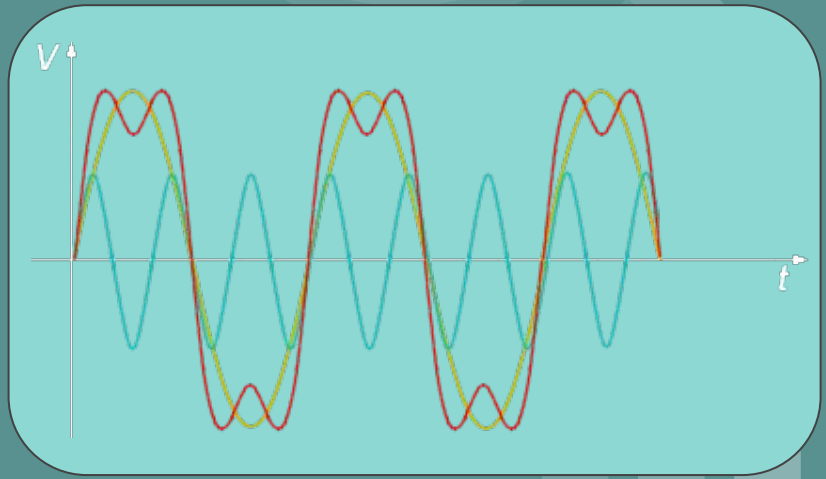
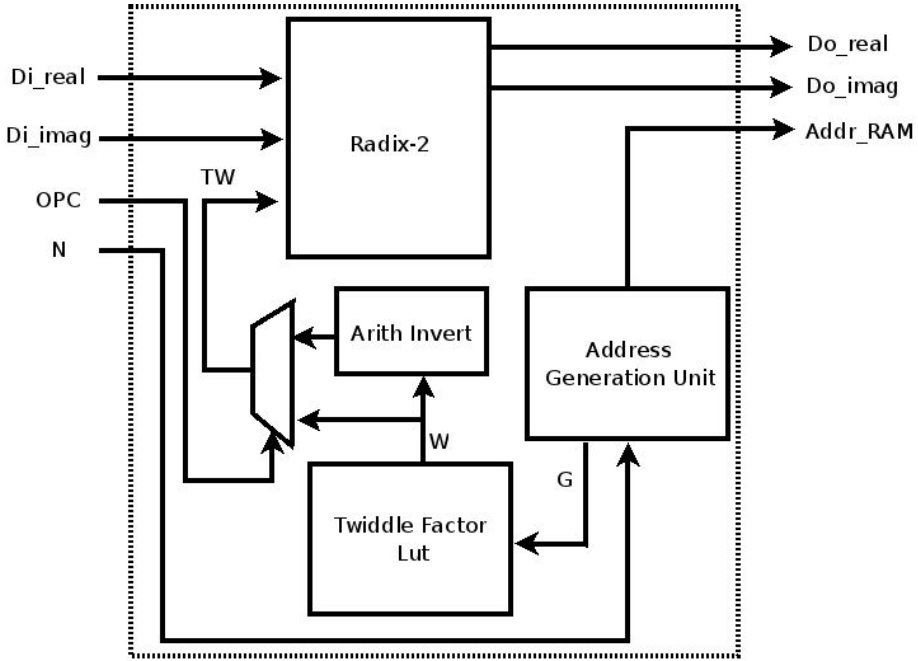
Fast Fourier Transform (FFT) - Hardware Implementation

Abdul Muizz



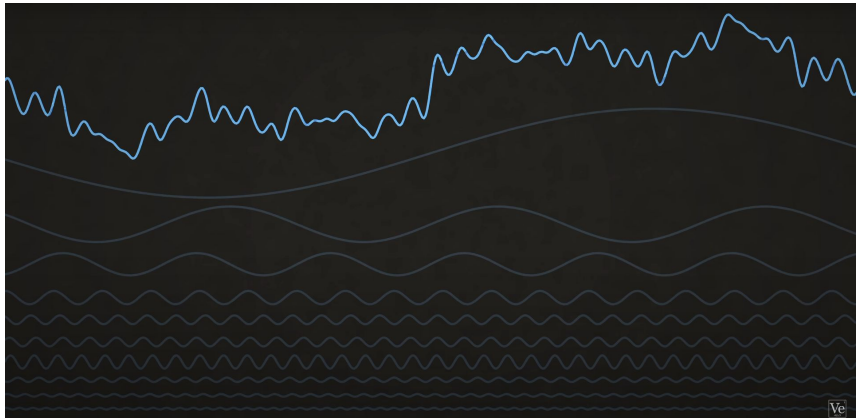
Scope and Background

- I wanted to implement a design for a Fast Fourier Transform (FFT). **Why?**
- FFT algorithm is critical to analyzing waveform frequency composition
- Critical in modern communications, imaging, radar, SDR, seismology, vibration, etc.
- Hardware implementation for FFT is crucial in these areas.



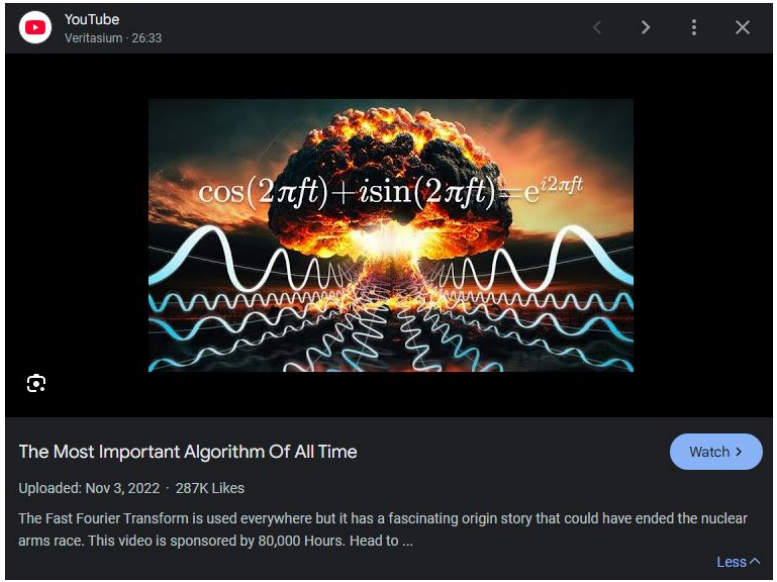
What is the Fourier Transform

- Officially: Any periodic/finite-support signal = sum of complex sinusoids.
- The Fourier Transform process involves taking in an input waveform and decomposing
- Output what sine/cosine frequencies it’s composed of.



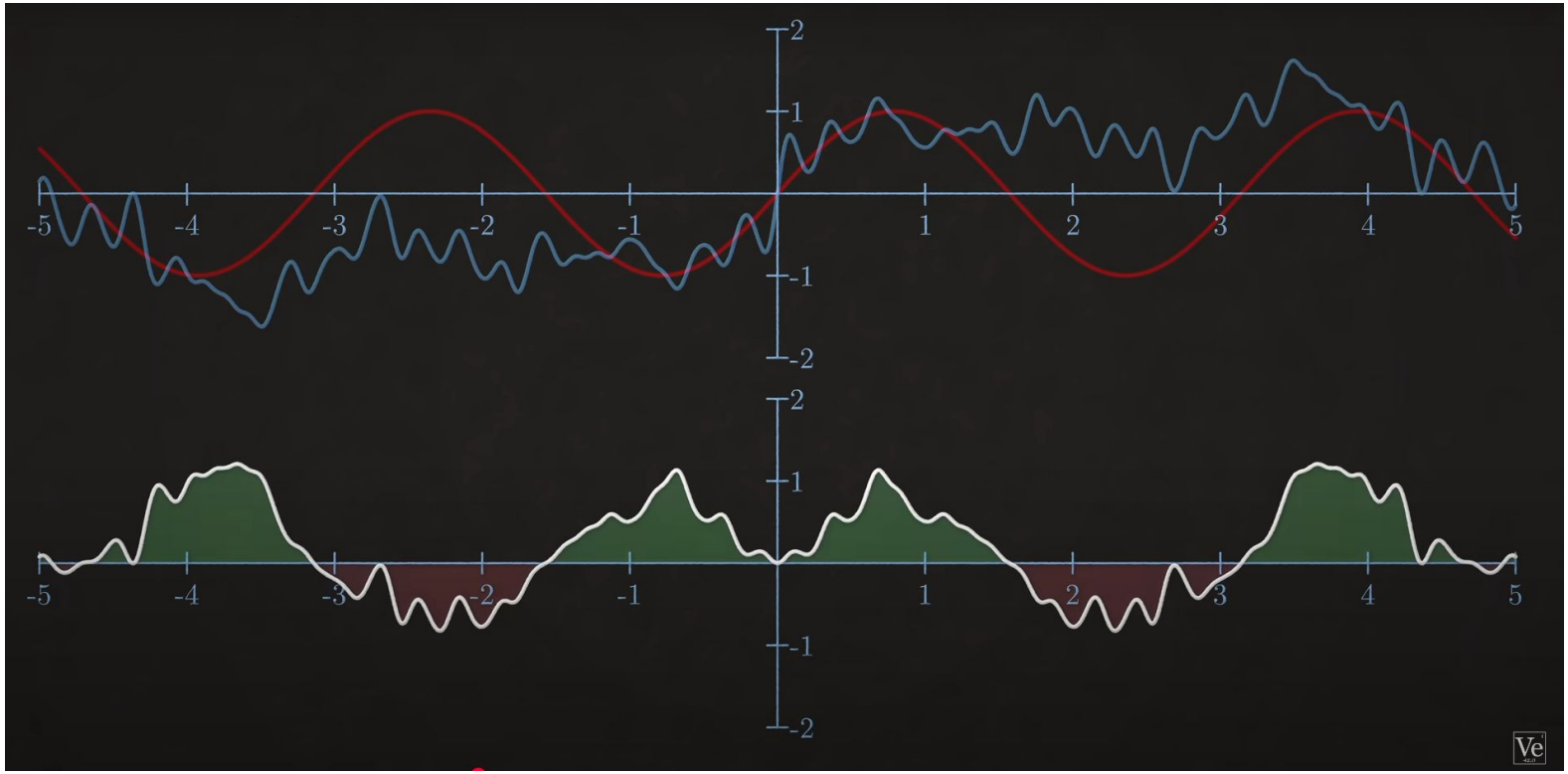
Some Notable Examples

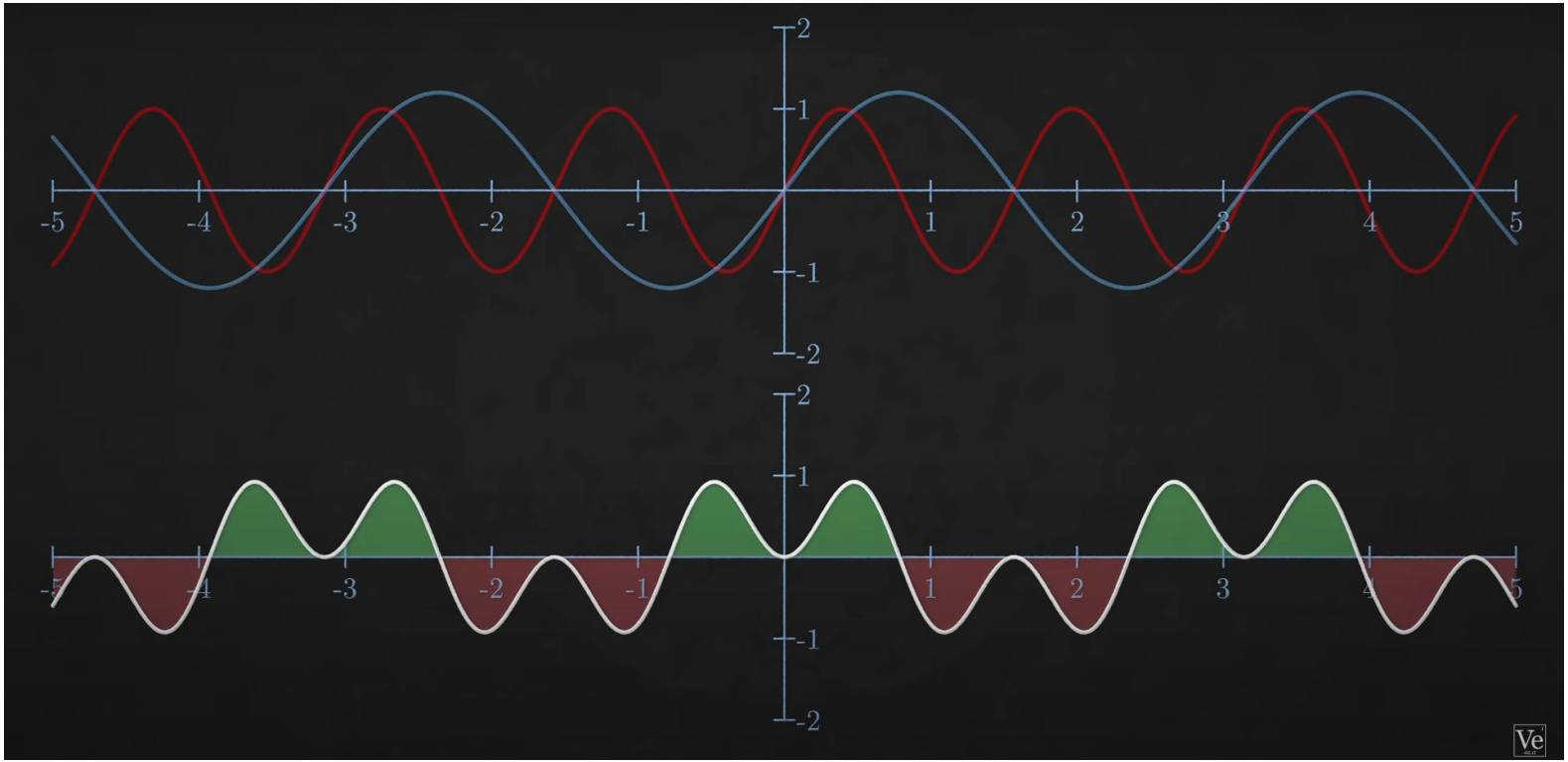
Seismology and Nuclear Arms Race

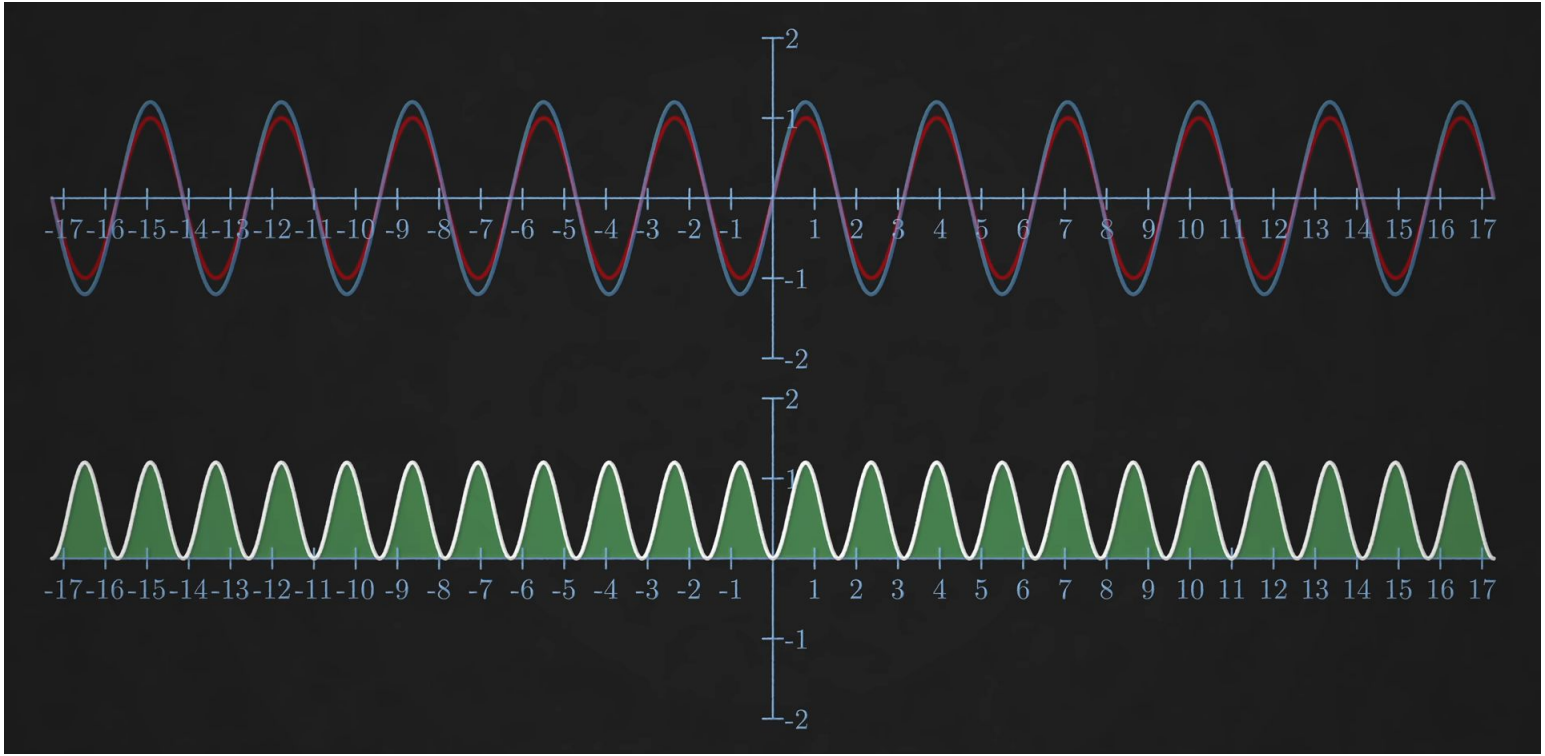


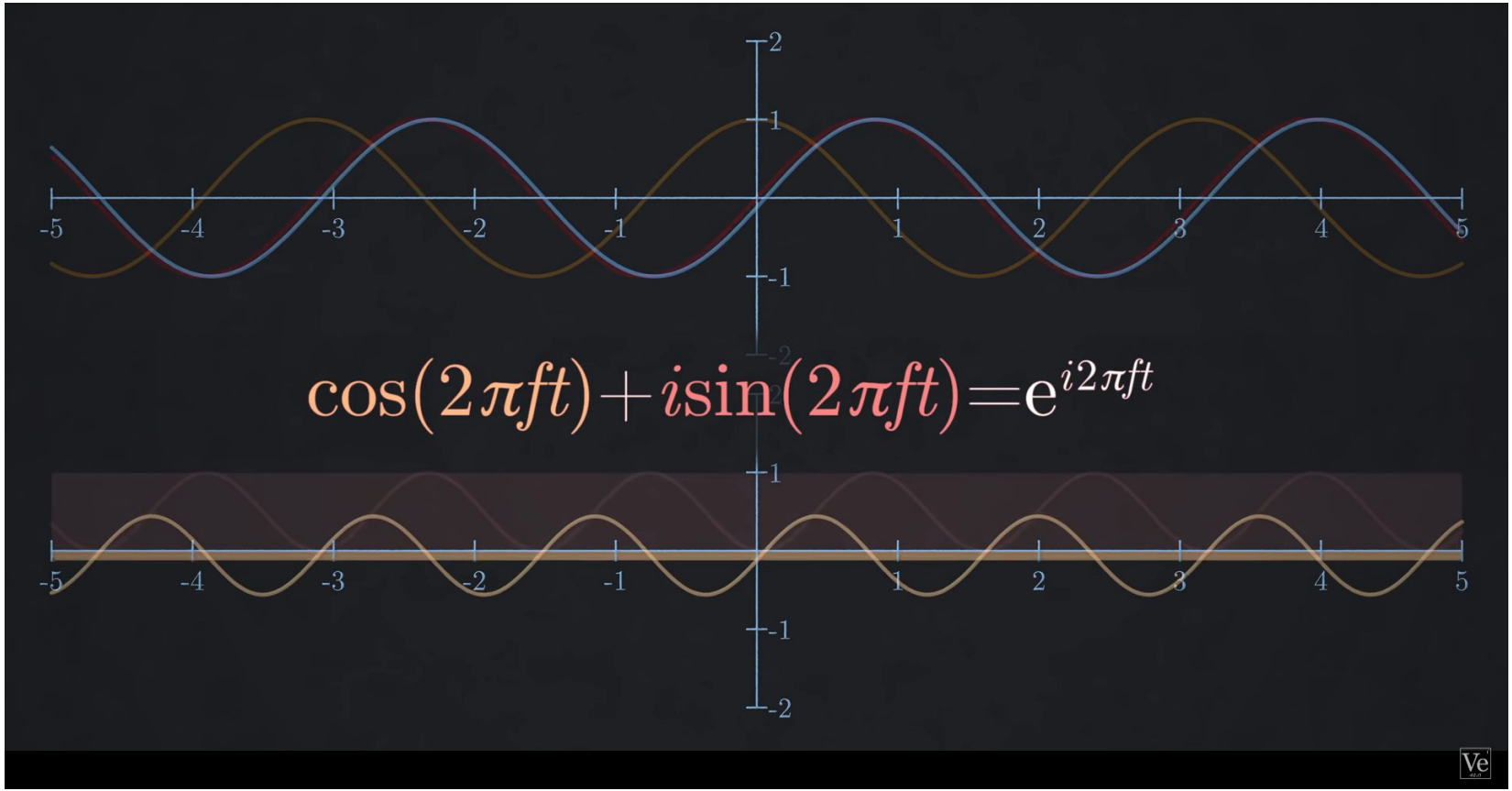
https://www.youtube.com/watch?v=nmgFG7PUHfo&t=535s&ab_channel=Veritasium

Motor Vibration Analysis (Co-Op @ Irving Tissue)


2
1
-1
-2
-5
-4
-3
-2
-1
1
2
3
4
5
2
1
-1
-2
-5
-4
-3
-2
-1
1
2
3
4
5


2
1
-1
-2
-5
-4
-3
-2
-1
1
2
3
4
5


2
1
-1
-2
-17 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
2
1
-1
-2
-17 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17

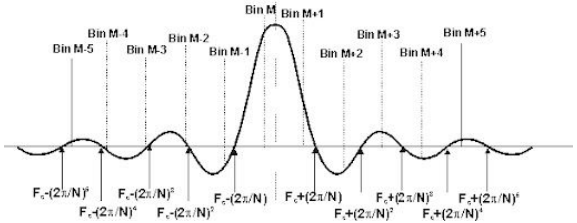

2
1
-1
-2
-5
-4
-3
-2
-1
1
2
3
4
5
$\cos(2\pi ft)+i\sin(2\pi ft)=e^{i2\pi ft}$
1
-1
-2
-5
-4
-3
-2
-1
1
2
3
4
5

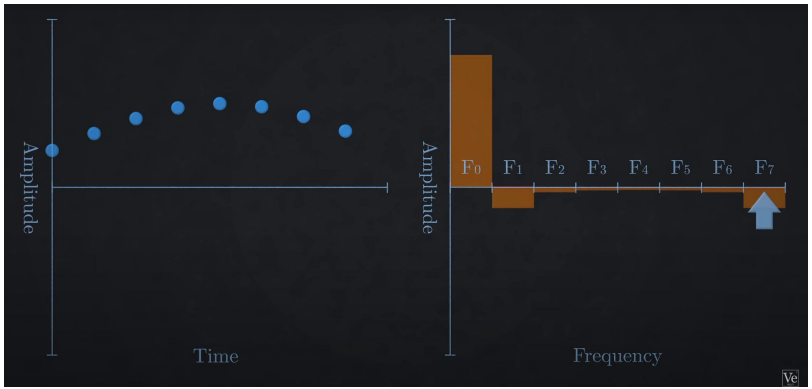
Discrete Fourier Transform (DFT)

- In computation, we can't sample a continuous waveform like this. Real world signals are finite.
- Instead we can use a discrete function:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}$$

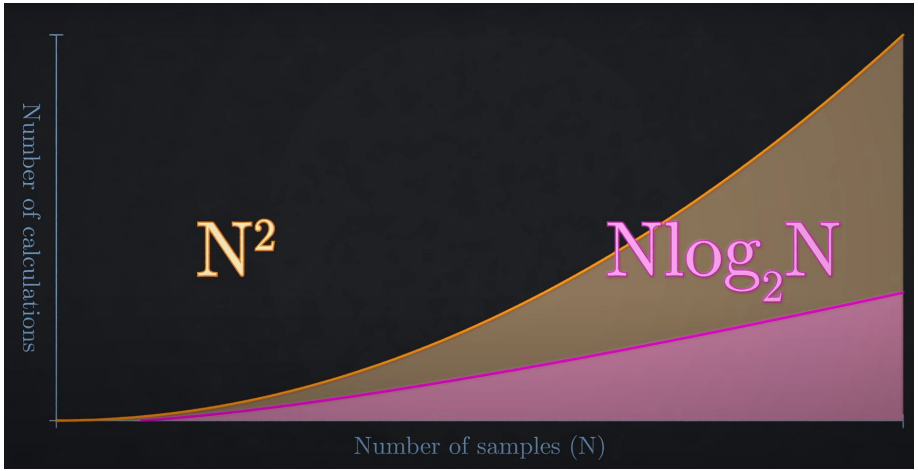
- One output bin needs N complex MACs -> $O(N^2)$ in total. (Based on # of samples)





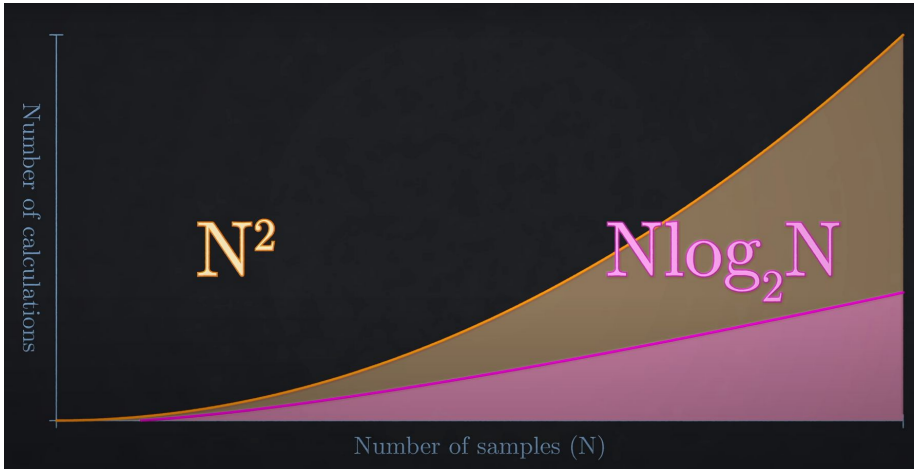
It is slow

- If you had 1 million samples, it would take 1 trillion calculations for a DFT
- For 1960's computers, that would take over three years.



Fast Fourier Transform (FFT)

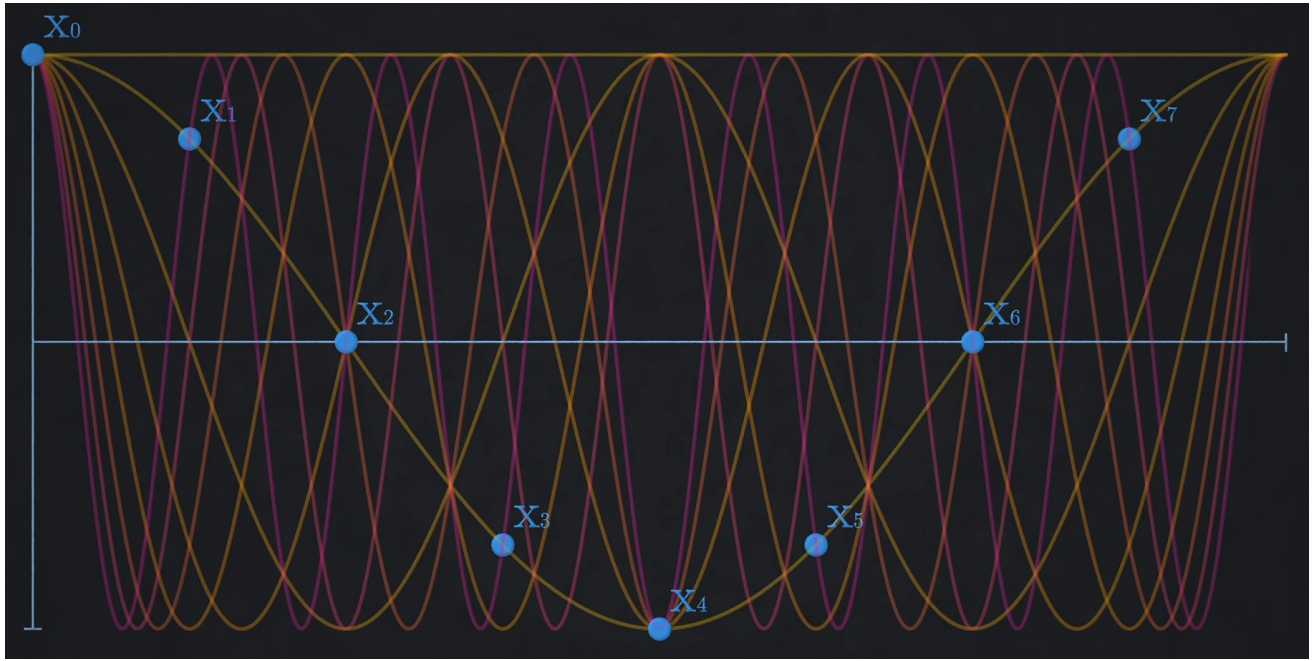
- In 1963, John Tukey showed a fellow scientist (Richard Garwin) a faster way to do these transforms.
- That same 3-year computation would take only 35 minutes.



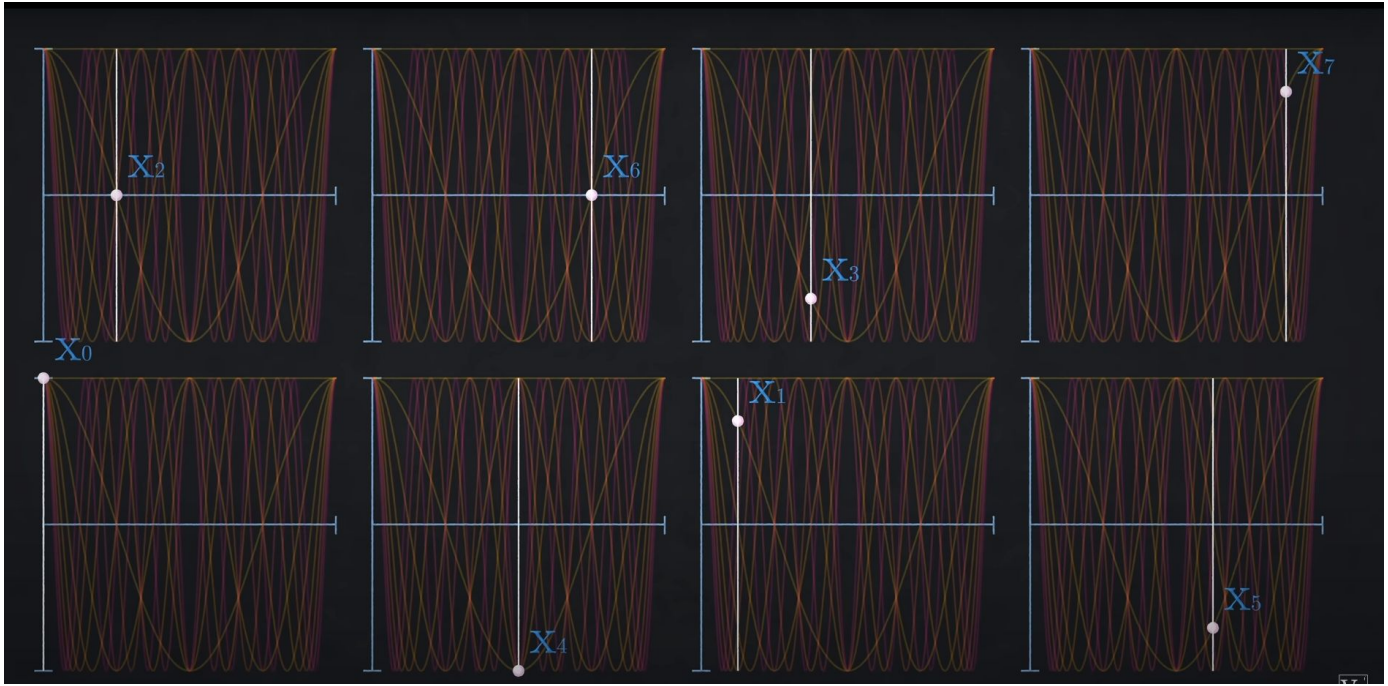
Why is it more efficient?

Hint: It's not because it has "fast" in the name

How Fast FFT Works

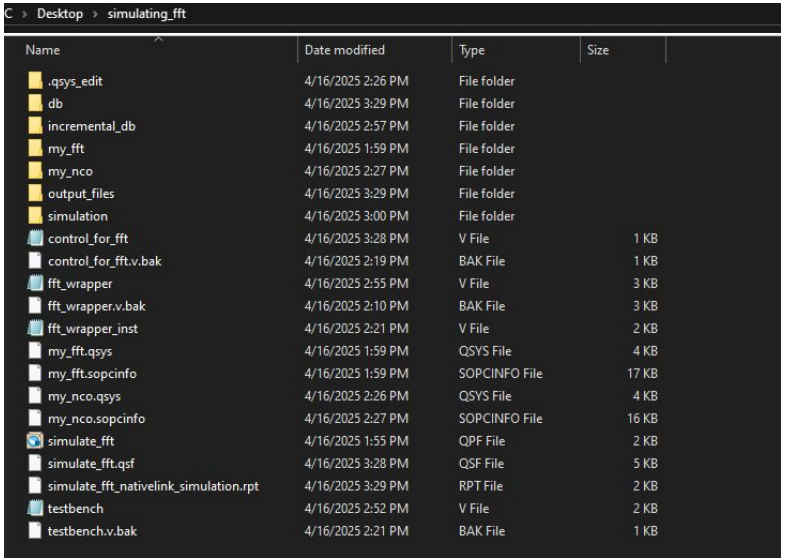


How Fast FFT Works

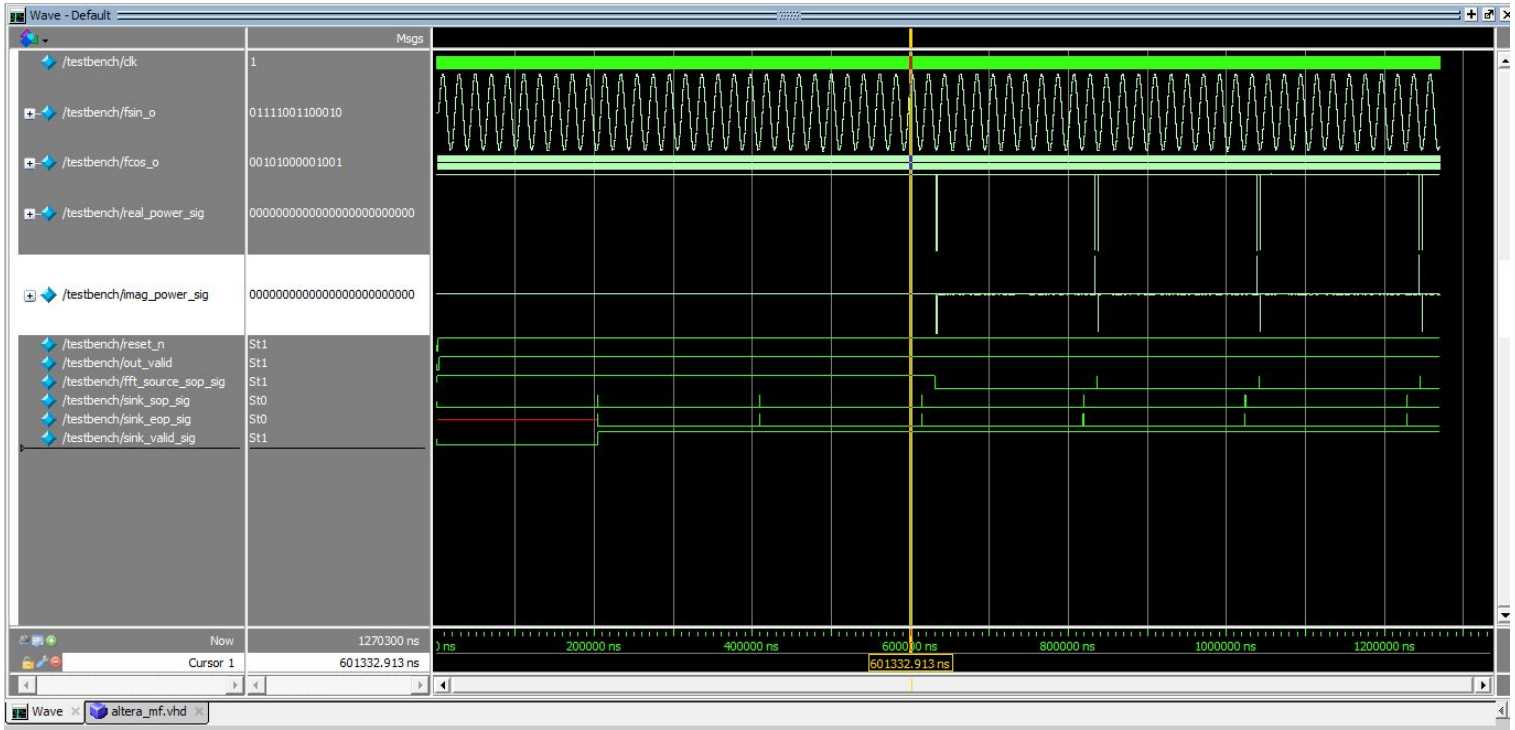


My Implementation

- Made with Intel Quartus, tested in ModelSim
- 40 MHz Sample clock measuring a 32-bit NCO front-end
- 1024-pt (bins) 14-bit bins.
- FFT MegaFunction from Visual Electric (computation) plus a Control Wrapper
- For my input sine wave I passed in a clean 390.625 KHz signal.

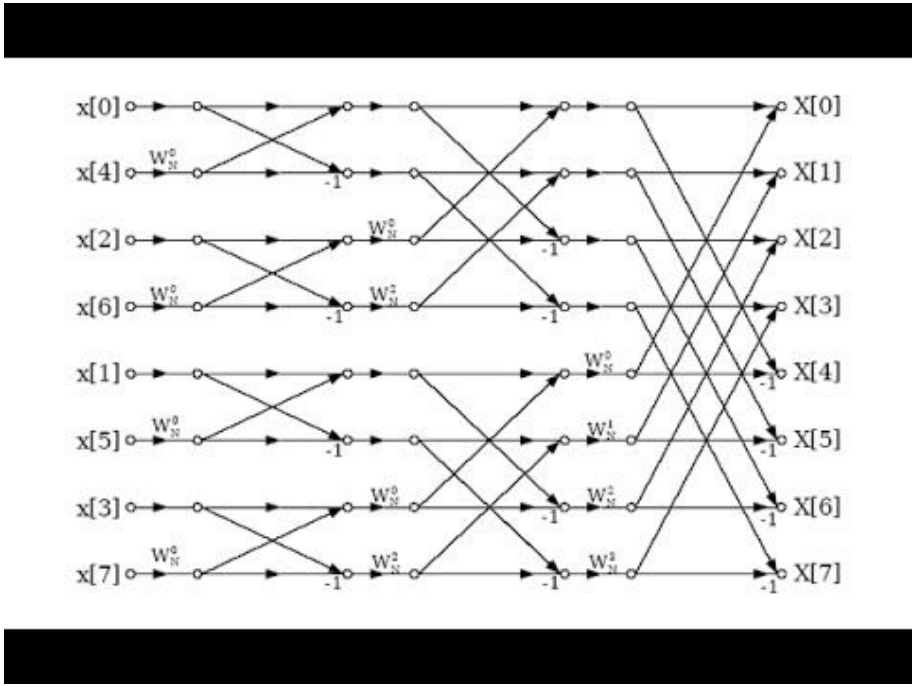


Resulting Output



Conclusions

- Speeding up an algorithm can sometimes be more elegant than adding pipelining or superscaling.
- In this case, intuitive understanding of the math allowed us to heavily reduce the number of calculations needed.
- FFT will reduce the calculation input delay for the given signal processing.
- Look into Radix-2 FFT



References

[1] Visual Electric, “FFT development on an FPGA - Simulation Design Flow using Quartus and Verilog (no board required).,” *YouTube*, Jul. 12, 2020. https://www.youtube.com/watch?v=DgRVqS4Dw9g (accessed Apr. 17, 2025).

[2] Visual Electric, “Building an FPGA-based Spectrum Analyzer with GUI control: PART 1: Design Scope,” *YouTube*, Jun. 13, 2020. https://www.youtube.com/watch?v=HGZWxZKRcCg (accessed Apr. 17, 2025).

[3] Visual Electric, “Setting up and testing the FFT MegaFunction in Quartus (Part 2 of FPGA Spectrum Analyzer design),” *YouTube*, Jul. 12, 2020. https://www.youtube.com/watch?v=BtTNeQszSJo (accessed Apr. 17, 2025).

[4] Veritasium, “The Algorithm That Transformed The World,” *www.youtube.com*, Nov. 03, 2022. https://www.youtube.com/watch?v=nmgFG7PUHfo

[5] J. W. Cooley and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Stanford*, 1965. https://web.stanford.edu/class/cme324/classics/cooley-tukey.pdf

[6] “FFT Intel® FPGA IP Core,” *Intel*, 2025. https://www.intel.com/content/www/us/en/products/details/fpga/intellectual-property/dsp/fft.html (accessed Apr. 17, 2025).

Questions?

Thank you for listening!