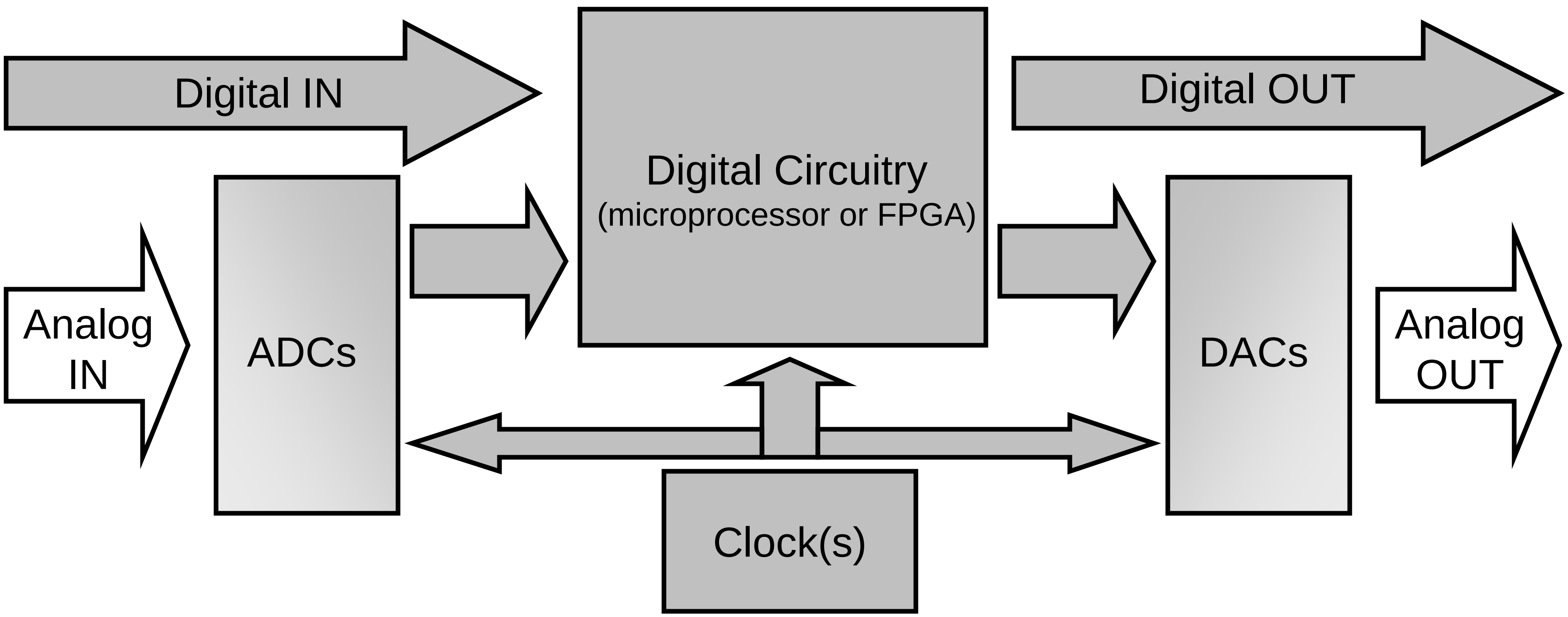
# Introduction to DSP
## or
## What is Digital Signal Processing?

### *Outline*

1. Basic DSP Architecture
2. Applications
3. Basic Algorithms
    a. FFT.
    b. Perfect mixer … divider.
    c. Digital filters (FIR and IIR filters).

# Basic DSP Architecture



# DSP applications

## *DSP Advantages:*

- DSP is frequently **cheaper (in money and time)** than an equivalent analog circuit (especially circuit development).
- Analog electronics is hard … but programming is **easier**.
- **Sometimes DSP is the only option:** DSP circuits can do certain operations that are not possible with analog circuitry.

## *A few DSP applications:*

- ***Long distance communication:*** digital communication has lower noise and can be compressed.
- ***Voice and image recognition:*** Use complex recognition algorithms.
- ***"Software" Radio:*** DSP can efficiently demodulate a signal from its RF carrier.

# DSP in Physics

DSP is relatively young and has not yet made broad in roads into physics instrumentation. There are a few outstanding examples:

- **Complex coincidence triggering** in particle physics and quantum optics.

- **Custom feedback loops:** When PID feedback just isn't good enough, a DSP-taylored feedback loop gain give you some extra feedback bandwidth.

- **DSP lock-in amplifiers:** Very stable and accurate, high dynamic range, and stable long-term integration.

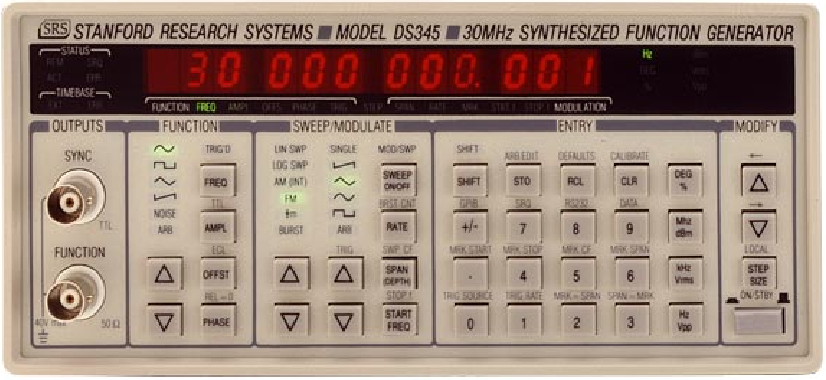
- **Synthesized signal generator:** these devices generate their output signals using DSP.

  → arbitrary waveforms possible.

  → Very quick frequency changes.

  → Phase continuous frequency changes!

  → Extremely stable.



# When should you use DSP ?

You should use DSP when …

… you know that you could solve your circuit design by processing your signals with a computer, i.e. Maple, MatLab, Mathematica, C/C++, etc ...

→ DSP just replaces the computer with a dedicated microprocessor or an FPGA.

→ DSP is faster, cheaper, and more stable than a desktop computer.

# Basic DSP algorithms

- ***Fast Fourier Transform***
- ***Multiplication … division***
- ***Digital filters***

# FFT: Fast Fourier Transform

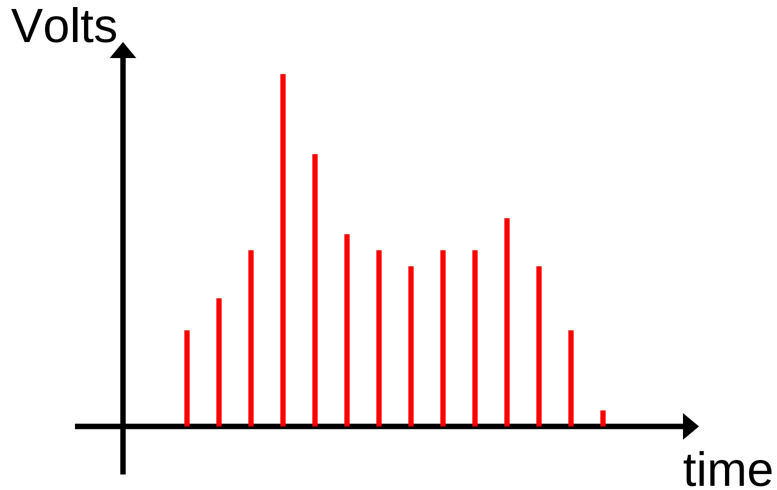
- Originally discovered by Gauss (~1805).
- Re-discovered by Cooley and Tukey (1965).
- Operates on a discrete set of N sampled values.
- Most FFT libraries require $N = 2^n$.
- Discrete fourier transform computation time ~ $N^2$.
- FFT computation time ~ $N \log_2(N)$.

  → FFT is cheaper and faster.

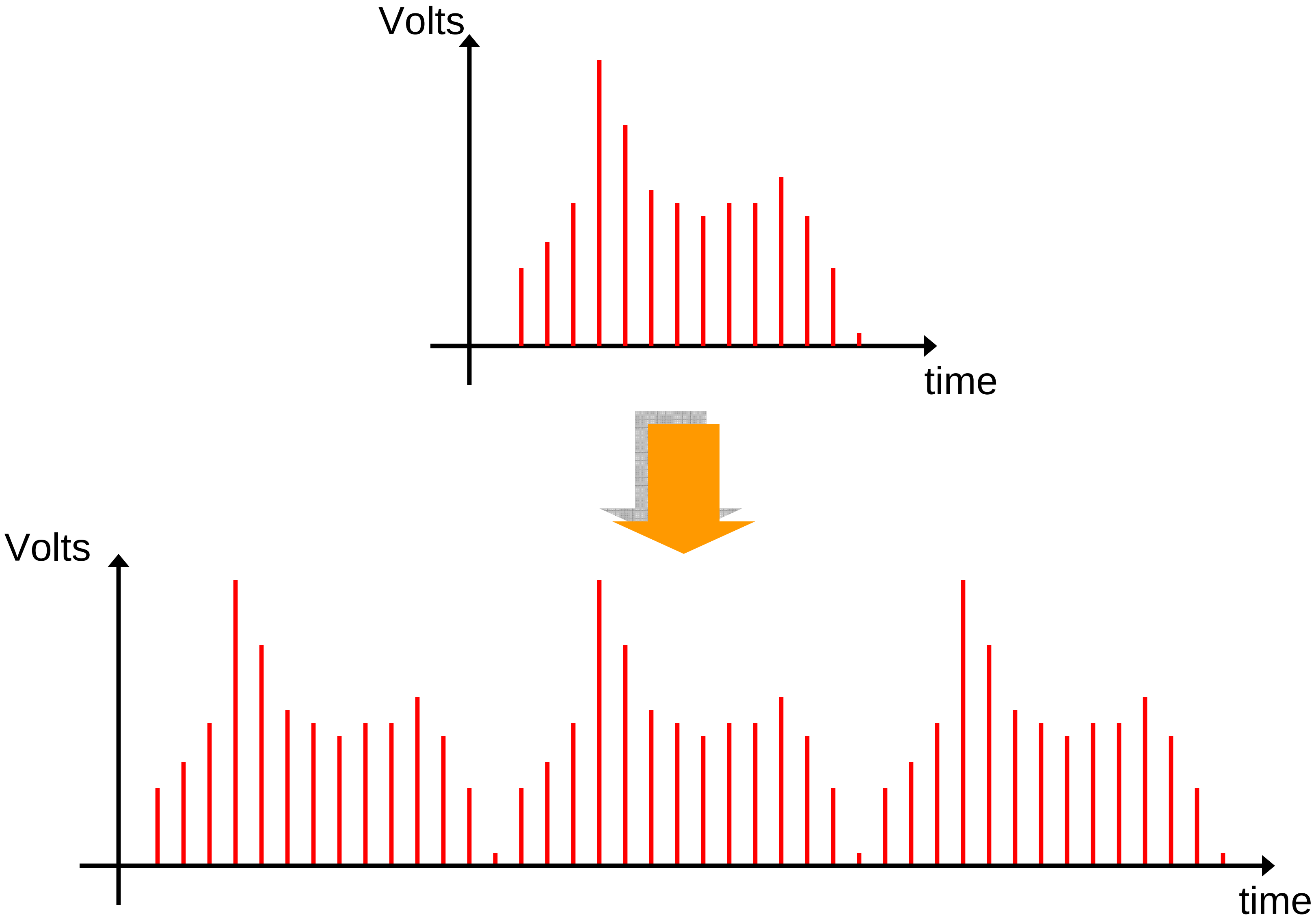
→ An FFT allows you to do DSP in frequency space.

→ The FFT is a standard programming library item for Microprocessors and FPGAs.

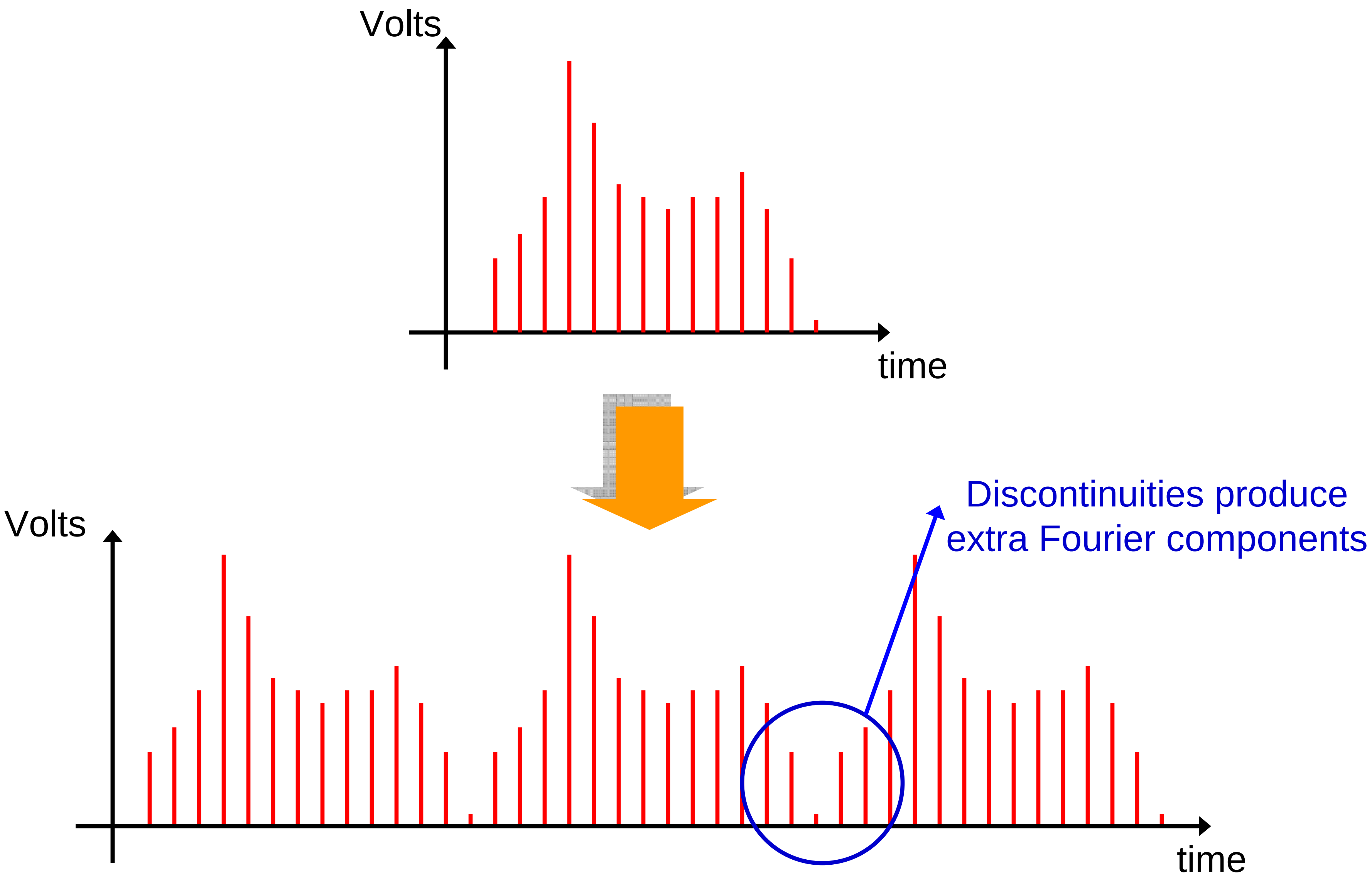
# Data Windowing (I)



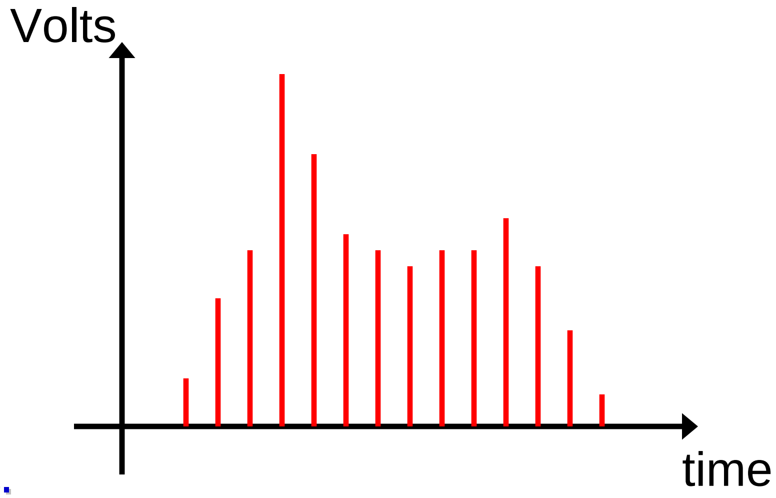
# Data Windowing (I)



# Data Windowing (I)

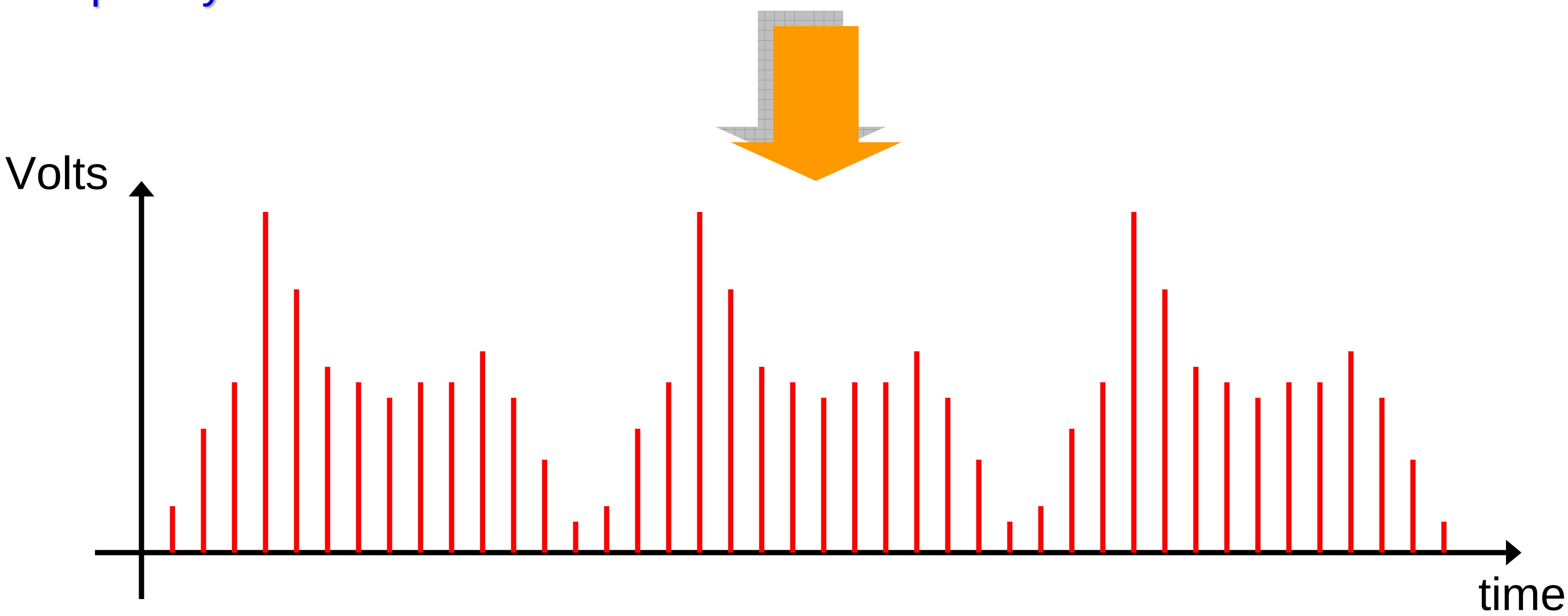


# Data Windowing (II)

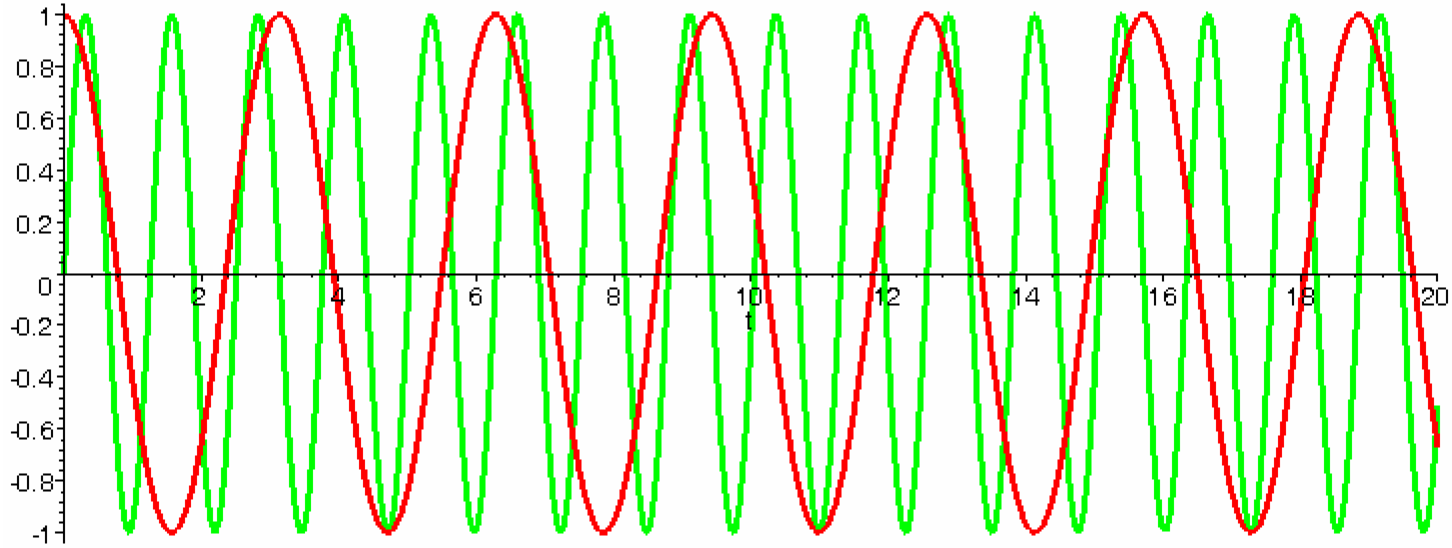


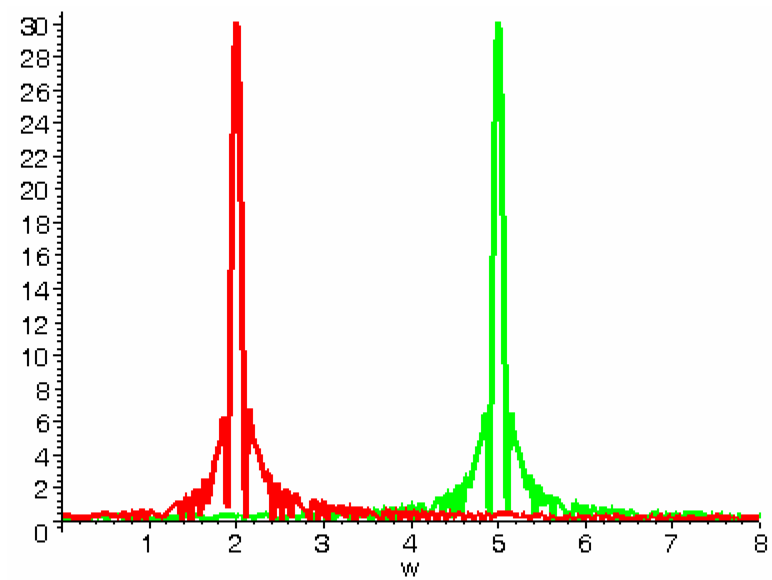

Smooth the data to ZERO at the edges.

→ Reduces spurious Frequency structures.



# Digital Mixing (I)

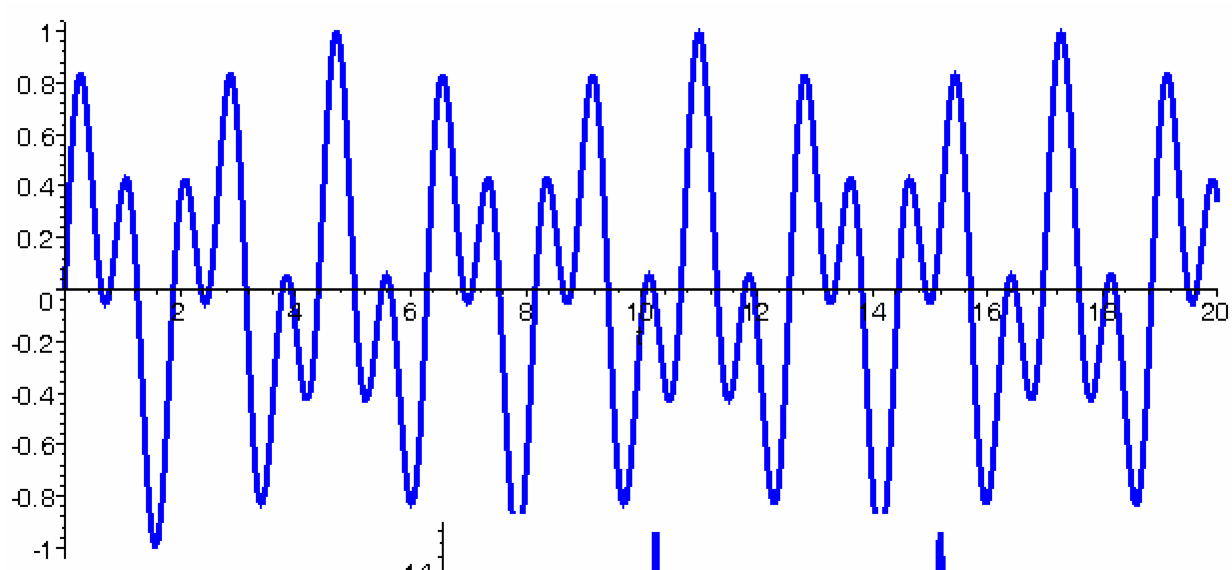


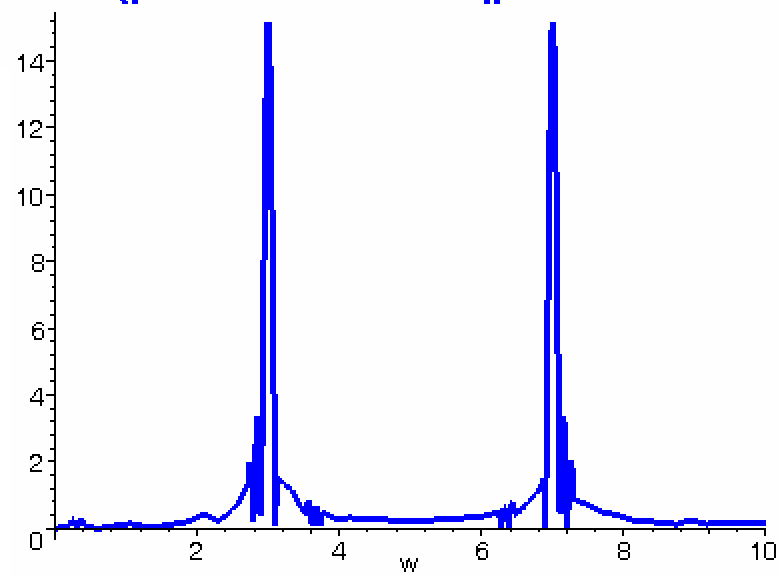

Fourier Transform

# Digital Mixing (II)

RED x GREEN = BLUE




Fourier Transform




Digital mixing is does not produce any extra harmonics, unlike analog mixers