FPGA-based True Random Number Generation

using Circuit Meta-stability with Adaptive Feedback Control

Mehrdad Majzoobi[1], Farinaz Koushanfar[1], and Srinivas Devadas[2]

[1] Rice University, ECE

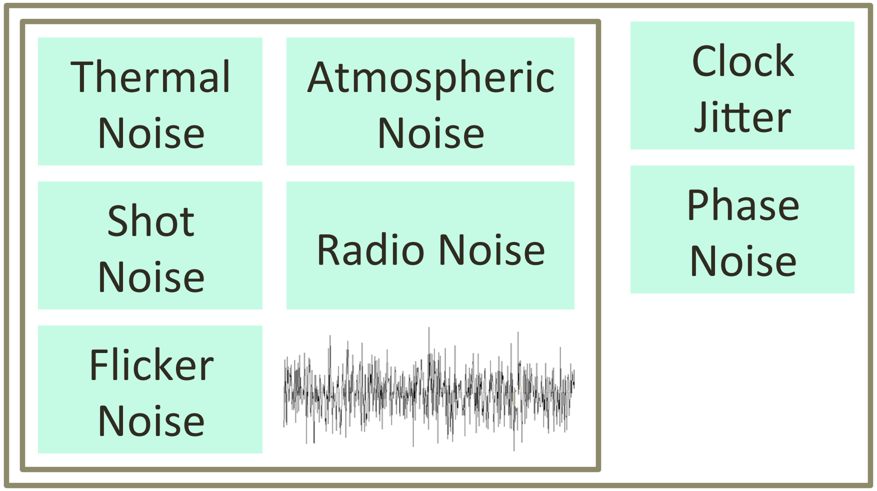
[2] Massachusetts Institute of Technology, EECS

Random Number Generator (RNG)

- Pseudo-RNG (PRNG)
  - Seed
    - Source of entropy, i.e., a longer random number from a shorter seed
  - Algorithm
- How is the PRNG seed generated?
  - Predictable?
    - E.g., Netscape browser[1]: srand(time(0))
- True-RNG
  - No seed
  - Based on a random physical phenomenon

[1] http://www.cs.berkeley.edu/~daw/papers/ddj-netscape.html

Sources of Randomness



Thermal Noise

Atmospheric Noise

Shot Noise

Radio Noise

Flicker Noise

Clock Jitter

Phase Noise

- Lavarand
  - Developed in 1996
  - Generate randomness from lava lamps
  - Efficiency/Cost?

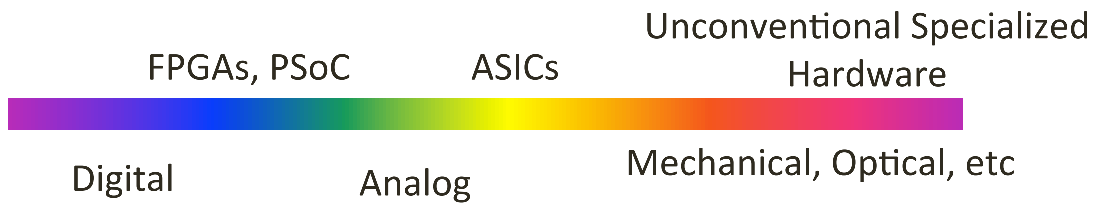
http://www.lavarnd.org/

Applications

- Generating
  - Keys
  - Nonce
  - Seeds
- Random numbers used in
  - Lottery
  - Gaming and Gambling
- Demand
  - Secure communication
  - Servers
  - E.g. Intel is embedding TRNGs in its new generation processors

http://spectrum.ieee.org/semiconductors/processors/behind-intels-new-randomnumber-generator/?utm_source=techalert&utm_medium=email&utm_campaign=090111

What is the Challenge?


FPGAs, PSoC

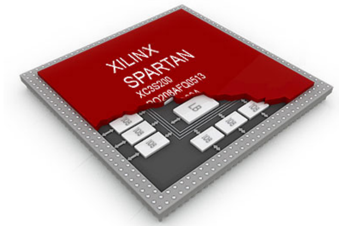
ASICs

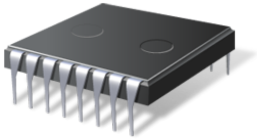
Unconventional Specialized Hardware

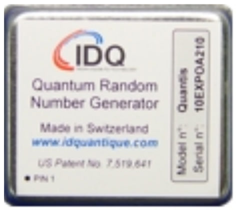
Digital

Analog

Mechanical, Optical, etc







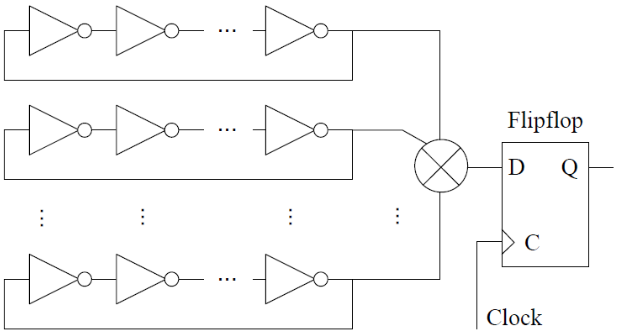
- Source of randomness?
- Implementation cost
  - The cost of generating one (entropy) bit
    - Quantum random number generation *
    - Specialized hardware (high cost)
  - Speed/throughput
  - Power
  - Ease of implementation
- Security
  - Biasing attacks

- FPGA
  - More FPGA designs than ASICs
  - Reconfigurable
  - Shorter Time-to-Market
  - Cheaper in low volume

* http://www.idquantique.com/

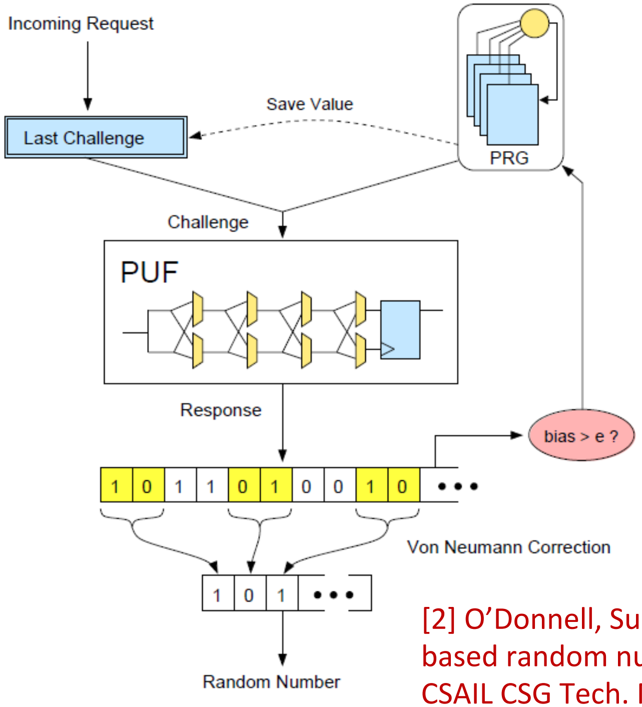
Related Work

- Analog TRNGs
- Digital TRNGs
    - Clock jitter
    - Metastability



Sampling ring oscillator jitter

[1] Sunar, Martin, Stinson: A provably secure true random number generator with built-in tolerance to active attacks. IEEE Trans. on Computers 58, 109–119 (2007)

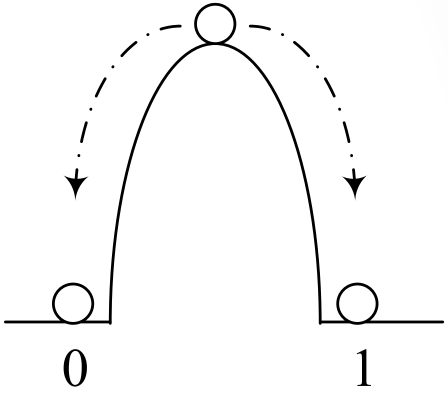


[2] O'Donnell, Suh, & Devadas: PUF-based random number generation: MIT CSAIL CSG Tech. Memo 481 2004

- Cons of popular ROs
    - Low entropy rate
    - Strong dependence on working condition
    - Can synchronize on perturbations or other ROs
    - High power consumption

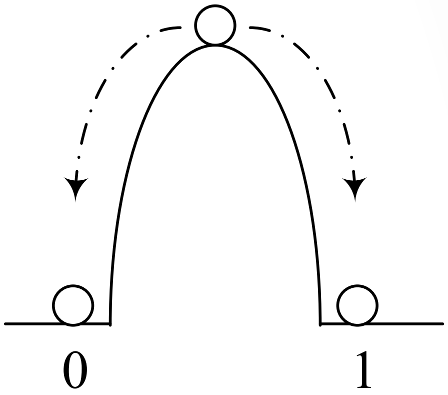
This Work

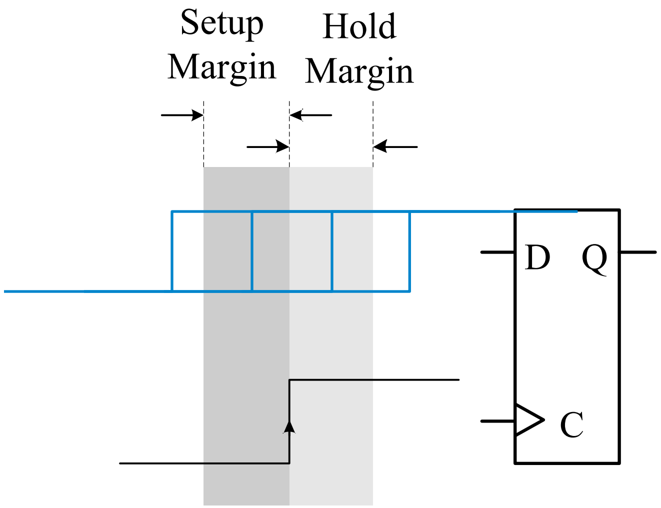
- Flip-flop metastability
  - circuit/ambient noise sensor
- Fine delay tuning
  - force metastable operation
- At-speed feedback mechanism
  - automatic tuning
  - Robust operation
  - Resilient against active attacks - biasing
- Simple design principle
  - Easy to observe how randomness relies on physical phenomena
- High entropy and throughput per unit area

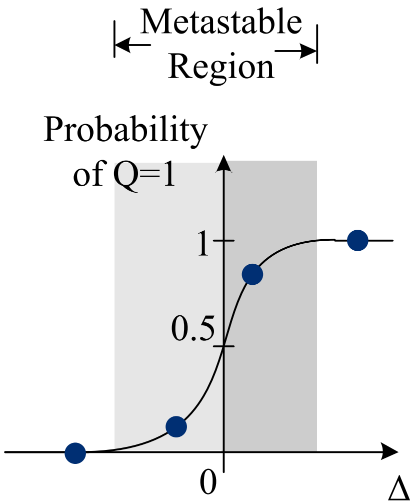


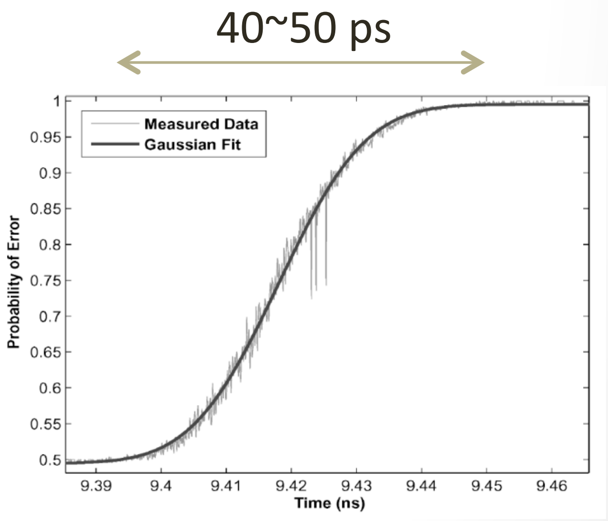
Flip-flop Metastability

- Metastability
  - Highly sensitive to noise
- Flip-flop
  - Delay difference
  - Simultaneous arrival





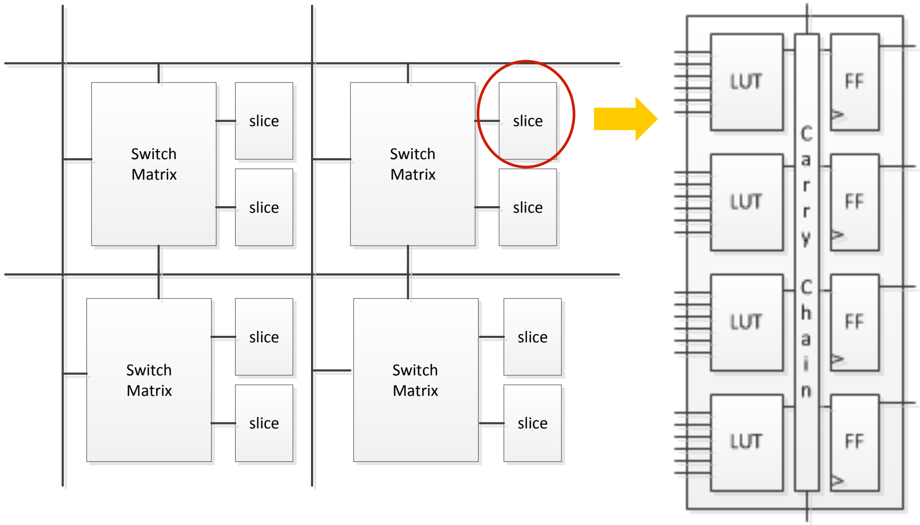




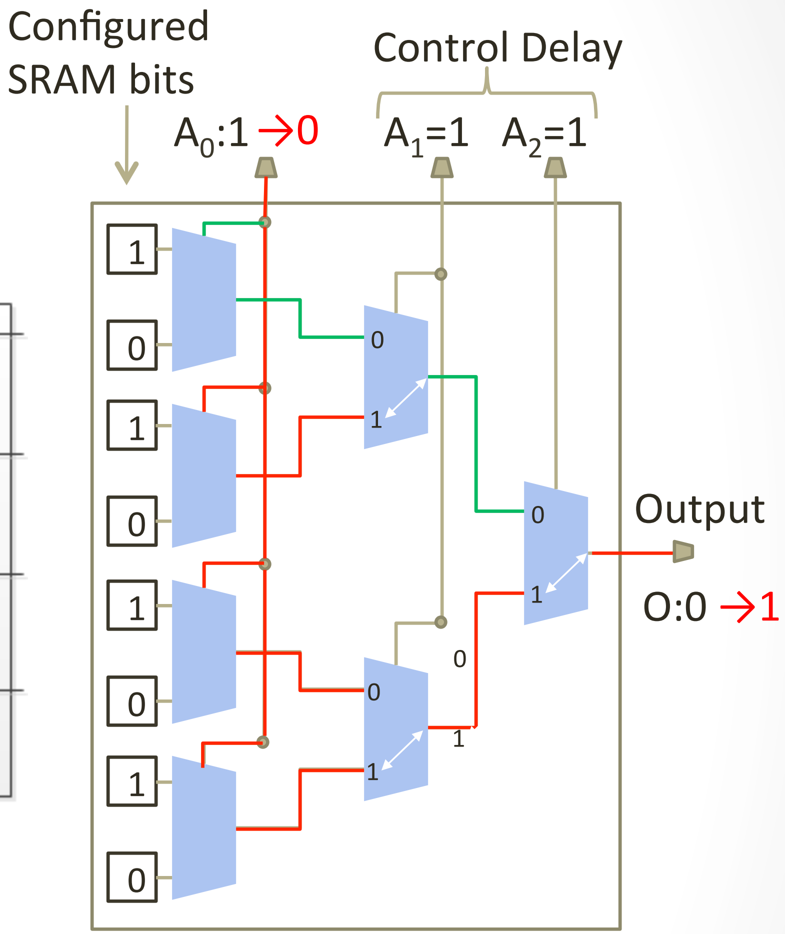
Majzoobi, Koushanfar, and Potkonjak Techniques for Design and Implementation of Secure Reconfigurable PUFs. *TRETS. Syst.* 2, 1, Article 5, 2009

Delay tuning

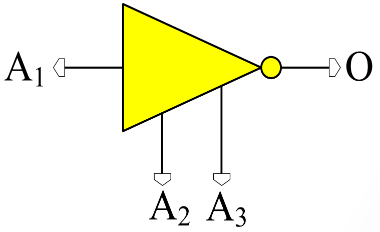
- FPGA
  - Lookup table (LUT)



- Programmable delay Line (PDL)
  - Incremental changes in propagation path
    - Tree-like network
  - Resolution ~ 10 ps



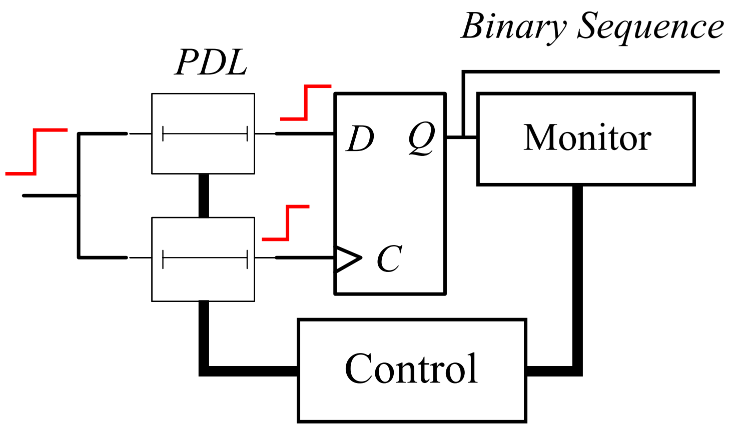
Example: 3-input LUT

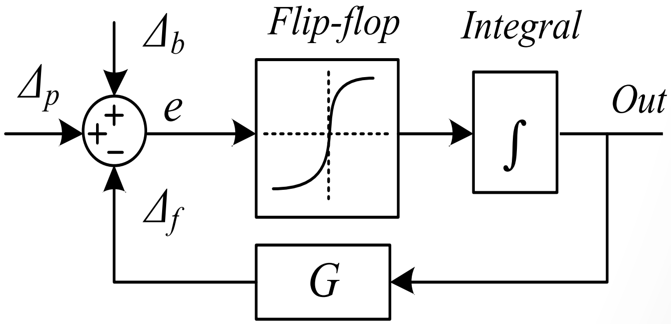


TRNG System Design

- Monitor bit probabilities
- Provide feedback to perform delay tuning
  - Monitor
    - Counter - accumulator
  - Control
    - Linear feedback – linear decoding

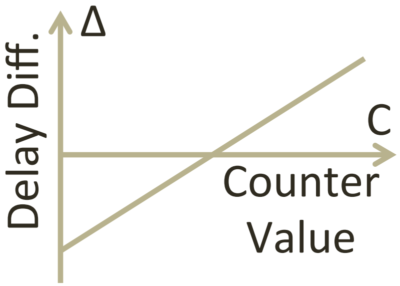
PI – proportional integral controller

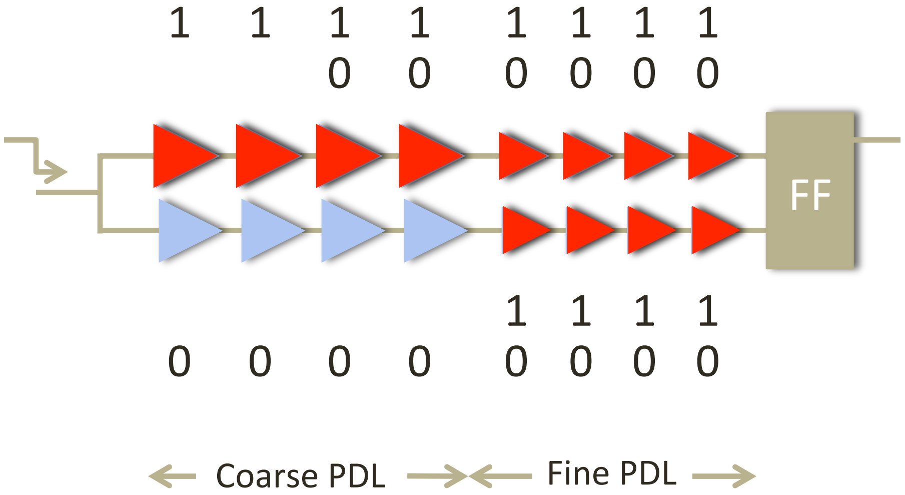


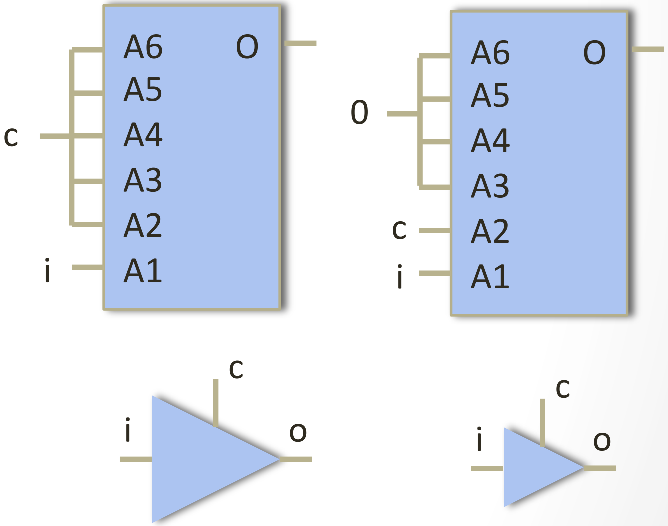


Implementation

- Coarse and fine delay tuning knobs
  - Synthesize delay within a target range
  - Fine PDL resolution = 32 x Coarse PDL resolution
- Linear Decoding



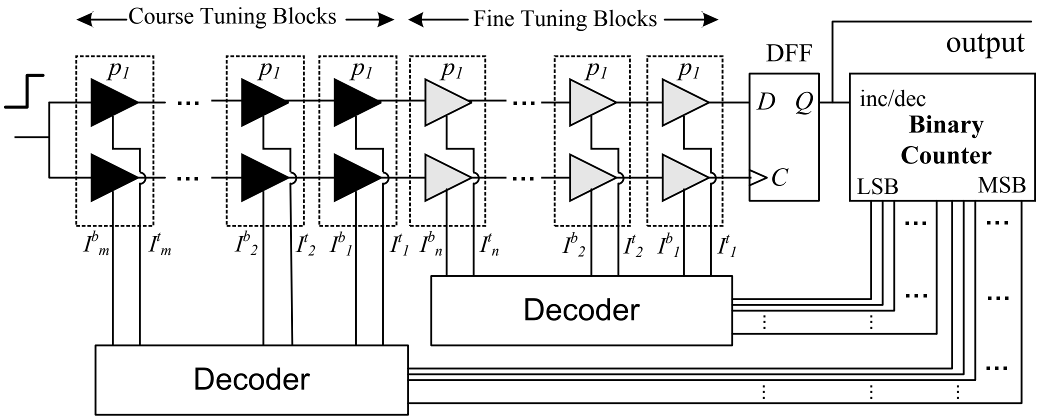




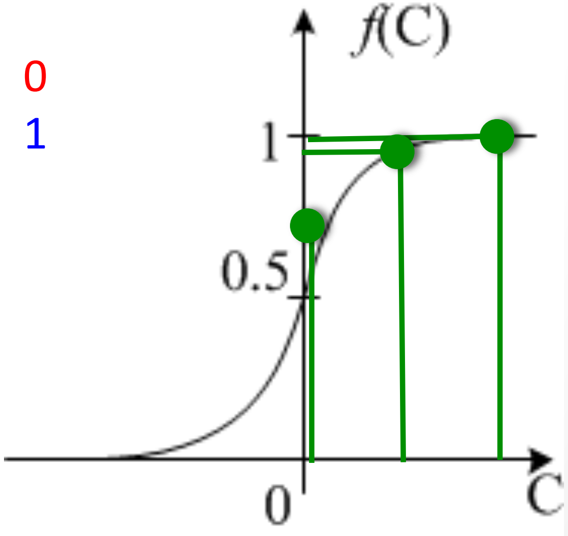
Random Walk

- 1D random walk through counter values
  - C ←C + x where x = {1,-1}, C = counter value
  - Prob{x = -1} = 1 - Prob{x = 1} = *f*(C)
- The farther from the center the higher the probability of moving toward the center

Increment Counter Value
Decrement Counter Value

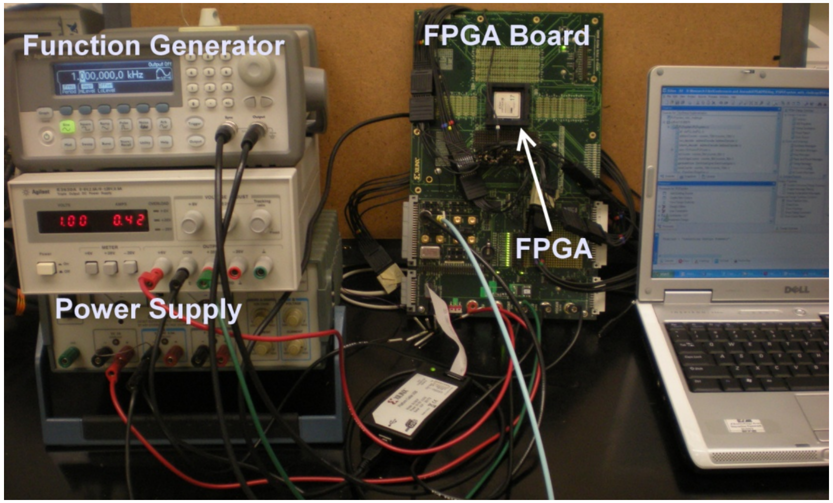


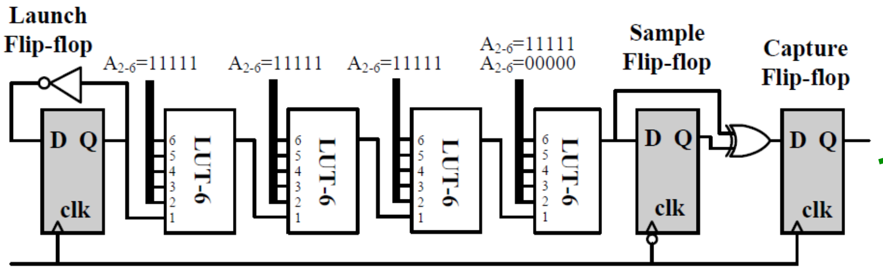
0
1

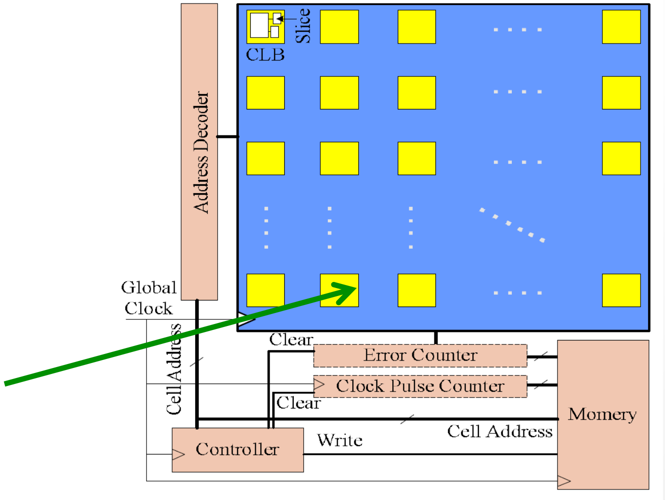


Measurement setup

- Measuring PDL resolution
- External lab function generator
- Linear sweep from 10MHz to 15MHz
  - Freq * 34 by internal PLL
  - Xilinx Virtex 5 XC5VLX110
- Record error rate
- 32x32 array

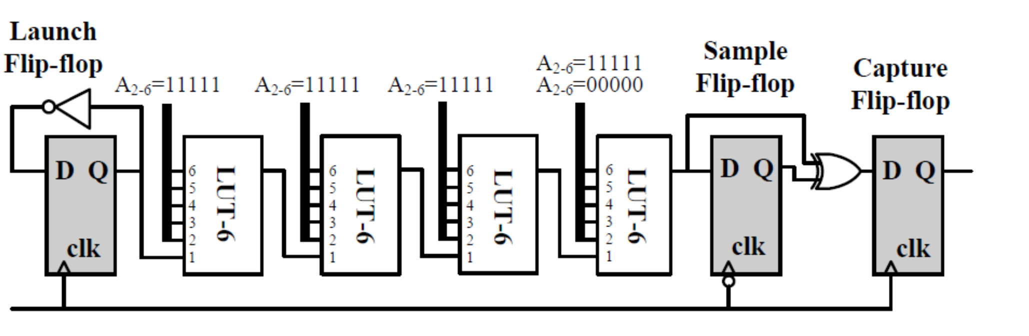


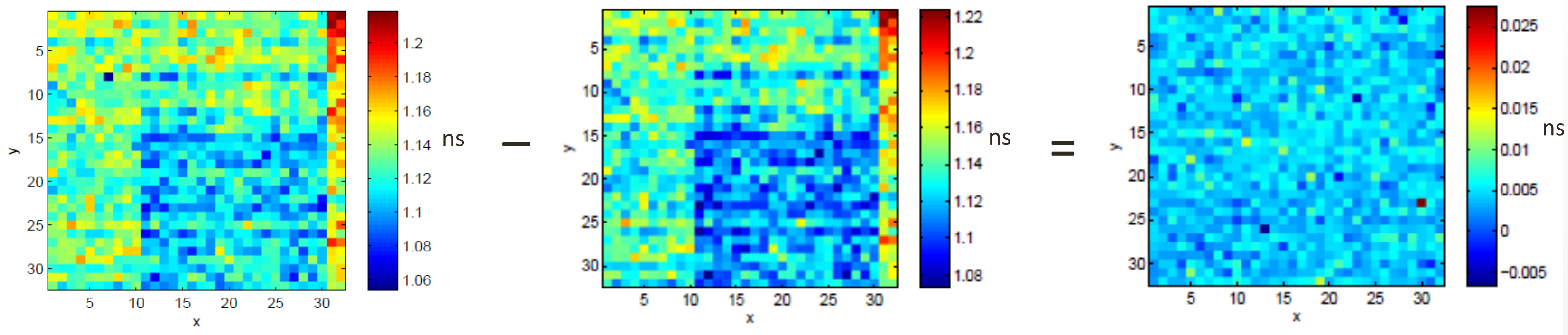




PDL Delay Measurement

- Delay difference
- Coarse delay tap
  - ~10ps
- Fine delay tap
  - ~10ps/32





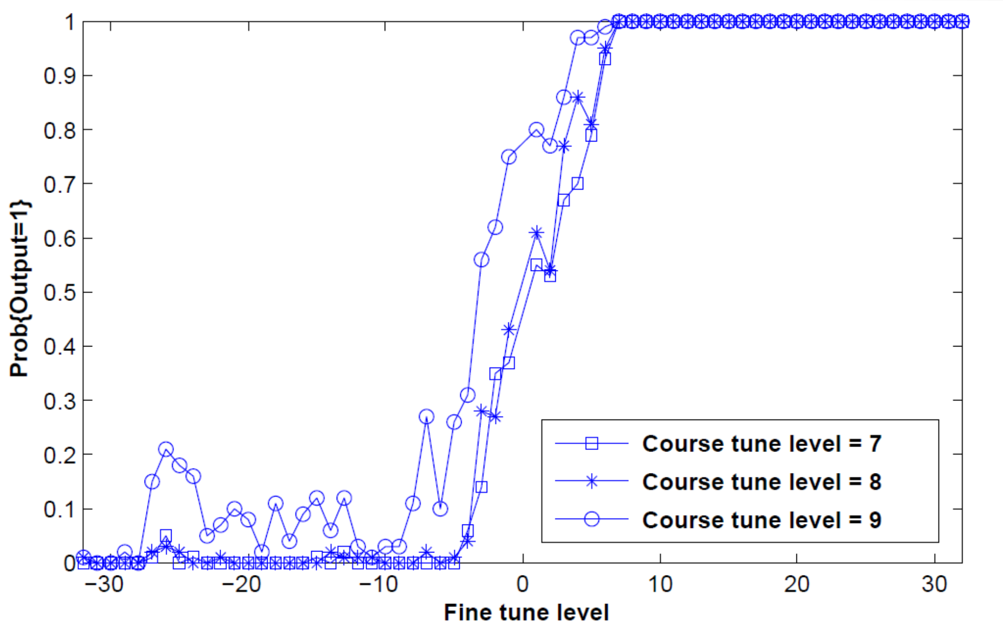
(a) Delay for $A_{2-6} = 00000$

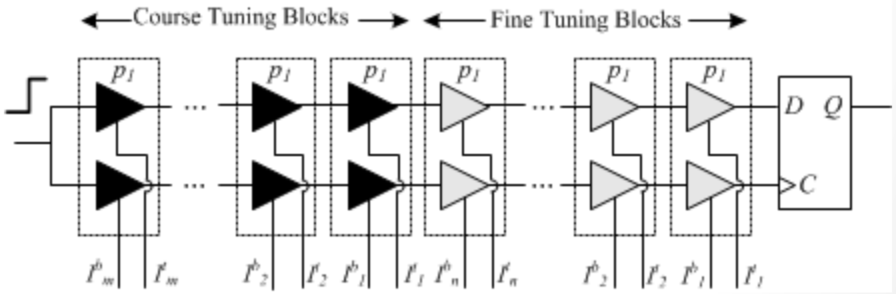
(b) Delay for $A_{2-6} = 11111$

(c) Delay difference

Tuning with PDLs

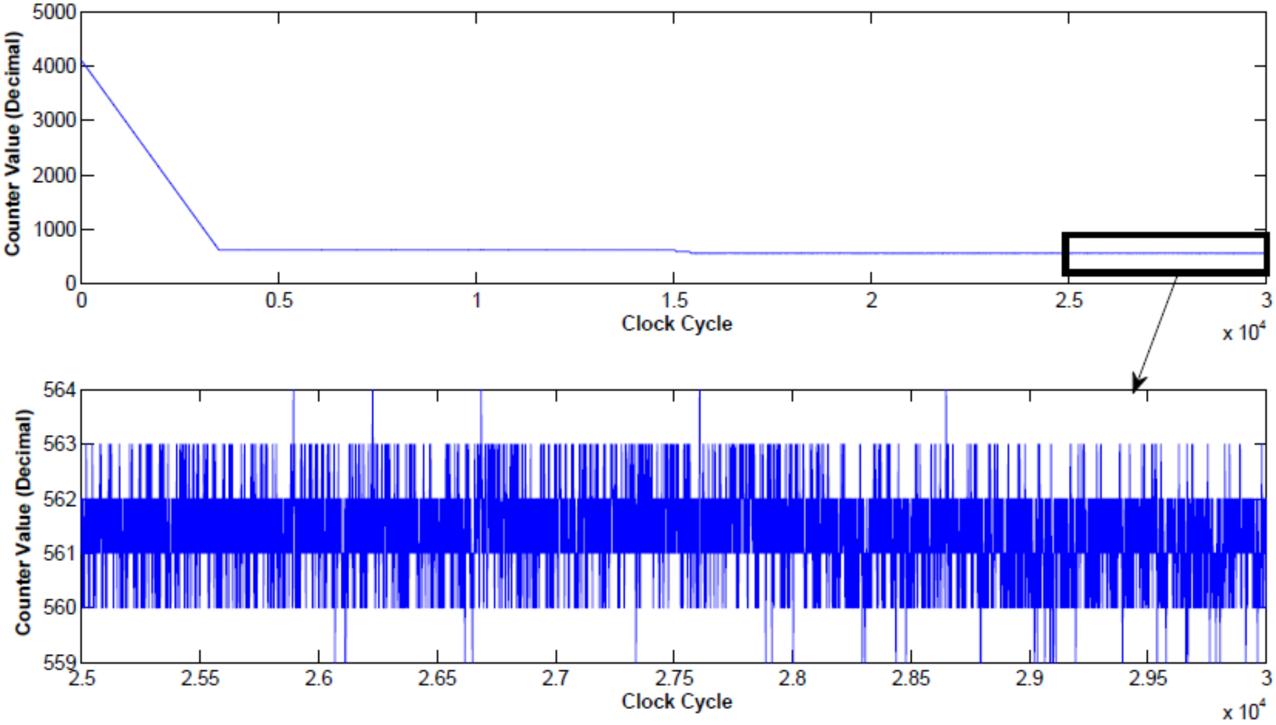
- Fine tuning stages
  - 32
- Coarse tuning stages
  - 32
- Measure probability
  - Repeat 1000 times
  - Normalize the number of 1s





Operation

- 10 bit counter (5 LSB/MSB bits control fine/coarse PDLs)
- The counter value finally settle around a constant value
    - 562
- It walks around the center values
    - Here: 564,563,562,561,560, 559



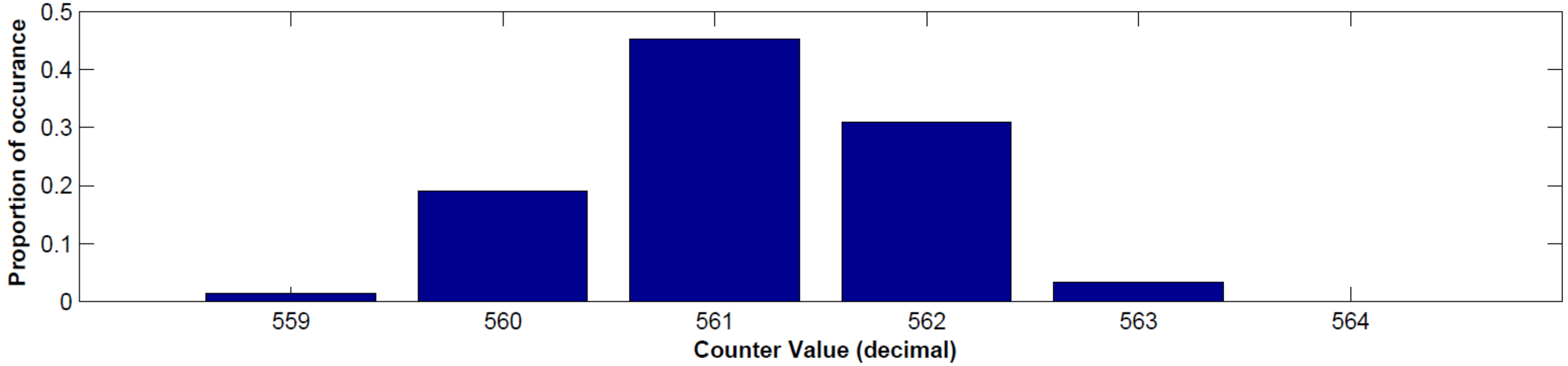
Steady state statistics

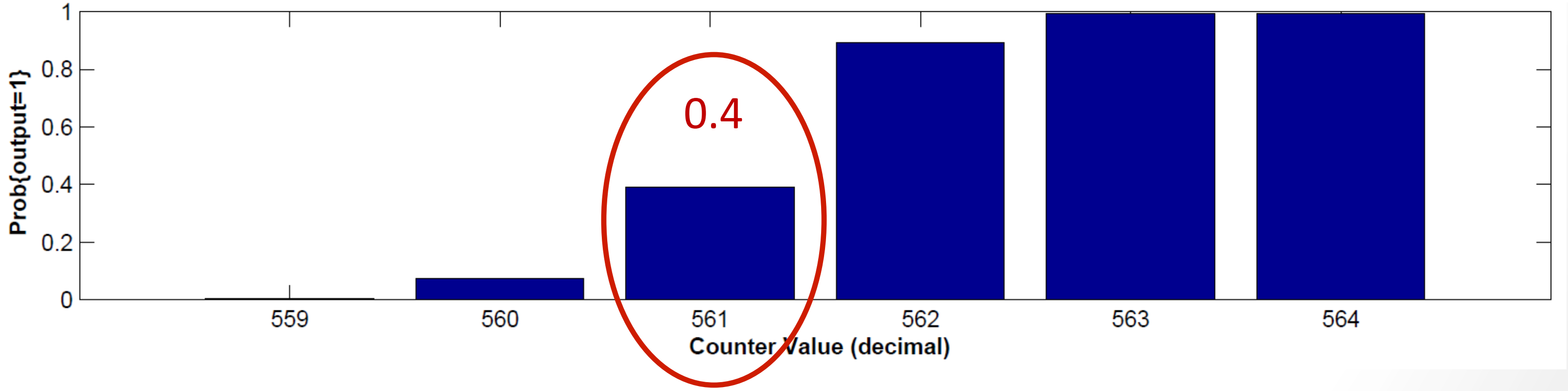
1: decrement

0: increment

- How many times a counter value appears
- What is the output bit probability associated with each counter values?

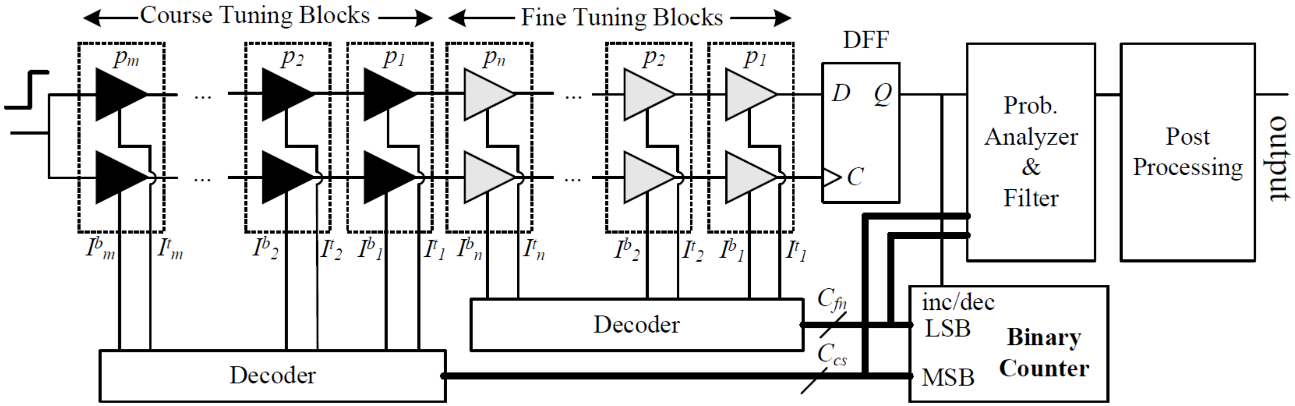
Output:	1	1	1	0	1	1	0	0	1	1	0	0
Counter:	564	563	562	561	562	561	560	561	562	561	560	561





Post-Processing

- Learning filter
  - Only output values when the counter value equals X
  - X is learned by measuring each bit probability for steady state counter values
  - In this case, X = 561
- Van Neumann correction



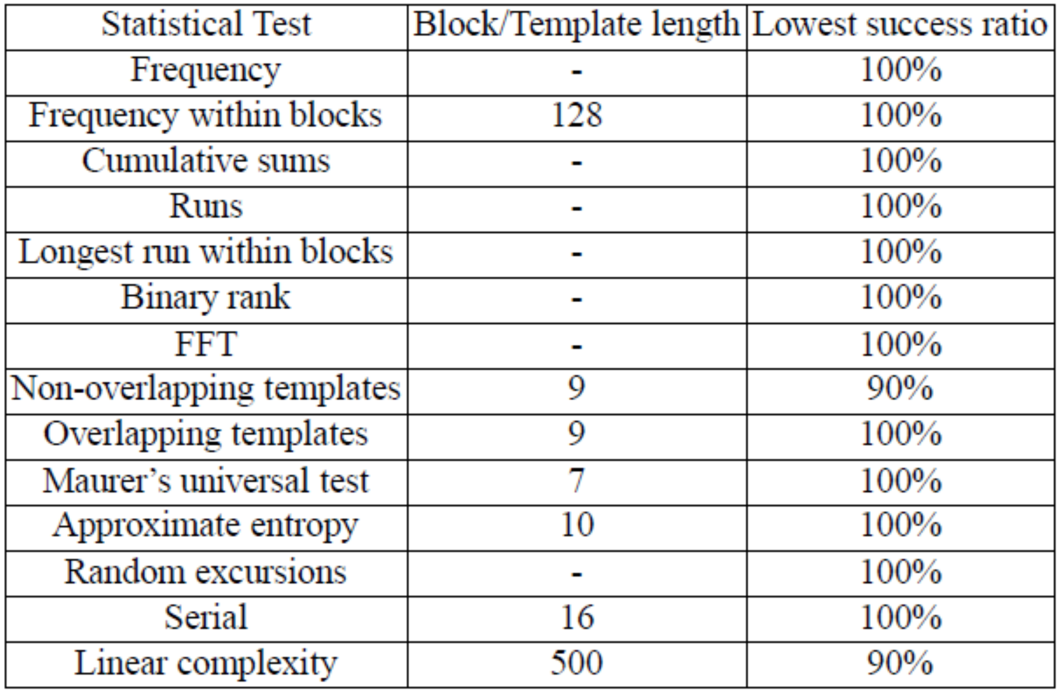
Cost

- Area
  - (32+32)x2 = 128 LUTs for the PDLs
  - 10 FFs = 10 bit counter
  - Decoder = 2 ROMs (5 bit address width – 128 bit word)
  - XC5VLX110T
    - 17,280 Slices
    - 296 18kb ROM
    - Can fit more than 100 TRNGs
- Speed
  - Forward path delay = 61.06ns
    - 16 Mbit/sec,
    - 2Mbit/Sec after post-processing
  - Overclocking
  - Parallel cores

Statistical Test Results

- NIST suite
- After filtering and post processing

Table 1: NIST Statistical Test Suite results.



Statistical Test	Block/Template length	Lowest success ratio
Frequency	-	100%
Frequency within blocks	128	100%
Cumulative sums	-	100%
Runs	-	100%
Longest run within blocks	-	100%
Binary rank	-	100%
FFT	-	100%
Non-overlapping templates	9	90%
Overlapping templates	9	100%
Maurer's universal test	7	100%
Approximate entropy	10	100%
Random excursions	-	100%
Serial	16	100%
Linear complexity	500	90%

Conclusion

- FPGA based true random number generation
  - Flip-flop meta-stability
  - Use precise delay tuning
    - Programmable delay line
    - Single LUT
  - To generate high quality random bits
- Self adjusting mechanism
  - Resilient to active attacks
- Throughput of 2MHz with one block
  - Can have TRNG blocks run in parallel
  - Can perform overclocking