Internet de las Cosas: Unidad 3

Sensores y actuadores

Dr. Rodrigo López Farías.

Ciclo escolar Ene-Junio 2020

Instituto Tecnológico de Querétaro
Ingeniería en Sistemas Computacionales

Índice

Programación de Entrada y Salida de Propósito General de la tarjeta de desarrollo

Estructura Arduino

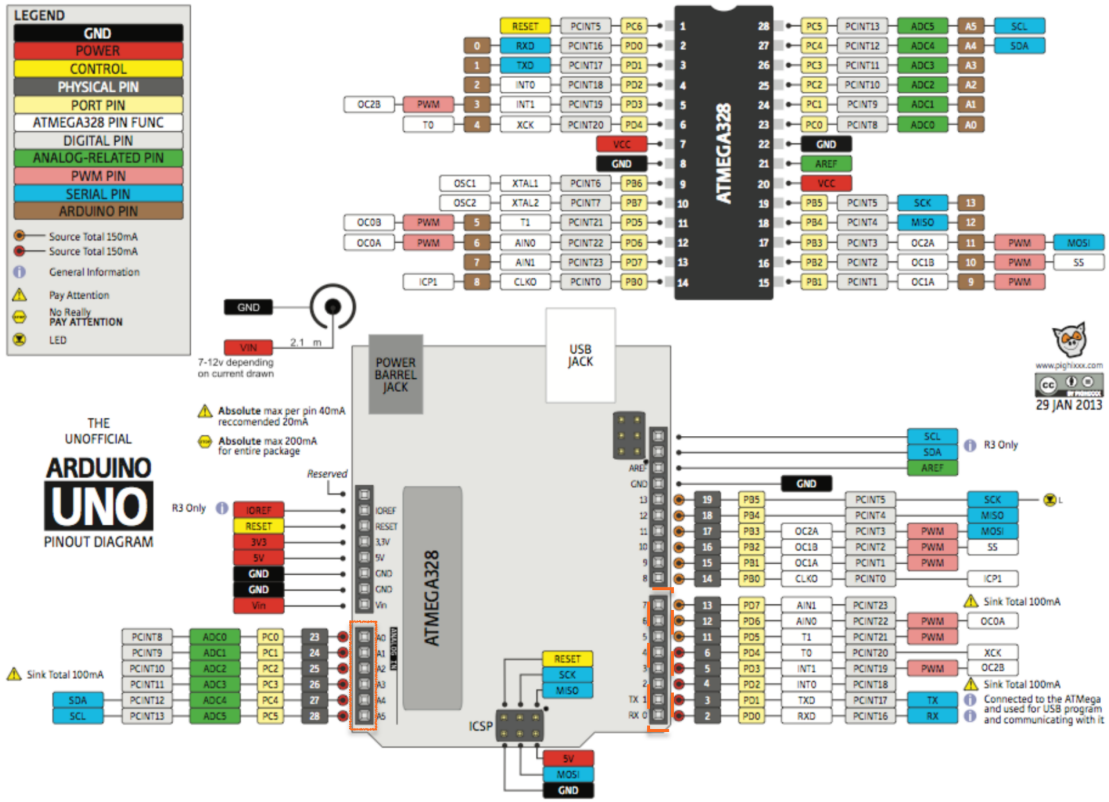


Figure 1: The Unofficial ARDUINO UNO Pinout Diagram [1]

[1] http://marcusjenkins.com/arduino-pinout-diagrams/

Pines de entrada y salida

Precauciones básicas

- La tarjeta tiene 14 pines de entrada y salida digitales
- Tiene 6 pines analógicos para lecturas análogas que tienen un rango definido.
- Las entradas y salidas funcionan con un voltaje de 0V a 5V.
- Máxima corriente por pin son 40mA.
- Arduino se alimenta de tres maneras: por el USB (5V), pines de alimentación(5V) y por el Jack (alimentación entre 7V y 12V).
- Se recomienda usar el USB solo para cargar programas.
- La fuente recomendada debe ser máximo 12 Volts corriente directa. La corriente puede variar ya que el Arduino solo consume la corriente necesaria.

Precauciones básicas para uso de los pines ES

- Los pines soportan una corriente máxima de 40mA de entrada y salida a 5Volts.
- Controlar la corriente requerida por los dispositivos conectados a la placa.
- Recordando Ley de Ohm

$$V = R \times I$$

Precauciones básicas para uso de los pines ES

Ejercicio:

Qué debo hacer si quiero conectar un componente (led) que utiliza a lo mas 20mA (0.02A) de corriente y la placa Arduino provee 5V?
Respuest: utilizar una resistencia de al menos:

$$\frac{5V}{0.02A} <= R$$

Solución: despejando la resistencia.

Al determinar el valor de la resistencia, identificarla utilizando el código de colores [2]

Se aconseja utilizar en este caso una resistencia mas grande para para no hacer trabajar la placa de Arduino al límite. e.g., emplear 5mA.

[2] https://www.inventable.eu/paginas/ResCalculatorSp/ResCalculatorSp.html

Precauciones básicas para uso de los pines ES

- Utilizar una fuente de voltaje externa (9V-12V) para poder proveer de niveles de corriente adecuados a los componentes conectados.
- Aislar con carcasa para evitar el contacto con líquidos y estática.
- Sensores que requieran de mas energía para funcionar, se requiere que se alimenten de manera independiente.
- Utilizar multímetro para verificar voltaje y corriente.

Características de lo Pines digitales de Arduino Uno

- Entrada - Salida Digital: pulso alto 1 es sobrepasar el umbral de 3V y pulso bajo 0 a partir de 2V.
- Pines digitales se definen como de entrada o salida con la función pinMode(*pinNumber,INPUT xor OUTPUT*) ejemplo para definir una salida:
  - pinMode(2,OUTPUT)
- Ejemplo para escribir en un pin de salida utilizando la función pinWrite:
  - pinWrite(3,[HIGH LOW])

  Donde HIGH: 1 y LOW : 0

Desarrollo de programas para actuadores. (Básico)

Ventana principal Arduino

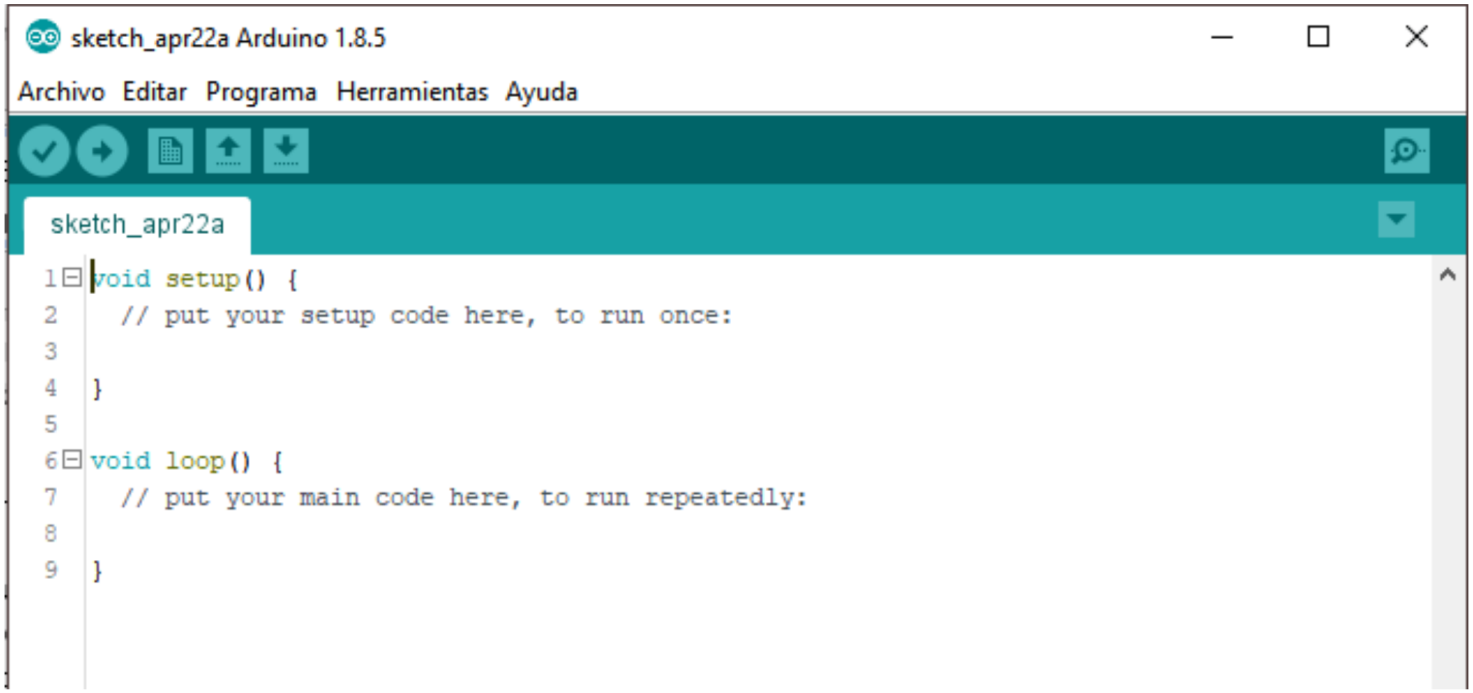


Figure 2: Ventana principal del IDE de Arduino

Código Ejemplo, programación pin de salida

```
[tabsize=4]


#define LED 2 //Damos el alias a nuestro pin 2.
void setup(){
pinMode(LED,OUTPUT); //Definimos el pin LED como salida.
}

void loop(){ //Definimos nuestra secuencia.
digitalWrite(LED,HIGH); //Mandamos un ALTO al LED.
delay(1000); //Tiempo en que permanece el LED prendido \un segundo".  digitalWrite(LED,LOW); //Mandamos u
delay(1000); //Tiempo en que permanece el LED apagado \un segundo".
}
```

Desarrollo de programas para sensores y actuadores (Básico)

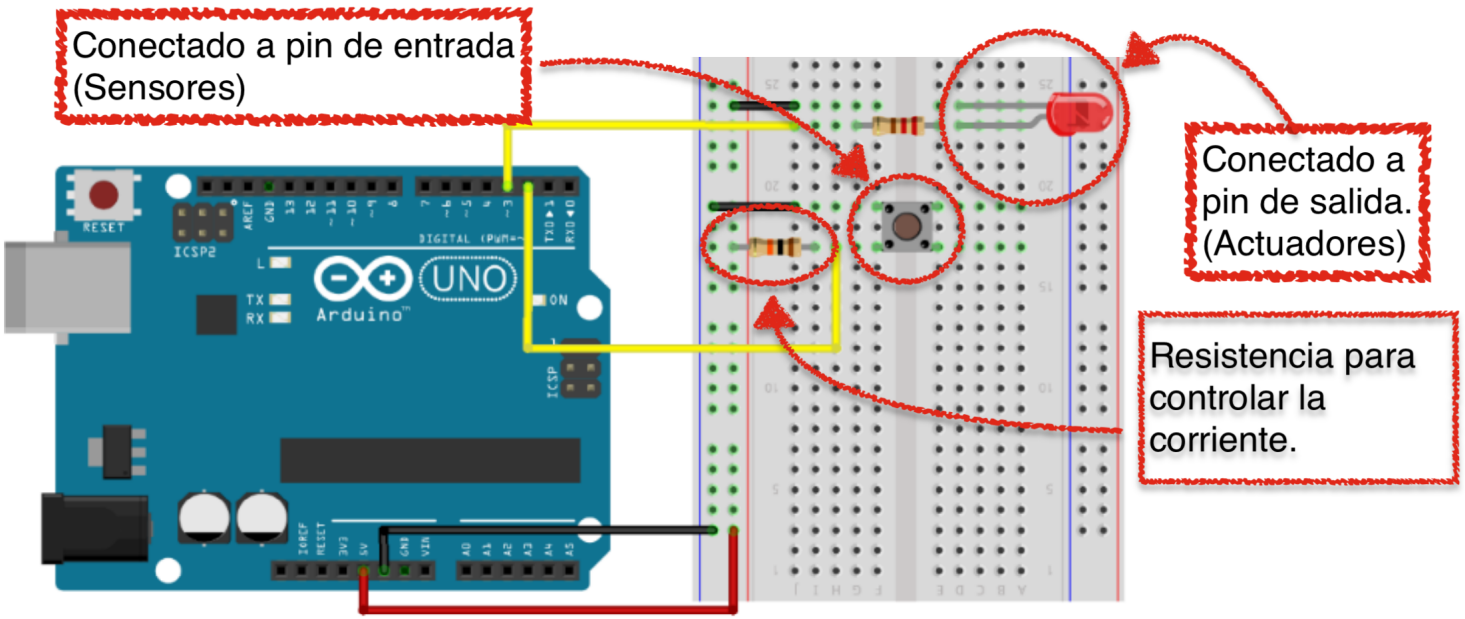


Figure 3: Esquema Digital Básico pines de salida

Código Ejemplo, programación pin de entrada y salida

```
#define LED 2  //Damos el alias a nuestro pin 2.
#define Push 3 //Damos el alias a nuestro pin 3.
byte Push_lee = 0;
void setup(){
        pinMode(LED,OUTPUT); //Definimos el pin LED como salida.
        pinMode(Push,INPUT); //Definimos el pin Push como entrada.
}

//Definimos el algoritmo principal
void loop(){
                Push_lee = digitalRead(Push); //Almacenamos en la variable la lectura del       pulsador.
        if(Push_lee == 1){                      //Cond. que si tenemos el valor de 1 entra en el.
                digitalWrite(LED,HIGH);
        }else{
                digitalWrite(LED,LOW); //Mandamos un BAJO al LED.
        }
}
```

Desarrollo de programas para sensores y actuadores analógicos (Básico) y Desarrollo de programas para adquisición de datos

Sensores Analógicos

La señal analógica tiene Valores continuos en un rango entre 0V y 5V.

Esta señal se convierte a señal digital por medio de un transductor.

Arduino UNO tiene 6 puertos analógicos del A0 al A5

Estos puertos son de entrada analógica o entrada y salida digital.

El firmware (bootloader) de Arduino tiene una resolución por defecto de 10 bits $2^{10} = 1024$ valores. (de 0 a 255)

Se puede cambiar el voltaje máximo de referencia con el pin AREF.

Para poder trabajar con los valores entrada de la señal analógica con el microcontrolador, es necesario convertirla primero una señal digital por medio de esta fórmula de escalamiento.

$$V_{dig} = \frac{V(2^n - 1)}{V_{ref}}$$

Donde: V es el voltaje analógico de entrada. $2^n - 1$ es el valor entero positivo máximo del convertidor y V_{ref} es el valor de voltaje (máximo) de referencia. Esta operación Arduino la hace internamente.

Pregunta: Cómo utilizar la fórmula para aplicar un nivel de voltaje desde la tarjeta Arduino uno a un pin de salida?

Código de ejemplo lectura de un Potenciómetro desde un pin analógico.

```
#define POT A0
int LeePot;
void setup() {
        \\ velocidad de transmisión en baudios (bits/seg)
  Serial.begin(9600);
  pinMode(POT,INPUT);
}
void loop() {
  LeePot = analogRead(POT);
  Serial.println(LeePot);
  delay(50);
}
```

Página 49.

Actuadores analógicos.

Arduino funciona con un microcontrolador que es de naturaleza digital. Lo que se hace es representar esa señal analógica con información digital usando una técnica llamada Modulación por Ancho de Pulso (*Pulse Width Modulation PWM*) la cual:

- Una señal analógica se representa con una señal cuadrada que adquiere solo valores Alto o Bajo.
- El porcentaje de la duración de el valor en alto da lugar al ancho de pulso (ciclo de trabajo[3]) que representa un valor en la señal analógica.
- Tiene resolución de 8 bits (2^8)

Arduino Uno cuenta con 6 pines en total que pueden trabajar con PWM. [4]

[3] Es la fraccion de un periodo el cual la senal esta activa

[4] Ver pag 55 del curso Arduino Básico de Saenz Flores.

Ejemplo: Programa escritura analógica en un pin.

Para escribir con estos pines usamos la función analogWrite([pin/alias],[valor/variable]);

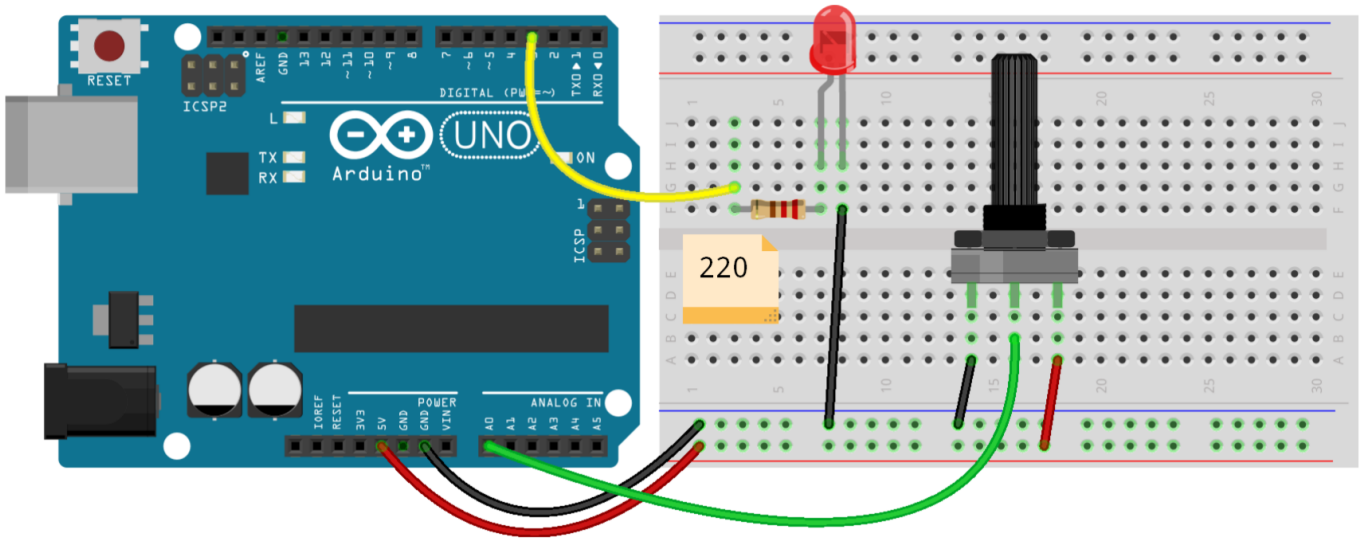
```
#define LED 3
void setup() {
        #Definimos el pin LED como salida
        pinMode(LED,OUTPUT);
}

#Programa principal dentro del loop
void loop() {

  for(int i = 1; i <= 255; i++){
    analogWrite(LED,i);
    delay(10);
    }
    delay(1000);
  for(int i = 255; i >= 1; i--){
    analogWrite(LED,i);
    delay(10);
    }
  delay(1000);
}
```

Si esta conectado un led al pin LED a la placa de Arduino. Que es lo que hace este programa.

Tarea: Práctica 2



Desarrollo de programas para actuadores

Estimación de frecuencias de muestreo

Frecuencia de Adquisición de datos con Arduino

Tiempo de muestreo

Definición

Es el tiempo que transcurre entre dos mediciones consecutivas y se expresa en frecuencia (Hz)

Es muy importante conocer:

- La frecuencia en la que un circuito digital lee los datos.
- La frecuencia en la que el sensor registra una medición.

Estimación de frecuencia de la tarjeta de desarrollo [5]

```
void setup(){
        Serial.begin(9600);
        muestreo();
}

void loop(){
}

void muestreo(){

        unsigned long time1=0;
        unsigned long time=0;
        Serial.println("************************");
        Serial.println("ENSAYO TIEMPO DE MUESTRO:");
        Serial.println("************************");
        for(byte i =0; i<4; i++){
        time1=micros();
        int A=analogRead(A0);
        time=micros()-time1;
        Serial.print(" Muestra: ");
        Serial.print(i+1);
        Serial.print(" Tiempo: ");
        (time);
        }

}
```

[5] https://booleanbite.com/web/adquisicion-de-datos-con-arduino-i-tiempo-de-muestreo-y-resolucion/

Estimación de frecuencia de muestreo teórico

- Teorema de muestreo de Nyquist-Shanon para una señal con banda limitada. Dice que para poder reconstruir adecuadamente esta señal, debemos de muestrear con al menos el doble de la frecuencia de la señal analógica.[6]
- Estimación de tiempo de retardo para señales no lineales y caóticas utilizando la fórmula de la información mutua [7]. (Método general)

[6] bit.ly/tnyqsh
[7] https://bit.ly/3amINtx

Estimación de frecuencia de muestreo teórico

- Teorema de muestreo de Nyquist-Shanon. Dice que para poder reconstruir adecuadamente esta señal, debemos de muestrear con al menos el doble de la frecuencia de la señal analógica.[8]
- Estimación de tiempo del tiempo retardo (usado como frecuencia de referecia) para señales no lineales y caóticas utilizando la fórmula de la información mutua [9]. (Método general).

[8] bit.ly/tnyqsh
[9] https://bit.ly/3amINtx

Estimación de frecuencia de muestreo teórico

Nyquist-Shanon nos dice que si conocemos la frecuencia de una señal tomemos muestras al doble de la frecuencia.

Si no la conocemos, podemos recurrir a calcular la fórmula de información mutua para mediciones en diferentes intervalos.

Estimación de frecuencia de muestreo teórico

$$I_\epsilon = \sum_{i,j} p_{i,j}(\tau) \ln p_{i,j} - 2 \sum_{i} p_i \ln p_i$$

$p_{i,j}$ es la probabilidad de transición de una medición $s(t)$ esté contenida en en intervalo i a uno j. p_i: es la probabilidad de observar el estado i.

Para valores pequeños de τ se esperan grandes valores de I_ϵ indicando poca información.

Para determinar el mejor valor, se calcula I_ϵ para cada retardo τ. El primer mínimo nos indica que es el retardo óptimo que nos da la mayor información de la señal.