xx(101590.1)

CINVESTAV - IPN
Centro de Investigación y de Estudios Avanzados del IPN
Unidad Guadalajara

Diseño y Verificación de un Generador de tramas SDH

TESIS QUE PRESENTA
ENRIQUE GONZALEZ GARCIA

PARA OBTENER EL GRADO DE
MAESTRO EN CIENCIAS

EN LA ESPECIALIDAD DE
INGENIERÍA ELÉCTRICA

Guadalajara, Jal., Diciembre de 2001

Diseño y Verificación de un Generador de tramas SDH

Tesis de Maestría en Ciencias
Ingeniería Eléctrica

por:

Enrique González García

Ingeniero Biomédico
Universidad Autónoma Metropolitana, 1993–1997
Becario del CONACYT, expediente no. **129168**

Directores de Tesis:

Dr. Deni Librado Torres Román
Dr. Arturo Veloz Guerrero

CINVESTAV del IPN Unidad Guadalajara, Septiembre de 2001.

Índice

Agradecimientos:

A mi madre, abuela y todas mis tías, que gracias a su cariño y educación me permitieron desarrollarme para alcanzar y lograr las metas que me propongo.

A mi única hermana por su enorme apoyo incondicional que siempre he tenido. A todos mis hermanos por sus consejos y experiencias que siempre me ayudaron en todo momento de mi vida.

A mis amigos Emigdio, Alberto y Carlos que de alguna forma u otra siempre me animaron para seguir terminar con este trabajo de tesis.

En Guadalajara llegue sin conocer a nadie pero con el paso del tiempo empecé a conocer diferentes personas las cuales en muchas ocasiones resultaron ser simples conocidos como sucede a lo largo de la vida, sin embargo, con Jacobo, Rogelio, Alejandro, Luis, Ivan, Eduardo, es diferente dado que considero que son de los nuevos amigos que duraran toda la vida independientemente del lugar geográfico en donde uno se encuentre.

A todos mis compañeros de trabajo de los cuales siempre estoy aprendiendo algo nuevo, a si como su disposición de compartir su experiencia, en especial a Jesús Palomino.

A todos mis compañeros de CINVESTAV Unidad GDL, por compartir esos momentos de trabajo duro cuando no importaba que tan dura fuera la tarea a realizar siempre nos apoyamos hasta cumplirla.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT), por brindarme el apoyo económico, durante dos años, para dedicarme por completo al estudio de esta maestría.

A mis asesores, El Dr. Deni Torres y Dr. Arturo Veloz, por abrirme la puerta al maravilloso mundo del diseño digital a sí como también por dirigir este trabajo y colaborar para el resultado del mismo.

A J. Hermosillo por contribuir con valiosos comentarios para la mejora del presente trabajo.

1 Capítulo I

Introducción

En el presente trabajo se expondrá el diseño y verificación de un generador multipuerto de tramas SDH(synchronous digital hierarchy).

El hombre siempre ha tenido la necesidad de interactuar con el resto del mundo; esta interacción se ha incrementado con el paso de los años. Conceptos como redes de voz y redes de datos se trabajaban por separado y eran transportados con formatos de tramas PDH(plesiochronous digital hierarchy); sin embargo, conceptos tales como la internet, teleconferencias, red digital de servicios integrados exigen tecnologías nuevas que permitan soportar conjuntamente ambos tipos de redes con mejor eficiencia y menor costo[9]. La respuesta a estas exigencias se está dando con las redes ópticas.

Existen protocolos usados en el mundo que definen todos los aspectos técnicos para el manejo de dichas redes, de los más importantes se encuentran SONET y SDH. El primero es el empleado en la Unión Americana y el segundo en la Comunidad Europea.

La implementación de dichos protocolos se realiza en circuitos lógicos de muy alta escala de integración, los cuales resultan ser sumamente complejos y por lo tanto susceptibles de errores. A manera de respuesta a este problema se establece dentro del diseño una metodología de verificación la cual consiste en simular diferentes condiciones de trabajo o estrés al sistema, observando la respuesta y compararla con una respuesta esperada. Dicho procedimiento permite la depuración antes de que se lleve a la implantación del sistema a una oblea de silicio.
Como piedra angular en el proceso de verificación se encuentra el dispositivo que permite estimular al diseño para simular condiciones de trabajo o estrés. Dicho dispositivo es conocido como: cama de pruebas, la cual consiste de un generador de estímulos y un analizador de estímulos. La cama de pruebas resulta ser otro diseño, el cual debe ser diseñado y verificado a su vez. Con lo expuesto anteriormente se pensaría que nunca se romperá el circulo vicioso de tener errores en ambos diseños, sin embargo cuando se ponen a trabajar conjuntamente cama de pruebas y diseño se comienzan a depurar mutuamente. Ya que si se encuentra una incompatibilidad en la respuesta se debe de determinar cual es el causante de dicho problema.

El siguiente paso es realizar pruebas a la oblea en el laboratorio a dicho proceso se le conoce como validación. Es importante hacer mención que la verificación tiene suma importancia en el proceso de diseño y en una metodología de diseño estas dos tareas deben ser realizadas de manera paralela[6].

En párrafos posteriores definiremos diferentes términos que hasta el momento solo se mencionan de forma superficial, como por ejemplo especificación, arquitectura, etc.

La presente tesis es parte de un proyecto más grande y por lo tanto en este trabajo solo se escribirá la especificación, arquitectura de un generador genérico de tramas SDH así como su verificación, el cual puede satisfacer algunas de las necesidades que exige la industria de Telecomunicaciones relacionada con diseños de circuitos que manejen el protocolo SDH.

1.1 Modelos de Verificación

La *verificación* tiene el propósito de cotejar que el diseño cumple con lo escrito en el documento de especificación[14,16], entendiéndose como la funcionalidad y limitantes definidas del circuito. Para lograr lo anterior se realizan un conjunto de pruebas a las cuales es sometido el diseño[17] para posteriormente una vez aprobadas dichas pruebas se obtiene la confianza para implementar el diseño en una oblea de silicio, cuando se realiza la implementación se procede con la *validación* la cual consiste en estimular la oblea y cotejar que la respuesta corresponda con lo establecido por el cliente[14].

Este conjunto de pruebas en la verificación exige producir una secuencia de señales por cada prueba; tales señales son los estímulos al circuito que se desea probar. El dispositivo que produce estas señales es llamado Generador y al dispositivo que recibe la respuesta del circuito al ser estimulado es llamado Analizador. Conjuntamente Generador y Analizador conforman una cama de pruebas (testbench)[10,15]. En la figura1 se muestra el modelado de una cama de pruebas y un diseño.

Figura 1. Estructura genérica de una cama de pruebas y un diseño bajo prueba.

Desde un grado mayor de abstracción, es posible representar la verificación de un sistema por medio del modelo de reconvergencia el cual es representado en la figura 2[6] la transformación es el proceso que toma un conjunto de entradas y produce una salida. La verificación consiste en observar que las salidas obtenidas realmente sean las que se esperaban. En caso de que no se cumpla la respuesta esperada se deben realizar modificaciones en la transformación. Este modelo se aplica continuamente hasta que se tenga la respuesta esperada.

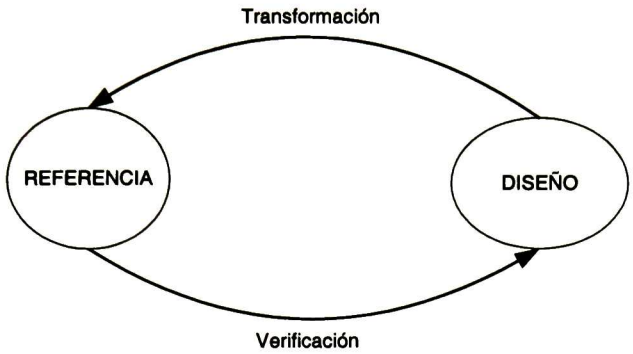

Figura 2. Modelo de reconvergencia.

Un diseño se puede codificar a nivel RTL (register transfer level) es codificado en un lenguaje de alto nivel como por ejemplo HDL (High Description Language), se estimula por medio de ciertos conjuntos de señales denominadas vectores de prueba. Si la respuesta que se produce no es la que se deseaba entonces debe de ser modificado hasta que se obtenga la respuesta esperada[6,7].
Resulta natural hacerse la pregunta con respecto a lo que es considerado "respuesta correcta" en la verificación. El punto de referencia es un documento llamado especificación del sistema(requerimientos), que debe ser consultado tanto por el diseñador como por el verificador, e incluso por cualquier otra persona que esté interesado en saber la funcionalidad y características especiales del dispositivo[4]. Lo anterior se puede representar en la figura 3.

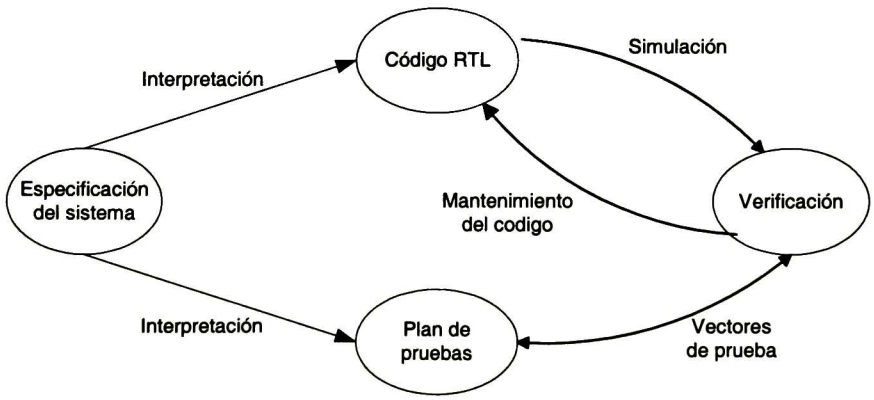

Figura 3. Proceso de verificación de un sistema

1.2 Proceso de Verificación

Como se mencionó con anterioridad, la verificación funcional se enfoca en el modelo de reconvergencia. El objetivo es asegurarse que el diseñador implemente el comportamiento descrito en el documento de especificación del circuito. Con la verificación funcional se determina la confianza de cómo fue transformada la especificación del circuito a un código RTL[6,5].

Es un buen momento para comentar un poco sobre una especificación dado que se ha establecido como punto de referencia entre diseñador y verificador. Una especificación es un documento escrito, en un lenguaje natural (español, inglés, etc), por personas que pueden tener diferentes niveles del dominio del lenguaje[3] interesadas en definir las características de un sistema. Por tal motivo resulta interesante tener conciencia de qué tanto el diseñador como el verificador pueden interpretar de diferente manera el documento y por tal motivo entrar en controversia respecto a los resultados obtenidos y los esperados; en tales casos es necesario realizar un análisis de la situación y determinar si es necesario reescribir alguna parte del documento con más detalle o simplemente sobre la base de lo que está escrito en la especificación y con las normas internacionales(ITU, ANSI, etc) se puede llegar a un punto de acuerdo.

1.2.1 Verificación Funcional

La verificación funcional tiene tres tipos de formas de ser desarrollada, las cuales son: considerar el diseño como caja negra, caja blanca y por último como caja gris[6,3].

La *caja negra* desconoce totalmente la estructura e implementación del módulo. En este modo no se tienen visibilidad ni controlabilidad, y si existe un problema resulta imposible determinar cual es la posible causa de falla. La principal ventaja es que no depende de una implantación en particular. Para poderle proveer "cierta" controlabilidad y visibilidad a las pruebas es necesario acceder a determinados registros internos. Lo anterior no es la única forma de poder obtener estas características deseadas ya que también se puede hacer uso de observar señales internas del diseño.

La segunda forma conocida es la llamada *caja blanca*, en la cual es posible saber y tener acceso a la estructura interna del módulo sabiendo detalles de cada submódulo. De esta manera es más fácil observar los resultados y determinar alguna discrepancia entre lo esperado y lo realizado, y en tal caso es posible determinar en donde se está generando algún posible problema.

La *caja gris* es un compromiso entre caja negra y caja blanca dado que se puede conocer la estructura interna del módulo pero sin profundizar en los detalles de cada submódulo.

1.2.2 Verificación Formal

La verificación formal se clasifica en dos categorías: Chequeo de equivalencias y chequeo del modelo.

El *chequeo de equivalencias* actualmente es el más empleado. Su uso actual es para comprobar que el código RTL y su correspondiente representación a un nivel de compuertas logicas (netlist) tienen el mismo comportamiento, así como también comparar el "netlist" vs netlist cuando se le realizo alguna modificación y se quiere que demostrar que a pesar de dicha modificación lógicamente son equivalentes ambas representaciones. Como última aplicación se tiene la comparación de dos códigos RTL y se quiere demostrar que son equivalentes, entendiéndose equivalencia al hecho de que a una misma entrada se obtiene la misma salida[6].

Como segunda categoría se encuentra el *chequeo de modelo*, el cual es menos empleado. En fases tempranas de desarrollo consiste en demostrar la equivalencia entre la especificación escrita en un lenguaje no coloquial (VHDL por ejemplo) y el código RTL generado. La desventaja de esto es que resulta muy complicado escribir una especificación en un lenguaje no coloquial, y desarrollar la herramienta que entienda estos códigos[6].

Las técnicas de verificación formal son muy costosas y complejas, y lo anterior hace que la verificación funcional tenga preferencia y mayor aceptación para las primeras etapas de verificación en donde la gran mayoría de los errores deben ser encontrados.

1.3 Metodología de Verificación

Lo que se expondrá a continuación es un conjunto de tareas generales que pueden ser aplicadas a un diseño de software o hardware que quiera ser verificado.

Las tareas a realizar son: análisis, diseño, implementación y mantenimiento del diseño[8,3]. En la etapa de análisis se determinan objetivos de las diferentes pruebas, cronología en la aplicación de las pruebas, requerimientos necesarios para comenzar la verificación. En el diseño lo que se pretende es generar y detallar las pruebas que se realizarán. Cuando se tiene lo anterior lo que procede es implementar las pruebas descritas en la etapa de diseño así como también ejecutarlas. Como último paso se tiene la fase de mantenimiento, y en esta etapa es donde se actualiza alguna prueba o se agrega otra prueba no contemplada en un principio; se puede decir que esta fase está presente en toda la vida de la verificación.

Todo lo anterior tiene un conjunto de documentos que permite formalizar la metodología, los cuales son: Especificación de un caso de prueba, Plan de

pruebas, especificación de las pruebas, casos de prueba, reportes de las pruebas.

La *especificación de un caso de prueba* define las entradas, resultados esperados, conjunto de detalles en la ejecución de la prueba así como también la secuencia de acciones para la ejecución[18].

El *plan de pruebas* es un documento que describe el alcance, estrategia, recursos y programar las actividades que se quieren probar. Identificando los detalles a probar, características de las pruebas, como deben realizarse las pruebas[18].

Los *reportes de las pruebas* son documentos que resumen las actividades de las pruebas y sus resultados. Además contienen una evaluación de los detalles correspondientes a las pruebas[18].

Existen tres niveles de profundidad en las pruebas, estos subgrupos de pruebas se conocen como pruebas por módulo, pruebas de subsistema y pruebas de sistema[5,4].

La *pruebas por módulo* se enfocan en objetivos particulares y todo el conjunto de pruebas debe permitir probar en forma exhaustiva a este módulo para de este modo lograr una gran confiabilidad cuando se comiencen las pruebas de subsistema. En las *pruebas de subsistema* se tienen interactuando diferentes módulos previamente probados de forma exhaustiva. Lo que se enfoca a este nivel de pruebas es la diferente interacción que se tiene entre ellos. Aquí las pruebas ya no son exhaustivas debido a que los tiempos de simulación son considerables respecto a las simulaciones que se hacían a un nivel de pruebas de un solo módulo. Por último se tienen las *pruebas de sistema* en donde ya se tienen integrados todos los diferentes subsistemas que conformarán el diseño, y el objetivo es verificar que el diseño funciona en su totalidad; las pruebas tienen por tarea conseguir la mayor interacción entre los diferentes subsistemas.

La especificación de las pruebas es un documento que describe el objetivo de cada prueba así como también los diferentes estímulos que deben ser aplicados y qué respuesta se espera de los estímulos. A cada una de estas pruebas escritas de esta forma se les llama casos de prueba y cuando se transforman los casos de prueba en un particular lenguaje se dice que se tiene vectores de prueba, que realmente es lo que se aplica al dispositivo a ser verificado. La respuesta que se obtiene después de ser aplicado un vector de prueba se llama reportes de prueba. La variedad del formato de los reportes depende de cada aplicación ya que por ejemplo si se quiere hacer regresiones es recomendable que los reportes salgan en blanco para indicar que no existió error en el dispositivo a ser probado[7]. Sin embargo, si lo que se quiere es hacer una verificación rápida la simple observación de las señales es suficiente. Cabe aclarar que hay ciertas ventajas y desventajas en ambos tipos de reportes como

son que en el primero se debe emplear más tiempo en la obtención de un reporte en blanco pero en regresiones la verificación resulta ser más rápida. En cambio con el otro formato de reporte se pierde la rapidez de la verificación pero se gana tiempo en la obtención de resultados, regularmente cuando se está en las primeras fases de diseño es más utilizada la observación de las señales.

Es recomendable utilizar herramientas que lleven una estadística de las líneas que se ejercitaron cuando se estuvo realizando la verificación puesto que con estas herramientas es muy factible poder descubrir líneas de código que no fueron ejercitadas en la verificación y que posiblemente se pueda llegar a presentar un error en el diseño que sea fabricado; como un ejemplo a este tipo de herramientas se tiene la "cobertura del código"[6,5,4].

1.4 Trama de la Jerarquía Digital Sincronía (SDH)

En párrafos anteriores se ha expuesto un panorama general referente a verificación. El siguiente tópico que se expondrá es referente a SDH dado que se hace necesario mostrar un panorama general, ya que en capítulos siguientes se hablará del diseño de un generador genérico de tramas SDH por lo tanto es conveniente la familiarización de términos.

Cuando se comenzaron las redes de comunicación se utilizaban protocolos de comunicación tales como E1/T1, sin embargo cuando surgió el concepto de tener una red de redes en donde las velocidades a las cuales se pretendían transportar la información resultaban muy ambiciosas tanto en costo como en complejidad, la respuestas a dichas necesidades fue la implementación las redes ópticas, las cuales contaban con su propio protocolo de comunicación. En la Estados Unidos a este protocolo se le llamo Synchronous Optical Network(SONET) mientras que en Europa unos años más tarde definieron su propio protocolo llamado Synchronous Digital Hierarchy(SDH). En ambos protocolos se define por ejemplo como poder transportar aplicaciones tales como: Asyschronous Tranfer Mode (ATM), Gigabit Ethernet o tramas E1 o T1[19]. Ambos protocolos son sumamente parecidos ya que cuentan con[20]: una organización de la trama muy similar, la razón de transmisión de información es similar, tienen criterios parecidos en la manera en que se sincroniza la trama , la multiplexación y demultiplexación de la trama es la misma, cuentan con el mismo control de errores. A pesar de estas similitudes también cuentan con ciertas diferencias las cuales son[20]: definición de los bytes de tara, la interfase óptica de SDH define más parámetros que la establecida en SONET, las definiciones en SONET y SDH son técnicamente iguales pero lingüísticamente diferentes.

Uno de los aspectos importantes en este protocolo es la definición de su trama por lo tanto a continuación se dará una revisión general al respecto.

Para comprender en términos generales la división de la trama es necesario explicar los cuatro niveles o capas en los que se concibe la arquitectura SDH, donde conceptualmente cada nivel requiere de los servicios de los niveles inferiores para ejecutar apropiadamente su función. El primero es el fotónico (photonic) que se encuentra relacionado con las propiedades ópticas en el medio físico de la transmisión; su principal función es la conversión de la trama de un medio de transporte eléctrico, el cual es referido como Módulo de Transporte Síncrono (syschronous transport module STM), a un medio óptico (optical channel OC). El siguiente nivel es llamado de Regeneración el cual está directamente ligado con la Tara de Sección de Regeneración (regeneration section overhead RSOH) de la trama SDH. Sus principales tareas son: monitoreo de errores, entramado, aleatorización de la señal y transporte de la tara. Como tercer nivel se encuentra el de Multiplexaje, que está directamente ligado con la Tara de Sección de Multiplexaje (multiplex section overhead MSOH); desempeña acciones de multiplexaje y sincronización las cuales son necesarias para la creación y mantenimiento de la carga útil de la trama SDH. Finalmente el nivel de trayectoria, cubre la transmisión desde un cliente hasta otro cliente, el cual está relacionado con una pequeña tara que está asociada directamente a la carga útil a transportar, Tara de trayecto(path overhead regeneration section overhead POH). Lo anterior puede ser observado en el siguiente diagrama.

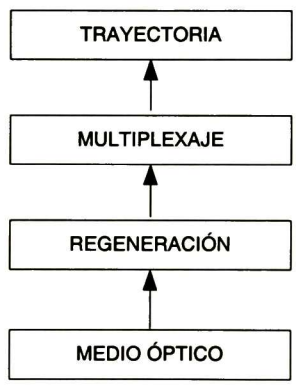

Diagrama1. Arquitectura de 4 capas de SDH

1.4.1 Trama SDH

La tara en la trama SDH de acuerdo al estándar[11] está dividida en RSOH, MSOH, y POH. A continuación se describirá cada un de los campos que conforman la trama, La estructura de la trama STM-N se muestra en la figura 4.

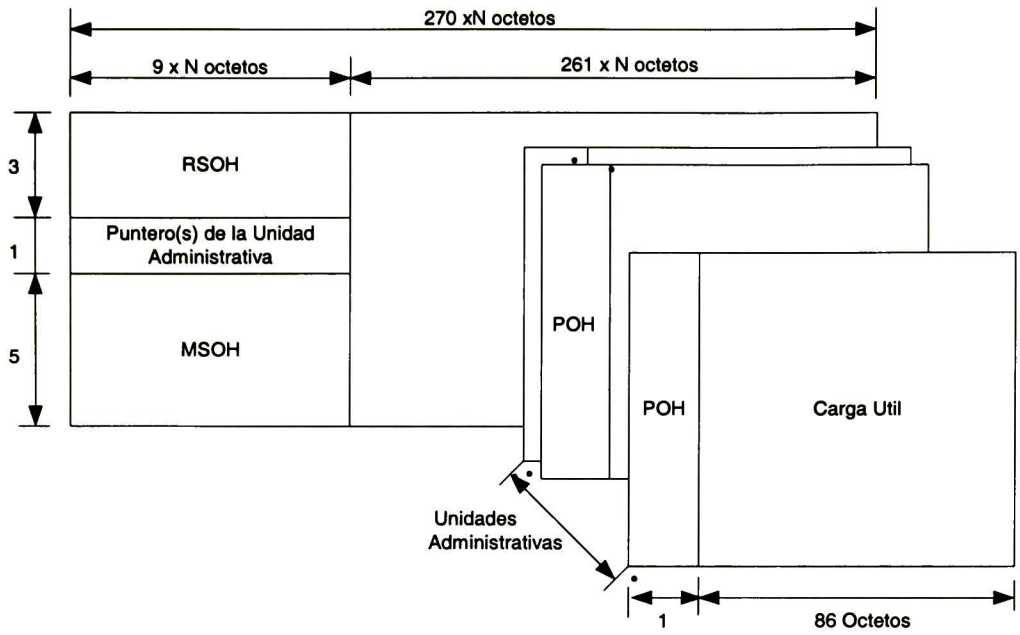

Figura 4 Estructura de la Trama SDH

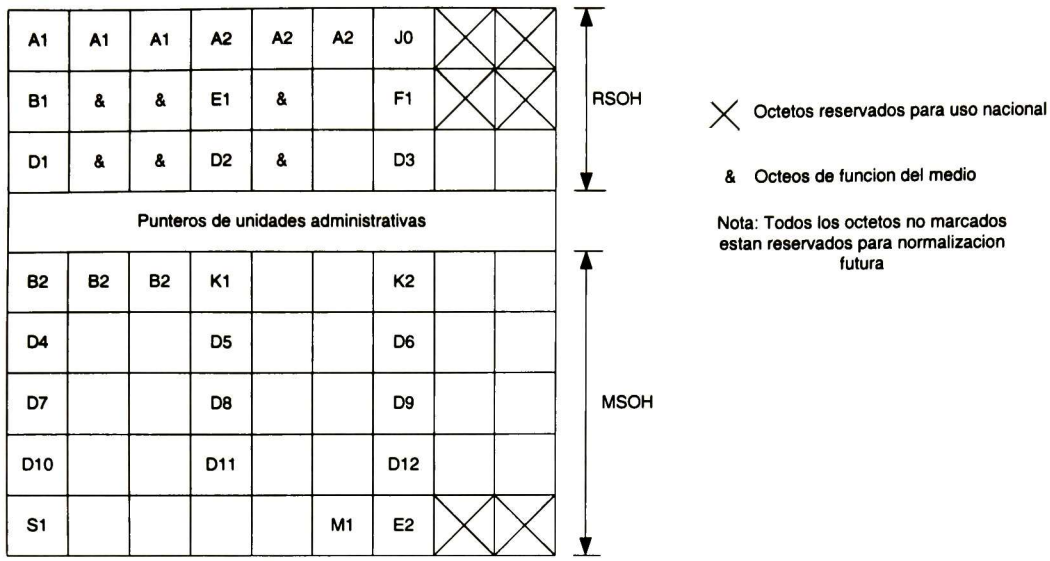

Figura 5. Estructura de la Tara

El RSOH de una trama STM -1 está formado por las primeras nueve columnas y 3 primeras filas de la trama, tal como se muestra en la figura 5, a continuación se da la descripción de la función de los bytes que conforman el RSOH de la trama.

- Bytes de trama(A1, A2): Estos dos bytes son empleados para la alineación de la trama, identificando el comienzo de cada trama STM-1 y no son aleatorizados durante el proceso de transmisión; tienen asignados por defecto el valor F628 en hexadecimal (Hex). La palabra de alineación de trama de una trama STM-N se compone de 3 x N octetos A1 seguidos de 3 x N octetos A2.
- Traza (J0): Este byte es usado para rastrear el origen de una trama STM-1 que viaja a través de la red SDH. En el caso de múltiples STM-1 dentro de STS-N el byte J0 es definido sólo para el primer STS-1.
- Reserva Z0: Se reservan para una futura normalización internacional.
- BIP-8(B1): Realiza un monitoreo de errores en la seccion de regeneración, por medio del cálculo de la paridad par de una trama STM-1 anterior, después de la aleatorización. El cálculo se inserta en el campo BIP-8 de la trama actual antes de entrar al proceso de aleatorización.
- Canal de voz (E1): Es utilizado como un canal de voz entre terminales remotas para poder hacer actividades de mantenimiento.
- Canal de usuario(F1): Campo utilizado para funciones de Operación administración y mantenimiento(OAM&P); por ejemplo conexiones temporales de canales de datos y voz.

- Canales de comunicación de datos(D1,D2,D3): Estos tres bytes son utilizados cuando se realizan operaciones OAM&P tales como alarmas y mensajes de mantenimiento entre otros mensajes.

Punteros

Otro importante componente en una trama son los bytes de punteros, los cuales se definen como:

- Punteros(H1, H2): Estos bytes son utilizados para permitir una alineación flexible y dinámica de un contenedor virtual (VC-n) dentro de una unidad administrativa (AU-n). La alineación dinámica significa que se permite al VC-n flotar dentro de la trama de AU-n, para lo cual se deben modificar H1, H2 dependiendo de las variaciones en la frecuencia. Los valores que pueden tomar los punteros se encuentran en el rango de 0 a 782 decimal.
- Puntero de acción(H3): Su objetivo es compensar la Carga útil STM-N de las variaciones de tiempo, su información contenida dependen de los bytes de punteros H1 y H2.

MSOH

El otro campo de tara es el de Multiplexación o MSOH *(Multiplex section Overhead),* encontrándose ubicado en las primeras nueve columnas y de las filas quinta a la novena, es posible observar la posición dentro de la trama en la figura 5, la descripción de cada campo se da a continuación.

- BIP-8(B2): Similar a B1, su diferencia radica en que contiene la paridad con entrelazado de bits de la trama de STM-N con paridad par antes de agregarle la tara de sección de regeneración, calculándose a partir de la trama anterior antes de la aleatorización y se coloca en los octetos B2 antes de la aleatorización.
- APS (K1, K2): Estos bytes son empleados para la conmutación de protección automática (APS) dentro de los anillos que se forman en la red SDH. La indicación de defecto distante de seccion de multiplexación (MS-RDI) se utiliza para devolver al extremo de transmisión la indicación de que el extremo de recepción ha detectado un defecto de sección entrante o está recibiendo una señal de indicación de alarma de sección de multiplexaje(MS-AIS). La MS-RDI se genera insertando un código 110 en las posiciones 6,7 y 8 del octeto K2 antes de la aleatorización. Debe tenerse en cuenta que el bit 8 es el menos significativo y es el ultimo en ser transmitido, mientras que el bit 1 es el más significativo siendo el primero en ser transmitido[11].
- Canales de comunicación de datos DCC (D4-D12): Proporcionan mensajes de OAM&P entre equipo de SDH a un nivel de sección de multiplexación, pueden ser generados internamente o obtenidos desde una fuente exterior.
- Estado de sincronización(S1): Este byte es usado para transportar el estado de sincronización, permitiendo elegir la mejor fuente de reloj entre varias fuentes potenciales.
- STM-N MS-REI(M1): Es utilizado para informar a la terminal remota del número de errores encontrados al analizar los bytes de paridad B2. Por otro lado, cuando se tiene una trama STS-N, el byte M1 se define en la tercera

trama STS-1 que la conforma. El byte M0 es usado para funciones de error remoto a un nivel de multiplexaje, es importante aclarar que actualmente este byte también es utilizado para transportar información relativa a errores de paridad del byte B2 y esta nueva redefinición solo es aplicada cuando se tienen tramas STM-64[11].

- Canal de transmisión (E2): Es empleado como un canal de voz similar al canal originado por el byte E1.

Tara de trayecto(POH): El POH de un VC-4-Xc se sitúa en la primera columna de la estructura de 9 filas por X x 261 columnas del VC-4-Xc. El POH de VC-4 se sitúa en la primera columna de la estructura de 9 filas por 261 columnas del VC-4. El POH de VC-3 se sitúa en la primera columna de la estructura de 9 filas por 85 columnas del VC-3. El POH de VC-4Xc/VC-4/VC-3 consta de 9 octetos designados por J1, B3, C2, G1, F2, H4, F3, K3 y N1 los cuales se muestran en la figura 6. bytes de POH.

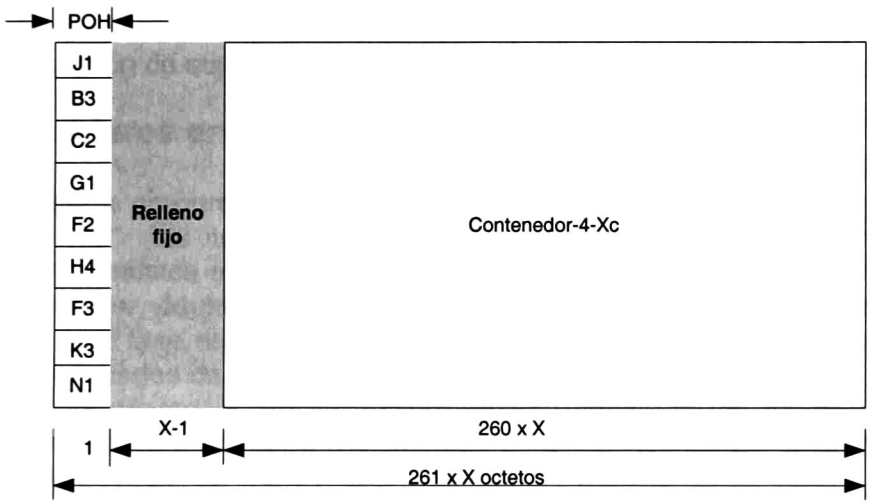

Figura 6. Estructura de un VC-4-Xc

A continuación se describen los octetos de POH.

- Octeto de rastro de ruta (J1): Es el primer octeto de contenedor virtual; su ubicación se indica mediante el puntero asociado a AU-n (n=3,4). Es empleado para que los equipos terminales mantengan una comunicación constante transmitiendo de manera repetitiva un identificador, y de tal modo que un terminal puede verificar la continuidad de su conexión.
- Octeto de trayectoria BIP-8(B3): Este campo es análogo a los campos BIP-8 empleado en sección de multiplexaje (B2) y en sección de regeneración(B1), donde la función es supervisión de errores de trayecto. Se calcula con base en todos los octetos del VC-4Xc/VC-4/VC-3 anterior, antes de la aleatorización.

- Octeto de señal (C2): Este octeto permite identificar que tipo de información conforma el contenedor virtual. Sus valores son determinados para cada tipo de carga útil.
- Estado de trayectoria (G1): Este campo indica el desempeño a nivel de trayectoria en un extremo, dando el desempeño y estatus de la conexión, por medio de dos señales de mantenimiento llamadas Remote Error Indicator in Path (REI-P), los bits 1-4, y Path Remote Defect Indicator (RDI-P), bits 5-7.
- Canal de usuario (F2, F3): Estos campos pueden ser usados por la red para proporcionar comunicación interna en la red.
- Indicador(H4): Este octeto proporciona un indicador de posición generalizado para cabidas útiles (por ejemplo, H4 puede utilizarse como un indicador de posición de multitrama para el VC-2/VC-1)
- Canal de conmutación de protección automática(APS):(K3) (bits 1 a 4): Estos bits se asignan para señalización de APS para protección a los niveles de trayecto de VC-4/3.
- Octeto de operación de red(N1): Este octeto se asigna para proporcionar una función de supervisión de conexión en cascada.

1.4.2 Punteros en SDH

Los punteros proporcionan un método para permitir una alineación flexible y dinámica de los tributarios virtuales dentro del SPE, entendiéndose por alineación dinámica el hecho de que los tributarios virtuales puedan flotar dentro del contenedor virtual. Así, el puntero es capaz de absorber las diferencias no solamente de fase de tributarios virtuales y de la tara de sección, sino también en las velocidades de trama.

Si hay una diferencia entre la velocidad de trama y la del tributario virtual el valor del puntero aumentará o disminuirá según la necesidad, acompañado por un octeto cuando se realiza una justificación positiva o un octeto de menos cuando se realiza una justificación negativa. Cuando la velocidad de la trama de la información es demasiado lenta o demasiado rápida con respecto a la trama SDH, la alineación del tributario virtual debe retroceder en más de un octeto entonces se realiza un redefinición de punteo, a dicho evento se le llama nuevo cambio de puntero (New Data Flag NDF), todo esto se lleva a cabo por medio de los octetos H1, H2 y H3.

1.4.3 Sincronización y temporización.

SDH funciona bajo el principio de un solo reloj para todo el sistema, para ello hace uso de la señal del sistema de posicionamiento global (GPS) la cual es usada por un conjunto de referencias primarias en toda la red (PRC), conocida como reloj de estrato 1 el cual tiene una precisión de ± 0.00001 partes por millón, en una jerarquía menor se encuentran los relojes de estrato 2 que son acertados en ± 0.016 partes por millón, los de estrato 3 son acertados en 4.6 partes por millón.
Todos los relojes utilizados en SDH deben ser de estrato 3 o mejores. Al utilizar relojes de estrato 3 permite a SDH multiplexaciónes de rangos de gigabit, así como también el equipo utilizado es más compacto y simple. No obstante las ventajas que se tienen al utilizar un solo reloj implican el problema de que el reloj sufre desfasamientos cuando es transmitido a través de fibra óptica para proporcionarlo a otros equipos SDH.

Para poder soportar SDH los problemas de desfase del reloj se emplean los punteros (H1/H2) junto con el byte H3, que en operación normal no es empleado, haciendo uso del concepto de justificación positiva y de la justificación negativa. La primera consiste en no insertar información en el byte que se encuentra enfrente del byte H3 por lo tanto el receptor no lo considerará.

La otra justificación radica en introducir información en el byte H3 y por lo tanto el receptor extraerá información de este byte. Para visualizar los valores que toman H1/H2 en cuando se realiza justificación se recurre a la siguiente figura 7:

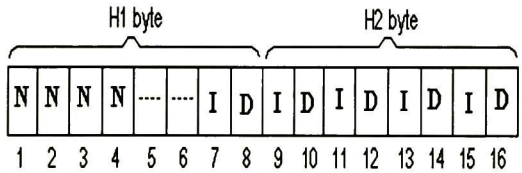

Figura 7. Codificación de Puntero de AU-n

Cuando opera la trama en forma normal los bits marcados con N toman el valor de 0110, los siguientes dos bits están indefinidos pero se recomienda que tengan 00, los restantes bits pueden tomar un valor de 0 a 782, indicando el número de bytes que existen entre el byte H3 y el comienzo del SPE (o byte J1).

Cuando se hace justificación positiva o negativa los bits marcados con N toman el valor de 0110. Para diferenciar una de la otra se hace por medio de los bits marcados con I (incremento) y D (decremento). Para la primera se invierten los bits I y el relleno positivo aparece en el byte que aparece

después del byte H3; para la segunda se invierten los bits D y se inserta información al byte H3.

Si las variaciones de frecuencia son de tal magnitud que no pueden ser absorbidas haciendo uso de una justificación positiva o negativa entonces se procede a realizar la operación de *nuevo puntero* (New Data Flag). Para ello los bits marcados con N toman el valor de 1001 además los bits I y D toman el valor del nuevo puntero (0-782) en la siguiente trama los bits N retornan al valor 0110 y el valor del nuevo puntero permanece.

2 Capítulo II.

Especificación de la cama de prueba, sección Generador.

Cuando se diseña una cama de prueba se tienen metodologías similares a las empleadas en un diseño. Una de las preguntas principales que se deben de establecer antes de comenzar con el diseño de la cama es si la cama se quiere que sea sintetizable o no; de la respuesta a dicha pregunta se pueden establecer arquitecturas completamente diferentes, las cuales pueden determinar en gran medida tanto el tiempo de desarrollo como el desempeño del dispositivo. Las camas de pruebas que se diseñan para ser sintetizables presentan problemas similares a los que se tienen cuando se diseña un circuito, como por ejemplo si se hace uso de un lenguaje como VHDL el diseño de la cama debe de cuidar el uso de instrucciones capaces de poder ser sintetizables, otro de los problemas sería que una vez sintetizada la cama de pruebas se debería realizar análisis de tiempo, simulación a un nivel de compuertas, verificación formal entre otras cuestiones. Sin embargo cuando se decide hacer una cama no sintetizable el diseñador puede hacer uso de todo el poder del lenguaje. Una cama sintetizable tiene la bondad de poder ser utilizada tanto en la verificación funcional como en pruebas de laboratorio, lo cual es muy conveniente para empresas relativamente pequeñas las cuales no cuenten con los recursos económicos suficientes para rentar o comprar estimuladores y analizadores de señales (tester). La cama diseñada en este trabajo es una cama no sintetizable, sin embargo el diseño que se tendrá de la cama permitiría en un futuro poderla transformar en una sintetizable si se requiriera.

Para comenzar el presente capítulo primero se describirá que se pretende con un documento de especificación.

Cuando se pretende sacar un nuevo producto a la venta se realizan una serie de estudios de mercado de lo que existe y falta, para posteriormente realizar una descripción del nuevo producto. Esta descripción podría estar carente de características técnicas e ingenieriles, lo cual no debe ser entendido como un documento carente de valor ya que es la base de partida en cualquier proyecto[3].

Como segundo documento se encuentra la especificación. En este documento se formaliza el entendimiento de los ingenieros que desarrollarán el producto, a partir de la lista de requerimientos. Resulta común que dentro de la especificación se anexen las características en forma ordenada, esto quiere decir que de acuerdo a un análisis de lo planeado se determinan las fases de desarrollo del diseño.

Es fundamental este documento para diseño y la verificación dado que de dicho escrito se desarrollarán otros más específicos técnicamente hablando, los cuales son la arquitectura y el plan de pruebas, que serán discutidos en posteriores capítulos.

2.1 Requerimientos

Diseñar una cama de prueba que sea capaz de ayudar a la verificación del circuito S19201 que es un circuito comercial el cual pertenece a la compañía Applied Micro Circuits Corporation(AMCC), éste maneja el estándar SDH. De acuerdo a la descripción pública que se tiene de este circuito[1], se propondrán una serie de requerimientos que permitan satisfacer las necesidades de prueba para llevar en un futuro las pruebas necesarias para verificarlo.
La cama debe:

- Ser modular e independiente (generador y analizador).
- Contener un puerto capaz de manejar un bus de 16 bits a 622.08 MHz.
- Contener cuatro puertos de 16 bits a 155 MHz.
- Ser capaz de soportar la programación independiente de los bytes de POH, MSOH, RSOH excepto los bytes B1, B2, B3 que son calculados en la trama.
- Permitir la entrada de reloj externo y generar uno interno.
- Permitir que el usuario programe la configuración.
- Poder ser interconectada con un diseño escrito en VHDL que tengan datos de 16 bits a una frecuencia de 622MHz.
- Generar tramas STM-64 , cuatro STM-16, y cualquier combinación valida de AU-U-16c, AU-A-4c, AU-4.
- El usuario debe poder programar los bytes de relleno en una trama.
- Poder calcular el bit interleaved parity(BIP).
- Ser capaz de soportar la sincronización de trama externa e interna para sincronizar la generación de tramas.
- Poder soportar movimientos de puntero, nuevo puntero, salto de puntero.
- Soportar la programación de movimientos de puntero.
- Ser capaz de insertar errores en la trama.
- Ser capaz de soportar cambios en cualquier byte de la trama.
- Ser capaz de insertar errores en A1, A2, H1,H2.
- Ser capaz de insertar errores en cualquier byte de la trama.
- Generar los bytes de carga útil como: constante, contador ascendente y descendente, PRBS.
- Permitir que los bytes J0 y J1 tengan traza. Esta puede ser de 64 ó 16 bytes dependiendo lo que el usuario determine; esta traza cumplirá con lo expuesto en el estándar de la ITU así como también que soporte mensajes que sean del tipo ISCCI.
- Poder activar o desactivar la aleatorización de las tramas generadas.

2.2 Descripción General

Tomando en cuenta aspectos de modularidad y mantenimiento se determinó partir el problema en diferentes módulos principales los cuales son:

1. manejador de generador
2. sistema de multiplexado
3. generador básico
4. control de multiplexado
5. calculador de BIP
6. inserción de errores

Cada uno de estos módulos tendrá un manejo de datos utilizando la unidad básica de información empleada en el estándar SDH: el byte[11]. En el diagrama 2 podemos observar la arquitectura general.

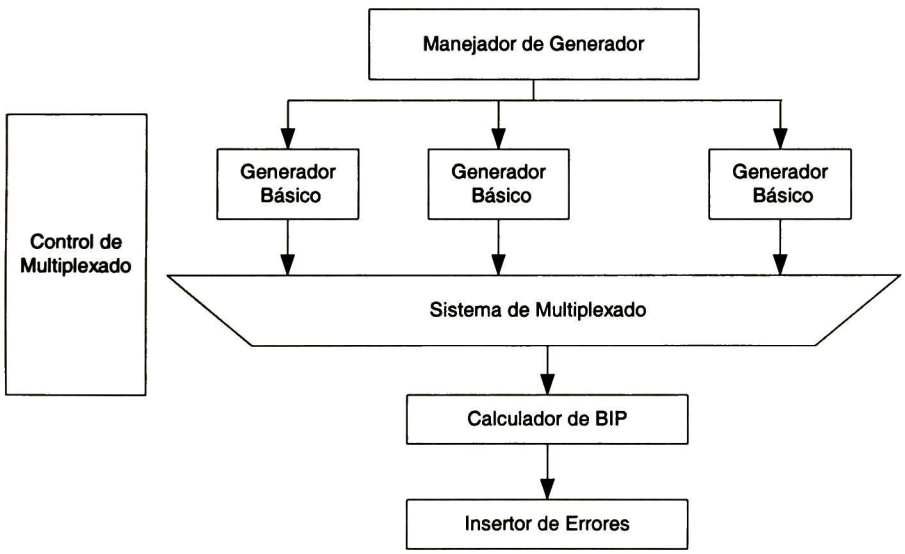

Diagrama 2. Arquitectura General de Multiplexado.

El dispositivo que tendrá la facilidad de ser el que permitirá configurar a la cama de prueba de acuerdo a lo deseado por el usuario es él modulo llamado manejador de generador. El sistema de multiplexado será el dispositivo que permita generar la formación de tramas de STM-X. Por otra parte un módulo más especializado es el generador básico el cual tendrá la tarea de poder generar AU-3, AU-4, AU-4-4c, AU-4-16c. El control de multiplexado es el que coordinará el multiplexado y la generación de las AU. Para calcular la paridad

par e insertarla en los bytes B2 y B1 se tendrá un módulo calculador de BIP. Cuando se quiera que una determinada trama contenga errores se contará con el módulo de inserción de errores.

2.2.1 Manejador de Generador

El manejador de generador será la interfaz entre el usuario y la cama. La forma de obtener información del usuario es por medio de un archivo de texto que deberá tener el nombre de SDH_frame.do; este archivo aceptará comandos muy particulares los cuales serán expuestos posteriormente. El manejador de generador lee el archivo y determina si existe algún error al utilizar algún comando o está empleando caracteres desconocidos. Cuando esto ocurre se le informa al usuario por un mensaje que le indica la línea donde se encontró el problema dentro del archivo SDH_frame, tales mensajes aparecerán en el monitor de la computadora, el usuario deberá corregir el archivo y nuevamente proceder a la ejecución del archivo. Hasta que el archivo se encuentre limpio de errores de sintaxis será posible que el generador comience a funcionar. La información que aparece en un comando es guardada en una base de datos, que estará siendo consultada constantemente por los diferentes módulos que conforman la cama de pruebas cuando así se requiera.

2.2.2 Sistema de multiplexado

El sistema de multiplexado estará conformado por diferentes multiplexores que permitirán construir tramas STM-64 o STM-16, a partir de AU generadas por el generador básico. Además transportará las diferentes señales habilitadoras a los 192 generadores básicos. Cuando se realice el multiplexado de tramas, se deberá respetar lo referente a movimientos de apuntador, nuevo puntero de las diferentes tramas que conforman la múltiplexación.

2.2.3 Control de multiplexado

El control de multiplexado será el encargado de coordinar al sistema de multiplexado, además determinará cuando comenzar con el proceso de aleatorización de bytes, cuando empezar el cálculo de los bytes B2 y B1 y cuando insertar el cálculo en la trama que se esta generando, determinará la inserción de errores en la trama ya formada.

2.2.4 Generador Básico

El generador básico será un módulo especializado en generar AU-3, AU-4, AU-4-4c, AU-4-16c; sin embargo no será capaz de formar tramas STM-X. Para poder abarcar cualquier generación de tramas STM-64 se empleará conjuntamente un sistema de multiplexado y 192 módulos básicos independientes entre sí. Además el módulo básico podrá realizar movimientos de apuntador positivo y negativo, y nuevo puntero en las AU que maneja. El tipo

de carga útil que transportará una trama podrá ser constante, contador ya sea ascendente o descendente, datos pseudoaleatorios; se podrá introducir mensajes de 64 ó 16 bits en el byte J0 y J1 en donde el tipo de mensaje podrá ser de acuerdo al estándar ITU. Cada módulo calculará la paridad del byte B3 y en este módulo se insertará. Este módulo debe ser capaz de aceptar una señal externa como fuente de sincronía o producir su propia sincronía.

2.2.5 Calculador de BIP

Este módulo es el que calcula la paridad par de los datos que salen por un puerto activo, el control de multiplexado le indica cuándo calcular paridad y cuándo insertar el cálculo en la trama. Cabe aclarar que el cálculo de los bytes B1,B2,B3 se realizan con el mismo algoritmo, la diferencia radica en cuales bytes de la trama son los que aplican para el cálculo de los diferentes bytes.

2.2.6 Aleatorizador de Bytes

Este módulo se encargará de hacer el proceso de aleatorización de bytes a todos los bytes de la trama que salgan por un puerto dado, exceptuando los bytes A1,A2,J0 y Z0.

2.2.7 Inserción de errores

Este módulo se encargará de insertar errores en la trama simulando que éstos son producidos en la fibra óptica, por tal motivo esta inserción se realizará después de haber hecho la aleatorización de bytes.

2.2.8 Interface de 16 Bits.

Para poder satisfacer el requerimiento de entregar a la salida del generador 16 bits, se quiere que exista un modulo encargado de entregar este formato. Dado que dentro del generador se realizará todas las operaciones usando el byte es necesario almacenar un byte generado y posteriormente concatenarlo con otro dato posteriormente generado, de esta forma entregar a la salida del generador los 16 bytes.

2.2.9 Manejo de Alarmas

La alarma AIS (Alarma Indication Signal) se generará introduciendo a la trama todos unos o todos ceros en todos los bytes. [12,13]

La alarma LOF (Loss of Frame) y SEF (Severed Error Frame) se produce introduciendo errores a los bytes A1 y A2.

La alarma AIS-P (Alarma Indication Signal of Path) se generará programando todos unos en los bytes de POH, Carga útil y punteros.

La alarma AIS-L (Alarma Indication Signal Of Line) a nivel de multiplexación se genera programando en todos unos los bytes de MSOH, POH y carga útil.

Las alarmas RDI y AIS se generan programando en los bits 6 a 8 los valores de "110" y "111" respectivamente en el byte K2.

Para generar un Remote Error Indication (REI) se genera programando el byte M2.

2.2.10 Comandos de la cama de prueba

La manera que tendrá cualquier usuario de la cama para poder realizar pruebas será por medio de un conjunto de comandos que a continuación se presentarán. Con dichos comandos el usuario tiene control absoluto sobre la generación de las tramas. El usuario debe respetar en todo momento la sintaxis de los comandos ya que en caso de que el usuario por error proporcione caracteres no permitidos la cama no podrá comenzar la generación de las tramas, presentándole en el monitor un mensaje de error en el cual se le indicará el número de línea donde se encuentra la instrucción desconocida.

2.2.10.1 HABILITACION_DE_TRAMA<X,Y,Z>

Este comando tiene como objetivo proporcionar información general sobre la estructura de trama que se querrá formar con la cama de pruebas en un puerto en particular. Esta instrucción tiene tres parámetros los cuales son: **X, Y, Z.**

X : Indica cual puerto se habilitará, dado que todos los puertos de la cama se encuentran deshabilitados. Los posibles valores que puede tomar son X=1... 64. Debe de tenerse cuidado en no repetir el puerto ya que se tomará él ultimo que se declaró. Además, la forma de utilización de los puertos es de forma ascendente y debe de considerarse que la trama de mayor orden que se puede formar en una STM-64.

Y : Indica el orden de la trama a ser generada en el puerto **X.** Sus valores permitidos son 64, 16, 4, 3.

Z : Indica si la trama a generar es concatenada o no. Sólo puede tomar los valores de 0 ó 1. Si es 1 indica que la trama a ser formada será concatenada. Cuando es 0 indica que la trama será no concatenada.

2.2.10.2 FUENTE_DE_SINCRONIA<X,Y>

Este comando determina a un puerto si la sincronía será tomada externamente o utilizará su propia fuente de sincronía. Si a un puerto no se le es indicado por medio de está instrucción la fuente de sincronía entonces el puerto empleará la sincronía interna. Esta instrucción tiene dos parámetros **X** y **Y**, los cuales se describen a continuación.

X : indica el puerto del cual se tomará la sincronía. Los posibles valores que puede tomar son números enteros que se encuentre entre 1 hasta 64.

Y: En 1 indica sincronía externa, y 0 indica sincronía interna, otro valor diferente a estos se considera como un error de sintaxis.

2.2.10.3 ALEATORIZACION <X,Y>

Esta instrucción indica si el puerto será aleatorizado o no. Cuenta con dos parámetros esta instrucción.

X: indica el puerto, los posibles valores que puede tomar es un número entero que se encuentra entre 1 hasta 64.

Y : Puede tomar los valores 0 ó 1. Cuando es 0 indica que no se quiere que las tramas que salgan de dicho puerto sean aleatorizadas. Si es 1 indicará que las tramas que salgan del puerto se les realizará el proceso de aleatorización.

2.2.10.4 ELEMENTO_DE_MULTIPLEXACION <X,W:Z>

Cuando se hace uso del comando HABILITACION_DE_TRAMA<X,Y,Z> solo se proporciona información de cual será el orden de la trama a formar, pero no se proporciona información de cuales serán los elementos de multiplexación que conforman la trama; por ejemplo se puede construir una trama STM-4, la cual pudiera estar conformada por dos tramas AU-4c, una trama AU-4 y una trama AU-3, pero también pudiera ser construida por cuatro AU-3. Este comando permite informar sobre este tipo de detalles. Consta de cuatro parámetros los cuales son:

X: Orden del elemento de multiplexación, sus posibles valores son: AU-3,AU-4, AU-4c, AU-4-16c, AU-4-64c

W:Z Dado que se tienen 64 generadores básicos que sirven para formar los elementos de multiplexación, es necesario indicar cuales de estos generadores básicos deben ser habilitados. W indica el primer generador básico y Z indica el ultimo generador básico que se requerirá para poder lograr lo que se programe en la opción X. Los posibles valores para W y Z es un número de 1 a 192 y W siempre es menor a Z.

2.2.10.5 ELEMENTO<X,Y,Z>

El usuario tiene dos opciones cuando quiera que se genere una trama, la primera es no definir en absoluto cualquier byte que conforma la trama y tener en cuenta que las tramas que se generarán tendrán valores por defecto. La otra opción es el definir desde un inicio un valor en particular para algún byte o todos los bytes de la trama, eso es decisión del usuario. Los valores por defecto que se tienen se pueden observar en la tabla1. Esta instrucción tiene tres parámetros: el primero define cuál byte será cambiado, el segundo determina a cuál generador básico pertenece el byte, y el ultimo redefine el nuevo valor que se quiere para el byte.

X:Puede tomar cualquiera los valores de A1, A2, J0, E1, F1, D1, D2, D3, K1, K2, D4, D5, D6, D7, D8, D9, D10, D11, D12, S1, M1, E2, J1, C2, G1, F2, H4, F3, K3, N1.

Y: Puede tomar cualquier valor entre 1 y 64.

Z: Puede ser cualquier número hexadecimal escrito en mayúsculas y que se encuentre en el rango de 00 a FF.

Nombre del Byte	Valor por omision (Hex.)
A1	F6
A2	28
J0	10
E1	E1
F1	F1
D1	D1
D2	D2
D3	D3
K1	31
K2	32
D4	D4
D5	D5
D6	D6
D7	D7
D8	D8
D9	D9
D10	D10
D11	D11
D12	D12
S1	S1
M1	22
E2	E2
J1	11
C2	C2
G1	G1
F2	F2
H4	H4
F3	23
K3	24
N1	25

Tabla1. Valores por defecto.

2.2.10.6 CARGA_UTIL<X,Y,Z>

Esta instrucción tiene la función de determinar qué tipo de carga útil se transportará en un elemento de multiplexación. Tiene tres parámetros que lo configuran, el primero define el generador básico, el segundo define que tipo de carga útil se generará, el tercero es el valor inicial con el cual comenzará el contador o la secuencia, dependiendo cual sea la programación de la opción Y.

X : Puede tomar cualquier valor entre 1 y 64.

Y: Puede tomar los siguientes valores: Constante, Contador_ascendente, Contador_descendente, Secuencia_seudo_aleatoria.

Z: Es un número hexadecimal escrito en mayúsculas y que se encuentra en el rango de 00 a FF Hex.

2.2.10.7 PUNTERO<W,X,Y,Z>

Esta instrucción tiene la función de poder definir el valor en particular de un elemento de multiplexación. Cuenta con cuatro parámetros donde el primero determina cual generador básico es al que se le definirá el puntero, el segundo parámetro determina el valor del puntero, y por tercer parámetro se tiene el que define el tipo de trama que se generará, el último parámetro determina el número de contenedor virtual al cual apuntará. Por defecto el valor de puntero es 522.

W : Puede tomar cualquier valor entre 1 y 64.

X: Puede tomar cualquier valor entre entero entre 0 y 782.

Y: Orden del elemento de multiplexación, sus posibles valores son: AU-3,AU-4, AU-4c, AU-4-16c, AU-4-64c.

Z: Puede tomar los valores de 1 a 3, este parámetro solo se toma en cuenta cuando la trama es AU-3. Dado que existirán tres contenedores virtuales en este tipo de trama los cuales necesitan un apuntador independiente cada uno.

2.2.10.8 MANIPULACIÓN_DE_PUNTERO<X,Y,Z>

Esta instrucción permite manejar los punteros de un elemento de multiplexación, de acuerdo a las necesidades del usuario. Con esta instrucción se pueden realizar movimientos de puntero positivos y negativos, nuevo puntero. El usuario puede determinar en cuales tramas se quiere realizar operaciones en el puntero, pudiéndose hacer cualquier operación que él desee, cuando ya decida el usuario que la cama continúe de forma normal en el manejo del puntero entonces este deberá indicárselo de forma explícita haciendo uso de la opción puntero normal de esta misma instrucción. El primer parámetro determina en cual elemento de multiplexación se quiere realizar una operación de puntero, el segundo

determina en cual trama se llevará a efecto dicha operación en el puntero, la tercera indica el tipo de operación que se le realizará al puntero.

X: Puede tomar cualquier valor entre 1 y 64.

Y: Puede tomar cualquier valor entre 1 y 500.

Z: Puede tomar los valores de Incremento, Decremento, Nuevo_Puntero, Puntero_Normal. Con Puntero_Normal la cama recobra el control del manejo del puntero.

2.2.10.9 CAMBIO<X,Y,Z>

Esta instrucción permite redefinir a un particular elemento de multiplexación un valor de los bytes de la tara que conforman la trama. Los cambios son referenciados a una trama en particular, en las subsecuentes tramas permanece el cambio realizado si es que no existe otro cambio en una posterior trama. El primer parámetro indica cual será el elemento de multiplexación que se le realizará una redefinición en alguno de sus bytes, el segundo determina el número de trama en donde tendrá efecto el cambio, y el tercero define cual byte en particular se quiere cambiar.

X: Puede tomar cualquier valor entre 1 y 64.

Y: Puede tomar cualquier valor entre 1 y 500.

Z: Puede tomar cualquiera los valores de A1, A2, J0, E1, F1, D1, D2, D3, K1, K2, D4, D5, D6, D7, D8, D9, D10, D11, D12, S1, M1, E2, J1, C2, G1, F2, H4, F3, K3, N1.

2.2.10.10 INSERTAR _ERROR<V,W,X,Y,Z>

Este comando permite al usuario insertar un error en cualquier puerto activo. Consta de cinco parámetros: el primer parámetro indica a cual puerto se le quiere insertar algún error, el segundo define en cual trama se insertara el error, el tercero y el cuarto determinan la columna y la fila respectivamente, el último define la máscara con la cual se realizará el error. Es importante aclarar que la fila y la columna deben de estar referidas a la estructura de la trama que se está generando en el puerto y no con respecto a alguno de sus elementos de multiplexación.

V: Los posibles valores que puede tomar cualquier valor entre 1 y 64.

W: Puede tomar cualquier valor entre 1 y 500.

X : Puede tomar un número entero entre 1 y 9

Y: Puede tomar un número entero entre 1 y 270 x Orden de la trama.

Z: Puede ser cualquier número hexadecimal escrito en mayúsculas y que se encuentre en el rango de 00 a FF Hex.

2.2.10.11 ALARMA_AIS<V,W,X>

Esta instrucción permite producir una alarma AIS en un puerto determinado. Consta de tres parámetros, el primero indica el número de puerto en donde se podrá hará presente la alarma AIS, el segundo parámetro define la trama en donde se dará inicio la alarma AIS y el ultimó parámetro define la trama donde se suspenderá la emisión de la alarma. Cabe aclarar que W debe ser menor a X.

V: Los posibles valores que puede tomar cualquier valor entre 1 y 64.

W: Puede tomar cualquier valor entre 1 y 500.

X: Puede tomar cualquier valor entre 2 y 500.

2.2.10.12 ALARMA_LOS<V,W,X>

Esta instrucción permite producir una alarma LOS en un puerto determinado. Consta de tres parámetros: el primero indica el número de puerto en donde se hará presente la alarma LOS, el segundo parámetro define la trama en donde se dará inicio la alarma LOS y el último parámetro define la trama donde se suspenderá la emisión de la alarma. Cabe aclarar que W debe ser menor que X .

V: Los posibles valores que puede tomar son cualesquiera entre 1 y 64.

W: Puede tomar cualquier valor entre 1 y 500.

X: Puede tomar cualquier valor entre 2 y 500.

2.2.10.13 ALARMA_AISR<V,W,X>

Esta instrucción permite producir una alarma AIS-R en un puerto determinado. Consta de tres parámetros, el primero indica el número de puerto en donde se hará presente la alarma AIS-R, el segundo parámetro define la trama en donde se dará inicio la alarma AIS-R y el último parámetro define la trama donde se suspenderá la emisión de la alarma. Cabe aclarar que W debe ser menor que X .

V: Los posibles valores que puede tomar son cualesquiera entre 1 y 64.

W: Puede tomar cualquier valor entre 1 y 500.

X: Puede tomar cualquier valor entre 2 y 500.

2.2.10.14 ALARMA_AISL<V,W,X>

Esta instrucción permite producir una alarma AISL en un puerto determinado. Consta de tres parámetros: el primero indica el número de puerto en donde se hará presente la alarma AIS-L, el segundo parámetro define la trama en donde se dará inicio la alarma AIS-L y el último parámetro define la trama donde se suspenderá la emisión de la alarma. Cabe aclarar que W debe ser menor que X

V: Los posibles valores que puede tomar son cualquiera entre 1 y 64.

W: Puede tomar cualquier valor entre 1 y 500.

X: Puede tomar cualquier valor entre 2 y 500.

2.2.10.15 HUELLA _J0<V,W>

Este comando tiene la finalidad de poder transmitir un mensaje ASCII de 15 bytes en el byte J0 y el cálculo de CRC-7 del mensaje transmitido, consta de dos argumentos los cuales son número de puerto y el mensaje.

V: Los posibles valores que puede tomar cualquier valor entre 1 y 64.

W: Mensaje de 15 caracteres del codigo ASCII.

2.2.10.16 HUELLA _J1<V,W>

Este comando tiene la finalidad de poder transmitir un mensaje ASCII de 15 bytes en el byte J1 y el cálculo de CRC-7 del mensaje transmitido, consta de dos argumentos los cuales son número de puerto y el mensaje.

V: Los posibles valores que puede tomar cualquier valor entre 1 y 64.

W: Mensaje de 15 caracteres del codigo ASCII.

3 Capítulo III.

Arquitectura de la cama de pruebas, sección Generador

En el presente capítulo se describirá la arquitectura de la cama de pruebas sección generador. La arquitectura general de la cama es mostrada en la figura 8. Como primer elemento se encuentra un archivo de configuración de la cama de pruebas, el cual es escrito por el usuario; este archivo contiene un conjunto de instrucciones que deberán ser leídas por el Manejador del Generador, el cual después de leer y validar que no existió ningún error de sintaxis en el archivo de entrada se dispone a guardar la información en diferentes bases de datos que los demás módulos del generador emplearán para generar tramas.

El generador básico es un módulo que lee las bases de datos que contienen la información referente a las tramas que serán generadas, y con esta información se encarga de conformar la trama que le sea pedida por el usuario. Se tendrán 64 generadores básicos los cuales podrán ser habilitados o no. Dado que el generador básico sólo es capaz de generar tramas STM-1 o STM-Nc y no es capaz por si solo de generar tramas STM-N entonces se tiene un sistema llamado Sistema de Multiplexado el cual será el encargado de conformar tramas STM-N. Para poder cumplir con dicho objetivo se hace uso de los 64 generadores básicos disponibles y dependiendo de que tipo de trama quiere conformar se habilita y deshabilita los generadores básicos, para así poder realizar el multiplexado necesario en la generación de tramas de orden mayor.
Cuando se tiene conformada la trama deseada se procede a la inserción de errores; los errores insertados pueden ser en cualquier lugar de la trama, esta operación la realiza el modulo llamado Insertor de errores.

Posteriormente si el usuario determina que la trama antes de ser transmitida se le aplique un proceso de aleatorización a los bytes de la trama (scrambler) entonces tal proceso lo lleva a cabo el módulo llamado Aleatorizador.
Independientemente si los datos fueron aleatorizados o no se procede con el cálculo e inserción del byte B1, tal cálculo lo realiza el módulo llamado Calculador de BIP8.

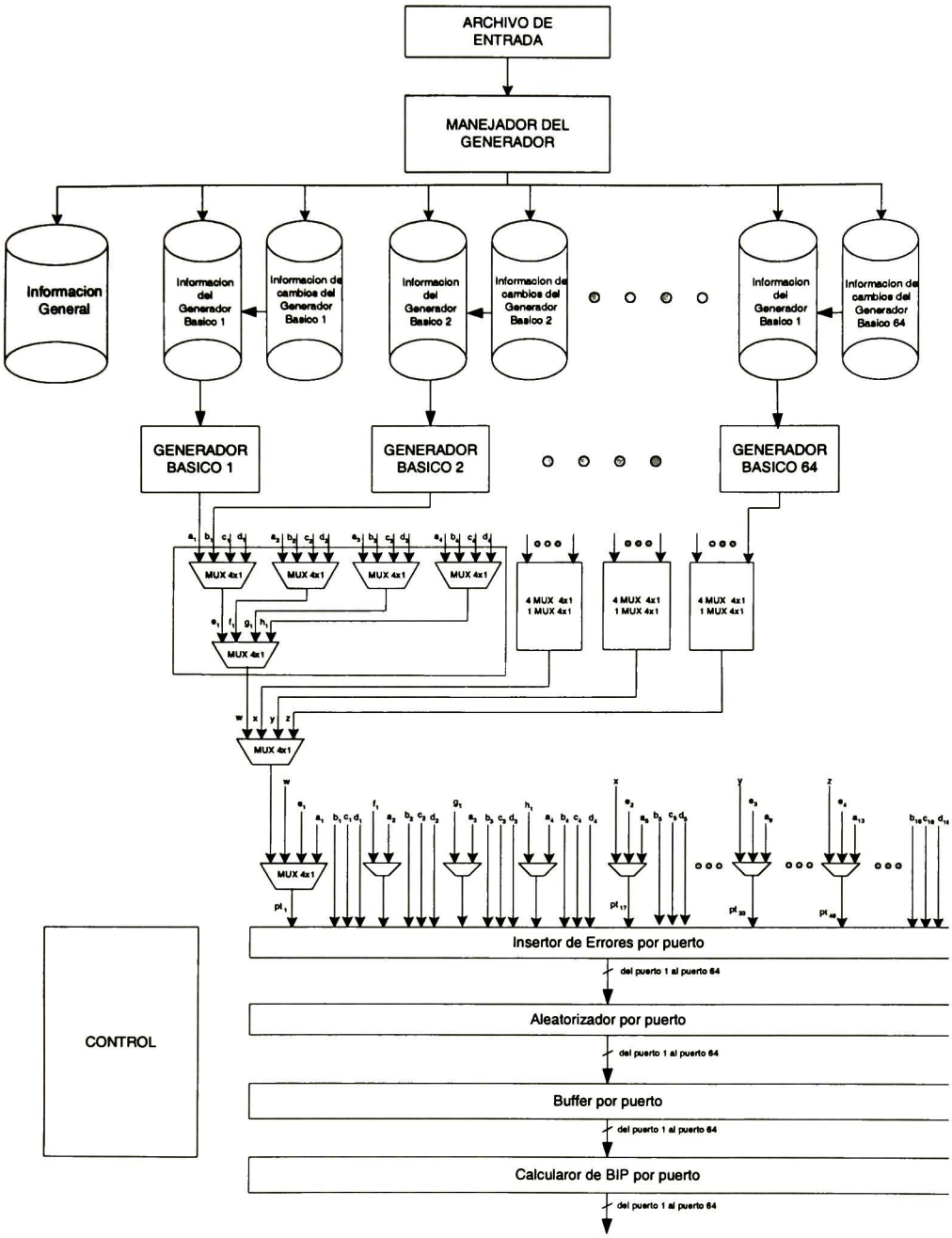

Figura 8. Arquitectura general de la Cama de Pruebas(Generador)

3.1 Generador Básico

Este módulo genera tramas STM-1 o STM-Nc donde N =1, 4, 16, 64; la letra c indica que es una trama concatenada. En la figura9 se muestra las entradas y salidas de este módulo así como también su descripción. La señal reset es una señal asíncrona que cuando es puesta a cero se suspende toda actividad del módulo y se reinicializa el módulo a sus valores de defecto. Cuando la señal de reset es puesta a uno el módulo puede comenzar a generar tramas. Otra señal que tiene importancia en jerarquía de funcionamiento es la señal de enable, si tal señal se encuentra en bajo el módulo se pone en estado de "espera" hasta que nuevamente sea puesta en alto dicha señal proseguirá con la generación de una trama. La entrada que determina que dentro de este módulo todos los procesos se lleven a cabo de forma síncrona es la señal llamada clk. El módulo cuenta con una señal de sincronía llamada A1_in con dicha señal se indica el inicio de una trama, si una trama aún no termina y esta señal se hace presente inmediatamente se tira la que se estaba formando para comenzar con una nueva la cual será alineada con este pulso. Este pulso siempre indica la presencia del byte A1 en la trama SDH.

La señal data es una salida que contiene los datos que conforman la trama, los datos salen en forma de byte. La señal J1 indica la presencia del byte J1, cuando esta puesta en alto. Otra salida es la señal A1_out, dicha señal es parecida a la señal A1_in salvo que presenta un retraso.

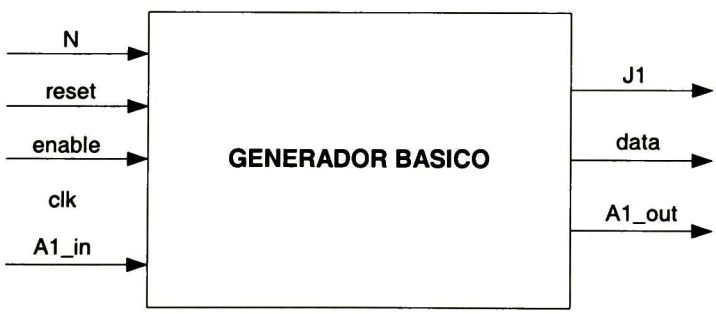

Figura 9 Entradas/Salidas del Generador Básico

Arquitectura interna del Generador Básico.

En la figura10 se muestran los diferentes módulos que conforman el generador básico. El generador básico lo conforman cinco generadores, un generador de puntero, un sistema de multiplexado, un control de multiplexado, control de fila y columna, alarmas, y un módulo encargado del cálculo de B2 y B3.

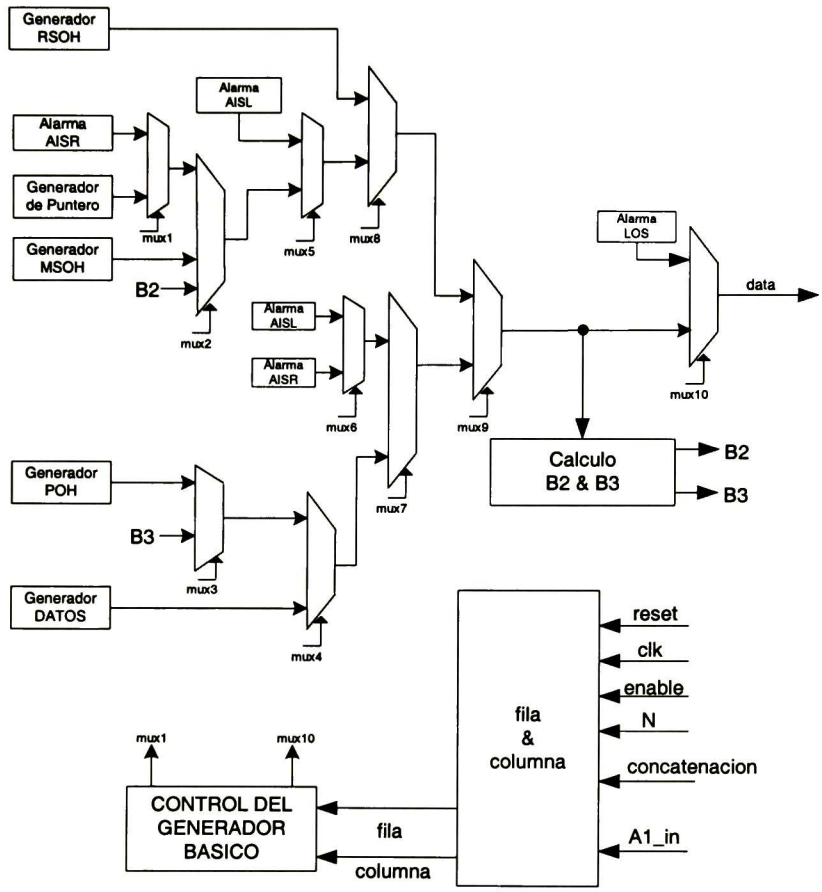

Figura 10 Arquitectura del Generador Básico.

3.1.1 Generador MSOH

Este generador proporciona todos los bytes que pertenecen al MSOH en una trama SDH. Sus entradas y salidas se describen más adelante. Los datos de MSOH generados serán siempre iguales para todas las tramas a menos que el usuario determine en cual trama se cambie algún byte. En tal caso el Generador MSOH comenzará a sacar el byte requerido sucesivamente hasta que nuevamente el usuario determine algún cambio.

Entradas
N : indica el orden de la trama, puede tener los valores de 1, 4, 16, 64. Si N=1 nos indicará que se quiere generar una trama STM-1, para los demás valores implicará que se quiere generar una trama concatenada.
Index: Si la trama en formación es concatenada entonces este número variará de la siguiente manera: cuando N=4 entonces index se incrementará de 1 hasta 12, cuando N=16 entonces index se incrementará de 1 hasta 48, cuando N=64 entonces index se incrementara de 1 hasta 192.
Num_gen: Es un número de referencia que diferencia un generador básico de otro.
Salidas:
A1: Entrega los bytes A1 de una trama SDH, el valor por defecto es F6 hex.

A2: Entrega los bytes byte A2 de una trama SDH, el valor por defecto es 28 hex.

J0: Entrega el byte J0 de una trama SDH, el valor por defecto es 10 hex, este byte puede ser generado para que transmita una traza. La traza cumplirá con lo especificado en un el estándar T50 de la ITU o con caracteres ASCII, la longitud de la traza puede ser de 1 byte,15 bytes o 64 bytes.

B1: Entrega el valor por defecto de B1 hex. Se debe tener presente que el cálculo de la paridad par se hace fuera del generador básico, además cuando se quiera formar una trama concatenada solamente existe un byte B1 y los bytes que se encuentran entre B1 y el byte E1 son indefinidos en el estándar y por tal motivo es posible que tomen cualquier valor, en particular este generador básico les asignará el valor de B1hex. El usuario estará en la libertad de modificar el valor por defecto de estos bytes indefinidos.

E1: Entrega el byte E1 de una trama SDH, el valor por defecto es E1 hex. Cuando se está generando una trama concatenada esta misma salida generarán los bytes indefinidos que se encuentran entre el byte E1 y el byte F1, el valor por defecto para estos bytes será E1 hex.

F1: Entrega el byte F1 de una trama SDH, el valor por defecto es F1 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte F1 y la carga útil, el valor por defecto para estos bytes será F1 hex.

D1: Entrega el byte D1 de una trama SDH, el valor por defecto es D1 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte D1 y el byte D2, el valor por defecto para estos bytes será D1 hex.

D2: Entrega el byte D2 de una trama SDH, el valor por defecto es D2 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte D2 y el byte D3, el valor por defecto para estos bytes será D2 hex.

D3: Entrega el byte D3 de una trama SDH, el valor por defecto es D3 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte D3 y la carga útil, el valor por defecto para estos bytes será D3 hex.

3.1.2 Generador RSOH

Este generador proporciona todos los bytes que pertenecen al RSOH en una trama SDH. Sus entradas y salidas se describen abajo. Los datos de RSOH generados por este módulo serán siempre iguales para todas las tramas a menos que el usuario determine que en determinada trama se cambie algún byte en la trama y este byte pertenezca a RSOH. En tal caso el Generador RSOH comenzará a sacar el nuevo byte requerido hasta que nuevamente el usuario determine algún cambio.

Entradas
N : indica el orden de la trama, puede tener los valores de 1, 4, 16, 64. Si N=1 nos indicará que se quiere generar una trama STM-1, para los demás valores esta implícito que se quiere generar una trama concatenada.

Index: Si la trama en formación es concatenada entonces este número variará de la siguiente manera: cuando N=4 entonces index se incrementará de 1 hasta 12, cuando N=16 entonces index se incrementará de 1 hasta 48, cuando N=64 entonces index se incrementara de 1 hasta 192.

Num_gen: Es un número de referencia que diferencia un generador básico de otro.
Salidas
K1: Entrega el byte K1 de una trama SDH, el valor por defecto es 31 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte K1 y el byte K2, el valor por defecto para estos bytes será 31 hex.

K2: Entrega el byte K2 de una trama SDH, el valor por defecto es 32 hex. Cuando se está generando una trama concatenada esta misma salida generará

los bytes indefinidos que se encuentran entre el byte K2 y la carga útil, el valor por defecto para estos bytes será 32 hex.

D4: Entrega el byte D4 de una trama SDH, el valor por defecto es D4 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte D4 y el byte D5, el valor por defecto para estos bytes será D4 hex.

D5: Entrega el byte D5 de una trama SDH, el valor por defecto es D5 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte D5 y el byte D6, el valor por defecto para estos bytes será D5 hex.

D6: Entrega el byte D6 de una trama SDH, el valor por defecto es D6 hex. Cuando se está generando una trama concatenada esta misma salida generará los bytes indefinidos que se encuentran entre el byte D6 y la carga útil, el valor por defecto para estos bytes será D6 hex.

D7: Entrega el byte D7 de una trama SDH, el valor por defecto es D7 hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte D7 y el byte D8, el valor por defecto para estos bytes será D7 hex.

D8: Entrega el byte D8 de una trama SDH, el valor por defecto es D8 hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte D8 y el byte D9, el valor por defecto para estos bytes será D8 hex.

D9: Entrega el byte D9 de una trama SDH, el valor por defecto es D9 hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte D9 y la carga útil, el valor por defecto para estos bytes será D9 hex.

D10: Entrega el byte D10 de una trama SDH, el valor por defecto es DA hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte D10 y el byte D11, el valor por defecto para estos bytes será DA hex.

D11: Entrega el byte D11 de una trama SDH, el valor por defecto es DB hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte D11 y el byte D12, el valor por defecto para estos bytes será DB hex.

D12: Entrega el byte D12 de una trama SDH, el valor por defecto es DC hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte D12 y la carga útil, el valor por defecto para estos bytes será DC hex.

S1: Entrega el byte S1 de una trama SDH, el valor por defecto es 51 hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes Z1 los cuales que se encuentran entre el byte S1 y el byte Z2, el valor por defecto para estos bytes será 51 hex.

M1: Cuando se está generando una trama se entrega el byte M1 cuando index es igual a 3. El valor por defecto para este byte será 22 hex. Cuando se index sea otro valor se entregará el mismo byte.

E2: Entrega el byte E2 de una trama SDH, el valor por defecto es E2 hex. Cuando se está generando una trama concatenada esta misma salida generara los bytes indefinidos que se encuentran entre el byte E2 y la carga útil, el valor por defecto para estos bytes será E2 hex.

3.1.3 Generador POH

Este generador proporciona todos los bytes que pertenecen al POH en una trama SONET. Sus entradas y salidas se describen abajo. Los datos de POH generados por este módulo serán siempre iguales para todas las tramas a menos que el usuario determine que en determinada trama se cambie algún byte de POH, cuando se realice tal cambio las subsecuentes tramas generadas tendrá el nuevo byte, siempre y cuando no exista en otra indicación de algún otro cambio para ese byte a generar por el POH.
Entradas
N : indica el orden de la trama, puede tener los valores de 1, 4, 16, 64. Si N=1 nos indicará que se quiere generar una trama STM-1, para los demás valores implica que se quiere generar una trama concatenada.

Index: Si la trama en formación es concatenada entonces este número variará de la siguiente manera: cuando N=4 entonces index sé incrementará de 1 hasta 12, cuando N=16 entonces index se incrementará de 1 hasta 48, cuando N=64 entonces index se incrementara de 1 hasta 192.

Num_gen: Es un número de referencia que diferencia un generador básico de otro.

Salidas

J1: Entrega el byte J1 de una trama SDH, el valor por defecto es 11 hex, este byte puede ser generado para que se transmita una traza. La traza cumplirá con

lo especificado en el estándar T50 de la ITU o con caracteres ASCII, la longitud de la traza puede ser de 1 byte, 15 byte o 64 byte. Cuando sé esta generando una trama concatenada esta misma salida generará, a demás, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es 11 hex.

C2: Entrega el byte C2 de una trama SDH, el valor por defecto es C2 hex. Cuando sé esta generando una trama concatenada esta misma salida generará, a demás, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es C2 hex.

G1: Entrega el byte G2 de una trama SDH, el valor por defecto es 62 hex. Cuando sé está generando una trama concatenada esta misma salida generará, además, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es 62 hex.

F2: Entrega el byte F2 de una trama SDH, el valor por defecto es F2 hex. Cuando se está generando una trama concatenada esta misma salida generará, además, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es F2 hex.

H4: Entrega el byte H4 de una trama SDH, el valor por defecto es 44 hex. Cuando se está generando una trama concatenada esta misma salida generará, además, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es 44 hex.

F3: Entrega el byte Z3 de una trama SDH, el valor por defecto es 23 hex. Cuando se está generando una trama concatenada esta misma salida generará, además, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es 23 hex.

K3: Entrega el byte Z4 de una trama SDH, el valor por defecto es 24 hex. Cuando se está generando una trama concatenada esta misma salida generará, además, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es 24 hex.

N1: Entrega el byte 25 de una trama SONET, el valor por defecto es 25 hex. Cuando se está generando una trama concatenada esta misma salida generará, además, los bytes de relleno fijo los cuales cumplirán la regla [(N*3)/3]-1, donde N es el orden de la trama. El valor por defecto de estos bytes es 25 hex.

3.1.4 Generador Carga Útil:

Este generador es capaz de producir los datos que se encuentran en el contenedor virtual de una trama SDH, puede producir cuatro tipos de datos los cuales son: dato constante, contador ascendente, contador descendente, datos pseudoaleatorios. El usuario puede determinar cuál tipo de dato quiere que contenga el contenedor virtual.
Entradas
N : indica el orden de la trama, puede tener los valores de 1, 4, 16, 64. Si N=1 nos indicará que se quiere generar una trama STM-1, para los demás valores implica que se quiere generar una trama concatenada.

Index: Si la trama en formación es concatenada entonces este número variará de la siguiente manera: cuando N=4 entonces index se incrementará de 1 hasta 12, cuando N=16 entonces index se incrementará de 1 hasta 48, cuando N=64 entonces index se incrementara de 1 hasta 192.

Num_gen: Es un número de referencia que diferencia un generador básico de otro.
Columna: Indica la columna respecto a la estructura SDH. Puede variar de 0 a 90x3xN.
Fila: Indica la fila respecto a la estructura SDH. Puede variar de 0 a 9.

Salida:
Data: Entrega el dato que se transportara en la carga útil de la trama SDH.

3.1.5 Generador de Puntero

Este módulo tiene la tarea de proporcionar la generación de puntero de acuerdo a los requerimientos que el usuario necesite, siendo capaz de hacer las siguientes operaciones: incremento de puntero, decremento de puntero, nuevo puntero, puntero normal.
Entradas
N : indica el orden de la trama, puede tener los valores de 1, 4, 16, 64. Si N=1 nos indicara que se quiere generar una trama STM-1, para los demás valores implica que se quiere generar una trama concatenada.

Index: Si la trama en formación es concatenada entonces este número variará de la siguiente manera: cuando N=4 entonces index se incrementará de 1 hasta 12, cuando N=16 entonces index se incrementará de 1 hasta 48, cuando N=64 entonces index se incrementara de 1 hasta 192.

Num_gen: Es un número de referencia que diferencia un generador básico de otro.

Salidas
H1: Entrega el byte H1 de una trama SDH, el valor por defecto es 95 hex.

H2: Entrega el byte H2 de una trama SDH, el valor por defecto es 22 hex.

3.1.6 Alarmas

Cada generador básico es capaz de producir las siguientes alarmas: AIS-R (Alarm Indication Signal in rute), AIS-L (Alarm Indication Signal in Line), LOS (Loos of Signal). Para producir una alarma AIS-R es necesario que tanto el valor de los punteros y que la carga útil sean FF Hex, los multiplexores mix1,mux6 y mux7 se combinan para permitir la salida de FF Hex que se tiene en el módulo Alarma AISR. Para producir una alarma AISL es necesario que toda la trama excepto la tara de sección de multiplexación sea igual FF Hex. Los multiplexores mix5, mux6 y mux7 se combinan para permitir la salida de FF Hex que se tiene en el módulo Alarma AISL. Cuando se quiere producir una alarma LOS es necesario que toda la trama sea FF Hex, para lograr ésto sólo es necesario que el multiplexor mux 10 permita la salida de FF Hex. del módulo Alarma LOS.
Cuando se quiera producir una alarma Remote Defect Indication (RDI) y Alarm Indication Signal (AIS) el usuario deberá programar el byte K2 poniendo en los bits 6 a 8 "110" o "111" respectivamente. Por último cuando se quiera simular un Remote Error Indication (REI) el usuario deberá programar el byte M1.

3.1.7 Cálculo de BIP

Existen tres diferentes cálculos de paridad con entrelazado de bits (bit interleaved parity BIP) en una trama SDH, en el generador básico no se calcula el BIP del byte B1. En el módulo básico sólo se calcula la paridad de los bytes B2 y el byte B3. Estos cálculos se realzan a este nivel dado que el cálculo de estos bytes no cambiara aunque en otra etapa de la cama de pruebas se quiera una trama de orden superior.
Entradas
N : indica el orden de la trama, puede tener los valores de 1, 4, 16, 64. Si N=1 nos indicará que se quiere generar una trama STM-1, para los demás valores implica que se quiere generar una trama concatenada.

Index: Si la trama en formación es concatenada entonces este número variará de la siguiente manera: cuando N=4 entonces index se incrementará de 1 hasta 12, cuando N=16 entonces index se incrementará de 1 hasta 48, cuando N=64 entonces index se incrementara de 1 hasta 192.

Num_gen: Es un número de referencia que diferencia un generador básico de otro.
Columna: Indica la columna respecto a la estructura SDH. Puede variar de 0 a 90x3xN.

Fila: Indica la fila respecto a la estructura SDH. Puede variar de 0 a 9.

Salidas
BIP_B2: Entrega el cálculo del byte B2 calculado de acuerdo a como se especifica en el estándar de la ITU G707.

BIP_B3: Entrega el cálculo del byte B3 calculado de acuerdo a como se especifica en el estándar de ITU G707

3.1.8 Estructura de Multiplexado Básico

Este módulo recibe todos los flujos de información que entregan los módulos: Generador MSOH, Generador RSOH, Generador de POH, Generador de Carga Útil, Generador de puntero. Con todos estos flujos de información realiza el multiplexado ya sea de una trama STM-1 o bien de una trama STM-Nc.

3.1.9 Control del Generador Básico

Este módulo se encarga de habilitar a los diferentes generadores, controlar las dimensiones de las tramas que se están generando, de acuerdo al tipo de trama que se está generando realiza el multiplexado pedido, cuando se realicen operaciones apuntador controlar la multiplexación y adecuada generación de bytes de acuerdo a la multiplexación.

3.1.10 Estructura de Multiplexado

A la estructura de multiplexado le llegan como fuentes de entrada todas las tramas provenientes de los diferentes generadores básicos; estas tramas pueden ser STM-1 o STM-Nc. Y como salida tendrá 64 puertos los cuales serán independientes entre sí, pero con las siguientes restricciones: si se quiere utilizar los 192 puertos entonces por cada puerto solo podrá tener como salida tramas STS-1. Cuando se pretenda tener una trama STS-192 este tipo de trama sólo podrá ser generada en el puerto 1. En general los puertos generarán apegándose a las reglas de multiplexado dadas en el estándar.

3.1.11 El insertador de Errores

Una vez que se terminó el proceso de multiplexación en el módulo Estructura de Multiplexado se procede con el proceso de insertar errores a la trama, si es que el usuario quiere que alguna trama en particular saliente de algún puerto contenga uno o varios valores diferentes a los que se estaban transportando. Esta inserción de errores se lleva a efecto considerando el orden de la trama en formación del puerto.

3.1.12 Aleatorizador

Una vez que se ha insertado los errores se procede con el proceso de aleatorización de bytes, esto se lleva a efecto utilizando el polinomio $1+X^6+X^7$ que es el que se establece en el estándar de la ITU-G707, este proceso de aleatorización de byte puede ser habilitado o deshabilitado dependiendo como el usuario lo determine. Se realiza de forma independiente en cada puerto.

3.1.13 Calculo de BIP-B1

El cálculo del byte B1 se lleva a efecto por cada puerto y se realiza sobre todos los bytes de la trama después de haber sido pasados por el proceso de aleatorización de bytes.

4 Capítulo IV

Verificación Funcional de la cama de pruebas, sección Generador

En este capítulo se describirá la verificación del generador, el cual fue básicamente probado utilizando las técnicas de la caja blanca y la de caja gris. La razón por la cual se utilizaron ambas técnicas, es porque en el presente trabajo se hizo tanto el diseño como la verificación del generador de tramas SDH, haciendo uso de ambas técnicas se permite encontrar, delimitar y solucionar problemas a un nivel muy básico de una forma clara y rápida[14].

El criterio que se usó para determinar cuál técnica utilizar en determinado momento consistió en probar como cajas blancas a todos los módulos que realizaban una función especifica y no estaban aún integrados con otros. Mientras que se verificaron como cajas grises todos los submódulos y el sistema integrado totalmente. Las pruebas del generador básico consistirán en probar cada uno de los generadores, la estructura de multiplexado, calculador de B2 y B3 como cajas blancas. Mientras que la integración del generador se realizara como caja gris.

4.1 Prueba GB_MSOH_1:

Objetivo: Se verificará que el generador básico genera sus valores por defecto y que es capaz de generar diferentes datos de acuerdo a lo que el usuario determine en el MSOH.

Excitación:

1. Configurar al módulo para simular que se está generando una trama STM-1. N=1, Num_gen=1.
2. Configurar al módulo para simular que se está generando una trama STM-1.N=1, Num_gen=34. Configurar para que el generador 34 genere el valor por defecto más uno.

Respuesta:

1. En los pines de salida del módulo deberán aparecer los siguientes datos:

Nombre del pin	Valor esperado (Hex.)
A1	F6
A2	28
J0	10
Z0	20
B1	B1
E1	E1
F1	F1
D1	D1
D2	D2
D3	D3

1 Debe de responder semejante al inciso anterior con la diferencia de que el valor esperado es igual al mostrado abajo.

Nombre del pin	Valor esperado (Hex.)
A1	F7
A2	29
J0	11
Z0	21
B1	B2
E1	E2
F1	F2
D1	D2
D2	D3
D3	D4

4.2 Prueba GB_MSOH_2:

Objetivo: Comprobar que el generador básico es capaz de generar los datos de MSOH, en una trama STM –Nc.

Excitación:

1. Configurar al módulo para simular que se está generando una trama STM-1 y que los bytes que genere de MSOH deben ser igual a sus diferentes valor por defecto
2. Configurar al módulo para simular que sé está generando una trama STM-4c y que los bytes que genere de MSOH deben ser iguales a su valor por defecto.
3. Configurar al módulo para simular que sé está generando una trama STM-16c y que los bytes que genere de MSOH deben ser iguales a su valor por defecto.
4. Configurar al módulo para simular que se está generando una trama STS-64c y que los bytes que genere de MSOH deben ser iguales a su valor por defecto.

Respuesta

Verificar que las tramas generadas correspondan con la configuración y valores por defecto.

4.3 Prueba GB_RSOH_1:

Objetivo: Se verificará que el generador básico genera sus valores por defecto y que es capas de generar diferentes datos de acuerdo a lo que el usuario determine.

Excitación:

1. Configurar al moduló para simular que se está generando una trama STM-1. Num_gen=1.
2. Configurar al moduló para simular que se está generando una trama STS-1.N=1, Num_gen=74. Configurar para que el generador 74 genere el valor por defecto más uno.

Respuesta:

1. En los pines de salida del módulo deberán aparecer los siguientes datos:

Nombre del pin	Valor por omisión (Hex.)
K1	31
K2	32
D4	D4
D5	D5
D6	D6
D7	D7
D8	D8
D9	D9
D10	DA
D11	DB
D12	DC
S1	S1
M1	22
E2	E2

2. Debe de responder semejante al inciso anterior con la diferencia que el debe aparecer el valor por omision más uno.

Nombre del pin	Valor por omisión (Hex.)
K1	32
K2	33
D4	D5
D5	D6
D6	D7
D7	D8
D8	D9
D9	DA
D10	DB
D11	DC
D12	DD
S1	S2
M1	23
E2	E3

4.4 Prueba GB_RSOH_2:

Objetivo: Comprobar que el generador básico es capaz de generar los datos de RSOH, en una trama STM –Nc

Excitación:

1. Configurar al módulo para simular que se está generando una trama STM-1 y que los bytes de RSOH que genere sean iguales a su valor por defecto.
2. Configurar al módulo para simular que se está generando una trama STM-4c y que los bytes de RSOH que genere sean iguales a su valor por defecto.
3. Configurar al módulo para simular que se está generando una trama STM-16c y que los bytes de RSOH que genere sean iguales a su valor por defecto.
4. Configurar al módulo para simular que se está generando una trama STM-64c y que los bytes de ROS que genere sean iguales a su valor por defecto.

Respuesta:

Verificar que las tramas generadas correspondan con la configuración y valores por defecto.

4.5 Prueba GB_POH_1:

Objetivo: Se verificará que el generador básico genera sus valores por defecto y que es capaz de generar diferentes datos de acuerdo a lo que el usuario determine en el POH.

Excitación:

1. Configurar al módulo para simular que se está generando una trama STM-1. Num_gen=1.
2. Configurar al módulo para simular que se está generando una trama STM-1. Num_gen=34. Configurar para que el generador 34 genere el valor por defecto más uno.

Respuesta:

1. En los pines de salida del modulo deberán aparecer los siguientes datos:

Nombre del pin	Valor por omisión (Hex.)
J1	11
C2	C2
G1	G1
F2	F2
H4	H4
F3	23
K3	24
N1	25

2 Debe de responder semejante al inciso anterior con la diferencia que el debe aparecer el valor por omisión más uno.

Nombre del pin	Valor por omisión (Hex.)
J1	12
C2	C3
G1	G2
F2	F3
H4	H5
F3	24
K3	25
N1	26

4.6 Prueba GB_POH_2:

Objetivo: Comprobar que el generador básico cuando se le indica que se generara una trama STM–Nc es capaz de generar los datos de POH acuerdo al orden de la trama.
Excitación:

1. Configurar al módulo para simular que se está generando una trama STM-1 y que los bytes de POH deben ser igual a sus diferentes valor por omisión.
2. Configurar al módulo para simular que se está generando una trama STM-4c y que los bytes que genere de POH deben ser igual a sus diferentes valor por omisión.
3. Configurar al módulo para simular que se está generando una trama STM-16c y que los bytes que genere de POH deben ser igual a sus diferentes valor por omisión.
4. Configurar al módulo para simular que se está generando una trama STM-64c y que los bytes que genere el generador POH deben ser igual a sus diferentes valor por omisión
5. Respuesta

   Verificar que las tramas generadas correspondan con la configuración y valores por defecto

4.7 GB_Conts:

Objetivo: Comprobar que el generador básico es capaz de producir un dato constante.
Excitación: Configurar el generador como STM-1, para que genere 55H.
Respuesta: En la salida del generador debe aparecer 55H.

4.8 GB_Conts_ Ascendente:

Objetivo: Comprobar que el generador es capaz de generar una cuenta ascendente en una trama STS-1.
Excitación: Configurar al generador para que genere una cuenta ascendente y que su cuenta inicial sea 56H, la entrada de fila debe variar de 1 a 9 y de 1 a 90.
Respuesta: El generador debe de comenzar con 56 y debe incrementarse de uno en uno con forme columna cambie y debe llegar al valor 255 para que la siguiente cuenta sea 0, 1,2 .. 56... 255.

4.9 GB_Conts_ Descendente:

Objetivo: Comprobar que el generador es capaz de generar una cuenta descendente en una trama STM-1.
Excitación: Configurar al generador para que genere una cuenta ascendente y que su cuenta inicial sea 56H, la entrada de fila debe variar de 1 a 9 y de 1 a 90.
Respuesta: El generador debe de comenzar con 56 y debe decrementarse de uno en uno con forme columna cambie y debe llegar al valor 0 para que la siguiente cuenta sea 255, 254,253 .. 56... 1,0 Hex.

4.10GB_P1:

Objetivo: Se verificará la correcta generación de puntero en tramas STM-1, STM-4c, STM-16c y STS-64c.
Excitación:
Configurar el generador básico para que el valor del puntero sea 422 decimal. La carga útil se debe configurar para que produzca un valor constante de 57Hex. Se procederá de igual manera en cada uno de los diferentes órdenes de las tramas indicadas en el objetivo de la prueba.
Respuesta:
La trama debe de contener los valores por defecto, y el valor del apuntador debe de aparecer en la posición definida para un valor de puntero de 422 decimal, dependiendo del orden de la trama.

4.11GB_P2:

Objetivo: Verificar la correcta generación de puntero cuando se realiza un movimiento de puntero positivo en tramas STM-1, STM-4c STS-16c y STS-64c.
Excitación:
Configurar el generador básico para que el valor del puntero sea 422 decimal. La carga útil se debe configurar para que produzca un valor constante de 57Hex. Permitir que se generen 5 tramas con el valor del puntero 422 decimal, en la sexta trama debe ser de ser configurado el generador para que realice un movimiento de apuntador positivo, en la séptima trama se debe de quitar la condición de movimiento de apuntador positivo. Posteriormente se permitirá que el generador produzca tres tramas con el ultimo valor de puntero. Realizar lo anterior con los diferentes ordenes de las tramas indicadas en el objetivo de la prueba.
Respuesta:

El puntero debe de aparecer en la posición del puntero igual a 422 decimal durante las cinco primeras tramas. En la quinta sexta trama deben de verificarse que los bits I del puntero cambien de acuerdo aun movimiento de puntero positivo y los bits D deben de permanecer sin alteración. En la séptima trama el valor de puntero debe ser 423 decimal y ese valor debe de permanecer en las subsecuentes tramas.

4.12GB_P3:

Objetivo: Verificar la correcta generación de puntero cuando se realiza un movimiento de puntero negativo en tramas STM-1, STM-4c, STM-16c, STM-64c.
Excitación:
Configurar el generador básico para que el valor del puntero sea 422 decimal. La carga útil se debe configurar para que produzca un valor constante de 57H. Permitir que se generen 5 tramas con el valor del puntero 422 decimal, en la sexta trama debe se de configurado el generador para que realice un movimiento de apuntador negativo, en la séptima trama se debe de quitar la condición de movimiento de apuntador positivo y permitir que el generador produzca tres tramas de con el ultimo valor del apuntador. Realizar lo anterior con las tramas indicadas en el objetivo de la prueba.

Respuesta:

El puntero debe de aparecer en la posición del puntero igual a 422 decimal durante las cinco primeras tramas. En la quinta sexta trama deben de verificarse que los bits D del puntero cambien de acuerdo a un movimiento de puntero negativo y los bits I deben permanecer sin alteración. En la séptima trama el valor de puntero debe ser 421 decimal y ese valor debe de permanecer en las subsecuentes tramas.

4.13GB_P4:

Objetivo: Verificar que el generador es capaz de soportar un movimiento de apuntador positivo, cuando el valor del puntero sea 782 decimal. Así como también se verificará que soporta un movimiento de apuntador negativo cuando el valor del puntero es cero decimal y por último verificar que puede realizar la operación de nuevo puntero. Lo anterior debe de verificar en tramas en tramas STM-1, STM-4c, STM-16c, STM-64c.
Excitación:

a) Configurar el generador básico para que el valor del puntero sea 422 decimal. La carga útil se debe configurar para que produzca un valor constante de 57Hex. Permitir que se generen 5 tramas con el valor del puntero 422 decimal, en la sexta trama debe ser configurado el generador para que realice la operación de nuevo puntero, el cual debe ser 200 decimal, en la séptima trama se debe de quitar la condición de nuevo puntero y poner la condición de puntero normal. Permitir que el generador produzca tres tramas. Realizar lo anterior con las tramas indicados en el objetivo de la prueba.

b) Configurar el generador básico para que el valor del puntero sea 0 decimal. La carga útil se debe configurar para que produzca un valor constante de 57 Hex. Permitir que se generen 5 tramas con el valor del puntero cero decimal, en la sexta trama debe ser configurado el generador para que realice la operación de

decremento de puntero, en la séptima trama se debe de quitar la condición de decremento de puntero y poner la condición de puntero normal. Permitir que el generador produzca tres tramas. Realizar lo anterior con las tramas indicados en el objetivo de la prueba.

c) Configurar el generador básico para que el valor del puntero sea 782 decimal. La carga útil se debe configurar para que produzca un valor constante de 57H. Permitir que se generen 5 tramas con el valor del puntero 782 decimal, en la sexta trama debe ser configurado el generador para que realice la operación de incremento de puntero, en la séptima trama se debe de quitar la condición de incremento de puntero y poner la condición de puntero normal. Permitir que el generador produzca tres tramas. Realizar lo anterior con las tramas indicados en el objetivo de la prueba.

Respuesta:

a) El puntero debe de aparecer en la posición del puntero igual a 422 decimal durante las cinco primeras tramas. En la sexta trama deben de verificarse que los bits N del puntero sean invertidos y el nuevo puntero aparezca. En la séptima trama los bits N deben aparecer indicando puntero normal y el valor de puntero debe ser 200 este valor debe de permanecer en las subsecuentes tramas.

b) El puntero debe de aparecer en la posición del puntero igual a 0 durante las cinco primeras tramas. En la sexta trama deben de verificarse que los bits D del puntero sean invertidos y los bits I y N no deben ser alterados; verificar que en la posición del byte H3 se transporte el byte correspondiente byte de POH. En la séptima trama los bits N deben aparecer indicando puntero normal y el valor de puntero debe ser 782 este valor debe de permanecer en las subsecuentes tramas.

c) El puntero debe de aparecer en la posición del puntero igual a 782 durante las cinco primeras tramas. En la sexta trama deben de verificarse que los bits I del puntero sean invertidos y los bits D y N no deben ser alterados; verificar que el byte que se encuentra a lado del byte H3 aparezca 00H, el cual debe sel considerado de relleno para conformar la trama. En la séptima trama los bits N deben aparecer indicando puntero normal y el valor de puntero debe ser 0 este valor debe de permanecer en las subsecuentes tramas.

4.14 GB_Change:

Objetivo: Comprobar que es posible programar todo los byte tramas: STM-1, STM-4c, STM-16c, STM-64c.

Excitación:
Cambiar todos los bytes de la trama para que tengan un valor diferente al indicado por omision, este cambio debe de tener efecto en la quinta trama.
Respuesta:
Verificar que todas las tramas antes de la sexta trama tengan los valores por omisión y que en las subsecuentes tramas el valor de cada byte corresponda al programado por el usuario.

4.15 GB_TraceJ0:

Objetivo: Constatar que el generador es capaz de transmitir una huella por el byte J0.
Excitación:

Configurar al generador para que transmita un mensaje de 15 bytes por el byte J0, haciendo uso del Código ASCII. Para esta prueba hacer uso de una trama STM-1.

Respuesta:

Leer el dato que se encuentra en el byte J0 durante las primeras 15 tramas y verificar que agrupando los 15 bytes correspondan al mensaje enviado por el usuario. En las siguientes 15 tramas verificar que el mensaje se sigue repitiendo. Además verificar que el mensaje se encuentra en ASCII.

4.16 GB_TraceJ1:

Objetivo: Constatar que el generador es capaz de transmitir una huella por el byte J1.
Excitación:

Configurar al generador para que transmita un mensaje de 15 bytes por el byte J1, haciendo uso del Código ASCII. Para esta prueba hacer uso de una trama STS-1.

Respuesta:

Leer el dato que se encuentra en el byte J1 durante las primeras 15 tramas y verificar que agrupando los 15 bytes correspondan al mensaje enviado por el usuario. En las siguientes 15 tramas verificar que el mensaje se sigue repitiendo. Además verificar que el mensaje se encuentra en ASCII.

4.17GE_Top1:

Objetivo: Verificar la multiplexación del generador y los movimientos de puntero a un nivel de STM-N.

Excitación:

En todas las subsecuentes pruebas se debe cotejar que las tramas se alinean con el pulso de sincronía.

a) La prueba consistirá en generar tramas STM-N, donde N=1,3,12,48. El valor del puntero inicial será 522 decimal no se realizaran movimientos de apuntador, ni tampoco cambio de nuevo puntero. Configurar en el puerto 1 para que genere una trama STM-1, el puerto 3 para que genere tramas STM-1c, el puerto 24 para que genere tramas STM-4c, puerto 48 para que genere tramas STM-16c.

b) La prueba consistirá en forma tramas STM-N con diferentes órdenes de múltiplexación utilizando diferentes generadores básicos. El valor del puntero inicial será 522 no se realizarán movimientos de apuntador, ni tampoco cambio de nuevo puntero.. Para lo anterior se empleará la siguiente tabla:

Trama a formar	Arreglo de generadores	Puerto de Salida
STM-1	1 generador básico que genere tramas STM-1	30
STM-1	1 generador básico que genere tramas STM-1	24
STM-4	4 generadores básicos que generen tramas STM-1	12
STM-16	16 generadores básicos que generen tramas STM-1	48
STM-16	4 generadores básicos que generen tramas STM-1 3 generador básicos que genere tramas STM-16c	48
STM-4	Utilizar cuatro diferentes generadores básicos que generen tramas STM-1.	24
STM-16	Utilizar dieciséis diferentes generadores básicos que generen tramas STM-1 cada uno de ellos.	1
STM-16	Utilizar cuatro diferentes generadores básicos que generen tramas STM-4c cada uno de ellos.	49

STM-16	Utilizar un generador básico que genere tramas STM-4. Un generador básico que genere tramas STM-4c. Utilizar cuatro generadores básicos que generen tramas STM-1. Utilizar dos generadores básicos que generen tramas STM-1c. Utilizar dos generadores básicos que generen tramas STM-1.	16
STM-64	Utilizar 64 generadores básico que generen tramas STM-1	1
STM-64	Utilizar 16 generadores básicos que generen tramas STM-4c	1
STM-64	Utilizar 4 generadores básicos que generen tramas STM-16	1

Respuesta:

a) Verificar que en los puertos 1,3,24,48 se generan tramas STM-1, STM-1c, STM-4c y STM-16c respectivamente. Donde el apuntador debe ser 522. checar los bytes H1,H2,H3.

b)Verificar que en el puerto definido en la tabla anterior se genere el orden de trama indicada, además de sea correcta la multiplexación.

4.18GE_Top2:

Objetivo: Verificar que el generador es capaz de soportar movimientos de apuntador, nuevo puntero.
Excitación:
Utilizar la tabla de la prueba GE_Top1, se realizarán movimientos de apuntador y nuevo puntero, la manera de hacer dichas operaciones es la siguiente: Poner el puntero desde el inicio de la prueba en 522 decimal, en la trama 2 hacer un decremento, en la trama 3 indicar puntero normal, en la trama 4 realizar un incremento de puntero, en la trama 5 se realiza puntero normal, en la trama 6 realizar un nuevo puntero el cual debe ser 0, en la trama 7 indicar puntero normal, en la trama 8 realizar un incremento de puntero, en la trama 9 indicar puntero normal, en la trama 10 realizar un incremento de puntero, en la trama 11 indicar puntero normal, en la trama 12 realizar un nuevo puntero el cual debe

ser 492, en la trama 13 indicar puntero normal, en la trama 14 realizar un incremento de puntero, en la trama 15 indicar puntero normal.
Respuesta:
Verificar que la estructura de multiplexación se respeta y además que se dada lo siguiente: en la trama 1 debe aparecer el puntero 522 decimal, en la trama 2 los bits D deben ser invertidos, en la trama 3 debe aparecer el puntero 521 decimal, en la trama 4 se deben invertir los bits I , en la trama 5 debe aparecer el puntero 522 decimal en la trama 6 se debe invertir los bits N y el nuevo puntero 0, en la trama 7 debe aparecer el puntero 0, en la trama 8 debe aparecer invertidos los bits I, en la trama 9 se debe presentar en el puntero 782 decimal, en la trama 10 se deben de invertir los bits I, en la trama 11 se debe presentar el puntero 0, en la trama 12 deben invertirse los bits N y el nuevo puntero 492 decimal, en la trama 13 debe aparecer el puntero 492 decimal, en la trama 14 debe aparecer los bits I invertidos y de la trama 15 en adelante debe aparecer el puntero 493 decimal.

4.19 GE_LOS

Objetivo:
Verificar la generación de una alarma LOS en un puerto.
Excitación:
Configurar el puerto 12 para que genere tramas STM-1, configurar el módulo para que en la trama 6 comience a generar una alarma LOS y en la trama 8 pare la generación de esta alarma.
Respueta:

Se debe de observar de la trama 1 a la trama 5, en el puerto 12, la generación de tramas STM-1 con sus valores por defecto en cada uno de sus bytes, a partir de la trama 6 hasta la trama 8 se debe de observar en todos los bytes de la trama el valor de FF Hex, cuando comience la trama 9 se deben de observar los diferentes valores por defecto en la trama.

4.20 GE_AISR

Objetivo:
Verificar la generación de una alarma AIS-R en un puerto.
Excitación:
Configurar el puerto 24 para que genere tramas STM-1, configurar el módulo para que en la trama 6 comience a generar una alarma AISR y en la trama 8 pare la generación de esta alarma.
Respueta:

Se debe de observar de la trama 1 a la trama 5, en el puerto 24, la generación de tramas STM-1 con sus valores por defecto en cada uno de sus bytes, a partir de la trama 6 hasta la trama 8 se debe de observar en todos los bytes pertenecientes tanto a los bytes del contenedor virtual como a los bytes H1,H2,H3 de la trama el valor de FF Hex, cuando comience la trama 9 se deben de observar nuevamente los diferentes valores por defecto en la trama. Nótese

que los bytes de ROS y MSOH no son afectados con esta alarma y por lo tanto deben de contener en todas las tramas sus valores por defecto.

4.21 GE_AISL

Objetivo:
Verificar la generación de una alarma AISL en un puerto.
Excitación:
Configurar el puerto 24 para que genere tramas STM-1, configurar el módulo para que en la trama 6 comience a generar una alarma AISL y en la trama 8 pare la generación de esta alarma.
Respueta:
Se debe de observar que en todas las tramas generadas el RSOH permanece con sus valores por defecto, lo mismo sucede con el resto de los bytes que conforman la trama de la trama 1 a la trama 5, empero, de la trama 6 a la trama 8 todos los bytes excepto los bytes de RSOH contienen el valor FF Hex. De la trama 9 en adelante toda la trama contendrá los valores por defecto.

4.22 GE_RDI

Objetivo
Verificar la generación de una alarma RDI en un puerto.
Excitación
Configurar el puerto 24 para que genere tramas STM-1, configurar byte K2 de que sale de este puerto para que contenga el valor 06 Hex a partir de la trama 6 hasta la trama 10 y a partir de la trama 11 configurar el byte K2 para que genere su valor por defecto.
Respuesta:
De la trama 1 a la trama 5 el byte K2 tendrá el valor por defecto, en las tramas 6 a 10 este byte debe de contener el valor de 06 Hex. y de la trama 11 en adelante este byte debe de contener su valor por defecto. El resto de los bytes pertenecientes a la trama serán los de defecto.

4.23 GE_AIS

Objetivo
Verificar la generación de una alrma AIS en un puerto.
Excitación
Configurar el puerto 24 para que genere tramas STM-1, configurar byte K2 de que sale de este puerto para que contenga el valor 07 Hex a partir de la trama 6 hasta la trama 10 y a partir de la trama 11 configurar el byte K2 para que genere su valor por defecto.
Respuesta:
De la trama 1 a la trama 5 el byte K2 tendrá el valor por defecto, en las tramas 6 a 10 este byte debe de contener el valor de 07 Hex. y de la trama 11 en adelante este byte debe de contener su valor por defecto. El resto de los bytes pertenecientes a la trama serán los de defecto.

4.24 GE_J0

Objetivo
Verificar que es posible mandar un mensaje en el byte J0.
Excitación:
Configurar el puerto 24 para que genere tramas STM-1, configurar el módulo para que genere el siguiente mensaje "Hola%mundo%$" en el byte J0.

Respuesta:
Se debe de guardar 16 bytes J0 en 16 tramas, verifique que el los valores obtenidos en los 16 bytes corresponden a el mensaje enviado en el byte J0.

4.25 GE_J1

Objetivo
Verificar que es posible mandar un mensaje en el byte J1.
Excitación:
Configurar el puerto 24 para que genere tramas STM-1, configurar el módulo para que genere el siguiente mensaje "Hola%mundo%$" en el byte J1.

Respuesta:
Se debe de guardar 16 bytes J0 en 16 tramas, verifique que el los valores obtenidos en los 16 bytes corresponden a el mensaje enviado en el byte J1.

5 Capítulo V
Resultados y Conclusiones.

La implementación del generador se realizó en el lenguaje VHDL. Los tiempos de simulación que se obtuvieron fueron realizados en una computadora de escritorio con procesador Pentium 3. Las mediciones de desempeño están referidas a 1ms. En la tabla siguiente se muestran estos tiempos.

Trama a formar	Arreglo de generadores	Tiempo de 1 ms de simulación.
STM-1	1 generador básico que genere tramas STM-1	5 :04 min
STM-4	4 generadores básicos que generen tramas STM-1	5:56 min
STM-16	16 generadores básicos que generen tramas STM-1	13:13 min
STM-16	4 generadores básicos que generen tramas STM-1 3 generador básicos que genere tramas STM-4c	17:42 min
STM-16	Utilizar cuatro diferentes generadores básicos que generen tramas STM-12c cada uno de ellos.	15:55 min
STM-16	Utilizar un generador básico que genere tramas STM-4. Un generador básico que genere tramas STM-4c. Utilizar cuatro generadores básicos que generen tramas STM-1. Utilizar un generadores básicos que generen tramas STM-4c.	18:11 min
STM-64	Utilizar 64 generadores básicos que generen tramas STM-1	23:05 min
STM-64	Utilizar 16 generadores básicos que generen tramas STM-4c	21:02 min
STM-64	Utilizar 4 generadores básicos que generen tramas STM-16	19:30 min

Características del Generador:

- El generador cuenta con un archivo de entrada llamado SDH_frame.do
- Genera tramas STM-1,4,16,64.
- Tiene 64 puertos a 155 520 Kbit/s
- Es capaz de configurar 16 puertos independientes a 622080 Kbit/s
- Es capaz de configurar 4 puertos independientes a 2 488 320 Kbits/s
- Es capaz de configurar un puerto a 9 953 280 Kbit/s
- Es capaz de configurar cualquier byte de una trama SDH que se genere por un puerto.
- Es capaz de obtener su sincronía externa o generarla internamente.
- Realiza el calculo de los bytes de paridad par.
- Realiza manejo de apuntadores.
- Es capaz de insertar errores dentro de la trama.
- Es capaz de reprogramar cualquier byte de la trama.
- Es capaz de generar carga útil del tipo: constante, contador, PRBS.
- Es capaz de generar traza tanto en J0 como en J1.
- Es capaz de habilitar o deshabilitar el aleatorizador de datos.
- Es capaz de simular la generación de alarmas.
- Cada puerto consta de una interfaz de 16 bits.
- Cuenta con dieciséis instrucciones capaces de configurar al generador.
- Número de líneas de código en VHDL del Generador: 6914.

6 Conclusiones

En esta tesis se definió la especificación, la arquitectura, el plan de verificación y la implementación de un generador de tramas SDH. Actualmente en la literatura se hace mención de camas de prueba como un concepto o una entidad que debe de existir cuando se verifica un circuito sin adentrarse en metodología de diseño, características de una cama de pruebas, metodología de verificación de la cama. En esta tesis es posible encontrar todos estos aspectos los cuales pueden servir de base para cualquier lector que esté interesado en implementar con éxito una cama de pruebas la cual no necesariamente debe ser para el protocolo SDH.

La arquitectura empleada en esta tesis puede ser empleada para conseguir una cama de pruebas sintetizable, la diferencia radicaría en el empleo de las instrucciones, las cuales deberán ser sintetizables.

Con la versatilidad de tener diferentes puertos, y manejar por un mismo puerto diferentes niveles de multiplexación permite poder utilizar a esta cama como una valiosa herramienta en la verificación funcional de circuitos que manejen el protocolo SDH a niveles de STM-64, 16, 4,1.

En una nueva versión de la cama seria bueno que cada puerto sea posible configurarlo como STM-64,16,4,1, además que exista una interfaz grafica capaz de configurar la generación de las tramas.

7 Bibliografía:

1. http://www.amcc.com/Products/Sonet/prod_236.htm

2. Joshi, Paul Loewenstein, "System Design Methodology of UltraSPARC" 32nd ACM/IEEE Design Automation Conference,1995.

3. Stephen T. Frezza, Steven P. Levitan, Panos K. Chrysanthis, "Requirements-Based Design Evaluation" 32nd ACM/IEEE Design Automation Conference,1995.

4. James Monaco, David Holloway, Rajes Raina, "Functional Verification Methodology for the PowerPC 604 Microprocessor" 33rd ACM/IEEE Design Automation Conference,1996.

5. Anoosh Hosseini, Dimitrios Mavroidis, Pavlos Konas, "Code Generation and Análisis for the Functional Verification of Microprocessors" 33rd ACM/IEEE Design Automation Conference,1996.

6. Bergeron Janick, "Writing Testbenches: functional verification of HDL models" Kluwer Academic Publishers 2000.

7. Hetzel Bill,"The Complete Guide to Software Testing", John Wiley & Sons, Inc Second Edition 1988.

8. Nahum Vladimir Castillo Felix, "Herramienta para el desarrollo y pruebas de algoritmos computacionales, con aplicación a manejadores externos de papel Hewlett Packard", Tesis del Cinvestav Unidad Guadalajara,2000.

9. Dirceu Cavendish, C&C Research Laboratories "Evolution of Optical Transport Technologies:From SONET/SDH to WDM" IEEE Común. Mag., vol. 38, no.6 June 2000, pp 164 –172.

10. Nilanjan Mukherjee and J.Chakraborty, "Built-In Self-Test: A Complete Test Solution for Telecommunication Systems" IEEE Comun. Mag., vol 37 no 6 June 1999 ,pp 72-78.

11. G.707, Series G: Transmisión Systems and Media, Digital Systems and Networks. Interfaz de nodo de red para la jerarquia digital sincronía. Oct 2000.

12. G.780, Series G: Transmisión Systems and Media, Digital Systems and Networks. Vocabulario de términos de redes y equipos de la jerarquía digital síncrona. July 2000.

13. G.783, Series G: Transmisión Systems and Media, Digital Systems and Networks.Characteristics of syschronous digital hierarchy equipment functional blocks. April 1997.

14. Ian Sommerville, "Software Engineering" Addisson Wesley 5th ed.1995.

15. Stefan Sjoholm, Lennart Lindh, "VHDL for Designers", Prentice Hall,1997

16. Cem Kaner, Jack Falk, "Testing Computer Software" John Wiley & Sons, Inc.1999.

17. Luis Alberto Núñez Rodríguez, "Matriz de Conmutación de Alta Velocidad: su Verificación y su Elemento de Salida", Tesis del Cinvestav Unidad Guadalajara,2000.

18. "IEEE Std 829-1998, IEEE Standard for Software Test Documentation, Software Engineering Volume Four: Resource and Technique Standards" The Institute of Electrical and Electronics Engineers, Inc.

19. Paul Bonenfant and Antonio Rodríguez, "Optical Data Networking", IEEE Común. Mag., vol. 38, no.3 March 2000, pp 63 –70.

20. Stamatios V. Kartalopoulos, "Understanding SONET/SDH and ATM" IEEE Press Editorial Board 1999.

Centro de Investigación y de Estudios Avanzados del IPN

Unidad Guadalajara

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, aprobó la tesis: DISEÑO Y VERIFICACIÓN DE UN GENERADOR DE TRAMAS SDH del(a) C. Enrique GONZALEZ GARCIA el día 21 de Diciembre de 2001

Dr. Deni Librado Torres
Román
Profesor Investigador 3A
CINVESTAV GDL
Guadalajara

Dr. Manuel Edgardo
Guzmán Renteria
Investigador Cinvestav 3A
CINVESTAV GDL
Guadalajara

M.C. José María Uruñuela
Martínez
Profesor, Depto Electronica
e Informatica
INSTITUTO TECNOLOGICO
DE ESTUDIOS SUPERIORES
DE OCIDENTE
Guadalajara