Firmware Division

Prof. Alberto Borghese
Dipartimento di Informatica
borghese@di.unimi.it

Università degli Studi di Milano

Riferimenti sul Patterson 5a ed.: 3.4, 3.5

Sommario

- Divisione intera
- Circuiti divisione intera
- Divisione e moltiplicazione

Moltiplicazione

128 x
214 =
----
512 + $x10^0$
128- + $x10^1$
256-- = $x10^2$

27392

$128x(4x10^0)+$ unità
$128x(1x10^1)+$ decine
$128x(2x10^2) =$ centinaia
27392

Prodotto come **somma dei prodotti parziali**. Ognuno dei prodotti parziali aggiunge il **moltiplicando pesato con il peso della cifra analizzata del moltiplicatore** x un numero di volte pari alla cifra analizzata del moltiplicatore.

Nel caso binario le cifre sono solo 0,1 ===➔

Algoritmi per la moltiplicazione

Il razionale degli algoritmi firmware della moltiplicazione è il seguente.

Si analizzano sequenzialmente i bit del moltiplicatore e:

1) Si mette 0 nella posizione opportuna (se il bit analizzato del moltiplicatore = 0).
2) Si mette una copia del moltiplicando nella posizione opportuna (se il bit analizzato del moltiplicatore è = 1).

Moltiplicando 1 1 0 1 1 x
Moltiplicatore 1 0 1 =
----
1 1 0 1 1 +
0 0 0 0 0 -
----
1 1 0 1 1
1 1 0 1 1 - -

Prodotto 1 0 0 0 0 1 1 1

La moltiplicazione viene effettuata come somme successive, con peso crescente, di uno tra i 2 valori: {moltiplicando, 0}

La divisione decimale

Dividendo

Divisore

Quoziente

```
 __ - -
3716 : 12 = 309
36--
-------
  116
    0-
------
  116
  108
  -----
      8
```

Resto

Dividendo = Divisore * Quoziente (quoto) + Resto

Razionale della divisione

```
 __ - -
3716 : 12 = 309
36--
-------
  116
    0-
------
  116
  108
  -----
      8
```

1. Analizzo il moltiplicatore

2. Identifico quale potenza di 10 iniziare a considerare. Questa è la **massima potenza, p,** tale per cui 10^p moltiplicata per il moltiplicatore mi dà un valore inferiore al divendo.

In questo caso:
p = 1 –> 10 * 12 = 120
p = 2 -> 100 * 12 = 1200
p = 3 – 1000*12 = 12000.

3. Determino qual è il **numero massimo di volte, q_2,** per cui il moltiplicatore x 10^p può essere moltiplicato per ottenere un numero inferiore al moltiplicando.

In questo caso:
q_2 = 1 -> 1 * 1200 = 1200
q_2 = 2 -> 2 * 1200 = 2400
q_2 = 3 -> 3 * 1200 = 3600
q_2 = 4 -> 4 * 1200 = 4800

NB la moltiplicazione x 10^p si sottoindende, allineando il q_2 x moltiplicatore alla cifra opportune. In questo caso 3x12 = 36 allineato alle centinaia.

Razionale della divisione

```
 __ - -
3716 : 12 = 309
36--
------
 116
  0-
------
 116
 108
 -----
   8
```

4. Ho stabilito p = 2, q_2 = 3 ➔ posso **erodere una quantità pari a: 3 * 12 * 10^2 = 3600**

3716 -	q_2 = 3 è la cifra del quoziente cioè di quanto stiamo erodendo
3600 =	p = 2 ➔ 10^2 è il peso del moltiplicatore

116	Resto parziale

Procedo ricorsivamente ripetendo i passi 2-4 sul resto fino a quando non ottengo un resto inferiore al divisore.

I passi successive al primo sono leggermente diversi.

Sviluppo della divisione - I

```
 __ - -
3716 : 12 = 309
36--
------
 116
  0-
------
 116
 108
 -----
   8
```

1. Analizzo il moltiplicatore

2. Considero la potenza p = **p-1.** La potenza 1 in questo caso.
p = 1 –> 10^1 * 12 = 120

3. Determino qual è il **numero massimo di volte, q_1,** per cui il moltiplicatore x 10^p può essere moltiplicato per ottenere un numero inferiore al moltiplicando.
In questo caso:
q_1 = 0 -> 0 * 120 = 0
q_1 = 1 -> 1 * 120 = 120

4. Ho stabilito p = 1, q_1 = 0 ➔ posso **erodere una quantità pari a: 0 * 12 * 10^1 = 0**

116 -	q_1 = 0 è la cifra del quoziente cioè di quanto stiamo erodendo
0- =	p = 1 ➔ 10^1 è il peso del moltiplicatore

116	Resto parziale

Sviluppo della divisione - II

```
3716 : 12 = 309
36--
-------
 116
  0-
------
 116
 108
 ---
   8
```

1. Analizzo il moltiplicatore

2. Considero la potenza p=**p-1, 0 in questo caso.**
   p = 0 –> 1 * 12 = 12

3. Determino qual è il **numero massimo di volte, q_0,** per cui il moltiplicatore x 10^p può essere moltiplicato per ottenere un numero inferiore al moltiplicando.
   In questo caso:
   q_0 = 8 -> 8 * 12= 96
   q_0 = 9 -> 9 * 12 = 108

4. Ho stabilito p = 0, q_0 = 9 ➔ posso **erodere una quantità pari a: 9 * 12 * 10^0 = 108**

116 -	q_0 = 9 è la cifra del quoziente cioè di quanto stiamo erodendo
108 =	p = 0 ➔ 10^0 è il peso del moltiplicatore

8	Resto totale

Razionale della divisione decimale

La divisione è l'operazione inversa del prodotto

2629 : 12 = 219 **(resto = 1)** ➔ 219 x 12 + 1 = 2629

[**($2 x 10^2$)** + **($1 x 10^1$)** + **($9 x 10^0$)**] x 12 **+ 1** = 2629

Da cui segue che ad ogni passo della divisione erodiamo una quantità via via decrescente. Al primo passo sottraiamo a entrambi i membri:

1. [**($1 x 10^1$)** + **($9 x 10^0$)**] x 12 +1 = 2629 **– ($2 x 10^2$) x 12** = 2629 – 2400 = 229 -> resto parziale.
   Abbiamo cioè eroso 24 centinaia al dividendo 2629.

2. Al passo successivo eroderemo al resto parziale 12 decine:
[**($9 x 10^0$)**] x 12 + 1 = 229 – **($1 x 10^1$) x 12** = 229 – 120 = 109 -> resto parziale

3. E al passo finale eroderemo 12x9=108 unità:
[1 = 109 – **($9 x 10^0$) x 12** = 1 -> **resto finale.**

Ad ogni passo si cerca di erodere il più possibile!

Confronto con la moltiplicazione

La divisione è l'operazione inversa del prodotto

2629 : 12 = 219 **(resto = 1)** ➔ 219 x 12 + 1 = 2629

Divisione: Ad ogni passo erodiamo (sottraiamo) un certo numero di volte il divisore moltiplicato per la massima cifra possibile (cifra del quoziente), moltiplicato per la potenza associata alla posizione della cifra del dividend, che stiamo considerando.

Opera per sottrazioni successive di quantità sempre più piccole proporzionali al divisore, producendo via via i resti parziali.

Moltiplicazione: ad ogni passo sommiamo il moltiplicando moltiplicato per il numero di volte indicato dalla cifra del moltiplicatore considerata e per la potenza associata alla posizione della cifra del moltiplicatore, che stiamo considerando.

Opera per somme successive di quantità sempre più grandi proporzionali al moltiplicatore, producendo via via le somme parziali.

La divisione tra numeri binari

Divisione decimale fra i numeri a = 1.001.010 e b = 1.000 a : b = ?

```
____
1001010 : 1000 = 1001          Dividendo : Divisore     74 : 8 = 9  resto 2
1000 -
-----                          Quoziente
   1010 -
  0000  =
---------                      Resti parziali
   1010 -
   0000 =
---------
   1010 –
   1000 =
----------
     10      Resto
```

Dividendo = Quoziente x Divisore + Resto

Confronto tra divisione tra numeri binari e decimali

In DECIMALE:

Ad ogni passo devo verificare QUANTE VOLTE il resto parziale contiene il divisore. Il risultato è un numero che va da 0 a 9 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. 0 = il divisore non è contenuto nel resto.

In BINARIO:

Ad ogni passo devo verificare SE il resto parziale contiene il divisore. Ovverosia se lo contiene 0 o 1 volta. Il risultato è un numero che può valere {0, 1}.

```
__ - -
3716 : 12 = 309
36--
-------
 116
   0-
------
 116
 108
-----
   8
```

```
____
1001010 : 1000 = 1001
1000 -
-----
   1010 -
 0000    =
---------
   1010 -
  0000   =
---------
  1010 –
  1000 =
----------
    10
```


Confronto tra divisione tra numeri binari e decimali

In DECIMALE:

Ad ogni passo devo verificare QUANTE VOLTE il resto parziale contiene il divisore. Il risultato è un numero che va da 0 a 9 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. 0 = il divisore non è contenuto nel resto.

In BINARIO:

Ad ogni passo devo verificare SE il resto parziale contiene il divisore. Ovverosia se lo contiene 0 o 1 volta. Il risultato è un numero che può valere {0, 1}.

```
__ - -
3716 : 12 = 309
36--
-------
 116
   0-
------
 116
 108
-----
   8
```

```
____
1001010 : 1000 = 1001
1000 -
-----
   1010 -
 0000    =
---------
   1010 -
  0000   =
---------
  1010 –
  1000 =
----------
    10
```

Peculiarità della divisione

- Come rappresento la condizione “il divisore è contenuto nel resto parziale”?

Esempi:

10 (resto parziale) non contiene 1000 (divisore)

1001 (resto parziale) contiene 1000 (divisore)

Per tentativi.

Eseguo la sottrazione.

Risultato >= 0 ➔ il resto parziale è maggiore del divisore.

Risultato < 0 → il resto parziale è minore del divisore (non contiene il divisore)

Osserviamo che nel secondo caso abbiamo fatto in realtà una sottrazione che non avremmo dovuto effettuare.

NB il calcolatore non può sapere se il resto parziale contiene il divisore fino a quando non ha effettuato la sottrazione.

La divisione tra numeri binari

Divisione decimale fra i numeri su 7 bit: a = 100 1010 e b = 000 1000   a : b = ?   74 : 8=?

Nel primo passaggio allineo il divisore alla sinistra della prima cifra (MSB) del dividendo.

Allocazione di 14 bit. Sottrazione -> somma in complemento a 2:

000 0000 100 1010 -	000 0000 100 1010 +	Resto parziale = dividendo - divisore * 2^7
000 1000 000 0000 =	111 1000 000 0000 =	$= 74–8*2^7 = 74-1024 =-950$
-----------------------	------------------------	
111 1000 100 1010	111 1000 100 1010	$(= -950\|_{10})$ Nuovo resto parziale provvisorio

Ripristino il resto parziale precedente: 0000 000 100 1010

Note e strategia di implementazione

All'inizio il divisore è allineato alla sinistra del dividendo: le cifre del dividendo sono allineate agli 0 del divisore e gli 0 del dividendo sono allineati alle cifre del divisore.

Il divisore viene spostato verso dx di 1 bit ad ogni passo.

Il quoziente cresce dal bit più significativo verso il bit meno significativo. Cresce verso dx.

Ci sono N+1 passi di divisione, il primo darà sempre 0 e si potrebbe omettere.

Occorre quindi effettuare a ogni passo:

- Resto parziale = Resto parziale – divisore * base corrente
- Ripristino del resto parziale (se sottrazione < 0)
- Shift quoziente a sx.
- Scrittura di 1 o 0 nel registro quoziente.
- Shift del divisore verso dx.

Utilizzo un unico registro per dividendo e resto (il resto si ottiene per erosione del dividendo).

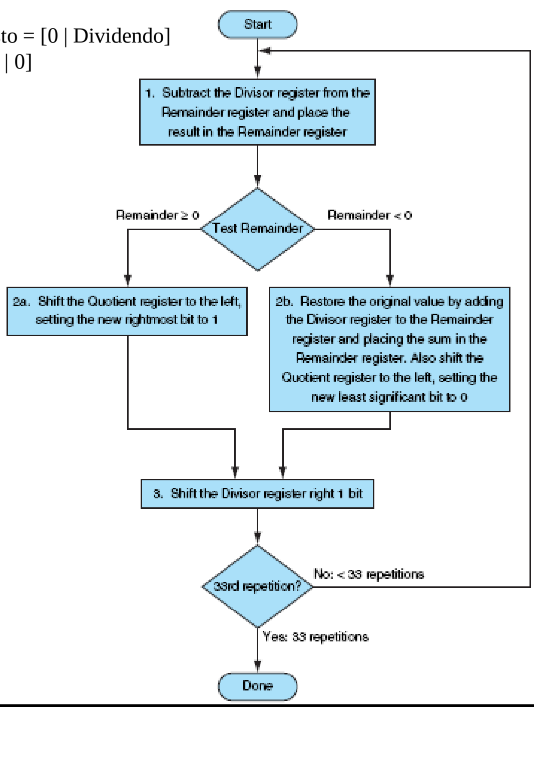
Considero il primo resto parziale uguale al dividendo (inizializzazione).

Il primo passo sarà "a vuoto" perchè produrrà sempre quoziente 0.

Divisione:: algoritmo per 32 bit

Inizializzazione: Resto = [0 | Dividendo]
Divisore = [Divisore | 0]
Quoziente = 0
k = 0

```
1001010 : 1000 = 1001
1000 -
-----
   1010 -
 0000    =
---------
   1010 -
   0000  =
---------
  1010 –
  1000 =
---------
     10
```



Esempio - 1

Divisione decimale fra i numeri a = 7 e b = 2 su 4 bit. a : b = ?

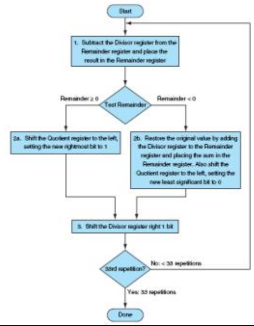
Inizializzo il divisore alla sinistra delle quattro cifre significative. La prima cifra del quoziente sarà sempre 0.

Iteration	Step	Quotient	Divisor	Remainder
0	Initial values	0000	0010 0000	0000 0111
1	1: Rem = Rem – Div	0000	0010 0000	1110 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0010 0000	0000 0111
	3: Shift Div right	0000	0001 0000	0000 0111

Divisore = $1x2^4x2^1 = 32$

Resto = 7

7 – 32 = -25 < 0



Esempio - 2

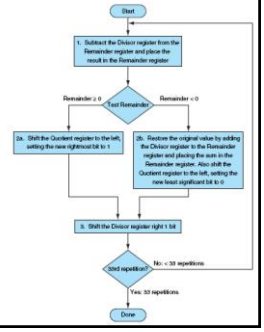
Divisione decimale fra i numeri a = 7 e b = 2 a : b = ?

Iteration	Step	Quotient	Divisor	Remainder
0	Initial values	0000	0010 0000	0000 0111
1	1: Rem = Rem – Div	0000	0010 0000	1110 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0010 0000	0000 0111
	3: Shift Div right	0000	0001 0000	0000 0111
2	1: Rem = Rem – Div	0000	0001 0000	1111 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0001 0000	0000 0111
	3: Shift Div right	0000	0000 1000	0000 0111

Divisore = $1x2^3x2 = 16$

Resto = 7

7 – 16 = -9 < 0



Esempio - 3

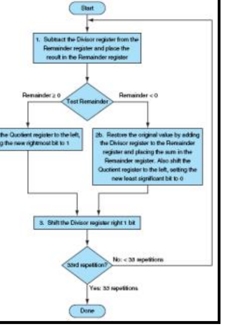
Divisione decimale fra i numeri a = 7 e b = 2   a : b = ?

Iteration	Step	Quotient	Divisor	Remainder
0	Initial values	0000	0010 0000	0000 0111
1	1: Rem = Rem – Div	0000	0010 0000	1110 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0010 0000	0000 0111
	3: Shift Div right	0000	0001 0000	0000 0111
2	1: Rem = Rem – Div	0000	0001 0000	1111 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0001 0000	0000 0111
	3: Shift Div right	0000	0000 1000	0000 0111
3	1: Rem = Rem – Div	0000	0000 1000	1111 1111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0000 1000	0000 0111
	3: Shift Div right	0000	0000 0100	0000 0111

Divisore = $1x2^2x2 = 8$

Resto = 7

7 – 8 = -1 < 0



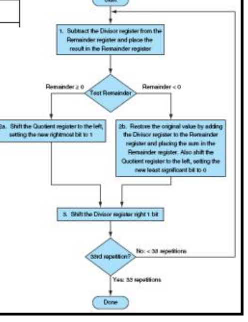
Esempio - 4

Iteration	Step	Quotient	Divisor	Remainder
0	Initial values	0000	0010 0000	0000 0111
1	1: Rem = Rem – Div	0000	0010 0000	1110 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0010 0000	0000 0111
	3: Shift Div right	0000	0001 0000	0000 0111
2	1: Rem = Rem – Div	0000	0001 0000	1111 0111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0001 0000	0000 0111
	3: Shift Div right	0000	0000 1000	0000 0111
3	1: Rem = Rem – Div	0000	0000 1000	1111 1111
	2b: Rem < 0 ⟹ +Div, sll Q, Q0 = 0	0000	0000 1000	0000 0111
	3: Shift Div right	0000	0000 0100	0000 0111
4	1: Rem = Rem – Div	0000	0000 0100	0000 0011
	2a: Rem ≥ 0 ⟹ sll Q, Q0 = 1	0001	0000 0100	0000 0011
	3: Shift Div right	0001	0000 0010	0000 0011

Divisore = $1x2^1x2 = 4$

Resto = 7

7 – 4 = +3 > 0 nuovo resto



Esempio - 5

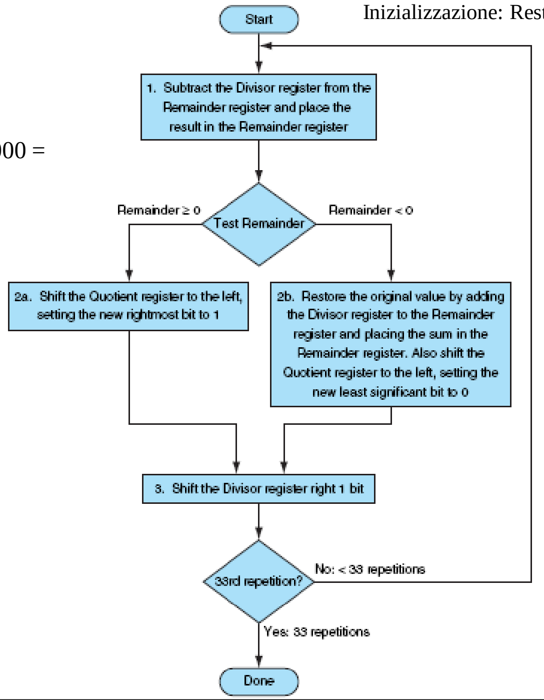
Iteration	Step	Quotient	Divisor	Remainder
0	Initial values	0000	0010 0000	0000 0111
1	1: Rem = Rem – Div	0000	0010 0000	1110 0111
	2b: Rem < 0 $\Longrightarrow$ +Div, sll Q, Q0 = 0	0000	0010 0000	0000 0111
	3: Shift Div right	0000	0001 0000	0000 0111
2	1: Rem = Rem – Div	0000	0001 0000	1111 0111
	2b: Rem < 0 $\Longrightarrow$ +Div, sll Q, Q0 = 0	0000	0001 0000	0000 0111
	3: Shift Div right	0000	0000 1000	0000 0111
3	1: Rem = Rem – Div	0000	0000 1000	1111 1111
	2b: Rem < 0 $\Longrightarrow$ +Div, sll Q, Q0 = 0	0000	0000 1000	0000 0111
	3: Shift Div right	0000	0000 0100	0000 0111
4	1: Rem = Rem – Div	0000	0000 0100	0000 0011
	2a: Rem ≥ 0 $\Longrightarrow$ sll Q, Q0 = 1	0001	0000 0100	0000 0011
	3: Shift Div right	0001	0000 0010	0000 0011
5	1: Rem = Rem – Div	0001	0000 0010	0000 0001
	2a: Rem ≥ 0 $\Longrightarrow$ sll Q, Q0 = 1	0011	0000 0010	0000 0001
	3: Shift Div right	0011	0000 0001	0000 0001

Divisore = $1x2^0x2 = 2$

Resto = 3

3 – 2 = +1 > 0 nuovo resto = resto finale

Inizializzazione: Resto = Dividendo



1001010 : 1000 =

74 : 8 =

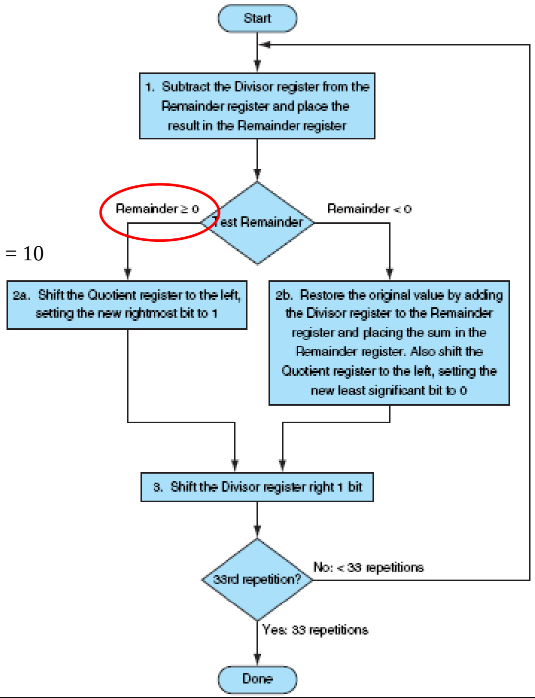
Divisione:: algoritmo

Divisione:: passo 1

01001010 : 1000 = 1

01000- - -   ($x2^3$)

00001010 > 0   (74-8*8) = 10



Divisione:: passo 2

01001010 : 1000 = 10

01000   ($x2^3$)

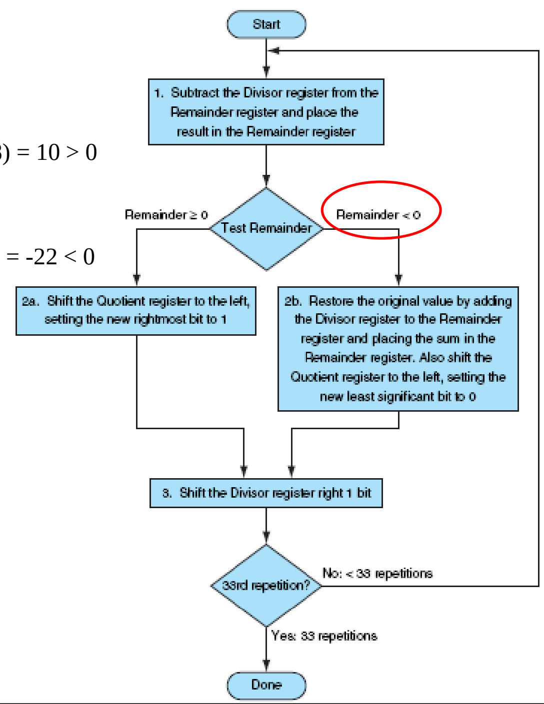
00001010 -   (74-8*8) = 10 > 0

- -1000 --   ($x2^2$)

11101011+   (10-8*4) = -22 < 0

- -1000 --

00001010



Divisione:: passo 3

01001010 : 1000 = 10

01000 $(x2^3)$

00001010 - $(74-8*8) = 10 > 0$

- -1000 -- $(x2^2)$

11101011+ $(10-8*4) = -22 < 0$

- -1000 -- $(x2^2)$

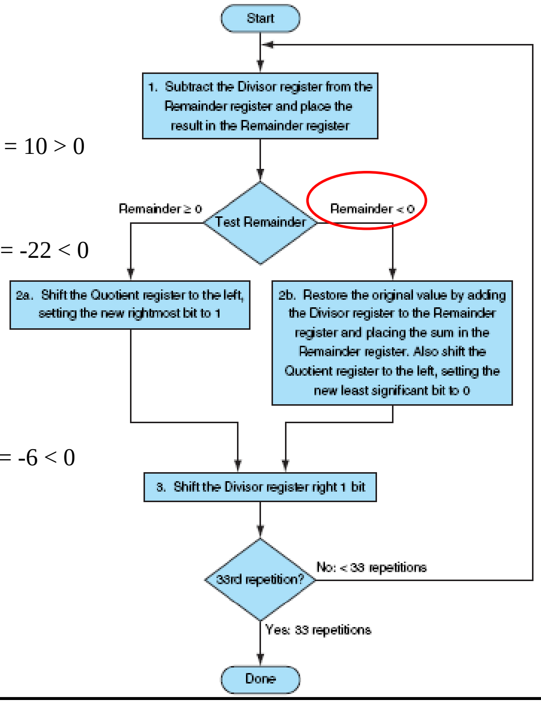
00001010 - 10

- - -1000 - $(x2^1)$

11110110 $(10-8*2) = -6 < 0$

- - -1000 + $(x2^1)$

00001010 10



Divisione:: passo 4

01001010 : 1000 = 10

01000 $(x2^3)$

00001010 - $(74-8*8) = 10 > 0$

- -1000 -- $(x2^2)$

11101011+ $(10-8*4) = -22 < 0$

- -1000 -- $(x2^2)$

00001010 - 10

- - -1000 - $(x2^1)$

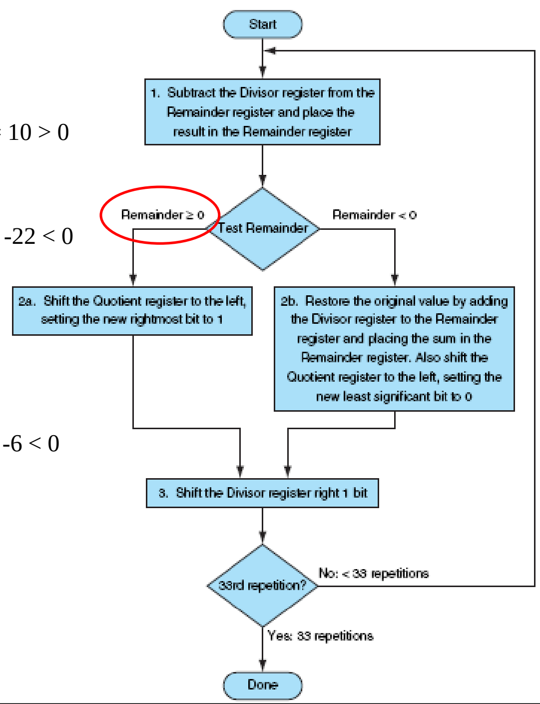
11110110 $(10-8*2) = -6 < 0$

- - -1000- + $(x2^1)$

00001010 - 10

- - - -1000 $(x2^0)$

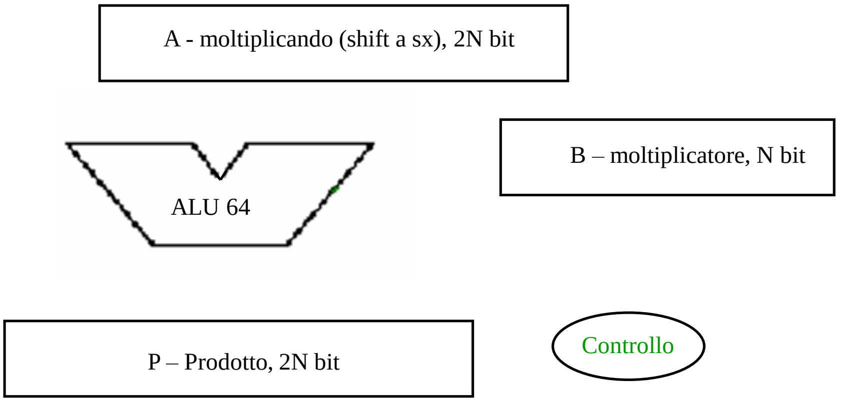
00000010



Sommario

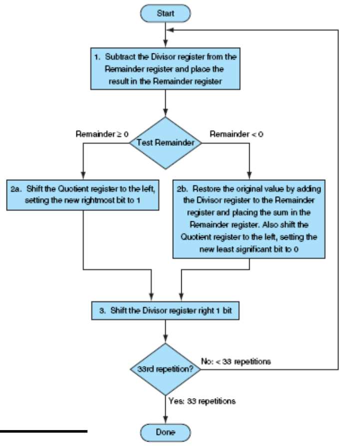
- Divisione intera
- Circuiti divisione intera
- Divisione e moltiplicazione

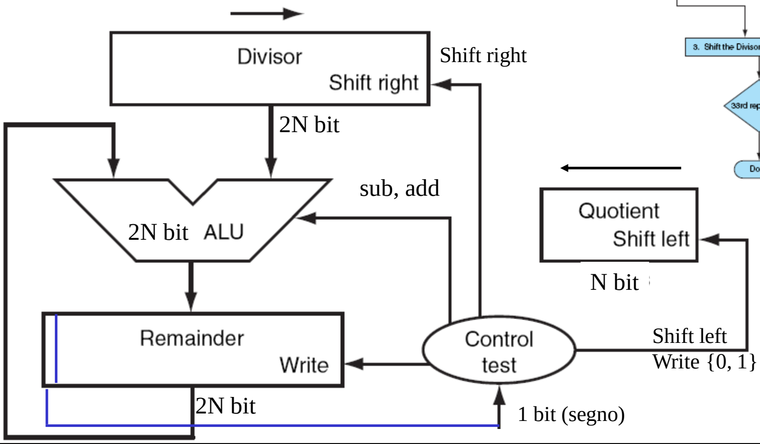
Implementazione circuitale gli stessi attori della moltiplicazione



Il circuito firmware della divisione

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

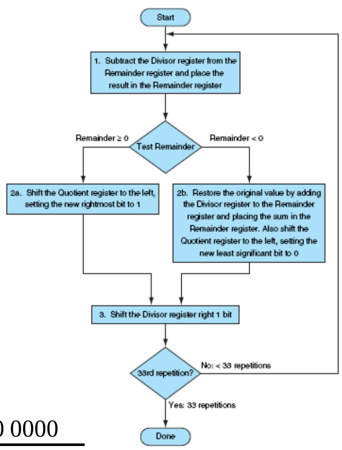


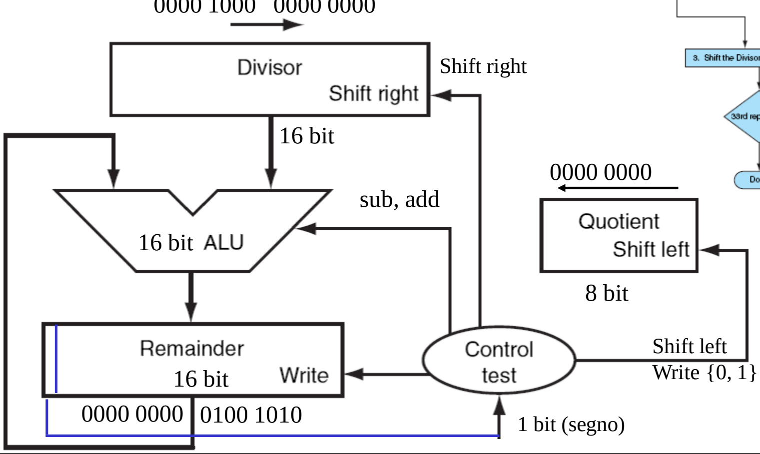


Il circuito firmware della divisione

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =
74 : 8 su 8 bit



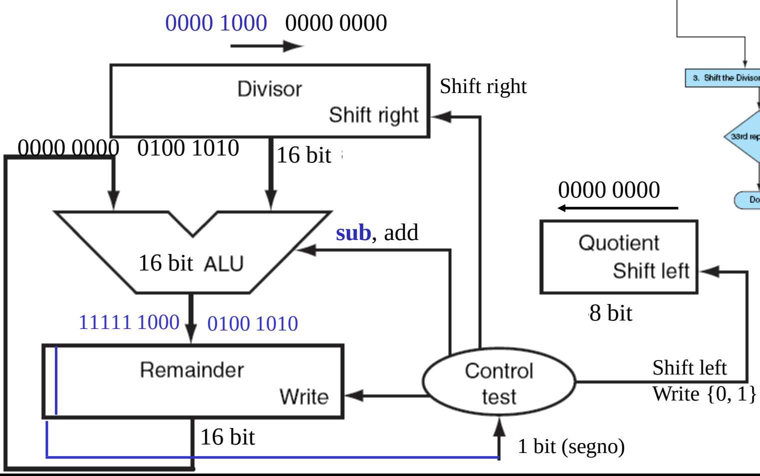


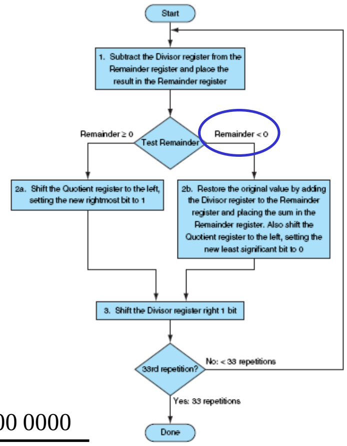
Il circuito firmware della divisione – passo 1a

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

74 - 8 x 2^8 = -1974 < 0



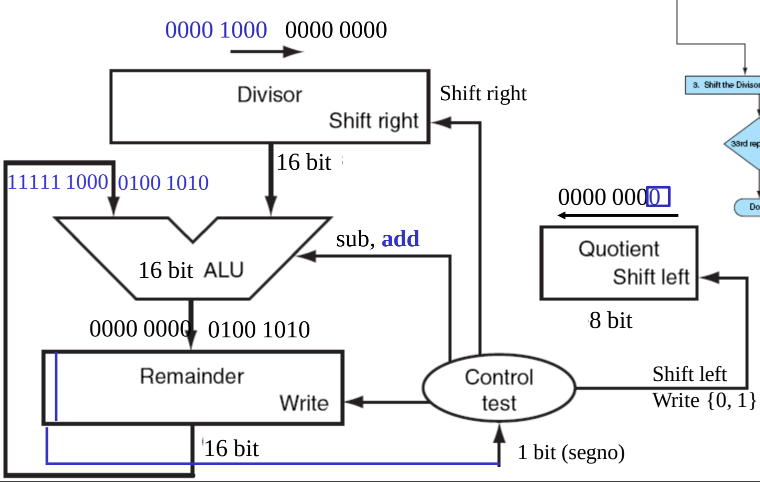


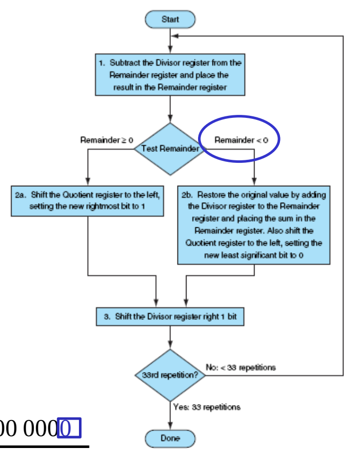
Il circuito firmware della divisione – passo 1b

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

-1974 + 8 x 2^8 = +74





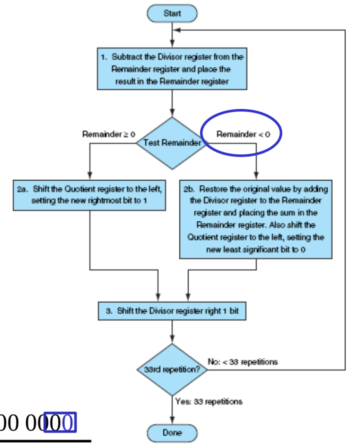
Il circuito firmware della divisione – passo 2

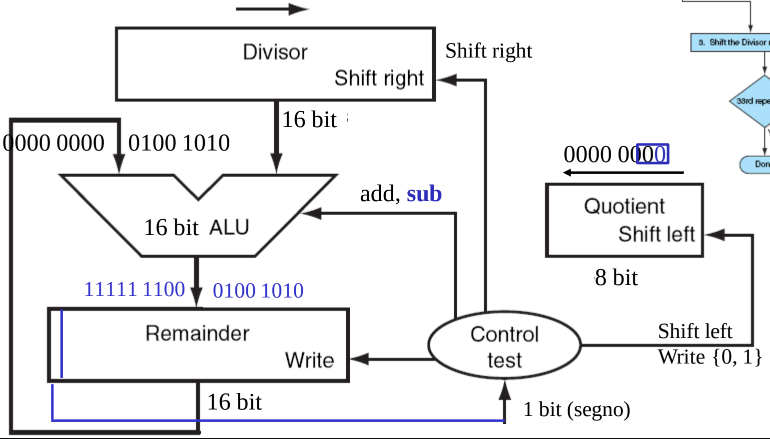
Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

$7 - 8 \times 2^7 = -950 < 0$

0000 0100 0000 0000





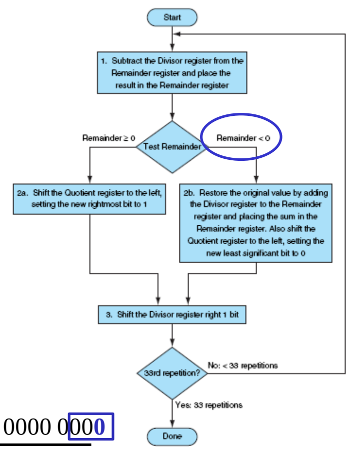
Il circuito firmware della divisione – passo 3

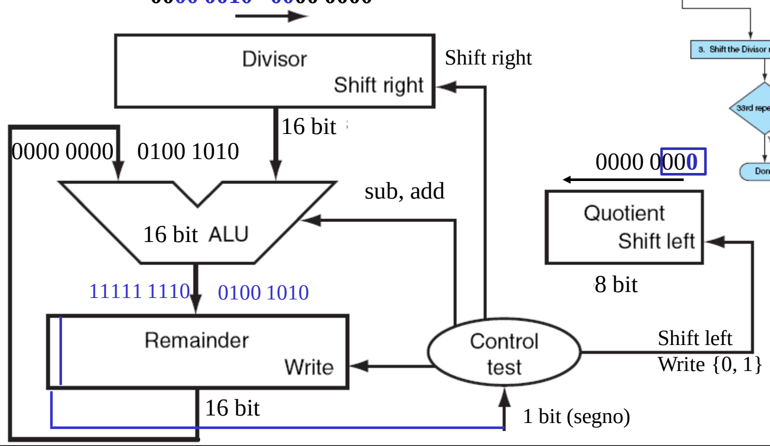
Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

$74 - 8 \times 2^6 = -438 < 0$

0000 0010 0000 0000



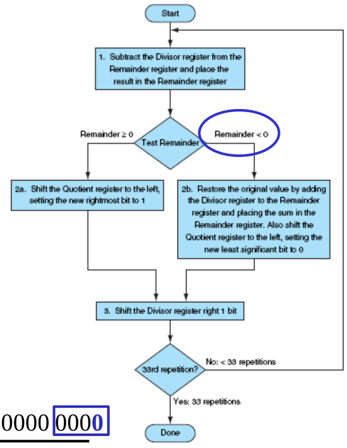


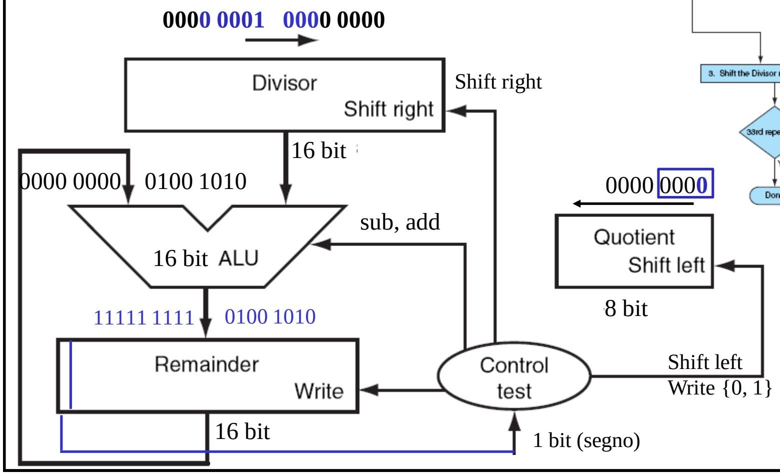
Il circuito firmware della divisione – passo 4

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

74 - 8 x 2^5 = -182 < 0



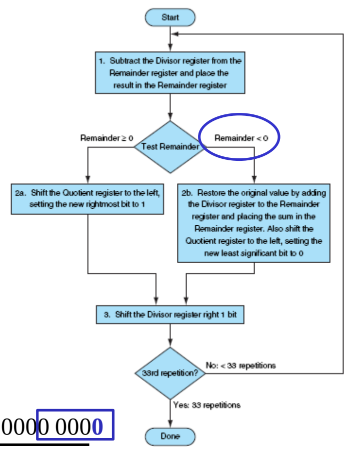


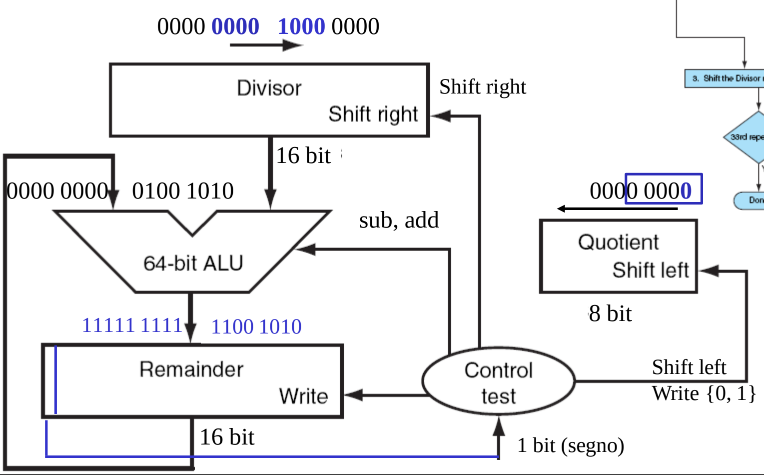
Il circuito firmware della divisione – passo 5

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

74 - 8 x 2^4 = -54 < 0



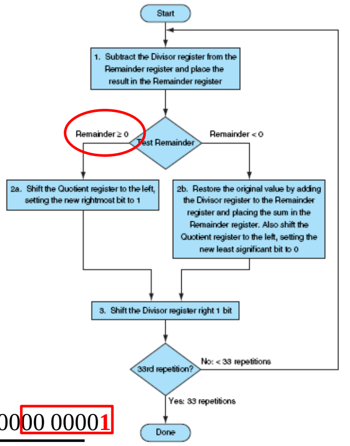


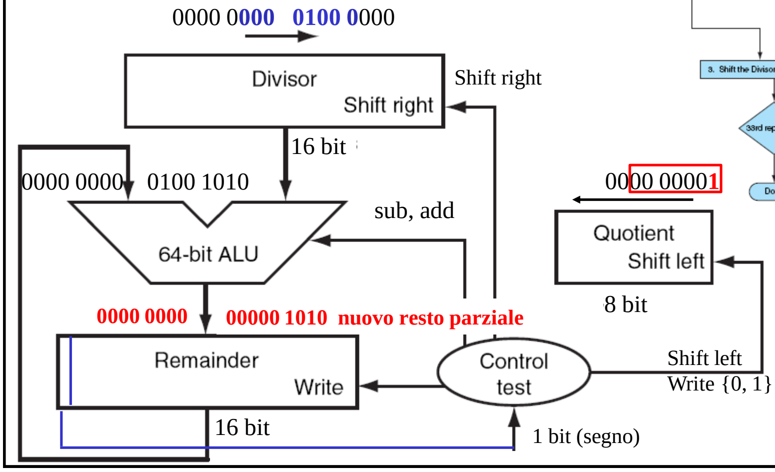
Il circuito firmware della divisione – passo 6

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

$74 - 8 \times 2^3 = +10 \geq 0$





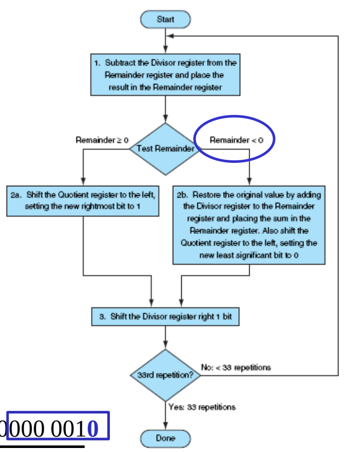
http://borghese.di.unimi.it\

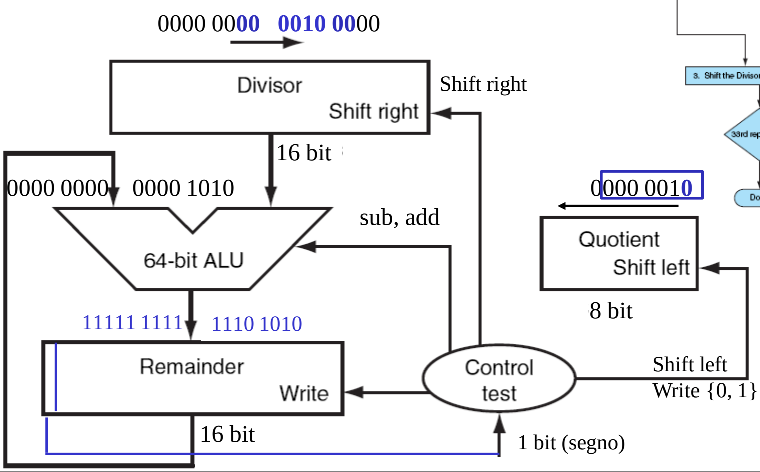
Il circuito firmware della divisione – passo 7

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

$10 - 8 \times 2^2 = -22 < 0$





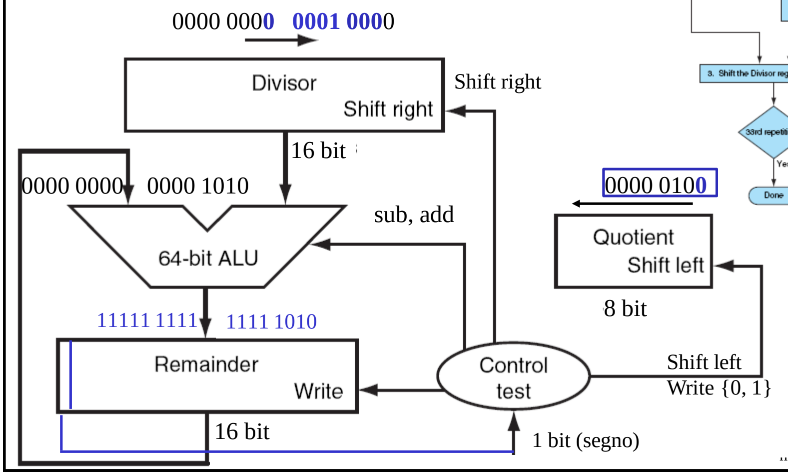
http://borghese.di.unimi.it\

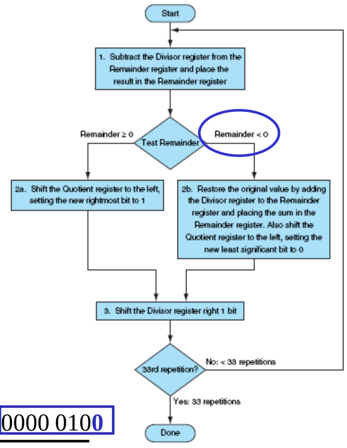
Il circuito firmware della divisione – passo 8

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

$10 - 8 \times 2^1 = -6 < 0$



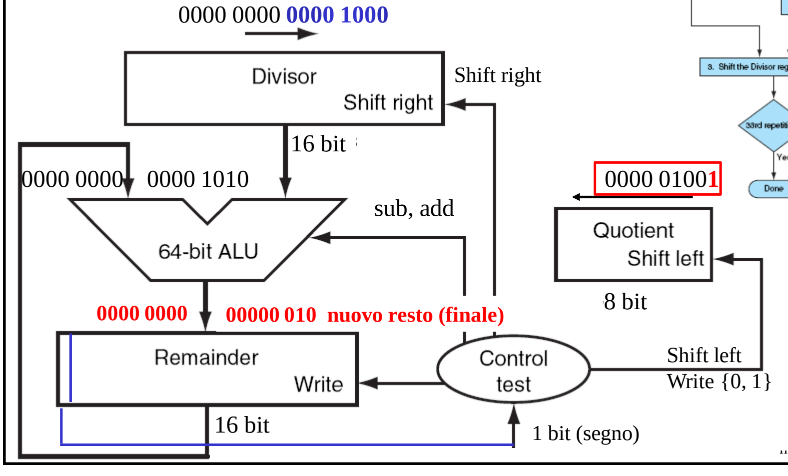


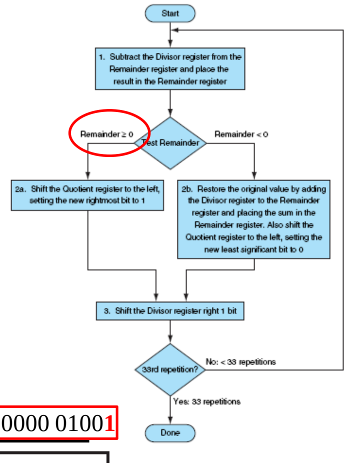
Il circuito firmware della divisione – passo 9

Inizializzazione:
-- Resto = 0 | Dividendo
-- Divisore | 0

1001010 : 1000 =

$10 - 8 \times 2^0 = +2 \geq 0$





Come ottimizzare il circuito della divisione

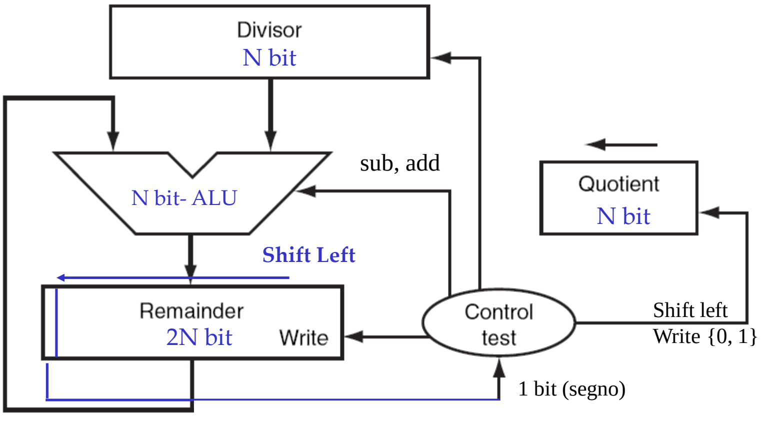
Il divisore si sposta verso dx di un bit ad ogni passo e viene sottratto al resto parziale.
Otteniamo lo stesso risultato se spostiamo il resto parziale a sx di un bit ad ogni passo.

Inizializziamo il resto come RESTO = 0 | DIVIDENDO.

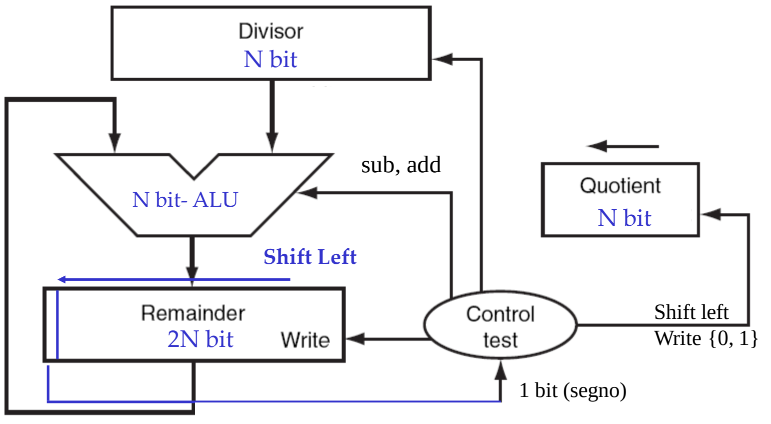
Ad ogni passo sposto il dividendo alias resto parziale di una posizione a sx ed inserisco un bit del quoziente.

Il circuito firmware con un’ottimizzazione

Inizializzazione: Resto = 0 | Dividendo



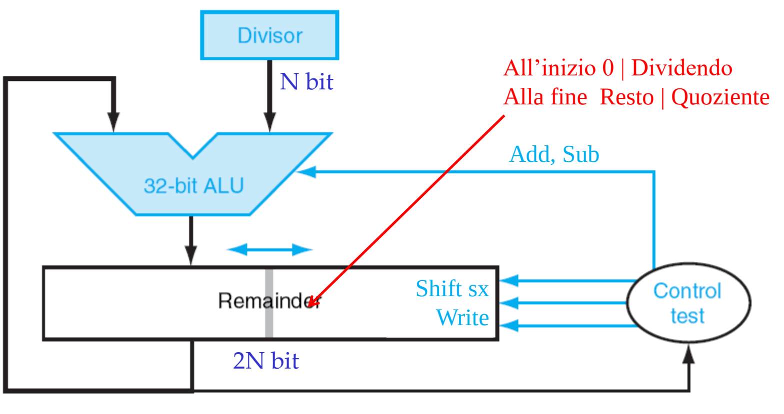
Razionale per un'ulteriore ottimizzazione



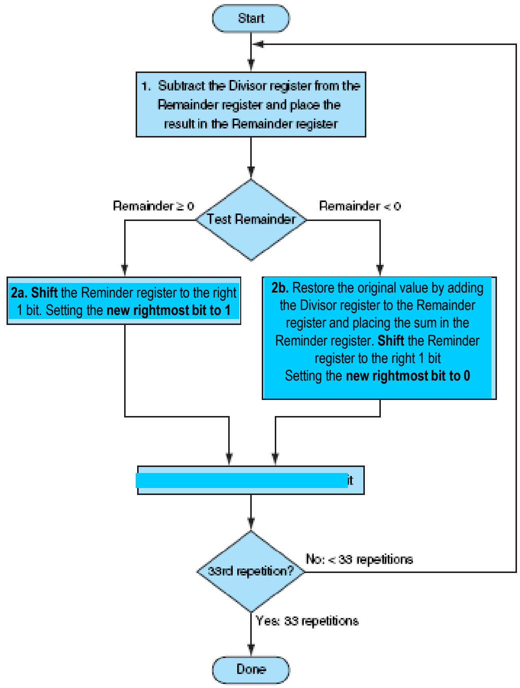
Il quoziente viene riempito un bit alla volta da dx a sx
Il dividendo viene spostato da dx a sx di una posizione

Il circuito firmware ottimizzato della divisione

Inizializzazione: Resto = 0 | Dividendo



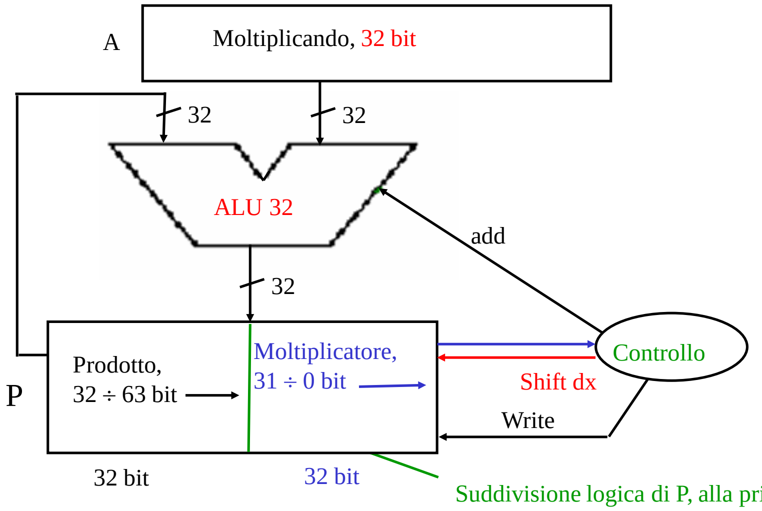
Divisione:: algoritmo



Sommario

- Divisione intera
- Circuiti divisione intera
- Divisione e moltiplicazione

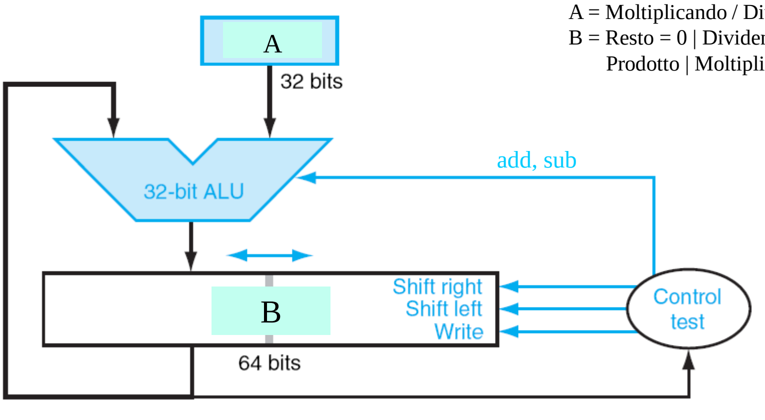
Circuito ottimizzato della moltiplicazione (su 32 bit)



Suddivisione logica di P, alla prima iterazione
P = 0 | Moltiplicatore

Un unico circuito per moltiplicazione e divisione

Inizializzazione:
A = Moltiplicando / Divisore
B = Resto = 0 | Dividendo / Prodotto | Moltiplicatore



Sommario

- Divisione intera
- Circuiti divisione intera
- Divisione e moltiplicazione