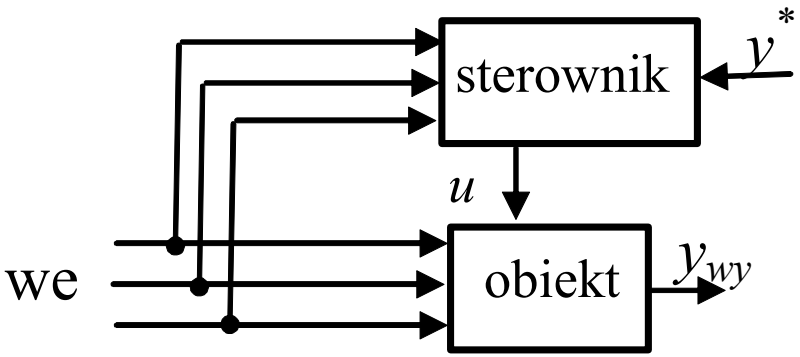
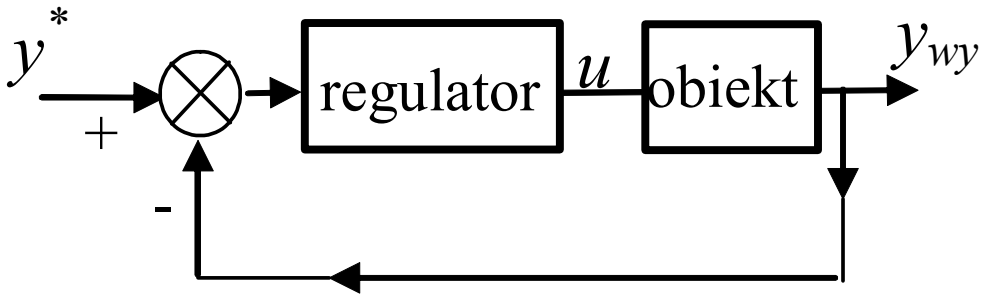
Podstawowe typy układów sterowania

Sterowanie w

układzie otwartym	układzie zamkniętym (regulacja)





- *znany „przepis” na sterowanie*
- *stabilne*
- *niedokładne*

- *sterowanie jest wypracowywane*
- *stabilne/niestabilne*
- *dokładne*

Feedforward Control *Feedback Control*

Process Control

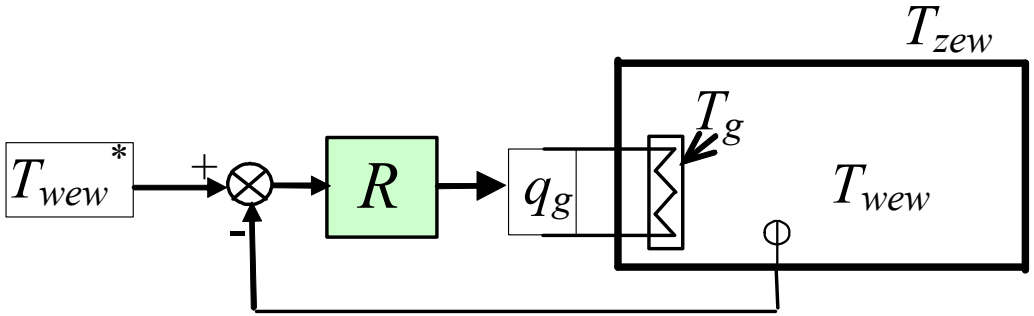
Projektowanie UR z regulatorem PID

1. Założenia projektowe, wybór układu i regulatora
   - co jest celem – wybór PV
   - jak zrealizować – wybór CV, warunki techniczne
   - sterowanie czy regulacja
   - algorytm regulacji (ciągła, binarna)

2. Dobór nastaw - zapewnienie stabilności
3. Ocena jakości – pomiar wskaźników jakości
4. Optymalizacja – szybciej i dokładniej

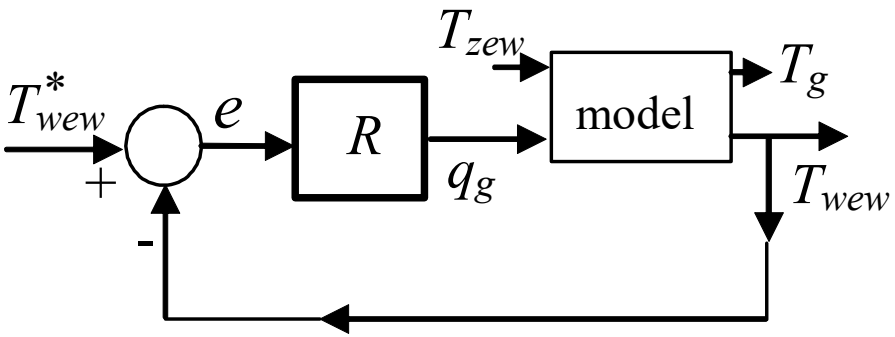
Podstawowy układ regulacji

PV = T_{wew}, CV = q_g, regulacja PI

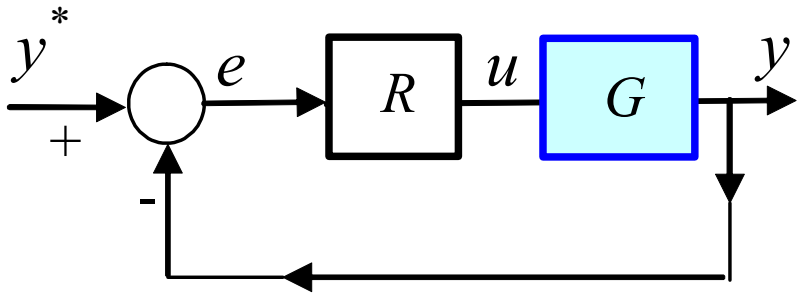


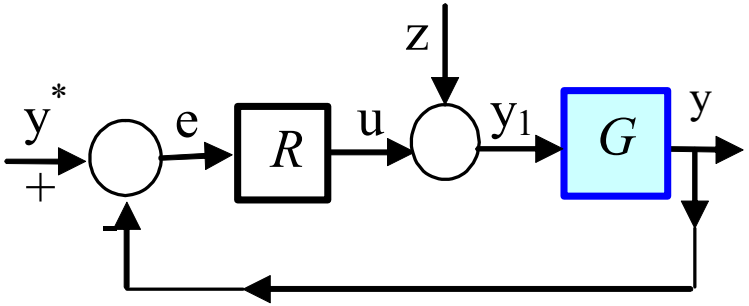
$$\dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u(t)$$
$$y(t) = \mathbf{C}x(t) + \mathbf{D}u(t)$$

MIMO



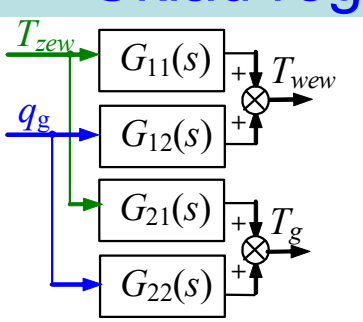
Podstawowy układ regulacji

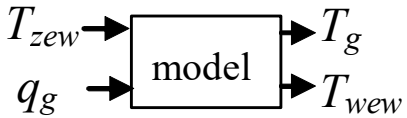


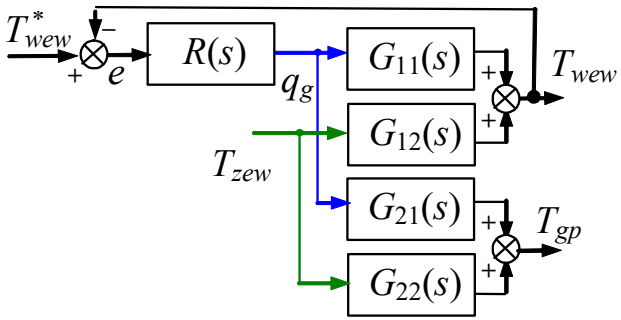


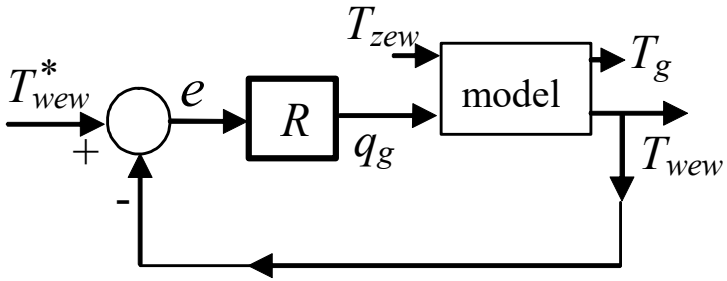
y – zmienna procesowa (PV – Process Variable)
y^* – wartość zadana (SP – Set-Point)
e – uchyb regulacji (control error)
u – zmienna sterująca (CV – Control Variable)
z – zakłócenia

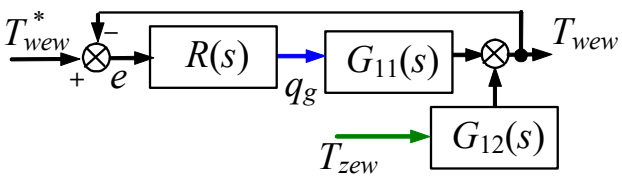
Układ regulacji – praktyczny i teoretyczny

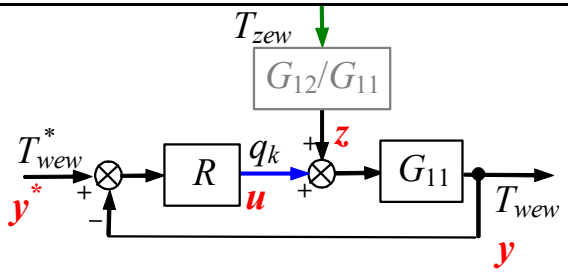


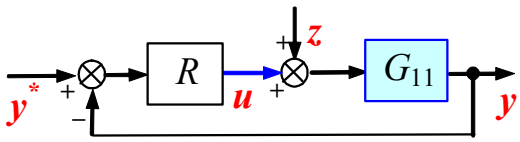


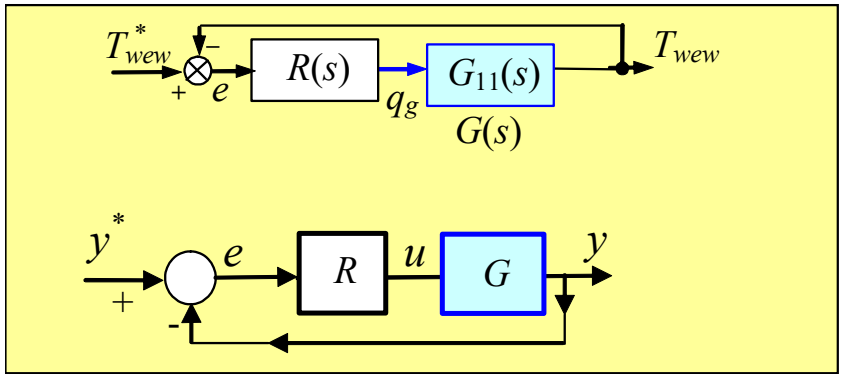


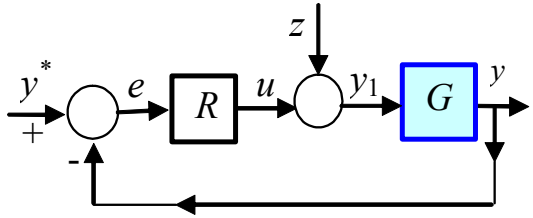




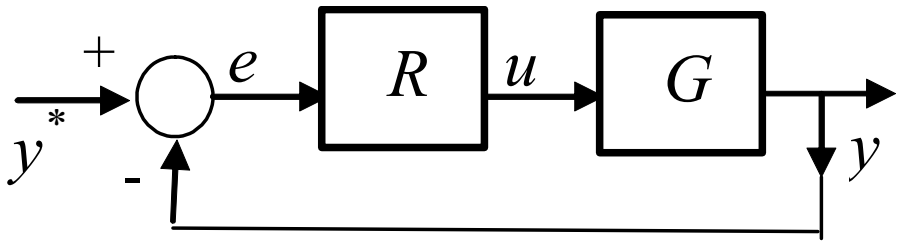








Transmitancje układu regulacji ciągłej (bez zakłóceń)



$$\begin{cases} Y(s) = G(s)U(s) \\ U(s) = R(s)E(s) \\ E(s) = Y^*(s) - Y(s) \end{cases}$$

$$\frac{Y}{Y^*} = G_z = \frac{RG}{1+RG}$$

t.układu zamkniętego

$$Y(s) = \frac{R(s)G(s)}{1+R(s)G(s)} Y^*(s)$$

$$\frac{E}{Y^*} = G_e = \frac{1}{1+RG}$$

t.uchybowa

$$E(s) = \frac{1}{1+R(s)G(s)} Y^*(s)$$

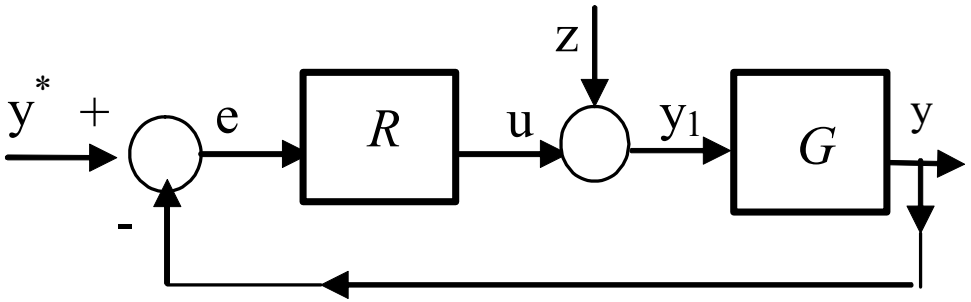
$$G(s) = \frac{L_o(s)}{M_o(s)}$$

$$G_z = \frac{L_o(s)L_R(s)}{M_o(s)M_R(s) + L_o(s)L_R(s)}$$

$$R(s) = \frac{L_R(s)}{M_R(s)}$$

$$G_e = \frac{M_o(s)M_R(s)}{M_o(s)M_R(s) + L_o(s)L_R(s)}$$

Transmitancje układu regulacji ciągłej (z zakłóceniami)



$$\begin{cases} Y(s) = G(s)Y_1(s) \\ Y_1(s) = Z(s) + U(s) \\ U(s) = R(s)E(s) \\ E(s) = Y^*(s) - Y(s) \end{cases}$$

$$\begin{bmatrix} E \\ Y \end{bmatrix} = \begin{bmatrix} \dfrac{1}{1+RG} & \dfrac{G}{1+RG} \\ \dfrac{RG}{1+RG} & \dfrac{-G}{1+RG} \end{bmatrix} \cdot \begin{bmatrix} Y^* \\ Z \end{bmatrix}$$

$$\begin{bmatrix} E \\ Y \end{bmatrix} = \begin{bmatrix} G_e & G_{ez} \\ G_z & G_{yz} \end{bmatrix} \cdot \begin{bmatrix} Y^* \\ Z \end{bmatrix}$$

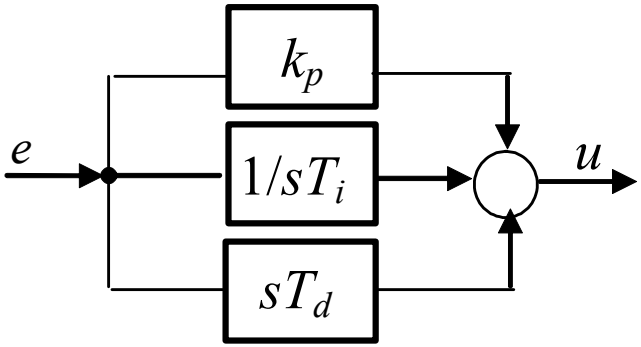
$$G_z = 1 - G_e$$
$$G_{yz} = -G_{ez}$$

- t.układu zamkniętego
$$G_z(s) = \frac{Y(s)}{Y^*(s)}$$
- t.uchybowa
$$G_e(s) = \frac{E(s)}{Y^*(s)}$$
- t.uchybowo-zakłóceniowa
$$G_{ez}(s) = \frac{E(s)}{Z(s)}$$
- t.wyjściowo-zakłóceniowa
$$G_{yz}(s) = \frac{Y(s)}{Z(s)}$$

Regulator PID: struktura

$$u(t) = k_p e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d \frac{de(t)}{dt} + U_0$$

$$R = K_p + \frac{1}{T_i s} + T_d s$$



PID-IND

(IND – INDependent algorithm)

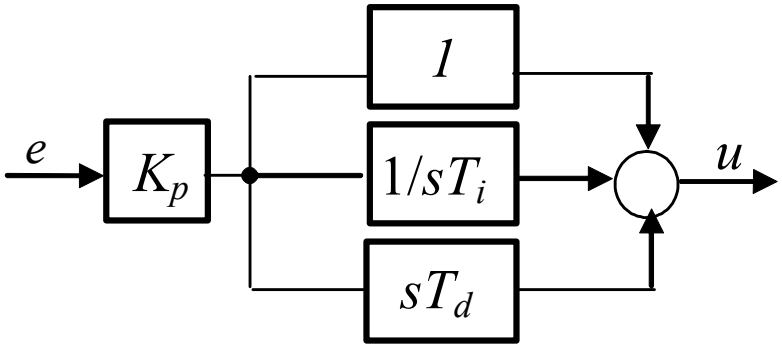
Simulink: PID Parallel

$$R_{par} = P + I\frac{1}{s} + D\frac{Ns}{s+N}$$

$$Ds\frac{1}{\frac{1}{N}s+1}$$

$$u(t) = K_p\left[e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right] + U_0$$

$$R = K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$



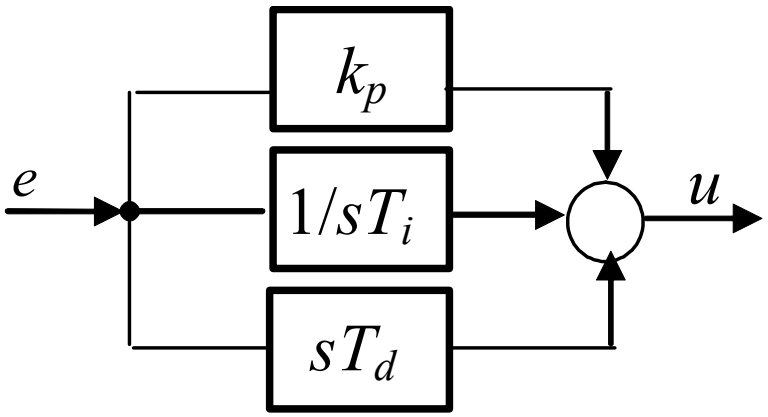
PID-ISA

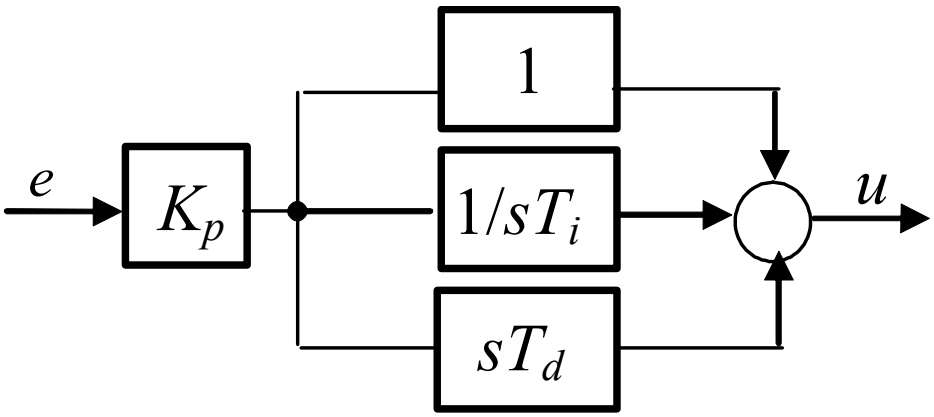
(ISA – Ideal Standard Algorithm)

Simulink: PID ideal

$$R_{id} = P\left(1 + I\frac{1}{s} + D\frac{Ns}{s+N}\right)$$

Regulator PID: nastawy (parametry)





$$R = k_p + \frac{1}{T_i s} + T_d s$$

$$R = K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$

K_p wzmocnienie
T_i czas całkowania (cz.zdwojenia)
T_d czas różniczkowania (cz.wyprzedzenia)

PID-kaskadowy (interacting)

PID kaskadowy (interacting):

$$K_p \frac{T_i s + 1}{T_i s}(T_d s + 1) = K_p\left(\frac{1}{T_d s} + 1\right)(T_d s + 1)$$

PI: $$K_p \frac{T_i s + 1}{T_i s} = K_p\left(\frac{1}{T_d s} + 1\right)$$

PID ISA (non-interacting):

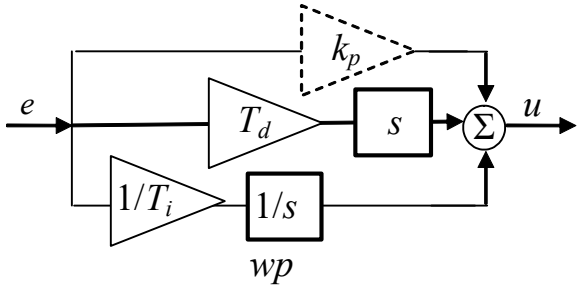
$$K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$

PI: $$K_p\left(1 + \frac{1}{T_i s}\right)$$

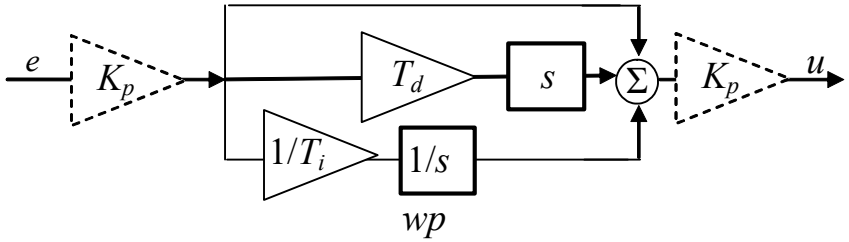
Regulator ciągły: Struktura PID

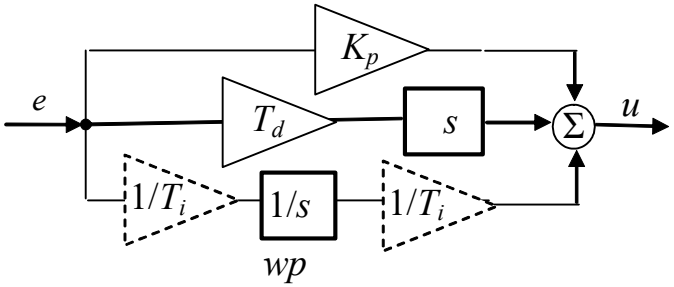
Realizacja PID - możliwości

$$R = K_p + \frac{1}{T_i s} + T_d s$$

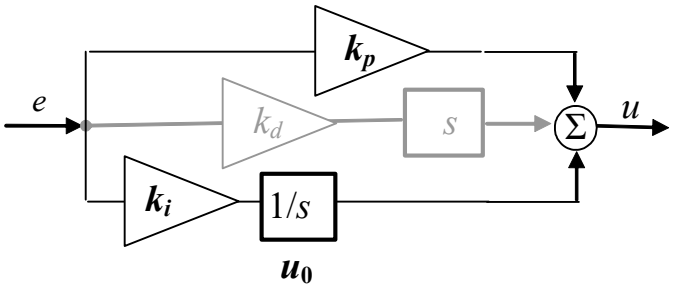


$$R = K_p(1 + \frac{1}{T_i s} + T_d s)$$

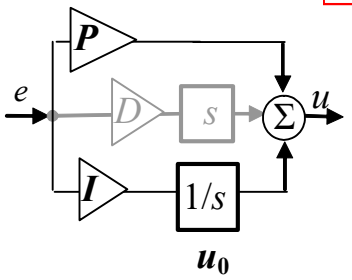




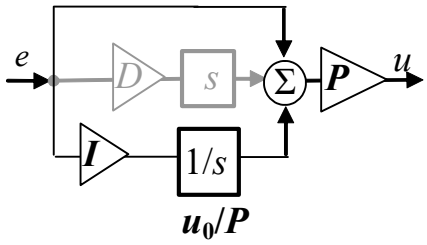
Realizacja PID - wybór



Simulink



PID Paralell



PID Ideal

Simulink -> Continous -> PID Controller

Function Block Parameters: PID Controller

PID Controller

This block implements continuous- and discrete-time PID control algorithms and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID Form: Parallel

Time domain:

- (•) Continuous-time
- ( ) Discrete-time

Main | PID Advanced | Data Types | State Attributes

Controller parameters

Proportional (P):	1
Integral (I):	1
Derivative (D):	0
Filter coefficient (N):	100

Compensator formula

$$P + I\frac{1}{s} + D\frac{N}{1+N\frac{1}{s}}$$

Initial conditions

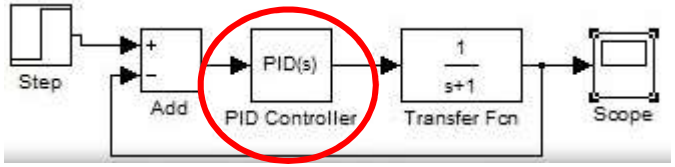
Source:	internal
Integrator:	0
Filter:	0

External reset: none

[ ] Ignore reset when linearizing

[x] Enable zero-crossing detection

OK



Function Block Parameters: PID Controller

PID Controller

This block implements continuous- and discrete-time PID control algorithms and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID Form: Ideal

Time domain:

- (•) Continuous-time
- ( ) Discrete-time

Main | PID Advanced | Data Types | State Attributes

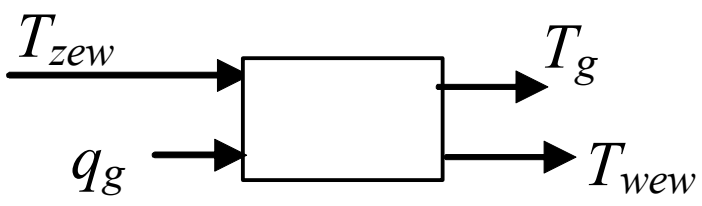
Controller parameters

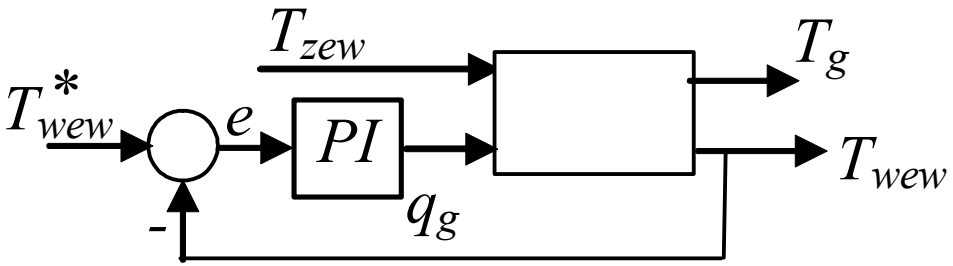
Proportional (P):	1
Integral (I):	1
Derivative (D):	0
Filter coefficient (N):	100

Compensator formula

$$P\left(1 + I\frac{1}{s} + D\frac{N}{1+N\frac{1}{s}}\right)$$

Regulator ciągły PI (zastosowanie e_k =0 w symulacji)





$$\begin{cases} 0 = K_{cg}\left(T_g - T_{wew}\right) - K_{cw}\left(T_{wew} - T_{zew}\right) \\ 0 = q_g - K_{cg}\left(T_g - T_{wew}\right) \end{cases}$$

$$\begin{cases} 0 = K_{cg}\left(T_g - T_{wew}\right) - K_{cw}\left(T_{wew} - T_{zew}\right) \\ 0 = q_g - K_{cg}\left(T_g - T_{wew}\right) \\ T_{wew} = T_{wew}^* \end{cases}$$

wejścia: T_{zew0}, q_{g0}

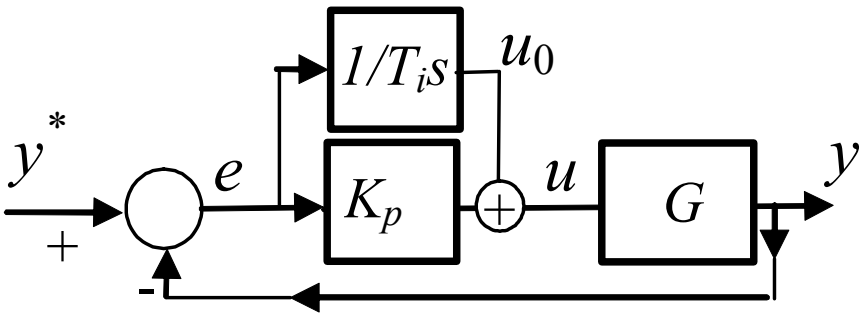
$T_{wew0} = \ldots$

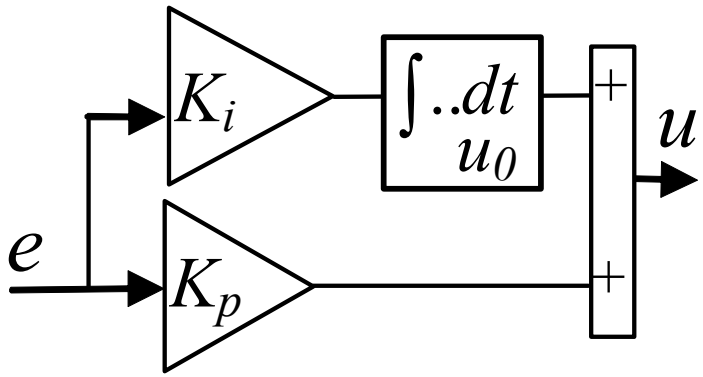
$T_{g0} = \ldots$

wejścia: $T_{zew0}, T_{wew0}^*, e = 0$

$q_{g0} = \ldots$ $\qquad T_{wew0} = T_{wew0}^*$

$T_{g0} = \ldots$





Patrz: 10_RegulatoryProste