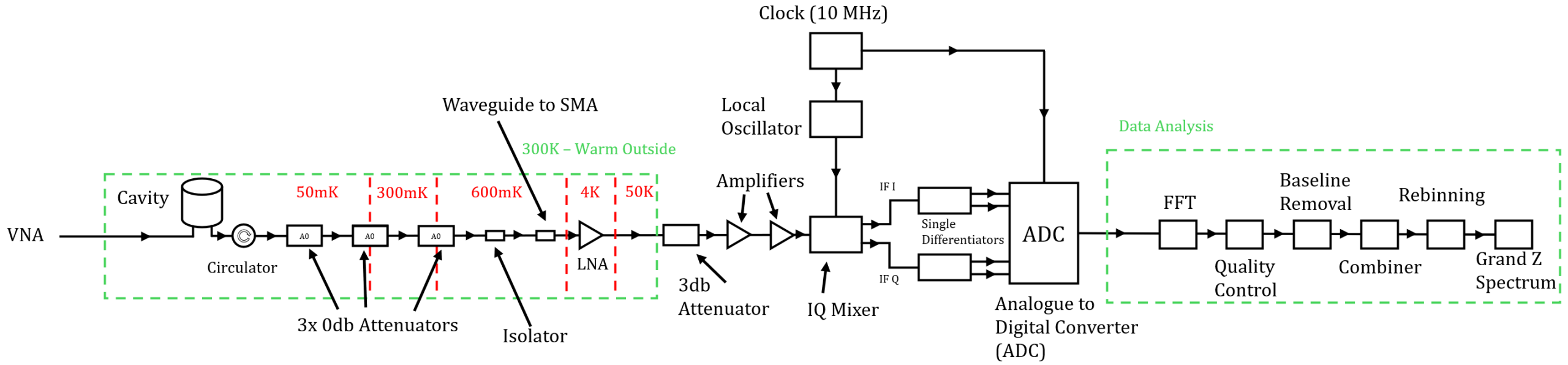
Simulating Axions

Blue Carn

Context

- To make a more accurate simulation, need to start from the beginning of the pipeline
- This means working in time domain rather than frequency domain
- We can then simulate the individual components of the RF chain
- Then FFT from time domain into the frequency domain, ready to be integrated into the data analysis

Experiment Pipeline

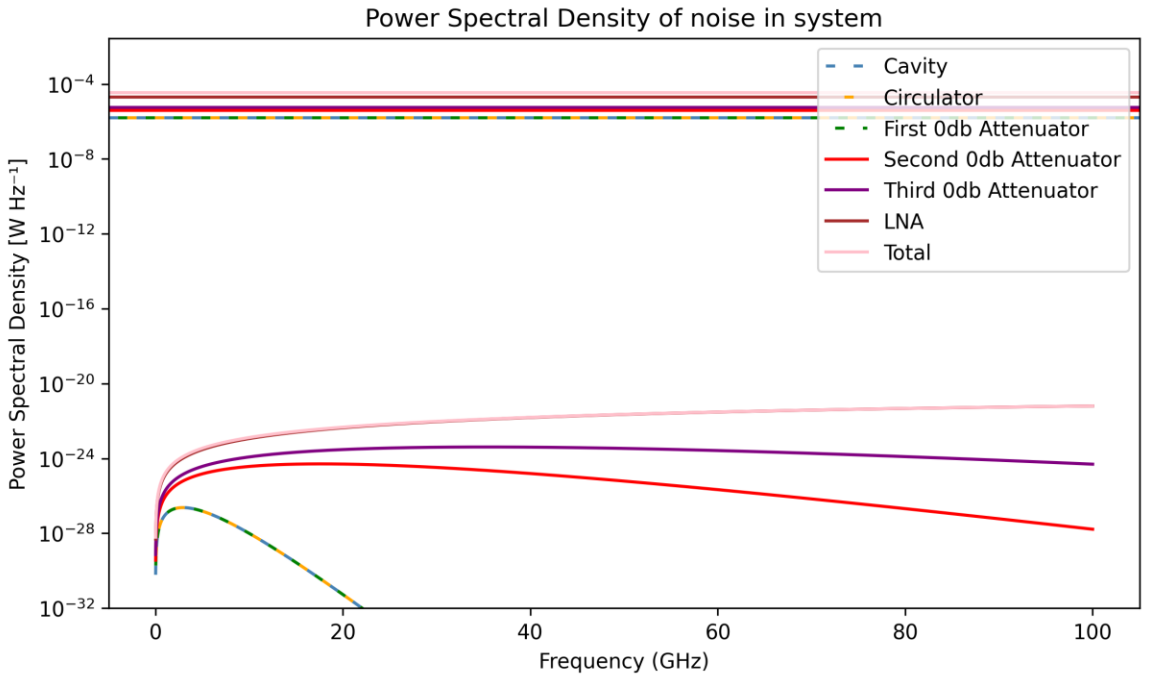


Noise

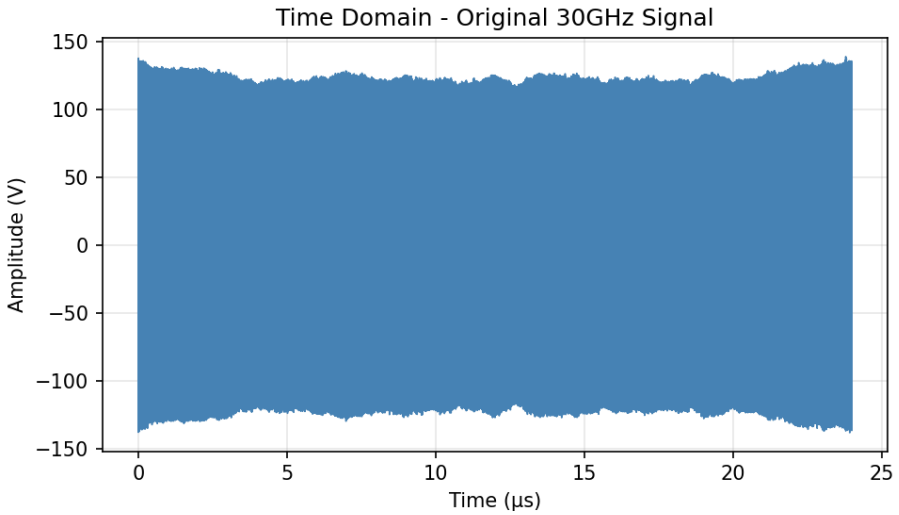
- We consider 2 types of noise
- Blackbody Radiation
  - Simulated in frequency domain
  - $B_\nu(T) = \frac{2h\nu^3}{c^2\left(e^{\frac{h\nu}{kT}}-1\right)}$
- Johnson-Nyquist Noise
  - Simulated in time domain
  - $V_{rms} = \sqrt{4kTZB}$
    - Z = 50Ω – impedance
    - B – bandwidth of component
      - Bandwidths can be found on datasheets of components
      - Widest is LNA at 23-42 GHz, so B = 19GHz for LNA
  - Generate random number within a normal distribution with $\sigma = V_{rms}$

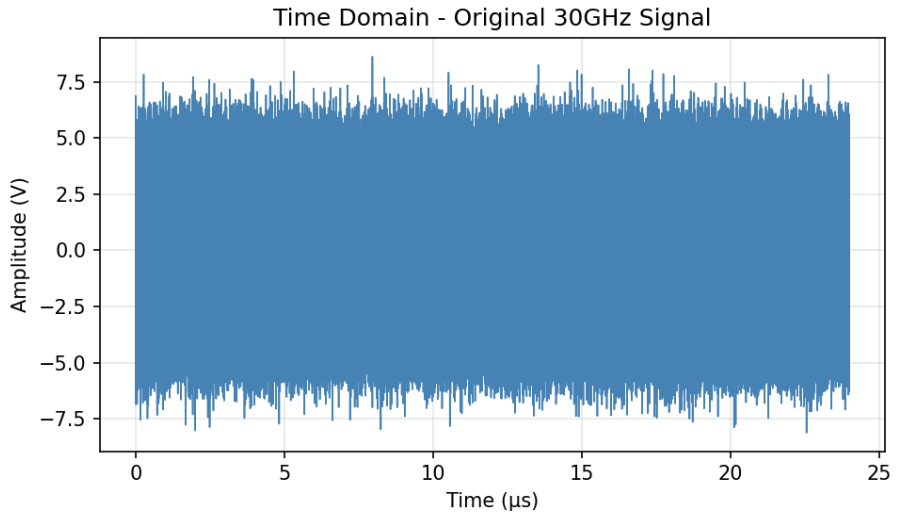
Noise

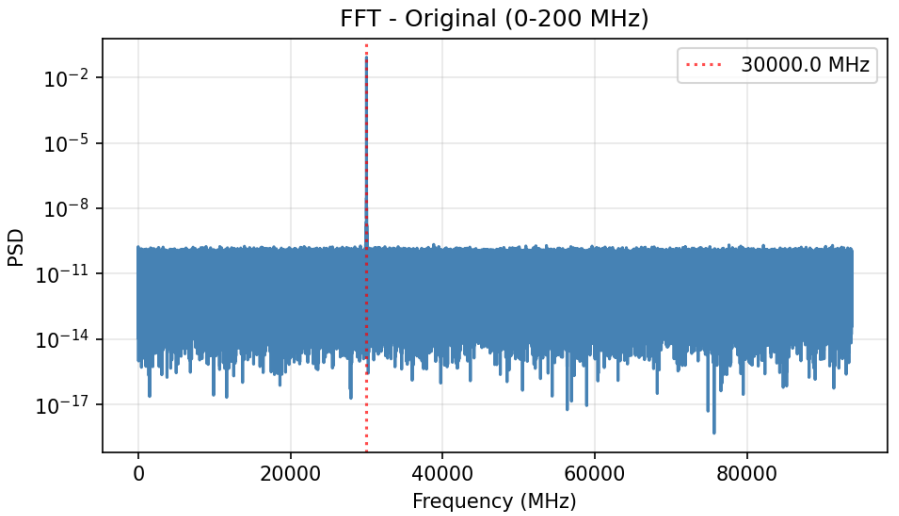
- Simulating components from within the cryostat
- Top part represents rms from Johnson-Nyquist noise
- Bottom half represents blackbody noise
- Johnson-Nyquist Noise dominates, mainly from the LNA

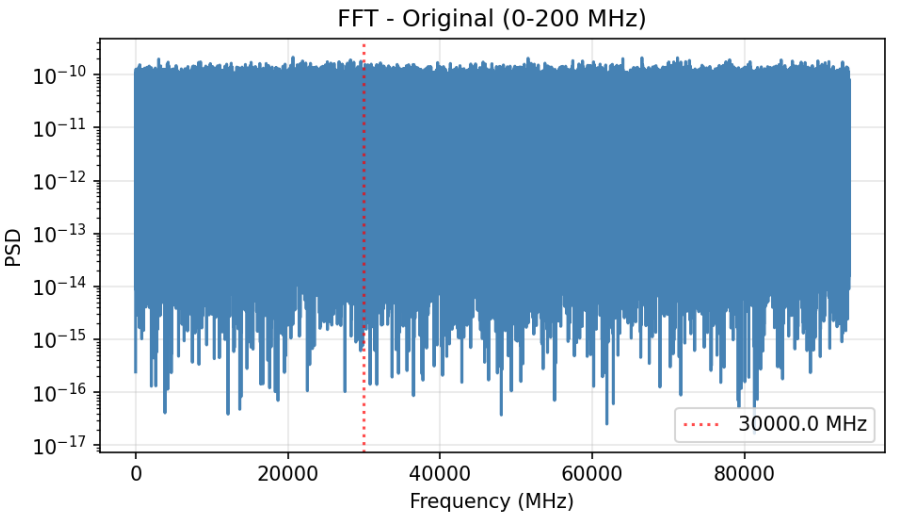


Noise + Large Signal vs Pure Noise



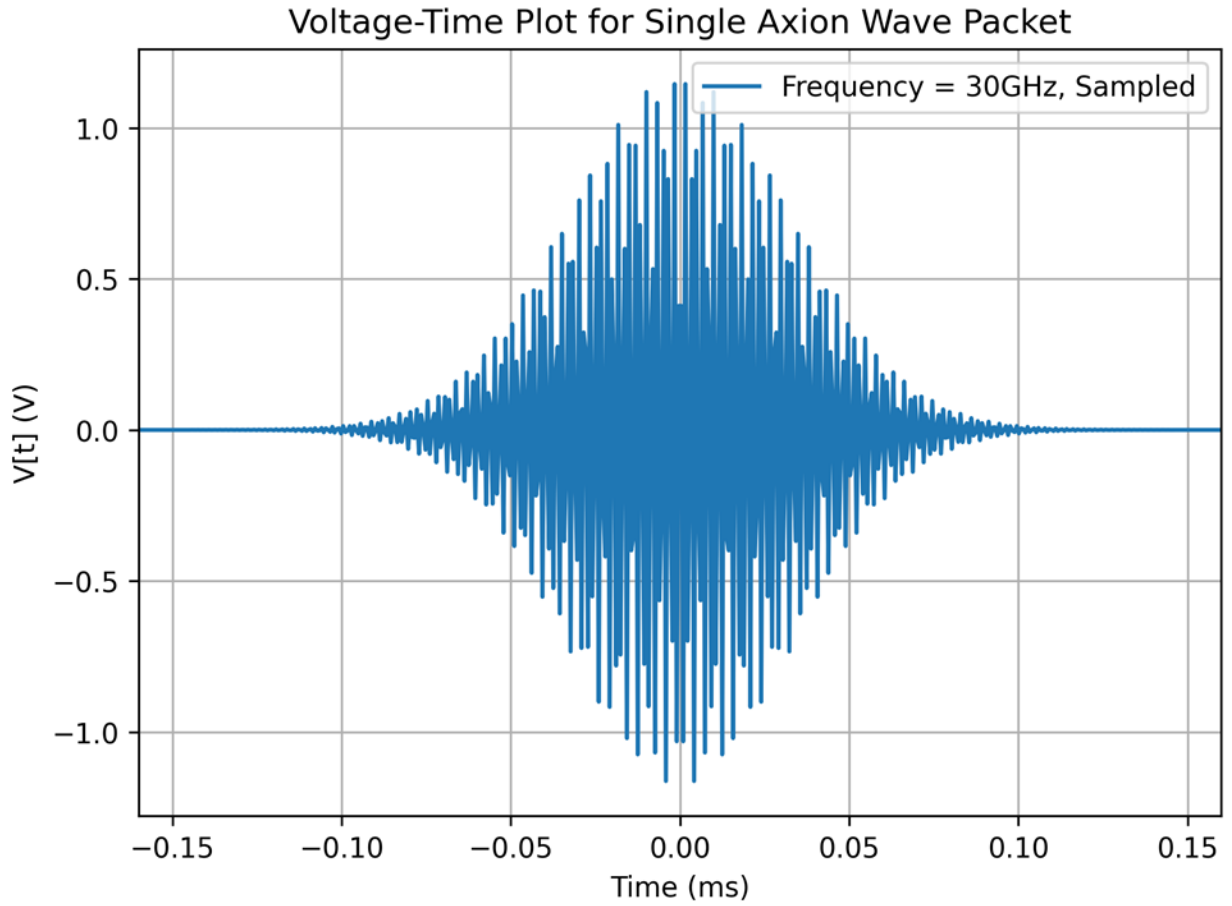






Signal - wave packets

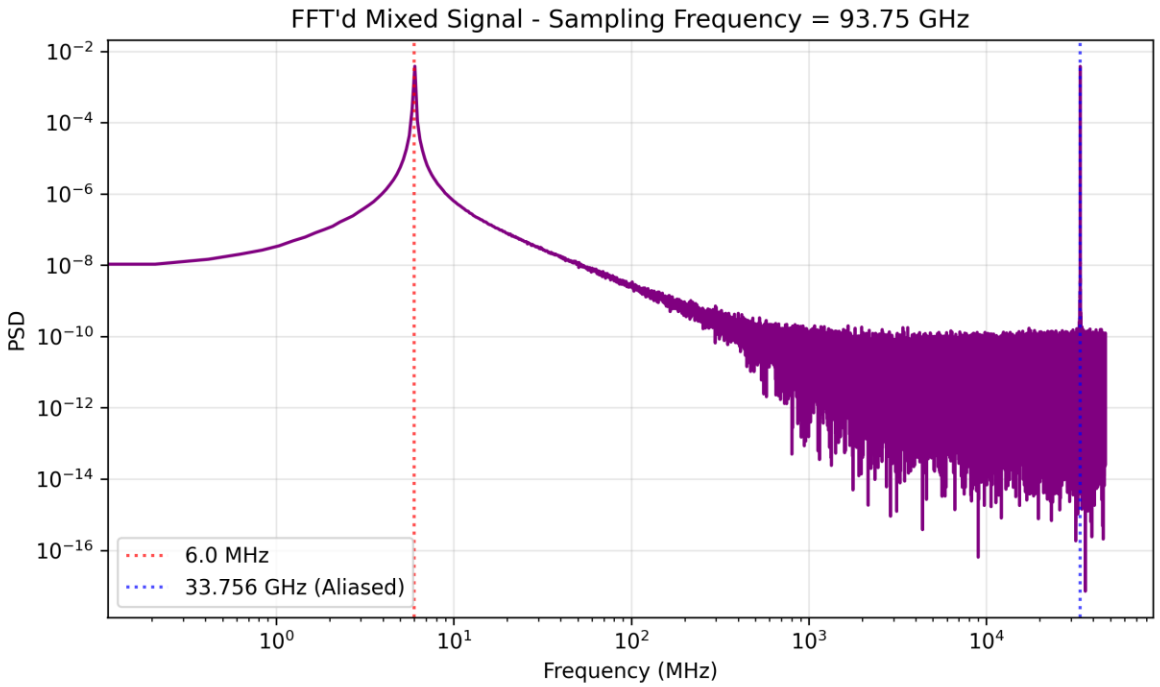
- Complex sinusoidal wave mixed with a gaussian envelope
- Occur with a random amplitude and phase at a random time
- Example of a singular wave packet (pre-sampled)

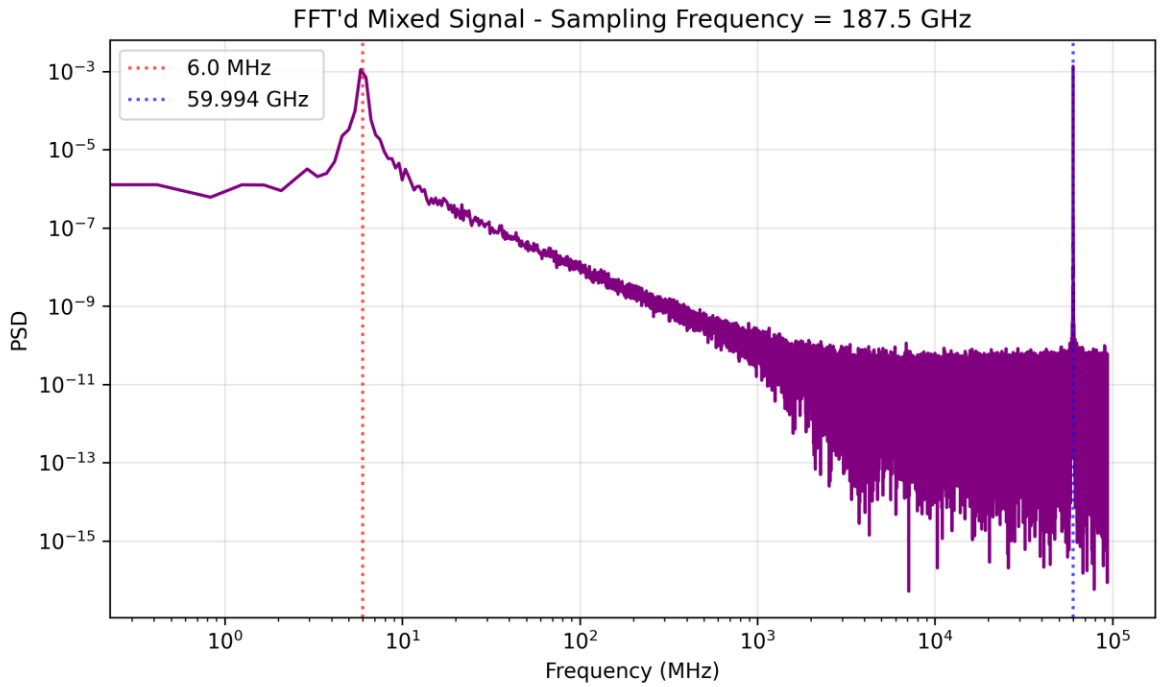


Pre-sampling

- Due to python limitations, all functions are discretised
- For our signal of 30GHz, we can choose the rate it is “pre-sampled”
- Sampling frequency for this pre-sampling must be higher than Nyquist limit otherwise aliasing occurs
- Pre-Sampling is currently set to 6.25 samples per period, which is a sampling frequency of 187.5GHz.
  - Note this is carried over to the final wave of 6MHz, meaning that wave is sampled 31250 times per period – a lot of data for just a 6MHz wave.

Example of Aliasing





What is downmixing?

- Process of converting a high frequency (f_{RF}) signal to a low frequency (f_{out}) signal whilst preserving the data
- This is done by injecting a very similar signal with frequency:

$$f_{LO} = f_{RF} - f_{out}$$

- E.g., if we want to downmix a 30GHz signal to 6MHz, f_{LO} = 29.99 GHz
- But what actually is downmixing?

The maths of downmixing

- Let's assume our RF signal has a function:

$$x_{RF} = A\cos(2\pi\ *30\text{GHz}\ *t)$$

- And our Local Oscillator signal has a function:

$$x_{LO} = B\cos(2\pi\ *29.994\text{GHz}\ *t)$$

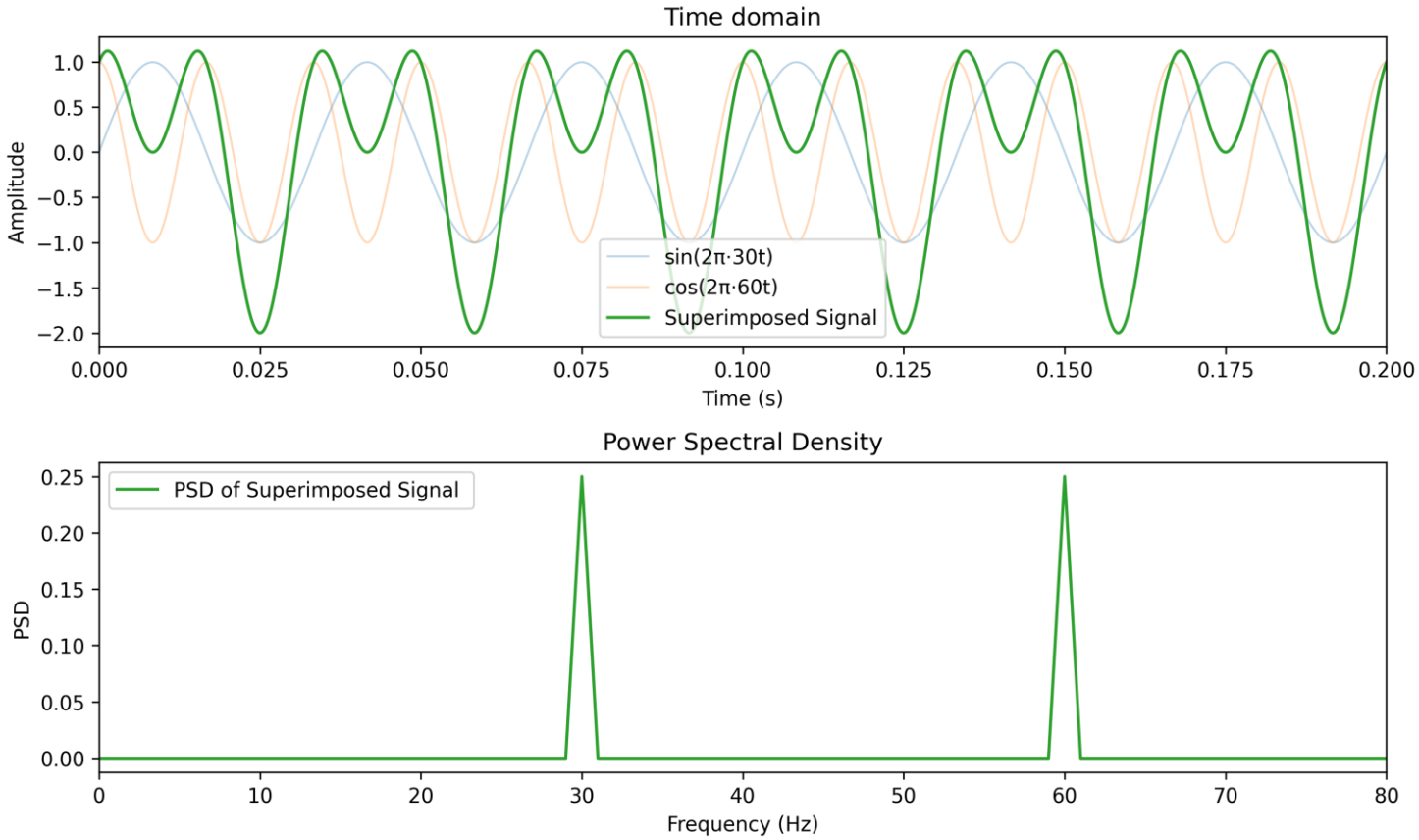
- A mixer multiplies them, and using the trigonometric identity:

$$\cos a\cos b = \frac{1}{2}[\cos(a-b)+\cos(a+b)]$$

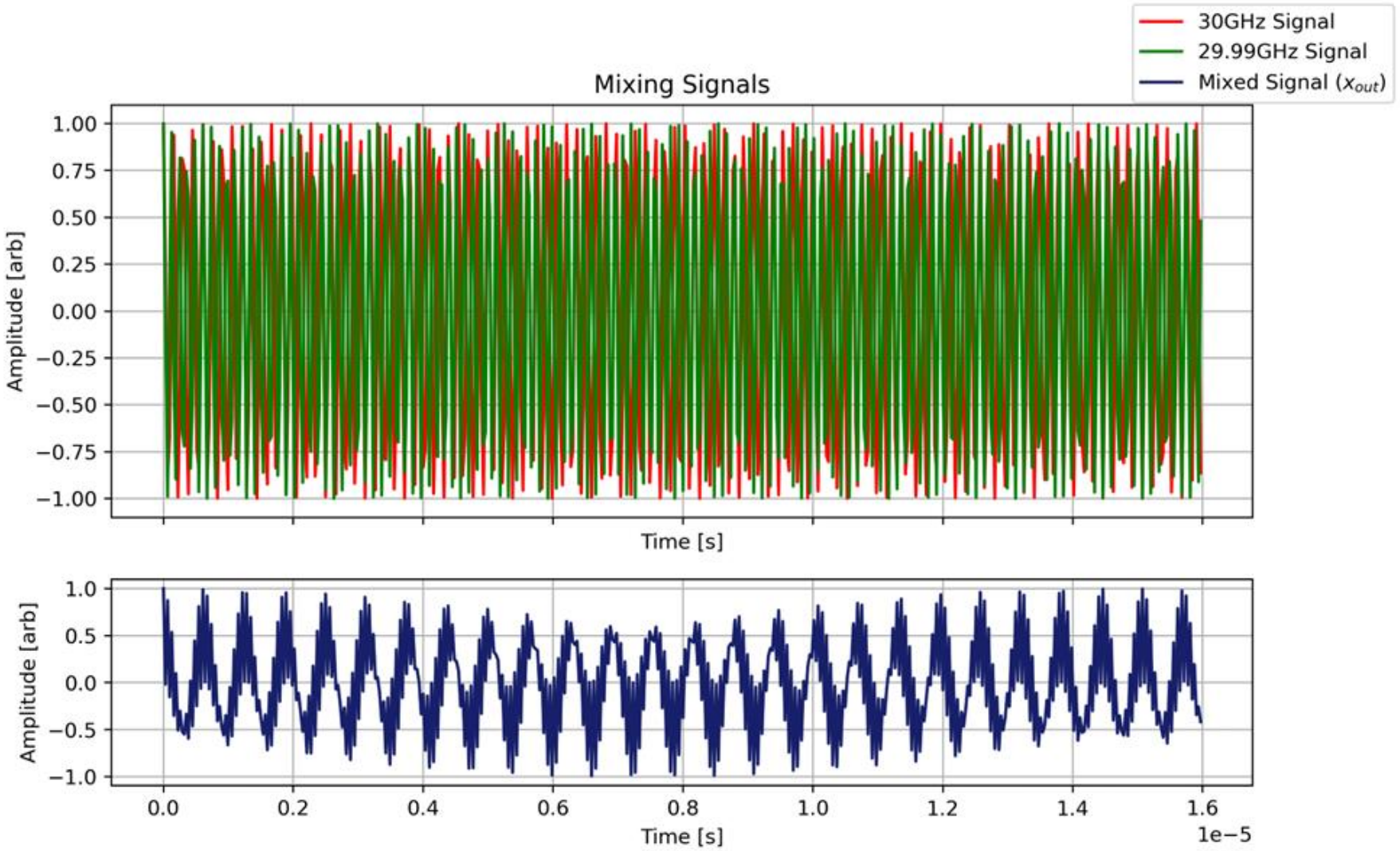
- We get:

$$x_{out} = \frac{AB}{2}[\cos(2\pi\ *0.006\text{GHz}\ *t)+\cos(2\pi\ *59.994\text{GHz}\ *t)]$$

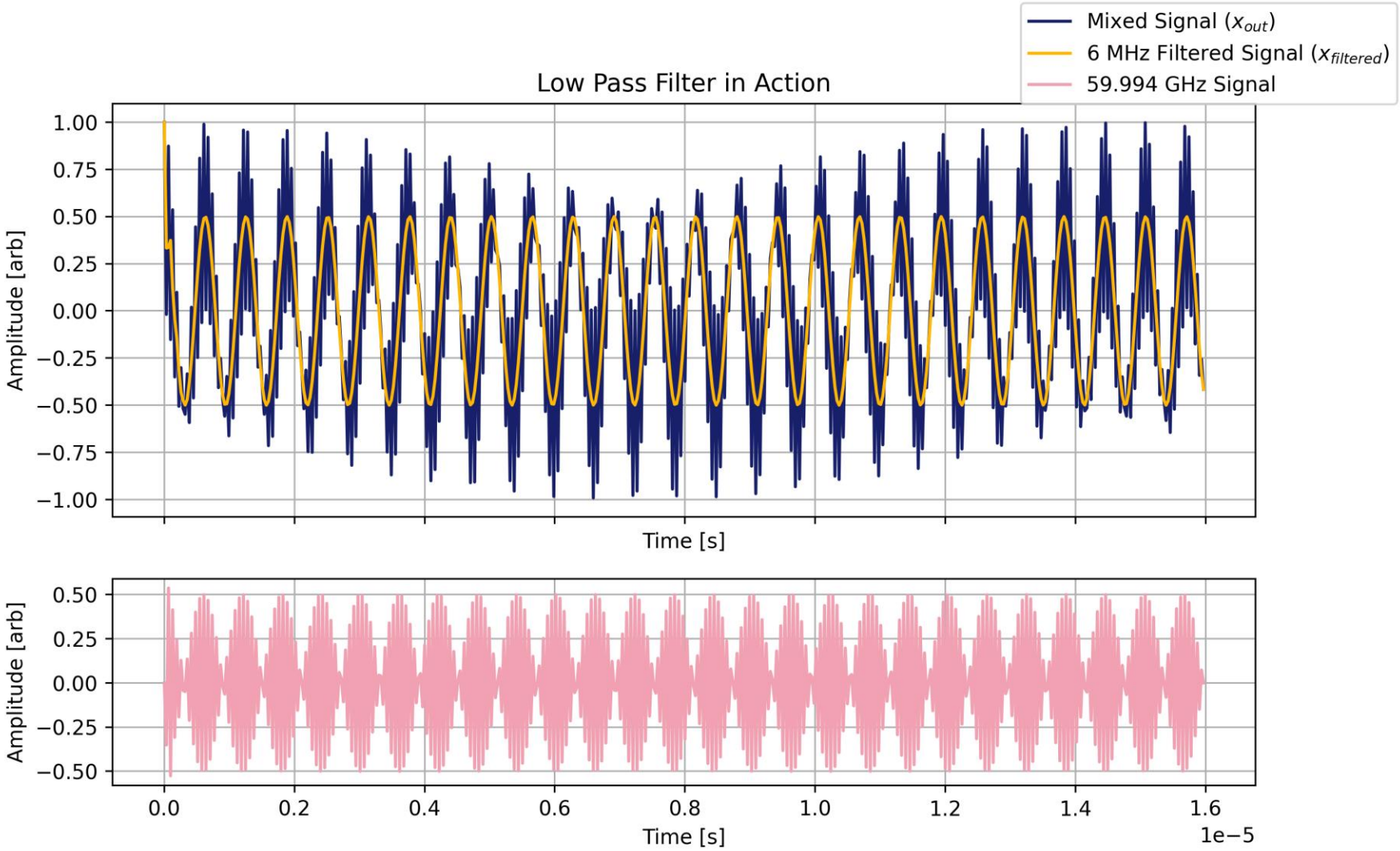
Basic Example



Using our simulation



Using our simulation



The maths of downmixing (con't)

- Dark Blue signal:

$$x_{out} = \frac{AB}{2}[\cos(2\pi * 0.006\text{GHz} * t) + \cos(2\pi * 59.994\text{GHz} * t)]$$

- We then apply a low pass filter to cut off the summed frequency, leaving us with:

$$x_{filtered} = \frac{AB}{2}\cos(2\pi * 0.006\text{GHz} * t)$$

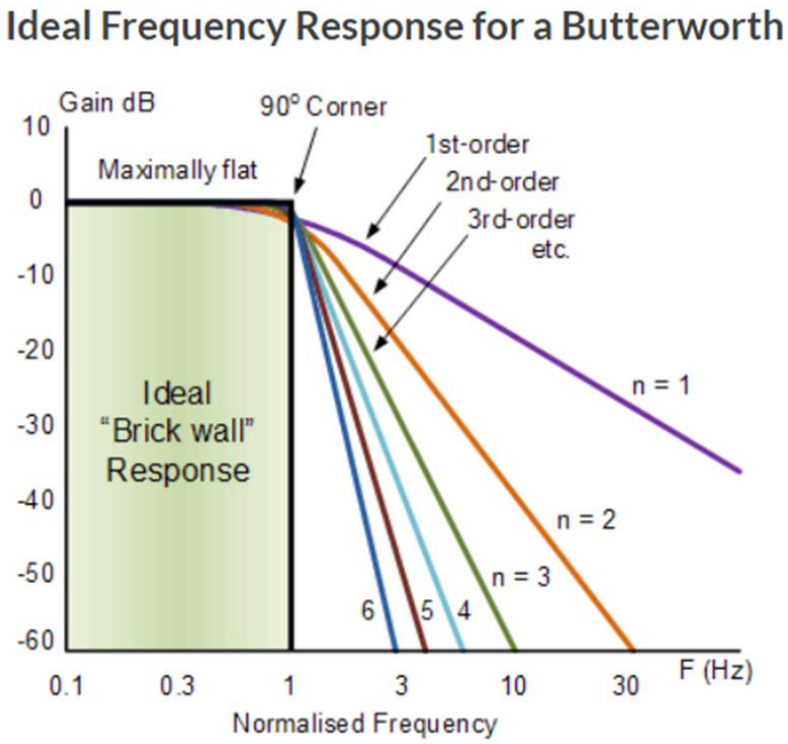
Why do we downmix?

- To directly sample GHz signals
  - Incredibly high sampling rate is needed
  - Enormous data storage
  - Excessive computational power

- Hardware Limitations
  - At higher frequencies, hardware becomes expensive and difficult to build
  - At lower frequencies, hardware is cheap and high quality

Filters

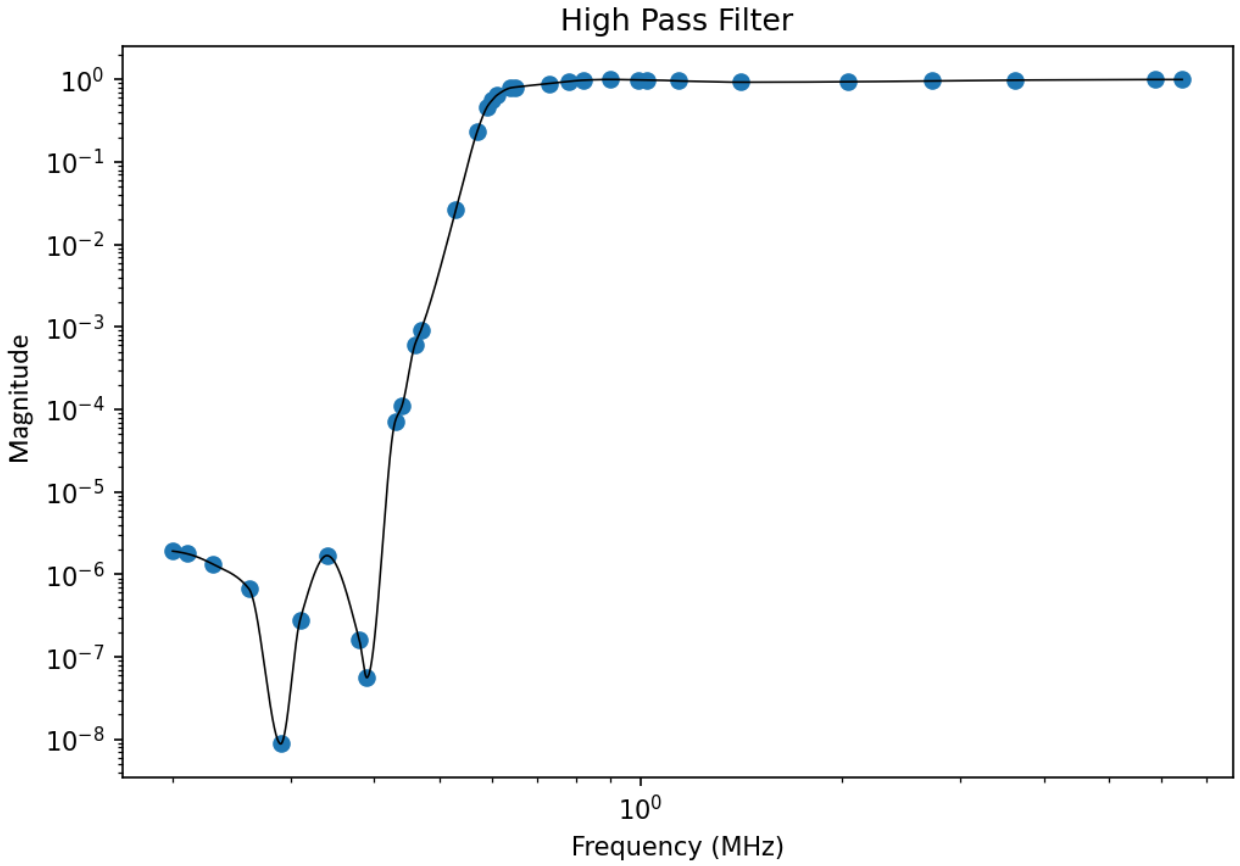
- Reduce signal after a given frequency
- Butterworth filter is the most common type
- For a “perfect” filter, n tend to infinity

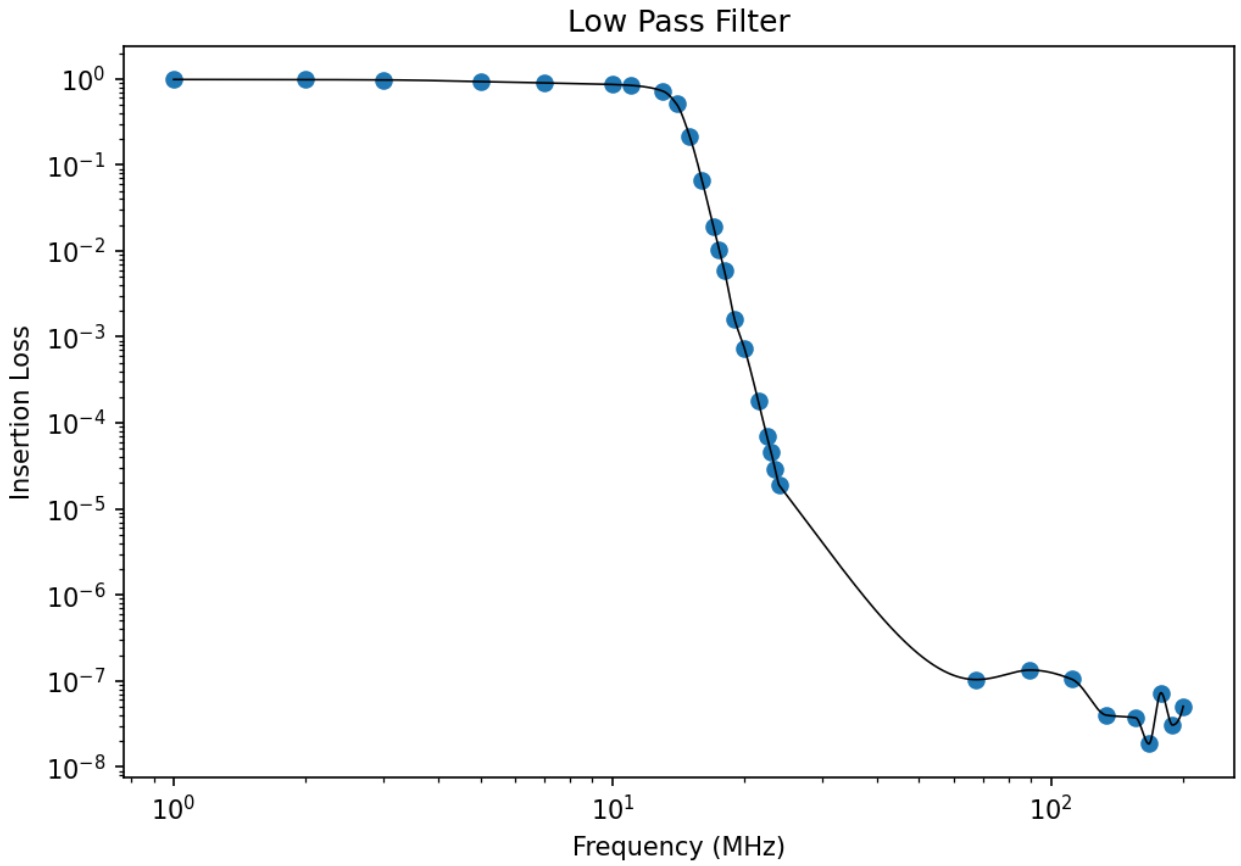


Types of filter

- Low pass filter
  - Cuts off unwanted high frequencies from mixing (that 59.994GHz component)
- Low pass filter
  - Cuts off unwanted low frequencies from the mains and radio sources

Our filters (from datasheet)





Signal Scaling

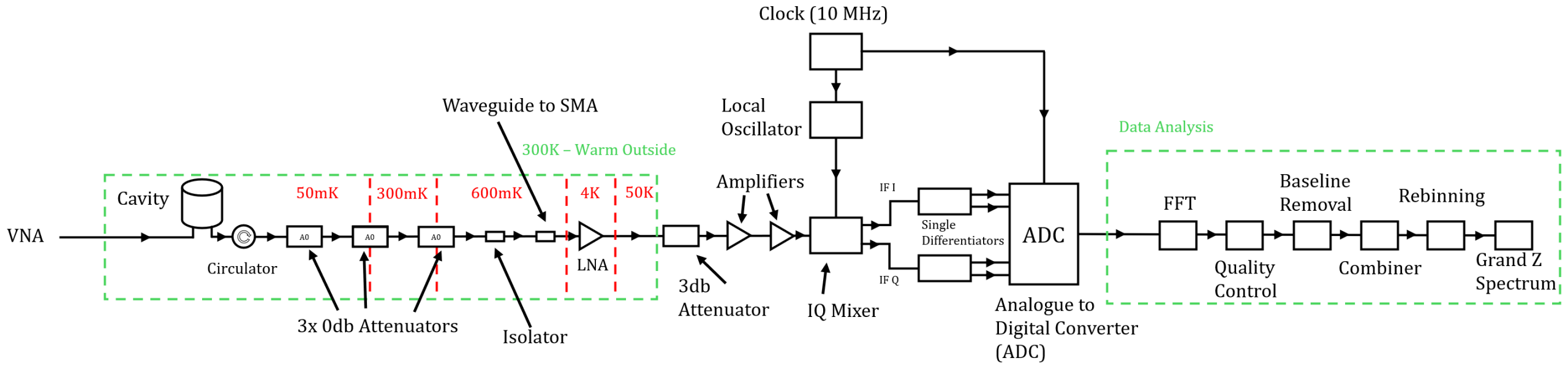
Power Scaling of the signal in accordance to the formula:

$$P_{\mathrm{sig}} = 1.79 \times 10^{-21} \left(\frac{V}{200}\right) \left(\frac{B}{7.6}\right)^2 C \left(\frac{g_y}{0.97}\right)^2 \left(\frac{\rho_a}{0.45}\right) \left(\frac{\nu_a}{750 \times 10^6}\right) \left(\frac{Q}{70000}\right)$$

- Parameters on the left
- Scaling from Amplifiers = 98dB

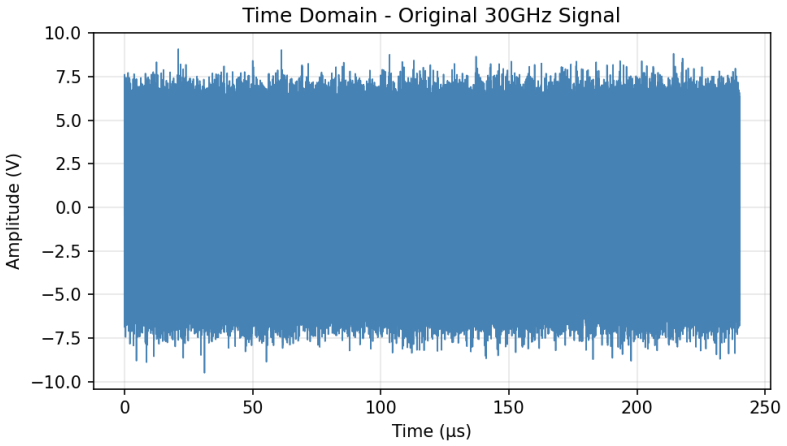
Parameter	Symbol	Value
Cavity volume	V	0.005383 L
Magnetic field	B	3 T
Form factor	C	0.69
Axion coupling (CAST benchmark)	$g_{a\gamma\gamma}^{\mathrm{CAST}}$	4×10^{-10}
Axion coupling (KSVZ benchmark)	$g_{a\gamma\gamma}^{\mathrm{KSVZ}}$	4.9×10^{-14}
Local axion density	ρ_a	0.45 GeV cm^{-3}
Axion frequency	ν_a	30 GHz
Quality factor	Q	2000
Impedance	Z	50 Ω
Integration time	t_{int}	1.6 ms
Sampling frequency	f_s	62.5 MHz
Nyquist frequency	f_{Nyq}	31.25 MHz
Intermediate frequency after mixing	f_{IF}	6 MHz
Total amplifier gain	G_{tot}	98 dB

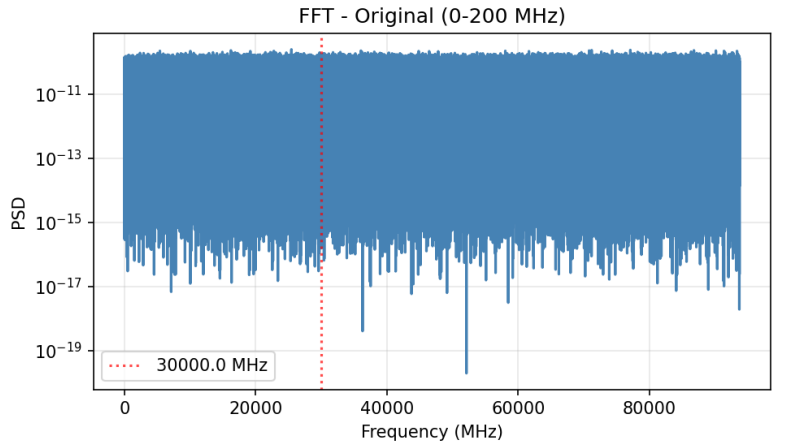
Experiment Pipeline



Voltage Graphs

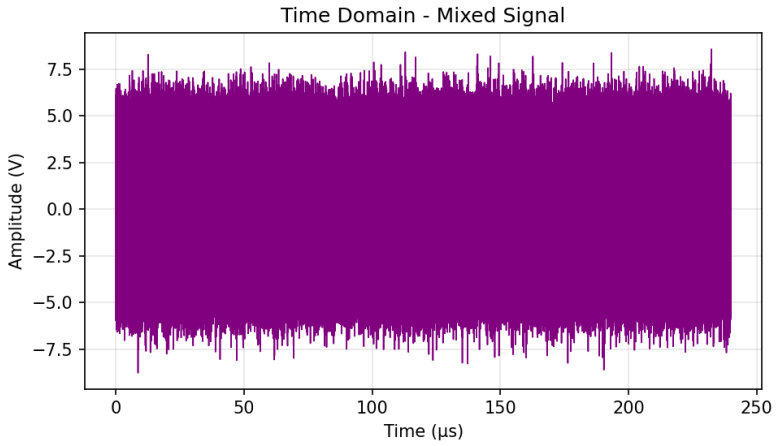
Raw Signal (Wave packets + Noise)

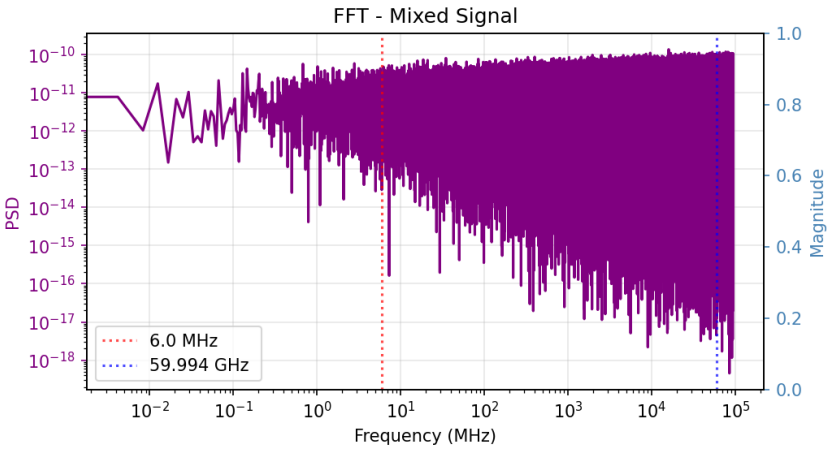




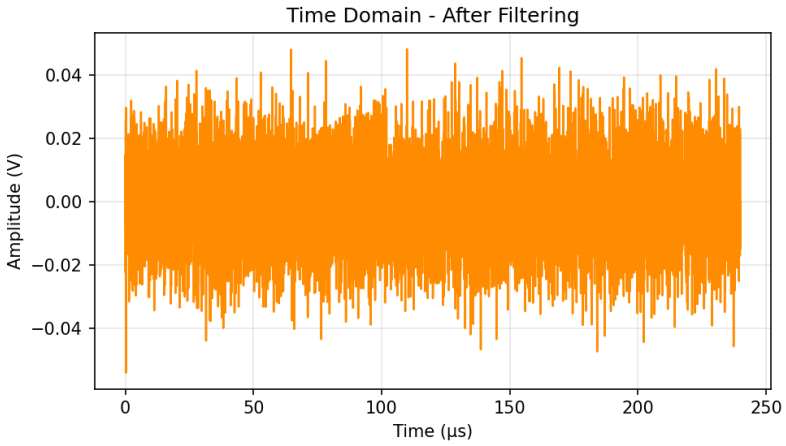
Raw Signal Mixed with Local Oscillator

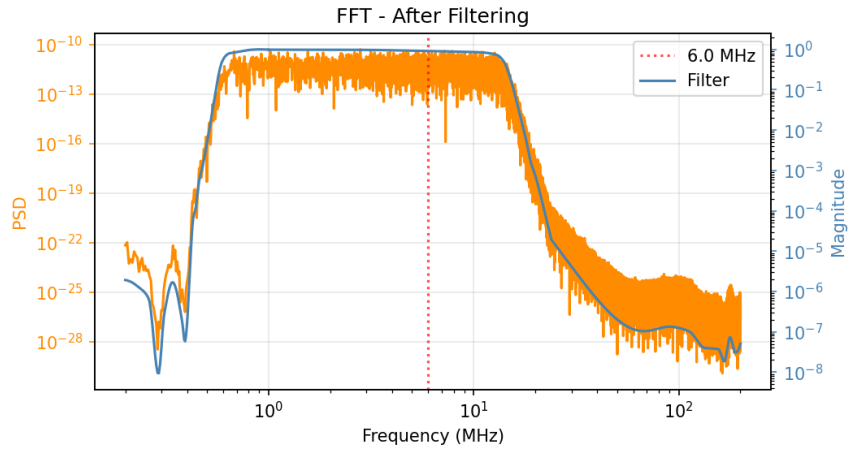
Isolating 6.0 MHz Component with 45000000 bins





Mixed Signal passed through Filters





Next Steps

- I need to pass my MPhys interview
- Need to integrate this simulation with the rest of the data analysis
- Need to optimise the code
  - For the full 1.6ms integration time we currently are aiming for, this generates many GBs of data, which takes a long time to generate and manipulate