Las pruebas integradas en FPGA aceleran el desarrollo

Artículo cedido por Agilent Technologies

Por Brad Frieden, Agilent Technologies

Más información en, www.agilent.com/find/fpga

Los puertos de barrido y núcleos de medida integrados en FPGA simplifican las tareas de depuración y validación, lo que permite ahorrar tiempo en el desarrollo de proyectos

La aparición de diversos núcleos de medida FPGA de uso general, junto con puertos de barrido y núcleos de medida basados en microprocesadores incorporados, ofrece nuevas opciones para la validación y depuración a nivel del sistema y la posibilidad de ahorrar un tiempo considerable. Las opciones disponibles presentan algunos compromisos que pueden afectar al número de patillas, los recursos necesarios y el nivel de abstracción de los datos capturados que llegan al diseñador. Estas opciones y estos compromisos plantean una serie de situaciones de validación y depuración que se describen a continuación.

Algunos compromisos básicos en la selección de herramientas

Uno de los primeros compromisos tiene que ver con el número de patillas FPGA (si hay alguna disponible) que un diseñador puede dedicar al proceso de depuración y validación. Si no puede dedicar patillas a la depuración, el diseñador deberá utilizar el analizador lógico integrado ChipScope Pro para obtener visibilidad interna en el FPGA. ChipScope Pro utiliza un núcleo de analizador lógico integrado (ILA) y bloques de RAM para capturar señales.

Si el diseñador puede dedicar algunas patillas FPGA a la depuración, dispondrá de una gran variedad de nuevas opciones. Una de estas opciones consiste en utilizar una serie de núcleos de medida que llevan las señales a patillas que se sondean con un analizador lógico o un osciloscopio de señales mixtas, como se muestra en la figura 1. Los núcleos de trazado de Agilent (ATC2, Agilent Trace Core) permiten realizar análisis de tiempos, análisis de estados y análisis de estados con compresión de patillas.

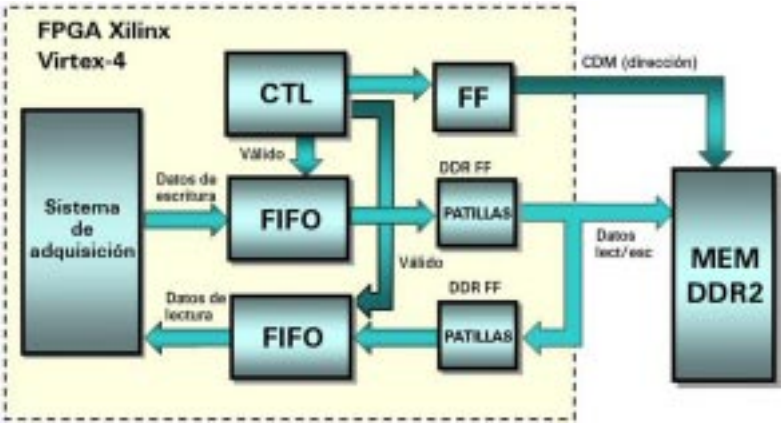


Figura 1. Sistema de medida de sonda dinámica FPGA con núcleo ATC2

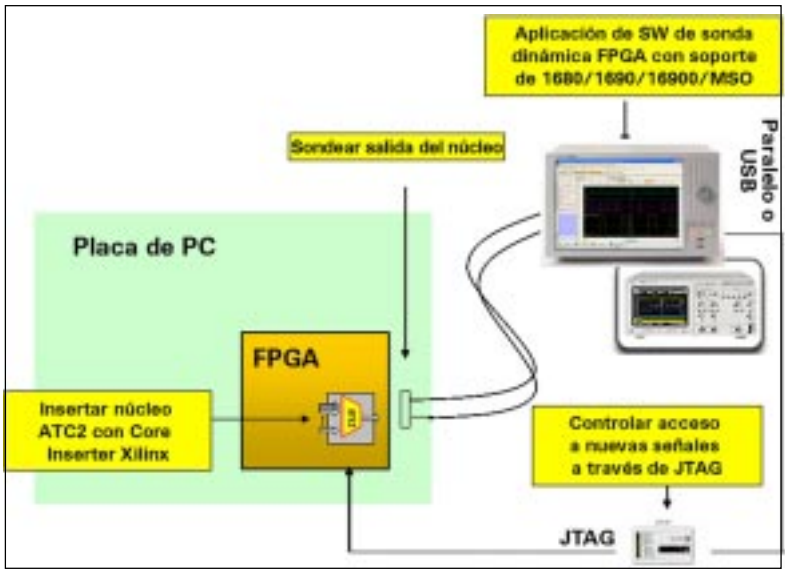
Aplicaciones de ejemplo

Consideremos una serie de aplicaciones que pueden beneficiarse del uso de puertos de barrido y núcleos de medida basados en analizadores lógicos. Incluirán las siguientes:

- Depuración de la interfaz de memoria DDR2 utilizando un núcleo de temporización
- Validación a nivel del sistema utilizando un núcleo de trazado de MicroBlaze y captura de temporización externa
- Verificación del flujo del programa utilizando un puerto de barrido PPC 405 de IBM

Depuración de la interfaz de memoria DDR2

Recientemente, un grupo de I+D de Agilent Technologies se enfrentó al desafío de activar una interfaz de memoria DDR2 implementada con FPGA Xilinx Virtex-4. Se realizaron pruebas iniciales con la placa de desarrollo Xilinx ML-403. El medio empleado para trazar señales a través del sistema de memoria fue un núcleo de temporización de medida ATC2, lo que permitió depurar el controlador de memoria.



El sistema digital adquirió y procesó datos, ubicándolos en primer lugar en la memoria DDR2 y, a continuación, recuperándolos de la memoria a demanda para su procesamiento, con la arquitectura que se muestra en la figura 2. La memoria funcionó con una velocidad de reloj de 200 MHz. En el diseño del prototipo, había algo en el sistema de memoria que estaba dañando los datos.

Sistema de depuración basado en el uso de núcleo de temporización de medida y un analizador lógico

Se empleó una metodología básica para validar y depurar el sistema de memoria. Se trataba de observar primero la ruta de control, luego la ruta de datos y, por último, la interacción entre las rutas de control y de datos. Varios aspectos importantes garantizaban el éxito de esta metodología.

1) acceso a suficientes bits de la máquina de estados del controlador de memoria para verificar los ciclos de escritura y lectura de interés

2) creación de un estímulo conocido y seguimiento del mismo hasta el final, incluida una relectura (read-back)

3) acceso como mínimo a unos pocos bits del bus de datos para trazar los datos hasta el final

4) sondeo para ver y comprobar la temporización aproximada entre las señales de control y datos

En este caso específico, el diseñador disponía de 10 patillas para depuración. Para trazar estas señales había que tener en cuenta también varios dominios de reloj. El núcleo de temporización ATC2 de Agilent ofrecía una buena opción en esta aplicación, ya que una única sonda de temporización podía sondear múltiples dominios de reloj y asignar las señales de dichos dominios a una serie fija de patillas.

Uno de los compromisos que supone el uso de un núcleo de temporización es que el diseñador debe soportar la oblicuidad (skew) entre canales que se forma debido a las diferentes longitudes de ruta necesarias para encaminar las señales de FPGA internas a patillas FPGA. El núcleo de temporización se ha diseñado intencionadamente para tratar esas rutas como "falsas rutas", con el fin de evitar sobrecargas en las herramientas de diseño y producir limitaciones de temporización difíciles para una medida. De este modo puede reservarse toda la energía para satisfacer las limitaciones de temporización del propio diseño.

El sistema de medida

Un analizador lógico portátil 16800 ejecutó la aplicación de medida de sonda dinámica FPGA de Agilent que se muestra en la figura 3. Se conectó al sistema de destino a través de JTAG para la programación FPGA y selección de MUX y a través de sondas de terminales móviles para la captura de señales.

Definición de bancos de señales

Se definieron bancos de señales para comprobar en primer lugar la máquina de estados básicos del controlador de memoria y, seguidamente, para trazar las señales que fluían por la memoria. Por último, los bancos de señales examinaron el control y los datos para verificar la temporización aproximada.

Banco 0: DQaout (4 bits inferiores), DQbout (4 bits inferiores), Vld, DQS

Banco 1: DQain (4 bits inferiores), DQbin (4 bits inferiores), Vld, DQS

Banco 2: máquina de estados de control

Banco 3: comando (dirección) en anillo de E/S

Banco 4: datos de escritura entrando FIFO

Banco 5: datos de escritura en anillo de E/S

Banco 6: datos de lectura en anillo de E/S

Banco 7: datos de lectura en salida FIFO

Aparentemente, la máquina de estados básicos ofrecía un funcionamiento correcto, al igual que los datos escritos en la memoria.

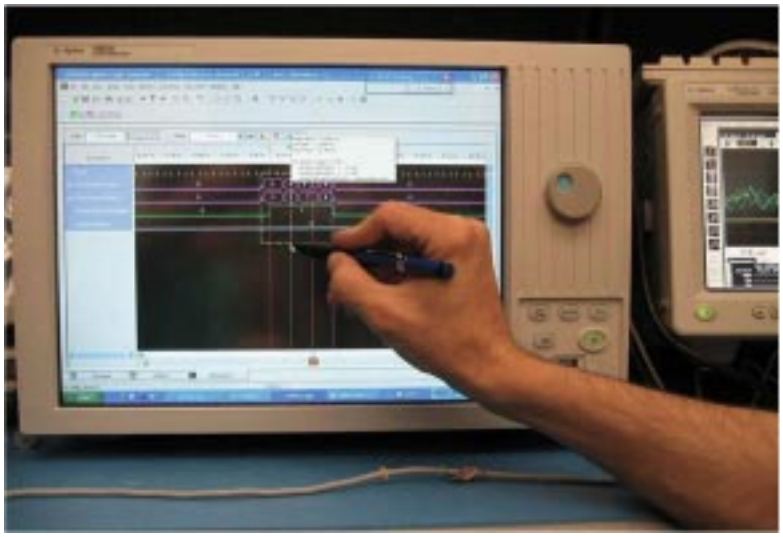


Figura 3. Analizador lógico portátil de la Serie 16800 de Agilent con sonda dinámica FPGA

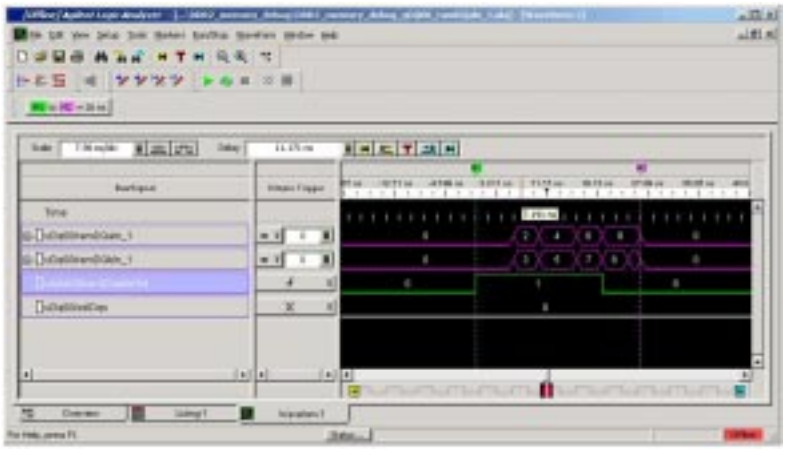


Figura 4. Vld presenta un ciclo de reloj anticipadamente y desalineado con los datos A y B de DQS

Los datos que se leían de la memoria tampoco presentaban errores. La medida final de los bancos, que examinaba tanto el control como los datos, reveló la relación inicial entre señal de muestreo y datos que se muestra en la figura 4. El diseño del controlador de memoria se modificó a través de un valor de registro para alinear Vld con los datos A y B de DQS. De este modo se resolvió el problema del deterioro de los datos.

Se podía haber estudiado la posibilidad de utilizar un núcleo de estados. Sin embargo, cada dominio del reloj hubiera requerido su propio núcleo de estados, y cada núcleo, su propia serie de patillas FPGA.

Validación a nivel del sistema con un núcleo de trazado de MicroBlaze y una traza externa correlacionada temporalmente

Desde hace diez años, con la aparición del almacenamiento en caché y del pipelining, se ha hecho más difícil trazar la ejecución de procesadores integrados. Sin embargo, con los avances en tecnología de trazado, los nuevos procesadores integrados en FPGA (como el procesador MicroBlaze integrado en FPGA de la familia Xilinx) ofrecen una excelente visibilidad sobre su funcionamiento. Esto es posible gracias a los nuevos núcleos de medida que sondean por detrás del caché, en combinación con ensambladores inversos que se ejecutan sobre un analizador lógico.

Asimismo, dada una conexión tan intensa con las señales del microprocesador, es posible establecer una estrecha correlación temporal entre la actividad del microprocesador y eventos externos como una interfaz FPGA con la memoria externa. Este sistema ofrece una gran ayuda en la validación y depuración a nivel del sistema.

Ejemplo de depuración con MicroBlaze

Se puede apreciar un buen ejemplo de esto en las medidas tomadas en un sistema en el que un procesador MicroBlaze integrado forma parte del diseño de un controlador de memoria. Se sondearon tanto el MicroBlaze como la memoria externa con el fin de validar el funcionamiento del sistema de memoria. Se trazó el flujo del programa, incluidos los comandos de escritura y lectura de MicroBlaze, sondeando el MicroBlaze con un núcleo de trazado de MicroBlaze. Las medidas de temporización de la memoria externa ayudaron a determinar si se había conseguido la interfaz de memoria adecuada.

El núcleo de trazado de MicroBlaze (MTC) se ubicó en el diseño empleando el generador de núcleos Xilinx desde el Embedded Developers Kit (EDK). En este paso fue necesario realizar algunos compromisos, en concreto en cuanto al número de patillas FPGA que podían dedicarse a la medida, pues afectaba a las señales de MicroBlaze que podían verse.

A continuación, en el analizador lógico, la interfaz entre la

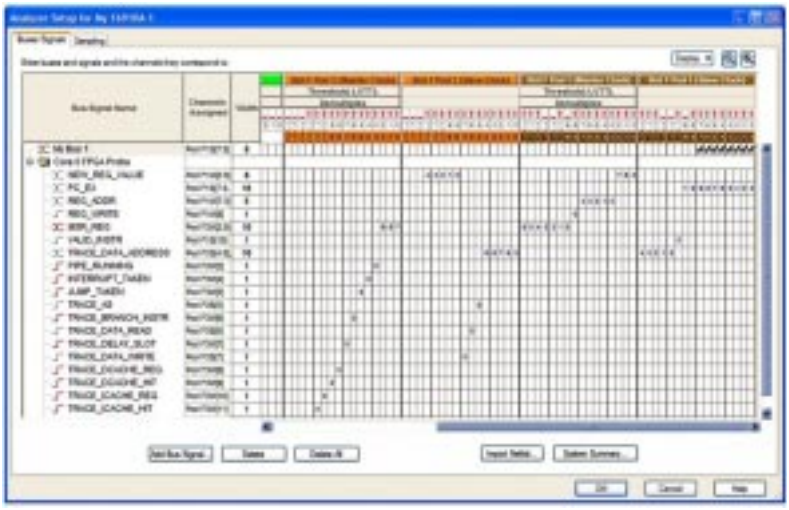


Figura 5. Asignaciones automáticas de canales y POD del analizador lógico para el trazado de MicroBlaze

sonda dinámica FPGA y el kit de herramientas de trazado de MicroBlaze guiaron al diseñador por distintos pasos para 1) conectar con la FPGA, 2) programarla con un diseño de instrumentación, 3) importar nombres de bus/señales y 4) realizar una asignación automática de patillas FPGA.

La etapa de asignación de patillas conmuta sistemáticamente cada una de las patillas de depuración de FPGA de modo que el analizador lógico pueda explorar todos sus canales para encontrar cada señal y, a continuación, realizar un seguimiento de dicha asignación. Con esta información, los canales del analizador lógico pueden configurarse automáticamente para el trazado de MicroBlaze, tal como se muestra en la figura 5. Esto permite al diseñador ahorrar mucho tiempo de configuración y eliminar la posibilidad de errores cuando se utiliza un proceso manual.

Vinculación de medidas de memoria externa

A continuación se utilizó una metodología interesante, que se describe en la figura 6, en la que las escrituras y lecturas de la memoria se realizaron a través de un sencillo programa que se ejecutaba en el MicroBlaze, y que en última instancia llevó a tomar medidas de temporización de hardware en la memoria. En primer lugar se realizó en el analizador lógico un seguimiento del flujo del programa, incluidos los comandos de lectura y escritura en memoria. El analizador lógico podía entonces buscar hacia adelante a partir de un comando de escritura o lectura para encontrar las escrituras y lecturas de memoria en el hardware. El disparo cuidadoso en las líneas de control de la memoria externa (RAS, CAS, WE) permitió al analizador lógico capturar medidas de temporización fina entre líneas de control y señales de datos (DQ, DQS, DDR_Clock).

A fin de realizar un seguimiento de la ejecución del MicroBlaze, así como de la temporización de la memoria externa, se "dividió" el analizador lógico en dos analizadores independientes, una función exclusiva de los analizadores lógicos de Agilent. Un analizador se vinculó al reloj del MicroBlaze, mientras que el otro generó su propio reloj en modo Timing.

Se conectaron sondas de terminales móviles de alto rendimiento a la memoria DDR. Con esta conexión, el analizador lógico podía buscar un valor de CONTROL de x04 (RAS=H, CAS=L, WE=L) para una escritura y de x05 para una lectura.

Trazado de la escritura de memoria en los niveles de ensamblaje, fuente correlacionada y temporización de hardware

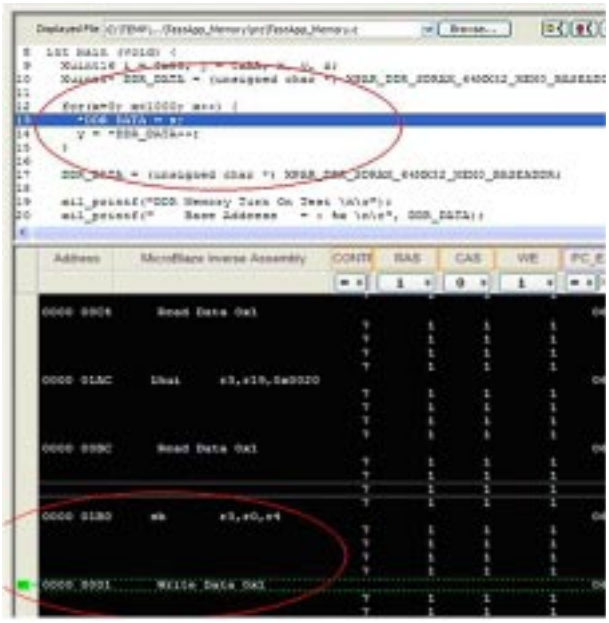


Figura 7. Comando de escritura de MicroBlaze visto en flujo de programa en los niveles de ensamblaje y fuente

En primer lugar, se vio una escritura de memoria en los niveles de ensamblaje y fuente, donde la señal TRACE_DATA_WRITE de MicroBlaze experimentó una subida después del reinicio del sistema. El programa ejecutado por el MicroBlaze escribió valores sucesivos en ubicaciones de memoria sucesivas. El caché debía desactivarse para forzar las escrituras y lecturas de la memoria externa. La figura 7 muestra el flujo del programa con el procesador MicroBlaze después de recorrer el código fuente utilizando la flecha verde hacia abajo en la interfaz del analizador lógico hasta ver una escritura de memoria.

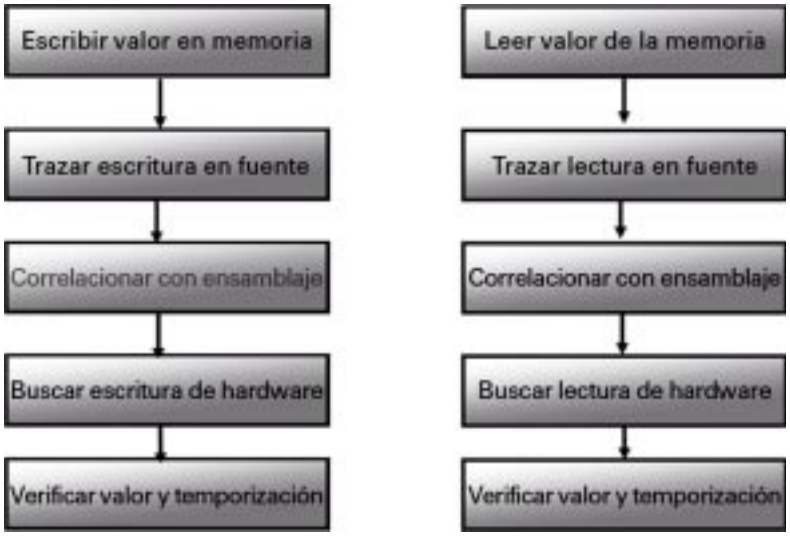


Figura 6. Proceso de validación de escrituras (a la memoria DDR) y lecturas correctas (desde la memoria DDR)

Figura 8. Valor de escritura de memoria del analizador de temporización de 4 GHz y validación de temporización

En este punto podría realizarse una búsqueda en los datos de línea del controlador de memoria externa para la siguiente condición de escritura (CONTROL = x04). El resultado de esta búsqueda indicaba con exactitud al analizador lógico el momento correcto para clasificar la temporización de memoria y verificar que era correcta. En la figura 8 se muestra el trazado de una escritura de memoria en hardware, con medidas específicas de validación de temporización de escritura. Cabe destacar que en el flanco de subida de DQS y CLK, los bits de datos eran estables, y que se escribió un valor de "1" en la memoria. Se empleó un proceso similar para comprobar si una escritura de memoria tenía los datos y la temporización correctos.

Verificación del flujo del programa con el puerto de barrido PPC 405 de IBM

Se obtuvo un escenario sorprendentemente distinto con un sistema que incorporaba un núcleo de procesador IBM 405 integrado y en el que se deseaba realizar un trazado básico para verificar el flujo del programa. El sistema incorporaba un puerto de barrido para el 405, lo que permitía realizar un seguimiento del flujo del programa midiendo sólo unas pocas patillas en la FPGA. En el ejemplo anterior con el procesador integrado MicroBlaze se medía y podía acceder a toda la información de control, dirección, datos y registro. Aquí sólo se accedía a información parcial de control y dirección a través del puerto de barrido, por lo que se hizo necesario realizar una "reconstrucción" significativa del trazado.

Figura 9. Proceso para la descodificación de puertos de barrido en el procesador IBM 405 integrado

Reconstrucción del trazado para dar sentido al "sinsentido"

Se necesitaba un proceso, que se describe en la figura 9, para determinar el flujo del programa. Un ensamblador inverso del analizador lógico puede reconstruir el trazado mediante la combinación de información. Existen varios requisitos, como conocer una dirección inicial para el programa que se está ejecutando, tener acceso a los archivos fuente y al archivo ejecutable (.elf) y obtener mensajes ocasionales de trazado de ramificación. Estos mensajes de trazado de ramificación ofrecen pistas sobre cómo se tomaron las ramificaciones del programa. Por ejemplo, si se toma una ramificación directa, el ensamblador inverso deberá buscar en el archivo .elf para comprobar adónde lleva esa ramificación.

Existe una diferencia sutil entre este trazado del 405 y el del MicroBlaze. Resulta difícil establecer una correlación temporal entre el trazado del flujo del programa del 405 y cualquier trazado externo, como el obtenido al sondear memoria externa. Si el caché está activado, no existe en realidad una correlación garantizada entre el trazado reconstruido del 405 y dichas medidas externas. Con el caché desactivado, la correlación se consigue a un nivel similar al que se obtiene con el MicroBlaze.

Resumen

El uso de núcleos de medida junto con analizadores lógicos puede resultar muy útil para validar y depurar los modernos sistemas digitales, incluido:

1) depuración general de sistemas de memoria o lógicos FPGA

2) análisis del flujo del programa en procesadores MicroBlaze integrados, o

3) análisis sofisticados de actividad interna de FPGA, correlacionada temporalmente con la operación del sistema externo.

Asimismo, si se utilizan procesadores IBM 405 integrados, también puede verse el flujo del programa con fines de validación mediante el uso de un puerto de barrido y software de análisis en un analizador lógico para reconstruir la actividad del procesador. Se necesitan algunas patillas FPGA y una cierta planificación para disparar o buscar mejor los datos de interés. No obstante, al final pueden ahorrarse muchas horas de desarrollo gracias a estos métodos de depuración.

Se facilitan explicaciones más detalladas de estos ejemplos de depuración en una reciente serie de cuatro artículos sobre lógica programable de DesignLine que se puede consultar en: http://www.pldesignline.com/news/191901655

Acerca del autor

Brad Frieden es ingeniero de desarrollo de aplicaciones en Agilent Technologies, especializado en depuración de FPGA con analizadores lógicos. En 1980 se licenció en ingeniería eléctrica (BSEE) por la Texas Tech University, y en 1991 obtuvo un máster en ingeniería eléctrica por la Universidad ee Texas en Austin.