Instituto Politécnico Nacional

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA
UNIDAD CULHUACAN

"DISEÑO DE UN MÓDULO UART EN FPGA UTILIZANDO RTL PARA LA IMPLEMENTACIÓN DE TÉCNICAS DE CIFRADO Y DESCIFRADO"

Que para obtener el grado de

"Maestro en Ingeniería En Seguridad y Tecnologías de la Información"

Presenta:

Christian Antonio Colin Cejudo

Directores:

Dr. Aldo Hernández Suarez

Dr. Gonzalo Issac Duchén-Sánchez

Mes 2025

Instituto Politécnico Nacional

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA

UNIDAD CULHUACAN

"DISEÑO DE UN MÓDULO UART EN FPGA UTILIZANDO RTL PARA LA IMPLEMENTACIÓN DE TÉCNICAS DE CIFRADO Y DESCIFRADO"

Que para obtener el grado de

"Maestro en Ingeniería En Seguridad y Tecnologías de la Información"

Presenta:

Christian Antonio Colin Cejudo

Directores:

Dr. Aldo Hernández Suarez	Dr. Gonzalo Issac Duchén-Sánchez

Presidente del Jurado	**Profesor titular**
TBD	TBD

DEDICATORIA

dedico este trabajo a y aa ....

porque ...

bla bla bla bla

bla bla

AGRADECIMIENTOS

Al CONACYT por la beca-crédito otorgada para la realización de mis estudios.

Así mismo agradezco a mis maestros, compañeros y a toda aquella persona que de alguna manera contribuyó al término de mis estudios de maestría.

Contenido

Resumen

Resumen:

La creciente demanda de seguridad informática robusta y eficiente ha impulsado la adopción de hardware especializado para operaciones criptográficas. Ante el aumento de las amenazas cibernéticas y la necesidad de procesar grandes volúmenes de datos en tiempo real, las soluciones criptográficas basadas en hardware ofrecen ventajas significativas en términos de rendimiento, resistencia a ataques y almacenamiento seguro de claves criptográficas.

Esta tesis presenta la implementación de un sistema de comunicación segura utilizando el protocolo UART (Universal Asynchronous Receiver-Transmitter) como base para un diseño a nivel de transferencia de registros (RTL) sobre una plataforma FPGA. El protocolo base fue modificado para introducir una capa adicional de seguridad a nivel de hardware. Además, se integraron técnicas criptográficas específicamente cifrado y descifrado en el diseño con el objetivo de mejorar la protección e integridad de los datos durante la transmisión. Los resultados demuestran la viabilidad de incorporar mecanismos criptográficos directamente en el hardware de comunicación, proporcionando una solución escalable y eficiente para sistemas embebidos seguros.

Palabras Clave: Hardware criptográáfico, Seguridad informática, Cifrado, Descifrado, Claves criptográficas, Firmas digitales, Autenticación, Amenazas cibernéticas, Procesamiento de datos, Protocolos de comunicación, UART (Universal Asynchronous Receiver-Transmitter),

FPGA (Field-Programmable Gate Array), Diseño RTL (Register Transfer Level), Capa de seguridad en hardware, Técnicas criptográficas, Infraestructuras críticas.

Abstract

Abstract: The increasing demand for robust and efficient information security has led to the growing adoption of specialized hardware for cryptographic operations. In response to the rise in cyber threats and the need to process large volumes of data in real time, hardware-based cryptographic solutions offer significant advantages in terms of performance, resistance to attacks, and secure storage of cryptographic keys.

This thesis presents the implementation of a secure communication system using the UART (Universal Asynchronous Receiver-Transmitter) protocol as the foundation for a Register Transfer Level (RTL) design on an FPGA platform. The base protocol was modified to introduce an additional hardware-level security layer. Furthermore, cryptographic techniques—specifically encryption and decryption—were integrated into the design to enhance data protection and integrity during transmission. The results demonstrate the feasibility of embedding cryptographic mechanisms directly into communication hardware, providing a scalable and efficient solution for secure embedded systems.

Palabras Clave: Cryptographic hardware, Information security, Encryption, Decryption, Cryptographic keys, Digital signatures, Authentication, Cyber threats, Data processing, Communication protocols, UART (Universal Asynchronous Receiver-Transmitter), FPGA (Field-Programmable Gate Array), RTL design (Register Transfer Level), Hardware security layer, Cryptographic techniques, Critical infrastructure.

Índice de figuras

Índice de figuras

Introducción

En la actualidad, la seguridad de la información se ha convertido en un pilar fundamental para el desarrollo de sistemas digitales confiables. La creciente sofisticación de las amenazas cibernéticas, sumada al incremento exponencial en la generación y transmisión de datos, exige soluciones cada vez más robustas y eficientes. En este contexto, el uso de hardware especializado para operaciones criptográficas ha emergido como una alternativa eficaz frente a las limitaciones del procesamiento criptográfico basado exclusivamente en software. El hardware criptográfico ofrece ventajas significativas, como mayor rendimiento, menor latencia, menor consumo energético y una mayor resistencia a ataques físicos y lógicos. Estas características lo convierten en una solución ideal para sistemas embebidos, dispositivos IoT, aplicaciones industriales y entornos donde la seguridad y la eficiencia operativa son prioritarias. Los arreglos de compuertas programables en campo, conocidos como FPGAs (Field Programmable Gate Arrays), se han convertido en una de las tecnologías más versátiles para el diseño de sistemas electrónicos. Estos dispositivos ofrecen una solución rentable especialmente para producciones de bajo volumen, ya que el costo inicial para obtener un prototipo es considerablemente menor que el de los circuitos integrados de aplicación específica, conocidos como ASIC (Application-Specific Integrated Circuit). Además, una ventaja significativa es su capacidad de reconfiguración durante la operación, lo cual permite que un solo dispositivo pueda desempeñar diversas funciones definidas previamente, optimizando espacio y reduciendo costos. Hoy

en día, los sistemas electrónicos están presentes en casi todos los aspectos de la vida cotidiana, desde productos de consumo hasta sistemas de control industrial, aplicaciones automotrices, de seguridad, y muchas más. La tendencia actual en diseño electrónico se caracteriza por la creciente complejidad de sus componentes, lo que demanda soluciones que sean fáciles de usar, versátiles, de bajo consumo y que lleguen rápidamente al mercado. La tecnología moderna en circuitos integrados hace posible integrar estos sistemas complejos en dimensiones muy reducidas; a estos se les denomina sistemas embebidos o "System on Chip"(SoC, por sus siglas en inglés). Dichos sistemas, diseñados para cumplir funciones específicas, combinan hardware y software creados específicamente para cada tarea. En este contexto, las FPGAs, gracias a su capacidad de reconfiguración, son herramientas valiosas para el desarrollo de prototipos o series pequeñas a costos accesibles. Entre los protocolos de comunicación más utilizados en sistemas embebidos se encuentra el UART (Universal Asynchronous Receiver-Transmitter), gracias a su simplicidad, bajo costo de implementación y amplia compatibilidad. Sin embargo, dicho protocolo carece de mecanismos nativos de seguridad, lo que lo hace vulnerable a interceptaciones, manipulaciones y accesos no autorizados. En este trabajo de tesis se propone el diseño e implementación de una capa de seguridad criptográfica sobre el protocolo UART, utilizando un enfoque a nivel de transferencia de registros (RTL) en una plataforma FPGA implementando un lenguaje de descripción de hardware por sus siglas en Ingles (HDL) utilizado para modelar y diseñar circuitos digitales. La modificación del protocolo permite no solo asegurar la transmisión de datos, sino también integrar técnicas de cifrado y descifrado directamente en el hardware. Esta solución busca demostrar la viabilidad de incorporar seguridad criptográfica eficiente en sistemas de comunicación embebidos, sin comprometer el rendimiento ni la escalabilidad.

Definición del problema

En la actualidad, la transmisión de datos en sistemas embebidos se realiza, en gran medida, a través de protocolos de comunicación simples y eficientes como el UART (Universal Asynchronous Receiver-Transmitter). Sin embargo, uno de los principales inconvenientes de este protocolo es la ausencia de mecanismos nativos de seguridad, lo que lo expone a una variedad de ataques como la intercepción de datos (sniffing), la manipulación de tramas, la inyección de comandos maliciosos y la suplantación de dispositivos. Este problema se agrava en aplicaciones donde el UART se emplea en contextos críticos, tales como sistemas industriales, redes de sensores, dispositivos médicos o entornos IoT, en los que la confidencialidad y la integridad de los datos son fundamentales. Aunque existen soluciones basadas en software para implementar cifrado, estas suelen generar una sobrecarga computacional considerable en microcontroladores de recursos limitados, afectando el rendimiento y la eficiencia del sistema. Por tanto, se identifica la necesidad de una solución que permita asegurar la transmisión de datos a través del protocolo UART sin comprometer el rendimiento, preferentemente mediante una implementación en hardware que garantice mayor robustez y eficiencia.

Justificación

El avance acelerado de la tecnología ha incrementado la interconexión entre dispositivos, especialmente en sistemas embebidos e infraestructura crítica. Esta conectividad expone a los sistemas a múltiples vectores de ataque que pueden comprometer la integridad, confidencialidad y disponibilidad de la información. Dado que UART es un protocolo ampliamente utilizado en microcontroladores, sensores y módulos de comunicación, fortalecer su seguridad representa una mejora significativa en la protección de sistemas donde el reemplazo por protocolos más complejos no es viable por razones de costo o recursos. El uso de FPGAs como plataforma de implementación permite un diseño flexible, reconfigurable y altamente optimizado. Además, al integrar funciones criptográficas directamente en el hardware, se reduce la dependencia del software y se incrementa la resistencia a ataques de canal lateral, lo que incrementa el nivel de seguridad general del sistema. Esta tesis, por tanto, contribuye con una propuesta de solución viable, eficiente y escalable para la seguridad de comunicaciones en sistemas embebidos, atendiendo a una necesidad real en la industria tecnológica actual.

Objetivo

Objetivo General

Diseñar e implementar una capa de seguridad criptográfica en el protocolo UART mediante un diseño a nivel RTL en FPGA, que integre técnicas de cifrado y descifrado para mejorar la seguridad en la transmisión de datos en sistemas embebidos.

Objetivos Específicos

- Analizar las vulnerabilidades del protocolo UART en contextos de comunicación embebida.
- Diseñar un módulo RTL que modifique el protocolo UART para incorporar una capa de seguridad.
- Integrar algoritmos de cifrado y descifrado simétrico en el diseño propuesto.
- Implementar el sistema en una FPGA para validar su funcionamiento y rendimiento.
- Evaluar la eficiencia y robustez del diseño frente a posibles ataques o vulnerabilidades.

Enfoque en Seguridad de la información

Antecedentes

En investigaciones previas se han propuesto mecanismos de seguridad sobre UART, pero en su mayoría dependen de software, lo que limita la eficiencia y vulnerabilidad ante ataques. Otros estudios han demostrado que la implementación de algoritmos criptográficos en FPGA, como AES en modo CTR, es factible y efectiva para mejorar la seguridad sin afectar el rendimiento significativamente.

CAPÍTULO 1

Marco Teórico Contextual

1.1. Marco Teórico Contextual

Para contextualizar adecuadamente el presente trabajo de tesis dentro del marco teórico, es necesario abordar el origen del algoritmo criptográfico estándar César, así como su evolución y aplicaciones a lo largo del tiempo. Asimismo, se explorarán las condiciones en las que surgieron los servicios y mecanismos de seguridad informática, y las formas en que estos han sido implementados en distintos entornos. Este análisis permite delimitar la problemática que se busca atender mediante la propuesta de implementación en hardware presentada en esta investigación. En consecuencia, se realiza una revisión del estado del arte en la disciplina, con especial énfasis en los sistemas embebidos que incorporan mecanismos de seguridad.

A través de esta revisión se busca comprender las bases que fundamentan el diseño de soluciones criptográficas en plataformas reconfigurables, como las FPGAs, y su relevancia en escenarios donde la protección de la información es crítica. Este marco teórico sienta las bases para el desarrollo metodológico y experimental abordado en los siguientes capítulos.

1.2. Fundamentos de la Criptografía clasica

La criptografía es la ciencia y arte de proteger la información mediante técnicas que transforman datos legibles (texto plano) en datos ininteligibles (texto cifrado), de modo que sólo las personas autorizadas puedan acceder a su contenido (Stallings, 2017). A lo largo de la historia, la criptografía ha evolucionado desde métodos simples de sustitución y transposición hasta complejos algoritmos basados en teoría matemática, álgebra moderna y teoría de números.

Los métodos clásicos, aunque hoy son inseguros frente a los ataques modernos, permiten comprender los conceptos fundamentales sobre el uso de claves, cifrado simétrico y análisis criptográfico. Uno de los algoritmos más representativos de la criptografía clásica es el cifrado César, también conocido como cifrado por desplazamiento.

1.3. El Cifrado César: Origen Histórico

El cifrado César recibe su nombre de Julio César, quien lo empleaba para cifrar mensajes militares. Según registros históricos, César utilizaba un desplazamiento de tres posiciones para codificar sus mensajes (Kahn, 1996). Aunque su simplicidad hoy lo convierte en un cifrado trivial, en su época ofrecía una protección básica frente a lectores no entrenados o no alfabetizados.

Este cifrado pertenece a la familia de los cifrados monoalfabéticos por sustitución, donde cada letra del alfabeto es reemplazada por otra, según una regla fija determinada por una clave de desplazamiento.

1.4. Funcionamiento del Algoritmo

Desde un punto de vista matemátoco, el cifrado César puede representarse mediante la siguiente función:

$$C(x) = (x + k) mod n$$

Donde:

- $C(x)$ es el carácter cifrado.
- x es el índice del carácter en el alfabeto $A = 0, B = 1, ..., Z = 25$
- k es la clave de desplazamiento.
- n el es número total de caracteres del alfabeto, usualmente $n = 26$ en el alfabeto latino.

El descifrado consiste en aplicar la operación inversa

$$P(x) = (x - k) mod n$$

Por ejemplo, si se cifra la palabra ***"HOLA"*** con un desplazamiento de $k = 3$ se obtiene ***"KROD"*** . Para descifrar, se aplica el desplazamiento inverso.

1.5. Seguridad y Vulnerabilidades

El cifrado César es vulnerable a varios tipos de ataques criptográficos:

- Ataque por fuerza bruta: Dado que existen solamente 25 posibles claves (excluyendo el desplazamiento nulo), un atacante puede probar todas las combinaciones en poco tiempo.
- Análisis de frecuencia: Cada idioma tiene una distribución característica de letras. En español, por las letras "E", "A" y "O" son las más comunes. Si se cifra un texto suficientemente largo, estas frecuencias se preservan, permitiendo identificar el desplazamiento utilizado.

Estas debilidades lo hacen inapropiado para cualquier uso moderno que requiera confidencialidad real, pero útil como herramienta de desarrollo.

1.6. Utilidad en el Diseño de Sistemas Criptográficos

A pesar de sus limitaciones, el cifrado César es frecuentemente utilizado en el diseño de sistemas criptográficos hardware/software con fines educativos y experimentales. Su estructura simple lo convierte en una opción ideal para:

- Introducir técnicas de procesamiento secuencial de datos.
- Implementar máquinas de estados finitos para codificación y decodificación.
- Evaluar el consumo de recursos lógicos y tiempos de propagación en plataformas como FPGAs.
- Comparar el comportamiento de cifrados más complejos respecto a algoritmos básicos en entornos de bajo nivel (VHDL, Verilog).

En proyectos de diseño digital con arquitectura RTL (Register Transfer Level), el cifrado César se utiliza como caso de estudio para optimizar operaciones de desplazamiento modular, codificación de caracteres ASCII y generación de claves.

CAPÍTULO 2

Antecedentes System Verilog

2.1. Evolución del diseño digital y el surgimiento de SystemVerilog

El desarrollo del diseño digital ha experimentado transformaciones significativas desde sus inicios, comenzando con el prototipado físico mediante placas de pruebas (protoboards) hasta llegar a sofisticadas plataformas de simulación y modelado asistido por computadora. Herramientas CAD (Computer-Aided Design) permitieron simular circuitos electrónicos con mayor precisión y eficiencia, aunque su uso inicial requería superar ciertas barreras de complejidad técnica.

Estas herramientas facilitaron la implementación de esquemas eléctricos y permitieron definir señales de entrada coherentes con la lógica del sistema, generando salidas fácilmente interpretables. Simuladores como SPICE se convirtieron en referentes dentro del ámbito académico e industrial, al ofrecer una modelación precisa del comportamiento eléctrico. Una vez validados los resultados, se procedía al diseño del circuito físico mediante herramientas espe-

cializadas en tarjetas de circuito impreso (PCB), como el entorno OrCAD, particularmente en su módulo PCB Layout.

Durante esta etapa, el Laboratorio de Computación Adaptable comenzó a implementar modelos de circuitos neuronales electrónicos mediante estas plataformas (Padrón, 1997). Posteriormente, ante el incremento en la complejidad de los modelos de redes neuronales, se incorporaron herramientas como MATLAB y su entorno gráfico SIMULINK, permitiendo una representación matemática directa de sistemas dinámicos. Esta evolución posibilitó una interpretación más eficaz de los resultados, tanto cuantitativa como cualitativamente (Padrón et al., 2000, pp. 338–349).

De forma paralela, la industria del hardware reconfigurable, especialmente a través de las tarjetas FPGA manufacturadas por Xilinx, permitió abordar problemas específicos mediante interfaces programables. Cada tarjeta está diseñada para ofrecer flexibilidad al usuario, quien puede programar sus componentes en lenguajes de descripción de hardware como Verilog o VHDL, e integrar diversas interfaces en función de la disponibilidad física o lógica del sistema. Esta versatilidad convirtió a los kits de desarrollo en herramientas fundamentales para el diseño de soluciones a medida, especialmente en contextos de prototipado rápido y validación funcional.

Conforme se incrementaron las necesidades de diseño a nivel de sistemas y la verificación se volvió una etapa crítica, surgió SystemVerilog como una extensión y evolución del lenguaje Verilog. SystemVerilog no solo incorporó mejoras en la descripción estructural y comportamental del hardware, sino que integró capacidades orientadas a la verificación funcional, programación orientada a objetos y modelado de sistemas complejos. Esta evolución respondió a la necesidad de unificar el flujo de diseño y verificación bajo un mismo lenguaje, facilitando la implementación de testbenches avanzados, interfaces reutilizables y entornos de verificación compatibles con metodologías modernas como UVM (Universal Verification Methodology).

En este sentido, el uso de tarjetas FPGA programadas con SystemVerilog permite combinar la precisión del diseño RTL con capacidades avanzadas de verificación, lo cual resulta esencial en entornos de desarrollo donde se busca garantizar funcionalidad, rendimiento y confiabilidad

desde las etapas tempranas del proyecto. La historia y evolución de este lenguaje refleja una tendencia hacia la consolidación de herramientas que integren diseño, simulación, verificación y síntesis dentro de un mismo entorno de desarrollo.

Cronología relevante del desarrollo de SystemVerilog

- 1984: Se introduce el lenguaje Verilog como herramienta para descripción de hardware por Gateway Design Automation, posteriormente adquirido por Cadence (IEEE, 2008).
- 1990s: Verilog se convierte en uno de los estándares principales para diseño digital, ampliamente utilizado en la industria electrónica.
- 1999: Se inicia el desarrollo de SystemVerilog como una extensión de Verilog para abordar las limitaciones en verificación y modelado (Accellera Systems Initiative, 2002).
- 2002: SystemVerilog es adoptado formalmente por Accellera como estándar para diseño y verificación.
- 2005: SystemVerilog es estandarizado por IEEE como el estándar IEEE 1800-2005, consolidando su uso en la industria (IEEE, 2005).
- 2012: Se publica la revisión IEEE 1800-2012, incorporando mejoras en síntesis y verificación.
- 2017: Nueva revisión IEEE 1800-2017 con ampliaciones en características para diseño y verificación de sistemas complejos.

2.2. Lenguajes de Descripción de Hardware (HDL) y su Aplicación en el Diseño Digital

Los lenguajes de descripción de hardware (HDL, por sus siglas en inglés) surgieron como respuesta a la necesidad de los diseñadores digitales de contar con herramientas formales que permitieran especificar, modelar y verificar sistemas digitales de forma estructurada y

en distintos niveles de abstracción. Estos lenguajes no solo facilitan la comunicación entre diseñadores, sino que también permiten una interacción fluida entre las herramientas de diseño asistido por computadora (CAD) y los propios modelos digitales (Terés et al., 1998).

Los entornos de desarrollo basados en HDL integran herramientas de compilación, simulación y síntesis, lo que permite validar el comportamiento funcional de un diseño antes de su implementación física. En la actualidad, los lenguajes HDL más utilizados son VHDL y Verilog, ambos estandarizados por el IEEE (Institute of Electrical and Electronics Engineers), y ampliamente adoptados en la industria del diseño digital.

VHDL: Descripción, Aplicación y Estructura VHDL (VHSIC Hardware Description Language) fue desarrollado originalmente para documentar y verificar circuitos integrados de alta velocidad dentro del programa VHSIC del Departamento de Defensa de los EE.UU. Su sintaxis y semántica están basadas en el lenguaje ADA, lo cual le proporciona una estructura sólida, orientada a sistemas críticos y de tiempo real.

VHDL permite modelar un sistema digital a través de diferentes niveles de descripción: comportamiento, transferencia de registros (RTL) y nivel lógico estructural. Esto permite abarcar prácticamente todo el ciclo de desarrollo digital, desde las especificaciones funcionales hasta la generación del prototipo, excluyendo únicamente el trazado físico o layout.

La correcta elección del nivel de abstracción en la descripción depende del objetivo de diseño. Por ejemplo, si se busca generar una implementación física mediante herramientas de síntesis, es preferible utilizar el nivel RTL. En cambio, para validar la funcionalidad de algoritmos complejos mediante simulación, el nivel algorítmico es más adecuado (Baena, 2010).

2.3. SystemVerilog: Lenguaje de Descripción y Verificación de Hardware

SystemVerilog es un lenguaje de descripción de hardware (HDL) y verificación funcional desarrollado como una extensión del lenguaje Verilog, con el objetivo de unificar el modelado estructural, la verificación orientada a objetos y la abstracción de sistemas en un solo entorno.

Su aparición responde a la necesidad creciente de los diseñadores digitales de contar con herramientas capaces de describir, validar y verificar sistemas digitales complejos, como SoCs (System-on-Chip), en múltiples niveles de abstracción.

A diferencia de VHDL, SystemVerilog incorpora elementos sintácticos y semánticos tanto de lenguajes de descripción como de programación (inspirado en C y C++), lo que lo convierte en un lenguaje híbrido y altamente expresivo. Fue estandarizado por el IEEE como el estándar IEEE 1800, inicialmente en 2005, y ha sido adoptado ampliamente en la industria por su compatibilidad con flujos de diseño RTL y metodologías de verificación como UVM (Universal Verification Methodology).

Enfoque de Verificación en SystemVerilog Una de las mayores fortalezas de SystemVerilog es su orientación a la verificación, lo que lo diferencia fundamentalmente de lenguajes como VHDL o Verilog puro.

2.4. Comparativa entre Lenguajes de Descripción de Hardware y Desarrollo de Software

Desde sus orígenes, las distintas herramientas y entornos de diseño asistido por computadora (CAD) han empleado lenguajes y formatos específicos para representar y gestionar los diversos elementos involucrados en el diseño electrónico. Algunos de estos lenguajes consistían en notaciones explícitas utilizadas por el usuario para describir entradas a determinadas herramientas, mientras que otros correspondían a formatos intermedios internos, optimizados para el procesamiento automatizado dentro del entorno CAD y generalmente ocultos al diseñador.

Sin embargo, estas descripciones se encontraban limitadas al ecosistema particular de cada herramienta, sin ofrecer portabilidad ni estandarización. La aparición de los lenguajes de descripción de hardware (HDL, por sus siglas en inglés) supuso un avance significativo al introducir un mayor grado de estandarización y al incorporar principios propios de la ingeniería de software en la especificación y modelado de hardware digital.

Desde el punto de vista sintáctico, los HDL presentan similitudes con los lenguajes de

programación de alto nivel (HLL), lo cual facilita su aprendizaje por parte de ingenieros con experiencia previa en desarrollo de software. Por ejemplo, Verilog comparte muchas características con el lenguaje C, mientras que VHDL hereda su estructura y semántica del lenguaje ADA. Estas semejanzas, si bien pueden acelerar la adopción de HDL, pueden también inducir errores conceptuales si se emplean estrategias propias del software en contextos donde se requiere una representación precisa del comportamiento físico del hardware.

Es fundamental que, cuando un modelo HDL esté destinado a la síntesis e implementación física (por ejemplo, en FPGAs o ASICs), el diseñador adopte una mentalidad orientada al hardware, describiendo la lógica con un enfoque estructural o de transferencia de registros (RTL), y no meramente algorítmico. Por otro lado, el uso de estructuras típicas del software puede ser apropiado cuando el objetivo sea únicamente la simulación funcional del sistema, permitiendo optimizar el rendimiento en las etapas de verificación y análisis temporal.

En resumen, los HDL son lenguajes formales de alto nivel, diseñados específicamente para representar circuitos electrónicos a diversos niveles de abstracción —desde compuertas lógicas básicas hasta sistemas digitales completos— y permiten un modelado flexible, estructurado y preciso del hardware, aprovechando conceptos tanto del diseño electrónico como del desarrollo de software, tal y como se muestran en la Figura 2.1.

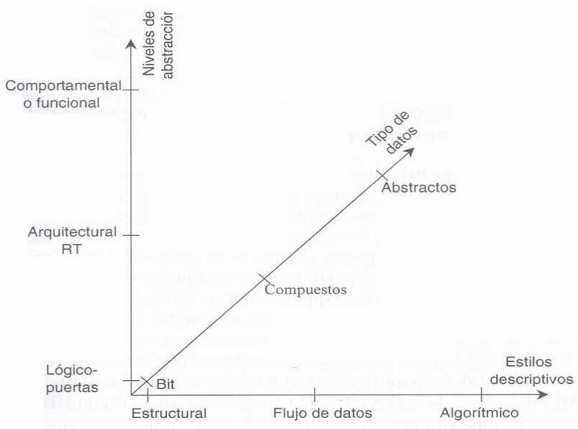

Imagen 2.1: Niveles de abstracción/precisión y estilos de modelado VHDL.

Los lenguajes de descripción de hardware (HDL) fueron desarrollados inicialmente con el propósito de modelar el comportamiento funcional de los componentes electrónicos, permitiendo su simulación previa a la implementación física. No obstante, también son ampliamente utilizados para la descripción estructural de circuitos digitales, facilitando su posterior síntesis y verificación mediante procesos de simulación validados 2.2.

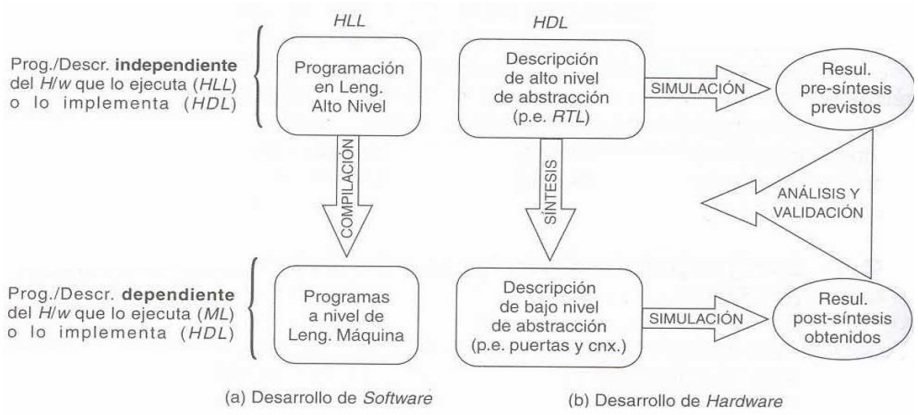

Imagen 2.2: Desarrollo en software versus hardware: (a) niveles de abstracción y lenguajes de alto nivel y (b) esquema básico del diseño descendente con HDL. .

A partir de las descripciones iniciales, los enfoques de diseño descendente (top-down) permiten aplicar procesos progresivos de síntesis que conducen gradualmente a niveles más detallados de implementación, hasta alcanzar una descripción física específica y dependiente de la tecnología utilizada. La validación en cada etapa del diseño se lleva a cabo mediante simulaciones y análisis funcionales, permitiendo iteraciones sucesivas de verificación y corrección hasta lograr un comportamiento conforme a los requisitos especificados 2.2(b).

2.5. Descripción de la tarjeta de desarollo Altera DE2-115

La tarjeta Altera DE2-115, desarrollada por Terasic Technologies, es una plataforma de prototipado basada en la FPGA Cyclone IV EP4CE115F29C7N de Altera (ahora Intel), orien-

tada a entornos educativos, de investigación y desarrollo de sistemas digitales avanzados. Esta tarjeta ofrece una arquitectura versátil que permite implementar desde diseños lógicos básicos hasta sistemas digitales complejos, incluyendo sistemas embebidos, controladores personalizados, procesamiento de señales digitales (DSP) y prototipos de SoC (System-on-Chip).

Entre sus características principales se destaca la presencia de 114,480 elementos lógicos (LEs), 3.888 Kbits de memoria RAM embebida, y 528 pines de entrada/salida de propósito general (GPIO), lo que proporciona una gran flexibilidad para interactuar con múltiples dispositivos periféricos. La tarjeta también incluye memorias externas como SDRAM de 128 MB, SRAM de 2 MB, y Flash NOR de 2 MB, además de soporte para almacenamiento mediante tarjeta SD.

La DE2-115 incorpora interfaces esenciales para el desarrollo de sistemas interactivos, tales como puertos USB, Ethernet Gigabit, salida VGA, entradas y salidas de audio, además de un conjunto de dispositivos de entrada/salida integrados como interruptores, botones, LEDs y displays de siete segmentos. Adicionalmente, cuenta con un oscilador de cristal de 50 MHz y conectores de expansión compatibles con módulos adicionales.

Esta plataforma es ampliamente utilizada en laboratorios académicos y en entornos de investigación aplicada debido a su compatibilidad con herramientas de desarrollo como Quartus Prime, facilitando la implementación de diseños en lenguajes HDL como VHDL y SystemVerilog. Su capacidad para realizar simulación, verificación y síntesis de diseños en una sola herramienta la hace ideal para proyectos de educación superior en ingeniería electrónica, mecatrónica y sistemas digitales.

Características técnicas destacadas:

FPGA: Altera Cyclone IV EP4CE115F29C7N con 114,480 elementos lógicos (LEs)

- Memoria:
  - 2 MB SRAM
  - 128 MB SDRAM
  - 2 MB Flash NOR

    - Tarjeta SD para almacenamiento externo

- Interfaces de usuario:
    - 18 interruptores (switches) y 18 botones pulsadores
    - 18 LEDs rojos y 9 LEDs verdes
    - 8 displays de 7 segmentos
- Conectividad:
    - Puertos USB tipo A y B
    - Puerto Ethernet 10/100/1000 Mbps (Gigabit)
    - Entrada/salida VGA
    - Conectores de expansión GPIO de 40 pines
- Audio y Video:
    - Entrada y salida de audio (jack de 3.5 mm)
    - Entrada de señal VGA (con ADC) y salida VGA
- Reloj: oscilador de cristal de 50 MHz
- Programación y depuración: Soporte JTAG, compatible con Quartus II

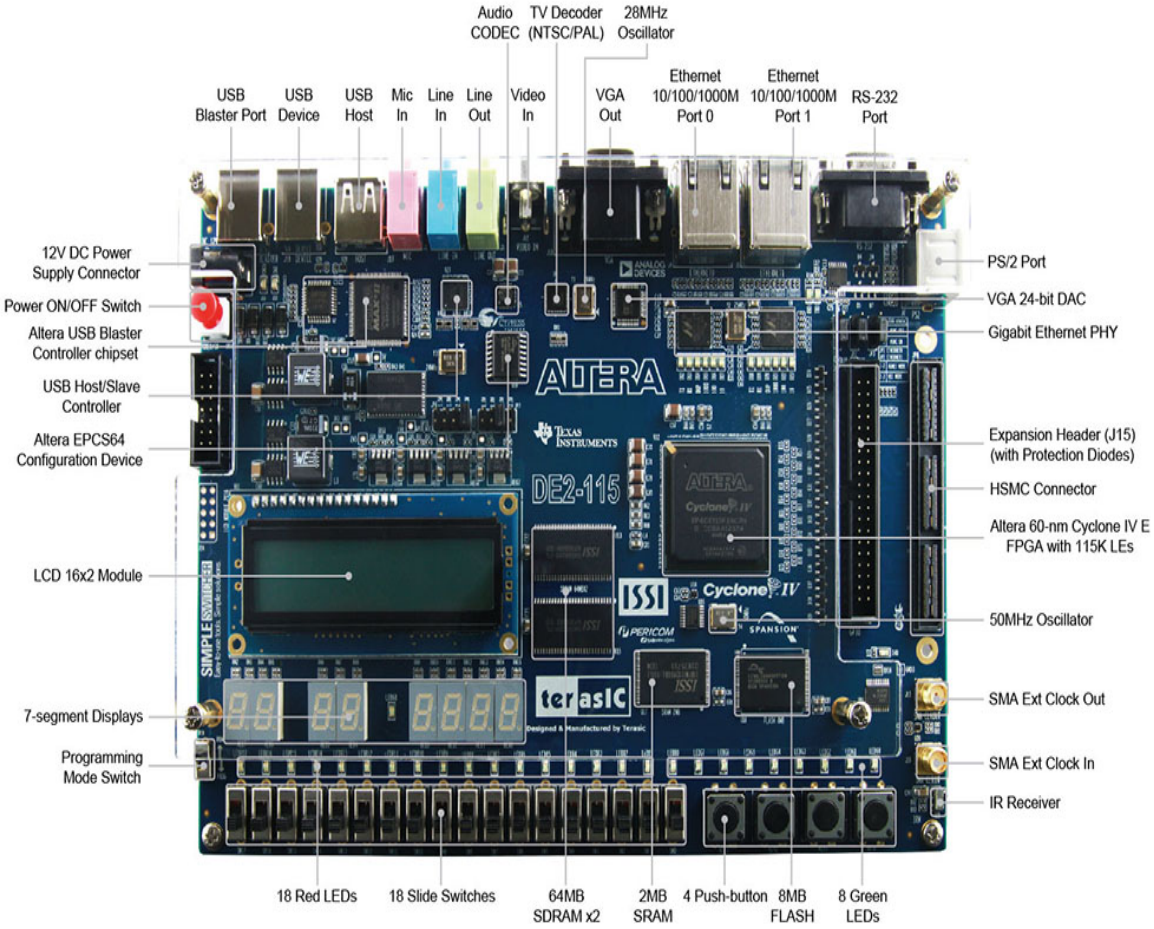

Imagen 2.3: Tarjeta de desarollo Altera DE2-115.

2.6. UART (Universal Asynchronous Receiver-Transmitter)

El UART es un protocolo de comunicación serie asíncrono ampliamente utilizado en sistemas embebidos por su simplicidad y bajo costo. Permite la transmisión de datos entre dispositivos mediante el envío de bits secuenciales. Aunque eficiente, UART no incluye mecanismos de seguridad nativos, lo que lo hace susceptible a ataques como sniffing, inyectado de datos, y suplantación de dispositivos.

2.7. FPGA (Field-Programmable Gate Array)

Las FPGAs son dispositivos semiconductores programables que permiten implementar circuitos digitales personalizados. Se utilizan en múltiples aplicaciones donde se requiere alta flexibilidad, paralelismo y velocidad. Las FPGAs son ideales para:

Implementar algoritmos criptográficos en hardware.

Realizar prototipos rápidos.

Diseños de sistemas de comunicación personalizados.

2.8. Diseño RTL (Register Transfer Level)

El diseño a nivel de transferencia de registros (RTL) es una metodología de modelado de hardware digital que describe cómo los datos se mueven entre registros y qué operaciones lógicas se aplican. Este nivel de abstracción permite desarrollar arquitecturas eficientes en FPGAs para tareas como cifrado, descifrado y control de flujos de datos.

2.9. Seguridad en Sistemas Embebidos

Los sistemas embebidos, al estar presentes en dispositivos IoT, sensores, sistemas médicos y automotrices, requieren niveles crecientes de seguridad. Las amenazas comunes incluyen acceso físico, clonación, manipulación de firmware y espionaje de datos. El uso de hardware

criptográfico integrado es una de las soluciones más efectivas para mitigar estos riesgos, sobre todo en dispositivos con recursos limitados.

2.10. Trabajos Relacionados

En investigaciones previas se han propuesto mecanismos de seguridad sobre UART, pero en su mayoría dependen de software, lo que limita la eficiencia y vulnerabilidad ante ataques. Otros estudios han demostrado que la implementación de algoritmos criptográficos en FPGA, como AES en modo CTR, es factible y efectiva para mejorar la seguridad sin afectar el rendimiento significativamente.

2.11. Metodología

2.11.1. Enfoque Metodológico

Este trabajo de investigación sigue una metodología de tipo experimental y aplicada, con enfoque cuantitativo, ya que se propone diseñar, implementar y evaluar un sistema de comunicación seguro basado en el protocolo UART, incorporando funciones criptográficas mediante diseño RTL en una FPGA.

2.12. Marco procedimental/ Etapas de Desarollo

La metodología se estructura en las siguientes etapas:

2.12.1. Análisis del protocolo UART

- Estudio de la estructura y funcionamiento del protocolo UART.
- Identificación de vulnerabilidades y limitaciones en cuanto a seguridad.

2.12.2. Diseño de la capa criptográfica

- Selección de un algoritmo de cifrado simétrico adecuado (por ejemplo, AES o XOR extendido para propósitos didácticos).
- Modelado de módulos de cifrado y descifrado a nivel RTL (Register Transfer Level).

2.12.3. Integración con el protocolo UART

- Modificación del módulo UART para incluir la capa criptográfica sin alterar su estructura base.
- Definición de flujos de datos seguros: entrada cifrada, salida descifrada.

2.12.4. Implementación en FPGA

- Codificación en Verilog.
- Síntesis, simulación y verificación funcional utilizando herramientas CAD (Quartus de Intel).
- Implementación física en una FPGA (por ejemplo, una placa DE10-Lite, Nexys A7 o similar).

2.12.5. Validación y pruebas

- Pruebas funcionales de transmisión segura UART en hardware.
- Medición de rendimiento: latencia, velocidad de transmisión, utilización de recursos.
- Comparación con versiones sin seguridad o implementaciones en software.

2.12.6.

CAPÍTULO 3

Diseño del sistema

3.1. Diseño Conceptual

3.1.1. Necesidades - Requerimientos

3.1.2. Arquitectura funcional (funciones)

3.1.3. Arquitectura física (módulos)

3.1.4. Propuestas de solución

3.1.4.1. Módulos ($M_1 \dots M_n$)

3.1.4.2. Integración (sistema)

3.1.5. Validación

3.1.6. Selección diseño conceptual

3.2. Diseño Detallado

3.2.1. Detalle módulo 1 (Ml)

3.2.1.1. Validación (Ml)

3.2.2. Detalle módulo 2 (M2)

3.2.2.1. Validación (M2)

3.2.3. Detalle módulo n (Mn)

3.2.3.1. Validación (Mn)

3.2.4. Integración sistema mecatrónico

3.2.5. Validación sistema mecatrónico

CAPÍTULO 4

Implementación del sistema

4.1. Implementación módulo 1 (Ml)

4.1.0.1. Verificación (Ml)

4.2. Implementación módulo 2 (M2)

4.2.0.1. Verificación (M2)

. . . .

4.3. Implementación módulo 2 (Mn)

4.3.0.1. Verificación (Mn

4.4. Implementación módulo 1 (Ml)

4.4.0.1. Verificación (Ml)

CAPÍTULO 5

Discusión / Análisis de resultados

5.0.1. Análisis de ingeniería

5.0.2. Análisis de costos

5.0.3. Análisis de valor

Conclusiones

una conclusión

Recomendaciones y trabajo futuro

Referencias Bibliográficas

- IEEE Standards Association. (1995). IEEE Standard Verilog Hardware Description Language (IEEE Std 1364-1995). Institute of Electrical and Electronics Engineers.
- IEEE Standards Association. (2005). IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language (IEEE Std 1800-2005). Institute of Electrical and Electronics Engineers.
- Palnitkar, S. (2003). Verilog HDL: A Guide to Digital Design and Synthesis (2nd ed.). Prentice Hall.
- Edad del Hierro
- Kahn, D. (1996). The codebreakers: The comprehensive history of secret communication from ancient times to the Internet (2nd ed.). Scribner.
- Schneier, B. (2015). Applied cryptography: Protocols, algorithms, and source code in C (20th anniversary ed.). Wiley.
- Singh, S. (2000). The code book: The science of secrecy from ancient Egypt to quantum cryptography. Anchor Books.
- Stallings, W. (2017). Cryptography and network security: Principles and practice (7th ed.). Pearson.

- Accellera Systems Initiative. (2002). SystemVerilog Language Reference Manual. Accellera. https://www.accellera.org/
- IEEE Standards Association. (2005). IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language (IEEE Std
- 1800-2005). IEEE. https://doi.org/10.1109/IEEESTD.2005.93210
- IEEE Standards Association. (2008). IEEE Standard Verilog Hardware Description Language. IEEE.
- Padrón, L. (1997). Implementación de modelos de circuitos neuronales electrónicos. Laboratorio de Computación Adaptable.
- Padrón, L., et al. (2000). Modelado y simulación de redes neuronales electrónicas usando MATLAB y SIMULINK. Revista de Computación Adaptable, 12(3), 338–349.
- Terasic Technologies. (s.f.). DE2-115 User Manual. Recuperado de https://www.terasic.com.tw
- Intel Corporation. (2020). Cyclone IV Device Handbook (Volume 1 & 2). Retrieved from https://www.intel.com
- Vahid, F., & Givargis, T. (2010). Embedded System Design: A Unified Hardware/Software Introduction (2nd ed.). Wiley.

Apéndices

Apéndice 1

Incluyen información que ayuda a interpretar parte del contenido del libro, o aspectos más técnicos y menos esenciales del libro, como información complementaria.

Ab hier beginnt der `Anhang`.

Der Zweite Anhang (Apéndice 2)

Incluyen información que ayuda a interpretar parte del contenido del libro, o aspectos más técnicos y menos esenciales del libro, como información complementaria.

Der dritte Anhang (Apéndice 3)

Incluyen información que ayuda a interpretar parte del contenido del libro, o aspectos más técnicos y menos esenciales del libro, como información complementaria.

Anexos

Anexo 1. Hoja de datos

Una ficha técnica, hoja técnica u hoja de datos (datasheet en inglés), también ficha de características u hoja de características, es un documento que resume el funcionamiento y otras características de un componente (por ejemplo, un componente electroóico) o subsistema (por ejemplo, una fuente de alimentación) con el suficiente detalle para ser utilizado por un ingeniero de diseño y diseñar el componente en un sistema.

Comienza típicamente con una página introductoria que describe el resto del documento, seguido por los listados de componentes específicos, con la información adicional sobre la conectividad de los dispositivos. En caso de que haya código fuente relevante a incluir, se une cerca del extremo del documento o se separa generalmente en otro archivo.

Las fichas técnicas no se limitan solo a componentes electrónicos, si no que también se dan en otros campos de la ciencia, como por ejemplo compuestos químicos o alimentos.

Anexo 2. Hoja de datos

Una ficha técnica, hoja técnica u hoja de datos (datasheet en inglés), también ficha de características u hoja de características, es un documento que resume el funcionamiento y otras características de un componente (por ejemplo, un componente electroóico) o subsistema (por ejemplo, una fuente de alimentación) con el suficiente detalle para ser utilizado por un ingeniero de diseño y diseñar el componente en un sistema.

Comienza típicamente con una página introductoria que describe el resto del documento, seguido por los listados de componentes específicos, con la información adicional sobre la conectividad de los dispositivos. En caso de que haya código fuente relevante a incluir, se une cerca del extremo del documento o se separa generalmente en otro archivo.

Las fichas técnicas no se limitan solo a componentes electrónicos, si no que también se dan en otros campos de la ciencia, como por ejemplo compuestos químicos o alimentos.

\bigskip

Anexo 3 (Anexo título 3)

Una ficha técnica, hoja técnica u hoja de datos (datasheet en inglés), también ficha de características u hoja de características, es un documento que resume el funcionamiento y otras características de un componente (por ejemplo, un componente electrónico) o subsistema (por ejemplo, una fuente de alimentación) con el suficiente detalle para ser utilizado por un ingeniero de diseño y diseñar el componente en un sistema.

Comienza típicamente con una página introductoria que describe el resto del documento, seguido por los listados de componentes específicos, con la información adicional sobre la conectividad de los dispositivos. En caso de que haya código fuente relevante a incluir, se une cerca del extremo del documento o se separa generalmente en otro archivo.

Las fichas técnicas no se limitan solo a componentes electrónicos, si no que también se dan en otros campos de la ciencia, como por ejemplo compuestos químicos o alimentos.

\bigskip