# Basics of multirate techniques. Efficient decimation and interpolation.

Signal Processing Systems Fall 2025

Lecture 8 (Thursday 20.11.)

# Outline

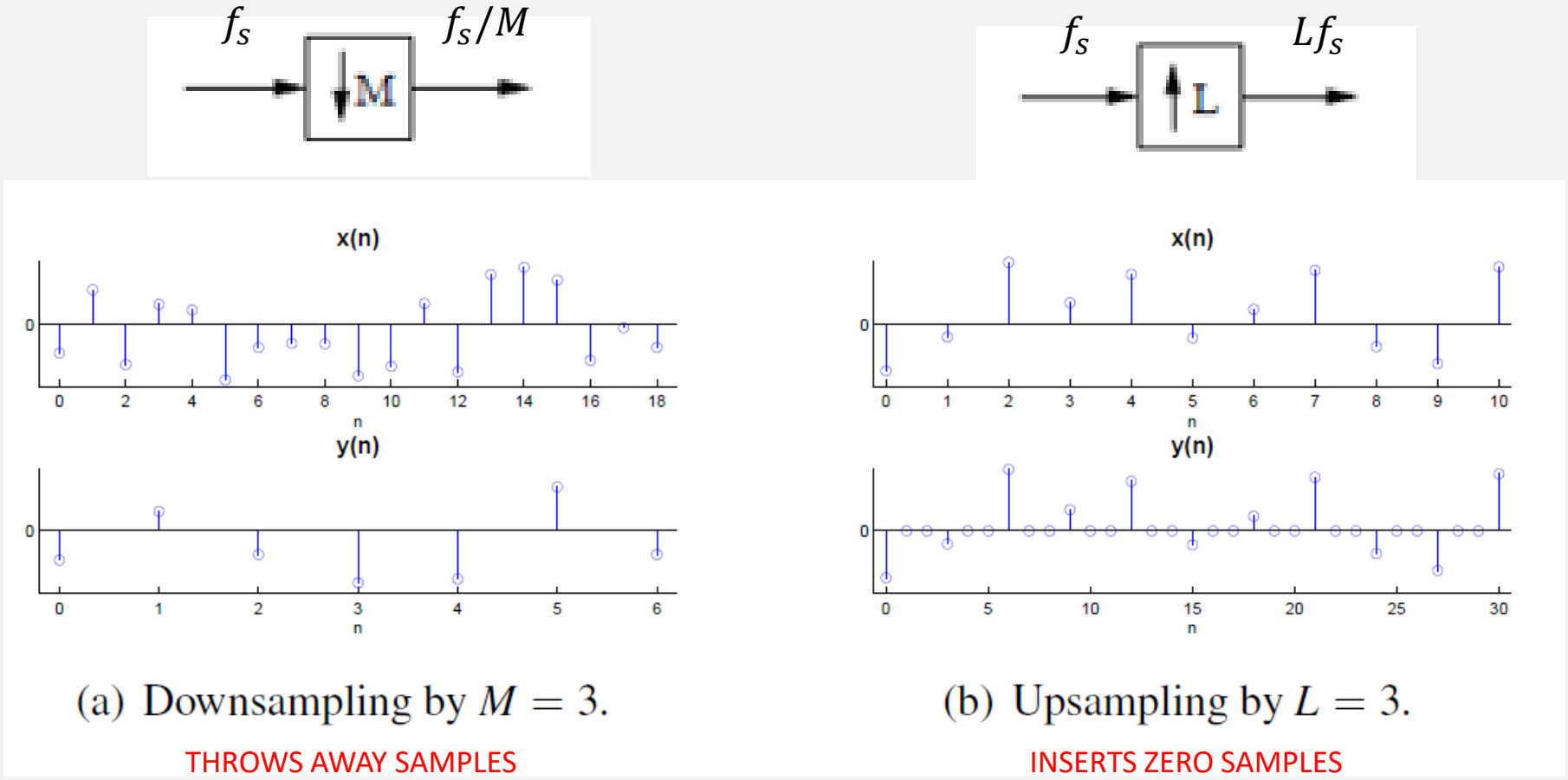
- Basic concepts
  - Downsampling / upsampling operations
  - Noble identities
  - Polyphase decomposition of signals and systems
- Implementing decimation and interpolation
  - Using noble identities and polyphase decomposition

# Motivation

- Multirate processing has many applications
  - Implementation of sample rate changers
  - Oversampling A/D conversion
  - Implementation of narrowband filters
  - Also role in implementation of filter banks
- Learning techniques for reducing computational needs
  - Noble identities, polyphase decomposition, commutation

# 1. Rate changing operations

Basic operations for multirate processing



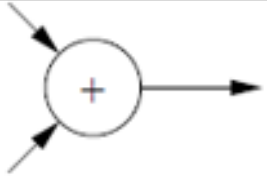
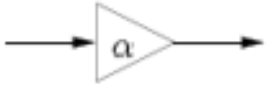
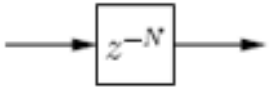
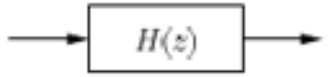

(a) Downsampling by $M = 3$.

THROWS AWAY SAMPLES
REDUCES SAMPLE RATE

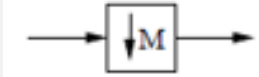
(b) Upsampling by $L = 3$.

INSERTS ZERO SAMPLES
INCREASES SAMPLE RATE

# Signal flow graph symbols and equations

| Name | Symbol | Output $y(n)$ | In Z-domain |
|---|---|---|---|
| Addition |  | $x_1(n) + x_2(n)$ | $Y(z) = X_1(z) + X_2(z)$ |
| Multiplication by $\alpha$ |  | $\alpha x(n)$ | $Y(z) = \alpha X(z)$ |
| Delay of $N$ time units |  | $x(n-N)$ | $Y(z) = z^{-N}X(z)$ |
| Downsampling by factor $M$ |  | $x(Mn)$ | $Y(z^M) = \frac{1}{M}\sum_{k=0}^{M-1} X(zW_M^k)$, where $W_M = e^{-j2\pi/M}$ |
| Upsampling by factor $L$ |  | $\begin{cases} x(n/L) & \text{if } n = kL \\ 0 & \text{otherwise} \end{cases}$ | $Y(z) = X(z^L)$ |
| System $H$ |  | | $Y(z) = H(z)X(z)$ |

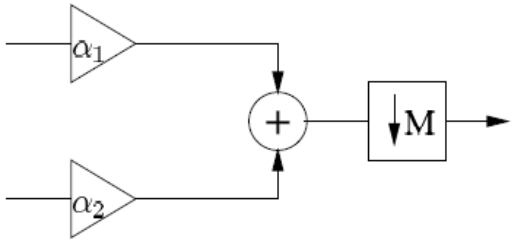
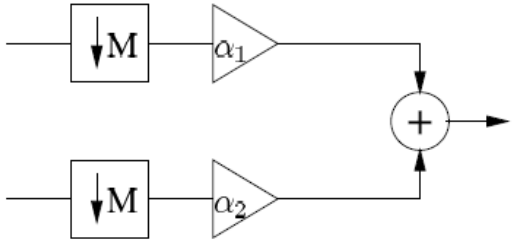
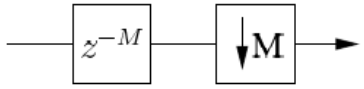
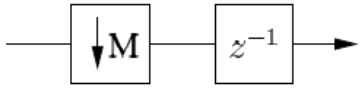
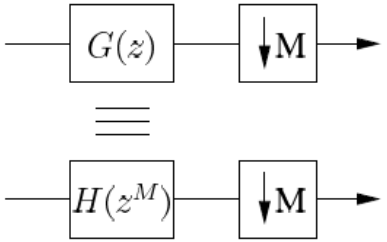
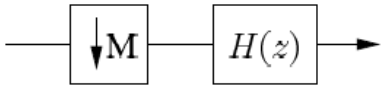
# 2. Noble identities

- 1-3 : related to downsampling 

- 4-6 : related to upsampling 

- Commutator

# Noble identities 1-3

| Name | Original structure | Transformed structure |
|---|---|---|
| Noble identity 1 |  |  |
| Noble identity 2 |  |  |
| Noble identity 3 |  |  |

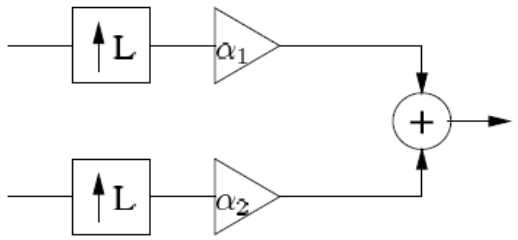
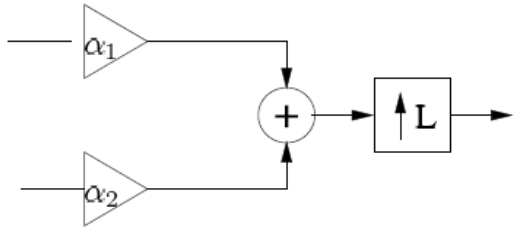
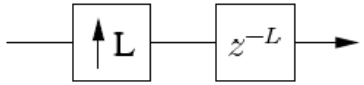
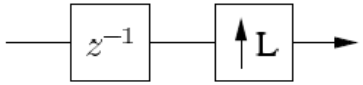
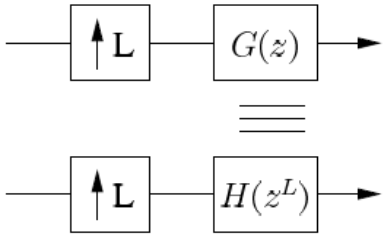
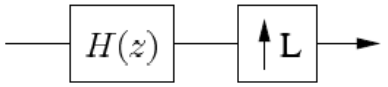
In the transformed structure, the rate of arithmetic operations is $(1/M)$th of the original structure.

The need for memory is reduced to $(1/M)$th of original structure.

Reduction of both arithmetic operation rate and memory to $(1/M)$th. Note that there is a constraint on the structure of G(z).

$G(z) = a + bz^{-M} + \text{c}\, z^{-2M} + \cdots$ ⇒ $\text{H}(z) = a + bz^{-1} + \text{c}\, z^{-2} + \cdots$

# Noble identities 4-6

| Name | Original structure | Transformed structure |
|---|---|---|
| Noble identity 4 |  |  |
| Noble identity 5 |  |  |
| Noble identity 6 |  |  |

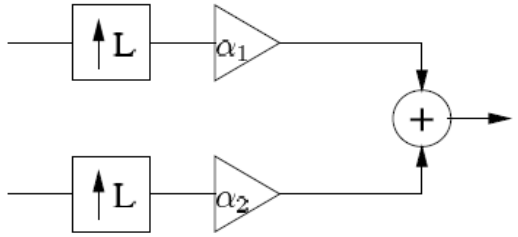
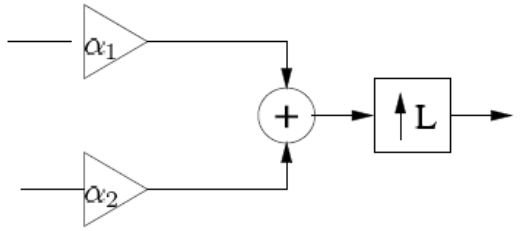
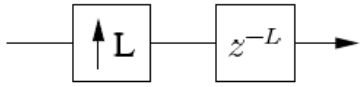
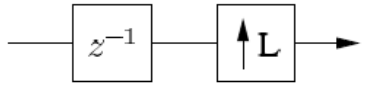
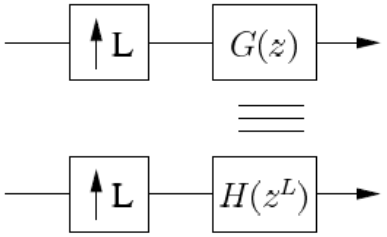
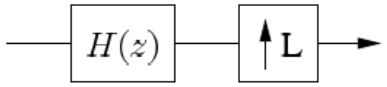
In the transformed structure, the rate of arithmetic operations is 1/*L*th of the original structure.

The need for memory is reduced to 1/*L*th of original structure.

Reduction of both arithmetic operation rate and memory to (1/*L*)th. Also here, there is a constraint on the structure of G(z).

$G(z) = a + bz^{-L} + \text{c}\, z^{-2L} + \cdots$ ⇒ $\text{H}(z) = a + bz^{-1} + \text{c}\, z^{-2} + \cdots$

# Noble identities 4-6

| Name | Original structure | Transformed structure |
|---|---|---|
| Noble identity 4 |  |  |
| Noble identity 5 |  |  |
| Noble identity 6 |  |  |

**Upsampling inserts zeros afterwards in transformed structures**, which looks strange. However, these mappings are applied in cases, where upsampling operations will be substituted by structures based on commutation.

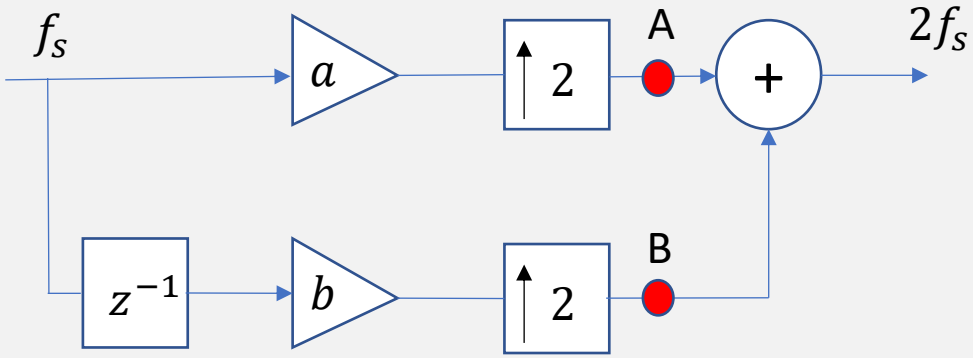
# Commutation & upsampling

- Consider the structure shown on the right, whose input is the sequence *x*(0), x(1), x(2), …
- At the points A and B, we have sequences

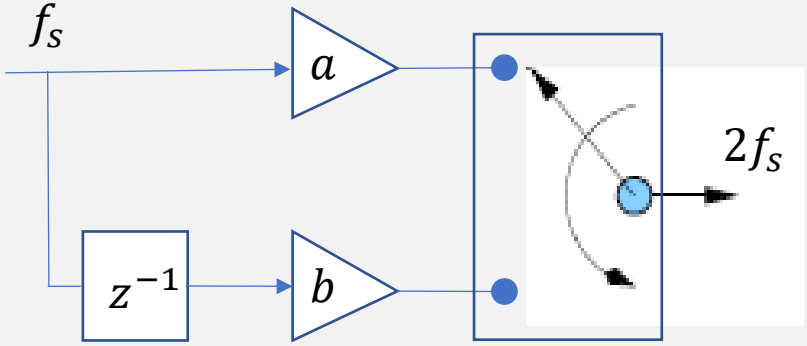
  A : *ax*(0) 0 *ax*(1) 0 *ax*(2) 0 *ax*(3) 0 …

  B : 0 *bx*(0) 0 *bx*(1) 0 *bx*(2) 0 *bx*(3) …
- Their sum makes up the output

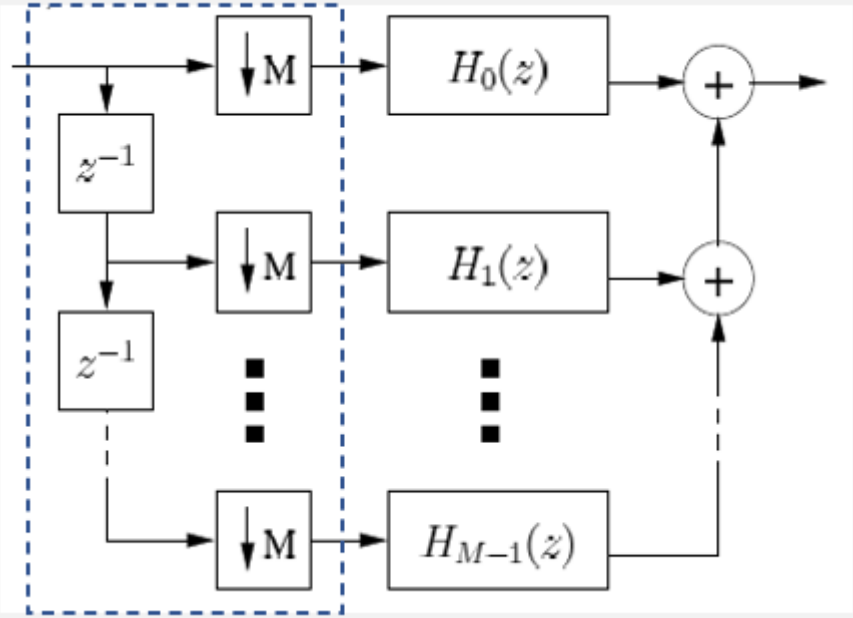
  *ax*(0) *bx*(0) *ax*(1) *bx*(1) *ax*(2) *bx*(2) *ax*(3) *bx*(3) …




- Computing sums is not sensible as the other term is always zero. Instead, we may pick samples from the outputs of the multipliers alternately. This is called **commutation**.
- As a result, **commutator** block replaces the upsampling / summing structure.
- Commutation operation can also be associated with the downsampling operation.

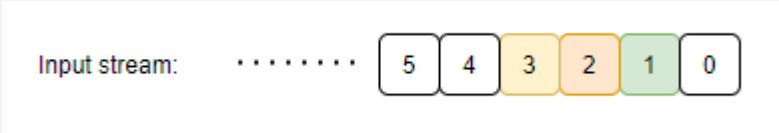


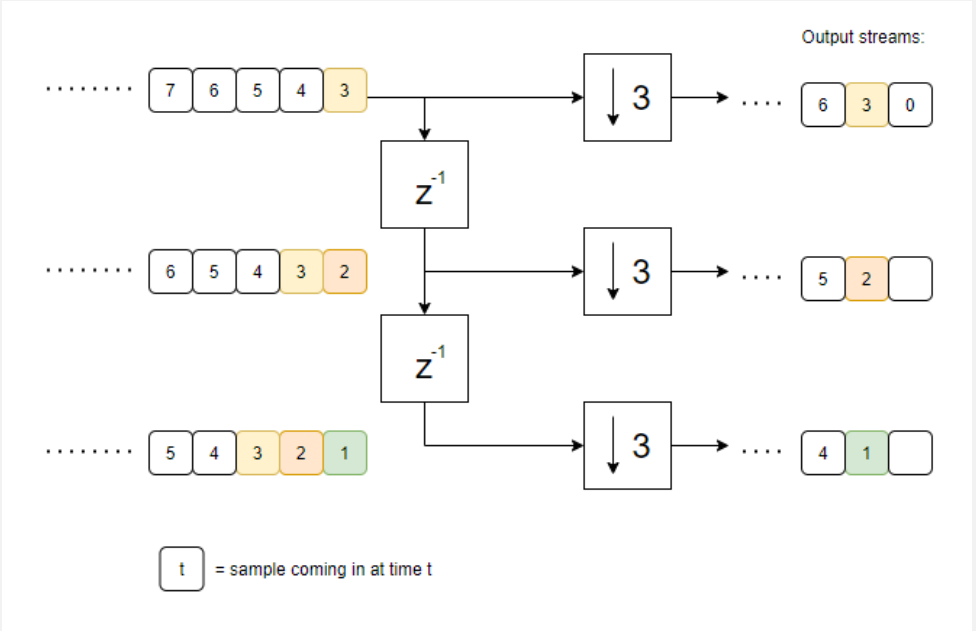
# Commutation & downsampling



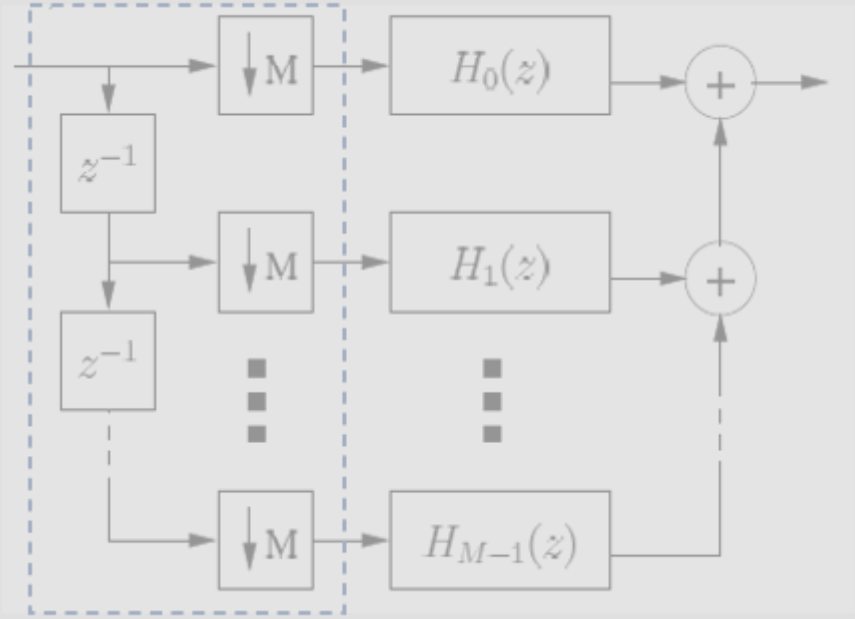

Whole input stream goes to the input of each downsampler.
Effectively, **downsamplers pick samples from different positions of the input stream**.

Example (M = 3)

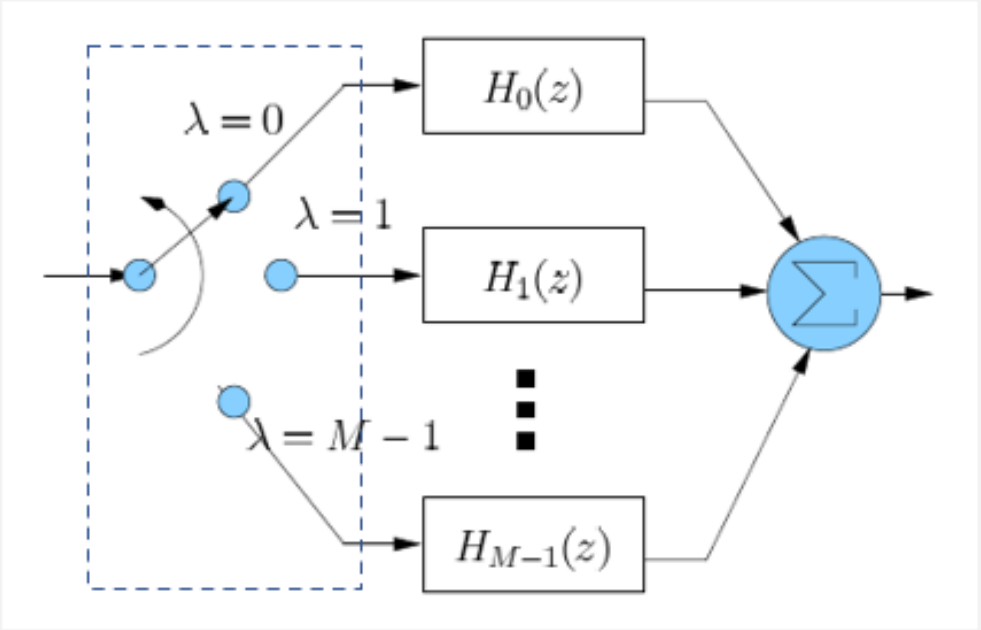


# Commutation & downsampling




Whole input stream goes to the input of each downsampler. Effectively, downsamplers pick samples from different positions of the stream.




Distribute samples to filter branches.
Note: first one goes to filter M-1, last to filter 0.

Once output of each filter has been updated, compute their sum (= system output).

# 3. Polyphase decomposition

- Noble identities provide a tool for deriving computationally efficient signal flow structures
- Accompanying tool is polyphase decomposition

# Polyphase decomposition of signals

**Discrete sampling** by factor *M*: every *M*th value of a signal $x(n)$ is retained and others set to zero.

This operation can be done for *M* different **phase offsets** $\lambda \in \{0,1,\dots,M-1\}$.

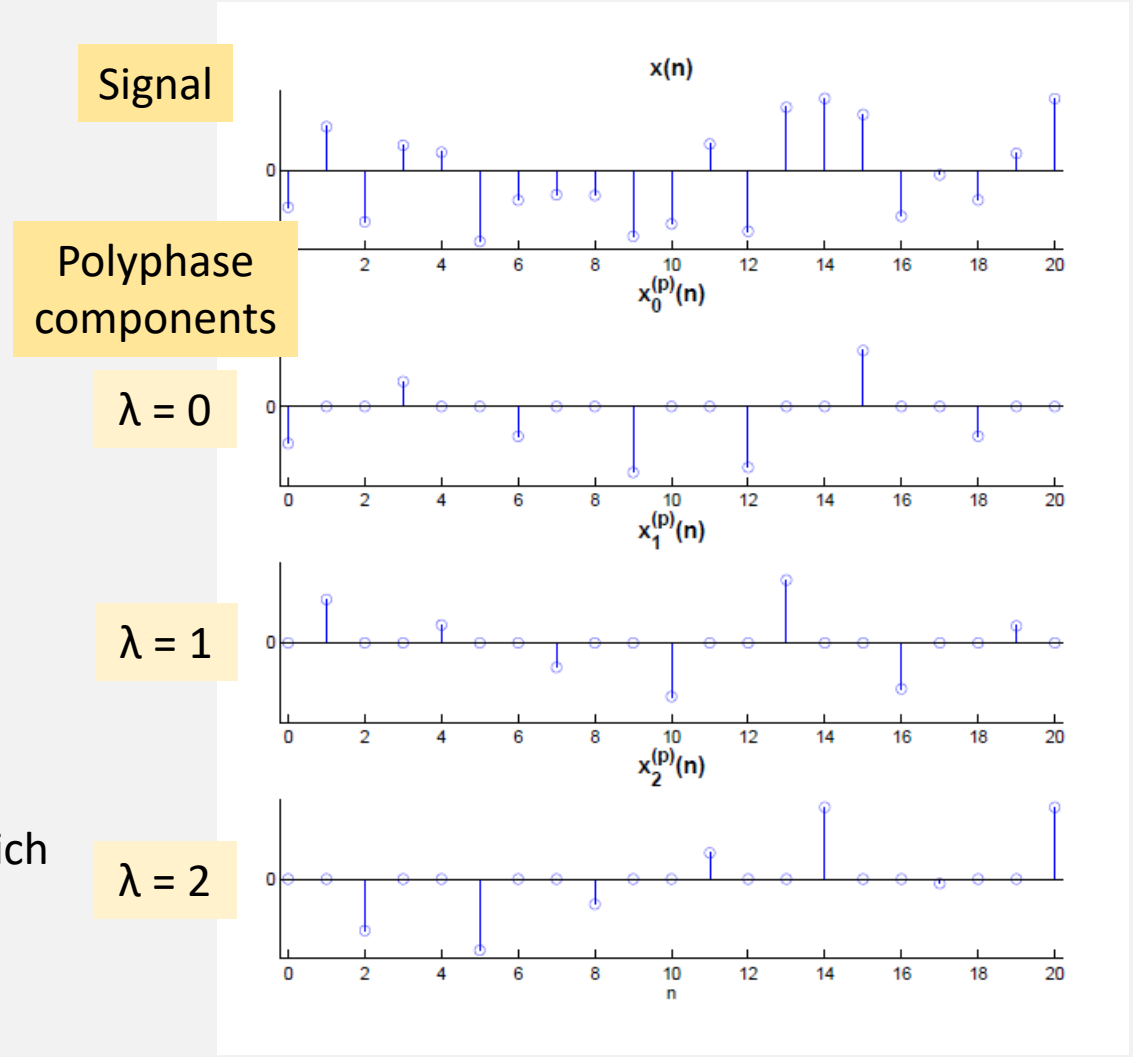
Resulting signals are called **polyphase components** of the signal $x(n)$ and they are denoted $x_\lambda^{(p)}(n)$.

$$x(n) = \sum_{\lambda=0}^{M-1} x_\lambda^{(p)}(n)$$

Based on this, the Z-transform of the signal represented as

$$X(z) = \sum_{\lambda=0}^{M-1} z^{-\lambda} X_\lambda^{(p)}(z^M)$$

where $X_\lambda^{(p)}(z)$ denotes the Z-transform of the sequence, which is constructed from those samples of $x_\lambda^{(p)}(n)$ which were retained from the signal $x(n)$. This is called **polyphase representation of the Z-transform**.

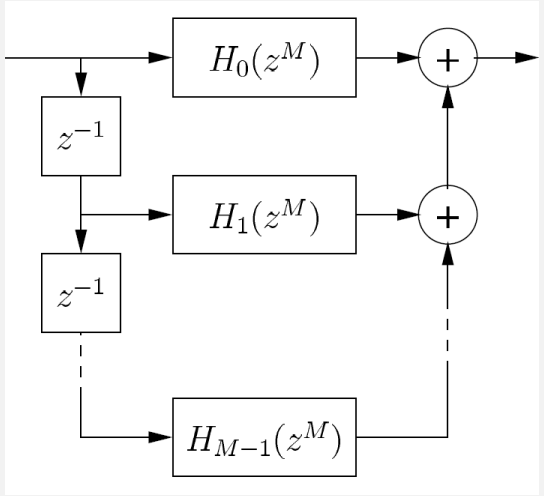


# Polyphase decomposition of systems

Similarly to signals, we can decompose systems. This is easily done for FIR filters, but can also be done for IIR filters.

Polyphase decomposition of the transfer function $H(z)$ for factor $M$ has the form

$$H(z) = \sum_{\lambda=0}^{M-1} z^{-\lambda} H_\lambda^{(p)}(z^M)$$




**Example**. FIR filter $H(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3} + a_4 z^{-4} + a_5 z^{-5}$

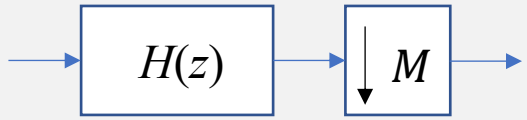
$M$=2 $$H(z) = \underbrace{(a_0 + a_2 z^{-2} + a_4 z^{-4})}_{H_0^{(p)}(z^2)} + z^{-1} \underbrace{(a_1 + a_3 z^{-2} + a_5 z^{-4})}_{H_1^{(p)}(z^2)}$$

$M$=3 $$H(z) = \underbrace{(a_0 + a_3 z^{-3})}_{H_0^{(p)}(z^3)} + z^{-1} \underbrace{(a_1 + a_4 z^{-3})}_{H_1^{(p)}(z^3)} + z^{-2} \underbrace{(a_2 + a_5 z^{-3})}_{H_2^{(p)}(z^3)}$$

**Note**. The polyphase decomposition contains terms of the form $H_\lambda^{(p)}(z^M)$, which is the form seen in the Noble Identities #3 and #6. (recall slides 7-8)

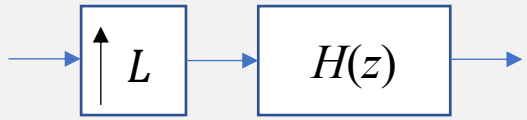
# 4. Decimation and interpolation

- In **decimation**, sample rate of the signal is reduced by a downsampler
- However, due to aliasing, downsampling must be preceeded by a (lowpass) filter, which reduces the bandwidth of the signal
- This is called anti-alias filtering




anti-alias filter

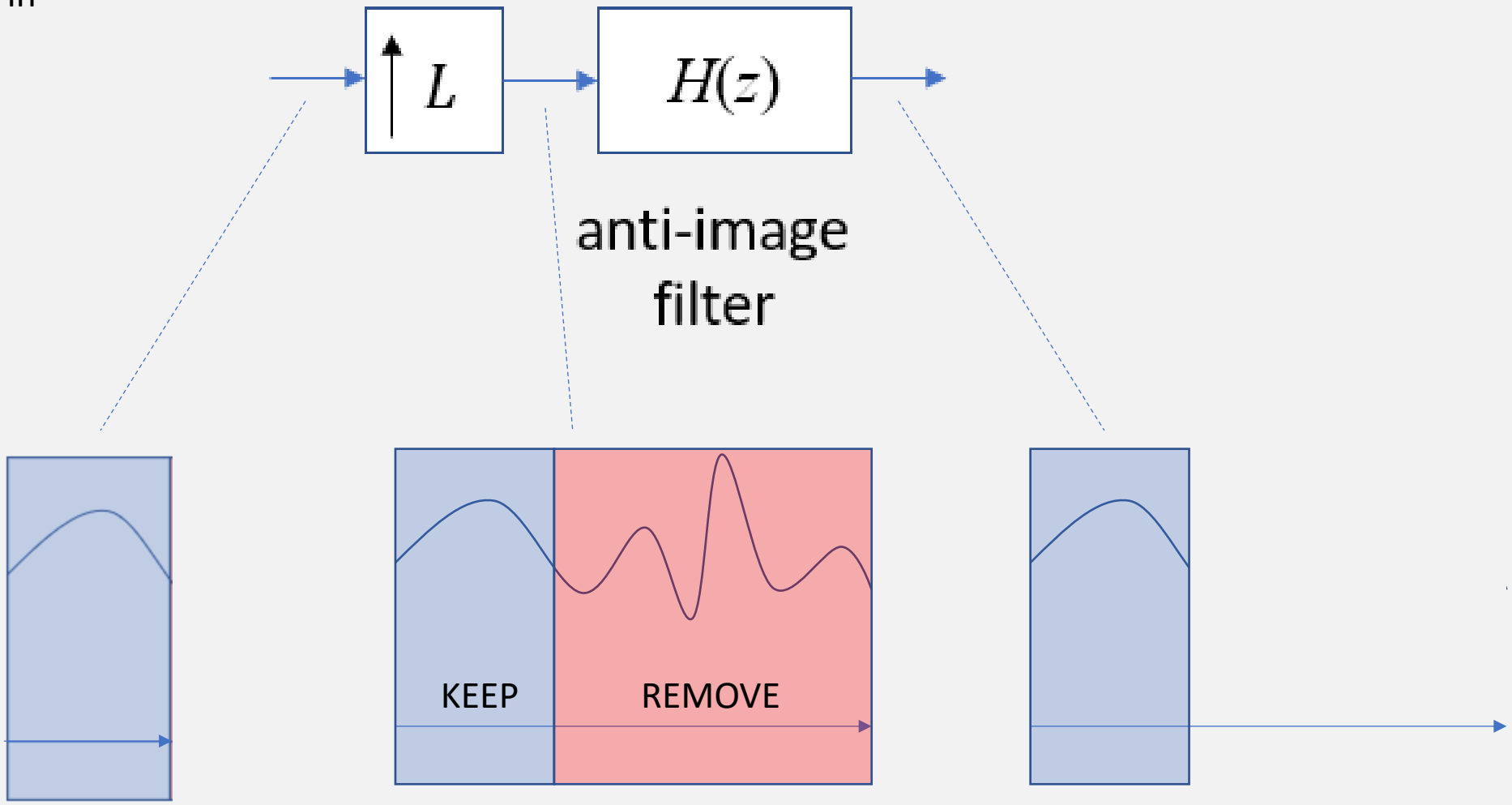
- In **interpolation**, sample rate of the signal is increased by an upsampler
- Upsampler just inserts zeros to the signal, so this is not the result we want. In the frequency domain, upsampling leads to imaging
- An anti-image filter which follows upsampling removes that imaging (interpolates new values to positions of zeros)




anti-image filter

- Anti-alias and anti-image filters in sample rate changers are typically implemented as FIR filters as they have linear phase response and better finite word length properties

Signal spectra in interpolation



# **Example**. Alias in image rescaling

Original (352x288)

Alias effects seen in high frequency regions

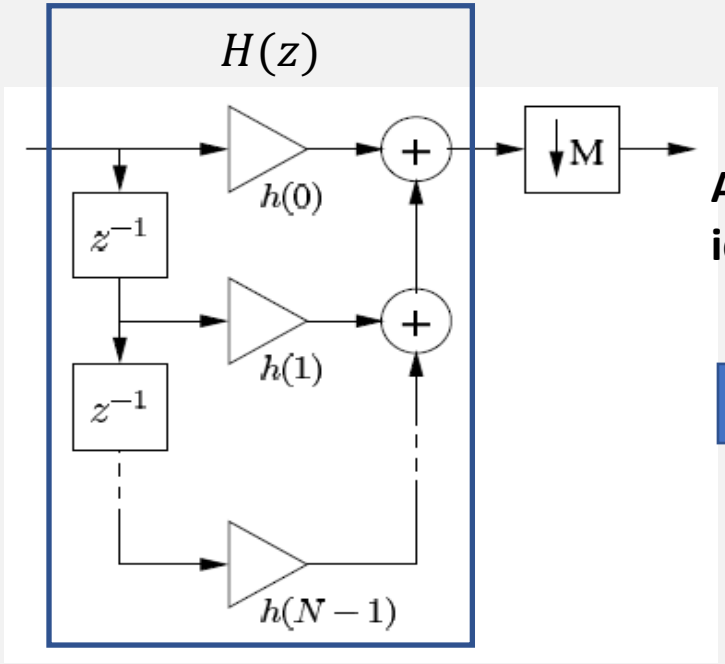
Downsampled (176x144)

Lowpass filtered+downsampled (176x144)

*Low-pass filtering before downsampling removes alias.*

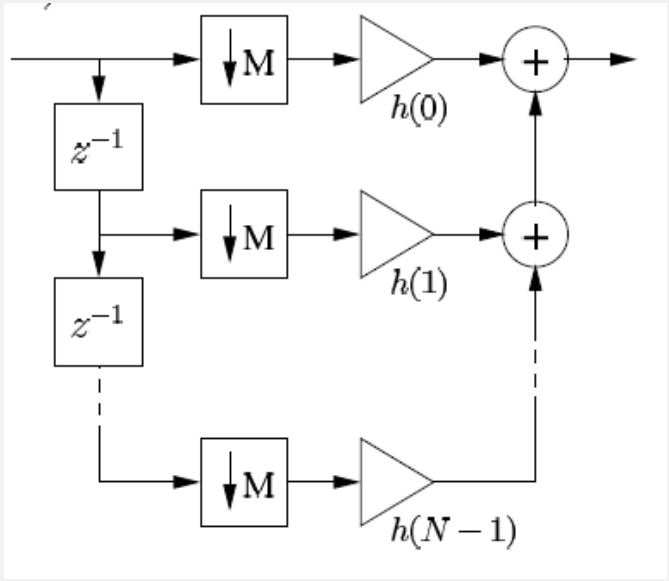
# Implementing decimation

- Anti-alias FIR filter $H(z) = \sum_{n=0}^{N-1} h(n) z^{-n}$



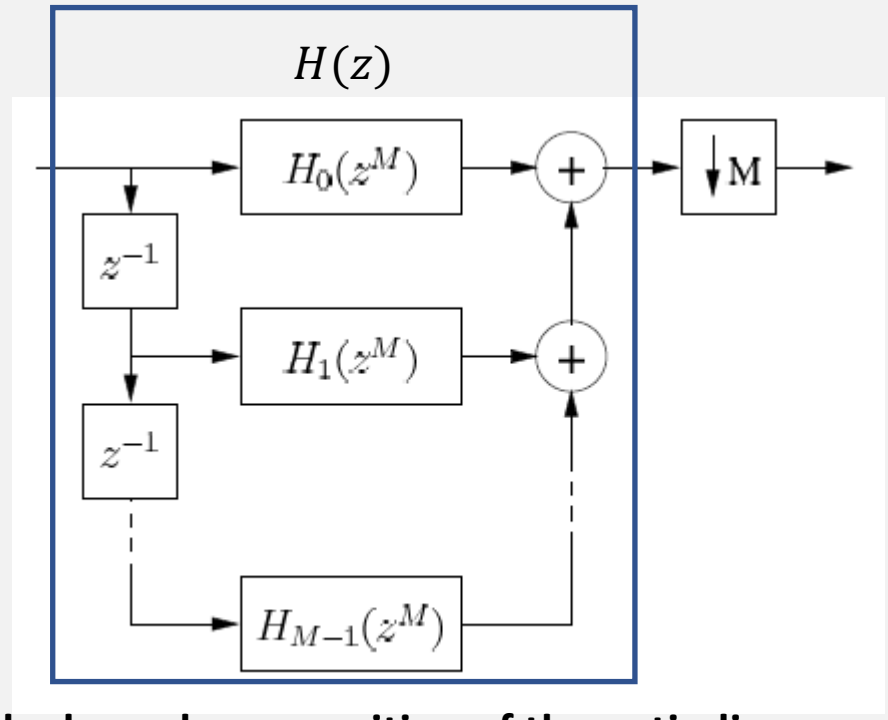

**Anti-alias FIR filter,**
**<u>direct form</u> structure**

**Applying noble identity #1**



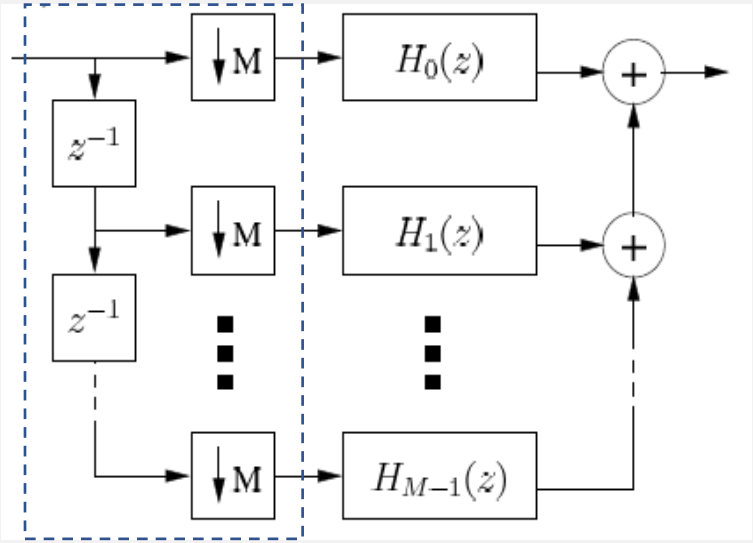

- The rate of arithmetic operations is reduced to (1/M)th fraction
- However, polyphase decomposition reveals another structure for implementation

$$H(z) = H_0(z^M) + z^{-1}H_1(z^M) + \ldots z^{-(M-1)}H_{M-1}(z^M)$$

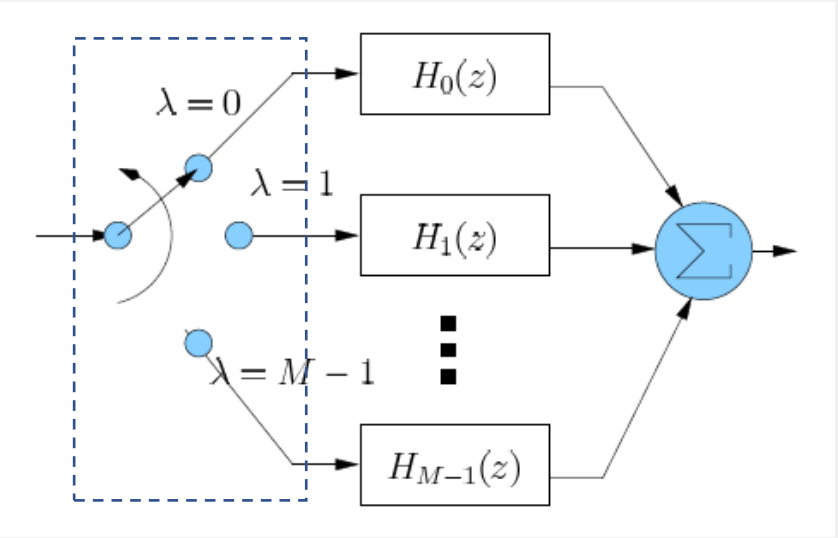



**Polyphase decomposition of the anti-alias filter, direct form structure**

**Applying noble identities #1 and #3**




**Commutation of the input**

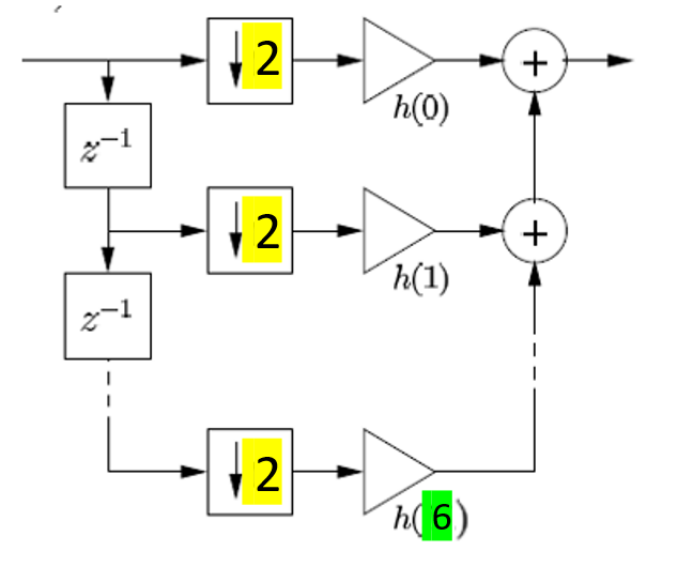


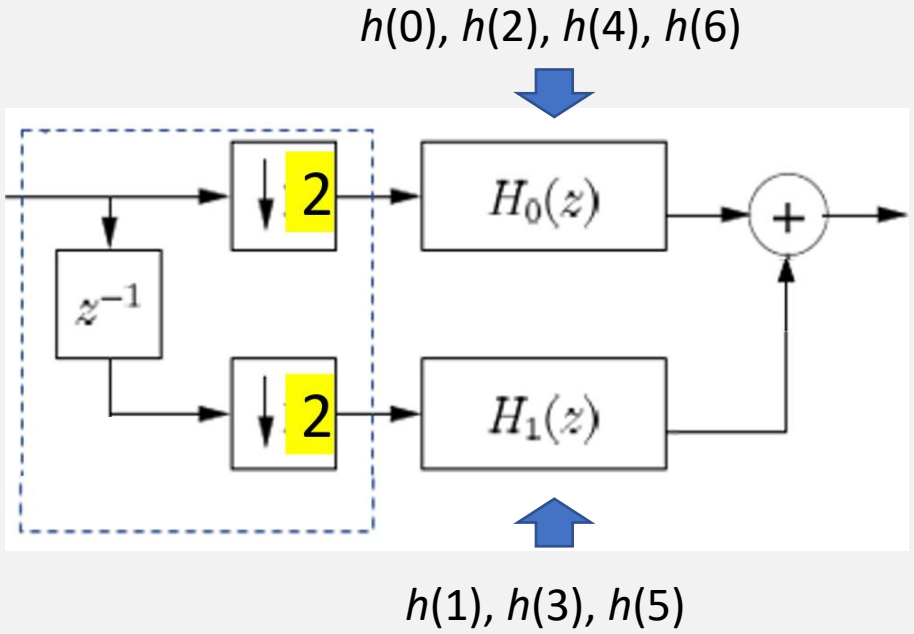

Commutator puts the first sample to the branch λ = *M*-1, second one to λ = *M*-2, and so on, the last one to λ = 0. Then, as all outputs of polyphase component filters have been updated, the sum can be computed to produce one output sample.

Polyphase decomposition helps to find the commutator based structure.

# Example. Comparison of structures

- Two solutions were obtained above
- Assume that the filter has N=7 coefficients and the downsampling factor is 2
- We get




- Both compute the same output
- But, the right one can be implemented using commutator

SKIPPED IN LECTURE

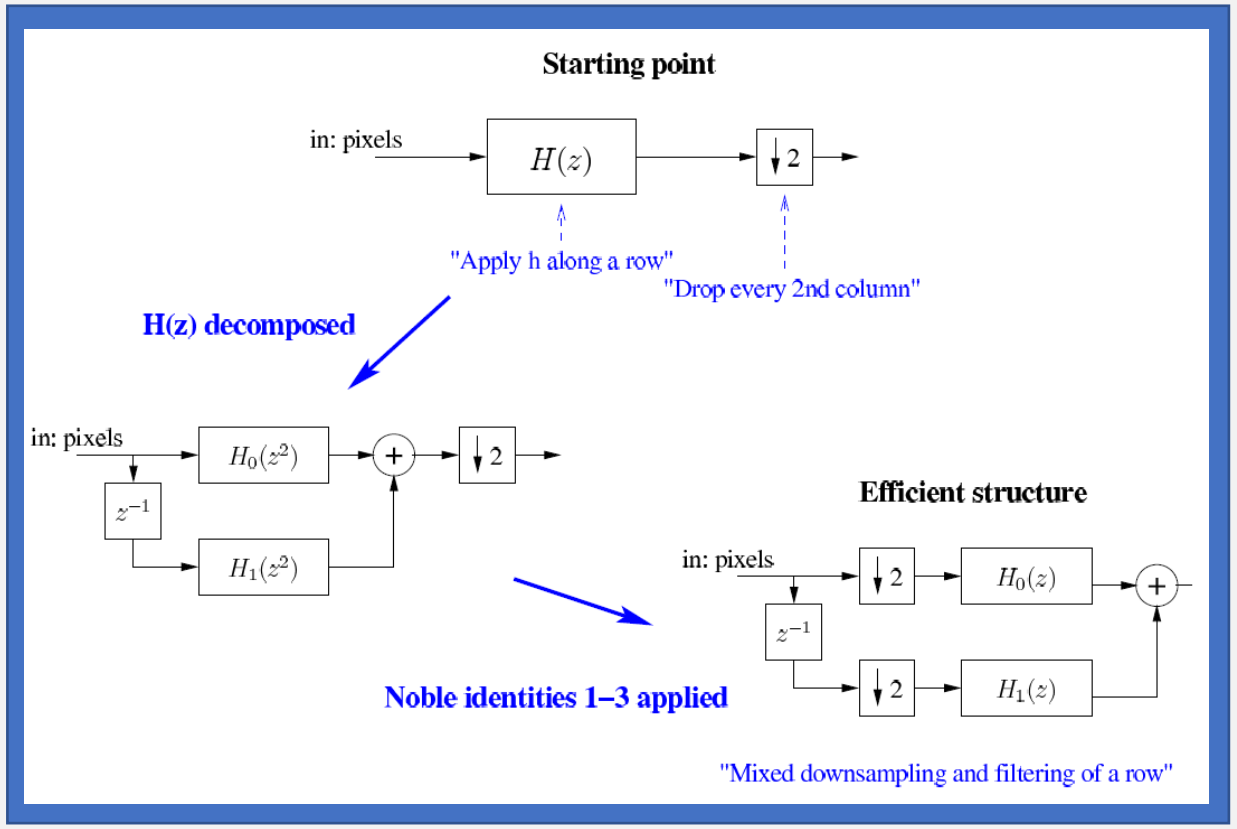
# **Example**. Image rescaling

- Lowpass filtering in the example used a separable 2-D filter
  - Can be implemented as 1-D filtering of the image rows followed by 1-D filtering of the columns
  - Used 1-D filtering can be implemented with polyphase decomposition and noble identities


Lowpass filtered+downsampled (176x144)

$$\mathbf{h} = [1/16, 1/4, 3/8, 1/4, 1/16]^T$$

$$\mathbf{h}\mathbf{h}^T = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}.$$



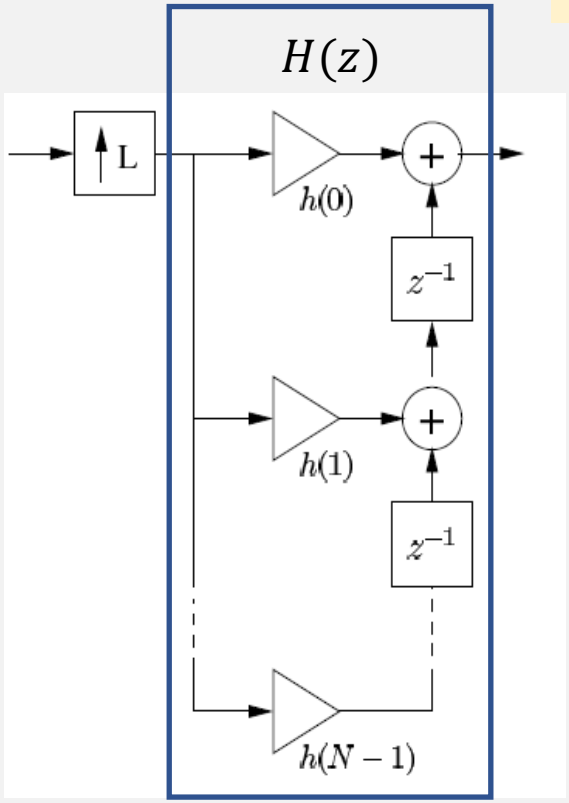

$$H(z) = (1/16) + (1/4)z^{-1} + (3/8)z^{-2} + (1/4)z^{-3} + (1/16)z^{-4}$$

$$H(z) = \underbrace{(1/16) + (3/8)z^{-2} + (1/16)z^{-4}}_{H_0(z^2)} + z^{-1}\underbrace{((1/4) + (1/4)z^{-2})}_{H_1(z^2)}$$

# Implementing interpolation

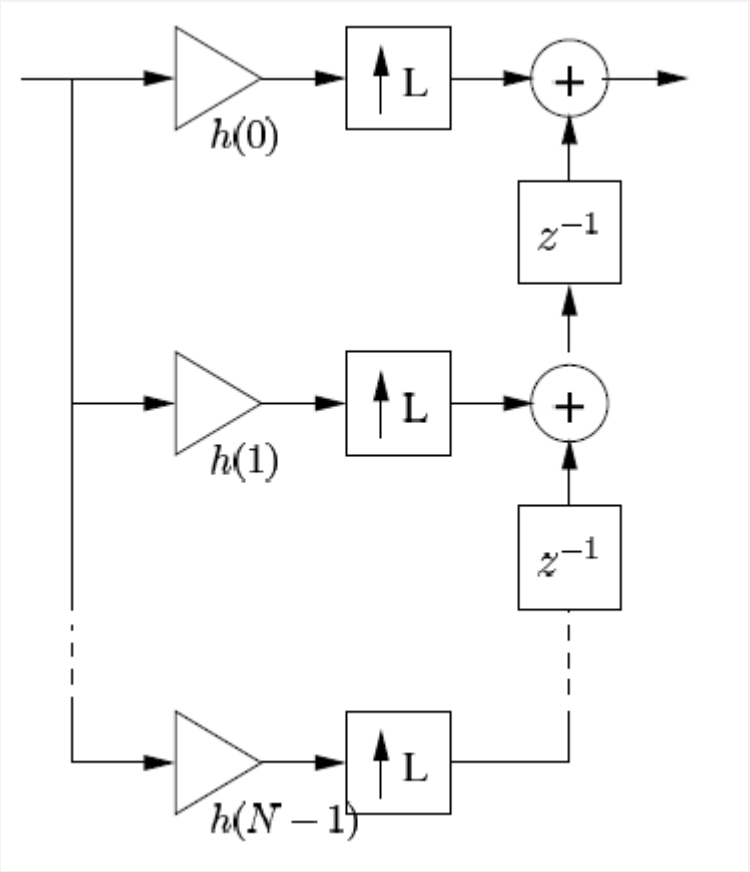
- Anti-image FIR filter

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n}$$



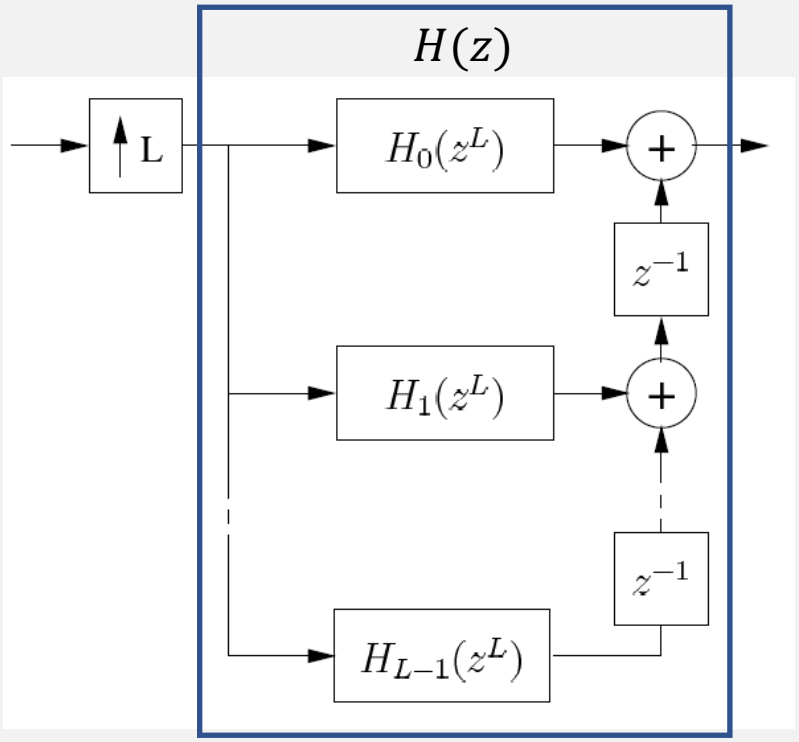

**Anti-image FIR filter, as <u>transposed direct</u> form structure**

**Applying noble identity #4**



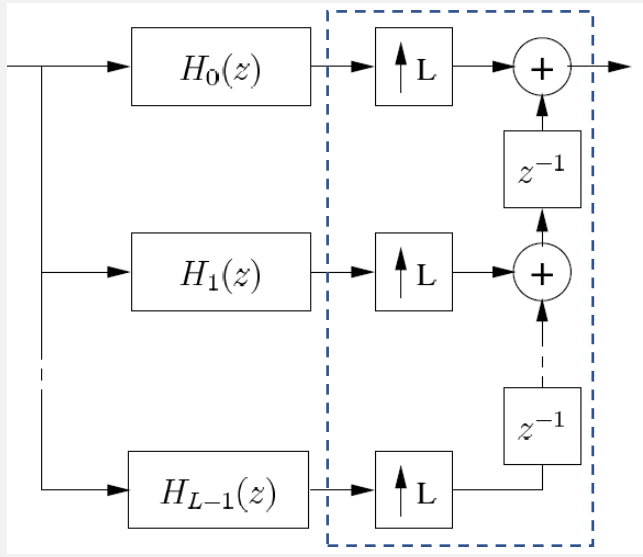

- The rate of arithmetic operations is reduced to (1/L)th fraction.
- Zeros after upsampling & addition – How to? Consider polyphase decomposition …

$$H(z) = H_0(z^L) + z^{-1}H_1(z^L) + \ldots z^{-(L-1)}H_{L-1}(z^L)$$

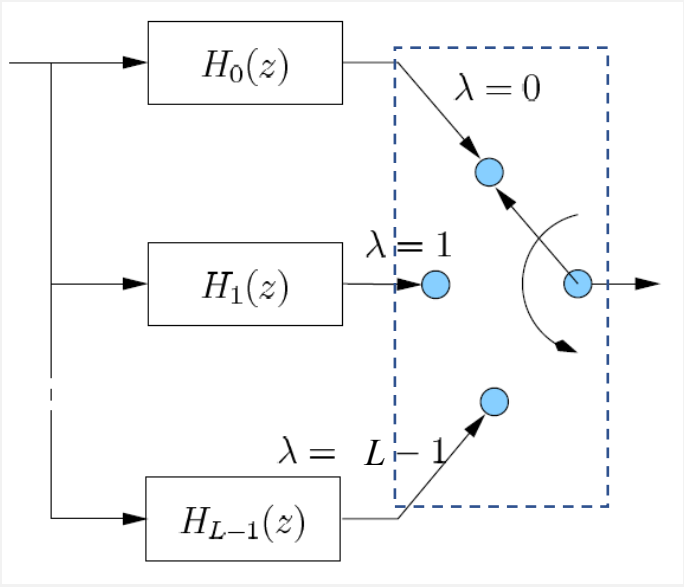



**Polyphase decomposition of the anti-image filter, transposed direct form**

**Applying noble identity #6**



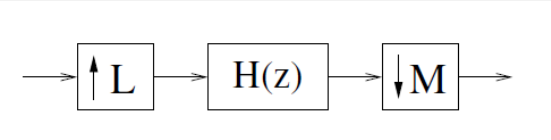

**Commutation of the output**




For each input sample: outputs of all component filters are updated. Commutator picks samples from each output to make up the output stream, first from the branch λ = *0*, then from λ = 1, and so on, finally from λ = *L*-1. For each input sample, we get L samples out.

# Design Task 4

- Problem 1 – Basic concepts of multirate processing
  - Sample rate conversion




- Problem 2 – Narrowband FIR filtering
- Problem 3 – Fixed-point properties of CIC filtering

  - Background for problems 2 & 3 discussed next Monday
  - Readings: intro4a.pdf & intro4b.pdf