Tarea 1 - Organización y Arquitectura de Computadoras

Reyna Méndez Cristian Ignacio - 320149579
López Molina Andrés Daniel - 319117026

5 de septiembre de 2024

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

Primeramente pasamos el número 31 en binario.

2	31	1
2	15	1
2	7	1
2	3	1
2	1	1

De donde copiamos los residuos de abajo hacia arriba, de este modo nos queda: $(11111)_2$, pero como la instrucción nos pide que lo representemos en 8 bits, entonces agregamos 3 0's a la izquierda, así... $(00011111)_2 = 31$ en 8 bits.

Ahora, ya que tenemos la representación de 31 en binario y nos piden representar -31 en complemento a 1, solamente es necesario voltear todos los valores, es decir, los 0's pasan a ser 1 y los 1's a 0. Así: $C_1(11100000)_2$ = -31 en 8 bits.

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

Seguimos los mismos pasos que en la pregunta anterior, primero pasamos a binario el 13:

2	13	1
2	6	0
2	3	1
2	1	1

Y ahora copiamos de abajo hacia arriba $(1101)_2$, pero como lo pide en 8 bits, agregamos 4 0's a la izquierda, es decir, $(00001101)_2 = 13$. Ahora, pasamos el número a complemento a 1, proceso explicado en la pregunta anterior: $C_1(11110010)_2$ = -13.

Luego, para pasar de complemento a 1, a complemento a 2 solo es necesario sumarle 1.

$$\begin{array}{r} 11110010 \\ +00000001 \\ \hline 11110011 \end{array}$$

Esto es igual a $C_2(11110011)_2$ = -13. Pero hay un truco más, el cuál conciste en fijar todos los números antes del primer uno, de derecha a izquierda y los demás dígitos cambian su valor, de 1's a 0 y 0's a 1. Veamos un ejemplo:

Tenemos $(00001101)_2$, el cuál es el 13 en binario, de este modo: $(00001101)_2$, el primer uno de derecha a izquierda, coloreado de un color rojo, se mantendra fijo (y todos los números anteriores a ese) y los números en azul, cambiarán de valor. Así: $(11110011)_2$ y veremos que ese valor es el mismo que $C_2(11110011)_2$ = -13, el cuál fue calculado anteriormente.

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

Para contestar esta pregunta, es necesario recapitular lo visto en el texto "Sistemas Numéricos y Representación de Números en una Computadora", hecho por el profesor José Galaviz Casas, en la página 13 donde nos habla de las desventajas del complemento a 2 con la idea que surge de sumar un 1 y quitar el 0 negativo, y está es que el número de números negativos representables está dado por 2^{k-1} de donde k es el número de bits pero en el caso de los números positivos está dado por $2^{k-1} - 1$, recordemos que el número de bits en este caso es de 4, por lo tanto, el rango de números representables para 4 bits es de: -16, -15, ..., 14, 15.

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.

Para responder está pregunta, ocuparemos el trucazo que explicamos con anterioridad. Tenemos el número $(10110101)_2$, el cuál está en complemento a 2, así que siguiendo la misma lógica, al aplicar el mismo algoritmo obtendremos el número original en binario.

Así, tenemos: $(10110101)_2$, el cuál pasará a ser $(01001011)_2$. Ahora, haciendo operaciones aritméticas básicas, tenemos: $1 * 2^6 + 1 * 2^3 + 1 * 2^1 + 1 * 2^0$ = 64 + 8 + 1 = 75. Por lo cuál, el número complemento a 2 se trataba del -75.

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.

Mismo truco, $(00110111)_2$ pasa a ser $(11001001)_2$, el cuál es la representación binaria del número en cuestión. Por lo cuál tenemos: $1*2^7+1*2^6+1*2^3+1*2^0$ = 128 + 64 + 8 + 1 = 201. Entonces el número en cuestión complemento a 2 era -201.

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

Para responder está pregunta, voy a basarme en el texto dado por el profe-

sor José Galaviz Casas y títulado como "Elementos cuantitativos de diseño de computadoras" donde menciona que Von Neumann, Eckert, Mauchly y Goldstine buscaban una mejor manera de proveer a una computadora del programa a ejecutar. De donde concluyen que el programa debe de tener la capacidad de ser almacenado, de ahí llevamos la primera unidad conocida como "memoria", después de eso, debía ser leída e interpretada por una entidad encargada de la ejecución, de ahí surge la segunda unidad llamada "unidad de control".

Posteriormente las instrucciones serían ejecutadas, con datos tomados de la misma memoria y la unidad ejecutante se llama unidad arítmetico lógica, la cuál es la tercera unidad. Por último, la cuarta unidad funcional principal, es la entrada/salida, esto debido a que tanto los datos como el programa debían proveerse a la computadora desde el exterior y los resultados, al final, debían ser enviados al exterior.

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

Primeramente, pasaremos los números de decimal a binario, y estos son: $(0011)_2 = 3$ y $(0110)_2 = 6$.

2	3	1
2	1	1

División respectiva del número 3.

2	6	0
2	3	1
2	1	1

División respectiva del número 6.

Ahora, aplicamos el truco para pasar los números binarios a complemento a 2, quedan de la respectiva forma: $(1101)_2$ = -3 y $(1010)_2$ = -6.

Y sumamos:

$$\begin{array}{r} 1101 \\ +1010 \\ \hline 10111 \end{array}$$

Pero como solo consideramos 4 bits, entonces el resultado queda: $(0111)_2$, para verificar que es correcto, pasaremos el resultado que está en complemento a 2 a binario, de este modo nos queda: $(1001)_2 = 1 * 2^3 + 1 * 2^0 = 8 + 1 = 9$. Por lo tanto, el número en cuestión era -9, lo cuál es correcto.

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

Vamos a pasar los números anteriormente mencionados a binario: $(1001)_2$ = 9 y $(0011)_2 = 3$.

2	3	1
2	1	1

División respectiva del número 3.

2	9	1
2	4	0
2	2	0
2	1	1

División respectiva del número 9.

Y ahora que tenemos dichos números en binario, solo es necesario realizar la suma:

$$\begin{array}{r} 1001 \\ +0011 \\ \hline 1100 \end{array}$$

Nota: Veamos que al sumar 1+1 en binario, sucede algo llamado "acarreo", esto debido a que la propia concepción de binario solo nos habla de 1's y 0's , sin embargo, es análogo a la propia suma decimal como la conocemos, si sumas dentro de las casillas un número mayor al 9, pones el último digito en cuestión y delegas las decenas excedentes a los dígitos de más adelante, en las siguientes preguntas aclararemos un poco más el propio concepto.

De este modo, tenemos al número binario $(1100)_2$, el cuál al pasarlo a decimal tenemos: $1 * 2^3 + 1 * 2^2 = 8 + 4 = 12$. Por lo tanto, el resultado es correcto.

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

10. Suma los siguientes dos números $(10011011)_2$ + (11101100)2. Explica qué sucede con el acarreo.

Primeramente para ver que sucede con el acarreo, es necesario ver que es "acarreo", el acarreo es la acción de "colocar" ó como popularmente se le conoce de "llevarse" un pedazo de la suma, recordemos que los números al ser binarios, solo pueden ser representados con 1's y 0's, y su valor es entorno a su posición, es decir que mientras más a la izquierda se encuentre el "1", más valor tendrá. Podemos ver con el apoyo de la tabla de arriba, que al sumar 0+1 = 1, 0+0= 0, 1+0=0 pero al sumar 1+1, este será igual a 0, con la diferencia de que en

realidad el resultado sería 2, y como anteriormente mencionabamos, mientras más a la izquierda este el "1", más valor tiene, y la forma de escribir 2 en binario es 0010. Análogo a lo conocido en sistema decimal, cortas las unidades y pasas las decenas, solo que en este caso, el 1 se suma a la siguiente parte de la suma. Veamos con un ejemplo:

$$\begin{array}{r} 10011011 \\ +11101100 \\ \hline 111110000 \\ \hline 110000111 \end{array}$$

Como vemos en la tercera línea, la cuál está representado con el color rojo, esos son los acarreos mencionados previamente. De donde al tener sumados 1+1 nos dará 10, de donde el "0" será colocado y el "1" es el acarreo.

11. Representa el número 39,1 en base 2 usando el estándar IEEE 754.

Pasaremos el número entero 39 a binario:

2	39	1
2	19	1
2	9	1
2	4	0
2	2	0
2	1	1

División respectiva del número 39.

De tal forma que nos queda: $(100111)_2$. Ahora, pasemos el punto decimal 0.1 a binario, para esto multiplicaremos hasta que se repita el ciclo o quede un entero. Así:

1. $0.1 \times 2 = 0.2$
2. $0.2 \times 2 = 0.4$
3. $0.4 \times 2 = 0.8$
4. $0.8 \times 2 = 1.6$
5. $0.6 \times 2 = 1.2$
6. $0.2 \times 2 = 0.4$

Obtención del decimal 0.1 a binario.

Por consiguiente, tenemos a $(000110...)_2$ como 0.1 en binario, por lo tanto el número en binario con todo y decimal queda como $(100111.000110...)_2 = 39.1$ y una vez obtenido como decimal, normalizamos:

$(1.00111000110...)_2$, al momento de normalizar (que es simplemente mover el punto hasta el primer uno del número de izquierda a derecha), tenemos que

contar cuantas veces se recorrió, esto es sencillo ya que el primer uno está representado con rojo, los números en azul son aquellos que también componían al 39 en binario, y los números en naranja eran los decimales del número original, es decir, el 0.1. De este modo, es fácil ver que el punto decimal fue recorrido 5 veces, llamemosle "n" al número de veces que se recorrió el punto.

Después de ello, es necesario recordar que el bit que representa al signo, es el de la izquierda, como estamos representando el número 39.1, este es positivo, por lo cuál el bit del signo es 0. Consiguiente a ello, calcularemos la representación del exponente, el cuál se hace sumando 127 al número de veces que se recorrió el punto, ó como anteriormente mencionabamos , sería 127 + n. Así 127+5 = 132. Ahora representemoslo en binario:

2	132	0
2	66	0
2	33	1
2	16	0
2	8	0
2	4	0
2	2	0
2	1	1

División respectiva del número 132.

Tenemos que el número binario del exponente queda como: $(10000100)_2$. Finalmente tenemos todos los componentes para representar el número dado en formato IEEE 754. Los cuales son el bit del signo, el exponente en 8 bits y la mantisa, la cuál no es más que el número 39.1 en binario pero con el punto recorrido hasta el primer uno, es decir: $(1.00111000110...)_2$. De donde la mantisa son los números en azul y naranja.

Por lo tanto la representación de 39.1 en formato IEEE 754 queda tal que: (01000010000111000110...) de donde bit del signo, exponente, mantisa.

12. Representa el número 576,65 en base 2 usando el estándar IEEE 754.

Una vez explicado detalladamente en el punto anterior, procederé a realizar la conversión. Primeramente 576 a binario:

2	576	0
2	288	0
2	144	0
2	72	0
2	36	0
2	18	0
2	9	1
2	4	0
2	2	0
2	1	1

División respectiva del número 576.

De este modo, queda $(1001000000)_2 = 576$. Ahora toca pasar el 0.65 a binario:

1. $0.65 \times 2 = 1.3$
2. $0.3 \times 2 = 0.6$
3. $0.6 \times 2 = 1.2$
4. $0.2 \times 2 = 0.4$
5. $0.4 \times 2 = 0.8$
6. $0.8 \times 2 = 1.6$
7. $0.6 \times 2 = 1.2$

Tenemos $(10100110)_2 = 0.65$ en binario, después normalizaremos:
$(1.00100000010100110...)_2$ y tenemos que "n" es igual a 9. El bit del signo es 0 (por ser un número positivo) y su exponente es 127+9 = 136. Así:

2	136	0
2	68	0
2	34	0
2	17	1
2	8	0
2	4	0
2	2	0
2	1	1

División respectiva del número 136.

Tenemos que $(10001000)_2 = 136$. La mantisa queda como $(00100000010100110...)_2$ por lo cuál la representación en formato IEEE 574 de 576.65 queda tal que (01000100000100000010100110...) de donde bit del signo, exponente, mantisa.

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

Una de las principales ventajas de la arquitectura de Von Neumann es que la Unidad de Control accede tanto a los datos como a las instrucciones desde una única memoria de manera unificada. Esto simplifica su diseño y desarrollo, lo que lo hace más económico y eficiente. Además, los datos provenientes de los dispositivos de entrada/salida y de la memoria se gestionan de forma similar, lo que contribuye a la simplicidad general del sistema.

En contraste con la misma arquitectura, esta presenta algunas fallas respecto a la secuencialidad de las instrucciones, esto deriva a que la implementación paralela de programas no esté permitida, además de eso, la segunda visible gran

falla es su cuello de botella, la cuál al agilizar y facilitar la recuperación de datos e instrucciones, esta solo puede ejecutarse una única vez y de manera secuencial.

14. La Arquitectura 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.

Uno de los cambios más significativos en la actualidad es la introducción del pipelining, una técnica que surge para superar las limitaciones de la arquitectura de Von Neumann al procesar datos simultáneamente. El pipelining permite la ejecución de múltiples instrucciones en paralelo, distribuyéndolas en diferentes etapas de procesamiento, lo que optimiza el rendimiento del sistema sin necesidad de aumentar la frecuencia de operación del procesador.

A esta mejora se suma el desarrollo de sistemas multiprocesador, los cuales emplean varios núcleos de CPU o procesadores interconectados para ejecutar tareas de manera paralela.

Por último, la adopción de la memoria caché ha sido usada para "parchar" el cuello de botella en el acceso a la memoria. La caché es una memoria pequeña y de alta velocidad que almacena temporalmente los datos e instrucciones más utilizados, reduciendo el tiempo de acceso a la memoria principal y mejorando el rendimiento general del sistema.

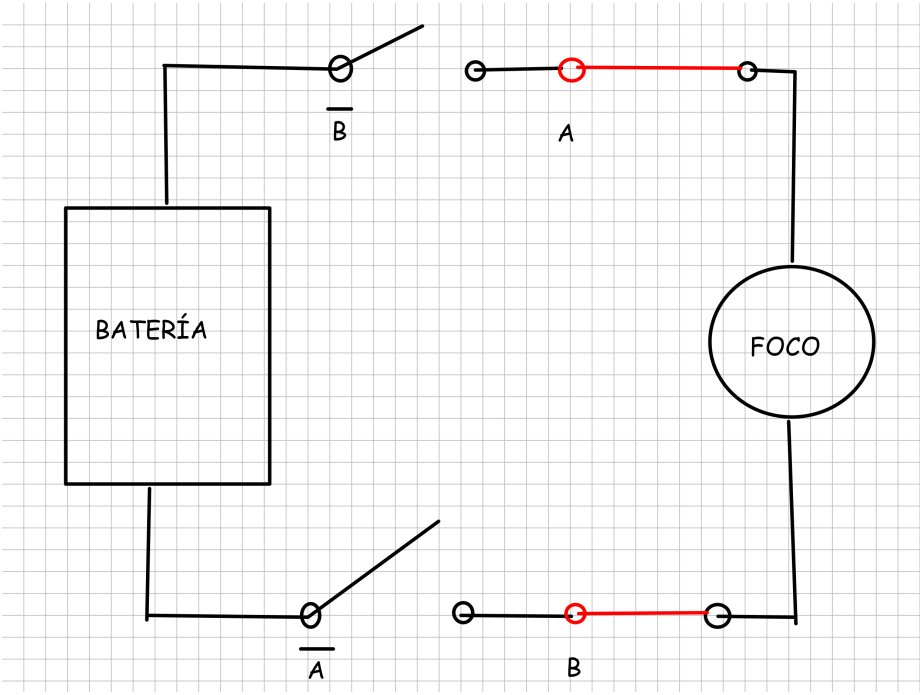
15. En la siguiente imagen, se nos muestra la disyunción y la conjunción proposicional usando interruptores. Usando ese mismo modelo ¿cómo sería un xor usando interruptores?

Recordemos que el XOR es una disyunción exclusiva, es decir, que el cambio ante la disyución que conocemos todos, es que si ambos valores son verdaderos, entonces retorna falso.

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Tabla respectiva del XOR.

Para la realización del diagrama, tomaremos 4 variables, las conocidas como A y B; además de sus "complementos", las cuales si A esta activa, entonces A complemento estará apagada y viceversa con B y B complemento.



0.1 Fuentes utilizadas

- Omega Academy. (2014, 11 septiembre). *Conversión de un número binario a formato IEEE 754* [Vídeo]. YouTube. https://www.youtube.com/watch?v=VlX4OlKvzAk
- Soluciones con el profe Marlon. (2022, 24 marzo). *Convertir número real a IEEE 754 Precisión Simple* [Vídeo]. YouTube. https://www.youtube.com/watch?v=cQ05pHCXH7w
- Pasos por ingeniería. (2022, 21 julio). *SUMA BINARIA CON SIGNO (números binarios POSITIVOS Y NEGATIVOS) 2* [Vídeo]. YouTube. https://www.youtube.com/watch?v=BuzAzuNHYPY
- Iannuttall. (2024, 24 julio). *Von Neumann Architecture — History & Use — Computer Science*. Teach Computer Science. https://teachcomputerscience.com/von-neumann-architecture/
- *Von Neumann Architecture: Meaning, Examples & Features*. (s. f.). Vaia. https://www.vaia.com/en-us/explanations/computer-science/computer-organisation-and-arch/von-neumann-architecture/
- Latam, M. (2021, 2 mayo). *Compuerta XOR*. Mecatrónica LATAM. https://www.mecatronicalatam.com/es/tutoriales/electronica/compuertas-logicas/compuerta-xor/