2의 보수 곱셈기 양자회로

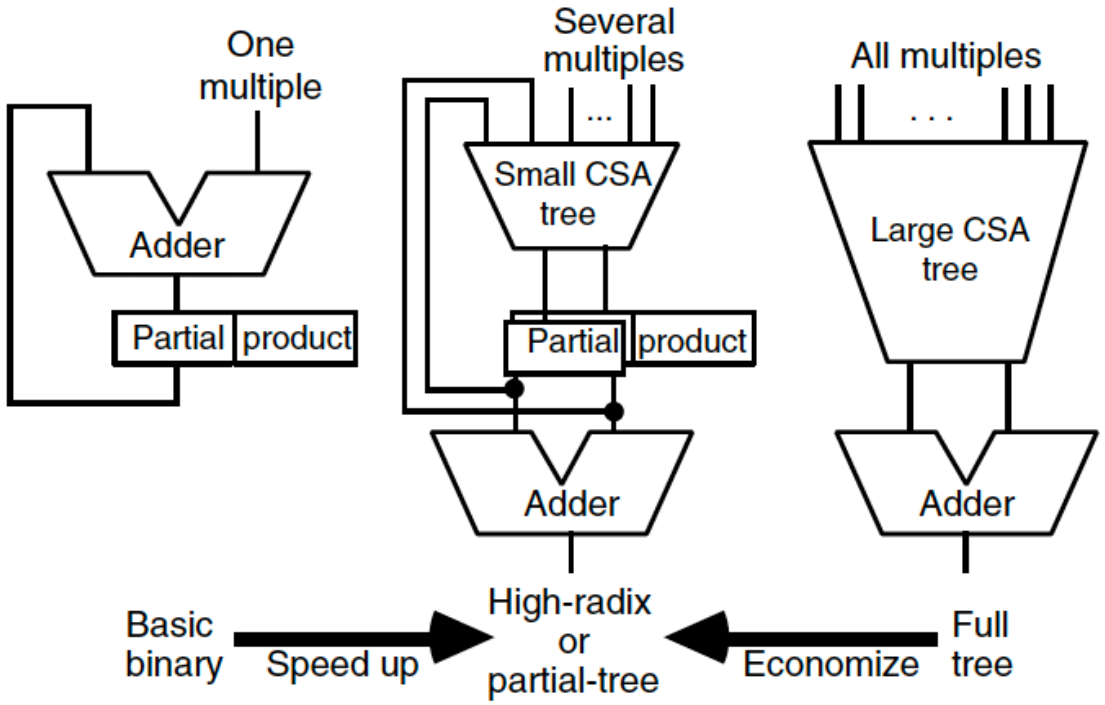
https://youtu.be/k21U9F_2Ug

서론

- 2의 보수 곱셈 양자 회로 구현
  - 1의 보수 :
    모든 비트 반전하여 음수표현, 0의 표현이 0과 -0 두가지
    (+5, 00000101) → 비트반전 (-5, 11111010)

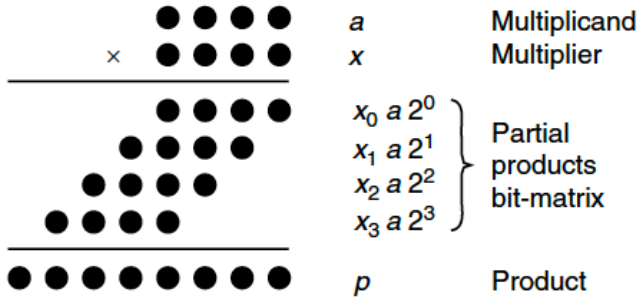
  - 2의 보수 : 모든 비트 반전 후 더하기 1하여 음수표현
    +5 (00000101) → 비트반전 (11111010) → 더하기 1 (11111011) : -5.
    - 컴퓨터상에서 주로 사용되는 표현
    - 양의 정수보다 복잡도가 있음
    - 뺄셈을 더하기로 연산 가능 : 보수를 취하고 더하기

  - 양자 회로 구현
    - 여러 방식에 따른 자원차이
    - Shift-Add 방식 : 자원 효율적
    - Tree 방식 : 고속
    - 고차 기수 : 균형



이진 곱셈

- 이진 곱셈



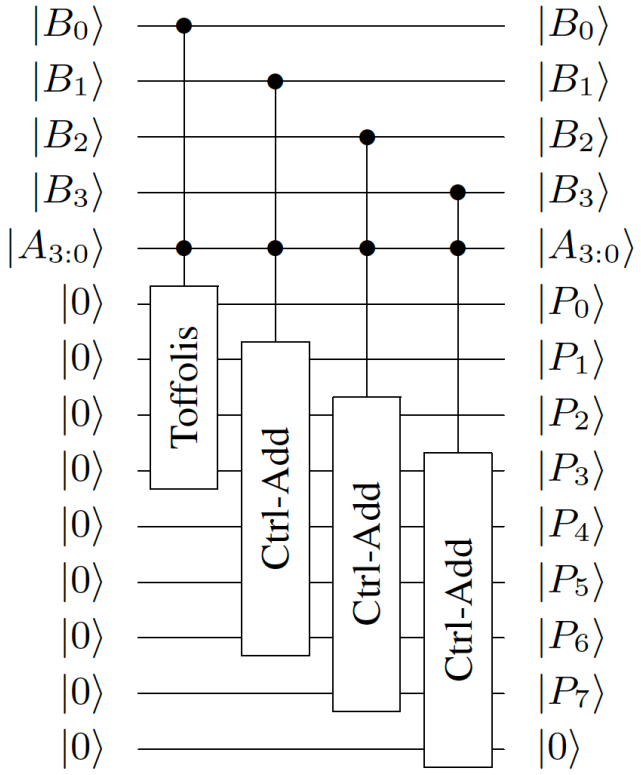
기본 과정

1. Multiplier의 최하위 비트부터 시작하여 Multiplicand에 곱함
2. 각 단계마다 피곱수를 한 자리씩 왼쪽으로 시프트
3. 모든 곱셈 결과를 합산

(a) Right-shift algorithm

a		1	0	1	0			
x		1	0	1	1			
$p^{(0)}$		0	0	0	0			
$+x_0 a$		1	0	1	0			
$2p^{(1)}$	0	1	0	1	0			
$p^{(1)}$		0	1	0	1	0		
$+x_1 a$		1	0	1	0			
$2p^{(2)}$	0	1	1	1	1	0		
$p^{(2)}$		0	1	1	1	1	0	
$+x_2 a$		0	0	0	0			
$2p^{(3)}$	0	0	1	1	1	1	0	
$p^{(3)}$		0	0	1	1	1	1	0
$+x_3 a$		1	0	1	0			
$2p^{(4)}$	0	1	1	0	1	1	1	0
$p^{(4)}$		0	1	1	0	1	1	1	0

- 기본 Shift-add 덧셈



- 양자회로 구현

2의 보수 곱셈

- 기본적으로 양수 곱셈과 유사,
- 피연산자 중 하나 또는 둘 모두가 음수일 때 추가 고려

(a) Right-shift algorithm

a			1	0	1	0			
x			1	0	1	1			
$p^{(0)}$			0	0	0	0			
$+x_0a$			1	0	1	0			
$2p^{(1)}$	0	1	0	1	0				
$p^{(1)}$		0	1	0	1	0			
$+x_1a$		1	0	1	0				
$2p^{(2)}$	0	1	1	1	1	0			
$p^{(2)}$		0	1	1	1	1	0		
$+x_2a$		0	0	0	0				
$2p^{(3)}$	0	0	1	1	1	1	0		
$p^{(3)}$		0	0	1	1	1	1	0	
$+x_3a$		1	0	1	0				
$2p^{(4)}$	0	1	1	0	1	1	1	0	
$p^{(4)}$		0	1	1	0	1	1	1	0

- 둘 다 양수

a			1	0	1	1	0		
x			0	1	0	1	1		
$p^{(0)}$			0	0	0	0	0		
$+x_0a$			1	0	1	1	0		
$2p^{(1)}$	1	1	0	1	1	0			
$p^{(1)}$		1	1	0	1	1	0		
$+x_1a$		1	0	1	1	0			
$2p^{(2)}$	1	1	0	0	0	1	0		
$p^{(2)}$		1	1	0	0	0	1	0	
$+x_2a$		0	0	0	0	0			
$2p^{(3)}$	1	1	1	0	0	0	1	0	
$p^{(3)}$		1	1	1	0	0	0	1	0
$+x_3a$		1	0	1	1	0			
$2p^{(4)}$	1	1	0	0	1	0	0	1	0
$p^{(4)}$		1	1	0	0	1	0	0	1	0
$+x_4a$		0	0	0	0	0			
$2p^{(5)}$	1	1	1	0	0	1	0	0	1	0
$p^{(5)}$		1	1	1	0	0	1	0	0	1	0

- 멀티플라이어가 양수

a			1	0	1	1	0		
x			1	0	1	0	1		
$p^{(0)}$			0	0	0	0	0		
$+x_0a$			1	0	1	1	0		
$2p^{(1)}$	1	1	0	1	1	0			
$p^{(1)}$		1	1	0	1	1	0		
$+x_1a$		0	0	0	0	0			
$2p^{(2)}$	1	1	1	0	1	1	0		
$p^{(2)}$		1	1	1	0	1	1	0	
$+x_2a$		1	0	1	1	0			
$2p^{(3)}$	1	1	0	0	1	1	1	0	
$p^{(3)}$		1	1	0	0	1	1	1	0
$+x_3a$		0	0	0	0	0			
$2p^{(4)}$	1	1	1	0	0	1	1	1	0
$p^{(4)}$		1	1	1	0	0	1	1	1	0
$+(-x_4a)$		0	1	0	1	0			
$2p^{(5)}$	0	0	0	1	1	0	1	1	1	0
$p^{(5)}$		0	0	0	1	1	0	1	1	1	0

- 멀티플라이어가 음수

양자 회로 구현 고려사항

- 2의 보수 변환
  - 2의 보수 변환 방법 : 모든 비트 반전 후 더하기 1 하여 음수표현
  - 더하기 1의 경우 케리가 발생 → 추가 토플리 게이트

  ➢ 뺄셈 활용
  A-B = $A+\overline{B}+1$
  A-B = $\overline{\bar{A}+B}$

- 피연산자에 따른 연산차이
  - If,else 없이 논리연산으로 동작필요
  - 부호의 결정
  - 0인경우…

a		1	0	1	1	0				
x		0	1	0	1	1				
$p^{(0)}$		0	0	0	0	0				
$+x_0a$		1	0	1	1	0				
$2p^{(1)}$	1	1	0	1	1	0				
$p^{(1)}$		1	1	0	1	1	0			
$+x_1a$		1	0	1	1	0				
$2p^{(2)}$	1	1	0	0	0	1	0			
$p^{(2)}$		1	1	0	0	0	1	0		
$+x_2a$		0	0	0	0	0				
$2p^{(3)}$	1	1	1	0	0	0	1	0		
$p^{(3)}$		1	1	1	0	0	0	1	0	
$+x_3a$		1	0	1	1	0				
$2p^{(4)}$	1	1	0	0	1	0	0	1	0	
$p^{(4)}$		1	1	0	0	1	0	0	1	0
$+x_4a$		0	0	0	0	0				
$2p^{(5)}$	1	1	1	0	0	1	0	0	1	0
$p^{(5)}$		1	1	1	0	0	1	0	0	1	0

a		1	0	1	1	0				
x		1	0	1	0	1				
$p^{(0)}$		0	0	0	0	0				
$+x_0a$		1	0	1	1	0				
$2p^{(1)}$	1	1	0	1	1	0				
$p^{(1)}$		1	1	0	1	1	0			
$+x_1a$		0	0	0	0	0				
$2p^{(2)}$	1	1	1	0	1	1	0			
$p^{(2)}$		1	1	1	0	1	1	0		
$+x_2a$		1	0	1	1	0				
$2p^{(3)}$	1	1	0	0	1	1	1	0		
$p^{(3)}$		1	1	0	0	1	1	1	0	
$+x_3a$		0	0	0	0	0				
$2p^{(4)}$	1	1	1	0	0	1	1	1	0	
$p^{(4)}$		1	1	1	0	0	1	1	1	0
$+(-x_4a)$		0	1	0	1	0				
$2p^{(5)}$	0	0	0	1	1	0	1	1	1	0
$p^{(5)}$		0	0	0	1	1	0	1	1	1	0

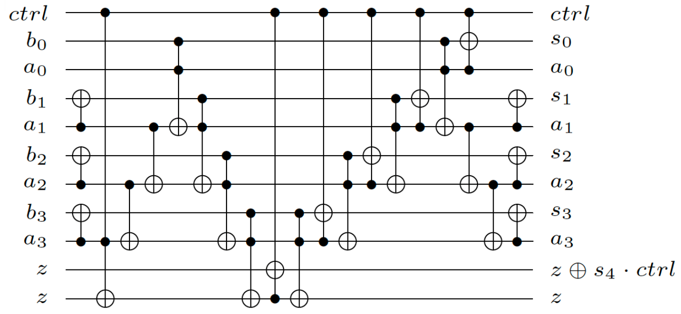
booth-recoding

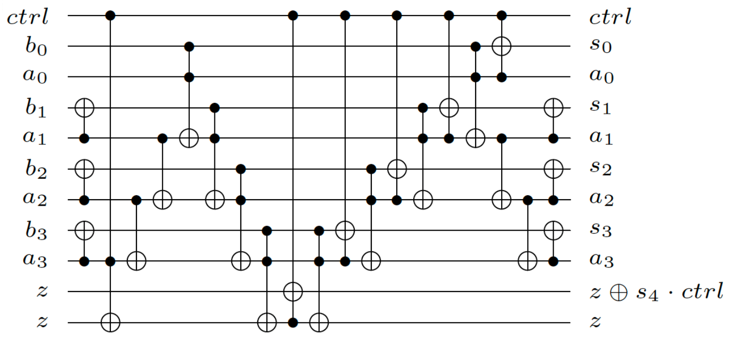
- 곱하는 수(multiplier)의 표현을 변형
  - ➢ 부호 계산이 간단

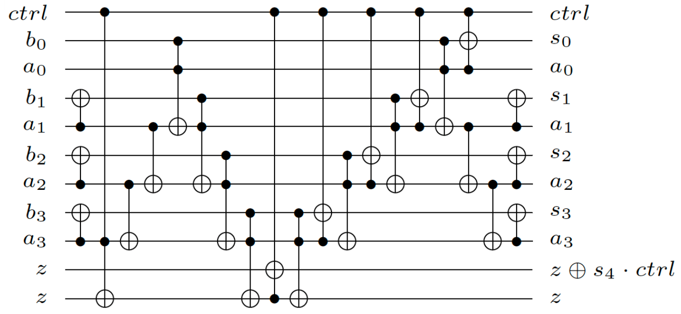
x_i	x_{i-1}	y_i	Explanation
0	0	0	No string of 1s in sight
0	1	1	End of string of 1s in x
1	0	$^-1$	Beginning of string of 1s in x
1	1	0	Continuation of string of 1s in x

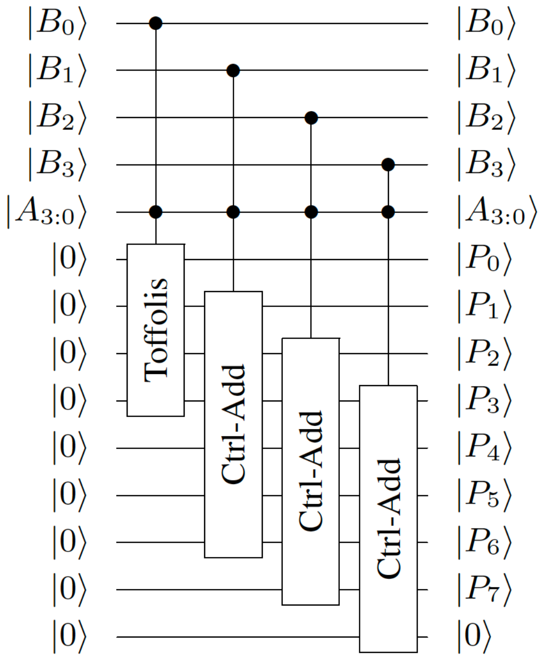
```
============================
a          1 0 1 1 0
x          1 0 1 0 1    Multiplier
y         -1 1-1 1-1    Booth-recoded
============================
p(0)       0 0 0 0 0
+y0a       0 1 0 1 0
----------------------
2p(1)    0 0 1 0 1 0
p(1)       0 0 1 0 1  0
+y1a       1 0 1 1 0
----------------------
2p(2)    1 1 1 0 1 1  0
p(2)       1 1 1 0 1  1 0
+y2a       0 1 0 1 0
----------------------
2p(3)    0 0 0 1 1 1  1 0
p(3)       0 0 0 1 1  1 1 0
+y3a       1 0 1 1 0
----------------------
2p(4)    1 1 1 0 0 1  1 1 0
p(4)       1 1 1 0 0  1 1 1 0
+y4a       0 1 0 1 0
----------------------
2p(5)    0 0 0 1 1 0  1 1 1 0
p(5)       0 0 0 1 1  0 1 1 1 0
============================
```

booth-recoding 양자회로

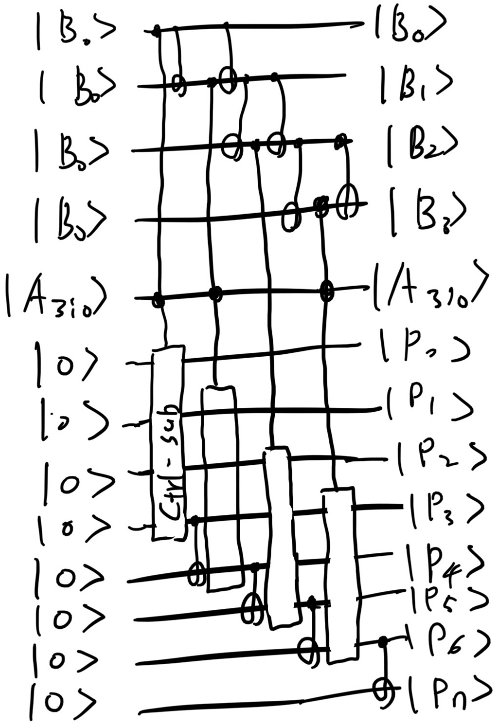




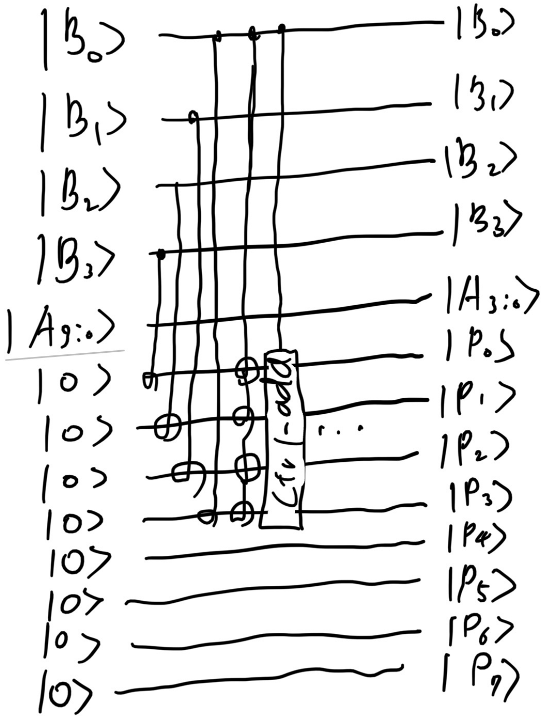




- 양의 정수 곱셈기



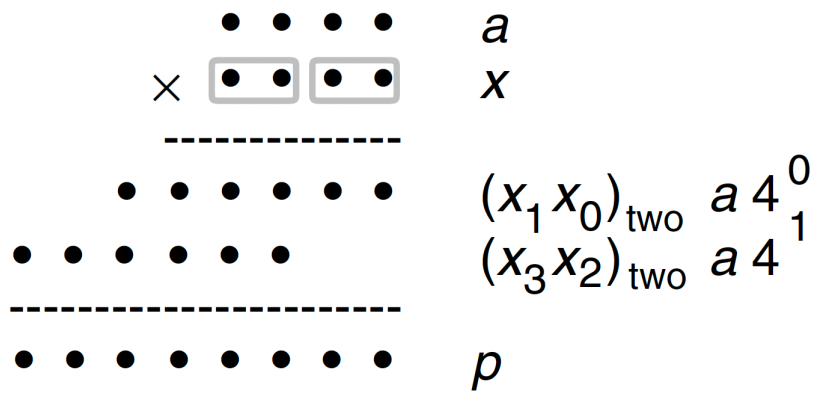
$\bar{A} + B$



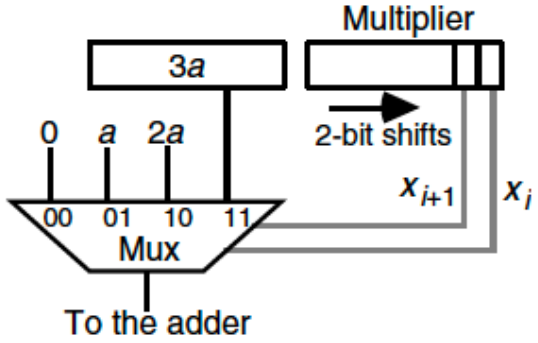
$A + \bar{B} + 1$

Radix-4

- 한 번의 연산으로 2비트의 곱해지는 수(Multiplicand)를 처리
  - 2a의 연산이 빠른경우 효율적



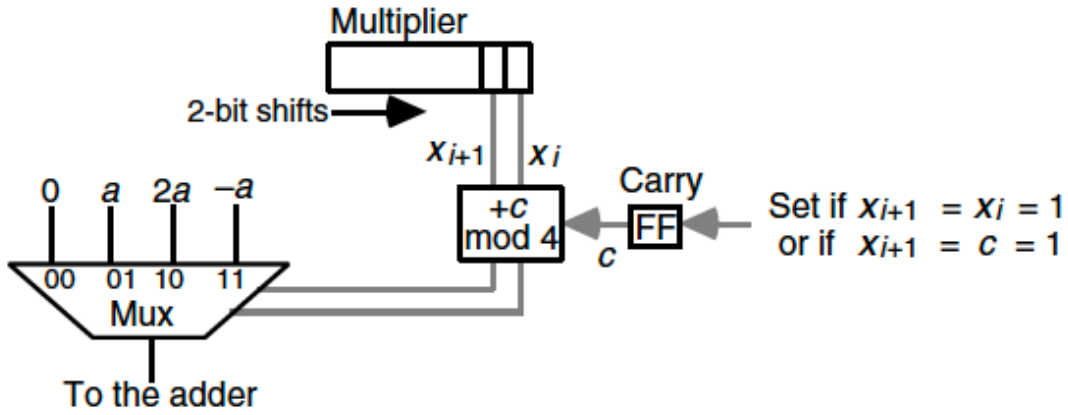
a				0	1	1	0			
$3a$	0	1	0	0	1	0				
x				1	1	1	0			
$p^{(0)}$				0	0	0	0			
$+(x_1x_0)_{two}\ a$	0	0	1	1	0	0				
$4p^{(1)}$	0	0	1	1	0	0				
$p^{(1)}$				0	0	1	1	0	0	
$+(x_3x_2)_{two}\ a$	0	1	0	0	1	0				
$4p^{(2)}$	0	1	0	1	0	1	0	0		
$p^{(2)}$				0	1	0	1	0	1	0	0

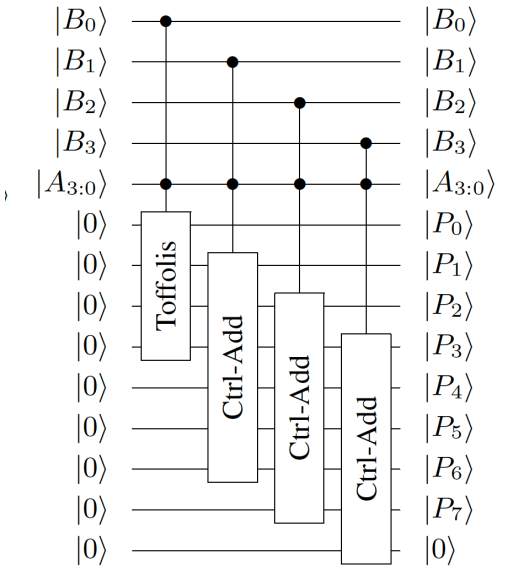


Radix-4 Booth's recoding

x_{i+1}	x_i	x_{i-1}	y_{i+1}	y_i	$z_{i/2}$	Explanation
0	0	0	0	0	0	No string of 1s in sight
0	0	1	0	1	1	End of a string of 1s in x
0	1	0	1	−1	1	Isolated 1 in x
0	1	1	1	0	2	End of a string of 1s in x
1	0	0	−1	0	−2	Beginning of a String of 1s in x
1	0	1	−1	1	−1	End one string, begin new string
1	1	0	0	−1	−1	Beginning of a string of 1s in x
1	1	1	0	0	0	Continuation of string of 1s in x

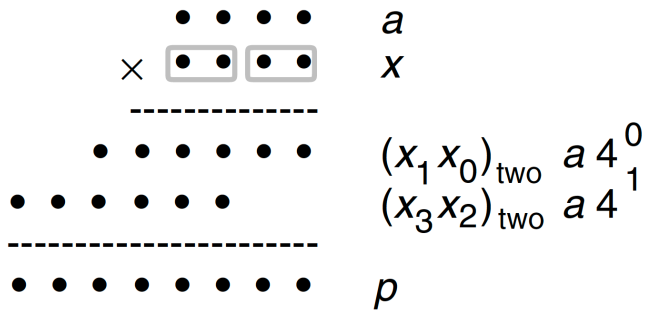
a			0	1	1	0			
x			1	0	1	0			
z				$\bar{1}$		$\bar{2}$	Radix-4 recoded version of x		
$p^{(0)}$	0	0	0	0	0	0			
$+z_0a$	1	1	0	1	0	0			
$4p^{(1)}$	1	1	0	1	0	0			
$p^{(1)}$	1	1	1	1	0	1	0	0	
$+z_1a$	1	1	1	0	1	0			
$4p^{(2)}$	1	1	0	1	1	1	0	0	
$p^{(2)}$			1	1	0	1	1	1	0 0



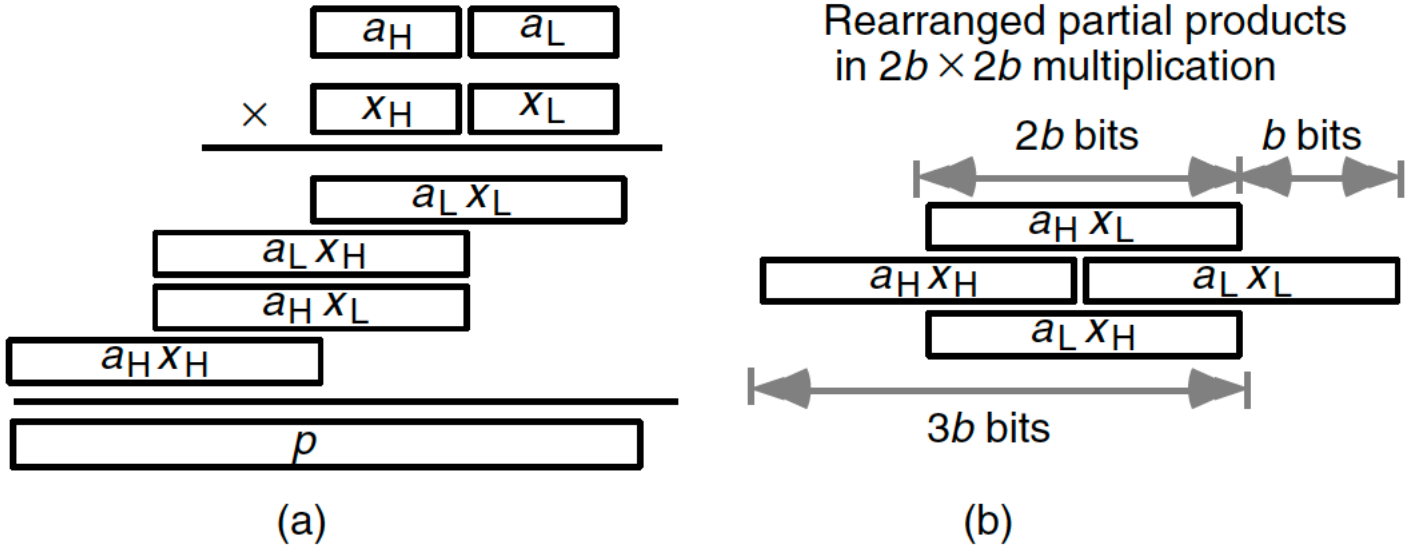


Radix-4 Booth's recoding

- 병렬
  - 4x2+ 4x2는 병렬 x



- 분할정복
  - 추가 큐비트 사용, 회로 깊이 감소



- 문제점

Baugh–Wooley 2's-complement multiplication

$$
\begin{array}{cccccccccc}
 & & & & a_4\bar{x}_0 & a_3x_0 & a_2x_0 & a_1x_0 & a_0x_0 & \\
 & & & a_4\bar{x}_1 & a_3x_1 & a_2x_1 & a_1x_1 & a_0x_1 & & \\
 & & a_4\bar{x}_2 & a_3x_2 & a_2x_2 & a_1x_2 & a_0x_2 & & & \\
 & a_4\bar{x}_3 & a_3x_3 & a_2x_3 & a_1x_3 & a_0x_3 & & & & \\
a_4x_4 & \bar{a}_3x_4 & \bar{a}_2x_4 & \bar{a}_1x_4 & \bar{a}_0x_4 & & & & & \\
\bar{a}_4 & & & & a_4 & & & & & \\
1 & \bar{x}_4 & & & x_4 & & & & & \\
\hline
p_9 & p_8 & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}
$$

(c) Baugh–Wooley method's bit-matrix

$$
\begin{array}{cccccccccc}
 & & & & -a_4x_0 & a_3x_0 & a_2x_0 & a_1x_0 & a_0x_0 & \\
 & & & -a_4x_1 & a_3x_1 & a_2x_1 & a_1x_1 & a_0x_1 & & \\
 & & -a_4x_2 & a_3x_2 & a_2x_2 & a_1x_2 & a_0x_2 & & & \\
 & -a_4x_3 & a_3x_3 & a_2x_3 & a_1x_3 & a_0x_3 & & & & \\
a_4x_4 & -a_3x_4 & -a_2x_4 & -a_1x_4 & -a_0x_4 & & & & & \\
\hline
p_9 & p_8 & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}
$$

(b) 2's-complement bit-matrix

$$
\begin{array}{cccccccccc}
 & & & & \overline{a_4x_0} & a_3x_0 & a_2x_0 & a_1x_0 & a_0x_0 & \\
 & & & \overline{a_4x_1} & a_3x_1 & a_2x_1 & a_1x_1 & a_0x_1 & & \\
 & & \overline{a_4x_2} & a_3x_2 & a_2x_2 & a_1x_1 & a_0x_2 & & & \\
 & \overline{a_4x_3} & a_3x_3 & a_2x_3 & a_1x_3 & a_0x_3 & & & & \\
a_4x_4 & \overline{a_3x_4} & \overline{a_2x_4} & \overline{a_1x_4} & \overline{a_0x_4} & & & & & \\
1 & & & 1 & & & & & & \\
\hline
p_9 & p_8 & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}
$$

(d) Modified Baugh–Wooley method

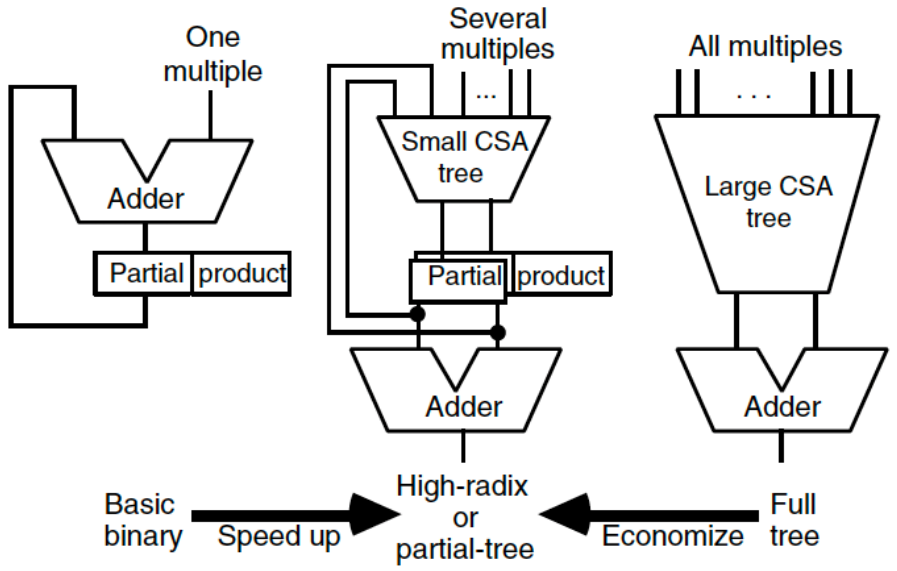
$$
\begin{array}{cccccccccccccccc}
 & & & & & & & & y_7 & y_6 & y_5 & y_4 & y_3 & y_2 & y_1 & y_0 \\
 & & & & & & & & x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \\
\hline
 & & & & & & & 1 & \overline{p_{70}} & p_{60} & p_{50} & p_{40} & p_{30} & p_{20} & p_{10} & p_{00} \\
 & & & & & & & \overline{p_{71}} & p_{61} & p_{51} & p_{41} & p_{31} & p_{21} & p_{11} & p_{01} & \\
 & & & & & & \overline{p_{72}} & p_{62} & p_{52} & p_{42} & p_{32} & p_{22} & p_{12} & p_{02} & & \\
 & & & & & \overline{p_{73}} & p_{63} & p_{53} & p_{43} & p_{33} & p_{23} & p_{13} & p_{03} & & & \\
 & & & & \overline{p_{74}} & p_{64} & p_{54} & p_{44} & p_{34} & p_{24} & p_{14} & p_{04} & & & & \\
 & & & \overline{p_{75}} & p_{65} & p_{55} & p_{45} & p_{35} & p_{25} & p_{15} & p_{05} & & & & & \\
 & & \overline{p_{76}} & p_{66} & p_{56} & p_{46} & p_{36} & p_{26} & p_{16} & p_{06} & & & & & & \\
 & p_{77} & \overline{p_{67}} & \overline{p_{57}} & \overline{p_{47}} & \overline{p_{37}} & \overline{p_{27}} & \overline{p_{17}} & \overline{p_{07}} & & & & & & & \\
\hline
\overline{s_{15}} & s_{14} & s_{13} & s_{12} & s_{11} & s_{10} & s_9 & s_8 & s_7 & s_6 & s_5 & s_4 & s_3 & s_2 & s_1 & s_0
\end{array}
$$

결론

- 2의 보수 곱셈 양자 회로의 기법에 따른 구현 및 자원 비교 목표
  - *Booth's recoding*
  - *Hight-Radix*
  - Baugh–Wooley 기반 트리
  - 여러기법 복합

  - 자원비교
    - 제한된 큐비트 : 임베디드 컴퓨터 유사
      리소스를 절약-보조 큐빗의 Uncompute를 포함

    - 보조 큐빗 대비 Toffoli-depth, Toffoli-count



Q & A