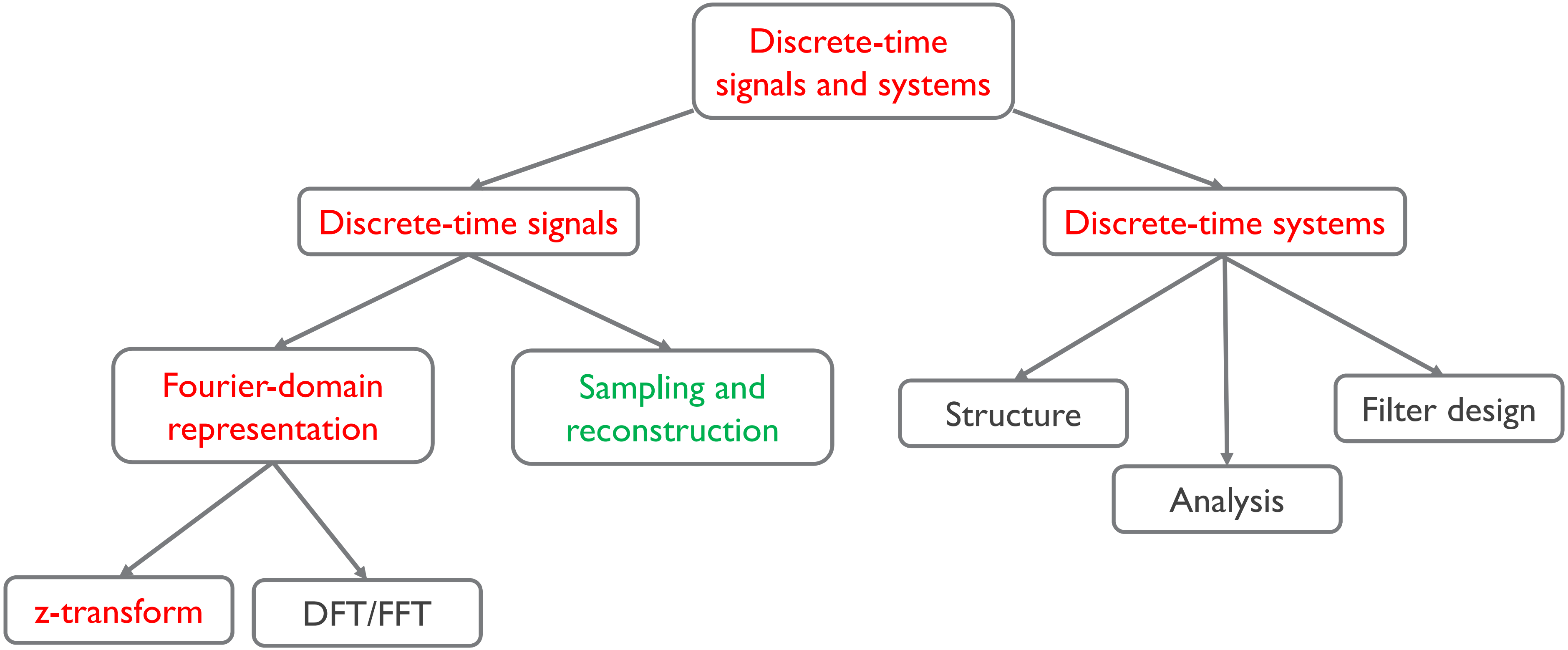
Digital Signal Processing

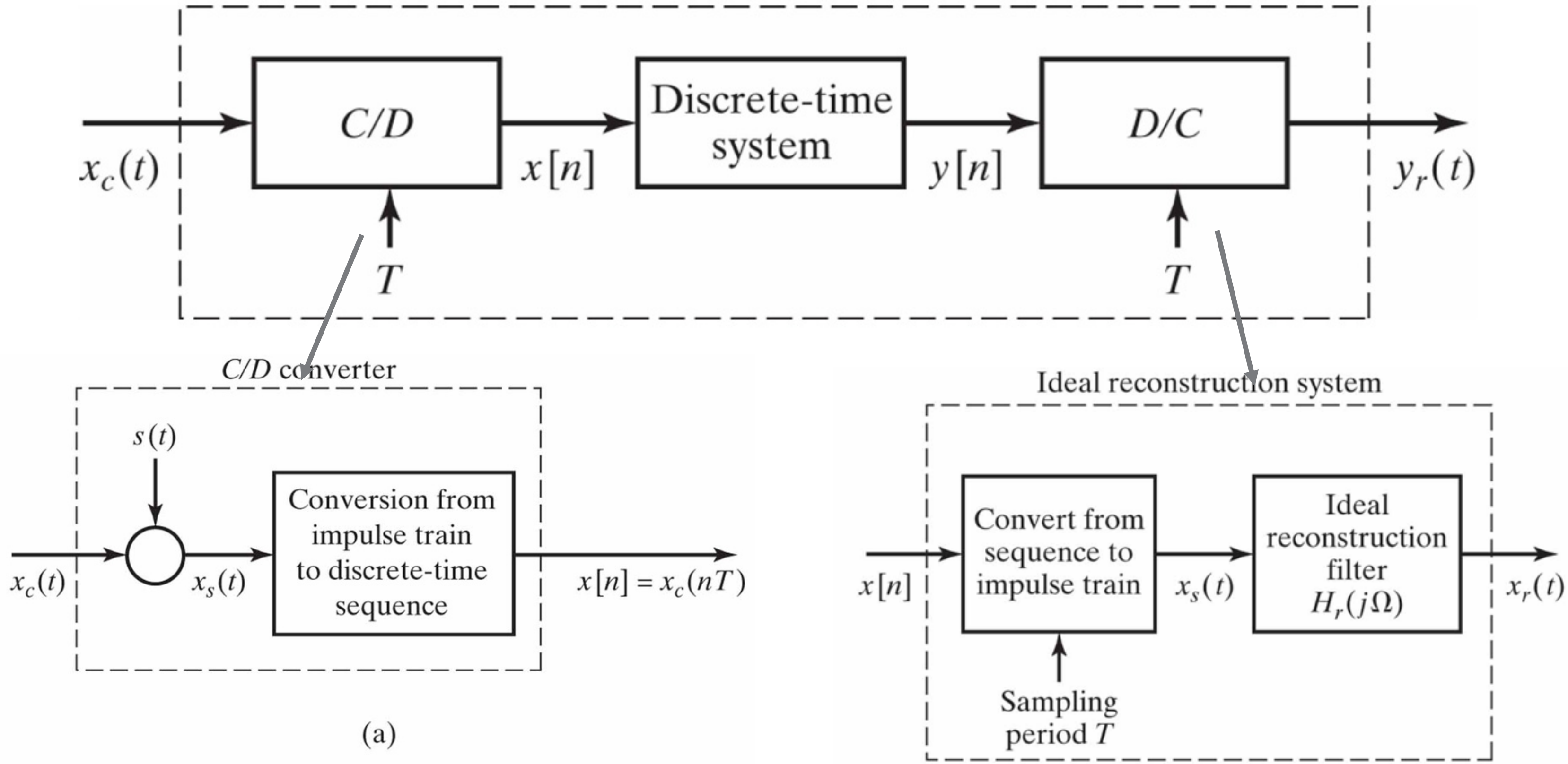
POSTECH
Department of Electrical Engineering
Junil Choi

Course at glance



Discrete-Time Processing of Continuous-Time Signals

Overall block diagram



- Overall system is continuous-time processing
- Continuous-time processing of discrete-time signals also possible

Output signal

- Necessary conditions
  - The discrete-time system is LTI
  - Continuous-time signal $x_c(t)$ is bandlimited
  - Sampling rate Ω_s is at or above the Nyquist rate $2\Omega_N$

- If all conditions are satisfied, the output signal becomes

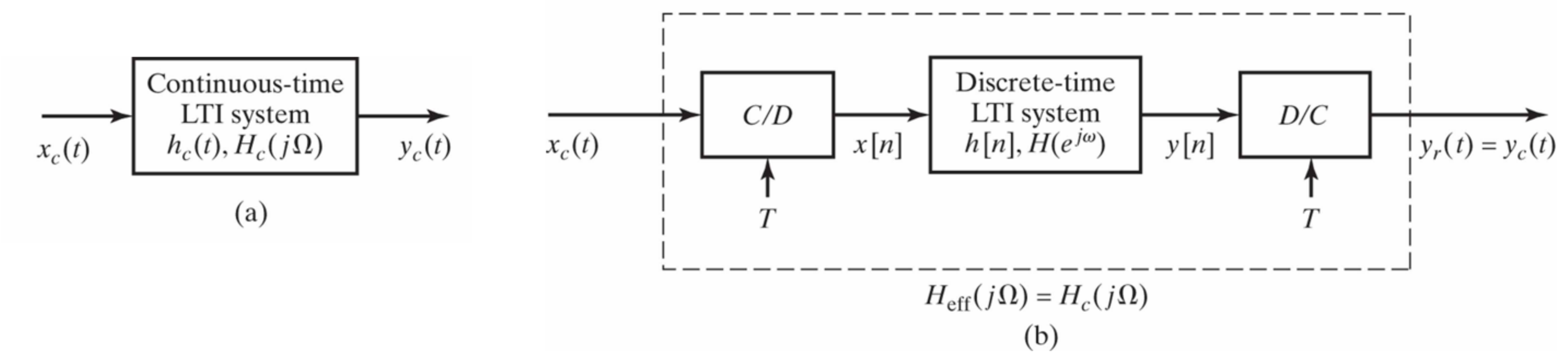
$$Y_r(j\Omega) = H_{\text{eff}}(j\Omega)X_c(j\Omega)$$

where

$$H_{\text{eff}}(j\Omega) = \begin{cases} H(e^{j\Omega T}), & |\Omega| < \pi/T \\ 0, & |\Omega| \geq \pi/T \end{cases}$$

Cutoff frequency of ideal lowpass filter

Impulse invariance



- Want to implement the continuous-time impulse response $h_c(t)$ using discrete-time system $h[n]$ or vise versa

- How to design $h[n]$ based on $h_c(t)$?

Impulse invariance

- Recall $H_{\text{eff}}(j\Omega) = \begin{cases} H(e^{j\Omega T}), & |\Omega| < \pi/T \\ 0, & |\Omega| \geq \pi/T \end{cases}$
- We want to have $H_{\text{eff}}(j\Omega) = H_c(j\Omega)$

$$\Rightarrow \quad H(e^{j\omega}) = H_c(j\omega/T), \quad |\omega| < \pi$$

- In time-domain: $h[n] = Th_c(nT)$

$$H(e^{j\omega}) = T\frac{1}{T}\sum_{k=-\infty}^{\infty} H_c\left(j\left(\frac{\omega}{T} - \frac{2\pi k}{T}\right)\right)$$

$$= H_c\left(j\frac{\omega}{T}\right), \quad |\omega| < \pi$$

Because $H_c(j\Omega) = 0, \quad |\Omega| \geq \pi/T$

Changing Sampling Rate Using Discrete-Time Processing

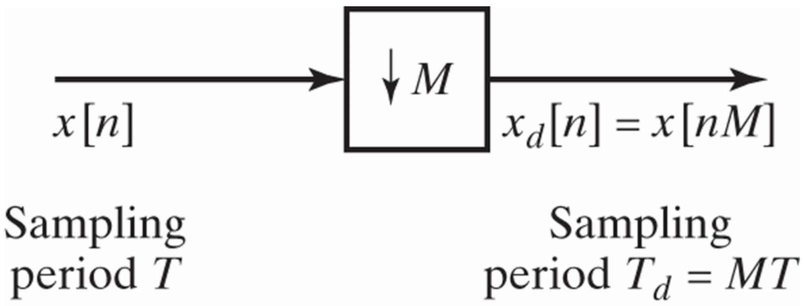
Resampling

- Sampling with sampling period *T*: $x[n] = x_c(nT)$

- Often necessary to change the sampling rate of a discrete-time signal
$$x_1[n] = x_c(nT_1), \text{ with } T \neq T_1$$
  - Resizing digital images
  - Video/audio conversion
- Direct approach is to reconstruct $x_c(t)$ from $x[n]$ and resample with sampling period T_1
  - Not a practical approach due to non-ideal hardware
  - Near-ideal filters are $$$$$$$
- Can we change the sampling rate by only dealing with discrete-time operations? YES!

Downsampling

Decreasing sampling rate by integer factor



- Usually called "downsampling"
- Sampling rate can be reduced by "sampling" the original sampled sequence
  - Original sampled sequence $x[n] = x_c(nT)$
  - New "sampled" sequence $x_d[n] = x[nM] = x_c(nMT)$
  - Keep one sample out of every *M* samples

  ➔ Operation called "compressor"
- The new sequence $x_d[n]$ is identical to the sequence obtained from $x_c(t)$ with the sampling period $T_d = MT$

Is reconstruction possible?

- Original sampling rate $\Omega_s = 2\pi/T$

- If $X_c(j\Omega) = 0$ for $|\Omega| \geq \Omega_N$, $x_c(t)$ can be reconstructed from $x_d[n]$ if

$$\pi/T_d = \pi/(MT) \geq \Omega_N \Rightarrow 2\pi/T_d \geq 2\Omega_N$$

- Sampling rate can be reduced to $1/M$ without aliasing if the original sampling rate *T* is at least M times the Nyquist rate

Frequency-domain representation

- DTFT of $x[n] = x_c(nT)$ is

$$X(e^{j\omega}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{T} - \frac{2\pi k}{T}\right)\right]$$

- DTFT of $x_d[n] = x[nM] = x_c(nT_d)$ with $T_d = MT$

$$X_d(e^{j\omega}) = \frac{1}{T_d}\sum_{r=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{T_d} - \frac{2\pi r}{T_d}\right)\right]$$

$$= \frac{1}{MT}\sum_{r=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{MT} - \frac{2\pi r}{MT}\right)\right]$$

Frequency-domain representation

- We can write $r = i + kM$ for $-\infty < k < \infty$ and $0 \leq i \leq M-1$

$$X_d(e^{j\omega}) = \frac{1}{MT}\sum_{r=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{MT} - \frac{2\pi r}{MT}\right)\right]$$

$$= \frac{1}{M}\sum_{i=0}^{M-1}\left\{\frac{1}{T}\sum_{k=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{MT} - \frac{2\pi k}{T} - \frac{2\pi i}{MT}\right)\right]\right\}$$

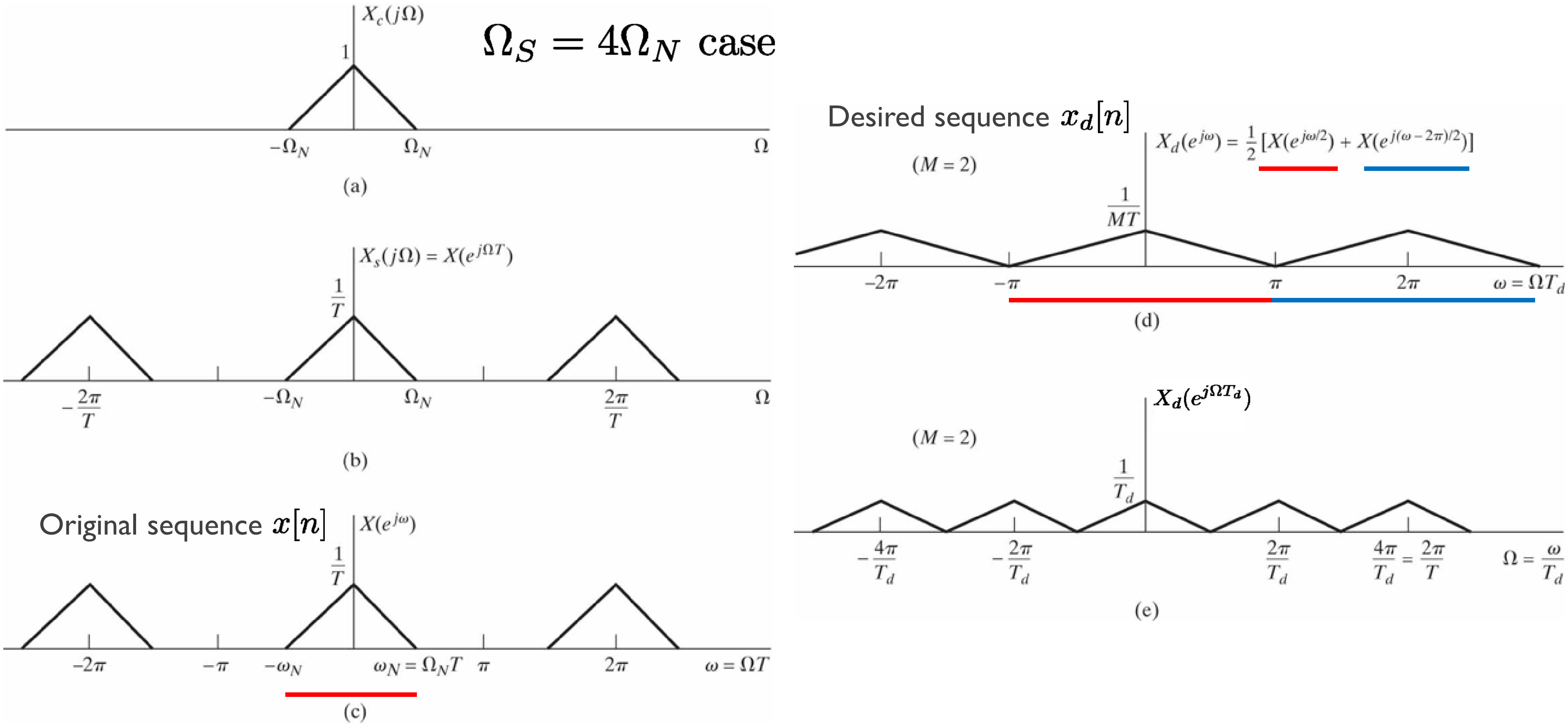
- Using DTFT of x[n]

$$X(e^{j(\omega-2\pi i)/M}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c\left[j\left(\frac{\omega - 2\pi i}{MT} - \frac{2\pi k}{T}\right)\right]$$

- We have $X_d(e^{j\omega}) = \frac{1}{M}\sum_{i=0}^{M-1} X(e^{j(\omega-2\pi i)/M})$

Scaled-copies of $X(e^{j\omega})$

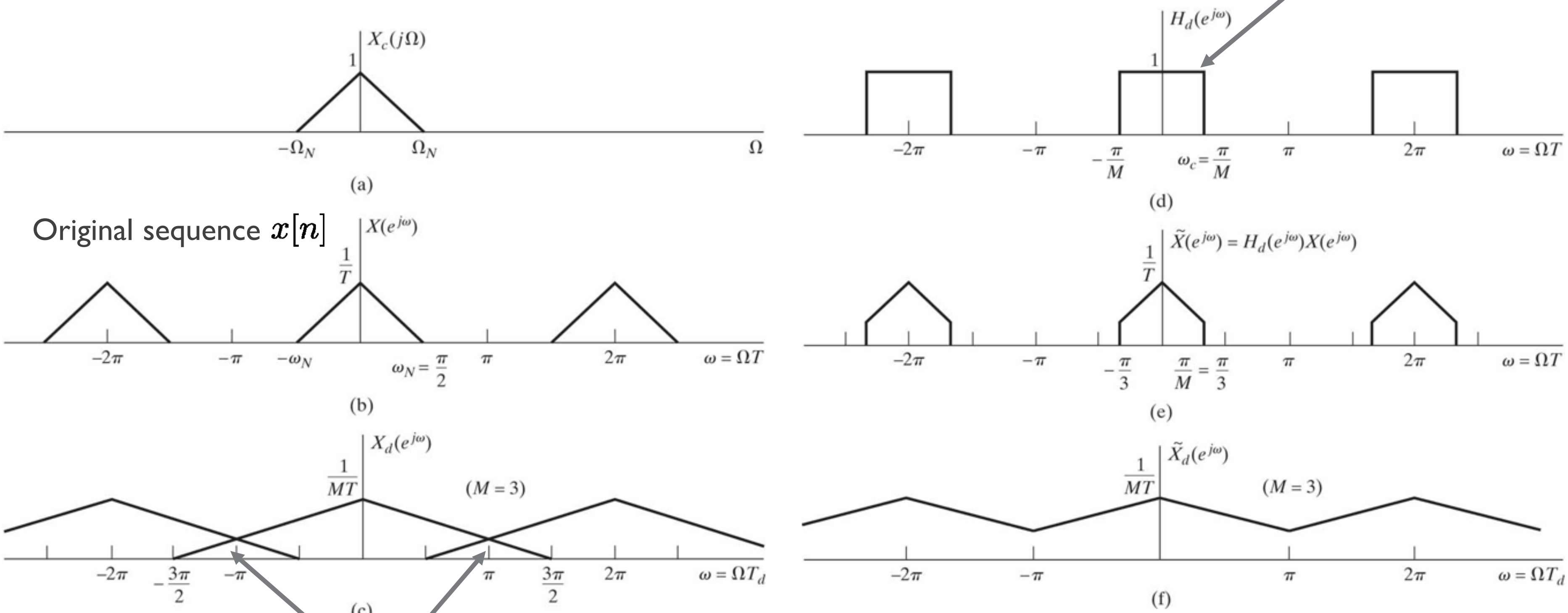
Example – no aliasing



Example – with aliasing

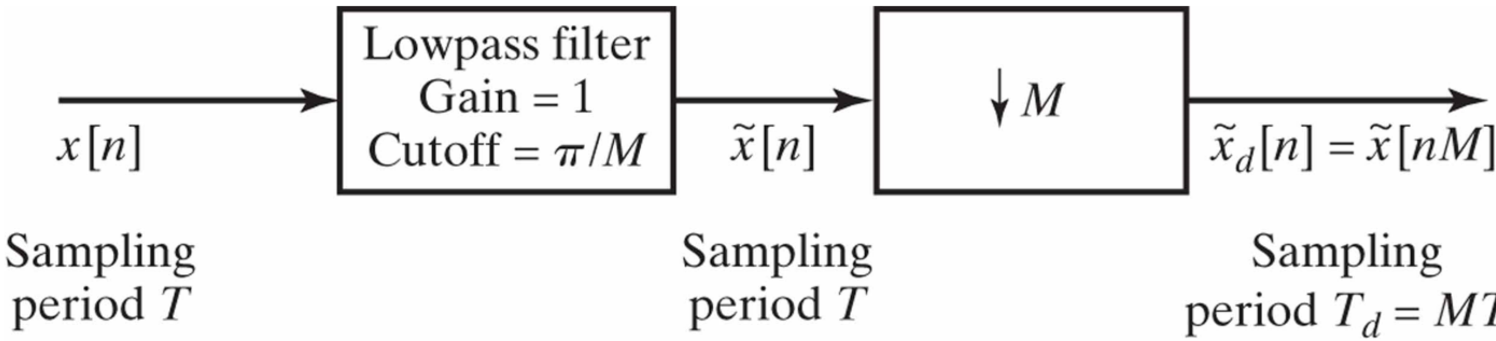
Prefiltering to avoid aliasing

Original sequence $x[n]$



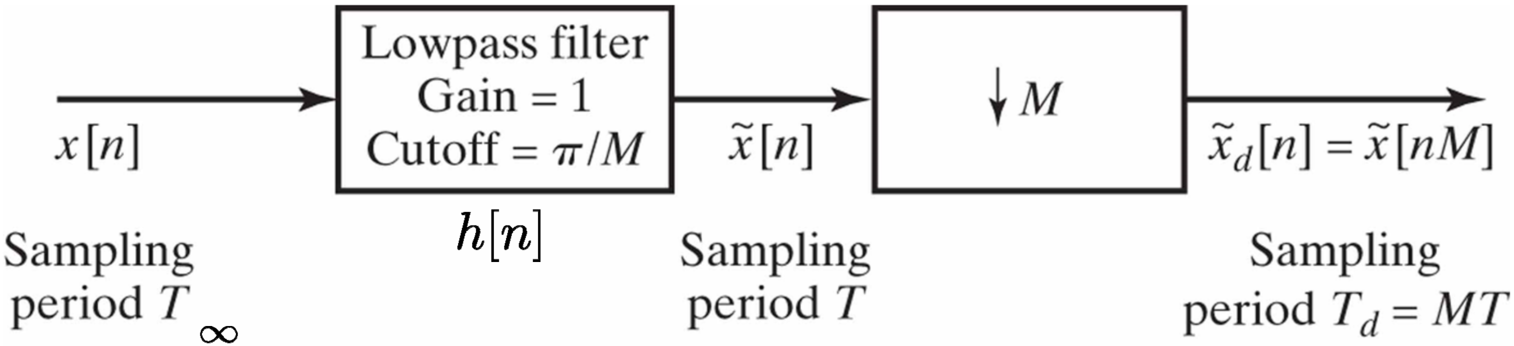
Aliasing occurs! To avoid aliasing, $\omega_N M \leq \pi$

A general downsampling system



- Lowpass filter to avoid aliasing
- The system also called "decimator" (in general, "downsampling")

Efficient implementation of downsampling

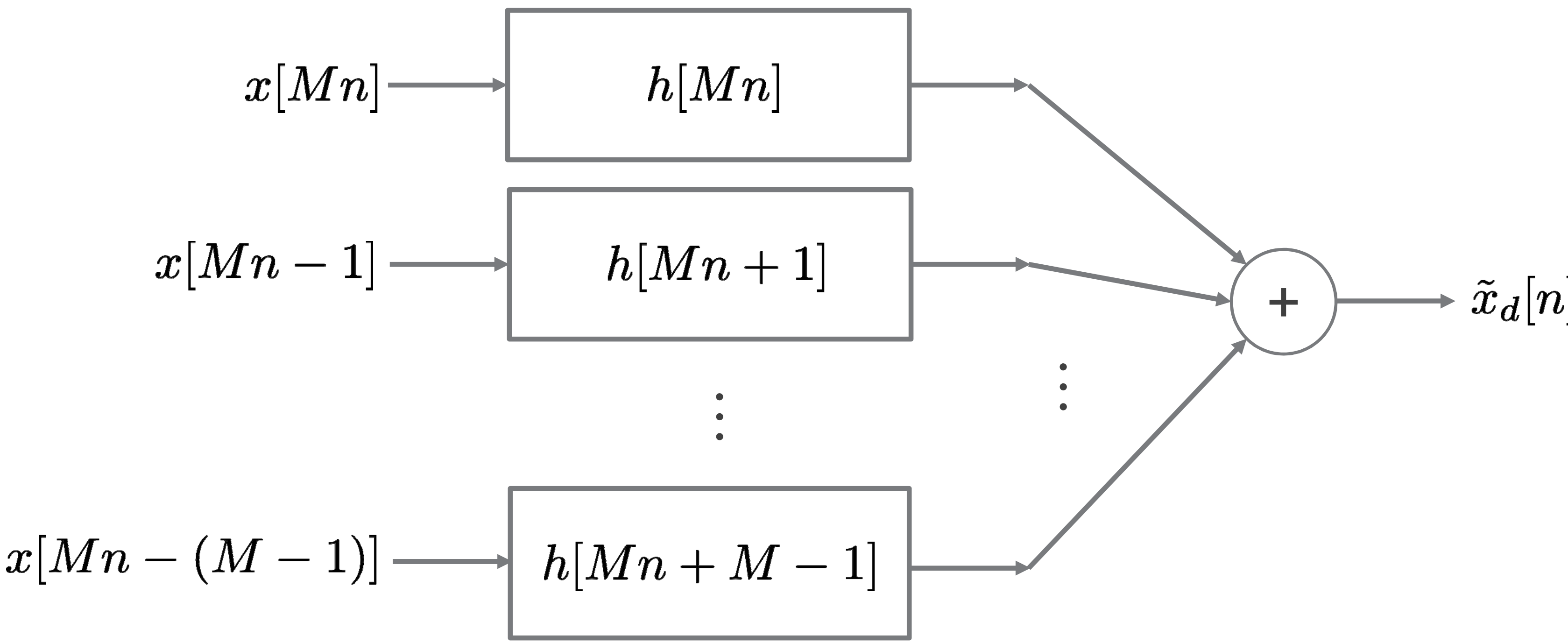


- $\tilde{x}[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k]$

$$\tilde{x}_d[n] = \sum_{k=-\infty}^{\infty} h[k]x[Mn-k] = \sum_{\ell=0}^{M-1} \sum_{k'=-\infty}^{\infty} h[k'M+\ell]x[Mn-(k'M+\ell)]$$

$$= \sum_{\ell=0}^{M-1} \sum_{k=-\infty}^{\infty} h[kM+\ell]x[M(n-k)-\ell] = \sum_{\ell=0}^{M-1} h[Mn+\ell] * x[Mn-\ell]$$

Block diagram representation



Upsampling

Increasing sampling rate by integer factor

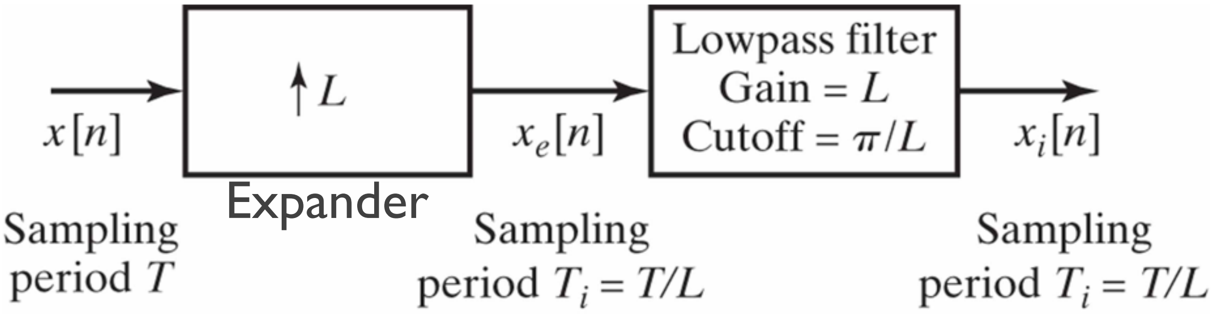
- Usually called "upsampling"
  - Downsampling → analogous to sampling a CT signal
  - Upsampling → analogous to D/C conversion

- Want to increase the sampling rate of x[n] by a factor of *L*

  - Obtain $x_i[n] = x_c(nT_i), \text{ where } T_i = T/L$

    from $x[n] = x_c(nT)$

Upsampling procedure



- It is obvious that $x_i[n] = x[n/L] = x_c(nT/L), \quad n = 0, \pm L, \pm 2L, \ldots$
- The output from the expander is

$$x_e[n] = \begin{cases} x[n/L], & n = 0, \pm L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases}$$

$$= \sum_{k=-\infty}^{\infty} x[k]\delta[n - kL]$$

How does $x_e[n]$ look like?

- The lowpass filter plays a role similar to the ideal D/C converter

Frequency-domain representation

- The Fourier transform of $x_e[n]$ is

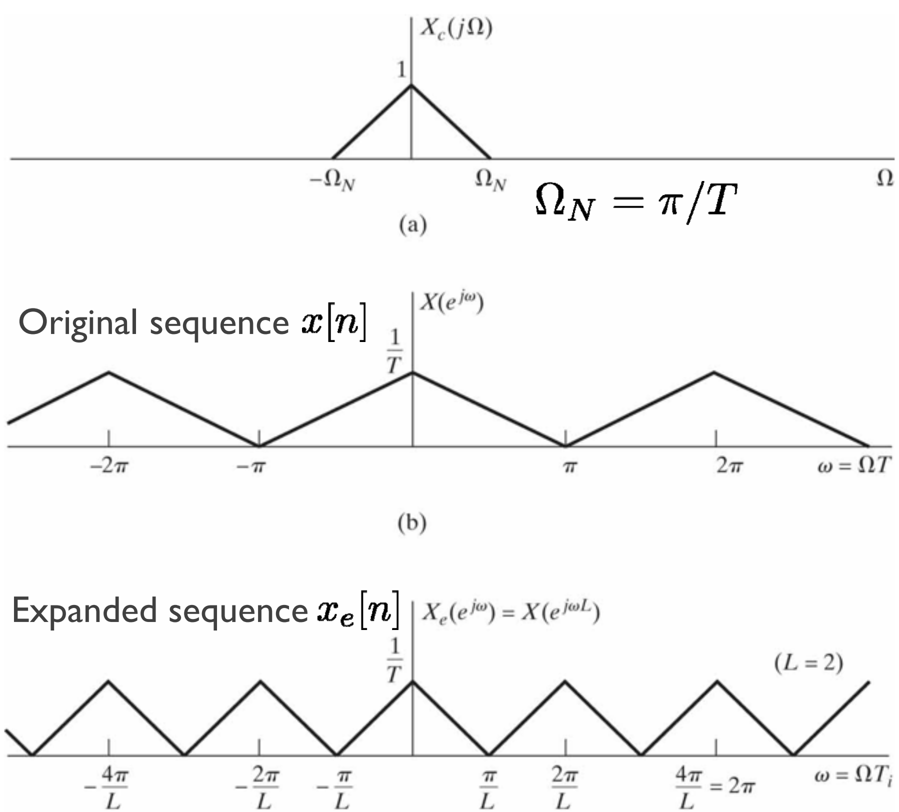
$$X_e(e^{j\omega}) = \sum_{n=-\infty}^{\infty}\left(\sum_{k=-\infty}^{\infty} x[k]\delta[n-kL]\right)e^{-j\omega n}$$

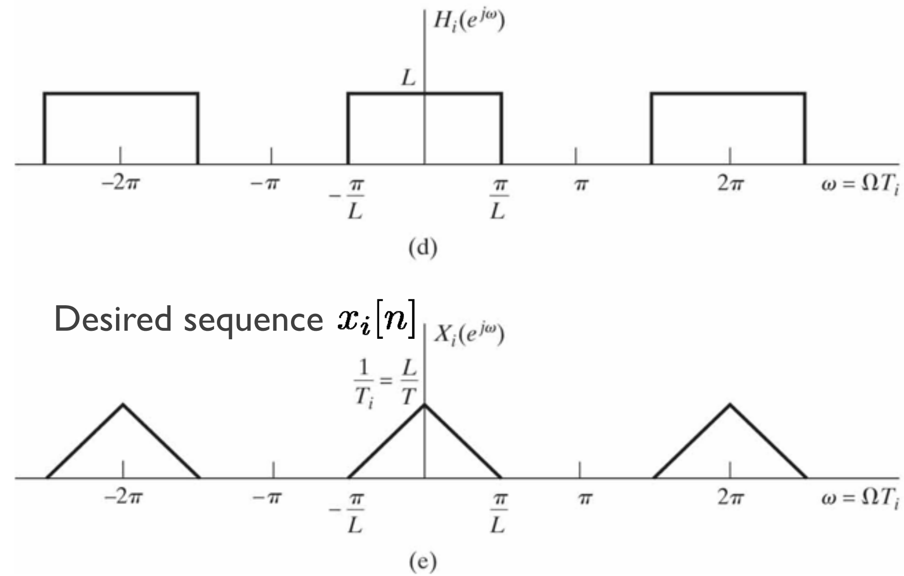
$$= \sum_{k=-\infty}^{\infty} x[k]e^{-j\omega Lk} = X(e^{j\omega L})$$

Frequency-scaled version of x[n]

ω replaced by ωL

Frequency-domain representation





- Lowpass filter removes *L* replicas
- Need to have a gain of *L*

Upsampling = interpolation

- If the input sequence $x[n] = x_c(nT)$ was obtained without aliasing
  ➔ The upsampled sequence $x_i[n]$ can perfectly recover $x_c(t)$

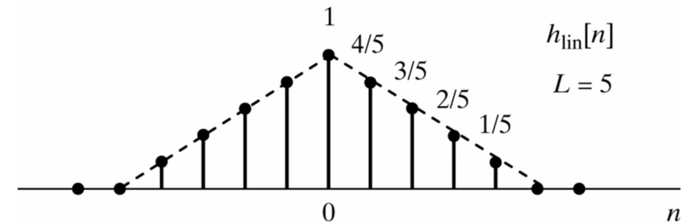
- $x_i[n]$ even has more samples than $x[n]$ in time domain
  ➔ Filling in the missing samples = interpolation!

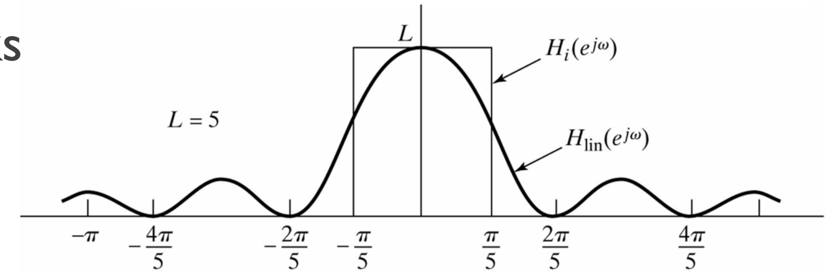
Practical linear interpolation

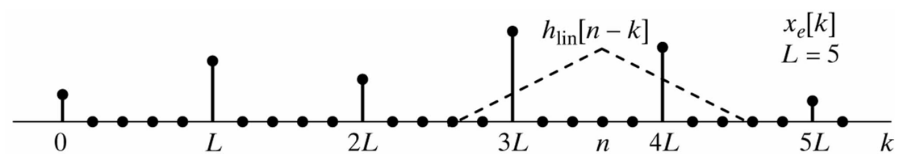
- Ideal lowpass filter is not possible in practice
  - Very good approximations are possible though
- Very simple linear interpolation also works

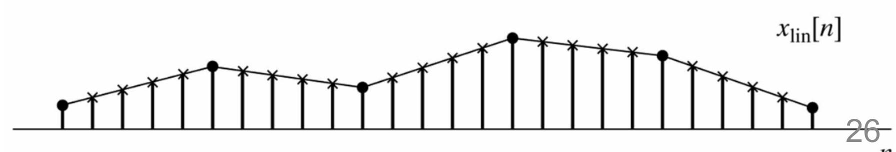
$$h_{\text{lin}}[n] \begin{cases} 1 - |n|/L, & |n| \le L \\ 0, & \text{otherwise} \end{cases}$$

$$H_{\text{lin}}(e^{j\omega}) = \frac{1}{L}\left[\frac{\sin(\omega L/2)}{\sin(\omega/2)}\right]^2$$



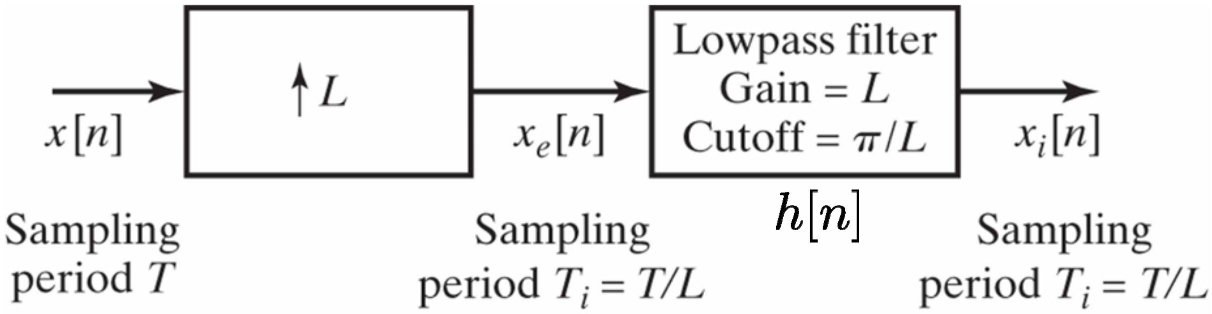






Efficient implementation of upsampling

- Recall



$$x_e[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-kL]$$

$$x_i[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-kL] * h[n]$$

$$= \sum_{k=-\infty}^{\infty} x[k]h[n-kL]$$

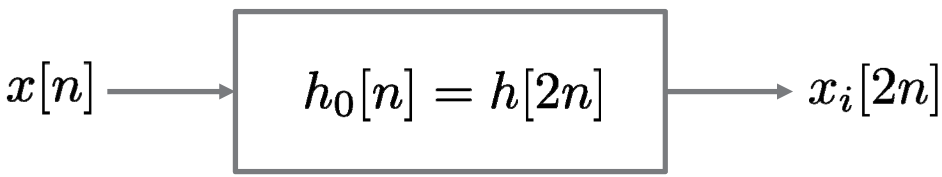
- Assume L=2 for simply illustration

$$x_i[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-2k]$$

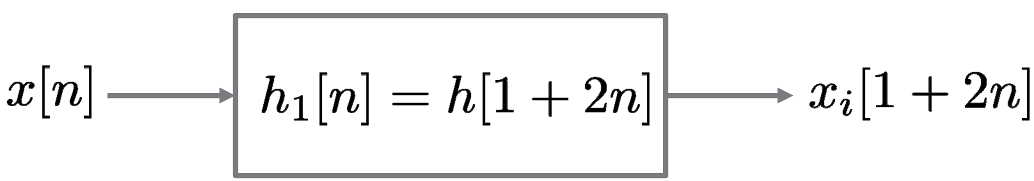
Efficient implementation of upsampling

- Consider $x_i[2n]$ and $x_i[2n+1]$ separately

$$x_i[2n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-2k] = x[n] * h_0[n], \text{ with } h_0[n] = h[2n]$$

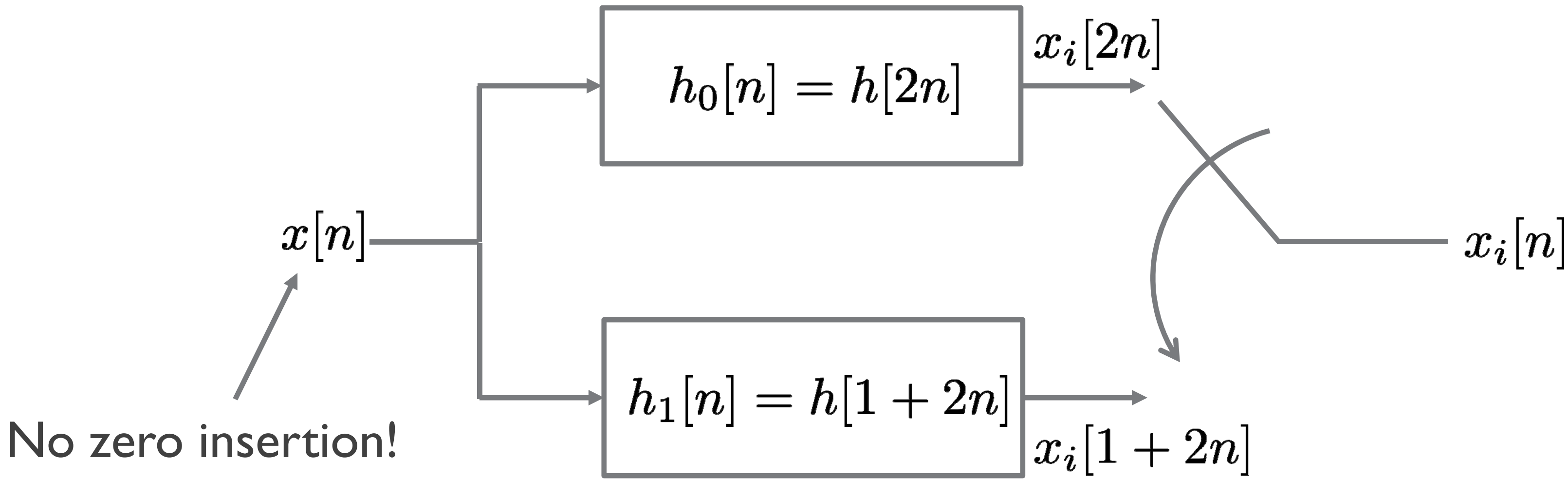


$$x_i[1+2n] = \sum_{k=-\infty}^{\infty} x[k]h[1+2n-2k] = x[n] * h_1[n], \text{ with } h_1[n] = h[1+2n]$$



Efficient implementation of upsampling

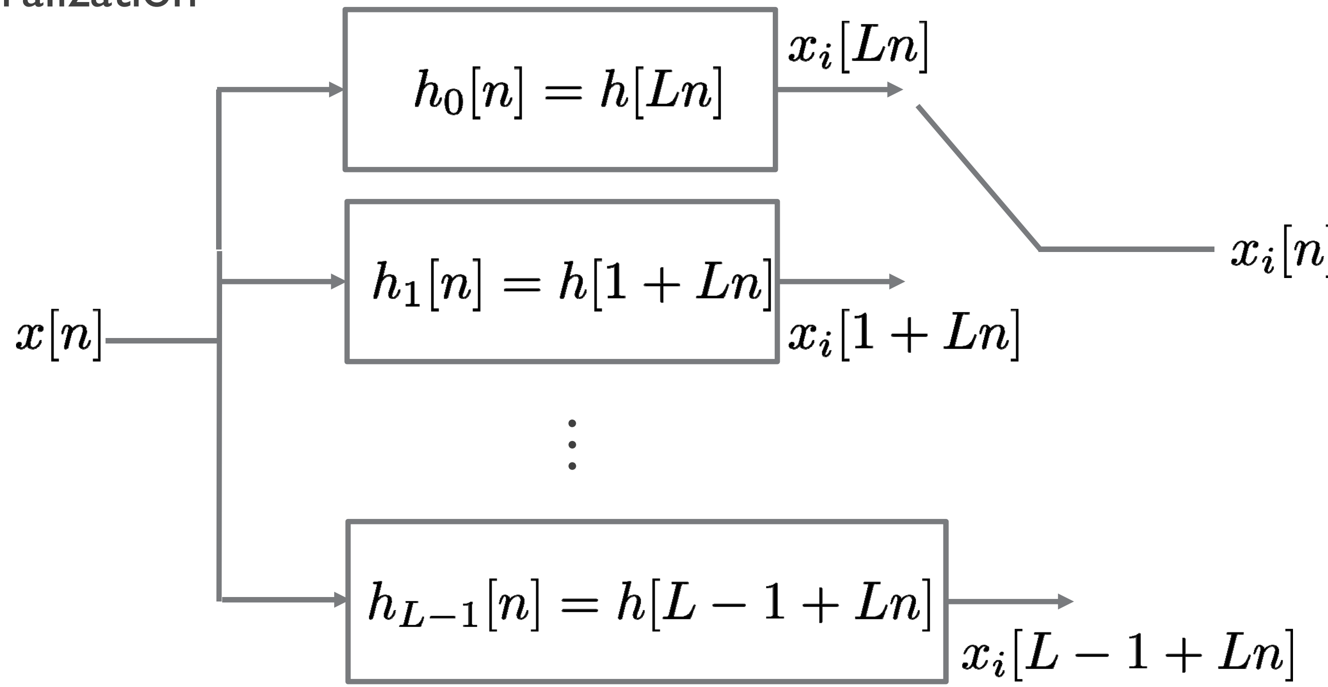
- Overall system



- "Commutator" operates at twice original sampling rate

Efficient implementation of upsampling

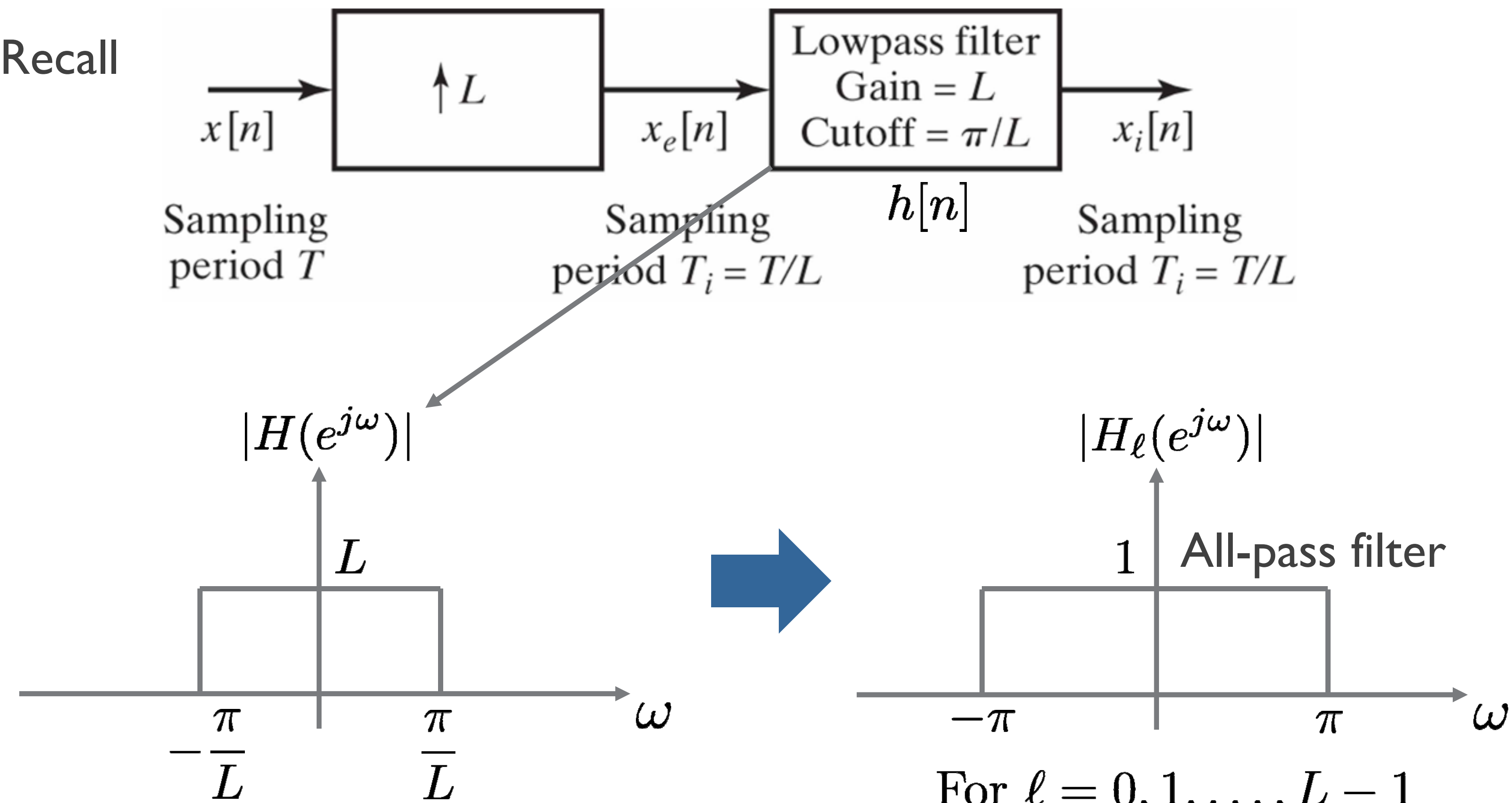
- Generalization



- "Commutator" operates at $L \cdot F_s$

Closer look into the system

- Recall



Frequency-domain representation

- Recall $h_\ell[n] = h[Ln+\ell] \overset{\mathcal{F}}{\longleftrightarrow} H_\ell(e^{j\omega})$?

- First consider $g_\ell[n] = h[n+\ell]$

  ➔ $G_\ell(e^{j\omega}) = e^{j\omega\ell} H(e^{j\omega})$

- Next $h_\ell[n] = g_\ell[Ln] = h[Ln+\ell]$

- Then $H_\ell(e^{j\omega}) = \frac{1}{L}\sum_{k=0}^{L-1} G_\ell\left(e^{j\frac{\omega - k2\pi}{L}}\right)$

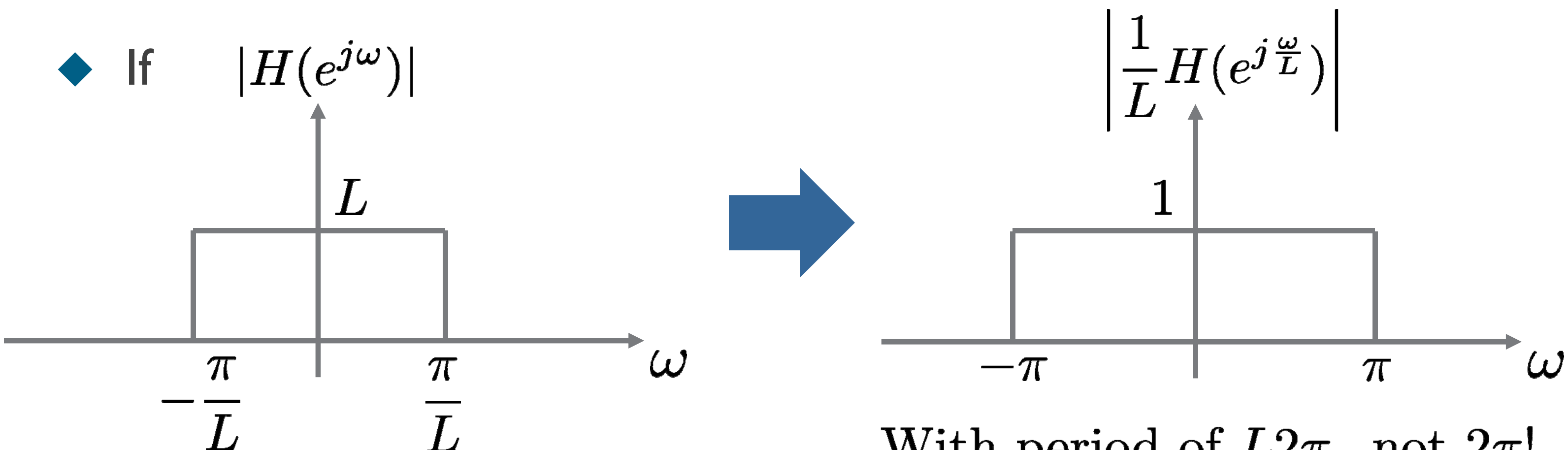
- Substitute $H_\ell(e^{j\omega}) = \frac{1}{L}\sum_{k=0}^{L-1} e^{j\frac{(\omega-k2\pi)\ell}{L}} H\left(e^{j\frac{\omega-k2\pi}{L}}\right)$

Frequency-domain representation

- Recall $H_\ell(e^{j\omega}) = \frac{1}{L}\sum_{k=0}^{L-1} e^{j\frac{(\omega-k2\pi)\ell}{L}} H\left(e^{j\frac{\omega-k2\pi}{L}}\right)$

$$= \left\{\frac{1}{L}\sum_{k=0}^{L-1} e^{-j\frac{k2\pi\ell}{L}} H\left(e^{j\frac{\omega-k2\pi}{L}}\right)\right\} e^{j\frac{\ell}{L}\omega}$$

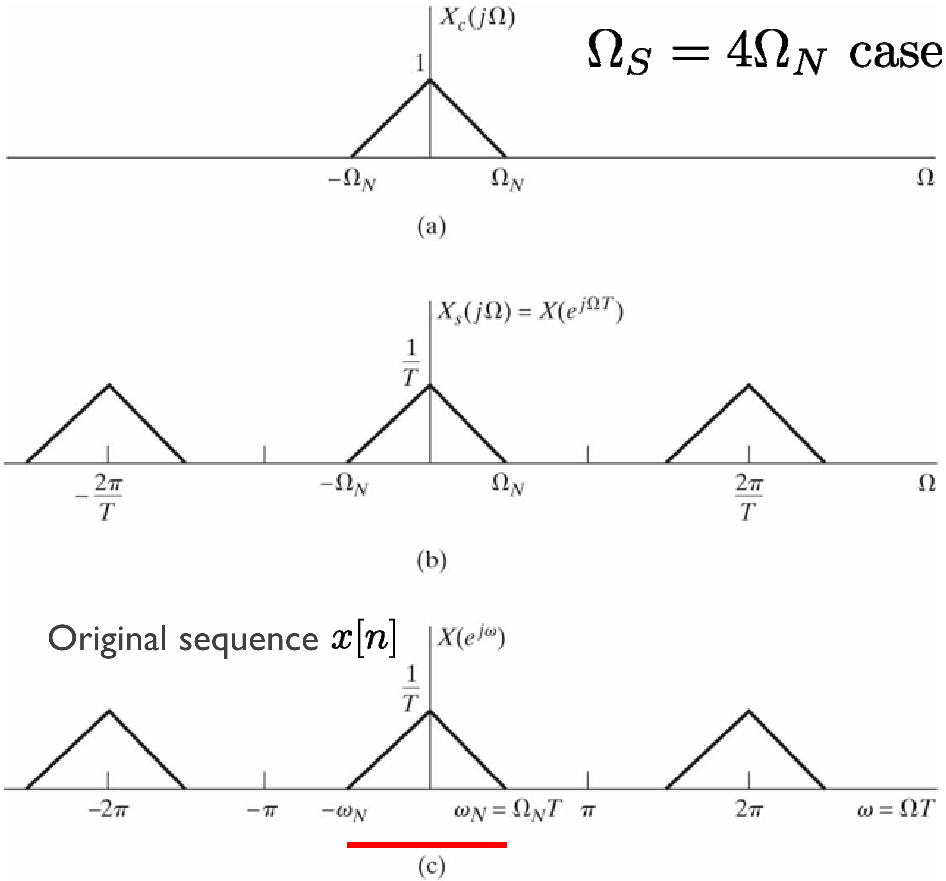
- If

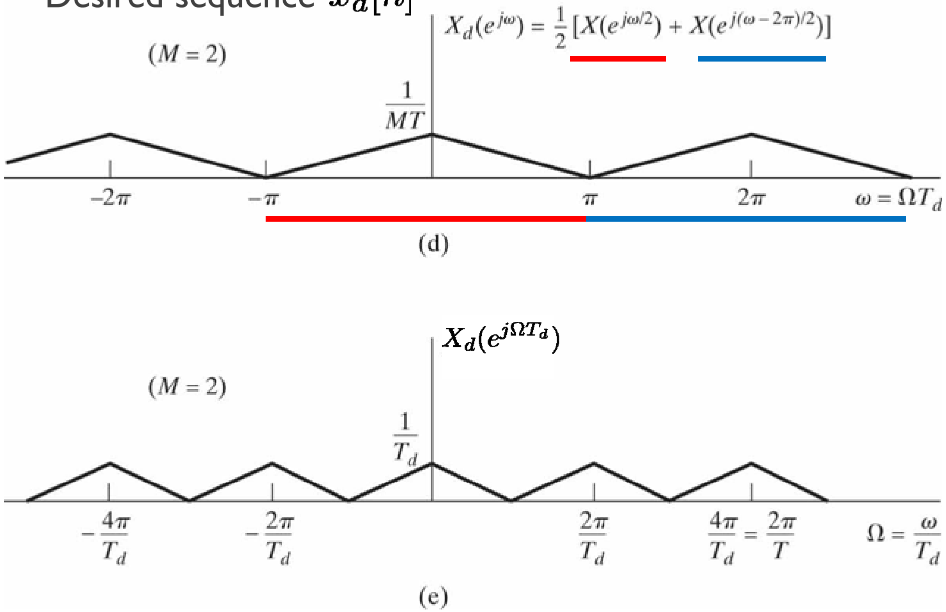


With period of $L2\pi$, not 2π!

Same as downsampling example

$\Omega_S = 4\Omega_N$ case





Frequency-domain representation

- Recall $H_\ell(e^{j\omega}) = \left\{ \frac{1}{L} \sum_{k=0}^{L-1} e^{-j\frac{k2\pi\ell}{L}} H\left(e^{j\frac{\omega-k2\pi}{L}}\right) \right\} e^{j\frac{\ell}{L}\omega}$

Only when k=0 contributes in $-\pi < \omega < \pi$

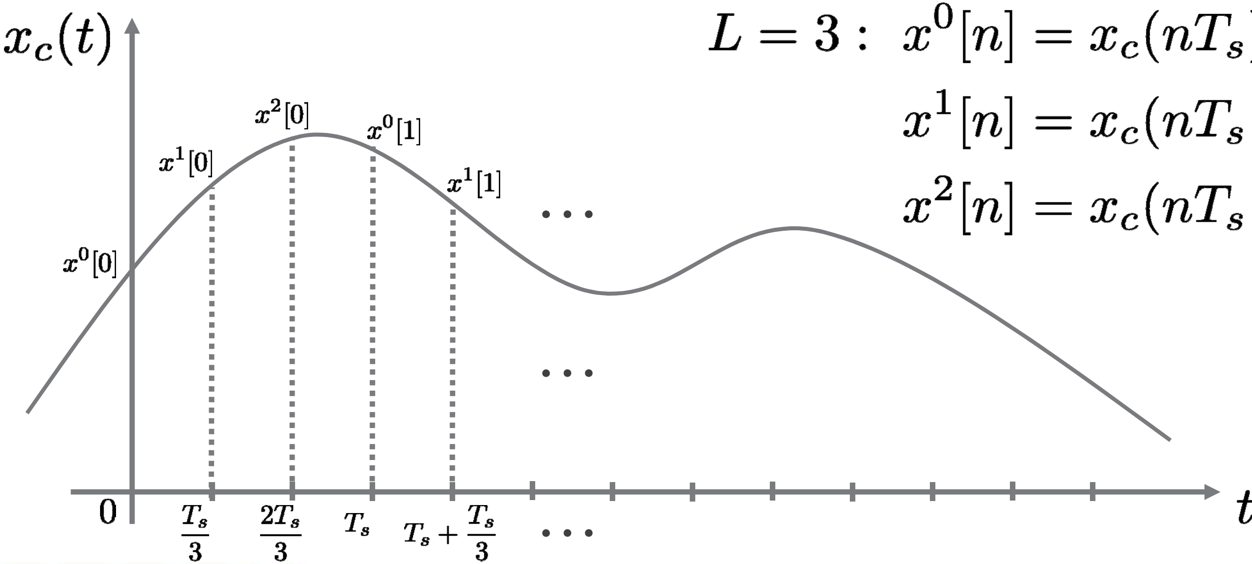
- All Fourier transform must have period of 2π
- Other values of k just serve to make $H_\ell(e^{j\omega})$ be periodic with period 2π
- Thus, with the ideal lowpass filter $H(e^{j\omega})$

$$H_\ell(e^{j\omega}) = e^{j\frac{\ell}{L}\omega}, \quad \text{for } |\omega| < \pi$$

Time-domain illustration

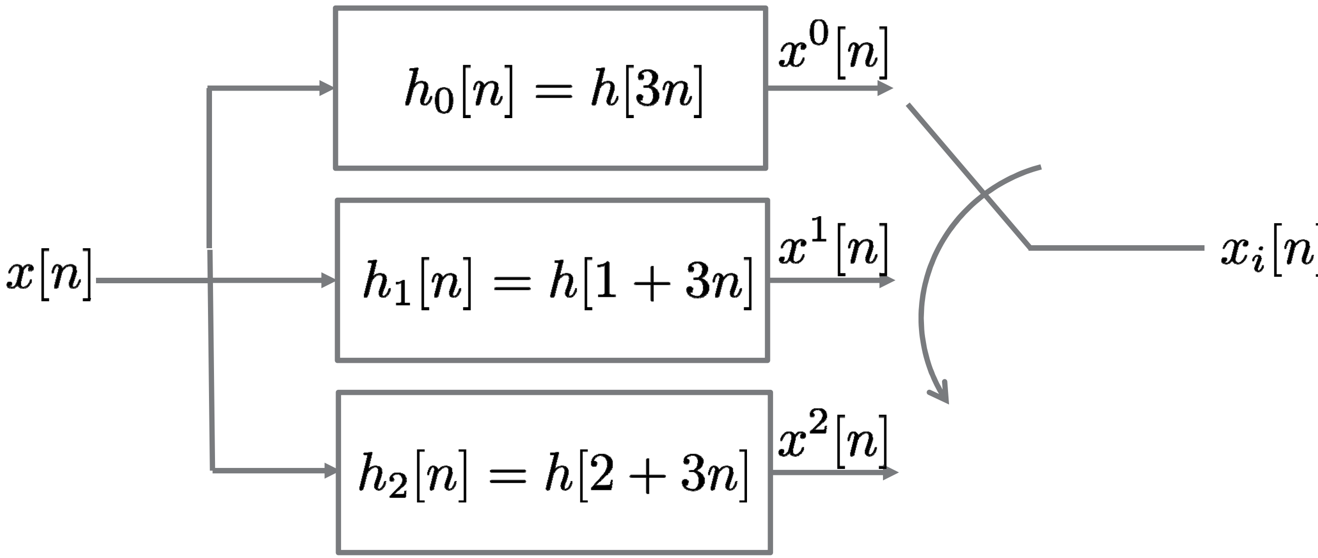
- Note $x[n-n_0] = x_c(nT_s - n_0T_s) \overset{\mathcal{F}}{\longleftrightarrow} e^{-j\omega n_0}X(e^{j\omega})$

- Thus $H_\ell(e^{j\omega}) = e^{j\frac{\ell}{L}\omega}$ translates into a time-shift of $\dfrac{\ell}{L}T_s$ in the time domain
  ➔ A fraction of a sample time-shift



$$L = 3: \quad x^0[n] = x_c(nT_s)$$
$$x^1[n] = x_c(nT_s + T_s/3)$$
$$x^2[n] = x_c(nT_s + 2T_s/3)$$

Efficient implementation of upsampling - revisit



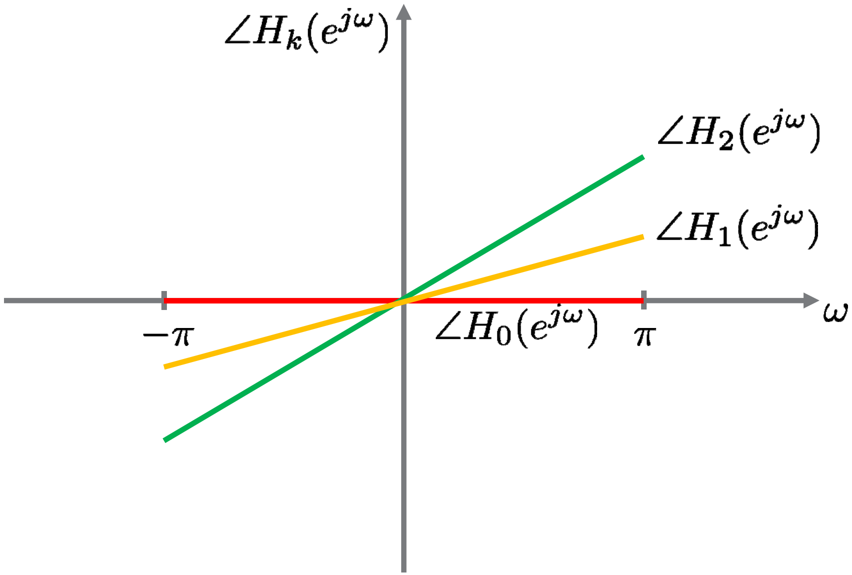
Why polyphase filter?

- For $|\omega| < \pi$,

$$H_0(e^{j\omega}) = 1$$

$$H_1(e^{j\omega}) = e^{j\frac{1}{3}\omega} \rightarrow \angle H_1(e^{j\omega}) = \frac{\omega}{3}$$

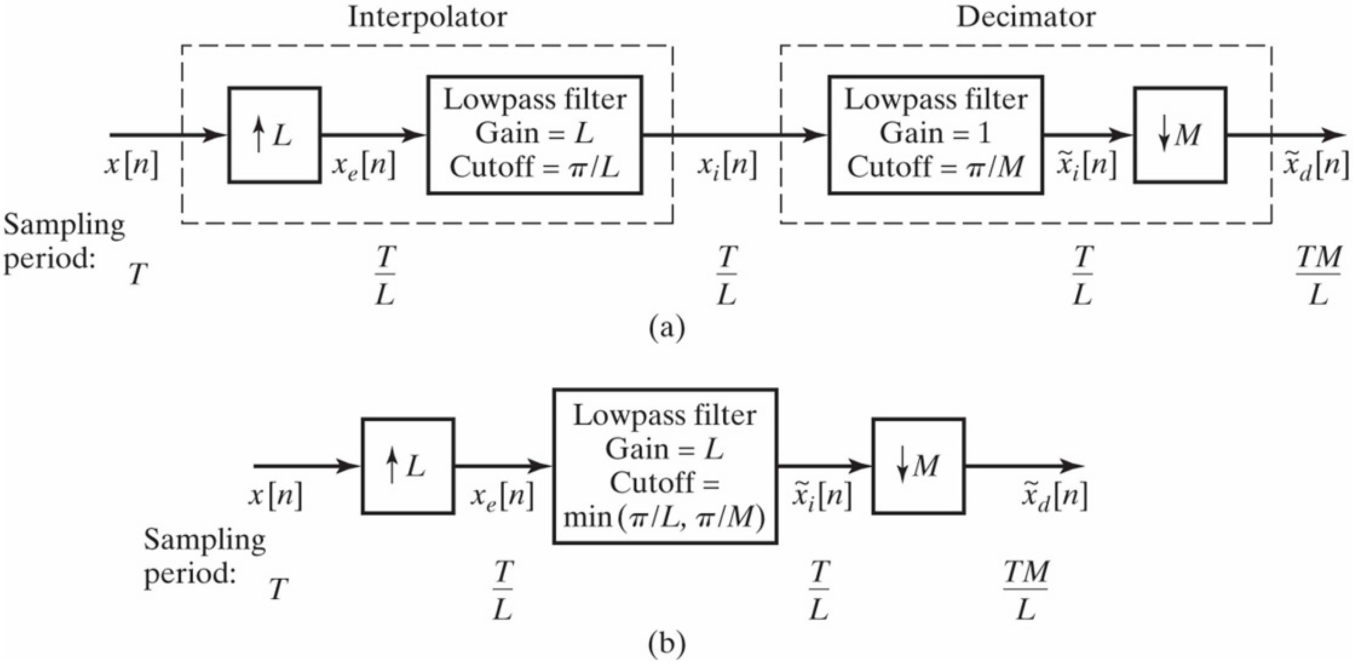
$$H_2(e^{j\omega}) = e^{j\frac{2}{3}\omega} \rightarrow \angle H_2(e^{j\omega}) = \frac{2\omega}{3}$$



Definition of polyphaser decomposition

- The polyphaser decomposition of a sequence is obtained by representing it as a superposition of M subsequences, each consisting of every Mth value of successively delayed versions of the sequence.

Changing sampling rate by a noninteger factor



- Combine interpolator and decimator
  - The order of two systems is important!
- Change sampling rate by a rational factor *L*/*M*
  = Change sampling period by a rational factor *M*/*L*