

Diseño de un módulo IP para controlar dispositivos DDS utilizando un núcleo de interfaz PCI

N. R. León¹, A. Martínez²

¹ Gran Kaiman Teleco s.a. (GKT s.a.), Departamento de Investigación y Desarrollo, nelia.rosa@gkt.cu

² LACETEL, Departamento de Investigación y Desarrollo, abdel@lacetel.cu

RESUMEN / ABSTRACT

El presente trabajo constituye una etapa del proyecto de integración y desarrollo nacional de dispositivos basados en tecnología de Síntesis Digital Directa (SDD) que se lleva a cabo en LACETEL[®]. Se logra un diseño en lenguaje de descripción de hardware (VHDL), capaz de conectar estos dispositivos al bus PCI, para intercambiar información a través de la PC. Se explican los bloques funcionales del diseño, y su relación con el núcleo de interfaz con el bus PCI generado con herramientas del fabricante Xilinx. Finalmente, se valida el diseño mediante simulación.

Palabras claves: FPGA, PCI, SDD, VHDL, Xilinx

IP module design for controlling DDS devices using a PCI core

The present paper is a stage in the project of national integration and development of devices based on Direct Digital Synthesis (DDS) technology, which is taking place in LACETEL[®]. A hardware description language (VHDL) design is achieved, what is capable of connecting those devices to PCI bus, to interchange information through the PC. Functional blocks that shape the design are explained, and how the design links with the PCI interface core generated with Xilinx tools. Finally, the design is validated through simulation.

Key words: DDS, FPGA, PCI, VHDL, Xilinx

INTRODUCCIÓN

En diversos equipos de Telecomunicaciones, se necesita producir y controlar distintas formas de onda en diferentes frecuencias, para lo que se utilizan generadores de funciones; entre ellos se destacan los Sintetizadores Digitales Directos (DDS, siglas en inglés). LACETEL[®], Instituto de Investigación y Desarrollo de Telecomunicaciones, ha desarrollado un DDS, módulo de propiedad intelectual (IP, siglas en inglés), denominado LCT3212B. Existe la necesidad de conectar este dispositivo y otros, como el AD9858 y el AD9912 (diseñados por otro fabricante), a cualquier computadora y controlar desde la misma de manera fácil y flexible algunas de sus funciones.

Para realizar esta conexión, es más factible la utilización del bus PCI (Peripheral Component Interconnect), que es un estándar industrial, independiente del tipo y velocidad del procesador, y se acomoda a los requerimientos de muchos sistemas y arquitecturas.^{1,2}

El diseño que se presenta está basado en dispositivos lógicos programables y el uso del lenguaje de descripción de hardware

VHDL. Específicamente se escogió la plataforma FPGA (Field Programmable Gate Array) Virtex-5 de Xilinx y la herramienta Xilinx ISE[™] 10.1. Para el desarrollo del núcleo de interfaz con el bus PCI se eligió la herramienta de diseño CORE Generator[™].

Teniendo en cuenta lo anterior, los objetivos de este trabajo son: desarrollar un módulo de control de bus serie para los dispositivos: LCT3212B, AD9858 y AD9912 e integrar el módulo diseñado al núcleo de interfaz PCI de Xilinx.

GENERALIDADES DEL DISEÑO.

Para establecer la comunicación del usuario con los dispositivos de tecnología DDS: LCT3212B, AD9858 y AD9912, a través de la PC, se necesita un software gráfico, que constituya la interfaz de comunicación. Este es el punto de partida para el diseño, y se recomienda que se implemente con posterioridad.

Se requiere, además, la utilización de un módulo que cumpla con los protocolos del bus PCI, núcleo que demanda exigencias muy estrictas de alto grado de dificultad, por lo que se utilizó la herramienta de diseño CORE Generator[™], que

permite implementar módulos IP optimizados de acuerdo a las diversas estructuras de los FPGAs de Xilinx. El núcleo generado lleva como nombre “32-Bit Initiator/Target v4 for PCI™” y fue configurado de acuerdo a las exigencias de la aplicación que le será conectada (el módulo de control diseñado). En la figura 1 se muestra dicha conexión.¹⁻⁴

El módulo IP de control de bus serie que se diseñó, es capaz de comunicarse con este núcleo de interfaz PCI y, a la vez, con los circuitos integrados antes mencionados.

Los dispositivos DDS funcionan como destinatarios, ya que se rigen por una instrucción previa enviada desde software, que indica si la operación es lectura o escritura y a qué registro se desea acceder; por esta razón se implementa la opción de “Solo Target”, que ofrece este núcleo PCI. Por otra parte, la comunicación entre los dispositivos y el controlador diseñado es serie, y mucho más lenta que la transmisión por el bus PCI; por lo tanto, basta con utilizar transferencias simples de 32 bits. Se eligió una velocidad de 66 MHz, por la tendencia de las computadoras actuales a aumentar la velocidad.⁵⁻⁹

DESCRIPCIÓN DE LA ARQUITECTURA

El diseño está basado en el lenguaje de descripción de hardware VHDL y sintetizado utilizando un FPGA de la familia Virtex-5 (el integrado XC5VLX50T).¹⁰

La figura 2 refleja la interfaz externa y los bloques funcionales de la arquitectura desarrollada; a la izquierda y arriba se observan las principales señales que se conectan al núcleo de interfaz PCI. El resto de las señales externas son las de comunicación con los DDS; se representan, además, las principales señales internas.

1) Bloque 1: Decodificación de la interfaz

El diseño monitorea las salidas provenientes de la interfaz del núcleo para responder a transacciones del target, ya que las señales utilizadas en estas transacciones están activas y disponibles en diferentes momentos.

La señal más importante es BASE_HIT(0) que indica que la interfaz del núcleo ha clamado la transacción del PCI actual, en ese ciclo se registra el sentido de la transacción (dada por S_WRDN), que se mantendrá válido mientras la máquina de estados del target esté en el estado S_DATA.^{3,4}

2) Bloque 2: Identificación de la dirección base

Para decodificar cualquier transferencia, dentro del espacio del módulo IP de control de bus serie, se necesita conocer la dirección base asignada por el BIOS (Basic Input Output System) en el arranque del sistema, la cual se almacena en el Registro de Direcciones Base 0 del espacio de configuración^{1,2}. Se decidió que el software de interfaz con el usuario haga una lectura a este espacio y su primer acceso al módulo de control de bus serie (después del arranque del sistema y cada vez que se reinicie) será una transición de escritura a la dirección base, y el dato corresponderá a un código inicial que indique la cantidad y el tipo de dispositivos DDS conectados, de manera tal que se puedan mapear en el espacio de memoria

y que exista coordinación entre el software y el módulo IP de control.

Por lo tanto, cada vez que se reinicie el sistema, el Bloque 2 almacena el valor del bus ADIO [31..0] como dirección base y en el próximo ciclo como el código inicial.

El código en cuestión comprende cuatro bytes, cada uno de los cuales porta en los tres bits menos significativos el tipo de dispositivo, y en los restantes cinco bits, la cantidad de dispositivos conectados de dicho tipo. De esta forma, el diseño puede soportar un total de 124 dispositivos, divididos en cuatro tipos diferentes, aunque por el momento se diseñó para tres (Tabla 1).

3) Bloque 3: Decodificación dentro del target

Este bloque realiza las siguientes funciones:

- Control de transacciones: A partir de la combinación de las señales S_TERM y S_READY, se permiten los modos: Normal, Espera, Desconexión sin datos (Reintentar) y Desconexión con datos.
- Conteo de latencia interna: La transacción no puede durar más de 16 ciclos de reloj (por exigencias del bus PCI). El arranque del contador se determina a partir de la señal ADDR_VLD. Si se vence, se activa el modo Reintentar, de lo contrario, se permite la lectura o escritura.
- Desplazamiento de registros: El módulo diseñado, reserva “buffers” internos (memorias FIFO de dos niveles) para almacenar los datos provenientes del PCI, cuya transferencia es muy rápida. Cuando el bloque transmisor termina de enviar una secuencia serie, activa una bandera para el desplazamiento de estos “buffers”.
- Chequeo de que la dirección pertenece al rango asignado al diseño: Mientras ADDR_VLD esté activa se chequea el bus ADIO, en caso negativo, la transferencia se ignora pues pertenece a otro dispositivo.
- Chequeo de que la dirección corresponde a uno de los dispositivos mapeados. En caso negativo, se activa el modo Desconexión con datos y se almacena una codificación, de manera tal que el software pueda realizar una lectura a la dirección base para saber el estado de error debido al direccionamiento.
- Decodificar la dirección correspondiente a una de las funciones dentro del dispositivo: Se lleva a cabo durante la fase de datos, independientemente del tipo de DDS, según el sentido de la transferencia, la validez del dato, el espacio en los “buffers” y el contador de latencia internos. Para el mapeo dentro del módulo IP se utilizan los siguientes bits de dirección:
 - ADDR [31..12] codifican el rango asignado al módulo IP
 - ADDR [11..10] identifican el byte dentro del código inicial, permitiendo conocer el tipo de dispositivo al que se accede.
 - ADDR [9..5] identifican el número del dispositivo, dentro del tipo identificado.

- ADDR [4..2] identifica el acceso a una función dentro del dispositivo (**¡Error! No se encuentra el origen de la referencia.**2).
- ADDR [1..0] no pertenece a la decodificación (por especificaciones del PCI).¹⁻⁴

4) Bloque 4: Generador de reloj serie

Con el objetivo de generar un reloj de comunicación serie, soportado por los tres tipos de dispositivos que maneja este módulo, se decidió que tuviera aproximadamente una frecuencia de 153600 Hz, que es la máxima que exige el LCT3212B. Esta se obtiene a partir del reloj del PCI, utilizando un divisor por 430.⁷⁻⁹

Este reloj (dds_clk) es común a todos los dispositivos y solo se activa mientras haya transferencia hacia/desde éstos.

Los Bloques 5 y 6 se sincronizan con las transiciones del reloj clk_serie, que tiene la misma frecuencia y está activo siempre.

5) Bloque 5: Transmisor serie

El transmisor se encarga de activar el pin apropiado de selección del dispositivo a través de dds_cs [123..0] (“chip select”), de habilitar la salida del reloj serie (dds_clk), de cargar los bits adecuados para la conversión paralelo-serie, la cual se implementa en este bloque utilizando un registro de desplazamiento y los datos serie se envían por la señal dds_datos. Se garantiza que al terminar cada transmisión la línea permanezca en tercer estado.

La comunicación serie ocurre en dos fases. La primera es un ciclo de instrucción (comando), que consiste de una palabra donde el bit más significativo (MSB, siglas en inglés) es una bandera que indica si la operación es de lectura o escritura, y un determinado número de bits (en dependencia del tipo de dispositivo) indican la dirección dentro del mapa de registros. En la segunda fase, se encamina la información hacia/desde el registro direccionado.

Por lo tanto, el Bloque 5 verifica si existe almacenada alguna instrucción para enviar a un dispositivo, diferencia el tipo de dispositivo al que se transmitirá y si la segunda fase es de lectura o escritura. Comprueba que la dirección que porta la instrucción esté definida dentro del mapa de registros del dispositivo y habilita la recepción cada vez que se espere un dato (con la señal detec_Rx).

Los dispositivos LCT3212B soportan comunicación RS232, por lo que se utiliza un formato de caracteres de 8 bits, más un bit de arranque y uno de parada (Figura 3), donde b1 es el menos significativo. Para facilitar la detección de la caída de nivel en la línea se envía un bit en ‘1’ antes del bit de arranque.

Al enviar una instrucción o dato a este dispositivo, se debe esperar por la recepción de una palabra de control, en dependencia del tipo de transferencia.

El AD9858, requiere de transmisión SPI. En caso de escritura, se envía la instrucción seguida de un ciclo en el que se desactiva el reloj (separación entre instrucciones y datos) y la línea de datos serie se pone en tercer estado, a continuación se envía el dato, cuyo número de bytes depende del registro direccionado. En caso de lectura se envía la instrucción, se deja la línea en tercer estado desactivando el reloj durante un

ciclo, y luego se espera la recepción del dato solicitado. Siempre se comienza por el MSB.

El AD9912 requiere de una instrucción de 16 bits, donde los bits 15 y 14 indican la longitud en bytes del dato que se va a enviar/recibir (sin contar la instrucción). La transmisión es SPI, ocurre igual al AD9858, excepto que no se requiere del ciclo en el que se desactiva el reloj. Además, comienza siempre por el MSB. En caso de que los bits de longitud indiquen una transferencia mayor de 3 bytes, sólo pueden transferirse 4 bytes, excepto cuando el registro accedido sea el de FTW0 (Frequency Tuning Word 0) que serán 6 bytes.

Cuando se escribe en el registro de dirección 00H de cualquiera de estos dos últimos tipos de dispositivos, se garantiza que no se varíe el bit que indica por defecto la transferencia comenzando por el MSB, ni el que indica que la comunicación serie es bidireccional por un solo pin.

Al finalizar todo el proceso de transmisión se desactiva el reloj serie y el “chip select” (dds_cs [123..0]), y se varían las banderas de instrucción y datos almacenados.⁷⁻⁹

6) Bloque 6: Receptor serie

Cuando el transmisor, después de enviar una instrucción, indica que se espera la recepción de datos (mediante detec_Rx), el receptor comienza a recibir de la línea dds_datos, tantos bytes como indique la instrucción, en dependencia del tipo de dispositivo (en caso de ser LCT3212B primero se debe detectar el inicio de recepción), como se explicó anteriormente. Luego de recibir todos los datos, se convierten de serie a paralelo, se almacenan, y se indica que la recepción fue válida con Rx_vld.

En caso de haberse completado el dato correspondiente a un registro, podrá ser leído desde software, y después de esa lectura se desactiva la bandera de validez del dato recibido (Rx_vld).

Los datos provenientes de los dispositivos se leen con el frente de caída del reloj serie. El orden de recepción de bits coincide con el explicado en el transmisor.

7) Bloque 7: Generador de otras señales

La señal dds_rst es una salida de “reset” global, común para todos los DDS conectados, que se activa con el nivel alto de RST (proveniente del núcleo PCI).

Existen señales que exige el núcleo de interfaz PCI como entradas y que no fueron utilizadas en los bloques anteriores, pues corresponden a otras funcionalidades (por ejemplo de un máster, de configuración dinámica o de interrupción), y que requirieron un tratamiento diferenciado en este bloque.^{3,4}

RESULTADOS Y DISCUSIÓN

Una vez finalizada la parte del diseño correspondiente a la programación, mediante el uso del lenguaje VHDL, se pasó a la comprobación del código, para lo que se utilizó la herramienta ModelSim XE III/Starter 6.2g, con el cual se depuraron y corrigieron los errores cometidos durante el diseño.

Las siguientes figuras muestran una secuencia de transiciones utilizadas como ejemplo. A la izquierda se observa el nombre de las principales señales y el valor que toman en el instante que indica el cursor amarillo.

En la figura 4 se observa una escritura en la dirección base “20000000H” del código inicial “001E113CH”, que significa que están conectados siete AD9858, dos LCT3212B y tres AD9912.

La próxima transición es una escritura en la dirección “20000064H” (que es la de instrucción del cuarto AD9858 conectado) del dato “00000088H”, que corresponde a la lectura del registro POW2 (Phase Offset Word 2), como se refleja en la figura 5. A partir de este momento se activa el reloj serie (dds_clk), se activa el “chip select” correspondiente (dds_cs) y comienza a enviarse serie esta instrucción (en dds_datos). Luego se observa el ciclo en que se desactiva dds_clk y se mantiene dds_datos en tercer estado hasta que el DDS envíe el dato solicitado.

Al terminar la recepción, es posible leer el dato desde software, lo cual tiene lugar leyendo en la dirección de registro del cuarto AD9858 conectado: “20000068H” (Figura 6). Puede comprobarse que el dato leído coincide con el que se obtiene serie.

CONCLUSIONES

Se integraron los conocimientos adquiridos durante el estudio del bus PCI, la tecnología DDS, los dispositivos integrados LCT3212B, AD9858 y AD9912 y el diseño de sistemas digitales utilizando dispositivos lógicos programables y lenguajes de descripción de hardware, en especial, la plataforma de FPGAs Virtex-5 de Xilinx y el VHDL.

Este estudio concluyó con el diseño de un módulo IP de control de bus serie para dispositivos DDS, dando posibilidades a la conexión de un gran número de dispositivos (124), consumiendo sólo un 26% de los recursos disponibles en el FPGA propuesto (XC5VLX50T).

La utilización del módulo IP de Xilinx para la interfaz con el bus PCI, permitió una mayor agilidad y eficiencia en el trabajo.

Se aprovecharon las similitudes entre los integrados DDS propuestos, para simplificar la estructura del diseño.

Se verificó la validez del diseño mediante la simulación con la herramienta ModelSim.

REFERENCIAS

1. **ABBOTT, D.:** “PCI Bus Demystified”. s.l. : *LLH Technology Publishing*, 2000.
2. **PCI SPECIAL INTEREST GROUP:** “PCI Local Bus Specification”. Revision 2.2, 1998.
3. **XILINX INC.:** “LogiCORE™ IP Initiator/Target v3.166 for PCI™ User Guide”, Septiembre de 2008, disponible en: <http://www.xilinx.com>.

4. **XILINX INC.:** “LogiCORE™ IP Initiator/Target v3.2 for PCI™ Getting Started Guide”, Septiembre 19, 2008, disponible en: <http://www.xilinx.com>.
5. **ANALOG DEVICES:** “A Technical Tutorial on Digital Signal Synthesis”, 1999, disponible en: <http://www.analog.com>.
6. **GOLDBERG, B.G.:** “Digital Frequency Synthesis Demystified”. s.l. : *LLH Technology Publishing*, 1999.
7. **LACETEL:** “PD-DDS IP Module LCT3212B VHDL. Soft-Intellectual Property Module”, 2009, disponible en: <http://www.lacetel.cu>
8. **ANALOG DEVICES:** “AD9858 1 GSPS Direct Digital Synthesizer”, 2003, disponible en: <http://www.analog.com>
9. **ANALOG DEVICES:** “AD9912 1GSPS Direct Digital Synthesizer with 14-Bit DAC”, 2007-2008, disponible en: <http://www.analog.com>.
10. **XILINX INC.:** “Virtex-5 Family Overview LX, LXT, and SXT Platforms (v3.0)”, Febrero de 2007, disponible en: <http://www.xilinx.com>

AUTORES

Nelia Rosa León González, Ingeniera en Telecomunicaciones y Electrónica. Labora como parte del Departamento de Investigación y Desarrollo de la empresa Gran Kaiman Teleco s.a. (GKT s.a.) y realiza soporte técnico a equipos de NGN de Huawei en esa misma empresa (Teléfono: 2069028 al 29).

Dirección postal: Villanueva # 571 apto. 1 e/ Santa Ana y Santa Felicia, Luyanó, 10 de Octubre. Ciudad de la Habana.

Abdel Martínez Alonso, Ingeniero en Telecomunicaciones y Electrónica, Investigador. Labora como parte del Departamento de I+D de LACETEL, Instituto de Investigación y Desarrollo de Telecomunicaciones (Teléfono: 6832824).

Dirección postal: Cervantes # 38 e/ Pinar del Río y Yumurí. Arroyo Naranjo. Ciudad de la Habana.

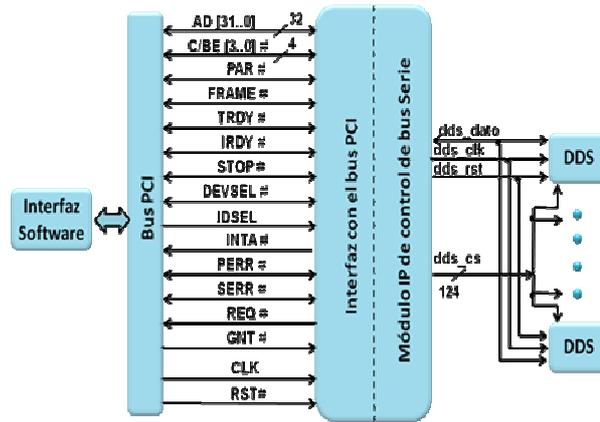


Figura 1: Esquema general de conexión de los DDS a la PC

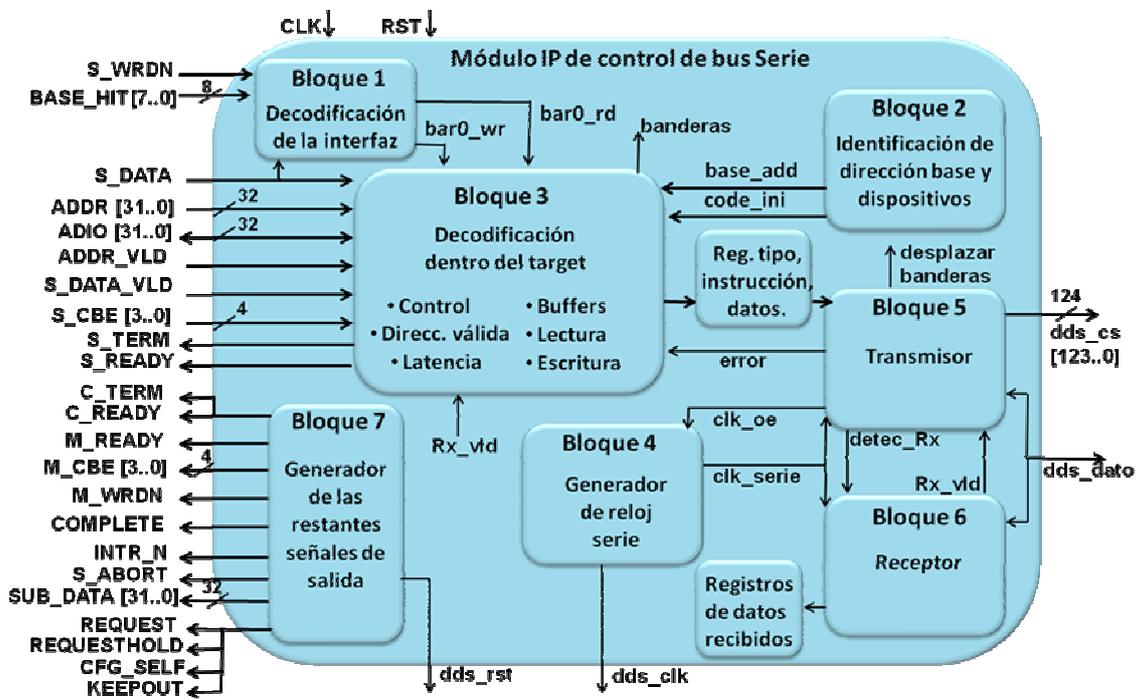


Figura 2: Interfaz externa y bloques funcionales del diseño.



Figura 3: Formato de comunicación del LCT3212B

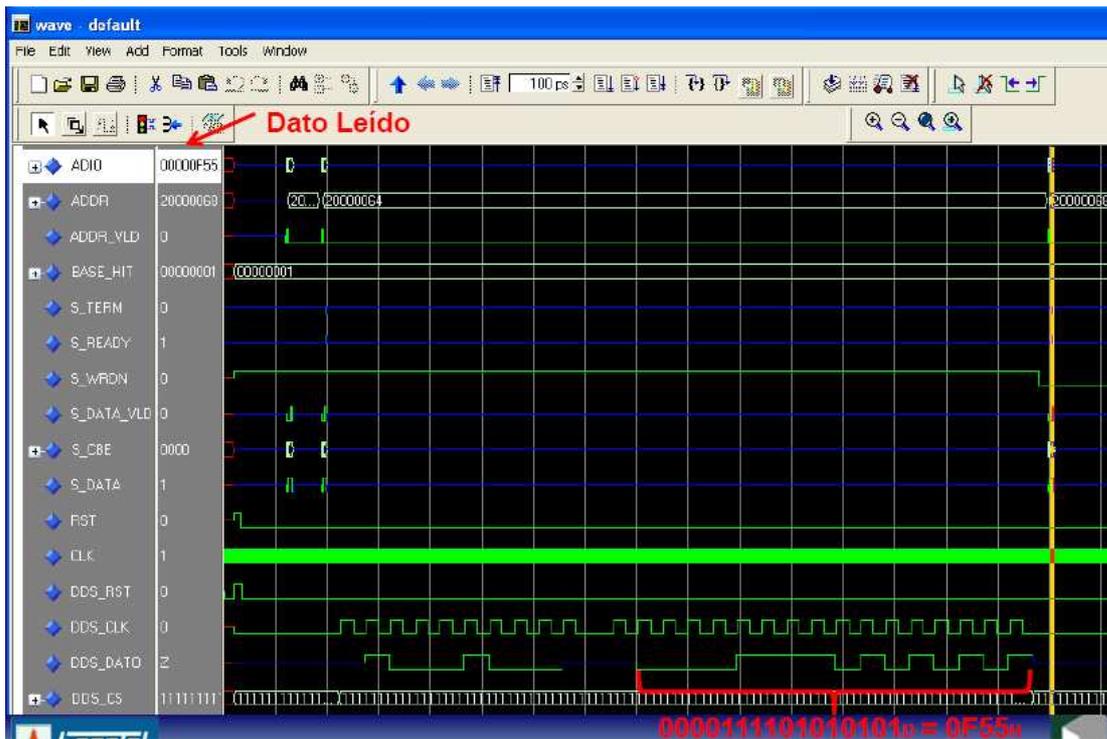


Figura 6: Lectura del registro del 4to AD9858

Tabla 1 Codificación del tipo de dispositivo.

Código	001	100	110
Tipo	LCT3212B	AD9858	AD9912

Tabla 2: Codificación dentro del dispositivo

ADD[4-2]	Función
000	Reservado para uso futuro.
001	Instrucción (8/16 bits)(escritura)
010	Registro (< 32bits) (escritura/lectura)
011	Registro (> 32 bits) (AD9912)(e/l)
100	Reservado para uso futuro.
101	Reservado para uso futuro.
110	Error debido a transmisión (lectura)
111	Reservado para uso futuro.