Universidad Intercontinental de la Empresa

Tecnologías de Automatización y Robotización para Empresas

IoT

Placa Arduino Uno

Práctica de Laboratorio 02

Entradas Digitales

y

Comunicación Serial

Profesor Eladio Dapena Gonzalez

Curso 2024-2025

Práctica 02: Entradas Digitales y Comunicación Serial.

1 Introducción

El objetivo principal de esta práctica es conocer el funcionamiento de las entradas digitales en Arduino.

Para ello se utiliza un pulsador normalmente abierto.

Básicamente, cuando se trata de entradas digitales, las consideraciones que se deben tomar son las siguientes:

Declarar el pin como entrada digital y utilizar correctamente la función de lectura.

2 Objetivos

a) **Configurar Pines de la placa Arduino Uno como entradas digitales**

b) **Uso de variables, constantes y funciones para lectura de pines digitales.**

c) **Comunicación serie, inicialización y escritura.**

2.1 Materiales Requeridos

- 1 Arduino UNO
- 1 ProtoBoard
- 5 LED
- 5 resistencia de 220
- 2 resistencia de 1 K
- 2 pulsador normalmente abierto
- Cables para conexiones
- Entorno de desarrollo IDE de Arduino.

3 Marco teórico

3.1 Pulsador / Interruptor

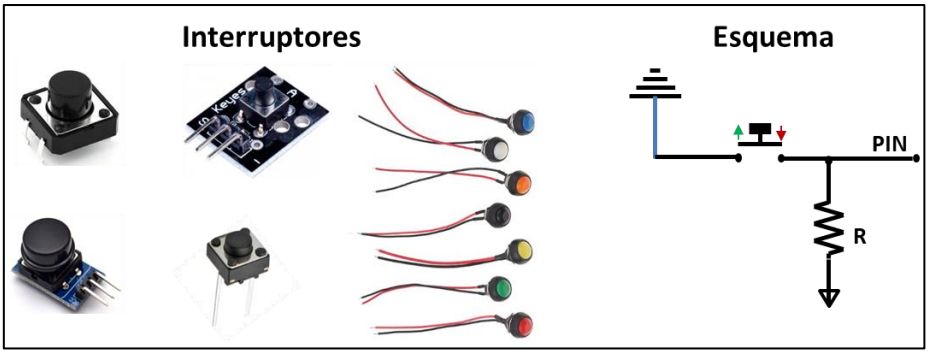
Es un dispositivo que tiene dos estados: el primero, permite el paso de corriente; y el segundo, no lo permite.

Existen dos tipos de pulsadores:

- Normalmente Abierto (NO).
- Normalmente Cerrado (NC).

Como su nombre lo indica, el pulsador normalmente abierto no tiene continuidad entre sus terminales cuando está en estado de reposo, siendo el caso contrario del pulsador normalmente cerrado.

Algunos modelos de pulsadores se pueden apreciar en la siguiente imagen:



3.2 Funciones.

Función	Descripción
pinMode(pin, Modo)	Función que se suele incluir en la función setup() del Bloque de Inicialización que permite configurar pines en alguno de los modos de entrada **INPUT** o salida **OUTPUT**. Los pines de Arduino funcionan por defecto como entradas, de forma que no es necesario definirlos explícitamente como entradas. El parámetro **pin** puede ser un valor literal, una variable o una constante en el rango de valores [0 - 13] para la placa Arduino Uno. Ejemplos: pinMode(2, INPUT); pinMode(3, OUTPUT);
digitalRead(pin)	Lee el valor que tienen un pin digital específico. La función retorna un valor **HIGH** o **LOW**. Ejemplo: valor = digitalRead(2);
Serialbegin(valor)	Inicia el puerto de comunicación serie y establece la velocidad de transmisión de datos en bits por segundo (baud) para la transmisión de datos en serie. Para comunicarte con el ordenador, utiliza como velocidad de trasmisión (**valo**r) alguna de estas velocidades: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 o 115200.
Serial.end()	Deshabilita la comunicación serial.
Serial.available()	Obtiene la cantidad de bytes (caracteres) disponibles para leer desde el puerto serial.
Serial.read()	Lee datos seriales entrantes.
Serial.print()	Imprime datos en el puerto serial como texto ASCII legible para humanos. Este comando puede adoptar muchas formas. Los números se imprimen utilizando un carácter ASCII para cada dígito.
Serial.println()	Imprime datos en el puerto serie como texto ASCII legible por humanos seguido de un carácter de retorno de carro (ASCII 13 o '\r') y un carácter de nueva línea (ASCII 10 o '\n').

4 Ejercicio 01. Encender Led con Pulsador (1 punto)

El objetivo de este ejercicio es programar el encendido de un LED conectado al **Pin 10** cuando se mantenga pulsado un botón cuyo estado se lee desde el **Pin 2**.

a) Leer el valor de la entrada digital

b) Conocer el funcionamiento del pulsador

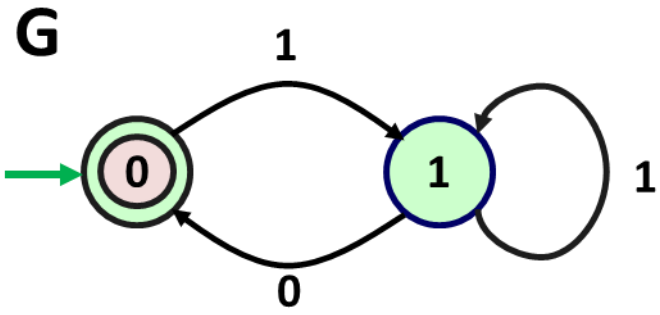
c) Conectar correctamente el pulsador

4.1 Componentes

Led y Pulsador.

4.2 Modelo DESs

Representación Gráfica



Conjunto de Estados X = { 0, 0 } → 0: Led Apagado 1: Led Encendido

Conjunto de Eventos E = { 0, 1} → 0: Pulsador abierto 1: Pulsador Cerrado

Función de Transición de Estados $f(X, e) \rightarrow X$

$f(0,1) \rightarrow 1$

$f(1,1) \rightarrow 1$

$f(1,0) \rightarrow 0$

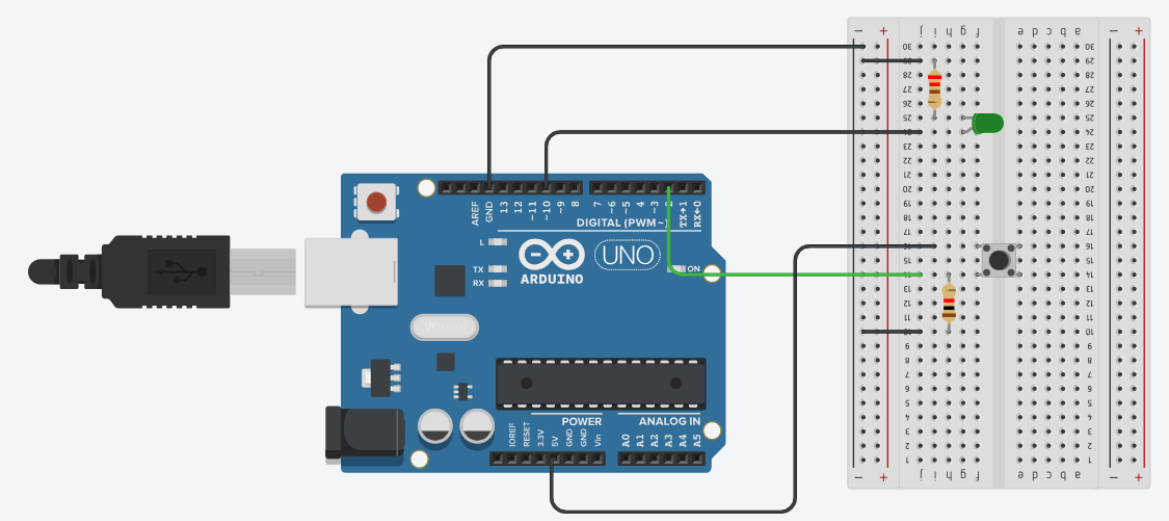
Estado Inicial $X_0 = 0$

Conjunto de Estado Marcados $X_m=\{0\}$

4.3 Conectar los componentes como se muestra en el siguiente esquema:

Led Pin 10

Pulsador Pin 2



Nota: **Verifique** sus conexiones con el **técnico del laboratorio** antes de continuar con la actividad.

4.4 Programa.

Abrir en su entorno de desarrollo (IDE Arduino) el programa **IoTP02E01.ino** descargado desde el material de la práctica para el ejercicio 01.

4.5 Prueba.

Cargar el programa en la placa Arduino para su ejecución.

4.6 Validar Ejercicio.

Mostrar al técnico del laboratorio el funcionamiento del modelo para el **registro en la plantilla de valoración** de la práctica**.**

5 Ejercicio 02. Incorporar segundo botón de entrada. (2 puntos)

Agregue un segundo botón con entrada desde el pin 5 al ejercicio 1 (Ver esquema).

El comportamiento ahora será el siguiente:

- Si se pulsa una vez el botón de encendido conectado a la placa Arduino por medio del **Pin 2** el Led_01 conectado al Pin 10 permanecerá encendido.
- Cuando se pulse el botón de apagado conectado a la placa Arduino por medio del **Pin 5** el Led_01 conectado al **Pin 10** se apaga hasta que se pulse el botón de encendido.

Modelo a Eventos Discretos.

Componentes: Pulsador 1 (P1), Pulsador 2 (P2), Led 01 (L1)

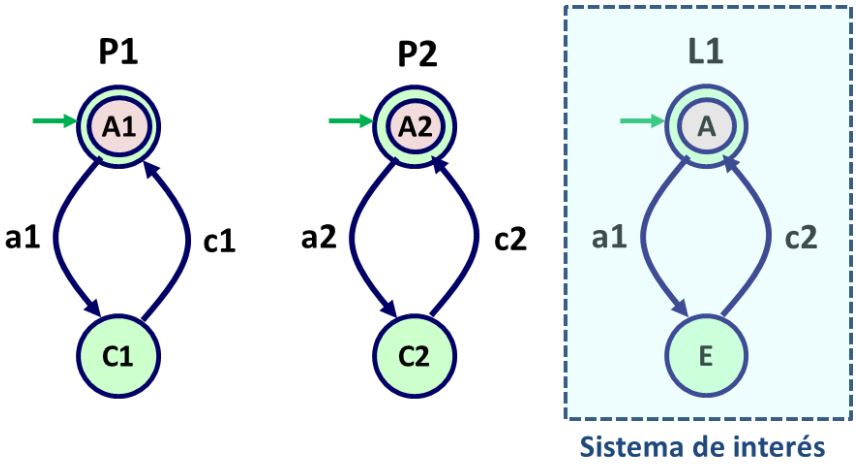
Estados:

P1 ={ Abierto (A1), Cerrado (C1)} Eventos { a1, c1 }

P2 ={ Abierto (A2), Cerrado (C2)} Eventos { a2, c2 }

L1 ={ Apagado (A), Encendido (E) } No Tiene eventos propios.

Autómatas.



Definición del autómata

X ***{ A , E }***

E ***{ a1 , a2 }***

f: X x E → X ***f(A,a1) = E*** ***f(E,a2) = A***

X_0 ***A***

$Xm \subseteq X$ ***A***

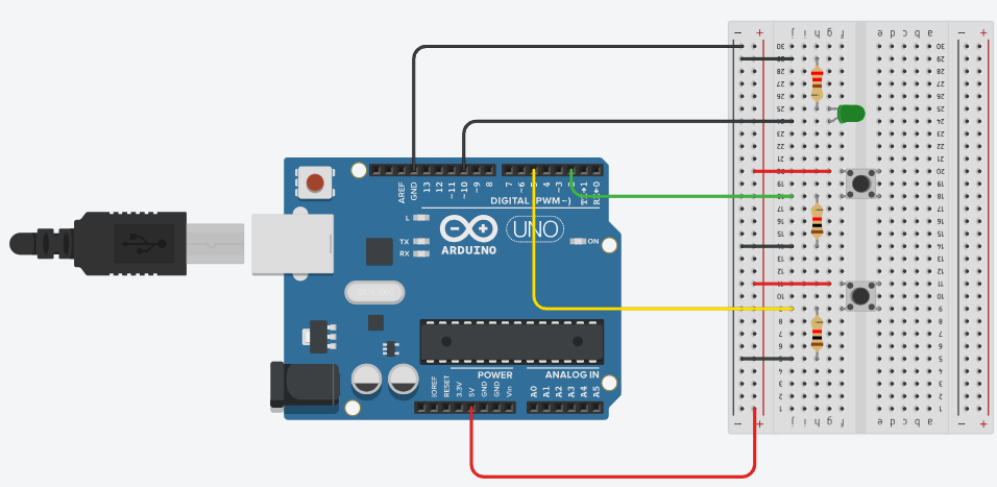
$L = \{ \varepsilon, a1, a1a2, a1a2a1, a1a2a1a2, a1a2a1a2a1, \ldots \}$

$Lm = \{ \varepsilon, a1a2, a1a2a1a2, a1a2a1a2a1a2, \ldots$

En dónde. a1= Pin Pulsador Encendido en **HIGH** **y** a1= Pin Pulsador Apagado en **HIGH**

El sistema sólo procesará los eventos a1 (botón de encendido pulsado) y a2 (botón de apagado pulsado)

5.1 Conectar los componentes y la placa Arduino como se muestra en el siguiente esquema:



- Realice las conexiones correspondientes según el esquema mostrado.

Nota: **Verifique** sus conexiones con el **técnico del laboratorio** antes de continuar con la actividad.

5.2 Programa.

Abrir en su entorno de desarrollo (IDE Arduino) el programa **IoTP02E02.ino** descargado desde el material de la práctica para el ejercicio 02.

5.3 Prueba.

Cargar el programa en la placa Arduino para su ejecución.

5.4 Validar Ejercicio.

Mostrar al técnico del laboratorio el funcionamiento del modelo para el **registro en la plantilla de valoración** de la práctica**.**

6 Ejercicio 03. Comunicación Serie con el ordenador (1 punto)

Agregar instrucciones de comunicación serial al programa del ejercicio anterior manteniendo el mismo esquema de conexiones en la placa Arduino.

6.1 Programa.

Abrir en su entorno de desarrollo (IDE Arduino) el programa **IoTP02E03.ino** descargado desde el material de la práctica para el ejercicio 03.

6.2 Prueba.

Cargar el programa en la placa Arduino para su ejecución.

6.3 Validar Ejercicio.

Mostrar al técnico del laboratorio el funcionamiento del modelo para el **registro en la plantilla de valoración** de la práctica**.**

7 Ejercicio 04. Actividad Práctica Autónoma Presencial (6 pts.).

Construir un esquema con 5 led y dos pulsadores (**+** , **-**) y el correspondiente programa en Arduino para un modelo que se comporte de la siguiente manera:

- **Estado Inicial.**
  Todos los ***Leds*** apagados.
- **Pulsador + .**
  Genera un evento que incrementa en uno el número de ***Leds*** encendidos, siempre que el número de ***Leds*** encendidos sea menor que 5, en otro caso se ignora el evento.
- **Pulsador - .**
  Genera un evento que disminuye en uno el número de leds encendidos, siempre que el número de ***Leds*** encendidos sea mayor que 0, en otro caso se ignora el evento.
- **Comunicación Serial.**
  El programa debe incluir la comunicación serie con la terminal indicando el estado del esquema referido a la cantidad de leds encendidos.

Ejemplo:

Nota: Verifique sus conexiones con el **técnico del laboratorio** antes de continuar con la actividad.

7.1 Prueba.

Cargar el programa en la placa Arduino para su ejecución.

7.2 Validar Ejercicio.

Mostrar al técnico del laboratorio el funcionamiento del modelo para el **registro en la plantilla de valoración** de la práctica**.**

8 Entrega de la Práctica 01

En la tarea AE09 Laboratorio IoT. Práctica 01 de la Unidad II en el apartado EVALUACIONES subir el fichero del Ejercicio 03 (**IoTP01E03.ino**).

La programación debe reflejar el comportamiento de un modelo de un Sistema a Eventos Discretos para la intersección de tráfico (Estados del Sistema, Eventos y Función de Transición de Estados).

Valoración (6 puntos)

- Funciona Correctamente (3 puntos).
- Diseñado como un Sistema a Eventos Discretos (3 puntos)