Universidad del País Vasco
Euskal Herriko Unibertsitatea

Facultad de Informática

Grado de Ingeniería Informática

▪ Proyecto Fin de Grado ▪

Ingeniería de Computadores

Marco-Reloj-Despertador basado en FPGA

David Anda Lorza

Septiembre del 2015

Director de proyecto:

Andoni Arruti Illarramendi

informatika fakultatea
facultad de informática

Agradecimientos

La finalización del trabajo fin de grado supone un gran punto de inflexión en el que merece la pena echar la vista atrás para darse cuenta del camino recorrido y de todo lo aprendido en él. Supone la consecución de un objetivo marcado hace muchos años atrás, la conclusión del trabajo y sacrificio de todos estos años.

Para llegar hasta aquí, es necesario contar con gente que te apoye y te ayude constantemente, por esto y por muchas más cosas tengo que dar las gracias a mis padres, Ander y Blanca, y a mi hermana Eider, así como al resto de mi familia.

También gracias a mi director de proyecto, Andoni, por sus consejos durante todo el trabajo y su ayuda incondicional. El buen trato recibido siempre ha sido de gran valor; con él aprendí los primeros pasos para empezar esta aventura en la que se ha convertido el proyecto.

Gracias a todos mis amigos, incluidos aquellos que empezaron siendo compañeros de carrera, ya que siempre me han animado cuando más lo necesitaba. Y, por supuesto, gracias a mi novia Esti por haber compartido estos años de tanto sufrimiento y a la vez de tantas alegrías; espero haberte ayudado tanto como tú lo has hecho.

Eskerrik asko!

Resumen

Gordon E. Moore, cofundador de Intel, predijo en una publicación del año 1965 que aproximadamente cada dos años se duplicaría el número de transistores presentes en un circuito integrado, debido a las cada vez mejores tecnologías presentes en el proceso de elaboración [1]. A esta ley se la conoce como Ley de Moore y su cumplimiento se ha podido constatar hasta hoy en día.

Gracias a ello, con el paso del tiempo cada vez se presentan en el mercado circuitos integrados más potentes, con mayores prestaciones para realizar tareas cada vez más complejas. Un tipo de circuitos integrados que han podido evolucionar de forma importante son los dispositivos de lógica programable, circuitos integrados que permiten implementar sobre ellos las funciones lógicas deseadas.

Hasta hace no muchos años, dichos dispositivos eran capaces de incorporar circuitos compuestos por unas pocas funciones lógicas, pero gracias al proceso de miniaturización predicho por la Ley de Moore, hoy en día son capaces de implementar circuitos tan complejos como puede ser un microprocesador; dichos dispositivos reciben el nombre de FPGA, siglas de *Field Programmable Gate Array*.

El presente proyecto tiene como objetivo construir un marco de fotos digital con reloj y despertador programable, valiéndose para ello de la FPGA *Cyclone II* de *Altera* y una pantalla táctil de la casa *Terasic*. Con este fin, se documentará en primera instancia los dispositivos a utilizar con sus características y posibilidades que plantean, para pasar posteriormente al diseño de la aplicación y su implementación e integración en la placa para comprobar su correcto funcionamiento.

Índice general

Índice de figuras y tablas

Capítulo 2

Capítulo 3

1

Introducción

En este primer capítulo se hará un breve resumen de los dispositivos utilizados para introducir las ideas básicas de los mismos. Por otro lado, se hablará de la motivación que hace nacer este proyecto, así como los objetivos que se pretenden cumplir durante su elaboración.

1.1. Descripción general

Para los que no conozcan la placa de desarrollo y educación DE2, diremos que fue creada por la empresa *Altera* en el año 2004 como vehículo para el aprendizaje de la lógica digital y manejo sobre las FPGAs, del inglés, *Field Programmable Gate Array*. Este panel fue diseñado para los profesores y alumnos de ambiente universitario con intención de facilitar el aprendizaje mediante una amplia gama de ejercicios y tareas que pueden desarrollarse gracias a la cantidad de opciones que ofrece el dispositivo; desde tareas sencillas para ilustrar conceptos fundamentales de la lógica digital, hasta otras más complejas que pueden ayudar a elaborar diseños más avanzados como el proyecto que nos ocupa.

Por otro lado, una de las características más importantes de la placa DE2, es la posibilidad que ofrece para agregarle periféricos que realicen diversas funciones controlados por la propia FPGA. En este proyecto, se ha decidido agregarle mediante un puerto de entrada/salida de propósito general o conector de expansión GPIO, una pantalla táctil que mostrará el funcionamiento del proyecto y con la que se podrá interactuar para visualizar y configurar las diferentes aplicaciones programadas.

Con ambos dispositivos interconectados, se pretende conseguir un prototipo funcional de un marco de fotos digital con reloj y despertador configurables. Para que la función del despertador o alarma se escuche como se espera en el momento de la activación, también se deberá agregar, como cabría esperar, unos cascos o altavoces a la salida de audio de la placa DE2.

1.2. Motivación

La principal razón para llevar a cabo este proyecto, radica en demostrar el potencial que se oculta tras los dispositivos de lógica programable mediante un prototipo atractivo y de uso sencillo que integre la tecnología táctil que se aplica a cada vez más aparatos electrónicos.

Con este trabajo se pretende profundizar más en una tecnología como las FPGAs, cada vez con más auge y cuya aplicación se extiende desde el procesamiento digital de señales, pasando por sistemas de visión para computadoras y reconocimiento de voz, hasta sistemas aeroespaciales y de defensa. Se trata de una plataforma con muchas posibilidades en el futuro y el hecho de interactuar con ella para este proyecto, facilita la comprensión de algunas de sus posibilidades y abre la puerta a posibles soluciones basadas en FPGAs. Cabe destacar que su uso en otras áreas es cada vez mayor, sobre todo en aquellas aplicaciones que requieren un alto grado de paralelismo.

La motivación por tanto, no es crear un dispositivo comercial con intención de superar los productos actuales en el mercado, se trata de aprender a elaborar un producto parecido a los ya existentes, conociendo los entresijos y problemas que conlleva; creando una base para que en un futuro se pueda ampliar y mejorar ofreciendo ideas innovadoras. A lo largo de este trabajo habrá que enfrentarse a problemas reales semejantes a los que se puede encontrar un ingeniero a lo largo de su vida laboral, lo que permitirá aplicar lo aprendido a lo largo de los años de formación universitaria.

Asimismo, llama la atención el tipo de programación que se realiza en las FPGA, que dista de la programación software habitual y que resulta, a mi modo de ver, interesante. En estos dispositivos compuestos por cientos de miles de celdas, la programación consiste en configurar dichas celdas con una función específica, ya sea como memoria, como multiplexor o con una función lógica tipo AND, OR, XOR, haciendo uso de un lenguaje de descripción de hardware HDL. Entre los HDL más conocidos se encuentran los utilizados en este proyecto, VHDL y Verilog. Por lo tanto, la función del programador es describir el hardware que tendrá la FPGA, definiendo la función lógica que realizará cada uno de los bloques lógicos configurables e interconectándolos entre sí para obtener el resultado deseado.

1.3. Objetivos

A continuación se detallan los objetivos que se plantean durante el desarrollo de este Proyecto Fin de Grado:

- Documentar los dispositivos utilizados y posibilidades que plantean.
- Profundizar en el aprendizaje del lenguaje de descripción de hardware VHDL utilizando la herramienta de diseño *Quartus II.*
- Aprender los conceptos básicos del lenguaje de descripción de hardware Verilog, para interpretar y modificar archivos fuente.
- Adaptar los archivos fuente de aplicaciones sencillas a la tarea a desarrollar.
- Diseñar e implementar los módulos necesarios para elaborar el prototipo.
- Interconectar y configurar cada módulo para obtener la aplicación completa.
- Construir un prototipo de un marco de fotos digital con reloj y despertador mediante la placa basada en FPGA *Altera* DE2 y la pantalla táctil.
- Elaborar el manual de usuario e instalación del prototipo.

2

Estado del arte

En este capítulo se pretende explicar brevemente la historia de los circuitos integrados y su evolución más reciente hasta llegar al dispositivo utilizado. Por lo tanto, se hará hincapié en la tarjeta de evaluación *Altera* DE2 y su FPGA *Altera Cyclone II*, así como el resto de periféricos necesarios para el correcto funcionamiento del prototipo y las herramientas de diseño utilizadas.

2.1. Historia de los circuitos integrados

El desarrollo de la microelectrónica ha permitido la aparición de diferentes tipos de circuitos integrados digitales, que pueden ser clasificados en tres grandes grupos atendiendo a su metodología de diseño y fabricación: circuitos integrados estándar, circuitos integrados de aplicación específica o ASICs y dispositivos de lógica programable o PLDs [3].

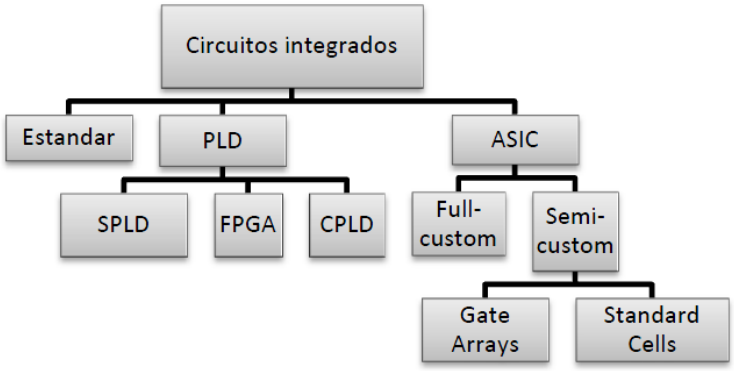


Figura 2.1: Clasificación de los circuitos integrados

Los circuitos integrados estándar poseen características lógicas y eléctricas perfectamente definidas. Este tipo de circuitos presenta la ventaja de su reducido coste y su gran fiabilidad debido a que se fabrican en grandes series. Sin embargo, para poder realizar circuitos un tanto complejos se necesitarán un gran número de circuitos de este tipo. Además, es muy común ver copias no autorizadas de los mismos.

Los ASICs son circuitos que se diseñan para ofrecer una funcionalidad específica que es determinada a la hora de fabricar el dispositivo; dichos dispositivos pueden clasificarse según su tecnología como *full-custom* o *semi-custom*.

En los circuitos *full-custom*, el diseñador tiene control total en el diseño de los dispositivos. Son los más rápidos y eficientes, pero son los que requieren más tiempo de diseño y tienen mayores costes.

Otros son los *semi-custom*, dan cierta libertad de diseño, pudiendo, por ejemplo, especificar las conexiones que deben ser realizadas entre los transistores. Dentro de este grupo pueden distinguirse dos tipos: *Gate Arrays* y *Standard Cells*. En la tecnología *Gate Arrays*, el fabricante pone a disposición del diseñador múltiples transistores tanto de canal n como de canal p, y es éste quien deberá especificar las conexiones de los transistores. Dichas conexiones se realizan en las últimas fases del proceso de fabricación, diseñando las capas de metalizaciones. En los *Standard Cells*, en cambio, el fabricante pone a disposición del diseñador una serie de librerías de componentes digitales básicos, denominados celdas y el diseñador especifica las celdas que requiere, así como las conexiones entre ellas [5].

En contraposición a los ASICs, existen los dispositivos programables, dispositivos que debido a su diseño no implementan una función concreta, si no que pueden ser programados, o más bien, configurados, para lograr el objetivo. En estos dispositivos de lógica programable o PLDs, el fabricante dispone de una gran cantidad de circuitos básicos en un sustrato de silicio, que pueden utilizarse para diferentes funciones cambiando sus conexiones internas.

Los PLDs, aparecieron alrededor de la década de 1970, empezando por los SPLDs "*Simple Programmable Logic Device*", cuya estructura interna está formada por un conjunto de matrices de puertas AND y puertas OR. Dentro del grupo de los dispositivos de lógica programable, podemos encontrar las PROM "*Programmable Read Only Memory*", las PLA "*Programmable Logic Array*" o las PAL "*Programmable Array Logic*", dispositivos que pueden ser programados para realizar una función lógica en base a una suma de productos de las entradas [4].

El avance en el desarrollo de estos dispositivos dio lugar a la aparición de los CPLDs "*Complex Programmable Logic Device*", circuitos compuestos por una serie de PLDs con la opción de poder interconectarlos entre sí para poder implementar funciones más complejas, aunque sin llegar a los niveles de integración que se podían obtener en aquella época con una ASIC.

En 1984, la empresa *Xilinx* fue la pionera al introducir la primera FPGA en el mercado, un dispositivo parecido a los CPLDs pero con la diferencia de estar compuesta por más bloques, que aún siendo más simples que los bloques de los CPLDs, con un flujo de desarrollo más fácil estaban más cerca de la capacidad de integración de un ASIC que de un CPLD, ocupando así el vasto vacío existente entre ambas plataformas.

Tanto los CPLDs como las FPGAs, contienen múltiples copias de elementos lógicos o celdas que se distribuyen en filas y columnas en el chip. Para llevar a cabo funciones más complejas los elementos lógicos pueden conectarse mediante una red de interconexión programable. Mientras que los CPLDs tienden a tener tiempos de propagación más rápidos y más predecibles, las FPGAs ofrecen una mayor densidad de integración. Aparte de estas diferencias, la diferencia fundamental entre las FPGAs y los CPLDs es su arquitectura y la tecnología de programación de la que hablaremos en profundidad más adelante.

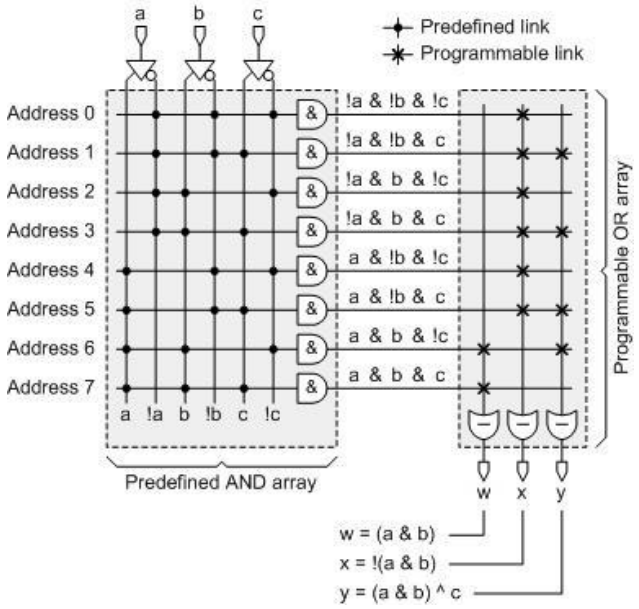


Figura 2.2: Programmable Read Only Memory - PROM

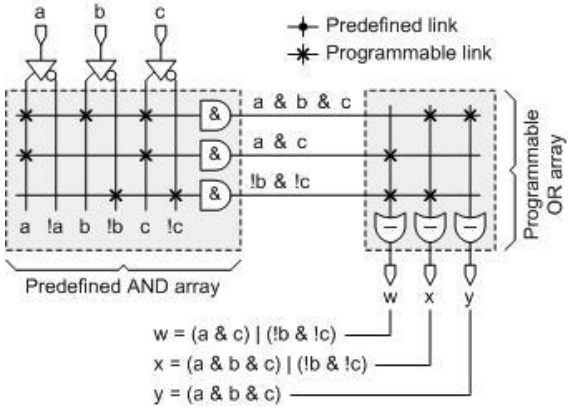


Figura 2.3: Programmable Logic Array – PLA

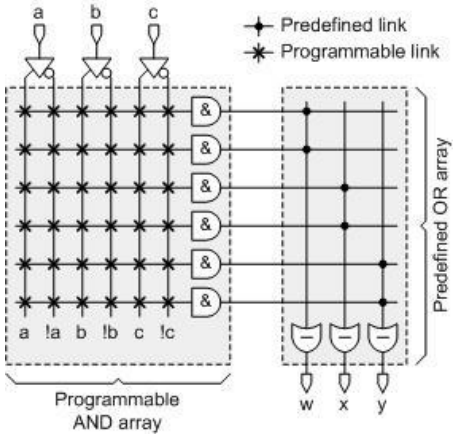


Figura 2.3: Programmable Array Logic – PAL

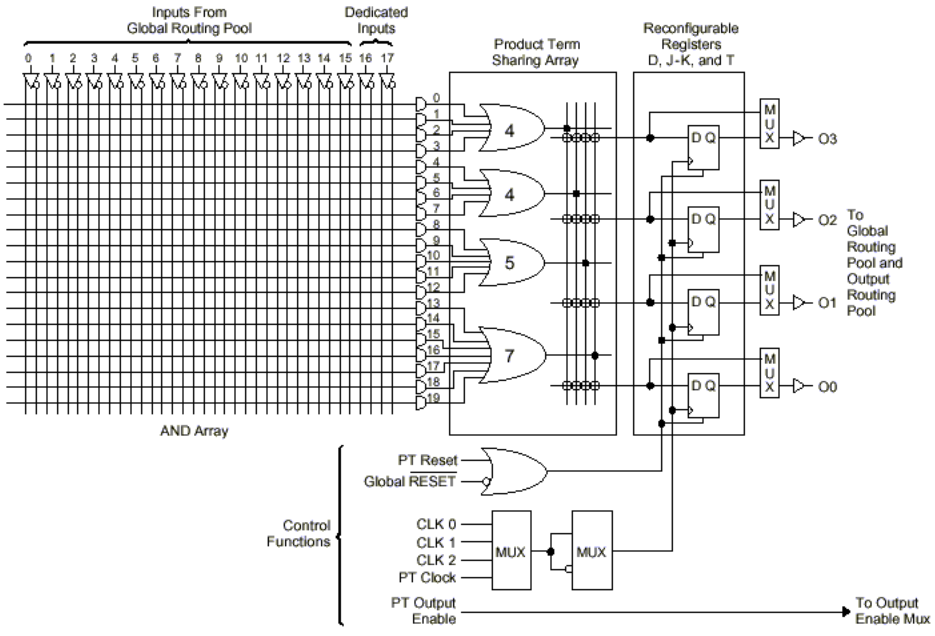


Figura 2.5: Complex Programmable Logic Device - CPLD

2.2. FPGAs

Las FPGAs [9] están formadas por una serie de bloques configurables, los cuales son capaces de implementar una función lógica y tienen una pequeña memoria, y mediante una red de interconexión, permite combinar más de un bloque básico para implementar funciones complejas que no pueden ser implementadas en un solo bloque básico. Dichos bloques, son diferentes dependiendo tanto del fabricante como del modelo concreto de la FPGA, siendo un parámetro determinante el número de bits máximo de la función que es capaz de implementar.

La forma más común de implementar una función en un bloque básico se hace mediante una tabla de valores o LUT "*Look-up Table*", guardando los valores de salida que tendría la función a implementar respecto a las entradas en una tabla. La salida de la función puede ir directamente a otro bloque básico mediante la red de interconexión o puede ser almacenada en un biestable, pudiendo implementar de esta forma circuitos síncronos con memoria.

El hecho de usar LUTs para implementar funciones es más costoso en hardware que usar directamente el circuito original que se desea implementar, pero de esta forma se consigue que el dispositivo pueda ser configurado tras su manufacturación. La tecnología usada para la configuración del dispositivo ofrece distintas características, siendo las más destacadas; la configuración, volátil o no volátil, y la usabilidad; pudiendo ser de un solo uso o de más. Si se usan fusibles o anti-fusibles, el dispositivo no ha de ser configurado cada vez, pero solo se podrá programar una única vez.

En el otro extremo están los dispositivos basados en memorias SRAM, los cuales son reprogramables tantas veces como sea necesario. Su inconveniente radica en la volatibilidad de la configuración, puesto que depende de la alimentación de dicha memoria SRAM. Esto supone que cada vez que se inicie el sistema, la reprogramación será necesaria, lo que conlleva añadir más hardware al sistema.

En un término medio están las basadas en memorias EEPROM o FLASH, pudiendo ser reprogramadas un número limitado de veces pero sin ser volátiles, por lo que la reconfiguración solo es necesaria en el caso de hacer algún cambio en la aplicación. A día de hoy el uso de las basadas en fusibles o anti-fusibles es anecdótico, siendo las más usadas las basadas en memorias SRAM y en menor medida, las basadas en memorias no volátiles como las EEPROM o FLASH.

Respecto a los fabricantes, existen diversas empresas que comercializan FPGAs, siendo dos las que copan el mercado y por lo tanto grandes competidoras entre ellas: *Xilinx* y *Altera*. Además de estas dos hay más empresas que desarrollan sus propias soluciones, aunque con un nivel de ventas mucho más bajo que las dos mencionadas anteriormente. Un par de ejemplos son *Lattice Semiconductor* y *Actel*, las cuales ofrecen productos más enfocados a las memorias no volátiles en comparación a las dos grandes, que se centran en ofrecer soluciones basadas en memorias SRAM.

En cuanto a la arquitectura, además de los bloques básicos, actualmente una FPGA incluye otros tipos de bloque en su interior con el objetivo de implementar funciones habituales de forma más eficaz, utilizando menos silicio y por lo tanto con un consumo menor y pudiendo obtener una frecuencia de funcionamiento mayor. Por ejemplo pueden tener bloques multiplicadores, de memoria RAM o para cálculos matemáticos intensivos como los usados en el procesado digital de señal, todos ellos conectados entre sí y con los bloques básicos mediante la misma red de interconexión [6].

Pese a que al principio la configuración y conexión de los bloques se hacía de forma manual, debido a la cada vez mayor complejidad y capacidad de los dispositivos, hoy en día se usan lenguajes de descripción de hardware como pueden ser Verilog o VHDL. Mediante estos lenguajes se describen los bloques hardware que se quieren implementar en la FPGA, y tras un proceso de sintetizado, generan el fichero de configuración de la FPGA, conocido comúnmente como *bitstream*, encargado tanto de configurar cada uno de los bloques del dispositivo como de la interconexión entre ellos. Estos lenguajes no pueden considerarse lenguajes de programación, ya que lo que hacen no es describir las instrucciones de un algoritmo secuencial, si no que describen bloques de hardware que se van a ejecutar de forma concurrente dentro de la FPGA.

Cuando aparecieron en el mercado su uso se limitaba a implementar sencillas maquinas de estado, tareas de procesamiento limitadas y sobre todo de comunicación entre otros dispositivos. Con el paso del tiempo sus características han mejorado mucho, pasando de tener pocos y simples bloques a contar con muchos y más complejos, por lo que su uso se ha incrementado en áreas en las que se hace un tratamiento de datos intensivo. Es por todo ello que actualmente suponen una alternativa a los ASICs, puesto que pueden implementar diseños con el equivalente a millones de puertas lógicas, con un menor tiempo de salida al mercado y con un coste económico más bajo para tiradas menores a las 100.000 unidades.

Estos dispositivos, son muy buenos para tareas de *data-flow*, pero flaquean en comparación a los procesadores convencionales en cuanto a tareas secuenciales, dado que aunque son capaces de implementarlas, el proceso de desarrollo es más costoso. Es por ello, que debido a la capacidad con las que cuentan, hoy en día son capaces de implementar procesadores completos en su interior, los cuales luego ejecutaran como si de un microprocesador al uso se trataran. Además, debido a las cada vez mayores capacidades de integración, hoy en día nos podemos encontrar con plataformas mixtas, que pueden llegar a incluir procesadores multi-núcleo ARM que operan a una frecuencia mayor que 1 GHz, junto a FPGAs de alta gama en la que implementar funciones específicas para descargar el procesador, todo ello implementando en el mismo dado de silicio. Ejemplo de esto son las arquitecturas *Zynq de Xilinx* o *Altera SoC de Altera*.

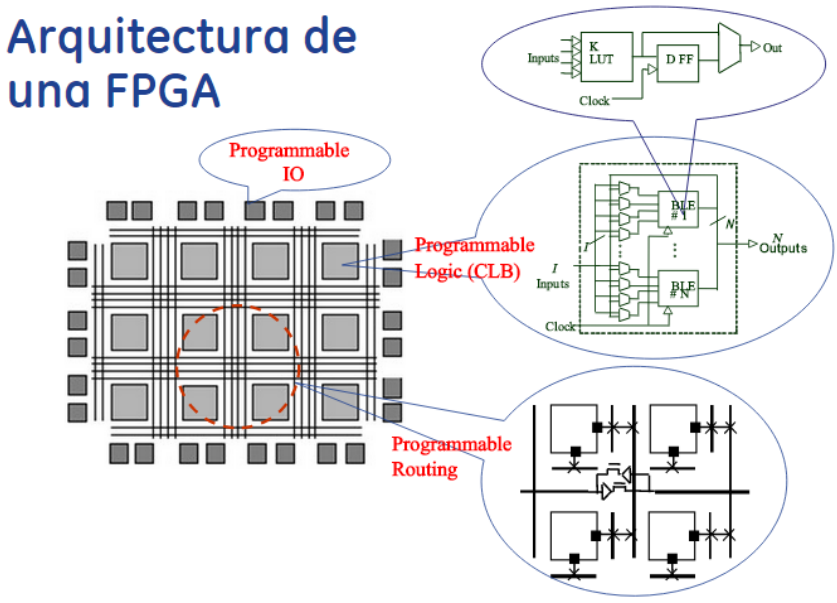


Figura 2.6: Arquitectura genérica de una FPGA

2.2.1. FPGA Altera Cyclone II

La familia de FPGAs *Cyclone II* de *Altera* [2], es la segunda revisión de la exitosa familia *Cyclone*, catalogada dentro de las familias de bajo coste de la compañía. A día de hoy la última revisión de la familia *Cyclone* es la *Cyclone V*, por lo que estamos ante una arquitectura que se ha quedado un poco obsoleta, puesto que data del año 2004. Pese a ello, *Altera* sigue soportándolas hoy en día y recomendado su uso para nuevos proyectos.

La principal característica de esta familia de FPGAs es que están fabricadas con una tecnología de 90 nanómetros de la compañía TMSC y que están orientadas a ofrecer un alto rendimiento manteniendo un bajo consumo, con el objetivo de ofrecer una alternativa a los ASIC de la época.

En cuanto a su arquitectura, el bloque principal de estas FPGAs es lo que *Altera* denomina como LAB "*Logic Array Block*", en los que se implementarán las funciones deseadas. Más concretamente, cada LAB está compuesto por 16 elementos lógicos o LEs, siendo cada uno de ellos capaz de implementar una función definida por el usuario. Los LABs están agrupados en una matriz bidimensional a lo largo de todo el circuito integrado, pudiendo contener desde 4.608 LEs en los dispositivos más modestos hasta 68.416 en los más avanzados.

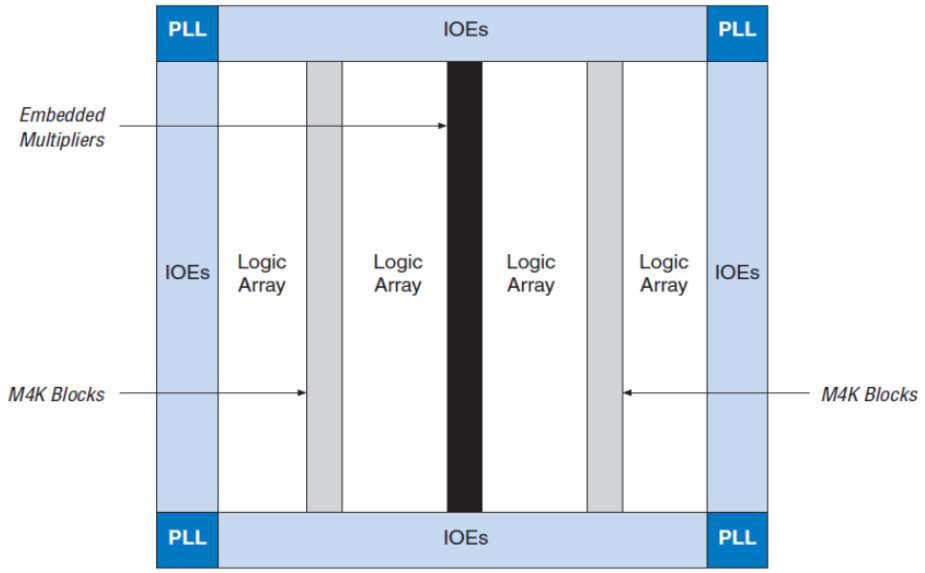


Figura 2.7: Diagrama de bloques general de una FPGA de la familia *Cyclone II* de *Altera*

Cada uno de los LEs está compuesto por una LUT de 4 entradas capaz de implementar funciones de hasta 4 variables, un registro capaz de operar como un biestable D, T, JK o SR, una conexión para el *carry* y la habilidad para interconectarse con otros LEs, ya sean del mismo LAB o de fuera de él.

Además, pueden operar en dos modos, el denominado como *Normal*, para implementar funciones lógicas genéricas, y el modo *Aritmético*, especialmente útil para implementar funciones como sumadores, contadores, acumuladores o comparadores.

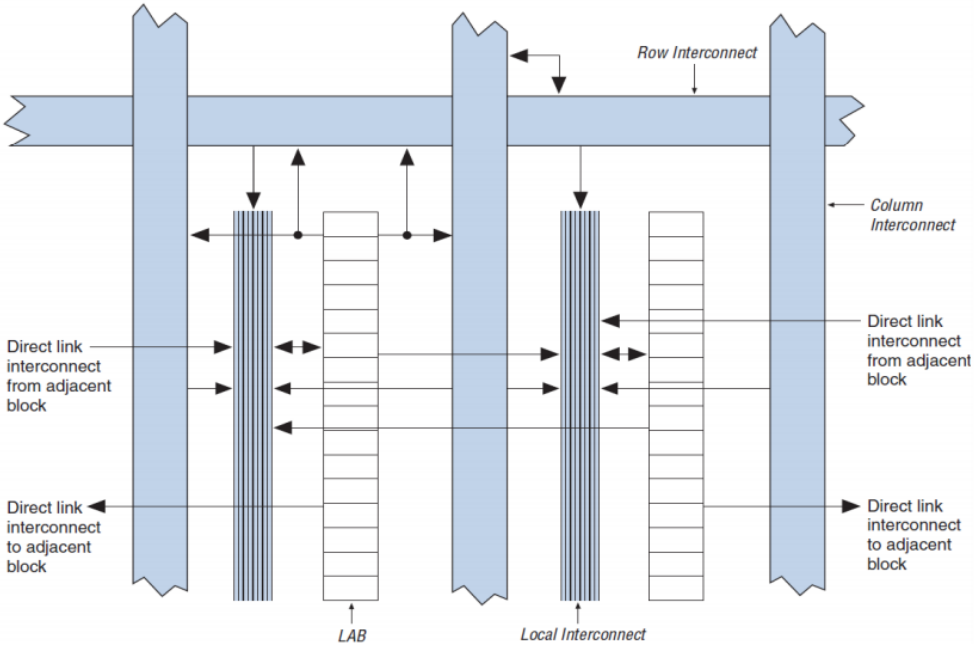


Figura 2.8: Estructura de interconexionado de los LABs

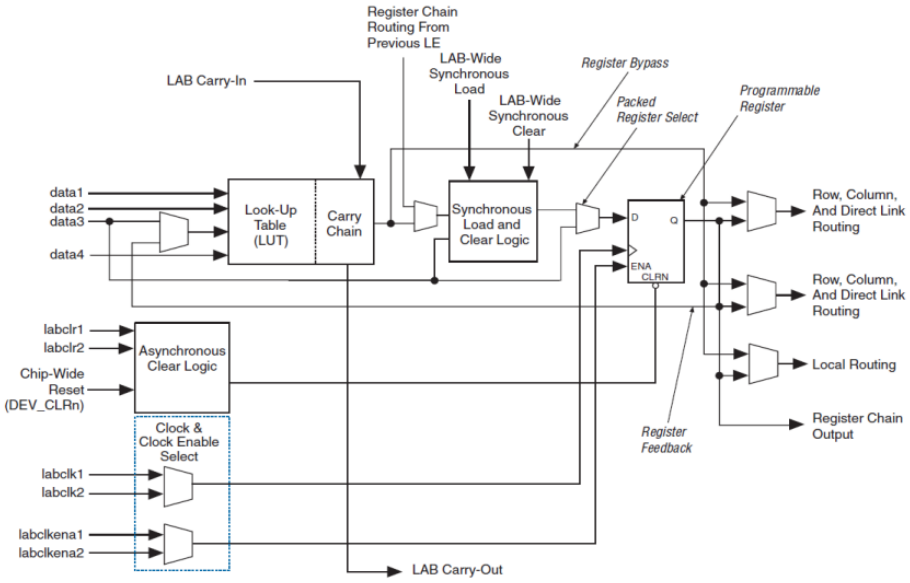


Figura 2.9: Diagrama de un bloque LE

Disponen de una red global para transmitir las señales de reloj, consistente en hasta 16 canales globales que llegan a todos los bloques. Esta es una de las partes más importantes en una FPGA, puesto que es muy importante que la señal del reloj llegue al mismo tiempo a todos los bloques para evitar problemas de sincronismo entre las diferentes partes que integren la aplicación que se desea desarrollar. A parte de esto, incluye hasta 4 PLL "*Phase-Locked Loops*" capaces de generar señales de reloj a diferentes frecuencias mediante multiplicadores y divisores.

En cuanto a la memoria de la que dispone, además de la ofrecida por cada uno de los LEs, dispone de bloques de memoria M4K, bloques *dual-port* de 4 Kb de capacidad con soporte para paridad, teniendo un total de 4.608 bits. Dichos bloques permiten accesos *dual-port* o *single-port* a palabras de hasta 36 bits a una velocidad de hasta 260 MHz. La cantidad de memoria disponible va desde los 119 hasta los 1.152 Kb de memoria.

Para aplicaciones de procesado digital de señal, la arquitectura *Cyclone II* dispone de bloques multiplicadores dedicados a ello, capaces de implementar dos operaciones de 9x9 bits o una de 18x18 bits a frecuencias de hasta 250MHz. Dichas operaciones también se puede implementar con el uso de la lógica programable, pero al ser operaciones costosas, y que por lo tanto requieren de bastantes recursos, es bastante común encontrarlas implementadas directamente en hardware para poder hacer este tipo de operaciones de forma más eficiente.

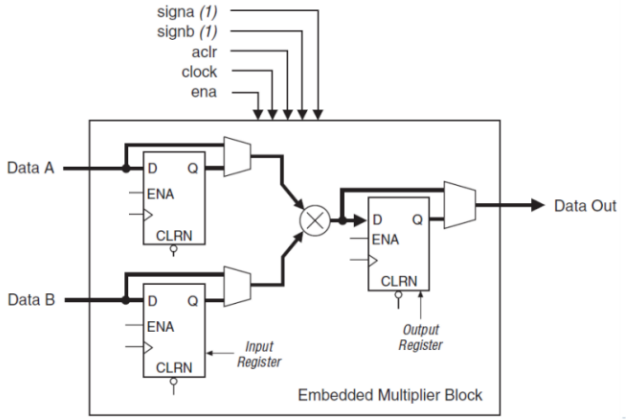


Figura 2.10: Diagrama de un bloque multiplicador

Finalmente, el último tipo de bloque relevante que podemos encontrar dentro de la arquitectura son los IOE "*Input/Output Element*", que son una interfaz entre los pines de entrada/salida y la lógica programable. Los pines de entrada y salida soportan diferentes estándares, tanto diferenciales como simples, tal y como pueden ser PCI (33/66 MHz, 32/64 bits), PCI-X, o LVDS a una velocidad máxima de 805 Mbps para entradas y 640 Mbps para salidas.

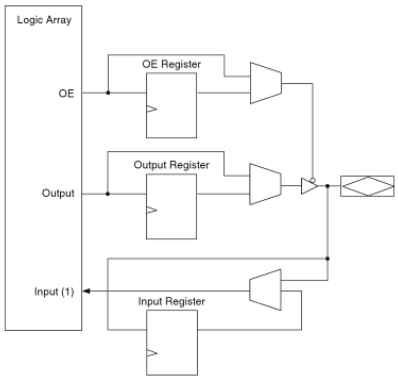


Figura 2.11: Diagrama de un bloque IOE

Para finalizar, la herramienta más usada para el desarrollo de aplicaciones para esta familia de FPGAs es el *Quartus II* de la propia *Altera*, el cual se puede obtener de manera gratuita a través de su página web, previo registro en ella.

Además de todas las herramientas para diseñar, sintetizar, depurar y demás tareas concernientes al desarrollo de aplicaciones, también se ofrece el procesador NIOS II, un *soft-core* que puede ser implementado dentro de la propia FPGA y que luego puede ser programado como si de un procesador se tratara. El procesador embebido NIOS II es altamente configurable desde la herramienta *SOPC Builder*, el cual mediante un asistente irá guiando al usuario para que como resultado obtenga el procesador que mejor se adecúe a sus necesidades, pudiendo configurar desde las propias características de la arquitectura del procesador a los periféricos que se incluyen o no. Una vez sintetizado y metido en la FPGA, solo o junto a periféricos desarrollados en algún HDL, este procesador se programará desde un entorno de desarrollo de software basado en Eclipse, pudiendo desarrollar para él como si de un procesador al uso se tratara.

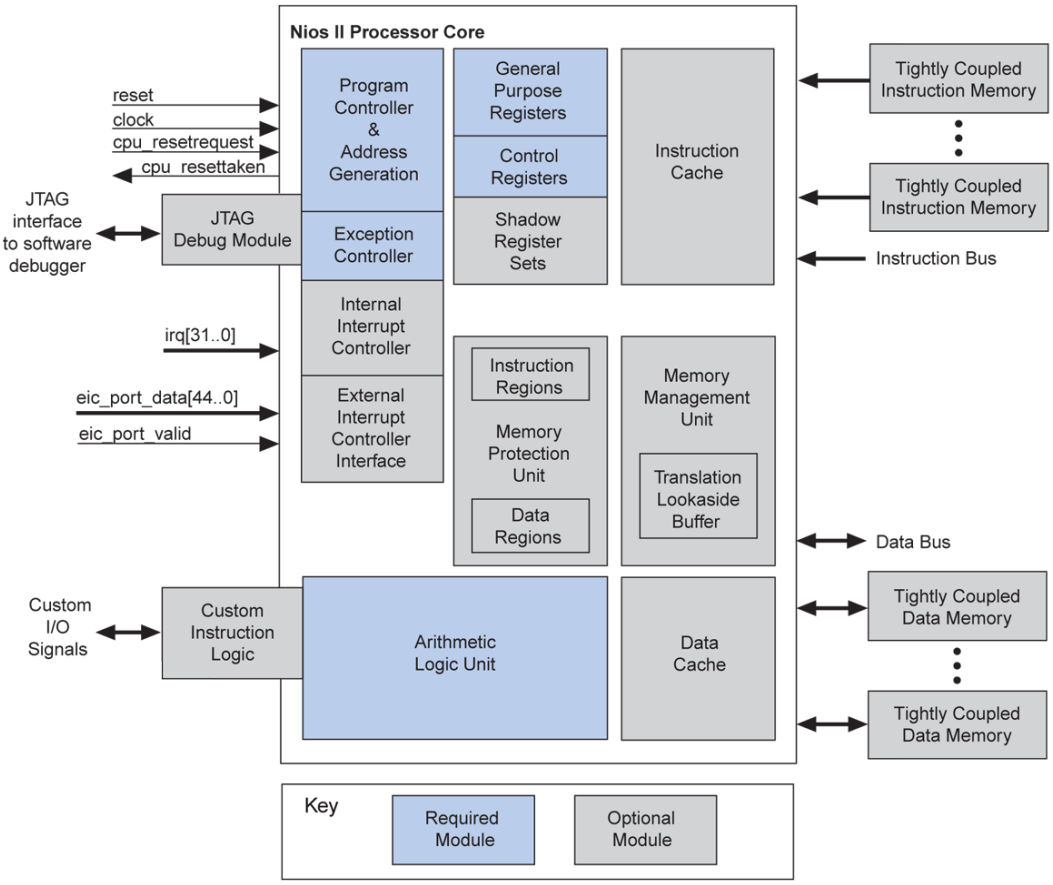


Figura 2.12: Diagrama de bloques del procesador embebido Nios II

Dentro de la familia de FPGAs *Cyclone II* de *Altera*, el modelo concreto que usa la tarjeta de evaluación que veremos a continuación es el modelo **2C35**, que se encuentra en la gama media dentro de su familia y que dispone principalmente de las siguientes características técnicas:

- Número de PLLs: 4
- Número de pines para señales de reloj: 16
- Número de bloques M4K: 105
- Capacidad de la memoria ofrecida por los bloques M4K: 483.840 bits
- Número de bloques multiplicadores: 35

- Número de pines de entrada/salida para el usuario: 475
- Número de canales LVDS: 201
- Número de elementos lógicos o LEs: 33.216
- Encapsulado BGA de 672 pines

2.3. Tarjeta de evaluación Altera DE2

La tarjeta de evaluación que vamos a usar para este proyecto es la placa DE2 (*Development and Education Board*) de *Altera* [7]. Tan solo con una FPGA no podemos hacer gran cosa, por lo que esta tarjeta de evaluación tiene todos los elementos que vamos a necesitar para implementar el prototipo del marco-reloj-despertador, salvo la pantalla táctil y los altavoces. De todas formas, la tarjeta tiene los conectores necesarios para que estas carencias no sean un problema.

En la siguiente lista podemos encontrar todos los elementos que incluye la tarjeta y que están conectados a la FPGA, por lo que podemos interactuar con ellos:

- FPGA *Altera* de la familia *Cyclone II*, concretamente el modelo *EP2C35F672C6N*
- *Altera* Serial Configuration device (EPCS16)
- USB Blaster para programar y depurar
- Memoria SRAM externa de 512 KB
- Memoria SDRAM externa de 8 MB
- Memoria Flash externa de 4 MB
- Socket para tarjetas SD
- 4 pulsadores
- 18 switches
- 18 LEDs rojos
- 9 LEDs verdes
- Oscilador de 50 y 27 MHz para usar como fuentes de reloj
- CODEC de audio de 24 bits, con conectores mini-jack de entrada de línea, salida de línea y entrada para micrófono
- DAC de 10 bits para señales VGA con conector hembra
- Decodificador de señales NTSC y PAL de televisión con conector coaxial de entrada
- Controlador Ethernet 10/100 Mbps con conector RJ45
- Controlador USB Host/Slave con conectores de tipo A y B
- Transceptor RS232 con conector DB9
- Conector PS/2 para teclado o ratón
- Transceptor IrDA para transmisión de datos infrarroja
- 2 slots GPIO de 40 pines para el usuario con protecciones mediante diodos
- 8 displays de 7 segmentos con punto
- Display LCD de 2x16 caracteres retro iluminado

Como podemos observar, se trata de una tarjeta de evaluación muy completa debido a que dispone de multitud de periféricos tanto de entrada como de salida. Además, en el CD que acompaña a la tarjeta se pueden encontrar multitud de ejemplos que hacen uso de ellos, junto a una gran cantidad de documentación con la que el usuario podrá aprender los entresijos de la tarjeta. Para nuestro caso, la tarjeta cumple con los requisitos mínimos que necesitamos para la implementación.

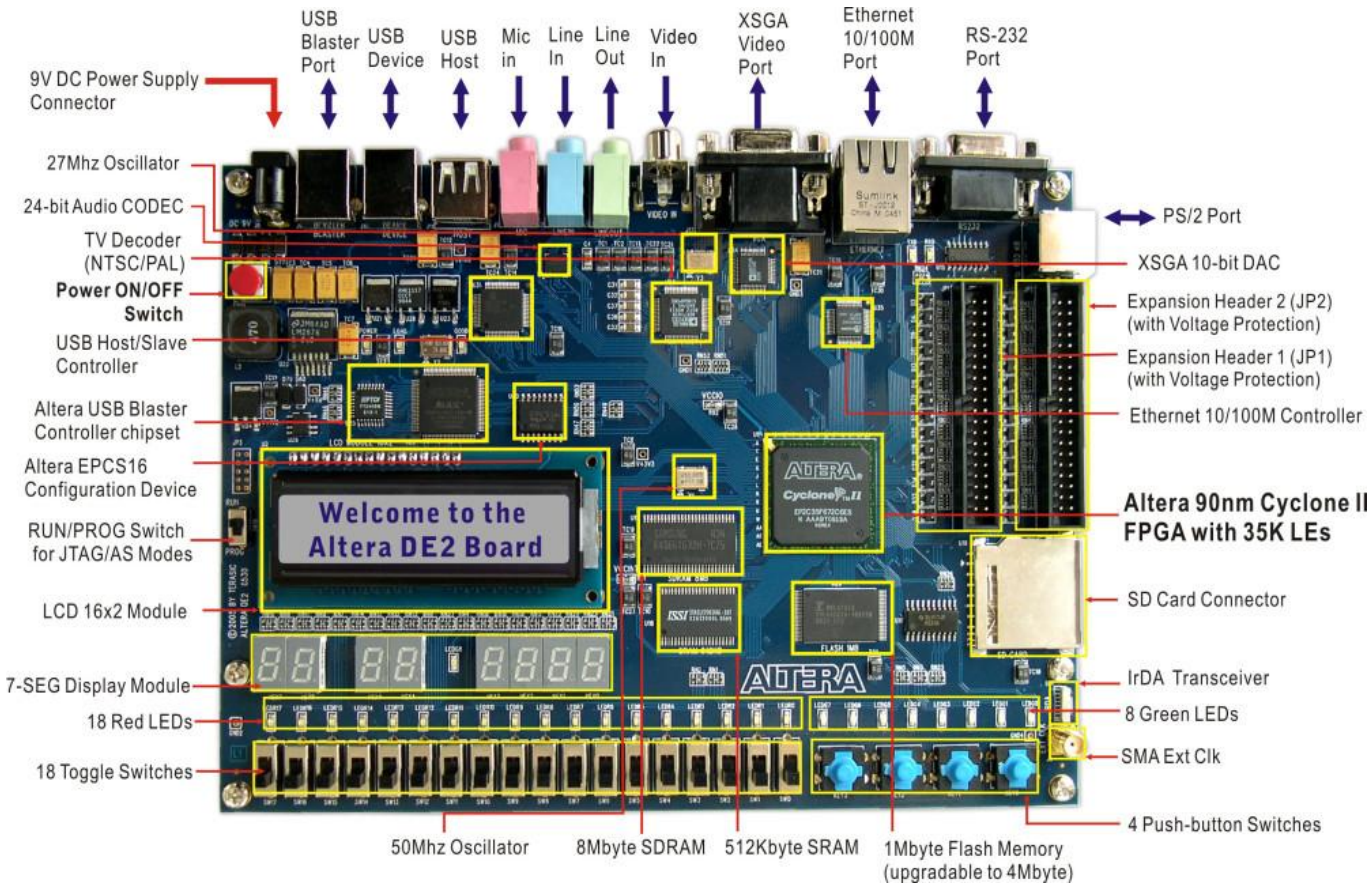


Figura 2.13: Vista general de los componentes de la tarjeta de evaluación *Altera* DE2

2.4. Pantalla táctil Terasic TRDB-LTM

La pantalla táctil *TRDB_LTM* (LTM) [8], ofrece todo lo necesario para desarrollar aplicaciones usando dicho panel táctil en un tablero de *Altera*, en nuestro caso el recién analizado DE2. El kit del LTM, contiene además de la propia pantalla táctil y el cableado necesario para su conexión a través del puerto de expansión, información útil como manuales y ejemplos de diseños y código fuente para la aplicación de una serie de demostraciones. Para este proyecto en concreto, ha sido muy útil el código fuente de una demo existente relacionada con un visor de fotos, gracias al cual se ha podido obtener el resultado final esperado.

Figura 2.14: Pantalla táctil TRDB-LTM de Terasic

Las características y componentes más importantes de esta pantalla, se enumeran a continuación:

- Módulo de TFT LCD en color de matriz activa *Toppoly TD043MTEA1*
- Soporte de 24 bits paralelos de interfaz RGB
- Control de registro de 3 hilos para la visualización y selección de función
- Construido incorporando contraste, brillo y modulación gamma
- Convertidor analógico digital *AD784* para la conversión de las coordenadas X/Y del punto de contacto con sus correspondientes datos digitales

Elemento	Descripción	Unidad
Tamaño de pantalla (Diagonal)	4.3	pulgadas
Relación de aspecto	15:9	-
Tipo de pantalla	Transmisiva	-
Área activa (HxV)	93.6 x 56.16	mm
Número de puntos (HxV)	800 x RGB x 480	puntos
Tamaño de punto (HxV)	0.039 x 0.117	mm
Disposición de color	Raya	-
Número de colores	16 Millones	-

Tabla 2.1: Especificaciones generales de la pantalla táctil TRDB-LTM de Terasic

En cuanto a los bloques por los que está compuesto el LTM, consta de tres componentes principales: el módulo LCD del panel táctil, el convertidor analógico digital y 40 pines del conector de expansión GPIO.

Todas las interfaces en el LTM están conectadas a la placa de evaluación *Altera* DE2 a través del conector de expansión de 40 pines. El módulo LCD y panel táctil llevan el control de las señales proporcionadas directamente desde la FPGA como las señales de entrada e imágenes del panel LCD. Por último, el convertidor AD convierte las coordenadas del punto de contacto con sus datos digitales correspondientes a la FPGA a través de la cabecera de expansión.

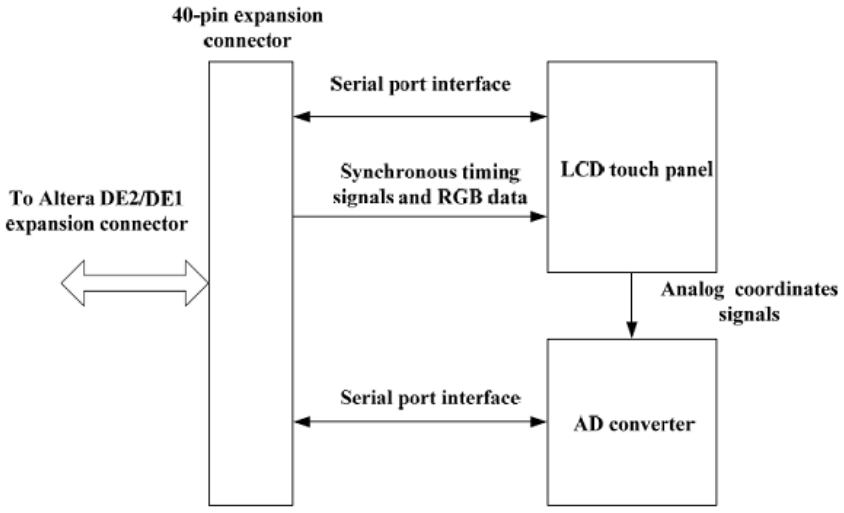


Figura 2.15: Diagrama de bloques de la pantalla táctil TRDB-LTM de Terasic

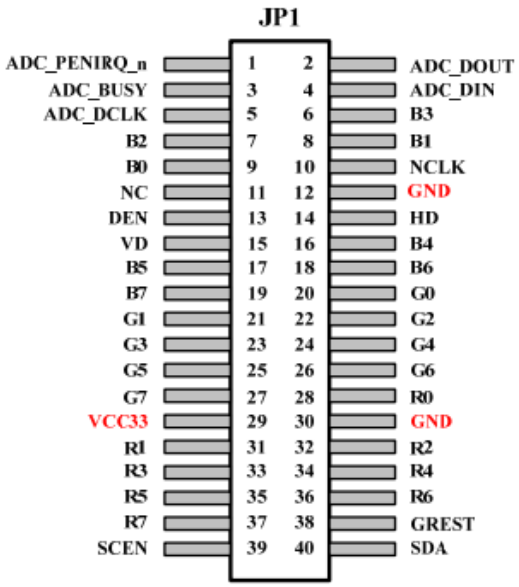


Figura 2.16: Descripción de la disposición de los pines del conector de expansión

2.5. Quartus II

La compañía *Altera* incluye junto con la tarjeta DE2, el programa *Quartus II* [10]. Este programa es la herramienta de software que provee *Altera* para el análisis y la síntesis de circuitos digitales y diseños realizados con lenguajes de descripción de hardware que se implementarán sobre FPGAs o CPLDs.

Este ambiente de programación soporta lenguajes de descripción de hardware como VHDL, Verilog o AHDL entre otros y sintetiza además, programas hechos a base de diagramas de bloques. También brinda una de las herramientas más importantes en el proceso de diseño digital como son los simuladores utilizados para el análisis temporal de las señales digitales, por ejemplo, *ModelSim*.

Una de las grandes ventajas de este programa es su edición web, que siendo gratuita, se puede descargar desde su página web y provee de estos servicios a la familia completa de bajo coste *Cyclone* de *Altera*.

Figura 2.17: Ambiente de programación de hardware *Quartus II* de *Altera*

3

Implementación

En este tercer capítulo, se podrá observar la unidad de control principal del proyecto encargada tanto del reloj y el despertador, como de las escrituras que deberán hacerse en la pantalla táctil. También encontraremos el esquema del sistema completo con cada uno de sus módulos y comentaremos la función que desempeñan los módulos más importantes dentro del sistema.

Antes de entrar a ver los entresijos de cada módulo y sus características, es necesario tener una visión general del funcionamiento del sistema. Por ello, ha de mencionarse en primer lugar, que el prototipo diseñado consta de cuatro modos principales y que tras cada uno, se ha de pasar por una fase de reescritura de la pantalla si algún elemento de la misma se ha visto modificado.

- *Modo reloj*: En este modo, se espera a que pasen los ciclos de reloj equivalentes a un segundo para hacer la reescritura completa de pantalla actualizando la hora, o bien se espera a que el usuario interactúe con los botones táctiles del menú para cambiar de modo.
- *Modo modificar reloj*: Una vez nos encontremos en la situación de poner el reloj en hora o modificarla, se deberá entrar en este modo al que se podrá acceder a través los menús. En esta situación, el reloj se detendrá a la espera de sufrir alguna modificación mediante las flechas habilitadas para ello en la pantalla. Tan pronto como se considere que la hora es correcta, se podrá volver a cualquiera de los modos de forma intuitiva gracias a los menús habilitados para ello.
- *Modo despertador:* Este modo es muy similar al anterior, dado que la modificación del despertador se hace de la misma forma que la del reloj gracias a las flechas habilitadas para esta función. La diferencia consiste en que, aparte de que los segundos en un despertador carecen de interés, es necesario indicar si se quiere activar o desactivar la alarma mediante un botón del menú con esa funcionalidad. A fin de resultar más cómodo e intuitivo para el usuario, esta acción activará un "chivato" en la pantalla que indica la situación del despertador, apagado o encendido.
- *Modo galería:* A priori, este será el modo principal ya que es el más vistoso por ser el único a color. Su funcionamiento es sencillo pese a tener los botones del menú ocultos para favorecer el visionado de las fotografías a pantalla completa. En la galería sólo habrá tres manipulaciones posibles, pasar a la siguiente foto, retroceder a la anterior o volver al modo reloj desde donde se podrá acceder al resto de modos.

En el anexo dedicado al manual de usuario se podrá encontrar la información detallada de cómo funciona cada uno de los modos de forma más extensa y gráfica.

3.1. Unidad de control principal

Una vez conocido el funcionamiento básico del sistema, resulta más sencillo comprender el siguiente diagrama ASM o máquina de estados algorítmica del módulo principal "*clock_control*", que se encarga de las transiciones entre los modos y del control de los mismos.

Diagrama ASM:
clock_control.vhd

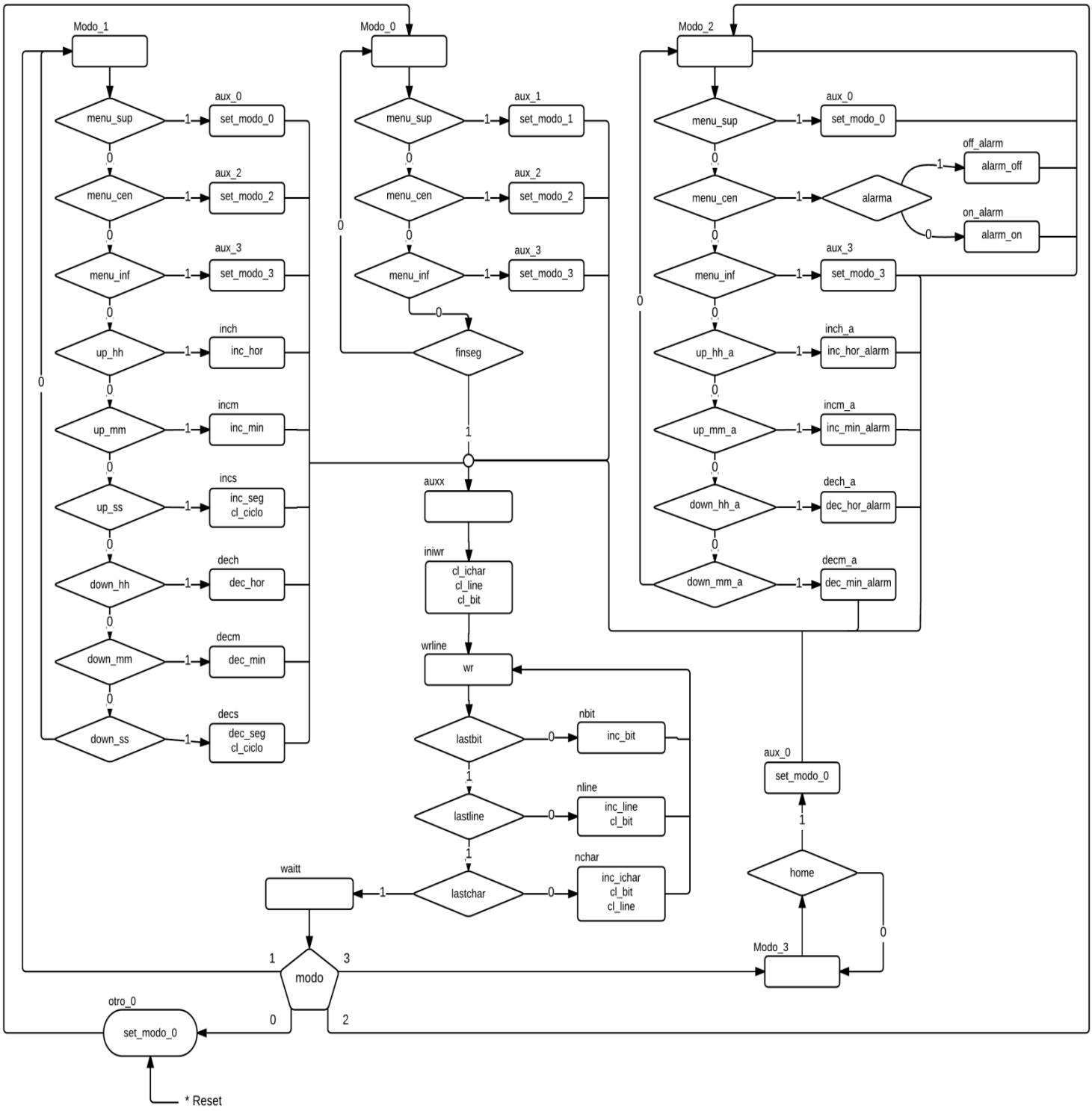


Figura 3.1: Diagrama ASM del módulo principal "*clock_control*"

Sin la unidad de proceso es difícil, por no decir imposible, saber con certeza cuándo se activan las señales necesarias para cambiar de estado, pero podemos apreciar en la figura superior los modos anteriormente mencionados.

El modo reloj se corresponde con el modo_0 y como se había comentado, se trata de un bucle de espera. Como se verá en la unidad de proceso, "finseg" se activará cuando los ciclos de reloj equivalgan a un segundo y mientras tanto, queda a la espera de detectar la activación de alguna de las señale del menú.

La rama que cuelga del modo_1 del diagrama ASM, refleja el comportamiento del modo modificar reloj y tras detectar alguna de las acciones en pantalla, pasa automáticamente a la reescritura de todo el contenido.

Asimismo, el modo_2 y el resto de estados que penden de él, plasman el comportamiento del modo despertador. Como se puede ver, tiene un esquema muy similar al modo encargado de la modificación del reloj, salvo por la señal que se encarga de activar o desactivar la alarma.

Todos ellos, incluido el modo galería que únicamente comprueba si debe cambiar de modo, pasan después de activar las señales correspondientes por una serie de estados que se encargan de la reescritura de la pantalla. Tras esto, se vuelve al modo en el que se encontraba el sistema antes de la escritura, siempre y cuando no se haya modificado manualmente el modo al que se quiere cambiar.

3.2. Unidad de proceso principal

Como ocurre en la mayoría de diseños complejos, es más fácil entender el funcionamiento de la unidad de control cuando se puede consultar la unidad de proceso, donde nacen las señales necesarias para cambiar de estado. Por eso, al igual que en el apartado anterior, se muestra la unidad de proceso del módulo principal "*clock_control*".

En la figura siguiente se puede observar como existen diferentes contadores, dos encargados del despertador y cuatro encargados del reloj. Los dos primeros, uno para los minutos y otro para las horas del despertador, se incrementan o decrementan atendiendo únicamente a las señales provenientes de la pantalla táctil. Los cuatro del reloj en cambio, además de tener en cuenta estas señales, se guían de las señales que activan el resto de contadores atendiendo a los ciclos de reloj que conforman un segundo, que se haya alcanzado el límite de 59 segundos o que se haya alcanzado el límite de minutos. Todos estos contadores o más bien sus resultados, desde los segundos, minutos y horas del reloj, hasta los minutos y horas de la alarma, pasan por un proceso de divisiones y módulos para separar sus cifras y poder mostrarlas independientemente en la pantalla.

Para eso, es necesario un multiplexor con tantas entradas como posiciones susceptibles de ser alteradas tenga la pantalla. En este caso, contamos con un multiplexor de 19 entradas ligado a una tabla de posiciones del mismo tamaño que alberga los puntos iniciales de escritura de cada carácter numérico, así como de aquellas posiciones donde puedan encontrarse caracteres modificables; ya sean los menús, las flechas o el chivato del despertador.

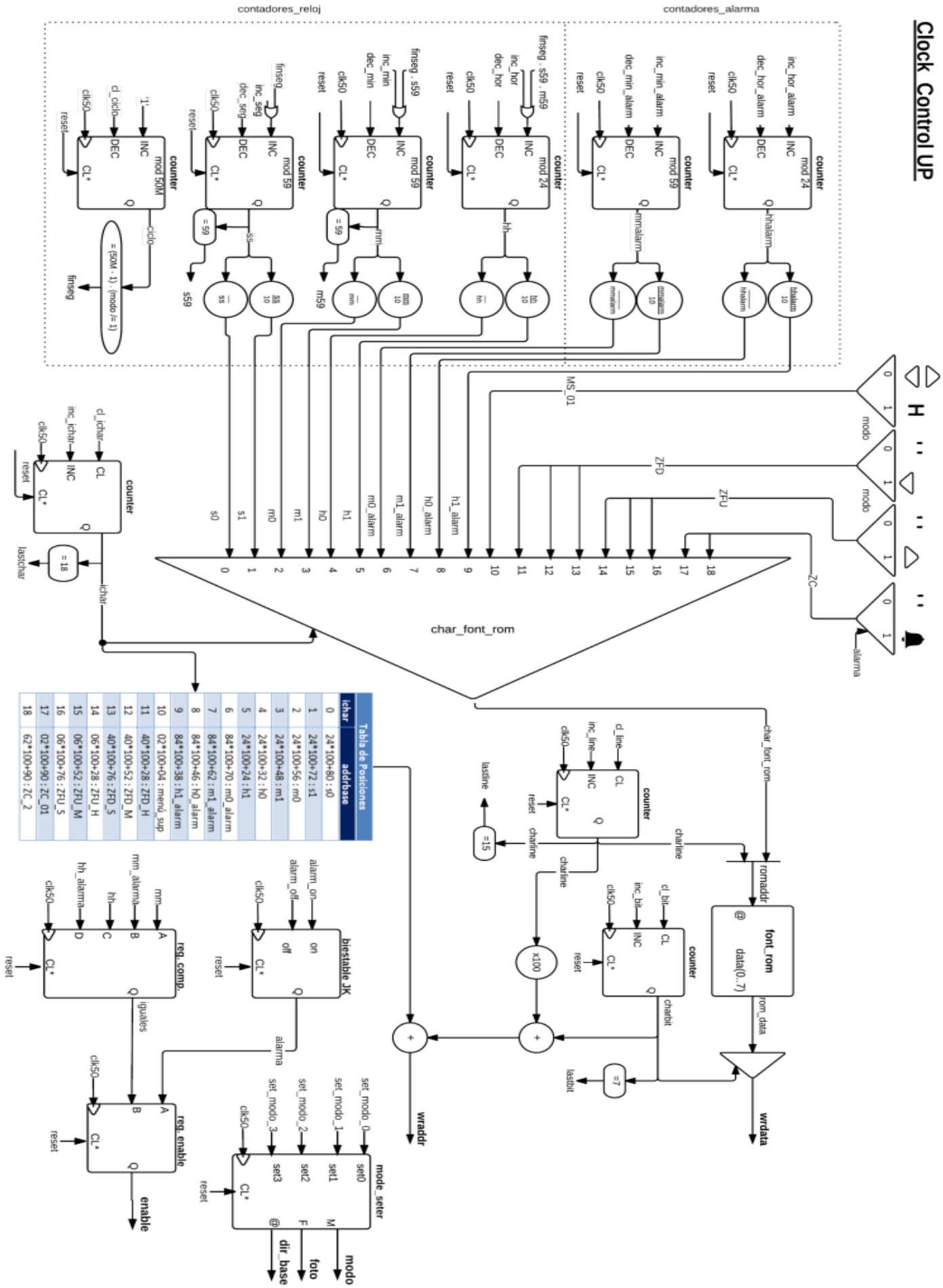


Figura 3.2: Unidad de proceso del módulo principal "*clock_control*"

La salida de este multiplexor junto con la salida de un contador encargado de contar las 16 líneas por las que se compone el largo de un carácter, van a parar a la memoria ROM, donde componen la dirección de memoria correspondiente al carácter necesario. Dicha memoria ROM, contiene un alfabeto de caracteres dedicados a la aplicación, tanto con los caracteres numéricos como los símbolos utilizados en el prototipo. Estos caracteres van escribiéndose uno a uno en la posición facilitada por la tabla antes mencionada, mientras otro contador lleva el control del ancho de cada carácter.

Por otro lado, este módulo cuenta con los elementos necesarios para activar el despertador cuando la hora del reloj y la del despertador coincidan. Para esta labor, se incorpora un comparador que activa la señal "*iguales*" cuando ambos tiempos son los mismos, así como un biestable JK que indica si la alarma está activada o no mediante la señal "*alarma*". Estas dos señales a su vez, activan la señal "*enable*" encargada de hacer sonar el despertador, siempre y cuando las dos anteriores estén activas.

Por último, y como se había mencionado previamente, la unidad de proceso "*clock_control*", se encarga de gestionar el modo en el que se encuentra el sistema y los cambios de modo que puedan hacerse a través de los menús del dispositivo. Esto se hace mediante un registro que además de indicar el modo en el que debe continuar el sistema, devuelve la dirección base de escritura donde se deberá comenzar a escribir en función del modo en el que se encuentre. Por si fuera poco, también activa la señal "*fotos*" cuando el modo se corresponde con la galería de imágenes, para ceder así la pantalla a las imágenes cargadas en la memoria flash de la FPGA.

3.3. Funcionalidad de los módulos auxiliares y diagrama RTL

En la figura 3.3, podemos observar el diagrama RTL o diagrama de transferencia de registros de todo el sistema, donde se aprecia la disposición de cada módulo con las distintas conexiones entre ellos.

Además del ya comentado "*clock_control*", nos encontramos con el módulo "*reset_delay*", que genera las señales de "*reset*" necesarias para inicializar el sistema cuando es necesario. Este módulo en particular, tiene la peculiaridad de que en lugar de generar una única señal de reinicio para todo el sistema, genera tres pero desfasadas en el tiempo. Esto se debe a que no todos los módulos pueden inicializarse al mismo tiempo ya que entrarían en conflicto entre ellos por los tiempos de espera necesarios y la falta de datos en momentos puntuales.

Los módulos "*flash_to_sdram_controller*", "*img_ram*" y "*sdram_control_4port*", tienen la función de controlar las imágenes que vienen de la memoria flash y todo aquello que queramos imprimir por pantalla, para escribirlas de forma controlada en la memoria SDRAM. Tras esto, se enviarán las imágenes y caracteres correspondientes a la pantalla táctil a través del conector de expansión GPIO, listas para ser mostradas. Por otro lado, los módulos "*lcd_spi_*controller" y "*adc_spi_controller*", se encargan de configurar la pantalla LCD y el sistema de transmisión de coordenadas respectivamente. Estos módulos se han tomado de una de las demostraciones incluidas en la documentación básica de la pantalla táctil, por lo que se puede acudir a dicha documentación para más información.

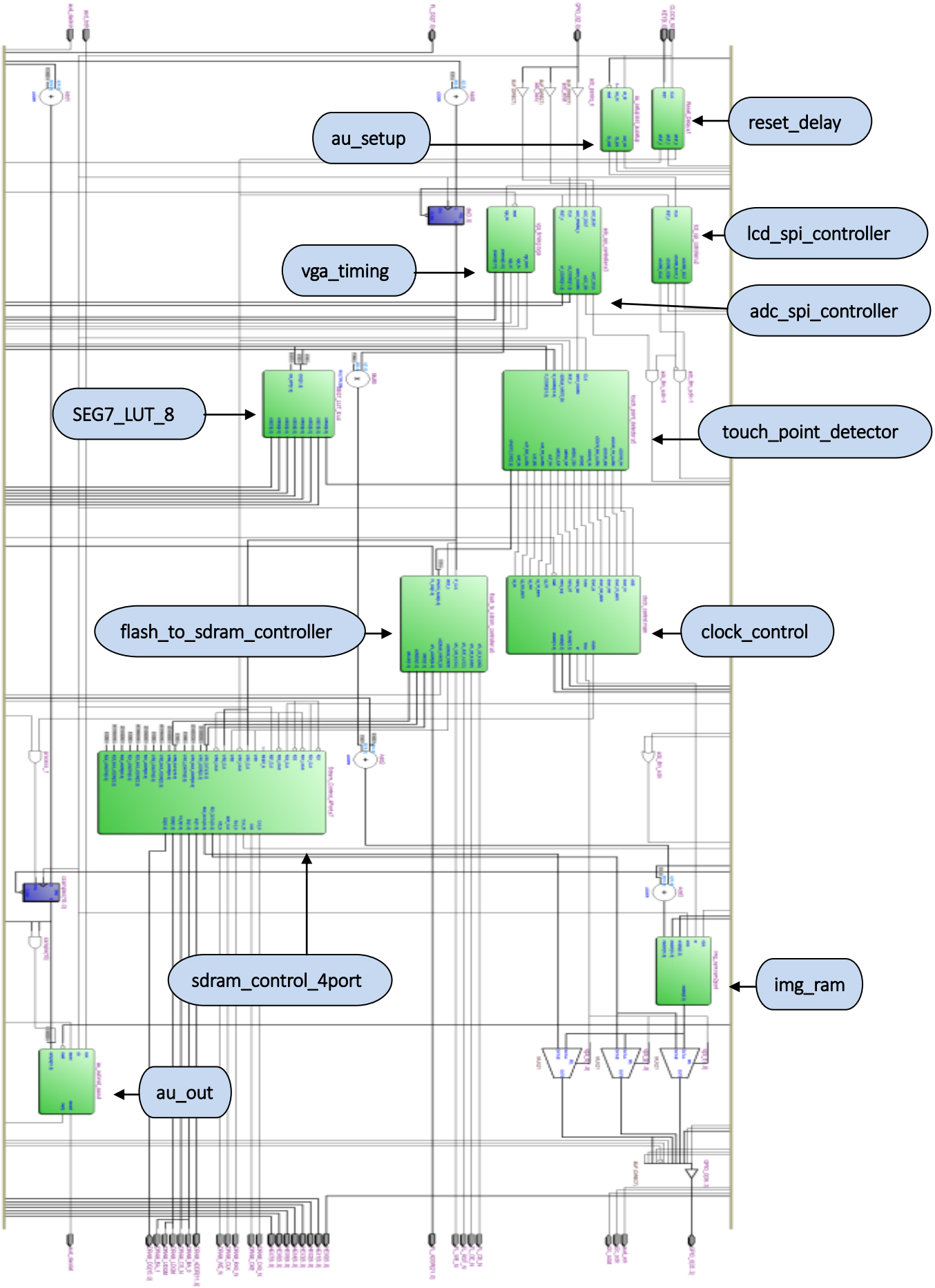


Figura 3.3: Diagrama RTL de interconexiones del sistema completo

Otro de los módulos importantes es el "*touch_point_detector*", que como se puede imaginar, se encarga de controlar y activar las señales relacionadas con las actuaciones sobre la pantalla táctil. Para diferenciar estas señales, ha sido necesario acotar las zonas de acción permitidas para el usuario mediante recuadros de diferentes tamaños sobre los puntos de interés, en función también del modo en el que se encuentre el sistema. Estas zonas se delimitan gracias a las coordenadas X e Y de la pantalla y haciendo uso del módulo "*SEG7_LUT_8*", podemos observar la última coordenada con la que se ha interactuado en los displays de 7 segmentos de la placa de evaluación *Altera* DE2, en formato hexadecimal.

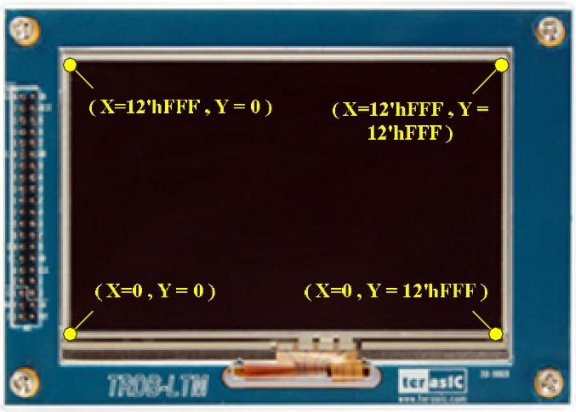


Figura 3.4: Disposición de las coordenadas en la pantalla táctil TRDB-LTM

El módulo denominado "*vga_timing*" es determinante para el buen funcionamiento del sistema, ya que se encarga de la generación de sincronismos para el resto de módulos, incluida la escritura en pantalla, teniendo en cuenta las dimensiones de esta y los retardos que se generan en la escritura.

Para finalizar, si atendemos a los dos módulos restantes, "*au_setup*" y "*au_out*", tienen entre ambos, la funcionalidad de generar y transmitir el sonido del despertador a través de la salida de audio de la placa DE2. Estos módulos se han reciclado de otros trabajos anteriores y se han modificado para su aplicación en este proyecto, de tal forma que se encuentran a la espera de la activación de la señal "*enable*" generada por "*clock_control*" para comenzar a transmitir el sonido.

Una de las tareas que más dedicación y quebraderos de cabeza ha supuesto, ha sido la combinación de la demostración "*ePhoto*" para la galería proveniente de la documentación de la pantalla LTM, con la visualización del reloj. Esto se debe a que cada uno de los dos modos utiliza una memoria diferente y a la hora de mostrar la información por pantalla, habrá que seleccionar en función del modo en el que se encuentre el usuario, la memoria desde la que se leerá dicha información. Concretamente, el modo galería vuelca las imágenes desde la FLASH en una memoria de pantalla RAM externa, mientras que el modo reloj utiliza como memoria de pantalla una RAM interna diferente. Para entender mejor este proceso, se explica a continuación con más detenimiento el proceso que sigue el sistema para que el modo galería funcione correctamente.

En primer lugar y tan pronto como el *bitstream* se descarga en la FPGA, los valores de registro del controlador de LCD utilizado para controlar la función de visualización del LCD, son configurados por el bloque "*LCD_SPI_Controller*". Mientras tanto, el bloque "*Flash_to_SDRAM_Controller*" leerá los datos RGB de la imagen almacenada en la memoria FLASH, para luego escribirlos en la memoria intermedia SDRAM.

En consecuencia, tanto las señales de control síncronas como los datos de imagen almacenados en la memoria intermedia SDRAM, serán enviados a la pantalla LTM a través del bloque "*VGA_Timing*".

Mediante ese proceso, se habrá cargado la imagen en la pantalla LTM, pero para el proyecto que nos ocupa, necesitamos también permitir la interacción del usuario con la pantalla. Por eso, cuando el usuario interactúe con el dispositivo táctil, las coordenadas X e Y del punto de contacto se obtendrán por el bloque "*ADC_SPI_Controller*" a través de la interfaz del puerto serie ADC. A continuación, el bloque de "*Touch_Point_Detector*" determinará si estas coordenadas están en un rango específico que tenga una acción programada. Si las coordenadas se ajustan al rango especificado en el bloque anterior, dicho bloque controlará el módulo "*Flash_to_SDRAM_Controller*" para leer los datos de la siguiente o anterior imagen de la FLASH; y se repetirán los pasos que se han mencionado antes para visualizar la imagen correspondiente.

A esto hay que añadirle que los datos correspondientes al reloj y el despertador, se guardan en una memoria de pantalla RAM interna pero de tal forma, que para su correcta visualización por pantalla, es necesaria la inversión de la misma. Si bien, existen otras formas de solucionar este problema, dado que tan sólo el modo galería se ve afectado por la inversión vertical de la pantalla, se optó por lo siguiente. Se invirtió verticalmente la escritura en pantalla mediante la activación de una variable de inversión en el módulo "*LCD_SPI_Controller*", arreglando así la visualización del resto de modos. Por otro lado, ya que el proceso de carga de imágenes en la FLASH resulta farragoso y estas no se modificarán con mucha frecuencia, se tomó la determinación de cargar las imágenes invertidas para que al visualizarse en la pantalla, estuvieran acorde con el resto de modos.

Otra de las cuestiones que ha consumido bastante tiempo de la implementación del proyecto ha sido la elaboración del archivo "*img_ram.mif*" que se carga nada más configurar el dispositivo. Este fichero contiene la información estática de toda la pantalla pixel a pixel en forma de una matriz de 12.000 posiciones. De todas ellas, 6.000 son para el modo reloj y su modificación y las otras 6.000 para el modo alarma, lo que equivale a la memoria total de pantalla RAM interna utilizada para la escritura de estos modos. La pega de este archivo es que debe escribirse prácticamente posición a posición para formar los marcos, caracteres o símbolos que no vayan a verse alterados en ningún momento y debe probarse con cada modificación hecha para comprobar si los píxeles marcados se encuentran en la posición deseada.

4

Conclusiones

Para finalizar, el capítulo cuarto, contará con las conclusiones del trabajo realizado incluyendo las lecciones aprendidas a lo largo del proyecto. Por otro lado, se plantearán posibles mejoras para futuras versiones que no se han incluido al encontrarse fuera de la planificación inicial.

4.1. Conclusiones del trabajo realizado

Tras la satisfactoria conclusión del presente proyecto, en el cual se han cumplido los objetivos planteados al principio, éstas son las conclusiones generales que se pueden arrojar.

En primer lugar, resulta primordial dejar clara la funcionalidad que se desea obtener del prototipo en los primeros pasos del proyecto y no alterarla demasiado. Esto es importante, ya que a medida que se avance y se complique el trabajo, encontraremos aspectos que nos gustaría mejorar y es más que probable que invirtamos más tiempo del esperado en detalles que no aportarán suficiente valor añadido al prototipo para que merezca la pena el tiempo y esfuerzo empleados para lograrlo.

Otra cuestión que debemos tener en cuenta, es que dividir el trabajo en tareas y módulos más sencillos e independientes, ofrece una serie de ventajas que debemos aprovechar. Entre éstas se encuentra, como ya se ha comentado, la reutilización de módulos de diferentes proyectos. De todas formas, se debe tener en cuenta que en ocasiones, la comprensión del proyecto del que se quiere obtener información y la posterior modificación de los módulos que se quieren reciclar, puede suponer costes de tiempo más elevados que diseñar e implementar directamente la solución deseada.

Por otro lado, ha sido esencial la división del proyecto entre sus distintos modos, ya que la mayoría podían funcionar de forma independiente facilitando la tarea de diseño e implementación así como las fases de pruebas. En este caso se realizó en primer lugar el modo reloj, para ir añadiéndole una vez verificados y probados el resto de módulos, la galería de fotos y el despertador. Para finalizar, se hicieron las interconexiones necesarias entre los distintos módulos y se retocaron los aspectos fundamentales para el correcto funcionamiento del sistema.

Respecto al funcionamiento del prototipo, cabe destacar una serie de cuestiones que se han de tener en cuenta a la hora de su uso. La primera y más importante, consiste en los tiempos de respuesta que ofrece el dispositivo. Antes de saturar el dispositivo con nuevas órdenes en el panel táctil, se ha de dejar cargar el contenido de la orden previa para que la escritura en pantalla se realice de forma correcta y no haya conflicto entre los modos.

Relacionado con lo anterior está la sensibilidad de la pantalla táctil. En dicha pantalla, cada nueva coordenada que se introduzca, pese a distanciarse en milímetros de la anterior, supone una orden nueva que el sistema procesará. Si bien esto es así, las pulsaciones sobre este periférico deben ser firmes, cortas y precisas para evitar variaciones en las coordenadas que supondrían el envío de varias órdenes de forma casi simultánea.

Gracias a la elaboración de este proyecto, un trabajo práctico centrado en la elaboración de un prototipo, he podido ampliar mis conocimientos sobre el manejo y funcionamiento de las FPGAs y sobre el entorno de programación hardware *Quartus II*. También ha resultado enriquecedor por la necesidad de aprender las nociones básicas de un lenguaje HDL como Verilog, con el que hasta la fecha, no estaba familiarizado pese a conocerlo. Este trabajo ha supuesto un acercamiento al mundo del diseño comercial al elaborar un producto existente en el mercado pero soportado por otra tecnología con mayor auge y proyección.

4.2. Próximos pasos y futuras mejoras

Una vez terminado el proyecto y visto el grado de satisfacción con el prototipo, son varias las futuras mejoras que podían realizarse sobre esta base para lograr un producto de mercado suficientemente atractivo para su producción.

En primer lugar y dado que se trata únicamente de un prototipo, el número de imágenes que se permiten en el dispositivo es el máximo que permite la memoria FLASH del mismo, que en el caso de esta placa de desarrollo de *Altera*, es de tan solo 4MB. Teniendo en cuenta el tamaño de la pantalla y que el formato y disposición de las fotos está limitado, el número de imágenes permitidas se traduce como tres fotografías dispuestas de un modo concreto y que han de ser cargadas en la placa con anterioridad. Tal y como está programado en este momento, esas tres fotografías cada una de 800 x 480 puntos, deben formar una sola imagen de 800 x 1440 puntos con ellas ensambladas en formato de imagen de mapa de bits BMP.

Lo que se pretende conseguir en un futuro, es cargar tantas imágenes independientes como se quiera a través del puerto USB o de la ranura para tarjetas SD del dispositivo para eliminar esta limitación, facilitando así el cambio de imágenes por parte del usuario.

Otra mejora que podría realizarse, sería incluir junto con el modo reloj, una línea de texto bajo la hora indicando la fecha completa del día en el que nos encontremos. Además de esto, se podría incluir un nuevo modo calendario, en el que se mostrara a pantalla completa el mes en cuestión y se permitiera avanzar y retroceder entre estos.

Con el fin de mejorar la estética de la aplicación, se había pensado que en lugar de realizar los menús y caracteres con imágenes de bits, se podría crear una biblioteca de imágenes a color con un formato atractivo y sustituirlos.

Por último, y pensando en la cada vez mayor tecnología que se está implantando en los hogares, sería posible aplicar esta idea de marco-reloj-despertador con las mejoras citadas, a por ejemplo, una mesilla de noche interactiva que proporcione estas funcionalidades además de otras como radio, música y recordatorios programables.

Bibliografía

[1] Gordon Moore, in *Cramming more components onto integrated circuits,* Electronics Magazine Vol. 38, No. 8, (April 19, 1965).

[2] Manual del dispositivo Cyclone II. Website: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyc2/cyc2_cii5v1.pdf

[3] Anant Agarwal and Jeffrey H. Lang, *Foundations of Analog and Digital Electronic Circuits* (Morgan Kufmann,Elsevier,San Francisco , 2005), Chap. 11.

[4] Kevin Skahill, *VHDL for programmable logic* (Addison-Esley, Massachusetts, 1996).

[5] Jose L. Martín Gonzalez, *Electronica digital*, (Delta publicaciones, Madrid, 2007).

[6] Cristian Sisterna, *Field programmable gate arrays(FPGAS),* Universidad Nacional de San Juan. Website: dea.unsj.edu.ar/sisdig2/

[7] Manual de usuario de la placa Altera DE2. Website: ftp://ftp.altera.com/up/pub/Webdocs/DE2_UserManual.pdf

[8] Manual de usuario de la pantalla táctil LTM. Website: http://www.terasic.com.tw/attachment/archive/237/TRDB_LTM_UserGuide_v1.22.pdf

[9] FPGA. Website: https://es.wikipedia.org/wiki/Field_Programmable_Gate_Array

[10] Información y descarga del programa Quartus II. Website: https://www.altera.com/products/design-software/fpga-design/quartus-ii/quartus-ii-web-edition.html

Anexo I: Manual de instalación

Antes de comenzar con ninguna de las acciones posteriores, es necesario conectar todos los elementos del dispositivo como se detalla a continuación para su correcto funcionamiento.

En primer lugar, ligaremos la placa de evaluación *Altera* DE2 con un ordenador que tenga previamente instalado el programa *Quartus II*, mediante el cable que une el puerto "*USB Blaster*" de la placa con un puerto USB del ordenador. Una vez hecho esto, se instalarán los controladores de la placa en nuestro ordenador para que podamos hacer uso de ella.

Mientras estos controladores se instalan, podemos ir conectando la pantalla táctil con la placa de desarrollo a través del puerto de expansión GPIO_0 de la placa usando el cable dispuesto para ello.

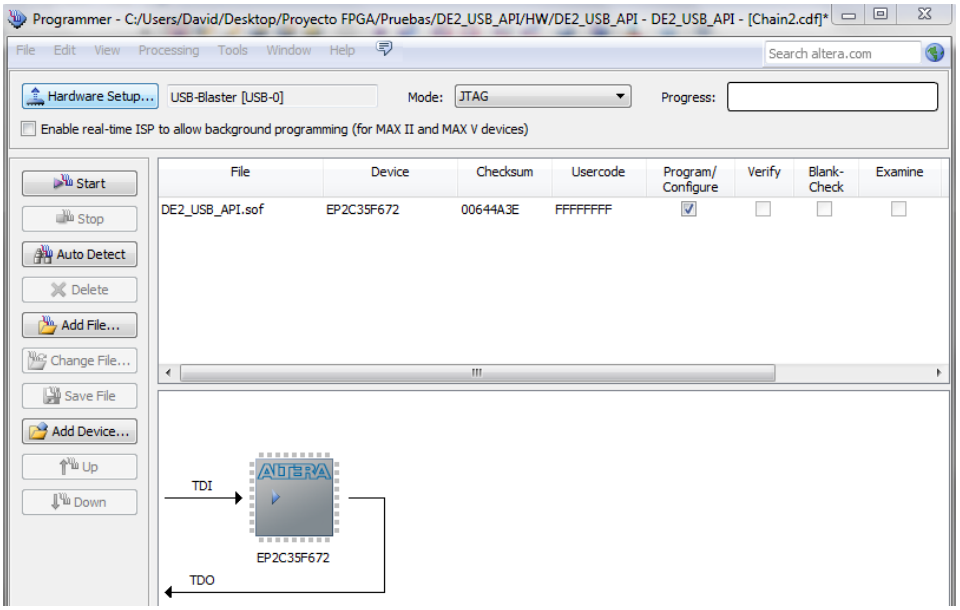
Por último, dotaremos al dispositivo de la corriente eléctrica necesaria para funcionar y encenderemos la placa de *Altera*.

Carga de fotos

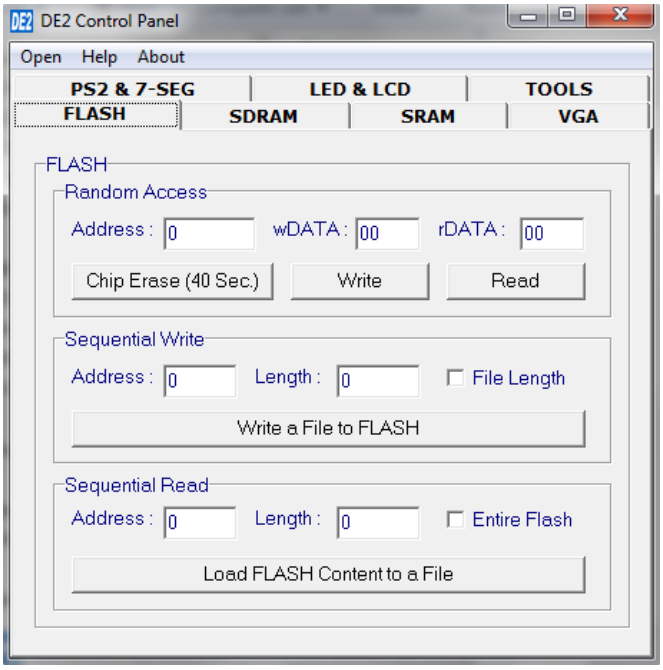
El primer paso para dotar al dispositivo de todas las funcionalidades programadas, será cargar en la placa de evaluación *Altera* DE2 las tres imágenes que queremos mostrar. Estas imágenes, cada una de 800 x 480 píxeles, deben formar una sola de 800 x 1440 píxeles en formato de mapa de bits BMP, e invertirse como se muestra a continuación.

Una vez tengamos la imagen unificada e invertida de las medidas especificadas y que no supere los 4MB de tamaño, procederemos a cargarla en la memoria FLASH del dispositivo. Al tratarse de una memoria no volátil, este proceso sólo tendremos que hacerlo la primera vez o cuando queramos cambiar las fotografías a mostrar. Para esta labor, será necesario seguir las instrucciones al pie de la letra tal y como se detallan a continuación.

1. Ejecutar el programa *Quartus II* → Tools → Programmer → Hardware setup: (USB-Blaster[USB-0]) → Add file → …\DE2_USB_API\HW\DE2_USB_API.sof → Start



2. Una vez hecho esto, ejecutaremos el controlador de software que se encuentra en …\DE2_USB_API\SW\DE2_Control_Panel.exe
3. En la ventana emergente, haremos lo siguiente:
    a. Open USB port 0
    b. Flash→Random Access → Chip Erase (40 Sec.)
    c. Flash →Sequential Write → File Length → Write a File to Flash → Seleccionaremos la imagen BMP que creamos anteriormente.

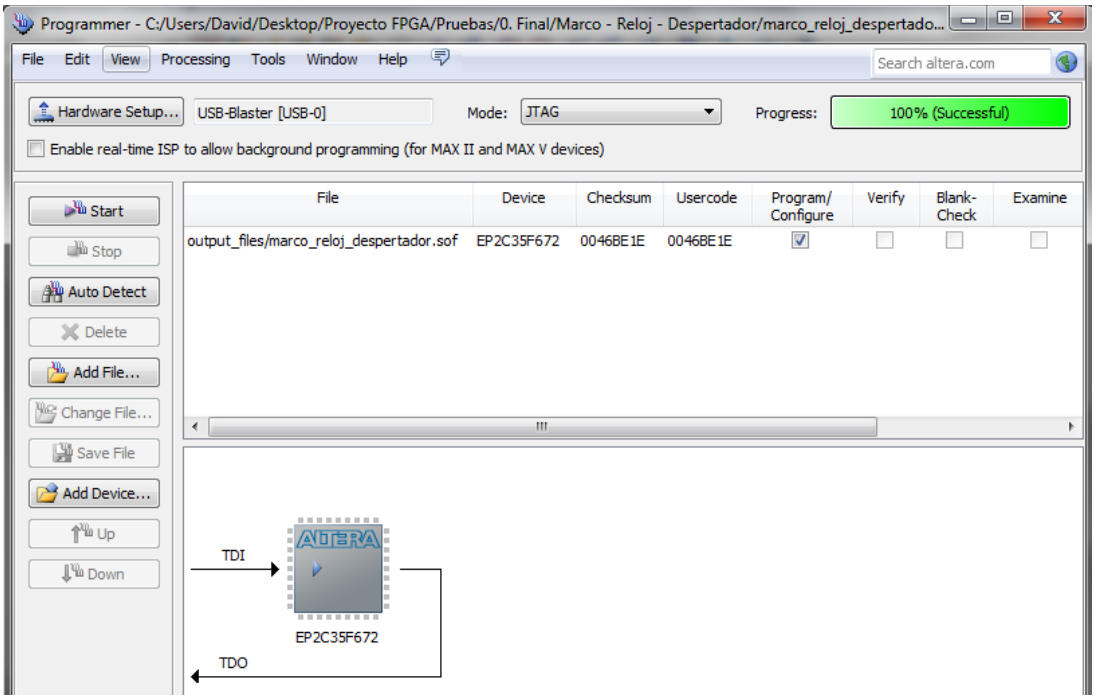


Una vez acabe el proceso de carga que puede tardar algunos minutos, podremos cerrar todas las ventanas ya que las fotos se habrán cargado en la memoria FLASH.

Arranque del sistema

Manteniendo todas las conexiones entre ordenador, placa y pantalla, abriremos de nuevo el programa *Quartus II* para programar esta vez el marco-reloj-despertador. Para esta función será necesario incorporarle a la placa DE2, unos cascos o altavoces a la salida de audio "*line out*", ya que en caso contrario, no podremos escuchar el sonido del despertador cuando este se active. Una vez abierto el programa, pasaremos a hacer lo siguiente:

1. *Quartus II* → File → Open Project → …\marco_reloj_despertador.qpf → Abrir
2. Una vez abierto y si no hemos alterado el código fuente, haremos clic en:
   Tools → Programmer → Hardware setup: (USB-Blaster [USB-0]) → Start
3. Si la ejecución ha finalizado con éxito, tendremos el dispositivo en funcionamiento.



Anexo II: Manual de usuario

Inicio del dispositivo

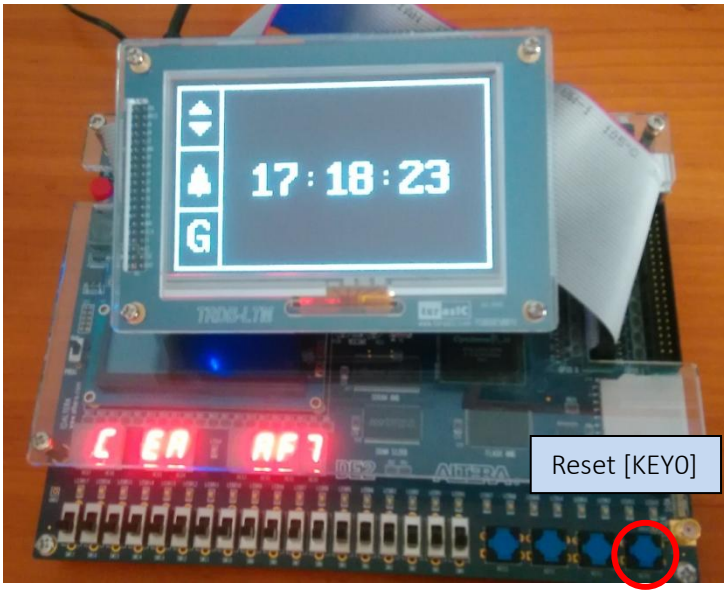
Una vez realizado todo el proceso de instalación, tendremos el prototipo encendido y en marcha. Siempre que arranquemos el sistema o lo reiniciemos mediante el botón de la esquina inferior derecha de la placa [*KEY0*], se iniciará por defecto en el modo reloj con todos sus dígitos a cero y contando. Tras este proceso, también se desactivará la alarma en caso de estar configurada y los valores de los displays de 7 segmentos pasarán a ser cero, por el contrario, las fotos permanecerán en la memoria FLASH aunque apaguemos el dispositivo.

Modo reloj

Este primer modo del dispositivo, muestra la pantalla táctil dividida en dos secciones diferentes, donde la mayor parte de ésta queda destinada a la visualización del reloj del prototipo, mientras que la sección izquierda deja cabida al menú de interacción del usuario. El menú siempre constará de tres botones con distintas funcionalidades dependiendo del modo en el que nos encontremos.

En este modo en concreto, nos encontramos lo siguiente:

- El primer botón táctil, identificado en la pantalla mediante una flecha ascendente y otra descendente, y situado en la esquina superior izquierda, sirve para cambiar al modo modificar reloj.
- El segundo botón táctil, con la imagen de una campana, y situado en la posición central del menú, sirve para cambiar al modo despertador.
- El tercer y último botón, indicado con una “G” mayúscula y situado en la esquina inferior izquierda de la pantalla, sirve para cambiar al modo galería.



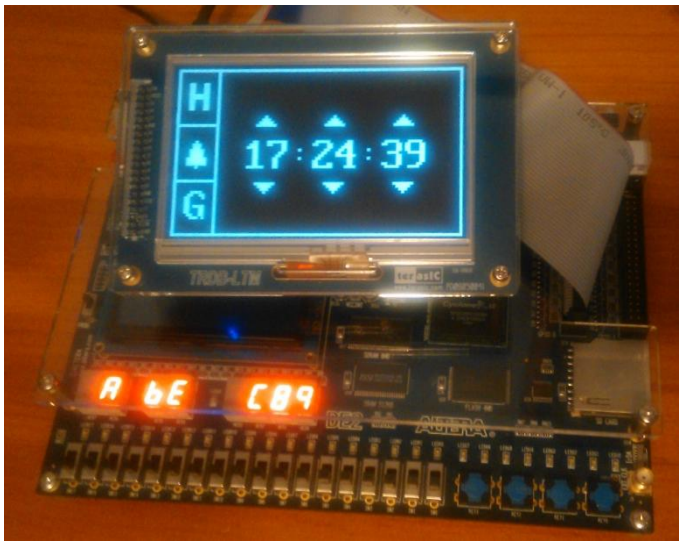
Modo modificar reloj

Este modo permite, como cabría esperar, la modificación de la hora del reloj mencionado previamente.

El reloj cuenta con horario, minutero y segundero, y todos ellos se pueden modificar utilizando las flechas que emergen encima y debajo de cada franja de números al entrar en este modo. Con el fin de facilitar la modificación del reloj, mientras se puedan modificar los valores de estos campos, el tiempo dejará de correr.

Tal y como resultaba en el modo anterior, se dispone de una pantalla dividida en dos secciones, de las mismas funciones y características que las antes mencionadas, únicamente variando el icono de la esquina superior izquierda por una "H" que permitirá volver a la pantalla del modo reloj.

Se podrá acceder directamente también tanto a la alarma como a la galería de la misma forma que en el modo anterior.



Modo despertador

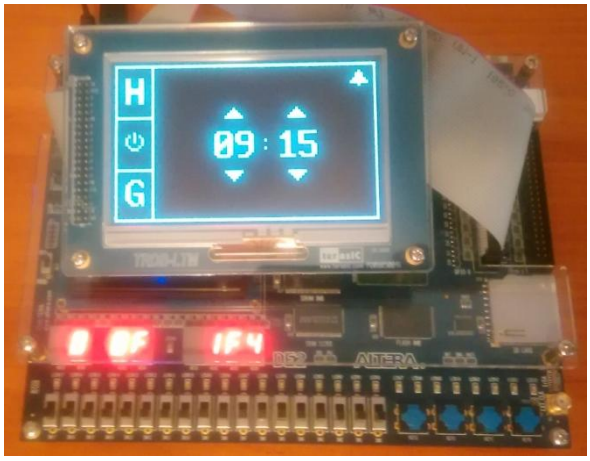
Este modo es el encargado de la configuración del despertador y de su activación y desactivación. Su uso resulta bastante intuitivo al igual que el del resto del prototipo y se asemeja bastante al modo encargado de modificar el reloj.

En la pantalla se puede observar un menú parecido al anterior, salvo por el cambio del botón central por la señal de encendido/apagado internacional. Este nuevo símbolo, sirve para activar el despertador con el valor numérico que se vea en ese momento en la pantalla o para desactivarlo. Sabremos si la alarma está activada gracias a una pequeña campana que hará de chivato en la esquina superior derecha mientras el despertador esté activado, ya sea en este modo o en el modo reloj; en caso contrario, este espacio permanecerá vacío.

La sección central cuenta en este caso con los elementos de la hora y los minutos, mientras que los segundos desaparecen ya que en un despertador, el segundero no tiene demasiado sentido. Sobre estos elementos se encuentran las flechas para aumentar o disminuir el valor numérico del despertador del mismo modo que se hacía en el modo modificar reloj.

Como es habitual, el despertador emitirá un sonido mientras los valores del reloj y del despertador coincidan. Este sonido persistirá a lo largo de un minuto, a no ser que se acceda a este modo y se desactive la alarma mediante el icono de encendido/apagado que se encuentra en la segunda casilla del menú izquierdo.

Al igual que en el resto de modos, los botones que no han sufrido alteraciones mantienen la funcionalidad que tenían.



Modo galería

Este modo consiste en un visor de las fotografías previamente cargadas como se ha explicado en el manual de instalación. Es posible acceder a él desde cualquiera de los modos a través del botón identificado con la letra “G” mayúscula, situado siempre en la esquina inferior izquierda del menú táctil.

Durante la visualización de cualquiera de las fotografías, se podrán llevar a cabo tres acciones diferentes en la pantalla táctil. Retroceder a la foto anterior pulsando sobre la franja lateral izquierda de la pantalla, avanzar a la siguiente pulsando sobre la franja lateral derecha, o volver al modo reloj accionando cualquier punto de la sección central.