Plataforma de desarrollo de bajo coste para implementar circuitos digitales en FPGAs mediante hardware y software libre

Albert Saiz-Vela, Pau Fontova, Tomàs Pallejà, Marcel Tresanchez, Juan Antonio Garriga, Concepció Roig
Departament d'Informàtica i Enginyeria Industrial
Escola Politècnica Superior, Universitat de Lleida (UdL)
Lleida, Spain
albert.saiz@udl.cat

***Resumen*—La implementación práctica de sistemas digitales en las sesiones de laboratorio de la asignatura de electrónica digital (y afines) puede realizarse mediante circuitos integrados (CIs) monolíticos (series TTL 7400 o CMOS 4000, por ejemplo) o mediante dispositivos de hardware reconfigurable como son las FPGAs. En la primera opción, los estudiantes suelen tener dificultades asociadas al proceso de montaje y testeo debido al gran número de conexiones cableadas que hay que implementar, mientras que, en la segunda, las dificultades suelen estar asociadas al acceso limitado y/o restringido que los estudiantes tienen para poder utilizar las placas de desarrollo hardware y el software para programarlas, ya que acostumbran a estar en el laboratorio y no siempre están a su disposición. Para intentar superar ambas problemáticas, este trabajo presenta una plataforma de desarrollo de bajo coste para la implementar circuitos digitales en FPGAs que: 1) es fácilmente replicable, 2) minimiza el número de conexiones cableadas y 3) está basada en el uso de herramientas de hardware y software libre. El objetivo que se pretende conseguir mediante esta estrategia metodológica es que los estudiantes puedan disponer en su casa de los recursos necesarios para que puedan practicar en el uso de las FPGAs sin ninguna limitación espacio-temporal y focalicen su atención en los aspectos teórico-funcionales de los circuitos y sistemas diseñados dejando en un segundo plano (pero sin obviarlos) los aspectos asociados a los procesos de montaje, cableado y test.**

***Palabras clave*—FPGA, electrónica digital, Verilog, Software y Hardware de código abierto / libre.**

I. INTRODUCCIÓN

El proceso de aprendizaje "clásico" de los estudiantes de un grado de ingeniería en relación con las asignaturas que contienen conceptos asociados a la electrónica digital se ha basado tradicionalmente en el uso de simuladores lógicos y su posterior implementación física para verificar que los resultados simulados se ajustan a los resultados experimentales obtenidos. Este ha sido el proceso que se ha llevado a cabo en la Escuela Politécnica Superior de Ingeniería de nuestra Universidad en los últimos 25 años. En nuestro caso, la implementación práctica de los circuitos y sistemas digitales propuestos en las actividades de laboratorio de la asignatura de electrónica digital (y afines) se ha realizado históricamente mediante el uso de circuitos integrados monolíticos como los de las series TTL 7400 o CMOS 4000, cables de conexión y el uso de placas de prototipado (más conocidas como *protoboards*). Este enfoque didáctico permite la reutilización del material, el coste es muy bajo, la curva de aprendizaje en relación a los procesos de montaje y testeo de los circuitos es rápida, es una metodología que ya suele ser conocida por nuestros estudiantes al realizar las prácticas de la asignatura de electrónica digital (ya que han trabajado utilizando el mismo enfoque en otras asignaturas del currículum en cursos inferiores: fundamentos de ingeniería electrónica, teoría de circuitos, etc…) y permite que nuestros estudiantes tengan la percepción que su aprendizaje es significativo y aprendan mediante el paradigma del *"learning by doing"* ya que el objetivo al finalizar las sesiones prácticas de laboratorio es que hayan podido diseñar e implementar físicamente un sistema digital (combinacional y secuencial). Sin embargo, a pesar de las ventajas de esta metodología, ciertamente presenta algunas dificultades asociadas al proceso de montaje y testeo, tal y como explicitan sistemáticamente (frecuentemente en forma de queja) los estudiantes cada año durante la realización de las prácticas. Si el sistema digital necesita más de cinco circuitos integrados monolíticos para ser implementado, el número de cables necesarios para conectar todos los bloques lógicos suele ser muy elevado tal y como se muestra en la Fig. 1. Además, la posibilidad de cometer errores de conexión se incrementa a medida que aumenta la complejidad de los sistemas digitales y se van añadiendo circuitos integrados. En este caso, a menudo son necesarias las dos horas de una sesión de laboratorio para hacer todo el montaje de los circuitos y, a veces, una vez se ha montado, el circuito no funciona y es extremadamente difícil encontrar el error de montaje. El hecho de que los estudiantes pasen dos horas realizando una tarea rutinaria resta tiempo para que éstos puedan debatir y analizar con el profesor en el laboratorio cuestiones relacionadas con el diseño y funcionamiento del circuito.

Una posibilidad para solucionar esta problemática es la implementación de los circuitos mediante el uso del hardware reconfigurable de las FPGAs (*Field Programmable Gate Array*) donde las conexiones entre bloques digitales se hacen internamente en el propio circuito integrado. De hecho, en muchas universidades y escuelas de ingeniería de todo el mundo, la implementación práctica de los sistemas digitales en las sesiones de laboratorio se realiza mediante esta opción [1]-[7]. Históricamente el mercado de las FPGAs ha estado dominado per tres fabricantes: Xilinx ®, Altera ® - adquirida por Intel ® - y en menor medida Lattice Semiconductor ® [8]. Este hecho ha provocado que las placas de desarrollo educa-

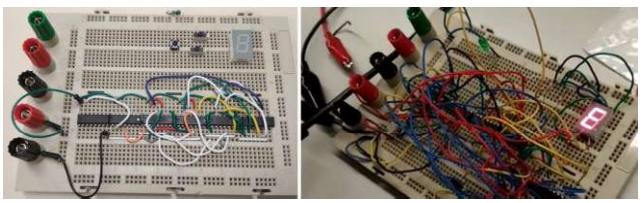

Fig. 1. Evolución de la implementación en el laboratorio de un sistema digital de complejidad media-alta mediante el uso de circuitos integrados monolíticos, cableado y protoboard.

cionales de estos fabricantes tengan un coste elevado. No obstante, aunque existen opciones "económicas" de kits/placas de desarrollo elaboradas por otras compañías a precios asequibles (de 70 € a 135 € por unidad en función del modelo de placa) como las placas DE0-Nano y Altera DE0 de Terasic ® [9] con FPGAs de Intel ® o las places Basys 2 Spartan-3E y Basis 3 Artix-7 de Digilent ® [10] con FPGAs de Xilinx ®, su uso normalmente requiere que los estudiantes utilicen un software privativo/propietario para programar la FPGA que suele requerir el registro previo en la web del fabricante de la FPGA para poder obtener una licencia gratuita para fines educacionales y poder utilizar el software de forma gratuita en algunos modelos concretos de FPGA.

Tradicionalmente el ecosistema alrededor de las FPGAs ha sido un entorno basado en soluciones de software propietarias de código privativo y hardware cerrado. Sin embargo, la influencia del movimiento *maker*, que desde el año 2005 aboga por el uso de tecnologías libres y abiertas, tanto a nivel de código como a nivel de hardware, es cada vez mayor en el ámbito tecnológico, industrial y académico. El éxito rotundo de plataformas *low-cost* como Arduino, Raspberry Pi, las impresoras 3D y las placas de sensores y/o actuadores alrededor de estas tres plataformas, y la aparición de una amplia comunidad de ingenieros, científicos, estudiantes y aficionados *makers* que comparten libremente a través de internet el conocimiento, tanto a nivel de código como a nivel de diseño hardware, está cambiando el paradigma del desarrollo hardware ya sea en el ámbito educativo, empresarial y/o en el ámbito de la investigación que se realiza en las escuelas de ingeniería.

Las FPGAs no ha sido inmunes a la influencia del tsunami provocado por el movimiento *maker*. En 2015, la ingeniera austriaca Claire Wolf (anteriormente conocida como Clifford Wolf) desarrolló mediante técnicas de ingeniería inversa un conjunto de herramientas de software capaz de programar los modelos FPGA iCE40 LP/HX 1K/4K/8K del fabricante Lattice Semiconductor ® y creó el proyecto IceStorm [11] donde obtuvo la primera *toolchain* de código libre que permitió programar una FPGA. Si bien en el ambiente *maker* estas FPGAs son conocidas como FPGAs libres, el concepto de "libre" es en este caso discutible, ya que Lattice Semiconductor ® no ha liberado ninguna información sobre la estructura interna de sus FPGAs, su formato de bitstream para poder programarlas, etc…. No obstante, tampoco ha puesto ningún impedimento legal para poder utilizar sus FPGAs con la *toolchain* del proyecto IceStorm. De hecho, no solo no ha retirado del mercado las FPGAs de la familia iCE40, sino que para Lattice Semiconductor ®, la aparición del proyecto IceStorm ha supuesto un aumento de popularidad significativo (intuimos que también en ventas) siguiendo el mismo proceso que supuso para los microcontroladores ATmega de Atmel ® (adquirida por Microchip ®) la aparición de la plataforma Arduino [12].

Así pues, la irrupción de una *toolchain* libre que permite programar FPGAs coincidiendo en el tiempo con el auge del movimiento *maker*, ha provocado la reciente aparición de numerosos proyectos alrededor de las FPGAs de la familia iCE40 siguiendo el paradigma del hardware libre y el software de código abierto. Proyectos donde se dispone de toda la información en relación al diseño hardware (esquemas, firmware, ficheros gerber de fabricación PCB, *Bill of Materials*, etc…) para poderlos replicar, modificar, compartir, etc.... y que, en algunos casos, han desembocado en la creación de pequeñas placas de entrenamiento con conectores de expansión que puede ser adquiridas de forma comercial a un precio muy competitivo (entre 35-60 €, IVA incluido). Es en este contexto donde han aparecido placas de entrenamiento como las Icezum Alhambra II, tinyFPGA, iCEBreaker, icoBoard, Upduino, BlackIce Mx, la iCE40HX8K-EVB, etc...[13].

Si nos centramos en las motivaciones que aparecen en la literatura recientemente publicada [14] para utilizar soluciones de software y hardware libre en el ámbito educativo, éstas son recurrentes: 1) se mejora la metodología docente ya que puede proveerse a los estudiantes de recursos para que puedan entender los conceptos abstractos asociados a la ciencia y a la ingeniería, 2) se reducen los costes ya que las tecnologías abiertas y libres tienden a ser más baratas que sus alternativas propietarias de forma que los costes de los laboratorios de prácticas disminuyen (un hecho especialmente favorable para las universidades de países subdesarrollados y/o con pocos recursos económicos), 3) los diseños hardware son libremente compartidos sin tener que pagar ninguna licencia por su uso, 4) se incrementa la creatividad y el interés de los estudiantes ya que pasan de un rol pasivo de usuarios tecnológicos a un rol activo de *makers* y 5) es posible el uso de los recursos hardware más allá del laboratorio en la universidad ya que éstos pueden ser adquiridos por los estudiantes a un bajo precio y/o incluso pueden ser desarrollados y fabricados por ellos mismos. De hecho, cada vez son más las universidades y escuelas de ingeniería que utilizan dichos recursos libres para implementar sus prácticas docentes en diferentes ámbitos y asignaturas [14]-[20]. Teniendo en cuenta estos argumentos y considerando, además, el bajo precio de adquisición que tienen algunas de las pequeñas placas de entrenamiento libres previamente mencionadas, se decidió apostar por el ecosistema de hardware y software libre para realizar la plataforma de desarrollo de bajo coste que se presenta en este trabajo. Este hecho no significa que consideremos "mejor" el uso de herramientas libres sobre las privativas para programar FPGAs (desde el punto de vista estrictamente pedagógico). Sin embargo, creemos y que el uso de herramientas libres para la programación de FPGAs puede suponer, en el ámbito educativo, un impacto similar al que ha tenido la plataforma Arduino [21] en la programación de microcontroladores.

El presente artículo está organizado de la siguiente forma: la Sección II describe el funcionamiento de la *toolchain* del proyecto IceStorm y muestra el conjunto de herramientas libres que permiten implementar fácilmente un sistema digital en las FPGAs de la familia iCE40, la Sección III muestra las características de la pequeña placa de entrenamiento con conectores de expansión que ha sido utilizada como núcleo de nuestra plataforma de desarrollo, la Sección IV muestra las características de la plataforma de desarrollo y un ejemplo de implementación de un sistema digital de complejidad media. Finalmente, la Sección V muestra la aplicación de la plataforma de desarrollo en una sesión de laboratorio real y los resultados de una encuesta realizada a nuestros estudiantes para poder analizar de forma preliminar el impacto de nuestra propuesta en su proceso de aprendizaje.

II. El proyecto IceStorm y el ecosistema de software/hardware libre en el ámbito de las FPGAs

La *toolchain* del proyecto IceStorm permite mediante una secuencia preestablecida de órdenes y programas sintetizar, implementar (*place and route*) y grabar los ficheros bitstream

en la FPGA tal y como se muestra en la Fig. 2. La síntesis se realiza mediante el programa *Yosis*, que es el encargado de convertir código de descripción hardware en lenguaje Verilog (*.v) a una *netlist* en formato BLIF (Berkeley Logic Interchange Format) (*.blif). La implementación (*place and route*) se realiza mediante el programa *nextpnr* que acepta como entradas ficheros en formato BLIF y genera un fichero en formato ASCII (*.asc) que contiene bloques de 0 y 1 para los bits de configuración de cada celda de la FPGA. Finalmente, el programa *icepack* convierte el fichero en formato ASCII generado por el programa *nextpnr* en un fichero binario (*.bin) y el programa *iceprog* permite grabar la información binaria en la FPGA. Cabe destacar que, aunque actualmente el programa *Yosis* únicamente permite realizar la síntesis mediante el uso del lenguaje de descripción hardware Verilog, los ingenieros Tristan Gingold y Pepijn De Vos están trabajando en el desarrollo de un *plug-in* que permita al programa *Yosis* sintetizar circuitos descritos mediante el lenguaje VHDL, aunque hay que remarcar que el estado del proyecto todavía está en una fase *beta* experimental [22].

En la Fig. 3 se muestra la placa de entrenamiento que ha sido utilizada en este trabajo como núcleo de la plataforma desarrollo de bajo que se presente en este trabajo: la placa Icezum Albrambra II diseñada y fabricada por la empresa española Alhambrabits ® [23]. El coste de la placa (60 €, IVA incluido), las facilidades para poder adquirirla, la disponibilidad de stock y sus características técnicas han sido motivos determinantes a la hora de escogerla como opción preferente respecto otras placas.

Otro de los factores que ha provocado la popularidad y el aumento del interés en las FPGAs de la familia iCE40 (y sus placas de entrenamiento) ha sido la posibilidad de utilizar la *toolchain* libre del proyecto IceStorm mediante el editor visual de esquemas Icestudio [24] que es un proyecto de software libre desarrollado y financiado inicialmente en el año 2016 por la empresa BQ ® y que a partir de 2017 viene siendo desarrollado y mantenido de forma altruista por la comunidad maker #FPGAWars liderada por Juan Gonzalez-Gomez (más conocido como @Obijuan) [25]. El editor Icestudio permite diseñar sistemas y bloques digitales complejos a partir de puertas lógicas básicas y dispositivos combinacionales y/o secuenciales simples (multiplexores, codificadores, decodificadores, flip-flops, etc…) de forma que se puede ir

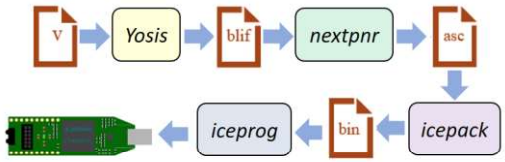

Fig. 2. *Toolchain* libre del proyecto IceStorm.

Fig. 3. Placa Icezum Alhambra II de la empresa Alhambrabits ®.

incrementando la complejidad de los bloques aplicando el paradigma de diseño hardware conocido como *bottom-up* hasta obtener el diseño del sistema digital completo, y, además, permite fácilmente la inclusión en el editor de esquemas de bloques diseñados con código Verilog embebido / integrado en el propio bloque. Esta última posibilidad es muy interesante a nivel docente ya que permite que se implementen los circuitos digitales mediante esquemas lógicos, mediante el lenguaje de descripción hardware Verilog o mediante una combinación de ambos tal y como se puede observar en la Fig. 4.

Una vez diseñado el circuito, el editor visual Icestudio internamente utiliza una llamada a la *toolbox* de software libre y multiplataforma Apio que permite verificar, sintetizar y cargar el diseño en la FPGA mediante la aplicación secuencial de los programas de la *toolchain* del proyecto IceStorm de forma transparente para el usuario [26]. Hay que remarcar que el editor visual de esquemas Icestudio es capaz de trabajar con un amplio número de placas de entrenamiento de FPGAs de la familia iCE40, entre ellas la que hemos escogido para este trabajo (Icezum Alhambra II). Este ha sido otro de los motivos determinantes para escoger esta placa de entrenamiento para el presente trabajo. En el caso de querer utilizar únicamente código Verilog para implementar los circuitos digitales y cargarlos en la FPGA directamente existe también la posibilidad de utilizar el entorno de desarrollo integrado Apio-IDE [27] que está basado en los proyectos de software libre Atom, Apio y PlatformIO tal y como se muestra en la Fig. 5. Finalmente, si se desea comprobar de forma simulada el correcto funcionamiento de los circuitos /bloques diseñados mediante el editor de esquemas Icestudio, existe la posibilidad de exportar el conjunto del sistema diseñado a código Verilog, generar los ficheros de *testbench* y mediante el simulador Icarus Verilog y el visor de señales GTKWave (ambos también software libre) generar y visualizar el cronograma correspondiente.

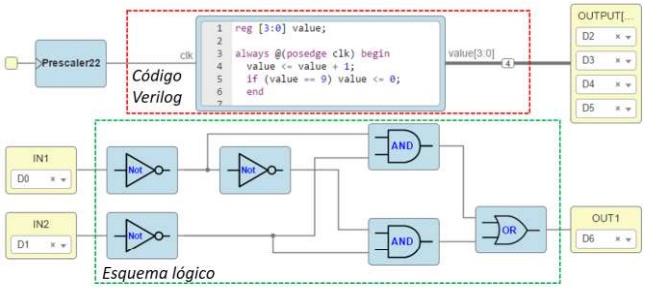

Fig. 4. Entorno de trabajo del editor visual de esquemas Icestudio (ejemplo de diseño realizado mediante la combinación de bloques implementados mediante esquemas lógicos y bloques código Verilog embebido).

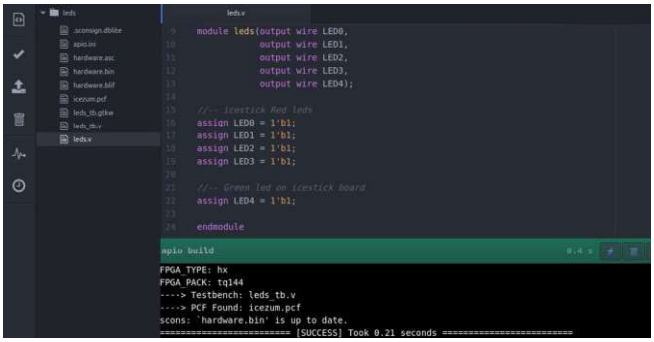

Fig. 5. Entorno de trabajo del editor integrado de desarrollo Apio-IDE (ejemplo de diseño implementado únicamente mediante código Verilog).

III. La placa Icezum Alhambra II

La Fig. 6 muestra el factor de forma y la distribución de pines de la placa de entrenamiento Icezum Alhambra II. Como se puede observar, el factor de forma de la placa Icezum Alhambra II y la distribución de pines de entrada/salida (E/S) coincide con el de la placa Arduino UNO [21]. Aunque este hecho pueda parecer una coincidencia sin importancia, en nuestro caso, se ha observado un especial interés por parte de los estudiantes al presentarles la placa Icezum Alhambra II y sus características. Creemos que, al ver por primera vez la placa Icezum Alhambra II, los estudiantes (especialmente aquellos que han tenido alguna experiencia con Arduino y su ecosistema de hardware abierto en etapas educativas inferiores como la secundaria) hacen una conexión directa entre ésta y la placa Arduino UNO y se sitúan en otro marco mental diferente al del de la típica (y aburrida) práctica de laboratorio de electrónica (que nunca sale bien) a la que suelen estar acostumbrados. Suponemos que las experiencias previas con el entorno del hardware libre han sido gratificantes y enriquecedoras para ellos (así lo indican cuando se les pregunta) y este hecho ayuda a que los estudiantes tengan una gran predisposición y atención para realizar los ejercicios prácticos de laboratorio mediante las FPGAs programables con herramientas libres. Cabe remarcar, además, que el hecho de compartir factor de forma y distribución de pines con la placa Arduino UNO permite la utilización de los numerosos *shields* del ecosistema Arduino, es decir, placas específicas de expansión que se conectan encima de la placa Arduino mediante los conectores de expansión y que añaden funcionalidades, periféricos, sensores, actuadores, etc…

La placa Icezum Alhambra II contiene una FPGA iCE40HX4K de Lattice Semiconductor ®. Si esta FPGA se programa mediante el software privativo iCECube2 ® del fabricante, el número de celdas básicas lógicas programables (Look-Up-Tables + Flip Flop) disponibles es de 3520. Sin embargo, si se utiliza la *toolchain* libre del proyecto IceStorm el número disponible de celdas básicas lógicas asciende a 7680 que corresponden al modelo iCE40HX8K. Realmente lo que sucede es que la FPGA iCE40HX4K es internamente una iCE40HX8K (modelo superior) que tiene limitado el número de celdas básicas programables por el propio software del fabricante. El hecho de utilizar la *toolchain* libre del proyecto IceStorm permite saltarse la limitación impuesta por el fabricante y disponer del doble de celdas básicas de programación. Así pues, con las herramientas libres es posible sintetizar en una FPGA iCE40HX4K sistemas digitales mucho más complejos que si se hace con las herramientas privativas, ya que el número de celdas básicas lógicas programables disponible es el doble. La programación de la FPGA puede realizarse vía USB mediante la UART ya que dispone del controlador USB FTDI ® 2232H.

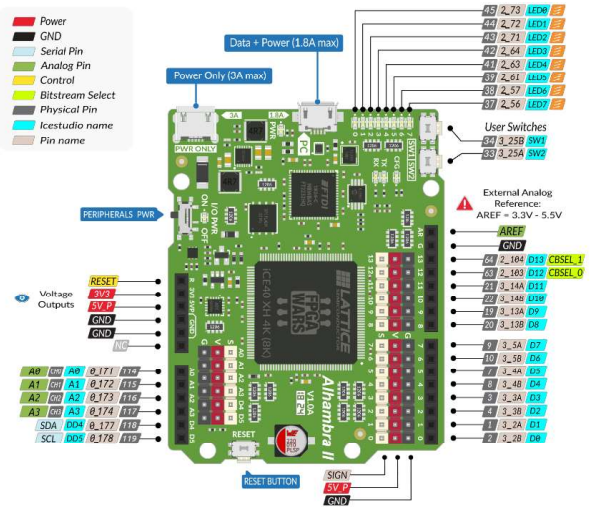

Fig. 6. Placa Icezum Alhambra II: factor de forma y distribución de pines.

A continuación, se enumeran las principales características técnicas de la placa Icezum Alhambra II:

- Interruptor ON/OFF para desconectar fácilmente los periféricos conectados a la placa.
- 8 LEDs SMD de propósito general (*LED0*…*LED7*).
- 2 pulsadores tipo *push-button* de propósito general (*SW1* y *SW2*).
- 32Mb de Memoria Flash para poder guardar hasta 30 bitstreams o datos del usuario.
- 20 pines de entrada / salida (E/S) (*D0*…*D13*, *A0*…*A3*, *DD4* y *DD5*) a 3.3 V (5 V tolerantes). Como entrada acepta niveles entre 3.3 V y 5V, como salida genera 3.3V.
- Resistencia de 200 Ω en serie en todos los pines de E/S para activación directa de LEDs.
- Conversor 12 bits A/D (4 canales)
- Los pines de selección de bitstreams para *cold boot* (a escoger entre 4 bitstreams que previamente hayan sido grabados en la memoria Flash) están accesibles mediante los pines de E/S de propósito general.
- Reguladores conmutados de 1A para las alimentaciones de 1.2 V y 3.3 V, lo que permite activar los PLLs y trabajar a velocidades mucho mayores.
- Oscilador MEMS de 12 MHz.
- Botón de reset tipo *push-button*.
- Alimentación de la placa mediante dos conectores USB (hasta 4.8 A máximos de entrada).
- Los pines de alimentación y E/S tienen protección permanente contra cortocircuito.

IV. La Plataforma de desarrollo de bajo coste

Aunque la placa Icezum Alhambra II dispone de 8 LEDs de propósito general y 2 pulsadores tipo *push-button*, estos periféricos no son suficientes para poder llevar a cabo el tipo de prácticas que tradicionalmente hemos llevado a cabo en la Escuela Politécnica Superior de Ingeniería de nuestra Universidad donde, normalmente, 1) se necesitan muchas más entradas de 1s y 0s mediante pulsador y/o interruptor (y la placa Icezum Alhambra II únicamente tiene dos pulsadores tipo push-button *SW1* y *SW2*) y 2) se requiere visualización de caracteres alfanuméricos mediante displays 7-segmentos (y la placa Icezum Alhambra II no dispone de ellos). Es verdad que se podría haber utilizado (aprovechando la compatibilidad de la placa Icezum Alhambra II con la placa Arduino UNO) alguno de los *shields* de bajo coste existentes en el mercado como por ejemplo los *shields* multifunción o similares que se muestran en la Fig. 7. Sin embargo, la utilización de dichos *shields* limita el uso de los pines de E/S de la FPGA y la crea-

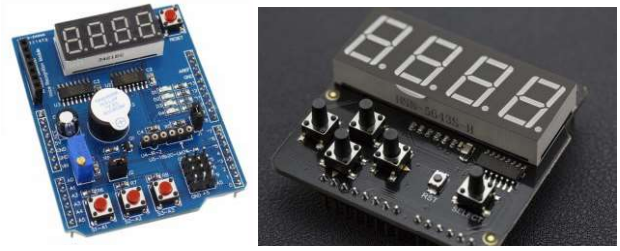

Fig. 7. *Shields* comerciales multifunción para placa Arduino UNO.

tividad y libertad a la hora de plantear un ejercicio práctico en el laboratorio ya que las conexiones entre los pines de E/S de la FPGA y los del *shield* viene predeterminada por el diseño hardware del *shield* en cuestión. Además, otro de los objetivos que ha perseguido este trabajo es que nuestros estudiantes dispongan de toda la información relacionada con la plataforma de desarrollo en un repositorio abierto (documentación, esquemas, firmware, ficheros gerber de fabricación PCB, *Bill of Materials*, etc…) siguiendo la filosofía de código abierto (software y hardware libre) que caracteriza al movimiento *maker*. De esta forma, aquellos estudiantes que quieran fabricarse su propia plataforma de desarrollo para poder practicar fuera de las horas de laboratorio tienen la posibilidad de fabricar su propia placa de circuito impreso (PCB) y soldar los componentes mediante el servicio de fabricación de PCBs del laboratorio de electrónica de nuestra Universidad (donde disponen de insoladora, taladros, ácidos, soldadores, etc…) o mediante la fabricación de la PCB en proveedores externos de bajo coste.

A. *Características técnicas*

Es por ello que para poder cumplir estos objetivos finalmente se ha optado por el diseño de una plataforma de desarrollo propia en forma de *shield* que cumple con las siguientes características:

- La placa de circuito impreso diseñada tiene unas dimensiones de 120 mm x 80 mm (compatible con placas emulsionadas de baquelita y/o fibra de vidrio comerciales de fácil adquisición y bajo coste).
- El diseño de la PCB ha sido realizado utilizando el software libre de diseño de circuitos impresos KiCAD [28] de forma que los estudiantes pueden utilizarlo para ver los diseños, fabricarlos, modificarlos, etc…sin coste alguno.
- El ruteado de la PCB ha sido realizado íntegramente mediante una sola cara (para facilitar el proceso de fabricación en el laboratorio y para reducir los costes de fabricación en caso de utilizar la opción del proveedor externo).
- Todos los componentes electrónicos utilizados son *through-hole* para facilitar el proceso de montaje y soldadura de los componentes por parte de los estudiantes.

La Fig. 8 muestra la placa de desarrollo diseñada y fabricada presentada en este trabajo. Como puede observarse, la placa de desarrollo se comporta mecánicamente como un *shield* de Arduino, es decir, se conecta a los conectores hembra de la placa Icezum Alhambra II de manera que la placa de desarrollo queda conectada como un *shield* encima de la placa Icezum Alhambra II. Contiene 4 pulsadores tipo push-button, dos interruptores tipo DIP deslizantes de 4 canales, un

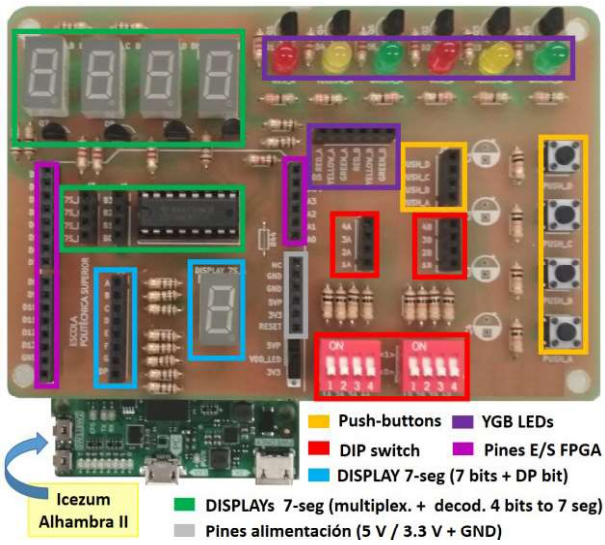

Fig. 8. Plataforma de desarrollo de bajo coste para la implementación de circuitos digitales en FPGAs mediante herramientas de hardware y software libres.

display de 7 segmentos controlado por 8 bits independientes (7 para los segmentos y 1 para el punto), 6 LEDs de colores (2 rojos, 2 verdes y 2 amarillos), 4 displays de 7 segmentos multiplexados, 20 pines accesibles de E/S de la FPGA y pines de alimentación (5V, 3.3 V y GND).

B. *Ejemplo de funcionamiento*

A continuación, se muestra un ejemplo del tipo de ejercicios (en este caso de complejidad media) que se pueden implementar mediante la plataforma de desarrollo presentada. La Fig. 9 muestra el esquema electrónico del sistema de multiplexación de los 4 displays de 7 segmentos de la placa de desarrollo. Como se puede observar, se trata de un bus de datos de 4 bits decodificados a 7 segmentos mediante el circuito integrado CD4511B y multiplexados mediante 4 transistores controlador mediante 4 bits de control. Imaginemos que se plantea un reto a los estudiantes y se les pide que diseñen un sistema digital que permita visualizar un número de 4 dígitos a través del sistema de los 4 displays de 7 segmentos donde el usuario deba introducir dos de los dígitos (unidades y decenas) codificados en binario mediante los dos interruptores DIP deslizantes de 4 canales mientras que las centenas y unidades de millar se encuentran prefijadas internamente en la FPGA. En la Fig. 10 se detalla una posible

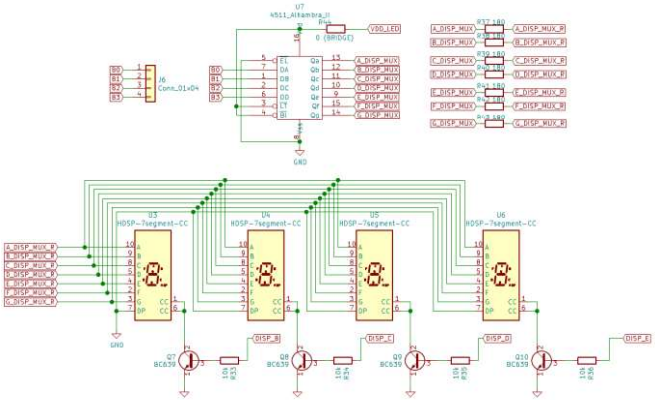

Fig. 9. Esquema electrónico del sistema de multiplexación de los 4 displays 7-segmentos de la plataforma de desarrollo de bajo coste.

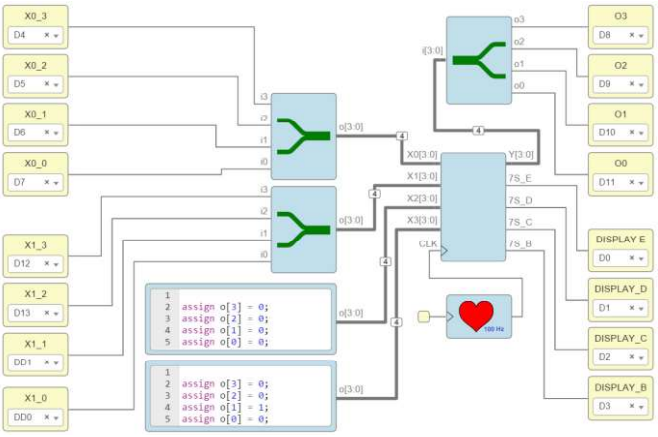

Fig. 10. Posible solución al reto planteado implementado mediante Icestudio donde se puede observar el bloque principal y asignación de los pines de E/S de la FPGA.

solución a este reto basada en la creación de un bloque principal con cuatro entradas de 4 bits (X3[3:0], X2[3:0] , X1[3:0] y X0[3:0]) que representan a cada uno de los 4 dígitos, 4 bits de salida (7S_B, 7S_C, 7S_D y 7S_E) sincronizados mediante una señal de reloj CLK que activan de forma cíclica uno de los 4 displays (dejando el resto inactivos) y una salida de 4 bits (Y[3:0]) sincronizada con la señal de reloj CLK y con los bits de selección que envía de forma cíclica al decodificador CD4511B cada uno de los 4 dígitos de las entradas. En este ejemplo, las centenas y unidades de millar se han prefijado a un valor de "0" y "2" respectivamente mediante dos pequeños bloques con código Verilog embebido. Las Fig. 11 y Fig. 12 muestran en detalle el contenido del bloque principal. Como puede observarse, éste está formado por cuatro multiplexores 4-1, un bloque que genera las señales de selección de los multiplexores Z0 y Z1 mediante un divisor de frecuencia de la señal de reloj CLK y un pequeño bloque formado por puertas lógicas que, en función del valor de Z0 y Z1, genera los 4 bits de activación de los displays: 7S_B, 7S_C, 7S_D y 7S_E. Nótese que para implementar correctamente el sistema diseñado en la placa de entrenamiento es necesario indicar en el editor de esquemas Icestudio qué entradas y salidas de la placa Icezum Alhambra II van a utilizarse (éstas están conectadas internamente a E/S de la FPGA) y conectar los periféricos de la placa de entrenamiento a las entradas y salidas correspondientes indicadas en la serigrafía de la placa tal y como se muestra en la Fig. 13 donde puede observarse la implementación cableada del reto propuesto y el correcto funcionamiento del sistema.

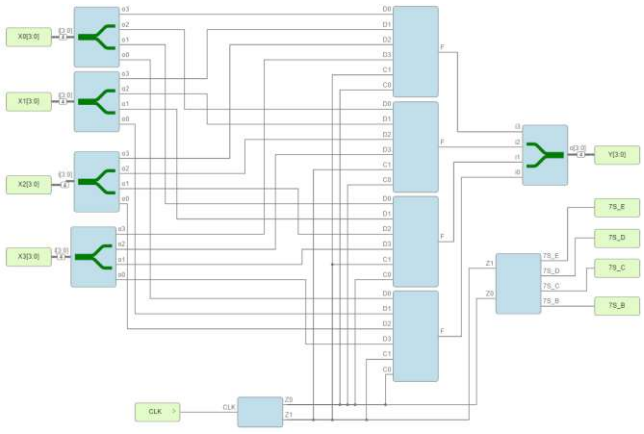

Fig. 11. Componentes internos del bloque principal.

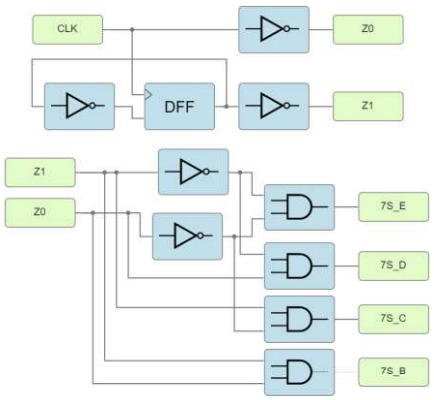

Fig. 12. Esquemas lógicos del circuito generador de señales de selección de los multiplexores (Z0 y Z1) y del circuito de activación de los displays.

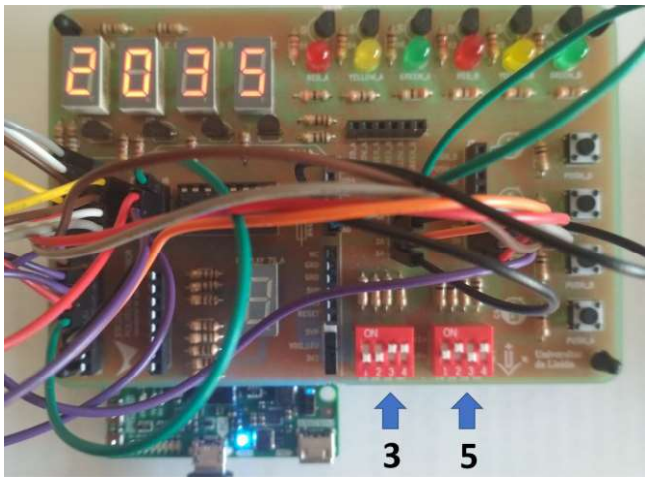

Fig. 13. Implementación práctica del reto planteado mediante la plataforma de desarrollo de bajo coste propuesta.

Nótese que, en el caso del sistema propuesto, únicamente se necesitan 16 conexiones cableadas para implementar este ejercicio, mientras que si se hubiera utilizado el sistema "clásico" se hubieran necesitado aproximadamente unas 50 conexiones cableadas en la *protoboard* y el uso de 6 circuitos integrados monolíticos: SN74HC153 (2x), SN74HC08 (1x), SN74HC04 (1x) y SN74HC74 (1x).

V. Resultados preliminares y Análisis de la experiencia

Para tener unos primeros indicios sobre el impacto en el de nuestra propuesta se ha realizado con nuestros estudiantes una práctica de laboratorio mediante el sistema "clásico" y luego se ha realizado otra sesión de laboratorio donde se ha implementado la misma práctica mediante el sistema que se propone en este trabajo: placa de desarrollo + placa Icezum Alhambra II + Icestudio. Finalmente se ha realizado una pequeña encuesta para conocer la opinión y la percepción de los estudiantes sobre el nuevo sistema de aprendizaje.

El ejercicio práctico ha consistido en el diseño y la implementación de un pequeño sistema digital combinacional que en función del valor de dos bits de entrada genera una salida determinada a través de un display 7-segmentos: 00 → "u", 10 → "d" y 01 → "l". La Fig. 14 muestra el esquema lógico del ejercicio en el editor visual Icestudio. La Fig. 15 muestra la implementación práctica del ejercicio mediante el uso de circuitos integrados monolíticos, cables de conexión y

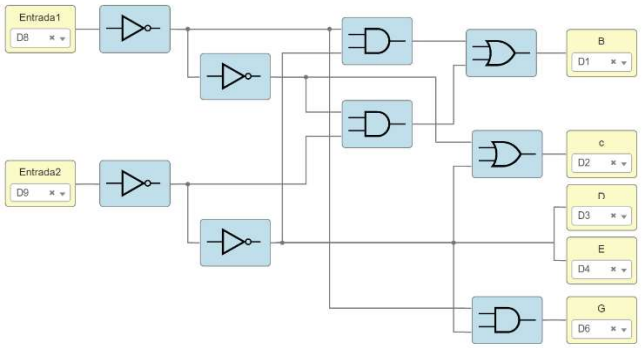

Fig. 14. Diagrama lógico del ejercicio práctico de *test* para nuestros estudiantes.

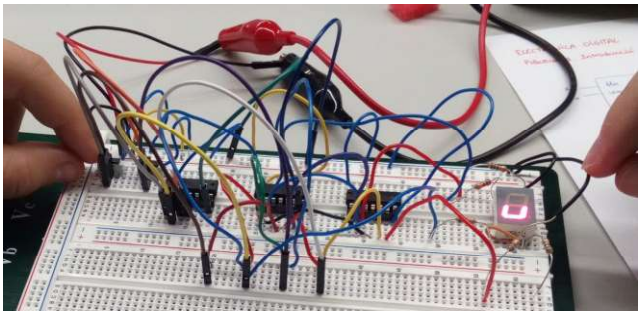

Fig. 15. Implementación ejercicio práctico mediante el sistema "clásico".

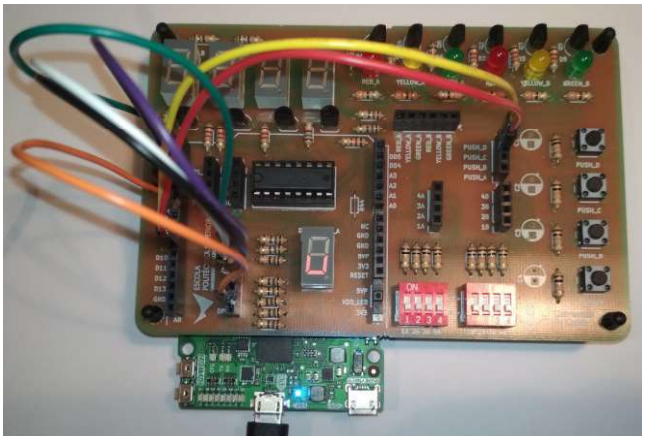

Fig. 16. Implementación del ejercicio práctico mediante la plataforma de desarrollo de bajo coste propuesta.

placas de prototipado. Obsérvese que en este caso el número de circuitos integrados comerciales necesarios para implementar el ejercicio es igual a 3 ya que el ejercicio práctico es bastante simple. Finalmente, la Fig. 16 muestra la implementación práctica mediante el sistema propuesto en este trabajo. Como se puede observar, el cableado del circuito de la Fig. 16 es mucho menor en el de la Fig. 15. Este hecho se hace más evidente a medida que la complejidad del circuito a implementar se incrementa y el número de circuitos integrados comerciales necesarios para implementar el ejercicio mediante aumenta.

Finalmente, la Fig. 17 muestra el resultado de las preguntas que se han realizado a una muestra de 30 estudiantes una vez finalizadas ambas sesiones de laboratorio y que se detallan a continuación: Q1) ¿Cuál de los dos sistemas crees que te permite una mejor implementación de los ejercicios prácticos de laboratorio de la asignatura de electrónica digital? y Q2) ¿Cuál de los dos sistemas crees que permite aprender mejor los conceptos asociados a la teoría y/o problemas expli-

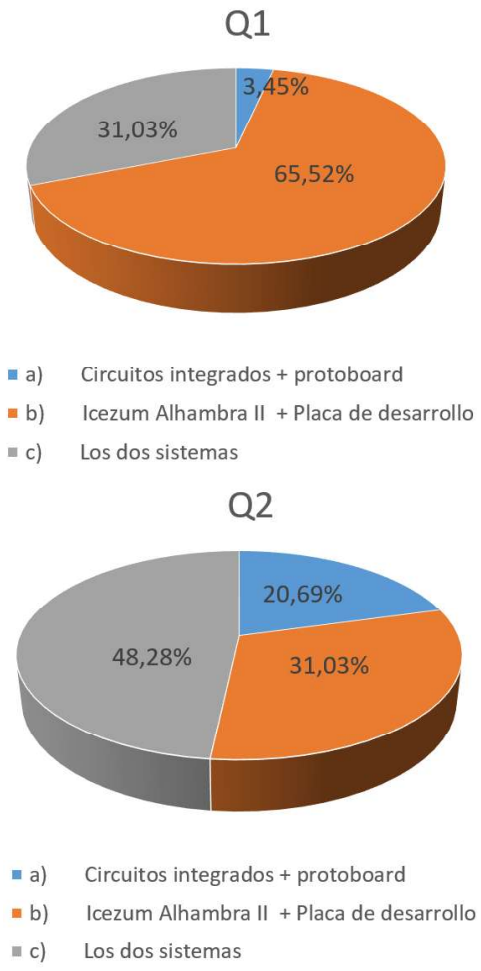

Fig. 17. Resultados de las preguntas Q1 y Q2 (muestra: 30 estudiantes).

cados en clase? Como puede observarse en la Fig. 17, en relación a la pregunta Q1 nuestros estudiantes creen / perciben que la implementación práctica de los ejercicios de prácticas de laboratorio de la asignatura de electrónica digital es mejor con el sistema propuesto (65,52 %) que con el sistema "clásico" (31,03%). Sin embargo, en relación a la pregunta Q2, una mayoría de estudiantes (48,28%) creen que ambos sistemas les permiten aprender de igual manera los conceptos asociados a la teoría y/o problemas explicados en clase frente un 20,69% de los estudiantes que prefiere el sistema clásico y un 31,03% de los estudiantes que prefiere el sistema propuesto. Este último resultado no ha dejado de sorprendernos ya que esperábamos que los estudiantes escogerían mayoritariamente el sistema propuesto en detrimento del sistema "clásico" (especialmente debido a las quejas que recibíamos recurrentemente sobre este último y que han sido comentadas en el presente trabajo). Creemos que el hecho que el ejercicio de testeo a la hora de comparar ambos sistemas sea tan simple ha influido en la respuesta proporcionada por los estudiantes y que si hubiéramos realizado la prueba con un ejercicio más complejo (número de circuitos integrados necesarios mayor que 5 o 6) la respuesta hubiera sido diferente. No obstante, este el resultado hace que valoremos la posibilidad de mantener (al menos de forma temporal) ambos sistemas a la hora de implementar los ejercicios prácticos de laboratorio mientras realizamos un estudio más profundo para obtener más datos objetivos y poder tomar una decisión definitiva ya que es posible que los estudiantes hayan valorado positivamente (y en contra de

nuestra intuición) el hecho de tener que enfrentarse a las dificultades del sistema "clásico" o el hecho de poder conocer, usar y trabajar (y disfrutar) con unos dispositivos, que aun siendo ya muy maduros, como son los circuitos de las series TTL 7400 o CMOS 4000, han sido fundamentales en el desarrollo y avance de la electrónica digital.

VI. Conclusiones

Si bien este trabajo no presenta estrictamente una nueva metodología docente en el ámbito de la enseñanza de la electrónica digital mediante el uso de FPGAs (el uso de plataformas de desarrollo comerciales está ampliamente publicado desde hace más de 15 años y está implementado en muchísimas universidades) creemos que su novedad radica 1) en la creación del concepto de plataforma de desarrollo de bajo coste que permite que los alumnos puedan trabajar cómodamente en sus casas y con total disponibilidad horaria y 2) en el uso en el aula de herramientas de hardware y software libre para programar las FPGAs. Los resultados preliminares obtenidos a partir de una encuesta realizada a nuestros estudiantes demuestran que éstos perciben que se mejora el proceso de implementación de los ejercicios prácticos del laboratorio de la asignatura de electrónica digital. Sin embargo, hay que seguir haciendo un estudio más profundo y sostenido en el tiempo (uno o varios cursos académicos) en la que los estudiantes utilicen la plataforma de bajo coste en sus casas de forma continuada para poder afirmar (o no) que la plataforma de desarrollo propuesta y la apuesta por el uso de herramientas de software /hardware libre en la implementación de circuitos digitales en FPGAs mediante herramientas libres mejora el proceso de aprendizaje de nuestros estudiantes.

Aunque la concepción de la plataforma de desarrollo de bajo coste se ha creado en base al uso de herramientas libres de reciente aparición, ya que nos ha parecido interesante explorar esta nueva opción tecnológica en el aula, su uso no es exclusivo en entornos de hardware y software libre. El mismo concepto podría haberse aplicado perfectamente (previa adaptación del diseño) al uso de placas de entrenamiento de marcas comerciales que suministran software de programación privativo a coste cero (licencias de estudiantes, licencias gratuitas para algunos modelos concretos de FPGA, período de prueba-evaluación gratuitos, etc…). En este sentido, este trabajo no pretende hacer apología del uso de las herramientas libres en detrimento de las herramientas privativas. Probablemente haría falta un estudio comparativo mucho más profundo para destacar objetivamente las bondades y fortalezas de un sistema u otro. No obstante, ciertamente no hemos sido ajenos a la influencia e impacto que está teniendo el movimiento *maker* en el ámbito tecnológico, industrial y académico actualmente a la hora de decidir el enfoque de este trabajo. Solo hace falta ver la contribución que ha tenido todo este movimiento en la pandemia de la COVID-19 (fabricación de viseras de protección y adaptación de máscaras mediante impresión 3D para su uso hospitalario o diseño de respiradores de emergencia, etc…) para darse cuenta que este movimiento ha llegado para quedarse y la Universidad no puede ser ajena a los cambios disruptivos que de vez en cuando aparecen en nuestra sociedad.

Agradecimientos

Este trabajo se ha realizado en el marco de la convocatoria de ayudas para la realización de proyectos de innovación y mejora de la docencia de la Universitat de Lleida (UdL) del curso 2018-19 y del proyecto: "Migració de les pràctiques de laboratori d'electrònica digital a una plataforma de desenvolupament de sistemes digitals de baix cost basada en l'ús de FPGAs lliures".

References

[1] U. Kretzschmar, J. Gomez-Cornejo, N. Moreira, U. Bidarte and A. Astarloa, "A versatile FPGA demonstration platform for academic use," *2014 XI Tecnologias Aplicadas a la Ensenanza de la Electronica (Technologies Applied to Electronics Teaching) (TAEE)*, Bilbao, 2014, pp. 1-6.

[2] E. Magdaleno, M. Rodríguez, D. Hernández, E. Rodríguez and F. Pérez, "Teaching methodology of the subject design of electronic systems using FPGA in the new master of industrial engineering," *2016 Technologies Applied to Electronics Teaching (TAEE)*, Seville, 2016, pp. 1-6.

[3] E. Todorovich, J. A. Marone and M. Vazquez, "Introducing Programmable Logic to Undergraduate Engineering Students in a Digital Electronics Course," in IEEE Transactions on Education, vol. 55, no. 2, pp. 255-262, May 2012.

[4] V. Kiray, S. Demir and M. Zhaparov, "Improving Digital Electronics Education with FPGA technology, PBL and Micro Learning methods, " in *Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*, Bali, 2013, pp. 445-448.

[5] C. Quintans, M. D. Valdes, M. J. Moure, L. Fernandez-Ferreira and E. Mandado, "Digital electronics learning system based on FPGA applications," *Proceedings Frontiers in Education 35th Annual Conference*, Indianopolis, IN, 2005, pp. S2G-7.

[6] H. N. Quang and T. M. Hoang, "A low-cost remote laboratory of field programmable gate arrays, " in *Proceedings of 2015 12th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Bangkok, 2015, pp. 172-176.

[7] G. Donzellini and D. Ponta, "From gates to FPGA: Learning digital design with Deeds," *2013 3rd Interdisciplinary Engineering Design Education Conference*, Santa Clara, CA, 2013, pp. 41-48.

[8] S. M. Trimberger, "Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology," in *Proceedings of the IEEE*, vol. 103, no. 3, pp. 318-331, March 2015.

[9] Terasic FPGA development boards. [Online]. Available: https://www.terasic.com.tw/ [Accessed: 15- March- 2020].

[10] Basys-3 FPGA board documentation. [Online]. Available: https://reference.digilentinc.com/reference/programmable-logic/basys-3/start [Accessed: 15- March- 2020].

[11] C.Wolf, "Project IceStorm". [Online]. Available: http://www.clifford.at/icestorm/ [Accessed: 15- March- 2020].

[12] D. Kushner, "The Making of Arduino: How five friends engineered a small circuit board that's taking the DIY world by storm" [Online]. Available: https://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino [Accessed: 27- April- 2020].

[13] iCE40-compatible free FPGA development boards. [Online]. Available: https://joelw.id.au/FPGA/CheapFPGADevelopmentBoards [Accessed: 15- March- 2020].

[14] R. Heradio *et al*., "Open-Source Hardware in Education: A Systematic Mapping Study," in *IEEE Access*, vol. 6, pp. 72094-72103, 2018.

[15] M. C. Rodriguez-Sanchez, A. Torrado-Carvajal, J. Vaquero, S. Borromeo and J. A. Hernandez-Tamames, "An Embedded Systems Course for Engineering Students Using Open-Source Platforms in Wireless Scenarios," in *IEEE Transactions on Education*, vol. 59, no. 4, pp. 248-254, Nov. 2016.

[16] J.M. Machado Cano, E. Ros Vidal, M. Rodríguez-Álvarez, "Migración de prácticas a una plataforma de hardware reconfigurable en la asignatura de Tecnología y Organización de Computadores," in *Proceedings of Jornadas SARTECO,* Málaga , 2017, pp. 705-711.

[17] J. Hormigo and A. Rodríguez, "Designing a Project for Learning Industry 4.0 by Applying IoT to Urban Garden," in *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 14, no. 2, pp. 58-65, May 2019.

[18] J. C. Martínez-Santos, O. Acevedo-Patino and S. H. Contreras-Ortiz, "Influence of Arduino on the Development of Advanced

Microcontrollers Courses," in *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 12, no. 4, pp. 208-217, Nov. 2017.

[19] I. Papaefstathiou, "IoT design course using open-source tools," *2016 IEEE Global Engineering Education Conference (EDUCON)*, Abu Dhabi, 2016, pp. 114-118.

[20] A. Fernández-Pacheco, S. Martin and M. Castro, "Implementation of an Arduino Remote Laboratory with Raspberry Pi, " *2019 IEEE Global Engineering Education Conference (EDUCON)*, Dubai, United Arab Emirates, 2019, pp. 1415-1418.

[21] Arduino: Open-source electronic prototyping platform enabling users to create interactive electronic objects. [Online] Available: https://www.arduino.cc/ [Accessed: 15- March- 2020].

[22] ghdl-yosys-plugin: VHDL synthesis with ghdl and yosys. [Online] Available: https://github.com/ghdl/ghdl-yosys-plugin [Accessed: 27- April- 2020].

[23] Icezum Alhambra II board. [Online] Available: https://alhambrabits.com/alhambra/ [Accessed: 15- March- 2020].

[24] Icestudio: Visual editor for open FPGA boards. [Online]. Available: https://Icestudio.io/ [Accessed: 15- March- 2020].

[25] FPGAWars: Exploring the dark side of free FPGAs. [Online]. Available: http://fpgawars.github.io/ [Accessed: 15- March- 2020].

[26] Apio, an open source ecosystem for open FPGA boards. [Online]. Available: https://apiodoc.readthedocs.io/en/stable/ [Accessed: 15- March- 2020].

[27] Experimental open FPGA IDE using Atom and Apio. [Online] Available: https://github.com/FPGAwars/apio-ide [Accessed: 15- March- 2020].

[28] KiCAD EDA: A Cross Platform and Open Source Electronics Design Automation Suite. Online] Available: https://www.kicad-pcb.org/ [Accessed: 15- March- 2020].