Arduino en Acción: Simulación de Señales Fisiológicas

Gonzalo Quilodrán Neira

9 de noviembre de 2025

Resumen

En este documento se presenta el desarrollo del taller *Arduino en Acción: Simulación de Señales Fisiológicas*, orientado a introducir el uso de Arduino, ademas de enseñar el uso de distintos sensores y actuadores. Todo lo relacionado a esta instancia, se encuentra en el Repositorio oficial del taller.

1. Introducción

En este documento se presenta una introducción al uso de Arduino como herramienta de apoyo en la Ingeniería Biomédica, orientada a comprender los principios de electrónica y programación que pueden ser aplicados a sistemas de medición y control biológico.

A través de la exploración de componentes básicos, se busca sentar las bases para el desarrollo de dispositivos capaces de registrar o generar señales relacionadas con el cuerpo humano.

El presente taller tiene como propósito integrar los fundamentos de la electrónica con conceptos biomédicos, fortaleciendo la comprensión de cómo la instrumentación electrónica y el procesamiento de señales son esenciales para el diseño de equipos y herramientas en el ámbito de la salud.

2. Fundamentos de Electrónica

La electrónica es la rama de la ingeniería que estudia el comportamiento y control del flujo de electrones a través de distintos materiales y componentes. Su comprensión es esencial para diseñar, analizar y construir circuitos capaces de procesar señales, controlar dispositivos y realizar tareas automáticas. En el contexto del taller, conocer los fundamentos electrónicos permite entender cómo funciona el hardware que se conecta al Arduino y cómo interactúan entre sí los distintos elementos del circuito.

Los conceptos principales son el voltaje (V), la corriente (I) y la resistencia (R). El voltaje representa la diferencia de potencial eléctrico entre dos puntos, la corriente es el flujo de carga eléctrica a través de un conductor, y la resistencia mide la oposición al paso de esa corriente. Estos tres parámetros se relacionan mediante la Ley de Ohm:

$$V = I \times R$$

Esta relación es la base para analizar y diseñar cualquier circuito, desde los más simples hasta los más complejos.

Dentro de los componentes básicos se encuentran las resistencias, que limitan la corriente; los condensadores, que almacenan energía temporalmente; los diodos, que permiten el paso de corriente en un solo sentido; y los transistores, que actúan como interruptores o amplificadores. Comprender el comportamiento de estos elementos es crucial para poder controlar señales eléctricas de manera segura y eficiente.

Además, se estudian configuraciones de circuitos en serie y paralelo, así como la interpretación de esquemas eléctricos, el uso de protoboards y la correcta conexión de componentes. Estos conocimientos sirven de base para comprender cómo el Arduino interactúa con sensores, actuadores y otros módulos externos, integrando la teoría electrónica con la práctica de la programación y el control.

2.1. Relación con la Ingeniería Biomédica

En la Ingeniería Biomédica, la electrónica tiene un papel esencial en la creación de dispositivos de diagnóstico, monitoreo y rehabilitación. Ejemplos de esto son los electrocardiógrafos, pulsómetros, prótesis activas o sistemas de estimulación eléctrica. El conocimiento de circuitos y sensores permite diseñar herramientas que capten y procesen señales biológicas, garantizando precisión y seguridad. A través del taller de Arduino, los estudiantes pueden explorar de manera accesible los principios detrás de estos sistemas, comprendiendo cómo los fundamentos electrónicos se aplican directamente al desarrollo tecnológico en el ámbito de la salud.

3. Fundamentos de Arduino

Para realizar proyectos de electrónica no es necesario ser un experto en el tema o tener un doctorado, gracias a la existencia de **Arduino**: placas con microcontroladores de hardware libre y un software de fácil acceso. Estas permiten desarrollar desde proyectos simples hasta aplicaciones avanzadas, donde la placa actúa como una especie de *cerebro*, enviando órdenes a los componentes conectados, como encender LEDs, apagar motores o detectar frecuencias.

Lo que hace destacar a las placas Arduino es su versatilidad y la facilidad para aprender a utilizarlas. Para comunicarnos con ellas utilizamos el **Entorno de Desarrollo Integrado (IDE) de Arduino**, programando principalmente en el lenguaje **C++**.

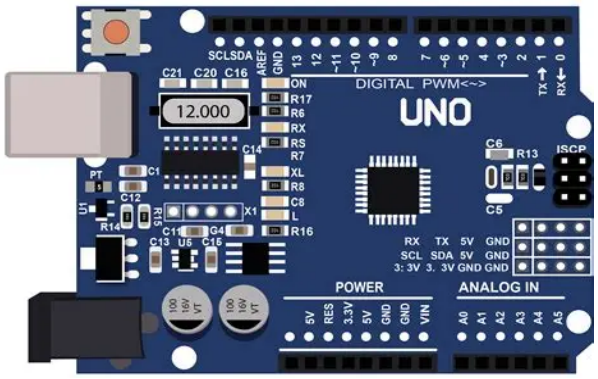

Figura 1: Placa Arduino UNO R3

Existen distintos tipos de placas Arduino, como la **Nano**, **Mega**, **Leonardo**, entre otras. En el contexto de este taller, utilizaremos la **Arduino UNO R3**, cuyas principales características se muestran en la siguiente tabla:

Microcontrolador	ATmega328P
Voltaje de operación	5V
Entrada de alimentación recomendada	7 a 12V
Pines digitales	14
Pines analógicos	6
Pines PWM	6

Como se aprecia en la tabla, la placa Arduino UNO cuenta con **26 pines principales**, alguno de los más importantes siendo:

- **Pines analógicos:** permiten leer valores continuos, proviniente de sensores analogos, como sensores de temperatura o luz. Estan enumerados del A0 a A5.
- **Pines digitales:** Se usan para entrada/salida de valores digitales, permiten endercer leds, leer sensores y activar actuadores. Estan enumerados del 0-13.
- **Pines PWM:** Estos pines emiten señales digitales que simulan valores analógicos, ideales para controlar la velocidad de motores o el brillo de LEDs. Estan marcados con el simbolo

- **pin VIN:** Su función es entregar una entrada para la alimentación de 5 voltios.
- **Pines GND:** Sirve como una entrada de tierra, se utiliza para conectar periféricos externos se alimenten de la placa Arduino
- **Pin 3.3v:** Pin que funciona como salida de 3.3 voltios.

Cada tipo de pin tienen su función y forma de entregar o recibir las entradas/salidas. Es importante tomar esto en cuenta.

4. Sensores, actuadores y periféricos

Bueno, ya conocemos el çerebro"de los proyectos con Arduino, pero por si solo no puede medir algo o reaccionar con lo que pasa a su alrededor, para ello existen dispositivos que permiten esto:

- **Sensores:** Estos dispositivos tienen la capacidad de detectar magnitud físicas o químicas, permiten transformarlas en variables eléctricas (llamadas de igual manera variables de instrumentación) para que la placa y su código trabajen con ella. Existen sensores de todo tipo, de temperatura, pH, humedad, pulsaciones, intensidad lumínica, etc. Los datos que obtienen los devuelven en salida digitales o analógicas.
- **Actuadores:** Estos dispositivos, actúan por medio de señales entregadas por el Arduino. Transforman la energía eléctrica, neumática o hidráulica en procesos. Se pueden catalogar en distintos tipos como bombas, electrónicos o motores.
- **Periféricos:** Este termino se le atribuye a los aparatos y dispositivos auxiliares e independientes al Arduino. Dentro de esta categoría se encuentran pantallas LCD, teclados, memorias externas, pantallas táctiles, etc.

Cada sensor, actuador o periférico funciona de forma distinta y entrega distintos datos. En el contexto del taller, nos interesa conocer los siguientes:

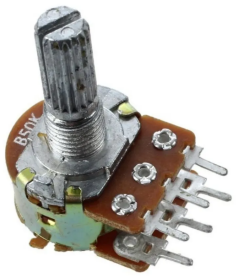

Figura 2: Potenciómetro

Figura 3: Buzzer

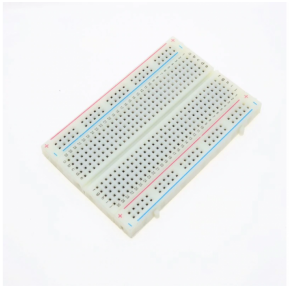

Figura 4: Protoboard

- **Potenciometro:** Es un componente eléctrico con un valor de resistncia variable que se puede ajustar manualmente. Dependiendo de la posición de su eje, devolvera valores entre 0v a 5v, y el codigo del arduino los traducira como valores enteros entre 0 y 1023 . 2
- **Buzzer o Zumbador activo:** Transductor que convierte señales eléctricas en sonido. Construido con una cerámica y dis pequeñas placas metalicas, ademas de un osilador que le permite vibrar para emitir sonidos audibles. Posee una terminal positiva y otra negativa. 3
- **LEDs:** Diodo emisor de luz (en ingles Light Emitting Diode) son fuentes de luz que se encienden al recivir corriente electrica. Las que utilizaremos tienen un **ánodo**, la pata positiva y más larga, y un **cátodo**, la pata negariva.

- **Resistencias:** Componentes eléctricos que transforman energía eléctrica en térmica (ejecto Joule). Su principal utilidad es la de limitar el voltaje que le llega a ciertos componentes, evitando que estos se dañen o se quemen.
- **Cables jumper:** Conductores flexifles que permiten conectar el Arduino con otros componentes sin soldarlo.
- **Protoboard:** También llamada placa de prueba, permite crear prototipos o circuitos electrónicos sin la nesecidad de soldar los componentes, permitiendo cambiarlos en caso de error o que estos se dañen. 4

5. Programación en C++

Se podría hacer un taller o varios para enseñar a programar en C++, pero lo principal que deben conocer son tres cosas:

- **Variables:** Como estamos trabajando en C++, es necesario declarar las variables antes de usarlas, especificando su tipo: **int** para valores numéricos enteros, **float** para valores numéricos decimales, **unsigned int/float** para valores enteros/decimales unicamente positivos, **str** para cadenas de texto o **char** para caracteres. Al trabajar con Arduino, se deben declarar variables para identificar en que pines están colocados los sensores/actuadores.
- **Funciones:** Como todo lenguaje de programación se pueden crear funciones, secciones de código que realizan una función en especifico. Uno puede crearlas o usar algunas ya existentes. De estas ultimas, las más útiles al trabajar con Arduino son:

PinMode(pin,modo)	Define si un pin especifico es de entrada o salida (INPUT/OUTPUT).
digitalWrite(pin, valor)	A un pin de salida, se envía un valor o bajo de energía (HIGH/LOW).
digitalRead(pin)	Lee un valor lógico de un pin.
analogRead(pin)	Lee un valor de un pin analógico.
analogWrite(pin,valor	Envia una señal PWM a un pin compatible con salida analógica.
delay(n)	Añade una espera de n milisegundos, deteniendo el código.
tone(pin, frecuencia)	Envía una señal de audio a un pin con una frecuencia especifica.
random(min,max)	Devuelve un valor aleatorio entre un valor mínimo y máximo.
abs(n)	Entrega el valor absoluto de un valor n.
Serial.begin(velocidad)	Inicializa la comunicación serial entre el pc y el circuit.
Serial.print(valor)	Envía al monitor serial un valor en especifico.

- **setup()/loop():** Estas también son funciónes, y son la base de la estructura del código. **Setup** Se ejecuta una sola vez al iniciar el Arduino, su función es dar un espacio para definir los pines, inicializar comunicación serial o cualquier otro ajuste inicial. Mientras que **loop** ejecuta lo que tenga en su interior continuamente, aqui van las instrucciones principales del programa.

6. Buenas practicas

Para trabajar de forma segura y eficiente en proyectos de electronica con Arduino, hay que tener ciertos cuidados y seguir las siguientes recomendaciones:

- Antes de conectar la placa a alguna alimentación electrica, recuerda revisar bien las conexiones, que los cables esten en su lugar y con la polaridad correcta.
- Aunque se pueden soldar componentes directo a la placa, es recomendable usar una Protoboart, asi en caso de un error o ante la quema de un componente, es más facil solucionarlo.
- Usar resistencias para evitar la queda de los componentes. Esto es especialmente util cuando trabajes con LEDs, sensores o botones.

- No trabajes y toques el circuito con las manos humedas.
- Respecto al codigo, es recomendable comentarlo, facilitando la comprención de terceros y permite recordar que hace cada función. También es recomendable tener una versión del código que funcione como una copia de seguridad.
- Evita usar los pines digitales 0 y 1, ya que estos están vinculados al puerto serial.

7. Proyecto del Taller

Ahora bien, ya con el conocimiento teórico, hay que ponerlo en practica, y que mejor que con la construcción de un circuito simple pero que pondrá aprueba lo aprendido. Lo que haremos es un **simulador de señales fisiológicas**, a través de señales eléctricas simularemos el pulso de una persona, tomando en cuenta que el pulso puede ser normal si esta entre **80 y 100 ppm**, si se fuera de ese rango se puede llegar a considerar como **Taquicardia o bradicardia**, o que puede llegar a un máximo, ya sea **0 ppm** donde no hay actividad cardíaca detectable, o si llega a la **frecuencia cardíaca máxima**. Este circuito simulativo puede ser mejorado o incluso medir pulsaciones reales, pero como el taller esta orientado a un nivel de inicializar en el tema, no se abordara mucho en ese tema.

7.1. Montaje del Circuito

Ya conociendo los sensores, actuadores y periféricos, ahora queda montar el circuito sobre la protoboart. Primero Conecten todo en su lugar, dejando la palanca del potenciómetro libre para poder manipularla, después conectar los cables Jumper al Arduino. Recuerda conectar el cable de 5v y el GND a los carriles positivo y negativo de la protoboar respectivamente.

El circuito debería quedar similar a la Fig. 5, los pines que conecten los sensores al Arduino pueden variar, pero eso deberá ser especificado en el código.

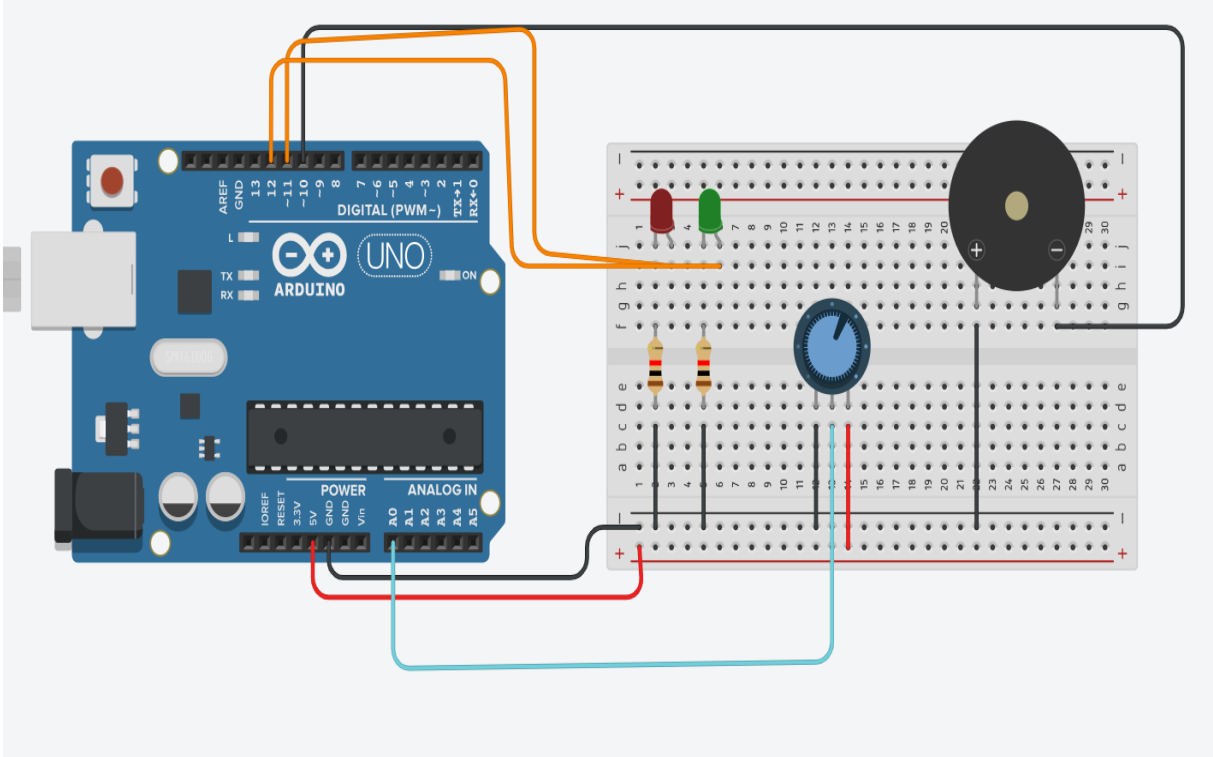

Figura 5: Estructura del circuito montado

7.2. Programación del circuito

Tras cmontar el circuito, toca programar el Arduino. Para eso antes hay que instalar el Arduino IDE 2.3.6 desde la Pagina de insalación del Arduino IDE. Trás instalarlo, puedes crear un archivo en blanco o utilizar el boceto del código que se encuentra en el repositorio (en este, se encuentran unas funciones básicas y una pseudo estructura que debes completar).

Lo principal que debe tener es la defínción de los pines a los que se conectaron los sensores y actuadores, que lea los valores del Potenciometro y divida su valor en cinco partes: los valores minimos (0) y máximos, los valores que equivalen a las secciones de valores mayores de o y menores de 80, los valores entre 80 y 100, y los valores mayores a 100 y menores que el máximo. Dependiendo de cada valor, deben hacer que las LEDs se enciendan o apagen, lo mismo con los pitidos del buzzer. Los intervalos a considerar son: en el mínimo y maximo la led roja de debe mantener activa, en]0,80[y en]100, máx.[a led roja debe parpardear, y de [80,100] laa led verde se debe mantener encendida.

Como el potenciometro devuelve valores desde 0 a 1023, la razón variara dependiendo del valor máximo colocado, respecto a esto último, la **Frecuencia cardiaca máxima** varia dependiendo de la edad de quien se hable: **Fcm = 220 - Edad**. En el contexto del taller, se considerara la edad como un valor superfluo, por lo tanto la razón de dividir los valores del potenciometro es —. En caso de que se desee cambiar eso, simplemente habria que realizar una regla de tres simples entre el valor máximo del potenciometro y el Fcm (1023==Fcm) y los valores 80 y 100 de frecuencia cardiaca.

7.3. Resultados Esperados

Tras armar y progrmar el circuito, deberia funcionar de la siguiente forma:

- **Caso 1:** Si el potenciometro entrega valores equivalente a 80 a 100 pulsaciones, la led verde se enciende y el buzzer emitira un pitido en ciclos espaciados y constantes.
- **Caso 2:** Si el potenciometro devuelve valores menores a 80 pulsaciones, la led roja se encendera y apagara en ciclos, el buzzer comenzara a emitir pitidos más lento y abruptos. Esto representando una **Bradicardia**.
- **Caso 3:** En caso de que el potenciometro entregue valores mayores a 100 pulsaciones, la led roja se encendera y apagara en ciclos, el buzzer emitira un pitido de forma más constante. Esto representando una **Taticardia**.
- **Caso 3:** En el caso que el potenciometro devuelva un valor equivalente a 0 pulsaciones, la led roja se mantendra encendida y el buzzer emitira un pitido constante. Esto representaria un **Paro cardiaco** o una **Muerte clinica** del paciente teoirco.
- **Caso 4:** En caso de que el Potenciometro devolviera un valor máximo, la led roja se encenderia y el buzzer sonaria aun más rapido. En este caso se representaria una **Taquicardias severas**. En caso de que esto se prolongue, llegaria un momento que el corazon no podria más, asi que se iria al caso 3.

En el caso de que esto no sea así, revise su código o la conexión de los pines. En el repositorio del taller, encontrara el código ideal de funcionamiento.

8. Conclusiones

La finalidad del desarrollo del circuito anterior era para poder entender tanto teóricamente como prácticamente el funcionamiento de las placas Arduino. Ahora con ese conocimiento, se pueden desarrollar todo tipo de proyectos, limitados solamente a la imaginación y a los componentes con los que uno cuente, incluso se puede mejorar este mismo circuito creado, tan solo cambiando el potenciómetro por un sensor que mida el pulso real. Arduino es nada más una herramienta más, que nos permite introducirnos al mundo de microcontroladores y estudiar la relación entre software y hardware, ademas de ser una plataforma versátil para la experimentación en el campo de la ingeniería médica, permitiendo la adquisición, interpretación y control de señales que simulan procesos fisiológicos. En conclusión, Arduino no solo facilita el aprendizaje de conceptos fundamentales de la electrónica y de la programación, también ayuda a potenciar la capacidad de diseñar y prototipar soluciones, mejorando el pensamiento abstracto de estudiantes, ingenieros o de interesados en el área.

Autorización y derechos de distribución

Este archivo cumple la función de guía para el taller de .ªrduino en Acción: Simulador de Señales Fisiológicasrealizado para el Congreso Anual de Ingeniería Médica por parte de la Sociedad de Robótica y Automatización de la Universidad de Concepción (RAS IEEE UdeC).

- Los talleristas de esta instancia fueron Antonia Morales, Gonzalo Quilodrán, Cristian Saavedra y Álvaro Novoa.
- El documento fue realizado por Gonzalo Quilodrán.
- El taller fue llevado a cabo el 14 de Nobiembre, 2025.

Se proibe la distribución con fines monetarios de este documento o del código relacionado al taller si previa consulta y notificación al RAS IEEE UdeC. Se permite el uso personal de los recursos.

Referencias

[1] Arduino. (s.f.). *Arduino — Home*. Recuperado el 7 de noviembre de 2025, de https://www.arduino.cc/

[2] Bricogeek. (s.f.). *Guía de modelos Arduino y sus características — Arduino UNO. Lab Bricogeek*. Recuperado el 7 de noviembre de 2025, de https://lab.bricogeek.com/tutorial/guia-de-modelos-arduino-y-sus-caracteristicas/arduino-uno

[3] Autodesk. (s.f.). *Tinkercad Dashboard*. Recuperado el 2 de octubre de 2025, de https://www.tinkercad.com/dashboard/des