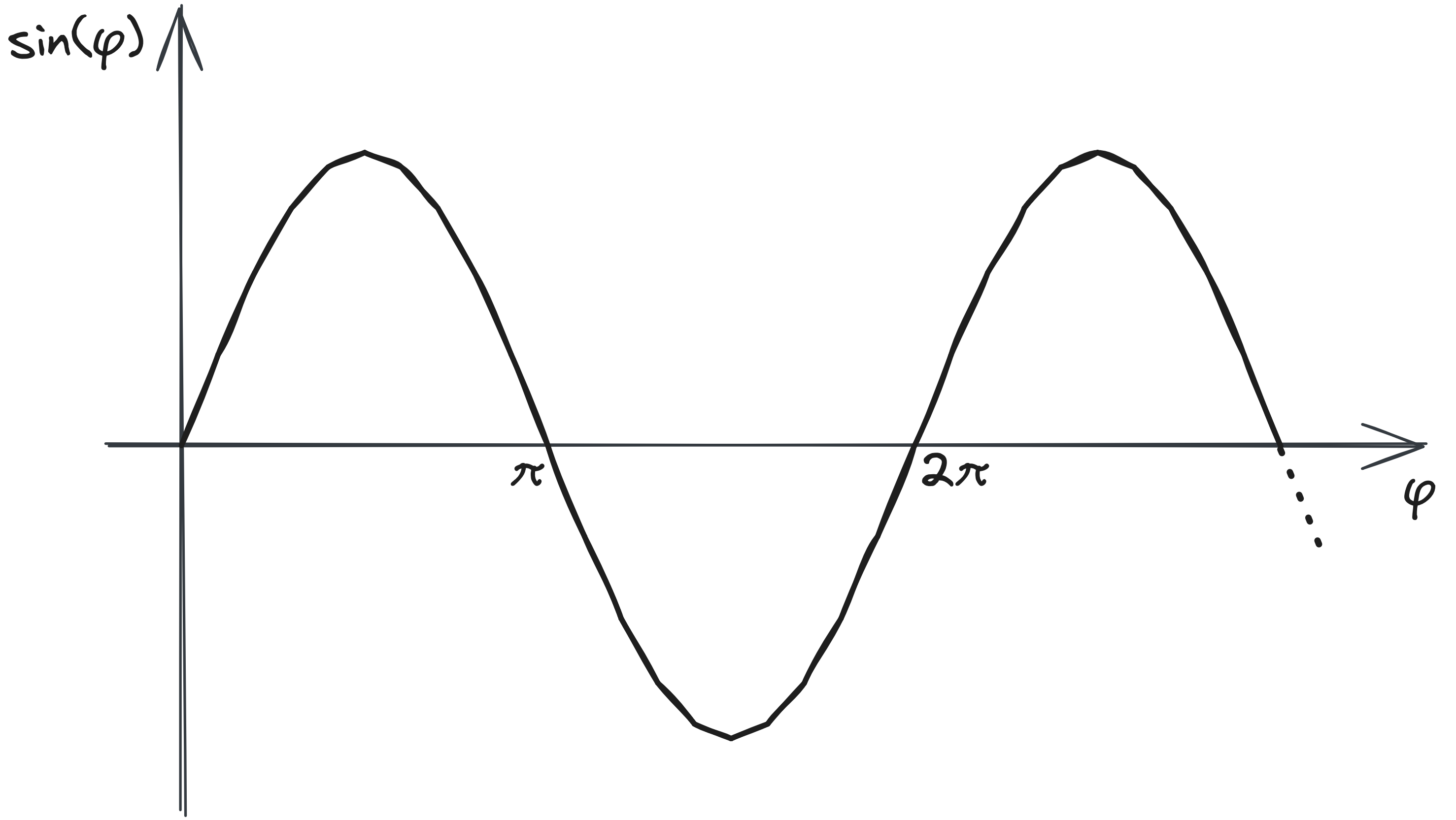
Numerically Controlled Oscillators (NCO)

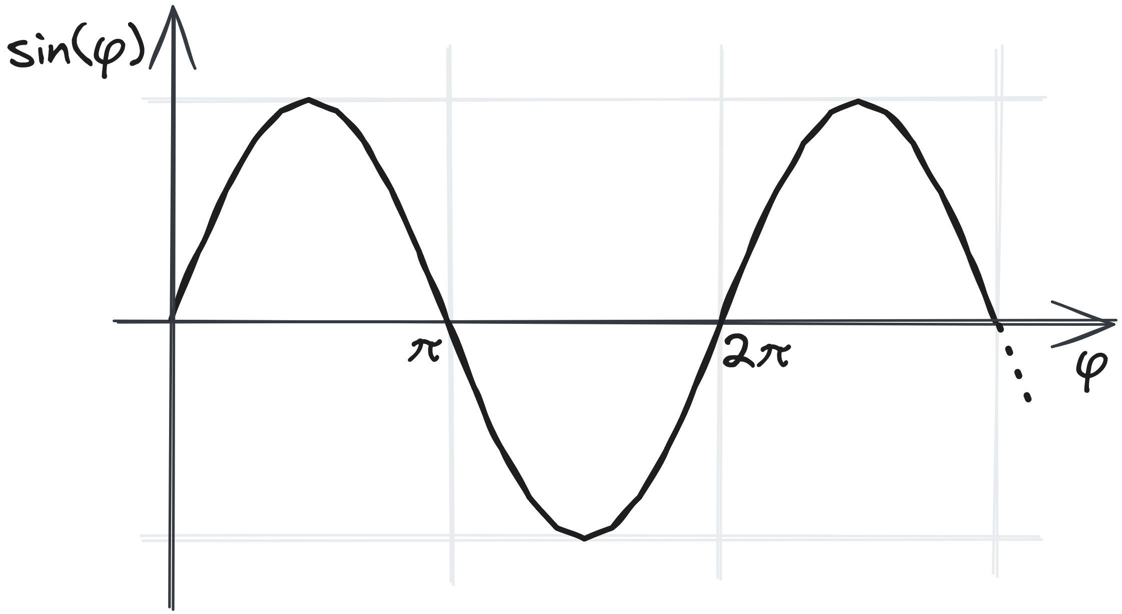
Shadi Youssef
@Radiohub

How can we generate a sinusoidal signal in the digital domain?



We can record the waveform in memory

To generate the sine wave, we simply playback the recording



φ	$\sin(\varphi)$
0	$\sin(0)$
φ_0	$\sin(\varphi_0)$
$2\varphi_0$ $\vdots$ $2\pi-\varphi_0$	$\sin(2\varphi_0)$ $\vdots$ $\sin(2\pi-\varphi_0)$
2π	$\sin(0)$
$2\pi+\varphi_0$ $\vdots$	$\sin(\varphi_0)$ $\vdots$

For now, assume recording has infinite resolution

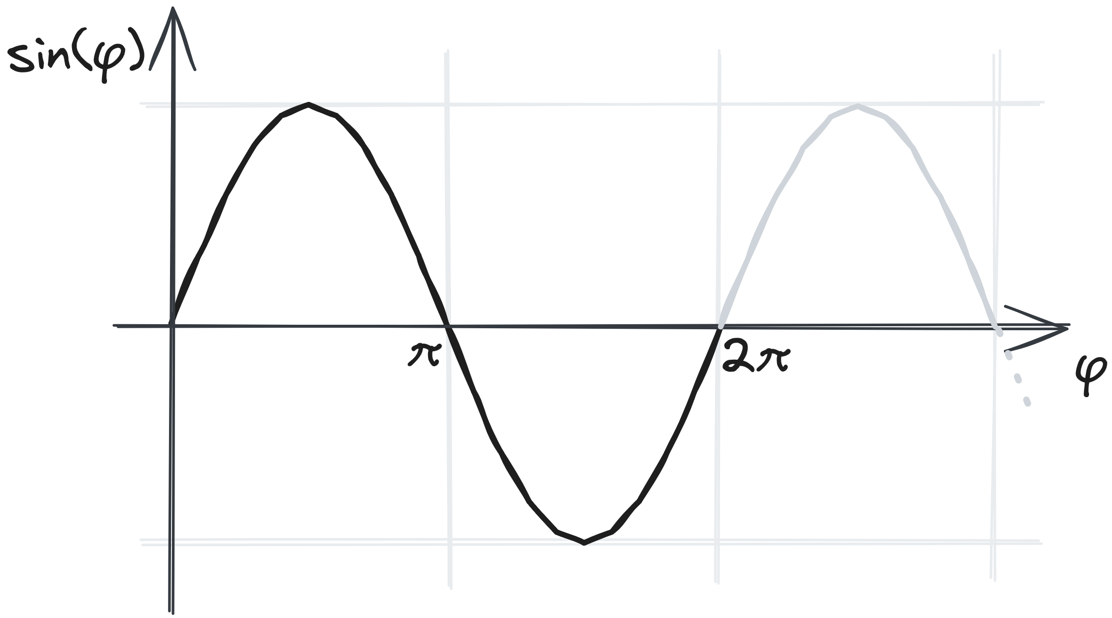
This means, φ_0 -> 0 and $\sin(\varphi)$ word length -> ∞

This requires infinite memory (more on this later)

A sine wave repeats every 2π

So we record one period only and play it back on repeat

This saves memory



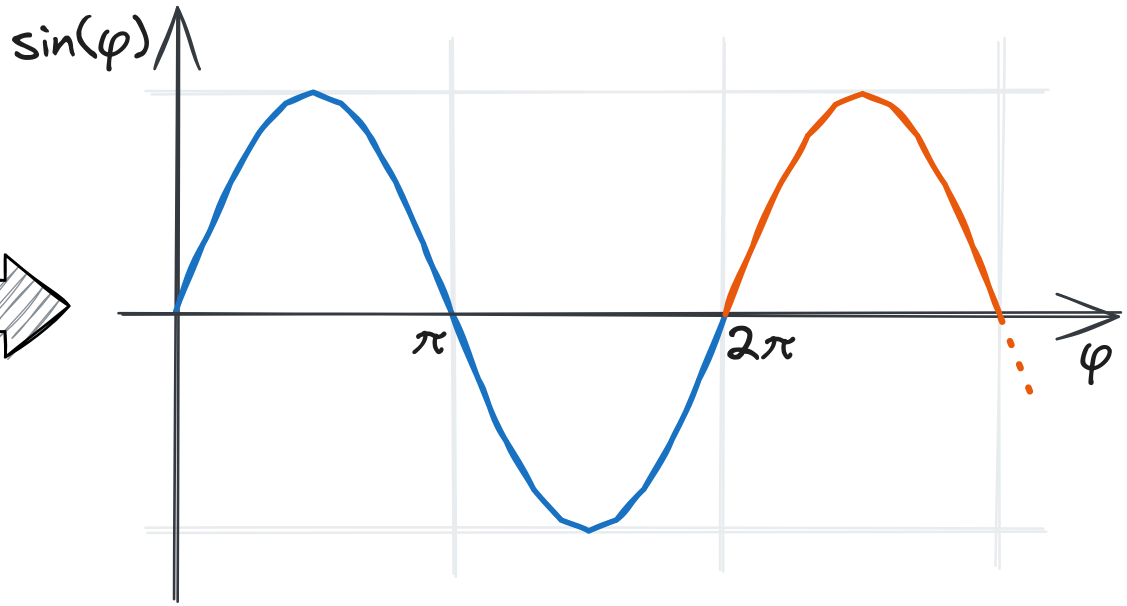
φ	$\sin(\varphi)$
0	$\sin(0)$
φ_0	$\sin(\varphi_0)$
$2\varphi_0$	$\sin(2\varphi_0)$
$\vdots$	$\vdots$
$2\pi-\varphi_0$	$\sin(2\pi-\varphi_0)$
2π	$\sin(0)$
$2\pi+\varphi_0$	$\sin(\varphi_0)$
$\vdots$	$\vdots$

We can save more memory by recording 1/4 period only and using a slightly more complicated playback scheme

By cycling through the memory entries, we playback the sine wave

Playback speed = 1 sample per clock cycle 1/fclk

φ	$\sin(\varphi)$
0	$\sin(0)$
φ_0	$\sin(\varphi_0)$
$2\varphi_0$	$\sin(2\varphi_0)$
$\vdots$	$\vdots$
$2\pi-\varphi_0$	$\sin(2\pi-\varphi_0)$



We don't need φ to read out the corresponding $\sin(\varphi)$ value

Instead we assign an index to each entry

We use the index to look-up values as needed during playback

We don't need this column

φ	$\sin(\varphi)$
0	$\sin(0)$
φ_0	$\sin(\varphi_0)$
$2\varphi_0$ $\vdots$ $2\pi-\varphi_0$	$\sin(2\varphi_0)$ $\vdots$ $\sin(2\pi-\varphi_0)$

We add this column instead

index	$\sin(\varphi)$
0	$\sin(0)$
1	$\sin(\varphi_0)$
2 $\vdots$ 2^M-1	$\sin(2\varphi_0)$ $\vdots$ $\sin(2\pi-\varphi_0)$

Look-up table (LUT)

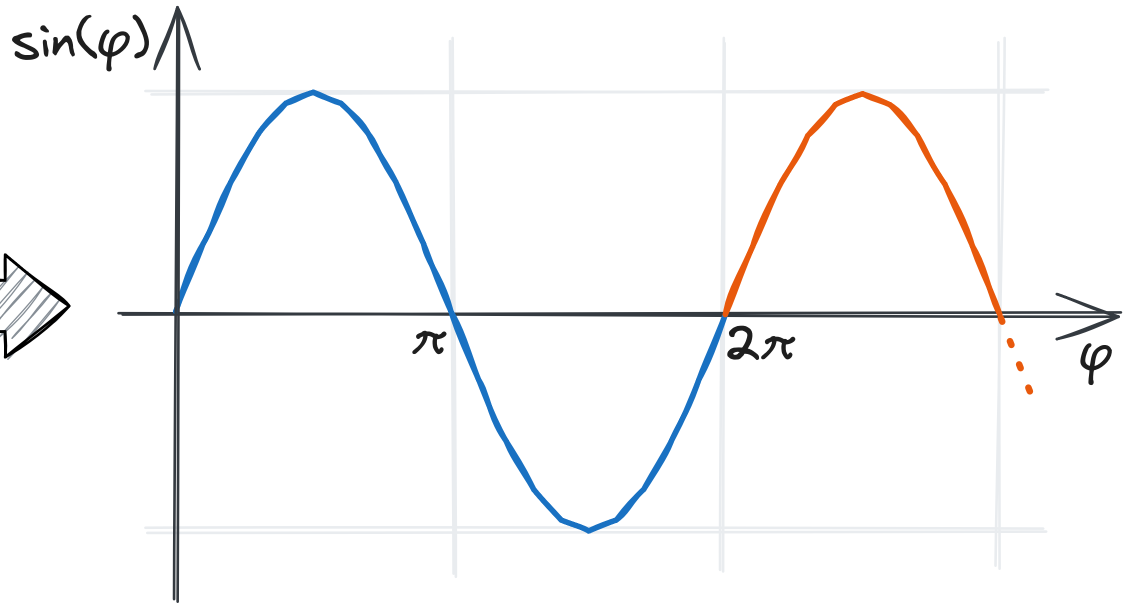
This means there are 2^M points in a 2π period, so phase step $\varphi_0 = 2\pi/2^M$

We are still assuming infinite resolution, so $\varphi_0 \to 0$ means $M \to \infty$

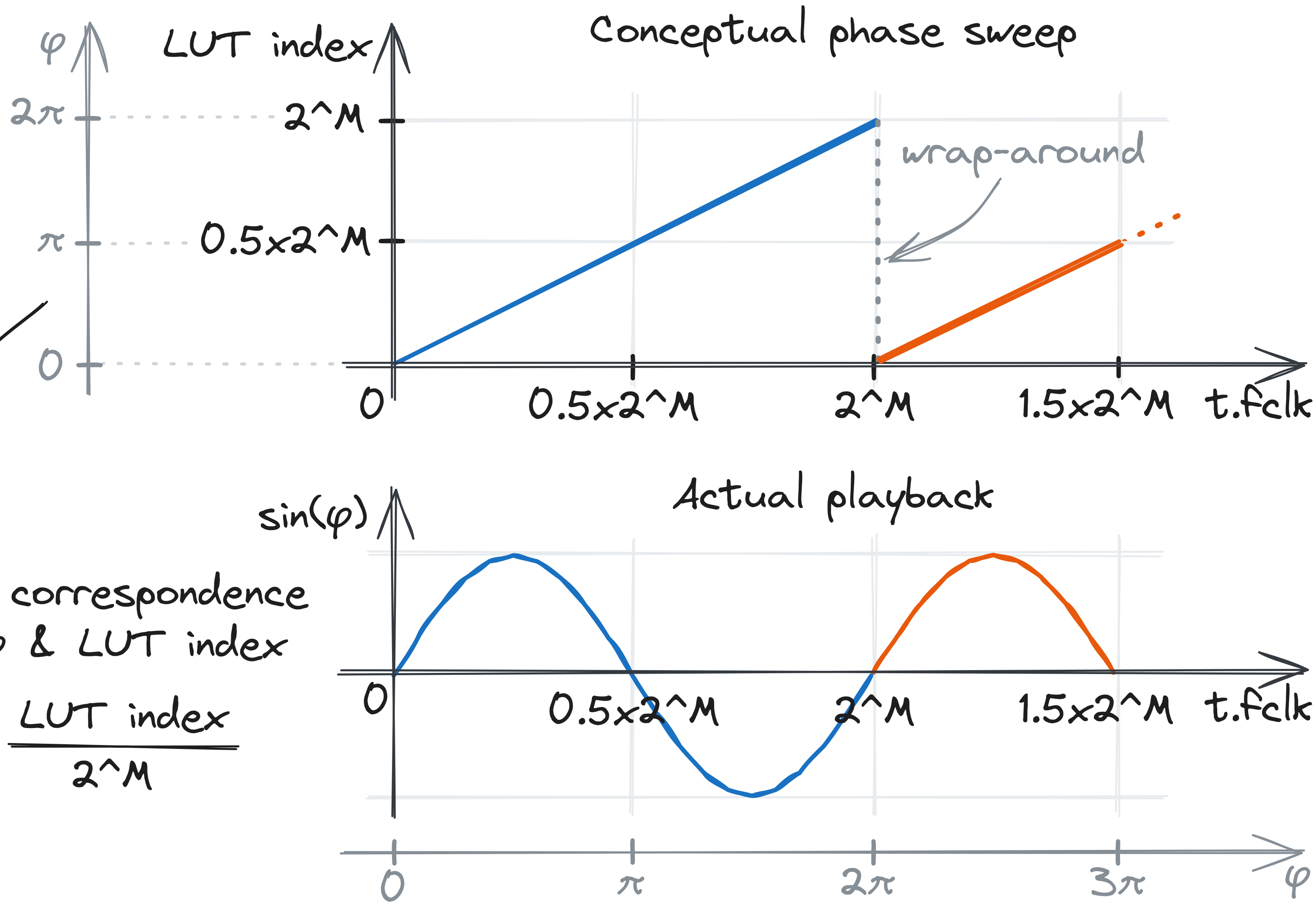
We now cycle through the LUT index instead to playback the sine wave

Playback speed still = 1 sample per clock cycle 1/fclk

index	$\sin(\varphi)$
0	$\sin(0)$
1	$\sin(\varphi_0)$
2	$\sin(2\varphi_0)$
⋮	⋮
2^M-1	$\sin(2\pi-\varphi_0)$



We can see that sweeping LUT index = sweeping sine wave phase



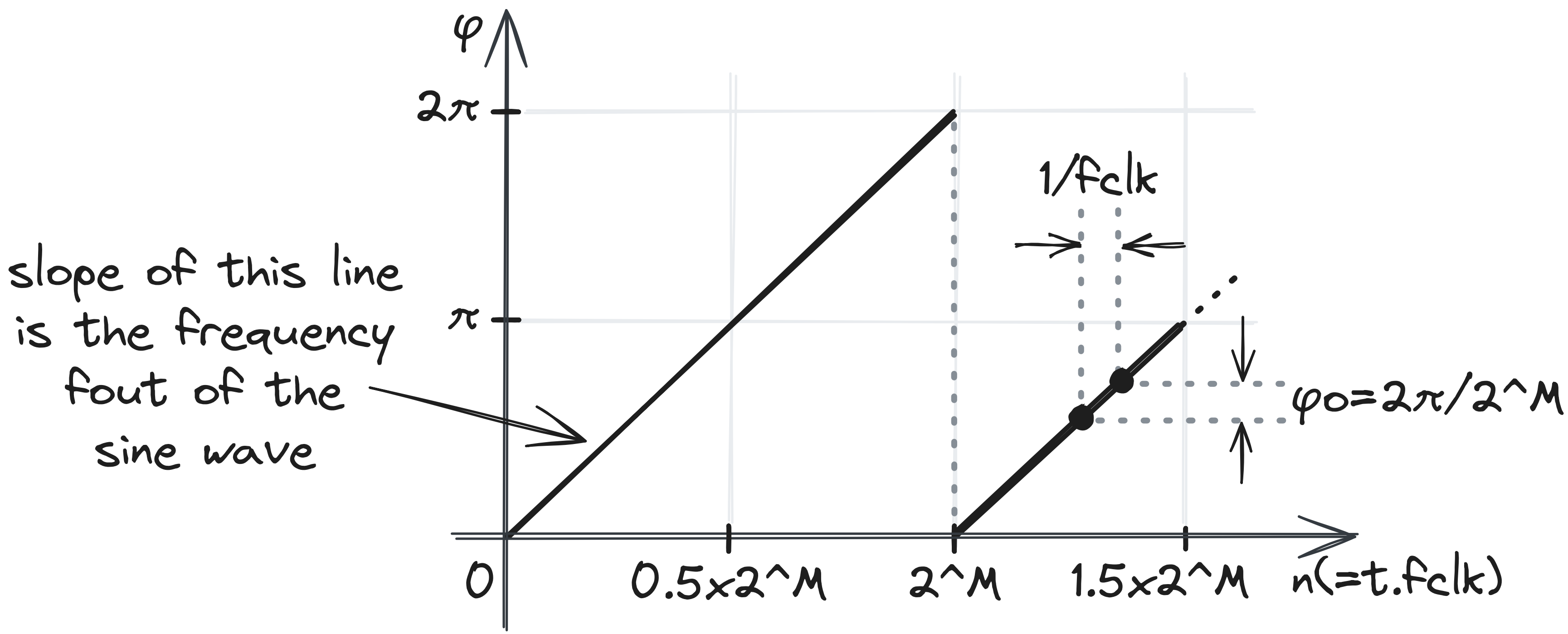
one-to-one correspondence between φ & LUT index

$$\frac{\varphi}{2\pi} = \frac{\text{LUT index}}{2^M}$$

Frequency = rate of change of phase

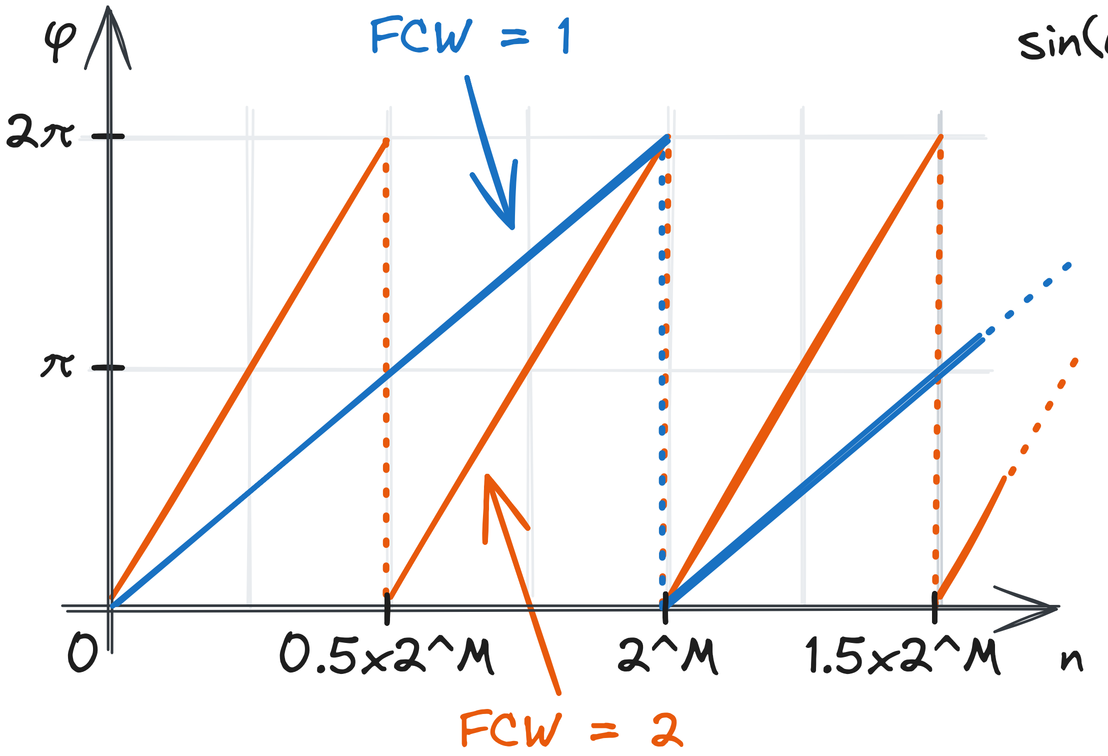
Output sine wave accumulates phase at a rate of φ_0 per clock cycle

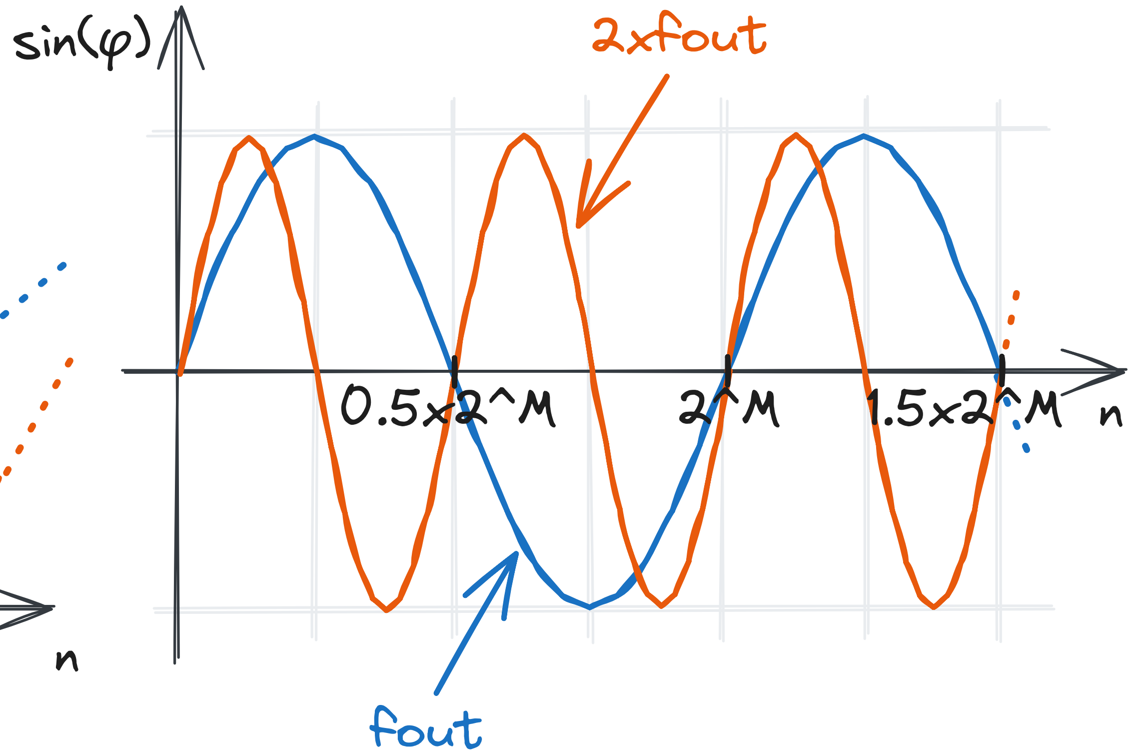
$$f_{out} = \frac{\varphi_0}{2\pi} f_{clk} = \frac{2\pi/2^M}{2\pi} f_{clk} = \frac{f_{clk}}{2^M}$$



To make frequency programmable, we simply skip LUT entries during playback
A frequency control word (FCW) sets the phase step per clock cycle
Now phase accumulates at a rate of FCW x φo per clock cycle

$$\boxed{f_{out} = \frac{FCW}{2^M} f_{clk}}$$





Summing up what we did so far

We started with a continuous-time sine wave

$$x(t) = \sin(\varphi(t))$$

whose phase accumulates linearly with time at a rate of fout

$$x(t) = \sin(2\pi \times fout \times t)$$

We generated a digital replica of the signal sampled at t = n/fclk intervals

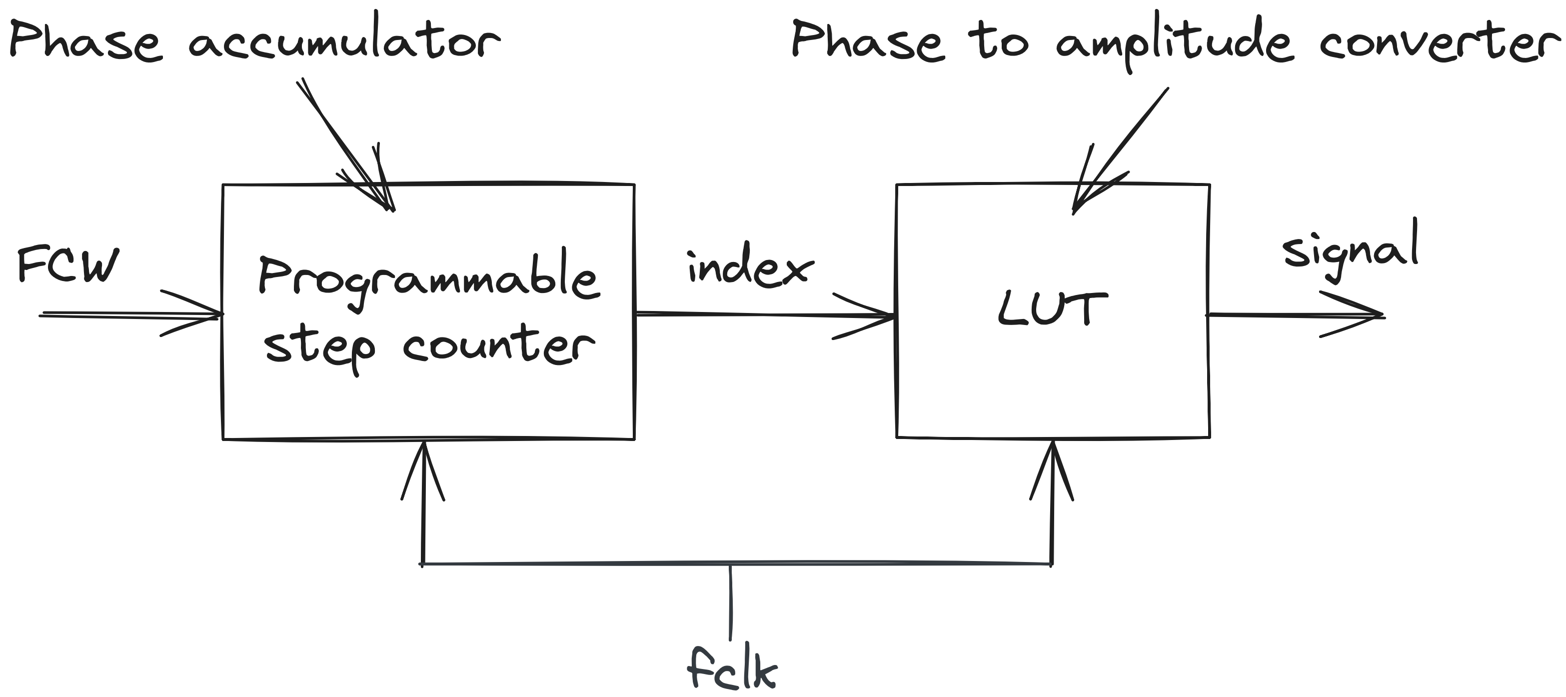
$$x[n] = \sin(2\pi \times fout/fclk \times n)$$

and we made its frequency programmable by introducing a control word FCW

$$x[n] = \sin(2\pi \times FCW/2^M \times n)$$

Putting it all together

We have a numerically controlled oscillator (NCO)

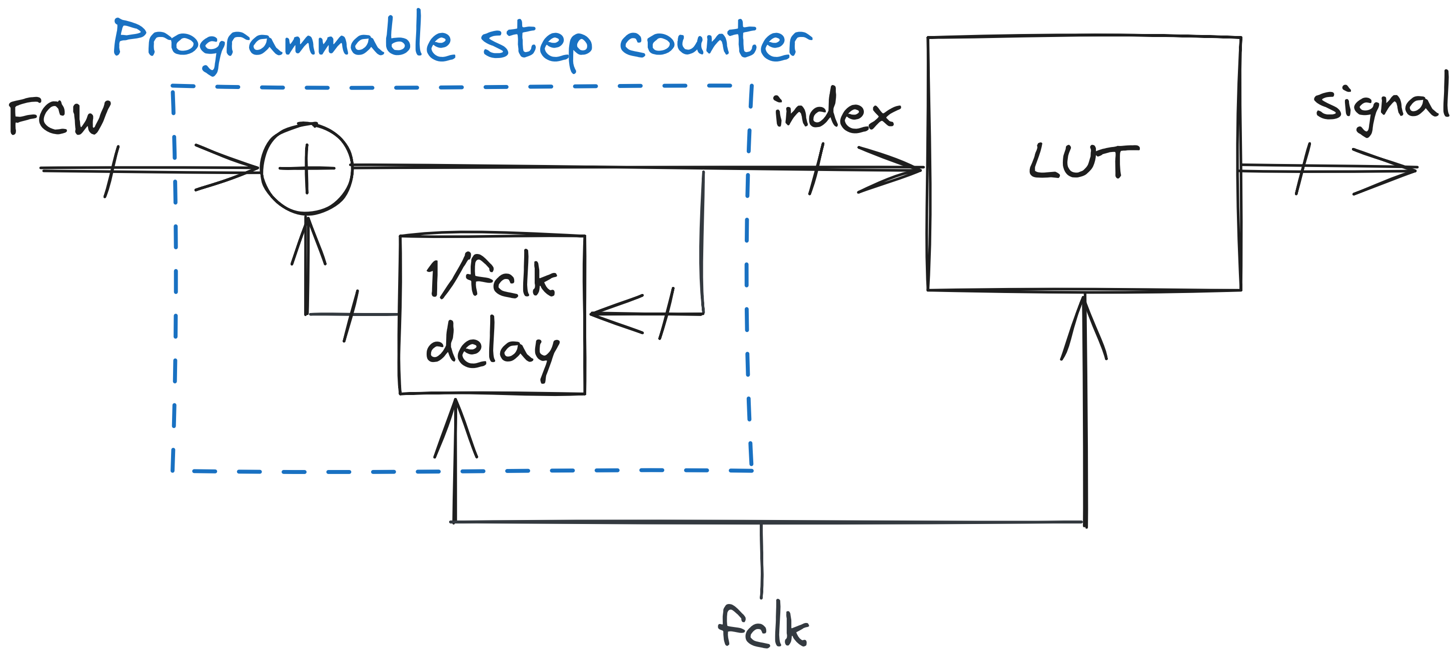


A programmable step counter is a simple accumulator

index[n] = index[n-1] + FCW

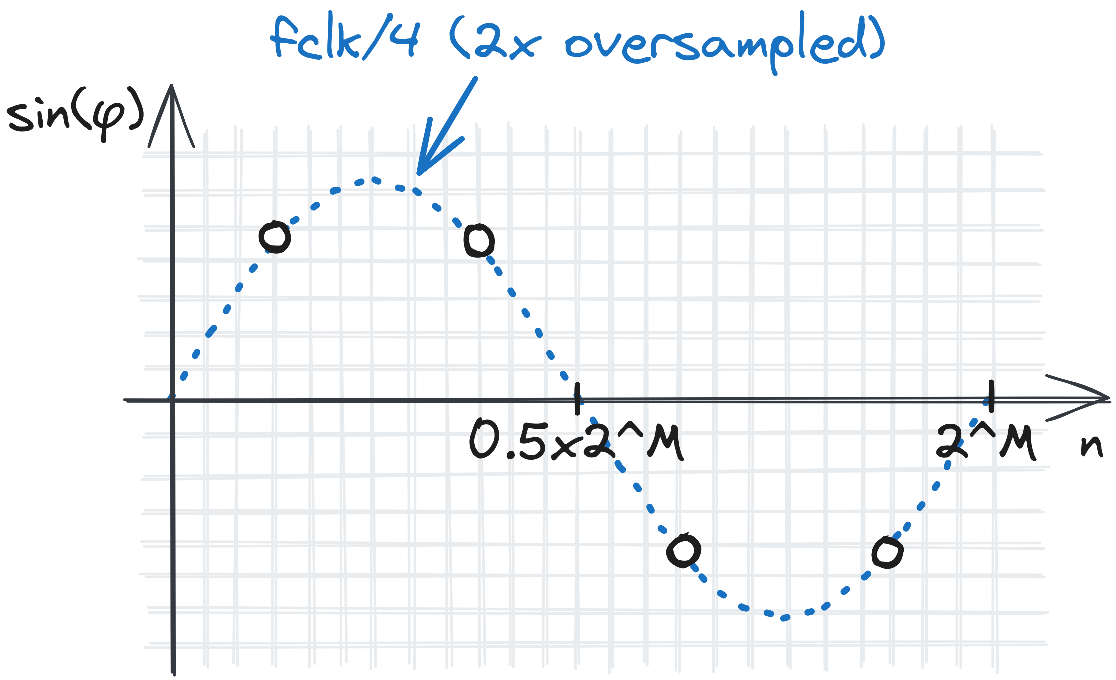
which is equivalent to

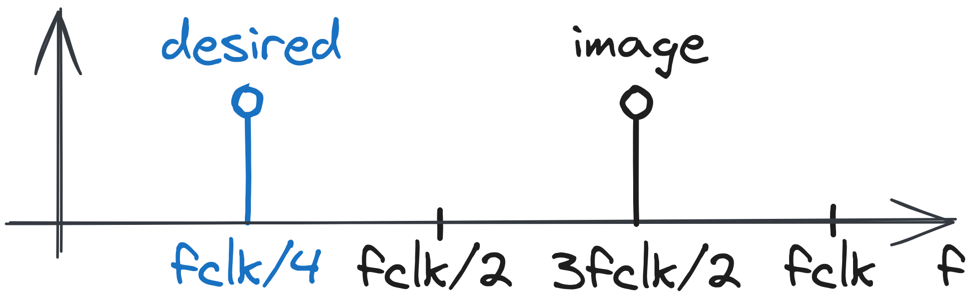
$\varphi[n] = \varphi[n-1] + FCW \times \varphi_0$



Frequency Range

We are generating a sampled version of a sine wave, so Nyquist applies

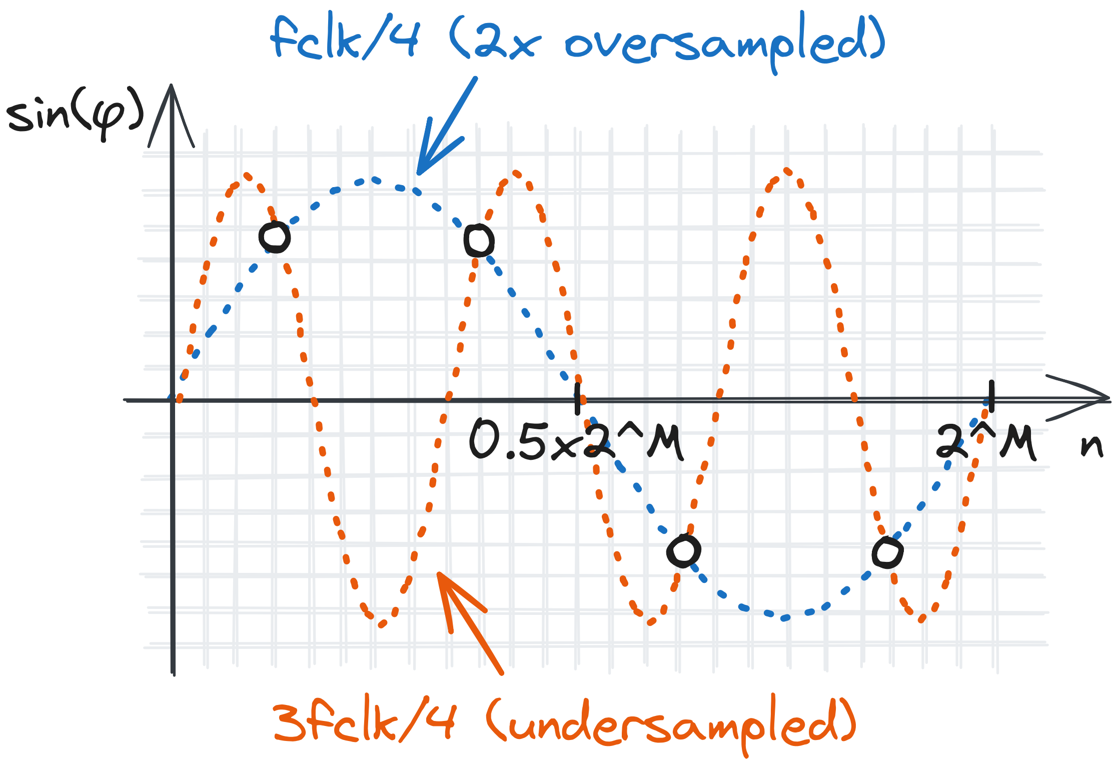


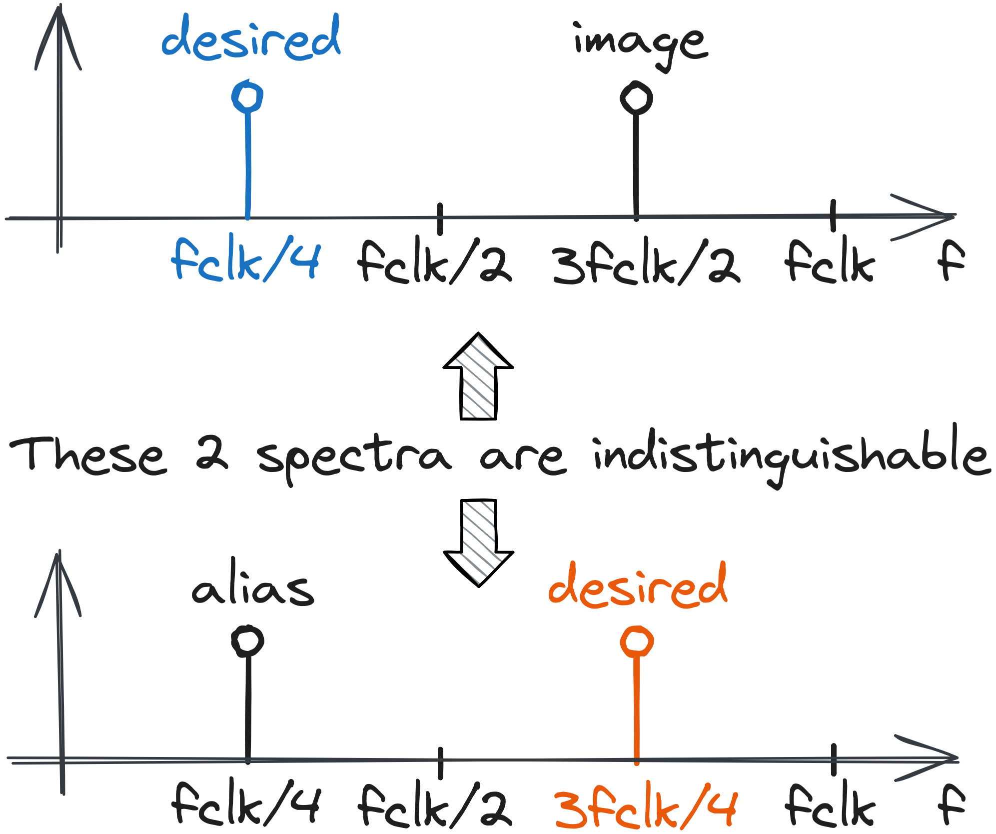


Frequency Range

The NCO can generate unambiguous frequencies from DC to fclk/2

$\boxed{f_{out} < f_{clk}/2}$





Frequency Range

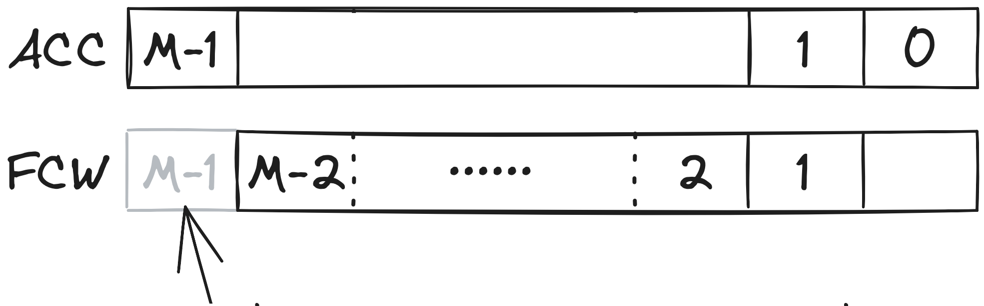
$$f_{out}/f_{clk} = FCW/2^M$$

No alias condition

$$f_{out}/f_{clk} < 1/2$$

which translates to

$$FCW < 2^{(M-1)}$$



For an M-bit accumulator design, FCW should be (M-1) bits long
Alternatively, FCW is M-bits long, but MSB is never set to 1

Frequency Range

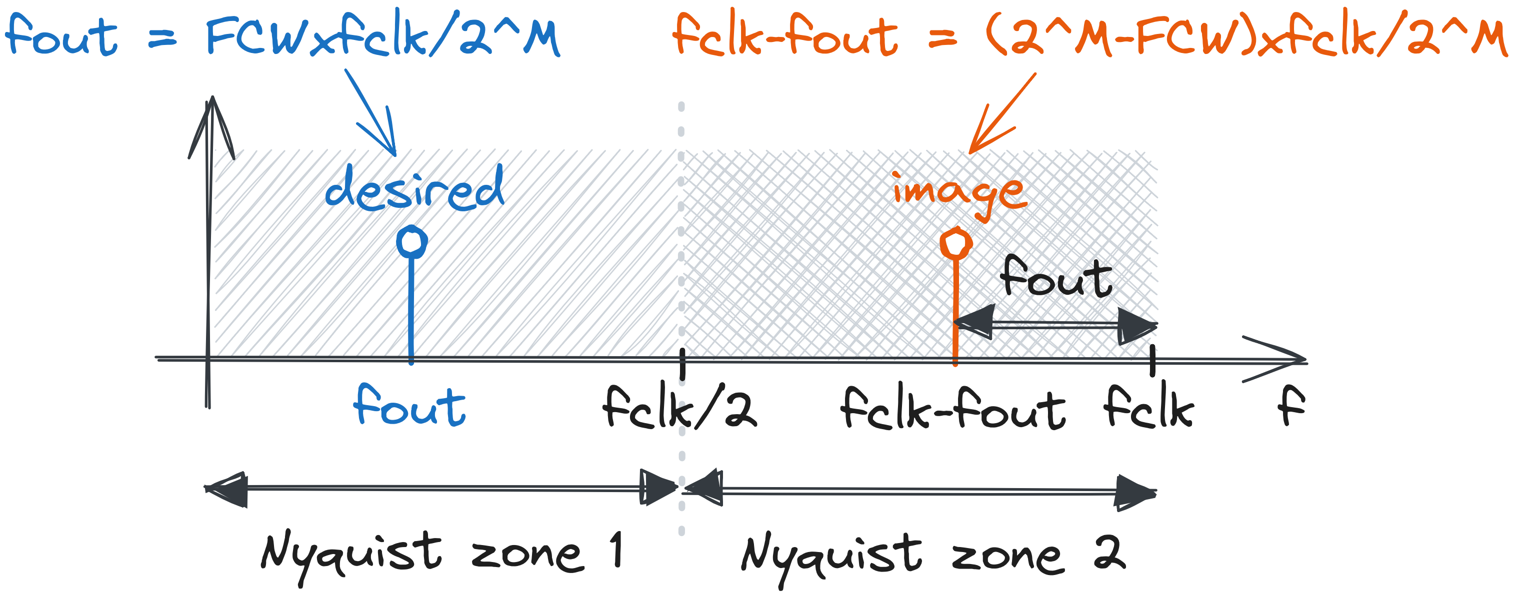
Frequencies corresponding to FCW and 2^M-FCW form signal-image pairs across 1st and 2nd Nyquist zones

Example:

M = 16 & fclk = 100MHz

For FCW = 2^13 -> fout = 12.5MHz

For FCW = 2^16-2^13 -> fout = 87.5MHz (=100-12.5MHz)

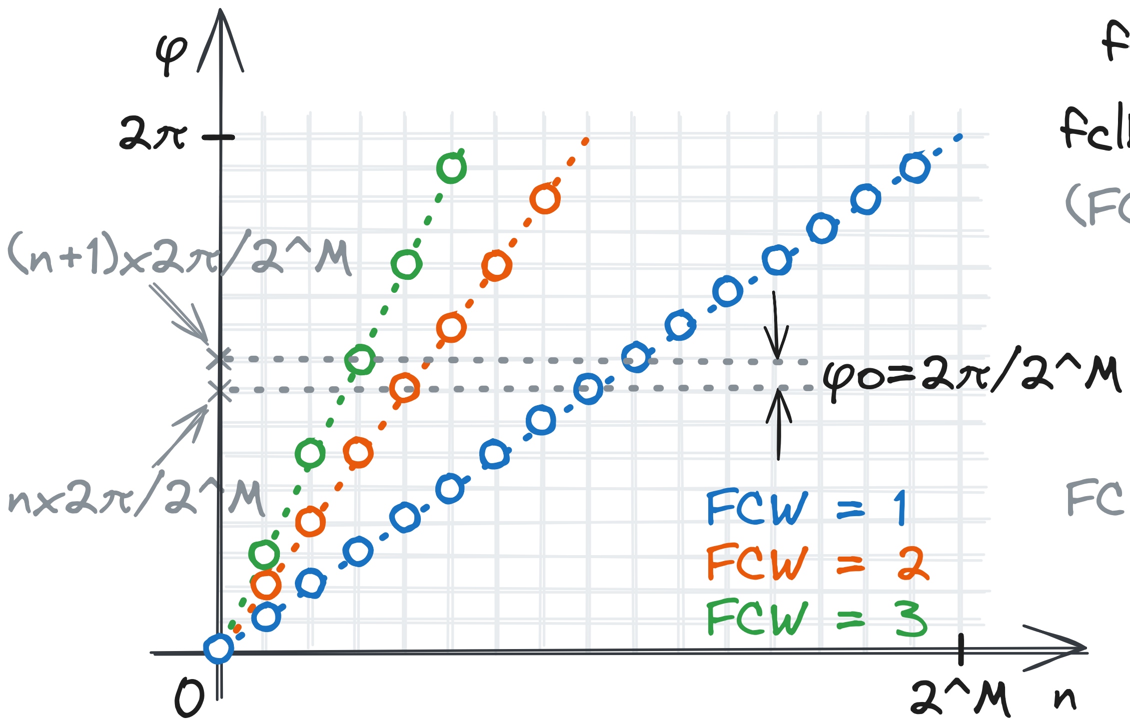


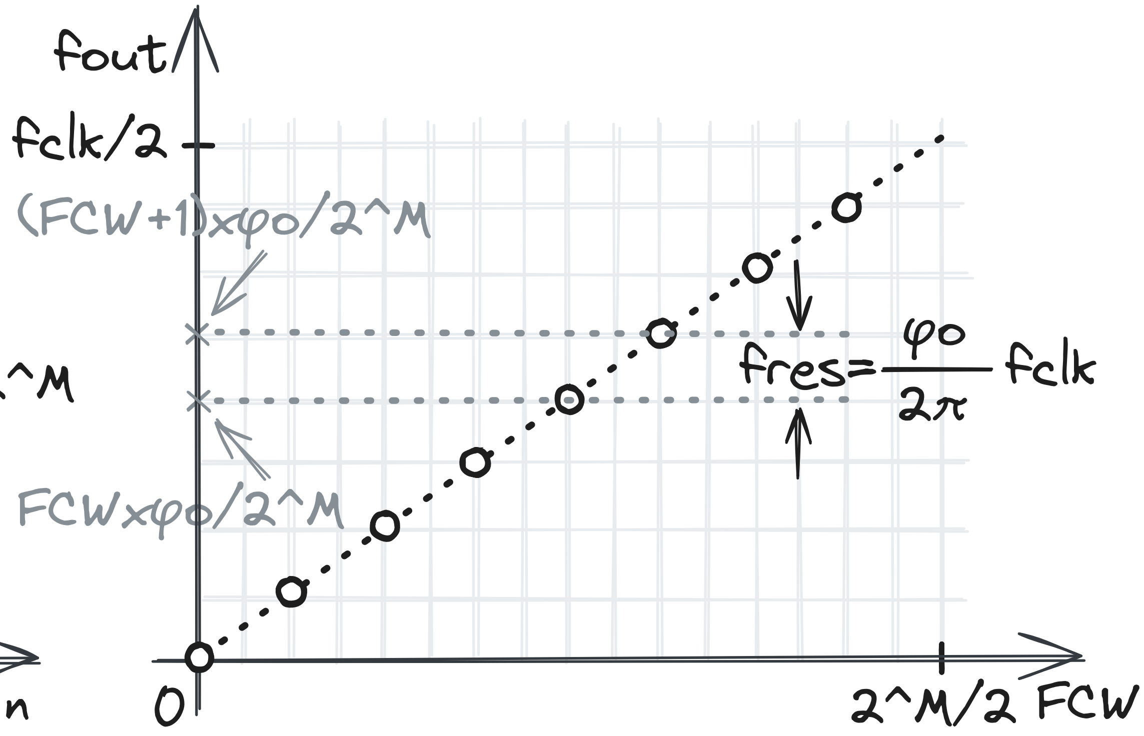
Frequency Resolution

Frequency resolution = min. frequency step possible

Min. frequency step possible corresponds to min. FCW increment, which is 1

$$\boxed{f_{res} = \frac{f_{clk}}{2^M}}$$





Frequency Resolution

How accurate is the output frequency relative to the clock frequency?

Frequency accuracy = fout/fclk = 1/2^M

We can achieve very high accuracy by increasing index word length

For Example

M = 32 -> NCO accuracy = 0.2 ppb

For comparison

Typical XTAL accuracy is 20 ppm

NCO is 10^5 better!

This means accuracy is determined by accuracy of fclk XTAL and not the NCO

Frequency Resolution

Finite frequency precision results in absolute frequency error < fres/2

Frequency error causes phase drift between ideal and actual sinusoid

Example:

Requirements: M = 32, fclk = 100MHz & fout = 13.56MHz

fres = fclk/2^M = 23.283mHz

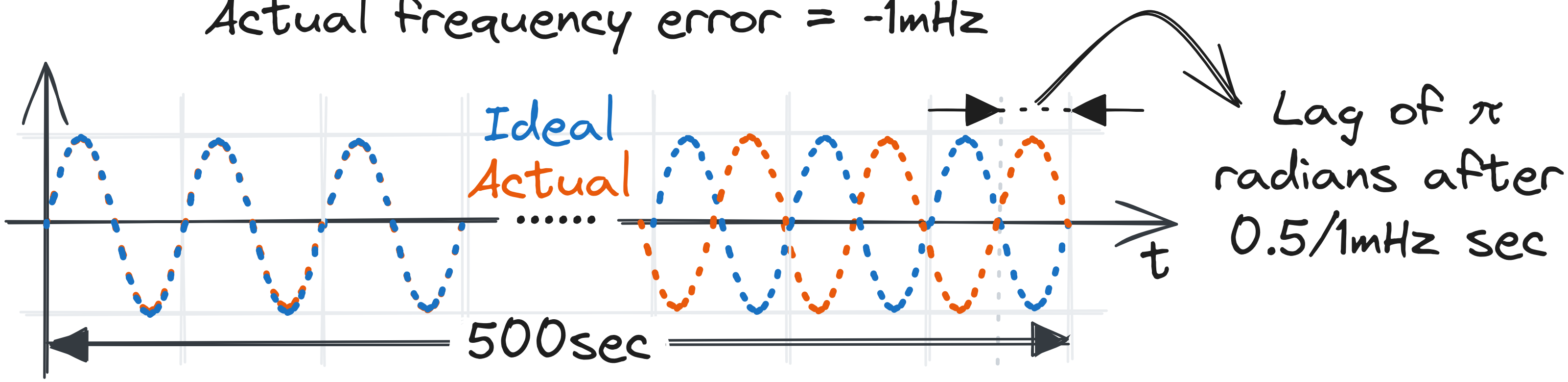
Max absolute frequency error = fres/2 = 11.641mHz

Ideal FCW = 2^Mxfout/fclk = 582,397,565.3376

Actual FCW (must be integer) = 582,397,565

Actual fout = 13.559MHz

Actual frequency error = -1mHz



Phase Truncation

High frequency resolution comes at the expense of large LUT size

Example:

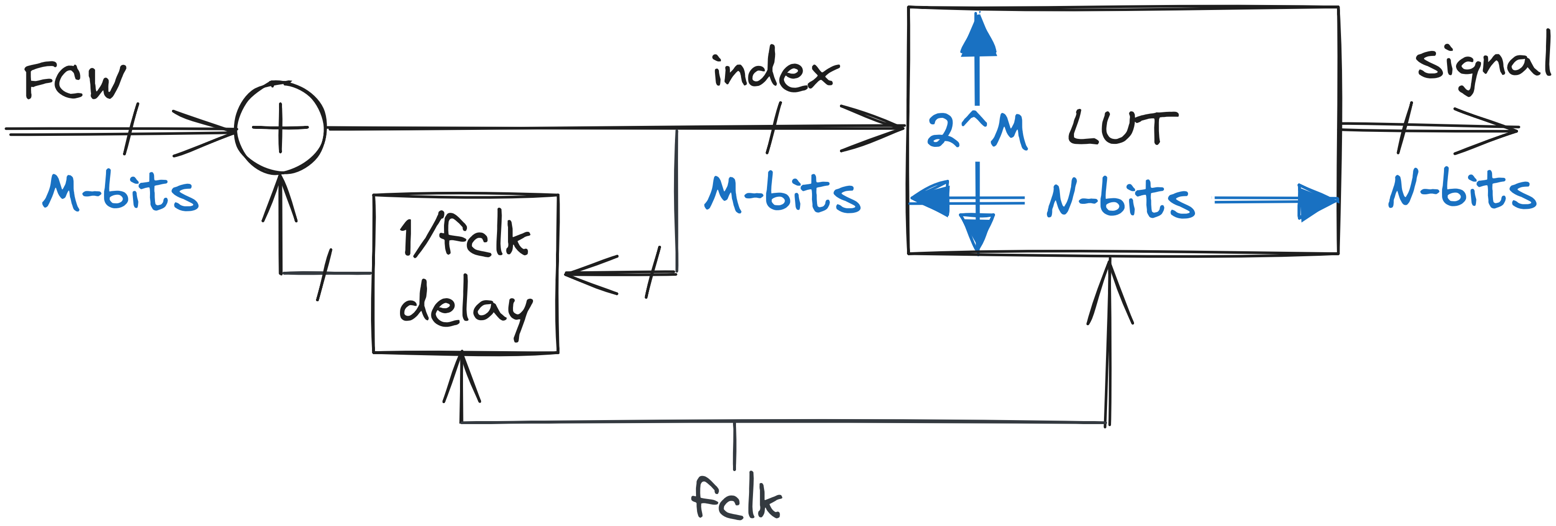
Assume frequency accuracy required < 1ppb

ppb error = 1e9 x fres/fclk = 1e9/2^M -> M = 30

Assuming quantization accuracy required > 70dB

SNR = 6N+1.76 -> N = 12

This means LUT size = 2^30x12/8 = 1.5GBytes!

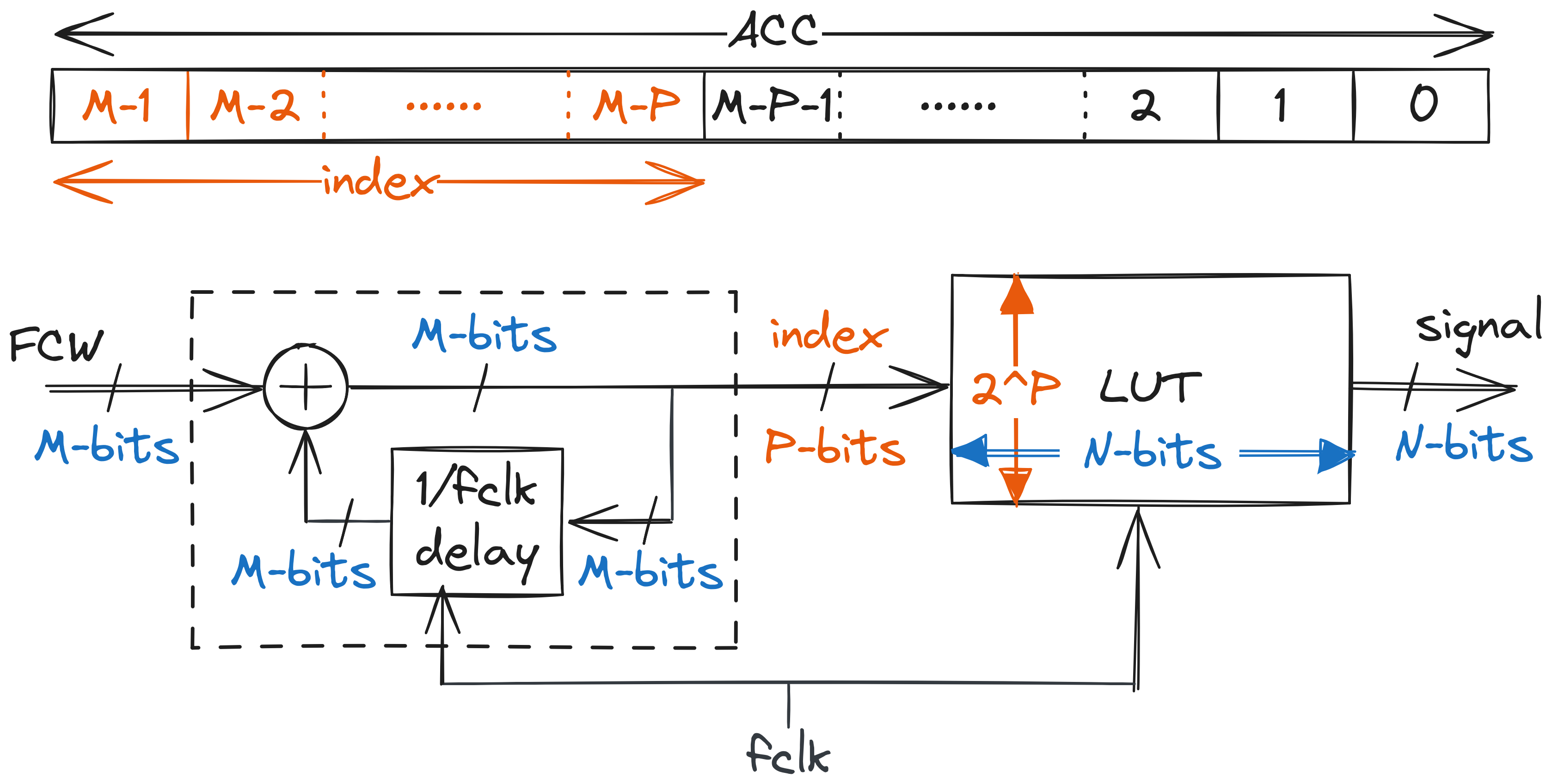


Phase Truncation

We reduce LUT size by truncating the ACC value to P MSBs to access LUT

Frequency resolution still = fclk/2^M

But LUT size is now reduced by a factor of 2^(M-P)



Phase Truncation

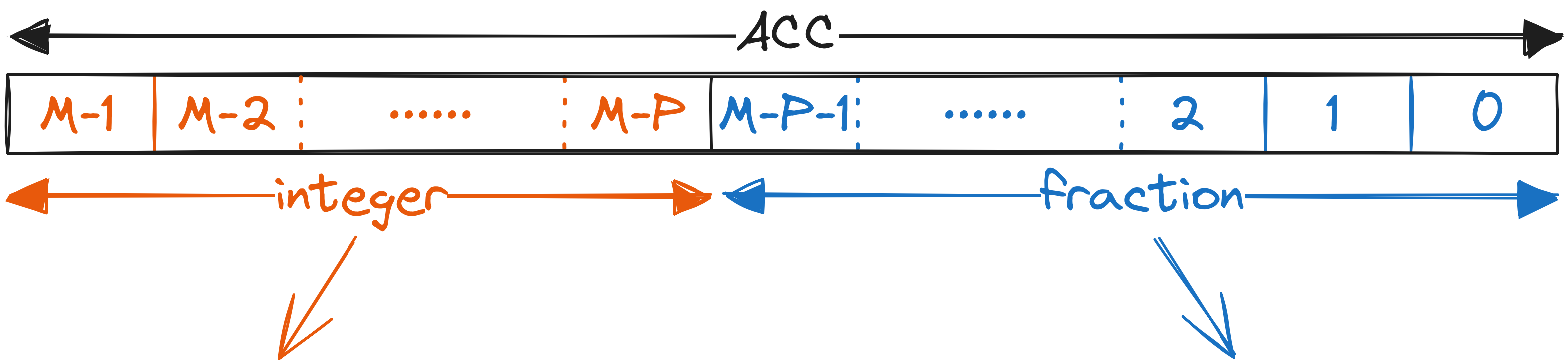
Truncation effectively introduces integer and fractional phase steps

Integer phase step $\varphi o = 2\pi/2^P$

Fractional phase step $\varphi f = 2\pi/2^M$

Only the integer portion is used to access the LUT

Dropping the fractional part introduces a phase truncation error

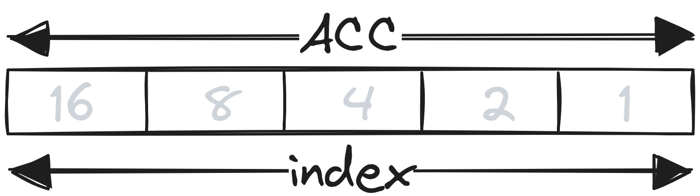


This portion is the LUT index. As far as the LUT is concerned, at any given clock cycle, the value stored here is phase of the sinusoid

ACC continues to accumulate fractional steps. At any given clock cycle, value stored here is the phase truncation error introduced

Phase Truncation

Example: M=P=5, FCW=3



$\varphi o = 360/2\text{^}5 = 11.25\text{deg.}$

$\varphi f = 0$

CLK	ACC (dec.)	ACC (bin.)	φ (deg.)
0	0	00000	0
1	3	00011	3x11.25
2	6	00110	6x11.25
3	9	01001	9x11.25
4	12	01100	12x11.25
5	15	01111	15x11.25
⋮	⋮	⋮	⋮

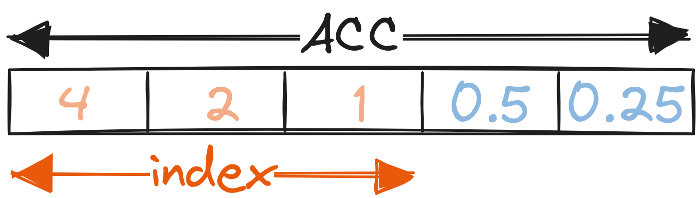
Truncation is equivalent to:

ACC/2^(M-P)

FCW/2^(M-P)

φox2^(M-P)

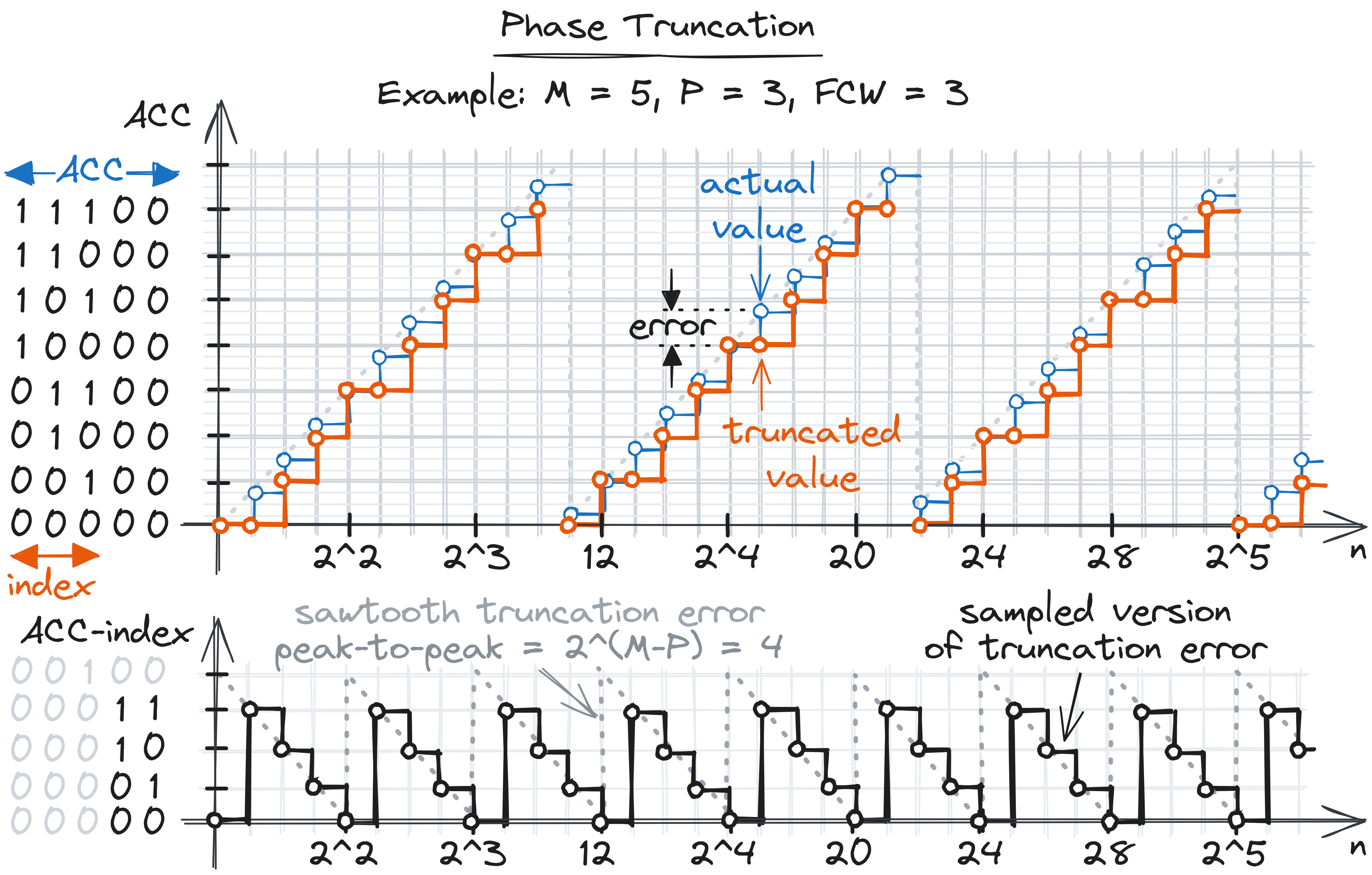
Example: M=5, P=3, FCW=3



$\varphi o = 360/2\text{^}3 = 45\text{deg.}$

$\varphi f = 360/2\text{^}5 = 11.25\text{deg.}$

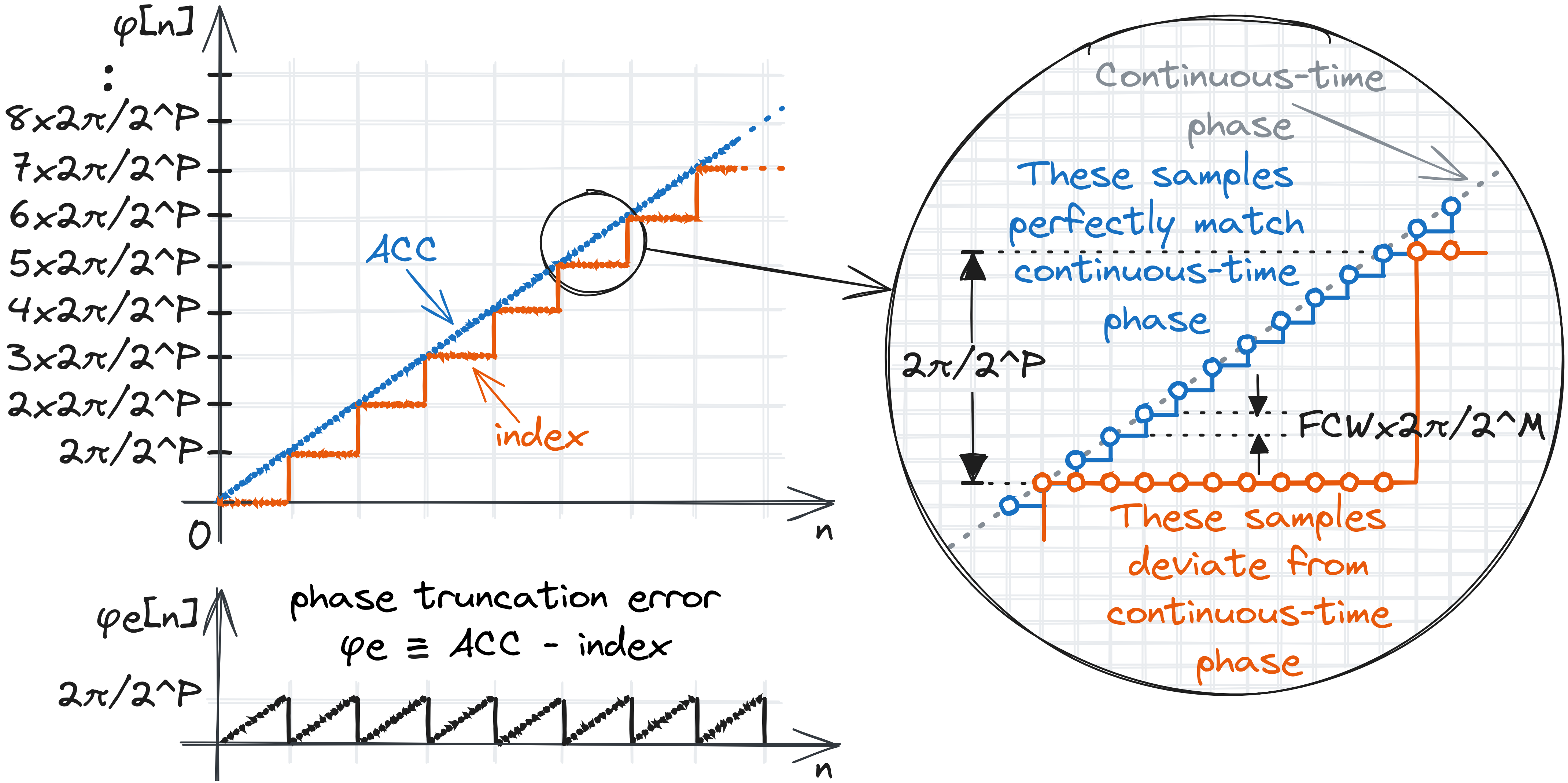
CLK	ACC (dec.)	ACC (bin.)	φ (deg.)	φ error (deg.)
0	0.0	00000	0	0
1	0.75	00011	0	3x11.25
2	1.5	00110	45	2x11.25
3	2.25	01001	2x45	1x11.25
4	0.0	01100	3x45	0
5	3.75	01111	3x45	3x11.25
⋮	⋮	⋮	⋮	⋮



Phase Truncation
Example: M = 5, P = 3, FCW = 3
ACC
ACC
1 1 1 0 0
1 1 0 0 0
1 0 1 0 0
1 0 0 0 0
0 1 1 0 0
0 1 0 0 0
0 0 1 0 0
0 0 0 0 0
index
actual value
error
truncated value
2^2
2^3
12
2^4
20
24
28
2^5
n
ACC-index
0 0 1 0 0
0 0 0 1 1
0 0 0 1 0
0 0 0 0 1
0 0 0 0 0
sawtooth truncation error
peak-to-peak = 2^(M-P) = 4
sampled version
of truncation error
2^2
2^3
12
2^4
20
24
28
2^5
n

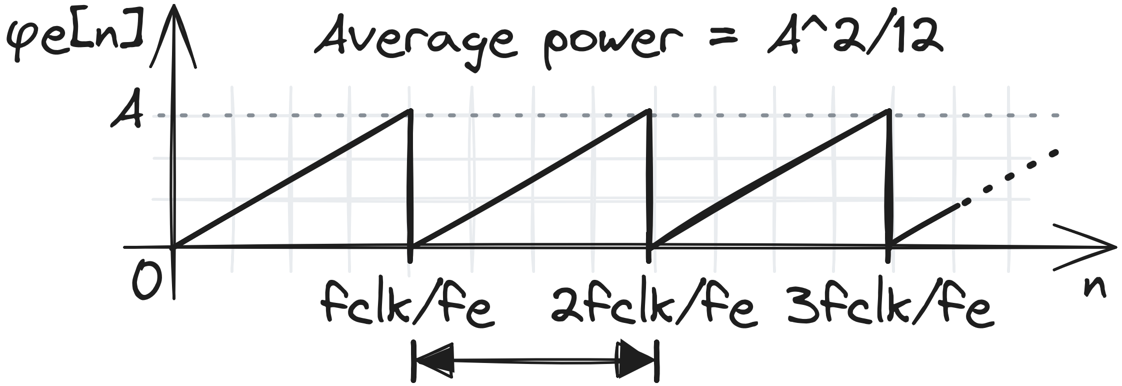
Phase truncation error is a sawtooth waveform

Worst-case (max.) peak-to-peak sawtooth amplitude = 2π/2^P



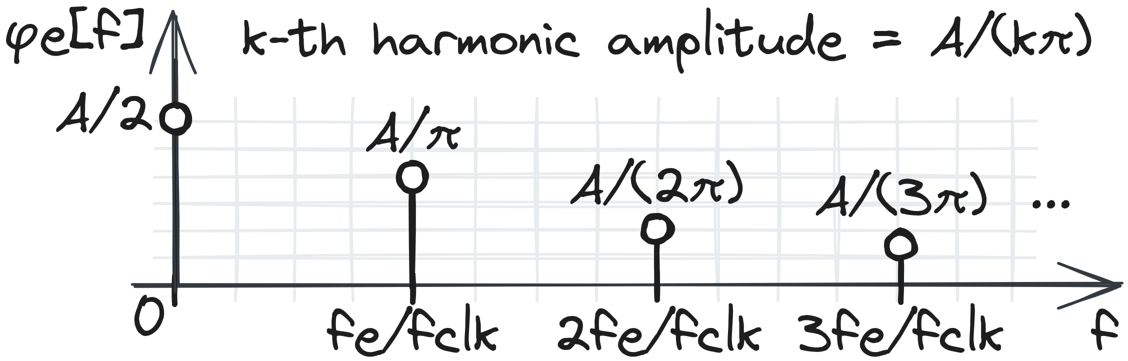
Let's examine the truncation error sequence φe[n]

Time domain



Fundamental period fclk/fe depends on FCW

Frequency domain

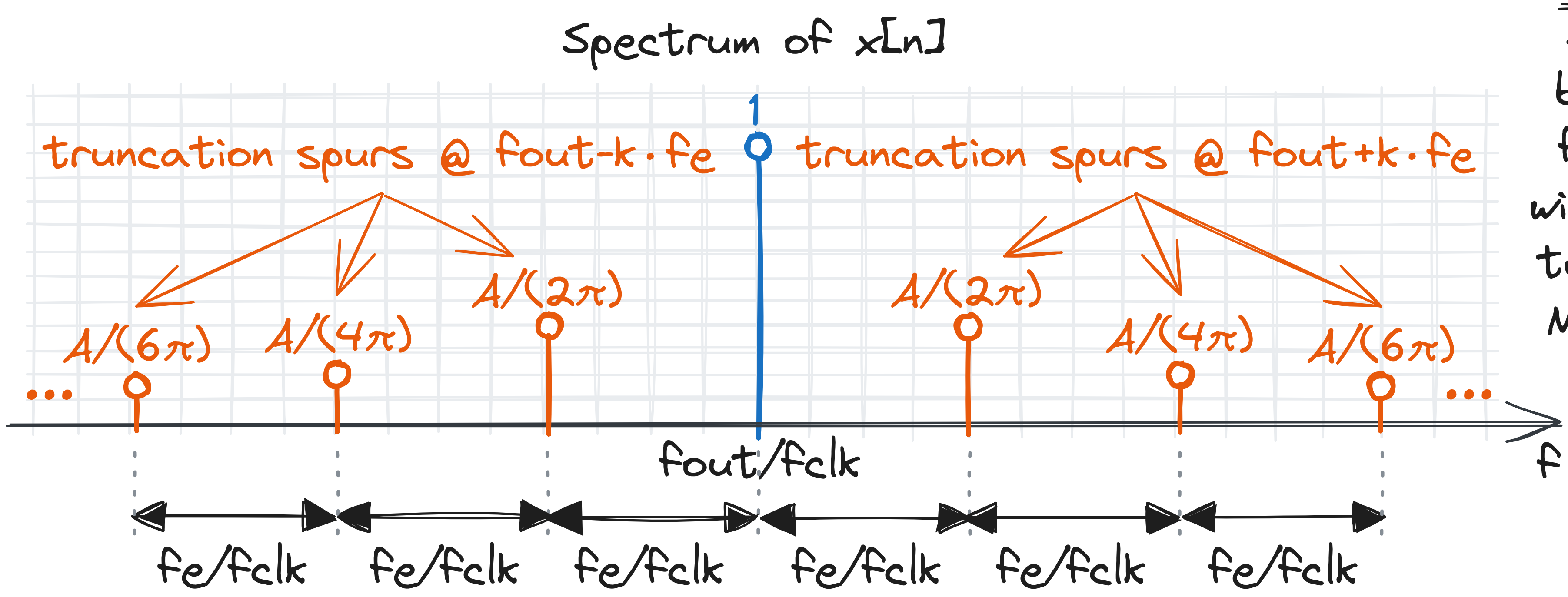


The generated sinusoid now has a phase error term $\varphi e[n]$ that causes spurs

$$x[n] = \sin(2\pi \times fout/fclk \times n + \varphi e[n])$$

Expanding the sum of angles and assuming $\varphi e[n]$ is small

$$x[n] = \underbrace{\sin(2\pi \times fout/fclk \times n)}_{\text{desired sinusoid}} + \underbrace{\varphi e[n] \times \cos(2\pi \times fout/fclk \times n)}_{\text{phase truncation error } xe[n]}$$



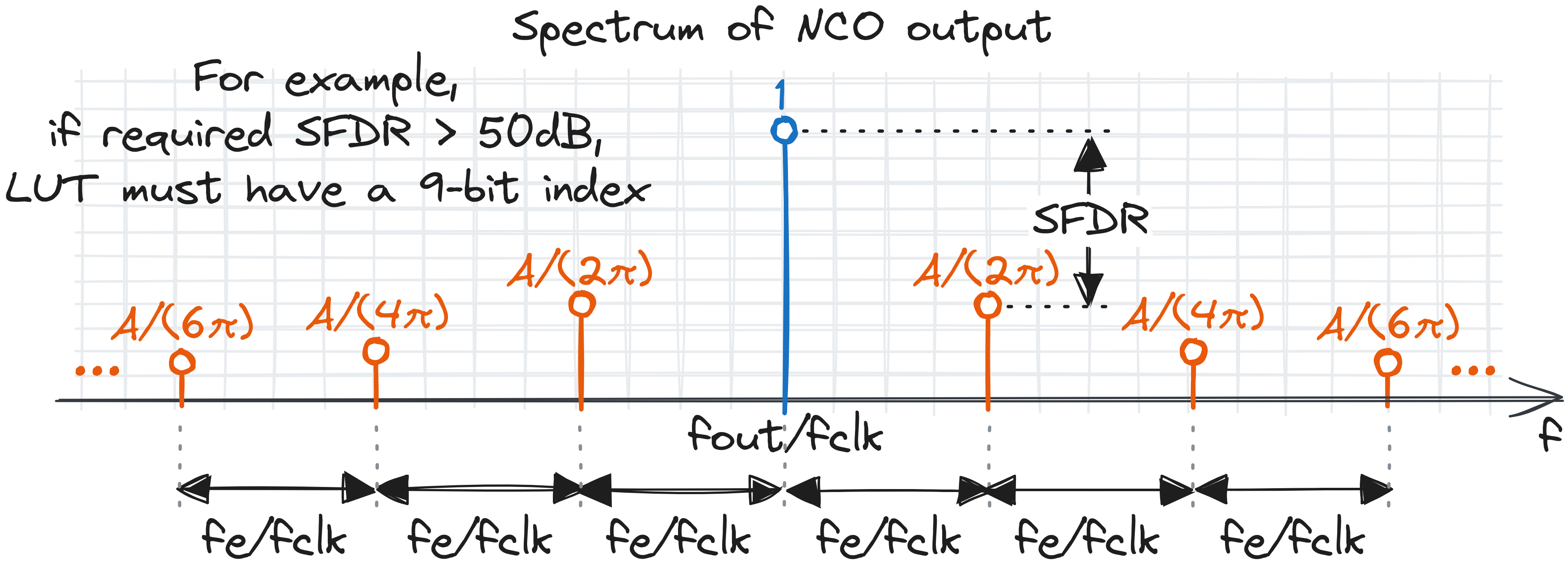
Note:
Spurs beyond fclk/2 will alias to first Nyquist zone

Spurious free dynamic range (SFDR) is the ratio of desired signal to highest spur

$$SFDR = 1/(A/(2\pi))$$

Worst-case (min.) SFDR corresponds to max. A which is $2\pi/2^P$

$$\boxed{SFDR = 6 \times P \text{ [dB]}}$$



We can also calculate SNR which is the ratio of desired signal to all spurs

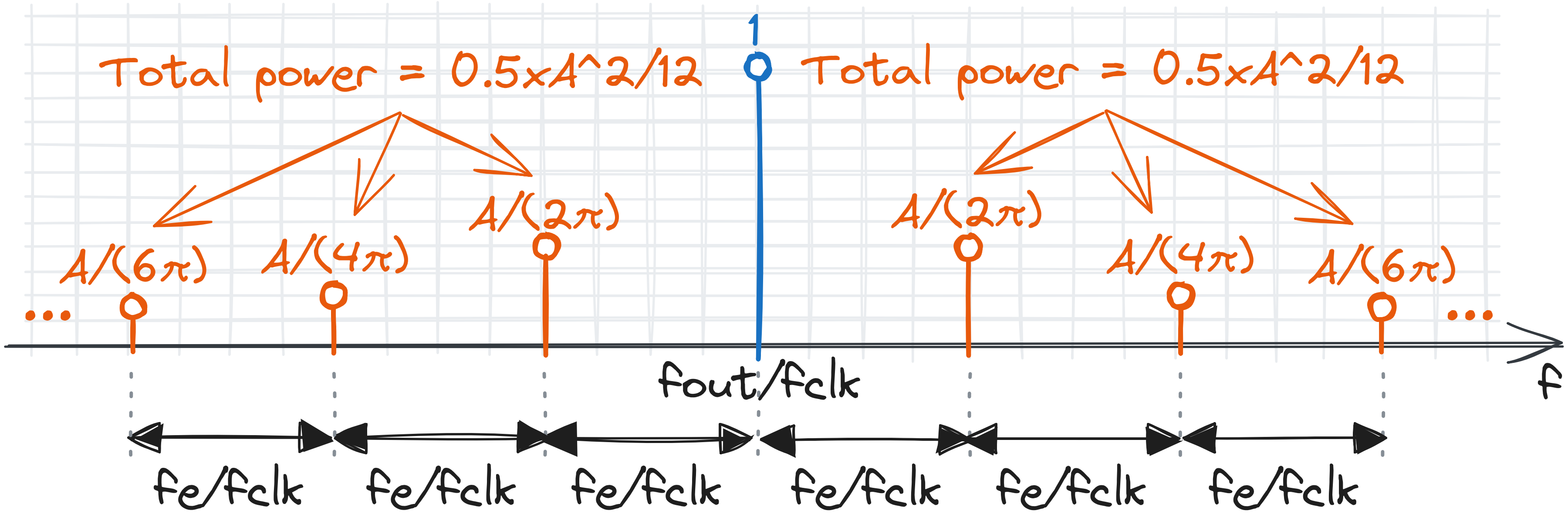
Assuming $\varphi e[n]$ and quadrature signal are uncorrelated

$$SNR = 1/(A^2/12)$$

Worst-case (min.) SNR corresponds to max. A which is $2\pi/2^P$

$$\boxed{SNR_P = 6.02 \times P - 5.17 \text{ [dB]}}$$

Spectrum of NCO output



Amplitude Quantization

Finite LUT word length causes amplitude quantization

P-bits — N-bit LUT word

index	$\sin(\varphi)$
0	$\sin(0)$
1	$\sin(2\pi/2^P)$
2	$\sin(2\times2\pi/2^P)$
⋮	⋮
2^P-1	$\sin((2^P-1)\times2\pi/2^P)$

2^P phase levels

2^N unique amplitude levels

If $N < P$, some amplitude levels are repeated

For example, if $N = P-1$, there is one $\sin(\varphi)$ value for every 2 index values

Amplitude Quantization

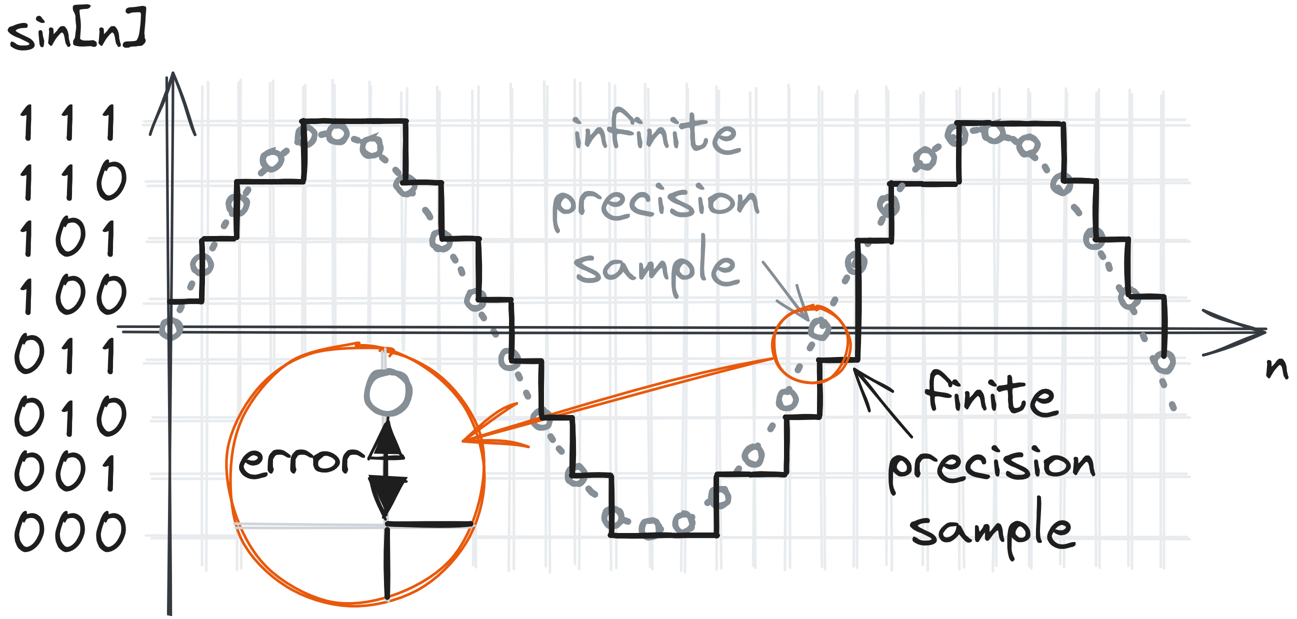
Amplitude quantization error is similar to DAC/ADC quantization noise

So, for an N-bit LUT word length, the familiar SNR relationship applies

$$\boxed{SNR_A = 6.02N + 1.76 \text{ [dB]}}$$

Example:

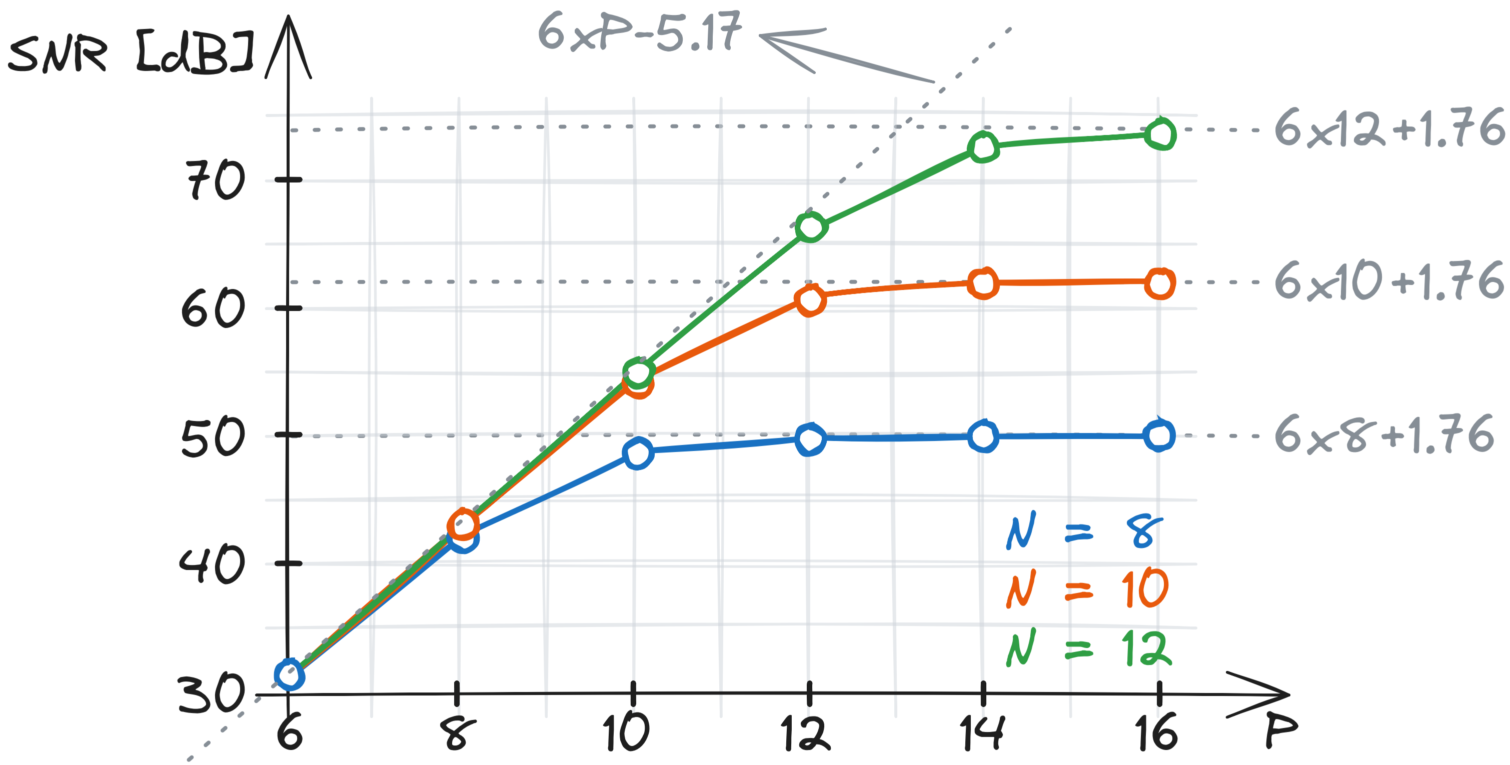
N = 3 -> 2^3 quantization levels -> SNR = 19.8dB



Phase truncation and amplitude quantization together determine signal quality

Total SNR is the uncorrelated sum of both types of errors

$$1/SNR = 1/SNR_A + 1/SNR_P$$

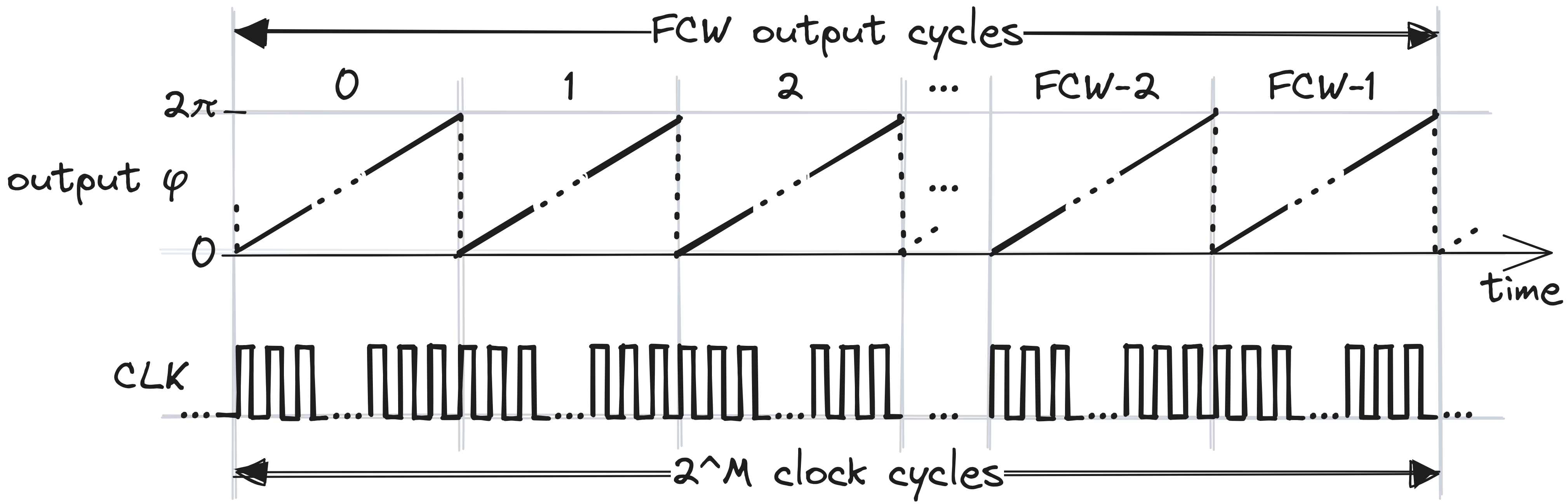


The NCO as a frequency divider

We can re-write the input-output frequency relation of an NCO

FCW/fout = 2^M/fclk

The above tells us that clocking an M-bit ACC fits 2^M clock cycles into FCW output cycles, for any value of FCW
From this perspective, the NCO is a frequency divider

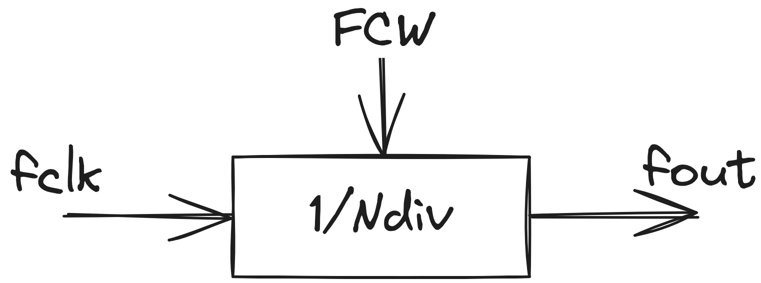


The NCO as a frequency divider

The division ratio is programmable via FCW

fout = fclk/Ndiv

Ndiv = 2^M/FCW



The NCO is a divider capable of integer & fractional division

For an M-bit ACC, there are 2^(M-1) alias-free FCW values

Of those, only M FCW values (2^0, 2^1, 2^2 .. 2^(M-1)) result in integer division

All other FCW values (the majority) result in fractional division

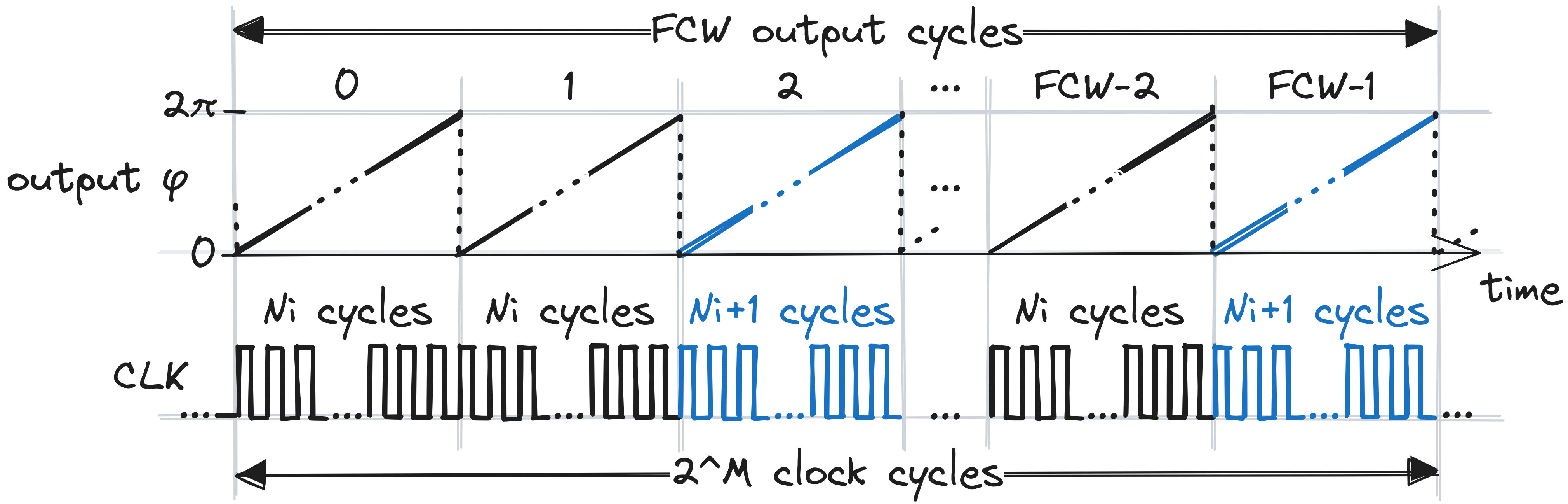
The NCO as a frequency divider

So, generally, division ratio has an integer part Ni and a fractional part K/FCW

$$Ndiv = Ni + K/FCW$$

Fractional division is achieved on "average" over FCW output cycles

The NCO divides by Ni for FCW-K output cycles and by Ni+1 for K output cycles



The NCO as a frequency divider

Example:

$$M = P = 5, \; FCW = 7$$

$$Ndiv = 2^5/7 = \underbrace{4}_{\text{Divide by 4 for 3 out of 7 cycles}} + \underbrace{4/7}_{\text{Divide by 5 for 4 out of 7 cycles}}$$

Divide by 4 for 3 out of 7 cycles

Divide by 5 for 4 out of 7 cycles

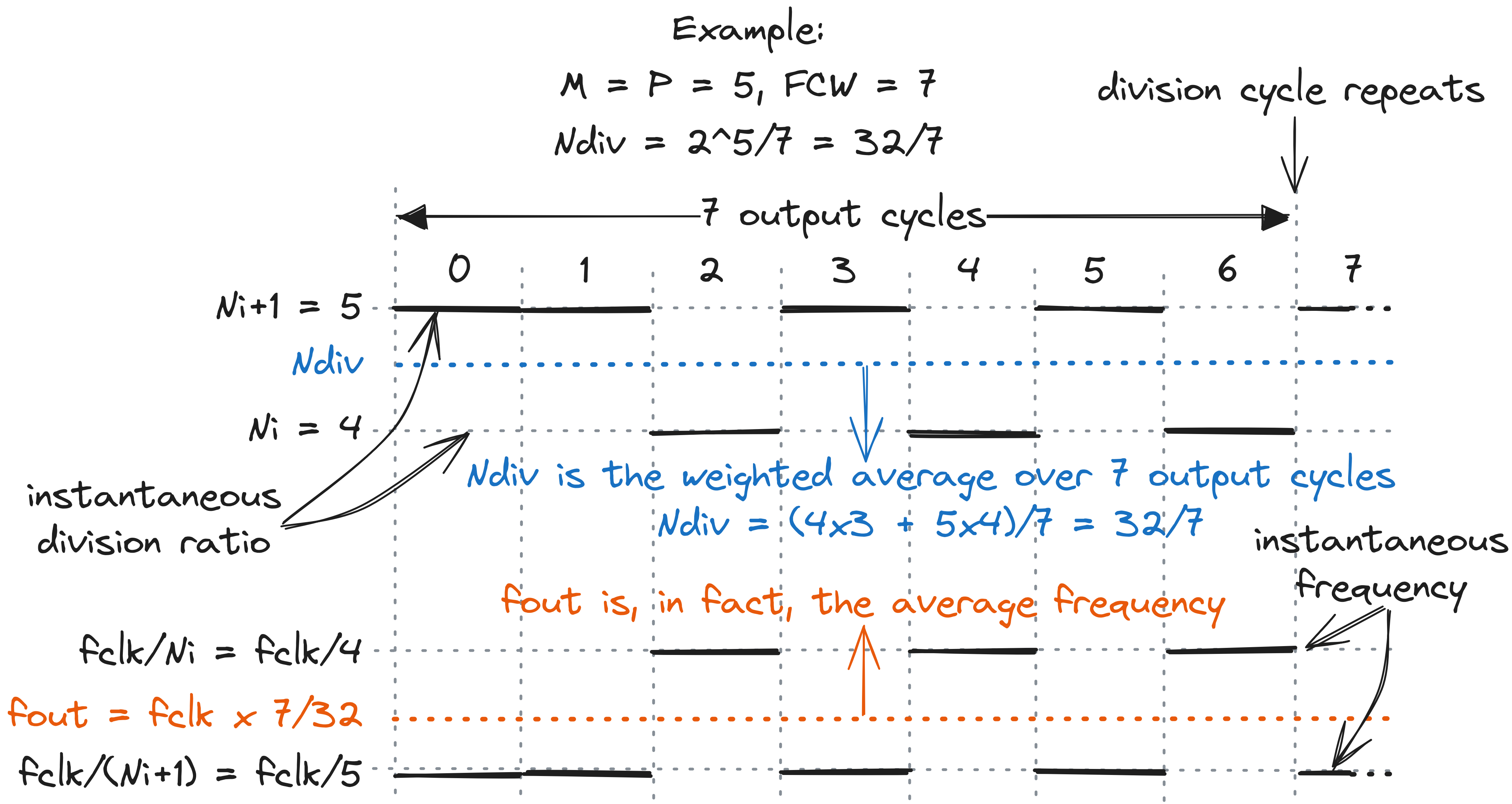
Output cycle #	ACC	# of clk cycles
0	0, 7, 14, 21, 28,	5
1	3, 10, 17, 24, 31,	5
2	6, 13, 20, 27,	4
3	2, 9, 16, 23, 30,	5
4	5, 12, 19, 26,	4
5	1, 8, 15, 22, 29	5
6	4, 11, 18, 25,	4
7	0, 7, ..	5
⋮		⋮

7 output cycles

2^5 clk cycles

division cycle repeats

The NCO as a frequency divider

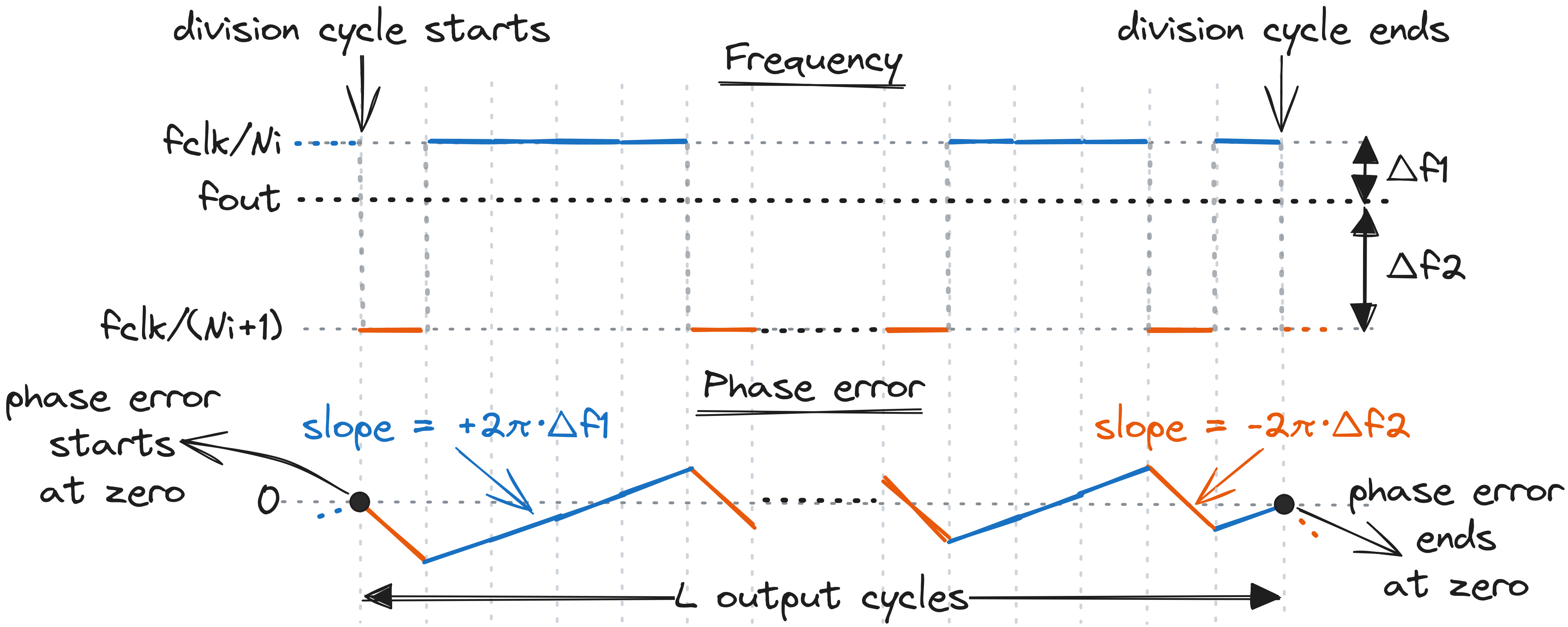


The NCO as a frequency divider

Difference between instantaneous & average frequency results in phase error

As we already know, phase error translates to output spurs

Phase error has a period of L/fout -> spurs at harmonics of fout/L



The NCO as a frequency divider

We can calculate spur frequencies by finding the phase error period L

How many clock cycles are in those L output cycles?

Remember that over the same L output cycles, ACC pattern also repeats

We also know that for an M-bit ACC, L can't exceed 2^M clock cycles (FCW = 1)

So, if FCW has a common divisor with 2^M, L is < 2^M clock cycles

And if FCW has no common divisor with 2^M, L = 2^M clock cycles

The greatest common divisor is GCD(FCW, 2^M)

And we call the number of clock cycles in L the grand repetition rate (GRR)

GRR = 2^M / GCD(FCW, 2^M)

Which results in spur frequencies at

fspur = k x GCD(FCW, 2^M)/2^M x fclk k = 0, 1, 2, ..

These spurs occur even without any phase truncation between ACC to LUT

The NCO as a frequency divider

Example 1:

$M = P = 5$, $FCW = 7$

$f_{out} = 7/32 \times f_{clk}$

$GRR = 2^5/GCD(2^5,7) = 32$ clock cycles

$f_{spur} = 0$, $f_{clk}/32$, $2 \times f_{clk}/32$, $3 \times f_{clk}/32$ .. etc.

Example 2:

$M = P = 8$, $FCW = 6$

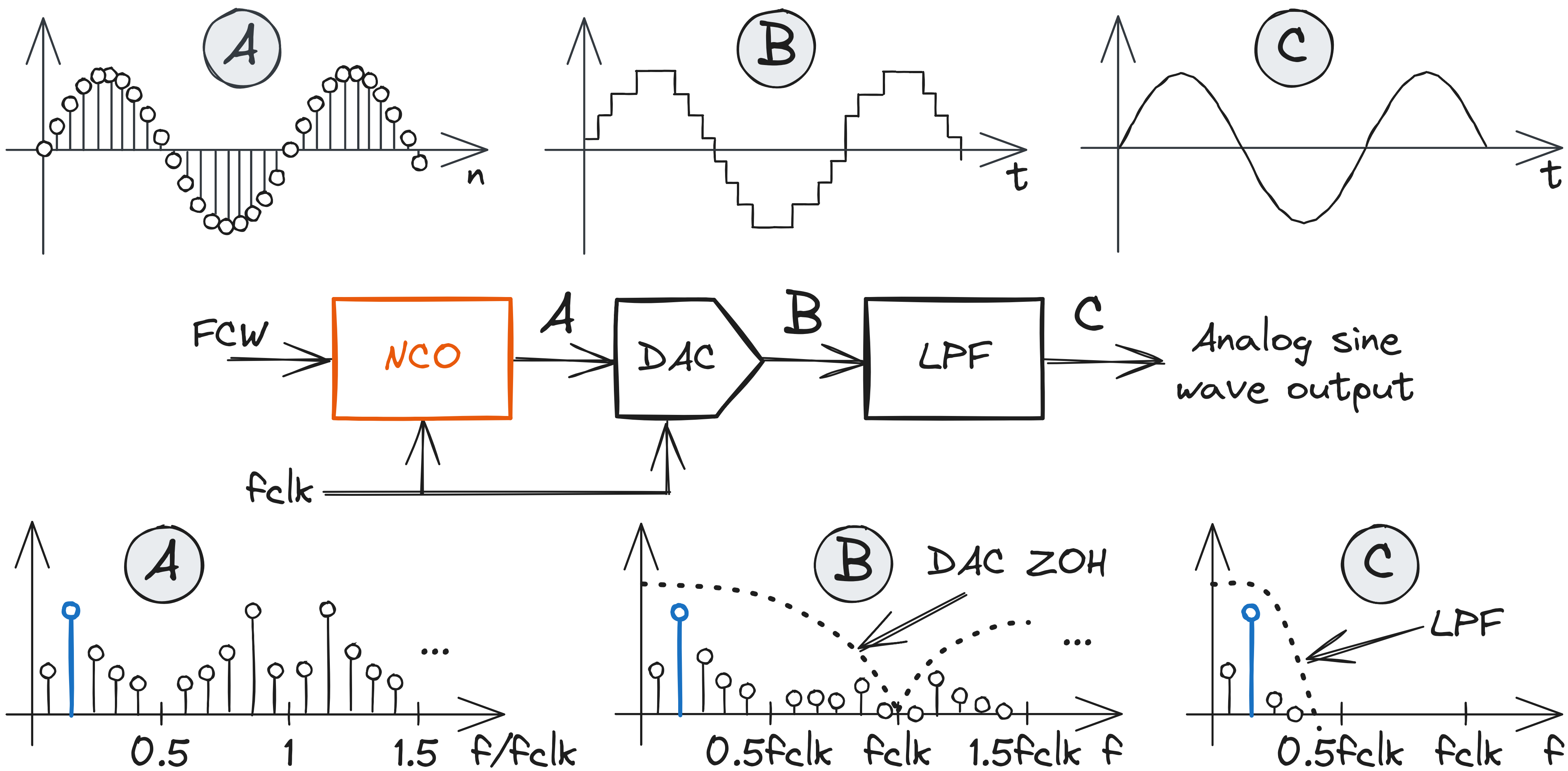
$f_{out} = 6/256 \times f_{clk} = 3/128 \times f_{clk}$

$GRR = 2^8/GCD(2^8,6) = 128$ clock cycles

$f_{spur} = 0$, $f_{clk}/128$, $2 \times f_{clk}/128$, $3 \times f_{clk}/128$ .. etc.

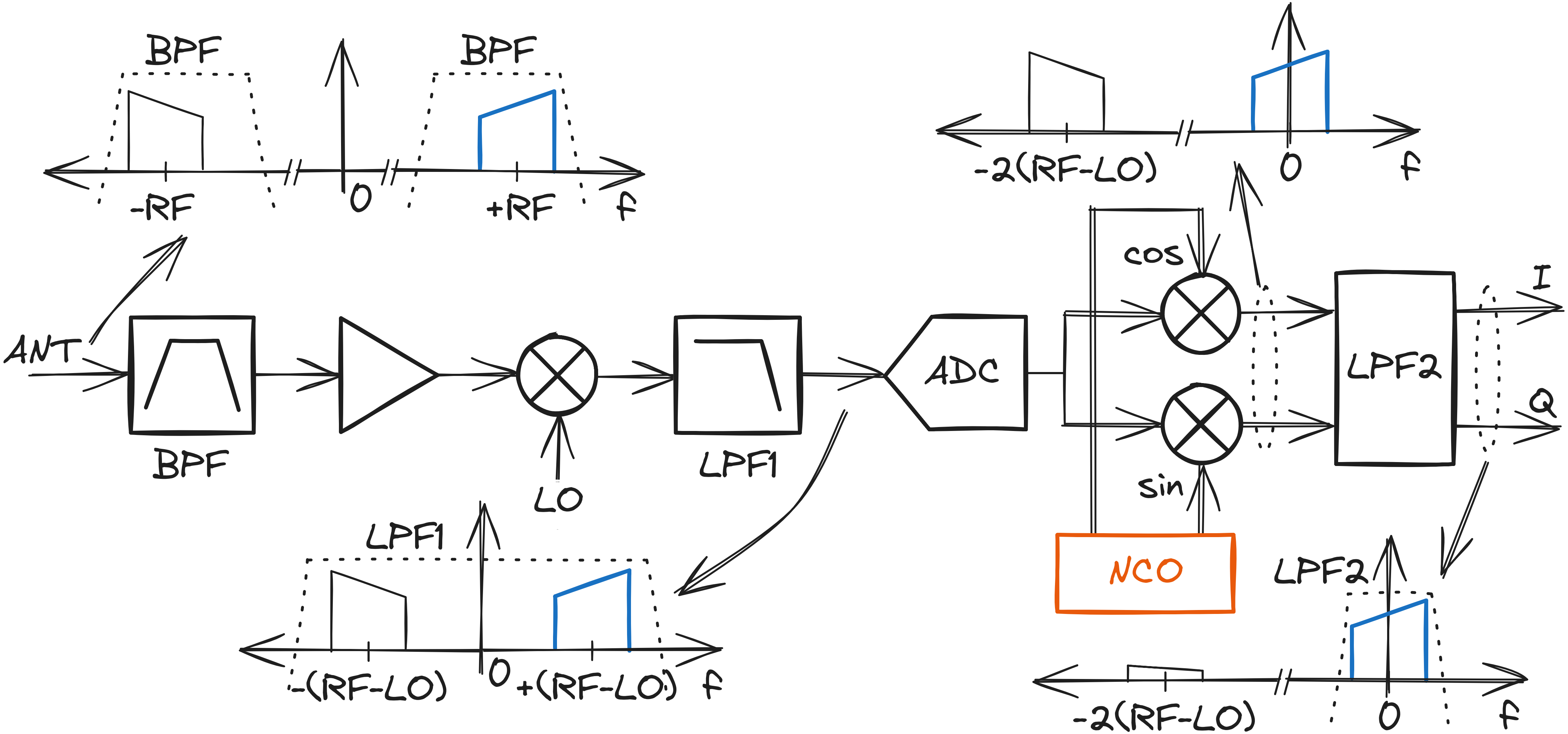
Applications of NCOs

Direct digital synthesis (DDS) of analog reference signal



Applications of NCOs

Digital IQ downconversion in wireless receivers



Applications of NCOs

Carrier synchronization in wireless receivers