Desarrollo de una interfaz RS-232 para el manejo de un coche de radiocontrol desde el PC

A. Muñoz, A. Millan, P. Ruiz-de-Clavijo, J. Viejo, E. Ostua, D. Guerrero
Grupo ID2 (Investigación y Desarrollo Digital).
Departamento de Tecnología Electrónica.
Escuela Técnica Superior de Ingeniería Informática.
Universidad de Sevilla. España.

Este trabajo presenta el desarrollo de un sistema de radiocontrol para un coche teledirigido. Se trata de un circuito que, conectado al interfaz serie de un PC, permite controlar el coche desde una aplicación en lenguaje G. Así, el proceso de desarrollo se plantea como una práctica de laboratorio muy interesante y atractiva para los alumnos que cubre materias como la programación de periféricos, la comunicación con equipos informáticos y el control de sistemas a nivel de software.

1. Introducción

En este trabajo se muestra el desarrollo completo de un sistema de radiocontrol de un coche teledirigido. Dicho sistema está orientado a la realización de prácticas de laboratorio dentro de la asignatura *Periféricos e Interfaces* (correspondiente a la titulación de *Ingeniería Técnica en Informática de Sistemas*) y cubre tanto el diseño del *hardware* como del *software*. El sistema propuesto cubre materias como la programación de periféricos, la comunicación con equipos informáticos y el control de sistemas a nivel de *software*. En concreto, el alumno aborda los siguientes aspectos: (a) comunicación a través de puerto serie, (b) programación de microcontroladores y (c) programación basada en flujos de datos (lenguaje G).

Por un lado, la programación del microcontrolador se realiza en lenguaje ensamblador mediante su correspondiente juego de instrucciones. Se ha decidido utilizar un microcontrolador en el sistema con objeto de familiarizar a los alumnos con su uso y además dotar al sistema de mayores posibilidades de versatilidad (para la realización de otros tipos de prácticas). Por otro lado, el lenguaje G se presenta como una opción muy adecuada para la programación de este tipo de sistemas, siendo a la vez un lenguaje muy sencillo. De esta forma, la realización de la práctica se estructura en tres etapas básicas: (1) realización del *hardware* de interacción con el sistema real sobre una PCB, (2) programación del sistema de comunicación en el microcontrolador y (3) programación del *software* de control en el PC. Con objeto de reducir la duración de la práctica se propone que los alumnos dispongan de la primera etapa ya terminada (PCB) aunque si se dispone de suficientes sesiones puede programarse como práctica en su totalidad.

El resto de este documento se estructura como sigue: en la sección 2.1 se explica el diseño experimental que debe seguirse para construir la PCB, en la sección 2.2 se comentan los detalles relativos a la parte *software* y en la sección 3 se presentan brevemente los resultados obtenidos.

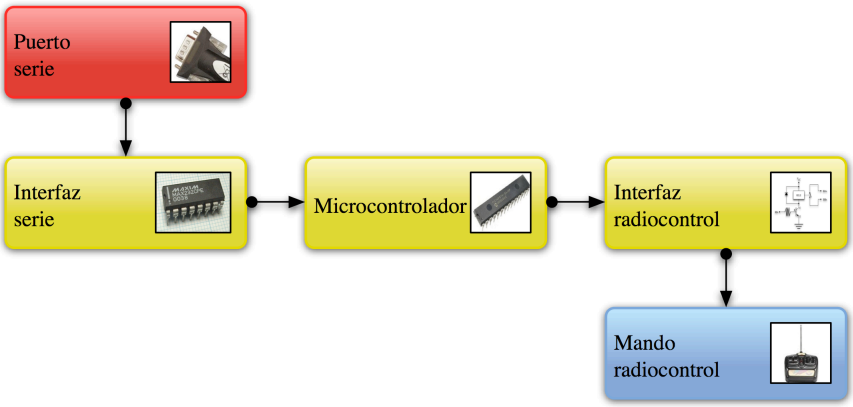

Figura 1: Diagrama de bloques del sistema *hardware*.

2. Diseño experimental

2.1. Diseño del *hardware*

El sistema *hardware* se ha organizado en tres bloques de complejidad reducida, como se muestra en la Fig. 1. De esta forma, se facilita una implementación por etapas, más adecuada al formato docente que se pretende.

En primer lugar se ha diseñado un bloque de comunicaciones (interfaz serie) que permite la conexión del sistema a un ordenador personal (PC). Debido a que el diseño gira en torno a un microcontrolador de la serie PIC y que éste incorpora una UART [1] en uno de sus bancos de entrada/salida, se ha decidido utilizar el puerto serie (RS-232) del PC para realizar esta conexión. Por tanto, ha sido necesario adaptar los niveles lógicos utilizados en el estándar RS232 v.24 (12 V y -12 V) a los niveles TTL utilizados en este tipo de microcontroladores; mediante un transceptor; concretamente de la serie MAX232 [2] (Fig. 2).

En segundo lugar, se ha colocado el propio microcontrolador. Éste se encarga de recibir las órdenes enviadas desde el PC a través de la interfaz RS-232 y activar las salidas en las líneas correspondientes. Concretamente, se ha utilizado un microcontrolador de la serie PIC 16F876 [3] que, básicamente, cuenta con los siguentes componentes:

- CPU interna de tipo RISC que opera a una frecuencia de hasta 20 Mhz de frecuencia.
- Memoria de programa de tipo *flash*.
- Memorias de datos de tipo RAM y EEPROM.
- Puertos de entrada/salida (de niveles TTL).
- Temporizadores.

En cuanto al conexionado del PIC, es necesario suministrarle las señales y tensiones básicas: alimentación, tierra, reloj, etc. Por su parte, las líneas de comunicación provenientes del PC (RX y TX) deben conectarse a las entradas de su UART (RC6/TX y RC7/RX). Además, se ha utilizado el banco de entrada/salida RB como puerto de salida del microcontrolador con objeto de gobernar un grupo de relés encargados de interactuar con el mando de radiocontrol. Dicho grupo de relés constituye el último bloque del sistema *hardware*.

Por último, el bloque de interfaz con el mando de radiocontrol está formado, como se ya se ha comentado, por un conjunto de relés. Así, simplemente se utiliza un transistor que funciona como interruptor;

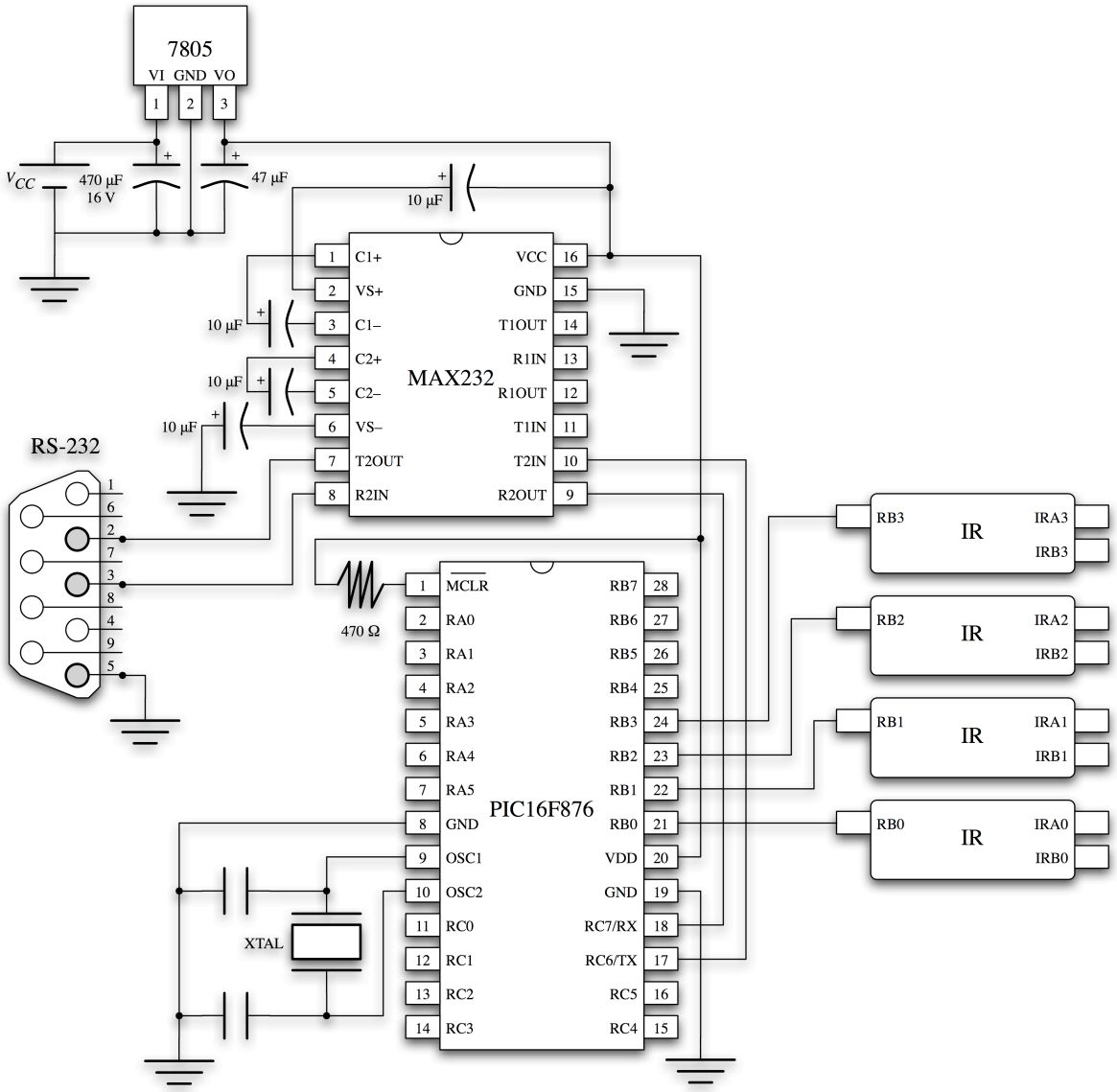

Figura 2: Esquema de componentes del sistema *hardware*.

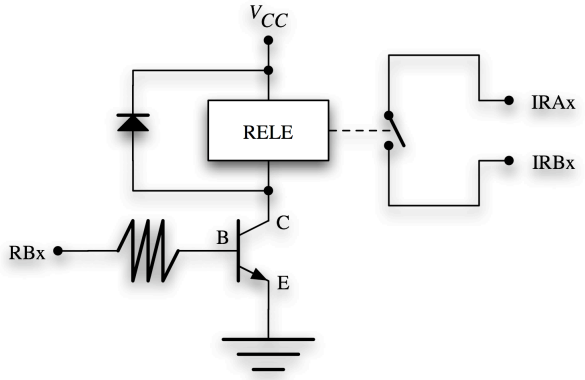

Figura 3: Etapa típica de la interfaz con el mando de radiocontrol.

controlado por la correspondiente línea del puerto RB (Fig. 3). De esta forma, se consigue controlar el mecanismo de apertura/cierra de cada relé lo que, a su vez, permite gestionar las diferentes órdenes enviadas al mando de radiocontrol.

Además, para generar la señal de alimentación de ambos chips se ha utilizado un regulador de tensión de la serie 7805 [4] con objeto de proteger el circuito en la medida de lo posible.

2.2. Diseño del *software*

2.2.1. Programación del microcontrolador

Una vez terminado el diseño del sistema hardware, debe escribirse el microcódigo que será ejecutado por el microcontralor. El microprograma necesario es relativamente sencillo y sólo tiene que encargarse de iniciar el PIC, preparar sus puertos de entrada/salida y configurar el sistema de comunicación mediante su unidad UART (a 9600 baudios); quedando a la espera de recibir algún dato. De esta forma, cada vez que la UART recibe un dato, debe generarse una interrupción por recepción cuyo gestor se encargará de leer el dato entrante y escribir en el puerto de salida. La función del PIC es por tanto convertir el dato, que llega en serie, al bus paralelo que lo une al sistema de relés y mantenerlo el tiempo correspondiente. Así, cada bit del dato escrito en el puerto de salida, corresponderá a una variación de tensión en la patilla del microcontrolador que utilizaremos para conmutar un transistor que a su vez conseguirá el efecto de cierre o apertura del relé de control.

El código propuesto es similar para toda la familia de microcontroladores de la serie PIC; siendo necesario acudir al manual de referencia del fabricante de cada modelo en concreto para adaptarlo a sus registros específicos. En este caso, como ya se ha comentado, se ha utilizado el modelo 16F876, así como la aplicación de ensamblado MPLAB [5] distribuida gratuitamente por el fabricante de esta serie (MICROCHIP). Concretamente, para el modelo escogido, se ha utilizado el puerto B (RB7-RB0) como salida, y los bits 7 y 8 del puerto C (RC6/Tx y RC7/Rx) como puerto serie; que es controlado por el módulo interno del microcontrolador denominado USART (*Universal Synchronous Asynchronous Receiver Transmitter*). El código desarrollado no se incluye aquí por motivos de espacio pero puede descargarse de la web de nuestro grupo (`http://www.dte.us.es/id2/`).

2.2.2. Programación del PC

El diseño del software de control a utilizar en el PC ha sido muy sencillo gracias a la facilidad de programación del lenguaje G. El programa correspondiente se ha generado mediante la herramienta LABVIEW [6] y consta básicamente de un panel de interruptores que permite controlar el coche (Fig. 5). En el diagrama en lenguaje G, cada interruptor se ha conectado a una constante binaria que representa la conmutación de relés necesaria para generar la orden asociada al interruptor (es decir, el movimiento deseado en el radiocontrol). Por último, estos datos son formateados mediante los controles de formación de arrays y de conversión de cadenas suministrados por LABVIEW y son transferidos al controlador de puerto serie *VISA Serial* (también suministrado por LABVIEW) para su envío (Fig. 5).

2.3. Montaje final

El sistema resultante se muestra en la Fig. 6. Una vez terminado, dicho sistema permite controlar el coche teledirigido desde el PC pudiendo enviarle todas las posibles órdenes soportadas por el mando de radiocontrol (adelante, atrás, giro izquierdo, adelante izquierda, atrás izquierda, giro derecho, adelante derecha y atrás derecha) desde la aplicación desarrollada en lenguaje G.

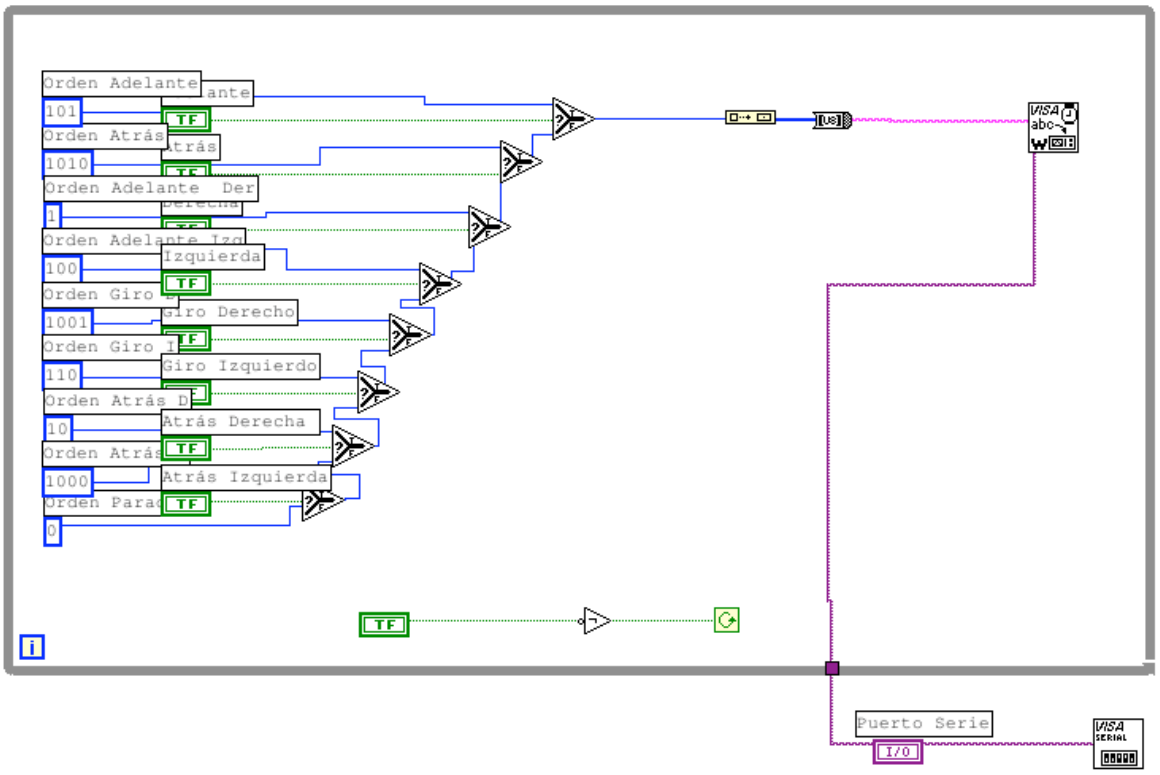

Figura 4: Código en lenguaje G utilizado.

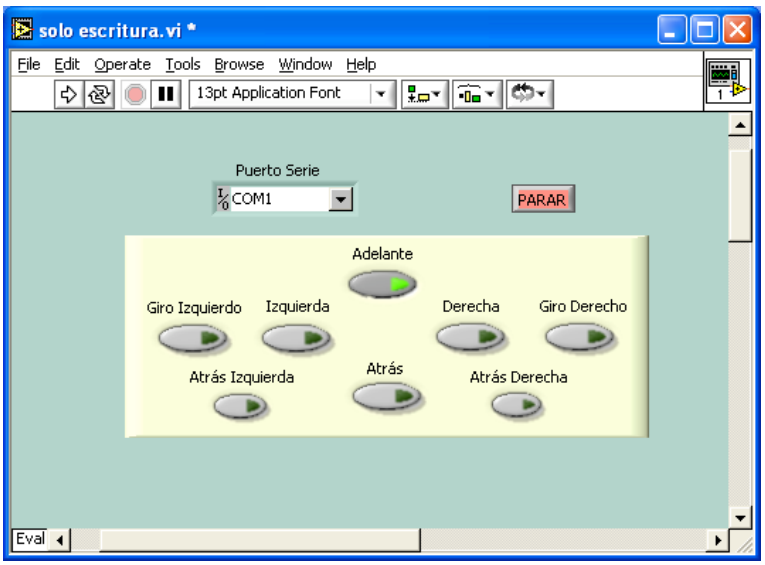

Figura 5: Panel de control disponible en el PC.

Figura 6: Fotografía del sistema desarrollado.

3. Conclusiones

En este trabajo se ha presentado la realización de una práctica de laboratorio consistente en la implementación de un sistema de radiocontrol para un coche teledirigido. El desarrollo del sistema cubre tanto el diseño del *hardware* como del *software* abarcando aspectos como la programación de periféricos, la comunicación con equipos informáticos y el control de sistemas a nivel de *software*. De esta forma, se proporciona a los alumnos un montaje muy interesante para ellos, en el que pueden poner en práctica tareas relacionadas con la comunicación a través de puerto serie, la programación de microcontroladores y la programación basada en flujos de datos (lenguaje G).

Referencias

[1] C. García de Celis, *El Universo Digital del IBM PC, AT y PS/2*. Grupo Universitario de Informática, 1994.

[2] "MAX232, MAX232I dual EIA-232 drivers/receivers (datasheet)." Texas Instruments Incorporated, October 2002.

[3] "PIC16F87X data sheet. 28/40-pin 8-bit CMOS flash microcontrollers." Microchip Technology Inc., 2001.

[4] "KA78XX/KA78XXA 3-terminal 1A positive voltage regulator." Fairchild Semiconductor Corporation, June 2001.

[5] "MPLAB IDE user's guide." Microchip Technology Inc., 2006.

[6] A. M. Lazaro and J. Fernandez, *LabVIEW 7.1: Programación gráfica para el control de instrumentación*. Thomson Learning Ibero, 2005.