UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

PLATAFORMA FPGA RECONFIGURABLE PARA IMPLANTAR EL SISTEMA DE CONTROL DE APUNTAMIENTO DEL SATÉLITE SATEDU

T E S I S
QUE PARA OBTENER EL TÍTULO DE:

INGENIERO ELÉCTRICO ELECTRÓNICO

P R E S E N T A:

JOSYMAR ITZAE GUZMÁN MERCADO

DIRECTOR DE TESIS: DR. ESAÚ VICENTE VIVAS

CIUDAD UNIVERSITARIA 2014

Agradecimientos

Este trabajo está dedicado a mis papás a quienes agradezco por inculcarme valores que me definen como la honestidad, el respeto, la responsabilidad y la justicia, por sus muestras de cariño, por acompañarme en cada entrenamiento de "fut", por estar presentes en los partidos cada sábado, por cada palabra de aliento, por cada corrección, por cada consejo, por impulsarme, apoyarme, por darme la oportunidad de terminar una carrera y más aún por permitirme compartirla con ustedes.

A mis hermanos (Nazul, Dení y Kenia) y familia (Gerardo, Tía Carmen, Beto, David, René, Daniel, Diana, Oscar) por confiar en mí, por aguantarme, por escucharme, por las experiencias vividas y su compañía.

A la UNAM por enriquecer mi forma de pensar, por permitirme cuestionar y razonar de muchas formas, por la diversidad de conocimiento disponible, por ser el primero de mis sueños, y ser el lugar que alberga los mejores recuerdos de mi vida.

Al Dr. Esaú Vicente V. por darme la oportunidad de formar parte de su grupo de trabajo donde encontré mi verdadera pasión y aumentó mi conocimiento, por guiarme y asesorarme en la elaboración de esta tesis. A mis profesores y sinodales el Ing. Ricardo Mota M. y el Ing. Moisés Rueda G. por compartir su conocimiento y ser fuente de inspiración sobre mi camino a seguir. A mis sinodales Dra. Fátima Moumtadi y M.I. Antonio Salva C. por su colaboración en la elaboración de este trabajo.

A Víctor y Jorge por la gran cantidad de variadas anécdotas y locuras vividas, pero sobre todo por la amistad sostenida durante muchos años.

A mis amigos Maru, Chio, Jessi, Naye, Oyuky, Gabo, Pancho, Galán, Lolo, Pastelín (los del O), Diego, Jonathan, Dibu, Chayo, Montoya, Paty (de la biblio), Pache, Eduardo, Pinky, Misael, Pablo, César, Daniel, Sosa, Ray, Dalay, Shaq, Fercho, Kenji, Quiroz, Viveros, Pame, Nandito, Jorge, Laura, Marilú por momentos de tensión (proyectos, tareas, exámenes), por momentos de desestrés (Ferchofiestas, trajineras, etc), por brindarme su amistad y formar parte de mi vida.

A mis amigos del IINGEN Rodrigo, Enkor, Miguel, Mario, Emilio y Eduardo con los cuales compartí una etapa nueva y fenomenal en mi vida. Sobre todo a Mario, Emilio y Eduardo quienes me asesoraron y me dieron pauta durante todo el desarrollo de esta tesis.

A mi novia por su cariño y alentarme día a día en cumplir cada uno de mis objetivos.

Contenido

Índice de figuras

Introducción

Desde la década de los 60's, varias instituciones (universidades, centros de investigación y desarrollo, ejército) alrededor del mundo, han desarrollado diferentes prototipos de simuladores que permiten recrear en tierra algunas de las condiciones que se presentan en el espacio exterior. Bajo diferentes arquitecturas, dichos simuladores permiten alcanzar a recrear con gran aproximación algunas de las condiciones medioambientales espaciales, una de las principales y mayormente emuladas es la falta de fricción.

Los satélites son parte fundamental de la vida moderna y contribuyen al desarrollo tecnológico global. Sirven para propósitos de toda índole siendo los más comunes: monitoreo de cambio climático, agricultura, telemedicina, sistemas de posicionamiento global (GPS) y comunicaciones. Esto hace importante el desarrollo de esta tecnología en nuestro país.

En el Instituto de Ingeniería de la UNAM (II-UNAM), desde hace varios años se trabaja intensamente en el desarrollo de tecnología espacial, que permita, por un lado, la formación de recursos humanos de alto nivel en el desarrollo de este tipo de tecnología en México, y por otro lado, la generación de sistemas didácticos, que permitan la capacitación en el área aeroespacial y el desarrollo de sistemas tecnológicos de diseño propio para utilización y validación, en primera aproximación, en sistemas terrestres y su posterior extrapolación a sistemas reales de vuelo.

Desde 2008 el II-UNAM, terminó el desarrollo del satélite educativo SATEDU. Este satélite es una plataforma didáctica, integrada por un rack de tarjetas electrónicas, cada una de las cuales representa un subsistema satelital. Uno de los objetivos de SATEDU es funcionar como herramienta didáctica de relativo bajo costo, que permita la capacitación y el acercamiento de la tecnología satelital a los alumnos de escuelas, universidades y centros de investigación en México.

En paralelo, otra línea de trabajo al interior, ha conseguido el desarrollo de algoritmos de control de orientación, los cuales han sido validados exitosamente en software por medio de simulaciones numéricas y visualizados por medio de modelos en realidad virtual.

Sin embargo, actualmente el grupo de desarrollo busca ir más allá de la simulación. Aprovechando mucha de la infraestructura generada en los últimos años por el grupo de trabajo, los esfuerzos se han volcado a la transferencia a hardware (propio y comercial) de los modelos matemáticos que actualmente corren en un ambiente de simulación en software.

La idea que persigue la transferencia a hardware de los algoritmos de control, tema fundamental de esta tesis, incluye el desarrollo de una tarjeta electrónica que cuente con un dispositivo que permita el desarrollo e integración de arquitecturas de cómputo polimórficas en un dispositivo FPGA.

En el presente trabajo de tesis se propone el diseño, ensamble y validación del sistema electrónico basado en una plataforma FPGA para implantar el sistema de control y apuntamiento de SATEDU. Este sistema (SMIN_V2) corresponde a la segunda generación del sistema SMIN_V1, desarrollado en el Instituto de Ingeniería, UNAM (II-UNAM). Dicho sistema integra módulos de potencia, comunicaciones, reconfiguración remota, entradas y salidas digitales, así como componentes de soporte. La característica de reconfiguración remota permite la reconfiguración total al vuelo de la arquitectura de cómputo descrita en él y de igual forma realizar pruebas de validación sobre la MSA (Mesa Suspendida en Aire), con objeto de verificar las maniobras de control de orientación en tres ejes para experimentos de recepción remota en tierra con SATEDU.

Esta tarjeta es compatible con la plataforma del satélite educativo SATEDU, y tiene los recursos para interactuar con dispositivos externos, tales como actuadores basados en ruedas inerciales.

Adicionalmente, cuenta con los medios necesarios en hardware y software que permiten utilizar a esta tarjeta fuera de SATEDU, como una plataforma didáctica para la enseñanza de dispositivos FPGA y tópicos asociados.

1

Proyecto SATélite EDUcativo (SATEDU)

Con el objetivo principal de entrenar y atraer a los jóvenes a la ciencia y la tecnología en el campo de los satélites, el IINGEN creó el satélite educativo SATEDU, un picosatélite o satélite artificial pequeño, que cumple con las normas internacionales de dimensiones, características y operación de satélites CubeSats, establecidas por la Universidad de Stanford (EUA, 1999), [28].

SATEDU es un Satélite Educativo, diseñado, fabricado y validado completamente en el Instituto de Ingeniería de la UNAM, para ser empleado en laboratorios escolares, aulas de clases, Tecnológicos, Universidades, Posgrados y Centros de Investigación, [27].

1.1 Experiencia previa a SATEDU

México ha experimentado en el área de tecnologías satelitales en diversas épocas, lanzando en la década de los ochentas los sistemas Morelos 1 y 2, los sistemas Solidaridad 1 y 2, y Satmex 5 (Morelos 3) en los noventas, éste último fue el primer satélite comercial mexicano lanzado mediante financiamientos privados por la empresa estatal Telecomm (Telecomunicaciones de México). Todos estos satélites han sido construidos fuera de nuestro país con tecnología extranjera, y aunque han sido satélites de clase mundial, no son obras mexicanas y por ahora no se ha podido colocar un satélite de tecnología mexicana en el espacio que sirva como propulsor para promover el desarrollo de tecnología mexicana.

El Instituto de Ingeniería (UNAM) ha tenido grandes experiencias dentro del área de desarrollo de satélites experimentales, una de estas fue el microsatélite experimental SATEX. En 1994 se pone en marcha el proyecto SATEX, desarrollado por un consorcio de instituciones mexicanas con el patrocinio y coordinación del extinto Instituto Mexicano de Telecomunicaciones. Contó con la participación de instituciones públicas de gran prestigio en México: el Instituto Politécnico Nacional (IPN), el Instituto de Ingeniería de la Universidad Nacional Autónoma de México (UNAM), el Instituto de Geografía de la UNAM, el Centro de Investigación y Desarrollo de Tecnología Digital (CITEDI), el Centro de Investigación Científica y de Educación Superior de Ensenada (CICESE), el Centro de Investigaciones Matemáticas (CIMAT), la Benemérita Universidad Autónoma de Puebla, el Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), [1].

El objetivo de SATEX fue la construcción y validación de un microsatélite experimental de bajo costo y así, consolidar la experiencia, el conocimiento y la capacidad adquirida con la preparación de varios ingenieros e investigadores que participaron en los proyectos de los satélites Morelos, Solidaridad; gente que, incluso, se preparó en el extranjero. [2]

La finalización de las actividades encargadas al Instituto de Ingeniería fue en 2005 cuando se dieron por terminados los sistemas de la computadora de vuelo, el subsistema de sensores, protocolos de comunicaciones, el software de vuelo y el software de estación terrena encargado de monitorear el satélite. Desafortunadamente, el satélite nunca fue lanzado debido a que no fue completamente terminado.

1.2 SATélite EDUcativo (SATEDU)

La experiencia obtenida durante el proyecto SATEX y los anteriores, muestra que para reducir la brecha tecnológica existente entre México y países de primer mundo, se necesita capacitar una gran cantidad de recursos humanos, además de decenas de personas trabajando en diferentes partes del proyecto.

La idea inicial de este satélite de laboratorio, fue el contar con una plataforma satelital con fines didácticos, de enseñanza y de desarrollo de tecnología satelital, atendiendo a la gran necesidad de transmitir de una forma sencilla y clara, los principales conceptos de la tecnología satelital a alumnos de niveles desde medio superior hasta posgrados. De esta manera surge SATEDU, acrónimo de SATélite EDUcativo, el cual es una plataforma de enseñanza para capacitación en tecnología espacial, única en su género en México y en el mundo.

Este satélite fue diseñado, construido y validado totalmente por un grupo multidisciplinario de estudiantes y académicos en el Instituto de Ingeniería de la UNAM.

1.2.1 Arquitectura de SATEDU

Los subsistemas de SATEDU se diseñaron conforme al estándar CubeSat. Las tarjetas electrónicas que integran a cada subsistema tienen una dimensión de 9 x 9 cm, formando un módulo que tiene las dimensiones de un contenedor de CDs, con un peso aproximado a 1 kg. El sistema SATEDU cuenta con todos los subsistemas que integran a un satélite pequeño real de vuelo espacial, incluso integra esquemas de protección contra el efecto *latch-up,* que se genera en ambiente espacial.

SATEDU constituye una arquitectura de cómputo distribuido, en la cual cada tarjeta electrónica incluye un microcontrolador que se encarga de interactuar, mediante conectores laterales, con la computadora de vuelo para la administración de tareas, suministro de energía y comunicación entre los demás subsistemas. En la FIGURA 1.1 se muestra SATEDU.

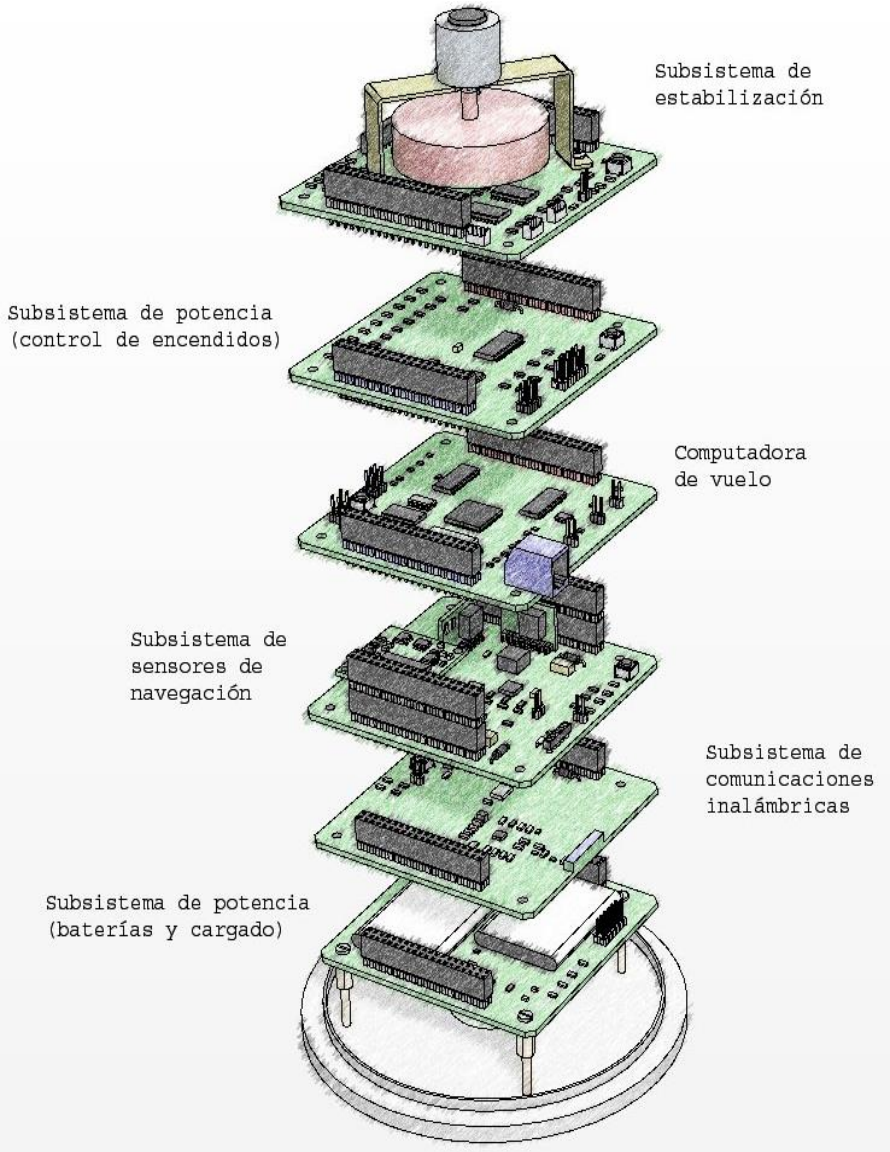


Figura 1.1. Arquitectura de SATEDU.

Subsistema de Computadora de Vuelo (CV)

La Computadora de Vuelo es una tarjeta electrónica que se encarga de coordinar las tareas de los demás subsistemas del satélite, en pocas palabras, es el cerebro del satélite, por ejemplo: gestiona la comunicación del sistema y entre subsistemas, su encendido y apagado, lectura de sensores, manejo de recursos de memoria, entre otras.

La CV está compuesta por un microprocesador SAB80C166 de Siemens, cuenta también con un microcontrolador PIC16F876A, una Memoria Flash de 32MB, 3 sensores de

temperatura y 3 arreglos de protección contra efecto *latch-up*. La FIGURA 1.2 muestra la tarjeta diseñada y fabricada para este subsistema.

Figura 1.2. Computadora de Vuelo de SATEDU.

Subsistema de Potencia (SP)

El Subsistema de Potencia controla el encendido y apagado de las fuentes de energía de los demás subsistemas mediante comandos recibidos desde la CV. El SP consta de dos tarjetas, la principal integra un microcontrolador PIC18F2321 y electrónica que regula y administra la energía a los subsistemas. La secundaria contiene un banco de 4 baterías de Litio-ion recargables, y un módulo de recarga. La FIGURA 1.3 muestra las tarjetas diseñadas para este subsistema.

Figura 1.3. Subsistema de Potencia de SATEDU.

Subsistema de Estabilización (SE)

El subsistema está conformado por una tarjeta que maneja dos métodos de estabilización activa: rueda inercial y bobinas de torque magnético. La rueda inercial es una pequeña masa circular sujeta a un motor, mientras que las bobinas de torque magnético son embobinados, uno en cada eje, para que en conjunto con la rueda inercial realicen maniobras de estabilización de SATEDU. La FIGURA 1.4 muestra la tarjeta diseñada y fabricada para este subsistema.

Figura 1.4. Subsistema de Estabilización de SATEDU.

Subsistema de Comunicaciones (SC)

El Subsistema de Comunicaciones está constituido por 2 tarjetas, ambas con circuitos integrados de RF empleados como medio de enlace de comunicación bidireccional, esto con el objeto de transferir datos de telemetría y comandos entre la estación terrena y SATEDU. Una de las tarjetas se encuentra conectada directamente a la PC que funciona como estación terrena, y la otra está integrada en SATEDU. La FIGURA 1.5 muestra la tarjeta diseñada y fabricada para este subsistema.

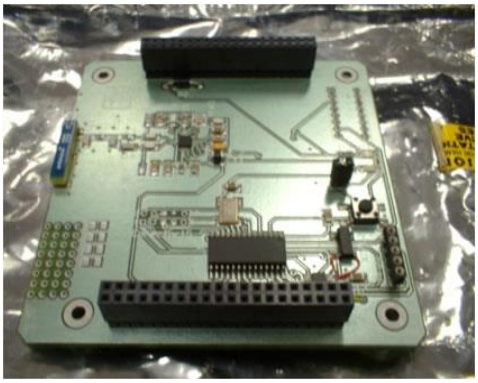


Figura 1.5. Subsistema de Comunicaciones de SATEDU.

Subsistema de Sensores de Estabilización (SSE)

El SSE monitorea el movimiento de SATEDU con base en la lectura de los sensores integrados en la tarjeta, algunos de ellos sirven para visualizar en la PC su posición en tiempo real. Los sensores que la componen son una brújula electrónica, un acelerómetro triaxial y 3 giróscopos electrónicos colocados de manera ortogonal para monitorear los 3 ejes del satélite. La FIGURA 1.6 muestra la tarjeta fabricada para este subsistema.

Figura 1.6. Subsistema de Sensores de Estabilización de SATEDU.

1.3 Resumen

SATEDU es una potente plataforma de enseñanza y desarrollo de tecnología satelital de bajo costo que, además de permitir a estudiantes de nivel medio superior y posgrados introducirse y capacitarse en esta línea de trabajo, posee la flexibilidad de integrar nuevos módulos o subsistemas con relativa facilidad, gracias a la arquitectura de hardware de SATEDU.

En el siguiente capítulo se da una breve introducción sobre los dispositivos FPGAs, pieza fundamental de la plataforma (SMIN_V2), posteriormente se aborda la descripción de hardware y software que integra la nueva plataforma propuesta que podrá ser integrada en SATEDU.

2

Aspectos generales sobre FPGAs

A principios de los años 80's se fabricaron los primeros circuitos integrados que contenían arreglos de compuertas lógicas con circuitos adicionales que al habilitar las conexiones indicadas permitían formar el circuito requerido. A estos dispositivos se les conoce como circuitos lógicos programables (PLDs por sus siglas en inglés). Psteriormente se incluyeron fip – flop's para tener máquinas de estados en un solo chip.

Hoy en día la complejidad de los dispositivos lógicos programables ha permitido adaptar sistemas completos en un chip o *System on a Chip* (SoC).

Los FPGAs son básicamente circuitos digitales programables con bloques lógicos e interconexiones reconfigurables.

En éste capítulo se introduce al FPGA, mencionando sus principales características y el por qué de la elección de éste dispositivo sobre otros que existen actualmente en el mercado, al finalizar la sección, se tendrá una idea general de su lógica de funcionamiento y arquitectura. ¿Qué es un FPGA?

2.1 ¿Qué es un FPGA?

El FPGA (del inglés *Field Programmable Gate Array*) es un dispositivo semiconductor que contiene bloques (o celdas) lógicos programables y una jerarquía de interconexión reconfigurable. La forma de definir el comportamiento del FPGA se hace mediante el lenguaje de descripción de hardware (HDL) o de forma esquemática. Una vez completado el diseño y sintetizado, se transfiere la configuración de bloques lógicos y conexiones al FPGA para obtener un circuitos personalizado.

De otra forma más sencilla se puede ver al FPGA como arreglos de celdas e interruptores programables. En pocas palabras, al programar el FPGA se describe la función de cada bloque o celda y la conexión con otros bloques o interruptores.

Existen (al menos) 4 compañías que fabrican FPGAs:

- Xilinx inventó dichos dispositivos y es el fabricante número uno.
- Altera es el segundo fabricante más conocido.
- Lattice y Actel son fabricantes menores.

2.2 Características básicas de FPGAs

Los ASICs (del inglés *Application Specific Integrated Circit)* son dispositivos personalizados para realizar alguna tarea en específico, estos tienen hileras y columnas con transistores que no están conectados entre sí inicialmente, tales conexiones se realizan al momento de implementar el circuito deseado como parte final del proceso de producción. A diferencia de los ASICs, los FPGAs pueden albergar varios circuitos hechos a la medida (mientras no exceda la cantidad de recursos) y reconfigurarlos sin necesidad del fabricante, de allí su nombre (en español): “Arreglo de compuertas programables en campo”.

El silicio reprogramable tiene la misma flexibilidad que un software que se ejecuta en un sistema basado en procesador, la diferencia radica en que el FPGA posee un procesamiento en paralelo, con ello se evita compartir recursos, esto es, cada tarea de procesamiento independiente se asigna a una parte del chip y puede ejecutarse de forma autónoma sin verse afectada por otros bloques de lógica.

Los FPGAs combinan lo mejor de los circuitos integrados de aplicación específica y sistemas basados en procesador, convirtiéndose en una plataforma de desarrollo flexible, reconfigurable y de alta capacidad de procesamiento.

2.3 Arquitectura de FPGAs

En los párrafos anteriores se comenta de forma general el concepto del FPGA y las principales características que lo diferencian de otros dispositivos. Considerando la complejidad de los FPGAs contemporáneos y la cantidad de textos que tratan su arquitectura y diseño, ésta sección será enfocada a proveer un vistazo de sus principales elementos. El objetivo es conocer los elementos de hardware básicos que componen los dispositivos FPGAs para hacer uso eficiente de sus recursos disponibles y potencializar sus capacidades.

Cada fabricante tiene una arquitectura propia para el FPGA pero en términos generales la arquitectura consiste de bloques lógicos programables/configurables, bloques de entrada/salida programables e interconexiones configurables (interruptores)

La estructura genérica de un FPGA se observa en la FIGURA 2.1.

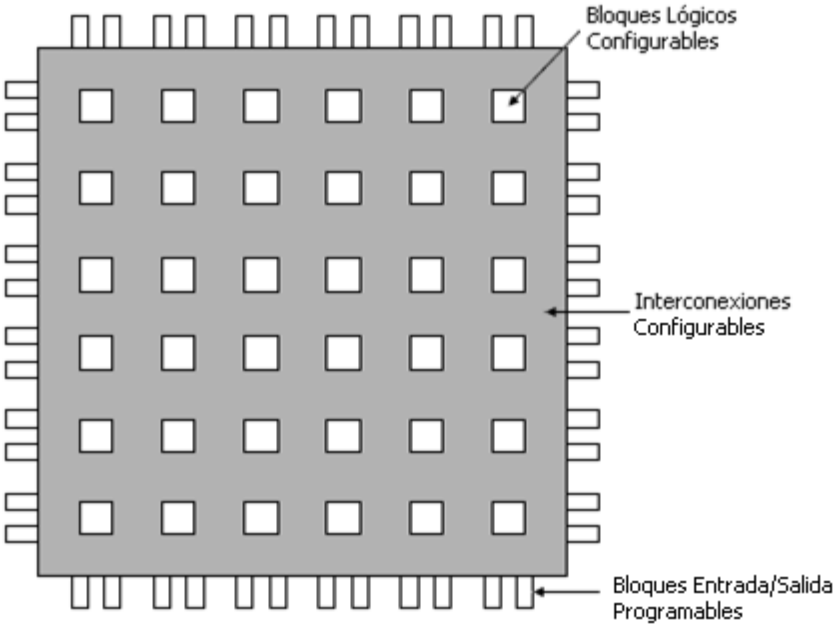


Figura 2.1. Elementos básicos dentro de FPGAs

2.3.1 CLBs (Configurable Logic Blocks)

Los bloques lógicos programables/configurables o mejor conocidos como *CLBs* son las unidades lógicas básicas de los FPGAs, la cantidad y sus características varían entre

dispositivos, sin embargo cada *CLB* puede ser generalizada como se muestra en la FIGURA 2.2, dichos bloques a su vez contienen elementos combinacionales (multiplexores y flip flop´s) y de memoria (SRAM).

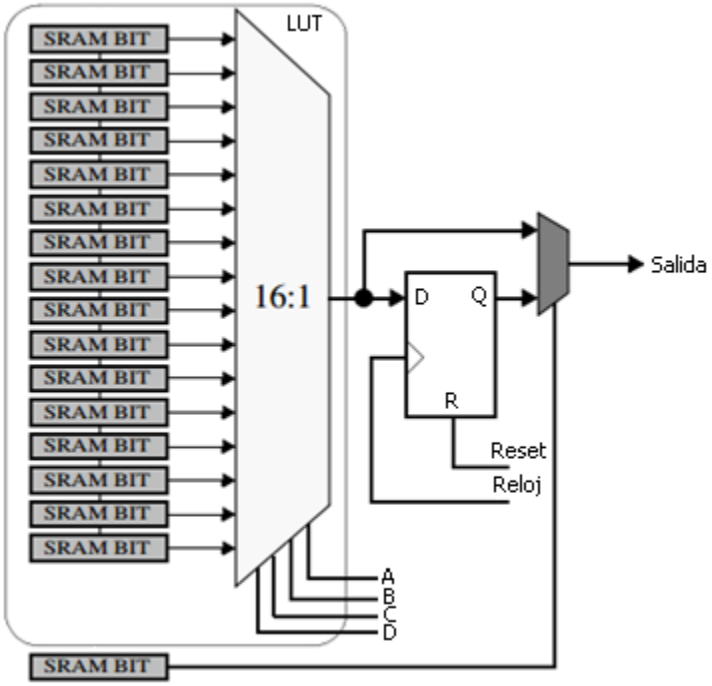


Figura 2.2. *CLB* simplificada de un FPGA

Analizando la imagen podemos observar una memoria SRAM en cada línea de un multiplexor, asociada a una localidad de memoria (LUT[1]), un flip-flop y finalmente un bloque de configuración que permite o no la salida del multiplexor.

Aunque un *CLB* realmente está constituido por entre 2 o 4 bloques como el anterior (*slice*), el diagrama de la figura 1.2 es suficiente para el desarrollo práctico de este trabajo.

Los CLBs se consideran generalmente como las unidades lógicas más pequeñas en los dispositivos comerciales disponibles de FPGAs.

2.3.2 Bloques de entrada/salida (IOB*)*

Los IOBs se encargan del flujo de datos desde y hacia el FPGA a través de los pines del chip. Cada bloque de entrada/salida provee una interfaz unidireccional o bidireccional programable.

[1] LUT. *Lookup table.* Es el recurso principal para implementar funciones lógicas dentro del FPGA.

El buffer de salida posee controles programables para habilitar salidas como *tri-state* u *open-collector* y *slew rate.*
El buffer de entrada cuenta con resistencias de *pull-up* o *pull-down* programables.

Los bloques pueden hacer uso de elementos de almacenamiento de salida y entrada. En la FIGURA 2.3 se tiene como ejemplo un diagrama de bloques simplificado de un bloque de entrada/salida.

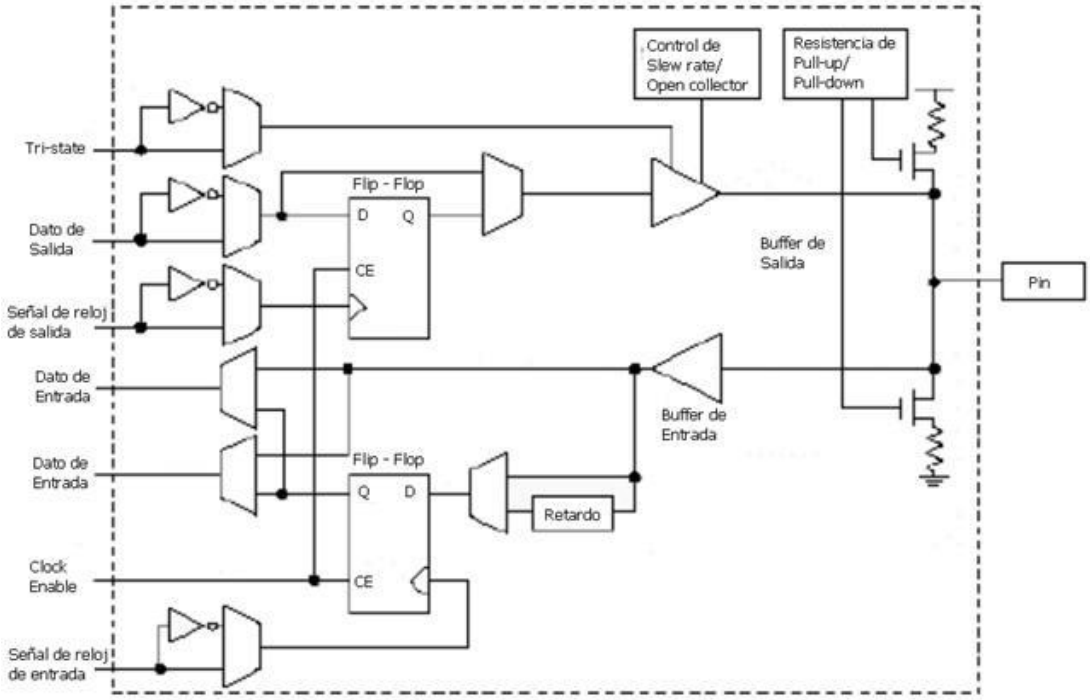


Figura 2.3. Bloque simplificado de entrada/salida

2.3.3 Interconexiones reconfigurables

La flexibilidad de un FPGA está directamente ligada a un equilibrio entre la creación de interconexiones y la eficiencia del área en uso. En el momento de crear canales de ruteo o interconexiones se incluye una jerarquía (long lines, hex lines, double lines, direct lines) para reducir retardos en la transmisión de señales. Los bloques de interconexión programables (interruptores) conectan internamente los elementos funcionales del FPGA y recursos externos.

Un ruteo eficiente tiene la mayoría de líneas de ruteo de forma local/directa/corta y un limitado número de conexiones que necesitan recorrer largas distancias. Por tanto, al proveer menor cantidad de recursos a las jerarquías más altas, la estructura de

interconexiones será más eficiente en el uso de recursos de ruteo y evitará los retardos en las señales.

La interconexiones son recursos presentes dentro de los FPGAs que, como su nombre lo indica, permiten la conexión *CLB*s-.*CLB*s y *CLB*s- *IOBs*. Las interconexiones se hacen posibles gracias a los canales de ruteo o de interconexión y/o a los bloques de interconexión (interruptores).

Los canales de ruteo o de interconexión constituyen una red programable de rutas para las señales de entrada y salida de los elementos funcionales dentro del FPGA. La FIGURA 2.4 muestra un ejemplo simplificado de conexión de un *CLB* y sus canales de ruteo.

El bloque de interconexión (interruptores) conecta los diferentes tipos de interconexiones a lo largo del dispositivo. FIGURA 2.5.

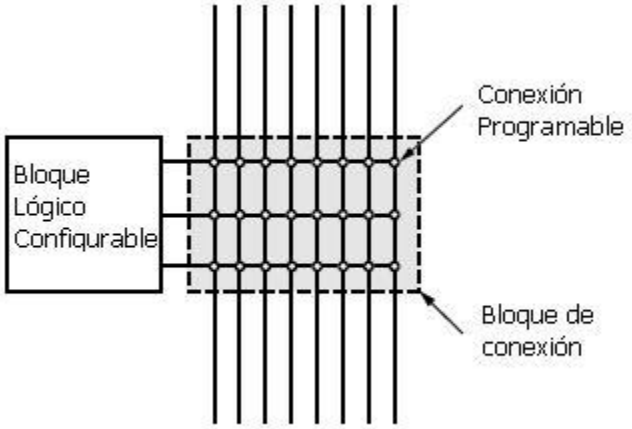


Figura 2.4. Bloque de conexión de un CLB.

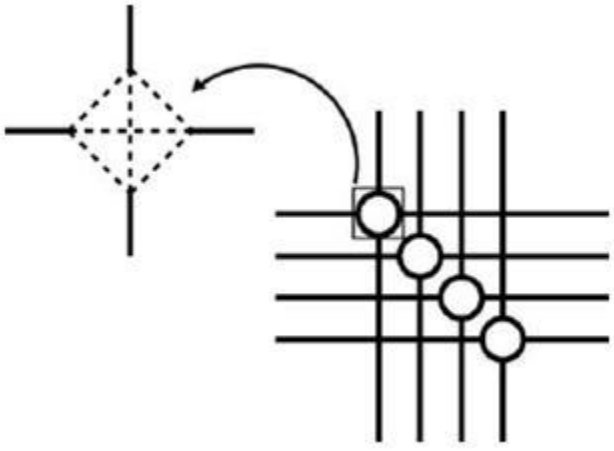


Figura 2.5. Imagen detallada de un bloque de interconexión.

2.3.4 RAM Embebida

La memoria se ha convertido en uno de los elementos de mayor importancia en el diseño de sistemas embebidos, los FPGAs incorporan bloques de memoria además de la SRAM de los LUTs.

La memoria junto con otros elementos potencializan la posibilidad de crear un sistema de cómputo totalmente funcional en un solo FPGA. [3]

Este tipo de memoria no retiene su contenido sin alimentación, lo cual quiere decir que el FPGA necesita ser configurado cada vez que se enciende. [4]

La arquitectura del FPGA varia respecto la marca y modelo del fabricante, los bloques de RAM embebida pueden localizarse en la periferia, en columnas o bien distribuidas a lo largo del dispositivo, FIGURA 2.6. Los bloques RAM se pueden programar combinando memorias y formar memorias de doble puerto (*dual-port RAM*) y memorias FIFO (*First In First Out*). [5] [6]

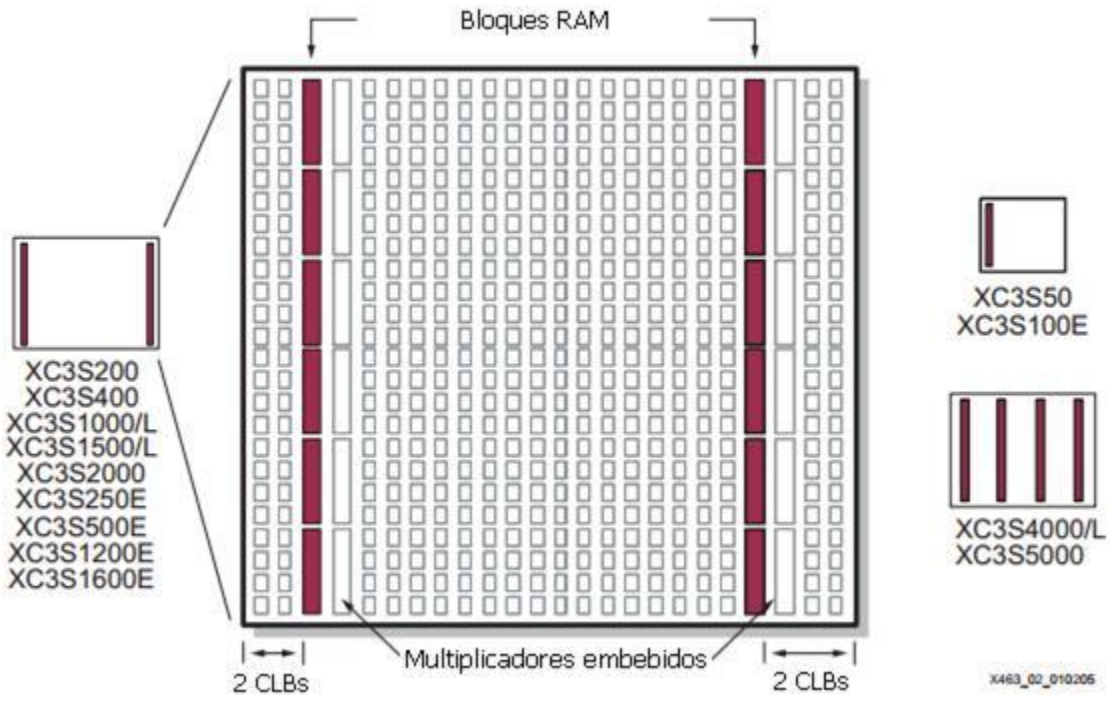


Figura 2.6. Bloques RAM en columnas de la familia Spartan 3 de Xilinx

2.3.5 Multiplicadores

Los multiplicadores potencializan la velocidad de procesamiento de algoritmos que usan multiplicaciones y sumas, con el mínimo uso posible de recursos del FPGA. Ciertamente, se podrían usar CLBs para crear multiplicadores, sin embargo, se tendría un retardo considerable en las operaciones y dependiendo de la aplicación, se ocuparía un gran número de CLBs. Es común encontrar multiplicadores cerca de los bloques de memoria RAM ya que procesan datos contenidos en ellos, FIGURA 2.7. [6] [7]

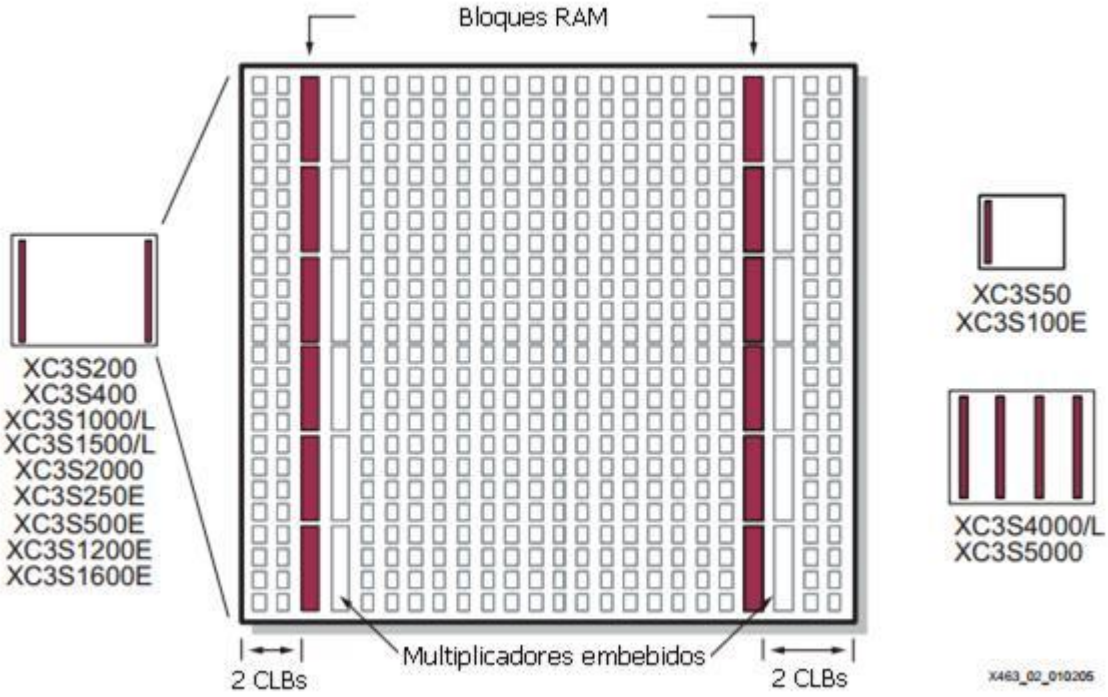


Figura 2.7. Multiplicadores contiguos a bloques RAM de la familia Spartan 3 de Xilinx

2.4 Programación de Arquitecturas

La programación de arquitecturas se refiere a la forma en la cual la información de la configuración se estructura dentro de un FPGA y cómo se programan los recursos del dispositivo. A continuación se mencionan brevemente las técnicas más comunes para realizar esta práctica.

SRAM. Esta tecnología tiene la ventaja de ser reprogramada tantas veces como el semiconductor lo permita, ideal para aplicaciones de computo reconfigurable. La desventaja radica en que el contenido se pierde una vez apagado el sistema.

Antifusible (*Antifuse*).Un FPGA que utiliza este mecanismo sólo puede ser programado una vez. Como su nombre lo sugiere, se comporta opuestamente a un fusible, se encuentra normalmente abierto y una vez programado se cierra.

EPROM, EEPROM y FLASH. Estas combinan las tecnologías de las dos anteriores, situándose en un punto intermedio, son reprogramables y no son volátiles, además, no requieren de un dispositivo auxiliar para guardar la configuración interna.

2.4.1 Aplicaciones

El cómputo reconfigurable (RC) está asociado directamente a FPGAs por las características antes descritas que ofrece el dispositivo sobre otras opciones.

Las primeras aplicaciones de cómputo reconfigurable fueron implementadas en el procesamiento de imágenes y señales, donde los FPGAs se emplearon para procesar gran parte de la señal, desde su paso por el convertidor A/D hasta obtener la FFT de la señal en cuestión, éste algoritmo se puede descomponer en una descripción de hardware y procesar paralelamente miles de muestras por segundo. Similar al proceso de imágenes, el cómputo reconfigurable se puede aplicar al procesamiento en tiempo real de video, en el área de seguridad de redes, en aplicaciones bio-informáticas, etc. Últimamente, el RC ha entrado al área de supercomputo, con FPGAs de alto desempeño, [5].

2.5 Resumen

Ahora que se cuenta con un panorama general de la arquitectura y los componentes existentes dentro de los FPGAs que, dependiendo de la familia varía la cantidad de recursos internos y su ubicación dentro del dispositivo, se cuenta con las herramientas necesarias para abordar del tema de tesis en cuestión.

En el próximo capítulo se expone brevemente el concepto de “Sistemas Embebidos”, las distintas plataformas donde se pueden encontrar dichos sistemas y se describe la integración del hardware de la plataforma propuesta, explicando sus características y capacidades del nuevo Sistema Mínimo (SMIN_V2).

3

Descripción del hardware de la plataforma FPGA

La llegada del microprocesador en los 70's potencializó los sistemas de control, permitiendo diseñar un sistema relativamente complejo usando un dispositivo (microprocesador) como el principal elemento de retroalimentación. Estos sistemas evolucionaron con el paso de los años, implementándose en distintas disciplinas con el fin de optimizar procesos y hacerlos cada vez más precisos y eficientes.

En este capítulo se presenta el proceso de diseño de la plataforma, partiendo de un concepto general, después separando los bloques funcionales que conforman el sistema hasta definir los requisitos que acotan la elección de los componentes.

Los sistemas digitales han evolucionado enormemente durante las últimas décadas debido a que la elevada densidad y velocidad de los dispositivos permiten plantear una gran gama de posibilidades al diseñar sistemas embebidos. Hoy en día existen microcontroladores de 8, 16 y 32 bits de procesamiento con compatibilidad en varios protocolos de comunicación (I^2C, CAN, SPI y UART). Para la mayoría de las aplicaciones un microcontrolador es adecuado, sin embargo, para aplicaciones donde se necesita adecuar lógica personalizada de forma rápida y con periféricos adicionales, la mejor opción la proporciona el FPGA por su flexibilidad, capacidad de procesamiento paralelo y el poder desarrollar computo reconfigurable.

3.1 Introducción a los Sistemas Embebidos Computacionales

Generalmente los Sistemas Embebidos computacionales son diseñados para realizar alguna tarea en particular, sin embargo, ahora vemos teléfonos celulares o televisores, por ejemplo, que son auténticos Sistemas Embebidos que integran funcionalidades adicionales para las que fueron diseñadas como revisar tu correo, navegar en internet y juegos, [8] [9].

La decisión acerca del tipo de plataforma computacional a usar se selecciona en la fase de diseño de la arquitectura del sistema, la cual, está íntimamente ligada con sus limitaciones operativas. Sin algún orden en particular se listan algunos tópicos que acotan el diseño de cualquier arquitectura embebida:

- Consumo de energía.
- Costo.
- Herramientas de programación disponibles.
- Facilidad de programación.
- Interacción en tiempo real con el exterior.
- Recursos (almacenamiento, comunicación).

A diferencia de una aplicación en una plataforma estándar, el diseño de un sistema embebido implica que el software y el hardware sean diseñados de forma paralela. Aunque no siempre es el caso.

El presupuesto y definición de requisitos del diseño fuerzan muy a menudo a la toma de decisiones antes que los diseñadores puedan seleccionar el diseño que mejor se ajuste al proyecto.

Sin importar la aplicación del sistema o bajo qué plataforma de cómputo se diseñará, el ciclo de vida del desarrollo de un sistema embebido sigue las siguientes siete premisas, [10]:

- Especificaciones del producto.
- Particionar el producto en componentes de software y hardware.
- Definir el número de iteraciones para depuración en hardware y software
- Tareas a ejecutar de hardware y software.
- Integración de los componentes de hardware y software.
- Pruebas del sistema, depuración y lanzamiento del producto.
- Mantenimiento y actualización.

3.2 Plataformas de cómputo

La electrónica digital incorporada en sistemas electrónicos, inicialmente en el microprocesador (1971), permitió por primera vez la configuración por software de plataformas digitales.

Microcontrolador/Microprocesador. Los sistemas digitales de control usan interrupciones para realizar algún procesamiento, cada interrupción en un sistema embebido de tiempo real está asociada con cierto límite en el tiempo de respuesta, tanto el microprocesador como el microcontrolador tratan dichas interrupciones de forma secuencial, por lo que es común que se implemente un administrador de tareas y se manejen de forma eficiente.

ASIC. Otra plataforma comúnmente encontrada para diseñar un sistema embebido son los ASICs, en ellos se pueden adaptar periféricos al diseño que se vaya a implementar para cerrar el proceso de producción y entregarlo al cliente. A pesar de que su costo unitario suele ser bajo en producción a grandes volúmenes, una vez cerrado el proceso de fabricación no es posible modificar su lógica interna.

CPLD. Los sistemas basados en CPLDs como unidad de proceso central son muy concurridos para diseños de alta velocidad y muy exactos en restricciones de tiempo de ejecución, su arquitectura es predominante en arreglos de compuertas AND – OR, los CPLDs son dispositivos de memoria no volátil, es decir, no se requiere de un dispositivo externo para su configuración. [17]

PSoC. Otra plataforma de desarrollo son los dispositivos PSoCs, la cual es tecnología que incorpora un microcontrolador, bloques con funciones lógicas y analógicas reconfigurable en un solo *chip*, inicialmente las herramientas de desarrollo permitían al usuario programar un PSoC de forma gráfica sin escribir una sola línea de código en C o ensamblador, sin embargo *Cypress* no recomienda hacer uso de dicha herramienta para desarrollo de sistemas en producción e incluyen compiladores de C y ensamblador totalmente compatibles con la herramienta gráfica. Los arreglos de matrices analógicas y digitales traslapadas hacen útil esta plataforma en tareas de procesamiento de señales mixtas, [11] [12] [13].

Los sistemas basados en plataformas FPGA poseen la capacidad de adaptar diseños hechos a la medida al modificar la lógica interna del dispositivo, restricción que la plataforma ASIC tiene; a diferencia de los microprocesadores y microcontroladores, el FPGA ofrece un procesamiento paralelo, el cual es totalmente independiente a cada tarea configurada en el dispositivo. La arquitectura de lógica combinacional incorporada en los FPGAs permite alojar diseños complejos y con gran capacidad de procesamiento de información, sin embargo, la mayoría de los FPGAs necesitan ser configurados externamente cada arranque o encendido del sistema, desventaja ante la tecnología CPLD, mientras los dispositivos PSoC tiene funcionalidades, bloques digitales y memoria programable limitados, los dispositivos FPGA se pueden configurar para optimizar los recursos integrados y adaptarlos de acuerdo a las necesidades del sistema.

3.3 Diseño usando FPGAs

Los FPGAs de hoy en día ofrecen una plataforma que soporta núcleos de procesamiento embebidos (Co-procesadores, periféricos o microprocesadores), además han incorporado una gran variedad de componentes digitales (memoria, multiplicadores, transceptores y muchos más) en un solo chip.

La disponibilidad de realizar diseños de alta densidad y su bajo costo han hecho del FPGA una herramienta con mucha flexibilidad al montar arquitecturas digitales personalizadas, y sobre todo, viable para desarrollar varias aplicaciones, por ejemplo, procesamiento de señales, audio, video, multimedia, comunicaciones, cómputo de alto desempeño adquisición y procesamiento de datos, entre otras.

A continuación se enlistan las principales características de los FPGAs:

- Reconfigurabilidad. Los FPGAs son dispositivos que pueden ser configurados en campo, de allí su nombre de Arreglos de Compuertas Programables en Campo (*Field Programmable Gate Arrays*), las cuales pueden agregar o modificar su arquitectura en cualquier momento.
- Diseño definido por software. El hardware es definido por lenguajes de descripción de hardware (HDL).
- Paralelismo. Los circuitos en un FPGA pueden ser diseñados en una estructura paralela, donde cada tarea de procesamiento se ejecuta de forma aislada, sin verse afectada por otros bloques de lógica.
- Alta velocidad. La plataforma FPGA cuenta con velocidades altas de reloj, aunado con procesamiento paralelo, puede superar a sistemas basados en procesadores.
- Confiabilidad. No hay un sistema operativo ni una capa entre el software y hardware que afecte la operatividad del FPGA, sólo se monta la arquitectura que describe el comportamiento de hardware.

- Seguridad y re-uso. Una vez implementada la arquitectura, es complicado revertir el proceso. Una vez verificada la arquitectura puede ser re-usada en un futuro.

La integración de componentes digitales, procesadores e interfaces de comunicación en un solo dispositivo hacen potencialmente viable elegir al FPGA para el desarrollo de un Sistema Embebido. La disponibilidad de herramientas de software para diseñar, simular, sintetizar, probar y programar FPGAs ofrecen a ingenieros y desarrolladores elegir la metodología que mejor se ajuste a su estilo de trabajo e intereses; migrar entre ambientes de desarrollo de fabricantes de FPGAs es relativamente simple ya que el proceso de diseño es similar.

3.4 Bloques funcionales requeridos para uso de la plataforma FPGA

Dentro las principales ventajas que ofrecen los sistemas basados en plataformas FPGAs, está la flexibilidad de reconfiguración de arquitecturas lógicas implementadas en ellos. Contar con un sistema reconfigurable en tiempo de ejecución, robustece la plataforma.

La tendencia de mejora en el área de desarrollo de satélites pequeños y el avance en la tecnología de circuitos integrados convergen a la posibilidad de integrar sistemas completos capaces de desempeñar tareas cada vez de mayor complejidad.

En el IINGEN surgió la necesidad de actualizar el sistema de control y orientación, de una plataforma basada en un microcontrolador de propósito general a una plataforma basada en FPGA. Persiguiendo la facilidad de realizar pruebas experimentales sobre una plataforma de simulación de vuelo satelital basada en una mesa suspendida en aire, que refleje con gran aproximación el comportamiento que tendría el sistema integrado a la plataforma de un satélite real una vez que haya sido puesto en órbita.

En esta tesis se toma como referencia el Sistema Mínimo (SMIN_V1), ver FIGURA 3.1, previamente validado y completamente operativo, y propone la integración de nuevos componentes que permiten la evolución del sistema para obtener una plataforma con los medios necesarios en hardware y software que permita la reconfiguración total de la arquitectura de cómputo descrita en el FPGA y realizar pruebas de validación de control de orientación en tres ejes.

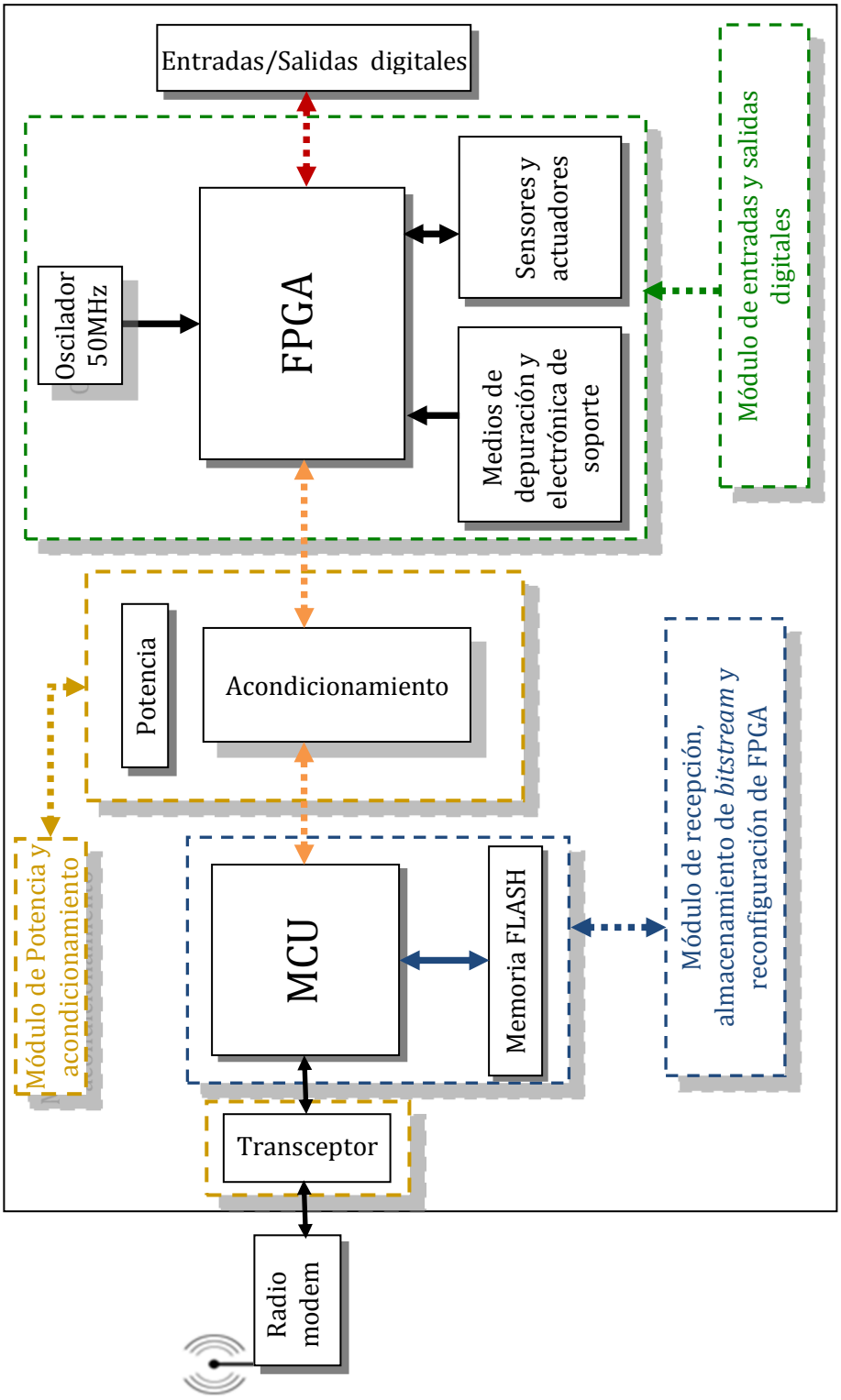


Figura 3.1. Diagrama de bloques de Sistema Mínimo Versión 1 (SMIN_V1).

A continuación se abordan los dos bloques principales que conforman el Sistema Embebido de esta tesis.

3.4.1 Estación Terrena

La Estación Terrena, como su nombre lo indica, es la unidad que interactúa directamente con el personal encargado de modificar o actualizar, transmitir y recibir información, controlar el estado del satélite y su situación orbital. Para fines de este trabajo nos referimos a la Estación Terrena para la transmisión del *bitstream* y recibir el estatus de la recepción de la información.

Para llegar a la transmisión del *bitstream* se requieren una serie de pasos secuenciales. Estos son:

- Definir la arquitectura de hardware.
- Obtención del archivo .xsvf.
- Creación de archivo en formato IntelHEX.
- Procesarlo en el CPU.
- Transmitirlo vía RS232.

La FIGURA 3.2 muestra de forma gráfica los pasos descritos en los puntos previos.

Figura 3.2 Generación y transmisión del *bitstream*.

En esta sección se mencionan de forma general los componentes que conforman a la Estación Terrena, en el próximo capítulo se tratará más a detalle este módulo, que es dedicado particularmente al software de operaciones de la plataforma FPGA propuesta.

3.4.2 Plataforma de desarrollo (SMIN_V2)

El Sistema Mínimo persigue la transferencia a hardware de algoritmos de control, incluyendo el diseño de una tarjeta completamente compatible con el satélite educativo SATEDU. Descrito en otras palabras, la actividad primordial es recibir, procesar y gestionar la información recibida desde la Estación Terrena para almacenarla, transmitirla y cargarla finalmente en el FPGA. Para comprender mejor la estructura del sistema, se divide en cuatro grupos básicos:

- Recepción y almacenamiento.
- Reconfiguración de FPGA.
- Alimentación o Potencia.
- Entradas y Salidas digitales.

La operación del sistema se puede ver de mejor manera en la FIGURA 3.3. El sistema recibe de forma inalámbrica la información para la reconfiguración del FPGA, posteriormente es tratada y almacenada en una memoria, una vez finalizada la sesión de transmisión/recepción de datos, se extrae de la memoria y es transferida al FPGA.

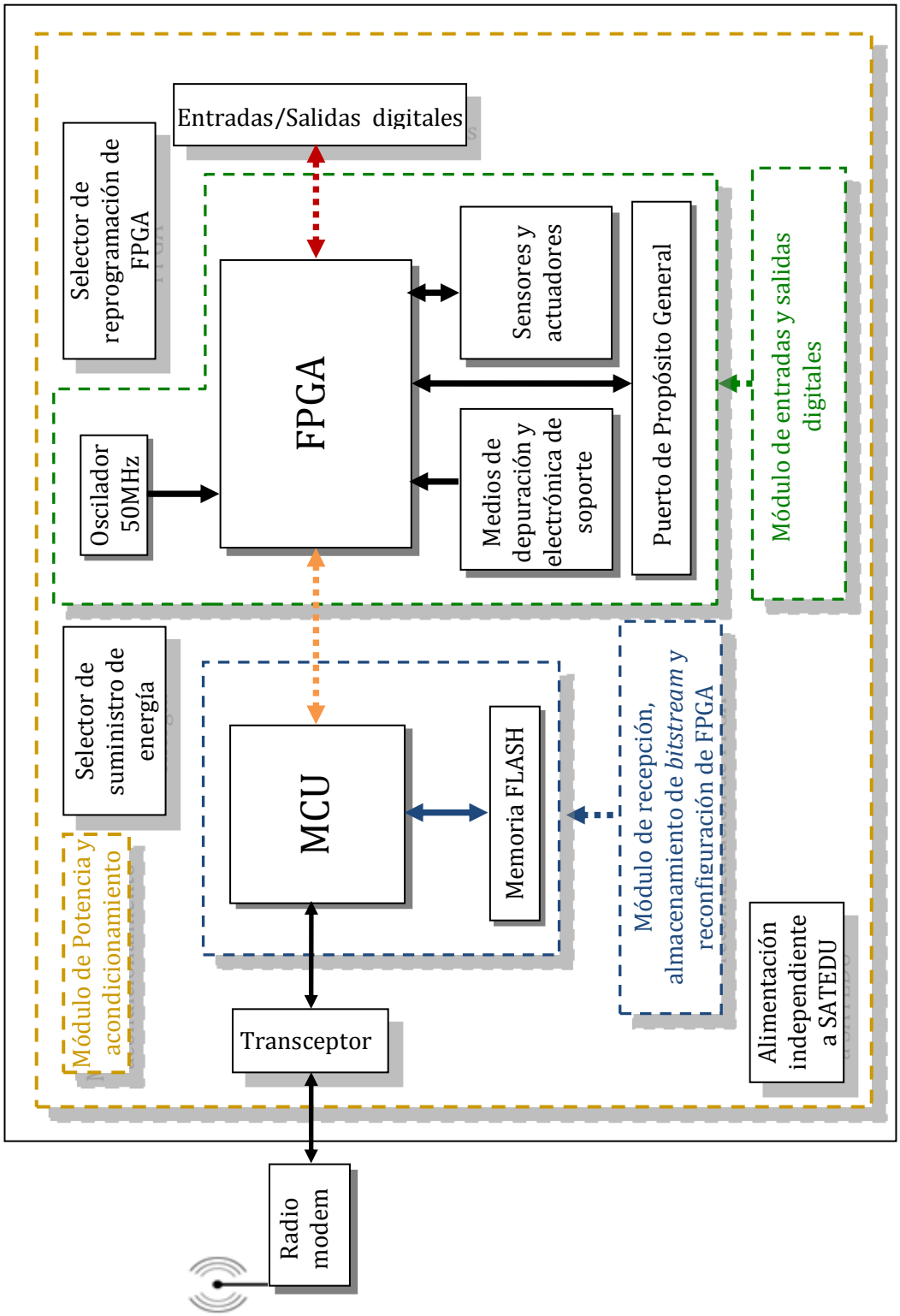


Figura 3.3. Bloques funcionales de la plataforma SMIN_V2.

Dentro del módulo de FPGA se encuentran los "medios de depuración y electrónica de soporte", el cual es útil para facilitar la detección y depuración de errores del sistema. Básicamente contiene LED´s y un interruptor (*dip siwtch*).

La plataforma contiene una sección de electrónica de soporte que se integra buscando albergar arquitecturas variadas y de distintos propósitos. Se incluye dentro del bloque de reconfiguración debido a que se puede hacer uso de ella con la habilitación de líneas de comunicación propias del FPGA.

Recepción, almacenamiento y reconfiguración de FPGA

Recepción y almacenamiento

Conforme el *bitstream* es recibido, el microcontrolador (PIC de Microchip) se encarga de decodificarlo de formato IntelHEX a formato xsvf, archivo que configura el FPGA. Posterior al procesamiento del *bitstream,* se almacena los datos obtenidos a una memoria FLASH. La FIGURA 3.4 ayuda a entender el flujo de datos de esta etapa.

El microcontrolador opera como la unidad central de control y posee la ventaja de contar con interfaces de comunicación implementadas como EUSART, SPI, I^2C, entre otras y compatibles con otros dispositivos estándares de propósito general (*ASSP*). Esto permite la reducción de desarrollo de código para la implementación del proceso de reconfiguración del FPGA.

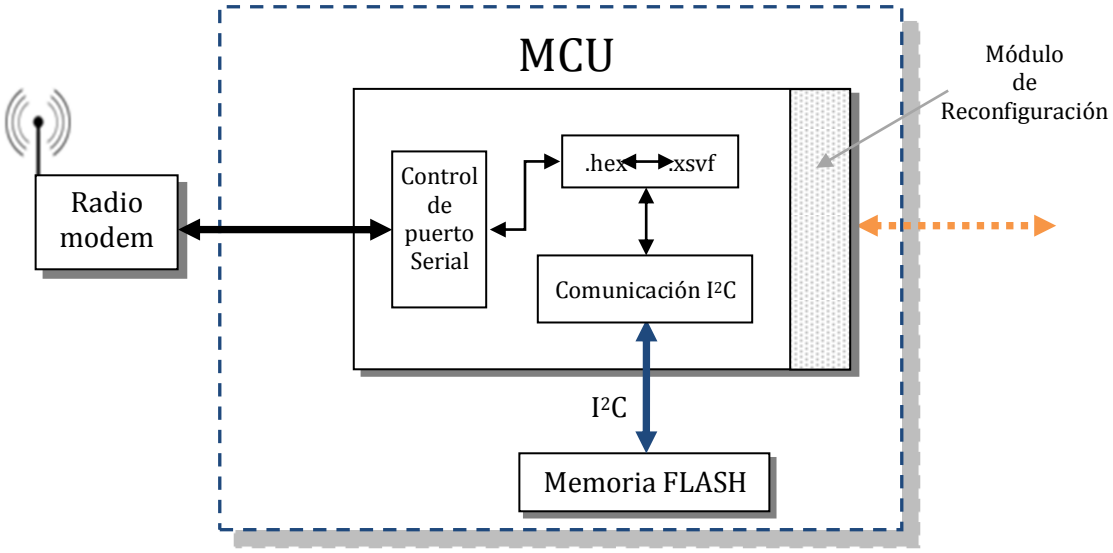


Figura 3.4. Bloques operativos de recepción y almacenamiento de *bitstream*.

FPGA XC3S1600E

Se utilizó un dispositivo con empaquetado FG320 que tiene 320 terminales y opera con señales de voltaje de 3.3V, 2.5V, 1.8, 1.5 y 1.2V. Posee ocho gestores de reloj digital DCMs, que operan en un intervalo de frecuencias que van desde 5MHz hasta 300MHz utilizando un oscilador externo (en el caso de este sistema opera con un oscilador de 50MHz) con divisores, multiplicadores y sintetizadores de frecuencia, ocho señales de reloj globales y ocho señales de reloj designadas para cada mitad del dispositivo. Integra una memoria RAM rápida de 648Kbits y 231Kbits de memoria RAM distribuida. Se compone por puertos designados para la configuración por comunicación JTAG IEEE 1149.1/1532, Master Serial, Slave serial, Master parallel Up y Down y SPI *Serial Flash*. Otros recursos internos del dispositivo se muestran en la TABLA 3.1.

			CLB Array (One CLB = Four Slices)									
Device	System Gates	Equivalent Logic Cells	Rows	Colums	Total CLBs	Total Slices	Distributed RAM bits	Block RAM bits	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differencial I/O Pairs
XC3S1600E	1600K	33.192	76	58	3.688	14.752	231K	648K	36	8	376	156

Tabla 3.1. Recursos internos del FPGA XC3S1600E, [14].

Microcontrolador PIC 18LF2520

Microcontrolador PIC de 8 bits y montaje superficial de la familia 18 de Michochip, cuenta con 1.5 kBytes en memoria, 256 kBytes de EEPROM y 32 kBytes de memoria FLASH programable. Posee un oscilador interno que trabaja a ocho frecuencias dentro del rango de 31kHz a 8MHz, 10 canales dedicados para conversión analógica/digital y es totalmente compatible con dispositivos de comunicación serial EUSART, SPI e I^2C en modo maestro y esclavo.

Se determinó utilizar este microcontrolador ya que cuenta con características prácticamente idénticas al utilizado en la primera versión del Sistema Mínimo V1 (SMIN_V1) pero con diferencias significativas, las de las más importantes son en el empaquetado, cantidad de pines y rango de voltaje de operación. El empaquetado SOIC (*Small Outiline Integrated Circuit*) es de montaje superficial, de dimensiones reducidas al contener menor número de terminales y operar a bajo nivel de voltaje (3.3V en este caso).

La TABLA 3.2 muestra las características de ambos microcontroladores.

Característica	18F4520	18LF2520
Frecuencia de operación	Hasta 40MHz	Hasta 40MHz
Memoria programable [bytes]	32728	32728
Memoria [bytes]	1536	1536
Memoria EEPROM [bytes]	256	256
Pines	40	28
Empaquetado	PDIP	SOIC
Modulo A/D (canales)	13	10

Tabla 3.2. Comparación entre microcontroladores.

Memoria FLASH SST25VF064C

Memoria compatible con comunicación serial SPI, con 64 Mbits de almacenamiento y páginas de 256 bytes, opera a una frecuencia máxima de 80 MHz y voltaje entre 2.7 y 3.6 V. El dispositivo reúne características importantes para la nueva versión del Sistema Mínimo V2 (SMIN_V2) como capacidad de almacenamiento, *buffer* y compatibilidad eléctrica.

La TABLA 3.3 muestra las características generales de la memoria integrada en la plataforma SMIN_V1 y su evolución SMIN_V2.

Característica	24LC1025	SST25VF064C
Voltaje de operación [V]	2.5 - 5.5	2.7 - 3.6
Protocolo de comunicación	I2C	SPI
Página (buffer)	128 bytes	256 bytes
Capacidad de almacenamiento	1024 bit	64 Mbits
Frecuencia de operación	Hasta 400 kHz	Hasta 80 MHz

Tabla 3.3. Características principales de memorias 24LC1025 y SST25VF064C.

El estándar de comunicación SPI permite mayor velocidad en la transferencia de datos, sin embargo requiere de 4 líneas de comunicación: SCK, SI, SO y CE.

- SCK (*Serial Clock*): Señal de referencia para transferir información desde y hacia la memoria.
- SI (*Serial Data Input*): Señal de entrada para transferir hacia la memoria comandos, direcciones y datos.
- SO (*Serial Data Output*): Señal de salida de datos hacia otros dispositivos.
- CE (*Chip Enable*): Señal que selecciona el dispositivo.

En las FIGURA 3.5 y FIGURA 3.6 se pueden ver las secuencias para la escritura y lectura de la memoria FLASH.

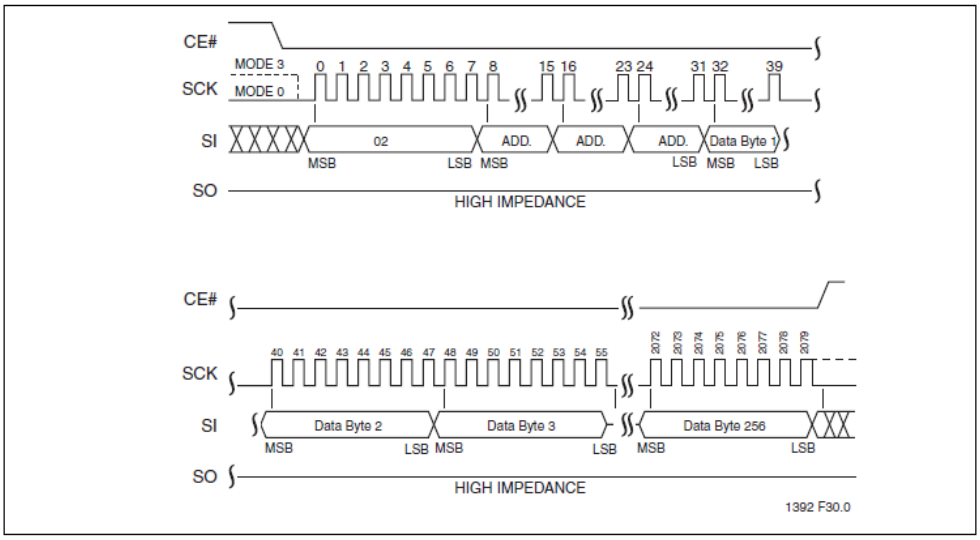


Figura 3.5. Secuencia de escritura de la memoria, [15].

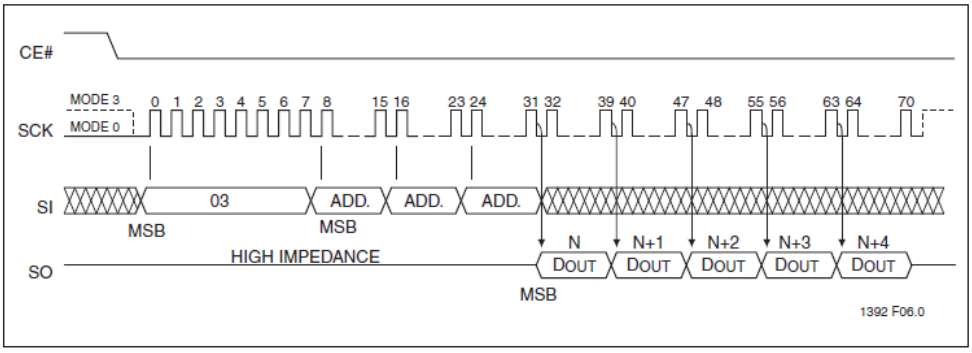


Figura 3.6. Secuencia de lectura de la memoria, [15].

Reconfiguración de FPGA

En este punto, como se muestra en la FIGURA 3.7 el microcontrolador recibe el comando de "reconfiguración" desde la Estación Terrena e inicia la extracción de la trama de datos almacenada en la memoria, la procesa y ejecuta las instrucciones grabadas en el archivo .xsvf. El archivo .xsvf contiene la información que se quiere transferir al dispositivo, [16].

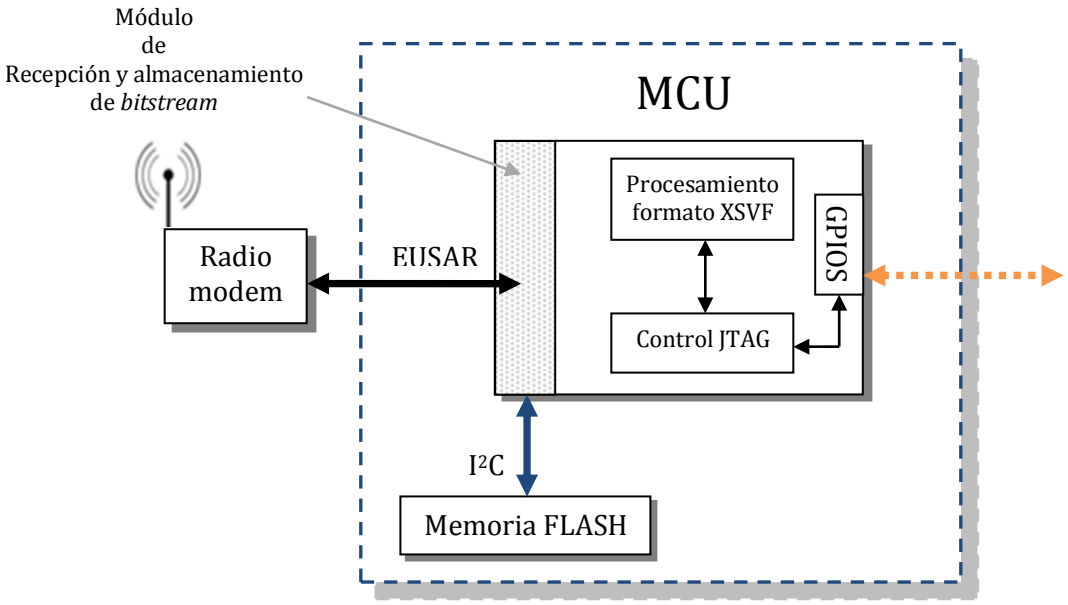


Figura 3.7. Bloques operativos de la etapa de reconfiguración.

La nueva versión de la tarjeta de reconfiguración hereda la lógica de carga de *bitstream* por el protocolo JTAG, implementada en la actual plataforma de desarrollo (SMIN_V1), existen cuatro fuertes razones para continuar sobre esta línea:

1. Las dimensiones físicas.
2. Implementación 100% funcional en SMIN_V1.
3. Menor tiempo para compatibilizar firmware con nuevos componentes electrónicos.
4. Material bibliográfico de soporte.

Protocolo JTAG

JTAG (*Joint Test Action Group) es* un método que se estandarizó como la norma IEEE 1149.1, estándar comúnmente usado para eliminar fallas en tarjetas complejas multicapa y con elementos de montaje superficial. Una gran característica del estándar es la posibilidad de la conexión de una cadena de dispositivos usando el mismo puerto de configuración, como se muestra en la FIGURA 3.8. [17]

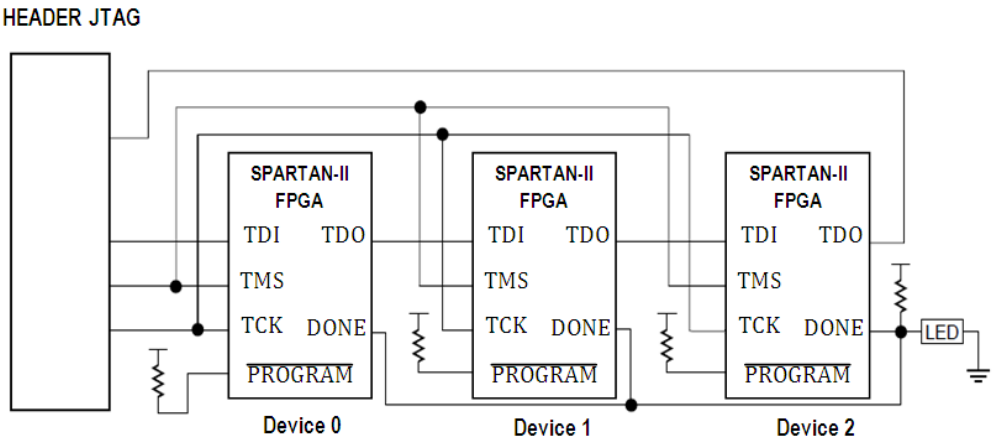


Figura 3.8. Dispositivos en cadena *Boundary-Scan*, [17].

La mayoría de dispositivos de la familia Xilinx son compatibles con el estándar IEEE 1149.1 *Test Access Protocol (TAP)* y la arquitectura *Boundary-Scan*, mejor conocido como JTAG. El estándar ofrece un medio para asegurar la integridad de cada componente y la interconexión entre ellos.

En particular el FPGA propuesto (XC3S1600E) incorpora en el chip todos los elementos definidos en el estándar IEEE 1149.1: TAP, controlador del TAP, el registro de instrucciones, el decodificador de las instrucciones, el registro de *Boundary-Scan*, *bypass,* incluso, el registro *USERCODE*. [17]

Entradas y Salidas digitales

Dentro de este bloque se encuentran terminales de entrada y salida conectadas directamente al FPGA con el propósito de habilitar la comunicación con otros dispositivos y validar el óptimo funcionamiento de las arquitecturas cargadas.

Las arquitecturas cargadas son bloques descritos en hardware (núcleos IP) que contendrán módulos de control, procesamiento y comunicación.

Sensores y actuadores

La plataforma de desarrollo tendrá habilitados puertos para adquirir señales provenientes de sensores de navegación y/o otros dispositivos para procesarlas mediante un bloque descrito en hardware (VHDL).

Los sensores proveerán de información a algoritmos de control de orientación cargados en la tarjeta FPGA desarrollada que realizarán cálculos y modificarán la dirección de navegación mediante la interacción de actuadores basados en ruedas inerciales.

Alimentación

Todo sistema electrónico requiere de energía eléctrica para su funcionamiento, debido a que tanto la plataforma desarrollada SMIN_V2 está diseñada para trabajar a bordo de SATEDU y de forma independiente, las restricciones en potencia son cruciales para el sistema embebido.

Por un lado se tiene un transceptor para compatibilizar los niveles de voltaje de las señales de comunicación entre la PC de la Estación Terrena y el MCU a bordo de la plataforma, y cuatro resistencias que sirven para acondicionar las señales de configuración JTAG entre el MCU y el FPGA.

Por otro lado se tienen tres reguladores de voltaje: 3.3V, 2.5V y 1.2V, estos elementos suministran energía eléctrica al sistema, con ello permitirá operar a la nueva tarjeta de forma independiente a SATEDU. El regulador de 3.3V ajusta el voltaje suministrado por una fuente externa cuando la tarjeta se opere fuera de SATEDU y lo distribuye a los demás elementos, el regulador de 3.3V es necesario ya que el microcontrolador y la memoria operan a ese nivel de voltaje. Los reguladores de 2.5V y 1.2V se emplean para el FPGA.

Todos los dispositivos de la familia Spartan-3E operan con tres niveles de voltaje. Dos de ellos son requeridos para la operación interna del FPGA: V_{CCINT} y V_{CCAUX}, [18].

V_{CCINT} alimenta a los elementos internos del FPGA designados a desempeñar funciones lógicas como: multiplicadores, el bloque de RAM embebida y CLBs. Será necesario un regulador de 1.2V para suministrar energía a esta terminal. [18]

V_{CCAUX} alimenta a elementos auxiliares del FPGA como: los bloques controladores del reloj (DCMs), pines configurables y al bloque JTAG. Se requiere de un regulador de 2.5V y alimentar a los elementos auxiliares necesarios. [18]

La tercer entrada es V_{CCO}, que alimenta a los *buffers* de los bloques de entrada y salida (IOBs) asociados a cada banco del FPGA. Todos las terminales V_{CCO} deben de estar conectadas al mismo nivel de voltaje de 3.3V para ofrecer compatibilidad con elementos externos con los que interactúa, ya que tanto la memoria y el MCU operan a un nivel de voltaje de 3.3V. [18]

La FIGURA 3.9, ilustra la distribución de los diferentes niveles de voltaje en cada banco que utilizan los FPGAs de la familia Spartan 3E.

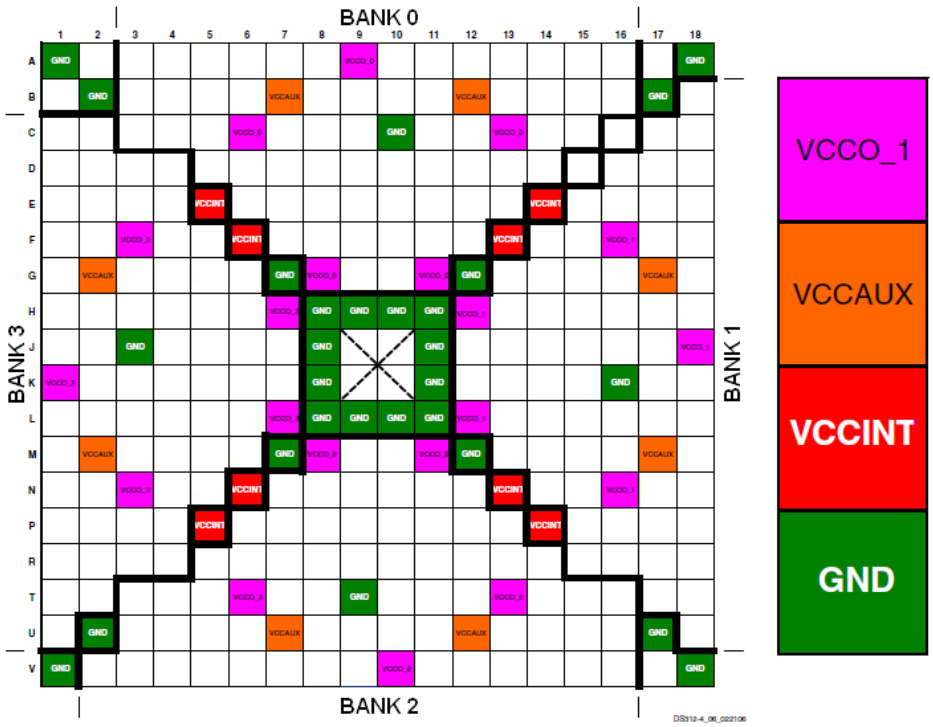


Figura 3.9. Distribución de voltajes y bancos del FPGA (XC3S1600E), [14].

3.5 Resumen

En este capítulo se presentó la idea de los elementos y características generales que se encuentran en un Sistema Embebido, además de las distintas plataformas comúnmente encontradas para este propósito; recordando rápidamente esa idea, un Sistema Embebido es aquel conjunto de hardware y software que desempeña alguna actividad de forma eficiente, diseñado para consumir un mínimo de energía y con la mínima cantidad de componentes, [9].

La tarjeta de desarrollo Spartan 3E starter kit y el desarrollo propio del sistema de reconfiguración remota (SMIN_V1) dejaron cimientos fuertes en la base del desarrollo del nuevo sistema SMIN_V2, para enriquecer y robustecer el desempeño de la plataforma principalmente en la capacidad de integración de arquitecturas de computo en el dispositivo FPGA, integrando nuevos componentes que permiten la evolución del sistema; haciendo reuso de algunos bloques funcionales y otros modificados para ser adaptados al nuevo diseño.

Ya que un sistema embebido comprende hardware y software, el siguiente capítulo será dedicado a describir el software que complementa y cierra el desarrollo de la plataforma FPGA.

4

Descripción del software de operaciones de la plataforma FPGA

Una de las características de los sistemas embebidos es la capacidad de diseñar e integrar elementos de hardware y software para crear una aplicación a la medida, [10].

La nueva plataforma FPGA SMIN_V2 pertenece a dicha categoría al combinar hardware y software específico durante el proceso de desarrollo del sistema. Parte del software se encuentra contenido dentro de dispositivos electrónicos y el resto es software de apoyo para interactuar con el subsistema mismo.

El contenido de este capítulo se enfoca principalmente en describir las herramientas en software integradas como parte del diseño final de la tarjeta SMIN_V2 basado en un FPGA.

4.1 Estación Terrena

La Estación Terrena es una interfaz gráfica desarrollada en un lenguaje de alto nivel que permite al usuario realizar fácilmente la transmisión del *bitstream* al FPGA.

Al igual que la plataforma FPGA SMIN_V2 está conformada por varios bloques, la GUI se puede dividir en tres grupos o bloques principales:

1. El primer paso es definir la arquitectura (archivo xsvf).
2. Después convertirlo a formato IntelHEX.
3. Finalmente transmitirlo con ayuda de la GUI.

4.1.1 Herramienta IMPACT de la suite ISE de XILINX

La herramienta iMPACT está contemplada dentro de la parte de la Estación Terrena para convertir el archivo de configuración .bit a .xsvf, paso inicial en la preparación del *bitstream* y enviarlo a la tarjeta SMIN_V2.

Esta herramienta es proporcionada por la suite ISE de XILINX, principalmente permite la configuración de dispositivos y generación de archivos de configuración. [19]

La configuración de dispositivos da la opción de configurar directamente FPGAs, CPLDs y memorias PROMS a través de interfaces alámbricas XILINX como MutiPRO Desktop Tool, cable *Parallel III*, o Platform Cable USB, en varios modos, [19]:

- Modo Boundary-Scan. FPGAs, CPLDs y PROMs de XILINX.
- Modo Slave Serial o MAP. Solo los FPGAs pueden ser configurados.
- Modo Configuración Desktop. CPLDs y PROMs se pueden programar.
- Modo Configuración Directa SPI. Dispositivos seriales FLASH con protocolo SPI.

En la opción de generación de archivos se puede crear archivos de configuración de tipo SVF, XSVF, entre otros.

Además iMPACT permite:

- Verificar los datos de configuración de la arquitectura.
- Depurar problemas de configuración.
- Ejecutar archivos .svf y .xsvf.

4.1.2 Programa IntelHEX

Un archivo IntelHEX es un archivo de texto que representa a un archivo binario en código ASCII. Cada línea, llamada registro, en un archivo IntelHEX contiene valores hexadecimales que representan código en lenguaje máquina y/o datos.

Cada registro o línea, está conformado por 5 campos, los cuales se listan enseguida:

1. **Código de inicio.** Símbolo ":" que indica el inicio del registro, que en hexadecimal es 03AH.
2. **Longitud del registro.** Primeros dos dígitos hexadecimales con la cantidad de bytes en el campo "Datos".
3. **Dirección.** Consta de cuatro dígitos hexadecimales en *big endian*, con la dirección de inicio de datos.
4. **Tipo de registro.** Dos dígitos hexadecimales que definen el tipo dato a almacenar. El tipo de registro es usado para interpretar el resto de la información del registro. Hay seis tipos de registro:

    - **'00'. Datos.** Indica que el registro contiene la dirección de 16 bits y los datos correspondientes.
    - **'01'. Fin de archivo.** Con este no existen datos y debe de encontrarse como último registro del archivo.
    - **'02'. Dirección Extendida de Segmento.** Indica que el valor de este registro se agrega a las direcciones de los registros de datos subsecuentes para obtener la dirección final donde se alojará la información. Este tipo de registro se utiliza para acceder a direcciones con más de 16 bits. Su longitud siempre es 02 y el campo dirección 0000.
    - **'03'. Dirección de Inicio de Segmento.** Especifica los valores iníciales de los registros, para procesadores 80x86. El campo de dirección es 0000, longitud 04 y los datos contienen dos bytes para el segmento de código y otros dos para el instrucción pointer.
    - **'04'. Dirección Lineal Extendida.** Con este registro permite obtener acceso a direcciones de memoria hasta de 32 bits; el valor del campo dato indica los bits del 16 al 31 de la dirección de memoria para los registros de datos subsecuentes. La dirección de memoria final se obtiene agregando dicho valor a la dirección de registros de datos. El campo de dirección siempre es 0000 y de longitud 02.
    - **'05'. Inicio de Dirección Lineal.** Contiene 4 bytes que se cargan en el registro EIP de los procesadores 80386 y superiores. Su campo de dirección vale 0000 y el de longitud 04.

5. **Datos.** Dígitos hexadecimales que contienen la información.
6. **Checksum.** Dos dígitos hexadecimales con complemento a dos de la suma de todos los elementos a excepción del código de inicio (":").

Los únicos registros utilizados son 00, 01 y 04.

Para la generación de archivos en formato IntelHEX se siguió el diagrama de flujo de la FIGURA 4.1.

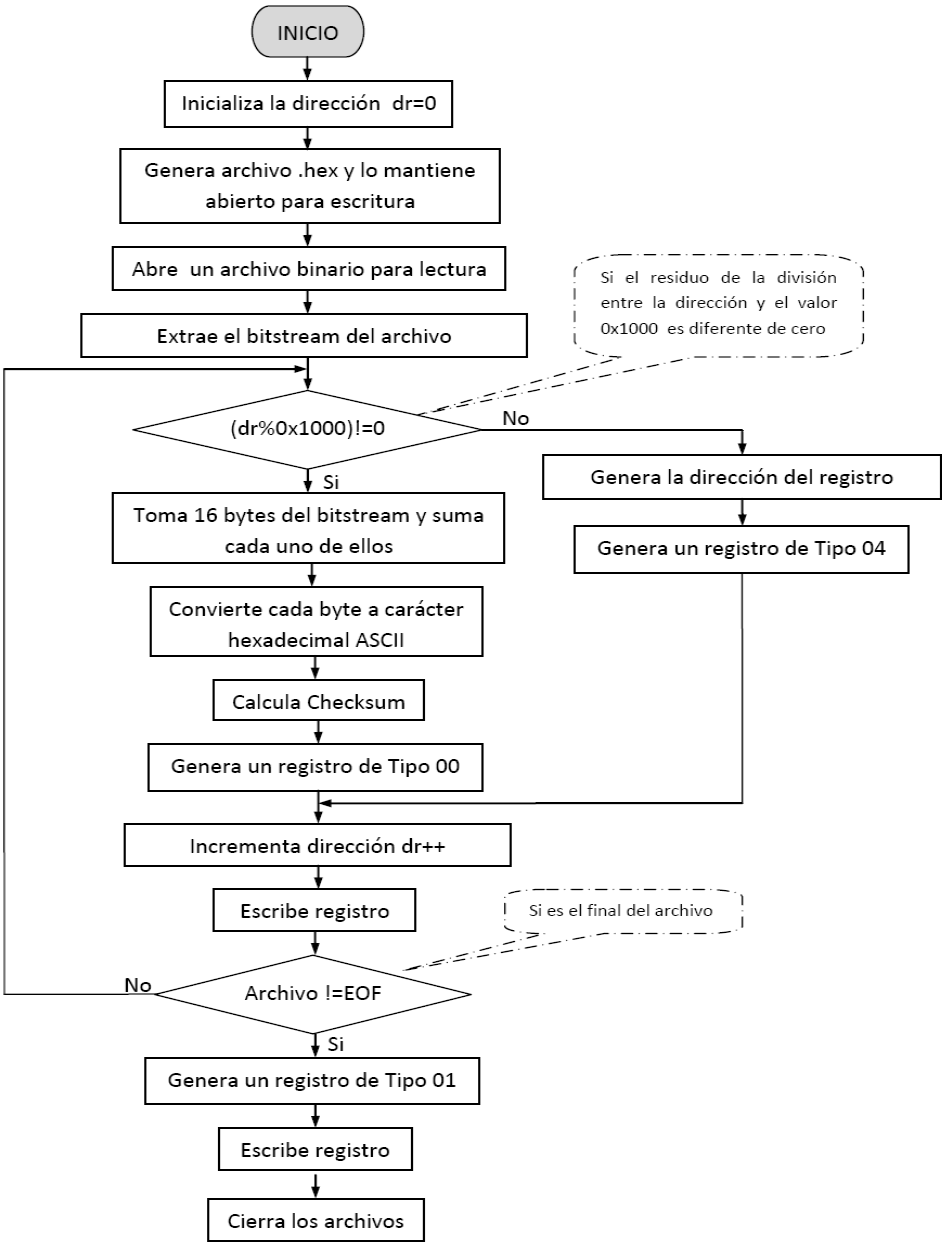


Figura 4.1. Diagrama de flujo de la herramienta IntelHEX.

4.1.3 GUI – Interfaz Gráfica de Usuario

Diversos productos electrónicos en el mercado proveen consigo un programa que permite al usuario interactuar con el sistema. En términos técnicos a ese programa se le denomina GUI (Graphical User Interface). [20]

Hoy en día el desempeño que tenga la GUI al comunicarse con su sistema determina qué tan fácil es de usar esa tecnología y qué tan bien funciona. [20]

Para el SMIN_V2, la interface no solo permite al usuario interactuar con el sistema, sino trabajar con él remotamente. La interfaz de usuario fue desarrollada bajo el ambiente de programación Microsoft Visual Studio y con el lenguaje C#.

La GUI tiene una estructura modular, distribuida en caja de selección, botones de comando y ventanas de despliegue de datos, haciendo una herramienta amigable e intuitiva en todo momento, desde seleccionar el puerto serial de la computadora, parámetros de comunicación serial, selección del archivo de reconfiguración que será almacenado en la memoria FLASH del Sistema Mínimo V2, para descargarlo posteriormente en el FPGA. Algunas de las tareas que se pueden realizar desde la GUI son las siguientes:

Verificar comunicación	• Envía el bit correspondiente para verificar comunicación. • Espera bit de respuesta.
Escritura a la memoria a bordo de la tarjeta	• Envía bit de escritura. • Extrae el archivo intelHex de alguna carpeta dentro de la PC. • Transmite el archivo por puerto serial. • Si existe un error en la comunicación, recibe bit de error y detiene la transmisión. Indica que se presentó un error y espera nuevas instrucciones. • Espera bit que indica que se terminó la transmisión.
Lectura de la memoria a bordo de la tarjeta	• Envía bit de lectura. • Recibe el archivo extraído de la memoria, si detecta algún error detiene la transmisión e indica que se presentó un error. • Imprime en pantalla el archivo recibido. • Termina la transmisión al recibir el bit que indica que el *bitstream* de la memoria se extrajo completamente.

Borrado de la memoria a bordo de la tarjeta	• Envía bit de borrado. • Espera el bit que indica que se completó la transmisión.
Reconfiguración del FPGA	• Envía el bit de reconfiguración y espera a que finalice la misma. • Si existe algún error en la reconfiguración. Recibe el bit de error y espera nuevas instrucciones.

La FIGURA 4.2 muestra el proceso tradicional del desarrollo de un producto, donde la GUI se deja hasta el final.

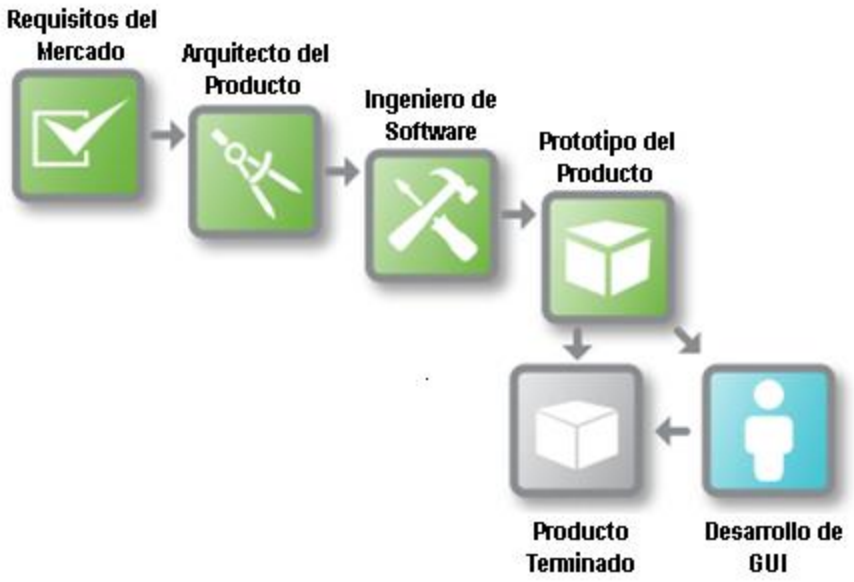


Figura 4.2. En el proceso tradicional de desarrollo de un producto se deja la GUI hasta el final. Las interfaces cumplen con especificaciones en ingeniería pero el factor "Wow!!" (factor explicado en el artículo [20]) divide buenos productos de los geniales.

4.2 Software de reconfiguración diseñado para la PC

El software de reconfiguración está diseñado para configurar al FPGA utilizando como interfaz al puerto paralelo de la PC, usando el protocolo de transferencia de datos JTAG. El proceso de configuración se lleva a cabo extrayendo de alguna carpeta en la PC el *bitstream* de reconfiguración en formato xsvf, y byte a byte se ejecutan los comandos necesarios para realizar la reconfiguración del FPGA. El programa original obtenido de Xilinx, se encuentra en lenguaje de programación C++, existe una versión desarrollada en Visual C++, esta se utiliza previamente para validar la integración del FPGA dentro de un PCB. De igual forma puede ser utilizada en este trabajo como herramienta de soporte para detectar fallas y garantizar la correcta operación del sistema propuesto.

La arquitectura del software contiene los elementos obligatorios definidos en el estándar IEEE 1149.1 para el protocolo de configuración JTAG. Estos elementos sólo comprenden al TAP, al controlador del TAP y el registro que contiene las instrucciones .xsvf.

Está estructurado en un programa principal llamado “micro.c” y varios subprogramas auxiliares llamados “ports.c” y “lenval.c”, que hacen uso de algunas bibliotecas propietarias de Xilinx como “micro.h”, “ports.h” y “lenval.h”, los cuales se describen en los siguientes párrafos:

- **micro.c:** Contiene las funciones para interpretar los comandos del archivo xsvf, al igual que para procesar los datos del mismo. Este programa llama a los subprogramas “ports.c” y “lenval.c”.
- **micro.h:** Contiene las funciones prototipo para la interfaz primaria que reproduce el archivo xsvf.
- **ports.c:** Contiene las rutinas que generan los valores de salida hacia los puertos JTAG del FPGA, al igual que las rutinas para leer el valor de salida TDO, y para leer un byte del archivo xsvf almacenado en el disco duro de la PC.
- **ports.h:** Contienen las declaraciones externas para proporcionar estímulos en los puertos JTAG.
- **lenval.c:** Contiene las rutinas para utilizar la estructura de datos lenVal definida en “micro.c” (esta estructura es un tipo de byte de orientación utilizado para almacenar un valor binario de longitud arbitraria).
- **lenval.h:** Contiene la descripción de la estructura de datos LenVal.

Estos programas también utilizan bibliotecas generales tales como: “conio.h”, “stdio.h”, “stdlib.h”, “string.h”, “time.h”, “io.h” y “dos.h”, las cuales habilitan el manejo de funciones estándar de entrada/salida, manejo de cadenas de caracteres alfanuméricos, manejo de puertos de la PC, etc.

En la FIGURA 4.3 se muestra el flujo del proceso de reconfiguración que realiza el software, donde:

El proceso “A” realiza los siguientes pasos en el TAP:

- Select DR
- Select IR
- Capture IR
- Exit1 IR
- Update IR

Mientras que el proceso “B” realiza los siguientes pasos:

- Select DR
- Capture DR
- Exit1 DR
- Update DR

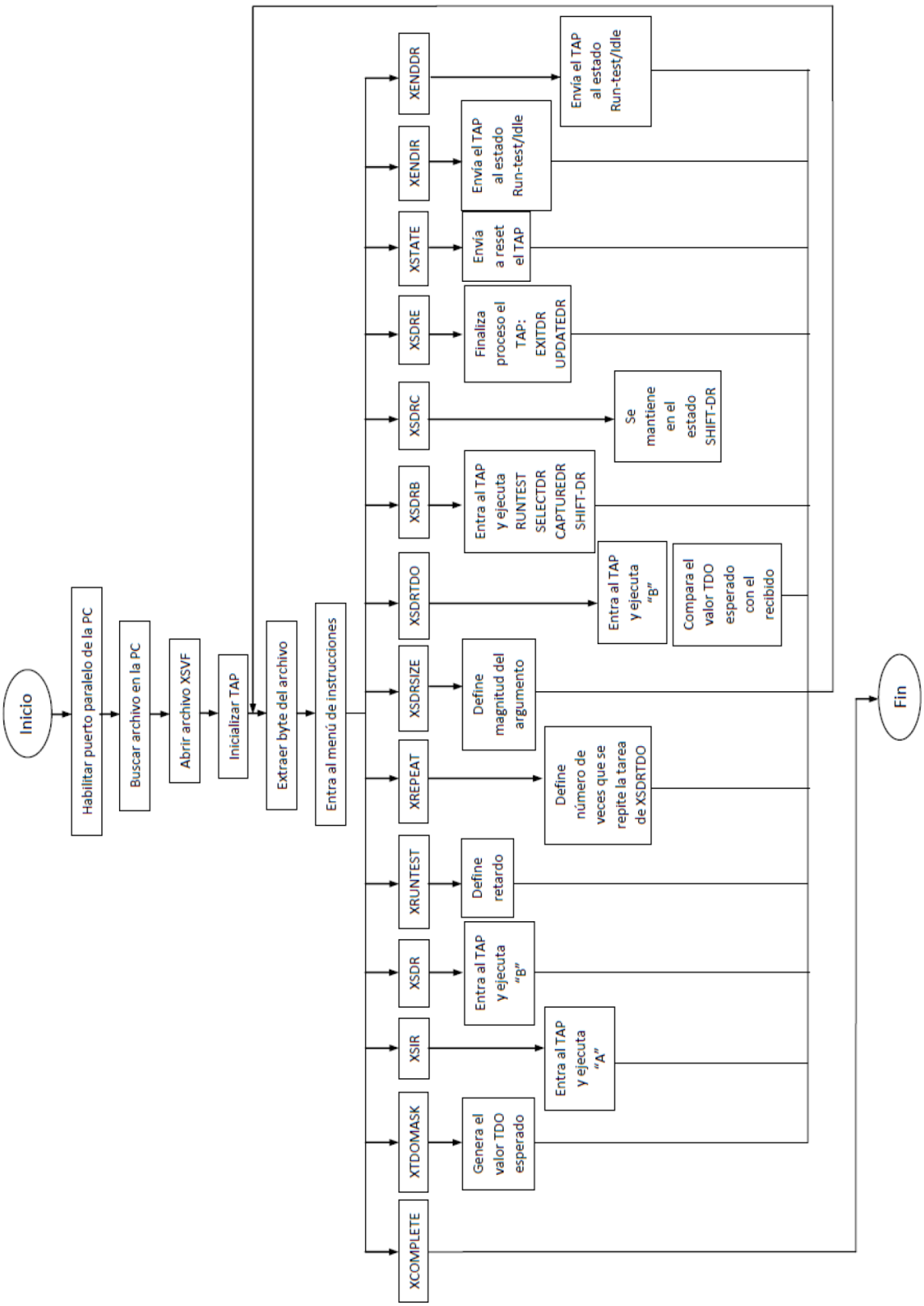


Figura 4.3. Diagrama de flujo del proceso de reconfiguración de FPGA utilizando la PC.

4.3 Sistema de control de apuntamiento

4.3.1 MPLAB IDE

MPLAB IDE es un ambiente de desarrollo destinado a crear aplicaciones para los microcontroladores y controladores digitales de señales de la marca Microchip. Es llamado un Ambiente de Desarrollo Integrado (IDE, del inglés *Integrated Development Enviroment*) porque provee un solo ambiente integrado para desarrollar código de programación para los microcontroladores PIC.

El programa incluye un editor de texto, macro-ensamblador, compilador ANSI C, y simulador para trabajar con cualquier microcontrolador PIC. El simulador puede operarse tanto en programas desarrollados en lenguaje ensamblador o ANSI C, [21].

MPLAB IDE permite:

- Crear y editar código fuente. Gracias a su editor de texto es posible crear y editar código.
- Manejador de proyectos. Agrupa y organiza archivos de código fuente, librerías, cabeceras, *linkers*, etc. y así compilar, descargar y trabajar con archivos específicos.
- Depurar código fuente. Provee características de simulación y emulación para depurar código como: ventanas de visualización de información, uso de "breakpoints", ejecución paso a paso, entre otras.

MPLAB C18

El "*MPLAB C Compiler*" (también conocido como MPLAB C18) es un compilador de lenguaje C con optimización para microcontroladores de la familia PIC18 de Microchip. Es un componente externo compatible con MPLAB IDE, que permite la depuración a nivel de código en conjunto con herramientas de desarrollo de Microchip, [22].

El uso del compilador C18 puede ser completamente gestionado desde la interfaz gráfica del ambiente IDE de MPLAB.

El compilador MPLAB MC18 tiene las siguientes características:

- Compatibilidad con el estándar ANSI 89.
- Integración con el entorno MPLAB IDE para su fácil manejo en la creación y depuración de proyectos de software.
- Generación de módulos de objetos reubicables para un reuso de código mejorado.
- Compatibilidad con módulos de objetos generados por el ensamblador MPASM, permitiendo completa libertad al mezclar código en lenguaje C y ensamblador, en un solo proyecto.
- Acceso transparente a la lectura y escritura de memoria externa.

- Fuerte apoyo para el ensamble en línea cuando se requiere un control total del
- proyecto.
- Generador de eficiencia de código con optimización en niveles múltiples.
- Extensas bibliotecas, incluyendo manipulación de periféricos como generadores de PWM, módulos I_2C, puerto serial, puerto SPI, manipulación de cadenas y bibliotecas matemáticas.
- Control completo del usuario sobre la localización de datos y código.

4.3.2 Propuesta de firmware de nueva plataforma FPGA

La CV y subsistemas de SATEDU se desarrollaron usando el compilador C18 de MPLAB, es por ello que para la nueva versión de la plataforma FPGA (SMIN_V2) se propone migrar al mismo compilador (C18), por ende desarrollar tanto el código asociado al *firmware* de recepción, transmisión y almacenamiento en memoria, como el *frimware* de reconfiguración con base en la lógica mostrada en la FIGURA 4.4 y FIGURA 4.5; esquemas de recepción y evaluación de comandos e, interrupción por recepción en puerto serie, respectivamente.

Lo anterior con la finalidad de promover un estándar durante la etapa de desarrollo, establecer una línea de referencia, robustecer el sistema, y crear la posibilidad de reusar código.

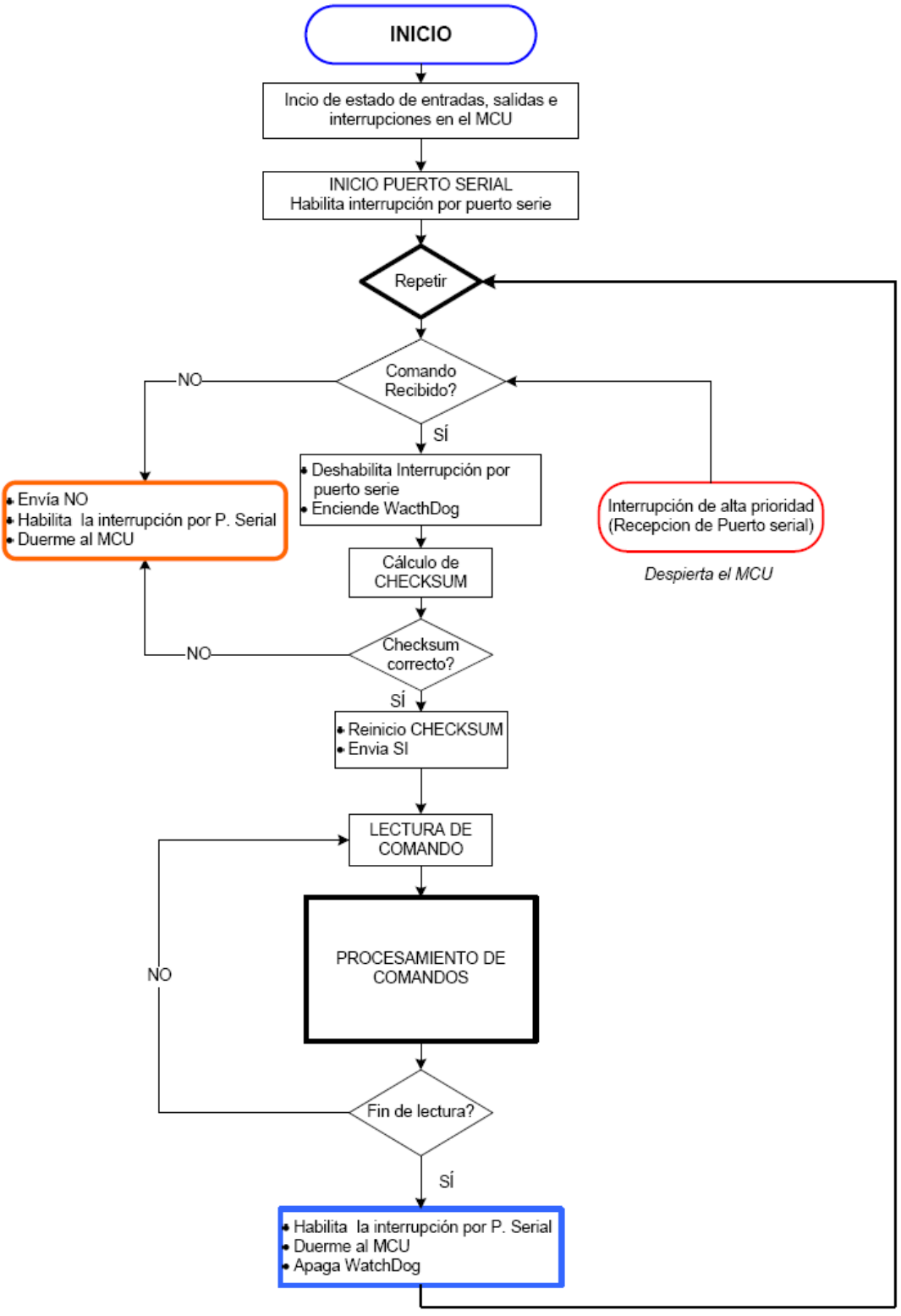


Figura 4.4. Esquema general de recepción y evaluación de comandos.

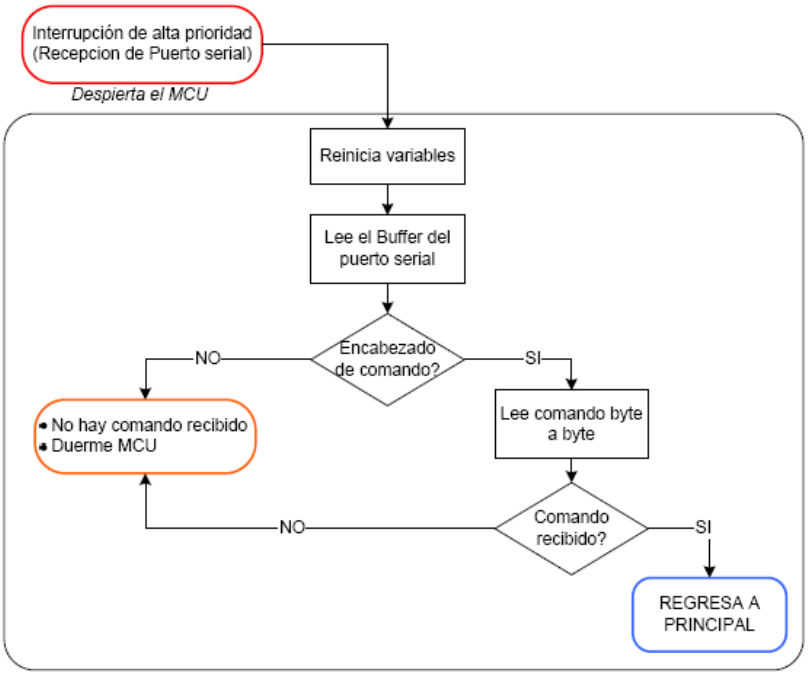


Figura 4.5. Esquema general de interrupción por recepción en el puerto.

Firmware de recepción, transmisión y almacenamiento en memoria

El *firmware* permite que el PIC, residente en la tarjeta de SMIN_V2, sea controlado desde la estación terrena, desde donde el usuario (operador) podrá enviar, por un lado el *bitstream* de reconfiguración para el FPGA y por otro, comandos que permiten la interacción Estación Terrena – SMIN_V2 y ejecutar diversas tareas de control y verificación de comunicación, escritura y lectura de datos de la memoria a bordo. En el caso, por ejemplo de la escritura, el *bitstream* de reconfiguración, una vez que sido transmitido desde la Estación Terrena, en el PIC se convierte a un archivo binario, para posteriormente almacenarlo en una memoria FLASH utilizando el bus de comunicación SPI. Para la lectura, utilizando la opción correcta en la GUI, se enviará el comando para extraer el *bitstream* de la memoria y convertirlo nuevamente a un archivo IntelHEX para después transmitirlo a la Estación Terrena y verificar los datos contenidos en la memoria.

Firmware de reconfiguración ajustado al PIC

El *firmware* de reconfiguración permite configurar al FPGA utilizando un PIC. La lógica de reconfiguración es similar a la lógica que lleva a cabo el software de reconfiguración ejecutado en la PC, a diferencia de que en el PIC se extrae el *bitstream* de reconfiguración de la memoria FLASH de forma serial, para después transmitirla a los puertos de

configuración JTAG del FPGA utilizando terminales de un puerto de propósito general del PIC.

Existe una versión previa del *firmware* desarrollada con el compilador CCS completamente funcional con la versión anterior al SMIN_V2, el desarrollo del *firmware* requirió de mucho tiempo destinado a investigación, análisis y depuración del código. Esta referencia será crucial para adaptarlo a la nueva versión del Sistema Mínimo V2 (SMIN_V2), posiblemente el módulo de mayor importancia por el nivel de complejidad y funcionalidad que tiene. Por ello es necesario analizar su estructura, realizar cambios de forma iterativa y verificar cambio con cambio, con el fin reducir errores que impacten la funcionalidad de reconfiguración.

En la FIGURA 4.6 se muestra el flujo del proceso en bloques del firmware, se observa que es similar al de la PC.

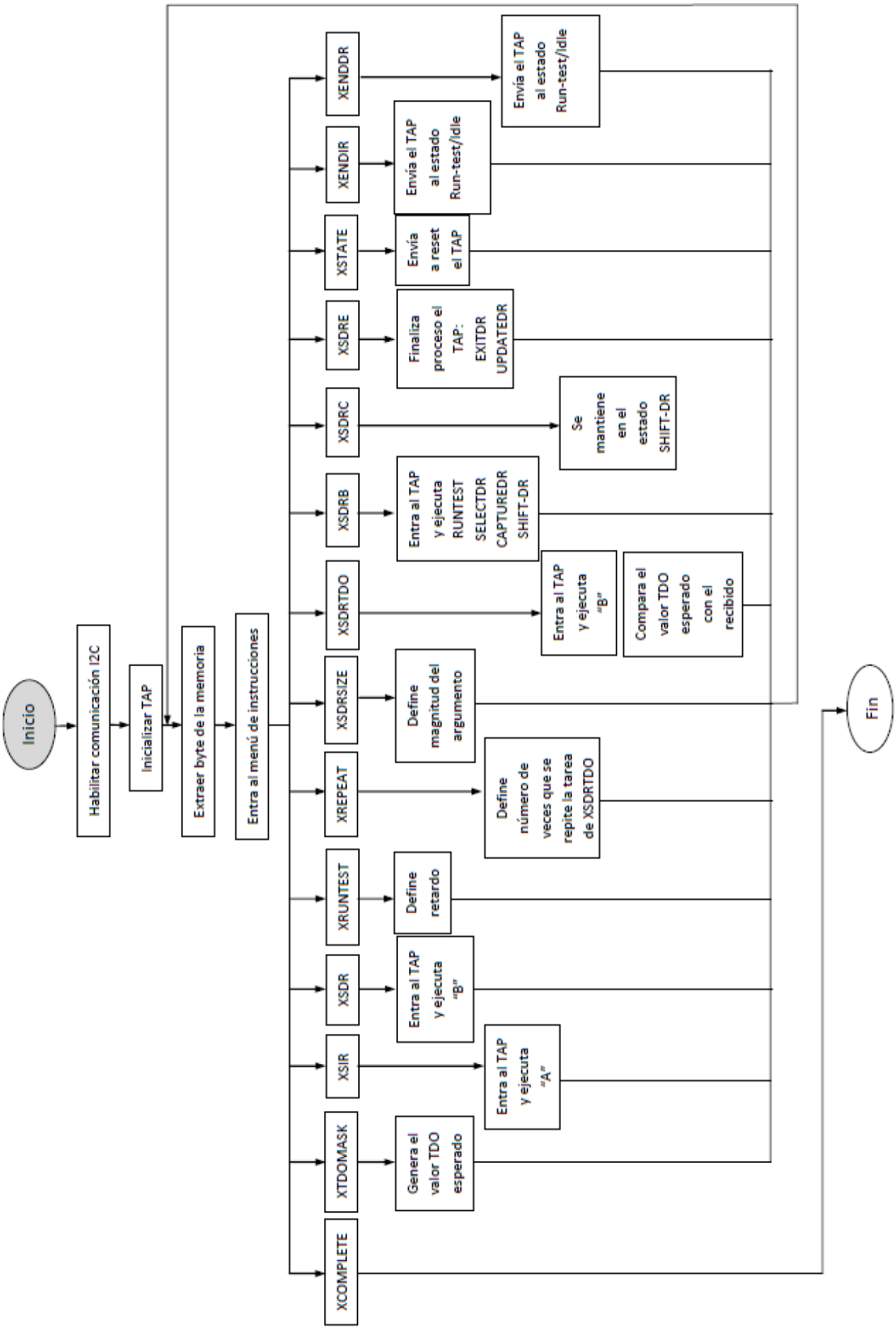


Figura 4.6. Diagrama de flujo del proceso de reconfiguración de FPGA utilizando el PIC.

4.4 Resumen

Después de ver las herramientas en software a utilizar se hace ver que en los sistemas embebidos existe una estrecha dependencia hardware – software para potencializar un producto, en este caso la plataforma FPGA reconfigurable SMIN_V2.

Aunque ambos *firmwares* estarán contenidos dentro del mismo MCU, su implementación se encuentra de forma separada, de esta forma simplifica la validación e integración de ambos módulos. Uno de ellos, "*firmware* de recepción, transmisión y almacenamiento en memoria", cambia radicalmente al proponer ajustarlo al esquema de recepción y evaluación de comandos e, interrupción por recepción en puerto serie. Mientras que el "*firmware* de reconfiguración ajustado al PIC" conserva su lógica y flujo, sin embargo, es un módulo complejo que requiere de gran mesura para adaptarlo al compilador C18.

5

Propuesta de manufactura y ensamble de la plataforma FPGA

Una tarjeta PCB (*Printed Circuit Board*) interconecta componentes electrónicos mediante pistas de material conductor, dichas pistas son el resultado de remover material conductor del resto de la tarjeta.

Las PCBs pueden ser de una cara (pistas en un lado de la tarjeta), dos caras o multicapa, mientras mayor sea el número de capas, mayor es la complejidad del diseño.

En este capítulo se explica el procedimiento de diseño de la PCB, manufactura y ensamble de la tarjeta con los componentes electrónicos que integra la nueva tarjeta SMIN_V2.

5.1 Diseño de PCB de plataforma FPGA

El circuito impreso de la nueva plataforma propuesta tiene una forma cuadrada de 8.9cm de largo por 8.9 cm de ancho, medidas que se ajustan a las dimensiones de SATEDU. Se plantea un diseño de dos capas debido a que el nuevo FPGA empleado tiene 320 terminales distribuidas a lo largo de la superficie de una de sus caras, formando un arreglo similar a una matriz (empaquetado BGA). La primera imagen, FIGURA 5.1 muestra la distribución de terminales de un FPGA de este tipo, la segunda, FIGURA 5.2, es la imagen proporcionada por el fabricante para ubicar sus conectores.

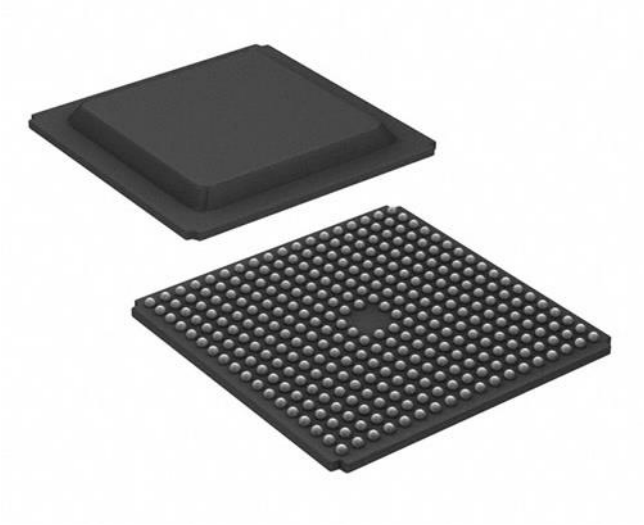


Figura 5.1. Distribución de terminales de FPGA Xilinx-XC3S1600E.

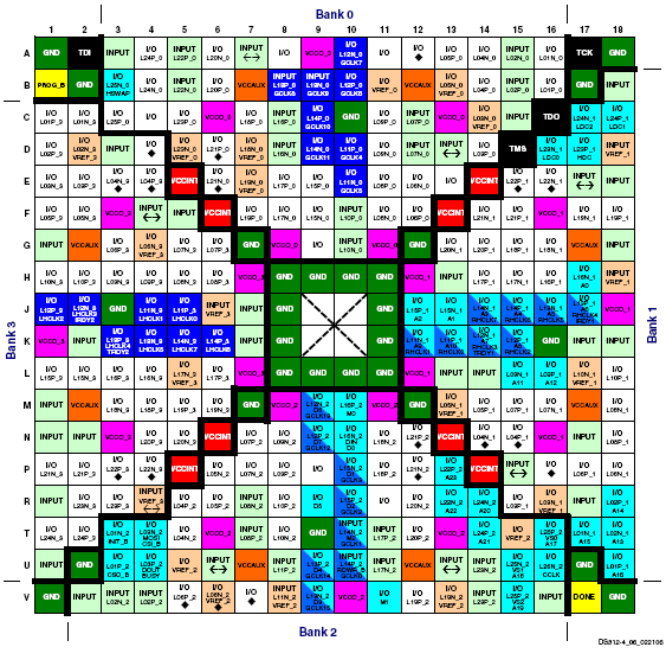


Figura 5.2. Identificación y ubicación de conexiones del FPGA Xilinx-XC3S1600E proporcionado por el fabricante, [14].

El diseño de la tarjeta se realizó con ayuda del software *Altium Designer* 6 de *Protel*, un ambiente de desarrollo electrónico que posee características para visualizar, editar, crear y administrar (entre muchas otras) el trabajo. Una característica destacada del programa es la capacidad de realizar un "auto ruteo" para interconectar de forma automática los distintos elementos electrónicos que integran la tarjeta de circuito impreso. La ventaja que benera, es que en poco tiempo quedan listas las conexiones, la desventaja, es que no toma en cuenta factores propios del sistema para discriminar rutas críticas. Es por ello que con un diseño de dos capas se hace relativamente más sencillo analizar y realizar las rutas de las pistas adecuadamente y evitar re trabajo.

El grosor de las pistas para conectar terminales del FPGA con algún otro componente es de 6mm, el resto tiene un grosor de 10mm. En la FIGURA 5.3, se aprecia la matriz de terminales del FPGA, en rojo las pistas de la cara superior y en azul de la cara inferior, de ser requerido un mayor número de conexiones con el FPGA se evaluaría la necesidad de una tercera capa. Más abajo la FIGURA 5.4 muestra la cara superior, en la FIGURA 5.5 la capa inferior y en la FIGURA 5.6 el diagrama esquemático.

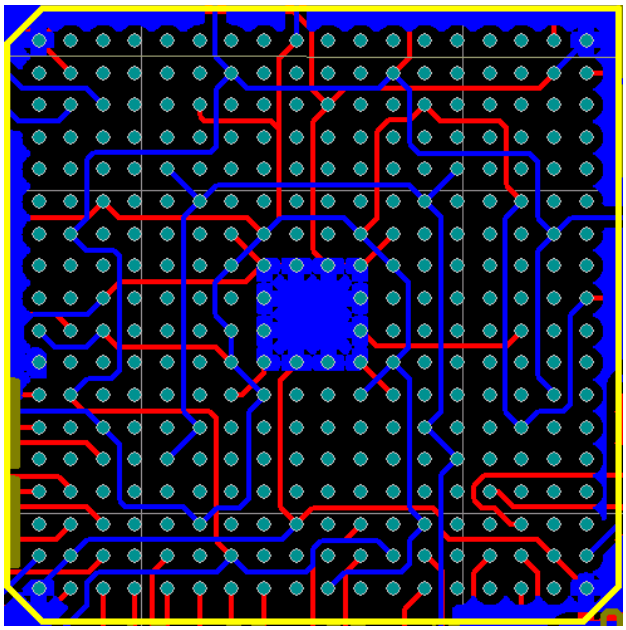


Figura 5.3. Pistas de cara superior (rojo) e inferior (azul) de terminales de FPGA.

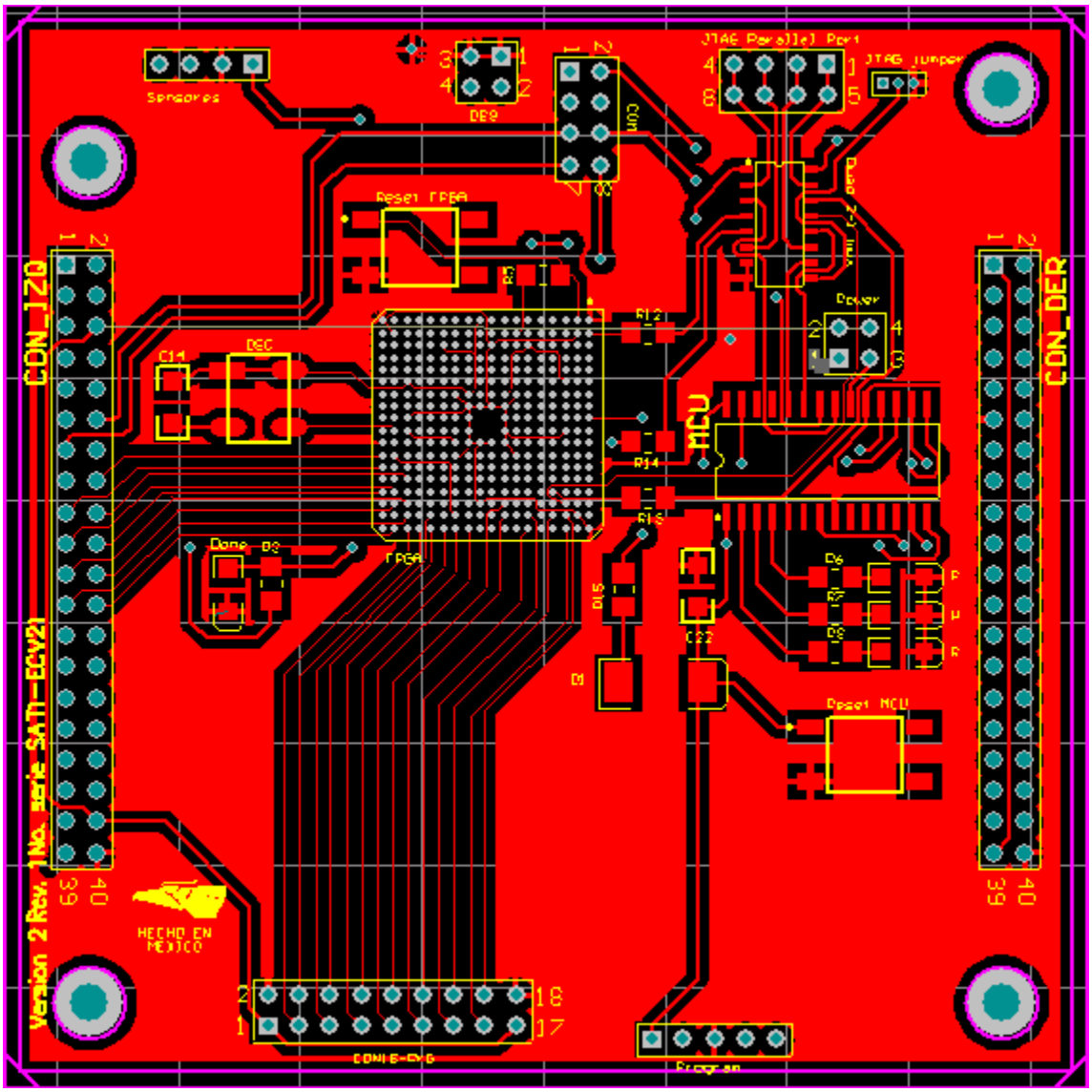


Figura 5.4. Capa superior de la SMIN_V2.

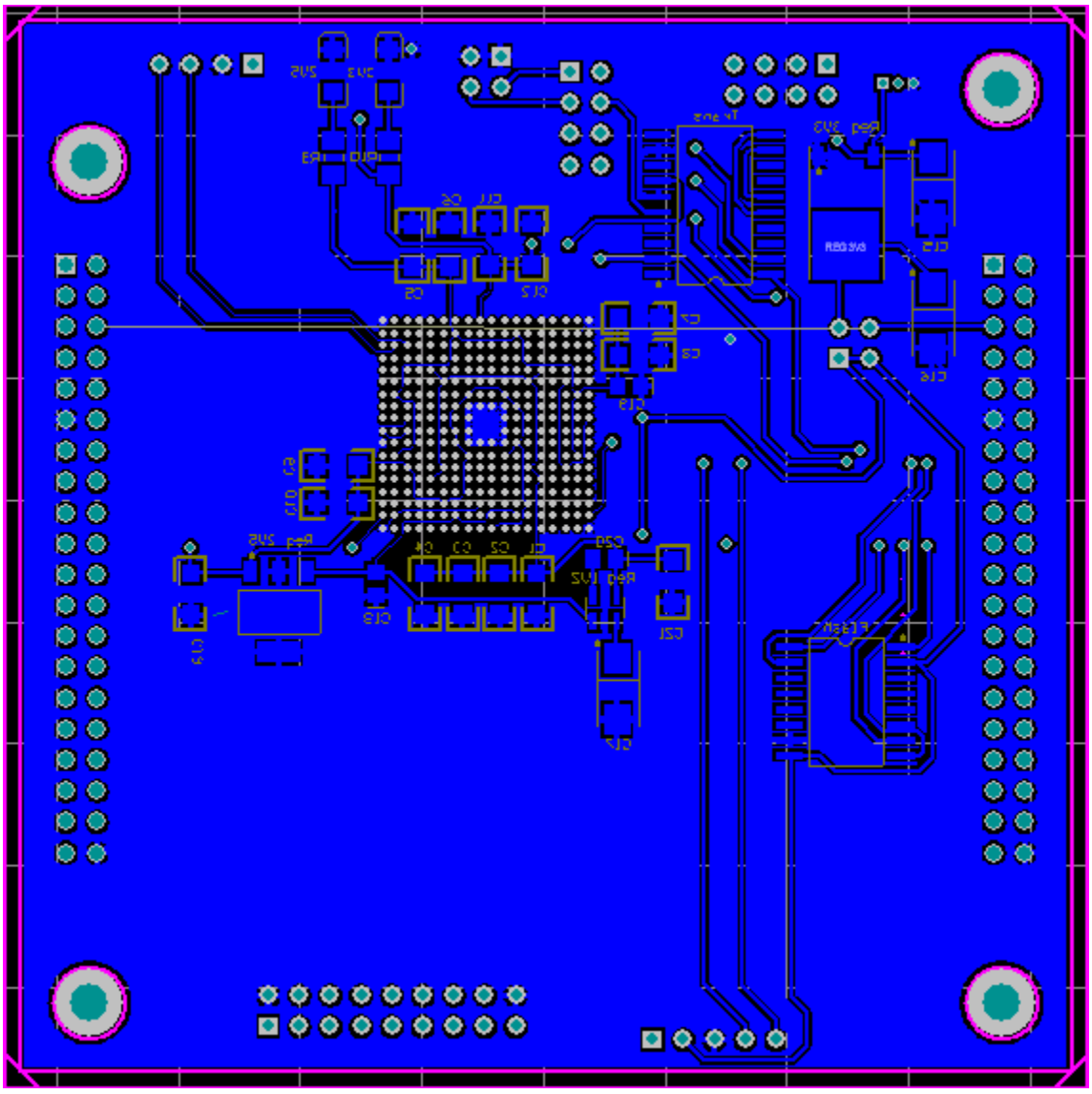


Figura 5.5. Capa inferior de la SMIN_V2.

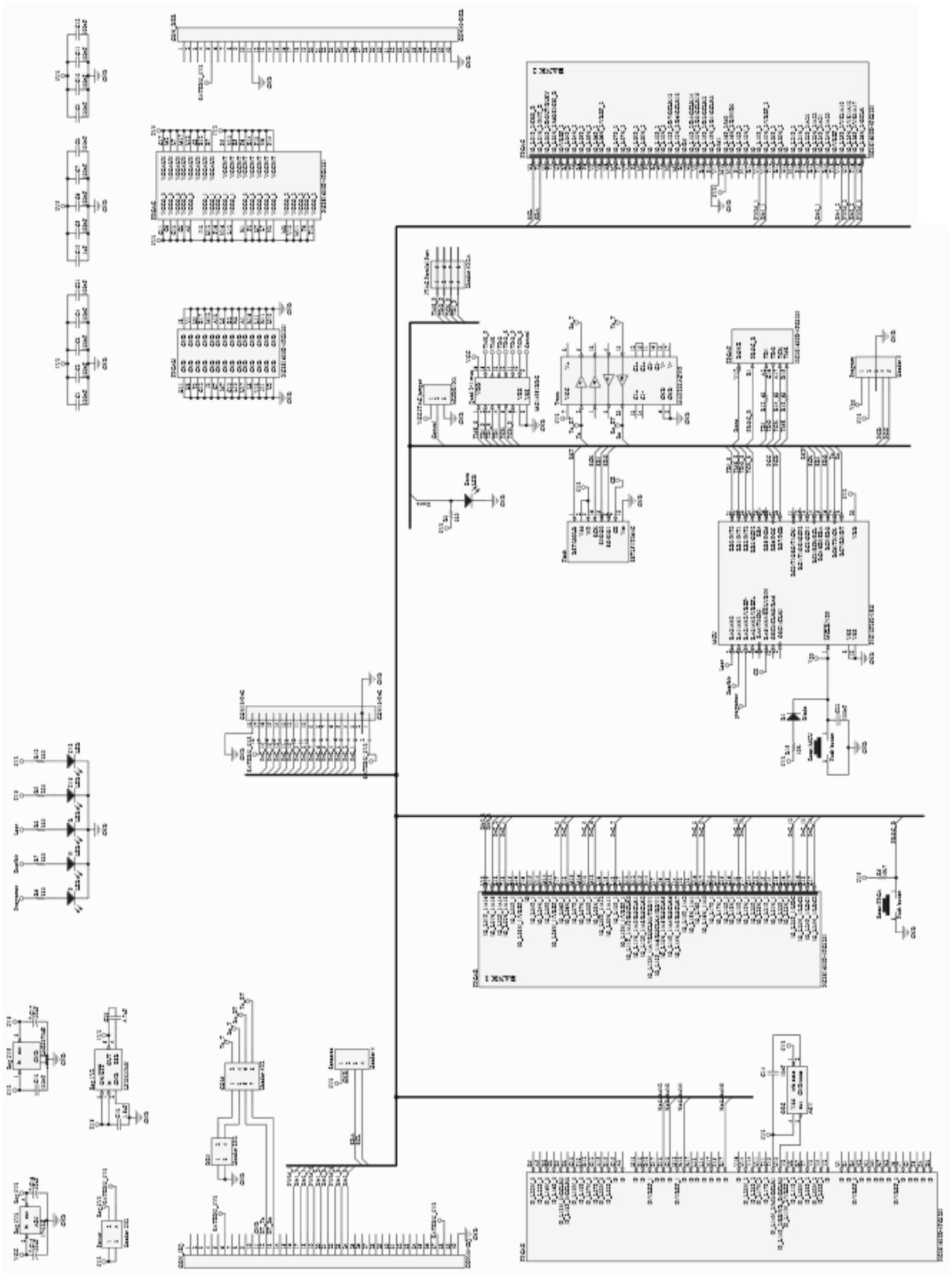


Figura 5.6. Diagrama esquemático del nuevo Sistema Mínimo (SMIN_V2).

5.2 Precauciones de manufactura de PCB

Existen distintos tipos de empaquetados de componentes electrónicos, así mismo, la tecnología empleada en la construcción de tarjetas de circuito impreso. En este caso, se mencionan tres:

- **Through hole**. También "thru-hole", se refiere al montaje de componentes que involucran piezas de metal que provienen del dispositivo, estas "patitas" o terminales son insertadas a través de orificios perforados en la PCB y posteriormente soldados a las pistas del circuito impreso.

- **SMT** (*Surface Mount Technology*). Tecnología empleada en la elaboración de tarjetas electrónicas donde el componente de montaje superficial (*Surface Mount Device*) se coloca sobre la superficie del circuito impreso. En la industria ha reemplazado en gran parte al método de construcción *Through hole*. Generalmente dispositivos compatibles con tecnología de montaje superficial son de menor tamaño que su contraparte *Through hole*.

- **BGA** (*Ball Grid Array*). Es un tipo de empaquetado similar al de los dispositivos de montaje superficial, la principal diferencia radica en que toda la superficie inferior puede ser usada en lugar de solo la periferia, y provee de mayor número de interconexiones. El uso de esta tecnología requiere un control preciso de soldado, generalmente realizado en procesos automatizados.

El FPGA Xilinx-XC3S1600E es de empaquetado BGA, en la hoja de especificaciones se encuentra información respecto a las ventajas al emplear dispositivos de este tipo sobre los de empaquetado QFP (*Quad Flat Pack*). La
TABLA 5.1 , tomada del datasheet de la tarjeta Spartan 3E, muestra un resumen de lo citado anteriormente.

Characteristic	Quead Flat Pack (QFP)	Ball Grid Array (BGA)
Maximum User I/O	158	376
Packing Density (Logic/Area)	Good	Better
Signal Integrity	Fair	Better
Simultaneous Switching Output (SSO) Support	Fair	Better
Thermal Dissipation	Gair	Better
Minimum Printed Circuit Board (PCB) Layers	4	4-6
Hand Assembly/Rework	Possible	Difficult

Tabla 5.1. Información tomada del datasheet de Xilinx de la plataforma de desarrollo Spartan 3E, [18].

Sin embargo, el uso de esta tecnología también trae desventajas, por ejemplo:

- Difícil montaje/soldado a mano.
- Difícil inspección.
- Uso de componente BGA impráctico durante desarrollo.
- Alto costo de equipo.

Con la versión previa del sistema (SMIN_V1) quedó la experiencia de presentarse cierta dificultad de integrar en el laboratorio del Instituto de Ingeniería de la UNAM un FPGA con empaquetado QFP, siendo de menor cantidad de terminales distribuidas en la periferia del componente electrónico al FPGA propuesto para la nueva versión (SMIN_V2). Por ello, se decidió tomar la iniciativa de consultar a personal con experiencia y equipo especializado en fabricación de PCBs y ensamble electrónico, particularmente el Ing. Aguiñaga, director de la compañía TEEIT que contestó amablemente las tres preguntas (respuestas en "*cursiva"*):

1. ¿Qué técnica(s) de soldado se usa(n) para dispositivos con matriz de terminales la parte inferior y en su contorno? Ventajas respecto hacerlo a mano.
   "*Supongo se refiere a componentes tipo BGA y QFP (chips) la soldadura a mano, requiere de mayor tiempo, tanto para soldar, como exposición del componente al calor directo, si se soldan en automático, el procedimiento es: Deposito de soldadura en PCB, después el montaje con las maquinas, el cual es más preciso (se evitan cortos entre pines y/o bolitas en caso del BGA), después viene el reflujo (para la soldadura), el cual es un horno programado en Tiempo y temperatura para que el circuito pase por las zonas a diferentes temperaturas del horno, la soldadura tiene uniformidad en su reflujo y se protege mejor al componente. Nosotros podemos realizar las técnicas Manual y en Automático, siempre es mejor el segundo."*

2. ¿Antes de aplicar soldadura se verifica coincidencia de PCB y terminales para evitar rotación de dispositivo? ¿Hay alguna técnica?
   "Si es necesario la revisión de componentes y PCB para determinar la correspondencia en tamaño (SMT) y distancia de perforaciones en caso de ser componentes de PTH, comprobar que se indique la polaridad para los componentes que lo requieren. La técnica que se utiliza es visual."

3. En caso de haber algún error en el proceso de soldado y continuidad, ¿qué técnica se realiza para corregirlo sin dañar el dispositivo en cuestión y garantizar su operación?
   "*Después del proceso de soldadura, llámese por ola (maquina) o reflujo (horno SMT) se tienen estaciones de inspección donde se revisan los circuitos en el aspecto de soldadura, eliminando cortos, soldaduras excesivas o "resoldando" donde hay*

faltantes, esta actividad no afecta al circuito, esta actividad, depende mucho de la habilidad que se tiene para realizar la operación de soldadura."

Sin mencionar que la calidad de pureza en el aire, lente de inspección, equipo adecuado, proceso bajo normas de calidad, etc. pero sobre todo, personal con experiencia en soldado de dispositivos con empaquetado BGA, impactan directamente al correcto funcionamiento de un dispositivo de alto costo monetario.

Dada la naturaleza del componente, conviene extremar precauciones durante todo el proceso de manufactura y evitar que esta etapa pueda causar problemas posteriores.

5.3 Resumen

Asegurar el ensamble de la nueva plataforma FPGA con cantidad mínima de errores evita en un futuro falsos diagnósticos en la depuración de hardware y software.

Ahora bien, se conocen las generalidades de SATEDU y FPGAs, núcleo central de procesamiento del SMIN_V2, la descripción de hardware y software que integra el sistema, incluso, la tarjeta de circuito impreso que contendrá la plataforma.

El siguiente paso a seguir es la validación de la nueva tarjeta del subsistema de control de orientación para el satélite educativo SATEDU, con los recursos necesarios para evaluar el funcionamiento de diferentes algoritmos de control de apuntamiento satelital. En el siguiente capítulo se aborda el tema sobre el esquema propuesto que contendrá el hardware y software necesario para operar dentro y fuera de SATEDU.

6

Propuesta de pruebas de integración y operación parcial de la plataforma FPGA

En la vida cotidiana escuchamos hablar de sistemas políticos, sociales, del sistema solar en geografía, el sistema nervioso en medicina, los sistemas operativos en ingeniería, sistemas energéticos, sistemas electrónicos, etc. Recordando la definición de "sistema", un sistema consiste en una estructura organizada de elementos (componentes, partes, bloques) con tareas particulares. Estos elementos influyen en el sistema para alcanzar un objetivo en particular, [26]. Un sistema complejo se puede dividir en subsistemas más sencillos que unidos realizan tareas del sistema original.

Para la integración final de la nueva tarjeta del Sistema Mínimo V2 se requirió la partición o subdivisión de la misma en bloques con el fin de validar elementos modulares del sistema.

En este capítulo se describirán las pruebas claves de validación para la integración del nuevo bloque de reconfiguración del FPGA.

Los Sistemas Embebidos son una "abstracción" de un producto o dispositivo. A un nivel de arquitectura, en un Sistema Embebido los componentes que lo conforman son en realidad una combinación de elementos de hardware y software interdependientes. Dicho de otra manera, está conformado por bloques con hardware y/o software. [8]

La validación de integración consiste propiamente en corroborar el óptimo funcionamiento de los módulos diseñados en hardware con el software asociado a ellos.

Se proponen realizar pruebas de forma sucesiva, es decir, iniciar por el bloque más sencillo y avanzar hacia los bloques de mayor complejidad, para finalizar con la integración del sistema completo. La finalidad de trabajar bajo este esquema es integrar la verificación del bloque sencillo dentro del bloque siguiente.

6.1 Verificación de PC - MCU

El objetivo de validar la comunicación PC - MCU se debe a que en la nueva plataforma SMIN_V2 se usa microcontrolador y memoria diferente al de la primera versión, por ello, es importante verificar que el código asociado a la comunicación, escritura y lectura funcionen correctamente previo a la carga del *bitstream*.

Esta prueba consiste básicamente en ejecutar el software contenido en el MCU, el cual, almacena directamente en la memoria FLASH un par de valores o datos definidos en el proceso de escritura y posteriormente lee dichas localidades de memoria para verificar la información contenida. El uso de cualquier aplicación que permita visualizar la comunicación serial (RS-232), por ejemplo el programa "Serial Port Monitor", permite corroborar visualmente el proceso de escritura y lectura de la memoria FLASH. De forma gráfica la FIGURA 6.1 muestra lo citado anteriormente.

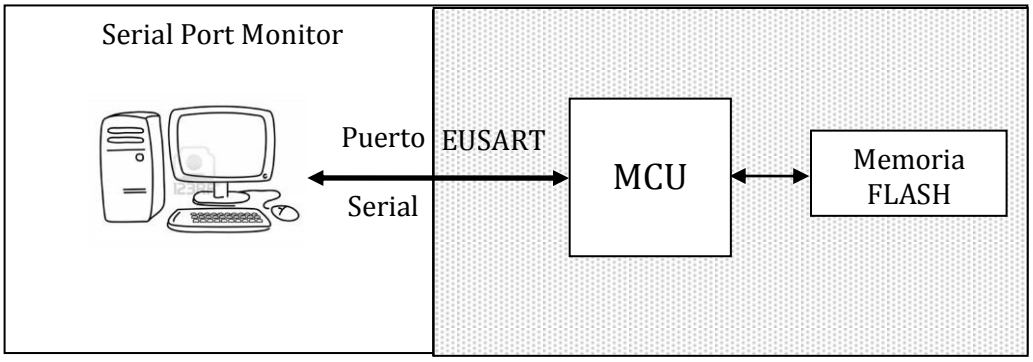


Figura 6.1. Diagrama de verificación de comunicación MCU memoria FLASH.

En la página de Microchip se encuentra una nota de aplicación con ejemplos para implementar el software del *driver* de la memoria FLASH SST25VF064C que en conjunto con su hoja de especificaciones permite entender el uso del dispositivo, [23].

6.1.1 Hardware a utilizar

A continuación se lista los componentes principales para llevar a cabo esta prueba.

- Microcontrolador 18LF2520.
- Modems RF o cable de comunicación serial.
- Memoria FLASH SST25VF064C.
- Transceptor MAX232 y conector DB9.
- Tarjeta prototipo de prueba.

En la tarjeta se conectan los elementos mencionados previamente, acompañados de dispositivos de soporte (resistencias, capacitores, etc). En la figura se muestra el módulo integrado.

6.1.2 Software a utilizar

Ahora se mencionan los programas empleados durante la ejecución de la validación de la escritura y lectura de la memoria FLASH.

- Serial Port Monitor.
- Rutina sencilla de recepción, transmisión y almacenamiento en memoria.

6.1.3 Desarrollo

No es necesario que el MCU sea programado con el *firmware* de recepción, transmisión y almacenamiento en memoria, basta con código que permita abrir el puerto de comunicación serial e implemente comandos de escritura y lectura de la memoria. En este caso en particular para validar la comunicación PC – MCU se proponen los siguientes valores para los parámetros:

- *Baudrate*: 9600 kbps.
- Bit de datos: 8.
- Bit de paridad: Sin paridad.
- Bit de parada: 1
- Control de flujo: Ninguno.

Los valores propuestos pueden cambiar siempre y cuando coincidan tanto en el software para el MCU como en la herramienta “Serial Port Monitor”.

Una vez programado, se conecta el cable serial, uno en la terminal DB9 de la tarjeta de prueba y el otro en el puerto serial de la PC (RS232 o USB por medio de un cable convertidor), se ejecuta y configura el programa “Serial Port Monitor”.

La configuración del programa "Serial Port Monitor" para comunicación serial es bastante simple, se ejecuta la aplicación (por ejemplo: C:\Archivos de programa\Eltima Software\Serial Port Monitor\SerialMonitor.exe), se crea una nueva sesión, luego se elige el puerto de comunicación COM, habilita la casilla "Dump View", "Terminal view", "Start monitoring now" y dar click sobre el botón "Start monitoring". La FIGURA 6.2 muestra los pasos iniciales de la configuración.

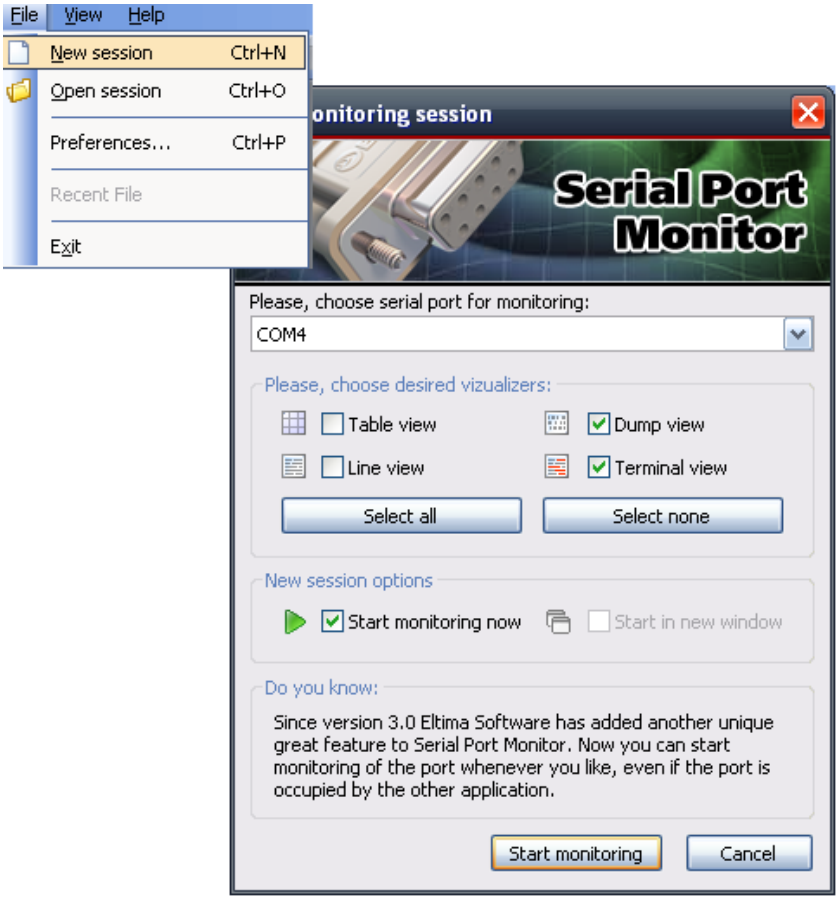


Figura 6.2. Pasos iniciales de configuración de "Serial Port Monitor".

Ahora, se selecciona la velocidad de comunicación, en este caso 9600 bits por segundo (*Baudrate*: 9600), con bit de datos en 8 (*Databits*: 8), con paridad ninguno (*parity*: No parity), bits de parada en uno (*Stopbits*: 1 stop bit) y control de flujo en Hardware (*Flow control*: *None*). La FIGURA 6.3 contiene la continuación de pasos de la configuración.

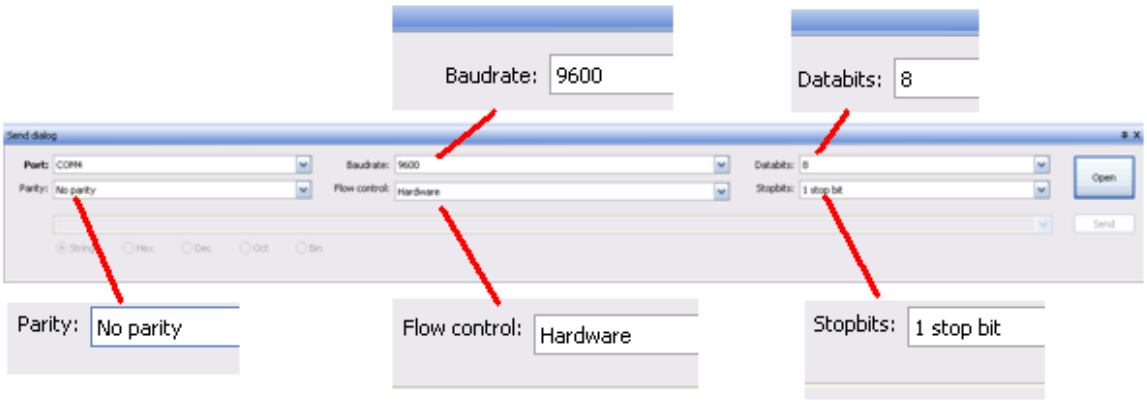


Figura 6.3. Continuación de configuración de "Serial Port Monitor".

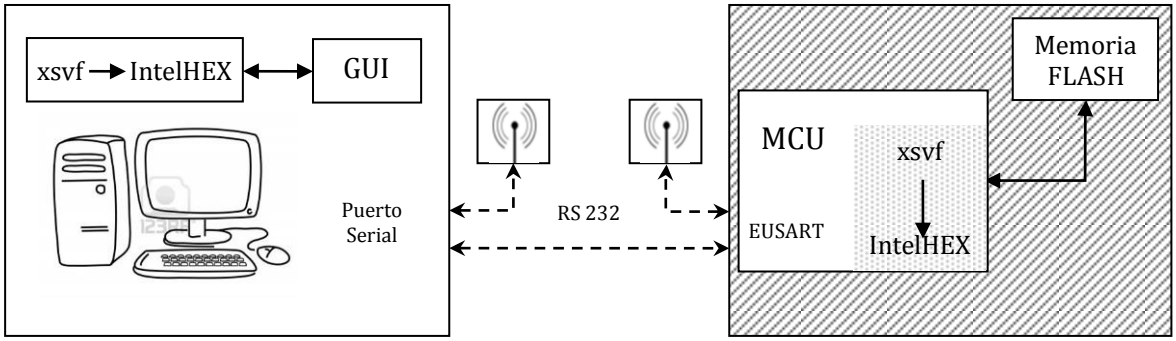
Finalmente presiona el botón "Open" para iniciar el monitoreo del puerto serial, Se mostrara un mensaje como el siguiente: "[22/05/2014 22:16:41] - Open COM4 port (C:\Archivos de programa\Eltima Software\Serial Port Monitor\SerialMonitor.exe)" para indicar que el puerto está en uso.

Por último se energiza la tarjeta prototipo y se mandan comandos hacia el MCU por medio de la herramienta "Serial Port Monitor", dependiendo el comando, la escritura o lectura de la memoria.

6.2 Validación de escritura - lectura de la memoria FLASH de forma alámbrica e inalámbrica

El siguiente paso consiste en integrar el *firmware* de recepción, transmisión y almacenamiento de un archivo en formato IntelHEX, tanto de forma alámbrica e inalámbrica.

El esquema para efectuar la prueba se muestra en la FIGURA 6.4. Inicialmente se crea y guarda el archivo IntelHEX a transmitir, desde la PC con ayuda de la GUI se elige dicho archivo. A través del puerto serial de la Estación Terrena (PC) se transmite la información del archivo original al MCU, este la decodifica para obtener el *bitstream* y almacenarlo en la memoria FLASH. Una vez terminado el proceso de escritura del archivo se solicita la lectura de la memoria, el MCU recibe el comando de lectura desde la Estación Terrena, extrae el *bitstream* de la memoria FLASH, la transmite hacia la PC y al finalizar la recepción del archivo se despliega en la pantalla de la interfaz de la Estación Terrena para verificar que la información del archivo almacenada en la memoria FLASH no fue corrompida.



Donde:

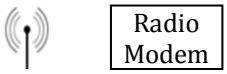


Figura 6.4. Diagrama de la verificación de escritura lectura del archivo IntelHEX en memoria FLASH.

6.2.1 Hardware a utilizar

En este caso el Hardware propuesto es:

- Microcontrolador PIC18LF2520.
- Modems RF o cable de comunicación serial.
- Memoria SST25VF064C.
- Conector DB9.
- Tarjeta de prueba (*protoboard*)

6.2.2 Software a utilizar

Las herramientas de Software para llevar a cabos esta validación es el siguiente:

- Programa IntelHEX.
- Interfaz Estación Terrena.
- Firmware de recepción, transmisión y almacenamiento en memoria.

6.2.3 Desarrollo

Se inicia por generar un archivo en formato IntelHEX a partir de un archivo de texto cualquiera sin importar el contenido del mismo; el programa IntelHEX se ejecuta y se escribe el nombre del archivo que se usará agregando la extensión. Como resultado se genera un archivo .hex dentro de la misma carpeta donde se encuentra el archivo ejecutable del programa IntelHEX. El proceso de generación debe de verse más o menos como se muestra en la FIGURA 6.5.

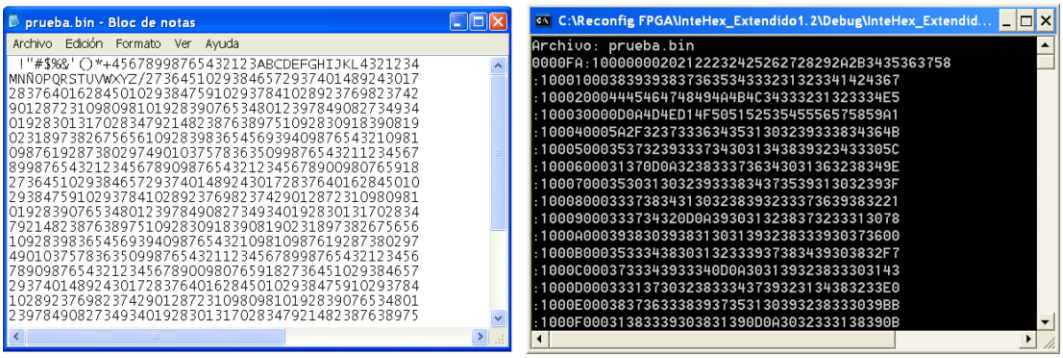


Figura 6.5. Generación de archivo .hex (IntelHEX) a partir de un archivo .xsvf.

Ahora, una vez con el firmware programado en el microcontrolador se conecta la tarjeta de pruebas con el conector DB9 y cable de comunicación serial RS232 a la PC. Se ejecuta la GUI, se definen los parámetros de comunicación de forma similar como con el "Serial Port Monitor" y se energizan los componentes en cuestión.

Después, para verificar la comunicación entre la Estación Terrena y el MCU, se pulsa el botón *Check* en la GUI, de esta forma se le solicita al MCU notifique con un *ACK* que el comando enviado fue recibido exitosamente, en la GUI se visualiza la petición y notificación. Con la comunicación lista se le indica al MCU que se realizará el proceso de escritura en la memoria con el botón *Write*, ahora se procede a seleccionar el archivo que contiene el *bitstream* a transmitir, esto se logra con el botón *Load file*, con él se despliega una ventana en la cual se selecciona el archivo y finalmente *Send all* inicia la transmisión del *bitstream.* La FIGURA 6.6 muestra un ejemplo de estos pasos.

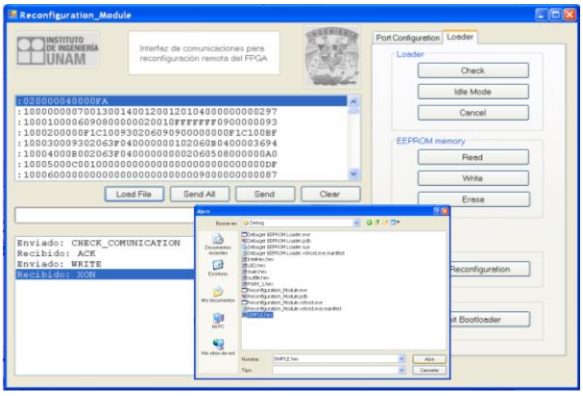


Figura 6.6. Selección de archivo IntelHel a transmitir.

La GUI tiene una barra que indica el avance en el proceso de escritura.

Una vez almacenado el *bitstream* en la memoria se procede a solicitar la lectura de la misma desde la GUI con el botón *Read*, en ese momento la GUI comienza a desplegar la información en memoria pudiendo visualizar si hubo alguna falla o el *bitstream* fue corrompido durante el proceso de escritura. Ver FIGURA 6.7 como ejemplo.

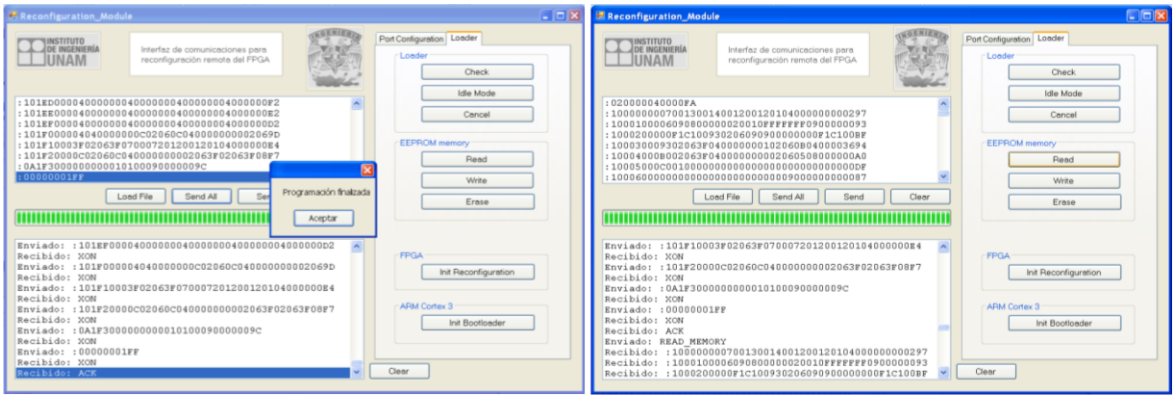


Figura 6.7. Finalización de transmisión de *bitstream* y lectura de memoria.

El proceso para llevar a cabo esta prueba de forma alámbrica e inalámbrica son muy similares, la variante es que se debe de sustituir el cable de comunicación serial por los radiomodems.

Un aspecto importante a considerar durante la ejecución de esta prueba y robustecer el sistema es crear distintas arquitecturas que impacten directamente en el tamaño de los archivos a transmitir (InteleHEX) y variar la velocidad de comunicación (*Baudrate*). Con esto se busca la depuración del módulo y su óptima operación.

6.3 Validación de *firmware* de reconfiguración de FPGA con la tarjeta de desarrollo Spartan 3E

El objetivo de esta prueba es validar, por un lado, el software de operación de la Estación Terrena y, por otro lado, el código *firmware* de reconfiguración que irá en el nuevo microcontrolador, y así comprobar que la lógica que describe la transferencia del *bitstream* mediante el protocolo JTAG es correcta.

Esta prueba inicia a partir de que el *bitstream* ha sido exitosamente en la memoria FLASH (ver sección 6.2). Una vez que se verificó que la información guardada está integra el siguiente paso es reconfigurar el FPGA. En la FIGURA 6.8 está compuesta por las etapas que conforman la validación del *firmware* de reconfiguración.

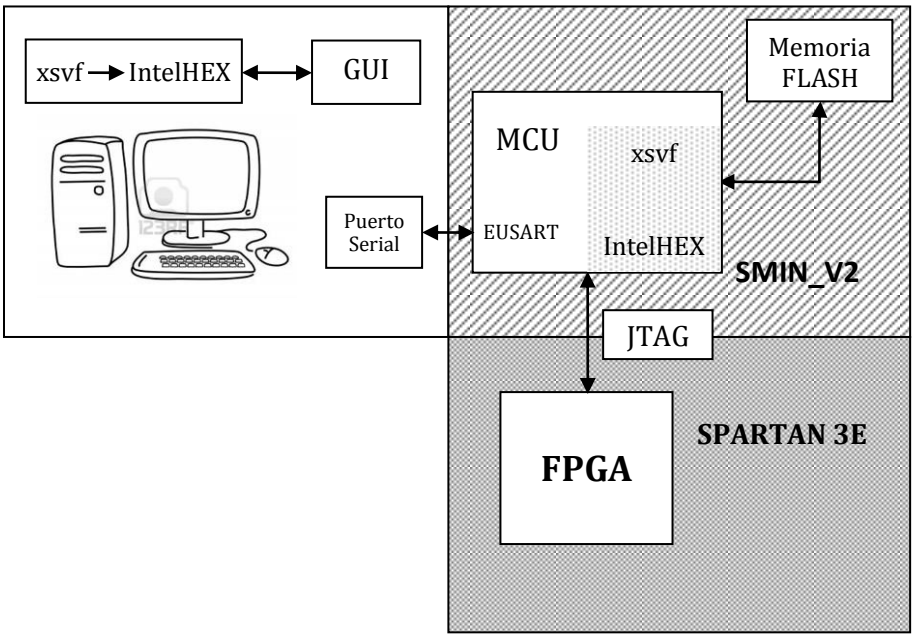


Figura 6.8. Diagrama de bloques de validación de carga de *bitsteam* en tarjeta Spartan 3E.

6.3.1 Hardware a utilizar

- Microcontrolador PIC18LF2520.
- Modems RF o cable de comunicación serial.
- Memoria SST25VF064C.
- Conector DB9.
- Tarjeta de prueba (*protoboard*).
- Tarjeta de desarrollo Spartan 3E.

6.3.2 Software a utilizar

- Herramienta ISE de Xilinx.
- Programa IntelHEX.
- Interfaz Estación Terrena.
- Firmware de recepción, transmisión y almacenamiento en memoria.
- Firmware de reconfiguración ajustado al PIC.

6.3.3 Desarrollo

Para reconfigurar el FPGA de la tarjeta de desarrollo Spartan 3E desde la nueva plataforma FPGA propuesta (SMIN_V2) es necesario seleccionar, en la tarjeta de desarrollo, el modo "JTAG *only"* mediante la configuración de unos *jumpers* (J28) y, en la tarjeta SMIN_V2, seleccionar el modo "FPGA externo" con la configuración de un *jumper* (JTAG jumper).

Lo primero a realizar es diseñar y sintetizar en la herramienta ISE de Xilinx una arquitectura que describe un sistema digital sencillo, por ejemplo, una compuerta AND y generar un archivo .xsvf. Posteriormente convertir el archivo .xsvf en formato IntelHex utilizando el software IntelHEX, que estará almacenado en la memoria FLASH. Adicionalmente el MCU debe contar con el *firmware* de reconfiguración ajustado al PIC.

Ahora se conecta físicamente la tarjeta de SMIN_V2 con la PC a través del cable serial, el SMIN_V2 con la tarjeta de desarrollo Spartan 3E a través del puerto JTAG y se energiza el sistema. Con ayuda de la GUI de la Estación Terrena se almacena el archivo intelHex dentro de la memoria FLASH, siguiendo los mismos pasos de la prueba anterior (ver sección 6.2), y finalmente presionando el botón *Init Reconfiguration* de la GUI iniciar el proceso de reconfiguración del FPGA de la tarjeta de desarrollo Spartan 3E.

6.4 Validación de reconfiguración de FPGA de la tarjeta SMIN_V2

En esta prueba no solo se valida la reconfiguración del FPGA del SMIN_V2, se pone a prueba todos los módulos existentes que conforman el sistema, esto implica evaluar su desempeño desde un bajo nivel de exigencia hasta el máximo.

En caso de surgir algún error durante la validación de reconfiguración del FPGA a bordo, es muy probable que sea debido a un problema de hardware entre las terminales del MCU y el FPGA.

Se usa la GUI de Estación Terrena para controlar las tareas a ejecutar por el MCU, unidad de procesamiento para generar las señales JTAG.

Esta prueba es muy similar a la anterior, sólo difiere en que se reconfigura el FPGA propio de la tarjeta SMIN_V2 propuesta. La
FIGURA 6.9 representa de forma gráfica el objetivo que persigue este trabajo de tesis.

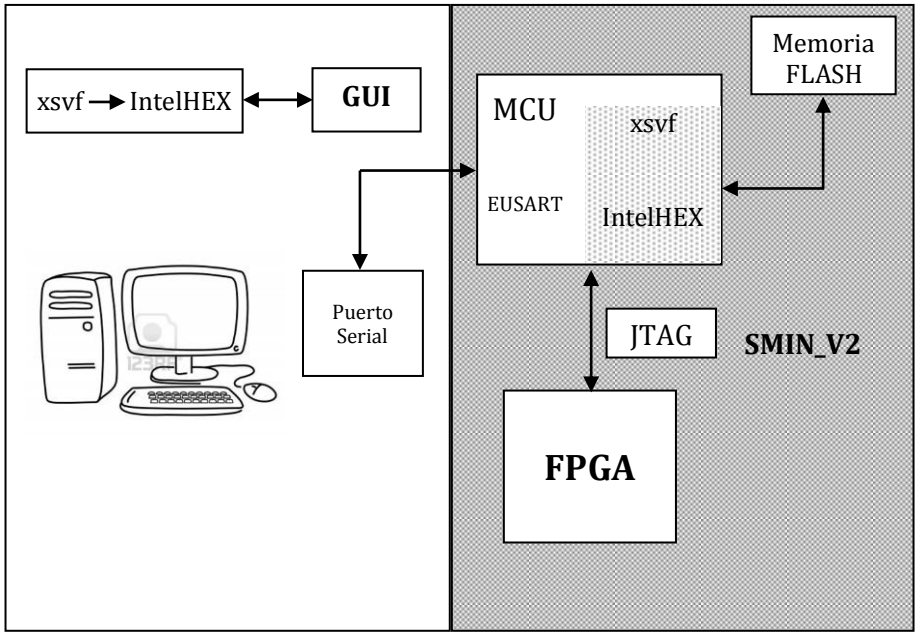


Figura 6.9. Diagrama de bloques de validación de reconfiguración del FPGA en tarjeta propuesta SMIN_V2.

6.4.1 Hardware a utilizar

- Microcontrolador PIC18LF2520.
- Modems RF o cable de comunicación serial.
- Memoria SST25VF064C.
- Conector DB9.
- Tarjeta de prueba (*protoboard*).
- Tarjeta de desarrollo Spartan 3E.

6.4.2 Software a utilizar

- Herramienta ISE de Xilinx.
- Programa IntelHEX.
- Interfaz Estación Terrena.
- Firmware de recepción, transmisión y almacenamiento en memoria.
- Firmware de reconfiguración ajustado al PIC.

6.4.3 Desarrollo

Hay que recordar que el SMIN_V2 debe de estar configurado en modo “FPGA externo” y el MCU debe ser programado con el *firmware* de recepción, transmisión y almacenamiento en memoria y *firmware* de reconfiguración ajustado al PIC.

Esta prueba inicia a partir de tener el archivo de reconfiguración en formato IntelHEX. Luego de generar estos archivos, se conectó físicamente la tarjeta de SMIN_V2 con la PC, de igual forma como se hizo en la prueba anterior (ver sección 6.3). Se carga en memoria el archivo IntelHEX, puede usarse algún archivo generado en pruebas anteriores.

Mediante la GUI de la Estación Terrena se transmite el archivo de reconfiguración elegido hacia el MCU y este lo almacene en la memoria FLASH. Una vez almacenado el *bitstream* de reconfiguración, se envía el comando de control para inicializar la reconfiguración del FPGA. Una vez que se finaliza este proceso, el MCU envía un byte que indica que el FPGA fue configurado de forma exitosa.

6.5 Resumen

En el capítulo se ilustra en forma de bloques los elementos que conforman al sistema y exhibe el esquema de verificación en forma sucesiva, persiguiendo incluir el bloque validado dentro de otro bloque a validar hasta el bloque más complejo. De esta forma se la detección de problemas y solución se vuelve más sencilla.

El capítulo siguiente se enfoca en la carga de nuevos programas en la plataforma FPGA propuesta, describiendo la forma correcta de crear y almacenar cualquier arquitectura de reconfiguración.

7

Carga de nuevos programas a la plataforma FPGA

Ahora que se cuenta con el conocimiento, del hardware que integra la nueva versión del Sistema Mínimo (SMIN_V2) así como de las herramientas de software utilizadas durante el proceso de validación, se pueden realizar arquitecturas personalizadas de la misma manera en como se desarrollaron las pruebas de validación.

En éste capítulo se muestra detalladamente el proceso de carga de nuevas configuraciones de computo a la nueva plataforma, desde generar la arquitectura con ayuda de las herramientas ISE e iMPACT, ambas de Xilinx, hasta transmitir, interactuar y reconfigurar el FPGA con la GUI de la Estación Terrena.

7.1 ISE suite de Xilinx

El desarrollo de la arquitectura inicia por crear la lógica o comportamiento que se quiere implantar en el FPGA, para ello se usa la herramienta ISE suite de Xilinx, una vez instalado el software se abre la aplicación o ejecutable. La FIGURA 7.1 muestra un ejemplo de la ubicación del ambiente de software integrado (*ISE*) en el explorador de Windows XP dentro del menú "Todos los programas".

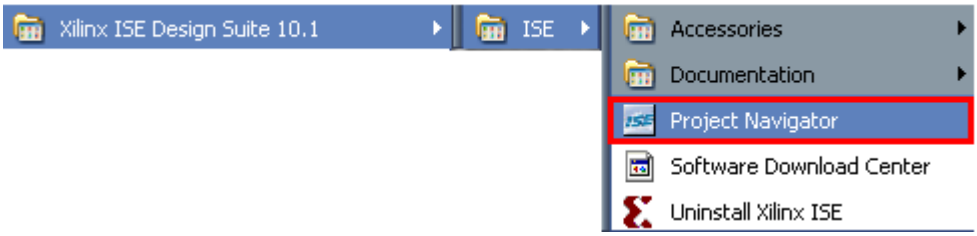


Figura 7.1. Ambiente de software integrado de Xilinx.

Se tiene que crear un proyecto y seleccionar su ubicación. Presionando el botón de *Next* lleva a la configuración del proyecto.

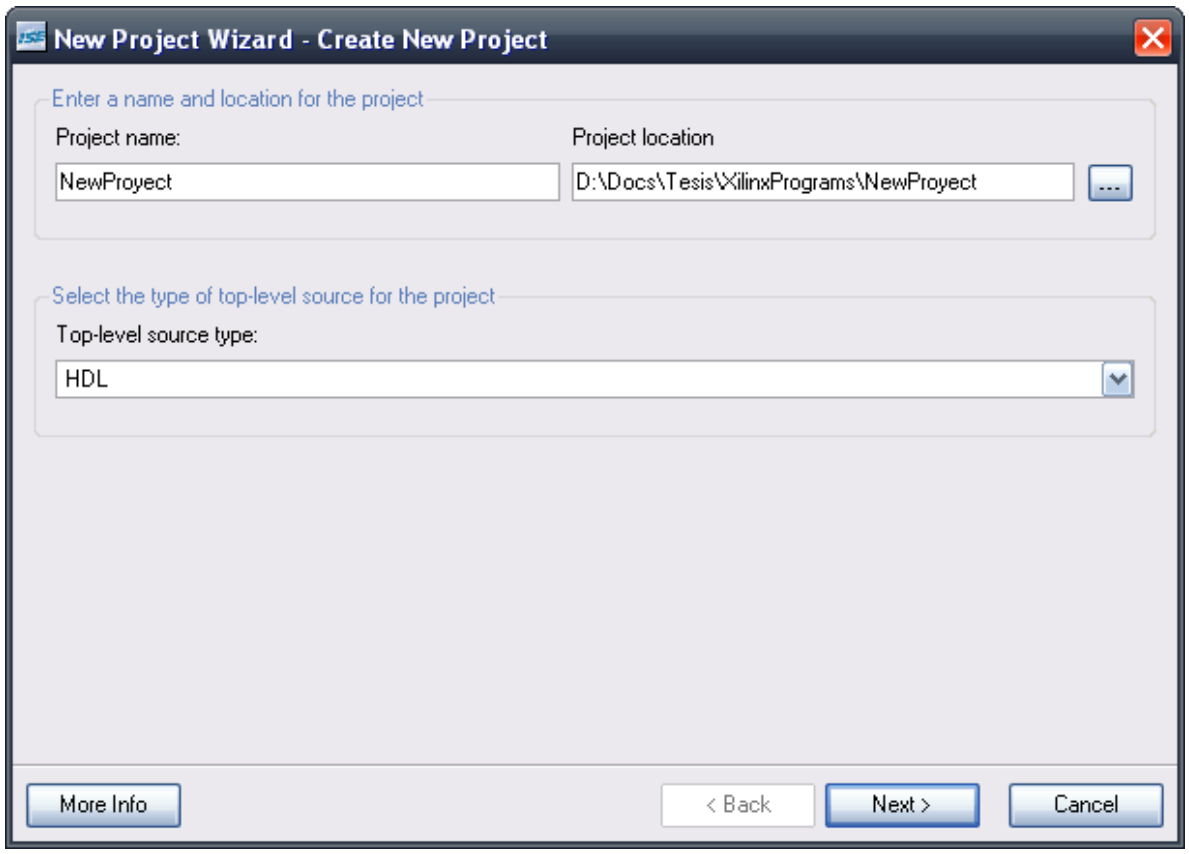


Figura 7.2. Ventana que muestra la creación de un nuevo proyecto en ISE de Xilinx.

Dentro de esta ventana se puede seleccionar una amplia gama de combinaciones. Para nuestro caso se muestran brevemente las opciones de configuración y los posibles valores que aplican.

- *Product Category.* Seleccionar la opción *All* o *General Purpose.*
- *Family* Se puede elegir distintos tipos de familias de dispositivos, en este caso, el FPGA del sistema propuesto SMIN_V2 corresponde a la familia Spartan3E.
- *Device*. Se puede elegir entre distintos dispositivos FPGAs dentro de la familia Spartan3E. el SMIN_V2 tiene un FPGA XC3S1600E.
- *Package*. FG320.

Ahora se conservan los siguientes parámetros con los valores por defecto:

- *Speed*: -5.
- *Top Level Source Type*: HDL.
- *Synthesis Tool*: XST (VHDL/Verilog).
- *Simulator*: ISE Simulator (VHDL/Verilog).
- *Preferred Language*: Verilog.

La FIGURA 7.3 muestra la ventana de configuración con los valores correspondientes para el FPGA XC3S1600E.

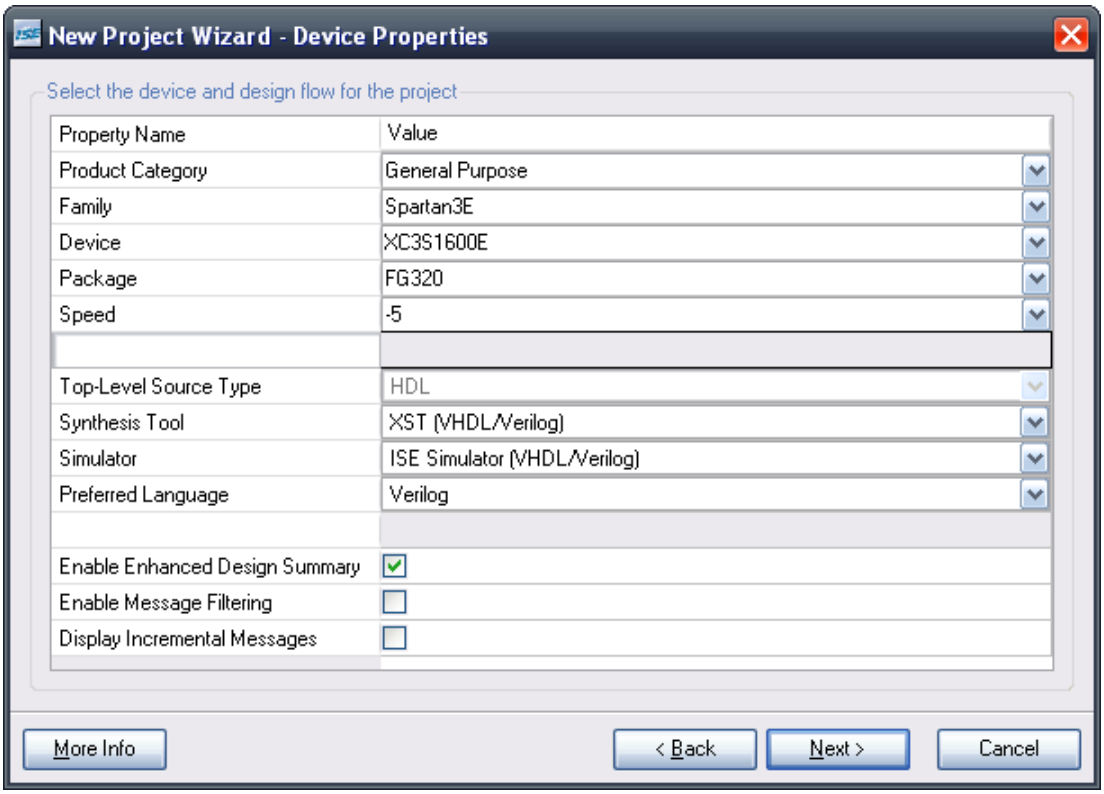


Figura 7.3. Configuración de proyecto para FPGA XC3S1600E.

Ahora se agrega una nueva fuente al proyecto, se presiona el botón *New Source* en la FIGURA 7.4 para iniciar su creación.

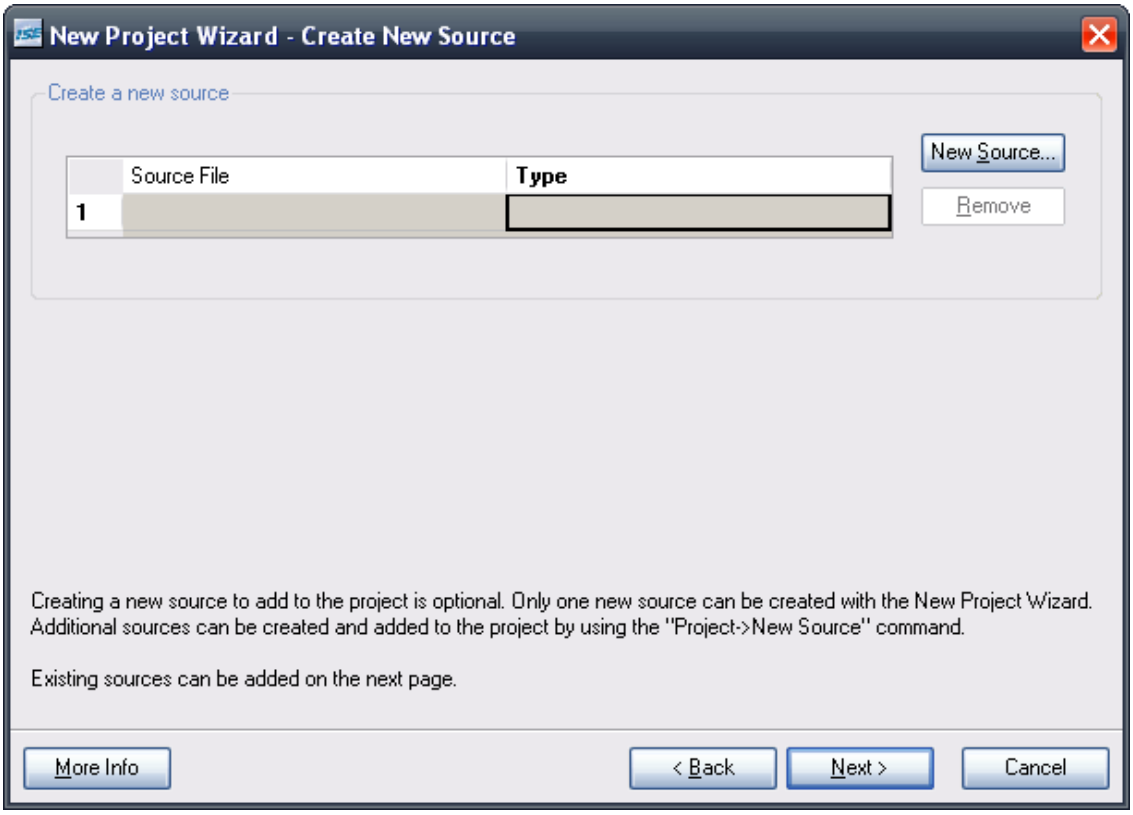


Figura 7.4. Creación de un nuevo archivo fuente.

Todo proyecto requiere de una fuente para crear la descripción de hardware que será implantada en el FPGA. Dicha fuente, al ser creada debe de especificar el tipo de archivo, nombre y dónde será guardado. La FIGURA 7.5 muestra lo antes mencionado.

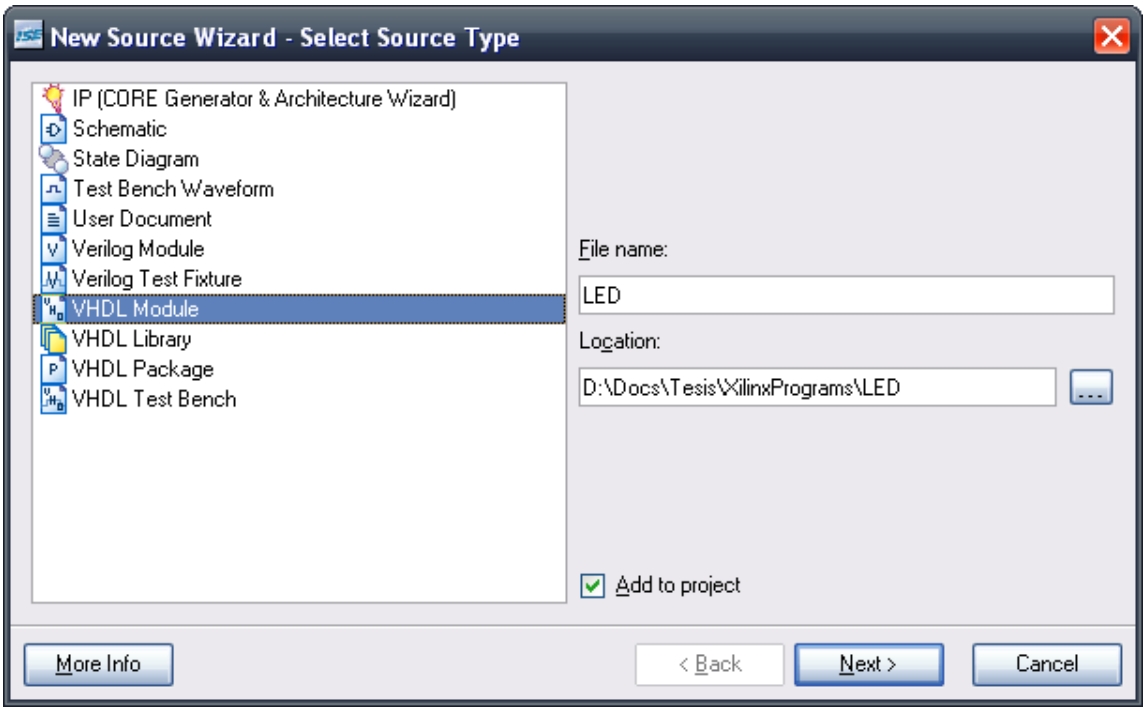


Figura 7.5. Selección de tipo de fuente, nombre y ubicación.

Ahora se define la cantidad de terminales que serán usadas como entradas y salidas, cada terminal debe de tener asignado un nombre único. Por ejemplo, en la FIGURA 7.6 se asigna sólo una terminal de salida.

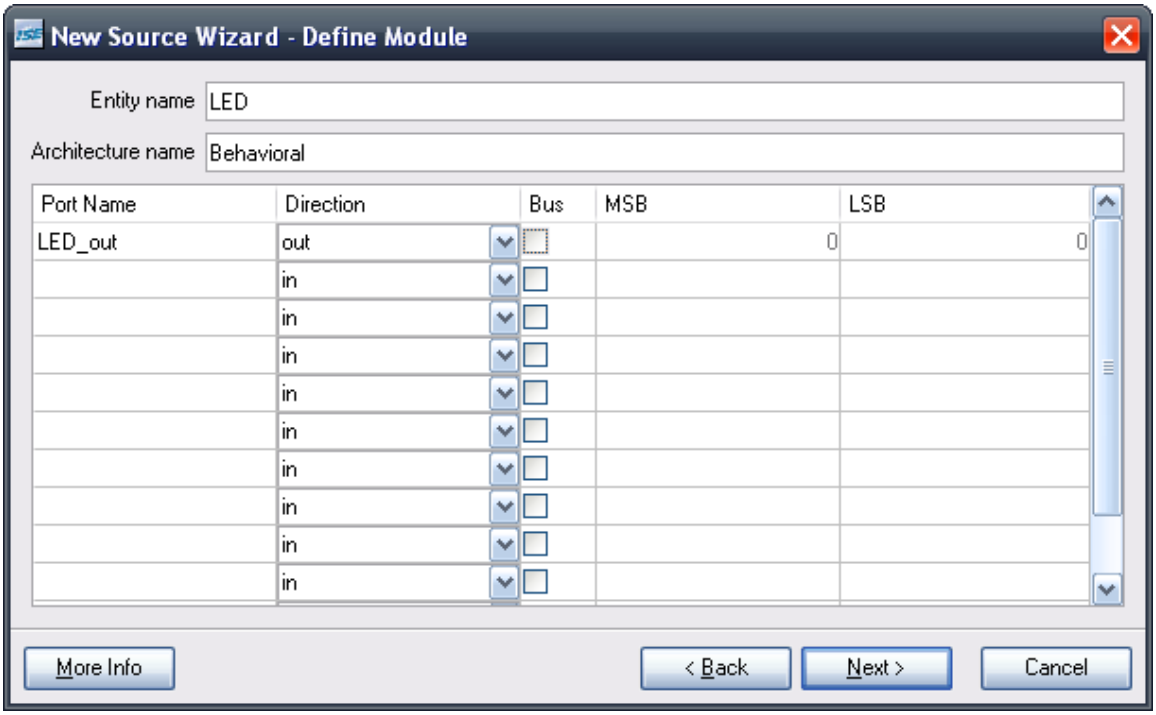


Figura 7.6. Asignación terminales a incluir en la arquitectura.

El archivo fuente ya está listo y muestra el resumen de los pasos realizados para su creación. La FIGURA 7.7 muestra la ventana de resumen

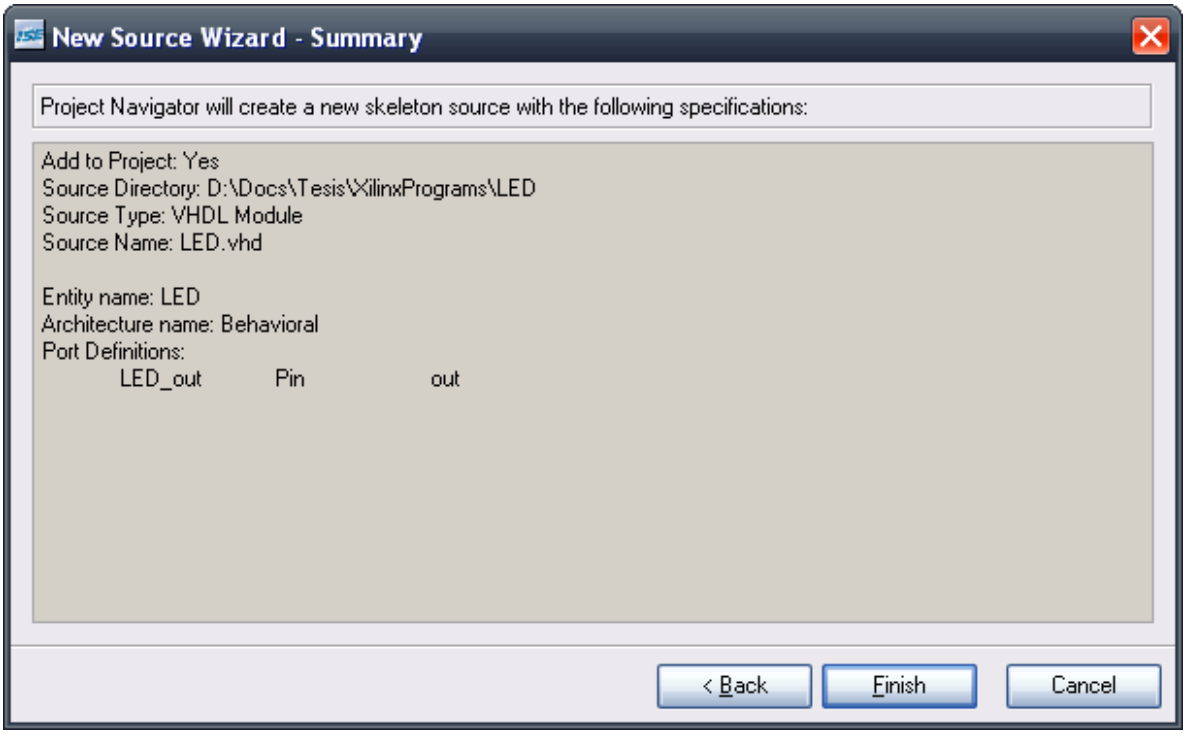


Figura 7.7. Resumen de fuente creada.

Al finalizar el resumen se muestra la fuente creada y su tipo como parte del nuevo proyecto. Para continuar hay que dar *click* en el botón *Next*, ver FIGURA 7.8.

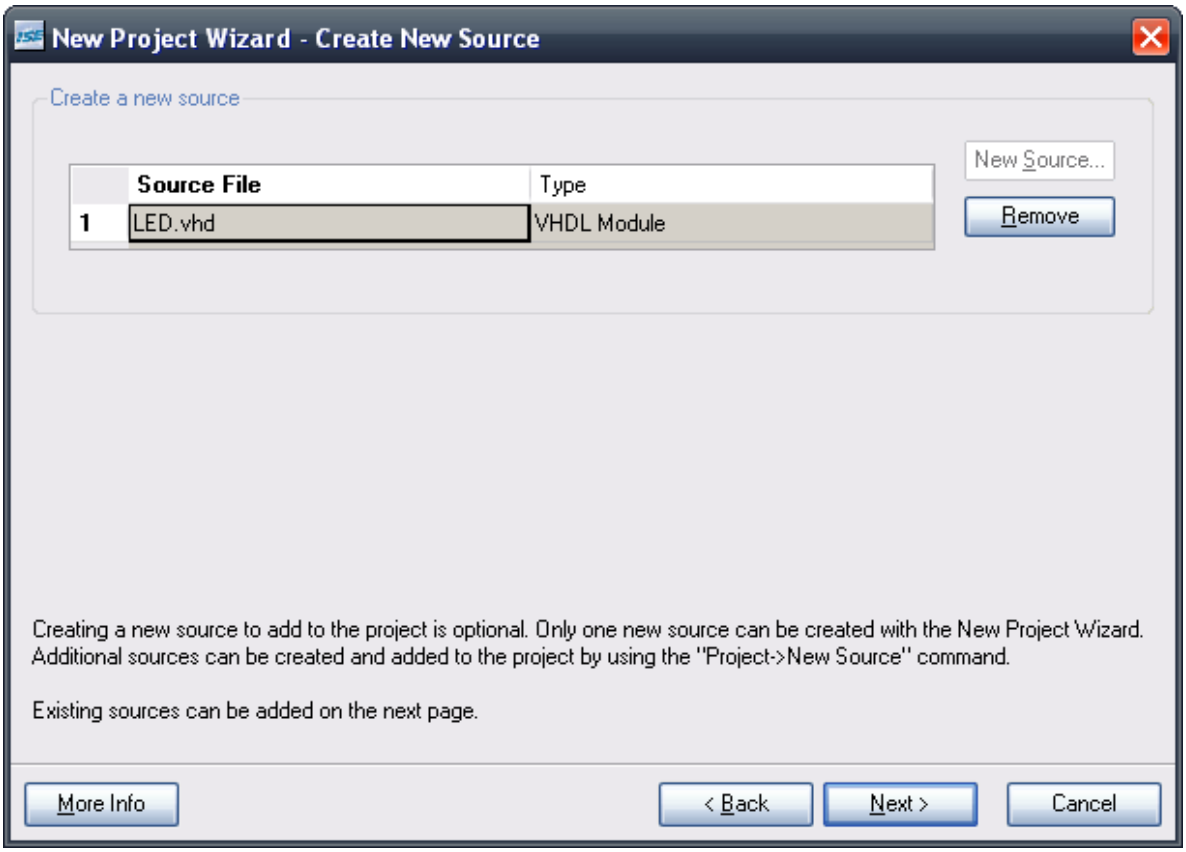


Figura 7.8. Fuente agregada al proyecto.

En caso de desear agregar o eliminar alguna otra fuente cargada en el proyecto se puede usar el botón *Add Source* o *Remove*, respectivamente. De no ser así dar *click* en el botón *Next* de la FIGURA 7.9.

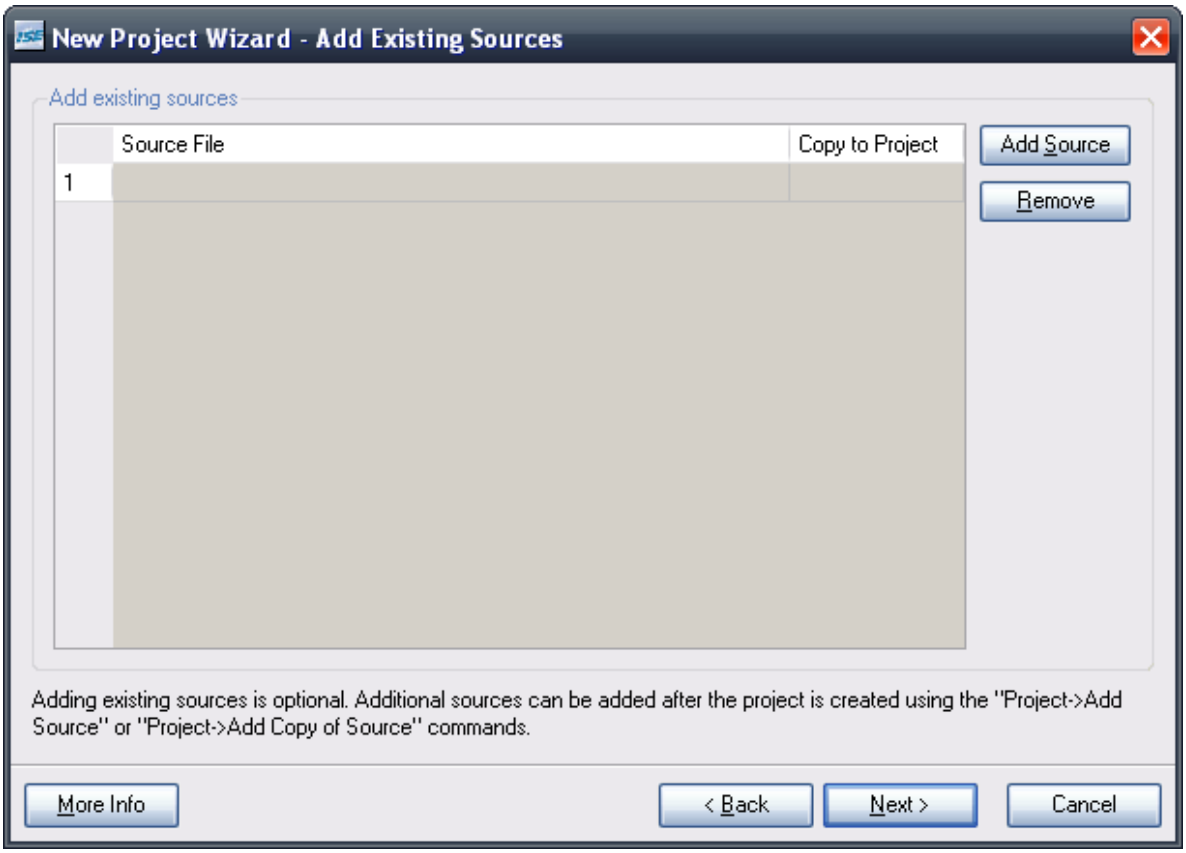


Figura 7.9. Ventana para agregar o eliminar fuentes (opcional).

Listo!!! El proyecto ha sido creado y se mostrará la ventana con el resumen del proyecto, la ventana debe ser similar a la mostrada en la FIGURA 7.10.

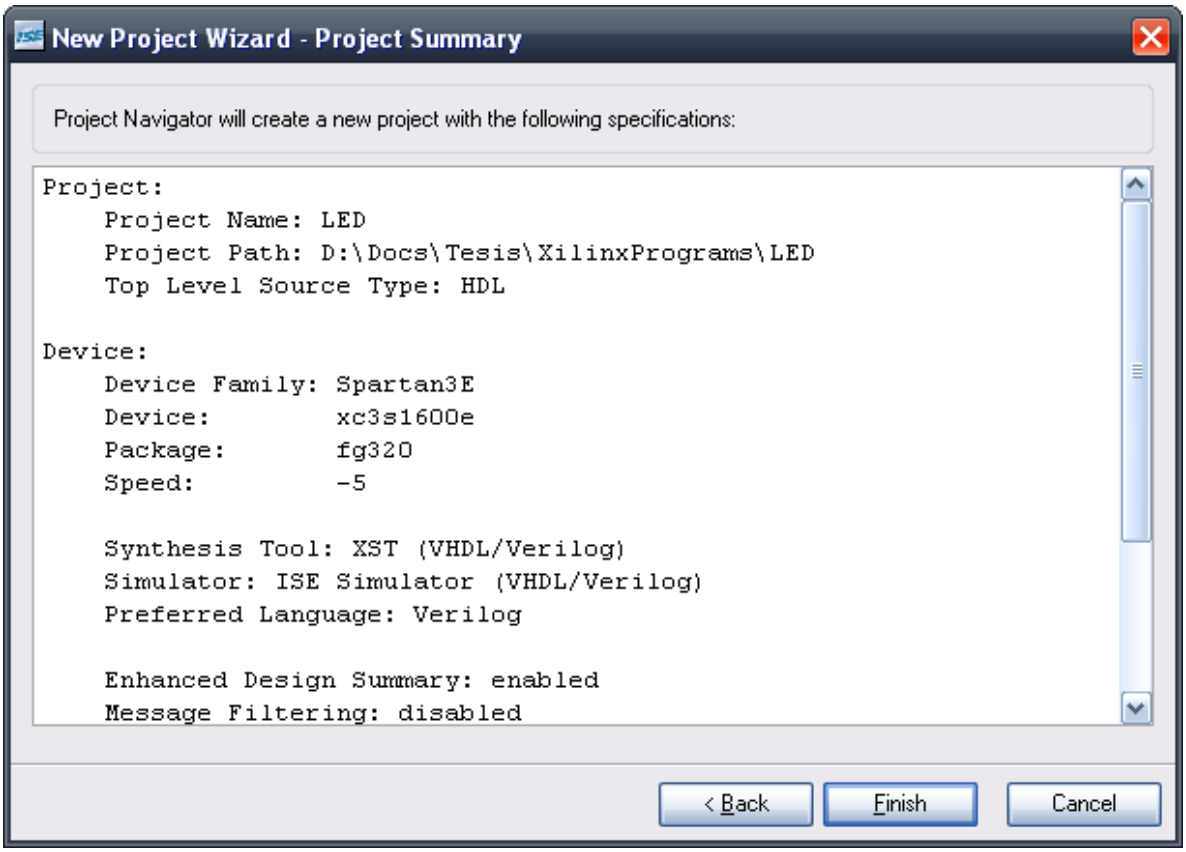


Figura 7.10. Resumen del proyecto creado.

Se da *click* en *Finish* y se abrirá la ventana de la herramienta ISE de Xilinx. Con el código creado automáticamente por la herramienta, de acuerdo a la cantidad de terminales seleccionadas y el tipo de fuente a usar, en este caso, se describe el comportamiento de la arquitectura de interés mediante el lenguaje de descripción de hardware (VHDL).

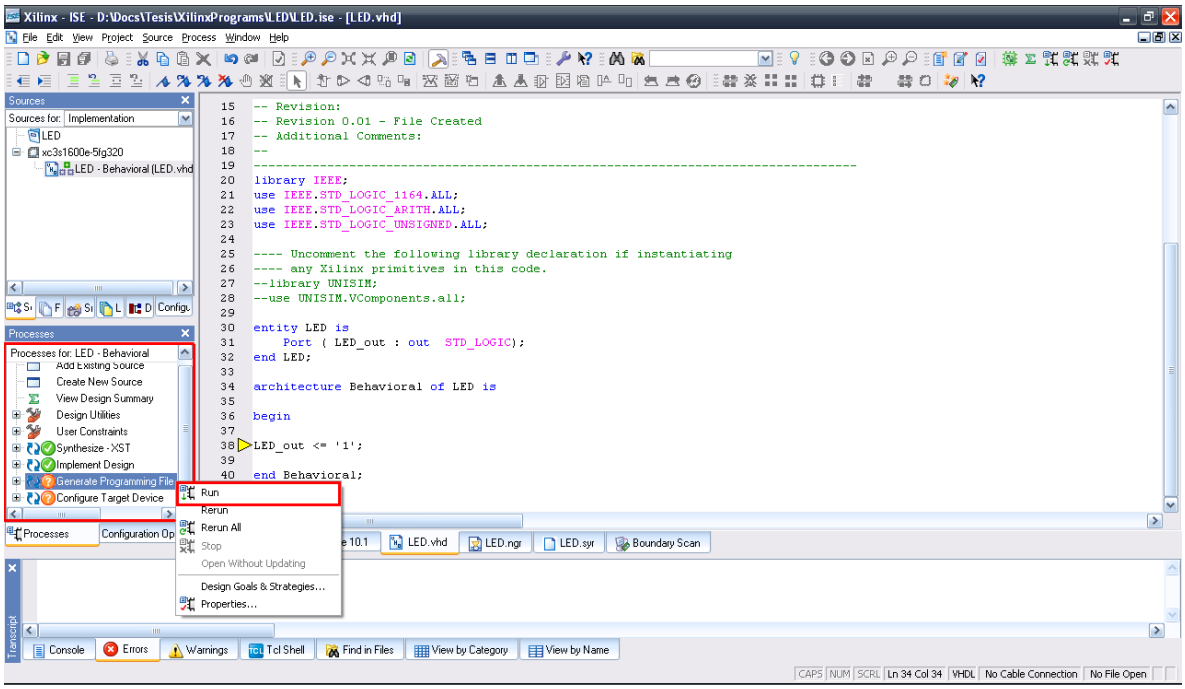


Figura 7.11. Ventana de trabajo de herramienta ISE de Xilinx.

Aunque la herramienta escribió código automáticamente, el comportamiento del sistema está vacío y tiene que ser implementada antes de continuar, de lo contrario, marcará errores.

Al finalizar la descripción de hardware se corren los próximos cuatro pasos para realizar la síntesis y creación del archivo de reconfiguración:

- *Synthesis – XST*.
- *Implement Design*.
- *Generate Programming File*.
- *Configure Target Device*.

Para todos los pasos anteriores se presiona el click derecho sobre el paso y se selecciona *Run*. En caso de no haber errores se mostrará una paloma blanca, de este tipo: . En la figura siguiente se muestran dos de los pasos ya ejecutados exitosamente y dos pendientes por ejecutar.

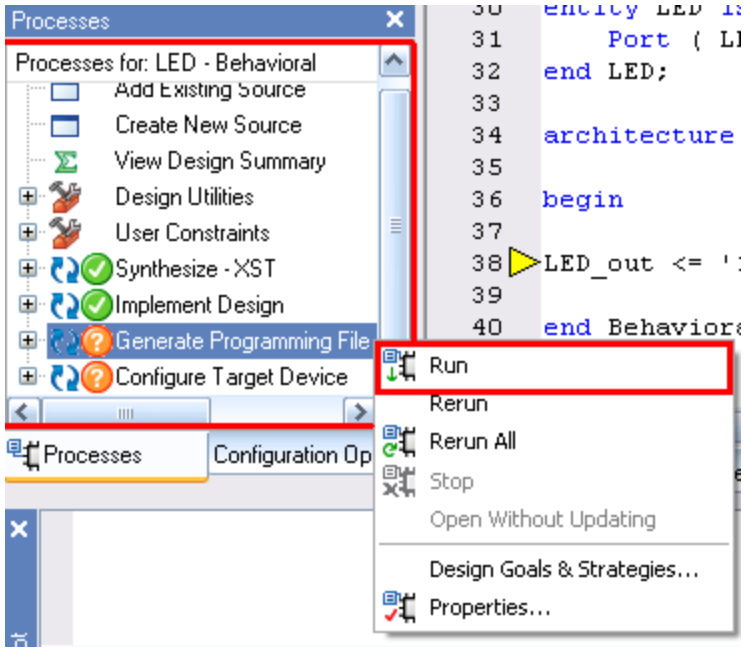


Figura 7.12. Síntesis y creación del archivo de reconfiguración.

Al ejecutar el último paso se iniciará la herramienta iMPACT de Xilinx desde el software ISE de Xilinx, se desplegará un mensaje de advertencia indicando que se la interfaz de iMPCAT se mostrará. Presionar OK para continuar.

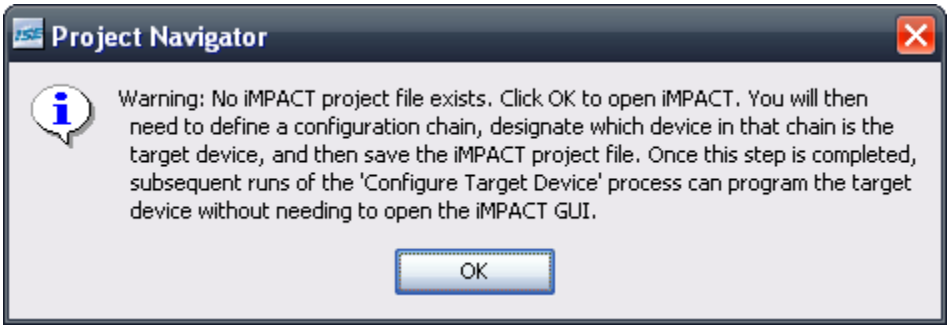


Figura 7.13. Ventana de advertencia de iMPACT.

La siguiente ventana permite configurar directamente al FPGA, entre otros dispositivos de Xilinx, sin embargo, se selecciona la opción de generar el archivo de reconfiguración .xsvf. Dar *click* en el botón *Finish* para seleccionar la carpeta destino del archivo de reconfiguración. La muestra la ventana.

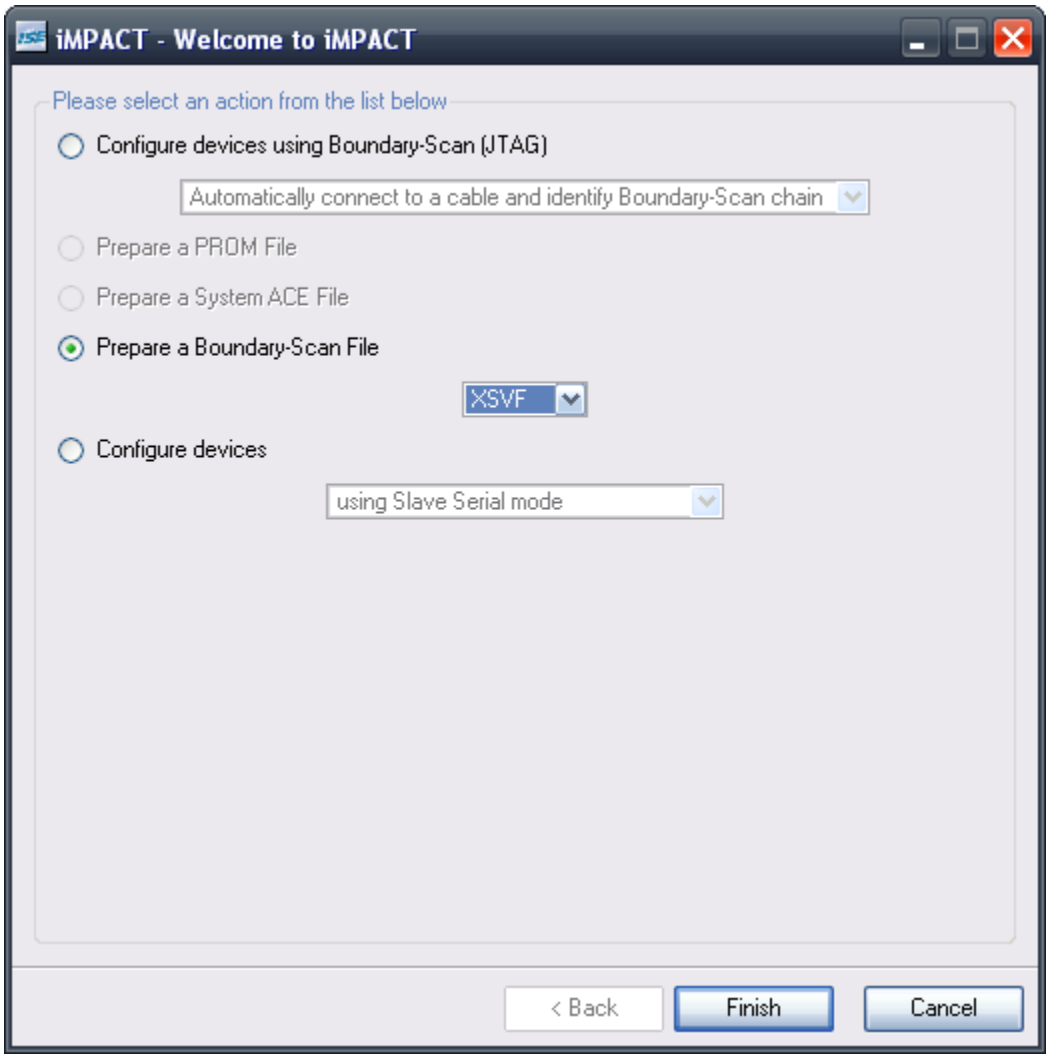


Figura 7.14. Selección de generación de archivo .xsvf.

7.2 Software InteLHEX

Recordando lo descrito en la sección 4.1.2, el archivo intelHex contiene caracteres en ASCII, que representan información binaria. Información en el archivo .xsvf que resulta de crear la arquitectura en la herramienta ISE e iMPACT de Xilinx.

Otra forma de ver esta herramienta IntelHEX es, simplemente como convertidor de archivos .xsvf a .hex para transmitirlo.

Nota: El software IntelHEX está diseñado para buscar el archivo .xsvf dentro de la misma carpeta donde su encuentra el archivo ejecutable InteleHex_Extendido.exe. De igual manera se colocará en la misma carpeta el archivo .hex creado. Por tanto, en caso de no

tener el archivo .xsvf dentro de la misma carpeta que la herramienta, esta mandara un error y se cerrará.

Su uso es sumamente sencillo, solo se ejecuta el archivo IntelHex_Extendido.exe y se desplegará una ventana con en la FIGURA 7.15. Se ingresa el nombre del archivo Ejemplo.xsvf y se presiona *enter*.

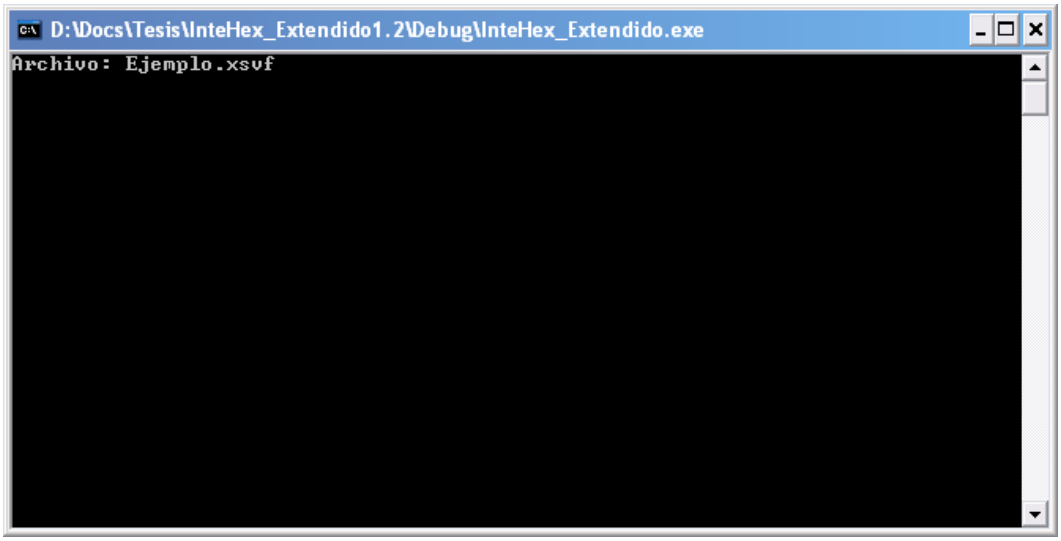


Figura 7.15. Ventana al abrir archivo ejecutable IntelHex_Extendido.exe.

Automáticamente se desplegará en pantalla los datos hasta terminar. El mensaje final en ventana es "Listo!!", tal cual se muestra en la FIGURA 7.16. Ahora el archivo Ejemplo.hex está creado.

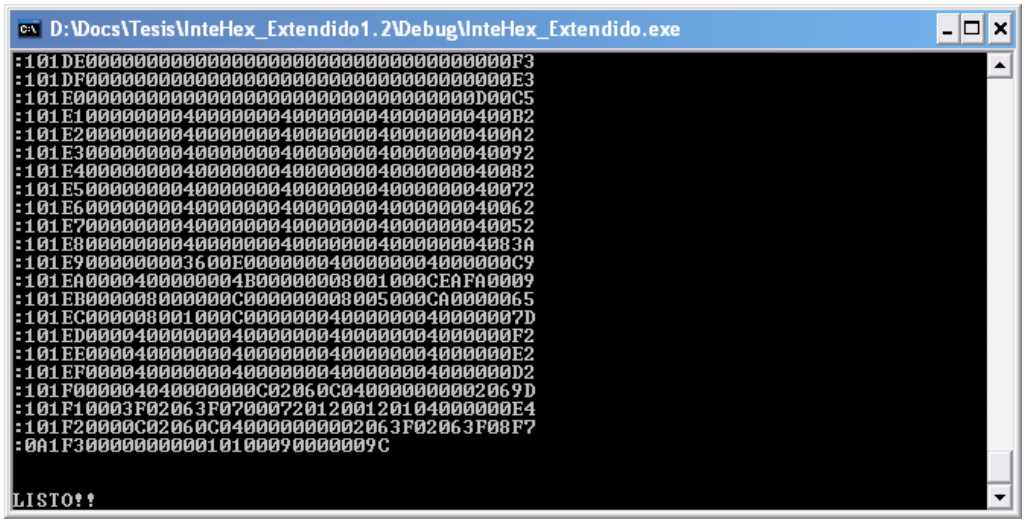


Figura 7.16. Ejecución terminada de software IntelHEX.

7.3 GUI Estación Terrena

La interfaz gráfica de usuario (GUI, en inglés *Graphical User Interface)* es software propio realizado 100% en el Instituto de Ingeniería de la UNAM. Diseñado específicamente para interactuar con la plataforma de reconfiguración del FPGA (SMIN_V1 y SMIN_V2).

Para hacer uso de esta herramienta se emplea el archivo ejecutable Reconfiguracion_Module.exe. La FIGURA 7.17 muestra la apariencia inicial de la GUI.

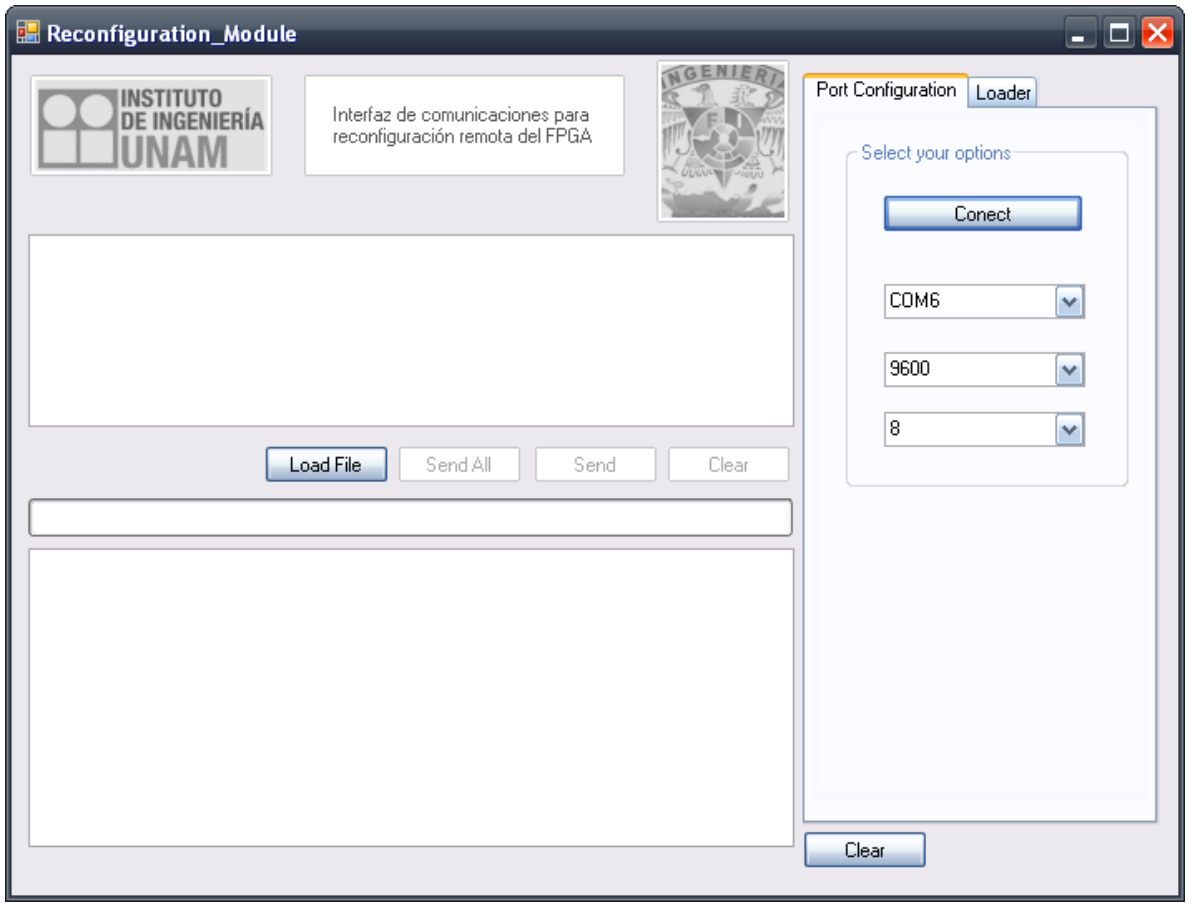


Figura 7.17. Interfaz gráfica (GUI) de Estación Terrena.

El procedimiento adecuado para reprogramar el FPGA es el siguiente:

1. Es buena práctica realizar la configuración de comunicación previamente a otros procesos, para verificar que ambos sistemas están sincronizados y evitar repetir el procedimiento.

   En la FIGURA 7.18 se resalta con un recuadro rojo la sección en cuestión. Hay tres listas desplegables, que se pueden identificar con una flecha que apunta hacia abajo, la primera en orden descendente selecciona el puerto de comunicación, ese puerto debe de ser el mismo donde se encuentra el cable serial conectado. La

segunda selecciona la velocidad de comunicación que va desde 2400 hasta 57600 kbps. La tercera la cantidad de bits de datos por carácter.

Para abrir el puerto y establecer comunicación hay que presionar el botón *Connect* y se despliega una ventana de aviso indicando la acción.

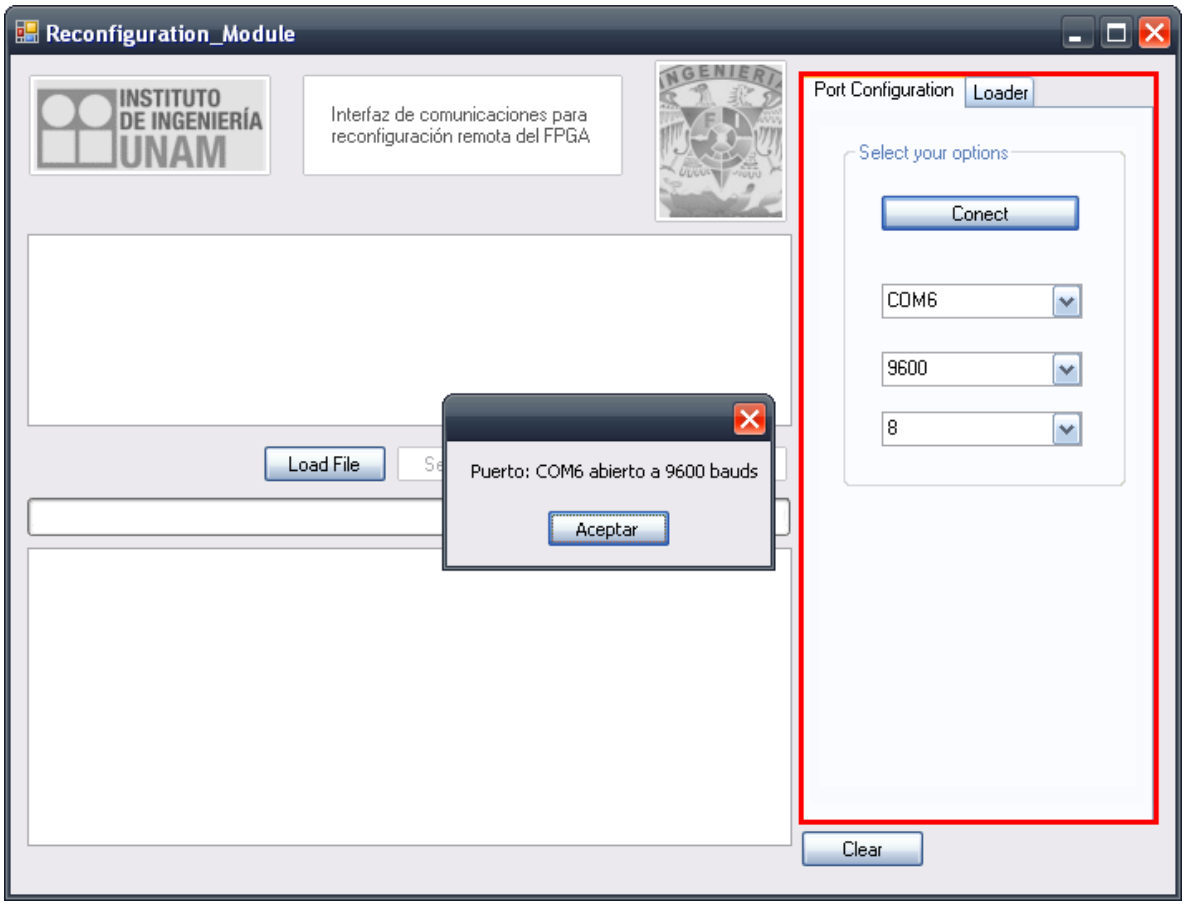


Figura 7.18. Configuración del puerto de comunicación de Estación Terrena.

Al dar *click* en el botón Aceptar la pestaña adjunta (*Loader*) a *Port Configuration* se autoselecciona.

2. En la FIGURA 7.19 se encierran tres botones que gestionan la comunicación Estación Terrena – plataforma SMIN_V2:

   - *Check.* Verifica la comunicación entre el MCU y la PC.
   - *Idle Mode.* Indica el MCU entrar en estado de bajo consumo.
   - *Cancel.* Interrumpe la transmisión del *bitstream*.

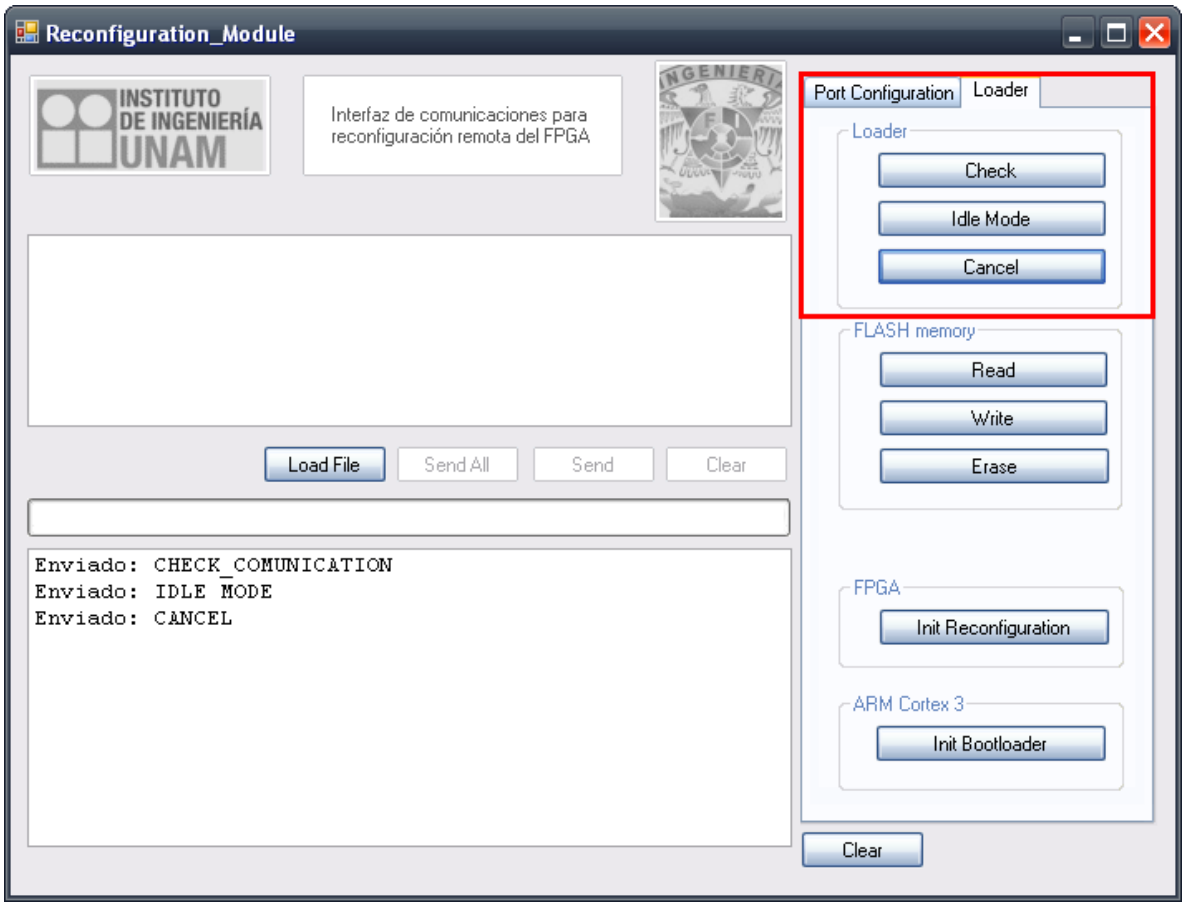


Figura 7.19. Verificación de comunicación Estación Terrena – SMIN_V2.

El resto de las opciones para la pestaña *Loader* se verá más adelante.

3. El siguiente paso es seleccionar el archivo a transmitir. La FIGURA 7.20 muestra el recuadro donde se ubica esta parte. El botón *Load File* abre una ventana de explorador para elegir el archivo a transmitir.

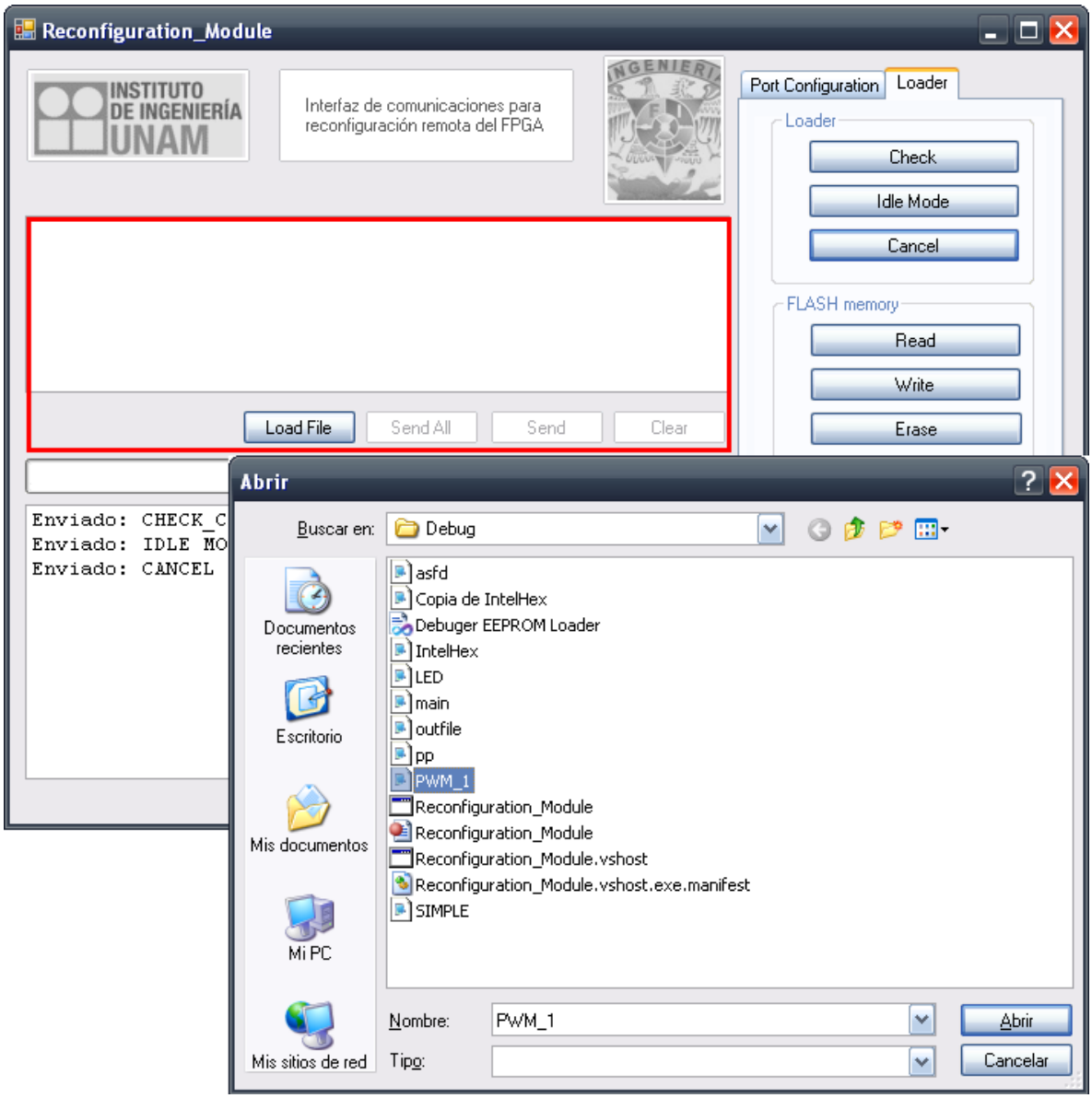


Figura 7.20. Selección de archivo en formato IntelHex a enviar.

Posteriormente se muestra el contenido del archivo y se habilitan los botones: (ver FIGURA 7.21)

- *Send All*. Inicia la rutina de envío de *bitstream*.
- *Send*. Envía una línea de información.
- *Clear*. Elimina los datos del archivo abierto.

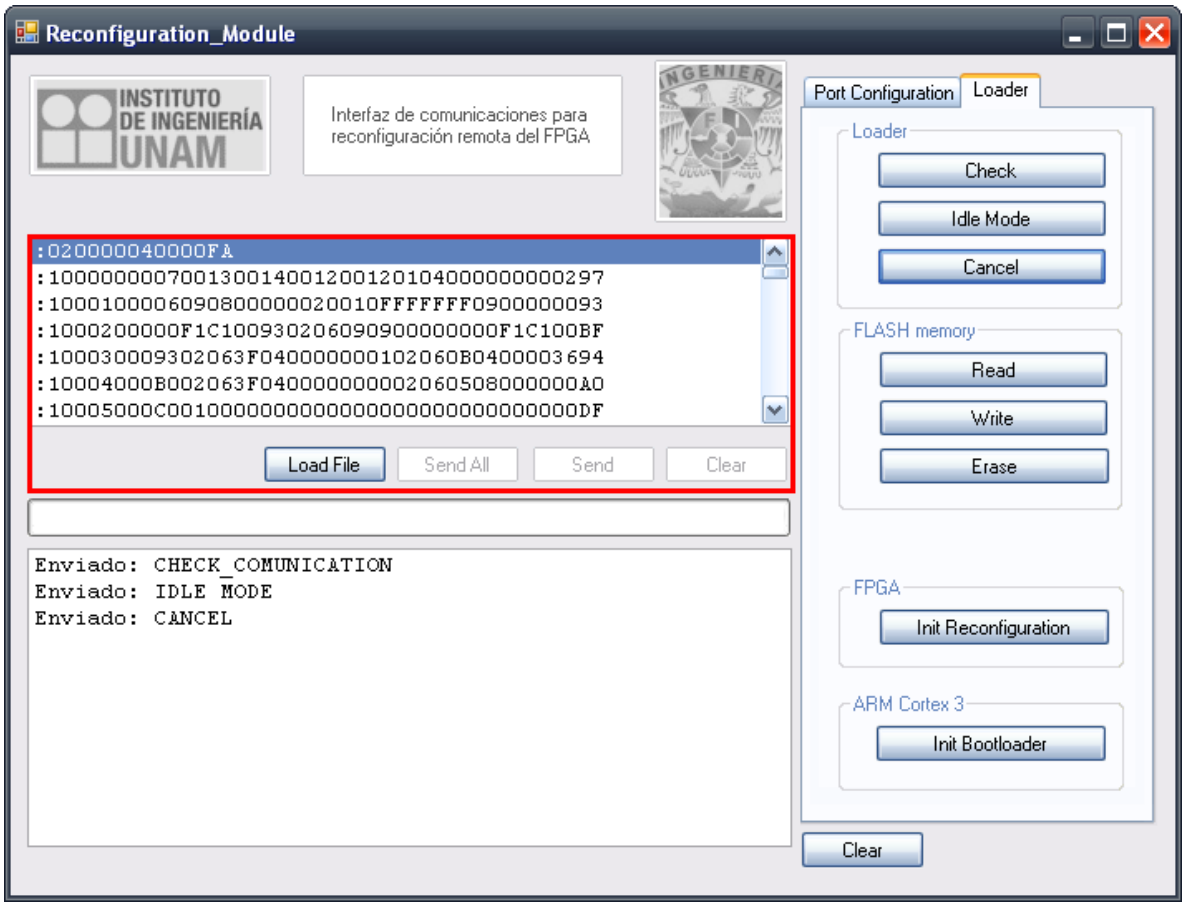


Figura 7.21. Archivo de reconfiguración listo para ser transmitido.

4. Ahora se le indica al MCU que se desea realizar la transmisión del archivo y almacenarlo en la memoria FLASH, esto se realiza al presionar el botón *Write* dentro de la sección *FLASH memory* de la pesaña *Loader*. Con esto el MCU está al pendiente de recibir la trama de información. El envío puede hacerse de dos maneras:

   - *Send All.* Transmité el *bitstream* en un solo proceso.
   - *Send.* Envía línea por línea del *bitstream*.

   No se recomienda hacerlo mediante la segunda opción ya que será mucho más tardado. El botón *Clear* borra el archivo cargado.

   El avance del proceso de transmisión del *bitstream* se puede referenciar a la barra de proceso en color verde (ver FIGURA 7.22).

   Para finalizar el paso de transferencia de *bitstream* a la memoria FLASH, se verifica la integridad de los datos presionando el botón *Read*.

En caso de haber detectado alguna falla en la transmisión y guardado de información el botón *Erase* envía el comando para borrar el contenido de la memoria.

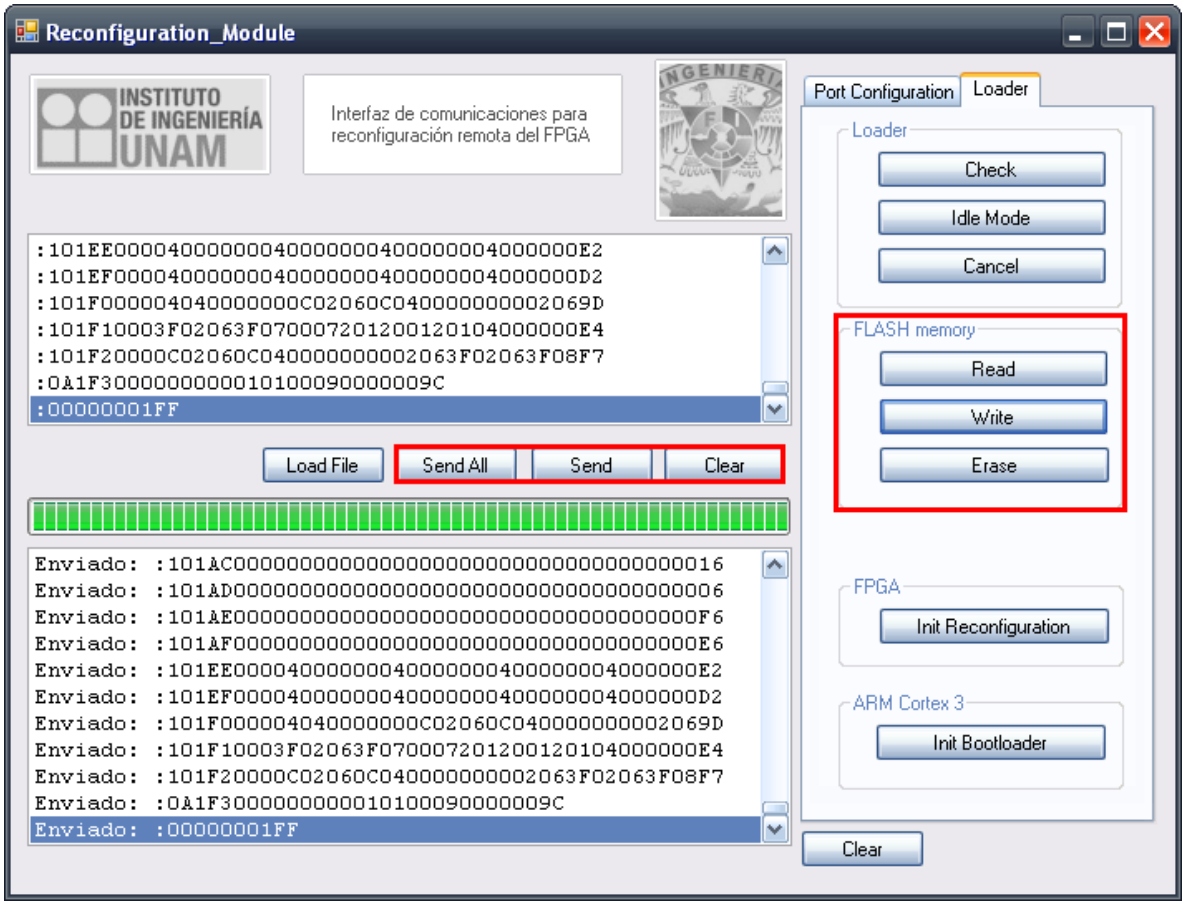


Figura 7.22. Transmisión de *bitsream*.

5. Finalmente se inicia la reconfiguración del FPGA presionando el botón *Init Reconfiguration* (ver FIGURA 7.23).

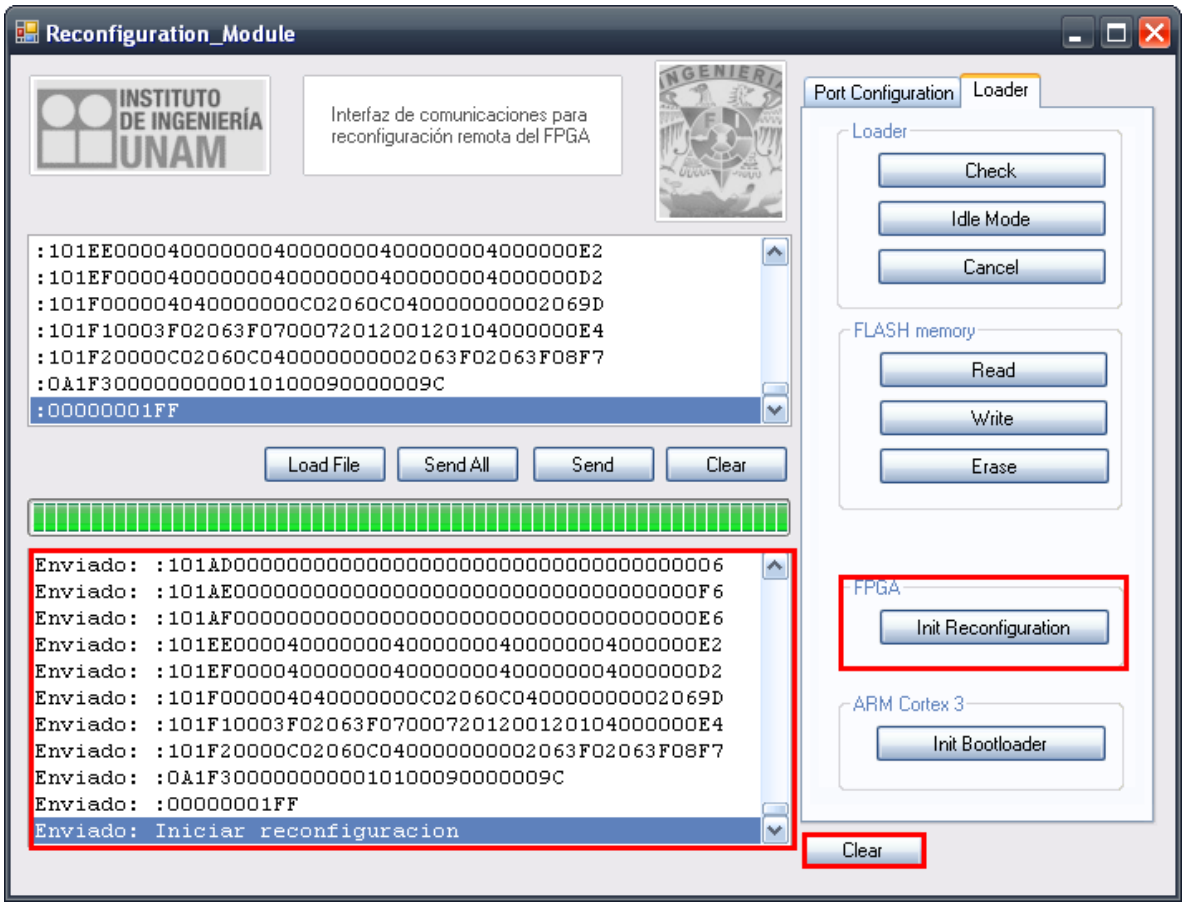


Figura 7.23. Reconfiguración de FPGA de SMIN_V2.

El botón *Clear* ubicado en la parte inferior limpia el campo de registro de datos transmitidos y recibidos.

El botón *Init Bootloader* se encuentra sin uso por el momento.

7.4 Resumen

Con los visto en este capítulo se posee el conocimiento suficiente para generar arquitecturas de cómputo, transmitirlas para ser cargadas en el FPGA e interactuar con el sistema.

8

Conclusiones, recomendaciones y trabajo futuro

En este capítulo se presentan las conclusiones obtenidas en el desarrollo de esta tesis, seguida de recomendaciones y trabajo futuro, con objeto de mejorar la calidad y el desempeño del sistema.

8.1 Conclusiones

De la tesis realizada se concluye lo siguiente:

- Se diseñó la tarjeta de circuito impreso (PCB) de la Plataforma FPGA Reconfigurableque integra y distribuye los nuevos componentes de forma óptima, tomando como base la experiencia adquirida durante el desarrollo del Sistema Mínimo SMIN_V1. Se incluyó un selector (*jumper*) y multiplexor, con el fin de, seleccionar de forma sencilla la fuente de señales JTAG que reconfigurará el FPGA. Ya que esta plataforma no sólo busca ser una tarjeta de soporte para implantar arquitecturas de cómputo polifórmicas en SATEDU, si no también ser usada como plataforma didáctica e independiente; se agregó un conector que recibe una fuente de alimentación externa. Aprovechando la gran cantidad de terminales de entrada/salida del nuevo FPGA, se contempló la inclusión de un puerto de propósito general con el fin de interconectar mayor cantidad de dispositivos externos con la tarjeta SMIN_V2.

- La integración del nuevo FPGA XC3S1600E, del microcontrolador PIC 18LF2520 y de la memoria FLASH SST25VF064C, permitieron adaptar niveles de alimentación y compatibilizar los rangos de operación entre los componentes electrónicos. Incluso, mejorar las características de la plataforma en términos de energía, en ampliación de recursos lógicos y en mayor capacidad para almacenar información. Con ello se asegura la implantación de algoritmos de determinación, estimación y control de orientación en el nuevo FPGA, y la posibilidad de almacenar de forma permanente en la nueva memoria FLASH un archivo de reconfiguración de respaldo, disponible en cualquier momento para reconfigurar el FPGA.

- Se ha desarrollado una Plataforma FPGA Reconfigurable para aplicación en el sistema educativo SATEDU. Sin embargo, este desarrollo tiene múltiples aplicaciones comerciales que cabe la pena explorar bajo nuevos proyectos.

8.2 Recomendaciones

- Dada la imperante necesidad de emplear componentes cada vez más complejos que mejoren la calidad de los diseños, se requiere de equipo de laboratorio con tecnología de vanguardia que permita manufacturar y ensamblar sistemas como el SMIN_V2. De esta forma, se podrá contribuir en la capacitación de ingenieros en el campo aeroespacial y contar con mejor infraestructura para reducir la brecha tecnológica entre México y países de vanguardia en el desarrollo de satélites. Actualmente en el Instituto de Ingeniería se están llevando a cabo procesos de compra de algunos equipos útiles para propósitos de manufactura.

- Se propone realizar pruebas de validación de forma sucesiva que garanticen la integración de los nuevos componentes. El software, tanto de interfaz (GUI) de Estación Terrena como ambos *firmwares*, que son necesarios para llevar a cabo la validación de cada prueba, requieren de mejoras en cuanto a la comunicación para la transmisión y almacenamiento del *bitstream*. Estos aspectos están contemplados como trabajo futuro y actualmente se está trabajando en ello. En cuanto a la conversión de archivos .xsvf a .hex se debe buscar incluir dicho proceso dentro de la interfaz de la ET. Aspecto en que también se está trabajando actualmente y particularmente en su integración.

- Para el desarrollo de aplicaciones más complejas, el uso de un Sistema Operativo en Tiempo Real (RTOS) que maneje las tareas a realizar y administre interrupciones de forma adecuada para garantizar el correcto comportamiento del sistema bajo ciertas restricciones de tiempo, es esencial. Existen en el mercado RTOS de licencia libre compatibles con el microncontrolador del SMIN_V2.

- Creación de Standard, referencia base que defina un conjunto de prácticas, roles, herramientas de computo, arquitectura de software, etc. para el desarrollo de aplicaciones y fabricación de proyectos de tecnología aeroespacial.

- Las PCs recientes disponen de mayor cantidad de puertos USB que de comunicación serial (RS232), de contemplar la sustitución de la comunicación serial en el sistema por comunicación USB, se actualizará el sistema y se incrementará la velocidad de transferencia de datos.

8.3 Trabajo Futuro

- Integrar y realizar pruebas de puesta a punto de la tarjeta de circuito impreso de SMIN_V2.

- Validación operativa de SMIN_V2, cargando arquitecturas sencillas en el FPGA y evaluar su funcionamiento con medios de depuración a bordo (leds e interruptores).

- Pruebas de operación con tarjeta de potencia.

- Integración y pruebas con tarjeta SMIN_V2 y tarjeta de potencia.

- Pruebas de compatibilidad con SATEDU.

- Con la tarjeta ensamblada, puesta a punto y validada operativamente, evaluar el desempeño de SATEDU con la nueva plataforma SMIN_V2 instrumentado sobre el

simulador satelital basado en una mesa suspendida en aire, cargando diferentes algoritmos de determinación, estimación y control de orientación (1 eje y tres ejes). El desempeño está ligado directamente a la realización de maniobras físicas de orientación y apuntamiento de la plataforma del simulador hacia objetivos específicos determinados en su periferia.

Bibliografía

[1] «http://satmex-garcialara.blogspot.mx/2012/12/satex-1.html,» [En línea].

[2] «http://gaceta.cicese.mx/ver.php?topico=seccion&ejemplar=64&id=359&from=buscador,» [En línea].

[3] I. Grout, Digital System Design with FPGAs and CPLDs, 2008.

[4] *xapp463,* 2005.

[5] G. P. Gokhale M, Reconfigurable Computing Accelerating Computation with Field Programmable Gate Arrays, 2000.

[6] S. H. a. A. DeHon, Reconfigurable Computing -The Theory and Practice of FPGA - Based Computation, 2008.

[7] *xapp467,* 2003.

[8] T. Noergaard, Embedded Systems Architecture, 2005.

[9] R. Dubey, Introduction to Embedded Systems Design Using Field Programmable Gate Arrays, 2009.

[10] A. S. Berger, Embedded Systems Design: An Introduction to Processes, Tools, and Techniques, 2002.

[11] «http://www.psoc-chile.es.tl,» [En línea].

[12] «http://www.louisreynier.com/fichiers/psoc.pdf,» [En línea].

[13] «http://www.cypress.com/?docID=3212,» [En línea].

[14] *XC3S1600E Data Sheet,* 2006.

[15] *SST25VF064C Flash memory datasheet,* 2011.

[16] *xapp053,* 2009.

[17] *xapp188,* 2008.

[18] *Spartan-3E FPGA Family: Complete Data Sheet,* 2006.

[19] «http://www.xilinx.com/itp/xilinx10/isehelp/pim_c_overview.htm,» [En línea].

[20] *Embedded GUIs,* 2008.

[21] *MPLAB IDE User's Guide,* EEUU, 2006.

[22] «http://www.microchip.com/Developmenttools/ProductDetails.aspx?PartNO=SW006011,» [En línea].

[23] «http://ww1.microchip.com/downloads/en/DeviceDoc/SST25VF064C.txt,» [En línea].

[24] R. S. Andrew G. Schmidt, Embedded Systems Design with Platform FPGA, 2010.

[25] *xapp462,* 2006.

[26] «http://www.rae.es/drae/srv/search?id=yHfxyKIWfDXX2kpDVRWA,» [En línea].

[27] «http://proyectos2.iingen.unam.mx/SATEDU/Default.htm,» [En línea].

[28] «http://www.iingen.unam.mx/es-mx/BancoDeInformacion/BancodeImagenes/Documents/SATEDU.pdf,» [En línea].