# DISEÑO E IMPLEMENTACION DE UNA TARJETA DE ADQUISICION Y CONTROL PARA PUERTO USB 2.0 UTILIZANDO UN FPGA


**JORGE ALEJANDRO PANTOJA JARAMILLO**
**JAIRO MONTILLA MUÑOZ**

**UNIVERSIDAD DEL CAUCA**
**FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES**
**DEPARTAMENTO DE ELECTRONICA INSTRUMENTACION Y CONTROL**
**POPAYÁN**
**2007**

# DISEÑO E IMPLEMENTACION DE UNA TARJETA DE ADQUISICION Y CONTROL PARA PUERTO USB 2.0 UTILIZANDO UN FPGA

**JORGE ALEJANDRO PANTOJA JARAMILLO**
**JAIRO MONTILLA MUÑOZ**

**Monografía para optar al titulo de**
**Ingeniero en Electrónica y Telecomunicaciones**

Director:
**Ing. Vladimir Trujillo Arias**

**UNIVERSIDAD DEL CAUCA**
**FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES**
**DEPARTAMENTO DE ELECTRONICA INSTRUMENTACION Y CONTROL**
**POPAYÁN**
**2007**

# INTRODUCCIÓN

La evolución es la tendencia natural del hombre y por lo tanto también lo es de las cosas que el crea. En este proceso muchas cosas tienden a desaparecer y otras cambian, realizándose mejoras sobre ellas.

En el campo de la electrónica es más que notoria la evolución de todo lo que le involucre. La evolución de las tarjetas de adquisición de datos es una muestra de ello en sus fases primarias con el puerto ISA, migrando luego al puerto PCI y ahora explorando las opciones que ofrece el puerto USB.

Este trabajo pretende dejar las bases de la exploración en las tecnologías USB y además desarrollar una tarjeta de adquisición de datos que aproveche los beneficios que el protocolo ofrece y que con los buses ISA o PCI no se consiguen. Un ejemplo de esos beneficios es la posibilidad de conexión o desconexión en caliente, la no necesidad de destapar el equipo para la conexión de la tarjeta, la portabilidad al estar vinculada a puerto de uso común y al ser un dispositivo externo, las grandes velocidades que se pueden conseguir gracias a la versión 2.0 del protocolo USB, entre otras.

A la vez se busca que la tarjeta pueda adaptarse fácilmente a las necesidades del ambiente industrial y de control, característica que se consigue con la utilización de los FPGAs. La posibilidad de cambiar radicalmente la funcionalidad de un dispositivo o la viabilidad de agregar nuevos elementos que permitan extender el uso del mismo son una característica muy deseada. El uso de los FPGA para tales fines y el diseño modular que se proyecta para la tarjeta de adquisición de datos serán un gran aporte a las bases de conocimiento y desarrollo que ya se tienen en el Departamento de Electrónica Instrumentación y Control de la universidad.

# 1 FUNDAMENTOS TEÓRICOS

## 1.1 INTRODUCCIÓN

El hombre pretende siempre que sus problemas se solucionen de manera sencilla, para ello se basa en el principio de escalabilidad. En la computación este paradigma se sigue cumpliendo y la especificación USB se ha propuesto como una solución a aquella tediosa labor de conectar dispositivos nuevos a un PC al que se le piensa brindar una nueva funcionalidad. En lo sucesivo de este capitulo se presenta una detallada descripción del protocolo USB, así como también un compendio de la teoría sobre tarjetas de adquisición de datos, ambos aspectos importantes para el desarrollo de esta monografía.

## 1.2 BUS SERIAL UNIVERSAL USB

El USB (Universal Serial Bus) o Bus Serial Universal es una interfaz para la transmisión serie de datos desarrollado por empresas líderes del sector de las telecomunicaciones y de la computación, entre ellas Compaq, Digital Equipment Corp, IBM, Intel, Microsoft, NEC y Northern Telecom. Fue introducida en el mercado de los PC´s y periféricos para mejorar las lentas interfaces serie (RS-232) y paralelo y los inconvenientes en la reconfiguración del hardware presentes en las anteriores arquitecturas de bus como la PCI, ISA y PCMCIA. Provee una mayor velocidad de transferencia (de hasta 100 veces más rápido) comparado con el puerto paralelo y el serial que son los puertos que se encuentran en la mayoría de los computadores. Tenía en un principio como objetivo el conectar periféricos relativamente lentos (ratones, impresoras, cámaras digitales, unidades ZIP, etc.)

de una forma realmente sencilla, rápida y basada en comunicaciones serie, aunque por sus características también podía conectarse hasta discos duros.
Esta interfaz de 4 hilos también distribuye 5V para la alimentación y puede transmitir datos a una velocidad de hasta 480 Mb/s en su versión 2.0. Es un bus serie que hace posible la conexión de hasta 127 periféricos a un único puerto de un PC, con detección y configuración automáticas, siendo esto posible con el PC encendido, sin tener que instalar software adicional, y sin tener que reiniciar el computador (plug and play), algo que con los puertos convencionales serie y paralelo no sucedía. Tampoco hay que preocuparse por conflictos de IRQs, instalar tarjetas adaptadoras para cada periférico o destapar el PC. Estos periféricos pueden ser: ratones, teclados, impresoras, escáneres, grabadoras de CD/DVD, discos duros, módems, cámaras digitales, etc.

El éxito de la interfaz USB ha sido tal que, actualmente todos los PCs tienen integrados al menos dos puertos USB para la conexión de dispositivos. A falta de puertos se pueden acoplar hubs USB para ampliar el número de dispositivos siempre que no se sobrepase el límite soportado de 127.

### 1.2.1 Características del USB

La especificación del USB proporciona una serie de características que pueden ser distribuidas en categorías. Estas características son comunes para todas las versiones (desde la 1.0 hasta la 2.0)

**Facilidad de uso para los usuarios:**

- Modelo simple para el cableado y los conectores
- Detalles eléctricos aislados del usuario (terminaciones del bus)
- Periféricos auto-reconocibles
- Periféricos reconfigurables y de conexión dinámica (Hot Swappable)

**Flexibilidad:**

- Amplio rango de tamaños de paquetes, permitiendo variedad de opciones de buffering de dispositivos
- Gran variedad de tasas de datos de dispositivos acomodando el tamaño del buffer para los paquetes y las latencias
- Control de flujo para el manejo del buffer construido en el protocolo

**Ancho de banda isócrono:**

- Se garantiza un ancho de banda y un flujo constante de datos con bajas latencias apropiadas para telefonía, audio y video.

**Amplia gama de aplicaciones y cargas de trabajo**

- Adecuando el ancho de banda desde unos pocos kb/s hasta varios Mb/s
- Soporta tanto el tipo de transferencia isócrono como el asíncrono sobre el mismo conjunto de cables.
- Conexiones múltiples, soportando operaciones concurrentes de varios dispositivos.
- Soporta hasta 127 dispositivos físicos.
- Soporta la transferencia de múltiples datos y flujos de mensajes entre el PC y los dispositivos

**Robustez**

- Manejo de errores y mecanismos de recuperación ante fallos implementados en el protocolo.
- Inserción dinámica de dispositivos
- Soporte para la identificación de dispositivos defectuosos.

**Implementación de bajo costo**

- Subcanal de bajo costo a 1.5 Mb/s
- Conectores y cables de bajo costo
- Adecuado para el desarrollo de periféricos de bajo costo

La tabla 1.1 muestra algunos de los detalles de cada una de las versiones del protocolo, que es lo que las diferencia unas de otras.

| Versión (año) | Rendimiento | | | Aplicaciones | Atributos |
|---|---|---|---|---|---|
| | Velocidad baja | Velocidad media | Velocidad alta | | |
| 1.0 (1996) | Dispositivos interactivos 10 - 100Kb/s | Telefonía, audio, video comprimido 500Kb/s - 10Mb/s | - | Teclados, ratones y dispositivos de juegos ISDD/RDSI | Bajo costo conexión en caliente, múltiples periféricos. Latencia y ancho de banda garantizados |
| 1.1 (1998) | Ídem | Ídem | - | Ídem | Soluciona problemas de ambigüedad de la versión 1.0 |
| 2.0 (2000) | Ídem | Ídem | Video, unidades de almacenamiento, video comprimido 25 – 500Mb/s | Video, discos duros y unidades de CD/DVD externos | Gran ancho de banda, latencia garantizada. |

**Tabla 1.1 Características Generales del USB**

Básicamente, USB 2.0 incluye todo lo que ofrece USB 1.1 y añade el modo de alta velocidad. USB 2.0 también usa el mismo tipo de cables y conectores para conectar los dispositivos de alta velocidad, sin embargo los hubs USB clásicos ralentizarán los dispositivos USB 2.0. Otro requisito es que es necesario un

controlador de host para USB 2.0 si queremos tener disponibles la conexión de alta velocidad con un dispositivo de este tipo.
Los hubs USB 2.0 tienen ahora mucho mas trabajo que hacer que en el USB 1.1 ya que necesitan manejar todo el tráfico de tres tipo de dispositivos con velocidades distintas. Conectando un dispositivo USB 1.1 en uno USB 2.0 funcionaría bien, pero no lo haría si lo hiciéramos al revés, además de ralentizarse a 12 Mb/s, posiblemente, el sistema operativo avisaría de su mal uso.

Cabe destacar que el USB 2.0 es simplemente una extensión y nunca llegara a reemplazar completamente al 1.1 ya que hay productos como teclados genéricos, ratones, joysticks o altavoces que no requieren la gran velocidad que ofrece la tecnología USB 2.0. Sólo dispositivos de alta velocidad como cámaras digitales o sistemas de alta capacidad de datos necesitarán el máximo de velocidad, aunque los PC los fabriquen sólo con puertos USB 2.0.

### 1.2.2 Topología del USB

La interconexión física USB consiste de una topología en estrella dispuesta en capas en donde un hub se encuentra en el centro de cada estrella. Cada capa consiste de una conexión punto a punto entre el host y un hub o función, o un hub conectado a otro hub o función. La figura 1.1 ilustra la topología del USB.

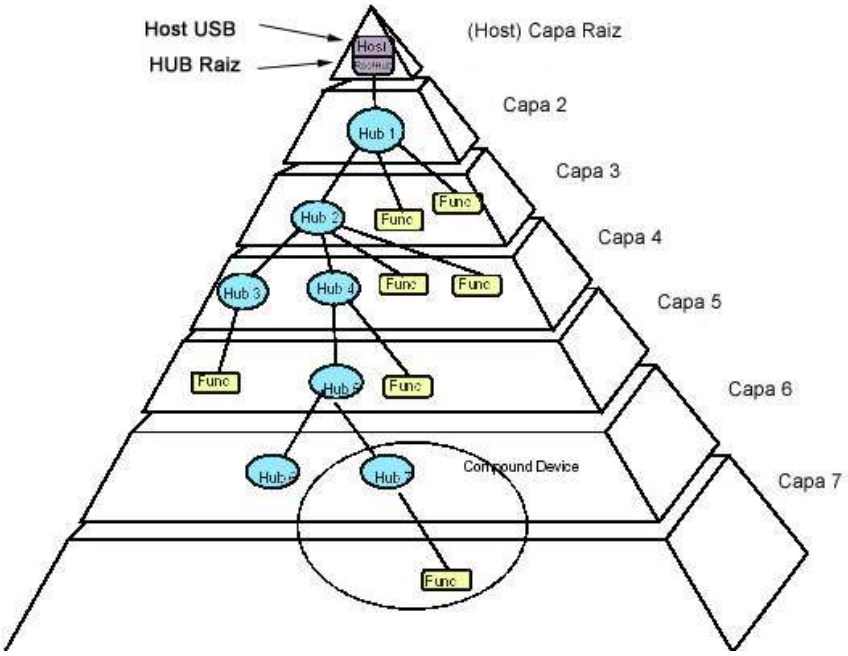



**Figura 1.1 Topología del USB**

El número máximo de dispositivos que puede conectar USB es de 127, pero debido a las constantes de tiempo permitidas para los tiempos de propagación del hub y el cable, el número máximo de capas permitido es de siete (incluida la capa raíz) con un máximo de longitud de cable entre el hub y el dispositivo de 5 metros.

La topología del bus USB se puede dividir en tres partes:

- La capa física: Como están conectados los elementos físicamente
- La capa lógica: Los roles y las responsabilidades de los elementos USB
- La relación software del cliente-función: Como se ven mutuamente el software del cliente y los interfaces de las funciones relacionadas

### 1.2.2.1 Capa física

Cada entorno físico USB esta compuesto por cinco tipos de componentes:

- El host
- El controlador del host

- Los enlaces
- Los dispositivos (hubs y funciones)

#### 1.2.2.1.1 El host

El host es el sistema de computación completo, incluyendo el software y el hardware, sobre el cual se sostiene el USB.

El host tiene la habilidad de procesar y gestionar los cambios de configuración que puedan ocurrir en el bus durante su funcionamiento. El host gestiona el sistema y los recursos del bus como el uso de la memoria del sistema, la asignación del ancho de banda del bus y la alimentación del bus. El host también ayuda al usuario con la configuración automática de los dispositivos conectados y reaccionando cuando son desconectados.

Un host puede soportar uno o más buses USB. El host gestiona cada bus independientemente de los demás. Los recursos específicos del bus como el ancho de banda asignado son únicos a cada bus. Cada bus está conectado al host a través de un controlador del host.

Sólo hay un host en cualquier sistema USB. La interfaz USB de un sistema computacional es conocido como el Controlador de Host y puede estar implementado como combinación de hardware, firmware o software. Integrado dentro del sistema de host hay un hub raíz que provee de un mayor número de puntos de acople al sistema.

#### 1.2.2.1.2 El controlador del host

El controlador de host está formado por el hardware y el software que permite a los dispositivos USB conectarse al host. Este controlador es el agente iniciador del bus, es decir es el que comienza las transferencias en el bus. El controlador de

bus es el maestro en un bus USB. Otros buses como PCI, permiten la presencia de múltiples maestros donde cada uno arbitra sus accesos al bus. El la arquitectura USB sólo hay un controlador de host por cada bus USB y por eso no hay arbitración para el acceso al bus. La figura 1.2 muestra una vista conceptual del controlador de bus USB.

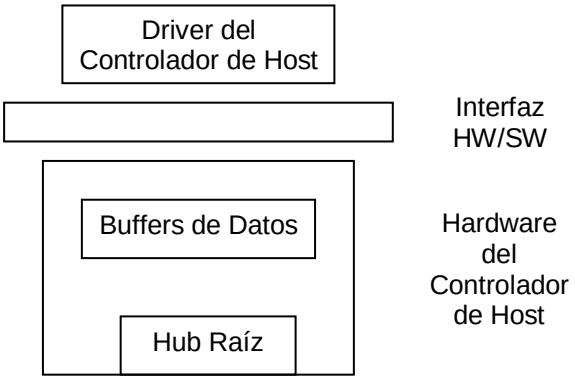



**Figura 1.2 Controlador de Host**

La parte software consiste en el driver del controlador de host (HCD). Este software interactúa con el hardware del controlador de host a través del interfaz hardware/software.

La parte hardware del controlador de host consiste en un hub raíz que proporciona los puertos USB y los buffers de datos (colas) donde son almacenados cuando son movidos hacia y desde la memoria.

El host USB interactúa con los dispositivos USB a través del controlador. Las funciones básicas del controlador de host son:

- Detectar la inserción o desconexión de dispositivos USB
- Gestionar el flujo de control entre el host y los dispositivos
- Gestionar el flujo de datos entre el host y los dispositivos

- Recopilar estadísticas de actividad y estado
- Proveer una cantidad limitada de energía a los dispositivos conectados

Hay dos implementaciones estandarizadas de la parte hardware de los controladores de host USB. Ambas proporcionan la misma funcionalidad y rendimiento para la interconexión. Estas implementaciones son:

- El Universal Host Controller Interface (UHCI) definido por Intel
- Open Host Controller Interface (OpenHCI o OHCI) definido por Microsoft

UHCI esta definido de tal forma que la parte software tiene una gran responsabilidad para mantener el hardware en funcionamiento. Esto permite a esta implementación ser relativamente simple y realizarse con un bajo número de puertas.

OHCI esta definido de tal forma que la parte hardware tiene más responsabilidad por el mantenimiento del flujo de datos, para que la parte software tenga menos trabajo que hacer. Esta otra implementación tiende a ser más compleja y tiene una cantidad mayor de puertas que la UHCI

#### 1.2.2.1.3 Los dispositivos

Un dispositivo es una colección de funcionalidad que lleva a cabo algún propósito de utilidad. Por ejemplo, un dispositivo podría ser un ratón, un teclado, una cámara, etc. Puede haber múltiples dispositivos simultáneamente en el mismo bus. Cada dispositivo lleva consigo información que puede ser útil para identificar sus características. Esta información se divide en tres categorías:

- Estándar: Esta es la información cuya definición es común a todos los dispositivos USB e incluye elementos como la identificación del fabricante, la clase, la gestión de energía.

- Clase: La definición de esta información varía dependiendo del aparato. Es una clasificación de los dispositivos en cuanto a sus prestaciones.

- USB Vendor: El fabricante del periférico puede poner aquí cualquier información deseada.

El software del host es capaz de determinar el tipo de dispositivo conectado haciendo uso de esta información y de un direccionamiento individual. Todos los dispositivos USB son accedidos por una dirección USB que es asignada dinámicamente cuando se conecta, mediante un proceso llamado enumeración. Cada aparato soporta además uno o más canales a través de los cuales el host puede comunicarse con el dispositivo. Una vez ha sido reconocido e identificado el dispositivo, el software del host puede hacer que los drivers del dispositivo apropiados obtengan el control del nuevo dispositivo conectado.

Cuando desconectamos el dispositivo, la dirección puede ser reutilizada para el próximo dispositivo conectado.

En cuanto a los tipos de dispositivos nos encontramos con dos clases:

- Hubs, que proporcionan los puntos de acople adicionales al USB.

- Funciones, que le dan al sistema la funcionalidad (HIDs, impresoras, unidades de almacenamiento, etc).

**Hubs**

Los hubs son un elemento clave en la arquitectura plug and play del USB. La figura 1.3 muestra un hub típico.

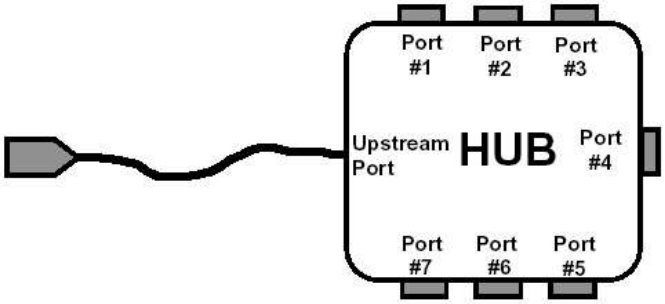



**Figura 1.3 Hub Típico**

Un bus tradicional puede ser dividido en segmentos de bus individuales conectados por puentes. Cada segmento tiene un número limitado de puntos de conexión debido a la limitada energía y la carga del bus. En la arquitectura USB, el número de puntos de conexión de dispositivos se expande añadiendo este tipo de dispositivos únicos. Los hubs expanden la arquitectura del USB de dos formas:

- Incrementando los puntos de conexión a través de puertos adicionales.
- Proporcionando energía a los dispositivos al expandir el bus.

Los Hubs sirven para simplificar la conectividad USB desde la perspectiva del usuario y proporcionar robustez y complejidad a un precio relativamente bajo. Según esta arquitectura, se pueden expandir el bus acoplando hubs adicionales, interconectando dichos hubs mediante enlaces. Cada hub convierte un punto simple de conexión en múltiples puntos, donde estos puntos de conexión se les llaman puertos.

Un hub requiere tener un puerto de subida (upstream port) y de 1 a N de bajada (downstream port).
El puerto de subida de un hub conecta el hub al host. Es el que eléctricamente está más cerca del controlador de host. Está numerado como el puerto 0.
Los puertos de bajada permiten la conexión a otro hub o a una función. Están numerados del 1 al N. Son los que más lejos están del controlador de host. Los

hubs pueden detectar conexiones y desconexiones en cada puerto de bajada y activar la distribución de energía para cada dispositivo de bajada. Cada puerto puede ser individualmente activado para cada dispositivo high-, full- o low-speed.

Los hubs tienen dos formas de obtención de la energía: a través del puerto de subida (bus-powered) o de una fuente externa (self-powered).

Un hub USB 1.x consiste en:

- El repetidor HUB, que es el responsable de gestionar la conectividad entre el puerto de subida y el de bajada, los cuales están operando a la misma velocidad. Actúa como repetidor de paquetes

- El controlador del hub hace posible el acceso del hub al host y viceversa. Por medio de este puede configurar el concentrador, monitorizar y controlar sus puertos.

Para la versión USB 2.0 hay un tercer elemento:

- Traductor de transacciones (TT) que proporciona los mecanismos que dan soporte a los dispositivos full-/low-speed tras el hub mientras se transmiten todos los datos entre el host y el hub en el modo hi-speed. Básicamente se usa para “traducir” las transacciones high-speed a transacciones full o low-speed.

En un bus USB 2.0, el host y todos los concentradores USB 2.0 intermedios transmiten siempre en modo high-speed, y sólo el tráfico entre el dispositivo full o low-speed y el concentrador USB 2.0 al que está conectado se produce a velocidad full o low-speed. De esta manera se pueden solapar las transferencias full y low-speed con transferencias high-speed con otros dispositivos, lo que

minimiza el impacto que las transmisiones a velocidades full y low-speed tienen sobre el ancho de banda disponible para los dispositivos high-speed.

Se puede decir que el concentrador mantiene aislados los segmentos del bus que usan full y low-speed de aquellos otros segmentos donde sólo se transmite en high-speed.

Gracias a los elementos incorporados por el hub, se consigue un comportamiento optimizado de la interfaz USB 2.0, de esta forma:

- Un hub USB 2.0 conectado a un puerto high-speed, siempre se comunica en modo high-speed en su puerto de flujo ascendente, mientras que permite la conectividad en sus puertos de flujo descendente tanto de dispositivos high-speed (a través del repetidor), como de dispositivos full/low-speed (a través del o de los TTs).

- Un hub USB 2.0 conectado a un puerto full-speed, siempre se comunica en modo full-speed en su puerto de flujo ascendente, y deshabilita el modo high-speed en todos sus puertos de flujo descendente. Quiere decir que los dispositivos high-speed conectados a este hub van a funcionar en modo full-speed. En este caso, las transferencias siempre se canalizan a través del repetidor, que se comporta como un repetidor USB 1.x (full/low-speed).

**Funciones**

Una función es un dispositivo USB que puede transmitir o recibir datos o información de control a través del bus. Una función se implementa típicamente como un dispositivo periférico separado con un cable que se conecta en un puerto o en un concentrador. Sin embargo, un dispositivo puede implementar funciones múltiples y poseer un concentrador embebido. Esto se conoce como un dispositivo compuesto y es visto por el host como un concentrador con uno o más dispositivos

Cada función contiene información de la configuración que describe sus capacidades y requerimientos de recursos. Antes de que una función pueda usarse, debe ser configurada por el host. Esta configuración incluye la asignación del ancho de banda USB y la selección de las opciones de configuración específicas de la función. Los siguientes son ejemplos de funciones:

- Dispositivo de interfaz humana (HID) como un ratón, teclado, tablas digitalizadoras, o un control de juegos.
- Dispositivo de imagines como un escáner, impresora, o cámara.
- Dispositivo de almacenamiento masivo como una unidad de CD/DVD, unidad de disco flexible, o una memoria removible.

#### 1.2.2.3 Capa lógica

El punto de vista lógico presenta capas y abstracciones que son relevantes para los distintos diseñadores e implementadores. La arquitectura lógica describe como unir el hardware del dispositivo USB a un driver del dispositivo en el host para que tenga el comportamiento que el usuario final desea.

La vista lógica de esta conexión es la mostrada en la figura 1.4. En el podemos ver como el host proporciona conexión al dispositivo, donde esta conexión es a través de un simple enlace USB. La mayoría de los demás buses como PCI, ISA, etc proporcionan múltiples conexiones al dispositivo y los drivers lo manipulan mediante algunas combinaciones de estas conexiones (I/O y direcciones de memoria, interrupciones y canales DMA).

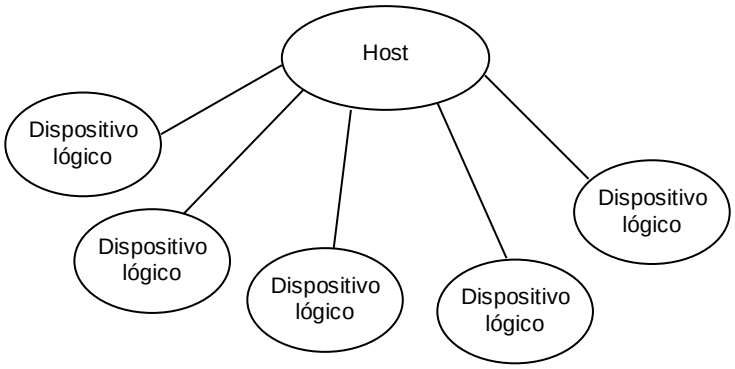



**Figura 1.4 Topología Lógica del USB**

Físicamente el USB tiene sólo un cable simple de bus que es compartido por todos los dispositivos del bus. Sin embargo, desde el punto de vista lógico cada dispositivo tiene su propia conexión punto a punto al host.

La Figure 1.5 muestra una apreciación global más profunda del bus USB, en ella se identifican las diferentes capas lógicas del sistema

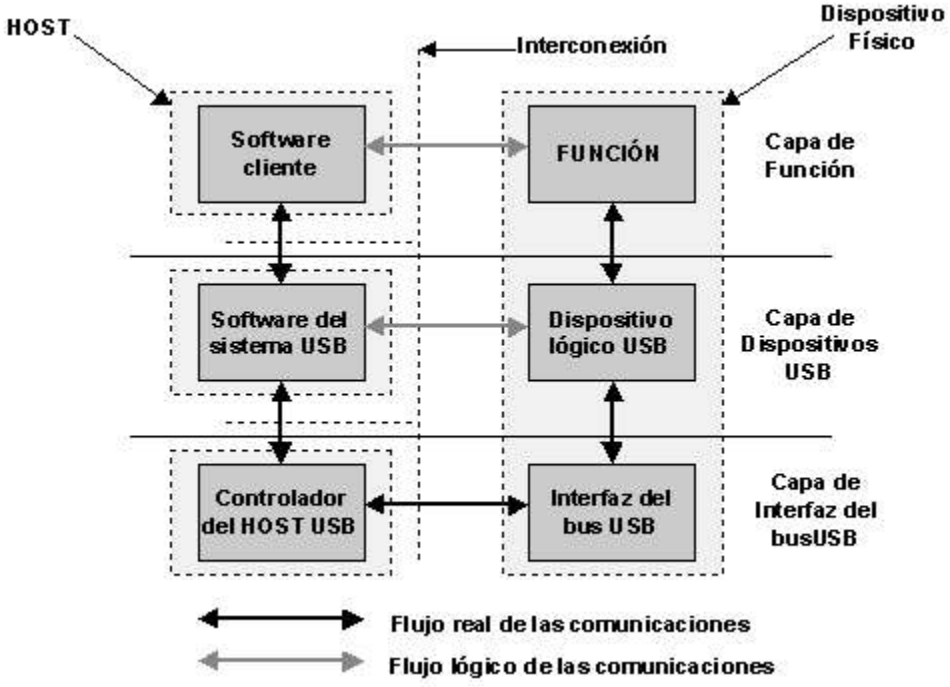



**Figura 1.5 Modelo Lógico USB**

En la figura 1.5 está materializada la conexión entre el controlador de host y un dispositivo o función. Este está constituido por hardware al final de un cable USB y realiza alguna función útil para el usuario.

El software cliente se ejecuta en el host y corresponde a un dispositivo USB; se suministra con el sistema operativo o con el dispositivo USB. El software del sistema USB, es el que soporta USB en un determinado sistema operativo y se suministra con el sistema operativo independientemente de los dispositivos USB o del software cliente.

El controlador de host USB está constituido por el hardware y el software que permite a los dispositivos USB ser conectados al host (descrito en el apartado 1.2.2.1.2).

Como se muestra en la figura 5, la conexión entre un host y un dispositivo requiere la interacción entre las capas. La capa de interfaz de bus USB proporciona la conexión física entre el host y el dispositivo. La capa de dispositivo USB es la que permite que el software del sistema USB realice operaciones genéricas USB con el dispositivo. La capa de función proporciona capacidades adicionales y especificas al host a través de una capa de software cliente apropiada asociada. Las capas de función y dispositivos USB tienen cada una de ellas una visión de la comunicación lógica dentro de su nivel, aunque la comunicación entre ellas se hace realmente por la capa de interfaz de bus USB.

#### 1.2.2.4 Relación Software Cliente-Función

A pesar de que la topología física y lógica del USB refleja la naturaleza compartida del bus, la manipulación de la interfaz de una función USB por parte del software cliente (CSw) se presenta con una vista distinta.

El software cliente para las funciones USB debe usar la interfaz de programación software USB para manipular sus funciones en lugar de manipularlas directamente a través de la memoria o los accesos I/O como pasa con otros buses (PCI, EISA, PCMCIA). Durante esta operación, el software cliente debería ser independiente a otros dispositivos que puedan conectarse al USB.

### 1.2.3 Flujo de Datos USB

El bus USB proporciona un mecanismo de comunicación entre el software del host y su función USB. Las funciones pueden tener requerimientos de flujo de comunicación diferentes para las distintas interacciones cliente-a-función. El bus USB proporciona una mejor utilización del bus en general, permitiendo la separación de los distintos flujos de comunicación de la función USB. Para realizar esto USB emplea dos elementos muy importantes dentro del modelo lógico del bus: los Endpoints (puntos finales) y los Pipes (conductos), los cuales se explican a continuación (ver figura 6).

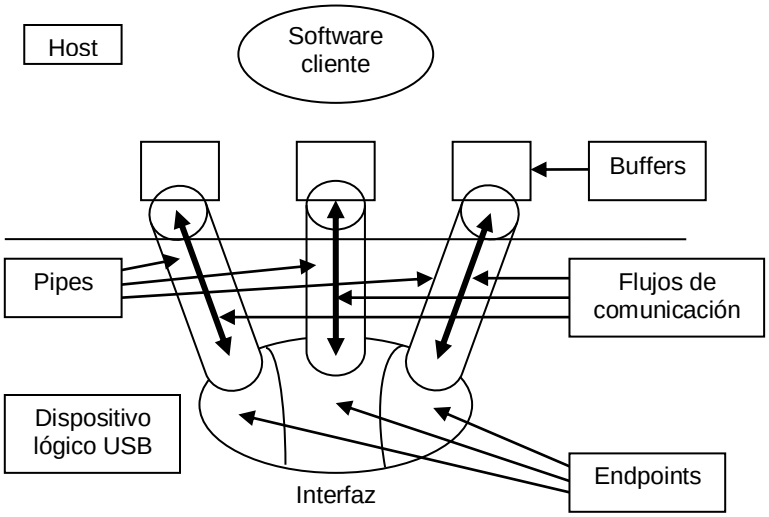



**Figura 1.6 Flujo de Comunicación USB**

### 1.2.3.1 Endpoints

Un endpoint es una porción hardware identificable única de un dispositivo USB que es la terminación de un flujo de comunicación entre el host y el dispositivo. Cada dispositivo USB está compuesto por una colección de endpoints independientes, y una dirección única asignada por el sistema en el momento de la conexión de forma dinámica. A su vez cada endpoint dispone de un identificador único dentro del dispositivo al que pertenece, a este identificador se le conoce como número de endpoint y viene asignado de fábrica. Cada endpoint tiene una determinada orientación de flujo de datos. La combinación de dirección, número de endpoint y orientación, permite referenciar cada endpoint de forma inequívoca. Cada endpoint es por si solo una conexión simple que soporta un flujo de datos en una única dirección, bien de entrada o bien de salida.

Cada endpoint se caracteriza por:

- Frecuencia de acceso al bus requerida
- Ancho de banda requerido
- Número de endpoint
- Tratamiento de errores requerido
- Máximo tamaño de paquete que el endpoint puede enviar o recibir
- Tipo de transferencia para el endpoint
- La orientación en la que se transmiten los datos

Existen dos endpoints especiales que todos los dispositivos deben tener, los denominados endpoints número 0 de entrada y de salida. El Software del Sistema USB usa estos endpoints como método de control predefinido para inicializar, configurar y manipular genéricamente el dispositivo en asocio con el Pipe de Control por Defecto. Estos endpoints están siempre accesibles mientras que el resto no lo estarán hasta que no hayan sido configurados por el host.

### 1.2.3.2 Pipes

Un pipe USB es una asociación entre un endpoint en un dispositivo y el software en el host. Los pipes permiten mover datos entre el software en el host, a través de un buffer, y un endpoint en un dispositivo. Existen dos tipos de pipes: Flujo (Stream) y Mensaje (Message).

Además un pipe se caracteriza por:

- La demanda de acceso al bus y el uso del ancho de banda
- Un tipo de transferencia
- Las características asociadas a los endpoints

El pipe que esta formada por dos endpoints con número cero se denomina pipe de control por defecto. Este pipe está siempre disponible una vez se ha conectado el dispositivo. El resto de pipes aparecen después que se configura el dispositivo. El pipe de control por defecto es utilizado por el software USB del sistema para obtener la identificación y configurar el dispositivo.

El software cliente normalmente realiza peticiones para transmitir datos a un pipe por medio de los Paquetes de Petición I/O (IRPs) y entonces, o bien espera, o bien es notificado de que se ha completado la petición. El software cliente puede causar que un pipe devuelva todas las IRPs pendientes. El cliente es notificado de que una IRP se ha completado cuando todas las transacciones del bus que tiene asociadas se han completado correctamente, o bien porque se han producido errores.

#### 1.2.3.2.1 Pipe de flujo

No necesita que los datos se transmitan con una cierta estructura. Los pipes de flujo son siempre unidireccionales y los datos se transmiten de forma secuencial: "first in, first out". Están pensadas para interactuar con un único cliente, por lo que

no se mantiene ninguna política de sincronización entre múltiples clientes en caso de que así sea. Un pipe de flujo siempre se asocia a un único endpoint en una determinada orientación.

#### 1.2.3.2.2 Pipe de mensaje

A diferencia de lo que ocurre con los de flujo, en los de mensaje la interacción del pipe con el endpoint consta de tres fases. Primero se realiza una petición desde el host al dispositivo, después se transmiten los datos en la dirección apropiada, finalmente un tiempo después se pasa a una fase llamada Status (estado). Para poder llevar a acabo este paradigma es necesario que los datos se transmitan siguiendo una determinada estructura.

Los pipes de mensajes permiten la comunicación en ambos sentidos, de hecho el Pipe de Control por Defecto es un pipe de mensaje. El software USB del sistema se encarga de que peticiones múltiples no se envíen al pipe de mensaje concurrentemente. Un dispositivo da respuesta únicamente a una petición de mensaje en un instante dado por cada pipe de mensaje.

Un pipe de mensajes se asocia a un par de endpoints de orientaciones opuestas con el mismo número de endpoint.

### 1.2.3.3 Frames y microframes

USB establece una unidad de tiempo base equivalente a 1 milisegundo denominada frame (trama) y aplicable a buses de velocidad media o baja, en alta velocidad se trabaja con microframes, que equivalen a 125 microsegundos. Los (micro)frames no son mas que un mecanismo del bus USB para controlar el acceso a este, en función del tipo de transferencia que se realice. En un (micro)frame se pueden realizar diversas transacciones de datos.

### 1.2.3.4 Tipos de transferencias

La interpretación de los datos que se transmitan a través de los pipes, independientemente de que se haga siguiendo o no una estructura USB definida, corre a cargo del dispositivo y del software cliente. No obstante, USB proporciona cuatro tipos de transferencias de datos sobre los pipes para optimizar la utilización del bus en función del tipo de servicio que ofrece la función y los requerimientos del software cliente. Estos cuatro tipos son:

- Transferencias de control
- Transferencias isócronas
- Transferencias de interrupción
- Transferencias en bloques ("Bulk")

#### 1.2.3.4.1 Transferencias de control

Las transferencias de control son usadas por el Software del Sistema USB para configurar los dispositivos cuando estos se conectan por primera vez y para otros propósitos específicos del dispositivo, incluyendo el control de otros pipes en el mismo. La entrega de los datos se hace sin perdidas. Es el único tipo de transferencia que emplea pipes de mensaje.

#### 1.2.3.4.2 Transferencias en bloque (Bulk)

Las transferencias en bloque consisten típicamente de cantidades grandes de datos, tales como los usados por impresoras, escáneres o cámaras. La entrega oportuna de los datos se asegura a nivel de hardware usando detección de errores en el. De igual modo, el ancho de banda utilizado por los datos en bloque puede variar, dependiendo de las otras actividades del bus.

La transferencia en bloque es una comunicación no periódica, típicamente empleada por dispositivos que requieren usar todo el ancho de banda disponible o en su defecto son demoradas hasta que el ancho de banda completo esté

disponible. Esto implica particularmente movimientos de imágenes o video, donde se requiere de gran potencial de transferencia en poco tiempo. Hacen uso de los pipes de flujo.

#### 1.2.3.4.3 Transferencias de interrupción

Es una transferencia en la que los datos se entregan de forma oportuna pero fiable con latencia limitada hacia o desde un dispositivo. Tales datos pueden presentarse por la transferencia de datos de un dispositivo en cualquier momento y son entregados por el bus USB a una proporción no menor que la especificada por el dispositivo.

Un dato de interrupción consiste típicamente de un evento de notificación que no se presenta frecuentemente y esta organizado como uno o más bytes. Un ejemplo de datos de interrupción se presenta en dispositivos como teclados o ratones. Hacen uso de los pipes de flujo.

#### 1.2.3.4.4 Transferencias Isócronas

Los datos Isócronos son continuos y se crean, entregan, y consumen en tiempo real. Los datos isócronos deben entregarse en la misma proporción que se recibieron para mantener la sincronía. Además de la proporción en la entrega, los datos isócronos pueden ser también sensibles a los retardos de entrega.

Un ejemplo típico de datos isócronos es la voz. Si la proporción en la entrega de esos flujos de datos no se mantiene, ocurrirán perdidas en dicho flujo debido a carencias o desbordamientos de datos en el búfer o la trama. Aun si los datos son entregados en una proporción adecuada por el hardware USB.

La entrega oportuna de datos isócronos se asegura a expensas de las pérdidas potenciales en el flujo de datos. En otras palabras, cualquier error en la transmisión eléctrica no es corregido por los mecanismos de corrección de errores a nivel hardware. En la práctica, la tasa de errores de bit del USB se espera que sea lo bastante pequeña como para ser un problema. Los flujos de datos isócronos USB se localizan en una porción dedicada del ancho de banda del bus

USB para asegurar que los datos puedan entregarse en la proporción adecuada. Hacen uso de los pipes de flujo.

Como un resumen general de los conceptos anteriores, se puede decir que: los dispositivos USB pueden tener una o más configuraciones posibles, que definen distintas formas de funcionamiento. Una determinada configuración es un conjunto de Interfaces, donde cada Interfaz especifica qué partes del hardware del dispositivo (endpoints) se comunican con el sistema. Es decir, cada posible configuración de un dispositivo USB es un conjunto de Interfaces y cada Interfaz es un conjunto de endpoints.

### 1.2.4 Capa de Protocolo

A nivel físico, el protocolo USB transporta los datos a través de paquetes, estos paquetes se transmiten dentro de uno o varios frames o microframes para llevar a cabo las transacciones que conforman a su vez un tipo de transferencia en particular, que permite el intercambio de información entre el dispositivo y el host. El controlador USB transmite paquetes que incluyen la dirección del dispositivo destino, y el dispositivo que detecta su dirección en el paquete responde y lleva a cabo la transferencia de datos con el controlador. De esta manera, el Controlador USB maneja la parte más compleja del protocolo, generando los paquetes de transferencias de datos las diferentes velocidades, y controlando la conexión lógica entre el sistema y las funciones internas de cada dispositivo.

La forma en la que las secuencias de bits se transmiten en USB es la siguiente; primero se transmite el bit menos significativo, después el siguiente menos significativo y así hasta llegar al bit más significativo.

En la transmisión se envían y reciben paquetes de datos, cada paquete de datos viene precedido por un campo Sync (sincronización) y acaba con el delimitador EOP (fin de paquete), todo esto se envía codificado además de los bits de relleno insertados (ver sección 1.6 Interfaz Eléctrica). En este punto cuando se habla de

datos se refiere a los paquetes sin el campo Sync ni el delimitador EOP, y sin codificación ni bits de relleno.

El primer campo de todo paquete de datos es el campo PID. El PID indica el tipo de paquete y por lo tanto, el formato del paquete y el tipo de detección de errores aplicado al paquete. En función de su PID podemos agrupar los diferentes tipos de paquetes en cuatro clases: Token, Data, Handshake y Special.

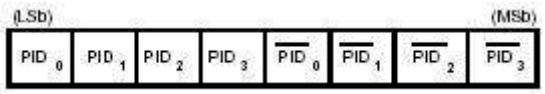
A continuación se explica el formato de los paquetes y sus campos, además de una vista general de como funciona el protocolo.

### 1.2.4.1 Formato de los campos

Cada paquete se divide en campos, a continuación se explica el formato de los distintos campos:

#### 1.2.4.1.1 Campo identificador de paquete (PID)

Es el primer campo que aparece en todo paquete. El PID indica el tipo de paquete, y por tanto el formato del paquete y el tipo de detección de error aplicado a este. Se utilizan cuatro bits para la codificación del PID, sin embargo el campo PID tiene ocho bits, que son los cuatro del PID seguidos del complemento a 1 de esos cuatro bits. Estos últimos cuatro bits sirven de confirmación del PID. Si se recibe un paquete en el que los cuatro últimos bits no son el complemento a 1 del PID, o el PID es desconocido, se considera que el paquete está corrupto y es ignorado por el receptor.

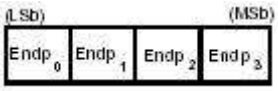



#### 1.2.4.1.2 Campo dirección

Este campo indica el dispositivo, a través de la dirección, que envía o es receptora del paquete de datos. Se utilizan siete bits, de lo cual se deduce que hay un máximo de 128 direcciones.


#### 1.2.4.1.3 Campo endpoint

Se compone de cuatro bits e indica el número de "enpoint" al que se quiere acceder dentro de una función o dispositivo, como es lógico este campo siempre sigue al campo dirección.

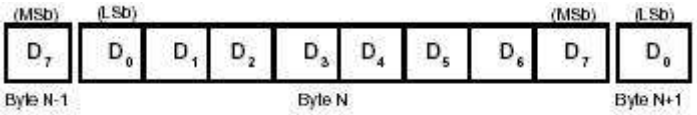



#### 1.2.4.1.4 Campo número de frame

Es un campo de 11 bits que es incrementado por el host cada (micro)frame en una unidad. El máximo valor que puede alcanzar es el 7FFH, si se vuelve a incrementar pasa a cero.

#### 1.2.4.1.5 Campo de datos

Los campos de datos pueden variar de 0 a 1024 bytes. En el dibujo se muestra el formato para múltiples bytes.




#### 1.2.4.1.6 Campo CRC (Cyclic Redundancy Checks)

El CRC se usa para proteger todos los campos no PID de los paquetes de tipo token y de datos. El CRC siempre es el último campo y se genera a partir del resto

de campos del paquete, a excepción del campo PID. El receptor al recibir el paquete comprueba si se ha generado de acuerdo a los campos del paquete, si no es así, se considera que uno o mas datos de un campo están corruptos, en ese caso se ignora el paquete.

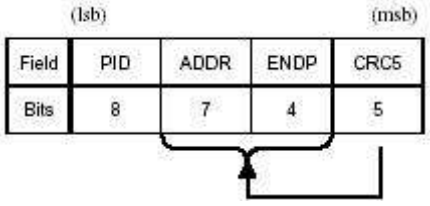
El CRC utilizado detecta todos lo errores de un bit o de dos bits. El campo de CRC es de cinco bits para los paquetes de tipo IN, SETUP, OUT, PING y SPLIT.

### 1.2.4.2 Formato de los paquetes

A continuación se describen los diferentes paquetes y sus formatos en función de los campos que lo conforman.

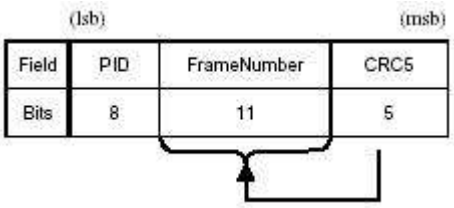
#### 1.2.4.2.1 Paquete tipo token

Un token está compuesto por un PID que indica si es de tipo IN, OUT o SETUP. El paquete especial de tipo PING también tiene la misma estructura que token. Después del campo PID viene seguido de un campo dirección y un campo endpoint, por último hay un campo CRC de 5 bits.



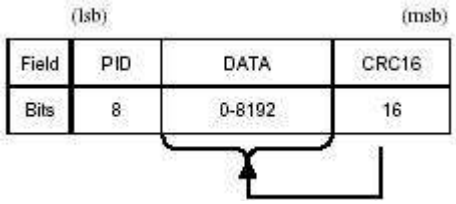

#### 1.2.4.2.2 Paquete inicio de frame (SOF)

Estos paquetes son generados por el host cada un milisegundo en buses de velocidad media y cada 125 microsegundos para velocidad alta. Este paquete está compuesto por un campo número de frame y un campo de CRC de 5 bits.



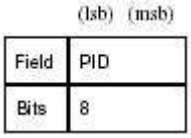
#### 1.2.4.2.3 Paquete de datos

Este paquete está compuesto por cero o más bytes de datos seguido de un campo de CRC de 16 bits. Existen cuatro tipos de paquetes de datos: DATA0, DATA1, DATA2 y MDATA. El número máximo de bytes de datos en velocidad baja es de ocho bytes, en media de 1023 bytes y en alta de 1024 bytes. El número de bytes de datos ha de ser entero.




#### 1.2.4.2.4 Paquetes "Handshake"

Los paquetes "handshake", en castellano apretón de manos, se utilizan para saber el estado de una transferencia de datos, indicar la correcta recepción de datos, aceptar o rechazar comandos, control de flujo, y condiciones de parada. El único campo que contiene un paquete de este tipo es el campo PID.




Existen cuatro paquetes de tipo "handshake" además de uno especial:

- ACK: Indica que el paquete de datos ha sido recibido y decodificado correctamente. ACK sólo es devuelto por el host en las transferencias IN y por una función en las transferencias OUT, SETUP o PING.
- NAK: Indica que una función no puede aceptar datos del host (OUT) o que no puede transmitir datos al host (IN). También puede enviarlo una función durante algunas fases de transferencias IN, OUT o PING. Por último se puede utilizar en el control de flujo indicando disponibilidad. EL host nunca puede enviar este paquete.

- STALL: Puede ser devuelto por una función en transacciones que intervienen paquetes de tipo IN, OUT o PING. Indica que una función es incapaz de transmitir o enviar datos, o que una petición a una tubería control no está soportada. El host no puede enviar bajo ninguna condición paquetes STALL.
- NYET: Sólo disponible en alta velocidad es devuelto como respuesta bajo dos circunstancias. Como parte del protocolo PING, o como respuesta de un hub a una transacción SPLIT indicando que la transacción de velocidad media o baja aún no ha terminado, o que el hub no está aún habilitado para realizar la transacción.
- ERR: De nuevo sólo disponible en alta velocidad y de nuevo formando parte del protocolo PING, permite a un hub de alta velocidad indicar que se ha producido un error en un bus de media o baja velocidad

#### 1.2.4.3 Transacciones

Los paquetes de tipo token tiene como objetivo indicar el inicio de una transacción de datos de una determina forma.

##### 1.2.4.3.1 Transacción IN

La transacción empieza con el envío de un paquete de tipo IN por parte del host a un determinado endpoint en una función. Un endpoint cuando recibe un paquete de tipo IN debe responder al host dependiendo si el paquete fue recibido con éxito o no. De igual modo el host debe responder dependiendo si el paquete de datos esta corrupto o no.

##### 1.2.4.3.2 Transacción OUT

El host envía un paquete OUT a un determinado endpoint y seguidamente envía el paquete de datos. Suponiendo que el endpoint ha decodificado correctamente el token, en caso contrario se ignora el token y lo datos, espera a recibir un paquete

de datos. El comportamiento del endpoint cuando recibe el paquete de datos depende de si el paquete esta corrupto o no.

#### 1.2.4.3.3 Transacción SETUP

La transacción SETUP es una transacción especial que tiene las mismas fases que una transacción OUT, que sólo se puede utilizar con endpoints de tipo control y cuya finalidad es la de indicar el inicio de la fase setup.

### 1.2.4.4 Token especial Split y transacción Split

El token especial SPLIT es utilizado para poder soportar transacciones en varias partes entre un hub que trabaja a velocidad alta con dispositivos de velocidad media o baja. Se definen nuevas transacciones que utilizan el token SPLIT, en concreto dos transacciones con sus respectivos paquetes: "start-split transaction (SSPLIT)" y "complete-split transaction (CSPLIT)".

La transacción split es utilizada solamente por el host y un hub cuando se quiere comunicar con un dispositivo de velocidad media o baja conectado con el hub. El host puede iniciar una transacción por partes a través del paquete SSPLIT y puede completarla, recibiendo las respuestas de la función de velocidad baja o media, a través del paquete CSPLIT. Esto permite al host realizar otras transacciones a velocidad alta sin tener que esperar a que acabe la transacción.

### 1.2.4.5 El protocolo y las transferencias

En este apartado se muestra de forma general las transacciones que se realizan para llevar a cabo cada tipo de transferencia.

De forma general toda función en principio está esperando a que el host le envíe un paquete, si este paquete es de tipo token entonces cambia el estado de la función iniciándose una transacción. También se debe de tener en cuenta, que

cuando o bien el host, o bien una función se encuentran en un estado que no es el de reposo, aparecen los timeouts como otra causa de error.

#### 1.2.4.5.1 Transferencias en bloques ("Bulk")

En las transferencias en bloques se puede hablar en parte de transacciones en bloques, pues la idea es enviar datos de paquete en paquete sin que tenga que haber un flujo de datos continuo. La diferencia reside en que si hablamos de transferencia, no se puede considerar que haya terminado hasta que el host haya conseguido enviar los datos, inclusive después de varios intentos fallidos después de un error.

Cada transacción en bloque se puede dividir en tres fases: token, datos y "handshake", si bien gracias al token PING pueden haber transacciones de dos fases: token y "handshake".

#### 1.2.4.5.2 Transferencias de control

Estas transferencias constan de tres fases: transacción setup, fase de datos y transacción de estado. La transacción siempre la inicia el host, y sirve para enviar información al endpoint para indicar que se quiere realizar una acción de control.

A continuación se inicia la fase de transferencia de datos, que consiste en una secuencia de transacciones de datos siempre en la misma dirección alternando DATA0 y DATA1. Esta fase no es obligatoria, la dirección se establece en la fase setup. Finalmente tiene lugar una transacción estado, esta transacción es idéntica a una transacción en bloques. La dirección de esta transacción es siempre la contraria a la de la fase de transferencia de datos, y si esta no exisitiese, la dirección es del endpoint al host.

#### 1.2.4.5.3 Transferencias de interrupción

Las transferencias de interrupción son solamente transacciones de tipo IN y OUT. Desde el punto de vista de las transacciones es muy similar a una transferencia en bloques.

#### 1.2.4.5.4 Transferencias isócronas

Una transferencia isócrona se plantea como una secuencia de transacciones muy sencillas para enviar o recibir datos. Estas transacciones no utilizan "handshakes" y por lo tanto no se reenvían paquetes, ya que el objetivo de la transferencia es simular un flujo constante de datos.

### 1.2.5 Interfaz Eléctrica

En la capa física del protocolo USB se utiliza un cable de 4 hilos para realizar la transmisión de una señal diferencial (D+ y D-) y de la alimentación del bus (5V y GND). La comunicación se lleva a cabo en una forma bidireccional y semi-dúplex, y se utiliza codificación NRZI (donde la transmisión de un 0 se representa con un cambio de nivel en la línea y la de un 1 con ningún cambio) con “bit stuffing” (relleno de bits), es decir la inserción de un cero tras la transmisión de 6 unos, para asegurar transiciones en la línea y permitir que el receptor mantenga su sincronización.

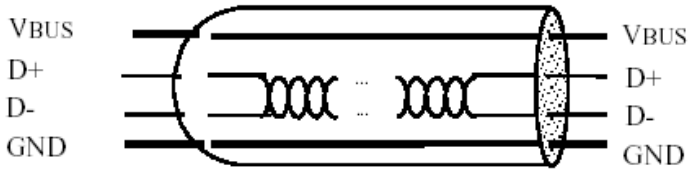



**Figura 1.7 Configuración del Cable USB**

Para la transmisión en modo high-speed, el transmisor activa una fuente de corriente interna, derivada a partir de su fuente de alimentación positiva, y dirige dicha corriente hacia una de las dos líneas de datos por medio de un conmutador de corriente de alta velocidad. De esta manera, el transmisor genera los estados diferenciales DIFF0 y DIF1, también llamados J y K, high-speed en el cable. El estado J se obtiene dirigiendo la corriente sobre la línea D+, mientras que el estado K se obtiene dirigiendo la corriente sobre la línea D-.

En modo high-speed, tanto el transmisor como el receptor activan unas resistencias de terminación entre cada línea y masa (45Ω +/- 10 %), de forma que el valor nominal de la corriente (17,78mA) produce un voltaje nominal en la línea de 400mV. El voltaje diferencial nominal (D+ - D-) es, por lo tanto, de +400mV para el estado J y de – 400mV para el estado K.

Por medio de estas líneas diferenciales USB puede transmitir y detectar diferentes estados eléctricos sobre el bus, que sirven para monitorización y control, entre ellos están:

- Sync: Teniendo en cuenta que K y J representan respectivamente nivel bajo y nivel alto, el patrón de señal Sync emitido, con los datos codificados, es de 3 pares KJ seguidos de 2 K para el caso de velocidad media y baja. Para velocidad alta es una secuencia de 15 pares KJ seguidos de 2 K. El patrón de señal Sync siempre precede al envío de cualquier paquete, teniendo como objetivo que el emisor y el receptor se sincronicen y se preparen para emitir y recibir datos respectivamente. Si partimos de que el estado de reposo de la señal es J, podemos interpretar Sync como una secuencia impar de "0's" y un "1" que se inserta antes de los datos.

- SE0 (Single-Ended 0): Las señales D+ y D- a 0V. Utilizado para la detección de conexión/desconexión de dispositivos, para indicar el EOP (fin de paquete) y para generar reset.

- IDLE: reposo o línea en alta impedancia, necesario para permitir transferencias semi-dúplex, detectar la conexión y desconexión de dispositivos y discriminar entre dispositivos FS y LS.

- SOP (principio de paquete): se indica mediante una transición IDLE a K.

- EOP (fin de paquete): A todo paquete le sigue EOP, cuya finalidad es indicar el final del paquete. En el caso de velocidad media y baja el EOP consiste en que, después del último bit de datos en el cual la señal estará o bien en estado J, o bien en estado K, se pasa al estado SE0 durante el periodo que corresponde con el ocupado por dos bits, finalmente se transita al estado J que se mantiene durante 1 bit. Esta última transición indica el final del paquete. En el caso de la velocidad alta se utilizan bits de relleno erróneos, que no están en el lugar correcto, para indicar el EOP. Concretamente, el EOP sin aplicar codificación consistiría en añadir al final de los datos la secuencia 0111 1111.

- Detección de dispositivo y discriminación FS/LS: cuando el transmisor deja la línea en IDLE, si hay un dispositivo conectado su polarización fuerza un estado J (DIFF0 si LS ó DIFF1 si FS), y si no lo hay, la polarización del transmisor fuerza un estado SE0.

- Reset: Transmisión de SE0 durante >= 10 ms.

#### 1.2.5.1 Identificación de la velocidad

Los dispositivos USB 1.x identifican su velocidad eléctricamente, mediante una resistencia de pull-up en la línea D+ (full-speed) o D- (low-speed). En cambio los dispositivos high-speed se identifican en principio como full-speed (mediante un pull-up en la línea D+), y durante el proceso de reset ejecutan un protocolo de bajo nivel (determinadas secuencias de señales eléctricas de niveles especiales) a través del cuál determinan si están conectados a un puerto fullspeed o high-speed.

Si el puerto es high-speed, detectará y responderá al protocolo iniciado por el dispositivo. Sólo si el concentrador y el dispositivo ejecutan el protocolo, se establece una comunicación high-speed entre ellos.

## 1.3 SISTEMAS DE ADQUISICIÓN DE DATOS

Un sistema de adquisición de datos, como su nombre lo indica, describe los productos y/o procesos utilizados para adquirir información, ya sea para caracterizar o para analizar algún fenómeno. En su forma más simple, el registro de temperaturas en un horno sobre un papel, es adquirir datos. Como la tecnología ha progresado, este tipo de procesos se ha simplificado y se ha hecho más exacto, versátil y confiable a través de equipamiento electrónico. Los equipos van desde registradores simples hasta sofisticados sistemas de computación. Los productos de adquisición de datos sirven como punto focal en un sistema, relacionando una gran cantidad de productos tales como, sensores, transductores que indican temperatura, flujo, nivel o presión.

### 1.3.1 Buses y Compatibilidad

Los computadores más populares en aplicaciones de adquisición de datos son los IBM PC y los compatibles. El bus del IBM PC/AT se conoce como bus ISA (Industry Standard Arquitecture). El bus PC/XT o S -100 ejecuta transferencias de a 8 bits y es capaz de trasferencias en tiempo real de hasta 100 kilo palabras (16 bits) por segundo. El bus ISA ejecuta trasferencias de 16 bits y puede lograr hasta 300 kilo palabras por segundo. Usando memoria FIFO y lógica especial la ISA puede transferir hasta 1.5 millones de palabras por segundo.

Debido a su ancho de banda de 16 bits, el bus ISA fue el rey indiscutido mientras los procesadores trabajaban solo con palabras de 16 bits, y solo corrieron hasta 8 MHz, su velocidad máxima, en otras palabras, hasta el 80286. Sin embargo, con la introducción del 80386 y 80486, cuyo bus de datos es de 32 bits de ancho y cuyas velocidades duplicaron los 8 MHz, se empezaron a alzar voces cuestionando, la eficiencia de este bus para dichos procesadores y los que siguieran. A pesar de esto en la actualidad, cuenta con la mayoría de la base instalada y prácticamente

la totalidad de los fabricantes continúan produciendo tarjetas y computadores que incluyen esta arquitectura de bus.

También hay disponibles en el mercado desde 1987, productos de adquisición para computadores IBM PS/2 y compatibles, que utilizan el bus microcanal o MCA, (Micro Channel Arquitecture) . Este bus, fue el primero de 32 bits destinado a PC's que existió. Sus características principales, además de ser más rápido, son que posee un protocolo especial para el control del bus, liberando de esta manera a la CPU de la obligación de controlarlo, como es en el bus AT.

Esta cualidad unida a mejoras de diseño físico del transporte de la información, capacidad para compartir las interrupciones y la capacidad para elegir entre 24 y 32 bits de ancho permiten al MCA alcanzar velocidades de transferencia de hasta 20Mbps. Sin embargo, en Septiembre 1989 IBM anunció tres modos adicionales de transferencia: el primero eleva la velocidad a 40 Mbps, el segundo a 80 Mbps y el tercero reservado para aplicaciones futuras, lo eleva hasta la increíble taza de 160 Mbps. Estos nodos son también conocidos como Micro Canal II. Sin embargo y a pesar de las increíbles mejoras en el rendimiento del bus MCA, este acarrea un problema grave y es que la arquitectura es completamente incompatible con la ISA, lo que deja en el camino a toda la tecnología ya existente para tarjetas de adquisición, a además está entretejida con tecnología exclusiva de IBM, lo que eleva los costos de fabricación por concepto de pago de licencias.

Un año después del lanzamiento del bus MCA, en Septiembre de 1988, nueve empresas: Compaq, AST, Epson, Hewlett Packard, NEC, Olivetti, Tandy, Wyse y Zenith, lanzaron al mercado la contrapartida de MCA, el bus E ISA (Extended Industry S a Arquitecture). Este bus fue diseñado con un claro propósito: ofrecer la potencia del MCA, sin sus desventajas. Las características de EISA duplican esencialmente las de MCA, un programa de instalación, arbitraje del bus, interrupciones compartidas y duplicación del ancho de banda sin ninguna de sus desventajas. Debido a un diseño especial del conector y de los contactos en las

tarjetas EISA un slot EISA acepta y hace funcionar, sin ningún inconveniente una tarjeta ISA, de este modo al comprar un PC con esta arquitectura, la tarjetas de adquisición con que se cuente, seguirán siendo útiles. Otras ventajas de este bus son que un mayor número de fabricantes lo apoyan y que no es necesario pagar patentes. Desde el punto de vista de la velocidad, EISA puede llegar hasta una velocidad de transferencia de 33 Mbps.

Por último, el bus STD desarrollado por Pro-Log y Mostek Corporation esta orientado hacia aplicaciones industriales y de control de procesos. Es el sistema de bus que más se verá en aplicaciones industriales. Cada tablero tiene solo unas pocas funciones, de manera que se puede diseñar con facilidad el sistema efectivo en cuanto a costo. Alrededor de 60 fabricantes hacen productos compatibles con los sistemas de bus STD, incluyendo tarjetas de salida para relés, tarjetas I/0 digitales y análogas, conversores A/D y D/A, impulsores de triacs e impulsores de motores paso a paso.

### 1.3.2 Tipos de Sistemas de Adquisición de Datos

Un sistema de adquisición de datos o sistema interfase computador permite introducir datos desde el mundo real al computador. Captura las señales producidas por sensores de temperatura, presión, flujo, etc.., y las convierte a una forma tal que el computador las puede comprender. Con un sistema de adquisición se puede usar el computador para capturar, monitorear, mostrar y analizar los datos. Si el sistema de adquisición tiene posibilidades de salida, el computador también se puede usar para controlar cuidadosamente el proceso, con el fin de obtener una máxima eficiencia. Los computadores personales tienen la mayor participación en la adquisición de los datos. Dependiendo del nivel de conocimiento del usuario, y los productos, el computador puede jugar varios roles.

El software también permite usar el computador para controlar procesos, además de almacenar los datos. Si la interfase tiene posibilidades de salida, mediante software se puede controlar el proceso, por ejemplo, mantener la temperatura constante, aumentar el flujo cada media hora, etc. El software de adquisición de los datos, trabaja con el hardware de la interfase para sacar el máximo provecho del recurso computacional. Además, para controlar el almacenamiento de los datos que hace la tarjeta o interfase, el software se diseña para que almacene los datos en Tablas como una planilla de datos, o en histogramas, gráficos de torta, gráficos de línea, y analizar los datos usando transformadas y aproximación polinómica. En resumen, es una excelente manera de presentar las inmensas prestaciones de un sistema de adquisición, en forma simple y amena.

Las dos maneras de clasificar los sistemas que relacionan al computador con la adquisición de datos, es hablar de sistemas de tarjeta incorporada (plug-in) y de sistemas externos o comunicados externos (stand-alone), Figura 1.8. Los primeros se refieren a las tarjetas que se incorporan directamente en una ranura de expansión del computador (slot), mientras que los segundos se refieren a aquellos que son remotos, permanecen alejados del computador y se comunican con él mediante un puerto de comunicaciones. En ambos casos, se requiere de software específico necesario para instruir al computador de como manejar los datos.

#### 1.3.2.1 Sistemas de Tarjeta Incorporada

Un sistema de tarjeta incorporada es exactamente eso, una tarjeta incorporada dentro de un slot libre del PC. Debido a que se introduce dentro del computador, éstas se diseñan para un tipo especifico de computador, tal como IBM PC o compatibles, o de la serie Apple II. Las tarjetas incorporadas tienen un terminal externo, es decir, que sale del computador, unido a ella donde se hacen las conexiones a los termopares, a los otros sensores y a las salidas. Estas tarjetas son sistemas completos, pues traen un menú de demostración, que permite

seleccionar cada tipo de entrada, rango, escalamiento en las unidades adecuadas, capacidades de entrada y salida digital, registradores remotos (data loggers) y control. Se pueden expandir rápida y fácilmente, basta poner otra tarjeta. Dependiendo del sistema, se pueden tener hasta 240 entradas análogas por IBM PC.

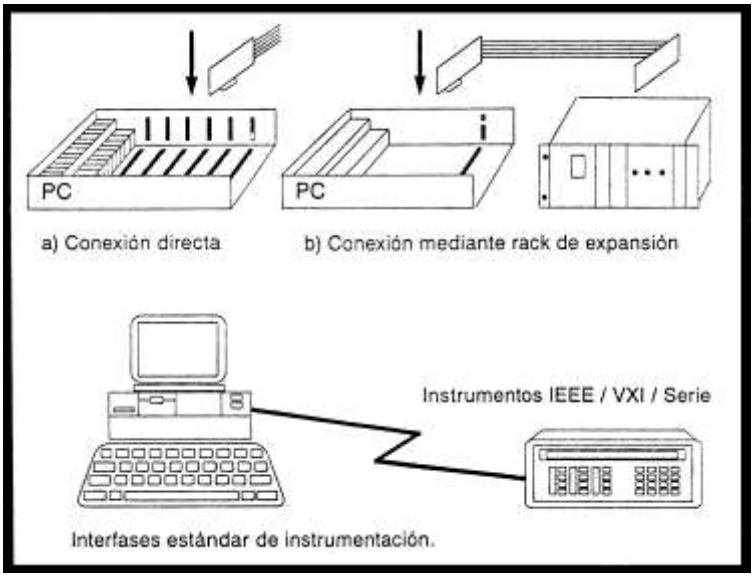



**Figura 1.8 Diferentes Sistemas de Adquisición de Datos**

Los sistemas de tarjeta incorporada se conectan directamente al bus del computador. Sus ventajas respecto del otro sistema son la velocidad y el costo (los gastos de empaquetamiento y alimentación eléctrica los proporciona el computador). Las tarjetas que se ofrecen en el mercado son principalmente para computadores tipo IBM PC o compatibles y los del tipo Apple Macintosh. El comportamiento de las tarjetas puede variar de acuerdo a: número y tipos de entradas (tensiones, termopares, on/off), salidas, velocidad y otras funciones requeridas. Cada tarjeta instalada en el computador está direccionada en una ubicación única del mapeo de memoria de entrada/salida. El mapeo de memoria en el computador es la dirección de memoria que el procesador utiliza, para tener acceso a un dispositivo especifico cuando es requerido por su programa.

Las tarjetas de adquisición de datos se encuentran disponibles con hasta cuatro tipos diferentes de funciones. La primera, es la entrada análoga, donde la variable

física del mundo real es introducida al computador. Esta función es el esqueleto de cualquier sistema. Muchas tarjetas también tienen salidas análogas, gracias a las cuales se puede contar con señales para controlar una válvula o un actuador. Las otras dos funciones son señales de entrada y salida digitales. Tal como su nombre lo indica, las entradas digitales le dicen al computador si algo está abierto o cerrado, tal como la alimentación de 220 Voltios a un calefactor. Una salida digital, por otra parte, recibe una señal del PC, y la utiliza para controlar algún interruptor, por ejemplo, conectar la potencia al calefactor. Muchas tarjetas de propósito general tienen una combinación de dos o más de estas funciones, sin embargo, también existen tarjetas específicas para ciertas funciones, como por ejemplo, pulsos de duración programable (timer programable).

Existen varias razones para seleccionar un sistema de adquisición del tipo de tarjeta incorporada. La primera es que son muy económicas, fluctuando entre 300 y 2000 dólares para 16 señales análogas de entrada. Esto significa aproximadamente un costo máximo de 2000/16 o 125 dólares por canal de entrada. Muchos sistemas comunicados tienen precios del orden de 350 dólares y más por canal. Otra ventaja de estos sistemas es que cada canal puede ser construido individualmente para diferentes tipos de entrada. Por ejemplo, se puede tener el canal 1 para termopar, el 2 para un voltaje, el 3 para corriente, etc.. Con sistemas comunicados no existe elección para las entradas, se deben tener 4 u ocho canales dedicados a ese tipo de entrada.

La tarjeta incorporada se diseñó para ser fácil de usar. Para usarla basta conectarla a un slot libre del PC. Cuando se enciende el computador, ya se está en condiciones de comenzar la adquisición de los datos. Muchas tarjetas vienen con su disco de software demostrativo y algunas con programas de aplicación estándar. Esto hace las cosas mucho más fáciles para el usuario quien no desea verse involucrado en la programación de la tarjeta. También existen paquetes de software disponibles que incrementan la capacidad de la tarjeta, más allá del programa estándar que viene con la tarjeta.

Por último, muchas tarjetas soportan Acceso Directo a la memoria (DMA) del PC. Este hecho, permite a la tarjeta almacenar los datos directamente en la memoria. Con el DMA se pueden velocidades de muestreo mucho mayores (más rápido), que con el sistema externo.

Hay una gran variedad de tarjetas para muchas aplicaciones diferentes. Los tres tipos principales de tarjetas incorporadas son: de adquisición de datos, tarjetas de comunicaciones y aquellas diseñadas para aplicaciones especiales.

Las tarjetas de comunicaciones permiten al PC conversar con otros dispositivos, otros PC's, impresoras, sistemas de adquisición comunicados .Utilizando un formato estándar como, la norma RS232C, el computador puede comandar otro dispositivo. Muchas tarjetas de comunicaciones habituales en el mercado, usan la norma RS232C. Sin embargo, para otras aplicaciones, han aparecido otros estándares tales como RS422 para distancias mayores, RS485 para comunicaciones entre más de dos equipos y la IEEE488 para las comunicaciones entre equipos científicos en un laboratorio.

El último tipo de tarjeta es el utilizado para aplicaciones especificas. En lugar de tener una variedad de funciones, tales como las que tiene una tarjeta de adquisición típica, esta tarjeta está diseñada para un tipo específico de entrada, tal como cromatografía, es decir es como un instrumento en forma de tarjeta. Junto con el software creado específicamente para esa función, esta tarjeta hace el mismo trabajo que un sistema completo, el que habría costado mucho más caro unos pocos años atrás.

#### 1.3.2.2 Sistemas Externos

En los sistemas externos, todo el sistema que realiza la adquisición, está montado en un chasis externo y se conecta al computador a través de un enlace de

comunicaciones estándar, por ejemplo, RS232, RS422, RS485 y IEEE488. Los sistemas comunicados, no dependen del tipo de computador, todo lo que el computador necesita es un puerto de comunicaciones serie o paralelo estándar. Esta interfase puede conversar con el computador a través de ese puerto, y existen en el mercado muchos niveles diferentes, desde unidades de canal único diseñadas para una entrada de propósito general, hasta sistemas modulares que pueden aceptar directamente sensores tales como termopares, termistores, de nivel, etc.

Un sistema de adquisición externo ofrece muchas ventajas cuando se le compara con los sistemas de tarjeta incorporada en el PC. Algunas de ellas son el tamaño y la potencialidad de ser expandidas según como requiera el usuario. Algunas de ellas aceptan solo dos entradas, mientras que otras pueden expandirse hasta 120 0 más entradas analógicas. Una gran ventaja de los sistemas comunicados, es que se puede comenzar de a poco y enriquecer el sistema a medida que los requerimientos crezcan. Trabajando en un sistema como éste, y de esta manera, se abaratan los costos iniciales de la inversión e incluso permite agregar más canales cada vez que se requiera en el futuro. Por último, cuando se construye y expande el sistema, el costo por canal base disminuye. Ya que cada sistema solo necesita un módulo CPU alimentado, no se necesitan unidades adicionales para la expansión el único costo en el que se incurre es por los canales reales que se necesitan.

Al contrario de los sistemas de tarjeta incorporada, los sistemas externos, se pueden colocar en casi cualquier sitio. Para instalar un sistema de tarjeta incorporada para adquirir datos en un experimento, se tiene que dedicar un PC en las cercanías del mismo; en efecto, dependiendo de la ubicación exacta se podrían incurrir en gastos tales como tener que comprar un PC “industrializado” o dedicado a ese experimento, que satisfaga incluso el ambiente que rodea al laboratorio. Sin embargo, con un sistema externo se puede colocar el sistema en casi cualquier sitio, incluso en un “rack” industrial. Estos últimos se recomiendan

para proteger el sistema. Además para interactuar con el computador anfitrión, no hay que preocuparse del tipo de cable para la comunicación, puesto que muchos sistemas comunicados pueden comunicarse con el anfitrión usando incluso la línea telefónica, a través de un modem.

Tal vez una de los mayores motivos para seleccionar un sistema externo, es el tipo de computador que el usuario debe tener. Las tarjetas incorporadas se diseñan para trabajar solamente con un tipo de computador, ya sea un IBM PC o compatible o un APPLE Macintosh. Si no se tiene uno de estos computadores, la tarjeta no trabajará. Como contrapartida, un sistema comunicado puede interactuar con cualquier computador que tenga un puerto de comunicaciones estándar: RS232C, RS422, RS485 o IEEE488.

Si se deseara mover el sistema de adquisición de un sitio a otro, es más fácil hacerlo en el sistema externo. Con una tarjeta incorporada es mucho más difícil, debido a que todos los cables terminan en el computador. Si el otro Computador está en un sitio distinto, se debe abrir el primero, desconectar los alambres, sacar la tarjeta y cerrarlo, volver a hacer las conexiones y por último hacer todo de nuevo para reinstalarlo en el otro computador.

El software y la facilidad de uso son también factores importantes a la hora de seleccionar el sistema de adquisición. Los sistemas externos, utilizan comandos simples, que son fáciles de aprender y memorizar. Han sido creados especialmente para trabajar con el sistema, con lo cual entregan una ayuda para trabajar el sistema al máximo de su capacidad.

Por último, si el sistema que se está pensando instalar ahora, es el primero de una serie de muchos otros, un sistema externo puede ser duplicado fácilmente para ser instalado en un segundo sitio, al igual que el software de control.

#### 1.3.2.3 Diferencias entre Ambos Sistemas

Los sistemas de tarjeta incorporada ofrecen algunas ventajas sobre su contraparte, los sistemas comunicados. La primera, es que al conectarse directamente al bus del PC, aprovechan las fuentes de alimentación propias de él, lo que las hace más baratas comparadas con el precio de la caja y de la fuente de alimentación que se requeriría para los sistemas comunicados. Respecto a los mismo, el espacio que ocupan uno y otro, las incorporadas no ocupan espacio exterior, puesto que aprovechan la propia caja de la CPU del PC.

La velocidad de transferencia de los datos es más rápida en los sistemas de tarjeta incorporada, puesto que los datos se transfieren directamente del bus a la memoria del PC, a la velocidad que permite el bus. Aunque los sistemas comunicados pueden adquirir datos en su memoria local a velocidades comparables la transferencia real de los datos al computador anfitrión está casi siempre limitada por el enlace de comunicaciones.

Sin embargo los sistemas comunicados también tienen sus ventajas. Al no estar ensamblados a un bus de computador específico, son independientes de él, pudiendo ser usados en computadores de “arquitectura cerrada”, que no posean slots libres. Además tienen la ventaja de que sus fuentes de alimentación y cajas que contienen los circuitos, han sido construidas especialmente para aplicaciones de adquisición de datos. Aunque los sistemas incorporados son perfectamente adecuados en medidas de hasta 16 bits de exactitud, mediadas más exactas o extremadamente rápidas pueden requerir de sistemas externos.

Antiguamente las diferencias entre ellos eran muy grandes. Las incorporadas eran más para iniciados, con fácil instalación y operación. Las comunicadas, eran para gente que estaba familiarizada con los computadores y la programación, ya que no se entregaba software con el producto. Los usuarios de estas últimas, estaban familiarizados con dispositivos electrónicos de medida, y no tenían miedo de

cambiar módulos, soldar componentes, etc. Sin embargo, debido a los avances de la tecnología, las comunicadas ofrecen ahora la misma flexibilidad que las incorporadas, manteniendo sus bondades de mayor exactitud y flexibilidad, y la capacidad de trabajar con más sensores y tipos de señales de entrada.

# 2 DESCRIPCIÓN DEL FPGA EMPLEADO

## 2.1 INTRODUCCIÓN

Uno de los objetivos principales al diseñar la tarjeta de adquisición de datos es permitir que en un futuro ella se pueda adaptar a los requerimientos que exigen los nuevos ambientes en el campo industrial y de control, haciendo que el cambio o adaptación a nuevos módulos se haga de una manera rápida, sencilla y que no implique mayores costos de readaptación o reelaboración de los módulos existentes. En ello juega un papel importante el FPGA, pues es el elemento que brindara estas características a la tarjeta desarrollada y por ende la selección de la misma es de gran importancia.

En lo sucesivo de este capitulo se dará una visión global de los FPGAs y se describirá la tarjeta de desarrollo con el FPGA que mas se adapta a las condiciones de diseño.

## 2.2 VISIÓN GLOBAL DE LOS FPGAs

En el momento en que se lleva a cabo la implementación de un sistema electrónico digital, todo diseñador tiene a su disposición un conjunto amplio de tecnologías. Una de las más populares actualmente es la de dispositivos de lógica programable (PALs, PLDs, FPGAs, etc). Los dispositivos de lógica programable más versátiles son las FPGA (Field Programmable Gate Array). Internamente, una FPGA esta compuesta por un conjunto de bloques iguales (CLBs) dispuesto de forma regular. Cada bloque contiene pequeñas memorias RAM y flips-flops que se

pueden configurar para realizar todo tipo de circuitos combinacionales y secuenciales de pequeña escala. Los bloques se pueden interconectar entre sí mediante conexiones también configurables. La configuración de la FPGA se realiza mediante una comunicación serie denominada bitstream, que puede estar almacenado en una memoria externa (PROM, EEPROM, RAM...) o provenir de cualquier otro sistema tal como un PC, un microcontrolador, otra FPGA, etc.

La tecnología FPGA permite realizar diseños a medida, de bajo coste de desarrollo, incluso para la producción de pocas unidades. Estas características la hacen muy interesante para realizar prototipos de manera rápida. En unas pocas semanas se puede tener un prototipo funcionando e interactuando con motores, sensores, o cualquier componente electrónico externo que necesite ser manipulado. Con los FPGAs se pueden construir periféricos complejos como medidores de distancia por ultrasonido, conversores serie paralelo, unidades de PWM, bloques de transmisión de datos, etc.

## 2.3 ESTRUCTURA DE UN FPGA

Las FPGAs tienen tres tipos de componentes en su interior, que brindan toda su funcionalidad:

- CLB's
- IO's
- Red de interconexión

La Figura 2.1 muestra los componentes listados anteriormente y su ubicación en la estructura de una FPGA.

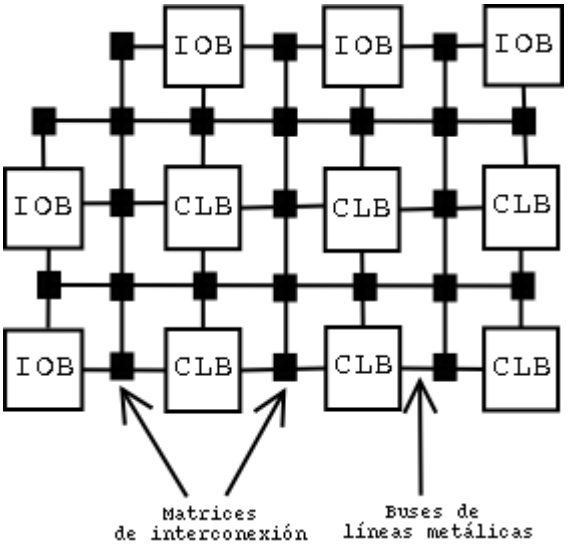



**Figura 2.1 Estructura de un FPGA**

Los CLBs (Bloques lógicos configurables) son las “estructuras” básicas. Se encuentran distribuidos uniformemente por toda el área de la FPGA siendo todos idénticos. Dependiendo del tipo de FPGA, estos bloques lógicos tienen unos componentes u otros, pero básicamente tienen latches, multiplexores y pequeñas memorias para implementar funciones combinacionales (lookup tables). Estos bloques son configurables, estableciéndose qué elementos se conectan con cuales y qué funciones combinacionales realiza, si es que realiza alguna.

Los IOBs (bloques de entrada/salida) son una especie de CLB especializado, que se encuentran junto a los pines del chip y tienen la función de interconectar la lógica interna con el exterior.

La red de interconexión es un conjunto de caminos formados por líneas metálicas y matrices de interconexión que permiten la conexión entre CLBs y CLBs con los IOBs.

En la figura anterior se identifican los rectángulos negros que representan las matrices de interconexión, a las que llegan buses de líneas metálicas. En el

interior de estas “cajas” se configuran las conexiones de las diferentes líneas. Los bloques exteriores son los IOBs y lo interiores los CLBs.

Todas las funciones lógicas, multiplexores, decodificadores, biestables, etc., se implementan utilizando los componentes internos de los CLBs. Cuanto mayor sea el circuito a diseñar, mayor cantidad de CLBs se necesitan. A continuación es necesario establecer las conexiones entre todos los CLBs con el exterior, a través de los IOBs. Es el software el que realiza esto, convirtiendo el diseño en un fichero de configuración para el FPGA (Bitstream).

## 2.4 FABRICANTES DE FPGAS

El uso de los FPGAs ha proliferado en muchos campos de la ciencia desde su aparición en 1985 cuando Xilinx presentó el XC2064, dispositivo que contenía aproximadamente 1000 compuertas lógicas, densidad que se ha incrementado de manera considerable los últimos años.

Xilinx es una de las empresas que más experiencia tiene en el desarrollo de FPGAs, dentro de sus principales productos Hardware se encuentran las familias Virtex, Spartan y Coolrunner. En sus desarrollos software se tienen las plataformas ISE, WebPACK, WebFITTER, Embedded Development Kit.

Altera es una de las compañías más importantes en el campo de los FPGAs y de dispositivos lógicos programables, fue pionera en la fabricación de plataformas de desarrollo para “sistemas en un dispositivo programable”. Ha introducido en el mercado las siguientes familias hardware: StratixF, APEXF, Mercury, Cyclone, ACEX, FLEX, MAX, Excalibur

Actel es también un fabricante conocido en el desarrollo de FPGAs, sus productos tienen diferentes tecnologías (SRAM, antifuse y flash). Entre los desarrollos hardware que brinda se encuentran las Axcelerator, ProASIC, EX, SX, MX. En la

parte software incursiona con las plataformas de desarrollo Libero Integrated Design Environment, Actel Designer, Silicon Explorer.

## 2.5 FABRICANTE Y FAMILIA SELECCIONADA

Realizando una exploración profunda de los diferentes fabricantes de FPGAs existentes en el mercado y de los productos que ofrecen a la fecha de realización de este proyecto se optó por la adquisición de una Tarjeta que contiene un FPGA Spartan-3 del fabricante Xilinx.

La tarjeta base para el desarrollo es proporcionado por la Empresa Opal Kelly de Estados Unidos y de aquí en adelante será referenciado como módulo XEM3001.

### 2.5.1 Familia de FPGAs Spartan-3

La familia Spartan-3 esta especialmente diseña para satisfacer las necesidades de alto volumen, de aplicaciones electrónicas sensibles a costos por el consumidor. Los ocho miembros de la familia ofrecen densidades que van de 50.000 a 5’000.000 de sistemas de compuertas.

La familia Spartan-3 se construye gracias al éxito de la familia Spartan-IIE y se realizo aumentando la cantidad de recursos lógicos, la capacidad de RAM interna, el número total de I/Os, y el nivel global de actuación. Las numerosas mejoras derivan de la innovadora tecnología Virtex-II. Éstas mejoras Spartan-3, combinadas con tecnología del proceso avanzada, entregan mayor funcionalidad y ancho de banda por dólar de lo que era posible previamente, fijando nuevas pautas en la industria de lógica programable.

Debido a su excepcional bajo costo, los FPGAs Spartan-3 satisfacen una gama amplia de aplicaciones electrónicas de consumidor, incluyendo acceso de banda ancha, redes de computadoras caseras y equipos de televisión digital.

La familia Spartan-3 es una alternativa superior para enmascarar ASICs programados. Los FPGAs evitan el alto costo inicial, los ciclos de desarrollo largos, y la inflexibilidad convencional inherente de los ASICs. También, la fácil programación del FPGA permite las actualizaciones de diseño en el campo sin la necesidad de reemplazos hardware, algo imposible de realizar con los ASICs.

### 2.5.2 Visión Global de la Arquitectura

La arquitectura de la familia Spartan-3 consiste en cinco elementos funcionales programables:

Bloques Lógicos Configurables (CLBs) que contienen Look-Up Tables basadas en RAM (LUTs) para implementar la lógica y elementos de almacenamiento que pueden usarse como flip-flops o latches. Los CLBs pueden programarse para realizar una amplia variedad de funciones lógicas así como para guardar datos.

Bloques Input/Output (IOBs) controlan el flujo de datos entre los pines I/O y la lógica interna del dispositivo. Cada IOB soporta flujo de datos bidireccionales además del funcionamiento tri-estado. Se incluyen veinticuatro señales diferentes estándar, incluyendo siete diferenciales de alto rendimiento. Se incluyen registros de Doble Tasa de Datos (DDR). La característica de Impedancia Digitalmente Controlada (DCI) proporciona terminales automáticas, simplificando los diseños de boards.

Los módulos de RAM proporcionan almacenamiento de datos en forma de bloques de 18-Kbit.

Los bloques de multiplicación aceptan dos números binarios de 18 bits como entradas y calculan el producto.

Estos elementos se organizan tal como se muestra en la Figura 1. UN anillo de IOBs rodea un conjunto de CLBs. Los dispositivos que van desde el XC3S200 hasta el XC3S2000 tienen dos columnas de bloques RAM. Los XC3S4000 y dispositivos de XC3S5000 tienen cuatro columnas de RAM. Cada columna esta hecha de varios bloques de RAM de 18-Kbits; cada bloque esta asociado con un multiplicador especializado. Los DCMs se colocan al final de las columnas de bloques RAM externos.

La familia Spartan-3 presenta una red con gran cantidad de pistas e interruptores que interconectan todos los cinco elementos funcionales, transmitiendo señales entre ellos. Cada elemento funcional tiene una matriz de interruptores asociada que permite múltiples conexiones.

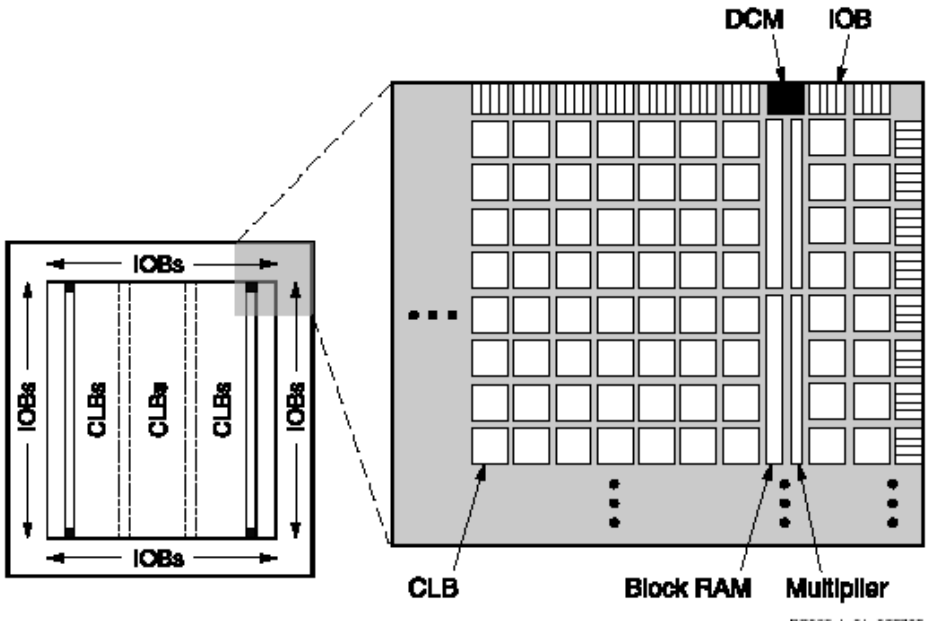



**Figura 2.2 Arquitectura de la Familia Spartan-3**

### 2.5.3 Descripción del Módulo XEM3001.

El modulo XEM3001 es una pequeña tarjeta de desarrollo cuyo componente principal es un FPGA Spartan-3 y que además provee una gran facilidad para el desarrollo de aplicaciones relacionadas con USB gracias a la integración de la capa física de dicho protocolo dentro de un bloque muy funcional.

#### 2.5.3.1 Diagrama de Bloques

El modulo posee tres puertos de acceso denominados JP1,JP2 y JP3. El manejo USB se realiza a través de un microcontrolador (CY68013). Las frecuencias de reloj para los distintos procesos del FPGA se generan gracias a la presencia de un PLL (CY22150). El corazón del modulo es el FPGA de la familia Spartan-3, un XC3S400 con empaquetado 4PQ208.

En la figura 2.3 se muestra el diagrama de bloques del modulo.

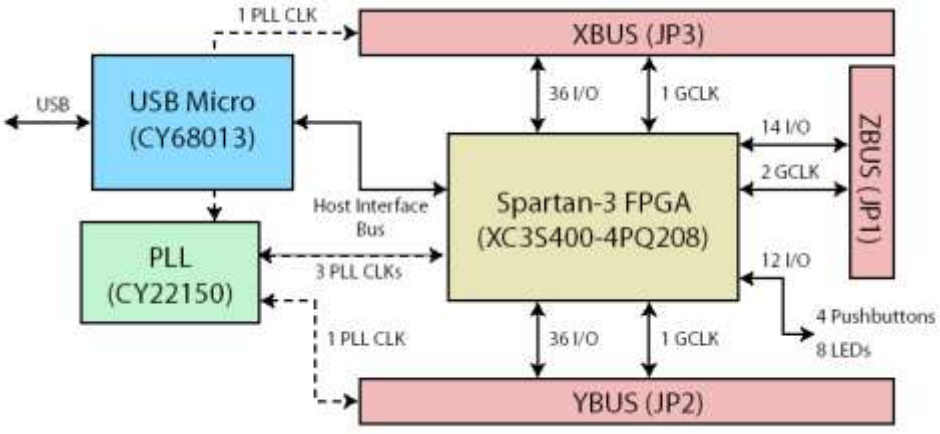



**Figura 2.3 Diagrama de Bloques del Modulo XEM3001**

#### 2.5.3.2 Alimentación del Modulo

El modulo XEM3001 se configura por defecto como autoalimentado, es decir que toma los 5 voltios de alimentación USB para generar los voltajes que necesita (3.3 V, 2.5 V y 1.8 V). Se puede aplicar una fuente de voltaje externo en cualquiera de los pines de 3.3 V en JP1, JP2 o JP3 siempre y cuando se retire el jumper J1, en cuyo caso no se toma la alimentación desde el USB.

Bajo condiciones normales el FPGA no configurado drena aproximadamente 125 mA desde un nodo de 3.3 V. Si se necesita proveer una corriente superior a 500 mA, el modulo debe alimentarse con su propia fuente, descartando el uso de la corriente que entrega USB.

#### 2.5.3.3 Interfase USB 2.0

Se utiliza un microcontrolador USB del fabricante Cypress, el CY68013 FX2 para hacer del XEM3001 un periférico USB. De esta manera el modulo se reconoce como un dispositivo USB Plug and Play en cualquier computador.

Las descargas al FPGA por el puerto USB se realizan de una forma mucho mas rápida que por cualquier otro de los puertos comunes existentes, además de brindar una gran portabilidad ya que la gran mayoría de los computadores tienen puertos USB o se les puede adaptar.

#### 2.5.3.4 Módulos Extra

Aunque el XEM3001 esta diseñado como un dispositivo de bajo costo, agrega unos cuantos módulos que son muy convenientes.

#### 2.5.3.4.1 EEPROM

Hay una pequeña EEPROM conectada al microcontrolador, pero no esta disponible directamente para el FPGA. La EEPROM se usa para almacenar código de inicio para el microcontrolador, datos de configuración del PLL y una cadena de identificación del dispositivo.

El dato de configuración del PLL se carga desde la EEPROM cada vez que un nuevo archivo de configuración es bajado al FPGA. De esa manera se tiene presente las señales de reloj que el usuario desea en los pines adecuados del FPGA.

Los datos de configuración del PLL se pueden cambiar en cualquier momento desde un dialogo de configuración en el FrontPanel (software de control de la tarjeta), al igual que la cadena de identificación del dispositivo.

#### 2.5.3.4.2 PLL

Se cuenta con un PLL CY22150 de Cypress con múltiples SALIDAS que puede proveer hasta 5 señales de reloj, tres están conectados al FPGA y otros dos a los conectores de expansión JP2 y JP3. El PLL es manejado con una señal de reloj de 48 MHz proveniente del microcontrolador USB, puede tener señales de reloj de salida hasta de 150 MHz. Su configuración puede realizarse desde el FrontPanel.

#### 2.5.3.4.3 LEDs y Pulsadores

Se cuenta con 8 LEDs y 4 pulsadores que son muy prácticos en el momento de realizar depuración de código o seguimiento de señales y que ahorran circuiteria adicional.

#### 2.5.3.4.4 Conectores de Expansión

Hay tres conectores de expansión disponibles (JP1, JP2 y JP3) para conectar el modulo XEM3001 a los demás módulos que se desarrollen para la tarjeta de adquisición de datos. Los conectores proporcionan conexiones para voltajes de 3.3 V, tierras, salidas PLL y 88 pines para propósito general de I/O en el FPGA.

#### 2.5.3.4.4 FrontPanel

El software de control del modulo XEM3001 se explica de manera detallada en el Anexo B de la monografía.

## 2.6 ANÁLISIS DEL PROYECTO

Una primera aproximación para el desarrollo de esta monografía es extraída del diagrama de contexto. En este diagrama se puede observar la forma en la que la tarjeta de adquisición se integra en un entorno completo de control.

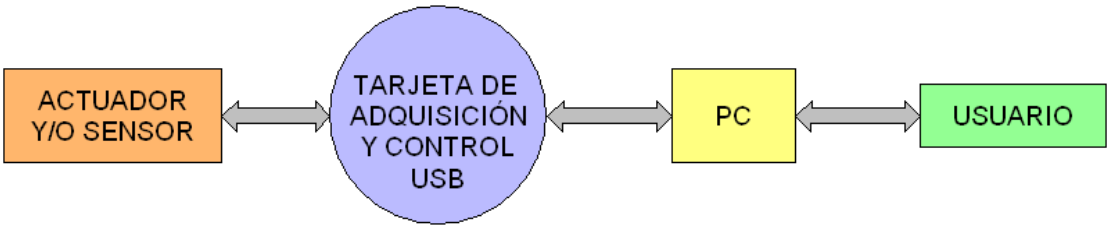



**Figura 2.4 Diagrama de Contexto**

En primera instancia se encuentra en el entorno a controlar una gran cantidad de sensores y actuadores que ayudaran a interactuar con el mundo físico.

Después se tiene la tarjeta de adquisición de datos USB, ella proporciona toda la funcionalidad para tomar las señales provenientes de los sensores, se podrá incluir directamente en ella un tratamiento de esos datos, hacer detección de errores, conversiones en unidades de ingeniería y en fin cualquier tipo de manipulación para acortar el trabajo que generalmente se hace en el PC con otro tipo de tarjetas. En la TAC USB también se procesan las señales de control que han sido enviadas desde el PC o aquellas que han sido generadas en la misma tarjeta y que serán enviadas a los actuadores para obtener cambios en el entorno físico y poder de esa manera realizar un control sobre el.

En el PC se encuentran alojados los elementos de control para la interacción con la TAC USB, ellos agrupan los controladores del dispositivo y las interfaces de usuario. Los controladores permiten detectar e identificar la tarjeta como un dispositivo USB de interfaz humana y la vinculan con el sistema operativo logrando de esta manera una interacción con ella. Los entornos de alto nivel con el que interactúa el usuario están contemplados en las aplicaciones que se desarrollen ya sea en XML, Java, C++, Labview o Matlab, a través de ellas el usuario interactúa de manera transparente con el entorno físico enviando los datos o las señales de control que necesita para afectar un entorno de control o mirando lo que desde el llega para tomar decisiones.

El elemento final de todo el contexto es el usuario, el será quien finalmente interactué y decida si lo que se refleja sobre el entorno que desea controlar es adecuado o no, aunque en muchos sistemas es escasa la participación de este actor, nunca deberá dejársele de lado ya que los sistemas pueden ser muy precisos y bien diseñados, mas aun así no existe nada perfecto en la vida y cuando haya la necesidad de tomar decisiones de índole humana un sistema no tendrá sentimientos y será difícil para él determinar el impacto que sus decisiones pueden causar sobre aquel a quien finalmente debe servir, una muestra de ello son los sistemas de control de la planta nuclear de Chernobyl, entre otros muchos.

Por ello el usuario nunca podrá dejarse totalmente de lado y en el caso de esta monografía no será la excepción.

El mundo de los sensores y actuadores es ya bastante extenso y para el existen muchos elementos que permiten su uso para obtener medidas y acciones de gran exactitud, entre ellos se tiene por ejemplo medidores de presión, elongación, aceleración, distancia, temperatura, nivel de acides, servo válvulas, motores paso a paso y muchos mas. En esta monografía no se abarca ese campo ya que ello la haría muy extensa y además escapa a los alcances para ella proyectada.

La tarjeta de adquisición de datos USB será abarcada en el capitulo 3 de esta monografía en su diseño y estructura física, en dicho capitulo se explica la manera como se abordan los módulos fundamentales que componen la tarjeta y todo lo concerniente al hardware de la misma. En el capitulo 4 se afronta la elaboración del software implicado en la tarjeta, se diseña cada uno de los módulos fundamentales y se explica el contexto general en el que estarán vinculados ellos.

# 3 DISEÑO DETALLADO DEL HARDWARE DE LA TAC

## 3.1 INTRODUCCIÓN

En esta parte se describe el proceso de diseño y posterior implementación del componente hardware de la TAC. El proceso del diseño se hace en dos etapas: diseño preliminar, en donde se realiza una descripción general de los bloques de la TAC y el diseño detallado, en donde se hace una descripción por módulos de los componentes seleccionados para su montaje, se describen las características generales, la disposición de pines, la conexión básica y la configuración de los circuitos integrados (CI) empleados. Se hace una explicación de los diagramas de tiempo de las señales que manejan dichos CI, la cual servirá para elaborar los diagramas de flujo que resumen el funcionamiento general del modulo y el desarrollo del componente software que lo controla, en el siguiente capitulo.

Los diagramas pin a pin del diseño completo de la TAC, así como el diseño de las tarjetas de circuito impreso se pueden consultar en el anexo A.

## 3.2 DISEÑO PRELIMINAR DE LA TAC

El diagrama de la figura 3.1 ilustra los principales bloques funcionales de los que se compone la TAC.

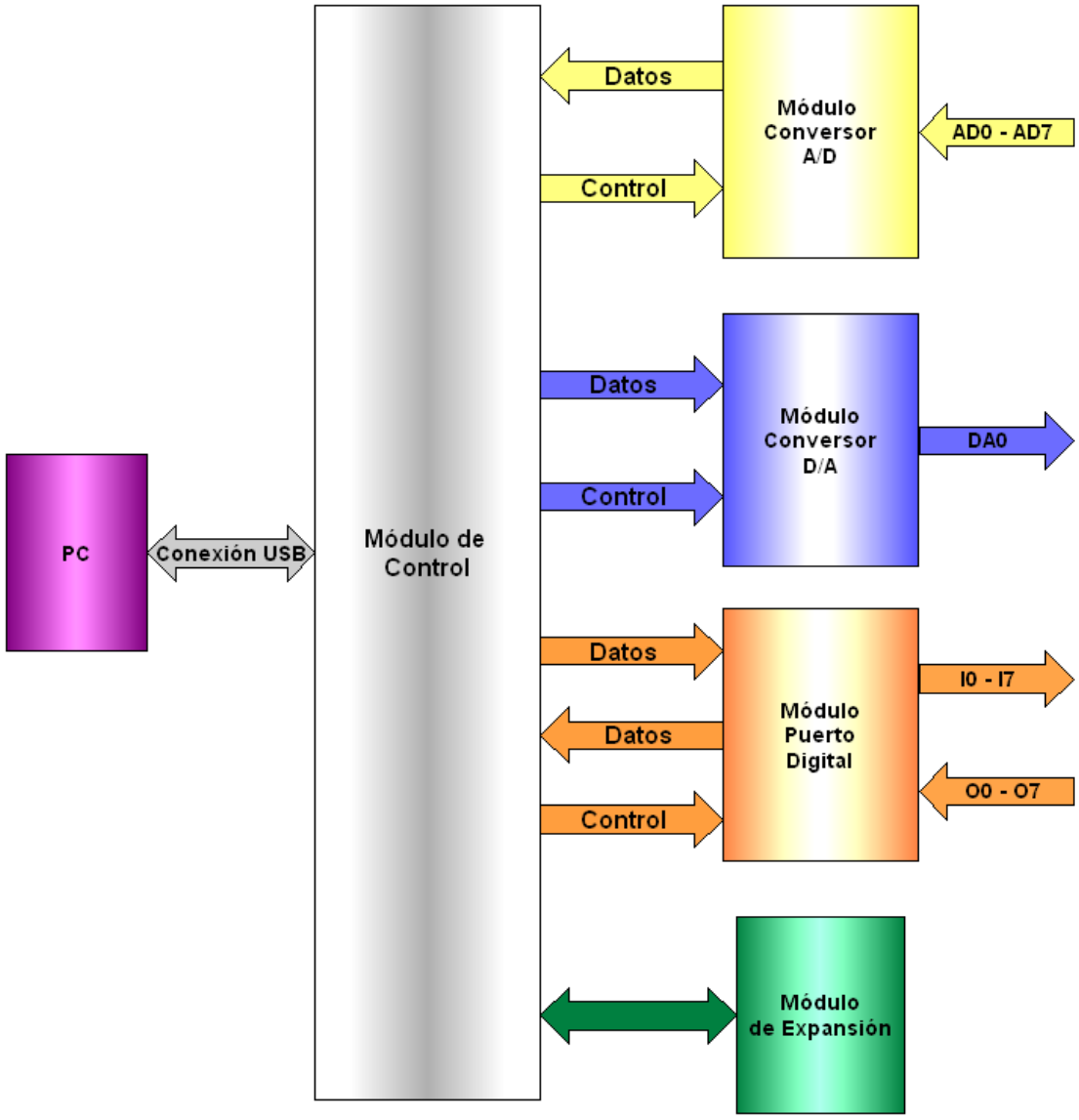



**Figura 3.1 Diagrama de Módulos de la TAC**

Cada uno de los anteriores módulos desempeñará las siguientes funciones:

**Módulo de Control:** esta compuesto por la tarjeta de desarrollo XEM3001 de la empresa Opal Kelly cuyo corazón es el FPGA de la familia Spartan III de Xilinx, descrita en el capitulo 2. Es el encargado de gestionar los procesos de

comunicación concernientes al protocolo USB, como son la conexión, intercambio de datos y desconexión de la TAC al PC. También esta encargado de gestionar el intercambio de datos entre los módulos de la TAC y el PC a través de la conexión por cable entre su interfaz USB y un puerto USB libre del PC. La comunicación entre este modulo y los demás se hace a través de los puertos JP1, JP2, y JP3 de la tarjeta de desarrollo, los cuales en conjunto ponen a disposición 88 líneas I/O del FPGA en total, que son suficientes para manejar todos los módulos.

**Módulo Conversor A/D:** esta compuesto por 8 líneas de entrada analógicas y tiene la función de realizar la conversión de señales analógicas que se apliquen en sus entradas en una representación digital que pueda ser entendida y procesada por el PC. Estas señales pueden provenir de algún sensor que mida una variable física, que requiera ser medida o controlada (como un sensor de temperatura).

**Módulo Conversor D/A:** esta compuesto por una línea de salida analógica y tiene la función de realizar la conversión de cualquier representación digital de una señal generada en el PC en su correspondiente representación analógica en dicha salida. Esta señal analógica de salida puede ser empleada para manipular un actuador (como una válvula) que controle una variable física o para generar formas de onda (en un generador de señales).

**Módulo de Puerto Digital:** esta compuesto de 8 líneas de entrada y 8 de salida, a través de las cuales se pueden introducir o sacar señales del PC directamente en su representación digital para ser usadas en propósitos diversos como controlar reles que manejen cargas de alta potencia o supervisar el estado de un interruptor on/off.

**Módulo de Expansión:** la tarjeta de desarrollo cuenta con varias líneas de I/O del FPGA, las cuales se emplearán para adicionar más módulos a la TAC. Para este

propósito se emplearan todas las 36 líneas disponibles en el puerto JP3, repartidas en cuatro grupos de nueve líneas cada uno, de tal forma que se disponga de cuatro puertos de expansión con nueve líneas de I/O del FPGA para conectar módulos adicionales.

Físicamente la TAC consta de las partes (tarjetas de circuito impreso) que se muestran en la figura 3.2, las cuales se describen a continuación:

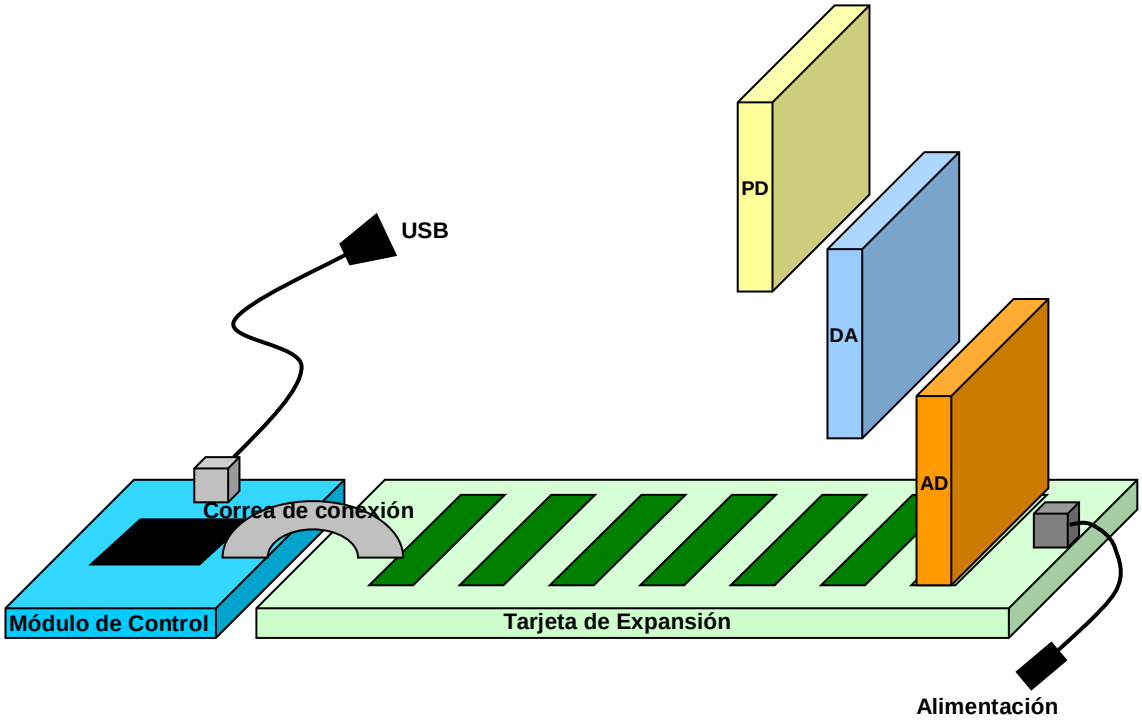



**Figura 3.2 Diagrama de Tarjetas de Circuito Impreso de la TAC**

**Modulo de control:** conformado el modulo de desarrollo XEM3001.

**Tarjeta de expansión:** conformado por una placa base que aloja los siete puertos de expansión de la TAC, tres de los cuales son utilizados por los módulos descritos anteriormente. Estos puertos de expansión constan de conectores tipo

"pin header" macho. Esta tarjeta además aloja el circuito de alimentación de voltaje que proporciona la polarización adecuada a las tarjetas que se conectan a los puertos de expansión. La comunicación entre la tarjeta de control y esta, se realiza por medio de correas de datos tipo "ribbon".

**Tarjeta del módulo conversor A/D:** contiene el circuito del conversor A/D y el conector hembra que le permite conectarse al respectivo puerto en la tarjeta de expansión.

**Tarjeta del módulo conversor D/A:** contiene el circuito del conversor D/A y el conector hembra que le permite conectarse al respectivo puerto en la tarjeta de expansión.

**Tarjeta del módulo de Puerto Digital:** contiene el circuito del puerto digital y el conector hembra que le permite conectarse al respectivo puerto en la tarjeta de expansión.

Los detalles del diseño de estas tarjetas se pueden consultar en el anexo A.

## 3.3 DISEÑO DETALLADO DE LA TAC

A continuación se realiza una explicación detallada del diseño y selección de los componentes que conforman cada uno de los módulos básicos de la TAC, descritos anteriormente.

### 3.3.1 Módulo Conversor A/D

Para este módulo se empleo el CI AD7829 de la empresa Analog Devices. Se escogió este circuito integrado por ser de uso general en sistemas de adquisición de datos, por poseer una velocidad de muestreo que se ajusta a los requerimientos de la TAC y por ser da fácil adquisición en el mercado. Este circuito integrado tiene las siguientes características:

- Resolución de 8 bits
- Tipo de conversión Half-Flash
- 8 canales de entrada analógicos unipolares
- Modulo Track-and-Hold interno
- Tiempo de conversión de 420ns
- Rango amplio de voltajes de alimentación 3V y 5V +/-10%
- Rangos de voltaje de entrada de 0V a 2Vp-p con $V_{DD}$ = 3V y de 0V a 2,5Vp-p con $V_{DD}$ = 5V
- Interfaz de control paralela flexible

#### 3.3.1.1 Configuración de pines

La figura 3.3 muestra la disposición de pines del circuito integrado:

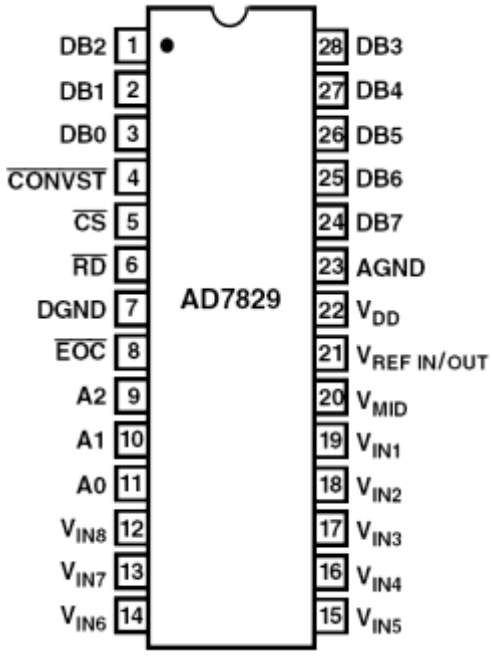



**Figura 3.3 Diagrama de Pines del CI AD7829**

### 3.3.1.2 Descripción de la función de los pines

El CI cuenta con seis entradas lógicas para el control completo del proceso de conversión y lectura de datos, las cuales se detallan a continuación:

- **CONVST:** señal de inicio de la conversión. La conversión inicia en el flanco de bajada de esta señal. Un nivel bajo en ella sitúa el circuito track-and-hold en modo hold (retención), el cual vuelve automáticamente al estado track después de 120ns.
- **CS:** Chip Select – activa o desactiva las salidas paralelas del CI (nivel alto, desactiva y bajo, activa)
- **RD:** Read - señal para la lectura del resultado de la conversión. Habilita los buffers de salida de datos (los saca del estado de alta impedancia).

- **$A_0$ $A_1$ $A_2$:** señales de selección del canal analógico de entrada, cuya señal se muestreara. El canal seleccionado depende del valor lógico establecido en estas entradas como lo muestra la tabla 1.

De igual manera el CI posee las siguientes señales de salida:

- **EOC:** End Of Conversion – señal de fin de la conversión. Un nivel lógico bajo en esta señal indica que la conversión se ha realizado satisfactoriamente.
- **$DB_0$ – DB7:** líneas de salida de datos. Se encuentran normalmente en estado de alta impedancia, hasta que las señales RD y CS van a estado bajo.

Los siguientes pines están relacionados con la polarización del CI:

- **$V_{DD}$:** voltaje de alimentación positivo, 3V ± 10% y 5V ± 10%.
- **AGND y DGND:** tierra analógica y digital respectivamente.
- **$V_{REF\ IN/OUT}$:** referencia de voltaje analógico de entrada y salida. En este pin se puede conectar a un voltaje de referencia externo, de no ser así, en el esta disponible el voltaje de referencia interno del CI. Cuando se usa como referencia interna, el pin se deja sin conexión o puede conectarse a tierra a través de un condensador de 0,1µF.

Los canales de entrada analógicos se manejan a través de las siguientes 8 señales:

- **$V_{IN1}$ a $V_{IN8}$:** cada entrada posee un span (corrimiento) de entrada de 2,5V y 2V, dependiendo del voltaje de alimentación $V_{DD}$. Este span puede centrarse en cualquier valor dentro del rango determinado por AGND y $V_{DD}$ usando el pin $V_{MID}$.

| $A_2$ | $A_1$ | $A_0$ | Entrada Analógica Seleccionada |
|---|---|---|---|
| 0 | 0 | 0 | $V_{IN1}$ |
| 0 | 0 | 1 | $V_{IN2}$ |
| 0 | 1 | 0 | $V_{IN3}$ |
| 0 | 1 | 1 | $V_{IN4}$ |
| 1 | 0 | 0 | $V_{IN5}$ |
| 1 | 0 | 1 | $V_{IN6}$ |
| 1 | 1 | 0 | $V_{IN7}$ |
| 1 | 1 | 1 | $V_{IN8}$ |

**Tabla 3.1 Tabla de Verdad para las Señales de Selección de Canal**

### 3.3.1.3 Conexión del CI

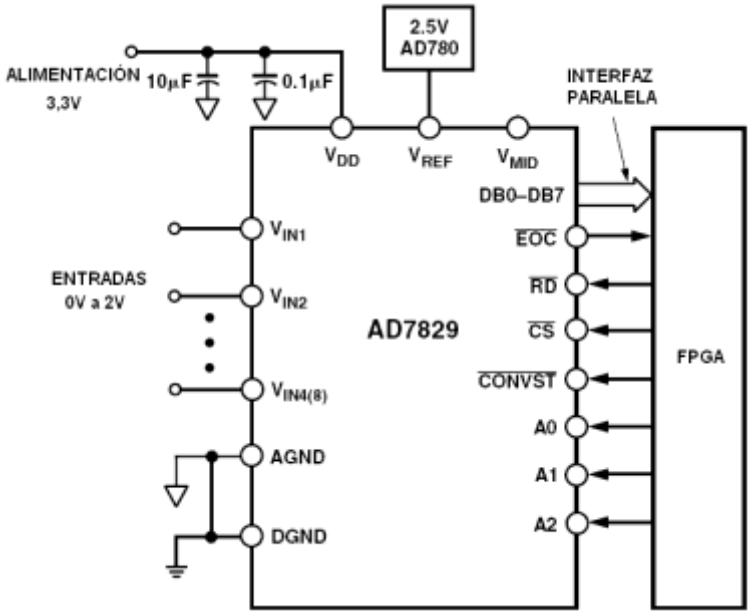



**Figura 3.4 Diagrama de Conexión del CI AD7829**

La figura 3.4 muestra la conexión del CI. Las tierras AGND y DGND se conectan entre si y deben estar muy próximas al CI para obtener una inmunidad al ruido buena. $V_{REF}$ se conectan a una referencia de voltaje y $V_{DD}$ se conecta a 3,3V con el fin de que el CI opere con los mismos niveles de voltaje bajo (tecnología LVT)

que soporta el FPGA. El fabricante del CI recomienda no dejar la señal CONVST sin estado fijo (flotando) cuando se aplique $V_{DD}$, puesto que esto colocaría el CI en un estado indeterminado. Por ello se opto por colocar una resistencia de pull-down entre DGND y este pin para mantenerla en bajo cuando se alimente el CI.

Como se mencionó en la descripción de pines, cada una de las 8 entradas analógicas posee un span de entrada de 2,5 o 2V dependiendo del valor de $V_{DD}$. El valor del span determina el rango de variación que puede tener la señal analógica de entrada y para fijarlo se emplea la señal $V_{MID}$. A través de esta señal se puede ajustar el valor del span en cualquier punto entre AGND y $V_{DD}$, como lo muestra la figura 3.5 para un valor de $V_{DD}$ de 3V. Como se puede notar, cuando no se aplica ningún voltaje a $V_{MID}$ el rango de entrada por defecto puede variar entre AGND y 2V centrado en 1V. Para este diseño la señal $V_{MID}$ se deja sin conexión, de forma que el rango de las señales analógicas que se pueden aplicar en las entradas del conversor pueden variar entre 0 y 2V, solamente.

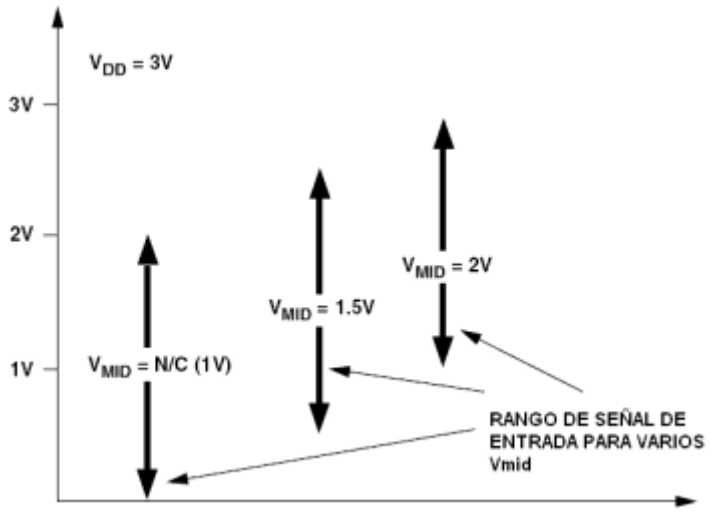



**Figura 3.5 Variación del Span de Entrada Analógico con $V_{MID}$**

Para generar el voltaje de referencia de 2,5V que requiere el conversor A/D se emplea el CI AD780, también de la empresa Analog Devices. La figura 3.6 muestra la conexión de este CI. El pin 8 (2,5 / 3,0 select) se deja abierto (NC) para que la referencia sea de 2,5V.

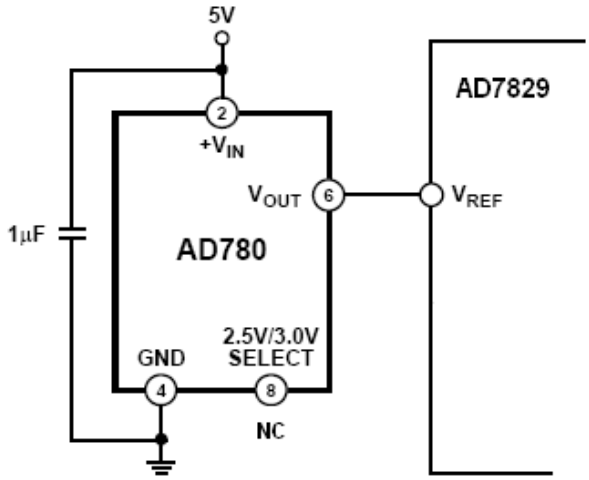



**Figura 3.6 Conexión de la Referencia de Voltaje**

### 3.3.1.4 Proceso de Conversión / Lectura

La figura 3.7 ilustra el diagrama de tiempos de las señales involucradas en el proceso de conversión del CI, para el muestreo de un canal o varios en forma secuencial.

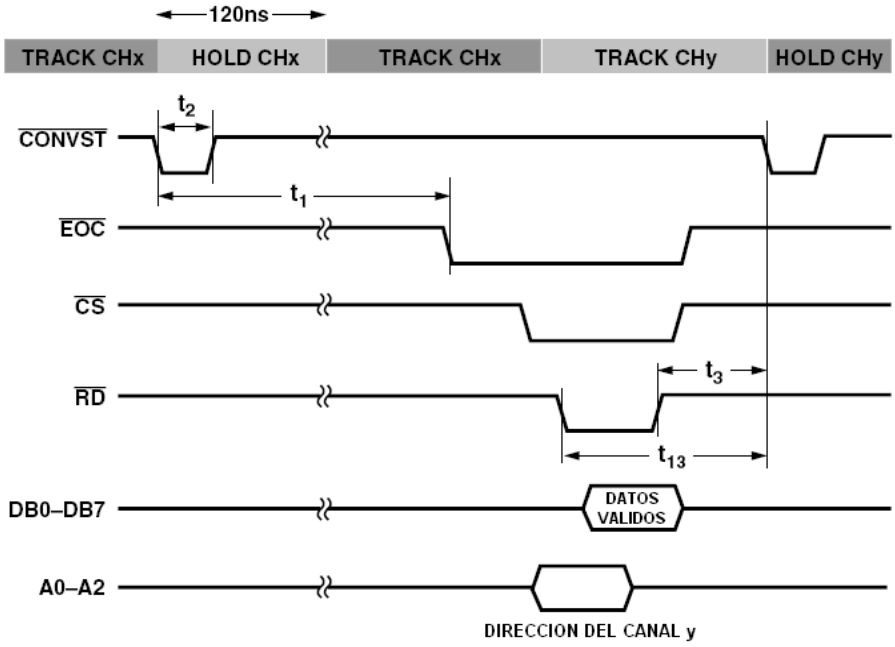



**Figura 3.7 Diagrama de Tiempos para el Muestreo de los Canales**

La selección del canal a muestrear se realiza por medio de las señales $A_0 – A_2$, de acuerdo a la combinación de valores dados en la tabla 1. Cuando se alimenta el CI la señal $V_{IN}$ seleccionada por defecto es la $V_{IN1}$. La dirección del siguiente canal a muestrear, para el caso del muestreo secuencial, se fija al inicio de la operación de lectura del canal anterior, es decir en el flanco de bajada de la señal RD, como lo muestra la figura 3.7. Existe un retardo de tiempo mínimo entre el flanco de bajada de RD y el siguiente flanco de bajada de la señal CONVST determinado por $t_{13}$ (200ns).

**Inicio de la conversión:**

El proceso de conversión, como se puede ver en la figura 3.7, comienza cuando la señal CONVST va al estado bajo y permanece ahí por un tiempo no menor a $t_2$ (20ns). El circuito track-and-hold va al modo hold y retiene la señal analógica de entrada durante 120ns para fijar la fase de adquisición. Después de retener la señal la conversión se lleva a cabo durante un tiempo no menor a 420ns ($t_1$), tiempo después del cual la señal EOC cambia a estado bajo para indicar el fin de esta. La señal EOC puede permanecer en este estado durante un tiempo no mayor a 110ns, sin embargo, puede ser reestablecida al estado alto por el flanco de subida de la señal RD.

**Lectura de la conversión:**

La lectura comienza cuando la señal RD se fija en el estado bajo inmediatamente después de que la señal CS también se establece en bajo. La señal RD debe permanecer en este estado por un tiempo no menor a 30ns y el tiempo entre el instante que esta vuelve al estado alto y el inicio de otra conversión, no debe ser inferior a los 30ns ($t_3$). Existe un retardo de aproximadamente 10ns entre el instante que la señal RD va al estado bajo y el momento en que los datos se hacen validos en los buffers de salida (señales DB0 a DB7), que hay que tener en cuenta.

#### 3.3.1.5 Conexión con el módulo de control

Para la comunicación con el modulo de control las líneas de datos y de control del CI se conectan con los puertos de la tarjeta de desarrollo como lo indica la tabla A.1 en el anexo A.

### 3.3.2 Módulo Conversor D/A

Es conformado por el CI CA3338 de la empresa Intersil, el cual posee las siguientes características básicas:

- Tasa de muestreo de 50MSPS
- Resolución de 8 bits
- Bajo consumo de potencia
- Entradas compatibles con TTL/CMOS
- Tiempo de establecimiento típico de 20ns
- Alimentación simple de +5V

La figura 3.8 y tabla 3.2 muestran la disposición y descripción funcional de cada uno de los pines del circuito integrado:

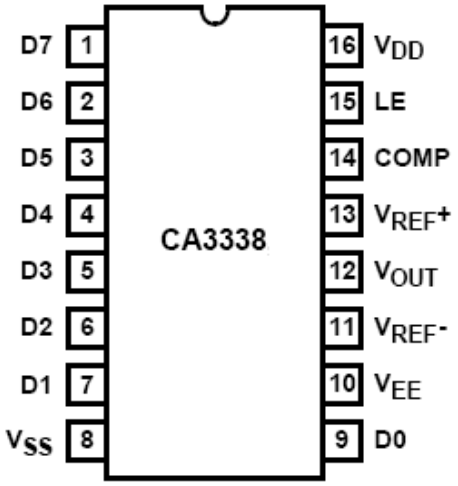



**Figura 3.8 Diagrama de Pines del CI CA3338**

| PIN | NOMBRE | DESCRIPCIÖN |
|---|---|---|
| 1 | D7 | Bit Mas Significativo |
| 2 | D6 | Bits |
| 3 | D5 | De Datos |
| 4 | D4 | De Entrada |
| 5 | D3 | (Alto = Verdad) |
| 6 | D2 | |
| 7 | D1 | |
| 8 | $V_{SS}$ | Tierra Digital |
| 9 | $D_0$ | Bit Menos Significativo. Bit de Datos de Entrada |
| 10 | $V_{EE}$ | Tierra Analógica |
| 11 | $V_{REF}$- | Entrada Negativa del Voltaje de Referencia |
| 12 | $V_{OUT}$ | Salida Analógica |
| 13 | $V_{REF}$+ | Entrada Positiva del Voltaje de Referencia |
| 14 | COMP | Entrada de Control del Complemento de Datos. Activa Alta |
| 15 | LE | Entrada de Habilitación de Latch. Activa Baja |
| 16 | $V_{DD}$ | Fuente de Alimentación Digital, +5V |

**Tabla 3.2 Descripción de Pines del CI CA3338**

### 3.3.2.1 Interfaz de control

El control del CI se realiza básicamente a través de la señal LE (Latch Enable), de acuerdo a la tabla 3.2 Esta señal es activa baja, por lo que el proceso de

conversión D/A se inicia después de que la señal aplicada en este pin cambia del estado alto al bajo (en el flanco de caída).

Los pines marcados como D0 a D7 corresponden a los pines donde se aplica la palabra de 8 bits de la señal digital que se convertirá a analógica.

La señal COMP (Data Complement) permite complementar la palabra aplicada en los pines de entrada. Si COMP es alta los bits de la palabra se invierten. Para el diseño esta señal se mantiene en estado bajo permanentemente.

### 3.3.2.2 Conexión del CI

La figura 3.9 muestra la conexión básica de funcionamiento del CI

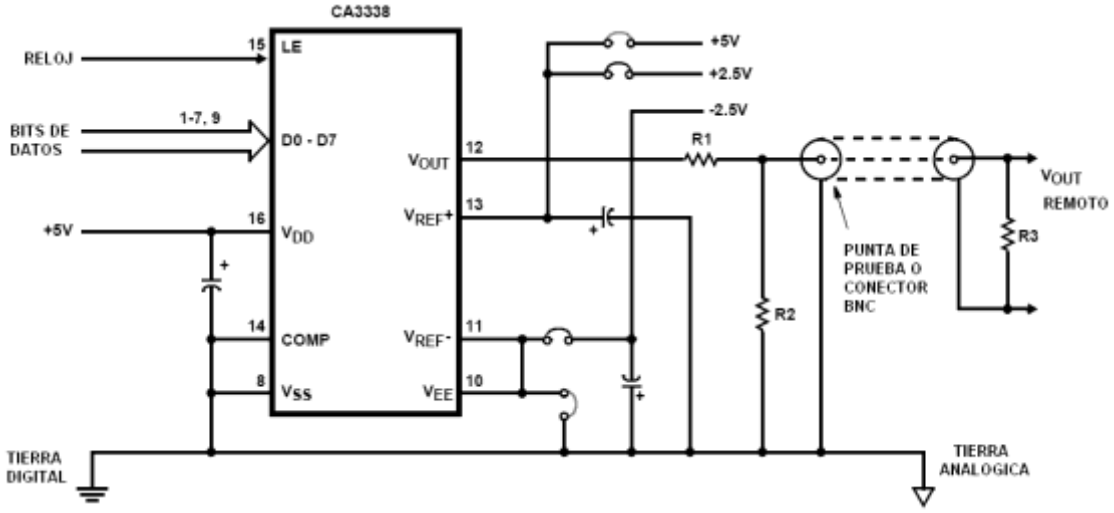



**Figura 3.9 Conexión Básica del CI CA3338**

En el diagrama de la figura 3.9 hay que tener en cuenta que dependiendo de los voltajes de referencia aplicados en los pines $V_{REF}^{+}$ y $V_{REF}^{-}$ se puede operar el conversor en modo unipolar o bipolar. Las conexiones grises en estos pines son para operación en modo unipolar y las conexiones negras para modo bipolar.

Para funcionamiento en modo unipolar $V_{REF}^-$ se conecta directamente a la tierra analógica $V_{EE}$, procurando establecer un camino de muy impedancia entre ellos y $V_{REF}^+$ se conecta a $V_{DD}$.

Para funcionamiento en modo bipolar $V_{REF}^-$ se conecta a un voltaje negativo, que depende del valor de $V_{DD}$ en una relación aproximada a $V_{DD}$ – 8V. Para el diseño de la TAC el conversor trabajara en modo bipolar, con el propósito de proporcionarle más funcionalidad.

Los valores de R1, R2 y R3 dependen de la impedancia del cable que se emplee para obtener el voltaje de salida analógico del conversor. Para una impedancia típica de 75Ω se deben emplear los siguientes valores: R1=18Ω, R2=130Ω y R3=75Ω.

Todos los condensadores son cerámicos con valor de 0,1µF.

### 3.3.2.3 Proceso de conversión

Como ya se menciono anteriormente, el proceso de conversión es controlado solamente por la señal LE. La figura 3.10 muestra los tiempos que hay que tener presente para llevar a cabo una conversión.

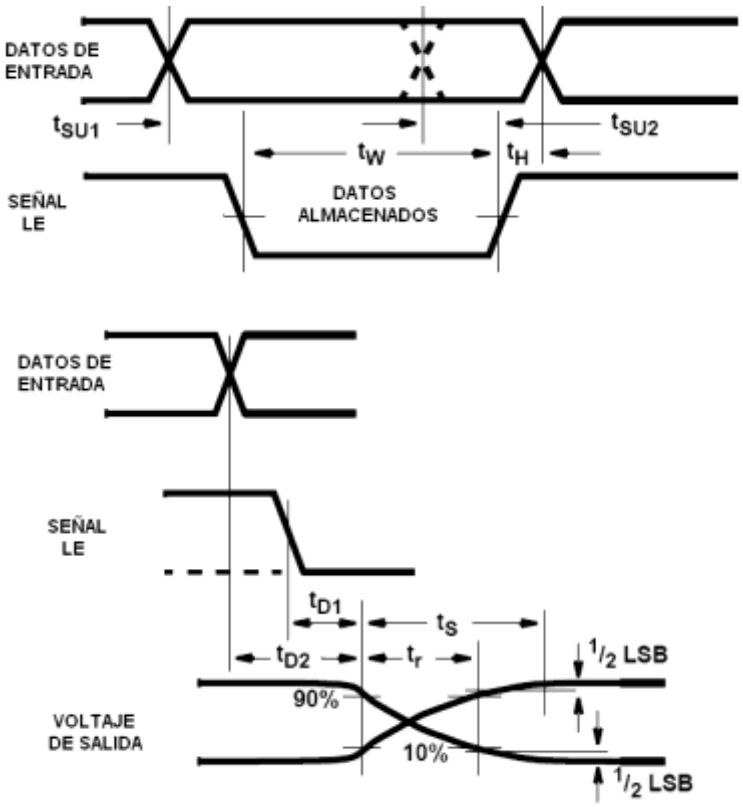



**Figura 3.10 Diagrama de Tiempos para la Conversión**

Una vez aplicados los datos digitales a convertir en las entradas se debe esperar un tiempo determinado por $t_{SU1}$ (2ns) para que estas se estabilicen. Después de este tiempo la señal LE puede ir a bajo para almacenar los datos en los buffers de entrada del conversor y debe permanecer en ese estado por un tiempo no menor a $t_W$ (25ns) para que la conversión se haga correctamente. La conversión se obtiene a la salida después de un tiempo $t_{D1}$ (25ns) contados a partir del flanco de bajada de LE. La siguiente conversión se puede hacer después de un tiempo determinado por $t_H$ (5ns) contados a partir del punto en que la señal LE vuelve al nivel alto.

### 3.3.2.4 Conexión con el modulo de control

Para la comunicación con el modulo de control las líneas de datos y de control del CI se conectan con los puertos de la tarjeta de desarrollo como lo indica la tabla A.1 en el anexo A.

### 3.3.3 Modulo de Puerto Digital

Para este modulo se opto por utilizar 16 líneas I/O de la tarjeta de desarrollo, 8 líneas de entrada y 8 de salida; puesto que el FPGA cuenta con suficientes y de este modo se aumenta la funcionalidad de la TAC.

#### 3.3.3.1 Puerto de salida

Debido a que las líneas I/O del FPGA no tienen la capacidad de surtir niveles altos de corriente para manejar cargas pesadas, las 8 líneas de salida de este puerto requieren que sean operadas a través de un CI que les permita conectar en ellas dispositivos con exigencias de corriente altos. El CI más adecuado para este propósito es el 74LS06, el cual está compuesto por seis impulsores inversores de corriente de colector abierto y posee las siguientes características básicas:

- Tiempo de propagación típico de 8ns
- Conversión de niveles de voltaje TTL a niveles CMOS
- Capacidad de manejo de corrientes altas
- Salidas de colector abierto para el manejo directo de cargas altas
- Entradas compatibles con la mayoría de circuitos TTL

La figura 3.11 muestra la disposición de pines del CI 74LS06.

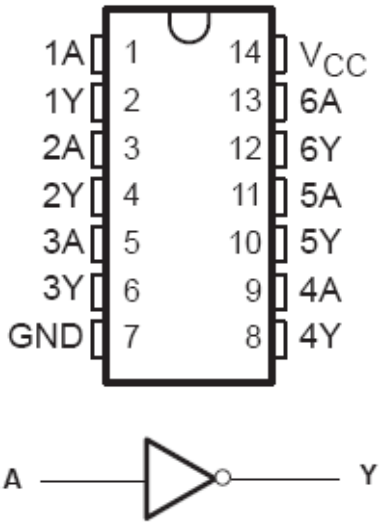



**Figura 3.11 Diagrama de Pines del CI 74LS06**

Las salidas de colector abierto de este CI le permiten conectarse con circuitos digitales que operen con niveles más altos de voltaje (CMOS) o para manejar cargas con consumo de corriente alto. El voltaje máximo de salida que puede soportar es de 30V y la corriente máxima que puede drenar es de 40mA, de modo que pueden conectarse en sus salidas directamente reles o lámparas indicadoras.

Para operar las 8 líneas de salida del puerto se emplean dos CI, ya que uno solo de ellos posee 6 impulsores.

**Conexión del CI:**

La figura 3.12 muestra la conexión de un impulsor del CI, los demás 7 se conectan de forma igual.

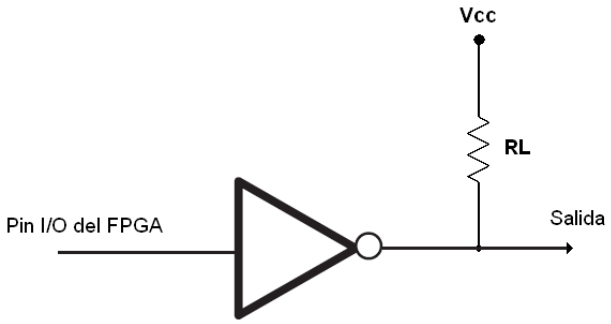



**Figura 3.12 Conexión Básica del CI 74LS06**

RL es la carga que se conecta a la salida y puede ser un rele o una resistencia de pull-up, cuyo valor depende del voltaje Vcc de salida del impulsor. El valor máximo de Vcc es de 30V. En el montaje final de la TAC estas salidas se dejan abiertas con el propósito de permitir que el usuario las adapte al uso específico que se les quiera dar. El valor de la resistencia de pull-up esta determinado por las siguientes ecuaciones:

$$R_{MAX} = \frac{V_O(\min) - V_{OH}}{N_1(I_{OH}) + N_2(I_{IH})}$$

$$R_{MIN} = \frac{V_O(\max) - V_{OL}}{I_{OL} - N_3(I_{IL})}$$

En donde: $N_1(I_{OH})$ es la corriente de salida del estado alto máxima total para todas las salidas conectadas a la resistencia de pull-up.

$N_2(I_{IH})$ es la corriente de entrada del estado alto máxima total para todas las entradas conectadas a la resistencia de pull-up.

$N_3(I_{IL})$ es la corriente de entrada del estado bajo máxima total para todas las entradas conectadas a la resistencia de pull-up.

El valor de los voltajes y corrientes máximos y mínimos de las ecuaciones se pueden consultar en la hoja de especificaciones del fabricante del CI.

### 3.3.3.2 Puerto de entrada

Las 8 líneas de entrada requieren el empleo de un CI traductor de nivel, puesto que los pines I/O del FPGA no toleran niveles de más de 3,3V en sus entradas. El CI mas adecuado para este propósito es el SN74AHC244 de la empresa Texas Instruments. Este CI esta diseñado para operar con tecnología de bajo voltaje (LVT) por lo que permite realizar la interfaz entre los niveles TTL (0 a 5V) que se pueden aplicar a las entradas de este puerto con los niveles de voltaje que soportan los pines de entrada del FPGA (0 a 3,3V). Este CI tiene las siguientes características básicas:

- Rango amplio en el voltaje de operación 2 a 6V.
- Salidas con capacidad alta de manejo de corriente. Maneja hasta 15 cargas TTL.
- Consumo bajo de potencia.
- Tiempo de propagación típico de 11ns.

La figura 3.13 muestra la configuración de pines de este CI.

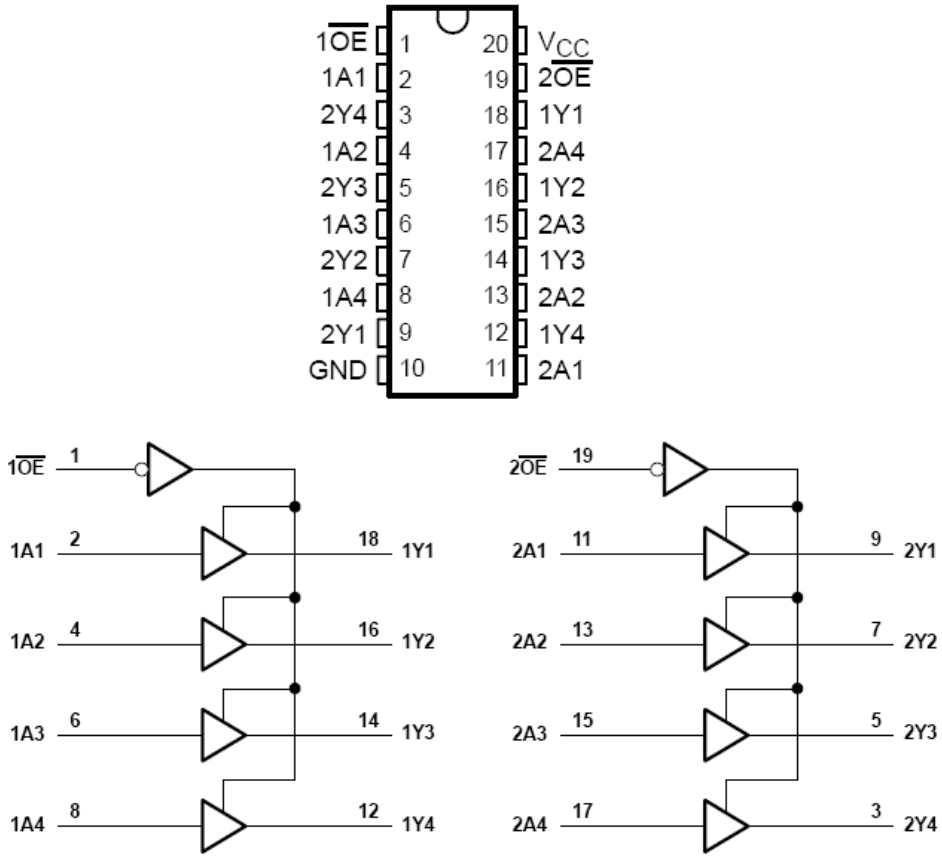



**Figura 3.13 Diagrama de Pines del CI 74AHC244**

Las entradas OE (pines 1 y 19) son señales de control que se emplean para activar o poner en modo de alta impedancia las salidas (Y) del CI. Estas señales son activas baja por lo que con un nivel bajo aplicado en ellas activan las salidas y entregan el nivel lógico aplicado en las entradas (A) y con un nivel alto establecen dichas salidas en estado de alta impedancia. Las entradas OE son controladas por el software de la TAC, de tal forma que las salidas del CI se activen cuando se requiera realizar una lectura de datos y permanezcan en estado de alta impedancia, cuando el modulo no este activo o no se desee hacer una lectura. Esto se hace también con el propósito de proteger a los pines del FPGA contra alguna conexión accidental que los pueda dañar.

**Conexión del CI:**

El CI se debe polarizar con un voltaje de 3,3V, para que la conversión entre las señales de entrada y salida se haga de manera correcta. En las entradas A del CI se aplica la palabra digital que se desee leer, con niveles TTL (0 a 5V), y en las salidas Y se obtiene la misma palabra traducida a los niveles de bajo voltaje (LVT) que soportan los pines I/O del FPGA.

#### 3.3.3.3 Conexión con el modulo de control:

Para la comunicación con el modulo de control las líneas de datos y la de control del CI se conectan con los puertos de la tarjeta de desarrollo como lo indica la tabla A.3 en el anexo A.

### 3.3.4 Módulo de Expansión

Este modulo consta simplemente de las líneas I/O del FPGA destinadas para conectar tarjetas futuras en las puertos de expansión de la TAC. La conexión de este modulo con el de control se detalla en la tabla A.2 del anexo A.

Además de estas líneas, cada ranura de expansión cuenta con dos líneas de tierra y dos de alimentación de voltaje (5V y 3,3V). Los detalles de la conexión de estos voltajes se pueden ver en el diagrama pin a pin de la tarjeta de expansión.

### 3.3.5 Alimentación de Potencia

Teniendo en cuenta los requerimientos de alimentación de los CIs que conforman los módulos descritos anteriormente, la TAC debe disponer de dos voltajes de alimentación: 3,3V y 5V.

El voltaje de alimentación de 5V se obtiene con el regulador de voltaje 7805, conectado como lo indica la figura 3.14.

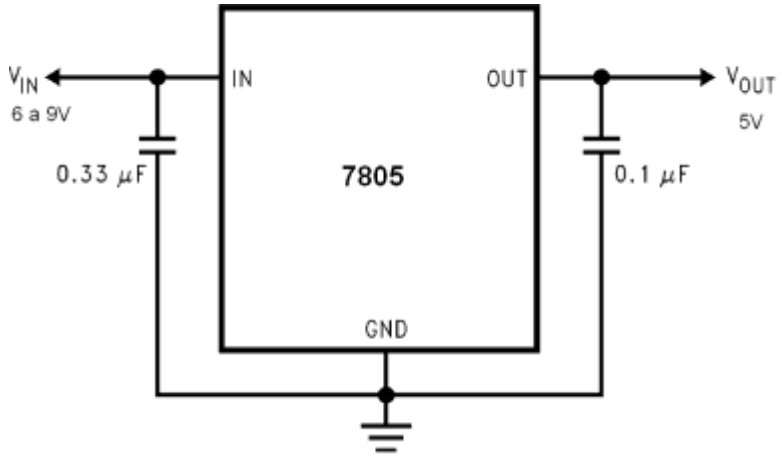



**Figura 3.14 Conexión de Regulador de Voltaje 7805**

El voltaje de alimentación de 3,3V se obtiene con el regulador de voltaje ajustable LM317, conectado como lo indica la figura 3.15.

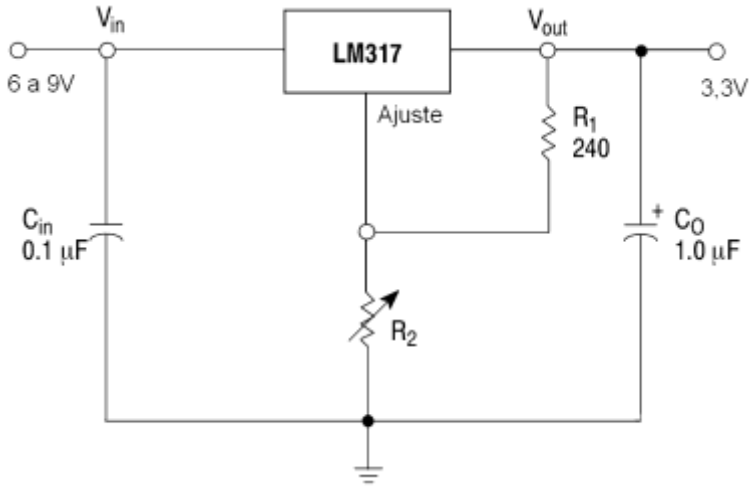



**Figura 3.15 Conexión del Regulador de Voltaje LM317**

En la figura 3.15 $R_2$ es un reóstato de 2KΩ que se ajusta hasta obtener a la salida del regulador los 3,3V deseados.

Ambos reguladores se alimentan con un voltaje DC común en sus entradas que varia entre 6 y 9V. Se seleccionó este rango, a pesar de que ellos soportan hasta 40V en sus entradas, para que no tengan que disipar una potencia muy alta que los sobrecaliente. Los condensadores conectados en las entradas y salidas sirven para filtrar ruidos que pueda introducir la fuente el voltaje de entrada ($V_{IN}$).

# 4. DISEÑO DETALLADO DEL SOFTWARE DE LA TAC

## 4.1 INTRODUCCIÓN

Se describe a continuación el diseño de los módulos fundamentales que harán parte de la tarjeta de adquisición de datos USB. Es importante agregar que todas las códigos se han desarrollado y compilado bajo el entorno de desarrollo ISE 8.2i de Xilinx y que es muy conveniente realizar las modificaciones sobre ese mismo entorno o uno superior para evitar problemas de compatibilidad en los archivos y en las estructuras de programación usadas.

## 4.2 MODULO CONVERSOR A/D

El diagrama de la figura 4.1 explica el proceso a seguir para la conversión de una señal aplicada en alguno de los 8 canales analógicos de entrada del conversor analógico digital.

Se ilustra los pasos a seguir para un ciclo de conversión, por lo tanto el proceso se debe repetir desde el punto “Iniciar Conversión”, tantas veces como dure la señal analógica a muestrear, y con una periodicidad (tiempo de conversión) determinado por la tasa de muestreo deseada. La tasa de muestreo no podrá ser mayor a 2 MSPS con tiempo de conversión de 420ns, siendo ello lo máximo soportado por el CI AD7829.

Los tiempos de retardo entre los procesos, indicados en el diagrama, se establecieron teniendo en cuenta los valores típicos recomendados por el fabricante del CI en la hoja de especificaciones

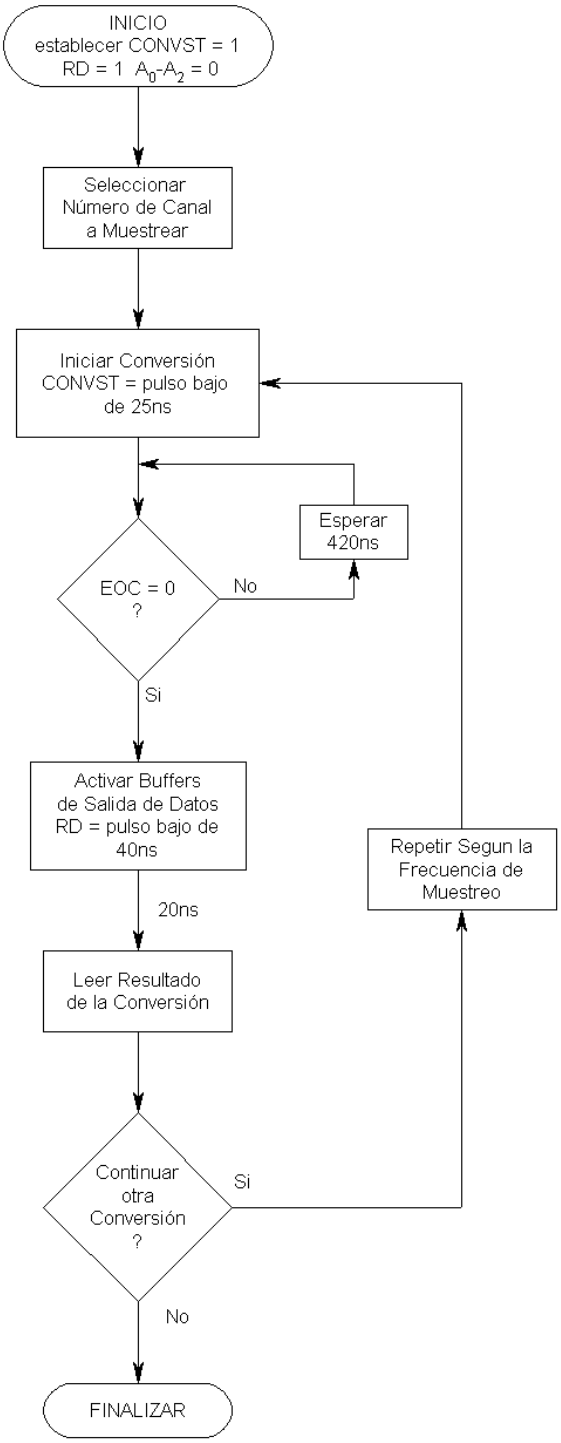



**Figura 4.1 Diagrama de Flujo del Proceso de Conversión A/D**

En su estado inicial el modulo fija todas las señales y las variables a sus niveles más convenientes. Después se selecciona el canal a muestrear cambiando los valores lógicos de la palabra A0 – A2. Se genera luego un pulso bajo de 25 ns que es enviado al pin CONVS. Posteriormente se mira si se ha llegado al fin de la conversión, si no es así se realiza una temporización de 420 ns para activar en seguida los buffers de salida de datos mediante un pulso bajo en RD de 40 ns. Después del pulso se genera un tiempo de 20 ns para permitir que los datos se establezcan en el puerto y poderlos leer. Finalmente se indaga si se desea una nueva conversión, si es así se vuelve al punto iniciar conversión, de lo contrario se finalizan las operaciones.

Para agregar este modulo a cualquier diseño de nivel superior se toma el archivo VHDL que le permite comportarse como un componente, el se encuentra en el CD con archivos digitales (Anexo C) en la carpeta **software/componentes.**

## 4.3 MODULO CONVERSOR D/A

El diagrama de flujo de la figura 4.2 es el resultado del análisis de las características comunes de los conversores D/A del ambiente electrónico. Más exactamente el diagrama de flujo responde al comportamiento del CI CA3338 de Intersil, descrito en el capitulo 3 de esta monografía.

El resultado de la descripción algorítmica del diagrama de flujo en código VHDL es un archivo nombrado DA.vhd, el cual se crea como un componente que puede reutilizarse en cualquier tipo de código de nivel superior lo que permite apreciar la modularidad de la TAC.

El código del modulo, totalmente comentado y con las debidas instrucciones para su uso están contenidos en el anexo C en la carpeta **software/componentes.**

Los pasos de conversión son bastante sencillos e intuitivos, cuando se coloca la palabra a convertir en los pines destinados para tal fin se cambia el valor lógico que saca al integrado de su estado de reposo, es decir que se cambia el valor de LE a 0, se espera 25 nanosegundos para poder leer el valor en los pines de salida que entregan el nivel analógico correspondiente a la palabra de entrada. Una vez leído el dato se regresa el circuito a su estado de reposo cambiando nuevamente el valor lógico de LE a 1 y se indaga por un nuevo proceso de conversión. Si no se desea realizar un nuevo proceso de conversión se va al estado de finalización de tareas, de lo contrario se retoman las acciones desde el establecimiento de la palabra en los pines de $D_0$ a $D_7$

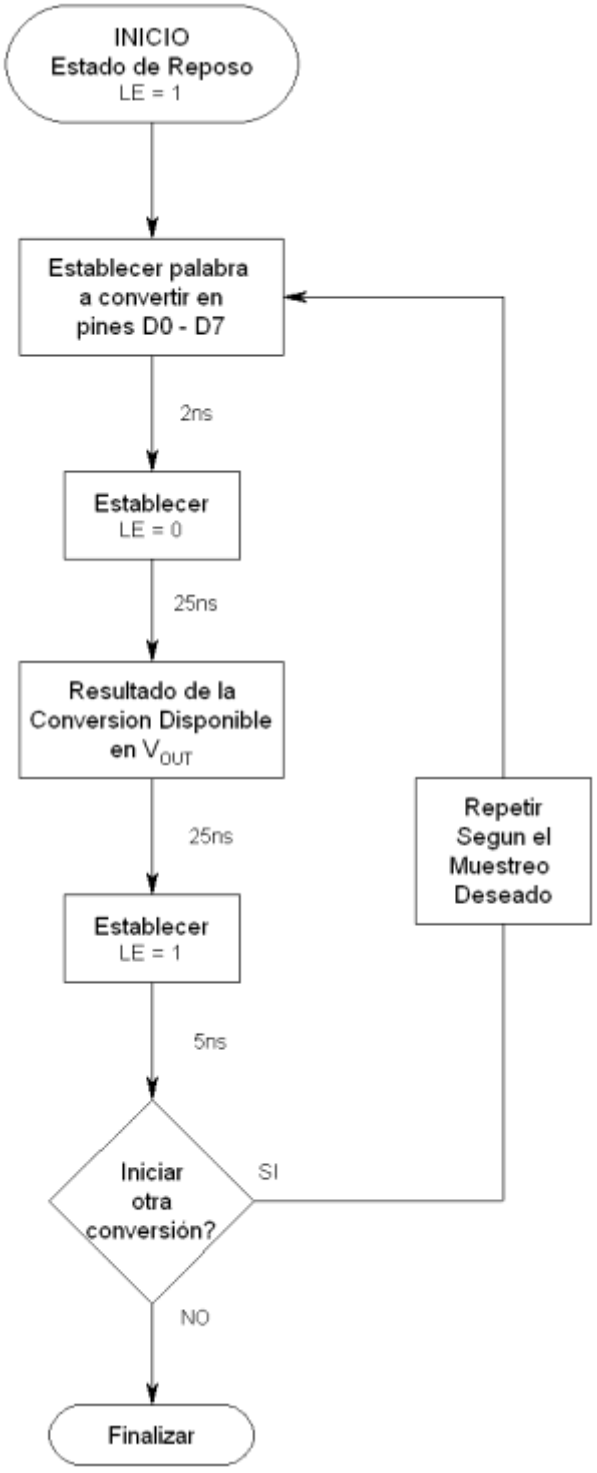



**Figura 4.2 Diagrama de Flujo del Conversor D/A**

## 4.4 MODULO DE PUERTO DIGITAL

El diseño del puerto digital es casi transparente puesto que la palabra se coloca a la entrada del búfer 74AHC244 e inmediatamente se obtiene la misma a la salida. En la parte software se debe tomar la decisión de leer una palabra o escribir una palabra en el puerto, de acuerdo con la opción elegida se da la orden a través del FPGA para seleccionar ya sea puertos de entrada o de salida y para enviar o recibir los datos a/o desde el PC. En caso de elegirse la opción de leer en el puerto se cambia el valor lógico de OE a 0 con lo cual se activan los buffers que protegen de sobrevoltajes al FPGA.

El diagrama de flujo de la figura 4.3 muestra el comportamiento del puerto digital anteriormente descrito

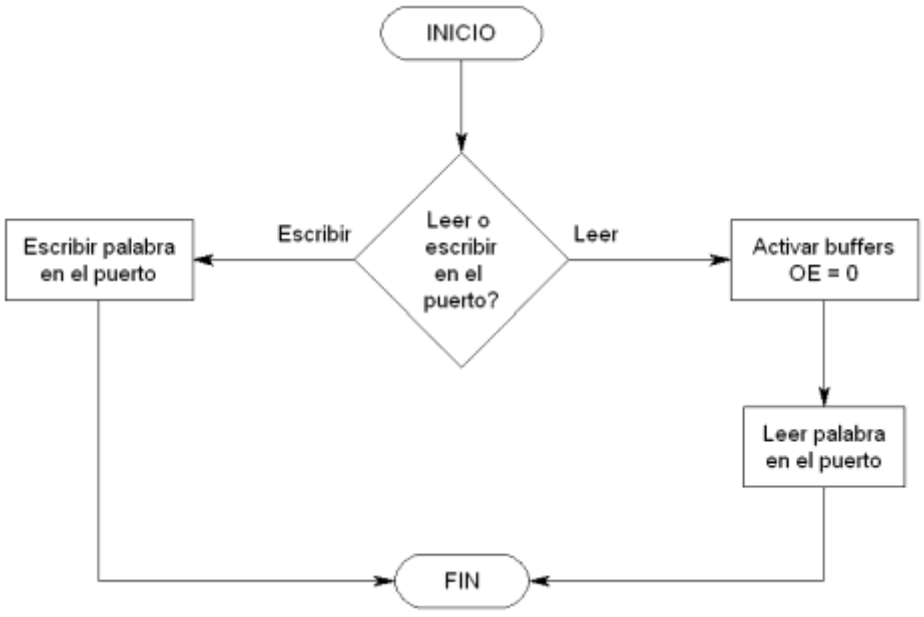



**Figura 4.3 Diagrama de Flujo del Puerto Digital**

El código del modulo esta contenido en la carpeta **software/componentes** con el nombre de pto_digital.vhd.

## 4.5 NIVEL SUPERIOR DE LA TAC

El nivel superior del software para la TAC es el encargado de controlar y gestionar los módulos desarrollados, agradándolos de la manera coherente de acuerdo a cada aplicación. El archivo producto de esas agrupaciones a sido denominado top_level.vhd, en él se encuentran contenidas todas las instrucciones necesarias para poder instanciar cualquiera de los tres módulos, ya sea el componente DAC, el ADC o los puertos digitales. Además se deja comentadas las líneas de código que se pueden usar para la instanciasión de cualquier otro modulo que se cree en el futuro.

El archivo principal que se desarrolla para la tarjeta tiene entonces la estructura de la figura 4.4

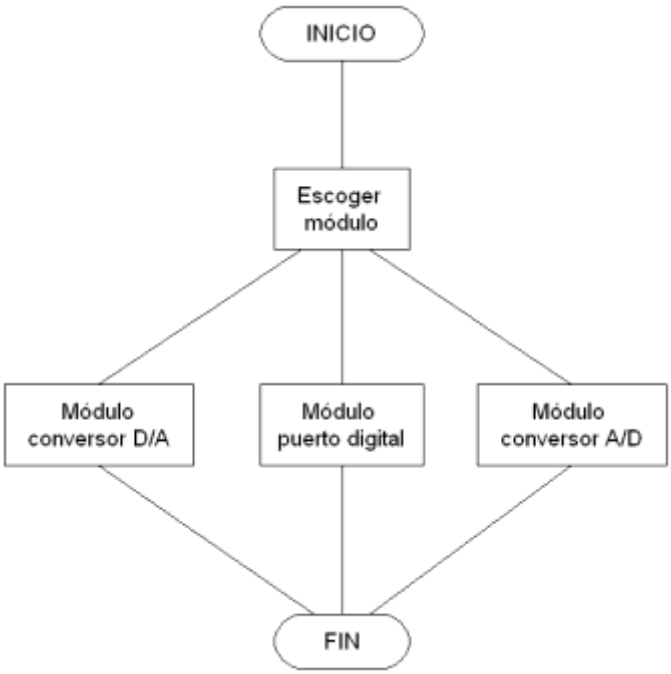



**Figura 4.4 Nivel Superior**

Dentro del archivo top_level.vhd se encuentran las siguientes partes que son de vital importancia y que constituyen el corazón de la TAC USB.

```
--*************************************************************
-- Puertos para el modulo de Conversión  Digital Analógico
--*************************************************************
                clk             : in std_logic;
                le_dac          : inout std_logic;
                salida_dac      : out std_logic _vector(3 downto 0);
```

Si no se desea el componente de conversión digital a analógica componente deben comentarse las líneas para que el modulo no sea tomado en cuenta en el momento de la compilación del código final que se bajará al FPGA y que será ejecutado por el mismo. Los comentarios en VHDL se consiguen anteponiendo dos guiones medios (- -) a cada línea. Debe recordarse la necesidad de comentar las variables que no se usarán en las líneas posteriores para evitar problemas de compilación.

Los componentes que se instancian en VHDL deben declararse con sus entradas y sus salidas y el tipo de dato que manejarán. En el programa principal se dejan instanciados el conversor AD el DA y los puertos digitales, si se necesitase agregar cualquier otro habría que hacerlo de la misma manera, siguiendo las normas de VHDL. En las siguientes líneas se muestra como debe agregarse el componente ***okWireIn*** que está diseñado para la recepción de datos desde el PC, el componente ***okWireOut*** se usan para el envío de datos hacia el PC. Los dos módulos se encargan de estructurar los datos en las tramas pertinentes del protocolo USB 2.0 y de realizar la gestión para la adecuada conexión con el host alojado en los computadores.

```
component okWireIn port (
                ti_clk          : in    std_logic;
                ti_control      : in    std_logic_vector(12 downto 0);
                ti_data         : in    std_logic_vector(15 downto 0);
                ep_addr         : in    std_logic_vector(7 downto 0);
                ep_dataout      : out   std_logic_vector(15 downto 0));
end component;
```

Para definir cada uno de los endpoints a usar es necesario descomentar la siguiente línea en el código principal.

```
-- signal epNwire : STD_LOGIC_VECTOR(15 downto 0):= x"0000";
```

Donde **N** corresponde a un entero positivo que va de 0 a 15, es decir que se permite instanciar un total de 16 endpoints.

Una vez se instancia un endpoint es necesario descomentar las siguientes líneas para tenerlo totalmente habilitado. El fragmento de código no es más que la conexión de las señales que manejará el módulo, en este caso el *okWireIn,* algo equivalente a cablear las entradas y salidas de un CI que hace parte de un gran diseño.

```
epN : okWireIn port map (
                ti_clk => ti_clk,
                ti_control => ti_control,
                ti_data => ti_data,
                ep_addr => x"N",
```

ep_dataout => ep**N**wire);

Nuevamente **N** corresponde a un entero positivo que va de 0 a 15.

Los pasos anteriores son los básicos para agregar cualquier nuevo módulo que sea desarrollado para la TAC. VHDL permite una gran facilidad para la creación de estructuras cada vez más grandes y complejas y para la adición de una gran cantidad de módulos a una estructura principal, tal y como una placa madre incorpora dentro de si una gran cantidad de circuitos diferentes para la conformación de un todo.

La figura 4.5 permite apreciar la forma en que se estructura un proyecto en el entorno de desarrollo ISE.

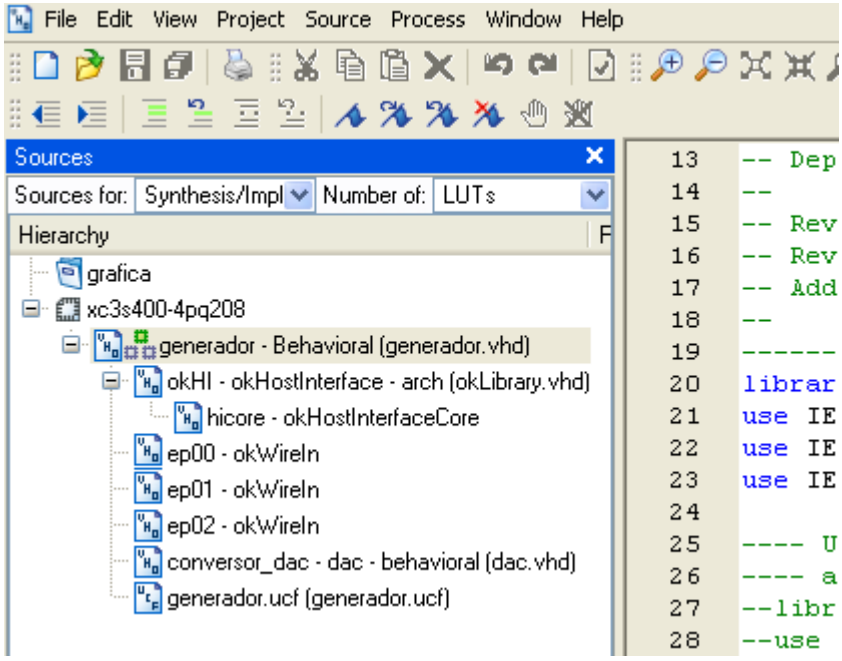



**Figura 4.5 Aplicación en ISE 8.2i**

La aplicación que se ha creado corresponde a un generador de onda haciendo uso del modulo conversor digital analógico. El modulo en VHDL generador.vhd corresponde al archivo de capa superior que en este caso a sido renombrado con el nombre de la aplicación especifica. La aplicación se encarga de generar varias frecuencias, de acuerdo con lo que el usuario seleccione desde el PC se usará una de esas frecuencias para determinar la velocidad de muestreo a la que trabajará el modulo DAC. También se sensa el tipo de onda que el usuario elije y de acuerdo con ello se producen los valores de muestreo desde el DAC para darle la forma a las señales.

Para llevar la aplicación a feliz termino, lo mismo que cualquier otra, se debe agregar una librería VHDL que permite el enlace con el host USB que se encuentra en el PC, dicha librería ha sido desarrollada por los proveedores del modulo XEM3001, quienes también entregan el componente okHostInterfaceCore.ngc el cual es instanciado y que debe estar presente en el proyecto desarrollado. La librería ha sido comentada y explicada en español para brindar una fácil interacción a la hora de desarrollar nuevas aplicaciones. Ella se encuentra en la carpeta **software/fuentes_vhdl** en el anexo C.

Los endpoints se instancian de la forma anteriormente descrita y cuando ello se ha realizado de la manera correcta aparecen tal como se ve en la anterior figura, de no ser así aparece un signo de interrogación en donde esta ep00, ep01 o ep02 y en general lo mismo sucederá para cualquier componente ausente del proyecto pero que haya sido instanciado.

Finalmente se agrega el componente DAC.vhd tal y como es entregado en las fuentes VHDL y con ello se tienen todos los módulos listos para integrar un proyecto completo.

Cuando se cuenta con el proyecto totalmente armado se sintetiza sin mayor inconveniente. Para la Implementación es necesario tener armado el archivo de distribución de pines de manera correcta, es decir que todos estén asignados y en los lugares indicados. Los archivos que se generan para este caso tienen la extensión UCF y son creados por la herramienta ISE al seleccionar la opción Assign Package Pins (Asignación de Pines). El archivo se puede mirar y editar como cualquier archivo de texto y puede apreciarse en la opción Edit Constraints. La figura 4.6 ilustra el lugar en donde aparecen esas opciones en la herramienta de desarrollo ISE 8.2i.

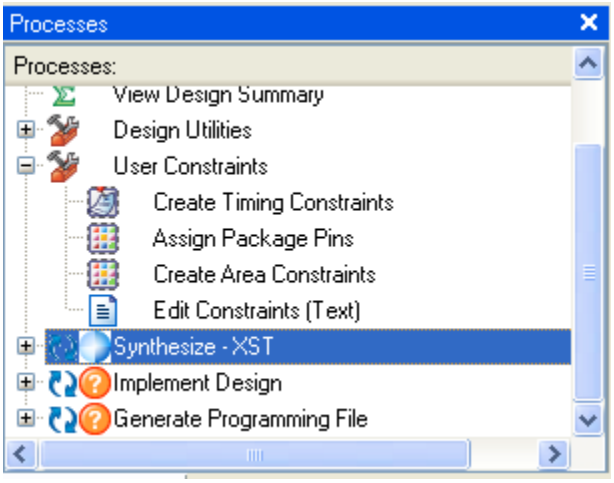



**Figura 4.6 Procesos de un Proyecto en ISE 8.1i**

Para la aplicación del generador de ondas el archivo (generador.ucf) lleva la información que se incluye a continuación a manera de ilustración del formato general con que deben contar

```
#------------
# Peripherals
#------------
#PACE: Start of Constraints generated by PACE

#PACE: Start of PACE I/O Pin Assignments
NET "clk"  LOC = "P80"  ; # reloj

NET "hi_addr<0>"  LOC = "P61"  ;
NET "hi_addr<1>"  LOC = "P62"  ;
NET "hi_addr<2>"  LOC = "P63"  ;
NET "hi_addr<3>"  LOC = "P64"  ;
```

```
NET "hi_busy"  LOC = "P81"  ;
NET "hi_clk"  LOC = "P79"  ;
NET "hi_cs"  LOC = "P57"  ;
NET "hi_data<0>"  LOC = "P67"  ;
NET "hi_data<10>"  LOC = "P95"  ;
NET "hi_data<11>"  LOC = "P96"  ;
NET "hi_data<12>"  LOC = "P97"  ;
NET "hi_data<13>"  LOC = "P100"  ;
NET "hi_data<14>"  LOC = "P101"  ;
NET "hi_data<15>"  LOC = "P102"  ;
NET "hi_data<1>"  LOC = "P68"  ;
NET "hi_data<2>"  LOC = "P72"  ;
NET "hi_data<3>"  LOC = "P74"  ;
NET "hi_data<4>"  LOC = "P86"  ;
NET "hi_data<5>"  LOC = "P87"  ;
NET "hi_data<6>"  LOC = "P90"  ;
NET "hi_data<7>"  LOC = "P92"  ;
NET "hi_data<8>"  LOC = "P93"  ;
NET "hi_data<9>"  LOC = "P94"  ;
NET "hi_irq"  LOC = "P85"  ;
NET "hi_rdwr"  LOC = "P58"  ;

NET "le_dac"  LOC = "p28"  ;
NET "salida_dac<0>"  LOC = "p26"  ;
NET "salida_dac<1>"  LOC = "p33"  ;
NET "salida_dac<2>"  LOC = "p29"  ;
NET "salida_dac<3>"  LOC = "p27"  ;
NET "salida_dac<4>"  LOC = "p24"  ;
NET "salida_dac<5>"  LOC = "p35"  ;
NET "salida_dac<6>"  LOC = "p34"  ;
NET "salida_dac<7>"  LOC = "p31"  ;

#PACE: Start of PACE Area Constraints

#PACE: Start of PACE Prohibit Constraints

#PACE: End of Constraints generated by PACE
```

Las asignaciones desde la **hi_addr** hasta **hi_rdwr** son esénciales ya que permiten el vinculo entre la TAC USB y el PC, para enviar y recibir datos y realizar a través de esas entradas y salidas todas las gestiones del protocolo USB 2.0. Las demás asignaciones de pines corresponden a la aplicación en particular. Dependiendo de cada una de ellas aquí se deberán agregar las entradas y las salidas y los respectivos pines en donde se desea tenerles siempre y cuando sean acordes con la tabla de asignaciones del Spartan-3 con empaquetado pq208, la cual se encuentra en las especificaciones del mismo.

El paso final es la implementación del proyecto, cuando esta acción se ejecuta se realizan todos los mapeos necesarios y se genera el archivo de tipo bitstream que es el que finalmente se descarga al FPGA. Con ello se culmina el desarrollo de cualquier aplicación o la creación de cualquier modulo nuevo.

Las interfaces de usuario de cada uno de los módulos así como de la aplicación están explicadas de manera detallada en el Anexo B. los respectivos códigos y los archivos fuente están contenidos en el CD de Archivos Digitales considerado el Anexo C.

# 5. PRUEBAS Y RESULTADOS

## 5.1. INTRODUCCIÓN

A continuación se hará referencia a las pruebas que se realizaron a los tres módulos hardware de la TAC empleando el software diseñado para controlarlos, el cual se describió en el capitulo 4.

## 5.2. PRUEBAS DEL MÓDULO CONVERSOR A/D

Las pruebas se realizaron aplicando en los 8 canales de entrada del módulo señales DC con valores aleatorios en el rango comprendido entre los 0 y 2V. Estas señales son producidas por un generador de señales con el propósito de que los resultados se obtuvieran con la mayor precisión posible.

Los valores de las señales analógicas de entrada se escogieron para que correspondieran teóricamente con el valor de la palabra digital que deberían representar. Esto se hizo con el propósito de poder comparar entre el valor teórico esperado para un valor de entrada analógico dado y el valor real obtenido. Estos valores se calcularon de la siguiente forma:

$$V_{entrada} = N^{o}\ paso * V_{RES}$$

En donde: $V_{RES} = \frac{V_{REF}}{2^{n}}$ (resolución en voltios)

Para el conversor: $V_{REF}$ = 2,5V (voltaje de referencia)

$n$ = 8 (resolución en bits)

Por lo que: $V_{RES}$ = 9,765mV

El Nº de paso es igual al valor decimal correspondiente a la palabra en binario.

En la tabla de 5.1 se muestran los resultados que se consiguieron.

| CANAL | NUMERO DE PASO | VOLTAJE DE ENTRADA (mV) | PALABRA TEORICA | PALABRA REAL |
|---|---|---|---|---|
| 1 | 0 | 0 | 00000000 | 00000010 |
| 1 | 25 | 244,125 | 00011001 | 00011000 |
| 2 | 50 | 488,25 | 00110010 | 00111000 |
| 2 | 75 | 732,375 | 01001011 | 01001110 |
| 3 | 100 | 976,5 | 01100100 | 01101001 |
| 3 | 125 | 1220,625 | 01111101 | 01111100 |
| 4 | 150 | 1464,75 | 10010110 | 10011011 |
| 5 | 175 | 1708,875 | 10101111 | 10110001 |
| 6 | 200 | 1953 | 11001000 | 11001011 |
| 7 | 225 | 2197,125 | 11100001 | 11100010 |
| 8 | 250 | 2441,25 | 11111010 | 11111101 |
| 8 | 255 | 2500 | 11111111 | 11111110 |

**Tabla 5.1 Resultados para el Módulo A/D**

La figura 5.1 muestra gráficamente los anteriores resultados.

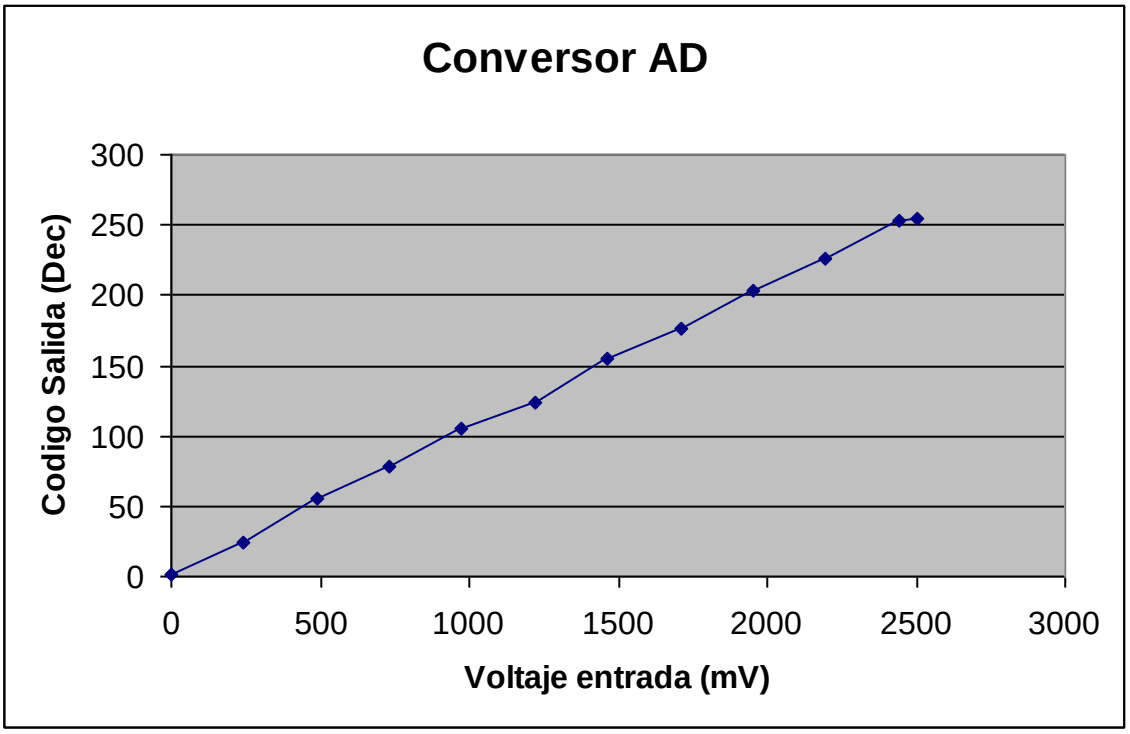



**Figura 5.1 Grafica de los Resultados Obtenidos para el Conversor AD**

## 5.3 PRUEBAS DEL MÓDULO CONVERSOR D/A

Las pruebas de este modulo consisten en aplicar palabras digitales de 8 bits en sus entradas y medir la señal analógica a la salida del mismo para confrontarlas con los valores teóricos esperados. Debido a que la resolución del conversor es de 8 bits significa que a la salida del módulo se pueden obtener 256 ($2^8$) valores de voltaje analógico distintos dentro del rango que va de 0V a 1V. Sin embargo para las pruebas se emplearon solamente 26 palabras digitales que variaran entre 00H y FFH en pasos de AH (10 decimal), con el fin de obtener una muestra representativa de valores.

Teóricamente el tamaño de paso de conversión en un conversor D/A esta dado por:

$$V_{paso} = \frac{1}{2^n} * V_{REF}$$

En donde: $n$ = resolución en bits

$V_{REF}$ = voltaje de referencia (voltaje analógico de salida máximo)

De modo que para el conversor:

$$V_{paso} = \frac{1}{2^8} * (1V) = 3{,}92mV$$

De acuerdo a esto el valor de voltaje de salida teórico para una palabra digital en particular esta dado por:

$$V_{salida} = V_{paso} * N^o\ paso$$

El número de paso es igual al valor decimal de la palabra digital de entrada.

En la tabla 5.2 se muestran los resultados obtenidos.

| PALABRA DIGITAL | NÚMERO DE PASO | VOLTAJE MEDIDO (mV) | VOLTAJE TEORICO (mV) |
|---|---|---|---|
| 00000000 | 0 | 33,8 | 0 |
| 00001010 | 10 | 77,6 | 39.2 |
| 00010100 | 20 | 117,6 | 78.4 |
| 00011110 | 30 | 158,2 | 117.6 |
| 00101000 | 40 | 198,4 | 156,8 |
| 00110010 | 50 | 239 | 196 |
| 00111100 | 60 | 279 | 235,2 |
| 01000110 | 70 | 320 | 274,4 |
| 01010000 | 80 | 359 | 313,6 |
| 01011010 | 90 | 400 | 352,8 |
| 01100100 | 100 | 440 | 392 |
| 01101110 | 110 | 480 | 431,2 |
| 01111000 | 120 | 519 | 470,4 |
| 10000010 | 130 | 559 | 509,6 |
| 10001100 | 140 | 599 | 548,8 |
| 10010110 | 150 | 639 | 588 |
| 10100000 | 160 | 679 | 627,2 |
| 10101010 | 170 | 718 | 666,4 |
| 10110100 | 180 | 758 | 705,6 |
| 10111110 | 190 | 797 | 744,8 |
| 11001000 | 200 | 837 | 784 |
| 11010010 | 210 | 876 | 823,2 |
| 11011100 | 220 | 916 | 862,4 |
| 11100110 | 230 | 956 | 901,6 |
| 11110000 | 240 | 996 | 940,8 |
| 11111010 | 250 | 1035 | 980 |
| 11111111 | 255 | 1074 | 1 |

**Tabla 5.2 Resultados del Modulo Conversor DA**

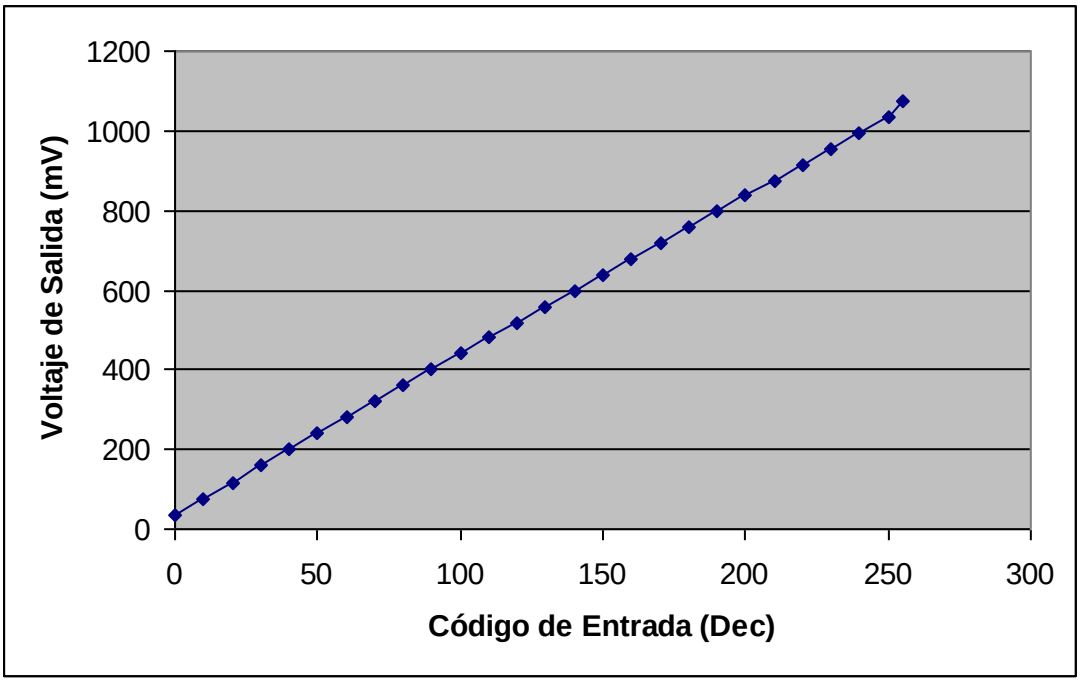



**Figura 5.2 Grafica de los Resultados Obtenidos para el Conversor DA**

De acuerdo a los resultados de la tabla 5.2 y la figura 5.1, obtenidos en la prueba, se puede concluir lo siguiente:

- El conversor posee un error de desplazamiento (offset), correspondiente a la diferencia entre el valor teórico y el medido cuando la entrada es OOH, de 33,8mV.

- El conversor posee un error de ganancia, correspondiente a la diferencia entre el valor teórico y el medido cuando la entrada es FFH, de 74mV.

- La linealidad del conversor es muy buena, por lo que los errores que pueda introducir por falta de ella son muy bajos.

## 5.4 PRUEBAS DEL MODULO DE PUERTO DIGITAL

Las pruebas del módulo de puertos digitales se realizaron en dos partes: prueba del puerto de salida y prueba del puerto de entrada.

### 5.4.1 Prueba del Puerto de Salida

Para la prueba del puerto de salida se conectaron a cada una de las 8 salidas de este 8 LEDs con resistencias de pull-up de 220Ω, respectivamente. La prueba consistió en observar la respuesta de los LEDs ante la presencia de una palabra digital escrita en la interfaz SW de control del modulo y constatar que coincidieran.

Los datos de la tabla 5.3 ilustran los resultados obtenidos, para algunos valores de palabra digital aleatorios.

| PALABRA DIGITAL ESCRITA | PALABRA DE SALIDA EN LOS LEDs |
|---|---|
| 00000000 | 00000000 |
| 00011001 | 00011001 |
| 00110010 | 00110010 |
| 01001011 | 01001011 |
| 01100100 | 01100100 |
| 01111101 | 01111101 |
| 10010110 | 10010110 |
| 10101111 | 10101111 |
| 11001000 | 11001000 |
| 11111111 | 11111111 |

**Tabla 5.3 Resultados para el Puerto de Salida**

Tal como se ve en la tabla 5.3, la respuesta obtenida en los LEDs es totalmente fiel a la palabra digital aplicada y no se induce ningún tipo de error. Este resultado es muy predecible, ya que se esta trabajando con niveles de voltaje digitales.

### 5.4.2 Prueba del Puerto de Entrada

Para la prueba del puerto de entrada se aplicaron en cada una de las 8 entradas niveles lógicos generados con un dip-switch de 8 bits. La prueba consistió en generar distintos valores de palabra digital con el dip-switch; leerlos a través de la interfaz de control SW del modulo y observar que coincidieran.

Los datos de la tabla 5.4 ilustran los resultados obtenidos, para algunos valores de palabra digital aleatorios generados.

| PALABRA DIGITAL GENERADA | PALABRA DIGITAL LEIDA |
|---|---|
| 00000000 | 00000000 |
| 00011001 | 00011001 |
| 00110010 | 00110010 |
| 01001011 | 01001011 |
| 01100100 | 01100100 |
| 01111101 | 01111101 |
| 10010110 | 10010110 |
| 10101111 | 10101111 |
| 11001000 | 11001000 |
| 11111111 | 11111111 |

**Tabla 5.4 Resultados para el Puerto de Entrada**

Tal como se ve en la tabla 5.4 la respuesta obtenida en la interfaz SW de control del modulo coincide exactamente con la palabra generada con el dip-switch. Este resultado demuestra que el modulo reconoce bien los niveles lógicos aplicados en sus entradas y el traductor de nivel realiza correctamente la conversión de voltajes lógicos.

# 6. CONCLUSIONES Y RECOMENDACIONES

- La tendencia natural de los dispositivos de adquisición de datos esta encaminada cada vez más a conseguir velocidades más elevadas, mayores tazas de muestreo en los conversores, mayor cantidad de entradas y salidas analógicas y digitales. La Tarjeta de adquisición USB desarrollada esta a la par con las tendencias mundiales y cumple con todas las características comunes de evolución.

- La modularidad permite brindar una mayor versatilidad en los dispositivos, gracias a ella es posible aumentar la potencialidad de herramientas fundamentales tales como los módulos conversores, los puertos digitales, los módulos de PWM, los generadores de señal, los generadores de onda, entre otros. El hecho de que muchos de esos módulos se puedan agregar en una sola tarjeta, con una mayor cantidad de variaciones software que hardware permite disminuir significativamente costo y aumentar un buena manera la eficiencia.

- El uso de dispositivos lógicos programables permite la creación de una gran cantidad de módulos que pueden adaptarse a una única tarjeta de adquisición de datos y control, con la facilidad que brinda la conexión de un módulo en una ranura y la descarga de su software de control en el dispositivo lógico programable. La TAC USB se adapta fácilmente a cualquier necesidad en el ambiente industrial y de control gracias a esa característica y además permite a los estudiantes la posibilidad de explorar todo su potencial de muchas formas diferentes ya que el estándar de

desarrollo VHDL para FPGAs no es más que un lenguaje que puede ser usado e interpretado de diversas maneras para conseguir un mismo fin.

- Los lenguajes de modelado hardware como VHDL disminuyen el nivel de complejidad estructural de grandes proyectos, gracias a ellos es posible convertir en líneas de código algo que de otro manera seria un gran conjunto de circuitos integrados y cables susceptibles en mayor cantidad de introducir errores o de causar problemas dado el nivel de complejidad física al que se puede llegar en un diseño. La TAC USB agrega una característica muy agradable en ese sentido ya que todo aquello que se desarrolle puede realizarse en su mayoría a nivel de descripción hardware, simulándose y compilándose cuantas veces sea necesario hasta conseguir el resultado deseado sin que por ello haya que modificar hardware e incurrir en nuevos costos.

- El protocolo USB 2.0 brinda la oportunidad de crear dispositivos con excelentes características tales como la velocidad, que puede ser hasta de 480 Mb/s, más que buena en el campo del control y la instrumentación industrial. La facilidad de conexión y desconexión en caliente son una muy deseable característica, ya que no hay necesidad de reiniciar el PC cada vez que se conecte el dispositivo y además el mismo se puede cambiar de lugar rápidamente dado que será un elemento periférico más que se le agrega al PC. Al desarrollarse una tarjeta que trabaja bajo el protocolo USB 2.0 se ha logrado seguir el flujo normal en cuanto a la tendencia de universalización que dicho protocolo tiene.

- Aunque se han desarrollado tres módulos muy útiles, es claro que la TAC USB tiene un gran potencial, por ello seria recomendable no dejarla caer en el olvido y contrario a ello fomentar la creación de nuevos módulos por parte de los estudiantes para anexarlos a los ya existentes y poder formar una gran colección de los más comunes y los más prácticos.

- El lenguaje VHDL es la herramienta principal con que contaran los estudiantes para el desarrollo de los módulos futuros, por ello es conveniente intensificar la enseñanza y manipulación de todo cuanto a ello se refiere, sobre todo teniendo en cuenta que el uso de FPGAs esta muy difundido en el mundo al igual que el de muchos otros dispositivos de lógica programable, y todos ellos representan potenciales campos de desempeño de los profesionales que salen de la facultad.

- En cuanto al protocolo USB 2.0 se refiere es más que claro que todas las aplicaciones y creaciones de dispositivos actuales tienden a utilizarlo gracias a su versatilidad. Desde de este punto de vista también se nota la necesidad de preparar a los estudiantes en su uso y de generar en ellos la motivación por explorarlo a fondo y tomarlo muy en cuenta para sus desarrollos.

## REFERENCIAS BIBLIOGRAFICAS

AXELSON, Jan. USB Complete: Everything You Need to Develop USB Peripherals (3rd Edición). Independent Publishers Group. Madison, WI, USA, 2005.

ANDERSON, Don. USB System Architecture (USB 2.0). ADDISON-WESLEY DEVELOPER'S PRESS. 2000

AREIBI, Shawki. ST.ONGE, Luck Michell.. A First Look at VHDL For Digital Design. School of Engineering, University of Guelph. Ontario. Canadá. 2003.

ASHENDEN, Peter J. The VHDL Cookbook. Dept. Computer Science University of Adelaida South Australia. Australia. 1990.

CORDOBA, G. Mario Andrés. Sistema de Control Fuzzy de Vuelo Basado en Dispositivos Empotrados DSPs - FPGAs. Monografía de trabajo de grado. Universidad del Cauca. Facultad de Ingeniería Electrónica y Telecomunicaciones. Colombia. 2001

HEURING, Vincent P., MURDOCCA, Miles J. Principles of Computer Architecture. Prentice Hall. 1999. Estados Unidos.

OLCOZ Serafín. TERÉS Lluis. TORROJA, Yago. VHDL Lenguaje estándar de diseño electrónico. MaGrawHill. 1998.

PEACOCK, Craig. USB in a Nutshell, Making Sense of the USB Standard. 2002

SPIEGEL, Jan Van der. VHDLTutorial. University of Pennsylvania Department of Electrical Engineering. Pennsylvania. Estados Unidos. 2003.

TOCCI, Ronald J. Sistemas Digitales Principios y Aplicaciones. Prentice Hall. Estados Unidos. 1995

USSELMANN, Rudolf. USB Function IP Core. Suiza. 2002

ANALOG DEVICES INC, AD7822/AD7825/AD7829 3 V/5 V, 2 MSPS, 8-Bit, 1-, 4-, 8-Channel Sampling ADCs. pdf Data Sheet. 2001

INTERSIL CORPORATION, CA3338 CMOS Video Speed, 8-Bit, 50 MSPS, R2R D/A Converters. pdf Data Sheet, 2004.

TEXAS INSTRUMENTS, SN74AHC244N OCTAL BUFFERS/DRIVERS WITH 3-STATE OUTPUTS. pdf Data Sheet, 2003.

ANALOG DEVICES INC, AD780 2.5 V/3.0 V High Precision Reference. pdf Data Sheet, 2004.

TEXAS INSTRUMENTS, SN74LS06 hex inverter buffers/drivers feature high-voltage open-collector outputs. pdf Data Sheet, 2004.

COMPAQ, HEWLETT-PACKARD, INTEL, LUCENT, MICROSOFT, NEC Y PHILIPS**,** Universal Serial Bus Specification. Revision 2.0. 2000.