Lab(3)

Faculty of Engineering,
Ain Shams University
Design of Measurement Systems, Sp. 2015

March 9, 2015

NATIONAL INSTRUMENTS
LabVIEW

LAB Goals:

Understanding:

- While and For Loops
- For Loop and 2D Arrays
- Shift Registers
- Case Structure
- Formula Node

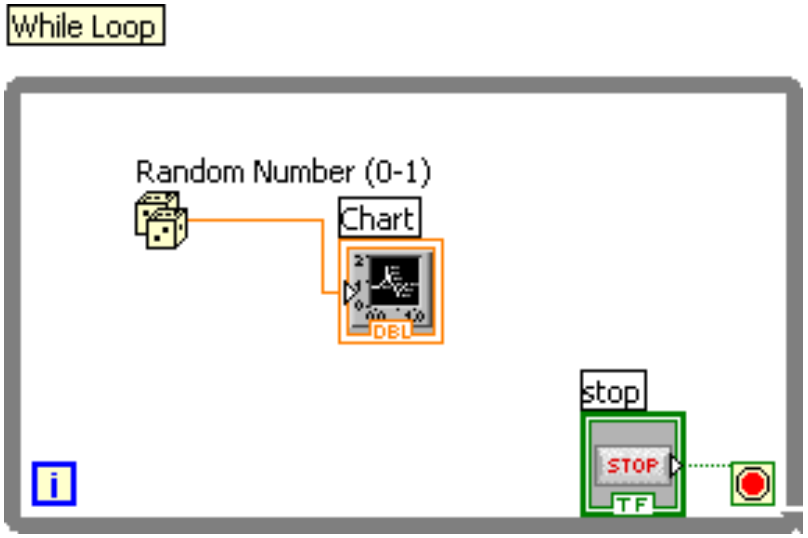
- Demonstrating different types of signals.
- Understanding FFT through various signal examples.
- Understanding different types of Filters through several set of examples.

Structures

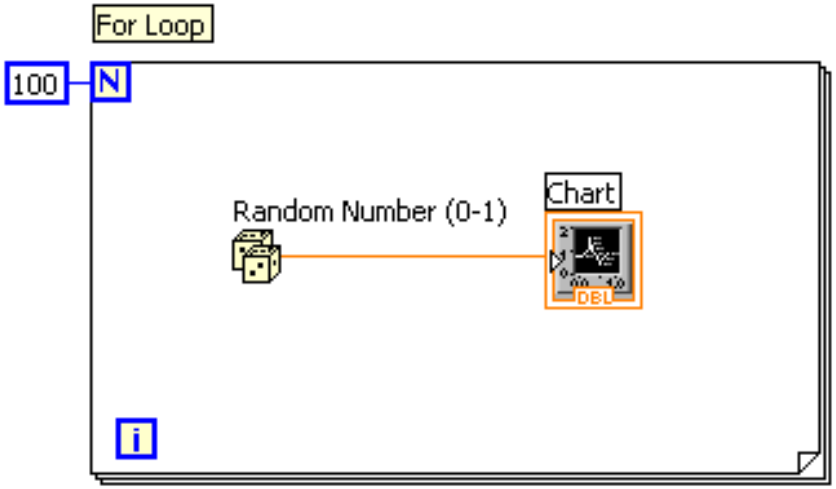
- While and For Loops
- For Loop and 2D Arrays
- Shift Registers
- Case Structure
- Formula Node

Loops

- While Loops
  - Have Iteration Terminal
  - Always Run at least Once
  - Run According to Conditional Terminal (Two different states)

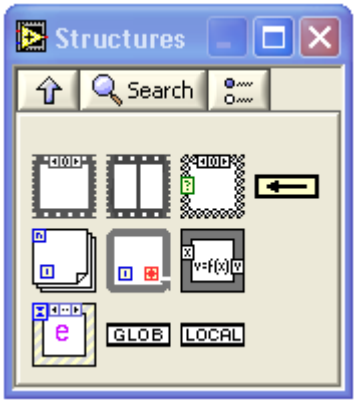


- For Loops
  - Have Iteration Terminal
  - Run According to input N of Count Terminal

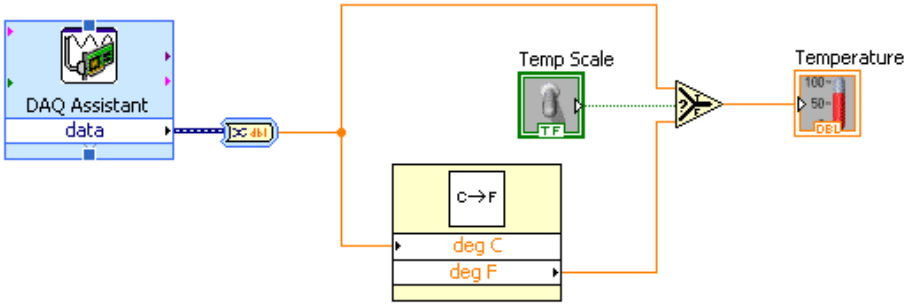


Loops (cont.)

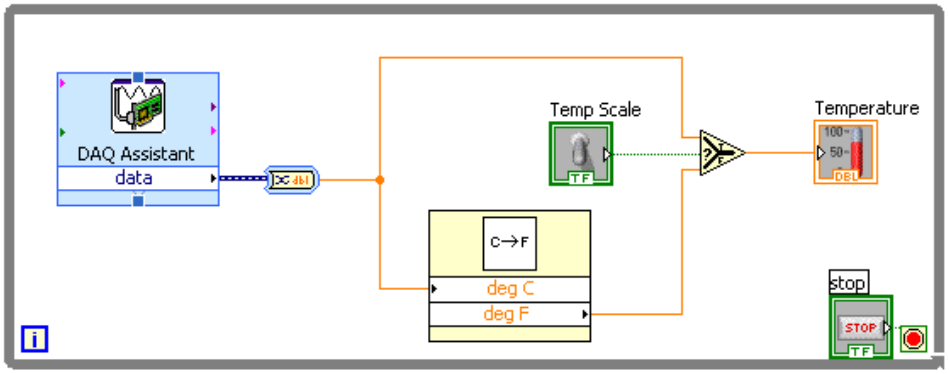
1. Select the loop



2. Enclose code to be repeated



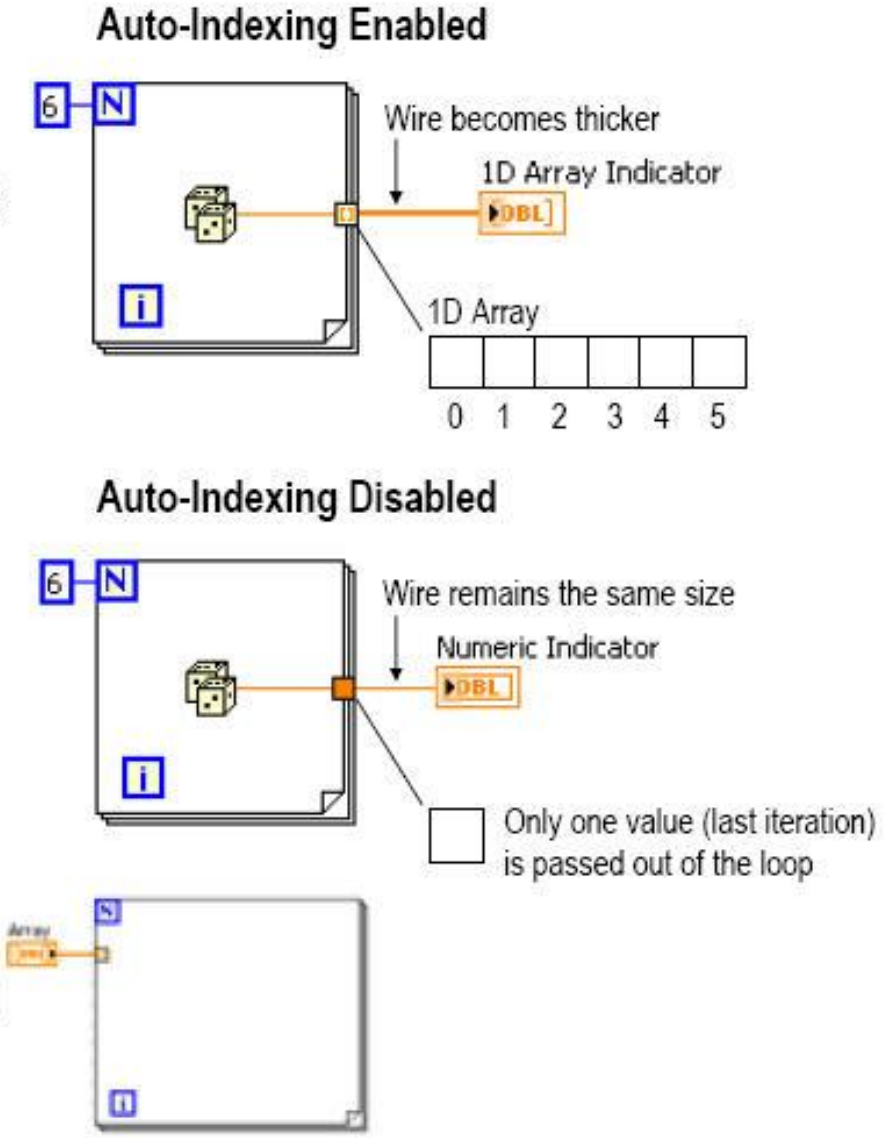
3. Drop or drag additional nodes and then wire



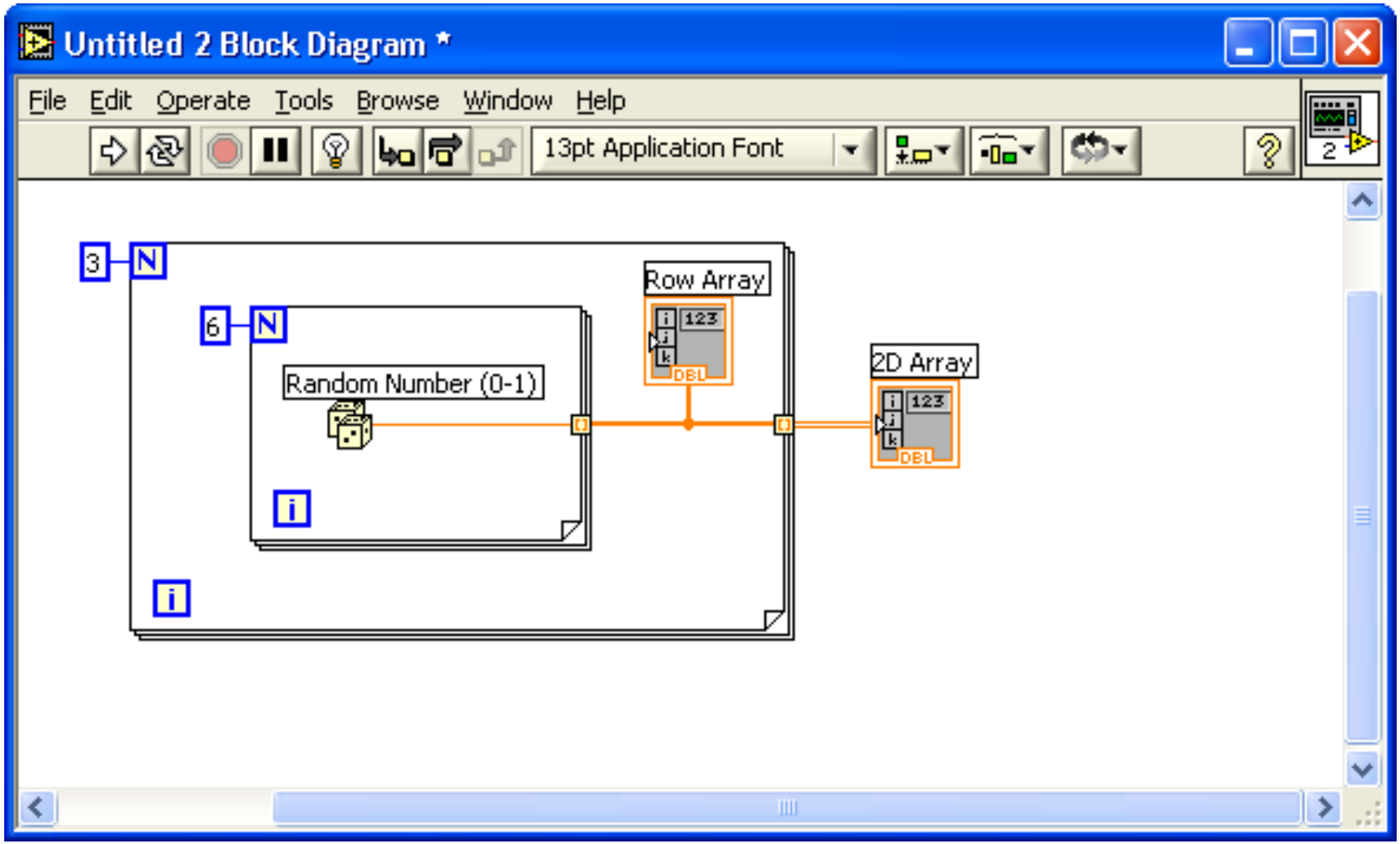
Auto-Indexing

- Loops can accumulate arrays at their boundaries with auto-indexing
- For Loops auto-index by default
- While Loops output the final value by default
- Right-click on tunnel and enable/disable auto-indexing

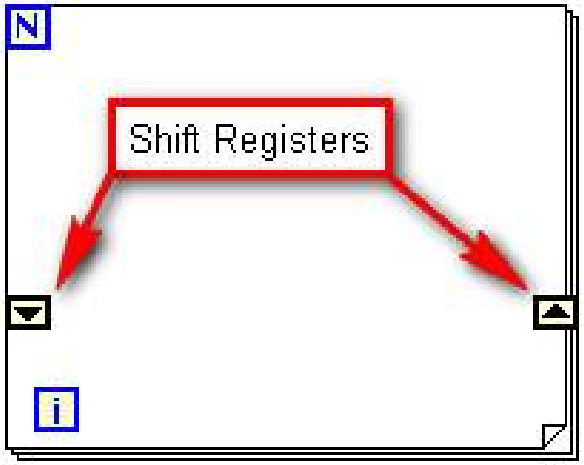
- Auto-indexing on Loop input converts arrays to indexed elements (default), which can be disabled at the node.



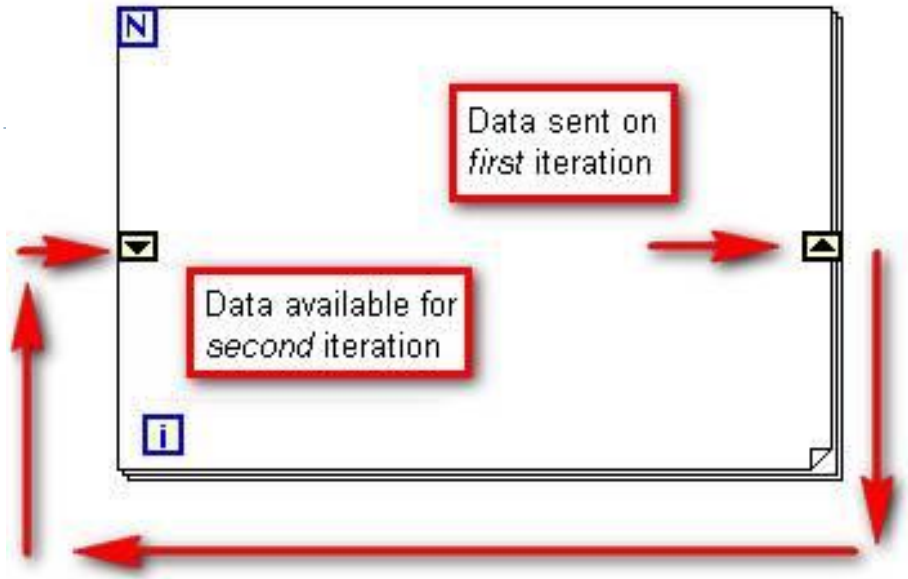
Creating 2D Arrays with For Loops (Auto-Indexing)



Shift Registers



- Selected via right-clicking the frame.
- Enables the result of an iteration to be passed to the next iteration.
- Can be used for any data type



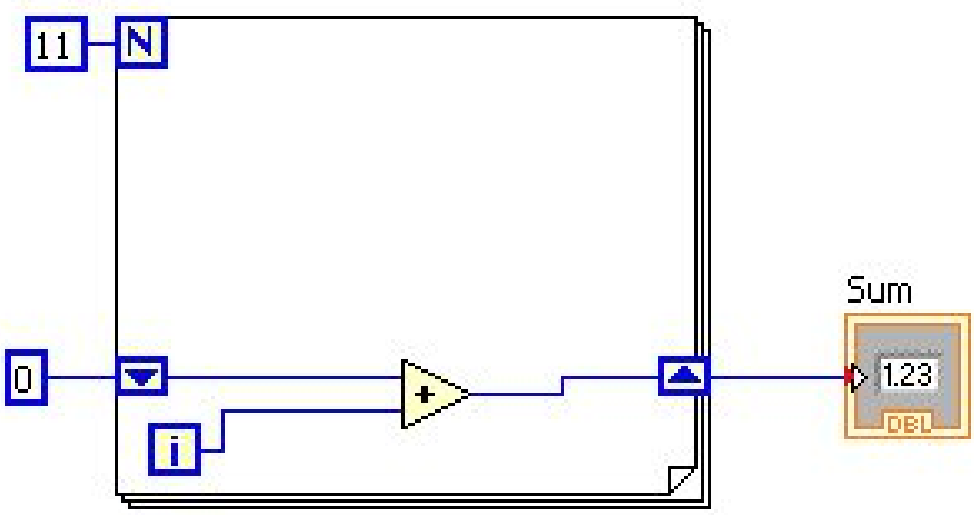
- Data comes into the loop by way of the shift register on the left side of the for loop and is passed to the next iteration of the loop through the shift register on the right side of the for loop.
- The initial value is set by wiring to the left terminal and the final iterations value is output at the right terminal.

Example

- C/C++ Code:

```
int i, n, sum;
n=11;
sum=0;
for (i=0; i<n; i++)
{
sum = sum + i;
}
```

LabVIEW VI:



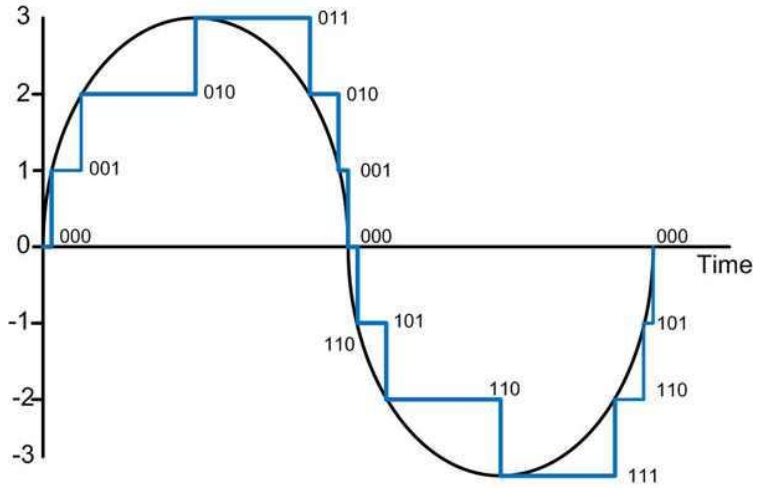
Exercise (1)

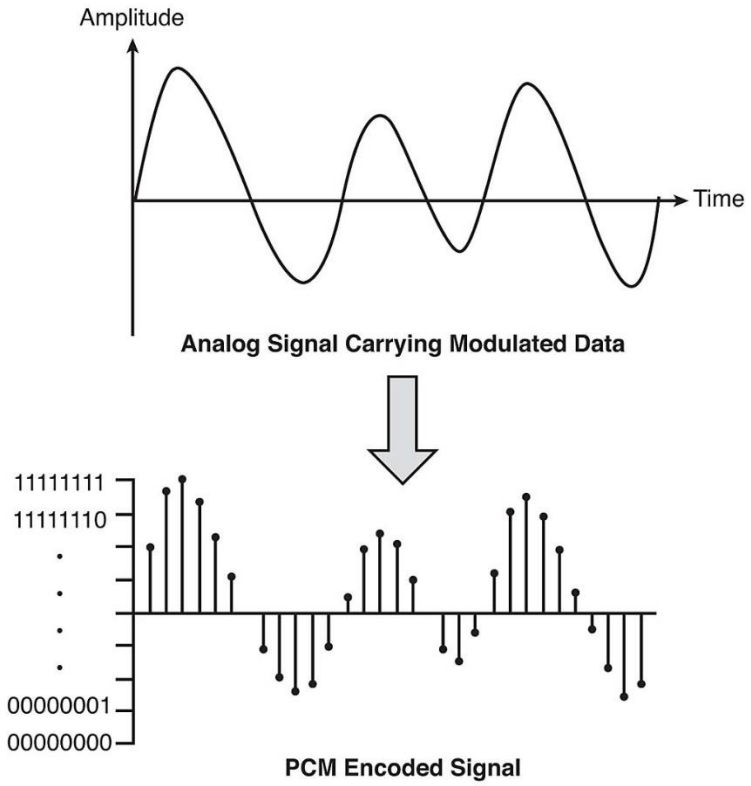
- Design a VI that generates a random number (0-100). Use shift registers with For Loop (20 iterations) to get the maximum of the generated 20 random numbers.

Note:

- Add Indicators for both the random number and the maximum number. Also add a delay (Wait function).
- You will have to use select function (Functions Palette >> Comparison>>Select)

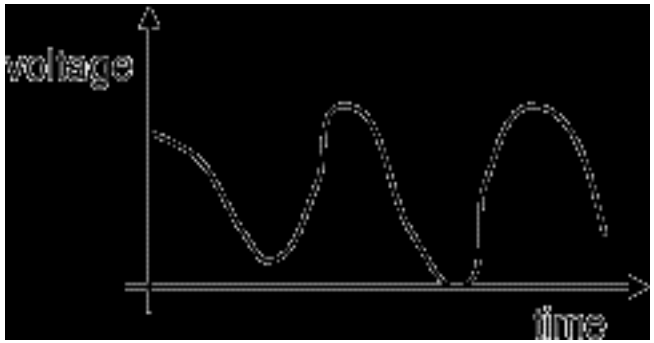
Signals


3
2
1
0
-1
-2
-3
011
010
010
001
001
000
000
000
Time
101
110
101
110
110
111


Amplitude
Time
Analog Signal Carrying Modulated Data
11111111
11111110
00000001
00000000
PCM Encoded Signal

What is signal?

- A *signal* is any kind of physical quantity that conveys information.



Analog Signal:

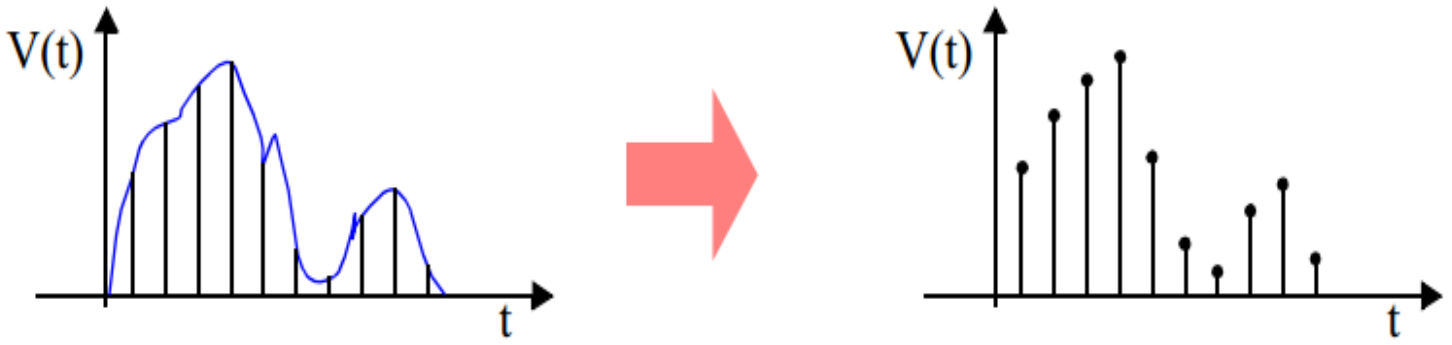
An *analog* signal is a kind of signal that is continuously variable.

- Temperature
- Pressure
- Position
- Speed
- Acceleration
- Sound
- Light Level

plus many others

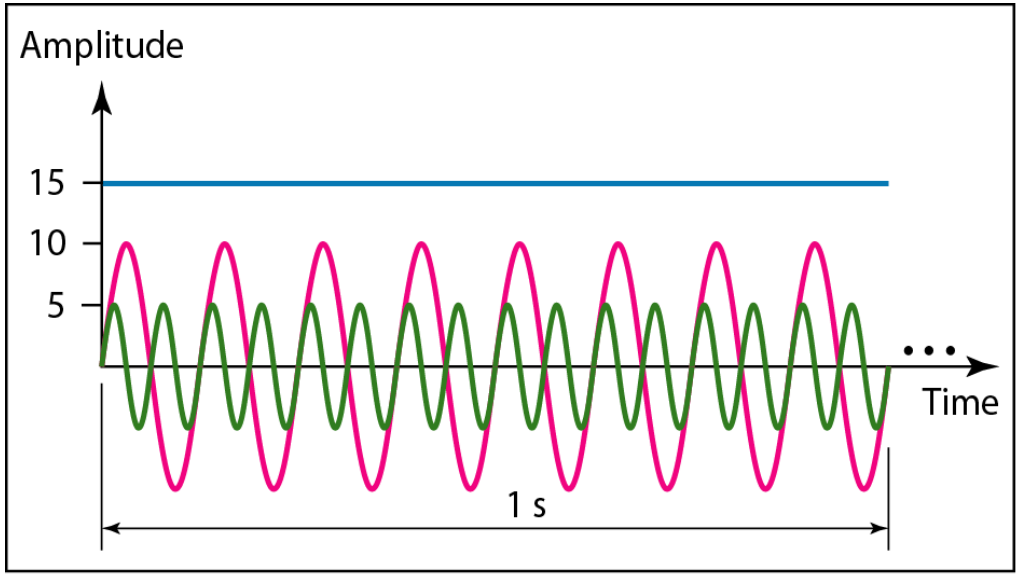
Digital Signal:

- A digital signal can have only a limited number of values.

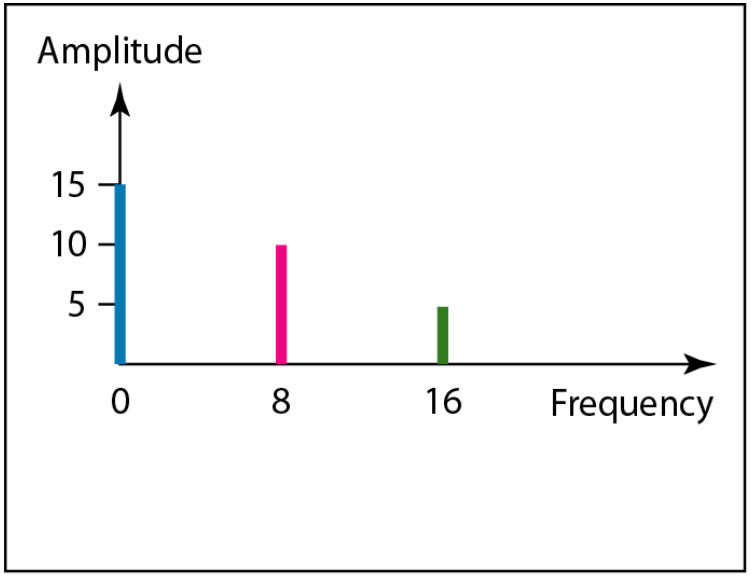


Frequency Domain Analysis:

- The frequency domain is more compact and useful when we are dealing with periodic functions.

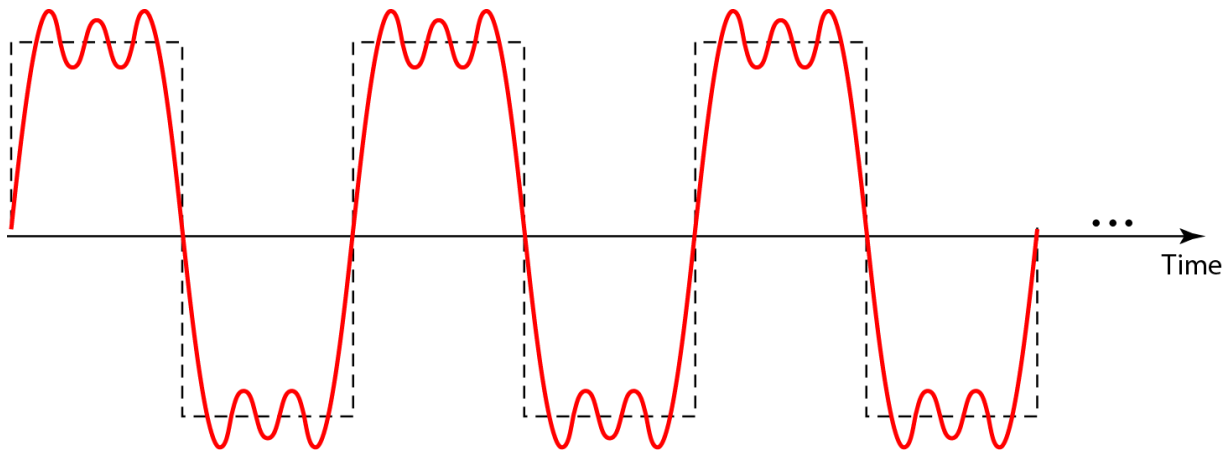


a. Time-domain representation of three sine waves with frequencies 0, 8, and 16

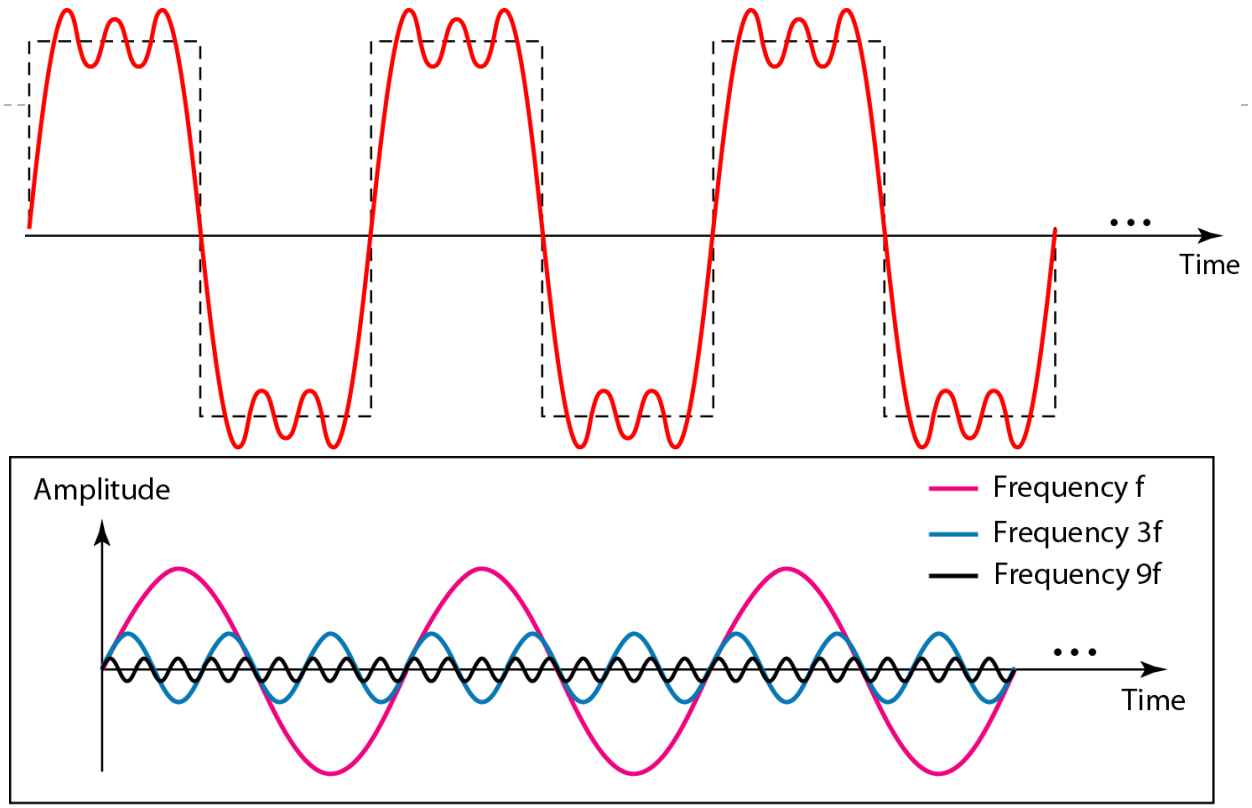


b. Frequency-domain representation of the same three signals

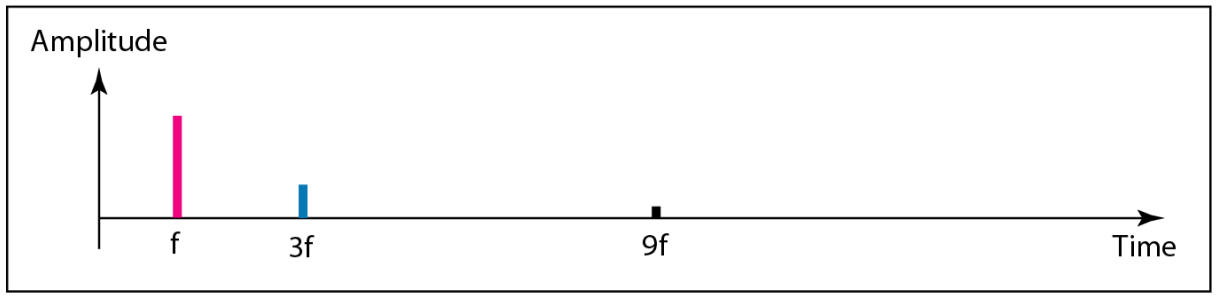
- According to Fourier analysis, any composite signal is a combination of simple sine waves with different frequencies, amplitudes, and phases.



Example:



a. Time-domain decomposition of a composite signal

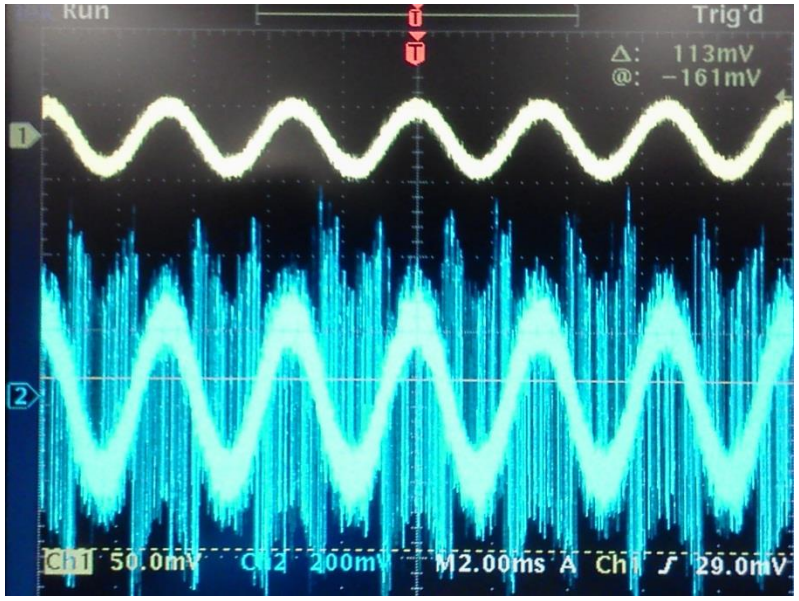


b. Frequency-domain decomposition of the composite signal

Filters

What is a Filter?

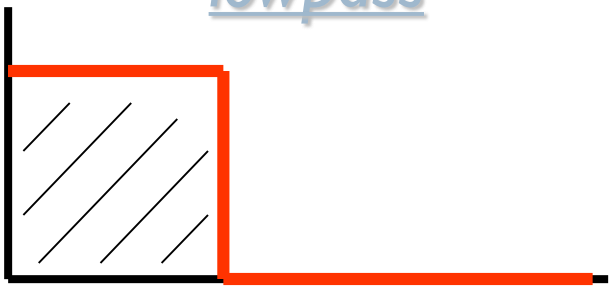
- A *filter* is a system/process that processes a signal in some desired fashion.
- Filtering is a frequency selective operation, in which some frequency components are suppressed and some others are passed.



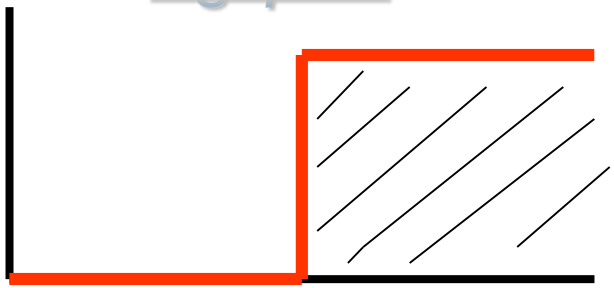
Background:

Four types of filters - "Ideal"

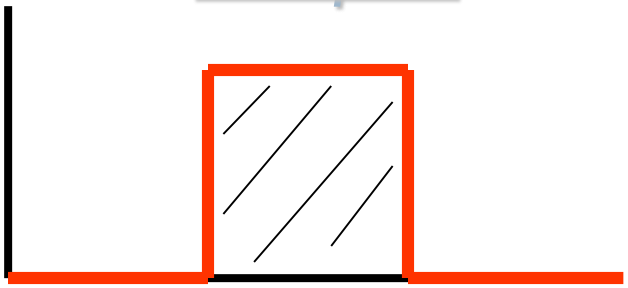
lowpass



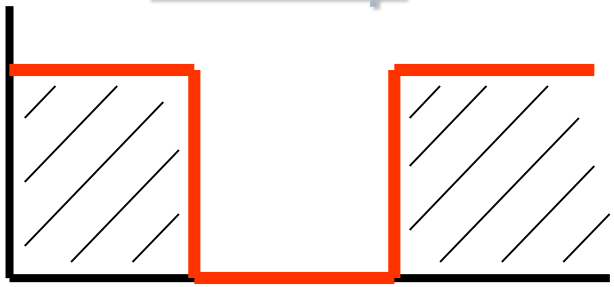
highpass



bandpass



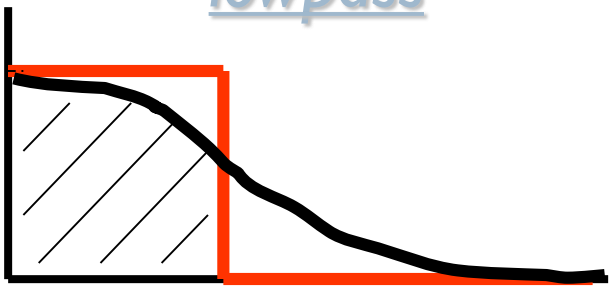
bandstop



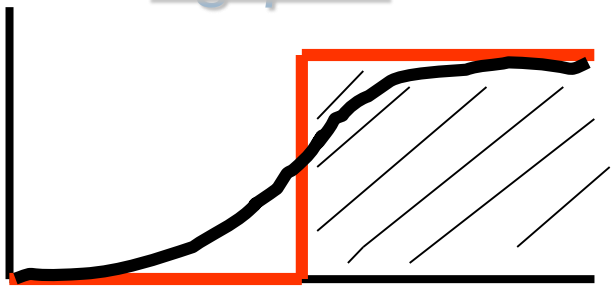
Background:

Realistic Filters:

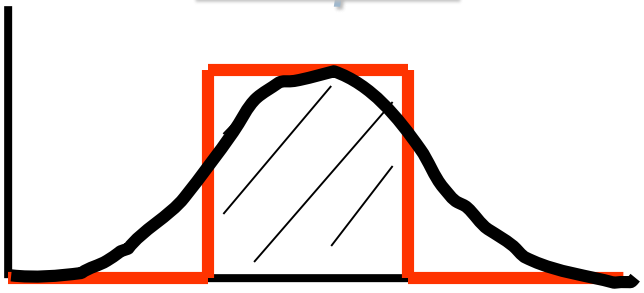
lowpass



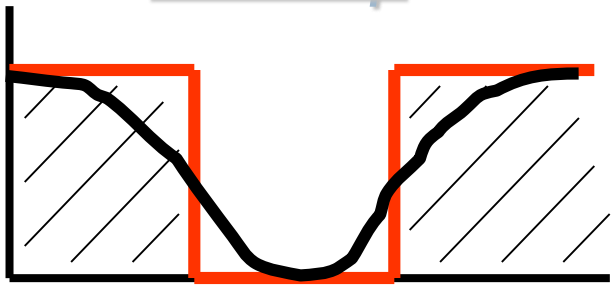
highpass



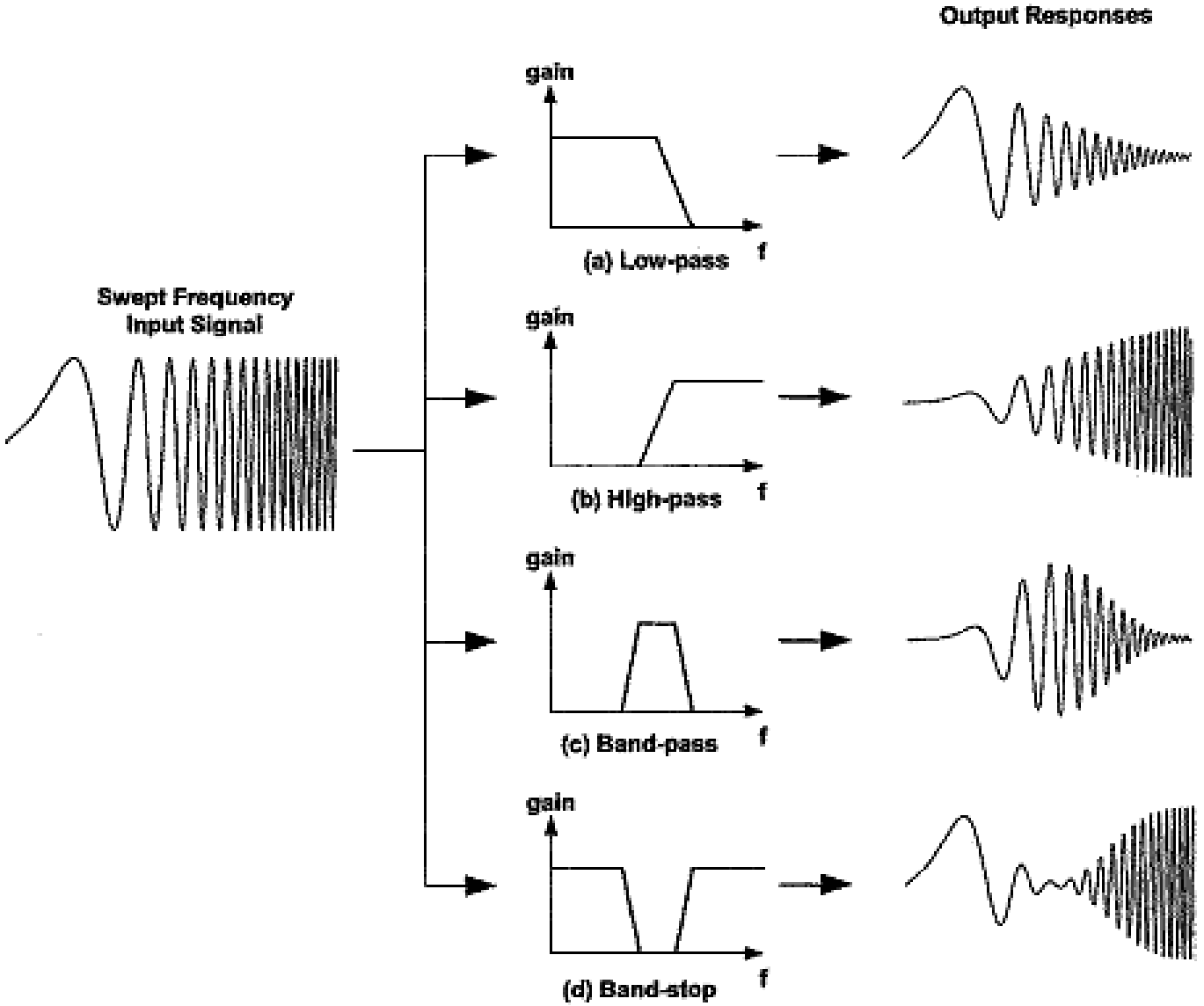
bandpass



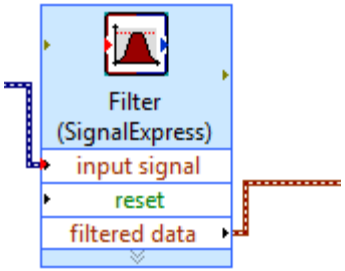
bandstop

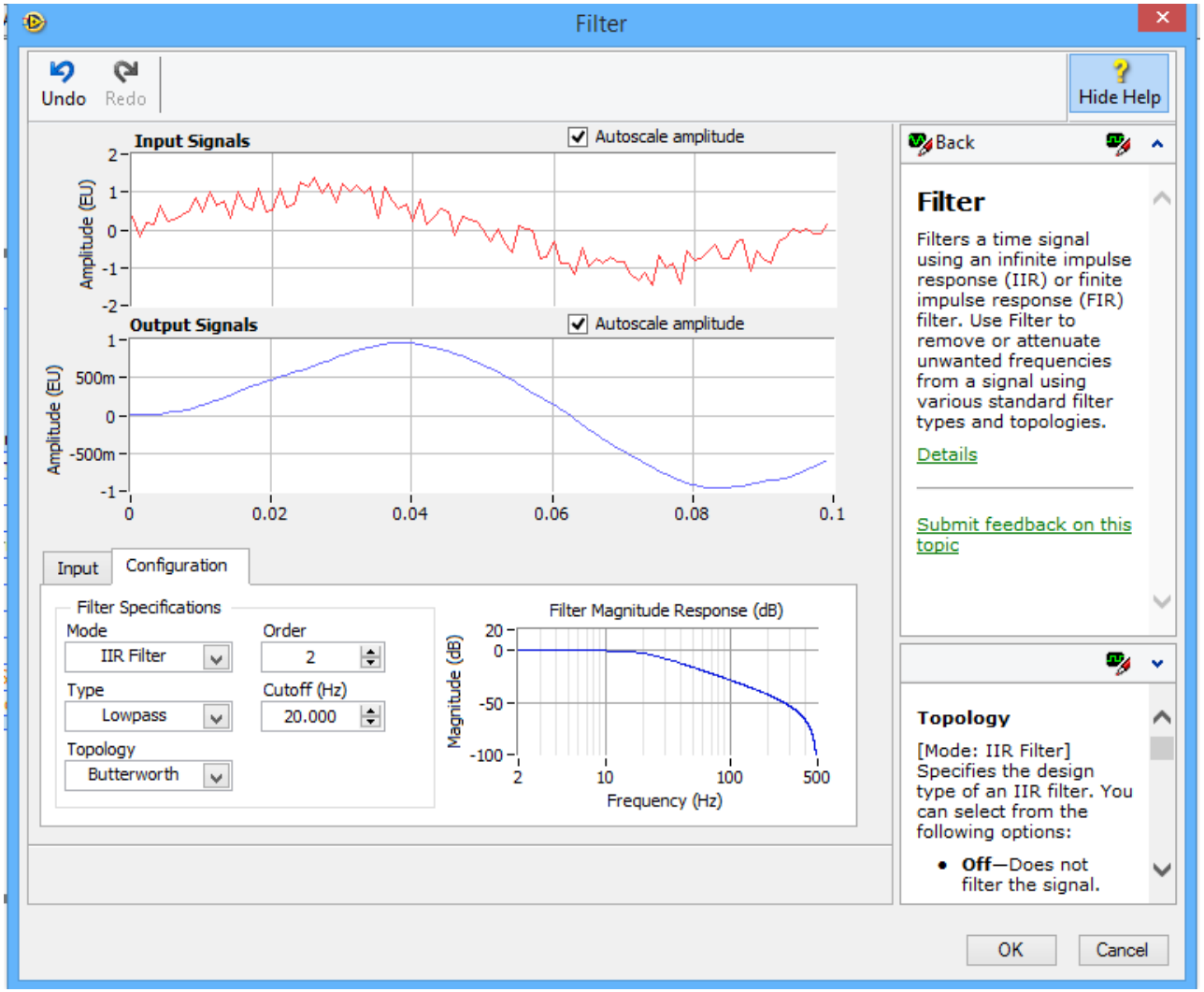


Example


Output Responses
gain
(a) Low-pass
f
Swept Frequency
Input Signal
gain
(b) High-pass
f
gain
(c) Band-pass
f
gain
(d) Band-stop
f

Filters in LabVIEW





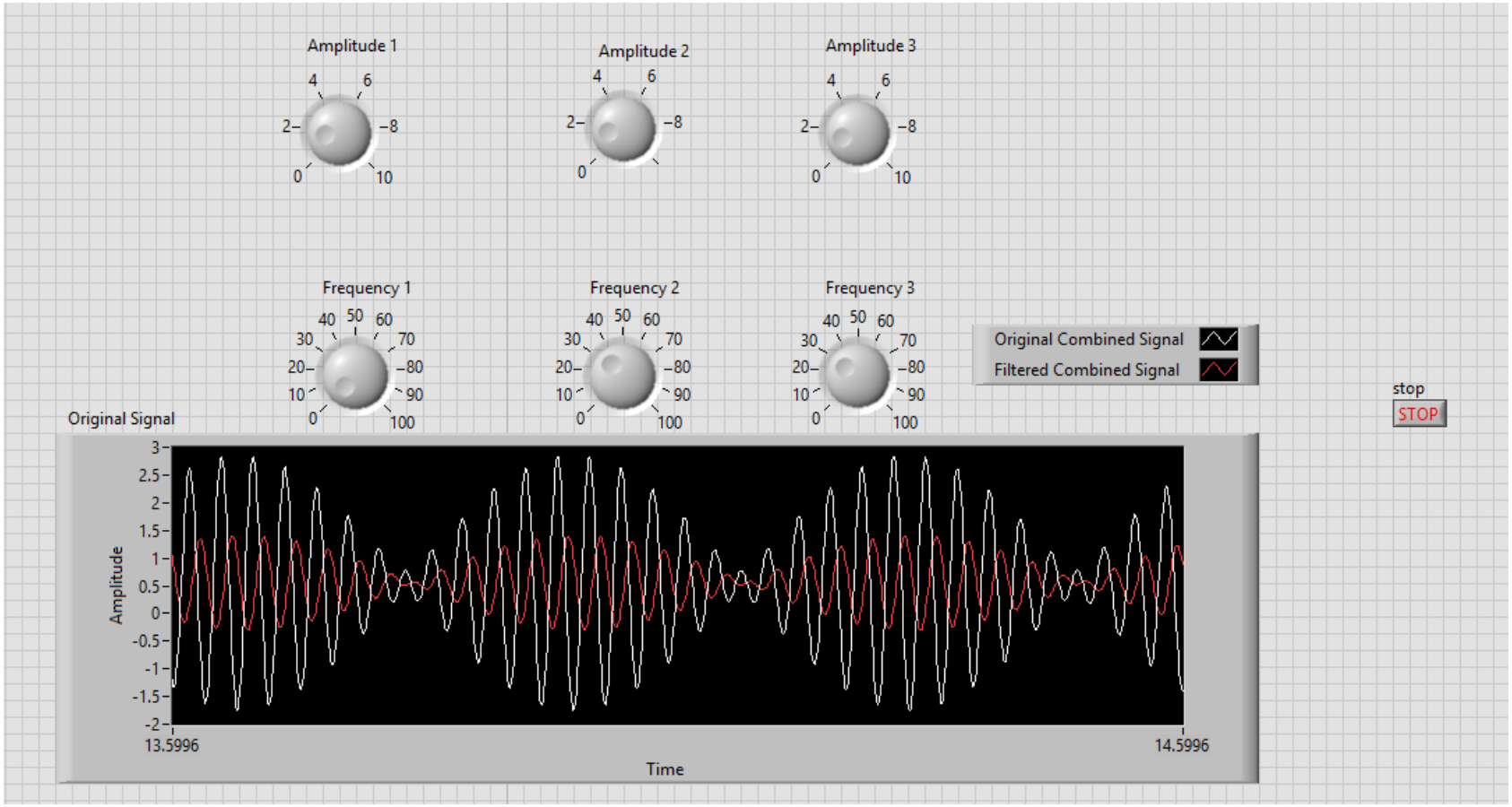
Exercise (3)

Design a VI that generates three simulated sine-wave signals with user-controlled amplitude and frequency Knob. The VI adds the three sine wave signals and perform a filter operation. Show both signals (Original and the filtered one) on a single waveform chart

Notes:

- Vary the signals frequencies and also the filter type (Low Pass - High Pass- Band Pass) to see the effect of these variations on the filtered signal.
- Vary the filter characteristics such as order and cut off frequency, and observe the filtered signal.

Exercise (3)


Amplitude 1
Amplitude 2
Amplitude 3
Frequency 1
Frequency 2
Frequency 3
Original Combined Signal
Filtered Combined Signal
stop
STOP
Original Signal
Amplitude
Time
13.5996
14.5996