FIR and IIR Filter Design & Implementation Structures

Wonyong Sung

2 x 1.5 hrs

School of Electrical Engineering
Seoul National University

Algorithm considerations for system design

- **Performance in terms of signal processing**
  - FIR is better in terms of phase linearity when compared with recursive filtering
    - For image processing, only FIR filtering is adequate.
- **Number of arithmetic ops.(multiplications)**
  - FIR filtering demands a lot of multiplications
    - But not always, for narrowband filtering, it needs a smaller one.
- **Algorithm complexity, parallel & regularity**
  - Seems more important in these days as there are abundant of arithmetic elements in a chip.
  - Parallel structure is good for HW based design.
- **If you start with a poor algorithm, there is not much way to recover the disadvantages!**
  - You need to consider both performance and implementation characteristics.

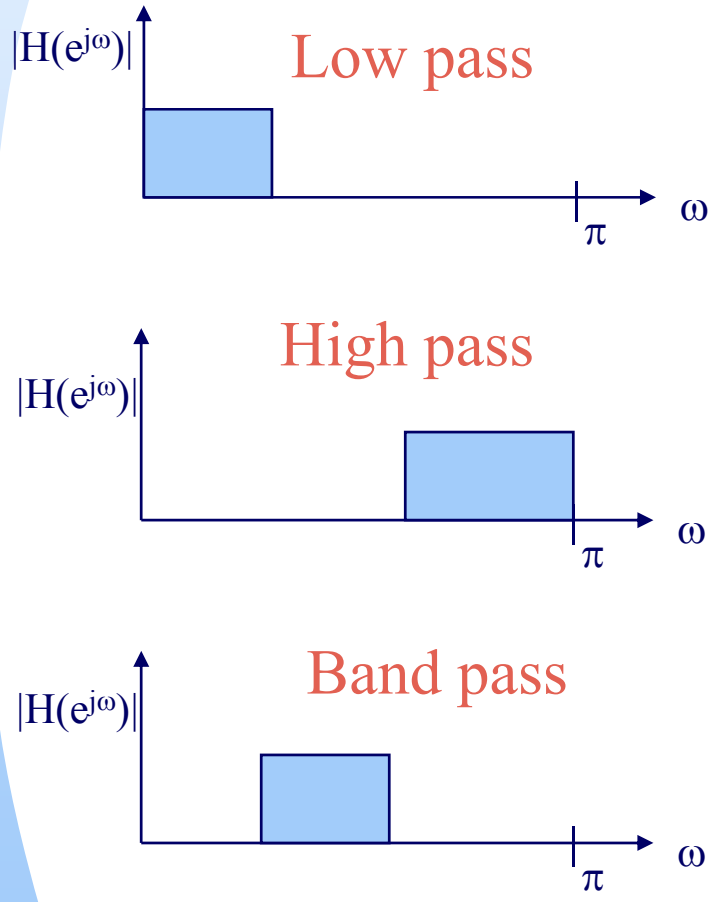
Digital and Analog Filters

- Analog filters are usually implemented in L(inductors) and C (capacitors). For high precision, crystal and ceramic, saw filters are used.
- Low frequency analog filters are bulky because of large inductance and capacitance needed. Analog filters have implementation advantages in high frequency.
- Digital filters are implemented in arithmetic operations, thus is very precise (5% accuracy for C, but 0.01% for digital)
- The number of arithmetic operations is proportional to the sampling frequency.

Digital Filters

- **Types of Digital Filters:**
  - low-pass, band-pass, high-pass, notch-filter, allpass, etc.
- **FIR and IIR Digital Filters**
- **Multiplierless filters**
- **Filters for sampling rate conversion**
- **Structures of Digital Filters**
  - Direct, cascade, parallel forms
  - State-space realizations
  - Orthogonal digital filter
- **Quantization Errors, Stability, accuracy**

Types of Digital Filters



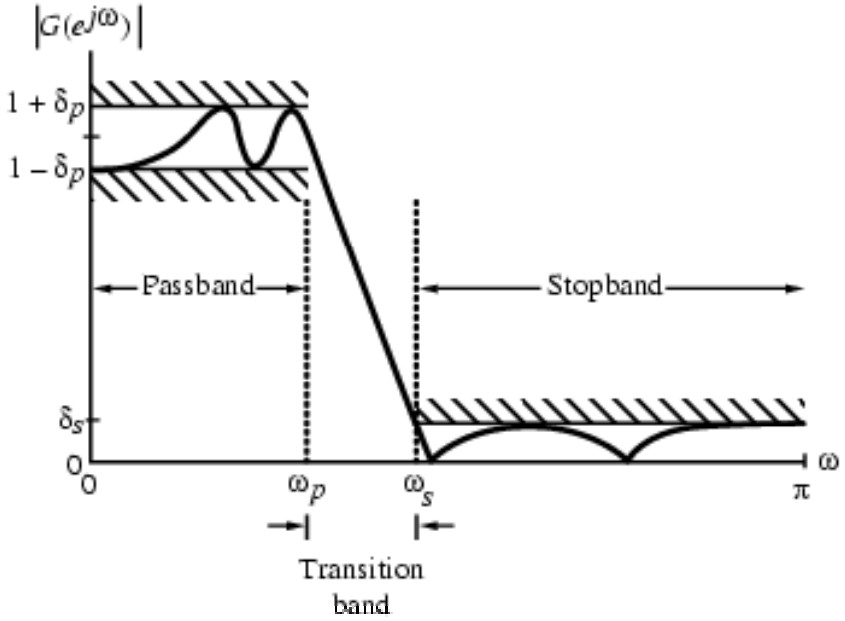
- **Usages:**
  - Low pass: anti-aliasing, smoothing, noise reduction
  - High pass: DC removal, baseline wander reduction
  - Band pass: noise reduction
- **Design:**
  - Choose FIR or IIR filter coefficients to approximate desired frequency response.
  - Usually the designed filter coefficients are not unique! Leaving large design space to be explored. Passband, stopband ripples.

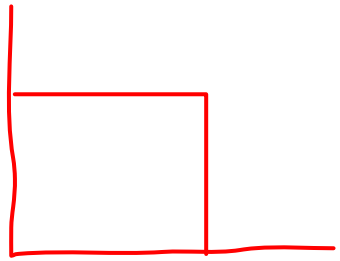
Filter design and implementation

- **Filter design: determining the transfer function (H(z)) from the given frequency domain specification. The location of poles and zeroes are determined.**
- **Filter implementation: determining the filter structure (direct form, 2^{nd} order cascade form, …) , pole-zero pairing if needed, word-length and memory structure for reducing the hardware cost, machine cycles, or power consumption.**

Digital filter specifications

- **For example the magnitude response $|G(e^{j\omega})|$ of a digital lowpass filter may be given as indicated below**





Why not ideal filter?

* Transition bandwidth is important for filter order determination.

Digital filter specifications

- In practice, passband edge frequency F_p and stopband edge frequency F_s are specified in Hz
- For digital filter design, normalized bandedge frequencies need to be computed from specifications in Hz using

$$\omega_p = \frac{\Omega_p}{F_T} = \frac{2\pi F_p}{F_T} = 2\pi F_p T$$

$$\omega_s = \frac{\Omega_s}{F_T} = \frac{2\pi F_s}{F_T} = 2\pi F_s T$$

Digital filter specifications

- In the passband $0 \le \omega \le \omega_p$ we require that with a deviation

$$\left|G(e^{j\omega})\right| \cong 1 \pm \delta_p$$

$$1-\delta_p \le \left|G(e^{j\omega})\right| \le 1+\delta_p, \quad |\omega| \le \omega_p$$

- In the stopband $\omega_s \le \omega \le \pi$ we require that with a deviation δ_s

$$\left|G(e^{j\omega})\right| \cong 0$$

$$\left|G(e^{j\omega})\right| \le \delta_s, \quad \omega_s \le |\omega| \le \pi$$

Digital filter specifications

Filter specification parameters

- ω_p - passband edge frequency
- ω_s - stopband edge frequency
- δ_p - peak ripple value **in the passband**
- δ_s - peak ripple value **in the stopband**

Digital filter specifications

- **Practical specifications are often given in terms of** loss function (in dB)
- $$\mathrm{G}(\omega) = -20\log_{10}\left|G(e^{j\omega})\right|$$
- Peak passband ripple

$$\alpha_p = -20\log_{10}(1-\delta_p) \quad \textbf{dB}$$

- Minimum stopband attenuation

$$\alpha_s = -20\log_{10}(\delta_s) \quad \textbf{dB}$$

FIR, IIR digital filters

- Let {h[n}: impulse response

  {x(n)}: input, {y(n)}: output
- Finite impulse response (FIR) filter:

$$y(n) = \sum_{j=0}^{J-1} h(j)x(n-j)$$

- Has only zeroes (no poles).
- Usually, implemented as a feed-forward type.
- Infinite impulse response (IIR) filter

$$y(n) = \sum_{i=1}^{P} a(i)y(n-i) + \sum_{k=0}^{Q} b(k)x(n-k)$$

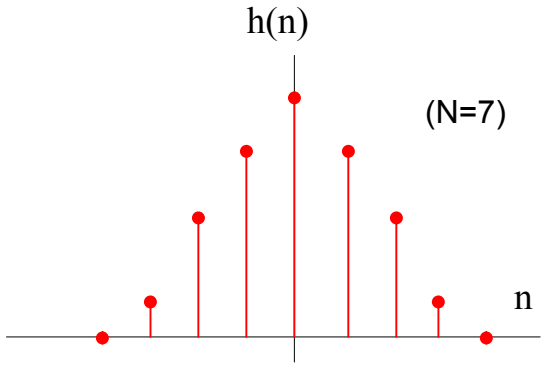
- Both poles and zeroes. The length of impulse (unitpulse)response may be infinite!
- Recursive formula will impact on computation methods (feedback).
- Stability concerns:
  - The magnitude of y(n) may become infinity even all x(n) are finite!
  - coefficient values,
  - quantization error

FIR filters

- **Direct form structure, which has a form of convolution, is usually used. Cascade or parallel forms are a little bit complex in terms of structure. The quantization effects of direct form FIR filters are still tolerable, in most cases.**
- **Symmetric coefficients FIR**
  - Linear phase: critical for image processing
  - Halve the # of multiplications
- **Filter design**
  - Windowing
  - CAD – Parks McClellan method
- **The needed order is usually high.**
- **Good for interpolation, decimation filtering**

Linear phase filter - symmetric FIR

- **h(n) = h(-n)**
- **Evaluate the frequency response (assuming that N is odd) and h(n) is real-valued**



$$H(z) = \sum_{n=n_1}^{n_2} h(n)\ z^{-n} = \sum_{n=-\frac{N-1}{2}}^{\frac{N-1}{2}} h(n)\ z^{-n}$$

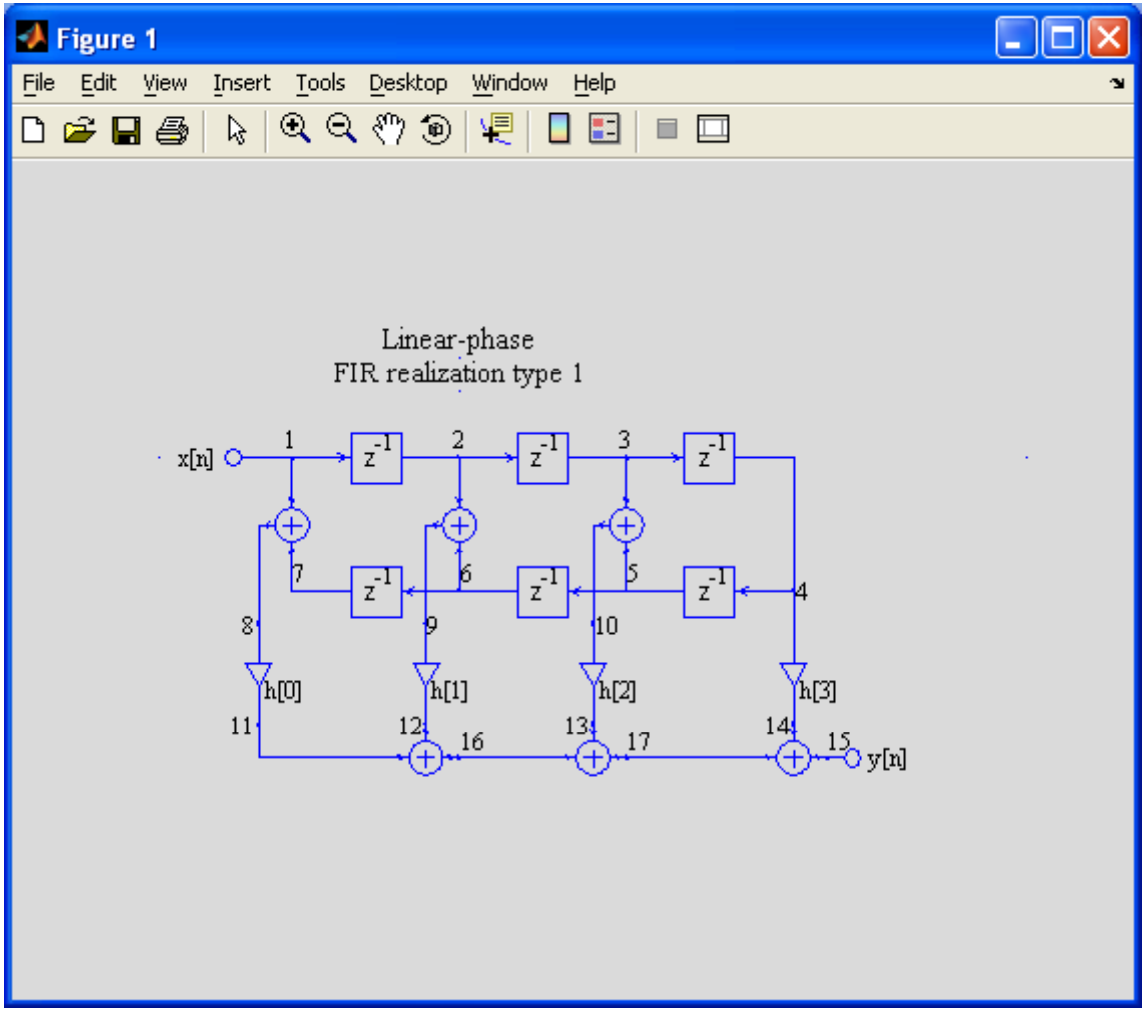
if $h(n) = h(-n)$ we get

$$H(e^{j2\pi\Omega}) = h(0) + \sum_{n=1}^{\frac{N-1}{2}} h(n)\ (e^{-j2\pi n\Omega} - e^{+j2\pi n\Omega})$$

$$H(e^{j2\pi\Omega}) = h(0) + 2\sum_{n=1}^{\frac{N-1}{2}} h(n)\ \cos[2\pi n\Omega]$$

The **frequency response is real**: phase shift is 0 or 180 degrees

Linear-phase type 1



Needed number of coefficients

- **For equiripple LP FIR filters:**

$$N_e = \frac{2}{3}\log\left[\frac{1}{10\, D_{pass}\, D_{stop}}\right]\frac{F_s}{F_{stop} - F_{pass}}$$

  - ✓Independent of BW (F_{pass})!
  - ✓Weak (logarithmic) dependence on the Pass band ripple level and the Stop band attenuation
  - ✓**Linear dependence on the transition band!**

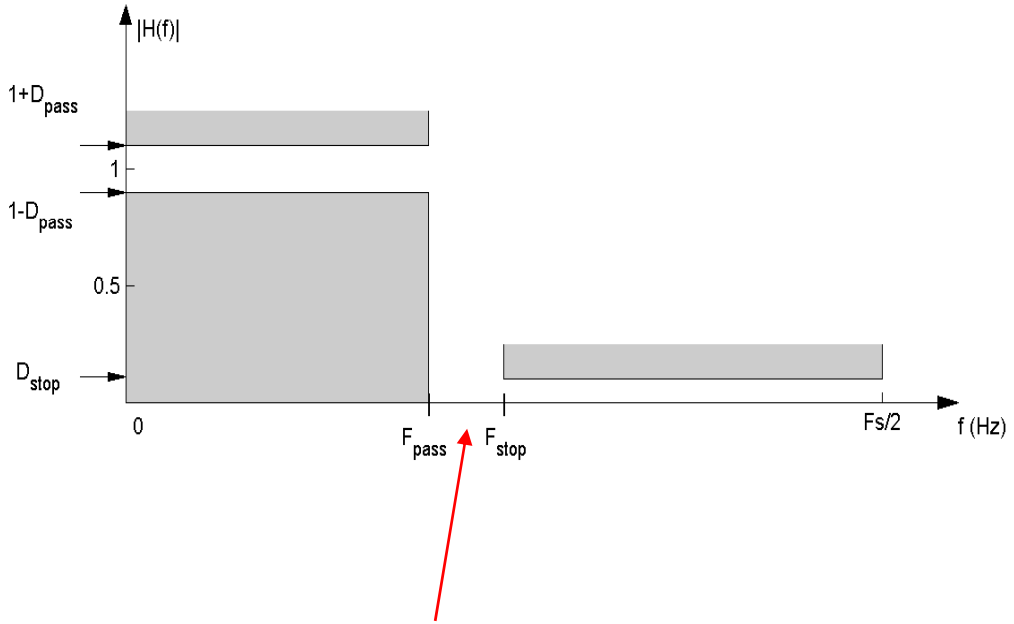
- Our example: Ne = 29 (compared to 31)
- Problem: Very narrow filters -> **Decimating**

Example

- **Develop an AM-SSB modulator using Simulink. There are a few methods for generating SSB signal. Here you try to use a bandpass filter. The filter specification that I suggest is a bandpass filter having the following specification. This filter is designed assuming that the message signal has the frequency range of 0.2KHz ~ 3.8KHz. Carrier freq = 12KHz, Sampling freq = 48KHz**
  - f1: stopband edge: 11.8KHz
  - f2: passband edge: 12.2KHz
  - f3: passband edge: 15.8KHz
  - f4: stopband edge: 16.2KHz
    - Passband ripple: -0.5dB~+0.5dB, stopband attenuation: 40dB

FIR filter design

- **Signal Processing Toolkit of MATLAB**
- **Define the** frequency response template**.**
- **Case of LPF:**
  - ✓ Pass band End F_{pass}
  - ✓ Pass band Ripple D_{pass}
  - ✓ Stop band Start F_{stop}
  - ✓ Stop band Attenuation D_{stop}



F_{stop}-F_{pass} = **Transition band**

Design of Equiripple Linear-Phase FIR Filters

- The linear-phase FIR filter obtained by minimizing the peak absolute value of

$$\varepsilon = \max_{\omega \in R} |E(\omega)|$$

  is usually called the ***equiripple FIR filter***

- After ε is minimized, the weighted error function $E(\omega)$ exhibits an equiripple behavior in the frequency range R

Design of Equiripple Linear-Phase FIR Filters

- **The general form of frequency response of a causal linear-phase FIR filter of length $2M+1$:**

$$H(e^{j\omega}) = e^{-jM\omega} e^{j\beta} \breve{H}(\omega)$$

where the amplitude response $\breve{H}(\omega)$ is a real function of ω

- **Weighted error function is given by**

$$E(\omega) = W(\omega)[\breve{H}(\omega) - D(\omega)]$$

where $D(\omega)$ is the desired amplitude response and $W(\omega)$ is a positive weighting function

Design of Equiripple Linear-Phase FIR Filters

- **For filter design,**

$$D(\omega) = \begin{cases} 1, & \text{in the passband} \\ 0, & \text{in the stopband} \end{cases}$$

- $\breve{H}(\omega)$ **is required to satisfy the above desired response with a ripple of** $\pm\delta_p$ **in the passband and a ripple of** δ_s **in the stopband**

Design of Equiripple Linear-Phase FIR Filters

- **Thus, weighting function can be chosen either as**

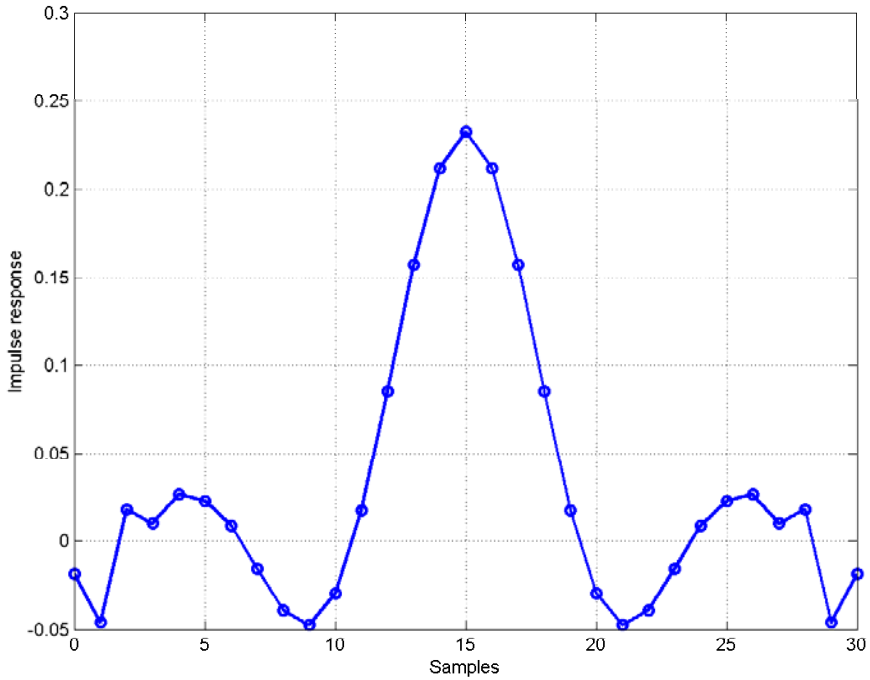
$$W(\omega) = \begin{cases} 1, & \text{in the passband} \\ \delta_p / \delta_s, & \text{in the stopband} \end{cases}$$

or

$$W(\omega) = \begin{cases} \delta_s / \delta_p, & \text{in the passband} \\ 1, & \text{in the stopband} \end{cases}$$

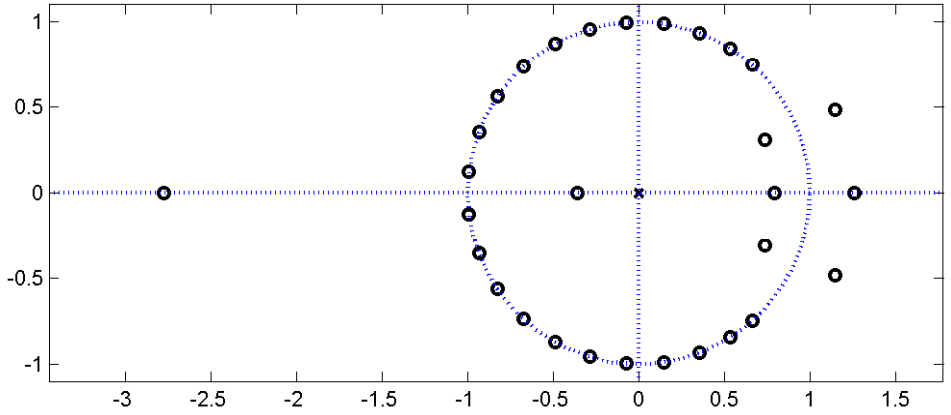
Example 1: Equiripple LP FIR (1)

- **LPF example:**
  - Pass band End F_{pass} = 0.1
  - Pass band Ripple D_{pass}=0.05 (5%)
  - Stop band Start F_{stop}= 0.13
  - Stop band Attenuation D_{stop} = 0.1 (1/10)
- **Minimum of 31 coefficients needed to achieved required specifications**
- **Even-symmetric impulse response**

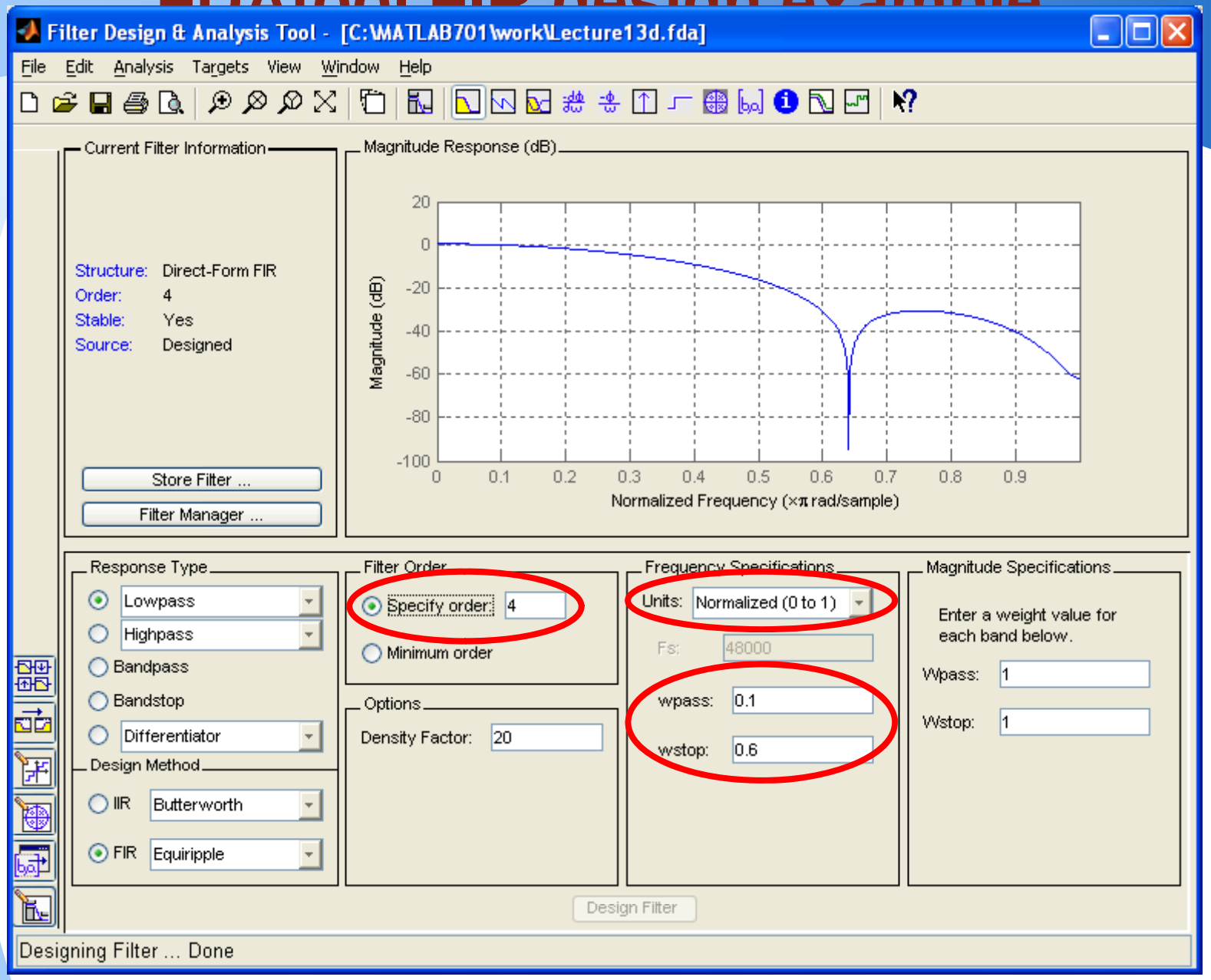


Example 1: Equiripple LP FIR (3)

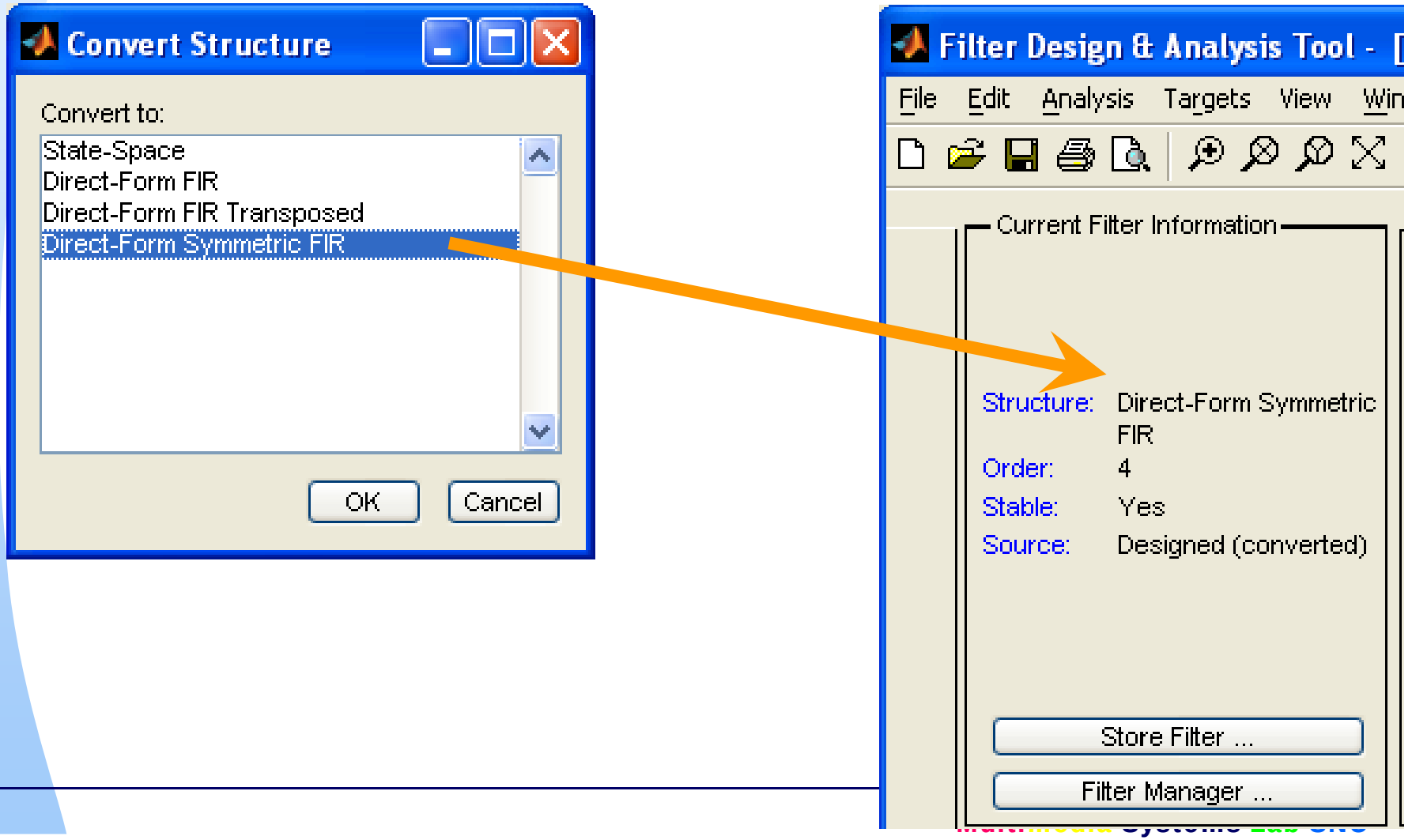
- **Pole-Zero plot in the z-plane**
- **FIR -> pole at zero (causal)**
- **Location of zeros:**
  - ✓ On the unit circle in the Stop band
  - ✓ Far from unit circle in Pass band -> ripples



FDAtool FIR design example



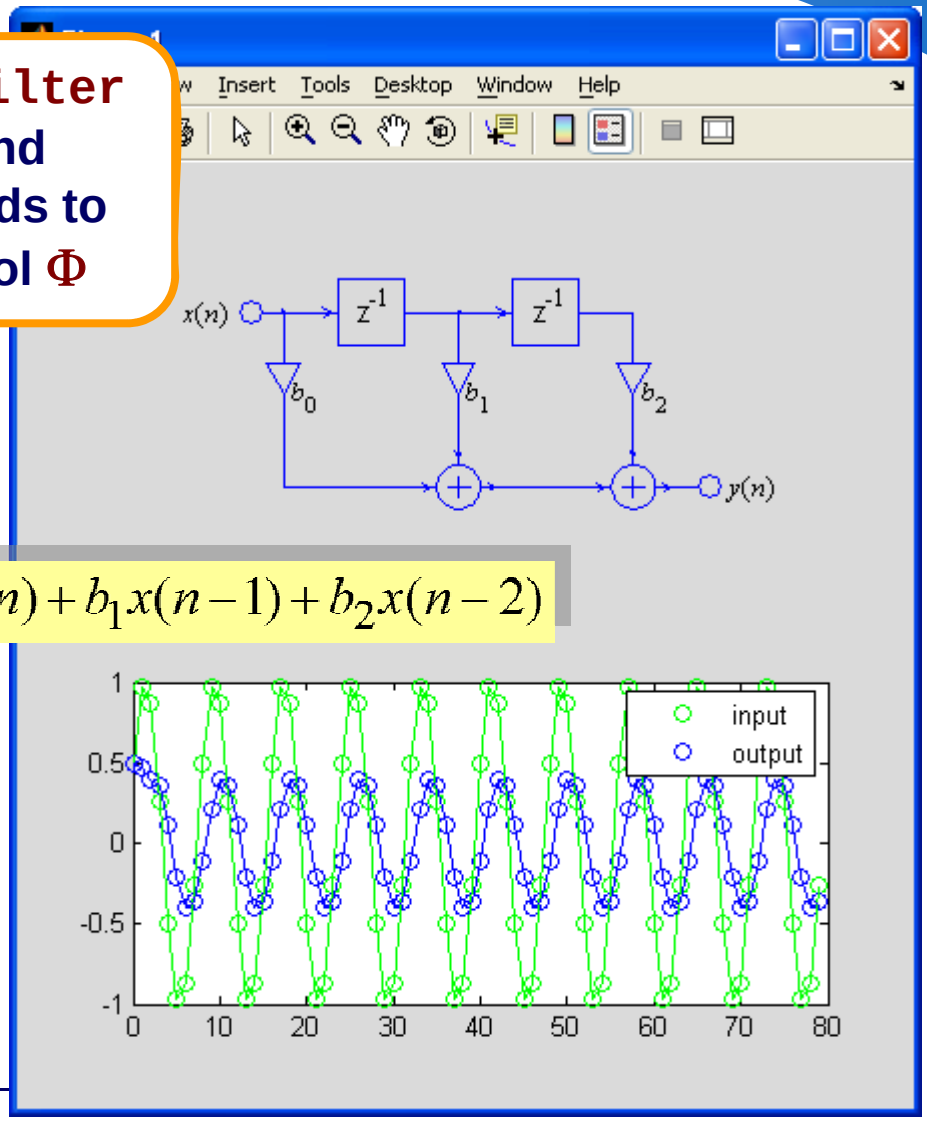
Edit, Convert Structure ...



MATLAB example 1

```
N = 80; k = 0:(N-1);
b0 = 1;
b1 = -1;
b2 = 1;
B = [b0 b1 b2];
f = 1/8;
x = sin(2*pi*f*k+pi/6);
y = filter(B,1,x);
subplot(2,1,1)
systemFIR(0,0,4,5,10,'b')
subplot(2,1,2)
plot(k,x,'go', k,y,'bo',...
     k,x,'g-', k,y,'b-')
legend('input','output')
```

MATLAB `filter` command corresponds to the symbol Φ



$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2)$$

IIR or recursive filter

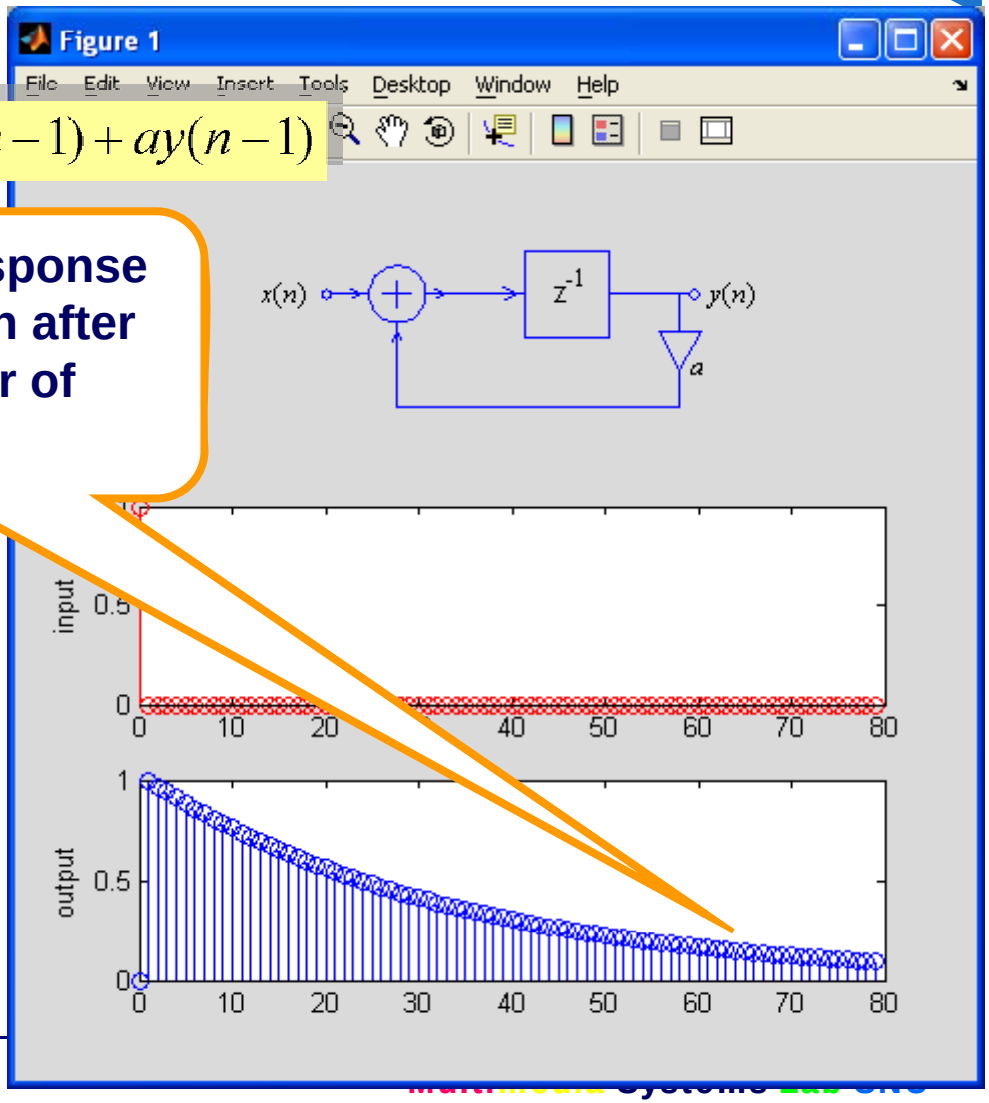
- **Utilize poles and zeros**
  - Poles are for shaping pass bands
  - Zeros are for stop bands
- **The poles should be inside of the unit circle in the z-domain for stability**
  - Finite word-length effects can affect
  - Direct forms are more susceptible.
- **Usually, the order needed is smaller when compared to FIR filters**
- **Direct forms are not good fixed-point implementations**
  - 2^{nd} order cascade, ...

MATLAB example 3

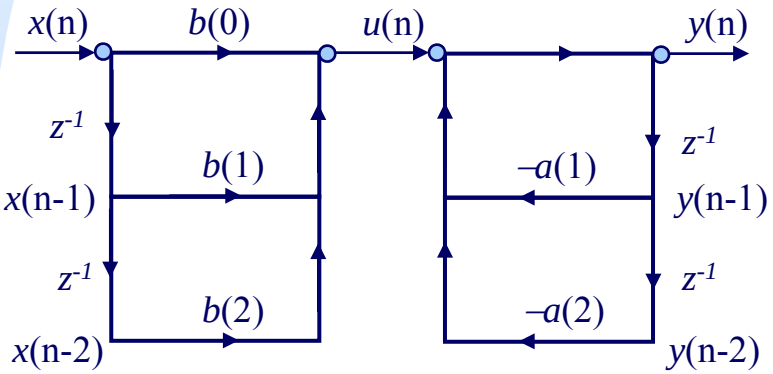
```
N = 80; k = 0:(N-1);
a = 0.97;
B = [0 1];
A = [1 -a];
x = (k==0);
y = filter(B,A,x);
subplot(3,1,1)
draw1stIIR(0,0,4,5,10,'b')
subplot(3,1,2)
stem(k,x,'r')
ylabel('input')
subplot(3,1,3)
stem(k,y,'b')
ylabel('output')
```

$$y(n) = x(n-1) + ay(n-1)$$

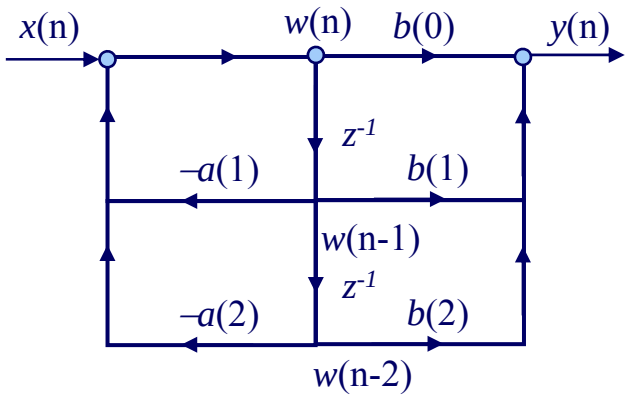
The impulse response does not vanish after finite number of samples



Basic 2nd Order IIR Structures



- **Direct form I realization**
- **5 Multiply 4 Additions per y(n)**
- **4 registers storing x(n-1), x(n-2), y(n-1), y(n-2)**



- **Direct form II realization a.k.a** BiQuad
- **5 multiply, 4 additions per y(n)**
- **2 registers storing w(n-1), w(n-2)**

Different structures determines different execution order, different numerical properties, but retain the same algebraic relation between input and output.

Cascaded and Parallel Structures

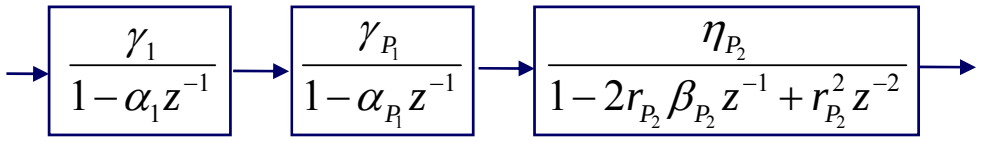
If the characteristic polynomial A(z) has real-valued coefficients, then it can be factorized as:

$$A(z) = \left(\prod_{i=1}^{P_1} (1 - \alpha_i z^{-1}) \right) \cdot \left(\prod_{k=1}^{P_2} (1 - 2r_k \cos \beta_k z^{-1} + r_k^2 z^{-2}) \right)$$

$$= \left(\prod_{i=1}^{P_1} A_i(z) \right) \cdot \left(\prod_{k=1}^{P_2} A_k(z) \right)$$

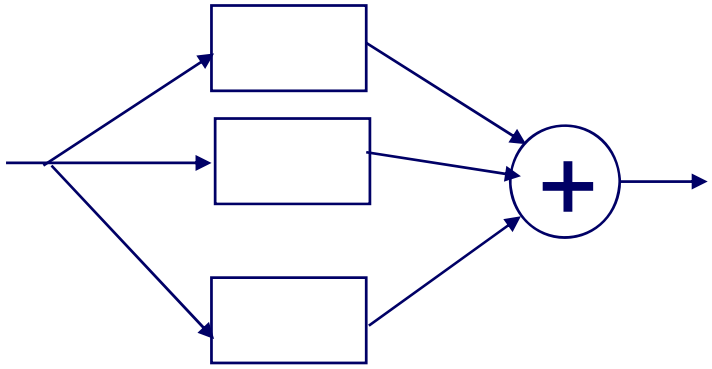
$$P = P_1 + P_2$$

Cascaded realization



Parallel realization

$$H(z) = \sum_{i=1}^{P_1} \frac{\mu_i}{1 - \alpha_i z^{-1}} + \sum_{k=1}^{P_2} \frac{\nu_k + \rho_k z^{-1}}{1 - 2r_k \cos \beta_k z^{-1} + r_k^2 z^{-2}}$$



2nd-order Sections

- Generally implement high-order IIR filters as a sum or cascade of 2nd-order filters to reduce the sensitivity of the overall response to the precision of each coefficient
- High-order polynomial is factored to find poles and zeros, and each 2nd-order filter (biquad) implements two zeros and two poles (real or complex conjugate)
- Poles and zeros are grouped in pairs—usually try to group in such a way that minimizes peak gain of each section
- If cascaded, 2nd-order sections are usually ordered from lowest gain to highest gain to minimize overflow likelihood
- Scaling may be needed for overall response or for each section individually (computation vs. quality)

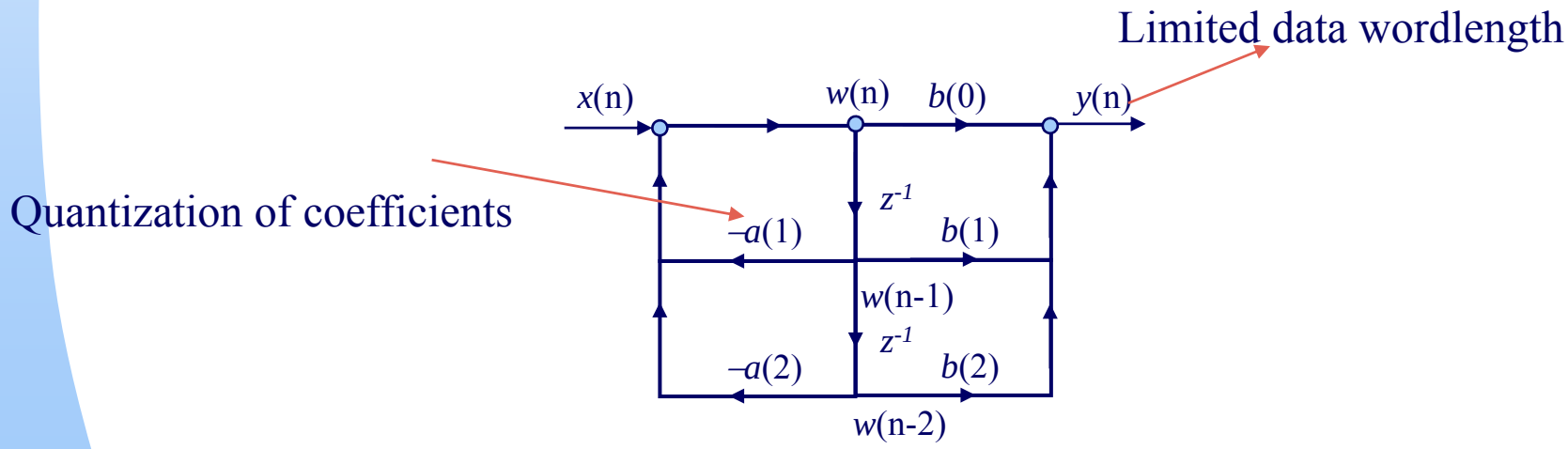
$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Pole-zero pairing

- **When pairing poles and zeros, it is better to combine nearby poles and zeroes for a same 2nd order structure.**
- **Purpose: to reduce the dynamic range of internal signal**
  - It is not good to magnify (attenuate) a signal and then attenuate (magnify) it much in terms of quantization noise sensitivity.
  - Criteria: ***Maximal dynamic range***: a ratio of (a) the maximum magnitude response computed over the whole frequency range to (b) the minimum magnitude response in the passband

Implementation Issues

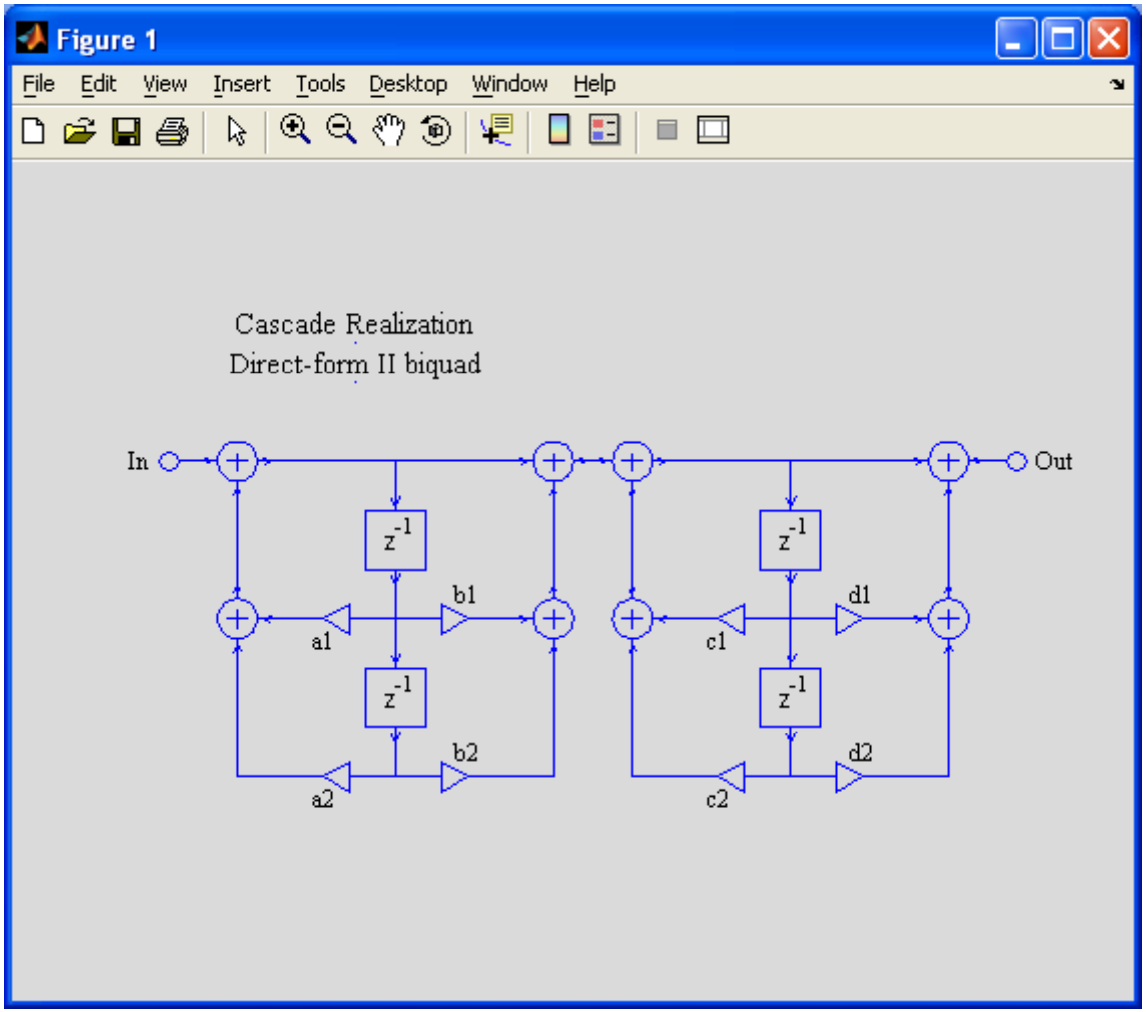
- **Coefficient quantization**
  - Due to limited coefficients word-length.
  - This affects the frequency response.
  - Especially, the pass-band or stop-band ripple can be increasing (make a frequency distortion.)
- **Roundoff error (least significant bits lost)**
  - Due to limited data word-length
  - This introduces random-like quantization noise. Make a noisier filter.



Engineering an IIR filter

- Verify filter performance with quantized coefficients
- Assess roundoff error by modeling the truncation as a *noise source* in the filter structure. Determine the transfer function from each roundoff noise source to the output.
- Calculate overflow situations by looking for peaks in frequency response from input to each accumulation point in the filter.

Cascade direct form II



Comparison of the complexity of different IIR filters

Structure	Number of multiplications	Number of additions and subtractions	Total number of operations	Required bitwidth
Direct form	16	16	40	21
Cascade	13	16	34	12
Parallel	18	16	39	11
Continued fraction	18	16	35	23
Ladder	17	32	50	14
Wave digital	11	30	47	12

From: Miodrag Bolic

Further reading

M. D. Lutovac, D. V. Tošić, B. L. Evans

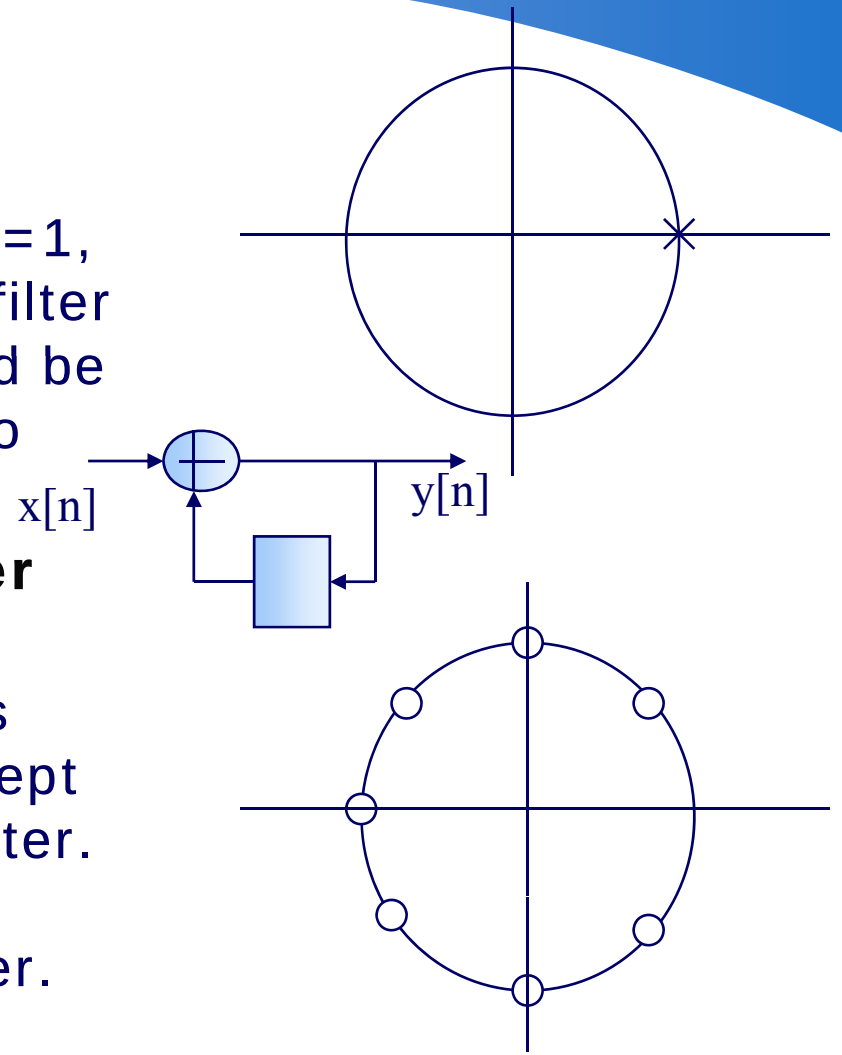
Filter Design for Signal Processing Using MATLAB and Mathematica

Prentice Hall
Upper Saddle River, New Jersey
ISBN 0-201-36130-2, (c) 2001

http://kondor.etf.bg.ac.yu/~lutovac/

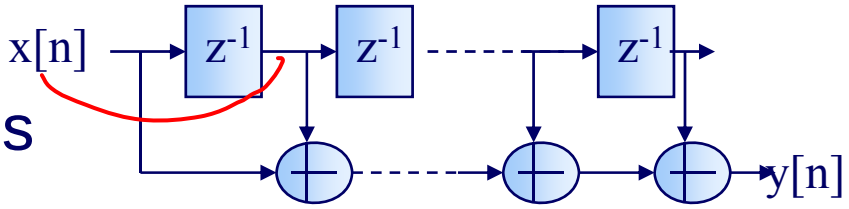
Multiplierless filters

- **Integrator**
  $H(z) = 1/1-z^{-1}$
  - Integrator has a pole at z=1, so it is a kind of lowpass filter with no multiplier. Should be careful for overflow due to DC accumulation.
- **Moving average (MA) filter**
  $H(z) = (1-z^{-N})/(1-z^{-1})$
  - MA filter has many zeroes around the unit circle except for z=1. It is a lowpass filter. It can be modified to a bandpass or highpass filter.
  - $= 1+z^{-1}+z^{-2}+\ldots+z^{-(N-1)}$
- **Good for FPGA's with DSP blocks**

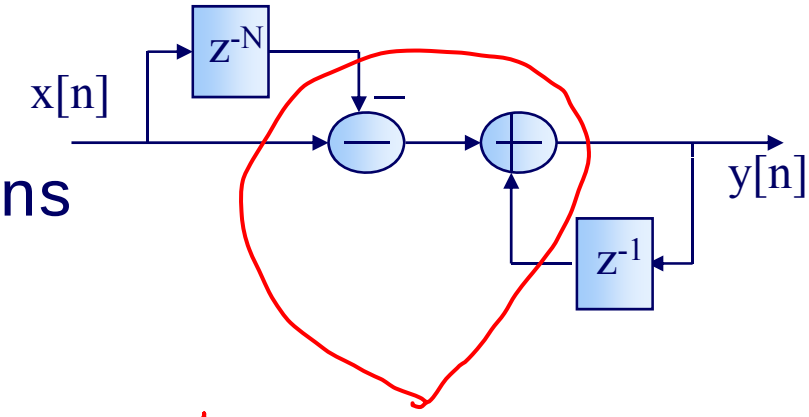


Two moving averager implementations

- **Feed-forward type:**
  - Higher number of arithmetic operations
  - Better stability



- **Feed-back type:**
  - Lower number of arithmetic operations
  - Should be careful against overflows



- **Both structures have the same frequency response with infinite precision arithmetic**

! there should not be quantization here

Key points

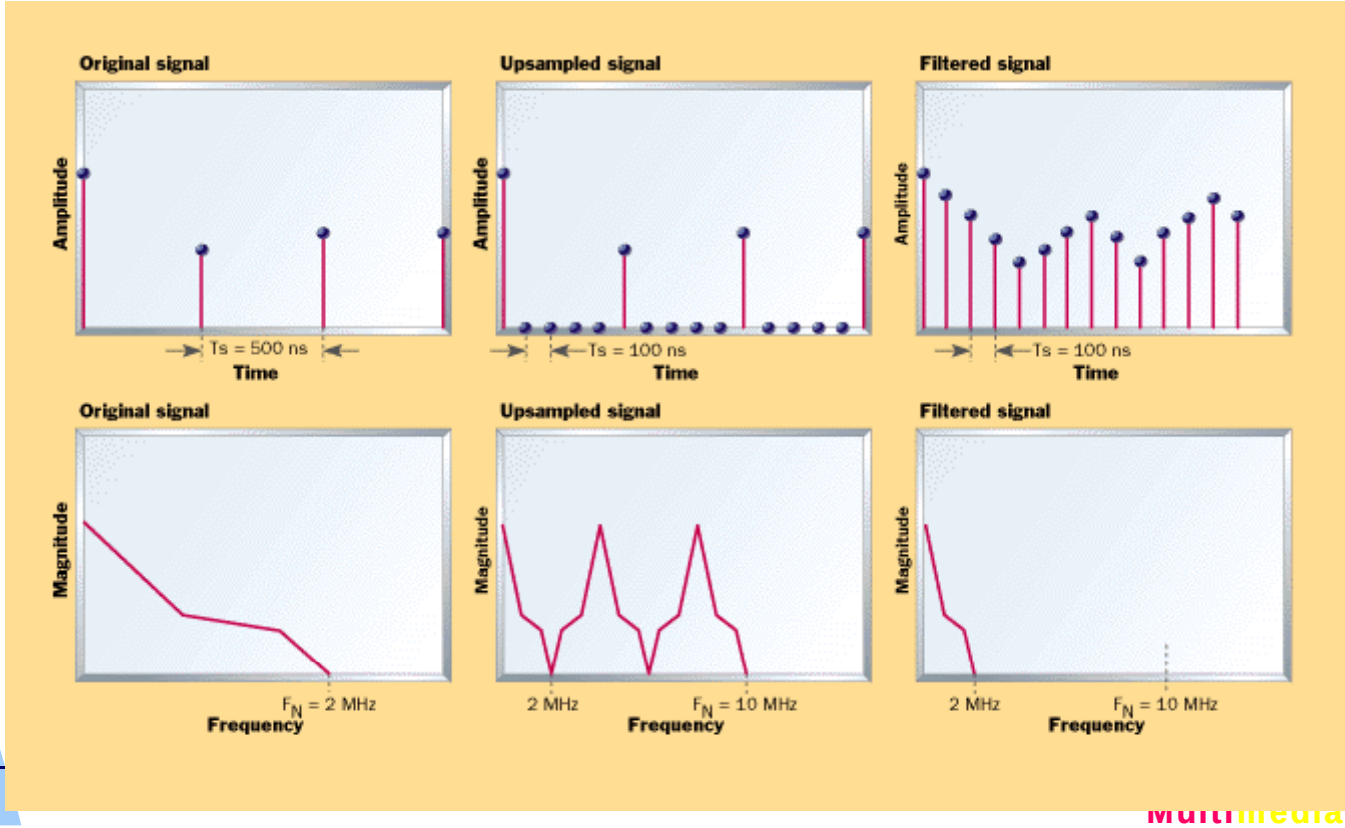
- **Different types and structures of digital filters may achieve the same or similar goals, but with different implementation costs, numerical properties**
- **Design space that can be explored**
  - Filter types: FIR, IIR, adaptive filters
  - Specifications/performance goals
  - Filter structures
    - Direct, parallel, cascade …
  - Numerical properties (floating point/fixed point, etc.)

Interpolation, Decimation, Narrow band, Multi-band filters

- **These filters are widely used for communication, especially modulation/demodulation**
- **The filter lengths can be very long because of narrowband filtering**
- **Assume decimation by 1:10**
  - The passband needs to be less than 1/10 pi. The practical transition bandwidth would be around 1/50 ~ 1/100 pi. Long FIR filter
  - What if the decimation ratio is 1:100
    - The transition bandwidth would be around 1/500~1/1000 pi -> Very long FIR filter
- **Is it a good idea to use IIR filters?**

Interpolation (Sample rate increase)

- Insert N-1 zeroes between the samples, then conduct lowpass filtering (pi/N).
- Among N tap inputs, only one is non-zero (no need to compute N-1 mult.)



Interpolation filter

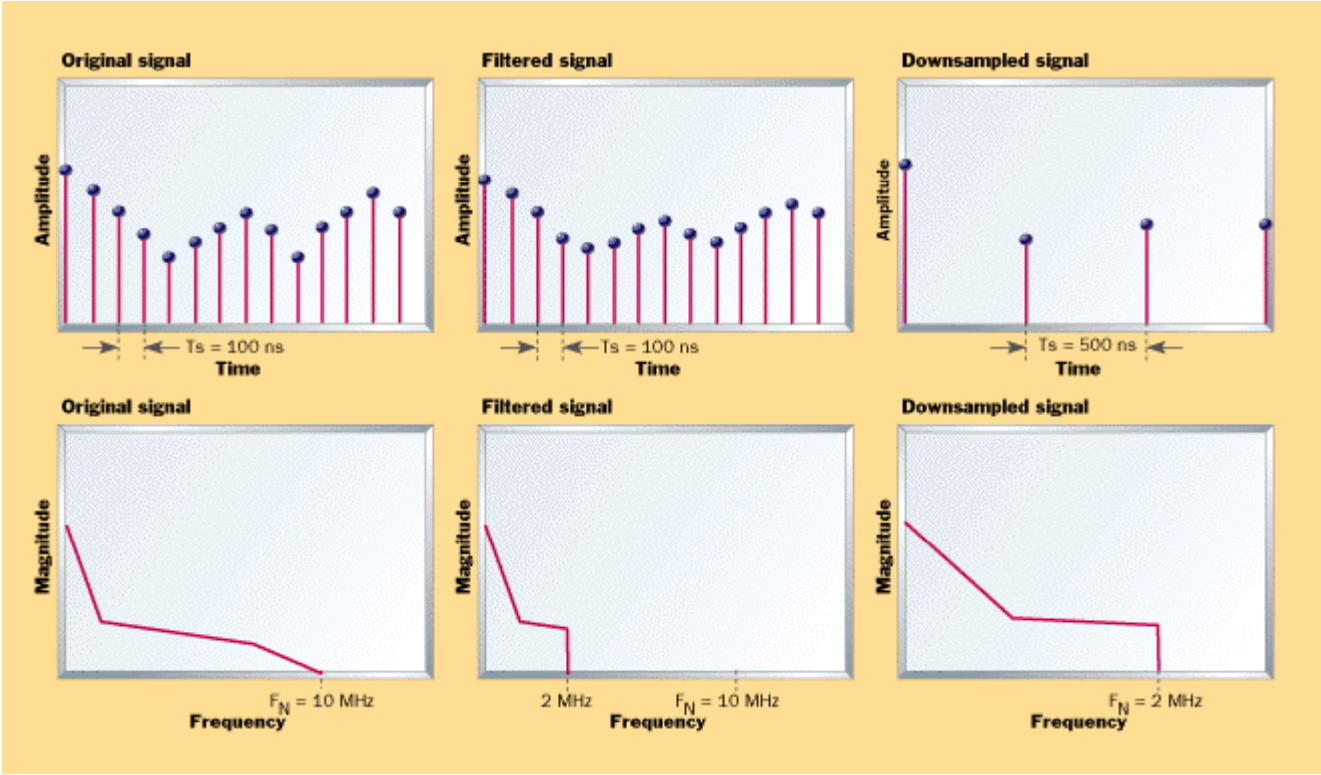
- **The filter length is proportional to the transition BW -> increasing with a large interpolation ratio.**
- **However, we do not need to compute all the output samples. Only 1 sample needed among every N samples.**
- **What is the filter structure?**
  - Do we need a long tap, mostly filled with zeroes?
  - It becomes a polyphase filter.
    - The filter length is reduced to 1/interpolation ratio
    - The coefficients are changing.

Here's an example of a 12-tap FIR filter that implements interpolation by a factor of four. The coefficients are h0-h11, and three data samples, x0-x2 (with the newest, x2, on the left) have made their way into the filter's delay line:

h0	h1	h2	h3	h4	h5	h6	h7	h8	h9	h10	h11	Result
x2	0	0	0	x1	0	0	0	x0	0	0	0	x2·h0+x1·h4+x0·h8
0	x2	0	0	0	x1	0	0	0	x0	0	0	x2·h1+x1·h5+x0·h9
0	0	x2	0	0	0	x1	0	0	0	x0	0	x2·h2+x1·h6+x0·h10
0	0	0	x2	0	0	0	x1	0	0	0	x0	x2·h3+x1·h7+x0·h11

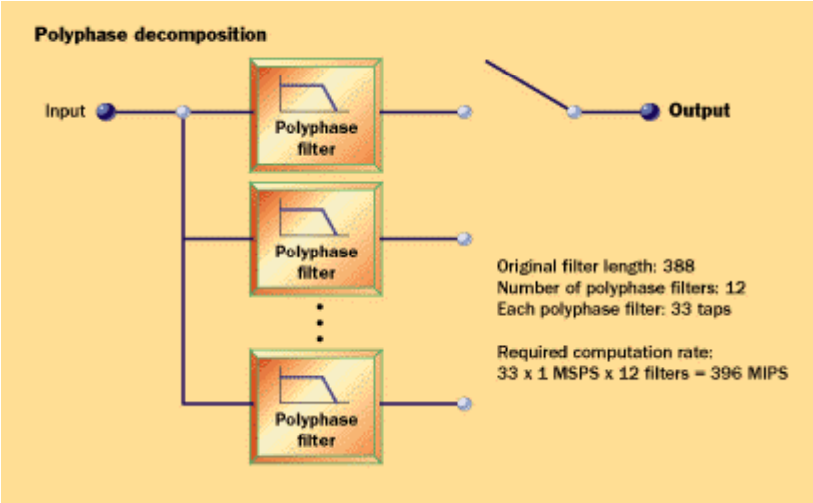
Decimation (Sample rate reduction)

- **Conduct lowpass filtering (pi/N).**
- **Among N output samples, select only one. So, we do not need to compute N-1 output samples.**
  - (This is not possible for recursive filters because of feedback!)



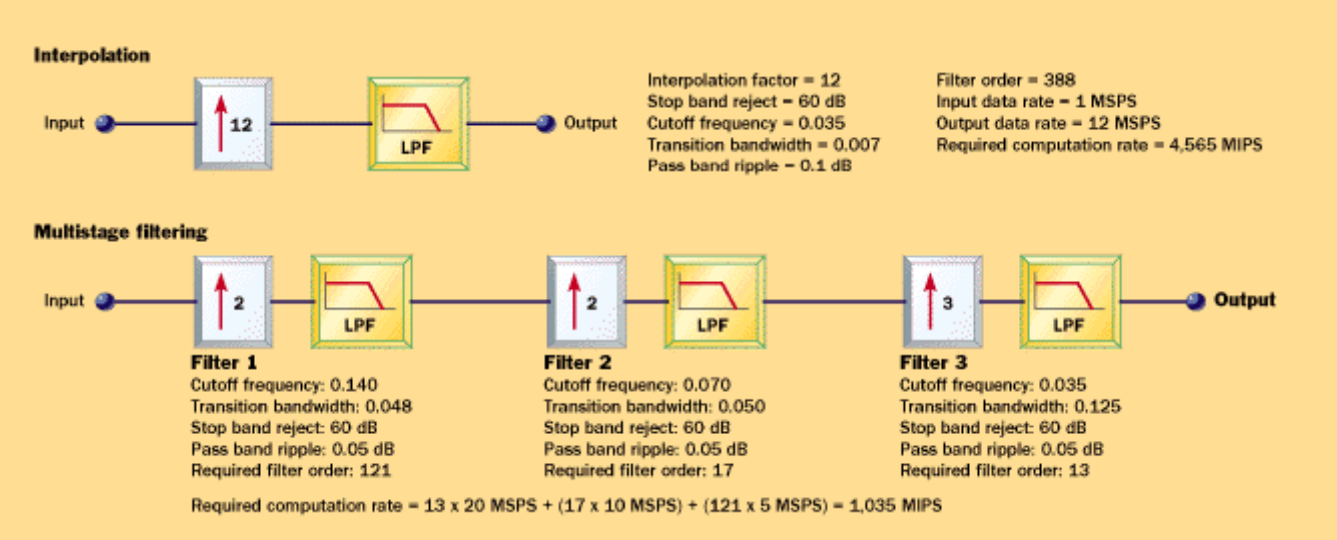
FIR filter implementation for decimation, interpolation

- For interpolation: (N-1) among N input values at the tap are zero! No need to conduct multiplications
- For decimation: We only need to compute 1 output sample at every N output.
- It is interpreted as polyphase filter with the tap length of about M/N.



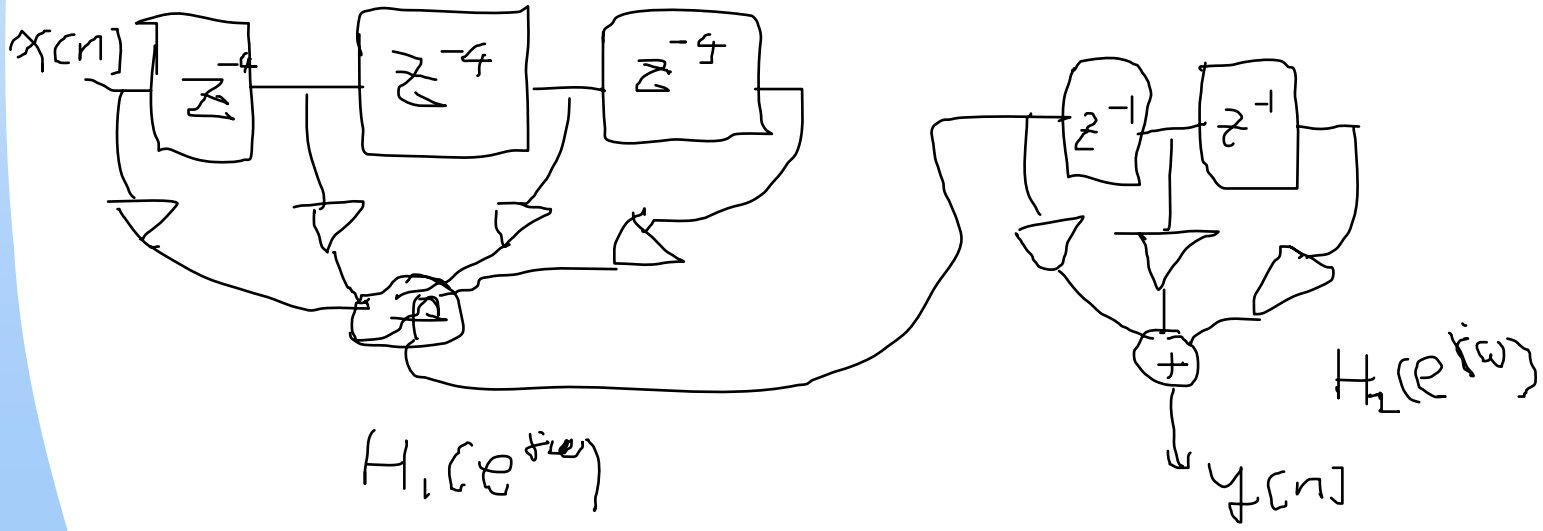
FIR filtering for narrowband signal

- **Narrow band filtering (large order needed): It is advantageous to conduct decimation followed by post-filtering (operates at low sampling rate) -> the total number of mult's is reduced.**
- **Application to IF filtering for digital radio**



Narrow bandpass filtering

- **IFIR (Interpolated FIR)**
  - Consists of two stages
  - First stage filter contains z-M delay blocks
    - This makes a narrow band filter, but there exist unwanted harmonic bands
  - 2nd stage eliminates harmonic bands



- **ifir - Interpolated FIR filter from filter specification**

Example with SSB generation

- What if the carrier frequency for AM in HW#1 is 2,048KHz. Assume that the sampling frequency for the carrier is 4*2,048KHz.
- Design the bandpass filter for removing the lower sideband.
- Assume that the message signal has the sampling frequency of 8KHz. Design the interpolation filter for interpolating the message signal.
- Consider a heterodyne modulator, where the 1st stage modulation frequency is 12KHz. The generated SSB is, then, modulated to 2,048KHz.

Discussion

- FIR filter can be implemented using direct form or fast convolution methods like FFT.
- IIR filters are often factored into products (cascade realization) or sum (parallel realization) of 1^{st} order or 2^{nd} order sections due to numerical concerns.
- There are numerous possible realization structures of the same digital filters.
- Digital filter coefficients are not always exact. It is possible to realize a digital filter with the same desired properties but different filter structures and coefficients to exploit favorable implementation alternatives. A state space model is a good example.

Quantization error, stability, overflow

- **An IIR filter is BIBO stable if all the poles of its transfer function H(z) lie within the unit circle in z-plane.**
- **A stable IIR digital filter may become unstable when its coefficients are subject to severe quantization due to finite length of registers.**
- **Hence an IIR filter that is stable when designed with a 32 bit machine may become unstable when implemented with an 8-bit micro-controller!**

- **Overflow**
  - Dynamic range of intermediate results must be bounded.
  - Otherwise, overflow check must be used and that is very costly.
  - Saturation arithmetic may reduce the error caused by overflow.