Informa2 S.A.S

Parcial 1

Juan Andrés Urbiñez Gómez
Marcela Flórez Orellano
Ferney Mejía Pérez

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

Índice

1. Sección de contenido

1.1. Análisis del problema

Análisis del problema: Para la solución del problema se hizo control sobre una matriz de LEDS de 8x8 a partir de las conexiones entre ocho integrados 74HC595 a la protoboard, la configuración de los puertos de Arduino y el uso de 64 resistencias. Al momento de conectar los integrados, descubrimos que podíamos usarlos de tal manera que la salida de uno fuera la entrada de otro y así formar un "puente" entre ellos.

Alternativa: Como idea inicial se crearon las funciones CLOCK1 y CLOCK2 con las que se trata de representar el patrón ingresado por su correspondiente en binario, de esta manera almacenar en CLOCK1 los LEDS que permiten el encendido y dejar pasar el resto de los datos, mediante CLOCK2; también se pensó en funciones que reciban los registros de entrada en pro a la acumulación de los datos ingresados y generar un arreglo con la posición de los LEDS, en el que se pudieran encender al gusto del usuario mediante el recorrido de un ciclo "while".

1.2. Esquema

- Crear e implementar las funciones de reloj necesarias para el manejo del integrado 74HC595.
- Creación de la función "minimenu" dirigida al usuario, para que elija entre la verificación del Hardware, el ingreso de un solo patrón, generar una animación o salir del programa.
- Creación de la función "verificación" la cual confirma la correcta conexión del Hardware
- Creación de la función "imagen" para almacenar el patrón ingresado por el usuario.
- Creación de la función "leerarreglo" que se encarga de leer los datos ingresados y permite encender los LEDS deseados, a partir de las funciones de CLOCK1, CLOCK2.
- Creación de la función "publick"para generar la animación a partir de un número de patrones y tiempo entre ellos definidos por el usuario.
- Creación de la función "manual" dirigida al usuario, para un mayor entendimiento del funcionamiento básico del programa.
- Creación de la función "apagarLEDS" que se encarga de apagar los LEDS, mediante un arreglo de 64 elementos iguales a cero.

- Creación de la función “apagarLEDS2d” que se encarga de apagar los LEDS, mediante un arreglo de n-patrones por 64 elementos igualados a cero.

1.3. Algoritmo implementado

Al iniciar el programa se mostrará la función “minimenu”, la cual gestiona las diferentes funciones desarrolladas en el código, por lo que le brinda al usuario distintos tipos de entrada, como el chequeo del funcionamiento de leds, el ingreso de patrones por el monitor serial ya sea para presentar uno solo o para generar una animación con ellos.

// Con la ayuda de la memoria dinámica, se crea un puntero que almacena un arreglo de 64 elementos.
// Se configuran los puertos digitales 11,12 y 13 del Arduino como “output”.

- Para la opción 1:

  La Función “verificacion()” se encarga de revisar que todos los leds enciendan correctamente.

  - Función CLOCK1() // Activación del reloj de la primera etapa de los registros de entrada.
  - Función CLOCK2() // Activación del reloj de la segunda etapa de registros de salida.

  Función “apagarLEDS()”

- Para la opción 2:

  //Se define un puntero para almacenar lo retornado por la función “imagen()”.

  La función “imagen()” le permite al usuario decidir entre encender un LED o no; en el caso afirmativo, le pide ingresar por el puerto serial el número del LED a encender, esta función se encarga de capturar dichas entradas y formar un arreglo con los valores ingresados cambiados a 1.

  La función “leerarreglo()” es de tipo puntero y recibe como parámetro de entrada un arreglo. Su funcionalidad se enfoca en el encendido de los LEDS mediante un while, la indexación de un contador en el arreglo recibido y el uso de las funciones CLOCK1() y CLOCK2().

  La función “reset()” es de tipo puntero y recibe un puntero, se encarga de convertir todos los elementos del arreglo ingresado en ceros y retorna el arreglo.

//Se elimina el arreglo.

- Para la opción 3:

  La función "publik()" solicita la cantidad de patrones que se desea formar y el tiempo de espera entre cada uno para producir la animación, para ello se hace uso de la memoria dinámica.

  //Se define "arreglo2D", un puntero de punteros que crea memoria dinámica dentro de la profundidad del puntero externo.

  //Se hace uso de tres ciclos for anidados:

  - El primero se encarga de crear los espacios para los elementos.
  - El segundo almacena la secuencia de LEDS por cada número de patrones ingresados.
  - El tercero indexa la posición y se cambia por los patrones ingresados

  //Se "limpia" el patrón que se va introduciendo para así evitar encender los mismos LEDS en el siguiente patrón.

  //Se hace uso de un ciclo while, el cual indexa cada posición del patrón de unos y ceros generados.

  //Con un condicional se identifica si cada posición contiene un uno o no, para finalmente usar las funciones CLOCK1 y CLOCK2.

  //Al terminar la función, se elimina el arreglo de la memoria.

- Para la opción 4:

  Función "apagarLEDS()"

1.4. Problemas presentados

Para el desarrollo de la actividad se presentaron diversos inconvenientes relacionados con el programa de Tinkercad, de los cuales resaltamos la demora al momento de cargar el programa, se eliminaban elementos del hardware y del software de forma abitaría, o simplemente los componentes no se podían ni mover ni eliminar.

Presentamos inconvenientes al establecer la función "publick" dado al poco entendimiento sobre matrices bidimensionales, memoria dinámica y el paso de las mismas a Tinkercad.

1.5. Evolucion del algoritmo

Para llevar una cronología en la evolución del algoritmo se creó una carpeta llamada “avances Tinkercad” en la que se encuentra toda la información al respecto.