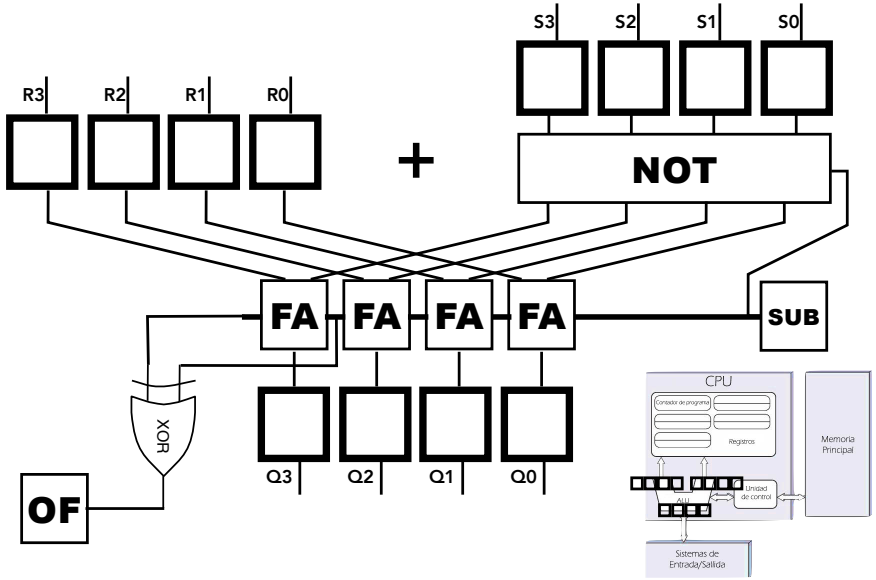
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS



ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN

Multiplicando

Multiplicador

	0	1	1	1	7
*	0	1	1	1	7

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN

	0	1	1	1	7
*	0	1	1	1	7

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN

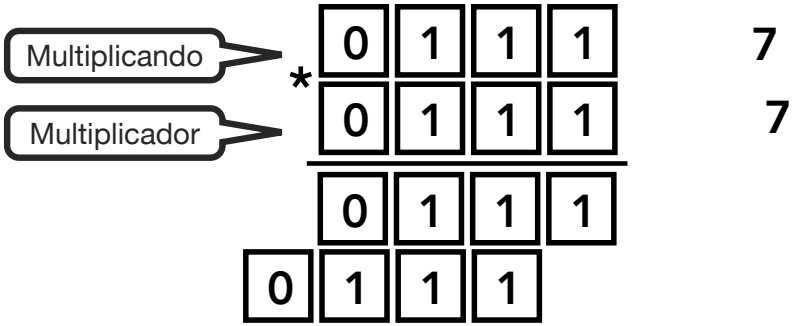
Multiplicando

Multiplicador

	0	1	1	1	7
*	0	1	1	1	7
	0	1	1	1	

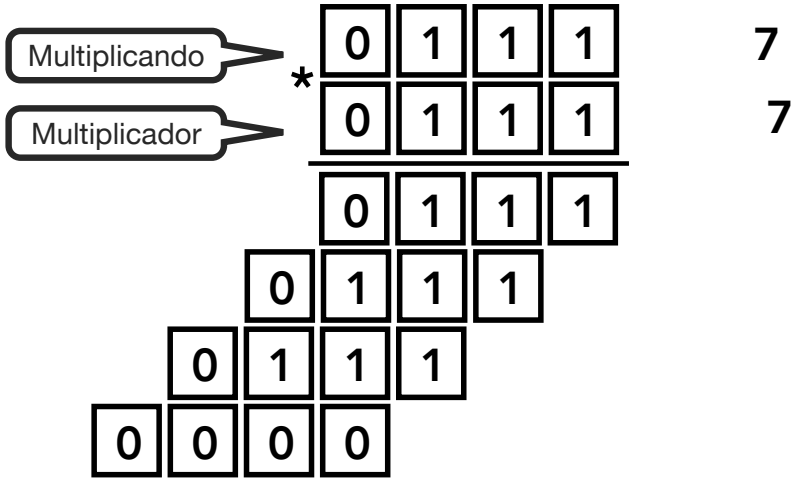
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



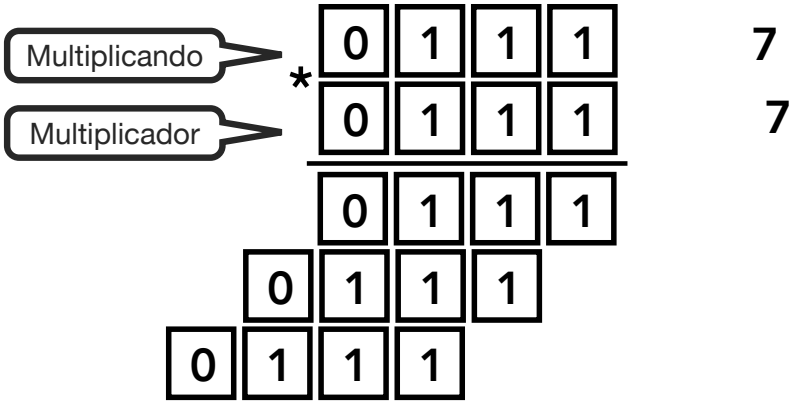
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



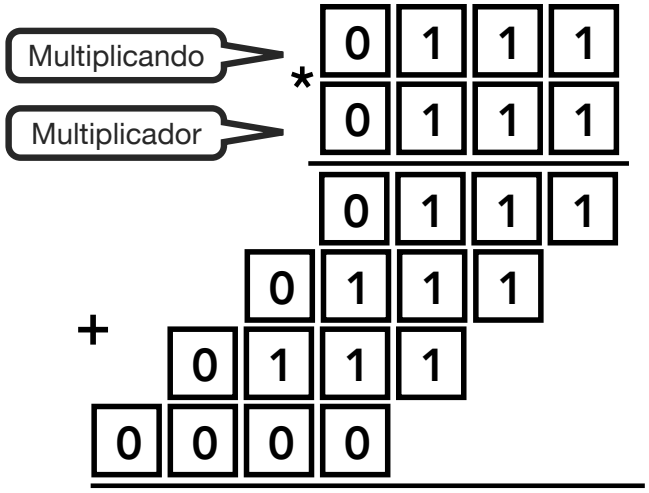
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



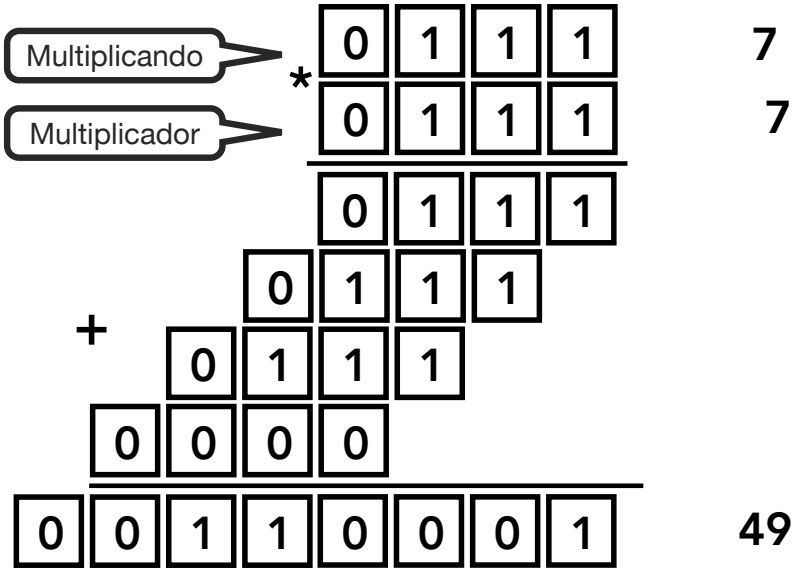
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



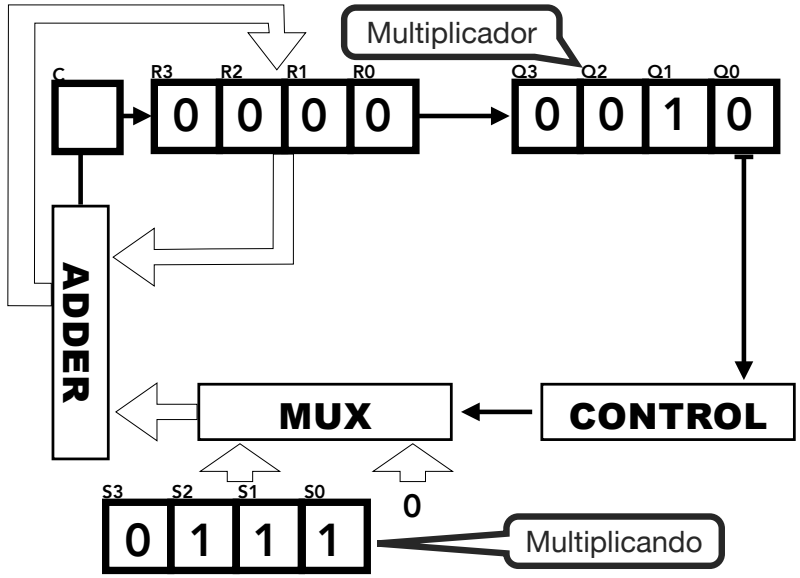
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



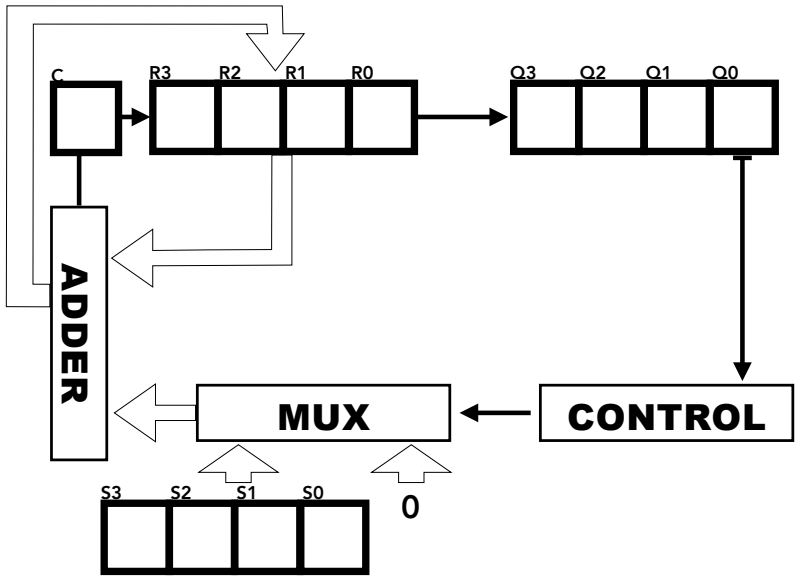
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



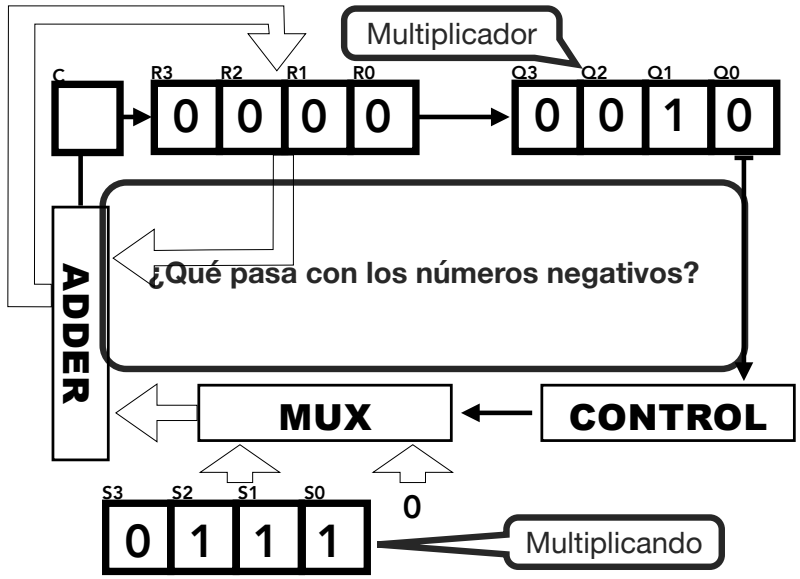
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



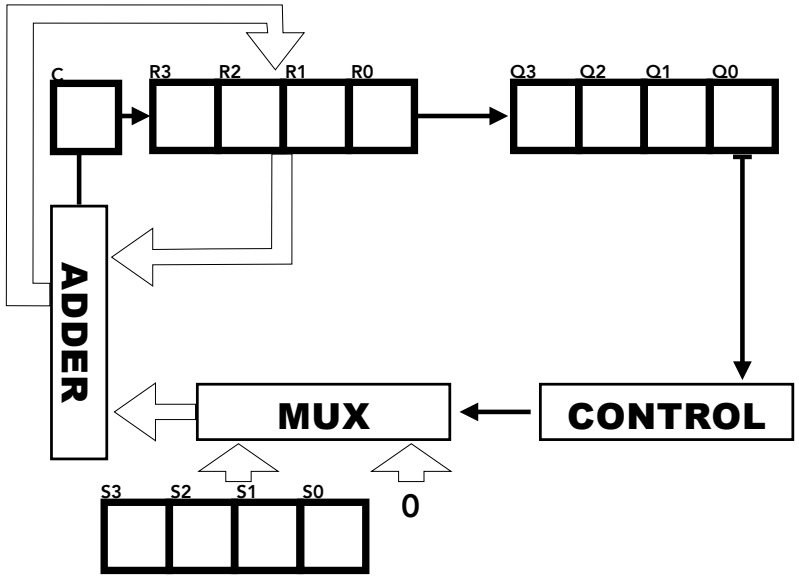
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



ORGANIZACIÓN DEL PROCESADOR

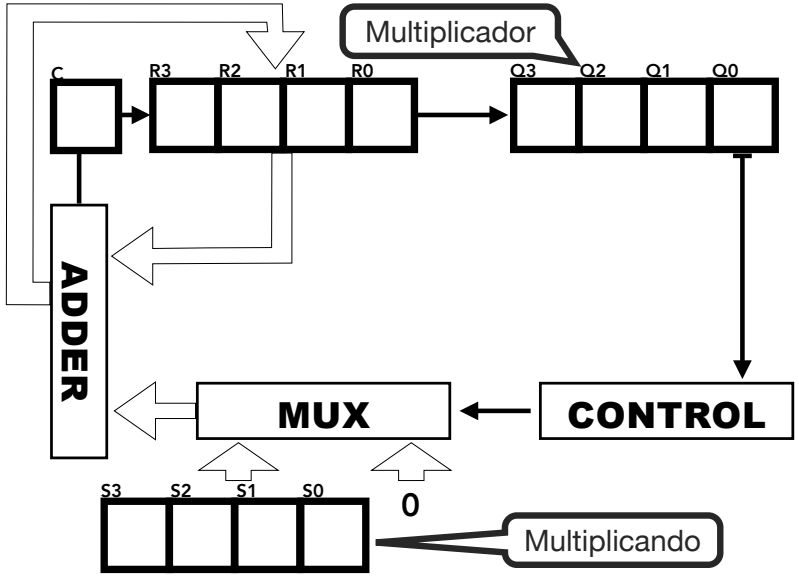
OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN

Multiplicación de enteros general (positivos y negativos):

- El resultado necesita un tamaño de bits equivalente a la suma de los operandos.
- El proceso calcula productos parciales y depende de los siguientes casos:
  - (a) **+ Multiplicando * + Multiplicador** : Se comienza con el producto parcial igual a 0 y el bit de *carry out* en 0. Se recorre el multiplicador de derecha a izquierda. Si es un 0 se *shiftea* ***con el carry out*** *a derecha* el producto parcial. Si es un 1, al producto parcial se le suma el multiplicando y luego se *shiftea* ***con el carry out*** *a derecha* el producto parcial.
  - (b) **- Multiplicando * + Multiplicador** : El proceso es igual que el caso anterior, pero el bit de *carry out* comienza con 1, y en cada operación de *shift a derecha* se mantiene el *carry out* con 1 (se extiende el complemento del número)
  - (c) **+ Multiplicando * - Multiplicador** : se complementan ambos números y se procede como en (b)
  - (d) **- Multiplicando * - Multiplicador** : se complementan ambos números y se procede como en (a)

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN



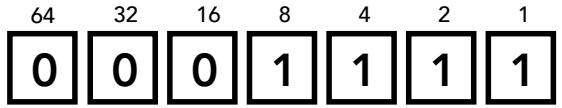
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

La idea del algoritmo de Booth es la de no operar por casos y la de tratar de realizar la menor cantidad posible de sumas (es más costoso que *shiftear*)

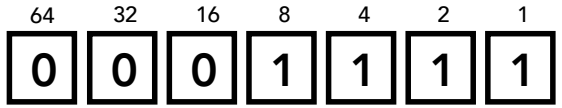
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH



ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

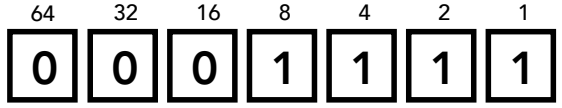


8 + 4 + 2 + 1 = 15

16 + 0 + 0 + 0 - 1 = 15

ORGANIZACIÓN DEL PROCESADOR

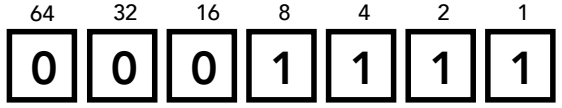
OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH



8 + 4 + 2 + 1 = 15

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

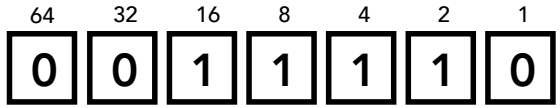


8 + 4 + 2 + 1 = 15

16 + 0 + 0 + 0 - 1 = 15

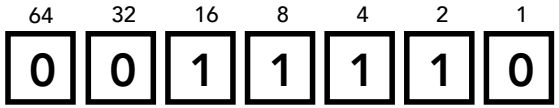
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH



ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

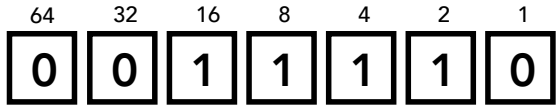


$16+8+4+2+0=30$

$32+0+0+0-2+0=30$

ORGANIZACIÓN DEL PROCESADOR

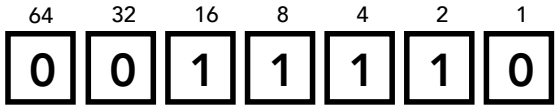
OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH



$16+8+4+2+0=30$

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH



$16+8+4+2+0=30$

$32+0+0+0-2+0=30$

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

128	64	32	16	8	4	2	1
0	1	1	1	0	1	1	1

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

128	64	32	16	8	4	2	1
0	1	1	1	0	1	1	1

$$64 + 32 + 16 + 0 + 4 + 2 + 1 = 119$$

$$128 + 0 + 0 - 16 + 8 + 0 + 0 - 1 = 119$$

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

128	64	32	16	8	4	2	1
0	1	1	1	0	1	1	1

$$64 + 32 + 16 + 0 + 4 + 2 + 1 = 119$$

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULT. BOOTH

128	64	32	16	8	4	2	1
0	1	1	1	0	1	1	1

$$64 + 32 + 16 + 0 + 4 + 2 + 1 = 119$$

$$128 + 0 + 0 - 16 + 8 + 0 + 0 - 1 = 119$$

1	0	0	-1	1	0	0	-1

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN

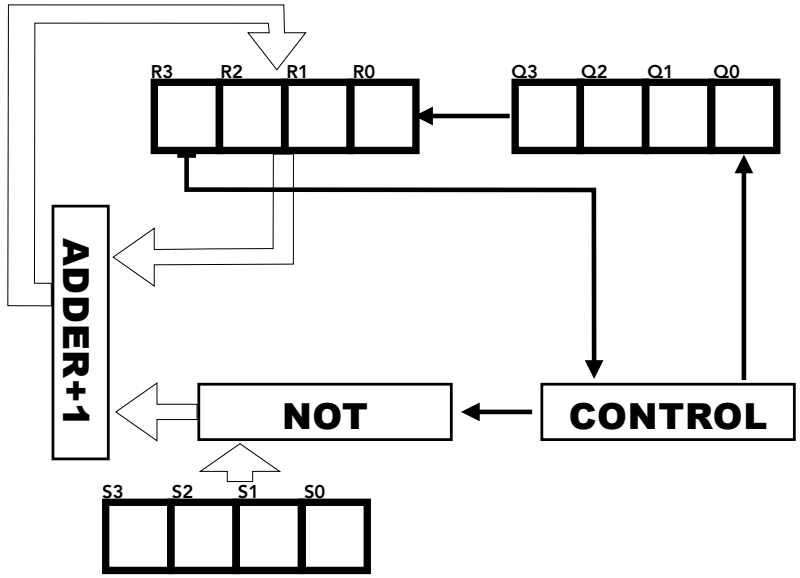
Algoritmo de Booth:

- El resultado necesita un tamaño de bits equivalente a la suma de los operandos.
- El proceso calcula productos parciales de manera similar que el algoritmo general pero en vez de mirar el último bit del **multiplicador** mira los últimos 2 y de acuerdo a los siguientes casos opera sobre el producto parcial:

0 0	Se shiftea a derecha
0 1	Se suma el multiplicando
1 0	Se suma el complemento a la base (negativo) del multiplicando
1 1	Se shiftea a derecha respetando la extensión de signo

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - DIVISIÓN



ORGANIZACIÓN DEL PROCESADOR

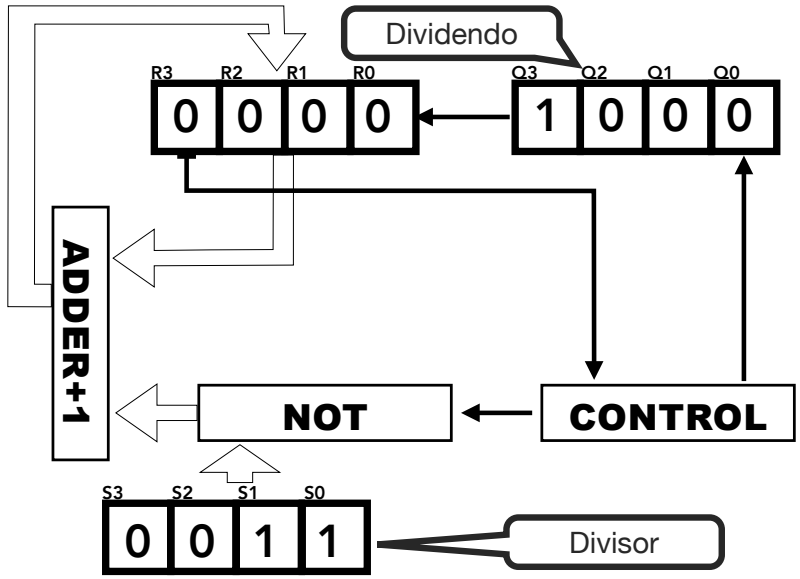
OPERACIONES CON NÚMEROS ENTEROS - MULTIPLICACIÓN

Algoritmo de Booth:

- La cantidad de operaciones (sumas) optimizadas es proporcional a la cantidad de secuencias contiguas de 1 en la representación del multiplicador.
- Existe un caso donde empeora.

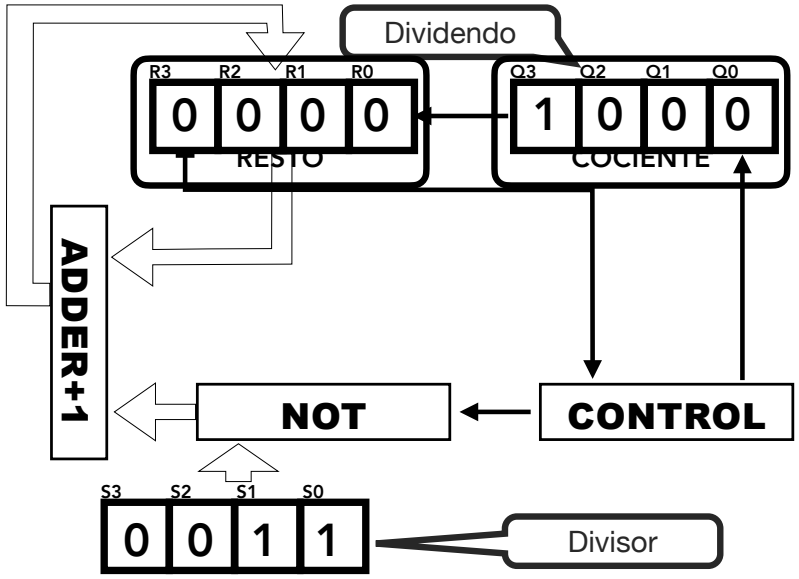
ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - DIVISIÓN



ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - DIVISIÓN



ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - DIVISIÓN

Algoritmo de la división:

Repetir ***n*** (*cantidad de bits*) veces los siguientes pasos:

(1) *Shiftear a izquierda* una vez los registro correspondientes al **Dividendo** (Q) con **0** y el **registro auxiliar** (R) con el bit más significativo de Q.

(2) Dejar en el registro auxiliar el resultado de restarle al mismo el **Divisor** (S). Notar que se suma el Comp.Base de (S)

(3) Si el signo (***bit más*** significativo del **registo auxiliar** R) es 1 (negativo), cambiar el ***bit menos*** significativo de **Dividendo** (Q_0) a 0 y restaurar el **registo auxiliar** sumando el **Divisor** (S), sino cambiar el ***bit menos*** significativo de **Dividendo** (Q_0) a 1.

ORGANIZACIÓN DEL PROCESADOR

OPERACIONES CON NÚMEROS ENTEROS - DIVISIÓN

Algoritmo de la división: (restauración)

Repetir ***n*** (*cantidad de bits*) veces los siguientes pasos:

(1) *Shiftear a izquierda* una vez los registro correspondientes al **Dividendo** (Q) con **0** y el **registro auxiliar** (R) con el bit más significativo de Q.

(2) Dejar en el registro auxiliar el resultado de restarle al mismo el **Divisor** (S). Notar que se suma el Comp.Base de (S)

(3) Si el signo (***bit más*** significativo del **registo auxiliar** R) es 1 (negativo), cambiar el ***bit menos*** significativo de **Dividendo** (Q_0) a 0 y restaurar el **registo auxiliar** sumando el **Divisor** (S), sino cambiar el ***bit menos*** significativo de **Dividendo** (Q_0) a 1.