

Arhitectura Calculatoarelor

Oprîtoiu Flavius
flavius.opritoiu@cs.upt.ro

4 Decembrie 2024
11 Decembrie 2024

Cap. 4 Analiza Funcțională și Sinteza Dispozitivelor de Înmulțire Binară

4.1 - Metode de înmulțire

Un înmulțitor calculează produsul $P = X \cdot Y$, unde

- ▶ operandul X se numește înmulțitor,
- ▶ operandul Y se numește deînmulțit

Se consideră operanzii fără semn X și Y , pe 4 biți:

$$X = 11_{(10)} = 1011_{(2)}$$

$$Y = 12_{(10)} = 1100_{(2)}$$

4.1 - Metode de înmulțire (contin.)

Ⓐ: Paper and pencil

$$\begin{array}{r}
 \begin{array}{cccccccc}
 & & & & 1 & 1 & 0 & 0 & \text{-----} & Y \\
 & & & & 1 & 0 & 1 & 1 & = x_3 x_2 x_1 x_0 & \text{---} & X \\
 \hline
 & & & & 1 & 1 & 0 & 0 & \text{-----} & x_0 Y 2^0 \\
 & & & 1 & 1 & 0 & 0 & \text{-----} & x_1 Y 2^1 \\
 & & 0 & 0 & 0 & 0 & \text{-----} & x_2 Y 2^2 \\
 & 1 & 1 & 0 & 0 & \text{-----} & x_3 Y 2^3 \\
 \hline
 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \text{---} & P = \sum_{i=0}^3 x_i Y 2^i = 132
 \end{array}
 \end{array}$$

Investiție hardware:

- ▶ 2 registre pe 4 biți pentru X și Y
- ▶ sumator multi-operand
- ▶ "gating"-ul de înmulțitului

4.1 - Metode de înmulțire (contin.)

(B): Păstrarea fixă a produselor parțiale

	1	1	0	0	-----	Y					
	1	0	1	1	$= x_3 x_2 x_1 x_0$	---	X				
0	0	0	0	0	0	0	0	0	-----	$P_0 := 0$	
	1	1	0	0	-----	$x_0 Y 2^0$					+
0	0	0	0	1	1	0	0	0	-----	$P_1 := P_0 + x_0 Y 2^0$	
	1	1	0	0	-----	$x_1 Y 2^1$					+
0	0	1	0	0	1	0	0	0	-----	$P_2 := P_1 + x_1 Y 2^1$	
	0	0	0	0	-----	$x_2 Y 2^2$					+
0	0	1	0	0	1	0	0	0	-----	$P_3 := P_2 + x_2 Y 2^2$	
	1	1	0	0	-----	$x_3 Y 2^3$					+
1	0	0	0	0	1	0	0	0	-----	$P_4 := P_3 + x_3 Y 2^3 = P$	

4.1 - Metode de înmulțire (contin.)

ⓑ: Păstrarea fixă a produselor parțiale (continuare)

Înmulțirea este realizată printr-o secvență de pași, cu pasul de iterație:

$$P_{i+1} = P_i + x_i \cdot Y \cdot 2^i, \text{ pentru } i \geq 0, P_0 := 0$$

P_i este un produs parțial și expresia $x_i \cdot Y \cdot 2^i$ este un produs de 1 bit.

Investiție hardware:

- ▶ 2 registre de 4 biți pentru X și Y
- ▶ registru de 8 biți pentru produsele parțiale
- ▶ sumator pe 8 biți
- ▶ mecanism de aliniere a produselor de 1 bit

4.1 - Metode de înmulțire (contin.)

Ⓒ: Păstrearea fixă a produselor de 1 bit

	1	1	0	0	-----	Y					
	1	0	1	1	$= x_3x_2x_1x_0$	---	X				
0000	0	0	0	0	-----	$P_0 := 0$	+				
	1	1	0	0	-----	x_0Y					
0000	1	1	0	0	-----	$P_0 := P_0 + x_0Y$	+				
000	0	1	1	0	0	-----		$P_1 := P_0 \cdot 2^{-1}$			
	1	1	0	0	-----	x_1Y	+				
001	0	0	1	0	0	-----		$P_1 := P_1 + x_1Y$			
00	1	0	0	1	0	0	-----	+			
	0	0	0	0	-----	x_2Y					
00	1	0	0	1	0	0	-----	+			
0	0	1	0	0	1	0	0		-----	$P_3 := P_2 \cdot 2^{-1}$	
	1	1	0	0	-----	x_3Y	+				
1	0	0	0	0	1	0		0	-----	$P_3 := P_3 + x_3Y$	
	1	0	0	0	0	1	0	0	--	$P_4 := P_3 \cdot 2^{-1}$	$= P$

4.1 - Metode de înmulțire (contin.)

Ⓒ: Păstrarea fixă a produselor de 1 bit (continuare)

Înmulțirea este realizată printr-o secvență de pași, cu pasul de iterație:

$$\begin{cases} P_i = P_i + x_i \cdot Y \\ P_{i+1} = P_i \cdot 2^{-1} \end{cases}, \text{ pentru } i \geq 0, P_0 := 0$$

Investiție hardware:

- ▶ 2 registre de 4 biți pentru X și Y
- ▶ registru de 8 biți pentru produsele parțiale cu facilitare de deplasare la dreapta
- ▶ sumator pe 4 biți

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire

Fie X și Y 2 numere fracționare, pe 8 biți, reprezentate în S.-M.

$$\begin{array}{rcccccccc}
 X & = & x_7 & . & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \\
 Y & = & \underbrace{y_7}_{\text{semn}} & . & \underbrace{y_6}_{2^{-1}} & \underbrace{y_5}_{2^{-2}} & \underbrace{y_4}_{2^{-3}} & \underbrace{y_3}_{2^{-4}} & \underbrace{y_2}_{2^{-5}} & \underbrace{y_1}_{2^{-6}} & \underbrace{y_0}_{2^{-7}}
 \end{array}$$

\longleftarrow ponderi \longrightarrow

Produsul $P = X \cdot Y$ este un număr fracționar în SM, pe 16 biți

$$P = p_{15} . p_{14} p_{13} \cdots p_2 p_1 p_0$$

- ▶ Most Significant Bit (MSB) este semnul: $p_{15} = x_7 \oplus y_7$
- ▶ părțile de magnitudine ale operandilor au 7 biți \Rightarrow magnitudinea produsului va avea 14 biți: $p_{14}, p_{13}, \dots, p_1$
- ▶ al 16-lea bit este p_0 și are valoarea 0

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

multiplier 2

declare register $A[7 : 0]$, $Q[7 : 0]$, $M[7 : 0]$, $COUNT[2 : 0]$;

declare bus $INBUS[7 : 0]$, $OUTBUS[7 : 0]$;

BEGIN : $A := 0$, $COUNT := 0$, } \leftarrow -----{ c_0 }

INPUT : $M := INBUS$;

$Q := INBUS$; \leftarrow -----{ c_1 }

TEST1 : if $Q[0] = 0$ then go to *RSHIFT*,

ADD : $A[7 : 0] := A[6 : 0] + M[6 : 0]$; \leftarrow -----{ c_2 }

RSHIFT : $A[7] := 0$, $A[6 : 0].Q := A.Q[7 : 1]$, } \leftarrow -----{ c_3 }

INCR. : $COUNT := COUNT + 1$;

TEST2 : if $COUNT \neq 1$ then go to *TEST1*,

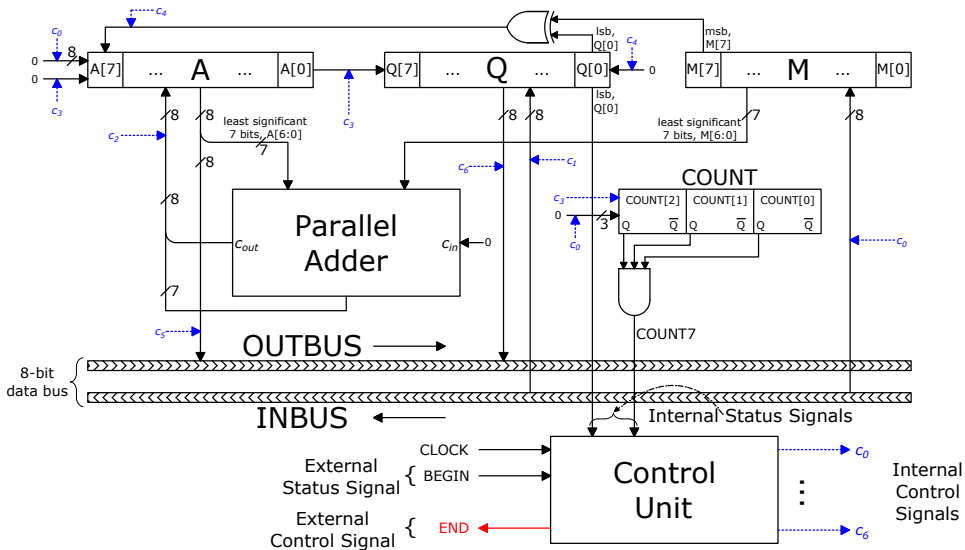
SIGN : $A[7] := Q[0]$ exor $M[7]$, $Q[0] := 0$; \leftarrow -----{ c_4 }

OUTPUT : $OUTBUS := A$; \leftarrow -----{ c_5 }

$OUTBUS := Q$; \leftarrow -----{ c_6 }

END : ----- \rightarrow { **END** }

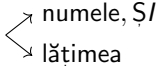
4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)



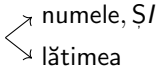
4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărime (contin.)

Pseudolimbajul folosit de algoritm:


1. declare registers

- ▶ definește regiștri; specifică  numele, S/
lățimea
- ▶ operatorul de concatenare: •; ex. $A[6 : 0].Q := A.Q[7 : 1]$

2. declare bus

- ▶ definește magistrale specificând  numele, S/
lățimea

3. Execuție sincronă:

- 
- operații non-conflictuale: executate concurent, separate prin 9
 - operații secvențiale: executate secvențial, separate prin 9

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

Pseudolimbajul folosit de algoritm (continuare):

4. Operatorul de atribuire este $\boxed{:=}$ și este folosit pentru încărcarea de valori binare în regiștri sau magistrale
- ▶ exor indică o operație cablată: în $A[7] := Q[0] \text{ exor } M[7]$ este implementată printr-o poartă EXOR

5. Controlul fluxului de execuție:

- ↙ sald necondiționat: go to *TEST1*
- ↘ salt condiționat: if $Q[0] = 0$ then go to *RSHIFT*

6. Citire/scriere simultană din/în registre:

$$A[7] := Q[0] \text{ exor } M[7], Q[0] := 0$$

- ▶ bit-ul $Q[0]$ este atât citit cât și scris în același ciclu de ceas

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

Elementele platformei HW:

- ▶ acumulatorul A : este folosit pentru adunarea produselor de 1-bit la produsul parțial; are facilități de deplasare la dreapta; stochează biții mai semnificativi ai produselor parțiale
- ▶ registrul înmulțitor Q : încărcat, inițial, cu înmulțitorul X ; are facilități de deplasare la dreapta (numele Q provine de la platforma HW specifică împărțirii binare)
- ▶ registrul de înmulțit M : stochează de înmulțitul Y ;
- ▶ sumatorul paralel: pe 7 biți; utilizat pentru adunarea produselor de 1-bit la produsul parțial
- ▶ contorul $COUNT$: păstrează evidența numărului de iterații executate
- ▶ unitatea de control generează semnalele de control (c_0, \dots, c_6 , **END**) în secvența corectă de execuție a algoritmului

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărimă (contin.)

Algoritmul folosește metoda a 3-a de înmulțire (păstrarea fixă a produselor de 1-bit):

$$\begin{cases} P_i = P_i + x_i \cdot Y & \longrightarrow \text{etichetele TEST1 și ADD} \\ P_{i+1} = P_i \cdot 2^{-1} & \longrightarrow \text{eticheta RSHIFT} \end{cases}$$

Produsele parțiale:

- ▶ până la setarea semnului rezultatului, (eticheta **SIGN**), toate produsele parțiale sunt pozitive, reprezentate în S.-M., independent de semnele operanzilor
- ▶ la începutul algoritmului, P_0 ocupă registrul A ($P_0 := 0$)
- ▶ după prima iterație, P_1 ocupă întreg registrul A și MSB-ul lui Q (P_1 este în $A[7 : 0].Q[7]$)
- ▶ după a doua iterație, P_2 ocupă întreg A -ul și primii 2 MSBs ai lui Q (P_2 este în $A[7 : 0].Q[7 : 6]$)
- ▶ în general, P_{i+1} ocupă un bit suplimentar în Q la fiecare nouă iterație; aceasta se petrece la eticheta **RSHIFT**, unde A concatenat cu Q este deplasat la dreapta

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

În fiecare iterație, registrul Q este deplasat la dreapta:

- ▶ la începutul algoritmului, $Q[0]$ conține pe x_0
- ▶ după prima iterație, $Q[0]$ conține pe x_1
- ▶ după a doua iterație, $Q[0]$ conține pe x_2

⇒ în oricare iterație, bitul curent al lui X , x_i , se află în $Q[0]$.

Acesta este motivul pentru care la **TEST1**, algoritmul testează bitul $Q[0]$.

Sumator paralel:

- ▶ pe 7 biți pentru că produsul parțial P_i și deînmulțitul Y sunt în S.-M. iar în S.-M. adunarea nu poate calcula semnul rezultatului în mod corect
- ▶ transportul de ieșire al adunării este stocat în $A[7]$
 - ▶ ⇒ se evită *overflow*-ul

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărime (contin.)

Contorul:

- ▶ numără 7 iterații:
 - ▶ pentru că magnitudinea înmulțitorului X are o lățime de 7 biți
- ▶ incrementat la eticheta **RSHIFT**
- ▶ semnalul $COUNT7$ este activat când conținutul numărătorului este 7 ($7_{(10)} = 111_{(2)}$)

Unitatea de control:

- ▶ secvențiază operațiile prin activarea semnalelor de control (c_0 , ..., c_6 , **END**)
- ▶ în fiecare ciclu de tact, cel puțin un semnal de control este activat
- ▶ semnalele de control sunt reprezentate prin linie întreruptă
-----▶●
- ▶ liniile de date sunt reprezentate prin linie solidă ———

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

Se consideră operanzii pe 4 biți, în S.-M., fracționari:

$$X = -0.625 = -5 \cdot 2^{-3} = 1.101_{S.-M.}$$

$$Y = -0.875 = -7 \cdot 2^{-3} = 1.111_{S.-M.}$$

Produsul P ai celor 2 operanzi, reprezentat în S.-M., pe 8 biți:

$$\begin{aligned} P &= X \cdot Y = (-5 \cdot 2^{-3}) \cdot (-7 \cdot 2^{-3}) = 35 \cdot 2^{-6} \\ &= 0.1000110_{S.-M.} \end{aligned}$$

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

Se consideră operanzii pe 4 biți, în S.-M., întregi:

$$X = -6 = 1110_{S.-M.}$$

$$Y = +7 = 0111_{S.-M.}$$

Produsul P ai celor 2 operanzi este obținut ca:

$$\begin{aligned} P &= X \cdot Y = -6 \cdot +7 = -35 \\ &= 10101010_{S.-M.} \end{aligned}$$

4.2 - Înmulțire binară secvențială pentru numere în Semn-Mărire (contin.)

