Journal of Engineering and Applied Sciences 13 (5): 1124-1129, 2018
ISSN: 1816-949X
© Medwell Journals, 2018

Proposal of Parallelization Structure for PingPong 256

[1]Ki Hwan Kim, [2]Tae Yong Kim, [2]Sang Gon Lee, [2]Won Tae Jang and [2]Hoon Jae Lee
[1]Department of Ubiquitous IT, DongseoUniversity School, 47011 Busan, Republic of Korea
[2]Division of Information and Communication Engineering, Dongseo University,
47011 Busan, Republic of Korea

Abstract: Cryptography is used in various fields such as network communication, disk storage, user and equipment authentication. However, most cryptographic devices use block ciphers or public key ciphers and there are few instances of using stream ciphers. This can be seen as a question of the reliability and security of stream ciphers. In this study, we improve the speed of PingPong stream cipher and we have designed a hardware structure for single and multi-channel PingPong 256 where we implemented in the simulation: first, the original one path was extended to four to increase performance. Second, it changes the variable clock and feedback functions to assure the randomness based on the same initial value.

Key words: Stream cipher, LFSR, VHDL, parallelization, PingPong 256, randomness

INTRODUCTION

In modern society, information has a monetary value which changes according to time, environment and people. The source of the information can be a group like a company or a country or it can be an individual. Most people encrypt private information such as identity information, so as not to be exposed to others.

Encryption schemes are classified into symmetric key ciphers or asymmetric key ciphers. Block ciphers and stream ciphers belong to symmetric key ciphers (Menezes *et al*., 1996; Wenbo, 2003). Since, the stream cipher use binary or word stream from a random generator as a key stream to encipher the plain text, the strength of the cipher is determined by the randomness and period of random number generator. A Linear Feedback Shift Register (LFSR) is a well-known tool of random number generator and the maximum period of a LFSR depends on the taps of a primitive polynomial. When we classify the stream cipher according to a combination of LFSR, it can be classified into a clock-controlled type, a nonlinear combination function and a nonlinear filter function (Gollmann and Chambers, 1989; Key, 1976). The security strength of a clock-controlled LFSR is increased by the clock adjustment of the LFSR. The nonlinear combining function has a memory-type or non-memory-type combining function that combines several linear LFSRs to increase the nonlinearity of the output sequence. The nonlinear filtered function has a nonlinear function that generates the output by nonlinearly combining each memory state of the linear LFSRs.

MATERIALS AND METHODS

PingPong 256 stream cipher

Base of software structure: PingPong 256 is the key sequence generator and as shown in Fig. 1, the output bits from the LFSR are input to the combined function (f_z), carry function (f_c) and memory function (f_d), respectively to produce the following memory states and key sequence bit.

The LFSR is supplied with an irregular clock and the number of irregular clocks supplied to one LFSR is obtained from the clock adjustment function (f_a, f_b) generated in the remaining LFSR. The LFSR selects a

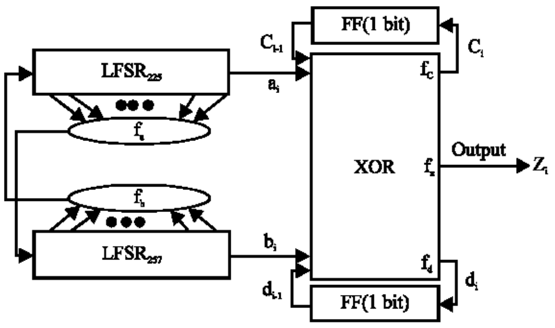

Fig. 1: PingPong 256 full structure

Correspondig Author: Hoon Jae Lee, Department of Information and Communication Engineering, Dongseo University School, 47011 Busan, Republic of Korea

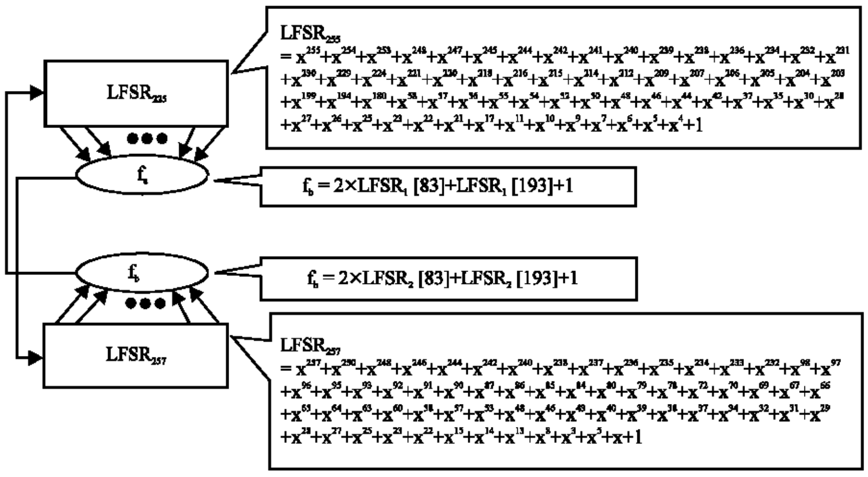
LFSR255
= x^255+x^254+x^253+x^248+x^247+x^245+x^244+x^242+x^241+x^240+x^239+x^238+x^236+x^234+x^232+x^231+x^230+x^229+x^224+x^221+x^220+x^218+x^216+x^215+x^214+x^212+x^209+x^207+x^206+x^205+x^204+x^203+x^199+x^194+x^180+x^58+x^57+x^56+x^55+x^54+x^53+x^50+x^48+x^46+x^44+x^42+x^37+x^35+x^30+x^28+x^27+x^26+x^25+x^23+x^22+x^21+x^17+x^11+x^10+x^9+x^7+x^6+x^5+x^4+1
LFSR255
fa
fb = 2×LFSR1 [83]+LFSR1 [193]+1
fb
fb = 2×LFSR2 [83]+LFSR2 [193]+1
LFSR257
LFSR257
= x^257+x^250+x^248+x^246+x^244+x^242+x^240+x^238+x^237+x^236+x^235+x^234+x^233+x^232+x^98+x^97+x^96+x^95+x^93+x^92+x^91+x^90+x^87+x^86+x^85+x^84+x^80+x^79+x^78+x^72+x^70+x^69+x^67+x^66+x^65+x^64+x^63+x^60+x^58+x^57+x^53+x^48+x^46+x^43+x^40+x^39+x^38+x^37+x^34+x^32+x^31+x^29+x^28+x^27+x^25+x^23+x^22+x^15+x^14+x^13+x^8+x^3+x^5+x+1

Fig. 2: Polynomial of PingPong 256 function

primitive polynomial as shown in Fig. 2 and does not allow all bits to be initialized to the 0 state. The clock control functions f_a and f_b are obtained by the current state of the two LFSRs and the LFSR is randomly clocked and generates carry, memory and key sequence output.

The functions f_c, f_d and f_z are defined according to the point 'i' of the result of the LFSR determined by the clock adjusting as follows:

$$z_i = f_z(a_i, b_i, c_{i-1}, d_{i-1}) = a_i \oplus b_i \oplus c_{i-1} \oplus d_{i-1} \quad (1)$$

$$c_i = f_z(a_i, b_i, c_{i-1}) = a_i \oplus b_i \oplus c_{i-1} \oplus d_{i-1} \quad (2)$$

$$d_i = f_z(a_i, b_i, d_{i-1}) = b_i \oplus (a_i \oplus b_{i-1}) \oplus d_{i-1} \quad (3)$$

PingPong 256 is a key sequence generator that receives 255 bits and 257 bits of LFSR (LFSR255, LFSR257) as an Initial Value (IV) and operates the LFSR irregularly by using mutual clock functions f_a and f_b. Also, c_i and d_i are computed using the exclusive or of the bits input through each LFSR and finally 1 bit is output using c_{i-1} and d_{i-1} which are initial bits of 0 bits.

PingPong 256 affects the length of the tabs in the LFSR and the raw polynomial of the short and long tabs in the LFSR is guaranteed to be the maximum period of time but the primitive polynomial of the long tab is relatively guaranteed. Thus, the shorter the length of a tab, the smaller the value changes for each register, so that, the risk of changing the clock control function has linearity exists, thus, the length of the raw tab, 16 or more is recommended for PingPong 256.

PingPong 256 creates a period such as Eq. 4 which prevents reuse of the same key sequence when encrypting long messages. It also has linear complexity such as Eq. 5 to withstand attacks using Berlekamp-Massey. Finally, a good statistical feature is that the frequency of the sequence of keys "0" or "1" should be approximately the same to withstand an attack (Golomb *et al.*, 1982). We have calculated the linear complexity and period of PingPong 256 as follows:

$$LC \geq 2^{4.6} \times 2^{\left\lceil \frac{512-11}{2} \right\rceil} = 2^{4.6} \times 2^{0.5} \times 2^{250} \approx 2^{256} \quad (4)$$

$$P \geq 2^{4.6} \times 2^{\left\lceil \frac{512-11}{2} \right\rceil} = 2^{4.6} \times 2^{0.5} \times 2^{250} \approx 2^{256} \quad (5)$$

Test of safety verification: Randomness tests based on stochastic modelling is very important to reserve random (i.e., pattern less) for the distribution of a set of data. In the experiment, the original polynomial of each LFSR in PingPong 256 uses a formula with 32 elements and a total of 2457600 bits are output and verified. The experiment was performed 5 times including frequency test, serial test, general serial test, poker test, autocorrelation test and it was repeated 5 times using different initial values. The experimental results are shown in Table 1. To prove that the proposed algorithm is safe, we set the effective range to within 5% using the standard distribution chart (Rukhin *et al.*, 2010).

The experimental results are shown in Table 1. The frequency test compares the total number of '0' or '1' to see if the two types show a ratio close to 50%.We confirmed the number of '0' or '1' bits in the experiment and confirmed that it was smaller than the criterion value

Table 1: Results of randomness test for PingPong-256

Test items	Criterion	Case 1	Case 2	Case 3	Case 4	Case 5
Frequency test						
Total bits	-	2457600	2457600	2457600	2457600	2457600
No. of 0's	-	1228356	1228977	1227880	1227573	1228084
No. of 's	-	1229244	1228623	1229720	1230027	1229516
Chi 2-1	3.841	0.321	0.051	1.378	2.45	0.834
Serial test						
0-->0	-	613887	614045	613592	612942	614413
0-->1	-	614469	614931	614288	614631	613670
1-->0	-	614468	614931	614287	614630	613670
1-->1	-	614775	613692	615432	615396	615846
Chi 2-2	5.991	0.352	1.888	1.46	2.797	4.304
Generalized serial test						
3-serial	9.488	0.653	4.285	2.143	4.877	6.134
4-serial	15.507	3.804	8.145	3.088	9.187	9.546
5-serial	26.296	15.596	11.509	21.069	14.963	13.799
Poker test						
Block word length 3	14.06	6.351	7.196	3.919	3.506	2.744
Block word length 4	24.996	8.206	6.648	10.6	13.751	13.828
Block word length 5	44.654	32.259	50.179	28.109	10.218	28.772
Auto-correlation test						
Value	Max ≤0.05	0.861091	0.175447	0.774747	0.555394	0.062589

3.841 in all experiments. A series test is based on the assumption that the number of consecutive 2 bits '00', '01', '10', '11' is the same. Experimental results show lower values than the criterion value 5.991 showing that the state change of consecutive bits is constant. The generalized serial test is a very useful test for the randomness or pseudo-randomness of sequences, especially, binary sequences. The poker test determines whether the sequences of length d each appear approximately the same number of times in S as would be expected for a random sequence. All of the experimental results were below the criterion value. The autocorrelation test is a test of the relationship between two bits of constant interval difference. All of the experimental results were below the criterion value.

RESULTS AND DISCUSSION

Design of parallelization channel of PingPong 256

Single channel structure: In PingPong 256, we use two LFSRs with one clock. However, the variable clock architecture will limit the performance of the operation on a single channel. If the LFSR is operated as a single circuit, each round it must be operated up to 4 times. As in Table 2, we have selected 14 clocks based on Fig. 1 where we have executed 5 rounds for each clock. In order to understand each round we have used 'R' and 'NR' for classification. However, 'R' appear if the LFSR is running otherwise 'NR' appear if the LFSR is not running. f_a and f_b are the results of the variable clock function where f_a is the operation running time of the LFSR 255 and f_b is the operation running time of the LFSR257. For instance, the LFSR255 operates and stop 2 times when we select the clock from 0 to 3where the LFSR257 operates 4 times with no stop. In order to improve the complexity of the research, we have chosen the circuit of four cases and selected one circuit for each round. As a result, the repeatability and complexity of the signal results is increased and the performance is improved by 4 times compared to the existing single structure.

Due to the variable clock structure, four state equations are selectively used in order to improve the delayed computation speed. The problem is shown in Table 2 through the structure modification. To solve this problem we have designed anew hardware structure for Single channel PingPong 256 structure which is showing in Fig. 3. Figure 4 shows the result of gate level simulation.

Parallelization channel structure: We observed that the single-channel PingPong 256 can be used as a generator with random properties. However, PingPong 256 which has a single channel structure can operate only 1 bit per clock, so there is a drawback that the speed is low. We have devised a parallel architecture of the PingPong 256 to connect separate channels to solve the problem of output limitation according to a single channel. First, we learned that the variable clock structure and channel and memory capabilities of the existing PingPong 256 architecture can affect output and help generate random numbers. That is connecting a separate channel does not mean that interworking is broken.

Table 2: Example of how to operate a single PingPong 256

Clock	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Round	0	0	0	0	1	1	2	2	2	3	3	3	4	5	5
$LFSR_{255}$	R	R	NR	NR	R	NR	R	R	R	R	R	R	R	R	R
f_a	4	3	2	1	2	1	1	0	0	2	1	0	1	1	0
f_b	2	1	0	0	1	0	3	2	1	3	2	1	1	2	1
$LFSR_{257}$	R	R	R	R	R	R	R	NR	NR	R	R	NR	R	R	NR

Round 0: clock 0~3, Round 1: clock 4~5, Round 2: clock 6~8, Round 3: clock 9~11, Round 4: 12, Round 5: clock 13~14

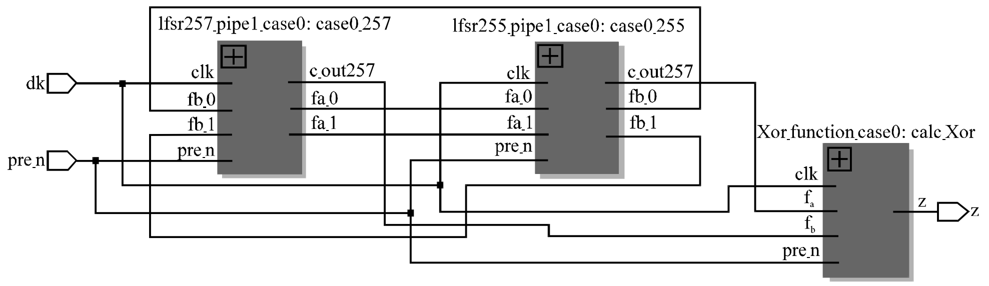

Fig. 3: Single channel PingPong 256 structure

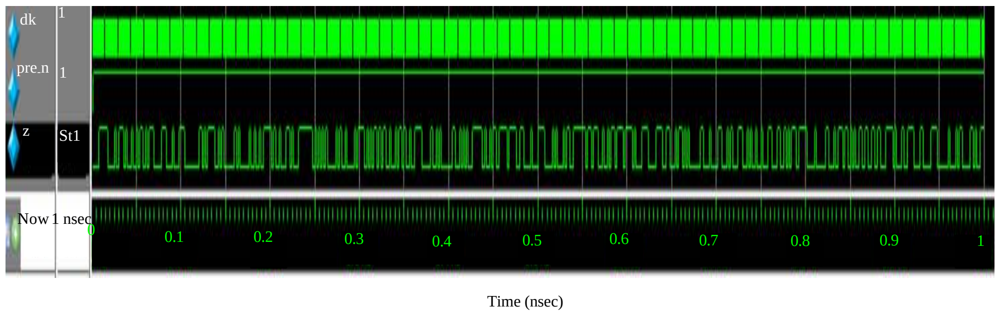

Fig. 4: Single-channel PingPong256 results from gate level test

For example, when each LFSR in channel A and each LFSR in channel B are mutually interconnected and a clock structure is connected to another channel, the order of operations and order of operation of the single channel depends on the state of the other channels. Because the variable clock structure performs the mixing of the sequence of operations without changing the period, it is not known without knowing the operation cycle of the LFSR at all. Also, it is difficult to pre-calculate and store the LFSR of PingPong 256 using a minimum of 255 bits.

Next, the LFSR used in PingPong 256 uses galois LFSRs to change several bits per clock. This can be used as a unique property because different types of LFSRs are used differently for different channels.

Finally, the LFSR has properties that have a maximum period, except when the bit streams are all initialized to ‘0’. This means that even if the LFSR is operated with multiple channels, the stored status value is not initialized to‘0’.

We have designed a new the parallel PingPong256 with four channels as shown in Fig. 5 to enable parallel operation by connecting the single channels using the above three characteristics.

Unlike the single channel PingPong 256, the parallel PingPong 256 uses the result of the variable clock function for the adjacent channel instead of the same channel and inputs the XOR function state of the other channel in addition to the input value of the XOR function.

Figure 6 shows gate level simulation result of PingPong 256 with 4 channels. The simulation result shows that the proposed structure can be used as a random number generator.

If you design a random number generator using this structure, you can improve the speed by using any number of single channel PingPong 256. Also, it is difficult to check the accumulated status value even if the initial value is exposed to the outside, so it is difficult to generate the same signal unless it is forced to initialize.

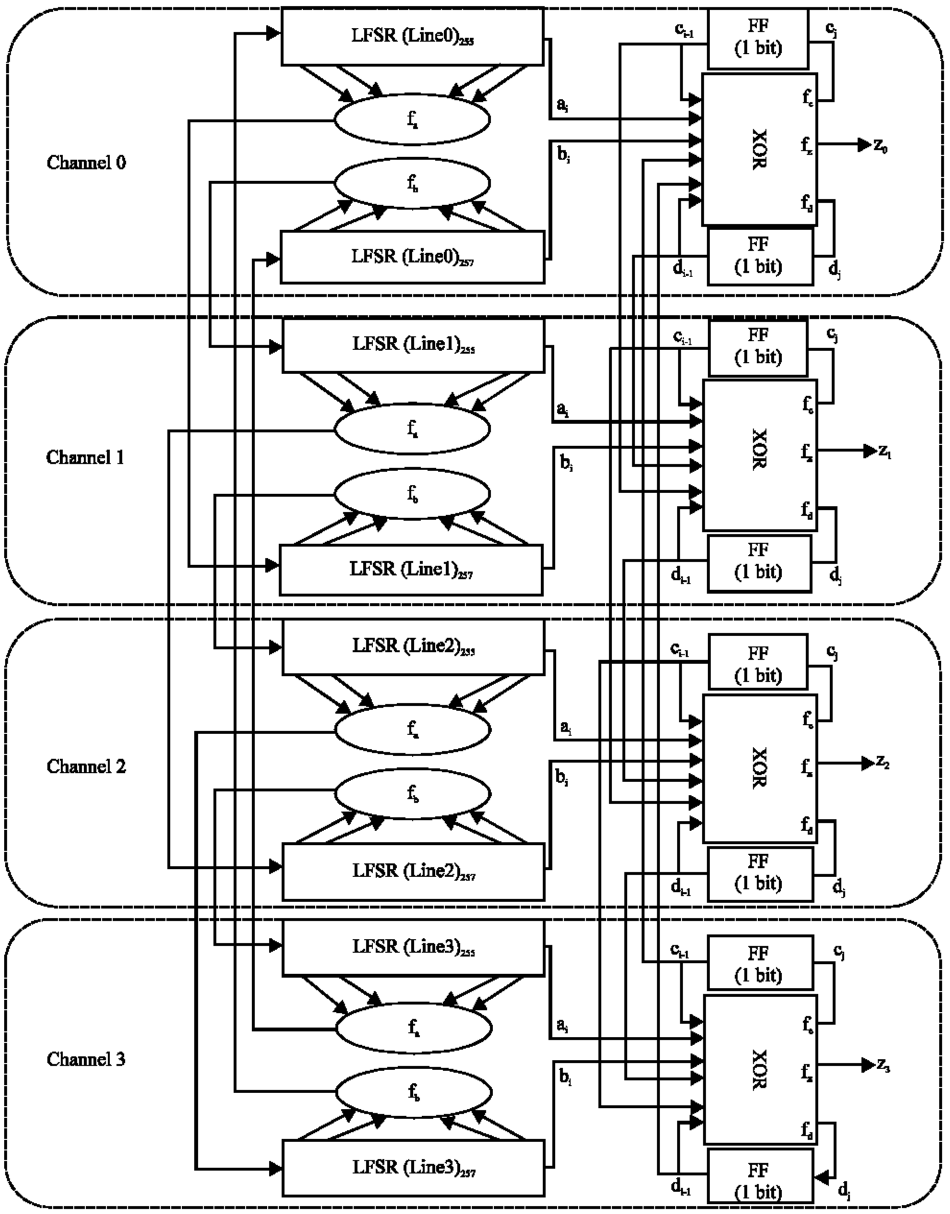

Fig. 5: Multi-channel PingPong 256 structure

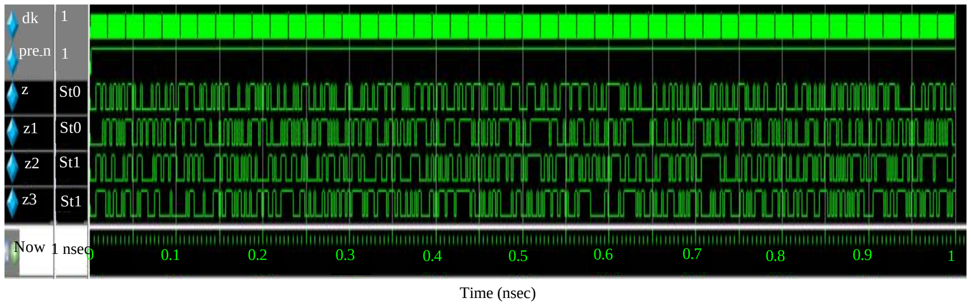

Fig. 6: Multi-channel PingPong 256 results from gate level test

CONCLUSION

Security is a must have element in all current communications technologies and is commonly used to protect or authenticate user information. In particular as Internet of Things (IoT) generation and mobile communication are developed, the company is constructing an environment to express and control the reality with digital information and it will create new value added by processing digital information collected later. However, since, the equipment requires mass production there are drawbacks that allow only low performance and limited communication. IoT device currently on sale is sending only fragmentary information such as temperature, humidity or gas and such information currently uses a block cipher code because the numbers have to be changed incrementally. The disadvantage at this time is that the time information can be inferred and the hardware implementation can be made possible at a lower cost using random numbers generators.

In this study, we explore linear complexity, cycles and structures of PingPonging 256 and the results of the simulations to implement single-channel and parallel channel architectures on hardware. Simulation results can be implemented in a virtual environment and may differ slightly from the physical environment but they are not expected to differ significantly. We also found that changes in computational structures in parallel structures had a large influence on output results. It is deemed necessary to experiment in expanding the operation channel to the same number of channels and methods to ensure reliability and safety of the encrypted text in the direction of the next study.

ACKNOWLEDGEMENTS

This research was supported by the Technological Innovation R&D Program (Assignment number: S2457495) funded by the Small and Medium Business Administration (SMBA, Korea) and This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (grant number: NRF-2016R1D1A1B01011908).

REFERENCES

Gollmann, D. and W.G. Chambers, 1989. Clock-controlled shift registers: A review. IEEE. J. Sel. Areas Commun., 7: 525-533.

Golomb, S.W., R.W. Lloyd, M.G. Richard and W.H. Alfred, 1982. Shift Register Sequences. Aegean Park Press, Laguna Hill, California, USA., ISBN:9780894120480, Pages: 247.

Key, E., 1976. An analysis of the structure and complexity of nonlinear binary sequence generators. IEEE. Trans. Inf. Theory, 22: 732-736.

Menezes, A., P. van Oorschot and S. Vanstone, 1996. Handbook of Applied Cryptography. 1st Edn., CRC Press, UK.

Rukhin, A., J. Soto, J. Nechvatal, M. Smid and E. Barker *et al.*, 2010. A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22 Revision 1a, National Institute of Standards and Technology, Gaithersburg, MD., April 2010, pp: 1-131.

Wenbo, M., 2003. Modern Cryptography: Theory and Practice. Prentice Hall, Upper Saddle River, New Jersey, USA., ISBN:978-81-317-0212-3, Pages: 721.