Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales

Implementación de funciones básicas de osciloscopios en FPGA

Proyecto final para optar por el título de Ingeniero Electrónico con Orientación en Sistemas Digitales

Facundo Aguilera
Director: Carlos Federico Sosa Páez
Co-director: Diego Esteban Costa

2009

A la memoria de Pato.

Agradecimientos

Agradezco la colaboración de todos mis compañeros y amigos que me apoyaron en el transcurso del desarrollo de este trabajo, en especial a Andrés Airabella, Marcelo Stivanello, Matías Quinteros, Jorge Fé y Sergio Calderón. A mi director Carlos Sosa Páez y mi co-director Diego Costa por su constante ayuda y orientación.

También agradezco al resto de los integrantes del LEIS, por crear un ambiente de trabajo ameno, en especial a Esteban Pelaez, Roberto Kiessling y Martín Murdocca. Al ICTP, por proveer la placa de desarrollo, documentación y ayudas utilizadas en este trabajo.

Finalmente, agradezco a toda mi familia y muy especialmente a mi novia Elisa por su compañía y amor durante esta etapa de mi vida.

Índice

Índice de capítulos

Índice de figuras

Índice de tablas

Resumen

En este trabajo se desarrolla un osciloscopio digital donde el control principal y el procesamiento de la señal de entrada están implementados en una FPGA. Un software para computadora personal provee una interfaz gráfica que emula el panel y la pantalla del osciloscopio. La comunicación entre la computadora y la FPGA se realiza a través de un puerto paralelo estándar. Se realiza un diseño basado en bloques para facilitar la reutilización de recursos.

El diseño está pensado para implementarse en la placa de desarrollo *RVI Prototype Board* provista por el ICTP (International Center For Theoretical Physics).

1. Introducción

1.1. Surgimiento del proyecto

El presente trabajo está enmarcado dentro del proyecto de investigación *Instrumentación virtual reconfigurable con FPGA* de la Facultad de Ciencias Físico Matemáticas y Naturales de la Universidad Nacional de San Luis, que trabaja en colaboración con el proyecto *RVI* (Reconfigurable Virtual Instrumentation) del *MLab* (Multidisciplinary Laboratory) perteneciente al ICTP (Iternational Center For Theoretical Physics).

El proyecto RVI propone el desarrollo de una plataforma de instrumentación virtual de código abierto utilizando FPGA (Field Programmable Gate Array), basada en una metodología de diseño en bloques. El uso de este método promueve la reutilización de módulos y librerías como una forma efectiva de enfrentar la creciente complejidad de los diseños. La publicación bajo licencias de código abierto permite a nuevos contribuyentes e investigadores basarse en las fuentes previamente desarrolladas, facilitando la incorporación de nuevos instrumentos o mejoras a la plataforma [1].

La plataforma de instrumentación virtual propuesta por RVI consiste en la disponibilidad de todo un conjunto de instrumentos utilizables a través de un software para PC e implementados en una placa de desarrollo. Dentro de este proyecto, el presente trabajo se focaliza en el desarrollo de un osciloscopio virtual con prestaciones básicas, como punto de partida para futuros diseños sobre este instrumento o para ser incorporado en conjunto con otros instrumentos en la plataforma.

El desarrollo final es un osciloscopio virtual simple pero muy útil para la mayoría de las aplicaciones de no muy alta frecuencia, con una interfaz gráfica para PC emulando los controles del instrumento y hardware implementado en una FPGA. El diseño está adaptado para utilizar los recursos disponibles en la placa de desarrollo *RVI Prototype Board* provista por el ICTP.

1.2. Objetivos

En los siguientes puntos se listan los objetivos generales y específicos del presente trabajo.

1.2.1. Objetivos generales

- Aplicar conocimientos adquiridos durante la carrera de grado en un desarrollo concreto.
- Adquirir nuevas habilidades para el desarrollo de dispositivos digitales.
- Adquirir nuevos conocimientos sobre instrumentación virtual y osciloscopios digitales.
- Aportar nuevos desarrollos en el campo de la investigación en instrumentación virtual implementada en FPGA.

1.2.2. Objetivos específicos

- Diseñar un osciloscopio digital con prestaciones básicas utilizando la placa de desarrollo con FPGA *RVI Prototype Board* junto con la placa de expansión *LP Data*

Conversion Daughter Board provistas por el ICTP, aprovechando los recursos que ésta posee.

- Diseñar un software para computadora personal que brinde una interfaz gráfica con un aspecto visual similar al de los osciloscopios tradicionales.
- Implementar una interfaz de comunicación estándar entre la computadora personal y la FPGA.
- El diseño final deberá poseer las características comunes de un osciloscopio digital.

1.3. Características que debe poseer el desarrollo

Para la realización de este trabajo, se pretende que el diseño final posea las siguiente características:

- Diseño modular, de forma que se permita la fácil modificación e incorporación de nuevas funciones al hardware o al software. Módulos de hardware compatibles con algún sistema de interconexión para sistemas embebidos estándar. Debe facilitarse la reutilización de los recursos.
- El software para PC debe incluir una interfaz gráfica con los elementos de control del osciloscopio y la representación de la señal, y debe permitir alguna forma de almacenamiento de los datos.
- Diseño intuitivo y de fácil utilización, tanto para el software como para el hardware.
- Adaptabilidad a cambios de hardware o software.
- Precisión y velocidad de procesamiento en la FPGA.
- Buen aprovechamiento de los recursos disponibles en la placa de desarrollo *RVI Prototype Board*.
- Funciones básicas que debe poseer el osciloscopio final: dos canales de entrada seleccionables y configurables en forma independiente, diferentes métodos de disparo, cambios de resoluciones vertical y horizontal, controles de posición de la imagen.

2. Estado del arte

En este capítulo se realiza un estudio sobre la actualidad en las tecnologías de instrumentación virtual en general y luego, más específicamente, sobre los osciloscopios. Se realiza un análisis sobre las principales características de los osciloscopios digitales disponibles en el mercado. También se hace una comparación entre los objetivos buscados por otros diseños y los del presente trabajo.

2.1. Instrumentación virtual

El concepto de instrumentación virtual comienza a aparecer cuando las computadoras empiezan a incorporar más capacidades computacionales, permitiéndoles realizar tareas en tiempo real que antes solo eran posibles en un equipo especializado.

A diferencia de los instrumentos tradicionales, los cuales están contenidos completamente en su propia "caja", un instrumento virtual está compuesto por algunas sub-unidades de hardware, una computadora de propósito general y algún software. Los instrumentos virtuales pueden ser muy simples, como por ejemplo un generador de ondas o un multímetro, o ser muy complejos, como una red de instrumentos de medición y control o un paquete amplio de instrumentos incluidos en un mismo software. Mediante el aprovechamiento de los recursos disponibles en las actuales computadoras, se reduce el material adicional necesario para la implementación del instrumento, permitiendo disponer de ciertos equipamientos a un costo reducido. En la Figura 2.1 se muestra un diagrama típico de un instrumento virtual [2].

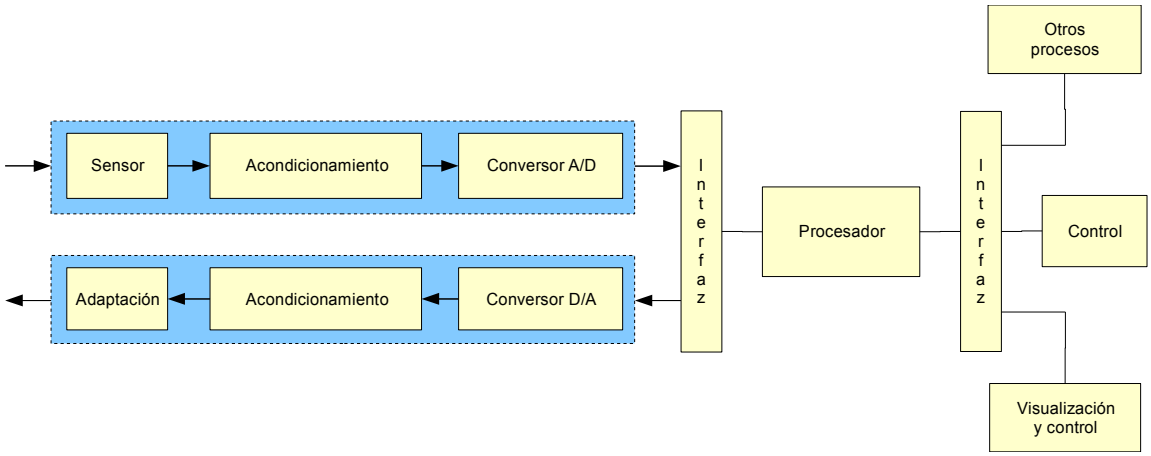

Figura 2.1: Elementos típicos de un instrumento virtual.

2.1.1. Actualidad

Hay una gran variedad de diseños en instrumentación virtual, tanto de marcas comerciales como desarrollados por instituciones, grupos de investigación y aficionados. Los productos disponibles en el mercado son muy completos, pero no permiten incorporar nuevas prestaciones debido a que sus diseños, en la mayoría de los casos, son cerrados. También debe considerarse que los costos de estos productos suelen ser muy elevados.

Por otro lado, los diseños no comerciales suelen utilizar una arquitectura de hardware

fija y poseen un diseño específico para los instrumentos implementados. En la mayoría de los casos, el procesamiento principal está centralizado en el software, limitándose el hardware al manejo de conversores de señales [3].

2.1.2. Hardware reconfigurable y diseño basado en bloques

Gracias a la tecnología de los PLD (Programmable Logic Device), es posible pensar en un **instrumento reconfigurable**. Se entiende por esto un versátil dispositivo de hardware capaz de ser reconfigurado en diferentes instrumentos electrónicos a través del uso de una herramienta de software. Este panorama se diferencia mucho de las arquitecturas de hardware más rígidas, ya que permite una total reimplementación de la lógica del instrumento, permitiendo brindar tanto a un usuario como a un desarrollador una mayor versatilidad, permitiendo obtener flexibilidad no solo en el software sino también en el hardware.

Existen abordajes anteriores en el campo de la instrumentación virtual, pero a diferencia del proyecto RVI, éstos han puesto poco énfasis en la construcción de bloques de FPGA reutilizables, los que facilitarían el desarrollo de nuevos sistemas o la mejora de los existentes [4] [5] [6] [7].

El sistema de instrumentación virtual reconfigurable con un diseño basado en bloques propone un dispositivo conectado a una PC a través de algún puerto (USB, Ethernet, paralelo, etc.). Una aplicación de software de alto nivel provee una interfaz de usuario con la que se puede seleccionar un instrumento virtual desde una lista de instrumentos y configurar el sistema para convertirse en él. La arquitectura modular permitiría la actualización de partes del software o del hardware en forma independiente [1].

El osciloscopio desarrollado en este trabajo está diseñado siguiendo la arquitectura propuesta por el proyecto RVI.

2.2. Osciloscopios

Los osciloscopios han sido desde hace muchos años instrumentos indispensables en el campo de la electrónica y la electricidad, ya sea para diseñar o para realizar reparaciones o detección de fallas. Su versatilidad ha permitido su utilización en otras disciplinas, como la medicina o la física.

La función principal de los osciloscopios es la representación de alguna señal eléctrica en forma gráfica. Para representar otros tipos de señales, se utilizan transductores que convierten las señales originales en una señal eléctrica.

En la mayoría de las aplicaciones se representa la señal en función del tiempo, aunque algunos osciloscopios pueden incorporar varias funciones adicionales [8] [9]. En la Figura 2.2 se muestra un osciloscopio digital típico.

2.2.1. Características de los osciloscopios disponibles en el mercado

Se puede realizar una principal división entre tipos de osciloscopios: los **analógicos** y los **digitales**. Aunque los osciloscopios analógicos son ampliamente usados, los digitales son más modernos y superan en la mayoría de los casos las prestaciones de los analógicos. El estudio de los osciloscopios analógicos está fuera del alcance de este informe.

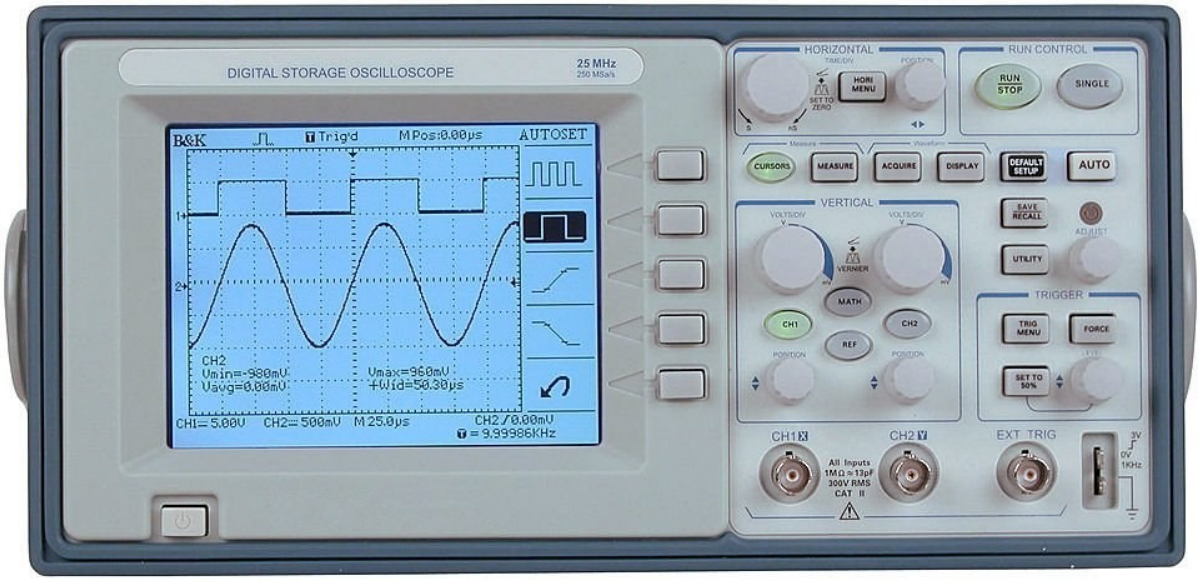

Figura 2.2: Osciloscopio digital de prestaciones típicas.

Existe una variedad muy grande de osciloscopios digitales, adaptados a diferentes aplicaciones y necesidades. A continuación se listan algunas de las especificaciones comparativas que pueden existir entre diferentes modelos, así como sus valores típicos. La lista fue obtenida principalmente a partir de los catálogos de las marcas de osciloscopios: BITSCOPE, ETC, Tektronix, PicoScope.

Ancho de banda: Esta es una de las principales características. El ancho de banda máximo de la señal de entrada que puede ser representado por un osciloscopio es muy variado, y puede ir desde los 10 MHz, cuando están diseñados para aplicaciones en educación o son de alta resolución vertical, hasta los 4 GHz, cuando son para mediciones más específicas en altas frecuencias. Los valores más comunes rondan entre los 20 y los 40 MHz.

Frecuencia de muestreo: Íntimamente relacionada con el ancho de banda, sus valores normalmente varían entre los 20 MS/s (millones de muestras por segundo) y los 40 GS/s, éstos últimos utilizados en osciloscopios de muy altas prestaciones para aplicaciones especiales. Los valores más comunes están entre los 80 y los 200 MS/s.

Canales: La cantidad de canales varía entre 1 y 16 canales. La mayoría de los modelos poseen dos canales, con los que pueden cubrirse la mayoría de las aplicaciones.

Resolución: La resolución vertical del conversor analógico-digital depende de las aplicaciones y de las velocidades para las que están diseñados los osciloscopios. El menor valor y mayormente utilizado es de 8 bits, pero existen de hasta 16 bits. Varios modelos permiten aumentar aún más la cantidad efectiva de bits a expensas de pérdidas en el nivel de detalle en la escala temporal.

Tamaño de memoria: La memoria almacena las muestras tomadas. En algunos equipos la memoria está compartida entre varios canales y en otros existe una independiente por canal. En todos los casos la cantidad de memoria utilizada en la adquisición es configurable. La cantidad de memoria disponible puede llegar hasta las 400 MS (millones de muestras) en sistemas de muy altas prestaciones. Los valores

comunes están entre los 2,5 kS y los 32 kS (miles de muestras). Otros equipos pueden utilizar la memoria de una PC para almacenar los datos, pero trabajan a bajas velocidades cuando funcionan en este modo.

Visualización y controles: Los osciloscopio tradicionalmente poseen una pantalla incorporada en la que se grafican las señales. Estas pantallas suelen ser LCD o CRT. Por otro lado, actualmente existe una gran línea de modelos que no incorporan la pantalla ni los controles, sino que la visualización y la configuración se realizan a través de una computadora personal. Podría decirse que estos últimos equipos siguen el concepto de instrumentación virtual para reducir el costo de los componentes. Para la comunicación con la PC normalmente se utiliza un puerto USB o Ethernet. También existen versiones que se conectan en forma interna a través del bus PCI.

Funciones incorporadas y características adicionales: Las funciones típicas suelen ser: diferentes formas de disparo (por nivel en una canal, externa, por señales de video, por flanco en una señal binaria, automático, con selección de sensibilidad temporal), diferentes combinaciones entre canales (independientes, superposición, suma, resta, etc.), representación en el dominio de la frecuencia. Otras características adicionales suelen estar incorporadas en equipos para propósitos más específicos, como por ejemplo, aislación galvánica entre las puntas y los conversores.

2.2.2. Publicaciones y diseños no comerciales

Existen muchas publicaciones y proyectos relacionados con la construcción de osciloscopios digitales. Varios tienen sus esquemas electrónicos, fuentes de software y demás recursos publicados en forma libre a través de Internet, de manera que pueden utilizarse como referencia para nuevos diseños. En el caso de los osciloscopios virtuales, las interfaces de comunicación con la PC utilizan los puertos USB, paralelo o serie. Existen proyectos cuya adquisición está controlada por un microcontrolador y otros en donde utilizan una FPGA para implementar el instrumento. Algunos diseñan su propio hardware (esquemas y PCB) y otros están pensados para ser implementados en alguna placa de desarrollo disponible comercialmente [10] [11] [12] [13] [14] [15].

A diferencia de otras iniciativas, el presente trabajo tiene la intención de crear un paquete de librerías de códigos de software y de módulos de hardware para FPGAs para la creación de un osciloscopio, con el objetivo de ser incluidos en una plataforma de desarrollo en la que se facilite la reutilización de estos recursos.

3. Herramientas de diseño

En este capítulo se analizan los recursos disponibles y se establecen ciertas consideraciones de diseño a partir de los objetivos del trabajo. Con esta información y las recomendaciones de la bibliografía, se realiza una selección de herramientas y métodos utilizados para realizar el diseño.

3.1. Diseño del hardware

Para la realización del trabajo se utiliza la placa de desarrollo *RVI Prototype Board* junto con la placa de expansión *LP Data Conversion Daughter Board*. Estas placas ya poseen todos los componentes necesarios para implementar el osciloscopio, por lo que el diseño del hardware se limita a la descripción del comportamiento de la FPGA en base a los dispositivos disponibles y a los objetivos del trabajo.

3.1.1. Placa de desarrollo

Las placas de desarrollo utilizadas constan de los siguientes componentes:

- **RVI Prototype Board (principal)**
  - FPGA ProASIC3E (A3PE1500)
  - Puertos de comunicación:
    - Puerto paralelo
    - RS232
    - USB
    - Ethernet 10/100
  - Conectores para incorporar placas de expansión
  - Extensión para memoria SDRAM
  - Indicadores y pulsadores para control y depuración
  - Circuito de configuración
- **LP Data Conversion Daughter Board (expansión)**
  - Conversor A/D (AD9201), conectores y acondicionamiento de señal
  - Conversor D/A (LTC1654), conectores y acondicionamiento de señal
  - Sensores de temperatura
  - Conectores para incorporar a la placa principal

En la Figura 3.1 se muestra la arquitectura de la placa de desarrollo, mostrando la disposición de los bloques listados [16]. En el anexo *Esquemas de RVI Prototype Board* pueden consultarse los esquemas de la placa.

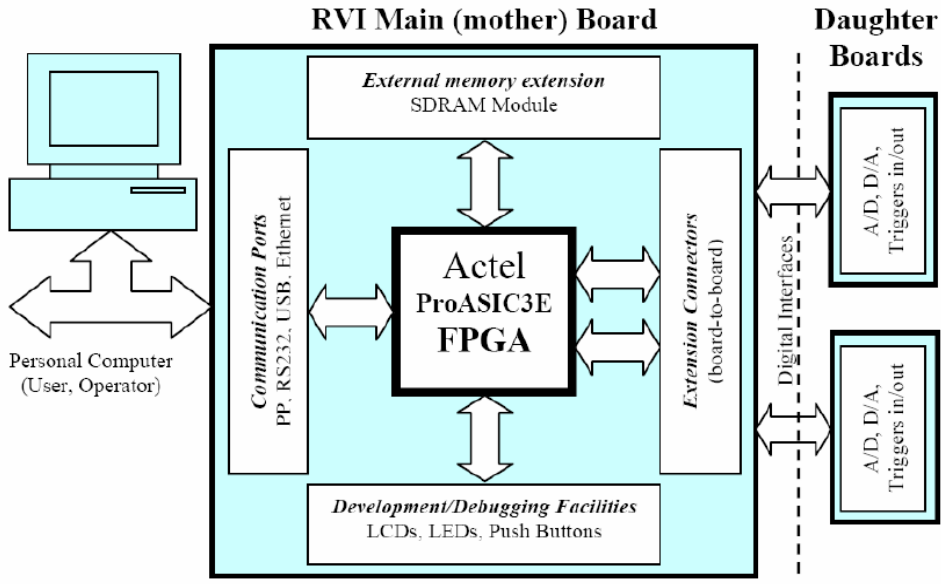

Figura 3.1: Arquitectura de la placa

3.1.2. Lenguaje de descripción de hardware

Se debe escoger un lenguaje de descripción de hardware y una herramienta de síntesis para realizar la implementación del diseño en la FPGA. Los lenguajes estándares más utilizados en la actualidad para realizar descripción lógica de hardware son VHDL y Verilog. La selección de un lenguaje u otro no depende de razones técnicas, debido a que siempre es posible obtener de alguna manera los mismos resultados con ambos, sino más bien de preferencias personales y disponibilidad de herramientas de síntesis y simulación [17]. Para el presente trabajo se ha usado VHDL por mayor familiaridad con su estructura.

3.1.3. Herramientas de software

El diseño en FPGA requiere de herramientas de software para realizar diferentes procesos a partir del código fuente. Las tareas principales que debe realizar el software son síntesis, posicionamiento, ruteo y simulación y existen diversas herramientas para realizar estos procesos [18]. Muchos fabricantes de FPGAs distribuyen algún software optimizado para sus productos, y normalmente son una buena opción a la hora de realizar diseños para una marca específica.

La placa de desarrollo posee una FPGA de Actel de la línea ProASIC3E. Esta empresa provee un completo paquete de herramientas de software para diseños en sus FPGA, con las que pueden realizarse tanto la síntesis, posicionamiento y ruteo como simulaciones del diseño en cada una de las etapas. Existe una versión con licencia gratuita de este conjunto de herramientas que puede descargarse desde su sitio web, y es la que se ha utilizado en este trabajo [19].

3.1.4. Modularidad e interfaz de comunicación entre módulos

Uno de los objetivos más importantes de este trabajo consiste en la obtención de un

diseño que pueda ser dividido en diferentes bloques (IP cores). Cada uno de estos bloques puede estar compuesto por sub-módulos, estableciéndose un esquema jerárquico. Siempre que se planean diseños a gran escala, se trabaja con este tipo de arquitectura. Entre sus ventajas se encuentran la facilidad de la comprensión del sistema en general y de cada una de sus etapas e interconexiones y la posibilidad de reutilización de componentes.

Para mejorar la portabilidad y reutilización de las partes del sistema se debe utilizar una interfaz estándar para la interconexión de los módulos principales. La adopción de un estándar asegura al usuario de alguno de los bloques la existencia de una guía en la cual basarse para realizar la integración del módulo en algún sistema. También facilita la incorporación de nuevas funcionalidades a un sistema específico.

Existen varias especificaciones de interconexión entre IP cores. Entre las más populares se encuentran CoreConnect de IBM, AMBA de ARM y WISHBONE, que fue diseñada por Siliscore y luego liberada al dominio público y mantenida por la organización OpenCores. Todas ellas persiguen el mismo objetivo: la de establecer un modo común de interconexión entre bloques de un sistema. En general, son todas muy similares y las principales diferencias tienen que ver con el conjunto de características provista y la rigidez o la flexibilidad de la especificación [20].

Los bloques principales de ese trabajo utilizan la arquitectura de interconexión propuesta por la especificación **WISHBONE** (Rev. B.3), por su simpleza y por considerarse la mayor promotora de la filosofía de código abierto en el campo de la lógica programable. Para un mayor detalle sobre las características de WISHBONE, puede consultarse el anexo *Descripción de la interfaz WISHBONE*.

3.1.5. Interfaz de comunicación con la PC

Para el diseño del instrumento virtual, debe seleccionarse algún puerto de la PC para comunicarse con el resto del hardware. En este caso, se selecciona uno de los puertos de comunicación disponibles en la placa de desarrollo.

Se optó por la utilización del puerto paralelo, utilizando el estándar **IEEE 1284** [21]. La ventaja de este modo es su simplicidad y practicidad en la implementación. La desventaja es que actualmente está cayendo en desuso, siendo reemplazado por otros puertos de comunicación más eficientes, aunque mucho más complejos.

El estándar 1284 del IEEE establece los siguientes modos de comunicación:

- Modo tradicional, compatible con Centronics. Solo salida
- Modo Nibble. Se utilizan la lineas de estado para ingreso de datos. Solo entrada.
- Modo Byte. Ingresan 8 bits usando el puerto de datos bidireccional. Solo entrada.
- EPP (Enhanced Parallel Port). Bidireccional.
- ECP (Extended Capability Port). Bidireccional

Según el estándar, no es necesario que estén soportados todos los modos de comunicación en el periférico, por ello se estable un protocolo de negociación de modo. Durante la negociación, el host (PC) consulta si determinado modo es soportado por el periférico. El modo básico, desde el que se comienza la negociación, es el modo tradicional y es el único que debe estar implementado de alguna manera en forma obligatoria (al menos para iniciar la negociación).

Para este trabajo se requiere de un puerto bidireccional que sea apropiado para realizar un puente entre la interfaz de comunicación del puerto paralelo y el bus interno utilizado en la FPGA. El modo de comunicación EPP permite implementar este puente de manera correcta y eficiente [22], y es el único modo utilizado en el periférico, junto con la correspondiente negociación recomendada por el estándar.

Utilizando el modo EPP se pueden realizan escrituras y lecturas de datos y de direcciones con un ancho de 8 bits a través del puerto paralelo. Las señales de handshake en la PC están implementadas por hardware sin hacer uso del procesador, facilitando el diseño del software y permitiendo realizar transferencias a altas velocidades.

3.2. Diseño del software

El software debe incluir una interfaz gráfica y algún mecanismo de comunicación con la placa, en esta caso, a través del puerto paralelo. Se pretende también que, así como la lógica del hardware, posea un diseño modular.

3.2.1. Diseño de la interfaz de usuario

Para el diseño de una interfaz gráfica, normalmente se utiliza un conjunto de librerías (o biblioteca) que facilitan la comunicación con el sistema operativo para la creación de los elementos gráficos. Estas bibliotecas normalmente se utilizan a través de algún lenguaje de programación específico, aunque a veces hay disponibles *bindings* a través de los cuales pueden utilizarse otros lenguajes. Muchas empresas, organizaciones y agrupaciones se dedican al mantenimiento de este tipo de bibliotecas, distribuyéndolas bajo términos de licencias comerciales o públicas. También suelen utilizarse entornos de desarrollo integradas o IDEs (Integrated Development Environment), los cuales agrupan muchas de las herramientas usadas por los desarrolladores de software, tales como compilador, depurado, editor de texto, diseñador gráfico de interfaces, ayudas, etc, en un entorno gráfico de usuario intuitivo.

Para el desarrollo del software en este trabajo se ha optado por la utilización del paquete de herramientas **Free Qt SDK**, que incluye, entre otras cosas, la biblioteca **Qt** y el IDE **Qt Creator**, que actualmente pertenecen a la Corporación Nokia y son distribuidos bajo los términos de la licencia GNU LGPL (GNU Lesser General Public License), considerándose software libre y de código abierto.

Qt es una biblioteca multiplataforma para el desarrollo de aplicaciones de software, con herramientas muy prácticas y eficientes para el diseño de interfaces gráficas. Se ha utilizado el lenguaje de programación C++, que es el lenguaje que usa Qt en forma nativa [24]. Una gran ventaja de Qt es que posee una buena documentación de sus bibliotecas y una gran cantidad de grupos de usuarios o foros de desarrolladores que pueden facilitar las tareas de diseño.

También se ha utilizado la biblioteca **Qwt**, que contiene un conjunto de componentes y utilidades para el diseño de interfaces gráficas usando Qt, pensados principalmente para aplicaciones técnicas. Es, básicamente, una extensión de la biblioteca Qt, y se utiliza en este trabajo por las herramientas que provee para la realización de gráficos en 2D. Qwt también es distribuido bajo los términos de la licencia Qwt, que está basada en la GNU LGPL, pero es menos restrictiva [25].

La idea de utilizar este tipo de herramientas es basar la arquitectura del software en herramientas de desarrollo de software libres, a diferencia de las interfaces diseñadas

por el proyecto RVI que han estado basadas en herramientas no libres, poco flexibles y de costos elevados como NI LabVIEW.

3.2.2. Sistema operativo

Si bien la biblioteca Qt es multiplataforma, es decir, existen versiones de la biblioteca disponibles para varios sistemas operativos (más específicamente Linux, Windows y Mac OS X) por lo que puede compilarse el mismo código fuente basado en Qt en cualquiera de estos sistemas, el software solo será probado en el sistema operativo Windows. Futuros trabajos podrán hacer las adaptaciones correspondientes para portar el software a otros sistemas.

Se ha elegido el sistema operativo Windows debido a que es el más utilizado en las carreras de electrónica de la Universidad, pensando en las posibles aplicaciones didácticas que pueda tener el osciloscopio.

4. Descripción de los módulos de hardware

En este capítulo se hace una descripción de la lógica implementada en la FPGA para el desarrollo del osciloscopio. Se realiza un desglose de los módulos componentes y una descripción de cada uno y de sus interconexiones.

Para facilitar su interpretación, los bloques de los esquemas presentados en este capítulo poseen colores para diferenciar el nivel jerárquico al que pertenecen, tal como se indica en la Figura 4.1.

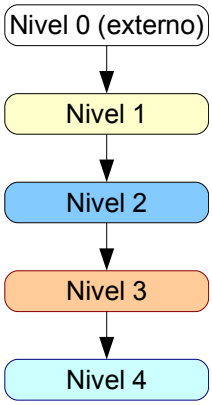

Figura 4.1: Colores de los niveles jerárquicos.

4.1. Descripción general

En esta sección se describirá el diseño en los niveles superiores para dar un panorama general de su funcionamiento. Se irán recorriendo progresivamente los diferentes niveles jerárquicos del sistema completo a lo largo de las siguientes secciones.

El sistema completo está integrado por la placa de desarrollo y una computadora de propósito general que tenga instalado el software desarrollado. En la Figura 4.2 se muestra una representación de los componentes del osciloscopio.

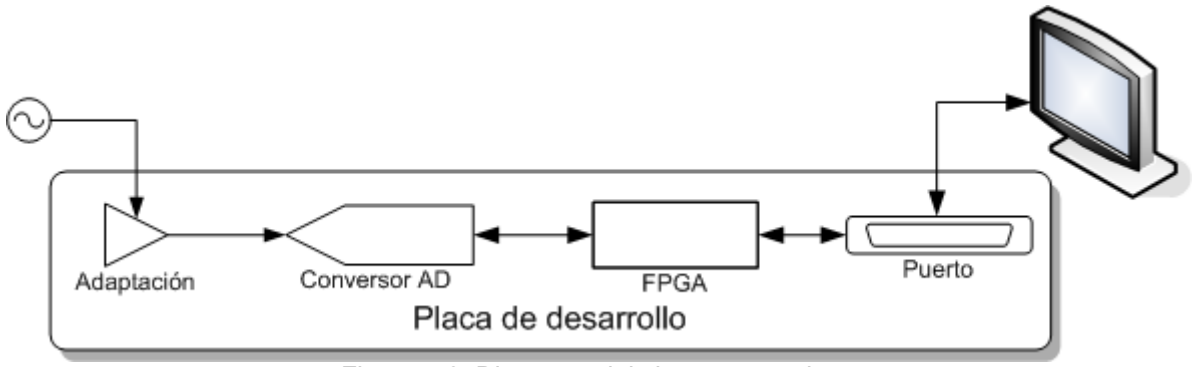

Figura 4.2: Diagrama del sistema completo.

4.1.1. Comportamiento a nivel externo

La placa *RVI Prototype Board* junto con la placa de expansión *LP Data Conversion Daughter Board* poseen el conversor analógico digital AD9201 de Analog Devices, con capacidad de realizar la conversión de dos señales en forma simultánea, y un conector para el puerto paralelo. También poseen conectores para anexar puntas para

osciloscopios y algunos componentes para realizar el acondicionamiento de las señales. A partir del esquema completo del hardware, puede obtenerse un diagrama simplificado de los componentes principales de la placa utilizados en este trabajo, como el mostrado en la Figura 4.3.

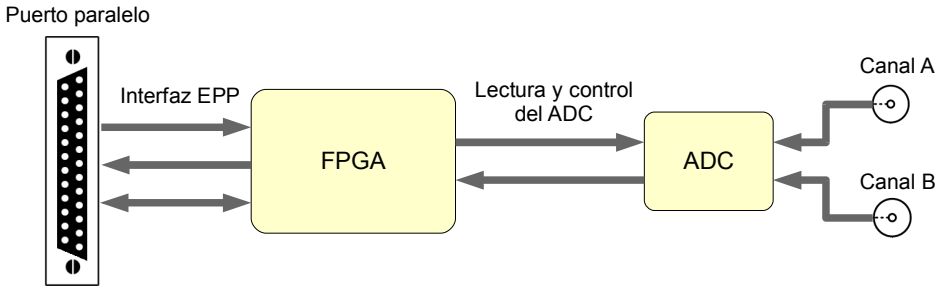

Figura 4.3: Esquema simplificado externo.

El conversor recibe una señal de reloj desde la FPGA. Realiza una conversión simultánea de los valores analógicos de cada canal a una frecuencia máxima de 20 MHz en un valor digital de 10 bits. Los líneas de salida de datos del conversor están conectadas directamente a la FPGA, junto con la señal de control para la selección del canal a leer desde el ADC, la señal de *sleep* y la de *chip-select*.

El conector para el puerto paralelo es del tipo DB25. Existen algunos componentes entre la FPGA y el conector que sirven para realizar funciones adicionales con las mismas líneas. Estos componentes actúan básicamente como multiplexores que son comandados por pulsadores y puentes de la placa, pero puede considerarse que existe una conexión directa entre los pines de la FPGA y los del conector del puerto paralelo cuando la placa está configurada de manera adecuada.

También se utilizan un reloj de 25 MHz incluido en la placa y un pulsador que actúa como reset.

4.1.2. Módulo superior

El sistema general debería poder realizar las siguientes funciones:

- Generar las señales adecuadas para poder comunicarse con la PC a través del puerto.
- Recibir las instrucciones de configuración de los componentes internos desde la PC.
- Recibir las instrucciones de comienzo y fin del funcionamiento del osciloscopio desde la PC.
- Controlar al conversor analógico digital para poder recibir los datos.
- Realizar algún tipo de procesamiento de los datos y almacenarlos en una memoria.
- Enviar los datos a la PC.

Estas funcionalidades son las implementadas en los módulos internos del sistema implementado en la FPGA.

En la Figura 4.4 se muestra el esquema general de la lógica desarrollada. Básicamente, el módulo superior (el que incluye a todos los demás módulos), consta de cuatro componentes principales interconectados.

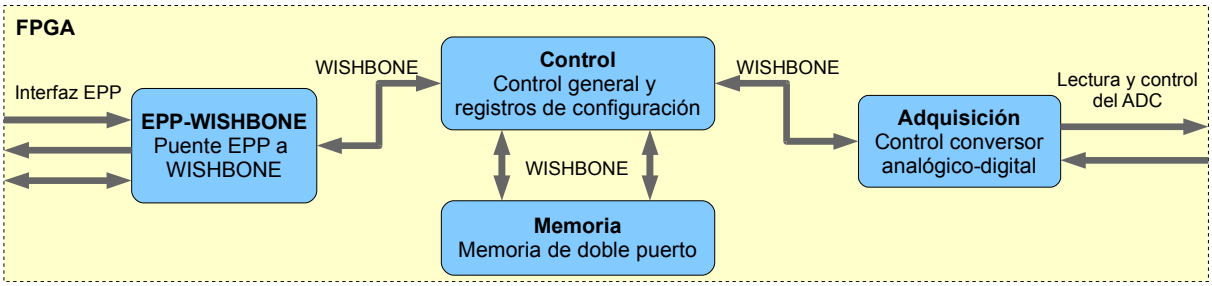

Figura 4.4: Esquema general.

Los componentes principales son:

- **EPP-WISHBONE**: Un puente de la interfaz EPP a la interfaz de interconexión interna WISHBONE.
- **Memoria**: Una memoria de doble puerto.
- **Adquisición**: Un módulo encargado de realizar el control de la adquisición. Genera las señales de reloj y selección de canales.
- **Control**: Un módulo central de control en el que están incluidas todas las funciones del osciloscopio y los registros de configuración. Aquí están implementadas las funciones del osciloscopio.

4.2. Puente EPP a WISHBONE

Este componente es el encargado de transformar la interfaz EPP (siguiendo el estándar IEEE 1284-2000), utilizada en el puerto paralelo, en una interfaz WISHBONE (siguiendo las especificaciones WISHBONE Rev. B.3) con un bus de datos de 16 bits y un bus de direcciones de 8 bits. Las Figura 4.5 y Figura 4.6 muestran el diagrama interno de este componente. La primera pertenece a un nivel superior y muestra la interconexión del puente con un módulo de extensión de ancho de bus de datos. La segunda muestra los elementos del puente.

4.2.1. Extensión del ancho del bus

Por razones expuestas posteriormente, los datos utilizados en otros módulos son de 16 bits. El puerto paralelo en modo EPP puede comunicar datos de hasta 8 bits en forma bidireccional, por lo que es necesario realizar una adaptación para la comunicación de los datos de 16 bits entregados por los los demás componentes.

Las posibles implementaciones consideradas para realizar la adaptación son: aprovechar las posibilidades de granularidad de la especificación WISHBONE, con la cual se pueden realizar transferencias de datos de unidades más pequeñas que las que la interfaz puede soportar, o incorporar un módulo intermedio que realice la adaptación del ancho del bus del puerto paralelo, el cual brindaría en forma transparente una interfaz con un bus de datos de 16 bits para el resto de los módulos. Por considerarla más práctica y simple, se optó por la segunda opción para este trabajo.

Como se puede observar en la Figura 4.5, el módulo EPP-WISHBONE puede dividirse en dos bloques principales: un puente EPP a WISHBONE con bus de datos de 8 bits y un módulo cuya función es la de realizar la extensión del ancho del bus.

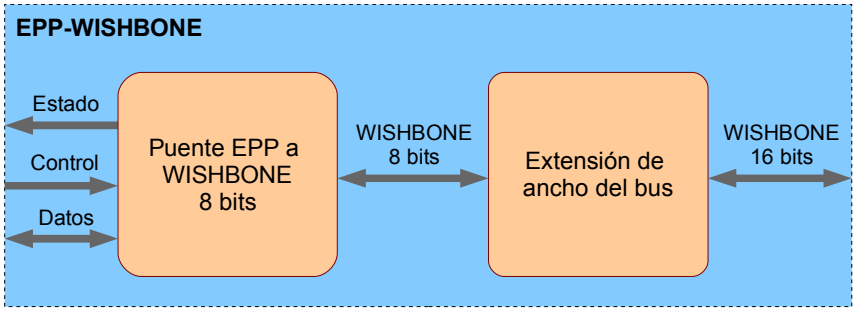

Figura 4.5: Diagrama interno del puente EPP-WISHBONE (16 bits).

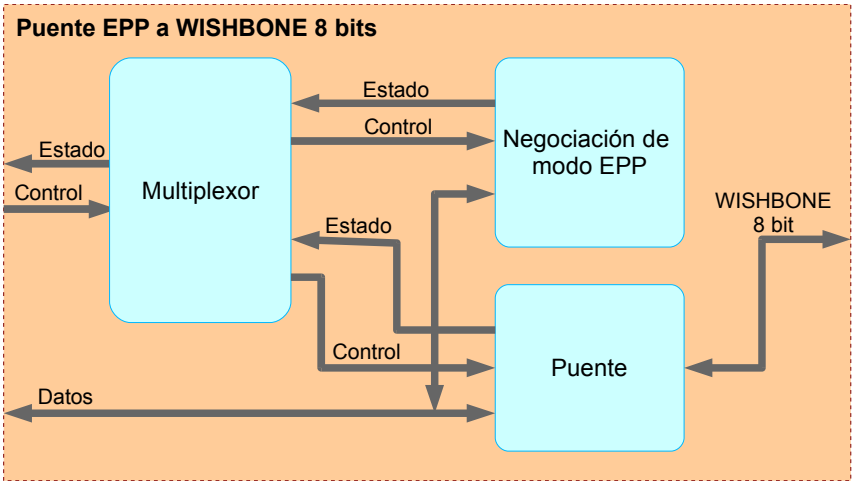

Figura 4.6: Diagrama interno del puente EPP-WISHBONE 8 bits

La extensión del ancho del bus funciona de la siguiente manera:

- **Escritura**: El primer dato escrito (de 8 bits) es almacenado en un registro interno. Cuando se está escribiendo el segundo dato, el bus de 16 bits se compone por el dato almacenado (byte menos significativo) y por el dato que se está escribiendo (byte más significativo), y las señales de handshake se trasmiten a través de la interfaz externa. Cuando el dispositivo destino responde, se reinicia el proceso.
- **Lectura**: Cuando el puente intenta leer un dato, las señales de handshake se transmiten a través de la interfaz hacia el resto de los componentes. Cuando el dispositivo destino responde, se transmite hacia el puente, a través del bus de 8 bits, el byte menos significativo del dato leído y se almacena en un registro interno el byte más significativo. En la segunda lectura desde el puente, se transmite a través del bus el byte almacenado y se reinicia el proceso.
- **Sincronismo**: Existen dos procesos que reinician el módulo al estado inicial (lectura/escritura del primer byte). El primero, es un contador interno que reinicia el módulo al llegar a su cuenta final si se encuentra esperando la lectura/escritura del segundo byte por un tiempo prolongado. El segundo es a través del bus de direcciones. Si el segundo byte intenta escribirse/leerse en una dirección diferente que el primero, se interpreta que se está intentando escribir/leer un nuevo dato de 16 bits.

La adaptación solo es realizada para el bus de datos. El bus de direcciones es

transferido sin modificaciones, permaneciendo con un ancho de 8 bits.

En resumen, para escribir un dato (de 16 bits) a través del puerto paralelo, se tendrá que escribir primero la dirección destino y luego se deberán trasmitir dos bytes seguidos, primero el menos significativo y luego el más significativo. Para leer un dato, se tendrá que escribir primero la dirección del dato y se deberá leer el puerto dos veces seguidas, en la primera lectura se recibirá el byte menos significativo y en la segunda el más significativo. Cada vez que se lee/escribe un dato un una dirección diferente, es seguro que el primer byte transmitido será el menos significativo.

4.2.2. Funcionamiento del puente de 8 bits

El puente EPP-WISHBONE básico proporciona una adaptación de la interfaz EPP a una interfaz WISHBONE con un bus de datos de 8 bits.

Para establecer el modo EPP descrito por el estándar IEEE 1284, primero debe realizarse una negociación del modo, la cual es un proceso en donde se intercambian determinados mensajes entre el host (PC) y el periférico. Básicamente, el proceso puede dividirse en las siguientes partes: host inicia modo de negociación y espera respuesta, periférico responde, host consulta si se soporta el modo EPP, si periférico responde afirmativamente, se establece el modo EPP. Las respuestas al host son generadas por el bloque *Negociación de modo EPP* de la Figura 4.6. Una vez establecido el modo, el módulo de negociación configura el *Multiplexor* para transferir las señales correspondientes al bloque *Puente*.

Cuando está establecido el modo EPP, comienza a funcionar el puente. A través del modo EPP pueden escribirse y leerse direcciones y datos. Cuando se escribe una dirección, ésta es almacenada en un registro y transferida directamente al bus de direcciones de la interfaz WISHBONE. Para la lectura/escritura de datos, se realiza una adaptación de las señales de handshake. Por ejemplo, cuando se intenta leer un dato desde la PC, se envían las señales correspondientes a través de la interfaz WISHBONE para realizar una lectura, luego de que el componente destino informa que tiene los datos disponibles, éstos son transferidos al puerto paralelo y el puente responde a la PC terminando el ciclo de lectura.

Para terminar el modo EPP y reiniciar el puerto, el host debe enviar una señal de finalización de modo (establecida por el estándar), la cual es monitoreada constantemente por el bloque de negociación, y, en caso de ser recibida, se restablece el multiplexor y será necesaria una renegociación para volver al modo EPP.

4.3. Adquisición

El módulo de adquisición es el encargado de generar las señales correspondientes para controlar al conversor analógico-digital. Las señales más importantes son las de reloj, para controlar la frecuencia de muestreo del conversor, y la de selección de canal. Las otras líneas que pueden controlarse son las de *sleep* y *chip-select*. En la Figura 4.7 se muestra un diagrama representativo del módulo.

El conversor analógico-digital incluido en la placa de desarrollo tiene una resolución de 10 bits. El bus de datos de la especificación WISHBONE puede ser de 8, 16, 32 ó 64 bits. Por esta razón, se ha utilizado un bus datos de 16 en la interfaz interna del módulo de adquisición.

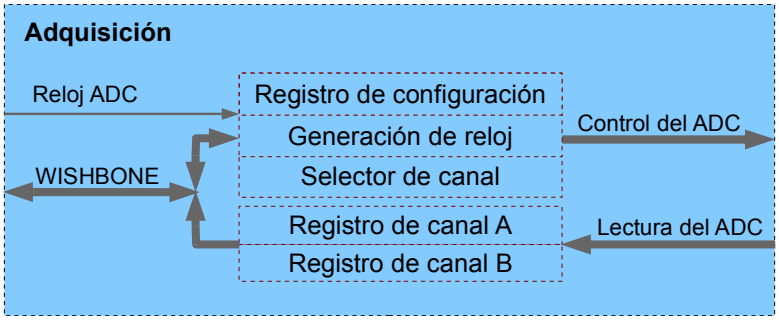

Figura 4.7: Diagrama interno del módulo de adquisición.

4.3.1. Selección de la frecuencia del reloj

El módulo posee dos entradas de reloj, una para la interfaz WISHBONE y otra para el conversor analógico-digital. La última puede ser transferida directamente al conversor o puede ser reducida en frecuencia, según la configuración de un registro interno.

Mediante el registro de configuración, se puede activar o desactivar la división de frecuencia y establecer el valor de división. El registro de división tiene 9 bits, y la frecuencia resultante, cuando el divisor está activado, es:

$$f_{div} = \frac{f_{clk}}{2(R_{div}+1)} \tag{4.1}$$

donde f_{div} es la frecuencia resultante, f_{clk} la frecuencia del reloj para el conversor y R_{div} es el valor de 9 bits del registro puede valores entre 0 y 511.

4.3.2. Obtención de los datos del conversor

El conversor analógico-digital realiza la conversión de las señales analógicas de los dos canales en forma simultánea, de forma tal que son accesibles en cada ciclo del reloj.

El módulo almacena en registros internos ambos canales en cada ciclo del reloj de salida hacia el conversor, uno en el flanco ascendente y otro en el descendente. Cada registro tiene asociada una bandera que indica si el registro ha sido leído.

Desde la interfaz WISHBONE es posible seleccionar el canal a leer a través del bus de direcciones. Cuando se realiza una lectura desde otro módulo, se entrega el valor almacenado del canal seleccionado y se activa la bandera de lectura asociada.

Si desde la interfaz WISHBONE se intenta realizar la lectura de un canal cuando su bandera de lectura está activada, el módulo insertará tiempos de espera y solo responderá cuando el registro sea actualizado con los datos del conversor. De esta manera, pueden utilizarse velocidades diferentes en el conversor y en la interfaz WISHBONE.

4.3.3. Registro de configuración

El registro tiene una dirección específica dentro del módulo y puede escribirse en un simple ciclo de escritura desde la interfaz WISHBONE. Desde este registro pueden establecerse los valores de las señales *sleep* y *chip select*, y puede configurarse la división de la frecuencia de reloj.

4.4. Memoria

Para almacenar los datos obtenidos por el conversor analógico digital a altas velocidades, debe utilizarse un buffer, ya que si los datos fueran enviados directamente a través del puerto paralelo, la velocidad de adquisición se vería reducida. Para implementarlo, se creó una memoria de doble puerto con la herramienta SmartGen de Actel, utilizando los bloques de memoria SRAM internos de la FPGA. A esta memoria se le realizó una adaptación de las señales de handshake para utilizarla a través de una interfaz WISHBONE. Se muestra en la Figura 4.8 una representación del esquema interno del módulo.

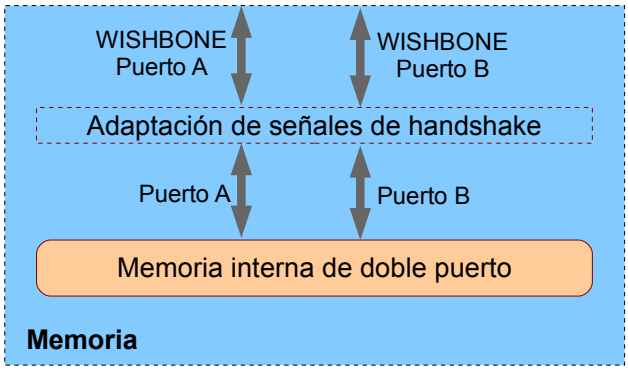

Figura 4.8: Diagrama interno de la memoria de doble puerto.

4.4.1. Tamaño

Debido a que el ancho del bus de datos utilizado es de 16 bits, aunque no sea ocupada completamente, se optó por generar una memoria interna también con un ancho de 16 bits. Los bloques SRAM internos de la ProASIC3E1500, utilizando un ancho de 16 bits, permiten obtener un rango de direcciones de hasta 15360 valores. Esto posibilita almacenar hasta 15360 muestras. En el Capítulo 6 se realiza un análisis más detallado sobre la utilización de recursos.

4.4.2. Utilización

La transferencia de datos se realiza a través de ciclos de lectura/escritura de la interfaz WISHBONE. Están soportados los ciclos individuales, en bloques y RMW de la especificación. Cada puerto utiliza relojes diferentes.

Para evitar entrar en ciertos tiempos prohibidos indicados por las hojas de datos de la FPGA, no pueden realizarse lecturas/escrituras desde ambos puertos en las mismas posiciones de memoria al mismo tiempo. Cuando se intente acceder a la misma posición de memoria desde ambos puertos, tendrá prioridad el puerto A y se generarán tiempos de espera en la interfaz WISHBONE del puerto B. El puerto B solo responderá cuando se haya completado la operación en el puerto A.

4.5. Control

El módulo de control es el componente principal del osciloscopio. Es el encargado de realizar las siguientes tareas:

- Almacenar las configuraciones recibidas desde la PC o comunicarse con el módulo de adquisición para configurarlo

- Comunicarse con el módulo de adquisición para para recibir sus datos, leyendo los canales adecuados según la configuración.
- Almacenar en forma consecutiva, en la memoria de doble puerto, los valores leídos desde el módulo de adquisición hasta una posición determinada por la configuración, es decir, el tamaño del buffer seleccionado.
- Permitir la selección de la base de tiempo del osciloscopio.
- Posibilitar la lectura de manera progresiva y controlada de los datos almacenados en la memoria de doble puerto.
- Implementar un mecanismo de disparo por pendiente y nivel.
- Iniciar o terminar el proceso de adquisición cuando es solicitado por la PC y permitir el monitoreo del estado de la conversión.

La Figura 4.9 representa la estructura interna del módulo de control. Este módulo posee cuatro interfaces WISHBONE, una tipo esclavo, encargada de recibir las órdenes desde la PC, y tres tipo maestro, encargadas de comunicarse con ambos puertos de la memoria y de comunicarse con el módulo de adquisición.

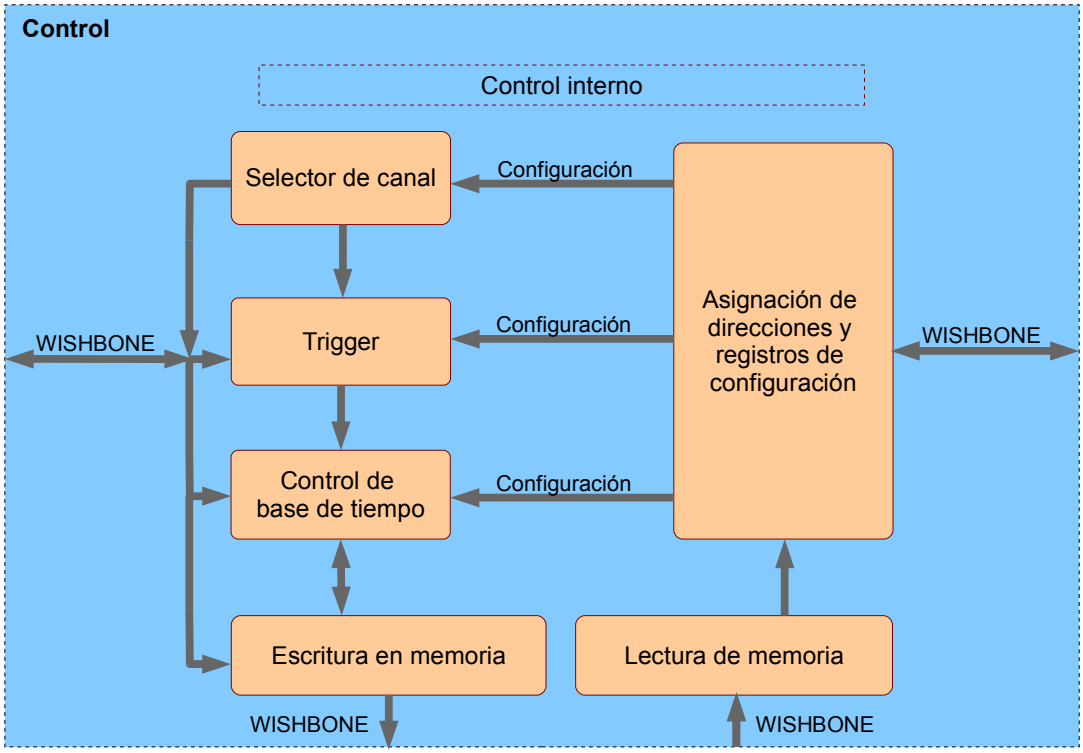

Figura 4.9: Diagrama interno del módulo de control.

4.5.1. Control de base de tiempo

Cuando el osciloscopio está en funcionamiento, el módulo de **control de la base de tiempo** supervisa las señales de handshake de la comunicación con el módulo de adquisición, y actúa sobre el módulo de escritura en memoria. Cuenta las respuestas recibidas desde la interfaz WISHBONE y habilita la escritura en memoria de los datos cada cierto tiempo. Como resultado de este proceso, algunos valores serán almacenados en el buffer y otros serán descartados.

La cantidad de datos descartados depende del valor de un registro de configuración.

Según su valor, se almacenará en memoria un valor cada *n* datos leídos, según la siguiente ecuación:

$$n = 2^{R_t + 1} \tag{4.2}$$

donde R_t es el valor del registro de configuración del selector de la base de tiempo, y tiene un tamaño de 5 bits, permitiendo valores entre 0 y 31. Por ejemplo, teniendo una frecuencia de muestreo de 20 MHz en el conversor, con un valor de R_t igual a 22, el período de muestro equivalente será:

$$T_{s\ eqiv} = 2^{22+1} \frac{1}{20\,\text{MHz}} = 0{,}42\,\text{s} \tag{4.3}$$

Deshabilitando este componente no se descartará ningún dato y se almacenarán a la velocidad de adquisición.

4.5.2. Selector de canal

El módulo de adquisición permite la selección del canal a leer través del bus de direcciones. El **selector de canal** genera la dirección de lectura para seleccionar el canal adecuado, según un registro de configuración.

El registro de configuración posee dos bits, con los cuales, según si están activos o no, es posible seleccionar solo el canal A, solo el B o ambos. En el caso de estar seleccionados ambos canales, se irán leyendo uno y otro en forma consecutiva. Con esta última configuración, el tamaño del buffer será repartido para almacenar los datos de los dos canales en forma equitativa.

El selector de canal también actúa sobre el control de la base de tiempo y sobre el mecanismo de disparo. Cuando están seleccionados los dos canales, el control de la base de tiempo permite el almacenamiento de los datos de los dos canales y luego comienza el tiempo de espera. En la Figura 4.10 se ejemplifica cómo serán almacenados los datos de los canales, indicando los tiempos relativos en los que fueron tomadas las muestras, donde $T_{s\ equiv}$ es el período de muestreo equivalente resultante, obtenido como se explicó anteriormente.

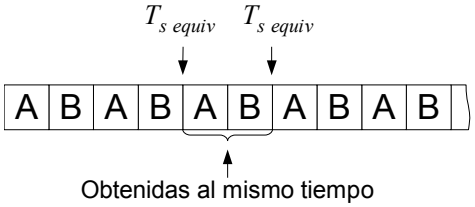

Figura 4.10: Ordenamiento en memoria resultante.

El mecanismo de disparo deberá utilizar la información del selector de canal para conocer a qué canal corresponde el dato con el que se está trabajando, como se explica a continuación.

4.5.3. Trigger

El **mecanismo de disparo** (trigger) funciona de forma similar que en los osciloscopios tradicionales. Cuando está activado, indica al osciloscopio cuándo comenzar a graficar

la señal. Como en otros osciloscopios, se puede seleccionar entre una pendiente creciente o decreciente y el nivel de tensión de esa pendiente en en el cual el mecanismo actuará. La Figura 4.11 ejemplifica estas configuraciones [26]. La selección del nivel, la pendiente y el canal que se monitorizará se realiza a través del registro de configuración del trigger.

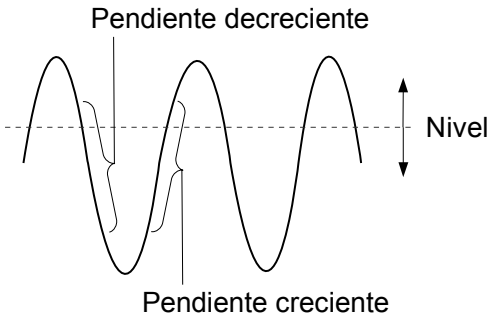

Figura 4.11: Mecanismo de disparo.

El registro de configuración también permite seleccionar el tiempo relativo en que se comenzará a graficar la señal, desde el punto en que se produjo el disparo. Este valor puede ser positivo o negativo, pero no puede superar en valor absoluto al tamaño del buffer. Si es negativo, se utilizarán los datos previamente almacenados en el buffer. Si es positivo, se esperará cierto tiempo para comenzar a enviar los datos a la PC.

Puede seleccionarse cualquiera de los dos canales para ser utilizado por el trigger. Este módulo recibe los datos del selector de canal para poder diferenciarlos al procesar los datos.

4.5.4. Escritura en memoria

El componente de **escritura en memoria** se encarga de almacenar los datos en el buffer. Se comunica con el módulo de adquisición a través de una interfaz WISHBONE, utilizando las direcciones provistas por el selector de canal, y almacena los datos en los intervalos indicados por el control de la base de tiempo, desde el momento señalado por el trigger (si está activado).

Los datos son almacenados en la memoria de doble puerto, en direcciones consecutivas, hasta completar el tamaño del buffer indicado por un registro de configuración. No se escriben más valores hasta que todos los datos sean leídos por la PC, lo cual es señalado por señales de control enviadas por el módulo de lectura de memoria.

Junto con los valores leídos, se almacena en la memoria el canal al cuál pertenece, utilizando un bit adicional. Esto permite diferenciar los canales cuando luego los datos son interpretados por la computadora.

4.5.5. Lectura de memoria

El bloque de **lectura de memoria** permite la lectura de los datos almacenados en el buffer desde el puente EPP-WISHBONE y se encarga de mantener el sincronismo entre los datos que se están utilizando y los que se están obteniendo (almacenando en memoria).

Este bloque se habilita según señales de control desde el trigger, si está activado, o, en

caso contrario, desde que se almacena el primer dato en memoria.

4.5.6. Asignación de direcciones y registros de configuración

Todas las configuraciones son almacenadas en un mismo módulo que se encarga de distribuirlas a los demás. El componente de **asignación de direcciones y registros de configuración** se encarga de esta tarea, y de proveer una interfaz WISHBONE para acceder a estos registros. También asigna direcciones para configurar el módulo de adquisición y para leer los datos del buffer a través del bloque de lectura de memoria.

Internamente, está compuesto de una tabla de asignación de direcciones y algunos componentes para generar las señales de handshake. En el Anexo *Asignación de direcciones internas* se muestra la composición de la tabla.

También se implementan en este módulo los controles necesarios para almacenar el registro de configuración del módulo de adquisición.

4.5.7. Control interno

El **control interno** interpreta el estado de los componentes del módulo y genera señales para comandarlos. Permite dos modos de funcionamiento: continuo y simple. El primero es el tradicional, donde los valores son obtenidos y mostrados continuamente en el visor del osciloscopio. El segundo suele estar incorporado en los osciloscopios digitales y solo muestra los valores en pantalla una sola vez, cuando es indicado a través de los controles desde la PC.

Este módulo también implementa los mecanismos de inicio y finalización de la adquisición. El osciloscopio comienza a funcionar cuando se escribe un uno lógico en un bit de un registro de configuración. El mecanismo de finalización se activa simplemente mediante la escritura de un dato en cualquiera de los registros de configuración, salvo que se escriba un uno en el bit de inicio, en cuyo caso, la adquisición es reiniciada.

5. Descripción del software

En este capítulo se realiza una descripción del software desarrollado para utilizar el osciloscopio virtual. Primero se comenta el modo de utilización del programa describiendo su interfaz de usuario y luego se realiza una descripción de la estructura de sus clases.

5.1. Utilización

En la Figura 5.1 se muestra una captura de la interfaz de usuario del osciloscopio. En los siguientes puntos se describe la función cada uno de los elementos de la interfaz gráfica.

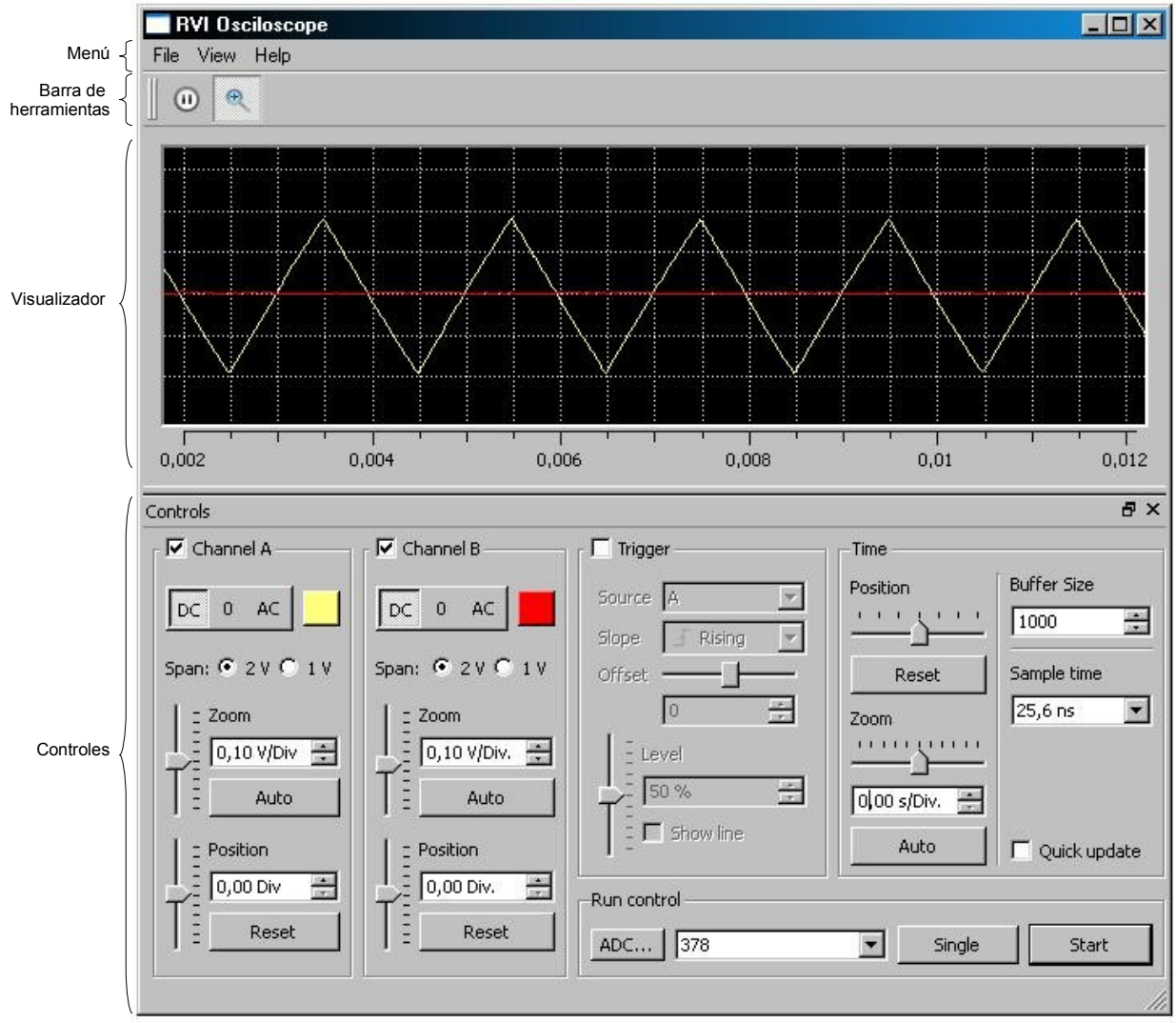

Figura 5.1: Interfaz gráfica del software del osciloscopio.

5.1.1. Menú

File

Save image: Permite almacenar una imagen con el contenido actual del visualizador

en el archivo indicado.

Save data: Permite almacenar los últimos datos que se han obtenido. El archivo destino será una tabla con las columnas separadas por comas. La primera contendrá los valores del tiempo, la segunda los valores del canal A y la tercera los del canal B.

Exit: Sale del programa.

View

Background color: Abre un cuadro de diálogo para seleccionar el color del fondo del visor.

Show controls: Muestra u oculta los controles. Útil para agrandar el tamaño del visualizador.

Dock controls: Incorpora el cuadro de controles en la ventana principal o lo separa. Útil para agrandar el tamaño del visualizador.

Help

Help: Abre una ayuda de utilización del programa.

About: Muestra información relacionada con la versión actual del programa.

5.1.2. Barra de herramientas

Zoom: Permite realizar un acercamiento a un área seleccionada del visor.

Pause: Congela la imagen mostrada en el visor, independientemente del estado de la adquisición.

5.1.3. Controles

En la Figura 5.2 se muestran los controles y se asigna un nombre a los elementos de control que no lo tienen indicado explícitamente en la interfaz, para mayor claridad en la explicación.

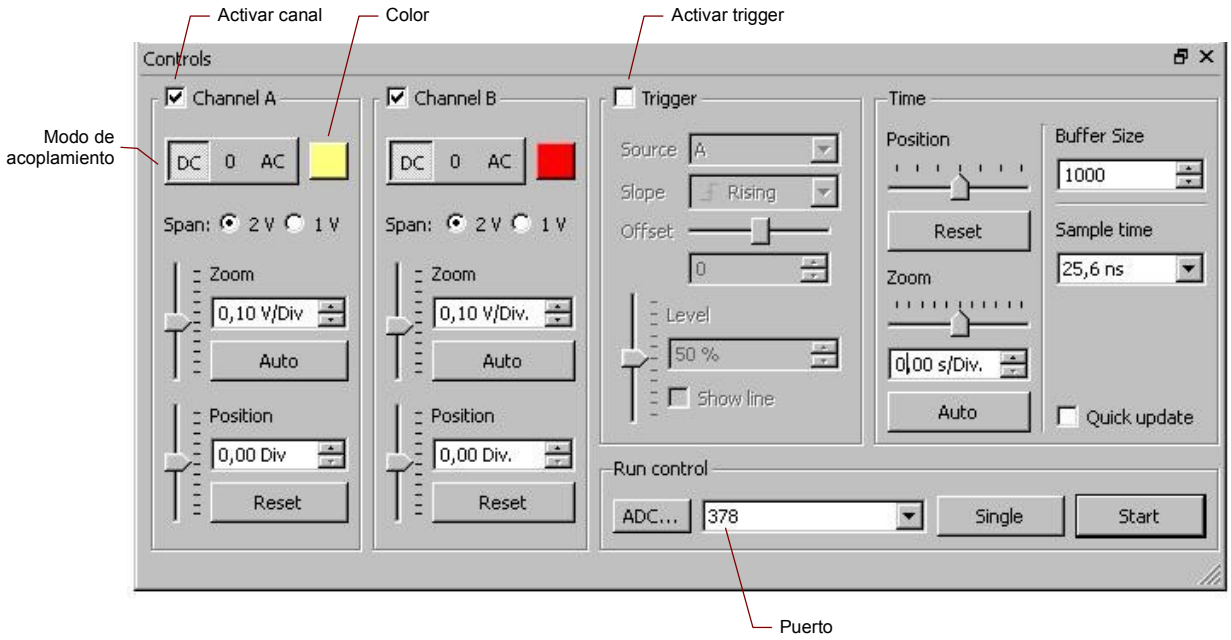

Figura 5.2: Cuadro de controles.

Canales A y B

Activar canal: Activa o desactiva el canal.

Color: Cambia el color de la forma de onda del canal.

Modo de acoplamiento: Permite adaptar los gráficos al modo de acoplamiento seleccionado en la placa. Debe coincidir con la configuración de los puentes para el canal, según la tabla Tabla 5.1. El valor 0 está generado por software y es independiente de los valores de los puentes.

Span: Permite adaptar los gráficos al rango de tensiones seleccionado en la placa. Debe coincidir con la configuración de los puentes, según la tabla Tabla 5.2. La configuración es la misma para ambos canales.

Zoom: Permiten seleccionar la relación de Volts por división mostrada en el visor.

Auto: Permite ajustar automáticamente la estaca a partir de los valores máximos de la onda.

Position: Permiten desplazar la onda hacia arriba o abajo en el visor.

Reset: Vuelve la onda a la posición vertical original (sin offset).

Canal	Jumper	DC	AC
A	JP1	1-2	abierto
	JP3	abierto	1-2
	JP10	1-2	2-3
B	JP2	1-2	abierto
	JP9	abierto	1-2
	JP11	1-2	2-3

Tabla 5.1: Configuración del acoplamiento.

Jumper	0-1V	0-2V
JP4	2-3	1-2

Tabla 5.2: Configuración del rango.

Trigger

Activar trigger: Activa o desactiva la función de trigger.

Source: Selecciona el canal monitorizado por el trigger. Solo serán visibles los canales activados.

Slope: Selecciona si el trigger es por pendiente positiva o negativa.

Offset: Selecciona la posición relativa dentro del buffer en la que serán mostrados los datos, a partir del momento en que se cumplen las condiciones establecidas para el trigger. Puede tener valores positivos y negativos, pero siempre menores en valor absoluto que el tamaño del buffer.

Level: Selecciona el nivel utilizado para el trigger.

Show line: muestra una línea horizontal en el visor que representa el nivel seleccionado del trigger.

Tiempo

Position: Desplaza las ondas hacia la izquierda o hacia la derecha.

Reset: Vuelve las ondas a su posición inicial, con el primer valor a la izquierda del visualizador.

Zoom: Permite realizar un ajuste de la relación segundos por división mostrada en el display. La aumenta moviendo el deslizador hacia la izquierda y la disminuye hacia la derecha. Si se mueve hasta las posiciones de los extremos, se cambia la frecuencia de muestreo.

Auto: Ajusta la escala automáticamente, a partir de los valores máximos del tiempo.

Sample time: Permite cambiar el período de muestreo.

Buffer Size: Permite seleccionar el tamaño del buffer utilizado, medido en cantidad de muestras almacenadas. El valor máximo es de 15360 y el mínimo de 10.

Quick update: Si está desactivado, las ondas mostradas en el visor se actualizarán luego de que se lea el buffer de la placa completamente. Activando esta opción, el visor puede actualizarse con mayor frecuencia. Esta opción es útil cuando se trabaja con frecuencias bajas.

Ejecución

ADC...: Abre un cuadro de diálogo que permite seleccionar la frecuencia de reloj utilizada en el conversor A/D.

Dirección del puerto: Selecciona la dirección base del puerto paralelo.

Single: Al presionarse el botón se inicia el osciloscopio en el modo simple. Se obtienen los datos hasta completar el buffer del osciloscopio una sola vez, luego se detiene la adquisición. Puede detenerse la adquisición antes presionándose nuevamente el botón.

Start: Al presionarse el botón se inicia el osciloscopio en el modo contínuo. Cada vez que se leen desde la PC todos los datos almacenados en el buffer de la placa, se comienza con una nueva adquisición. Puede detenerse la adquisición volviendo a presionarse el botón.

5.2. Diseño

El software está diseñado para ser utilizado en el sistema operativo Windows XP SP2 o superior, no ha sido probado en versiones previas. En los puntos siguientes, se realiza una breve descripción de la estructura de las clases en C++ y de las bibliotecas de software utilizadas.

5.2.1. Puerto paralelo

En Windows XP (y en otras versiones de este sistema basadas en NT), por razones de seguridad, existe una estructura de privilegios para acceder a los recursos del sistema. Solo los procesos que son confiables para el sistema operativo pueden acceder a los puertos de entrada/salida (como el puerto paralelo), previniendo que procesos de

usuario menos confiables causen conflictos al modificar los puertos. Solo el kernel y los controladores de dispositivos se ejecutan con el nivel de privilegios suficiente como para modificar los puertos. Por ello, los procesos de usuario deben comunicase con un controlador de dispositivos que controle el acceso para poder utilizarlos [27]. Otra posibilidad es utilizar ciertas utilidades que desactivan la protección del sistema para los puertos seleccionados, pero este mecanismo puede resultar más complicado para el usuario final y resta seguridad al sistema.

Para posibilitar el acceso al puerto paralelo desde el software, en este trabajo se utilizó la biblioteca dinámica para Windows llamada **io.dll**. Ésta provee un set de instrucciones útiles para leer y escribir en los puertos de entrada/salida. Posee un pequeño controlador embebido en el archivo .dll que habilita los puertos cuando son requeridos por la aplicación. En el sitio web del desarrollador, se pueden descargar tanto la biblioteca como varios ejemplos sobre cómo acceder a sus funciones desde varios lenguajes de programación, incluyendo C++. Esta biblioteca puede utilizarse en forma gratuita, pero no es de código abierto [28].

Para facilitar la utilización del puerto desde el programa, se creó una estructura de clases. El programa se comunica con la clase de utilización del puerto, que brinda funciones para realizar tareas con el puerto, como la negociación, la notificación de errores y la escritura y lectura de una forma transparente. La clase de acceso al puerto trabaja a más bajo nivel, y provee las funciones básicas utilizables a través de la biblioteca io.dll. En la Figura 5.3 se muestra una representación de esta estructura.

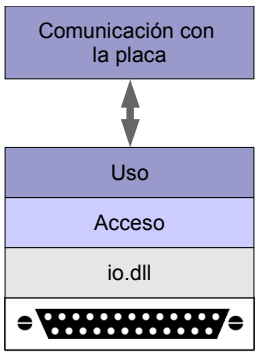

Figura 5.3: Estructura de clases del puerto paralelo

5.2.2. Interfaz gráfica

El programa principal del osciloscopio está basado en las bibliotecas Qt versón 4.5.1 para Windows. En la Figura 5.4 se muestra una representación de la estructura del programa.

Al ejecutarse el programa, se carga la ventana principal que contiene el menú, la barra de herramientas y la barra de estado. Dentro de ella se incorporan el **visualizador** y los **controles**. Cada uno está diseñado en una clase separada, para brindar una estructura basada en bloques al software.

La clase donde está implementado el cuadro de controles es la encargada de realizar las comunicaciones con la placa y con el plotter. Hacia el plotter, envía los vectores con los valores de las señales y de tiempo y las señales de actualización. Hacia la placa, se comunica a través de la clase **comunicación con la placa**.

La clase de comunicación con la placa se encarga de realizar la traducción de los

parámetros que recibe en un bloque de datos con los valores adecuados para configurar el osciloscopio, y luego de enviarlo a través del puerto paralelo. Cuando el osciloscopio se encuentra en funcionamiento, éste módulo recibe los datos, los almacena en memoria RAM y se encarga de verificar e indicar el estado del osciloscopio. En caso de utilizarse nuevos puertos para la comunicación con la placa, como un puerto USB, podrán crearse nuevas clases de comunicación como ésta sin importantes modificaciones en las demás partes del código.

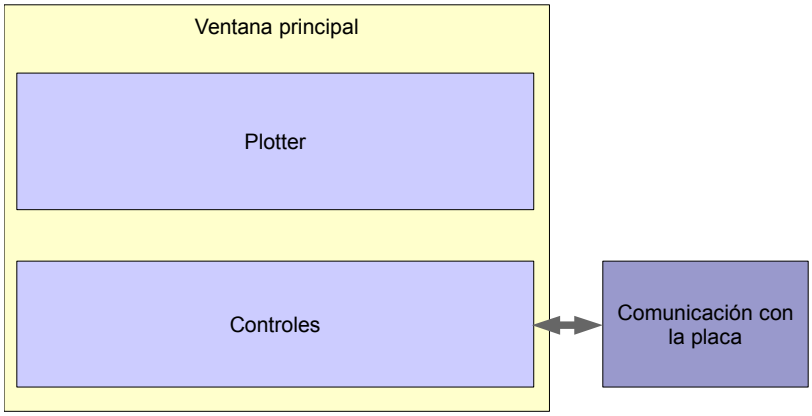

Figura 5.4: Estructura de la interfaz gráfica.

6. Conclusiones

Aquí se realiza un análisis de los resultados obtenidos a partir del desarrollo de este trabajo y de las implementaciones o mejoras futuras posibles basadas en este proyecto.

6.1. Especificaciones finales

Las especificaciones finales dependen del diseño realizado en este trabajo y de los componentes de la placa de desarrollo. En los siguientes puntos se realiza una descripción de las especificaciones resultantes del osciloscopio.

6.1.1. Análisis del ancho de banda

Existen dos limitaciones principales sobre la velocidad a la que se utilizan los datos, una está dada por la frecuencia máxima de conversión soportada por el conversor analógico/digital y otra por velocidad de comunicación a través del puerto paralelo.

El conversor A/D soporta una frecuencia de adquisición máxima de 20 MHz. Es posible leer los dos canales en el mismo ciclo de reloj, por lo que ambos pueden utilizar una frecuencia de muestreo máxima de 20 MHz.

Con una frecuencia de muestreo de 20 MHz, el ancho de banda máximo teórico que puede ser reconstruido es de 10 MHz. El módulo de control implementado en la FPGA es capaz de almacenar los datos de ambos canales en el buffer a esa frecuencia, por lo que no impone ninguna limitación en este diseño.

Por otro lado, la velocidad de comunicación a través del puerto paralelo depende mucho de los recursos disponibles en la PC utilizada. La velocidad media de comunicación suele ser de un megabyte por segundo (8 Mb/s), aunque en la práctica se obtienen valores entre los 500 kB/s y los 2 MB/s, dependiendo de circunstancias muy variables, como la cantidad de recursos utilizados por los programas que se encuentran en ejecución en la PC.

Los datos primero son almacenados directamente en el buffer a la velocidad soportada por el conversor. En la pantalla, los datos se representan a medida que se van leyendo desde el buffer a través del puerto paralelo. Luego, la pantalla se actualiza completamente cuando el buffer comienza a llenarse nuevamente.

Debe considerarse que si los datos fueran mostrados en tiempo real, cuando se trabaja a velocidades elevadas, sería imposible observar todas las actualizaciones del visualizador. En la mayoría de los casos, basta con que la pantalla del osciloscopio sea actualizada cada cierto tiempo, no necesariamente en tiempo real. Esto hace que la limitación impuesta por el puerto paralelo sea de menor importancia, siendo solo de interés cuando se requiere almacenar en la computadora una cantidad de datos mayor que la capacidad del buffer del osciloscopio y luego visualizarlos (esta característica no está implementada en este trabajo).

6.1.2. Tabla de especificaciones

En la Tabla 6.1 se muestra un resumen de las especificaciones del instrumento. El software requiere de una PC con Windows XP o superior, con puerto paralelo que soporte el modo EPP.

Máxima frecuencia de muestreo	20 MS/s
Resolución vertical	10 bits
Número de canales	2
Rangos de entrada (depende de configuración de los puentes de la placa)	-1 V a 1 V -2 V a 2 V
Acoplamiento de entrada (depende de configuración de los puentes de la placa)	AC DC
Capacidad de la memoria buffer por canal	15360 (usando un canal) 7680 (usando dos canales)
Trigger	Por nivel, pendiente positiva o negativa Con función de offset
Comunicación con PC	Puerto paralelo (EPP)
Modos de funcionamiento	Simple Contínuo

Tabla 6.1: Especificaciones finales.

6.1.3. Aplicaciones

Este osciloscopio posee varias de las características más comunes encontradas en los instrumentos comerciales y son suficientes para poder utilizarlo en la mayoría de las aplicaciones de frecuencias no muy elevadas. Es de bajo coste, liviano y pequeño (si no se considera la PC), y podrá ser utilizado en cualquier laboratorio que disponga de computadoras. Será muy útil en aplicaciones didácticas, ya sea para la enseñanza sobre la utilización de osciloscopios digitales como para el estudio de su arquitectura.

6.1.4. Recursos de hardware utilizados

Los recursos utilizados de la placa de desarrollo son:

- Conversor analógico digital, junto con los dos canales analógicos de entrada. Este conversor está incorporado en la placa de expansión LP Data Conversion Daughter Board.
- Conectores de la placa de expansión.
- FPGA.
- Puerto paralelo.
- Un pulsador de RESET.

El reporte de la síntesis de la lógica implementada en la FPGA indica la siguiente utilización de recursos resumida:

```
Core Cells         : 2073 of 38400 (5%)
IO Cells           : 39 (~9%)

  RAM/ROM Usage Summary
```

```
Block Rams : 60 of 60 (100%)
```

Las altas prestaciones de la FPGA incorporada en la placa permiten una escasa ocupación de recursos internos para la implementación del osciloscopio, consumiendo solo un 5% del total de celdas disponibles. Esto y el diseño modular utilizado, facilitarán la implementación de nuevas funcionalidades en los futuros trabajos realizados sobre el tema.

La FPGA utilizada posee 60 *Block Rams*, los cuales son bloques de memorias SRAM embebidas en el chip. Cada bloque posee una capacidad total de 4608 bits y tiene puertos de entrada y salida que pueden configurarse con diferentes tamaños para la lectura y para la escritura. Los arreglos posibles son 256x18, 512x9, 1kx4, 2kx8 y 4kx1 bits. En este caso, el osciloscopio utiliza una memoria con un ancho de bus de datos de 16 bits, y los tamaños útiles para este ancho son 1kx4, 2kx8 y 4kx1, los cuales brindan una capacidad de 4096 bits [29]. Utilizando combinaciones de estos tres arreglos, se puede conseguir un rango de hasta 15360 direcciones, según la siguiente fórmula:

$$\frac{4096 \cdot 60}{16} = 15360 \tag{6.1}$$

La herramienta SmartGen desarrollada por Acel permite la creación de memorias de doble puerto de diferentes tamaños, creando automáticamente un módulo con combinaciones de Block Rams. Utilizando esta herramienta para generar una memoria con 16 bits de ancho y 15360 datos de profundidad, se reporta una utilización del 100% de bloques de memoria. Si en diseños futuro se requiere la utilización de memoria RAM, las posibilidades son: compartir la memoria de doble puerto del osciloscopio utilizando la interfaz WISHBONE, generar varias memorias de menor capacidad dedicadas utilizando Block Rams, reduciendo las prestaciones del osciloscopio, o utilizar una memoria externa.

6.2. Cumplimiento de los objetivos

Se ha logrado desarrollar un osciloscopio virtual con funciones implementadas en una FPGA y visualización realizada a través de un software para PC. Se ha utilizado un diseño de hardware modular, facilitando futuras investigaciones a partir del desarrollo.

Se han aprovechado y utilizado correctamente los elementos incorporados en la *RVI Prototype Board*, sin necesidad de utilizar componentes adicionales.

Se han adquirido nuevos conocimientos en el diseño de sistemas digitales, sobre todo en la utilización de la lógica programable en el campo de la instrumentación. Se han aplicado los conocimientos adquiridos durante la carrera de Ingeniería en el diseño de un sistema concreto.

6.3. Posibles implementaciones futuras

Se pretende la utilización de este trabajo como caso de estudio o como base para realizar futuros proyectos. Todos los códigos fuente utilizados para el desarrollo de este osciloscopio serán publicados en forma abierta y de libre utilización. En los puntos siguientes se hacen algunos comentarios sobre las posibles incorporaciones que pueden hacerse a este trabajo.

6.3.1. Mejoras de ancho de banda

El ancho de banda, como se ha comentado previamente, posee limitaciones debido al puerto paralelo y debido al conversor analógico/digital.

La limitación más importante es causada por el conversor y reduce la frecuencia de muestreo. Esta limitación solo puede ser superada cambiando de conversor analógico digital, es decir, modificando componentes del hardware. El diseño de la *RVI Prototype Board*, facilita este trabajo mediante la incorporación de conectores para placas de expansión.

El puerto paralelo presenta limitaciones cuando se requiere adquirir grandes cantidades de datos en forma continua. Puede mejorarse la velocidad de comunicación con la PC incorporando nuevos módulos lógicos que se comuniquen a través de otros puertos de mayor velocidad incorporados en la placa de desarrollo, como USB o Ethernet. Esto se ejemplifica en la Figura 6.2, descrita más adelante. Este problema también puede mejorarse, en cierta forma, incorporando una memoria externa en el zócalo de la *RVI Prototype Board*, de forma que sea posible la utilización de un buffer con mayor capacidad de almacenamiento.

6.3.2. Nuevas funcionalidades

Las nuevas funcionalidades del osciloscopio, tales como operaciones entre los canales, analizadores de espectro, etc. se pueden incorporar de la siguiente manera: se diseñan los módulos necesarios para implementarlas, se le asigna una dirección a su registro de configuración en el módulo correspondiente, se le asignan canales de lectura si es necesario y finalmente se generan los estados y señales adecuadas dentro del control interno para su utilización. En el esquema de la Figura 6.1 se presenta un ejemplo sobre las posibles ubicaciones de los nuevos módulos dentro del bloque de control general y registros de configuración.

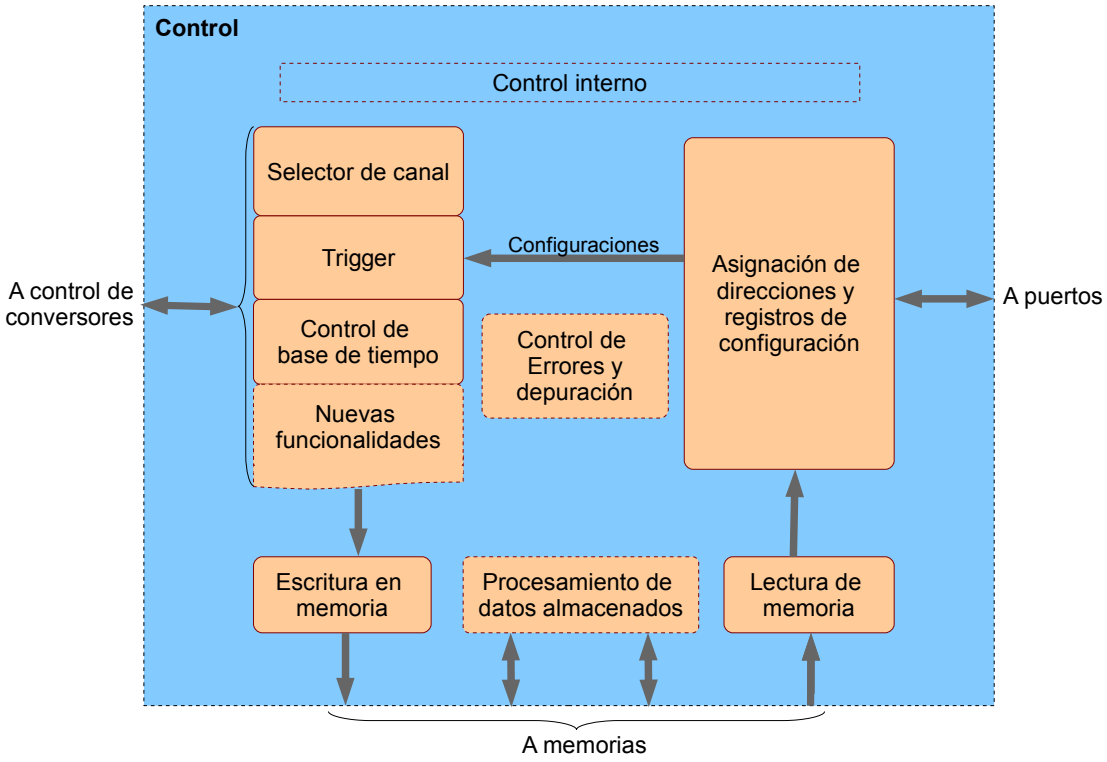

Figura 6.1: Incorporación de nuevas funcionalidades en el osciloscopio.

6.3.3. Nuevos instrumentos

En la Figura 6.2 se muestra una arquitectura propuesta para la incorporación de nuevos instrumentos y puertos de comunicación, realizada a partir del esquema mostrado en la Figura 4.4. Como puede observarse, se reutilizan los módulos previamente desarrollados y se incorporan los nuevos, creando bloques de interconexión para compartir los buses. Mediante la implementación de una arquitectura de este tipo, se podría disponer de todo un paquete de instrumentos disponibles en una misma placa y utilizables a través de una aplicación para PC.

Los bloques de interconexión entre núcleos deberán proveer todas las herramientas necesarias para el correcto funcionamiento del sistema, como una apropiada asignación de direcciones, árbitros para asignar intervalos de tiempo a las interfaces tipo MAESTRO, etc. También deberá asegurarse una correcta distribución de las memorias, pudiendo utilizarse una memoria externa en caso de que sea necesario aumentar la capacidad.

Instrumentos diferentes o con mayores prestaciones pueden requerir de la utilización de una nueva placa de expansión y de la incorporación de nuevos módulos lógicos en la FPGA. Por otro lado, gracias la utilización de un dispositivo lógico programable, es posible reimplementar completamente los módulos a través del software, extendiendo aún más la lista de posibles instrumentos que pueden desarrollarse.

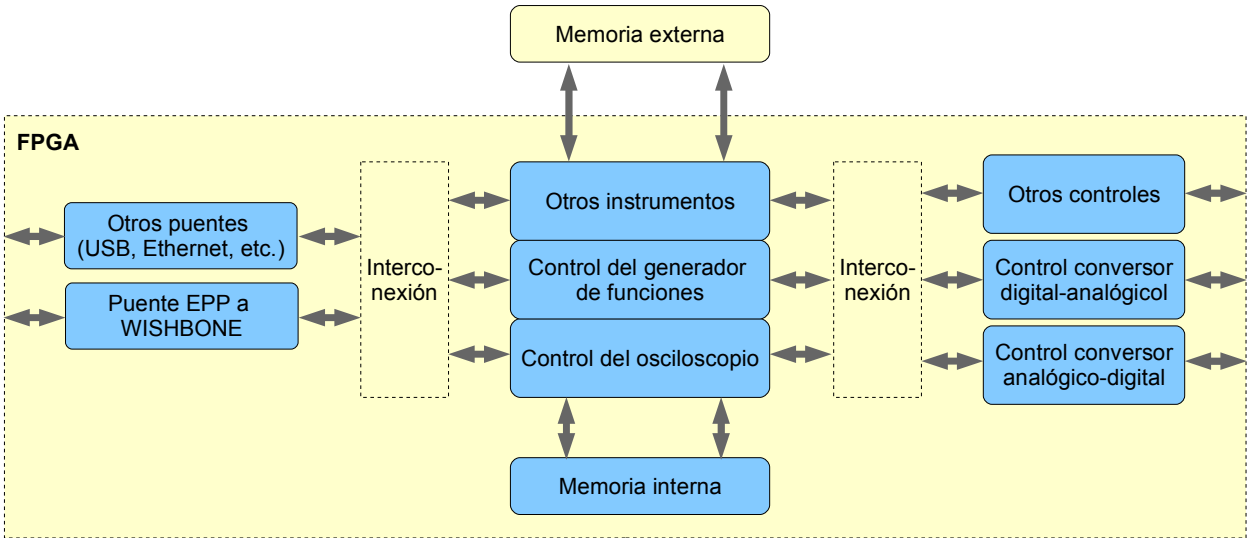

Figura 6.2: Esquema propuesto para futuros trabajos.

Anexo I. Esquemas de RVI Prototype Board

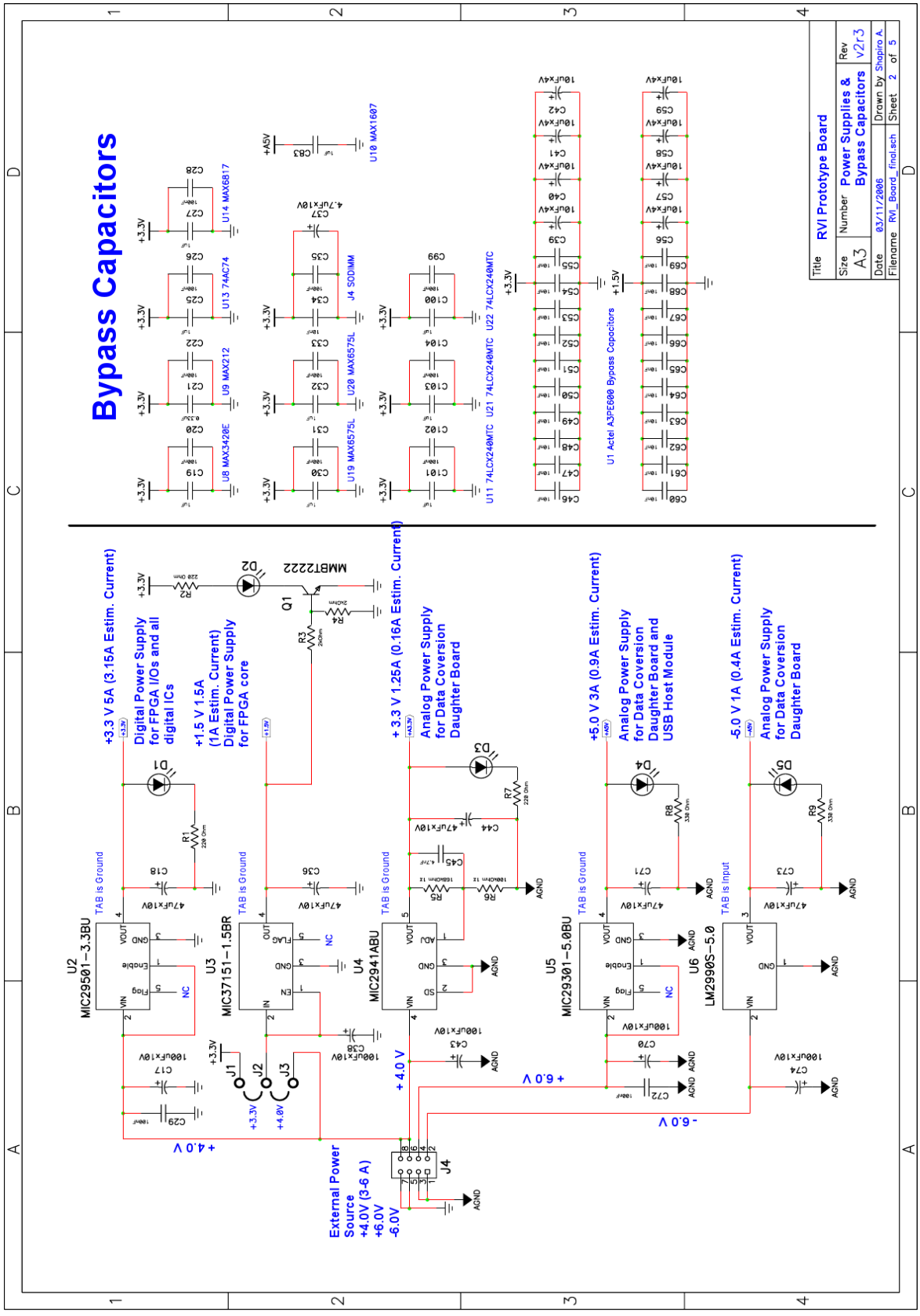

Figura I.1: Alimentación.

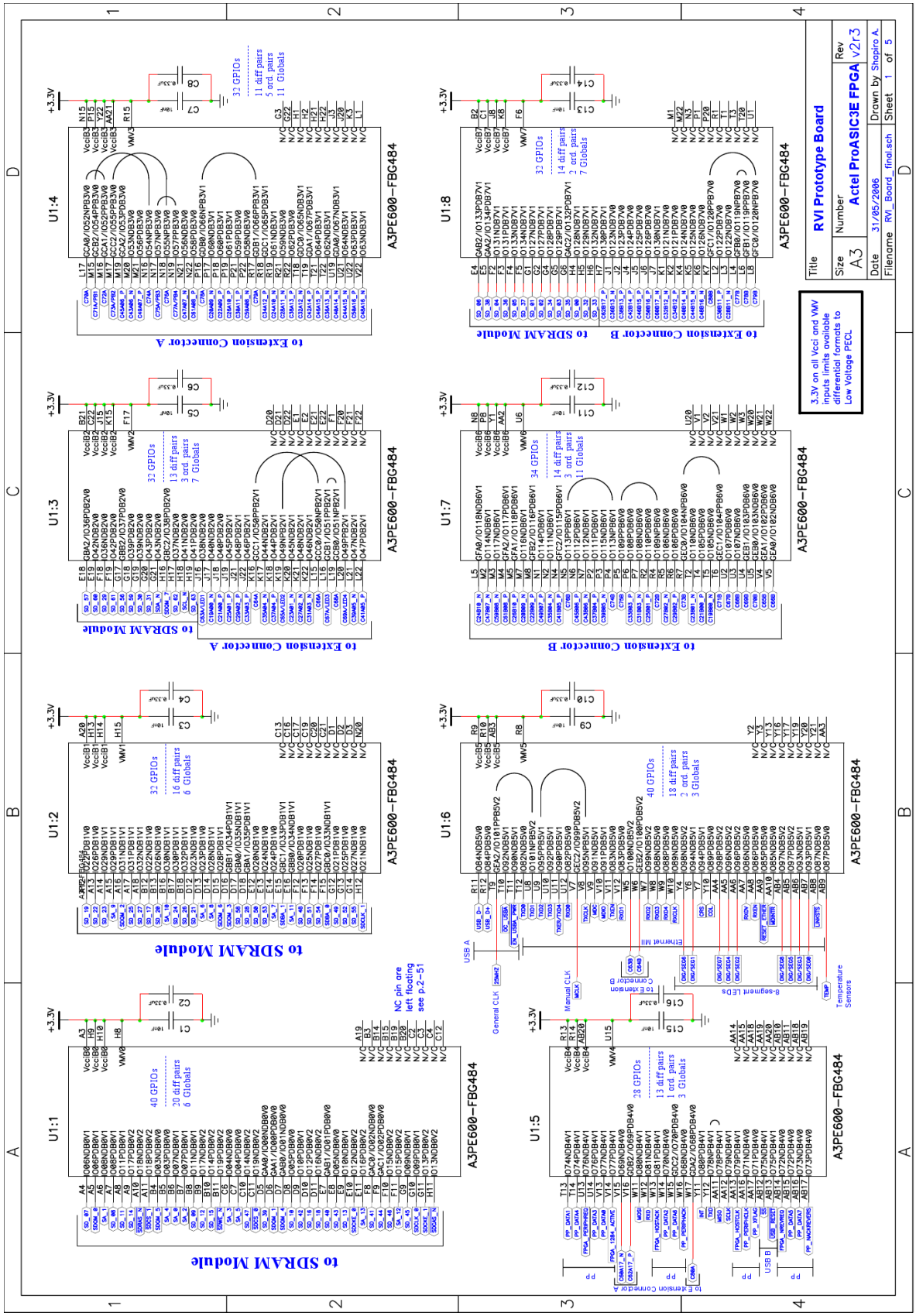

Figura I.2: FPGA.

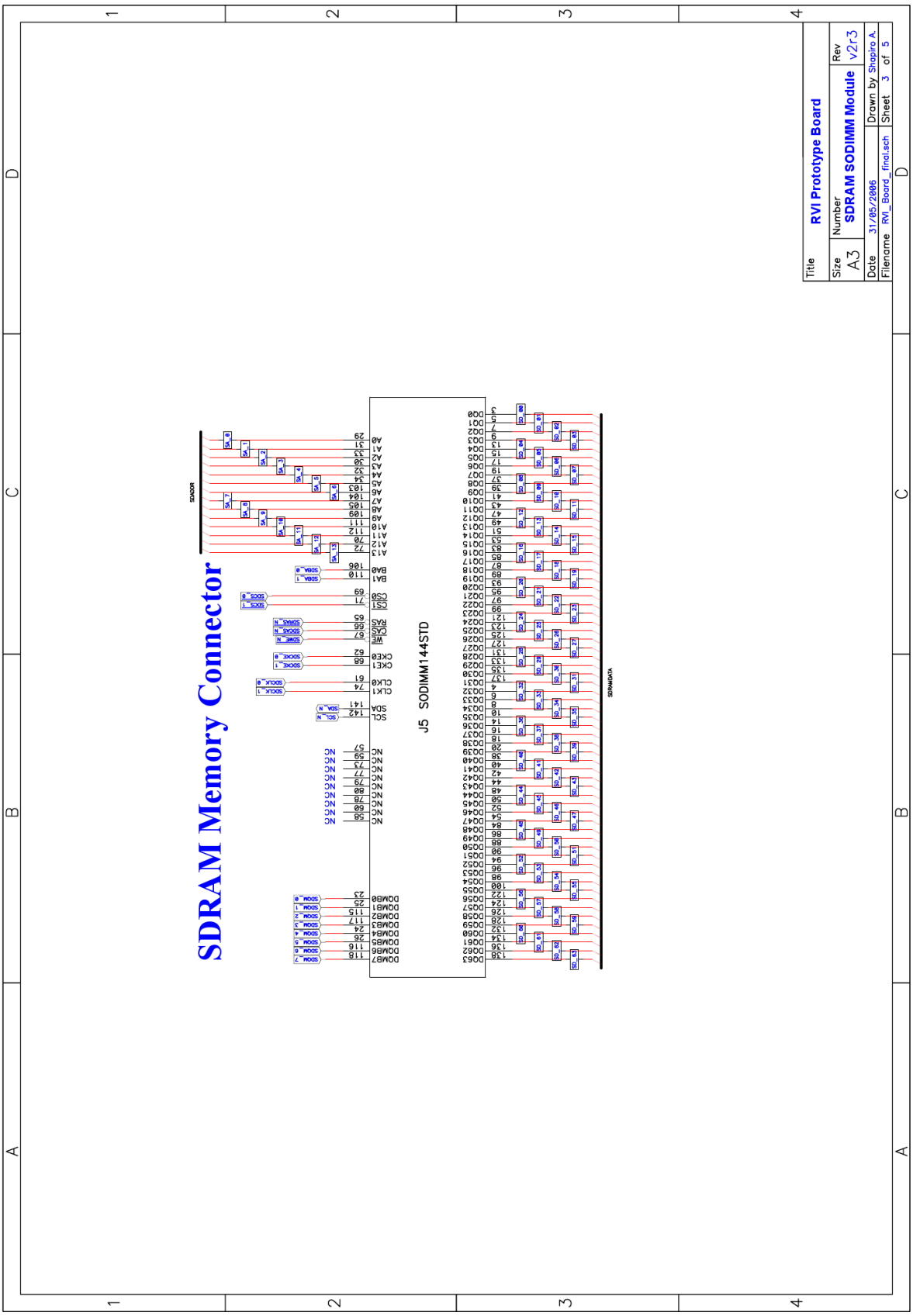

Figura I.3: Conector de memoria SDRAM.

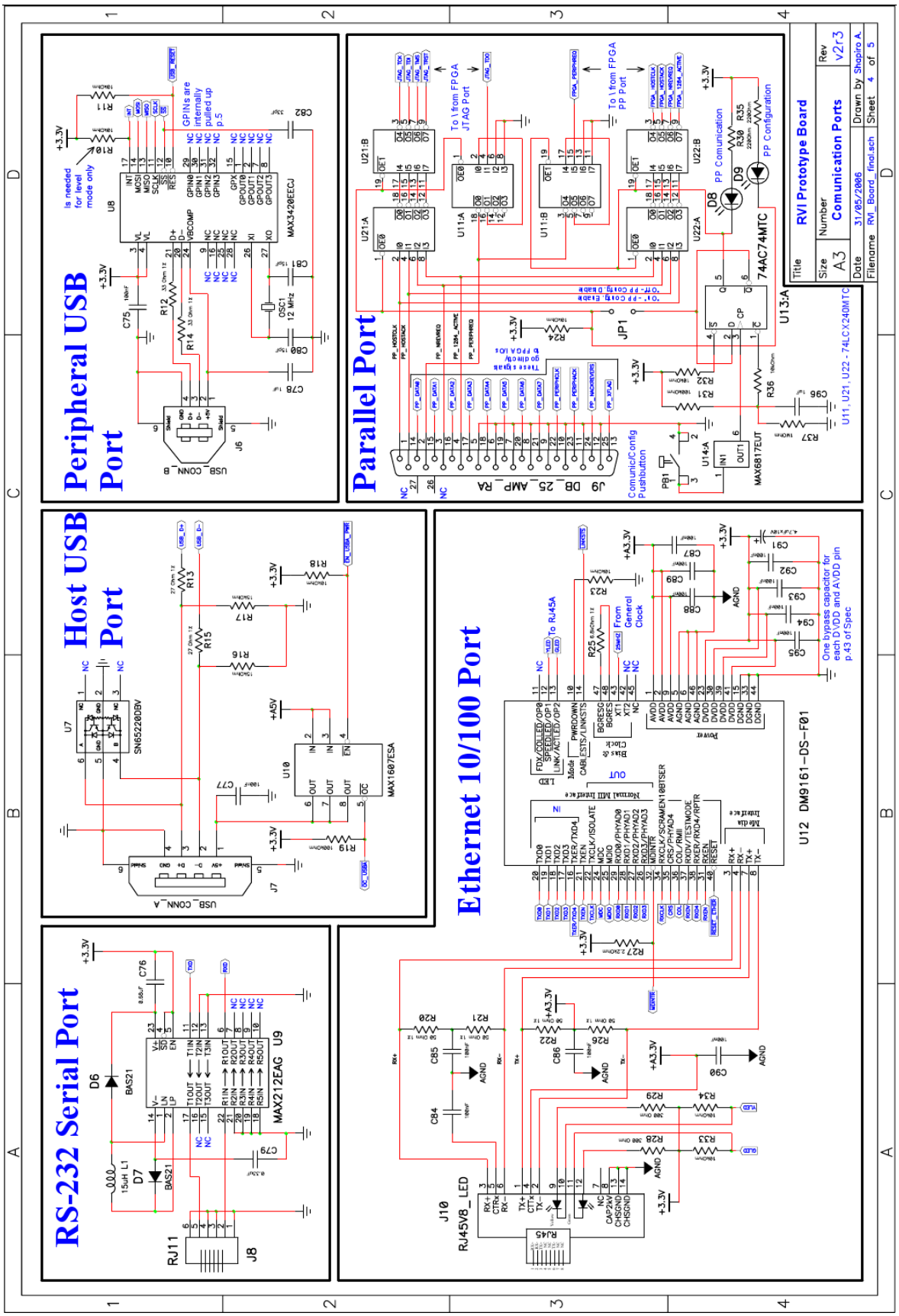

Figura I.4: Puertos de comunicación.

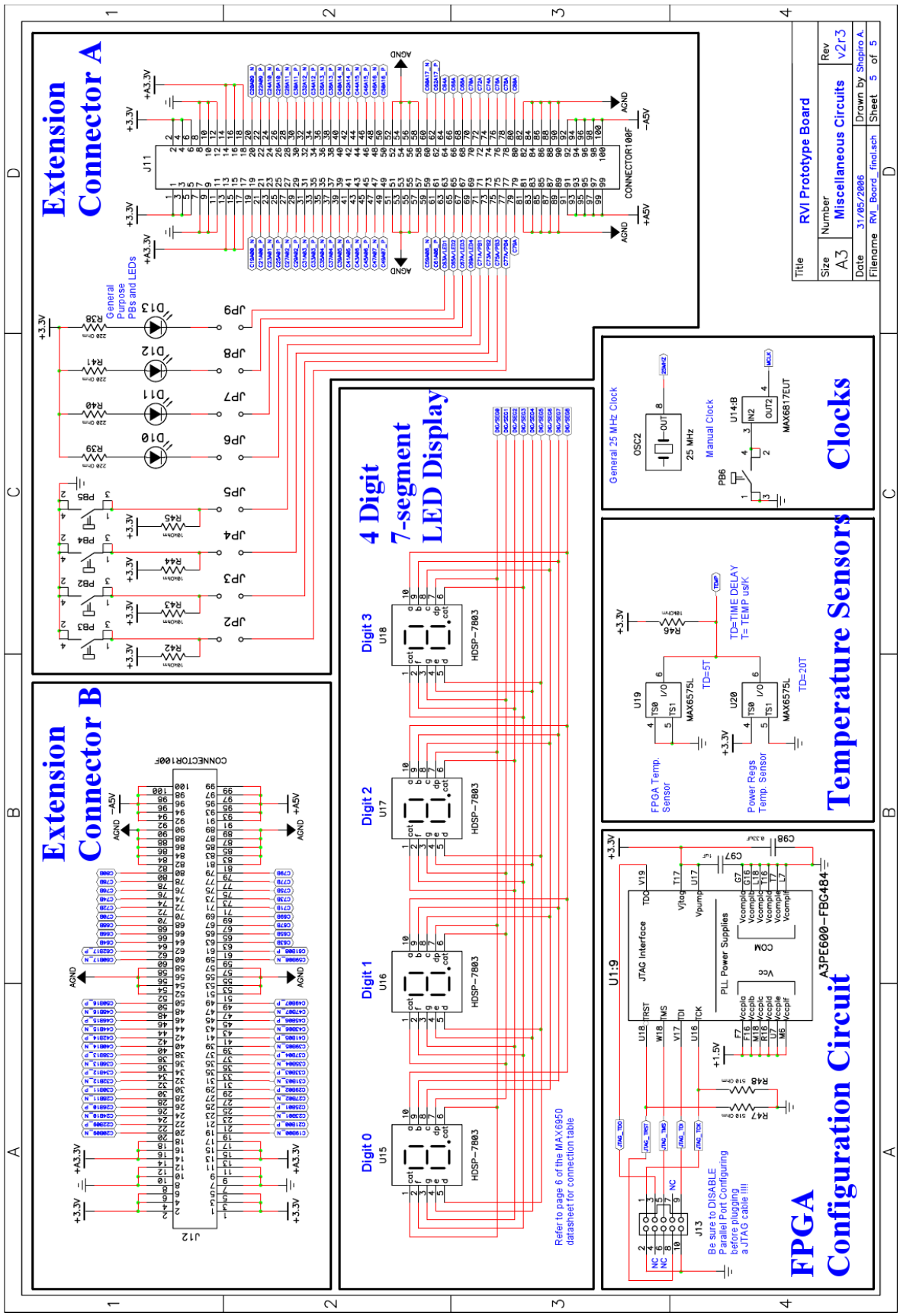

Figura I.5: Conectores, controles, visualizadores, pulsadores y sensores.

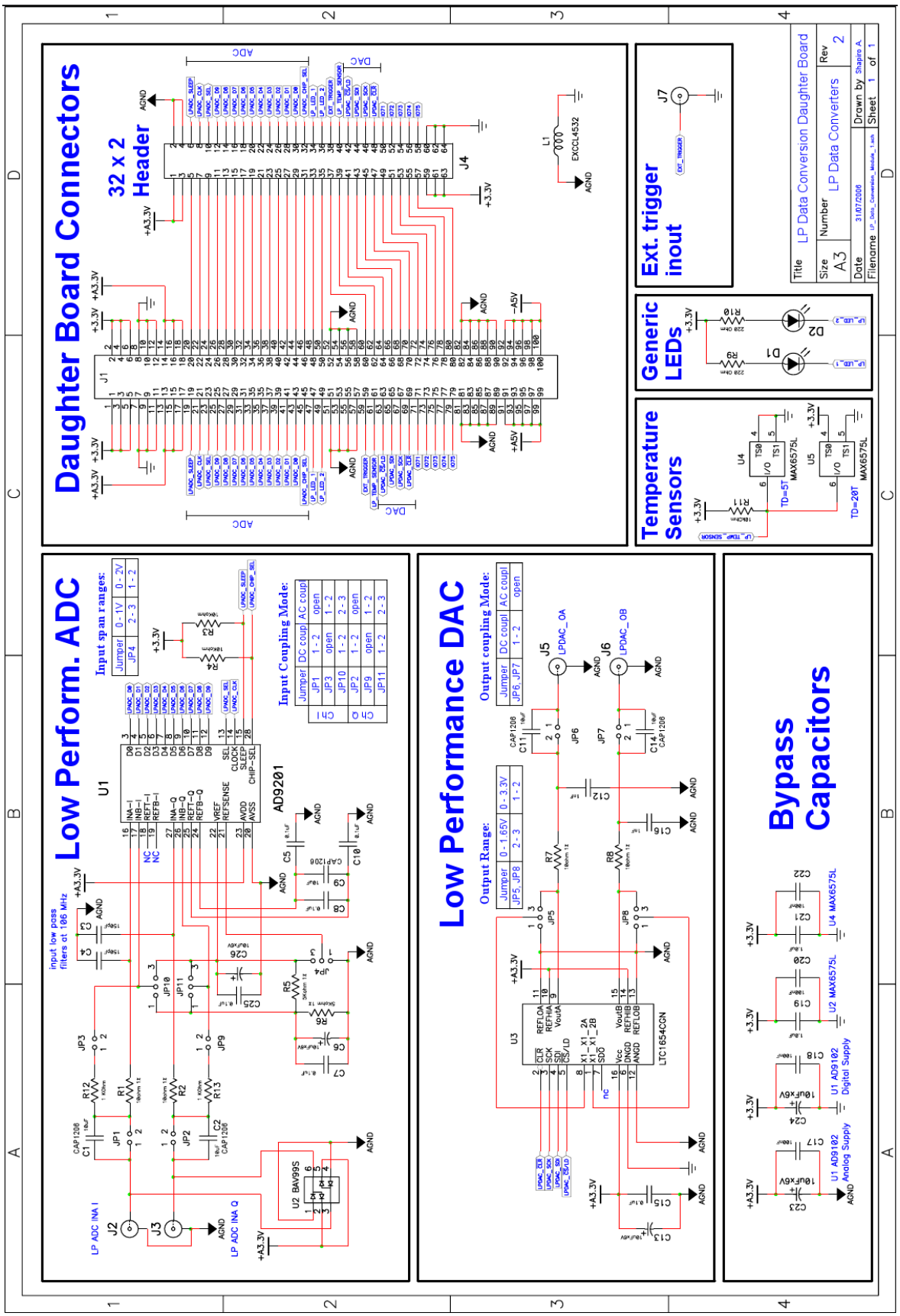

Figura I.6: Placa de expansión LP Data Convertion Daughter Board.

Anexo II. Descripción de la interfaz WISHBONE

En este anexo se realiza una breve descripción de la interfaz de interconexión de núcleos de propiedad intelectual (IP cores) utilizada en este trabajo. Para mayor detalle, puede consultarse la especificación WISHBONE [30].

II.1 ¿Qué es?

Anteriormente, los módulos internos de los diseños desarrollados para implementar en un chip utilizaban esquemas de interconexión no estándar, lo que los hacía difíciles de integrar en otros sistemas. Posteriormente, fueron creadas especificaciones para la interfaz de interconexión de éstos módulos.

WISHBONE es una arquitectura de interconexión de núcleos de propiedad intelectual portables para sistemas embebidos. Su objetivo principal es promover la reutilización de diseños, a través de la creación de una interfaz común entre IP cores. No requiere de herramientas de desarrollo ni de una tecnología de hardware específicas.

Actualmente, la especificación es administrada por la organización OpenCores y mantenida en forma abierta y de libre utilización. Puede ser usada en cualquier diseño y producción de sistemas embebidos, sin derechos de autor ni obligaciones comerciales hacia OpenCores.

II.2 Características importantes

- Interfaz de hardware simple y compacta, que requiere pocas compuertas lógicas.
- Soporta diseños estructurales, indispensable en equipos de diseño de proyectos relativamente grandes.
- Anchos de bus de datos y de direcciones modulares.
- Diferentes métodos de interconexión de núcleos soportadas, incluyendo: punto a punto, bus compartido, crossbar switch y flujo de datos.
- Permite el uso de etiquetas definidas por el usuario. Son útiles para agregar información al bus de direcciones, al bus de datos o al ciclo del bus. Por ejemplo, pueden ser útiles para identificar información como: ciclos de datos, bits de paridad o de corrección de errores, vectores de interrupción, operaciones de control de caché, etc.
- Usa una arquitectura MAESTRO / ESCLAVO.
- Tiene capacidades de multiprocesamiento (multi MAESTRO).
- Diseño síncrono, con especificación de tiempos variable.
- El protocolo de handshake permite a cada IP core regular su velocidad de tranferencia de datos.
- Independiente de la tecnología de hardware (FPGA, ASIC, etc.). Independiente de la tecnología de las herramientas de síntesis, routeo y disposición. Independiente de los métodos de testeo y simulación.
- Fácil de entender y de implementar. Requiere un mínimo de documentación estándar

(llamada *WISHBONE DATASHEET* en la especificación) y permite a los usuarios de los IP cores una rápida evaluación e integración.

- Es de libre utilización.
- Es la especificación recomendada para la distribución de IP Cores de código abierto recomendada por la organización OpenCores.

II.3 Elementos descritos en la especificación

En la Figura II.1 se muestran todos los componentes definidos por la especificación. A modo de ejemplo, se muestra la conexión directa entre solo dos interfaces, una MAESTRO y otra ESCLAVO, aunque, en el caso de utilizarse más interfaces, está permitida la incorporación de una arquitectura de interconexión entre los componentes definida por el usuario, llamada INTERCON dentro de la especificación. Se incluyen los nombres de los buses de datos de datos y de dirección (en MAESTRO: DAT_I, DAT_O, ADR_O), del las líneas de handshake (en MAESTRO: WE_O, SEL_O, STB_O, ACK_I, CYC_O) y las etiquetas definidas por el diseñador (TAGN_O, TAGN_I). El módulo SYSCON controla las señales de reloj y de reinicio, y no está especificado (definido por el usuario).

Está permitido el uso de señales que no correspondan con ninguna de las especificadas, pero debe aclararse que no pertenecen a la interfaz WISHBONE dentro de las hojas de datos del IP Core.

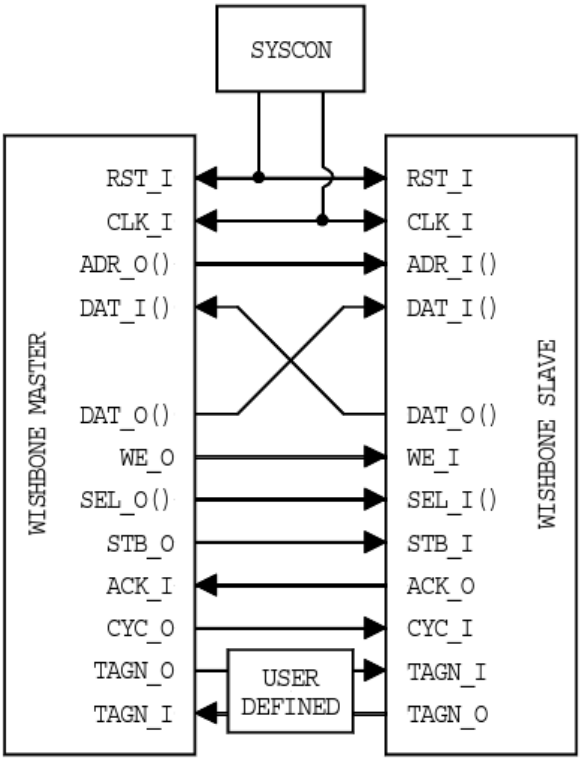

Figura II.1: Componentes de WISHBONE.

Anexo III. Asignación de direcciones internas

A continuación, se describe cómo están asignadas las direcciones internamente en el módulo de control descrito en el Capítulo 4..

III.1 Tabla de asignaciones

En la Tabla III.1 se muestra cómo están asignadas las direcciones internas del módulo, si son de lectura o escritura y el nombre otorgado a cada uno de los bits.

Dir.	Nombre	Modo	15	14	13	12	11	10	9	8
			7	6	5	4	3	2	1	0
0	RunConf_R	L/E						TScal04	TScal03	TScal02
			TScal01	TScal00	TScalEn	TrCh	TrSlope	TrOn	Cont	Start
1	Channels_R	L/E	[R]	[R]	[R]	[R]	[R]	[R]	[R]	[R]
			[R]	[R]	[R]	[R]	[R]	[R]	RCh01	RCh00
2	BuffSize_R	L/E			BuffS13	BuffS12	BuffS11	BuffS10	BuffS09	BuffS08
			BuffS07	BuffS06	BuffS05	BuffS04	BuffS03	BuffS02	BuffS01	BuffS00
3	TrigLvl_R	L/E							TrLvl09	TrLvl08
			TrLvl07	TrLvl06	TrLvl05	TrLvl04	TrLvl03	TrLvl02	TrLvl01	TrLvl00
4	TrigOff_R	L/E		TrOff14	TrOff13	TrOff012	TrOff11	TrOff10	TrOff09	TrOff08
			TrOff07	TrOff06	TrOff05	TrOff04	TrOff03	TrOff02	TrOff01	TrOff00
5	ADCConf_R	L/E					ADPD11	ADPD10	ADPD09	ADPD08
			ADPD07	ADPD06	ADPD05	ADPD04	ADPD03	ADPD02	ADPD01	ADPD00
8	Data_O	L	StatF01	StatF00	[R]	[R]	[R]	DCh00	Dat00	Dat00
			Dat00	Dat00	Dat00	Dat00	Dat00	Dat00	Dat00	Dat00
9	Error_O	L	[R]	[R]	[R]	[R]	[R]	[R]	[R]	[R]
			[R]	[R]	[R]	[R]	[R]	[R]	[R]	[R]

Tabla III.1: Asignaciones de direcciones.

L: Lectura

E: Escritura

[R]: Reservado para futuros diseños.

Espacios blancos: Sin funcionalidades asignadas. Actualmente no se pueden escribir ni leer.

III.2 Descripción de las asignaciones

Se describe la utilidad de las posiciones de memoria descritas en la tabla. Los bits son activos por alto, salvo que se indique lo contrario.

RunConf_R

Registro de configuración general. Aquí se activan y desactivan las características que estarán presentes durante el funcionamiento del osciloscopio.

Start: Inicia de la conversión. Es puesto automáticamente en cero cuando se inicia la adquisición.

Cont: Activa el modo continuo.

TrOn: Activa la funcionalidad de trigger.

TrSlope: Cuando está activo el trigger, '1' indica pendiente positiva y '0' indica pendiente negativa.

TrCh: Selecciona el canal que utilizará el trigger. '0' canal A, '1' canal B.

TScalEn: Activa la funcionalidad de selección de escala de tiempo. Cuando está desactivado, se utiliza la frecuencia del módulo de adquisición.

TScal04-00: Cuando está activada, indica el valor utilizado en el módulo de selección de escala de tiempo, según se describe en la ecuación (4.2).

Channels_R

Selección de canales activos. Los bits reservados están pensados para ser utilizados en nuevos canales definidos internamente.

RCh01-00: Cada posición activa un canal diferente. Por ejemplo, “10” activa el canal B y desactiva el canal A. La desactivación de todos los canales no está contemplada.

BuffSize_R

Indica el tamaño del buffer utilizado en la memoria de doble puerto, es decir, la posición de memoria máxima utilizada.

BuffS13-00: Indican el tamaño del buffer.

TrigLvl_R

Nivel del trigger

TrLvl09-00: Indican el nivel utilizado para el trigger.

TrigOff_R

Valor de offset utilizado en el trigger.

TrOff14-00: Posición de memoria relativa desde la que se activó el trigger, desde la que se comenzarán a mostrar los valores en el visualizador del software. Admite valores negativos en complemento a dos y su valor absoluto debe ser menor que el tamaño del buffer.

ADCConf_R

Registro de configuración del módulo de adquisición.

ADPS08-00: Si está activada, es el valor de división de frecuencia del reloj del conversor, según la ecuación (4.1).

ADPSEn: Activa la división de frecuencia del del reloj.

ADSleep: Activa la señal de *Sleep* hacia el conversor.

ASCS: Activa la señal *Chip Select* hacia el conversor.

Data_O

De solo lectura. Cuando el osciloscopio está en funcionamiento, se debe leer continuamente esta dirección para obtener los datos almacenados en el buffer y

conocer el estado de la adquisición.

Dat09-00: Estos son los valores obtenidos.

DCh00: El canal al que corresponde el valor indicado en Dat09-00.

StatF01-00: Indican el estado de la adquisición, como se indica en la Tabla III.2. La indicación de buffer par o impar se utiliza en el modo continuo. Un buffer impar significa que se está leyendo el buffer una cantidad de veces impar. De modo similar para el buffer par. Por ejemplo, la primera vez que se lee el contenido del buffer, se indicará que es impar. Luego de que el buffer es leído completamente, la segunda vez se indicará que es par, y así consecutivamente. Este valor es utilizado por el software para conocer los valores desde los cuales comenzar a graficar la señal. Los valores reservados serán útiles en el caso de que sean incluidos nuevos canales o bits de estado.

Valor (binario)	**Significado**
00	Parado. Esperando ser inicializado.
01	Funcionando. Leyendo buffer impar.
11	Funcionando. Leyendo buffer par.
10	Parado. Hubo un error. (No implementado)

Tabla III.2: Descripción del indicador de estado.

Error_O

Esta posición es de solo lectura y no está implementada. Está reservada para utilizarse en caso de que se incluya un módulo de detección de errores interno, para indicar ciertos valores de depuración o códigos de error.

Referencias

[1] Andrés Cicuttin, María Liz Crespo, Alexander Shapiro, Nizar Abdallah, *A Block-Based Open Source Approach for a Reconfigurable Virtual Instrumentation Platform Using FPGA Technology*, The IEEE International Conference on Reconfigurable Computing and FPGAs, sep. 2006.

[2] Harold Goldberg, *What is Virtual Instrumentation*, IEEE Instrumentation & Measurement Magazine, dic. 2000, pp. 10-13.

[3] Viktor Smieško, Karol Kováč, *Virtual Instrumentation And Distributed Measurement Systems, Journal of Electrical Engineering*, Vol. 55, No. 1-2, 2004, pp 50-56.

[4] G.R. Tsai, M.C. Lin, G.S. Sun, Y.S. Lin, *Single chip FPGA-based reconfigurable instruments*, The International Conference on Reconfigurable Computing and FPGAs, sep. 2004.

[5] J.W. Hsieh, G.R. Tsai, M.C. Lin, *Using FPGA to implement a n-channel arbitrary waveform generator with various add-on functions*, 2nd IEEE International Conference on Field-Programmable Technology, dic. 2003, pp. 296–298.

[6] M.J. Moure, M.D. Valdés, E. Mandado, *Educational Applications of Reconfigurable Hardware Based Virtual Instruments*, 29th ASEE/IEEE Frontiers in Education Conference, nov. 1999, pp. 12C6/17.

[7] M.D. Valdés, M.J. Moure, C. Quintáns, E. Mandado, A Data Acquisition Reconfigurable Coprocessor for Virtual Instrumentation Applications, International Conference on Field-Programmable Logic and Applications, sep. 2003, pp. 1107-1110.

[8] *Conceptos Básicos de los Osciloscopios Analógicos y Digitales*, Tektonix Inc., 1993. http://www.tek.com (consultada el 21/09/2009).

[9] William D. Cooper, Albert D. Helfrick, *Instrumentación electrónica moderna y técnicas de medición*, Prentice Hall, 1992.

[10] Chandan Bhunia, Saikat Giri, Samrat Kar, Sudarshan Haldar, Prithwiraj Purkait, *A Low-Cost PC-Based Virtual Oscilloscope*, IEEE Transactions On Education, Vol. 47, No. 2, may. 2004, pp 295-299.

[11] Pablo Hoffman, Martín Szmulewicz, *Osciloscopio USB*, trabajo final de ingeniería Universidad ORT Uruguay, 2006.

[12] Stephen Pickett, *An Alternative Oscilloscope*, proyecto libre publicado en OpenCores, 2005. http://www.opencores.com (consultada el 22/09/2009).

[13] fpga4fun.com, *Hands-on A digital oscilloscope.* http://www.fpga4fun.com (consultada el 22/09/2009).

[14] Charles Cheung, *FPGA Digital Oscilloscope*, proyecto final del curso ECE 5760 de la universidad Universidad Cornell, 2007. http://www.ece.cornell.edu/ (consultada el 20/09/2009).

[15] Schuster Andreas, *FPGA Based Oscilloscope with Java GUI*, proyecto libre publicado en SourceForge, dic. 2008. http://skol.sourceforge.net/ (consultada el 20/09/2009).

[16] Andrés Cicuttin, María Liz Crespo, Alexander Shapiro, Nizar Abdallah, *Building an Evolvable Low-Cost HW/SW Educational Platform – Application to Virtual Instrumentation*, IEEE International Conference on Microelectronic Systems Education, 2007.

[17] D.J. Smith, *VHDL and Verilog compared and contrasted plus modeled example written in VHDL, Verilog and C*, 33rd Design Automation Conference Proceedings, jun. 1996.

[18] Skahill K., *VHDL for Programmable Logic*, Addison-Wesley,1996.

[19] Actel Corporation. http://www.actel.com/ (consultada el 22/09/2009).

[20] Rudolf Usselmann, *OpenCores SoC Bus Review Rev. 1.0*, organización OpenCores, ene. 2009. http://www.opencores.org (consultada el 21/09/2009).

[21] *IEEE Standard Signaling Method or a Bidirectiona Parallel Peripheral Interface for Personal Computers*, IEEE Std 1284-2000, oct. 2000. http://standards.ieee.org/ (consultada el 21/09/2009).

[22] A. Trapanotto, D. Brengi, S. Tropea, *Puente IEEE1284 en modo EPP a bus Wishbone*, Southern Programmable Logic Conference, mar. 2006.

[23] Craig Peacock, *Interfacing the Enhanced Parallel Port*, publicada en el sitio web Beyond Logic, 2002. http://www.beyondlogic.org/ (consultada el 21/09/2009).

[24] Qt – A cross-platform application and UI framework. http://qt.nokia.com/ (consultada el 21/09/2009).

[25] Qwt - Qt Widgets for Technical Applications. http://qwt.sourceforge.net/ (consultada el 21/09/2009).

[26] *Oscilloscope Guide*, corporación BK PRECISION. http://www.bkprecision.com/ (consultada el 29/09/09).

[27] Craig Peacock, *PortTalk - A Windows NT I/O Port Device Driver*, publicada en el sitio web Beyond Logic, 2002. http://www.beyondlogic.org/ (consultada el 21/09/2009).

[28] Sitio web Geek Hideout. http://www.geekhideout.com/contact.shtml (consultada el 05/10/2009).

[29] *SRAM and FIFO Memories in Actel's Low-Power Flash Devices*. Actel Corporation. http://www.actel.com/ (consultada el 22/09/2009).

[30] *WISHBONE System-on-Chip (SoC) Interconection Architecture for Portable IP Cores*, OpenCores Organization, sep. 2002. http://www.opencores.org (revisada el 26/06/09).