

Shift Registers and Counters

By
Lakshmi

*Things that appear hard are normally trivial,
If you do the obvious in the obvious way.
After all, how "tough" can "easy" be?*
-- Anonymous

What is an LFSR?

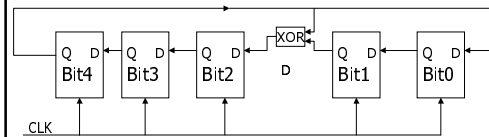


- Linear Feedback Shift Register
- An N-bit LFSR counts through all non-zero bit patterns in a pseudo-random order.
- Useful for the Project!!!
 - Random Number Generator for collision back-off
 - Error Checking

Tips on LFSR...

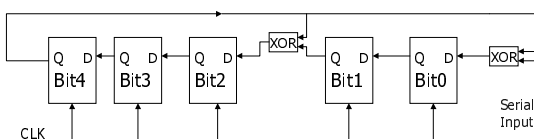
- Things to do
 - Place a linear array of D Flip Flops and feedback the last output to the first. (FF(N-1) output to FF(0) input)
 - For an N-bit LFSR look at the table in Page 6 for a polynomial starting with x^N .
 - For every term x^K in the polynomial compute an XOR of outputs of FF(N-1) and FF(K-1) and feed it to FF(K) as input (FF numbered from 0 to N-1)
- Things NOT to worry about
 - What is a Galois Field?
 - What is an irreducible polynomial?

An Example 5-bit LFSR



- The polynomial from the table is x^5+x^2+1
- Only x^2 is present in the polynomial
- take an XOR of Bit4 and Bit1 and feed it to Bit2 as input.
- Magic!!! You have a 5-bit LFSR in front of you**

Parity Computation



- Plug in the actual bits serially(one after another)
- The final bits in the flip flops are the parity bits
- Note: The Last N(=5) bits of the serial input have to be 0**
- This is for computing the N parity bits of the serial input**

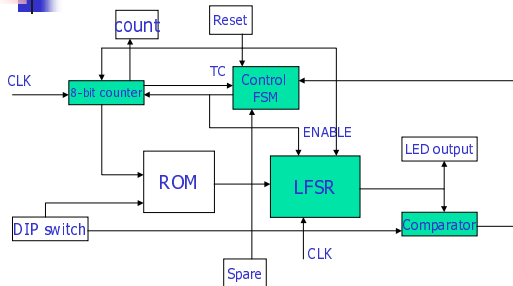
Magic Error Checking



- Consider a 4-bit LFSR
- Serial Input: 11001000111
(Note 11= 15-4 bits)
- 11001000111 0000 ->1010
- 11001000111 1010 ->0000
- Introduce error in 7th bit
- 11000000111 1010 ->0111
- 00001000000 1010 ->0111
- Isn't 0111 equal to 7?

Figure out how lazy I am. Refer to error correction example in the handout!!!

What to do in the lab?



Modes of Operation

- Mode 1:
 - LFSR would shift 256 bits
 - Display final parity in 8 LEDs
- Mode 0:
 - Process sequence until pattern in DIP switch is equal to parity bits
 - Final counter will be displayed in LEDs (**position of error**)
 - If output is "0" then no errors in message
- Mode controlled by SPARE button

Things to do

- Run MODE=1 with all DIP switches set to 0. Compute parity bits as shown in LEDs.
- Enter parity bits on the switches, run MODE=1 and observe the output. **Is it 0?**
- Introduce only one error in the ROM –**modify ROM schematic**
- Use MODE=1 operation to detect the error.
- Use **MODE=0** operation to find position of error.

Is there anything easier than copying?

Control FSM

- Want Extra Credit! Do this logic as pre-lab.**
- 3 inputs: TC output of counter, comparator and MODE from SPARE button
- 2 states:
 - ACTIVE state: upon RESET, enables the counter and LFSR
 - Waits until either TC from counter or PAR=LED signal from comparator (if MODE=0) and enters DONE state
 - DONE state: Can leave this state only thru RESET
- Hooray!!! Need only 1 FF for the FSM

Things to know

- The last 8 bits of the ROM are set to 0 and are always read from the DIP switches
- Galois fields knowledge is useless for the lab!
- Xchecker cable clock runs at 1Mhz makes it difficult to observe what's going on. Use of **16-bit counter to divide clock by 65536!**
- Want to observe cycle-by-cycle, use **Apply button** in Hardware Debugger but should **bypass the 2¹⁶ bit counter**
- To change ROM contents use INIT attribute by clicking on a ROM block

Before you Wake up!



- Do Control FSM logic as pre-lab
- Learn how to build an 8-bit LFSR
- Read the lab sheet twice before starting the lab
- Get your understanding about parity bits and error correction clear

*Every closed eye is sleeping, and every open eye is not seeing.
--Bill Cosby*