SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

INSTITUTO TECNOLOGICO de la laguna

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

"Diseño de un módulo FPGA para aplicaciones de procesamiento digital de señales"

POR

Ing. Marcos Enrique Cid Chávez.

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

DIRECTOR DE TESIS
Dr. Francisco Gerardo Flores García.

CODIRECTOR DE TESIS
M.C. Juan Sifuentes Mijares

ISSN: 0188-9060

TECNOLÓGICO DE LA LAGUNA

RIITEC: (06)-TMCIE-2013

Torreón, Coahuila. México,
Julio 2013

Subsecretaría de Educación Superior
Dirección General de Educación Superior Tecnológica
Instituto Tecnológico de la Laguna

"2013, Año de la Lealtad Institucional y Centenario del Ejercito Mexicano"

Dependencia: **DEPI**
Oficio: **DEPIJ/184/2013**
Asunto: **Autorización de impresión de tesis.**

Torreón, Coah., 26/ Agosto/2013

C. MARCOS ENRIQUE CID CHAVEZ.
CANDIDATO AL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA.
PRESENTE

Después de haber sometido a revisión su trabajo de tesis titulado:

"Diseño de un módulo FPGA para aplicaciones de procesamiento digital de señales"

Habiendo cumplido con todas las indicaciones que el jurado revisor de tesis hizo, se le comunica que se le concede la autorización con número de registro **RIITEC: (06)-TMCIE-2013**, para que proceda a la impresión del mismo.

ATENTAMENTE

DR. JOSE LUIS MEZA MEDINA
Jefe de la División de Estudios de Posgrado e Investigación

ESTADOS UNIDOS MEXICANOS
SECRETARIA DE EDUCACIÓN PUBLICA
INSTITUTO TECNOLÓGICO
de la laguna
División de Estudios de Posgrado e Investigación

"2013, Año de la Lealtad Institucional y Centenario del Ejercito Mexicano"

Torreón, Coah., 26/Agosto /2013

DR. JOSE LUIS MEZA MEDINA
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
PRESENTE

Por medio de la presente, hacemos de su conocimiento que después de haber sometido a revisión el trabajo de tesis titulado:

"Diseño de un módulo FPGA para aplicaciones de procesamiento digital de señales"

Desarrollado por el C. **MARCOS ENRIQUE CID CHAVEZ**, con número de control **M05130840** y habiendo cumplido con todas las correcciones que se le indicaron, estamos de acuerdo que se le conceda la autorización de la fecha de examen de grado para que proceda a la impresión de la misma.

ATENTAMENTE

DR. FRANCISCO GERARDO FLORES GARCIA.
Asesor/Director

M.C JUAN SIFUENTES MIJARES.
Coasesor

DR. FRANCISCO VALDES PEREZGASGA.
Comité Tutorial

M.C. SERGIO SALAS HUERTA.
Comité Tutorial

Dedicatoria

A mis padres, Daniel y María del Carmen.

Agradecimientos

A mi asesor y tutor Dr. Francisco Gerardo Flores Garcíapor el apoyo y orientación que me ofreció, gracias a sus conocimientos y enseñanzas ha sido posible la realización de este proyecto.

A toda mi familia por haberme apoyado durante el transcurso de estos años.

Al Consejo Nacional de Ciencia y Tecnología por darme la oportunidad de continuar con mi preparación profesional y por el apoyo económico recibido.

Resumen

Los sistemas digitales requieren comunicarse con diversos dispositivos mediante la instrumentación electrónica para la medición y control de diferentes variables. Las formas de comunicación son muy variadas, sin embargo, la tecnología de los arreglos de compuertas programables en campo (FPGA) ofrece una solución rápida y adaptable a los requerimientos especificos del diseño.

Las redes inalámbricas de sensores son un ejemplo de estos sistemas que requieren comunicarse con sistemas digitales, representando una solución integral para el monitoreo de variables climáticas dentro del sector agrícola.

En esta tesis se propone un método para el diseño e implementación del estándar de comunicación*bus* Interfaz de Periféricos Seriales(SPI) utilizando la tecnología de arreglos de compuertas programables en campo y el lenguaje de descripción de hardware VHDL. Con este trabajo se sentaron las bases para establecer la transmisión de datos entre dispositivos electrónicos a través del estándar de comunicación SPI y se cumple el primer paso en el desarrollo de un módulo FPGA para aplicaciones de procesamiento digital de señales.

Palabras Clave: FPGA, VHDL, SPI.

Abstract

Digital systems require communicating with other devices using electronic instrumentation for the measurement and the control of several variables.These communications are varied. Field programmable gate arrays (FPGA) offer a quick and adaptable way to establish these interfaces tospecific requirements of a given design.

Wireless sensor networks are an exampleof devices requiring communication with digital systems and represent a complete solution for the monitoring climatic variables within the agricultural sector.

This thesis proposes a method for the design and implementation of the SPI communication standard using field programmable gate arrays and the hardware description language VHDL. This work is a base to establish data transmission between electronic devices using the SPI communication standard. Additionally, it represents a first step towards the development of a FPGA module for digital signal processing application.

Keywords: FPGA, VHDL, SPI.

Índice General

Índice de Figuras

Índice de Tablas

Nomenclatura

FPGA	Arreglo de Compuertas Programable en Campo
SPI	Interfaz de Periféricos Seriales
IOB	Bloque de Entrada/Salida
CLB	Bloque Lógico Configurable
RAM	Memoria de Acceso Aleatorio
DCM	Administrador de Reloj Digital
RISC	Computadora con Conjunto de Instrucciones Reducido
RTL	Nivel de Transferencia de Registros
UART	Transmisor-Receptor Asíncrono Universal
GPIO	Entradas / Salidas de Propósito General

Capítulo 1. Introducción

El acelerado desarrollo de las nuevas tecnologías de dispositivos electrónicos ha marcado un importante avance en todos los sectores de la sociedad. Áreas como la instrumentación electrónica para la medición y control de variables a través de sensores y sistemas remotos han sufrido grandes avances gracias a las bondades de las nuevas y eficientes tecnologías, permitiendo crear diseños especializados. En el pasado, la tecnología FPGA estaba disponible solamente para ingenieros con un profundo conocimiento del diseño de hardware digital. Sin embargo, el surgimiento de herramientas de diseño de alto nivel ha permitido que los dispositivos FPGA estén siendo cada vez más utilizados en diversas aplicaciones debido a su bajo costo, rendimiento, funcionalidad especializada, excepcional potencia de cómputo y capacidad de reprogramación [1].

El *bus* Interfaz de Periféricos Seriales (SPI por sus siglas en inglés) es un estándar de comunicaciones usado principalmente para la transferencia de información entre dispositivos electrónicos. El *bus* SPI se ha establecido como uno de los estándares de comunicación más utilizados en la actualidad debido a las ventajas que brinda respecto a la transmisión de datos de manera paralela. Tales ventajas incluyen la reducción del número de conductores que a su vez impacta de manera directa en el precio y tamaño de un sistema digital.

El gran avance que han tenido los dispositivos electrónicos así como los sistemas de comunicación hace posible contar con herramientas para el desarrollo de diseños complejos en circuitos de hardware digital. Es por esto que la utilizacióndel protocolo de comunicación SPI para la implementación de tecnología FPGA en diseños que requieren un alto grado de flexibilidad y capacidad de cómputo brinda un amplio campo de aplicación dentro de áreas como la instrumentación electrónica, incluyendo las redes inalámbricas de sensores;permitiendo extender los posibles usos de su arquitectura para otras funciones avanzadas.

Capítulo 2. Marco Teórico

2.1 Arreglos de Compuertas Programables en Campo

Un arreglo de compuertas programable en campo es un dispositivo de propósito general que contiene una matriz de bloques lógicos comunicados por conexiones programables por el usuario. La configuración de un FPGA por lo general se realiza mediante el lenguaje de descripción de hardware VHDL. Cuando se configura un FPGA, el circuito interno se conecta de tal manera que se implementa en la circuitería la aplicación de programación. A diferencia de los procesadores, los FPGA utilizan circuitos dedicadosal procesamiento lógico y no contienen un sistema operativo. Los FPGA funcionan de manera paralela, por lo que diferentes operaciones de procesamiento no interfieren entre sí para hacer uso de los mismos recursos, teniendo como resultado que el rendimiento de una parte de la aplicación no se vea afectado al añadir un procesamiento adicional, así como tener múltiples procesos que se pueden ejecutar a diferentes velocidades.

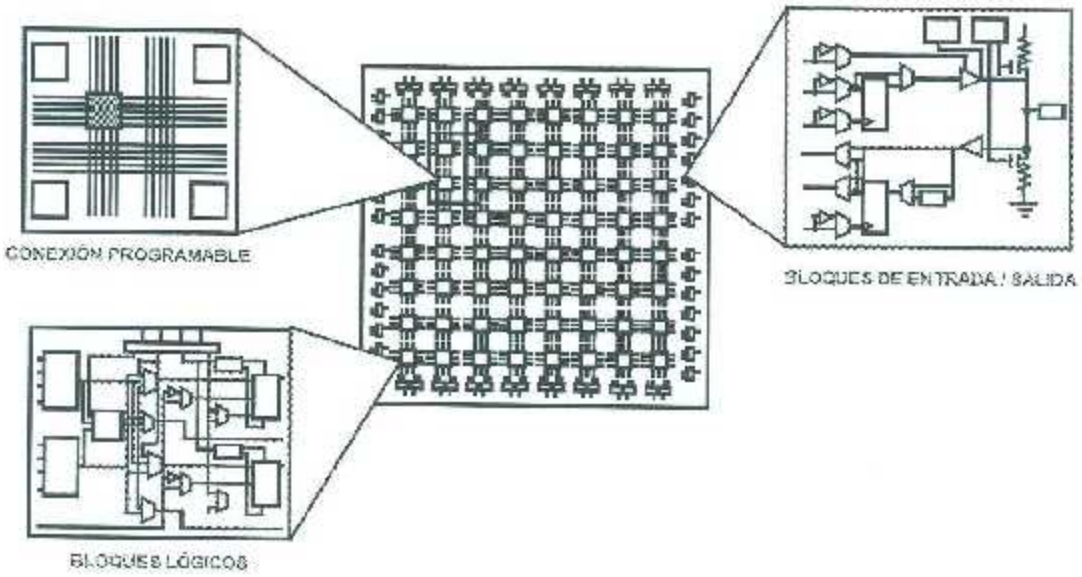


Figura 2.1 Estructura interna del FPGA.

El FPGA contiene bloques lógicos e interconexiones reconfigurables y le proporciona al usuario los medios para configurar la intersección entre los bloques de lógica así como la función de cada bloque lógico. Estos bloques lógicos se pueden configurar para realizar funciones que van desde simples compuertas lógicas como AND y XOR hasta funciones complejas como las que realiza un microprocesador.

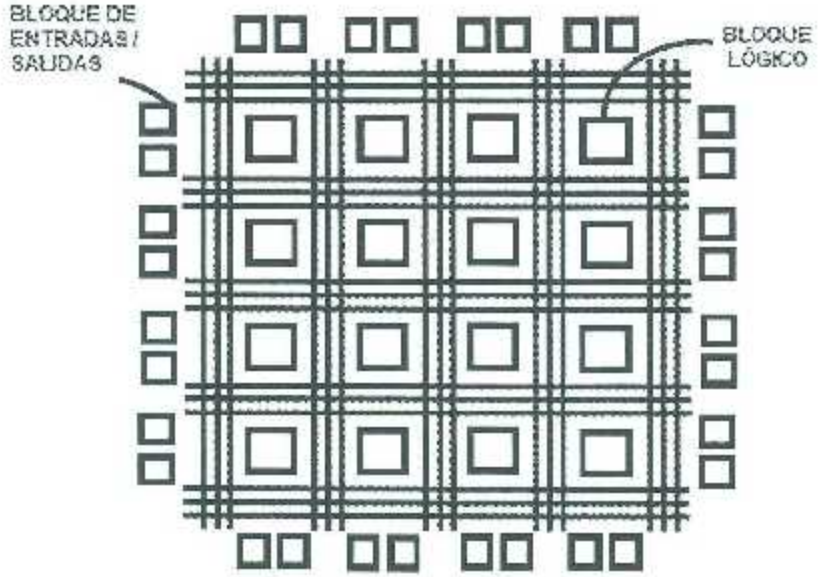


Figura 2.2 Estructura interna simplificada del FPGA

2.1.1 Arquitectura Interna del FPGA Spartan III

El tamaño, estructura, número de bloques y la cantidad y conectividad de las conexiones varían en las distintas arquitecturas. La arquitectura de las FPGA de la familia Spartan III de Xilinx consta de los siguientes elementos funcionales programables:

- **Bloques de Entrada/Salida (IOBs).**
- **Bloques Lógicos Configurables (CLBs).**
- **Bloques de Memoria RAM.**
- **Bloques de Multiplicación.**
- **Administradores de Reloj Digital (DCMs).**

Los elementos descritos están organizados como se muestran en la Figura 2.3. Los bloques de entrada y salida rodean los bloques lógicos configurables, entre este arreglo se encuentra una columna compuesta por varios bloques de memoria RAM de 18 KBits, cada uno asociado a un multiplicador dedicado. Los administradores de reloj digital se encuentran en los extremos de estas columnas.

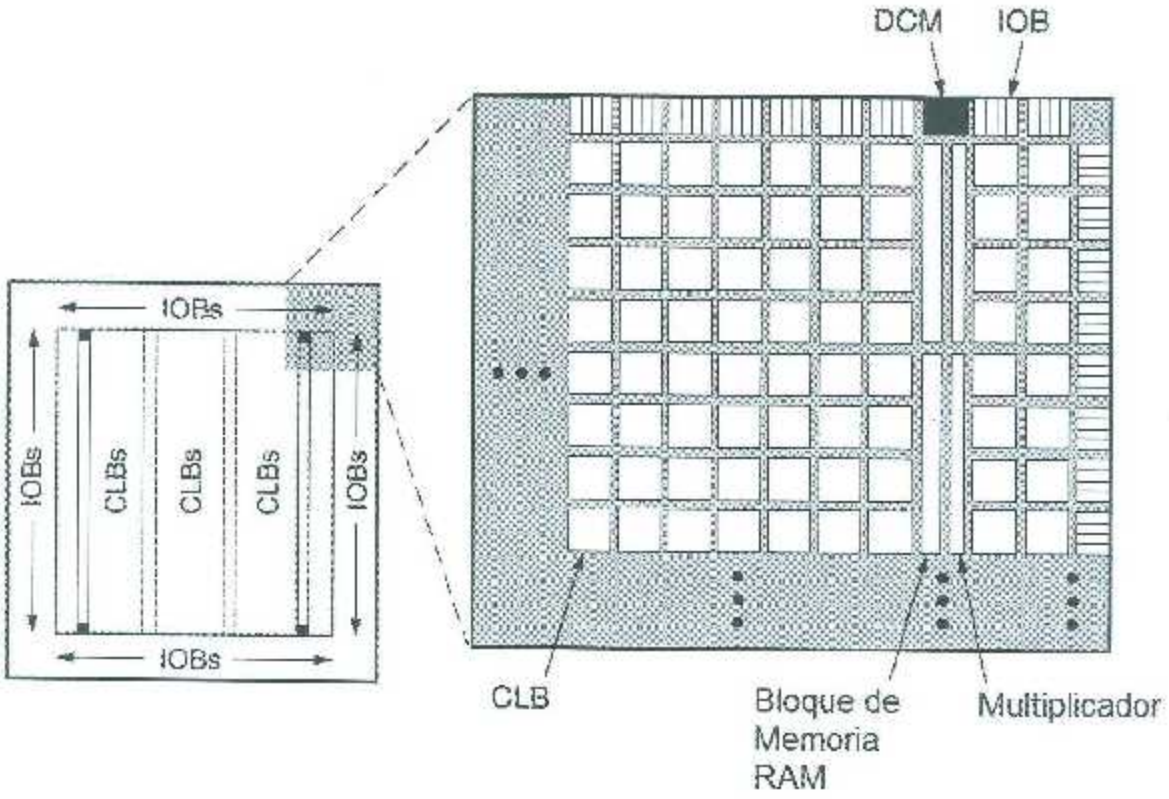


Figura 2.3 Arquitectura interna del FPGA Spartan III.

2.1.1.1 Bloques de Entrada/Salida

Proporcionan una interfaz bidireccional programable entre un pin de entrada / salida y la lógica interna de la FPGA.

2.1.1.2 Bloques Lógicos Configurables

El bloque básico de la red que compone la FPGA es la "rebanada". Cada CLB se compone de cuatro "rebanadas" interconectadas. Los CLBs constituyen el recurso lógico principal para implementar circuitos síncronos o

combinacionales.Las cuatro "rebanadas" de un CLB cuentan con los siguientes elementos en común: dos generadores de funciones lógicas, dos elementos de almacenamiento, multiplexores, lógica de acarreo y compuertas aritméticas. Los dos pares de "rebanadas" utilizan estos elementos para entregar funciones lógicas, aritméticas y de almacenamiento. Además un par soporta dos funciones adicionales: almacenamiento de datos usando RAM distribuida y corrimiento de datos con registros de 16 bits. El generador de funciones basado en RAM, también conocido como *Look-Up Table*(LUT), es el recurso principal para implementar funciones lógicas dentro del FPGA, que además puede ser configurado como RAM distribuida o como registro de corrimiento de 16 bits.

2.1.1.3 Bloques de Memoria RAM

Todos los dispositivos Spartan III cuentan con bloques de memoria RAM, organizados como bloques síncronos configurables de 18 Kbits. Los bloques de RAM almacenan grandes cantidades de datos de manera más eficiente que la RAM distribuida.

2.1.1.4 Bloques de Multiplicación Dedicados

La Spartan III cuenta con multiplicadores embebidos que aceptan dos palabras de 18 bits como entradas para entregar un producto de 36 bits. Los buses de entrada al multiplicador aceptan datos en complemento a dos (ya sea 18 bits con signo, o 17 bits sin signo). Para cada bloque de RAM existe un multiplicador conectado y colocado de manera próxima para permitir un manejo eficiente de los datos.

2.1.1.5 Administradores de Reloj Digital

La Spartan III tiene cuatro bloques para un control flexible y completo de:

- **Frecuencia.** Con una señal de reloj de entrada, el DCM puede generar un amplio rango de diferentes frecuencias de reloj de salida.
- **Fase.** El DCM proporciona la capacidad de cambiar la fase de todas las señales de reloj de salida con respecto a la señal de reloj de entrada [2].

2.1.2 Tarjeta de Desarrollo NEXYS 2

La Nexys 2 es una potente plataforma de diseño de sistemas digitales que utiliza una Xilinx Spartan 3E FPGA. Es una herramienta muy útil que permite la implementación de una amplia variedad de diseños sin la necesidad de componentes adicionales, así como para trabajar con procesadores embebidos como lo es Microblaze™ de Xilinx. La tarjeta de desarrollo Nexys 2 se compone de los siguientes elementos [3]:

- Xilinx Spartan-3E FPGA.
- Puerto USB2, para alimentación de la tarjeta, configuración del dispositivo y transferencia de datos a alta velocidad.
- Puerto PS/2.
- Puertos de datos VGA y RS232.
- Memoria SDRAM de 16 MB.
- Memoria Flash ROM de 16 MB.
- Oscilador de 50 MHz.
- 75 entradas/salidas de la FPGA conectadas a los puertos de expansión.
- 8 diodos emisores de luz (LED).
- 4 despliegues de siete segmentos.
- 4 botones.
- 8 interruptores deslizables.

2.1.3 Lenguaje de Descripción de Hardware VHDL

Es un lenguaje de descripción y modelado de hardwarediseñado para describir la funcionalidad y organización de sistemas digitales tales como los arreglos de compuertas programables en campo y circuitos integrados. Un lenguaje de descripción de hardware con VHDL es inherentemente paralelo, debido a que se constituye de comandos y procesos, los cuales corresponden a compuertas lógicas, a través de los cuales es posible ejecutar de manera paralela el procesamiento de los datos.

Un sistema digital puede ser descrito a través de los siguientes niveles de representación:

- Descripción de comportamiento.
- Descripción estructural.

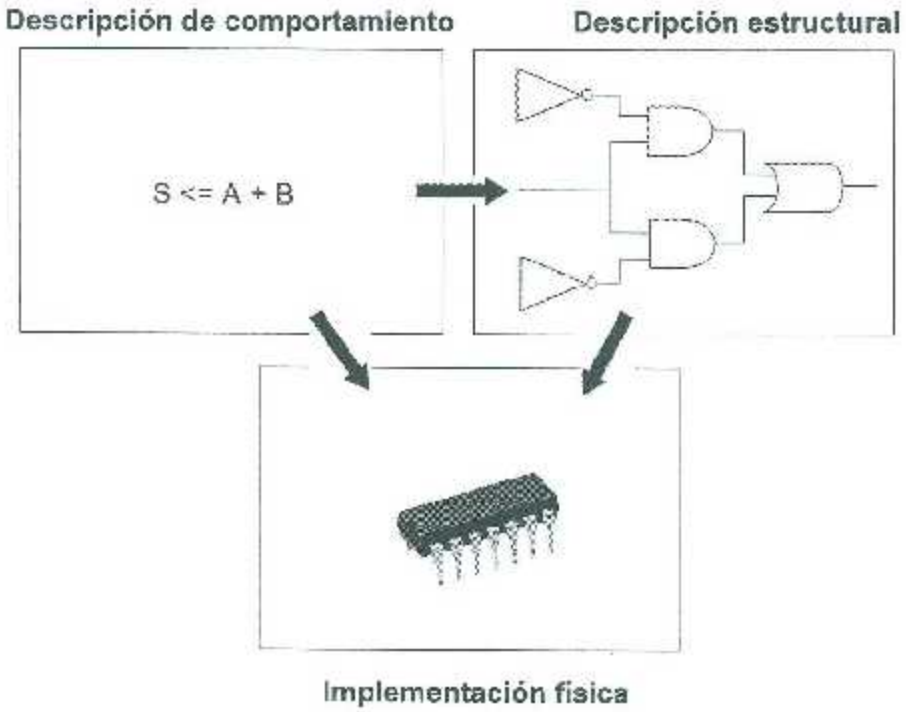


Figura 2.4 Niveles de representación.

El más alto nivel de representación es el de comportamiento, que describe el sistema en términos de cómo se comporta más que en términos de componentes y las interconexiones entre estos. Una descripción de comportamiento especifica la relación entre las señales de entrada y salida, la cual puede ir desde una expresión booleana hasta algoritmos complejos.

El nivel estructural describe un sistema como un arreglo de compuertas y componentes que son interconectados para llevar a cabo una función deseada. Una descripción estructural puede ser comparada a un esquemático de

compuertas lógicas interconectadas. Es una representación usualmente cercana a la realización física de un sistema digital.

VHDL permite describir un sistema digital a nivel estructural o de comportamiento. El nivel de comportamiento se puede dividir en dos diferentes tipos: flujo de datos y algorítmico. La representación a través del flujo de datos describe como se mueven los datos a través del sistema, esto es usualmente realizado en términos de flujo de datos entre registros. El modelo de flujo de datos hace uso de sentencias concurrentes que se ejecutan en paralelo tan pronto como el dato llega a la entrada. Por otro lado, las sentencias secuenciales se ejecutan en la secuencia en la cual fueron especificadas. VHDL permite asignaciones de señales tanto concurrentes como secuenciales que determinarán la manera en que serán ejecutadas.

Un sistema digital en VHDL consta de una entidad que puede contener otras entidades, que entonces son consideradas componentes de la entidad de nivel superior. Cada entidad se modela mediante la declaración de una entidad y un cuerpo de arquitectura. Se puede considerar la declaración de la entidad como la interfaz al exterior que define las señales de entrada y salida, mientras el cuerpo de la arquitectura contiene la descripción de la entidad y se compone de entidades, procesos y componentes interconectados, todos operando de manera concurrente.

VHDL utiliza palabras claves reservadas que no pueden ser utilizadas como nombres o identificadores. Para las palabras clave y los identificadores definidos por el usuario no se distinguen entre mayúsculas y minúsculas. Las líneas con comentarios comienzan con dos guiones adyacentes y serán ignoradas por el compilador. VHDL también ignora saltos de línea y espacios extra. En el lenguaje VHDL siempre se debe declarar el tipo de cada objeto que puede tener un valor, como señales, constantes y variables [4].

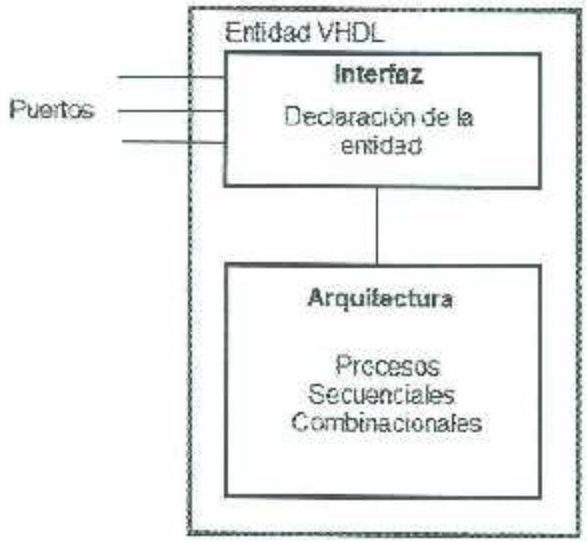


Figura 2.5 *Entidad VHDL compuesta por una interfaz y la descripción de la arquitectura.*

2.1.4Procesador embebido

El MicroBlaze es un microprocesador virtual que se construye mediante la combinación de bloques de código llamados núcleos dentro de un arreglo de compuertas programables en campo de Xilinx. La ventaja de este enfoque es que sólo se utilizan los recursos del microprocesador que el usuario especifique, por lo que es posible adaptarlo a las necesidades específicas de la aplicación.

El procesador MicroBlaze cuenta con una arquitectura de 32 bits tipo RISC Harvard [5], optimizada para su implementación en FPGAs de Xilinx con buses de instrucciones y de datos separados que funcionan a altas velocidades para la ejecución de programas y acceso a unidades de memoria al mismo tiempo.El MicroBlaze puede emitir una nueva instrucción cada ciclo, las cuales se ejecutan a través de un pipeline de tres etapas, cuenta con 32 registros de propósito general, con una unidad aritmética lógica y una unidad de corrimiento. El pipeline de MicroBlaze es de tipo paralelo, dividido en tres etapas: buscar, decodificar y ejecutar. Cada etapa toma un ciclo de reloj para completarse, por consecuencia, toma tres ciclos de reloj completar la instrucción. Cada etapa se activa en cada ciclo de reloj por lo que es posible la ejecución de tres instrucciones de manera simultánea.

El microprocesador MicroBlaze sigue el modelo de arquitectura Harvard, donde datos e instrucciones son almacenados en memorias diferentes. Debido a la capacidad de reconfiguración de los FPGAs, es posible configurar el sistema con diferentes opciones sobre los buses, pudiéndose reducir de este modo el tamaño final del sistema.

2.2Estándar de Comunicación SPI

El *bus*denominado Interfaz de Periféricos Seriales (SPI) es una interfaz de comunicación serial utilizada en diversos dispositivos electrónicos que opera en modo full dúplex, es decir, permite en envio y recepción de datos de manera simultánea, de manera síncrona y con una interfaz amo / esclavo donde el dispositivo amo inicializa la transferencia de datos (puede operar con un dispositivo amo y con uno o más dispositivos esclavos).

La sincronización y la transmisión de datos se realiza a través de las siguientes cuatro señales:

- SCLK: Esta señal es generada por el dispositivo amopara sincronizar la transferencia de datos entre el amo y el esclavo. Con cada pulso de reloj se lee o se envía un bit.
- MOSI: Salida de datos del amo y entrada de datos al esclavo.
- MISO: Salida de datos del esclavo y entrada de datos al amo.
- SS: Salida del amo a la terminal Chip Select (CS) del esclavo. Señal generada por el amo para seleccionar el esclavo que se activará. Señal de activación con nivel bajo.

De estas cuatro señales, MOSI y MISO son líneas de transmisión de datos y SCLK y SS son líneas de control.

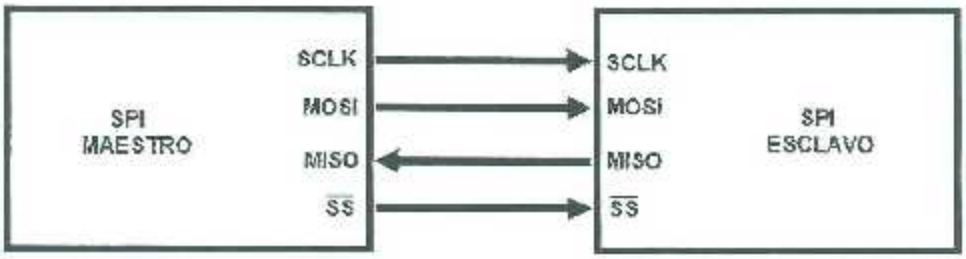


Figura 2.6 *Comunicación entre un amo y un esclavo a través de protocolo SPI.*

Para iniciar la comunicación, el dispositivo amo transmite un cero lógico a través de la línea de selección de esclavo (esto debido a que la línea de selección de esclavo se activa con un 0 lógico). Enseguida el dispositivo amo configura la señal de reloj, a una frecuencia menor o igual a la que soporta el dispositivo esclavo. Tales frecuencias se encuentran dentro del rango de 1 – 100 MHz. Si se requiere un periodo de espera, por ejemplo, para realizar una conversión de una señal análoga a digital, el amo debe esperar al menos ese periodo requerido antes de iniciar con la generación de los pulsos de reloj.

Durante cada ciclo de reloj se realiza la transmisión de datos de manera full dúplex, es decir, el dispositivo amo envía un bit a través de la línea MOSI y el esclavo lo recibe a través de la misma línea de transmisión de datos, y el dispositivo esclavo envía un bit a través de la línea MISO y el amo lo recibe a través de la misma línea. Es posible que la comunicación entre amo y esclavo ocurra solamente en un sentido, ya sea que solamente el dispositivo amo o el esclavo realicen el envío de datos.

La transmisión de datos normalmente implica la utilización de dos registros de corrimiento de una determinada longitud de palabra. Una vez realizada la transmisión de datos entre los registros del amo y el esclavo, cada uno puede tomar el dato transmitido y realizar alguna operación determinada con dicho dato, como puede ser, guardar ese valor en la unidad de memoria. Si existen más datos que serán transmitidos el proceso se repite.

La transmisión de datos involucra un determinado número de ciclos de reloj. Cuando no hay dato por transmitir el dispositivo amo detiene la generación de la señal de reloj y normalmente desactiva la señal de selección de esclavo enviando

un 1 lógico. Por lo general, la transmisión de datos normalmente consiste de palabras de 8 bits, y el amo inicializa la transmisión de datos conforme sea requerido por la aplicación. Sin embargo, el manejo de otras longitudes de palabra son también de amplia utilización, como son las palabras de 16 bits, o las de 12 bits que diversos convertidores digital - análogo, análogo - digital utilizan para llevar a cabo la transmisión de datos. Se debe tener en consideración que cada dispositivo esclavo que no sea activado a través de su línea de selección debe ignorar las señales de reloj y MOSI, y no debe transmitir a través de MISO. El dispositivo amo debe seleccionar sólo un esclavo a la vez.

Además de configurar la frecuencia a la que se debe realizar la transmisión de datos, el dispositivo amo debe configurar la polaridad de reloj y la fase de reloj. Comúnmente se hace referencia a estos dos parámetros como CPOL y CPHA respectivamente.

Por lo tanto, se tienen cuatro modos de operación, basados en la consideración de estos dos parámetros. El dispositivo amo y esclavo deben de utilizar el mismo modo de operación para establecer la comunicación de manera precisa. El bit de control de polaridad de reloj CPOL especifica si la señal de reloj tendrá un estado normalmente en nivel alto o en nivel bajo, y no tiene un efecto en el formato de transmisión de datos. El bit de control de fase CPHA especifica uno de los dos formatos fundamentales para la transmisión de datos [6].

2.2.1 Formato de transferencia de datos con bit de control CPHA = 0

En este modo de operación el primer flanco de la señal de reloj SCLK se usa para realizar la transmisión del primer bit de datos del dispositivo esclavo hacia el amo y el primer bit de datos del amo al esclavo. Un medio ciclo de reloj después, aparece el segundo flanco en la señal de reloj SCLK. Cuando este segundo ciclo de reloj ocurre, el valor a transmitir se carga ya sea en el bit más significativo o en el bit menos significativo del registro de corrimiento. Después de este segundo flanco en la señal de reloj, se lleva a cabo la transmisión del siguiente bit. Este proceso continúa por el total de longitud de palabra que requieran los componentes a utilizar dentro de una determinada aplicación,

transmitiendo el dato en los flancos impares de la señal de reloj, y realizando el corrimiento de los datos en los flancos pares de la señal de reloj, como se muestra en la Figura 2.7.

En este caso, después del flanco número 16 en la señal de reloj, el dato que estaba previamente en el registro del dispositivo amo ahora deberá estar en el registro del dispositivo esclavo, y el dato que estaba en el registro del dispositivo esclavo ahora deberá estar en el dispositivo amo.

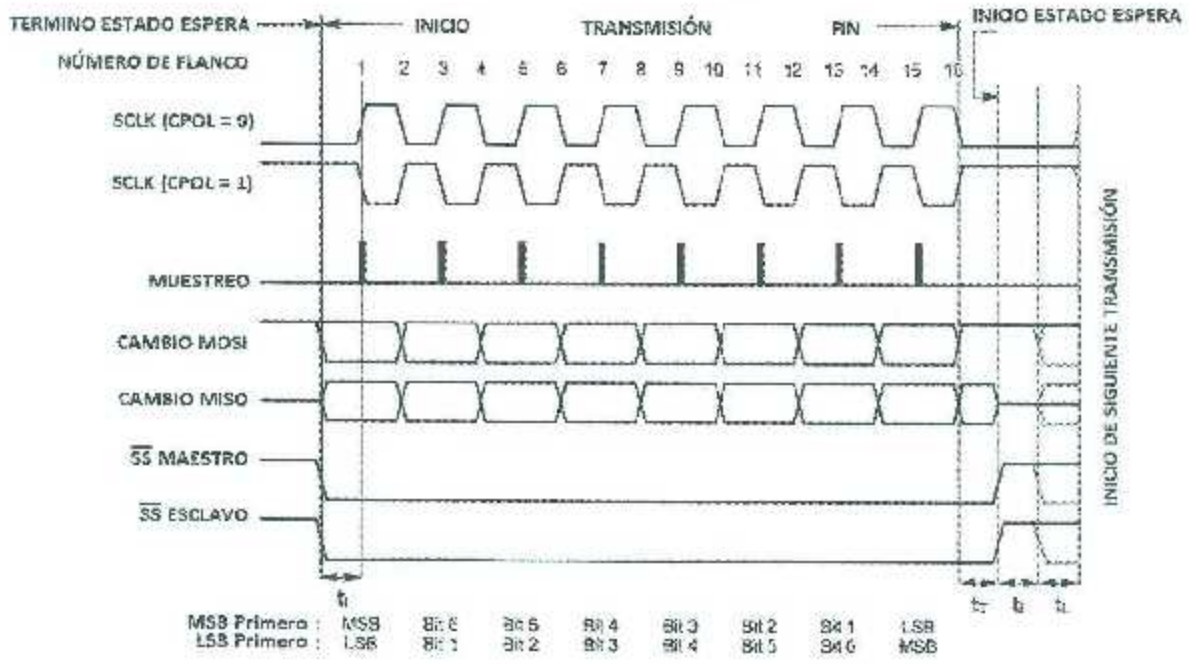


Figura 2.7 *Diagrama de tiempo de transferencia SPI con CPHA = 0.*

De la Figura 2.7:

t_L = Mínimo tiempo de espera antes del primer flanco de reloj.

t_T = Mínimo tiempo de espera posterior al último flanco de reloj.

t_I = Mínimo tiempo de espera entre transferencias de datos (Mínimo tiempo de señal SS en nivel alto).

t_L, t_T, y t_I = Mínimo 1/2 SCLK.

2.2.2 Formato de transferencia de datos con bit de control CPHA = 1

Algunos periféricos requieren el primer flanco de reloj antes de que el primer bit de dato esté disponible para ser transmitido. El primer flanco de la señal SCLK ocurre inmediatamente después del medio ciclo de reloj (señal de retraso). Un medio ciclo después, ocurre el segundo ciclo en la señal de reloj SCLK. Durante este flanco se carga el dato para el dispositivo amo y esclavo. Cuando el tercer flanco ocurre, se realiza el corrimiento del valor previamente cargado ya sea en el bit más significativo o en el bit menos significativo del registro SPI, dependiendo de la configuración establecida. En la Figura 2.8, se muestra como este proceso continúa por un total de 16 flancos en la señal de reloj, transmitiendo el dato en los flancos pares de la señal de reloj, y realizando el corrimiento de los datos en los flancos pares de la señal de reloj.

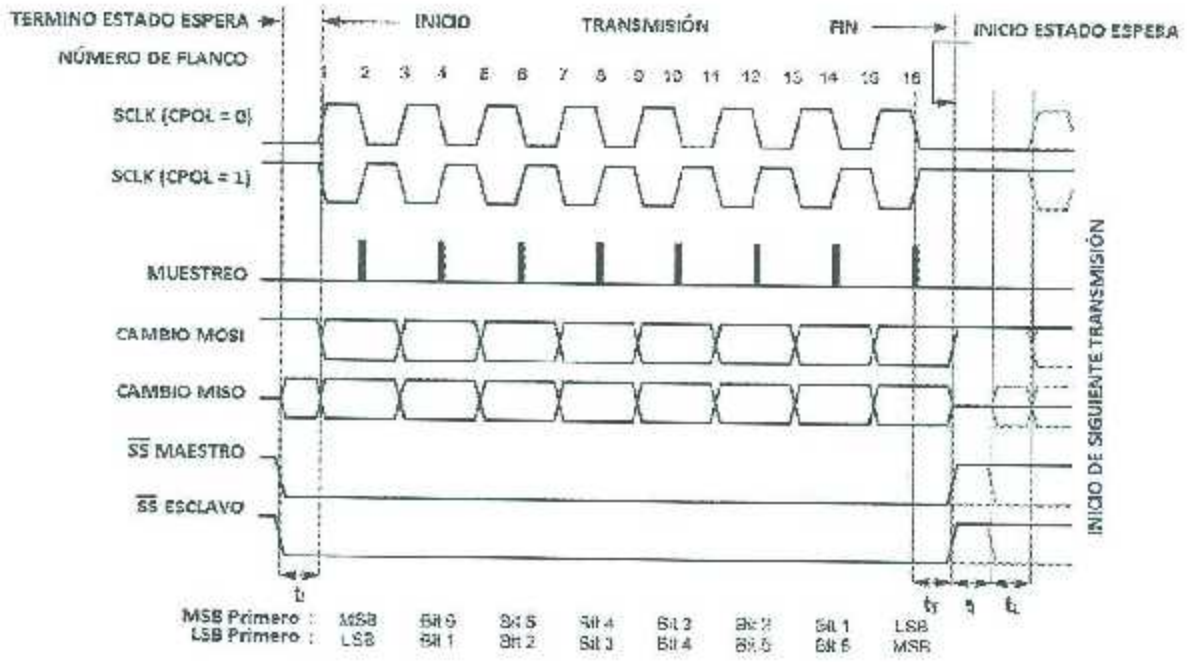


Figura 2.8 Diagrama de tiempo de transferencia SPI con CPHA = 1.

De la Figura 2.8:

t_L = Mínimo tiempo de espera antes del primer flanco de reloj.

t_T = Mínimo tiempo de espera posterior al último flanco de reloj.

t_I = Mínimo tiempo de espera entre transferencias de datos (Mínimo tiempo de señal $\overline{SS}$ en nivel alto).

t_L, t_T, y t_I = Mínimo 1/2 SCLK.

2.3Convertidor Analógico-Digital

Un convertidor análogo-digital es un dispositivo electrónico capaz de convertir una señal analógica (usualmente voltaje) en un número digital que representa el valor de la amplitud de dicha señal. La conversión implica la cuantificación de la entrada, por lo tanto se tiene una pequeña cantidad de error. Un convertidor analógico-digital realiza las conversiones (muestreo de la entrada) de manera periódica, teniendo como resultado una secuencia de valores digitales que han convertido una señal analógica continua en el tiempo y continua en amplitud en una señal digital discreta en el tiempo y discreta en amplitud.

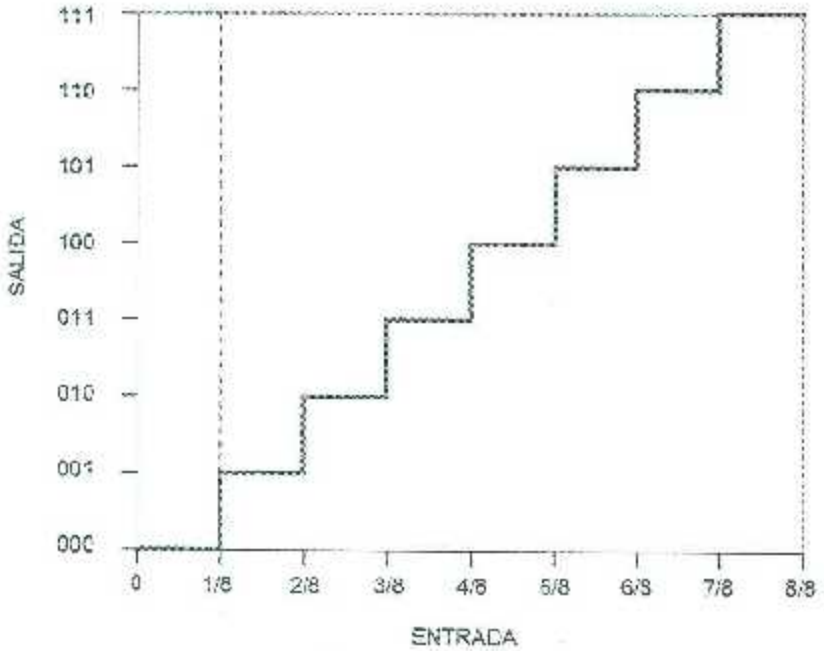


Figura 2.9 *Representación de un convertidor análogo-digital de 3 bits de resolución.*

2.3.1Resolución

La resolución del convertidor indica el número de valores discretos que puede producir dentro del rango de valores analógicos. Por lo general, los valores son almacenados en forma binaria, por lo que la resolución se expresa normalmente en bits. Por lo tanto, el número de valores discretos disponibles, es una potencia de dos. Por ejemplo un convertidor analógico digital de 8 bits de resolución puede convertir una señal analógica de entrada a uno de los 256 valores diferentes, ya que 2^8 = 256. Los valores pueden representar rangos de 0 a 255, o de -128 a 127, dependiendo del tipo. La resolución también se puede definir eléctricamente, y se expresa en Volts. La resolución Q del convertidor analógico-digital es igual al cambio mínimo de voltaje requerido para garantizar un cambio en el nivel de salida.

La resolución de en términos de voltaje de un convertidor analógico-digital es igual a su rango de medición de voltaje dividido por el número de valores discretos:

$$Q = \frac{E_{FSR}}{2^M - 1}$$

Donde:

M = Resolución del convertidor en bits.

E_{FSR}= Rango completo de medición de voltaje.

$$E_{FSR} = V_{REFHI} - V_{REFLOW}$$

Donde:

V_{REFHI} =Límite superior del rango de medición de voltaje.

V_{REFLOW} = Límite inferior del rango de medición e voltaje.

La mayoría de los convertidores análogo-digital son del tipo linear, lo que implica que el rango de valores de entrada tiene una relación lineal con el valor de salida.

Un convertidor análogo-digital tiene diferentes fuentes de errores. El error de cuantificación y de no linealidad (asumiendo que el convertidor es del tipo lineal) son propios a cualquier convertidor análogo-digital.

2.3.2Error de Cuantificación

La salida de un convertidor análogo-digital es una representación cuantificada de la señal análoga original. El termino cuantificación se refiere a la subdivisión de un rango en pequeños incrementos medibles, es decir, el rango total de entrada permitido es dividido entre un número finito de regiones con un incremento fijo. El proceso de cuantificación tiene el potencial de introducir una inexactitud conocida como error de cuantificación. Por lo tanto, la magnitud del error de cuantificación en un determinado instante de muestreo puede ser entre cero y la mitad de un bit menos significativo. El error de cuantificación es debido a la resolución finita de la representación digital de una señal. Si el rango de voltaje de entrada se mantiene constante, un convertidor de más alta resolución tendrá menos error de cuantificación debido a que el rango es dividido entre incrementos más pequeños.

2.3.3No linealidad

El error de no linealidad es causado por factores físicos, el cual provoca que la salida se desvíe de una función lineal de su entrada (o de cualquier otra función, en caso de que el convertidor sea de otro tipo). Por lo general, estos errores se pueden disminuir con la calibración del dispositivo o pueden ser prevenidos mediante pruebas. Parámetros importantes son la no linealidad integral y la no linealidad diferencial. Estas no linealidades reducen el rango de las señales que pueden ser digitalizadas por el convertidor, reduciendo también la resolución efectiva del convertidor análogo-digital [7].

Capítulo 3. Desarrollo

3.1 Etapa de Diseño

En la etapa de diseño se realizó un análisis de los recursos requeridos para llevar a cabo la aplicación deseada así como las características con las que debe cumplir. Se describió el funcionamiento de aplicaciones SPI en dispositivos FPGA utilizando el lenguaje de descripción de hardware VHDL así como la herramienta MicroBlaze.

En base a esto se propuso la electrónica requerida para llevar a cabo la implementación de un diseño funcional utilizando el lenguaje de programación VHDL.

3.1.1 Descripción

La Figura 3.1 muestra un esquema general de la implementación de un dispositivo amo SPI dentro de un FPGA. Se tiene un dispositivo amo integrado dentro de un sistema y uno o más dispositivos esclavos.

Se observa el bus SPI a través del cual el dispositivo amo se comunica con el dispositivo esclavo y que contiene las cuatro señales que especifica el protocolo: SCLK, MOSI, MISO y SS. El dispositivo amo se encarga de generar la señal de activación y la señal de reloj hacia el dispositivo esclavo, a través de las líneas MOSI y MISO se lleva a cabo la transmisión de datos.

Se tienen las señales a través de las cuales el usuario lleva a cabo el control, configuración y transmisión de información con el dispositivo amo. El dispositivo amo envía el dato recibido desde el esclavo hacia la lógica del usuario y/o recibe a través del bus de comunicación con la lógica del usuario el dato a transmitir hacia el dispositivo esclavo.

De este modo se lleva a cabo la transmisión de la información desde el dispositivo esclavo, el cual se encarga de adquirir la señal, hacia el dispositivo amo. Al terminar la transmisión de datos a través del protocolo SPI, el usuario cuenta con el dato dentro del FPGA.

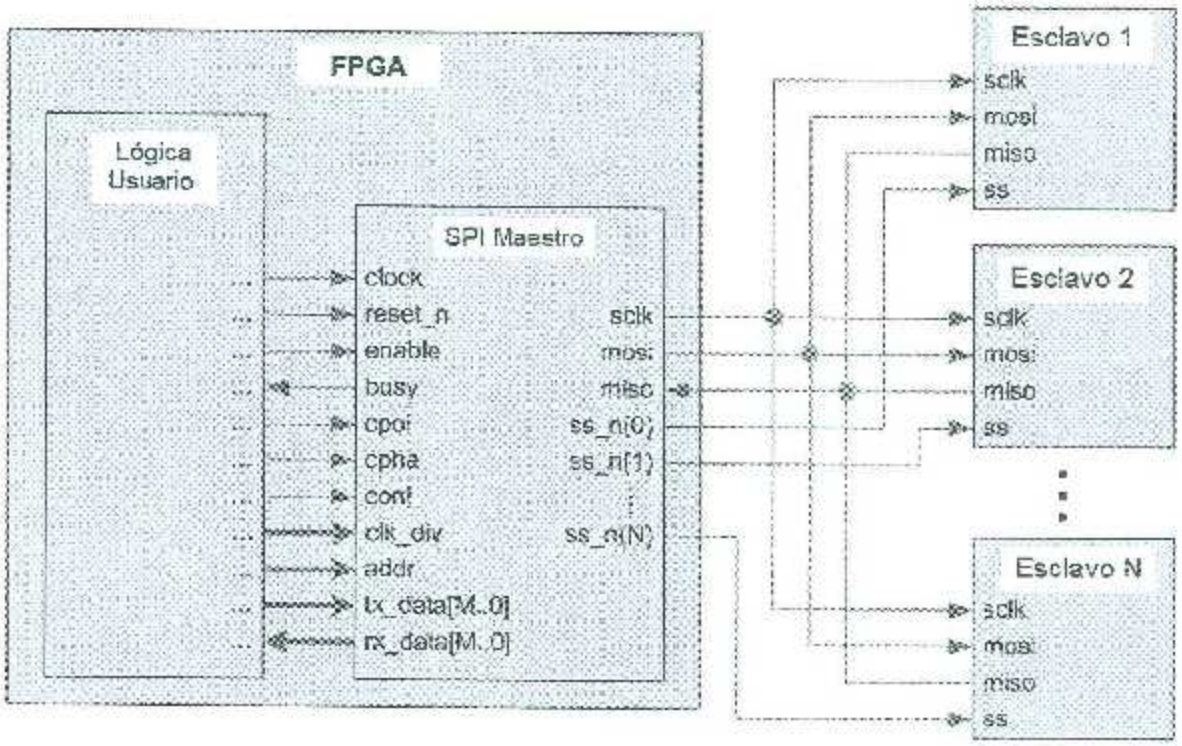


Figura 3.1 *Esquema general de SPI amo integrado en un sistema.*

El dispositivo amo inicia la transmisión de datos al enviar a nivel bajo la línea de selección de esclavo (SS). La línea de reloj serial (SCLK), generada por el amo, proporciona una señal de reloj síncrona. El amo transmite el dato a través de la línea MOSI y recibe los datos a través de la línea MISO.

El dispositivo amo se puede comunicar con múltiples esclavos mediante la utilización de diversas configuraciones. En la configuración más común, cada esclavo cuenta con una línea independiente de selección de esclavo (SS) pero comparte las líneas SCLK, MISO y MOSI con los demás esclavos. Cada esclavo ignora las líneas compartidas cuando su línea de SS no está en nivel bajo. Esta topología se muestra en la Figura 3.1.

3.1.1.1Descripción de Puertos

La Tabla 3.1 describe los puertos utilizados para la implementación de dispositivo amo SPI en el FPGA.

Tabla 3.1 Descripción de puertos.

Puerto	Tamaño	Modo	Tipo de Dato	Interfaz	Descripción
enable	1	Entrada	Standard Logic	Lógica Usuario	Permiso para iniciar transferencia de datos
clock	1	Entrada	Standard Logic	Lógica Usuario	Reloj de la FPGA
reset	1	Entrada	Standard Logic	Lógica Usuario	Reset asíncrono del sistema
cpol	1	Entrada	Standard Logic	Lógica Usuario	Bit de control polaridad de reloj
sclk	1	Salida	Standard Logic	Esclavo	Reloj SPI
ss	1	Salida	Standard Logic	Esclavo	Señal de selección de esclavo
miso	1	Entrada	Standard Logic	Esclavo	Entrada del amo salida del esclavo
tx_buffer	12	Entrada	Standard Logic Vector	Lógica Usuario	Dato recibido
count	6	Entrada	Standard Logic Vector	Lógica Usuario	Número de cuentas del reloj de la FPGA por 1/2 ciclo de sclk

La señal *clock* y *count* definen la frecuencia de *sclk*, *clock*es el cristal oscilador de 50 MHz con el que cuenta la tarjeta do desarrollo Nexys 2 utilizado para operar la lógica síncrona dentro del FPGA. La señal *count*permite al usuario definir la velocidad a la cual se llevará a cabo la transmisión de datos entre el esclavo y el amo, *count*es el número de periodos de *clock*entre transiciones de *sclk*. La frecuencia de *sclk* se describe mediante la siguiente ecuación:

$$f_{sclk} = \frac{f_{clock}}{2 \times count}$$

Cuando *count* es igual a 1, la frecuencia de *sclk* es la mitad de la frecuencia de la señal *clock* y está configurado bajo la máxima velocidad de transferencia de datos alcanzable.

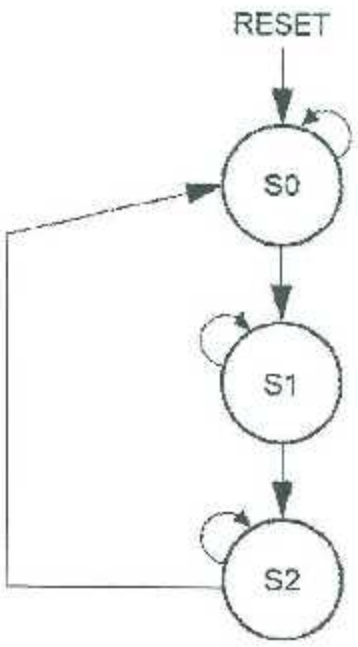


Figura 3.2*Representación general diagrama de estados SPI en FPGA.*

La secuencia se basa en tres estados, durante el estado S0 se mantiene la señal de *SS* en nivel alto y se configura el bit de control de polaridad CPOL. Permanecerá en este estado hasta que el permiso para pasar al siguiente estado este en nivel alto. Antes de pasar al siguiente estado se realiza la configuración de todos los parámetros para llevar a cabo la transmisión de datos. Para esta aplicación se agregó un estado S1 en el cual se introduce un retraso para realizar la conversión, una vez que termina la espera Durante el estado S2 se realiza la transmisión de datos, para esto, se envía a nivel bajo la señal de *SS*, se genera la señal de reloj *sclk* y se transmiten los datos en base a registros de corrimiento. Al finalizar la trasmisión de datos, se manda a nivel alto la señal SS y se regresa al estado S0, donde permanecerá hasta que el permiso se active de nueva cuenta. Es posible observar en la Figura 3.2 que se cuenta con un *reset*asíncrono, a través del cual se manda al estado S0 y se inicializan todos los registros y señales.

Con base en el análisis de los dispositivos con los que se desea realizar la comunicación así como a la utilización de herramientas como divisores de frecuencia, máquinas de estados y registros de corrimiento fue posible hacer un

programa a la medida e implementar el estándar SPI mediante la utilización de tecnología FPGA utilizando la menor cantidad de recursos posible.

3.1.2Electrónica

Para llevar a cabo la captura análoga se utilizó el convertidor análogo a digitalde 12 bits de resolución MAX187 (Maxim Integrated, San Jose, California, USA).Se eligió pues cuenta con comunicación SPI, además de que tiene un bajo consumo de energía y una alta velocidad de conversión, haciendo este convertidor ideal para aplicaciones de procesamiento digital de señales.

Como módulo FPGA se utilizó la tarjeta de desarrollo Nexys 2 (Digilent Inc, Pullman, Washington, USA). Esta tarjeta permite la utilización de puertos de expansión a través de los cuales es posible realizar la comunicación directamente con la FPGA. Al contar con múltiples dispositivos de entrada / salida, permite el desarrollo de aplicaciones de una manera accesible, brindando mayor flexibilidad al módulo.

En el anexo A se incluye un resumen de las hojas de datos y manual de referencia de los dispositivos mencionados.

3.2Etapa de Programación

Se utilizó el programa Xilinx ISE 8.2i (Xilinx Inc, San Jose, California, USA), para realizar la generación del código en VHDL. Este programa proporciona las herramientas para poder diseñar, simular e implementar aplicaciones basadas en tecnología FPGA. A partir de esta versión, se tuvieron mejoras significativas que permiten un mejor aprovechamiento y optimización de recursos.

3.2.1Programación

Inicialmente se debe generar un nuevo proyecto y se deben definir las propiedades del dispositivo a programar. La Fig. 3.3 muestra las propiedades específicas de la FPGA que contiene la tarjeta de desarrollo Nexys 2.

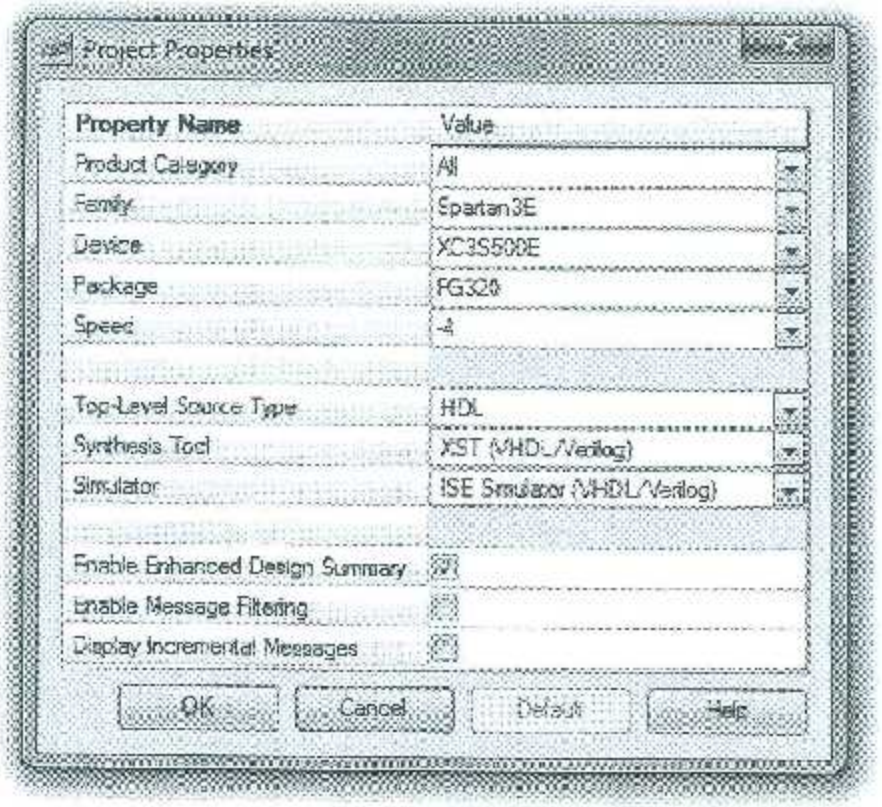


***Figura 3.3** Propiedades FPGA de tarjeta de desarrollo Nexys 2.*

Enseguida se debe agregar un módulo VHDL al proyecto, es posible definir los puertos desde la generación del módulo, o durante la realización del programa. Este módulo contendrá el programa con el que se llevará a cabo la implementación del estándar SPI.

A continuación se presenta el programa utilizado para realizar la comunicación y se muestra con texto en color verde la descripción de cada parte del código (para agregar texto como comentario en una línea dentro del programa se utilizan dos guiones, y se resalta el texto en color verde). Este programa se aplica específicamente para realizar la transmisión de datos entre la tarjeta de desarrollo Nexys 2 y el convertidor análogo-digital MAX187, y sirve de referencia para aplicaciones similares en las que se desee realizar la transmisión de datos a través del protocolo SPI utilizando tecnología FPGA.

```
--Se deben definir las librerías del estándar IEEE que se
utilizarán en el programa.
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

--Enseguida se realiza la definición de la entidad (entity).
La entidad define como un modelo descrito en VHDL se conecta
con otros modelos en VHDL. La entidad define también el
nombre del modelo y permite definir cualquier parámetro que
pasará al modelo utilizando jerarquía. El método a través del
cual se realiza la conexión de las entidades entre sí es
usando PORT. La declaración de puertos define el tipo de
conexión y la dirección.

entityMAX187 is
     PORT(
          clock     :     IN    STD_LOGIC;
          enable    :     IN    STD_LOGIC;
          reset     :     IN    STD_LOGIC;
          cpol      :     IN    STD_LOGIC;
          sclk      :     OUT   STD_LOGIC;
          ss        :     OUT   STD_LOGIC;
          miso      :     INSTD_LOGIC);
endMAX187;

--Mientras que la entidad define los aspectos referentes a la
interfaz y parámetros del modelo, la arquitectura define el
comportamiento. Después de la declaración del nombre de la
arquitectura y antes de la declaración "begin", se realiza la
declaración de las señales locales o variables.
```

```
architecture Behavioral ofMAX187is
    SIGNAL    clk_toggles :  STD_LOGIC_VECTOR(4 DOWNTO 0);
    SIGNAL    count       :  STD_LOGIC_VECTOR(5 DOWNTO 0);
    SIGNAL    count_delay:   STD_LOGIC_VECTOR(8DOWNTO 0);
    SIGNAL    aux_ss      :  STD_LOGIC;
    SIGNAL    aux_sclk    :  STD_LOGIC;
    SIGNAL    tx_buffer   :  STD_LOGIC_VECTOR(11 DOWNTO 0);
    SIGNAL    aux_dato    :  STD_LOGIC_VECTOR(11  DOWNTO  C);
    SIGNAL    assert_data:   STD_LOGIC;
    TYPE machine IS     (ready,delay,execute);
    SIGNAL    state       :  machine;
Begin
  Asignación de valor a la señales ss y sclk

ss <= aux_ss;
sclk <= aux_sclk;

--El proceso es la unidad funcional básica en VHDL. El
proceso en VHDL es el mecanismo a través del cual
declaraciones secuenciales se ejecutan en la secuencia
correcta, con la posibilidad de ejecutar más de un proceso al
mismo tiempo. Cada proceso es conformado por la lista de
sensibilidad, declaraciones y asignaciones. En primera
instancia se tiene un reset asíncrono, si reset es igual a 1
entonces se inicializan todos los registros y señales y se
manda al estado ready (S0), si no se cumple la condición
anterior pasa a la parte síncrona del proceso donde se lleva
a cabo la transmisión de datos en base a la utilización de
máquinas de estados. Esta parte del proceso depende de la
detección de flanco de subida en la señal de reloj. La
máquina de estados se implementa a través de la instrucción
case, en primer instancia se tiene el estado ready donde se
```

encuentra en espera de la activación del permiso para iniciar la transferencia de datos. Si la señal enable es igual a 1 se inicializan todos los registros y variables y se manda al estado delay (Si), si no se cumple la condición anterior permanecerá en el estado ready (S0).

```
PROCESS(clock, reset)
    BEGIN
        IF(reset = '1') THEN
            aux_ss <= '1';
            aux_sclk <= cpol;
            tx_buffer <= (OTHERS => '0');
            state <= ready;
            count <= (OTHERS => '0');
        ELSIF(clock'EVENT AND clock = '1') THEN
            CASE state IS
                WHEN ready =>
                    aux_ss <= '1';
                    aux_sclk <= cpol;
                    IF(enable = '1') THEN
            assert data <= '1';
            aux_ss <= '1';
                        aux_sclk <= cpol;
                        clk_toggles <= "00000";
                        count <= (OTHERS => '0');
                        count_delay<= (OTHERS => '0');
                        tx_buffer <= (OTHERS => '0');
                        state <= execute;
                    ELSE
                        state <= ready;
                    END IF;
```

--En el estado delay (S1) se envía a nivel bajo la señal de SS yse agrega un retraso de 10 microsegundos para llevar a cabo la conversión conforme a hoja de datos del convertidor MAX187.

```
WHEN delay =>
    aux_ss <= '0';
    IF count_delay = "111110100" THEN
        count <= (OTHERS => '0');
        state <= execute;
    ELSE
        count <= count + 1;
        state <= delay;
    END IF;
```

--En el estado executo (S2) se mantiene en nivel bajo la señal de SS. Se utiliza la variable count para contar el número de periodos de clock entre transiciones de aux_sclk, y se utiliza para obtener una frecuencia de 1 MHz a partir del reloj de 50 MHz con el que cuenta la tarjeta de desarrollo. La variable clk_toggles lleva el número de transiciones del reloj spi (aux_sclk). La variable assert_data se utilizada para transmitir el dato cada vez que se completa un ciclo completo del reloj spi. Una vez alcanzado el máximo número de transiciones del reloj spi se termina la transferencia de datos, para lograr esto se envía a nivel alto la señal de SS, se pasa el dato a otra variable para que pueda ser utilizado por otros procesos y se envía de nuevo al estado de ready (S0) para permanecer en espera de otra solicitud de transmisión de datos.

```
                WHEN execute =>
                    aux_ss <= '0';
                    IF count = "11001" THEN
                        count <= (OTHERS => '0');
                        assert_data <= NOT assert_data;
                        clk_toggles <= clk_toggles + 1;

                        IF clk_toggles <= "11001" THEN
                            aux_sclk <= NOT aux_sclk;
                            state <= execute;
                        ELSE
                            aux_ss <= '1';
                            state <= ready;
                            aux_dato <= tx_buffer;
                        END IF;

                        IF assert_data = '1' THEN
                            tx_buffer  <=  tx_buffer(10
                            DOWNTO 0) & miso;
                            state <= execute;
                        END IF;

                    ELSE
                        count <= count + 1;
                        state <= execute;
                    END IF;

                END CASE;
            END IF;
END PROCESS;
END Behavioral;
```

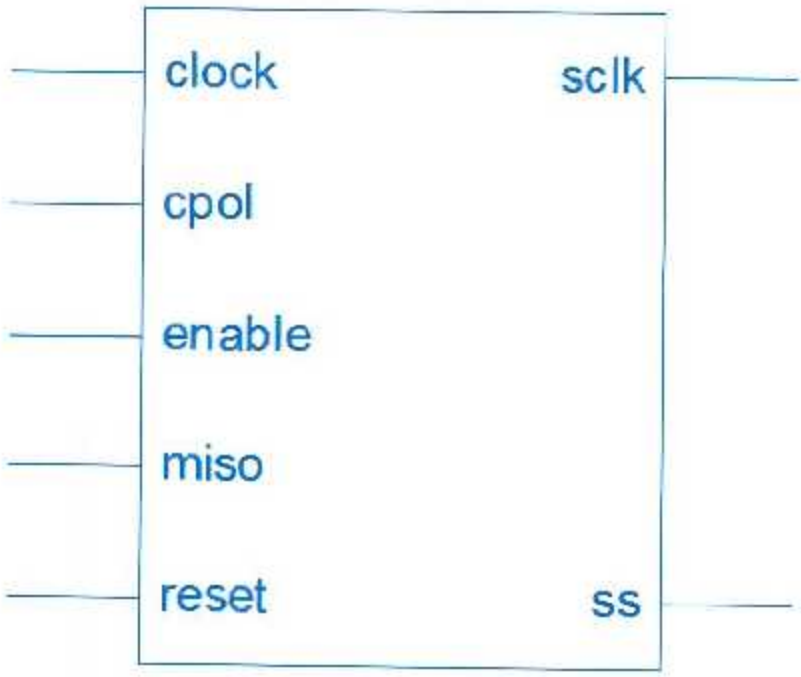


Figura 3.4*Representación esquemática del diseño.*

De la Fig. 3.4 se observan las salidas del lado derecho:*ss* y *sclk,* y las entradas al lado izquierdo: *clock, cpol, enable, misoy reset.* Las señales de *ss, sclk* y *miso* son las utilizadas para establecer la comunicación entre la FPGA y el convertidor análogo-digital, y las demás señales cumplen la función de control y configuración del módulo. Es importante considerar que con este programa se ha generado hardware digital. No son un conjunto de instrucciones que se ejecutan de manera secuencial almacenadas en una unidad de memoria. Por lo que estas señales son la entrada y salida del sistema generado, constituido por diversos elementos digitales interconectados de una manera específica. El usuario solamente realiza la programación en leguaje VHDL, y el *software* se encarga de realizar las operaciones posteriores, hasta generar el archivo de extensión *.bit*que será con el que se programará el FPGA.

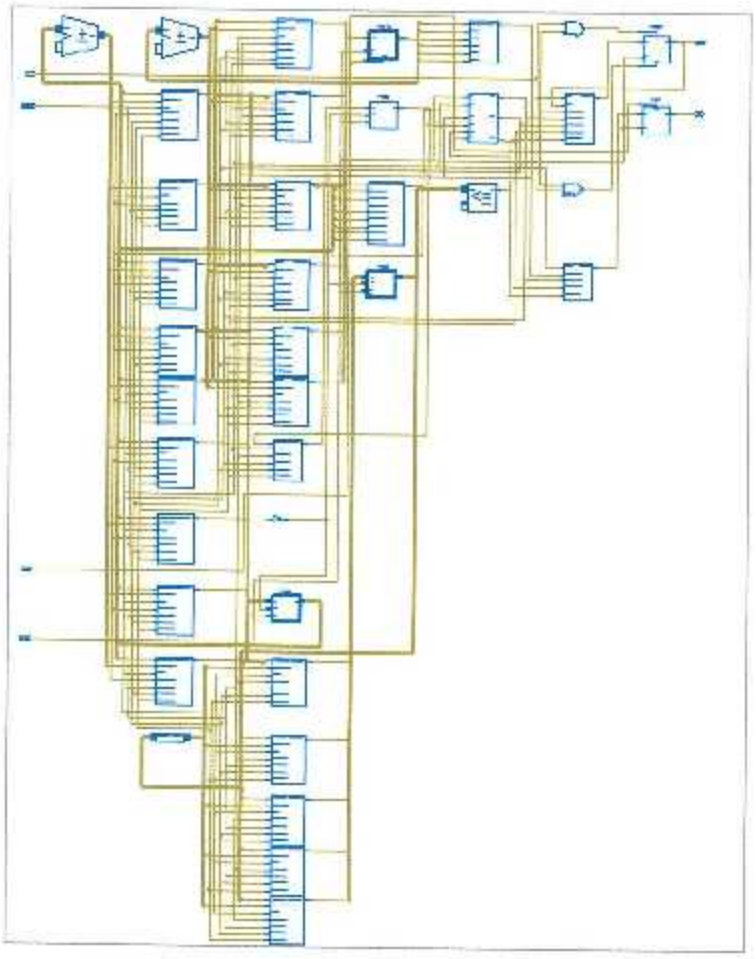


Figura 3.5 Hardware digital generado.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	27	9,312	1%	
Number of 4 input LUTs	76	9,312	1%	
Logic Distribution				
Number of occupied Slices	42	4,656	1%	
Number of Slices containing only related logic	42	42	100%	
Number of Slices containing unrelated logic	0	42	0%	
Total Number of 4 input LUTs	76	9,312	1%	
Number of bonded IOBs	7	232	3%	
IOB Flip Flops	1			
Number of GCLKs	1	24	4%	
Total equivalent gate count for design	686			
Additional JTAG gate count for IOBs	336			

Figura 3.6 Resumen de elementos del FPGA utilizados.

Para la asignación de pines que se utilizarán de la tarjeta de desarrollo se debe generar un archivo con la extensión *.ucfy* se debe agregar al proyecto. Este archivo contiene los puertos definidos en la entidad al inicio del programa y su ubicación en la tarjeta de desarrollo. A continuación se muestra el contenido del archivo *.ucf*.

```
NET "clock"     LOC = "B8"   ;
NET "cpol"      LOC = "G18"  ;
NET "enable"    LOC = "H18"  ;
NET "miso"      LOC = "K12"  ;
NET "reset"     LOC = "K18"  ;
NET "sclk"      LOC = "L17"  ;
NET "ss"        LOC = "L15"  ;
```

Capítulo 4. Resultados

Se obtuvo el diseño y la implementación de un sistema de adquisición de datos basado en tecnología FPGA, estableciendo comunicación con un convertidor analógico-digital de 12 bits de resolución a través del estándar SPI. A través de los puertos de expansión se realizó la comunicación entre la tarjeta de desarrollo Nexys 2 y el convertidor MAX187, en donde se definió como entrada un pin donde el FPGA recibirá el dato de manera serial a través de la señal MISO, y se definieron dos pines como salida para enviar al convertidor las señales de SS y SCLK generadas por el FPGA requerida para efectuar la transmisión de datos, como se muestra en la Figura 4.1.

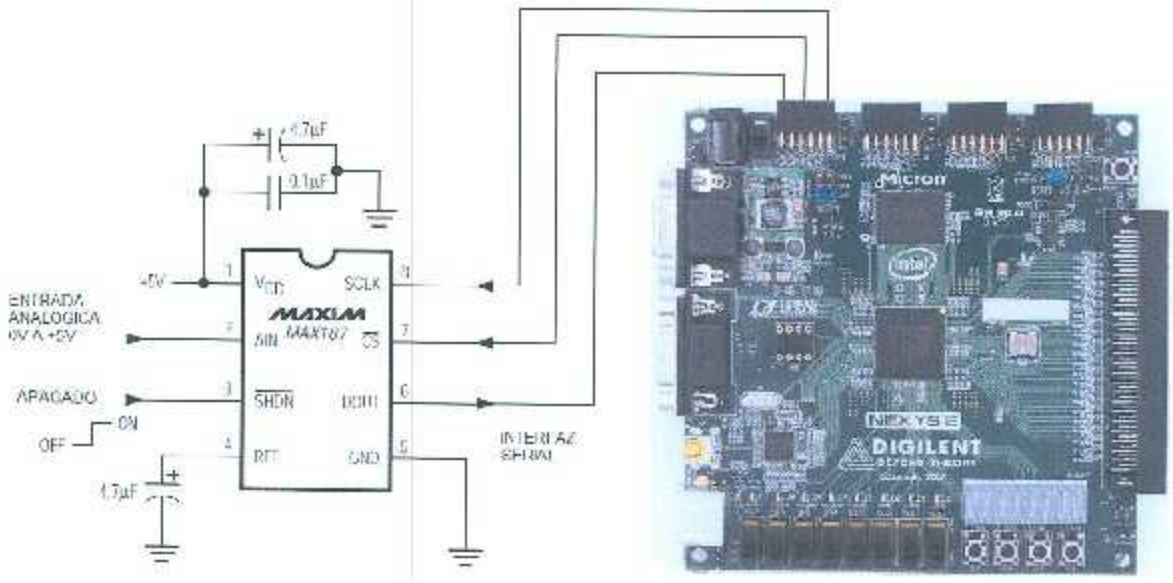


***Figura 4.1** Implementación de diseño propuesto para transmisión de datos.*

El diseño propuesto fue realizado a la medida, considerando las características y especificaciones de cada uno de los dispositivos utilizados para este proyecto, por lo que en aplicaciones similares se debe revisar la documentación de los mismos para realizar la comunicación. Sin embargo este

desarrollo establece el procedimiento y las bases para llevar a cabo la implementación.

El convertidor análogo-digital manufacturado por *Maxim Integrated*resulta una buena opción para el diseño e implementación de esta aplicación ya que opera de manera confiable, presenta un error de no linealidad muy reducido y bajo consumo de energía, por mencionar algunos. La tarjeta de desarrollo Nexys 2 fabricada por *Digilent Inc*, es una excelente opción para el desarrollo de aplicaciones basadas en tecnología FPGA, ya que cuenta con la Xilinx Spartan 3E, componente manufacturado por la compañía líder en productos y soluciones de tecnología FPGA. Una vez finalizado el desarrollo de la aplicación mediante esta tarjeta, es posible pasar a la etapa de elaboración de prototipos utilizando solamente los componentes requeridos. Esto representa un ahorro durante el proceso de desarrollo de aplicaciones comparado con otras tecnologías.

Como resultado de la evaluación y análisis respecto de la manera más conveniente para llevar a cabo la implementación de transmisión de datos a través del protocolo SPI utilizando tecnología FPGA, durante el desarrollo de este proyecto se efectuaron aplicaciones basando su operación en la utilización del procesador embebido MicroBlaze. Con la herramienta XPS de Xilinx es posible diseñar una plataforma donde el núcleo del sistema sea un microprocesador, en donde se integran componentes de *hardware* y *software* como periféricos (Módulos IP), según los requerimientos de la aplicación deseada. A través de estas aplicaciones se demuestra el procedimiento para desarrollar y programar el FPGA en base a la herramienta MicroBlaze.

Se logró la implementación de aplicaciones que involucran la utilización de entradas / salidas de propósito general, temporizadores, interrupciones, comunicación SPI y del módulo transmisor - receptor asíncrono universal. La herramienta MicroBlaze representa otra perspectiva respecto a la utilización de tecnología FPGA, y las aplicaciones desarrolladas sirven de referencia para su utilización en proyectos futuros donde resulte más conveniente esta solución.

Capítulo 5. Conclusiones

Los sistemas de adquisición de datos y aplicaciones de comunicación que involucran señales de sensores complejos requieren de la capacidad de procesar de manera rápida y eficiente en energía las señales obtenidas, así como también capacidad de cómputo de manera flexible y confiable. Este trabajo permitió establecer la manera de implementar los sistemas de lógica configurable para la digitalización y manejo de variables empleando el procesamiento digital, así como la utilización del lenguaje de descripción de hardware VHDL para comunicación entre dispositivos que utilizan el estandar SPI. Sin embargo, cada dispositivo cuenta con características específicas de funcionamiento por lo que es necesario adaptar el algoritmo presentado en este trabajo para lograr una correcta comunicación.

Este sistema representa una alternativa tecnológica de bajo costo para el desarrollo de sistemas de adquisición de datoscon un enfoque de aplicación en el sector agrícola a través de redes de sensores inalámbricos. Los nodos que componen una red de sensores inalámbricos por lo general están basados en la utilización de microcontroladores, los cuales tienen una limitada capacidad de cómputo respecto al desarrollo de diversas aplicaciones. La tecnología FPGA resulta más eficiente para la realización de cálculos complejos en comparación con los microcontroladores, además, la capacidad de reconfiguración con la que cuenta la tecnología FPGA brinda mayor flexibilidad al sistema en el que sea utilizada.

El programa presentado en este proyecto basado en el lenguaje de descripción de hardware VHDL se diseño lo más genérico posible, por lo que resulta útil para cualquier desarrollo basado en tecnología FPGA que requiera comunicación a través del estándar SPI con cualquier dispositivo que transmita o reciba datos en base a este protocolo. La comunicación lograda resultó satisfactoria debido a que cumple con los requerimientos establecidos en este proyecto.

Referencias Bibliográficas

[1] U. Meyer-Baese,*Digital Signal Processing with Field Programmable Gate Arrays*.Nueva York, Estados Unidos: Springer, 2007.

[2] *Spartan-3A FPGA Family: Data Sheet*. Especificación DS529, Xilinx Inc., Agosto 2010.

[3] *Nexys™ 2 Spartan-3E FPGA Board*. Documento disponible en Internet. Digilent Inc., 2012.
http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,789&Prod=NEXYS2
[Consulta: 5 Noviembre 2011]

[4] P. Wilson, *Design Recipes for FPGAs*. Oxford, Reino Unido: Elsevier Inc., 2007.

[5] *MicroBlaze Processor Reference Guide*. Guía de referencia UG081, Xilinx Inc., Enero 2008.

[6] *Serial Peripheral Interface Bus*[texto en línea]. Wikipedia, 2012.
http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus
[Consulta: 18 Enero 2012]

[7] *Analog-to-digital converter* [texto en línea]. Wikipedia, 2012.
http://en.wikipedia.org/wiki/Analog-to-digital_converter
[Consulta: 22 Enero 2012]

[8] C. H. Roth Jr. y L. Kurian,*Digital Systems Design Using VHDL*. Stamford, Estados Unidos: Cengage Learning, 2008.

[9] P. Chu, *FPGA Prototyping by VHDL Examples: Xilinx Spartan-3 Version*. Hoboken, Estados Unidos: John Wiley & Sons, Inc., 2008.

[10] S. Kilts,*Advanced FPGA Design: Architecture, Implementation, and Optimization*. Hoboken, Estados Unidos: John Wiley & Sons, Inc., 2007.

[11] P. Ashenden,*The Designer´s Guide to VHDL*. Burlington, Estados Unidos: Elsevier Inc., 2008.

[12] S. Tapp, *Configuring Xilinx FPGAs with SPI Serial Flash*. Nota de Aplicación XAPP951, Xilinx Inc., Septiembre 2010.

[13] *Extended Spartan-3A Family Overview.* Especificación DS706, Xilinx Inc., Febrero 2011.

[14] *Spartan-3 Generation FPGA User Guide.* Guía de Usuario UG331, Xilinx Inc., Junio 2011.

[15] M. Peattie,*Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode.* Nota de AplicaciónXAPP502, Xilinx Inc., Agosto 2009.

[16] *MAX187/MAX189 +5V, Low-Power, 12-Bit Serial ADCs.* Hoja de Datos 19-0196; Rev 0; 10/93, Maxim Integrated, 1993.

[17] Velasco Martínez, Víctor Daniel:*Diseño de Nodos y de una Red Inalámbrica de Sensores para Mediciones Agropecuarias.* Tesis de Maestría, Instituto Tecnológico de la Laguna, Diciembre 2009.

A Resúmenes de Hoja de Datos y Manual de Referencia

Se presentan en este anexo los resúmenes de la hoja de datos y manual de referencia que presentan las características y la funcionalidad del circuito integrado y la tarjeta de desarrollo que se utilizaron en este proyecto. Las hojas de datos son propiedad de los respectivos fabricantes y se presentan con el objetivo de servir como referencia rápida en la lectura de esta tesis.

19-0196; Rev 0; 10/93

EVALUATION KIT MANUAL FOLLOWS DATA SHEET

MAXIM

+5V, Low-Power, 12-Bit Serial ADCs

General Description

The MAX187/MAX189 serial 12-bit analog-to-digital converters (ADCs) operate from a single +5V supply and accept a 0V to 5V analog input. Both parts feature an 8.5µs successive-approximation ADC, a fast track/hold (1.5µs), an on-chip clock, and a high-speed 3-wire serial interface.

The MAX187/MAX189 digitize signals at a 75ksps throughput rate. An external clock accesses data from the interface, which communicates without external hardware to most digital signal processors and microcontrollers. The interface is compatible with SPI™, QSPI™, and Microwire™.

The MAX187 has an on-chip buffered reference, and the MAX189 requires an external reference. Both the MAX187 and MAX189 save space with 8-pin DIP and 16-pin SO packages. Power consumption is 7.5mW and reduces to only 10µW in shutdown.

Excellent AC characteristics and very low power consumption combined with ease of use and small package size make these converters ideal for remote DSP and sensor applications, or for circuits where power consumption and space are crucial.

Applications

Portable Data Logging

Remote Digital Signal Processing

Isolated Data Acquisition

High-Accuracy Process Control

Features

- 12-Bit Resolution
- ±½ LSB Integral Nonlinearity (MAX187A/MAX189A)
- Internal Track/Hold, 75kHz Sampling Rate
- Single +5V Operation
- Low Power: 2µA Shutdown Current
  1.5mA Operating Current
- Internal 4.096V Buffered Reference (MAX187)
- 3-Wire Serial Interface, Compatible with SPI, QSPI, and Microwire
- Small-Footprint 8-Pin DIP and 16-Pin SO

Ordering Information

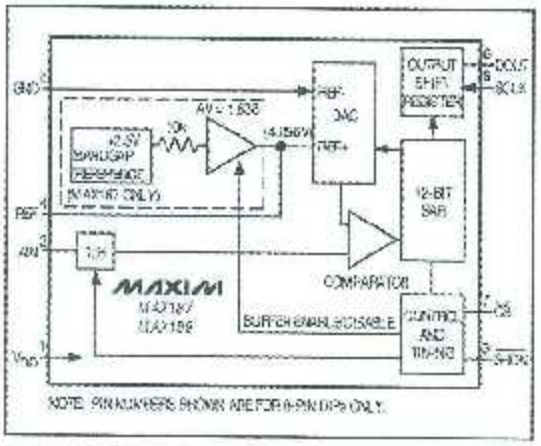
PART	TEMP. RANGE	PIN-PACKAGE	ERROR (LSB)
MAX187ACPA	0°C to +70°C	8 Plastic DIP	±½
MAX187BCPA	0°C to +70°C	8 Plastic DIP	±1
MAX187CCPA	0°C to +70°C	8 Plastic DIP	±2
MAX187ACWE	0°C to +70°C	16 Wide SO	±½
MAX187BCWE	0°C to +70°C	16 Wide SO	±1
MAX187CCWE	0°C to +70°C	16 Wide SO	±2
MAX187BC/D	0°C to +70°C	Dice*	±1

Ordering Information continued on last page.

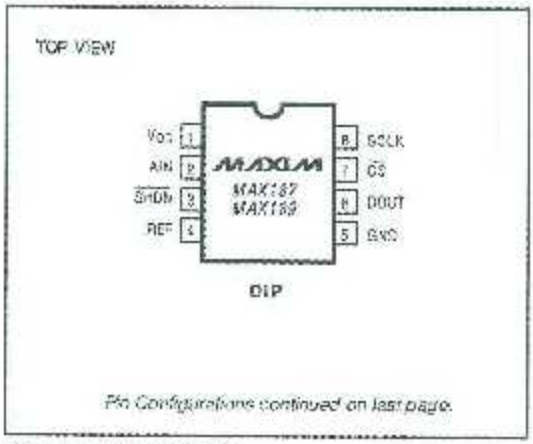
* Dice are specified at T_A = +25°C, DC parameters only.

** Contact factory for availability and processing to MIL-STD-883.

Functional Diagram



Pin Configurations



Pin Configurations continued on last page.

™ SPI and QSPI are trademarks of Motorola. Microwire is a trademark of National Semiconductor.

+5V, Low-Power, 12-Bit Serial ADCs

ABSOLUTE MAXIMUM RATINGS

VDD to GND ..... -0.3V to +6V
AIN to GND ..... -0.3V to (VDD + 0.3V)
REF to GND ..... -0.3V to (VDD + 0.3V)
Digital Inputs to GND ..... -0.3V to (VDD + 0.3V)
Digital Outputs to GND ..... -0.3V to (VDD + 0.3V)
SHDN to GND ..... -0.3V to (VDD + 0.3V)
REF Load Current (MAX187) ..... 4.0mA Continuous
REF Short-Circuit Duration (MAX187) 20sec
DOUT Current ..... ±20mA

Continuous Power Dissipation (T_A = +70°C)
8-Pin Plastic DIP (derate 9.09mW/°C above +70°C) 500mW
16-Pin Wide SO (derate 9.70mW/°C above +70°C) 478mW
8-Pin CERDIP (derate 8.00mW/°C above +70°C) 440mW
Operating Temperature Ranges:
MAX187_C_ /MAX189_C_ ..... 0°C to +70°C
MAX187_E_ /MAX189_E_ ..... -40°C to +85°C
MAX187_MJA/MAX189_MJA ..... -55°C to +125°C
Storage Temperature Range ..... -60°C to +150°C
Lead Temperature (soldering, 10sec) ..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS

(VDD = +5V ±5%; GND = 0V; unipolar input mode; 75ksps, f_{CLK} = 4.0MHz, external clock (50% duty cycle); MAX187—internal reference: VREF = 4.096V, 4.7µF capacitor at REF pin, or MAX189—external reference: VREF = 4.096V applied to REF pin, 4.7µF capacitor at REF pin; T_A = TMIN to TMAX unless otherwise noted.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
DC ACCURACY (Note 1)						
Resolution					12	Bits
Relative Accuracy (Note 2)		MAX18_A			±½	LSB
		MAX18_B			±1	
		MAX18_C			±2	
Differential Nonlinearity	DNL	No missing codes over temperature			±1	LSB
Offset Error		MAX18_A			±1½	LSB
		MAX18_B/C			±3	
Gain Error (Note 3)		MAX187			±3	LSB
		MAX189A			±1	
		MAX189B/C			±3	
Gain Temperature Coefficient		External reference, 4.096V		±0.8		ppm/°C
DYNAMIC SPECIFICATIONS (10kHz sine wave input, 0V to 4.096V_{p-p}, 75ksps)						
Signal-to-Noise plus Distortion Ratio	SINAD		70			dB
Total Harmonic Distortion (up to the 5th harmonic)	THD				-80	dB
Spurious-Free Dynamic Range	SFDR		80			dB
Small-Signal Bandwidth		Rolloff -3dB		4.5		MHz
Full-Power Bandwidth				0.8		MHz

+5V, Low-Power, 12-Bit Serial ADCs

ELECTRICAL CHARACTERISTICS (continued)

(V_{DD} = +5V ±5%; GND = 0V; unipolar input mode; 75ksps, f_{CLK} = 4.0MHz, external clock (50% duty cycle); MAX187—internal reference: V_{REF} = 4.096V, 4.7µF capacitor at REF pin, or MAX189—external reference: V_{REF} = 4.096V applied to REF pin, 4.7µF capacitor at REF pin, T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
CONVERSION RATE							
Conversion Time	t_{CONV}			5.5		8.5	µs
Track/Hold Acquisition Time	t_{ACQ}			1.5			µs
Throughput Rate		External clock, 4MHz, 13 clocks				75	ksps
Aperture Delay	t_{APD}				10		ns
Aperture Jitter					<50		ps
ANALOG INPUT							
Input Voltage Range						0 to V_{REF}	V
Input Capacitance (Note 4)					16		pF
INTERNAL REFERENCE (MAX187 only, reference buffer enabled)							
REF Output Voltage	V_{REF}	T_A = +25°C		4.076	4.096	4.116	V
		T_A = T_{MIN} to T_{MAX}	MAX187_C	4.060		4.132	
			MAX187_E	4.050		4.140	
			MAX187_M	4.040		4.150	
REF Short-Circuit Current						30	mA
REF Tempco		MAX187AC/BC			±30	±50	ppm/°C
		MAX187AE/BE			±30	±60	
		MAX187AM/BM			±30	±80	
		MAX187C			±30		
Load Regulation (Note 5)		0mA to 0.6mA output load			1		mV
EXTERNAL REFERENCE AT REF (Buffer disabled, V_{REF} = 4.096V)							
Input Voltage Range				2.50		V_{DD} + 50mV	V
Input Current					200	350	µA
Input Resistance				12	20		kΩ
Shutdown REF Input Current					1.5	10	µA

+5V, Low-Power, 12-Bit Serial ADCs

ELECTRICAL CHARACTERISTICS (continued)

(V_{DD} = +5V ±5%; GND = 0V; unipolar input mode; 75ksps, f_{CLK} = 4.0MHz, external clock (50% duty cycle); MAX187—internal reference: V_{REF} = 4.096V, 4.7µF capacitor at REF pin, or MAX189—external reference: V_{REF} = 4.096V applied to REF pin, 4.7µF capacitor at REF pin; T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
DIGITAL INPUTS (SCLK, CS, SHDN)							
SCLK, CS Input High Voltage	V_{INH}			2.4			V
SCLK, CS Input Low Voltage	V_{INL}					0.8	V
SCLK, CS Input Hysteresis	V_{HYST}				0.15		V
SCLK, CS Input Leakage	I_{IN}	V_{IN} = 0V or V_{DD}				±1	µA
SCLK, CS Input Capacitance	C_{IN}	(Note 4)				15	pF
SHDN Input High Voltage	V_{INSH}			V_{DD} - 0.5			V
SHDN Input Low Voltage	V_{INSL}					0.5	V
SHDN Input Current	I_{INS}	SHDN = V_{DD} or 0V				±4.0	µA
SHDN Input Mid Voltage	V_{INM}			1.5		V_{DD} - 1.5	V
SHDN Voltage, Floating	V_{SHFT}	SHDN = open			2.75		V
SHDN Maximum Allowed Leakage, Mid Input		SHDN = open		-100		100	nA
DIGITAL OUTPUT (DOUT)							
Output Voltage Low	V_{OL}	I_{SINK} = 5mA				0.4	V
		I_{SINK} = 16mA			0.3		
Output Voltage High	V_{OH}	I_{SOURCE} = 1mA		4			V
Three-State Leakage Current	I_L	CS = 5V				±10	µA
Three-State Output Capacitance	C_{OUT}	CS = 5V (Note 4)				15	pF
POWER REQUIREMENTS							
Supply Voltage	V_{DD}			4.75		5.25	V
Supply Current	I_{DD}	Operating mode	MAX187		1.5	2.5	mA
			MAX189		1.0	2.0	
		Power-down mode			2	10	µA
Power-Supply Rejection	PSR	V_{DD} = +5V, ±5%; external reference, 4.096V; full-scale input (Note 6)			±0.06	±0.5	mV

+5V, Low-Power, 12-Bit Serial ADCs

TIMING CHARACTERISTICS

(V_{DD} = +5.0V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Track/Hold Acquisition Time	t_{ACQ}	CS = high (Note 7)		1.5			µs
SCLK Fall to Output Data Valid	t_{DO}	C_{LOAD} = 100pF	MAX18_ _C/E	20		150	ns
			MAX18_ _M	20		200	
CS Fall to Output Enable	t_{DV}	C_{LOAD} = 100pF				100	ns
CS Rise to Output Disable	t_{TR}	C_{LOAD} = 100pF				100	ns
SCLK Clock Frequency	f_{SCLK}					5	MHz
SCLK Pulse Width High	t_{CH}			100			ns
SCLK Pulse Width Low	t_{CL}			100			ns
SCLK Low to CS Fall Setup Time	t_{CSO}			50			ns
CS Pulse Width	t_{CS}			500			ns

Note 1: Tested at V_{SS} = −5V.
Note 2: Relative accuracy is the deviation of the analog value at any code from its theoretical value after the full-scale range has been calibrated.
Note 3: MAX187—internal reference, offset nulled; MAX189—external +4.096V reference, offset nulled. Excludes reference errors.
Note 4: Guaranteed by design. Not subject to production testing.
Note 5: External load should not change during conversion for specified ADC accuracy.
Note 6: DC test, measured at 4.75V and 5.25V only.
Note 7: To guarantee acquisition time, t_{ACQ} is the maximum time the device takes to acquire the signal, and is also the minimum time needed for the signal to be acquired.

+5V, Low-Power, 12-Bit Serial ADCs

Pin Description

PIN		NAME	FUNCTION
DIP	WIDE SO		
1	1	V_{DD}	Supply voltage, +5V ±5%
2	3	AIN	Sampling analog input, 0V to V_{REF} range
3	6	SHDN	Three-level shutdown input. Pulling SHDN low shuts the MAX187/MAX189 down to 10µA (max) supply current. Both MAX187 and MAX189 are fully operational with either SHDN high or floating. For the MAX187, pulling SHDN high enables the internal reference, and letting SHDN float disables the internal reference and allows for the use of an external reference.
4	8	REF	Reference voltage—sets analog voltage range and functions as a 4.096V output for the MAX187 with enabled internal reference. REF also serves as a +2.5V to V_{DD} input for a precision reference for both MAX187 (disabled internal reference) and MAX189. Bypass with 4.7µF if internal reference is used, and with 0.1µF if an external reference is applied.
5	—	GND	Analog and digital ground
—	10	AGND	Analog ground
—	11	DGND	Digital ground
6	12	DOUT	Serial data output. Data changes state at SCLK's falling edge.
7	15	CS	Active-low chip select initiates conversions on the falling edge. When CS is high, DOUT is high impedance.
8	16	SCLK	Serial clock input. Clocks data out with rates up to 5MHz.
—	2,4,5,7,9,13,14	N.C.	Not internally connected. Connect to AGND for best noise performance.

Detailed Description

Converter Operation

The MAX187/MAX189 use input track/hold (T/H) and successive approximation register (SAR) circuitry to convert an analog input signal to a digital 12-bit output. No external hold capacitor is needed for the T/H. Figures 3a and 3b show the MAX187/MAX189 in their simplest configuration. The MAX187/MAX189 convert input signals in the 0V to V_{REF} range in 10µs, including T/H acquisition time. The MAX187's internal reference is trimmed to 4.096V, while the MAX189 requires an external reference. Both devices accept external reference voltages from +2.5V to V_{DD}. The serial interface requires only three digital lines, SCLK, CS, and DOUT, and provides easy interface to microprocessors (µPs).

Both converters have two modes: normal and shutdown. Pulling SHDN low shuts the device down and reduces supply current to below 10µA, while pulling SHDN high or leaving it floating puts the device into the operational mode. A conversion is initiated by CS falling. The conversion result is available at DOUT in unipolar serial format. A high bit, signaling the end of conversion (EOC), followed by the data bits (MSB first), make up the serial data stream.

The MAX187 operates in one of two states: (1) internal reference and (2) external reference. Select internal reference operation by forcing SHDN high, and external reference operation by floating SHDN.

Analog Input

Figure 4 illustrates the sampling architecture of the ADC's analog comparator. The full-scale input voltage depends on the voltage at REF.

REFERENCE	ZERO SCALE	FULL SCALE
Internal Reference (MAX187 only)	0V	+4.096V
External Reference	0V	V_{REF}

For specified accuracy, the external reference voltage range spans from +2.5V to V_{DD}.

+5V, Low-Power, 12-Bit Serial ADCs

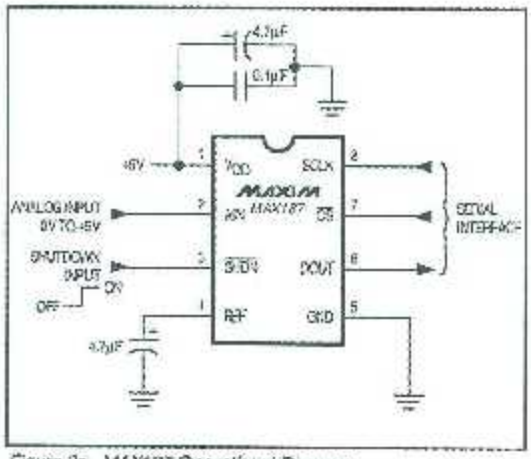


Figure 3a. MAX187 Operational Diagram

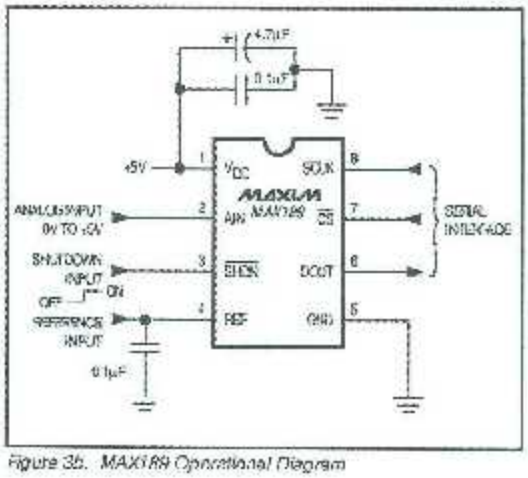


Figure 3b. MAX189 Operational Diagram

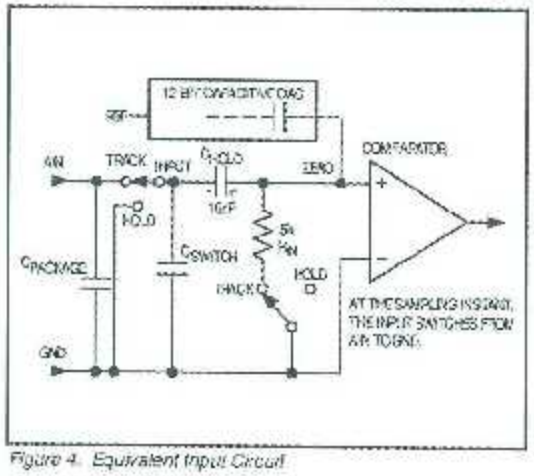


Figure 4. Equivalent Input Circuit

Track/Hold

In track mode, the analog signal is acquired and stored in the internal hold capacitor. In hold mode, the T/H switch opens and maintains a constant input to the ADC's SAR section.

During acquisition, the analog input AIN charges capacitor C_{HOLD}. Bringing $\overline{CS}$ low ends the acquisition interval. At this instant, the T/H switches the input side of C_{HOLD} to GND. The retained charge on C_{HOLD} represents a sample of the input, unbalancing the node ZERO at the comparator's input.

In hold mode, the capacitive DAC adjusts during the remainder of the conversion cycle to restore node ZERO to 0V within the limits of a 12-bit resolution. This action is equivalent to transferring a charge from C_{HOLD} to the binary-weighted capacitive DAC, which in turn forms a digital representation of the analog input signal. At the conversion's end, the input side of C_{HOLD} switches back to AIN, and C_{HOLD} charges to the input signal again.

The time required for the T/H to acquire an input signal is a function of how quickly its input capacitance is charged. If the input signal's source impedance is high, the acquisition time lengthens and more time must be allowed between conversions. Acquisition time is calculated by:

$$t_{ACQ} = 9\,(R_S + R_{IN})\,16pF,$$

where R_{IN} = 5kΩ, R_S = the source impedance of the input signal, and t_{ACQ} is never less than 1.5µs. Source impedances below 5kΩ do not significantly affect the AC performance of the ADC.

+5V, Low-Power, 12-Bit Serial ADCs

Input Bandwidth

The ADCs' input tracking circuitry has a 4.5MHz small-signal bandwidth, and an 8V/µs slew rate. It is possible to digitize high-speed transient events and measure periodic signals with bandwidths exceeding the ADC's sampling rate by using undersampling techniques. To avoid aliasing of unwanted high-frequency signals into the frequency band of interest, an anti-alias filter is recommended. See the MAX274/MAX275 continuous-time filters data sheet.

Input Protection

Internal protection diodes that clamp the analog input allow the input to swing from GND - 0.3V to V_{DD} + 0.3V without damage. However, for accurate conversions near full scale, the input must not exceed V_{DD} by more than 50mV, or be lower than GND by 50mV.

If the analog input exceeds the supplies by more than 50mV beyond the supplies, limit the input current to 2mA, since larger currents degrade conversion accuracy.

Driving the Analog Input

The input lines to AIN and GND should be kept as short as possible to minimize noise pickup. Shield longer leads. Also see the *Input Protection* section.

Because the MAX187/MAX189 incorporate a T/H, the drive requirements of the op amp driving AIN are less stringent than those for a successive-approximation ADC without a T/H. The typical input capacitance is 16pF. The amplifier bandwidth should be sufficient to handle the frequency of the input signal. The MAX400 and OP07 work well at lower frequencies. For higher-frequency operation, the MAX427 and OP27 are practical choices. The allowed input frequency range is limited by the 75ksps sample rate of the MAX187/MAX189. Therefore, the maximum sinusoidal input frequency allowed is 37.5kHz. Higher-frequency signals cause aliasing problems unless undersampling techniques are used.

Reference

The MAX187 can be used with an internal or external reference, while the MAX189 requires an external reference.

Internal Reference

The MAX187 has an on-chip reference with a buffered temperature-compensated bandgap diode, laser-trimmed to +4.096V ±0.5%. Its output is connected to REF and also drives the internal DAC. The output can be used as a reference voltage source for other components and can source up to 0.6mA. Decouple REF with a 4.7µF capacitor. The internal reference is enabled by pulling the SHDN pin high. Letting SHDN float disables the internal reference, which allows the use of an external reference, as described in the *External Reference* section.

External Reference

The MAX189 operates with an external reference at the REF pin. To use the MAX187 with an external reference, disable the internal reference by letting SHDN float. Stay within the voltage range +2.5V to V_{DD} to achieve specified accuracy. The minimum input impedance is 12kΩ for DC currents. During conversion, the external reference must be able to deliver up to 350µA DC load current and have an output impedance of 10Ω or less. The recommended minimum value for the bypass capacitor is 0.1µF. If the reference has higher output impedance or is noisy, bypass it close to the REF pin with a 4.7µF capacitor.

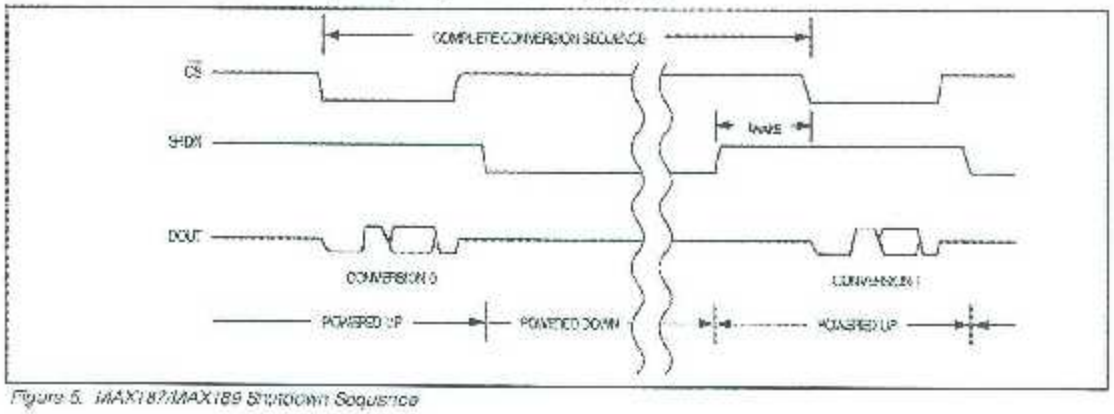


Figure 5. MAX187/MAX189 Shutdown Sequence

+5V, Low-Power, 12-Bit Serial ADCs

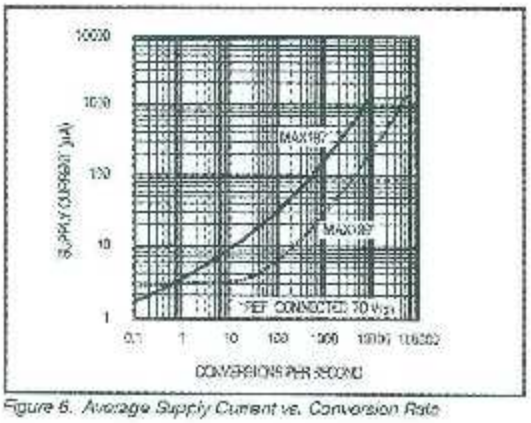


Figure 6. Average Supply Current vs. Conversion Rate

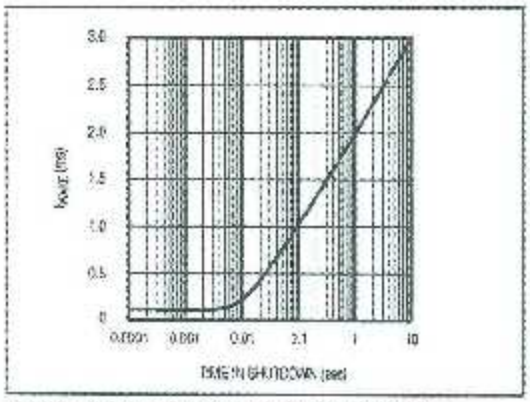


Figure 7. tWAKE vs. Time in Shutdown (MAX187 only)

Serial Interface

Initialization After Power-Up and Starting a Conversion

When power is first applied, it takes the fully discharged 4.7µF reference bypass capacitor up to 20ms to provide adequate charge for specified accuracy. With SHDN not pulled low, the MAX187/MAX189 are now ready to convert.

To start a conversion, pull CS low. At CS's falling edge, the T/H enters its hold mode and a conversion is initiated. After an internally timed 8.5µs conversion period, the end of conversion is signaled by DOUT pulling high. Data can then be shifted out serially with the external clock.

Using SHDN to Reduce Supply Current

Power consumption can be reduced significantly by shutting down the MAX187/MAX189 between conversions. This is shown in Figure 6, a plot of average supply current vs. conversion rate. Because the MAX189 uses an external reference voltage (assumed to be present continuously), it 'wakes up' from shutdown more quickly, and therefore provides lower average supply currents. The wakeup-time, tWAKE, is the time from SHDN deasserted to the time when a conversion may be initiated. For the MAX187, this time is 2µs. For the MAX189, this time depends on the time in shutdown (see Figure 7) because the external 4.7µF reference bypass capacitor loses charge slowly during shutdown (see the specifications for shutdown, REF Input Current = 10µA max).

External Clock

The actual conversion does not require the external clock. This frees the µP from the burden of running the SAR conversion clock, and allows the conversion result to be read back at the µP's convenience at any clock rate from 0MHz to 5MHz. The clock duty cycle is unrestricted if each clock phase is at least 100ns. Do not run the clock while a conversion is in progress.

Timing and Control

Conversion-start and data-read operations are controlled by the CS and SCLK digital inputs. The timing diagrams of Figures 8 and 9 outline the operation of the serial interface.

A CS falling edge initiates a conversion sequence: The T/H stage holds input voltage, the ADC begins to convert, and DOUT changes from high impedance to logic low. SCLK must be kept inactive during the conversion. An internal register stores the data when the conversion is in progress.

End of conversion (EOC) is signaled by DOUT going high. DOUT's rising edge can be used as a framing signal. SCLK shifts the data out of this register any time after the conversion is complete. DOUT transitions on SCLK's falling edge. The next falling clock edge produces the MSB of the conversion at DOUT, followed by the remaining bits. Since there are 12 data bits and one leading high bit, at least 13 falling clock edges are needed to shift out these bits. Extra clock pulses occurring after the conversion result has been clocked out, and prior to a rising edge of CS, produce trailing 0s at DOUT and have no effect on converter operation.

+5V, Low-Power, 12-Bit Serial ADCs

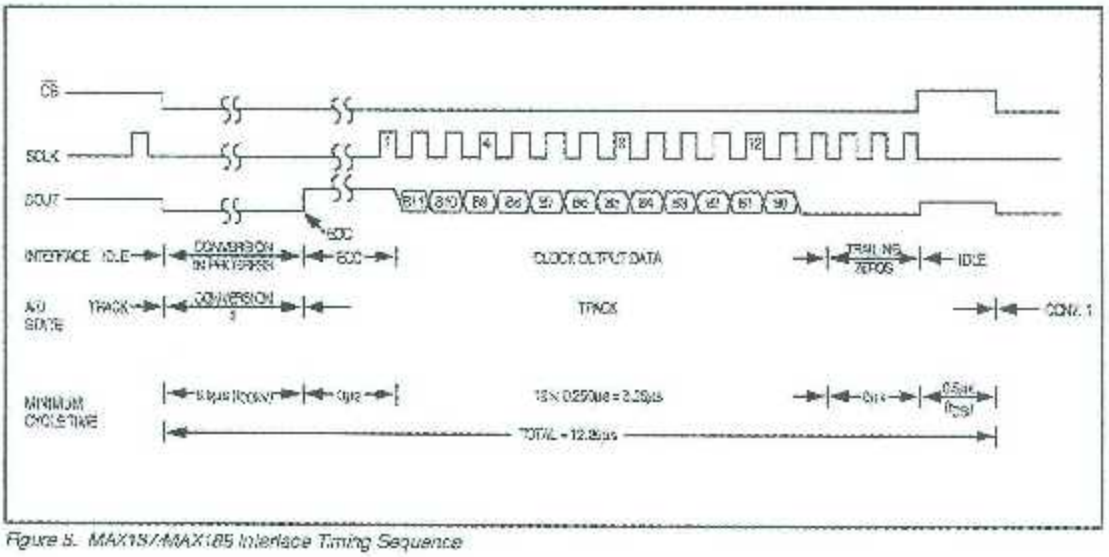


Figure 8. MAX187/MAX189 Interface Timing Sequence

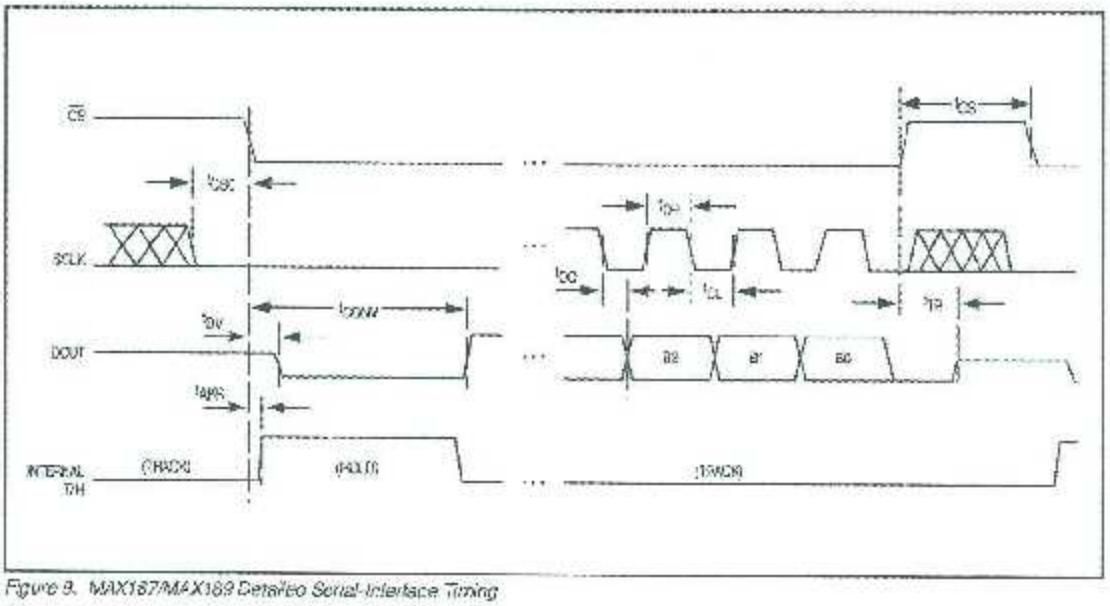


Figure 9. MAX187/MAX189 Detailed Serial-Interface Timing

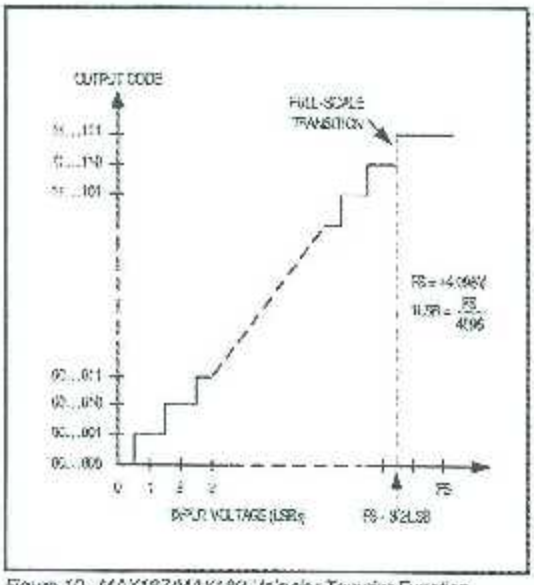


Figure 10. MAX187/MAX189 Unipolar Transfer Function, 4.096V = Full Scale

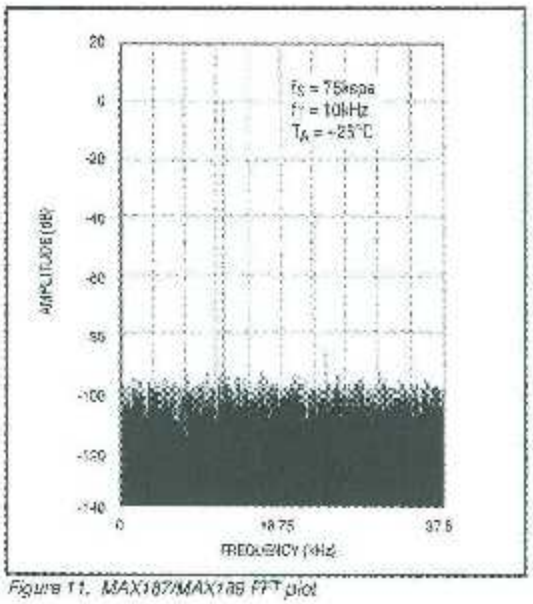


Figure 11. MAX187/MAX189 FFT plot

Minimum cycle time is accomplished by using DOUT's rising edge as the EOC signal. Clock out the data with 13 clock cycles at full speed. Raise CS after the conversion's LSB has been read. After the specified minimum time, t_{ACQ}, CS can be pulled low again to initiate the next conversion.

Output Coding and Transfer Function

The data output from the MAX187/MAX189 is binary, and Figure 10 depicts the nominal transfer function. Code transitions occur halfway between successive integer LSB values. If V_{REF} = +4.096V, then 1 LSB = 1.00mV or 4.096V/4096.

Dynamic Performance

High-speed sampling capability and a 75ksps throughput make the MAX187/MAX189 ideal for wideband signal processing. To support these and other related applications, Fast Fourier Transform (FFT) test techniques are used to guarantee the ADC's dynamic frequency response, distortion, and noise at the rated throughput. Specifically, this involves applying a low-distortion sine wave to the ADC input and recording the digital conversion results for a specified time. The data is then analyzed using an FFT algorithm that determines its spectral content. Conversion errors are then seen as spectral elements outside of the fundamental input frequency. ADCs have traditionally been evaluated by specifications such as Zero and Full-Scale Error, Integral Nonlinearity (INL), and Differential Nonlinearity (DNL). Such parameters are widely accepted for specifying performance with DC and slowly varying signals, but are less useful in signal-processing applications, where the ADC's impact on the system transfer function is the main concern. The significance of various DC errors does not translate well to the dynamic case, so different tests are required.

Signal-to-Noise Ratio and Effective Number of Bits

Signal-to-noise plus distortion (SINAD) is the ratio of the fundamental input frequency's RMS amplitude to the RMS amplitude of all other ADC output signals. The input bandwidth is limited to frequencies above DC and below one-half the ADC sample (conversion) rate.

The theoretical minimum ADC noise is caused by quantization error and is a direct result of the ADC's resolution: SINAD = (6.02N + 1.76)dB, where N is the number of bits of resolution. An ideal 12-bit ADC can, therefore, do no better than 74dB. An FFT plot of the output shows the output level in various spectral bands. Figure 11 shows the result of sampling a pure 10kHz sine wave at a 75ksps rate with the MAX187/MAX189.

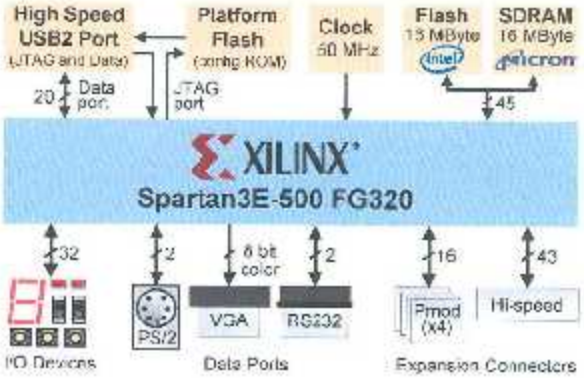
Digilent Nexys2 Board Reference Manual

Revision: June 21, 2008

DIGILENT®
www.digilentinc.com
215 E Main Suite D | Pullman, WA 99163
(509) 334 6306 Voice and Fax

Overview

The Nexys2 circuit board is a complete, ready-to-use circuit development platform based on a Xilinx Spartan 3E FPGA. Its on-board high speed USB2 port, 16Mbytes of RAM and ROM, and several I/O devices and ports make it an ideal platform for digital systems of all kinds, including embedded processor systems based on Xilinx's MicroBlaze. The USB2 port provides board power and a programming interface, so the Nexys2 board can be used with a notebook computer to create a truly portable design station.



The Nexys2 brings leading technologies to a platform that anyone can use to gain digital design experience. It can host countless FPGA-based digital systems, and designs can easily grow beyond the board using any or all of the five expansion connectors. Four 12-pin Peripheral Module (Pmod) connectors can accommodate up to eight low-cost Pmods to add features like motor control, A/D and D/A conversion, audio circuits, and a host of sensor and actuator interfaces. All user-accessible signals on the Nexys2 board are ESD and short-circuit protected, ensuring a long operating life in any environment.

The Nexys2 board is fully compatible with all versions of the Xilinx ISE tools, including the free WebPack. Now anyone can build real digital systems for less than the price of a textbook.

- *500K-gate Xilinx Spartan 3E FPGA*
- *USB2-based FPGA configuration and high-speed data transfers (using the free Adept Suite Software)*
- *USB-powered (batteries and/or wall-plug can also be used)*
- *16MB of Micron PSDRAM & 16MB of Intel StrataFlash ROM*
- *Xilinx Platform Flash for nonvolatile FPGA configurations*
- *Efficient switch-mode power supplies (good for battery powered applications)*
- *50MHz oscillator plus socket for second oscillator*
- *60 FPGA I/O's routed to expansion connectors (one high speed Hirose FX2 connector and four 6-pin headers)*
- *8 LEDs, 4-digit 7-seg display, 4 buttons, 8 slide switches*
- *Ships in a plastic carry case with USB cable*

Figure 1: Nexys2 block diagram and features

Power Supplies

The Nexys2 board input power input bus can be driven from a USB cable, from a 5VDC-15VDC, center positive, 2.1mm wall-plug supply, or from a battery pack. A shorting block loaded on the "power select" jumper selects the power source. The USB circuitry is always powered from the USB cable - if no USB cable is attached, the USB circuitry is left unpowered.

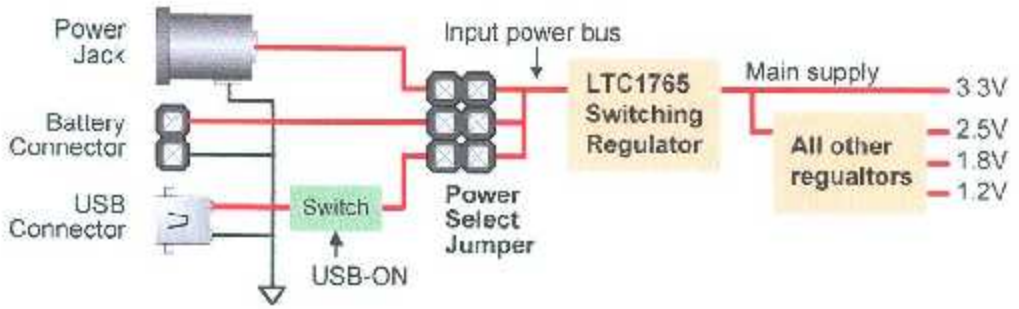


Figure 2: Nexys2 power supply block diagram

The input power bus drives a 3.3V voltage regulator that supplies all required board current. Some devices require 2.5V, 1.8V, and 1.2V supplies in addition to the main 3.3V supply, and these additional supplies are created by regulators that take their input from the main 3.3V supply. The primary supplies are generated by highly efficient switching regulators from Linear Technology. These regulators not only use USB power efficiently, they also allow the Nexys2 to run from battery packs for extended periods.

Total board current depends on the FPGA configuration, clock frequency, and external connections. In test circuits with roughly 20K gates routed, a 50MHz clock source, and all LEDs illuminated, about 200mA of current is drawn from the 1.2V supply, 50mA from the 2.5V supply, and 100mA from the 3.3V supply. Required current will increase if larger circuits are configured in the FPGA, and if peripheral boards are attached. The table above summarizes the power supply parameters.

The Nexys2 board can also receive power from (or deliver power to) a peripheral board connected to a Pmod connector or to the large 100-pin expansion connector. Jumpers near the Pmod connectors and large expansion connector (JP1 – JP5) can connect the Nexys2's input power bus to the connector's power pins. The Pmod jumpers can be used to route either the input power bus or regulated 3.3V to the Pmod power pins, while the expansion connector jumper can only make or break a connection with the input power bus.

USB power is supplied to the USB circuitry directly, but to the rest of the board through an electronic switch (Q1 in the Nexys2 schematic). The on-board USB controller turns on switch Q1 only after informing the host PC that

Table 1: Nexys2 Power Supplies

Supply	Device	Amps (max/typ)
3.3V main	IC6: LTC1765	3A/100mA
2.5V FPGA	IC7: LTC3417	1.4A/50mA
1.2V FPGA	IC7: LTC3417	1.4A/200mA
1.8V SRAM	IC8: LTC1844	150mA/90mA
3.3V USB	IC5: LTC1844	150mA/60mA

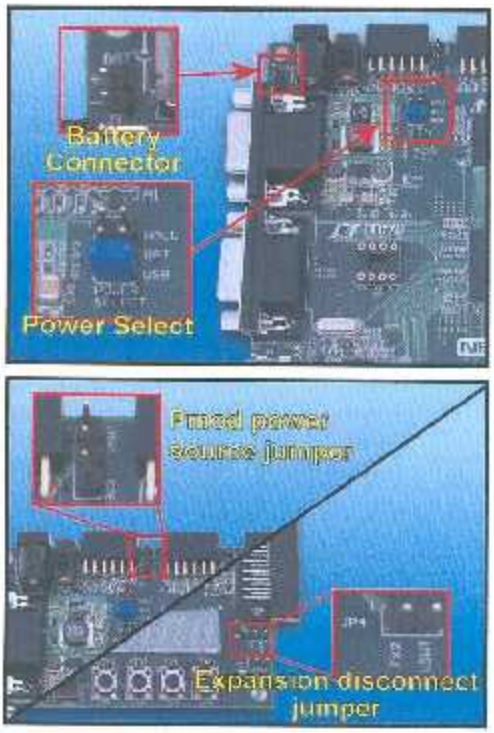


Figure 3: Nexys2 power supply jumpers

more than 100mA will be drawn through the USB cable (as required by the USB specification). A USB host can supply only 500mA of current at 5VDC. When using USB power, care must be taken to ensure the Nexys2 board and any attached peripheral boards do not draw more than 500mA, or damage to the host may result. The Nexys2 board typically consumes about 300mA of USB current, leaving about 200mA for peripheral boards. If peripheral boards require more current than the USB cable can supply, an external power supply should be used.

The Nexys2 board uses a six layer PCB, with the inner layers dedicated to VCC and GND planes. The FPGA and the other ICs on the board all have a large complement of bypass capacitors placed as close as possible to each VCC pin. The power supply routing and bypass capacitors result in a very clean, stable, and low-noise power supply.

FPGA and Platform Flash Configuration

The FPGA on the Nexys2 board must be configured (or programmed) by the user before it can perform any functions. During configuration, a "bit" file is transferred into memory cells within the FPGA to define the logical functions and circuit interconnects. The free ISE/WebPack CAD software from Xilinx can be used to create bit files from VHDL, Verilog, or schematic-based source files.

The FPGA can be programmed in two ways: directly from a PC using the on-board USB port, and from an on-board Platform Flash ROM (the Flash ROM is also user-programmable via the USB port). A jumper on the Nexys2 board determines which source (PC or ROM) the FPGA will use to load its configuration. The FPGA will automatically load a configuration from the Platform Flash ROM at power-on if the configuration Mode jumper is set to "Master serial". If the Mode jumper is set to "JTAG", the FPGA will await programming from the PC (via the USB cable).

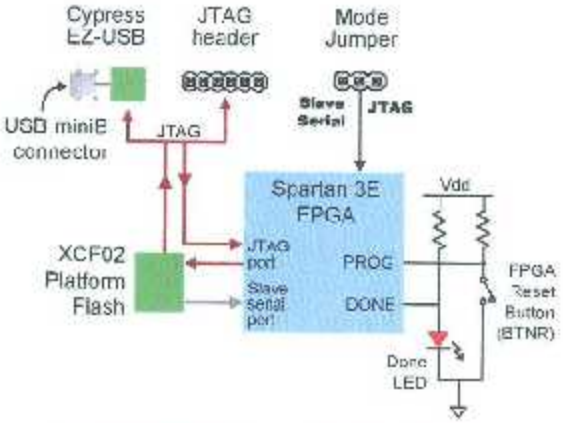


Figure 4: Nexys2 programming circuits

Digilent's freely available PC-based Adept software can be used to configure the FPGA and Platform Flash with any suitable file stored on the computer. Adept uses the USB cable to transfer a selected bit file from the PC to the FPGA or Platform Flash ROM. After the FPGA is configured, it will remain so until it is reset by a power-cycle event or by the FPGA reset button (BTNR) being pressed. The Platform Flash ROM will retain a bit file until it is reprogrammed, regardless of power-cycle events.

To program the Nexys2 board using Adept, attach the USB cable to the board (if USB power will not be used, attach a suitable power supply to the power jack or battery connector on the board, and set the power switch to "wall" or "bat"). Start the Adept software, and wait for the FPGA and the Platform Flash ROM to be recognized. Use the browse function to associate the desired .bit file with the FPGA, and/or the desired .mcs file with the Platform Flash ROM. Right-click on the device to be programmed, and select the "program" function. The configuration file will be sent to the FPGA or Platform Flash, and the software will indicate whether programming was successful. The configuration

"done" LED will illuminate after the FPGA has been successfully configured. For further information on using Adept, please see the Adept documentation available at the Digilent website.

The Nexys2 board can also be programmed using Xilinx's iMPACT software by connecting a suitable programming cable to the JTAG header. Digilent's JTAG3 cable or any other Xilinx cable may be used.

A demonstration configuration is loaded into the Platform Flash on the Nexys2 board during manufacturing. That configuration, also available on the Digilent webpage, can be used to check all of the devices and circuits on the Nexys2 board.

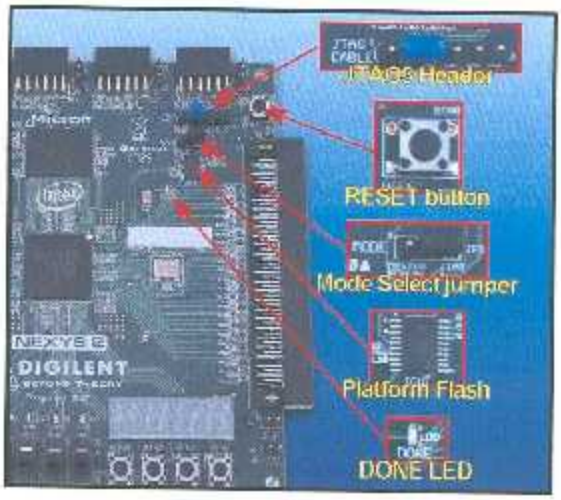


Figure 5: Nexys2 board programming circuits

Clocks

The Nexys2 board includes a 50MHz oscillator and a socket for a second oscillator. Clock signals from the oscillators connect to global clock input pins on the FPGA so they can drive the clock synthesizer blocks available in FPGA. The clock synthesizers (called DLLs, or delay locked loops) provide clock management capabilities that include doubling or quadrupling the input frequency, dividing the input frequency by any integer multiple, and defining precise phase and delay relationships between various clock signals.

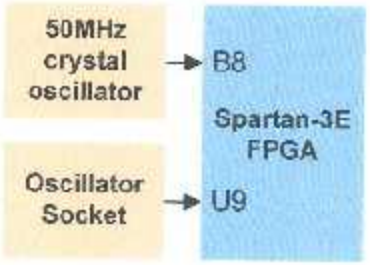


Figure 6: Nexys2 clocks

User I/O

The Nexys2 board includes several input devices, output devices, and data ports, allowing many designs to be implemented without the need for any other components.

Figure 7: Nexys2 board I/O devices

Inputs: Slide Switches and Pushbuttons

Four pushbuttons and eight slide switches are provided for circuit inputs. Pushbutton inputs are normally low, and they are driven high only when the pushbutton is pressed. Slide switches generate constant high or low inputs depending on their position. Pushbutton and slide switch inputs use a

series resistor for protection against short circuits (a short circuit would occur if an FPGA pin assigned to a pushbutton or slide switch was inadvertently defined as an output).

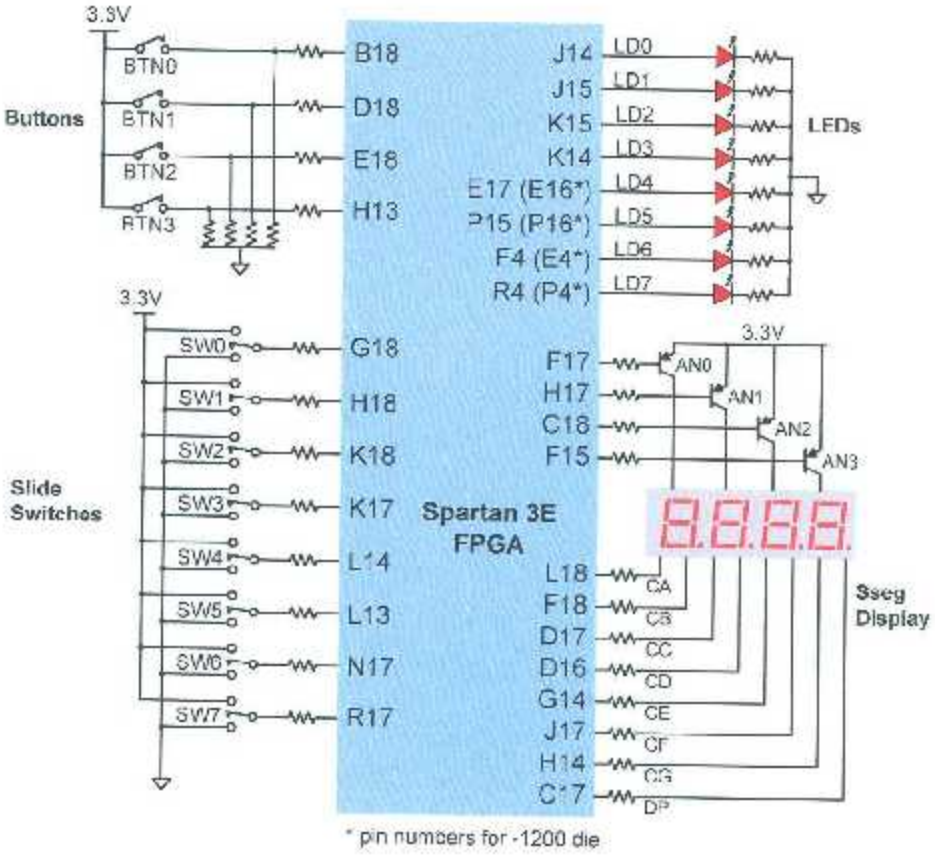


Figure 8: Nexys2 I/O devices and circuits

Outputs: LEDs

Eight LEDs are provided for circuit outputs. LED anodes are driven from the FPGA via 390-ohm resistors, so a logic '1' output will illuminate them with 3-4ma of drive current. A ninth LED is provided as a power-on LED, and a tenth LED indicates FPGA programming status. Note that LEDs 4-7 have different pin assignments due to pinout differences between the -500 and the -1200 die.

Outputs: Seven-Segment Display

The Nexys2 board contains a four-digit common anode seven-segment LED display. Each of the four digits is composed of seven segments arranged in a "figure 8" pattern, with an LED embedded in each segment. Segment LEDs can be individually illuminated, so any one of 128 patterns can be displayed on a digit by illuminating certain LED segments and leaving the others dark. Of these 128 possible patterns, the ten corresponding to the decimal digits are the most useful.

Peripheral Connectors

The Nexys2 board provides four two-row 6-pin Pmod connectors that together can accommodate up to 8 Pmods. The four 12-pin connectors each have 8 data signals, two GND pins, and two Vdd pins. All data signals include short circuit protection resistors and ESD protection Diodes. A jumper block adjacent to each Pmod connector can connect the Pmod's Vdd signal to the Nexys2 board's 3.3V supply or to the input power bus (VU). If the jumper is set to VU and USB power is driving the main power bus, care should be taken to ensure no more than 200mA is consumed by the Pmod. Further, if the jumper is set to VU, a voltage source connected to the Pmod can drive the main power bus of the Nexys2 board, so care should be taken to avoid connecting conflicting power supplies.

The Pmod connectors are labeled JA (nearest the power jack), JB, JC, and JD (nearest the expansion connector). Pinouts for the Pmod connectors are provided in the table below.

More than 30 low-cost are available for attachment to these connectors. Pmods can either be attached directly, or by using a small cable. Available Pmods include A/D and D/A converters, motor drivers, speaker amplifiers, distance measuring devices, etc. Please see www.digilentinc.com for more information.

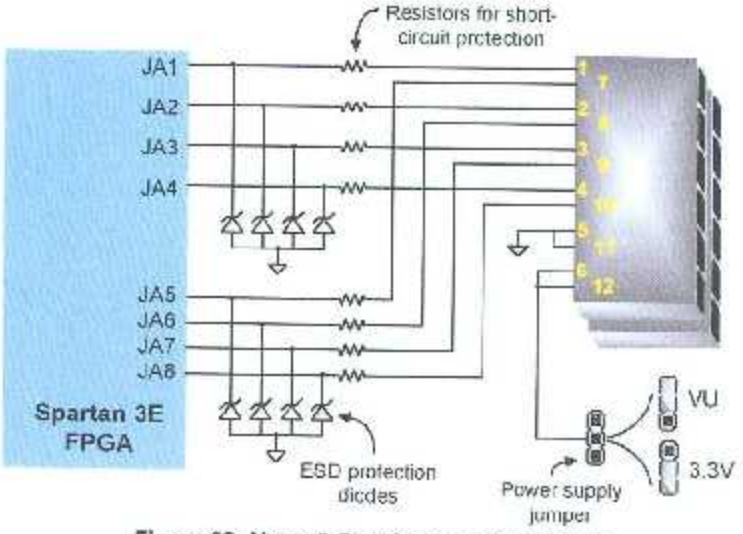


Figure 23: Nexys2 Pmod connector circuits

Table 3: Nexys2 Pmod Connector Pin Assignments

Pmod JA		Pmod JB		Pmod JC		Pmod JD	
JA1: L15	JA7: K13	JB1: M13	JB7: P17	JC1: G15	JC7: H15	JD1: J13	JD7: K14[1]
JA2: K12	JA8: L16	JB2: R18	JB8: R16	JC2: J16	JC8: F14	JD2: M18	JD8: K15[2]
JA3: L17	JA9: M14	JB3: R15	JB9: T18	JC3: G13	JC9: G16	JD3: N18	JD9: J15[3]
JA4: M15	JA10: M16	JB4: T17	JB10: U18	JC4: H16	JC10: J12	JD4: P18	JD10: J14[4]

Notes: [1] shared with LD3 [2] shared with LD3 [3] shared with LD3 [4] shared with LD3