

9/2/03 (1)

CEG 5010: Reconfigurable Computing
Efficient Shift Registers, Counters and
Random Number Generators

*"If I had more time, I would have
written a shorter letter"*

- Marcus T. Cicero

3-Mar-08 (2)

Introduction

- Efficient FPGA design is normally nonobvious
 - with practice, you can learn when to use the special features
- This lecture
 - Shift registers
 - Linear feedback shift register (LFSR)
 - Using RAM for shift registers
 - Virtex SRL16
 - True random number generator
 - Generalised LFSR

3-Mar-08 (3)

Shift register

- Basic building block in digital circuit design
- Features
 - high speed due to simple logic & interconnection
 - low interconnection costs

3-Mar-08 (4)

Shift registers

- Applications
 - delay line
 - serial-parallel conversion
 - parallel-serial conversion
 - boundary scan
 - serial buses
 - bit serial signal processing (more about this in future lectures)

3-Mar-08 (5)

Straightforward Implementation

```

if (clk'event and clk = '1') then
  sr <= srin & sr(sr'high downto (sr'low+1));
end if;

```

How many slices for an N bit SR?

3-Mar-08 (6)

Better shift registers

- Is there a more efficient way to implement a shift register?
 - (of course or I probably wouldn't ask this question)
 - (actually knowing when there *isn't* a better way is also an important skill)

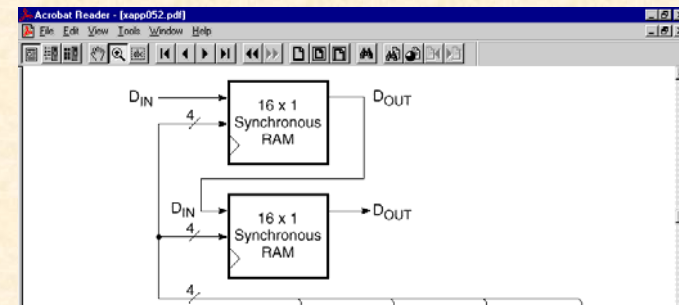
3-Mar-08 (7)

RAM

- Remember
 - Xilinx RAM has 16x better density than a LUT
 - Shift registers only need serial access to the memories
 - i.e. do not need all of the outputs in parallel, only need the first input and the last output

3-Mar-08 (8)

Shift register using RAM



3-Mar-08 (9)

Address generation: counters

- Ripple carry counter
 - Synchronous counter
 - Gray code counter
 - LFSR
- What are the advantages and disadvantages of each?

3-Mar-08 (10)

Counters

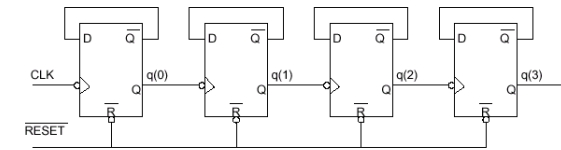


Figure 4.1: A 4 bit ripple counter circuit. The output of one flip-flop clocks the next one, hence the term "ripple" count.

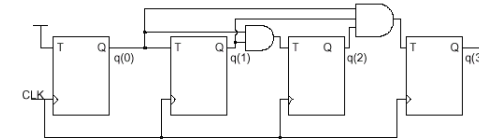


Figure 4.2: A 4 bit synchronous counter circuit. All flip-flops are clocked simultaneously so the outputs change (almost) simultaneously.

3-Mar-08 (11)

Gray code

0000	1100
0001	1101
0011	1111
0010	1110
0110	1010
0111	1011
0101	1001
0100	1000

Table 4.1: 4 bit Gray codes.

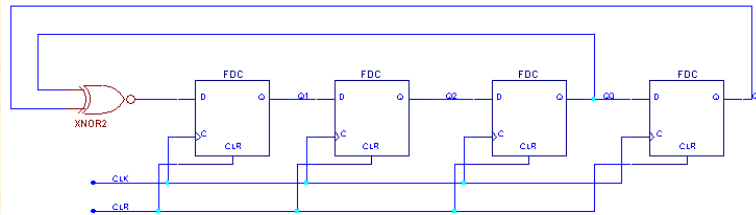
3-Mar-08 (12)

LFSR

- Linear feedback shift register
 - XNOR of certain outputs are fed back to the input
 - it is maximum-length if it has a period of $2^N - 1$
 - lets look at a length 4 maximum length LFSR

3-Mar-08 (13)

4 bit Maximal Length LFSR



3-Mar-08 (14)

4 bit LFSR

- How does it count?

3-Mar-08 (15)

4 bit LFSR

- 0, 1, 3, 7, E, D, B, 6, C, 9, 2, 5, A, 4, 8, 0
- period is $2^4 - 1 = 15$
- can be used as pseudorandom number generators
- note it only has a 2 input XOR gate as the critical path!
 - Compare with an adder's carry chain

3-Mar-08 (16)

Larger maximal length LFSRs (see XAPP052 for more)

- N=4 (4,3)
- N=8 (8,6,5,4)
- N=16 (16,15,13,4)
- N=32 (32,22,2,1)
- N=64 (64,63,61,60)
- N=128 (128,126,101,99)
- N=168 (168,166,153,151)

3-Mar-08 (17)

Primitive polynomials

- A polynomial P of degree m over a finite field F_p is *primitive*, iff P is monic, $P(0) \neq 0$ and $\text{ord}(P(x)) = p^m - 1$.
 - Monic means the coefficient of the term with the highest power is 1
 - If $P(0) \neq 0$, $\text{ord}(P)$ is smallest integer e for which $P(x)$ divides $x^e - 1$
 - E.g. $P(x) = x^4 + x^3 + 1$ is primitive
- The feedback values for an XOR based maximum length LFSR correspond to the primitive polynomial
- Programs which compute primitive polynomials
 - Web based but only supports small n
<http://wims.unice.fr/wims/wims.cgi?session=VX756ADCFA.2&lang=en&module=tool%2Falgebra%2Fprimpoly.en>

3-Mar-08 (19)

LFSR counters

- Maximal LFSR period is $2^N - 1$
 - sometimes we want the period $< 2^N - 1$
 - Arbitrary period counter with shorter period can be made by adding a synchronous reset circuit

3-Mar-08 (18)

n	XNOR from	n	XNOR from	n	XNOR from	n	XNOR from
3	3,2	45	45,44,42,41	87	87,74	129	129,124
4	4,3	46	46,45,26,25	88	88,87,17,16	130	130,127
5	5,3	47	47,42	89	89,51	131	131,130,84,83
6	6,5	48	48,47,21,20	90	90,89,72,71	132	132,130
7	7,6	49	49,40	91	91,90,6,7	133	133,132,82,81
8	8,6,5,4	50	50,49,24,23	92	92,91,80,79	134	134,77
9	9,5	51	51,50,36,35	93	93,91	135	135,124
10	10,7	52	52,49	94	94,73	136	136,135,11,10
11	11,9	53	53,52,38,37	95	95,84	137	137,118
12	12,6,4,1	54	54,53,18,17	96	96,94,49,47	138	138,137,131,130
13	13,4,3,1	55	55,31	97	97,91	139	139,138,134,131
14	14,5,3,1	56	56,55,35,34	98	98,87	140	140,111
15	15,14	57	57,50	99	99,97,54,52	141	141,140,110,109
16	16,15,13,4	58	58,39	100	100,63	142	142,121
17	17,14	59	59,58,38,37	101	101,100,95,94	143	143,142,123,122
18	18,11	60	60,59	102	102,101,38,35	144	144,143,75,74
19	19,6,2,1	61	61,60,46,45	103	103,94	145	145,93
20	20,17	62	62,61,6,5	104	104,103,94,93	146	146,145,87,86
21	21,19	63	63,62	105	105,89	147	147,146,110,109
22	22,21	64	64,63,61,60	106	106,91	148	148,121
23	23,18	65	65,47	107	107,105,44,42	149	149,148,40,39
24	24,23,22,17	66	66,65,57,56	108	108,77	150	150,97
25	25,22	67	67,66,58,57	109	109,108,103,102	151	151,148
26	26,6,2,1	68	68,59	110	110,109,98,97	152	152,151,87,86
27	27,5,2,1	69	69,67,42,40	111	111,101	153	153,152
28	28,25	70	70,69,55,54	112	112,110,69,67	154	154,152,27,25
29	29,27	71	71,65	113	113,104	155	155,154,124,123
30	30,6,4,1	72	72,66,25,19	114	114,113,33,32	156	156,155,41,40
31	31,29	73	73,49	115	115,114,101,100	157	157,150,131,130
32	32,22,2,1	74	74,73,59,58	116	116,115,46,45	158	158,157,132,131
33	33,20	75	75,74,65,64	117	117,115,99,97	159	159,128
34	34,27,2,1	76	76,75,41,40	118	118,85	160	160,159,142,141
35	35,33	77	77,76,47,46	119	119,111	161	161,143
36	36,29	78	78,77,59,58	120	120,113,9,2	162	162,161,75,74
37	37,5,4,3,2,1	79	79,10	121	121,103	163	163,162,104,103
38	38,6,5,1	80	80,79,43,42	122	122,121,83,82	164	164,163,151,150
39	39,35	81	81,77	123	123,121	165	165,164,135,134
40	40,38,21,19	82	82,79,47,44	124	124,87	166	166,165,128,127
41	41,38	83	83,82,38,37	125	125,124,18,17	167	167,161
42	42,41,20,19	84	84,71	126	126,125,90,89	168	168,166,153,151
43	43,42,38,37	85	85,84,58,57	127	127,126		
44	44,43,18,17	86	86,85,74,73	128	128,126,101,99		

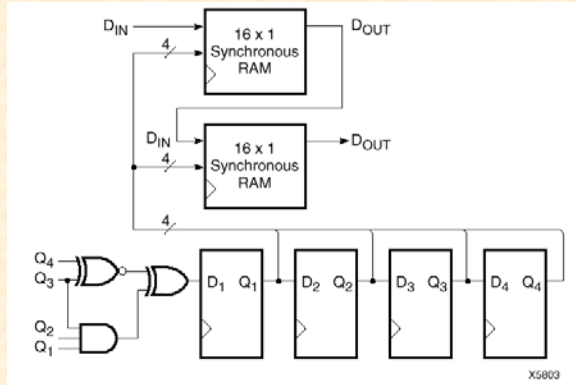
3-Mar-08 (20)

LFSR

- How many slices to implement a
 - 32 bit shift register?
 - 63 bit shift register?

3-Mar-08 (21)

32x1 Shift register using RAM

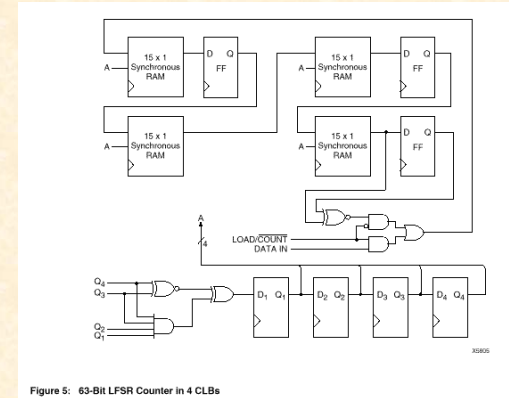


2 Slices

X5803

3-Mar-08 (22)

Efficient 63 bit LFSR



4 Slices

Figure 5: 63-Bit LFSR Counter in 4 CLBs

X5805

3-Mar-08 (23)

63 bit LFSR

- Notice the LOAD/!COUNT line
 - allows loading of the value of the LFSR
- Design can be expanded
 - 127 bit LFSR in 6 slices
 - 159 bit LFSR in seven slices (period is 7e47 or many many times longer than the life of the Universe)
- Speed - no long critical path means design works at max speed

3-Mar-08 (24)

100x8 SR in 26 slices

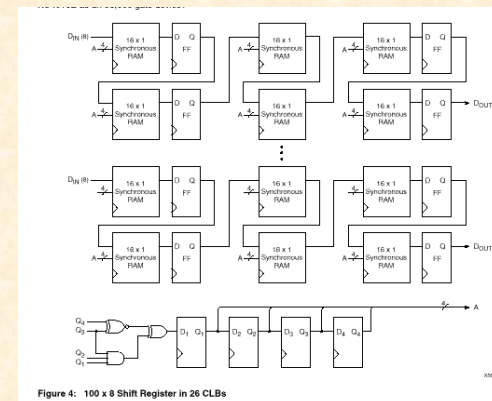


Figure 4: 100 x 8 Shift Register in 26 CLBs

X5804

3-Mar-08 (25)

Virtex SRL

- Virtex has a macro called “Shift Register Lookup Table” (SRL) for implementing SRs

3-Mar-08 (26)

LFSR in Virtex devices

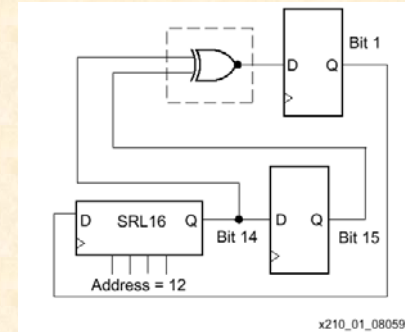


Figure 1: 15-bit LFSR Implemented Using Half of a CLB (one slice)

3-Mar-08 (27)

4 slices (52 bit LFSR)
8 slices (118 bit LFSR)

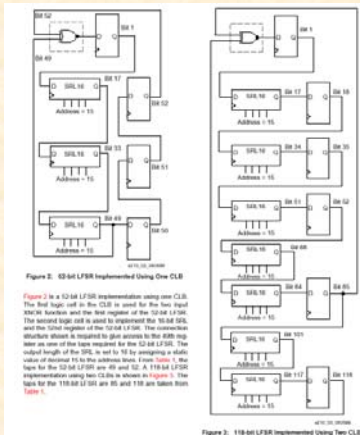


Figure 2: 52-bit LFSR implemented using four CLB slices

Figure 3: 118-bit LFSR implemented using eight CLB slices

3-Mar-08 (28)

LUT as a mux

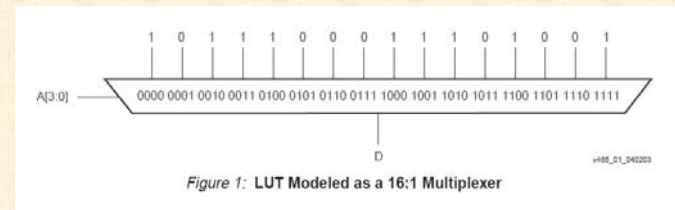


Figure 1: LUT Modeled as a 16:1 Multiplexer

LUT as an addressable SR

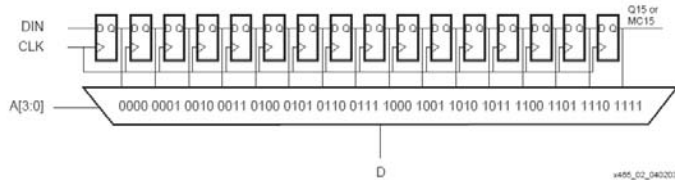


Figure 2: LUT Configured as an Addressable Shift Register

- Length of SR can be controlled using the D output

Virtex 5

- Virtex-5 slice has 4x 6-input LUTs, each LUT can be a 64x1 or 32x2 RAM
- SRL16x2 and SRL32 primitives are available per LUT
- Usage similar to SRL16 examples here (both SR output and LUT contents available though)

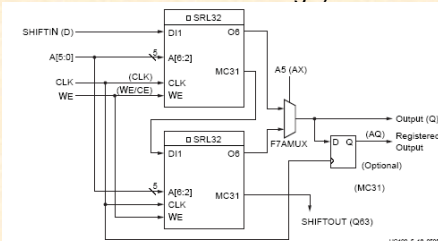


Figure 5-18: 64-bit Shift Register Configuration

SRLC32E

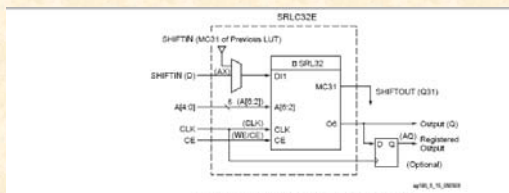


Figure 5-15: 32-bit Shift Register Configuration

Figure 5-15 illustrates an example shift register configuration occupying one function generator.

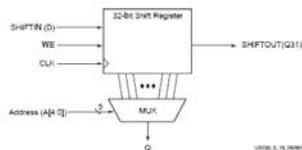


Figure 5-16: Representation of a Shift Register

True Random Number Generator

- Introduction
- Ring Oscillator based noise source
- Alternating Sequence Generator
- Results
- Conclusion

3-Mar-08 (33)

Introduction

- Random number generators (RNGs) are in widespread use, main applications are in
 - Simulation
 - Cryptography
- Propose random number generators for these applications
 - Physical RNG (PRNG) based on oscillator phase noise

3-Mar-08 (34)

What is a random number generator?

- A cryptographically secure random bit generator (CSRBG) is one which produces sequences for which there is no polynomial time algorithm which, on input of the first l bits of the output sequences, can predict the $(l+1)$ st bit of s with a probability significantly greater than 0.5

3-Mar-08 (35)

Applications

- Designed for embedded FPGA applications
 - Small
 - Fast
- For cryptographic applications
 - Secure

3-Mar-08 (36)

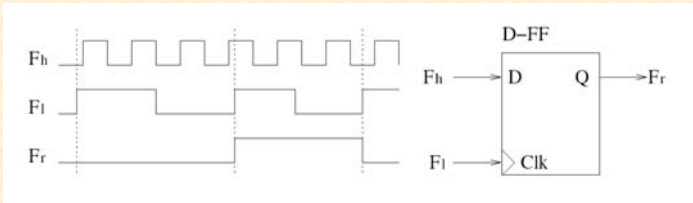
Physical random number generators

- Oscillator sampling
- Direct amplification of transistor/resistor noise
- Discrete time chaos

3-Mar-08 (37)

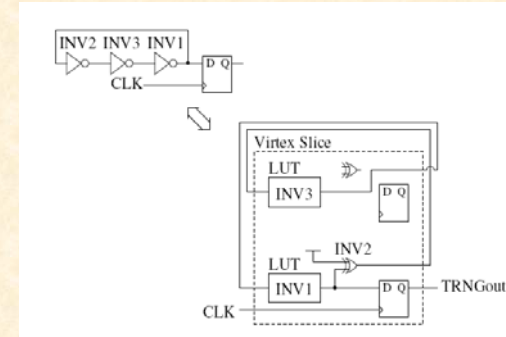
Oscillator phase noise source

- One method to produce a real random number source in an FPGA is to sample a high frequency signal with a low quality low frequency clock
- If the phase noise of the low frequency oscillator is of the same order as the period of the high frequency clock, output is quite random



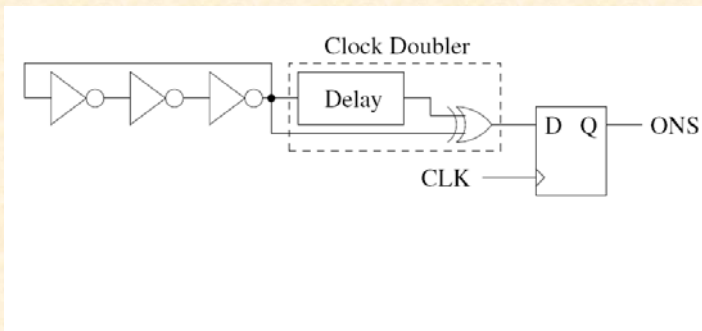
3-Mar-08 (38)

Ring Oscillator based noise source



3-Mar-08 (39)

Increased Randomness via Clock Doubling



3-Mar-08 (40)

Poker Test (ONS only)

- Checks that each m-bit string occurs the correct number of times
- Passed if the result is between 1.03 and 57.4

(iii) Poker test

Let m be a positive integer such that $\lfloor \frac{n}{m} \rfloor \geq 5 \cdot (2^m)$, and let $k = \lfloor \frac{n}{m} \rfloor$. Divide the sequence s into k non-overlapping parts each of length m , and let n_i be the number of occurrences of the i^{th} type of sequence of length m , $1 \leq i \leq 2^m$. The poker test determines whether the sequences of length m each appear approximately the same number of times in s , as would be expected for a random sequence. The statistic used is

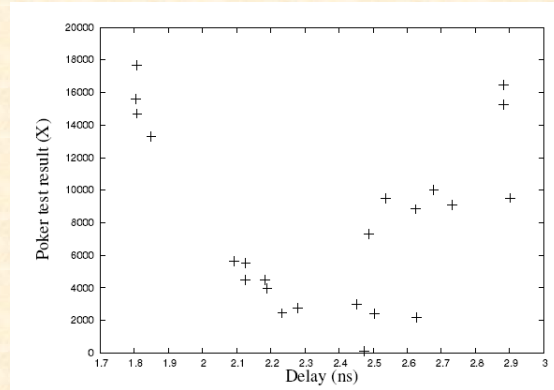
$$X_3 = \frac{2^m}{k} \left(\sum_{i=1}^{2^m} n_i^2 \right) - k \quad (5.3)$$

which approximately follows a χ^2 distribution with $2^m - 1$ degrees of freedom. Note that the poker test is a generalization of the frequency test: setting $m = 1$ in the poker test yields the frequency test.

From Menezes, "Handbook of Applied Cryptography"

3-Mar-08 (41)

Poker Test (ONS only)



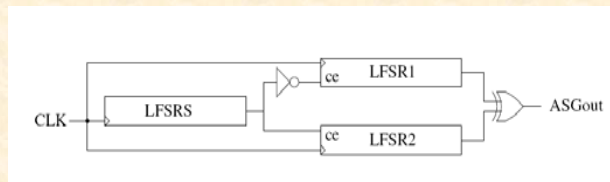
3-Mar-08 (42)

PRNG

- At this point, have a mediocre physical RNG that doesn't pass Poker test
- Can turn into a good RNG by reducing sampling clock rate and passing through a parity filter to deskew non-uniform distribution (Tsoi, Leung and Leong, FCCM03)
 - slow
- How can we make a fast one?

3-Mar-08 (43)

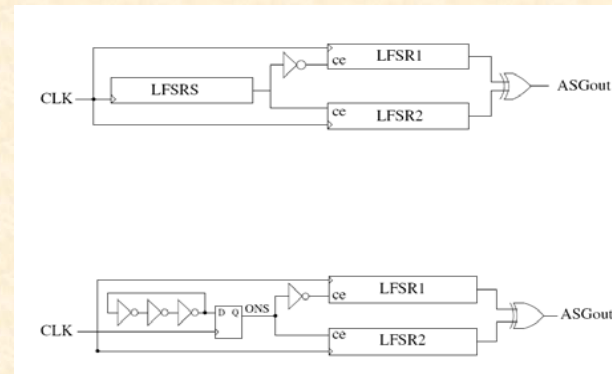
Alternating Step Generator



- Selected among many stream ciphers because of its compact implementation
- Recent research has shown weaknesses in this cipher (but any other can be used)

3-Mar-08 (44)

Modified Alternating Step Generator



3-Mar-08 (45)

LFSRs

- 127 and 129 bit LFSRs based on primitive polynomials with approx equal 1 and 0 coefficients
 - $LFSR1(x) = x^{127} + x^{102} + x^{61} + x^{59} + x^{58} + x^{57} + x^{56} + x^{55} + x^{54} + x^{53} + x^{52} + x^{51} + x^{50} + x^{49} + x^{48} + x^{47} + x^{46} + x^{45} + x^{44} + x^{43} + x^{42} + x^{41} + x^{40} + x^{39} + x^{38} + x^{37} + x^{36} + x^{35} + x^{34} + x^{33} + x^{32} + x^{31} + x^{30} + x^{29} + x^{28} + x^{27} + x^{26} + x^{25} + x^{24} + x^{23} + x^{22} + x^{21} + x^{20} + x^{19} + x^{18} + x^{17} + x^{16} + x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$
 - $LFSR2(x) = x^{129} + x^{119} + x^{62} + x^{61} + x^{60} + x^{59} + x^{58} + x^{57} + x^{56} + x^{55} + x^{54} + x^{53} + x^{52} + x^{51} + x^{50} + x^{49} + x^{48} + x^{47} + x^{46} + x^{45} + x^{44} + x^{43} + x^{42} + x^{41} + x^{40} + x^{39} + x^{38} + x^{37} + x^{36} + x^{35} + x^{34} + x^{33} + x^{32} + x^{31} + x^{30} + x^{29} + x^{28} + x^{27} + x^{26} + x^{25} + x^{24} + x^{23} + x^{22} + x^{21} + x^{20} + x^{19} + x^{18} + x^{17} + x^{16} + x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

3-Mar-08 (46)

Implementation

- Results for Virtex XCV300E-8 (including computer interface)
 - Period 7.482ns, 129 slices (4%), 4 BRAM (12%)
 - High frequency clock can be > 400 MHz (133 MHz used)
 - Ring oscillator freq > 800 MHz
- Reported results are for no clock doubler but doubled version passes tests as well
- Passes Crush, NIST and Diehard tests

3-Mar-08 (47)

NIST and Diehard Tests

Test	P-value	Pass Rate
Frequency	0.145326	0.9900
Block Frequency	0.657933	0.9700
Cusum-Forward	0.383827	1.0000
Cusum-Reverse	0.867692	1.0000
Runs	0.289667	0.9700
Long Run	0.759756	0.9900
Rank	0.514124	0.9900
FFT	0.779188	1.0000
Aperiodic Templates	0.657933	0.9600
Periodic Templates	0.289667	0.9900
Universal	0.162606	1.0000
Approximate Entropy	0.924076	0.9900
Random Excursions	0.637119	0.9565
Serial1	0.534146	1.0000
Serial2	0.262249	1.0000
Lempel Ziv	0.616305	0.9900
Linear Complexity	0.637119	1.0000

Test	P-value
Birthday Spacings	0.310619
Overlapping 5-Permutation (chisqr 66.743792)	0.994677
Overlapping 5-Permutation (chisqr 107.948832)	0.253086
Binary Rank (31x31)	0.155
Binary Rank (32x32)	0.080
Binary Rank (6x8)	0.051318
Bitstream	0.008018
OPSO	0.996754
OQSO	0.011809
DNA	0.050285
Steam Count-the-1	0.066896
Byte Count-the-1	0.040476
parking Lot	0.921990
Min. Distance	0.496703
3D Spheres	0.016095
Squeeze	0.456598
Overlapping Sums	0.080856
Runs up	0.053444
Runs down	0.738119
Craps	0.985720

3-Mar-08 (48)

TRNG Summary

- Combines weak RNG with high speed stream cipher to produce a physical noise source
- Clock doubler increases oscillator phase noise
- Small area, high output rate, good statistical properties
- ASG makes a very low cost implementation, other stream ciphers can be used

3-Mar-08 (49)

Generalized LFSRs

- A variation useful for cryptographic applications
 - Feedback taps are programmable (can be updated after synthesis)
 - Polynomial has high weight so it has many feedback taps requiring high fan-in XNOR gate
- Question: how do we design such a circuit?

3-Mar-08 (50)

Block diagram

3-Mar-08 (51)

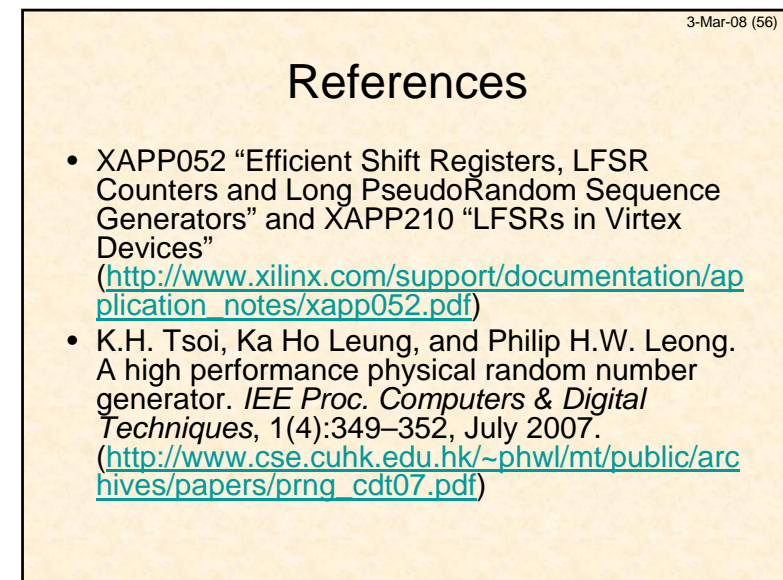
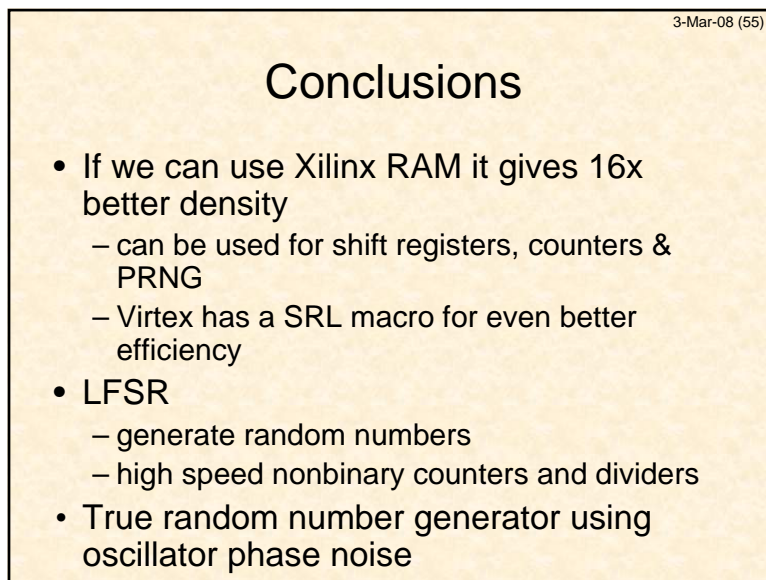
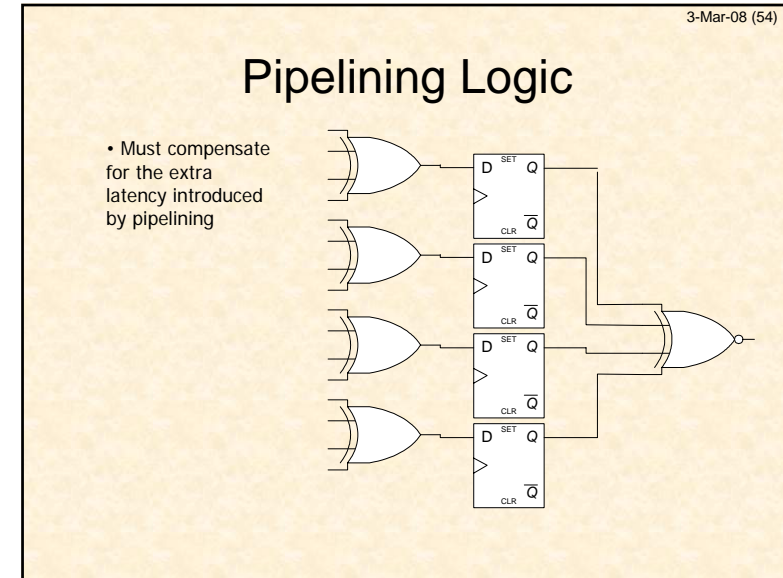
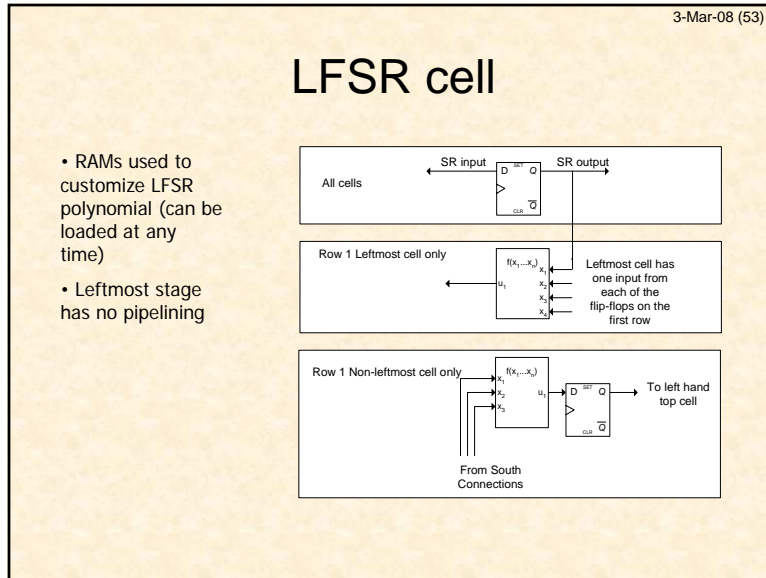
Implementing the LUT

- SR used to program the LUTs to implement the required feedback arrangement **WITHOUT CHANGING THE BITSTREAM**
- Can be reduced from 2 levels of logic to 1 by pipelining

3-Mar-08 (52)

Generalized LFSR Floorplan

- Serpentine arrangement reduces wire lengths



3-Mar-08 (57)

Review Questions

- Make sure you understand how the SRL16 is implemented in an FPGA
- Make sure you can implement an LFSR in VHDL
- When is an LFSR more efficient than a counter for counting applications?