INSTITUTO POLITECNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERIA Y TECNOLOGIAS AVANZADAS

TELEFONIA

2TM9

PRACTICA 3: MATRIZ DE CONMUTACION DE 5X6

AGUILAR CEJA LUIS ANGEL

Objetivo: realizar una matriz de conmutación de 5x6, pasando por las partes de diseño e implementación física de la matriz.

Para la realización de esta practica se utilizo el programa proteus para hacer una simulación y comprobar el correcto funcionamiento de la matriz, en este caso la matriz asignada tiene control a la salida en la primera etapa, mientras que en la segunda etapa se realizo un control a la salida como se puede apreciar en las imágenes del diseño de la matriz, para este caso se observo que se utilizaban muchos switch para la realización de la matriz, lo cual podría ser optimizado al momento de la realización física.

Circuito lógico:

Como es observable, este circuito requería de una gran cantidad de switch para su correcto funcionamiento, además de una gran cantidad de compuertas, con lo cual se busco como objetivo reducir este numero de componentes para el circuito final.

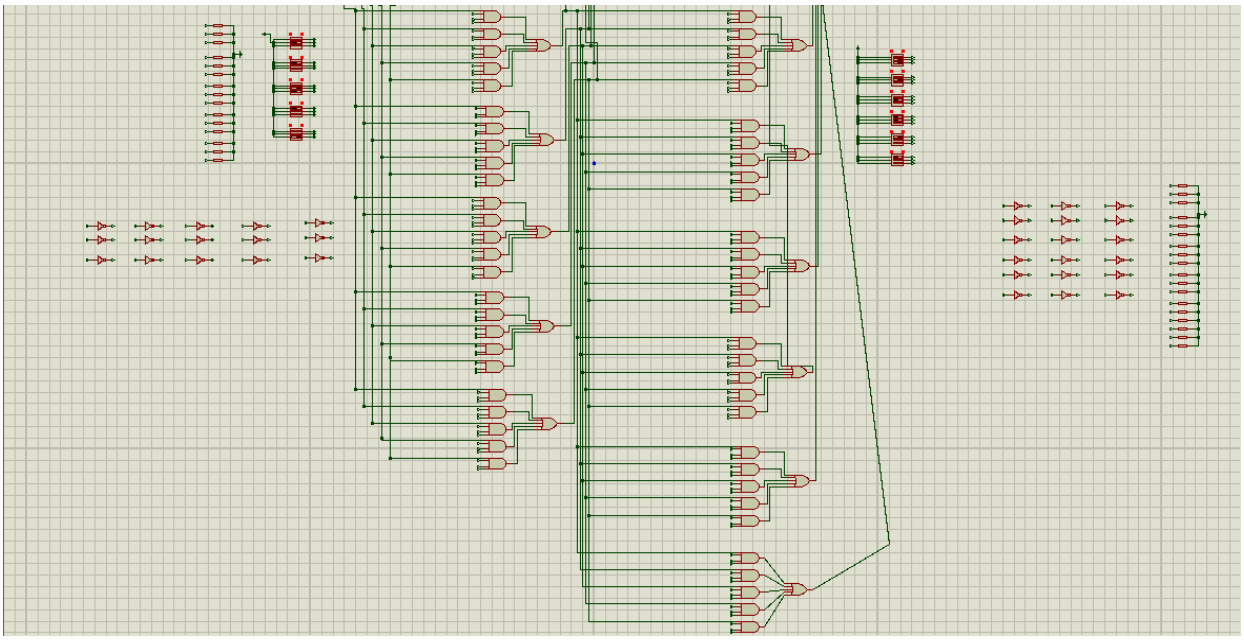


Ilustración 1.esquema de la matriz

Para la primera etapa se hizo uso de 5 controles a la salida, cada control con tres hilos para poder realizar su función correctamente y gestionar las 5 salidas de forma efectiva.

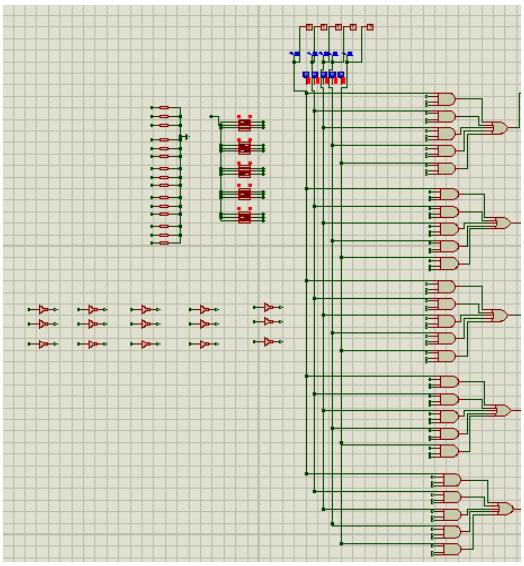


Ilustración 2.1a etapa de la matriz (control a la salida)

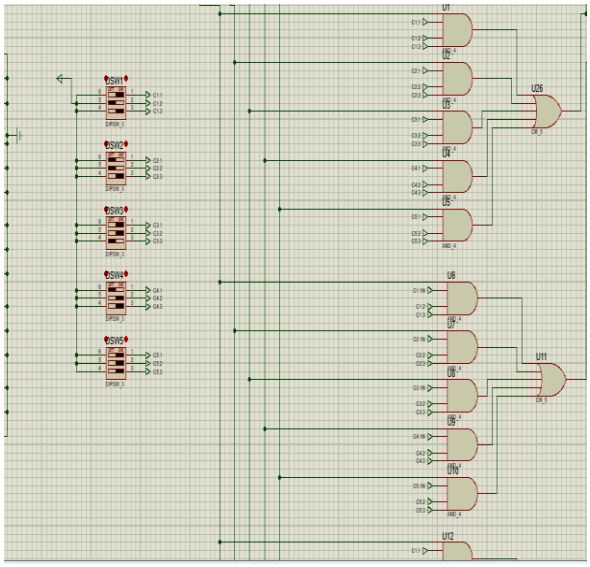


Ilustración 3.acercamiento a la 1ª etapa de la matriz

Para la realización de la segunda etapa se requirió del uso de 6 controles para poder obtener 6 salidas respectivamente, aunque el numero de hilos de cada control no cambio debido a que con esos 3 hilos había combinaciones suficientes para la gestión de todas las entradas.

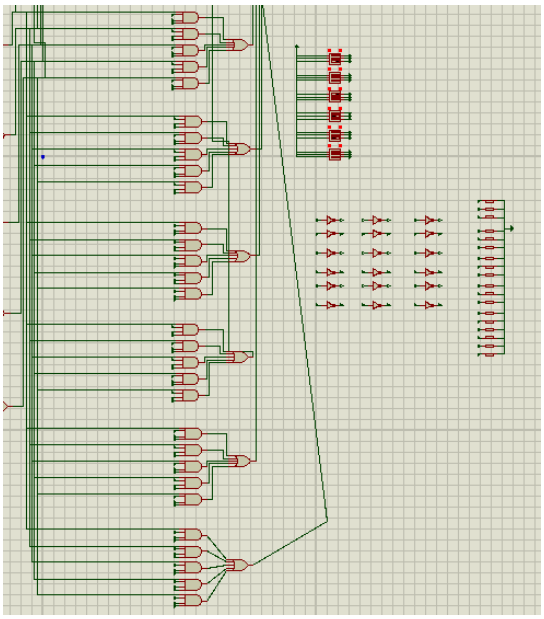


Ilustración 4.2da etapa de la matriz(control a la entrada)

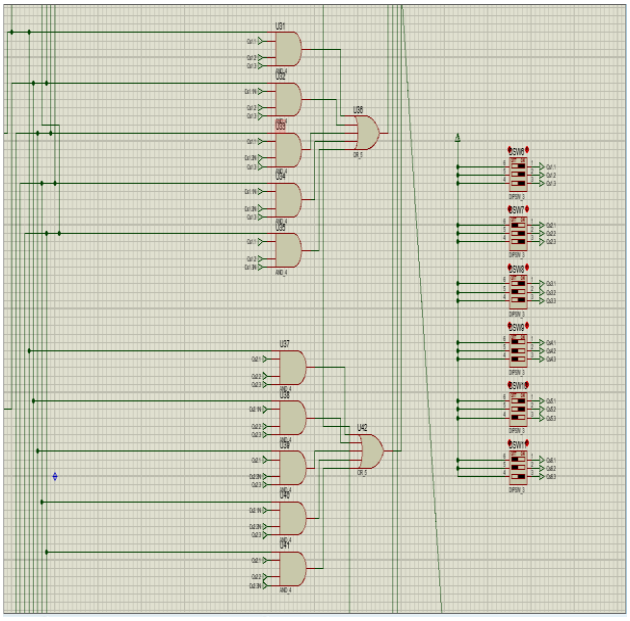
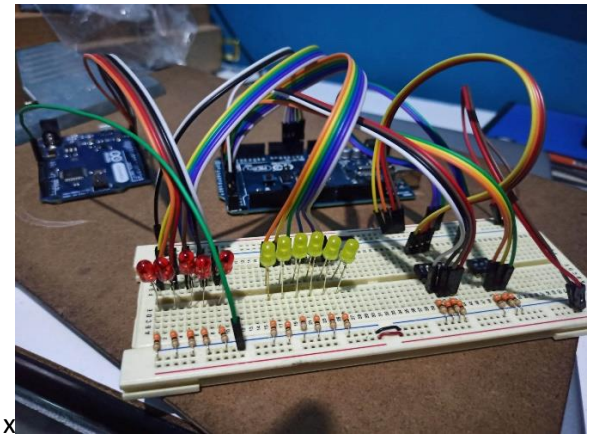


Ilustración 5.acercamiento a la segunda etapa de la matriz

Circuito eléctrico:

En cuanto al circuito final, se realizó la siguiente conexión que consta de la utilización de dos arduinos, uno para la generación de las señales a utilizar como entradas y otro para hacer la tarea de la matriz, las señales en cuestión se vieron reflejadas visualmente mediante leds y se logro reducir el número de switch a 7,tres para la selección de la señal de entrada que se quiere transmitir, otros 3 para señalar por que salida se va a transmitir la entrada seleccionada.



x

Ilustración 6.circuito eléctrico funcional

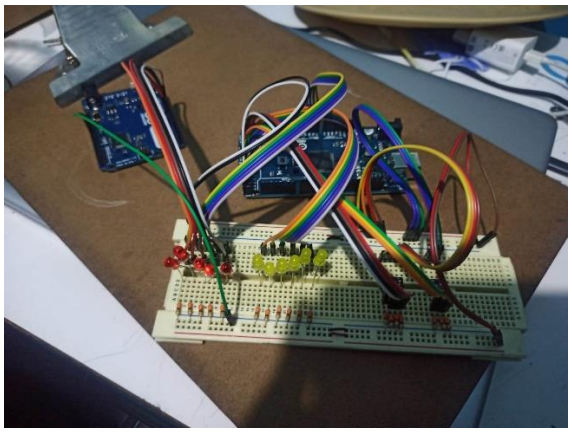


Ilustración 7.circuito eléctrico funcional

Diagrama eléctrico:

Para una mejor vista de las conexiones, se elaboró una simulación en proteus, mostrando las conexiones físicas realizadas en el circuito final, empleando un Arduino para la generación de las 5 señales de entrada que se utilizaron para comprobar el correcto funcionamiento de la matriz.

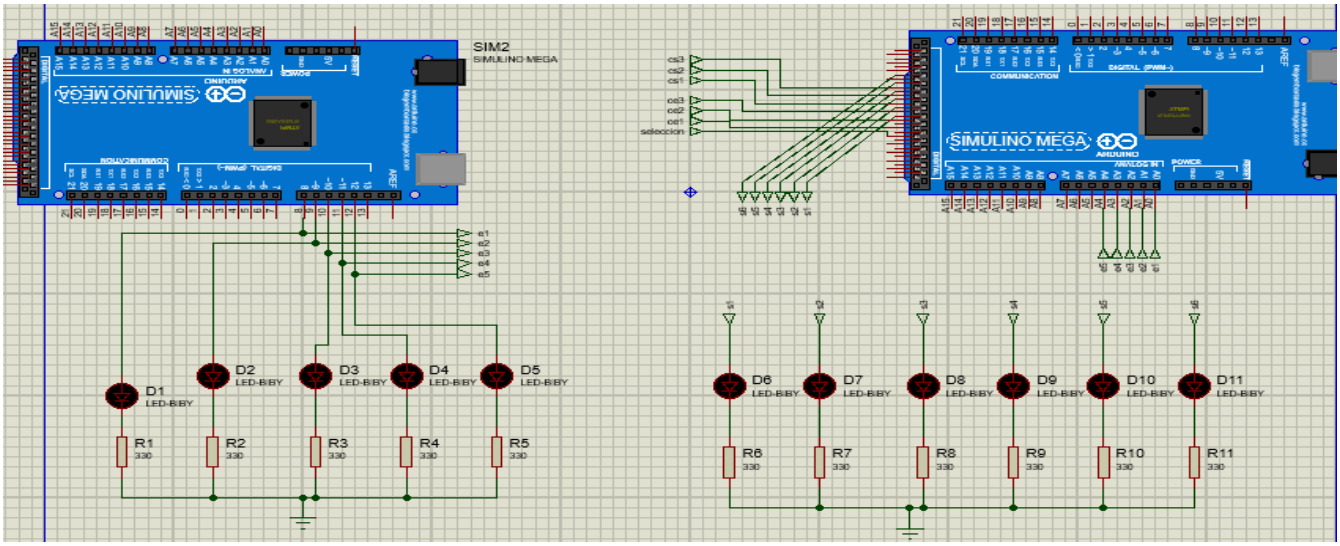


Ilustración 8.Diagrama de conexiones de la matriz

Diagrama de flujo:

Al momento de utilizar dispositivos programables, podemos utilizar la lógica para simplificar el hardware en vez de utilizar tantos componentes como se vio anteriormente en las primeras ilustraciones, para este caso particular se empleó la siguiente lógica para lograr el mismo resultado de la matriz de conmutación, optimizando de manera efectiva el hardware.

Para explicar de manera breve lo que se realizo en el programa se realizo un diagrama de flujo, el cual explica cómo funciona la lógica de este.

Básicamente, el programa está en un bucle, el cual consiste en la lectura de las 5 entradas y no es hasta que se activa el switch de selección, que entra en acción una función de configuración, la cual lee los valores de los demás switch, que son configurados antes de la activación del switch de selección para configurar que entrada se busca transmitir (combinación de variables ce1,ce2 y ce3) y porque salida se busca transmitir(combinación de variables cs1,cs2,cs3).

Dependiendo de esta configuración, las variables auxiliares (desde aux1 hasta aux6) tendrán distintos valores, teniendo el valor de 1 si se quiere transmitir la primera entrada, valor de 2 si se transmitirá la segunda y así sucesivamente hasta el valor de 5,teniendo el caso especial donde su valor es cero, dónde se busca que se deje de transmitir por dicha salida.

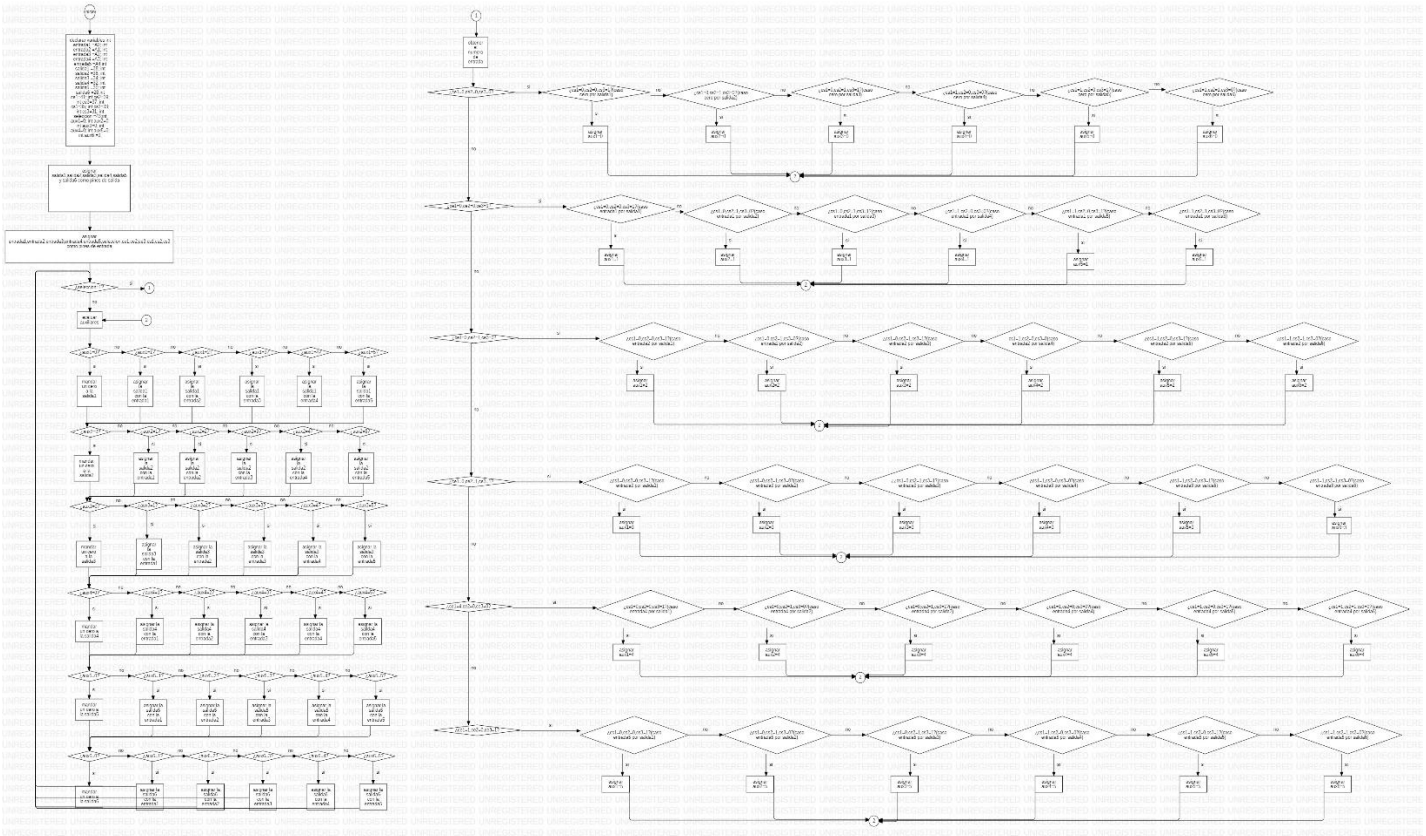
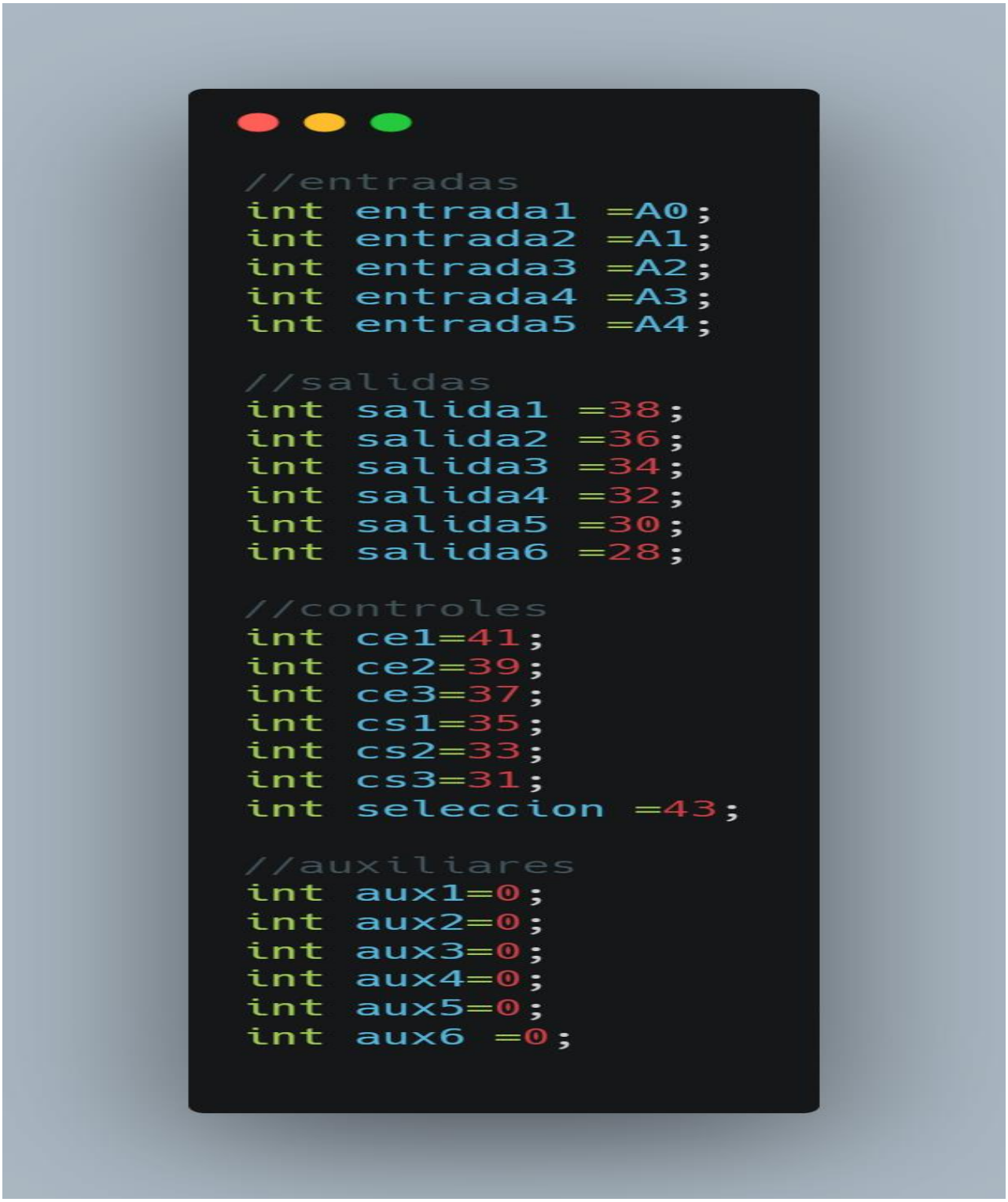


Ilustración 9.Diagrama de flujo del programa realizado

Código del programa:



```
//entradas
int entrada1 =A0;
int entrada2 =A1;
int entrada3 =A2;
int entrada4 =A3;
int entrada5 =A4;

//salidas
int salida1 =38;
int salida2 =36;
int salida3 =34;
int salida4 =32;
int salida5 =30;
int salida6 =28;

//controles
int ce1=41;
int ce2=39;
int ce3=37;
int cs1=35;
int cs2=33;
int cs3=31;
int seleccion =43;

//auxiliares
int aux1=0;
int aux2=0;
int aux3=0;
int aux4=0;
int aux5=0;
int aux6 =0;
```

Ilustración 10.programa realizado (declaración de variables)

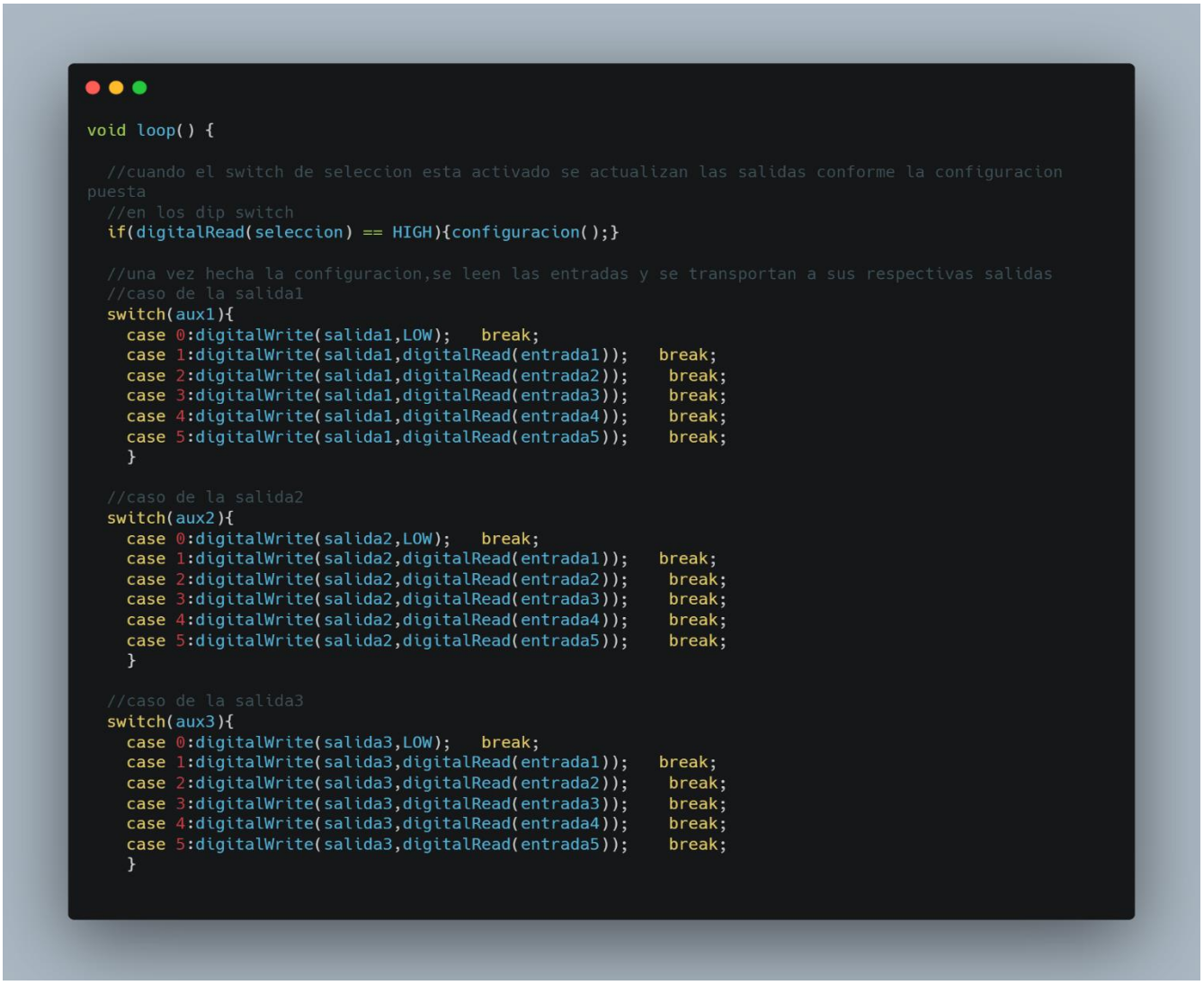
```
void setup() {

  pinMode(salida1, OUTPUT);
  pinMode(salida2, OUTPUT);
  pinMode(salida3, OUTPUT);
  pinMode(salida4, OUTPUT);
  pinMode(salida5, OUTPUT);
  pinMode(salida6, OUTPUT);

  pinMode(entrada1, INPUT);
  pinMode(entrada2, INPUT);
  pinMode(entrada3, INPUT);
  pinMode(entrada4, INPUT);
  pinMode(entrada5, INPUT);
  pinMode(seleccion, INPUT);
  pinMode(ce1, INPUT);
  pinMode(ce2, INPUT);
  pinMode(ce3, INPUT);
  pinMode(cs1, INPUT);
  pinMode(cs2, INPUT);
  pinMode(cs3, INPUT);

}
```

Ilustración 11.programa realizado (declaración de entradas y salidas)



```
void loop() {

  //cuando el switch de seleccion esta activado se actualizan las salidas conforme la configuracion
puesta
  //en los dip switch
  if(digitalRead(seleccion) == HIGH){configuracion();}

  //una vez hecha la configuracion,se leen las entradas y se transportan a sus respectivas salidas
  //caso de la salida1
  switch(aux1){
    case 0:digitalWrite(salida1,LOW);   break;
    case 1:digitalWrite(salida1,digitalRead(entrada1));   break;
    case 2:digitalWrite(salida1,digitalRead(entrada2));    break;
    case 3:digitalWrite(salida1,digitalRead(entrada3));    break;
    case 4:digitalWrite(salida1,digitalRead(entrada4));    break;
    case 5:digitalWrite(salida1,digitalRead(entrada5));    break;
    }

  //caso de la salida2
  switch(aux2){
    case 0:digitalWrite(salida2,LOW);   break;
    case 1:digitalWrite(salida2,digitalRead(entrada1));   break;
    case 2:digitalWrite(salida2,digitalRead(entrada2));    break;
    case 3:digitalWrite(salida2,digitalRead(entrada3));    break;
    case 4:digitalWrite(salida2,digitalRead(entrada4));    break;
    case 5:digitalWrite(salida2,digitalRead(entrada5));    break;
    }

  //caso de la salida3
  switch(aux3){
    case 0:digitalWrite(salida3,LOW);   break;
    case 1:digitalWrite(salida3,digitalRead(entrada1));   break;
    case 2:digitalWrite(salida3,digitalRead(entrada2));    break;
    case 3:digitalWrite(salida3,digitalRead(entrada3));    break;
    case 4:digitalWrite(salida3,digitalRead(entrada4));    break;
    case 5:digitalWrite(salida3,digitalRead(entrada5));    break;
    }
```

Ilustración 12.parte principal del programa

```
   //caso de la salida4
  switch(aux4){
    case 0:digitalWrite(salida4,LOW);   break;
    case 1:digitalWrite(salida4,digitalRead(entrada1));   break;
    case 2:digitalWrite(salida4,digitalRead(entrada2));    break;
    case 3:digitalWrite(salida4,digitalRead(entrada3));    break;
    case 4:digitalWrite(salida4,digitalRead(entrada4));    break;
    case 5:digitalWrite(salida4,digitalRead(entrada5));    break;
    }

  //caso de la salida5
  switch(aux5){
    case 0:digitalWrite(salida5,LOW);   break;
    case 1:digitalWrite(salida5,digitalRead(entrada1));   break;
    case 2:digitalWrite(salida5,digitalRead(entrada2));    break;
    case 3:digitalWrite(salida5,digitalRead(entrada3));    break;
    case 4:digitalWrite(salida5,digitalRead(entrada4));    break;
    case 5:digitalWrite(salida5,digitalRead(entrada5));    break;
    }

  //caso de la salida6
  switch(aux6){
    case 0:digitalWrite(salida6,LOW);   break;
    case 1:digitalWrite(salida6,digitalRead(entrada1));   break;
    case 2:digitalWrite(salida6,digitalRead(entrada2));    break;
    case 3:digitalWrite(salida6,digitalRead(entrada3));    break;
    case 4:digitalWrite(salida6,digitalRead(entrada4));    break;
    case 5:digitalWrite(salida6,digitalRead(entrada5));    break;
    }
}
```

Ilustración 13.parte principal del programa 2

```
void configuracion(){
  //evaluamos cual entrada se va a transmitir
  //caso para la entrada 0, es decir, cuando se quiere desactivar una salida
  //la logica del siguiente if se repite para todos los casos,variando unicamente los valores de la
entrada seleccionada
      if(digitalRead(ce1) == 0 && digitalRead(ce2) == 0 && digitalRead(ce3) == 0){

          //este caso es para que no salga nada por la salida seleccionada
          //evaluamos en donde se quiere la salida

          if(digitalRead(cs1)==0 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
              //sacamos por salida 1
              //el numero del aux indica la salida por la que se saldra,mientras que el
              //valor asignado indica la señal de entrada que se va a transmitir
              aux1 = 0;
            }

          else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
            //sacamos por salida 2
              aux2 = 0;
            }

          else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 1){
            //sacamos por salida 3
              aux3 = 0;
            }

          else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 0){
            //sacamos por salida 4
              aux4 = 0;
            }

          else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
            //sacamos por salida 5
              aux5 = 0;
            }

          else if(digitalRead(cs1)==1 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
            //sacamos por salida 6
              aux6 = 0;
            }
        }
```

Ilustración 14.funcion para la configuración de las entradas/salidas

```
//caso para la entrada 1
if(digitalRead(ce1) == 0 && digitalRead(ce2) == 0 && digitalRead(ce3) == 1){
    //evaluamos en donde se quiere la salida
    if(digitalRead(cs1)==0 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
        //sacamos por salida 1
        aux1 = 1;
      }

    else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
      //sacamos por salida 2
        aux2 = 1;
      }

    else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 1){
      //sacamos por salida 3
        aux3 = 1;
      }

    else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 0){
      //sacamos por salida 4
        aux4 = 1;
      }

    else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
      //sacamos por salida 5
        aux5 = 1;
      }

    else if(digitalRead(cs1)==1 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
      //sacamos por salida 6
        aux6 = 1;
      }
  }

  // caso para la entrada 2
  if(digitalRead(ce1) == 0 && digitalRead(ce2) == 1 && digitalRead(ce3) == 0){
    //evaluamos en donde se quiere la salida
    if(digitalRead(cs1)==0 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
        //sacamos por salida 1
        aux1 = 2;
      }

    else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
      //sacamos por salida 2
        aux2 = 2;
      }

    else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 1){
      //sacamos por salida 3
        aux3 = 2;
      }

    else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 0){
      //sacamos por salida 4
        aux4 = 2;
      }

    else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
      //sacamos por salida 5
        aux5 = 2;
      }

    else if(digitalRead(cs1)==1 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
      //sacamos por salida 6
        aux6 = 2;
      }
  }
```

Ilustración 15.funcion para la configuración de las entradas/salidas 2

```
    // caso para la entrada 3
if(digitalRead(ce1) == 0 && digitalRead(ce2) == 1 && digitalRead(ce3) == 1){
  //evaluamos en donde se quiere la salida
  if(digitalRead(cs1)==0 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
      //sacamos por salida 1
      aux1 = 3;
    }

  else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
    //sacamos por salida 2
      aux2 = 3;
    }

  else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 1){
    //sacamos por salida 3
      aux3 = 3;
    }

  else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 0){
    //sacamos por salida 4
      aux4 = 3;
    }

  else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
    //sacamos por salida 5
      aux5 = 3;
    }

  else if(digitalRead(cs1)==1 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
    //sacamos por salida 6
      aux6 = 3;
    }
}

//caso para la entrada 4
if(digitalRead(ce1) == 1 && digitalRead(ce2) == 0 && digitalRead(ce3) == 0){
  //evaluamos en donde se quiere la salida
  if(digitalRead(cs1)==0 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
      //sacamos por salida 1
      aux1 = 4;
    }

  else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
    //sacamos por salida 2
      aux2 = 4;
    }

  else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 1){
    //sacamos por salida 3
      aux3 = 4;
    }

  else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 0){
    //sacamos por salida 4
      aux4 = 4;
    }

  else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
    //sacamos por salida 5
      aux5 = 4;
    }

  else if(digitalRead(cs1)==1 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
    //sacamos por salida 6
      aux6 = 4;            }
}
```

Ilustración 16.funcion para la configuración de las entradas/salidas 3

```
        // caso para la entrada 5
        if(digitalRead(ce1) == 1 && digitalRead(ce2) == 0 && digitalRead(ce3) == 1){
          //evaluamos en donde se quiere la salida
          if(digitalRead(cs1)==0 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
              //sacamos por salida 1
              aux1 = 5;
            }

          else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
            //sacamos por salida 2
              aux2 = 5;
            }

          else if(digitalRead(cs1)==0 && digitalRead(cs2) ==1 && digitalRead(cs3) == 1){
            //sacamos por salida 3
              aux3 = 5;
            }

          else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 0){
            //sacamos por salida 4
              aux4 = 5;
            }

          else if(digitalRead(cs1)==1 && digitalRead(cs2) ==0 && digitalRead(cs3) == 1){
            //sacamos por salida 5
              aux5 = 5;
            }

          else if(digitalRead(cs1)==1 && digitalRead(cs2) ==1 && digitalRead(cs3) == 0){
            //sacamos por salida 6
              aux6 = 5;
            }
        }
    }
```

Ilustración 17.funcion para la configuración de las entradas/salidas 4

Para resumir este programa, podemos decir que la función de configuración hace la función de la primera parte de la matriz de conmutación, pues esta se encarga de llevar la entra seleccionada hasta un punto intermedio ,mientras que la segunda parte de la matriz se observa en el loop principal, pues ahí es donde con la configuración obtenida de esta función ,se transmite efectivamente la señal deseada sin interrumpir la transmisión de las otras salidas.

Con esto podemos concluir que existen varias formas de realizar una matriz de conmutación, como se pudo observar a lo largo de esta practica que se vieron dos formas, tanto la realizada con compuertas lógicas como lo es también la versión programada para disminuir el hardware, obteniendo en ambos casos la resolución exitosa de la asignación que se dio, logrando transmitir distintas señales al mismo tiempo por distintas salidas, cosa que es fundamental para las comunicaciones.