Mejora de un laboratorio remoto de Electrónica Digital mediante el uso de una Raspberry Pi 4

Óscar Oballe-Peinado, Julián Castellanos-Ramos, José Antonio Sánchez-Durán, Rafael Navas-González, José Antonio Hidalgo-López
Departamento de Electrónica
Universidad de Málaga
Málaga, España
oballe@uma.es, jcramos@uma.es, jsd@uma.es, rjnavas@uma.es, jahidalgo@uma.es

***Abstract*—Tras la puesta en marcha de una primera versión de un laboratorio remoto de Electrónica Digital, surgen nuevas necesidades y mejoras, sobre todo en términos de escalabilidad y de nuevos periféricos. En este trabajo se presenta una migración, del conjunto del laboratorio remoto, a una Raspberry Pi 4. Las características de este miniordenador hacen que pueda actuar como servidor de aplicación, además de realizar la monitorización y la emulación de los periféricos de la placa de desarrollo donde se verifican los diseños.**

Keywords—e-learning, laboratorio remoto, Raspberry Pi, periféricos virtuales

I. Introducción

En la docencia de la electrónica, cada vez es más necesario disponer de un laboratorio remoto donde realizar prácticas a distancia [1]. En el caso de los laboratorios remotos de electrónica digital basados en placas de evaluación, la mayoría de las soluciones emplean una señal de video en tiempo real para poder observar el estado de los distintos periféricos del sistema [2]. Esta aproximación, además de costosa, requiere de un gran ancho de banda para las comunicaciones entre la aplicación cliente y el servidor.

En este trabajo, se propone una mejora sobre un laboratorio remoto que no precisa de una señal de video para la monitorización de los periféricos del sistema de desarrollo. En su lugar, representa el estado de los periféricos de una placa de desarrollo Nexys 3 [3], desarrollada por Digilent Inc. y basada en una FPGA, usando una representación virtual de los mismos a partir de los datos resultantes de una comunicación bidireccional entre los periféricos de entrada/salida de la placa y la aplicación web.

II. Descripción del Trabajo

El laboratorio remoto, en su primera versión, fue implementado usando un ordenador personal como servidor web y de aplicación [4]. Durante la fase de pruebas e implantación, ha tenido una gran aceptación por parte de docentes y estudiantes, que lo han usado como complemento a las sesiones presenciales de laboratorio [5]. Dados los buenos resultados obtenidos, se prevé su uso en cada vez más asignaturas y, por lo tanto, entre un mayor número de alumnos. Esto nos lleva a la necesidad de escalar el sistema para dar servicio a una mayor demanda de uso. Por todo ello, y tras analizar distintas opciones, se ha optado por el desarrollo del laboratorio remoto sobre una Raspberry Pi 4 [6], por la capacidad que ofrece a la hora de emular, a partir de sus pines de entrada/salida, dispositivos digitales externos que se puedan conectar a la Nexys 3 por los puertos de expansión para periféricos externos (Peripheral Modules, Pmod™), así como por su bajo coste y consumo (Fig. 1).

Este trabajo ha sido financiado por el Proyecto de Innovación Educativa PIE19-141 de la Universidad de Málaga

Así, la Raspberry Pi 4 ejerce como servidor web, para ofrecer a los clientes la interfaz del laboratorio remoto, y como dispositivo que es capaz de actuar sobre los periféricos de entrada básicos de la placa de desarrollo (botones e interruptores) y monitorizar los periféricos de salida (ledes y displays de siete segmentos).

Como se ha comentado, dado que la Raspberry Pi 4 tiene un puerto de expansión con pines digitales de entrada/salida, puede realizar la emulación de periféricos externos. Así, periféricos como un teclado hexadecimal, que anteriormente eran emulados por un microcontrolador desde una placa Arduino, pueden implementarse mediante procesos programados en la Raspberry Pi 4 utilizando algunos puertos digitales conectados a los Pmod™ de la Nexys 3. De la misma manera, se desarrolla, mediante rutinas que corren en la propia Raspberry Pi 4, un puerto serie que comunica el ordenador del cliente con la Nexys 3.

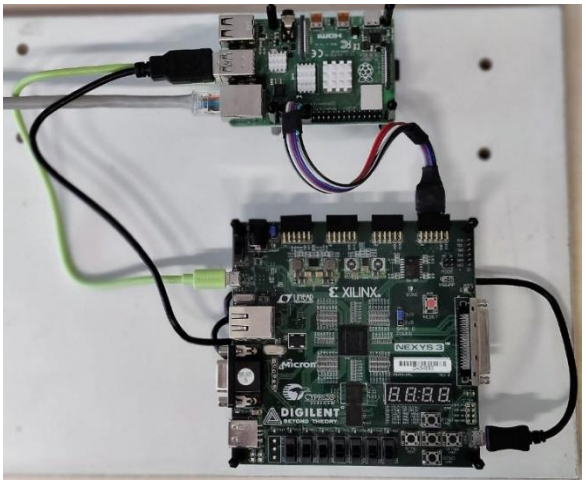

Fig. 1. Fotografía del sistema de laboratorio remoto basado en una Raspberry Pi 4.

Finalmente, se ha mejorado la funcionalidad del periférico de salida de los ledes, que inicialmente sólo eran capaces de mostrar dos estados, encendido o apagado. Ahora, permite mostrar 16 estados, que se les hace corresponder con 16 niveles de verde. De esa forma, se pretende emular el comportamiento de los distintos niveles de luminosidad que se aprecian en un led cuando se excita con una señal modulada en el ancho del pulso (Pulse-Width Modulation, PWM).

III. Adaptación del Sistema sobre una Raspberry Pi 4

La Raspberry Pi 4 consta de un procesador Broadcom BCM2711 Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz, con opciones de 2GB, 4GB y 8GB de memoria RAM. Posee 26 pines digitales de entrada/salida de propósito general y 4 puertos USB tipo A (2 USB 3.0 y 2 USB 2.0), así como un puerto Gigabit Ethernet. En nuestro caso, el

laboratorio remoto se ha desarrollado sobre el modelo con 4GB de RAM.

Todas estas características hacen que pueda considerarse como una solución "todo en uno" para el desarrollo del laboratorio remoto propuesto.

Dado que la Raspberry Pi 4 corre sobre un sistema operativo Linux (Raspberry Pi OS), el cual establece prioridades a los distintos procesos que se ejecutan de una manera pseudo paralela, se ha optimizado el sistema para que cumpla con las restricciones temporales que son necesarias. Las rutinas para la monitorización de los periféricos propios de la Nexys 3, así como las encargadas de la emulación de los periféricos externos, se han desarrollado en Python 3.

IV. Implementación de Periféricos

A continuación, se enumeran los periféricos mejorados o añadidos con respecto a la primera versión del laboratorio remoto.

A. Emulación de un Teclado Hexadecimal

El teclado hexadecimal es un periférico que estaba inicialmente emulado haciendo uso de una placa Arduino UNO basada en un microcontrolador. Dado que la Raspberry Pi 4 tiene puertos digitales de entrada/salida, y estos pueden ser controlados mediante software, se puede sustituir al microcontrolador realizando la oportuna programación de los puertos digitales para que se comporte como el teclado hexadecimal real. Hay que evaluar si la rutina programada, que corre bajo un sistema operativo, cumple con las restricciones temporales del circuito que realiza el barrido del teclado para la detección de la tecla pulsada.

El teclado real consta de una botonera dispuesta en una matriz de 4 filas y 4 columnas. Cuando se pulsa una tecla, se cortocircuita la fila y la columna correspondiente. El teclado se conecta a la Nexys 3 por uno de los puertos Pmod™. Desde el punto de vista de la Nexys 3, las filas se conectan a 4 pines configurados como señales bidireccionales, mientras que las columnas se conectan a 4 pines de entrada. El método para detectar la tecla pulsada consiste en realizar un barrido por las filas, colocando un nivel lógico "1" en una fila, mientras el resto de los pines conectados a las filas se ponen en alta impedancia "HZ". De esta forma, la columna (señal de entrada) que esté a nivel lógico "1", junto con la fila que en ese instante esté siendo seleccionada, definen de forma unívoca la tecla pulsada.

La rutina de detección de tecla pulsada debe realizar dos funciones:

- Detectar la tecla que el cliente ha pulsado.
- Generar las señales necesarias en las columnas para que la Nexys 3 detecte la tecla pulsada.

La primera tarea se realiza a través de código JavaScript y una conexión basada en WebSocket entre la aplicación web y el servidor. Una vez recibida, la información sobre la tecla pulsada es publicada en un servidor Redis que hace las veces de bróker de mensajes.

La segunda función, suscrita al canal Redis descrito anteriormente, corre 4 hilos de ejecución dentro de la rutina, uno por cada una de las filas. En cada hilo, se evalúa el valor de la fila y en función de la tecla pulsada que se deba emular, se activa o no la columna correspondiente. Desde el punto de vista de la Raspberry Pi 4 las 4 señales de las filas son entradas digitales, mientras que las 4 señales de las columnas son salidas digitales. En la Fig. 2 se puede ver un esquema de las conexiones entre la Raspberry Pi 4 y la Nexys 3.

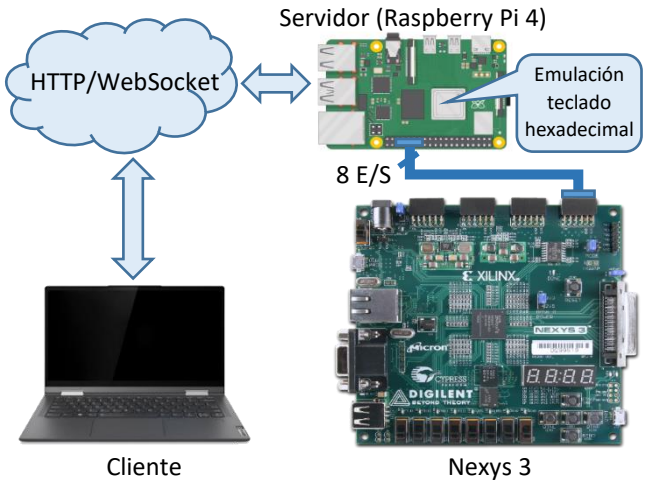

Fig. 2. Esquema de las conexiones entre la Raspberry Pi 4 y la Nexys 3 para la emulación del teclado hexadecimal.

B. Emulación de la Luminosidad de los LED

La modulación por ancho de pulso de una señal es una técnica habitual en electrónica digital para controlar la cantidad de energía que se envía a una carga. En el caso de un led, conseguimos que éste presente mayor o menor intensidad lumínica utilizando una salida digital. Sólo hay que tener en cuenta que la frecuencia de la señal debe ser lo suficientemente alta para que el ojo humano no perciba el parpadeo (los cambios de la señal digital de valor lógico "1" a "0" y viceversa), lo que se consigue a partir de los 50Hz, aproximadamente.

En las prácticas que se realizan en las asignaturas de electrónica digital a las que van dirigidas el uso del laboratorio remoto, se emplea esta estrategia para controlar la luminosidad de los ledes de la placa de evaluación Nexys 3.

Como cabía esperar, el laboratorio remoto es incapaz de monitorizar el estado de los ledes cuando son excitados con este tipo de señales y transmitir el mismo efecto a los elementos gráficos que representan dichos ledes en la aplicación web. Cuando un led es excitado con la señal PWM está cambiando su estado a una frecuencia superior a 50Hz. Aunque el ancho de banda empleado por los datos en las comunicaciones es muy bajo, por las características del sistema, es imposible mostrar los cambios sin perder información. El efecto es que el indicador del led parpadea, pero no a la frecuencia que lo está haciendo realmente.

Para solucionar este problema, en vez de enviar el estado real de los ledes (on/off), se calcula localmente (en la FGPA) un valor proporcional al ciclo de trabajo de la señal PWM que excita a los ledes. De esta manera, el cliente recibe este valor, y los indicadores de los ledes muestran distintos niveles del color seleccionado en función de dicho valor.

La información que se transmite a los indicadores es almacenada en unos registros. En la Nexys 3 hay 8 ledes, por lo que inicialmente se empleaba un registro de 1 byte para almacenar la información de su estado (1 bit por led). Para realizar el cambio propuesto, se propone almacenar la información de cada led en un registro de 4 bits, permitiendo

así detectar hasta 16 niveles de luminosidad distintos. Esto obliga a aumentar el tamaño de los registros necesarios de 1 a 4 bytes.

Por otro lado, el módulo encargado de almacenar la información en el registro de indicadores y controles debe realizar la tarea previa de calcular el valor proporcional al ciclo de trabajo de la señal de salida de los ledes en un registro de 4 bits. Se trata de un circuito digital capaz de medir la frecuencia y los ciclos en alto o bajo de una señal. En la Fig. 3 se muestra un diagrama de bloques de dicho circuito con sus entradas y salidas.

Si una señal es continua con valor lógico “0”, el valor proporcional almacenado para ese led debe ser “0”. Por el contrario, si el valor de la señal es continuo con valor lógico “1”, el valor almacenado debe ser 15 (valor máximo representado con 4 bits). Cualquier señal PWM debe ser representada con algún valor comprendido entre el 0 y el 15.

Para que el sistema sea eficiente y no agotar los recursos propios de la FPGA, dicho módulo se debe realizar con el menor número de recursos posibles (evitando multiplicadores y divisores, por ejemplo). Para poder hacer esto, es necesario imponer una serie de restricciones, pero que a su vez permita un resultado aceptable. La primera de ellas es imponer que la salida del módulo del cálculo del ciclo de trabajo sea de 4 bits, permitiendo 16 niveles de luminosidad, y otra es limitar el rango de frecuencias de las señales PWM con las que trabajar (6,25MHz – 40Hz). Por ello, las prácticas que incluyan señales PWM deben tener en cuenta estos aspectos.

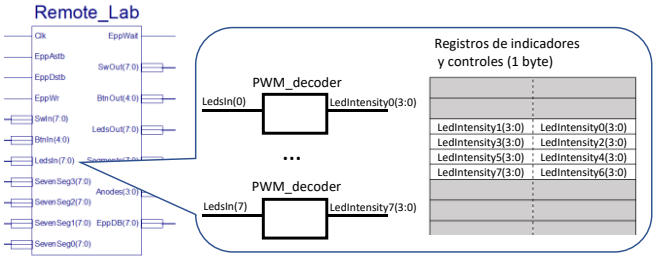

Fig. 3. Diagrama del módulo para el cálculo del ciclo de trabajo de la señal PWM que excita a los ledes.

C. *Emulación de un Puerto Serie*

La Nexys 3 posee un puerto USB-UART para implementar comunicaciones serie. Ésta es una función que se usa en algunas de las prácticas de las asignaturas donde se usa el laboratorio remoto, permitiendo comunicar la Nexys 3 con un ordenador para, por ejemplo, mostrar mensajes por pantalla de los datos recibidos desde la placa de desarrollo, o bien para que desde el ordenador se puedan enviar datos usando, por ejemplo, el teclado.

La Raspberry Pi 4 tiene puertos USB que pueden ser usados para enviar y recibir datos mediante un protocolo serie. La implementación del puerto serie se realiza mediante rutinas programadas en Python 3 que corren bajo el sistema operativo Raspberry Pi OS en la Raspberry Pi 4. Son dos las funciones que se deben realizar:

- Comunicar la Raspberry Pi 4 con la Nexys 3 mediante el puerto USB-UART.
- Conectar el cliente web con la Raspberry Pi 4 mediante una comunicación bidireccional usando WebSocket.

La primera tarea se implementa creando una conexión serie, siguiendo un protocolo RS232 con transmisión de 8 bits de datos, 1 bit de stop, sin bit de paridad ni señales de control de flujo, y a una velocidad de 115200 baudios.

Por otro lado, mediante el mismo sistema de suscripción a mensajes usado anteriormente (Redis) se establece la comunicación entre el servidor, que recibe la información del cliente web, y la rutina de la Raspberry Pi 4 que implementa el puerto serie.

De esta forma cuando desde la aplicación web se envía un dato, éste es recibido por la rutina, que a su vez lo transfiere a la Nexys 3 mediante el puerto serie abierto de la Raspberry Pi 4. Si es la Nexys 3 la que envía un dato, éste es recibido por la rutina y lo transfiere al servidor Websocket mediante el sistema de mensajes Redis. En la Fig. 4 se observa un esquema de las conexiones establecidas para la implementación de este periférico.

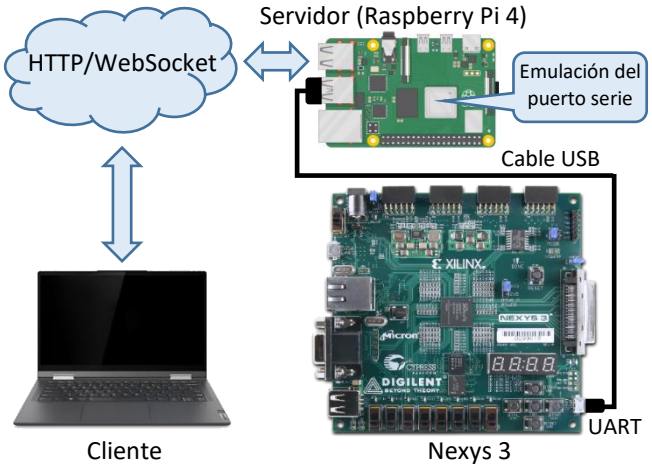

Fig. 4. Esquema de las conexiones para la implementación del puerto serie.

V. Resultados

La migración de todos los servicios del servidor necesarios para el correcto funcionamiento del laboratorio remoto (HTTP/HTTPS, WebSocket, PostgreSQL, Redis, etc.) han sido testados en la Raspberry Pi 4, obteniendo unos resultados de rendimiento más que satisfactorios en cuanto a carga de trabajo de la CPU y uso de memoria RAM.

Por otro lado, el módulo hardware que analiza las señales con las que se excitan los ledes de la placa de desarrollo ha sido diseñado para codificar, mediante 16 posibles valores, la luminosidad que ofrecen en el sistema real y, de esta forma, poder emularla, mediante distintos niveles de color en las réplicas de dichos ledes en la aplicación del laboratorio remoto. En este caso, para minimizar el uso de recursos de la FPGA destinados a dicha función, se establece un rango de frecuencias PWM concreto.

En relación con la capacidad de emular periféricos externos mediante procesos que leen y escriben sobre los pines digitales de entrada/salida de la Raspberry Pi 4, se ha desarrollado uno que hace las veces de un teclado hexadecimal. Se comprueba que es posible sustituir con ello a un microcontrolador que realizaba dicha tarea en la primera versión del laboratorio remoto, aunque se ha observado que la frecuencia de barrido del teclado por parte de la FPGA influye en el tiempo de CPU de los procesos que emulan el comportamiento del teclado. Este detalle debe ser considerado

en el enunciado de las prácticas, de forma que la frecuencia de barrido empleada sea la menor posible, siempre que la lectura del teclado se realice correctamente.

Para finalizar, el proceso encargado de las comunicaciones USB-UART ha sido desarrollado sin muchas complicaciones a partir de las funciones propias de Python 3. Dado que el ancho de banda de este protocolo serie es muy bajo. El sistema posee recursos para una correcta emulación del periférico.

VI. Conclusiones

En este trabajo se presenta un laboratorio remoto de electrónica digital implementado sobre en una placa de evaluación basada en una FPGA. El laboratorio se ha desarrollado sobre una Raspberry Pi 4 que incluye un servidor web y de aplicación, para el uso a distancia del sistema; un servidor de base de datos, para la gestión de usuarios y uso; y procesos que monitorizan y actualizan en ambos sentidos los periféricos del sistema de desarrollo sobre el que está basado.

La Raspberry Pi 4 es un sistema compacto, barato y de bajo consumo, con capacidad suficiente para el correcto funcionamiento del laboratorio remoto. Esto permite que pueda ser fácilmente escalable y, de esta forma, se pueda atender a un mayor número de usuarios concurrentes cuando así se precise.

Por otro lado, se han mejorado y añadido alguno de los periféricos de los que consta el laboratorio remoto:

- Se ha mejorado la representación de los ledes en la aplicación web. Un módulo analiza las señales con los que son excitados y, en el caso de ser señales PWM, obtiene 16 valores distintos permitiendo mostrar distintas tonalidades en función del ciclo de trabajo de la señal.
- Se ha emulado el comportamiento de un periférico externo (teclado hexadecimal) desde los propios pines de entrada/salida de la Raspberry Pi 4. Hasta el momento, esta tarea se realizaba desde un microcontrolador, por lo que el coste se reduce de forma significativa.
- Se ha añadido un nuevo periférico, el puerto serie, por su versatilidad en las prácticas de asignaturas donde se instancian microcontroladores en la propia FPGA.

Por tanto, con todas estas prestaciones, se puede concluir que la apuesta por el desarrollo de esta actualización del laboratorio remoto, basada en una Raspberry Pi 4, ha sido una decisión acertada que permite la escalabilidad del sistema y la mejora del mismo, incluyendo la emulación de nuevos periféricos como una solución “todo en uno”.

Referencias

[1] P. Orduña *et al.*, “The WebLab-Deusto Remote Laboratory Management System Architecture: Achieving Scalability, Interoperability, and Federation of Remote Experimentation,” in *Cyber-Physical Laboratories in Engineering and Science Education*, Cham: Springer International Publishing, 2018, pp. 17–42.

[2] C. A. Mayoz, A. L. Da Silva Beraldo, A. Villar-Martinez, L. Rodriguez-Gil, W. F. M. De Souza Seron, and P. Orduna, “FPGA remote laboratory: Experience of a shared laboratory between UPNA and UNIFESP,” *Proc. - 2020 14th Technol. Appl. to Electron. Teach. Conf. TAEE 2020*, Jul. 2020, doi: 10.1109/TAEE46915.2020.9163773.

[3] Digilent Inc., “Nexys 3 Spartan-6 FPGA Trainer Board - Digilent.” https://digilent.com/reference/programmable-logic/nexys-3/start (accessed Mar. 14, 2021).

[4] Ó. Oballe-Peinado, J. Castellanos-Ramos, J. A. Sanchez-Durán, R. Navas-González, A. Daza-Márquez, and J. A. Botín-Córdoba, “FPGA-Based Remote Laboratory for Digital Electronics,” in *2020 XIV Technologies Applied to Electronics Teaching Conference (TAEE)*, Jul. 2020, pp. 1–5, doi: 10.1109/TAEE46915.2020.9163676.

[5] Ó. Oballe-Peinado, J. Castellanos-Ramos, R. Navas-González, J. A. Sánchez-Durán, D. Rosas-Cervantes, and A. Daza-Márquez, “Evaluación de un Laboratorio Remoto de Electrónica Digital,” in *5th International Virtual Conference on Educational Research and Innovation*, 2021, pp. 344–348, [Online]. Available: https://www.civinedu.org/conference-proceedings-2021/.

[6] “Raspberry Pi 4 Model B.” https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (accessed Mar. 14, 2022).