Implementation and comparison of TRNG designs on FPGA

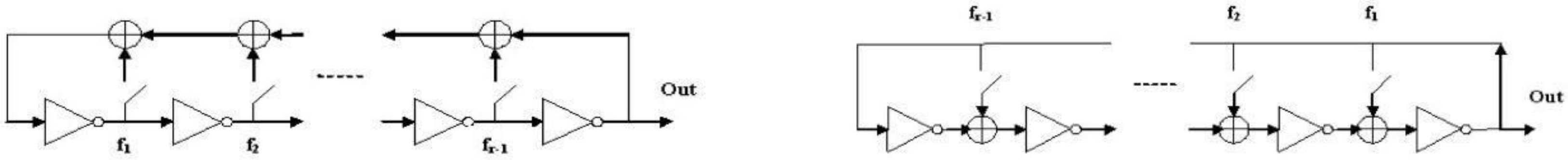
Abhijith Kashyap
Srihari Sankar

Overview

- True Random Number Generators use natural phenomenon to generate random bit streams
    - Unlike Pseudo RNG which follow an algorithmic approach
- 2 TRNG designs are implemented on FPGA -
    1. FIGARO TRNG
    2. Metastability TRNG
- Performance evaluated using NIST Statistical Test Suite
- Area and implementation complexity discussed

FIGARO TRNG

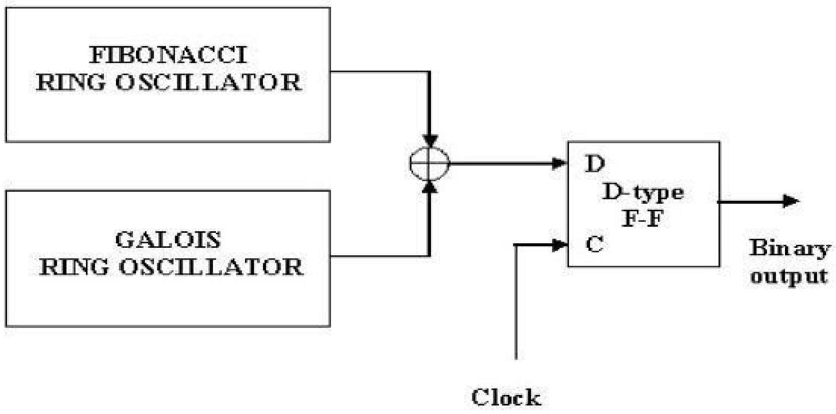
- Particular design proposed by Golic et. al.
- FIbonacci-GAlois-Ring-Oscillator TRNG
- Uses Fibonacci and Galois LFSR like structures with DFF replaced by inverters
- Has more entropy than typical ring oscillator of just inverters



Fibonacci RO on the left and Galois RO on the right

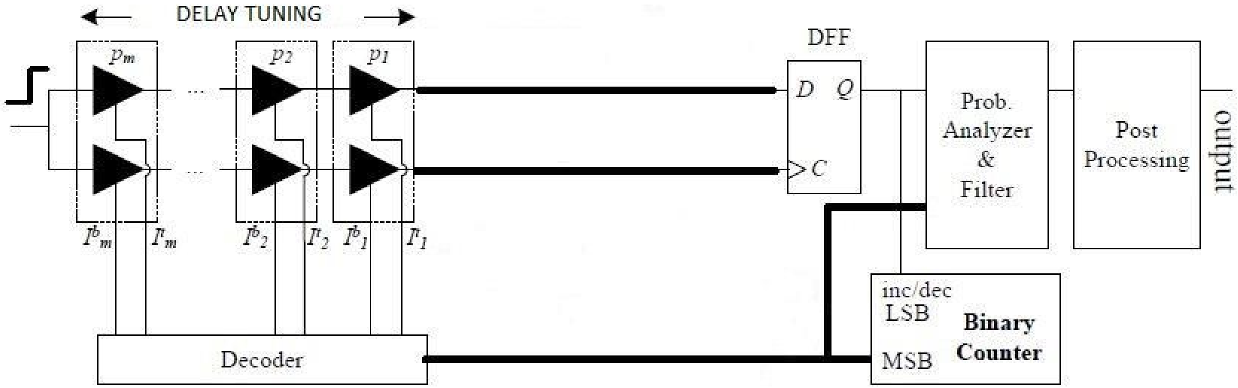
FIGARO TRNG

- FIRO + GARO are free running oscillators
- XOR of the 2 ROs sampled by a slower system clock
- Their phase information wrt system clock is the source of randomness



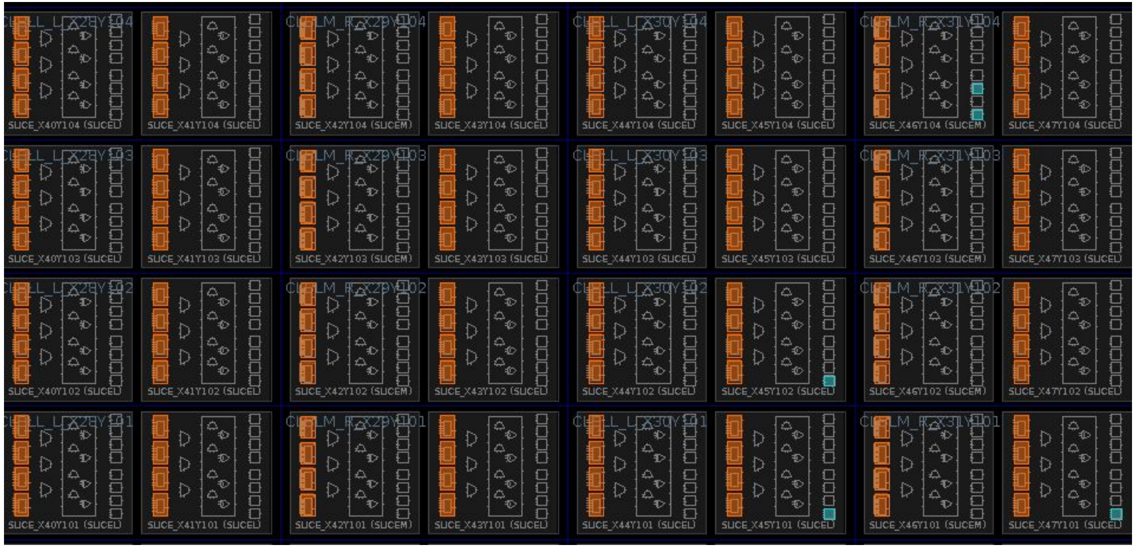
Metastability TRNG

- Particular design proposed by Majzoobi et. al.
- Get a flop in the metastable region (setup-hold window)
  - Ensure both clock and data of DFF are driven by same net
  - Add configurable delay lines on paths to compensate for path variations



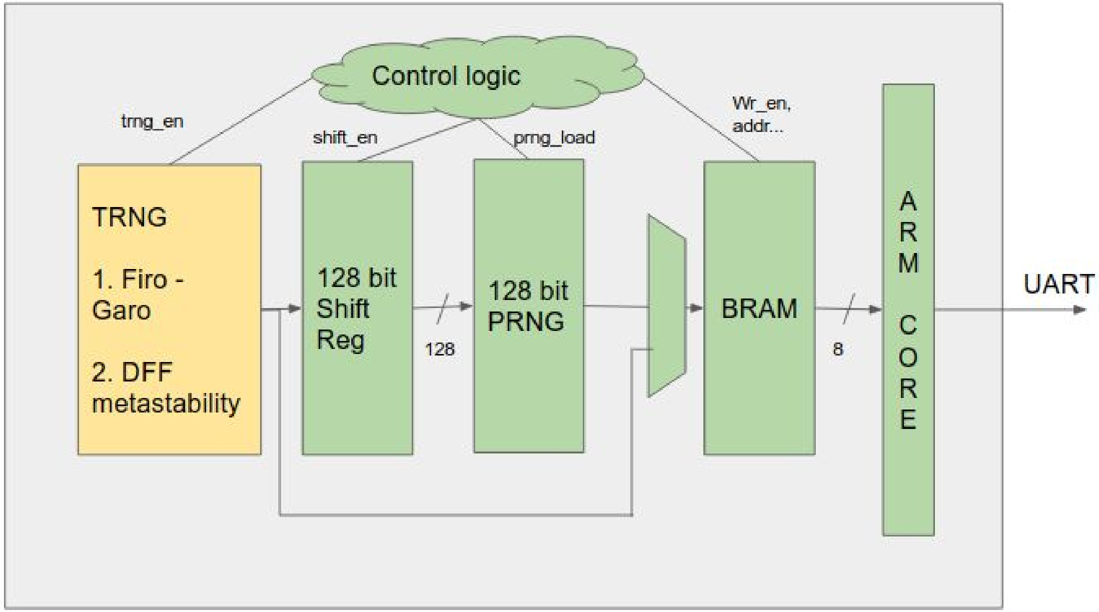
Metastability TRNG

- Requires manual place and route of delay lines and sampling DFF as shown below
- Design is not easily portable across different FPGAs



FPGA System

- Implemented on Xilinx Zynq Zedboard
- Logged data out through UART



Comparison

Design	**NIST Performance**	**Area**	**Implementation Complexity**
FIGARO	Good	High	Low
Metastability	Average	Low	Very High

- Overall, FIGARO performs much better with quality of random data while requiring less effort to implement on FPGAs
- Metastability based TRNG fails some NIST tests and requires significantly more effort on FPGAs