Linear Feedback Shift Register Pseudo-Random Number Generator

Computação Reconfigurável

João Ferreira, 93305
Roberto Graça, 93020

1. Summary

With a linear-feedback shift register (LFSR) it is possible to generate a sequence of pseudo-random numbers, on request, given a seed. The LFSR chosen for this project is a 32-bit Fibonacci LFSR. Although the Galois LFSR has less propagation delay and can run at a faster frequency, the Fibonacci LFSR has the advantage that most of the bits that are going to be used on the next state are visible in the shift register and that makes it easier to implement. Using the same seed, the developed specialised hardware was tested against a software implementation to validate the results and compare the computation time.

2. Algorithm

The algorithm starts by taking the given 32-bit seed and storing it in a vector. Then, every clock cycle the vector changes its state. The bit positions that affect the next state are called “taps”. In order to achieve a maximum-length LFSR, i.e., a LFSR that is capable of generating a sequence of 2^31-1 distinct values before repeating that sequence over again, we choose the following taps, [31,30,29,9], which one corresponds to a bit of the vector. The new state is composed of the previous state shifted by 1 bit to the right and a new bit obtained by applying the formula:

state(31) xor state(30) xor state(29) xor state(9)

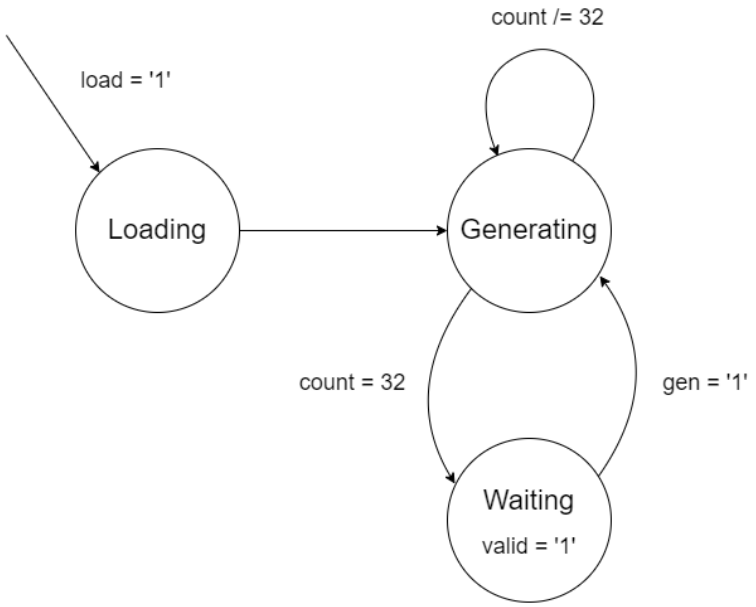
The state transition is as shown below:

```
new_state <= (state(31) xor state(30) xor state(29) xor state(9)) & state(31 downto 1);
```

After 32 clock cycles, a new number is generated.
More can be learned at https://en.wikipedia.org/wiki/Linear-feedback_shift_register

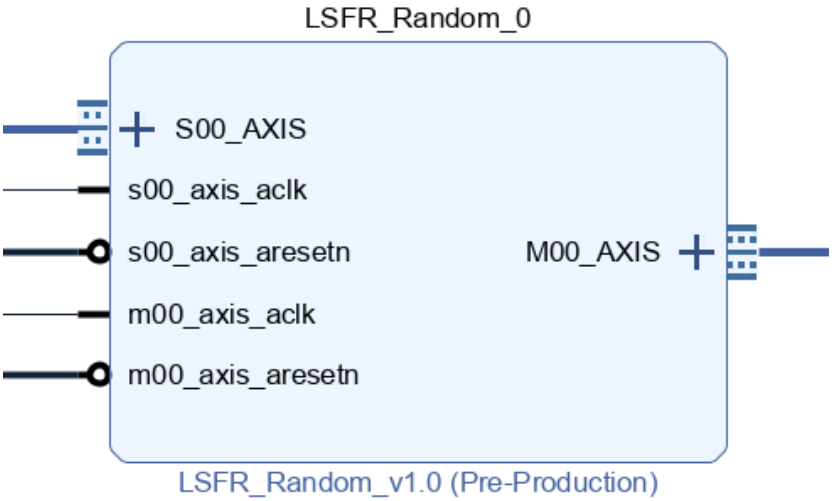
The following diagram represents the state machine of the coprocessor. The “load” signal can be asserted at any time, doing so a new seed is loaded (Loading state). In the next cycle, a new number starts to get generated (Generating). After 32 clock cycles the number

is ready to be consumed (Waiting), at that time a new random number can be generated by asserting the signal “gen”.



State machine of LFSR random number generator.

3. Block Diagram



The coprocessor communicates with the soft processor through AXIStream interfaces. The slave interface is responsible to receive the seed from the processor, at any moment the seed might be loaded. The master interface forwards the newly generated random number when a new number is available and the processor is ready to receive it. Immediately after the seed being loaded or the random number being read the coprocessor starts to generate a new number.

4. Results

To test the efficiency of both software and hardware implementations, a program was written to generate 4000 numbers using both implementations. The times obtained were 187041 microseconds for the software implementation and 1321 microseconds for the hardware implementation.
After analysing the results, we can observe that the hardware implementation is about 140 times faster than the software implementation. In conclusion, because there is a large gap between the timing results of both implementations this operation is most suited to be executed on hardware.

5. Contributions and auto-evaluation

Auto-evaluation -> 19
João Ferreira -> 50 %
Roberto Graça -> 50 %