proof of validity of the polar method is elaborated upon in [17].

Most importantly, the polar method can easily achieve high maximum output values with little hardware. As indicated in Algorithm 2, the maximum output value is calculated as a logarithm of the minimum value of the square operation of random variables U_1 and U_2. By contrast, the Box–Muller method directly requires calculation of a logarithm of the random variable x_1. In our implementation, we can produce a maximum output value of 53.3 by using only 4 bits (all for the fraction) to represent each uniform random variable (U_1 and U_2 in Algorithm 2). To achieve such a high maximum output value using the Box–Muller method, we need to use more than 1000 bits to represent the uniform random variable x_1 in Algorithm 1, which is almost impossible to implement in hardware.

C. Architecture

1) Polar Method: Based on Algorithm 2, the block diagram of the polar method for two AWGN generators is developed and shown in Fig. 3. It can easily be simplified to a single generator.

We use eight independent linear feedback shift registers (LFSRs) to generate two 4-bit uniformly distributed random variables U_1 and U_2. All bits represent fractions; thus, U_1 and U_2 are uniformly distributed over [0, 0.9375]. The number

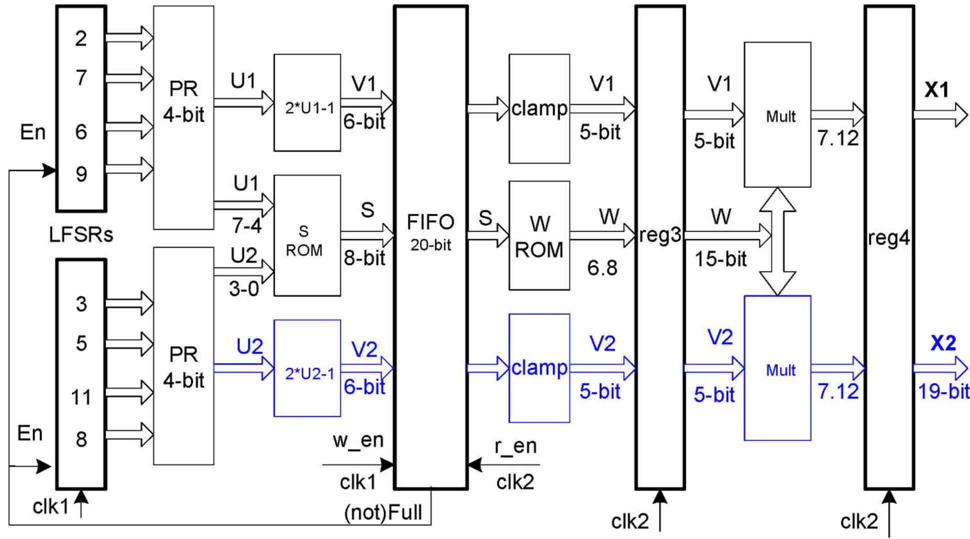

Fig. 3. Block diagram of the polar method.

inside the LFSRs represents the length of each LFSR. V_1 and V_2 are generated using signed adders. As computing S involves lots of additions and multiplications, we use a ROM-based design, where the concatenation of U_1 and U_2 is set to be the address, and S is set to be the data stored in the ROM. If computed $S \geq 1$, zero is stored in the ROM.

As the polar method [17] throws out some data, the Do loop runs faster than lines 7 and 8 in Algorithm 2. We insert a synchronizing first-in first-out (FIFO) to achieve a constant output rate. Two clock signals are used in the FIFO: *clk1* for writing and *clk2* for reading. Writing is enabled when S is not equal to 0, whereas reading is always enabled. The LFSRs are disabled when the FIFO is full. The FIFO is 20 bits in width. By setting the depth of the FIFO to be 16 and *clk2* to be half of *clk1*, we achieve a constant output rate.

We also use a ROM-based design to calculate W, where S denotes the address, and W denotes the data stored in the ROM. As S is between $[0.00390625, 0.99609375]$, W is between $[53.2835, 0.0886]$, which can be denoted using 6 bits for the integer part and 8 bits for the fractional part (6.8).

Finally, two signed multipliers are used to generate two Gaussian variables X_1 and X_2. Each variable is 19 bits in width: 1 bit for sign, 6 bits for integer, and 12 bits for fraction (7.12). The width can be truncated according to applications.

2) Our CLT Method Implementation: Traditionally, the CLT method employs an accumulator, which slows down the output speed by a factor of N, where N is the number of accumulated variables [5]. We propose our CLT method, as shown in Fig. 4, which does not exhibit the speed penalty while improving accuracy. Instead of acuminating data in one stage, the pipelined architecture takes data from previous stages and, hence, can output data every clock cycle.

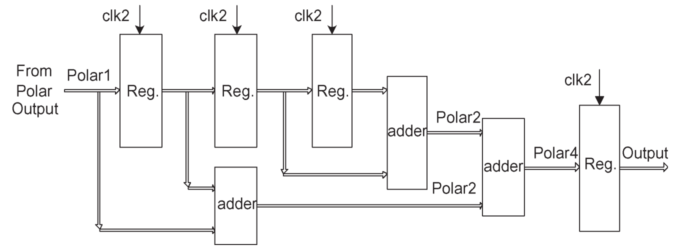

Fig. 4. Our CLT method ($N = 4$).

TABLE I
$Q(x)$ RELATIVE ERROR OF OUR GENERATORS

x	Theory Q(x)	Relative Error of Our Generators		
		Figure 3	Figure 3 + Figure 4	
		10,000	10,000	500,000
0	0.5000	2.76 %	1.02 %	0.24%
0.2	0.4207	-2.50 %	0.50 %	0.42%
0.4	0.3446	1.69 %	0.26 %	0.55%
0.6	0.2743	-0.10 %	1.06 %	0.80%
0.8	0.2119	1.88 %	1.74 %	1.09%
1.0	0.1587	4.70 %	3.21 %	1.20%
1.2	0.1151	-7.17 %	3.99 %	1.49%
1.4	0.0808	-4.29 %	4.95 %	1.85%
1.6	0.0548	-3.06 %	6.57 %	2.37%

D. Experimental Results

Statistical properties of an AWGN generator should be based on evaluating samples from at least one period. The period of our generator may reach around 2^N, where N is the sum of the lengths of all LFSRs; in our case, N equals 51. Statistically evaluating 2^{51} samples requires a lot of hardware resources and time. Our experiments demonstrate that the statistical results of thousands of samples are a good approximation. Table I shows the $Q(x)$ accuracy of our generators with 10 000 and 500 000 samples. The samples are taken from a simulator of our AWGN generator design and then passed to a UNIX workstation to do a statistical analysis. Even for the 500 000 samples, it takes the workstation a whole night to finish the process.

We can see that our method with the parameters shown in Fig. 3 (simplified to a single generator) implements a high-precision AWGN generator, even with a limited number of samples. Our CLT method shown in Fig. 4 can further reduce the variation of the distribution. Moreover, note that the relative

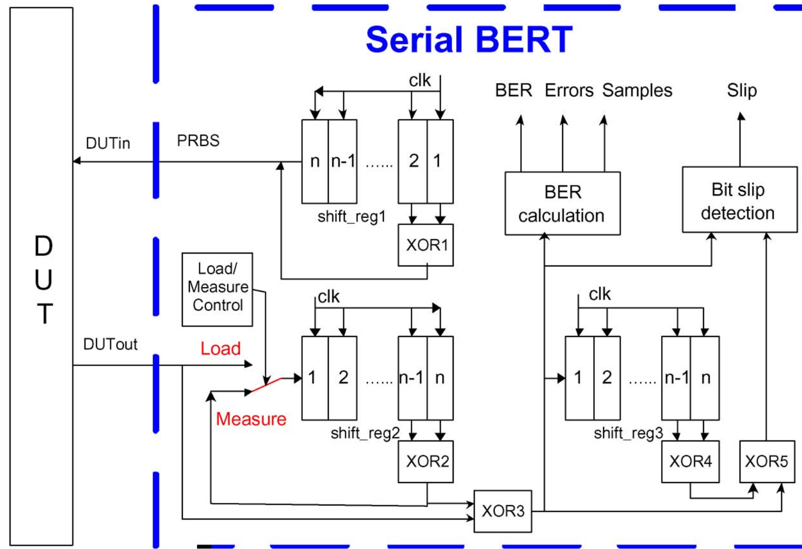

Fig. 5. Block diagram of the serial BERT.

error of $Q(x)$ decreases when the number of samples increases. The limited number of samples is the main reason for the error.

We also evaluated the accuracy of our AWGN generator by calculating the *kurtosis* values [18]. Kurtosis characterizes the relative peakedness or flatness of a distribution compared to the Gaussian distribution, which has a theoretical value of 3. Based on the 500 000 samples, we measured a kurtosis of 2.95, which is 2% less than the required value.

V. BERT Core Design

As discussed in Section II, the basic concept of a BER measurement is as follows: The pattern generator sends a data stream to a DUT, and the error detector conducts a bit-by-bit comparison of the received signal from the DUT.

A. Serial BERT Design

A serial BERT sends serial bit sequence patterns to a DUT and evaluates the output from the DUT. The DUT can be any serial digital communication link. The structure of a serial BERT is proposed and shown in Fig. 5.

In this scheme, the shift register *shift_reg1* and the gate XOR1 form an LFSR. As the pattern generator of the serial BERT, the LFSR generates pseudorandom bit sequences (PRBSs). These sequences are then sent to the DUT.

Before a measurement begins, the *load/measure* switch is set to be in the *load* state until the *shift_reg2* is fully loaded with the contents of the *shif_reg1*. The switch is changed to the *measure* state to start the BER measurement. The shift register *shift_reg2*, the switch, and the gate XOR2 are used for synchronization. They generate a reference pattern by replicating the PRBS from the *shift_reg1* but delaying the phase. During the synchronization process, it is assumed that all the bits are correctly transmitted.

The gate XOR3 serves as a comparator, comparing the pattern from the DUT with the reference pattern. If the test pattern is correctly transmitted by the DUT, then the two inputs of XOR3 should be of the same value in each clock cycle. In a real BER measurement, the output of XOR3 is monitored every clock cycle: if a “1” is detected, a transmission error is counted; otherwise, the transmission is error free.

In a real communication system, the transmission errors are in forms of single-bit errors, error bursts, or bit slips. Bit slips result from a bit loss or a bit repeat. If a bit slip happens, only the repeated or lost bits should be counted as errors. We employ a mechanism to distinguish between error bursts and bit slips and to eliminate false long-term errors.

In Fig. 5, the shift register *shift_reg3* and the gates XOR4 and XOR5 perform bit slip detection. The solution is based on the fact that the addition or superimposition of two PRBSs shifted in phase relative to each other produces another PRBS [19]. By monitoring the output of XOR3 and XOR5, it can be determined whether a bit slip happens.

B. Parallel BERT Design

A parallel BERT is used to test communication interfaces that transmit parallel data. The design of the parallel BERT is based on the serial BERT presented in Section V-A. Basically, a k-bit parallel BERT, where k is the width of the parallel data (bit0 $\sim$ bit$(k-1)$) can be built using k independent serial BERTs that have the same load time. The parallel BERT sends pseudorandom word sequences (PRWSs) to the DUT. To qualify randomness of the generated sequences, the independence of each of the serial BERTs is very important, which means that the length of the shift registers in each of the serial BERTs should be different.

When k independent serial BERTs are directly put together to build a parallel BERT, each of the serial BERTs has circuits for the *load/measure* switch control and bit slip detection. However, the circuits for the *load/measure* switch control of each bit of the parallel data should change the *load/measure* state at the same time. Also, the parallel BERT should be capable of distinguishing between error bursts and word slips instead of bit slips in a serial BERT. Therefore, only one of

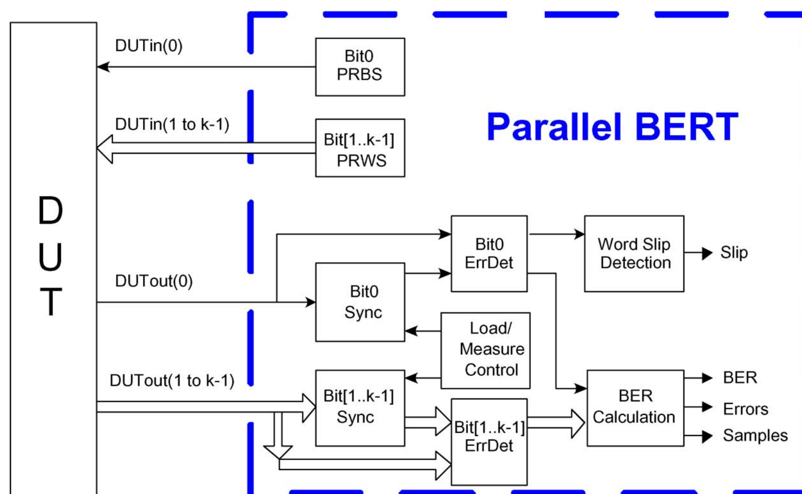

Fig. 6. Block diagram of the parallel BERT.

TABLE II
SYNTHESIS RESULTS OF THE AWGN AND BERT

Function Block	Logic Elements	ESB Bits	fmax
AWGN	336/4800 (7%)	5856/49152 (11%)	73.48 MHz
BERT	384/4800 (8%)	0/49152 (0)	160.3 MHz
BERT + CDR	837/4800 (17%)	320/49152 (<1%)	160.3 MHz

the k such control circuits is needed for the switch control and word slip detection. Fig. 6 shows the structure of the proposed parallel BERT. In the design, the serial BERT control circuitry for bit0 is used for the *load/measure* switch control and the word slip detection.

A parallel BERT interfaces a DUT with parallel data, which requires lots of connection wires and stringent timing specifications. The connection interface can be greatly simplified by inserting CDR interfaces between the parallel BERT and the DUT. More details about the CDR circuitry are discussed in Section VI-A.

C. Synthesis Results

The BERT and the previously discussed AWGN designs are built in very high speed integrated circuit hardware description language (VHDL) and can target almost any FPGA devices. The synthesis has been done using Quartus II tools by Altera [22]. Table II shows the synthesis results of the AWGN design and the parallel BERT designs based on the Altera Mercury FPGA EP1M120F484C7 device. The embedded system block bits are used to implement ROM-based functions, such as S and W in Fig. 3. Other logic functions are implemented in the logic elements (LEs). Each LE contains a four-input lookup table, which can quickly implement any function with four variables. The LE also contains a programmable register and a carry chain [20].

As shown in Table II, the AWGN and the BERT only occupy a small part the FPGA device (less than 20%). There are enough resources in the FPGA to implement other application-specified functions in a real BER testing system, such as data storage, protocol implementations, special test controls, and user logic circuits.

VI. CASE STUDIES

This section presents the applications of the proposed BERT and AWGN cores for testing of high-speed serial interfaces and baseband-encoded channels, where they fare favorably with the existing methods.

A. High-Speed Serial Interface Testing

We test gigabit serial interfaces of the Altera Mercury FPGA devices by building our core into the devices. The Mercury gigabit transceiver is implemented using the high-speed differential interface (HSDI) to transmit and receive high-speed serial data streams (up to 1.25 Gb/s). Fig. 7 shows the block diagram of one of the eight HSDI transceiver channels (channel 4) of an EP1M120F484C7 device [20].

As shown in Fig. 7, the HSDI transmitter includes a synchronizer and a serializer, whereas the receiver includes a clock recovery unit, a deserializer, and a synchronizer. The HSDI phase-locked loop circuitry is dedicated to providing clocks for the transceiver. As the clock is encoded into the data signal in the transmitter and recovered at the receiver side, the whole transceiver is often referred to as the CDR interface. Based on its structure, the setup to test the transceiver is developed and shown in Fig. 8.

In the testing setup, the HSDI transceiver is built by instantiating the Altera block [21]. The data width of the BERT is 8 bits. The glue logic is developed to interface the BERT and the transceiver. The *Error/Slip Injection* block inserts errors or word slips for the purpose of the demonstration of the BERT functionalities. The *8B10B encoder* encodes the 8-bit sequences to 10-bit sequences to ensure enough bit transitions in the serial link for date recovery [23]. A FIFO is used to ensure that there are always data ready for transmission after a test begins.

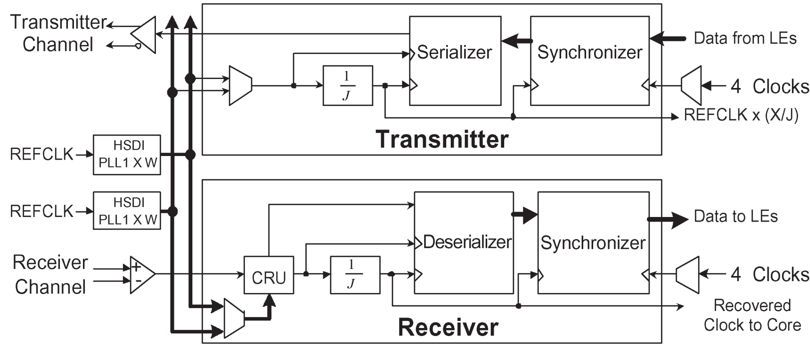

Fig. 7. HSDI transceiver block diagram.

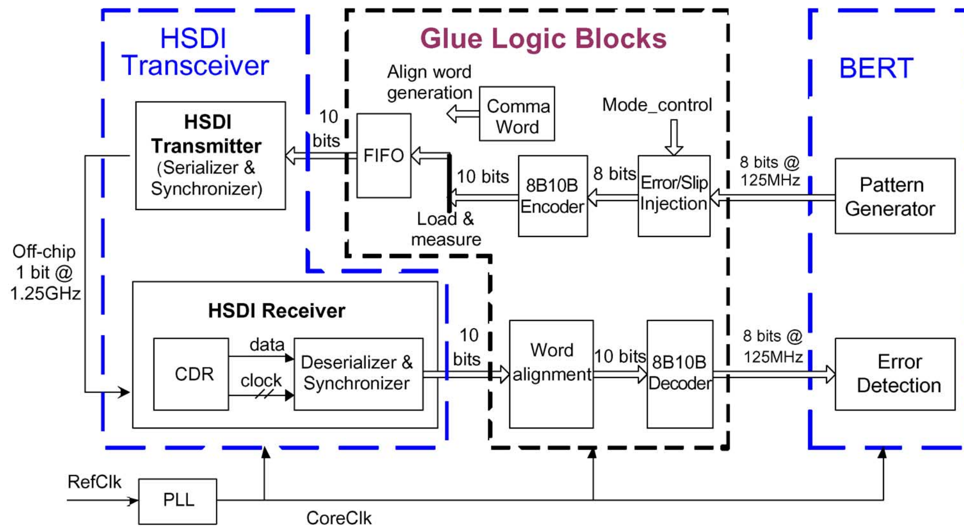

Fig. 8. Testing setup for an HSDI transceiver.

Comma words are inserted at the start of the testing for word alignment. The *8B10B Decoder* recovers the 8-bit PRWSs sent by the BERT.

The testing setup is implemented in VHDL, targeting the EP1M120F484C7 device using Quartus II software. The synthesized results are downloaded onto an Altera Mercury HSDI CDR Demo board. The outputs of the transmitter are connected to the inputs of the receiver by two subminiature version A cables.

We obtained a zero BER both from simulations and from running real tests in the board when no error or bit slip was injected. The zero BER experiment results demonstrate the functional correctness of the HSDI transceiver and the feasibility of the testing setup.

B. Baseband Transmission Testing

Based on the AWGN model discussed in Section II-A and our BER testing scheme from Section III, we develop a testing setup for a digital baseband communication (Fig. 9).

In this system, one is used to transmit data "1," and zero is used to transmit data "0." Assuming that data 1s and 0s have equal occurring probabilities, the average energy of transmitted signals is

$$E = (E_0 + E_1)/2 = 0.5.$$

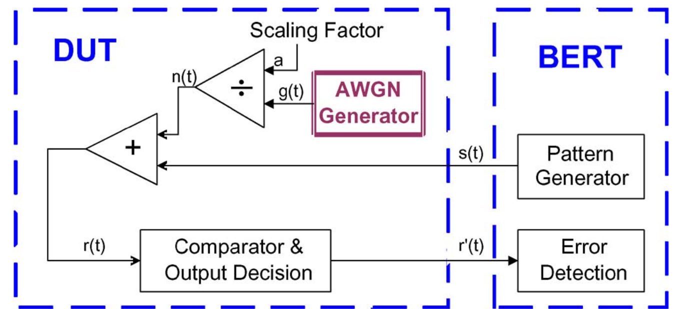

Fig. 9. BER testing setup for a digital baseband.

In the testing setup, the channel is emulated by scaling the AWGN generator with a zero mean and a unit variance by a factor of a. In this case, the energy of the noise is

$$N_o = 2/a^2.$$

As a result, we have

$$\text{SNR} = E/N_0 = a^2/4.$$

As shown in Fig. 9, the transmitter consists of the pattern generator, and the receiver consists of the comparator and the output decision block. If the noise-corrupted signal $r(t)$ is bigger than 0.5 (threshold voltage), $r'(t)$ is set to 1; otherwise,

TABLE III
BER MEASUREMENTS FOR A DIGITAL BASEBAND

a	SNR (dB)	Total bits	Our BER	Agilent BER	Error (%)
2	0.6	2024	1.62e-1	1.65e-1	1.85
3	4.1	12448	6.58e-2	6.61e-2	0.45
4	6.6	12448	2.32e-2	2.40e-2	3.33
5	8.5	20000	5.65e-3	5.32e-3	4.32
6	10.1	1000000	1.20e-3	1.31e-3	8.40
7	11.5	1000000	1.76e-4	1.83e-4	3.83
8	12.6	10000000	1.62e-5	1.78e-5	8.99

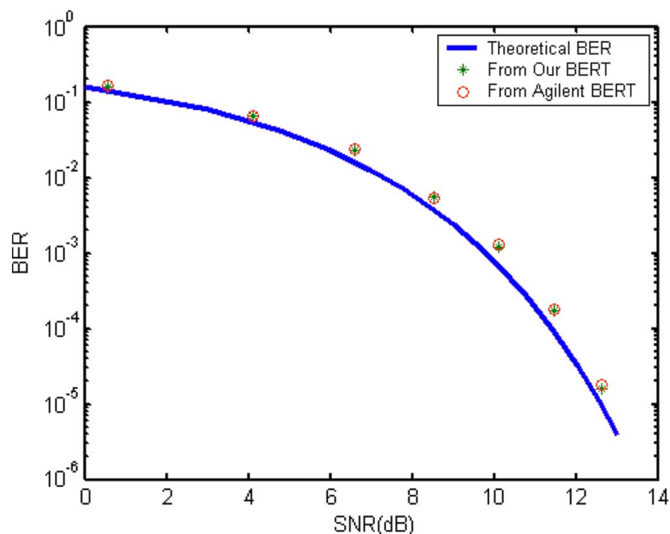

Fig. 10. Measured BER versus theoretical BER.

$r'(t)$ is 0. Table III lists the test results. The measurements were taken while running the testing setup in an Altera Mercury FPGA board with a clock of 50 MHz. The measurement is first done using our BERT and then an Agilent BERT.

Fig. 10 shows the plot of the theoretical BER, the measured BERs using our BERT, and an Agilent 81200 BERT as a function of the input SNR.

As indicated in Fig. 10, the measured BER using our BERT perfectly coincides with that from the expensive Agilent BERT and is close to the theoretical BER. The plot demonstrates the validity of our testing scheme.

In the testing setup, the pattern is generated using an LFSR, which produces more 1s than 0s, as all zeros is not a valid pattern, whereas all ones is. Therefore, the actual signal energy is bigger than the theoretical value of 0.5; for example, if the length of the LFSR is 3, the signal energy it produces is 4/7. We take this factor into consideration when calculating the SNRs in Table III.

In the aforementioned testing, it takes less than 1 s to generate the point at 1.62e-5 BER, whereas software simulations take hours. Furthermore, regardless of the generation scheme, going down to low error rates requires many samples just to exhibit errors—for BER $= 10^{-12}$ at a 1-GHz data rate, it takes 3 h (assuming running 10^{13} bits to guarantee a 10^{-12} BER level). In production, the normal practice to qualify the BER performance at such low levels is through extrapolation [24]. However, if the direct BER measurements at 10^{-12} or lower are needed, our AWGN can accomplish it, as its maximum output value reaches 53.

Although the aforementioned experiment is based on testing a digital baseband system, the proposed BER testing scheme applies to any AWGN digital transmission system. Furthermore, the AWGN module can be modified to emulate more complex channels, such as Rayleigh channels.

VII. CONCLUSION

In this paper, we present a versatile and low-cost BER testing scheme. The scheme combines a BERT and an AWGN generator in a single FPGA device, which is suitable for the testing and characterization of a wide range of signal communication interfaces. The novel structure of our AWGN generator enables us to generate a much bigger maximum output value than the existing solutions (53 versus seven), which is essential for evaluating the performance of AWGN communication systems at low BER applications. In addition, our CLT implementation can speed up the traditional accumulator approach by four times or more, which is useful to boost the accuracy of any AWGN generator. Most importantly, the combination of an AWGN and a BERT in our solution is only a small fraction of the cost, volume, and energy requirements of a standard-alone BERT (e.g., [3]) and a standard-alone AWGN generator. Furthermore, the FPGA-based solution makes it easy to interface with DUTs.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their guidance.

REFERENCES

[1] E. A. Newcombe and S. Pasupathy, “Error rate monitoring for digital communications,” *Proc. IEEE*, vol. 70, no. 8, pp. 805–825, Aug. 1982.
[2] M. Courtoy, “Rapid system prototyping for real-time design validation,” in *Proc. 9th Int. Workshop Rapid Syst. Prototyping*, 1998, pp. 108–112.
[3] Agilent Technologies, *Agilent 81200 Data Generator/Analyzer Data Sheet*, 2002.
[4] *48 Gb/s BER Test System Datasheet*, Anritsu Corp., Kanagawa, Japan, 2002.
[5] A. Gazel, E. Boutillon, J. L. Danger, G. Gulak, and H. Lamaari, “Design and performance analysis of a high speed AWGN communication channel emulator,” in *Proc. IEEE PACRIM Conf.*, Aug. 2001, pp. 374–377.
[6] Xilinx Inc., *Additive White Gaussian Noise (AWGN) Core v1.0*, Oct. 2002. Production Spec.
[7] D. Lee, W. Luk, J. Villasenor, and P. Y. K. Cheung, “A Gaussian noise generator for hardware-based simulations,” *IEEE Trans. Comput.*, vol. 53, no. 12, pp. 1523–1534, Dec. 2004.
[8] J. G. Proakis, *Digital Communications*. New York: McGraw-Hill, 2001.
[9] The MathWorks, Inc. [Online]. Available: http://www.mathworks.com
[10] *Using the Communication Blockset*, MathWorks, Inc., Natick, MA, Jul. 2002.
[11] *SAPP661: RocketIO Transceiver Bit Error Rate Tester, V2.0*, Xilinx, Inc., San Jose, CA, Jun. 2003.
[12] Nallatech Ltd., *Complete Hardware-Based Solution for Bit Error Rate Testing*, Sep. 2002.
[13] Y. Fan, Z. Zeljko, and M. Chiang, “A versatile high speed bit error rate testing scheme,” in *Proc. Int. Symp. Quality Electron. Des.*, 2003, pp. 395–400.
[14] M. F. Schollmeyer and W. H. Tranter, “Noise generators for the simulation of digital communication systems,” in *Proc. 24th Annu. Simul. Symp.*, Apr. 1–5, 1991, pp. 264–275.

[15] P. Atiniramit, "Design and implementation of an FPGA-based adaptive filter single-use receiver," M.S. thesis, Virginia Polytechnic Inst., Blacksburg, VA, 1999.

[16] Y. Fan and Z. Zilic, "A novel scheme of implementing high speed AWGN communication channel emulators in FPGAs," in *Proc. Int. Symp. Circuits Syst.*, May 23–26, 2004, vol. 2, pp. II-877–II-880.

[17] D. E. Knuth, *The Art of Computer Programming*. Reading, MA: Addison-Wesley, 1998.

[18] D. Ruppert, "What is kurtosis? An influence function approach," *Amer. Stat.*, vol. 41, no. 1, pp. 1–5, Feb. 1987.

[19] R. Kiefer, *Test Solutions for Digital Networks*. Heidelberg, Germany: Huthig GmbH, 1998.

[20] Altera Corp., *Mercury Programmable Logic Device Family Data Sheet*, Jan. 2003.

[21] Altera Corp., *Mercury Gigabit Transceiver MegaCore Function User Guide*, Feb. 2002.

[22] *Introduction to Quartus II Manual*, Altera Corp., San Jose, CA, Jul. 2003.

[23] Altera Corp., *8B10B Encoder/Decoder MegaCore Function User Guide*, Dec. 2002.

[24] Y. Fan, Y. Cai, L. Fang, A. Verma, B. Burcanowski, Z. Zilic, and S. Kumar, "An accelerated jitter tolerance test technique on ATE for 1.5 GB/s and 3 GB/s serial-ATA," in *Proc. IEEE ITC*, 2006, pp. 1–10.

Yongquan Fan (S'07) received the B.S. degree in electrical engineering from Beijing University of Aeronautics and Astronautics, Beijing, China, in 1991 and the M.S. degree in electrical engineering from McGill University, Montreal, QC, Canada, in 2003. He is currently working toward the Ph.D. degree at McGill University.

He is also a Staff Engineer with the Test Engineering of Storage Peripheral Group, LSI Corporation, where he is developing test solutions for high-speed mixed signal products. His research concentrates on accelerating BER and jitter testing.

Zeljko Zilic (M'97–SM'07) received the Ph.D. degree from the University of Toronto, Toronto, ON, Canada, in 1997.

From 1997 to 1998, he was a Member of Technical Staff with the FPGA Division, Microelectronics Group, Lucent Technologies. He is currently an Associate Professor with McGill University, Montreal, QC, Canada. He is also on the Editorial Board of the *Journal of Multiple-Valued Logic and Soft Computing* and the *International Journal of Software and Information Technologies*. He has published more than 100 research papers and coauthored the book *Verification by Error Modeling* (Kluwer, 2003). He is the holder of four patents in the area of clock and power management. His current research interests include various aspects of system design, test, and verification.

Prof. Zilic has served as a member of Technical Program Committees for the ACM International Symposium on FPGAs, the IEEE International Test Conference, the Midwest Circuits and Systems Symposium, and the Electronic Circuits and Systems Conference. He has been granted the Chercheur Strategique Research Chair from the Province of Quebec. He was a recipient of the Myril B. Reed Best Paper Award from the IEEE International Midwest Symposium on Circuits and Systems in 2001, a Best Paper Award from the Design and Verification Conference in 2005, and several honorary mention awards. For his undergraduate teaching, the National Council of Deans of Engineering and Applied Science and Sandford Fleming Foundation awarded him with the Wighton Fellowship in 2006. As a Vice President of Research and Development, he helped start up Monroi, Inc., which is a company that was given an Entrepreneur of the Year Award by the St. Laurent Chamber of Commerce in 2004.