

# **Parcial I 2021-1 Informática**

## **2**

Informe de Desarrollo de Proyecto

**Juan Diego Cabrera Moncada**  
**Julian David Quintero Marin**  
**Julian Montenegro Pinzón**

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

# Índice

1. Introducción y desarrollo del proyecto	2
2. Análisis del problema y consideraciones para la alternativa de solución propuesta	2
3. Esquema de descripción de las tareas a definir en el desarrollo del algoritmo	3
4. Problemas de desarrollo presentados	4
5. Evolución del algoritmo y consideraciones a tener en cuenta en la implementación	11
6. Algoritmo implementado	16

## **1. Introducción y desarrollo del proyecto**

Este es el documento correspondiente al grupo de 3 personas conformado por los estudiantes del curso de Informática 2: Juan Diego Cabrera Moncada, Julian David Quintero Marin y Julian Montenegro Pinzón. El proyecto a desarrollar busca la visualización de patrones en una matriz de LEDs de manera efectiva, donde dichos patrones y el intervalo de tiempo entre cada patrón a mostrar son dados por el usuario.

## **2. Análisis del problema y consideraciones para la alternativa de solución propuesta**

El desarrollo de la solución al problema parte de la premisa de la posibilidad de controlar el encendido y apagado de LEDs de la matriz de manera organizada con base en la posición en la que cada LED se encuentre. De este modo, retomando como base el ejemplo dado por el profesor de Informática 2 Jonathan Ferney en el cual se hace uso de un circuito integrado 74HC595, se puede evidenciar que, si conectáramos la matriz de LEDs de forma que cada columna de LEDs se encuentre en serie usando el mismo código de ejemplo, se evidencia que se encenderían todos los LEDs de cada columna. Por lo cual se considera que la solución se debe fundamentar en el uso de las filas y columnas de la matriz como coordenadas de referencia para llevar a cabo la localización de cada LED de manera efectiva y organizada. Una consideración para el desarrollo de la solución es el uso de un componente capaz de inhibir la falta de voltaje y conservar la idea de plantear un sistema de coordenadas con filas y columnas. Con esto en mente, se busca la integración de transistores NPN y PNP como componentes de control de la electricidad que recorre el circuito, específicamente para el control de cada LED una vez localizados en su respectiva coordenada (Fila como X, Columna como Y). El día martes se opta por una alternativa de solución que deseché el uso de transistores y se crea un circuito cuya organización debe tenerse en cuenta para la creación del código correspondiente y así enlazar el circuito con el código como la solución propuesta del problema. De este modo, la solución propuesta radica en el uso del pin SER de un integrado que puede ser llamado el Integrado Master, pues las salidas de éste se encuentran conectadas a los pins SER de otros integrados, que llamaremos los Integrados Branch. Se busca que se ingrese electricidad por el pin SER del Integrado Master y, a través del correcto manejo de los pines SRCLK y RCLK de este integrado, la electricidad sea enviada por las salidas correctas de acuerdo con el integrado que se requiera usar para poder asignarle un estado a un LED en específico. De este modo, cada integrado está conectado a una columna de la matriz de LEDs, por lo cual se debe tomar estos integrados como la referencia para las columnas de la matriz mientras que las salidas correspondientes a cada integrado son establecidas como las coordenadas de la fila en la que se encuentra cada LED. Para continuar con el análisis del problema nos hemos reunido los tres compañeros a través de una videollamada, en la que leímos el problema y sacamos posibles soluciones

para abordar el proyecto de la forma más sencilla, aunque en ese momento no teníamos la certeza de cuál opción era más viable. Decidimos implementar 9 circuitos integrados(74HC595) en nuestro proyecto en donde el primer circuito integrado recibe los datos seriales y a partir de este circuito salen las señales para los datos seriales de los otros 8 circuitos integrados. Todos los registros de desplazamiento (SRCLK) y registros de salida (RCLK) están conectados a los pines digitales del Arduino. De esta forma el usuario puede ingresar los 8 datos (0=led apagado, 1=led encendido) por el puerto serial y activar los flancos de subida para hacer el corrimiento y salida de datos.

### 3. Esquema de descripción de las tareas a definir en el desarrollo del algoritmo

Las tareas a definir en el desarrollo del algoritmo corresponden a las siguientes:

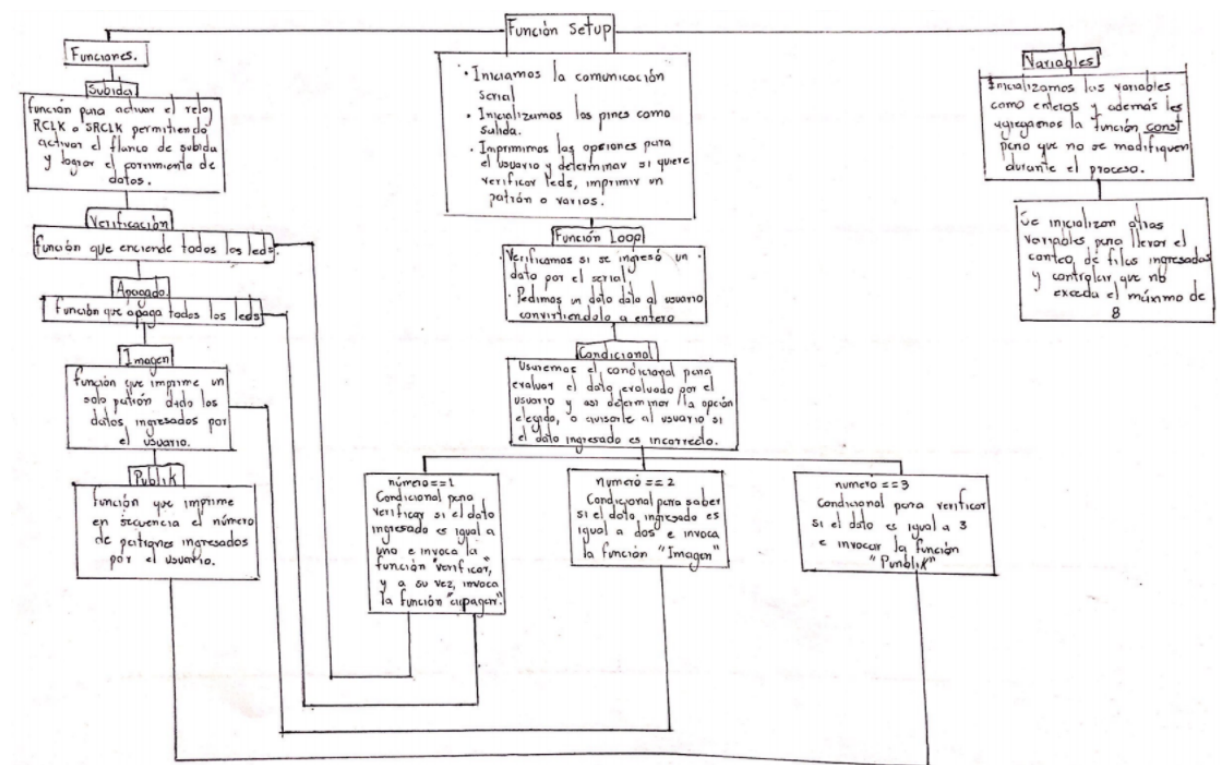


Figura 1: Tareas a desarrollar en el algoritmo

## 4. Problemas de desarrollo presentados

Al probar el circuito descrito en la primera sección de análisis del problema, en el cual cada columna está conectada en serie, se evidencia, por medio de la medición por multímetro, que el voltaje que se transmite por cada LED (1.6 voltios aproximadamente) es menor al que registraba con el ejemplo dado por el profesor (4.8 voltios aproximadamente). El transistor NPN que aparece en la imagen no tiene ninguna relevancia en el circuito. Posterior a ello, se procede

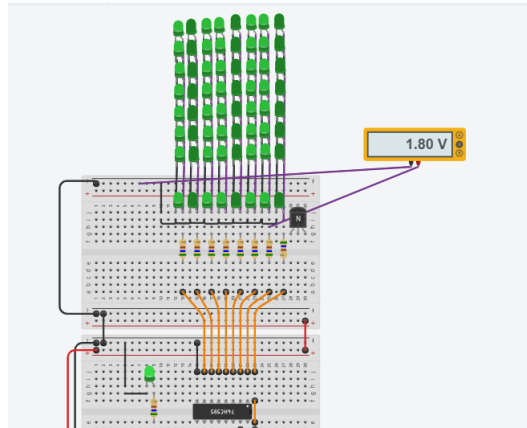


Figura 2: Prueba con columnas LED en serie basada en el circuito de ejemplo

a realizar una serie de conexiones introduciendo nuevos componentes con base en el análisis del problema inicialmente descrito (2). Para nuestro circuito base inicialmente usamos un Arduino uno, dos placas de prueba (protoboard's), transistores NPN, PNP, resistencias, diodos emisores de luz y dos circuitos integrados SN74HC595 que serán los protagonistas en el proyecto pues serán los encargados de entregar el proceso de control coordinado con nuestro código, usaremos 5 pines del Arduino, cuatro para los puertos SRCLK y RCLK de cada uno de los integrados y uno para los dos puertos SER de los integrados, desde este pin entregaremos la información que será transmitida a la matriz de leds.

Se finaliza el montaje del circuito base, después de esto se inicia el proceso de análisis de control para llegar al fin de proyectar el patrón deseado a nuestra matriz de leds 8X8. Se replantea el montaje del circuito base, en el cual se elimina el uso de transistores y en su lugar, se utiliza como estructura de control una cantidad de 9 integrados 74HC595, uno actúa como el Master cuyas salidas se conectan a los integrados Branch (Rama), dicho nombre puesto arbitrariamente para mayor claridad, y con cada integrado Branch se controla el estado de una columna de la matriz de 8x8 de LEDs. Cabe destacar que se conectan resistencias de 560 ohmios con el objetivo de restringir el flujo de electricidad a una magnitud con la cual se evite el daño o deterioro de LEDs durante el proceso de ejecución de la simulación y el código. Se usan 5 puertos digitales donde el puerto digital 2 está conectado al puerto serial SER del Integrado

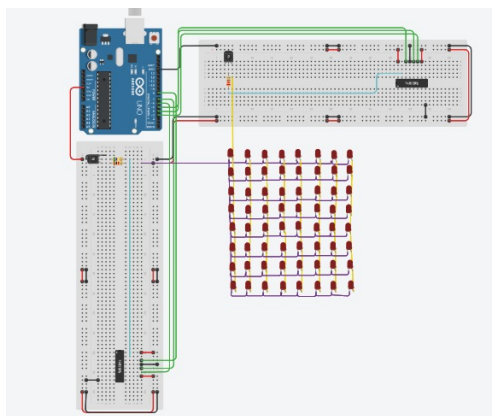


Figura 3: Circuito base de conexiones Versión 1

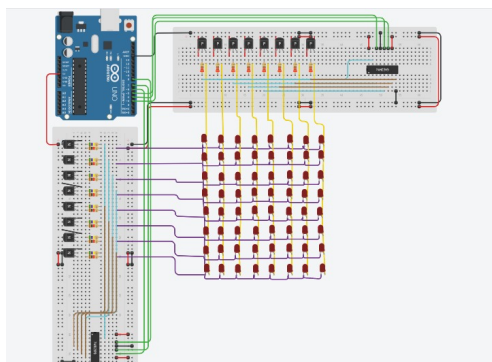


Figura 4: Circuito base de conexiones Versión 2

Master 74HC595, el 4 al puerto RCLK del mismo integrado, el 5 al puerto RCLK, y el 11 y 12 a los puertos RCLK Y SRCLK respectivamente de los integrados Branch. Posteriormente, se hizo progreso con respecto al montaje de este circuito base modificado hasta el punto en el que dicho circuito sea funcional. Por lo cual, se procedió a desarrollar pruebas con código una vez el circuito fue finalizado en su totalidad. En cuanto al código correspondiente a las pruebas, en las cuales satisfactoriamente se encendieron todos los LEDs de la matriz fue la siguiente:

```
// Programa desarrollado , compilado y ejecutado en https://https://www.tinkercad
const int SER = 2; //Puerto digital 2 Puerto serial
const int RCLK = 4; //Puerto digital 4 Puerto registro de salida
const int SRCLK = 5; //Puerto digital 5 Puerto registro de desplazamiento
const int ARCLK = 11; //Puerto digital 11 Puerto de registro de salida controlad
const int ASRCLK = 12; //Puerto digital 12 Puerto de registro de desplazamiento
```

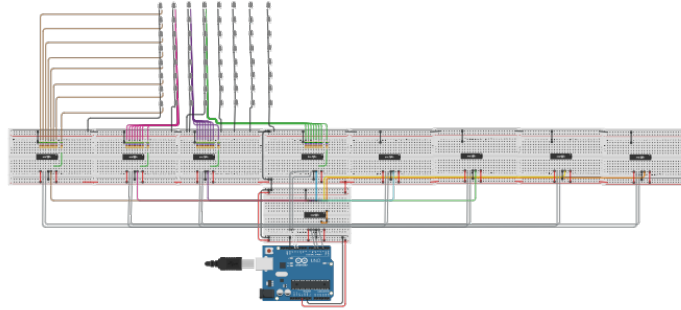


Figura 5: Modificación 1 del Circuito Base

```

void setup()
{
    // PASO 1 Configuración de puertos digitales como salida
    pinMode( SER , OUTPUT);
    pinMode( RCLK, OUTPUT);
    pinMode( SRCLK, OUTPUT);
    pinMode(ARCLK,OUTPUT);
    pinMode(ASRCLK,OUTPUT);

    //Paso 2 Inicialización de puertos digitales
    digitalWrite(SER,LOW);
    digitalWrite(RCLK,LOW);
    digitalWrite(SRCLK,LOW);
    digitalWrite(ASRCLK,LOW);
    digitalWrite(ARCLK,LOW);

    /*for(unsigned short i=0; i<=7; i++){
        digitalWrite(SER,1);
        digitalWrite(SRCLK,0);
        digitalWrite(SRCLK,1);
        digitalWrite(SRCLK,0);
    }
    digitalWrite(RCLK,0);
    digitalWrite(RCLK,1);
    digitalWrite(RCLK,0);
    */
}

```

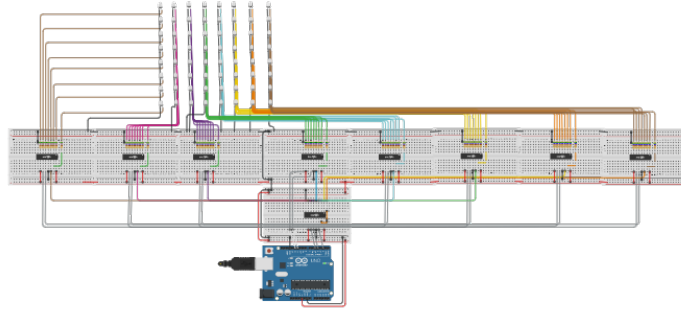


Figura 6: Modificación 2 del Circuito Base

```

//digitalWrite(12,0);
//digitalWrite(12,1);
//digitalWrite(12,0);

//digitalWrite(11,0);
//digitalWrite(11,1);
//digitalWrite(11,0);
delay(1);
for(unsigned short i=1; i<=8; i++){
    digitalWrite(SER,0);
    digitalWrite(SRCLK,0);
    digitalWrite(SRCLK,1);
    digitalWrite(SRCLK,0);
}
digitalWrite(RCLK,0);
digitalWrite(RCLK,1);
digitalWrite(RCLK,0);

digitalWrite(12,0);
digitalWrite(12,1);
digitalWrite(12,0);

digitalWrite(11,0);
digitalWrite(11,1);
digitalWrite(11,0);
delay(1);

```



```

}

void loop()
{
    for(unsigned short i=0; i<=7; i++){
        digitalWrite(SER,1);
        digitalWrite(SRCLK,0);
        digitalWrite(SRCLK,1);
        digitalWrite(SRCLK,0);
    }
    digitalWrite(RCLK,0);
    digitalWrite(RCLK,1);
    digitalWrite(RCLK,0);

    digitalWrite(ASRCLK, 0);
    digitalWrite(ASRCLK, 1);
    digitalWrite(ASRCLK, 0);

    digitalWrite(ARCLK, 0);
    digitalWrite(ARCLK, 1);
    digitalWrite(ARCLK, 0);
    //digitalWrite(12,0);
    //digitalWrite(12,1);
    //digitalWrite(12,0);

    //digitalWrite(11,0);
    //digitalWrite(11,1);
    //digitalWrite(11,0);
    delay(1);
}

```

Posterior a ello, se define una serie de funciones, entre las cuales se encuentra la función verificación, con la cual se inician a realizar pruebas con el objetivo de entender y comprobar las ideas que se tienen sobre el funcionamiento del circuito dependiendo de los datos que se ingresen.

```

const int SER = 3; //Puerto digital 3 Puerto serial
const int SRCLK = 4; //Puerto digital 4 Puerto registro de desplazamiento
const int RCLK = 5; //Puerto digital 5 Puerto registro de salida
const int ASRCLK = 6; //Puerto digital 6 Puerto de registro de desplazamiento co
const int ARCLK = 7; //Puerto digital 7 Puerto de registro de salida controlador
char *ptr_user=NULL;
char user[8];
int i=0;

void subida(int);
void salida_datos(short, short, int, short, bool);

```

```

void verificacion();

void setup()
{
    Serial.begin(9600);
    for(unsigned short ini=3; ini<=7; ini++){
        pinMode(ini,OUTPUT);
        digitalWrite(ini,LOW);
    }
    Serial.write("Bienvenido/a. \Ingrese 1 para verificar LEDs\n");
    Serial.write("Ingrese 2 para mostrar un unico patron de LEDs\n");
    Serial.write("Ingrese 3 para mostrar una serie de patrones de LEDs\n");
    verificacion();
}

void loop()
{
}

void salida_datos(short dato, int clock, short itera_limit, bool imprimir){
    for(unsigned short i=0;i<=itera_limit; i++){
        if(dato!=2) digitalWrite(SER,dato);
        subida(clock);
        if(imprimir==1) subida(clock+1);
    }
}

void subida(int clock){
    digitalWrite(clock,0);
    digitalWrite(clock,1);
    digitalWrite(clock,0);
}

void verificacion(){
    salida_datos(1,SRCLK,7,1);
    salida_datos(2,ASRCLK,7,1);
}

```

El próximo avance corresponde al desarrollo de código basado en el esquema básico hecho (Se puede ver en Figura 7). Debido a que inicialmente se planteó incorrectamente el sistema de referencia las pruebas iniciales no fueron exitosas, al final se completa la creación de un código de pruebas para demostrar la funcionalidad del programa para mostrar en la matriz de LEDs el patrón ingresado por el usuario, usando ceros y unos, fila por fila. Además, se hacen comentarios para tener mejor claridad de la funcionalidad de cada sección del código.

```

const int SER = 3; //Puerto digital 3 Puerto serial
const int SRCLK = 4; //Puerto digital 4 Puerto registro de desplazamiento
const int RCLK = 5; //Puerto digital 5 Puerto registro de salida
const int ASRCLK = 6; //Puerto digital 6 Puerto de registro de desplazamiento co

```

```

const int ARCLK = 7; //Puerto digital 7 Puerto de registro de salida controlador
int contador = 0;
int *ptr_cont=NULL;

//Prototipos Funciones
void subida(int);
void salida_datos(short, short, int, short, bool);
void verificacion();

void setup()
{
    Serial.begin(9600);
    for(unsigned short ini=3; ini<=7; ini++){
        pinMode(ini,OUTPUT);
        digitalWrite(ini,LOW);
    }
    Serial.write("Bienvenido/a. Ingrese 1 para verificar LEDs\n");
    Serial.write("Ingrese 2 para mostrar un unico patron de LEDs\n");
    Serial.write("Ingrese 3 para mostrar una serie de patrones de LEDs\n");
    Serial.println("");
}

void loop()
{
    if(Serial.available()>0){
        const int length = 8;
        char fila[length] = {0};
        int user = Serial.readBytes(fila, length);
        for(int i=0; i<=7; i++){
            Serial.print(fila[i]);
            salida_datos(*(fila+i)-48,SRCLK,0,0); //readBytes lee caracteres, por tabl
        }
        Serial.println("");
        salida_datos(2,RCLK,0,0);
        salida_datos(2,ASRCLK,0,0);
        ptr_cont=&contador;
        *ptr_cont= *ptr_cont + 1;
        if(*ptr_cont==8) {
            Serial.println(" Asi se ve la matriz de LEDs desde el monitor en serie, don
            salida_datos(2,ARCLK,0,0);
            *ptr_cont=0;
        }
    }
}

//Definicion Funciones de Respaldo
void salida_datos(short dato, int clock, short itera_limit, bool imprimir){

```

```

/*Dato=0 o 1, 2 para no pasar datos por el SER del Integrado Master
Clock=Registro de reloj
Itera_limit=Veces que se repite en columnas seguidas
Imprimir = 1 solo si se quiere activar el reloj de salida*/
for(unsigned short i=0;i<=itera_limit; i++){
    if(dato!=2) digitalWrite(SER,dato);
    subida(clock);
    if(imprimir==1) subida(clock+1);
}
}
void subida(int clock){ //Funcion para flanco de subida
    digitalWrite(clock,0);
    digitalWrite(clock,1);
    digitalWrite(clock,0);
}
void verificacion(){ //Funcion Encendido Todos los LEDs
    salida_datos(1,SRCLK,7,1);
    salida_datos(2,ASRCLK,7,1);
}

```

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

En un inicio se plantea que el algoritmo requiere considerar el uso de componentes de modo que se establezca un sistema de coordenadas con filas y columnas manejable por medio del uso eficiente del circuito integrado (CI) 74HC595. Para ello, se establece como consideración para la implementación de este sistema el uso de 2 circuitos integrados 74HC595 donde uno de ellos se usa para controlar las filas de la matriz de LEDs y el otro para las columnas, así como el uso de 5 puertos digitales del Arduino: 1 de ellos se encuentra conectado al pin SER tanto del circuito integrado de las filas (CIF) como el de las columnas (CIC); y los 4 restantes para controlar por aparte los pines asignados a los relojes de registro de desplazamiento y a los relojes de registro de salida de cada CI, es decir, 1 puerto digital se asigna a cada pin. El día martes 20 de abril de 2021 se replantea esta consideración con base en el replantamiento del circuito hecho, teniendo como consideración para la implementación del algoritmo que el circuito usa 9 integrados de la forma explicada en la sección de análisis del problema (2), se debe tener en cuenta que el algoritmo debe centrarse en el control del puerto digital conectado al pin SER del Integrado Master y el uso en el momento adecuado del código de los pines SRCLK y RCLK, principalmente con mayor cuidado en los Integrados Branch. En cuanto a la evolución del algoritmo, en primer lugar se desarrollan las pruebas de encender todos los LEDs, cuyo código corresponde a los dos primeros escrito en la sección 4 con base en el siguiente planteamiento: Primero se asegura que se ingresen 8 datos (Específicamente

unos) a través del puerto serial SER del primer 74HC595 de manera automática al ejecutar el código en Tinkercad de modo que, a través del uso del SRCLK, se encuentren en la salida adecuada y son enviados usando el RCLK a cada uno de los SER de los otros 8 integrados, que son ingresados a cada 74HC595 con el uso del ASRCLK. Como en este caso, las salidas imprimen todas 1 para encender todos los LEDs, se usa el ASRCLK y el ARCLK al tiempo y se verifica que no hay un cambio significativo al hacer esto. Posterior a la verificación del funcionamiento de la función verificacion(), se busca crear un algoritmo en el cual se permita al usuario digitar fila a fila lo que desea que se muestre en la matriz de LEDs y a la vez el usuario pueda ver un resultado final de cómo queda la matriz de LEDs mostrando en el monitor serial un arreglo de 8x8 donde se visualicen las combinaciones de ceros y unos digitadas por el usuario. Para establecer las

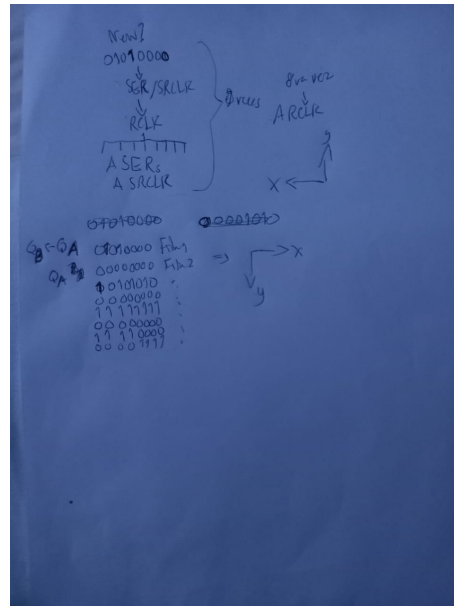


Figura 7: Esquema básico en borrador de procesos a seguir en el código

coordenadas que hay que tener en consideración y el sistema de referencia a tomar de acuerdo a cómo se ingresan los datos. En el desarrollo del algoritmo se hace un esquema básico, como se muestra en la imagen superior, de lo que se quiere hacer en código, viendo a su vez a través de un ejemplo cuál sería el sistema de referencia final que se debe tomar para el desarrollo de pruebas con código. Nuestro último problema en el desarrollo, el cual no pudimos resolver a pesar de la implementación de comandos para usar memoria dinámica fue el hecho de usar un arreglo en el cual se exigía una gran cantidad de memoria para guardar los patrones que el usuario ingresase, desafortunadamente no pudimos completar esta opción y por lo cual fue descartada. Sin embargo, aunque la función está incompleta, sus parámetros de entrada, que corresponden al número

de patrones que el usuario desea ingresar y el intervalo de tiempo entre ellos, pueden ser ingresados con normalidad haciendo uso de memoria dinámica. Así mismo se hizo uso de memoria dinámica con aquellas variables que consideramos que únicamente iban a utilizarse para el desarrollo de una función en específico.

```

const int SER = 3; //Puerto digital 3 Puerto serial
const int SRCLK = 4; //Puerto digital 4 Puerto registro de desplazamiento
const int RCLK = 5; //Puerto digital 5 Puerto registro de salida
const int ASRCLK = 6; //Puerto digital 6 Puerto de registro de desplazamiento co
const int ARCLK = 7; //Puerto digital 7 Puerto de registro de salida controlador

//Prototipos Funciones
void subida(int);
void salida_datos(short, short, int, short, bool);
void verificacion();
void imagen();
void apagado();
void publik(int, float);
void instrucciones();

void setup()
{
    Serial.begin(9600);
    for(unsigned short ini=3; ini<=7; ini++){
        pinMode(ini,OUTPUT);
        digitalWrite(ini,LOW);
    }
    Serial.write(" Bienvenido/a. \Ingrese \1\para\ verificar \LEDs\n" );
    Serial.write(" Ingrese \2\para\ mostrar \un\ unico \patron\ de \LEDs\n" );
    Serial.write(" Ingrese \3\para\ mostrar \una\ serie\ de\ patrones\ de\ LEDs\n\en\ un\ int
    Serial.println("");
}

void loop()
{
    if(Serial.available()>0){
        int numero = Serial.parseInt();
        if(numero==1){
            Serial.println(" Verificacion \de\ Encendido \de\ LEDs\ (10 \segundos)" );
            verificacion();
            delay(10000);
            Serial.println(" Apagado \de\ LEDs" );
            apagado();
        }
        else if(numero==2){
            Serial.println(" Matriz \de\ leds \para\ un\ unico \patron" );

```

```

        imagen();
    }
    else if(numero==3){
        int *ptr_contar=NULL;
        ptr_contar = new int;
        *ptr_contar = 0;
        float *ptr_time=NULL;
        ptr_time = new float;
        *ptr_time = 0;
        bool verificar_ingreso = false;
        Serial.println("Matriz para serie de patrones en un tiempo determinado");
        Serial.println("Ingrese por favor el numero de patrones que desea, ingresar");
        while(verificar_ingreso==false){
            delay(500);
            if(Serial.available()>0){
                *ptr_contar = Serial.parseInt();
                verificar_ingreso=true;
            }
        }
        verificar_ingreso=false;
        Serial.println("Ingrese por favor el intervalo de tiempo entre cada patron");
        Serial.println("En caso de ingresar un numero con decimales, separe la par");
        while(verificar_ingreso==false){
            delay(500);
            if(Serial.available()>0){
                *ptr_time = Serial.parseFloat();
                verificar_ingreso=true;
            }
        }
        verificar_ingreso=false;
        publik(*ptr_contar, *ptr_time);
        delete [] ptr_contar;
        delete [] ptr_time;
    }
    else{
        Serial.println("Numero inexistente.");
    }
    Serial.println("Si desea. Ingrese otro numero de los dados como opcion al in");
}
}

```

*//Definicion Funciones de Respaldo*

```

void salida_datos(short dato, int clock, short itera_limit, bool imprimir){
    /*Dato=0 o 1, 2 para no pasar datos por el SER del Integrado Master
    Clock=Registro de reloj
    Itera_limit=Veces que se repite en columnas seguidas

```

```

    Imprimir = 1 solo si se quiere activar el reloj de salida*/
    for(unsigned short i=0;i<=itera_limit; i++){
        if(dato!=2) digitalWrite(SER,dato);
        subida(clock);
        if(imprimir==1) subida(clock+1);
    }
}
void subida(int clock){ //Funcion para flanco de subida
    digitalWrite(clock,0);
    digitalWrite(clock,1);
    digitalWrite(clock,0);
}
void verificacion(){ //Funcion Encendido Todos los LEDs
    salida_datos(1,SRCLK,7,1);
    salida_datos(2,ASRCLK,7,1);
}
void apagado(){ //Funcion Apagado Todos los LEDs
    salida_datos(0,SRCLK,7,1);
    salida_datos(2,ASRCLK,7,1);
}
void imagen ()
{
    instrucciones();
    int *ptr_cont=NULL;
    ptr_cont=new int;
    *ptr_cont=0;
    while(*ptr_cont<=7){
        delay(500);
        if(Serial.available(>0){
            const int length = 8;
            char fila[length] = {0};
            int user = Serial.readBytes(fila ,length);
            for(int i=0;i<=7;i++){
                Serial.print(fila[i]);
                salida_datos(*(fila+i)-48,SRCLK,0,0); //readBytes lee caracteres, por
            }
            Serial.println("");
            salida_datos(2,RCLK,0,0);
            salida_datos(2,ASRCLK,0,0);
            *ptr_cont= *ptr_cont + 1;
            if(*ptr_cont==8) {
                Serial.println("Asi se ve la matriz de LEDs desde el monitor en serie");
                salida_datos(2,ARCLK,0,0);
            }
        }
    }
}
}

```



```

    delete [] ptr_cont;
}
void publik(int num_patrones, float tiempo){
    /*instrucciones();
    unsigned long int *ptr_patrones = new unsigned long int[8*num_patrones]{0};
    int i = 0;
    while(i<=(num_patrones*8)){
        if(Serial.available()>0){
            ptr_patrones[i] = Serial.parseInt();
            Serial.println(ptr_patrones[i]);
            i=i+1;
        }
    }*/
}

void instrucciones(){
    Serial.println("Por_favor._Ingrese_filas_a_filas_del_patron_a_mostrar");
    Serial.println("Para_ello,_use_ceros_y_unos_para_indicar_que_el_LED_debe_estar");
    Serial.println("Recuerde_que_va_a_ingresar_la_primera_filas_la_cual_se_ubicar");
    Serial.println("La_segunda_filas_se_encuentra_debajo_de_ella,_y_asi_sucesivamen");
    Serial.println("Para_ingresar_cada_filas_asegurese_de_haber_escrito_8_numeros");
    Serial.println("Ejemplo_de_filas_con_los_4_leds_que_estan_mas_a_la_izquierda_en");
    Serial.println("Una_vez_escritas_las_8_filas,_en_el_monitor_serial_se_muestra");
}

```

## 6. Algoritmo implementado

Esta sección se ubica de última debido a problemas en la subida de imágenes al pdf, pues éstas se posicionaban de manera errónea en el documento. En cuanto al desarrollo del algoritmo corresponde al siguiente:

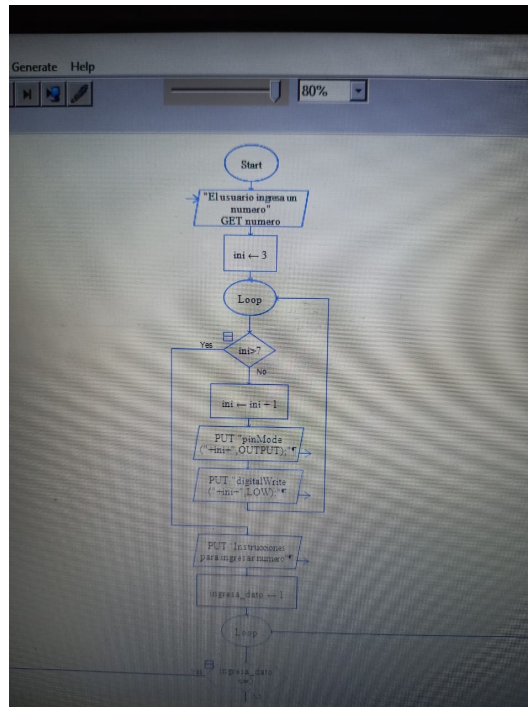


Figura 8: Inicio del algoritmo

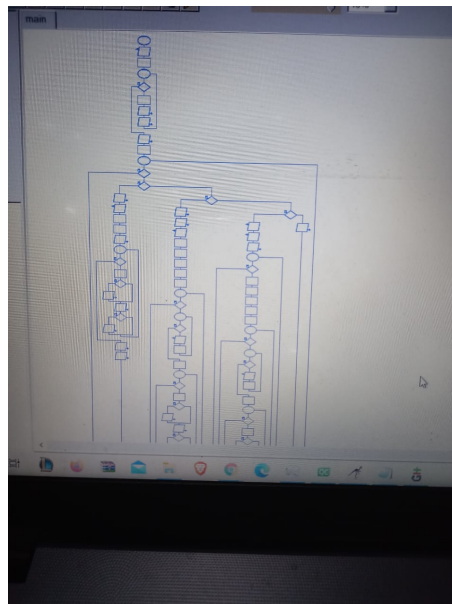


Figura 9: Diagrama del algoritmo 1

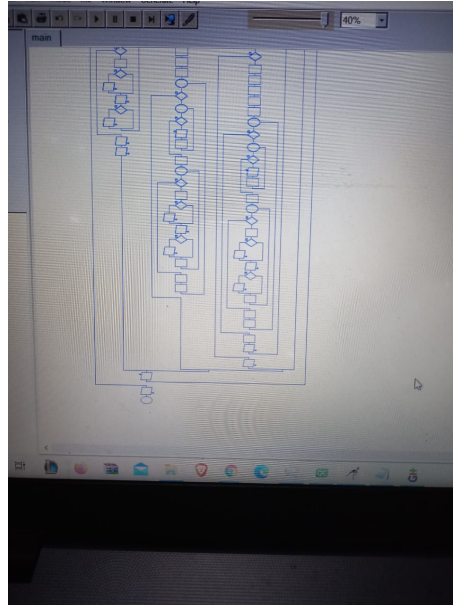


Figura 10: Diagrama del algoritmo 2

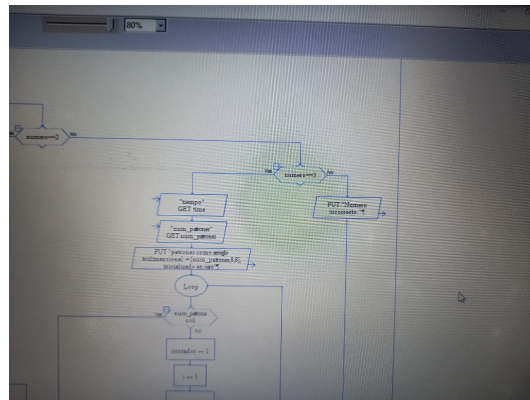


Figura 11: Avances Algoritmo 1

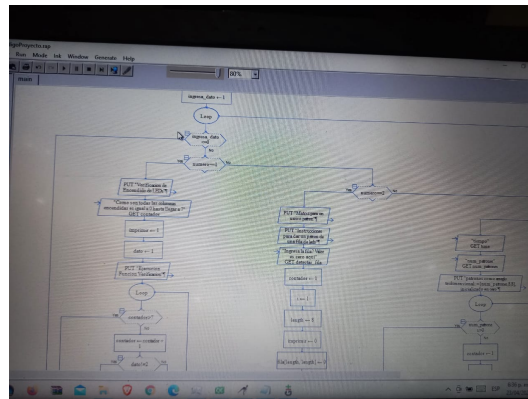


Figura 12: Avances Algoritmo 2

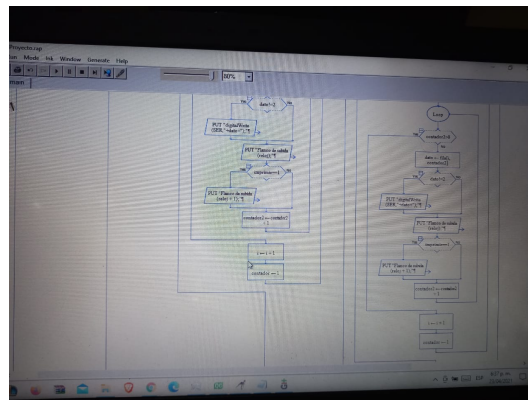


Figura 13: Avances Algoritmo 3

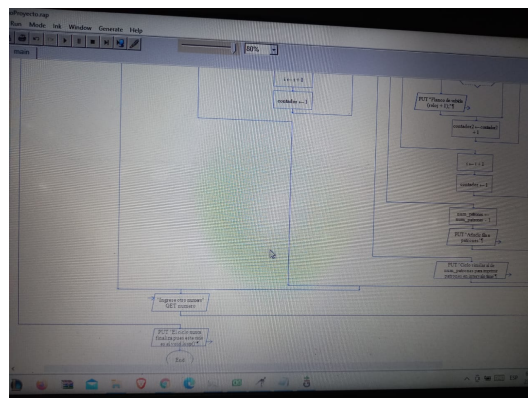


Figura 14: Avances Algoritmo 4

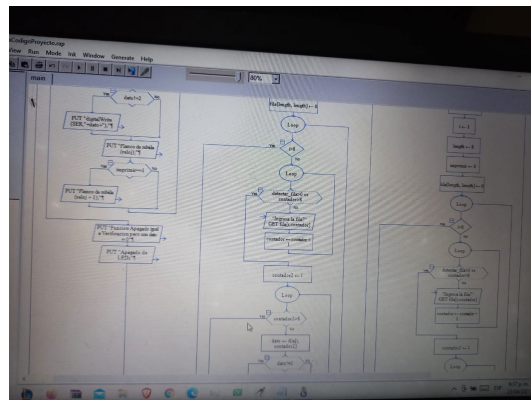


Figura 15: Avances Algoritmo 5