



Universidad
Nacional
de Quilmes

Dado digital numérico en arduino

Trabajo final

Introducción a la programación de microcontroladores con Arduino

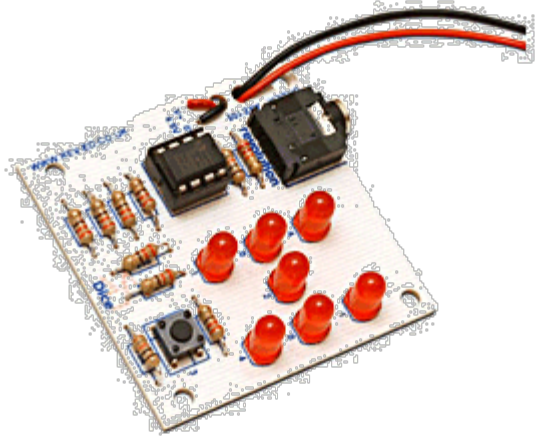
Susana Rosito Sayago
Tatiana Molinari
Alan Rodas Bonjour

Objetivo y alcance del trabajo

La idea era realizar un dado electrónico, capaz de mostrar el resultado de forma numérica (prendiendo leds que muestran un número). El dado debería ser capaz de presentar los números del 0 (cero) al 9 (nueve) y dar la sensación de interactividad y aleatoriedad.

Motivaciones

La idea surgió a partir de ejemplos de dados electrónicos sencillos, utilizando 7 leds, y un pulsador, y prendiendo cada uno de los leds de forma de mostrar una tirada del dado.

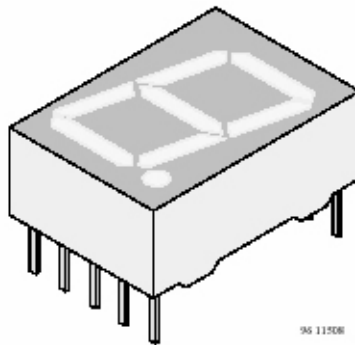


Un ejemplo de dado

electrónico visto

La disposición de los leds, permite representar como máximo 7 números, del 1 al 6, y el 0, el cual, a primera impresión para el usuario que presione el botón que activa el dispositivo, no parecería un resultado, sino una falla del dispositivo que no presenta ningún led encendido.

Decidimos tomar esta idea y combinarla con la idea de “números digitales”. De esta forma, podemos representar hasta 10 números distintos, e incluso si se quisiera, más.



Un ejemplo de numero digital

Pensamos que sería buena idea que al presionar el botón, comienzan a

distintos números, cada vez más lento, hasta quedar completamente detenido en un número, el cual sería el resultado de la tirada. Esto genera un efecto de interactividad y llama la atención del usuario.

idea que al presionar el aparecer rápidamente

Forma de trabajo

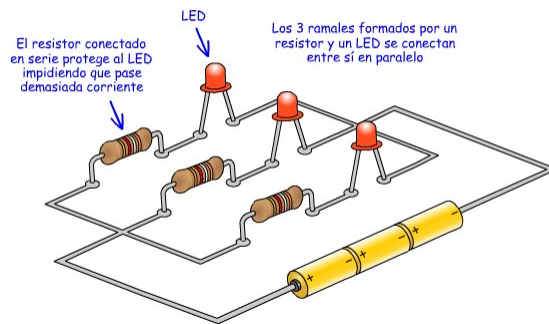
Por problemas organizacionales el equipo estuvo dividido la mayor parte del tiempo. En primer lugar, Alan y Susana se encargaron de conseguir los materiales electrónicos necesarios. Susana y Tatiana se encargaron de soldar la piezas y Alan se encargó del código. Finalmente, las chicas realizaron la presentación del trabajo al resto de la clase.

Materiales

- Una placa perforada
- 21 Leds de alto brillo color rojo
- 10 cables
- Un botón pulsador no bloqueante

Construcción

Se colocaron los leds en filas de 3 y se soldaron de forma paralela. Es decir, todas las patas positivas de la fila de LEDs se conectaban al mismo punto, mientras que las negativas, se conectaban a un punto en común para todas las filas, de donde salía un cable a tierra.



Ejemplo de conexión

paralela

Debido a los voltajes y amperajes utilizados, no se necesitaron resistencias, aunque hubiera sido recomendable usarlas, esto permitió abaratar los costos de construcción.

Cada uno de los puntos positivos, tendría un cable conectado a un pin del Arduino, de forma que al prender un solo pin, se prendería toda la hilera de LEDs.

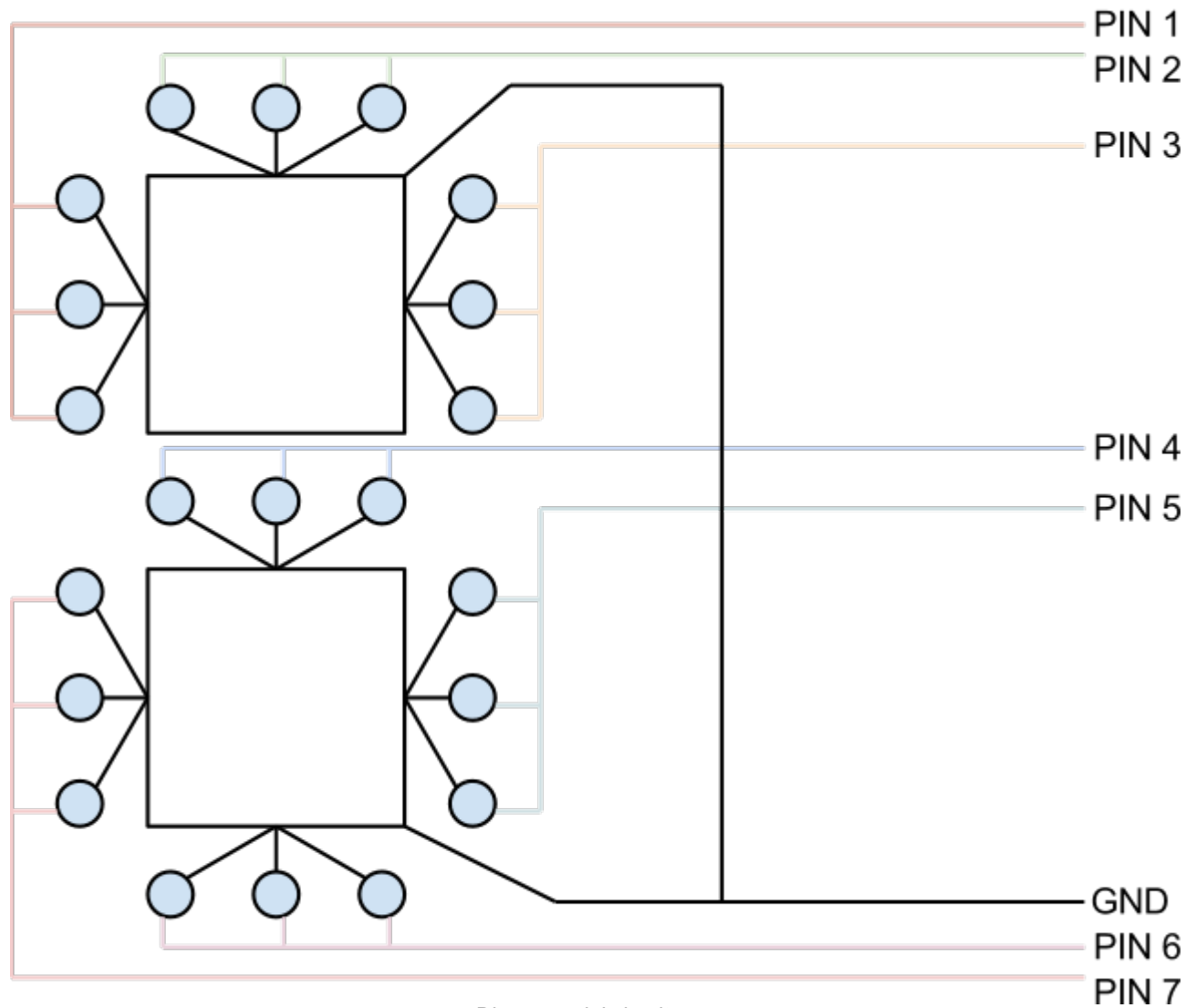
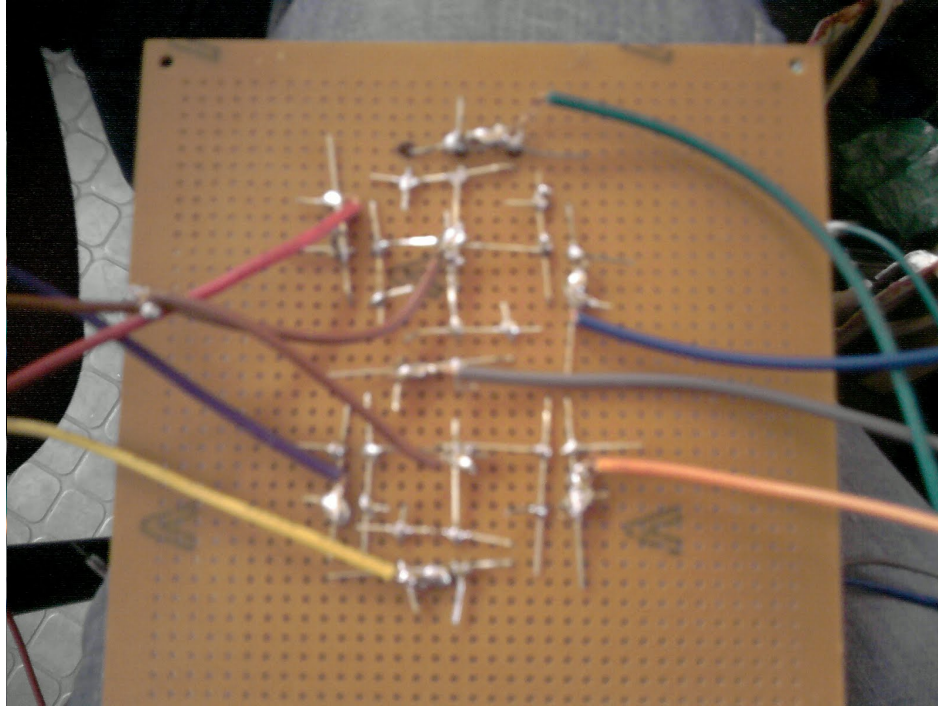


Diagrama del circuito

El diseño en paralelo permitía además, que si uno de los leds sufría un desperfecto, el resto de los leds de la fila seguirían funcionando correctamente. Esto no habría sido posible si se hubieran conectado de forma serial.



Muestra de las soldaduras

Finalmente, el botón, se conectaría con su parte positiva directamente a un pin de arduino, y la parte negativa a tierra.



El botón pulsador

Código

El código fue escrito desde cero, basándose en los ejemplos más sencillos de prender un LED, y de tomar el valor de un pulsador que provee el mismo entorno de desarrollo de Arduino. Se utilizaron arrays para representar el estado de los LEDs de acuerdo a cada número, y la función random de arduino proveía la funcionalidad más básica del dado. También se agregaron bloques de código que generan tiradas para mostrar interactividad, y un modo stand by.

Dificultades

El principal problema surgió por la falta de disponibilidad de tiempo de los integrantes, lo cual hacía que hubiera que trabajar a distancia. Sin embargo, las herramientas tecnológicas actuales, como el mail y el chat, solucionaron bastante este problema.

Un grave inconveniente fue el tema de verificar el funcionamiento del código. En la programación de hardware no se disponen de herramientas de debugging, y también es posible imprimir en consola, por lo cual, es bastante difícil verificar que el programa esta haciendo exactamente lo que se requiere.

Uno de los inconvenientes en el código que requería verificar que los valores fueran correctos, era el verificar cuando el botón estuviera presionado. El botón no parecía en principio funcionar correctamente. Esto llevo un buen tiempo de resolver.

Conclusiones

Podemos obtener como conclusiones finales los siguientes puntos.

- Programar microcontroladores requiere altas capacidades de abstracción, ya que no existen herramientas de debugging.
- La forma de realizar los circuitos puede influir seriamente en el código que uno esta realizando, por lo que es necesario conocer el circuito antes de programar.
- Si bien el costo de los materiales es, en principio, caro, no son prohibitivos, con lo cual, pueden realizarse cosas muy interesantes para el uso cotidiano con bajo costo.

Fue en general, una grata experiencia, que nos deja con ganas de haber contado con más conocimientos de electrónica para poder haber hecho algo más complejo.