Análisis y Diseño

Parcial 2

JESUS ANTONIO IBARRA AGUDELO
JUAN ANDRES URBIÑEZ GOMEZ

Departamento de Ingeniería Electrónica y Telecomunicaciones
Universidad de Antioquia
Medellín
Septiembre de 2021

Índice

1. Sección de contenido

1.1. Análisis del problema

Análisis del problema: Se busca diseñar un sistema el cual permita presentar en una pantalla con leds RGB la nacionalidad de los competidores que han llegado al podio de triunfadores para los juegos olímpicos de París 2024.

Teniendo en cuenta este problema, se hará un control de matrices de leds RGB 16x16, utilizando tiras de NeoPixel conectadas a la protoboard de tal forma que, al implementar el algoritmo correspondiente, hará que estos leds RGB se enciendan de tal manera que nos mostrara los colores de la bandera de nacionalidad de los ganadores de los juegos olímpicos.

Consideraciones de las alternativas de solución:

- La plataforma de Tinkercad, ya que se podrá realizar el circuito sin tener que preocuparse por riesgo que conlleva un circuito físico.
- Se utilizará una aplicación de desarrollo de programa llamada QT, que nos permitirá la facilidad de hacer el código que se implementará para realizar el proyecto propuesto.
- Se realizará el código que cumpla con las condiciones propuestas para este proyecto. Dicho código consistirá en crear una clase llamada "ImageRead" que nos permitirá leer una imagen de archivo jpg.
- Además, se creará unas funciones que nos permitirá submuestrear y sobremuestrear la imagen, permitiendo adaptar la imagen de tal forma que se podrá encender los leds correspondientes en la matriz 16x16. La imagen modificada se guardará en un archivo txt.
- En Tinkercad se realizará un código que nos permitirá recibir la matriz que esta guardada en el archivo txt, de tal forma se podrá encender los leds RGB según la matriz recibida

1.2. Esquema

- Investigar técnicas o métodos de sobremuestreo y submuestreo, y cómo funcionan.
- Buscar cómo funcionan las características principales del espacio de colores RGB.
- Crear una clase en QT llamada "ImagenRead", donde estará la función que nos permitirá leer el archivo jpg.
- Crear una función "conteo" que nos permitirá contar la cantidad de pixeles rojos, verdes y azules que hallan en un pixel.

- Se hará unas nuevas funciones llamadas “submuestrear” y “sobremuestrear” donde estarán las funciones que nos permitirán modificar la imagen que se guardara en forma de matriz en un archivo txt.
- En tinkercad, se creará un circuito utilizando tiras de NeoPixel conectadas a la protoboard.
- Una vez el circuito está hecho, se creará una función que nos permitirá ingresar la matriz hecha en QT.
- Se hará el código correspondiente para encender las tiras de NeoPixel, de tal forma que nos mostrará el archivo jpg en la matriz de leds 16x16.

1.3. Algoritmo diseñado

- Lo primero que se hace es crear la clase “ImagenRead” y le incluimos la librería QImagen. En el constructor, utilizando las funciones que vienen incluidas en la librería que agregamos, podremos ingresar la dirección del archivo jpg y con un ciclo, podremos iterar en toda la imagen pixel por pixel, lo que nos permitirá leer la imagen.
- En la clase que ya hemos creado, se hará una función llamada “conteo”, esta función nos permitirá contar la cantidad de rojos, verdes y azules que hallan en un pixel. Una vez que sabemos la cantidad de pixeles, podremos compararla con nuestra matriz 16x16, esto nos permitirá saber si se debe submuestruar o sobremuestrar.
- Se creará la función de “submuestreo”, debido a la función conteo, ya tenemos la cantidad de pixeles de la imagen, con este dato podremos submuestrar la imagen de ser necesario, es decir, modificaremos el ancho y el largo de la imagen de tal forma que podrá concordar con nuestra matriz de 16x16. En otras palabras, al utilizar el método correcto, podremos crear una versión más pequeña de la imagen original que seguirá conservando su aspecto, esta versión más pequeña será la que tendremos que incluir en nuestro circuito y al hacer que los leds se enciendan, nos mostraran la versión pequeña de nuestra imagen.
- Se creará la función de “sobremuestreo”, debido a la función conteo, ya tenemos la cantidad de pixeles de la imagen, con este dato podremos sobremuestrar la imagen, es decir, hacer el proceso inverso de submuestreo. Si en submuestreo hacíamos la imagen más pequeña para concordar con nuestra matriz, aquí, en sobremuestreo, haremos una versión más grande de la imagen original mientras que aún conserva su aspecto, esta versión más grande será la que enviaremos a nuestro circuito y al hacer que los leds se enciendan, nos mostrará esta versión más grande de nuestra imagen.
- Se agregará una nueva función que se llamará “matriz”, esta función nos permitirá recibir la imagen modificada mediante el submuestreo o el so-

bremuestreo, dándonos la posibilidad de crear una matriz que guardará los píxeles de color de la imagen.

- Por último, se creará una función llamada "guardar matriz", esta función será la responsable de recibir la matriz hecha por la función antes mencionada y guardarla en un archivo txt. Esta matriz que se guardó, será la que se enviara el circuito en tinkercad y los leds se encenderán dependiendo de los datos suministrados por este archivo.

- Ya en tinkercad, se define los pines que se van a utilizar en el circuito y la cantidad de leds que se va a utilizar, en este caso 16x16. No se nos puede olvidar incluir la librería "Adafruit NeoPixel" ya en esta librería se encuentra la clase necesaria que nos permitirá encender los leds de la tira NeoPixel.

- Lo que nos queda por hacer es configurar el código para encender los leds de la tira NeoPixel de tal forma que concuerde con la matriz ya leída, así podremos lograr que nuestro circuito nos muestre la nacionalidad de los ganadores de los juegos olímpicos de París 2024.

1.4. Algoritmo diseñado

- No llevar los leds al límite, es decir, que sus valores RGB no sean todas de 255, porque el simulador se buggea con estos valores y no se encenderán todos los leds. Si se quiere el color blanco, entonces por lo menos un pixel debe ser menos a 255, por ejemplo, 255,255,254 así todavía podemos obtener el color blanco.

- Tener en cuenta que al terminar una fila de las tiras NeoPixel, saltara al primer led de la segunda fila, es decir, tenemos una matriz 16x16, comenzamos en la fila 1 columna 1 (1,1), cuando llevamos al último led de la primera fila (1,16) saltaremos a la segunda fila e iniciaremos con el primer led (2,1); y así sucesivamente hasta completar la matriz (16,16).

- Estar seguros de que las tiras NeoPixel estén debidamente conectadas para que pueda saltar a la siguiente fila de tal forma como se mencionó en la consideración anterior.

- Se utilizará un arreglo tridimensional 3x16x16 para poder hacer la matriz, teniendo en cuenta el primer índice serán los colores (rojo, verde o azul), el segundo serán las filas y el último será el valor del color separado en columnas.

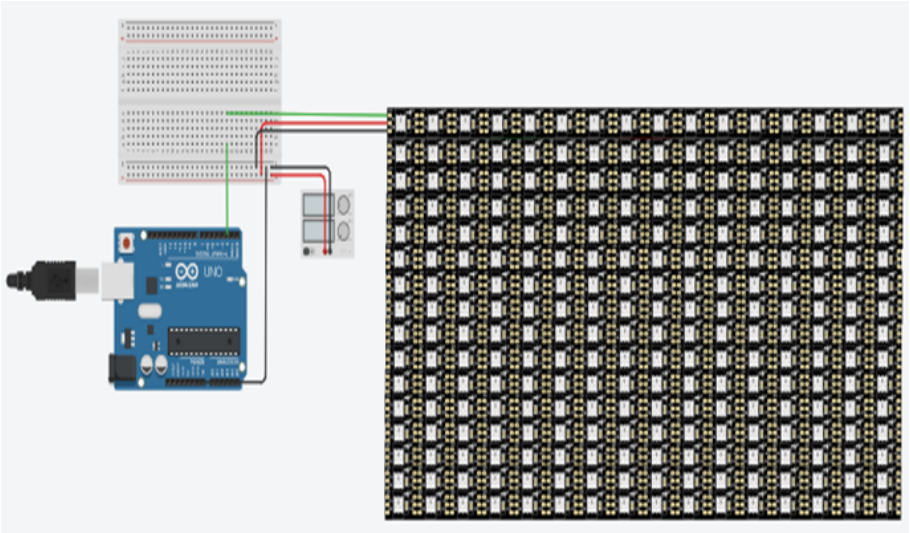

Figura 1: Protipo del circuito