**Taller de Microcontroladores 2019**

**Ingeniería de Sistemas**

# **Práctica I:** Entradas y salidas digitales

Al Arduino lo podemos ver como un conjunto de pines o puertos de entrada/salida y conjunto de periféricos y una ALU que procesa los datos. Los pines de entrada sirven para "escuchar" y capturar información del exterior, ejemplo: pulsadores, sensores, lectoras, etc; la ALU sirve para procesar el programa cargado y finalmente los pines de salida sirven para enviar información hacia el exterior.
Al hablar de Entradas o Salidas digitales se entiende que tanto la información que escuchan como las respuestas pueden tener solamente dos estados: +5 voltios o 0 voltios.
En esta lógica las señales digitales pueden manejar los siguientes valores o estados:

| Estado digital | 5 voltios | 0 voltios |
|---|---|---|
| Opción 1 | HIGH | LOW |
| Opción 2 | 1 | 0 |
| Opción 3 | TRUE | FALSE |

Todo programa en Arduino debe tener dos funciones básicas: setup() y loop().
La sintaxis (forma de escribir) es la siguiente:

```
void setup(){
}
void loop(){
}
```

La función setup() sirve para definir el comportamiento inicial de la placa Arduino, es decir cuando la a una fuente de alimentación (pila, batería). Algunos ejemplos de uso de la función setup() son: Que pines voy a utilizar, cuáles serán entradas y cuáles salidas, iniciar alguna variable, etc
La función loop() sirve para definir todas las tareas que Arduino ejecutará repetidamente mientras esté conectada. Por ejemplo: Leer el valor de las entradas, escribir en los pines de salida, enviar alertas, emitir sonidos, enviar mensajes por el puerto serial, etc.

**Ejercicios**

1) Implementar un sketch que permita blinkear un led cada un segundo.
2) Modifique el sketch anterior de manera tal que permita realizar un blink del tipo CountDown entre 5 y 0 segundos:

3) Implementar un sketch que "copie" el estado de un pulsador. ¿Qué sucede si no hay pull-up?

4) Modifique el sketch anterior para que cada evento del pulsador provoque un cambio en el led. ¿Logra el comportamiento esperado? ¿Cómo lo solucionaría?

5) Modifique el sketch del ejercicio 3) para que el evento se produzca ante una interrupción externa.

6) Implemente un segundero binario de 5 bits de manera tal de poder visualizar su salida en 5 leds.

7) Implemente un contador unario de longitud 5 que permita contar eventos producidos en un pulsador.

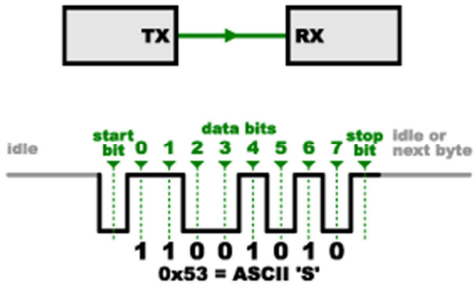
8) Modifique el ejercicio anterior de manera tal de implementar un dado digital

9) (con consola) implemente un medidor de frecuencia. Modifique el sketch de manera tal de poder medir además el ciclo útil

# ARDUINO Y EL PUERTO DE SERIE

Prácticamente todas las placas Arduino disponen al menos de una unidad UART que operan a nivel TTL 0V / 5V. Por su parte, Arduino Mega disponen de 4 unidades UART TTL 0V / 5V. Debido a que los puertos serie están físicamente unidos a distintos pines de la placa Arduino, no podemos usar como entradas o salidas digitales los pines asociados con ese puerto de serie en uso.

En Arduino UNO los pines empleados son 0 (RX) y 1 (TX). En el caso de Arduino Mega el puerto de serie 1 está conectado a los pines 0 (RX) y 1 (TX), el puerto de serie 1 a los pines 19 (RX) y 18 (TX) el puerto de serie 2 a los pines 17 (RX) y 16 (TX), y el puerto serie 3 a los pines 15 (RX) y 14 (TX).




Muchos modelos de placas Arduino disponen de un conector USB o Micro USB conectado a uno de los puertos de serie, lo que simplifica el proceso de conexión con una PC.

**Material Multimedia (youtube):** Arduino Proyecto 2 - Hola Mundo - Entendiendo la Comunicacion Serial Arduino - PC

**Ejercicio 1.** Realice la función ECO, para esto se deberá enviar un valor (ASCII) desde el teclado y el arduino debe retornar dicho valor.

- ➔ Pista 1: Configurar la consola serie del IDE del arduino a una velocidad de 38400 baudios.
- ➔ Pista 2: Leer qué funcionalidad brinda la plataforma arduino para el uso de la comunicación serie. https://www.arduino.cc/en/Reference/Serial

**Ejercicio 2.** Escriba un sketch que permita ingresar los siguientes comandos para encender o apagar el led 13. Luego el arduino deberá contestar “Led 13 Encendido!” o “Led 13 apagado!” según corresponda.

- Comando para encender: “ON:”
- Comando para apagar: “OFF:”

- ➔ Pista 1: notar que los comandos terminan con un símbolo especial.

**Ejercicio 3.** Modificar el sketch anterior para que se pueda seleccionar los leds que se quieren encender o apagar

- Comando para encender: “ON:1,12,3;”
- Comando para apagar: “OFF:2,1,9;”

**Ejercicio 4.** Crear un sketch que permita decodificar el valor binario que se está introduciendo a través de 5 pines configurados como entrada, el valor debe ser reportado a la PC solo cuando cambie con respecto a su estado anterior.

**Ejercicio 5.** Crear un sketch que utilice al Arduino como una calculadora y permita sumar y/o restar números.

- Posible comando: “=5+4;”
- Posible más complejo: “=5+4-3+8-1;”

**Manejando valores analógicos con Arduino**

**Ejercicio 1.** Informar al PC del estado de angular de un potenciómetro, el mismo será configurado como divisor resistivo (por la cátedra).

**Ejercicio 2.** Tomar el estado de angular de un potenciómetro y transmitirlo por alguno de los puertos afines como pulso PWM que active un led.

**Ejercicio 3.** Tomar el estado de angular de un potenciómetro y usarlo como activador de un motor tipo servo (con movimiento angular de 180 grados).