



---

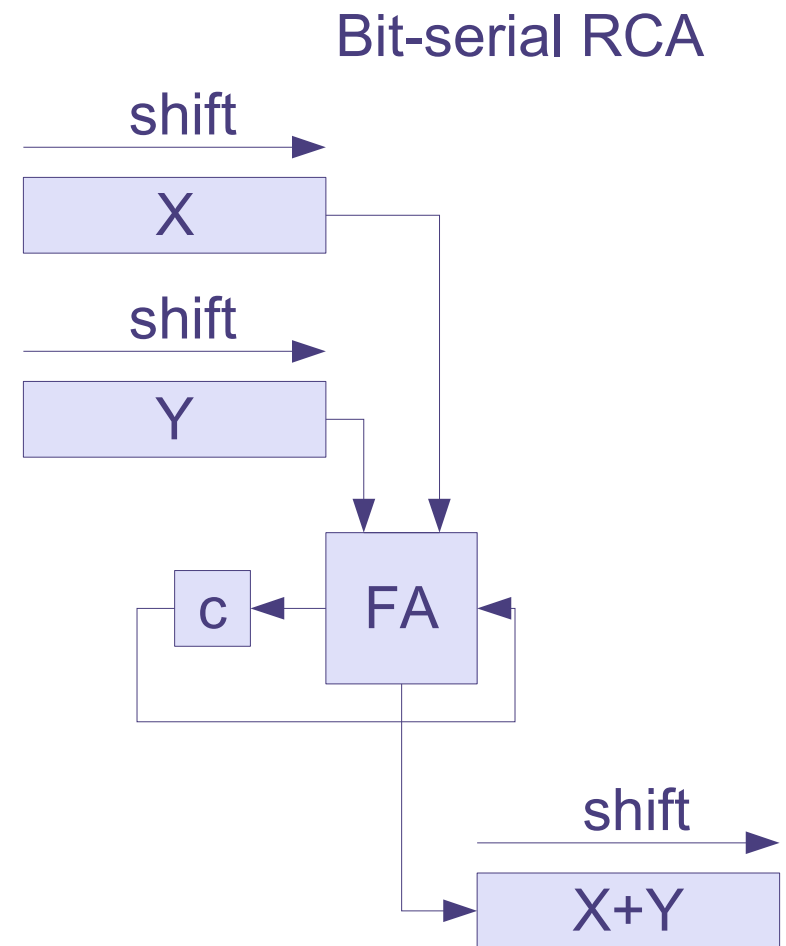
# *Szybkie układy mnożące*

# Operacja mnożenia

- Operacje dodawania i mnożenia są podstawą algorytmów obliczania wartości innych złożonych funkcji matematycznych oraz przetwarzania sygnałów
- Implementacje
  - bitowo-szeregowe
  - sekwencyjne dodawanie wieloargumentowe
    - mnożenie przy poszerzonej bazie (*high-radix*)
  - drzewiaste dodawanie wieloargumentowe
    - drzewa (regularne i nieregularne) CSA
  - metoda "dziel i zwyciężaj" (*Divide & Conquer*)

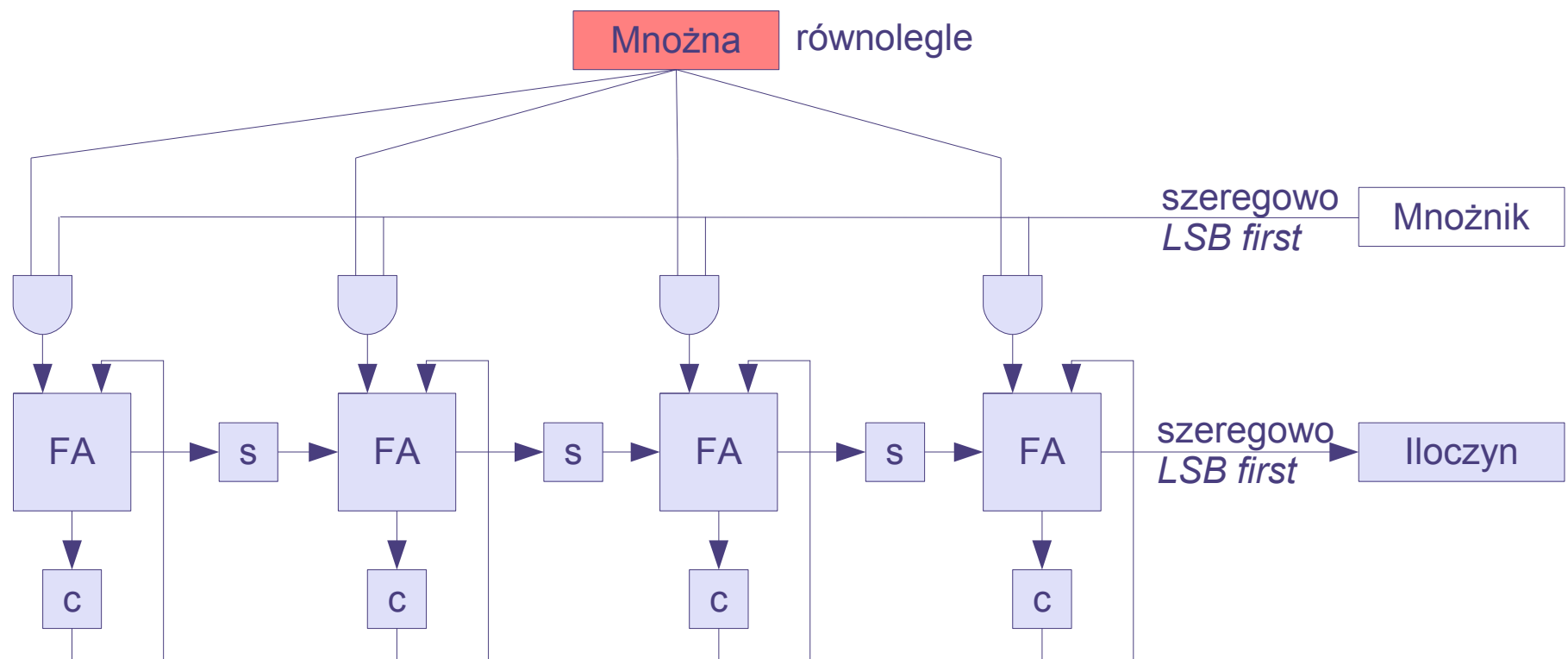
# Dodawanie bitowo-szeregowe (RCA)

- ⦿ Zalety (dla implementacji VLSI)
  - ⦿ mała liczba wyprowadzeń
  - ⦿ krótkie połączenia
  - ⦿ duża szybkość zegara
  - ⦿ mała powierzchnia
  - ⦿ niski pobór mocy
- ⦿ Dobrze nadaje się do przetwarzania potokowego
- ⦿ Szybkość bez zmian:  $\Theta(k)$



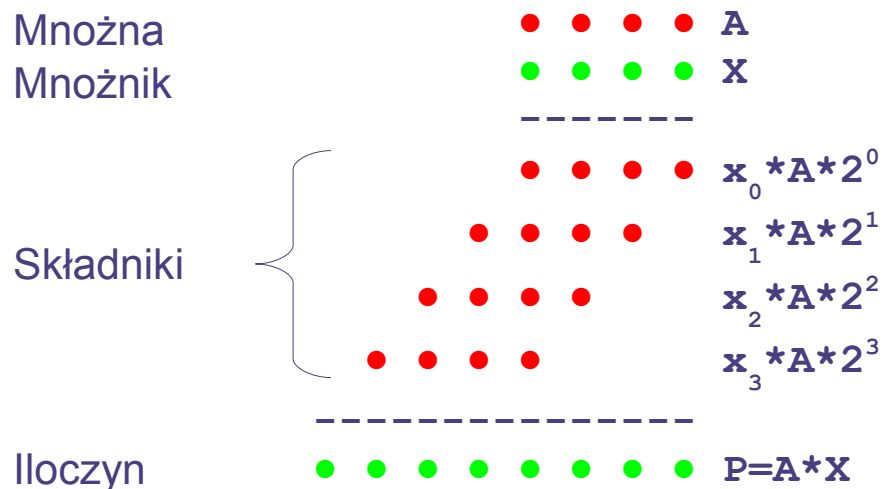
# Mnożenie bitowo-szeregowo

- 🕒 Rozwiązanie częściowo-szeregowe (*semi-serial*)
  - 🕒 mnożenie liczb 4bitowych (k-bitowych)
  - 🕒 szybkość:  $2k$  cykli ( $\Theta(k)$ )
  - 🕒 wady: dedykowany sprzęt



# Układy mnożące sekwencyjne

## Prosta implementacja sprzętowa i programowa



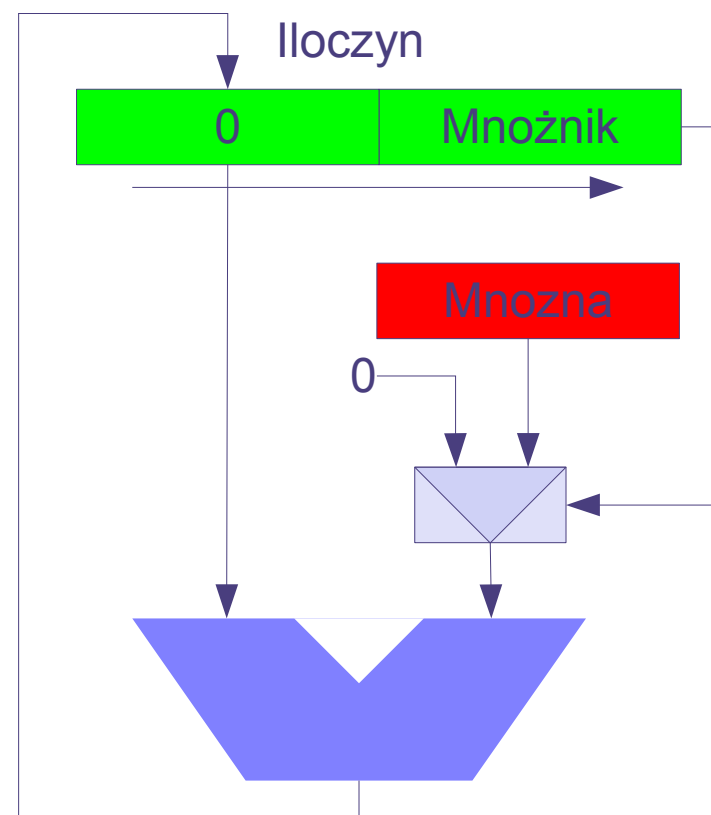
cyfra dwójkowa [0,1]

$$P^{i+1} = P^i + x_i * A * 2^i$$

sumy częściowe      przesunięcie na i-pozycję

Składniki dodawania są zawsze iloczynem:  $A * x_i$  (mnożna \* cyfra mnożnika), a przesunięcie zapewnia odpowiednią wagę składnika.

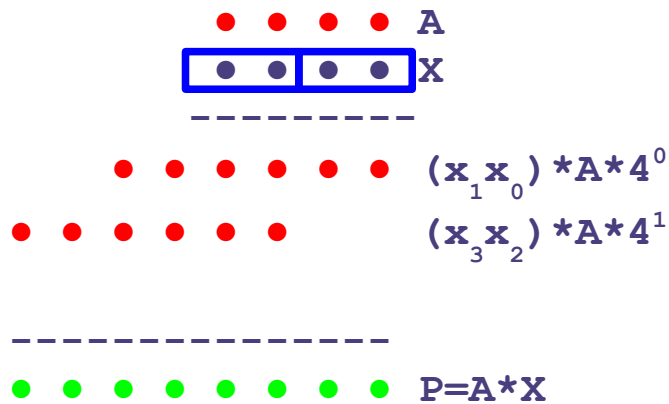
Przy bazie 2 (radix-2) składniki mogą mieć tylko dwie wartości: 0 i A.



Zamiast przesuwania mnożnej, można przesunąć rejestr iloczynu i stosować węższy sumator

# Mnożenie przy poszerzonej bazie

- Redukcja liczby składników dodawania
  - dla radix-4, liczba składników jest dwa razy mniejsza
  - cyfry radix-4 zajmują dwa bity i mogą wynosić 0,1,2,3
  - składniki dodawania mogą wynosić 0, A, 2A, 3A
  - obliczanie 3A jest kłopotliwe



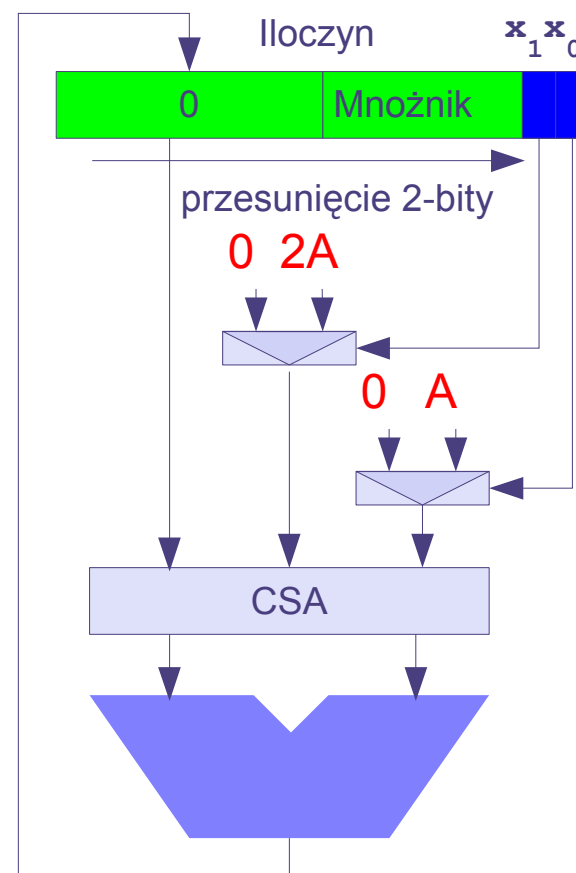
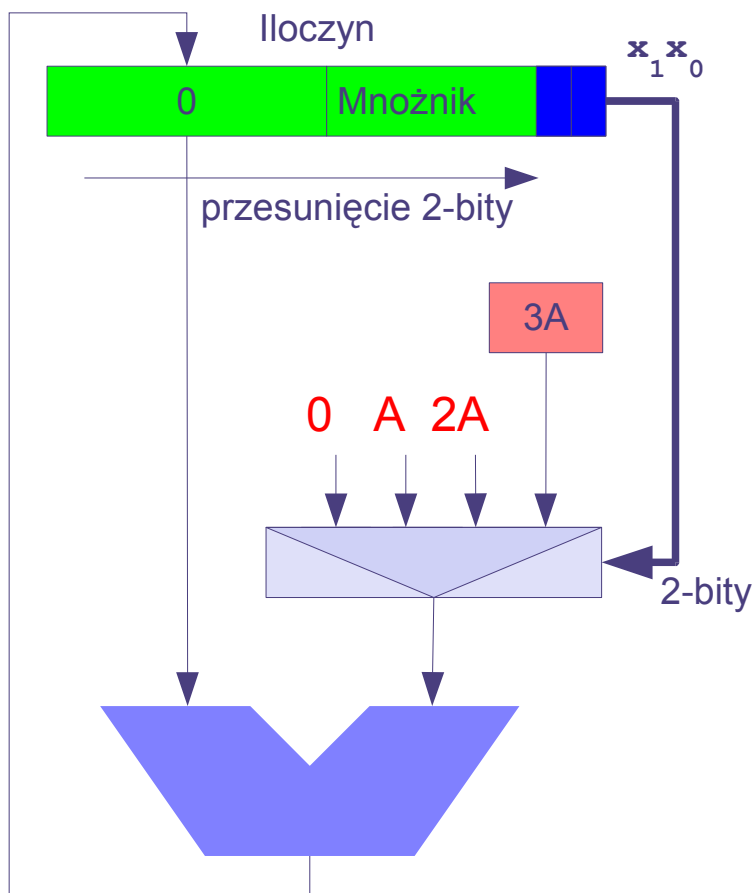
cyfra czwórkowa [0,3]

$$P^{i+1} = P^i + x_i * A * 4^i$$

przesunięcie na pozycję odpowiadającą cyfrze czwórkowej (2 bity na cyfrę)

# Mnożenie przy poszerzonej bazie

- Nietypowe wartości składników (np.  $3A$  dla radix-4) można obliczać wstępnie lub stosować sumator CSA
- Arytmetyka liczb dodatnich (unsigned)



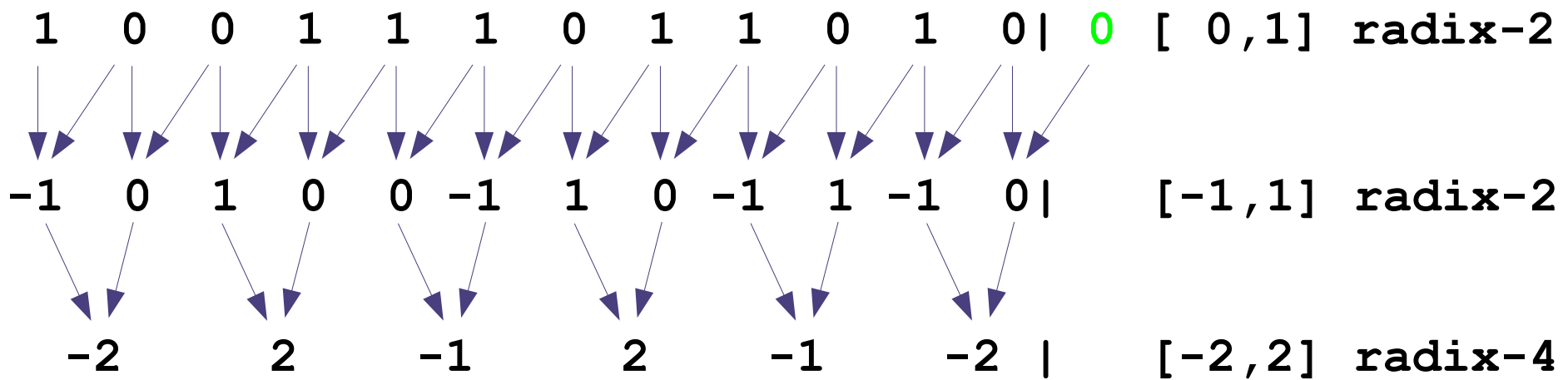
# Mnożenie liczb ze znakiem

- Algorytm Booth'a przy poszerzonej bazie
- Konwersja mnożnika  $[0,1]$  na  $[-1,1]$ ,  $[-2,2]$ ,  $[-4,4], \dots$

00,11 – kodowane jako 0

10 – początek bloku jedynek – kodowane jako -1

01 – koniec bloku jedynek – kodowane jako 1



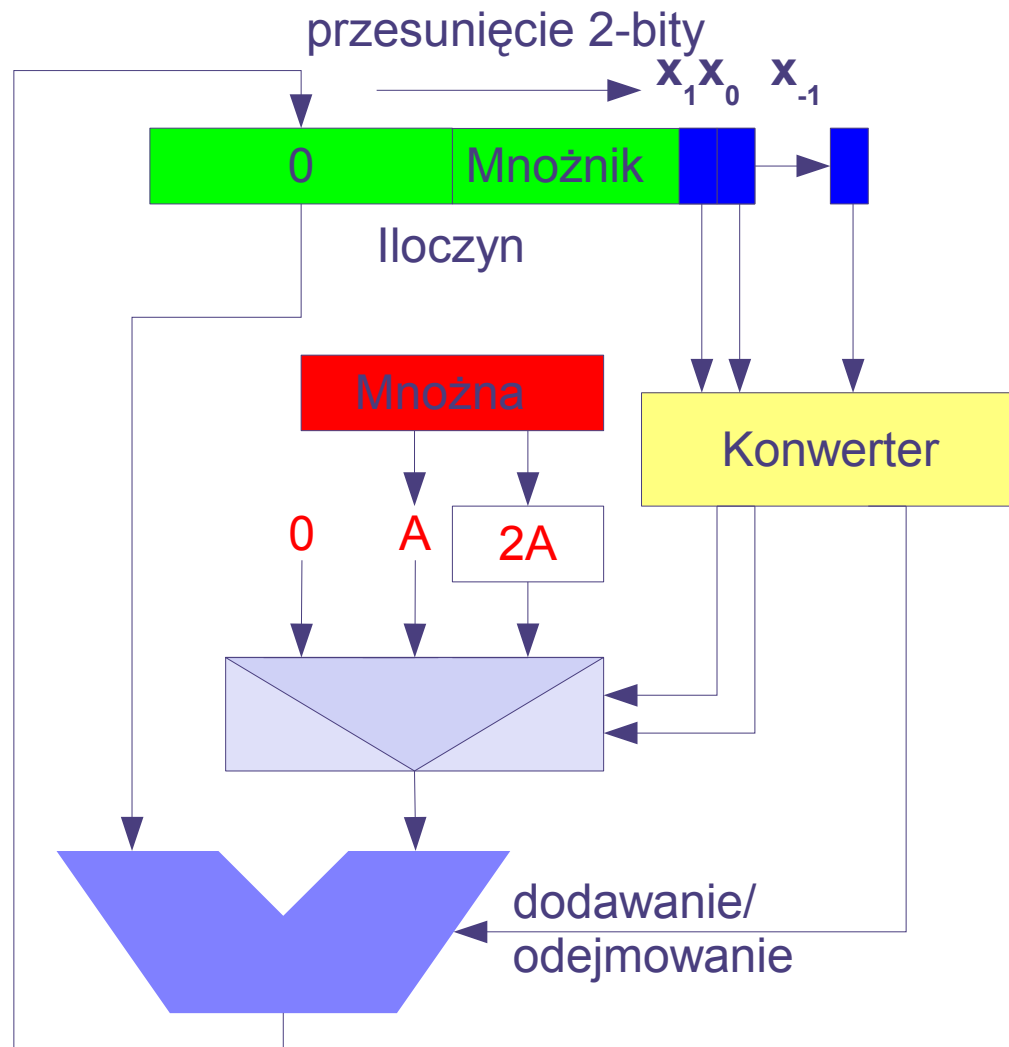
- Konwersja jest szybka (nie ma propagacji)
- Przy wyższych bazach, cyfry mają znak, ale jest ich mniej, co ułatwi obliczanie składników mnożenia

# Mnożenie liczb ze znakiem

## Algorytm Booth'a dla radix-4

Konwersja dla Radix-4

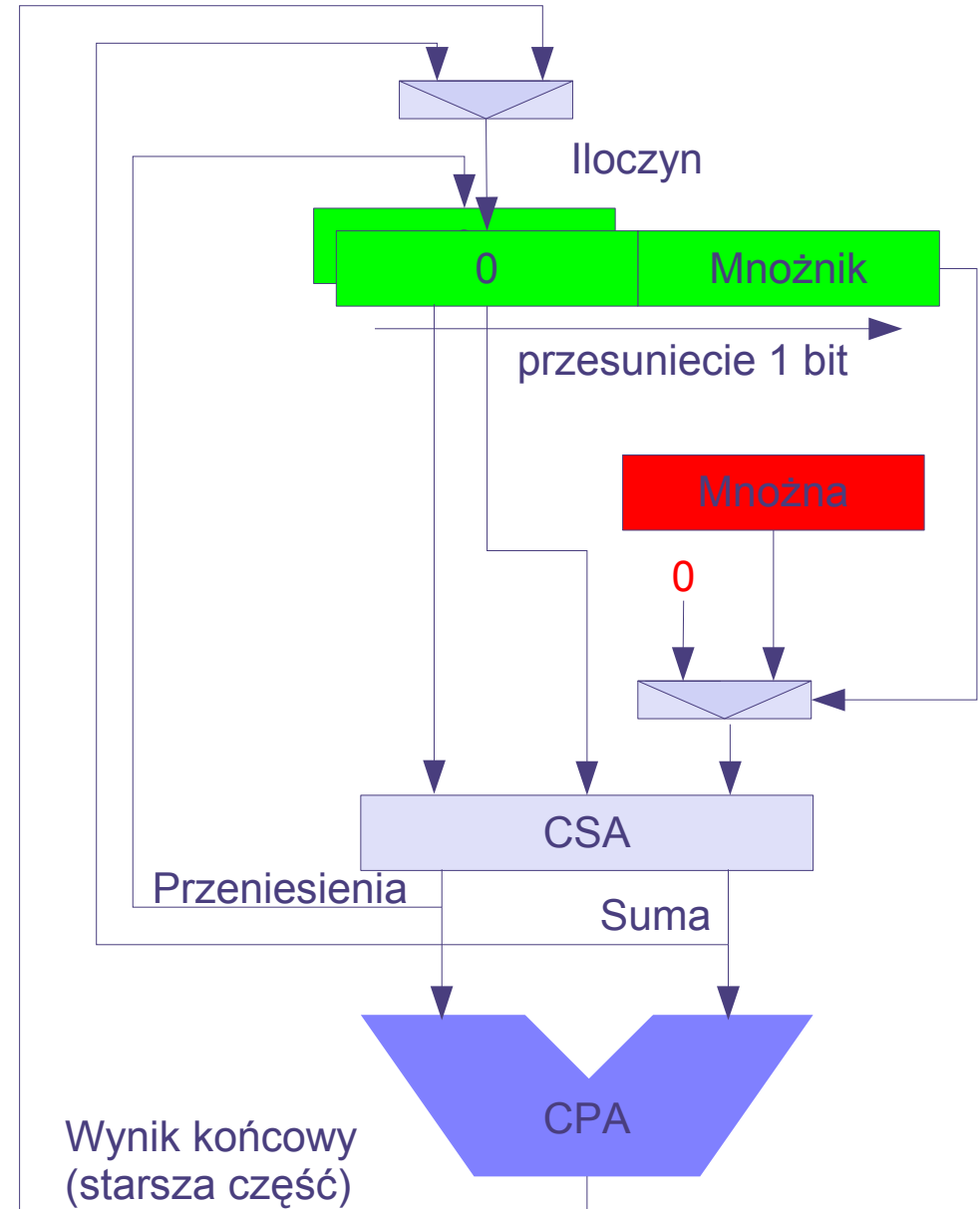
radix-2 [0, 1]	radix-2 [-1, 1]	radix-4 [-2, 2]
000	0 0	0
001	0 1	1
010	1 -1	1
011	1 0	2
100	-1 0	-2
101	-1 1	-1
110	0 -1	-1
111	0 0	0



# Przyspieszenie sumowania – CSA

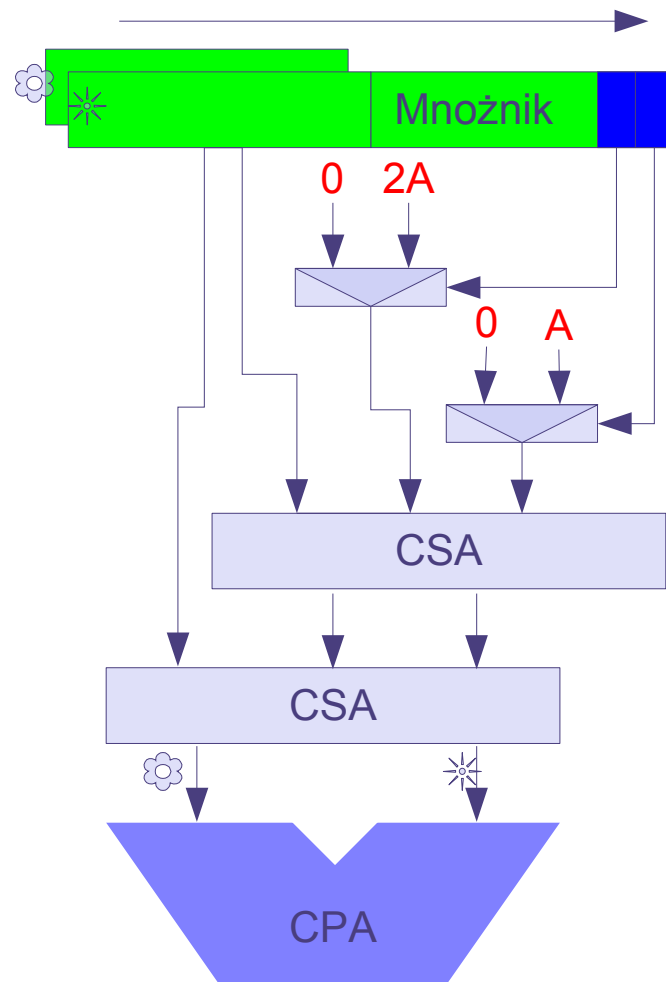
- Dodawanie składników wykonuje CSA –  $\Theta(k)$
- Starsza część wyniku to dwie liczby (redund.)
- Sumator CPA potrzebny jest tylko raz –  $\Theta(\log k)$
- Uwaga na przesuwanie iloczynu !!!
- Sybkość:  $\Theta(k + \log k)$

Przykład dla radix-2

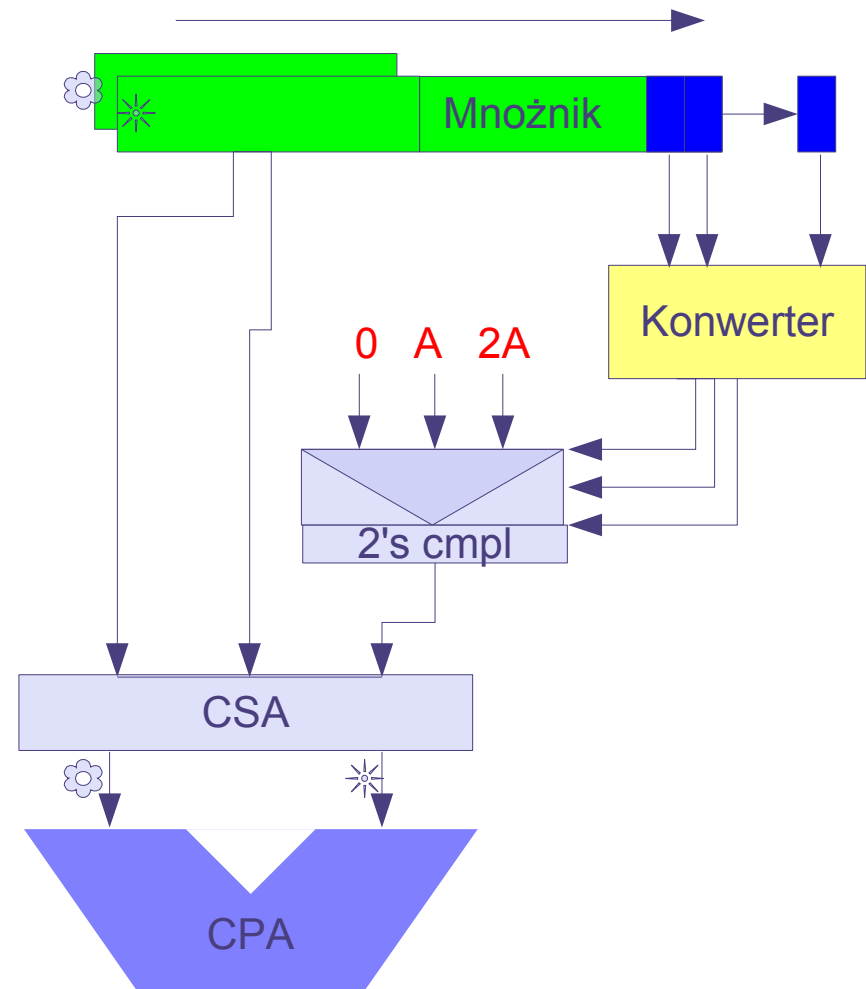


# Szybkie układy mnożące sekwencyjne

Artytmetyka *unsigned*  
(częściowe drzewo CSA)



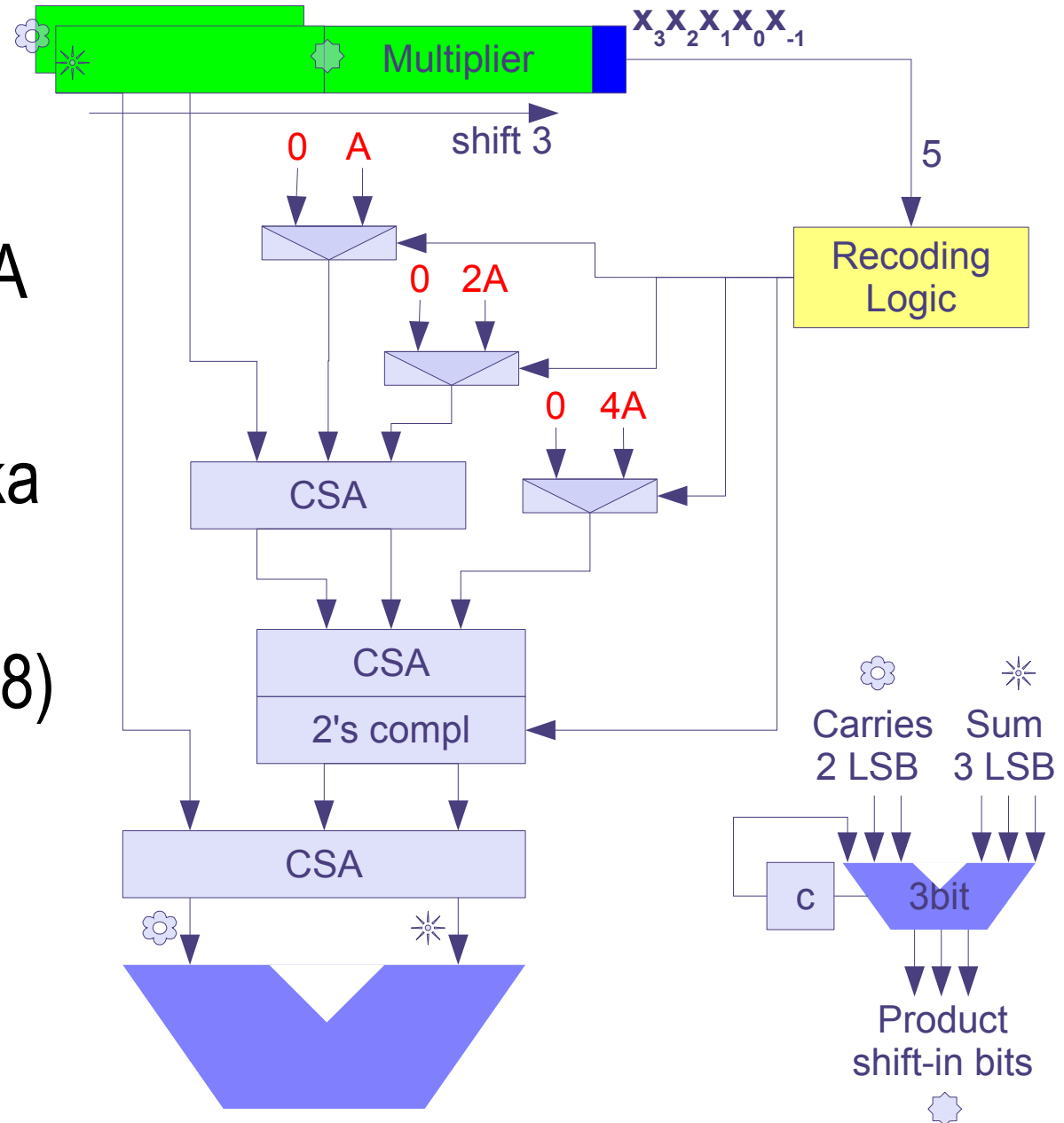
Artytmetyka *signed*  
(algortm Booth'a)



Radix-4 Multipliers

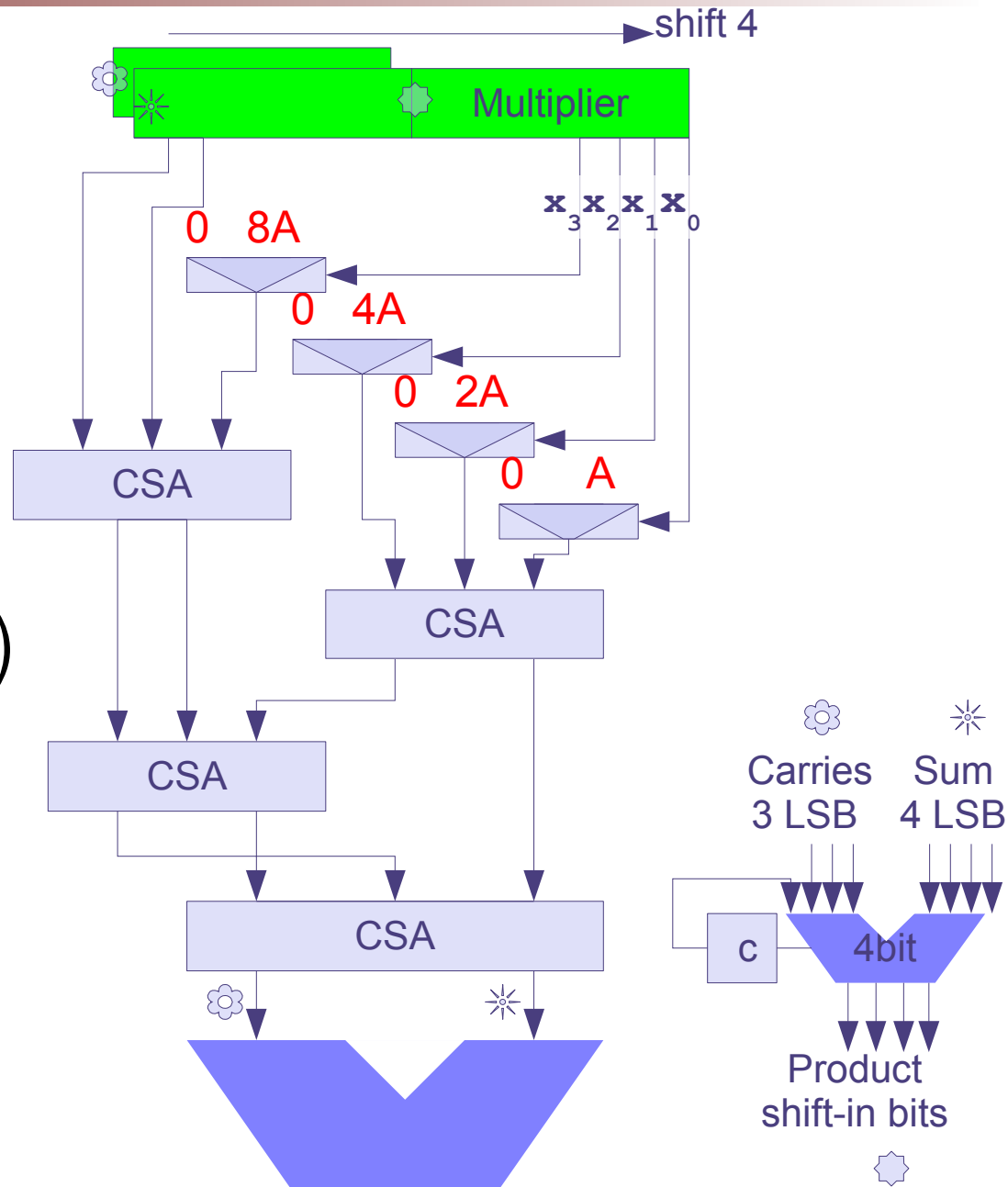
# Przykład: Radix-8, signed

- Składniki sumowań  $0, \pm 1A, \pm 2A, \pm 3A, \pm 4A$
- Konwersja oparta na 5 bitach mnożnika
- Iloczyn przesuwany o 3 bity (cyfra radix-8)



# Przykład: Radix-16, unsigned

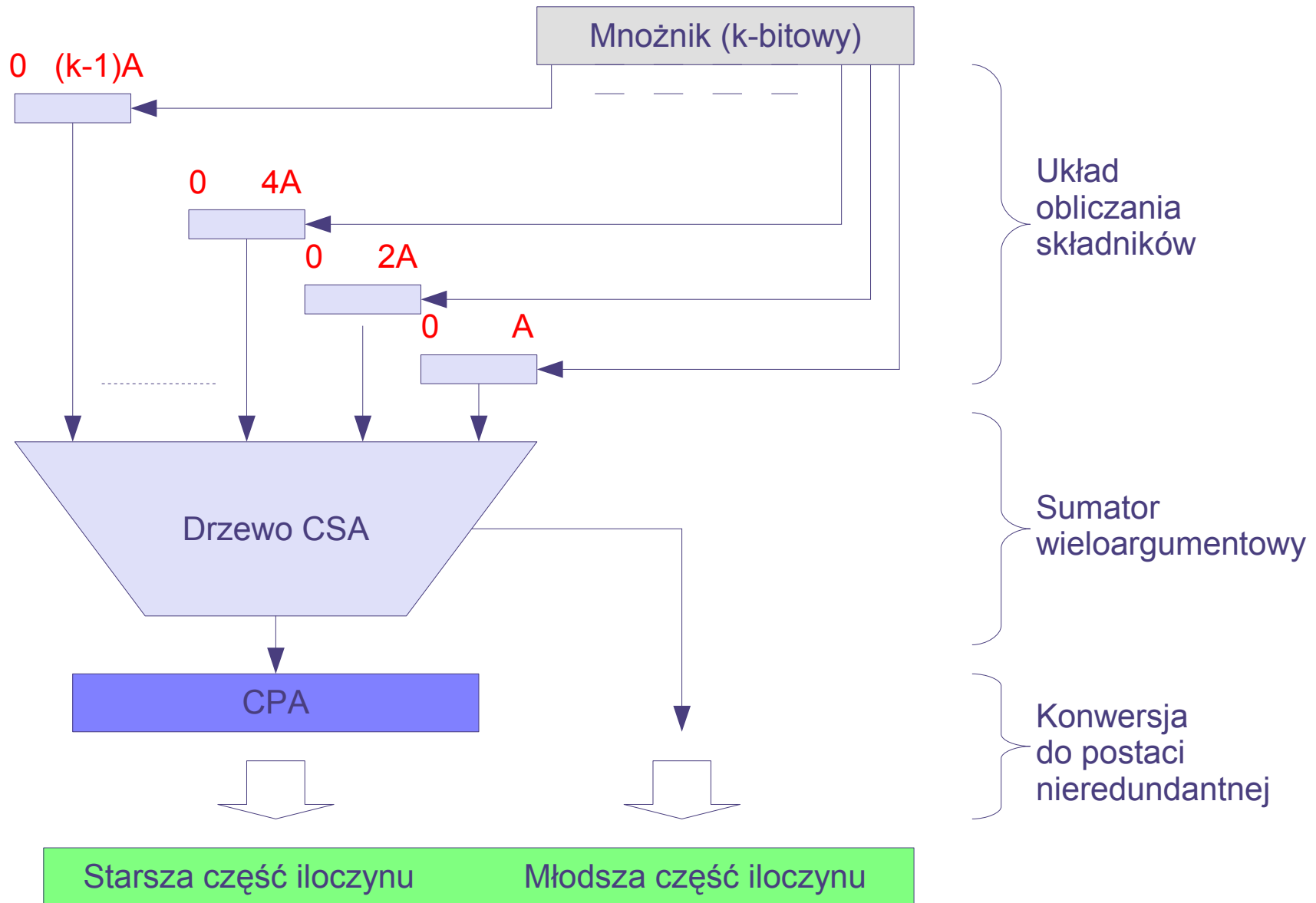
- Składniki sumowań  
0, 1A, 2A, ... 15A
- Obliczanie składników  
w drzewie CSA
- Iloczyn przesuwany  
o 4 bity (cyfra radix-16)



# Drzewiaste układy mnożące

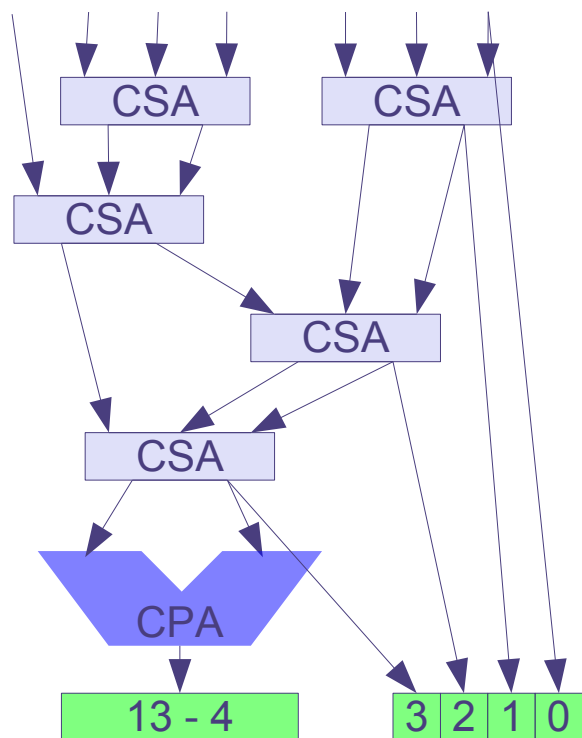
- Wszystkie składniki podawane jednocześnie
- Składniki – wielokrotności mnożnej
- Pełne drzewo redukujące oparte na CSA
- Końcowa konwersja wyniku na CPA
- Struktury regularne i nieregularne
- Szybkość dodawania w drzewie:  $\Theta(\log k)$
- Bardzo droga realizacja (rozmiar)

# Drzewiaste układy mnożące

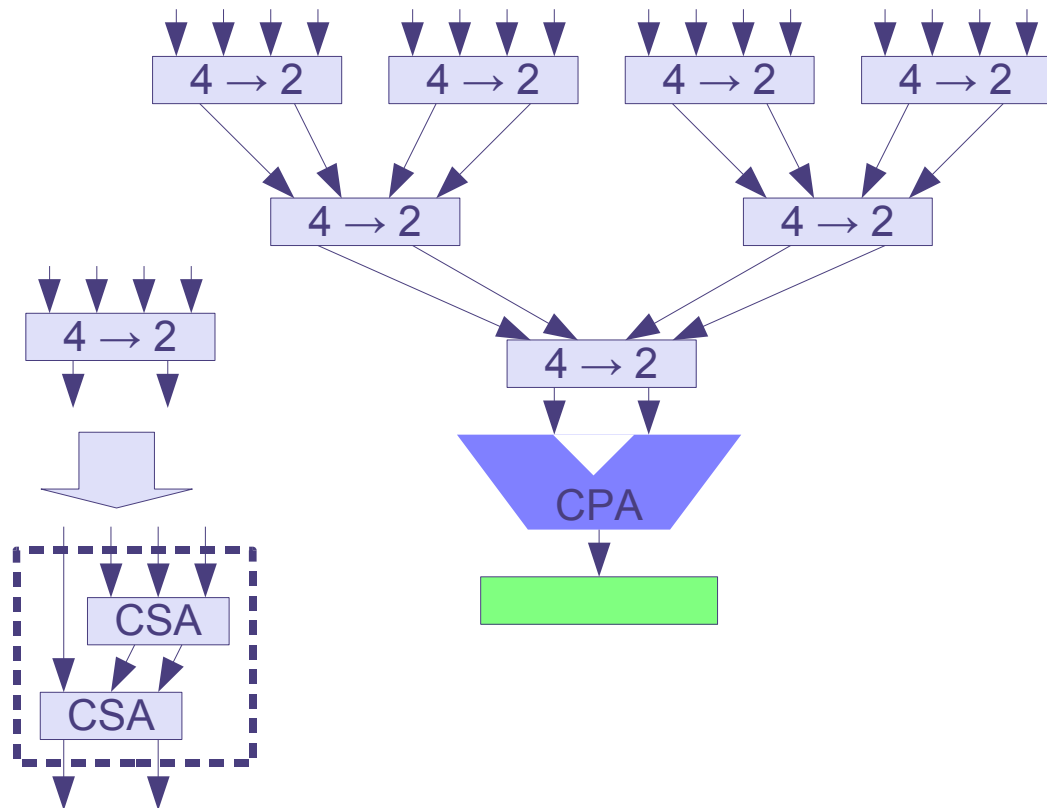


# Struktura drzew CSA

Drzewa nieregularne:  
np. mnożenie liczb 7-bitowych  
(dodawanie 7 składników)

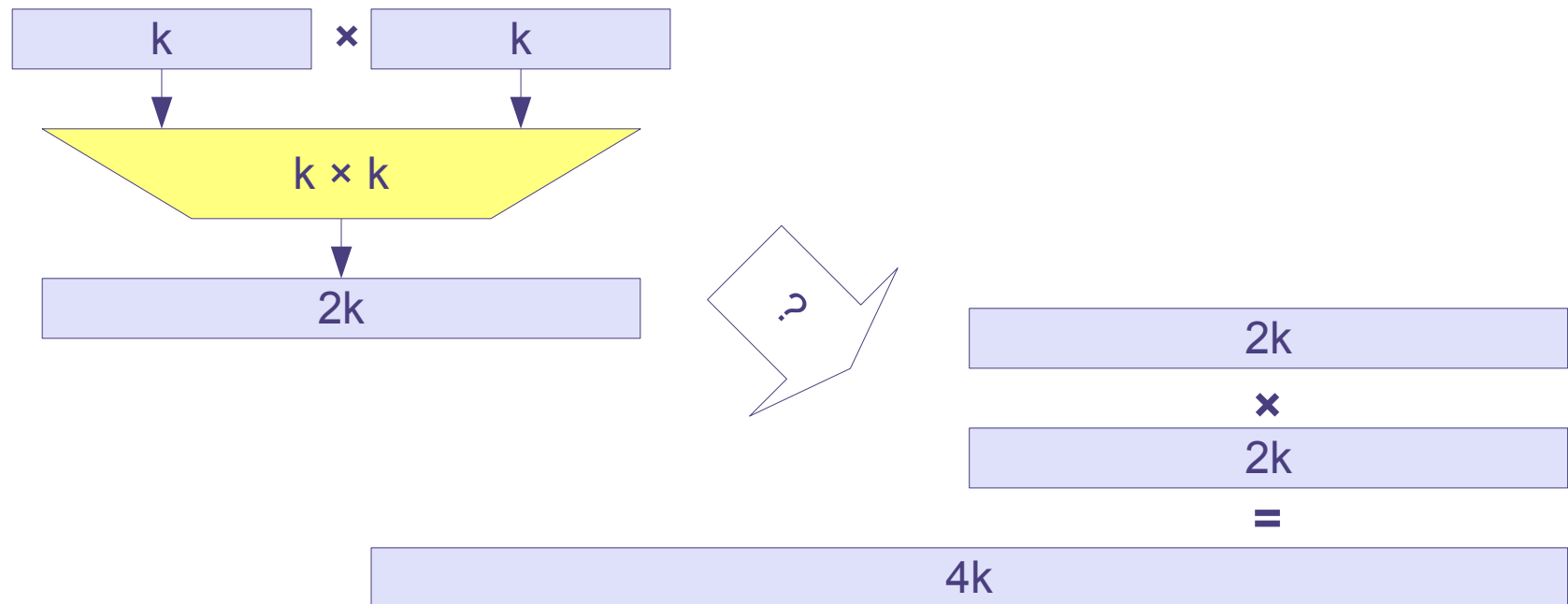


Drzewa regularne:  
np. mnożenie liczb 16-bitowych  
(dodawanie 16 składników)



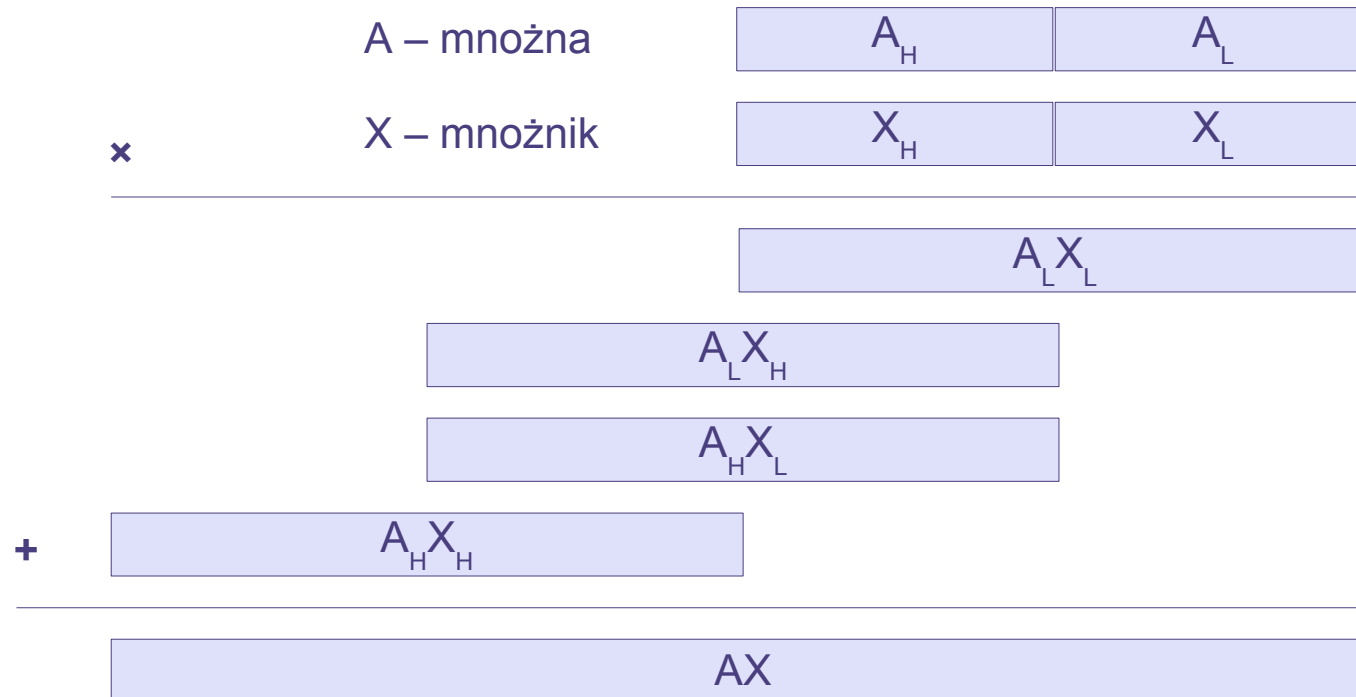
# Mnożenie „długich” liczb

- Jak pomnożyć „długie” liczby dysponując „krótszymi” jednostkami mnożącymi
- np.  $2k * 2k$  za pomocą mnożarki  $k*k$  ?



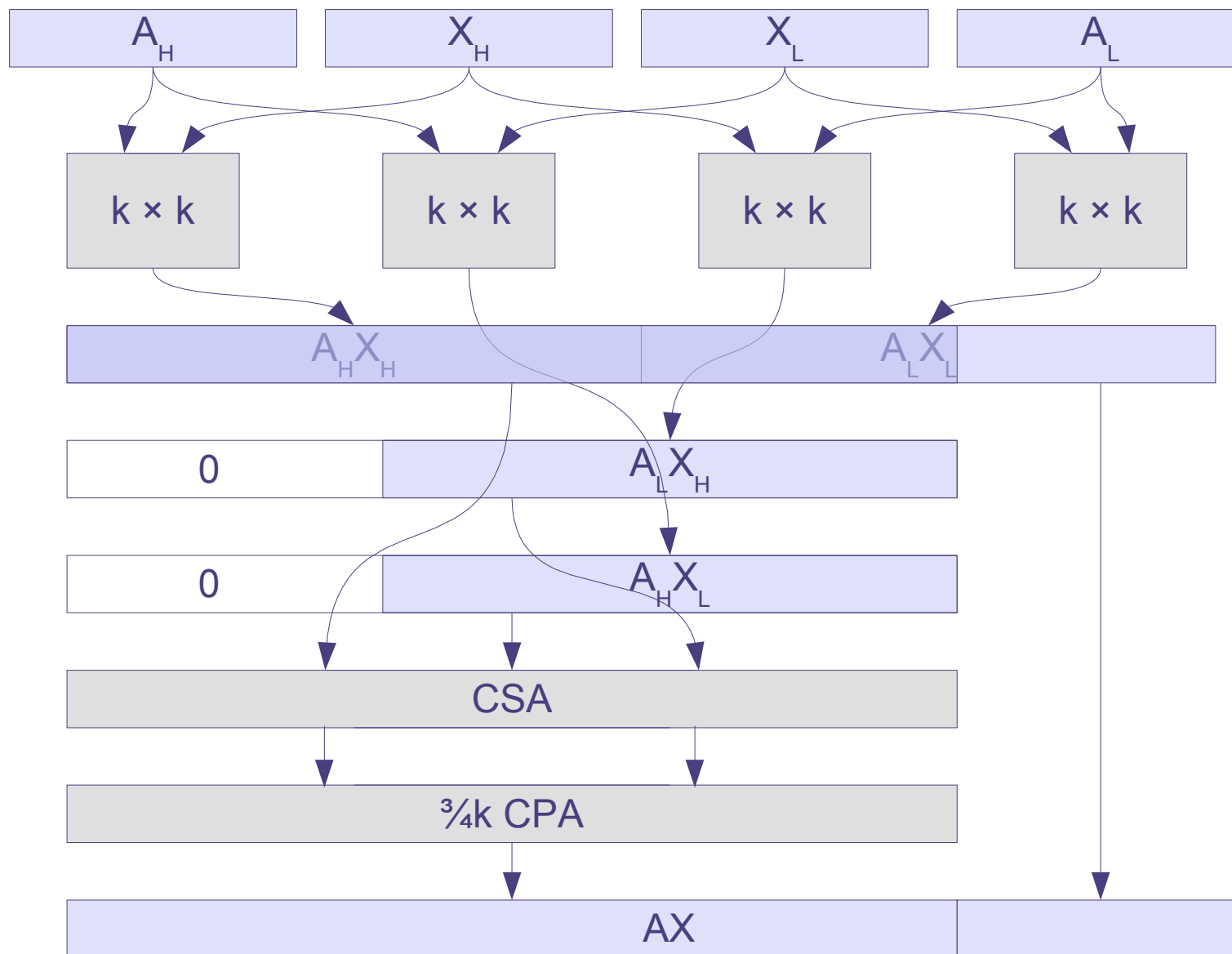
# Dziel i zwyciężaj (*Divide & Conquer*)

- Rozkład liczb ( $2k \rightarrow 2 \times k$ )
- Obliczenia iloczynów częściowych ( $4 \times 2k$ )
- Sumowanie iloczynów częściowych ( $\sum 4 \times 4k$ )



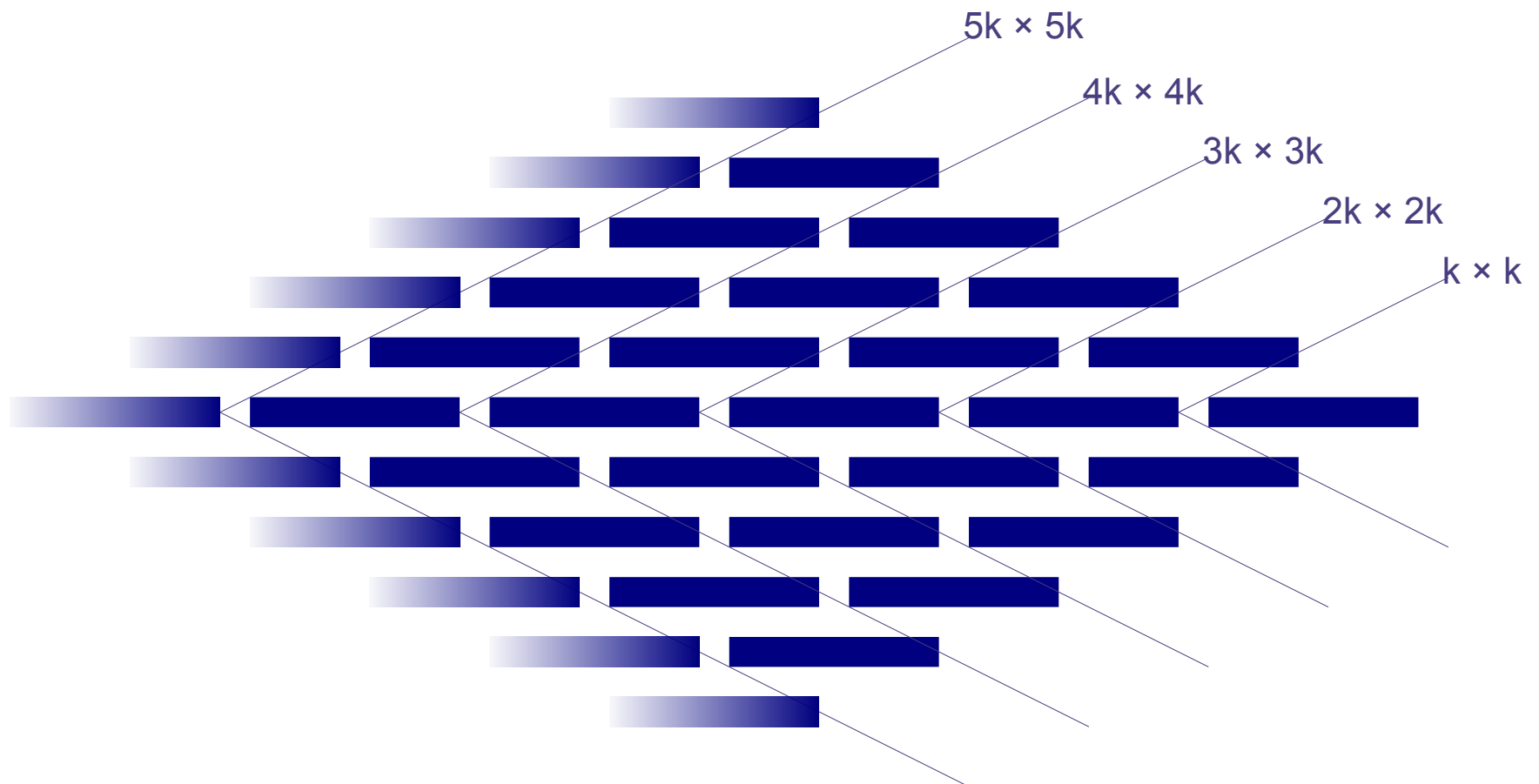


# Dziel i zwyciężaj (2k x 2k)



# Dziel i zwyciężaj - ogólnie

- Mnożenie dwóch  $n \times k$ -bitowych liczb za pomocą  $k$ -bitowych mnożarek wymaga  $n \times n$  mnożeń i dodania  $2n-1$  składników



# Dziel i zwyciężaj - ogólnie

