UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

INSTRUMENTACIÓN

PRÁCTICA #1

Introducción al Lenguaje Arduino

Alumno
VÁSQUEZ CASTAÑEDA
CARLOS ANTONIO

Profesor
JOSÉ MANUEL
RAMÍREZ ZÁRATE

Grupo 395

Matrícula: 1155057

Febrero 12, 2020

Práctica #1: Introducción al Lenguaje Arduino

Carlos A. Vásquez Castañeda 1155057 Grupo 395

Febrero 12, 2020

Introducción

El desarrollo de la tecnología ha permitido condensar un gran poder de procesamiento en dispositivos de apenas algunos centímetros. Estos dispositivos en la actualidad se encuentran en la gran mayoría de los electrónicos modernos y nos brindan la posibilidad de obtener cierto comportamiento de nuestros componentes. El *microcontrolador* permite la utilización de sistemas "inteligentes" o, al menos programables.[1] Este dispositivo es capaz de toomar decisiones a través de la información que le es otorgada mediante sensores o cualquier dispositivo que sea capaz de enviar datos.

Objetivo

Familiarizar al estudiante con el entorno de desarrollo de Arduino y su funcionamiento mediante la utilización de un Arduino UNO o similar, un software de simulación y la creación del circuito físico.

Marco Teórico

A continuación se presentan algunas definiciones claves que serán de utilidad a medida que continuamos con el reporte:

1. Universidad Politécnica de Cartagena, "Introducción a los Microcontroladores," `www.bolanosdj.com.ar/MICRO/INTRODUCMICRCONT.pdf`.

- Circuito Integrado: un circuito integrado es una combinación de resistores, capacitores, diodos y transistores, todo dentro de una misma placa y con una conexión dada entre ellos.[2]

- Microcontrolador: se utiliza para el gobierno de uno o varios procesos. En grandes rasgos, son circuitos integrados que son capaces de ejecutar órdenes que fueron grabadas en su memoria.

- Arduino: es un plataforma de electrónicos de uso libre basado en hardware y software fácil de utilizar. Los circuitos Arduino son capaces de mandar y recibir señales I/O para lograr la toma de decisiones y poder realizar distintas funciones.[3]

- EEPROM: por sus siglas en inglés significa Electronically Erasable Programmable Read Only Memory, y, como su nombre lo dice, esta memoria es capaz de reprogramarse para cargar distintos programas y que el comportamiento de nuestro microcontrolador varíe.

Materiales y Equipo

- Arduino UNO
- Arduino IDE
- Proteus 8 Professional
- LEDs (1)
- Resistencia 220 Ω (1)
- Referencias de internet (StackExchange, Arduino.cc)

2. Scotten W. Jones, "Introduction to Integrated Circuit Technology," 2012, `https://www.icknowledge.com/freecontent/Introduction%20to%20IC%20technology%20rev%205.pdf`.

3. Arduino, "What is Arduino?," 2020, `https://www.arduino.cc/en/Guide/Introduction`.

Desarrollo

Como introducción al lenguaje de programación de Arduino se programará el dispositivo para el encendido y apagado de un Diodo Emisor de Luz (LED), el cual tendrá este comportamiento indefinidamente. Dada la arquitectura del lenguaje de programación de Arduino y la arquitectura del dispositivo, existen tres pasos generales para escribir un programa en Arduino.

El primer paso consiste en definir variables y librerias que se utilizarán dentro de nuestro programa. Éstos darán una funcionalidad mucho mayor al circuito y nos ayudarán a tener un código que sea fácil de mantener.

El segundo paso es donde se definen las entras y salidas del microcontrolador en el circuito del Arduino. Arduino cuenta con un microcontrolador (el ATmega328p) que es capaz de definir sus distintos puntos de acceso como entradas o salidas, las cuales le permiten enviar información o leer información de otros sensores/dispositivos.

Finalmente, después de haber definido variables, librerias y las entras y salidas, podemos proseguir a realizar el programa como tal. En esta sección se le dará la funcionalidad deseada al microcontrolador y se comportará de esta manera de forma indefinida, en un ciclo infinito (por lo menos mientras esté energizado el circuito).

```
1  int led = 13;
```

La línea de código anterior representa el primer paso. Dado que nuestro programa es lo suficientemente simple no necesitaremos añadir librerías externas. Sin embagro, la utilización de una variable es bastante útil para asignarle un nombre a uno de los contactos del microcontrolador. Si en algún momento nuestro programa se vuelve bastante complejo y necesitamos modificarlo, es mejor simplemente modificar una variable definida con anterioridad a modificar todo el código subsiguiente. Esto es a lo que se refiere "código mantenible".

```
1  void setup () {
2    pinMode(led, OUTPUT);
3    }
```

Las siguientes líneas de código representan el paso número dos, donde estamos definiendo el contacto 13 como una salida. Podemos notar que en lugar de utilizar un 13, utilizamos el

nombre de la variable como tal. Esto vuelve más amigable el código y fácil de leer.

```
1  void loop () {
2    digitalWrite(led, 1);
3    delay(1000);
4    digitalWrite(led, 0);
5    delay(1000);
6  }
```

La última parte añadida al código representa el paso número tres, donde se repiten las acciones de manera indefinida. Aquí podemos observar que ordenamos a Arduino a escribir un *1* o un voltaje alto en el pin 13. Después de realizar esto esperamos aproximadamente un segundo y volvemos a realizar el mismo procedimiento, pero esta vez escribiendo una señal de voltaje bajo, o un *0*.

Diagrama de Flujo

Observando el proceso mediante bloques de diagrama de flujo luciría similar a esto:

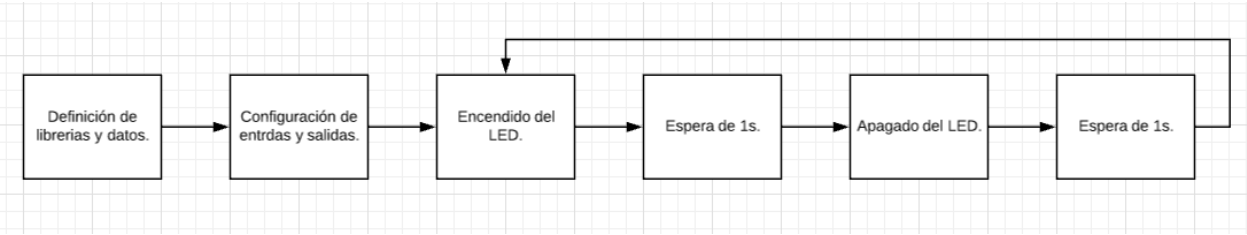

Figure 1: Proceso del programa elaborado anteriormente.

El proceso es bastante sencillo dado que la funcionalidad es bastante limitada, sin embargo, posteriormente con la lectura y escritura de datos y otros elementos añadidos, los diagramas de flujo se volverán una herramienta que nos permitirá visualizar el proceso a gran escala y entenderlo mucho mejor.

Código

A continuaciön se muestra el código en su totalidad.

```
1  int led = 13;
2
3  void setup () {
4    pinMode(led, OUTPUT);
5  }
6
7  void loop () {
8    digitalWrite(led, 1);
9    delay(1000);
10   digitalWrite(led, 0);
11   delay(1000);
12 }
```

Diagrama Esquemático

Una vez heco el código y después de haber revisado si existen errores, podemos proceder a la realización del circuito. Dado que nuestro circuito es bastante sencillo, las conexiones también lo serán.

Como se ha expresado en el diagrama de flujo, nuestras conexiones comenzarán desde el circuito del Arduino, específicamente desde el contacto #13, dado que este contacto lo hemos definido como salida. Desde este contacto obtendremos la señal que viajará a través de un conductor, pasando por el LED y a su vez por una resistencia de 220 Ω que regula la corriente y evita que nuestros dispositivos se dañen. Esta resistencia estará conectada al nivel de referencia o tierra para completar el circuito y que se genere una corriente eléctrica.

A continuación se muestra el diagrama esquemático:

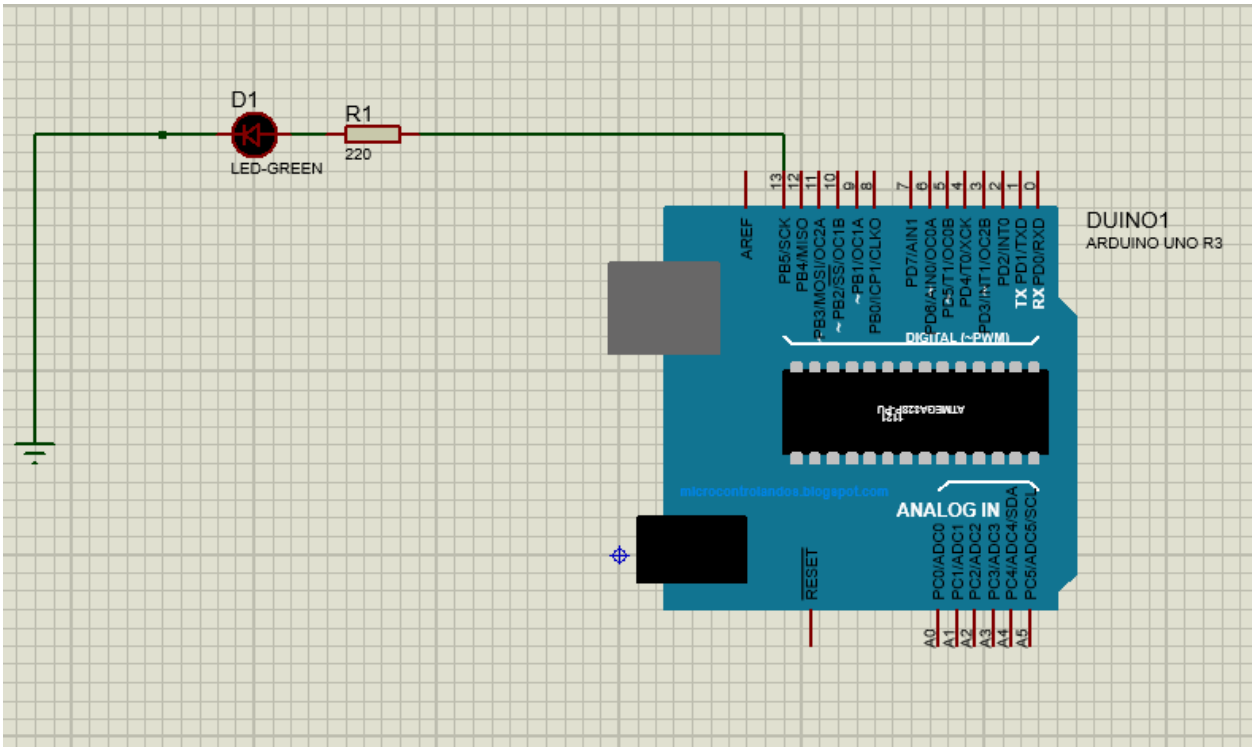

Figure 2: Diagrama de las conexiones necesarias y componentes utilizados.

Conclusión

La utilización de los diversos componentes electrónicos en la actualidad es tan amplia que incluso como persona promedio es bastante útil comprenderlos y si es posible modificarlos. Esta práctica demuestra en su mayoría que la manipulación de estos componentes no tiene que ser complicada. Dado el código tan sencillo, se encuentra lo más optimizado posible, lo cual es algo que se busca más y más en los distintos aparatos electrónicos que se diseñan. Para lograr hacerlos más pequeños se necesita aprovechar de manera inteligente la capacidad y memoria de cada uno de los componentes, entonces es absolutamente necesaria la optimizacón del código para disminuir los tiempos de procesamiento.

Anexos

A continuación se presenta el *hexdump* del archivo producido por el ensamblador. Estos archivos hexadecimales son de gran utilidad en programas más complejos para su *debugging*.

```
:100000000C945C000C9479000C9479000C947900A9
:100010000C9479000C9479000C9479000C9479007C
:100020000C9479000C9479000C9479000C9479006C
:100030000C9479000C9479000C9479000C9479005C
:100040000C942E010C9479000C9479000C94790096
:100050000C9479000C9479000C9479000C9479003C
:100060000C9479000C947900000000070002010054
:10007000000304060000000000000000000102040864
:1000800010204080010204081020010204081020002
:10009000040404040404040402020202020203032E
:1000A000030303030000000250028002B000000CC
:1000B0000000240027002A0011241FBECFEFD8E043
:1000C000DEBFCDBF11E0A0E0B1E0EAE1F4E002C0A4
:1000D00005900D92A230B107D9F721E0A2E0B1E07E
:1000E00001C01D92AB30B207E1F70E94FE010C94F3
:1000F0000B020C94000061E0809100010C94C3009D
:10010000CF93DF93C0E0D1E061E088810E94FF00DF
:1001100068EE73E080E090E00E949D0160E08881DD
:100120000E94FF0068EE73E080E090E0DF91CF91E5
:100130000C949D01833081F028F4813099F0823055
:10014000A1F008958630A9F08730B9F08430D1F459
:1001500080918000 8F7D03C0809180008F77809395
:1001600080000895 84B58F7702C084B58F7D84BDEB
:1001700008958091B0008F7703C08091B0008F7D8B
:100180008093B0000895CF93DF9390E0FC01E45892
:10019000FF4F2491FC01E057FF4F8491882361F1C8
:1001A00090E0880F991FFC01E255FF4FC591D49153
:1001B000FC01EC55FF4FA591B491611109C09FB7A7
:1001C000F894888120958223 8883EC912E230BC09C
:1001D000623061F49FB7F8948881322F3095832381
:1001E0008883EC912E2B2C939FBF06C08FB7F89479
:1001F000E8812E2B28838FBFDF91CF9108951F9325
:10020000CF93DF93282F30E0F901E859FF4F849115
:10021000F901E458FF4FD491F901E057FF4FC49121
:10022000CC23C1F0162F81110E949A00EC2FF0E030
:10023000EE0FFF1FEC55FF4FA591B4919FB7F894B7
:10024000111104C08C91D095D82302C0EC91DE2B03
:10025000DC939FBFDF91CF911F9108951F920F9262
:100260000FB60F9211242F933F938F939F93AF93C9
:10027000BF938091030190910401A0910501B09179
```

```
:1002800006013091020123E0230F2D3720F401965F
:10029000A11DB11D05C026E8230F0296A11DB11DA9
:1002A000209302018093030190930401A093050120
:1002B000B0930601809107019091080 1A091090176
:1002C000B0910A010196A11DB11D80930701909381
:1002D0000801A0930901B0930A01BF91AF919F91CA
:1002E0008F913F912F910F900FBE0F901F901895F7
:1002F0003FB7F8948091070190910801A0910901FE
:10030000B0910A0126B5A89B05C02F3F19F00196B0
:10031000A11DB11D3FBFBA2FA92F982F8827820F8B
:10032000911DA11DB11DBC01CD0142E0660F771FDB
:10033000881F991F4A95D1F70895CF92DF92EF92C7
:10034000FF92CF93DF936B017C010E947801EB0158
:10035000C114D104E104F10479F00E9478016C1B0E
:100360007D0B683E7340A0F381E0C81AD108E10814
:10037000F108C851DC4FECCFDF91CF91FF90EF90A7
:10038000DF90CF900895789484B5826084BD84B561
:10039000816084BD85B5826085BD85B5816085BD80
:1003A000EEE6F0E0808181608083E1E8F0E0108299
:1003B000808081826080838081816080 83E0E8F0E0DA
:1003C000808081816080 83E1EBF0E0808184608083C4
:1003D000E0EBF0E0808181608083EAE7F0E08081FB
:1003E000846080838081826080838081816080835B
:1003F000808081806880831092C10008950E94C301AB
:100400000E947B00C0E0D0E00E9480002097E1F3D2
:0A0410000E940000F9CFF894FFCF1E
:02041A000D00D3
:00000001FF
```

References

Arduino. “What is Arduino?” 2020. https://www.arduino.cc/en/Guide/Introduction.

Cartagena, Universidad Politécnica de. “Introducción a los Microcontroladores.” www.bolanosdj.com.ar/MICRO/INTRODUCMICRCONT.pdf.

Jones, Scotten W. “Introduction to Integrated Circuit Technology.” 2012. https://www.icknowledge.com/freecontent/Introduction%20to%20IC%20technology%20rev%205.pdf.