III. PUERTOS ENTRADA SALIDA

PUERTOS ENTRADA SALIDA

III. PUERTOS ENTRADA SALIDA

III. PUERTOS ENTRADA SALIDA

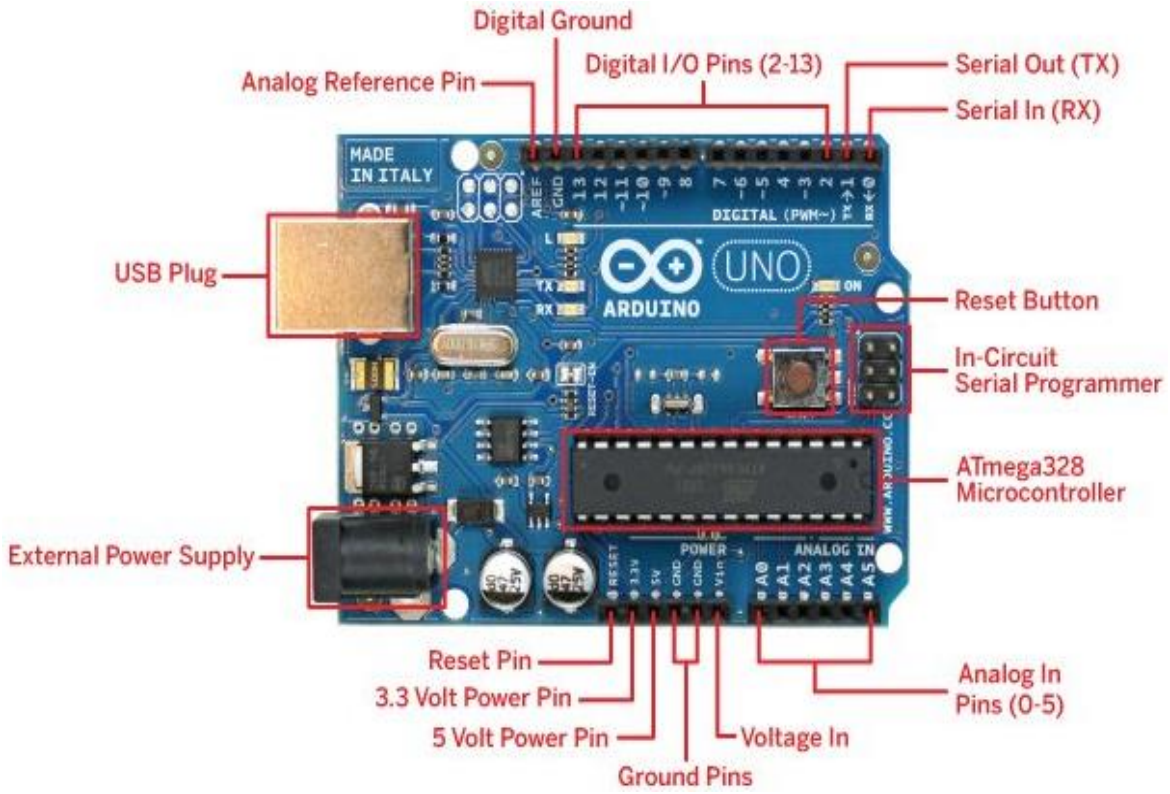
3.1 Puertos

Los pines en la placa Arduino (ATmega8 y ATmega168) poseen tres puertos (El ATmega328(Arduino Uno) usa la misma configuración de pines que el ATmega168):

B (Pines digitales del 8 al 13).

C (Entrada Analógica)

D (Pines Digitales del 0 al 7)



III. PUERTOS ENTRADA SALIDA

3.1.1 Port D

El PORTD mapea los pines digitales del 0 al 7.

DDRD – El registro de configuraci6n del modo de los pines del puerto D – lectura/escritura.

PORTD – Registro de datos del Puerto D – lectura/escritura.

PIND – Registro de pines de entrada – solo lectura.

III. PUERTOS ENTRADA SALIDA

3.1.2 Port B

El PORTB mapea los pines digitales del 8 al 13. Se debe recordar que los bits altos (6 y 7) están mapeados a los pines del cristal de cuarzo y no pueden ser usados.

DDRB – El registro de configuración del modo de los pines del puerto B – lectura/escritura.

PORTB – Registro de datos del puerto D – lectura/escritura.

PINB – Registro de pines de entrada – solo lectura.

III. PUERTOS ENTRADA SALIDA

3.1.3 Port C

El PORTC mapea los pines de entrada analógica del 0 al 5.

DDRC – El registro de configuración del modo de los pines del puerto B – lectura/escritura.

PORTC – Registro de datos del Puerto D – lectura/escritura.

PINC – Registro de pines de entrada – solo lectura.

Cada bit de estos registros corresponde con un solo pin; Ejemplo: El bit menos significativo de los registros DDRB, PORTB, y PINB hace referencia al pin PB0 (pin digital 8).

III. PUERTOS ENTRADA SALIDA

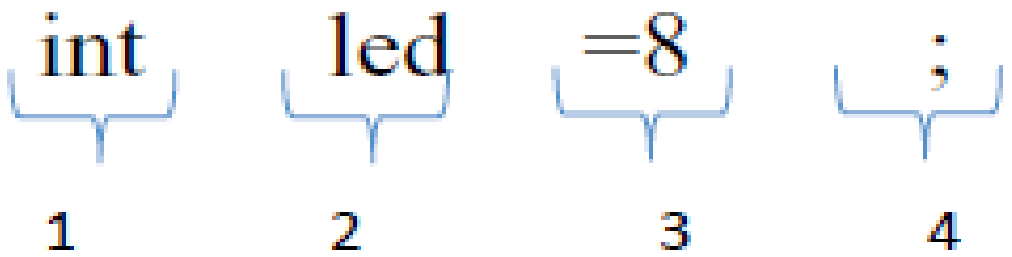
3.2 Como Declara Un Pin

Se lo hace de la siguiente manera:

Para entender el manejo de variables se realizará el mismo programa, pero cambiando el pin de uso con una variable, de esta forma en vez de llamar al número del pin a usar, se llamará al nombre que se le ha asignado a determinado pin, antes de la programación del void setup (), vamos a inicializar una variable de tipo entero (integer), esta variable puede contener números sin decimales hasta un valor máximo de 65535 y un mínimo de -65535, es un valor más que suficiente para nuestro programa; la estructura de programación es siguiente:

III. PUERTOS ENTRADA SALIDA

3.2 Como Declara Un Pin



1 -> Tipo de dato (integer).

2 -> Nombre de la variable.

3 -> Asignación de valor a la variable (opcional).

4 -> Fin de línea de código

III. PUERTOS ENTRADA SALIDA

3.2 Como Declara Un Pin

Utilizar los pines para lo cual se los deben declarar con cualquier nombre de variable el cual se lo debe igualar al número de PIN a utilizar.

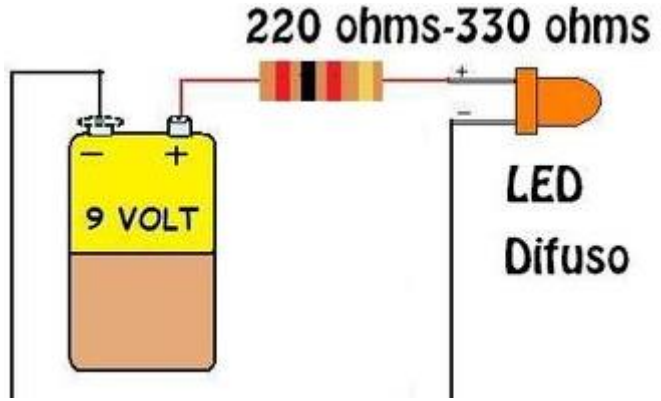
Si se desea declarar 5 pines (desde el digito 0 a digito 5), en este caso el nombre de la variable es irrelevante en consideración con el número de los pines digitales los cuales siempre comenzaran a partir desde el 0 hasta el 5 en este caso.

```
int pin1 = 0;
int pin2 = 0;
int pin3 = 0;
int pin4 = 0;
int pin5 = 0;
int pin6 = 0;
```

III. PUERTOS ENTRADA SALIDA

3.3 Puestos Como Salida:

Las salidas digitales en un programa se refieren a que el Arduino mediante sus pines digitales envía estados lógicos hacia un elemento pasivo (recibe voltaje de otro dispositivo), que por el momento los pines los usaremos para encender y apagar leds, un led soporta hasta un máximo de 3.3v y dependiendo del color resiste cierto amperaje, para ellos se ha estandarizado por un cálculo circuital que es necesario colocar una resistencia de 220 ohmios o 330 ohmios antes del led como indica la figura.



III. PUERTOS ENTRADA SALIDA

3.3 Puestos Como Salida:

void setud (): Sirve para declarar si el pin se va a usar como INPUT (0) o como OUTPUT (1).

Si se desea declarar 7 pines tendríamos que repetir pinMode 7 veces.

```
int pin1 = 0; //Declaracion de pines
int pin2 = 0;
int pin3 = 0;
int pin4 = 0;
int pin5 = 0;
int pin6 = 0;

void setup() {
pinMode(pin1,OUTPUT); //Declaracion de pines como salida
pinMode(pin2,OUTPUT);
pinMode(pin3,OUTPUT);
pinMode(pin4,OUTPUT);
pinMode(pin5,OUTPUT);
pinMode(pin6,OUTPUT);
}
```

III. PUERTOS ENTRADA SALIDA

3.3.1 Como Definir Un Estado

La instrucción o comando es digitalWrite (número de pin [valor entero], tipo [HIGH/LOW]), este comando lo que hace enviar un valor al pin de salida el cual puede ser 0 o 5 voltios. De esta manera se explica cómo es que un pin puede mandar el voltaje necesario para encender un Led.

```
void loop() {
  pinMode(pin1, HIGH); //Declaracion de estados
  pinMode(pin2, LOW);
  pinMode(pin3, LOW);
  pinMode(pin4, HIGH);
  pinMode(pin5, HIGH);
  pinMode(pin6, LOW);
  delay(500);
}
```

III. PUERTOS ENTRADA SALIDA

3.3.1 Como Definir Un Estado

Lo que se hace es encender el pin con el valor HIGH y luego con la función delay(500) se crea un retardo de medio segundo.

Con la función delay() hay que tener mucho cuidado ya que cada vez que se llama detiene la ejecución del bloque void loop(), dejando como en un estado suspendido a Arduino.

III. PUERTOS ENTRADA SALIDA

3.4 Ejercicios De Aplicación Puestos Salida

3.1.4 Encendido De Led’s

El programa realiza el encendido de los leds de forma alternada

III. PUERTOS ENTRADA SALIDA

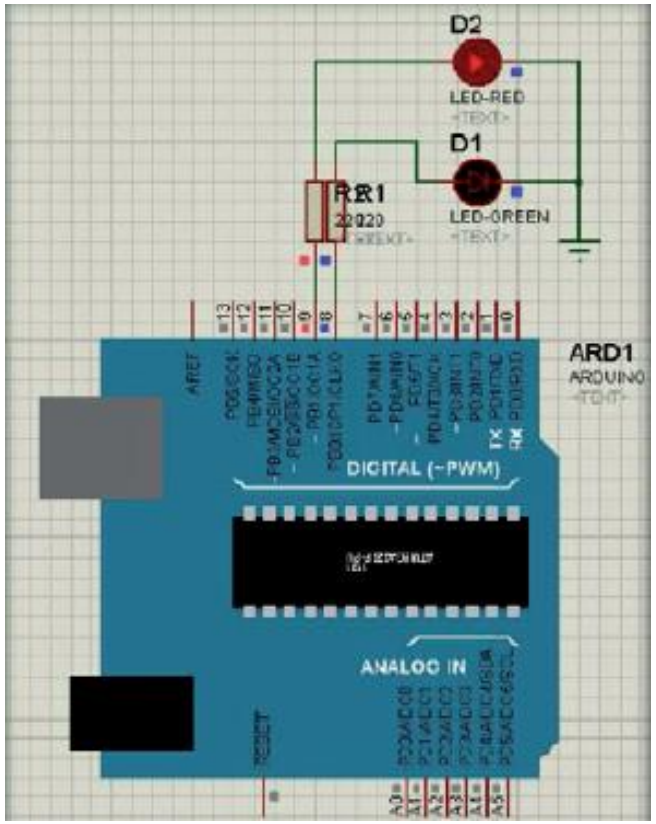
3.4 Ejercicios De Aplicación Puestos Salida

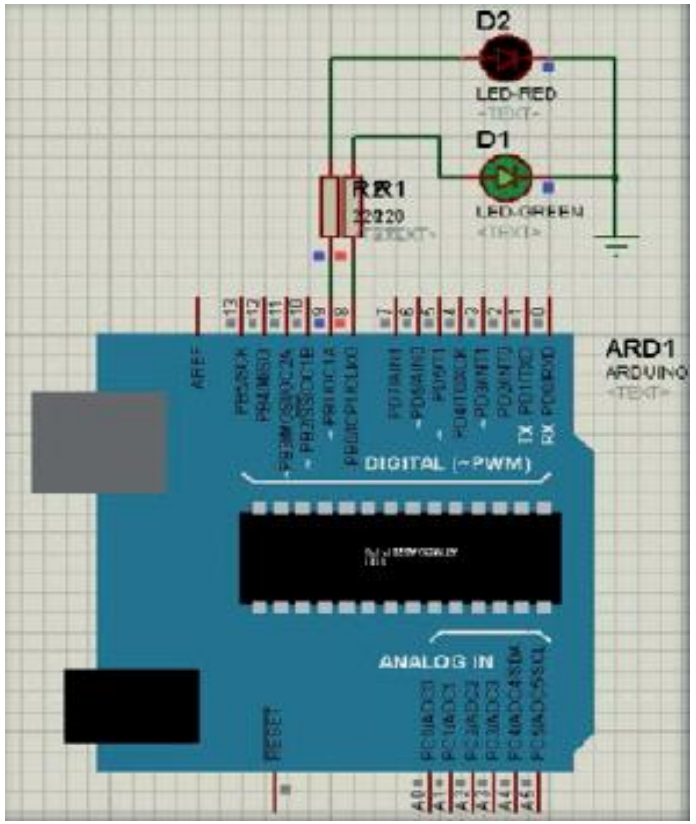
Programa En Arduino

```
int led_rojo = 9; //Variable para led 1
int led_verde = 8; //Variable para led 2

void setup() {
  pinMode(led_rojo, OUTPUT); //Pin decalrado como salida
  pinMode(led_verde, OUTPUT); //Pin declarado como salida
}

void loop() {
  digitalWrite(led_rojo, HIGH); //Encender led 1
  delay(1000); //Delay
  digitalWrite(led_rojo, LOW); //Apagar led 1
  delay(1000); //Delay
  digitalWrite(led_verde, HIGH); //Encender led 2
  delay(1000); //Delay
  digitalWrite(led_verde, LOW); //Apagar led 2
  delay(1000); //Delay
}
```

III. PUERTOS ENTRADA SALIDA

Simulación





III. PUERTOS ENTRADA SALIDA

3.4 Ejercicios De Aplicación Puestos Salida

3.4.2 Juego De Led’s

Programa básico para aprender sobre la declaración de puertos. Consiste en crear un juego de luces.

III. PUERTOS ENTRADA SALIDA

3.4.2 Juego De Led’s

Programa En Arduino

```
int j = o; //Contador 2
int k = 0;
int m = o;
int n = 0;

void setup() {
  for (; i < 7; i++) {
    pinMode(vector_leds[i], OUTPUT); //Declaracion de los pines de salida mediante un ciclo for
  }
}
void loop() {
  juego_luces5();
  juego_luces1();
  juego_luces4(); //Llamar a los distintas clases creadas
  juego_luces2();
  juego_luces3();
}

void juegos_luces1() { //Clase1: Prender todos las leces en secuencia
  for (; k < 2; k++) { //For k
    for (; j < 7; j++) { //For j anidado
      digitalWrite(vector_leds[j], HIGH);
      delay(200);
      digitalWrite(vector_leds[j], LOW);
      delay(200);
    }
    j = 0;
  }
}
```

III. PUERTOS ENTRADA SALIDA

3.4.2 Juego De Led’s

Programa En Arduino

```
void juegos_luces2() { //Clase2: Prender leces azules
  for (; l < 3; l++) { //For l
    for (j = l; j < 7; j++) { //For j anidado
      digitalWrite(vector_leds[j], HIGH);
      delay(175);
      digitalWrite(vector_leds[j], LOW);
      delay(175);
    }
    j = 0;
  }
}

void juegos_luces3() { //Clase3: Prender luces verdes
  for (; m < 5; m++) { //For m
    for (j = 2; j < 5; j = j + 2) { //For j anidado
      digitalWrite(vector_leds[j], HIGH);
      delay(150);
      digitalWrite(vector_leds[j], LOW);
      delay(150);
    }
    j = 0;
  }
}
```

III. PUERTOS ENTRADA SALIDA

3.4.2 Juego De Led’s

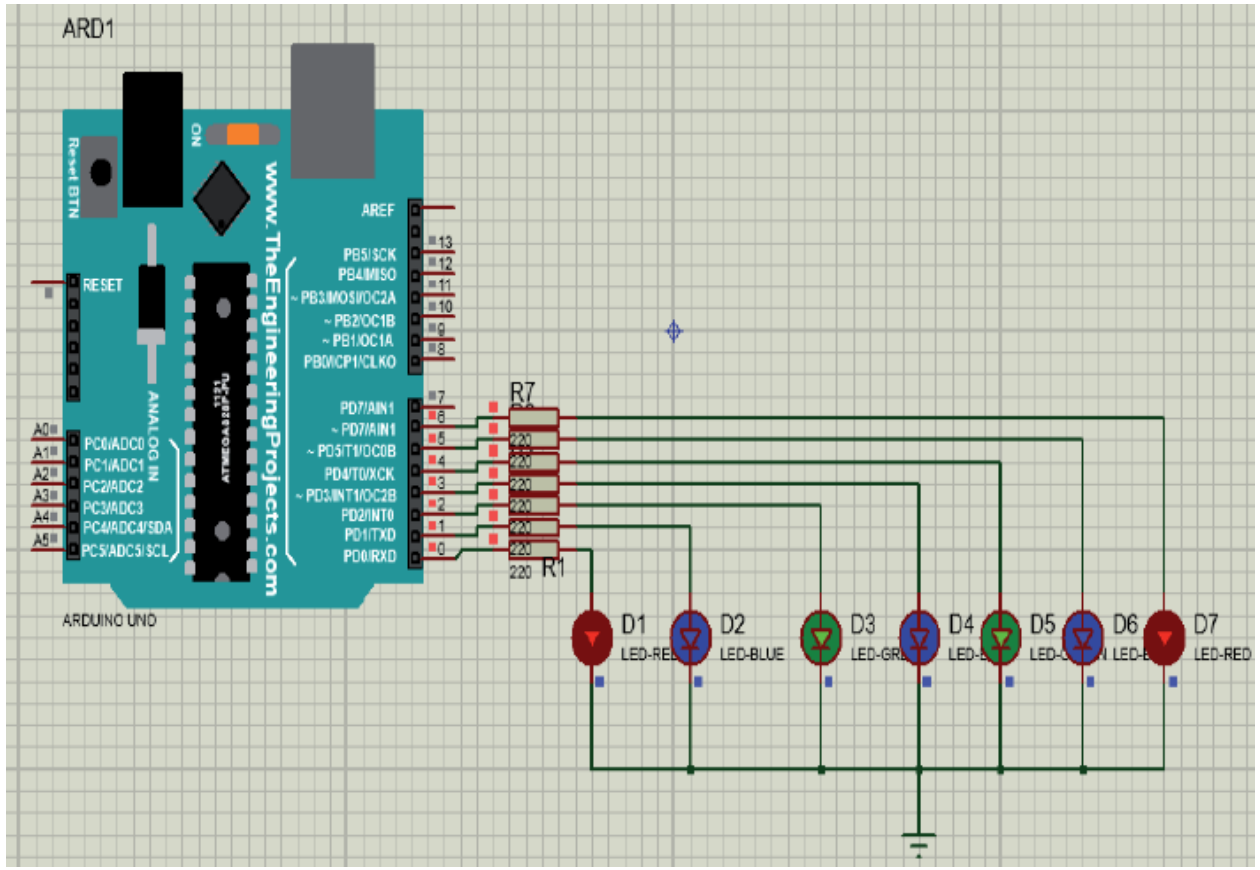
Programa En Arduino

```
void juegos_luces4() { //Clase4: Prender luces rojas
  for (; n < 7; n++) { //For n
    for (; j < 7; j = j + 6) { //For j anidado
      digitalWrite(vector_leds[j], HIGH);
      delay(100);
      digitalWrite(vector_leds[j], LOW);
      delay(100);
    }
    j = 0;
  }
}

void juegos_luces5() { //Clase5: Prender todos las leces en secuencia
  digitalWrite(vector_leds[0], HIGH);
  delay(200);
  digitalWrite(vector_leds[1], HIGH); //Encendemos todos los led's
  delay(200);
  digitalWrite(vector_leds[2], HIGH);
  delay(200);
  digitalWrite(vector_leds[3], HIGH);
  delay(200);
  digitalWrite(vector_leds[4], HIGH);
  delay(200);
  digitalWrite(vector_leds[5], HIGH);
  delay(200);
  digitalWrite(vector_leds[6], HIGH);
  delay(200);
}
```

III. PUERTOS ENTRADA SALIDA

3.4.2 Juego De Led’s

Simulación



III. PUERTOS ENTRADA SALIDA

3.4 Ejercicios De Aplicación Puestos Salida

3.4.3 Programa Para Controlar Un Led RGB Con Arduino

Descripción

Programa para controlar un Led RGB con Arduino, muestra los 3 colores principales uno después de otro (rojo, verde y azul). Permite elegir entre tres colores (amarillo (y), naranja (o) o rosado (p)) usando sus siglas en inglés.

III. PUERTOS ENTRADA SALIDA

3.4.3 Programa Para Controlar Un Led RGB Con Arduino

Programa En Arduino

```
/*----Declaracion de variables para cada color R G B----*/
int rled = 11; //Pin PWM 11 para led rojo
int bled = 10; //Pin PWM 10 para led azul
int gled = 9; //Pin PWM 9 para led verde

/*----Declaracion de variables auxiliares----*/
int i; //Variable para ciclos repetitiovos
int repeat = 5; //Variables para cantidad limite de repeticiones

void setup() {
  /*----Se inicializan para PWM como salida----*/
  pinMode(rled, OUTPUT);
  pinMode(bled, OUTPUT);
  pinMode(gled, OUTPUT);
}

void loop() {
  for (i = 0; i < repeat; j++) //Se repite la ejecucion de la funcion rgbon () "repeat veces rgbon();
  delay(1000); //Se espera 1 segundo
  colors("y"); //Se enciende el LED en color amarillo (y de yellow)
  delay(1000);
  colors("o"); //Se enciende el LED en color naranja (o de orange)
  delay(1000);
  colors("o"); //Se enciende el LED en color rosado (p de pink)
  delay(1000);
}
```

III. PUERTOS ENTRADA SALIDA

3.4.3 Programa Para Controlar Un Led RGB Con Arduino

Programa En Arduino

```
/*----Funcion para mostrar colores principales cada 500ms----*/
void rgbon() {
  analogWrite(rled, 255); //Se enciende color rojo
  delay(500); //Se esperan 500ms
  analogWrite(rled, 0); //Se apaga color rojo
  analogWrite(bled, 255); //Se enciende color azul
  delay(500); //Se esperan 500ms
  analogWrite(bled, 0); //Se apaga color azul
  analogWrite(gled, 255); //Se enciende color verde
  delay(500); //Se esperan 500ms
  analogWrite(gled, 255); //Se apaga color verde
}
```

III. PUERTOS ENTRADA SALIDA

3.4.3 Programa Para Controlar Un Led RGB Con Arduino

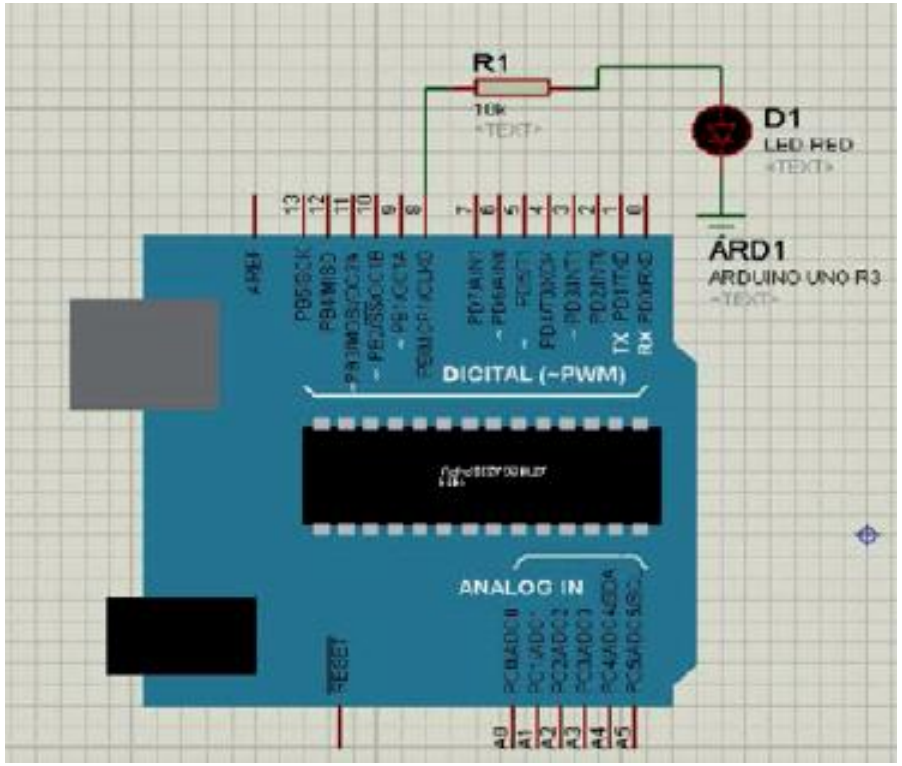
Programa En Arduino

```
/*----Funcion que permite escoger entre colores amarillo, naranja o rosado----*/
void colors(char color) { //La funcion recibe un parametro que guarda en variable color
  switch (color) { //Se compara variable color con dato guardado
    case 'y':
      analogWrite(rled, 255); //Si color =='y' se enciende el color amarillo
      analogWrite(gled, 255); //Mezclando r=255 / g=255 / b=0
      analogWrite(bled, 0);
      break;
    case 'o':
      analogWrite(rled, 255); //Si color =='o' se enciende el color naranja
      analogWrite(gled, 180); //Mezclando r=255 / g=180 / b=0
      analogWrite(bled, 0);
      break;
    case 'p':
      analogWrite(rled, 255); //Si color =='p' se enciende el color rosado
      analogWrite(gled, 0); //Mezclando r=255 / g=0 / b=255
      analogWrite(bled, 255);
      break;
  }
}
```

III. PUERTOS ENTRADA SALIDA

3.4.3 Programa Para Controlar Un Led RGB Con Arduino

Simulación



III. PUERTOS ENTRADA SALIDA

3.5 Puertos Como Entrada

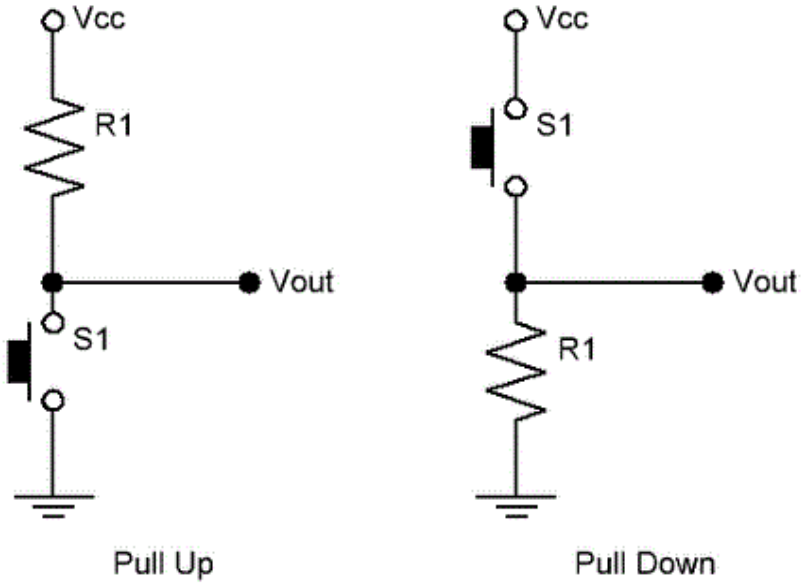
Se consideran entradas digitales en el momento que se recibe 0 lógico o 1 lógico del exterior, sea desde un sensor, un switch, un botön, etc. Las entradas digitales solo reconocen dos tipos de entrada, 0 lógico o 1 lógico donde el umbral de cada uno es el siguiente:

- 0 lógico: 0 voltios y 1,8 voltios
- 1 16gico: 2,1 voltios y 5 voltios

III. PUERTOS ENTRADA SALIDA

3.5 Puertos Como Entrada

El rango entre 1,9 y 2,1 voltios está considerado como error, donde el sistema puede entender como 1 o 0 lógico y puede dar fallos, para asegurarnos el correcto funcionamiento y en los casos que utilicemos switchs o pulsadores como en las simulaciones utilizaremos configuraciones con resistencias de modo PULL UP o PULL DOWN.



III. PUERTOS ENTRADA SALIDA

3.5 Puertos Como Entrada

Para poder recibir datos, es necesario conocer lo que son las estructuras de control, por el momento veremos el funcionamiento de la estructura if else.

if else es una estructura de control que permite redirigir un curso de acción según la evaluación de una condición simple, sea verdadera o falsa.

III. PUERTOS ENTRADA SALIDA

3.5 Puertos Como Entrada

Si la condición es verdadera, se ejecuta el bloque se sentencias 1, de lo contrario, se ejecutará el bloque de sentencia 2, todo lo que está dentro de los paréntesis.

```
IF (Condición){
     (Bloque de sentencias 1)
       }
        ELSE {
        (Bloque de sentencias 2)
        }
```

Cambiando la estructura de programación para usarla directamente con el Arduino sería de la siguiente forma:

```
if(digitalRead(pin a usar)==HIGH){
  //LÍNEAS DE PROGRAMACIÓN
}
else
{
   //LÍNEAS DE PROGRAMACIÓN
}
```

III. PUERTOS ENTRADA SALIDA

3.6 Ejercicio De Aplicación Puertos Como Entrada

3.6.1 Encendido Y Apagado De Led

Descripción

El sistema realizara el funcionamiento básico del encendido y apagado de un led en relación con el estado de un dipswitch, la configuración debe ser en pull-down.

III. PUERTOS ENTRADA SALIDA

3.6.1 Encendido Y Apagado De Led

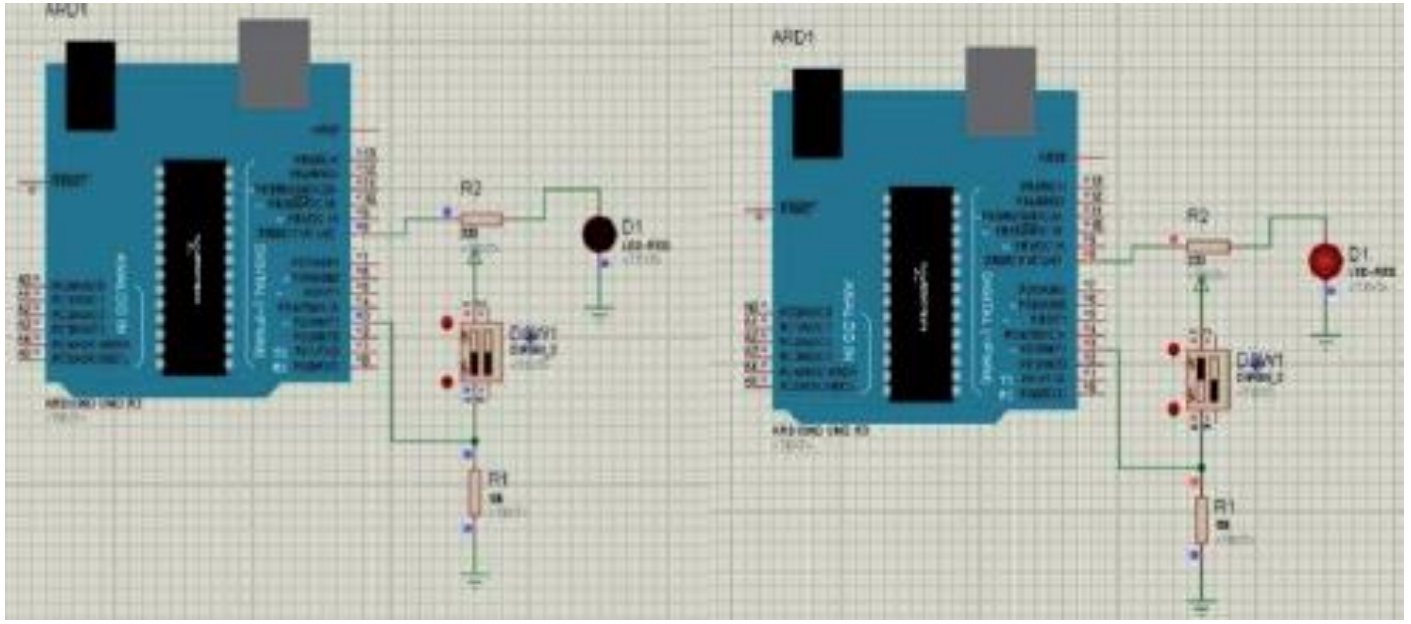
Programa En Arduino

```
//Declarar variables
int sw = 3;
int led = 8;

void setup() {
  pinMode(sw, INPUT);
  pinMode(led, OUTPUT);
}

void loop() {
  if (digitalRead(sw) == HIGH) {
    digitalWrite(led, HIGH);
  }
  else {
    digitalWrite(led, LOW);
  }
}
```

III. PUERTOS ENTRADA SALIDA

3.6.1 Encendido Y Apagado De Led

Simulación



III. PUERTOS ENTRADA SALIDA

3.6 Ejercicio De Aplicación Puertos Como Entrada

3.6.2 Activación De Led Mediante Pulsador

Descripción

El programa realizado cambia el dipswitch por un pulsador. El cual cuando se presiona una vez enciende el led, si se presiona nuevamente el led se apaga. Este proceso se lo puede realizar mediante el uso de una variable llamada encendido que cambia su valor de 1 a 0 o viceversa cada vez que el pulsador es presionado. Finalmente, es necesario utilizar un delay antirebotes. Con el fin de retardar la lectura del programa y se acople a la velocidad de un humano al presionar un botón.

III. PUERTOS ENTRADA SALIDA

3.6.2 Activación De Led Mediante Pulsador

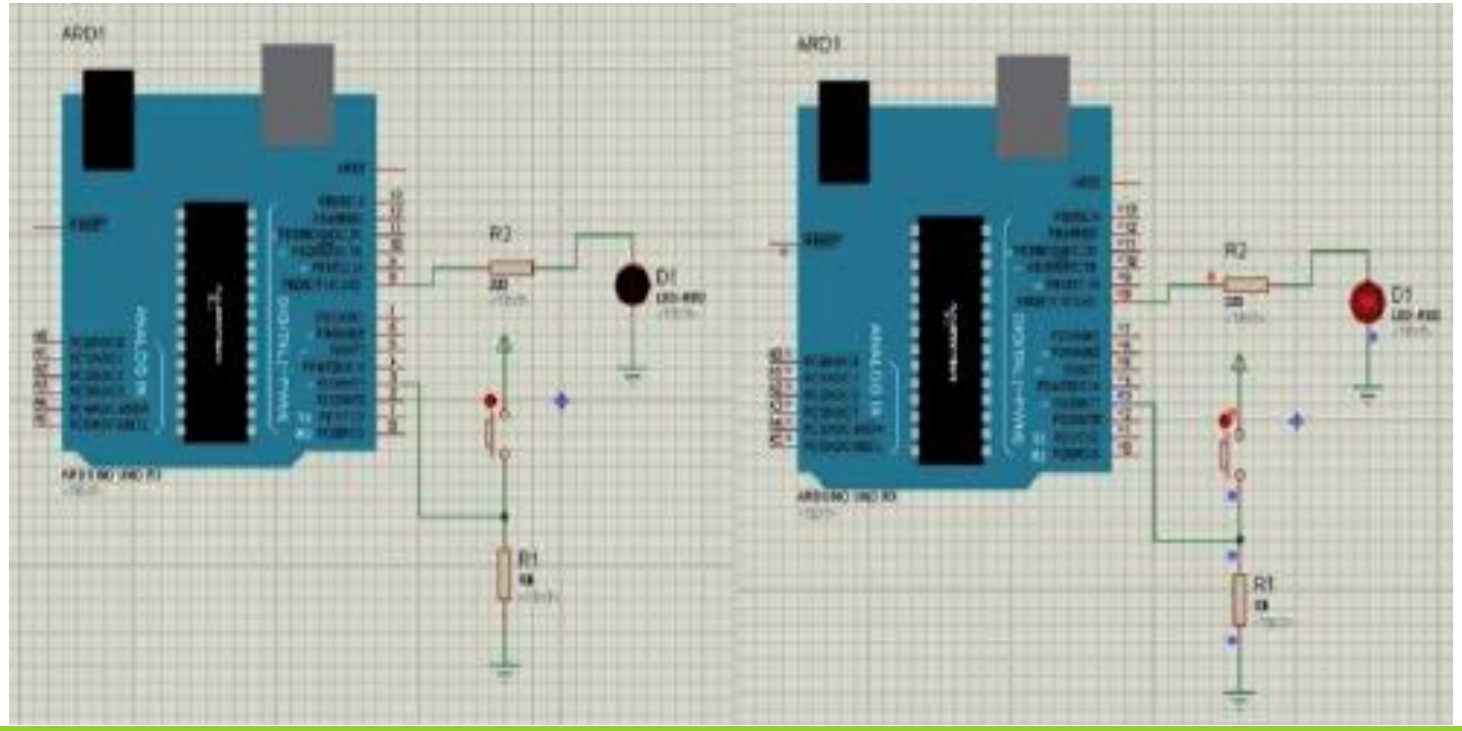
Programa En Arduino

```
//Declarar variables
int sw = 3;
int led = 8;
int encendido = 0; //Variable de encendido de led

void setup() {
  pinMode(sw, INPUT);
  pinMode(led, OUTPUT);
}

void loop() {
  if (digitalRead(sw) == HIGH) {
    delay(200); //Delay anti rebotes
    encendido=1-encendido;
    }
    if(encendido==1)
    digitalWrite(led, HIGH);
    else
    digitalWrite(led,LOW);
  }
```

III. PUERTOS ENTRADA SALIDA

3.6.2 Activación De Led Mediante Pulsador

Simulación



III. PUERTOS ENTRADA SALIDA

3.6 Ejercicio De Aplicación Puertos Como Entrada

3.6.3 Contador de 0 a 99 con display

Descripción

El sistema realiza un contador de 0 a 99 mediante la presión de pulsador que suma de uno en uno los valores, los mismos que son visualizados de displays de 7 segmentos.

Programa En Arduino

III. PUERTOS ENTRADA SALIDA

3.6.3 Contador de 0 a 99 con display

Programa En Arduino

```
if (digitalRead(Reset) == HIGH && aux3 == true) { //Codigo para resetear
  aux3 = false;
  T1 = 0;
  T2 = 0;
}

if (digitalRead(Reset) == LOW && aux3 == true)(aux3=true) { //Condición
  BCD(T1, d1, c1, b1, a1);
  BCD(T2, d2, c2, b2, a2);
}
```

III. PUERTOS ENTRADA SALIDA

3.6.3 Contador de 0 a 99 con display

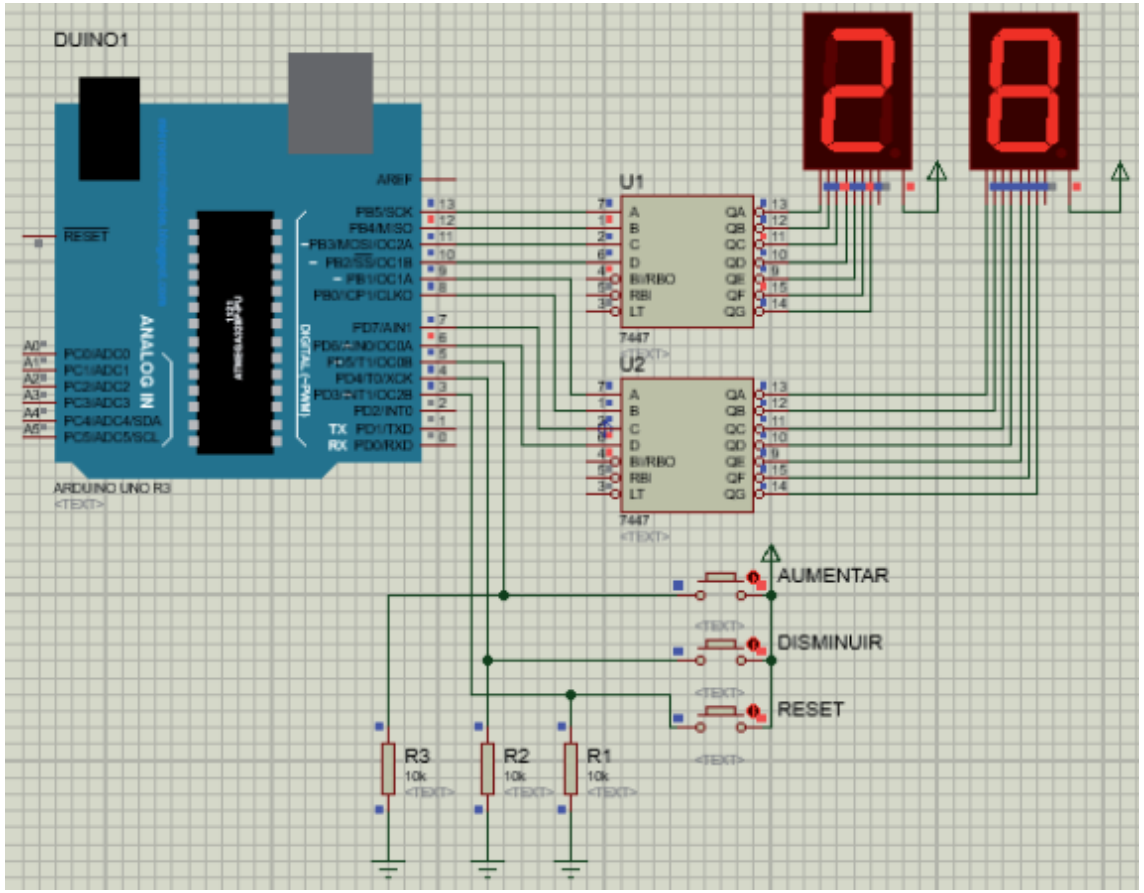
Programa En Arduino

```
void BCD(byte Contador, byte D, byte C, byte B, byte A) {
  switch (Contador) {
    case 0: //Caso 1
      digitalWrite(D, LOW); digitalWrite(C, LOW); digitalWrite(B, LOW); digitalWrite(A, LOW); break;
    case 1: //Caso 2
      digitalWrite(D, LOW); digitalWrite(C, LOW); digitalWrite(B, LOW); digitalWrite(A, HIGH); break;
    case 2: //Caso 3
      digitalWrite(D, LOW); digitalWrite(C, LOW); digitalWrite(B, HIGH); digitalWrite(A, LOW); break;
    case 3: //Caso 4
      digitalWrite(D, LOW); digitalWrite(C, LOW); digitalWrite(B, HIGH); digitalWrite(A, HIGH); break;
    case 4: //Caso 5
      digitalWrite(D, LOW); digitalWrite(C, HIGH); digitalWrite(B, LOW); digitalWrite(A, LOW); break;
    case 5: //Caso 6
      digitalWrite(D, LOW); digitalWrite(C, HIGH); digitalWrite(B, LOW); digitalWrite(A, HIGH); break;
    case 6: //Caso 7
      digitalWrite(D, LOW); digitalWrite(C, HIGH); digitalWrite(B, HIGH); digitalWrite(A, LOW); break;
    case 7: //Caso 8
      digitalWrite(D, LOW); digitalWrite(C, HIGH); digitalWrite(B, HIGH); digitalWrite(A, HIGH); break;
    case 8: //Caso 9
      digitalWrite(D, HIGH); digitalWrite(C, LOW); digitalWrite(B, LOW); digitalWrite(A, LOW); break;
    case 9: //Caso 10
      digitalWrite(D, HIGH); digitalWrite(C, LOW); digitalWrite(B, LOW); digitalWrite(A, HIGH); break;
  }
}
```

III. PUERTOS ENTRADA SALIDA

3.6.3 Contador de 0 a 99 con display

Simulación



III. PUERTOS ENTRADA SALIDA

3.7 Ejercicios Propuestos

3.7.1 Programa Para Generar Diferentes Juegos De Luces Navideñas

3.7.2 Programa Para Activar 4 Juegos Mediante Un Dipswitch De 4

3.7.3 Realizar Un Sumador Binario Del 0 Al 32 Con 5 Led´s Y Un Pulsador