Universidad Nacional Autónoma de México

Facultad de Ciencias

Tarea I

Organización y Arquitectura de las Computadoras

319089660 - Bonilla Reyes Dafne

Profesor: Galaviz Casas José de Jesús

Ayudante: Lezama Hérnandez María Ximena
Ayudante: Montes Incin Sara Doris
Ayudante Laboratorio: Pérez Villanueva Ricardo Enrique

Febrero 2023

Tarea 01: Sistemas Numéricos

1. Preguntas

1. Expresar -31 y $+31$ en 8 bits en el sistema de complemento a 1.

   Usando la definición vista en clase para complemento a 1 tenemos que,

   El complemento a 1 *a* $b-1$ *de un número* r*, representado en* k *dígitos en base* b *se define como*

   $$C_{b-1}(r_b) = (b-1_k \dots b-1_1) - r_b$$

   desde donde $b-1$ *es el valor máximo de un dígito en base* b*.*

   Para esto, primero obtengamos la representación binaria de 31 en 8 bits

   $$31_{10} = \begin{array}{r} 15 \\ 2\overline{)31} \\ 1 \end{array} \Rightarrow \begin{array}{r} 7 \\ 2\overline{)15} \\ 1 \end{array} \Rightarrow \begin{array}{r} 3 \\ 2\overline{)7} \\ 1 \end{array} \Rightarrow \begin{array}{r} 1 \\ 2\overline{)3} \\ 1 \end{array} \Rightarrow \begin{array}{r} 0 \\ 2\overline{)1} \\ 1 \end{array}$$

   $\Rightarrow 31_{10} = 00011111_2$

   Y ahora obtengamos el complemento a 1 haciendo uso de la fórmula,

   $$C_1(31_{10}) = 11111111_2 - 00011111_2 = 11100000_2$$

   que de hecho, es la negación de 31 en base 2.

   Por otro lado, para -31 notemos que su complemento a 1 es exactamente 31 en binario en 8 bits, ya que es la negación de $C_1(31)$.

   Por lo tanto, $C_1(-31) = 00011111_2$.

2. Expresar $+13$ y -13 en 8 bits en el sistema de complemento a 2.

   Usando la definición vista en clase para complemento a 2 tenemos que,

   El complemento a 2 *a* $b-1$ *de un número* r*, representado en* k *dígitos en base* b *se define como*

   $$C_b(r_b) = (10_k \dots 0_1) - r_b$$

   desde donde $b-1$ *es el valor máximo de un dígito en base* b*.*

   Para esto, primero obtengamos la representación binaria de 13 en 8 bits

   $$13_{10} = \begin{array}{r} 6 \\ 2\overline{)13} \\ 1 \end{array} \Rightarrow \begin{array}{r} 3 \\ 2\overline{)6} \\ 0 \end{array} \Rightarrow \begin{array}{r} 1 \\ 2\overline{)3} \\ 1 \end{array} \Rightarrow \begin{array}{r} 0 \\ 2\overline{)1} \\ 1 \end{array}$$

   $\Rightarrow 13_{10} = 00001101_2$

   Y ahora obtengamos el complemento a 2 haciendo uso de la fórmula,

   $$C_2(13_{10}) = 10000000_2 - 00001101_2 = 11110011_2$$

   lo cual también se puede obtener usando que de derecha a izquierda, en cuanto se encuentre el primer 1, el restante se escribe como el complemento.

   Por otro lado, para -13 notemos que su complemento a 2 es exactamente 13 en binario en 8 bits, ya que es la negación de $C_2(13)$.

   Por lo tanto, $C_2(-13) = 00001101_2$.

3. ¿Cuál es el rango de números representables en complemento a 2 con 4 bits?

   Recordemos que para n bits, es posible representar de -2^{n-1} a $2^{n-1}-1$ enteros en complemento a 2. (Wikipedia, 2023)

   En el caso de 4 bits, tenemos $n = 4$, entonces el rango de números representables va de

   $$-2^{4-1} \text{ a } 2^{4-1} - 1 = -2^3 \text{ a } 2^3 - 1 = -8 \text{ a } 7$$

   Por lo tanto, los números representables en complemento a 2 con 4 bits va de -8 a 7.

4. El número $(10110101)_2$ es un número de 8 bits incluyendo el bit signo en complemento a 2. Da su equivalente en base decimal.

   Siguiendo el procedimiento visto en clase, para convertir 10110101_2 a base decimal, desarrollaremos las potencias en base 2 de acuerdo a su posición, tomando en cuenta que si tenemos un 1 entonces se suma esa potencia, en caso contrario, se ignora.

   También, es importante mencionar que el bit más significativo de 10110101_2 es 1, lo que nos indica que tenemos un número negativo. Esto solo cambiará el desarrollo de las potencias para el bit más significativo, en este caso 128, que será tomado como número negativo, entonces

   $$\begin{aligned} &-1(2^7) + 0(2^6) + 1(2^5) + 1(2^4) + 0(2^3) + 1(2^2) + 0(2^1) + 1(2^0) \\ &= -1(128) + 0 + 32 + 16 + 0 + 4 + 0 + 1 \\ &= -128 + 0 + 32 + 16 + 0 + 4 + 0 + 1 = -75 \end{aligned}$$

   Por lo tanto, $(10110101)_2 = -75_{10}$.

5. El número $(00110111)_2$ es un número de 8 bits incluyendo el bit signo en complemento a 2. Da su equivalente en base decimal.

   Análogamente al ejercicio anterior, para convertir 00110111_2 a base decimal, desarrollaremos las potencias en base 2 de acuerdo a su posición, tomando en cuenta que si tenemos un 1 entonces se suma esa potencia, en caso contrario, se ignora.

   Así mismo, también notemos que el bit más significativo de 00110111_2 es 0, lo que nos indica que tenemos un número positivo. Esto no cambia el desarrollo de las potencias, entonces

   $$\begin{aligned} &0(2^7) + 0(2^6) + 1(2^5) + 1(2^4) + 0(2^3) + 1(2^2) + 1(2^1) + 1(2^0) \\ &= 0 + 0 + 32 + 16 + 0 + 4 + 2 + 1 = 55 \end{aligned}$$

   Por lo tanto, $(00110111)_2 = 55_{10}$.

6. Menciona las 4 unidades funcionales principales de una computadora y describe su funcionamiento.

   Las 4 unidades funcionales principales de una computadora son:

   - Unidad de control:

     Las funciones de las unidades de entrada/salida, aritmético-lógica y de memoria deben estar coordinadas para que se realicen de manera secuencial, es decir, el procesador acepta la entrada, la coloca en la memoria, procesa la entrada almacenada y genera la salida. La unidad de control es la encargada de coordinar esta secuencia, así como también es responsable de llevar a cabo todas las instrucciones almacenadas de un programa.

     La unidad de control envía señales a otras unidades y detecta sus estados. Las señales de temporización reales que gobiernan la transferencia de datos entre la unidad de entrada, el procesador, la memoria y la unidad de salida son generadas por la unidad de control. De esta forma, las unidades funcionales de las computadoras cooperan para generar resultados útiles.

- Unidad aritmético-lógica:

  La unidad aritmético-lógica (ALU por sus siglas en inglés) realiza casi todas las operaciones aritméticas y lógicas de una computadora. Los cálculos aritméticos incluyen suma, resta, multiplicación y división, mientras que las operaciones lógicas que realiza son las de AND, OR y NOT. Las decisiones lógicas implican la comparación de dos elementos de datos para ver cuál es mayor, menor o igual.

  Los operandos se traen a la ALU desde la memoria y se almacenan en elementos de almacenamiento de alta velocidad llamados registros. Después, de acuerdo con las instrucciones, la operación se realiza en la secuencia requerida.

- Unidad de memoria:

  Se define a la unidad de memoria como el área de almacenamiento en donde se guardan los datos asociados con los programas, que a su vez tienen el conjunto de instrucciones que comunican a la unidad aritmético-lógica las operaciones que deben realizar.

  La memoria se puede clasificar en dos tipos: memoria primaria y memoria secundaria.

  1. *Memoria primaria:* La memoria primaria también conocida como memoria principal es una sección de la memoria de la computadora a la que el procesador accede directamente mediante el bus de datos. Es la memoria más rápida que funciona a velocidades electrónicas y es volátil, lo que significa que cuando la computadora se apaga, se pierde todo lo que contiene. Los programas deben almacenarse en esta memoria mientras se ejecutan. La memoria contiene una gran cantidad de celdas de almacenamiento de semiconductores. Cada uno capaz de almacenar un bit de información. Estos se procesan en un grupo de sitios fijos llamados palabra. Para proporcionar un fácil acceso a una palabra en la memoria, se asocia una dirección distinta con cada ubicación de palabra. Los programas deben residir en la memoria durante la ejecución. La RAM, ROM y memoria caché forman parte de la memoria primaria.

  2. *Memoria secundaria:* La memoria secundaria se usa cuando una gran cantidad de datos y programas deben almacenarse a largo plazo, en particular información a la que se accede con poca frecuencia. No es volátil, lo que significa que los datos se almacenan permanentemente independientemente del apagado. Los ejemplos más comunes de memoria secundaria son los discos magnéticos, las cintas magnéticas y los discos ópticos. Es más lenta y menos costosa en comparación con la memoria primaria.

- Unidad de entrada y salida:

  - Entrada: Las unidades de entrada son utilizadas por la computadora para leer los datos o instrucciones que guían a la unidad aritmético-lógica sobre qué operaciones deben realizarse. Así mismo, también controla el movimiento de datos entre la computadora y sus dispositivos de entrada y salida (teclados, mouse, joysticks, panel táctil, etc.). Los dispositivos de entrada convierten los datos de los humanos en una forma comprensible para las computadoras, es decir, se traduce a su código binario equivalente a través de un cable y se alimenta a la memoria o al procesador.
  - Salida: La unidad de salida es la contraparte de la unidad de entrada. La unidad de salida se refiere a los dispositivos de salida que se utilizan para comunicar datos de las computadoras a los usuarios. Su función básica es enviar información retenida o generada dentro de una computadora al mundo exterior. Los dispositivos de salida muestran información de una manera que el usuario puede entender. Algunos ejemplos de dispositivos de salida son impresoras, bocina, monitores, etc.

7. Realiza la siguiente operación $-3-6$ de base decimal en base binario representando los números en 4 bits.

   Primero, veamos que el resultado de la suma en base 10 es -9.

   Ahora, representemos a los sumandos en 4 bits usando complemento a 2 dado que son números negativos

- Para -3

  $3_{10} = 0011_2 \Rightarrow -3_{10} = C_2(0011) = 1101$

- Para -6

  $6_{10} = 0110_2 \Rightarrow -6_{10} = C_2(0110) = 1010$

Y realicemos la suma en binario de ambos números

$$\begin{array}{cccccc} & & 1 & 1 & 0 & 1 \\ + & & 1 & 0 & 1 & 0 \\ \hline & \mathbf{1} & 0 & 1 & 1 & 1 \end{array}$$

Notemos que obtuvimos 1 bit de acarreo y que el bit más significativo es 0, lo cual indica que nuestra suma nos dio un número positivo, lo cual es incorrecto, ya que el resultado de la suma de dos números negativos es otro número negativo.

Esto sucede debido a que -9 sale del rango de enteros representables en 4 bits. Por el ejercicio 3, recordemos que el rango para 4 bits en complemento a 2 va de -8 a 7.

Por lo tanto, -9 no se puede representar en una arquitectura de 4 bits.

8. Realiza la siguiente operación $9 + 3$ de base decimal en base binario representando los números en 4 bits.

   Primero, veamos que el resultado de la suma en base 10 es 12.

   Ahora, representemos a los sumandos en 4 bits

   - Para 9

     $9_{10} = 1001_2$

   - Para 3

     $3_{10} = 0011_2$

   Y realicemos la suma en binario de ambos números

$$\begin{array}{cccccc} & & 1 & 0 & 0 & 1 \\ + & & 0 & 0 & 1 & 1 \\ \hline & & 1 & 1 & 0 & 0 \end{array}$$

   Notemos que el bit más significativo es 1, lo cual indica que nuestra suma nos dio un número negativo, lo cual es incorrecto, ya que el resultado de la suma de dos números positivos es otro número positivo.

   Esto sucede debido a que 12 sale del rango de enteros representables en 4 bits. Por el ejercicio 3, recordemos que el rango para 4 bits en complemento a 2 va de -8 a 7. No obstante, veamos que de hecho uno de los sumandos, es decir, 9, sale del rango de 4 bits.

   Por lo tanto, 12 no se puede representar en una arquitectura de 4 bits.

9. Realiza la siguiente suma de 2 bits.

A		B		Acarreo	Suma
0	+	0	=	0	0
0	+	1	=	0	1
1	+	0	=	0	1
1	+	1	=	1	0

Veamos que esta es la tabla de verdad de un semi-sumador de 2 bits, más comúnmente conocido como half adder. Este es capaz de hacer la suma algebraica entre dos números binarios de un bit cada uno. Half adder es un circuito que recibe como entrada dos dígitos binarios y da como salida a su suma, más el acarreo.

Notemos que en base 2, las sumas de $0+0$, $0+1$ y $1+0$ son fáciles de calcular, de hecho, funcionan como en el sistema de decimal, por lo que obtendremos 0, 1 y 1 respectivamente. Sin embargo, veamos que pasa con la suma de $1+1$.

En este caso, obtendríamos 2 si trabajamos en base 10, pero recordemos que en base 2, solo tenemos a $\{0,1\}$, por lo que este número se sale de los valores con los que podemos trabajar.

Sin embargo, para solucionar esto, se decidió usar la convención del bit de acarreo. Análogo a lo que pasa en base 10 cuando en una suma obtenemos un número mayor o igual a 10, si sumamos $1+1$ obtenemos 0 más un bit de acarreo.

En el procesador de la computadora, la bandera de acarreo (generalmente indicada como la bandera C de carry) es un solo bit en un registro de estado del sistema/registro de bandera que se usa para indicar cuándo se ha generado un acarreo o préstamo aritmético a partir del bit de la unidad aritmético-lógica más significativa.

10. Suma los siguientes dos números $(10011011)_2 + (11101100)_2$. Explica qué sucede con el acarreo.

    Tenemos que,

$$\begin{array}{cccccccccc} & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ + & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}$$

    Lo que nos da la siguiente tabla de verdad, en donde A = 10011011 y B = 11101100

A		B		Acarreo	Suma
1	+	0	=	0	1
1	+	0	=	0	1
0	+	1	=	0	1
1	+	1	=	1	0
1	+	0	=	1	0
0	+	1	=	1	0
0	+	1	=	1	0
1	+	1	=	1	1

    Veamos que a partir del bit en la posición 4, encontramos una suma de la forma $1+1$. Por el ejercicio 9, recordemos que esto implica que se tendrá un bit de acarreo, que de hecho, se continuará sumando hasta los bits más significativos, obteniendo así un resultado de 9 bits.

    Por lo que podemos concluir que este número no es representable en una arquitectura de 8 bits.

11. Representa el número 39.1 en base 2 usando el estándar IEEE 754. Siguiendo los pasos vistos en clase, tenemos lo siguiente:

    1. *Paso 1.* Obtener la representación binaria de 39.1
       - Para 39

$$39_{10} = \begin{array}{ccccccccccc} & & 19 & & 9 & & 4 & & 2 & & 1 & & 0 \\ & 2\overline{)39} & \Rightarrow & 2\overline{)19} & \Rightarrow & 2\overline{)9} & \Rightarrow & 2\overline{)4} & \Rightarrow & 2\overline{)2} & \Rightarrow & 2\overline{)1} & \\ & 1 & & 1 & & 1 & & 0 & & 0 & & 1 & \end{array}$$

       $\Rightarrow 39_{10} = 100111_2$

– Para 0.1

$$0.1 \times 2 = 0.2$$
$$0.2 \times 2 = 0.4$$
$$0.4 \times 2 = 0.8$$
$$0.8 \times 2 = 1.6$$
$$0.6 \times 2 = 1.2$$

$\Rightarrow 0.1_{10} = 00011_2$

Por lo tanto, $39.1 = 100111.0011_2$

2. *Paso 2.* Normalizar la mantisa

   Movemos la coma hasta el 1 más significativo y análogo a la notación científica, la potencia de 2 será el número de veces que movimos la coma.

$$39.1 = 100111.0011_2 \Rightarrow 1,0011100011 \times 2^5$$

3. *Paso 3.* Sesgar el exponente

   Usando que el exponente sesgado se define como $n + 127$ en donde n es la potencia de 2 en la mantisa, entonces

$$5 + 127 = 132$$

   Y obtenemos la representación binaria de 132.

$$132_{10} = \begin{array}{r|l} 66 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 33 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 16 & 2 \\ 1 & \end{array} \Rightarrow \begin{array}{r|l} 8 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 4 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 2 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 1 & 2 \\ 1 & \end{array} \Rightarrow \begin{array}{r|l} 0 & 2 \\ & \end{array}$$

   $\Rightarrow 132_{10} = 10000100_2$

4. *Paso 4.* Juntar todo

   Unimos los números obtenidos de la siguiente forma:

signo	exponente sesgado	mantisa normalizada

   Y ya que 39.1 es un número positivo, entonces el signo del número será 0, obteniendo así

0	10000100	0011100011

Por lo tanto, 01000010000111000011_2 es la representación en base 2 de 39.1 usando el estándar IEEE 754.

12. Representa el número 502 millones en base 2 usando el estándar IEEE 754.

Siguiendo los pasos vistos en clase, tenemos lo siguiente:

1. *Paso 1.* Obtener la representación binaria de 502 millones

   Primero, expresamos a 502 millones en notación científica y como es un número muy grande, haremos una excepción en mostrar las divisiones para obtener su representación en binario, entonces

$$\Rightarrow 502_{10} \times 10^6 = 11101111010111110100110000000_2$$

2. *Paso 2.* Normalizar la mantisa

   Movemos la coma hasta el 1 más significativo y análogo a la notación científica, la potencia de 2 será el número de veces que movimos la coma.

$$502 \times 10^6 = 11101111010111110100110000000_2 \Rightarrow 1.11011110101111101001100000000 \times 2^{28}$$

3. *Paso 3.* Sesgar el exponente

   Usando que el exponente sesgado se define como $n + 127$ en donde n es la potencia de 2 en la mantisa, entonces

   $$28 + 127 = 155$$

   Y obtenemos la representación binaria de 155.

   $$155_{10} = \begin{array}{r|l} 77 & 2 \\ 1 & \end{array} \Rightarrow \begin{array}{r|l} 38 & 2 \\ 1 & \end{array} \Rightarrow \begin{array}{r|l} 19 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 9 & 2 \\ 1 & \end{array} \Rightarrow \begin{array}{r|l} 4 & 2 \\ 1 & \end{array} \Rightarrow \begin{array}{r|l} 2 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 1 & 2 \\ 0 & \end{array} \Rightarrow \begin{array}{r|l} 0 & 2 \\ 1 & \end{array}$$

   $\Rightarrow 132_{10} = 10011011_2$

4. *Paso 4.* Juntar todo

   Unimos los números obtenidos de la siguiente forma:

signo	exponente sesgado	mantisa normalizada

   Recordemos que la mantisa ocupa de 23 bits en el estándar IEEE 754. Sin embargo, nuestra mantisa es de 28 bits. Notemos que podemos omitir los últimos 5 ceros, ya que es el periodo infinito de ceros después del punto decimal, quedándonos así solo con los 23 bits que necesitamos. Y ya que 502 millones es un número positivo, entonces el signo del número será 0, obteniendo así

0	10011011	11101111010111101001100

Por lo tanto, 01001101111101111010111101001100 es la representación en base 2 de 502 millones usando el estándar IEEE 754.

13. ¿Qué ventajas y desventajas puedes encontrar en el modelo de la arquitectura de Von Neuman? Argumenta tu respuesta.

    - Ventajas:

      Una de las mayores ventajas de la arquitectura de Von Neumann es que la unidad de control recupera instrucciones y datos de la misma manera desde una unidad de memoria, simplificando el diseño del chip del microcontrolador. Esto genera una enorme ventaja, ya que el contenido de la RAM se puede utilizar tanto para el almacenamiento de variables como para el almacenamiento de instrucciones de programas.

      Por otro lado, su memoria se utiliza por completo, ya que los programadores tienen el control de ella, haciendo que esta sea fácil de desarrollar y diseñar, aumentando su eficiencia y velocidad.

    - Desventajas:

      No se permiten ejecuciones paralelas de programas debido al procesamiento de instrucciones en serie, por lo que solo se puede acceder a un bus a la vez. Esto da como resultado que la CPU se vea forzada a esperar continuamente a que lleguen los datos necesarios desde o hacia la memoria, ya que es más rápida que un bus de datos. A esta desventaja la conocemos como el cuello de botella de Von Neuman. Esta desventaja provoca un rendimiento limitado entre la CPU y la memoria en comparación con la cantidad de memoria. También es propensa a la sobrescritura de un espacio en memoria, debido a que se guardan datos y programas en el mismo lugar.

14. La arquitectura de Von Neuman fue descrita por el matemático y físico John von Neumann y otros, en el *primer borrador de un informe sobre el EDVAC*. Pero la computación de 1945 a la actualidad ha dado pasos agigantados, aumentando la complejidad de la arquitectura inicial, la base de su funcionamiento es la misma. ¿Qué cambios aprecias hoy en día en tu computador que no se ven descritos por el diagrama dado en 1945? Argumenta tu respuesta.

    La mayoría de computadoras en la actualidad conservan en su mayoría la arquitectura de Von Neuman, a excepción de algunas que usan la arquitectura de Hardvard. Podemos ver principalmente cambios

en la organización que implementa la arquitectura, es decir, en los circuitos lógicos. En esta rama en particular se ha avanzado demasiado, esto se puede ver en la motherboard de cualquier computadora actual contra una de hace algunas décadas. Las computadoras actuales tienen la capacidad de procesar datos de manera más eficiente usando menor tamaño en Hardware que hace 60 años.

Por otro lado, veamos que a diferencia de la EDVAC, las computadoras actuales incluyen tarjetas gráficas y de video, más de un núcleo en el procesador y mayor capacidad de almacenamiento y procesamiento.

Las computadoras actuales tienen la capacidad de realizar cálculos más complejos y en menor tiempo gracias a cambios en hardware y software. Por ejemplo, actualmente con las unidades de almacenamiento sólidas, accedemos a espacios en memoria en menor tiempo que en discos duros. La arquitectura de Von Neuman favorecía el uso de una única memoria como medio de almacenamiento de datos y programas, y actualmente le damos gran importancia al uso de la caché y la memoria RAM para evitar problemas como el cuello de botella.

2. Punto extra

1. En la siguiente imagen, se nos muestra la disyunción y la conjunción proposicional usando interruptores. Usando ese mismo modelo ¿cómo sería un xor usando interruptores?

   Recordemos que la puerta XOR es una puerta lógica digital que proporciona una salida verdadera cuando el número de entradas verdaderas es impar. Una puerta XOR implementa un o exclusivo de lógica. Si ambas entradas son falsas o ambas son verdaderas, se obtiene como resultado una salida falsa.

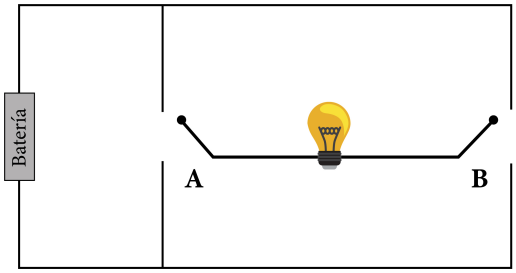

Figura 1: Representación de XOR usando interruptores.

3. Referencias

1. Basic Structure Of Computers. (s. f.). https://www.jbiet.edu.in/coursefiles/cse/HO/cse2/DLD1.pdf
2. Functional Units of Digital System - javatpoint. (s. f.). https://www.javatpoint.com/functional-units-of-digital-system
3. Editorial Team. (2022, 27 noviembre). Functional Units of Computer. Being Intelligent. https://beingintelligent.com/functional-units-of-computer.html
4. GeeksforGeeks. (2021, 14 noviembre). Functional Components of a Computer. https://www.geeksforgeeks.org/functional-components-of-a-computer/
5. Wikipedia contributors. (2023, 12 febrero). Two's complement. Wikipedia. https://en.wikipedia.org/wiki/Two'scomplement
6. What is the ≪biggest≫ negative number on a 4-bit machine? (2010, 15 febrero). Stack Overflow. https://stackoverflow.com/questions/2263654/what-is-the-biggest-negative-number-on-a-4-bit-machine

7. YOSTHIN ARIZA. (2019, 7 febrero). Suma de binarios - 4 Bits [Vídeo]. YouTube. https://www.youtube.com/watch?v=2NQgZ8EqH0I
8. Wikipedia contributors. (2022, 15 septiembre). Carry (arithmetic). Wikipedia. https://en.wikipedia.org/wiki/Carry(arithmetic)
9. Wikipedia contributors. (2021, 30 diciembre). Carry flag. Wikipedia. https://en.wikipedia.org/wiki/Carryflag