Pre-Silicon Power Analysis of Cryptographic Hardware

Nicole Lesperance

Power Side Channel Attacks

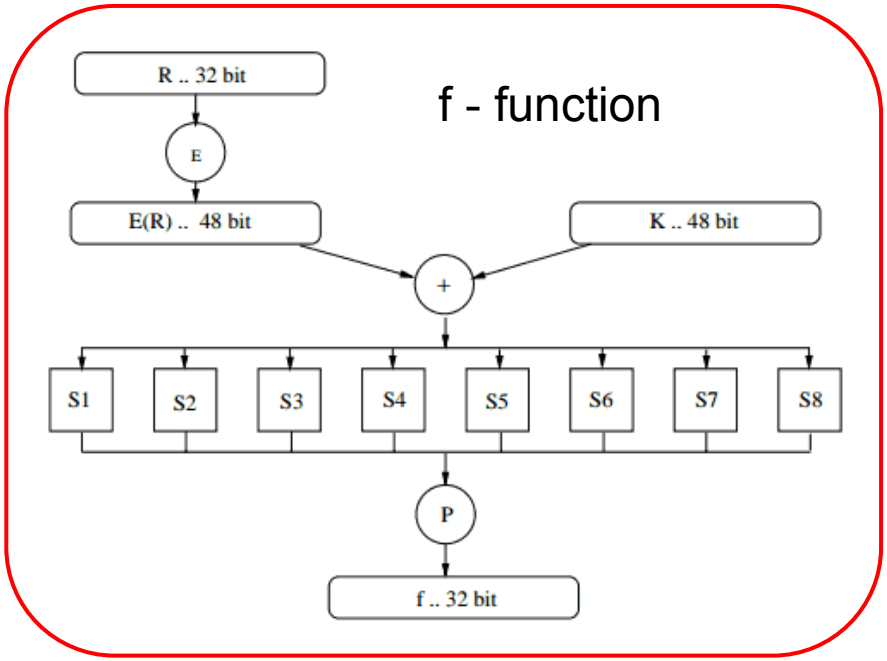
Simple Power Analysis: Examine a single power trace to determine key bits

- Need model correlating key bits with power

Differential Power Analysis:

- Divide samples into 2 groups (high and low power) using an "oracle"
- Oracle uses power model to estimate power consumption from predicted value B
- B must depend on key and ciphertext (or plaintext)

Differential Power Analysis: DES Example



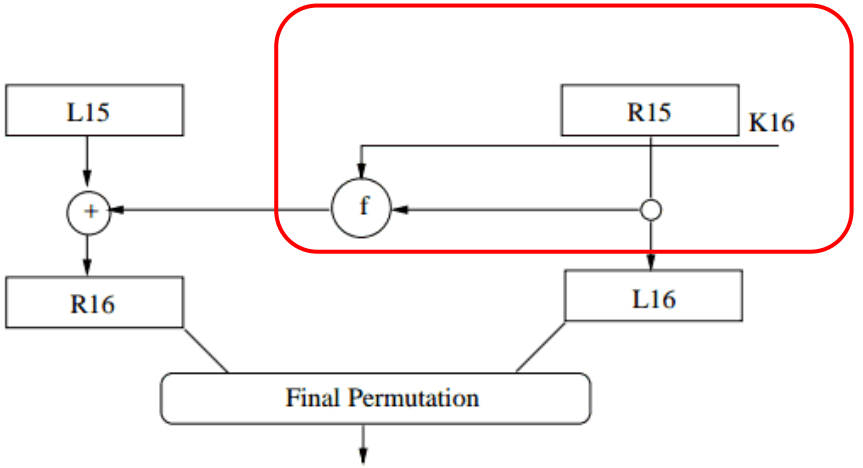


Fig. 3. DES last round

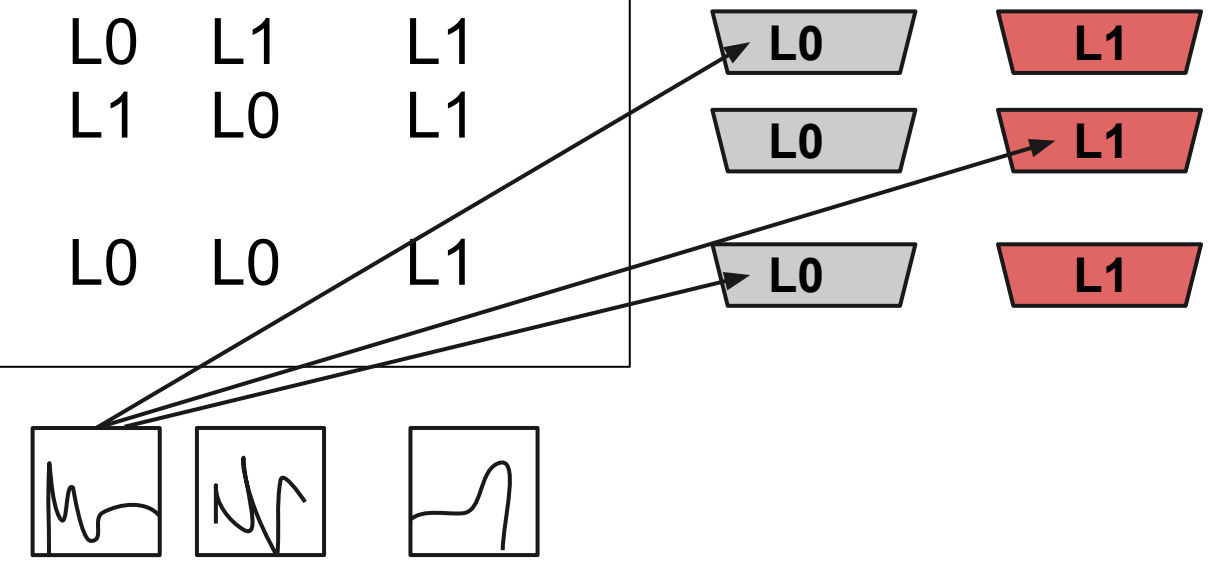
If the attacker knows the ciphertext value (C), it is possible to enumerate all possible values for 6 bits of K16 (Ki) and predict the value of a bit in L15 (L)

DES Attack Cont.

1. Take many power measurements of different messages encrypted using same key: S[0..999][0..9999]
2. Define a key dependent selection function: L = D(ki, C) where Ki = 0..63
3. For each of the 64 possible Ki values and each of the 1000 encryptions, predict L.

Des Attack Cont.

Hypothesis	S0	S1	...	S999
Ki = 000000	L0	L1		L1
Ki = 000001	L1	L0		L1
...				
Ki = 111111	L0	L0		L1



Correct hypothesis is the one where bins L0 and L1 differ the most

DES Power Trace

Power Reference

Correct Key Guess

Incorrect Guess

Incorrect Guess

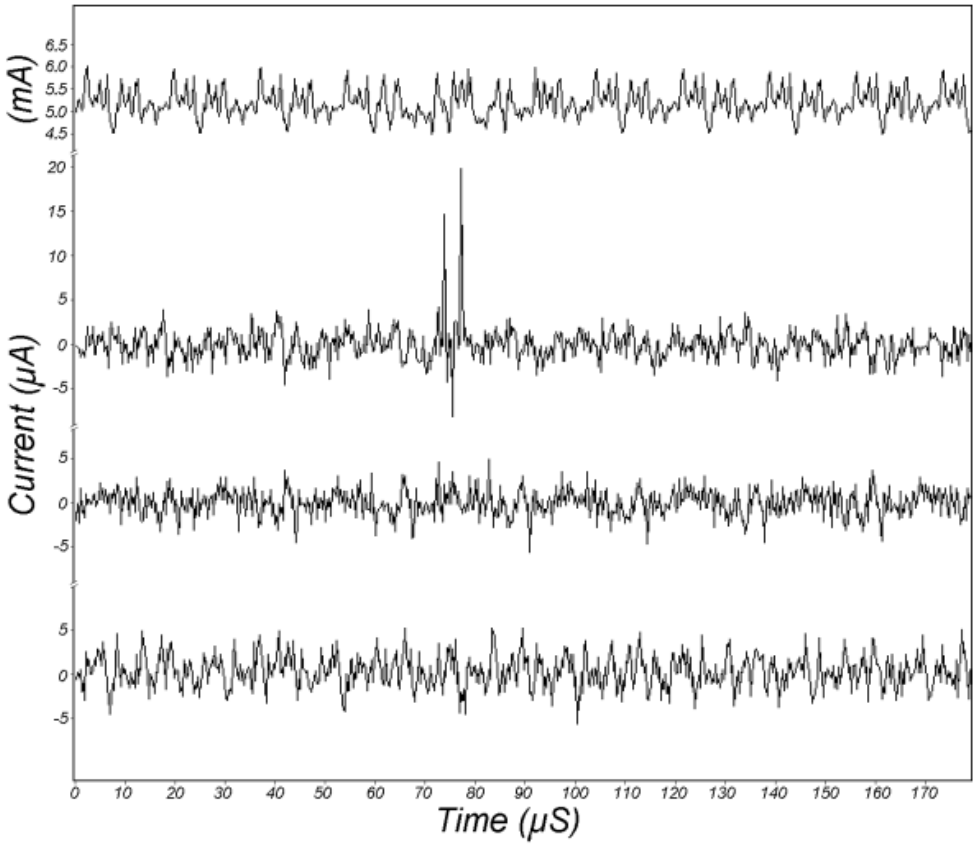


Figure 4: DPA traces, one correct and two incorrect, with power reference.

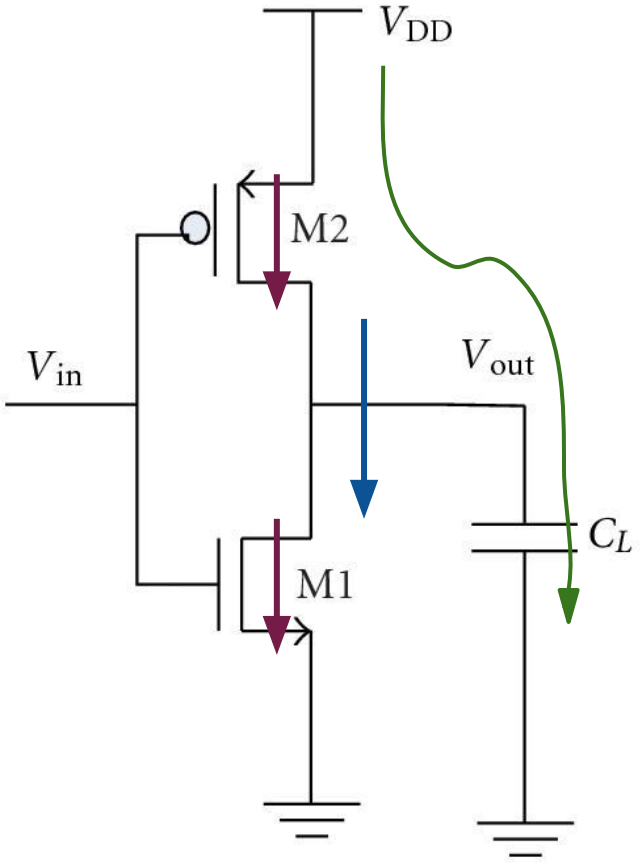
Paul Kocher, Joshua Jaffe, and Benjamin Jun, "Differential Power Analysis", in **CRYPTO '99 Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology**, Pages 388-397

Goals

1. Identify if an implementation is susceptible to simple or differential power analysis before chip is manufactured
2. Determine efficacy of countermeasures
3. Develop a metric comparing different power analysis countermeasures

CMOS Power

1. Static Power
   - Subthreshold leakage of "off transistor" (transistor width, Vdd, Vth)
2. Dynamic Power
   - Capacitive ($P_{dyn} = C_L V_{dd}^2 f$)
   - Short Circuit (Temp, C_L, slopes of V_{in} and V_{out})



Power Models

- **Instruction Level:** # of instructions executed
- **RT Level:** Toggle Count, Hamming Weight of data in registers
- **Transistor Level:** Take net capacitances into account

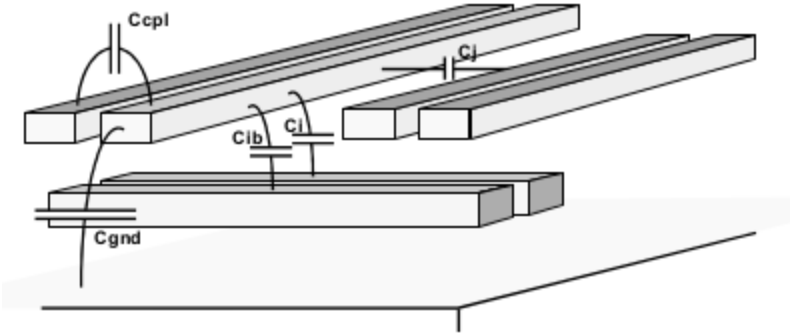


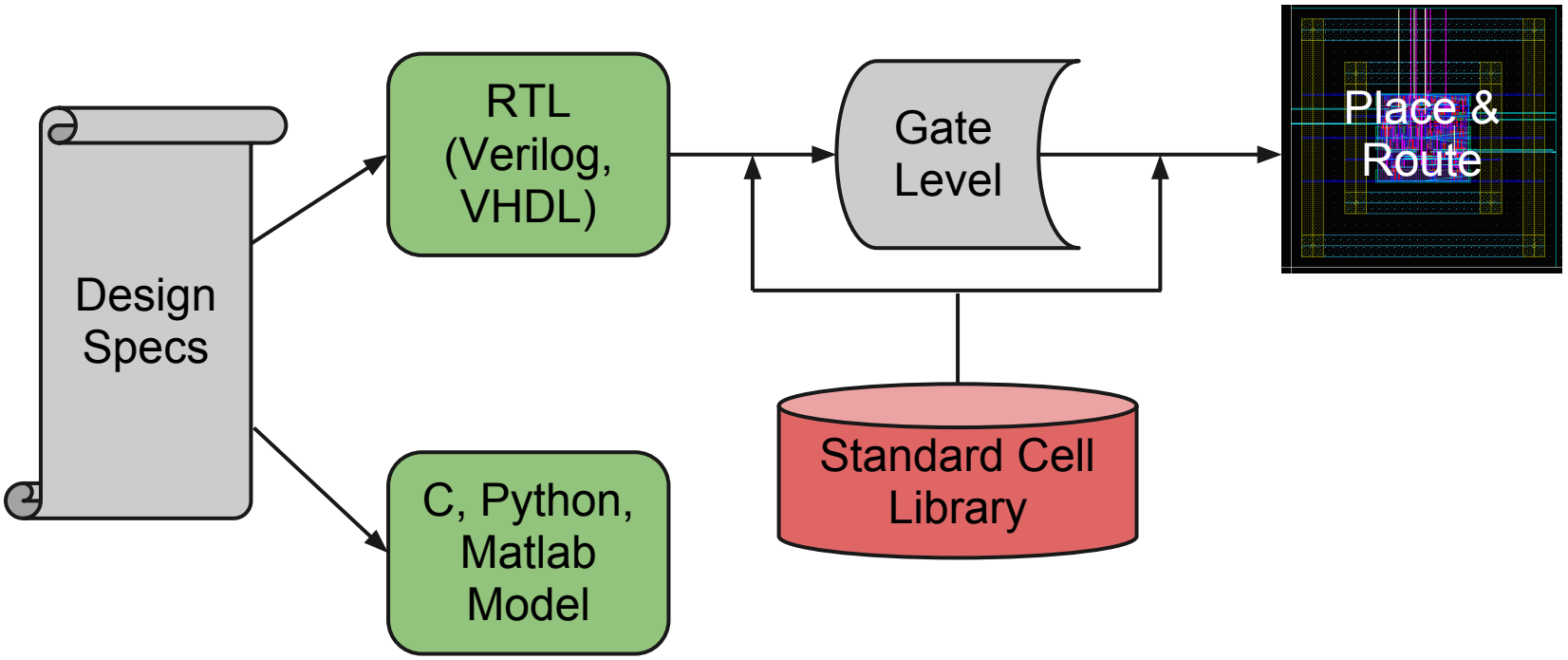
Figure 2. Interconnect capacitance decomposition.

Countermeasures

Goal: Reduce correlation between intermediate results and power consumption

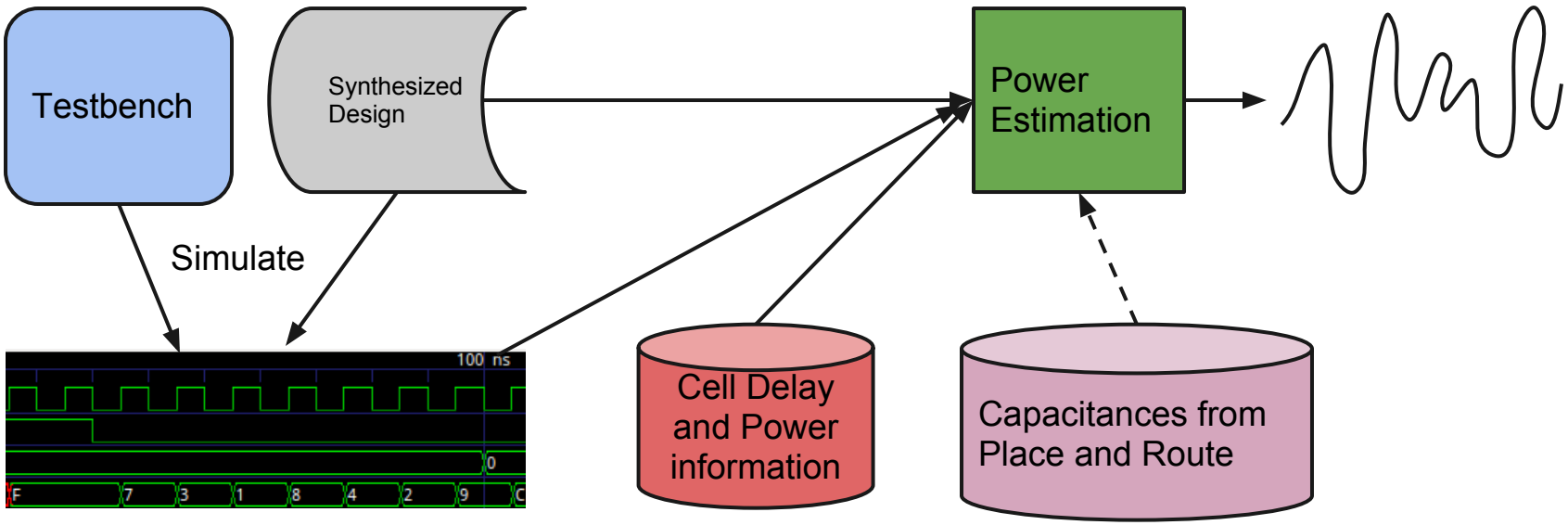
- Algorithmic countermeasures
- Data masking
- Gate level masking
- Place and route techniques

Hardware Design Flow



Project Goal

- Use EDA power estimation tools to analyze power consumption of a cryptographic hardware implementation



Synopsys Ecosystem

- **VCS** (Verilog Simulator)
- **Design Compiler** (Synthesis)
- **IC Compiler** (Place and Route)
- **Prime Time PX** (Power Analysis)

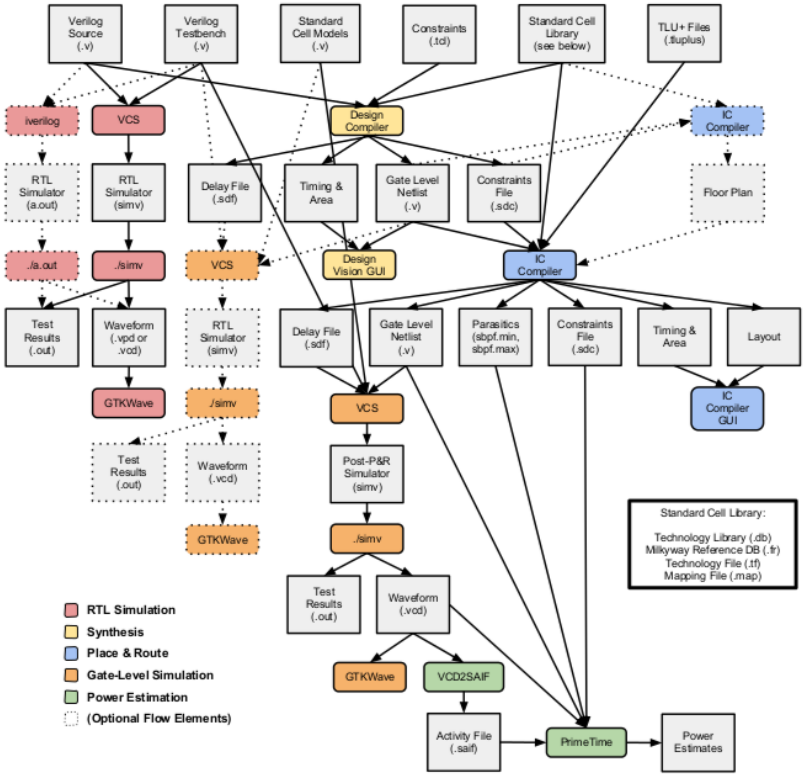


Figure 1: ECE5950 Toolflow

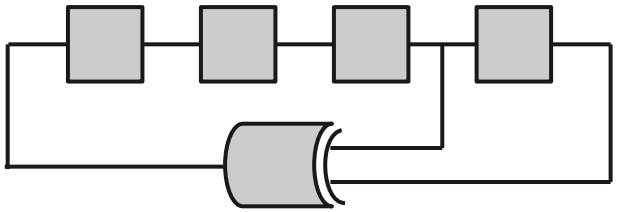
4-bit LFSR

Verilog RTL

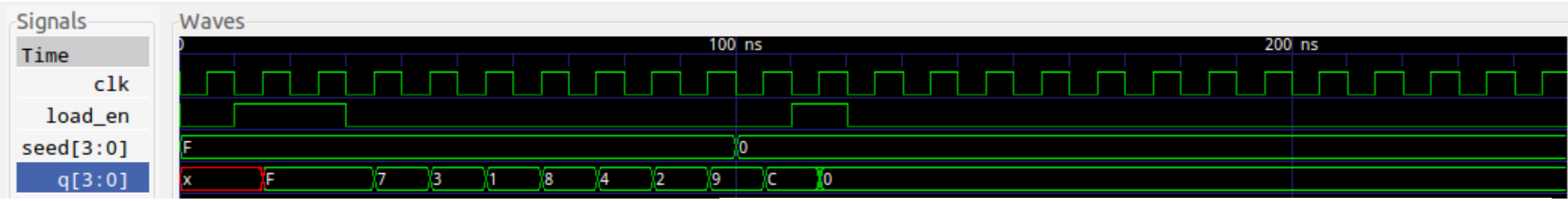
```
module lfsr
(
    clk,
    seed,
    load_en,
    q
);

input clk, load_en;
input [3:0] seed;
output [3:0] q;

reg [3:0] q;

always @ (posedge clk)
begin
    if (load_en)
        q <= seed;
    else
    begin
        q[2:0] <= q[3:1];
        q[3] <= q[0] ^ q[1];
    end
end

endmodule
```

Verilog Gate Level

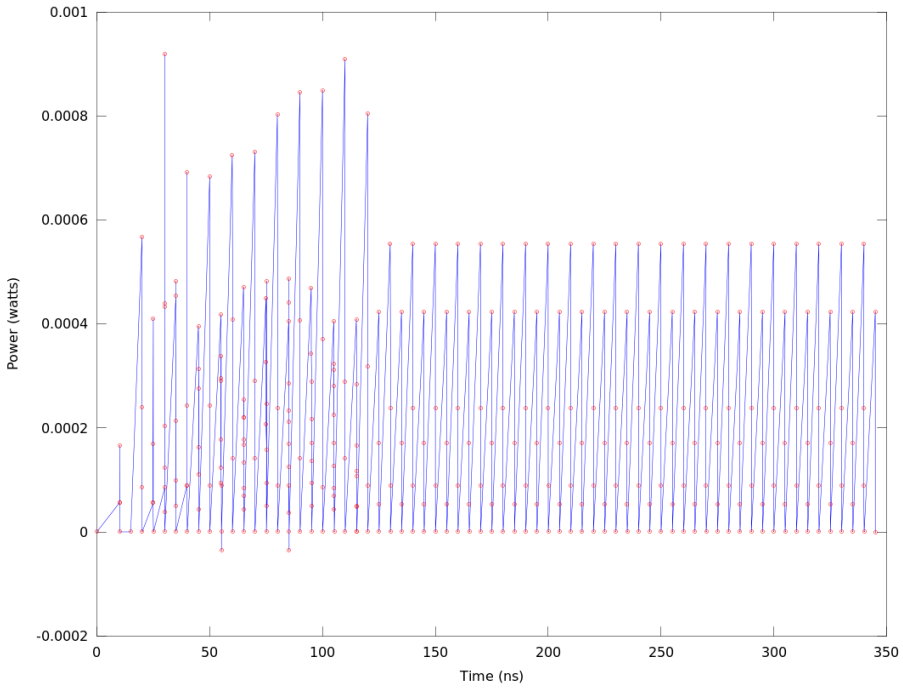
```
module lfsr ( clk, seed, load_en, q );
  input [3:0] seed;
  output [3:0] q;
  input clk, load_en;
  wire   N7, n3, n4;

  SDFF_X1 q_reg_0_ ( .D(q[1]), .SI(seed[0]), .SE(load_en), .CK(clk), .Q
(q[0])
        );
  DFF_X1 q_reg_3_ ( .D(N7), .CK(clk), .Q(q[3]) );
  SDFF_X1 q_reg_2_ ( .D(q[3]), .SI(seed[2]), .SE(load_en), .CK(clk), .Q
(q[2])
        );
  SDFF_X1 q_reg_1_ ( .D(q[2]), .SI(seed[1]), .SE(load_en), .CK(clk), .Q
(q[1])
        );
  XNOR2_X1 U6 ( .A(q[0]), .B(q[1]), .ZN(n4) );
  NAND2_X1 U7 ( .A1(load_en), .A2(seed[3]), .ZN(n3) );
  OAI21_X1 U8 ( .B1(n4), .B2(load_en), .A(n3), .ZN(N7) );
endmodule
```



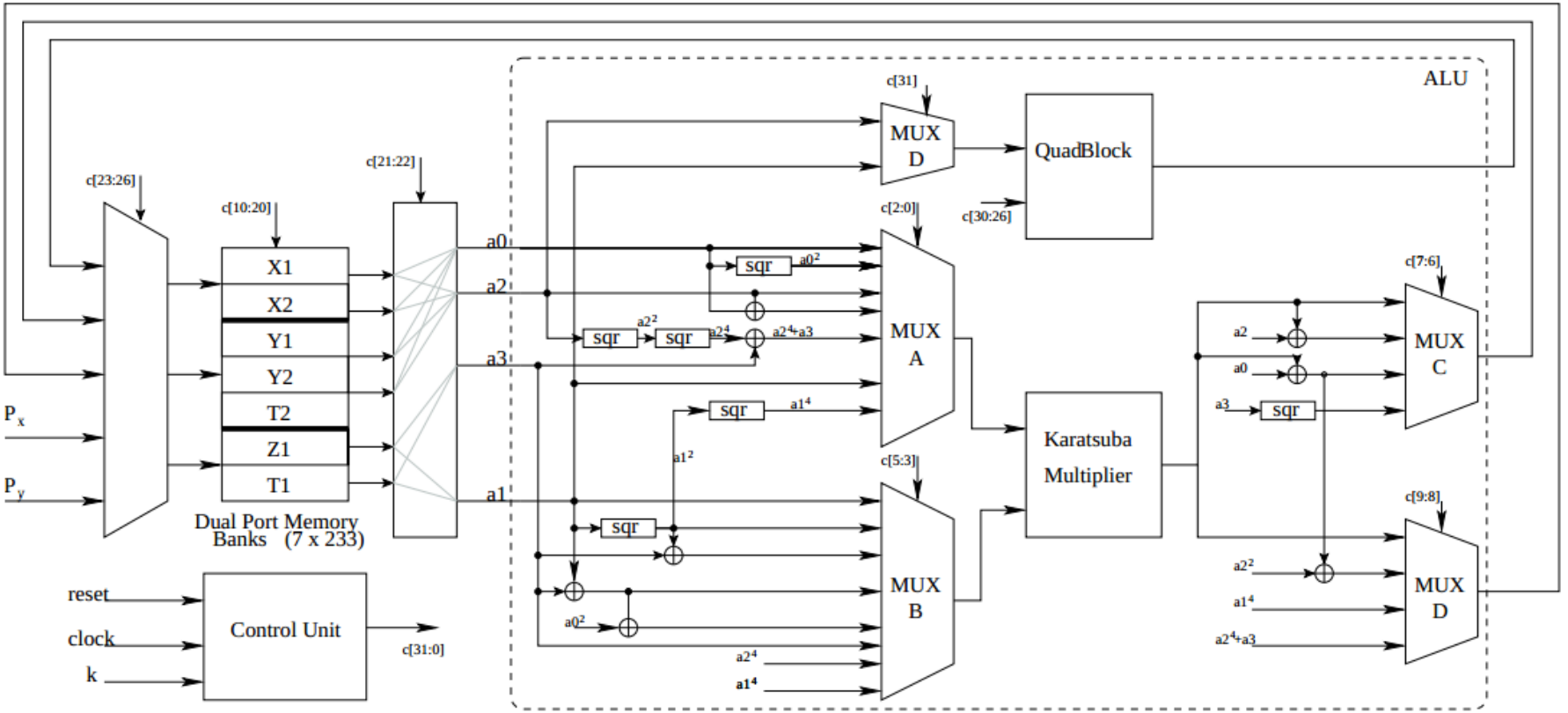
LFSR Simulation (Gate Level)



4-bit LFSR:
PrimeTime time based power analysis



Elliptic Curve Processor



High Performance Elliptic Curve Crypto-Processor for FPGA Platforms, Chester Rebeiro and Debdeep Mukhopadhyay, 12th IEEE VLSI Design and Test Symposium, Bangalore, July 2008.

The Curve

D.1.3.2.2 Curve B-233 (FIPS pg. 93), GF(2^{233})

n = 6901746346790563787434755862277025555839812737345013555379383634485463

Polynomial Basis:

* b = 066 647ede6c 332c7f8c 0923bb58 213b333b 20e9ce42 81fe115f 7d8f90ad
* Gx = 0fa c9dfcbac 8313bb21 39f1bb75 5fef65bc 391f8b36 f8f8eb73 71fd558b
* Gy = 100 6a08a419 03350678 e58528be bf8a0bef f867a7ca 36716f7e 01f81052

Design Statistics

- **Target clock period:** 100ns (f = 10MHz)
- **Area:** 65.1% Registers, 34.3% ALU, <1% everything else
- **Global Operating Voltage** = 1.1V
- **Power consumption:** 65% ALU, 34% Registers

Asymmetric Crypto: Key dependent control flow

One way function: a ^ e mod n

- Time consuming to compute a ^ e
- How many multiplications to compute M^15?
- M -> M^2 -> M^3 -> M^4 ... -> M^15 - OR -
- M -> M^2 -> M^3 -> M^6 -> M^7 -> M^14 -> M^15

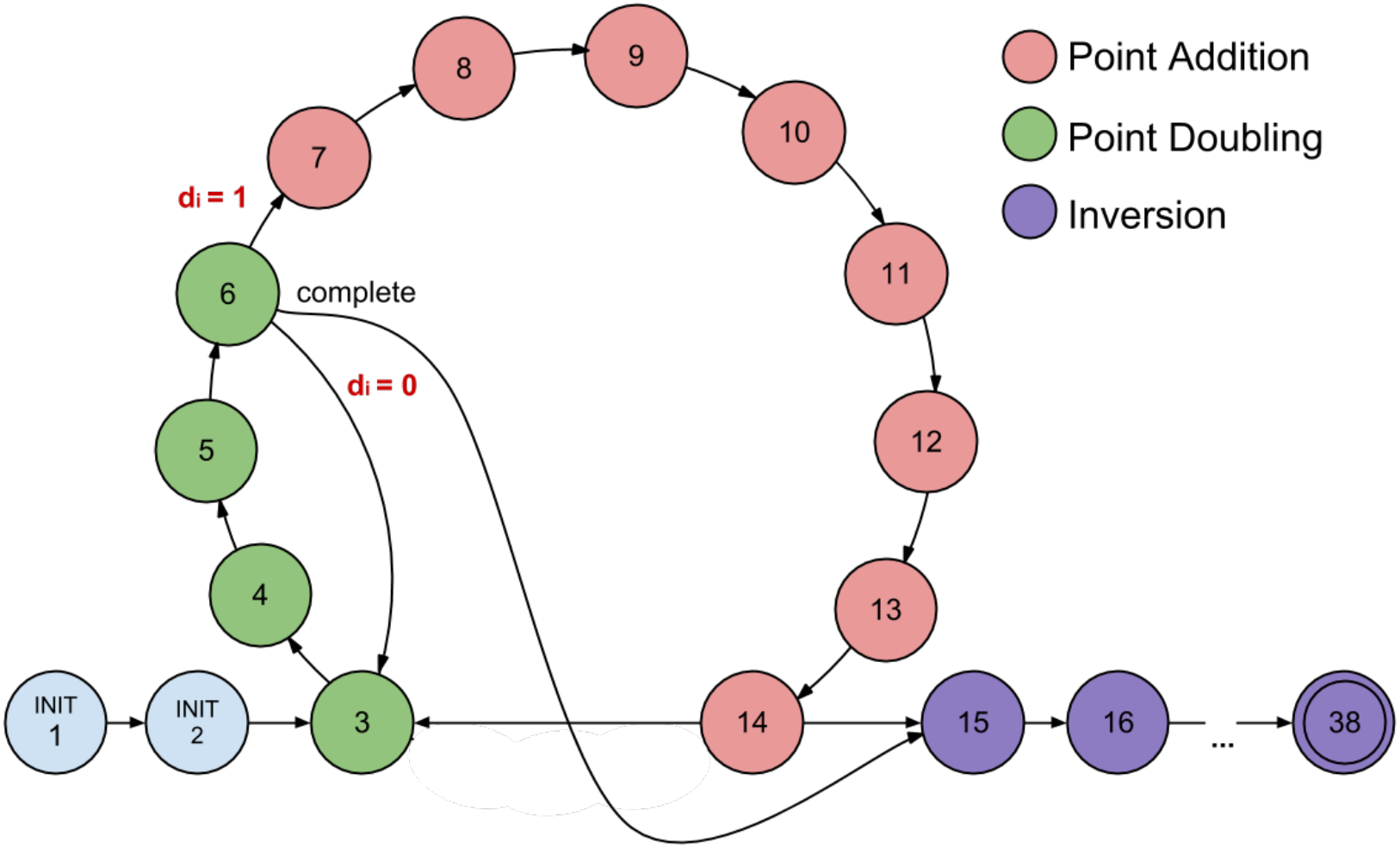
The Binary Method
Input: M, e, n.
Output: $C = M^e \bmod n$.

1. **if** $e_{k-1} = 1$ **then** $C := M$ **else** $C := 1$
2. **for** $i = k-2$ **downto** 0
   - 2a. $C := C \cdot C \pmod n$
   - 2b. **if** $e_i = 1$ **then** $C := C \cdot M \pmod n$
3. **return** C

e = 10010110110110

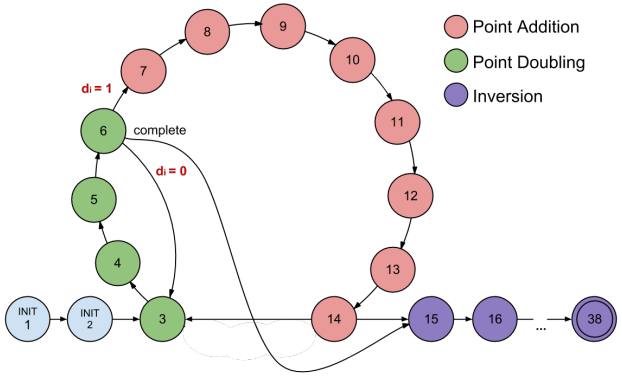
# of multiplications depends of Hamming Weight of e!

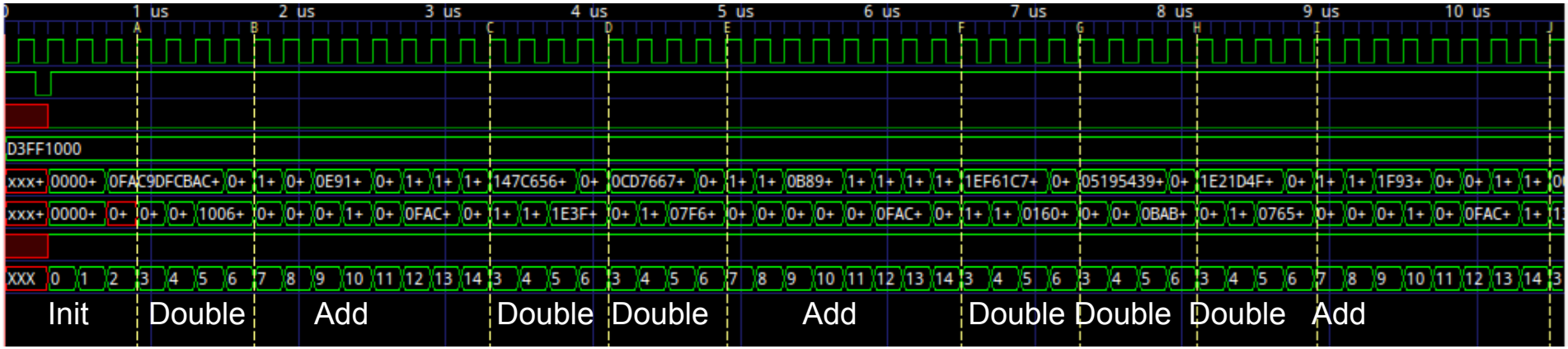
ECP State Machine



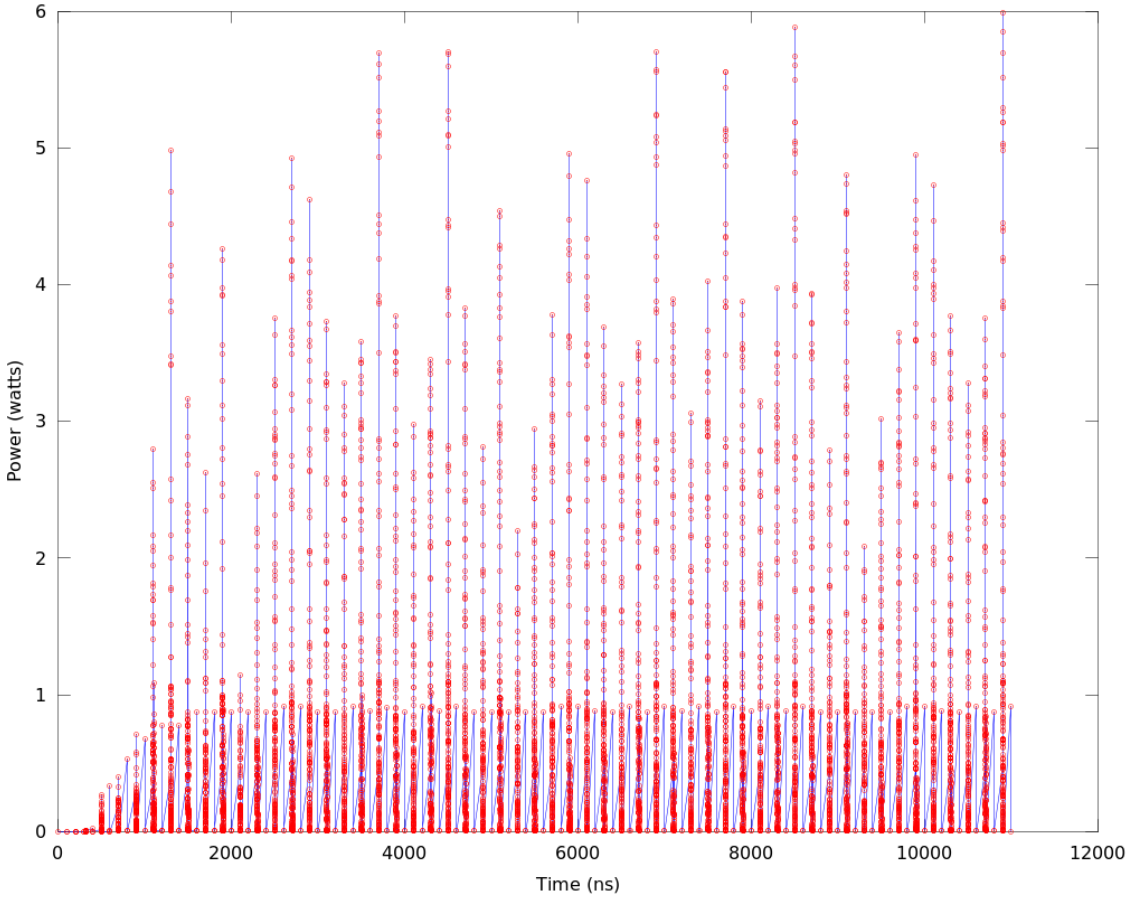
Simulation Waveform

d = 110100110101



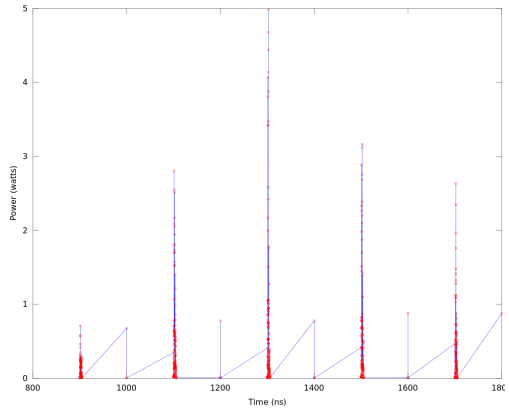


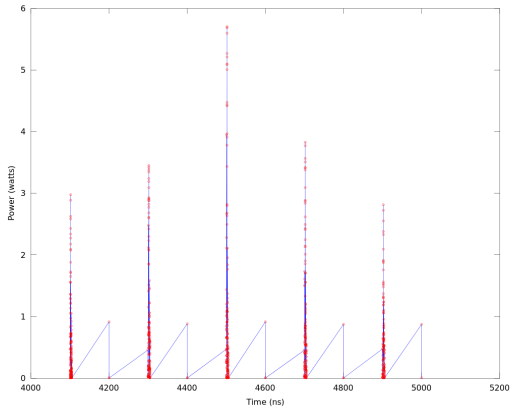
Power

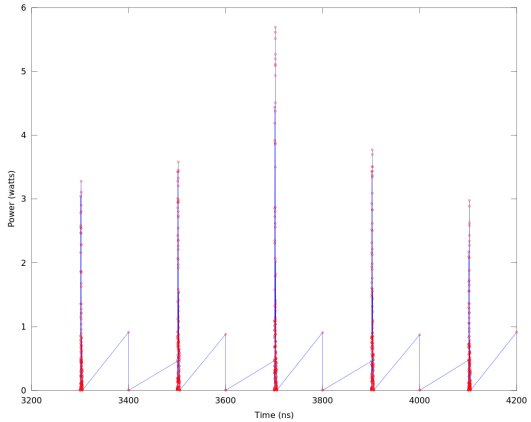

Power (watts)
Time (ns)
0
1
2
3
4
5
6
0
2000
4000
6000
8000
10000
12000

Double and Add

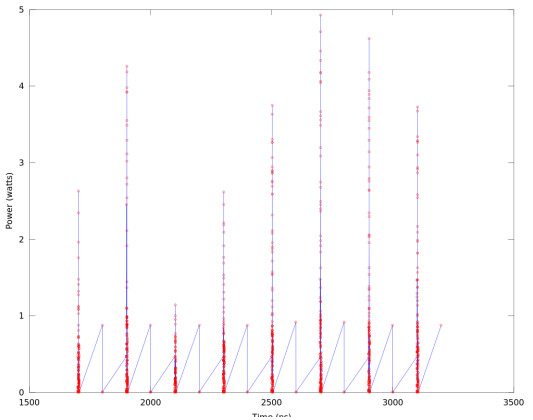
Double

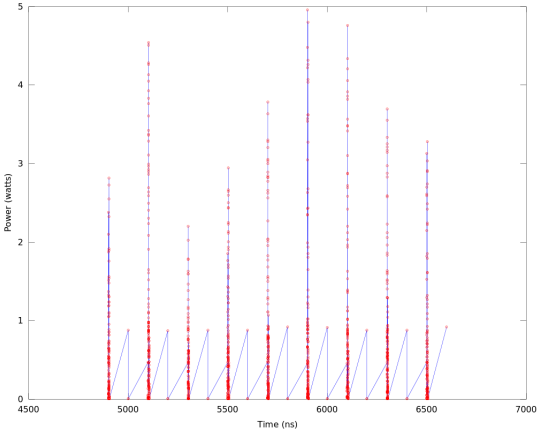


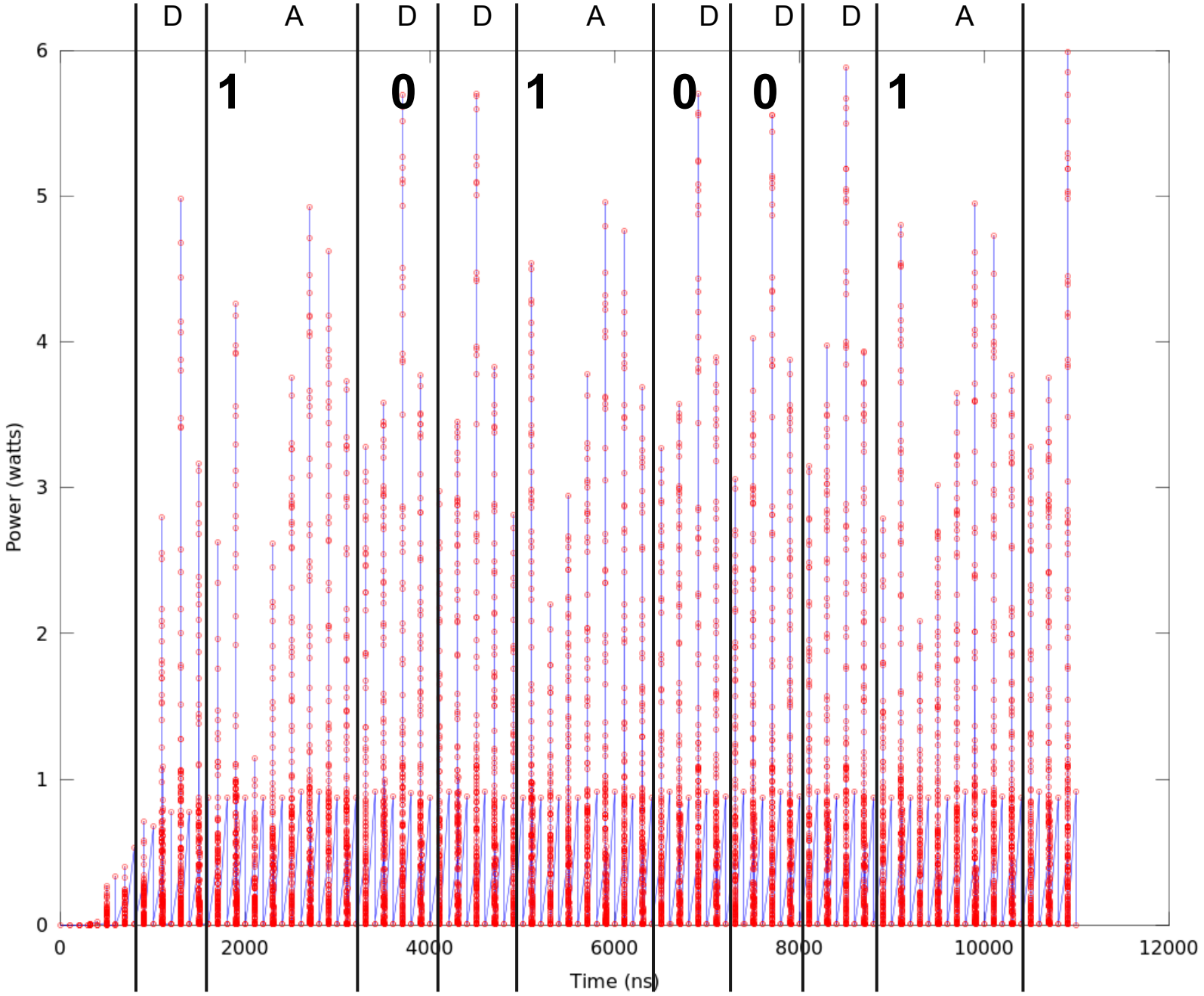




Add






D
A
D
D
A
D
D
D
A
1
0
1
0
0
1
Power (watts)
Time (ns)
0
1
2
3
4
5
6
0
2000
4000
6000
8000
10000
12000

Conclusions

- Assumptions made for various power models greatly influences analysis
- Need to compare variance in power consumption at different levels of abstraction