# Trabajo Entregable 1 - Introducción al Diseño Lógico

Participantes:
- Federico Goncalves - 03866/5
- Joaquín Guzmán - 03751/4
- Tomás Gamarra - 03852/8

# Objetivo

La finalidad de este trabajo es, encontrar y explicar el circuito equivalente que brinde la siguiente funcionalidad. Partiendo de 16 entradas lógicas provenientes de 16 sensores que están activas en bajo, cuya prioridad está dada por su número de sensor. Los sensores están etiquetados como $S_0$, de menor prioridad, hasta el $S_{15}$, de máxima prioridad. Con estas entradas, dos display de 7 segmentos deben de mostrar el número del sensor con mayor prioridad activo. Por ende, tenemos 16 entradas lógicas que tendrán que pasar por un circuito que resulte en 4 salidas lógicas (4 bits que pueden representar $2^4$ = 16 números). Estas 4 salidas lógicas serán entradas de otro circuito, donde este último tendrá como objetivo convertir estas 4 entradas que representan un número entre 0 y 15, a 14 salidas lógicas. Estas 14 salidas representan la cantidad de entradas necesarias para dos display de 7 segmentos cada uno ( sin tener en cuenta el Dot-point).

# Marco Teórico

Para llegar a este circuito, primero debemos identificar las partes o “bloques” del circuito total. Podemos interpretar al primer circuito que toma las 16 entradas, y obtiene las 4 salidas, como un **codificador**. Luego el segundo circuito que toma estas 4 entradas y obtiene 14 salidas podríamos interpretarlo como un **decodificador**.
Un detalle no menor acerca del display de 7 segmentos, es que existen dos tipos de display de este estilo. Uno llamado Display de Ánodo común, y otro llamado Display de Cátodo común. Estos se diferencian en la forma en que las entradas se consideran activas. En el caso de Ánodo común sus entradas serán activas en bajo, mientras que en Cátodo común sus entradas serán activas en alto.

Como no se sabe que tipo de Display se va a utilizar, si de Ánodo común o de Cátodo común, se debe proveer una entrada adicional en el decodificador (KC/AC) que permite permutar entre ambas tecnologías a elección según el tipo de display utilizado.
El bit KC/AC valdrá 1 en el caso de ánodo común y valdrá 0 en el caso de cátodo común.

Por otro lado, si contamos con que el display mostrará el número del sensor activo con más prioridad, no tenemos forma de contemplar el caso en el que ningún sensor esté activo. Para esto, decidimos agregar a la salida del codificador un bit de validez, donde éste se considere activo en el momento en que ninguna entrada esté activa.

# Desarrollo

Para el desarrollo del diseño del circuito, se realizarán los siguientes pasos:

**1- Diagrama de bloques:** Se comenzará haciendo un diagrama de bloques del circuito, asignando a cada bloque las entradas, salidas y funcionalidades antes mencionadas en el marco teórico.

**2- Tablas de verdad:** A partir del diagrama de bloques realizaremos una tabla de verdad para cada componente del circuito según su lógica, esto será un primer paso para encontrar la función lógica a cada salida de la componente.

**3- Obtención de funciones lógicas:** Habiendo hecho la tabla de verdad de la componente se procederá a hallar la función lógica correspondiente a cada salida. Además, cuando sea conveniente, aplicaremos el método de mapas de Karnaugh para hallar funciones lógicas en su mínima expresión.

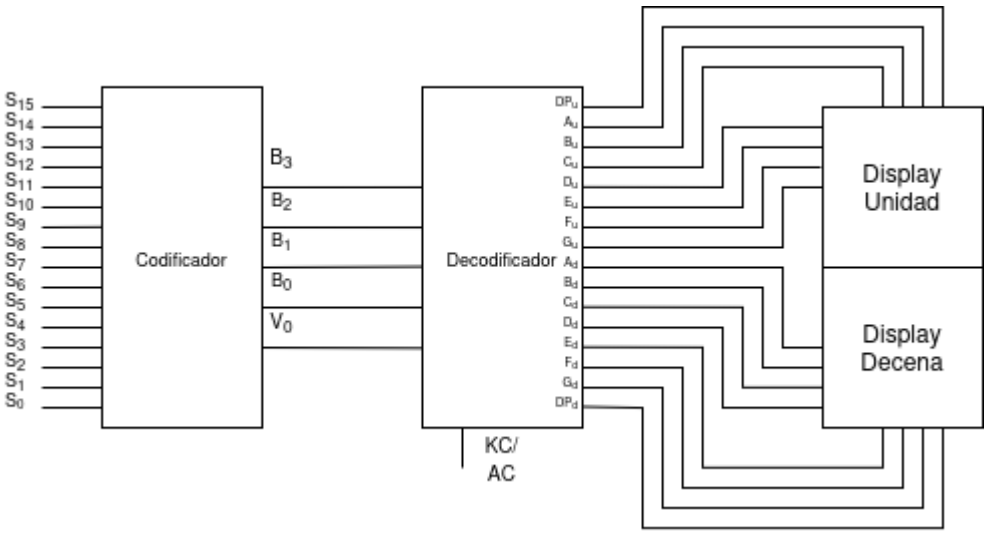
**4- Implementación de funciones lógicas:** Teniendo las funciones lógicas de cada salida de cada componente, las implementaremos con compuertas lógicas y hallaremos el circuito equivalente a la componente representada por un bloque.

**5- Implementación del circuito:** Finalmente, haremos un diagrama de conexiones del circuito diseñado, reemplazando los bloques por sus equivalentes en compuertas lógicas.

**6- Cálculo de parámetros del circuito:** Una vez hallado el circuito, determinaremos el valor que las resistencias de cada led han de tener para que la tensión y corriente se mantengan dentro del rango de funcionamiento del mismo.

# 1- Diagrama de Bloques

Al momento de definir el diagrama de bloques, con lo mencionado en el marco teórico podemos diagramarlo de la siguiente manera:




Podemos ver que cada entrada del Codificador está dada por $S_x$ , con ***x*** entre 0-15 siendo los números más altos los de mayor prioridad. Luego cada salida la tomamos como $B_i$ , siendo ***i*** de 0-3 representando los 4 bits que representan los valores de ***x***. El bit de validez se tomará como activo (válido) en alto.

En el Decodificador utilizamos $B_i$ como entradas, y sus respectivas salidas están dadas por los 7 segmentos del display. Estos 7 segmentos se diferencian desde A-G, utilizando subíndices ***u*** ó ***d*** para diferenciar entre el display de unidades y el display de decenas.
En este mismo decodificador contamos con la entrada lógica pensada para diferenciar entre los dos tipos de display anteriormente mencionado, llamado KC/AC.

# 2- Tablas de Verdad

Procedemos a realizar las tablas de verdad para las componentes del circuito **Codificador** y **Decodificador**.

## Codificador

Por la lógica misma del **codificador**, este ha de recibir la señal de activo de uno o varios sensores y según el que más prioridad tenga dar una salida, que codificada en BSS representa el número de sensor de mayor prioridad que está activo. Es teniendo esto en mente que interpretamos las entradas para cada caso y colocamos una salida acorde.

Al tener 16 entradas, una tabla de verdad con todas las posibles combinaciones de entrada daría una tabla de $2^{16}$ filas, lo que hace que la construcción de esa tabla sea totalmente inviable. Analizando el comportamiento lógico del **codificador** podemos darnos cuenta que en los casos en el que un sensor esté activo y los sensores de mayor prioridad no lo estén, el valor del resto de los sensores de menor prioridad le será indiferente a la salida. Hallamos así una forma de simplificar la tabla. Si colocamos una **X** cuando los valores de los sensores no afectan la salida, podemos llegar a la tabla de abajo.

| $S_0$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $S_5$ | $S_6$ | $S_7$ | $S_8$ | $S_9$ | $S_{10}$ | $S_{11}$ | $S_{12}$ | $S_{13}$ | $S_{14}$ | $S_{15}$ | $B_3$ | $B_2$ | $B_1$ | $B_0$ | $V_0$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 |
| X | X | X | X | X | X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| X | X | X | X | X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| X | X | X | X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| X | X | X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| X | X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| X | X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| X | X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| X | X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| X | X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| X | X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| X | X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| X | X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| X | X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

## Decodificador

La lógica del **decodificador** es generar a partir de las 4 entradas (que en BSS son el número de sensor activo de mayor prioridad) 14 salidas de manera que correspondan a 2 displays de 7 segmentos que muestran el número de sensor activo. De estas 14 salidas, 7 corresponderán al display de las unidades y 7 al de las decenas.

Explicada la lógica del **codificador**, realizamos la tabla de verdad correspondiente.

*(Nota: En el diagrama de bloques aparece como entradas adicionales KC/AC y $V_0$ , el cómo afectan estas entradas a la salida del decodificador se tratará más adelante en la sección de "Obtención de funciones lógicas".)*

| $B_0$ | $B_1$ | $B_2$ | $B_3$ | $A_u$ | $B_u$ | $C_u$ | $D_u$ | $E_u$ | $F_u$ | $G_u$ | $A_d$ | $B_d$ | $C_d$ | $D_d$ | $E_d$ | $F_d$ | $G_d$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |

# 3- Obtención de Funciones Lógicas

## Codificador

A partir de la tabla de verdad, vamos a analizar cada bit de salida por separado.

Para encontrar la función lógica del bit $B_0$, vemos que este responde con un 1 al momento en el que los sensores de número impar se encienden. Por lo que podríamos realizar un primer acercamiento a la función lógica del bit $B_0$ de la forma siguiente:

$$B_0 = \overline{S_1} + \overline{S_3} + \overline{S_5} + \overline{S_7} + \overline{S_9} + \overline{S_{11}} + \overline{S_{13}} + \overline{S_{15}}$$

Como podemos ver, esta función no es válida para nuestro objetivo, ya que esta función lógica no tiene en cuenta que al momento de encenderse cualquier otro sensor que no esté en la función, el bit $B_0$ debería de apagarse. Ejemplo claro: Al momento de mostrar en display el número 2 *($0_3 0_2 1_1 0_0$ en binario con $B_0 = 0$ y $B_1 = 1$)*, debería de estar activo el sensor $S_2$. Pero si seguimos la función lógica de $B_0$ anterior, si se activa el sensor $S_0$ entonces el número que se muestra será 3 *($0_3 0_2 1_1 1_0$ en binario con $B_0 = 1$ y $B_1 = 1$).* Esto es erróneo, y es debido a que la función lógica no tiene en cuenta la prioridad de los sensores que tienen número mayor.
Para resolverlo, a cada entrada activa que ya tenemos, debemos aplicarle la condición de prioridad.

$$B_0 = \overline{S_1}\,(S_2 S_3 S_4 S_5 S_6 S_7 S_8 S_9 S_{10} S_{11} S_{12} S_{13} S_{14} S_{15}) + \overline{S_3}\,(S_4 S_5 S_6 S_7 S_8 S_9 S_{10} S_{11} S_{12} S_{13} S_{14} S_{15}) + \overline{S_5}\,(S_6 S_7 S_8 S_9 S_{10} S_{11} S_{12} S_{13} S_{14} S_{15}) + \overline{S_7}\,(S_8 S_9 S_{10} S_{11} S_{12} S_{13} S_{14} S_{15}) + \overline{S_9}\,(S_{10} S_{11} S_{12} S_{13} S_{14} S_{15}) + \overline{S_{11}}\,(S_{12} S_{13} S_{14} S_{15}) + \overline{S_{13}}\,(S_{15}) + \overline{S_{15}}$$

Podríamos simplificar esta función, eliminando los casos donde los sensores impares están desactivados, ya que estos están contemplados dentro de la función para los casos que están encendidos o apagados.

$$B_0 = \overline{S_1}\,(S_2 S_4 S_6 S_8 S_{10} S_{12} S_{14}) + \overline{S_3}\,(S_4 S_6 S_8 S_{10} S_{12} S_{14}) + \overline{S_5}\,(S_6 S_8 S_{10} S_{12} S_{14}) + \overline{S_7}\,(S_8 S_{10} S_{12} S_{14}) + \overline{S_9}\,(S_{10} S_{12} S_{14}) + \overline{S_{11}}\,(S_{12} S_{14}) + \overline{S_{13}}\,S_{14} + \overline{S_{15}}$$

Siguiendo la misma lógica, planteamos las funciones lógicas para cada salida a partir de la tabla de verdad, luego se agrega la condición de prioridad de entradas, y finalmente se simplifica si es el caso..

$$B_1 = (\overline{S_2} + \overline{S_3})\,(S_4 S_5 S_8 S_9 S_{12} S_{13}) + (\overline{S_6} + \overline{S_7})\,(S_8 S_9 S_{12} S_{13}) + (\overline{S_{10}} + \overline{S_{11}})\,(S_{12} S_{13}) + \overline{S_{14}} + \overline{S_{15}}$$

$$B_2 = (\overline{S_4} + \overline{S_5} + \overline{S_6} + \overline{S_7})\,(S_8 S_9 S_{10} S_{11}) + \overline{S_{12}} + \overline{S_{13}} + \overline{S_{14}} + \overline{S_{15}}$$

$$B_3 = \overline{S_{12}} + \overline{S_{13}} + \overline{S_{14}} + \overline{S_{15}}$$

En el caso del bit de validez $V_0$, éste nos permite diferenciar entre el escenario en el que se encuentra encendido únicamente el sensor $S_0$ del escenario donde ningún sensor lo está. Si $V_0 = 0$ quiere decir que ningún sensor está encendido , entonces no debería mostrarse nada en los displays. En cambio si $V_0 = 1$ significa que algún sensor está activo. Por lo tanto esto nos lleva a la conclusión de que $V_0 = 0$ solo en el caso que todos los sensores estén

apagados , como la lógica de activación es negativa para los sensores entonces cuando todos esten High (1) deberia haber un circuito lógico que convierta eso en el resultado $V_0$ = 0. Esa es la compuerta NAND.

$$V_0 = \overline{(S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8 S_9 S_{10} S_{11} S_{12} S_{13} S_{14} S_{15})}$$

## Decodificador

Para hallar las funciones lógicas que representan a cada bloque dentro del decodificador, empleamos el método de Karnaugh por el cual podemos en base a la tabla de verdad de la función lógica obtener una expresión simplificada, la cual luego podemos transformar en su circuito lógico equivalente. A continuación mostraremos cómo quedan los K-Map para las 14 salidas que obtenemos del decodificador, así como también la función lógica que surge de ellos.

## Display de unidades

### Salida Au

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 1 | 1 | 1 | 0 |
| $\overline{B_0} \cdot B_1$ | 1 | 1 | 0 | 1 |
| $B_0 \cdot B_1$ | 1 | 0 | 1 | 1 |
| $B_0 \cdot \overline{B_1}$ | 0 | 1 | 1 | 1 |

$$Au = (B_2 \cdot B_0) + (B_3 \cdot \overline{B_1}) + (\overline{B_3} \cdot B_1) + (\overline{B_2} \cdot \overline{B_0})$$

Salida Bu

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 1 | 1 | 1 | 1 |
| $\overline{B_0} \cdot B_1$ | 1 | 1 | 1 | 0 |
| $B_0 \cdot B_1$ | 1 | 1 | 0 | 1 |
| $B_0 \cdot \overline{B_1}$ | 1 | 1 | 1 | 0 |

$$Bu = (B_3 \cdot (\overline{B_1} + \overline{B_0})) + (\overline{B_3} \cdot B_1 \cdot B_0) + \overline{B_2} + (\overline{B_1} \cdot \overline{B_0})$$

Salida Cu

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 1 | 1 | 0 | 1 |
| $\overline{B_0} \cdot B_1$ | 0 | 1 | 1 | 1 |
| $B_0 \cdot B_1$ | 1 | 1 | 1 | 1 |
| $B_0 \cdot \overline{B_1}$ | 1 | 1 | 1 | 1 |

$$Cu = (\overline{B_3} \cdot B_2) + (B_3 \cdot B_1) + B_0 + (\overline{B_2} \cdot \overline{B_1})$$

Salida Du

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 1 | 1 | 1 | 0 |
| $\overline{B_0} \cdot B_1$ | 1 | 1 | 0 | 1 |
| $B_0 \cdot B_1$ | 1 | 0 | 1 | 0 |
| $B_0 \cdot \overline{B_1}$ | 0 | 0 | 1 | 1 |

$$Du = ((B_3 \cdot B_2) \cdot (B_0 + \overline{B_1})) + ((\overline{B_3} \cdot B_1) \cdot (\overline{B_0} + \overline{B_2})) + (\overline{B_2} \cdot \overline{B_0}) + (B_2 \cdot \overline{B_1} \cdot B_0)$$

Salida Eu

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 1 | 1 | 1 | 0 |
| $\overline{B_0} \cdot B_1$ | 1 | 1 | 0 | 1 |
| $B_0 \cdot B_1$ | 0 | 0 | 0 | 0 |
| $B_0 \cdot \overline{B_1}$ | 0 | 0 | 0 | 0 |

$$Eu = \overline{B_0} \cdot ((\overline{B_3} \cdot B_1) + \overline{B_2} + (B_3 \cdot \overline{B_1}))$$

Salida Fu

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 1 | 1 | 0 | 1 |
| $\overline{B_0} \cdot B_1$ | 0 | 1 | 1 | 1 |
| $B_0 \cdot B_1$ | 0 | 0 | 1 | 0 |
| $B_0 \cdot \overline{B_1}$ | 0 | 1 | 0 | 1 |

$$Fu = ((B_3 \cdot B_1) \cdot (\overline{B_0} + B_2)) + (B_2 \cdot ((\overline{B_3} \cdot \overline{B_1}) + (B_1 \cdot \overline{B_0}))) + (\overline{B_2} \cdot \overline{B_1} \cdot \overline{B_0}) + (B_3 \cdot \overline{B_2} \cdot \overline{B_1})$$

Salida Gu

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 0 | 1 | 1 | 1 |
| $\overline{B_0} \cdot B_1$ | 1 | 0 | 1 | 1 |
| $B_0 \cdot B_1$ | 1 | 0 | 1 | 0 |
| $B_0 \cdot \overline{B_1}$ | 0 | 1 | 1 | 1 |

$$Gu = (B_2 \cdot (B_3 + \overline{B_0} + \overline{B_1})) + (\overline{B_3} \cdot \overline{B_2} \cdot B_1) + (B_3 \cdot \overline{B_1})$$

Display de decenas

Salida Ad

$$Ad = 0$$

Salida Bd

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 0 | 0 | 1 | 0 |
| $\overline{B_0} \cdot B_1$ | 0 | 1 | 1 | 0 |
| $B_0 \cdot B_1$ | 0 | 1 | 1 | 0 |
| $B_0 \cdot \overline{B_1}$ | 0 | 0 | 1 | 0 |

$$Bd = B_3 \cdot (B_2 + B_1)$$

Salida Cd

| | $\overline{B_2} \cdot \overline{B_3}$ | $\overline{B_2} \cdot B_3$ | $B_2 \cdot B_3$ | $B_2 \cdot \overline{B_3}$ |
|---|---|---|---|---|
| $\overline{B_0} \cdot \overline{B_1}$ | 0 | 0 | 1 | 0 |
| $\overline{B_0} \cdot B_1$ | 0 | 1 | 1 | 0 |
| $B_0 \cdot B_1$ | 0 | 1 | 1 | 0 |
| $B_0 \cdot \overline{B_1}$ | 0 | 0 | 1 | 0 |

$$Cd = B_3 \cdot (B_2 + B_1)$$

Salida Dd

$$Dd = 0$$

Salida Ed

$$Ed = 0$$

Salida Fd

$$Fd = 0$$

Salida Gd

$$Gd = 0$$

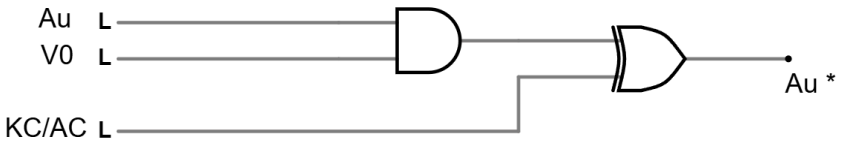
## Aclaraciones importantes

1. Circuito lógico para los casos donde la salida es 0 independientemente de la entrada : Para estos casos basta con no conectar ningún cable hacia los pines del display asociado , ya que siempre estarán apagados.

2. Como ya se vio en el desarrollo de los K-map para el decodificador solo utilizamos 4 entradas ( $B_0 B_1 B_2 B_3$ ) en vez de las 6 entradas que tiene ( $B_0 B_1 B_2 B_3 V_0$ KC/AC ). Esto es debido a que tanto la entrada $V_0$ como KC/AC agrega una lógica mucho más sencilla la cual nos permite quitarlos del desarrollo del K-map correspondiente. La lógica se basa en que cuando $V_0 = 0$ las salidas del decodificador tienen que mantener todos los segmentos apagados. Pero dependiendo de si el display está configurado como cátodo común o ánodo común las salidas deberán ser 1 o 0 respectivamente. Para cumplir con ambos requerimientos se implementa a la salida de cada bloque dentro del decodificador la función lógica AND con el bit $V_0$ , si el bit $V_0 = 1$ las salidas permanecen igual mientras que si $V_0 = 0$ las salidas se convierten a 0. Consecuentemente hace falta una lógica que convierta esas salidas en el valor necesario para que los segmentos estén apagados o encendidos dependiendo del bit KC/AC. Esa lógica es implementada mediante una compuerta XOR entre la salida (luego de haber pasado por el AND con $V_0$) y el bit KC/AC. De esta forma si KC/AC = 1 , al aplicarle el XOR se negara el valor lo cual tiene sentido ya que para ánodo común los segmentos se encienden con 0 (lógica negativa). En cambio cuando KC/AC = 0 , al aplicarle el XOR se mantiene invariante el valor lo cual hace sentido ya que para cátodo común los segmentos se encienden con 1 (lógica positiva).

La lógica descrita en 2. se hará previamente a enviar la salida final a cada pin de cada display , primero efectuando el AND con $V_0$ y luego el XOR con KC/AC.

A continuación una imagen que evidencia el comportamiento descrito , en este caso con un ejemplo para la salida Au, pero esto sucederá para cada una de las salidas a cada pin del display :
Au : Representa a la salida luego de pasar por la función lógica para la salida Au.
Au* : Representa la salida real que irá conectada al pin del display.

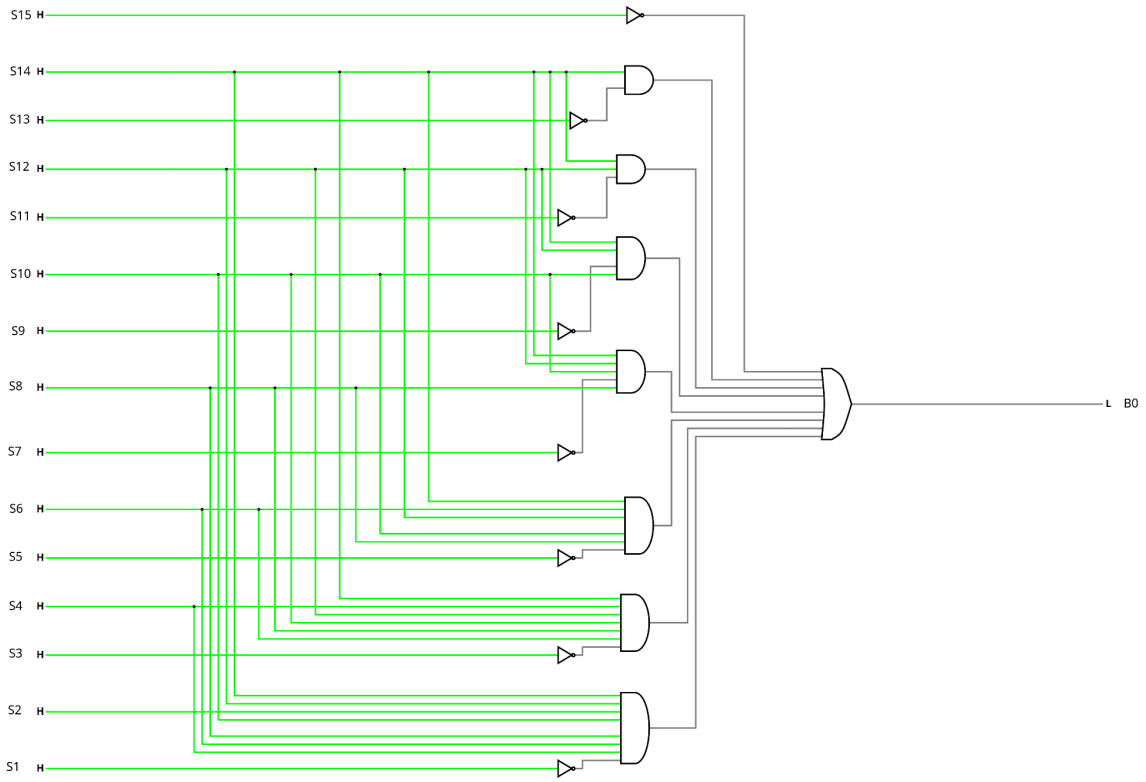



## 4- Implementación de funciones lógicas

En base a la obtención de las funciones lógicas que describimos anteriormente en la sección “Obtención de funciones lógicas” podremos armar los circuitos lógicos que representan a los distintos bloques que actúan dentro del codificador y decodificador. Cabe aclarar que las imágenes están por separado , pero que las entradas para cada uno de estos circuitos serían las mismas , para facilitar el entendimiento visual decidimos mostrarlas por separado. Además se incluye un link para cada imagen que hace referencia a una página de modelado de circuitos donde se puede interactuar con él y ver su funcionamiento para las distintas entradas.

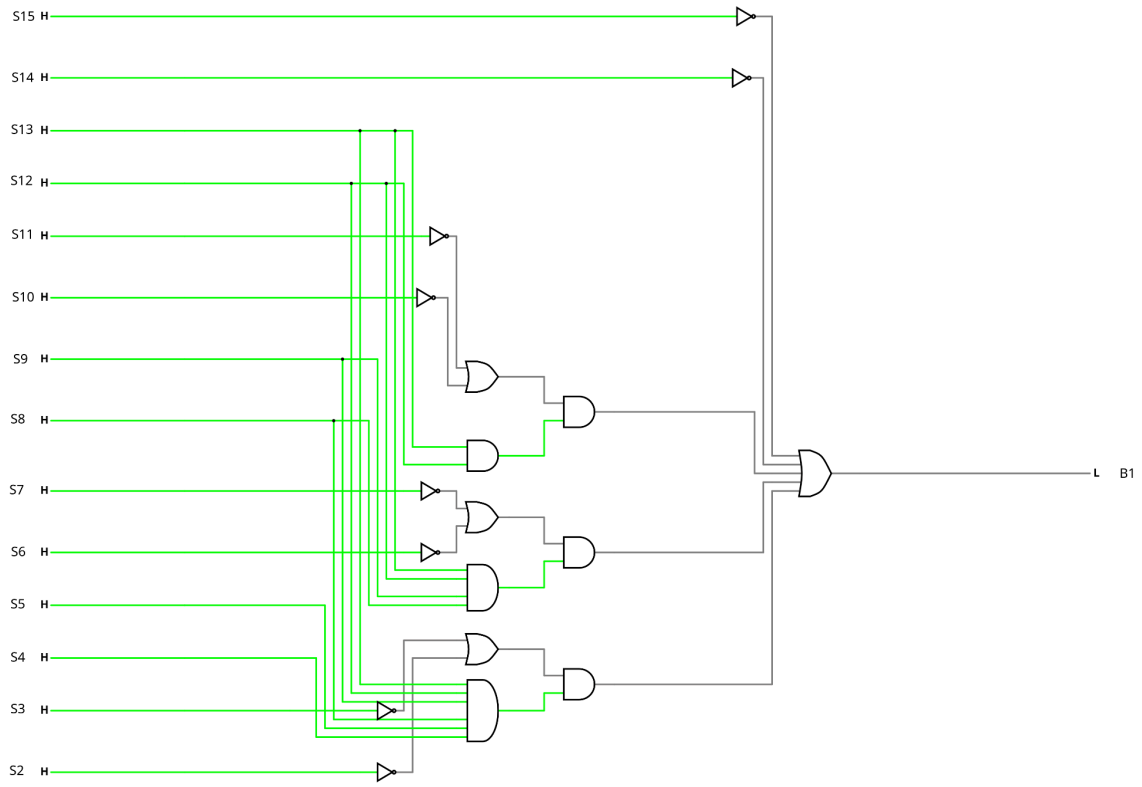
# Codificador

## Salida $B_0$



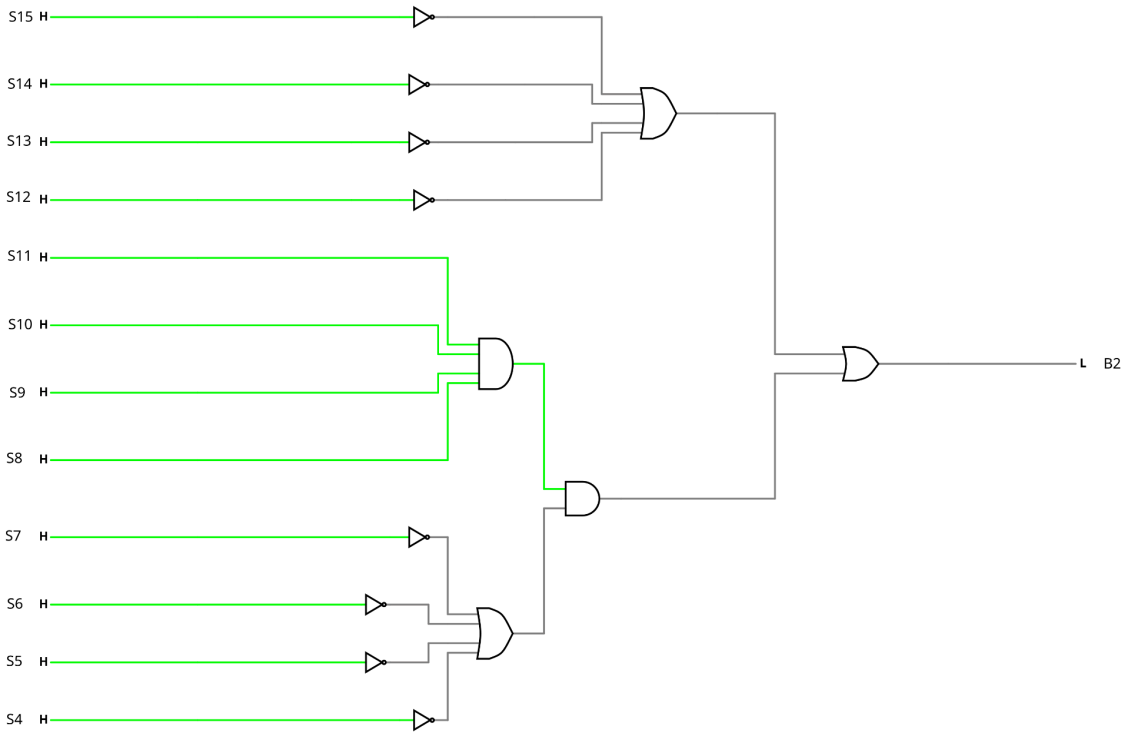

Link

## Salida $B_1$



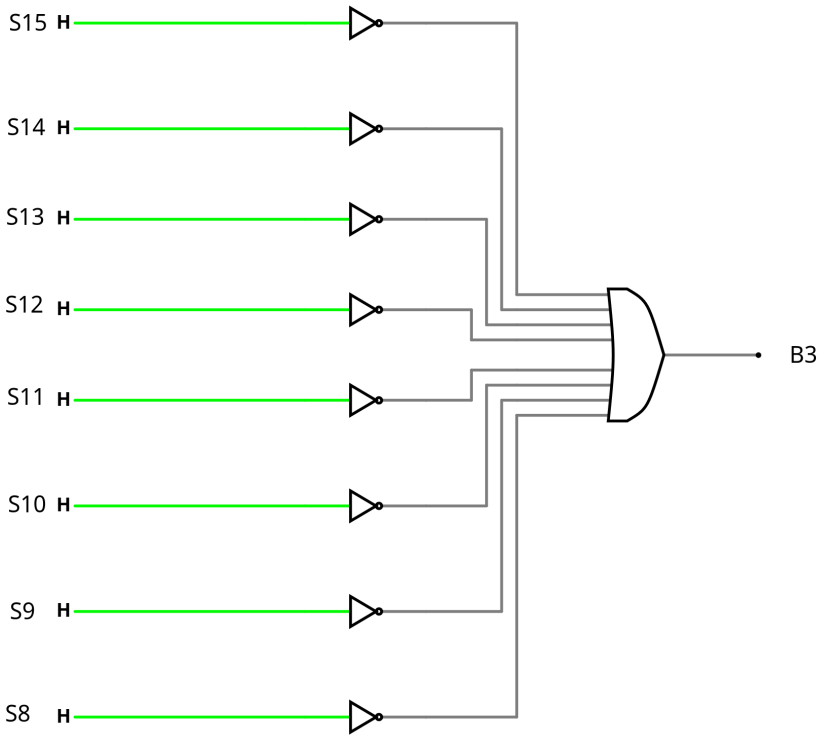

Link

## Salida $B_2$



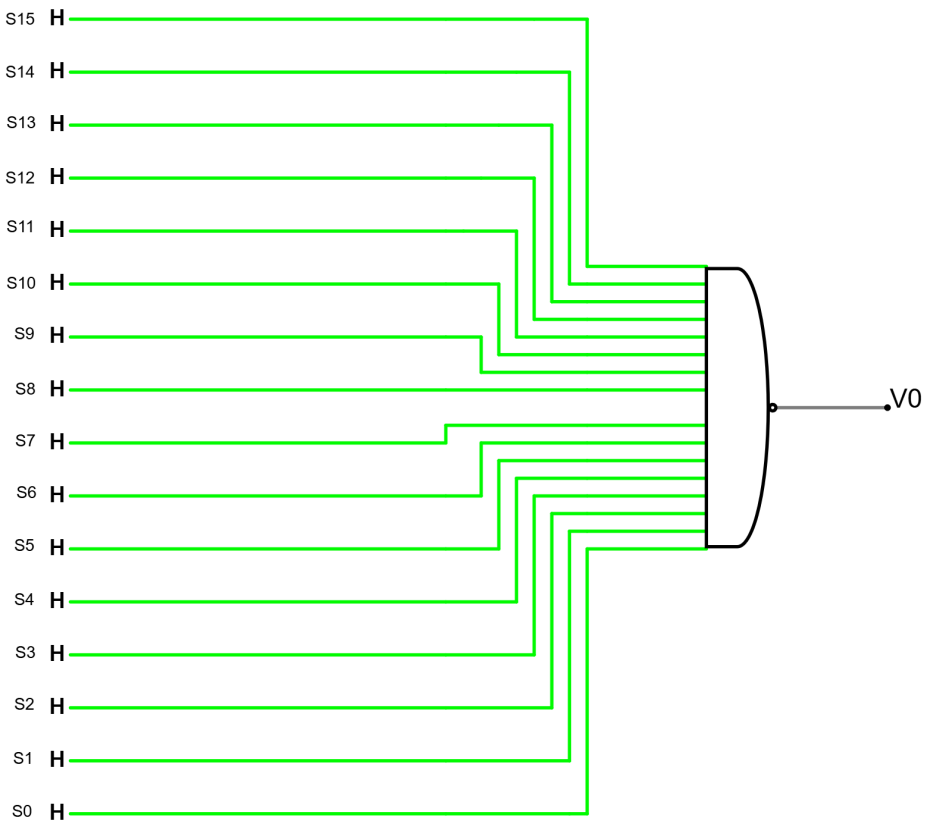

Link

## Salida $B_3$




Link

Salida $V_0$

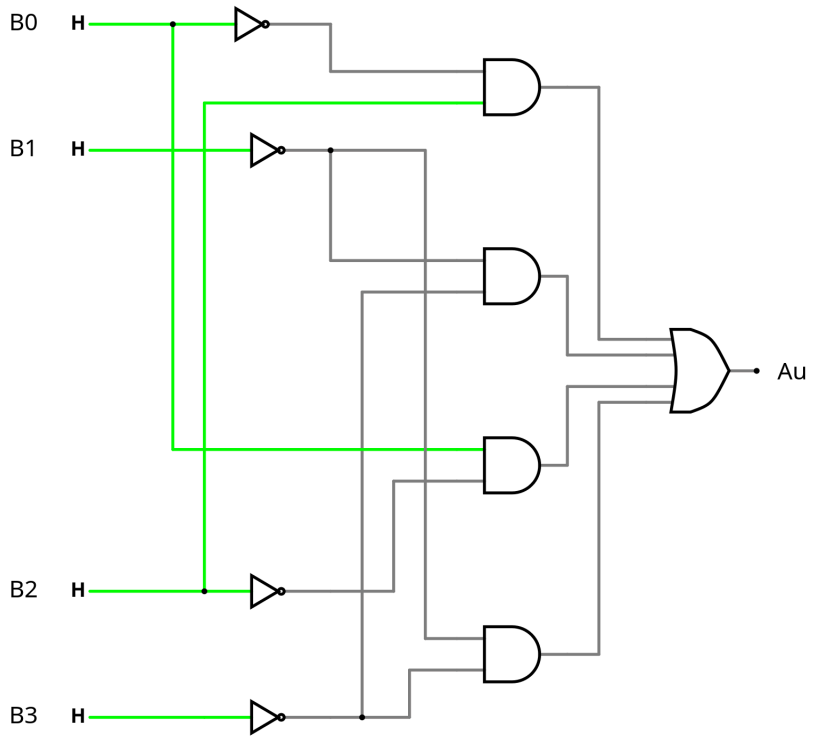



Link

# Decodificador

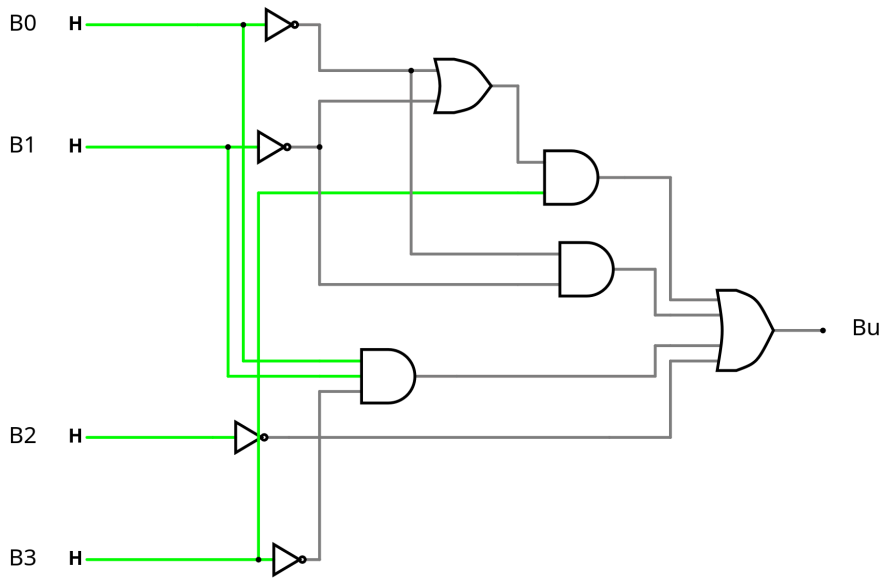
## Display de unidades

### Salida $A_u$



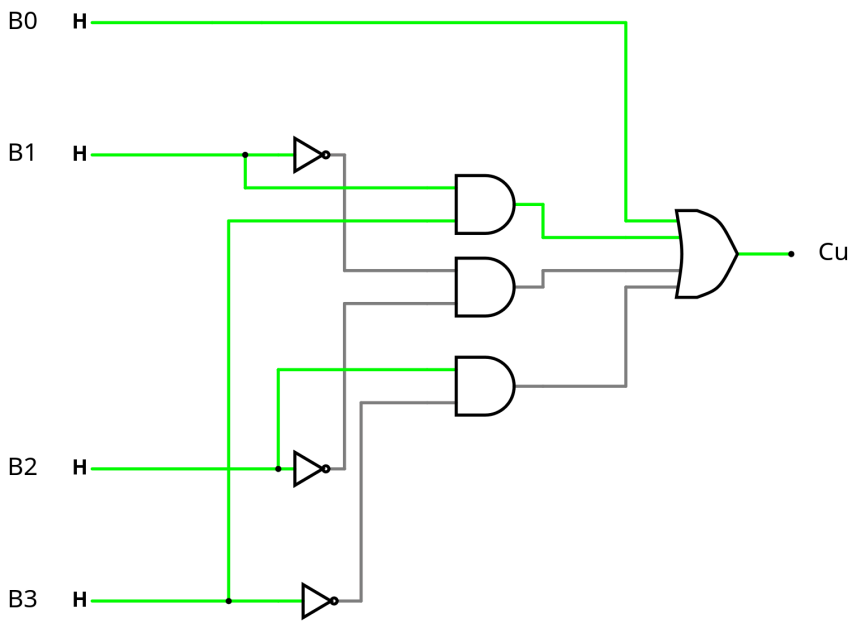

Link

### Salida $B_u$



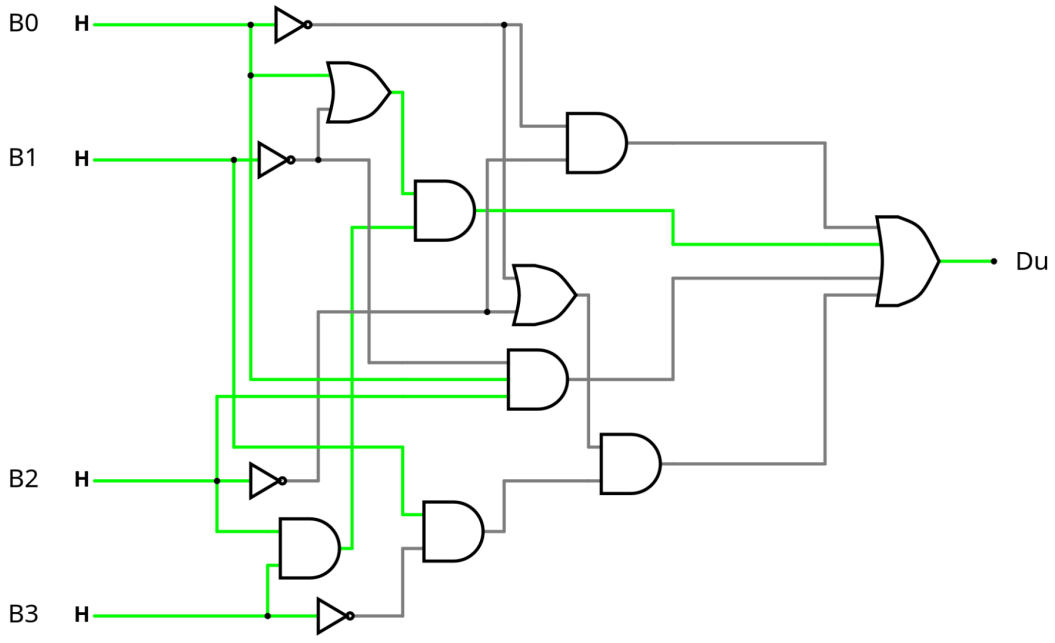
Link

Salida $C_u$



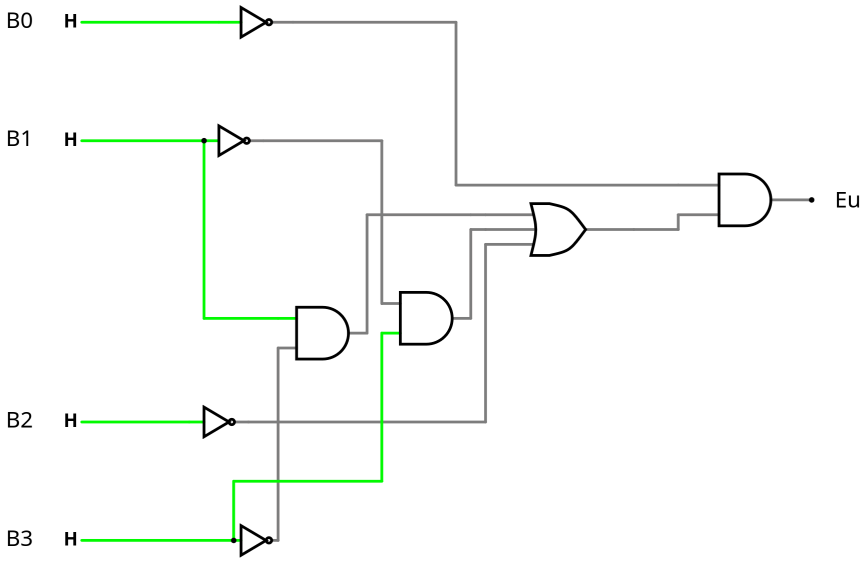

Link

Salida $D_u$



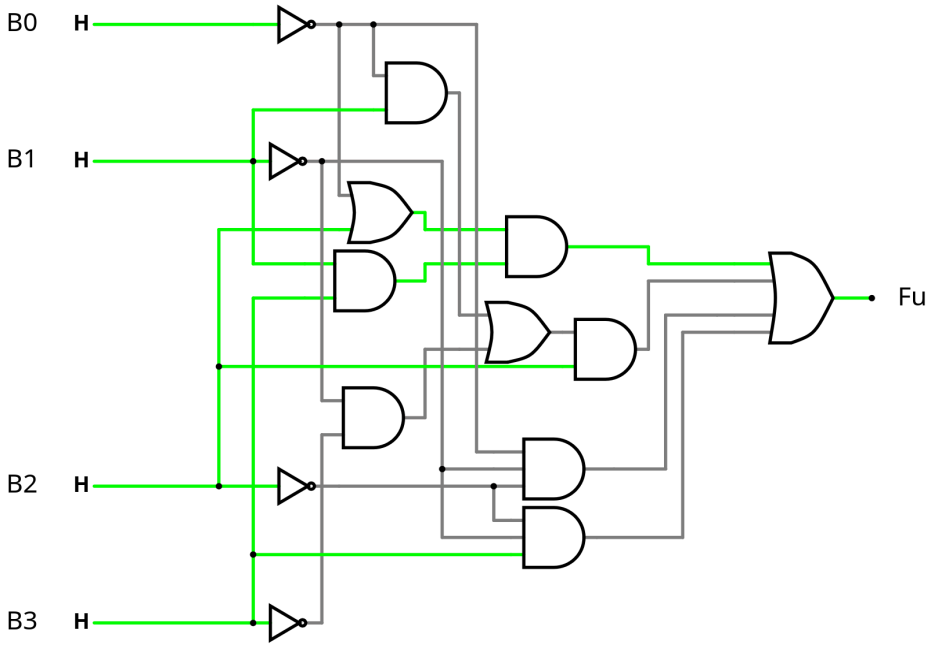

Link

Salida $E_u$



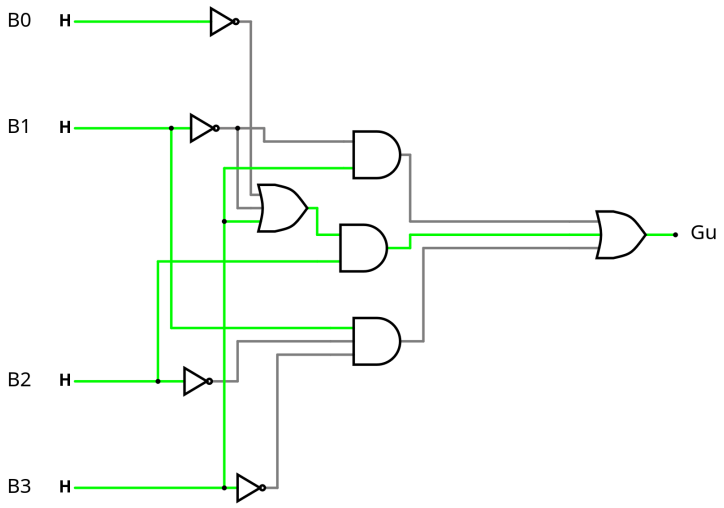

Link

Salida $F_u$




Link

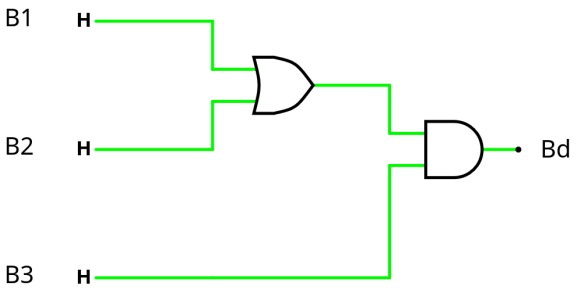
Salida $G_u$




Link

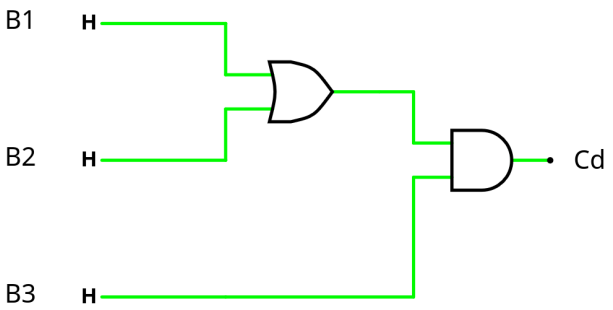
Display de decenas

Salida Bd




Link

Salida Cd

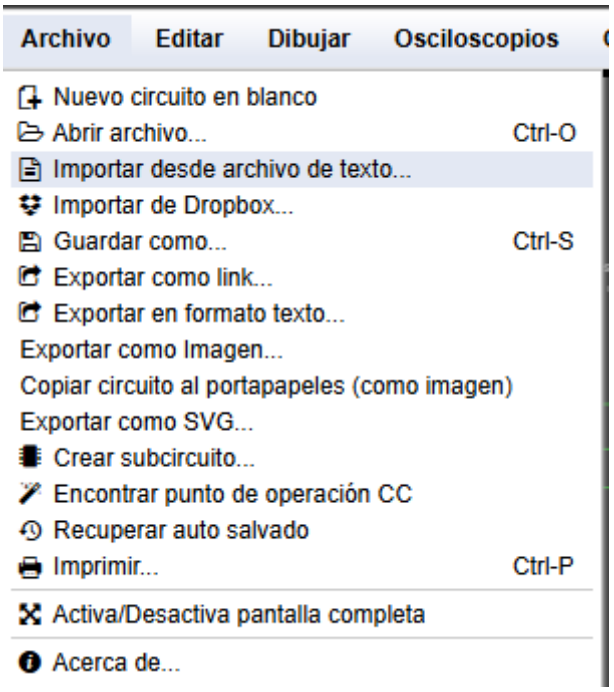


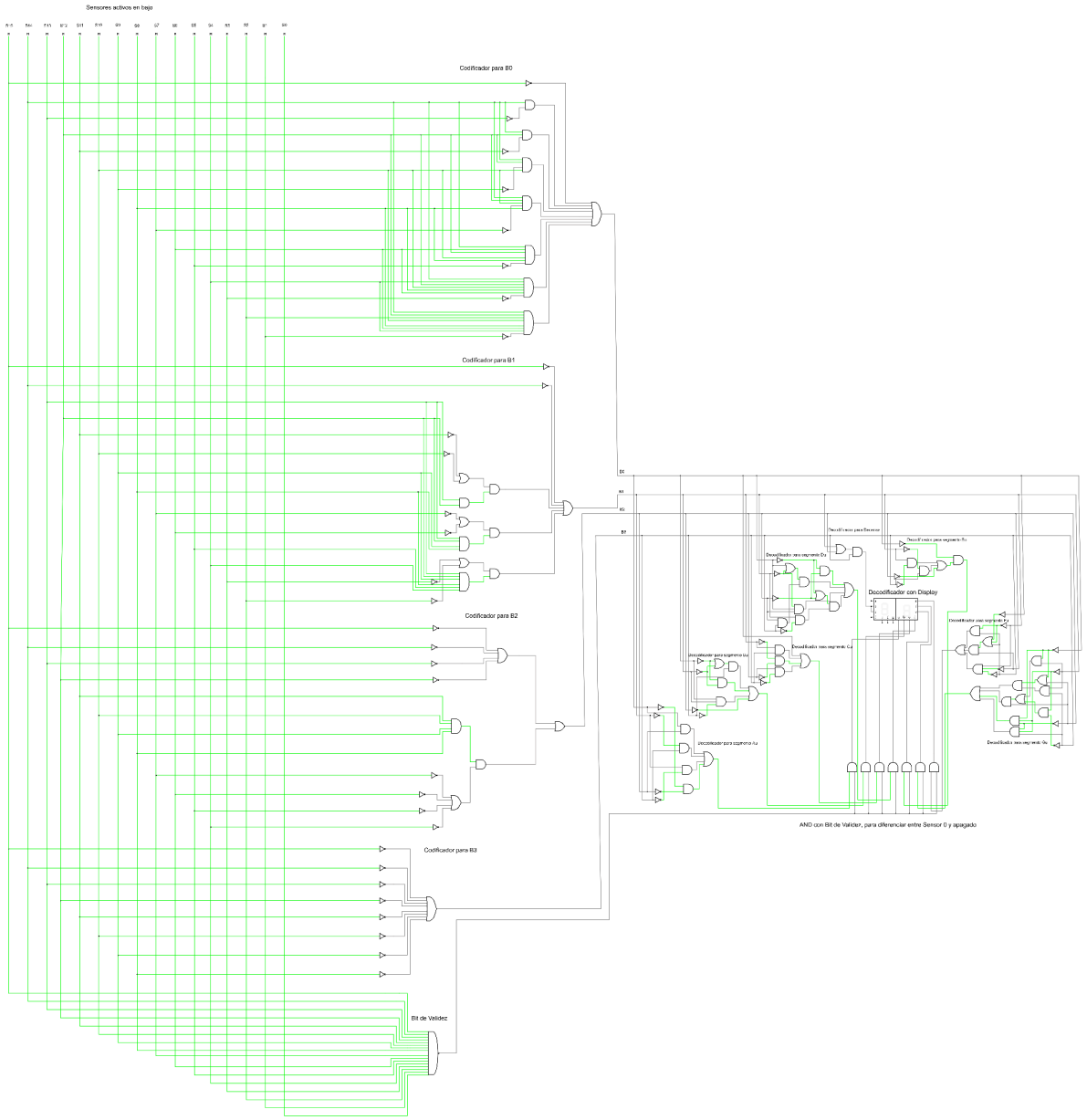

Link

## 5- Implementación del Circuito

A continuación dejaremos un link donde se podrá descargar un archivo de texto, que se deberá utilizar para importar en la página utilizada para los circuitos, donde se podrá ver cómo queda el conjunto de todos los circuitos que representan a cada bloque actuando en conjunto y de forma funcional e interactiva.

Link






# 6- Cálculo de Parámetros del Circuito

Para asegurar un funcionamiento adecuado, el LED correspondiente a un segmento del display ha de operar a una tensión de 2.2V y a una corriente mínima de 10mA y una máxima de 25 mA. Sabiendo que los displays se alimentan con una fuente de tensión continua de 5V, es seguro que tendremos que agregar una resistencia a cada LED del display para que no se dañen por el alto voltaje. Por ley de Ohm, sabemos que la corriente que fluya por el LED estará relacionada con la tensión de la fuente de la manera siguiente:

$$I = V / R$$

Sabiendo entonces que los LEDs operan a 2.2V y que la VCC vale 5V, buscaremos que los resistores proporcionen una caída de tensión tal que:

$$5V - UR = 2.2V$$
$$UR = 2.8V$$

Podemos entonces concluir que la tensión del resistor tendrá que ser siempre de 2.8V.

Se nos pide calcular el valor de la resistencia que ha de tener cada LED para que opere a corriente mínima y máxima, aplicando la ley de Ohm, hallamos esos valores para cada caso.

$$I = 10mA \Rightarrow R = 2.8V / 10mA \Leftrightarrow R = 280\Omega$$

$$I = 25mA \Rightarrow R = 2.8V / 25mA \Leftrightarrow R = 112\Omega$$