

Implementacja i walidacja układów dyskretnych - część rozszerzona

dr hab. inż. Dominik Sierociuk

Materiały zostały współfinansowane przez Unię Europejską ze środków Europejskiego Funduszu Społecznego w ramach Programu Operacyjnego Wiedza Edukacja Rozwój 2014-2020, projekt „NERW PW Nauka-Edukacja-Rozwój-Współpraca”



**Fundusze
Europejskie**
Wiedza Edukacja Rozwój

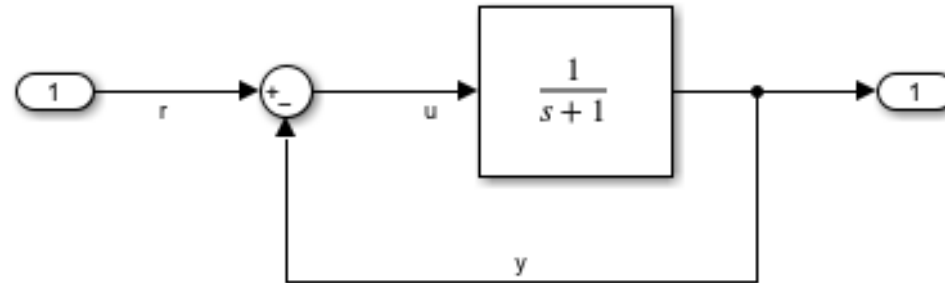


**Rzeczpospolita
Polska**

Unia Europejska
Europejski Fundusz Społeczny



Pętla sprzężenia zwrotnego



$$G_z(s) = \frac{Y(s)}{R(s)} = \frac{G(s)}{1 + G(s)} = \frac{\frac{l(s)}{m(s)}}{1 + \frac{l(s)}{m(s)}} = \frac{\frac{l(s)}{m(s)}}{\frac{m(s) + l(s)}{m(s)}} = \frac{l(s)}{l(s) + m(s)}$$

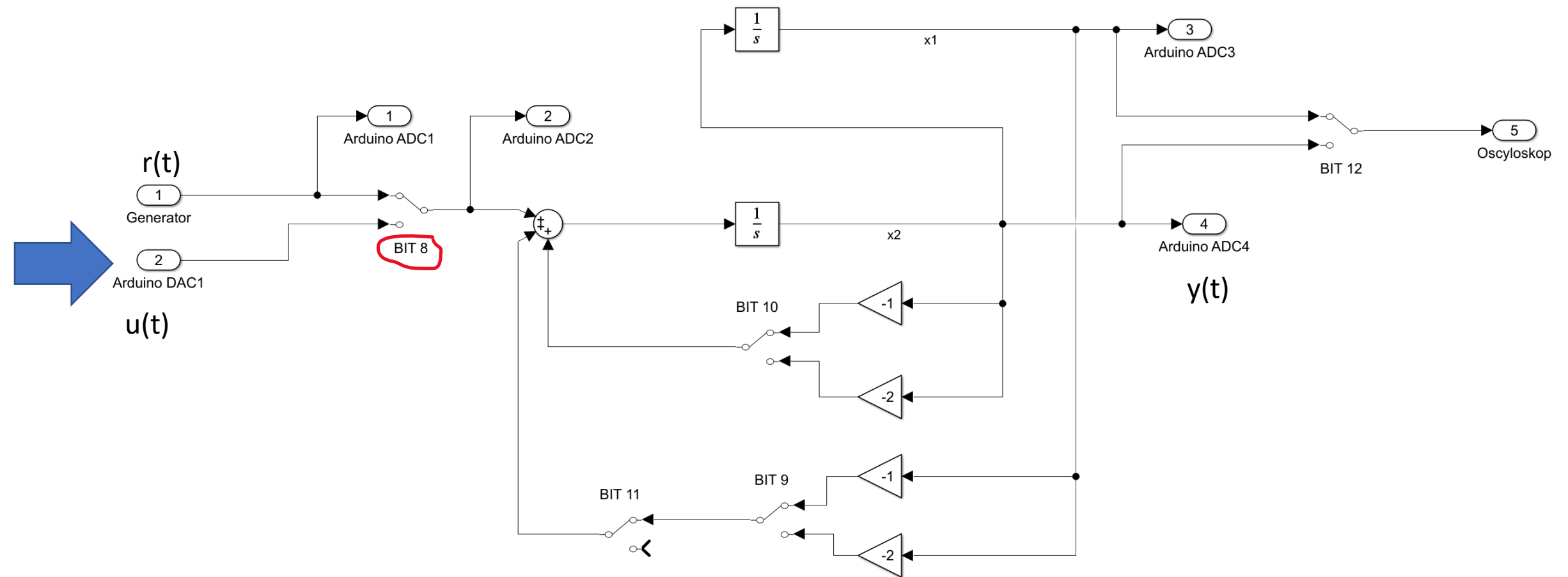
Dla $G(s) = \frac{1}{s+1}$ mamy $G_z(s) = \frac{1}{s+2}$

$$y(\infty) = \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} s \frac{1}{s+1} \frac{1}{s} = 1$$

$$y(\infty) = \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} s \frac{1}{s+2} \frac{1}{s} = \frac{1}{2}$$

Uwaga: użycie twierdzenia o wartościach granicznych wymaga układu stabilnego

Schemat modelu analogowego I i II rzędu



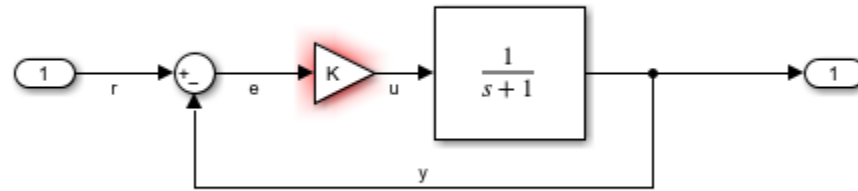
$$u(t) = e(t) = r(t) - y(t)$$

```
void setup()
{
  analogReadResolution(12);
  Wire.begin();
  //Serial.begin(9600);
  Serial.begin(115200);
  pinMode(13, OUTPUT);

  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);

  digitalWrite(8, HIGH); /* 0 - sterowanie z generatora, 1- sterowanie z Arduino */
  digitalWrite(9, LOW); /*A_1,2 0 - parametr -1, 1 - parametr -2 */
  digitalWrite(10, LOW); /*A_2,2 0 - parametr -1, 1 - parametr -2 */
  digitalWrite(11, HIGH); /* 0 - rząd 2, 1 - rząd 1*/
  digitalWrite(12, HIGH); /* wyjście oscyloskopowe 0 - x_1 (rząd 2), 1 - x_2 (rząd 1)*/
```

Regulator P



$$G(s) = \frac{1}{s+1} \quad G_{reg}(s) = K \quad G_z(s) = \frac{K}{s+1+K}$$

$$y(\infty) = \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} s \frac{K}{s+1+K} \frac{1}{s} = \frac{K}{1+K}$$

Uwaga: użycie twierdzenia o wartościach granicznych wymaga układu stabilnego

$$u(k) = Ke(k) = K(r(k) - y(k))$$

Uchyb

$$e(k) = r(k) - y(k)$$

$$G_{ez}(s) = \frac{E(s)}{R(s)} = \frac{1}{1 + G(s)} = \frac{1}{1 + \frac{l(s)}{m(s)}} = \frac{1}{\frac{m(s) + l(s)}{m(s)}} = \frac{m(s)}{l(s) + m(s)}$$

$$G(s) = \frac{1}{s+1} \quad G_{reg}(s) = K \quad G_{ez}(s) = \frac{s+1}{s+1+K}$$

$$e(\infty) = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} s \frac{s+1}{s+1+K} \frac{1}{s} = \frac{1}{1+K}$$

Uwaga: użycie twierdzenia o wartościach granicznych wymaga układu stabilnego

Uwaga 2: zwiększając K zmniejszamy uchyb, ale możemy pogorszyć stabilność

Regulator PI

$$m^\#(0) \neq 0, l(0) \neq 0$$

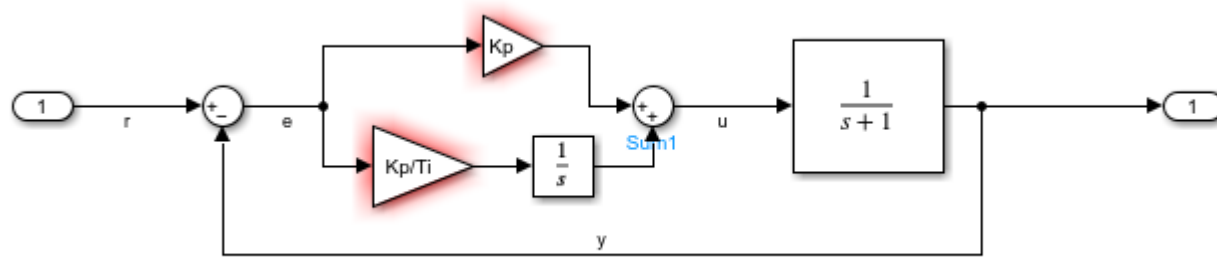
$$G(s) = \frac{l(s)}{s^p(m^\#(s))} \quad r(t) = t^n 1(t) \quad R(s) = \frac{n!}{s^{n+1}}$$

$$G_{ez}(s) = \frac{E(s)}{R(s)} = \frac{s^p(m^\#(s))}{l(s) + s^p(m^\#(s))}$$

$$e(\infty) = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} s \frac{s^p(m^\#(s))}{l(s) + s^p(m^\#(s))} \frac{n!}{s^{n+1}} = \lim_{s \rightarrow 0} \frac{s^{p+1}(m^\#(s))}{l(s) + s^p(m^\#(s))} \frac{n!}{s^{n+1}}$$

$$e(\infty) = \begin{cases} 0 & \text{dla } p > n \\ \text{const} & \text{dla } p = n \\ \infty & \text{dla } p < n \end{cases}$$

Regulator PI (2)



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

$$u(t) = K_p e(t) + \frac{K_p}{T_i} \int_0^t e(\tau) d\tau$$

Dyskretna realizacja

$$u_k = K_p e_k + \frac{K_p}{T_i} I_k \quad \text{gdzie } I_k \text{ jest aproksymacją całkowania}$$

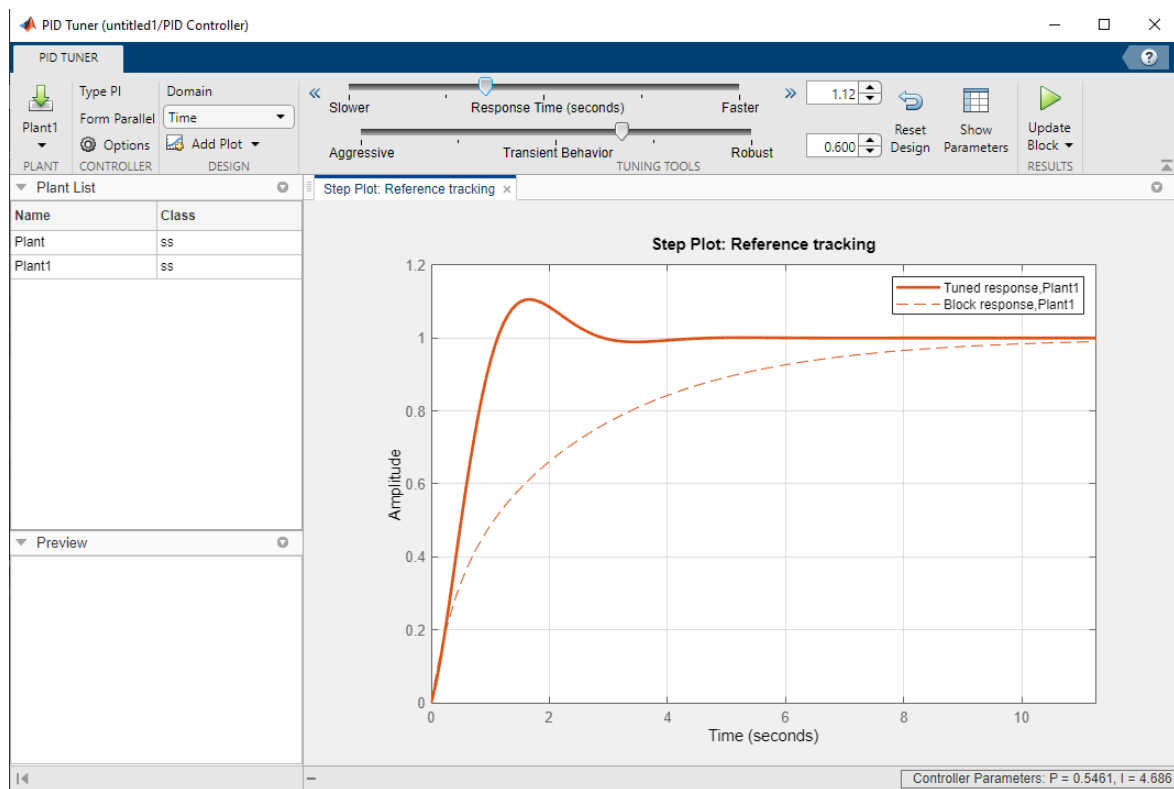
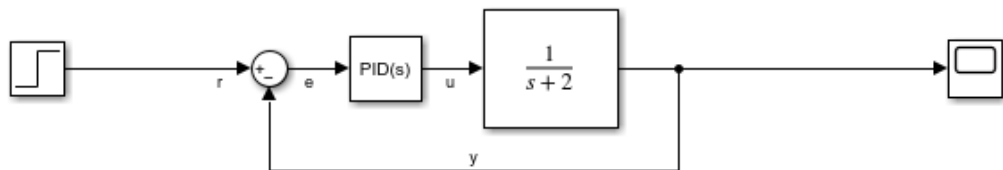
Regulator PI (3)

Aproksymacje całkowania:

- Aproksymacja prostokątami $I_k = I_{k-1} + he_k$
gdzie h – okres całkowania
- Aproksymacja trapezami $I_k = I_{k-1} + \frac{h}{2}(e_k + e_{k-1})$

Prawo sterowania
$$u_k = K_p e_k + \frac{K_p}{T_i} I_k$$

Dobór nastaw w Simulinku



Block Parameters: PID Controller

PID 1dof (mask) (link)

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: PI Form: Parallel

Time domain:

Continuous-time

Discrete-time

Discrete-time settings

Sample time (-1 for inherited): -1

Compensator formula

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

Main Initialization Output Saturation Data Types State Attributes

Controller parameters

Source: internal

Proportional (P): 1

Integral (I): 1

Automated tuning

Select tuning method: Transfer Function Based (PID Tuner App) **Tune...**

Enable zero-crossing detection

OK Cancel Help Apply

Co można dalej wykonać

- Analiza działania układu regulacji dla innych nastaw regulatora, innych typów wymuszeń
- Wykonanie badań dla układu drugiego rzędu i różnych parametrów