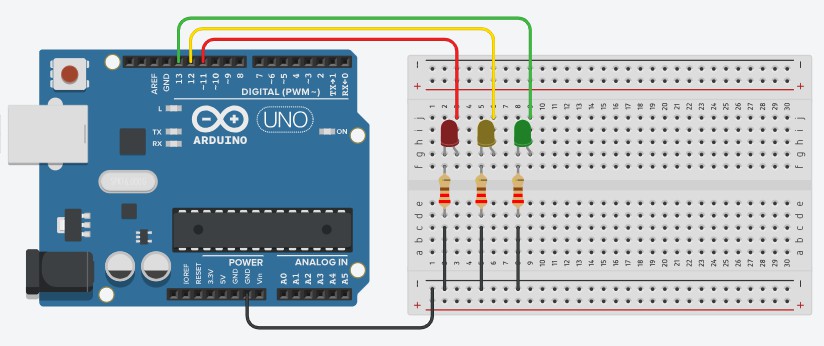
1. IDENTIFICACIÓN DEL ESPACIO ACADÉMICO.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Programa** | Ingeniería de Sistemas | | | |
| **Espacio Académico/curso** | Electiva de programación (Internet of Things) | | **Semestre** | 6 |
| **Área** | N/A | **Grupo** | E | |
| **Tipo de entrega** | Informe de laboratorio | | | |
| **Descripción de la**  **actividad:** | En esta actividad, utilizaremos una placa Arduino para simular el control de 3 LEDs. | | | |
| **Objetivo del laboratorio:** | Relacionar el comportamiento de los leds con los conceptos de un sistema operativo y sus procesos, ilustrando la gestión de múltiples tareas concurrentes, al igual que los semáforos coordinan el flujo de  vehículos para evitar conflictos. | | | |
| **Palabras clave:** | Procesos Concurrentes, Planificación (Scheduling), Rutinas de Interrupción, Sincronización y Exclusión Mutua, algoritmo Round Robin (Asignación de Tiempo Equitativa), Ciclo Continuo, Starvation (Evitar  Inanición), Interrupción al Terminar el Quantum. | | | |
| **Nombres completos grupo de trabajo: (Máx. 3 integrantes)** | * WILMER ALEXANDER BERMUDES MUJANAJINSOY * LUIS ARMANDO OBANDO TREJO | | | |

****

**Ilustración 1.** Montaje de referencia

|  |
| --- |
| 0. Enunciado de la actividad. |
| Elementos de laboratorio:  1 Placa Arduino  3 LEDs (Amarillo, rojo, verde) 3 Resistencias  1 Protoboard  1 Pulsador (2 o 4 pines)  Contruye el montaje propuesto en la **Ilustración 1** y programa el Sketch de Arduino en TinkerCad y en físico a través de la Protoboard, de manera que permita visualizar por consola un menú de opciones y ejecutar cada una de las siguientes acciones:  **Main menú:**  [1]. Turn on Led red [2]. Turn off Led red [3]. Turn on Led yellow [4]. Turn off Led yellow [5]. Turn on Led green [6]. Turn off Led green [7]. Turn on all  [8]. Turn off all  [9]. Intermitence (all)  Adiciona al circuito (En TinkerCad y en físico con la Protoboard) un Pulsador de dos pines en mod**o INPUT\_PULLUP** que permita ejecutar las siguientes acciones en paralelo y sin generar conflictos con la interfaz del Main main gestionado desde la consola.  **Pulsación 1:** Enciende el LED rojo, los demás apagados. **Pulsación 2:** Enciende el LED verde, los demás apagados. **Pulsación 3:** Enciende el LED amarillo, los demás apagados. **Pulsación 4:** Apaga todos los LEDs.  **Pulsación 4:** Enciende todos los LEDs.  **Pulsación 6:** Intermitencia (ciclo infinito).  No hacer uso de **delay()**. Deben manejar el rebote del pulsador con millis() o mediante lógica de software. |
| 1. Diagrama esquemático Tinkercad |
|  |
| 2. Tabla de componentes Tinkercad |
|  |

|  |
| --- |
|  |
| 3. Imagen montaje circuito en Tinkercad |
|  |
| 3. Fotografía de montaje en físico |
|  |
| 4. Código fuente |
| int const LED\_ROJO = 13;  int const LED\_AMARILLO = 12;  int const LED\_VERDE = 11;  int const BTN = 7; // Pulsador en pin 7 (INPUT\_PULLUP)  char opt;  int btnState;  int lastBtnState = HIGH;  unsigned long lastDebounceTime = 0;  unsigned long debounceDelay = 50;  // Control de acciones por pulsador  int accion = 0;  // Variables para intermitencia sin delay  unsigned long prevMillis = 0;  int estadoBlink = LOW;  void setup() {  pinMode(LED\_ROJO, OUTPUT);  pinMode(LED\_AMARILLO, OUTPUT);  pinMode(LED\_VERDE, OUTPUT);  pinMode(BTN, INPUT\_PULLUP);  Serial.begin(9600);  Serial.println(" :: WELCOME :: ");  Serial.println("Press any key to show menu ");  }  void menu() {  Serial.println(" MAIN MENU ");  Serial.println("[1] turn on led red ");  Serial.println("[2] turn off led red ");  Serial.println("[3] turn on led yellow ");  Serial.println("[4] turn off led yellow ");  Serial.println("[5] turn on led green ");  Serial.println("[6] turn off led green ");  Serial.println("[7] turn on all ");  Serial.println("[8] turn off all ");  Serial.println("[9] intermitence ");  Serial.println("Press any option: ");  }  void ejecutarAccion(int a) {  // Apagar todos primero  digitalWrite(LED\_ROJO, LOW);  digitalWrite(LED\_AMARILLO, LOW);  digitalWrite(LED\_VERDE, LOW);  switch (a) {  case 1: // LED rojo  digitalWrite(LED\_ROJO, HIGH);  break;  case 2: // LED verde  digitalWrite(LED\_VERDE, HIGH);  break;  case 3: // LED amarillo  digitalWrite(LED\_AMARILLO, HIGH);  break;  case 4: // Todos apagados  break;  case 5: // Todos encendidos  digitalWrite(LED\_ROJO, HIGH);  digitalWrite(LED\_AMARILLO, HIGH);  digitalWrite(LED\_VERDE, HIGH);  break;  case 6: // Intermitencia sin delay  unsigned long currentMillis = millis();  if (currentMillis - prevMillis >= 500) {  prevMillis = currentMillis;  estadoBlink = !estadoBlink;  digitalWrite(LED\_ROJO, estadoBlink);  digitalWrite(LED\_AMARILLO, estadoBlink);  digitalWrite(LED\_VERDE, estadoBlink);  }  break;  }  }  void loop() {  // =======================  // Gestión Serial (igual que antes)  // =======================  if (Serial.available() > 0) {  menu();  opt = Serial.read();  if (opt == '1') digitalWrite(LED\_ROJO, HIGH);  if (opt == '2') digitalWrite(LED\_ROJO, LOW);  if (opt == '3') digitalWrite(LED\_AMARILLO, HIGH);  if (opt == '4') digitalWrite(LED\_AMARILLO, LOW);  if (opt == '5') digitalWrite(LED\_VERDE, HIGH);  if (opt == '6') digitalWrite(LED\_VERDE, LOW);  if (opt == '7') {  digitalWrite(LED\_ROJO, HIGH);  digitalWrite(LED\_AMARILLO, HIGH);  digitalWrite(LED\_VERDE, HIGH);  }  if (opt == '8') {  digitalWrite(LED\_ROJO, LOW);  digitalWrite(LED\_AMARILLO, LOW);  digitalWrite(LED\_VERDE, LOW);  }  if (opt == '9') {  // Intermitencia con delay (solo si se activa desde Serial)  for (int i = 0; i < 5; i++) {  digitalWrite(LED\_ROJO, HIGH);  digitalWrite(LED\_AMARILLO, HIGH);  digitalWrite(LED\_VERDE, HIGH);  delay(500);  digitalWrite(LED\_ROJO, LOW);  digitalWrite(LED\_AMARILLO, LOW);  digitalWrite(LED\_VERDE, LOW);  delay(500);  }  }  }  // =======================  // Gestión Pulsador (con debounce)  // =======================  int reading = digitalRead(BTN);  if (reading != lastBtnState) {  lastDebounceTime = millis();  }  if ((millis() - lastDebounceTime) > debounceDelay) {  if (reading == LOW && btnState == HIGH) {  // Cambio válido: avanzar a la siguiente acción  accion++;  if (accion > 6) accion = 1; // Ciclo  }  btnState = reading;  }  lastBtnState = reading;  // Ejecutar acción del pulsador  ejecutarAccion(accion);  } |
| 5. Enlace o URL del repositorio GitHub |
| El código del sketch de Arduino creado en TinkerCad debe estar versionado en el repositorio de GitHub. El Repo debe estar configurado comoPúblico. |
| 6. Enlace o URL del laboratorio en TinkerCad |
| https://www.tinkercad.com/things/gPaCb8rnqGC-menu-leds-con-pulsador |
| 7. Enlace video funcionamiento |
| La URL del video en donde se evidencie el funcionamiento del circuito. |

|  |  |
| --- | --- |
| Rúbricas de evaluación  Uso exclusivo del docente | |
| Estética en la implementación del circuito | 0.0 – 5.0 (10%) |
| Cumplimiento de requisitos hardware | 0.0 – 5.0 (40%) |
| Cumplimiento de requisitos software | 0.0 – 5.0 (40%) |
| Refactorización de código | 0.0 – 5.0 (10%) |