

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

15-Feb-04 (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
 - Generalized LFSR
 - True random number generator

15-Feb-04 (3)

References

- Ideas are from Xilinx applications note XAPP052 “Efficient Shift Registers, LFSR Counters and Long PseudoRandom Sequence Generators” and XAPP210 “LFSRs in Virtex Devices”

15-Feb-04 (4)

Shift register

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

15-Feb-04 (5)

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)

15-Feb-04 (6)

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?

15-Feb-04 (7)

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)

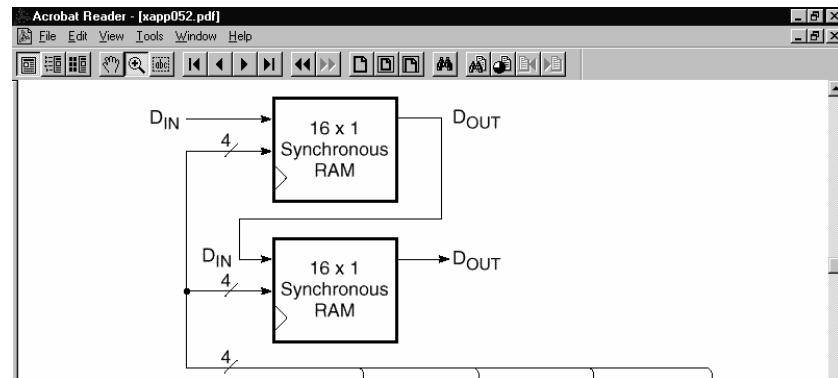
15-Feb-04 (8)

RAM

- Remember
 - Xilinx RAM has 16x better density than CLBs
 - 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

15-Feb-04 (9)

Shift register using RAM



15-Feb-04 (10)

Address generation: counters

- Ripple carry counter
- Synchronous counter
- Gray code counter
- LFSR

- What are the advantages and disadvantages of each?

15-Feb-04 (11)

Counters

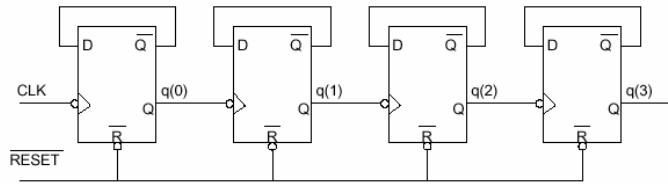


Figure 4.1: A 4 bit ripple counter circuit. The output of one flip-flop clocks the next one, hence the term "ripple" count.

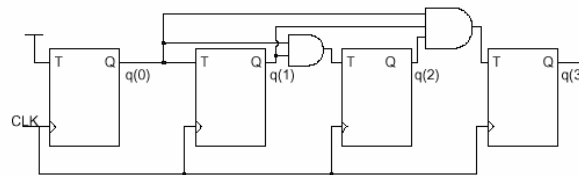


Figure 4.2: A 4 bit synchronous counter circuit. All flip-flops are clocked simultaneously so the outputs change (almost) simultaneously.

15-Feb-04 (12)

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.

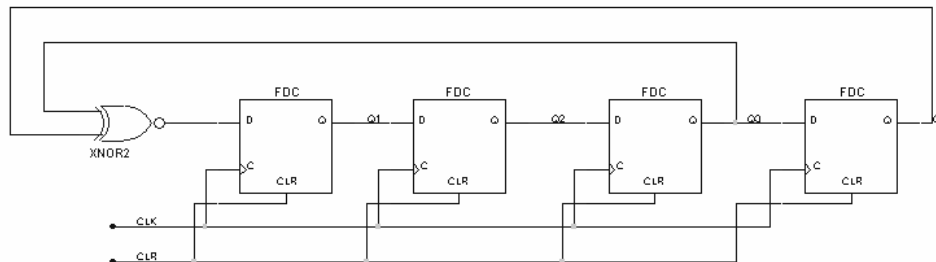
15-Feb-04 (13)

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

15-Feb-04 (14)

4 bit Maximal Length LFSR



15-Feb-04 (15)

4 bit LFSR

- How does it count?

15-Feb-04 (16)

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

15-Feb-04 (17)

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)

15-Feb-04 (18)

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>
 - C program that supports large n
<http://users2.ev1.net/~sduplichan/primitivepolynomials/primitivepolynomials.htm>

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	132,103
7	7,6	49	49,40	91	91,90,8,7	133	133,132,82,81
8	8,8,5,4	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	137,116
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	102,101,36,35	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	67,66,56,57	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	31,28	73	73,48	115	115,114,101,100	157	157,156,131,130
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	36,25	78	78,77,59,58	120	120,113,9,2	162	162,161,75,74
37	37,5,4,3,2,1	79	79,70	121	121,103	163	163,162,104,103
38	38,6,5,1	80	80,79,43,42	122	122,121,63,62	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		

5-Feb-04 (19)

15-Feb-04 (20)

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

15-Feb-04 (25)

100x8 SR in 26 slices

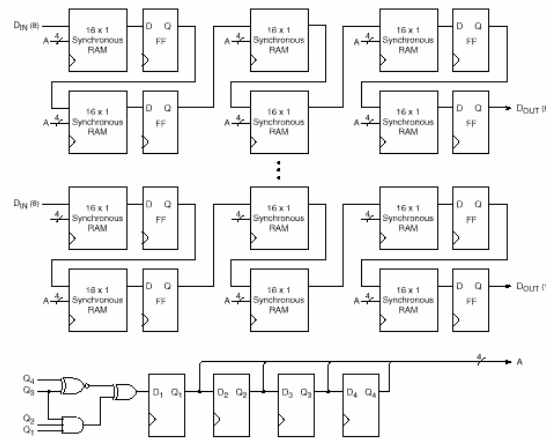


Figure 4: 100 x 8 Shift Register in 26 CLBs

x0804

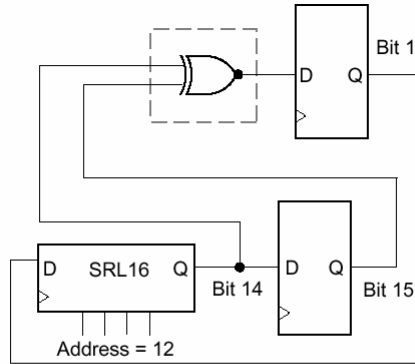
15-Feb-04 (26)

Virtex SRL

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

15-Feb-04 (27)

LFSR in Virtex devices



x210_01_080599

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

15-Feb-04 (28)

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

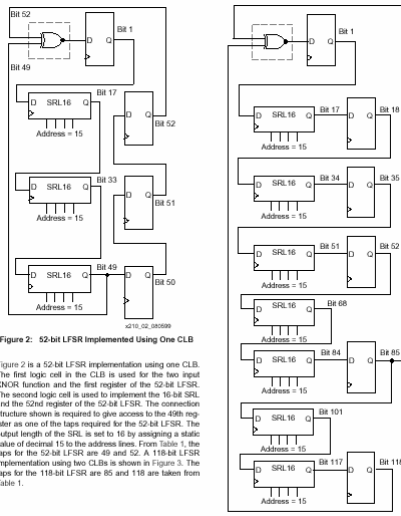


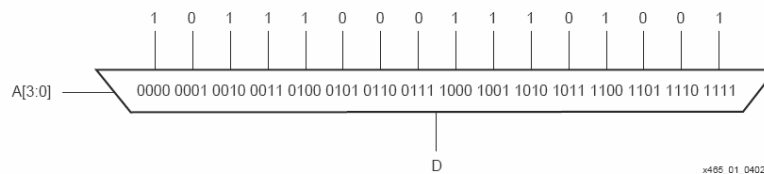
Figure 2: 52-bit LFSR Implemented Using One CLB

Figure 2 is a 52-bit LFSR implementation using one CLB. The first logic cell in the CLB is used for the two input XOR function and the first register of the 52-bit LFSR. The second logic cell is used to implement the 16-bit SRL and the 52nd register of the 52-bit LFSR. The connection structure shown is required to give access to the 49th register as one of the taps required for the 52-bit LFSR. The output length of the SRL is set to 16 by assigning a static value of decimal 15 to the address lines. From Table 1, the taps for the 52-bit LFSR are 49 and 52. A 118-bit LFSR implementation using two CLBs is shown in Figure 3. The taps for the 118-bit LFSR are 95 and 118 are taken from Table 1.

Figure 3: 118-bit LFSR Implemented Using Two CLBs

15-Feb-04 (29)

LUT as a mux

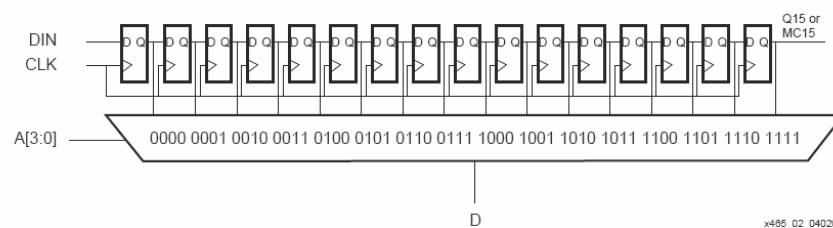


x486_01_040203

Figure 1: LUT Modeled as a 16:1 Multiplexer

15-Feb-04 (30)

LUT as an addressable SR



x486_02_040203

Figure 2: LUT Configured as an Addressable Shift Register

- Length of SR can be controlled using the D output

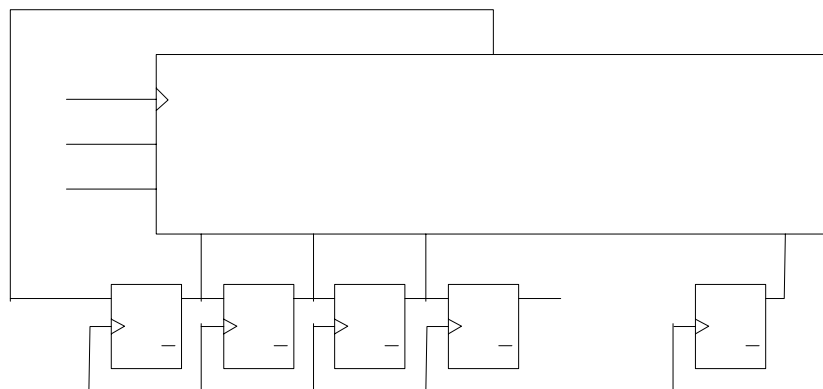
15-Feb-04 (31)

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?

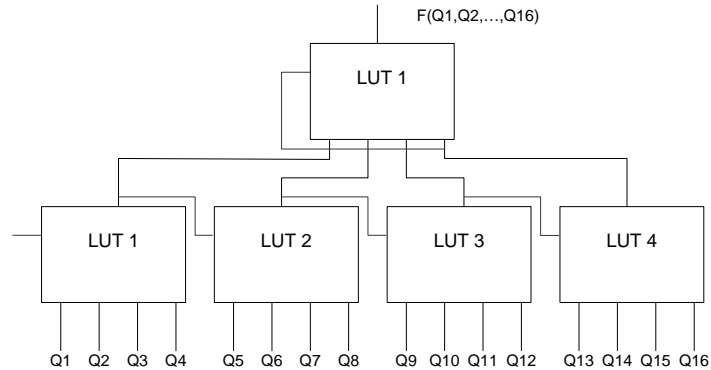
15-Feb-04 (32)

Block diagram



15-Feb-04 (33)

Implementing the LUT

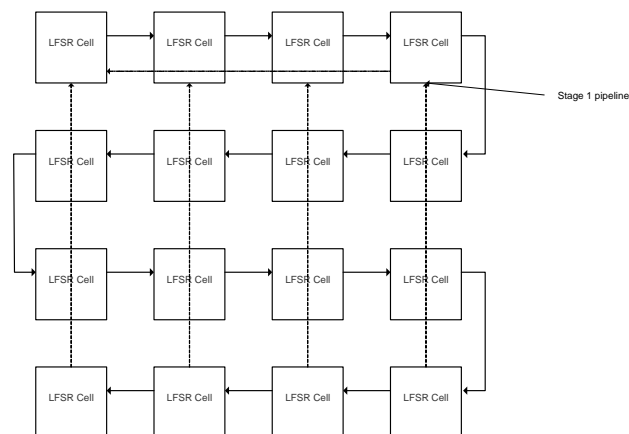


- SR used to program the LUTs to implement the required feedback arrangement **WITHOUT CHANGING THE BITSTREAM**
- Can be reduced from 2 levels of logic to 1 by pipelining

15-Feb-04 (34)

Generalized LFSR Floorplan

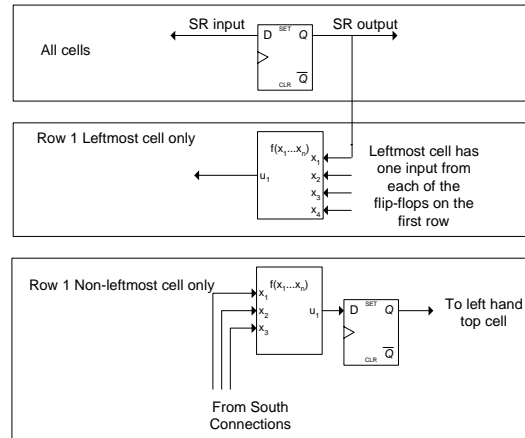
- Serpentine arrangement reduces wire lengths



15-Feb-04 (35)

LFSR cell

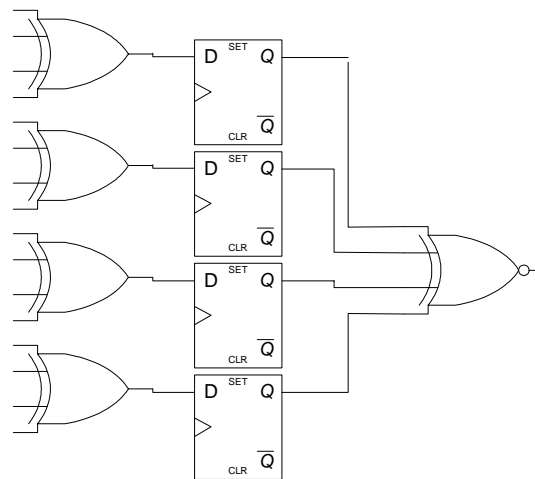
- RAMs used to customize LFSR polynomial (can be loaded at any time)
- Leftmost stage has no pipelining



15-Feb-04 (36)

Pipelining Logic

- Must compensate for the extra latency introduced by pipelining



15-Feb-04 (37)

True random number generator

- 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

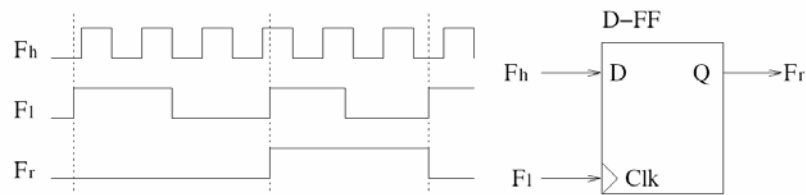


Figure 1: Oscillator sampling using D-type flip-flop.

15-Feb-04 (38)

Low frequency clock with high phase noise

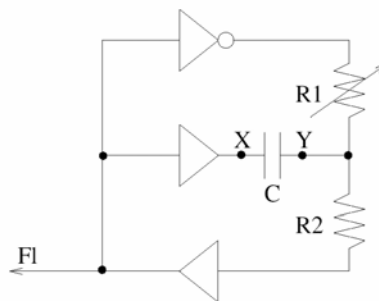


Figure 4: Low frequency clock circuit. Note that the inverter and two buffers in the oscillator circuit were implemented using the input/output blocks (IOBs) of the FPGA.

15-Feb-04 (39)

Low frequency clock

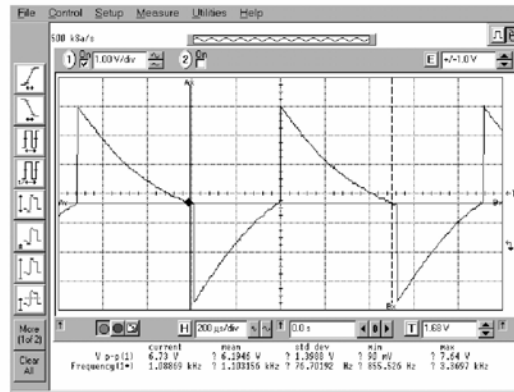


Figure 5: Oscilloscope trace of oscillator node Y.

15-Feb-04 (40)

Conclusions

- If we can use Xilinx RAM it gives 16x better density
 - can be used for shift registers, counters & PRNG
 - Virtex has a SRL macro for even better efficiency
- LFSR
 - generate random numbers
 - high speed nonbinary counters and dividers
- True random number generator using oscillator phase noise