UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TEMA:

Análisis del desempeño de enlaces p2p utilizando la tarjeta FPGA De1 de Altera

AUTOR:

Bohórquez Zúñiga, David Isaac

Trabajo de Titulación previo a la obtención del título de

INGENIERO EN TELECOMUNICACIONES

TUTOR:

M. Sc. Palacios Meléndez, Edwin Fernando

Guayaquil, Ecuador

13 de septiembre del 2019

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por el Sr. **Bohórquez Zúñiga, David Isaac** como requerimiento para la obtención del título de **INGENIERO EN TELECOMUNICACIONES**.

TUTOR

_____

M. Sc. Palacios Meléndez, Edwin Fernando

DIRECTOR DE CARRERA

_____

M. Sc. Heras Sánchez, Miguel Armando

Guayaquil, 13 de septiembre del 2019

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Bohórquez Zúñiga David Isaac**

DECLARÓ QUE:

El trabajo de titulación: **"Análisis del desempeño de enlaces p2p utilizando la tarjeta FPGA De1 de Altera",** previo a la obtención del Título de **Ingeniero en Telecomunicaciones**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, 13 de septiembre del 2019

EL AUTOR

_____

Bohórquez Zúñiga, David Isaac

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **Bohórquez Zúñiga, David Isaac**

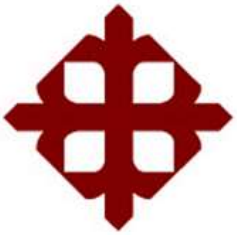
Autorizó a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del Trabajo de Titulación: **"Análisis del desempeño de enlaces p2p utilizando la tarjeta FPGA De1 de Altera"**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, 13 de septiembre del 2019

EL AUTOR

_____

Bohórquez Zúñiga, David Isaac



UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

f. _____

M. Sc. ROMERO PAZ, MANUEL DE JESÚS

DECANO

f. _____

M. Sc. HERAS SANCHEZ, MIGUEL ARMANDO

DIRECTOR DE CARRERA

f. _____

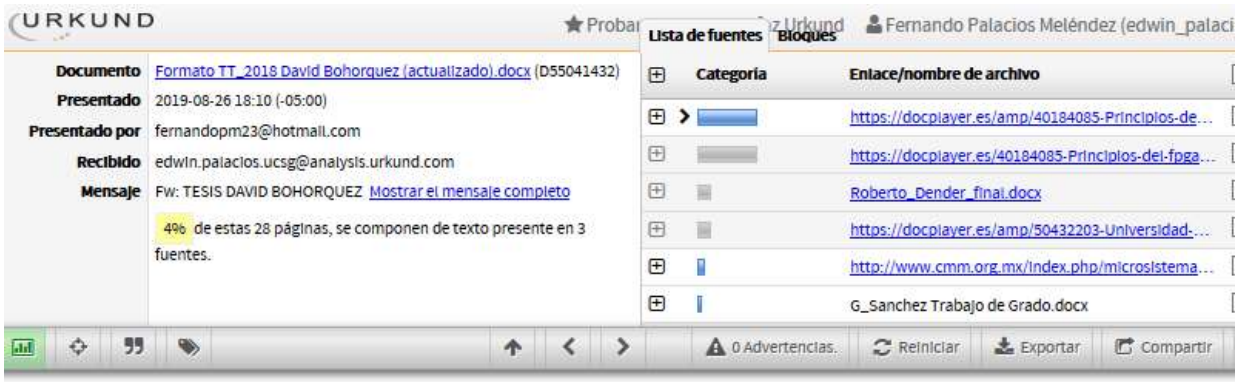
M. Sc. CORDOVA RIVADENEIRA, LUIS SILVIO

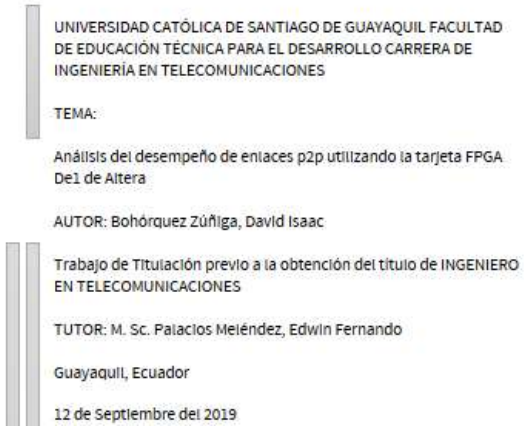
OPONENTE

REPORTE DE URKUND

REPORTE URKUND

Informe del Trabajo de Titulación de la Carrera de Ingeniería en Telecomunicaciones, con **4%** de coincidencias perteneciente al estudiante, **DAVID ISAAC BOHÓRQUEZ ZUÑIGA**.




UNIVERSIDAD CATÓLICA DE SANTIAGO DE GUAYAQUIL FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TEMA:

Análisis del desempeño de enlaces p2p utilizando la tarjeta FPGA De1 de Altera

AUTOR: Bohórquez Zúñiga, David Isaac

Trabajo de Titulación previo a la obtención del título de INGENIERO EN TELECOMUNICACIONES

TUTOR: M. Sc. Palacios Meléndez, Edwin Fernando

Guayaquil, Ecuador

12 de Septiembre del 2019

Atte.

M. Sc. Edwin Fernando Palacios Meléndez
TUTOR TRABAJO DE TITULACIÓN
DOCENTE TITULAR AUXILIAR – TIEMPO COMPLETO

DEDICATORIA

Dedico este trabajo principalmente a Dios, por haberme dado la vida y haberme permitido culminar mis estudios universitarios. A mis padres y mis abuelos por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por su total apoyo en todos estos años. Por sus consejos, comprensión, amor, ayuda en los momentos difíciles, y por ayudarme con los recursos necesarios para estudiar. Me han dado todo lo que soy como persona, mis valores, mis principios, mi carácter, mi empeño, mi perseverancia, mi coraje para conseguir mis objetivos.

Todo este trabajo ha sido posible gracias a ellos

EL AUTOR

_____

Bohórquez Zúñiga, David Isaac

AGRADECIMIENTO

Primeramente me gustaría agradecerte a ti Dios por bendecirme para llegar hasta donde he llegado, porque hiciste realidad este sueño anhelado. A mis padres por brindarme los recursos necesarios y apoyarme siempre para culminar mis estudios, por los valores que me han inculcado y por haberme dado la oportunidad de haber tenido una excelente educación en el transcurso de mi vida. A la Facultad Técnica para el desarrollo de la Universidad Católica de Santiago de Guayaquil por darme la oportunidad de estudiar y ser un profesional. A mi director de tesis, Ingeniero Fernando Palacios por su esfuerzo y dedicación, quien con sus conocimientos, su experiencia, su paciencia y su motivación ha logrado en mí que pueda terminar mis estudios con éxito. Me gustaría también agradecer a mis profesores durante toda mi carrera profesional porque todos han aportado a mi formación universitaria. Muchas gracias y que Dios los bendiga

EL AUTOR

_____

Bohórquez Zúñiga, David Isaac

Índice General

Índice de Figuras

Capítulo 2

Capítulo 3

Resumen

Para el presente documento se da a conocer en forma general el funcionamiento del análisis del desempeño p2p de la tarjeta FPGA DE1 de altera, Lo interesante del trabajo de titulación, fue la búsqueda de información debido a que no se ha manejado temas de titulación que involucren los análisis de desempeño, ni del manejo de la tarjeta FPGA DE1 de Altera, siendo esta muy completa con respecto a las tarjetas disponibles en el Laboratorio de Electrónica de la Carrera de Ingeniería en Telecomunicaciones de la Facultad de Educación Técnica para el Desarrollo.

En el Capítulo 1, se detallan la justificación, antecedentes y definición del problema de investigación, así como también, el objetivo general, objetivos específicos, idea a defender y la metodología empleada.

En el Capítulo 2, se analiza, describe y explora el Estado del Arte de la tarjeta FPGA DE1 de altera, para que el lector se familiarice con el tema.

En el Capítulo 3, se valida el trabajo de titulación mediante el análisis de desempeño p2p en la que el lector podrá hacer uso del mismo en el Laboratorio de Electrónica, donde va a reposar la tarjeta DE1 de altera.

En el Capítulo 4, se finaliza con las conclusiones y recomendaciones que se dan una vez concluida el trabajo de titulación.

Palabras claves: FPGA, P2P, ALTERA, QUARTUS, TARJETA, DE1.

Capítulo 1: Descripción General

1.1. Introducción.

La fusión teórica y la práctica para desarrollar aplicaciones prácticas de procesamientos digitales de señales son fundamentales en la formación de los estudiantes de la Carrera de Ingeniería en Telecomunicaciones, para lo cual este trabajo de componente práctico presenta un diseño básico para implementar en experimentos de sistemas digitales.

La utilización de los microprocesadores FPGAs, son muy útiles en la aplicación de la mayoría de los conocimientos impartidos por los docentes de nuestra facultad técnica para el desarrollo en la universidad católica de Santiago de Guayaquil, y desarrollar o gestionar estas prácticas es de gran importancia el saber sus fundamentos.

Es por eso por lo que se presenta en este trabajo de titulación un sistema fácil de realizar y accesible para los estudiantes e investigadores, esta placa entrenadora es sencilla con los elementos básicos para poder aplicar cualquier teoría o practica de sistemas digitales que es considerado muy importante por nuestros docentes. El aprendizaje de la utilización estos elementos se imparten en las clases de la materia, y determina la complementación de la formación.

1.2. Antecedentes.

Durante la búsqueda de información de trabajos relacionados con el tema del trabajo de titulación, se pudo constatar que en la carrera de ingeniería en telecomunicaciones no existen trabajos donde se analice o modele el desempeño de transmisión de enlaces p2p en las tarjetas FPGA DE1 de Altera.

1.3. Definición del Problema.

Necesidad de realizar el análisis del desempeño de transmisión de las tarjetas Fpga DE1 de altera utilizando comunicaciones punto a punto (P2P).

Ya que no son tratados a profundidad y tampoco hay trabajos de titulación donde se analice estas tarjetas, ni teóricamente ni a través de simuladores.

1.4. Justificación del Problema.

Con la integración de conocimientos de las tarjetas FPGA, se ha logrado tener pequeñas aplicaciones didácticas que mejoren el aprendizaje de los alumnos de la FETD en la UCSG. La Carrera de Ingeniería en Telecomunicaciones tiene disponible en su Laboratorio de Electrónica, tarjetas FPGA tanto de Altera (tarjeta DE0-nano) como Xlinx, pero que no han sido actualizados desde el 2007. La programación VHDL aprendida en Sistemas Digitales II y en Laboratorio de Digitales, permiten que podamos utilizar sin inconveniente la tarjeta DE1 de Altera.

1.5. Objetivos del Problema de Investigación.

1.5.1. Objetivo General.

Realizar el análisis de desempeño de enlaces p2p utilizando la plataforma de procesamiento digital Quartus II para la tarjeta FPGA DE1 de ALTERA.

1.5.2. Objetivos Específicos.

- Describir los funcionamientos teóricos y prácticos de los enlaces p2p utilizando las tarjetas Fpga De1 de altera

- Establecer comunicación entre 2 tarjetas Fpga De1 de altera.

- Conocer las ventajas del desempeño del enlace p2p de las tarjetas Fpga de1 de altera.

1.6. Hipótesis.

El trabajo de titulación propuesto, se enfoca en el análisis de desempeño p2p basado en la plataforma Quartus II lo que permitirá ver el comportamiento de la velocidad de la tarjeta FPGA DE1 de altera con un red p2p.

1.7. Metodología de Investigación.

El diseño de investigación, es del tipo exploratorio que trata de examinar un tema o problema de investigación poco abordado, del cual todavía generan dudas. Es decir que solo hay guías no investigadas y vagamente relacionadas con respecto al problema de investigación. También es explicativo, porque se pretende establecer las causas del evento estudiado.

Capítulo 2: Fundamentación Teórica

2.1. Breve Reseña.

En primer lugar conviene aclarar que P2P es el acrónimo de peer-to-peer. Se trata de una red de ordenadores que reciben el nombre de "nodos", que se conectan para compartir datos de todo tipo y en la que cada máquina puede actuar como cliente o servidor. Esto choca frontalmente con la arquitectura cliente-servidor adoptada en la mayoría de las infraestructuras informáticas corporativas, en las que un ordenador central (el servidor) cumple las demandas de otros adheridos a su red (los clientes). (Argudo S., 2016)

Además, este ordenador central es quien determina qué privilegios tienen sus clientes con respecto al acceso a recursos de la red e información disponible en la misma. En una red P2P este sistema de privilegios desaparece. En una aplicación que haga uso de las redes P2P cada máquina tiene los mismos privilegios que su vecina, esté físicamente próxima (por ejemplo, otro ordenador en la red doméstica) o no (como serían dos máquinas que se comparten información entre sí separadas por miles de kilómetros de distancia). Cada nodo o peer representan una fracción de los recursos de la red.

La capacidad de la máquina aporta potencia de proceso, almacenamiento o ancho de banda de red a otras máquinas participantes, sin que haya un ordenador central que lo coordine todo. Dichos recursos se consumen y se proveen por igual por cada una de las máquinas en función de su potencia. En otras palabras, la red P2P tradicional funciona interconectando varios equipos entre sí a través de un hub o concentrador, lo que permite que cada uno de los usuarios de dicha red pueda obtener la información que necesite de cualquier otro miembro.

2.2. Principio de funcionamiento: Todos los nodos de la red p2p.

Este sistema evita la opacidad de la red debido a los cortafuegos o las funciones de traducción dinámica de direcciones IP (NAT). Con respecto a la

lista de archivos o documentos compartidos, corresponde a cada cliente, según su grupo de trabajo, transmitir a los demás la información correspondiente. Estas funciones de traducción de direcciones pueden ser proporcionadas en un número de casos por un servidor central. Pero, en cualquier caso, todos los pares se reconocen entre sí "escuchando" el tráfico en un puerto TCP que es específico para ellos. Interrogarse entre sí de la misma manera que los Servidores de Nombres (DNS) para determinar la dirección IP correspondiente al nombre virtual deseado. (Alcaine Sanchez, 2015)

Este sistema evita la opacidad de la red debido a los cortafuegos o las funciones de traducción dinámica de direcciones IP (NAT). Con respecto a la lista de archivos o documentos compartidos, corresponde a cada cliente, según su grupo de trabajo, transmitir a los demás la información correspondiente. Estas funciones de traducción de direcciones pueden ser proporcionadas en un número de casos por un servidor central.

Pero, en cualquier caso, todos los pares se reconocen entre sí "escuchando" el tráfico en un puerto TCP que es específico para ellos. Dirección IP correspondiente al nombre virtual deseado. Este sistema evita la opacidad de la red debido a los cortafuegos o las funciones de traducción dinámica de direcciones IP (NAT). Con respecto a la lista de archivos o documentos compartidos, corresponde a cada cliente, según su grupo de trabajo, transmitir a los demás la información correspondiente.

Estas funciones de traducción de direcciones pueden ser proporcionadas en un número de casos por un servidor central. Pero, en cualquier caso, todos los pares se reconocen entre sí "escuchando" el tráfico en un puerto TCP que es específico para ellos. Dirección IP correspondiente al nombre virtual deseado. Este sistema evita la opacidad de la red debido a los cortafuegos o las funciones de traducción dinámica de direcciones IP (NAT).

Con respecto a la lista de archivos o documentos compartidos, corresponde a cada cliente, según su grupo de trabajo, transmitir a los demás la información correspondiente. Estas funciones de traducción de direcciones pueden ser proporcionadas en un número de casos por un servidor central. Pero, en cualquier caso, todos los pares se reconocen entre sí "escuchando" el tráfico en un puerto TCP que es específico para ellos. Con respecto a la lista de archivos o documentos compartidos, corresponde a cada cliente, según su grupo de trabajo, transmitir a los demás la información correspondiente. Estas funciones de traducción de direcciones pueden ser proporcionadas en un número de casos por un servidor central. Pero, en cualquier caso, todos los pares se reconocen entre sí "escuchando" el tráfico en un puerto TCP que es específico para ellos.

Con respecto a la lista de archivos o documentos compartidos, corresponde a cada cliente, según su grupo de trabajo, transmitir a los demás la información correspondiente. Estas funciones de traducción de direcciones pueden ser proporcionadas en un número de casos por un servidor central. Pero, en cualquier caso, todos los pares se reconocen entre sí "escuchando" el tráfico en un puerto TCP que es específico para ellos.(Alcaine Sanchez, 2015)

2.3. Las Características de p2p.

En esta sección se presentan las características principales del modelo de igual a igual:

- Descentralización: el hecho de que cada nodo maneje sus propios recursos evita la centralización del control. Un sistema P2P puede operar sin necesidad de una administración centralizada, lo que evita los cuellos de botella y aumenta la resistencia del sistema a fallas y fallas. (Chicaiza García, Diego Francisco, 2017)

- Escalado: implica la cooperación de una gran cantidad de nodos (hasta miles o millones) para compartir recursos mientras se mantiene un buen rendimiento del sistema. Esto significa que un sistema P2P

debe ofrecer métodos bien adaptados con un entorno en el que hay que compartir un gran volumen de datos, una gran cantidad de mensajes que se intercambiarán entre ellos. Una gran cantidad de nodos que comparten sus recursos a través de una red ampliamente distribuida.

- Auto-organización: como los sistemas P2P a menudo se implementan en Internet, la participación de un nuevo nodo en un sistema P2P no requiere una infraestructura costosa. Solo tiene un punto de acceso a Internet y sepa que otro nodo ya está conectado para conectarse al sistema. Un sistema P2P debe ser un entorno abierto; es decir, un usuario en un nodo debe poder conectar su nodo al sistema sin la necesidad de contactar a una persona y sin tener que pasar por una autoridad central.

- Autonomía del nodo: cada nodo gestiona sus recursos de forma autónoma. El decide qué parte de sus datos compartir Se puede conectar o desconectar en cualquier momento. También tiene la autonomía para administrar su poder de computación y su capacidad de almacenamiento.

- Heterogeneidad: debido a la autonomía de los nodos con hardware y/o arquitecturas de software, los sistemas P2P deben tener Desarrollar técnicas apropiadas para resolver problemas relacionados con la heterogeneidad de los recursos.

- Dinámico: debido a la autonomía de los nodos, cada nodo puede abandonar el sistema en cualquier momento, lo que hace que sus recursos desaparezcan del sistema. Se pueden agregar nuevos recursos al sistema cuando se conectan nuevos nodos. Luego, debido a la inestabilidad del nodo, los sistemas P 2P deben ser capaces de manejar una gran cantidad de recursos altamente variables. La salida de un nodo del sistema (o la falla de un nodo) no debe deshabilitar el

sistema. Debe tolerarse y tener un impacto "pequeño" en el rendimiento de todo el sistema.

2.4. Clasificación y comparación de las redes P2P

Se pueden identificar tres categorías amplias de sistemas P2P, como se muestra en la Figura 1.1: centralizada, descentralizada e híbrida. La categoría de descentralizado se puede dividir en una estructura descentralizada y desestructurada descentralizada. La principal diferencia entre estos sistemas es el mecanismo utilizado para buscar recursos en la red P2P.

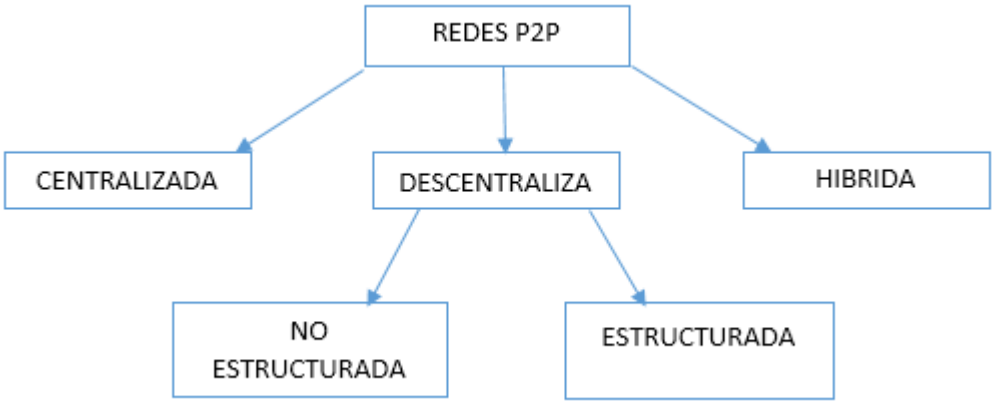


Figura 2. 1: Clasificación p2p
Elaborado por: Autor.

2.4.1. Arquitectura Centralizada

Este tipo de arquitectura presenta la primera generación de redes P2P. Se basa en un servidor central que contiene cierta información sobre los recursos (especialmente archivos) compartidos por los participantes de la red. Los usuarios se conectan al servidor para comunicarse entre ellos e intercambiar archivos.(Medrano L., 2015)

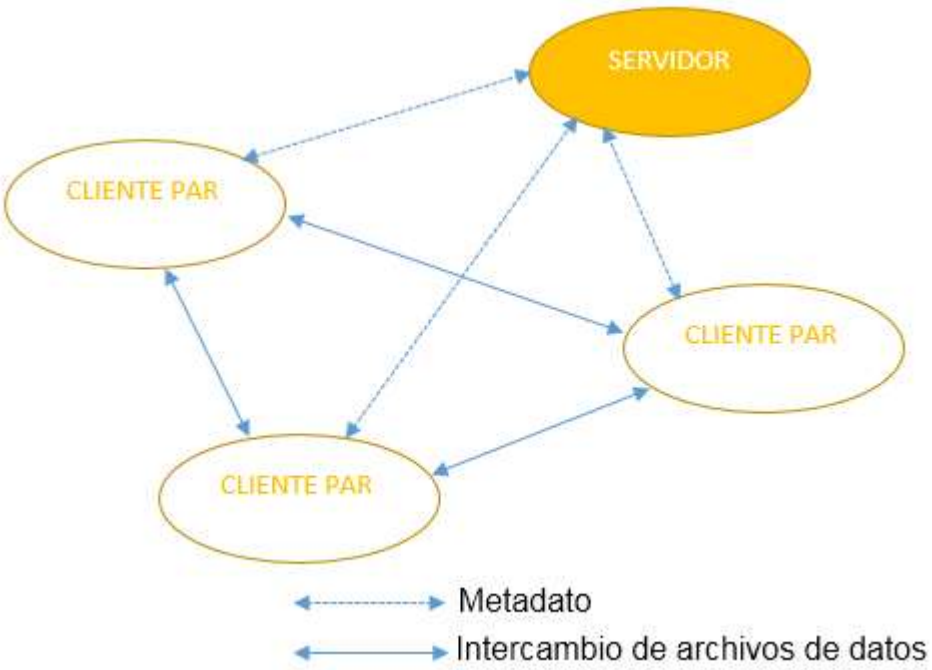


Figura 2. 2: Arquitectura Centralizada
Elaborada por: Autor.

La función principal del servidor es la gestión de la búsqueda de recursos mediante la identificación de los nodos que almacenan los archivos. Después de la conexión con el servidor y el descubrimiento del nodo que posee el archivo deseado, la comunicación y el intercambio de este archivo, entre el par solicitante y el proveedor par, se realiza directamente. La figura 2.2 muestra esta arquitectura.

2.4.2. Arquitectura Descentralizada

Esta arquitectura es de naturaleza plana donde todos los nodos de la red cumplen las mismas tareas. Ellos juegan el papel de servidor y clientes al mismo tiempo. Los pares de dicha red a menudo se llaman SERVIDOR (SERVIDOR + CLIENTE). Esta arquitectura no tiene un servidor central, porque un par deja la red no afecta el sistema. Se puede designar dos tipos de arquitecturas descentralizadas: desestructuradas y estructuradas.(Arias, P., 2015)

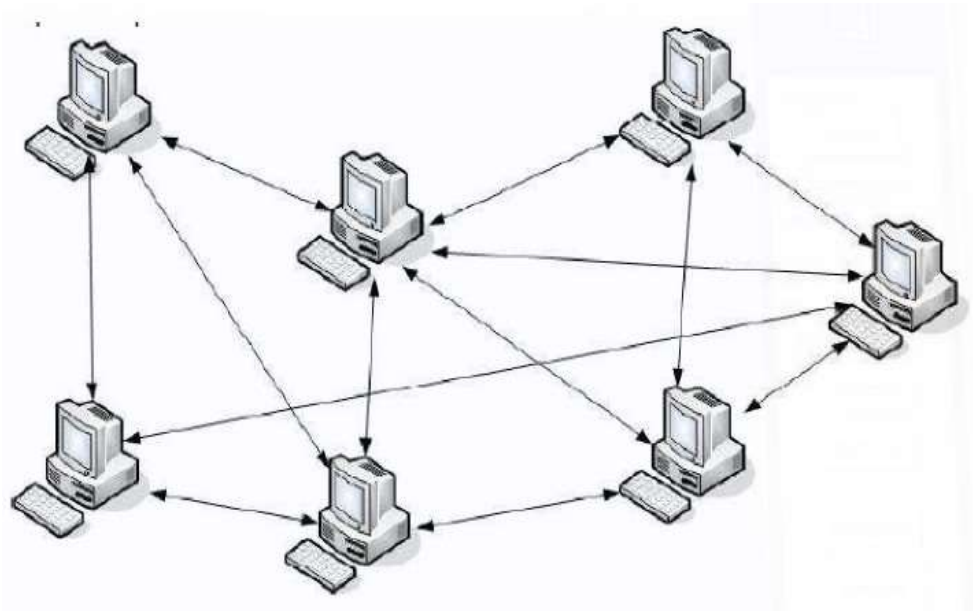


Figura 2. 3: Arquitectura descentralizada
Elaborada por: Autor.

2.4.2.1. Arquitectura no estructurada

Esta clasificación se basa en la noción de inundación que cuando un usuario puede descubrir un recurso, transmite una solicitud que contiene el nombre del recurso a los otros vecinos hasta llegar al cliente que tiene acceso al recurso. Para evitar la inundación de la red durante mucho tiempo, el sistema asocia a cada solicitud un temporizador TTL "Time To Live", el valor asignado a la red. TTL usualmente es 7. Cuando llega a cero, la solicitud ya no se devuelve.

La principal desventaja de este mecanismo es que TTL = 0 antes de la ruta completa de la red, lo que puede llevar a la falla de una búsqueda aunque el artículo deseado está disponible en la red P2P.

Estas arquitecturas no estructuradas son extremadamente resistentes a los nodos que entran y salen del sistema. Por otro lado, el mecanismo de investigación actual va mal en la escala y genera una carga significativa para los participantes en la red. Este método se usa en el sistema: Gnutella y FreeNet.

Aquí se explica un ejemplo de una red descentralizada P2P no estructurada, Gnutella.

Gnutella

Gnutella fue la primera red P2P totalmente descentralizada. Creado en marzo de 2000 por Justin Frankelet Tom Pepper, un sucesor de Napster, cuya centralización estaba cargada de debilidad, Gnutella se benefició de esta experiencia. Gnutella es un protocolo abierto y descentralizado para búsquedas distribuidas en una topología plana de pares.

Según Gnutella, todos los pares son a la vez servidor y cliente. Este protocolo no tiene un directorio centralizado y no tiene control sobre la topología o ubicación de los archivos. La red se forma con pares que se unen a la red de acuerdo con algunas reglas simples. La ubicación de los datos no se basa en ningún conocimiento de la topología. Para ubicar un objeto, un cliente pregunta a sus vecinos quienes, ellos mismos, preguntan a sus vecinos. Este sistema simplemente permite la entrada y salida de los clientes, pero el mecanismo utilizado va mal en la báscula y genera cargas pesadas en la red.(Villada L., 2016)

2.4.2.2. Arquitectura Estructurada

Los sistemas P2P estructurados emplean un algoritmo de búsqueda. Este algoritmo es completamente determinista, y los enlaces entre pares se establecen de acuerdo con reglas bien definidas.

La infraestructura de búsqueda funciona como una Tabla Hash Distribuida (DHT), implementada punto a punto. Esta estructura permite el descubrimiento eficiente de datos mediante claves. Es particularmente apropiado para el desarrollo de redes a gran escala. En esta categoría, uno puede colocar Chord, CAN (Red direccionable de contenido).(Nicolini A., 2018)

Una tabla hash distribuida (DHT) primero hash el nombre del archivo buscado. Para este propósito, el DHT utiliza una función hash estándar, como SHA-1 o MD5, para garantizar que cada nombre de archivo coincida con su huella como una cadena de bits de longitud fija. Esta función hash garantiza que, para dos recursos diferentes, las claves generadas sean únicas. Por lo tanto, la clave puede identificar y encontrar el recurso de manera confiable.

Segundo hash, la dirección IP da el identificador de un usuario. Se trata entonces de almacenar de forma distribuida las parejas (clave, identificador) en los nodos de la red para que cada recurso en la red está asociado con la dirección del usuario que posee el recurso. La redundancia en el almacenamiento también se introduce de modo que el inicio de un nodo del sistema no causa la pérdida de los metadatos que almacena y hace imposible el acceso a estos datos.

Los DHT tienen buenas propiedades debido al uso de un modelo P 2P puro: ningún compañero juega un papel particular o central, y todos actúan de una manera estrictamente equivalente.

Entre las propiedades de DHT se encuentran los siguientes:

- Rendimiento: en condiciones de funcionamiento normales, la cantidad de saltos requeridos es limitada. Por ejemplo, en una comunidad de 106 pares que usan una base de identificador hexadecimal, la longitud promedio de una consulta es de aproximadamente cinco saltos.

- Escalado ('escalabilidad'): dos características dan a los DHT un buen rendimiento de escalado. El primero está relacionado con el número promedio de saltos requeridos para el enrutamiento de aplicaciones, que sigue siendo pequeño, incluso en comunidades con un gran número de participantes. El segundo está relacionado con las tablas de enrutamiento, que también tienen un tamaño razonable en comparación con el número de participantes.

- Confiabilidad**:** el uso de un algoritmo de descubrimiento y enrutamiento permite, para una clave determinada, determinar la ID de par más cercana. En condiciones estáticas, una respuesta sin consulta significa que el recurso requerido no está disponible en la comunidad.

- Tolerancia a las fallas**:** Debido a la falta de centralización, que excluye cualquier punto central, las DHT muestran buena tolerancia a las eliminaciones de nudos al azar. Las consultas se pueden reenviar incluso si algunos de los nodos desaparecen. Por otro lado, cada nodo raíz de un recurso particular es similar a un punto central. Los mecanismos de redundancia a menudo se ponen en marcha para evitar la inaccesibilidad de un recurso presente en un DHT. Un ejemplo de una red estructurada descentralizada P2P, Chord.

- Chord: Chord organiza su espacio de direcciones siguiendo un anillo cuyas 2matrices (o identificadores, id) se ordenan a lo largo de su circunferencia. Cada par, así como cada recurso, tiene un identificador obtenido por una función hash SHA-1 (en este caso m = 160 bit) que garantiza una distribución homogénea de recursos en el anillo de acordes. Chord se limita a una función de enrutamiento, es decir, dado un identificador, Chord localiza al par responsable que es el que tiene el identificador más pequeño mayor o igual que el del de recursos. Cuando un compañero se une a la red, es compatible con algunas de las credenciales asignadas a su sucesor directo, y cuando abandona la red, todos sus identificadores se atribuyen a su sucesor. Por lo tanto, Chord no ofrece todas las primitivas de almacenamiento / búsqueda / replicación para operar un DHT, y mucho menos una aplicación funcional para compartir archivos. El interés de Chord es proporcionar una estructura simple y eficiente para enrutar mensajes. Para esto, cada par de la red mantiene una tabla de punteros que contiene la información id, ip, port para un cierto número de pares. Cuanto más grande es esta lista, más costoso es mantenerla, pero cuanto más rápida sea la ruta (el caso extremo es conocer a todos los pares de la red, el enrutamiento solo necesita un mensaje). Chord logra encontrar un compromiso entre el tamaño de la tabla de enrutamiento con O (log (N)) ingresado y el enrutamiento requirió mensajes O (log (N)). Para ello, un par, de la identificación CurrentID elige cada puntero estable de tal manera que el par es el único elige

el intervalo [(CurrentID + 2i) - (+ 2i CurrentID +1)] con i + 1 <m. Por lo tanto, un compañero conoce un sucesor que representa una porción del anillo que es el doble de grande para cada entrada nueva. El enrutamiento es iterativo, paso a paso, y cada uno de los pares envía la solicitud al sucesor conocido más cercano al identificador buscado. Los principales límites de Chord son, por un lado, su topología de anillo que hace necesario recorrer todo el espacio de direcciones en el peor de los casos, y por otro lado, la ausencia de primitivas para operar un DHT. . Estas debilidades son resueltas por la arquitectura Kademlia.(Manchado S., 2016)

2.4.3. Arquitectura Hibrida

En la práctica, los nodos de una red P2P tienen capacidades de almacenamiento y procesamiento caracterizadas por una gran disparidad que aprueba la creación del modelo híbrido que es un acoplamiento entre Modo P2P y arquitectura Cliente / Servidor.(Anzures, M., 2017)

Las redes híbridas utilizan un número suficiente de servidores (los llamados super-pares) para evitar el riesgo en el caso de la desaparición de uno de ellos. Cada servidor tiene un nodo central de un grupo (clúster) que incluye un conjunto de pares organizados en P2P.

La siguiente figura muestra claramente la arquitectura parcialmente híbrida:

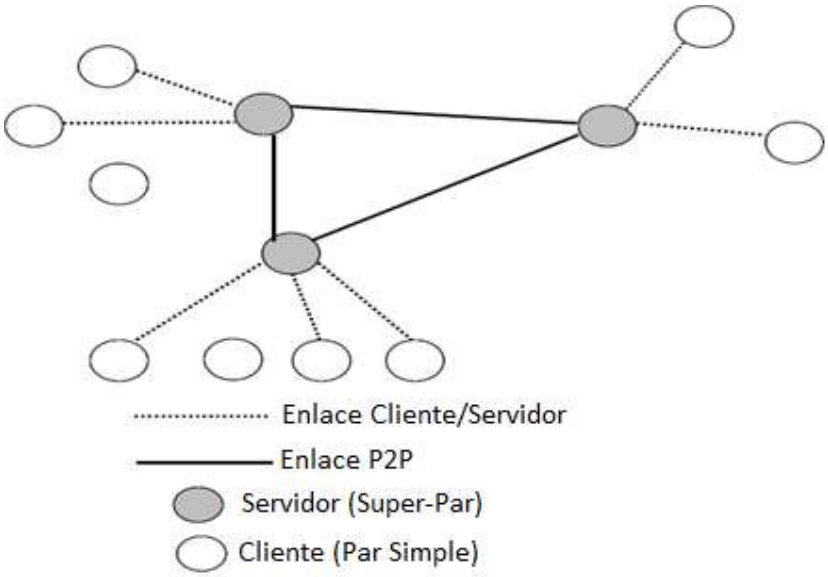


Figura 2. 4: Arquitectura Hibrida.
Elaborada por: Autor.

Se distinguen dos tipos de redes híbridas:

- **Híbridos estáticos**: los superpares se designan manualmente desde el principio. Este último puede desempeñar el papel del cliente al mismo tiempo, como se puede ver en el ejemplo de la red eDonkey.

- **Híbridos dinámicos**: bajo ciertas condiciones, el software del cliente decide transformar un nodo cliente en un súper-par, como en el ejemplo de Kazaa.

2.4.4. Estudio comparativo de las arquitecturas p2p

El siguiente cuadro muestra un estudio exhaustivo de la arquitectura P2P.

- **Red punto-a-punto centralizada**

  Ventajas:

  - Acceso a los recursos
  - Búsquedas evaluadas
  - Carga mínima en pares

Desventajas:

- Disponibilidad del servidor
- Costo del servidor
- Confianza en la administración del servidor

- **Red de punto a punto de estructura descentralizada**

  Ventajas:

  - Distribución completa.
  - Escala de paso/Carga homogénea.
  - Almacenamiento de datos DTH.

  Desventajas:

  - Solo búsquedas exactas.
  - Estructura a mantener.

- **Red punto a punto no estructurada y localizada**

  Ventajas:

  - Distribución completa.
  - Descentralización.
  - Búsquedas no exactas.

  Desventajas:

  - Carga de escala limitada (inundación).
  - Homologo no homogéneo (Super pares).

2.5. Ventajas y desventajas de P2P

Se puede resumir las diferentes ventajas y desventajas de los sistemas P2P de la siguiente manera:

2.5.1. Ventajas

A partir de las propiedades citadas anteriormente, se puede decir que los sistemas P2P son más adecuados para entornos con grandes cantidades de recursos para compartir.

- Evitar la creación de cuellos de botella.

- Sube por la escalera.

- Aprovechar recursos significativos distribuidos en una gran cantidad de nodos en la red.

- Mejore el uso del ancho de banda, los recursos de procesamiento y el espacio de almacenamiento.

- Acelere la finalización de la tarea reduciendo el tiempo de procesamiento a través de enlaces directos entre pares.

- Evite el único punto de falla gracias a la distribución redundante de recursos y la descentralización de los sistemas P2P. Esta característica ofrece estos tipos de sistemas más confiabilidad y dureza.

- Aumentar el rendimiento del sistema al compartir la carga de trabajo y aumentar la autonomía y la agregación de recursos, lo que aumenta el rendimiento de las redes P2P.

- Permite a los usuarios mantener el control de sus recursos. Además, pueden unirse o abandonar el sistema en cualquier momento.

2.5.2. Desventajas

A pesar de que las redes P2P son muy populares y muy útiles hoy en día, pero tienen desventajas, se citan algunas de ellas:

- **Problemas de comportamiento del usuario**: los sistemas de intercambio de archivos punto a punto pueden sufrir fácilmente de una anarquía general. El anonimato ayuda, los miembros a veces se sienten tentados a tener un comportamiento malicioso. Sin un mínimo de control, se pueden ver actividades como propagación de virus o trabajo independiente.

- **Problemas de comportamiento empresarial**:
  Una violación de la privacidad: Nuestra privacidad no está necesariamente protegida cuando se utilizan herramientas de igual a igual de hecho, las direcciones IP de los usuarios se pueden recuperar cuando el software está centralizado.

  La contaminación de las redes: Sin siquiera anunciar un ultimátum, varias empresas proponen contaminar las redes de intercambio de música entre individuos mediante la incorporación de archivos de menor calidad o incorrectos. El objetivo es hacer que estas redes libres y anárquicas sean menos atractivas.

Un puñado de compañías independientes está desarrollando nuevas tecnologías para contaminar las redes. Sus clientes potenciales son las industrias de la música y el cine, que están tratando de evitar los descargadores gratuitos en todas las formas para beneficiar a los sistemas seguros y rentables.

- **Propaganda**

Ya no se ignora que los sistemas peer-to-peer están invadidos en su mayoría por anuncios publicitarios, aunque hoy en día las versiones del software se ofrecen sin publicidad.

- **Las redes punto a punto permanecen imperfectas debido a**:

• Confianza y certificado

- El anonimato
- La seguridad
- La actuación

2.6. Alcance de las redes P2P

Desde su aparición, las redes P2P se consideran como un sistema de intercambio de archivos. Hasta el momento, esta aplicación tiene la tasa de tráfico más alta de las diferentes redes P2P. Es por esta razón que el término P2P a veces se puede confundir con los sistemas de intercambio de archivos, mientras que el último representa solo un caso de los tipos de aplicación basados en el paradigma P2P.

Los sistemas P2P se utilizan en varias categorías de aplicaciones que se pueden dividir en cuatro clases: computación distribuida, difusión de contenido, programas de mensajería, Streaming P2P.(Santos S., 2017)

2.6.1. Calculo distribuido

Incluso si los recursos de una computadora son limitados, la agrupación de muchas computadoras permite obtener representaciones teóricamente considerables. Este es uno de los objetivos de los sistemas P2P, que está ampliamente ilustrado en la literatura por el ejemplo de los matemáticos (y físicos) con sus cálculos matriciales. El objetivo es dividir un gran cálculo, por ejemplo, el de la inversión de la matriz en pequeños tratamientos más o menos independientes que uno podría dividir entre los pares. Este tipo de aplicación se conoce como GRID Computing.

2.6.2. Divulgación de contenido

El propósito de este tipo de aplicación suele ser compartir archivos, que es el objetivo principal para el cual se introdujeron los sistemas P2P. Estos sistemas se utilizan para crear una red de nodos para buscar y transferir archivos. En esta categoría se encuentran sistemas como: Napster, Gnutella, KAZAA, Freenet.

El uso compartido de archivos en una red P2P suele estar implícito en los casos en que los archivos deben transmitirse a un gran número de

personas (por ejemplo, transmitiendo una transmisión de video).o audio, la aplicación P2P más adecuada para este tipo de aplicación es BitTorrent.

Una segunda característica que permite la entrega de contenido es la de almacenar y publicar contenido. El objetivo de estos sistemas es crear un espacio distribuido que permita el almacenamiento y publicación de datos. Estos serán utilizados por los compañeros que tienen el privilegio de acceso. Ejemplos de OceanStore e Ivy se destacan en esta categoría.

2.6.3. Programas de mensajería

En la actualidad, existen servicios de correo electrónico basados en P2P, los usuarios pueden enviar y recibir correos electrónicos de forma segura, sin necesidad de un servidor central. Para almacenar temporalmente los mensajes que garantizan la confidencialidad de los corresponsales, se utiliza un sistema de autenticación y encriptación para proteger sus contenidos; un buen ejemplo es Jeftel.com. El software de mensajería instantánea también se puede ver como P 2P, los ejemplos incluyen ICQ, AIM que sin duda utiliza un servidor, pero solo para la resolución de direcciones (las direcciones utilizadas pueden ser alternativas a las direcciones). IP.

2.6.4. Streaming P2P

Streaming P2P es el hecho de ver transmisiones en vivo producidas y / o retransmitidas por otros compañeros de la red para evitar o al menos reducir la congestión que podría ocurrir en los servidores de descarga, todo sucede. Para las personas que quieran acceder al archivo, Swarmplayer, que le permite reproducir videos en streaming utilizando BitTorrent, promete ser una verdadera revolución en el campo.

2.7. FPGA

En el nivel más alto, un FPGA es un circuito de silicio reprogramable. Al usar bloques lógicos preconstruidos y recursos de enrutamiento programables, puede configurar este circuito para implementar funciones de hardware personalizadas, sin tener que usar un modelo o soldador. Simplemente desarrolle tareas de procesamiento digital basadas en software y compílelas como un archivo de configuración o un flujo de bits que contenga

información sobre cómo se deben conectar los componentes. Además, los FPGA son completamente reconfigurables y pueden adoptar instantáneamente una nueva "personalidad" si recompila una nueva configuración de circuito. Hasta ahora, solo los ingenieros con experiencia en diseño de hardware digital podían usar la tecnología FPGA. Sin embargo, la generalización de las herramientas de diseño de alto nivel está cambiando las reglas de programación de FPGA, con nuevas tecnologías para convertir diagramas gráficos o incluso código C ANSI en circuitos de hardware digital.

Si los FPGA tienen tanto éxito en cualquier industria, es porque reúnen lo mejor de los ASIC y los sistemas basados en procesadores. Por lo tanto, ofrecen sincronización de hardware que garantiza velocidad y confiabilidad, pero son más rentables que los ASIC personalizados. Los circuitos reprogramables también tienen la misma flexibilidad de ejecución de software que un sistema basado en procesador, pero no están limitados por la cantidad de núcleos de proceso disponibles. A diferencia de los procesadores, los FPGA son realmente paralelos por naturaleza, por lo que muchas operaciones de procesamiento diferentes no compiten por la utilización de recursos. Cada tarea de procesamiento independiente se asigna a una sección específica del circuito y, por lo tanto, puede ejecutarse de forma autónoma sin depender de ningún modo de otros bloques lógicos. Como resultado, puede aumentar la cantidad de procesamiento realizado sin afectar el rendimiento de parte de la aplicación.(Yuquilema H., 2014)

2.7.1. Las cinco principales fortalezas de la tecnología FPGA

- Rendimiento
- Tiempo de lanzamiento al mercado
- Costo
- Fiabilidad
- Mantenimiento a largo plazo

Rendimiento: a medida que aprovechan el paralelismo de hardware, los FPGA ofrecen mayor potencia de cómputo que los procesadores de señal digital (DSP) porque no necesitan el modelo de ejecución secuencial y

realizan más operaciones por ciclo. Reloj. BDTI, una empresa líder en análisis y evaluación comparativa, ha publicado estudios que demuestran que los FPGA pueden ofrecer potencia de procesamiento por dólar varias veces mayor que la de una solución DSP en algunas aplicaciones2. El control de las entradas y salidas (E / S) en el nivel de hardware proporciona tiempos de respuesta más cortos, así como características específicas que satisfacen mejor las necesidades de la aplicación.

Tiempo de lanzamiento al mercado: con el aumento de los tiempos de llegada al mercado, la tecnología FPGA proporciona una solución flexible con capacidades de prototipo rápido. Entonces puede probar una idea o concepto y luego verificarlo en hardware sin pasar por el largo proceso de crear un ASIC3 personalizado. Luego, puede realizar las modificaciones necesarias en su FPGA en unas pocas horas en lugar de unas pocas semanas. El hardware listo para usar actualmente en el mercado también ofrece diferentes tipos de E / S ya conectados a un chip FPGA programable por el usuario. La multiplicación de herramientas de software de alto nivel disponibles en el mercado permite reducir el tiempo de aprendizaje con las capas de abstracción. Estas herramientas a menudo incluyen núcleos de propiedad intelectual (funciones precompiladas) útiles para el control avanzado y el procesamiento de señales.

Costo: los costos de ingeniería no recurrentes (NRE) de los ASIC personalizados son mucho más altos que los de las soluciones de hardware basadas en FPGA. La gran inversión inicial que requieren los ASIC está ampliamente justificada para los OEM, por ejemplo, que pueden entregar miles de circuitos por año. Sin embargo, la mayoría de los usuarios finales necesitan hardware personalizado para algunas decenas o cientos de sistemas en desarrollo. Por naturaleza, los circuitos programables no implican costos de fabricación o largos tiempos de montaje. Las necesidades de la mayoría de los sistemas cambian con el tiempo; sin embargo, la modificación gradual de un FPGA representa un costo insignificante en comparación con el gasto considerable requerido para volver a designar un ASIC.

Fiabilidad: mientras que las herramientas de software proporcionan el entorno de programación, los chips FPGA son una verdadera implementación de hardware para la ejecución de software. Los sistemas basados en procesadores a menudo incluyen varias capas de abstracción para ayudar con la planificación de tareas y la asignación de recursos entre diferentes procesos. La capa de controladores controla los recursos de hardware y el sistema operativo administra la memoria del procesador y el ancho de banda. En cada núcleo del procesador, solo se puede ejecutar una instrucción a la vez; por lo tanto, los sistemas basados en procesador aún pueden tener tareas prioritarias conflictivas. Los FPGA, que no usan un sistema operativo, minimizan los problemas de confiabilidad porque proporcionan una ejecución verdaderamente paralela y hardware determinista para cada tarea.

Mantenimiento a largo plazo: como se puede apreciar, los chips FPGA son escalables y le ahorran el gasto de tiempo y dinero que implica el rediseño de los ASIC. Las especificaciones de los protocolos de comunicación digital, por ejemplo, cambian con el tiempo. Pero las interfaces basadas en ASIC pueden causar problemas de mantenimiento y compatibilidad. Como son reconfigurables, los circuitos de FPGA pueden adaptarse a las modificaciones que puedan ser necesarias. A medida que un producto o sistema evoluciona, puede incorporar mejoras funcionales sin perder tiempo tratando de rediseñar el hardware o cambiar el diseño del circuito.

2.8. Altera DE1 Soc.

El Kit de desarrollo DE1 SoC fabricado por Terasic y potenciado por ALTERA Cyclone V SoC es una plataforma de diseño robusta que combina la tecnología de un procesador de doble núcleo ARM Cortex A9 con la flexibilidad de los dispositivos programables FPGA Soc. Esta herramienta permite tener todas las ventajas que brindan los FPGA sumado a un procesador ARM que trabaja en modo HPS (Hard Processor System), capaz de controlar dispositivos de audio, video, almacenamiento, ethernet, así como también dispositivos para trabajar todos los aspectos básicos de lógica digital, lo que convierte a esta tarjeta en un módulo ideal para el aprendizaje sobre tecnología FPGA.

Altera University Program provee documentación, tutoriales y laboratorios prácticos para aprender todo lo referente al diseño e implementación de circuitos FPGA a través de esta tarjeta, pero su funcionalidad no se ve limitada al programa de aprendizaje de Altera ya que una vez se tienen los conceptos claros las aplicaciones antes discutidas pueden ser implementadas con esta tarjeta.

La tarjeta trabaja con la suite de programación Quartus II que dispone de una versión web totalmente gratuita y con la cual se pueden completar perfectamente los aspectos básicos de diseño para FPGA, la tarjeta puede ser programa con VHDL, Verilog e IP cores, la suite trabaja perfectamente en sistemas operativos de 64 bits (Windows o Linux) y puede ser descargada desde el sitio web del fabricante.

Figura 2. 5: Tarjeta DE1 de altera
Elaborada por: Altera.

2.8.1. Especificaciones

La tarjeta DE1-SoC dispone de los componentes necesarios para realizar todos los laboratorios y tutoriales disponibles en el programa universitario de ALTERA, desde el sitio web de TERASIC se puede descargar el CD correspondiente a la tarjeta.

El CD contiene los manuales de funcionamiento, demostraciones y hojas de datos de cada uno de los componentes incluidos.

La disposición final de los elementos incluidos se puede observar en la figura 1.8.

El diagrama de bloques de la figura 1.9 ilustra la conexión interna los elementos, cada uno está conectado directamente al chip FPGA SoC de forma que el usuario tenga la disponibilidad de cada uno de los elementos de forma independiente. El número que reflejan las flechas corresponden a los pines del FPGA que están conectados a ese dispositivo en específico.

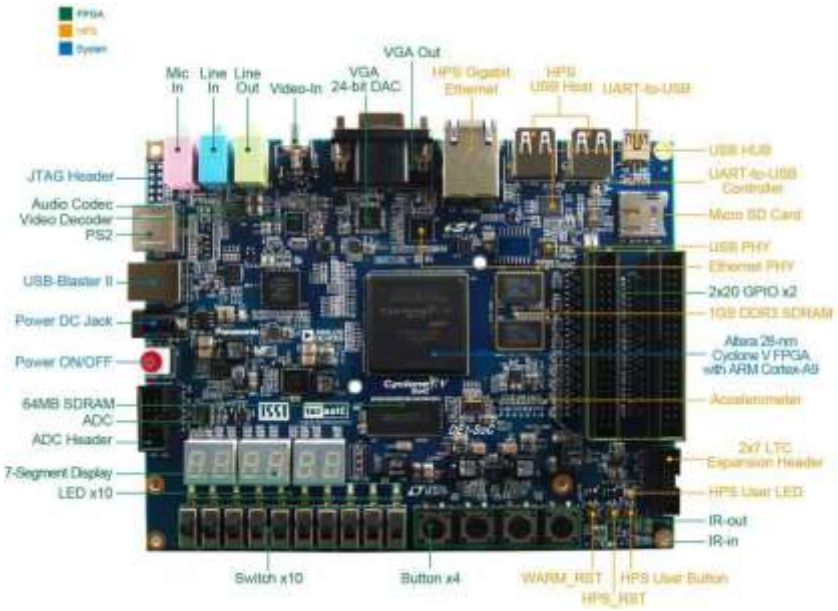


Figura 2. 6: Mapa de elementos de la tarjeta DE1-Soc
Elaborada por: Altera.

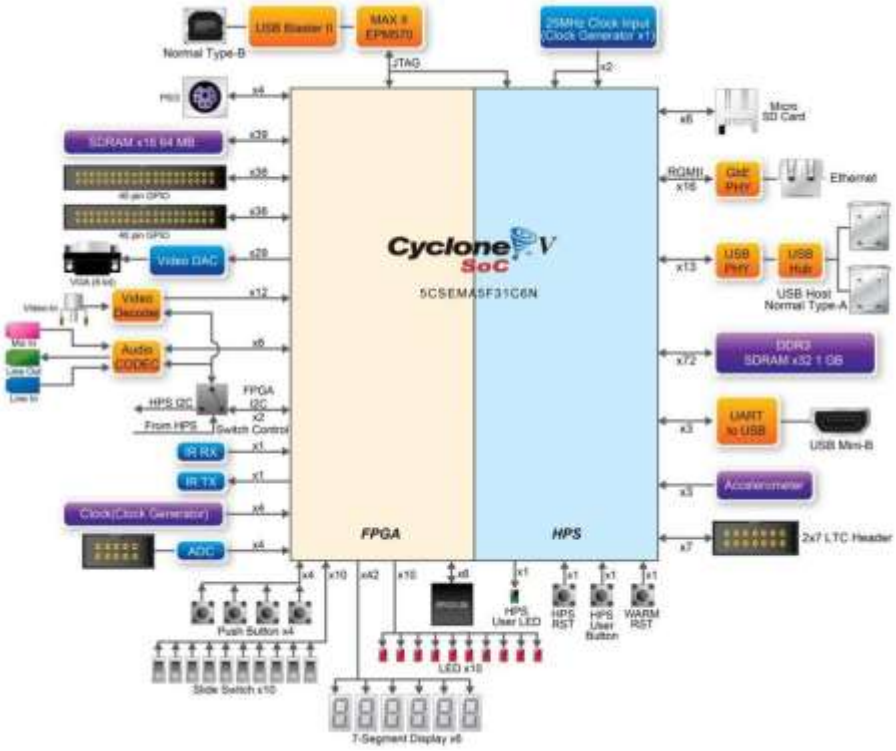


Figura 2. 7: Diagrama de Bloques de la tarjeta DE1-Soc
Elaborada por: Altera.

A continuación se enlistan las especificaciones técnicas de cada uno de los componentes de la tarjeta así como una breve descripción de su función:

2.8.2. Fpga De1 Soc.

- Chip FPGA Cyclone V SoC 5CSEMA5F31C6: es un chip de la familia Cyclone V SE, con modulo SoC, modulo para expansión de memoria, sin módulo PCI, cuenta con 85K LE, tipo de empaquetamiento FBGA, dispone de 896 pines, trabaja en rangos de temperatura comercial (0 ®C a 85 ®C) y posee el máximo grado de velocidad.

- Procesador Dual Core ARM Cortex A9: 800 MHz, doble núcleo, arquitectura ARMv7-A, lanzado en el año 2008, compatible con OpenCL, funciona en modo HSP.

- 4,450 Kbits de memoria embebida.

- 6 Módulos PLL que cuentan con compensación de retrasos, síntesis de precisión para la señal de reloj y pueden trabajar en modo integral o fraccionario.

- 4 módulos DLL (Delay Locked-Loop).

- 87 bloques de tipo DSP.

- 2 controladores de memoria física.

- Su estructura constructiva está basada en ALM's de 8 entradas.

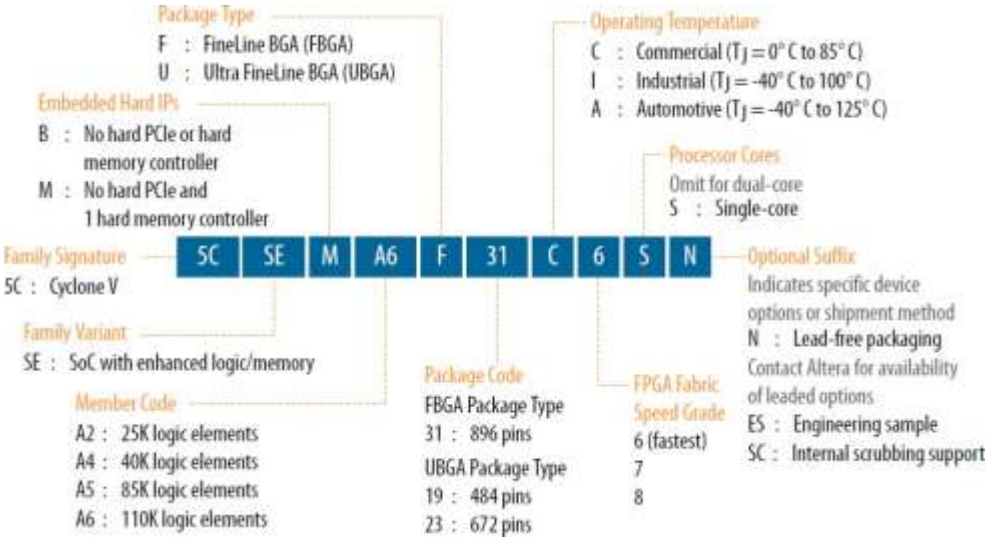


Figura 2. 8: Nomenclatura para dispositivos de la familia Cyclone V SE.
Elaborada por: Altera.

2.8.3. Configuración y Depuración

- Dispositivos EPCS128: Memoria Flash de 32 Mb, con interfaz de periférico serial (SPI), módulos I/O, capacidad de retención de 20 años, puede trabajar como DDR, buffer de 256 bytes, se utiliza para guardar los programas de configuración ya que el FPGA es volátil, el chip está configurado para extraer los datos de la memoria flash al iniciar, de esta forma cualquier programa guardado dentro de la misma se cargará automáticamente al FPGA al encender la tableta.

- Programador USB-BLASTER II incluido sobre la placa, con conector USB de tipo B.

2.8.4. Memoria

- Memoria SDRAM externa para chip FPGA: modelo IS4245R86400D, capacidad de 64 Mb, con voltaje de operación a 3.3V, el modulo puede ser utilizado para guardar de forma volátil los programas o para las funciones descritas en el sección anterior.

- Memoria DDR3 externa para procesador HPS: modelo IS43/46TR85120A, capacidad de 1 Gb, de uso exclusivo para el microprocesador, funciona de forma similar a una memoria RAM en una PC estándar.

- Módulo de expansión de memoria de almacenamiento SD: se recomienda usar memorias de categoría 10, y un valor mínimo de 8 Gb, esta memoria será la encargada de bootear el sistema operativo para que funcione el microprocesador.

2.8.5. Comunicaciones

- 2 puertos USB 2.0 modelo USB330, interfaz de comunicación ULPI, conector tipo A, diseñados para uso general, cualquier dispositivo conectado a ellos necesitara el driver necesario.

- USB a UART, conector tipo mini B, modelo número FT232R UART IC, los dispositivos UART son aquellos que utilizan un protocolo serial denominado Universal Asynchronous Receiver-Trasmitter para la transmisión y recepción de datos.

- Transceptor Ethernet 10/100/1000Mbps que cumple con la norma IEEE 802.3 con soporte RGMII (Reduced Gigabit Media

Independent Interface) es cual es un estándar que tiene como propósito reducir el número de pines para establecer la comunicación entre la MAC (Media Access Control) y el PHY (Physical Leyer Protocol).

- Puerto PS2 para mouse o teclado estándar.

- Trasmisor\Receptor infrarrojo.

- Multiplexor I^2C, es un bus de comunicaciones de tipo serial con una velocidad de 100Kbps.

2.8.6. Conectores

- 2 conectores header de 40 pines, cada pin representa un GPIO (input/output de propósito general), el nivel de voltaje en estos pines es de 3.3 V cuando están activos, en cada conector se encuentran disponibles 36 pines de datos, 2 pines fijos GND, 1 pin +5 V y 1 pin

- +3.3 V, son pines con entrada protegida por si hay sobre tensión, cada pin puede manejar un máximo de 1.5 mA.

- Los pines de propósito general tienen una capacidad de transferencia de datos de 875 Mbps LVDS para la trasmisión y 840 Mbps LVDS para la recepción.

- 1 conector header de 10 pines exclusivo para conectar ADC.

- 1 conector LTC.

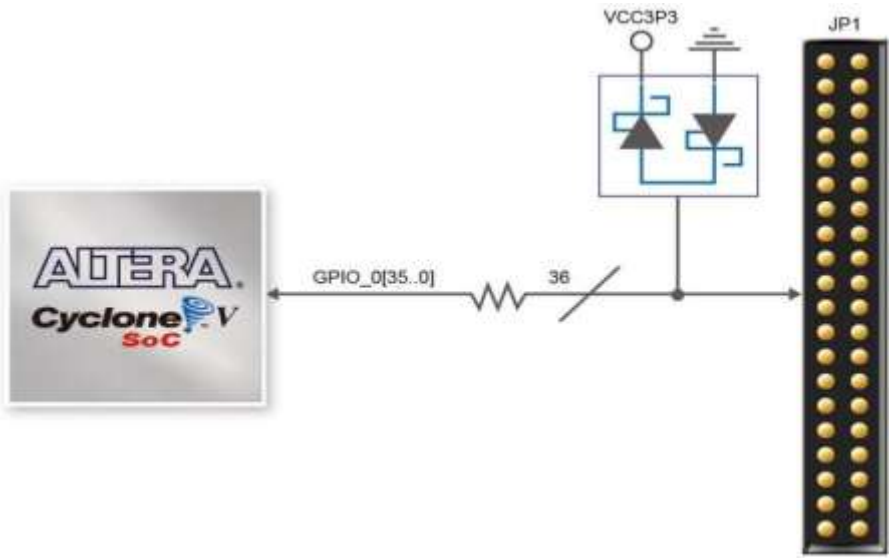


Figura 2. 9: Diagrama de conexión de pines GPIO.
Elaborada por: Altera.

2.8.7. Video

- ADC de 10 bits modelo ADV7123 dedicado a la trasmisión de señales de video en formato VGA.
- Video decodificado de 10 bits modelo ADV7180 para la recepción de señales de video en formato NTSC/PAL/SECAM.

2.8.8. Audio

- Decodificador de 24 bits modelo WM8731, con puertos Line in, Line out y headphones jack, utiliza el driver realtek ‘94 para el controlar el dispositivo.

ADC

- 12 bits de resolución, 8 canales de entrada, con un rango de 1 millón de muestras por segundo, los rango de voltaje para la entrada analógica son 0-2.5 V o 0-5 V, este rango es definido por el usuario mediante un registro.

2.8.9. Elementos de lógica digital

- 4 botones pulsadores, en estado de reposo su salida es ‘1’ lógico, mientras que al ser presionado brindan un ‘0’ lógico, su salida está

conectada directamente a un dispositivo 74HC245 que actúa como buffer no inversor.

- 6 displays de 7 segmentos ánodo común, se necesita un ‘0’ lógico para encender cada segmento.
- 10 interruptores de doble estado.
- 11 led (10xFPGA y 1xHPS)
- 2 botones de reset HPS

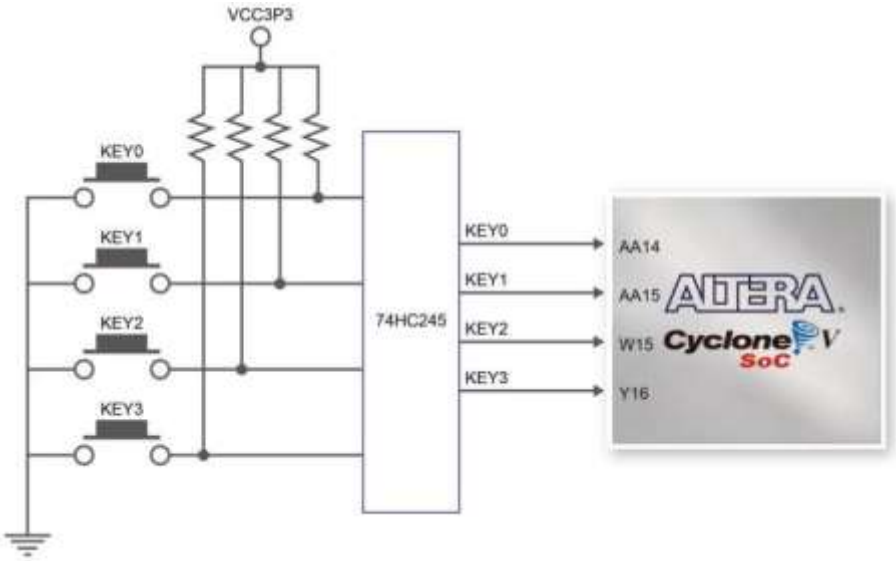


Figura 2. 10: Diagrama de conexión de para botones.
Elaborada por: Altera.

2.8.10. Sensores

- Acelerómetro de 3 ejes (X, Y, Z) con resolución de 10 bits, únicamente disponible a través de HPS.

2.8.11. Fuente

- Fuente de voltaje 12V@5ª

2.8.12. Circuitería de reloj

- 4 relojes de 50 MHz de uso exclusivo para FPGA.
- 2 relojes de 25 MHz (1 para HPS y 1 para Ethernet Transceiver).

- 1 reloj de 24 MHz para controlador USB (este reloj no está disponible para el usuario).

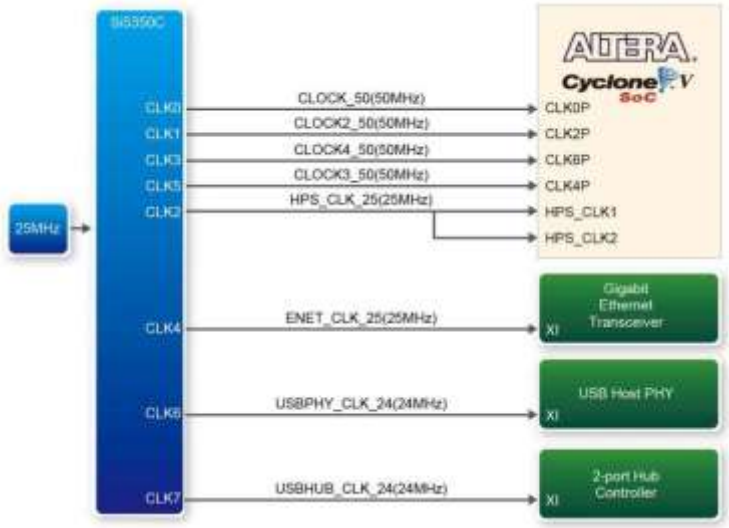


Figura 2. 11: Diagrama de bloques para la circuitería de reloj.
Elaborada por: Altera.

2.8.13. Configuración de Fpga

La tarjeta DE1-SoC puede ser configurada para modo EPCS o HPS, esto se logra con un DIPSWITCH de 6 bits situado en la parte posterior de la tarjeta, la figura 1.14 muestra la ubicación física dentro de la tarjeta.

La diferencia fundamental entre los modos es la integración de sus componentes ya que el modo EPCS solo utiliza el FPGA, mientras que el modo HPS incluye el procesador ARM al trabajo del FPGA. Algunos periféricos incluidos en la tarjeta solo pueden ser operados en el modo HPS.

Cada modo posee sus ventajas y desventajas propias del desarrollo de circuitos específicos pero en forma general, el modo EPCS permite operar el FPGA de forma volátil y no volátil, la ejecución de programas es más rápida y no está sujeto un sistema operativo, en cambio el modo HPS le provee potencia extra a la tarjeta, y activa todos los periféricos disponibles, con la

limitante de estar sujeto a un sistema operativo, las FPGA manejan varios ecosistemas como por ejemplo Linux y android, este sistema se encarga de controlar los procesos internos.

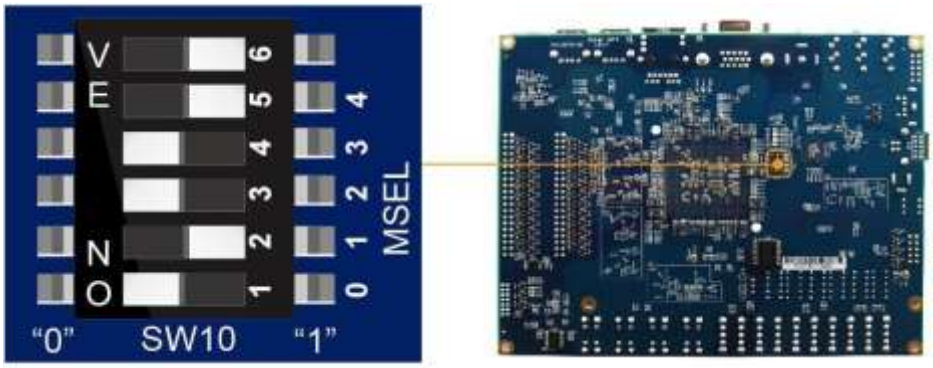


Figura 2. 12: Dipswitch de configuración de tarjeta DE1-Soc.
Elaborada por: Altera.

Por defecto la tarjeta viene configurada para funcionar en modo AS (Active Serial) de tal forma que al energizarse los datos contenidos en la memoria EPCS sean descargados al chip FPGA, esta configuración corresponde a la configuración MSEL*4:0+ = "10010", si el usuario necesita reconfigurar el FPGA para que funcione mediante una herramienta de software (Linux) la configuración debe ser MSEL*4:0+ = "01010", existe un tercer modo de configuración utilizado para cargar Linux LDXE Desktop desde una memoria SD, para acceder a este, la combinación debe ser MSEL*4:0+ = "00000".

2.8.14. Métodos de programación

El FPGA puede ser programado por dos métodos diferentes:

- **Programación por JTAG**: Este método emplea el protocolo IEEE JTAG (Joint Test Action Group), los bits son descargados directamente hacia el chip FPGA, los datos de programación estarán presentes en el chip siempre y cuando este se mantenga energizado, recordando que el FPGA en sí mismo es un dispositivo volátil, este método de programación es ideal para la

prueba de códigos ya que la elaboración de un archivo *.sof no requiere un proceso extra más allá de una compilación que ejecuta el mismo software.

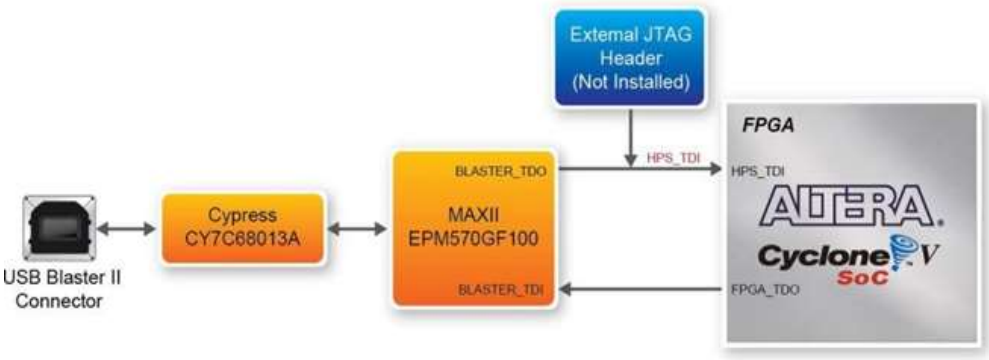


Figura 2. 13: Diagrama de bloques para programación JTAG
Elaborada por: Altera.

- **Programación por Active Serial**: Este método emplea archivos de extensión .JIC los cuales son configurados en la memoria EPCS256 de forma que la programación sea permanente para el chip FPGA, los archivos requeridos para la configuración no son creados por defecto, requieren un proceso extra que será discutido en el capítulo siguiente.

Figura 2. 14: Diagrama de bloques para programación Active Serie
Elaborada por: Altera.

Capítulo 3: Implementación de un sistema de comunicación usando Fpga.

La manipulación de los microprocesadores es complicada ya que para que este funcione en un proyecto determinado es necesario crear el hardware que se ajuste a los requerimientos de alimentación y protección de estos, para ello existen las placas entrenadoras, las que permiten que un usuario pueda implementar un sistema previamente con varios lenguajes como son VHDL, Verilog, esquema de bloques, entre otros.

En este capítulo se describirá el procedimiento y los elementos utilizados para la implementación de un sistema de comunicaciones utilizando el protocolo UART permitiendo compartir información de una placa entrenador FPGA Altera a otra, por consiguiente, se realizará pruebas previas entre esta y un computador para poder visualizar mediante un software especializado la información recibida.

3.1. Elementos utilizados para la implementación de un sistema de comunicación con FPGA Altera.

En esta sección del capítulo se detalla el procedimiento y elementos a utilizar para el sistema de comunicación con FPGA Altera.

3.1.1. Placa entrenadora FPGA Altera DE1

Esta placa de entrenamiento es diseñada para el ámbito estudiantil y de desarrollo ya que cuenta con el microprocesador embebido en una pcb que proporciona ciertas conexiones y puertos predefinido por el fabricante.

Esta placa ofrece un amplio conjunto de características que lo hacen adecuado para el uso en un entorno de investigación y desarrollo de sistemas digitales avanzados. Altera proporciona adicionalmente un software y tutoriales en su página web oficial.

En la figura 3.1 se puede contemplar el modelo de tarjeta FPGA Altera que se eligió para el propósito del diseño de un sistema de comunicación.

Figura 3. 1: Tarjeta Altera DE1.
Elaborado por: Autor.

Esta tarjeta cuenta con un núcleo Cyclone II 2C20 que tiene 484 pines, todos los componentes importantes de esta tarjeta están conectados a los pines del microprocesador permitiendo al usuario o investigador el control total de las operaciones de la placa.

Para experimentación la DE1 posee interruptores y pulsadores que permiten utilizarlos como interfaz de entrada al microprocesador, además de led que permiten visualizar la salida de resultados así también como cuatro display de 7 segmentos.

Para experimentaciones más avanzadas también cuenta con SRAM, SDRAM y una memoria flash, además cuenta con una entrada de RS232 y PS/2.

La tarjeta altera DE1 cuenta con tres osciladores de 50MHz, 27MHz, 24MHz y un conector sma para un reloj de fuente externa que permitirá la utilización de la velocidad de reloj necesaria para el desarrollo o experimentación de proyectos.

Como salidas adicionales, esta placa cuenta con dos bloques de 40 pines de propósito general (FPIO) con un nivel de voltaje de 3.3V que permiten conectar cualquier dispositivo adicional según el diseño de los proyectos.

Una de las ventajas de trabajar con esta placa entrenadora es que ya posee integrado el USB Blaster que permite la carga del código de programación al microprocesador JTAG, AS y PS.

La siguiente figura 3.2 demuestra el contenido del paquete de la tarjeta obtenida para el diseño del sistema de comunicación en FPGA.

Figura 3. 2: Kit FGPA Altera DE1.
Elaborado por: Autor.

El microprocesador Cyclone II 2C20F484C7, figura 3.3, es de bajo costo de alta densidad el cual forma parte de la placa entrenadora, tiene ciertas características interesantes como es, por ejemplo, su alta densidad de elementos lógicos (Les) que en este modelo llega hasta los 18752 proveyendo unas 622 entradas y salidas lógicas.

Figura 3. 3: EP2C20F484C7.
Elaborado por: Autor.

Los dispositivos Cyclone II pueden admitir sistemas digitales complejos en un solo micro a bajo costo a diferencia de otros proveedores de FPGA, estos ofrecen un 60% más rendimiento que su competencia con la mitad del consumo de energía.

Este microprocesador puede llegar a funcionar con reloj de 260MHz, entradas y salidas de alta velocidad y estándar, entre otras funcionalidades que permiten al investigador crear o estructurar sistemas digitales complejos.

3.1.2. Quartus II v13.

Este software es proporcionado por los fabricantes de la tarjeta entrenadora, con este es posible el diseño electrónico digital utilizando varios tipos de lenguajes y esquemáticos.

La interfaz de usuario es muy amigable ya que permite la utilización intuitiva para que el investigador desarrolle un producto digital y pueda medir los resultados, en la figura 3.4 se presenta el software que se utilizara.

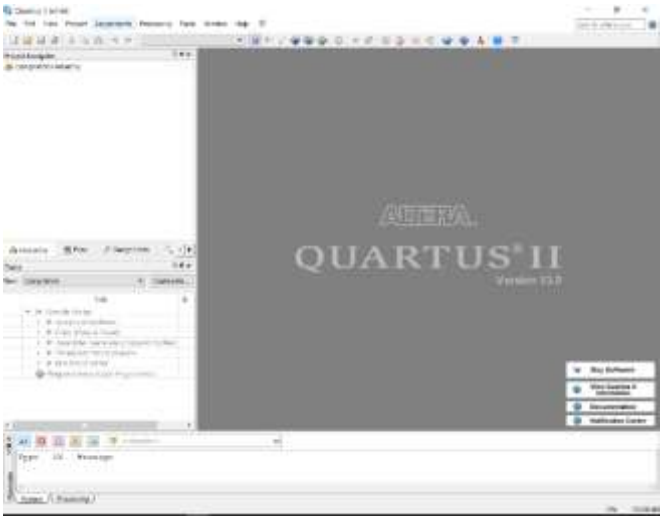


Figura 3. 4: Software Quartus.
Elaborado por: Autor.

Este software permite la creación de un sistema digital mediante varios lenguajes como son: HDL, Verilog, VHDL, Tcl, entre otras como se aprecia en la figura 3.5 obtenida a partir de la creación de un nuevo proyecto.

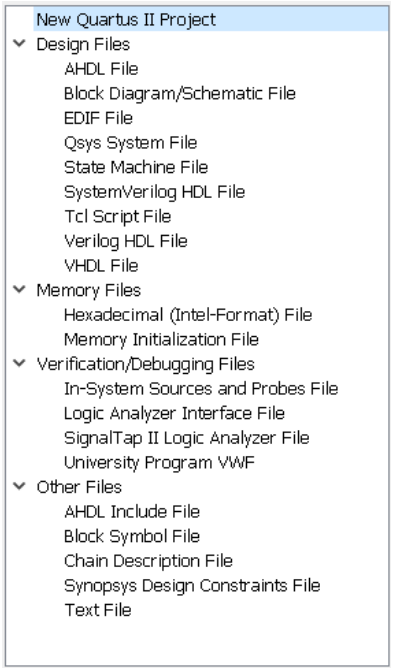


Figura 3. 5: Tipos de archivos para creación de sistemas digitales en Quartus.
Elaborado por: Autor.

3.1.3. Utilitario Hércules

Hércules es un utilitario que permite visualizar un terminal de puerto serie (RS232 o RS485), figura 3.6, además también de un terminal UDP/IP y terminal de cliente TCP/IP, inicialmente fue creado para uso interno de la empresa HW Group, pero ya hoy es freeware, es decir de uso libre y gratuito.

Este software tiene soporte completo para Windows 7, 8 y 10, el terminal puerto serie implementado está funcionando con los puertos serie virtuales COM12, por ejemplo, y puede verificar y controlar todas las líneas (CTS, RTS, DTR, DSR, RI, CD.

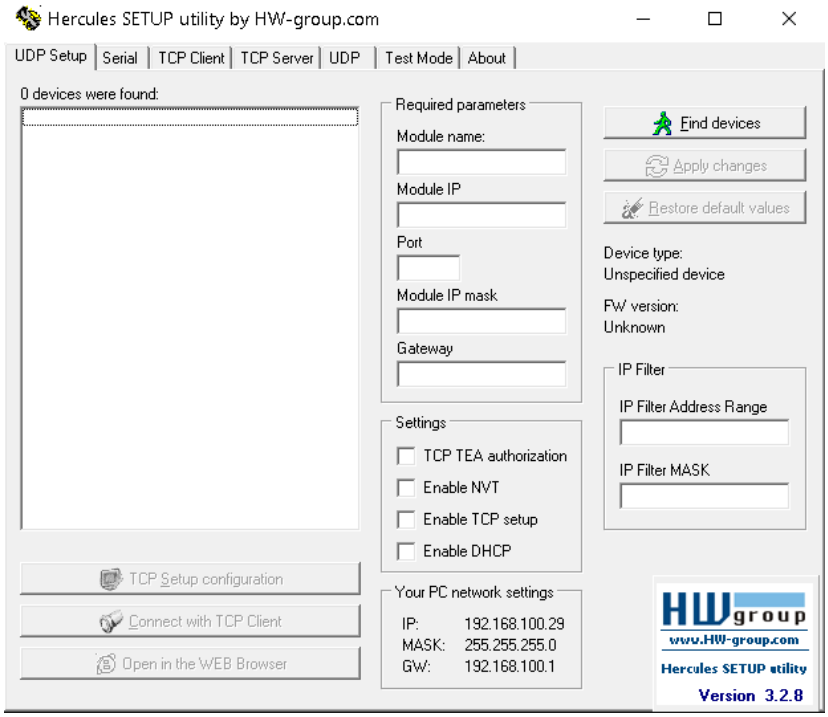


Figura 3. 6: Utilitario Hércules.
Elaborado por: Autor.

En este caso, se utilizará el modo serial, ya que se realizarán pruebas desde el fpga hacia la computadora para visualizar los datos transmitidos por este de forma serial, y como es de esperarse, este software permite la selección del puerto com virtual y la configuración de velocidad de transmisión en baudios, como se puede apreciar en la figura 3.7.

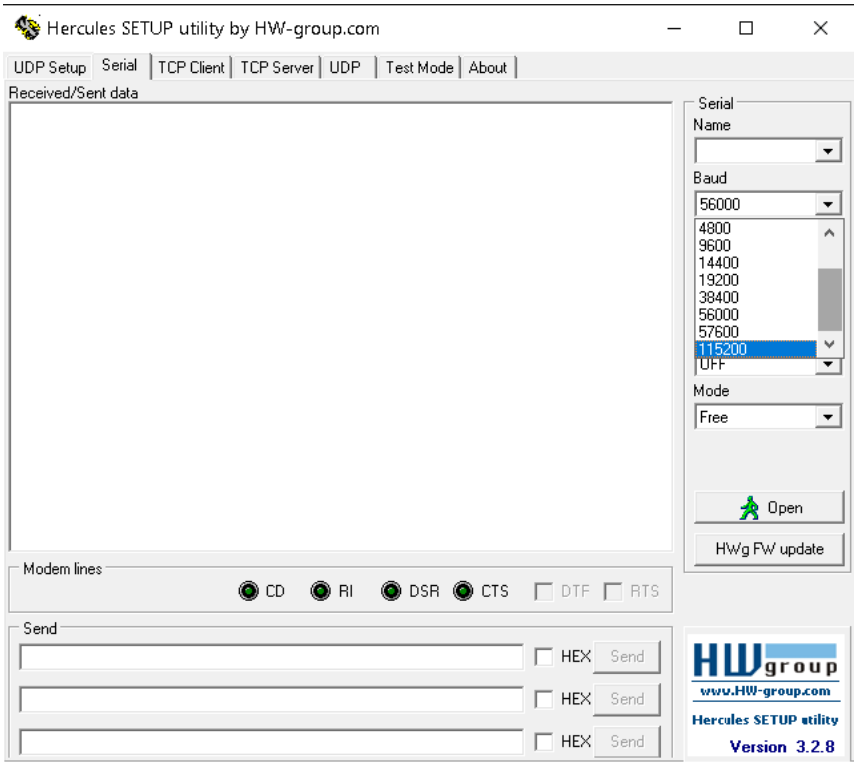


Figura 3. 7: Configuración de velocidad de puerto serie.
Elaborado por: Autor.

3.1.4. Modulo Bluetooth HC05

El módulo HC05 es un dispositivo electrónico embebido que permite la conectividad por bluetooth a cualquier otro de forma inalámbrica, posee un puerto serial que facilita la operación del mismo y su transmisión lo realiza de forma transparente desde cualquier microcontrolador o microprocesador respetando los niveles TTL de comunicación.

En la figura 3.8 se aprecia el modelo que se utilizara para pruebas inalámbricas del sistema de comunicación del FPGA.

Figura 3. 8: Módulo de comunicación inalámbrica bluetooth HC05.
Elaborado por: Autor.

Entre sus características más importantes están:

- El voltaje de operación es de 3.6-6V DC
- Frecuencia Banda ISM 2.4GHz.
- Consumo de corriente de 50mA.
- Potencia de transmisión de 4dBm, clase 2.
- Alcance de 10 metros.
- Velocidad de transmisión de 1200bps hasta 1.3Mbps.
- Su modulación es GFSK.
- Interfaz de comunicación TTL.
- Sensibilidad de -84dBm a 0.1% BER.

Este dispositivo permite reemplazar el enlace cableado serial por uno inalámbrico de forma transparente, además, se puede configurar como master o slave, siendo ideales para los proyectos de investigación o implementación de prototipos de cualquier índole.

3.2. Diseño del sistema de comunicación con FPGA.

En esta sección del capítulo se describirá la estructuración que tendrá el sistema desarrollado con fines de investigación y comprobar que un sistema de comunicaciones puede ser implementado sin ningún problema en un microprocesador altera.

3.2.1. UART

UART por sus siglas en ingles Universal Asynchronous Receiver/Transmitter es un puerto de hardware que permite la comunicación asíncrona de varios micro controladores o hardware embebido dando posibilidad a transmitir o recibir datos pequeños al mundo exterior.

Los niveles que trabaja este puerto son TTL, es decir consume muy poco ya que los voltajes lógicos son muy pequeños, en esta etapa es posible comunicar directamente de microcontrolador a microcontrolador, pero es

necesario pasar a una capa física como el RS232 para aumentar la eficiencia de esta comunicación.

Un nivel TTL va desde los 0 lógicos con 0 voltios a los 1 lógicos que son representados con 5 o 3.3 voltios, como se dijo anteriormente, estos consumen muy poca energía y a partir de aquí se pueden convertir en cualquier estándar establecidos para la comunicación.

El funcionamiento es sencillo, la trama de datos que genera este puerto se aprecia en la figura 3.9, envía datos en forma de bits (1 y 0 lógicos) y puede representar casi cualquier carácter utilizando el código ASCII, es decir, si se necesita enviar la letra A el valor asociado para este carácter es 97, convirtiendo a binario queda representado con la serie 01100001, que es prácticamente lo que se requiere transmitir.

Figura 3. 9: Trama para la comunicación serie UART.
Elaborado por: Autor.

Muchas personas relacionan el estándar RS232 con la comunicación UART, pero cabe recalcar que ambas son diferentes, si se lo ilustra en forma de capas, el UART vendría a ser la capa 2 de enlace de datos, y el RS232 tendría su campo de acción en la capa 1 física.

3.2.2. Análisis del protocolo de comunicaciones RS232

El protocolo RS232 es un estándar mundial que fue implementado en el año 1962 y se sigue utilizando en la actualidad en el ámbito de micro controladores y microprocesadores para transmitir información pequeña de manera asíncrona ya que su velocidad es baja comparando las tecnologías y estándares actuales.

A pesar de todo lo dicho anteriormente, el RS232 aún sigue proporcionando un servicio de gran valor para varias industrias y negocios como el caso de la comunicación entre microcontroladores y microprocesadores.

A nivel de software la conectividad proporcionada por un puerto RS232 se basa en su velocidad en baudios (300, 1200, 2400, 4800, 9600, 19200, etc.), la verificación de datos y paridad, los bits de parada terminado el ultimo bit de dato y por supuesto la cantidad de bit por dato que oscilan entre 7 y 8 bits.

Cabe recalcar que la velocidad se conoce que se la mide en Kbps pero este no debe confundirse con los baudios, ya que este representa el número de cambios en la señal modulada por segundo, y es en ello que se enfoca la investigación para realizar mediante el FPGA la generación de las señales que permitirán la transmisión de un dato mediante el conocimiento de la estructura de la trama correspondiente a la comunicación asíncrona UART pasando por el convertidor MAX232 necesario para en los niveles del estándar RS232.

En la figura 3.10 se presenta como está constituida la trama de la comunicación serie en RS232.

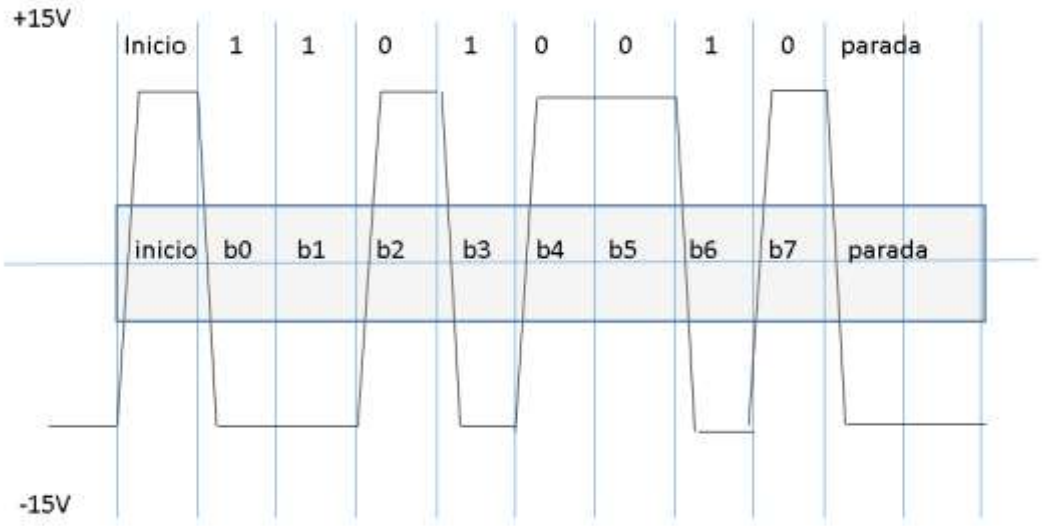


Figura 3. 10: Trama para la comunicación serie RS232.
Elaborado por: Autor.

3.2.3. Análisis de protocolo Bluetooth para utilización en capa física del sistema de comunicaciones con FPGA

El protocolo bluetooth es utilizado en muchos dispositivos electrónicos en la actualidad, la necesidad básica de todos estos es la comunicación. Este

estándar de comunicación es de corto alcance ofreciendo establecer enlaces para distintos escenarios y propósitos.

Este estándar de comunicación es como se dijo anteriormente de corto alcance y adicional es de bajo consumo, esta última característica es muy importante para el desarrollo de prototipos y proyectos que puedan tener un gran impacto en la aplicación.

Este protocolo se encuentra constituida por varias capas, estas se pueden ver organizadas por grupos como son: el de transporte, middleware y de aplicación como se muestra en la figura 3.11, los datos fluyen a través de todas las capas.

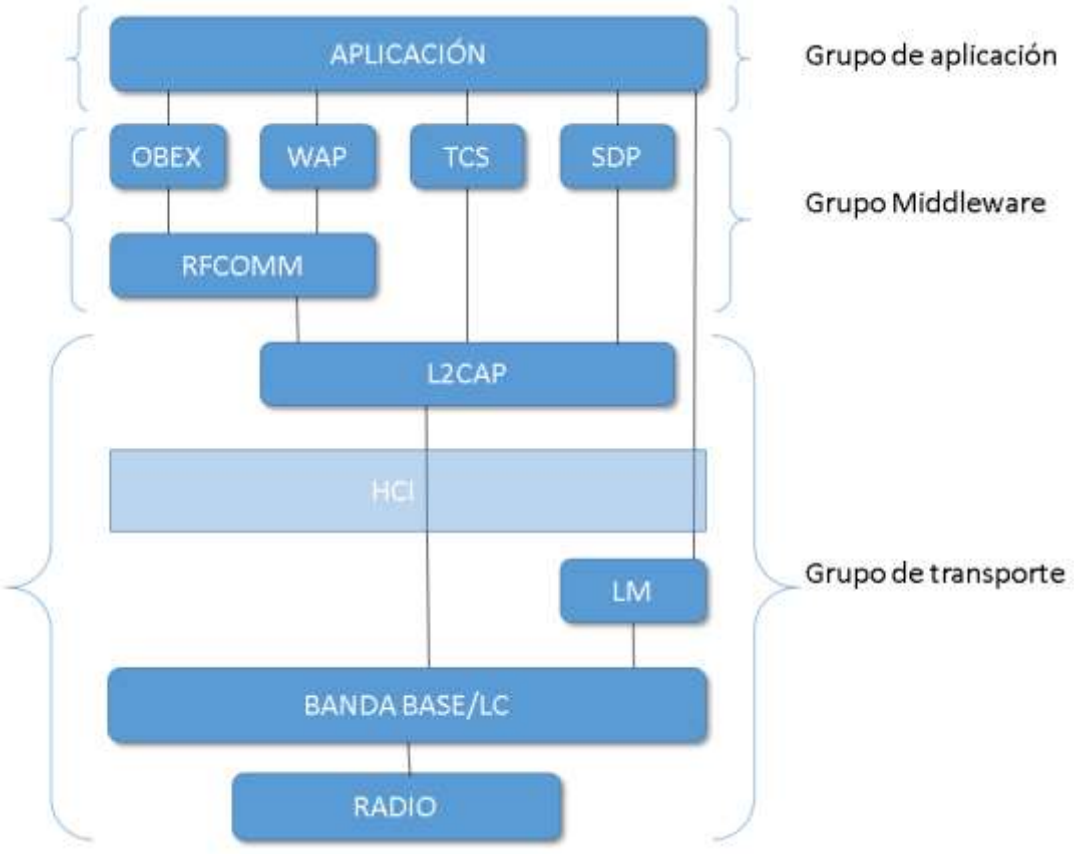


Figura 3. 11: Capas constituidas para el protocolo bluetooth.
Elaborado por: Autor.

3.2.4. Diseño de funcionamiento

En esta sección del capítulo descrito se presentará las pautas y diagramas que se realizó para desarrollar el sistema de comunicación

utilizando en esta ocasión el protocolo serie asíncrono en un FPGA, para ello es necesario separar por etapas y designar la tarea que desempeñara cada una.

En la figura 3.12 se presenta un diagrama de bloques básico de cómo está conformada un sistema de comunicación.

Figura 3. 12: Diagrama de bloques de un sistema de comunicación básico.
Elaborado por: Autor.

En este caso se podrá construir para ambos extremos del sistema de comunicación utilizando FPGA programando en lenguaje de descripción de hardware VHDL, para ello hay que diagramar el funcionamiento que se desea tener de dicho sistema, figura 3.13.

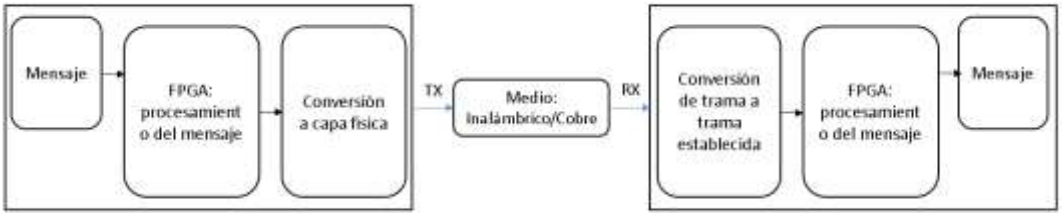


Figura 3. 13: Diagrama de bloques del sistema de comunicación UART en FPGA en un sentido.
Elaborado por: Autor.

En la primera parte del diagrama de bloques anteriormente presentado describe la generación del mensaje por parte del usuario, para ello se habilitará los interruptores que posee la tarjeta FPGA DE1 y será una trama de 8 bits que gracias al código ASCII se puede describir con cualquier caracter.

Posteriormente el FPGA debe adquirir este mensaje descrito por el usuario para construir la trama que será enviada a la capa física del sistema,

este será de dos maneras, mediante cable de cobre utilizando el convertidor de niveles RS232 integrado en la tarjeta e inalámbrico mediante el modulo bluetooth HC05.

Para ello es necesario crear dos códigos que describan el hardware que debe funcionar en ambos extremos, uno que se encargue de enviar un mensaje propuesto de 8 bits y el siguiente hardware se encargara de recibir dicho paquete de datos para representar en los leds que posee la placa integrada.

En el diagrama de flujo que se presenta en la siguiente figura 3.14 se muestra el funcionamiento del hardware encargada de generar el mensaje y transmitirlo por los medios que se propuso anteriormente en este documento.

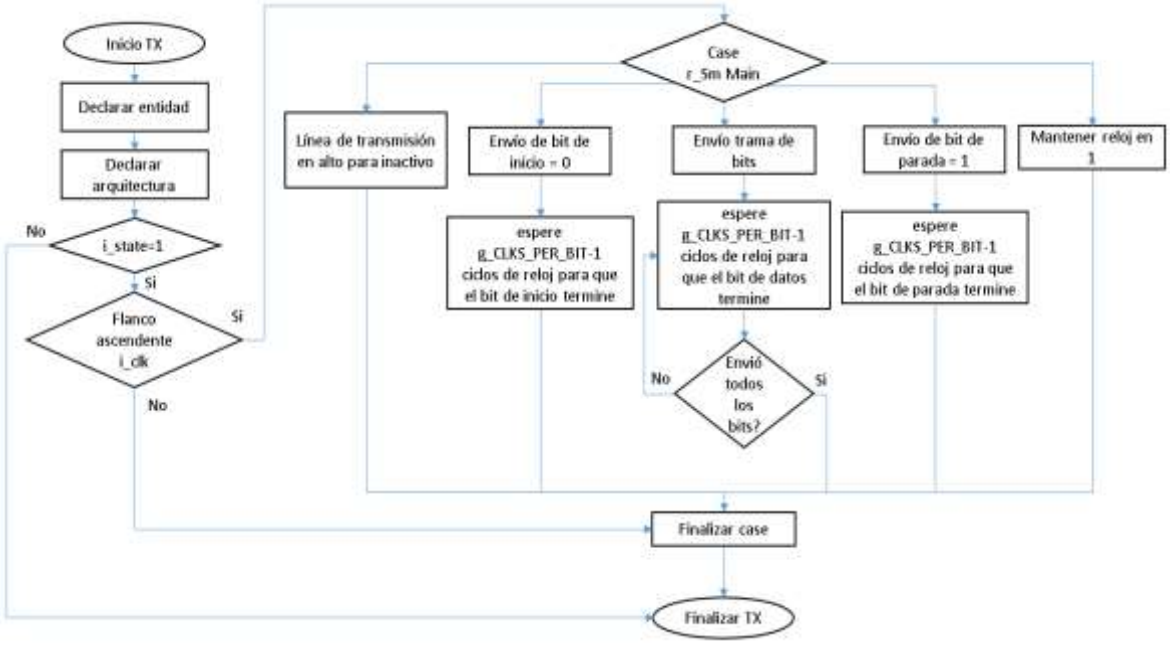


Figura 3. 14: Diagrama de flujo para el código de transmisión.
Elaborado por: Autor.

En el diagrama anterior se muestra la primera parte el inicio del proceso declarando la entidad y su arquitectura respectivamente, posteriormente mediante un cuadro de control if se pregunta si el estado del switch 9 está en alto o bajo ya que este determina la continuidad del proceso para evitar saturar el buffer de la capa física.

Luego de haber determinado el estado de la señal state, se procede a cuestionar por medio de otro cuadro de control el flanco ascendente del reloj

para posteriormente se ejecuta una sentencia de control switch para los diferentes casos que puede estar ejecutando en determinado momento.

Se tiene cinco casos en los cuales se realizará diferentes procesos de acuerdo a su estado inicial, siendo el análisis de la línea de transmisión, el envío de bit de inicio, envío de la trama, envío de bit de parada y por último el proceso que ayuda a mantener el reloj en 1 cuando se realice el conteo respectivo.

Cada uno se ejecutará de acuerdo al momento lógico del reloj que se encuentre, como es una comunicación asíncrona no es necesario enviar el estado del reloj.

Cuando se trata de una comunicación UART en un microcontrolador, este está diseñado para efectuar las interrupciones dentro para atender el mensaje recibido, en el caso de los FPGA no es necesario las interrupciones ya que está diseñada para trabajar en paralelo con otros procesos.

Así mismo, en la parte de recepción del sistema de comunicación se debe realizar un diagrama de flujo especificando la función del proceso a ejecutarse, en la figura 3.15 se muestra el diagrama de flujo diseñado para el fpga que se encargara de recibir la trama de manera asíncrona.

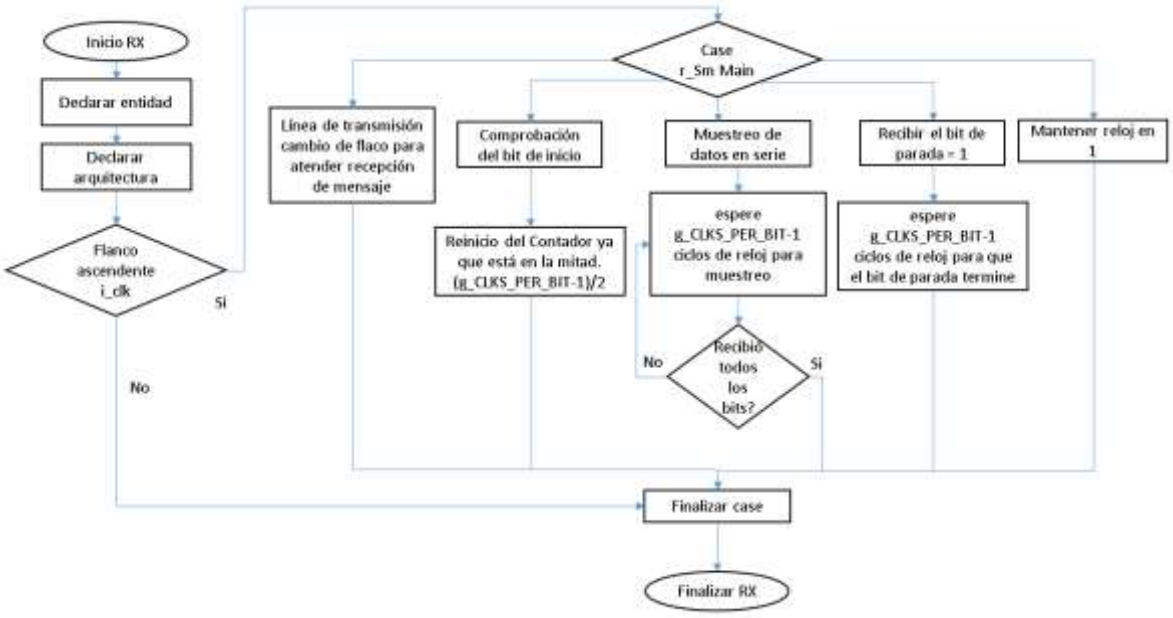


Figura 3. 15: Diagrama de flujo para recibir mensaje desde FPGA.
Elaborado por: Autor.

En este caso para recibir un mensaje por un medio de transmisión alámbrico o inalámbrico se debe iniciar en la declaración de entidad y

arquitectura como se muestra en la primera parte del diagrama de flujo anterior, posteriormente se procede a cuestionar el flanco del reloj para iniciar la sentencia del switch case, mediante realizara ciertas funciones de acuerdo a su estado.

En este caso no es necesario tener un estado de control de recepción, ya que este se realiza de manera asíncrona y siempre estará atento a los cambios del flanco de la línea de comunicación.

Para determinar la velocidad de muestreo del FPGA es necesario realizar un cálculo entre la frecuencia del cristal elegido y su velocidad de transmisión en baudios, como se muestra en la siguiente figura 3.16.

$$\frac{Frecuencia\ Clk}{Vel.baudios} = muestreo$$

$$\frac{50000000}{9600} = 5208{,}33$$

$$\frac{50000000}{14400} = 3472{,}22$$

$$\frac{50000000}{19200} = 2604{,}16$$

$$\frac{50000000}{115200} = 434{,}02$$

Figura 3. 16: Calculo para velocidad de muestreo por bit según frecuencia de reloj y velocidad de transmisión.
Elaborado por: Autor.

En este caso se elige la velocidad de reloj a 50MHz ya que es uno de los que se tiene presente en la placa FPGA altera de1 y por ser de mayor velocidad para evitar que el muestreo muy alto ya que son inversamente proporcional a la velocidad de transmisión.

3.3. Implementación de código VHDL para sistema de comunicación UART en FPGA.

A continuación, en esta sección del capítulo se detalla el procedimiento que se realizó para el diseño del sistema de comunicación utilizando FPGA Altera DE1.

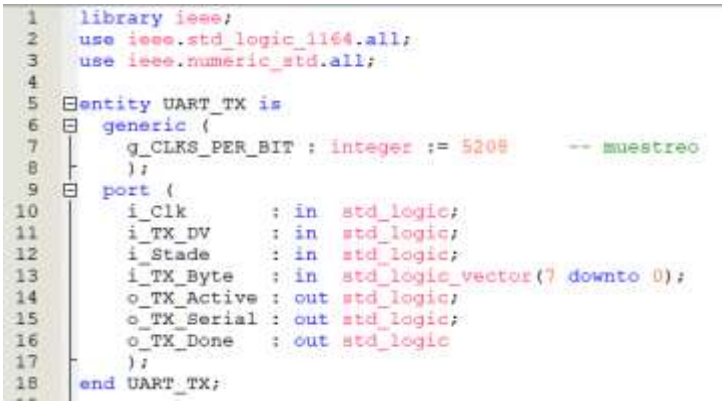
Una vez realizado el diseño de la aplicación se procede a diseñar un proyecto nuevo en el software Quartus, figura 3.17, para derivar todo el diagrama de flujo que se planteó anteriormente a código de descripción de hardware.

Figura 3. 17: Creación de nuevo proyecto en Quartus.
Elaborado por: Autor.

Una vez creado el proyecto es necesario crear un nuevo archivo para VHDL, este contendrá el código inicialmente para el transmisor para realizar pruebas de funcionamiento con los dos tipos de medios de trasmisión propuestos en este documento.

Una vez creado el proyecto que se llamará UART_TX, se procede a crear el código, a continuación, se describirá el código escrito por partes para la comprensión del lector.

En la primera parte del código CHDL se muestra la librería a utilizar y la entidad que describirá los puertos de entrada y salida, en la figura 3.18 se presenta dicho contenido.



```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity UART_TX is
6    generic (
7      g_CLKS_PER_BIT : integer := 5208     -- muestreo
8      );
9    port (
10     i_Clk       : in  std_logic;
11     i_TX_DV     : in  std_logic;
12     i_Stade     : in  std_logic;
13     i_TX_Byte   : in  std_logic_vector(7 downto 0);
14     o_TX_Active : out std_logic;
15     o_TX_Serial : out std_logic;
16     o_TX_Done   : out std_logic
17     );
18 end UART_TX;
```

Figura 3. 18: Librería y entidad para UART_TX.
Elaborado por: Autor.

Como se indicó previamente, el muestreo está definido por la velocidad de reloj y de transmisión de datos, con los cálculos previos se define un valor de 5208 para trabajar a 9600 baudios, posteriormente se definen los puertos que permiten la construcción del hardware de manera digital en el FPGA.

A continuación, se construye la arquitectura RTL para la entidad UART_TX, en la figura 3.19 se muestra esto, existen varias señales entre ellas están r_SM_Main que permitirá el control principal de la transmisión, un contador para el reloj, un contenedor para la trama de bits, un vector para la construcción de la trama de datos, y una señal lógica para el estado lógico de transmisión.

```
21 architecture RTL of UART_TX is
22
23   type t_SM_Main is (s_Idle, s_TX_Start_Bit, s_TX_Data_Bits,
24                      s_TX_Stop_Bit, s_Cleanup);
25   signal r_SM_Main : t_SM_Main := s_Idle;
26
27   signal r_Clk_Count : integer range 0 to g_CLKS_PER_BIT-1 := 0;
28   signal r_Bit_Index : integer range 0 to 7 := 0;  -- 8 Bits Total
29   signal r_TX_Data   : std_logic_vector(7 downto 0) := (others => '0');
30   signal r_TX_Done   : std_logic := '0';
31
```

Figura 3. 19: Arquitectura RTL para la entidad UART_TX.
Elaborado por: Autor.

A continuación, se procede a describir el inicio del proceso principal, este como se detalló previamente en el diagrama de flujo inicia con un doble if para determinar el estado del switch i_state que permite la ejecución de transmisión de datos y así evitar saturar el buffer de la capa física sea RS232 o Bluetooth, en la figura 3.20 se muestra lo descrito.

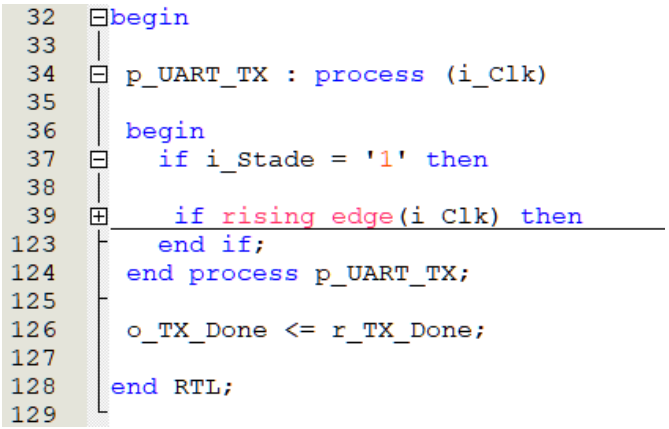


Figura 3. 20: Inicio de proceso para transmisión de mensaje.
Elaborado por: Autor.

Posteriormente en la figura 3.21 se muestra de manera desplegada el código del proceso la sentencia de control switch que permite de acuerdo al caso realizar determinadas acciones, para el primer caso cuando s_Idle este dada por parámetros que se muestran en la imagen ejecutará una sentencia if que evalúa i_TX_DV y permitirá que r_TX_Data adquiera el valor de i_Tx_Byte, y r_SM_Main obtenda el valor de s_TX_Start_Bit en caso de ser afirmativo, caso contrario r_SM_Main obtendrá el valor de s_Idle.

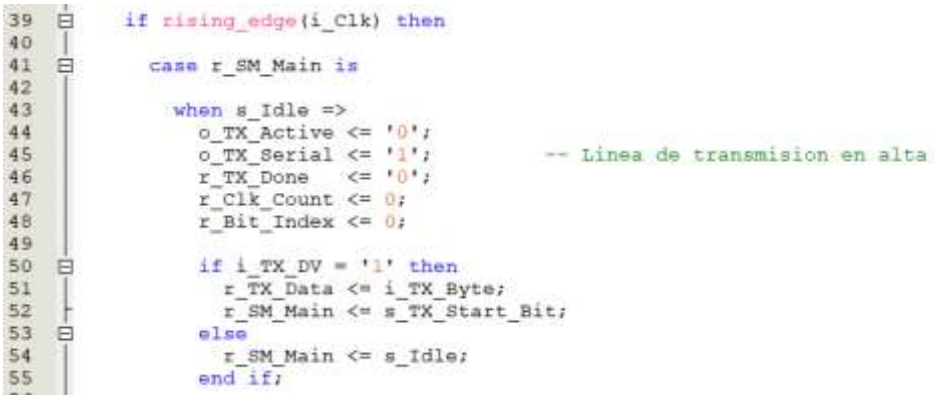


Figura 3. 21: Primer caso de sentencia switch.
Elaborado por: Autor.

Para el siguiente caso que comprende el envío de bit de inicio igual a cero, es decir el cambio de flanco descendente se muestra en la figura 3.22 la sentencia if que permite la espera de los ciclos de reloj necesarios para que este termine.

```
58          -- Envio de bit de inicio: bit = 0
59        when s_TX_Start_Bit =>
60          o_TX_Active <= '1';
61          o_TX_Serial <= '0';
62
63          -- Esperar g_CLKS_PER_BIT - 1 ciclo de reloj para que el bit de inicio termine
64          if r_Clk_Count < g_CLKS_PER_BIT-1 then
65            r_Clk_Count <= r_Clk_Count + 1;
66            r_SM_Main   <= s_TX_Start_Bit;
67          else
68            r_Clk_Count <= 0;
69            r_SM_Main   <= s_TX_Data_Bits;
70          end if;
```

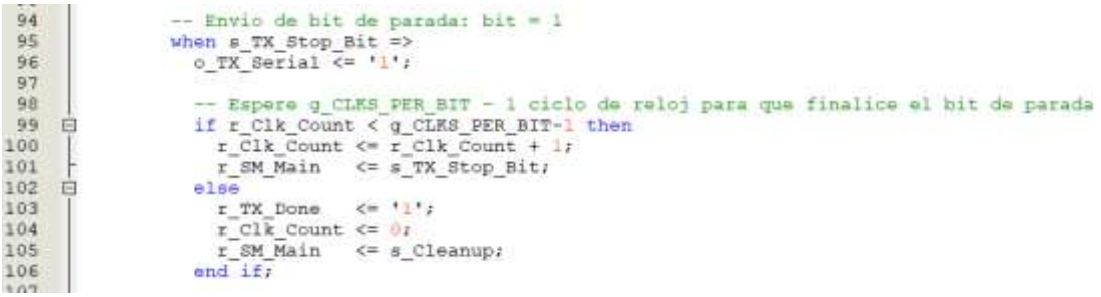
Figura 3. 22: Caso de envío de bit de inicio.
Elaborado por: Autor.

En el tercer caso de la sentencia switch corresponde a la construcción de la trama de datos que se enviara por la línea de transmisión asignada, figura 3.23, este debe esperar así mismo que se complete el muestreo para terminar, además se encuentra en el código dos sentencias if anidadas, la primera que realizar este proceso y la última permite comprobar si se han enviado todos los bits correspondientes al mensaje.

```
73          -- Espere g_CLKS_PER_BIT - 1 ciclo de reloj para que los bits de datos terminen
74        when s_TX_Data_Bits =>
75          o_TX_Serial <= r_TX_Data(r_Bit_Index);
76
77          if r_Clk_Count < g_CLKS_PER_BIT-1 then
78            r_Clk_Count <= r_Clk_Count + 1;
79            r_SM_Main   <= s_TX_Data_Bits;
80          else
81            r_Clk_Count <= 0;
82
83            -- Comprueba si se ha enviado todos los bits
84            if r_Bit_Index < 7 then
85              r_Bit_Index <= r_Bit_Index + 1;
86              r_SM_Main   <= s_TX_Data_Bits;
87            else
88              r_Bit_Index <= 0;
89              r_SM_Main   <= s_TX_Stop_Bit;
90            end if;
91          end if;
92
```

Figura 3. 23: Tercer caso de sentencia switch.
Elaborado por: Autor.

Para el cuarto caso de la sentencia de control principal, comprueba si la salida del serial está en flanco 1 para poder ejecutar la construcción del bit de parada, este espera el muestreo para finalizar, en la figura 3.24 se muestra dicho proceso y el cual asigna los valores a las salidas y señales correspondientes de la arquitectura del hardware.



```
94          -- Envio de bit de parada: bit = 1
95          when s_TX_Stop_Bit =>
96            o_TX_Serial <= '1';
97
98            -- Espere g_CLKS_PER_BIT - 1 ciclo de reloj para que finalice el bit de parada
99            if r_Clk_Count < g_CLKS_PER_BIT-1 then
100             r_Clk_Count <= r_Clk_Count + 1;
101             r_SM_Main   <= s_TX_Stop_Bit;
102           else
103             r_TX_Done   <= '1';
104             r_Clk_Count <= 0;
105             r_SM_Main   <= s_Cleanup;
106           end if;
```

Figura 3. 24: Cuarto caso: bit de parada.
Elaborado por: Autor.

Como última parte del código, figura 3.25, se muestra el caso que permite mantener el reloj en uno con un tipo de estado de t_SM_Main, siendo s_Cleanup de acuerdo a los estados de tres señales importantes para el hardware, además muestra para otros casos siempre r_SM_Main será s_Idle, finalizando los casos, sentencias if y el proceso principal.

```
109         -- Mantiene reloj en 1
110         when s_Cleanup =>
111           o_TX_Active <= '0';
112           r_TX_Done   <= '1';
113           r_SM_Main   <= s_Idle;
114
115
116         when others =>
117           r_SM_Main <= s_Idle;
118
119       end case;
120     end if;
121
122
123   end if;
124 end process p_UART_TX;
125
126 o_TX_Done <= r_TX_Done;
127
128 end RTL;
```

Figura 3. 25: Último caso y finalización de proceso.
Elaborado por: Autor.

El resultado de compilación del código se muestra en la figura 3.26, permitiendo observar la utilización de los recursos del fpga altera con un total de 48 elementos lógicos utilizados que equivale a menos del 1% disponibles, total de registros 32 y 14 pines.

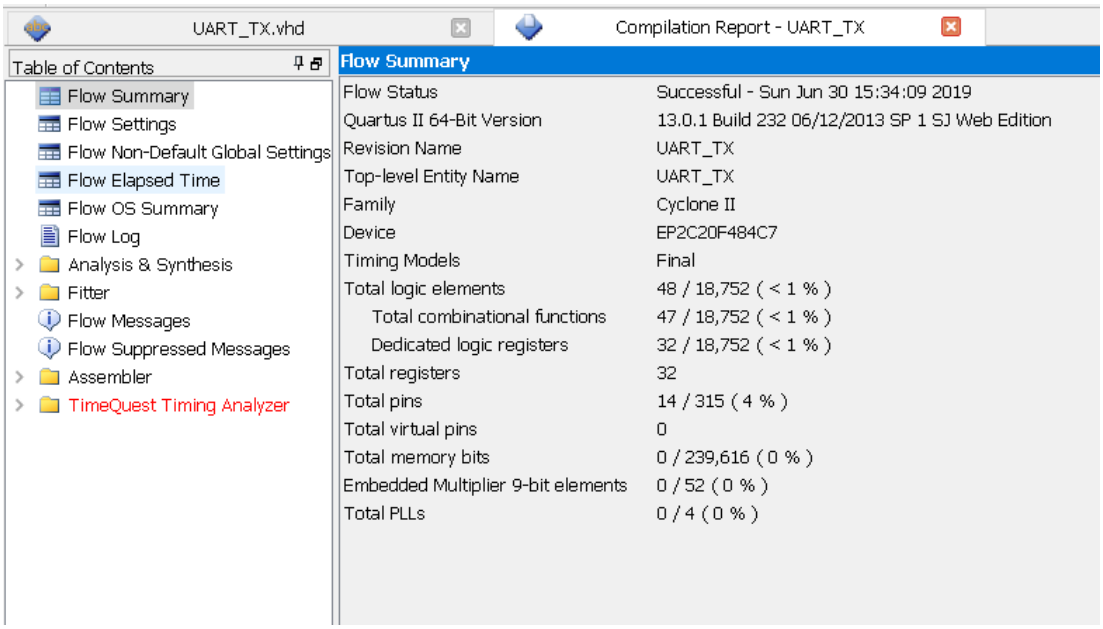


Figura 3. 26: Resultado de compilación de UART_TX.
Elaborado por: Autor.

Posterior a la compilación es posible obtener el diagrama de estado, figura 3.27, permitiendo así a la observación de estos y comprobar su correcto funcionamiento de acuerdo a lo que se diseñó previamente esperando un resultado positivo.

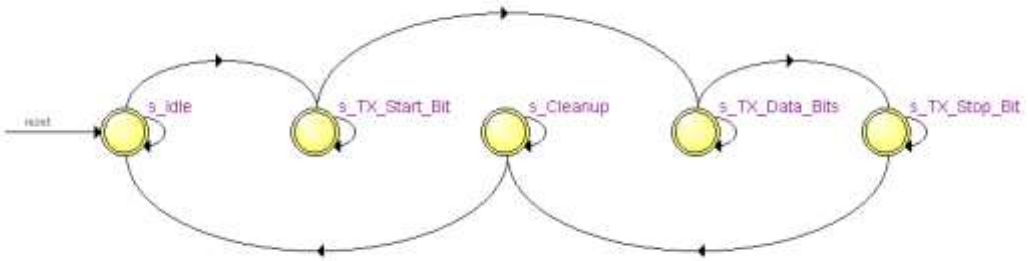


Figura 3. 27: Diagrama de estados de UART_TX.
Elaborado por: Autor.

A continuación, se procede a crear el proyecto para la recepción de mensajes por UART basado en el diagrama de flujo diseñado en la sección anterior.

En la figura 3.28 se muestra el encabezado y entidad del programa realizado para la recepción del sistema de comunicación, este detalla la librería utilizada como en el anterior caso, además de la entrada del reloj y serial, salida de Dv y un arreglo para mostrar los 8 bits de la trama recibida.

```vhdl
library ieee;
use ieee.std_logic_1164.ALL;
use ieee.numeric_std.all;

entity Uart_rx is
  generic (
    g_CLKS_PER_BIT : integer := 5208     -- Muestreo
    );
  port (
    i_Clk       : in  std_logic;
    i_RX_Serial : in  std_logic;
    o_RX_DV     : out std_logic;
    o_RX_Byte   : out std_logic_vector(7 downto 0)
    );
end Uart_rx;

```

Figura 3. 28: Encabezado y entidad para Uart_rx en VHDL.
Elaborado por: Autor.

En la arquitectura se describe los tipos de señales que se tendrá para este caso, en la figura 3.29 se aprecia y dato type que hereda los estados de las señales posteriores, además de un contenedor para guardar los estados del resultado del muestreo de la trama.

```vhdl
architecture rtl of Uart_rx is

  type t_SM_Main is (s_Idle, s_RX_Start_Bit, s_RX_Data_Bits,
                     s_RX_Stop_Bit, s_Cleanup);
  signal r_SM_Main : t_SM_Main := s_Idle;

  signal r_RX_Data_R : std_logic := '0';
  signal r_RX_Data   : std_logic := '0';

  signal r_Clk_Count : integer range 0 to g_CLKS_PER_BIT-1 := 0;
  signal r_Bit_Index : integer range 0 to 7 := 0;  -- 8 Bits Total
  signal r_RX_Byte   : std_logic_vector(7 downto 0) := (others => '0');
  signal r_RX_DV     : std_logic := '0';

```

Figura 3. 29: Arquitectura de Uart_rx en VHDL.
Elaborado por: Autor.

Como en el proyecto de transmisión se mostró de forma resumida el contenido del código, en este caso será igual, en la figura 3.30 se presenta al lector la estructura principal de los dos procesos donde el primero es para el muestreo de los puertos de entrada de rx (p_Sample) y el último proceso (p_UART_RX) posee la sentencia if que determina el flanco del reloj y contendrá los casos del switch para los diferentes tipos de estados a ejecutar.

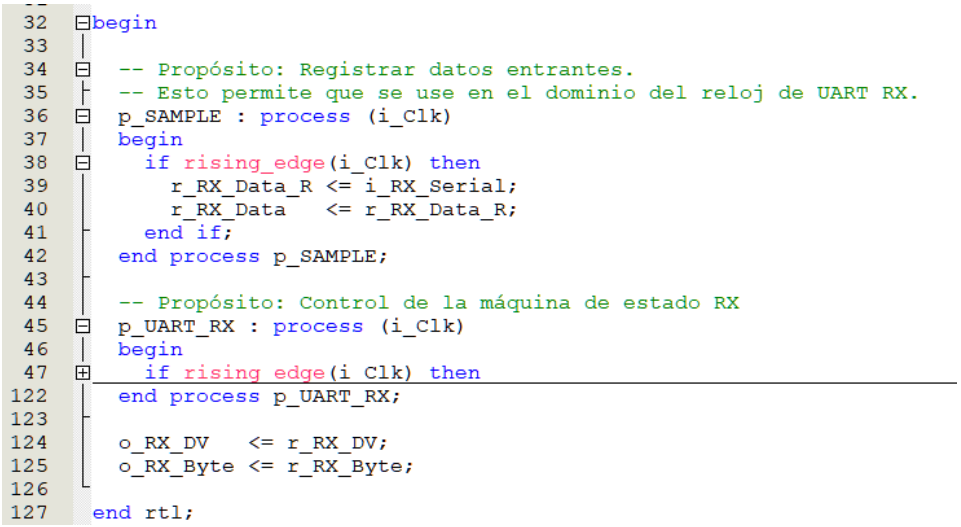


Figura 3. 30: Estructura del código Uart_rx.
Elaborado por: Autor.

El proceso p_Sample como se indicó en la figura anterior contiene una sentencia if que determina el flanco del reloj y asigna los valores de entrada del serial y datos a otras variables manejables por software.

Posteriormente en el proceso p_UART_RX inicia el if de control de flanco del reloj y dentro contiende los cinco casos, el primero como se puede apreciar en la figura 3.31 ayuda a la detección del flanco de bajada para el bit de inicio de la trama.

```
44   -- Propósito: Control de la máquina de estado RX
45   p_UART_RX : process (i_Clk)
46   begin
47     if rising_edge(i_Clk) then
48
49       case r_SM_Main is
50
51         when s_Idle =>
52           r_RX_DV     <= '0';
53           r_Clk_Count <= 0;
54           r_Bit_Index <= 0;
55
56           if r_RX_Data = '0' then       -- detectar el bit de inicio
57             r_SM_Main <= s_RX_Start_Bit;
58           else
59             r_SM_Main <= s_Idle;
60           end if;
```

Figura 3. 31: Proceso p_UART_TX inicio de flanco de bajada para bit de inicio de la trama.
Elaborado por: Autor.

El siguiente caso es para asegurarse que el bit de inicio está activo, posteriormente se reinicia el contador para colocar la variable clk_count en cero, en la figura 3.32 se muestran los if anidados dentro del caso en cuestión.

```
63          -- Compruebe bit de inicio para asegurarse de que todavia está bajo
64        when s_RX_Start_Bit =>
65          if r_Clk_Count = (g_CLKS_PER_BIT-1)/2 then
66            if r_RX_Data = '0' then
67              r_Clk_Count <= 0;  -- reinicio del contador
68              r_SM_Main   <= s_RX_Data_Bits;
69            else
70              r_SM_Main   <= s_Idle;
71            end if;
72          else
73            r_Clk_Count <= r_Clk_Count + 1;
74            r_SM_Main   <= s_RX_Start_Bit;
75          end if;
```

Figura 3. 32: Comprobación de bit de inicio y reinicio del contador.
Elaborado por: Autor.

A continuación, figura 3.33, en el siguiente estado como en el caso anterior cuando se trataba de enviar los datos, era necesario la espera del muestreo por parte del bit de reloj, en este caso se hace lo mismo, pero para confirmar los datos contenidos en la trama y confirmación de estos.

```
78          -- Espere g_CLKS_PER_BIT - 1 ciclo de reloj para muestrear datos en serie
79        when s_RX_Data_Bits =>
80          if r_Clk_Count < g_CLKS_PER_BIT-1 then
81            r_Clk_Count <= r_Clk_Count + 1;
82            r_SM_Main   <= s_RX_Data_Bits;
83          else
84            r_Clk_Count             <= 0;
85            r_RX_Byte(r_Bit_Index) <= r_RX_Data;
86
87            -- Compruebe si hemos enviado todos los bits
88            if r_Bit_Index < 7 then
89              r_Bit_Index <= r_Bit_Index + 1;
90              r_SM_Main   <= s_RX_Data_Bits;
91            else
92              r_Bit_Index <= 0;
93              r_SM_Main   <= s_RX_Stop_Bit;
94            end if;
95          end if;
```

Figura 3. 33: Inicio de función para recibir y comprobación de datos de la trama.
Elaborado por: Autor.

Posteriormente el momento de recibir los datos en serie, es necesario la espera del bit de parada, para ello se construye el siguiente estado como se mostró en el diagrama de flujo en el diseño del sistema de comunicación, este recibe el bit de parada y espera la cantidad del muestreo menos uno para finalizarlo, figura 3.34.

```
98          -- Recibir el bit de parada. Bit de parada = 1
99        when s_RX_Stop_Bit =>
100         -- Espere g_CLKS_PER_BIT - 1 ciclo de reloj para que finalice el bit de parada
101         if r_Clk_Count < g_CLKS_PER_BIT-1 then
102           r_Clk_Count <= r_Clk_Count + 1;
103           r_SM_Main   <= s_RX_Stop_Bit;
104         else
105           r_RX_DV     <= '1';
106           r_Clk_Count <= 0;
107           r_SM_Main   <= s_Cleanup;
108         end if;
```

Figura 3. 34: Espera de bit de parada y finalización.
Elaborado por: Autor.

Por ultimo antes de la compilación se construye la función del estado de cleanup, este mantiene el reloj en 1, y termina los casos y procesos, en la figura 3.35 se muestra lo descrito.

```
111         -- Mantiene reloj en 1
112         when s_Cleanup =>
113           r_SM_Main <= s_Idle;
114           r_RX_DV   <= '0';
115
116
117         when others =>
118           r_SM_Main <= s_Idle;
119
120       end case;
121     end if;
122   end process p_UART_RX;
123
124   o_RX_DV   <= r_RX_DV;
125   o_RX_Byte <= r_RX_Byte;
126
127 end rtl;
```

Figura 3. 35: Último caso y finalización del proceso.
Elaborado por: Autor.

Los siguientes resultados están dados después de la compilación del proyecto, en la figura 3.36 se muestra la cantidad de elementos lógicos utilizados que para este se utiliza 62 que al igual que el proyecto anterior es menos del 1% del total de recursos disponibles por la FPGA.

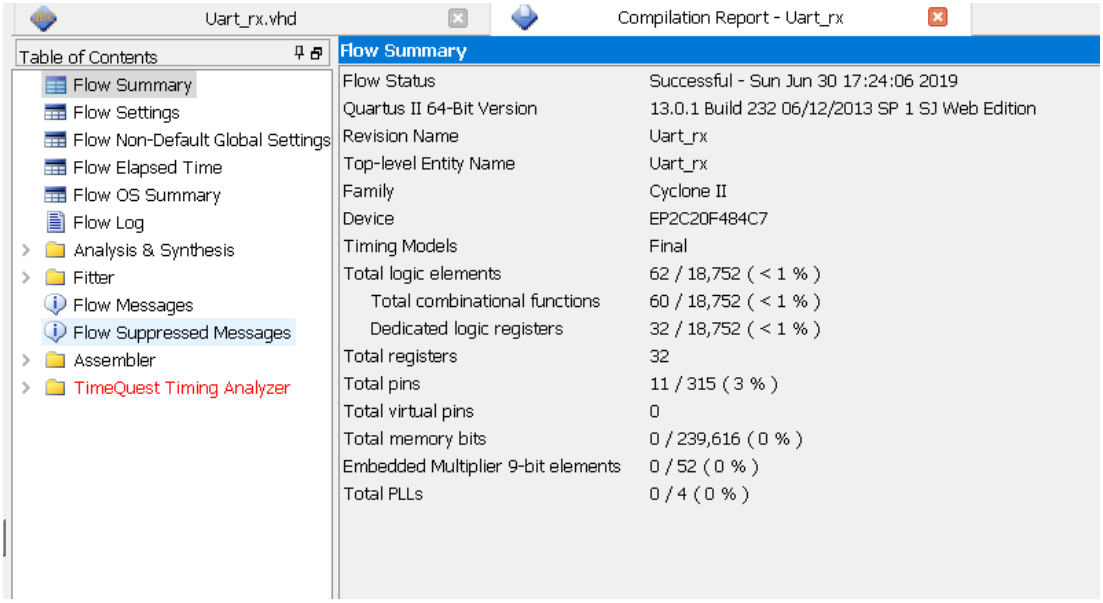


Figura 3. 36: Resultado de compilación de proyecto Uart_rx.
Elaborado por: Autor.

Una vez realizado el proyecto, se adquiere el diagrama de estados para comprobación de su lógica, en la siguiente sección se muestra su implementación en la tarjeta y sus resultados, figura 3.37.

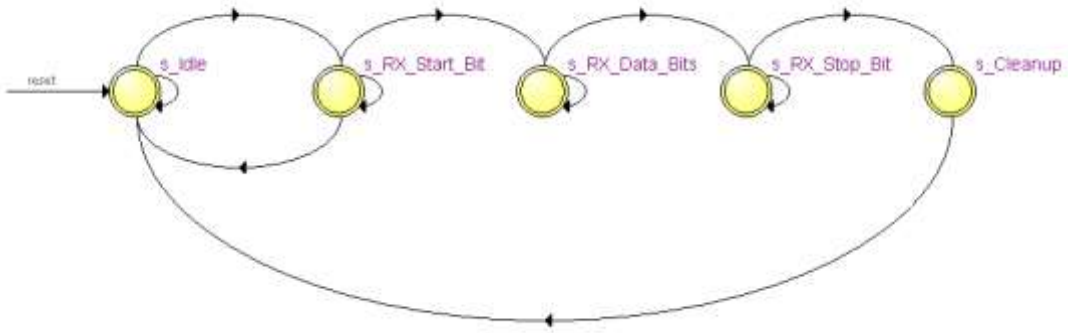


Figura 3. 37: Diagrama de estados para Uart_rx.
Elaborado por: Autor.

3.4. Implementación y resultados obtenidos del sistema de comunicación con FPGA.

En esta última sección de este capítulo se presenta al lector los resultados y cómo fue posible la implementación físicamente en la tarjeta FPGA altera De1, configurando el pin planner y con herramientas para visualizar las tramas creadas por mí.

En la figura 3.38 se presenta los pines seleccionados para el proyecto de transmisión uart (Uart_TX), como se puede apreciar los pines de entrada estará dado por los interruptores que posee la placa integrada dando el valor de los bits que se transmitirá, cabe recalcar que estos al momento de ser recibidos deberán dar un carácter de acuerdo a la tabla de código ASCII.

Adicionalmente, el pin del reloj clk se asignará al cristal con mayor frecuencia integrado, para este caso es de 50MHz, y los pines de salida para serial, active, done, dv serán asignados a los puertos GPIO de expansión.

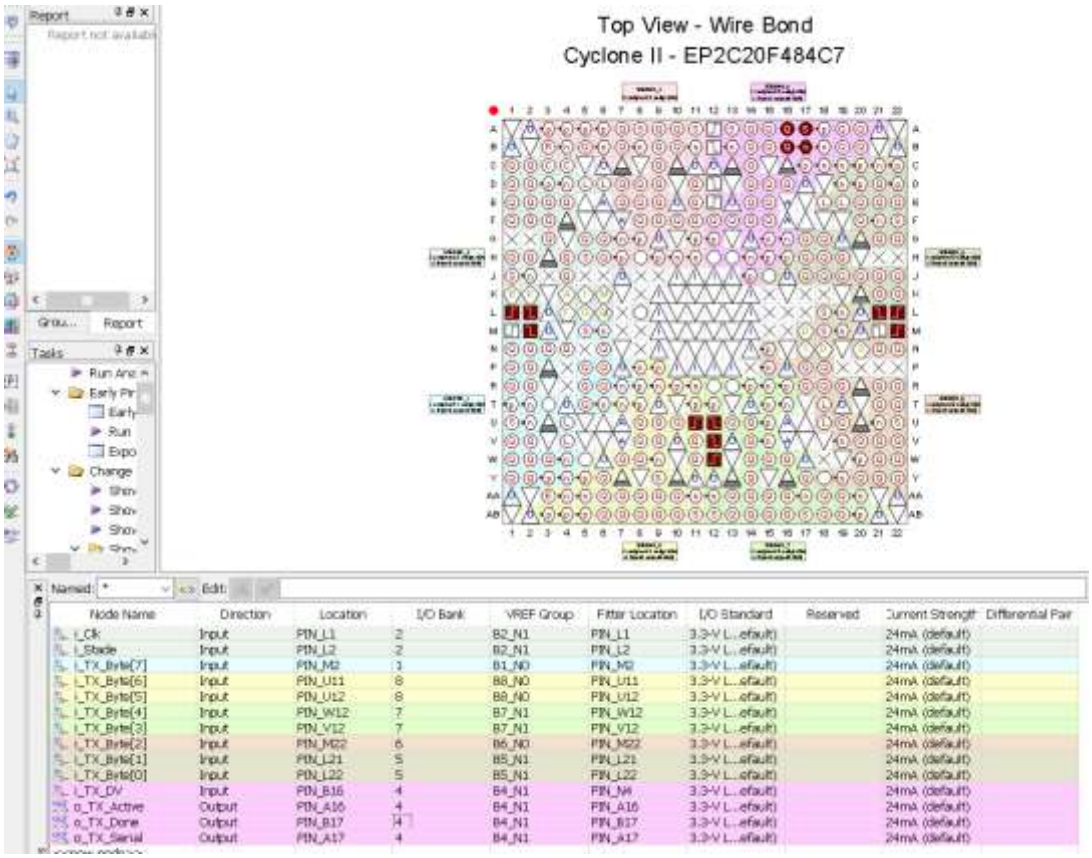


Figura 3. 38: Configuración de pin planner para proyecto de Uart_tx.
Elaborado por: Autor.

En la siguiente figura 3.39 se presenta a sí mismo la configuración del pin planner del proyecto Uart_rx que se encargara de recibir la trama y poder presentarla con luces de los leds integrados en la placa.

En este caso no cuenta con los mismos puertos de transmisión ya que el sistema inicialmente trabaja comunicando a otra Fpga y solo el medio será de cobre directo conectando los respectivos pines.

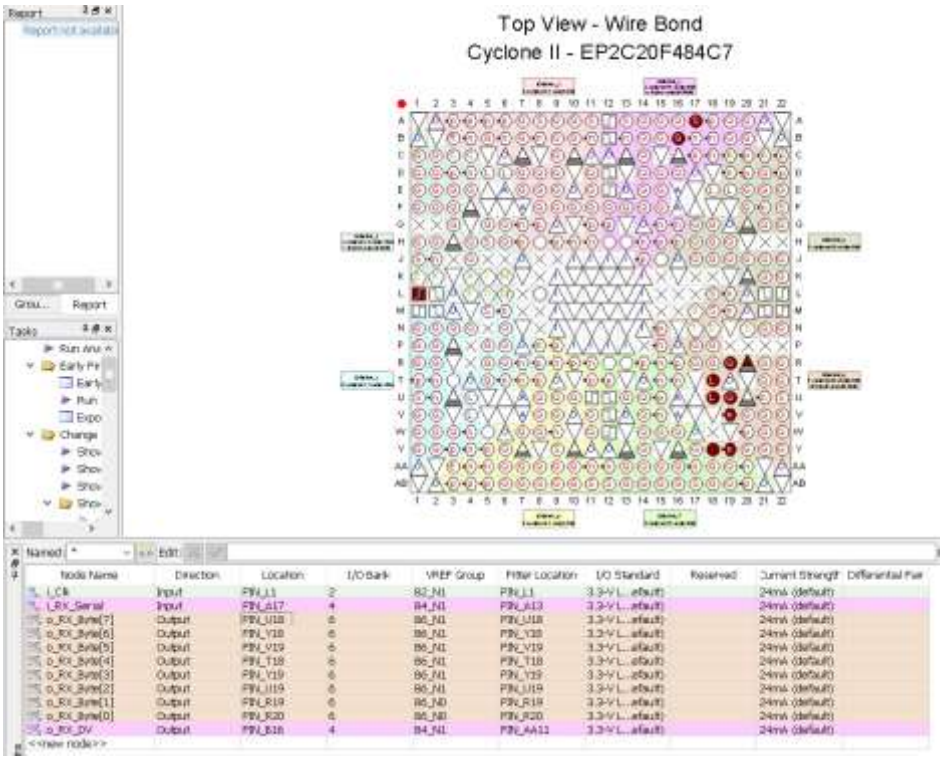


Figura 3. 39: Configuración de pin planner para proyecto de Uart_rx.
Elaborado por: Autor.

Se procede a conectar ambas placas para realizar la prueba física del sistema, con un resultado positivo, en la figura 3.40 se presenta como están conectadas y su resultado.

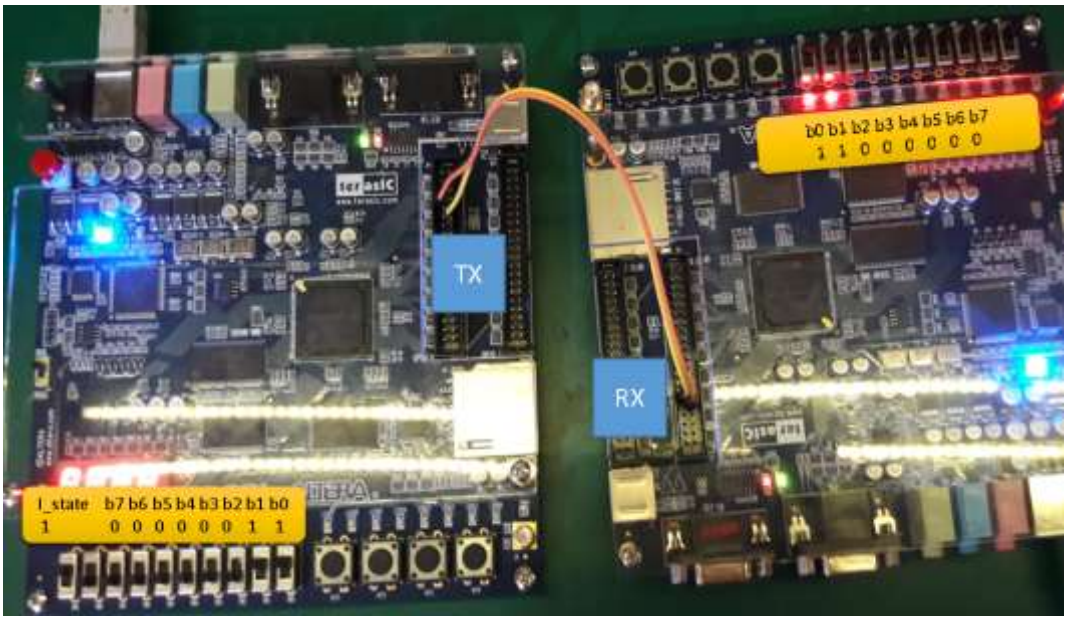


Figura 3. 40: Funcionamiento de diseño de hardware para TX y RX.
Elaborado por: Autor.

En la trama dispuesta de manera práctica se seleccionó una trama conformada por 6 bits en bajo y dos en alto representada por los interruptores

integrados, la trama seria: 00000011. Después del proceso realizado por la fpga que transmitirla por cable directo a la tarjeta que tiene la configuración uart_rx presenta el resultado en los leds rojos como se ve en la imagen, presentando los mismos bits de la trama.

A continuación, se procede a realizar pruebas convirtiendo la convirtiendo la trama en los niveles RS232 asignando nuevamente los pines correspondientes en el pin planner de tx serial en el integrado que realiza dicha tarea, para ello se conecta la tarjeta con código uart tx en el puerto correspondiente como indica en la figura 3.41.

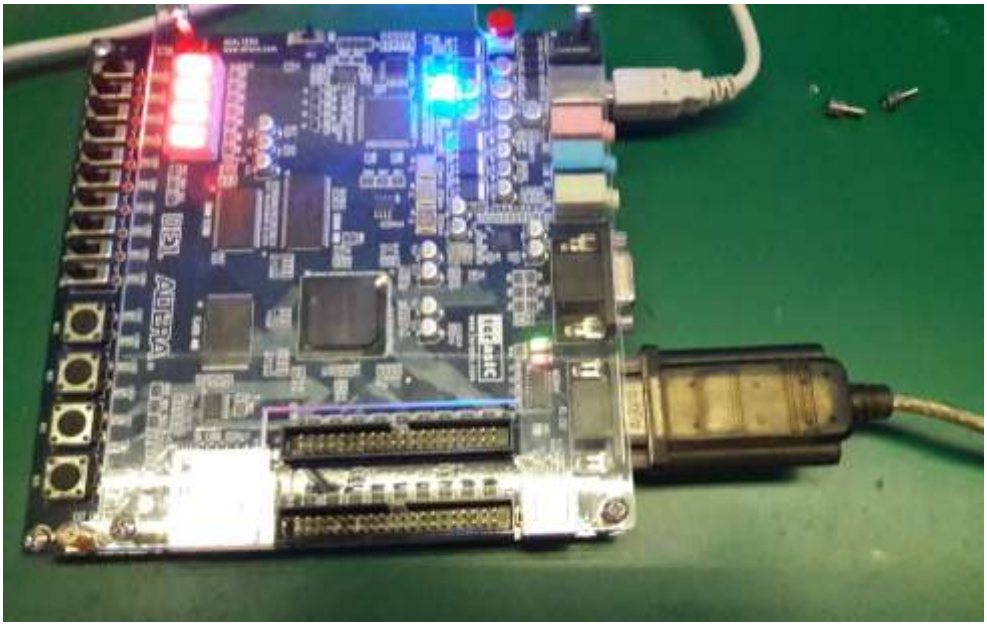


Figura 3. 41: Conexión de fpga altera puerto RS232 a pc.
Elaborado por: Autor.

Se realizaron varias pruebas basadas en la tabla que se presenta en la figura 3.42 del código ascii para determinar los bits que deben ser activados para recibir determinado carácter, para ello se utilizó también el software hercules que a continuación se detallara.

Binario	Dec	Hex	Representación
0010 0000	32	20	espacio ( )
0010 0001	33	21	!
0010 0010	34	22	"
0010 0011	35	23	#
0010 0100	36	24	$
0010 0101	37	25	%
0010 0110	38	26	&
0010 0111	39	27	'
0010 1000	40	28	(
0010 1001	41	29	)
0010 1010	42	2A	*
0010 1011	43	2B	+
0010 1100	44	2C	,
0010 1101	45	2D	-
0010 1110	46	2E	.
0010 1111	47	2F	/
0011 0000	48	30	0
0011 0001	49	31	1
0011 0010	50	32	2
0011 0011	51	33	3
0011 0100	52	34	4
0011 0101	53	35	5
0011 0110	54	36	6
0011 0111	55	37	7
0011 1000	56	38	8
0011 1001	57	39	9
0011 1010	58	3A	:
0011 1011	59	3B	;
0011 1100	60	3C	<
0011 1101	61	3D	=
0011 1110	62	3E	>
0011 1111	63	3F	?

Binario	Dec	Hex	Representación
0100 0000	64	40	@
0100 0001	65	41	A
0100 0010	66	42	B
0100 0011	67	43	C
0100 0100	68	44	D
0100 0101	69	45	E
0100 0110	70	46	F
0100 0111	71	47	G
0100 1000	72	48	H
0100 1001	73	49	I
0100 1010	74	4A	J
0100 1011	75	4B	K
0100 1100	76	4C	L
0100 1101	77	4D	M
0100 1110	78	4E	N
0100 1111	79	4F	O
0101 0000	80	50	P
0101 0001	81	51	Q
0101 0010	82	52	R
0101 0011	83	53	S
0101 0100	84	54	T
0101 0101	85	55	U
0101 0110	86	56	V
0101 0111	87	57	W
0101 1000	88	58	X
0101 1001	89	59	Y
0101 1010	90	5A	Z
0101 1011	91	5B	[
0101 1100	92	5C	\
0101 1101	93	5D	]
0101 1110	94	5E	^
0101 1111	95	5F	_

Binario	Dec	Hex	Representación
0110 0000	96	60	`
0110 0001	97	61	a
0110 0010	98	62	b
0110 0011	99	63	c
0110 0100	100	64	d
0110 0101	101	65	e
0110 0110	102	66	f
0110 0111	103	67	g
0110 1000	104	68	h
0110 1001	105	69	i
0110 1010	106	6A	j
0110 1011	107	6B	k
0110 1100	108	6C	l
0110 1101	109	6D	m
0110 1110	110	6E	n
0110 1111	111	6F	o
0111 0000	112	70	p
0111 0001	113	71	q
0111 0010	114	72	r
0111 0011	115	73	s
0111 0100	116	74	t
0111 0101	117	75	u
0111 0110	118	76	v
0111 0111	119	77	w
0111 1000	120	78	x
0111 1001	121	79	y
0111 1010	122	7A	z
0111 1011	123	7B	{
0111 1100	124	7C	|
0111 1101	125	7D	}
0111 1110	126	7E	~

Figura 3. 42: Código ascii representado en binario, decimal y hexadecimal.
Elaborado por: Autor.

El resultado obtenido en el pc se presenta en la figura 3.43 donde se utilizó el software hercules para poder visualizar el contenido de la trama recibida por el puerto rs232, inicialmente se envía una trama conformada por 00110000 y luego 00110001 y según la tabla corresponde al valor cero y uno respectivamente.

Los resultados son muy buenos ya que es posible enviar una cantidad de caracteres de acuerdo al código ascii permitiendo implementar cualquier diseño electrónico y comunicarlo con otros tipos de controladores digitales.

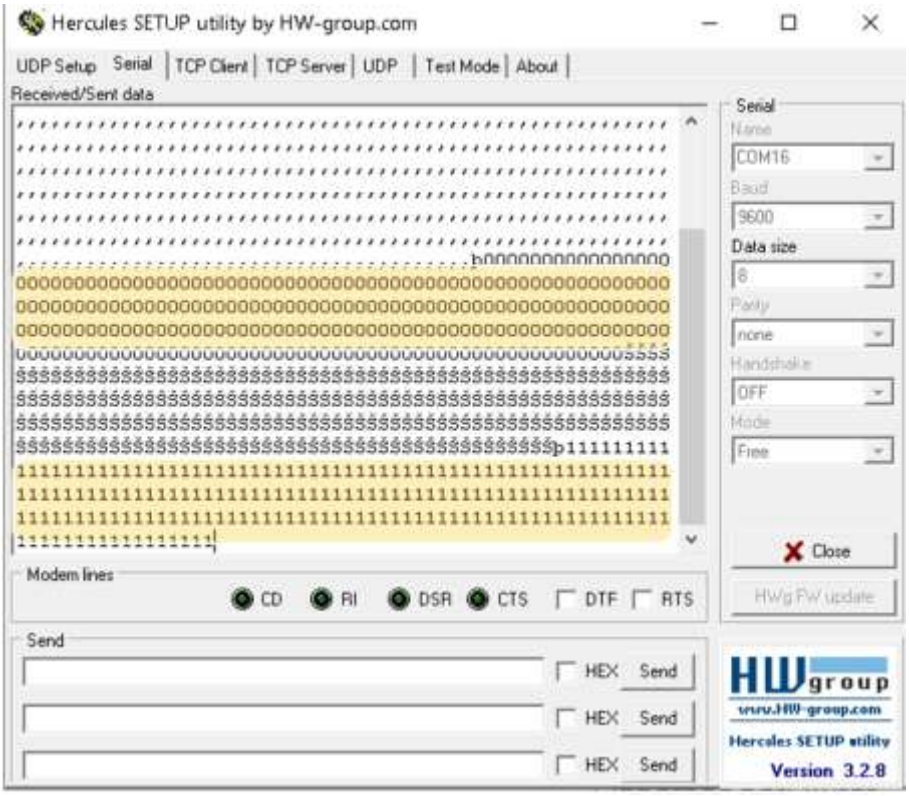


Figura 3. 43: Resultado de comunicación por RS232.
Elaborado por: Autor.

A continuación, se presenta en la figura 3.44 el resultado hecha por pruebas de conectividad inalámbrica, para ello se utilizó un módulo bluetooth HC-05 en el puerto de expansión de la fpga GPIO utilizando el pin de tx para efecto.

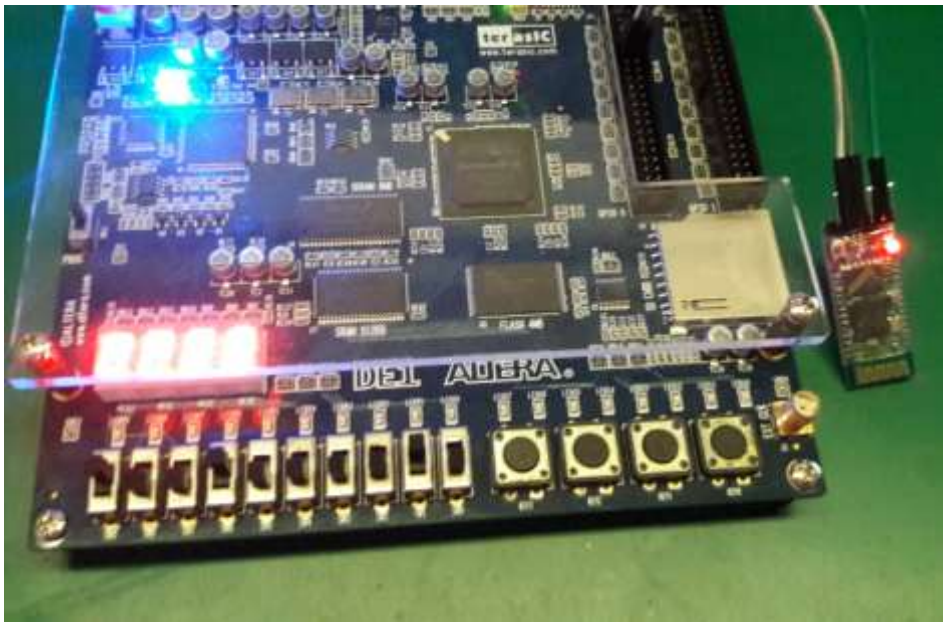


Figura 3. 44: Conectividad a modulo bluetooth HC-05.
Elaborado por: Autor.

Para visualizar el resultado de la transmisión se utilizó un teléfono Android y una aplicación que permite mostrar los datos recibidos por este puerto bluetooth, como se aprecia en la figura 3.45.

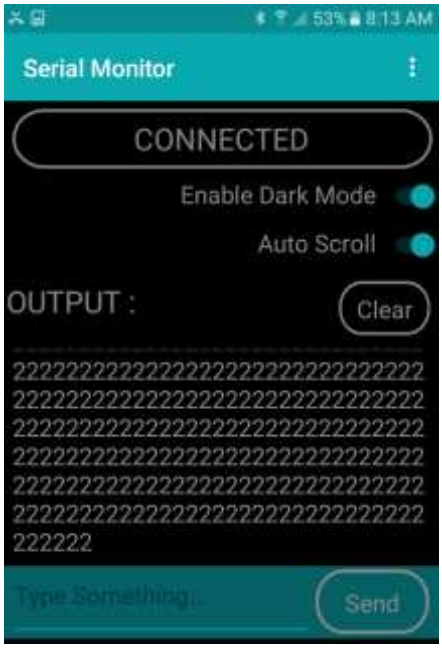


Figura 3. 45: Monitor serie bluetooth en android.
Elaborado por: Autor.

Y para finalizar las pruebas realizadas, también se efectuó la recepción de datos por parte de la tarjeta FPGA mediante el mismo puerto bluetooth, en la imagen 3.46 se muestra el resultado obtenido.

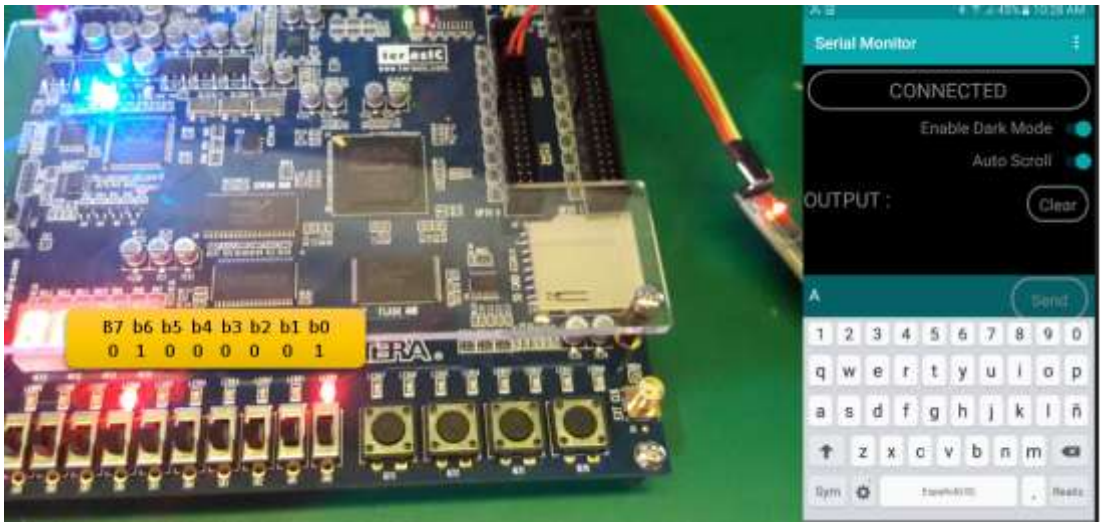


Figura 3. 46: Recepción de trama por bluetooth.
Elaborado por: Autor.

Si se basa en la tabla del código ascii la letra A mayúscula está representada por la trama en binario: 01000001, por lo tanto, se comprueba el funcionamiento del sistema de comunicación mediante el fpga altera DE1.

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES.

4.1. Conclusiones.

- Se ha cumplido con el primer objetivo específico, en la cual se describió el Estado del Arte o Fundamentación Teórica del Sistema p2p, los cuales permiten integrar dispositivos electrónicos (Hardware) y plataformas de programación (Software).

- Se pudo comprobar que la mejor herramienta o plataforma de programación es QUARTUS II de Altera, que trabaja de manera conjunta con la tarjeta FPGA DE1 de Altera mismo.

- Se pudo comprobar que el proceso de un sistema p2p, no es fácil de lograr, debido a la integración de varias herramientas, pero lo más importante fue el aprendizaje logrado y la idea de transmitir el conocimiento básico de manejar la tarjeta FPGA DE1 de Altera.

4.2. Recomendaciones.

- Si bien es cierto que se deja constancia de la entrega de los equipos utilizados en el presente trabajo de titulación, sugiero que la FETD compre más tarjetas de desarrollo FPGA para Sistemas Digitales Avanzados, y que puedan salir temas de investigación para propuestas de trabajo de graduación.

- Capacitar tanto a docentes como estudiantes en el uso de las tarjetas FPGA, y su aplicación en los programas de estudios de materias como Digitales I, Digitales II, Laboratorio de Digitales, Sistemas Microprocesador.

Bibliografía

Alcaine Sanchez, A. (2015). *Propiedad Intelectual y Redes P2P*. Recuperado de https://riunet.upv.es/handle/10251/56040

Anzures, M. (2017). *Un enfoque metodológico semántico basado en un modelo arquitectónico para el desarrollo de groupware*. Recuperado de http://digibug.ugr.es/handle/10481/48880

Argudo S. (2016). P2P: Historia, qué es y para qué sirve el protocolo peer-to-peer. Recuperado el 6 de junio de 2019, de https://www.malavida.com/es/analisis/redes-p2p-evolucion-de-un-protocolo-mas-alla-de-las-descargas-006483

Arias, P. (2015). *Diseño de una red LAN/WAN segura para el Tribunal Constitucional aplicando la metodología de 3 capas de CISCO*. Recuperado de http://repositorio.puce.edu.ec/handle/22000/6371

Chicaiza, D. (2017). Estudio de las tecnologías de Seguridad Perimetral Informáticas y Propuesta de un plan de Implementación para la Agencia Nacional de Tránsito. Recuperado el 24 de junio de 2019, de http://repositorio.puce.edu.ec/handle/22000/7896

Manchado S. (2016). *Especificación y análisis del protocolo Chord en Maude—E-Prints Complutense*. Recuperado de https://eprints.ucm.es/16753/

Medrano L. (2015). Diseño e implementación de un prototipo combinado de video vigilancia utilizando tecnología BPL, Ethernet para le Laboratorio LTI. Recuperado el 24 de junio de 2019, de https://bibdigital.epn.edu.ec/handle/15000/3797

Nicolini A. (2018). *Aplicación de mecanismos reactivos y argumentativos para la búsqueda temática en redes P2P*. Recuperado de http://repositoriodigital.uns.edu.ar/handle/123456789/4111

Santos S. (2017). *Análisis de los dispositivos de red en ambiente virtual—Repositorio Institucional USAC*. Recuperado de http://www.repositorio.usac.edu.gt/7744/

Villada L. (2016). *Exploración de arquitecturas basadas en P2P para la distribución de televisión por suscripción (IPTV)*. Recuperado de https://repository.upb.edu.co/handle/20.500.11912/2991

Yuquilema H. (2014). *Artículo Científico—Diseño e implementación de una tarjeta electrónica basada en FPGA orientada a aplicaciones didácticas en el laboratorio de VLSI.* Recuperado de http://repositorio.espe.edu.ec/jspui/handle/21000/9043

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Bohórquez Zúñiga, David Isaac** con C.C: # 092411796-3 autor del Trabajo de Titulación: **Análisis del desempeño de enlaces p2p utilizando la tarjeta FPGA De1 de Altera,** previo a la obtención del título de **INGENIERO EN TELECOMUNICACIONES** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 9 de marzo del 2018

f. _____

Nombre: Bohorquez Zuñiga, David Isaac

C.C: 092411796-3

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA			
FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN			
TÍTULO Y SUBTÍTULO:	Análisis del desempeño de enlaces p2p utilizando la tarjeta FPGA DE1 de Altera.		
AUTOR(ES)	BOHORQUEZ ZUÑIGA, DAVID ISAAC		
REVISOR(ES)/TUTOR(ES)	M. Sc. EDWIN F. PALACIOS MELÉNDEZ		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	Ingeniería en Telecomunicaciones		
TITULO OBTENIDO:	Ingeniero en Telecomunicaciones		
FECHA DE PUBLICACIÓN:	9 de marzo del 2018	**No. DE PÁGINAS:**	84
ÁREAS TEMÁTICAS:	Sistemas Microcontroladores y Comunicaciones Inalámbricas		
PALABRAS CLAVES/ KEYWORDS:	**FPGA, P2P, ALTERA, QUARTUS, TARJETA, DE1.**		

RESUMEN/ABSTRACT:

Para el presente documento se da a conocer en forma general el funcionamiento del análisis del desempeño p2p de la tarjeta FPGA DE1 de altera, Lo interesante del trabajo de titulación, fue la búsqueda de información debido a que no se ha manejado temas de titulación que involucren los análisis de desempeño, ni del manejo de la tarjeta FPGA DE1 de Altera, siendo esta muy completa con respecto a las tarjetas disponibles en el Laboratorio de Electrónica de la Carrera de Ingeniería en Telecomunicaciones de la Facultad de Educación Técnica para el Desarrollo. En el Capítulo 1, se detallan la justificación, antecedentes y definición del problema de investigación, así como también, el objetivo general, objetivos específicos, idea a defender y la metodología empleada. En el Capítulo 2, se analiza, describe y explora el Estado del Arte de la tarjeta FPGA DE1 de altera, para que el lector se familiarice con el tema. En el Capítulo 3, se valida el trabajo de titulación mediante el análisis de desempeño p2p en la que el lector podrá hacer uso del mismo en el Laboratorio de Electrónica, donde va a reposar la tarjeta DE1 de altera. En el Capítulo 4, se finaliza con las conclusiones y recomendaciones que se dan una vez concluida el trabajo de titulación.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** 0969497356	E-mail: David.bohorquez.1990@hotmail.es
CONTACTO CON LA INSTITUCIÓN: COORDINADOR DEL PROCESO DE UTE	**Nombre:** Palacios Meléndez Edwin Fernando	
	Teléfono: +593-9-68366762	
	E-mail: edwin.palacios@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	