

INSTITUTO TECNOLÓGICO DE BUENOS AIRES

22.12 - ELECTRÓNICA III

---

## Trabajo Práctico N°1

---

*Grupo 4*

BERTACHINI, Germán	58750
DIEGUEZ, Manuel	56273
GALDEMAN, Agustín	59827
LAGUINGUE, Juan Martín	57430

Profesores:

DEWALD, Kevin

WUNDES, Pablo



PRESENTADO EL 5 DE SEPTIEMBRE DE 2019

# Índice

<b>1. Cálculo de Resolución y rango con convención de punto fijo</b>	<b>2</b>
<b>2. Implementación de módulos en verilog</b>	<b>2</b>
2.1. Demultiplexor de 4 salidas . . . . .	2
2.2. Codificador de 4 entradas . . . . .	3
<b>3. Conversor a código de Gray</b>	<b>4</b>

## 1. Cálculo de Resolución y rango con convención de punto fijo

El sistema binario es un sistema de numeración en el que los números se representan utilizando únicamente dos cifras. Es fundamental para la lógica computacional, debido a la simpleza y naturalidad con el que puede aplicarse al funcionamiento de una computadora o una máquina digital en general. El código binario permite representar números enteros, racionales e incluso signados. Existen diferentes maneras de representar el signo mediante el sistema binario, pero la más usada es por el complemento a dos del número. Existen además distintas convenciones para representar un número racional en binario. En este ejercicio se usará una de ellas: la convención de números en punto o coma fija. En ella se trabaja con un número fijo de bits y se acuerda dejar una cantidad determinada de bits para trabajar la parte fraccionaria del número.

En este ejercicio se escribió un programa al cual se le ingresa si el número es signado o no, cuantos bits de parte entera y cuantos bits de parte fraccionaria hay. Con esos datos se devuelven dos características del número: su resolución, o la cantidad más pequeña representable con la convención ingresada, y su rango, el número más grande representable menos el más pequeño.

El programa se escribió en Python 3.7 y no se requieren de paquetes externos para su funcionamiento.

## 2. Implementación de módulos en verilog

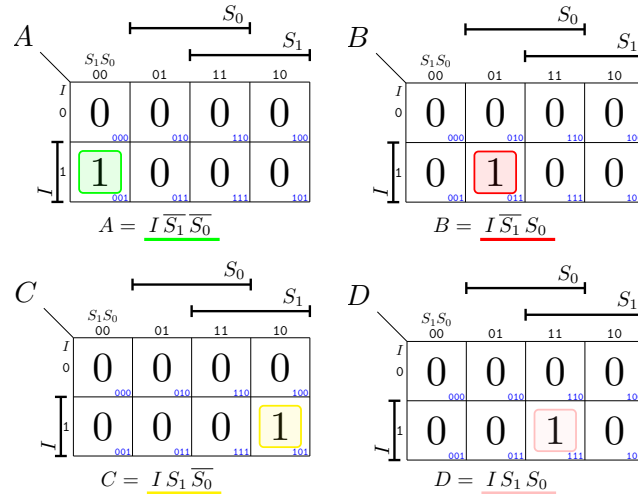
### 2.1. Demultiplexor de 4 salidas

A continuación, se analiza la tabla de verdad de un multiplexor de 4 salidas:

$I$	$S_1$	$S_0$	$A$	$B$	$C$	$D$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

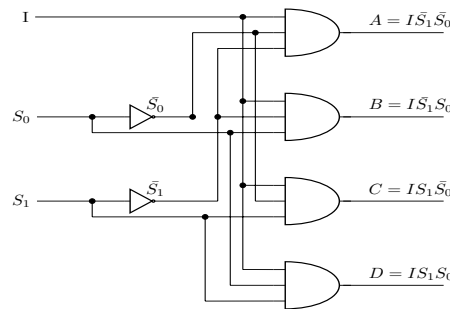
**Tabla 1:** Tabla de verdad del Demultiplexor

Cada salida distinta nos permitirá diagramar un mapa de Karnaugh propio, los mismos se presentan a continuación:



**Figura 1:** Mapas de Karnaugh de las salidas del Demultiplexor

Se procede a implementar el circuito hallado mediante los mapas:



**Figura 2:** Circuito Demultiplexor de 4 salidas

Respecto del diseño en verilog,

## 2.2. Codificador de 4 entradas

A continuación, se analiza la tabla de verdad de un codificador de 4 entradas:

A	B	C	D	S <sub>1</sub>	S <sub>0</sub>	E
1	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	0	1	0	0
0	0	0	1	1	1	0
X	X	X	X	X	X	1

**Tabla 2:** Tabla de verdad del Codificador

Cada salida distinta nos permitirá diagramar un mapa de Karnaugh propio. Se

contempla el caso de un error cuando las entradas no sean propias a las de un codificador. Los mapas se presentan a continuación:

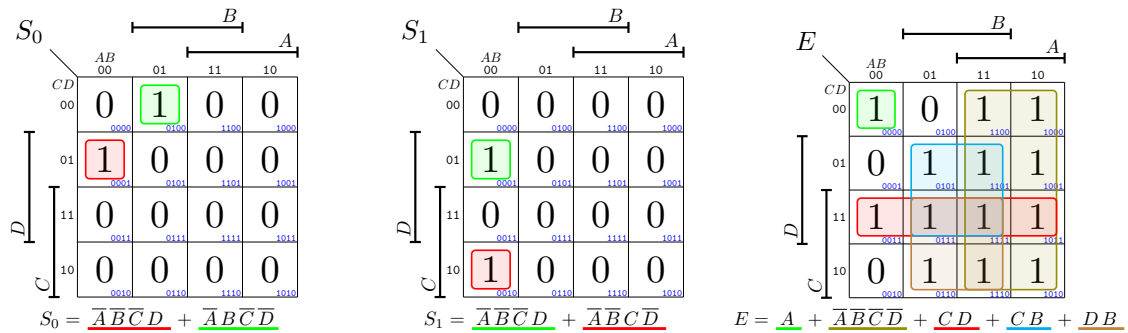


Figura 3: Mapas de Karnaugh de las salidas del Codificador

Por claridad, se coloca, por un lado, el circuito propio al codificador, y por otro, el utilizado para detectar un error. Sin embargo, los mismos podrían estar integrados. Los circuitos propuestos son los siguientes:

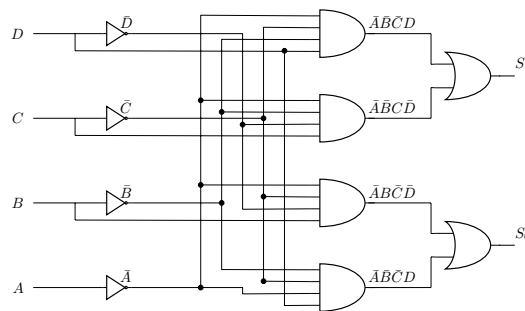


Figura 4: Circuito Codificador de 4 entradas

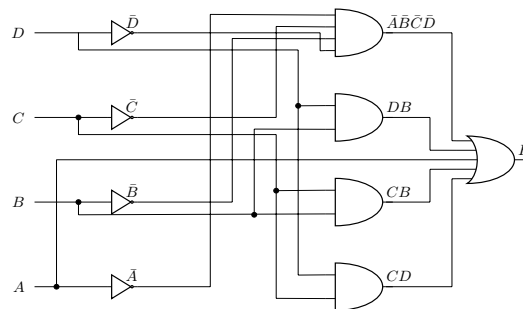


Figura 5: Circuito detector de error - Encoder

### 3. Conversor a código de Gray

Para este ejercicio, realizamos el desarrollo de un circuito lógico capaz de convertir un número binario de 4 bits a su equivalente de código de Gray, esto resulta en la siguiente tabla de verdad:

Entrada				Salida			
$X_1$	$X_2$	$X_3$	$X_4$	$Y_1$	$Y_2$	$Y_3$	$Y_4$
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

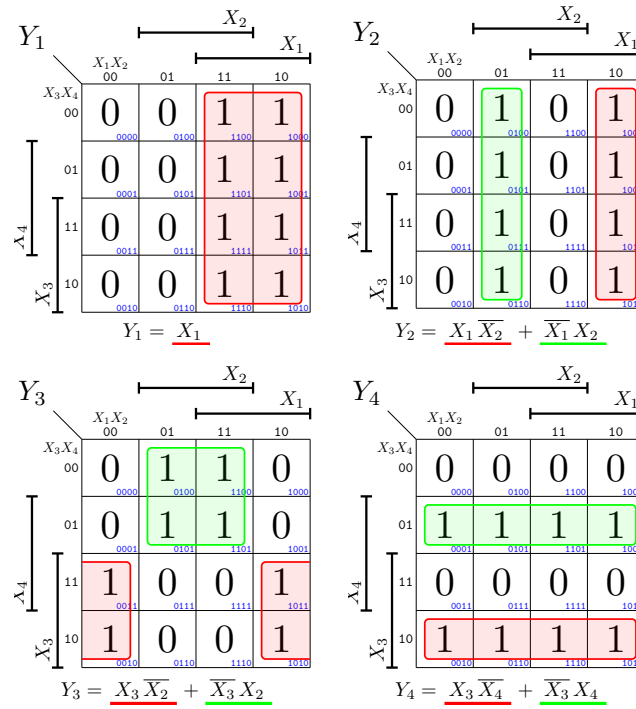
De la tabla de verdad obtenemos las siguientes ecuaciones en función de los minterminos:

$$\begin{aligned}
 Y_4 &= m_1 + m_2 + m_5 + m_6 + m_9 + m_{10} + m_{13} + m_{14} \\
 Y_3 &= m_2 + m_3 + m_4 + m_5 + m_{10} + m_{11} + m_{12} + m_{13} \\
 Y_2 &= m_4 + m_5 + m_6 + m_7 + m_8 + m_9 + m_{10} + m_{11} \\
 Y_1 &= m_8 + m_9 + m_{10} + m_{11} + m_{12} + m_{13} + m_{14} + m_{15}
 \end{aligned}$$

Que al reemplazar cada mintermino por su correspondiente expresión obtenemos:

$$\begin{aligned}
 Y_4 &= \overline{X_1} \cdot \overline{X_2} \cdot \overline{X_3} \cdot X_4 + \overline{X_1} \cdot \overline{X_2} \cdot X_3 \cdot \overline{X_4} + \overline{X_1} \cdot X_2 \cdot \overline{X_3} \cdot X_4 + \overline{X_1} \cdot X_2 \cdot X_3 \cdot \overline{X_4} + X_1 \cdot \\
 &\quad \overline{X_2} \cdot \overline{X_3} \cdot X_4 + X_1 \cdot \overline{X_2} \cdot X_3 \cdot \overline{X_4} + X_1 \cdot X_2 \cdot \overline{X_3} \cdot X_4 + X_1 \cdot X_2 \cdot X_3 \cdot \overline{X_4} \\
 Y_3 &= \overline{X_1} \cdot \overline{X_2} \cdot X_3 \cdot \overline{X_4} + \overline{X_1} \cdot \overline{X_2} \cdot X_3 \cdot X_4 + \overline{X_1} \cdot X_2 \cdot \overline{X_3} \cdot \overline{X_4} + \overline{X_1} \cdot X_2 \cdot \overline{X_3} \cdot X_4 + X_1 \cdot \\
 &\quad \overline{X_2} \cdot X_3 \cdot \overline{X_4} + X_1 \cdot \overline{X_2} \cdot X_3 \cdot X_4 + X_1 \cdot X_2 \cdot \overline{X_3} \cdot \overline{X_4} + X_1 \cdot \overline{X_2} \cdot X_3 \cdot X_4 \\
 Y_2 &= \overline{X_1} \cdot X_2 \cdot \overline{X_3} \cdot \overline{X_4} + \overline{X_1} \cdot X_2 \cdot \overline{X_3} \cdot X_4 + \overline{X_1} \cdot X_2 \cdot X_3 \cdot \overline{X_4} + \overline{X_1} \cdot X_2 \cdot X_3 \cdot X_4 + X_1 \cdot \\
 &\quad \overline{X_2} \cdot \overline{X_3} \cdot \overline{X_4} + X_1 \cdot \overline{X_2} \cdot \overline{X_3} \cdot X_4 + X_1 \cdot \overline{X_2} \cdot X_3 \cdot \overline{X_4} + X_1 \cdot \overline{X_2} \cdot X_3 \cdot X_4 \\
 Y_1 &= X_1 \cdot \overline{X_2} \cdot \overline{X_3} \cdot \overline{X_4} + X_1 \cdot \overline{X_2} \cdot \overline{X_3} \cdot X_4 + X_1 \cdot \overline{X_2} \cdot X_3 \cdot \overline{X_4} + X_1 \cdot \overline{X_2} \cdot X_3 \cdot X_4 + X_1 \cdot \\
 &\quad X_2 \cdot \overline{X_3} \cdot \overline{X_4} + X_1 \cdot X_2 \cdot \overline{X_3} \cdot X_4 + X_1 \cdot X_2 \cdot X_3 \cdot \overline{X_4} + X_1 \cdot X_2 \cdot X_3 \cdot X_4
 \end{aligned}$$

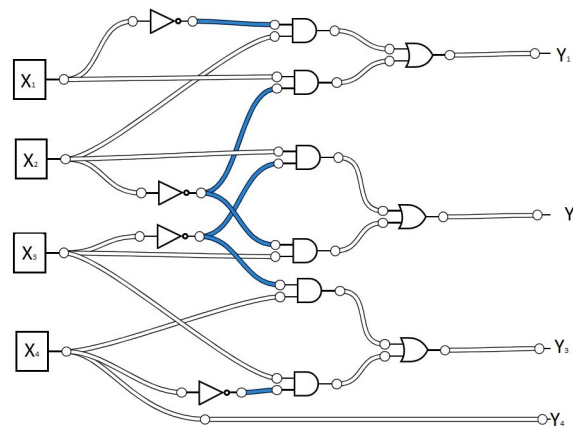
Tenemos unas funciones muy larga y como las tenemos expresadas en minterminos podemos simplificarlas por medio del mapa de Karnaugh. Ésto nos da a lugar a los siguientes mapas de Karnaugh y funciones de salida simplificadas:



**Figura 6:** Mapas de Karnaugh de las salidas  $Y_1$ ,  $Y_2$ ,  $Y_3$  e  $Y_4$

$$\begin{aligned}
 Y_4 &= X_3 \cdot \bar{X}_4 + \bar{X}_3 \cdot X_4 & Y_3 &= X_2 \cdot \bar{X}_3 + \bar{X}_2 \cdot X_3 \\
 \text{Formula de } Y_4 && \text{Formula de } Y_3 & \\
 Y_2 &= X_1 \cdot \bar{X}_2 + \bar{X}_1 \cdot X_2 & Y_1 &= X_1 \\
 \text{Formula de } Y_2 && \text{Formula de } Y_1 &
 \end{aligned}$$

De los valores obtenidos podemos realizar el siguiente circuito conformado por compuertas OR, AND y NOT:



**Figura 7:** Implementación del convertor a código de Gray