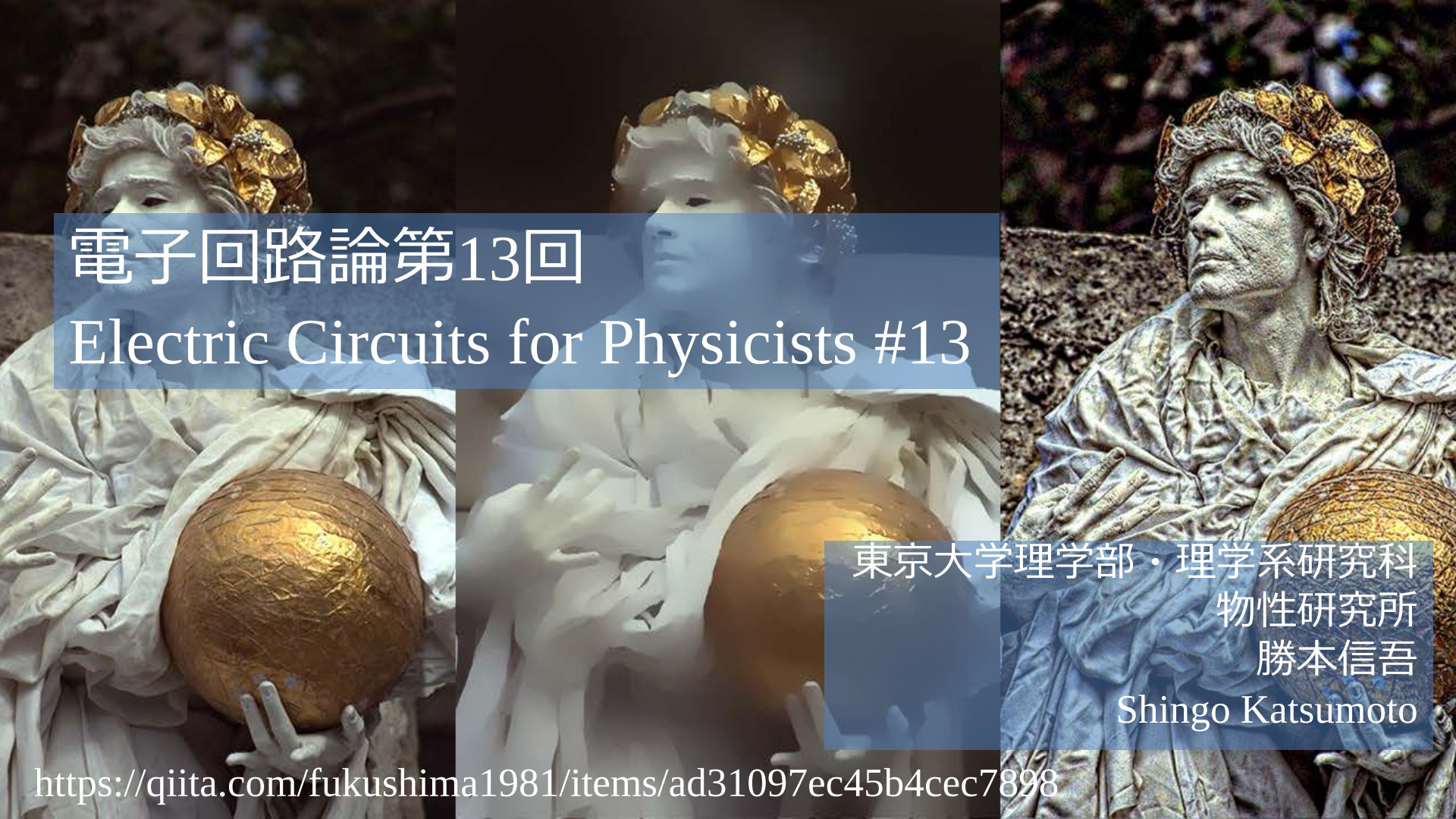


電子回路論第13回
Electric Circuits for Physicists #13

東京大学理学部・理学系研究科
物性研究所
勝本信吾
Shingo Katsumoto

https://qiita.com/fukushima1981/items/ad31097ec45b4cec7898

Outline

6.4 Discrete signal

- 6.4.1 Sampling theorem
- 6.4.2 Pulse amplitude modulation (PAM)
- 6.4.3 Discrete Fourier transform
- 6.4.4 z-transform
- 6.4.5 Transfer function of discrete time signal

Ch.7 Digital signals and circuits

- 7.2 Logic gates
- 7.3 Implementation of logic gates
- 7.4 Circuit implementation and simplification of logic operation

Quein-McCluskey algorithm

(Example) $Y = \bar{A} \cdot \bar{B} \cdot C \cdot D + B \cdot C \cdot D + A \cdot B \cdot \bar{C} + A \cdot \bar{B} \cdot C \cdot D$

$$
\begin{aligned}
Y &= \bar{A} \cdot \bar{B} \cdot C \cdot D + (A + \bar{A}) \cdot B \cdot C \cdot D + A \cdot B \cdot \bar{C} \cdot (D + \bar{D}) + A \cdot \bar{B} \cdot C \cdot D \\
&= \bar{A} \cdot \bar{B} \cdot C \cdot D + A \cdot B \cdot C \cdot D + \bar{A} \cdot B \cdot C \cdot D + A \cdot B \cdot \bar{C} \cdot D \\
&+ A \cdot B \cdot \bar{C} \cdot \bar{D} + A \cdot \bar{B} \cdot C \cdot D
\end{aligned}
$$

Or in binary: Y = 0011+1111+0111+1101+1100+1011

Classification with the number of 1

Compression with $A + \bar{A} = 1$ occurs between different classes with number difference 1.

Num.of 1	smallest	compress1	compress2
2	0011	0_11	__11
	1100	_011	__11
3	0111	110_	
	1011	_111	
	1101	1_11	
4	1111	11_1	

Quein-McCluskey algorithm

Original terms →

Y = __11+110_+11_1

Search for redundant terms.
Put circles if the original term contains the expression.
Then indispensable ones should be marked with double circles.

	smallest					
	0011	1100	0111	1011	1101	1111
__11	○		○	○		○
110_		○			○	
11_1					○	○

	smallest					
	0011	1100	0111	1011	1101	1111
__11	◎		◎	◎		◎
110_		◎			◎	
11_1					○	○

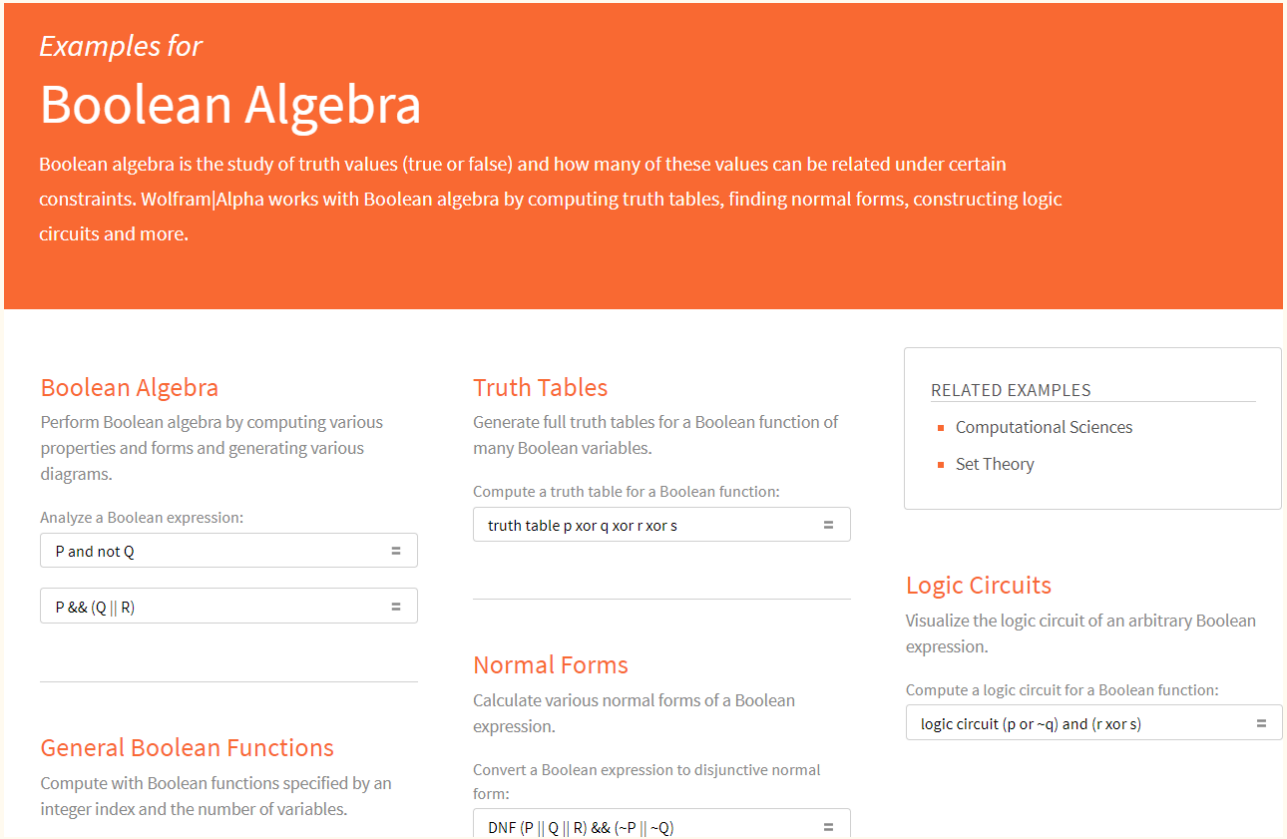
Single circle in one column

Give priority to already indispensable terms.

Final form Y = __11+110_

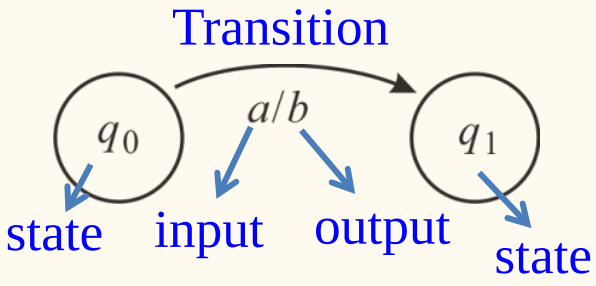
Simplification of logic: Wolfram alpha

https://www.wolframalpha.com/

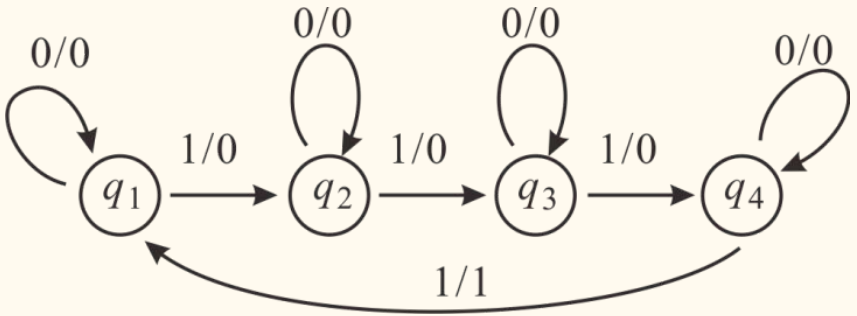


Design of sequential logic circuit: State diagram

State (transition) diagram:



Ex) 2-bit counter with two T-FF



FF output:

$Q_n^{(1)}, Q_n^{(2)}$

Karnaugh map simplification

		input x					
		0			1		
		next		out	next		out
$Q_n^{(1)}$	$Q_n^{(2)}$	$Q_{n+1}^{(1)}$	$Q_{n+1}^{(2)}$	y	$Q_{n+1}^{(1)}$	$Q_{n+1}^{(2)}$	y
0	0	0	0	0	1	0	0
0	1	0	1	0	1	1	0
1	0	1	0	0	0	1	0
1	1	1	1	0	0	0	1

Design of sequential logic: Karnaugh map simplification

$$Q_{n+1}^{(1)} = f(Q_n^{(1)}, Q_n^{(2)}, x)$$
$$Q_{n+1}^{(2)} = g(Q_n^{(1)}, Q_n^{(2)}, x)$$

	$Q_n^{(1)}$	$Q_n^{(2)}$	$Q_{n+1}^{(1)}$ $x=0$	$Q_{n+1}^{(1)}$ $x=1$	$Q_{n+1}^{(2)}$ $x=0$	$Q_{n+1}^{(2)}$ $x=1$
$\overline{Q_n^{(1)}}\,\overline{Q_n^{(2)}}$	0	0	0	1	0	0
$\overline{Q_n^{(1)}}\,Q_n^{(2)}$	0	1	0	1	1	1
$Q_n^{(1)}\,Q_n^{(2)}$	1	1	1	0	1	0
$Q_n^{(1)}\,\overline{Q_n^{(2)}}$	1	0	1	0	0	1

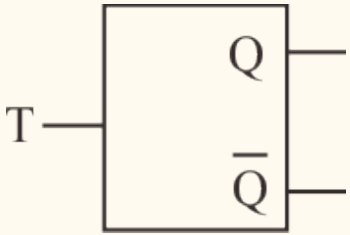
$$\therefore Q_{n+1}^{(1)} = x \cdot \overline{Q_n^{(1)}} + \overline{x} \cdot Q_n^{(1)}$$

similarly $Q_{n+1}^{(2)} = \overline{x} \cdot Q_n^{(2)} + Q_n^{(2)} \cdot \overline{Q}_n^{(1)} + x \cdot \overline{Q}_n^{(2)} \cdot Q_n^{(1)}.$

Design of sequential logic circuit: State diagram

Recursion equation:

$$Q_{n+1}^{(1)} = \overline{x} \cdot Q_n^{(1)} + x \cdot \overline{Q}_n^{(1)},$$
$$Q_{n+1}^{(2)} = \overline{x} \cdot Q_n^{(2)} + Q_n^{(2)} \cdot \overline{Q}_n^{(1)} + x \cdot \overline{Q}_n^{(2)} \cdot Q_n^{(1)}.$$



T	Q	Q
↓	0	0
↓	1	1
↑	0	1
↑	1	0

Characteristic equation (recursion equation)

T-FF: $Q_{n+1} = \overline{T}Q_n + T\overline{Q}_n$

$$Q_{n+1}^{(1)} = \overline{x} \cdot Q_n^{(1)} + x \cdot \overline{Q}_n^{(1)},$$
$$Q_{n+1}^{(2)} = (\overline{x} + \overline{Q}_n^{(1)}) \cdot Q_n^{(2)} + (x \cdot Q_n^{(1)}) \cdot \overline{Q}_n^{(2)}$$
$$= \overline{(x \cdot Q_n^{(1)})} \cdot Q_n^{(2)} + (x \cdot Q_n^{(1)}) \cdot \overline{Q}_n^{(2)}$$

$$y = xQ_n^{(1)}Q_n^{(2)}$$

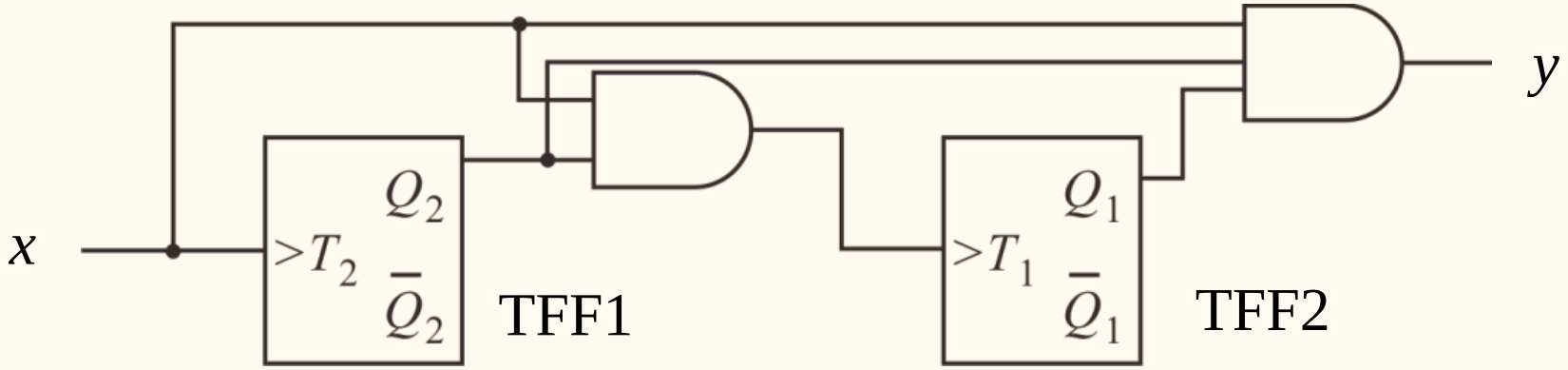
Design of sequential logic circuit: State diagram

$$Q_{n+1}^{(1)} = \overline{x} \cdot Q_n^{(1)} + x \cdot \overline{Q}_n^{(1)},$$

$$Q_{n+1}^{(2)} = (\overline{x} + \overline{Q}_n^{(1)}) \cdot Q_n^{(2)} + (x \cdot Q_n^{(1)}) \cdot \overline{Q}_n^{(2)}$$

$$= \overline{(x \cdot Q_n^{(1)})} \cdot Q_n^{(2)} + (x \cdot Q_n^{(1)}) \cdot \overline{Q}_n^{(2)}$$

$$y = xQ_n^{(1)}Q_n^{(2)}$$



7.5 AD/DA converter circuit

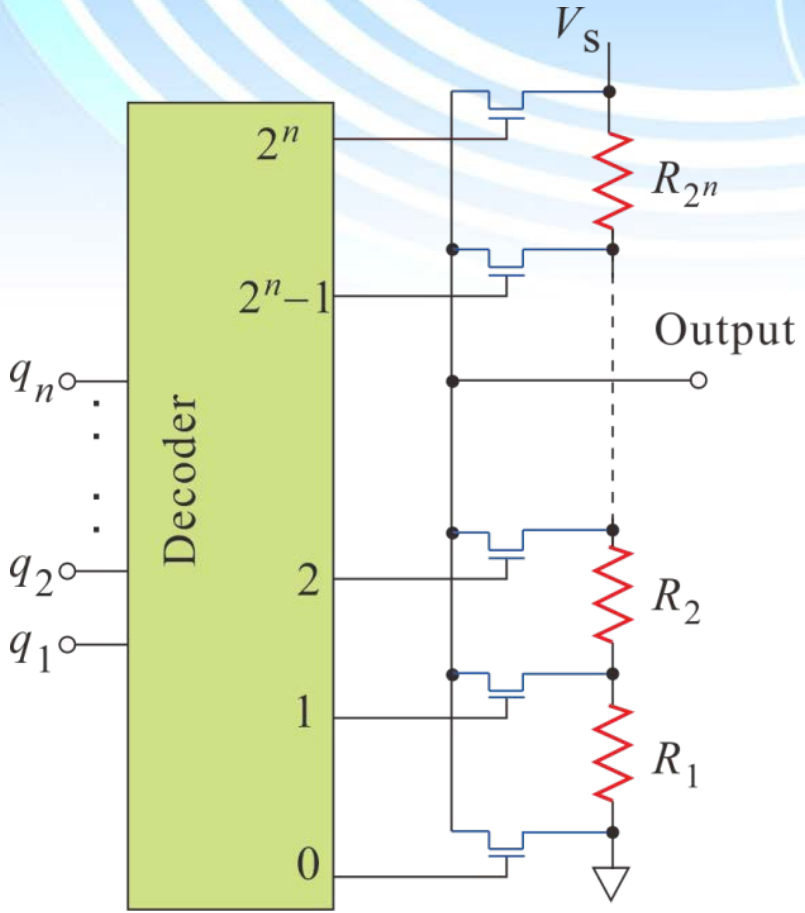
7.5.1 Digital to Analog conversion

Resistor string type DA converter

n bits converter

→ 2^n outputs!, 2^n resistors!

$$V_{\text{out}} = \frac{p_{\text{input}}}{2^n} V_{\text{S}}$$



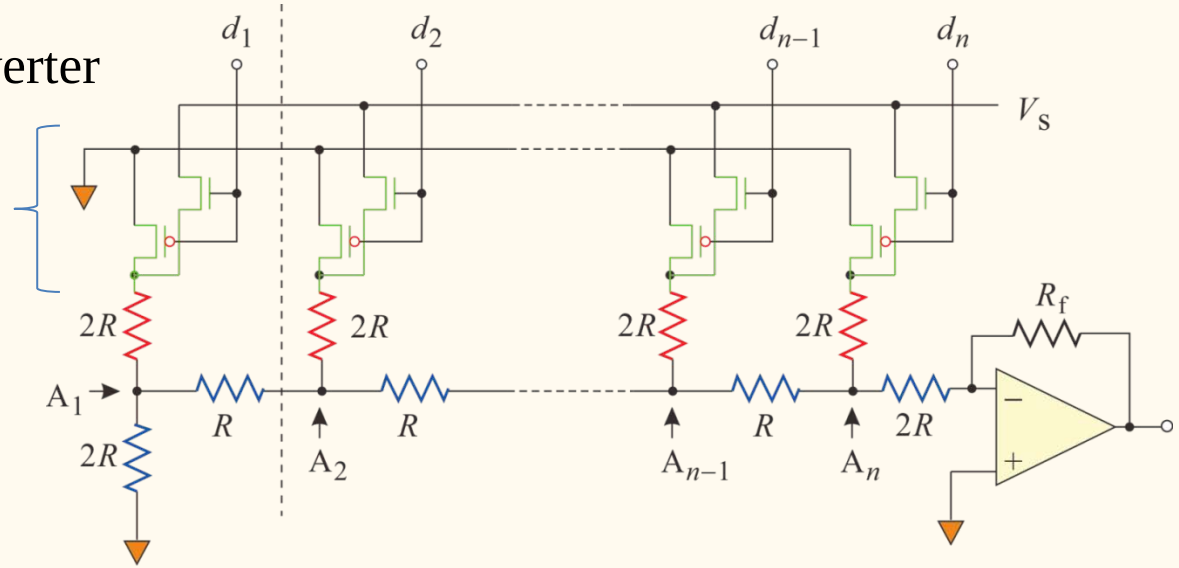
7.5.1 Digital to Analog conversion

Resistor ladder type DA converter

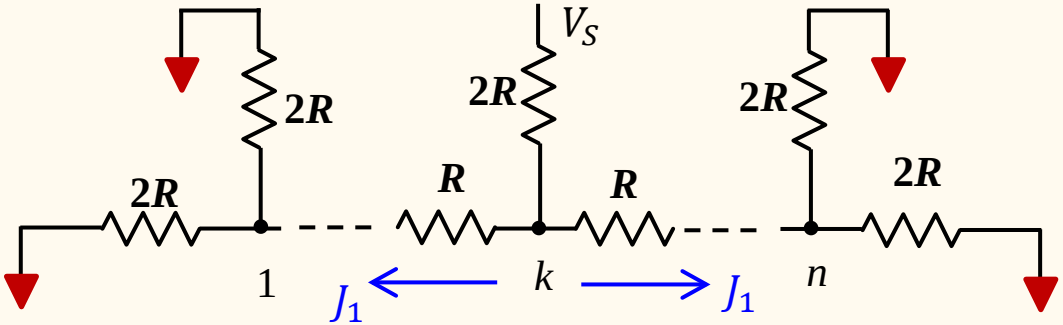
MOS switch array

ON: V_S

OFF: ground

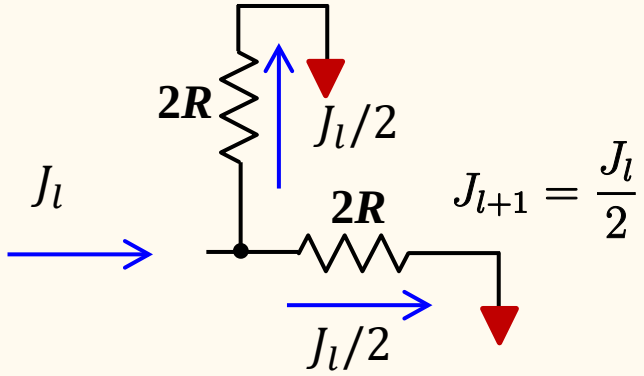


Input $(0,0,\cdots,0,1,0,\cdots,0)$ $d_k = 1$, others = 0



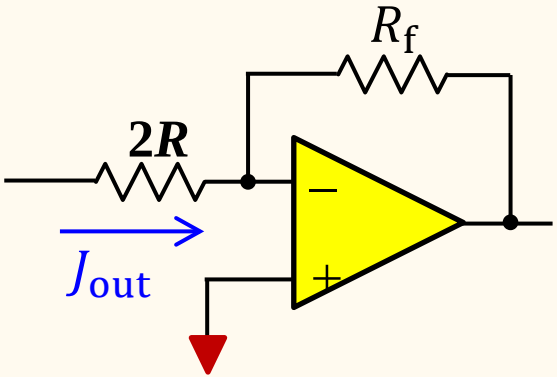
$$J_1 = \frac{V_S}{2 \cdot (2R + R)} = \frac{V_S}{6R}$$

7.5.1 Digital to Analog conversion



$$J_{\text{out}}\underset{n}{(\ 0\cdots0\ }\ \underset{k}{1}\ \underset{1}{0\cdots0\ )} = \frac{V_S}{3R}\left(\frac{1}{2}\right)^{n-k+1}$$

$$= \frac{V_S}{6\cdot 2^n R}2^k$$

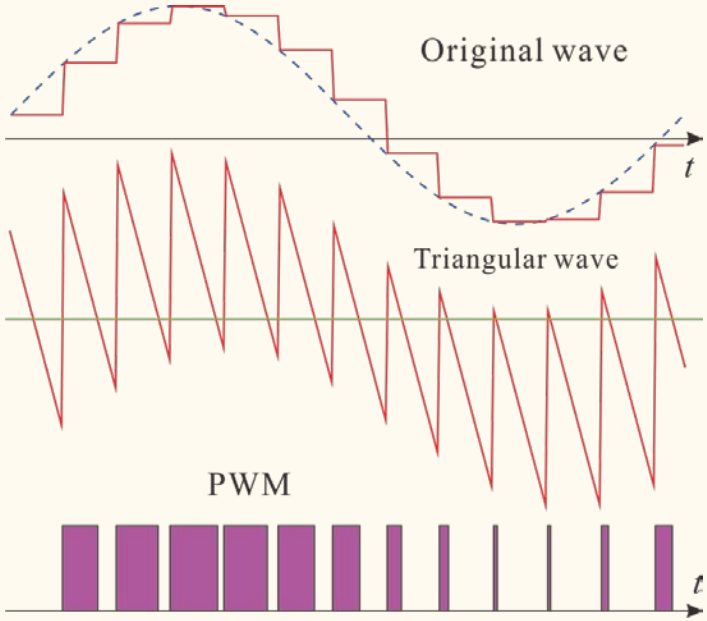


From the superposition theorem:

$$V_{\text{out}}(\{d_i\}) = -\frac{1}{3\cdot 2^n}\frac{R_f}{2R}V_S\sum_{k=1}^{n}2^k d_k$$

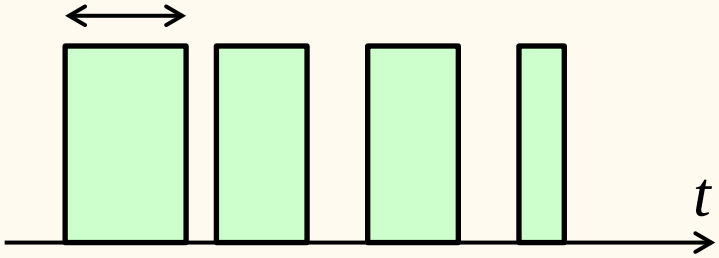
7.5.1 Digital to analog converter

Pulse width modulation (PWM)



Digital signal →

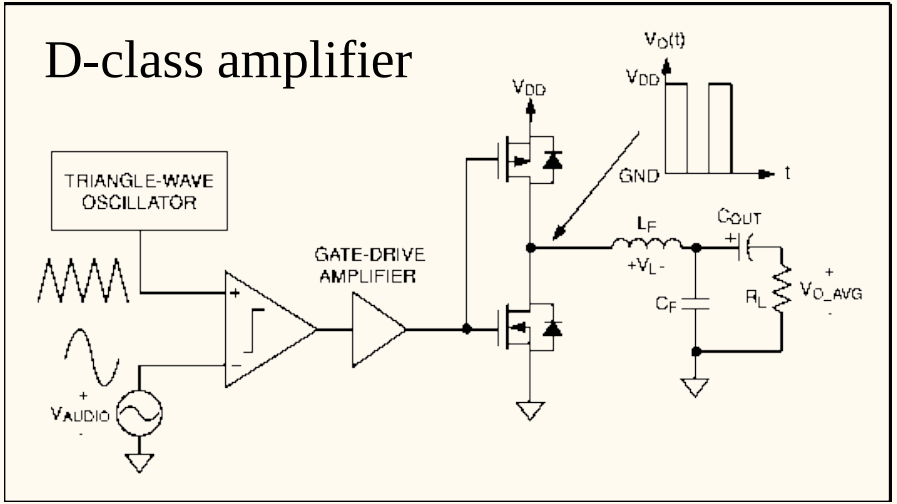
PWM signal with a counter



Low pass filter

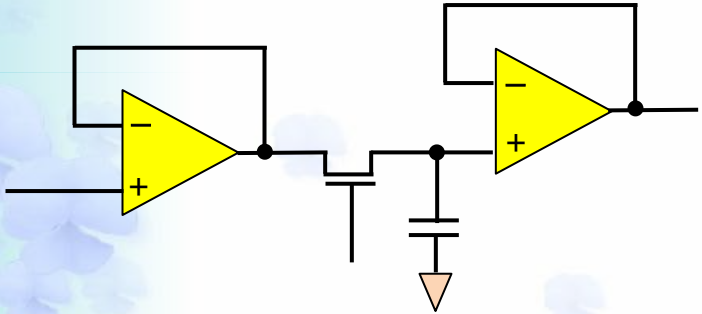
Analog signal

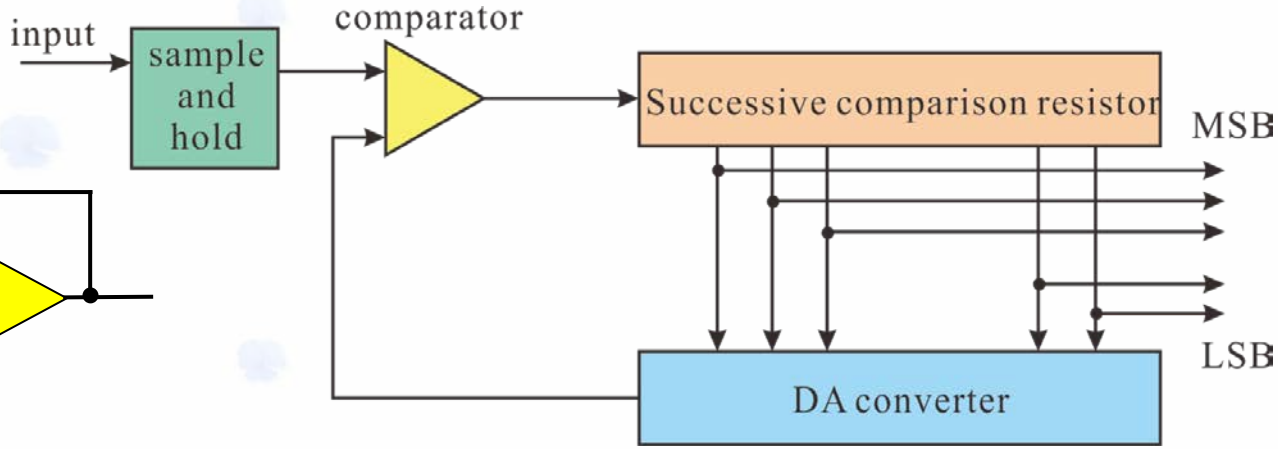
D-class amplifier



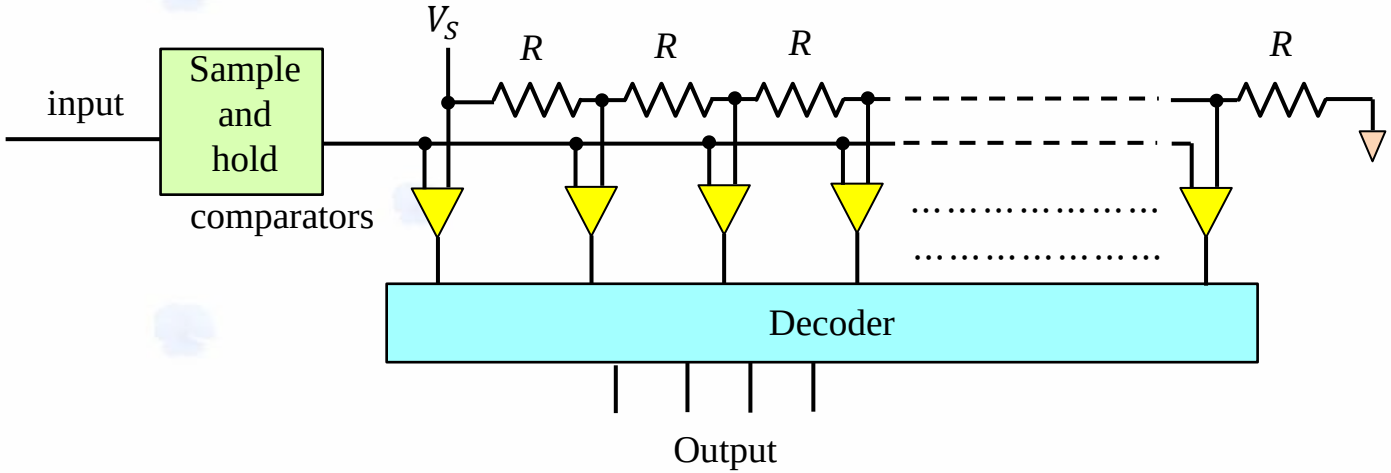
7.5.2 Analog-digital converter

Successive comparison type AD converter



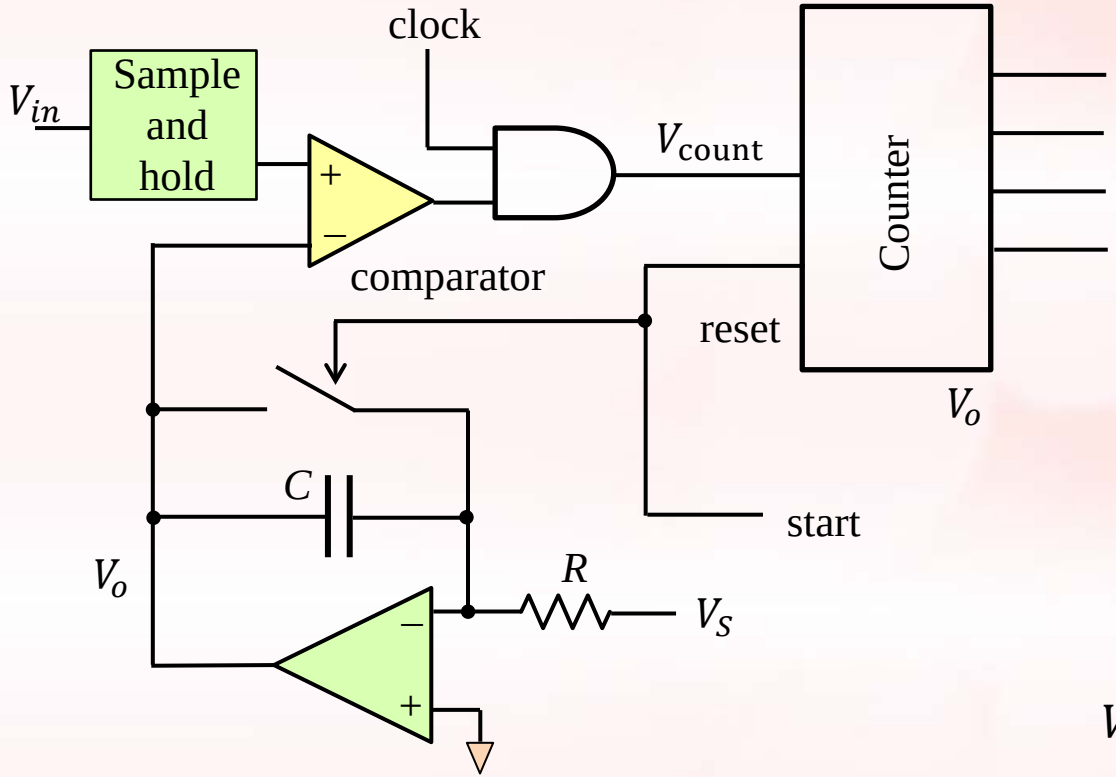


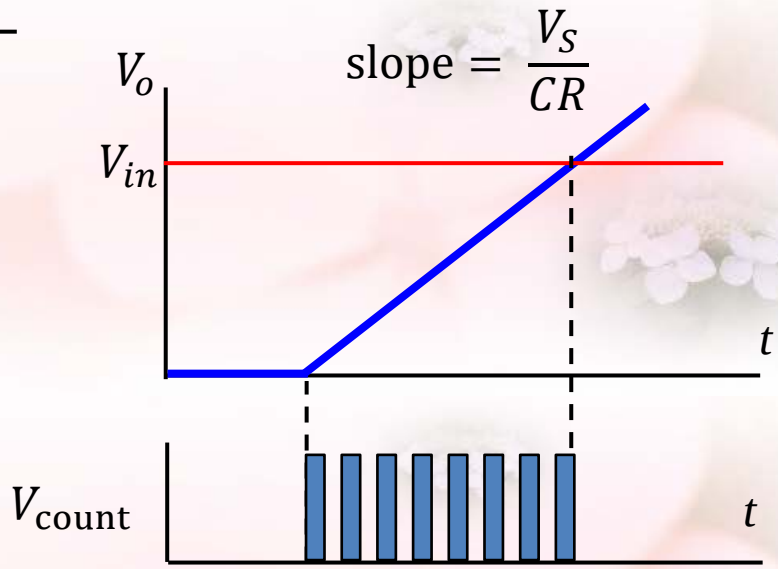
Flush type



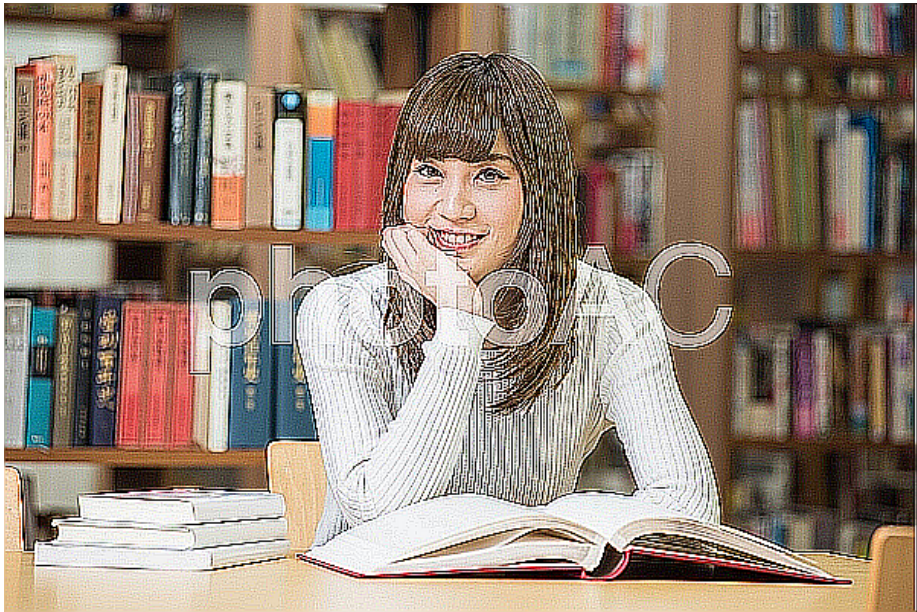
7.5.2 Analog-digital converter

Integrating Analog-Digital converter





7.6 Digital Signal Processing



Edge enhancement

7.6 Digital filter as digital signal processing

Digital filtering (signal processing) is a kind of mapping :

$$\{x_i\} = (x_0, x_1, \cdots)$$

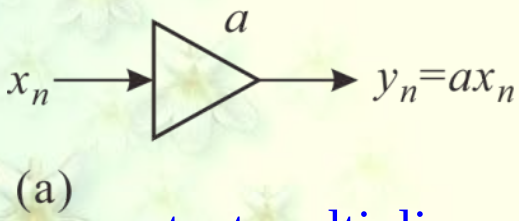
$$\Downarrow$$

$$\{y_i\} = (y_0, y_1, \cdots)$$

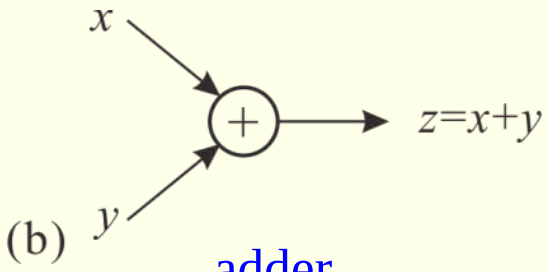
$$y_n = F(x_{n-k}, x_{n-k+1}, \cdots, x_n)$$

We here assume synchronized circuit action with a clock signal.

Block diagram representation of operations

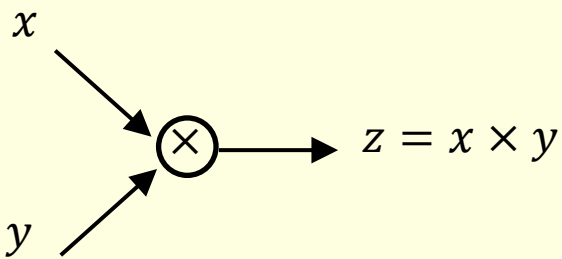


(a) constant multiplier



(b) adder

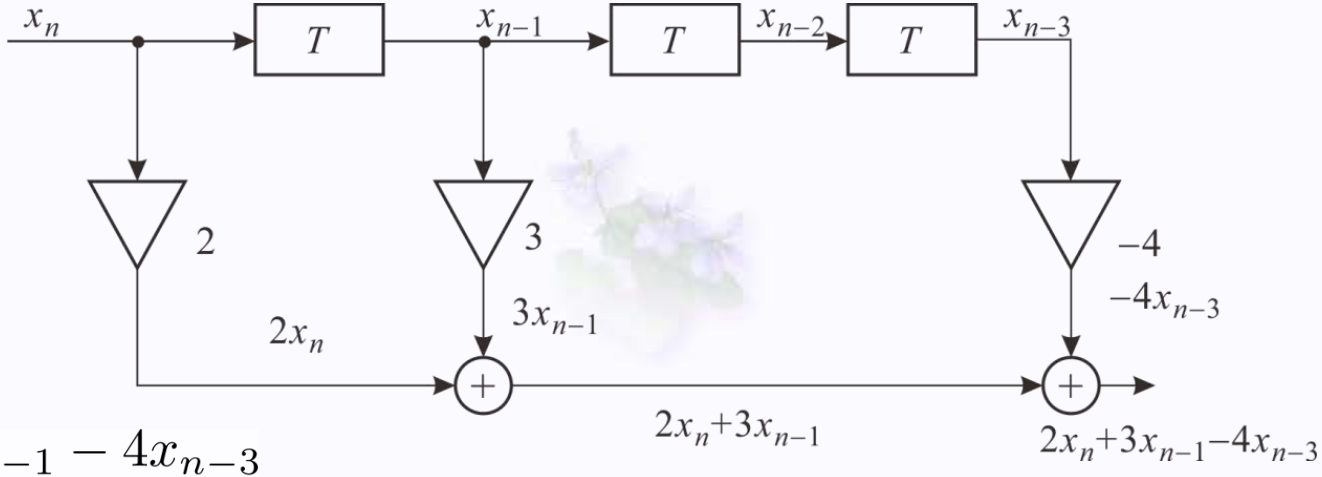
(c) delay (shift register)



(d) multiplier

Block diagram and representation in z-space

Example:



$$y_n = 2x_n + 3x_{n-1} - 4x_{n-3}$$

In z-space, *i.e.*

$$X(z) = \sum_{n=0}^{\infty} x_n z^{-n}, \quad Y(z) = \sum_{n=0}^{\infty} y_n z^{-n}$$

$$Y(z) = 2X(z) + 3z^{-1}X(z) - 4z^{-3}X(z)$$

$$= (2 + 3z^{-1} - 4z^{-3})X(z)$$

$$\therefore H(z)(\text{transfer function}) = 2 + 3z^{-1} - 4z^{-3}$$

How to realize block diagrams

Digital signal processors (DSPs)

- Needs writing program before installation.
- Programmable with high-level languages like C, etc.
- Decimal expression of numbers (floating point, fixed point)
- Numerical processing packages (FFT, etc.)

ex) DSP package (Texas instrument)

Micro-processors

- Needs writing program (high-level languages are possible)
- Comparatively slow action.
- Commands specialized for digital signal processing.

ex) dsPIC (microchip technology)

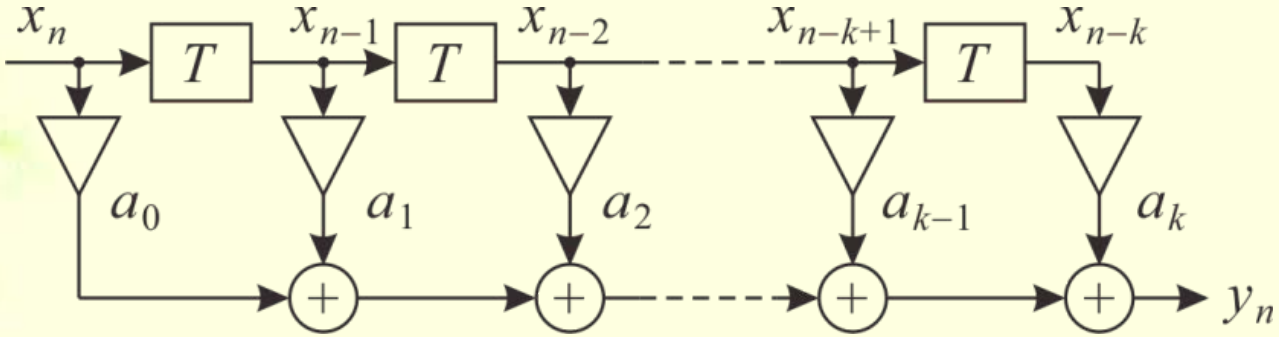
Field programmable gate array (FPGA)

- Fast action.
- Some complication in programming (almost solved)

ex) FPGA (MachX02) (lattice semiconductor)

FIR filter

Finite Impulse Response (FIR) filter



$$y_n = \sum_{j=0}^{k} a_j x_{n-j}$$

$$X(z) = \sum_{n=0}^{\infty} x_n z^{-n}, \quad Y(z) = \sum_{n=0}^{\infty} y_n z^{-n}$$

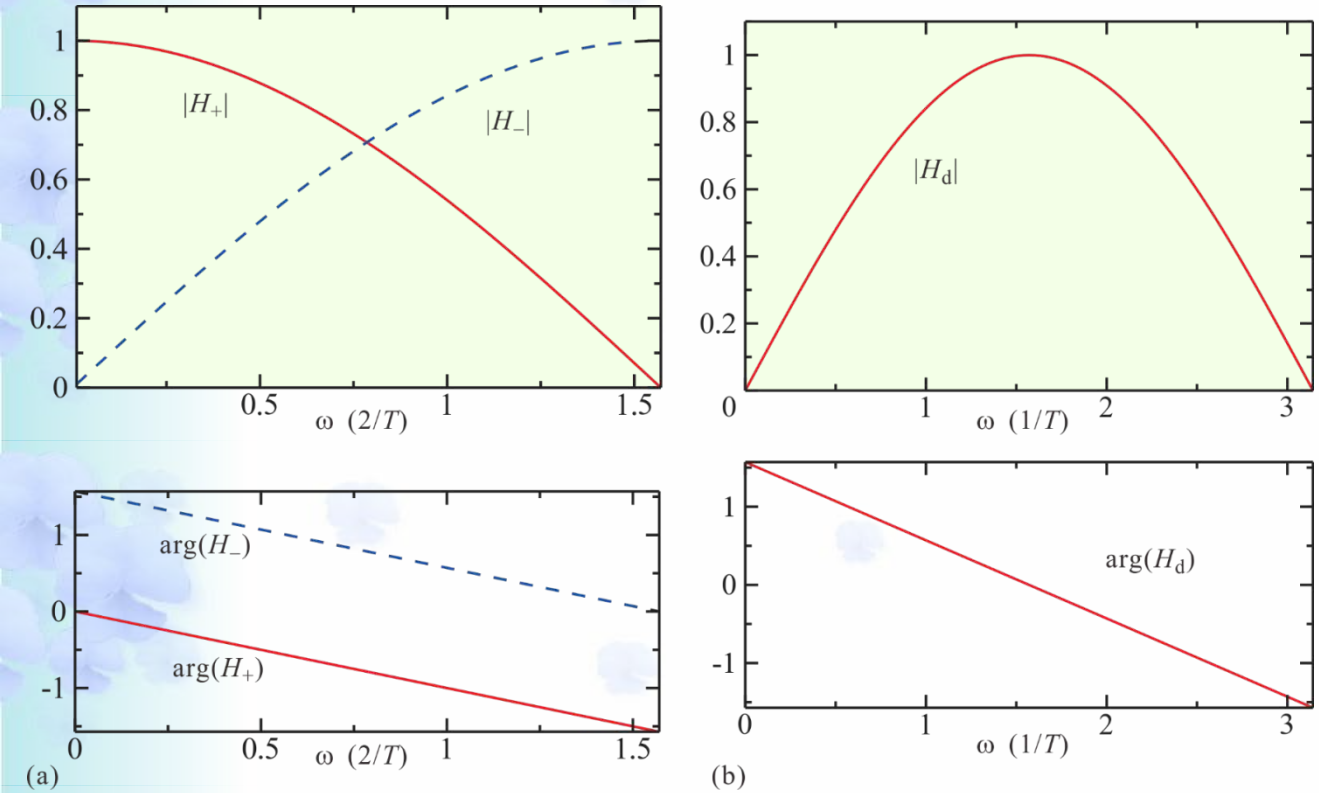
$$H(z) = \sum_{j=0}^{k} a_j z^{-j}$$

Response in frequency domain $z = e^{i\omega\tau}$

$$H(e^{i\omega\tau}) = \sum_{j=0}^{k} a_j e^{-ij\omega\tau}$$

A simple example of FIR filter

$F_{\pm}(x_n, x_{n-1}) = (x_n \pm x_{n-1})/2$ +: moving average, –: differentiation

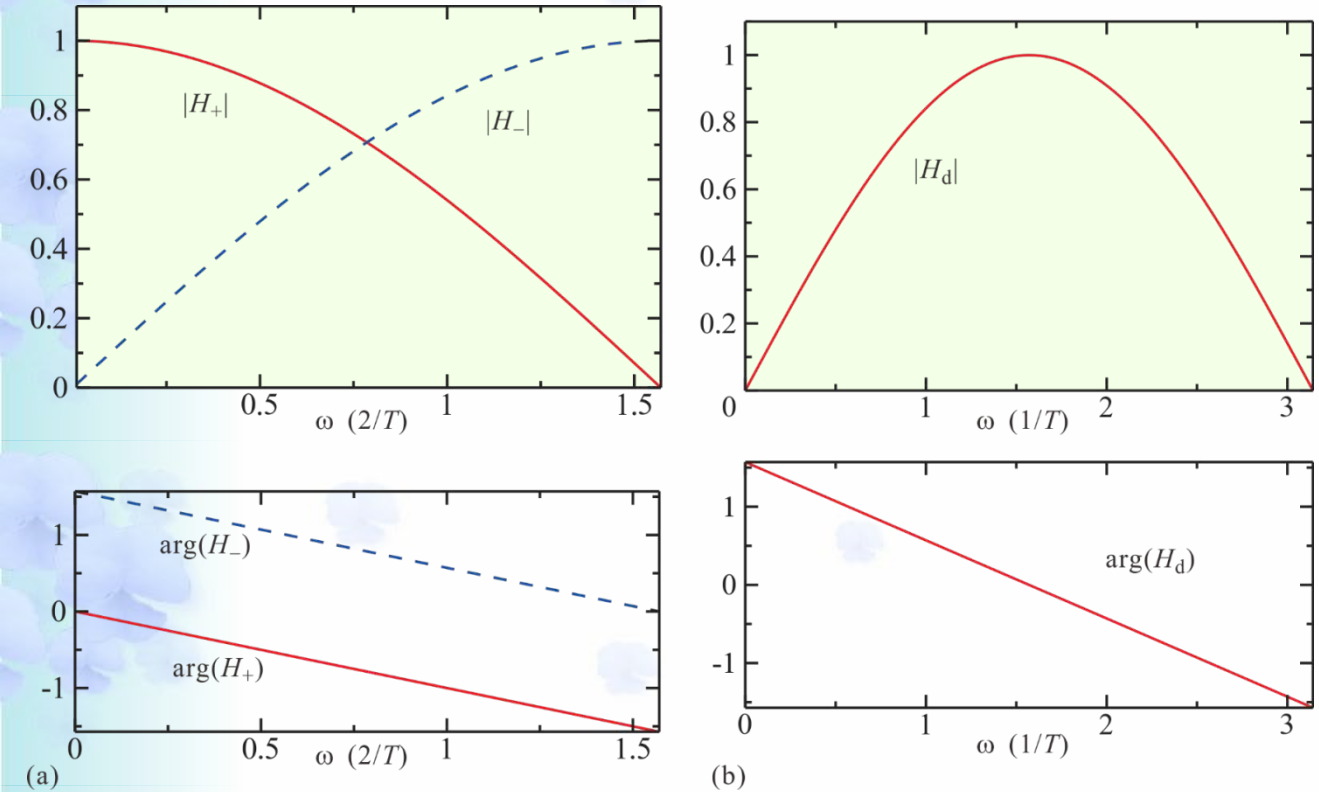


$$H_{\pm}(e^{i\omega\tau}) = e^{-i\omega\tau/2} \begin{pmatrix} \cos(\omega\tau/2) \\ i\sin(\omega\tau/2) \end{pmatrix}$$

A simple example of FIR filter

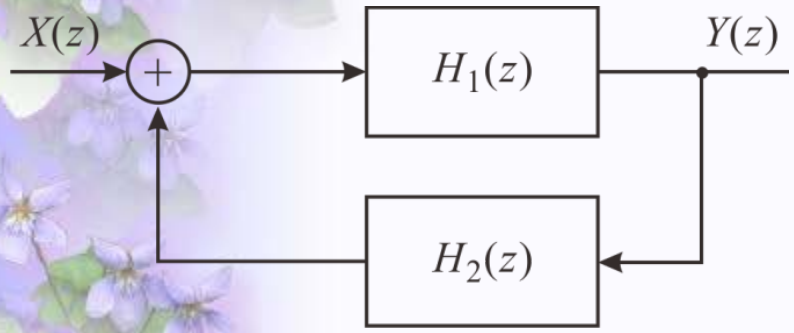
$$F_{\mathrm{d}} = [(x_n + x_{n-1}) - (x_{n-1} + x_{n-2})]/2 = (x_n - x_{n-2})/2$$

: Differentiation of moving average



$$H_{\mathrm{d}}(e^{i\omega\tau})$$
$$= (1 - e^{-2i\omega\tau})/2$$
$$= ie^{-i\omega\tau} \sin \omega\tau$$

Feedback and transfer function



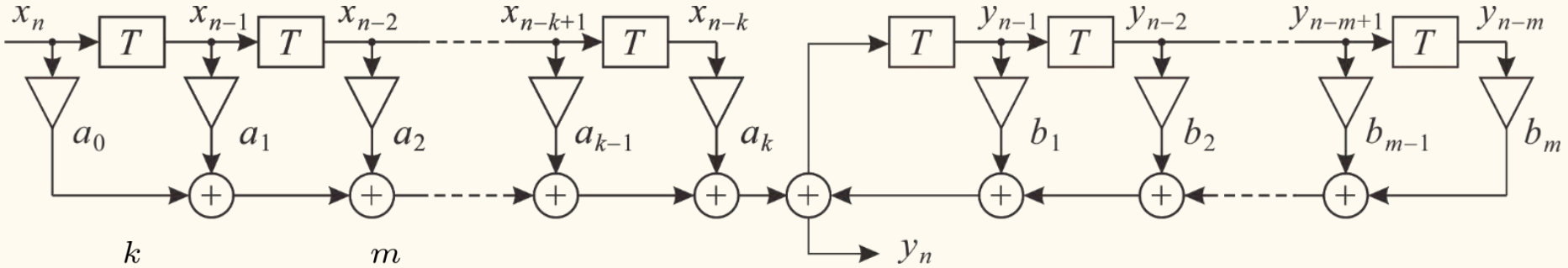
$$Y(z) = H_1(z)W(z) = H_1(z)(X(z) + H_2(z)Y(z)),$$

$$\therefore Y(z) = \frac{H_1(z)}{1 - H_1(z)H_2(z)}X(z)$$

$$H(z) = \frac{H_1(z)}{1 - H_1(z)H_2(z)}$$

$$\text{(transfer function)} = \frac{\text{(direct gain)}}{1 - \text{(feedback transfer gain)}}$$

Infinite Impulse Response (IIR) Filter



$$y_n = \sum_{l=0}^{k} a_l x_{n-l} + \sum_{j=1}^{m} b_j y_{n-j}$$

Stability condition: $\lim_{n\to\infty} y_n = 0$

For impulse response y_n

$$Y(z) = X(z)\sum_{l=0}^{k} a_l z^{-l} + Y(z)\sum_{j=1}^{m} b_j z^{-j}$$

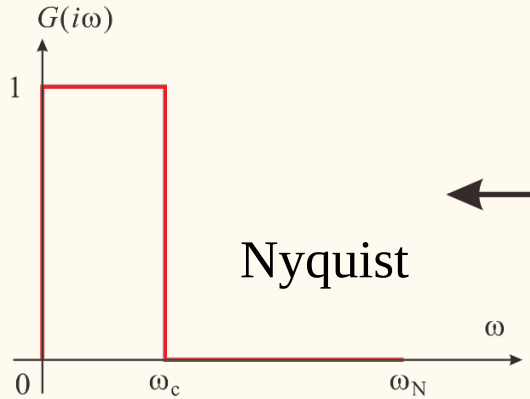
$$H(z) = \frac{Y(z)}{X(z)} = \sum_{l=0}^{k} a_l z^{-l} \Bigg/ \left(1 - \sum_{j=1}^{m} b_j z^{-j}\right)$$

Conversion of z-transform: $|z|>1$ the poles should be in $|z|<1$

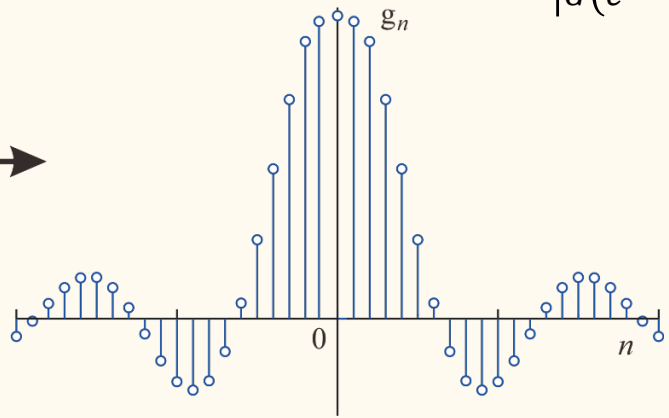
Design of FIR filter: Window function

Ideal low pass filter: $G(e^{i\omega\tau}) = \begin{cases} 1, & |\omega| \le \omega_c, \\ 0, & \omega_c < |\omega| \le \omega_N \end{cases}$ Nyquist frequency

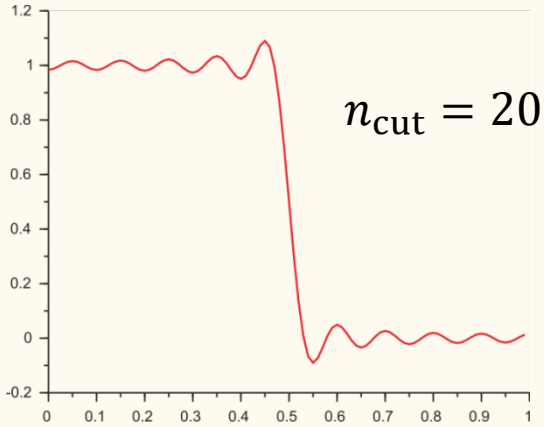
$$G(e^{i\omega\tau}) = \frac{\omega_c}{\omega_N} \sum_{n=-\infty}^{\infty} \frac{1}{n\pi} \mathrm{sinc}\left(n\frac{\omega_c}{\omega_N}\right) e^{-ni\omega\tau} = \gamma_c \sum_{n=-\infty}^{\infty} \frac{1}{n\pi} \mathrm{sinc}(n\gamma_c) z^{-n}$$



Nyquist



$|G(e^{i\omega\tau})|$

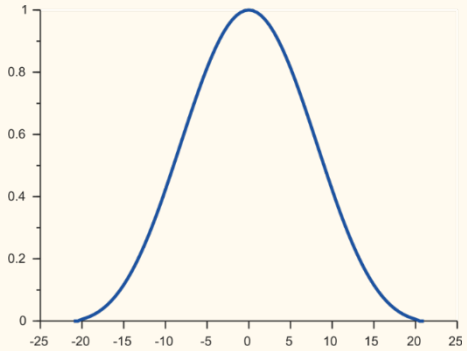


$n_{\mathrm{cut}} = 20$

Cut the series at a finite number

Ripples in frequency characteristics

Design of FIR filter: Window function

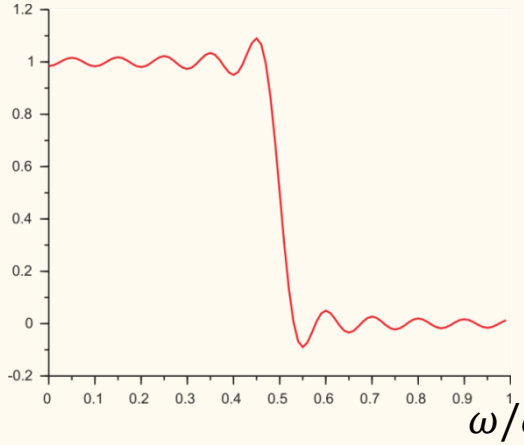


Sudden cutting of z-transform series → Ripples:
Cut with a smooth function

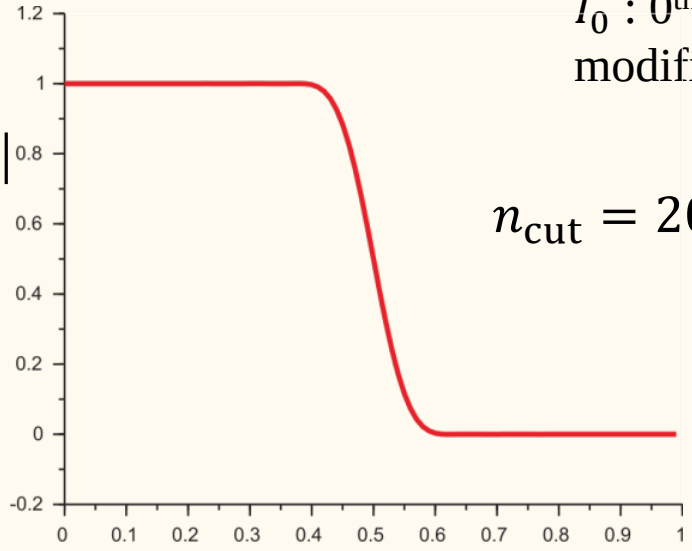
Kaiser window $w_n = \begin{cases} \dfrac{I_0\left(\alpha\sqrt{1-(n/L)^2}\right)}{I_0(\alpha)} & |n| \le L, \\ 0 & |n| > L \end{cases}$

I_0 : 0th order 1st type modified Bessel function

$|G(e^{i\omega\tau})|$



$|G(e^{i\omega\tau})|$



$n_{\text{cut}} = 20$

Design of IIR filter

Transfer function is generally represented by a rational function (有理式).

A way to design IIR filter: modification of analog filter transfer function (s-z transform).

Butterworth filter: $$\Xi(s) = \sum_{k=0}^{N-1} \frac{\omega_k}{s - s_k}, \quad s_k = r_c \exp\left[i \left\{ \frac{\pi}{2} + \frac{(2k+1)\pi}{2N} \right\} \right] \quad (1)$$

$$\xi(t) = u_H(t) \sum_{k=0}^{N-1} w_k \exp(s_k t)$$

Heaviside function

Time discretization with $\tau = 1$:

$$h_n = h_{Hn} \sum_{k=0}^{N-1} w_k e^{n s_k},$$

$$\therefore H(z) = \sum_{k=0}^{N-1} \frac{w_k}{1 - \exp(s_k) z^{-1}}$$

This form is obtained by replacement of $(s - s_k)^{-1} \to (1 - \exp(s_k) z^{-1})^{-1}$

in eq.(1). This is called impulse invariant method.

Design of IIR filter (Bilinear z-transform method)

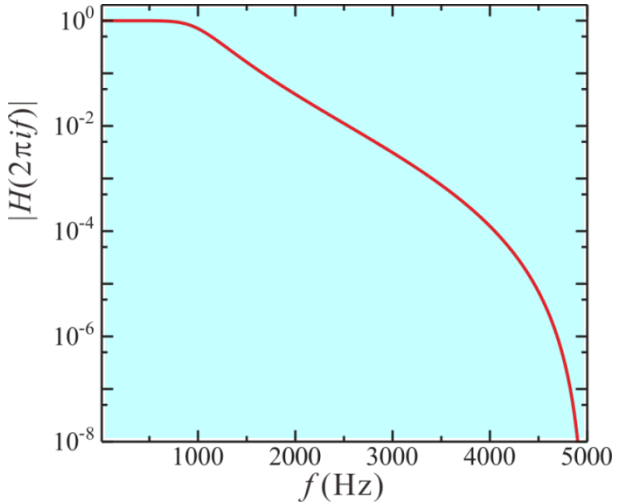
Bilinear z-transform (双一次z変換法) :

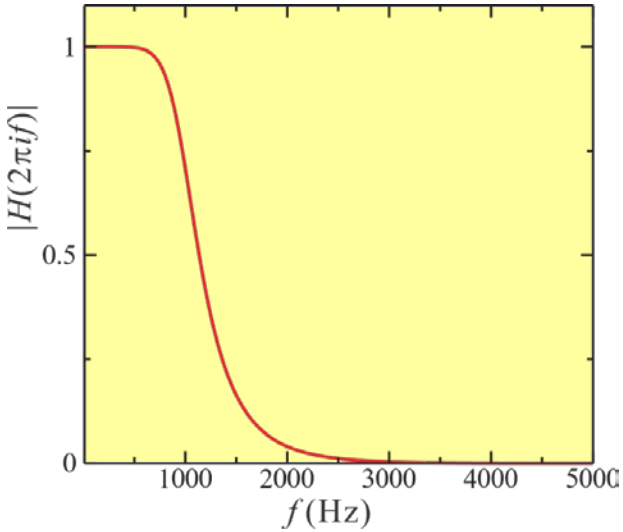
$$s \to \frac{1 - z^{-1}}{1 + z^{-1}}$$

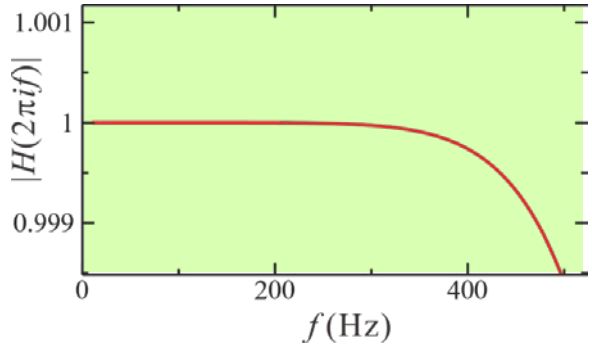
$$i\Omega = \frac{1 - e^{-i\omega}}{1 + e^{i\omega}} \qquad \Omega = \tan\frac{\omega}{2}$$

example) 4th Butterworth:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + b_4 z^{-4}}{1 - a_1 z^{-1} - a_2 z^{-2} - a_3 z^{-3} - a_4 z^{-4}}$$

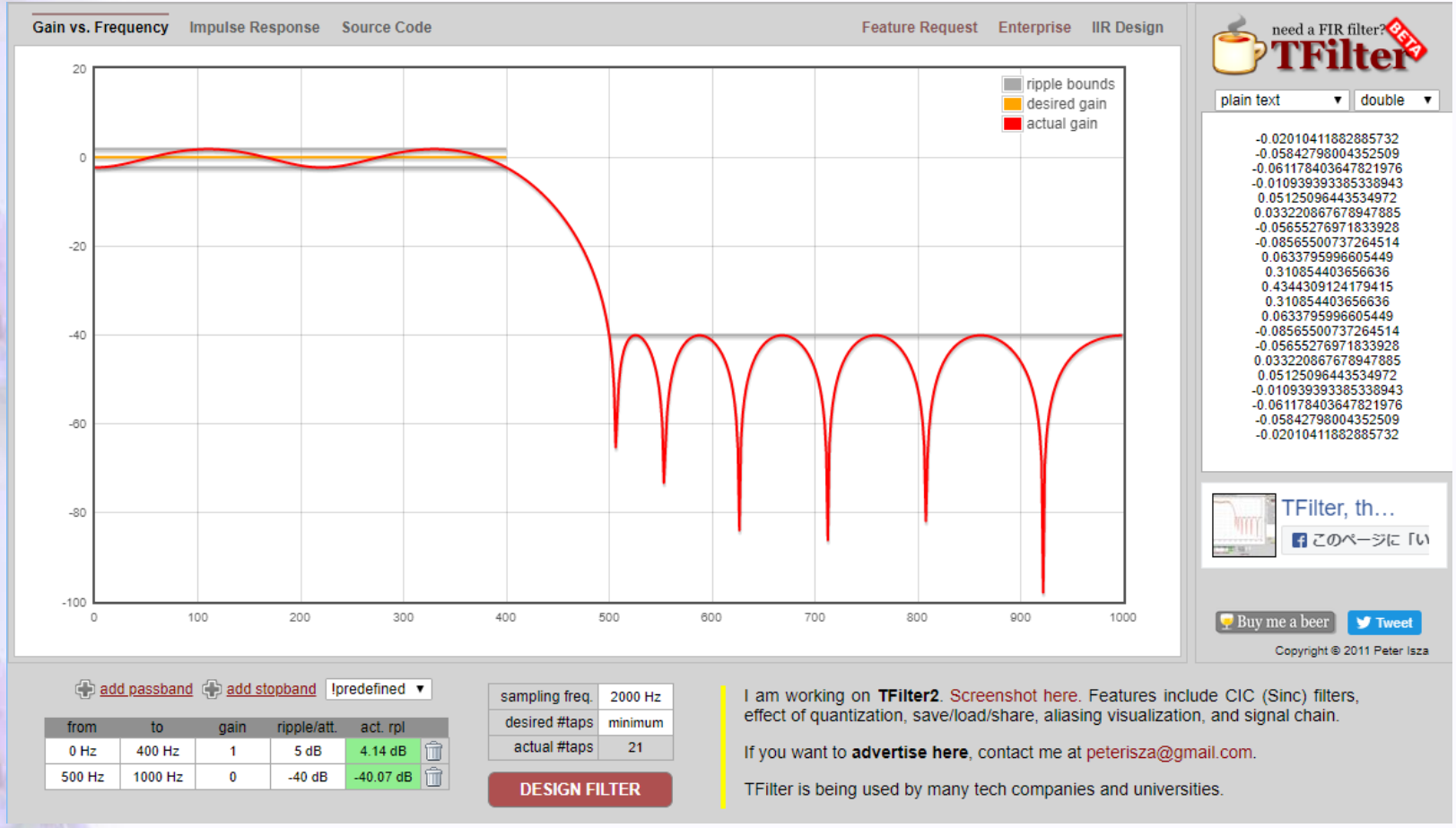




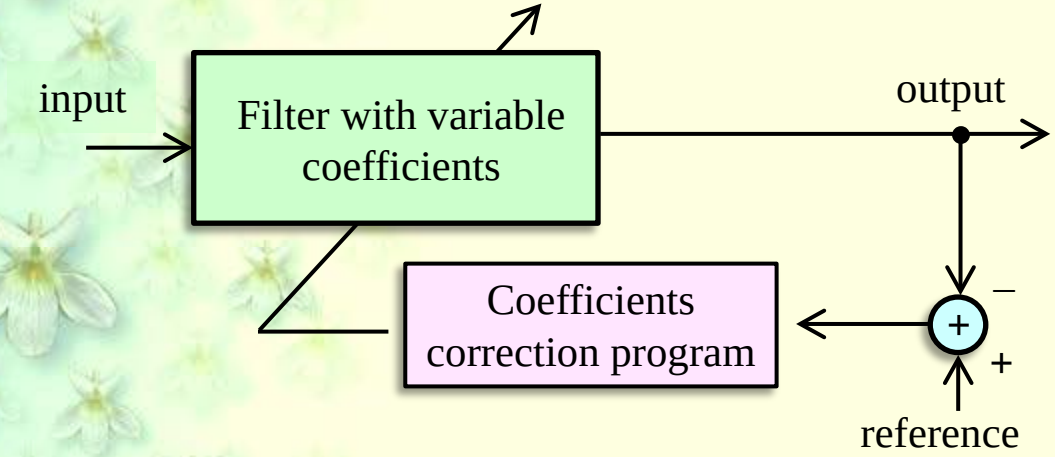


Digital filter design web application

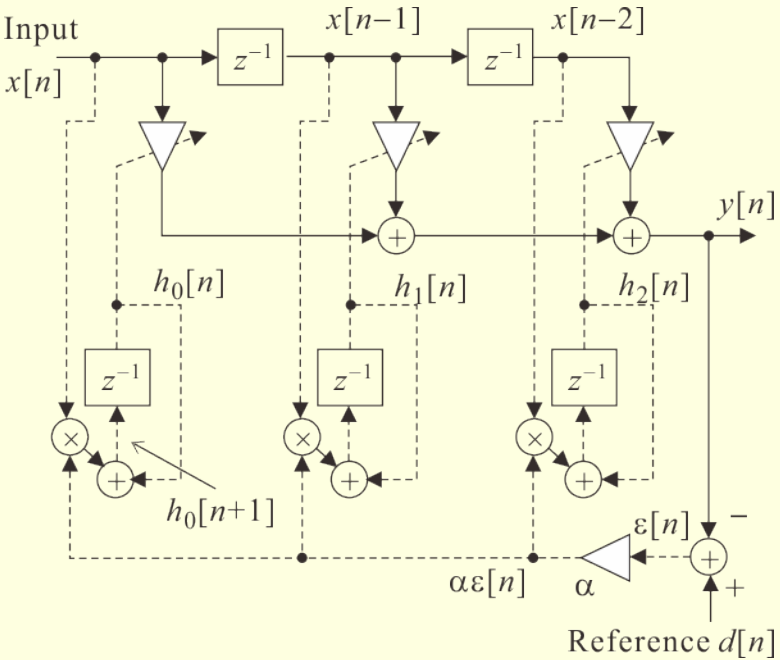
http://t-filter.engineerjs.com/



Adaptive filter



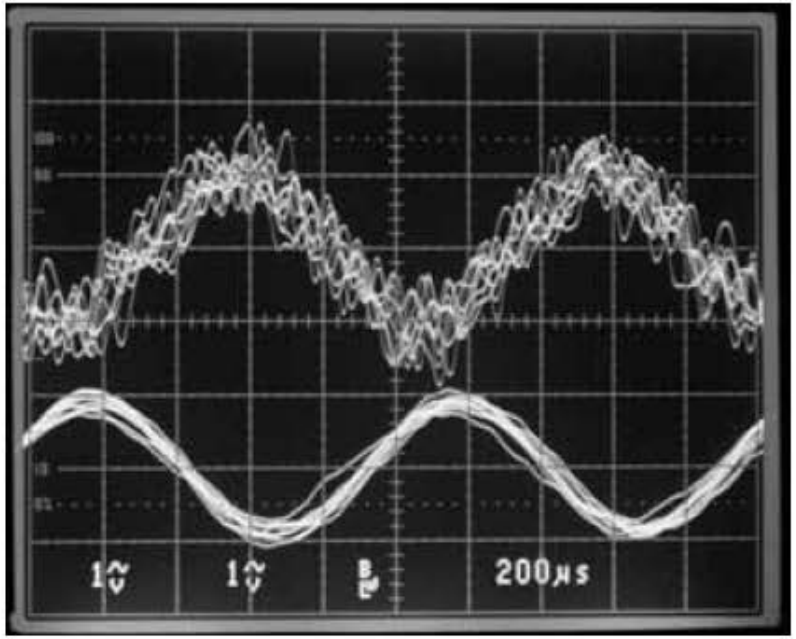
block diagram example



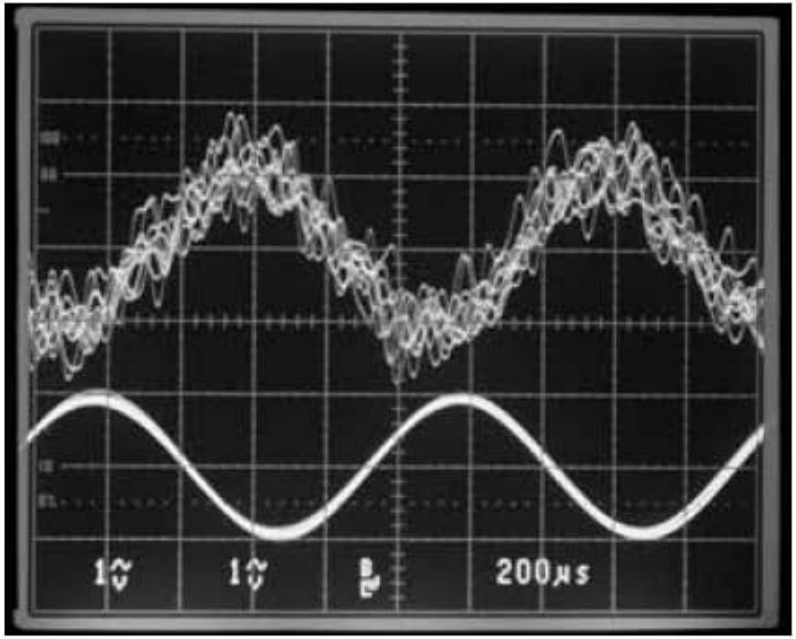
Least mean square method:

$$h_k[l+1] = h_k[l] + 2\alpha\epsilon[l]x[l-k]$$

Adaptive filter (adaptive line enhancer)



(a) $\mu = 1 \times 10^{-3}$の場合



(b) $\mu = 1 \times 10^{-5}$の場合