elektron ámbar

Technical Report

Revista elektron, Vol. 9, No. 2, pp. 56-67 (2025)
https://doi.org/10.37537/rev.elektron.9.2.220.2025

Topología Daisy Chain en FPGA para modernización naval: implementación y extensión con ARP

Daisy Chain Topology on FPGA for naval modernization: implementation and extension with ARP

Emiliano S. Gallo [*#1], Ricardo L. Cayssials [#2], Christian L. Galasso [*†#3] y Alan E. Arias [#4]

[#] *Depto. de Ingeniería Electrónica, Universidad Tecnológica Nacional - F.R.B.B.*
Bahía Blanca, Argentina
[1] emilianogallo@frbb.utn.edu.ar
[2] rcayssials@frbb.utn.edu.ar

[†] *Universidad de la Defensa Nacional – FADARA – ESOA*
Punta Alta, Argentina.
[*] *Servicio de Análisis Operativos, Armas y Guerra Electrónica de la Armada*
Punta Alta, Argentina
[3] clgalasso@frbb.utn.edu.ar

Recibido: 29/09/25; Aceptado: 09/11/25

***Abstract*— Operational consoles, human-machine interfaces implemented in Surface Units of the Naval Fleet Command, facilitate the visualization of critical data and system management through operator command input. However, their operability and upgrade potential are restricted by dependence on proprietary hardware. This report describes a component of a broader project whose purpose is to replace the "Optronic Sensor Control Console" (CCSO) with a commercial computer (PC). This modernization seeks to adapt the system to new operational demands, enable the incorporation of new functionalities, and eliminate dependence on proprietary components. This article specifically addresses the operational logic of a Daisy Chain topology, used as an interconnection strategy between devices. Its concept, digital implementation in VHDL, and validation through a SystemVerilog testbench are presented, highlighting its role in the proposed architecture.**

***Keywords*— Daisy Chain, digital design, FPGA, modernization, VHDL.**

***Resumen*— Las consolas de operaciones, interfaces de interacción humana implementadas en las Unidades de Superficie del Comando de la Flota Naval, facilitan la visualización de datos críticos y la gestión de sistemas mediante la entrada de comandos por parte de los operadores. Sin embargo, su operatividad y potencial de actualización están restringidos por la dependencia de hardware propietario. Este informe describe un componente de un proyecto más amplio cuyo propósito es sustituir la "Consola de Control de Sensor Optrónico" (CCSO) por una computadora comercial (PC). Esta modernización busca adaptar el sistema a nuevas demandas operativas, permitir la incorporación de nuevas funcionalidades y eliminar la dependencia de componentes propietarios. En este artículo se aborda específicamente la lógica de funcionamiento de una topología Daisy Chain, utilizada como estrategia de interconexión entre dispositivos. Se presenta su concepto, su implementación digital en VHDL y la validación mediante un testbench en SystemVerilog, destacando su rol en la arquitectura propuesta.**

***Palabras clave*— Daisy Chain, diseño digital, FPGA, modernización, VHDL.**

I. Introducción

Los sistemas embarcados en unidades navales suelen presentar restricciones operativas asociadas a su naturaleza de tiempo real duro, al uso de hardware propietario y a la dependencia de protocolos específicos. Estos factores dificultan las tareas de mantenimiento, encarecen la incorporación de nuevas funcionalidades y limitan la capacidad de modernización.

En el caso particular del subsistema de control de un sensor optrónico, la arquitectura original se estructuraba en torno a la CCSO, módulos intermedios denominados "Placas de Transporte" y una computadora naval (en adelante, CN). Con el paso del tiempo, la obsolescencia de los componentes, el elevado costo de los repuestos y la dependencia de proveedores únicos hicieron necesaria una modernización.

Propuestas de modernización alternativas consideraban la modificación completa de los sistemas y su reemplazo por nuevas soluciones propietarias. Sin embargo, dichas propuestas resultaban impracticables debido a sus elevados costos, el prolongado tiempo fuera de operación requerido y, nuevamente, la dependencia de soluciones propietarias. La solución propuesta en este trabajo considera una modernización gradual de los diferentes sistemas, teniendo como limitación principal las restricciones técnicas de la infraestructura disponible en las unidades navales.

El presente trabajo se enmarca en el proyecto "Modifi-

cación de la lógica de la placa ETHER-TMIO para implementar una conexión entre una computadora naval y una comercial". El objetivo principal de dicho proyecto es desarrollar un diseño digital que permita reemplazar la CCSO por un sistema compuesto por una PC y un microcontrolador (uC), comunicados mediante Ethernet, manteniendo la compatibilidad con el protocolo original y cumpliendo con las restricciones temporales impuestas por la CN.

En este contexto, uno de los primeros desafíos abordados fue resolver las incompatibilidades temporales entre la CN y la PC. Para ello, se implementó en una FPGA [1,2] una topología daisy chain [3,4], capaz de gestionar [5] la progresión del protocolo, asignar prioridad a los periféricos que componen la CCSO, mantener el estado de los mensajes para cada periférico (ya que se trabaja con mensajes apropiativos) y aprovechar al máximo el medio de comunicación. Además, cada eslabón de la cadena gestiona la comunicación con un periférico en particular. Los datos recibidos o transmitidos se almacenan o leen desde una memoria dual port. En la Fig. 1 se busca contrastar el sistema original con el sistema moderno.

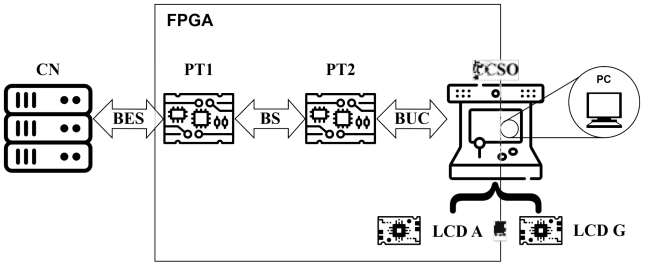

Fig. 1. Contraste entre sistema original vs sistema moderno.

Una vez resueltas las incompatibilidades temporales, fue necesario implementar un diseño digital capaz de transmitir y recibir tramas Ethernet. Para ello, además de incorporar los componentes encargados de transmisión y recepción, se diseñó una Daisy Chain encargada de la multiplexación de memorias, además de ofrecer versatilidad para soportar distintos protocolos de comunicación.

En este trabajo se describe la lógica de funcionamiento y el rol de esta topología, destacando su flexibilidad y adaptabilidad para la incorporación de nuevas funcionalidades de manera eficiente. La contribución principal del trabajo radica en la implementación de un eslabón especializado para el Protocolo de Resolución de Direcciones (ARP), que demuestra cómo la topología Daisy Chain puede extenderse más allá de la multiplexación de memorias para integrar protocolos de comunicación completos. Esta capacidad de combinar el principio de propagación secuencial con la ejecución de protocolos confirma la adaptabilidad de la arquitectura propuesta a contextos mixtos de tiempo real y no real, manteniendo la estructura modular de la solución.

II. Sistema original

La Fig. 2 muestra el sistema original diseñado con una arquitectura distribuida basada en hardware propietario y protocolos específicos del fabricante. Su función principal era vincular la CN con la CCSO, permitiendo al operador visualizar información proveniente del sensor y enviar comandos de control.

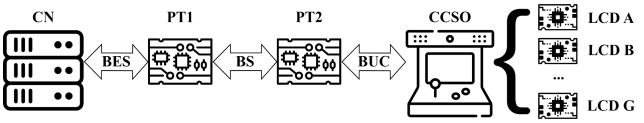

Fig. 2. Sistema original.

A. Elementos principales

Computadora Naval (CN): Actúa como la unidad central de gestión, funcionando como servidor del sistema. Se encarga de emitir instrucciones, así como de recibir y transmitir información hacia los periféricos que integran la CCSO. La comunicación de datos se realiza en formato paralelo a través del Bus de Entrada/Salida (BES), el cual opera con un ancho de palabra (24 bits) y temporización establecidos por el fabricante.

Placa de Transporte 1 (PT1): Situada junto a la CN, esta placa recibe datos desde el BES y los convierte a formato serial para enviarlos por el Bus Serie (BS) hacia la segunda placa de transporte. También puede realizar el proceso inverso, convirtiendo datos seriales provenientes de la PT2 a formato paralelo para entregarlos a la CN.

Placa de Transporte 2 (PT2): Ubicada junto a la CCSO, la PT2 realiza la función complementaria a la PT1. Transforma datos seriales recibidos desde el BS a formato paralelo para transferirlos a la CCSO mediante el Bus de Unidad de Control (BUC). Asimismo, serializa los datos generados por la CCSO para enviarlos hacia la PT1.

Consola de Control de Sensor Optrónico (CCSO): Representa la interfaz de operación física, equipada con una pantalla alfanumérica, controles de mando y módulos electrónicos específicos. La gestión de los periféricos se lleva a cabo mediante placas conocidas como Lógica de Control de Dispositivo (LCD), con una placa dedicada para cada periférico de la CCSO.

Buses de comunicación:

- **Bus de Entrada/Salida (BES):** Enlace paralelo entre la CN y la PT1. Este bus posee 24 bits de salida y 24 bits de entrada (bus de 48 bits), y utiliza un clock de 4 MHz.
- **Bus Serie (BS):** Enlace serie entre la PT1 y la PT2. Es un bus de 10 pares diferenciales, 5 de entrada y 5 de salida, donde 6 de esos pares son para datos y los 4 restantes para handshake. La tasa de transferencia es de 12 Mb/s.
- **Bus de Unidad de Control (BUC):** Conexión paralela entre PT2 y la CCSO. Este bus posee 24 bits de salida y 24 bits de entrada (bus de 48 bits), y utiliza un clock de 4 MHz.

B. Flujo de datos

Los datos generados por la CN se transmitían a través del BES hacia la PT1, donde se convertirán a formato serial para su envío por el BS. Posteriormente, la PT2 transformaba estos datos nuevamente a formato paralelo y los entregaba al BUC, permitiendo la interacción con la CCSO, ya sea para enviar comandos o intercambiar información.

En el flujo inverso, desde la CCSO hacia la CN, los datos de los periféricos se envían en formato paralelo a la PT2, que los serializa para su transmisión por el BS. Finalmente, la PT1 convierte estos datos a formato paralelo para su entrega a la CN.

III. Propuesta de modernización

Con el objetivo de reemplazar la CCSO original por una computadora comercial, surgieron nuevas problemáticas derivadas de este cambio. Estas problemáticas se dividen en dos categorías principales: aquellas asociadas a los requerimientos de tiempo real duro y las relacionadas con aspectos no temporales críticos. Esta distinción permite abordar cada una de manera modular, reutilizando soluciones validadas para los desafíos temporales y extendiendo la arquitectura hacia nuevos requisitos funcionales, tales como la gestión eficiente de paquetes de red.

Para el caso de los requerimientos de tiempo real, la diferencia del comportamiento temporal entre la CN y la PC generaba incompatibilidades que podían comprometer la correcta transmisión de datos y el cumplimiento estricto del protocolo original. Como se detalló en trabajos previos [3,5], la solución consistió en implementar una FPGA como intermediario, configurada con una topología Daisy Chain. Esta solución coordina la secuencia de comunicación, asigna prioridades entre periféricos y asegura el cumplimiento de los tiempos de respuesta exigidos por el protocolo, sin alterar la funcionalidad del sistema original. En este contexto, la Daisy Chain actúa como un mecanismo de arbitraje determinístico que replica el comportamiento de la CCSO dedicada, garantizando los tiempos requeridos por el sistema y asegurando un aprovechamiento eficiente del bus de comunicación mediante la propagación secuencial de señales y datos.

Por otro lado, los desafíos no temporales surgen de la integración de la PC en el flujo de datos, donde se requiere flexibilidad y escalabilidad en la gestión de paquetes de información. A diferencia de los procesos estrictamente temporizados, aquí el foco está en la interoperabilidad con redes modernas y en maximizar el throughput sin colisiones. Para resolver este aspecto, se extendió el uso de la topología Daisy Chain hacia un nuevo rol: el de multiplexor de memorias dual-port. Por un lado, estas memorias, permiten almacenar mensajes completos provenientes de la CN que luego son transmitidos hacia la red a través de Ethernet; por otro, habilitan la recepción de paquetes desde la red, cuyo payload se guarda en memoria para su posterior entrega a la CN.

Una característica destacada de esta extensión es la incorporación de un eslabón especializado dedicado al protocolo ARP. A diferencia de los demás eslabones, cuya función se limita a la multiplexación de memorias, este integra la lógica propia de la Daisy Chain con la implementación completa del protocolo de resolución de direcciones físicas. De este modo, la arquitectura no solo mantiene la modularidad y orden secuencial de la topología, sino que también demuestra su versatilidad al admitir la inserción de eslabones con funciones más complejas. En particular, el eslabón “ARP” maneja la difusión de solicitudes (requests) y la generación de respuestas (replies).

En síntesis, la propuesta de modernización se apoya en una doble aplicación de la topología Daisy Chain: en primer lugar, como mecanismo de arbitraje determinístico que asegura el cumplimiento de los requisitos de tiempo real; y en segundo lugar, como multiplexor flexible de memorias dual-port que habilita la integración de protocolos de red. La inclusión de un eslabón especializado en ARP constituye un aporte significativo, ya que combina el principio de propagación secuencial de la topología con la ejecución de un protocolo de comunicación, confirmando la adaptabilidad de esta arquitectura a contextos mixtos de tiempo real y no real.

A. Problemáticas identificadas

La modernización introduce desafíos que, aunque interrelacionados, se clasifican claramente en temporales y no temporales, permitiendo soluciones diferenciadas pero complementarias.

- **Incompatibilidad temporal entre CN y PC (tiempo real):** La CN opera bajo tiempos de respuesta estrictos y determinísticos, mientras que la PC introduce variaciones que pueden violar la secuencia temporal del protocolo.
- **Manejo de prioridades (tiempo real):** En el sistema original, la CN gestiona de manera implícita la prioridad de los periféricos. Con la PC, esta coordinación requiere un mecanismo intermedio para evitar colisiones o pérdidas de datos.
- **Mensajes apropiativos y gestión de estado (tiempo real):** Los mensajes en el protocolo original son apropiativos, lo que permite que la CN controle de manera flexible el uso del medio de comunicación. Esto significa que la CN puede alternar entre envío y recepción de palabras de distintos periféricos, según las necesidades de cada intercambio. Para asegurar la correcta transmisión de datos, es necesario mantener el estado de la comunicación de cada periférico por separado, de modo que cada mensaje se complete sin pérdidas y se aproveche eficientemente la capacidad del bus.
- **Comunicación full-dúplex (tiempo real):** La CN tiene la capacidad de leer y escribir de manera simultánea, haciendo máximo provecho del medio de comunicación.
- **Transmisión/recepción de datos vía Ethernet:** El medio utilizado para comunicarse con la PC/uC es Ethernet. Por lo tanto, el diseño digital implementado por la FPGA debe tener la capacidad de hacer adición dinámica de encabezado protocolarios (Ethernet II, IPv4 y UDP) y de recuperar el payload.
- **Verificación de integridad:** Se presenta la necesidad de implementar algoritmos de cálculo de CRC y Checksum para garantizar la fiabilidad de la red.
- **Protocolos de red:** La implementación de protocolos de red, como es el caso del protocolo de resolución de direcciones, es necesario ya que sin este, no se podría establecer comunicación con los distintos dispositivos que integren la red.

Estas problemáticas motivaron el desarrollo de un diseño digital modular, con componentes dedicados a resolver cada desafío específico.

B. Sistema moderno

El sistema moderno mantiene la CN como controlador principal y utiliza a la FPGA [6] como interfaz para comunicarse con la PC y el microcontrolador. En esta arquitectura, las placas de transporte del sistema original (PT1 y PT2 de la Fig. 2) son reemplazadas por la FPGA, manteniendo el bus BES de comunicación con la CN y aprovechando el cableado serie —compuesto por pares trenzados— fue reutilizado para implementar un puerto Ethernet (ETH), permitiendo aprovechar la infraestructura física existente para establecer una nueva capa de comunicación digital. Las partes que lo componen son las siguientes:

- **FPGA:** Esta implementa un diseño digital encargado de resolver las incompatibilidades temporales, asignar prioridades a los distintos periféricos que integran la CCSO y maximizar el aprovechamiento del medio. Además, la FPGA se encarga de la transmisión y recepción de tramas Ethernet, resolviendo los requisitos asociados a tiempos diferidos.
- **PC/Microcontrolador:** Componen el reemplazo de la CCSO, donde la PC emula la CCSO original y el microcontrolador se encarga de realizar conversiones analógico-digitales.

En conjunto, esta arquitectura permite reemplazar la consola original sin comprometer la integridad del protocolo, manteniendo la funcionalidad de todos los periféricos y habilitando la incorporación de nuevas funciones en la PC. En la Fig. 3 se presenta una imagen representativa del sistema moderno que se plantea.

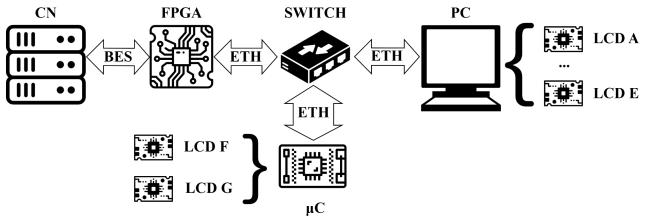

Fig. 3. Sistema moderno.

Cabe destacar que la CCSO, además de estar compuesta por los periféricos y sus correspondientes módulos LCD, incluía una interfaz de usuario que permitía la visualización de datos asociados a determinados periféricos y el accionamiento de comandos vinculados a otros. En el sistema modernizado, el procesamiento de comunicaciones e información originalmente realizado por las LCD fue incorporado dentro de la FPGA, mientras que la visualización de datos y el accionamiento de comandos son gestionados por una PC. Esta última, mediante un software desarrollado específicamente, emula la interfaz original de la CCSO y cumple el rol de interfaz de usuario, permitiendo la interacción del operario con los periféricos del sistema.

Habiendo establecido la necesidad de una FPGA para resolver las incompatibilidades del sistema moderno, a continuación se describe el diseño digital implementado en la FPGA.

IV. Diseño digital

Este trabajo forma parte de un proyecto mayor de modernización de un sistema naval, que implica el reemplazo de una CCSO por una computadora comercial. Para ello, se resuelven las problemáticas identificadas en la sección anterior, clasificadas en tiempo real duro y no crítico. La Fig. 4 muestra el esquema del diseño digital implementado en la FPGA.

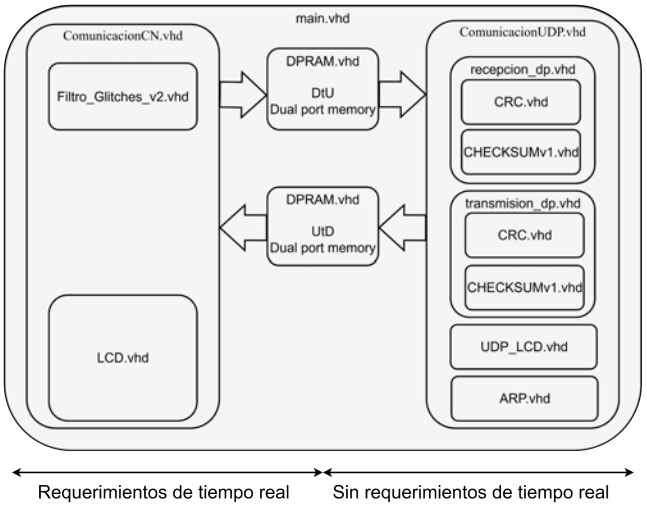

Fig. 4. Diseño digital.

A. Descripción de bloques

La Fig. 4 ilustra los componentes principales del diseño, cada uno con funciones específicas para abordar las problemáticas identificadas. A continuación, se presenta una descripción breve de cada uno.

- **main:** Entidad superior que instancia los componentes “ComunicacionCN” y “ComunicacionUDP”, además de las memorias dual-port que actúan como interfaz entre ellos.
- **ComunicacionCN:** Componente encargado de resolver los requisitos de tiempo real duro. Esta entidad única instancia internamente los componentes "Filtro_Glitches_v2" y múltiples componentes "LCD" (uno por cada periférico), interconectados en topología Daisy Chain.
- **Filtro_Glitches_v2:** Como indica su nombre, filtra las señales provenientes de la CN a través del bus BES para eliminar glitches y ruido.
- **LCD:** Gestiona la comunicación entre la CN y un periférico específico. Controla la progresión del protocolo de comunicación, asegurando el cumplimiento de los tiempos requeridos y optimizando el uso del medio. Por cada periférico (contando con un total de 7) de la CCSO se instancia un componente “LCD”, interconectados para conformar la topología Daisy Chain. La interconexión establece la prioridad de cada periférico.
- **DPRAM:** Memoria de doble puerto (dual-port RAM) implementada mediante una IP propia, utilizando bloques BRAM internos de la FPGA. El ancho de palabra es de 8 bits y el tamaño de cada memoria se dimensiona según los requisitos del periférico asociado. El diseño contempla desde periféricos de baja tasa de datos (por ejemplo, 4 palabras de 3 bytes, lo que equivale a una memoria de 16 bytes) hasta periféricos de alta tasa (460 palabras de 3 bytes, implementados con una memoria de 2 KB). Cada módulo “LCD” gestiona entre una y dos memorias, dependiendo de si el periférico opera de forma unidireccional o bidireccional. En el caso unidireccional, el componente “LCD” realiza la lectura de los datos almacenados en la memoria cuando el flujo es desde el periférico hacia la CN, o la escritura cuando el flujo es desde la CN hacia el periférico. En cambio, si el periférico intercambia datos en ambas direcciones,

se utilizan dos memorias: una destinada a la lectura y otra a la escritura.

- **ComunicacionUDP:** Encargado de resolver las problemáticas no temporales. Instancia los componentes "transmision_dp", "recepcion_dp", "UDP_LCD" y "ARP".
- **transmision_dp:** Transmite los datos almacenados en las memorias dual-port, agregando headers de los protocolos UDP, IPv4 y Ethernet II. Instancia los componentes "CRC" y "Checksum_v1" para verificación de integridad.
- **recepcion_dp:** Procesa los datos recibidos por la interfaz Ethernet de la FPGA. Tras verificar que la trama está destinada a la FPGA, almacena el payload en la memoria dual-port correspondiente al periférico emisor. Instancia CRC y Checksum_v1 para validación.
- **CRC:** Componente utilizado por "transmision_dp" y "recepcion_dp" para calcular la redundancia cíclica (Cyclic Redundancy Check) y verificar la integridad de los datos, detectando posibles corrupciones.
- **Checksum_v1:** Calcula la suma de verificación (checksum) para validar la integridad de los datos en los protocolos IP y UDP.
- **UDP_LCD:** Multiplexa las memorias dual-port tanto en recepción como en transmisión. Cuando se recibe una trama, el payload es escrito en memoria y el "UDP_LCD" correspondiente al periférico emisor gestiona la multiplexación para su posterior envío a la CN. El mismo mecanismo se aplica para la transmisión. Se instancian varios "UDP_LCD", cada uno asociado a un periférico. Su interconexión conforma una topología Daisy Chain, alineada con las prioridades establecidas en "ComunicacionCN".
- **ARP:** Gestiona el protocolo de resolución de direcciones (Address Resolution Protocol). Además, incorpora lógica de difusión de datos similar a "UDP_LCD", lo que lo hace compatible con la topología Daisy Chain y permite su integración como eslabón en la cadena.

De esta manera, se ha descrito de forma simplificada la función de cada componente del diseño digital. A lo largo de este trabajo, se hará énfasis en la topología Daisy Chain implementada en "ComunicacionUDP", particularmente en el componente "ARP". A continuación, se presenta el fundamento extendido de esta topología, destacando su adaptación al contexto no temporal.

V. Fundamento de la topología daisy chain

La topología Daisy Chain [7,8] se basa en la conexión secuencial de múltiples módulos o componentes, conformando una cadena lineal de dispositivos. En este esquema, la información llega de manera simultánea a cada eslabón, pero solo el dispositivo correspondiente la procesa, mientras que los demás actúan como repetidores. Esto garantiza un flujo de datos ordenado, en el cual la prioridad y la secuencia de transmisión dependen de la posición dentro de la cadena.

En el caso particular de este proyecto, la topología Daisy Chain permite gestionar tanto las operaciones de escritura, realizadas por el módulo de transmisión, como las de lectura, ejecutadas por el módulo de recepción. Para ello, se implementa la multiplexación entre la memoria que contiene los datos a enviar y aquella destinada al almacenamiento de la información recibida, posibilitando su acceso de manera simultánea.

Al tratarse de una arquitectura modular, esta topología brinda flexibilidad para incorporar nuevos componentes encargados de ejecutar protocolos específicos sin necesidad de rediseñar por completo el sistema de interconexión. Un ejemplo de ello es la integración de un módulo para el protocolo de resolución de direcciones.

En conclusión, la topología Daisy Chain, adaptada en "ComunicacionUDP", cumple con los requisitos del proyecto al realizar la multiplexación de memorias dual-port, donde cada eslabón gestiona la/s memoria/s asociadas a un periférico específico. Asimismo, su diseño modular habilita la integración de funcionalidades avanzadas, como el protocolo ARP, asegurando una comunicación robusta y escalable entre la FPGA, la PC y el microcontrolador. Esta adaptabilidad la posiciona como un pilar clave para la modernización propuesta, ya que facilita la evolución futura del sistema.

VI. Implementación de la topología daisy chain

Considerando las problemáticas identificadas y la solución propuesta, esta sección detalla la implementación funcional de la topología Daisy Chain dentro del diseño digital. El propósito es explicar al lector cómo cada eslabón se asocia a un periférico específico, cuál es su función y cómo la lógica implementada cumple con los objetivos establecidos. Posteriormente, se profundiza en el componente "ARP", que no solo forma parte de la cadena, sino que también ejecuta su protocolo homónimo, destacando su rol distintivo.

Cada eslabón está asociado a un periférico en particular. A su vez, cada periférico puede requerir una o dos memorias, según la comunicación sea unidireccional o bidireccional. En consecuencia, el eslabón no solo se asocia a un periférico, sino también a su/s memoria/s correspondiente/s, realizando multiplexación cuando resulte necesario.

La cadena completa se construye mediante la integración de múltiples eslabones que interactúan de forma ordenada y coordinada para multiplexar las memorias durante las operaciones de lectura y escritura, permitiendo así la transmisión o recepción de datos según corresponda. La interconexión entre los módulos establece prioridades de multiplexación, de manera que, cuando existen varios mensajes listos en memoria para su transmisión, los datos se envían conforme a su relevancia operativa.

En los subapartados siguientes se detalla primero la estructura y funcionalidad de un eslabón individual, y luego la forma en que estos se integran para conformar la cadena completa.

A. Componente UDP_LCD

Cada eslabón multiplexa la memoria cada vez que el periférico requiere transmitir o recibir datos, siempre que sea el seleccionado para la operación correspondiente.

Al momento de instanciar un eslabón, se configuran diversos "generics": dirección del dispositivo, dimensiones de la memoria asociada (ancho de palabra y profundidad) y habilitación de recepción.

Mediante la dirección del dispositivo, el componente queda asociado a un periférico específico y a su/s memoria/s correspondiente/s.

Antes de detallar la lógica de funcionamiento, se presentan los puertos con sus señales asociadas, acompañados de una breve descripción para facilitar su comprensión.

1) Puertos

Las señales asociadas a puertos homónimos se organizaron en bloques para simplificar su explicación posterior. A continuación, en la Fig. 5 se presenta un diagrama representativo del componente.

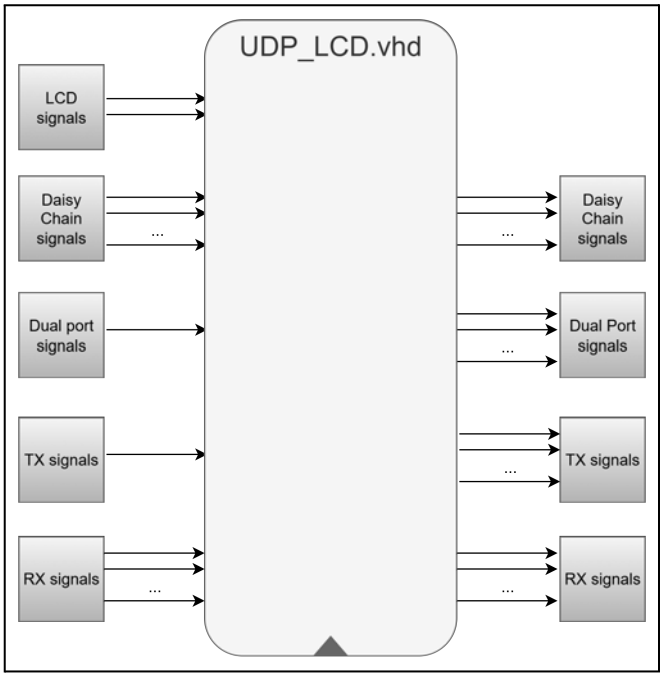

Fig. 5. Componente UDP_LCD.

En la Fig. 5 se distinguen, por bloques, las interfaces de entrada y de salida, agrupando las señales según su función. A continuación, se describen cada una de las señales.

LCD signals

Este conjunto de señales proviene del eslabón vinculado al mismo periférico en la topología Daisy Chain implementada en el componente "ComunicacionCN". Las señales que integran este grupo son:

- **READY_TX:** Indica que la memoria asociada a ambos eslabones tiene un mensaje disponible para ser transmitido.
- **FRAME_LENGTH:** Permite que el componente "LCD" informe al "UDP_LCD" la longitud del mensaje, expresada en bytes.
- **ACK_TX:** Sirve para confirmar el acuse de recibo de la transmisión.

Estas señales se emplean para el proceso de transmisión de datos.

Daisy Chain signals

Para este conjunto de señales, como se observa en la figura anterior, se distinguen un subconjunto entrante y otro saliente. Para las señales entrantes:

- DA_TX_PREV.
- LENGTH_PREV.
- DATA_TX_PREV.

Y el caso de las salientes tenemos:

- DA_TX_NEXT.
- LENGTH_NEXT.
- DATA_TX_NEXT.

Mediante estas señales se implementa la lógica de difusión, donde "DA_TX", "LENGTH" y "DATA_TX" representan la dirección de dispositivo, la longitud del mensaje a transmitir expresada en bytes y el dato a transmitir, respectivamente.

Un detalle a tener en cuenta es que, en el caso del primer eslabón —de menor prioridad, como se explicará más adelante—, los datos entrantes al componente corresponden al estado de reposo.

Adicionalmente, hay dos buses de 6 bits denominados "DA_SELECTED_TX" y "DA_SELECTED_RX". Mediante estas señales se indica qué dirección de dispositivo (o eslabón) es la seleccionada para transmitir o recibir, respectivamente.

Dual Port signals

Este conjunto de puertos se emplea para la gestión de memorias dual-port. Incluye señales de escritura: "WR_DP", "AD_WR" y "Din_DP", que representan respectivamente la señal de escritura, la dirección donde se pretende escribir y el dato que se envía a la memoria de doble puerto.

Por otro lado, se incluyen las señales destinadas a la lectura de memoria: "RD_DP", "AD_RD" y "Dout_DP", que corresponden a la señal de lectura, la dirección a acceder y el dato leído de la memoria, respectivamente.

TX signals

Como se puede observar en la Fig. 5, el componente "UDP_LCD" posee "TX signals" entrantes y salientes. Las entrantes:

- **RD_DS:** Dedicada a la acción de lectura.
- **ADDRESS_RD_DS:** Esta señal indica la dirección de memoria que se desea leer.

Por otro lado, las salientes:

- **DATA_TX_NEXT:** También agrupado en el conjunto "Daisy Chain signal", correspondiente al último eslabón de la cadena. Representa el dato leído por el componente de transmisión.
- **TX_ENDED:** Señal empleada para indicar que la transmisión del mensaje fue realizada.

Este conjunto de señales vincula al componente "UDP_LCD" con el componente de transmisión.

RX signals

Estas señales vinculan al componente "UDP_LCD" con el componente "recepcion_dp". Este conjunto incluye señales tanto entrantes como salientes. Para el caso de las entrantes:

- **WR_DS:** Dedicada a la acción de escritura.
- **ADDRESS_WR_DS:** Esta señal indica la dirección de memoria que se desea escribir.
- **DATA_WR_DS:** Representa el dato que desea ser escrito en memoria.
- **CRC_CHECK, CRC_FAIL:** Señales utilizadas para validar los datos en memoria.

Saliente:

- **ACK_RX:** Indica el acuse de recibo de los datos enviados por el componente de recepción.

2) Lógica de funcionamiento

Conociendo las señales con las que interactúa el componente, se procede a describir su lógica interna. Como se adelantó, este componente multiplexa memorias tanto para la transmisión como para la recepción de datos. Por esta

razón, ambos procesos se explican por separado, con el objetivo de detallar de manera clara cómo se llevan a cabo.

En primer lugar, se describe el proceso de transmisión tomando como caso hipotético un único eslabón, para luego extrapolar el funcionamiento a la cadena completa. Finalmente, se presenta el proceso de recepción.

Transmisión

Para ilustrar el funcionamiento, se considera un único eslabón encargado de gestionar la multiplexación de memoria asociada al periférico "A". En este escenario, la CN transmite datos hacia dicho periférico. El eslabón "LCD" del componente "ComunicacionCN" almacena el mensaje en la memoria dual-port correspondiente y, mediante las señales denominadas "READY_TX" y "FRAME_LENGTH", informa al eslabón "UDP_LCD" que existe un mensaje disponible en memoria y su longitud expresada en bytes.

El componente "UDP_LCD", mientras se encuentra en estado de reposo, actúa como repetidor: las señales salientes del grupo "Daisy Chain signals" contienen exactamente la misma información que las señales entrantes del mismo grupo.

Cuando el componente "LCD" activa la señal "READY_TX", el componente "UDP_LCD" deja de actuar como repetidor y comienza a transmitir su dirección de dispositivo asociada (periférico A) por el puerto "DA_TX_NEXT" (ver Anexo A, Listado 1).

A través de la señal "DA_SELECTED_TX" se decide cuál es el eslabón que procederá con la transmisión del mensaje. En este caso, al tratarse de un único eslabón, la dirección de dispositivo llega al final de la cadena y es seleccionado para iniciar la transmisión (ver Anexo A, Listado 2).

Posteriormente, al contener "DA_SELECTED_TX" la dirección del periférico A (THIS), el componente "UDP_LCD" difunde la longitud del payload a transmitir (ver Anexo A, Listado 3), y multiplexa las señales correspondientes de lectura, dirección de lectura y dato leído (ver Anexo A, Listado 4), habilitando al componente de transmisión para acceder a la memoria asociada y enviar los datos.

Finalmente, cuando el componente de transmisión concluye el envío, genera un evento en la señal "TX_ENDED". Como respuesta, el componente "UDP_LCD" activa la señal "ACK_TX" para informar al componente "LCD" que el mensaje en memoria fue correctamente transmitido.

Recepción

Considerando nuevamente el caso hipotético de un único eslabón, donde el periférico "B" transmite datos hacia la CN, el proceso de recepción se desarrolla de la siguiente manera.

El componente de recepción, a través de la señal "DA_SELECTED_RX", selecciona el eslabón encargado de multiplexar la memoria que debe ser escrita, en este caso, la asociada al periférico "B".

Cuando "DA_SELECTED_RX" contiene la dirección de dispositivo correspondiente al periférico "B", el componente "UDP_LCD" habilita al componente de recepción para realizar la escritura en la memoria vinculada. Para ello asigna las señales de escritura, dirección de escritura y dato a almacenar en la memoria dual-port.

Una vez almacenado el payload recibido, el componente de recepción valida los datos utilizando las señales "CRC_CHECK" y "CRC_FAIL". En caso de detectarse un error, los datos se descartan y permanecen en espera hasta ser sobrescritos por información válida. Una vez confirmada la validez de los datos, el componente "UDP_LCD" activa la señal "ACK_RX", confirmando la correcta recepción del mensaje para su posterior lectura por parte de la CN.

B. Componente ARP

La topología Daisy Chain permite incorporar nuevos eslabones, ya sea por adición de periféricos o implementación de funcionalidades específicas, como protocolos de red.

En este trabajo surgió la necesidad de implementar el protocolo de resolución de direcciones (ARP). Esto derivó en el desarrollo de un eslabón que, además de incorporar la lógica de difusión asociada a la topología Daisy Chain, es capaz de ejecutar las funciones propias del protocolo ARP. Para cumplir con esta funcionalidad, el componente debe:

- Enviar mensajes "ARP Request" al inicio del sistema, cuando aún no se dispone de las direcciones físicas de la PC y del microcontrolador.
- Repetir estas solicitudes de manera periódica en intervalos predefinidos.
- Procesar las respuestas "ARP Reply" recibidas, almacenando las direcciones de forma válida en memoria.
- Responder a solicitudes "ARP Request" generadas por otros dispositivos conectados a la red.

A continuación, se describe la implementación del protocolo de resolución de direcciones. Primero se presentan los puertos del componente, y posteriormente se detalla la lógica de funcionamiento interno.

1) Puertos

De manera análoga al componente "UDP_LCD", las señales fueron agrupadas en bloques según su función. En la Fig. 6 se muestra un diagrama representativo del componente.

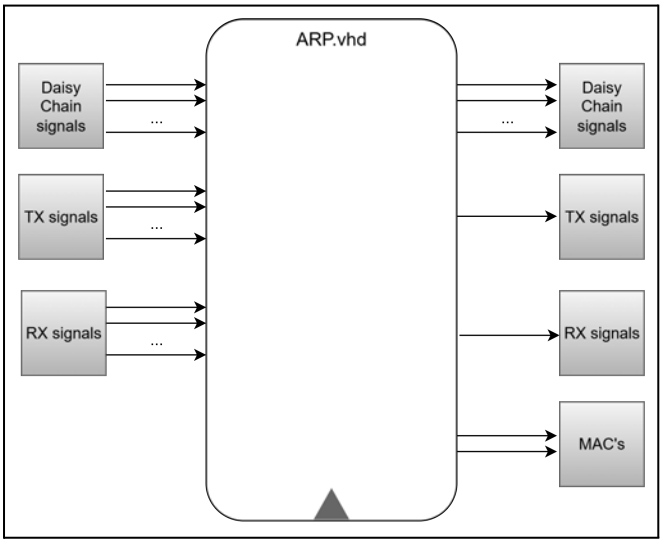

Fig. 6. Componente ARP.

En la figura se observan los distintos conjuntos de señales, que se describen a continuación:

Daisy Chain signals

De la misma forma que para el componente "UDP_LCD", el componente "ARP" posee un conjunto denominado

"Daisy Chain Signals" que incluye señales entrantes y salientes del mismo, con la misma función que en el componente previamente presentado. Las señales entrantes resultan:

- DA_TX_PREV.
- LENGTH_PREV.
- DATA_TX_PREV.

Y las señales salientes:

- DA_TX_NEXT.
- LENGTH_NEXT.
- DATA_TX_NEXT.

En el conjunto entrante también están presentes las señales "DA_SELECTED_TX" y "DA_SELECTED_RX", utilizadas para la selección del eslabón a transmitir o recibir, respectivamente. Estas señales aseguran la compatibilidad con la lógica de difusión de la topología.

TX signals

A diferencia del componente "UDP_LCD", este módulo no posee memorias asociadas, ya que los campos de los mensajes a transmitir son datos fijos. Los únicos campos variables corresponden a la operación (solicitud o respuesta) y a la dirección IP destino, ambos valores conocidos en tiempo de ejecución.

Este conjunto de señales, tiene un subconjunto entrante y uno saliente. Para el caso de las entrantes tenemos:

- **RD:** Dedicada a la acción de lectura.
- **ADDRESS_RD:** Esta señal indica el byte que se desea leer.
- **DATA_TX_NEXT:** Dato que se desea leer.
- **TX_ENDED:** Señal que indica que la transmisión ha sido exitosa.

La señal saliente corresponde al acuse de recibo "ACK_TX". Todo este conjunto de señales vincula al componente "ARP" con el componente encargado de realizar la transmisión de datos.

RX signals

De la misma forma que para el caso de la transmisión, el componente ARP no se le asignó una memoria para almacenar los datos recibidos; en su defecto, cada vez que el componente de recepción hace escritura de un dato, este es registrado directamente por el componente "ARP".

Este conjunto de señales está conformado por un subgrupo entrante y uno saliente. Para el caso de las señales entrantes tenemos:

- **WR:** Dedicada a la acción de escritura.
- **ADDRESS_WR:** Esta señal indica el byte que se desea registrar.
- **DATA_WR:** Dato que se desea registrar.
- **CRC_CHECK, CRC_FAIL:** Señales utilizadas para validar los datos enviados.

La señal de salida es el acuse de recibo de los datos por parte del componente "ARP", denominada "ACK_RX". Todo este conjunto vincula al componente en cuestión con el componente encargado de la recepción de datos.

MAC's

Este conjunto de señales saliente está conformado por las señales denominadas "rMAC1" y "rMAC2", las cuales representan las direcciones físicas de la PC y el microcontrolador, respectivamente. Estas señales permiten la actualización dinámica de las direcciones en la red.

2) Lógica de funcionamiento

Al momento de instanciar el componente ARP, este posee "generics", denominados como: "DEVICE_ADDRESS", "MAC_ADDRESS", "IP_ADDRESS", "REFRESH_TIME" y "REPLY_TIME". Estos se utilizan para asignar la dirección de dispositivo asociada al componente, la dirección física y lógica de la FPGA, el tiempo de refresco para solicitar nuevamente las direcciones físicas y el tiempo de espera de respuesta por parte de los dispositivos que componen la red.

Por otro lado, el componente tiene la capacidad de efectuar solicitudes, transmitiendo "ARP Requests" a los dispositivos que integran la red, y de responder solicitudes transmitiendo "ARP Reply". Además, procesa las "ARP Reply" y "ARP Requests" recibidas. A continuación, se divide la explicación sobre el proceso de transmisión y de recepción.

Lógica de transmisión del protocolo ARP

El componente "ARP" tiene la capacidad de transmitir dos tipos de mensajes. Uno con el objetivo de solicitar la dirección física de los dispositivos que integran la red, y otro para responder a las solicitudes por parte de los mencionados dispositivos.

Al inicio del sistema, la FPGA que implementa el diseño desconoce las direcciones físicas de la PC y del microcontrolador. Para resolver esta situación, se diseñó una máquina de estados finitos (FSM) cuya función es emitir las solicitudes "ARP Request". Esta FSM está compuesta por cuatro estados: "RQ_PC", "RQ_uC", "Waiting" y "Finish" (ver Anexo A, Listado 5). A continuación se describe la función de cada uno:

- **RQ_PC:** Estado inicial de la FSM. En este estado se activa la señal interna "TX", lo que provoca la configuración de los campos variables del mensaje:
  - **OP:** definido como "ARP Request".
  - **Target MAC:** establecido en 000000000000, ya que se desconoce la dirección física destino.
  - **Target IP:** asignado con la dirección IP correspondiente a la PC.

  Una vez finalizada la transmisión de la solicitud —detectada mediante el flanco de la señal "TX_ENDED"— se produce una transición hacia el estado "RQ_uC" o "Waiting", dependiendo de si la dirección física del microcontrolador aún no ha sido resuelta o si ya se encuentra registrada, respectivamente.
- **RQ_uC:** El comportamiento es análogo al estado anterior, activando "TX" y configurando los mismos campos, salvo que en este caso el valor asignado a Target IP corresponde a la dirección lógica del microcontrolador. Finalizada la transmisión, la FSM pasa al estado "Waiting".
- **Waiting:** Durante este estado se inicia un temporizador cuyo valor está definido por el parámetro generic "REPLY_TIME". Al vencerse el temporizador, las transiciones posibles son:
  - RQ_PC, si no se recibió respuesta desde la PC.
  - RQ_uC, si no se recibió respuesta desde el microcontrolador.
  - Finish, si ambas direcciones físicas fueron correctamente resueltas.
- **Finish:** Estado final en el que se confirma que las direcciones físicas de los dispositivos fueron obtenidas exitosamente.

De esta manera, al inicio del sistema se garantiza la resolución de direcciones físicas mediante solicitudes "ARP Request". El procedimiento se repite de forma periódica, con una cadencia definida en el parámetro generic "REFRESH_TIME".

Finalmente, cuando se recibe una solicitud ARP Request proveniente de la PC o del microcontrolador, el componente activa la señal TX, asigna el valor REPLY al campo OP, y completa los campos Target MAC y Target IP con la dirección física y lógica del solicitante, respectivamente.

Mecanismo de recepción y procesamiento de ARP

En el caso de la recepción, el componente puede procesar dos tipos de mensajes provenientes de la PC o del microcontrolador: solicitudes (ARP Request) y respuestas (ARP Reply).

Durante este proceso, a medida que se reciben los datos del payload, el componente de recepción realiza la escritura secuencial de los mismos. El componente "ARP" intercepta dichas escrituras y, a partir de la dirección indicada por el módulo receptor, identifica a qué campo de la trama ARP corresponde cada dato recibido. De esta forma, los valores se registran internamente y se validan contra los campos esperados. Según el valor contenido en el campo de operación:

- **Request:** El componente genera y transmite una respuesta, siguiendo el procedimiento explicado en la subsección anterior.
- **Reply:** Los datos son almacenados y, en el caso de las direcciones físicas obtenidas, estas se disponibilizan a través de las salidas "rMAC1" o "rMAC2", dependiendo del dispositivo al que correspondan.

C. Daisy Chain

La topología Daisy Chain constituye el mecanismo de interconexión entre los distintos eslabones que conforman el sistema. Cada eslabón dispone de un conjunto de señales de entrada y salida —agrupadas como Daisy Chain signals— que permiten enlazar secuencialmente los módulos asociados a los periféricos.

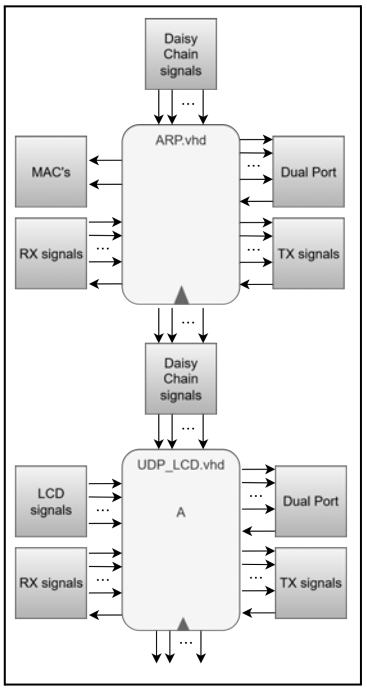

Fig. 7. Topología Daisy Chain.

En la Fig. 7 se esquematiza este concepto: las señales salientes (Daisy Chain signals) del componente "ARP" se conectan a las señales entrantes del eslabón correspondiente al periférico "A", y así sucesivamente hasta completar la cadena. Más allá de la conexión física, la topología introduce un mecanismo de arbitraje que define prioridades entre los eslabones.

La prioridad se establece de acuerdo con la posición relativa en la cadena. Cada eslabón, al transmitir sus datos propios como la dirección de dispositivo asociada al mismo, puede interrumpir la propagación de las señales entrantes para transmitir las propias. De este modo, sólo la dirección asociada al eslabón ubicado más adelante en la cadena logra llegar al final, garantizando un criterio jerárquico entre periféricos.

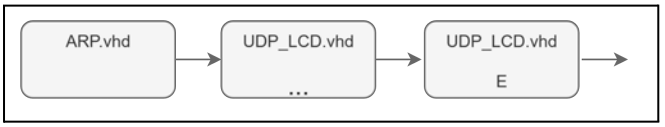

Fig. 8. Topología Daisy Chain - Prioridades

En la Fig. 8 se ilustra este mecanismo: mientras que el primer eslabón (ARP) posee la menor prioridad, el sexto eslabón (E) prevalece sobre los anteriores (ARP, A, B, C y D).

Este enfoque no solo permite gestionar prioridades de manera sencilla, sino que además ofrece escalabilidad. La incorporación de nuevos periféricos requiere únicamente la adición de un eslabón y las memorias asociadas, sin necesidad de rediseñar la lógica global.

Una vez completado el desarrollo digital de los componentes que integran la cadena, se procedió a verificar su correcto funcionamiento y a comprobar que se cumplieran los objetivos planteados. A continuación, se describe el proceso de verificación mediante testbench en SystemVerilog.

VII. Verificación

Cada eslabón incluye lógica para difundir datos entrantes o transmitir los propios. En el caso del componente "ARP", además de contar con dicha lógica, implementa el protocolo homónimo. Por esta razón, en una primera instancia se seleccionó a este componente como dispositivo bajo ensayo (DUT), ya que el objetivo de la verificación era comprobar, en primer lugar, que el componente tuviera la capacidad de implementar el protocolo ARP y de realizar la difusión, para poder ser integrado en la cadena.

Una vez que se comprobó que el componente poseía ambas capacidades, el DUT pasó a ser la Daisy Chain completa, continuando de esta manera hasta realizar una prueba integral del sistema.

La validación del componente "ARP" implementado en la FPGA es fundamental ya que sin él, la obtención de direcciones físicas no sería posible, lo que derivaría en la incapacidad de transmisión. Con el objetivo de comprobar el correcto funcionamiento se desarrolló un entorno de verificación en SystemVerilog, aprovechando su capacidad de modelar interacciones complejas entre los componentes encargados de realizar la transmisión y recepción de mensajes.

A. Metodología de verificación

La estrategia combinó pruebas dirigidas y pruebas aleatorias con restricciones, permitiendo validar la correcta respuesta del DUT frente a distintos escenarios de transmisión y recepción.

Las pruebas dirigidas se centraron en estimular al DUT con un paquete ARP de 28 bytes válidos, con campos fijos, verificando su comportamiento ante la llegada de paquetes "ARP Reply" y confirmando la obtención de las direcciones físicas correspondientes.

Por otro lado, las pruebas aleatorias restringidas variaron direcciones MAC e IP de origen y destino entre valores reales y falsos, emulando condiciones de operación y verificando la robustez del diseño.

Asimismo, se verificó que al inicio del sistema el componente realizara "ARP Request" a la PC y al microcontrolador, y que posteriormente un driver específico se encargara de dar las respuestas, comprobando la secuencia de solicitud y respuesta, así como los reintentos en caso de vencimiento del temporizador.

B. Componentes del testbench

El entorno de verificación se estructuró con los siguientes bloques principales:

- **Clase ARP_FRAME:** Define los campos de la trama ARP, con variables aleatorias (rand) para direcciones MAC e IP, y restricciones que limitan los valores a combinaciones válidas. Incluye campos fijos (HTYPE, PTYPE, HLEN, PLEN, OP) que aseguran el formato estándar.
- **Generator:** Instancia que, a partir de "ARP_FRAME", construye byte a byte el paquete ARP y lo envía a través de un mailbox, permitiendo encapsular el tráfico simulado.
- **DriverRX:** Recibe los datos generados y se encarga de su escritura para que el DUT los procese mediante la interfaz "ARP_if", gestionando las señales de control de escritura (WR, ADDRESS_WR, DATA_WR). Emula el comportamiento del componente receptor.
- **DriverTX:** Se encarga de leer los datos transmitidos por el DUT y gestionar el handshake entre el DUT y el componente de transmisión. Este driver simula el comportamiento del componente "transmision_dp".
- **Environment:** Integra generador y drivers, coordinando la ejecución concurrente mediante bloques fork-join. Esto permite simular la transmisión y recepción comprobando casos de funcionamiento real.
- **Interfaz ARP_if:** Encapsula todas las señales de comunicación con el DUT, incluyendo relojes, resets, buses de datos y señales de la topología Daisy Chain (DA_SELECTED, READY, ACK, TX_ENDED, RX_ENDED). Incluye además modports utilizados por los drivers.

C. Casos de prueba

Se implementaron pruebas específicas para validar los siguientes aspectos:

- **Recepción de tramas válidas:** Se realizó la generación y escritura secuencial de los 28 bytes de la trama ARP, confirmando que los campos de direcciones MAC/IP coincidieran con los valores esperados, obteniendo como salida por parte del DUT las direcciones físicas correspondientes a la PC y al microcontrolador. También se verificó que al momento de recibir paquetes "ARP Request", el DUT realizara la transmisión de la respuesta al dispositivo solicitante.
- **Pruebas con direcciones falsas:** Se generaron escenarios con direcciones MAC e IP inválidas para comprobar que el DUT descartara o procesara adecuadamente dichas tramas.
- **Interacción de TX/RX:** Se verificó que el componente "ARP", al inicio del sistema, realizara la transmisión de datos correspondiente a las solicitudes de dirección física hacia ambos dispositivos que componen la red, donde el "DriverTX" realizaba la lectura de los 28 bytes. Posteriormente, el "DriverRX" escribía los datos correspondientes a las respuestas, verificando que la secuencia completa se cumpliera en su totalidad.
- **Integración con la topología Daisy Chain:** Se verificó que el componente difundiera los datos entrantes cuando no era el seleccionado y, al serlo, transmitiera sus propios datos.

D. Resultados

El análisis de las formas de onda obtenidas durante la simulación demostró que el componente "ARP" realizó de forma satisfactoria las "ARP requests" a los dispositivos integrados en la red al inicio del sistema, reiterando dicha solicitud en caso de vencimiento del temporizador. Además, se comprobó que el componente posee la capacidad de refrescar periódicamente las direcciones físicas.

Por otro lado, al recibir paquetes válidos "ARP Reply", estos fueron procesados, la dirección física fue obtenida correctamente y presentada en el correspondiente puerto de salida. En caso de recibir un "ARP Request", el componente realizó la transmisión de la correspondiente "ARP Reply" al dispositivo solicitante.

Las pruebas aleatorias confirmaron la robustez del diseño frente a combinaciones variables de MAC e IP, sin errores de transmisión. Asimismo, se comprobó que el DUT respetó la lógica de difusión en la topología Daisy Chain, integrándose adecuadamente en el esquema jerárquico de prioridades.

Si bien la validación basada en formas de onda y la verificación funcional permitió verificar el correcto funcionamiento básico, se identificó como mejora futura la implementación de Hardware-in-the-Loop (HIL) [9], lo que permitiría automatizar la detección de errores y ampliar la cobertura funcional con interacción real de los componentes del sistema. Con ello se robustecerá el proceso de verificación de cara a futuras ampliaciones del sistema.

VIII. Detalles técnicos de implementación

El desarrollo digital del sistema se realizó íntegramente en VHDL, utilizando como dispositivo de destino una FPGA Xilinx Spartan-6 [10] XC6SLX25. Esta elección respondió a su disponibilidad de recursos y bajo consumo, ofreciendo 24.051 LUTs, 30.064 flip-flops y 52 bloques de RAM de 18 Kb (936 Kb en total), suficientes para implementar la lógica de los 8 eslabones de la topología Daisy Chain.

La Tabla I presenta la utilización de recursos de la topología Daisy Chain implementada, desglosando los componentes "UDP_LCD" estándar y el componente "ARP" especializado. El componente "ARP", debido a su complejidad funcional (FSM de 4 estados, temporizadores y validación de campos del protocolo), utiliza aproximadamente 41 veces más recursos lógicos que un eslabón "UDP_LCD" estándar. La topología completa, conformada

por 7 eslabones “UDP_LCD” y el módulo “ARP”, ocupa 772 LUTs (5%), 588 registros (2%) y 343 slices (9%) de los recursos disponibles en la FPGA.

TABLA I

RECURSOS UTILIZADOS - TOPOLOGÍA DAISY CHAIN

Componente	Cantidad	Slice Register	Slice LUTs	Slices
UDP_LCD	7	0	112	77
ARP	1	588	660	266
Total Daisy Chain	8	588	772	343
Recursos disponibles	-	30.064	15.032	3.758
Utilización (%)	-	2%	5%	9%

La interacción analógica y digital con los periféricos externos fue gestionada mediante una placa Nucleo-STM32-F429ZIT6 [11]. Este microcontrolador, perteneciente a la familia STM32F4 de STMicroelectronics, integra un núcleo ARM Cortex-M4 con FPU y opera hasta 180 MHz. Incorpora tres ADCs de 12 bits y dos DACs de 12 bits con resolución completa (0–4095), brindando la capacidad de conversión necesaria para la comunicación con los periféricos asociados.

IX. CONCLUSIÓN

El trabajo desarrolla y valida una solución práctica para la modernización de la CCSO mediante el empleo de una FPGA como intermediario, basando la estrategia en la topología Daisy Chain donde actúa como multiplexor de memorias dual port, permitiendo organizar y direccionar los datos asociados a cada periférico.

Los resultados alcanzados evidencian que la topología Daisy Chain proporciona un marco modular y escalable para la incorporación de nuevas funciones. La integración de un eslabón especializado para la ejecución del protocolo ARP constituye un aporte significativo: demuestra que la misma estructura puede evolucionar desde la multiplexación de memorias hasta la implementación de protocolos completos de comunicación, manteniendo la coherencia jerárquica y la robustez del esquema.

En términos de impacto, el sistema desarrollado preserva la compatibilidad con el hardware naval existente, elimina la dependencia de componentes propietarios y habilita la integración transparente con PCs y microcontroladores a través de Ethernet. De este modo, la arquitectura concebida no solo resuelve la problemática puntual de modernización de la CCSO, sino que establece un marco versátil, escalable y adaptable para futuras evoluciones del sistema.

ANEXOS

A. Fragmentos de descripción en VHDL

Listado 1 - Lógica de difusión de dirección de dispositivo.

```
DA_TX_NEXT <= THIS when READY_TX = '1' else
              DA_TX_PREV;
```

Listado 2 - Lógica de selección de dirección de dispositivo.

```
DA_SELECTED_TX_NEXT <=
"111111" when TX_ENDED = '1' else
DA_NEXT_TX when DA_SELECTED_TX = "111111" else
DA_SELECTED_TX;
```

Listado 3 - Lógica de difusión de longitud de payload.

```
LENGTH_NEXT<=
FRAME_LENGTH when DA_SELECTED_TX = THIS else
LENGTH_PREV;
```

Listado 4 - Lógica de difusión DATA y gestión de lectura.

```
DATA_TX_NEXT    <=
DATA_RD when DA_SELECTED_TX = THIS else
DATA_TX_PREV;

RD <=
READ_DS when DA_SELECTED_TX = THIS else
'0';
```

Listado 5 - FSM - ARP RQ.

```
Next_State     <=
RQ_uC    when (State = RQ_PC) and (TX_ENDED_PREV =
'0'  and  TX_ENDED  =  '1')  and  (rMAC2_int  =
x"FFFFFFFFFFFF") and (DA_SELECTED_TX = THIS) else

WAITING when (State = RQ_PC) and (TX_ENDED_PREV =
'0' and TX_ENDED = '1') and (rMAC2_int /=
x"FFFFFFFFFFFF") and (DA_SELECTED_TX = THIS) else

WAITING when (State = RQ_uC) and (TX_ENDED_PREV =
'0' and TX_ENDED = '1') and (DA_SELECTED_TX =
THIS) else

RQ_PC   when (State = WAITING) and (Tout_RLY =
'1') and (rMAC1_int = x"FFFFFFFFFFFF") and
(DA_SELECTED_TX = THIS) else

RQ_uC   when (State = WAITING) and (Tout_RLY =
'1') and (rMAC1_int /= x"FFFFFFFFFFFF") and
(DA_SELECTED_TX = THIS) else

FINISH  when (State = WAITING) and (CRC_CHECK =
'1') and (OP = REPLY) and (rMAC1_int /=
x"FFFFFFFFFFFF") and (rMAC2_int /=
x"FFFFFFFFFFFF") else

State;
```

REFERENCIAS

[1] C. Galasso, G. Friedrich, A. Antonini, y G. Diaz, “Desarrollo de un prototipo basado en FPGA,” en *IV Congreso de Microelectrónica Aplicada (CMA)*, Argentina, 2013, pp. 59–64.

[2] C. Galasso y G. Friedrich, “FPGA del Kit al prototipo,” en *Congreso Argentino de Sistemas Embebidos (CASE)*, Argentina, 2013, p. 50.

[3] E. Gallo, R. Cayssials, C. Galasso, y A. Arias, “Implementación de Daisy Chain en VHDL,” en *Congreso Argentino de Sistemas Embebidos (CASE)*, Argentina, 2025, pp. 23–26.

[4] Analog Devices, “Daisy Chain.” [Online]. Available: https://www.analog.com/en/resources/glossary/daisy_chain.html

[5] E. Gallo, R. Cayssials, C. Galasso, y A. Arias, “Lógica de funcionamiento Daisy Chain,” en *Congreso Virtual de Microcontroladores y sus Aplicaciones*, Argentina, 2025, aceptado para publicación.

[6] C. Galasso, A. Laiuppa, J. Ermantraut, S. Leoni, D. Martinez, y M. Paz, “Aplicación práctica de co-diseño de HW y SW para la apertura de un sistema de tiempo real,” en *53° JAIIO – SAIC, Simposio Argentino de Ingeniería en Computación*, Argentina, 2024, pp. 67–80.

[7] Texas Instruments, "BQ79616-Q1 Daisy Chain Communications," Application Report SLVAEP4, 2019. [Online]. Available: https://www.ti.com/lit/an/slvaep4/slvaep4.pdf

[8] M. M. Mano y M. D. Ciletti, “Digital Design with an Introduction to the Verilog HDL, VHDL, and SystemVerilog,” 6th ed., Pearson, 2017.

[9] Ansys, “What is Hardware-in-the-Loop Testing?” [Online]. Available: https://www.ansys.com/simulation-topics/what-is-hardware-in-the-loop-testing

[10] Xilinx Inc. “XC6SLX45T-2FGG484I – Spartan-6 FPGA Family,” Product Specification DS160, 2011. [Online]. Available: https://docs.amd.com/v/u/en-US/ds160

[11] STMicroelectronics, "STM32F429ZI High-performance STM32F4 MCU with DSP and FPU," Datasheet DS8626, Rev 9, 2024. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f429zi.pdf