Algoritmo de Booth: Multiplicación binaria con signo

Curso de Diseño Lógico EL3307

Introducción

El algoritmo de Booth es un método utilizado para realizar multiplicaciones binarias de números con signo, especialmente en sistemas digitales que representan los números en complemento a dos. Fue desarrollado por Andrew Booth en 1951 y su principal objetivo es reducir la cantidad de operaciones necesarias durante la multiplicación, aprovechando las secuencias de bits repetidos del multiplicador.

Principio de funcionamiento

El método se basa en analizar pares de bits consecutivos del multiplicador: el bit menos significativo actual (Q_0) y un bit adicional llamado Q_{-1} que inicialmente vale 0. A partir de este par de bits, el algoritmo decide si se debe sumar, restar o no hacer nada con el multiplicando. Posteriormente, se realiza un desplazamiento aritmético hacia la derecha (ASR) para preparar el siguiente paso.

Registros utilizados

Los registros que intervienen en el proceso son los siguientes:

- A: acumulador, donde se almacenan los resultados parciales.
- Q: multiplicador.
- M: multiplicando.
- Q_{-1}: bit adicional de control.
- Contador: lleva el número de iteraciones, equivalente al número de bits.

Antes de comenzar, se inicializa $A = 0$, Q con el valor del multiplicador, M con el valor del multiplicando y $Q_{-1} = 0$. Luego, el proceso se repite tantas veces como bits tenga el multiplicador.

Reglas del algoritmo

En cada iteración se observan los bits (Q_0, Q_{-1}) y se aplica la siguiente regla:

(Q_0, Q_{-1})	Operación
(0,0)	No hacer nada
(0,1)	$A = A + M$
(1,0)	$A = A - M$
(1,1)	No hacer nada

Después de realizar la operación correspondiente, se efectúa un desplazamiento aritmético a la derecha (ASR) sobre la concatenación (A, Q, Q_{-1}). Este desplazamiento conserva el bit de signo (MSB de A) y mueve todos los bits una posición a la derecha, de modo que el nuevo Q_{-1} toma el valor del bit menos significativo de Q. Finalmente, tras realizar tantas iteraciones como bits tenga el multiplicador, el resultado completo de la multiplicación se obtiene uniendo los contenidos de A y Q.

Desplazamiento aritmético a la derecha (ASR)

El desplazamiento aritmético a la derecha (ASR) tiene la función de preservar el signo del número. A diferencia del desplazamiento lógico, que siempre introduce ceros en el bit más significativo, el ASR repite el bit de signo, garantizando que los números negativos sigan siendo negativos y los positivos sigan siendo positivos. Este comportamiento permite que el algoritmo funcione correctamente con números representados en complemento a dos.

Eficiencia del algoritmo

El algoritmo de Booth es eficiente porque, en lugar de sumar repetidamente el multiplicando por cada 1 del multiplicador, detecta secuencias de bits iguales. Por ejemplo, una secuencia de 1111 en el multiplicador se puede representar mediante una sola resta y una suma, lo que reduce significativamente el número de operaciones necesarias.

Ventajas y desventajas

Ventajas:

- Funciona correctamente con números con signo.
- Reduce el número de operaciones para multiplicadores con secuencias largas de bits iguales.
- Es adecuado para implementaciones en procesadores y sistemas digitales.

Desventajas:

- Requiere un control lógico más complejo.
- Su eficiencia depende del patrón de bits del multiplicador.

Conclusión

El algoritmo de Booth permite multiplicar números binarios con signo de manera eficiente utilizando una combinación de sumas, restas y desplazamientos aritméticos. El resultado final se obtiene después de procesar todos los bits del multiplicador, conservando la corrección del signo y reduciendo el número de operaciones necesarias en comparación con la multiplicación binaria convencional.

Ejemplo Conceptual del Algoritmo de Booth: Multiplicación 2×6

Datos iniciales

Se desea multiplicar:

$$2 \times 6$$

Trabajando con números de 4 bits en complemento a dos:

$$M = 0010 \quad (2_{10})$$
$$Q = 0110 \quad (6_{10})$$
$$A = 0000, \quad Q_{-1} = 0$$

El contador inicial es 4, correspondiente al número de bits del multiplicador.

Reglas de decisión

En cada paso se observan los bits (Q_0, Q_{-1}):

(Q_0, Q_{-1})	Acción
(0,0)	No hacer nada
(0,1)	$A = A + M$
(1,0)	$A = A - M$
(1,1)	No hacer nada

Después de aplicar la operación, se realiza un desplazamiento aritmético a la derecha (ASR) sobre la combinación (A, Q, Q_{-1}).

Iteraciones del algoritmo

Ciclo	(Q_0, Q_{-1})	Operación	A antes	A después	(A, Q, Q_{-1}) antes del ASR	(A, Q, Q_{-1}) después del ASR
Inicial	—	—	0000	—	A=0000, Q=0110, Q_{-1}=0	—
1	(0,0)	No hacer nada	0000	0000	0000 0110 0	0000 0011 0
2	(1,0)	$A = A - M = 0000 - 0010 = 1110$	0000	1110	1110 0011 0	1111 0001 1
3	(1,1)	No hacer nada	1111	1111	1111 0001 1	1111 1000 1
4	(0,1)	$A = A + M = 1111 + 0010 = 0001$	1111	0001	0001 1000 1	0000 1100 0

Resultado final

Al finalizar las cuatro iteraciones:

$$A = 0000, \quad Q = 1100$$

Concatenando ambos:

$$(A, Q) = 0000\ 1100 = 00001100_2$$

En decimal:

$$00001100_2 = 12_{10}$$

Por tanto:

$$2 \times 6 = 12$$

Explicación del proceso

1. Primer ciclo: $(Q_0, Q_{-1}) = (0, 0)$. No se realiza ninguna operación, luego se aplica el desplazamiento aritmético.
2. Segundo ciclo: $(Q_0, Q_{-1}) = (1, 0)$. Cambio de 0 a 1, se resta el multiplicando. $A = 1110$. Se realiza el desplazamiento.

3. Tercer ciclo: $(Q_0, Q_{-1}) = (1, 1)$. No hay cambio, no se hace nada. Solo se desplaza.

4. Cuarto ciclo: $(Q_0, Q_{-1}) = (0, 1)$. Cambio de 1 a 0, se suma el multiplicando. $A = 0001$. Luego del desplazamiento, se obtiene el resultado final.

Interpretación conceptual

El multiplicador 0110_2 (6) puede analizarse mediante las transiciones de bits. El cambio de 0 a 1 en el segundo bit indica el inicio de una secuencia de unos, lo que provoca una resta del multiplicando ($A = A - M$). El cambio de 1 a 0 en el bit más significativo marca el final de la secuencia, generando una suma del multiplicando ($A = A + M$).

De esta forma, el patrón 0110_2 equivale a:

$$2^3 - 2^1 = 8 - 2 = 6$$

Por tanto, el algoritmo de Booth interpreta la secuencia como una *resta seguida de una suma*, y no como múltiples sumas. El resultado final es:

$$M \times (2^3 - 2^1) = 2 \times (8 - 2) = 12$$

lo cual coincide con el producto esperado $2 \times 6 = 12$.

Conclusión

El algoritmo de Booth obtiene el mismo resultado que la multiplicación binaria tradicional, pero de forma más eficiente, detectando las secuencias de bits del multiplicador y aplicando solo las operaciones necesarias. Los desplazamientos aritméticos garantizan la posición correcta de los resultados parciales y la conservación del signo. En este ejemplo, después de cuatro iteraciones, el producto final 2×6 se obtiene correctamente como $00001100_2 = 12_{10}$.

Diseño de Máquinas de Estados con Diagramas ASM

El **diseño de máquinas de estados con diagramas ASM (Algorithmic State Machine)** es una forma gráfica de describir el comportamiento de una máquina secuencial, similar a un diagrama de flujo de programación más que a un diagrama de estados tradicional.

Los diagramas ASM utilizan tres tipos principales de elementos [1]:

1. **State box (cuadro de estado):**
   Cada cuadro representa un estado de la máquina, mostrando su nombre y, opcionalmente, el código o las salidas de tipo Moore que se activan en ese estado. La principal diferencia con un nodo de un diagrama de estados convencional es que el cuadro de estado tiene una única salida que indica la transición hacia el siguiente estado, representada por una flecha que conduce a un cuadro de decisión o a otro cuadro de estado.

2. **Decision box (cuadro de decisión):**
   Divide una transición en dos posibles rutas basadas en una expresión lógica (condición). Si la condición es verdadera (1), se sigue la ruta etiquetada con "1"; si es falsa (0), se sigue la ruta etiquetada con "0". Cada salida puede conducir a un cuadro de estado o a otro cuadro de decisión, y pueden colocarse varios cuadros de decisión en serie cuando existen múltiples condiciones.

3. **Conditional output box (cuadro de salida condicional):**
   Se coloca sobre una ruta de salida de un cuadro de decisión para representar salidas de tipo *Mealy*, que dependen tanto del estado actual como de las entradas. Aunque en el diagrama parezca que estas salidas se activan solo en el instante del flanco de reloj cuando ocurre la transición, en realidad permanecen activas durante todo el ciclo de reloj mientras las condiciones correspondientes sean verdaderas.

En conjunto, los diagramas ASM combinan características de los diagramas de estados y de los diagramas de flujo, proporcionando una representación clara y estructurada del comportamiento de máquinas secuenciales digitales.

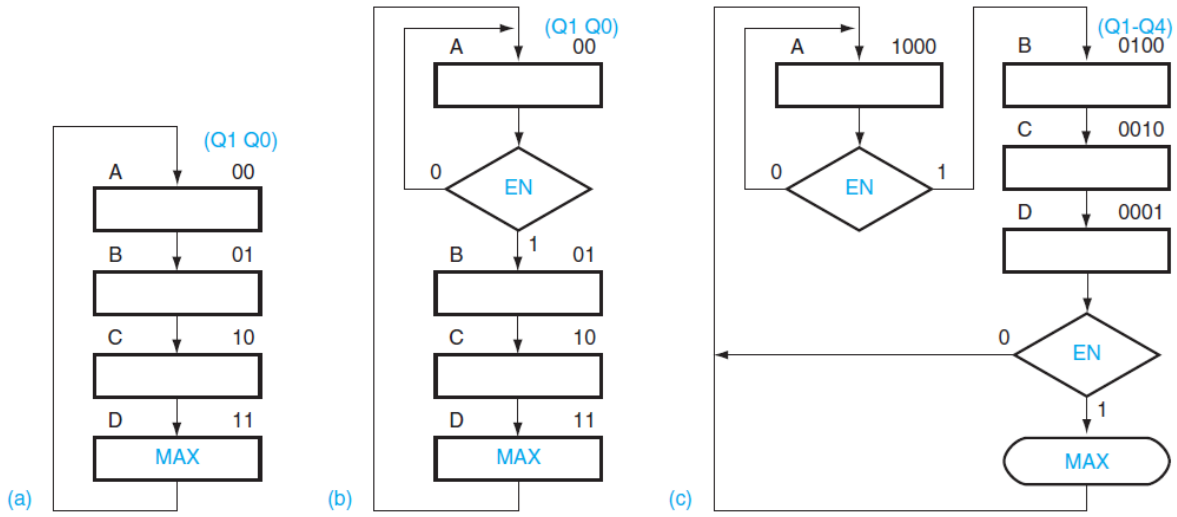

Figura 1: Diagramas ASM. (a) contador módulo-4 de funcionamiento libre. (b) contador módulo-4 con habilitación. (c) contador módulo-4 con una salida de tipo Mealy.

Ejercicio 1

Hacer el diagrama ASM del algoritmo de Booth.

Ejercicio 2

Algoritmo de división

La división se puede implementar por medio del siguiente algoritmo [2]:

```
R' = 0
for i = N-1 to 0
   R = {R' << 1, Ai}
   D = R - B
   if D < 0 then Qi = 0, R' = R   // R < B
   else          Qi = 1, R' = D   // R ≥ B
R = R'
```

Figura 2: Algoritmo de división.

El **algoritmo de división binaria** para números sin signo de N bits permite obtener el cociente Q y el residuo R mediante un proceso secuencial de desplazamientos y restas controladas.

El procedimiento parte de inicializar el *resto parcial* $R' = 0$. Luego, se toman los bits del *dividendo* A, uno por uno, desde el más significativo hasta el menos significativo. En cada ciclo, el valor actual de R' se desplaza una posición a la izquierda y se añade el bit correspondiente de A, formando un nuevo valor temporal:

$$R = \{R' \ll 1, A_i\}$$

A continuación, se **resta el divisor** B de este nuevo valor parcial:

$$D = R - B$$

Si el resultado D es negativo (el bit de signo es 1), significa que B no cabe en R. En ese caso, el bit de cociente $Q_i = 0$ y el valor del resto parcial se mantienen sin cambios ($R' = R$). Si el resultado D es positivo o cero ($D \geq 0$), el divisor sí cabe, por lo que se asigna $Q_i = 1$ y el nuevo resto parcial se actualiza a $R' = D$.

El proceso se repite hasta procesar todos los bits del dividendo, generando así el cociente completo Q y el residuo final R. La relación final se cumple como:

$$A = Q \cdot B + R$$

o, equivalentemente:

$$\frac{A}{B} = Q + \frac{R}{B}$$

Ejercicio Propuesto

Diseñe e implemente el **diagrama ASM (Algorithmic State Machine)** correspondiente al algoritmo de división binaria descrito.
El diagrama ASM debe mostrar claramente la secuencia de operaciones y decisiones que permiten ejecutar el algoritmo de división binaria paso a paso.

Referencia

Para una visualización interactiva del algoritmo, puede consultarse el siguiente enlace: https://dhruvpatel004.github.io/Booth-Multiplication-Algorithm/#hero

Referencias

[1] Wakerly, J. F. (2018). *Digital Design: Principles and Practices* (5th ed.). Pearson, New York.

[2] Harris, D. M., & Harris, S. L. (2021). *Digital Design and Computer Architecture: RISC-V Edition*. Morgan Kaufmann, Cambridge, MA.

[3] Patel, D. (2023). *Booth Multiplication Algorithm*. Disponible en: https://dhruvpatel004.github.io/Booth-Multiplication-Algorithm/#hero (Consultado el 21 de octubre de 2025).