Fully Digital Chaotic Oscillators Applied to
Pseudo Random Number Generation

Thesis by
Abhinav S. Mansingka

Submitted in Partial Fulfillment of the Requirements for the
degree of
Master of Science

King Abdullah University of Science and Technology
Division of Physical Science and Engineering
Electrical Engineering Program

Thuwal, Makkah Province, Kingdom of Saudi Arabia
May, 2012
The undersigned approve the thesis of Abhinav S. Mansingka

<table>
<thead>
<tr>
<th>Dr. Mohamed-Slim Alouini</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Committee Member</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dr. Hossam A. H. Fahmy</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Committee Member</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dr. Khaled Nabil Salama</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Committee Chair (Thesis Supervisor)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

King Abdullah University of Science and Technology

2012
ABSTRACT

Fully Digital Chaotic Oscillators Applied to Pseudo Random Number Generation

Abhinav S. Mansingka

This thesis presents a generalized approach for the fully digital design and implementation of chaos generators through the numerical solution of chaotic ordinary differential equations. In particular, implementations use the Euler approximation with a fixed-point twos complement number representation system for optimal hardware and performance. In general, digital design enables significant benefits in terms of power, area, throughput, reliability, repeatability and portability over analog implementations of chaos due to lower process, voltage and temperature sensitivities and easy compatibility with other digital systems such as microprocessors, digital signal processing units, communication systems and encryption systems. Furthermore, this thesis introduces the idea of implementing multidimensional chaotic systems rather than 1-D chaotic maps to enable wider throughputs and multiplier-free architectures that provide significant performance and area benefits.

This work focuses efforts on the well-understood family of autonomous 3rd order “jerk” chaotic systems. The effect of implementation precision, internal delay cycles and external delay cycles on the chaotic response are assessed. Multiplexing of
parameters is implemented to enable switching between chaotic and periodic modes of operation. Enhanced chaos generators that exploit long-term divergence in two identical systems of different precision are also explored. Digital design is shown to enable real-time controllability of 1D multiscroll systems and 4th order hyperchaotic systems, essentially creating non-autonomous chaos that has thus far been difficult to implement in the analog domain.

Seven different systems are mathematically assessed for chaotic properties, implemented at the register transfer level in Verilog HDL and experimentally verified on a Xilinx Virtex 4 FPGA. The statistical properties of the output are rigorously studied using the NIST SP. 800-22 statistical testing suite. The output is adapted for pseudo random number generation by truncating statistically defective bits. Finally, a novel post-processing technique using the Fibonacci series is proposed and implemented with a non-autonomous driven hyperchaotic system to provide pseudo random number generators with high nonlinear complexity and controllable period length that enables full utilization of all branches of the chaotic output as statistically secure pseudo random output.
Acknowledgments

I would like to offer my sincerest thanks to Prof. Khaled N. Salama for guiding me in my research and providing me with an environment where I have been able to work productively over my time at the King Abdullah University of Science and Technology. His mentorship, keen sense of understanding and strong support for my work have been the driving forces that have enabled the completion of this thesis. I would also like to extend my heartfelt thanks to Dr. (now Prof.) Ahmed G. Radwan for introducing me to my research that connects nonlinear dynamics with digital circuit design. His keen insight and extremely positive demeanour motivated me to perform to the best of my ability in what was my first experience as a full-time member of a research group. Special thanks to M. Affan Zidan for showing me the ropes, always being available to help, providing valuable feedback and collaborating throughout the course of my research. Thanks also to Mohamed L. Barakat for teaching me about encryption and being a strong collaborator, ensuring that my research was actually practically applicable in security systems. To Profs. Hussein M. Alnuweiri and Hossam A. H. Fahmy, thank you for giving me the opportunity to be a teaching assistant for your computer architecture courses. To my thesis committee (Profs. Mohamed Slim-Alouini, Hossam A. H. Fahmy and Khaled N. Salama), thank you all for quick review of my thesis and for your invaluable time, feedback and support. This environment has made my time at KAUST intellectually stimulating and fruitful.

On a personal front, I would like to thank Dr. Amro M. Elshurafa for making my
time at work enjoyable and educational at the same time (a tough balance indeed), Hesham Omran for coming with me to Seoul to my very first technical conference and Justine Mink for ensuring that my coffee intake was far above the average for a normal graduate student. Further thanks to all my friends at KAUST (Said, Ben, Jenna, Jill, Al-Fadhel, Daniel, Al-Jeffery, Rasha, ... the list goes on and on) for ensuring that coming to the Kingdom of Saudi Arabia was an enriching experience. I wish you all the very best in all your future endeavours and am sure that the friendships forged at the coast of the Red Sea are built to stand the test of time for years to come.

Lastly, but certainly not the least, I would like to thank my family for their unconditional support and for providing me with a strong sense of purpose, duty and diligence. To my friends in India and across the globe (especially Sarjan, Aditya, Jay and Jay), thank you for always being there for me and pushing me to be a better man. As I transition to life in industry with Oracle in Silicon Valley, I am looking forward to new challenges and the long and winding road ahead.
# TABLE OF CONTENTS

**List of Abbreviations**

**List of Symbols**

**List of Illustrations**

**List of Tables**

I Introduction

I.1 Objectives and Contributions ........................................... 29

II Digital Chaos Generators

II.1 Description of Chaotic Systems ........................................... 31
   II.1.1 System 1: Modulus Nonlinearity ................................. 32
   II.1.2 System 2: Signum Nonlinearity ................................. 34
   II.1.3 System 3: Quadratic Nonlinearity .............................. 36
   II.1.4 System 4: Cubic Nonlinearity ................................ 39
   II.1.5 System 5: Maximum Function Nonlinearity ................. 41
   II.1.6 System 6: 1D Multiscroll - Staircase Nonlinearity ....... 43
   II.1.7 System 7: 4D Hyperchaos - Signum Nonlinearity .......... 46

II.2 Digital Implementation ................................................ 49
   II.2.1 System 1: Modulus Nonlinearity ............................... 51
II.2.2 System 2: Signum Nonlinearity ........................................ 52
II.2.3 System 3: Quadratic Nonlinearity ..................................... 54
II.2.4 System 4: Cubic Nonlinearity .......................................... 55
II.2.5 System 5: Maximum Function Nonlinearity ......................... 57
II.2.6 System 6: 1D Multiscroll - Staircase Nonlinearity ............... 58
II.2.7 System 7: 4D Hyperchaos - Signum Nonlinearity ................. 63
II.3 Area and Performance Results ............................................. 65

III. Chaotic Response: Analysis, Enhancement and Control .......... 68

III.1 System 2 - Signum Nonlinearity ....................................... 69

III.1.1 Implementation Bus Width ............................................ 69
III.1.2 Internal Delay Cycles .................................................. 71
III.1.3 External Delay Cycles .................................................. 72

III.2 System 6: 1D Multiscroll - Parameter Variation ................... 73

III.2.1 Number of Scrolls ..................................................... 73
III.2.2 Position of Scrolls ..................................................... 74
III.2.3 System Parameter ...................................................... 74
III.2.4 Euler Step Size ......................................................... 75

III.3 System 7: 4D Hyperchaos - Variation of Attractor Size .......... 76

III.4 Enhanced Chaos Generators ............................................ 77

III.4.1 New Attractors ......................................................... 78
III.4.2 Maximum Lyapunov Exponents ..................................... 78
III.4.3 Time Series and Frequency Response .............................. 81

III.5 Periodic and Chaotic Controllability ................................ 82

IV. Autonomous Chaos Generators as PRNGs ............................. 85

IV.1 Analysis of Short-Term Predictability ................................. 86

IV.1.1 Number of Integer Bits ............................................... 87
VII.4 Advance both orbits one iteration and calculate the new separation $d_1$.

VII.5 Evaluate $\log \frac{d_1}{d_0}$ in any convenient base.

VII.6 Readjust one orbit so its separation is $d_0$ in the same direction as $d_1$.

VII.7 Repeat steps 4-6 many times and calculate the average of step 5.

VIII. NIST SP. 800-22 Test Suite

VIII.1 Frequency (Monobit) Test

VIII.2 Block Frequency Test

VIII.3 Cumulative Sums Test

VIII.4 Runs Test

VIII.5 Longest Run Test

VIII.6 Rank Test

VIII.7 Discrete Fourier Transform (Spectral) Test

VIII.8 Non-Overlapping Template Matching Test

VIII.9 Overlapping Template Matching Test

VIII.10 Maurer’s “Universal Statistical” Test

VIII.11 Approximate Entropy Test

VIII.12 Random Excursions Test

VIII.13 Random Excursions Variant Test

VIII.14 Serial Test

VIII.15 Linear Complexity Test

References
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D</td>
<td>one dimensional</td>
</tr>
<tr>
<td>4D</td>
<td>four dimensional</td>
</tr>
<tr>
<td>ADD</td>
<td>binary addition</td>
</tr>
<tr>
<td>AND</td>
<td>bitwise logical AND</td>
</tr>
<tr>
<td>CSK</td>
<td>chaos-shift keying</td>
</tr>
<tr>
<td>DS-CDMA</td>
<td>direct-sequence code division multiple access</td>
</tr>
<tr>
<td>DSP</td>
<td>digital signal processing</td>
</tr>
<tr>
<td>FF</td>
<td>flip-flop</td>
</tr>
<tr>
<td>FFT</td>
<td>fast Fourier transform</td>
</tr>
<tr>
<td>FOM</td>
<td>figure of merit</td>
</tr>
<tr>
<td>FPGA</td>
<td>field programmable gate array</td>
</tr>
<tr>
<td>LFSR</td>
<td>linear feedback shift register</td>
</tr>
<tr>
<td>LUT</td>
<td>4-input look-up table</td>
</tr>
<tr>
<td>MIMO</td>
<td>multiple-input multiple-output</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>MLE</td>
<td>maximum Lyapunov exponent</td>
</tr>
<tr>
<td>MSB</td>
<td>most significant bit</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology (USA)</td>
</tr>
<tr>
<td>NLFSR</td>
<td>non-linear feedback shift register</td>
</tr>
<tr>
<td>ODE</td>
<td>ordinary differential equation</td>
</tr>
<tr>
<td>OR</td>
<td>bitwise logical OR</td>
</tr>
<tr>
<td>PRNG</td>
<td>pseudo random number generator</td>
</tr>
<tr>
<td>SUB</td>
<td>binary subtraction</td>
</tr>
<tr>
<td>TRNG</td>
<td>true random number generator</td>
</tr>
<tr>
<td>XOR</td>
<td>bitwise logical exclusive OR</td>
</tr>
</tbody>
</table>
List of Symbols

$s$  variable in the characteristic polynomial - denotes the eigenvalues at equilibrium points

$h$  the step size for the Euler approximation

$J$  the Jacobian matrix of a coupled system of differential equations

$\lambda$  Indicates the value of the maximum Lyapunov Exponent

$X_n$  $n$-th dynamical variable in a coupled system of differential equations
List of Illustrations

| II.1 | The modulus nonlinearity used for System 1. | 33 |
| II.2 | MATLAB simulations of (a) \( X_1 - X_2 \), (b) \( X_1 - X_3 \) and (c) \( X_2 - X_3 \) attractors of the modulus nonlinearity chaotic system using 40,000 iterations. | 34 |
| II.3 | The signum nonlinearity used for System 2. | 35 |
| II.4 | MATLAB simulations of (a) \( X_1 - X_2 \), (b) \( X_1 - X_3 \) and (c) \( X_2 - X_3 \) attractors of the signum nonlinearity chaotic system using 40,000 iterations. | 36 |
| II.5 | The quadratic nonlinearity used for System 3. | 37 |
| II.6 | MATLAB simulations of (a) \( X_1 - X_2 \), (b) \( X_1 - X_3 \) and (c) \( X_2 - X_3 \) attractors of the quadratic nonlinearity chaotic system using 40,000 iterations. | 38 |
| II.7 | The cubic nonlinearity used for System 4. | 40 |
| II.8 | MATLAB simulations of (a) \( X_1 - X_2 \), (b) \( X_1 - X_3 \) and (c) \( X_2 - X_3 \) attractors of the cubic nonlinearity chaotic system using 20,000 iterations showing both coexistent attractors. | 41 |
| II.9 | The maximum function nonlinearity used for System 5. | 42 |
| II.10 | MATLAB simulations of (a) \( X_1 - X_2 \), (b) \( X_1 - X_3 \) and (c) \( X_2 - X_3 \) attractors of the maximum function nonlinearity chaotic system using 20,000 iterations. | 43 |
II.11 The staircase nonlinearity used for System 6. .................................. 45
II.12 MATLAB simulations of $X_1 - X_2$, $X_1 - X_3$ and $X_2 - X_3$ attractors of the 1-D multiscroll system with staircase nonlinearity using 20,000 iterations showing (a)-(c) 3 scrolls ($L = 1, U = 5$) and (d)-(f) 7 scrolls ($L = -7, U = 5$). ..................................... 46
II.13 The signum nonlinearity used for System 7. ........................................ 48
II.14 MATLAB simulations of (a) $X_1 - X_2$, (b) $X_1 - X_3$, (c) $X_1 - X_4$, (d) $X_2 - X_3$, (e) $X_2 - X_4$, (f) $X_3 - X_4$ attractors of the 4-D hyperchaotic system with signum using 20,000 iterations. ........................................ 49
II.15 Generalized schematic for the digital implementation of $n$-th order coupled chaotic systems. ................................................................. 50
II.16 Digital Implementation of System 1 (Modulus Nonlinearity) ............... 51
II.17 Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 1 (Modulus nonlinearity) from a Xilinx Virtex 4 FPGA. .................................................. 52
II.18 Digital Implementation of System 2 (Signum Nonlinearity) ............... 53
II.19 Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 2 (Signum nonlinearity) from a Xilinx Virtex 4 FPGA. .................................................. 53
II.20 Digital Implementation of System 3 (Quadratic Nonlinearity) ............... 54
II.21 Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 3 (Quadratic nonlinearity) from a Xilinx Virtex 4 FPGA. .................................................. 55
II.22 Digital Implementation of System 4 (Cubic Nonlinearity) ............... 56
II.23 Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 4 (Cubic nonlinearity) from a Xilinx Virtex 4 FPGA. .................................................. 56
II.24 Digital Implementation of System 5 (Maximum Function Nonlinearity) 57

II.25 Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 5 (Maximum function nonlinearity) from a Xilinx Virtex 4 FPGA 58

II.26 (a) Digital Implementation of System 6 (1D Multiscroll with Staircase Nonlinearity) and detailed implementations of (b) the staircase nonlinearity and (c) the hardware multiplication for $\alpha$ and $h$ 59

II.27 Oscilloscope traces of $X_1 - X_2$ attractors from the digitally implemented chaos generator with different controllable parameters to yield different (a)-(d) Number of scrolls with $(L_{in}, h_{in}, a_{in}) = (0, 6, 15)$ and $U_{in} = [2, 5, 10]$, (e)-(h) Position of scrolls with $(h_{in}, a_{in}) = (6, 15)$, $L_{in} = [-5, -3, 1]$ and $U_{in} = L_{in} + 4, [5$ scrolls], (i)-(l) System parameter with $(L_{in}, U_{in}, h_{in}) = (-3, 2, 6)$ and $a_{in} = [3, 8, 11]$ and (m)-(p) Euler step size with $(L_{in}, U_{in}, a_{in}) = (-3, 2, 15)$ and $h_{in} = [2, 7, 10]$ respectively. Results are drawn on an oscilloscope from the output of a Xilinx Virtex 4 FPGA 61

II.28 Verilog simulation of 1,000,000 iterations of the digital chaos generator with $L_{in} = -63, U_{in} = 62, a_{in} = 15, h_{in} = 8$. The result is 126 scrolls centered around zero 63

II.29 Digital Implementation of System 7 (4D Hyperchaos with Signum Nonlinearity) 64

II.30 Bitwise breakdown illustrating the relationship between $D(t)$ and $D_{in}(t)$. 64

II.31 Oscilloscope traces of FPGA output of the attractors from the 4D digital chaos generator for (a) $X_1 - X_2$, (b) $X_1 - X_3$, (c) $X_1 - X_4$, (d) $X_2 - X_3$, (e) $X_2 - X_4$ and (f) $X_3 - X_4$. The traces are scaled to display the entire available phase space $[-1, 1]$ and $D(t) = D \approx 0.375$. 65
III.1 Oscilloscope output of digital $X_1 - X_2$ attractors of (a) 10-bit, (b) 12-bit, (c) 16-bit and (d) 32-bit implementations of System 2.

III.2 (a) Time evolution of MLE for 10, 12, 16, 32 and 64-bit bus width implementations of System 2 and (b) Saturated MLE of different bus width implementations of System 2.

III.3 (a) Effect of number of internal delay cycles on MLE for a 16-bit implementation of System 2 and (b) Oscilloscope output of a 16-bit $X_1 - X_2$ attractor for 18 internal delay cycles.

III.4 Oscilloscope output of $X_1 - X_2$ attractors for a 16-bit implementation of System 2 with delay in $X_1$ of (a) 10, (b) 20, (c) 30, and (d) 40 cycles.

III.5 Change in MLE v/s number of scrolls: $(L_{in}, h_{in}, \alpha_{in}) = (0, 6, 15)$ and $U_{in} = 2, 3, 4, ..., 33$ [3 – 34 scrolls].

III.6 Change in MLE v/s position of scrolls: $(h_{in}, \alpha_{in}) = (6, 15)$, $L_{in} = -7, -6, -5, ..., 8$ and $U_{in} = L_{in} + 7$ [8 scrolls].

III.7 Change in MLE v/s value of system parameter: $(L_{in}, U_{in}, h_{in}) = (0, 3, 6)$ and $\alpha_{in} = 0, 1, 2, ..., 15$ [4 scrolls].

III.8 Change in MLE v/s Euler step size: $(L_{in}, U_{in}, \alpha_{in}) = (0, 3, 15)$ and $h_{in} = 0, 1, 2, ..., 15$ [4 scrolls].

III.9 Output histograms using 10,000,000 samples using $D(t) = 0.25$ (red) and $D(t) \approx 0.375$ (blue) for (a) $X_1$, (b) $X_2$, (c) $X_3$ and (d) $X_4$.

III.10 Schematic of the enhanced chaos generators using arithmetic and logic operations.

III.11 Oscilloscope traces of $X_1 - X_2$ attractors of original 32-bit and addition, subtraction, bitwise AND, bitwise OR, and bitwise XOR of 32-bit and 16-bit implementations of (a)-(f) System 1, (g)-(l) System 2, (m)-(r) System 3 and (s)-(x) System 4 respectively.
III.12 Variation of MLE with bus width in (a) SUB and (b) XOR processed implementations of (i) System 1, (ii) System 2, (iii) System 3, (iv) System 4. A system with bus width shown on the x-axis is operated with a 32-bit implementation with the same initial conditions ($X_1 = 0, X_2 = 0.5, X_3 = 0$).

III.13 Oscilloscope output of time series and FFT of the (a)-(b) original 32-bit implementation, and (c)-(d) subtraction, and (e)-(f) bitwise XOR of the 32-bit and 16-bit implementations of System 3.

III.14 Oscilloscope trace of the output time waveforms of the $X_1$ variable illustrating controllable operation of (a) system 1, (b) system 2, (c) system 3 and (d) system 4. The output waveform of $X_1$ is shown in yellow and the control parameter $M$ is shown in blue. The results are experimental output from the FPGA.

IV.1 Defective Bits due to short-term predictability in $\{X, Y, Z\}$.

IV.2 No. of defective bits v/s $N_B$ with $D = 0.5$ and $h = 2^{-5}$.

IV.3 Number of defective bits versus bus width for different integer widths $N_I$ with $D = 0.5$ and $h$ from $2^{-5}$ to $2^{-9}$.

IV.4 Same scale oscilloscope trace from a Xilinx FPGA of $X - Y$ attractors with (a) $D = 1.375$, (b) $D = 2.5$ and (c) $D = 3.375$.

IV.5 Maximum absolute value of $\{X, Y, Z\}$ for $D$ from 0.3 to 20.

IV.6 Number of defective bits versus value of $D$ for different $N_I$ with $N_B = 32$ and $h = 2^{-5}$. $D$ is valued such that the resulting attractor is guaranteed to be bounded within the phase space defined by each $N_I$.

V.1 Top-level schematic of the proposed PRNG based on non-autonomous hyperchaos and Fibonacci series post-processing.
V.2 256-bit LFSR using the feedback polynomial from (V.1) to generate a maximum-length sequence.

V.3 Schematic of the proposed post processor with a constant rotation in Loop 1 and a variable rotation controlled by the Fibonacci series in Loop 2.

V.4 Schematic of the variable rotation using the Fibonacci series. A 64-bit barrel shifter is used in conjunction with a state machine that cycles through the appropriate $k$-values from (V.5).

V.5 Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$, (c) $X_1 - X_4$, (d) $X_2 - X_3$, (e) $X_2 - X_4$, (f) $X_3 - X_4$, (g) $Y_1 - Y_2$, (h) $Y_1 - Y_3$, (i) $Y_1 - Y_4$, (j) $Y_2 - Y_3$, (k) $Y_2 - Y_4$, (l) $Y_3 - Y_4$ attractors of the (a)-(f) original hyperchaotic system with LFSR forcing and (g)-(l) Fibonacci post-processed output respectively.

V.6 Oscilloscope traces of the time series and frequency response of (a)-(b) $X_1$ [Original hyperchaotic system with LFSR forcing] and (c)-(d) $Y_1$ [After Fibonacci Post-Processing].

V.7 Output histograms using 10,000,000 samples for the LFSR-driven hyperchaotic system without post-processing (red) and after post-processing (blue) for (a) $X_1$ and $Y_1$, (b) $X_2$ and $Y_2$, (c) $X_3$ and $Y_3$, (d) $X_4$ and $Y_4$.

V.8 Cumulative percentage difference in output bitstreams over time for two different executions of the system using different keys ($K_{PRNG} = 2$ and $K_{PRNG} = 3$) (a) original hyperchaotic system and (b) after Fibonacci Post-Processing. The ideal value over the long-term is 50%.

V.9 Entropy of each output bitstream for (a) the original hyperchaotic system with LFSR forcing and (b) after applying the proposed Fibonacci post-processing. $K_{PRNG} = 1$ is used.
List of Tables

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>II.1</td>
<td>Area and Performance Results on the Xilinx Virtex 4 FPGA for all seven implemented systems.</td>
<td>66</td>
</tr>
<tr>
<td>III.1</td>
<td>Maximum Lyapunov exponent for Chaotic Systems. All processed systems have a 32-bit bus width.</td>
<td>80</td>
</tr>
<tr>
<td>III.2</td>
<td>Optimized Parameters for Periodic and Chaotic Systems</td>
<td>82</td>
</tr>
<tr>
<td>IV.1</td>
<td>NIST SP. 800-22 test results for the original systems 1 to 5 along with experimental area and throughput results from the Xilinx Virtex 4 FPGA</td>
<td>91</td>
</tr>
<tr>
<td>IV.2</td>
<td>NIST SP. 800-22 test results for chaotic systems 1 to 5 after truncation of defective bits along with experimental area and throughput results from the Xilinx Virtex 4 FPGA</td>
<td>92</td>
</tr>
<tr>
<td>IV.3</td>
<td>Comparison to previous chaos-based PRNGs</td>
<td>93</td>
</tr>
<tr>
<td>V.1</td>
<td>NIST SP. 800-22 test results for the original hyperchaotic system (with 256-bit LFSR) and the corresponding Fibonacci post-processed output along with experimental area and throughput results from the Xilinx Virtex 4 FPGA</td>
<td>106</td>
</tr>
<tr>
<td>V.2</td>
<td>NIST SP. 800-22 test results for the original hyperchaotic system (with 256-bit LFSR) and the corresponding Fibonacci post-processed output along with experimental area and throughput results from the Xilinx Virtex 4 FPGA.</td>
<td>107</td>
</tr>
</tbody>
</table>
Chapter I

Introduction

Chaos theory has been well-studied in mathematics and covers a broad spectrum of systems that exhibit strong sensitivity to initial conditions while still maintaining bounded solutions. The earliest work in chaos began as a subset of ergodic theory inspired by physical phenomena through the works of Poincaré, Birkhoff (three-body problem), Kolmogorov (turbulence and astronomy), Cartwright and Littlewood (radio engineering) \[1\] \[2\] \[3\] \[4\]. Subsequent work by Lorenz \[5\], Mandelbrot \[6\] and others played key roles in formalizing the mathematical definitions of chaos theory arising from nonlinear systems. In particular, “Chua’s circuit” \[7\] was the first physical observation of chaos on an electronic circuit and since then, a wealth of literature has been presented on the subject of harnessing chaotic dynamics for a wide variety of applications.

In recent years, chaos theory has been exploited in digital communication systems \[8\] for applications in chaos-shift keying (CSK) \[9\] \[10\] \[11\] \[12\] \[13\] \[14\], direct-sequence code division multiple access (DS-CDMA) \[15\] \[16\] \[17\], pulse-position baseband modulation \[18\] and multiple-input multiple-output (MIMO) communications \[19\]. There has also been a tremendous thrust in applying chaos to information security \[20\] \[21\] \[22\] \[23\] \[24\] \[25\] \[26\], at the heart of which is the design of random
number generators [27], emphasizing application of chaos for both pseudo random number generators (PRNGs) [28, 29, 30, 31, 32, 33] and true random number generators (TRNGs) [34, 35, 36, 37, 38, 39]. Hardware implementations of random number generators are in general, necessary in a broad field of applications including stochastic simulations [40] and are becoming increasingly common in microprocessors for on-chip security solutions [41] and also to facilitate easy availability of statistically robust random numbers for use by software [42, 43].

Most circuit implementations of chaos have emphasized the analog domain [44, 37, 45, 46, 47, 48, 49, 50, 51]. It has been shown that tolerances in analog realizations remarkably degrade the quality of the chaos generator output [52]. Furthermore, analog chaos requires large capacitors for the storing of state variables and have several issues with environmental sensitivity. In general, digital design provides benefits over analog implementation in terms of area efficiency, repeatability, portability, power consumption, and integrability with integrated circuit technology [43]. On that note, the digital implementation of chaos has focused exclusively on 1D chaotic maps [28, 29, 30, 31, 33] for applications as PRNGs. From a dynamical perspective, 1D maps are limited in that they suffer from windows of periodicity and provide only a single dimensional output. Overcoming statistical defects requires significant analysis, uses additional hardware and lowers the effective achievable throughput. Furthermore, 1D chaotic maps like the logistic map are reliant on multipliers for non-linear characteristics and thus incur a significant throughput penalty due to the computational complexity of hardware multiplication [31].

To overcome these limitations, this thesis considers higher order chaotic systems that provide multidimensional throughput while simultaneously enabling the possibility of multiplier-free architectures for high-performance systems. The search for such chaotic systems begins at the Poincaré-Bendixson Theorem [53], which states that the existence of chaotic dynamics in systems of nonlinear ordinary differential
equations (ODEs) requires that those systems be of at least the third order. As a result, several third order systems, notably the Lorenz [5], Rössler [54], and Chen [55] systems have been presented. A special class of so-called “jerk” systems was exhaustively explored by Sprott [56, 57, 58, 59] and is of the form \( \dddot{x} = J(\dddot{x}, \dot{x}, x) \), wherein the term “jerk” arises from the mechanical nomenclature of the third derivative of distance with respect to time (i.e. time derivative of acceleration) as the “jerk”. These systems represent algebraically simple chaotic flows that are the easiest to implement on-chip [56, 60, 61]. Multi-scroll chaos is also well understood [62, 63, 64, 65, 66, 67, 68, 69] and has been realized through analog circuits [62, 67, 70] but the limited dynamic range of analog components and low supply voltages has restricted the number of scrolls that can be realized [71] unless mixed-signal approaches are used [72]. Lastly, non-autonomous (i.e. driven) chaotic systems have been extensively studied and implemented in analog form using periodic and pulse-excitation based forcing [37, 45, 46, 47, 48, 49] with demonstrated applications as analog TRNGs [37]. Hyperchaotic systems have also been studied [73] and physically implemented for multi-torus chaos generation [74]. In particular, 4th order hyperchaotic systems are unstable in two directions and have complex and deterministic but yet unpredictable dynamics that make them especially desirable for the applications suggested above. However, progress in implementing such systems in hardware has been limited [74].

The possible area efficiency, high throughput and relatively easy digital implementation of chaos-based PRNGs motivates good post-processing techniques to overcome statistical flaws in the output. The Von Neumann technique [75], XOR correctors [76, 27], truncation of defective bits [32], hash-function post processing [77, 42], and linear code correctors [78, 79] are some known solutions to enhance random properties of PRNGs [80]. While most previous solutions resolve statistical defects, none of them preserves the raw PRNG/TRNG throughput and some incur a huge hardware
overhead and furthermore, none of the previously demonstrated techniques guarantees high differential sensitivity. This thesis considers a new approach to hardware post-processing using feedback loops, XOR operations, rotation and the Fibonacci series. Overall, chaos-based PRNGs are particularly attractive for cryptographic applications because most well-known and mathematically rigorous PRNGs such as the Mersenne Twister [81], DX generators [82] and linear feedback shift registers (LFSRs) are all linear and thus predictable after a certain number of iterations are observed, even though the output has exceedingly long period. Cryptographically secure non-linear PRNGs such as the non-linear feedback shift registers (NLFSRs) [83] and the Blum-Blum-Shub generator [84] suffer from other issues in design and implementation and are difficult to accurately characterize.

The organization of this thesis is as follows: Seven different chaotic systems are considered, five of which are classical “jerk” systems [56] that use (i) modulus, (ii) signum, (iii) quadratic, (iv) cubic and (v) maximum function nonlinearities, the sixth is a non-autonomous 1D multiscroll system [68] (also a jerk system) and the last is a 4D non-autonomous hyperchaotic system first proposed in [73] that uses a signum nonlinearity. Chapter 2 first introduces the mathematics behind each system and performs stability analysis that verifies the existence of chaotic dynamics through the observation of bounded strange attractors in phase space and through the calculation of the maximum Lyapunov exponent (MLE). A positive MLE confirms the presence of chaotic dynamics as it indicates that two different solutions that begin at arbitrarily close initial conditions will diverge in time exponentially [85, 86] according to the expression \( \delta S(t) \approx e^{\lambda t} \delta S(0) \), where \( \lambda \) is the MLE. Larger MLEs imply greater unpredictability and enhanced chaotic sensitivity. As such, once a positive MLE is observed for the output time series, this thesis concludes that chaotic dynamics are present. A brief background on the procedure used to estimate the MLE is shown in Appendix A. The rest of Chapter 2 addresses the physical implementation of these
systems in the digital domain using the Euler approximation and a fixed-point representation scheme in a nonlinear feedback pipelined architecture. The designs in this thesis emphasize simplicity and the use of standard components such as registers, adders, multiplexors, multipliers and basic logic gates at the register transfer level to facilitate easy portability to all the application spaces mentioned above. All circuits are synthesized and experimentally verified on a Xilinx Virtex 4 XC4VSX35-10FF668 field programmable gate array (FPGA).

Chapter 3 conducts chaotic response analysis by examining the effect of the implementation bus width, internal delay cycles and external delay cycles in System 2 (signum nonlinearity) [87], the effect of variations in number of scrolls, Euler step size, system parameter and position of scrolls on the chaotic response for the non-autonomous 1D multiscroll system [88] and the effect of the controllable parameter on attractor size for system 7 (4D hyperchaos). Subsequently, the section discusses enhanced chaotic oscillators that exploit the long-term divergence in identical systems caused by different implementation precisions using jerk systems (1-4). Two identical systems with the same initial conditions are mixed using arithmetic (ADD, SUB) and bitwise logical (AND, OR, XOR) operations to yield new chaotic attractors that have never been demonstrated before and arise purely from the digital implementation scheme as they exploit differential precisions. These new chaotic attractors have no correspondence in the analog domain and provide a significant chaos enhancement as measured by the MLE. Lastly, this chapter also looks into simple multiplexing of system parameters to enable switching between chaotic and periodic modes of operation in real-time through a single controllable parameter and presents a successful hardware implementation of the same.

Chapter 4 uses the NIST SP. 800-22 test suite [89] to assess the statistical randomness of the chaotic output from the autonomous systems (1-5). The detailed descriptions of the tests in the NIST suite are shown in Appendix B. Short-term
predictability is analyzed and minimized such that the largest possible statistically random throughput is achieved. PRNGs are created using the chaos generators by truncating the statistically defective bits [32]. The resulting PRNGs are compared to previous work in the literature, indicating significantly superior performance. However, PRNGs based on autonomous chaotic systems have a significantly large number of defective bits, leading to a significant throughput penalty caused by truncation. Furthermore, both the key-space and the period length of the system are completely dependent on the implementation bus width. From both perspectives, larger bus widths are desirous for longer periods and larger key-spaces. However, given the pipelined digital architectures in this thesis, increases in bus width reduce the operating frequency of the system and thus eventually diminish area-efficiency as bus widths become very large. From an analytical perspective, it is very difficult to estimate the period of nonlinear chaotic systems [90, 91]. Brute-force techniques have only gone so far as period lengths of \(2^{32}\) and thus this aspect of analysis is very challenging to address mathematically.

Chapter 5 considers the limitations in autonomous chaos-based PRNGs by focusing on the non-autonomous 4D hyperchaotic system (system 7). In general, hardware PRNGs are evaluated on the basis of period length, area, throughput, performance against statistical randomness tests and the size of the key-space. A 256-bit LFSR is used to drive the hyperchaotic circuit, thus decoupling the key space from the implementation width of the chaotic oscillator. Furthermore, a lower bound of periodicity is guaranteed and is specified by the periodicity of the LFSR, thus far surpassing the period of autonomous chaotic systems and simplifying periodicity analysis. A new post-processing technique based on the Fibonacci series, XOR correction and feedback is presented that suppresses statistical flaws while enhancing system sensitivity to small changes in the driving function of the non-autonomous system, strengthening differential resistance. This system is compared against previous literature as
in Chapter 4 while also comparing period length and key-space and assessing the non-autonomous hyperchaos-based PRNG in relation to the autonomous ones.

Finally, the conclusion summarizes the results of this thesis. It also provides a list of publications that have arisen from the research conducted on the digital implementation of chaotic oscillators. Lastly, it outlines concepts that can be explored in future research in this area.

I.1 Objectives and Contributions

The main objective of this thesis is to establish a generalized framework for the digital implementation of chaotic systems of differential equations and subsequently apply these systems for secure pseudo random number generation.

The main contributions of this thesis include:

- A general framework for the fully digital design of higher order systems of chaotic ordinary differential equations.
- Physical FPGA implementations of autonomous and non-autonomous third and fourth order chaotic systems.
- The analysis of the effect of implementation parameters on the chaotic response of the realized systems.
- The presentation of enhanced chaotic systems that are created by mixing two identical systems of different implementation bs widths to exploit the long-term divergence caused by differences in precision.
- The presentation of chaotic oscillators that can controllably switch between chaotic and sinusoidal output in real-time.
- The analysis of short-term predictability of chaotic systems as it pertains to the fully digital implementation of those systems by examining the number of statistically
defective bits for different implementation parameters through the NIST SP. 800-22 test suite.

- The implementation of autonomous chaos-based PRNGs by truncating defective bits to yield statistically random output that passes all NIST SP. 800-22 tests.
- The implementation of a robust non-autonomous hyperchaos-based PRNG that is driven by a 256-bit LFSR and is post-processed using a novel technique based on the Fibonacci series, bitwise XOR, rotation and feedback.
Chapter II

Digital Chaos Generators

II.1 Description of Chaotic Systems

In this thesis, a total of seven basic chaotic systems are digitally implemented, each of which uses a different nonlinear function to yield different chaotic dynamics. This section assesses the mathematical properties of each of the chaotic systems under study to provide basic intuition regarding the nature of the expected dynamics. Five third order systems are investigated that have (i) modulus, (ii) signum, (iii) quadratic, (iv) cubic and (v) maximum function nonlinearities, all of which have been previously presented in [56] and constitute the well-known class of “jerk” equation based chaotic systems. A 1D multiscroll system based on the staircase nonlinearity [68] is also implemented with real-time controllability of the number and position of scrolls, system parameter, and Euler step size. This thesis also presents a simple 4th order hyperchaotic system based on the signum nonlinearity that was first proposed in [73] that is made non-autonomous by introducing a controllable parameter that varies attractor size in real-time. In particular, the mathematical focus is on the location of equilibrium points, the Jacobian (i.e. linearization) at those points, the eigenvalues and corresponding expected dynamics, the shape of the chaotic attractors (phase
portraits) and the estimation of the maximum Lyapunov exponent.

### II.1.1 System 1: Modulus Nonlinearity

This system is described by the following state-space matrix representation of three first order ordinary differential equations:

\[
\begin{bmatrix}
    \dot{X}_1 \\
    \dot{X}_2 \\
    \dot{X}_3 
\end{bmatrix} = 
\begin{bmatrix}
    0 & 1 & 0 \\
    0 & 0 & 1 \\
    0 & -0.5 & -0.25 
\end{bmatrix}
\begin{bmatrix}
    X_1 \\
    X_2 \\
    X_3 
\end{bmatrix} 
+ 
\begin{bmatrix}
    0 \\
    0 \\
    0.25|X_1| - 0.5 
\end{bmatrix} 
\tag{II.1}
\]

Solving the above system for the steady-state solution where \((\dot{X}_1, \dot{X}_2, \dot{X}_3) = (0, 0, 0)\) gives the equilibrium points as:

\[
\begin{bmatrix}
    X_1^* \\
    X_2^* \\
    X_3^* 
\end{bmatrix} = 
\begin{bmatrix}
    \pm 2 \\
    0 \\
    0 
\end{bmatrix} 
\tag{II.2}
\]

The Jacobian at these points is given by the linearization of the chaotic system:

\[
J_{\pm 2,0,0} = 
\begin{bmatrix}
    0 & 1 & 0 \\
    0 & 0 & 1 \\
    \pm 0.25 & -0.5 & -0.25 
\end{bmatrix} 
\tag{II.3}
\]

The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the stability of the proposed chaotic system. The eigenvalues are obtained by determining the roots of the characteristic equation of the Jacobian:

\[
s^3 + 0.25s^2 + 0.5s \mp 0.25 = 0 
\tag{II.4a}
\]

\[
s_{(+2,0,0)} = \{0.351, -0.301 \pm 0.788i\} 
\tag{II.4b}
\]
\[ s_{(-2,0,0)} = \{-0.432, 0.091 \pm 0.755i\} \] (II.4c)

The results indicate that \( s_{(+2,0,0)} \) is a **spiral saddle point of index 1** due to one positive real eigenvalue and a pair of complex conjugate eigenvalues with negative real parts. Similarly, \( s_{(-2,0,0)} \) is a **spiral saddle point of index 2** due to one negative real eigenvalue and a pair of complex conjugate eigenvalues with positive real parts [47]. The positive real parts in the complex eigenvalues at \((-2,0,0)\) will enable oscillation around that point, while the positive real eigenvalue will enable repulsion from \((+2,0,0)\).

The nonlinear function \(|X_1|\) is shown in Figure II.1, wherein it is piecewise linear and a single discontinuity at the origin provides nonlinear characteristics.

```
|X_1|
```

**Figure II.1**: The modulus nonlinearity used for System 1.

MATLAB simulations of the chaotic system are shown in Figure II.2 using the Euler approximation with a step size of \(2^{-5}\), indicating chaos by visual inspection due to the presence of a strange attractor. Nonlinear time series analysis using the software based on [85] gives the MLE as 0.052, thus confirming the presence of chaotic dynamics. The attracting equilibrium point at \((-2,0,0)\) is clear from Figure II.2(a) and Figure II.2(c). In particular, the resulting chaotic attractor shows good correspondence with the analog attractor that has been shown in [56].
II.1.2 System 2: Signum Nonlinearity

This system is described by the following state-space matrix representation of three first order ordinary differential equations:

\[
\begin{bmatrix}
\dot{X}_1 \\
\dot{X}_2 \\
\dot{X}_3
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-1 & -1 & -0.5
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
2\text{sgn}(X_1)
\end{bmatrix}
\] (II.5)

Solving the above system for the steady-state solution where \((\dot{X}_1, \dot{X}_2, \dot{X}_3) = (0, 0, 0)\) gives the equilibrium points as:

\[
\begin{bmatrix}
X_1^* \\
X_2^* \\
X_3^*
\end{bmatrix} =
\begin{bmatrix}
0, \pm 2 \\
0 \\
0
\end{bmatrix}
\] (II.6)

The Jacobian at these points is given by the linearization of the chaotic system:

\[
J_{(0,0,0)} = \lim_{k \to \infty} \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
k & -1 & -0.5
\end{bmatrix} \quad J_{(\pm 2,0,0)} = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-1 & -1 & -0.5
\end{bmatrix}
\] (II.7)
The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the
stability of the proposed chaotic system. The eigenvalues are obtained by determining
the roots of the characteristic equation of the Jacobian:

\[ s^3 + 0.5s^2 + s - k = 0, \quad s^3 + 0.5s^2 + s + 1 = 0 \]

\[ s_{(0,0,0)} = \{\mu, -\nu \pm \sigma i\}, \quad \mu, \nu > 0 \] (II.8b)

\[ s_{(\pm2,0,0)} = \{-0.804, 0.152 \pm 1.105i\} \] (II.8c)

The results indicate that qualitatively, \( s_{(0,0,0)} \) is a spiral saddle point of index 1
due to one positive real eigenvalue and a pair of complex conjugate eigenvalues with
negative real parts. Similarly, \( s_{(\pm2,0,0)} \) are spiral saddle points of index 2 due to one
negative real eigenvalue and a pair of complex conjugate eigenvalues with positive
real parts [17]. The positive real parts in the complex eigenvalues at \((\pm2,0,0)\) will
enable oscillation around those points, while the positive real eigenvalue will enable
repulsion from \((0,0,0)\). The nonlinear function \( \text{sgn}(X_1) \) is shown in Figure II.3
wherein it is piecewise linear and a single discontinuity at the origin provides nonlinear
characteristics.

![Figure II.3: The signum nonlinearity used for System 2.](image)

MATLAB simulations of the chaotic system are shown in Figure II.4 using the
Euler approximation with a step size of $2^{-5}$, indicating chaos by visual inspection due to the presence of a strange attractor. Nonlinear time series analysis using the software based on [85] gives the MLE as 0.155, thus confirming the presence of chaotic dynamics. The attracting equilibrium points at $(\pm 2, 0, 0)$ are clear from Figure II.4(a) and Figure II.4(c). In particular, the resulting chaotic attractor shows good correspondence with the analog attractor that has been shown in [56].

Figure II.4: MATLAB simulations of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of the signum nonlinearity chaotic system using 40,000 iterations.

II.1.3 System 3: Quadratic Nonlinearity

This system is described by the following state-space matrix representation of three first order ordinary differential equations:

$$
\begin{bmatrix}
\dot{X}_1 \\
\dot{X}_2 \\
\dot{X}_3
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & -2 & -0.5
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
-0.5(X_1)^2 + 2
\end{bmatrix}
$$

(SII.9)

Solving the above system for the steady-state solution where $(\dot{X}_1, \dot{X}_2, \dot{X}_3) =$
$(0, 0, 0)$ gives the equilibrium points as:

$$
\begin{bmatrix}
X_1^* \\
X_2^* \\
X_3^*
\end{bmatrix} =
\begin{bmatrix}
\pm 2 \\
0 \\
0
\end{bmatrix}
$$

(II.10)

The Jacobian at these points is given by the linearization of the chaotic system:

$$
J_{(\pm 2, 0, 0)} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
\mp 2 & -2 & -0.5
\end{bmatrix}
$$

(II.11)

The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the stability of the proposed chaotic system. The eigenvalues are obtained by determining the roots of the characteristic equation of the Jacobian:

$$
s^3 + 0.5s^2 + 2s \pm 2 = 0
$$

(II.12a)

$$
s_{(+2,0,0)} = \{-0.864, 0.182 \pm 1.510i\}
$$

(II.12b)

Figure II.5: The quadratic nonlinearity used for System 3.
\[
s_{(-2,0,0)} = \{0.703, -0.601 \pm 1.576i\}
\]  

(II.12c)

The results indicate that qualitatively, \( s_{(-2,0,0)} \) is a spiral saddle point of index 1 due to one positive real eigenvalue and a pair of complex conjugate eigenvalues with negative real parts. Similarly, \( s_{(+2,0,0)} \) is spiral saddle point of index 2 due to one negative real eigenvalue and a pair of complex conjugate eigenvalues with positive real parts \[47\]. The positive real parts in the complex eigenvalues at \((+2,0,0)\) will enable oscillation around that point, while the postive real eigenvalue will enable repulsion from \((-2,0,0)\). The nonlinear function \((X_1)^2\) is shown in Figure II.5 wherein this particular function is nonlinear throughout its domain.

MATLAB simulations of the chaotic system are shown in Figure II.6 using the Euler approximation with a step size of \(2^{-5}\), indicating chaos by visual inspection due to the presence of a strange attractor. Nonlinear time series analysis using the software based on \[85\] gives the MLE as 0.089, thus confirming the presence of chaotic dynamics. The attracting equilibrium point at \((+2,0,0)\) is clear from Figure II.6(a) and Figure II.6(c). In particular, the resulting chaotic attractor shows good correspondence with the analog attractor that has been shown in \[56\].

Figure II.6: MATLAB simulations of (a) \(X_1 - X_2\), (b) \(X_1 - X_3\) and (c) \(X_2 - X_3\) attractors of the quadratic nonlinearity chaotic system using 40,000 iterations.
II.1.4 System 4: Cubic Nonlinearity

This system is described by the following state-space matrix representation of three first order ordinary differential equations:

\[
\begin{bmatrix}
\dot{X}_1 \\
\dot{X}_2 \\
\dot{X}_3
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0.5 & -1 & -0.5
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
-2^{-5}(X_1)^3
\end{bmatrix}
\]  

(II.13)

Solving the above system for the steady-state solution where \((\dot{X}_1, \dot{X}_2, \dot{X}_3) = (0, 0, 0)\) gives the equilibrium points as:

\[
\begin{bmatrix}
X_1^* \\
X_2^* \\
X_3^*
\end{bmatrix} =
\begin{bmatrix}
0, \pm 4 \\
0 \\
0
\end{bmatrix}
\]  

(II.14)

The Jacobian at these points is given by the linearization of the chaotic system:

\[
J_{(0,0,0)} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0.5 & -1 & -0.5
\end{bmatrix}
J_{(\pm4,0,0)} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-1 & -1 & -0.5
\end{bmatrix}
\]  

(II.15)

The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the stability of the proposed chaotic system. The eigenvalues are obtained by determining the roots of the characteristic equation of the Jacobian:

\[
s^3 + 0.5s^2 + s - 0.5 = 0, \quad s^3 + 0.5s^2 + s + 1 = 0
\]  

(II.16a)

\[
s_{(0,0,0)} = \{0.378, -0.438 \pm 1.067i\}
\]  

(II.16b)

\[
s_{(\pm4,0,0)} = \{-0.804, 0.152 \pm 1.105i\}
\]  

(II.16c)
The results indicate that qualitatively, $s_{(0,0,0)}$ is a spiral saddle point of index 1 due to one positive real eigenvalue and a pair of complex conjugate eigenvalues with negative real parts. Similarly, $s_{(\pm 4,0,0)}$ are spiral saddle points of index 2 due to one negative real eigenvalue and a pair of complex conjugate eigenvalues with positive real parts. The positive real parts in the complex eigenvalues at $(\pm 4,0,0)$ will enable oscillation around that point, while the positive real eigenvalue will enable repulsion from $(0,0,0)$. The nonlinear function $(X_1)^3$ is shown in Figure II.7 wherein this particular function is nonlinear throughout its domain.

![Figure II.7: The cubic nonlinearity used for System 4.](image)

MATLAB simulations of the chaotic system are shown in Figure II.8 using the Euler approximation with a step size of $2^{-3}$, indicating chaos by visual inspection due to the presence of a strange attractor. Nonlinear time series analysis using the software based on [85] gives the MLE as 0.058, thus confirming the presence of chaotic dynamics. Since a large enough step size is used, both saddle points on index 2 contribute to the chaotic attractor. In ordinary circumstances, the attracting equilibrium point is determined by the initial condition, resulting in two coexistent attractors within the same system. For this case, the attracting equilibrium points are clearly observable at $(\pm 4,0,0)$ from Figure II.8(a) and Figure II.8(c).
II.1.5 System 5: Maximum Function Nonlinearity

This system is described by the following state-space matrix representation of three first order ordinary differential equations:

\[
\begin{bmatrix}
\dot{X}_1 \\
\dot{X}_2 \\
\dot{X}_3
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & -1 & -0.5
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix}
+ 
\begin{bmatrix}
0 \\
0 \\
-8\max(X_1, 0) + 0.25
\end{bmatrix}
\] (II.17)

Solving the above system for the steady-state solution where \((\dot{X}_1, \dot{X}_2, \dot{X}_3) = (0, 0, 0)\) gives the equilibrium points as:

\[
\begin{bmatrix}
X_1^* \\
X_2^* \\
X_3^*
\end{bmatrix} =
\begin{bmatrix}
0.03125 \\
0 \\
0
\end{bmatrix}
\] (II.18)

The Jacobian at this point is given by the linearization of the chaotic system:

\[
J_{(0,0.03125,0,0)} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-8 & -1 & -0.5
\end{bmatrix}
\] (II.19)

Figure II.8: MATLAB simulations of (a) \(X_1 - X_2\), (b) \(X_1 - X_3\) and (c) \(X_2 - X_3\) attractors of the cubic nonlinearity chaotic system using 20,000 iterations showing both coexistent attractors.
The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the stability of the proposed chaotic system. The eigenvalues are obtained by determining the roots of the characteristic equation of the Jacobian:

\[ s^3 + 0.5s^2 + s + 8 = 0 \]  

(II.20a)

\[ s_{(0.03125,0,0)} = \{-2, 0.75 \pm 1.854i\} \]  

(II.20b)

The results indicate that \( s_{(0.03125,0,0)} \) is spiral saddle point of index 2 due to one negative real eigenvalue and a pair of complex conjugate eigenvalues with positive real parts \[17\]. The positive real parts in the complex eigenvalues at \((0.03125,0,0)\) will enable oscillation around that point and form a chaotic attractor. The nonlinear function \( \max(X_1,0) \) is shown in Figure II.9 wherein it is piecewise linear and a single discontinuity at the origin provides nonlinear characteristics.

![Figure II.9: The maximum function nonlinearity used for System 5.](image)

MATLAB simulations of the chaotic system are shown in Figure II.10 using the Euler approximation with a step size of \(2^{-3}\), indicating chaos by visual inspection due to the presence of a strange attactor. Nonlinear time series analysis using the software based on \[85\] gives the MLE as 0.089, thus confirming the presence of chaotic dynamics. The attracting equilibrium point at \((0.03125,0,0)\) is clear from Figure II.10(a).
and Figure II.10(c). In particular, the resulting chaotic attractor shows good correspondence with the analog attractor that has been shown in [56].

Figure II.10: MATLAB simulations of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of the maximum function nonlinearity chaotic system using 20,000 iterations.

II.1.6 System 6: 1D Multiscroll - Staircase Nonlinearity

This non-autonomous system is described by the following state-space matrix representation of three first order ordinary differential equations with a single parameter $\alpha \in [0.47, 1)$ [68]:

$$
\begin{bmatrix}
\dot{X}_1 \\
\dot{X}_2 \\
\dot{X}_3
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-\alpha & -\alpha & -\alpha
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
\alpha N_{L,U}(X_1)
\end{bmatrix}
$$

(II.21a)

$$
N_{L,U}(X_1) = \begin{cases}
U & X_1 \in (U, \infty) \\
L & X_1 \in (-\infty, L) \\
\lfloor X_1 \rfloor & X_1 \in [L, U], \text{ and } \lfloor X_1 \rfloor \text{ odd} \\
\lfloor X_1 \rfloor + 1 & X_1 \in [L, U], \text{ and } \lfloor X_1 \rfloor \text{ even}
\end{cases}
$$

(II.21b)

where $(L, U)$ are odd-valued lower and upper bounds and $\lfloor \circ \rfloor$ denotes the floor function (i.e. rounding down to the nearest integer). The upper and lower bound, along
with the system parameter \( \alpha \) can be controlled in real-time. Solving the above system for the steady-state solution where \( (\dot{X}_1, \dot{X}_2, \dot{X}_3) = (0, 0, 0) \) gives the equilibrium points as:

\[
\begin{bmatrix}
X_1^* \\
X_2^* \\
X_3^*
\end{bmatrix}
= \begin{bmatrix} m \\ 0 \\ 0 \end{bmatrix}, \quad m \in [L, U], m \in Z^+
\]  

(II.22)

These can be split into two sets of odd values \( m_o \) and even values \( m_e \). The Jacobian at these points is given by the linearization of the chaotic system:

\[
J_{(m_e,0,0)} = \lim_{k \to \infty} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ k & -\alpha & -\alpha \end{bmatrix}, \quad J_{(m_o,0,0)} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\alpha & -\alpha & -\alpha \end{bmatrix}
\]  

(II.23)

The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the stability of the proposed chaotic system. The eigenvalues are obtained by determining the roots of the characteristic equation of the Jacobian:

\[
s^3 + \alpha s^2 + \alpha s - k = 0, \quad s^3 + \alpha s^2 + \alpha s + \alpha = 0
\]  

(II.24a)

\[
s_{(m_e,0,0)} = \{\mu, -\nu \pm \sigma i\}, \quad \mu, \nu > 0
\]  

(II.24b)

\[
s_{(m_o,0,0)} = \{-0.739, 0.119 \pm 0.814i\}, \alpha = 0.5
\]  

(II.24c)

The results indicate that qualitatively, all the even valued equilibrium points \( m_e \) are \textit{spiral saddle points of index 1} due to one positive real eigenvalue and a pair of complex conjugate eigenvalues with negative real parts. Similarly, the odd-valued equilibrium points \( m_o \) are \textit{spiral saddle points of index 2} due to one negative real eigenvalue and a pair of complex conjugate eigenvalues with positive real parts [47]. The positive real parts in the complex eigenvalues at all \( m_o \) will enable oscillation around that
point, while the positive real eigenvalue will enable repulsion from all $m_e$. As a result, scrolls will form at every odd-valued equilibrium point $m_o$ and will be connected by the points at $m_e$. The nonlinear function $N_{L,U}(X_1)$ is shown in Figure II.11 wherein it is piecewise linear and discontinuities at saddle points of index 1 provide nonlinear characteristics. This is based on the signum nonlinearity used for System 2.

![Figure II.11: The staircase nonlinearity used for System 6.](image)

The total number of scrolls is given by the total number of break-points within $L$ and $U$ (inclusive):

$$N_{\text{scrolls}} = \frac{U - L}{2} + 1, \quad U > L$$ (II.25)

MATLAB simulations of the chaotic system are shown in Figure II.12 for 3 and 7 scroll systems using the Euler approximation with a step size of $2^{-3}$, indicating chaos by visual inspection due to the presence of a strange attractor. Nonlinear time series analysis using the software based on [85] gives the MLE as 0.137 for 3-scroll output, thus confirming the presence of chaotic dynamics. The attracting equilibrium points at $(m_o, 0, 0)$ are clear from Figure II.12(a),(d) and Figure II.12(c),(f).
II.1.7 System 7: 4D Hyperchaos - Signum Nonlinearity

This nonautonomous system is described by the following state-space matrix representation of four first order ordinary differential equations [73]:

\[
\begin{bmatrix}
\dot{X}_1 \\
\dot{X}_2 \\
\dot{X}_3 \\
\dot{X}_4 \\
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-4 & -4 & -4 & -4 \\
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
X_3 \\
X_4 \\
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
0 \\
-4D(t)\text{sgn}(X_1) \\
\end{bmatrix}
\]

where \(D(t)\) is a function that varies in time, is externally controlled and is bounded such that \(D(t) \in [0.25, 0.375]\). Solving the above system for the steady-state solution
where \((\dot{X}_1, \dot{X}_2, \dot{X}_3, \dot{X}_4) = (0, 0, 0, 0)\) gives the equilibrium points as:

\[
\begin{bmatrix}
X_1^* \\
X_2^* \\
X_3^* \\
X_4^*
\end{bmatrix}
= \begin{bmatrix}
0, \pm D(t) \\
0 \\
0 \\
0
\end{bmatrix}
\]  

(II.27)

The Jacobian at these points is given by the linearization of the chaotic system:

\[
J_{(0,0,0,0)} = \lim_{k \to \infty} \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
k & -4 & -4 & -4
\end{bmatrix}
J_{(\pm D(t),0,0,0)} = \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-4 & -4 & -4 & -4
\end{bmatrix}
\]  

(II.28)

Critically, both linearizations, and thus qualitative dynamics, are independent of the control parameter \(D(t)\). The eigenvalues of the Jacobian at the equilibrium points enable an assessment of the stability of the proposed chaotic system. The eigenvalues are obtained by determining the roots of the characteristic equation (in \(s\)) of the Jacobian \((J)\):

\[
s^4 + 4s^3 + 4s^2 + 4s - k = 0, \quad s^4 + 4s^3 + 4s^2 + 4s + 4 = 0
\]  

(II.29a)

\[
s_{(0,0,0,0)} = \{\mu_1, -\mu_2, -\nu \pm \sigma i\}, \quad \mu_1, \mu_2, \nu > 0
\]  

(II.29b)

\[
s_{(\pm D(t),0,0,0)} = \{-2.947, -1.225, 0.0861 \pm 1.049i\}
\]  

(II.29c)

The results indicate that qualitatively, \(s_{(0,0,0,0)}\) is a spiral saddle point of index 1 due to one positive real eigenvalue, one negative real eigenvalue and a pair of complex conjugate eigenvalues with negative real parts. Similarly, \(s_{(\pm D(t),0,0,0)}\) are spiral saddle points of index 2 due to two negative real eigenvalue and a pair of
complex conjugate eigenvalues with positive real parts \[47\]. The positive real parts in
the complex eigenvalues at \((\pm D(t), 0, 0, 0)\) will enable oscillation around that point,
while the postive real eigenvalue will enable repulsion from \((0, 0, 0, 0)\). Note that in
this case, the system is unstable in two dimensions due to the 4-D structure. The
nonlinear function \(\text{sgn}(X_1)\) is shown in Figure II.13 wherein it is piecewise linear
and a single discontinuity at the origin provides nonlinear characteristics. The same
nonlinearity was used for System 2.

![Figure II.13: The signum nonlinearity used for System 7.](image)

MATLAB simulations of the chaotic system are shown in Figure II.14 using the
Euler approximation with a step size of \(2^{-3}\) and a constant \(D(t) = 0.375\), indicating
chaos by visual inspection due to the presence of a strange attactor. Nonlinear time
series analysis using the software based on \[85\] gives the MLE as 0.177 and a second
positive Lyapunov exponent as 0.027, thus confirming the presence of hyperchaotic
dynamics \[73\]. In this particular case, the attracting equilibrium point is determined
by the initial condition with two coexistent attractors within the same system. The
attracting equilibrium points at \((\pm D(t), 0, 0, 0)\) are clear from Figure II.14(a) and
Figure II.14(c). Further sections will explore the exploitation of the non-autonomous
nature of this system for enhanced statistical properties.
II.2 Digital Implementation

All the non-linear ODEs presented in the previous section have been verified to have chaotic dynamics through the presence of strange attractors with phase-space boundedness, and one or two positive Lyapunov exponents for chaos (systems 1-6) and hyperchaos (system 7) respectively. A general approach for the implementation of \( n \)-th order chaotic ODEs is shown in Figure II.15, wherein the next state of each of the \( n \) state variables (stored in registers) is calculated using combinational logic. Fundamentally, the digital implementation of chaotic systems thus requires two key considerations: (i) the number representation scheme and (ii) the numerical technique employed to approximate the solution of the chaotic system of ODEs.

Numbers can be represented in either fixed-point or floating-point formats. In general, fixed-point implementations are more area and power-efficient, provide higher performance and are portable in several digital applications as they use standard
logic cells. Therefore, for the purposes of this thesis, only fixed-point representations are used for all digital implementations. As for the numerical technique, the 4th order Runge-Kutta, midpoint and Euler methods have all been considered for chaotic systems [92]. Out of these, the Euler method showed the best chaotic response, the lowest area and the highest performance. Therefore, only the Euler approximation is used for all of the systems under study in this work. Close inspection of Figure II.15 indicates that the routing complexity increases tremendously as the order of the system increases. Fortunately, all of the systems under study fall under the category of “jerk” (systems 1-6) or “hyperjerk” (system 7), wherein they can be explicitly written as third or fourth order ODEs. This suggests that feedback from all state variables is only required for the calculation of $X_n$, greatly reducing design complexity. Other known chaotic systems such as the well-known Lorenz [5], Rössler [54], Chen [55] and Chua [7] systems do not have this benefit. Furthermore, all scalar parameters are optimized to powers of two, such that scalar multiplications are eliminated in favour of arithmetic shifts in the binary domain, greatly minimizing hardware and optimizing performance.

This section will digitally implement each of the chaotic systems proposed in the previous section, highlighting key aspects of each design. In particular, the overall block diagram, sketch of the non-linear function and the resulting oscilloscope traces.
from experimental verification on a Xilinx Virtex 4 FPGA will be emphasized.

II.2.1 System 1: Modulus Nonlinearity

Using the Euler approximation with step size $h$, the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.1) and are given in matrix representation as:

$$
\begin{bmatrix}
X_{1}^{t+h} \\
X_{2}^{t+h} \\
X_{3}^{t+h}
\end{bmatrix} = 
\begin{bmatrix}
1 & h & 0 \\
0 & 1 & h \\
0 & -0.5h & 1-0.25h
\end{bmatrix}
\begin{bmatrix}
X_{1}^{t} \\
X_{2}^{t} \\
X_{3}^{t}
\end{bmatrix} + 
\begin{bmatrix}
0 \\
0 \\
0.25h |X_{1}^{t}| - 0.5h
\end{bmatrix}
$$

The resulting circuit schematic for this implementation (using $h = 2^{-5}$) is shown in Figure II.16 wherein the registers $\{X_1, X_2, X_3\}$ store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 32-bits is used with 4-bits representing the sign and integer part and the remaining 28-bits representing the fractional part. The term $0.5 - 0.25|X_1|$ is precomputed and stored in a temporary register to ease the combinational bottleneck at the input of the $X_3$ register. In the circuit, the nonlinear function $|X_1|$ is implemented using a simple adder/subtractor wherein the value of

![Figure II.16: Digital Implementation of System 1 (Modulus Nonlinearity)](image-url)
0.25X_1 is added to 0.5 if the most significant bit (MSB) is 1 and subtracted when the MSB is 0, giving the appropriate functionality. The attractors of the digitally implemented circuit are shown in Figure II.17 and are drawn as the oscilloscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental verification of chaotic behaviour and good correspondence with MATLAB simulations in Figure II.2.

Figure II.17: Oscilloscope traces of (a) X_1 − X_2, (b) X_1 − X_3 and (c) X_2 − X_3 attractors of System 1 (Modulus nonlinearity) from a Xilinx Virtex 4 FPGA.

### II.2.2 System 2: Signum Nonlinearity

Using the Euler approximation with step size h, the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.5) and are given in matrix representation as:

\[
\begin{bmatrix}
X_1^{t+h} \\
X_2^{t+h} \\
X_3^{t+h}
\end{bmatrix} =
\begin{bmatrix}
1 & h & 0 \\
0 & 1 & h \\
-h & -h & 1 - 0.5h
\end{bmatrix}
\begin{bmatrix}
X_1^t \\
X_2^t \\
X_3^t
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
2h \text{sgn}(X_1^t)
\end{bmatrix}
\] (II.31)

The resulting circuit schematic for this implementation (using h = 2^{−5}) is shown in Figure II.18 wherein the registers \{X_1, X_2, X_3\} store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 32-bits is used with 4-bits representing the
Figure II.18: Digital Implementation of System 2 (Signum Nonlinearity)

sign and integer part and the remaining 28-bits representing the fractional part. The term $X_1 - 2\text{sgn}(X_1)$ is precomputed and stored in a temporary register to ease the combinational bottleneck at the input of the $X_3$ register. In the circuit, the nonlinear function $\text{sgn}(X_1)$ is implemented using a simple adder/subtractor wherein 2 is added to the value of $X_1$ if the MSB of $X_1$ is 1 and subtracted when the MSB is 0, giving the appropriate functionality. The attractors of the digitally implemented circuit are shown in Figure [II.19] and are drawn as the oscilloscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental verification of chaotic behaviour and good correspondence with MATLAB simulations in Figure [II.4].

Figure II.19: Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 2 (Signum nonlinearity) from a Xilinx Virtex 4 FPGA.
II.2.3 System 3: Quadratic Nonlinearity

Using the Euler approximation with step size $h$, the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.9) and are given in matrix representation as:

\[
\begin{bmatrix}
X_1^{t+h} \\
X_2^{t+h} \\
X_3^{t+h}
\end{bmatrix} =
\begin{bmatrix}
1 & h & 0 \\
0 & 1 & h \\
0 & -2h & 1 - 0.5h
\end{bmatrix}
\begin{bmatrix}
X_1^t \\
X_2^t \\
X_3^t
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
-0.5h(X_1^t)^2 + 2h
\end{bmatrix}
\] (II.32)

The resulting circuit schematic for this implementation (using $h = 2^{-5}$) is shown in Figure II.20 wherein the registers \{X_1, X_2, X_3\} store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 32-bits is used with 4-bits representing the sign and integer part and the remaining 28-bits representing the fractional part. The term $2^{-6}(X_1)^2 - 2^{-4}$ is precomputed and stored in a temporary register to ease the combinational bottleneck at the input of the $X_3$ register. In the circuit, the nonlinear function $(X_1)^2$ is implemented using a $32 \times 32$ multiplier and truncating the bottom 32-bits. Since the integer width is 4-bits, this represents an effective right shift of 4-bits. A further right-shift of 2-bits fully incorporates the Euler step size, giving the appropriate functionality. The attractors of the digitally implemented circuit are

Figure II.20: Digital Implementation of System 3 (Quadratic Nonlinearity)
shown in Figure II.21 and are drawn as the oscilloscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental verification of chaotic behaviour and good correspondence with MATLAB simulations in Figure II.6.

Figure II.21: Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 3 (Quadratic nonlinearity) from a Xilinx Virtex 4 FPGA.

II.2.4 System 4: Cubic Nonlinearity

Using the Euler approximation with step size $h$, the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.13) and are given in matrix representation as:

\[
\begin{bmatrix}
X_1^{t+h} \\
X_2^{t+h} \\
X_3^{t+h}
\end{bmatrix} =
\begin{bmatrix}
1 & h & 0 \\
0 & 1 & h \\
0.5h & -h & 1 - 0.5h
\end{bmatrix}
\begin{bmatrix}
X_1^t \\
X_2^t \\
X_3^t
\end{bmatrix}
+ 
\begin{bmatrix}
0 \\
0 \\
-2^{-5}h(X_1^t)^3
\end{bmatrix}
\] (II.33)

The resulting circuit schematic for this implementation (using $h = 2^{-3}$) is shown in Figure II.22 wherein the registers $\{X_1, X_2, X_3\}$ store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 32-bits is used with 4-bits representing the sign and integer part and the remaining 28-bits representing the fractional part. In the circuit, the nonlinear function $(X_1)^3$ is implemented using two $32 \times 32$ multipliers.
and truncating the bottom 32-bits after each multiplication. Since the integer width is 4-bits, this represents an effective right shift of 8-bits and fully incorporates the Euler step size, giving the appropriate functionality. The attractors of the digitally implemented circuit are shown in Figure II.23 and are drawn as the oscilloscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental verification of chaotic behaviour and good correspondence with MATLAB simulations in Figure II.8.

Figure II.22: Digital Implementation of System 4 (Cubic Nonlinearity)

Figure II.23: Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$ and (c) $X_2 - X_3$ attractors of System 4 (Cubic nonlinearity) from a Xilinx Virtex 4 FPGA.
II.2.5 System 5: Maximum Function Nonlinearity

Using the Euler approximation with step size $h$, the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.17) and are given in matrix representation as:

\[
\begin{bmatrix}
X_{1}^{t+h} \\
X_{2}^{t+h} \\
X_{3}^{t+h}
\end{bmatrix} =
\begin{bmatrix}
1 & h & 0 \\
0 & 1 & h \\
0 & -h & 1 - 0.5h
\end{bmatrix}
\begin{bmatrix}
X_{1}^{t} \\
X_{2}^{t} \\
X_{3}^{t}
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
-8h\text{max}(X_{1}^{t}, 0) + 0.25h
\end{bmatrix}
\] (II.34)

The resulting circuit schematic for this implementation (using $h = 2^{-3}$) is shown in Figure II.24 wherein the registers \{X_1, X_2, X_3\} store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 32-bits is used with 4-bits representing the sign and integer part and the remaining 28-bits representing the fractional part. The term $\text{max}(X_1, 0) - 0.03125$ is precomputed and stored in a temporary register to ease the combinational bottleneck at the input of the $X_3$ register. In the circuit, the nonlinear function $\text{max}(X_1, 0)$ if calculated by performing a bitwise AND operation on every bit of $X$ with the inverse of the most significant bit. This in effect yields the output as zero whenever $X$ is negative and $X$ otherwise, thus giving the appro-

Figure II.24: Digital Implementation of System 5 (Maximum Function Nonlinearity)
appropriate functionality. The attractors of the digitally implemented circuit are shown in Figure II.25 and are drawn as the oscilloscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental verification of chaotic behaviour and good correspondence with MATLAB simulations in Figure II.10.

II.2.6 System 6: 1D Multiscroll - Staircase Nonlinearity

Using the Euler approximation with step size \( h \), the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.21a) and are given in matrix representation as:

\[
\begin{bmatrix}
X_{1}^{t+h} \\
X_{2}^{t+h} \\
X_{3}^{t+h}
\end{bmatrix} =
\begin{bmatrix}
1 & h & 0 \\
0 & 1 & h \\
-\alpha h & -\alpha h & 1 - \alpha h
\end{bmatrix}
\begin{bmatrix}
X_{1}^{t} \\
X_{2}^{t} \\
X_{3}^{t}
\end{bmatrix}
+ \begin{bmatrix}
0 \\
0 \\
-\alpha h N_{L,U}(X_{1}^{t})
\end{bmatrix} \quad (II.35)
\]

The resulting circuit schematic for this implementation (using \( h = 2^{-3} \)) is shown in Figure II.26(a) wherein the registers \( \{X_{1}, X_{2}, X_{3}\} \) store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 32-bits is used with 8-bits representing the
sign and integer part and the remaining 24-bits representing the fractional part. For this particular system, the system parameter $\alpha$ and the Euler step size $h$ are made controllable in real-time to assess the effect on chaotic response and to capture non-autonomous dynamics in the system. Of particular interest are the specific circuit implementations of the nonlinear staircase and the multipliers used for $\alpha$ and $h$.

Figure II.26(b) illustrates the hardware implementation of the staircase function described in (II.21a). Only the most significant 7-bits of the integer part are needed for computation. Eliminating the fractional bits effectively yields $\lfloor X_1 \rfloor$ which rounds $X_1$ towards $-\infty$. The output is one-padded by 1 bit and zero-padded by 24 bits on the least significant side to yield odd-valued integer results. This yields the relationship between the control parameters $(L_{in}, U_{in})$, the effective bounds $(L, U)$ and the number
of scrolls $N_{\text{scrolls}}$ as:

$$(L, U) = (2L_{in} + 1, 2U_{in} + 1) \quad (\text{II.36})$$

$$N_{\text{scrolls}} = U_{in} - L_{in} + 1, \quad U_{in} > L_{in} \quad (\text{II.37})$$

Both $L_{in}$ and $U_{in}$ are 7-bits wide (signed) and represent the break-points and centers of the scrolls at both ends. Due to limitations of overflow, the maximum possible number of scrolls is thus $2^7 - 2 = 126$ scrolls. Furthermore, since the upper bound must be greater than the lower bound and because the end-points are disallowed, the total legal combinations of $L_{in}$ and $U_{in}$ are given by:

$$N_{\text{comb}} = (2^7 - 3) + (2^7 - 4) + ... + 1 = 7875 \quad (\text{II.38})$$

While the other configurations are illegal in terms of chaos, they still provide an output that is usable for cryptosystems due to additional nonlinearities from arithmetic overflow.

To provide further controllability, the system parameter and Euler step size are implemented with multipliers that are realized according to Fig. II.26(c). Both the step size and the system parameter are designed as unsigned integers with 4-bit bus widths enabling $2^4 \times 2^4 = 256$ different combinations for the same $(L_{in}, U_{in})$. Since the most significant 32-bits are taken from the 36-bit output of the multiplier, the result is effectively already right-shifted by 4 bits and must be subsequently accounted for. Constants $p$ and $q$ specify arithmetic right-shifts to further scale the multiplier output and provide a scaled base value when the 4-bit input is zero, respectively. Since these are hard-coded shifts that only rewire buses, they do not require any additional hardware.

For the Euler multipliers for the step size $h$, constants $p$ and $q$ are chosen as 2 and
5 respectively such that the full Euler step size is:

\[ h = 2^{-5} + 2^{-6} h_{in} \quad h_{in} = 0, 1, 2, ..., 15. \]  \hspace{1cm} \text{(II.39)}

Since the valid range of chaotic parameters for this system \cite{68} is \( \alpha \in [0.47, 1] \), \( p \) is

Figure II.27: Oscilloscope traces of \( X_1 - X_2 \) attractors from the digitally implemented chaos generator with different controllable parameters to yield different (a)-(d) Number of scrolls with \( (L_{in}, h_{in}, a_{in}) = (0, 6, 15) \) and \( U_{in} = [2, 5, 10] \), (e)-(h) Position of scrolls with \( (h_{in}, a_{in}) = (6, 15) \), \( L_{in} = [-5, -3, 1] \) and \( U_{in} = L_{in} + 4 \), [5 scrolls], (i)-(l) System parameter with \( (L_{in}, U_{in}, h_{in}) = (-3, 2, 6) \) and \( a_{in} = [3, 8, 11] \) and (m)-(p) Euler step size with \( (L_{in}, U_{in}, a_{in}) = (-3, 2, 15) \) and \( h_{in} = [2, 7, 10] \) respectively. Results are drawn on an oscilloscope from the output of a Xilinx Virtex 4 FPGA.
and $q$ are both chosen as 1 such that the overall system parameter is:

$$\alpha = 2^{-1} + 2^{-5} \alpha_{in} \quad \alpha_{in} = 0, 1, 2, ..., 15.$$  \hspace{1cm} (II.40)

The attractors of the digitally implemented circuit are shown in Figure II.27 and are drawn as the oscilloscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental verification of chaotic behaviour and good correspondence with MATLAB simulations in Figure II.12. A large set of attractors are shown for variation of each parameter to illustrate the controllable aspect of the implemented circuit. Figure II.27 (a)-(d) clearly illustrate 3, 6 and 11 scrolls respectively only by changing the value of $U_{in}$ according to (II.37). Figure II.27 (d)-(f) show how the scrolls can be shifted in the $X_1$-direction (indicated by the constant position of the origin) by incrementing lower and upper bound by the same value to maintain a constant 5 scrolls as per (II.37). While visual inspection indicates identical attractors, the nonlinear function itself was modified to yield the different results. Therefore, all such cases have completely different trajectories and are treated as distinct attractors even though they have the same shape. Variations in the system parameter in Figure II.27 (g)-(i) or the Euler step size in Figure II.27 (j)-(l) maintain the number of scrolls but modify the shape due to differences in system characteristics and numerical precision respectively. Since the oscilloscope lacks the resolution to illustrate the full dynamic range of the implemented system, the output of the circuit from a Verilog simulation of 1,000,000 iterations for the full 126-scrolls is shown in Figure II.28. The dynamic range is limited by the number of integer bits and can be made arbitrarily large per required specifications, thus eliminating one of the major issues previously associated with multi-scroll chaos [71]. Overall, the relatively simple digital oscillator proposed in this paper can provide $7875 \times 256 = 2016000$ different legal modes of chaotic oscillation out of a possible $2^{22} = 4194304$ keys due to the restrictions on $L_{in}$.
Figure II.28: Verilog simulation of 1,000,000 iterations of the digital chaos generator with \( L_{in} = -63, U_{in} = 62, a_{in} = 15, h_{in} = 8 \). The result is 126 scrolls centered around zero.

and \( U_{in} \).

II.2.7 System 7: 4D Hyperchaos - Signum Nonlinearity

Using the Euler approximation with step size \( h \), the equations that numerically calculate the next state of each variable are derived from the functional form prescribed in (II.26) and are given in matrix representation as:

\[
\begin{pmatrix}
X_{t+h}^1 \\
X_{t+h}^2 \\
X_{t+h}^3 \\
X_{t+h}^4
\end{pmatrix} =
\begin{pmatrix}
1 & h & 0 & 0 \\
0 & 1 & h & 0 \\
0 & 0 & 1 & h \\
-4h & -4h & -4h & 1 - 4h
\end{pmatrix}
\begin{pmatrix}
X_t^1 \\
X_t^2 \\
X_t^3 \\
X_t^4
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
0 \\
-4hD^t\text{sgn}(X_t^1)
\end{pmatrix}
\] (II.41)

The resulting circuit schematic for this implementation (using \( h = 2^{-3} \)) is shown in Figure II.29 wherein the registers \( \{X_1, X_2, X_3, X_4\} \) store the state of the system while the computation is done through combinational logic. A fixed-point two’s complement representation with a total bus width of 16-bits is used with 1-bits representing the sign and integer part and the remaining 15-bits representing the fractional part, giving the constraint that \( \{X_1, X_2, X_3, X_4\} \in [-1, 1) \). The term \( 0.5(X_1 - D(t)\text{sgn}(X_1)) \) is precomputed and stored in a temporary register to ease the combinational bottleneck at the input of the \( X_4 \) register. In the circuit, the nonlinear function \( \text{sgn}(X_1) \) is
simply implemented by selecting either addition or subtraction of $X_1$ from $C(t)$ based on whether the MSB (sign) of $X_1$ is 1 (negative) or 0 (positive) respectively. Very conveniently, $X_4 - 4hX_4 = 0.5X_4$ is exploited to minimize computation. Similarly, all values are right shifted by 1-bit before the calculation of the next state of $X_4$ to avoid arithmetic overflow and simultaneously, to incorporate $4h = 0.5$. The forcing function $D(t)$ is defined as:

$$D(t) = \frac{1}{4} + \frac{2^{12} - 1}{2^{15}}D_{in}(t), \quad D_{in}(t) \in \{0, 1\} \quad (II.42)$$

where $D_{in}(t)$ is the effective binary-valued forcing function that is an input to the system. This guarantees that for the 16-bit implementation presented here, $D(t) \in [0.25, 0.375]$ such that $D(t) \neq 0 \ \forall \ t$ and is bounded. Through rigorous simulation, this also guarantees that $\{X_1, X_2, X_3, X_4\} \in [-1, 1)$ for all possible values of $D(t)$ over all time. In effect, the system has a 1-bit control parameter that drives the

$$0 \ 0 \ 1 \ 0 \ D_{in} \ D_{in} \ D_{in} \ \ldots \ D_{in} \ D_{in} \ D_{in}$$

Figure II.29: Digital Implementation of System 7 (4D Hyperchaos with Signum Non-linearity)

Figure II.30: Bitwise breakdown illustrating the relationship between $D(t)$ and $D_{in}(t)$. 
Figure II.31: Oscilloscope traces of FPGA output of the attractors from the 4D digital
chaos generator for (a) $X_1 - X_2$, (b) $X_1 - X_3$, (c) $X_1 - X_4$, (d) $X_2 - X_3$, (e) $X_2 - X_4$
and (f) $X_3 - X_4$. The traces are scaled to display the entire available phase space
$[-1, 1)$ and $D(t) = D \approx 0.375$.

12 least significant bits of $D(t)$, as illustrated in Figure II.30. The attractors of the
digitally implemented circuit are shown in Figure II.31 and are drawn as the oscil-
oscope traces of the output from a Xilinx Virtex 4 FPGA, indicating experimental
verification of chaotic behaviour and good correspondence with MATLAB simulations
in Figure II.14.

II.3 Area and Performance Results

All of the proposed systems are synthesized and experimentally verified on a Xilinx
Virtex 4 XC4VSX35-10FF668 FPGA that consists of a total of 30,720 4-input look-up
tables (LUTs) and 30,720 flip-flops (FFs). Additionally, the Virtex 4 FPGA family
also has digital signal processing (DSP) blocks that can be used to optimize multiplication. However, to facilitate a fair area comparison between all the systems under study, compiler and synthesizer directives restrict the use of these units. Table II.1 summarizes the utilization and performance of all the implemented systems. The gate count is conservatively estimated as \( G_c = 8 \times (LUTs + FFs) \) and the area efficiency is established as a figure of merit (FOM) given by:

\[
FOM = \frac{\text{Throughput}}{\text{Area}} = \frac{\text{Freq. (MHz) } \times \text{Bits}}{8 \times (LUTs + FFs)} \tag{II.43}
\]

All third order systems are implemented with a 32-bit bus width, resulting in 96 output bits by accounting for all three state variables. The fourth order hyperchaotic system is implemented with a 16-bit bus width, giving a total of 64 output bits by accounting for all four state variables. It is evident that the presence of multiplications in the quadratic, cubic and 1D multiscroll systems results in a significant area and frequency penalty, much like 1D chaotic maps that utilize multipliers. As a result, the benefits of pipelined architectures for multiplier-based chaotic systems are greatly diminished. The 1D multiscroll system has diminished performance but its fully controllable nature represents an interesting aspect that is worth exploring. Notably, System 5 (maximum function nonlinearity) shows the highest throughput mainly

<table>
<thead>
<tr>
<th>System</th>
<th>LUTs (units)</th>
<th>FFs (units)</th>
<th>Gc (units)</th>
<th>Bits (no.)</th>
<th>Freq. (MHz)</th>
<th>Th. (Mb/s)</th>
<th>FOM (Gc/Th)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus</td>
<td>317</td>
<td>128</td>
<td>3560</td>
<td>96</td>
<td>151.16</td>
<td>14511</td>
<td>4.08</td>
</tr>
<tr>
<td>Signum</td>
<td>314</td>
<td>128</td>
<td>3536</td>
<td>96</td>
<td>150.99</td>
<td>14495</td>
<td>4.10</td>
</tr>
<tr>
<td>Quadratic</td>
<td>912</td>
<td>128</td>
<td>8320</td>
<td>96</td>
<td>62.14</td>
<td>5965</td>
<td>0.72</td>
</tr>
<tr>
<td>Cubic</td>
<td>2058</td>
<td>128</td>
<td>17488</td>
<td>96</td>
<td>36.84</td>
<td>3537</td>
<td>0.20</td>
</tr>
<tr>
<td>Maximum</td>
<td>325</td>
<td>128</td>
<td>3624</td>
<td>96</td>
<td>184.50</td>
<td>17712</td>
<td>4.89</td>
</tr>
<tr>
<td>1-D Multiscroll</td>
<td>1267</td>
<td>103</td>
<td>10960</td>
<td>96</td>
<td>40.02</td>
<td>3842</td>
<td>0.35</td>
</tr>
<tr>
<td>Hyperchaos</td>
<td>128</td>
<td>80</td>
<td>1664</td>
<td>64</td>
<td>205.71</td>
<td>13165</td>
<td>7.91</td>
</tr>
</tbody>
</table>

Table II.1: Area and Performance Results on the Xilinx Virtex 4 FPGA for all seven implemented systems.
because its circuit does not have an adder/subtractor unit as is the case in System 1 and 2 (modulus and signum nonlinearities). System 7 (4D hyperchaos) shows the best figure of merit (area efficiency) mainly because it is implemented with only a 16-bit bus width but has the benefit of four outputs rather than three as is the case in all the other systems.

Previous literature [56, 60] has emphasized the simplicity of the circuit implementations of chaos in the analog domain. In the same vein, digital implementations use standard components such as adders, multipliers, multiplexors, flip-flops and basic logic gates, enabling easy porting to larger processing units. Additionally, digital design does not particularly require much knowledge of circuit components and eliminates tedious calculations associated with multiple feedback loops in analog circuits while also eliminating the need for proper matching of passive components. The focus in digital design is on the number representation system (fixed or floating point) and the numerical technique used to approximate the solutions to the ODE. If, as in this thesis, fixed-point representations and the Euler approximation are used, circuit design is greatly simplified and the resulting chaos generator facilitate the observation of complex non-linear dynamics with interesting mathematical properties through minimal effort on the part of the designer.
Chapter III

Chaotic Response: Analysis, Enhancement and Control

This chapter assesses the impact of variations in implementation and controllable parameters on the chaotic response of the previously implemented oscillators. While findings here are presented only for individual chaotic systems, the principles developed are easily transferable to other chaotic systems, given that all the systems under study fall under a similar sub-class of “jerk” and “hyperjerk” chaotic systems. The implementation bus width, internal delay cycles and external delay cycles are investigated for System 2 [87]. Subsequently, the effect of all the various controllable parameters in System 6 (1D Multiscroll with Staircase Nonlinearity) are studied in detail [88]. The effect of size variations are also assessed for the 4D Hyperchaotic system. Enhanced chaotic oscillators that exploit the long-term divergence in two identical systems implemented with different precision using arithmetic (ADD, SUB) and logic (AND, OR, XOR) operations are examined and give new attractors with higher Lyapunov exponents. Lastly, controllable oscillators that switch implementation parameters through multiplexing are demonstrated and switch reliably between chaotic and sinusoidal output.
### III.1 System 2 - Signum Nonlinearity

#### III.1.1 Implementation Bus Width

System 2 (with signum nonlinearity) is implemented with 10-bit, 12-bit, 16-bit and 32-bit bus widths (6, 8, 12 and 28 bits in the fractional part) with the resulting $X_1 - X_2$ attractors shown in Figure III.1. For a bus width of 10 bits in Figure III.1(a) the $X_1 - X_2$ attractor is in fact periodic rather than chaotic. However, the outline of the attractor resembles the general shape illustrated by the double scroll, indicating that the system, while being the same, does not produce chaotic output due to insufficient precision. The 12-bit $X_1 - X_2$ attractor in Figure III.1(b) shows a reliable chaotic output, while the 16-bit attractor in Figure III.1(c) and the 32-bit $X_1 - X_2$ attractor in Figure III.1(d) are relatively indistinguishable visually. The attractors notably show behavior similar to the Chua double-scroll attractor [7]. This verifies the idea that there exists a threshold precision beyond which the system behaves truly chaotic. Insufficient precision yields non-chaotic behavior. Therefore, the Euler approximated chaotic ODE can give either chaotic or periodic output based solely on the bus width and is in accordance with ideas proposed by Sprott in [47].

The maximum Lyapunov exponent (MLE) is calculated using 250,000 iterations for each case [85]. Figure III.2(a) shows the evolution of the MLE for systems with

![Figure III.2(a)](image)

Figure III.1: Oscilloscope output of digital $X_1 - X_2$ attractors of (a) 10-bit, (b) 12-bit, (c)16-bit and (d) 32-bit implementations of System 2.
different bus widths. A 10-bit bus width notably gives a negative MLE, indicating no chaotic behavior. The positive steady-state value of the exponent for 12-bit, 16-bit, 32-bit and 64-bit systems proves the chaotic behavior of the generated data over the long term. The MLE saturates at -0.0346, 0.137, 0.163, 0.151 and 0.144 for the 10-bit, 12-bit, 16-bit, 32-bit and 64-bit implementations respectively. Furthermore, the saturated MLE is calculated for systems with bus widths from 12-bits to 64-bits. Figure III.2(b) illustrates the saturated MLE normalized to the 12-bit
value of 0.137. Beyond the threshold minimum precision required for chaos (12-bits),
the MLE increases to a peak to 0.163 at the 16-bit implementation, approximately a
20% improvement. This increase corresponds to the need for more precision to fully
capture the chaotic behavior of the system. After this peak, the MLE decreases with
increasing precision, but remains greater than the baseline MLE of the 12-bit imple-
mentation. This reduction in MLE beyond the peak value corresponds to increased
precision that results in greater accuracy and lower truncation error. Truncation er-
ror corresponds to a non-linearity contributed by the numerical approximation and
larger truncations correspond to more non-linear behavior and hence a higher MLE.

III.1.2 Internal Delay Cycles

The original system 2, shown in (II.5) is modified to introduce a delay element
according to:

\[ \ddot{X}(t) = -0.5 \dot{X}(t - nh) - \dot{X}(t) - X(t) + 2 \text{sgn}(X(t)) \]  (III.1)

This modification is implemented in the original system using a series of \( n \) registers

Figure III.3: (a) Effect of number of internal delay cycles on MLE for a 16-bit imple-
mentation of System 2 and (b) Oscilloscope output of a 16-bit \( X_1 - X_2 \) attractor for
18 internal delay cycles.
to obtain $\tilde{X}$ delayed by $nh$. The clock period of the oscillator corresponds exactly to one individual time-step in the Euler approximation. The resulting systems designed with a 16-bit bus width experimentally exhibit the same attractors as those shown in Figure [III.1] by visual inspection. However, a plot of the normalized MLE (to zero-delay value of 0.163) for systems versus the number of internal delay cycles is shown in Figure [III.3(a)] which illustrates that the chaotic response of the delayed systems improves until a threshold delay of 14 cycles at a value of 0.219, a 34% increase in MLE compared to the baseline. Experimental results show that the system produces non-chaotic output beyond a 20 cycle delay while the chaotic response rapidly degrades after the optimal delay of 14 cycles. Figure [III.3(b)] illustrates the attractor found at a delay of 18 cycles, indicating that the system is at the upper threshold of chaos and increasing the delay further would diminish the chaotic response.

### III.1.3 External Delay Cycles

Using the original system, the $X_1$ output is delayed by $m$ clock cycles and is plotted versus $X_2$ in Figure [III.4]. The $m$-cycle delay is achieved by using $m$ registers in series. Therefore, the outputs of the system are now $\{X_1(t - mh), X_2(t), X_3(t)\}$. Since the original system has not been modified, the original chaotic properties are preserved. However, the shape of the attractor is modified, rotating in phase-space

![Oscilloscope output of $X - 1 - X_2$ attractors for a 16-bit implementation of System 2 with delay in $X_1$ of (a) 10, (b) 20, (c) 30, and (d) 40 cycles.](attachment:image.png)
per the amount of delay introduced. As such, the spread of values over 3-D phase space can be regulated. Delays of 10, 20, 30 and 40 clock cycles are shown.

III.2 System 6: 1D Multiscroll - Parameter Variation

The system is proved to be chaotic under variation of every parameter by calculating the MLE, using 250,000 iterations for each case [85]. This procedure also enables the chaotic response analysis of the chaos generator with respect to changes in each of the controllable parameters.

III.2.1 Number of Scrolls

Figure III.5 shows variation in MLE with the number of scrolls, normalized to a baseline MLE of 0.1366 for 3 scrolls. In general, MLE increases with increasing number of scrolls and stabilizes at approximately 30% above the baseline MLE. This can be attributed to the larger net basin of attraction afforded by larger numbers of scrolls. The MLE is calculated from 3 to 34 scrolls as further scrolls cannot be fully

Figure III.5: Change in MLE v/s number of scrolls: \((L_{in}, h_{in}, \alpha_{in}) = (0, 6, 15)\) and \(U_{in} = 2, 3, 4, ..., 33\) [3 – 34 scrolls].
captured in the 250,000 iteration limit indicated by the program based on [85].

### III.2.2 Position of Scrolls

Figure [III.6] illustrates the change in the MLE with the position of the scrolls, normalized to a baseline MLE of 0.165 for \((L_{in}, U_{in}) = (-7, 8)\), i.e. 16 scrolls. This parameter shows the smallest effect on the MLE, with the change never exceeding approximately 5% in absolute terms because the chaos remains similar with the same number of scrolls, Euler step size and system parameter and only the attractors are different due to their positions in phase space. This suggests that shifting positions does not really affect the chaotic dynamics of the system.

![Graph showing change in MLE with position of scrolls](image)

Figure III.6: Change in MLE v/s position of scrolls: \((h_{in}, \alpha_{in}) = (6, 15)\), \(L_{in} = -7, -6, -5, ..., 8\) and \(U_{in} = L_{in} + 7\) [8 scrolls].

### III.2.3 System Parameter

Figure [III.7] indicates the effect of variation in the system parameter on the MLE, normalized to a baseline MLE of 0.1875 for \(\alpha_{in} = 0 (\alpha = 2^{-1})\) and 4 scrolls. Here, the MLE is seen to decrease with increasing values of \(\alpha\) and stabilizes at a value approximately 15-20% lower than the baseline MLE and can be rationalized, in this case, in terms of the larger basin of attraction at each scroll for lower system parameter, as
evidenced in Figure II.27(g)-(i).

Figure III.7: Change in MLE v/s value of system parameter: \((L_{in}, U_{in}, h_{in}) = (0, 3, 6)\) and \(\alpha_{in} = 0, 1, 2, ..., 15\) [4 scrolls].

### III.2.4 Euler Step Size

Finally, Figure III.8 illustrates the percentage change in the MLE with the value of the Euler step size, normalized to a baseline MLE of 0.0628 for \(h_{in} = 0\) (\(h = 2^{-5}\)) and 4 scrolls. This particular parameter shows the greatest change in MLE with the values saturating at approximately 250% higher than the baseline MLE. This dependence arises because the Euler approximation introduces intrinsic nonlinearities associated.

Figure III.8: Change in MLE v/s Euler step size: \((L_{in}, U_{in}, \alpha_{in}) = (0, 3, 15)\) and \(h_{in} = 0, 1, 2, ..., 15\) [4 scrolls].
with numerical discretization. As the step size increases, these nonlinearities become stronger and contribute more heavily to the chaotic response of the oscillator.

III.3  System 7: 4D Hyperchaos - Variation of Attractor Size

Based on the mathematics established in Chapter 2, the input forcing function $D_{in}(t)$ controls the location of equilibrium points and thus simply controls only the size of the resulting attractor and not the qualitative dynamics of the circuit. As shown previously, the system parameter $D(t)$ is determined by the forcing function as in (II.42) as:

$$D(t) = \frac{1}{4} + \frac{2^{12} - 1}{2^{15}} D_{in}(t), \quad D_{in}(t) \in \{0, 1\}$$  \hspace{1cm} (III.2)

Figure III.9: Output histograms using 10,000,000 samples using $D(t) = 0.25$ (red) and $D(t) \approx 0.375$ (blue) for (a) $X_1$, (b) $X_2$, (c) $X_3$ and (d) $X_4$. 

- (a) $X_1$
- (b) $X_2$
- (c) $X_3$
- (d) $X_4$
The histograms of each of \{X_1, X_2, X_3, X_4\} for the proposed system with \(D(t) = 0.25\) and \(D(t) \approx 0.375\) are shown in Figure III.9 over 10,000,000 iterations and approximates the probability mass function. It is clear that each of \{X_1, X_2, X_3, X_4\} is symmetric about the origin and the larger value of \(D\) gives a larger spread across the available phase space.

### III.4 Enhanced Chaos Generators

Using Systems 1 to 4 (all with Euler step size \(h = 2^{-5}\)), the outputs of the same chaotic system, implemented with different bus widths, undergo arithmetic (addition, subtraction) or bitwise logical (AND, OR, XOR) operations to produce new attractors with an enhanced chaotic response that have never been reported before. These attractors are unique to digitally implemented chaos as they exploit designs of differing precisions and at the same time, employ bitwise logical transformations that have no correspondence in the analog domain. The four systems under study are implemented with 16-bit (12 fractional bits) and 32-bit (28 fractional bits) bus widths and new chaotic systems involving either subtraction, addition, bitwise AND, bitwise OR or bitwise XOR operations on the two output sets are presented according to the following expression, shown for \(X_1\) and illustrated in Figure III.10:

\[
X_{1,\text{new}} = X_{1,32} \circ X_{1,16} \tag{III.3}
\]

In (III.3), \(\circ\) denotes either addition, subtraction, bitwise AND, bitwise OR or bitwise XOR. 16-bit and 32-bit implementations use the same initial conditions \((X_1 = 0, X_2 = 0.5, X_3 = 0)\) to avoid encountering the known phenomenon of initial condition sensitivity. The initial conditions were appropriately chosen near the values indicated in [57] to ensure that the output is chaotic. These enhanced chaotic systems emerge from the digital implementation scheme and are unique to digital chaos.
III.4.1 New Attractors

The attractors of the digitally implemented chaotic systems are shown in Figure III.11. The transformed attractors are comparable in size to the original systems, illustrating that identical systems with different precision exhibit long-term divergence in spite of identical initial conditions. Subtractive processing yields non-trivial non-zero attractors that are nearly centered around zero and show similar shape for systems 1-4, suggesting that the subtraction output characteristics are dominated by precision limitations in the Euler approximation. Addition attractors maintain a shape roughly similar to the original attractors. Since the logical operations are bitwise, the output of the processed systems (AND, OR, XOR) is noticeably more random and spread over a larger region of phase space.

III.4.2 Maximum Lyapunov Exponents

The MLE for all four systems under study after all the output transformations are summarized in Table III.1. Output transformation yields significant chaotic response.
Figure III.11: Oscilloscope traces of $X_1 - X_2$ attractors of original 32-bit and addition, subtraction, bitwise AND, bitwise OR, and bitwise XOR of 32-bit and 16-bit implementations of (a)-(f) System 1, (g)-(l) System 2, (m)-(r) System 3 and (s)-(x) System 4 respectively.

Improvement of the original systems by a factor of 6-12x for addition and subtraction, 20-70x for bitwise AND, 29-92x for bitwise OR, and 33-98x for bitwise XOR. Enhanced chaotic systems yield a non-linear time series as the output that is treated as the solution to a chaotic ODE arising from the enhanced mode generated in the digital domain. The fact that the enhanced time series arises from the processing of two outputs of the same chaotic source with different precision forms the basis on which the enhanced modes are classified as chaotic as well. To further examine the effect of output processing, the bus widths are varied in case of subtraction and XOR.
systems, as shown in Figure III.12. Each 32-bit system is processed with the second bus width ranging between 16 and 31 bits for subtraction and XOR operations and the corresponding saturated MLE is plotted versus bus width. A general pattern of decreasing MLE is observed across all systems for both operations, although even the smallest possible precision variation of 1 bit between 32-bit and 31-bit implementations is sufficient to create long-term divergence in the output solution.

Table III.1: Maximum Lyapunov exponent for Chaotic Systems. All processed systems have a 32-bit bus width.

<table>
<thead>
<tr>
<th>Type</th>
<th>System 1</th>
<th>System 2</th>
<th>System 3</th>
<th>System 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original 16-bit</td>
<td>0.053</td>
<td>0.163</td>
<td>0.094</td>
<td>0.071</td>
</tr>
<tr>
<td>Original 32-bit</td>
<td>0.052</td>
<td>0.155</td>
<td>0.089</td>
<td>0.058</td>
</tr>
<tr>
<td>Addition</td>
<td>0.567</td>
<td>1.009</td>
<td>1.172</td>
<td>0.409</td>
</tr>
<tr>
<td>Subtraction</td>
<td>0.563</td>
<td>0.912</td>
<td>1.184</td>
<td>0.667</td>
</tr>
<tr>
<td>Bitwise AND</td>
<td>2.872</td>
<td>3.392</td>
<td>5.138</td>
<td>5.008</td>
</tr>
<tr>
<td>Bitwise OR</td>
<td>4.871</td>
<td>4.797</td>
<td>4.442</td>
<td>4.393</td>
</tr>
<tr>
<td>Bitwise XOR</td>
<td>5.205</td>
<td>5.375</td>
<td>5.799</td>
<td>5.538</td>
</tr>
</tbody>
</table>

Figure III.12: Variation of MLE with bus width in (a) SUB and (b) XOR processed implementations of (i) System 1, (ii) System 2, (iii) System 3, (iv) System 4. A system with bus width shown on the x-axis is operated with a 32-bit implementation with the same initial conditions \( X_1 = 0, X_2 = 0.5, X_3 = 0 \).
III.4.3 Time Series and Frequency Response

The time series and the fast Fourier transform (FFT) of the digitally implemented System 3 are shown in Figure III.13. The spectra and waveform of the original 32-bit implementation are similar to those of subtraction system in Figure III.13(d) because the characteristics of the original system are preserved across arithmetic operations. The bitwise XOR system is significantly different with faster transitions and less smooth output waveforms that illustrate larger high-frequency components due to the non-arithmetic processing. Frequency response broadening indicates that logical processing gives an intrinsically more random result than arithmetic processing.

Figure III.13: Oscilloscope output of time series and FFT of the (a)-(b) original 32-bit implementation, and (c)-(d) subtraction, and (e)-(f) bitwise XOR of the 32-bit and 16-bit implementations of System 3.
III.5  Periodic and Chaotic Controllability

To incorporate controllability, a selection input $M$ controls whether the output of the system is periodic ($M=1$) or chaotic ($M=0$). This is simply done by adjusting the system parameters of Systems 1 to 4 with multiplexors and would be difficult to implement in the analog domain. Systems 1 to 4 are re-stated in their original ODE forms with relevant parameters $A, B, C, D$ as follows:

$$\ddot{X} = -A\dot{X} - B\dot{X} + G(X)$$  \hspace{1cm} (III.4)

where $G(X)$ defines the nonlinear functions such that,

$$G(X) = \begin{cases} 
C|X| - D & \text{System 1} \\
-CX + D \cdot \text{sgn}(X) & \text{System 2} \\
-C\left(\frac{x^2}{D} - D\right) & \text{System 3} \\
-CX\left(\frac{x^2}{D} - 1\right) & \text{System 4} 
\end{cases}$$  \hspace{1cm} (III.5)

Table III.2 illustrates optimized parameters required for chaotic and periodic operation of each system. The chaotic parameters have already been presented in previous sections and only the periodic ones are introduced here. Being mindful of the fact that

<table>
<thead>
<tr>
<th>System</th>
<th>Mode</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Periodic</td>
<td>0.5</td>
<td>0.5</td>
<td>0.25</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>Chaotic</td>
<td>0.25</td>
<td>0.5</td>
<td>0.25</td>
<td>0.5</td>
</tr>
<tr>
<td>2</td>
<td>Periodic</td>
<td>0.25</td>
<td>1</td>
<td>1</td>
<td>-0.5</td>
</tr>
<tr>
<td>2</td>
<td>Chaotic</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Periodic</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Chaotic</td>
<td>0.5</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Periodic</td>
<td>0.25</td>
<td>1</td>
<td>0.25</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>Chaotic</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td>16</td>
</tr>
</tbody>
</table>
Figure III.14: Oscilloscope trace of the output time waveforms of the $X_1$ variable illustrating controllable operation of (a) system 1, (b) system 2, (c) system 3 and (d) system 4. The output waveform of $X_1$ is shown in yellow and the control parameter M is shown in blue. The results are experimental output from the FPGA.

digital oscillator output is periodic by definition, “periodic operation” in this context refers to the output being of a single-tone sinusoidal nature with a negative maximum Lyapunov exponent while the “chaotic operation” refers to the mode where the digital system outputs a chaotic attractor whose time evolution approximates that of the continuous-time chaotic system. Time waveforms shown in Figure III.14.
clearly illustrate this idea of periodic or chaotic behavior for high or low selection input, respectively. A finite amount of time is required transition between chaotic and stable periodic behavior and is dependent on the system state \( \{X_1, X_2, X_3\} \) at the transition of the selection input. Transitions from periodic to chaotic behavior are almost instantaneous. The systems can thus express either single-tone periodic or chaotic oscillation, allowing for a truly controllable scheme of chaos.
Chapter IV

Autonomous Chaos Generators as PRNGs

This chapter assesses the statistical properties of the chaotic output of the implemented systems by using the NIST SP. 800-22 statistical testing suite [89]. These tests rigorously inspect the statistical qualities of the output bitstreams from the chaos generators and assess the degree of predictability by comparing statistical properties to those expected from truly random bitstreams. In all cases, since chaos generators realize the numerical solution of nonlinear ODEs, all produce a clearly defined attractor in phase space, indicative of short-term predictability [38]. First, short-term predictability from a digital implementation perspective is assessed to examine the relationship between the Euler step size, implementation bus widths and attractor size and the number of statistically defective high-significance bits in System 1. Using this knowledge, PRNGs are created out of all the autonomous systems under study (Systems 1 to 5) by truncating these defective bits and only sending bitstreams with good statistical properties to the output [32], resulting in full passage of NIST SP. 800-22 statistical tests while, since the systems are autonomous, the initial conditions of the chaos generator specify the seed value of the PRNG.
IV.1 Analysis of Short-Term Predictability

Consider a very slightly modified version of System 1 as follows:

\[
\dot{Z} = \ddot{Y} = \dot{X} = J(X, Y, Z) \tag{IV.1a}
\]

\[
J(X, Y, Z) = -0.25Z - 0.5Y + 0.25|X| - D \tag{IV.1b}
\]

where \( D \) controls attractor size \([56]\) and \((X_1, X_2, X_3) = (X, Y, Z)\) is just simple renaming. The Euler approximation (with step size \( h = 2^{-k} \)) is applied just as in Chapter 2. In the digital domain, short-term predictability presents statistical defects in the high-significance bits of the output as illustrated in Figure [IV.1] Given that any of the chaotic systems under study is designed with an implementation bus width of \( N_B \)-bits, with \( N_I \) integer bits and \( N_F \) fraction bits in a two’s complement scheme, short-term predictability depends on \( N_I \), the Euler step size \( h \) and the attractor size (through \( D \)). If the chaotic output is to be implemented as a PRNG, defective bits need to be discarded \([32]\). Defects are assessed by testing the output bits through the NIST SP. 800-22 suite. The set of high-significance bits in each of \( \{X, Y, Z\} \) that fails the tests are judged defective.

Figure IV.1: Defective Bits due to short-term predictability in \( \{X, Y, Z\} \).
IV.1.1 Number of Integer Bits

Figure IV.2 illustrates the number of defective bits versus $N_B$ with each line denoting a different number $N_I$ with $D = 0.5$ and $h = 2^{-5}$. The effect of varying $N_F$ is simultaneously assessed. For a given $N_I$, the number of defective bits is independent of $N_F$ and for a given $N_B$, an increase in $N_I$ correspondingly decreases $N_F$, increasing in the number of defective bits. The resulting design principle suggests minimizing $N_I$ to maximize the number of fast-moving low-significance bits.

![Figure IV.2: No. of defective bits v/s $N_B$ with $D = 0.5$ and $h = 2^{-5}$.](image)

IV.1.2 Euler Step Size

The Euler step size $h$ has an upper bound such that the numerical solution accurately models the behavior of the ODE [88]. Furthermore, $h = 2^{-k}$ is required for hardware optimization. Figure IV.3 shows number of defective bits versus $N_I$ for different values of $h$ with $N_B = 64$ and $D = 0.5$. Analysis indicates that higher Euler step sizes correspond to fewer defective bits simply because there are more truncation nonlinearities and thus greater randomness. Furthermore, the effect of different $N_I$ and the Euler step size $h$ are independent as the effect from Figure IV.2 is also clearly observable in Figure IV.3 for a constant step size.
Figure IV.3: Number of defective bits versus bus width for different integer widths $N_I$ with $D = 0.5$ and $h$ from $2^{-5}$ to $2^{-9}$.

Figure IV.4: Same scale oscilloscope trace from a Xilinx FPGA of $X - Y$ attractors with (a) $D = 1.375$, (b) $D = 2.5$ and (c) $D = 3.375$.

IV.1.3 Size of the Attractor

The parameter $D$ controls attractor size and has a minimum value necessary for chaos \cite{56}. In this case, the lower bound is $D \approx 0.22$. The upper bound is limited by the size of the available phase space (specified by $N_I$). The effect of different $D$ is shown in Figure IV.4 (a)-(c) where the $X - Y$ attractor increases in size for increasing $D$. Figure IV.5 plots $\max (|X|, |Y|, |Z|)$ versus $D$ that results in a linear relationship:

$$\max (|X|, |Y|, |Z|) \approx 10.45D \quad \text{(IV.2)}$$
Figure IV.5: Maximum absolute value of \( \{X, Y, Z\} \) for \( D \) from 0.3 to 20.

Figure IV.6: Number of defective bits versus value of \( D \) for different \( N_I \) with \( N_B = 32 \) and \( h = 2^{-5} \). \( D \) is valued such that the resulting attractor is guaranteed to be bounded within the phase space defined by each \( N_I \).

Figure IV.6 shows the number of defective bits against the value of \( D \) for different \( N_I \) with \( N_B = 32 \) and \( h = 2^{-5} \), indicating that the best result is achieved with a value of \( D \) that maximizes the use of phase space, irrespective of the integer width. Therefore, this parameter should be optimized in conjunction with the integer width such that true optimality can be attained. Mathematically, this can be derived using the relationship from \( \text{(IV.3)} \). For a given integer width \( N_I \), the range of \( \{X, Y, Z\} \) is \([−2^{N_I−1}, 2^{N_I−1}]\) and the optimal \( D \) is thus:
\[ D_{opt} \approx \frac{2^{N_I - 1}}{10.45} \quad (IV.3) \]

Note that \( D \) should be rounded to the available precision of the fixed-point implementation and should always be kept lower than the optimal value to ensure the absence of arithmetic overflow.

**IV.1.4 Analysis Summary**

Based on the analysis, minimizing short-term predictability requires: (a) The highest possible Euler step size and (b) optimizing attractor size to fully utilize the phase space. This maximizes PRNG throughput of a given chaotic system by minimizing the number of defective bits. In general, the size of the attractor in phase space may be fixed for a particular chaotic system and cannot be manipulated. In such cases, the integer width should simply be minimized such that phase space utilization is maximized.

**IV.2 Systems 1 to 5 as PRNGs**

Given that the original systems presented in Chapter 2 have fixed parameters, the integer widths used previously are already optimal and the largest possible Euler step sizes are already being used. Table \[IV.1\] displays the NIST SP. 800-22 results of the native chaotic systems. The full suite consists of 15 different tests that examine different aspects of each bitstream, including frequency, runs, cumulative sums, FFT, entropy, serial, template matching, etc. Implementation parameters and area utilizations on the Xilinx Virtex 4 FPGA are also included, where \( Gc = 8 \times (LUTs + FFs) \) approximates a gate count for the design and the FOM is as defined in \[II.43\] and expresses the area efficiency. The native systems demonstrate overall statistical failure due to the presence of defective bits. Table \[IV.2\] displays the NIST SP. 800-22
Table IV.1: NIST SP. 800-22 test results for the original systems 1 to 5 along with experimental area and throughput results from the Xilinx Virtex 4 FPGA.

<table>
<thead>
<tr>
<th>NIST SP. 800-22 Results</th>
<th>Sys. 1 PV</th>
<th>PP</th>
<th>Sys. 2 PV</th>
<th>PP</th>
<th>Sys. 3 PV</th>
<th>PP</th>
<th>Sys. 4 PV</th>
<th>PP</th>
<th>Sys. 5 PV</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monobits</td>
<td>× 0.74</td>
<td>× 0.89</td>
<td>× 0.75</td>
<td>✓ 0.98</td>
<td>× 0.74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block Frequency</td>
<td>× 0.76</td>
<td>× 0.80</td>
<td>× 0.77</td>
<td>× 0.89</td>
<td>× 0.76</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cumulative Sums</td>
<td>× 0.74</td>
<td>× 0.87</td>
<td>× 0.76</td>
<td>✓ 0.96</td>
<td>× 0.74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Runs</td>
<td>× 0.63</td>
<td>× 0.65</td>
<td>× 0.65</td>
<td>× 0.66</td>
<td>× 0.64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Longest Run</td>
<td>× 0.45</td>
<td>× 0.51</td>
<td>× 0.56</td>
<td>× 0.68</td>
<td>× 0.69</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rank</td>
<td>× 0.72</td>
<td>× 0.78</td>
<td>× 0.82</td>
<td>× 0.82</td>
<td>× 0.84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFT</td>
<td>× 0.73</td>
<td>× 0.71</td>
<td>× 0.74</td>
<td>× 0.78</td>
<td>× 0.73</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N. O. Temp.</td>
<td>× 0.49</td>
<td>× 0.54</td>
<td>× 0.60</td>
<td>× 0.68</td>
<td>× 0.68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O. Temp.</td>
<td>× 0.44</td>
<td>× 0.50</td>
<td>× 0.53</td>
<td>× 0.67</td>
<td>× 0.69</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Universal</td>
<td>× 0.51</td>
<td>× 0.55</td>
<td>× 0.61</td>
<td>× 0.72</td>
<td>× 0.74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>App. Entropy</td>
<td>× 0.39</td>
<td>× 0.45</td>
<td>× 0.51</td>
<td>× 0.61</td>
<td>× 0.58</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R. Excur.</td>
<td>× 0.94</td>
<td>× 0.92</td>
<td>× 0.95</td>
<td>× 0.94</td>
<td>✓ 0.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R. Excur. Var.</td>
<td>✓ 0.99</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 0.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial</td>
<td>× 0.43</td>
<td>× 0.49</td>
<td>× 0.54</td>
<td>× 0.61</td>
<td>× 0.59</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L. Complexity.</td>
<td>× 0.83</td>
<td>× 0.93</td>
<td>✓ 0.96</td>
<td>✓ 0.99</td>
<td>✓ 1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall Result</td>
<td>Fail</td>
<td>Fail</td>
<td>Fail</td>
<td>Fail</td>
<td>Fail</td>
<td>Fail</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Experimental Results on the Xilinx Virtex 4 FPGA

<table>
<thead>
<tr>
<th>(N_B, N_I, − log₂ h)</th>
<th>(32, 4, 5)</th>
<th>(32, 4, 5)</th>
<th>(32, 4, 5)</th>
<th>(32, 4, 3)</th>
<th>(32, 4, 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUTs</td>
<td>317</td>
<td>314</td>
<td>912</td>
<td>2058</td>
<td>325</td>
</tr>
<tr>
<td>FFs</td>
<td>128</td>
<td>128</td>
<td>128</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>Gc</td>
<td>3560</td>
<td>3536</td>
<td>8320</td>
<td>17488</td>
<td>3624</td>
</tr>
<tr>
<td>Output Bits/Cycle</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
</tr>
<tr>
<td>Frequency [MHz]</td>
<td>151.16</td>
<td>150.99</td>
<td>62.14</td>
<td>36.84</td>
<td>184.50</td>
</tr>
<tr>
<td>Throughput [Mb/s]</td>
<td>14511</td>
<td>14495</td>
<td>5965</td>
<td>3537</td>
<td>17712</td>
</tr>
<tr>
<td>FOM [Th./Gc]</td>
<td>4.08</td>
<td>4.10</td>
<td>0.72</td>
<td>0.20</td>
<td>4.89</td>
</tr>
</tbody>
</table>

results after truncating defective bits. This leads to full passage of all tests and does not require any additional hardware because statistically defective output is simply being suppressed. However, there is significant throughput degradation as observed in the number of output bits per cycle. This has a corresponding effect on the figure of merit indicating that the effective area efficiency of the implemented PRNGs is significantly lower than the native chaotic output.
Table IV.2: NIST SP. 800-22 test results for chaotic systems 1 to 5 after truncation of defective bits along with experimental area and throughput results from the Xilinx Virtex 4 FPGA.

<table>
<thead>
<tr>
<th></th>
<th>Sys. 1</th>
<th></th>
<th>Sys. 2</th>
<th></th>
<th>Sys. 3</th>
<th></th>
<th>Sys. 4</th>
<th></th>
<th>Sys. 5</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
<td>PV PP</td>
</tr>
<tr>
<td>Monobits</td>
<td>✓ 1.00</td>
<td>✓ 0.98</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>Block Frequency</td>
<td>✓ 1.00</td>
<td>✓ 0.98</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>Cumulative Sums</td>
<td>✓ 1.00</td>
<td>✓ 0.98</td>
<td>✓ 0.99</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>Runs</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 0.96</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>Longest Run</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>Rank</td>
<td>✓ 0.97</td>
<td>✓ 0.96</td>
<td>✓ 0.98</td>
<td>✓ 1.00</td>
<td>✓ 0.98</td>
<td>✓ 0.98</td>
<td>✓ 0.98</td>
<td>✓ 0.98</td>
<td>✓ 0.98</td>
<td>✓ 0.98</td>
</tr>
<tr>
<td>FFT</td>
<td>✓ 1.00</td>
<td>✓ 0.93</td>
<td>✓ 0.96</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>N. O. Temp.</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
</tr>
<tr>
<td>O. Temp.</td>
<td>✓ 0.95</td>
<td>✓ 1.00</td>
<td>✓ 0.98</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>Universal</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>App. Entropy</td>
<td>✓ 1.00</td>
<td>✓ 0.93</td>
<td>✓ 0.98</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
<tr>
<td>R. Excur. Var.</td>
<td>✓ 0.98</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
<td>✓ 0.99</td>
</tr>
<tr>
<td>Serial</td>
<td>✓ 0.97</td>
<td>✓ 1.00</td>
<td>✓ 0.98</td>
<td>✓ 1.00</td>
<td>✓ 0.96</td>
<td>✓ 0.96</td>
<td>✓ 0.96</td>
<td>✓ 0.96</td>
<td>✓ 0.96</td>
<td>✓ 0.96</td>
</tr>
<tr>
<td>Lin. Compl.</td>
<td>✓ 0.97</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
<td>✓ 1.00</td>
</tr>
</tbody>
</table>

| Overall Result   | Pass   | Pass   | Pass   | Pass   | Pass   | Pass   |

<table>
<thead>
<tr>
<th>Experimental Results on the Xilinx Virtex 4 FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>((N_B, N_I, \log_2 h))</td>
</tr>
<tr>
<td>LUTs</td>
</tr>
<tr>
<td>FFs</td>
</tr>
<tr>
<td>Gc</td>
</tr>
<tr>
<td>Output Bits/Cycle</td>
</tr>
<tr>
<td>Frequency [MHz]</td>
</tr>
<tr>
<td>Throughput [Mb/s]</td>
</tr>
<tr>
<td>FOM [Th./Gc]</td>
</tr>
</tbody>
</table>

Furthermore, the multiplier-based architectures (Systems 3 and 4) are significantly larger and slower due to the complexity of hardware multiplication. It is clear that piecewise-linear nonlinearity-based architectures (1, 2 and 5) are far more area-efficient than multiplication nonlinearity-based chaotic systems (3 and 4). As a result, piecewise-linear functions enable full exploitation of digital implementations and result in smaller, analytically simpler and higher-throughput designs. All of the
implemented PRNGs use the initial conditions of the \( \{X_1, X_2, X_3\} \) state registers as seed values. The initial conditions can be completely arbitrary just so long as they are not precisely any of the equilibrium points of the implemented systems. This is because the equilibrium points are stationary states for the non-linear feedback pipeline and using those initial conditions will not give a chaotic attractor.

### IV.3 Comparison to Existing Literature

Table IV.3 compares the results of the five PRNGs presented in this chapter against previous chaos-based PRNGs demonstrated in the literature as well as a simple LFSR and an NLFSR. All of the previously implemented chaos-based work has been presented in the last 5 years (2007-2012). Out of the five proposed PRNGs, the piecewise-linear cases (Systems 1, 2 and 5) significantly surpass previous work in area efficiency terms and even the slower multiplier-based ones are comparable. It is however, overwhelmingly clear that the digital implementation of higher-order chaotic systems

<table>
<thead>
<tr>
<th>System</th>
<th>Area (Gc)</th>
<th>Th. FOM (Mb/s)</th>
<th>NIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFSR [33]</td>
<td>223</td>
<td>500</td>
<td>2.24 Fail</td>
</tr>
<tr>
<td>NLFSR [33]</td>
<td>2514</td>
<td>400</td>
<td>0.16 -</td>
</tr>
<tr>
<td>Addabbo, 2007 [28]</td>
<td>Rényi Map</td>
<td>3988</td>
<td>200 0.05 Pass</td>
</tr>
<tr>
<td>Chen, 2010 [29]</td>
<td>Logistic Map</td>
<td>9622</td>
<td>200 0.02 Pass</td>
</tr>
<tr>
<td>Li, 2010 [30]</td>
<td>Logistic Map</td>
<td>9136</td>
<td>200 0.02 Fail</td>
</tr>
<tr>
<td>Chen, 2010 [31]</td>
<td>Logistic Map</td>
<td>31655</td>
<td>3200 0.10 Pass</td>
</tr>
<tr>
<td>Zidan, 2011 [32]</td>
<td>ODE</td>
<td>2464</td>
<td>1180 0.48 Pass</td>
</tr>
<tr>
<td>Li, 2012 [33]</td>
<td>Logistic Map</td>
<td>11903</td>
<td>6400 0.54 Pass</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposed Autonomous PRNGs</th>
<th>Area (Gc)</th>
<th>Th. FOM (Mb/s)</th>
<th>NIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>System 1: Modulus</td>
<td>ODE</td>
<td>3560</td>
<td>5895 1.66 Pass</td>
</tr>
<tr>
<td>System 2: Signum</td>
<td>ODE</td>
<td>3536</td>
<td>6946 1.96 Pass</td>
</tr>
<tr>
<td>System 3: Quadratic</td>
<td>ODE</td>
<td>8320</td>
<td>3169 0.38 Pass</td>
</tr>
<tr>
<td>System 4: Cubic</td>
<td>ODE</td>
<td>17488</td>
<td>2100 0.12 Pass</td>
</tr>
<tr>
<td>System 5: Maximum</td>
<td>ODE</td>
<td>3624</td>
<td>10517 2.90 Pass</td>
</tr>
</tbody>
</table>
yields PRNGs of higher efficiency due to multidimensional output and eliminates the dependence on multiplications as with the logistic map typically used for chaos-based PRNGs.

**IV.3.1 Drawbacks of Autonomous Chaos-Based PRNGs**

While it has been shown that the autonomous chaos-based PRNGs presented in this thesis far surpass existing literature, they have some key issues that limit the application spaces where they can be used. Firstly, a large section of the chaotic output (at least 40%) has been shown to be statistically defective and are discarded to enable passage of the NIST SP. 800-22 tests. A post-processing technique that can preserve the entire chaotic throughput would be extremely desirable so long as it does not incur a significant hardware penalty.

Secondly, the key-space of the autonomous systems is fully determined by the bus width of the implementation. As computational power continues to increase, longer-period PRNGs will become necessary for cryptographic and communication applications. The pipelined structures proposed in this thesis imply that clock frequencies will decrease for higher implementation widths while area utilizations will increase. Indeed, the systems will produce more statistically random bits but the returns diminish as implementation bus widths scale higher up and eventually, area efficiency will suffer. Perhaps more significantly, the Euler approximation used for the digital implementation establishes a certain lower bound of sensitivity to initial conditions. If the change in initial conditions is less than the Euler step size, it is possible that the change will get completely suppressed due to truncation error. This presents significant challenges from a security perspective.

Lastly, from an analytical perspective, it is very difficult to estimate the period of these chaotic systems [90][91]. Brute-force calculations have been done previously [33] but those were for a maximum possible period of $2^{32}$. In this case, the period can
go as high as $2^{96}$, rendering brute-force approaches computationally unfeasible. As a result, it is hard to make rigorous claims as to the period of the proposed PRNGs.

All three of these concerns prompt a detailed consideration of non-autonomous chaos as a possible solution with the help of a novel post-processing technique. The next chapter will use System 7 to create an optimal PRNG with a controllable period and key-space that are both completely decoupled from the chaotic system itself.
Chapter V

Non-Autonomous Hyperchaos with Post-Processing as PRNG

This chapter implements a complete and robust PRNG consisting of a linear feedback shift-register, the 4D hyperchaotic generator (System 7) and a novel post-processing technique that incorporates rotation, feedback and XORing to provide strong differential sensitivity and suppress statistical defects in the output to enable full utilization of all chaotic output bits as PRNG. The top-level schematic of the proposed PRNG is shown in Figure V.1 where a 256-bit LFSR drives the 4D hyperchaotic system which is, in turn, post-processed to yield statistically random output. Since the physical implementation of System 7 has already been elaborated on in significant detail in Chapter 2 and parameter variation was assessed in Chapter 3, this chapter will focus

![Figure V.1: Top-level schematic of the proposed PRNG based on non-autonomous hyperchaos and Fibonacci series post-processing.](image-url)
on the forcing function (i.e. LFSR) and the novel post-processing technique.

V.1 LFSR as a Forcing Function

Given the non-autonomous form of System 7 with a 1-bit driving input $D_{in}$ as in Figure [I.30] a clear design choice must be made to select an appropriate forcing function that will supply that bitstream. Linear feedback shift registers have long been used as rudimentary PRNGs because they are very simple to design, can operate at very high frequencies and can be implemented with arbitrarily long period lengths. However, being linear, they are perfectly predictable once a certain number of iterations are observed. The forcing function provides two key components to the overall PRNG: (i) the key-space and (ii) the lower bound of the period.

Consider the implementation of an $N_P$-bit LFSR. This system has a period of $2^{N_P} - 1$ iterations and thus also has a choice of $2^{N_P} - 1$ initial conditions as seed values. Taking advantage of this aspect guarantees that the key-space is determined wholly by the LFSR and not by the chaos generator, significantly easing analysis. Now, considering the fact that the driving term in the chaos generator has a period of $2^{N_P} - 1$, it is guaranteed that the output of the chaos generator has at least that period. As a result, the computational issues associated with period estimation are also eliminated as the lower bound of the period is fully determined by the LFSR and not by the chaos generator. Henceforth, let the seed value of the LFSR be known as $K_{PRNG}$ which is 256-bits wide.

For the purposes of this work, a 256-bit LFSR is implemented with the following maximum-length feedback polynomial [93]:

$$x^{256} + x^{254} + x^{251} + x^{246} + 1 \quad (V.1)$$

indicating taps at the 256th, 254th, 251st and 246th flip-flops in the LFSR, as shown in
Figure V.2. This single bit is the $D_{in}$ input to the hyperchaotic system, enumerated in (II.26) and (II.42) with the corresponding schematics in Figure II.29 and Figure II.30.

V.2 Fibonacci Series Post-Processing

Clearly, the 4D hyperchaotic system itself has many interesting mathematical properties wherein two positive Lyapunov exponents suggest very sensitive system dynamics. However, the attractor shape reflects a degree of short-term predictability [38] and must be addressed using post-processing to preserve throughput rather than the truncation approach implemented in Chapter 4. In this section, a new post-processing technique is proposed that uses two rotation and XOR feedback loops: the first loop suppresses short-term predictability using a fixed rotation while the second enhances differential sensitivity using a variable rotation controlled by the Fibonacci series. From Fig. II.29 it is clear that for the 16-bit implementation, 64-bits are provided as the output from the system. The schematic of the proposed post-processing technique is shown in Fig. V.3 with the two feedback loops highlighted. The function $R_i$ indicates a rotation by $i$-bits to the right.
Figure V.3: Schematic of the proposed post processor with a constant rotation in Loop 1 and a variable rotation controlled by the Fibonacci series in Loop 2.

V.2.1 Feedback Loop 1: Static Rotation

Loop 1 implements a rotation and subsequent XOR in feedback. The rotation amount is fixed at 1-bit, axiomatically guaranteeing that the rotation amount and the total input width (64) are relative primes. If loop 2 is neglected, the resulting output after $C$ cycles at the $m$-th bit position is:

$$Y_m^C = \bigoplus_{j=0}^{C-1} X_{[m+j]\mod 64}$$

Critically, since 1 and 64 are relative primes, the subscript of $X$ will go through all available values from 0 to 63. This means that all 64 input bits to this feedback loop contribute to each of the 64 output bits equally from a statistical perspective. This suggests that the expectation value of each output bit is equal due to a long-term average and is given by:

$$E(Y_m) = E \left[ \bigoplus_{j=0}^{63} X_j \right] \quad \forall \ m \in [0, 63], m \in \mathbb{Z}^+$$
The resultant expression for the expectation value of each output bit can be readily shown as [94]:

$$E(Y_m) = \frac{1}{2} - \frac{1}{2} \prod_{j=0}^{63} (1 - 2E(X_j)) + F$$

(V.4)

wherein it is now a function of the expectation value of each individual input bit-stream. The large number of XORs guarantees that $E(Y_m) \approx 0.5$, approximately ideal. It cannot be guaranteed that each individual input bit is statistically independent of all others given that they would all arise from the same chaotic generator. This of course means that non-zero correlation between each input bitstream would cause some degradation in the expectation value of each output bit, all of which are bundled in the single parameter $F$. However, $F$ would be small and thus have a very small impact on the overall solution. The expression above is meant to provide an intuitive understanding of the feedback dynamics in Loop 1. By suppressing bias in the output bitstreams, this loop has ensured the elimination of short-term predictability while simultaneously enabling maximum utilization of phase space.

**V.2.2 Feedback Loop 2: Dynamic Rotation with Fibonacci Series**

Considering the non-autonomous nature of the hyperchaotic system, it is essential that a new post-processing technique be introduced to quickly propagate a small difference in the input forcing function directly to the output, thus enhancing differential sensitivity and thus, key sensitivity. $R_k$ is a function that performs a variable rotation by $k$-bits to the right. The rotation amount is specified by a Fibonacci series according to:

$$k_j = \begin{cases} 
  j & 0 \leq j \leq 2 \\
  k_{j-1} + k_{j-2} & 3 \leq j \leq M 
\end{cases}$$

(V.5a)

$$j = i \mod(M + 1)$$

(V.5b)
\[ M = \{ i : k_i < 64, \ k_{i+1} \geq 64, \ i \in \mathbb{Z}^+ \} \]  \hspace{1cm} (V.5c)

In this work, for a 16-bit implementation, \( M = 9 \) and \( j \) is an integer in \([0, 9]\). The necessary circuit is implemented using a 64-bit 10-to-1 multiplexor with hardwired rotations at each input. The selection input is driven by a simple modulo 10 counter. The resulting schematic is shown in Fig. V.4. The presence of this loop guarantees that a single change at any bit position at any time will propagate to all the other bits within \( M \) cycles, giving equal sensitivity at each output bit position, rendering strong differential performance for a small deviation in keys.

![Schematic of the variable rotation using the Fibonacci series. A 64-bit barrel shifter is used in conjunction with a state machine that cycles through the appropriate \( k \)-values from (V.5).](image)

**V.3 Experimental Results**

Figure V.5 shows the attractors for the full system using \( K_{PRNG} = 1 \). Clearly, the attractor shape is slightly modified due to the pseudo random forcing function. After post-processing, the attractor shape and thus short-term predictability is completely suppressed, enabling full utilization of phase space. Furthermore, Figure V.6 shows the time series and the frequency response of \( X_1 \) (native chaotic output) and \( Y_1 \) (post-processed output). Post-processing guarantees a noticeably more random time series and a favorable white noise-like spectrum. These experimental results indicate that the hardware implementation is functioning as expected.

Figure V.7 shows the histograms of \( \{X_1, X_2, X_3, X_4\} \) and \( \{Y_1, Y_2, Y_3, Y_4\} \) over
Figure V.5: Oscilloscope traces of (a) $X_1 - X_2$, (b) $X_1 - X_3$, (c) $X_1 - X_4$, (d) $X_2 - X_3$, (e) $X_2 - X_4$, (f) $X_3 - X_4$, (g) $Y_1 - Y_2$, (h) $Y_1 - Y_3$, (i) $Y_1 - Y_4$, (j) $Y_2 - Y_3$, (k) $Y_2 - Y_4$, (l) $Y_3 - Y_4$ attractors of the (a)-(f) original hyperchaotic system with LFSR forcing and (g)-(l) Fibonacci post-processed output respectively.
Figure V.6: Oscilloscope traces of the time series and frequency response of (a)-(b) $X_1$ [Original hyperchaotic system with LFSR forcing] and (c)-(d) $Y_1$ [After Fibonacci Post-Processing].

Figure V.7: Output histograms using 10,000,000 samples for the LFSR-driven hyperchaotic system without post-processing (red) and after post-processing (blue) for (a) $X_1$ and $Y_1$, (b) $X_2$ and $Y_2$, (c) $X_3$ and $Y_3$, (d) $X_4$ and $Y_4$. 
Figure V.8: Cumulative percentage difference in output bitstreams over time for two different executions of the system using different keys ($K_{PRNG} = 2$ and $K_{PRNG} = 3$) (a) original hyperchaotic system and (b) after Fibonacci Post-Processing. The ideal value over the long-term is 50%.

10,000,000 iterations, approximating the probability mass function. It is clear that after post-processing, the statistical bias in the output is completely suppressed to give a uniform distribution. To illustrate the differential sensitivity of the oscillator, Figure V.8 shows, on a bitwise basis, the long-term cumulative percentage difference between two executions using $K_{PRNG} = 2$ and $K_{PRNG} = 3$ over time. The application of Fibonacci series post-processing ensures that full divergence is at all output bit positions (the ideal value of 50% is attained). Without post-processing, some high-significance bits do not fully incorporate small changes because the native system numerically solves an ODE and those changes do not propagate effectively to those bits.

Of particular interest is the information entropy associated with each output bit,
Figure V.9: Entropy of each output bitstream for (a) the original hyperchaotic system with LFSR forcing and (b) after applying the proposed Fibonacci post-processing. $K_{PRNG} = 1$ is used.

A firm indicator of long-term unpredictability of bitstreams. For the implemented system, entropy is assessed for the entire 64-bit output for a 1,000,000 bit sample, using the mathematical formulation described in [89] with an order of 10. In base 2, the maximum entropy per bit is 1, for a fair coin-toss. The results are shown in Figure V.9(a) wherein clearly the high-significance bits of the native chaotic system have very low entropy. In particular, the failures in the NIST SP. 800-22 tests are highlighted, indicating that a very high confidence in good entropy is needed for passage of the tests. Figure V.9(b) shows the same entropy graph after applying Fibonacci post-processing. Entropy enhancement in high-significance bits is evident such that all output bits are now within 0.041% of the maximum value of 1 and each bitstream passes NIST tests, enabling full utilization of all output bits.

To summarize, the NIST statistical test results are shown in Table V.2 for the hy-
Table V.1: NIST SP. 800-22 test results for the original hyperchaotic system (with 256-bit LFSR) and the corresponding Fibonacci post-processed output along with experimental area and throughput results from the Xilinx Virtex 4 FPGA.

<table>
<thead>
<tr>
<th>NIST SP. 800-22 Results</th>
<th>Original System</th>
<th>Post-Processed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PV</td>
<td>PP</td>
</tr>
<tr>
<td></td>
<td>PP</td>
<td>PP</td>
</tr>
<tr>
<td>Monobits</td>
<td>✓</td>
<td>0.95</td>
</tr>
<tr>
<td>Block Frequency</td>
<td>×</td>
<td>0.81</td>
</tr>
<tr>
<td>Cumulative Sums</td>
<td>✓</td>
<td>0.95</td>
</tr>
<tr>
<td>Runs</td>
<td>×</td>
<td>0.61</td>
</tr>
<tr>
<td>Longest Run</td>
<td>×</td>
<td>0.59</td>
</tr>
<tr>
<td>Rank</td>
<td>×</td>
<td>0.78</td>
</tr>
<tr>
<td>Fast Fourier Transform</td>
<td>×</td>
<td>0.67</td>
</tr>
<tr>
<td>Non-Overlapping Template Matching</td>
<td>×</td>
<td>0.57</td>
</tr>
<tr>
<td>Overlapping Template Matching</td>
<td>×</td>
<td>0.50</td>
</tr>
<tr>
<td>Universal</td>
<td>×</td>
<td>0.59</td>
</tr>
<tr>
<td>Approximate Entropy</td>
<td>×</td>
<td>0.42</td>
</tr>
<tr>
<td>Random Excursions</td>
<td>✓</td>
<td>0.94</td>
</tr>
<tr>
<td>Random Excursions Variant</td>
<td>✓</td>
<td>1.00</td>
</tr>
<tr>
<td>Serial</td>
<td>×</td>
<td>0.55</td>
</tr>
<tr>
<td>Linear Complexity</td>
<td>✓</td>
<td>0.98</td>
</tr>
<tr>
<td>Overall Result</td>
<td>Fail</td>
<td>Pass</td>
</tr>
</tbody>
</table>

Experimental Results on the Xilinx Virtex 4 FPGA

<table>
<thead>
<tr>
<th></th>
<th>Total LUTs</th>
<th>Total FFs</th>
<th>Gate Count (Gc)</th>
<th>Output Bits/Cycle</th>
<th>Frequency [MHz]</th>
<th>Throughput [Mb/s]</th>
<th>FOM [Th./Gc]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>129</td>
<td>336</td>
<td>3720</td>
<td>64</td>
<td>205.71</td>
<td>13165</td>
<td>3.54</td>
</tr>
<tr>
<td></td>
<td>713</td>
<td>468</td>
<td>9448</td>
<td>64</td>
<td>205.71</td>
<td>13165</td>
<td>1.39</td>
</tr>
</tbody>
</table>

perchaotic output \(\{X_1, X_2, X_3, X_4\}\) and the final post-processed output \(\{Y_1, Y_2, Y_3, Y_4\}\) as illustrated in Figure V.1. The native chaotic system suffers from short-term predictability and thus fails the tests. Output post-processing guarantees that the full output bus width passes the NIST tests and provides differential sensitivity for better statistical differentiation between output streams for different seed values as seen in Figure V.8.
V.4 Comparison to Previous Work

The area utilization and performance specifications are also shown in Table V.2. While the LFSR and the proposed post-processing have added a significant hardware cost, the end-result is a secure PRNG with a well-known lower bound for the period, sensitive keys and high throughput. This full system is compared against the systems proposed in Chapter 4 and previous literature just as before, but this time also illustrating periodicity and key-space of each system.

This final PRNG is compared against the autonomous systems implemented in Chapter 4. Clearly, the 256-bit LFSR and the output post-processing introduce a huge hardware overhead, increasing the total area utilization by approximately a factor of 5. The LFSR accounts for 256 FFs and 2 LUTs ($\approx 21.8\%$ of $Gc$), the hyperchaotic generator is 128 LUTs and 80 FFs ($\approx 17.6\%$ of $Gc$), feedback loop 1 and the register in loop 2 of the post-processor totally occupy 64 LUTs and 128 FFs ($\approx 16.3\%$ of $Gc$) and the Fibonacci series variable rotation circuit occupies the remaining 519 LUTs and 4 FFs ($\approx 44.0\%$ of $Gc$). It should be noted that this is

<table>
<thead>
<tr>
<th>System</th>
<th>Area (Gc)</th>
<th>Th. (Mb/s)</th>
<th>FOM</th>
<th>Period</th>
<th>Key</th>
<th>NIST SP. 800-22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addabbo [28]</td>
<td>3988</td>
<td>200</td>
<td>0.05</td>
<td>$P = 2^{31}$</td>
<td>$K = 2^{32}$</td>
<td>✓</td>
</tr>
<tr>
<td>Li [30]</td>
<td>9136</td>
<td>200</td>
<td>0.02</td>
<td>$P = 2^{21}$</td>
<td>$K = 2^{32}$</td>
<td>×</td>
</tr>
<tr>
<td>Chen [31]</td>
<td>31655</td>
<td>3200</td>
<td>0.10</td>
<td>$P = 2^{128}$</td>
<td>$K = 2^{128}$</td>
<td>✓</td>
</tr>
<tr>
<td>Zidan [32]</td>
<td>2464</td>
<td>1180</td>
<td>0.48</td>
<td>$P \in [2^{21}, 2^{48})$</td>
<td>$K &lt; 2^{48}$</td>
<td>✓</td>
</tr>
<tr>
<td>Li [33]</td>
<td>11903</td>
<td>6400</td>
<td>0.54</td>
<td>$P \in [2^{53}, 2^{65})$</td>
<td>$K = 2^{280}$</td>
<td>✓</td>
</tr>
<tr>
<td>Proposed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>3560</td>
<td>5895</td>
<td>1.66</td>
<td>$P \in [2^{32}, 2^{96})$</td>
<td>$K &lt; 2^{96}$</td>
<td>✓</td>
</tr>
<tr>
<td>S2</td>
<td>3536</td>
<td>6946</td>
<td>1.96</td>
<td>$P \in [2^{32}, 2^{96})$</td>
<td>$K &lt; 2^{96}$</td>
<td>✓</td>
</tr>
<tr>
<td>S3</td>
<td>8320</td>
<td>3169</td>
<td>0.38</td>
<td>$P \in [2^{32}, 2^{96})$</td>
<td>$K &lt; 2^{96}$</td>
<td>✓</td>
</tr>
<tr>
<td>S4</td>
<td>17488</td>
<td>2100</td>
<td>0.12</td>
<td>$P \in [2^{32}, 2^{96})$</td>
<td>$K &lt; 2^{96}$</td>
<td>✓</td>
</tr>
<tr>
<td>S5</td>
<td>3624</td>
<td>10517</td>
<td>2.90</td>
<td>$P \in [2^{32}, 2^{96})$</td>
<td>$K &lt; 2^{96}$</td>
<td>✓</td>
</tr>
<tr>
<td>S7</td>
<td>9448</td>
<td>13165</td>
<td>1.39</td>
<td>$P \in [2^{256}, 2^{320})$</td>
<td>$K = 2^{256}$</td>
<td>✓</td>
</tr>
</tbody>
</table>
not an accurate reflection of the actual size of the implemented circuit given the
limitations of mapping constraints on FPGAs, wherein the slices on this FPGA are
not optimized for large width multiplexing, resulting in excessive hardware usage. In
spite of this, the full system 7 is still superior to the best previous chaos-based PRNG
and also has a comparable lower-bound on the period and a comparable key-space.
Essentially, this analysis indicates that the implemented system with Fibonacci post-
processing is indeed efficient and can potentially be greatly optimized if a transistor-
level implementation is completed to facilitate a fair comparison to [33]. Still, for
occupying a lower area, the hyperchaos-based PRNG produces more than double
the throughput of the reseeding-mixing PRNG in [33] with better periodicity and a
comparable key space. Note that the key space of the hyperchaotic system can be
greatly increased if the initial conditions of the state variables \( \{X_1, X_2, X_3, X_4\} \) are
also considered, and would take the available key space for the proposed PRNG up to
\( 2^{320} \) without considering losses due to Euler truncation. For the autonomous systems,
the key space is guaranteed to be lower than the total of \( \{X_1, X_2, X_3\} \) widths due to
diminished sensitivity caused by truncation in the Euler approximation. The period
is verified with brute force up to \( 2^{32} \) but no guarantees can be made beyond that
other than the known upper bound that is equal to the maximum length sequence
for the total of \( \{X_1, X_2, X_3\} \) bus widths (i.e. \( 2^{96} \)).
Chapter VI

Conclusion

VI.1 Summary

This thesis has introduced a generalized framework for the fully digital hardware implementation of chaotic ordinary differential equations. Specifically, the implementations presented use fixed-point number representations with the Euler approximation to numerically solve the chaotic systems. Seven different systems (Jerk: (i) Modulus, (ii) Signum,, (iii) Quadratic, (iv) Cubic, (v) Maximum Function, (vi) 1-D multiscroll and (vii) Hyperchaos with Signum) were successfully assessed for nonlinear dynamics through stability analysis and simulations, with subsequent digital implementation and experimental verification on a Xilinx Virtex 4 FPGA. Subsequently, the effect of implementation parameters and controllable parameters were assessed, including the implementation bus width, delay cycles, Euler step size, system parameters and the number ad position of scrolls in the case of 1-D multiscroll chaos. The effect of changing attractor size was observed for non-autonomous hyperchaos. The autonomous systems (1-5) were implemented as PRNGs by truncating the defective high-significance bits and concatenating the remainder. These PRNGs pass all the NIST SP-800-22 tests and surpass previous implementations of chaos-based PRNGs by a significant
margin. However, to improve statistical properties and to provide larger periodicity and key-space, the 4-D hyperchaotic system is implemented as a PRNG with a 256-bit LFSR as a forcing function and novel Fibonacci series post-processing that preserves the full chaotic throughput while providing differential sensitivity for small changes in the input driving function. This system is shown to have the best periodicity and key-space with a compromise in the area-efficiency. The resulting PRNGs can be readily applied to encryption and digital communication systems for secure applications.

VI.2 Publications from this Work

VI.2.1 Published/Accepted


VI.2.2 Under Review


VI.2.3 In Preparation


VI.3 Future Research Work

Particular to Chapter 5, extensive research can be conducted to determine an appropriate forcing function for the PRNG apart from an LFSR. Very long period PRNGs such as the Mersenne Twister could also be used. This could give cryptographically secure highly long-period PRNGs that can be easily implemented in hardware and since the system is fully digital, can be translated into pseudocode that can be applied to any programming language for a software implementation. Future work can include a full transistor-level implementation of one of these chaos-based PRNGs to enable a fair comparison to the other previous implementations that have relied on transistor-level design rather than register-transfer level, as in this thesis. True area efficiency can only be attained if the system is optimized at the transistor-level and post-layout and post-silicon verification is obtained.

Shifting focus to application spaces, the chaos-based pseudo random number generators presented in this thesis can be readily applied to encryption systems. Since the implemented systems offer high throughput and complex nonlinear dynamics, there is potential for strong image and video encryption through both block and stream ciphers. Furthermore, emphasis on fully digital spreading codes for digital communication systems can also be further explored, given the high security arising from nonlinear encoding, so long as synchronization can be achieved by some means.

From an analytical perspective, given the digital domain established for implementation of these systems, mathematical research can look into different piecewise linear functions to generate different types of multi-scroll chaos apart from 1-D, including 2-D and 3-D grids and other more complex designs. Previous circuit implementations of different types of multiscroll chaos have been limited in that it is difficult to create complex nonlinearities using analog components. Keeping a digital implementation in mind would enable new avenues for the formulation of nonlinearities that can describe new, controllable chaotic attractors.
APPENDICES
Chapter VII

Calculating the Maximum Lyapunov Exponent

In [47], Sprott outlines a general procedure to estimate the maximum yapunov exponent from a nonlinear time series. This process is outlined as follows:

VII.1 Start with any initial condition in the basin of attraction

Even better would be to start with a point known to be on the attractor, in which case step 2 can be omitted.

VII.2 Iterate until the orbit is on the attractor

This requires some judgement or prior knowledge of the system under study. For most systems, it is safe just to iterate a few hundred times and assume that is sufficient. It usually will be, and in any case, the error incurred by being slightly off the attractor is usually not large unless you happen to be very close to a bifurcation point.
VII.3 Select (almost any) nearby point (separated by \(d_0\))

An appropriate choice of \(d_0\) is one that is about 1000 times larger than the precision of the floating point numbers that are being used. For example, in (8-byte) double-precision (minimum recommended for such calculations), variables have a 52-bit mantissa, and the precision is thus \(2^{-52} = 2.22 \times 10^{-16}\). Therefore a value of \(d_0 = 10^{-12}\) will usually suffice.

VII.4 Advance both orbits one iteration and calculate the new separation \(d_1\)

The separation is calculated from the sum of the squares of the differences in each variable. So for a 2-dimensional system with variables \(x\) and \(y\), the separation would be \(d = [(x_a - x_b)^2 + (y_a - y_b)^2]^{\frac{1}{2}}\), where the subscripts \((a\) and \(b)\) denote the two orbits respectively.

VII.5 Evaluate \(\log \left| \frac{d_1}{d_0} \right|\) in any convenient base

By convention, the natural logarithm (base-e) is usually used, but for maps, the Lyapunov exponent is often quoted in bits per iteration, in which case you would need to use base-2. (Note that \(\log_2 x = 1.4427 \log_e x\)). You may get run-time errors when evaluating the logarithm if \(d_1\) becomes so small as to be indistinguishable from zero. In such a case, try using a larger value of \(d_0\). If this doesn’t suffice, you may have to ignore values where this happens, but in doing so, your calculation of the Lyapunov exponent will be somewhat in error.
VII.6  Readjust one orbit so its separation is $d_0$ in the same direction as $d_1$

This is probably the most difficult and error-prone step. As an example (in 2-dimensions), suppose orbit $b$ is the one to be adjusted and its value after one iteration is $(x_{b1}, y_{b1})$. It would then be reinitialized to $x_{b0} = x_{a1} + \frac{d_0(x_{a1} - x_{a1})}{d_1}$ and $y_{b0} = y_{a1} + \frac{d_0(y_{a1} - y_{a1})}{d_1}$.

VII.7  Repeat steps 4-6 many times and calculate the average of step 5

You might wish to discard the first few values you obtain to be sure the orbits have oriented themselves along the direction of maximum expansion. It is also a good idea to calculate a running average as an indication of whether the values have settled down to a unique number and to get an indication of the reliability of the calculation. Sometimes, the result converges rather slowly, but a few thousand iterates usually suffices to obtain an estimate accurate to about two significant digits. It is a good idea to verify that your result is independent of initial conditions, the value of $d_0$, and the number of iterations included in the average. You may also want to test for unbounded orbits, since you will probably get numerical errors and the Lyapunov exponent will not be meaningful in such a case.

This procedure is generalized in [86] to obtain Lyapunov exponents from any time series and was subsequently written as an executable computer program based on the work done in [85]. This resulting software is used throughout this thesis for the calculation of the MLE.
Chapter VIII

NIST SP. 800-22 Test Suite

This comprehensive testing suite, provided by the National Institute of Standards and Technology (NIST), rigorously examines the statistical properties of sets of bitstreams to assess whether those bitstreams fall within the bounds of statistical randomness. It is composed of a battery of 15 statistical tests, each of which are briefly summarized here. The descriptions of these tests have been taken from [89].

VIII.1 Frequency (Monobit) Test

The focus of the test is the proportion of zeroes and ones for the entire sequence. The purpose of this test is to determine whether the number of ones and zeros in a sequence are approximately the same as would be expected for a truly random sequence. The test assesses the closeness of the fraction of ones to 0.5, that is, the number of ones and zeroes in a sequence should be about the same. All subsequent tests depend on the passing of this test.
VIII.2  Block Frequency Test

The focus of the test is the proportion of ones within $M$-bit blocks. The purpose of this test is to determine whether the frequency of ones in an $M$-bit block is approximately $M/2$, as would be expected under an assumption of randomness. For block size $M = 1$, this test degenerates to test 1, the Frequency (Monobit) test.

VIII.3  Cumulative Sums Test

The focus of this test is the maximal excursion (from zero) of the random walk defined by the cumulative sum of adjusted $(-1, +1)$ digits in the sequence. The purpose of the test is to determine whether the cumulative sum of the partial sequences occurring in the tested sequence is too large or too small relative to the expected behavior of that cumulative sum for random sequences. This cumulative sum may be considered as a random walk. For a random sequence, the excursions of the random walk should be near zero. For certain types of non-random sequences, the excursions of this random walk from zero will be large.

VIII.4  Runs Test

The focus of this test is the total number of runs in the sequence, where a run is an uninterrupted sequence of identical bits. A run of length $k$ consists of exactly $k$ identical bits and is bounded before and after with a bit of the opposite value. The purpose of the runs test is to determine whether the number of runs of ones and zeros of various lengths is as expected for a random sequence. In particular, this test determines whether the oscillation between such zeros and ones is too fast or too slow.
VIII.5 Longest Run Test

The focus of the test is the longest run of ones within $M$-bit blocks. The purpose of this test is to determine whether the length of the longest run of ones within the tested sequence is consistent with the length of the longest run of ones that would be expected in a random sequence. Note that an irregularity in the expected length of the longest run of ones implies that there is also an irregularity in the expected length of the longest run of zeroes. Therefore, only a test for ones is necessary.

VIII.6 Rank Test

The focus of the test is the rank of disjoint sub-matrices of the entire sequence. The purpose of this test is to check for linear dependence among fixed length substrings of the original sequence.

VIII.7 Discrete Fourier Transform (Spectral) Test

The focus of this test is the peak heights in the Discrete Fourier Transform of the sequence. The purpose of this test is to detect periodic features (i.e., repetitive patterns that are near each other) in the tested sequence that would indicate a deviation from the assumption of randomness. The intention is to detect whether the number of peaks exceeding the 95% threshold is significantly different than 5%.

VIII.8 Non-Overlapping Template Matching Test

The focus of this test is the number of occurrences of pre-specified target strings. The purpose of this test is to detect generators that produce too many occurrences of a given non-periodic (aperiodic) pattern. For this test and for the subsequent
Overlapping Template Matching test, an \( m \)-bit window is used to search for a specific \( m \)-bit pattern. If the pattern is not found, the window slides one bit position. If the pattern is found, the window is reset to the bit after the found pattern, and the search resumes.

**VIII.9 Overlapping Template Matching Test**

The focus of the Overlapping Template Matching test is the number of occurrences of pre-specified target strings. Both this test and the Non-overlapping Template Matching test use an \( m \)-bit window to search for a specific \( m \)-bit pattern. As with the non-overlapping test, if the pattern is not found, the window slides one bit position. The difference between this test and the non-overlapping test is that when the pattern is found, the window slides only one bit before resuming the search.

**VIII.10 Maurer’s “Universal Statistical” Test**

The focus of this test is the number of bits between matching patterns (a measure that is related to the length of a compressed sequence). The purpose of the test is to detect whether or not the sequence can be significantly compressed without loss of information. A significantly compressible sequence is considered to be non-random.

**VIII.11 Approximate Entropy Test**

As with the Serial test, the focus of this test is the frequency of all possible overlapping \( m \)-bit patterns across the entire sequence. The purpose of the test is to compare the frequency of overlapping blocks of two consecutive/adjacent lengths \( (m \text{ and } m + 1) \) against the expected result for a random sequence.
VIII.12 Random Excursions Test

The focus of this test is the number of cycles having exactly $K$ visits in a cumulative sum random walk. The cumulative sum random walk is derived from partial sums after the $(0, 1)$ sequence is transferred to the appropriate $(-1, +1)$ sequence. A cycle of a random walk consists of a sequence of steps of unit length taken at random that begin at and return to the origin. The purpose of this test is to determine if the number of visits to a particular state within a cycle deviates from what one would expect for a random sequence. This test is actually a series of eight tests (and conclusions), one test and conclusion for each of the states: $-4, -3, -2, -1$ and $+1, +2, +3, +4$.

VIII.13 Random Excursions Variant Test

The focus of this test is the total number of times that a particular state is visited (i.e., occurs) in a cumulative sum random walk. The purpose of this test is to detect deviations from the expected number of visits to various states in the random walk. This test is actually a series of eighteen tests (and conclusions), one test and conclusion for each of the states: $-9, -8, -1 and +1, +2, +9$.

VIII.14 Serial Test

The focus of this test is the frequency of all possible overlapping $m$-bit patterns across the entire sequence. The purpose of this test is to determine whether the number of occurrences of the $2m$ $m$-bit overlapping patterns is approximately the same as would be expected for a random sequence. Random sequences have uniformity; that is, every $m$-bit pattern has the same chance of appearing as every other $m$-bit pattern. Note that for $m = 1$, the Serial test is equivalent to the Frequency (Monobit) test.
VIII.15 Linear Complexity Test

The focus of this test is the length of a linear feedback shift register (LFSR). The purpose of this test is to determine whether or not the sequence is complex enough to be considered random. Random sequences are characterized by longer LFSRs. An LFSR that is too short implies non-randomness.
REFERENCES


[65] J. Lü, F. Han, X. Yu, and G. Chen, “Generating 3-D multi-scroll chaotic at-


nonlinear transconductor,” *IEE Electronics Letters*, vol. 38, no. 14, pp. 685–686,
2002.

[68] G. Xie, P. Chen, and M. Liu, “Generation of multidirectional multiscroll attrac-
tors under the third-order jerk system,” *Proceedings of the International Sympo-

hysteresis based time-delay differential equation,” *International Journal of Bi-

[70] J. Lü, S. Yu, and G. Chen, “Experimental verification of multidirectional multi-
scroll attractors,” *IEEE Transactions on Circuits and Systems.I: Regular Papers*,

[71] J. Lü and G. Chen, “Generating multiscroll chaotic attractors: Theories, meth-
ods and applications,” *International Journal of Bifurcation and Chaos*, vol. 16,
no. 4, pp. 775–858, 2006.


[94] R. B. Davies, “Exclusive or (xor) and hardware random number generators,” http://www.robertnz.net/pdf/xor2.pdf [Ret. 28/02/2012].