DISEÑO DE UN TURBO DECODIFICADOR ANALÓGICO PARA EL CÓDIGO DE CORRECCIÓN DE ERRORES UMTS DE 40 BITS

Jorge Luis Lagos Benites, Carlos Silva Cárdenas.
lagos.jl@pucp.edu.pe, csilva@pucp.edu.pe

Grupo de Microelectrónica, Pontificia Universidad Católica del Perú
Av. Universitaria S/N, Lima 32, Lima – Perú

RESUMEN

El presente trabajo detalla el diseño de un turbo decodificador analógico para el código de corrección de errores de 40 bits definido en el estándar UMTS para comunicaciones celulares de tercera generación. El algoritmo de decodificación elegido para la implementación de los decodificadores constitutivos es el algoritmo Log-MAP, y se muestra cómo las operaciones requeridas por dicho algoritmo pueden ser implementadas de manera directa empleando sencillos bloques analógicos. El funcionamiento del diseño final es simulado a nivel de captura esquemática sobre un proceso CMOS estándar de 0.35µm, obteniéndose resultados promisorios.

1. INTRODUCCIÓN

Debido a su impresionante capacidad de corrección de errores, que posibilita la comunicación a tasas de bit cercanas a la capacidad del canal, los turbo códigos [1] han recibido en los últimos años una enorme atención en los campos de la teoría de la información y las telecomunicaciones, siendo su inclusión en estándares de última generación una consecuencia directa de su excelente desempeño. En particular, esta clase de códigos de corrección de errores ha sido la elegida en la implementación de los dos más importantes estándares de comunicación celular de tercera generación (UMTS y CDMA2000). No obstante, el alto grado de complejidad de los algoritmos empleados en la decodificación de los turbo códigos ha impuesto desde un inicio un límite sobre las prestaciones mostradas por las implementaciones existentes de turbo decodificadores. Si bien es cierto existen algoritmos sub-óptimos capaces de ofrecer distintos grados de reducción en la complejidad de implementación, éstos lo logran a costa de una consecuente disminución en la capacidad de corrección de errores, ofreciendo un desempeño por debajo del óptimo. A pesar de estas desventajas, dichas simplificaciones han sido largamente empleadas en las implementaciones existentes de turbo decodificadores digitales, logrando reducciones significativas en cuanto a latencia de operación y consumos de área y potencia.

En vista de lo anterior, los esfuerzos de investigación de los últimos años se han dirigido a la búsqueda de alternativas que permitan lograr implementaciones más eficientes en términos de área, latencia y consumo de potencia, destacando la alternativa de decodificación analógica, inicialmente propuesta por Loeliger [2] y Hagenauer [3]. Las principales ventajas que ofrece esta vía incluyen la posibilidad de implementar directamente las operaciones del algoritmo de decodificación óptimo (sin efectuar aproximaciones), y la eliminación de los pasos discretos inherentes al proceso de decodificación iterativa. Esto trae como consecuencia un mejor desempeño en términos de la tasa de errores de bit para una relación señal-a-ruido dada, y un considerable incremento de la velocidad máxima de operación (véase [4] para una reseña del estado del arte al respecto).

Enmarcado en este contexto, el presente trabajo detalla el diseño del segundo turbo decodificador analógico reportado a la fecha para el código de corrección de errores de un estándar de comunicaciones real (siendo el otro el de la referencia [5]). En la Sección 2 se presenta la teoría detrás de la operación del turbo decodificador considerado, mientras que la arquitectura y diseño del mismo se detallan en la Sección 3. Los resultados preliminares de desempeño se presentan en la Sección 4. Finalmente, las conclusiones y direcciones futuras de trabajo se incluyen en la Sección 5.

2. ASPECTOS TEÓRICOS

2.1. El turbo código UMTS de 40 bits

Los turbo códigos son códigos de corrección de errores basados en la concatenación de dos o más códigos

convolucionales. En este trabajo se ha considerado el turbo código empleado en la especificación UMTS estandarizada por el *Third Generation Partnership Project* (3GPP) para comunicaciones celulares [6].

Tal como se observa en la Figura 2.1-1, el turbo codificador UMTS está compuesto por la concatenación paralela de dos codificadores convolucionales recursivos y sistemáticos (CCRS) idénticos. La estructura de cada uno de estos codificadores se muestra en la Figura 2.1-2.

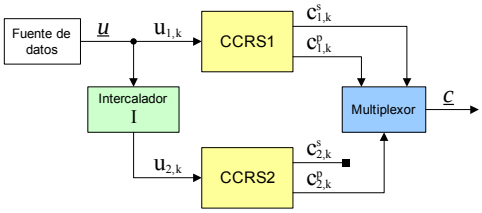

Figura 2.1-1 Turbo codificador UMTS

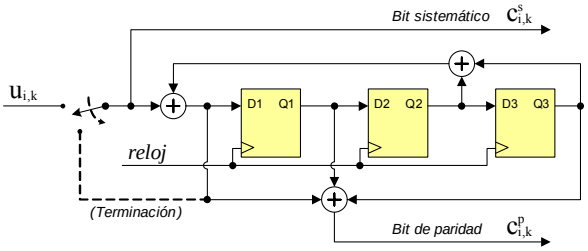

Figura 2.1-2 Estructura interna de cada CCRS [6]

Como puede apreciarse, cada CCRS no es sino una simple máquina de estados que experimenta transiciones con cada ciclo de reloj según los bits de información u_k, produciendo simultáneamente dos bits codificados: un bit sistemático $c^s_{i,k}$ (idéntico al bit de información) y un bit de paridad $c^p_{i,k}$. Cuando se grafican en función del tiempo las posibles transiciones entre los ocho estados de esta máquina se obtiene el "diagrama de *trellis*" del código, el cual es mostrado en la Figura 2.1-3. Nótese que al tratarse de un código convolucional binario, existirán siempre sólo dos "ramificaciones" saliendo y llegando a cada estado. En la Figura se han indicado con distintos colores las transiciones que ocurren para los diferentes valores que puede tomar cada bit de datos u_k (azul cuando es "0" y rojo cuando es "1").

La operación del turbo codificador es como sigue. En un inicio cada CCRS es inicializado en el estado cero y su conmutador conectado en la posición superior, permitiendo el "ingreso" de la secuencia de datos $\underline{u}$ dividida en bloques de 40 bits. Los bits de cada bloque son codificados por el CCRS1 en su orden natural "$u_{1,k}$" y por el CCRS2 en su versión permutada "$u_{2,k}$", la cual es obtenida empleando un bloque intercalador "I" siguiendo un intrincado patrón de permutación, diseñado con el fin de maximizar la independencia estadística de estas secuencias. Las secuencias codificadas producidas por sendos CCRS son concatenadas empleando un multiplexor para producir la trama final de datos codificados $\underline{c}$. Nótese que en este proceso los bits sistemáticos del CCRS2 son desechados, obteniéndose de esta manera tres bits de codificación ($c^s_{1,k}$, $c^p_{1,k}$ y $c^p_{2,k}$) por cada bit de datos u_k.

Una vez que los 40 bits del bloque de datos han sido codificados (produciendo 120 bits codificados), ambos CCRS son forzados a regresar al estado cero, conectando sus respectivos conmutadores en la posición inferior. Ya que se necesitan tres ciclos de reloj para asegurar el retorno de cada CCRS al estado cero, este proceso genera 12 "bits de terminación", pues ahora, para cada ciclo de reloj, los cuatro bits de sendos CCRS (incluyendo los bits sistemáticos del CCRS2) son tomados en cuenta. Estos 12 bits de terminación son anexados a los 120 previamente obtenidos, logrando una longitud total de trama de datos codificados de 132 bits y una tasa de codificación de 40/132 ≈1/3. Esta trama es luego modulada y transmitida a través del canal.

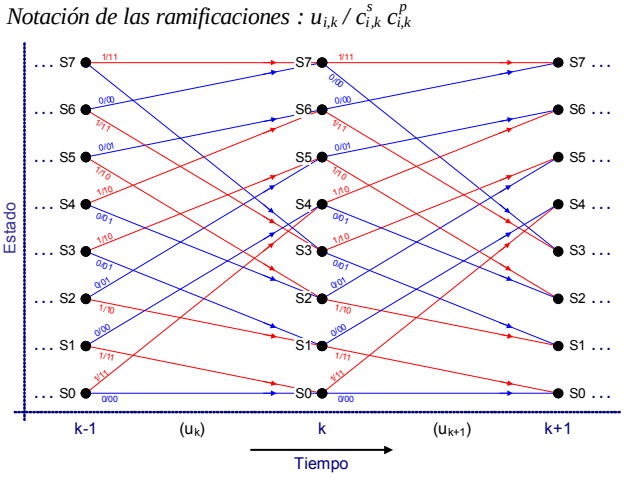

Figura 2.1-3 Diagrama de trellis del turbo código 3GPP

2.2. Decodificación óptima del turbo código UMTS

Tal como lo fuera demostrado en el trabajo seminal [1], la decodificación de un turbo código se reduce a la decodificación iterativa de cada uno de los códigos convolucionales que lo constituyen. El algoritmo MAP (*maximum-a-posteriori*) y su variante logarítmica, el algoritmo Log-MAP, constituyen los métodos óptimos de decodificación para dichos códigos [7]. A continuación se presenta un resumen de las operaciones necesarias para la implementación del algoritmo Log-MAP en la decodificación del turbo código UMTS de 40 bits. Como se mostrará en la Sección 3, se ha elegido este algoritmo por la facilidad y eficiencia de su implementación analógica empleando sencillos bloques funcionales.

La estructura básica en consideración es la mostrada en la Figura 2.2-1. Tal como se aprecia, el turbo decodificador está compuesto por la concatenación de dos decodificadores constitutivos, denominados módulos SISO (*soft-input, soft-output*), los cuales operan en un lazo iterativo de realimentación. La secuencia "$\underline{y}$" capturada a la salida del canal es demodulada y demultiplexada con el objeto de proveer a cada módulo SISO con los símbolos correspondientes al código que debe decodificar: el SISO1 decodifica los bits del CCRS1 y el SISO2 hace lo propio para el CCRS2. En el proceso global de decodificación, estos módulos intercambian entre ellos información con el propósito de mejorar sus decisiones, según un proceso iterativo que es detenido una vez que la mejora en la secuencia decodificada se vuelve marginal.

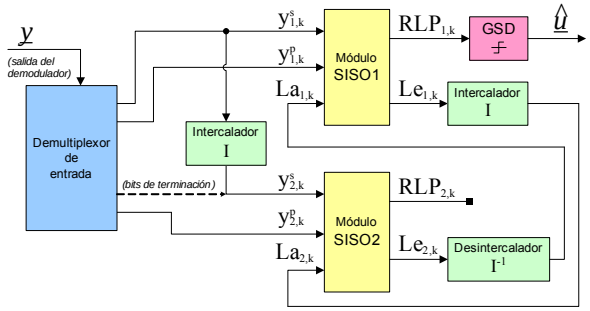

Figura 2.2-1 Estructura del turbo decodificador

El algoritmo Log-MAP permite obtener a la salida de cada SISO la *razón logarítmica de probabilidad (RLP) a posteriori* de cada bit de información (condicionada a la secuencia recibida R_1^N), la cual es definida según:

$$RLP(u_k) \triangleq \ln\left\{\frac{P\left[u_k = 1 \mid R_1^N\right]}{P\left[u_k = 0 \mid R_1^N\right]}\right\} \qquad 2.2\text{-}1$$

La decisión final acerca de cada bit de información u_k es tomada de acuerdo con el signo de su respectiva RLP. Para el cálculo de las RLPs, cada SISO hace uso de los valores recibidos a la salida del canal (secuencia R_1^N) y de la información *a priori "La"* que le provee el otro SISO. Cada módulo produce además *información extrínseca "Le"*, la cual es empleada como información *a priori* por el otro SISO en la siguiente iteración.

El algoritmo Log-MAP permite evaluar la RLP para cada bit de datos a partir del cálculo de ciertas métricas definidas sobre el diagrama de *trellis* del código. En el tratamiento que sigue, *P[ ]* denota probabilidad, *k* el tiempo discreto, y S_k el estado del CCRS en el tiempo *k*. Las variables *m* y *m'* denotan alguno de los *M* estados del CCRS, mientras que R_1^N y R_k^l denotan la secuencia total de símbolos recibidos y la porción de ésta entre los tiempos *k* y *l*, respectivamente. Se asume que el canal de transmisión es del tipo AWGN, y está corrompido por ruido blanco gaussiano y aditivo de varianza σ^2.

Para cada estado *m* en el *trellis* y para cada unidad de tiempo k, se definen las correspondientes ***métricas de estado progresiva "$A_k(m)$"*** y ***regresiva "$B_k(m)$"*** según:

$$A_k(m) \triangleq \ln\left\{P\left[S_k = m, R_1^N\right]\right\} \qquad 2.2\text{-}2a$$

$$B_k(m) \triangleq \ln\left\{P\left[R_{k+1}^N \mid S_k = m\right]\right\} \qquad 2.2\text{-}2b$$

De otro lado, a la ramificación del *trellis* que corresponde a la transición entre los estados *m'* (tiempo *k-1*) y *m* (tiempo *k*) que ocurre para $u_k = i$ se le asocia la ***métrica de ramificación:***

$$C_k^i(m, m') \triangleq \ln\left\{P\left[u_k = i, S_k = m, R_k \mid S_{k-1} = m'\right]\right\} \qquad 2.2\text{-}3$$

El algoritmo Log-MAP permite calcular las métricas de estado $A_k(m)$ y $B_k(m)$ de manera recursiva, a partir de las métricas de ramificación y las condiciones iniciales apropiadas:

$$A_k(m) = \max_{m'=1}^{M}{}^{*}\left\{A_{k-1}(m') + \max_{i=0}^{1}{}^{*}\left\{C_k^i(m', m)\right\}\right\}$$

"recursión progresiva" 2.2-4a

$$A_0(S0) = 0, \quad A_0(m) = -\infty \;\forall m \neq 0 \qquad 2.2\text{-}4b$$

$$B_k(m) = \max_{m'=1}^{M}{}^{*}\left\{B_{k+1}(m') + \max_{i=0}^{1}{}^{*}\left\{C_{k+1}^i(m, m')\right\}\right\}$$

"recursión regresiva" 2.2-5a

$$B_N(S0) = 0, \quad B_N(m) = -\infty \;\forall m \neq 0 \qquad 2.2\text{-}5b$$

En las anteriores expresiones el operador *max** es el *logaritmo de Jacobi*, definido según:

$$\max^*(a_1, a_2) \triangleq \ln\left(e^{a_1} + e^{a_2}\right) \qquad 2.2\text{-}6$$

La Figura 2.2-2 muestra algunos ejemplos del uso de las anteriores expresiones en el cálculo de las métricas de estado progresivas y regresivas del turbo código:

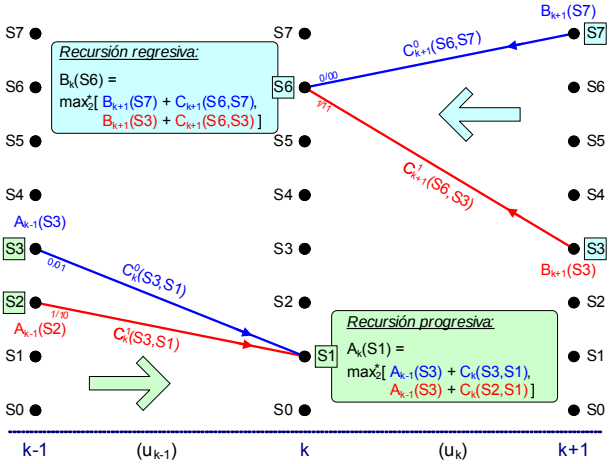

Figura 2.2-2 Ejemplos de recursiones

El cálculo de las métricas de ramificación se hace a partir de los valores de paridad y sistemático obtenidos del canal, y de la información *a priori* proporcionada por el otro módulo SISO. El cálculo de cada métrica de ramificación se realiza en función a tres componentes:

2.2-7a

$$\tilde{C}_k^i(m',m) = \begin{cases} C_{y_k^s}^{c_k^s} + C_{y_k^p}^{c_k^p} + C_{La_k}^i & ,\exists(m',m)/u_k = i \\ -\infty & ,\nexists(m',m)/u_k = i \end{cases}$$

Nótese que las métricas de ramificación sólo toman valores finitos para los ramificaciones existentes en el *trellis*. Las componentes en la ecuación 2.2-7a corresponden a los valores de los símbolos sistemáticos y de paridad obtenidos a la salida del demodulador (afectados por una constante conocida como "fiabilidad del canal"), y a la información *a priori* proporcionada al módulo SISO por el otro decodificador:

$$C_{y_k^s}^{c_k^s} \equiv \begin{cases} -L_c y_k^s & ,c_k^s = '0' \\ +L_c y_k^s & ,c_k^s = '1' \end{cases} \quad ,c_k^s \text{ del ramal } (m',m) \qquad 2.2\text{-}7b$$

$$C_{y_k^p}^{c_k^p} \equiv \begin{cases} -L_c y_k^p & ,c_k^p = '0' \\ +L_c y_k^p & ,c_k^p = '1' \end{cases} \quad ,c_k^p \text{ del ramal } (m',m) \qquad 2.2\text{-}7c$$

$$C_{La_k}^i \equiv La_k(i-1/2) = \begin{cases} -La_k/2, & i = '0' \\ +La_k/2, & i = '1' \end{cases} \qquad 2.2\text{-}7d$$

$$L_c \equiv 1/\sigma^2 \qquad \text{"fiabilidad del canal"} \qquad 2.2\text{-}7e$$

Definidas todas las anteriores cantidades, la RLP *a posteriori* de la ecuación 2.2-1 es obtenida según:

$$RLP(u_k) = \max_{m'=0}^{M-1}{}^*\left\{A_{k-1}(m') + \max_{m=0}^{M-1}{}^*\left\{\tilde{C}_k^1(m',m) + B_k(m)\right\}\right\} - \max_{m'=0}^{M-1}{}^*\left\{A_{k-1}(m') + \max_{m=0}^{M-1}{}^*\left\{\tilde{C}_k^0(m',m) + B_k(m)\right\}\right\} \qquad 2.2\text{-}8$$

La Figura 2.2-3 muestra un ejemplo del uso de la expresión 2.2-8 en el cálculo de RLP de un bit u_k:

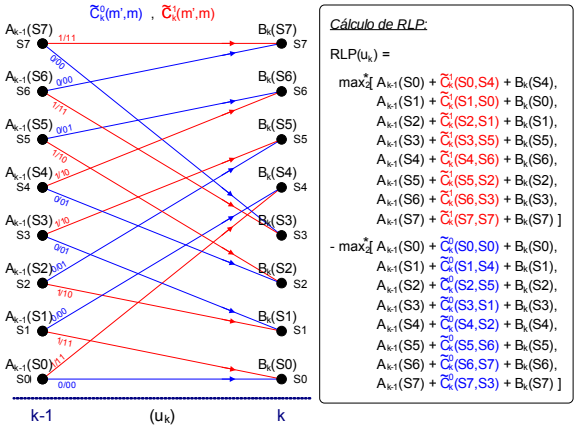

Figura 2.2-3 Ejemplo del cálculo de una RLP

Finalmente, la información extrínseca producida por cada módulo SISO es obtenida considerando una descomposición de la ec. 2.2-8 que relaciona a la RLP, la información *a priori*, la información sistemática y la información extrínseca producida por cada SISO:

$$RLP(u_k) = La_k + 2Lc \cdot y_k^s + Le_k$$
$$\Rightarrow Le_k = RLP(u_k) - 2Lc \cdot y_k^s - La_k \qquad 2.2\text{-}9$$

Tal como se ha descrito hasta el momento, el algoritmo Log-MAP resulta inestable, por cuanto las recursiones progresivas pueden fácilmente ocasionar el desbordamiento de las métricas de estado. La solución generalmente aceptada para este problema consiste en reemplazar a dichas métricas por sus versiones normalizadas, las cuales se obtienen restando, en cada etapa de las recursiones, el valor de cualquiera de ellas. En nuestra implementación, las métricas normalizadas se obtienen restando el valor en el estado cero:

$$A_k(m) \equiv A_k(m) - A_k(S0) \qquad 2.2\text{-}10a$$

$$B_k(m) \equiv B_k(m) - B_k(S0) \qquad 2.2\text{-}10b$$

El desempeño del algoritmo Log-MAP no se ve afectado en absoluto por la normalización, gracias a una importante propiedad del logaritmo de Jacobi:

$$\max_{i=1}^{b}{}^*\{a_i + \theta\} \equiv \max_{i=1}^{b}{}^*\{a_i\} + \theta \quad \forall\, \theta\ cte. \qquad 2.2\text{-}11$$

Así, cualquier constante "θ" en las métricas será cancelada en las ecuaciones de propagación y cálculo de RLP, dejando inafecto el desempeño del algoritmo.

3. ARQUITECTURA Y DISEÑO DEL TURBO DECODIFICADOR

3.1. Implementación analógica del algoritmo

La arquitectura del turbo decodificador presentado se basa principalmente en la implementación analógica del algoritmo Log-MAP empleando la técnica introducida por Gaudet *et al.* en [8] y revisada por los autores en [9]. No obstante, en el presente trabajo los autores plantean algunas adiciones y modificaciones con el objeto de adaptar la técnica a un código de calibre industrial como el aquí considerado. La implementación consiste en la realización de las siguientes operaciones:

Operación	Ecuación
Normalización de métricas de estado	2.2-10
Generación de métricas de ramificación	2.2-7
Propagación de métricas de estado	2.2-4,5
Generación de RLP	2.2-8
Generación de información extrínseca	2.2-9

Tabla 3.1-1 Operaciones implementadas

La técnica seguida se basa en la representación de las métricas del algoritmo Log-MAP a través de voltajes y corrientes directamente proporcionales a éstas:

$$V_{A_k(m)} \propto R.K.A_k(m) + VA_k \qquad 3.1\text{-}1a$$

$$V_{B_k(m)} \propto R.K.B_k(m) + VB_k \qquad 3.1\text{-}1b$$

En las anteriores expresiones R, K son factores multiplicativos que se mantienen constantes a lo largo de todo el diseño. Asimismo, VA_k y VB_k son constantes aditivas que si bien pueden variar entre etapas contiguas de las recursiones, se mantienen constantes para todas las métricas de una misma etapa. La normalización de las métricas (ecs. 2.2-10a,b) se consigue empleando bloques transconductores diferenciales que permiten obtener corrientes proporcionales a las métricas normalizadas:

$$I_{A_k(m)} = -K.A_k(m) + IA \quad (IA = cte.) \qquad 3.1\text{-}2a$$

$$I_{B_k(m)} = -K.B_k(m) + IB \quad (IB = cte.) \qquad 3.1\text{-}2b$$

Las métricas de ramificación también son representadas por corrientes (obtenidas a partir de los símbolos demodulados y de la información *a priori* empleando transconductores), de acuerdo con:

$$I_{C_k^i(m',m)} = -K.C_k^i(m',m) + IC \; (IC = cte.) \qquad 3.1\text{-}3$$

3.2. Celdas analógicas básicas

Modeladas de esta manera las métricas del algoritmo, las operaciones requeridas por éste son realizadas empleando las celdas básicas descritas a continuación.

3.2.1.Transconductor diferencial

Este circuito, mostrado en la Figura 3.2.1, es empleado para la normalización de las métricas de estado progresivas y regresivas, y también en la generación de las métricas de ramificación. Su funcionamiento viene descrito por las expresiones 3.2.1a y 3.2.1b.

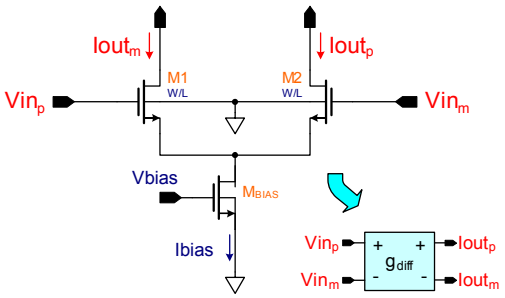

Figura 3.2.1 Transconductor diferencial

$$I_{out_p} = 0.5I_{bias} + 0.5g_{diff}(V_{in_p} - V_{in_m})$$

$$I_{out_m} = 0.5I_{bias} - 0.5g_{diff}(V_{in_p} - V_{in_m}) \qquad 3.2.1a$$

$$g_{diff} = \sqrt{\mu_n C_{ox}(W/L) I_{bias}} \qquad 3.2.1b$$

3.2.2.Circuito max2*

Este circuito, ilustrado en la Figura 3.2.2, es empleado en la generación de las métricas de estado para el cálculo del logaritmo de Jacobi de dos entradas (ec. 2.2-6). El funcionamiento de esta celda se basa en la operación de los transistores M1 y M2 saturados en inversión débil, lo cual les permite ofrecer una característica de transferencia exponencial. Asimismo, M3 y M4 operan en la región triódica de la inversión fuerte, presentando una resistencia promedio "R". El comportamiento de este circuito está dado por las ecuaciones 3.2.2a y 3.2.2b. En dichas expresiones, V_T es el voltaje térmico, n el factor de inclinación de M1 y M2, e I_{DS0} es una constante dependiente del proceso.

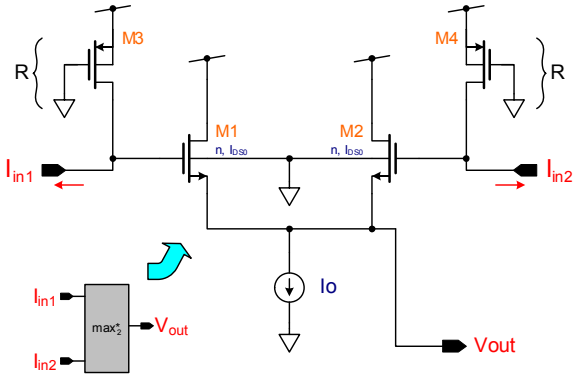

Figura 3.2.2 Circuito max*$_2$

$$V_{out} = nV_T \max^*\left(-RI_{in1}/nV_T, -RI_{in2}/nV_T\right) + V_0 \qquad 3.2.2a$$

$$V_0 = V_{dd} - nV_T \ln\left(I_0/I_{DS0}\right) \qquad 3.2.2b$$

3.2.3.Circuito max$_8$*

La última celda básica consiste en una extensión del circuito max$_2$* para el caso de 8 entradas, la cual es empleada en la generación de las RLP. Este circuito se muestra en la Figura 3.2.3, y su ley de funcionamiento está dada por las ecuaciones 3.2.3a y 3.2.3b.

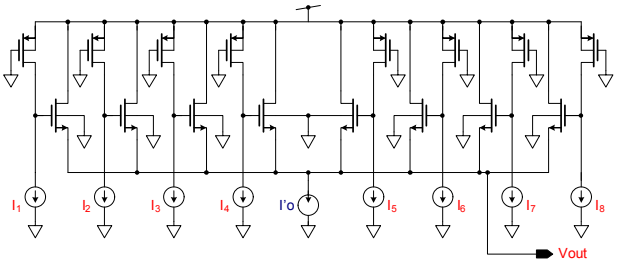

Figura 3.2.3 Circuito max*$_8$

$$V_{out} = nV_T \max_{i=1}^{8}{}^*\left(-RI_i/nV_T\right) + V_0 \qquad 3.2.3a$$

$$V_0 = V_{dd} - nV_T \ln\left(I_o'/I_{DS0}\right) \qquad 3.2.3b$$

3.3. Sub-módulos

A continuación se detallan los sub-módulos empleados en la implementación de las operaciones requeridas por el algoritmo Log-MAP listadas en la Tabla 3.1-1.

3.3.1.Normalizador de métricas

La normalización de las métricas se realiza empleando dos transconductores conectados en paralelo, según se ilustra en la Figura 3.3.1. La aplicación de las ecs. 3.2.1 a este circuito da por resultado la expresión 3.3-1, la cual tiene la forma deseada que se postuló en la ec. 3.1-2a.

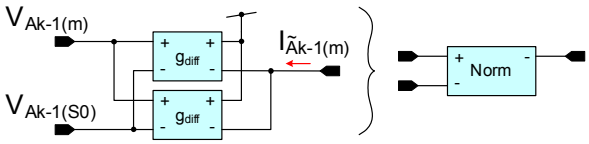

Figura 3.3.1 Normalizador de métricas de estado

$$I_{\tilde{A}_{k-1}(m)} = I_{bias} - g_{diff}\left(V_{A_{k-1}(m)} - V_{A_{k-1}(S0)}\right)$$

$$= -g_{diff} RK\left(A_{k-1}(m) - A_{k-1}(S0)\right) + I_{bias} \qquad 3.3\text{-}1a$$

$$= -K.\tilde{A}_{k-1}(m) + IA$$

$$\Leftrightarrow \quad \boxed{g_{diff} R = 1} \quad (\wedge\ IA = I_{bias}) \qquad 3.3\text{-}1b$$

Es importante notar que el correcto funcionamiento de este módulo requiere que en todo momento se cumpla la condición 3.3-1b, lo cual en la práctica no es factible de ser logrado a la perfección debido a las características no-ideales de los transconductores y de los resistores en los circuitos $\max^*_2$ (i.e., g_{diff} y R presentan variaciones).

3.3.2.Generador de métricas de ramificación

Las métricas de ramificación son generadas a partir de sus tres componentes dadas por las ecuaciones 2.2-7. Para ello se emplean tres bloques transconductores a los que se aplican las salidas del demodulador (símbolos sistemáticos y de paridad) y la información *a priori*, según se muestra en la Figura 3.3.2:

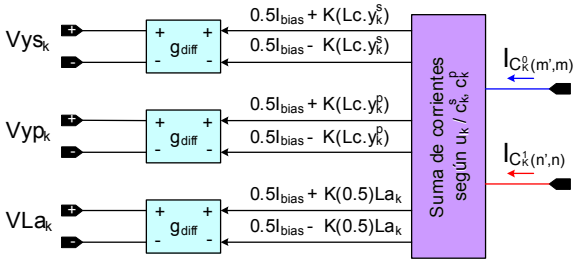

Figura 3.3.2 Generador de métricas de ramificación

$$V_{y_k^{s,p}} = (2K / g_{diff}) Lc.y_k^{s,p} \qquad 3.3\text{-}2a$$

$$V_{La_k} = (K / g_{diff}) La_k \qquad 3.3\text{-}2b$$

Las tensiones diferenciales de entrada, que vienen a ser las entradas al circuito decodificador, se fijan según las ecuaciones 3.3.2a y 3.3.2b. Las corrientes de salida producidas por los transconductores contienen todos los términos necesarios para implementar cualquiera de las componentes de las ecuaciones 2.2-7b, c y d. Entonces, para generar una métrica de ramificación para la transición entre dos estados (m',m), estas corrientes son sumadas según los valores de u_k, c^s_k y c^p_k correspondientes a dicha transición. La aplicación de las ecs. 3.3.1 a este circuito da por resultado las expresiones 3.3-2c,d, logrando la forma deseada para las métricas de la ramificación (ecs. 2.2-7):

$$I_{\tilde{C}_k^0(m',m)}$$

$$= -K(\pm L_c.y_k^s \pm L_c.y_k^p \pm 0.5.La_k) + IC$$

$$= -K\tilde{C}_k^0(m',m) + IC\ , IC = (3/2) I_{bias} \qquad 3.3\text{-}2c$$

$$I_{\tilde{C}_k^1(n',n)}$$

$$= -K(\mp L_c.y_k^s \mp L_c.y_k^p \mp 0.5.La_k) + IC$$

$$= -K\tilde{C}_k^1(n',n) + IC\ \ , IC = (3/2) I_{bias} \qquad 3.3\text{-}2d$$

Nótese que el circuito de la Figura 3.3.2 genera en realidad dos métricas de ramificación, correspondientes a los dos posibles valores del bit de datos u_k considerado. Las métricas de ramificación generadas en forma de corrientes por este módulo son empleadas en la propagación de las métricas progresivas y regresivas, la generación de RLPs, y también en el cálculo de la información extrínseca a "fluir" entre los módulos SISO.

3.3.3.Generador de métricas de estado

La propagación de las métricas de estado progresivas se realiza empleando estructuras como la mostrada en la Figura 3.3.3a, donde se aprecia la generación de la métrica $A_k(Sz)$ a partir de la propagación de las métricas $A_{k-1}(Sx)$ y $A_{k-1}(Sy)$ de acuerdo con el *trellis* del código y previa normalización de éstas respecto de $A_{k-1}(S0)$.

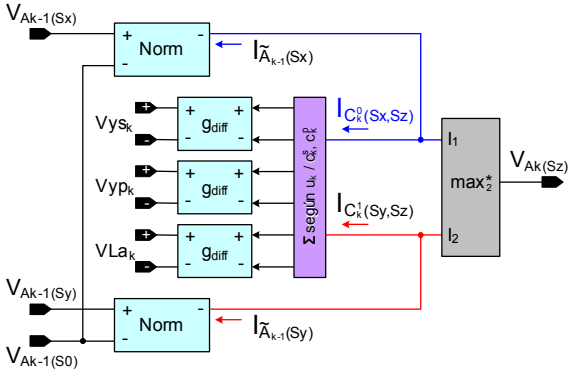

Figura 3.3.3a Propagación básica de las MEP

Nótese que las métricas progresivas a ser propagadas son primero normalizadas, y las corrientes resultantes son sumadas en los nodos de entrada del bloque $\max^*_2$ junto con las corrientes correspondientes a sus respectivas métricas de ramificación. La salida final del bloque se obtiene de la aplicación de la ec. 3.2.2 para las corrientes de entrada consideradas, resultando las expresiones mostradas en las ecuaciones 3.3.3c. y d:

$$I_1 = I_{\tilde{A}_{k-1}(Sx)} + I_{\tilde{C}^0_k(Sx,Sz)}$$

$$= -K[\tilde{A}_{k-1}(Sx) + \tilde{C}^0_k(Sx,Sz)] + IA + IC \qquad 3.3.3a$$

$$I_1 = I_{\tilde{A}_{k-1}(Sy)} + I_{\tilde{C}^1_k(Sy,Sz)}$$

$$= -K[\tilde{A}_{k-1}(Sx) + \tilde{C}^1_k(Sy,Sz)] + IA + IC \qquad 3.3.3b$$

$$V_{out} = nV_T \max^*\left(\frac{-RI_1}{nV_T}, \frac{-RI_2}{nV_T}\right) + V_0$$

$$= nV_T \max^*\begin{bmatrix}\left(\frac{RK}{nV_T}\right)\left(\tilde{A}_{k-1}(Sx) + \tilde{C}^0_k(Sx,Sz)\right), \\ \left(\frac{RK}{nV_T}\right)\left(\tilde{A}_{k-1}(Sy) + \tilde{C}^1_k(Sy,Sz)\right)\end{bmatrix}$$

$$-(R)(5/2)I_{bias} + V_0$$

$$= RK.A_k(Sz) + VA_k = V_{A_k(Sz)} \qquad 3.3.3c$$

$$\Leftrightarrow \boxed{RK = nV_T} \quad \wedge VA_k = V_0 - (5/2)RI_{bias} \qquad 3.3.3d$$

La generación de todas las métricas progresivas en cada etapa del *trellis* se logra apilando 8 veces el bloque de la Figura 3.3.3 (una por cada estado), obteniendo un "generador de métricas de estado progresivas (GMEP)":

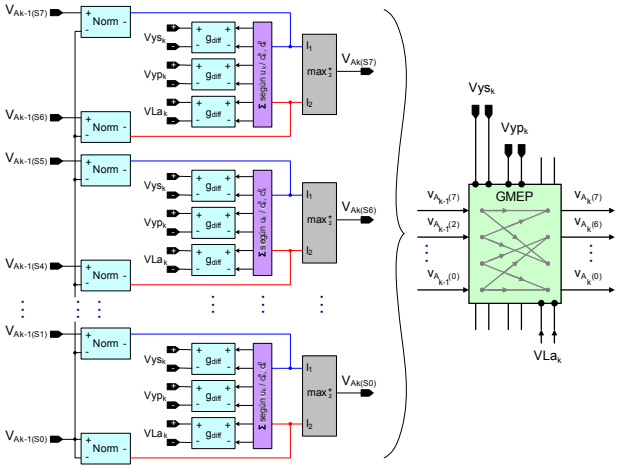

Figura 3.3.3b Bloque GMEP

Es importante notar que el cumplimiento ideal de las condiciones 3.3-1b y 3.3.3d puede en la práctica presentar ciertas desviaciones, debido principalmente a las características no-ideales de los transconductores y de los resistores PMOS. Existe también otra fuente de error en la inicialización de las recursiones, ya que en la práctica las condiciones de inicialización 2.2-4b y 2.2-5b son implementadas empleando valores finitos.

La propagación de las métricas de estado regresivas se realiza de manera idéntica a la vista para el caso de las métricas progresivas, con la diferencia de que esta vez la dirección de la propagación debe invertirse. En la Figura 3.3.3c se muestra la estructura empleada, a partir de la cual se construyen bloques generadores de métricas de estado regresivas (GMER) de manera similar a lo mostrado en la Figura 3.3.3b.

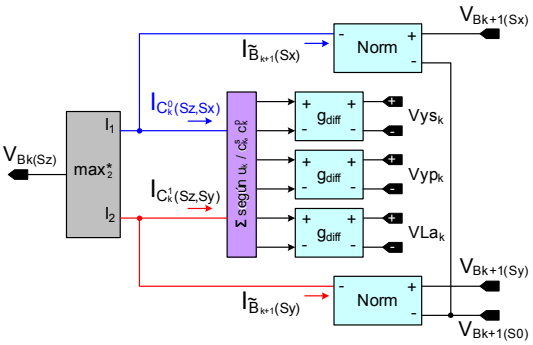

Figura 3.3.3c Propagación básica de las MER

3.3.4.Generador de razón logarítmica de probabilidad

Una vez que los anteriores módulos han sido empleados para generar todas las métricas de estado y de ramificación, la RLP de cada bit de es generada empleando la estructura mostrada en la Figura 3.3.4a. El principio de funcionamiento es muy parecido al visto en las sub-secciones 3.3.2 y 3.3.3, con la diferencia de que ahora intervienen ambas métricas de estado, y que se emplean dos módulos $\max^*_8$ para obtener los dos términos de la ecuación 2.2-8. La RLP final es obtenida en forma de voltaje diferencial entre las salidas de estos módulos, según se indica en las ecuaciones 3.3.4.

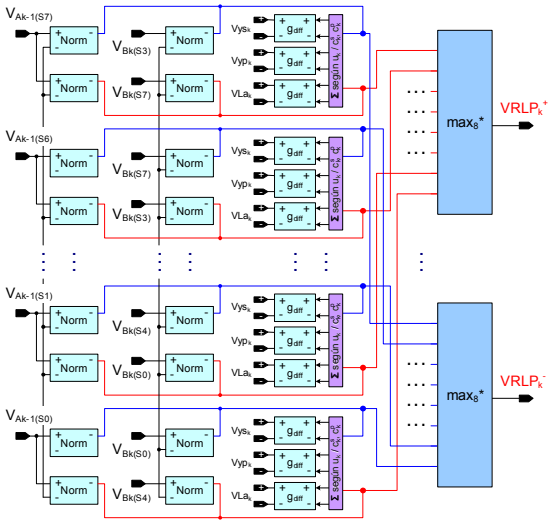

Figura 3.3.4a Bloque generador de RLP (GRLP)

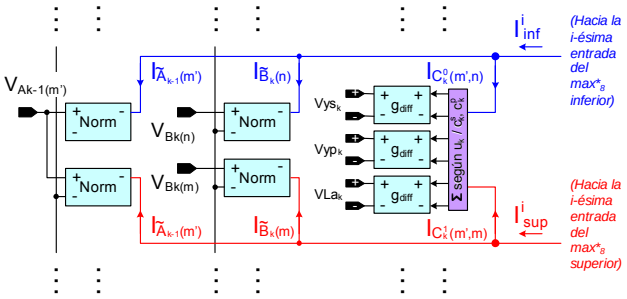

Figura 3.3.4b Detalle de las secciones del GRLP

- Corrientes en las entradas de los max*$_8$: 3.3.4a

$$I^i_{sup} = I_{\tilde{A}_{k-1}(m')} + I_{\tilde{C}^1_k(m',m)} + I_{\tilde{B}_k(m)}$$
$$= -K[\tilde{A}_{k-1}(m') + \tilde{C}^1_k(m',m) + \tilde{B}_k(m)] + (7/2)I_{bias}$$
$$I^i_{inf} = I_{\tilde{A}_{k-1}(m')} + I_{\tilde{C}^0_k(m',n)} + I_{\tilde{B}_k(n)}$$
$$= -K[\tilde{A}_{k-1}(m') + \tilde{C}^0_k(m',n) + \tilde{B}_k(n)] + (7/2)I_{bias}$$

- Tensiones de salida de los max*$_8$: 3.3.4b

$$VRLP^+_k = nV_T \max^{8}_{i=1}{}^*\left(RI^i_{sup}/nV_T\right) + V_0$$
$$= nV_T \max^{8}_{i=1}{}^*\left[\left(RK/nV_T\right)\begin{pmatrix}\tilde{A}_{k-1}(m') \\ +\tilde{C}^1_k(m',m) + \tilde{B}_k(m)\end{pmatrix}\right] - (R)(7/2)I_{bias} + V_0$$
$$VRLP^-_k = nV_T \max^{8}_{i=1}{}^*\left(RI^i_{inf}/nV_T\right) + V_0$$
$$= nV_T \max^{8}_{i=1}{}^*\left[\left(RK/nV_T\right)\begin{pmatrix}\tilde{A}_{k-1}(m') \\ +\tilde{C}^0_k(m',n) + \tilde{B}_k(n)\end{pmatrix}\right] - (R)(7/2)I_{bias} + V_0$$

- Condición para el funcionamiento del bloque max*$_8$:

$$\boxed{RK = nV_T} \qquad 3.3.4c$$

- Tensión diferencial de salida: 3.3.4d

$$VRLP = VRLP^+_k - VRLP^-_k$$
$$= RK \max^{8}_{i=1}{}^*\left[\tilde{A}_{k-1}(m') + \tilde{C}^1_k(m',m) + \tilde{B}_k(m)\right] - RK \max^{8}_{i=1}{}^*\left[\tilde{A}_{k-1}(m') + \tilde{C}^0_k(m',n) + \tilde{B}_k(n)\right]$$
$$= \underline{RK.RLP(u_k)}$$

3.3.5.Generador de información extrínseca

El funcionamiento de todos los módulos vistos hasta ahora parte de asumir que en todo momento se dispone de la información *a priori* "La$_k$", en la forma de un voltaje diferencial (ec. 3.2-2b). De acuerdo con lo explicado en la sección 2.2, dicha información *a priori* no es otra cosa más que la información extrínseca producida por el otro módulo SISO, la cual puede ser obtenida a partir de la RLP empleando la relación 2.2-9. El circuito empleado para esta tarea es el mostrado en la Figura 3.3.5, el cual provee la información extrínseca en la forma de un voltaje diferencial. La operación de este módulo es mostrada en las ecuaciones 3.3.5.

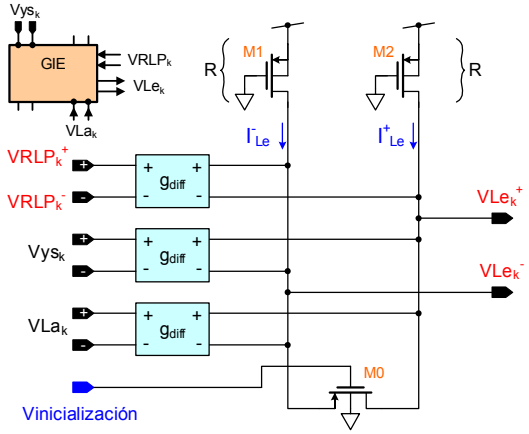

Figura 3.3.5 Generador de información extrínseca (GEI)

$$I^+_{Le} = (3/2)I_{bias} - (K/2)[RLP(u_k) - 2Lc\cdot y^s_k - La_k]$$
$$= (3/2)I_{bias} - (K/2)Le_k$$
$$I^-_{Le} = (3/2)I_{bias} + (K/2)[RLP(u_k) - La_k - 2Lc\cdot y^s_k]$$
$$= (3/2)I_{bias} + (K/2)Le_k$$
$$VLe^+_k = Vdd - RI^+_{Le} = Vdd - (3/2)RI_{bias} + (RK/2)Le_k$$
$$VLe^-_k = Vdd - RI^-_{Le} = Vdd - (3/2)RI_{bias} - (RK/2)Le_k$$
$$\Rightarrow VLe_k = VLe^+_k - VLe^-_k = \underline{RK.Le_k = (K/g_{diff})Le_k} \qquad 3.3.5$$

Nótese que este módulo consta además de una entrada de inicialización, la cual permite cortocircuitar las salidas diferenciales y forzar un valor de información extrínseca nulo. Tal como se indica en la sección 3.4, esta condición es necesaria para un correcto funcionamiento del esquema de turbo-decodificación.

Por último, cabe resaltar que la estructura de la Figura 3.3.5 empleada para la implementación del bloque GIE presenta un desempeño mucho mejor al del módulo previamente reportado por los autores en [9], presentando una reducción drástica en el número de transistores empleados con un consecuente incremento en la rapidez de operación de este bloque.

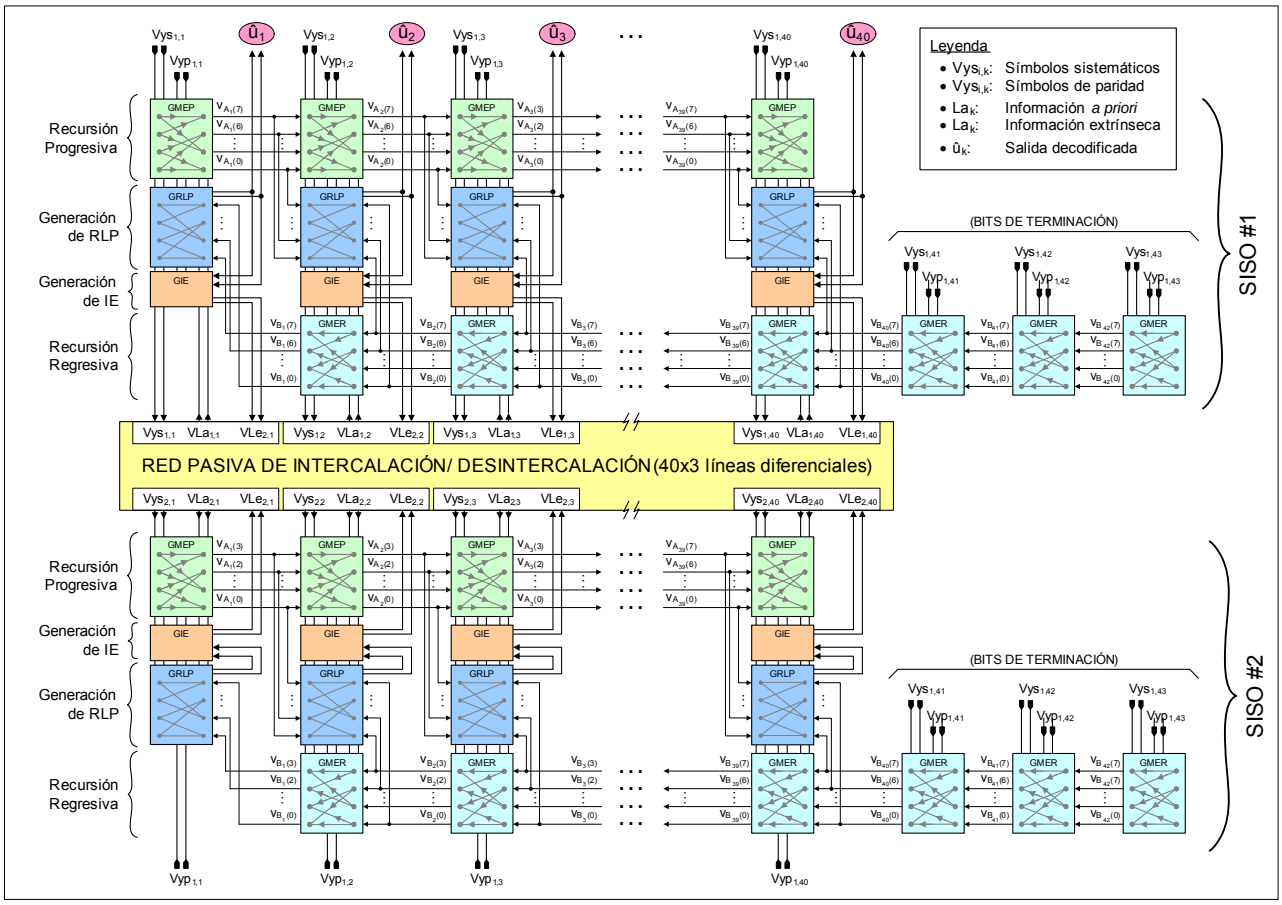

Figura 3.4-1 Estructura completa del turbo decodificador diseñado

3.4. Turbo decodificador completo

El sistema completo de decodificación se ilustra en la Figura 3.4-1, en donde el bloque intercalador implementa la permutación entre las señales de acuerdo con el patrón especificado por el estándar UMTS [6].

Los decodificadores constitutivos que componen el sistema, i.e. los módulos SISO de la Figura 2.2-1, son implementados empleando los bloques ya presentados. Nótese que en cada una de estas redes las correspondientes recursiones progresiva y regresiva se implementan a través de la conexión en cascada de varios bloques GMEP y GMER. Las salidas de estos bloques son a su vez empleadas por los bloques GRLP para generar las correspondientes razones logarítmicas de probabilidad. Finalmente, dichas RLP son empleadas en cada uno de los SISO para la generación de la información extrínseca a ser intercambiada entre los módulos a través del intercalador.

La operación del decodificador es como sigue. En un primer paso, los símbolos demodulados son aplicados a las entradas del circuito, y el camino de realimentación de información *a priori* es "abierto" por medio de la activación de la señal de inicialización de los bloques GIE. Esto se hace con el objeto de lograr que en un inicio los módulos SISO comiencen su funcionamiento sin rezagos de información *a priori* de la última secuencia decodificada. Este estado de funcionamiento independiente es mantenido durante un intervalo de "$T_{inicialización}$" segundos, luego de los cuales las señales de inicialización de los GIE son desactivadas, cerrando efectivamente el lazo de decodificación iterativa. Los módulos SISO comienzan entonces a intercambiar datos, recibiendo información *a priori* y generando información extrínseca. Al final de un tiempo "$T_{decodificación}$", la salidas de los bloques GRLP son muestreadas con el fin de evaluar las RLPs y los valores de los bits de la secuencia final decodificada, los cuales son obtenidos según el signo de la RLP. Luego de esto el ciclo se repite, cargando los nuevos símbolos del demodulador correspondientes a la siguiente trama de datos a decodificar. Bajo estas condiciones la tasa de decodificación del circuito viene dada por:

$$R = 40/(T_{inicialización} + T_{decodificación}) \quad \text{[bps]} \qquad 3.4.1$$

3.5. Metodología de diseño

El flujo de diseño comenzó implementando un modelo de alto nivel del turbo código en consideración y del algoritmo Log-MAP aplicado a su decodificación, el cual fue codificado en MATLAB®. Este modelo permitió validar el desempeño teórico del código para distintas condiciones de trabajo, obteniéndose las curvas de tasa de errores de bit mostradas en la Figura 3.5-1. Cabe resaltar la imposibilidad práctica de efectuar este tipo de mediciones de desempeño a través de simulaciones de la arquitectura diseñada empleando modelamiento a nivel de componentes (SPICE), debido al enorme tiempo que tardaría la realización de un número suficientemente grande de dichas simulaciones como para obtener una la lectura significativa del BER a valores altos de EbNo. Este modelo proveerá además a los autores de una cota comparativa contra la cual evaluar el desempeño real del circuito una vez fabricado.

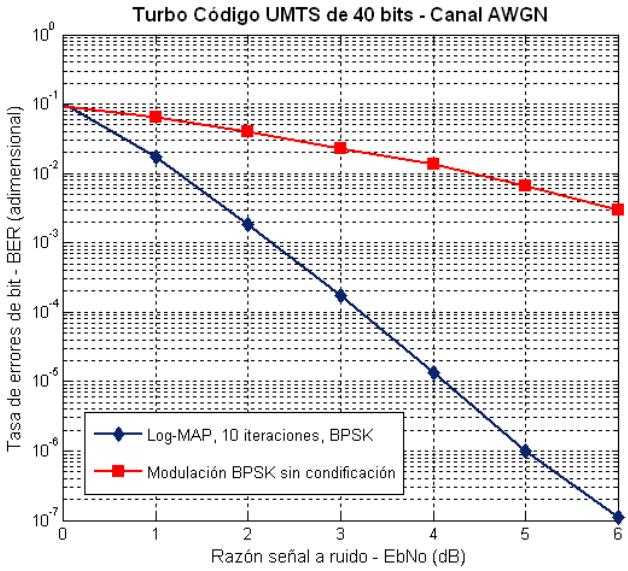

Figura 3.5-1 Desempeño teórico del código

Luego de validar el funcionamiento del algoritmo de decodificación, se elaboró un modelo de primer orden del circuito decodificador, modelando cada uno de los bloques que lo componen empleando las expresiones ideales presentadas en la sección anterior. Con este modelo se determinaron los valores óptimos para las constantes g_{diff}, R y K a emplear en la implementación, así como las corrientes de polarización de las celdas. Este modelo permitió asimismo analizar en una primera aproximación la robustez de la arquitectura ante potenciales variaciones en los parámetros de las celdas básicas del decodificador (especialmente g_{diff} y R), así como los efectos del truncamiento de las métricas de estado iniciales a valores finitos en las recursiones progresiva y regresiva.

4. RESULTADOS DE SIMULACIÓN

La arquitectura descrita en las secciones previas fue implementada a nivel de captura esquemática sobre el proceso CMOS de 0.35µm de AMS, empleando la herramienta Cadence®.

En la Figura 4.1 se muestran los resultados de simulación para la decodificación del bit "u8" de 4 tramas de datos consecutivas, transmitidas con un EbNo de 0.3dB. En este ejemplo, se han empleando un tiempos de inicialización y decodificación iguales a 100ns, tomando el procesamiento de cada trama 200ns. Los valores correctos para este bit son, respectivamente, "1", "0", "1" y "0", mas puede verse como en todas las tramas el ruido del canal ha provocado valores errados para las salidas sistemáticas que le corresponden (nótese la polaridad relativa de las señales Vys_1_8_p y Vys_1_8_n). No obstante, todos estos errores son corregidos por el circuito, obteniéndose lo valores correctos de RLP en la salida. En esta figura puede apreciarse además la evolución de las señales de información a priori y extrínseca, así como el empleo de la señal de inicialización de los bloques GIE de acuerdo con lo señalado en la sección 3.4.

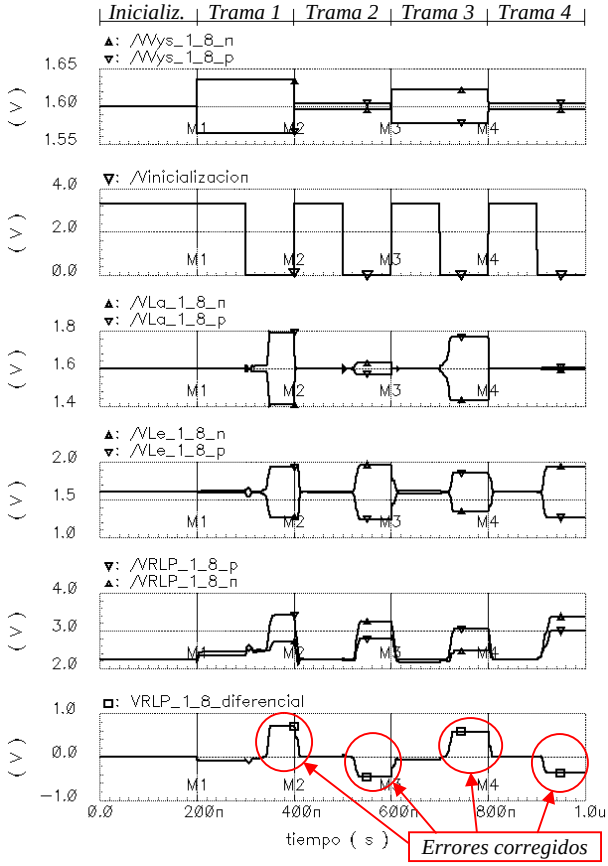

Figura 4.1 Funcionamiento del decodificador

Empleando la ec. 3.4.1 con los valores mencionados en el ejemplo anterior para los tiempos de inicialización y decodificación, se obtiene una tasa de decodificación efectiva de 200Mbps. Un resumen con las principales medidas del desempeño simulado del turbo decodificador se presenta asimismo en la Tabla 4.1, en donde además se hace una comparación con aquellas del núcleo decodificador de la referencia [5]. Se observa que si bien el consumo de potencia es relativamente elevado, las figuras de mérito más importantes –la tasa de decodificación y el consumo de energía por bit decodificado y estado en el trellis– son excelentes. Al mejor conocimiento de los autores, estos son los mejores resultados reportados a la fecha para un decodificador analógico del turbo código UMTS de 40 bits. Cabe resaltar, no obstante, que una comparación en términos de consumo de área es imperativa, y será realizada una vez que la arquitectura sea implementada por los autores a nivel de layout.

Medida de desempeño	Ref. [5]	Este trabajo
Número de transistores	30,000	40,000
Tensión de alimentación (V)	3.3	3.3, 5.0, 6.7
Potencia (W)	0.0068	1.37
Energía / bit / estado (nJ)	1.4	0.86
Tasa de decodificación (Mbps)	2.0	200

Tabla 4.1 Desempeño simulado del decodificador

5. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo se ha presentado la concepción y el diseño de un turbo decodificador analógico en tecnología CMOS 0.35µm para el código de corrección de errores UMTS de 40 bits. El desempeño simulado de este circuito es satisfactorio y se sitúa por encima de las implementaciones existentes a la fecha. Los autores continuarán con la implementación del diseño a nivel de layout con el objeto de evaluar cuán robusta se muestra esta arquitectura ante las variaciones propias de los defectos de fabricación, y para obtener una estimación realista del área ocupada por el diseño y de la potencia total consumida por el circuito integrado.

6. BIBLIOGRAFÍA

[1] C. Berrou *et al., "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes",* IEEE Intrnl. Conf. on Communications, Geneva, Suiza, p. 1064-1070, Mayo 1993.

[2] H. Loeliger *et al., "Probability Propagation and Decoding in Analog VLSI",* Proc. 1998 IEEE Int. Symposium on Information Theory, Cambridge, USA, agosto de 1998.

[3] J. Hagenauer, M. Winklhofer, *"The analog decoder",* Proc. 1998 IEEE International Symposium on Information Theory, Cambridge, USA, agosto de 1998.

[4] Winstead , C. Schlegel, *"Analog.Decoding: the State of the Art",* IEEE Int. Symposium on Spread-Spectrum Techniques and Applications, Sydney, Australia, septiembre de 2004.

[5] D. Vogrig *et al., "A 0,35 µm CMOS Analog Turbo Decoder for the 40-bit, rate 1/3, UMTS Channel Code",* IEEE Journal of Solid-State Circuits, marzo de 2005.

[6] Third Generation Partnership Project (3GPP), *Document 25.212*, www.3gpp.org/ftp/Specs/2000-12/R1999/25_series/25212-350.zip, diciembre de 2000.

[7] P. Robertson, *et al., "A comparison of optimal and suboptimal MAP decoding algorithms operating in the log domain",* Proc. Int. Conf. Communications, junio de 1995.

[8] V. Gaudet, P Gulak, *"A 13.3Mbps 0.35mm CMOS Analog Turbo Decoder IC with a Configurable Interleaver",* 2003 IEEE Int. Solid-State Circuits Conf., febrero de 2003.

[9] J. Lagos, C. Silva, *"Diseño de un Módulo SISO para la Decodificación Analógica de Turbo Códigos",* XI Workshop IBERCHIP 2005, Salvador de Bahía, Brasil, marzo de 2005.