Informa2 S.A.S.

Parcial Informatica 2

Mariana Noreña Vasquez
Santiago Velez Arboleda
Carolina Jimenez Restrepo

Despartamento de Ingeniería Electrónica y
Telecomunicaciones
Universidad de Antioquia
Medellín
Abril de 2021

Índice

1. Sección introductoria

En el mundo de la ingeniería cada día se busca crear ideas y soluciones practicas que ayuden con la innovación de todos los sectores, este proyecto nos da un claro ejemplo de esto, mensajes luminosos en las fachadas de muchos lugares comerciales que atraen personas y crean en ellas una imagen llamativa del lugar, que permite recordarlo con más facilidad, detrás de estas sorprendentes fachadas hay todo un proceso ingenieril, el cual se compone de dispositivos electrónicos y líneas de código que combinados nos dan como resultado un brillante y llamativo mensaje.

2. Sección de contenido

2.1. Análisis del problema y consideraciones para la alternativa de solución propuesta

Implementar un arreglo que permita mostrar cada fila de la matriz de leds.Para programar el encendido y apagado de cada led y poder formar una figura, utilizamos la técnica pixel art con la cual realizamos la matriz de los caracteres que se imprimen, en la silueta o parte más oscura del carácter se pondrán unos (1) y en la parte clara ceros (0), esto nos permite que en el 1 el led este prendido y en el 0 apagado.
Para el montaje del circuito se utilizan 3 pines los cuales van conectados a dos circuitos integrados, se indago sobre la utilización de estos y como poder interconectarlos, se comparten los relojes y las entradas a través de la salida invertida. Para la matriz de leds se utilizan resistencias para cada fila y columna, las cuales son de 560 Ω.

En la Figura (1), se presenta el diseño de la matriz de los caracteres usados.

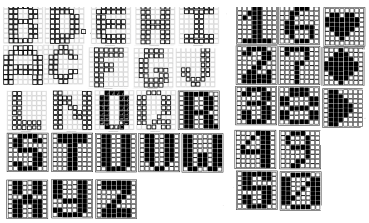


Figura 1: Matriz caracteres

2.2. Esquema de tareas definidas en el desarrollo del algoritmo.

El primer paso para el desarrollo del algoritmo fue realizar el montaje del circuito ya que con este se pudo ir probando nuestro algoritmo. Para elaborar

el algoritmo se tuvo en cuenta las especificaciones dadas y a partir de ellas se planearon las tareas o pasos para lograr cumplir con el objetivo pedido.
La primer tarea a realizar es la función de verificación, para la cual se definió una matriz en número decimal tomados de la conversión de unos (1) en número binario que representa la matriz de leds 8x8, con la cual por medio de un ciclo muestra cada fila de leds iluminada y así como lo dice su nombre verificar que todos los leds estén encendidos. De esta misma manera se implementó una función apagar, la cual fue desarrollada con el mismo método de verificación, la matriz es representada por números cero (0) los cuales indican que el led no se enciende, esta función se crea para evitar que los leds permanezcan encendidos mientras el usuario decide ejecutar otra acción.
La segunda tarea a realizar es definir que patrones pueden ser impresos, para esto se tiene: números del 0-9, letras (A-Z) excepto la Ñ y como requisito se ingresan en mayúscula, caracteres especiales, rombo, corazón y símbolo de play. Tomando en cuenta esto se procede a elaborar la función imagen la cual esta compuesta de una matriz que contiene números decimales tomados de la conversión de números binarios los cuales representan cada columna del patrón antes descrito e indica si se enciende o permanece apagado el led, para las filas se utilizo un arreglo o mascara la cual tiene el mismo contenido de números decimales y su función es ir corriendo los bits para así tener control de filas y columnas de cada patrón a imprimir, ya que se debe ir iluminando por filas y columnas la matriz de leds.
La tercer tarea es realizar la función publik en la cual se utiliza la función imagen y se agrega un tiempo para controlar el tiempo que debe haber entre patrón y patrón, el tiempo escogido por el usuario se multiplica por 1000 para que el delay nos sirva como controlador de tiempo entre cada visualización.
Por ultimo se realizó el menú del programa, el cual esta compuesto por tres opciones.
Opción 1: Verificación
Opción 2: Imagen
Opción 3: Patrones consecutivos

2.3. Código implementado

A continuación, se presenta el código desarrollado para elaborar el programa.

```
//Pines
int SERF = 2; //Entrada
int SRCLKF = 4; //Registro de entrada
int RCLKF = 3; // Registro de salida

int n=0;
```

```
//Arreglo para recorrer las filas

int Filas[] = {127, 191, 223, 239, 247, 251, 253, 254};

//Caracteres a mostrar

int A[] = {0,60,102,102,126,102,102,102};
int B[] = {120,72,72,112,72,68,68,124};
int C[] = {0,30,32,64,64,64,32,30};
int D[] = {0,56,36,34,34,36,56,0};
int E[] = {0,56,32,56,32,32,60,0};
int F[] = {0,60,32,56,32,32,32,0};
int G[] = {0,62,32,32,46,34,62,0};
int H[] = {0,36,36,60,36,36,36,0};
int I[] = {0,56,16,16,16,16,56,0};
int J[] = {0,28,8,8,8,40,56,0};
int K[] = {0,36,40,48,40,40,40,0};
int L[] = {0,32,32,32,32,32,60,0};
int M[] = {0,0,68,170,146,130,130,0};
int N[] = {0,34,50,42,38,34,0,0};
int O[] = {0,60,66,66,66,66,60,0};
int P[] = {0,56,36,36,56,32,32,0};
int Q[] = {0,60,66,66,66,70,62,1};
int R[] = {0,56,36,36,56,36,36,0};
int S[] = {0,60,32,60,4,4,60,0};
int T[] = {0,124,16,16,16,16,16,0};
int U[] = {0,66,66,66,66,36,24,0};
int V[] = {0,34,34,34,20,20,8,0};
int W[] = {0,130,146,84,84,40,0,0};
int X[] = {0,66,36,24,24,36,66,0};
int Y[] = {0,68,40,16,16,16,16,0};
int Z[] = {0,60,4,8,16,32,60,0};

//NUMEROS
int N1[] = {24, 28, 26,24,24,24,126};
int N2[] = {60, 102, 96, 48,24,76,126,126};
int N3[] = {60, 102, 96, 60 ,96 , 102,102,60};
int N4[] = {112, 104, 100, 98, 126,96,96,96};
int N5[] = {126,102,6,126,96,102,102,60};
int N6[] = {56,12,4,6,126,70,68,56};
int N7[] = {62,34,32,24,8,8,8,8};
int N8[] = {60,60,195,195,60,195,195,60};
int N9[] = {28,34,34,60,32,32,24,12};
int N0[] = {60, 195, 227, 211, 203,199, 195, 60};

//CARACTERES ESPECIALES
```

```
int CORAZON[] = {0,54,127,127,62,28,8,0};
int ROMBO[] = {8, 28,62,127,127,62,28,8};
int PLAY[] = {4,12,28,60,60,28,12,4};

//Prototipos Funciones
void verificacion();
void apagar();
void imagen(char);
void publik(char *, int, float);


void setup(){
  // Entradas y registros
  int puertos[]={ 2, 3 , 4};

  for(int i = 2; i<5; i++){
    pinMode(puertos[i], OUTPUT);
  }

  Serial.begin(9600);
}


void loop(){

  //MENU DE OPCIONES
  int op;

  Serial.println("1. Verificar  2. Imagen   3. Patrones consecutivos");

  delay(3000);
  while(Serial.available()>0){
        op=Serial.parseInt();

    switch (op){

      case 1: //Verificacion
      //delay(1000); //Tiempo para que muestre la funcion de verificacion
      verificacion();
      apagar();

      break;

      case 2: //Imagen
      Serial.println("Ingrese el patron: ");
      delay(3000);
```

```
while(Serial.available()>0){
  char crt=Serial.read();
  imagen(crt);
  apagar();
  crt='\0';
}

break;

case 3: //Publik

int cantP;
float tmp;

Serial.println("Ingrese la cantidad de patrones a mostrar: ");
delay(3000);
while(Serial.available()>0){ //Recibir la cantidad de protrones
  cantP=Serial.parseInt();
  Serial.println(cantP);
}

Serial.println("Ingrese el tiempo entre los patrones: ");
delay(3000);
while(Serial.available()>0){ //Para ingresarlo se pone .
  tmp=Serial.parseFloat();
  Serial.println(tmp);
}

char patrones[10];
Serial.println("Ingrese patrones: ");
for(int i=0; i<cantP; i++){
  delay(3000);
  while(Serial.available()>0){
    char car=Serial.read();
    patrones[i]=car;
    Serial.println(patrones[i]);
  }
}
publik(patrones, cantP, tmp);
apagar();
break;

default:

Serial.println("No se encontro la opcion ingresada");
```

```
      break;
    }
  }
}

void verificacion(){

  int *matriz = new int [8]{255,255,255,255,255,255,255,255};

  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, matriz[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(200);
  }
  delete[] matriz;
}

void apagar(){
  int *matriz = new int [8]{0,0,0,0,0,0,0,0};
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, matriz[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
  }
  delete[] matriz;
}

void publik(char *array, int n, float t){
  int tiempo=t*1000;
  for(int s=0; s<n; s++){
    delay(tiempo);
    imagen(array[s]);
    delay(tiempo);
  }

}

void imagen(char let){
  switch (let){

    case '0':
    while(n<10){
      for(int i=0; i<8; i++){
```

```
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N0[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
      apagar();
break;

case '1':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N1[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
      apagar();
break;

case '2':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N2[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case '3':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
```

```
      shiftOut(SERF, SRCLKF, MSBFIRST, N3[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
      apagar();
break;

case '4':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N4[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case '5':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N5[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case '6':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N6[i]);
```

```
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case '7':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N7[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case '8':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N8[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case '9':
while(n<10){
  for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, MSBFIRST, N9[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
```

```
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case 'A':
while(n<10){
    for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, LSBFIRST, A[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case 'B':
while(n<10){
    for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, LSBFIRST, B[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
      delay(10);
    }
    n++;
}
n=0;
apagar();
break;

case 'C':
while(n<10){
    for(int i=0; i<8; i++){
      digitalWrite(RCLKF,LOW);
      shiftOut(SERF, SRCLKF, LSBFIRST, C[i]);
      shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
      digitalWrite(RCLKF,HIGH);
```

```
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'D':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, D[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'E':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, E[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'F':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, F[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
```

```
  }
  n++;
}
n=0;
apagar();
break;

case 'G':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, G[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'H':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, H[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'I':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, I[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
```

```
  n++;
}
n=0;
apagar();
break;


case 'J':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, J[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'K':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, K[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'L':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, L[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
```

```
  n++;
}
n=0;
apagar();
break;

case 'M':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, M[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'N':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, N[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'O':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, O[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
```

```
}
n=0;
apagar();
break;

case 'P':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, P[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'Q':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, Q[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'R':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, R[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
```

```
n=0;
apagar();
break;

case 'S':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, S[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'T':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, T[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'U':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, U[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
```

```
apagar();
break;

case 'V':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, V[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'W':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, W[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'X':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, X[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
```

```
break;

case 'Y':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, Y[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'Z':
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, LSBFIRST, Z[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;

case 'h': //CORAZON
while(n<10){
  for(int i=0; i<8; i++){
    digitalWrite(RCLKF,LOW);
    shiftOut(SERF, SRCLKF, MSBFIRST, CORAZON[i]);
    shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
    digitalWrite(RCLKF,HIGH);
    delay(10);
  }
  n++;
}
n=0;
apagar();
break;
```

```
    case 'r': //ROMBO
    while(n<10){
      for(int i=0; i<8; i++){
        digitalWrite(RCLKF,LOW);
        shiftOut(SERF, SRCLKF, MSBFIRST, ROMBO[i]);
        shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
        digitalWrite(RCLKF,HIGH);
        delay(10);
      }
      n++;
    }
    n=0;
    apagar();
    break;

    case 'p': //SIGNO DE PLAY
    while(n<10){
      for(int i=0; i<8; i++){
        digitalWrite(RCLKF,LOW);
        shiftOut(SERF, SRCLKF, MSBFIRST, PLAY[i]);
        shiftOut(SERF, SRCLKF, LSBFIRST, Filas[i]);
        digitalWrite(RCLKF,HIGH);
        delay(10);
      }
      n++;
    }
    n=0;
    apagar();
    break;

    default:
    Serial.println("Patron ingresa invalido.");
    break;
  }

}
```

Link del circuito en tinkercat:
https://n9.cl/7xpek

2.4. Problemas de desarrollo que se presentaron

Al momento del montaje del circuito se presentaron varios problemas algunos de ellos son: las conexiones a los circuitos integrados, también la comunicación entre ellos y como enviar los datos entre los seriales.
Al momento de probar los arreglos de los patrones estos se imprimían invertidos o se prendían leds que no debían prenderse.
En cuanto a la salida por consola el mensaje con instrucciones para el usuario se repetía de forma indefinida ya que se encontraba en la función loop.

2.5. Evolución del algoritmo y consideraciones a tener en cuenta en la implementación

Como ya se tenia un esquema de como desarrollar el algoritmo se facilitó su implementación, al encontrarnos con los problemas antes descritos se indago sobre la solución correcta para cada uno de ellos.
Para dar solución al problema presentado durante el montaje del circuito se revisó el datasheet del integrado para profundizar en la apropiada conexión entre ellos, al analizar las conexiones agregamos el cable de potencia que permitió prender y apagar todos los leds.
En la implementación se debe tener en cuenta una serie de instrucciones presentadas a continuación:
Inicialmente se desplegará un menú con las siguientes opciones:
-Verificar: Le permitirá ver que todos los led se encienden.
-Imagen: Le permitirá ver un patrón que desee.
-Publick: Le permitirá ver una secuencia de patrones los cuales debe escoger al igual que el tiempo de duración entre ellos.
Siga las instrucciones que se le van mostrando en pantalla para hacer uso correcto del programa.
Este programa cuenta con una serie de restricciones al momento de elegir los patrones a visualizar a continuación se mostrarán cuales puede seleccionar.
Los patrones que se pueden visualizar son:
- Números (0-9): Ingrese el número que quiere ver
- Letras (A-Z): exceptuando Ñ; para ello ingrese la letra en mayúscula.
- Caracteres especiales: corazón, rombo, play.
Para verlos haga lo siguiente:
Si va a imprimir el corazón ingrese h, para el rombo ingrese r y para simbolo play ingrese p.
Cuando desee ejecutar la opción 3 del menú se recomienda que al momento de ingresar los patrones que quiere ver en los leds, ingrese el primer patrón y cuando este aparezca ingrese el segundo y así sucesivamente.

3. Inclusión de imágenes

3.1. Circuito

En la Figura (2), se presenta el diseño del circuito empleado para realizar el parcial

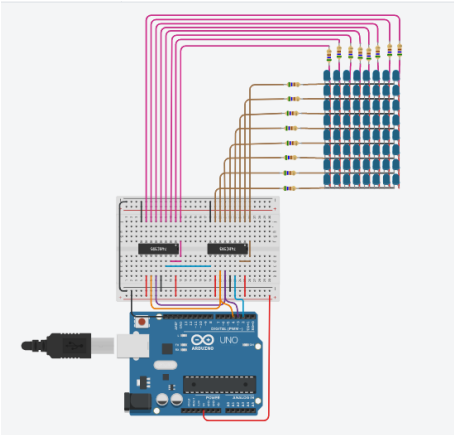
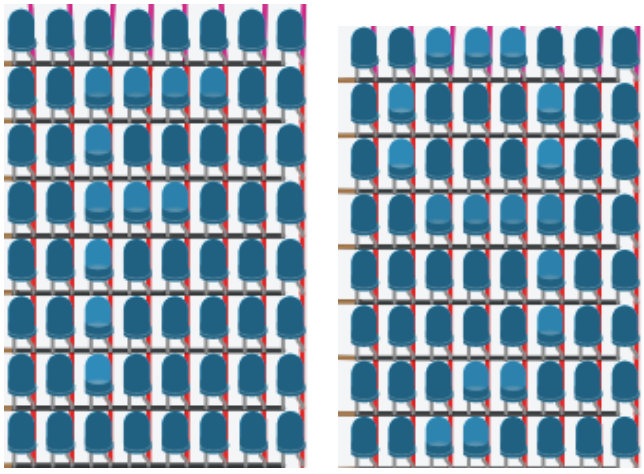


Figura 2: Montaje

3.2. Matriz de leds

En la Figura (3), se presentan la matriz de leds con diferentes patrones



Letra F Número 9

Figura 3: Matrices leds

4. Conclusión

De acuerdo al objetivo principal, imprimir una serie de patrones en una matriz de leds, elaboramos un proyecto que cumple con lo esperado y propuesto.