# Bloques Funcionales del Reverse Traffic Channel basados en un Modelo Generalizado LFSR

# Reverse Traffic Channel Functional Blocks based on LFSR Generalizing Model

Carlos Pérez, MSc.[1], Julio García, MSc.[2]
[1]Grupo de Microelectrónica y Sistemas Digitales
Universidad Surcolombiana
[2]Grupo de Control y Procesamiento Digital de Señales
Universidad Nacional de Colombia Sede Manizales
carpecc@yahoo.com, jcgarciaa@bt.unal.edu.co

Recibido para revisión 26 de Marzo de 2007, aceptado 15 de Junio de 2007, versión final 31 de julio de 2007

***Resumen*— Se presenta una metodología de diseño, basada en *Radios Definidos por Software* (SDR) , y orientada al modelado de los bloques funcionales *CDMA Reverse Traffic Channel.* Además de una metodología de validación dirigida a la comprobación de resultados obtenidos luego de la implementación funcional de los bloques en lenguaje VHDL. Se propone un procedimiento de generación automática del código VHDL correspondiente a los bloques analizados, mediante una estructura generalizada modificada, basada en *Registros de Corrimiento Realimentados Linealmente* (LFSR), que permita generar funciones adicionales de salida, desde la combinación lineal de las posiciones del registro. Se propone un modelo matricial para la configuración de los lazos de realimentación y las funciones de salida. El bloque se caracteriza como un *Modelo Oculto de Markov* (HMM). Para la metodología de validación se propone la generación de archivos gráficos (*.vec) a partir de los resultados obtenidos con Simulink, los cuales pueden ser utilizados en entornos de desarrollo digital como MAX PLUS II o QUARTUS II. Los archivos generados permiten la simulación directa y la correspondiente verificación de los resultados. Para la generación de las descripciones VHDL se realiza una aproximación basada en entidades (entity-based), como un enfoque hacia la descomposición jerárquica basada en unidades funcionales donde una entidad VHDL se considera como una abstracción de un objeto hardware, en este caso un Bloque Funcional.**

*Palabras Clave*—**Registros de Corrimiento Realimentados Linealmente LFSR, CRC, FEC, Codificación Convolucional, CDMA Reverse Traffic Channel, IS-95.**

***Abstract*—The document presents a design methodology, based on Software Defined Radios (SDR), and focused to model CDMA Reverse Traffic Channel functional blocks. Also, a validation methodology is presented, in order to check results obtained after functional implementation of blocks in VHDL.**
**A procedure for automatic VHDL code generation of analyzed blocks is proposed, by means of a generalized structure, based on modified Linear Feedback Shift Registers (mLFSR), which allow the generation of additional functional outputs from linear combination of register positions. Also, a matrix model is proposed, in order to setup feedback loops and functional outputs. Blocks are modeled as a Hidden Marvok Model (HMM). For validation methodology, graphic file generation using *.vec files is proposed, using results from Simulink, which may be used in development environments like MAX PLUS II or QUARTUS II. Generated files allow direct simulation and corresponding results checking. For the generation of VHDL code, an approach entity-based approach is made, as a focus to hierarchical decomposition based on functional units where a VHDL entity is taken as a hardware object abstraction, in this case a Functional Block.**

***Keywords*—Linear Feedback Shift Registers LFSR, CRC, FEC, Convolutional Coding, CDMA Reverse Traffic Channel IS-95.**

## I. INTRODUCCIÓN

LOS *Radios Definidos por Software* (SDR) están enfocados a la construcción de dispositivos de comunicación reconfigurables que permitan cambios funcionales en campo [3] y que faciliten la adaptación a los estándares que surgen ante las cambiantes demandas del mercado. Estos desarrollos van de la mano con nuevas herramientas como el lenguaje de programación VHDL y dispositivos lógicos programables como los FPGA (*Arreglos Lógicos Programables en Campo*). En los procesos de diseño y validación de este tipo de sistemas

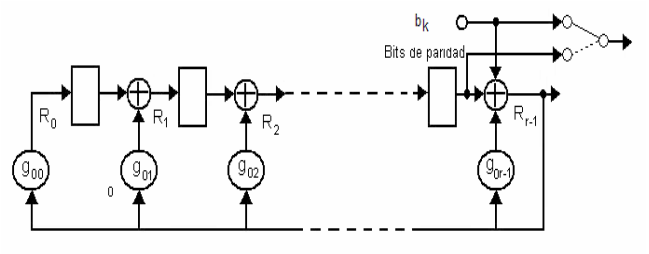



**Figura 1.** Codificador cíclico

se debe manejar una cantidad significativa de datos binarios que debe corresponder a la descripción de cada uno de los elementos funcionales del sistema a desarrollar. Es por esto que se hace necesario generar procesos formales de diseño y validación. En la sección dos se presenta la metodología de diseño propuesta. Se expone el modelo generalizado propuesto, detallando las características particulares de los lazos de realimentación y las trayectorias de salida, las cuales se configuran a través de dos matrices básicas, cuyos modelos también son presentados. La tercera sección presenta una aproximación matemática del modelo propuesto para el diseño de los bloques funcionales, considerando su correspondencia con un *Modelo Oculto de Markov*. La sección cuatro presenta tres casos de estudio en los cuales se aplican y evalúan los procedimientos y modelos propuestos utilizando tres bloques funcionales del *CDMA Reverse Traffic Channel IS-95*. Inicialmente se presenta el proceso completo de diseño y evaluación del bloque Codificador Convolucional, donde se verifica la correspondencia entre los datos generados por el *CDMA Reference Blockset de Simulink* y los que entrega el diseño realizado en VHDL. De igual manera se muestran los resultados encontrados en el diseño del bloque Repetidor, el cual contiene funciones adicionales de salida que pueden ser validadas con los datos obtenidos en *Simulink*. Adicionalmente se presenta la implementación un generador CRC, el cual se compara con los resultados obtenidos en algunas referencias bibliográficas. Finalmente en la sección cinco se presentan las conclusiones del trabajo y algunas recomendaciones.

## II. Modelo Propuesto

Los bloques funcionales *CDMA Reverse Traffic Channel* realizan tratamiento de datos digitales, los cuales son procesados en bloques o de forma cíclica en la medida en que van llegando, haciendo uso de estructuras tipo buffer que pueden ser consideradas como registros. Se presenta un modelo generalizado basado en registros de corrimiento que permite la configuración dinámica a partir de conjuntos de datos binarios organizados como vectores o polinomios, que caracteriza los procesos internos del registro y el flujo de datos de salida, y que puede ser aplicado al diseño de otros bloques.
Es posible estructurar un modelo generalizado que herede características particulares de cada uno de los bloques analizados, de tal forma que se puedan configurar mediante la manipulación de dos elementos característicos, como son, las Trayectorias de Realimentación y las Funciones de Salida.

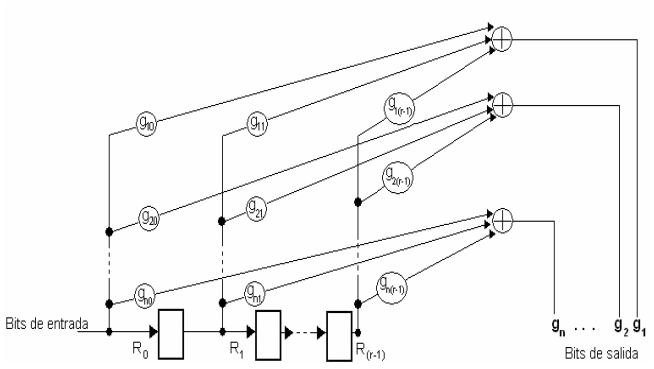



**Figura 2.** Codificador Convolucional

Estas trayectorias y funciones son implementadas a nivel físico mediante operadores booleano representados por sumas XOR y productos mediante compuertas AND.

*A. Trayectorias de Realimentación*

Las trayectorias de realimentación son utilizadas en las operaciones de codificación de redundancia cíclica principalmente, y están determinadas por polinomios generadores relacionados con los códigos CRC utilizados. Las cajas de la figura 1 representan flip-flops o elementos de retardo unitario [2] que mantienen su valor de salida 0 o 1 durante un ciclo de reloj. El contenido de los flip-flops es inicialmente cero y se desplaza en sentido de las flechas con cada pulso de reloj. Los multiplicadores presentes en las trayectorias de realimentación están directamente asociados con los coeficientes $g_{0i}$ del polinomio generador que gobierna la realimentación.
La función generadora asociada a las trayectorias de realimentación se representa por medio del polinomio de grado r-1 en (1), donde r es la cantidad de posibles trayectorias de realimentación.

$$G_0(X) = g_{00}+g_{01}X+g_{02}X^2+\ldots+g_{0i}X^i+\ldots+g_{0(r-1)}X^{r-1} \quad (1)$$

Las ramas que realimentan el valor de la última posición en el ciclo anterior ($R_{r-1}(k-1)$) afectan de forma directa los valores en las diferentes posiciones del registro de corrimiento y las operaciones de desplazamiento para el cálculo de los valores en el presente ciclo ($k$), que a su vez dependen de los valores de las posiciones anteriores en el ciclo anterior ($k-1$) y del valor de la entrada actual, tal como se describe en (2) y (3):

$$R_0(k) = b_k \text{ xor } v\,[g_{00}R_{r-1}(k-1)] \quad (2)$$

$$R_i(k) = R_{i-1}(k-1) \text{ xor } \{g_{0i}[b_k \text{ xor } R_{r-1}(k-1)]\} \quad ; i = 1,2,\ldots,r-1 \quad (3)$$

Donde:

- $b_k$ es la actual entrada
- $R_0(k)$ es el valor actual de la primera posición del registro de corrimiento.
- $R_i(k)$ es el valor actual de la posición i−ésima del registro de corrimiento.
- $R_{i-1}(k-1)$ es el valor de la posición i−1 del registro de

corrimiento en el ciclo anterior.

• $R_{r-1}\ (k-1)$ es el valor de la última posición del registro de corrimiento en el ciclo anterior.

*B. Funciones de salida*

Las funciones de salida están constituidas por sumadores tipo XOR que realizan convoluciones módulo 2 en cada ciclo de reloj utilizando los datos presentes en el registro de corrimiento, tal como se observa en la Figura 2.

Las *n* funciones se salida generan *n* dígitos binarios por cada desplazamiento ocurrido en el registro. Las salidas están asociadas a funciones generadoras representadas por los polinomios de grado *r-1* del sistema de ecuaciones (4).

$$\begin{aligned} G_1(X) &= g_{10} + g_{11}X + \ldots + g_{1(r-1)}X_{r-1} \\ G_2(X) &= g_{20} + g_{21}X + \ldots + g_{2(r-1)}X_{r-1} \\ &\vdots \\ G_n(X) &= g_{n0} + g_{n1}X + \ldots + g_{n(r-1)}X_{r-1} \end{aligned} \quad (4)$$

El conjunto de ecuaciones (5) representa las *n* funciones de salida del sistema.xxxxxxx

$$\begin{aligned} Y_1(k) &= g_{10}R_0(k) \quad g_{11}R_1(k) \quad \ldots \quad g_{1(r-1)}R_{r-1}(k) \\ Y_2(k) &= g_{20}R_0(k) \quad g_{21}R_1(k) \quad \ldots \quad g_{2(r-1)}R_{r-1}(k) \\ &\vdots \\ Y_n(k) &= g_{n0}R_0(k) \quad g_{n1}R_1(k) \quad \ldots \quad g_{n(r-1)}R_{r-1}(k) \end{aligned} \quad (5)$$

A partir de este conjunto de ecuaciones de salida se obtiene la función generalizada (6), descrita como una sumatoria tipo XOR.

$$Y_j(k) = \bigoplus_{i=0}^{r-1} g_{ji}R_i(k) \qquad ,j = 1,2,\ldots,n \quad (6)$$

*2.3 Matriz Generadora*

A partir de la descripción generalizada de los polinomios característicos de las funciones de realimentación y de salida se puede conformar la matriz (7) de dimensiones *(n + 1) _ (r)* que ofrece una representación global de la configuración del sistema correspondiente al sistema de ecuaciones dado por (2), (3) y (5).

$$G = \begin{pmatrix} g_{00} & g_{01} & \cdot & \cdot & \cdot & g_{0(r-1)} \\ g_{10} & g_{11} & \cdot & \cdot & \cdot & g_{1(r-1)} \\ g_{20} & g_{21} & \cdot & \cdot & \cdot & g_{2(r-1)} \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ g_{n0} & g_{n1} & \cdot & \cdot & \cdot & g_{n(r-1)} \end{pmatrix} \quad (7)$$

Con estas representaciones se obtiene un modelo generalizado que conjuga las funcionalidades de realimentación y salida tal como lo muestra la Figura 3. Dentro de la metodología de diseño se plantea una herramienta de generación automática de código VHDL, a partir de las descripciones dadas para las funciones de realimentación y salida, las cuales están contenidas en la matriz general, que permite, mediante un

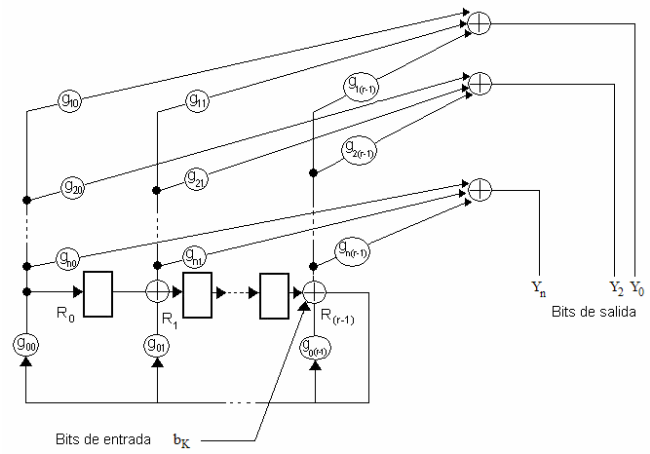



**Figura 3.** Modelo generalizado

programa en *Matlab*, generar un mapeo de la forma descrita en [7] con el fin de generar los objetos y procedimientos dirigidos a la implementación en lenguaje descriptor de hardware. Por ejemplo, para un Codificador Convolucional se trabajan solamente funciones de salida, sin generar lazo de realimentación alguno, mientras que para un codificador CRC se configuran los lazos de realimentación gobernados por los datos de la primera fila de la matriz de configuración y una salida correspondiente a la última posición de memoria del registro.

## III. Aproximación al Modelo

El funcionamiento de cada uno de los bloques que conforman el *CDMA Reverse Traffic Channel* compromete una serie de conceptos matemáticos que se conservan en el modelo propuesto. En los casos particulares del generador CRC, el codificador convolucional y el repetidor, que son los bloques que se analizan en los casos de estudio, se presentan descripciones matemáticas relacionadas con las características de la información generada, la cual corresponde a procesos estocásticos que están relacionados con aleatoriedad de los datos de entrada, los cuales se pueden modelar como un *Proceso Oculto de Markov*.

*A. Modelos de Markov*

Estos modelos representan procesos estocásticos discretos observables, ya que la salida es el conjunto de estados en cada tiempo. Su característica básica radica en la mínima cantidad de memoria asociada debido a que el estado presente depende solo del estado anterior. Bajo estas condiciones se plantea la suposición markoviana en (8).

$$P(X_i \mid X_1^{i-1}) = P(X_i \mid X_{i-1}) \quad (8)$$

donde $X_1^{i-1} = X_1, X_2, \ldots, X_{i-1}$. Bajo esta restricción, y considerando el Teorema de Bayes, se tiene que (9) es la probabilidad de que ocurra una secuencia de *n* variables

$X = X1, X2, \ldots, Xn.$

$$P(X_1,X_2,...,X_n) = P(X_1)\prod_{i=1}^{n} P(X_i|X_1^{i-1}) =$$

$$P(X_1)\prod_{i=1}^{n} P(X_i|X_{i-1}) \quad (9)$$

*B. Modelo Oculto de Markov*

Un Modelo Oculto de Markov está definido como un proceso estocástico doblemente embebido donde existe un proceso estocástico implícito que no es observable. El proceso observable solo puede ser asociado probabilísticamente con el proceso estocástico observable produciendo la secuencia de características que se pueden observar [1]. El modelo oculto de Markov se puede definir mediante las siguientes características:

- **O** = $\{o_1, o_2, ..., o_M\}$. Alfabeto de salida o conjunto discreto de M símbolos.
- Ω = $\{1, 2, ...,N\}$. Conjunto de estados no observables.
- **A** = $\{a_{ij}\}_{NxN}$. Matriz de probabilidad de transición, donde $a_{ij}$ es la probabilidad de pasar del estado $i$ al estado $j$, y se describe como $a_{ij}=P(s_t= j|s_{t-1} = i)$.
- **B** = $\{b_i(k)\}$. Matriz de probabilidad de salida, donde $b_i(k)$ es la probabilidad de obtener el símbolo de salida $o_k$ en el estado $i$. Con esto se tiene $b_i(k) = P(X_t =o_k|s_{t = i})$.
- $\boldsymbol{\pi}$ = $\{\pi_i\}$. Distribución inicial de estados, donde $\pi_i =P(s_0 = i)$, *para* $1 <= i <= N$

Una especificación completa del Modelo Oculto de Markov está dado por el conjunto $\{N,M, \Phi\}$, donde $\Phi = (A,B, \pi)$. Se manejan dos suposiciones para un modelo de primer orden:

*1. La suposición de independencia a la salida*

$$P(X_t|X_1^t, s_1^t) = P(X_t|s_t) \quad (10)$$

Indica que la probabilidad de que un símbolo sea emitido en el tiempo t depende solo del estado $s_t$.

*2. La suposición de Markov*

$$P(s_t|s_1^{t-1}) = P(s_t|s_{t-1}) \quad (11)$$

Indica que el estado actual solo depende del anterior.

El proceso que relaciona las salidas con los estados del registro esta dado por el sistema de ecuaciones (5) y se puede asociar a la ecuación (10), mientras que el proceso asociado a los cambios de estados está condicionado estocásticamente con los cambios en la entrada del sistema y se encuentra descrito en (2) y (3), asociados con (11).

*C. Aproximación al Modelo Oculto de Markov*

El codificador convolucional genera una codificación continua de datos convirtiéndose en una máquina de 2r estados que entrega n bits de salida antes de pasar al nuevo estado. La salida generada $Y^k$ y el nuevo estado $S^{k+1}$ dependen del estado presente $S^k$ y de la entrada $b_k$ [6] de acuerdo a

$$Y^k = f(S^k, b_k) \quad (12)$$

y

$$S^{k+1} = g(S^k, b_k) \quad (13)$$

Estas características se conservan para el generador CRC, el cual maneja lazos de realimentación, y para los demás bloques del sistema CDMA que podrían ser implementados mediante el modelo propuesto. Considerando estas descripciones, es posible asociar (13) con (11) y (12) con (10), teniendo en cuenta que los estados en (11) y (10) son estocásticamente dependientes de los datos de entrada. A partir del análisis de las características del sistema, podemos asociarlo directamente con un proceso subyacente no observable e interno de cada bloque (los estados S del registro) y con un proceso observable (las salidas Y ) que es dependiente del anterior, lo cual nos indica que cada bloque corresponde estructural y funcionalmente a un *Proceso Oculto de Markov*.

## IV. Casos de Estudio y Resultados

Se trabaja con *Matlab* y *Simulink* debido a las ventajas de interacción que existen entre los dos y a la disponibilidad de estas herramientas, lo cual evita la inversión de recursos adicionales relacionados con la compra de licencias. En el caso de *Simulink*, se encuentran dos ventajas adicionales importantes que son la posibilidad de un entorno gráfico de operación y la capacidad de entregar herramientas de simulación relacionadas con el estándar IS-95 a través del *CDMA Reference Blockse*t, las cuales permiten generar datos tendientes a la futura validación de los procesos implementados en los entornos de programación de *Matlab* y en *VHDL*. En cuanto a *Matlab*, la capacidad de utilizarlo como un entorno de programación, favorece los trabajos de diseño previos ya que permite la implementación y depuración de los algoritmos que describen los bloques funcionales del *CDMA Reverse Traffic Channel* y la posterior generación de las estructuras *VHDL* correspondientes. En lo relacionado con la revisión y la refinación del código, el entorno de programación de *Matlab*, favorece una eficiente depuración principalmente por la ya mencionada capacidad de conexión con *Simulink* a través de herramientas como los módulos To Workspace que permiten la obtención de datos orientados a la validación de los algoritmos. Igualmente se presenta la capacidad de conexión con el entorno *Max Plus II* de *Altera Corp.* a través de una herramienta básica de generación de código *VHDL* desarrollada en este trabajo, la cual utiliza las estructuras generalizadas propuestas para generar las estructuras *VHDL* correspondientes a los bloques específicos. Se presenta el diseño de un modelo generalizado para sistemas basados en Registros de Corrimiento utilizados en *CDMA IS-95*, el cual permite lograr modelos estructurales y funcionales específicos a partir del modelo general. Estos registros están principalmente presentes en algunos bloques funcionales del *CDMA Reverse Traffic Channel* tales como el Codificador Convolucional y el Generador CRC del Indicador de Calidad de Trama (FQI).

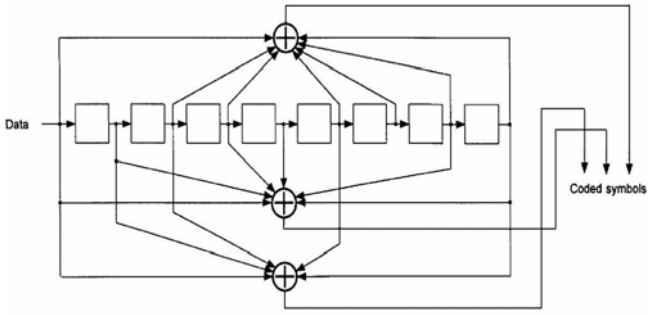



**Figura 4.** Codificador Convolucional para el canal Reverse: r=1/3, K=9

*A. Diseño y validación del Codificador Convolucional*

Se describe el proceso de validación del bloque Forward Error Correction (FEC) perteneciente al *CDMA Reverse Traffic Channel* para una velocidad *Full Data Rate*. Este bloque funcional se encarga de generar un proceso de corrección de errores en adelanto usando una codificación con tasa 1/3.

*A.1 Configuración del FEC.* El FEC implementado es un registro de corrimiento del longitud K = 9 y tres sumadores tipo XOR (figura 4). Los lazos de realimentación que se pueden configurar con el vector *G0 = [000000000]* y tres salidas *($Y_1(t)$, $Y_2(t)$ y Y3(t))* producto de una combinación de las diferentes posiciones del registro de corrimiento y configurables según las siguientes presentaciones del vector de configuración $O_j$

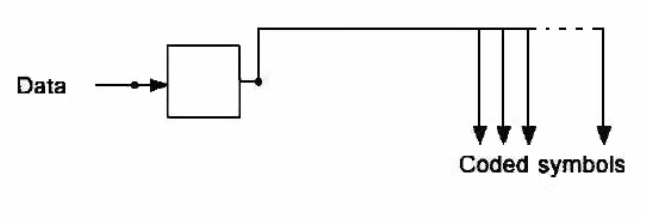



**Figura 5.** Representación del Symbol Repeater mediante registros de corrimiento

*O1 = G1 = [101101111]*
*O2 = G2 = [110110011]*
*O3 = G3 = [111001001]*

El flujo de datos de salida está conformado por conjuntos de tres bits que corresponden a la codificación de cada bit de entrada.

*A.2 Validación.* Se utilizó un tiempo de simulación de 20ms, tiempo en el que se transmite una trama. En el caso del bloque *IS-95A Rev Ch Convolutional Encoder*, este recibe un arreglo de datos binarios por parte del bloque *IS-95A Rev Ch CRC Generator* con una longitud de 288 bits, de los cuales el bloque FEC toma los 192 primeros que corresponden a la trama de 20 ms, y los transforma en 576 a través de un proceso de codificación convolucional con relación 1/3.

Tanto el bloque *IS-95A Rev Ch CRC Generator* como el bloque *IS-95A Rev Ch Convolutional Encoder* están conectados al *Command Window* de *Matlab* a través de dos bloques *To Workspace* llamados crct y fec, respectivamente. La figura 6 ilustra el bloque.

*A.3 Generación del archivo *.vec.* Se desarrolló un programa en *Matlab* llamado fec96vec.m el cual toma el arreglo crct y lo

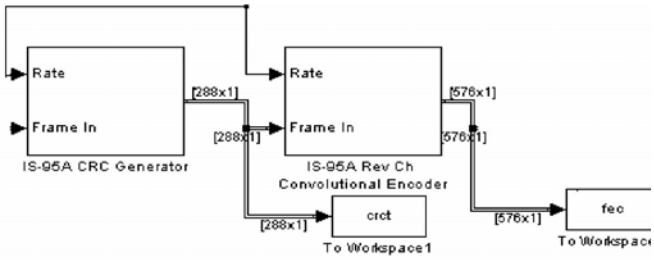



**Figura 6.** Bloques To Workspace utilizados

transforma según el formato utilizado por los archivos tipo vector de *Max+Plus II*, asignado un tiempo de inicio y un tiempo de finalización de la simulación y asignando los puertos de entrada y los intervalos en que se transmiten los datos. En este caso el puerto más significativo es el *datain* que recibe los 192 datos procedentes del bloque CRC. Luego de correr el programa fec96vec.m se obtiene un archivo de texto que contiene la información necesaria para que la aplicación gráfica de simulación de *MaxPlus II* pueda obtener los datos de entrada.

*B. Configuración y validación del Symbol Repeater*

Para llevar las diferentes velocidades de entrada a un valor estándar, el bloque *Symbol Repeater* genera una repetición de cada uno de los bits provenientes del *FQI Encoder* (CRC

Generator) de acuerdo a la tabla 1. La figura 7 presenta un esquema generalizado del bloque repetidor, el cual, en cualquier caso solo debe contener un Registro unitario a partir del cual se generan las n salidas correspondientes al factor n de repetición. El registro de corrimiento será de longitud K = 2 sin realimentación con *F = [00]*. Aplicando al caso *Half Data Rate* que duplica los bits de entrada, se tendrá un sistema con dos salidas configuradas por los vectores *O1 = G1 = [01]*
*O2 = G2 = [01]*

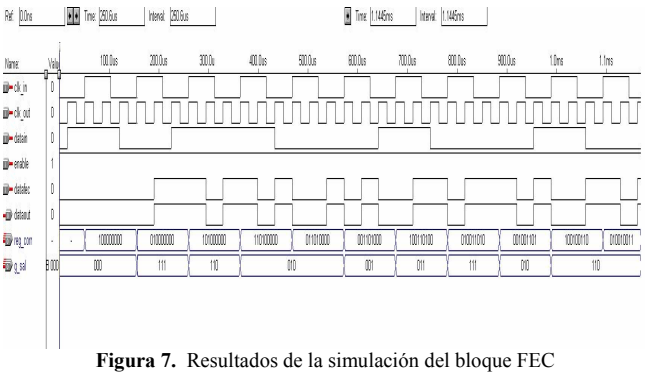


**Figura 7.** Resultados de la simulación del bloque FEC

*B.1 Validación.* Luego de simular el repetidor para una velocidad *Half Data Rate* se obtienen los resultados de la

figura 8, en la cual se aprecia una entrada llamada datarep que contiene la información resultante en la simulación en *Simulink*. Esta información fue adicionada a los datos que se aplican a los puertos de entrada en el archivo .vec con el fin de permitir una inspección visual de la validez de las simulaciones. Las señales de entrada y salida son prácticamente las mismas, la diferencia básica se marca por los relojes de entrada y salida que tienen una relación de velocidad *1/n*, en este caso *1/2*.

### *C. Diseño y validación del generador CRC*

El CRC utiliza Registros de Corrimiento Realimentados Linealmente (LFSR) configurados de acuerdo a un Polinomio Generador. Dependiendo de la velocidad del canal se utiliza una función generadora representada por un polinomio de grado n. La función de salida se genera por la suma XOR entre el dato de entrada y la última posición (Rr−1(k)) del registro

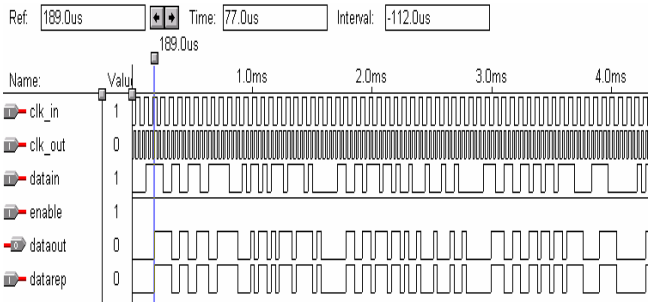



**Figura 8.** Resultados de la simulación del bloque Repeater

de corrimiento. Para la validación del CRC de utilizo un ejemplo presentado en el capítulo 5 de la referencia [4] con polinomios de orden 3.

*C.1. Configuración Generador CRC3.* Se validó el bloque haciendo uso del generador CRC-3 de la figura 9 que utiliza un registro de corrimiento de longitud K = 3 con lazos de realimentación dados por F = [1100] y una salida que se configura a partir del vector O1 = [1000] para i < 4, y O1 = [0001] para 4 i < (4 + 3).

### *C.2. Validación*

El codificador recibe la cadena de datos de entrada I = [1110] y adiciona en la salida un vector crc = [100] que corresponde a un procesamiento a partir de la ecuación x3+ x + 1. Los resultados de la simulación aparecen en la figura 10.

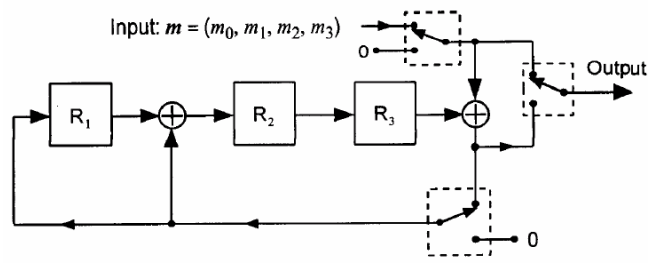



**Figura 9.** Codificador CRC de orden 3

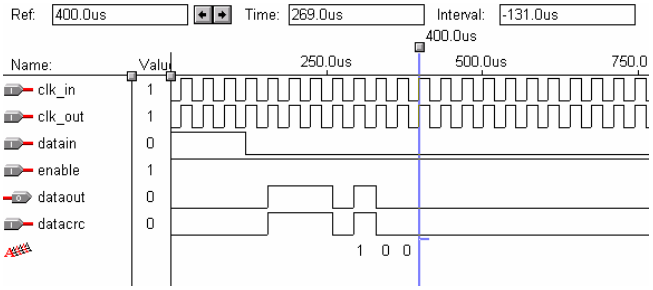



**Figura 10.** Simulación para el codificador CRC 3

## V. CONCLUSIONES

Por medio de la clase *Funct_Block* se integran las características funcionales generales basadas la aplicación de registros de corrimiento con propiedades de realimentación y generación de funciones de salida. Esta clase se instancia en varias clases y objetos relacionados con los bloques funcionales de *CDMA Reverse Traffc Channel*.

Al generar especificaciones para la estructuración de una matriz de configuración (similar funcionalmente a la que se propone en [5]) se pueden definir características operacionales relacionadas con las funciones de salida y los procesos de realimentación de los registros de corrimiento. La herramienta de traducción automática, similar a la que se propone en [7], permite mapear las definiciones de clases correspondientes a los bloques funcionales como parejas *entidad-arquitectura* en un código *VHDL*, ofreciendo la posibilidad de utilizar parámetros configurables, lo cual permite adicionalmente evaluar bloques funcionales con especificaciones diferentes a las del estándar *IS-95*.

Mediante la aplicación de la metodología propuesta se puede realizar una validación de los bloques funcionales, tal como se presentó en los casos de estudio de la sección 4.

Generando una interfaz entre el *CDMA Reference Blockset de Simulink* y la aplicación *Simulator* del entorno de desarrollo *Max+Plus II* se pueden comparar datos por medio de un archivo tipo vector *(*.vec),* el cual es utilizado para generar la descripción gráfica del comportamiento de los diferentes puertos de una entidad *VHDL*. Los datos utilizados para la generación del archivo *.vec* son extraídos de *Simulink* y procesados en consola por medio de un programa de *Matlab*.

Se encontraron algunos inconvenientes en la generación de los tiempos de reloj para entrada y salida del bloque debido a que la cantidad oscilaciones en ambos casos no es un número entero para un periodo de tiempo de 20 ms, por lo tanto se presenta un pequeño desfase.

## REFERENCIAS

[1] J. Baker. Stochastic modeling for automatic speech understanding. Readings in Speech Recognition, pages 297–307, 1990.
[2] S. Haykin. Communication Systems. John Wiley and Sons, 2001.
[3] H. R. Julián Busques. Desarrollo y simulación de una estación base gsm/cdma utilizando software radio. Revista Facultad de Ingenier´ya U.T.A., (10):3–10, 2002.
[4] J. S. Lee and L. E. Milles. CDMA Systems Engineering Handbook. Artech House, 2002.
[5] V. S. R. Damasevicius. Application of the object-oriented principles for

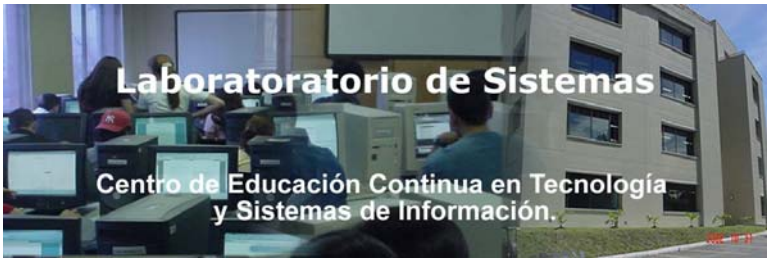
hardware and embedded system design. INTEGRATION,the VLSI journal, (38):309–339, 2004.
[6] R. Soleymani. Turbo Coding for Satellite and Wireless Communications. Kluwer Academic Publishers, 2002.
[7] B. H. William McUmber. Uml-based analysis of embedded systems using a mapping to vhdl. Proceedings of the IEEE International Symposium on High Assurance Software Engineering, pages 56–63, 1999.

**Carlos Alberto Pérez Camacho.** Nacido en Neiva el 14 de Mayo de 1979. Obtuvo el grado de Ingeniero Electrónico en la Universidad Nacional de Colombia en 1998, el grado de Magister en Ingeniería Eléctrica en la Universidad Nacional de Colombia el año 2007. Se desempeña como docente en las áreas de Comunicaciones y Electrónica Digital de la Universidad Surcolombiana desde el año 2006 y realiza trabajos en el Grupo de Microelectrónica y Sistemas Digitales de esta misma universidad.

**Julio César García Alvarez** (Student'96, M'04). Nacido en Armenia, el 17 de Junio de 1977. Obtuvo el grado de Ingeniero Electrónico en la Universidad Nacional de Colombia en 1998, el grado de Magister en Ingeniería Electrónica y de Comunicaciones en la Universidad de los Andes en 2000.

Trabajó en el proyecto de investigación: Seguimiento del Corazón de la ballena Vía Satélite (SCVS) de 1999 a 2000. Actualmente trabaja como profesor de tiempo completo en la Universidad Nacional de Colombia Sede Manizales desde 2001. Sus áreas de investigación son la Integración de Servicios en Telecomunicaciones, Análisis de Tráfico, y Propagación Electromagnética.

# Universidad Nacional de Colombia Sede Medellín
# Facultad de Minas




**Misión**

Ofrecer servicios de apoyo a la docencia en cuanto a la operación de los computadores y del software adecuado con miras al desarrollo integral de los futuros ingenieros.

**Visión**

Avanzamos en la búsqueda de convertir el Laboratorio de Sistemas e Informática en una dependencia ágil, moderna, facilitadora de procesos y cambios, atenta a las necesidades de otras dependencias de la Universidad, cuya labor apoyamos y articulamos. Serviremos con dinamismo, amabilidad y efectividad a todos los integrantes de la comunidad universitaria y a la sociedad en general. Uniremos esfuerzos para construir un ambiente de trabajo cada vez más positivo que propicie la participación, la creatividad y el desarrollo profesional de los integrantes del equipo de trabajo. Propendemos por un Laboratorio como instrumento gestor y generador de proyectos de investigación sustentado en un equipo interdisciplinario de trabajo en torno a la informática aplicada a la ingeniería.

**Cursos**

- Lenguajes de Programación: Diseño de Páginas Web en ASP.NET con VB.NET, Programación Web PHP y MYSQL, MS-Visual Basic Básico y Avanzado, Java
- Generales: Latex, ARCGIS, MS-Office (Word, Excel y PowerPoint), Excel Financiero, Excel Avanzado, Mantenimiento de Hardware y Software Niveles I y II, MS-Access Básico, MS-Project Básico (Programación y Gerencia de Proyectos), AUTOCAD 2D Básico y 3D, Matlab, Moodle de Apoyo a los Procesos de Enseñanza y Aprendizaje.

Para mayor información, por favor comunicarse a los teléfonos: 4255313, 4255312, 4255355. Calle 59A No. 63 – 020 Medellín (Colombia), Bloque M7, quinto piso.
Email: labsis@unalmed.edu.co http://xue.unalmed.edu.co/cursos