Hybrid CORDIC 2.A Sine/Cosine Generator Architecture

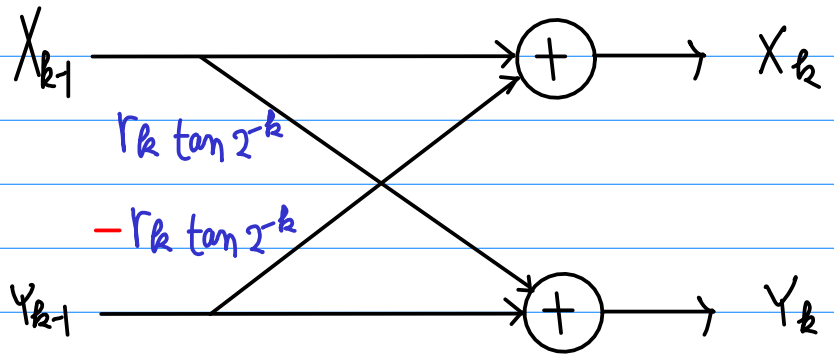
20171028

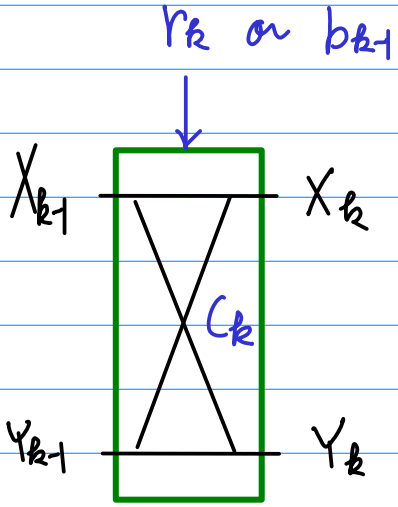
.

The details moved to

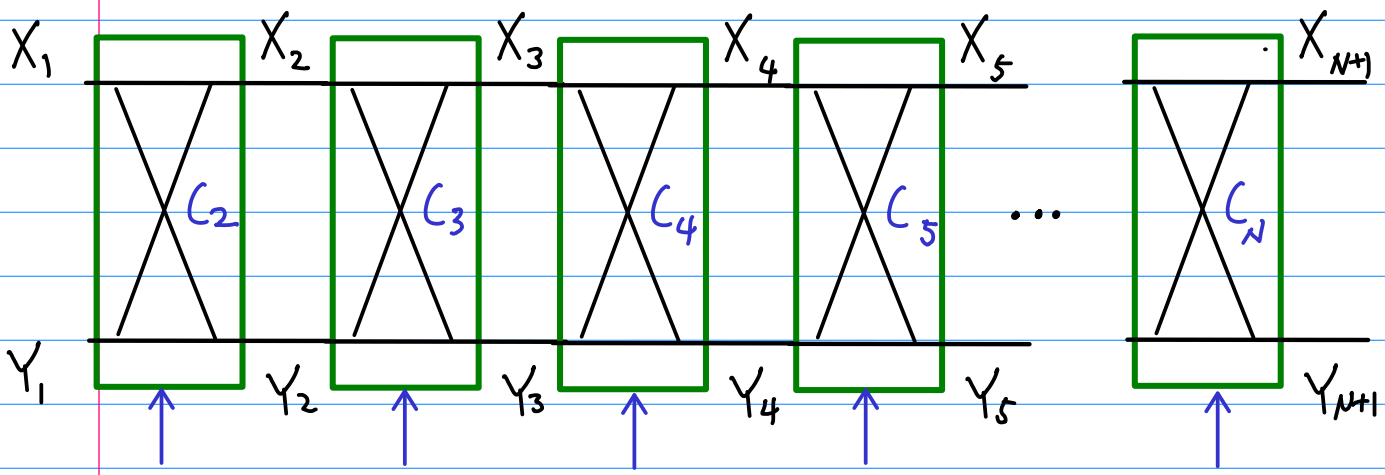
https://en.wikiversity.org/wiki/Butterfly_Hardware_Implementations

.





Butterfly



$C_2 = \tan\left(\frac{1}{2^2}\right)$ $C_3 = \tan\left(\frac{1}{2^3}\right)$ $C_4 = \tan\left(\frac{1}{2^4}\right)$ $C_5 = \tan\left(\frac{1}{2^5}\right)$

$K \cos\phi_0 \rightarrow X_1$ $\quad X_{N+1} \rightarrow \cos\theta$

$K \sin\phi_0 \rightarrow Y_1$ $\quad Y_{N+1} \rightarrow \sin\theta$

$\theta \rightarrow \{b_1, b_2, \cdots, b_N\}$

the initial (X_0, Y_0) always the same

merge the first $B/3$ butterflies

→ $2^{B/3}$ words ROM implementation

→ no need $\tan \theta_k$ multipliers

→ $\{b_1, b_2, \cdots, b_{B/3}\} \Rightarrow$ address

accesses

$$\cos\left(\phi_0 + \sum_{k=1}^{B/3} b_k 2^{-k-1}\right)$$

$$\sin\left(\phi_0 + \sum_{k=1}^{B/3} b_k 2^{-k-1}\right)$$

Lower Half of the 1st Quadrant

- all positive X_k & Y_k
- no need sign extension
- reduce the loads
- high speed

Merging Butterflies

merge m final butterflies

$$\begin{pmatrix} X_k \\ Y_k \end{pmatrix} \longrightarrow \begin{pmatrix} X_{k+m} \\ Y_{k+m} \end{pmatrix} \quad \text{directly}$$

$$X_{k+m} = X_k - Y_k \sum_{i=k}^{k+m-1} r_i \tan 2^{-i}$$

$$Y_{k+m} = Y_k + X_k \sum_{i=k}^{k+m-1} r_i \tan 2^{-i}$$

valid merging $k \geq (B-1)/2$

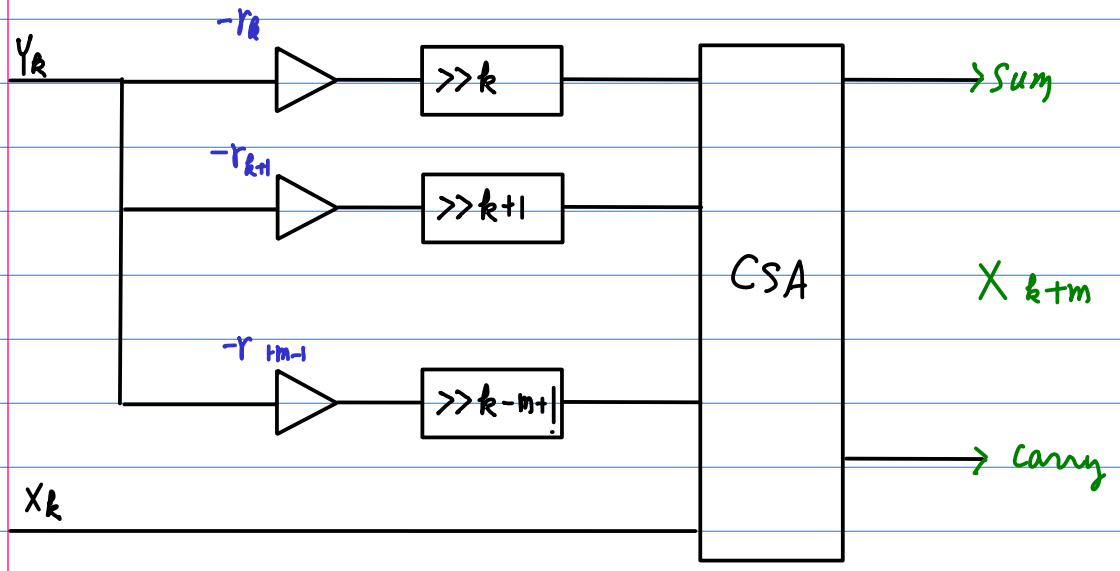
$\tan(2^{-i}) = 2^{-i}$ $k \geq B/3$

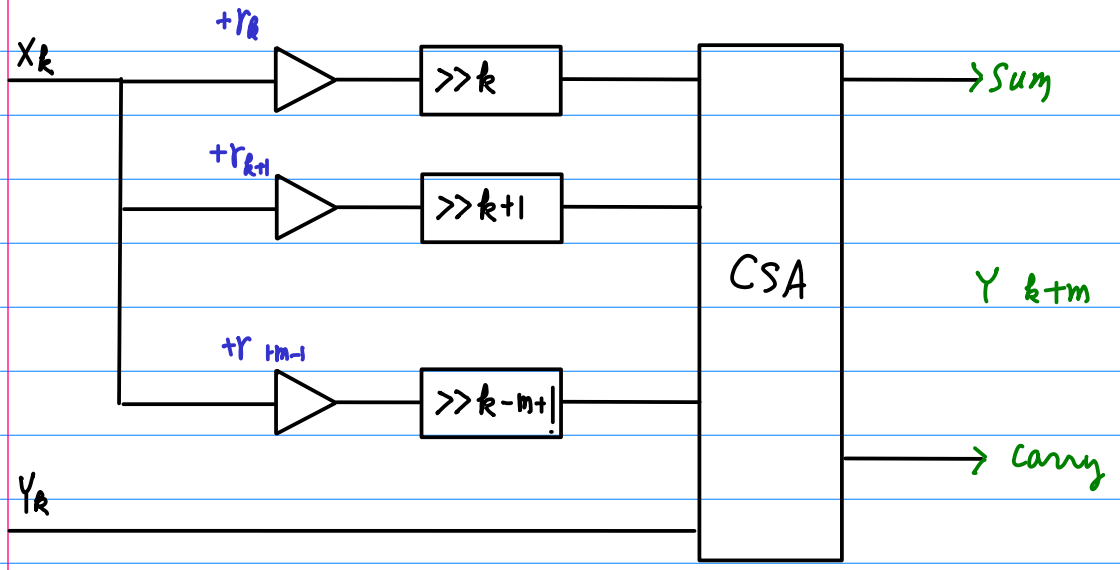
look ahead by m

the individual terms in the summation can be computed independently and summed in parallel

$$X_{k+m} = X_k - Y_k \sum_{i=k}^{k+m-1} r_i \tan 2^{-i}$$

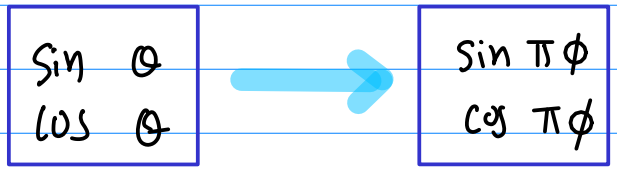
$$Y_{k+m} = Y_k + X_k \sum_{i=k}^{k+m-1} r_i \tan 2^{-i}$$





+ reduced latency
+ reduced routing
+ only the half for a single-ended system.

Output Stage

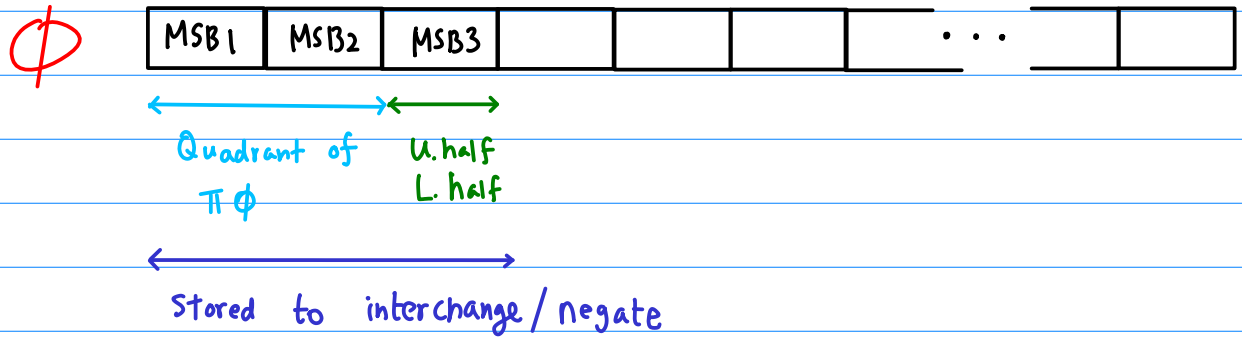


$\theta \in [0, \frac{\pi}{4}]$ → $\phi \in [-1, +1]$

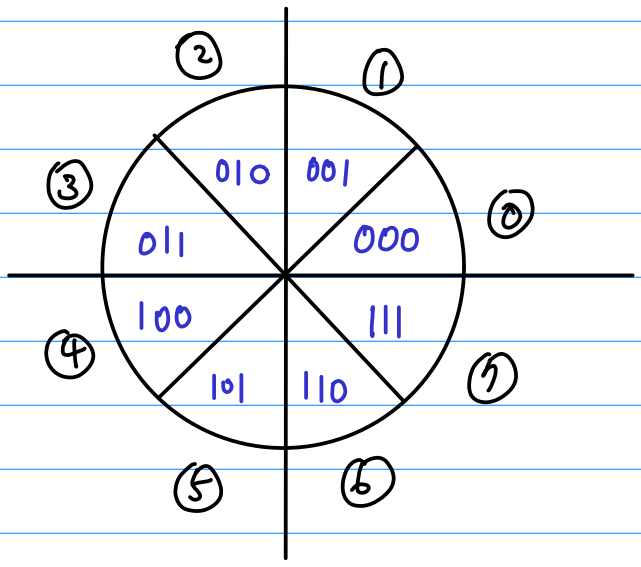
{ negation
  interchange

* negation before interchange

Normalized angle ϕ



MSB of ϕ	ϕ	X_{inv}	Y_{inv}	Swap	$\cos \pi\phi$	$\sin \pi\phi$
0 0 0	⑥	0	0	0	$\cos\theta$	$\sin\theta$
0 0 1	①	0	0	1	$\sin\theta$	$\cos\theta$
0 1 0	②	0	1	1	$-\sin\theta$	$\cos\theta$
0 1 1	③	1	0	0	$-\cos\theta$	$\sin\theta$
1 0 0	④	1	1	0	$-\cos\theta$	$-\sin\theta$
1 0 1	⑤	1	1	1	$-\sin\theta$	$-\cos\theta$
1 1 0	⑥	1	0	1	$\sin\theta$	$-\cos\theta$
1 1 1	⑦	0	1	0	$\cos\theta$	$-\sin\theta$



IC Implementation

clock : 100 MHz

acc : 36-bit (22-bit + 14-bit)

precision : 16-bit

advantage over traditional ROM lookup table approach

accumulator : 36-bit = 22-bit + 14-bit

carry select adder

.

Speed & layout consideration

36-bit output ⟶ truncated to 22-bit

22-bit radian converter

π/4 multiplier

SDFR ≥ 100dBc

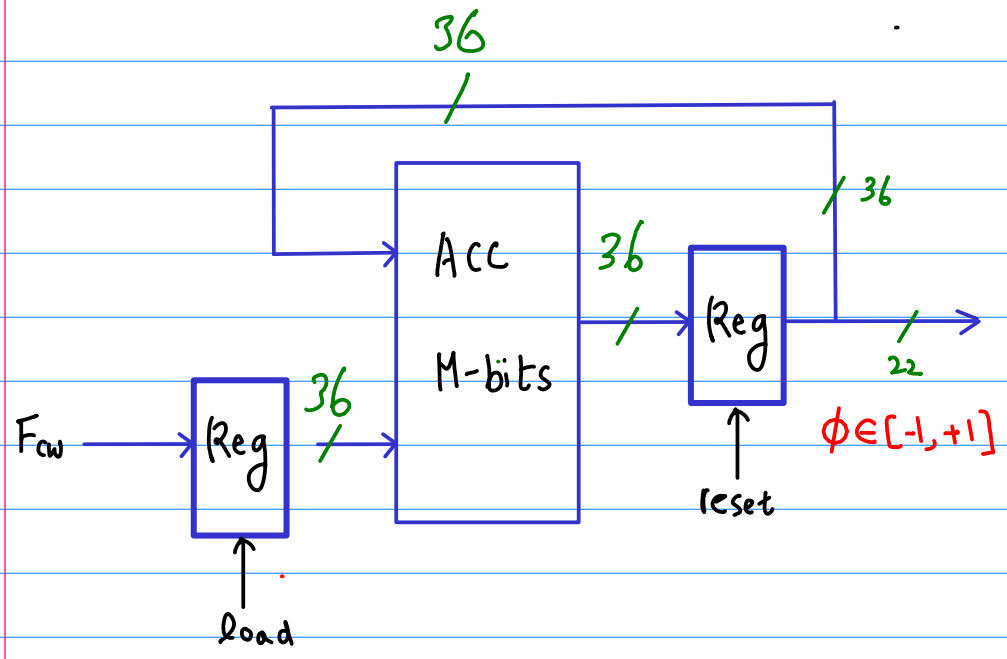
19-bit ROM address

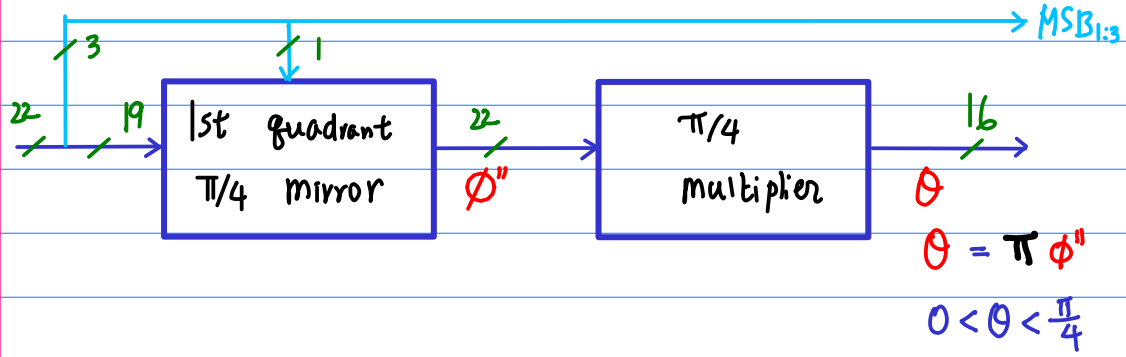
for the similar performance

⇒ 2^{19} words HUGE!

Coarse / fine ROM arch

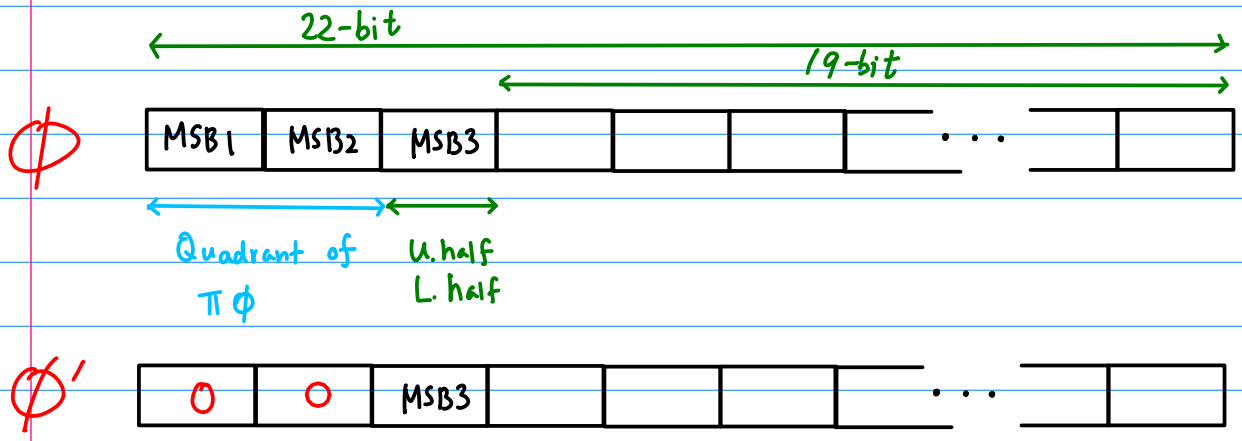
[2] H. T. Nicholas and H. Samueli, "A 150 MHz direct digital frequency synthesizer in 1.25-μm CMOS with −90 dBc spurious performance," *IEEE J. Solid-State Circuits*, vol. 26, pp. 1959–1969, Dec. 1991.





$\theta = \pi \phi''$

$0 < \theta < \frac{\pi}{4}$



$r > \frac{\pi}{4}$: Upper Half $(MSB_3 = 1)$ $\phi'' = \phi'$

$r < \frac{\pi}{4}$: Lower Half $(MSB_3 = 0)$ $\phi'' = 0.5 - \phi'$

radian converter $\theta = \pi \phi''$

all internal angle
~ represented as
fractional binary 2's complement numbers

$\theta = (\pi/4)(4\phi'')$

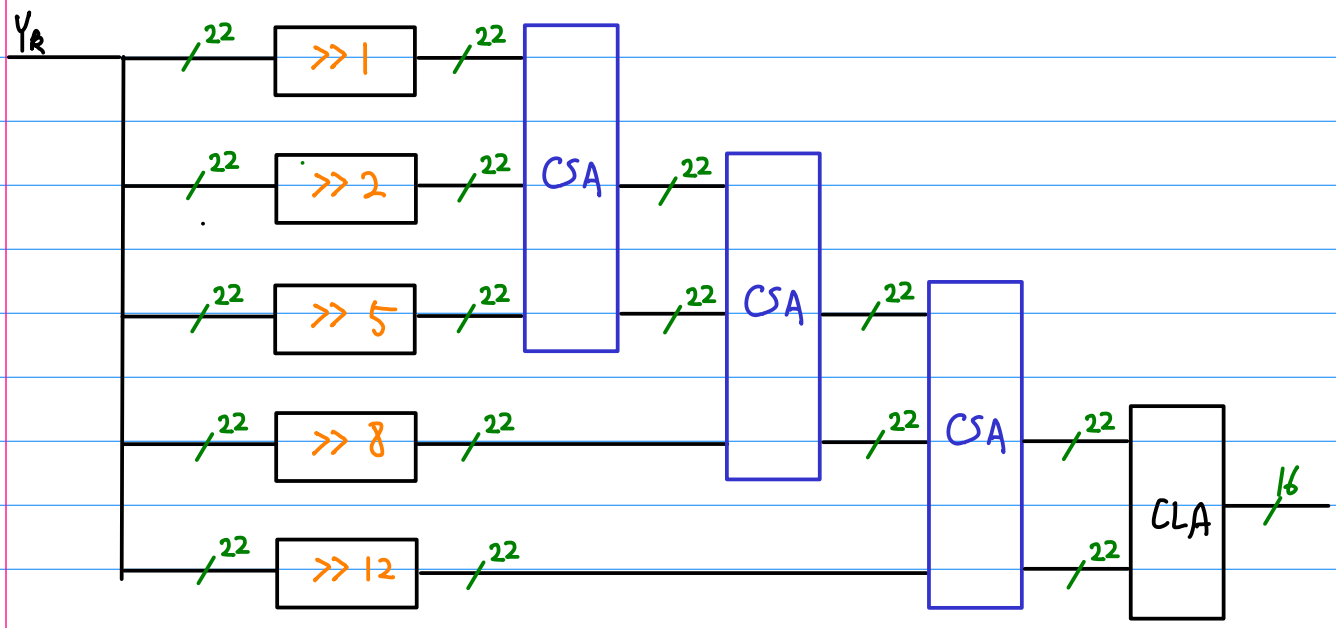
$(\pi/4) = 2^{-1} + 2^{-2} + 2^{-5} + 2^{-8} + 2^{-12}$

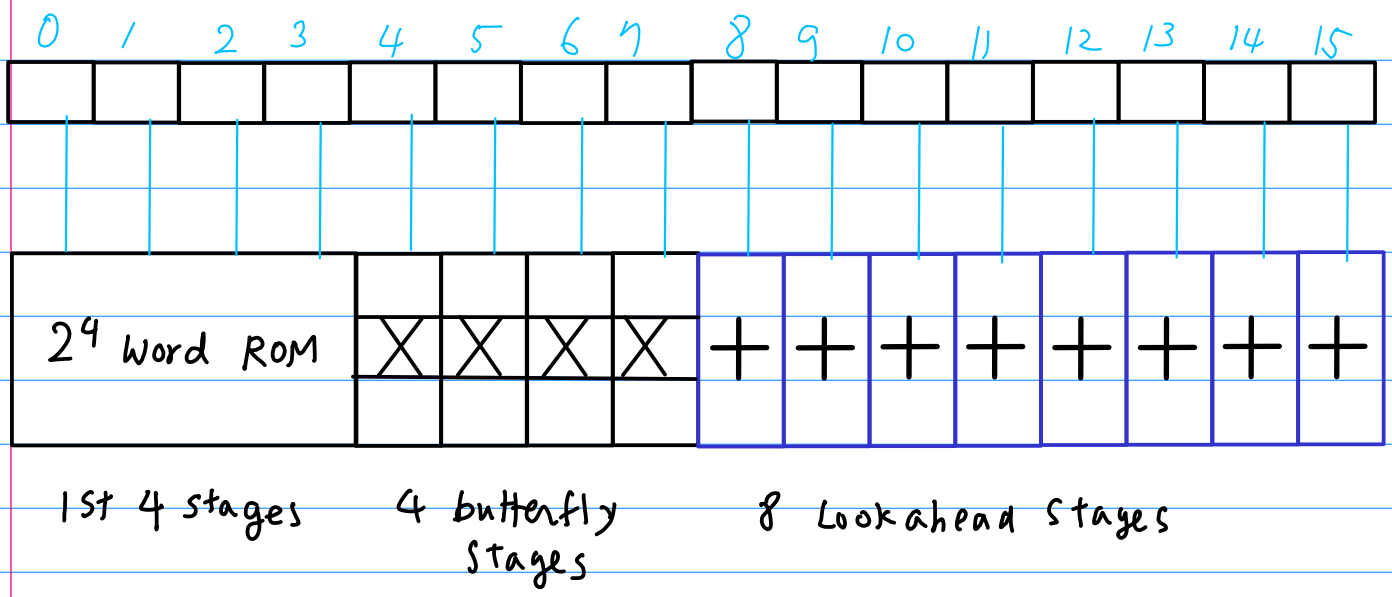
1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	0	1	0	0	1	0	0	0	1

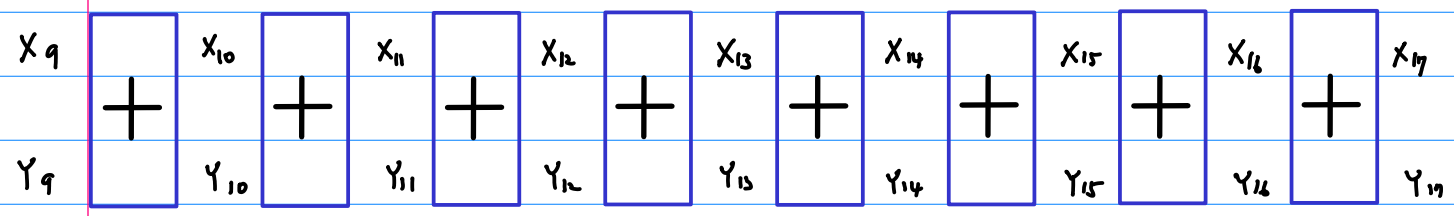
$2^{-1} = 0.5$
$2^{-2} = 0.25$
$2^{-5} = 0.3125$
$2^{-8} = 0.00390625$
$2^{-12} = 0.000244141$

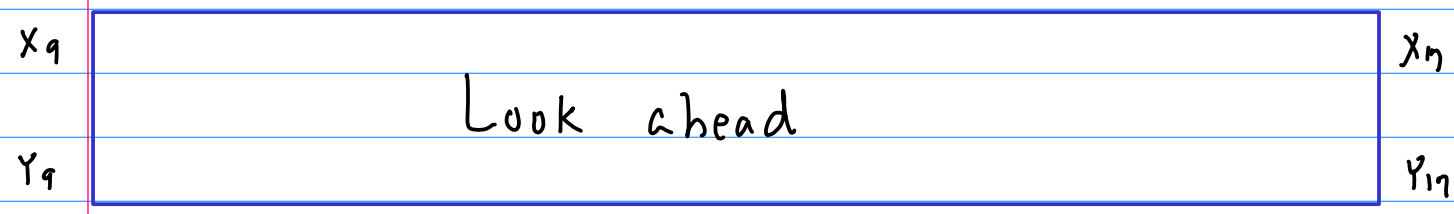
$\pi/4 = 0.78539816\overline{3} = 0.78540039$

only 1st 5 partial products


Y_R
22
>> 1
>> 2
>> 5
>> 8
>> 12
CSA
CSA
CSA
CLA
16


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
2^9 Word ROM
X X X X
+ + + + + + + +
1st 4 stages
4 butterfly stages
8 Lookahead Stages


X_9
X_{10}
X_{11}
X_{12}
X_{13}
X_{14}
X_{15}
X_{16}
X_{17}
+
+
+
+
+
+
+
+
Y_9
Y_{10}
Y_{11}
Y_{12}
Y_{13}
Y_{14}
Y_{15}
Y_{16}
Y_{17}

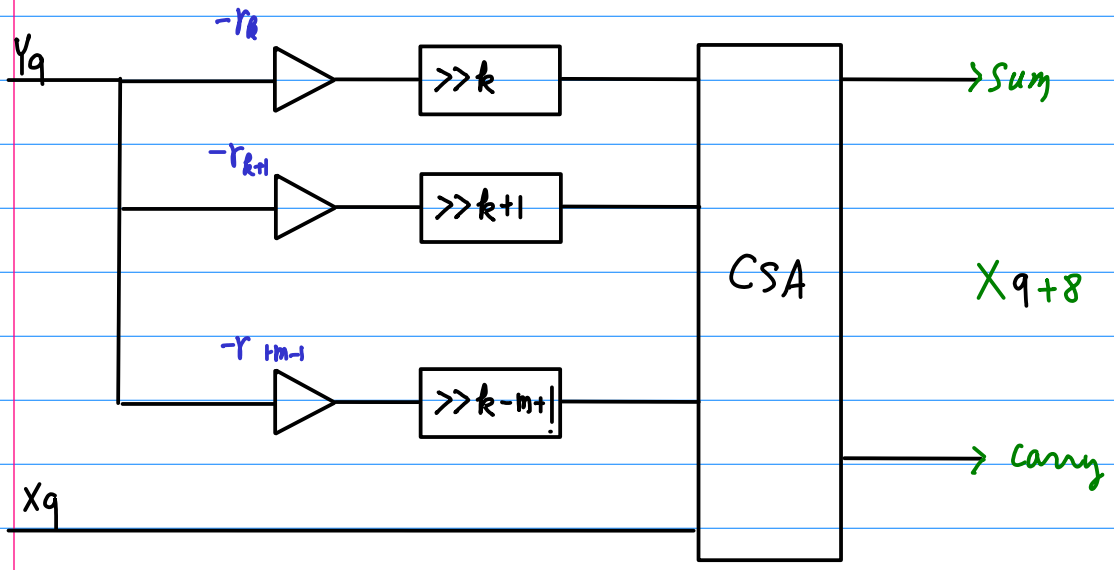

X_9
X_{17}
Look ahead
Y_9
Y_{17}

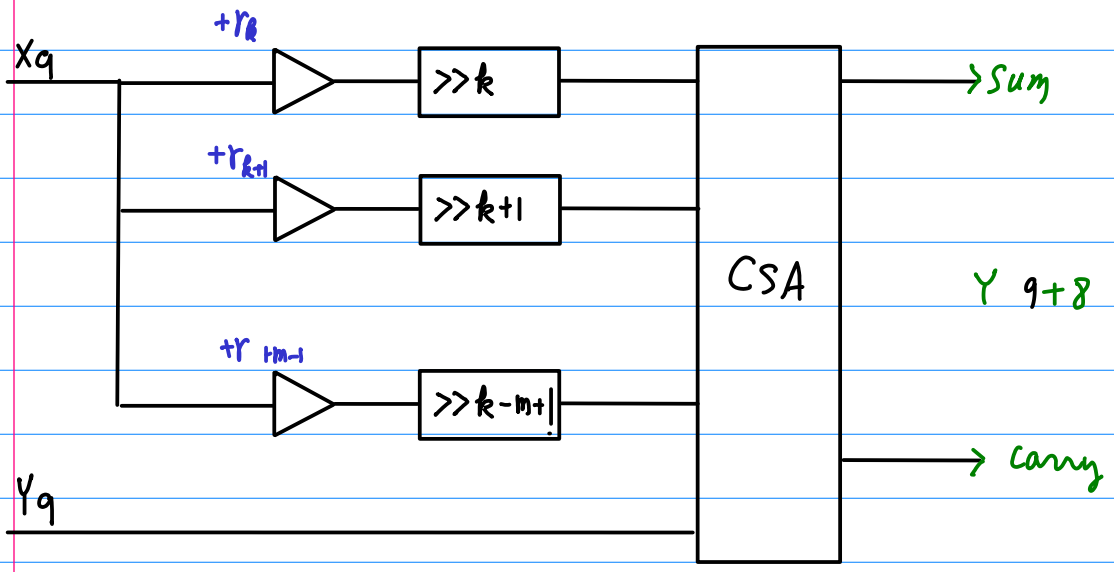
$9 + 8 = 17$

Look ahead

$$X_{k+m} = X_k - Y_k \sum_{i=k}^{k+m-1} r_i \tan 2^{-i}$$

$$Y_{k+m} = Y_k + X_k \sum_{i=k}^{k+m-1} r_i \tan 2^{-i}$$





(k)	θ_k 22-bit binary $\tan\theta_k$ 22-bit binary	θ_k decimal $\tan\theta_k$ decimal

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22		
(1)		1																					0.50000	
		1			0	1	0	1	1	1	1	0	1	1	0	1	0	0	1	1	1	1	0.54630	
(2)			1																				0.25000	
			1						1	0	1	0	1	1	1	1	0	0	0	0	1	0	0.25534	
(3)				1																			0.12500	
				1					0	0	0	1	0	1	0	1	0	1	1	1	0	1	0.12565	
(4)					1																		0.06250	
					1								0	0	1	0	1	0	1	0	1	0	0.06258	
(5)						1																	0.03125	
						1											0	1	0	1	0	1	0.03126	
(6)							1																0.01562	
							1														1	0	0.01562	
(7)								1															0.00781	
								1															0.00781	
(8)									1														0.00390	
									1														0.00390	
(9)										1.													0.00195	
										1													0.00195	

```
>> t'
ans =

    0
    1
    2
    3
    4
    5
    6
    7
    8
    9
   10
   11
   12
   13
   14
   15

>> t'/16
ans =

         0
    0.0625
    0.1250
    0.1875
    0.2500
    0.3125
    0.3750
    0.4375
    0.5000
    0.5625
    0.6250
    0.6875
    0.7500
    0.8125
    0.8750
    0.9375

>> pi/4
ans =  0.78540
>>
```

0000	0111 1111 1110 1010 1010 10
0001	0111 1111 0111 1010 1100 10
0010	0111 1110 0110 1011 1000 10
0011	0111 1100 1110 1101 1110 10
0100	0111 1010 1111 0011 0110 01
0101	0111 1000 0111 1101 1111 01
0110	0111 0101 1001 0000 0001 01
0111	0111 0010 0010 1100 1010 11
1000	0110 1110 0101 0111 0010 00
1001	0110 1010 0001 0011 0100 10
1010	0110 0101 0110 0101 0110 01
1011	0110 0000 0101 0010 0010 01
1100	0101 1010 1101 1110 1001 10

0000	0000 0011 1111 1111 1010 10
0001	0000 1011 1111 1011 0000 00
0010	0001 0011 1110 1010 0101 11
0011	0001 1011 1100 0101 1101 00
0100	0010 0011 1000 0101 0111 11
0101	0010 1011 0010 0001 1010 11
0110	0011 0010 1001 0010 1011 11
0111	0011 1001 1101 0001 0011 11
1000	0100 0000 1101 0101 1111 00
1001	0100 0111 1001 1001 1101 01
1010	0100 1110 0001 0110 0010 01
1011	0101 0100 0100 0100 0110 01
1100	0101 1010 0001 1110 0110 01

```
C = [ "01111111111010101010" ;
    "01111111011110101100010" ;
    "01111100111011011100010" ;
    "01111100111011011111010" ;
    "01111010111110011011001" ;
    "01111000011111011111101" ;
    "01110101100100000000101" ;
    "01110010000101100101011" ;
    "01101110010101011001000" ;
    "01101010000100110100010" ;
    "01100101011001010110011" ;
    "01100000010100100011001" ;
    "01011010110111101001100" ]


S = [ "000000111111111111101010" ;
      "000010111111110110000000" ;
        "000100111111010100101111" ;
        "000110111110001011101000" ;
        "001000111000010101111111" ;
        "001010110010000110101111" ;
        "001100101001001010111111" ;
        "001110011101000100111111" ;
        "010000001101010111110000" ;
        "010001111001100111010111" ;
        "010011100001011000100111" ;
        "010101000100010000110011" ;
        "010110100001111001100111" ]


CV = zeros(rows(C), 1);
Cn = 2^rows(C') / 2;
for i=1:rows(C)
  CV(i) = bin2dec(C(i, :)) / Cn;
end

SV = zeros(rows(S), 1);
Sn = 2^rows(S') / 2;
for i=1:rows(S)
  SV(i) = bin2dec(S(i, :)) / Sn;
end
```

```
>> subplot(2,1,1)
>> axis([0, 2*pi 0 1])
>> plot(x, CV)
>> axis([0, 2*pi 0 1])
>> subplot(2,1,2)
>> plot(x, SV)
>> axis([0, 2*pi 0 1])
>>
```