Implementación de un Shrinking Generator y comunicación Serial para la generación de números aleatorios

Carlos Álvaro González E.

Manuel Felipe Pineda L.

Universidad Tecnológica de Pereira
2012

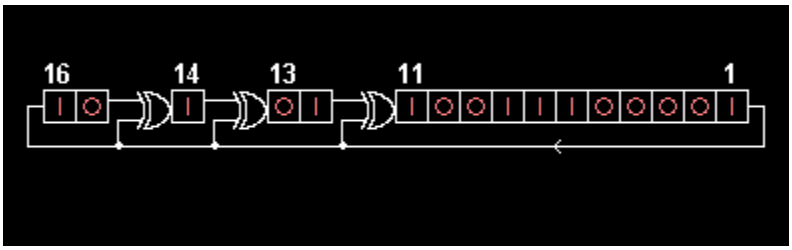
1. Introducción/Conceptos Básicos

-Shrinking Generator: Traducido como Generador de Reducción, es una forma de generar números pseudo-aleatorios, diseñado para ser usado en un Cifrador de Flujo, Fue publicado en Crypto en 1993 por Don Coppersmith, Hugo Krawczyk y Yishay Mansour. Utiliza dos registros de corrimiento de retroalimentación lineal (Linear Feedback Shift Registers), uno para generar los bits de salida, y otro para escoger que bits se mostraran a la salida (si el bit del segundo registro es '1', el bit generado por el primer registro se muestra a la salida, sino, se descarta), ambos registros llevan un reloj. Si los LFSR son coprimos en tamaño, se tiene que el Shrinking Generator es criptográficamente seguro.

-Linear Feedback Shift Register: abreviado LFSR**,** traducido registro de corrimiento de retroalimentación lineal, es un registro de corrimiento que utiliza como entrada una función lineal de sus estados anteriores, el mas usado es un XOR entre los últimos dos estados del registro para determinar la entrada, en caso de que el registro fuera por ejemplo "1001", su siguiente estado sería "1100", y el siguiente "0110", etc. El número inicial que alimenta el registro se conoce como semilla, y se planta solo una vez.

-Configuración de Galois para LFSR: Esta configuración provee la misma salida que un LFSR Común, pero con un desfase en el tiempo (la configuración lleva un reloj), en ella existen posiciones de bits conocidas como "grifos" o "llaves", todos los bits que no estén en una posición de grifo son corridos de manera natural, sin alterarse, en cambio a los bits que están en posiciones de grifo se les realiza un XOR con la salida actual del registro antes de realizar el corrimiento, además de eso, el bit de salida se convierte en el bit de entrada, se tiene un polinomio para determinar las posiciones que son grifos, el polinomio $x^{32}+x^{21}+x^{16}$, indicaría que las posiciones 32, 21 y 16 son grifos, esto se puede expresar en bits con un registro en el cual la posición 32, 21 y 16 son '1', mientras el resto de posiciones son '0'. Fue nombrado tras el matemático Francés Évariste Galois

Ejemplo:



LFSR de 16 bits con configuración de Galois en la cual las posiciones 15, 14 y 12 son posiciones de grifo. (se podrían representar por el polinomio $x^{15}+x^{14}+x^{12}$)

-Máquina de Estados: Es un modelo matemático utilizado tanto en programación como en el diseño de circuitos lógicos, que se basa en una máquina abstracta que se puede encontrar en uno de varios estados (la lista de estados debe ser finita, y la máquina solo puede estar en un estado al tiempo), y cambia de estado si se activa o cumple una condición (a esto se le llama transición)

-Comunicación serial: Tipo de comunicación entre interfaces electrónicas (computadores, procesadores, FPGA's, cualquier tipo de dispositivo electrónico que pueda enviar a o recibir datos), en la cual los datos no se envían de manera paralela, sino bit por bit, la comunicación seria suele hacerse a través de un puerto RS232, que no solo tiene un pin para la transmisión de datos, sino también: uno para expresar que se recibió de manera satisfactoria, uno para expresar que se transmitió de manera satisfactoria, uno para expresar que el set de datos está listo, uno para expresar una petición para enviar, uno para expresar permiso para enviar, y por supuesto tierra, no es necesario usar todos los pines, se pueden usar solo los que necesite el diseño. A la hora de realizar e implementar el diseño, se deben tener en cuenta las capacidades de transmisión del cable y el puerto que se estén usando (velocidad de transmisión)

2. Objetivos

-General: Implementar en una tarjeta de desarrollo (Específicamente una Nexys 2 de Digilent) un Shrinking generator con comunicación serial a un PC que siembre la semilla y reciba los números pseudo-aleatorios

-Específicos:

- Implementar Linear Feedback Shift Registers en un lenguaje de descripción de Hardware (VHDL)
- Realizar la implementación de un puerto RS232 a través de VHDL
- Integrar los componentes anteriores a través de una unidad de control, también realizada en VHDL

3. Diseño e Implementación

Nuestro diseño está planteado de manera modular, a continuación se describe cada módulo, sus entradas y salidas, y su funcionamiento.

-Divisor de Frecuencia a 115200Hz:

Entradas:

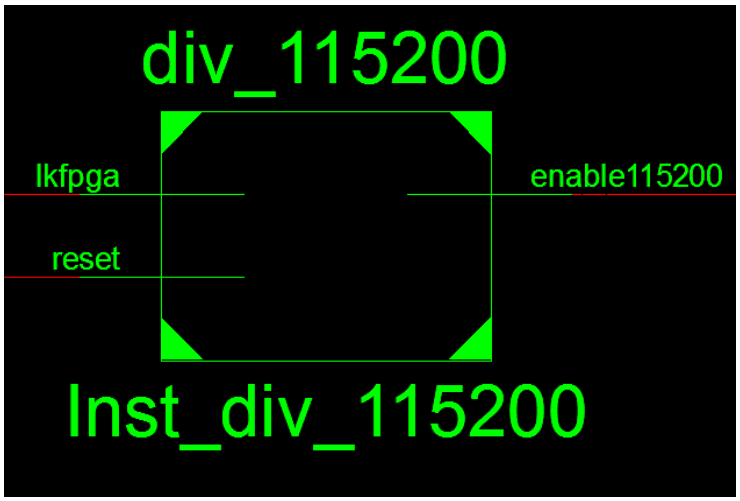
1. Reloj (clkFPGA): Reloj de 50 MHz proveniente de la FPGA
2. Reset (reset): Activo alto, detiene el funcionamiento del dispositivo

Salidas:

1. Reloj de 115200: Reloj de 115200HZ

Descripción: un simple divisor de frecuencia que funciona para adaptarse a una de las velocidades de transmisión de datos del cable para el puerto RS232, en nuestro caso 115200Hz

Diagrama de Bloque:



-Módulo TX:

Entradas:

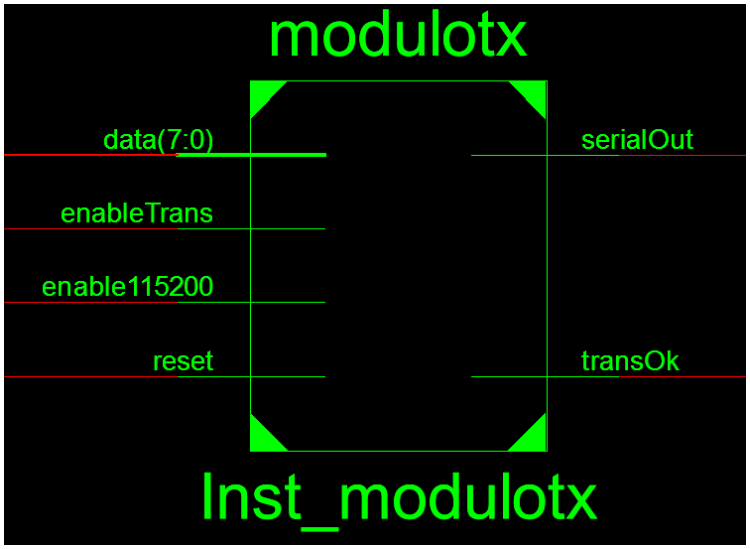
1. Enable/Reloj (enable115200): Proveniente del divisor de frecuencia de 115200Hz, representa la velocidad de transmisión de datos
2. Reset (reset): Activo alto, detiene el funcionamiento del dispositivo
3. Activar Transmisión (enableTrans): Activa la transmisión serial de datos
4. Data (data): recibe el dato de forma paralela, que luego será transmitido en forma serial

Salidas:

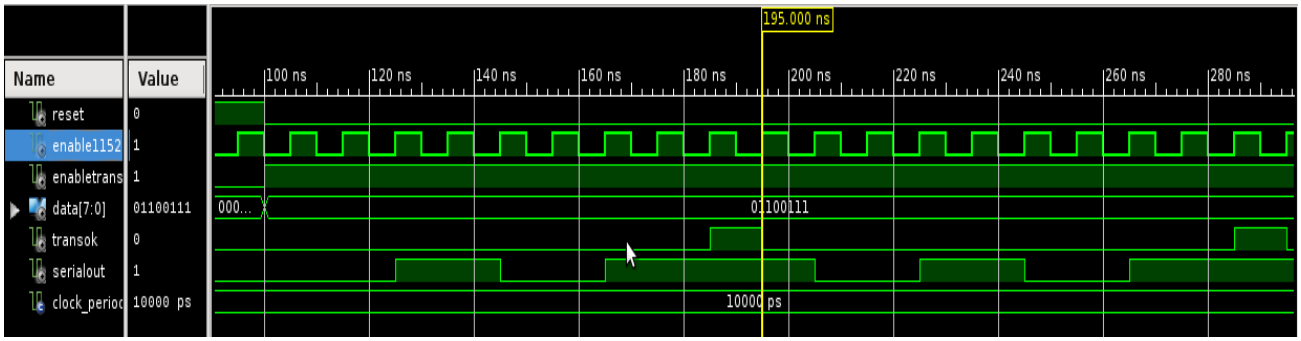
1. Salida Serial (serialOut): salida serial del dato anteriormente en paralelo
2. Transmisión exitosa (transOk): bit que indica que la transmisión se realizó de forma exitosa

Descripción: Módulo de transmisión del RS232, velocidad de transmisión 115200Hz, utiliza pines para autorizar transmisión, y para indicar que la transmisión se realizó con éxito, cada transmisión es de 8 bits de longitud, es decir cada dato paralelo recibido, es de 8 bits

Diagrama de Bloque:



Test Bench:



-Módulo RX:

Entradas:

1. Enable/Reloj (enable115200): Proveniente del divisor de frecuencia de 115200Hz, representa la velocidad de transmisión de datos
2. Reset (reset): Activo alto, detiene el funcionamiento del dispositivo
3. Habilitar Recepción (enableRecep): activa la recepción de datos seriales
4. Entrada Serial (serialIn): dato serial que es recibido y será convertido a paralelo

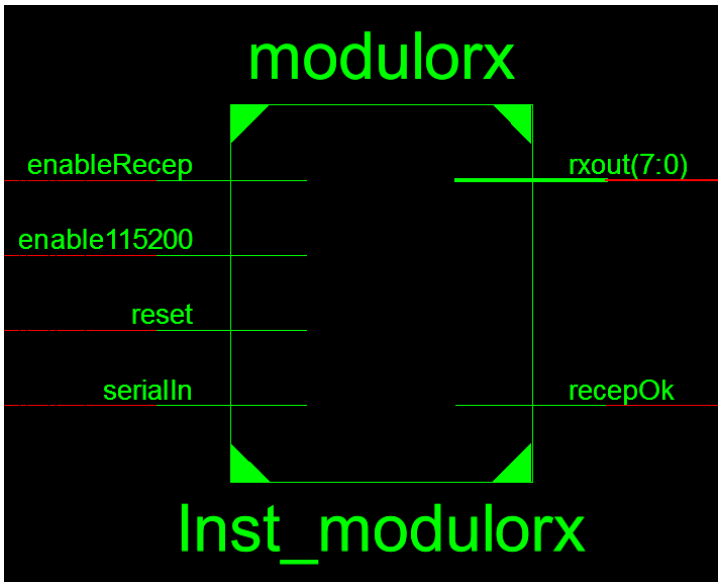
Salidas:

1. Data Out (rxOut): Data paralelo de salida
2. Recepción exitosa (receOk): bit que indica que la recepción del dato serial fue exitosa

Descripción: Módulo de recepción del RS232, velocidad de transmisión 115200Hz, utiliza pines para autorizar recepción, y para indicar que la recepción del dato fue

exitosa, cada transmisión es de 8 bits de longitud, es decir, se reciben siempre 8 bits en serie que luego son enviados de forma paralela

Diagrama de Bloques:



-LFSR 128 Bits

Entradas:

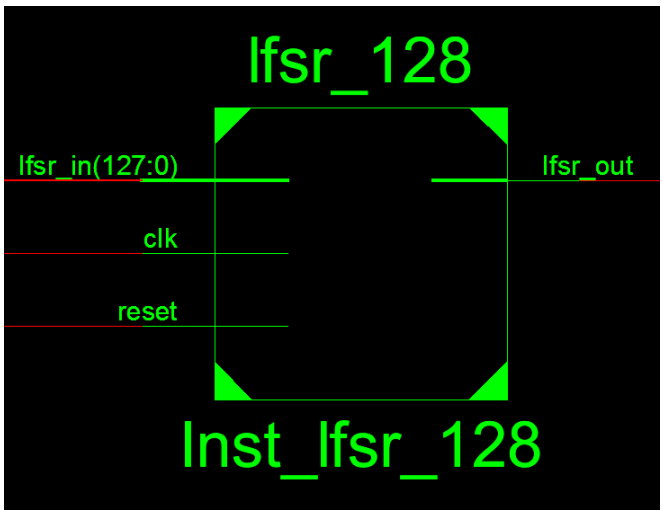
1. Reloj (clk): Reloj de la FPGA, velocidad de 50MHz
2. Reset (reset): Activo alto, detiene el funcionamiento del dispositivo
3. Semilla (lfsr_in): primera semilla para alimentar el lfsr de 128 bits

Salidas:

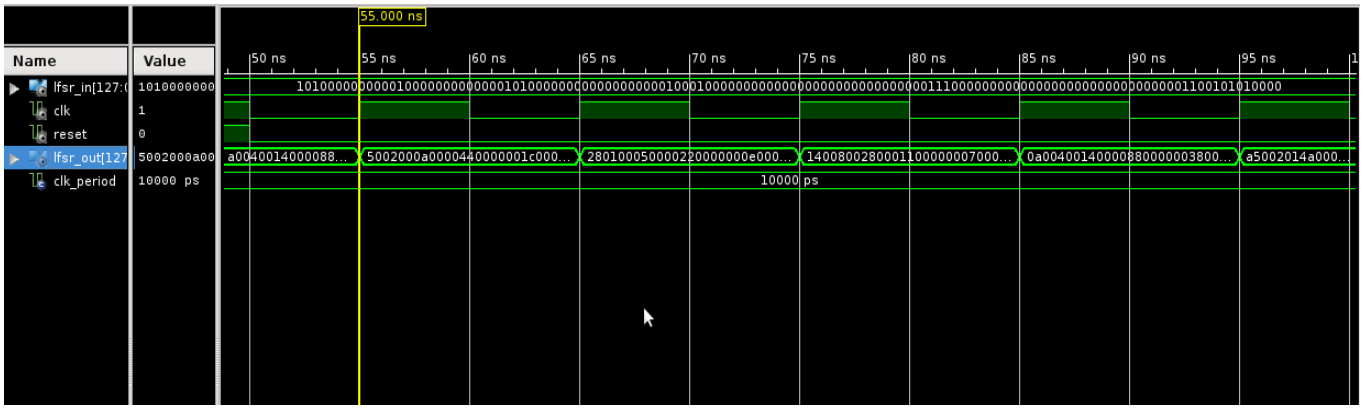
1. Dato de Salida (lfsr_out): salida de 128 bits del LFSR

Descripción: Linear Feedback Shift Register de 128 bits con configuración de Galois y velocidad de corrimiento de 50MHz (es decir realiza un corrimiento cada $2x10^{-8}$ segundos), el polinomio utilizado para seleccionar las posiciones de grifo está representado por el hexadecimal ***A0000014000000000000000000000000***, y es un polinomio que debería proveer un periodo máximo según una Nota de Aplicación (Application Note) provista por Xilinx, llamada *Efficient Shift Registers, LFSR Counters, and Long PseudoRandom Sequence Generators*, escrita por Peter Alfke.

Diagrama de Bloque:



Test Bench:



-LFSR 127 Bits

Entradas:

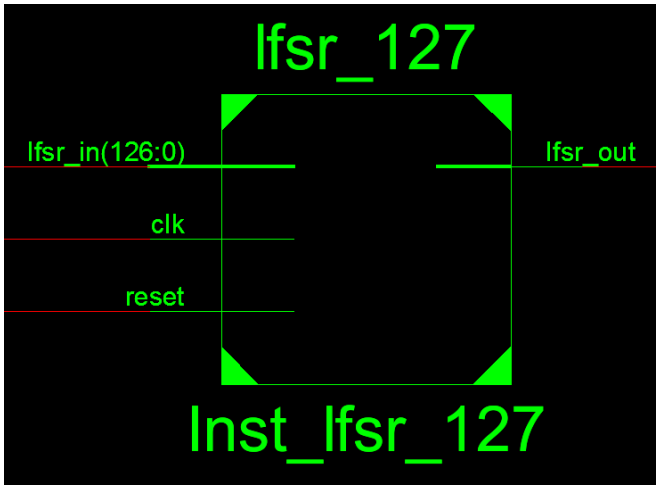
1. Reloj (clk): Reloj de la FPGA, velocidad de 50MHz
2. Reset (reset): Activo alto, detiene el funcionamiento del dispositivo
3. Semilla (lfsr_in): primera semilla para alimentar el lfsr de 127 bits

Salidas:

1. Dato de Salida (lfsr_out): salida de 128 bits del LFSR

Decripción: En base funciona igual que el LFSR de 128 Bits, solo que con una longitud de 127 bits, esto debido a que para obtener la mayor seguridad criptográfica los dos LFSR usados en el shrinking generator deben ser coprimos, el polinomio para las posiciones de Grifo (sacado de la misma fuente que el del anterior) para este LFSR está representado por el hexadecimal ***60000000000000000000000000000000*****,** desperdiciando el primer bit (ya que este registro es de 127 bits)

Diagrama de Bloque:



-Codificador de Registro

Entradas:

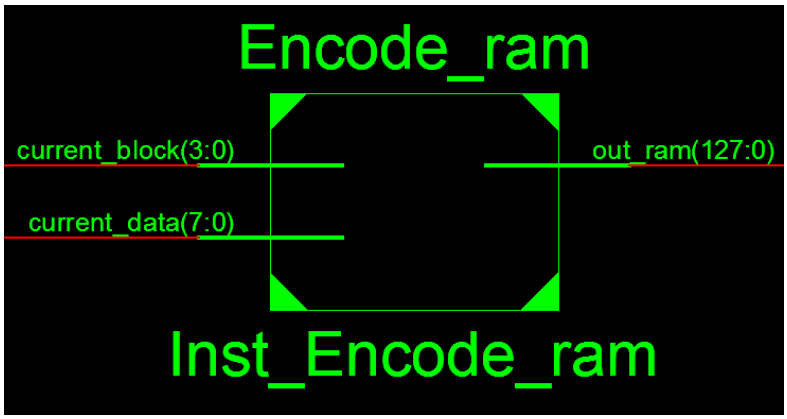
1. Dato Actual (current_data): Dato actual de 8 bits recibido del RX
2. Bloque Actual (current_block): Indica el Bloque Actual de los 128 bits a ser escrito (cada bloque tiene 8 bits de longitud)

Salidas:

1. Dato de Salida (out_ram): salida de 128 bits (llena) para enviar al LFSR 128

Descripción: Este módulo se encarga de recibir los datos que envía el RX, de 8 bits cada uno, y los va concatenando en un registro de 128 bits que luego será enviado al LFSR128, es asíncrono.

Diagrama de Bloque:



-Decodificador de Registro

Entradas:

1. Bloque Actual (current_block): Bloque actual de 8 bits perteneciente a la entrada de 128 bits

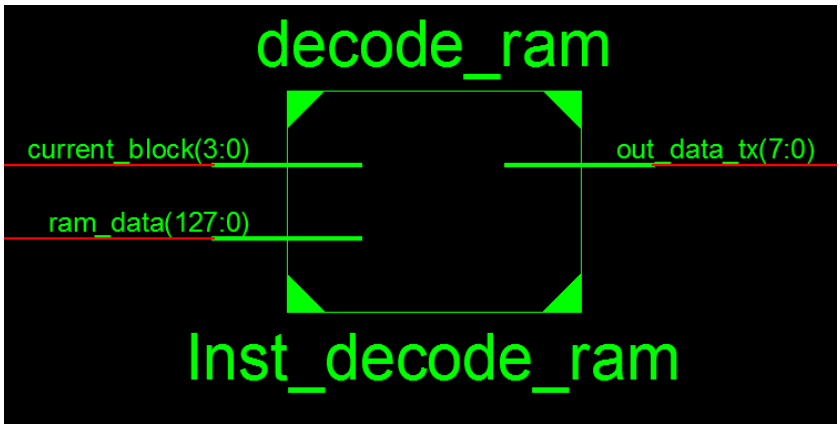
2. Dato de Ram (ram_data): Dato de 128 bits proveniente de la RAM que será enviado en bloques de 8 bits

Salidas:

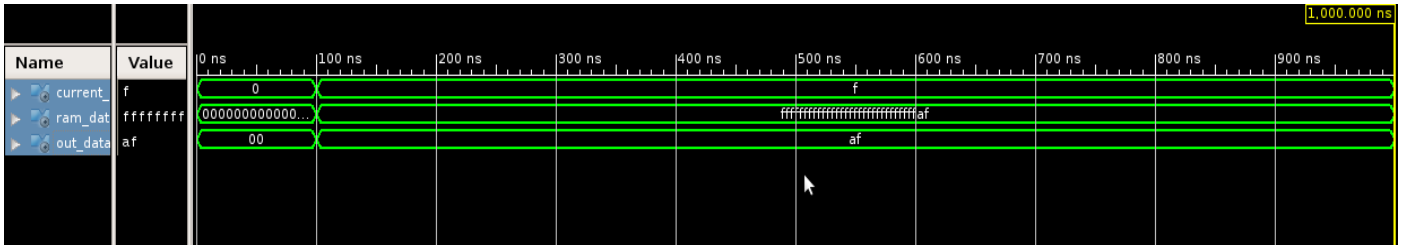
1. Dato de Salida (out_data_tx): salida de 8 bits a ser transmitida por el tx

Descripción: Contrario al codificador, este módulo toma un dato de 128 bits y lo envía en bloques de 8 bits para ser transmitidos por el tx, es asíncrono.

Diagrama de Bloque:



Test Bench:



-RAM:

Entradas:

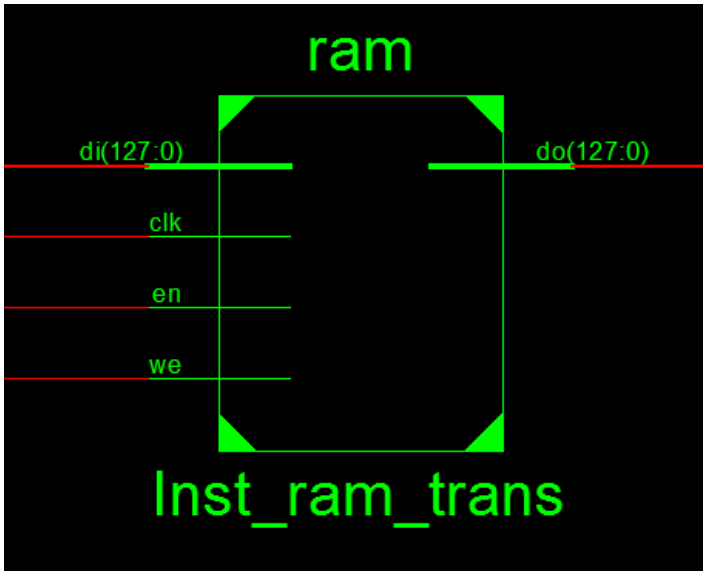
1. Reloj (clk): Reloj de 50MHz de la FPGA
2. Write Enable (we): Activa o desactiva el permiso de escritura en RAM
3. Enable (en): Habilita o deshabilita el módulo
4. Dato de Entrada (di): dato de entrada para escritura

Salidas:

1. Dato de Salida (do): dato de salida de 128 bits leído de RAM.

Descripción: Ram del tipo “lectura primero” tal y como está descrita en el capítulo 3 del manual XST, “HDL Coding Tachniques”, esta Ram en específico no necesita address pues solo tiene un registro

Diagrama de Bloque:



-Unidad de Control

Entradas:

1. Reset (reset): detiene la operación del módulo (Activo alto)
2. Entrada Serial (serial_in): Entrada serial, que será la futura semilla del lfsr128
3. Reloj (clk): Reloj de 50MHz proveniente de la FPGA
4. Recepción Exitosa (recep_ok): Indica que la recepción del dato serial fue exitosa
5. Llenado exitoso (fill_ok): Indica que el llenado del Shrinking Generator fue exitoso
6. Transmisión exitosa (trans_ok): Indica que la transmisión desde el tx fue realizada de manera exitosa

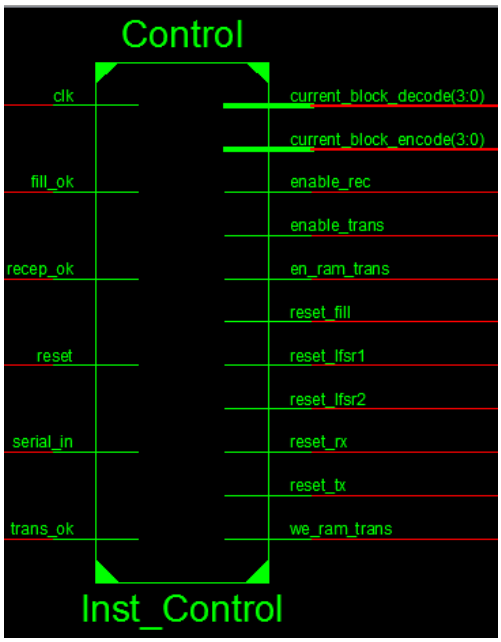
Salidas:

1. Enable Reception (enable_rec): Habilita o deshabilita la recepción de datos
2. Bloque Actual para codificacion (current_block_encode): indica el bloque de 8 bits para el codificador de registro
3. Habilitador de escritura en RAM (we_ram_trans): Habilita o Deshabilita la escritura en RAM
4. Habilitador de RAM (en_ram_trans): Habilita el módulo RAM
5. Reset LFSR 128 (reset_lfsr1): Detiene la operación del módulo LFSR128
6. Reset LFSR 127 (reset_lfsr2): Detiene la operación del módulo LFSR127
7. Reset Shrinking Generator (reset_fill): Detiene la operación del módulo Shrinking Generator
8. Reset TX (reset_tx): Detiene la operación del módulo TX
9. Reset RX (reset_rx): Detiene la operación del módulo RX

10. Bloque actual para decodificación (current_block_decode): indica el bloque de 8 bits para el decodificador de registro
11. Habilitar Transmisión (enable_trans): Habilita la transmisión del TX.

Descripción: Es en base una máquina de estados que se puede encontrar en uno de cuatro estados distintos: recibiendo los datos de manera serial desde un ente externo (receiving), llenando la semilla para el Shrinking Generator (Filling), Utilizando el Shrinking Generator (Shrinking), o Enviando los datos (Sending). Es la encargada de controlar la habilitación y deshabilitación de cada uno de los otros módulos que intervienen tanto en la comunicación serial, como en la generación de los números aleatorios, de acuerdo a cada uno de estos estados.

Diagrama de Bloque:



-Shrinking Generator:

Entradas:

1. Reset (reset): Detiene la Operación del módulo (Activo Alto)
2. Entrada de LFSR128 (lfsr1): bit enviado a partir del LFSR128
3. Entrada de LFSR127 (lfsr2): bit enviado a partir del LFSR127
4. Estado de la RAM (ram_state): estado de la Ram en un instante de tiempo (se utiliza para saber cuando está llena la memoria)

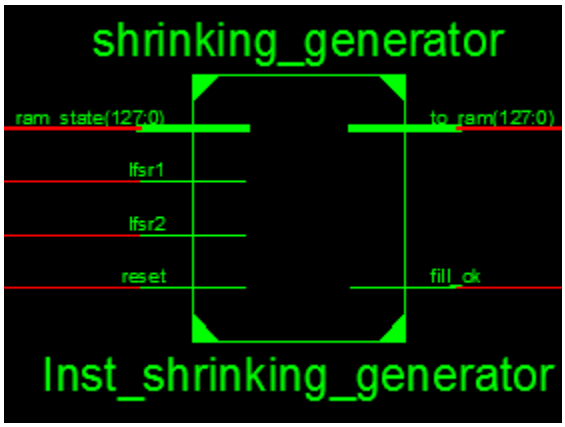
Salidas:

1. Dato de Salida (to_ram): dato de salida de 128 bits para almacenar en RAM
2. Llenado Exitoso (fill_ok): indica que el llenado de la RAM se ha realizado con Éxito

Descripción: el núcleo del generador de números aleatorios, guarda un bit enviado desde el LFSR128 o no, en un registro de 128 bits de acuerdo al bit enviado por el

LFSR127, si es un 1, lo guarda, sino lo descarta, cuando se llenan los 128 bits, envía el dato a RAM, y envía confirmación de que el llenado se ha realizado con éxito

Diagrama De Bloque:



-PRNG (Módulo Completo)

Entradas:

1. Clock (clk): Reloj de la FPGA (50MHz)
2. Reset (reset): Detiene el funcionamiento del dispositivo
3. Entrada Serial (serial_in): entrada serial que será la semilla del generador de números aleatorios

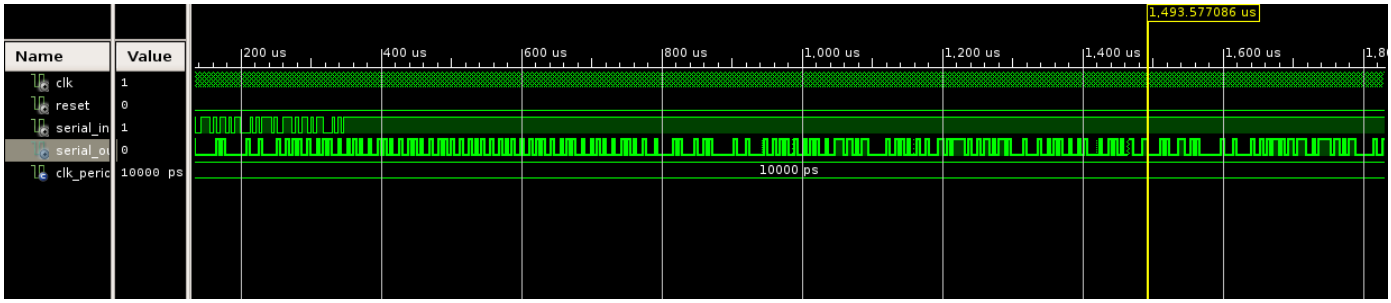
Salidas:

1. Salida Serial (serial_out): Salida serial, son los números aleatorios generados

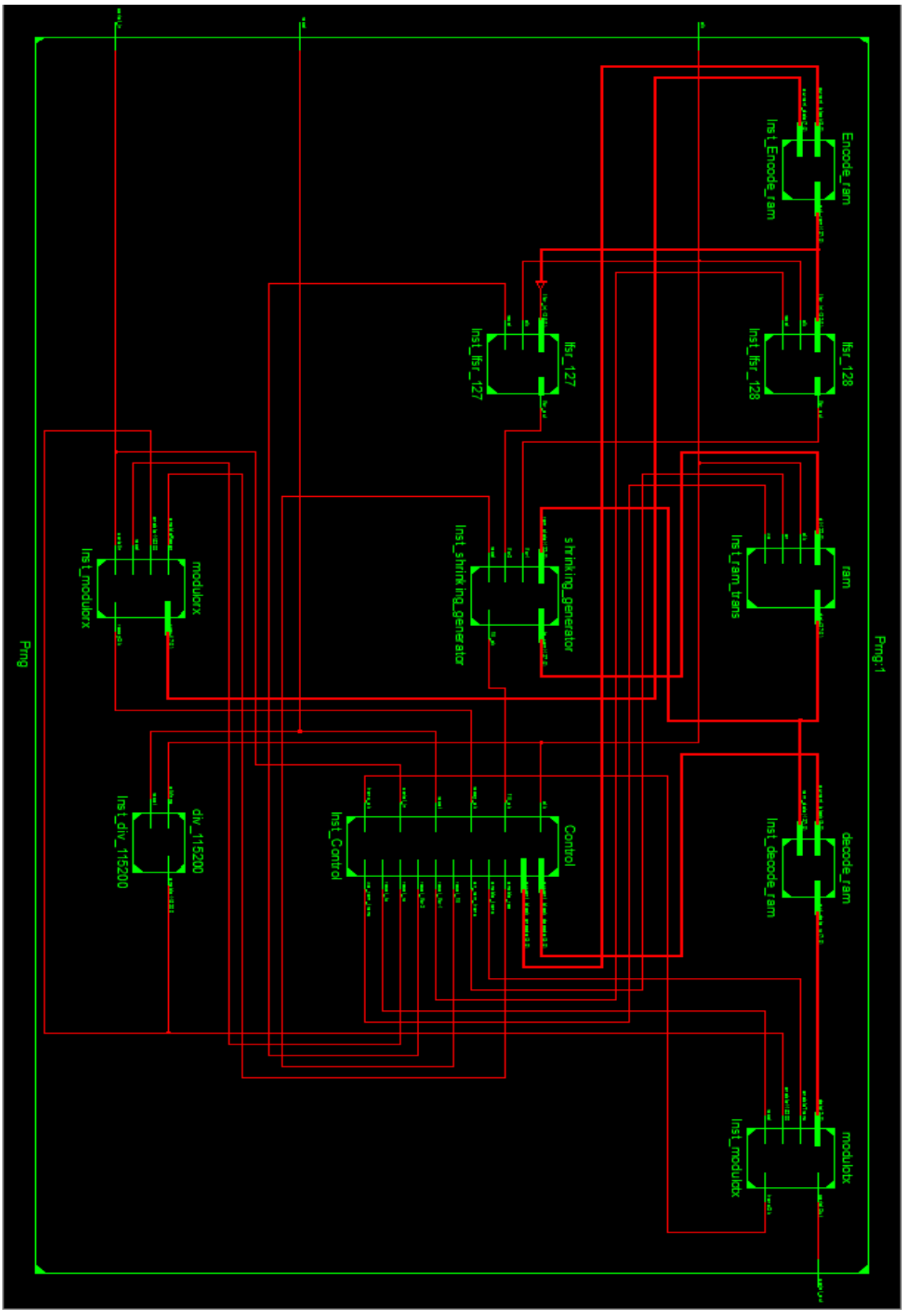
Descripción: Unión de todos los módulos anteriores, recibe una semilla a través de una comunicación serial, y envía números aleatorios de la misma manera, la velocidad de transmisión serial es 115200.

Diagrama de Bloques: ver la siguiente página

Test Bench:



Esquemático General (Diagrama de Bloques):



4. Notas Adicionales

-El proyecto completo (códigos VHDL, documentación, etc) puede ser descargado como parte de un repositorio GIT, o visto online, en la siguiente dirección https://github.com/pin3da/shrinking_generator

-Para probar el diseño se realizó comunicación serial con una PC corriendo Linux Debian, usando una interfaz realizada en Python, en dicha comunicación el PC plantaba la semilla (desde entrada estándar, es decir escrita por el usuario) para el Shrinking Generator, y a su vez recibía los números generados y los mostraba en pantalla como caracteres, una de las versiones mostraba un número cuando el usuario presionara la tecla Enter, la otra versión simplemente mostraba números en pantalla uno tras otro, estas interfaces se pueden encontrar en la carpeta “Python” del Repositorio destinado a este proyecto; Tener en cuenta que a pesar de que nuestro proyecto trabaja con esta interfaz, debería también funcionar para cualquier otra, realizada en cualquier otro lenguaje de programación.

-Este Documento se encuentra a su vez dentro de la carpeta “Documentation” en el repositorio destinado a este proyecto

5. Bibliografía

- Efficient Shift Registers, LFSR Counters, and Long PseudoRandom Sequence Generators, Xilinx Application Note, Peter Alfke, July 7 1996 (Version 1.1)

- Computer Security and Cryptography, Alan G Konheim.

- Searching for the Optimum Correlation Attack in Fast Software Encryption, 2nd International Workshop, Anderson.

- Instant Ciphertext- only Cryptanalysis is Encrypted Comunication, Proceedings of crypto, Barkan, E, Biham, And N. Keller.

- Cipher system, the protection of communications, H. Beker And F. Piper.

- Weaknesses in the key scheduling Algorithm RC4, A. Biryukov, A. Shamir And D.Shamir.

- XST, chapter 3, HDL coding techniques.

- http://en.wikipedia.org/wiki/LFSR

- http://en.wikipedia.org/wiki/State_machine