Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

Utilizando Arduino Due en la docencia de la entrada/salida

Sergio Barrachina Mir, Germán Fabregat Llueca, José Vicente Martí Avilés
Dpto. de Ingeniería y Ciencia de los Computadores
Universitat Jaume I
Castellón
{barrachi,fabregat,vmarti}@uji.es

Resumen

La problemática de la entrada/salida y su gestión suele formar parte de las asignaturas de introducción a la arquitectura de computadores. La propia naturaleza del tema y su diversidad hace que las sesiones prácticas se lleven a cabo habitualmente, bien sobre dispositivos específicos sencillos, bien sobre simuladores, lo que las aleja de los dispositivos reales y les resta vistosidad.
Sin embargo, es posible utilizar dispositivos actuales y sencillos, como las tarjetas Arduino, para presentar a los estudiantes una visión más real y atractiva de la entrada/salida, manteniendo a su vez la sencillez de uso de los entornos y sistemas empleados, lo que consideramos prioritario en los primeros cursos de grado.
En nuestro caso, puesto que actualmente fundamentamos nuestra docencia en arquitectura de computadores sobre la arquitectura ARM, hemos optado por el modelo Arduino Due, que dispone de un microcontrolador, el ATSAM3X8E, que implementa la versión Cortex-M3 de la arquitectura ARM.
Para poder realizar las prácticas de entrada/salida hemos modificado ligeramente el entorno Arduino para poder incluir programas en ensamblador, y hemos diseñado una pequeña tarjeta con un led RGB y un pulsador, lo que ha permitido proponer ejercicios sencillos pero vistosos. Los propios dispositivos del microcontrolador de la Arduino DUE han bastado para abarcar otros aspectos de la entrada/salida y presentar ejemplos de mayor complejidad para incentivar a los estudiantes.
La primera experiencia con este entorno ha sido satisfactoria tanto para el profesorado de las asignaturas en las que se ha utilizado como para los estudiantes, en quienes además se ha fomentado el interés en continuar trabajando con las tarjetas Arduino en sus propios proyectos.

Abstract

The input/output (I/O) and its management is often part of the introductory courses to computer architecture. The very nature of this topic and its diversity makes that the practice sessions often take place either on simple specific devices, or on simulators, which hide the complexity of actual I/O devices and subtracts their appealing.
However, it is possible to use today existing and simple devices such as Arduino boards to introduce students to a more realistic and attractive vision of the I/O, while maintaining the ease of use of the required environments and systems, which we consider a priority on first degree courses.
In our case, since currently we base our teaching on computer architecture on the ARM architecture, we have opted for the Arduino Due model, which has a microcontroller, ATSAM3X8E, which implements the Cortex-M3 version of the ARM Architecture.
To carry out the laboratory sessions on I/O we have slightly modified the Arduino IDE in order to accept assembly source code. In addition, we have designed and built a small board with an RGB led and a switch, which allowed us to propose simple but colourful exercises. The built-in I/O included in the ARM controller of the Arduino DUE board have proved enough to explore other important aspects of I/O as well as to offer more complex examples to incentivate the students on the subject.
The first experience with this environment has been satisfactory for both teachers and students, who also have fostered interest in continuing to work with Arduino cards in their own projects.

Palabras clave

Arquitectura de Computadores, Entrada/Salida, Arduino Due, ARM, Thumb, Ensamblador.

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

1. Motivación

Suele ser habitual que los cursos de arquitectura de computadores cambien periódicamente sus contenidos para así poder seguir el paso a la espectacular evolución de la arquitectura de los computadores. Por otro lado, la mayoría de los cursos de arquitectura de computadores recurren normalmente a un hardware determinado para explicar los conceptos relacionados con la materia [7].

En 2013 tomamos la decisión de modificar la arquitectura de referencia de las asignaturas relacionadas con la materia de Arquitectura de Computadores. Decidimos cambiar de la arquitectura MIPS a la ARM.

La arquitectura ARM presenta muchas características que la distinguen de otras arquitecturas contemporáneas y, por otro lado, al estar basada en RISC, es relativamente sencilla [5]. Además, el hecho de que ARM sea una arquitectura actual y ampliamente difundida, especialmente en dispositivos móviles, *smartphones* y *tablets*, es un factor especialmente motivador para los estudiantes [6].

Una vez tomada la decisión de cambiar a la arquitectura ARM, se inició un proceso de redefinición de las guías docentes y de los materiales utilizados en la enseñanza tanto teórica como práctica de las distintas asignaturas relacionadas con la materia de Arquitectura de Computadores.

En concreto, en la asignatura *Estructura de Computadores*, de primer curso, primer semestre, uno de los objetivos fue el de redefinir la docencia tanto teórica como práctica de la parte correspondiente a la gestión de la entrada/salida.

Hasta ese momento, en las prácticas de dicha asignatura [2] se utilizaba el simulador `QtSpim`[1], antes llamado `xspim` en su versión para GNU/Linux. El simulador `xspim` nos parecía un entorno adecuado para las prácticas de entrada/salida de la asignatura debido a que dicho simulador incorpora un pequeño programa monitor (cuyo código fuente está disponible y puede ser adaptado) y simula un conjunto de dispositivos de entrada/salida: teclado, pantalla y reloj, que pueden sincronizarse por consulta de estado o por interrupciones.

Puesto que era necesario adaptar las prácticas de entrada/salida a la arquitectura ARM, nos planteamos si queríamos continuar con una experiencia docente basada en el uso de un simulador. En ese caso, el estudiante debería programar en ensamblador el código necesario para interactuar con los dispositivos de entrada/salida proporcionados por el simulador que fuera a utilizarse. O si por el contrario, apostábamos por un enfoque cercano a lo que se ha dado en llamar *computación física* [10]. En este caso, el estudiante interactuaría con dispositivos físicos, debería ser consciente de qué es lo que quiere que pase en la realidad y podría observar cómo su aplicación es capaz, o no, de reaccionar adecuadamente ante eventos externos. Siguiendo dicho enfoque, el estudiante podría relacionar fácilmente la secuencia de acciones a las que un dispositivo tiene que dar respuesta, con la programación que haya realizado de dicho dispositivo.

Creemos que esta segunda opción es mucho más enriquecedora para el estudiante que simplemente limitarse a observar en la pantalla de un simulador si su código de gestión de la entrada/salida se comporta como teóricamente debería. Es más, puesto que mucha de la problemática de la entrada/salida está directamente relacionada con la interacción hombre-máquina, esta segunda opción ofrece al estudiante la oportunidad de enfrentarse a un escenario real, en el que si no se toman las debidas precauciones, no todo funcionará como se espera.

Así pues, una vez optamos por la interacción con dispositivos físicos, quedaba por decidir qué componente hardware utilizar en el laboratorio. Puesto que nuestras restricciones era que dicho componente estuviera basado en la arquitectura ARM y que fuera de bajo coste, para que el estudiante pudiera adquirirlo en el caso de querer experimentar por su cuenta, la tarjeta Arduino Due, basada en ARM y con un precio económico, nos dio la impresión de ser una buena elección.

Otra posible alternativa, de la que hay experiencias que prueban que es una buena opción, hubiera sido la de utilizar la vídeo-consola Nintendo DS [9]. Sin embargo, la mayor complejidad de este dispositivo y de su entorno, y el hecho de que una tarjeta desnuda como la Arduino DUE dan al estudiante una visión más evidente del hardware, nos reforzaron en nuestra decisión.

La tarjeta Arduino[2], además de surgir del mundo académico, ha sido utilizada ampliamente y con éxito en la docencia universitaria. Por ejemplo, en [8] se describe cómo se ha empleado dicha tarjeta para la docencia de sistemas empotrados, con un enfoque basado en proyectos. De igual forma, en [11] y [12] se describen las ventajas que proporciona dicho entorno, en uno de los casos en combinación con los robots LEGO, para la docencia de programación de computadores a estudiantes que no son de ingeniería informática y que se enfrentan por primera vez a la programación.

En concreto, el modelo Due de Arduino[3] presenta las siguientes características que consideramos interesantes desde el punto de vista de la docencia de la entrada/salida utilizando ARM como arquitectura de referencia:

- Microcontrolador Atmel SAM3X8E con una CPU ARM Cortex-M3.

[1] `http://spimsimulator.sourceforge.net/`

[2] `http://arduino.cc/`

[3] `http://arduino.cc/en/Main/arduinoBoardDue`

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

- 54 pines de entrada/salida digitales (de los cuales 12 pueden utilizarse como salidas PWM).
- 12 entradas analógicas.
- 4 UARTs (puertos serie hardware).
- Conexión USB con capacidad *on-the-go* (OTG).
- 2 conversores digital a analógico.
- Reloj (a 84 Mhz).
- Conectores SPI y JTAG.

Todo lo anterior por menos de 45 €.

Por si fuera poco, debido a la gran acogida que ha tenido Arduino por parte de entusiastas del *hágalo usted mismo* [13], el ecosistema a su alrededor es simplemente impresionante: existen multitud de proyectos libremente accesibles, una gran cantidad de extensiones *hardware* (*shields*) que se acoplan directamente sobre la tarjeta Arduino, y una ingente cantidad de *software* libre y de documentación.

En cuanto al entorno de desarrollo Arduino, éste proporciona un conjunto de bibliotecas y funciones destinadas al manejo de la entrada/salida y a la comunicación de la tarjeta Arduino con el computador a través del puerto USB. Por no mencionar las bibliotecas proporcionadas por los fabricantes de los distintos *shields*, que se pueden incorporar fácilmente al entorno para poder utilizarlos de una forma sencilla.

Por otro lado, la programación de la tarjeta Arduino se realiza habitualmente en C/C++. Es más, el entorno de desarrollo Arduino solo soporta dichos lenguajes. Teniendo en cuenta la gran variedad de modelos Arduino, esta decisión entendemos que está motivada por la intención de que si alguien desarrolla un código en C/C++ para una determinada tarjeta Arduino, éste podrá utilizarse en otros modelos de tarjetas Arduino sin ningún problema. No ocurriría lo mismo si se permitiera programar en lenguaje ensamblador, ya que dicho programa estaría ligado a la arquitectura concreta de un determinado modelo de tarjeta Arduino, perdiéndose la compatibilidad software entre tarjetas.

Sin embargo, para el caso que nos ocupa, la gestión de la entrada/salida a bajo nivel, las funciones en C/C++ proporcionadas por el entorno Arduino abstraen una gran parte de la problemática relacionada con la gestión de la entrada/salida desde el punto de vista de un arquitecto de computadores. Así que aparentemente no podíamos contar con el entorno de desarrollo de Arduino para programar directamente la tarjeta Arduino Due, a no ser que decidiéramos ocultar lo que realmente ocurre a nivel del lenguaje máquina.

Afortunadamente, y puesto que el entorno de desarrollo de Arduino es software libre, fuimos capaces de introducir las modificaciones necesarias para que dicho entorno aceptara directamente código en ensamblador.

En vista de todo lo anterior, decidimos adquirir unas cuantas tarjetas Arduino Due y evaluar cómo podríamos utilizarlas para la docencia de la gestión de la entrada/salida. A partir de dicho estudio, desarrollamos el recurso docente que presentamos en este artículo y que ha sido utilizado más que satisfactoriamente durante el curso 2014/15 en los Grados en Ingeniería Informática y en Matemática Computacional de la Universitat Jaume I.

El resto del artículo está organizado como sigue. En el Apartado 2 se muestra el contexto docente en el que se ha desarrollado el recurso docente propuesto. En el Apartado 3 se describe el material del que consta este recurso docente. El Apartado 4 muestra los resultados obtenidos. Finalmente, en el apartado 5 se describen las conclusiones y trabajo futuro.

2. Contexto docente

La asignatura Estructura de Computadores es de formación básica y se imparte en primer curso, primer semestre en los Grados en Ingeniería Informática y en Matemática Computacional de la Universitat Jaume I.

Al ser una asignatura de formación básica, tiene asignada parte de la siguiente competencia fijada por Resolución de 8 de junio de 2009, de la Secretaría General de Universidades[4]: «Conocimiento de la estructura, organización, funcionamiento e interconexión de los sistemas informáticos, los fundamentos de su programación, y su aplicación para la resolución de problemas propios de la ingeniería».

Con el objetivo de dar respuesta a dicha competencia, la asignatura se ha estructurado en tres temas, el tercero de los cuales abarca los sistemas de entrada/-salida. Para dicho tema se han definido los objetivos formativos que se detallan a continuación.

El estudiante, al finalizar la asignatura, deberá ser capaz de:

1. Describir la problemática general de la entrada/-salida.
2. Definir los términos de latencia, tasa de transferencia y productividad y ser capaz de estimar o calcular sus valores partiendo de datos suficientes.
3. Caracterizar los sistemas de entrada/salida en función de su comportamiento, interlocutor, tasa de transferencia y latencia.
4. Describir la estructura general de los dispositivos de entrada/salida.
5. Enumerar y describir las formas por medio de las cuales el procesador puede acceder a los dispositivos de entrada/salida.
6. Enumerar los tipos de registros de los dispositivos de entrada/salida y explicar cómo y para qué se utilizan.

[4]Publicada en el BOE el 4 de agosto de 2009

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

7. Explicar y evaluar el funcionamiento, así como las ventajas y desventajas, de los dos sistemas de sincronización de la entrada/salida: I) consulta de estado (*polling*) e II) interrupciones.
8. Explicar y evaluar el funcionamiento, así como las ventajas y desventajas, de los dos sistemas de transferencia de datos de entrada/salida: I) transferencias por programa realizadas por el procesador y II) acceso directo a memoria (DMA).
9. Describir las características generales de la entrada/salida de propósito general (GPIO) y de los mecanismos de temporización de los ordenadores.
10. Analizar programas escritos en ensamblador Thumb 2 que gestionen dispositivos de entrada/-salida.
11. Desarrollar programas en ensamblador Thumb 2 que gestionen dispositivos de entrada/salida.

Para la consecución de los anteriores objetivos formativos, se ha optado por dividir la materia objeto de estudio en los siguientes cinco bloques:

1. Introducción y problemática general de la entrada/salida.
2. Aspectos generales de la entrada/salida de propósito general (GPIO) y de la temporización.
3. Gestión de la entrada/salida por medio de consulta de estado.
4. Gestión de la entrada/salida por medio de interrupciones.
5. Transferencia de datos por procesador y por acceso directo a memoria.

En cuanto a la carga de trabajo del estudiante, se han asignado 1,8 créditos ECTS. La mayor parte del trabajo se realizará a lo largo de 5 semanas en los siguientes tipos de actividades:

- 5 sesiones de teoría de 2 horas cada una.
- 4 sesiones de laboratorio de 2 horas cada una.
- 18 horas de preparación entre sesiones.

La organización temporal de las sesiones de teoría y laboratorio se muestra en la Figura 1. Conviene destacar que se ha apostado por realizar las sesiones de teoría y laboratorio en el mismo día para evitar que la sesión de teoría y su laboratorio correspondiente se desincronicen debido a la presencia de festivos. Esta sincronización es importante ya que nos permite seguir una metodología docente en la que el estudiante prepara por su cuenta el contenido que se desarrollará en la siguiente sesión, tanto en la clase de teoría como en la de prácticas.

Además de las horas ya indicadas, el estudiante deberá dedicar 9 horas de estudio personal para exámenes —no necesariamente en esas semanas—.

En cuanto a la metodología docente, se ha seguido el siguiente planteamiento basado en la utilización

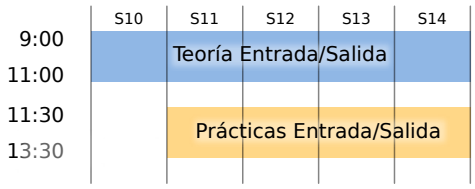

Figura 1: Organización temporal de las sesiones de teoría y laboratorio

de entregables y en la metodología de clase invertida (*flipped classroom*):

- En primer lugar, el estudiante debe preparar de forma individual una parte de cada una de las sesiones de teoría y laboratorio con antelación. Para ello, una semana antes se le proporciona un entregable en el que se indica qué documentación debe consultar, además de un conjunto de preguntas y ejercicios sencillos que debería ser capaz de resolver a partir de dicha documentación.
- La clase de teoría comienza con la resolución de forma conjunta de las dudas que hayan podido surgir durante la realización del entregable, así como la corrección de las preguntas y ejercicios planteados. Una vez hecho lo anterior, se profundiza sobre el tema en cuestión y se plantean nuevos problemas que deben ser resueltos en equipos de tres estudiantes.
- Por último, cada equipo realiza en el laboratorio la correspondiente práctica sobre el tema en cuestión.

La evaluación se realiza de forma continua y tiene en cuenta los entregables realizados, la nota obtenida en las sesiones de laboratorio, la nota obtenida en el examen de objetivos básicos correspondiente y la parte correspondiente a la entrada/salida en el examen teórico. Para la gestión y calificación de las actividades se ha utilizado la herramienta descrita en [1].

3. Entrada/salida con Arduino

El recurso docente presentado en este artículo pretende proporcionar material teórico y de laboratorio con el objetivo de facilitar la comprensión por parte del estudiante de la gestión de la entrada/salida desde el punto de vista de la arquitectura de computadores. Consta de:

- Los tres últimos capítulos de [3].
- Programas de ejemplo.
- El entorno de desarrollo de Arduino modificado para aceptar ficheros en ensamblador.
- Esquemático de un circuito sencillo con un pulsador y un diodo LED RGB.

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

- Solución del profesor.

Todo el material anterior, salvo la solución del profesor, que está disponible bajo demanda, puede descargarse desde la siguiente dirección web:
`http://lorca.act.uji.es/book/practARM/`

Conviene tener en cuenta que para poder realizar las prácticas propuestas, tan solo será necesario disponer, por cada equipo de estudiantes, del siguiente material:

- 1 Computador con GNU/Linux, MacOSX o Windows.
- 1 Tarjeta Arduino Due.
- 1 Tarjeta de entrada/salida basada en el esquemático proporcionado (ver Figura 2).

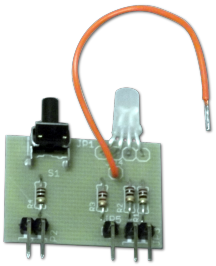

Figura 2: Tarjeta de entrada/salida

Los siguientes subapartados describen con un poco más de detalle los distintos materiales que forman este recurso docente.

3.1. Capítulos de entrada/salida

La parte del libro «Introducción a la arquitectura de computadores con QtARMSim y Arduino» [3] dedicada a la entrada/salida se ha organizado en tres capítulos.

El primero de ellos proporciona una introducción teórica a la entrada/salida. Comienza hablando de las generalidades y problemática asociada a la entrada/salida. Describe la estructura de los sistemas y dispositivos de entrada/salida. Continúa con la gestión de la entrada/salida por medio de consulta de estado y de interrupciones. Por último, describe las transferencias de datos y la DMA. Es un capítulo eminentemente teórico, que sirve como introducción teórica de cada uno de los temas comentados.

El segundo de los capítulos se centra en los dispositivos de entrada/salida. El primer apartado trata los dispositivos de entrada/salida relacionados con la entrada/salida de propósito general (GPIO). Primero desde un punto de vista genérico y después, particularizado a la GPIO del ATSAM3X8E (el microcontrolador incorporado en la tarjeta Arduino Due). El segundo apartado abarca la gestión del tiempo. Al igual que antes, en un primer lugar de forma genérica, para después particularizarlo a la gestión del tiempo del ATSAM3X8E. El tercer apartado cubre la gestión de excepciones e interrupciones en el ATSAM3X8E. El último apartado, el controlador de DMA del ATSAM3X8E.

El último de los tres capítulos está destinado al entorno de prácticas. Comienza describiendo la tarjeta Arduino Due, la tarjeta de entrada/salida, el entorno de desarrollo y la creación de proyectos en dicho entorno. A continuación, se proponen ejercicios para las cuatro sesiones de laboratorio que se detallan seguidamente.

Introducción a la entrada/salida. En esta primera sesión se muestra cómo encender o a apagar el LED de la tarjeta de entrada/salida; cómo escribir información por medio del puerto serie de la tarjeta Arduino; y cómo acceder a la información del reloj de tiempo real para obtener la fecha y hora actual de la tarjeta Arduino Due.

Entrada/salida por consulta de estado. En esta sesión se incide sobre la consulta de estado y su problemática. El estudiante deberá programar el entorno Arduino para detectar la pulsación del pulsador de la tarjeta de entrada/salida y actuar en consecuencia. También tendrá que programar una alarma en el reloj en tiempo real de la tarjeta Arduino y averiguar por consulta de estado si la alarma ha saltado o no.

Entrada/salida por interrupciones. En esta sesión el estudiante debe configurar la tarjeta Arduino para activar las interrupciones y probar las distintas configuraciones disponibles. Se propone utilizar el pulsador de la tarjeta de entrada/salida para comprobar cómo varía el tratamiento de las interrupciones en función de la configuración seleccionada. El estudiante también deberá programar el reloj en tiempo real para fijar una alarma de tal forma que cuando ésta salte, se genere una interrupción (y comparar el comportamiento de esta práctica con el observado en la práctica anterior, en la que se averiguaba si la alarma se había generado o no por medio de la consulta de estado).

Modulación por ancho de pulso (PWM), acceso directo a memoria (DMA) y puerto USB. En primer lugar, el estudiante verá cómo se puede disminuir la intensidad percibida del LED de la tarjeta de entrada/salida sin más que modificar la proporción de tiempo que éste permanece encendido en un ciclo de trabajo. En la segunda parte de la práctica, explorará la utilización del controlador de acceso directo a memoria (DMA) para la transferencia de bloques de memoria. En la última parte de la práctica, deberá poner en funcionamiento un ejemplo más elaborado capaz de programar la tarjeta Arduino Due de tal forma que ésta simule que es un ratón, primero, y un teclado, después.

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

3.2. Programas de ejemplo

Como ya se ha comentado anteriormente, el objetivo de las prácticas es el de realizar programas en ensamblador de ARM que sean capaces de configurar e interactuar con los dispositivos de entrada/salida presentes en la tarjeta Arduino Due.

No sería realista pretender que en cada sesión de dos horas de laboratorio los estudiantes fueran capaces de abordar desde cero todos los programas que se proponen en cada sesión. Por tanto, para la mayoría de los ejercicios propuestos se proporciona una primera versión que los estudiantes deberán ir modificando, completando o extendiendo en función de lo que deban realizar finalmente.

Dichos programas de ejemplo se proporcionan también como parte de este recurso docente. En ellos se han ocultado fragmentos de código y valores concretos de etiquetas, máscaras y direcciones de E/S para que los alumnos los completen, manteniendo la estructura del programa y los comentarios descriptivos de las acciones que debe realizar cada sección del código y cada instrucción para facilitar la comprensión por parte del alumno. La Figura 4 muestra un fragmento de código a modo de ejemplo.

3.3. Entorno de desarrollo modificado

El entorno de desarrollo de Arduino, que puede descargarse desde su página web, no soporta la realización de proyectos Arduino en otros lenguajes de programación que no sean C o C++. Afortunadamente, puesto que dicho entorno de desarrollo es software libre, hemos podido estudiar su código y modificarlo para que también aceptara y compilara programas en lenguaje ensamblador.

Utilizando esta versión modificada es posible desarrollar proyectos que tengan ficheros fuente tanto en C/C+, como en ensamblador de ARM. Esto nos ha permitido graduar la dificultad de las prácticas de tal forma que determinadas acciones se realizan inicialmente en C/C++ y posteriormente se muestra qué acciones se deben realizar a bajo nivel para conseguir el mismo efecto.

3.4. Circuito sencillo con un pulsador y un diodo LED RGB

Aunque existen multitud de extensiones *hardware* (*shields*) de entrada/salida para la tarjeta Arduino Due, nos planteamos cuál sería el circuito más simple que permitiera realizar prácticas de entrada/salida que fueran sencillas pero suficientemente vistosas.

Llegamos a la conclusión de que puesto que la tarjeta Arduino Due ya proporciona otros dispositivos de entrada/salida como un temporizador, un reloj de tiempo real y un controlador de DMA, era suficiente con un circuito simple como el mostrado en la placa de la Figura 2. Dicho circuito está formado por 4 resistencias, un pulsador y un diodo LED RGB. De esta forma, es posible provocar eventos en la tarjeta por medio del pulsador y visualizar su salida por medio del diodo LED RGB. Hay que tener en cuenta que, al tratarse de un diodo LED RGB, se dispone en realidad de tres salidas, cada una de ellas asociada a uno de los colores: rojo, verde y azul. Además, se ha diseñado una placa con dicho circuito de tal forma que pueda insertarse directamente en la tarjeta Arduino Due, tal y como se puede ver en la Figura 3.

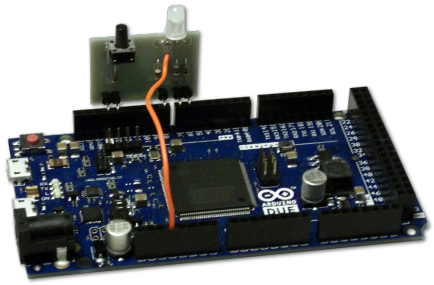

Figura 3: Tarjeta Arduino/Due con la tarjeta de entrada/salida conectada

Desde nuestro punto de vista, la utilización de un circuito extremadamente sencillo en lugar de un *shield* prefabricado presenta dos ventajas. Por un lado, el circuito propuesto tiene pocos componentes y un estudiante podría reproducirlo por su cuenta por muy poco dinero (ni siquiera es necesario que fabrique una placa impresa, podría utilizar directamente una placa de prototipos).

Por otro lado, al utilizar un circuito tan sencillo, el estudiante es consciente de que dichos elementos (pulsador y LED RGB) están directamente conectados a los pines de entrada/salida de la tarjeta Arduino. Así, cuando posteriormente programe la GPIO a bajo nivel, podrá ver que su programa en ensamblador estará configurando, consultando o activando justamente dichos pines de entrada/salida.

Un *shield* más complejo (por ejemplo una pantalla LCD) podría resultar más vistoso, pero además de ser más caro, una de dos, o se habría añadido una complejidad innecesaria en el caso de gestionar su entrada/salida a bajo nivel, o se hubiera ocultado el funcionamiento de la entrada/salida en el caso de haber utilizado las funciones en C/C++ proporcionadas por el correspondiente fabricante.

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

3.5. Otros dispositivos interesantes

La diversidad de dispositivos presentes en el microcontrolador ATSAM3X8E hace que sea necesario buscar un equilibrio adecuado entre qué es necesario para abarcar todos los objetivos de la asignatura y qué puede resultar útil y vistoso, motivando al estudiante, pero añade una complejidad excesiva para primer curso.

Nuestra elección se concretó en el uso, como dispositivos fundamentales, de la GPIO (entrada/salida de propósito general) y del reloj de tiempo real como elemento de temporización. Ambos permiten su gestión mediante prueba de estado o interrupciones, y son suficientemente sencillos y fáciles de comprender.

Para el segundo grupo, se utilizó el PWM (modulación de anchura de pulsos) y la interfaz de dispositivo USB. Mediante modificaciones sencillas del código que se les proporcionaba, los estudiantes pudieron ver un uso más avanzado de la entrada/salida, sin que hubiera una gran complejidad conceptual en los ejercicios.

4. Resultados

Una vez finalizado el curso, realizamos una encuesta con 3 preguntas de escala Likert de 1 a 5, siendo 1 «totalmente en desacuerdo» y 5 «totalmente de acuerdo», y 2 de respuesta abierta. Contestaron a dicha encuesta 28 estudiantes (un $19\,\%$ de los que siguieron la asignatura).

Las primeras dos preguntas, relacionadas con el objetivo docente del recurso y la satisfacción del estudiante, «Considero que las prácticas con Arduino me han servido para entender mejor cómo funciona y se programa la entrada/salida» y «Recomendaría que se siguiera utilizando Arduino en la docencia de arquitectura de computadores» obtuvieron una mediana de 4 y 5, respectivamente. La tercera pregunta, dirigida exclusivamente a repetidores, «Considero más interesantes las prácticas de E/S sobre Arduino de este curso que las que se realizaron el curso pasado con el simulador XSPIM», obtuvo una mediana de 5.

Las respuestas obtenidas en la penúltima pregunta, «¿Qué aspectos positivos destacarías de las prácticas con Arduino?», destacaron la motivación que supone un sistema real y en concreto el Arduino, así como la mayor facilidad en entender el funcionamiento de la entrada/salida sobre dicho sistema, frente al uso de un simulador.

Por último, las respuestas a la pregunta «¿Qué crees que convendría mejorar de las prácticas con Arduino?» incidieron en que les hubiera gustado que la asignatura dispusiera de más tiempo para poder profundizar más en la arquitectura de computadores y que se hubiera explicado con más detalle el IDE de Arduino, que les resultó un poco desconcertante al principio.

Al margen de la encuesta, merece la pena destacar el que estudiantes que ya habían superado la asignatura, cuando se han enterado por sus compañeros que habíamos utilizado Arduino en las prácticas, nos han pedido permiso para poder acudir como oyentes a las prácticas del próximo curso.

En cuanto al rendimiento académico, comparado con respecto al curso anterior, se ha aumentado el porcentaje de estudiantes que han seguido la asignatura del $83\,\%$ al $88\,\%$ y el número de aprobados del $49\,\%$ al $56\,\%$. Si bien es cierto que no es posible concluir que las mejoras en los resultados estén directamente relacionadas con la utilización de este recurso docente, ya que hay una gran variedad de factores que podrían haber motivado dichas mejoras.

Por último, la valoración de los profesores de laboratorio también ha sido muy positiva.

5. Conclusiones y trabajo futuro

Consideramos que el recurso docente presentado ha permitido completar en buena medida los objetivos que nos planteamos al decidir orientar la asignatura entorno a la arquitectura ARM. También creemos, basándonos en los resultados académicos del presente curso, así como en las respuestas dadas por los estudiantes tras el desarrollo de las prácticas, que el uso de la tarjeta Arduino Due ha sido una elección acertada.

Además, consideramos que se ha proporcionado a los estudiantes una base de conocimientos suficiente para que puedan continuar desarrollando sus propios proyectos con sistemas Arduino y, lo que consideramos más importante, creemos haber suscitado su interés por el desarrollo de estos proyectos que, en definitiva, les permitirá mejorar múltiples aspectos de su formación en arquitectura de computadores y en diseño de sistemas informáticos.

Para los próximos años se prevé mejorar y completar tanto la documentación de los aspectos teóricos como los ejercicios prácticos, con nuevos ejemplos que incidan sobre otros dispositivos integrados en el ATSAM3X8E, que se dejarán como trabajo de ampliación de conocimientos para los estudiantes.

Por otra parte, en el caso de que se ampliara el simulador descrito en [4] para soportar el conjunto de instrucciones Thumb II, que corresponde a la arquitectura presente en la tarjeta Arduino Due, sería deseable, en el futuro, enlazar ambos sistemas.

Referencias

[1] Sergio Barrachina Mir, Asunción Castaño Álvarez, Maribel Castillo Catalán, Germán León Na-

Andorra La Vella, del 8 al 10 de julio 2015
ISBN: 978-99920-70-10-9

```
# Espera activación pulsador y regresa
# Declaraciones de constantes
   .equ PIOB,     0x????????    @ Dirección base del PIO B
   .equ PIO_PER,  0x??          @ Offset Pio Enable Register
   .equ PIO_IDR,  0x??          @ Offset Interrupt Disable Register
   .equ PIO_PUER, 0x??          @ Offset Pull Up Enable Register
[...]
# void pulsador()
# Espera pulsación pulsador y regresa devolviendo el CHIP ID
pulsador:
   # r0 - Dirección base del PIOB
   # r1 - Máscara del pulsador
   # r2 - Contenido del registro de E/S del PIOB
   push {lr}            @ No usamos los registros r4-r7
   # Configuración del pin del pulsador como entrada con pull-up
   ldr  r0, =PIOB               @ Dirección base del Controlador PIOB
   ldr  r1, =PULMSK             @ Máscara con el bit del pulsador activado
   str  r1, [r0, ???????]       @ Deshabilita interrupciones asoc. al pin
   str  r1, [r0, ???????]       @ Activa el pull-up
[...]
```

Figura 4: Ejemplo de fragmento de código proporcionado a los estudiantes

varro, Rafael Mayo Gual y Enrique S. Quintana Ortí Aplicación para la gestión y calificación de actividades ECTS. En Actas de las XIX Jornadas de Enseñanza Universitaria de la Informática, Castellón, 2013.

[2] Sergio Barrachina Mir, Maribel Castillo Catalán, José M. Claver Iborra y Juan C. Fernández Fernández. Prácticas de introducción a la arquitectura de computadores con el simulador SPIM. Pearson Educación, 2013.

[3] Sergio Barrachina Mir, Maribel Castillo Catalán, Germán Fabregat Llueca, Juan Carlos Fernández Fernández, Germán León Navarro, José Vicente Martí Avilés, Rafael Mayo Gual y Raúl Montoliu Colás. Introducción a la Arquitectura de Computadores con Qt ARMSim y Arduino. http://lorca.act.uji.es/book/practARM/, 2014.

[4] Sergio Barrachina Mir, Germán Fabregat Llueca, Juan Carlos Fernández Fernández y Germán León Navarro. ARMSim y QtARMSim: simulador de ARM para docencia. En Actas de las XXI Jornadas de Enseñanza Universitaria de la Informática, Andorra, 2015.

[5] Alan Clements. Selecting a processor for teaching computer architecture. En *Microprocessors and Microsystems*, 23(5), 281-290. Elsevier, 1999.

[6] Alan Clements. ARMs for the poor: Selecting a processor for teaching computer architecture. En *Frontiers in Education Conference (FIE)*, pp. T3E 1–6. IEEE, 2000.

[7] Alan Clements. The undergraduate curriculum in computer architecture. En *IEEE Micro*, 20(3), 13–22. IEEE, 2000.

[8] Peter Jamieson. Arduino for teaching embedded systems. Are computer scientists and engineering educators missing the boat? In Proceedings of the 2010 International Conference on Frontiers in Education: Computer Science and Computer Engineering, 289–294, 2010.

[9] Edurne Larraza-Mendiluze, Néstor Garay-Vitoria, José I. Martin, Javier Muguerza, Txelo Ruiz-Vazquez, Iratxe Soraluze, José F. Lukas y Karlos Santiago, Game-Console-Based Projects for Learning the Computer Input/Output Subsystem, IEEE Transactions on Education, vol.56, no.4, 453–458, 2013.

[10] Dan O'Sullivan y Tom Igoe. Physical Computing: Sensing and Controlling the Physical World with Computers. Course Technology Press, 2004.

[11] Miguel Ángel Rubio, Carolina Mañoso Hierro y Ángel Pérez de Madrid y Pablo. Using Arduino To Enhance Computer Programming Courses In Science And Engineering. In Proceedings of the EDULEARN13 Conference, pp. 5127–5133, Barcelona, 2013.

[12] Miguel Ángel Rubio, Carolina Mañoso Hierro, Rocío Romero Zaliz y Ángel Pérez de Madrid y Pablo. Uso de las plataformas LEGO y Arduino en la enseñanza de la programación. En Actas de las XX Jornadas de Enseñanza Universitaria de la Informática, Oviedo, 2014.

[13] Phillip Torrone. Why the Arduino won and why it's here to stay. http://makezine.com/2011/02/10/why-the-arduino-won-and-why-its-here-to-stay/, febrero 2011.