



INSTITUTO FEDERAL

Ceará

Campus Fortaleza



PPGET - IFCE

Programa de Pós-graduação em
Engenharia de Telecomunicações

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E
TECNOLOGIA DO CEARÁ-IFCE**

**PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E
INOVAÇÃO**

**COORDENAÇÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA DE
TELECOMUNICAÇÕES**

JORGE FREDERICSON DE MACEDO COSTA DA SILVA

**CARACTERIZAÇÃO DE GERADOR FÍSICO DE NÚMEROS ALEATÓRIOS
BASEADO EM EFEITO AVALANCHE USANDO PLATAFORMAS
COMPUTACIONAIS EMBARCADAS**

FORTALEZA – CE

2016

JORGE FREDERICSON DE MACEDO COSTA DA SILVA

**CARACTERIZAÇÃO DE GERADOR FÍSICO DE NÚMEROS ALEATÓRIOS
BASEADO EM EFEITO AVALANCHE USANDO PLATAFORMAS
COMPUTACIONAIS EMBARCADAS**

Dissertação apresentada à Coordenação do Programa de Pós-Graduação em Engenharia de Telecomunicações (PPGET/IFCE) como requisito final para a obtenção do grau de mestre em Engenharia de Telecomunicações.

Área de concentração: Informação Quântica

Orientador: Prof. Dr. Fábio Alencar Mendonça

FORTALEZA – CE
2016

Catálogo na fonte: Biblioteca Waldyr Diogo de Siqueira

S586c Silva, Jorge Fredericson de Macedo Costa da.

Caracterização de gerador físico de números aleatórios baseado em efeito avalanche usando plataformas computacionais embarcadas . / Jorge Fredericson de Macedo Costa da Silva. - Fortaleza: IFCE, 2016.

f.102

Dissertação (Engenharia de Telecomunicações).

Orientador: Fábio Alencar Mendonça

1.COMUNICAÇÃO QUÂNTICA. 2.SISTEMA EMBARCADO. 3.EFEITO AVALANCHE 4. GERAÇÃO DE NÚMEROS ALEATÓRIOS. Título.

CDD 621.38150285

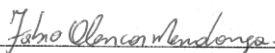
JORGE FREDERICSON DE MACEDO COSTA DA SILVA

**CARACTERIZAÇÃO DE UM GERADOR FÍSICO DE NÚMEROS
ALEATÓRIOS BASEADO EM EFEITO AVALANCHE USANDO
PLATAFORMAS COMPUTACIONAIS EMBARCADAS**

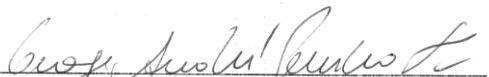
Dissertação apresentada ao Programa de
Pós-Graduação em Engenharia de
Telecomunicações do Instituto Federal de
Educação, Ciência e Tecnologia do Ceará,
como requisito parcial para a obtenção do
título de Mestre em Engenharia de
Telecomunicações. Área de Concentração:
Informação Quântica.

Aprovada em: 30/05/2016

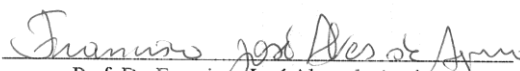
BANCA EXAMINADORA



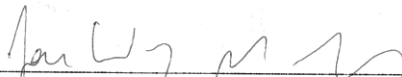
Prof. Dr. Fábio Alencar Mendonça
Instituto Federal de Educação, Ciência e Tecnologia do Ceará – IFCE



Prof. Dr. George André Pereira Thé
Universidade Federal do Ceará – UFC



Prof. Dr. Francisco José Alves de Aquino
Instituto Federal de Educação, Ciência e Tecnologia do Ceará – IFCE



Prof. Dr. José Wally Mendonça Menezes
Instituto Federal de Educação, Ciência e Tecnologia do Ceará – IFCE

A Deus.
Aos meus avós e meus pais.
Aos meus amigos e a todos os homens de bem.

AGRADECIMENTOS

Primeiramente, a Deus pelo dom da vida, saúde e pelo sentimento de curiosidade que me guia desde a infância e que me acompanha até os dias atuais, reforçando o meu gosto pela Pesquisa, Desenvolvimento e Inovação.

Aos meus avós, pais e irmã, que sempre me incentivam, acompanham e dividem comigo todos os momentos desde os primeiros suspiros. A eles o meu eterno agradecimento por todos os esforços feitos ao longo da minha trajetória educacional.

A minha companheira, Katielle Dantas, que ainda na graduação, já dividia os momentos de pesquisa, desde primeiro artigo aceito, e hoje colega neste programa de pós-graduação e sempre dando o apoio incondicional nos momentos chave que norteiam minha carreira acadêmica.

Ao prof. Dr. Fábio Alencar Mendonça pela disposição e paciência na orientação deste trabalho. Pelas tardes e noites de discussão no Laboratório GDESTE sobre o tema desta dissertação e as perspectivas que este trabalho pode propiciar.

Aos colegas de profissão e amigo prof. Dr. José Wally Mendonça Menezes e Prof. Msc Carlos Maurício Jaborandy de Mattos Dourado Jr. pelos conselhos dados ao longo da jornada acadêmica e pelos debates sobre lecionar no IFCE, sobretudo o papel da pesquisa num campus do interior do estado. Aos caros professores do Programa de Pós-Graduação em Engenharia de Telecomunicações do IFCE por toda a contribuição.

Aos colegas que fizeram, fazem e farão o Grupo de Desenvolvimento em Sistemas de Telecomunicações e Sistemas Embarcados (GDESTE). Em especial, ao amigo pra todas as horas Alexandre Castro, com que tenho o privilégio de coordenar as ações do Grupo, ao trio Renan Soeiro, Maurício Almeida e Fillipe Diego, com quem tenho a honra de dividir momentos de alegria desde a graduação e a Bertrand Wadi e Wesley Alves com quem tive o prazer de dividir alguns momentos deste trabalho.

Por fim, aos companheiros de trabalho do IFCE, *campus* Tauá. Sobretudo, os docentes do curso de Telemática: Tiago Façanha, Edmilson Moreira, Kaio Jonathas, Dayse Correia e Gilderlan Tavares.

Muito obrigado a todos.

“Uma experiência nunca é um fracasso,
pois sempre vem a nos mostrar algo.”
(Thomas A. Edison)

RESUMO

A geração de números aleatórios tem uma grande variedade de aplicações, desde simulações numéricas para eventos meteorológicos, jogos e loterias, geração de ruído para *jammer* em camada física, protocolos de segurança e criptografia, etc. Assim, ela pode ser implementada de duas maneiras: uma física (*hardware*) ou lógica (*software*). Do ponto de vista físico, é possível gerar blocos lógicos digitais ou usar componentes eletrônicos e optoeletrônicos para servir como fonte de entropia não determinística. Tais fontes de entropia são sinais inerentes à natureza da estrutura física dos elementos e, no geral, são baseado em efeitos físicos aleatórios ou quânticos. Já do ponto de vista lógico, o mais comum é a implementação de uma fórmula matemática e, a partir de uma semente, a sequência vai evoluindo na geração dos valores. Ainda assim, quando há a geração de sequência ditas aleatórias, é necessário validar tal efeito randômico. Isso é feito por meio de alguns órgãos que instituem normas e testes para aferir o grau de aleatoriedade das amostras a serem analisadas. Com base nisso, este trabalho propõe o desenvolvimento de um sistema embarcado que irá realizar leituras de uma fonte de entropia, baseada no efeito avalanche de semicondutores, e analisar as amostras com base nos testes propostos pela entidade certificadora para checar a eficiência do sistema proposto para uma eventual aplicação em Sistemas de Comunicações Quânticas.

Palavras-chave: Geração de Números Aleatórios; Efeito Avalanche; Comunicações Quânticas; Sistemas Embarcados.

ABSTRACT

The random number generation is a technique that consists of a variety of applications like numerical simulations for weather events, games and lotteries, noise generation for jammer in physical layer, security and encryption protocols and so on. Thus, this technique can be implemented in two ways: physical (hardware) or logical (software). From the physical point of view, it is possible to generate digital logic blocks or it can use electronic and optoelectronic components to serve as a source of not deterministic entropy. Such entropy sources are inherent to the nature of the physical structure of the elements and, in general, are based on random or quantum physical effects. On the other way from the logical point of view, commonly, a mathematical formula is implemented and, from a seed, the sequence evolves in the generation of values. But still, when there is generation of random sequences, it is necessary validate such random effect. This validation could be done by some agencies which have the function to establish standards and tests to assess the randomness degree of the analyzed samples. According to this, this paper proposes the development of an embedded system that will perform readings from the entropy source based on the avalanche breakdown of semiconductors and analyzing the samples based on the tests proposed by the certifier to check the system's efficiency proposed for possible application in quantum Communications Systems.

Keywords: Random Number Generation; Avalanche effect; Quantum communications; Embedded systems.

Lista de Figuras

Figura 1.1 – Exemplo de <i>hardware</i> gerador de sequência pseudoaleatória	1
Figura 2.1 – Blocos Lógicos do GNPA implementado no ambiente ISE da Xilinx	8
Figura 2.2 – Montagem em bancada de um GNPA usando FPGA	9
Figura 2.3 – Diodo semicondutor formado pelas junções do tipo “n” e tipo “p”	11
Figura 2.4 – Polarização de semicondutor formado por junção do tipo “p” e do tipo “n”. (a) Polarização Direta de um Diodo. (b) Polarização Reversa de um Diodo	11
Figura 2.5 – Curva característica de um semicondutor de Silício	14
Figura 2.6 – Região de Ruptura de um semicondutor de Silício	15
Figura 2.7 – Curva característica de um Diodo Túnel em comparação a um Diodo comum	17
Figura 2.8 – Ilustração de uma partícula incidente numa barreira de potencial retangular	18
Figura 2.9 – Análise da Barreira de Potencial e probabilidade de Transmissão	19
Figura 2.10 – Análise da Barreira de Potencial para um Diodo Túnel	20
Figura 2.11 – Representação da evolução temporal de uma partícula quântica sobre o efeito do observador por meio da Esfera de Bloch	22
Figura 2.12 – Experimento para validar a superposição dos estados quânticos	23
Figura 2.13 – Arquiteturas internas do FPGA: Matriz Simétrica, <i>Sea-of-Gates</i> , <i>Row-Based</i> e PLD Hierárquico	28
Figura 2.14 – Dispositivos internos de um FPGA. (a) CLB, IOB e SB. (b) Funcionalidades do SB	29
Figura 2.15 – Plataforma de Desenvolvimento Arduino UNO Revisão 3	30
Figura 2.16 – Plataforma de Desenvolvimento Arduino DUE	31
Figura 2.17 – Plataforma FPGA SPARTAN 3E XILINX	31

Figura 3.1 – Exemplo de extrator de sinal para GFNA. “A” é um amplificador e “C” amplificador comparador com ajuste de limiar. O resultado gera uma sequência por meio de Flip-Flop tipo D	35
Figura 3.2 – Dois tipos de FROs, a esquerda um de alta velocidade e a direita um de baixa velocidade. A frequência é determinada pelos delays internos e capacitâncias esporádicas	36
Figura 3.3 – Modelo de aquisição de dados com alta entropia proposto	37
Figura 3.4 – Aquisição de Dados via sensor de imagens presente em celulares	38
Figura 3.5 – Uso de cristal fotônico PPKTP em cenários totalmente ópticos	39
Figura 3.6 – GQNA <i>Quantis</i>	40
Figura 3.7 – Modelo proposto para GQNA usando único foto detector	40
Figura 3.8 – Modelo proposto para GQNA usando sensor CCD ou CMOS	41
Figura 3.9 – Proposta de GQNA de alta velocidade usando Fotodiodo PIN	42
Figura 3.10 – GQNA Óptico de alta velocidade e alta capacidade	42
Figura 3.11 – Estrutura de funcionamento do LRNG	43
Figura 3.12 – TRNG baseado em efeito de semicondutores	45
Figura 3.13 – RNG baseado em efeito avalanche pós processado por blocos lógicos digitais	46
Figura 3.14 – Gerador T12RNG	47
Figura 3.15 – Gerador baseado em FRO implementado num FPGA	48
Figura 3.16 – RNG baseado em emissão/detecção de fótons coerentes	48
Figura 3.17 – RNG com fotorreceptor PIN	49
Figura 4.1 – Fluxograma do Experimento com Efeito Avalanche	51
Figura 4.2 – Circuito de geração e amplificação do sinal de Avalanche usando transistores, fase analógica	52
Figura 4.3 – Inversor Schmitt Trigger de 6 canais, fase digital	52

Figura 4.4 – Caracterização do Efeito Avalanche	53
Figura 4.5 – Esquemático e PCB do GFNA manufaturado	53
Figura 4.6 – Sistema de Testes das placas do GFNA em bancada para análise de alimentação, ruído e distorções do sinal de avalanche	54
Figura 4.7 – Ensaio do GFNA com Arduíno DUE para coleta das amostras e execução do pós processamento e dos testes de aleatoriedade	54
Figura 4.8 – Ensaio do GFNA com FPGA para coleta das amostras e execução do pós processamento e dos testes de aleatoriedade	54
Figura 4.9 – Fluxograma da Lógica do Algoritmo de von Neumann	55
Figura 4.10 – Efeitos Avalanche a uma tensão de 18V avaliado na saída do transistor Q3	56
Figura 4.11 – Efeitos Avalanche a uma tensão de 18V avaliado na saída do ST, sinal digitalizado	56
Figura C.1 – Esquemático da Placa de Aquisição do Sinal de Avalanche	70
Figura C.2 – Roteamento da Placa de Aquisição do Sinal de Avalanche	70
Figura C.3 – Vista 3D da Placa Manufaturada	70
Figura D.1 – Componente que recebe o vetor do GFNA e componente de comunicação Serial	71
Figura C.9.1 – Divisão das amostras em blocos para teste	80

Lista de Tabelas

Tabela 2.1: Resumo dos Teste Estatísticos propostos pelo NIST	25
Tabela 2.2: Principais placas da plataforma de desenvolvimento Arduíno	27
Tabela 4.1: Comparativo das Plataformas e seu desempenho nos Testes	57
Tabela C.4.1: Tamanho mínimo das amostras em relação ao tamanho do bloco	77
Tabela C.4.2: Relação entre os valor M, K e N	78

Lista de Acrônimos

AC - Autômato Celular

APD - *Avalanche Photodiode*

ARM - *Advanced RISC Machine*

ASIC - *Application Specific Integrated Circuits*

BCD - *Binary-Coded Decimal*

BJT - *Base Junction Transistor*

CCD - *Charge-coupled device*

CLB - *Configurable Logic Block*

CLPD - *Complex Programmable Logic Device*

CMOS - *Complementary Metal Oxide Semiconductor*

dB - deciBell

DSP - *Digital Signal Processor*

FGPA - *Field Programmable Gate Array*

FRO - *Free Running Oscillator*

GFNA - Gerador Físico de Números Aleatórios

GNPA - Gerador de Números Pseudoaleatórios

GQNA – Gerador Quântico de Números Aleatórios

GSL – *GNU Scientific Library*

HDL - *Hardware Description Language*

I²C - *Inter-Integrated Circuit*

IDE - *Integrated Development Environment*

IEEE - *Institute of Electrical and Electronics Engineers*

INPI - Instituto Nacional da Propriedade Industrial

IOB - *Input/Output Block*

LCD - *Liquid Crystal Display*

LED - *Light Emitting Diode*

LRNG - *Linux Random Number Generator*

NIST - *National Institute of Standards and Technology*

PCI - *Placa de Circuito Impresso*

PIA - *Programmable Interconnection Array*

PLD - *Programmable Logic Device*

PPKTP - *Periodically-Poled Potassium Titanyl Phosphate*

PWM - *Pulse Width Modulation*

Qubit - *Quantum Bit*

R&S - *Rohde & Schwarz*

RF - *Rádio Frequência*

RNGs – *Random Numbers Generators*

SoC - *System-on-a-chip*

SPI - *Serial Peripheral Interface*

ST - *Schmitt Trigger*

TRNG – *True Random Number Generation*

TTL - *Transistor-Transistor Logic*

UART - *Universal Asynchronous Receiver/Transmitter*

USB - *Universal Serial Bus*

USPTO - *United States Patent and Trademark Office*

XOR - *Exclusive OR*

SUMÁRIO

1. INTRODUÇÃO E ORGANIZAÇÃO DO TEXTO	1
1.1 Motivações do Trabalho	3
1.2 Objetivos	4
1.2.1 Objetivo Geral	4
1.2.2 Objetivos Específicos	4
2. GERADORES DE NÚMEROS ALEATÓRIOS	5
2.1 Geradores de Números Pseudoaleatórios	6
2.2 Geradores Físicos de Números Aleatórios	9
2.2.1 Efeitos Físicos em Semicondutores	10
2.2.2 Efeitos Zener e Efeito Avalanche	15
2.2.3 Diodo Túnel e Tunelamento Quântico	17
2.3 Geradores Quânticos de Números Aleatórios	20
2.4 Testes Estatísticos	23
2.5 Plataformas Computacionais Embarcadas	26
2.5.1 Arduíno UNO	28
2.5.2 Arduíno Due	29
2.5.2 FPGA Xilinx Spartan 3E	30
3. REVISÃO BIBLIOGRÁFICA	32
3.1 Geradores de Números Pseudoaleatórios	32
3.2 Fontes de Entropia não determinísticas e seus usos em GFNA	34
3.3 Geradores Quânticos de Números Aleatórios	39
3.4 GFNA Baseados em Sistemas Digitais	43
3.5 GFNA Baseados em Sistemas Embarcados e FPGA	46
3.6 Patentes	48

3.7 Conclusões	49
4. METODOLOGIA E RESULTADOS	50
4.1 Materiais e Métodos	50
4.1.1 Experimento com Efeito Avalanche	51
4.1.2 Metodologia utilizada	52
4.2 Resultados	55
4.2.1 Fontes de Entropia	55
4.2.2 Eficiência do Algoritmo de Pós Processamento	57
4.2.2 Testes Estatísticos	58
4.3. Análise dos Resultados	58
4.3.1 Fontes de Entropia	58
4.3.2 Eficiência do Algoritmo de Pós Processamento	59
4.3.3 Testes Estatísticos	59
3.7 Conclusões	60
5. CONCLUSÕES E PERSPECTIVAS	61
5.1 Conclusões	61
5.2 Perspectivas par novos trabalhos	62
5.3 Trabalhos decorrentes desta dissertação	62
REFERÊNCIAS BIBLIOGRÁFICAS	63
APÊNDICE	67
APÊNDICE A – Projetos de Iniciação Científica Aprovados	68
APÊNDICE B – Orientações de Trabalho de Conclusão de Curso (TCC)	69
APÊNDICE C – Projeto de Placas	70
APÊNDICE D – Blocos Lógicos Implementados	71
ANEXOS	72
ANEXO A – Algoritmo de von Neumann	73

ANEXO B – Distribuições de Probabilidades Específicas	74
B.1 – Distribuição Normal	74
B.2 – Função Erro Complementar	74
B.3 – Função Gama Incompleto	74
ANEXO C – Testes Estatísticos sugeridos pelo NIST	75
C.1 – Teste de Frequência	75
C.2 – Teste da Frequência em um Bloco	75
C.3 – Teste de Corrida	76
C.4 – Teste para a Maior Corrida de 1s dentro de um Bloco	77
C.5 – Teste de Posto da Matriz Binária	78
C.6 – Teste Espectral da Transformada Discreta de Fourier	78
C.7 – Teste de Casamento de Padrão sem Superposição	79
C.8 – Teste de Casamento de Padrão com Superposição	80
C.9 – Teste de Estatística Universal de Maurer	80
C.10 – Teste de Complexidade Linear	81
C.11 – Teste Serial	82
C.12 – Teste da Entropia Aproximada	82
C.13 – Teste de Somas Cumulativas	83
C.14 – Teste de Excursões Aleatórias	83
C.15 – Teste Variante de Excursões Aleatórias	84

CAPÍTULO 1

INTRODUÇÃO E ORGANIZAÇÃO DO TEXTO

Muitos métodos e modelos nas Ciências Exatas e Engenharias necessitam de números aleatórios, ou mais precisamente geradores de números aleatórios não determinísticos, para o estudo e análise de fenômenos relacionados à natureza destes processos. São exemplos desses processos: cálculos baseado no método de Monte Carlo, simulações numéricas, pesquisas estatísticas, funções aleatórias, meteorologia, loteria, etc. Atualmente, esses números são focados basicamente em aplicações que usam criptografia e segurança cibernética, como *Smart Grid*, *e-banking*, transações via Internet, dentre outras (STIPČEVIĆ, 2011).

Para essas aplicações, são usadas diversas formas para geração desses números aleatórios, podendo ser usado um *hardware* extra ao sistema, como o da Figura 1.1, ou por meio de algoritmos para geração de sequências pseudoaleatórios a partir de uma semente, *Seed*. Esses algoritmos são facilmente encontrados na literatura, a citar alguns tipos: geradores congruentes lineares, autômato celulares, geradores baseados em sistemas caóticos, geradores de deslocamento retroalimentados, etc.

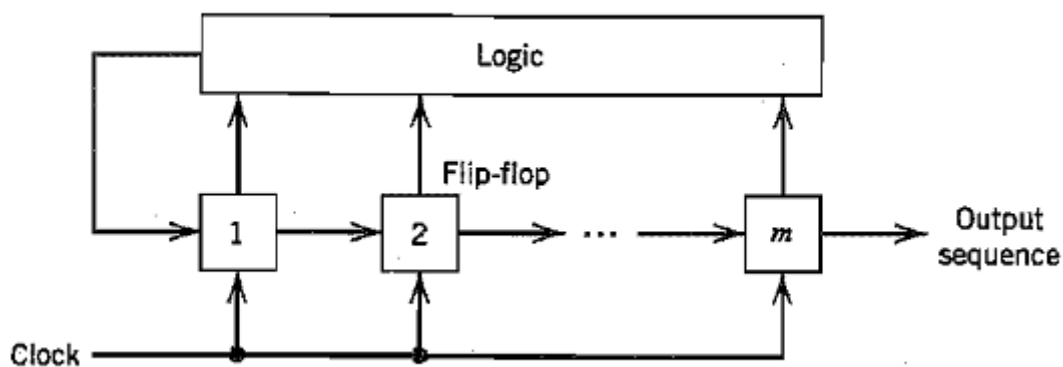


Figura 1.1 – Exemplo de *hardware* gerador de sequência pseudoaleatória [11].

Partindo do *hardware*, tem-se um grande leque a ser explorado tanto como fonte de sinal para gerador quanto como componentes para fabricação e *design* de geradores. Os componentes semicondutores, *Flip-Flops* e elementos optoeletrônicos fazem o papel

de fonte, já as portas lógicas e os FPGAs (*Field Programmable Gate Array*) dão suporte para construção ou aprimoramento deles.

O funcionamento desses geradores são baseados nos efeitos gerados a partir construção dos elementos semicondutores. Efeitos como ionização de impacto, ruído Zener, Efeito Avalanche, *jitter*, detecção de fótons únicos, dentre outros, são usados como entrada em sistemas digitais, FPGAs ou mesmo em Microcontroladores para compor blocos de sistemas de simulação, sistemas de comunicação ou segurança da informação.

Após a coleta das sequências por estas fontes, é necessário uma etapa de processamento destas amostras, pois elas podem estabelecer uma correlação entre os bits gerados. Isso ocorre, sobretudo por conta do processo de fabricação dos elementos, ainda que eles tenham natureza aleatória. Assim, há uma série de algoritmos destinados ao pós processamento das fontes geradoras de sinal.

Por último, as amostras para serem classificadas como aleatórias devem ser testadas. Esses testes são propostos por uma série de entidades e órgãos a depender do contexto no qual o gerador será usado. Um deles é o *National Institute of Standards and Technology* (NIST) que oferece uma bateria com quinze testes a fim de validar as sequências como aleatórias.

Nesse contexto, a escolha do gerador deve ser feita de maneira a satisfazer as condições implicadas ao tipo de fenômeno a ser analisado, portanto deixando margem nula para escolhas arbitrárias. Um meio de se obter essa escolha é com base em uma série de testes estatísticos, lembrando que para alguns processos nem sempre tais testes indicarão qual metodologia é a ideal. Por exemplo, em esquemas criptográficos a geração de números aleatórios é vital ao processo, porque o uso da sequência aleatória irá ditar o grau de segurança associado à mensagem transmitida, como no algoritmo *One-time-Pad*.

Assim, esta dissertação é composta por cinco capítulos que apresentam as etapas de realização e conclusão da pesquisa referente aos geradores de números aleatórios.

No Capítulo 2, será apresentada uma abordagem sobre os geradores de números pseudoaleatórios e geradores de números aleatórios, bem como os elementos e os efeitos físicos que introduzem entropia não determinística aos sistemas. Aqui, também será mostrado os testes estatísticos e as plataformas computacionais usadas no trabalho.

O Capítulo 3 está direcionado ao estudo do estado da arte sobre os dispositivos geradores de números aleatórios e geradores quânticos de números aleatórios. Assim, o capítulo traz uma série de ensaios usando os mais diversos componentes eletrônicos e optoeletrônicos e as mais diversas plataformas de eletrônica digital e computacionais embarcadas.

No Capítulo 4, será descrito os ensaios e os materiais usados ao longo da pesquisa. Além disso, serão apresentados os equipamentos utilizados para levantar as medidas em laboratório, a metodologia empreendida na realização dos experimentos práticos, as abordagens relacionadas aos testes de aleatoriedade e os resultados alcançados a fim de fortalecer a aplicabilidade deste dispositivo.

Por fim, no Capítulo 5 será mostrado as conclusões desta pesquisa, ressaltando as vantagens e desvantagens do sistema proposto, além de citar desdobramento e perspectivas a partir deste trabalho.

1.1 Motivações do trabalho

Nas Ciências Exatas e na Engenharia a previsibilidade é uma característica muito importante no momento de planejar ações em determinadas aplicações. Por exemplo, na previsão do tempo há uma série de variáveis a serem calculadas para chegar a uma estimativa sobre as próximas vinte e quatro (24) horas do dia. Para tanto, tal modelo deve ser dotado de uma estatística capaz de chegar ao mais próximo do real.

Assim como na previsão do tempo, em sistemas geração Eólica a previsão do vento nas próximas horas é muito importante, porque a incidência do vento nas pás é mensurada conforme uma distribuição probabilística, geralmente a distribuição de *Weibull*. Essa característica pode ser encontrada num modelo de geração de números aleatórios (RNGs – *Random Numbers Generators*).

Aplicações na camada física também podem fazer uso dos geradores de sinal aleatório. Em aparelhos bloqueadores de sinal, ou em *jammers*, há um bloco no sistema dedica a geração, seja de sinal, seja de uma sequência de bits aleatórios.

Um outro cenário em que os RNGs geram bastante expectativas é na criptografia, pois numa sequência pseudoaleatória pode pôr em risco a integridade da chave e, por conseguinte, da mensagem caso a sequência evolutiva seja descoberta. No entanto, um desses geradores para serem implementados no campo da criptografia devem passar por

uma bateria de testes para validar sua condição de aleatoriedade e, assim, garantir segurança aos usuários dessa tecnologia.

Atualmente, os modelos que empregam essas técnicas podem ser divididos em algoritmos e blocos lógicos pseudoaleatórios (GNPA) ou por meio do uso de elementos com fonte de entropia não determinística formando os Geradores Físicos de Números Aleatórios (GFNA) e há também os que usando efeitos quânticos para obtenção de tal entropia chamados Geradores Quânticos de Números Aleatórios (GQNA – *Quantum Random Numbers Generators*).

A partir dessa necessidade, este trabalho procura a caracterização de um gerador físico de números aleatórios, bem como a sabatina das sequências geradas em testes de aleatoriedade fornecidos pelo NIST em algumas plataformas computacionais embarcadas. O gerador a ser caracterizado nesta dissertação abrirá uma perspectiva para a criação de GQNA a partir dos estudos feitos nessa pesquisa.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é desenvolver um dispositivo capaz de gerar números aleatórios com uma fonte de entropia não determinística baseada no efeito avalanche junto a uma plataforma de Sistemas Embarcados *open source* e uma reconfigurável do tipo FGPA.

1.2.2 Objetivos Específicos

1. caracterizar dispositivos eletrônicos como geradores de números aleatórios;
2. elaborar uma rotina de testes dos bits gerados;
3. descrever um *Software-Core* para aquisição dos bits para FGPA;
4. desenvolver uma prova de conceito sobre geradores eletrônicos de números aleatórios;
5. criar um dispositivo gerador de números aleatórios seguro e confiável de baixo custo.

CAPÍTULO 2

Neste capítulo, será apresentado uma descrição dos geradores de números pseudoaleatórios, gerador físico de números aleatórios e geradores quânticos de números aleatórios. Em seguida, serão descritos os testes implementados para testar as sequências geradas e as três plataformas computacionais embarcadas usadas nos testes. Ressalta-se que o detalhamento das fontes de entropia não determinísticas, assim como a exposição sobre os efeitos físicos que a geram será detalhado na subsecção de geradores físicos.

GERADORES DE NÚMEROS ALEATÓRIOS

Um número aleatório é um número gerado por um processo cujo resultado não pode ser previsível e que futuramente não possa ser representado de maneira confiável, ou seja, o bit anterior não pode fornecer elementos para estimar o próximo bit da sequência. Logo, o desafio está em desenvolver uma metodologia ou um dispositivo que permita satisfazer essa condição para implantar tais números nas aplicações (IDQ, 2010).

No entanto, definir se um número qualquer é ou não produzido por um gerador é simplesmente improvável, daí o estudo de aleatoriedade dos processos de geração serem baseados em sequências. Dado isso, para verificar se tais sequências são ou não aleatórias e se ela tem tamanho infinito, basta aplicar o teorema de Shannon para avaliar se a quantidade de informação presente nela é também infinita. Ora, na computação um recurso infinito é não implementável, posto que o computador é uma máquina de estados finitos. Logo, tal exercício mostra-se bastante complexo de ser reproduzido (IDQ, 2010).

Agora, para o caso de uma sequência finita de números gerados é possível atestar sua aleatoriedade. Isso é uma tarefa difícil e complicada, pois é necessário verificar as características estatísticas da sequência, como a equiprobabilidade de todos os números. Por exemplo, numa sequência binária, composta apenas por zeros e uns, o bit zero tem que ter a mesma probabilidade de ocorrência do bit um, independentemente da sequência a ser analisada, não deixando margem para dúvidas ou senso comum (IDQ, 2010).

Nesse cenário, temos duas definições bem interessantes sobre como implementar uma sequências prática de números aleatórios. Segundo Knuth (1981), uma sequência de números aleatórios é uma sequência de números independentes com uma distribuição

específica e uma probabilidade específica de assumir qualquer valor em um dado intervalo de valores. Já de acordo com Schneier (1996), uma sequência que tem as mesmas propriedades estatísticas como bits aleatórios, é imprevisível e não pode ser reproduzida de forma confiável.

Assim, é enfatizado em ambas definições que a sequência para ser considerada aleatória não pode haver correlação entre os bits gerados. Portanto, a partir deste ponto, quando for referenciado uma sequência de bits aleatórios, será considerado o caso de um gerador prático (IDQ, 2010).

Todas essas exigências são para validar o uso dos geradores em uma série de aplicações nas quais os números aleatórios podem oferecer uma visão diferente sobre o problema a ser analisado. Em resumo, a seguir temos um panorama geral de onde eles podem aplicados:

1. criptografia;
2. cálculos científicos;
3. jogos;
4. loterias.

A seguir, será dado enfoque na construção de três tipos de geradores: Geradores de Números Pseudoaleatórios, Geradores Físicos de Números Aleatórios e Geradores Quânticos de Números Aleatórios.

2.1 Geradores de Números Pseudoaleatórios

Geradores de números pseudoaleatórios são bastante conhecidos na literatura e já são usados há bastante tempo na Ciência da Computação e Telecomunicações. Eles são, na maioria dos casos, modelos matemáticos que produzem sequências determinísticas a partir de uma entrada inicial, chamada semente (*Seed*). Logo, é possível prever toda a sequência sabendo o evento inicial do sistema. É importante lembrar que em algumas aplicações esses geradores não comprometem o resultado final (STIPČEVIĆ, 2011).

A grande vantagem desses geradores são seu custo baixo, tanto financeiro quanto computacional, facilidade de implementação e familiaridade no ambiente dos computadores tradicionais. Agora, possuem algumas desvantagens nas aplicações citadas

anteriormente. Por exemplo, para uma sequência de bits muito grande, a correlação entre os bits gerados reduz a confiabilidade do sistema de criptografia (COSTA, 2011).

A maneira mais simples e rápida de se implementar um GNPA é a implementação de uma equação matemática em um computador digital, onde se observa as seguintes características:

- período de repetição;
- distribuição probabilística, geralmente uniforme;
- geração rápida em relação ao custo computacional.

Um exemplo dessas sequências é o algoritmo do Quadrado do Meio proposto por von Neumann. Esse algoritmo parte de uma semente, um número qualquer, e partir disso ele é elevado ao quadrado, daí os dígitos centrais são usados como os próximos elementos da sequência, é importante definir apenas a quantidade de bits a serem usados para efetuar o corte, ou preencher com zeros quando necessário. O exemplo a seguir mostra como obter 11 passos a partir de uma semente x_0 , onde são selecionados três dígitos:

- $x_0 = 121$;
- $(121)^2 = 014641 \rightarrow x_1 = 464$;
- $(464)^2 = 215296 \rightarrow x_2 = 529$;
- $(529)^2 = 279841 \rightarrow x_3 = 984$;
- $(984)^2 = 968256 \rightarrow x_4 = 825$;
- $(825)^2 = 680625 \rightarrow x_5 = 062$;
- $(062)^2 = 003844 \rightarrow x_6 = 384$;
- $(384)^2 = 147456 \rightarrow x_7 = 745$;
- $(745)^2 = 555025 \rightarrow x_8 = 502$;
- $(502)^2 = 252004 \rightarrow x_9 = 200$;
- $(200)^2 = 040000 \rightarrow x_{10} = 000$;
- $(000)^2 = 000000 \rightarrow x_{11} = 000$;

O exemplo acima mostra que o algoritmo apresenta um problema quando a sequência gerada é composta por bits zeros, porque todos os bits seguintes nos próximos passos da sequência também serão zero. Além disso, é possível também haver repetição da sequência ao longo do tempo. Outros exemplos de algoritmos de GNPA serão apresentados no Capítulo 3.

Na Figura 2.1, é possível ver uma outra forma de implementação de GNPA. Essa é feita em FPGA de um gerador de números pseudoaleatório e na Figura 2.2 sua versão física em bancada. No seu funcionamento, é possível perceber sua ordem e o seu período de geração. A ordem indica o conjunto de valores que é fornecido inicialmente e o período aponta para o tamanho da sequência gerada até a primeira repetição.

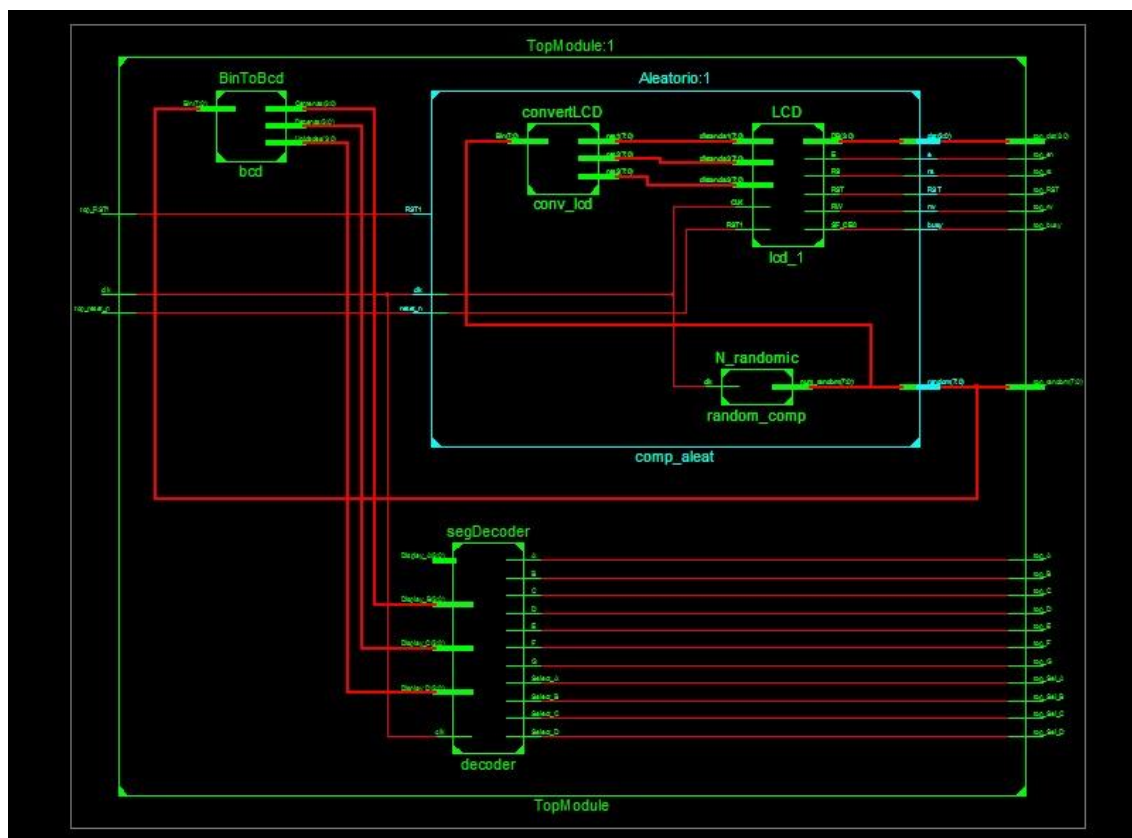


Figura 2.1 – Blocos Lógicos do GNPA implementado no ambiente ISE da Xilinx.

Fonte: Autor

Os blocos da figura acima representam os circuitos lógicos digitais que implementam o gerador PRNG. O bloco “BinToBCD” faz a implementação da codificação BCD (*Binary-Coded Decimal*) que nada mais é que uma conversão binária para decimal. Já os blocos “convertLCD” e “LCD” fazem as funções lógicas de decodificação da informação de saída do FPGA para o periférico display de LCD presente na placa. De modo análogo, o “segDecoder” implementa a conversão dos sinais do FPGA via registrador de deslocamento para a exibir a informação nos displays de sete segmentos. Por fim, o bloco “N_randomic” é um *hardware* implementado a partir do modelo da Figura 1.1. O resultado da montagem pode ser visto na Figura 2.2.

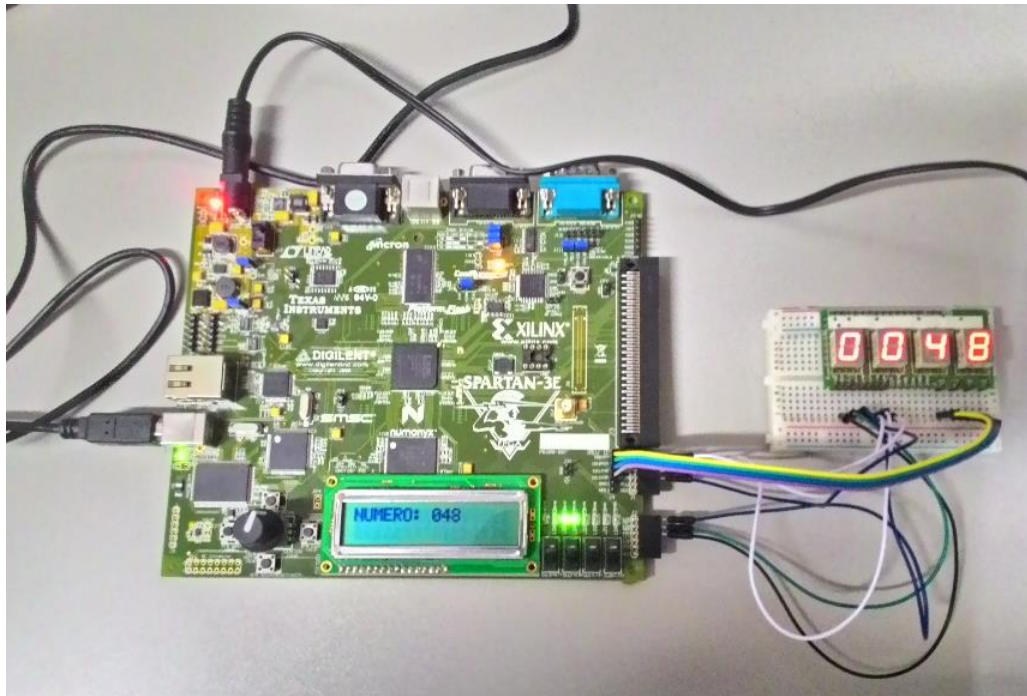


Figura 2.2 – Montagem em bancada de um GNPA usando FPGA.

Fonte: Autor

2.2 Geradores Físicos de Números Aleatórios

Geradores Físicos de Números aleatórios são aqueles que usam algum princípio físico como base para geração das sequências e, em alguns casos, podem ser considerados aleatórios. Para isso, é necessário a adição de um *hardware* ao processador principal ou, ainda, a utilização de algum periférico anexo ao computador para extrair tal efeito. É desejável que o efeito físico que serve como entrada nesse sistema deve possuir uma Entropia não determinística, caso contrário será um gerador pseudoaleatório.

Dessa forma, tais geradores podem produzir sequências imprevisíveis e, de fato, aleatórias, entretanto seu resultado deve ser cuidadosamente analisado antes de seguir para as aplicações. Isso ocorre porque a distribuição nem sempre é uniforme e os fenômenos físicos naturais podem gerar bits tendenciosos ou correlacionados. Daí a importância da realização de testes para certificação e validação desses bits, ainda que, tenham sido gerados por fenômenos físico conhecidos (SOARES, 2013).

Nesse contexto, a informação aleatória dado como entrada ao processador deve ser usada conforme algumas exigências da aplicação, assim como no gerador PRNG. Além do cuidado com a análise dos bits gerados, o desgaste dos elementos, em geral semicondutores ou optoeletrônicos, que compõe o circuito de geração do sinal e aquisição

dos dados também merece uma atenção especial, pois interferem diretamente na geração da informação (SOARES, 2013).

2.2.1 Efeitos Físicos em Semicondutores

Os efeitos físicos extraídos para os geradores normalmente são feitos com base nos efeitos de tensão e corrente em semicondutores ou em materiais optoeletrônicos. Neste trabalho, um dos focos será a análise de um desses efeitos em materiais semicondutores. Tais elementos têm como fator de escolha tanto o seu custo de aquisição quanto a aplicação na qual será destinada as sequências a serem geradas.

Os semicondutores e optoeletrônicos mais usados neste cenário são:

- a. diodo Zener;
- b. transistor Bipolar;
- c. diodo Túnel;
- d. foto receptores do tipo CMOS;
- e. fotodiodo de Avalanche;

Em “a”, efeito a ser explorado é o Efeito Zener e em “b” é explorado o Efeito Avalanche. Em “c”, o efeito de tunelamento quântico e em “d”, a captura de fótons e sua conversão em sinais eletromagnéticos é usado como fonte de entropia. Por último, em “e” o ruído de ionização de impacto na formação da avalanche com fator de ganho é o fenômeno a ser explorado. É importante ressaltar que os princípios de “b” e “e” são os mesmos, porém no componente óptico fenômeno é bem mais exacerbado.

É possível também explorar os materiais Fotônicos como fonte de Entropia não determinística. Esses elementos são cristais feitos com componentes dopados e sua interação com a luz faz com que eles possam produzir efeitos não determinísticos. Todavia, seu uso fica restrito a algumas aplicações devido ao seu alto custo e destinado aqueles com domínio e matéria prima para fabricação desses materiais.

Para fins de análise desses efeitos, parte-se sempre da construção de um semicondutor, portanto, como atua as junções do tipo “p” e do tipo “n” quando estão direta ou reversamente polarizadas. Assim, as junções em seu estado natural buscam o Equilíbrio Térmico por meio do que é conhecido como barreira de potencial, ou camada de depleção, onde a largura dela é dada pelo equilíbrio entre a corrente de difusão e a

corrente de deriva. Na Figura 2.3 é mostrado um Diodo Semicondutor para ilustrar esses conceitos.

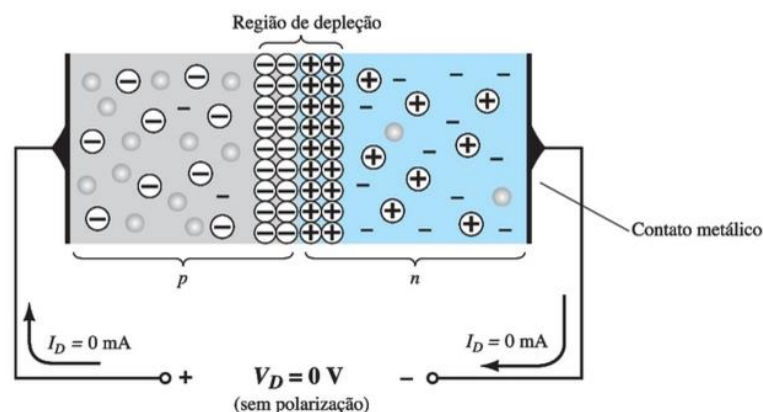


Figura 2.3 – Diodo semicondutor formado pelas junções do tipo “n” e tipo “p” [4].

Criado um semicondutor baseado nas junções “p” e “n” é possível polarizá-la de duas formas: uma direta e uma reversa (ver Figura 2.4). Na polarização direta, o potencial elétrico na junção “p” cresce em relação a junção “n”, fazendo com que a corrente dos portadores majoritários de carga que flui ao longo do elemento cresça e a corrente dos outros portadores de carga, minoritários, permaneça a mesma do equilíbrio térmico. Já na polarização reversa, o potencial da junção “p” é menor que na junção “n”, logo há um crescimento da barreira da camada de depleção, resultando numa corrente negligenciável (geralmente na ordem de nA). Neste caso, a corrente de portadores minoritários é a principal fonte de corrente no semicondutor.

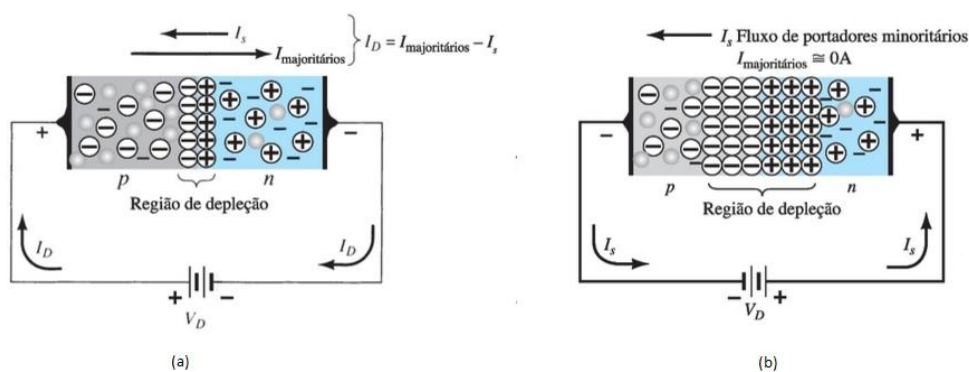


Figura 2.4 – Polarização de semicondutor formado por junção do tipo “p” e do tipo “n”. (a) Polarização Direta de um Diodo. (b) Polarização Reversa de um Diodo [4].

Devido a concentração de portadores de cargas do tipo “p” e do tipo “n”, é possível deduzir uma equação que leve em considerações tais efeitos. Na equação 2.1 é dado a concentração dos portadores no equilíbrio térmico:

$$\frac{P_p}{P_n} = \exp\left(\frac{qV}{kT}\right) \quad (2.1)$$

Em que P_p é concentração de portadores do tipo “p”, P_n é concentração de portadores do tipo “n”, q é quantidade de carga, V é a tensão térmica, k é a constante de Boltzmann e T é a temperatura absoluta.

A partir de (2.1), é possível, agora, fazer análise sobre os efeitos da polarização na elemento. Admitindo a polarização chega-se a equação (2.2), que nada mais é que a simples mudança de análise da quantidade de portadores e explicitar que eles sofrem efeito da distância, logo sua energia é reduzida a medida que se afastam da barreira de potencial.

$$\frac{P(-x_{p0})}{P(x_{n0})} = \exp\left(\frac{q(V_0 - V)}{kT}\right) \quad (2.2)$$

Em que x_{p0} e x_{n0} são os referenciais em relação a camada de depleção.

Lembrando que na polarização direta, tem-se baixo nível de injeção de elétrons, então é possível dizer que $P(-x_{p0})$ é igual a P_p e portanto chega-se a equação (2.3).

$$P(x_{n0}) = P_p \times \exp\left(\frac{qV}{kT}\right) \quad (2.3)$$

O resultado da equação 2.3 mostra que há um crescimento no número de lacunas na situação de equilíbrio térmico. Assim, há o avanço de lacunas na região “n” e excesso de elétrons na região “p”. O excesso de concentração das lacunas na borda da região “n” da junção “pn” é dado pela equação (2.4).

$$\Delta P_n = P(x_{n0}) - P_n = P_n \times \left(\exp\left(\frac{qV}{kT}\right) - 1\right) \quad (2.4)$$

Para fins de simplificação, na análise das concentrações há mudanças das coordenadas para análise, visando uma situação quase eletricamente neutra. E as equações (2.5) e (2.6) mostra esse novo foco de análise:

$$\delta P(x_n) = \Delta P_n \times \exp\left(\frac{-x_n}{L_p}\right) \quad (2.5)$$

$$\delta P(x_p) = \Delta n_p \times \exp\left(\frac{-x_p}{L_n}\right) \quad (2.6)$$

De posse deste resultado, agora é estimar a corrente de difusão na situação de quase neutralidade gerada pelo excesso de lacunas na região “n”. Essa distribuição é dado pela equação (2.7):

$$I_p(x_n) = -qAD_p \times \frac{d\delta P(x_n)}{d(x_n)} = qA\left(\frac{D_p}{L_p}\right) \times \delta P(x_n) \quad (2.7)$$

A corrente de difusão é máxima na região “n”, na situação de quase neutralidade, portanto x_n é zero. Logo, a equação (2.8) pode ser reescrita como:

$$I_p(x_n = 0) = qA\left(\frac{D_p}{L_p}\right) p_n \times (\exp(\frac{qV}{kT}) - 1) \quad (2.8)$$

De modo análogo, a corrente I_n na situação contrária, ou seja, na quase neutralidade com o excesso de elétrons fluindo para borda do lado “p” da junção “pn” é dado por:

$$I_n(x_p = 0) = qA\left(\frac{D_n}{L_n}\right) n_p \times (\exp(\frac{qV}{kT}) - 1) \quad (2.9)$$

Assumindo a recombinação entre os pares elétrons lacunas nesta região de troca de cargas na situação já citada, a corrente total é dada pela diferença entre o fluxo de corrente que flui das lacunas, e dos elétrons, nos seus respectivos sentidos naturais devido a polarização direta. Então a equação (2.10) expressa a corrente total que flui ao longo da junção “pn”, também conhecida como equação do Diodo.

$$I = qA\left(\frac{D_p}{L_p} p_n + \frac{D_n}{L_n} n_p\right) \times (\exp(\frac{qV}{kT}) - 1) = I_0 (\exp(\frac{qV}{kT}) - 1) \quad (2.10)$$

Já na polarização reversa o fluxo de corrente através da junção “pn”, chamado de corrente de saturação reversa, possui um fator muito maior que os fatores da exponencial (26 mV a temperatura ambiente), assim é dado pela equação (2.11):

$$I = -I_0 = -qA\left(\frac{D_p}{L_p} p_n + \frac{D_n}{L_n} n_p\right) \quad (2.11)$$

Isso ocorre na situação de polarização reversa, porque os portadores internos ao elemento semiconductor sofrem ação do campo elétrico de modo a deixar as regiões de borda com carga nula. Dentro de um comprimento de difusão conhecido os portadores são extraídos da região de depleção num fenômeno conhecido como extração de portadores minoritários.

Para ilustrar o funcionamento prático das equações acima citadas, a Figura 2.5 mostra a curva de um semiconductor. Doravante, o foco será a região de polarização reversa, porque há dois efeitos que podem romper a barreira de potencial na polarização reversa e que serão usados no gerador caracterizado neste trabalho, o Efeito Zener e o Efeito Avalanche (SIGNH, 2016).

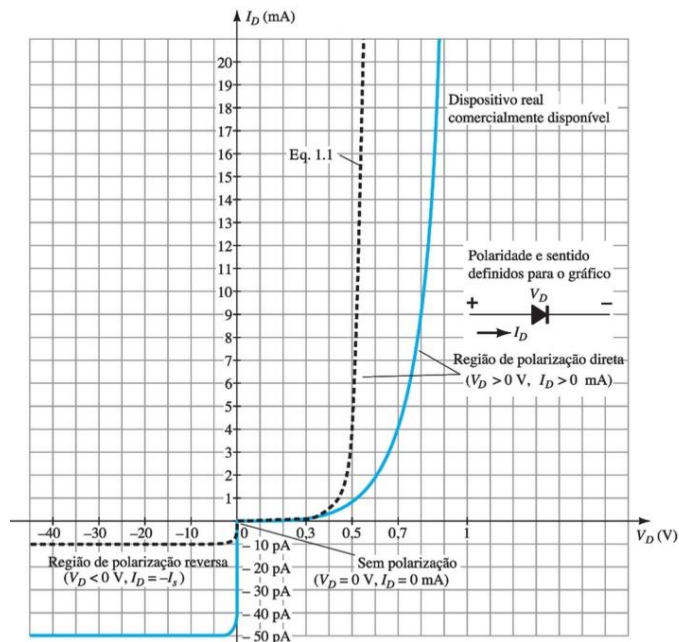


Figura 2.5 – Curva característica de um semiconductor de Silício [4].

A Figura 2.5 passa a impressão de que a tensão na polarização reversa pode crescer indefinidamente, contudo ela apenas representa a um modelo para ilustrar graficamente o comportamento matemático da equação do Diodo. Há um ponto em que essa aplicação de tensão, polarizando a junção “pn” reversamente, irá bruscamente mudar esta curva característica. Esse potencial que resulta esta mudança é conhecida como Tensão de Ruptura, *Breakdown Voltage* (V_{BV}). O surgimento desse fato novo pode ser visto na Figura 2.6 (BOYLESTAD & NASHELSKY, 2013).

Isso ocorre devido a velocidade dos portadores minoritários, que são responsáveis pela corrente de saturação reversa, também aumentar. Logo, a energia cinética média alcançará um nível de energia suficiente para aumentar o número de colisões entre as estruturas atômicas estáveis. Esse fenômeno é conhecido como ionização de impacto e, com isso, esse portadores irão produzir novas ionizações até o ponto de criar uma avalanche atingindo V_{BV} (BOYLESTAD & NASHELSKY, 2013).

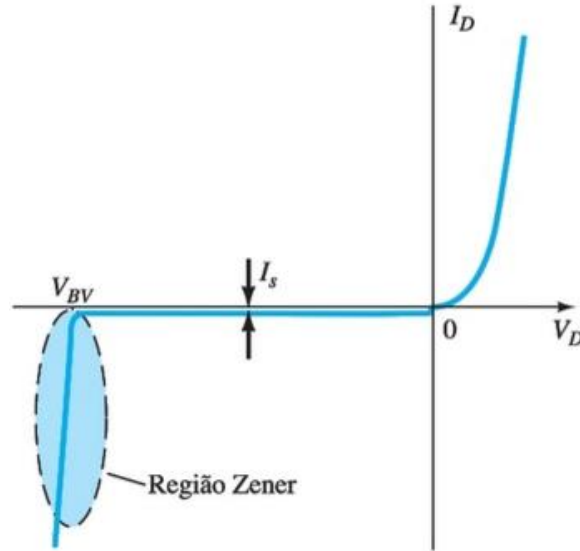


Figura 2.6 – Região de Ruptura de um semiconductor de Silício [4].

2.2.2 Efeitos Zener e Efeito Avalanche

Há dois efeitos que podem gerar a ruptura da barreira de potencial nas junções do tipo “pn”, o Efeito Zener e o Efeito Avalanche. Ambos estão relacionados com a dopagem do elemento semiconductor e fazem uso dessas características para criação de dois dispositivos bastante usados na eletrônica, o Diodo Zener e o Fotodiodo de Avalanche (SIGNH, 2016); (BOYLESTAD & NASHELSKY, 2013).

O Efeito Zener ocorre devido ao tunelamento quântico dos portadores de carga ao longo da junção. Tal efeito deve ter como base uma junção com elementos altamente dopados. A probabilidade de ocorrência do tunelamento pode ser calculada pela Equação 2.12 (ZEGHBROECK, 2016):

$$\Theta = \exp\left(-\frac{4\sqrt{2m^*}E_g^{\frac{3}{2}}}{3q\hbar\varepsilon}\right) \quad (2.12)$$

Em que m^* é a massa efetiva do elétron, q é a carga, $\hbar$ é a constante de Planck normalizada, E_g é a energia, ε é o campo elétrico.

A corrente de tunelamento é obtida através do produto entre os portadores de carga, a velocidade e a densidade dos portadores. A velocidade é obtida pela equação de Richardson, que é a velocidade média com que os portadores se aproximam da barreira de potencial, quanto a densidade dos portadores é igual a densidade de elétrons livres

multiplicados pela probabilidade deles tunelarem (ZEGHBROECK, 2016). Assim, chega-se a equação 2.13:

$$J_n = qvn\Theta \quad (2.13)$$

Já o Efeito Avalanche faz com que os portadores de carga sejam acelerados devido ao alto campo elétrico na região de depleção e, com isso, a colisão dos portadores principais com outras fontes de cargas, ionização de impacto, multiplicando tal efeito. Para a avalanche ocorrer, o semicondutor não precisa estar altamente dopado nas junção “pn” (SIGNH, 2016).

A ionização de impacto pode ser quantificada pelas constantes de ionização dos elétrons e das lacunas (α_n ou α_p) e essas constantes são definidas como a mudança na densidade dos portadores (n) com posição dividida pela densidade dos portadores (ver equação 2.14) (ZEGHBROECK, 2016).

$$d_n = \alpha_n \times n \times dx \quad (2.14)$$

Vale ressaltar que a ionização causa a adição de novos elétrons e lacunas no sistema. Assumindo que os coeficientes de ionização são os mesmo para elétrons e lacunas, essa multiplicação (M) pode ser expressa pela equação 2.15 (ZEGHBROECK, 2016).

$$M = (1 - \int_{x_1}^{x_2} \alpha dx)^{-1} \quad (2.15)$$

Note que a região compreendida pela integral, de x_1 até x_2 , é a camada interna a depleção, onde o campo elétrico é presumidamente constante e grande o suficiente para gerar a ionização de impacto. Já fora dessa região, o campo não deve ser capaz de produzir a ionização de impacto. Ainda, nesta equação é necessário o cuidado para que a integral não atinja o valor unitário, pois teremos uma divergência no fator M , portanto para cada elétron gerado pela ionização tem-se um par elétron-lacuna adicional para evitar essa condição de anormalidade físico-matemática (ZEGHBROECK, 2016).

Por fim, o fator M é comumente expressado, de maneira empírica, através da equação 2.16, como função da tensão aplicada (V_a) e a tensão de ruptura (V_{br}) (ZEGHBROECK, 2016).

$$M = (1 - |\frac{V_a}{V_{br}}|^n)^{-1}, \quad \forall \quad 2 < n < 6 \quad (2.16)$$

2.2.3 Diodo Túnel e Tunelamento Quântico

O Diodo Túnel ficou conhecido pela primeira vez através do experimento feito por Leo Esaki em um Laboratório da Sony Corporation em 1958. Esaki desenvolveu e constatou em laboratório uma das propriedades da Mecânica Quântica, fundamental deste elemento, ocorre devido ao efeito quântico de tunelamento dos elétrons através de uma junção “pn” fortemente dopada (LEÃO, 2009).

Em relação ao Diodo Zener, que também funciona baseado no tunelamento, o Diodo Túnel possui capacidade de operação na faixa de Giga Hertz e, geralmente, são usados em aplicações que exigem altas frequências, inclusive nas faixas de radiofrequência (RF), sobretudo como oscilador. Por isso, esse dispositivo é altamente atrativo na indústria e comunicação sem fio e de periféricos de alto desempenho.

O Diodo Túnel se diferencia de um diodo convencional, porque este dispositivo possui uma região de resistência negativa. Ora, esta condição implica o aumento da tensão com redução da corrente elétrica, servindo como uma situação rara em relação a primeira Lei de OHM. Na Figura 2.7, é mostrado essa região de atuação deste elemento. (BOYLESTAD & NASHELSKY, 2013).

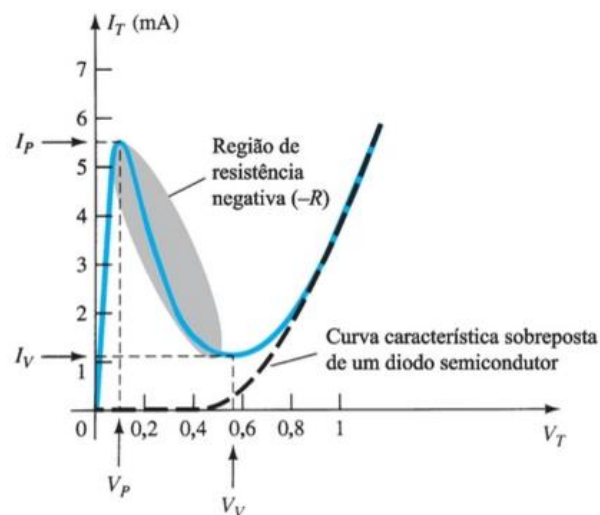


Figura 2.7 – Curva característica de um Diodo Túnel em comparação a um Diodo comum [4].

Na sua fabricação, eles são dopados intensamente por materiais do tipo “n” e do tipo “p” para formar a junção em um nível muito maior que em um diodo comum. Por isso, sua Camada de Depleção é muito reduzida em relação aos demais diodos. É essa camada fina que permite o “atravessar” em vez de forçar uma transposição em função da tensão de polarização reversa, ou até mesmo tensão de ruptura. Além disso, esse

“atravessar” é feito em velocidades muito grandes, o que ajuda a explicar sua atuação em altas frequências em relação aos demais diodos (BOYLESTAD & NASHELSKY, 2013).

Na Figura 2.7, é possível perceber que a região de resistência negativa é ditada por quatro parâmetros: I_p é a corrente de pico, I_v é a corrente de vale, V_p é a tensão de pico e V_v é a tensão de vale. A tensão de vale é, em geral, limitada em 600 mV, logo requer um cuidado com esses dispositivos, pois uma medição com equipamento inadequado irá danificar o dispositivo (BOYLESTAD & NASHELSKY, 2013).

Do ponto de vista da Física Quântica, dado uma partícula qualquer com massa e energia cinética que possa colidir com uma barreira de potencial de espessura finita e que sua energia seja menor que a energia da barreira e se a barreira foi fina, há uma probabilidade dessa partícula atravessar essa barreira, isto é, ser transmitida. Esse efeito é conhecido como Tunelamento Quântico (LEVI, 2006).

Isso ocorre de maneira análoga nas interações entre potenciais eletromagnéticos, contudo no eletromagnetismo clássico considera-se um grande número de partículas nas interações entre as ondas evanescentes, já no caso quântico apenas uma partícula. Assim, o tunelamento é um efeito puramente de natureza quântica. Para elétrons, a densidade de corrente de tunelamento depende exponencialmente da espessura da barreira, L , e da energia dela, V_0 , e, por isso, tende a ser desprezível quando a relação entre a energia da barreira e sua largura possuem um valor grande, Figura 2.8 (LEVI, 2006).

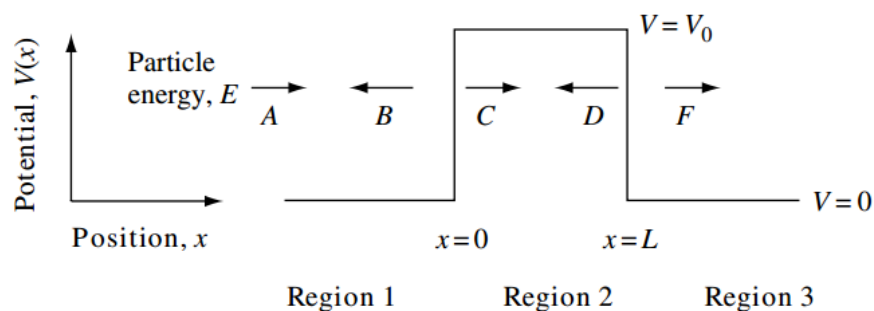


Figura 2.8 – Ilustração de um partícula incidente numa barreira de potencial retangular [21].

Na prática, esse efeito tem impacto no desempenho de uma série de dispositivos eletrônicos como os transistores do tipo CMOS e os transistores Bipolar de Junção (BJT). Em componentes CMOS, a corrente de fuga através da barreira do óxido contribui diretamente para imposição de um limite de redução e escala na miniaturização desse

componente. Em componentes do tipo BJT, pode-se usar esse efeito para melhorar o desempenho dos componentes (LEVI, 2006).

A solução para o poço finito é análoga aquela para o oscilador harmônico, logo a resposta é uma função ressonante. Neste caso, a ressonância devido ao tunelamento ocorre apenas na transmissão e esse fato é explicado pela redução da contribuição exponencial no espalhamento da função de onda ao longo da barreira de potencial. Na Figura 2.9, é proposto um cenário que ajuda a compreender tal efeito e o mesmo ocorre no Diodo Túnel (LEVI, 2006).

Assim como no poço de potencial, Figura 2.8, há sempre uma ressonância simétrica de menor energia associada com o potencial do poço. Para este caso, tal ressonância tem energia menor, ver o tracejado na Figura 2.9 (a), e a função de onda associada é simétrica, porque o potencial é simétrico. Devido a largura finita e a energia da barreira de potencial em ambos os lados do poço, há uma probabilidade finita de que uma partícula pode sofrer tunelamento. Essa probabilidade é ilustrada na Figura 2.9 (b), onde é observado inclusive um pico de transmissão, quando há ressonância (LEVI, 2006).

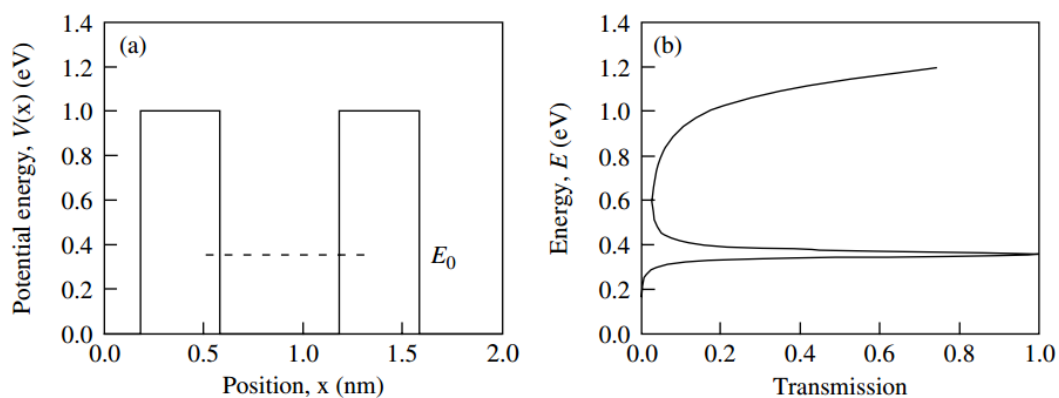


Figura 2.9 – Análise da Barreira de Potencial e probabilidade de Transmissão [21].

Hoje, há técnicas que permitem replicar esse fenômeno ressonante em cristais para formação de elementos como o Diodo Túnel, efeito na Figura 2.10. Espera-se que as características de tensão e corrente na heterojunção deva ser proporcional ao coeficiente de transmissão de elétrons via tunelamento. Enquanto esse estágio não é alcançado, há uma região de resistência negativa de interesse em aplicações eletrônicas (LEVI, 2006).

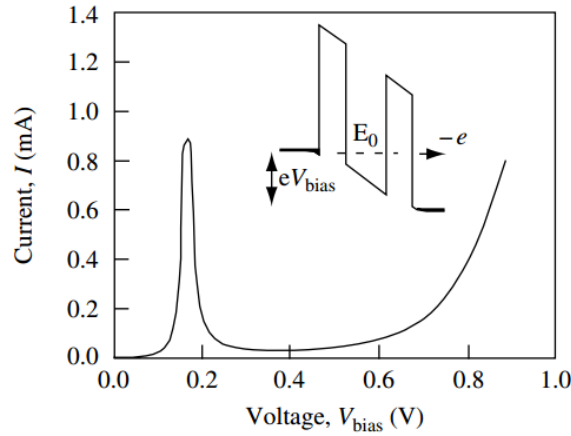


Figura 2.10 – Análise da Barreira de Potencial para um Diodo Túnel [21].

2.3 Geradores Quânticos de Números Aleatórios

Um Gerador Quântico de Números Aleatórios (GQNA) é aquele que faz uso dos conceitos da Mecânica Quântica para gerar a sequência de bits aleatórios. Isso ocorre, porque ao contrário da Física Clássica, a Física Quântica tem natureza totalmente estatística. Tal característica permite que dispositivos sejam criados a partir do uso dessas propriedades físicas, logo é possível ter um processo que é verdadeiramente aleatório.

Um simples fóton, um quantum de luz, pode ser usado como portador de informação, um *Qubit*, contudo outras partículas podem ser usadas para tal, não somente aquelas elementares. Por ser quântico, o *Qubit* é matematicamente representado como a combinação linear, vide equações 2.17 e 2.18, dos estados Quânticos “0” e “1” (STIPČEVIĆ, 2011):

$$\psi = \alpha|0\rangle + \beta|1\rangle \quad (2.17)$$

$$\alpha^2 + \beta^2 = 1 \quad (2.18)$$

Há outras grandezas, elementares ou não, que podem se encaixar no cenário descrito acima, todavia todas elas sofrerão o fenômeno conhecido como Efeito do Observador e é ele que garante a geração verdadeiramente aleatória. O estado quântico de uma partícula qualquer é especificado por uma direção no espaço de Hilbert, logo é um vetor unitário ($|\psi\rangle$) na direção que representa este estado (LEVI, 2006).

As grandezas observáveis são representadas em função de seus operadores hermitianos, Posição ($|\mathbf{x}\rangle$) e Momentum ($|\mathbf{p}\rangle$), onde sua evolução temporal é dada pela

derivada do vetor de estado pelo tempo ($\frac{\partial|\Psi\rangle}{\partial t}$). Na base $\{|\mathbf{x}\rangle\}$, valem as considerações da equação 2.19:

$$\mathbf{x} = x \text{ e } \mathbf{p} = -i\hbar \frac{\partial}{\partial x} \quad (2.19)$$

Logo, o Hamiltoniano, que representa a energia mecânica (soma da energia cinética com a energia potencial) do sistema é dado pela equação 2.20:

$$H = \frac{p^2}{2m} + V(x) = -\frac{\hbar}{2m} \frac{\partial^2}{\partial x^2} + V(x) \quad (2.20)$$

Agora que tem-se a expressão do Hamiltoniano, aplica-se os operadores da Mecânica Quântica, que agem sobre as funções de onda em uma dimensão, que é (2.21):

$$H|\psi\rangle = E|\psi\rangle \quad (2.21)$$

Note que E é um autovalor de H , e que pertence ao conjunto dos números Reais, e que $|\psi\rangle$ é um auto estado de $\mathbf{H}$, e que H é a energia deste estado. Assim, a equação 2.22, mostra a evolução temporal do estado (2.23):

$$|\psi(t)\rangle = \exp\left(\frac{-iHt}{\hbar}\right)|\psi(0)\rangle \quad (2.22)$$

$$\frac{\partial|\psi(t)\rangle}{\partial t} = \frac{-iH}{\hbar}|\psi(t)\rangle \leftrightarrow i\hbar \frac{\partial|\psi(t)\rangle}{\partial t} = H|\psi(t)\rangle \quad (2.23)$$

Assim, chega-se a expressão equivalente as segunda lei de Newton na Mecânica Quântica e a uma expressão que a Equação de Schrödinger dependente do tempo. Para finalizar essa abordagem, projeta-se o Hamiltoniano ($\mathbf{H}$) na base $\{|\mathbf{x}\rangle\}$ e obtem-se a equação 2.24:

$$\langle x|H|\psi\rangle = \langle x|\frac{i\hbar\partial}{\partial t}|\psi\rangle = i\hbar \frac{\partial\psi(x,t)}{\partial t} = -\frac{\hbar}{2m} \frac{\partial^2\psi(x,t)}{\partial x^2} + V(x)\psi(x,t) \quad (2.24)$$

Para interpretar a informação contida em $\psi(x,t)$, é introduzida a interpretação estatística de Max Born, equação 2.25, com o conceito de densidade de probabilidade. Pois assim, será possível calcular o quão provável é encontrar a partícula com valor “M” no intervalo entre o momento atual e a sua evolução temporal.

$$\text{Densidade de Probabilidade} = |\langle M|\psi\rangle|^2 \quad (2.25)$$

Como $\psi(x,t)$ é dado pelo $\langle \mathbf{x} | \psi \rangle$, a analogia com a equação de Born é válida desde que $\mathbf{M}$ seja considerado como $\mathbf{x}$. Se essa condição for satisfeita, então $|\psi(x,t)|^2$ fornece uma probabilidade de encontrar a partícula ao longo da sua evolução, Figura 2.11. A normalização desse valor é dado pela equação 2.26:

$$\int_{-\infty}^{+\infty} |\psi(x,t)|^2 dx = \int_{-\infty}^{+\infty} \psi^*(x,t) \times \psi(x,t) dx = 1 \quad (2.26)$$

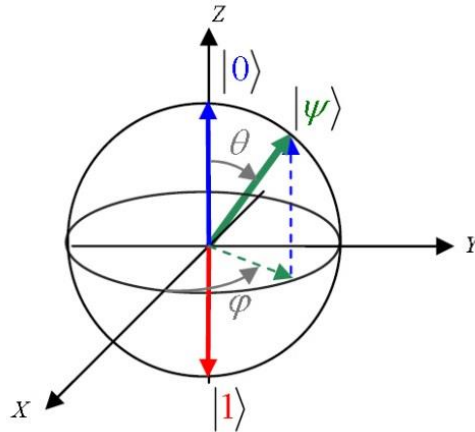


Figura 2.11 – Representação da evolução temporal de uma partícula quântica sobre o efeito do observador por meio da Esfera de Bloch.

Fonte: adaptado de https://upload.wikimedia.org/wikipedia/commons/e/e9/Sphere_bloch.jpg.

Por fim, essa série de equações nos conduzem ao princípio da incerteza de Heisenberg. Agora, trabalhando com um espaço de Hilbert e seu conjugado, o valor médio da posição e do momentum são dados pelas equações 2.27 e 2.28 e o princípio da incerteza pela equação 2.29.

$$\sigma_x = \sqrt{\langle x^2 \rangle - \langle x \rangle^2} \quad (2.27)$$

$$\sigma_p = \sqrt{\langle p^2 \rangle - \langle p \rangle^2} \quad (2.28)$$

$$\sigma_x \times \sigma_p \geq \hbar/2 \quad (2.29)$$

Aplicando essa teoria em alguns cenários de laboratório, pode-se chegar a um GQNA. Na literatura, o descrito na Figura 2.12. Considere uma feixe de luz circularmente polarizado, logo tem-se uma combinação de uma componente horizontal com uma vertical. Em seguida, tal feixe tem como obstáculo um divisor de feixe 3 dB, ou 50%, isso fará com que o feixe tenha metade da potência linear vertical para uma saída e a outra metade horizontal para a outra saída (STIPČEVIĆ, 2011).

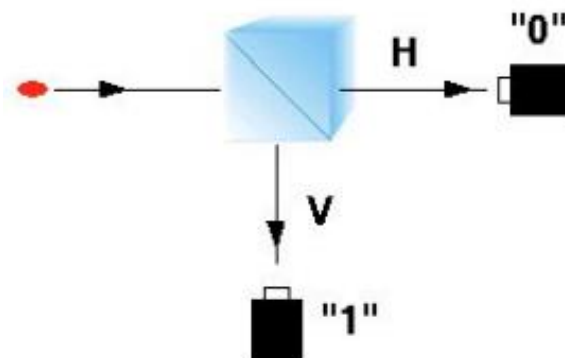


Figura 2.12 – Experimento para validar a superposição dos estados quânticos [40].

Com isso, fótons terão igual probabilidade, de ir por qualquer uma das portas, na figura rotuladas de Porta “0” e Porta “1”, gerando neste caso um cenário de incerteza total, portanto do ponto de vista teórico tem-se um GQNA perfeito com a garantia quântica de aleatoriedade. Na prática, geradores baseados neste ou em outros princípios podem encontrar algumas dificuldade devido aos componentes e sua fabricação, como: imperfeições, impurezas, desalinhamento de feixes no divisor, efeito de memória no fotodetector, etc. Apesar de tudo isso, qualquer que seja grandeza medida independente do processo gerador de bits, pode-se estimar com precisão (STIPČEVIĆ, 2011).

Em síntese, o diferencial de um GQNA é sua segurança na geração de efeitos verdadeiramente aleatórios. Os ensaios práticos chegaram perto o suficiente do referencial teórico e permite uma avaliação independente das imperfeições que podem ser introduzidas pelos materiais usadas com ajuda de um pós processamento. A maior desvantagem destas soluções atualmente são os componentes, que são bastante específicos e caros. Como perspectiva, as soluções baseadas em chips ópticos em crescente evolução hoje tornam este cenário cada vez mais atrativo (STIPČEVIĆ, 2011).

2.4 Testes Estatísticos

As sequências obtidas a partir de geradores de números aleatório devem passar por um processo de validação estatística para atestar tal característica. Não há um limite para o números de testes que possam ser aplicados nas sequências, entretanto à medida em que se aplica mais teste, mais confiável é a sequência. Os testes são divididos em Testes Empíricos e Testes Teóricos (KNUTH, 1981).

Os Teste Teóricos são aqueles que envolvem cálculos matemáticos para as propriedades estabelecidas para o teste. São específicos para cada classe de geradores e

se concentram na estrutura intrínseca de cada gerador para derivar características de comportamento da sequência. Enfim, atestam a qualidade de um gerador de acordo com algum critério matemático sem exigir geração de números. Testes Empíricos têm foco no resultados do gerador e não com a origem dos números, portanto testam se as propriedades coerentes estão probabilisticamente dentro do esperado para uma distribuição aleatória (SOARES, 2013).

Em uma análise de uma sequência de números aleatórios por meio de um teste é muito importante saber como os dados serão testados. É necessário destacar também que os testes devem ser executados em sequências distintas e não sobre as mesmas sequências para garantir a independência dos resultados. Lembrando que para cada teste há uma recomendação sobre o tamanho da sequência e para cada nível de aprovação há um critério, dependendo no nível de significância (SOARES, 2013).

Ao longo da implementação dos testes, é possível ocorrer algumas falhas, em geral elas são proporcionais ou unidirecionais. As falhas proporcionais são aquelas que ocorrem quando um gerador falha num teste com muita frequência e as unidirecionais são aquelas que ocorrem quando um gerador falha de forma inconsistente. Note que um gerador pode alternar momentos nos resultados destes testes (KENNY, 2012).

Neste trabalho há uma fonte de entropia como entrada e, com isso, a entrada, ainda que seja sempre conhecida, não deve ser possível prever o próximo valor a ser gerado, logo tal gerador deve passar nos testes para uniformidade e independência da sequência produzida e o tempo deve ser otimizado visando eficiência (L'ECUYER, 1996).

O NIST, órgão que este trabalho se baseia para realização dos testes, lista algumas possíveis explicações para detalhar os motivos de falhas de um conjunto de dados em algum teste estatístico, são elas (NIST, 2010):

- teste incorretamente implementado;
- teste subdesenvolvido, isto é, quando não há maturidade tanto na teoria probabilística quanto a complexidade não está suficientemente desenvolvida para facilitar uma análise rigorosa do teste;
- implementação inadequada no RNG, ou seja, quando o *Hardware* ou o *Software* deste dispositivo falha devido a um defeito no projeto ou erro na codificação;

- escolha incorreta dos parâmetros de entrada: parâmetros como tamanho da sequência, tamanho da amostra, tamanho do bloco e modelo.

Há quinze testes que compõe o pacote do NIST, que foram desenvolvidos para testar a aleatoriedade de sequências binárias produzidas por *Hardwares* ou *Softwares*. A Tabela 2.1 mostra um resumo desses testes e o parâmetro que ele avalia (NIST, 2010).

Tabela 2.1: Resumo dos Teste Estatísticos propostos pelo NIST.

Fonte: NIST, 2010.

Teste Estatístico	Parâmetro Avaliado
Frequência (Monobit)	Zeros e uns igualmente distribuídos
Frequência de um Bloco	Uns distribuídos ao longo de um bloco
Teste de Corrida	Testa total de números em uma sequência
Maior Corrida de uns dentro de um Bloco	Tamanho da cadeia de uns dentro de um bloco
Posto da Matriz Binária	Testa dependência linear entre substrings de tamanho fixo na sequência
Espectral da Transformada discreta de Fourier	Padrões Repetitivos
Casamento de padrão sem superposição	Ocorrências irregulares de um modelo pré especificado
Casamento de padrão com superposição	Ocorrências irregulares de um modelo pré especificado
Estatística universal de Maurer	Incompressibilidade da sequência
Complexidade Linear	Tamanho do registrador de deslocamento de realimentação linear
Serial	Não uniformidade na distribuição conjunta para sequências de comprimento m
Entropia Aproximada	Não uniformidade na distribuição conjunta para sequências de comprimento m
Somas Cumulativas	Muitos zeros e uns igualmente em fase precoce ou tardia da sequência
Excursões Aleatórias	Desvio da distribuição do número de visitas de um passeio aleatório para um determinado estado
Variantes de excursão aleatória	Desvio da distribuição do número de visitas (em muitos passeios aleatórios) para um determinado estado

Além do NIST, há outros dois pacotes de testes bastante usados nos trabalhos sobre geradores de números aleatórios, o ENT e o DIEHARD. O ENT é um programa que aplica vários testes um arquivo que contém uma sequência de bytes armazenados e exibe tais resultados. Esse programa é usado para avaliar geradores de números pseudoaleatórios para encriptação, amostragem estatística, algoritmos de compressão e outras aplicações onde a densidade de informação do arquivo é de interesse. Os testes feitos pelo ENT são: Entropia, Qui-Quadrado, Média Aritmética, Monte Carlo e Coeficiente de Correlação Serial (WALKER, 2016).

O outro pacote de testes usado é o DIEHARD. Ele é uma biblioteca científica *open source* (GSL) nativa em linguagem C e operado por linha de comando para execução dos

testes. Um outro objetivo é informar o motivo pelo qual ocorreu uma falha no gerador em um dos testes, onde tais informações possam ser extraídas durante o processo de teste e colocados em forma utilizável. Assim, esse projeto se propõe a funcionar como um pacote multiplataforma genérico de testes, inclusive integrado os testes do NIST nesta sua biblioteca (BROWN, 2016).

2.5 Plataformas Computacionais Embarcadas

Até os anos 1980, processamento de informações estava praticamente restrito aos computadores das grandes empresas, os *mainframes*. Já na década de 90, tal importância foi transferida para os computadores pessoais, PCs. Isso mostrava uma nova tendência, que era o desaparecimento do computador, posto que eles iriam reduzir muito tamanho, bem como os *softwares* estariam em todos os dispositivos tecnologicamente possíveis, dando, assim, os primeiros passos da computação ubíqua, computação pervasiva e ambientes inteligentes. Nesse período, também houve significativo avanço no setor de Telecomunicações, culminando na Convergência Digital (MARWEDEL, 2006).

Esse cenário trouxe uma nova tendência chamada computação em todo lugar, ou computação ubíqua, computação pervasiva e ambientes inteligentes. Assim, os sistemas embarcados surgem como forma de processar as informações de sistemas de sistemas que já operam em uma grande variedade de produtos e que, em geral, não são diretamente visíveis aos usuários. Pode-se citar como exemplos de sistemas embarcados, equipamentos de telecomunicações, computadores de bordo de sistemas de transporte, equipamentos de automação em fábricas, computadores de eletrodomésticos e em eletrônicos de consumo em geral, etc. (MARWEDEL, 2006).

Esses componentes são frequentemente associados a sensores e atuadores, as vezes são chamados de Computação Física, para controlar o ambiente onde estão inseridos. Também são conhecidos pelas suas características de sistemas dependes, pois necessitam atender os seguintes aspectos: confiabilidade, manutenibilidade, disponibilidade, segurança física e segurança lógica. Além disso, eles devem atender as seguintes premissas: eficiência energética, tamanho adequado do código fonte, operação de tempo real eficiente, tamanho, peso e custo reduzido (MARWEDEL, 2006).

Uma outra abordagem dada aos Sistemas Embarcados é a que possuem *hardware* fixo e componentes programáveis, *firmware*. Por exemplo, os Microcontroladores

presentes a um bom tempo nos eletrodomésticos. Porém, há também os Processadores Digitais de Sinais (DSPs), que são elementos de uso exclusivo e robustos para processamento de sinal. Por fim, há, ainda, os que possuem *hardware* reconfigurável, os CLPD (*Complex Programmable Logic Device*) e FPGA (*Field Programmable Gate Array*), esses são implementados por meio de uma linguagem de descrição de *hardware* (HDL) (OLIVEIRA & ANDRADE, 2006).

Nesse trabalho foram usadas três plataformas para desenvolvimento de Sistemas Embarcados que serão descritas a seguir: Arduíno UNO, Arduíno DUE e um FPGA Spartan 3E. Os Arduínos possuem arquiteturas diferentes e tem sua computação fornecida pelo fabricante. Já o FPGA possui um arquitetura reconfigurável, ou seja, fornece a capacidade de criar o próprio circuito lógico digital do modelo proposto.

O Arduíno é uma plataforma de desenvolvimento que tem um *hardware* fácil de usar e possui um ambiente de criação de *software* livre, que permite ao usuário criar programas que podem interagir com o ambiente físico. Por ser de fácil entendimento e possuir várias versões, vide Tabela 2.2, é muito útil para estudantes com pouco conhecimento em eletrônica e em linguagens de programação que se encontram em fase de aprendizado na área tecnológica. Além disso, é usado, também, no mercado em diversas áreas e em vários contextos, como automação e controle, telemetria ou em sistemas automobilísticos (MARGOLIS, 2011).

Tabela 2.2: Principais placas da plataforma de desenvolvimento Arduíno.

Fonte: OLIVEIRA, K. D. Desenvolvimento de Sistema Embarcado para Contagem de Fluxo de Pessoas em Atração de Parque Aquático Cearense. Monografia. p. 14.

Nome	Processador	Clock	E/S Digitais	Pinos Analógicas	EEPROM (KB)	SRAM (KB)	FLASH (KB)
DUE	AT91SAM3X8E	84MHz	54	14	-	96	512
Leonardo	ATmega32U4	16MHz	20	12	1	2,5	32
Mega2560	ATmega2560	16MHz	54	16	4	8	256
MegaADK	ATmega2560	16MHz	54	16	4	8	256
UNO	ATmega328P	16MHz	14	6	1	2	32

De acordo com Noble, a palavra Arduíno refere-se a três ferramentas separadas, que unidas formam um kit de ferramentas chamado de Arduíno. Primeiro, há o microcontrolador Arduíno, disponível em várias formas a partir de 12 dólares. Em segundo lugar, há a linguagem e o compilador, que criam o código para este

microcontrolador, e que, muito parecido com a linguagem Processing, simplifica muitas das tarefas que desafiavam os *designers* e desenvolvedores ao trabalhar com *hardware* e interação física. Finalmente, há o ambiente de programação Arduino, que, mais uma vez como o IDE *Processing*, é um IDE de código aberto simples construído em Java (NOBLE, 2009).

Já um FPGA possui quatro arquiteturas internas utilizadas em circuitos lógico programáveis: Matriz Simétrica, *Sea-of-Gates*, *Row-Based* e PLD Hierárquico. Os circuitos são compostos por quatro entidades, os CLBs, Switch Boxes, IOBs e canais de roteamento. A Figura 2.13 mostra as arquiteturas e a Figura 2.14 mostra as entidades internas do FPGA (ORDONEZ *et al*, 2006).

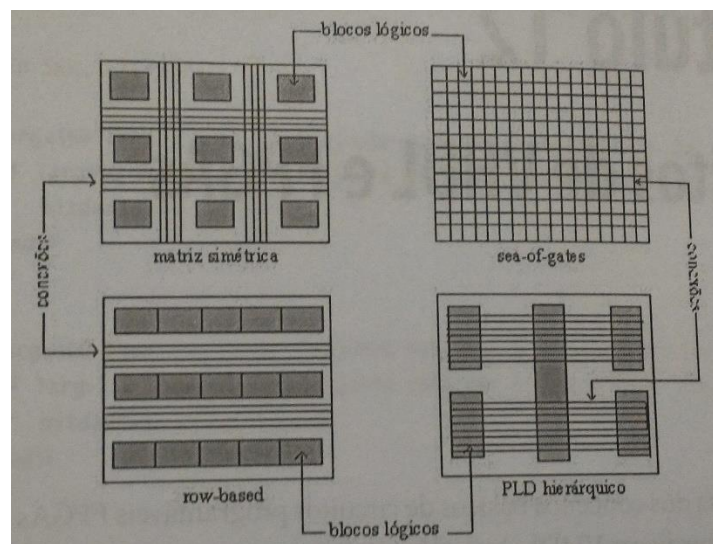


Figura 2.13 – Arquitetura internas do FPGA: Matriz Simétrica, *Sea-of-Gates*, *Row-Based* e PLD Hierárquico [32].

A arquitetura *Sea-of-Gates* é um circuito composto por transistores ou blocos de baixa complexidade. A sua vantagem é a grande disponibilidade de portas lógicas por área, porém seu roteamento é feito na áreas sobre as células o que pode inutilizar algumas partes. Na arquitetura *Row-Based*, os blocos lógicos estão dispostos horizontalmente e o roteamento é feito em uma área dedica para tal. Essas duas arquiteturas originaram-se das metodologias dos projetos de ASICs (ORDONEZ *et al*, 2006).

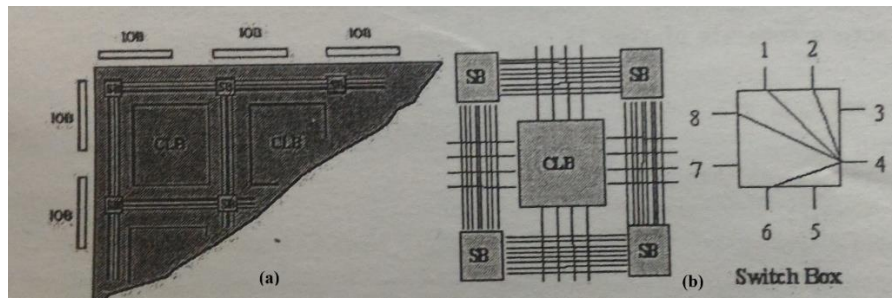


Figura 2.14 – Dispositivos internos de um FPGA. (a) CLB, IOB E SB. (b) Funcionalidades do SB [32].

A arquitetura PLD hierárquico é constituída por uma matriz de blocos lógicos, os *logic arrays blocks*, que são interligados por um recurso de roteamento, a matriz programável de interconexão (PIA). A hierarquia vem devido ao fato dos blocos lógicos serem agrupados entre si. E a matriz Simétrica é a mais flexível, pois possui canais verticais e horizontais para roteamento (ORDONEZ *et al*, 2006).

O roteamento é a interconexão entre os blocos por meio de uma rede de camadas metálicas. As conexões físicas entre os blocos lógicos são feitas por transistores controlador por bits de memória ou por chaves de interconexão (*switch matrix*) (ORDONEZ *et al*, 2006).

Em paralelo a isso, as aplicações atuais exigem um grande desempenho dessas arquiteturas e na busca por essas melhorias difundiu-se o processo de reconfiguração. Além disso, é possível acoplar um FPGA a um processador de propósito geral, tornando possível a exploração eficiente do potencial dessa característica. Isso fica bastante evidente, pois é possível dispensar unidade funcionais encontradas nesse processadores de propósito geral, focando o FPGA apenas no que aplicação exige de especificidade, reduzindo o número de ciclos necessário para execução (ORDONEZ *et al*, 2006).

Os métodos de reconfiguração mais comuns nesses dispositivos são classificados como (ORDONEZ *et al*, 2006):

- **Reconfiguração Total:** todo dispositivo é reconfigurado;
- **Reconfiguração Parcial:** apenas uma parte do dispositivo é reconfigurada;
- **Reconfiguração Dinâmica:** reconfiguração em tempo de execução;
- **Reconfiguração Extrínseca:** reconfigura parcialmente o sistema, mas apenas considerando um FPGA como parte atômica da reconfiguração;

- **Reconfiguração Intrínseca:** reconfigura parcialmente cada FPGA que compõe o sistema.

2.5.1 Arduíno UNO

O Arduíno UNO, Figura 2.15, é a última versão, na terceira revisão, da série de placas Arduíno, os Arduínos Duemilanove e Diecimila. Ele possui o Microcontrolador Atmega328P (máquina RISC de 8 bits), que dá suporte aos recursos disponíveis descritos a seguir e possui um bootloader que em conjunto com o IDE facilita todo o processo de criação dos Sistemas Embarcados com essa plataforma (MONK, 2013); (LIMA & VILLAÇA, 2012).

O Arduíno UNO, baseado na arquitetura do seu microcontrolador, possui 14 pinos de entrada/saída digitais (onde 6 deles podem ser usados como PWM), 6 entradas analógicas, *clock* de 16 MHz, conexão USB, alimentação via jack, botão de reset e conexão ICSP. Assim, ele pode ser alimentado tanto pela USB quanto por uma fonte DC ou bateria. Ideal para iniciantes, pois suas peças podem ser facilmente substituídas (ARDUÍNO, 2016).



Figura 2.15 – Plataforma de Desenvolvimento Arduíno UNO Revisão 3.

Fonte: adaptado de Arduino.cc.

2.5.2 Arduíno Due

O Arduíno DUE, Figura 2.16, é uma placa baseada no Microcontrolador Atmel SAM3X8E ARM Cortex-M3 CPU e é o primeiro Arduíno de 32 bits e baseado na Arquitetura ARM (*Advanced RISC Machine*). Ele possui 54 pinos de entrada e saída digitais, onde doze podem ser usados como PWM (*Pulse Width Modulation*), doze entradas analógicas, 4 interfaces de comunicação Serial, *clock* de 84 MHz, dois conversores Digital-Analógico, duas interfaces de comunicação I²C, interface SPI e

JTAG. Também possui um botão de reset e um botão para apagar a memória. Seu funcionamento é na tensão nominal de 3,3V em corrente contínua (ARDUÍNO, 2016).

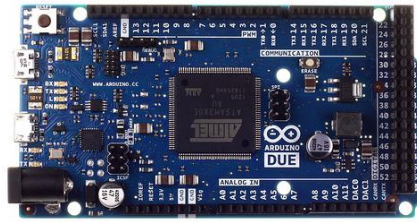


Figura 2.16 – Plataforma de Desenvolvimento Arduino DUE.

Fonte: adaptado de Arduino.cc

2.5.3 FPGA Xilinx Spartan 3E

O kit Spartan 3E possui vários periféricos de comunicação, PS/2, comunicação serial UART via DB9, macho e fêmea, por meio do protocolo RS232 (EIA232), serial USB e uma porta *Fast Ethernet* (IEEE 802.3). Para comunicações sem fio o kit oferece suporte por meio de conectores SMA, bem como conectores de alta velocidade, PMOD e VMOD. Computacionalmente, a plataforma possui um *clock* de 50MHz, memória flash de 16 MB, 500 mil portas, Memória RAM de 16 MB e até 92 I/O (XILINX, 2015).

Além disso, há interfaces Homem-Máquina presentes no kit são: oito leds, quatro chaves sem trava, quatro com trava, um potenciômetro e um display LCD 16x2 ligado a 4 bits e uma saída DB15, para comunicação de vídeo via VGA e padrão RGB. Ressaltando que todos os I/O podem ser acessados para conexão com periféricos externos, respeitando a alimentação de 3V3. A plataforma possui ainda canais para conversão analógico-digital, mas não foram usados neste trabalho (XILINX, 2015).

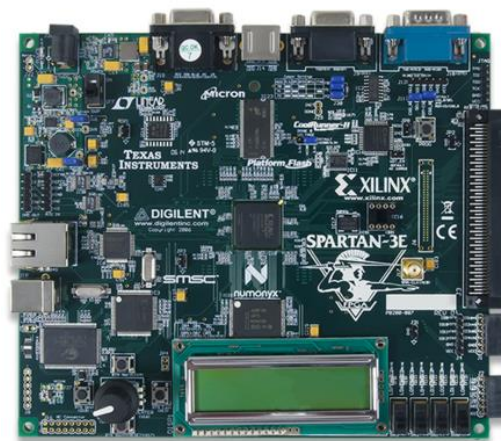


Figura 2.17 – Plataforma FPGA SPARTAN 3E XILINX [44].

CAPÍTULO 3

Neste capítulo, é feita uma revisão da literatura sobre o estado da arte dos dispositivos usados como fonte de entropia não determinística para GPNA, GFNA e GQNA. Depois, é feito um estudo sobre os geradores e seus princípios de funcionamento, bem como as plataformas computacionais usadas na sua construção. Vale ressaltar que alguns modelos não usam computadores embarcados ou arquitetura reconfigurável, mas sim blocos lógicos digitais, portas lógicas e flip-flops.

REVISÃO BIBLIOGRÁFICA

3.1 Geradores de Números Pseudoaleatórios

Na literatura, são encontrados vários geradores de número pseudoaleatório baseado em solução de equações matemáticas, que em geral, são facilmente implementadas em várias linguagens de programação para que o computador possa usar esses dados a partir de uma semente. Alguns destes GPNA são: Uniformes, Lineares, Portáteis, dentre outros (L'ECUYER, 1994).

Um gerador uniforme é aquele que evolui em um espaço S , composto por um número finito de estados, portanto plenamente aplicável num computador, a partir de um lei de recorrência, como a equação 3.1, onde S_n é chamado de semente, f é uma função de transição e a saída pode ser determinada, mesmo após o n -ésimo passo, por meio de uma função (L'ECUYER, 1994):

$$S_n = f(S_{n-1}), n \geq 1 \quad (3.1)$$

Uma das características desse modelo é sua limitação espacial no intervalo $[0,1]$, o que pode ser ampliado, além disso, como “ S ” é finito, uma sequência qualquer gerada a partir desse modelo deve ser periódica. Ainda é observado um período “ p ” da sequência, que para fins de otimização deve ser o mais próximo da cardinalidade de S , quanto maior for a frequência de uso desse gerador. Por fim, em sequências binárias, sendo b o número de bits, o valor ótimo para “ p ” é dado por:

$$p = 2^b - 1 \quad (3.2)$$

Os geradores Lineares são aqueles fornecidos na maioria dos sistemas computacionais. Eles envolvem a geração de uma sequência de inteiros num espaço regido pela lei de recorrência apresentada na equação 3.3.

$$I_j + 1 = (aI_j + c) \times \text{mod}(m), j = 1, 2, 3, \dots \quad (3.3)$$

I é a sequência de inteiros, “a” e “c” são inteiros que agem como multiplicador e incremento respectivamente e a operação “mod” representa o resto da divisão de um número por outro e “m” é o módulo. Analisando a recorrência, o período da sequência gerada é, em geral, menor que o módulo, contudo se o período for igual ao módulo provocará uma sequência na qual não terá influência do valor da semente em sequências longas. Assim, este modelo tem como vantagens a velocidade na geração dos bits, facilidade de implementação e repetição numa mesma máquina e como desvantagem é a sua não portabilidade, alta correlação entre os bits e os bits menos significativos sofrem menos o efeito da aleatoriedade que os mais significativos (L’ECUYER, 1994).

Partindo da equação 3.3, isto é, a partir dos geradores lineares é possível criar novos cenários analisando o fator de incremento e suas implicações no módulo e no multiplicador. Por exemplo, há mostras teóricas e empíricas que fazendo o incremento igual a zero os bits do gerador são tão relevantes quanto se o incremento for diferente de zero, desde que seja feita uma boa escolha de multiplicador e módulo. Assim, na busca para produzir um cenário implementável em qualquer linguagem, em qualquer máquina, mesmo comportamento estatístico e semente confiável, surgem os geradores portáteis (L’ECUYER, 1994).

Esses geradores possuem algumas dificuldades de implementação. Fazendo o incremento ser igual a zero, matematicamente o efeito da recursão deixa de existir, logo sua implementação, ainda que em linguagens de alto nível, fica inviabilizada pelo tamanho dos registradores, em geral de 32 bits. Caso seja implementado num registrador maior (64 bits), o algoritmo torna-se não portátil. Outra saída é usar um algoritmo que consegue realizar essas somas usando registrador de 32 bits sem usar valores intermediários maiores que essa capacidade (PRESS *et al.*, 1992).

É importante destacar que há outros tipos de geradores de sequência pseudoaleatório, bem como há melhorias para os modelos apresentados acima. Como exemplos de outros geradores usando propriedades matemáticas, pode-se citar as variações dos lineares, multivariáveis e multivariáveis com incremento pseudoaleatório,

geradores não lineares e a composição de geradores já citados (L'ECUYER, 1996). A seguir irá ser visto uma outra maneira de implementar uma sequência pseudo aleatória, agora usando um sistema dinâmico.

De maneira mais complexa, é possível gerar tais sequências usando uma ferramenta conhecida como Autômato Celular (AC). Um AC é um Sistema Dinâmico discreto no espaço e no tempo, constituído por componentes individuais (células) que possuem estados que se alteram no tempo (evolução ou interação), de acordo com um conjunto de regras de transição, e baseando-se no estado da célula e/ou na sua vizinhança (JUSTO, 2013).

Assim, em Justo (JUSTO, 2013) é proposto um modelo de gerador de números pseudoaleatório baseado em AC, cuja semente é descrita pelo conjunto de equações 3.4-6 como ferramenta de criptografia. O mecanismo proposto é concatenar o estado das células, c_i , para compor blocos de informação pseudoaleatória ao longo de cada iteração do sistema. Logo, todos os estados das células, (c_i, t) , serão armazenados num vetor, β , tal que $|\beta|$ dependerá do número de iterações, t , necessárias para preencher o tamanho da informação a ser criptografada. Em 3.5, “B” é um vetor de um byte, isto é, $|\beta|/8$ e em 3.6, “Y” é um vetor composto de “p” blocos inteiro positivo, $Y \leq |B| / \rho$, e concatenado pela operação de “ou-exclusivo” (XOR) nos “B” consecutivos. Ressaltando que para esse algoritmo proposto foram realizados os testes de aleatoriedade ENT e DIEHARD.

$$\beta_{n(i-1)+(j+1)+(t_{mn})} = S(c_{n(i-1)+(j-1)}, t) \quad (3.4)$$

$$B_i = \sum_{j=1}^9 2^{8-j} \beta_{8(i-1)+j} \quad (3.5)$$

$$Y_i = B_{(i-1)(\rho+1)} \oplus \dots \oplus B_{(i-1)(\rho+\rho)} \quad (3.6)$$

3.2 Fontes de Entropia não determinísticas e seus usos em GFNA

A geração via *hardware*, se vale de fenômenos físicos para extrair as informações aleatórias, pois estes podem se comportar de maneira não determinística, por isso, são mais adequados para uma geração desses valores. Como exemplo desses processos, podem ser citados o ruído Johnson, ruído Zener, Efeito avalanche, decaimento radioativo e divisão de caminho do fóton em duas vias por feixes, dentro outros fenômenos da natureza, sobretudo aqueles de natureza quântica (STIPČEVIĆ, 2011).

A extração desses ruídos deve ser feita com bastante cuidado e deve passar por um pós processamento dos bits gerados, além dos testes estatísticos. O ruído Johnson é um ruído que é gerado devido a diferença de potencial aplicada aos terminais de um componente resistivo, idealmente aleatório e quantificável, contudo para cadeias longas ele gera correlação entre os bits, podendo, assim, inviabilizar o uso deste fenômeno para GFNA (STIPČEVIĆ, 2011).

Já o efeito Zener, detalhado no Capítulo 2, é causado pelo tunelamento dos elétrons na barreira de potencial, formando, em teoria, uma característica aleatório, porém como é um efeito de difícil isolamento, pode ocasionar efeito de memória na geração da sequência, sobretudo quando associado a resistores. Além disso, componentes associados ao diodo podem inserir ruído, o que dificulta a caracterização/medição do ruído Zener e numa conversão analógico-digital ou amplificação do sinal, pode ser introduzidos componente de tensão não lineares devido a limitação de largura de banda desses componentes, limitando ainda mais este componente (STIPČEVIĆ, 2011).

Na Figura 3.1, é mostrado um esquema para extração de bit de um sinal aleatório. Nesse modelo, pode-se ajustar a frequência de zeros e uns, todavia alguns problemas são encontrados. O ajuste dessa probabilidade está relacionado ao tempo de resposta do componente, um Flip-Flop tipo D. Outro problema é a estabilidade do sistema, pois o componente pode sofrer efeito da temperatura e isso impacta na variação do *jitter*¹, assim, é necessário um pós processamento. Em suma, é necessário um alto controle na confecção de geradores baseados em ruídos, neste caso o uso do *jitter*. Devido a esses e outros problemas que implicam a perda do efeito de aleatoriedade, em geral por correlação entre os bits gerados (STIPČEVIĆ, 2011).

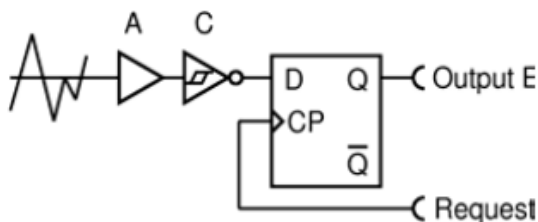


Figura 3.1 – Exemplo de extrator de sinal para GFNA. “A” é um amplificador e “C” amplificador comparador com ajuste de limiar. O resultado gera sequências por meio de Flip-Flop tipo D [40].

¹ *Jitter* é uma variação estatística no atraso dos entre uma informação e outra. Keller, Alexandre. **Asterisk na prática**. 2ª ed. Novatec: São Paulo, 2014.

Para contornar esse efeito de condicionamento do sinal aleatório, o autor sugere o uso de um *Free Running Oscillator* (FRO), Figura 3.2. Esse componente é obtido através de uma realimentação em um inversor, ou em uma cascata de inversores. Assim, a frequência, que via de regra varia com as oscilações de tensão e temperatura, potencializando, com entradas de oscilações rápidas, o efeito do *jitter* na frequência e na fase do sinal, uma vez que cada componente possui seu atraso. Então, a ideia base do FRO é que um rápido seja amostrado por um lento e, com isso, minimize o efeito do *jitter*, caso isso não ocorra haverá correlação entre os bits (STIPČEVIĆ, 2011).



Figura 3.2 – Dois tipos de FROs, a esquerda um de alta velocidade e a direita um de baixa velocidade. A frequência é determinada pelos *delays* internos e capacitâncias esporádicas [40].

Chindris e Muresan (2008) propõe um gerador baseado no efeito avalanche com alta entropia, ou seja, aquele com elementos semicondutores sensível a ionizações de impacto ou com alta incidência dela. A entropia e a informação são ligadas pelo Teorema da Entropia de Shannon, Equação 3.7, e a fonte de tal efeito é o efeito avalanche presente nos transistores de base junção, onde $I(x)$ é a função informação presente em um evento, que é dada em função de sua probabilidade de ocorrência.

$$H(x) = E(I(x)) = - \sum_{i=1}^n p(x_i) \log_2 p(x_i) \quad (3.7)$$

Partindo deste princípio, eles montam um circuito para amplificar o ruído gerado pelo efeito avalanche e chegam a um sinal de 20 KHz de frequência. Depois disso, eles fazem um condicionamento do sinal, filtragem da componente DC, para um conversor analógico digital em amostras de vinte microssegundos a dez bits de capacidade e depois envia essas sequências, no total de 1100, a um *baudrate* de 115200 bits por segundo pela serial do SoC implementado.

A Figura 3.3 mostra um modelo proposto para aquisição destes dados de alta entropia. Para isso, os autores montam um arranjo com transistores para produção do efeito avalanche e fazem análise do sinal em um FPGA, condicionando as amostras junto a medição de temperatura, a saída dos dados são enviados por meio de uma interface de

comunicação serial para análise em um equipamento externo. Neste caso, não há pós processamento das amostras e, segundo os autores, o propósito é apenas mostrar um modelo de aquisição de dados de alta entropia, no entanto é importante destacar que eles conseguem um taxa de quase 50% entre a geração de bits zeros e bits uns (CHINDRIS & MURESAN, 2008).

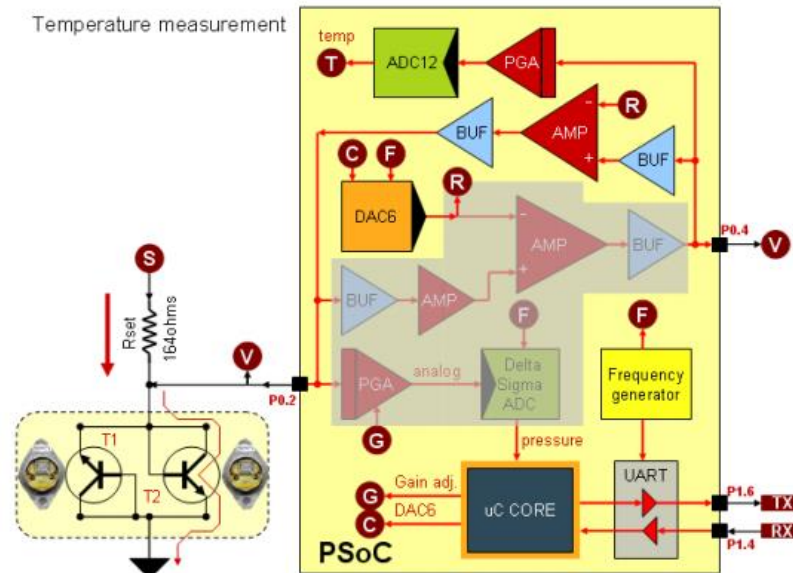


Figura 3.3 – Modelo de aquisição de dados com alta entropia proposto [8].

É possível também encontrar fontes de entropia não determinísticas em sensores eletrônicos, sobretudo nos sensores de captura de imagens e em sensores que são bastante sensíveis a ruídos (sensores de medição inercial, por exemplo). Usando um acelerômetro, Hong e Liu (2015) propõem um protocolo de geração de números aleatórios a partir do ruído produzido por este sensor.

Para isso, há dois passos a serem seguidos para extração da semente: remoção dos dados de rastreamento inicial e eliminação da correlação entre os múltiplos sensores (eixo-x, eixo-y e eixo-z), o que indica pós processamento dos dados. Assim, eles propuseram três estágios sequenciais: “WASH”, onde é esvaziado e removido os padrões de previsibilidade dos dados gerados pelo sensor, seja pelo usuário, seja pelo dispositivo, buscando o estado de não estacionaridade. “RINSE”, onde é inserido a imprevisibilidade e é aplicada uma Transformada Rápida de Fourier na sequência gerada e o “SPIN”, onde os dados do acelerômetro deverão gerar a sequência de composição da semente suficientemente não previsível, introduzindo uma sequência conhecida como Mersenne Twister (HONG & LIU, 2015).

Ainda neste trabalho, o pós processamento consiste em pegar amostras a 100 Hz do acelerômetro e aplicar a códigos conhecidos para formar datagramas de até vinte mil números para depois aplicar os testes estatísticos do NIST. No entanto, este trabalho aplica apenas seis dos quinze testes proposto pelo órgão e falha no Teste de Complexidade Linear e mantém uma margem satisfatória nos demais.

O princípio de geração de imagens é outra fonte explorada na geração de entropia para GFNA. Em Zang (ZANG *et al.*, 2014) é usado um sensor de captura de imagem via câmera de um smartphone, conforme a Figura 3.4. O sensor presente no aparelho é o IMX105 da Sony com resolução de 3264 x 2448 *pixels* e cada unidade possui oito bits associados para gerar uma imagem do tipo “Bitmap”. A captura das imagens devem ter o tempo mínimo de 33,3 milissegundos. A câmera possui certificação ISO800.

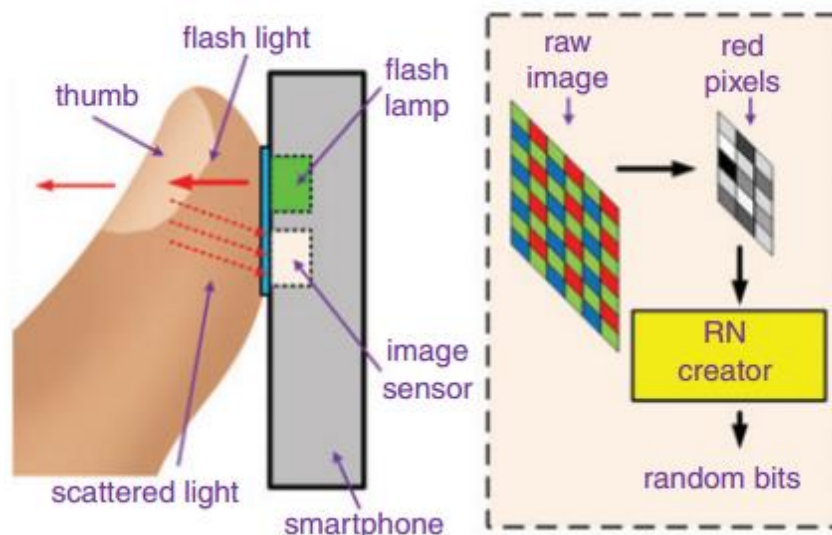


Figura 3.4 – Aquisição de Dados via sensor de imagens presente em celulares [45].

Os resultados desse trabalho são bastante relevantes no que diz respeito ao pós processamento das informações, pois ele foi submetido aos quinze testes do NIST e obteve resultado satisfatório em todos eles, isto significa que os testes ficaram todos superior a 98% da condição de aleatoriedade. As amostras usadas foram mil amostras de 1 Mbit, além disso, os autores estão trabalhando numa interface em *Android* para visualização da geração desse fenômeno (ZANG *et al.*, 2014).

Outra maneira de se conseguir efeitos não determinísticos é por meio de materiais fotônicos, como as células de Kerr ou células de Pockels. Esses efeitos rotacionam um feixe de luz, ou fótons, quando estes são forçados a atravessar materiais característicos de tais efeitos, portanto limitando o número de fótons desse feixe e o ângulos de rotação é

possível gerar situações não determinísticas. Valendo ressaltar que o efeito Kerr é de segunda ordem, enquanto o efeito Pockels é de primeira ordem, e esses seguem as Equações 3.8 e 3.9, respectivamente (IWANSSON, 1999):

$$\Delta\varphi = 2\pi BLE^2 \quad (3.8)$$

$$\Delta\varphi = \left(\frac{2\pi}{\lambda}\right)n^3rLE \quad (3.9)$$

Em que $\Delta\varphi$ é a variação de fase entre os dois meio feixes óticos, B é o coeficiente de Kerr, L é o comprimento ótico percorrido pelo feixe, E módulo do campo elétrico, λ o comprimento de onda, n o índice de refração do meio e r é o coeficiente de Pockels.

Um exemplo disso é o uso do cristal *Periodically-Poled Potassium Titanyl Phosphate* (PPKTP), que é um material fotônico, não linear, usados para limitar os fótons a serem introduzidos numa fibra monomodo e, assim, servir de entrada para um gerador de números aleatórios totalmente ópticos, vide a Figura 3.5 (FIORENTINO, *et al.* 2006).

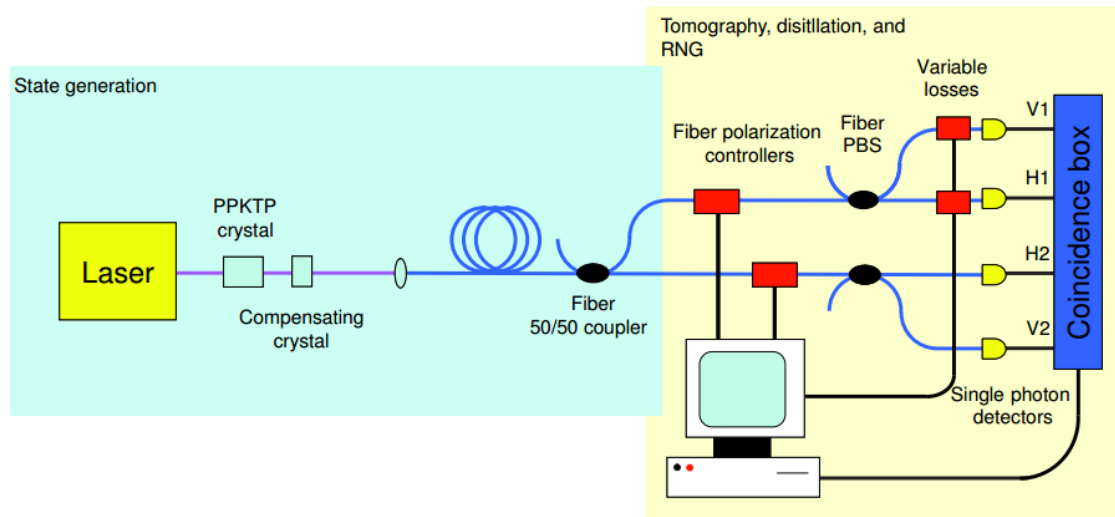


Figura 3.5 – Uso de cristal fotônico PPKTP em cenários totalmente ópticos [7].

3.3 Geradores Quânticos de Números Aleatórios

A maioria dos GQNA baseiam-se no princípio da Figura 2.12, entretanto há outros geradores baseados em outros princípios quânticos para a geração de sequências aleatórias, no qual objeto desta subseção. Vale ressaltar que alguns destes GQNA possui licença e certificação para comercialização e uso em cenários reais, principalmente na parte de criptografia dos dados.

O *Quantis*, Figura 3.6, é um GQNA comercial, que faz uso do princípio mencionado acima, possui baixo custo e uma taxa de 16 Mbit/sec, que pode ser acoplado na porta PCI de uma placa mãe qualquer ou ainda por meio de uma porta USB. O fabricante garante uma variação de até 10% na geração dos bits, que devem ser equiprováveis, assim as porcentagem variam de 45% até 55% na geração de bits zeros ou uns. Ele tem em sua definição base uma checagem tanto da fonte de emissão de luz quanto dos dois receptores ópticos (IDQ, 2010).



Figura 3.6 – GQNA Quantis [13].

Uma outra maneira de se extrair propriedades da Física Quântica para a construção de GQNA é apresentada por Soares (SOARES *et al.*, 2014). Neste trabalho, um esquema com um único detector de fótons únicos, Figura 3.7.

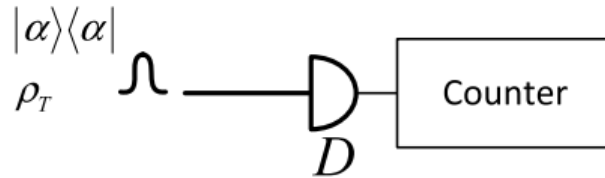


Figura 3.7 – Modelo proposto para GQNA usando único foto detector [39].

O princípio de operação é dado pela contagem de fótons no APD, se for par é gerado bit zero ou se for ímpar é gerado bit um. A probabilidade de geração do bit, por exemplo, é dado pela Equação 3.10, onde N é o número de pulsos de gatilho gerados ao longo do tempo.

$$P_0 = \sum_{n=0,2,4,6,\dots}^N \binom{N}{n} P_n (1-P)^{N-n} \quad (3.10)$$

Esse modelo foi submetido é implementado experimentalmente e submetido aos testes do NIST, tido como satisfatórios. Esse modelo sofre alguns problemas como uma fonte de fótons únicos, tal propriedade impacta no desempenho do gerador, além disso, o APD sobre alguns efeitos que interferem na contagem, como a contagem de escuro e o *Afterpulsing*. Em relação aos modelos com dois detectores, ele é mais barato, mais

simples de se implementar e gera sequências com baixa polarização, como desvantagem ele é mais lento na geração que um gerador que faz uso de dois fotodetectores (SOARES *et al.*, 2014).

Outro GQNA encontrado na literatura é baseado na captura de imagens, esquema da Figura 3.8, proposto por Sanguinetti (2014). O aparato experimental montado pela equipe combina sensores, elementos ópticos e saída para computadores para manipulação dos bits gerados. O princípio envolvido é a geração das imagens por meio da conversão dos fótons, que são injetados num divisor de feixe até um fotodetector para, daí, serem amostrados e pós processados. A equação 3.11, mostra a entropia do sistema de geração, baseada numa distribuição de Poisson.

$$H(x) = \frac{\bar{n}}{\ln 2} [1 - \ln \bar{n}] + \frac{\exp(-\bar{n})}{\ln 2} \sum_{m=0}^{\infty} \frac{\bar{n}^m \ln(m!)}{m!} \quad (3.11)$$

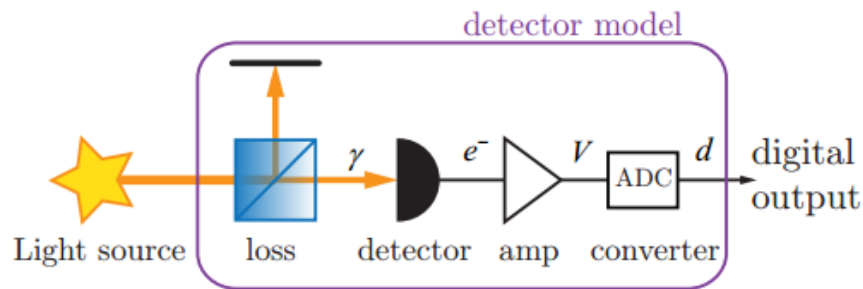


Figura 3.8 – Modelo proposto para GQNA usando sensor CCD ou CMOS [35].

Os sensores de imagem usados são uma câmera astronômica (ATIK383L) e a câmera do Nokia N9, cuja câmera apresenta melhor desempenho. Dessa forma, os autores conseguem gerar taxas um pouco acima de 1 Gbit e tais sequências foram submetidas tanto aos testes do NIST quanto aos DIEHARD e outros testes complementares e obteve êxito em todas estas baterias. O pós processamento é feito por um FPGA, sendo possível a captação de até 3 Gbps para aquisição, mas é informado que 1 Mbps atende a ampla maioria das aplicações.

A taxa de geração de bits aleatórios também é muito importante, pois a medida que a computação evolui, segundo a lei de Moore, o processamento dobra a cada 18 meses, e para os geradores acompanhar este crescimento implica seu uso nestas aplicações. O modelo proposto por Wei (2015) é usar um fotodiodo com elemento intrínseco, ou fotodiodo PIN, e um LED superluminescente (SLED) como fonte de

entropia junto a um sistema de filtragem e amplificação para mensurar a tensão e aplicar ao pós processamento, o ensaio pode ser visto na Figura 3.9.

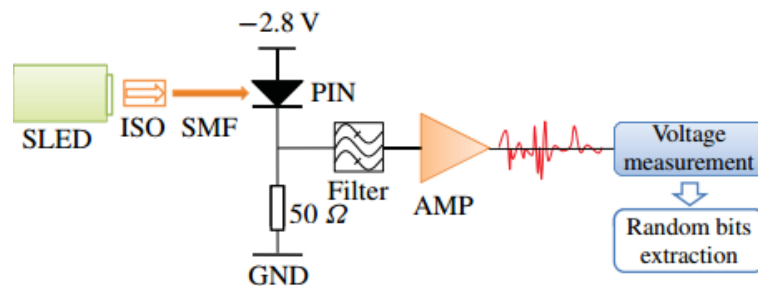


Figura 3.9 – Proposta de GQNA de alta velocidade usando Fotodiodo PIN [43].

Partindo deste experimento, o foco dele é a geração em altas velocidades, isso ocorre devido à natureza dos componentes selecionados, muito deles usados em circuitos de rádio frequência. Assim, as taxas de geração chegam a dezenas de Gbps de amostras por segundo para aplicar o pós processamento, sugerido pelos autores o uso de um FPGA e o algoritmo de von Neumann (ver Apêndice A) para reduzir a correlação, pois neste trabalho foi observado os sinais em um osciloscópio no limite de sua capacidade. Os testes aplicados foram aqueles proposto pelo NIST e pelo DIEHARD obtendo resultados satisfatórios em todos os testes em ambos os cenários.

O último trabalho a ser exposto também com foco na geração sequências de maneira rápida, é proposto por Nie (2015). O experimento realizado é análogo a primeira ideia proposta para um GQNA, contudo há algumas modificações na parte óptica do sistema, vide Figura 3.10, e na parte de controle sobre o fenômeno físico envolvido. O pós processamento é feito em um FPGA que se encarrega de despachar os dados coletados numa matriz de Toeplitz para rodar os testes estatísticos no MATLAB®.

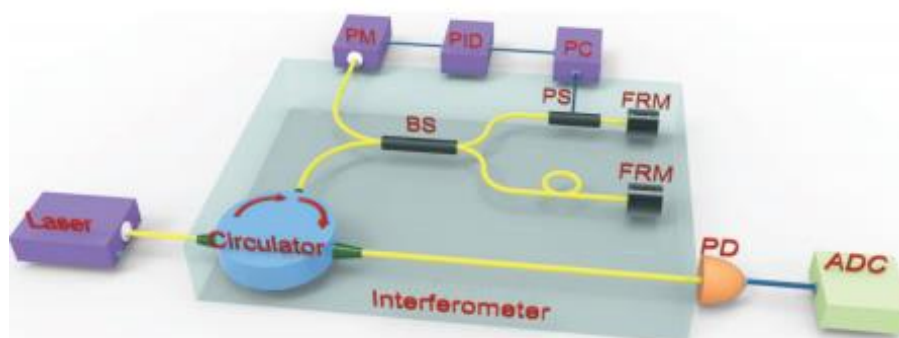


Figura 3.10 – GQNA Óptico de alta velocidade e alta capacidade [27].

Com isso, os autores conseguiram resultados práticos na geração a uma velocidade de 68 Gbps e as sequências passaram nos 15 testes pospostos pelo NIST com taxa superior a 99% em todos os testes.

3.4 GFNA Baseados em Sistemas Digitais

Os geradores físicos podem ser implementados também por meio de blocos lógicos digitais ou por meio de operações lógicas numa sequência obtida por um *hardware* externo. O primeiro trabalho que será usado para ilustrar este exemplo é o uso do Sistema Operacional Linux para realizar leituras dos periféricos de um computador e a partir dos sinais captados aplicar uma série de operações lógicas a fim de produzir um algoritmo seguro baseado em GFNA.

Guterman (2006) propõe o chamado LRNG (*Linux Random Number Generator*). Esse mecanismo se apoia em três *procedures* assíncronas, a primeira coleta a entropia via Sistema Operacional baseado em seus eventos internos, a segunda usa essa entropia para alimentar o gerador por meio de funções e a terceira atualiza a saída e atualiza o sistema de coleta. Feito isso, um algoritmo de hash (SHA-1) é alimentado.

As fontes de entropia são classificadas de três formas: primária (512 bits), secundária (128 bits) e *urandom* (128 bytes). Tais fontes são captadas por meio dos eventos de mouse (12 bits), teclado (8 bits), disco rígido (3 bits) e interrupções (4 bits), que depois de geradas são inseridas em uma estrutura, Figura 3.11, para gerar o resultado que é entrada no algoritmo de criptografia SHA-1.

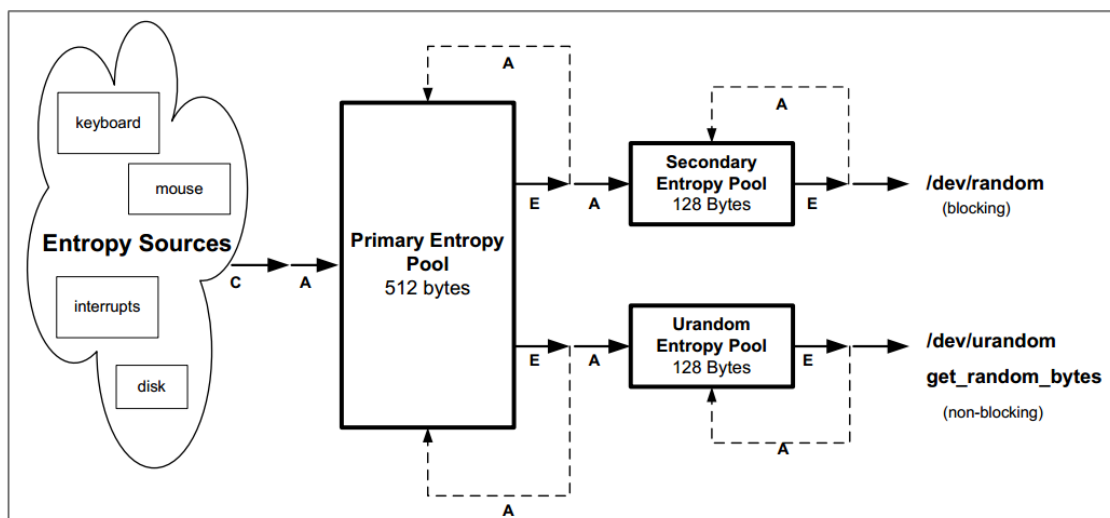


Figura 3.11 – Estrutura de funcionamento do LRNG [10].

O ambiente de teste foi o openWRT, que é um sistema operacional baseado em Linux para roteadores sem fio, assim foram feitos os algoritmos de extração das informações das fontes de entropia e a montagem dos blocos de informação em palavras de 32 bits. Essa montagem, como as fontes possui tamanhos de geração distintos, foi feita com base em operações ou-exclusivo (XOR). Além disso, é verificada a ocorrência de certos bits visando a eliminação da correlação entre os bits gerados. Montado a estratégia de geração, foram simulados ataques para visualizar a segurança deste modelo.

As conclusões deste trabalho fazem alusão as limitações da solução em relação a plataforma usada, ou seja, o grande número de distribuições Linux disponíveis, que possam não ser atendidas pelo LRNG e o modelo, por ser aberto, pode ser facilmente auditado e melhorado pela comunidade. Ao final, ficam algumas sugestões de melhorias, a que mais se assemelha desta dissertação é explorar o efeito da entropia sem que num dado momento o sistema fique com efeito de memória, isto é, torne-se um GNPA.

O uso da eletrônica pode ser explorado também sem ser pelo uso de um elemento físico adicional. O trabalho de Böhl (2014) faz uso de um oscilador em anel, experimento semelhante ao apresentado na Figura 3.2, a fim de produzir efeitos a partir das imperfeições dos circuito analógicos ligados em cascata. Esse oscilador tem o *design* projetado para operar na faixa próxima de 1GHz, onde o autor propõe uma sequência de compressões usando portas XOR implementadas em um FPGA para obter alta entropia, aqui ela é associada ao *jitter* dos elementos e baixo efeito de polarização das amostras.

Tal entropia é estimado com base na Equação 3.13 e leva conta os seguintes fatores: k é a constante de Boltzmann, T a temperatura absoluta, P é a potência total dissipada no oscilador, V_{dd} é a tensão de alimentação, V_c é a tensão característica do MOSFET (Equação 3.14), η é uma constante de valor usado na simulação igual a 0,952 e ΔT é o *jitter* acumulado pelos elementos.

$$\sigma_{\Delta T} = \sqrt{\frac{8}{3n}} \sqrt{\frac{kT}{P} \frac{V_{dd}}{V_c}} \sqrt{\Delta T} \quad (3.12)$$

$$V_c = \frac{3}{8} \left(\frac{V_{dd}}{2} - 0,615 \right) \quad (3.13)$$

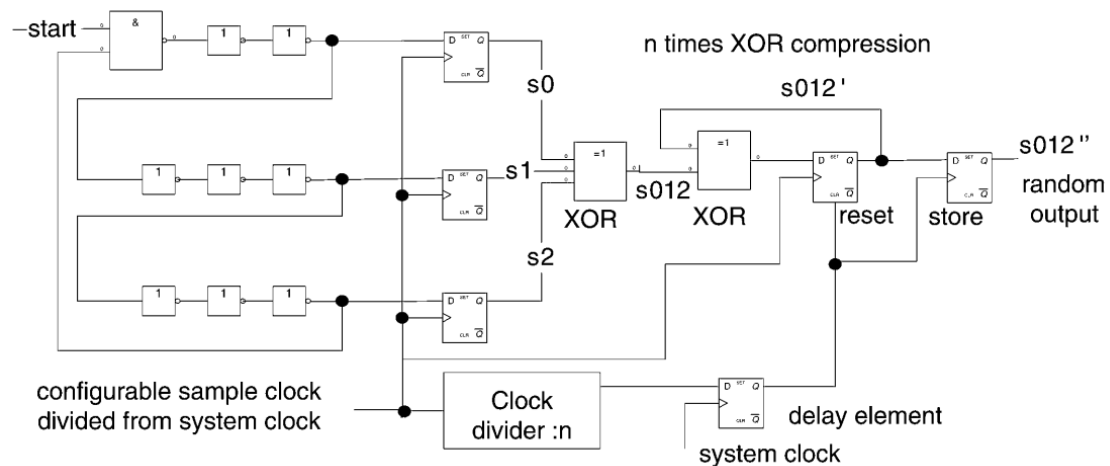


Figura 3.12 – TRNG baseado em efeito de semicondutores [2].

A Figura 3.12 mostra o experimento proposto por Böhl, ressaltando que o essas “n” compressões XOR são implementadas num FPGA de alta capacidade para aplicar os resultados aos testes estatísticos. Com relação aos testes padrões do NIST ou DIEHARD, o gerador proposto não foi submetido, contudo o autor faz uma análise para eliminar a correlação usando o teorema do limite central, onde uma quantidade de amostras são calculadas por uma unidade de computação.

Um último experimento baseado em Sistemas Lógico Digitais é proposto por Logue e Platt (2015). A fonte usada por eles é baseada na avalanche de semiconductor, usando transistores bipolares, e todo o pós processamento é feito com base nos elementos combinacionais e sequenciais oriundos da eletrônica digital. Após a etapa de aquisição do sinal de entropia, o pós processamento proposto é feito com base em registradores de deslocamento, cascata de Flip-Flops, e portas XOR, visando implementar o algoritmo de von Neumann. O *clock* do sistema é feito por meio de um Circuito Integrado 555 e o armazenamento dos bits é feito em buffers de dados. O diagrama geral do sistema pode ser visto na Figura 3.13.

Esse sistema consegue gerar bits a uma taxa de 60 Hz e não faz menção a realização de algum teste estatístico certificado ou próprio. Uma vantagem deste sistema é o seu baixíssimo custo de confecção, contudo qualquer alteração que se deseje aplicar deverá ser feita com reposição de algum circuito integrado ou, ainda, um recálculo das

componentes analógicas envolvidas no processo de pós processamento dos bits.

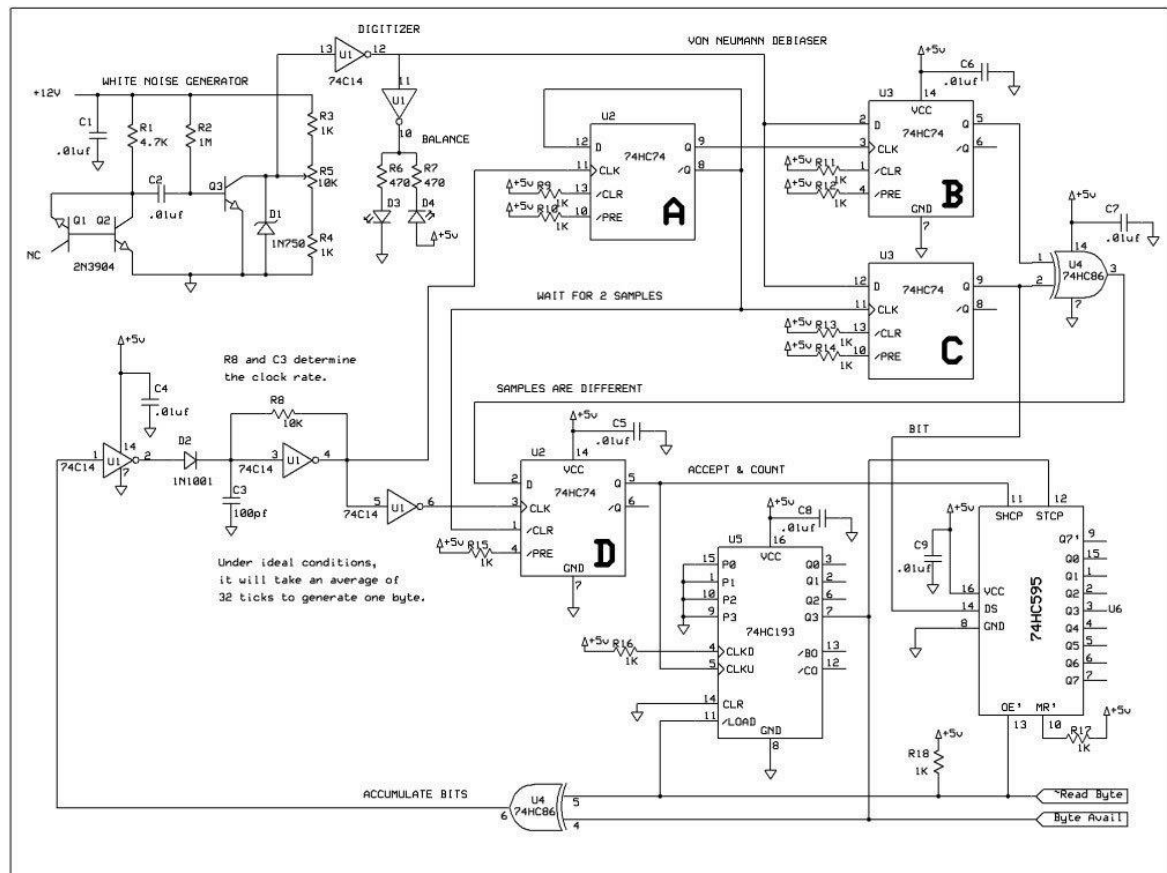


Figura 3.13 – RNG baseado em efeito avalanche pós processado com blocos lógicos [23].

3.5 GFNA Baseados em Sistemas Embarcados e FPGA

Outro cenário que é encontrado nos GFNA é o uso de Microcontroladores ou FPGA como fonte de pós processamento dos bits gerados pelas fontes de entropia não determinístico. Os trabalhos com FPGA tem dominado este cenário devido a busca por taxas de geração cada vez maiores e essas arquiteturas permitem isto, porém há alguns cenários de aplicação que arquiteturas de 8 bits ou 32 bits podem atender as demandas das aplicações de maneira satisfatória.

Um trabalho usando Microcontroladores é o gerador chamado de T12RNG, Figura 3.14, também baseado no efeito avalanche a partir de transistores BJT. Tal dispositivo usa o microcontrolador Attiny12 de arquitetura AVR, mesma fabricante do microcontrolador do Arduino. Ele é de baixa capacidade e recursos bastante limitados, tendo inclusive o autor tendo que fazer adaptações para poder comunicar o gerador com

o seu computador pessoal para visualização dos dados aleatórios gerados em um *software* do tipo terminal.

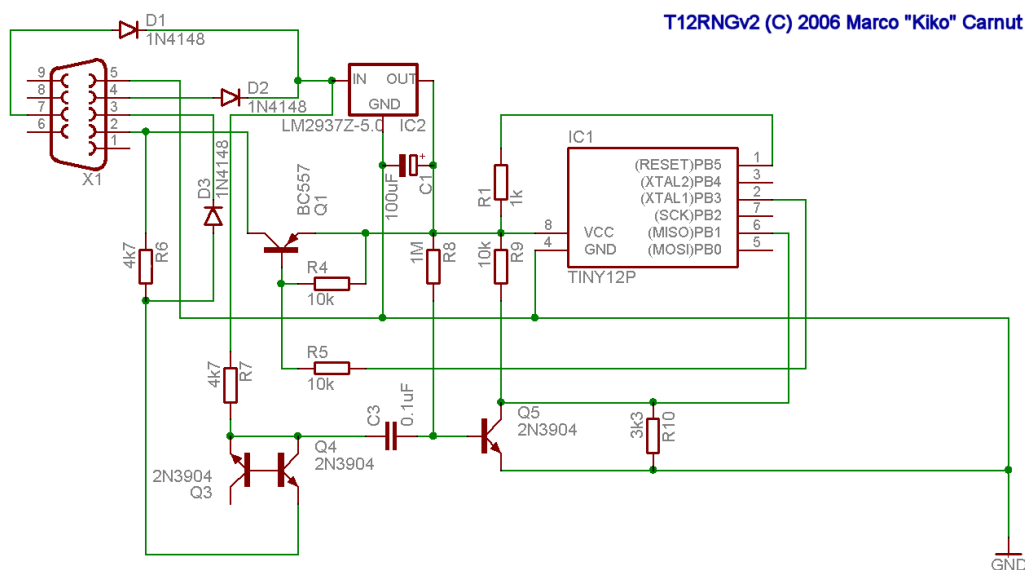


Figura 3.14 – Gerador T12RNG [41].

Duas adaptações interessantes são feitas no gerador para torná-lo mais amigável ao usuário, uma delas é a criação de um circuito para receber alimentação da porta serial tipo DB-9 e um circuito para comunicar os dados via serial, apenas no “sentido de ida”, posto que o micro não possui este recurso. Ele possui uma taxa de geração de 60 bytes por segundo, cujo foco é aplicação em criptografia de senhas e complemento a GNPA.

O *firmware* salvo no attiny extrai os bits de saída do comparador analógico a uma taxa de 9600 amostras por segundo e aplica um pós processamento baseado no algoritmo de von Neumann, o que reduz esse número de amostras, visando manter a proporção entre os bits zeros e uns. Os testes que esse gerador foi submetido foi adaptado pelo autor e são baseados no padrão ENT, ressaltando que para passar nos testes foi feita uma função mais complexa na mistura dos bits, geralmente são usados operações XOR.

Ao longo deste capítulo, foram apresentados dois trabalhos que fazem uso de arquiteturas do tipo FPGA ou abrem perspectivas para uso desta tecnologia para confecção ou pós processamento de dados tanto em GFNA quanto em GQNA, sobretudo aqueles que visam a geração de sequências grandes e rápidas de serem geradas. São eles: NIE *et al.* e Gabriel Chindris & Marius Muresan.

O trabalho de Danger (2009) faz um comparativo entre vários geradores baseados no princípio de anéis osciladores e suas variações. O pós processamento feito é também

com base no algoritmo de von Neumann e os testes aplicados são os do NIST, tendo sucesso em todos com menor taxa em 98%. Por fim, os autores tomam como parâmetro os seguintes quesitos para comparação com outros cinco modelos: ordem de complexidade com relação à capacidade (uso de células) e taxa de geração. Isso para validação do modelo proposto por eles.

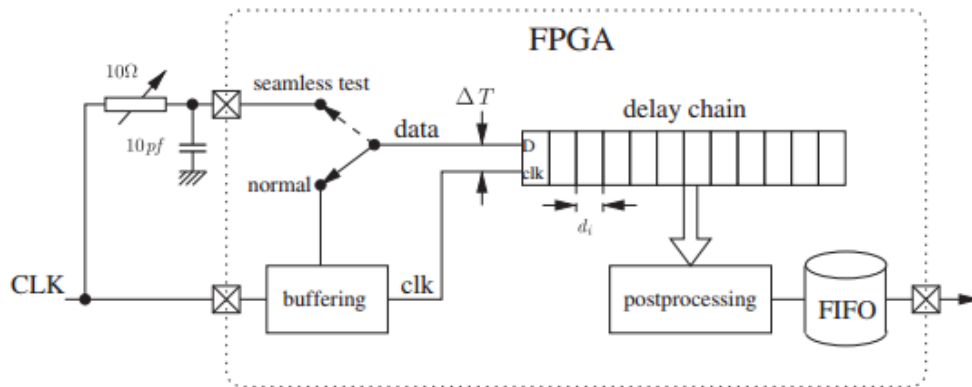


Figura 3.15 – Gerador baseado em FRO implementado num FPGA [6].

3.6 Patentes

Uma pesquisa foi feita em bases de patentes nacional e internacionais sobre métodos utilizados para geração quântica de números aleatórios. Como já visto, há técnicas diferentes com relação ao processo de detecção do processo não determinístico e a plataforma computacional a ser usada. Dentre as patentes encontradas, duas possuem mais aderência com a dissertação.

A patente de NETWORK DISTRIBUTED QUANTUM RANDOM NUMBER GENERATION (US20120221615 A1), utiliza técnicas pluralidade de estados coerentes na emissão e detecção de fótons, onde cada estado indica uma quantidade de fótons que irá gerar a corrente necessária para alimentar o circuito gerador da sequência.

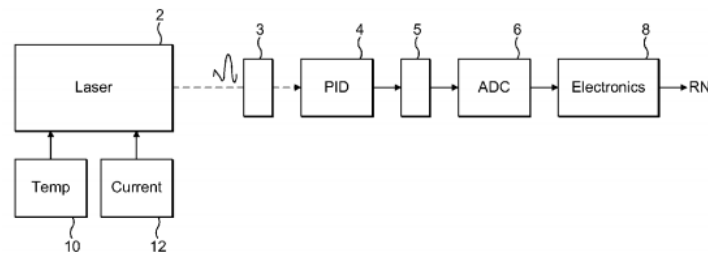


Figura 3.16 – RNG baseado em emissão/detecção de fótons coerentes.

Fonte: Google Patents. Adaptado de <http://www.google.com/patents/US8930429>.

Já a patente ULTRAFast QUANTUM RANDOM NUMBER GENERATION PROCESS AND SYSTEM THEREFORE (US20130036145 A1) utiliza etapas de funcionamento de lasers simples e com grande largura de banda de modulação por meio de um contador de impulsos elétricos, transformando a fase aleatória dos impulsos como fonte de entrada para o sistema por meio de um fotodiodo PIN, conforme a Figura 3.14.

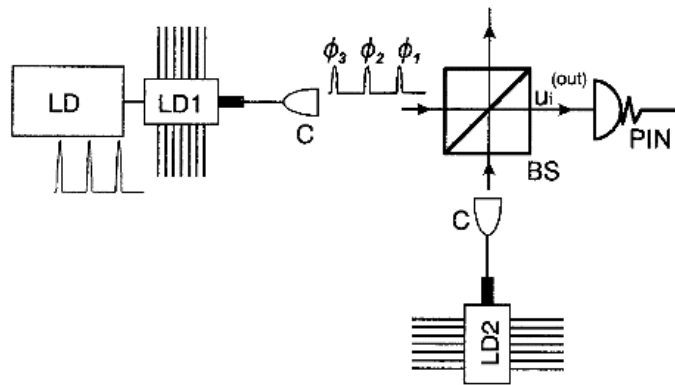


Figura 3.17 – RNG com fotorreceptor PIN.

Fonte: Google Patents. Adaptado de <https://www.google.ch/patents/US20130036145>.

3.7 Conclusão

Ao final deste capítulo, percebe-se que na revisão apresentada há um grande número de pesquisa em várias frentes, eletrônica, materiais, teoria da informação, física quântica, sistemas embarcados, *etc.* que buscam produzir geradores verdadeiramente aleatórios com altas taxas de geração e possam ser replicados nas plataformas de comunicações usadas atualmente.

CAPÍTULO 4

Este capítulo é destinado à metodologia usada no trabalho, assim como os resultados obtidos e sua análise. Ao longo dele, são expostos os materiais e métodos usados, as curvas, o pós processamento dos dados e como os testes foram aplicados. Após isso, é feita uma mostra dos resultados coletados a partir do sistema em suas diversas plataformas computacionais. Por fim, é feita a análise dos resultados.

METODOLOGIA E RESULTADOS

4.1 Materiais e Métodos

Os ensaios foram realizados no Laboratório do Grupo de Pesquisa em Sistemas de Telecomunicações e Sistemas Embarcados (GDESTE/IFCE) vinculado ao Programa de Pós-Graduação em Engenharia de Telecomunicações (PPGET/IFCE). O trabalho contou com o uso de duas plataformas computacionais embarcadas Arduino (Arduino UNO e Arduino Due) e um kit FPGA Spartan 3E, todos descritos no Capítulo 2. A aquisição do sinal foi feita inicialmente em um *protoboard* e depois foi confeccionada uma placa de circuito impresso (PCI). As medidas foram feitas por meio de um osciloscópio de alto desempenho, RTO1014 da R&S, um multímetro de bancada, U3401A da *Agilent*, e um fonte de bancada, E3631A da *Agilent*, para alimentação do sistema, devido ao seu baixo ruído. Após a aquisição dos dados pelas plataformas, eles são pós processados e submetidos aos testes por meio da ferramenta Matlab© (R2015a), cuja comunicação entre o Sistema Embarcado e o ele é serial UART do tipo RS-232 disponível em todas elas. A fonte de entropia usada foi o efeito avalanche por meio de um arranjo entre transistores BJT de baixo ruído, 2N3904, e para amplificação do sinal de avalanche também é usado o mesmo tipo de transistor. O fluxo do experimento pode ser visto na Figura 4.1. É importante destacar que os dados não ficam armazenados de maneira permanente nas plataformas embarcadas, porém é possível fazê-lo.

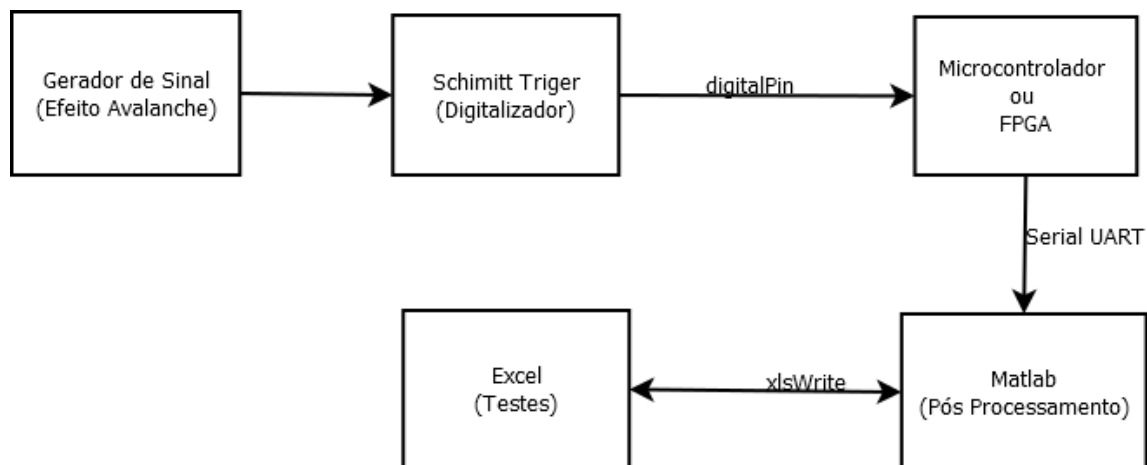


Figura 4.1 – Fluxograma do Experimento com Efeito Avalanche.

Fonte: Autor.

4.1.1 Experimento com Efeito Avalanche

A fonte de entropia não determinística usada no trabalho é baseada no efeito avalanche extraída por meio da polarização reversa da junção “pn” de um transistor polarizada com coletor aberto. Com o sinal gerado, ele é amplificado também por um transistor igual aos geradores do sinal, pois ele apresenta baixo ruído. A alimentação do sistema é contínua e com um máximo observado de 15V até 18V, a depender do transistor usado. Após a etapa de amplificação, o sinal é limitado por um Diodo Zener de 4,7 V (1N750), para não danificar os componentes digitais usados, uma vez que todos são TTL.

A Figura 4.2 mostra o esquemático de ligação do gerador de sinal (LOGUE & PLATT, 2015). Diante disso, há duas possibilidades: a primeira é trabalhar com sinal analógico por meio de conversor analógico digital para processar o sinal ou digitalizar este sinal por meio de um componente eletrônico para otimizar o código no microcontrolador ou no FPGA. Assim, foi optado pelo segundo caminho, porque será necessário um pós processamento da sequência, economizando a implementação tanto no Arduino quanto no FPGA relativo a conversão analógico digital.

A digitalização do sinal é feita por meio do circuito integrado 74HC14, que é um inversor do tipo Schmitt Trigger (ST) que opera no modo TTL. A Figura 4.3 mostra que a saída de sinal, que é entrada no ST e a sua saída já vai direto nos pinos digitais do Microcontrolador ou do FPGA. Isso é feito por que o interesse neste momento é apenas a geração das sequências e só depois aplicar o pós processamento, além deste componente ser bastante usado e ter baixo custo. Vale ressaltar que o ST também possui uma margem

de histerese, o que dá mais confiabilidade ao sistema na durante as oscilações entre os valores de tensão na geração dos bits.

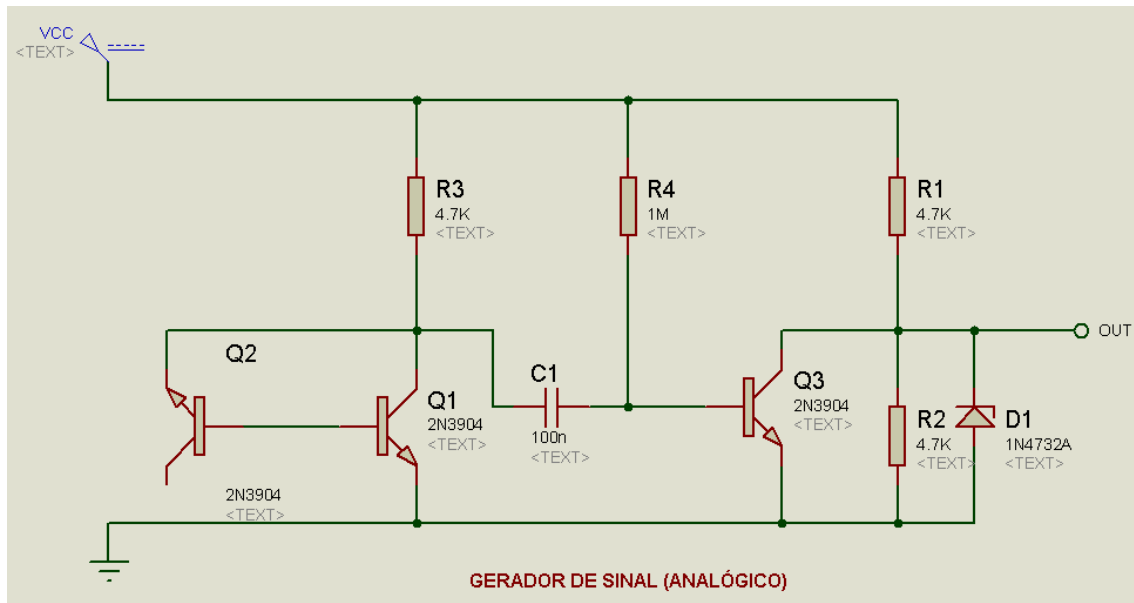


Figura 4.2 – Circuito de geração e amplificação do sinal de Avalanche usando transistores, fase analógica.

Fonte: Autor.

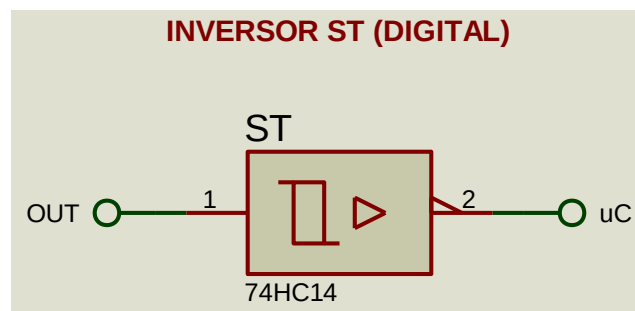


Figura 4.3 – Inversor Schmitt Trigger de 6 canais, fase digital.

Fonte: Autor

Os sinais de saída do ST são associados aos pinos digitais do Arduino ou do Spartan para serem repassados a um computador para pós processamento e para entrar na bateria de testes.

4.1.2 Metodologia Utilizada

A sequência de trabalho norteou-se pela determinação das fonte de entropia não determinística, no caso o efeito avalanche nos transistores, montados em protoboard, Figuras 4.4, para validar tais efeitos nos aparelhos de medições e caracterizar suas curvas.

Após os testes com protoboard, foi manufaturada uma placa de aquisição de sinal, Figuras 4.5 e 4.6, visando evitar falhas de mau contato e melhor organização dos componentes para análise. Também foi manufaturada uma placa para a etapa de digitalização do ST, porém esta, as vezes, apresentava um comportamento fora do padrão do componente.

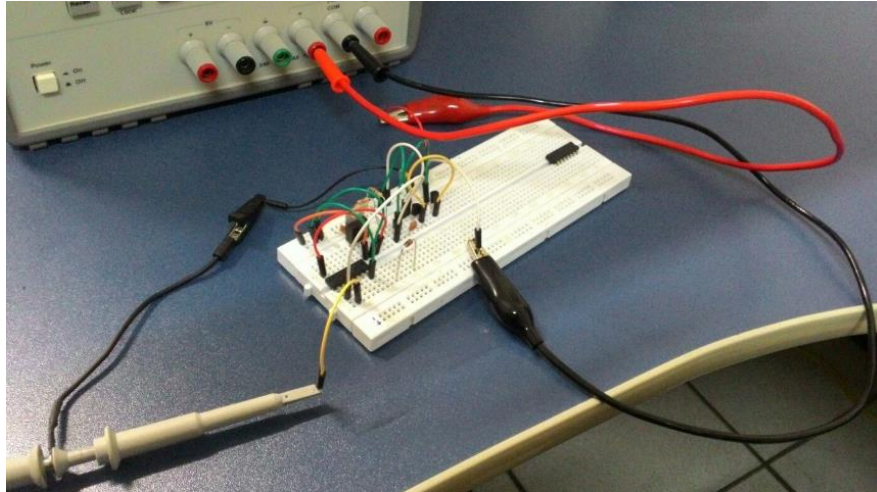


Figura 4.4 – Caracterização do Efeito Avalanche.

Fonte: Autor.

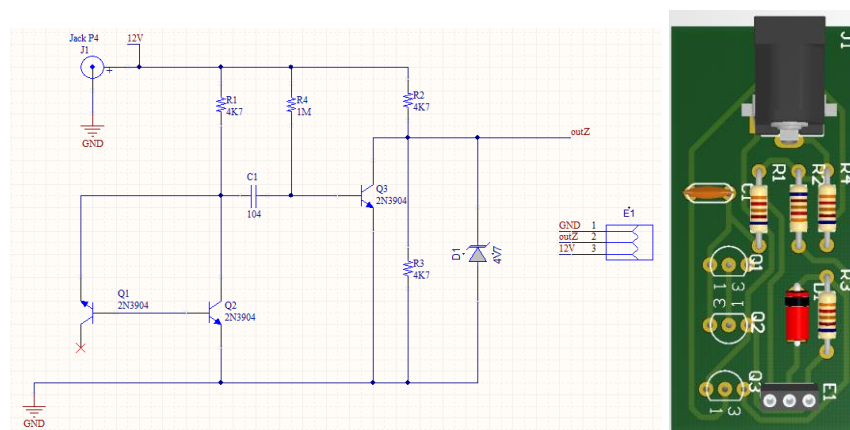


Figura 4.5 – Esquemático e PCB do GFNA manufaturado.

Fonte: Autor.

Assim, o sinal gerado pela placa sendo similar ao sinal gerado anteriormente, vai para a etapa de digitalização por meio do inversor ST e, com isso, fica apto a trabalhar com as ferramentas computacionais, Figuras 4.7 e 4.8.

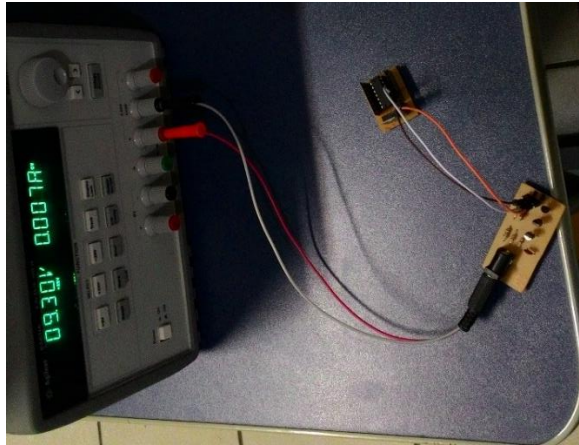


Figura 4.6 – Sistema de Testes das placas do GFNA em bancada para análise de alimentação, ruído e distorções do sinal de avalanche.

Fonte: Autor.

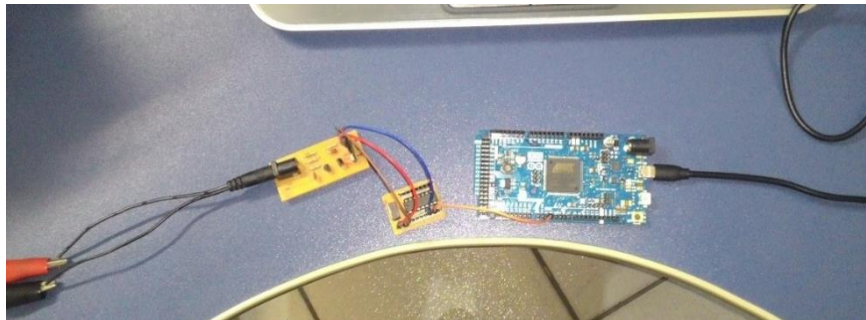


Figura 4.7 – Ensaio do GFNA com Arduino DUE para coleta das amostras e execução do pós processamento e dos testes de aleatoriedade.

Fonte: Autor.

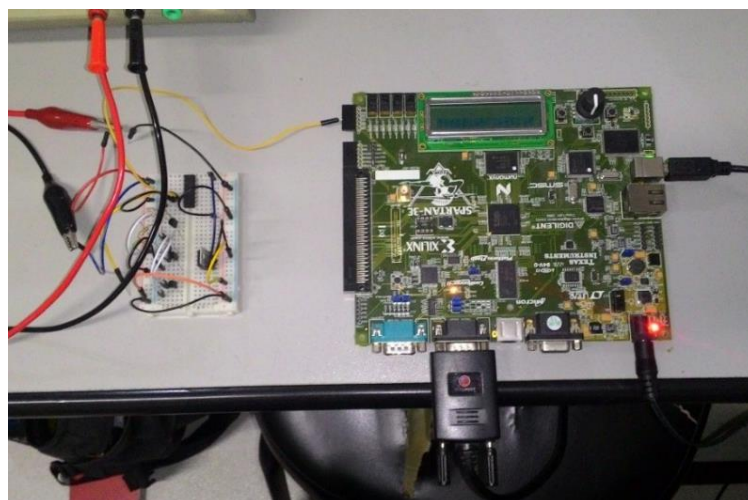


Figura 4.8 – Ensaio do GFNA com FPGA para coleta das amostras e execução do pós processamento e dos testes de aleatoriedade.

Fonte: Autor.

Por fim, as plataformas capturam os dados gerados e os enviam para o Matlab© realizar o pós processamento e executar a bateria de testes sugeridos pelo NIST, ressaltando que esta etapa é feita em computador do tipo *Desktop*. O pós processamento dos dados é feito baseado no algoritmo de von Neumann, Figura 4.9. Esse algoritmo é usado para manter a uniformidade da sequência, isto é, a quantidade de zeros e uns tenderem a ser igualmente distribuídos ao longo da sequência.

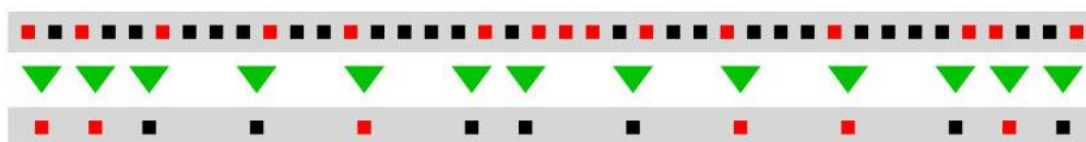


Figura 4.9 – Fluxograma da Lógica do Algoritmo de von Neumann [23].

Portanto, por meio desse algoritmo grandes sequências de zeros e uns são evitadas a um custo computacional razoável. Nota-se que os quadrados vermelhos são os bits 1 e os pretos são os bits 0 e que após esse pós processamento há uma redução do número de bits, neste caso de 30 para 13, tal redução é chamada de eficiência do algoritmo.

4.2 Resultados

Os resultados deste trabalho estão divididos em três momentos: o traçado dos sinais das fontes de entropia, a eficiência do algoritmo nas três plataformas propostas e na análise dos testes estatísticos aos quais os sistemas foram submetidos. Aqui, eles serão apenas expostos e na próxima subseção serão analisados.

4.2.1 Fontes de Entropia

A fonte de entropia baseada no efeito avalanche apresenta várias curvas a depender da tensão de alimentação do sistema que varia de um valor inicial a depender do transistor, devido ao seu processo de fabricação conter imperfeições e impurezas nos semicondutores, até o valor máximo de 18 volts. Daí, temos três tipos de curvas: efeito avalanche na saída do transistor gerador, a avalanche retificada após passar pelo diodo zener e a curva após passar pelo ST já digitalizada. Elas podem ser observadas nas Figuras 4.10, 4.11, 4.12 e 4.13, destacando que esses experimentos foram mensurados a partir da montagem feita no protoboard, conforme a Figura 4.4.

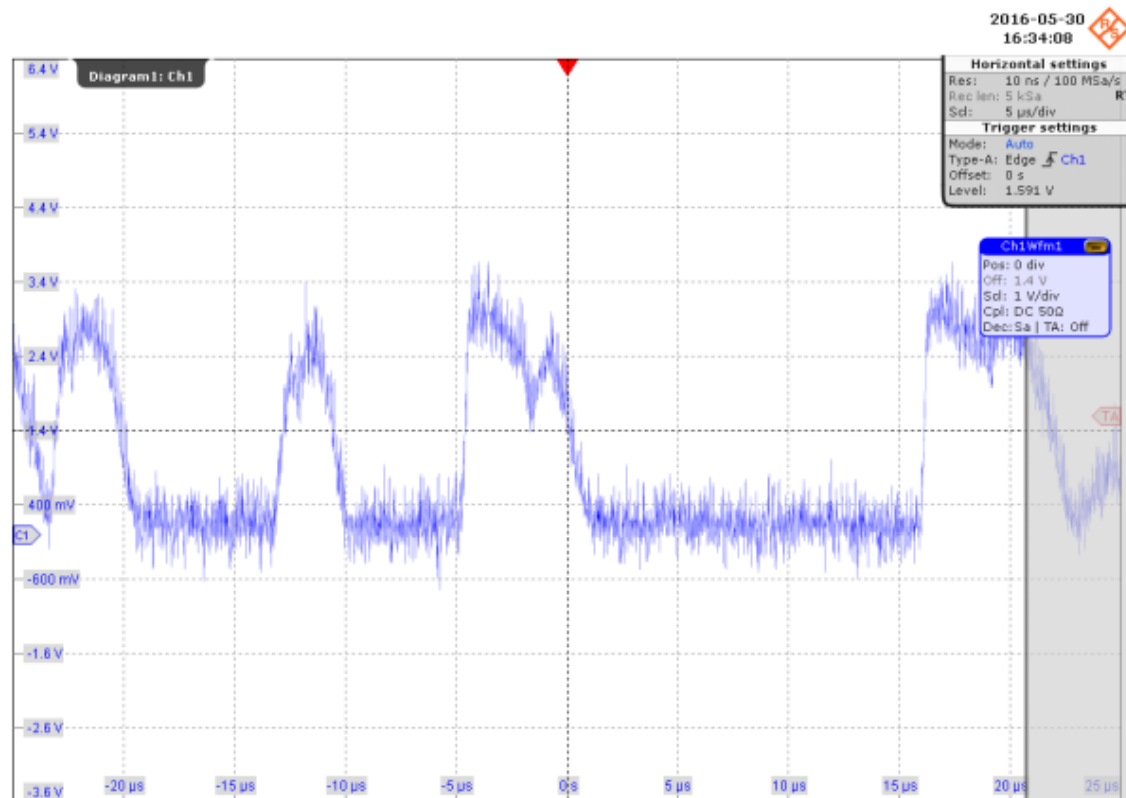


Figura 4.10 – Efeitos Avalanche a uma tensão de 18V avaliado na saída do transistor Q3.

Fonte: Autor.

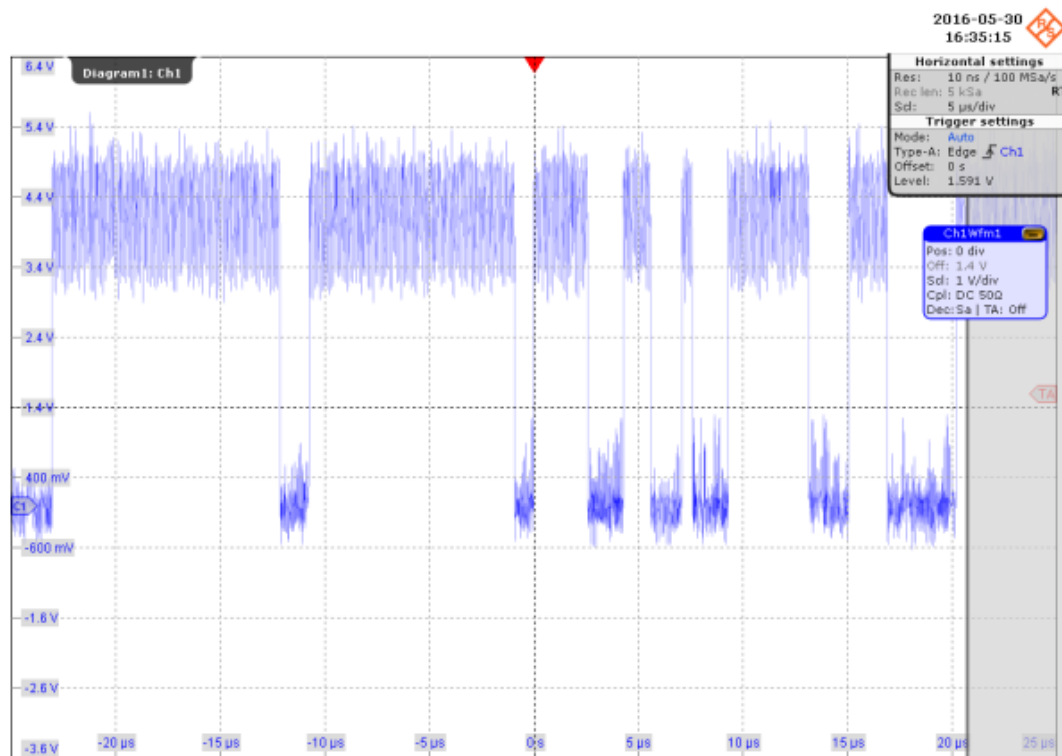


Figura 4.11 – Efeitos Avalanche a uma tensão de 18V avaliado na saída do ST, sinal digitalizado.

Fonte: Autor.

4.2.2 Eficiência do Algoritmo de Pós Processamento

A eficiência do algoritmo é a medida de quantos bits podem ser usados depois do pós processamento em relação a quantidade total gerada. Assim, as plataformas capturam os dados por meio de suas portas de entrada e saída digitais e enviam pela interface serial para um computador do tipo *Desktop* com o Matlab®.

A leitura serial do *software* divide grupos de 8 bits a cada leitura a uma taxa de transferência de 9600 bits por segundo, *baudrate*, formando uma matriz final de 512x8 bits que são depois remapeados para uma estrutura de dados de 1x4096 elementos para serem submetidos ao algoritmo de von Neumann. Com isso, podemos compor várias amostras de tamanhos distintos, no caso deste trabalho, foi adotado amostras de 4096 bits e de 1 Mb armazenadas em uma planilha para serem submetidas aos testes propostos nas três plataformas. Os resultados provenientes dessas amostras mostram que a eficiência do algoritmo de pós processamento independe do tamanho da sequência, uma vez que a amostra de 4096 tem um percentual semelhante à da amostra de um 1Mb.

4.2.3 Testes Estatísticos

Os testes estatísticos foram implementados de acordo com o pacote de teste do NIST, que contém quinze testes, porém houve dois testes que não foram implementados, o de complexidade linear (Teste 10) e o soma cumulativa (Teste 13). O primeiro testa se há realimentação no sistema e, assim, aferir correlação e o segundo testa sequências longas de zeros e uns no começo ou no final da amostra, visando obter previsibilidade. A Tabela 4.1 mostra o comportamento das plataformas após passar pelos testes.

Tabela 4.1: Comparativo das Plataformas e seu desempenho nos Testes.

Fonte: Autor

Teste Estatístico	Arduíno UNO	Arduíno DUE	FPGA
Frequência (Monobit)	SIM	SIM	SIM
Frequência de um Bloco	SIM	SIM	SIM
Teste de Corrida	NÃO	NÃO	NÃO
Maior Corrida de uns dentro de um Bloco	NÃO	NÃO	NÃO
Posto da Matriz Binária	SIM	SIM	SIM
Espectral da Transformada discreta de Fourier	SIM	NÃO	NÃO
Casamento de padrão sem superposição	SIM	SIM	SIM
Casamento de padrão com superposição	SIM	SIM	SIM
Estatística universal de Maurer	SIM	SIM	SIM
Complexidade Linear	N/I*	N/I	N/I
Serial	NÃO	NÃO	NÃO
Entropia Aproximada	NÃO	NÃO	NÃO
Somas Cumulativas	N/I	N/I	N/I
Excursões Aleatórias	SIM	SIM	SIM
Variantes de excursão aleatória	SIM	SIM	SIM

4.3 Análise dos Resultados

Assim como na parte anterior, a análise também será dividida nas três partes e seguindo a mesma linha de raciocínio anterior.

4.3.1 Fontes de Entropia

A fonte usada tem um preço bastante acessível e para mensurar os dados também não são necessários equipamentos muito sofisticados. As observações mostram que a tensão de alimentação do circuito está diretamente ligada com a ocorrência do efeito a partir de uma tensão inicial, geralmente em torno de 9,5V, que desencadeia a primeira avalanche e que os elementos eletrônicos, como a solda e a dimensão das trilhas da placa manufaturada, impactam na geração desse efeito.

O efeito sofre diretamente com o aumento da tensão de alimentação do circuito até um tensão máxima, pois se trata de um semicondutor e eles sofrem degeneração caso saturem a tensão máxima ou a temperatura máxima de operação. Outro fator é a qualidade da fonte de alimentação que pode distorcer o sinal de avalanche, para isso foram usadas duas fontes de bancada fabricantes diferentes com limiares de ruídos bem diferentes para alimentar o sistema. Isso é mais perceptível no Fotodiodo de Avalanche.

As plataformas que fazem as leituras do gerador estão programadas para fazerem leituras a cada 20 microssegundos, portanto são feitas teoricamente 5000 leituras de 8 bits no registrador de entrada e saída de cada microcontrolador. Essas leituras são limitadas pela comunicação serial feita entre a plataforma e o Matlab©. Assim, uma maneira de aprimorar este sistema é usar o conversor analógico digital para montar um sistema de comunicação e trabalhar diretamente com o sinal gerado, entretanto, o custo computacional para isso é alto e descartaria o uso do Arduíno UNO, a solução mais barata dentre as apresentadas.

Então, para esta fonte é aconselhável testar os níveis de tensão a fim de se obter a melhor sequência em consonância com a plataforma usada. Por exemplo, tanto no Arduíno DUE quanto no FPGA pode-se ter taxa maiores e, portanto, fazer mais leituras e ter mais bits para pós processar.

4.3.2 Eficiência do Algoritmo de Pós Processamento

O pós processamento é usado para evitar previsibilidade e reduzir ao máximo a correlação entre os bits ou entre um grupo de bits. O uso do algoritmo de von Neumann garante um equilíbrio entre os bits, tentando deixá-los na proporção em torno de 50% para cada valor. Como já visto no Capítulo 3, ele é bastante usado pelos desenvolvedores de gerador, pois ele possui alto rendimento e custo computacional razoável, sendo algumas vezes implementado por blocos lógicos ou embarcado no gerador.

Nesse caso, a fonte e o pós processamento por este algoritmo não foram suficientes para que o gerador obtivesse sucesso em todos os testes, logo é preciso montar uma outra estratégia, ou ainda, propor um novo algoritmo de pós processamento para a solução proposta com essa fonte. É importante destacar que o foco do algoritmo não é apenas deixá-lo favorável a passar nos testes tem-se que ter em mente a eficiência sobre a sequência que resta após essa ação, aqui obteve-se uma eficiência em torno de 48%.

É importante destacar que esse algoritmo pode implicar o êxito na aprovação em outros pacotes de testes, como o DIEHARD e o ENT. Ainda é possível que ele seja compatível no pós processamento de outras fontes de entropia não usadas neste trabalho.

4.3.3 Testes Estatísticos

Os testes feitos com 10 amostras de 4096 bits e 10 de 1 Mb para as três plataformas e os resultados estão na Tabela 4.1. Assim, em nove dos treze testes realizados as plataformas obtiveram os mesmos resultados satisfatórios. Isso significa que o equipamento para ser certificado deve ser revisto nos testes não satisfatórios, caso queira ser validado pelo NIST.

Esses resultados mostram que a fonte tem um comportamento equivalente independentemente da plataforma usada no gerador. A priori os resultados foram distintos a medida que os testes eram executados nas diferentes plataformas, porém isso ocorreu devido a um erro na implementação dos testes, que tão logo foram corrigidos. Tal fato deu-se por conta da formação da amostra usada como parâmetro de entrada nos testes.

Analisando os testes que não tiveram êxito, tem-se o teste de Corrida (Teste 3), o teste de Maior corrida de uns dentro de um bloco (Teste 4), o teste Serial (Teste 11) e o teste de Entropia Aproximada (Teste 12).

O Teste 3 avalia se os zeros e uns são como esperado em uma sequência aleatória e para isso ele avalia as oscilações é rápida ou lenta. A falha neste teste está relacionada com o algoritmo de pós processamento usado, von Neumann, pois ele não deixa formar grandes cadeias de zeros ou de uns, logo tem-se variações rápidas entre os bits. Uma solução para este teste pode ser feita com base na troca da fonte ou o uso de outro algoritmo de pós processamento.

O Teste 4 avalia se o comprimento de uns dentro de um bloco é consistente em relação a sequência de fato aleatória, o análogo ocorre com os zero, logo basta um teste. Aqui, também tem-se influência do pós processamento, uma vez que os blocos de uns serão bastante variados ao longo da sequência.

Já o Teste 11 testa faz uma bateria de procedimentos para testar a uniformidade da distribuição de padrões em um dado comprimento. Neste caso, as sequências testadas não apresentaram padrão com a função especificada, a função gama incompleto.

Por fim, o Teste 12 avalia padrões repetidos ao longo da sequência, com um procedimento análogo ao anterior. Assim, é buscado comparar dois grupamentos, um com tamanho especificado e outro com tamanho acrescido de uma unidade com distribuição definida como aleatória. Logo, por motivos análogos ao apresentado no Teste Serial, este teste também teve insucesso.

4.4 Conclusão

Assim, percebe-se que possível caracterizar um Gerador Físico de Números Aleatórios (GFNA) desde análise do ruído, passando pelo pós processamento dos dados até a etapa dos testes estatísticos. Além disso, o uso das plataformas de sistemas embarcados, tanto Microcontroladores quanto FPGAs, mostram um comportamento igual, dando ao sistema um leque maior de possibilidade de inserção nos mais diversos cenários de aplicação.

CAPÍTULO 5

O presente trabalho investigou a possibilidade de caracterizar um Gerador Físico de Números Aleatórios usando plataformas de Sistemas Embarcados. Para isso, montou-se uma prova de conceito em laboratório, cuja configuração possibilitasse capturar sinais de fontes de entropia de forma a verificar através de equipamentos de bancada uma série de medidas e testes de confiabilidade para essa caracterização, avaliação e utilização deste gerador. Assim, neste capítulo será apresentado as conclusões sobre o trabalho, bem como as perspectivas relacionadas a ele.

CONCLUSÕES E PERSPECTIVAS

5.1 Conclusões

Neste trabalho, foi possível caracterizar a fonte de entropia não determinísticas, baseada no efeito avalanche. Essa fonte pode ser explorada até a etapa de testes, onde mostrou seu potencial para aplicações de baixo custo em três plataforma para Sistemas Embarcados, duas plataformas Arduíno e FPGA.

Através dos resultados obtidos nos ensaios, comprovou-se que o algoritmo usado no pós processamento tem um comportamento uniforme na medida em que aumenta-se o tamanho da amostra e apresenta esse comportamento nas três plataformas usadas e esse valor é tido como satisfatório devido a seu custo computacional.

A implementação do pós processamento e dos testes no ambiente fora dos Sistemas Embarcados foi para evitar problemas relacionados as restrições de processamento e memória destas plataformas. Agora, com real dimensionamento tanto das operações quanto da complexidade dos algoritmos de testes e pós processamento é viável começar a embarcar, também, essa parte de *software* e tornar o gerador ainda mais independente.

Com relação aos testes, esse gerador teve êxito em nove dos treze testes ao qual foi submetido, ressaltando que isso já foi após detecção e correção de falhas até chegar nesta quantidade. Na literatura, essa fonte tem sucesso num outro pacote de testes, o ENT,

contudo o autor do gerador T12RNG cita modificações no algoritmo para implicar a validação nesta fase.

É importante destacar que o uso do FPGA permite que o trabalho tenha um *hardware* próprio para funcionar como gerador, dando maior grau de inovação, pois assim ele poderá integrar qualquer parte de um sistema de comunicação cujo bloco de geração aleatória é fruto deste trabalho.

Por fim, o gerador caracterizado apresenta pelo efeito avalanche e com a plataformas Arduíno UNO tem baixo custo e eficiência para um determinado grupo de aplicação no estado que finaliza este trabalho. Estendendo o êxito para os quinze testes, ele poderá ser usado em quaisquer cenários onde sejam requisitados a natureza aleatória da sequência gerada.

5.2 Perspectivas para novos trabalhos

Como desdobramentos deste trabalhos, apresentamos as seguintes propostas:

1. Buscar o êxito nos testes no qual se tem falhado, Testes 3, 4, 11 e 12;
2. Melhorar e propor novos algoritmos de pós processamento das sequências geradas;
3. Implementar o código de pós processamento de maneira a embarcar o código nas plataformas usadas;
4. Testar as sequências geradas por esta solução nos outros pacotes de testes exposto neste trabalho, o ENT e o DIEHARD.

5.3 Trabalhos decorrentes desta dissertação

- Apêndice A: Projetos de Iniciação Científica Aprovados;
- Apêndice B: Orientações de Trabalho de Conclusão de Curso (TCC);
- Apêndice C: Projeto de Placa;
- Apêndice D: Blocos Lógicos Implementados.

REFERÊNCIAS

- [1] ARDUÍNO. **Arduíno**. Disponível em: <<http://Arduíno.cc>> Acesso em: 10 jan. 2016.
- [2] BÖHL, Eberhard. **Simple true random number generator for any semi-conductor technology**. *IET Computers & Digital Techniques*, 2014. p. 239-245.
- [3] BROWN, R. G. **Dieharder: A Random Number Test Suite**. Disponível em: <<https://www.phy.duke.edu/~rgb/General/dieharder.php>> Acessado em: 15 janeiro 2016.
- [4] BOYLESTAD, Robert L. & NASHELSKY, Louis. **Dispositivos Eletrônicos e Teoria de Circuitos**. 11ª ed. São Paulo: 2013. 766p, *il*.
- [5] COSTA, E. B. G. **Ataques Quânticos a Geradores de Números Pseudo-Aleatórios**. Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, Programa de Pós-Graduação em Ciência da Computação. Campina Grande/PB 2011, 141p, *il*. Dissertação (Mestrado).
- [6] DANGER *et al.* **High speed true random number generator based on open loop structures in FPGAs**. *Microelectronics Journal* 40 (2009). p. 1650–1656.
- [7] FIORENTINO, *et al.* **All-fiber-optic quantum random number generator**. *Conference on Lasers and Electro-Optics and Quantum Electronics and Laser Science Conference*, 2006. p. 1-2.
- [8] CHINDRIS, GABRIEL; MURESAN, MARIUS. **Bipolar Junction Effects for High Entropy Data Harvesters**. *Proceedings 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, 2008. p. 449-452.
- [9] GRIFFITHS, David J. **Mecânica Quântica 2e**. 2ª Ed. São Paulo: Pearson, 2011. 350p, *il*.
- [10] GUTTERMAN *et al.* **Analysis of the Linux Random Number Generator**. *Proceedings IEEE Symposium on Security and Privacy*. 2006. p. 385-400.
- [11] HAYKIN, Simon. **Sistemas de Comunicação, analógicos e digitais**. 4ª ed. Porto Alegre: Bookman, 2007. 837p, *il*.

- [12] HONG, S. LEE & LIU, CHANG. **Sensor-Based Random Number Generator Seeding**. *IEEE Access* vol. 3 (2015) p. 562-568.
- [13] IDQ. **ID Quantique with paper: Random Number Generation using quantum physics versão 3.0**. Disponível em:
<<http://www.idquantique.com/images/stories/PDF/quantisrandom-generator/quantis-whitepaper.pdf>>. Acesso em: 07 julho. 2015.
- [14] IWANSSON, K. *et al.* **Handbook of Sensor and Actuators: Measuring Currente, Voltage and Power**. 1ª ed. Elservier, 1999. 268p, *il.*
- [15] JUSTO, M. J. M. **Autômatos Celulares Caóticos aplicados na Criptografia e Criptoanálise**. Universidade de São Carlos, Instituto de Física de São Carlos, Programa de Pós-Graduação em Física Aplicada. São Carlos/SP, 2013, 121p, *il.* Dissertação (Mestrado).
- [16] KENNY, C. **Random Number Generators: An Evaluation and Comparison of Random.org and Some Commonly Used Generators**. April 2005. Trinity College Dublin.
- [17] KNUTH, D. E. **The art of computer programming**, 2nd ed: Adisson Wesley, vol. 2, 1981.
- [18] L'ECUYER, P. **Combined multiple recursive generators**. *Annals of Operations Research* vol. 44, nº 5, 1996. p. 816-822.
- [19] L'ECUYER, P. **Uniform random number generation**. *Annals of Operations Research*, vol. 53, 1994. p. 77-120.
- [20] LEÃO, A. F. **Abordagem Teórico – Experimental do Efeito do Tunelamento Quântico em Dispositivos Semicondutores**. Universidade Federal do Vale do São Francisco, Programa de Pós-Graduação em Ciências dos Materiais. Juazeiro/BA, 2009, 136p, *il.* Dissertação (Mestrado).
- [21] LEVI, A. F. J. **Applied Quantum Mechanics**. 2nd ed. Cambridge: 2006. p. 558 *il.*
- [22] LIMA, C. B. & VILLAÇA, M. V. M. **AVR e Arduíno: Técnicas de Projeto**. 2ª ed. Clube de Autores: Joinville/SC, 2012. p. 632 *il.*
- [23] LOGUE, Aaron & PLATT, Charles. **Really, Really Random Number Generator**, *Make Magazine*, USA: 2015.
- [24] MARGOLIS, Michael. **Arduíno Cookbook**. 2nd. ed. USA: O' Reilly 2013. 632p, *il.*

- [25] MARWEDEL, Peter. ***Embedded Systems Design***. Springer: Netherlands, 2006.
- [26] MONK, S. **Programação com Arduino: começando com Sketches**. Porto Alegre: Bookman, 2013. 160 p. (Série Tekne).
- [27] NIE *et al.* **68 Gbps quantum random number generation by measuring laser phase fluctuations**. *Review of Scientific Instruments* 86, 2015. p. 1-15.
- [28] NIST. **Natl. Inst. Stand. Technol. Spec. Publ. 800-22rev1a, 131 pages (April 2010)**. Disponível em: <csrc.nist.gov/publications/nistpubs/800-22-rev1a/SP800-22rev1a.pdf>. Acesso em: 01 setembro 2015.
- [29] NOBLE, Joshua. **Programming Interactivity**. 1st ed. USA: O' Reilly 2009. 713p, *il.*
- [30] OLIVEIRA, A. S; Andrade, F. S. **Sistemas Embarcados Hardware e Firmware na Prática**. 1^a ed. São Paulo: Erika, 2006. 315p, *il.*
- [31] OLIVEIRA, K. D. **Desenvolvimento de Sistema Embarcado para Contagem de Fluxo de Pessoas em Atração de Parque Aquático Cearense**. Instituto Federal de Educação, Ciência e Tecnologia do Ceará, Departamento de Telemática. Fortaleza/CE, 2014, 28p, *il.* Trabalho de Conclusão de Curso de Graduação (Monografia).
- [32] ORDONEZ, E. D. M; PENTEADO, C. G; SILVA, A. C. R. **Microcontroladores e FPGAs Aplicações em Automação**. 1^a ed. São Paulo: Novatec, 2005. 378p, *il.*
- [33] PATENTS. **Google Patents**. Disponível em: < <https://www.google.ch/patents/>> Acessado em: 23 de janeiro de 2016.
- [34] PRESS, W. H. *et. al.* **Numerical Recipes in C – The Art of Scientific Computing**. Cambridge University Press, 2nd ed. 1992.
- [35] SANGUINETTI *et al.* **Quantum random number generation on a mobile phone**, *Physics Reviews X* 4, 2014. p. 1-5.
- [36] SCHNEIER, B. **Applied Cryptography: Protocols, Algorithms, and Source Code in C**. John Wiley & Sons, New York, 1996.
- [37] SIGNH, Pritpal. **ECE 3590 Lecture 10 Summer 1997**. Disponível em: <<http://www.ece.villanova.edu/~singh/lec10/sld001.html>> Acessado em: 30 de janeiro de 2016.

- [38] SOARES, E. J. L; **Geradores quanto-ópticos de números aleatórios**. Universidade Federal do Ceará, Centro de Tecnologia, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia de Teleinformática. Fortaleza/CE, 2013. 54p, *il.* Dissertação (Mestrado).
- [39] SOARES, *et al.* **Quantum Random Number Generator Using Only One Single-Photon Detector**. *IEEE Photonics Technology Letters* vol. 26, 2014. p. 851-853.
- [40] STIPČEVIĆ, Mario. **Quantum random number generators and their use in cryptography**. *Proceedings of the 34th International Convention MIPRO*. Croatia: Opatija, 2011. p. 1474-1479.
- [41] T12RNG. **T12RNG is a very low cost, simple and compact hardware random number generator (HRNG)**. Disponível em:
<<http://www.postcogito.org/Kiko/T12RandomNumberGenerator.html>> Acessado em: 30 de junho de 2015.
- [42] WALKER, J. **Ent: A Pseudorandom Number Sequence Test Program**, Disponível em: <<http://www.fourmilab.ch/random/>> Acessado em: 10 jan. 2016.
- [43] WEI *et al.* **High-Speed and Bias-Free Optical Random Number Generator**, *IEEE Photonics Technology Letters* vol. 24, 2015. p. 437-439.
- [44] XILINX. **Spartan-3E FPGA Starter Kit Board User Guide**. Disponível em:
<http://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf> Acessado em: 09 de julho de 2015.
- [45] ZANG *et al.* **Portable true random number generator for personal encryption application based on smartphone camera**. *Electronics Letters* vol. 50, 2014. p. 1841-1843.
- [46] ZEGHBROECK, B. **Principles of Semiconductor Devices**. Disponível em:
<<http://ecee.colorado.edu/~bart/book/book/title.htm>> Acessado em: 02 de fevereiro de 2016.

APÊNDICE

APÊNDICE A – Projetos de Iniciação Científica Aprovados.

- **PROJETO: Estudo de Efeito Peltier para Resfriamento de Fotodiodos Avalanche para Aplicações em Detectores de Fótons Únicos. EDITAL N° 01/2015 – PRPI;**
- **PROJETO: Gerador de Números Aleatórios Baseado em FPGA. EDITAL N° 04/2014 – PRPI;**
- **PROJETO: Gerador Quântico de Números Aleatórios Usando Plataforma ARM Cortex M3. EDITAL N° 09/2013 – PRPI.**

APÊNDICE B – Orientações de Trabalho de Conclusão de Curso (TCC).

- Adriano Leitão Vaz. **Criptografia Quântica - Revisão da Literatura.** 2015. Trabalho de Conclusão de Curso. (Graduação em Engenharia de Telecomunicações) - Instituto Federal do Ceará - Reitoria. Orientador: Jorge Fredericson de Macedo Costa da Silva.
- Bertrand Wadi Kiasolwa. **Aplicações de FPGA em Engenharia de Telecomunicações.** Orientação em Andamento. Trabalho de Conclusão de Curso. (Graduação em Engenharia de Telecomunicações) - Instituto Federal do Ceará - Reitoria. Orientador: Jorge Fredericson de Macedo Costa da Silva.

APÊNDICE C – Projeto de Placas

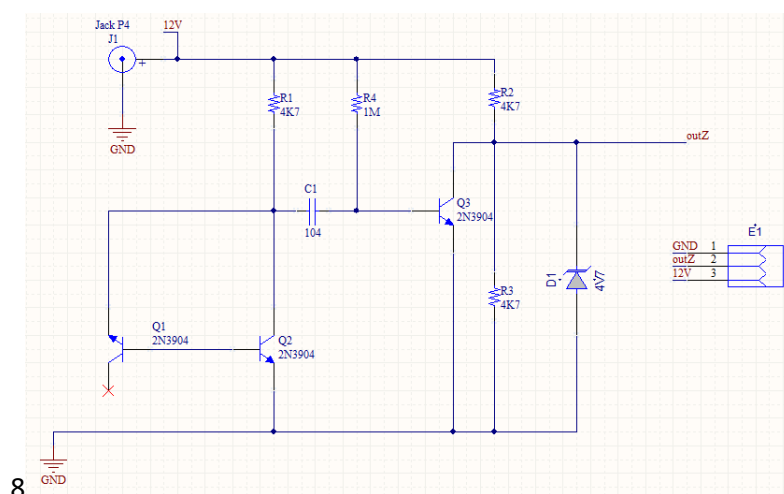


Figura C.1 – Esquemático da Placa de Aquisição do Sinal de Avalanche.

Fonte: Autor

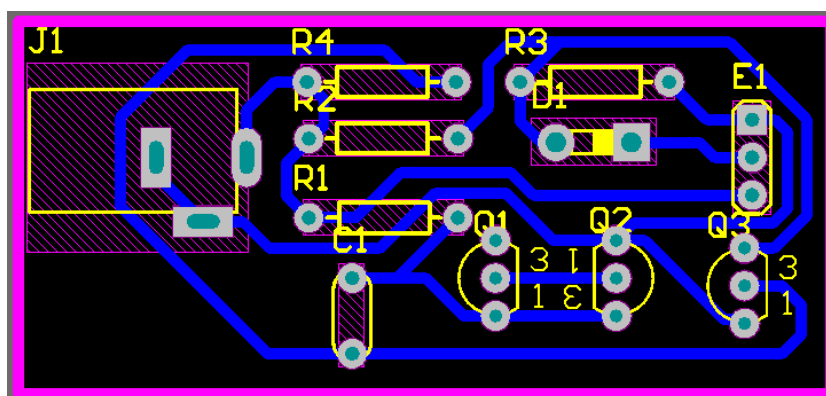


Figura C.2 – Roteamento da Placa de Aquisição do Sinal de Avalanche.

Fonte: Autor

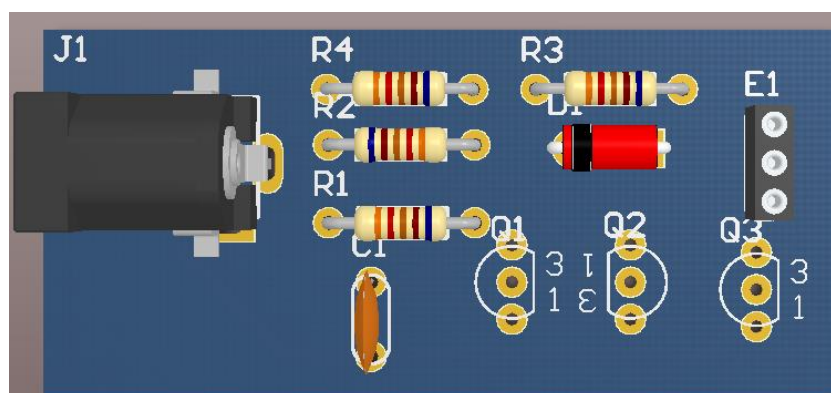


Figura C.3 – Vista 3D da Placa Manufaturada.

Fonte: Autor

APÊNDICE D – Blocos Lógicos Implementados

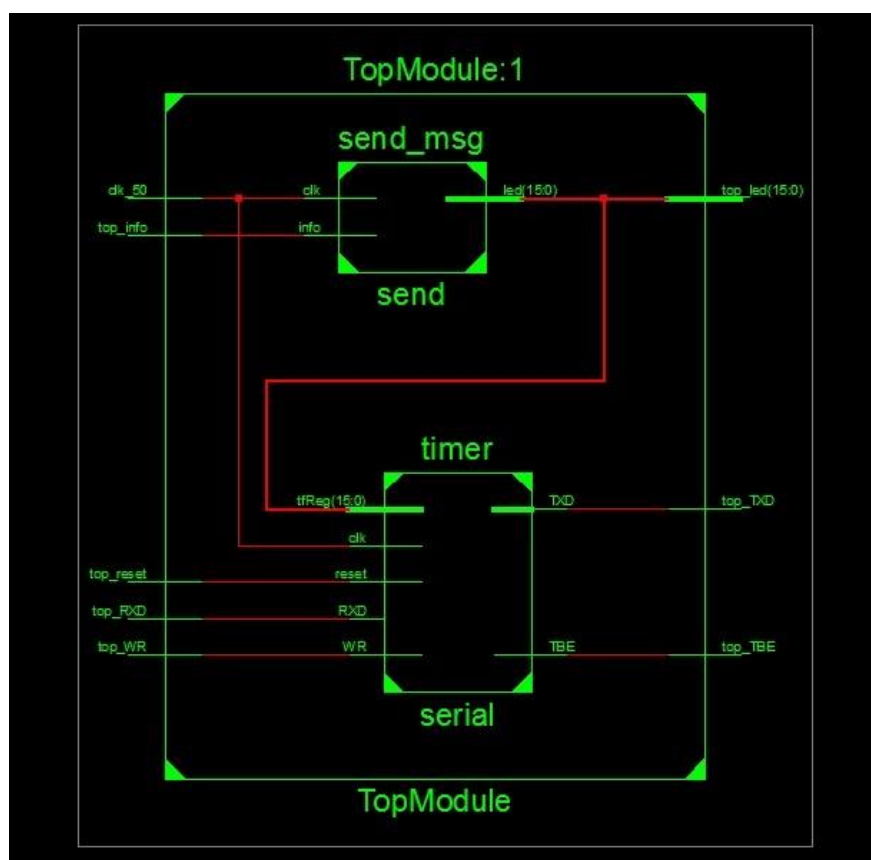


Figura D.1 – Componente que recebe o vetor do GFNA e componente de comunicação Serial.

Fonte: Autor

ANEXOS

ANEXO A – Algoritmo de von Neumann

John von Neumann mostrou que uma série de aleatória de estados que estão desigualmente ponderados pode ser convertido numa saída uniforme ponderada por amostragem, os estados em pares, ignorando pares idênticos e copiando o primeiro estado em cada par desigual. Assim, a partir de um fluxo de 0s e 1s que não é uniformemente ponderado, os estado em pares não sobrepostos devem ser substituídos, logo onde tem-se dois dígitos iguais é possível ignorá-los e “seguir” a diante. Com isso, onde há bit 1 seguido por um bit 0, cria-se um bit 1, caso contrário bit 0 (LOGUE & PLATT, 2015).

Isso pode ser demonstrado usando uma analogia com uma caixa com bolas, sendo 25% com bolas pretas e 75% de bolas vermelhas. Logo, a probabilidade de se escolher uma bola vermelha ou uma preta aleatoriamente é igual a sua quantidade ($1/4$ ou $3/4$). E a probabilidade de se retirar um par vermelho-preto é igual a de se tirar um par preto-vermelho ($3/16$). Isso continua a ser verdade mesmo que não se saiba a razão vermelho-preto, desde que as amostras permaneçam coerentes. O processo é um pouco ineficiente, mas o resultado funciona adequadamente (LOGUE & PLATT, 2015). A Figura 4.9 ilustra este processo.

ANEXO B – Distribuições de Probabilidades Específicas

Este anexo é destinado a mostrar as distribuições probabilísticas usadas nos testes estatísticos. Os testes pautam-se em basicamente três funções: normal, função erro complementar e função gama incompleto. Assim, será dado enfoque nas equações e uma breve explicação sobre cada uma delas.

B.1 – Distribuição Normal

A distribuição normal apresenta dois parâmetros: a média (μ) e a variância (σ^2). A densidade de probabilidade é dada pela Equação B.1.1, a seguir:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (\text{B.1.1})$$

B.2 – Função Erro Complementar

A função erro complementar foi criada para calcular a integral da distribuição normal. Sua formulação matemática é dada pela Equação B.2.1:

$$\text{erfc}(z) = \frac{2}{\sqrt{\pi}} \int_z^{\infty} \exp(-u^2) du \quad (\text{B.2.1})$$

B.3 – Função Gama Incompleto

A função gama incompleto é uma variação da função gama completa, Equação B.3.1, definida por uma parte integrante com um limite superior finito em vez de infinito. As Equações B.3.2 mostra as aproximações usadas no cálculo dela:

$$\Gamma(z) = \int_0^{\infty} t^{z-1} \exp(-t) dt \quad (\text{B.3.1})$$

$$P(a, x) = \frac{\gamma(a, x)}{\Gamma(a)} = \frac{1}{\Gamma(a)} \int_0^x \exp(-t) t^{a-1} dt \quad (\text{B.3.2})$$

Em que o limite de integração é dado pela seguinte análise: $P(a, 0) = 0$ e $P(a, \infty) = 1$. O mesmo vale para o seu complementar, ou seja, $Q(a, x) = 1 - P(a, x)$: $Q(a, 0) = 1$ e $Q(a, \infty) = 0$.

ANEXO C – Testes Estatísticos sugeridos pelo NIST

Nesta secção será abordado os testes estatísticos usados. Conforme dito no Capítulo 2, o trabalho norteou-se pelo pacote disponibilizados pelo NIST, que é composto por quinze testes para validação das sequências binárias geradas e que são ditas aleatórias. Logo, teremos a descrição deles e suas formulações matemáticas, lembrando que em todos a regra de decisão tem nível de 1% para definir se a amostra é ou não aleatória para aquele teste. Ressaltando que todas as definições e equações foram retiradas do documento editado pelo NIST [28].

C.1 - Teste de Frequência (Monobit)

O objetivo deste teste é estimar a proporção de bits zero e um da sequência, pois é esperado numa sequência realmente aleatória o mesmo número de cada bit, ou seja 50% de bits zeros e 50% de bits uns. Com isso, o teste avalia a proximidade da fração entre o número de bits do valor $\frac{1}{2}$, ou seja, quantidades iguais. Para efeito de cálculo, os bits zero são “transformados” em -1 e, então, é feito uma soma com os bits +1, esperando um resultado em torno do valor zero. Portanto, analisando o tamanho da amostra e o valor desta soma é possível concluir se a sequência se comporta ou não dentro do esperado.

Ressaltando que o recomendado para este teste é uma amostra de no mínimo 100 bits. As equações C.1.1 e C.2.1 mostram como é computado o resultado do teste.

$$S_{obs} = \frac{|S_n|}{\sqrt{n}} \quad (C.1.1)$$

$$P - value = \text{erfc}\left(\frac{S_{obs}}{\sqrt{2}}\right) \quad (C.1.2)$$

Em que S_{obs} é a estatística observada, S_n é a soma dos valores das amostras, n é o número de bits da amostra, $P - value$ é a probabilidade definida pela função erro complementar, $\text{erfc}()$.

C.2 - Teste da Frequência em um Bloco

Neste teste, o objetivo é determinar se a frequência de bits um num bloco de M bits é aproximadamente $M/2$, assumindo a aleatoriedade. A sequência é dividida em blocos de tamanhos iguais e em cada um dos blocos a proporção de 1s e 0s é verificada da mesma maneira que no teste anterior, o mesmo ocorre se M for de tamanho unitário.

O teste depende do tamanho da sequência, do tamanho dos blocos e dos níveis de tolerância esperados.

Como no teste anterior, o tamanho da amostra recomendada é de no mínimo 100 bits. No caso da divisão não ser exata, os bits não usados, que “sobram” da divisão, são descartados. A formulação do teste segue as equações C2.1, C2.2, C.2.3 E C.2.4:

$$N = \left\lfloor \frac{n}{M} \right\rfloor \quad (C.2.1)$$

$$\pi_i = \frac{\sum_{j=1}^M \mathcal{E}_{(i-1)M+j}}{M} \quad (C.2.2)$$

$$X^2_{(obs)} = 4 \times M \times \sum_{i=1}^N (\pi_i - 1/2)^2 \quad (C.2.3)$$

$$P-value = igamc(N/2, X^2_{(obs)}/2) \quad (C.2.4)$$

Em que $X^2_{(obs)}$ é a estatística observada, N é o total de blocos, n é o número de bits da amostra, π_i é proporção de 1s dentro de cada bloco M , $P-value$ é a probabilidade definida pela função gama incompleto, $igamc()$.

C.3 - Teste de Corrida

É considerada corrida uma sequência de bits idênticos, logo o objetivo deste teste é verificar o número total de corridas em uma amostra, verificando se as oscilações entre bits zero e um é rápida ou lenta. Com isso, sequências de bits iguais são possíveis, mas quanto maiores e mais frequentes, podem indicar alguma falha em potencial do gerador.

Aqui, o tamanho mínimo recomendado da amostra também é de 100 bits. As formulações são descritas a seguir pelas equações C.3.1, C.3.2 e C.3.3:

$$\pi = \frac{\sum_j \mathcal{E}_j}{n} \quad (C.3.1)$$

$$V_{n(obs)} = \sum_{k=1}^{n-1} r(k) + 1 \quad (C.3.2)$$

$$P-value = \text{erfc}\left(\frac{|V_{n(obs)} - 2n\pi(1-\pi)|}{2\sqrt{2n\pi(1-\pi)}}\right) \quad (C.3.3)$$

Em que $V_{n(obs)}$ é a estatística baseada na função $r(k)$, que assume valor 0 se $\varepsilon_k = \varepsilon_{k+1}$ e valor 1 caso contrário, n é o número de bits da amostra, π é proporção de 1s da amostra, $P-value$ é a probabilidade definida pela função erro complementar, $erfc()$.

C.4 - Teste para a Maior Corrida de 1s dentro de um Bloco

O objetivo deste teste é aferir a mais longa corrida de 1s dentro de blocos de M bits. Assim, procura-se determinar se o comprimento da corrida mais longa dos bits um dentro da sequência testada é consistente com o comprimento da corrida mais longa dos 1s que seria esperados em uma sequência aleatória. A diferença entre este teste e o anterior é que ao invés de contar as oscilações entre 0s e 1s, este teste conta o tamanho máximo da sequência de 1s em um determinado bloco de tamanho conhecido e o compara com o encontrado em outros blocos.

O tamanho mínimo da amostra é mostrado na Tabela C.4.1 e as formulações com base nas equações C.4.1 e C.4.2:

Tabela C.4.1 – Tamanho mínimo das amostras em relação ao tamanho do bloco [28].

n Mínimo	M
128	8
6272	128
750000	10^4

$$X^2_{(obs)} = \sum_{i=0}^K \frac{(v_i - N\pi_i)^2}{N\pi_i} \quad (C.4.1)$$

$$P-value = igamc\left(\frac{K}{2}, \frac{X^2_{(obs)}}{2}\right)$$

Em que $X^2_{(obs)}$ é a estatística observada, n é o número de bits da amostra, π_i é proporção de 1s dentro de cada bloco M , $P-value$ é a probabilidade definida pela função gama incompleto, $igamc()$, o valor de K e N são definidos de acordo com a Tabela C.4.2.

Tabela C.4.2 – Relação entre os valor M, K e N [28].

M	K	N
8	3	16
128	5	49
10 ⁴	6	75

C.5 - Teste de Posto da Matriz Binária

Esse teste foca na classificação das matrizes adjuntas da sequência de entrada e checka se há dependência entre as subsequências de tamanho fixo da sequência original. As equações deste teste são descritas a seguir:

$$N = \left\lfloor \frac{n}{MQ} \right\rfloor \quad (\text{C.5.1})$$

$$X^2_{(obs)} = \frac{(F_M - 0,2888N)^2}{0,2888N} + \frac{(F_{M-1} - 0,5776N)^2}{0,5776N} + \frac{(N - F_M - F_{M-1} - 0,1336N)^2}{0,1336N} \quad (\text{C.5.2})$$

$$P - value = \exp\left(-X^2_{(obs)} / 2\right) \quad (\text{C.5.3})$$

Em que $X^2_{(obs)}$ é a estatística observada, n é o número de bits da amostra, Q é o número de colunas em cada matriz adjunta, M é o tamanho do bloco, $P - value$ é a probabilidade, F_M é uma função atrelada ao cálculo do ranking. O recomendado para as equações C.5.1-3 são amostras cuja Q possa ser no mínimo 32.

C.6 - Teste Espectral da Transformada Discreta de Fourier

Este teste tem como objetivo detectar padrões repetitivos de valores que apareçam próximos uns dos outros. Estes padrões aparecem como picos no gráfico de uma transformada de Fourier discreta de uma sequência. Com base nisso, o teste identifica dois detalhes que descartariam a sequência: se há muitos picos, logo há muito padrões repetitivos e se existem poucos picos, assim indicando que a sequência não pode ser considerada aleatória.

Aqui, o tamanho mínimo da amostra de ser de 1000 bits. As equações C.6.1-5 mostram a formulação matemática:

$$M = |S'| \quad (C.6.1)$$

$$T = \sqrt{\left(\log\left(\frac{1}{0,05}\right)n\right)} \quad (C.6.2)$$

$$N_0 = \frac{0,95n}{2} \quad (C.6.3)$$

$$d = \frac{(N_1 - N_0)}{\sqrt{\frac{n(0,95) \times (0,05)}{4}}} \quad (C.6.4)$$

$$P - value = \operatorname{erfc}\left(\frac{|d|}{\sqrt{2}}\right) \quad (C.6.5)$$

Em que n é o número de bits da amostra, S' é a substring que contém os elementos a serem analisados espectralmente, T é o limiar de pico, N_0 é a expectativa teórica do número de picos, d é a diferença normalizada entre o observado e o limiar de pico e P -value é a probabilidade definida pela função erro complementar, $\operatorname{erfc}()$.

C.7 - Teste de Casamento de Padrão sem Superposição

O objetivo deste é o número de ocorrências de sequências com alvo pré-especificado. É utilizada uma janela de m bits para buscar um padrão de m bits específicos. Se o padrão não for encontrado, a janela desliza uma posição bit. Caso o padrão seja encontrado, a janela é redefinida para o bit após o padrão encontrado e a busca recomeça.

A formulação matemática é dada pelas equações C.7.1-4:

$$\mu = \frac{(M - m + 1)}{2^m} \quad (C.7.1)$$

$$\sigma = M \left(\frac{1}{2^m} - \frac{2m-1}{2^{2m}} \right) \quad (C.7.2)$$

$$X^2_{(obs)} = \sum_{j=1}^N \frac{(W_j - \mu)^2}{\sigma^2} \quad (C.7.3)$$

$$P - value = \operatorname{igamc}\left(\frac{N}{2}, \frac{X^2_{(obs)}}{2}\right) \quad (C.7.4)$$

Em que $X^2_{(obs)}$ é a estatística observada, m é o tamanho da janela, σ é a variância, μ é a média, M é o tamanho do bloco, P -value é a probabilidade, função gama incompleto.

C.8 - Teste de Casamento de Padrão com Superposição

O objetivo deste teste é o número de alvos pré-especificados nas sequências. Sua diferença em relação ao teste anterior é quando um padrão for encontrado a janela desliza somente um bit antes de voltar a busca. A formulação matemática está nas equações C.8.1-4:

$$\lambda = \frac{(M - m + 1)}{2^m} \quad (C.8.1)$$

$$\eta = \frac{\lambda}{2} \quad (C.8.2)$$

$$X^2_{(obs)} = \sum_{i=0}^5 \frac{(v_i - N\pi_i)}{N\pi_i} \quad (C.8.3)$$

$$P - value = igamc\left(\frac{5}{2}, \frac{X^2_{(obs)}}{2}\right) \quad (C.8.4)$$

Em que $X^2_{(obs)}$ é a estatística observada, m é o tamanho da janela, n é o tamanho dos bits na string, N é o número de blocos independente de M , M é o tamanho do bloco, $P-value$ é a probabilidade, função gama incompleto.

C.9 - Teste de Estatística Universal de Maurer

O objetivo deste teste é a detecção, ou não, da capacidade da sequência ser comprimida sem perda de informação. Entende-se por compressão, a possibilidade de existir uma sequência menor capaz de definir por completo a sequência original. É importante destacar que uma sequência compressível é considerada não aleatória. A Figura C.9.1 mostra como devem ser divididas as amostras para realização do teste. Já as equações C.9.1-3 mostram os valores a serem computados.

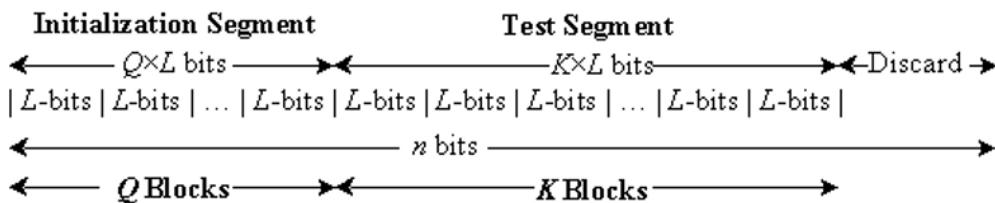


Figura C.9.1 – Divisão das amostras em blocos para teste [28].

$$f_n = \frac{1}{K} \sum_{i=Q+1}^{Q+K} \log_2(i - T_j) \quad (C.9.1)$$

$$P - value = \text{erfc} \left(\left| \frac{f_n - \text{expectedValue}(L)}{\sqrt{2}\sigma} \right| \right) \quad (C.9.2)$$

$$\sigma = c \sqrt{\frac{\text{variance}(L)}{K}} \quad (C.9.3)$$

Em que T_j é uma tabela de entrada correspondente a representação decimal do i^{th} bloco L -bit, f_n é a soma dos números de dígitos distantes entre os *templates* L , Q é número de blocos inicial da sequência, P -value é a probabilidade definida pela função erro complementar, $\text{erfc}()$.

C.10 - Teste de Complexidade Linear

O objetivo deste teste é determinar o tamanho de um registrador de deslocamento de realimentação linear (LFSR). O teste busca determinar se a sequência é ou não complexa o suficiente para ser considerada aleatória. Para isso, divide-se a sequência em blocos de tamanho determinado e usa o algoritmo de Berlekamp-Massey para encontrar o valor mínimo de bits que geram todos os valores deste bloco. Valores pequenos indicam baixa complexidade linear, logo um algoritmo potencialmente fraco para a maioria das aplicações. As equações C.10.1-5 ilustram os passos matemáticos do teste:

$$n = MN \quad (C.10.1)$$

$$\mu = \frac{M}{2} + \frac{(9 + (-1)^{M+1})}{36} + \frac{(M/3 + 2/9)}{2^M} \quad (C.10.2)$$

$$T_i = (-1)^M \cdot (L_i - \mu) + 2/9 \quad (C.10.3)$$

$$X^2_{(obs)} = \sum_{i=0}^K \frac{(v_i - N\pi_i)}{N\pi_i} \quad (C.10.4)$$

$$P - value = \text{igamc} \left(\frac{K}{2}, \frac{X^2_{(obs)}}{2} \right) \quad (C.10.5)$$

Em que $X^2_{(obs)}$ é a estatística observada, n é o tamanho dos bits na string, P -value é a probabilidade, função gama incompleto.

C.11 - Teste Serial

O objetivo é testar todas as possíveis combinações dos termos que aparecem em sequência e comparar suas ocorrências com a distribuição probabilística esperada. Nas equações C.11.1-5, pode-se observar os passos deste teste:

$$\psi^2_m = \frac{2^m}{n} \sum_{i_1 \dots i_m} \left(v_{i_1 \dots i_m} - \frac{n}{2^m} \right)^2 = \frac{2^m}{n} \sum_{i_1 \dots i_m} v_{i_1 \dots i_m}^2 - n \quad (C.11.1)$$

$$\nabla \psi^2_m = \psi^2_m - \psi^2_{m-1} \quad (C.11.2)$$

$$\nabla^2 \psi^2_m = \psi^2_m - 2\psi^2_{m-1} + \psi^2_{m-2} \quad (C.11.3)$$

$$P - value1 = igamc(2^{m-2}, \nabla \psi^2_m) \quad (C.11.4)$$

$$P - value2 = igamc(2^{m-3}, \nabla^2 \psi^2_m) \quad (C.11.5)$$

Em que ψ é a estatística observada, n é o tamanho da amostra, m é o tamanho dos blocos, $P-value1$ e $P-value2$ é a probabilidade, função gama incompleto.

C.12 - Teste da Entropia Aproximada

Semelhante ao teste anterior, o teste trabalha comparando dois tamanhos de agrupamentos diferentes, um com M bits e outro com $M+1$ como o esperado em uma sequência aleatória. O resultado não difere bastante do teste serial, exceto no fato em que dois agrupamentos são testados e seus resultados utilizados para gerar um único resultado. As equações C.12.1-4 mostram os cálculos do teste:

$$C_i^m = \frac{\#i}{n} \quad (C.12.1)$$

$$\varphi^{(m)} = \sum_{i=0}^{2^m-1} \pi_i \log \pi_i, \text{ onde: } \pi_i = C_j^3; j = \log_2 i \quad (C.12.2)$$

$$X^2 = 2n[\log 2 - ApEn(m)], \text{ onde } ApEn(m) = \varphi^{(m)} - \varphi^{(m+1)} \quad (C.12.3)$$

$$P - value = igamc\left(2^{m-1}, \frac{X^2}{2}\right) \quad (C.12.4)$$

Em que X^2 é a estatística observada, n é o tamanho da amostra, m é o tamanho do bloco, $P-value$ é a probabilidade, função gama incompleto.

C.13 - Teste das Somas Cumulativas

O objetivo deste teste é a excursão máxima de zeros do passeio aleatório definido pela cumulativa das somas ajustadas (-1, +1) na sequência. Esta soma cumulativa pode ser considerada aleatória. Dada uma sequência aleatória, as excursões do passeio aleatório deve estar próxima de zero. As equações C.13.1 e C.13.2 mostram os passos do teste:

$$z = \max_{1 \leq k \leq n} (S_k) \quad (C.13.1)$$

$$P-value = 1 - \sum_{k=\left(\frac{-n}{z}+1\right)/4}^{\left(\frac{n-1}{2}\right)/4} \left[\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right) - \Phi\left(\frac{(4k-1)z}{\sqrt{n}}\right) \right] + \sum_{k=\left(\frac{-n}{z}-3\right)/4}^{\left(\frac{n-1}{z}\right)/4} \left[\Phi\left(\frac{(4k+3)z}{\sqrt{n}}\right) - \Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right) \right] \quad (C.13.2)$$

Em que S é a soma parcial, Φ é distribuição normal de probabilidade, $P-value$ é a probabilidade, função normal.

C.14 - Teste de Excursões Aleatórias

O objetivo deste teste é determinar o número de ciclos existentes em K visitas dentro de uma sequência e verificar se este número é consistente com sua probabilidade. Considere um ciclo um caminho completo até retornar ao ponto inicial. Assim, o teste busca determinar se o número de visitas a um determinado estado dentro de um ciclo se desvia daquilo que se poderia esperar de uma sequência aleatória. As equações C.14.1 e C.14.2 ilustram esse teste:

$$X^2_{(obs)} = \sum_{k=0}^5 \frac{(V_k(x) - J\pi_k(x))^2}{J\pi_k(x)} \quad (C.14.1)$$

$$P-value = igamc\left(\frac{5}{2}, \frac{X^2_{(obs)}}{2}\right) \quad (C.14.2)$$

Em que $\chi^2_{(obs)}$ é a estatística observada, π_k é o probabilidade do estado ocorrer, v_k é número total de ciclos em que um estado x ocorre, $P-value$ é a probabilidade, função gama incompleto.

C.15 - Teste Variante de Excursões Aleatórias

O objetivo deste teste é o número total de vezes que um determinado estado é visitado, isto é, sua ocorrência, num andar na soma cumulativa aleatória. O teste detecta desvios do número esperado de visitas a vários estados do passeio aleatório, logo a ideia a contar o número de vezes que o objeto foi em determinada posição. As equações C.15.1 mostra como é computado o valor do teste:

$$P-value = \operatorname{erfc} \left(\frac{|\xi(x) - J|}{\sqrt{2J(4|x| - 2)}} \right)$$

Em que ξ é número total de vezes que o estado x ocorre em J ciclos, $P-value$ é a probabilidade, função erro complementar.