

Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania
(projekt grupowy)

Sprawozdanie z projektu i ćwiczenia laboratoryjnego
nr 3, zadanie nr 10

Stanislau Stankevich, Rafał Bednarz, Ostrysz Jakub

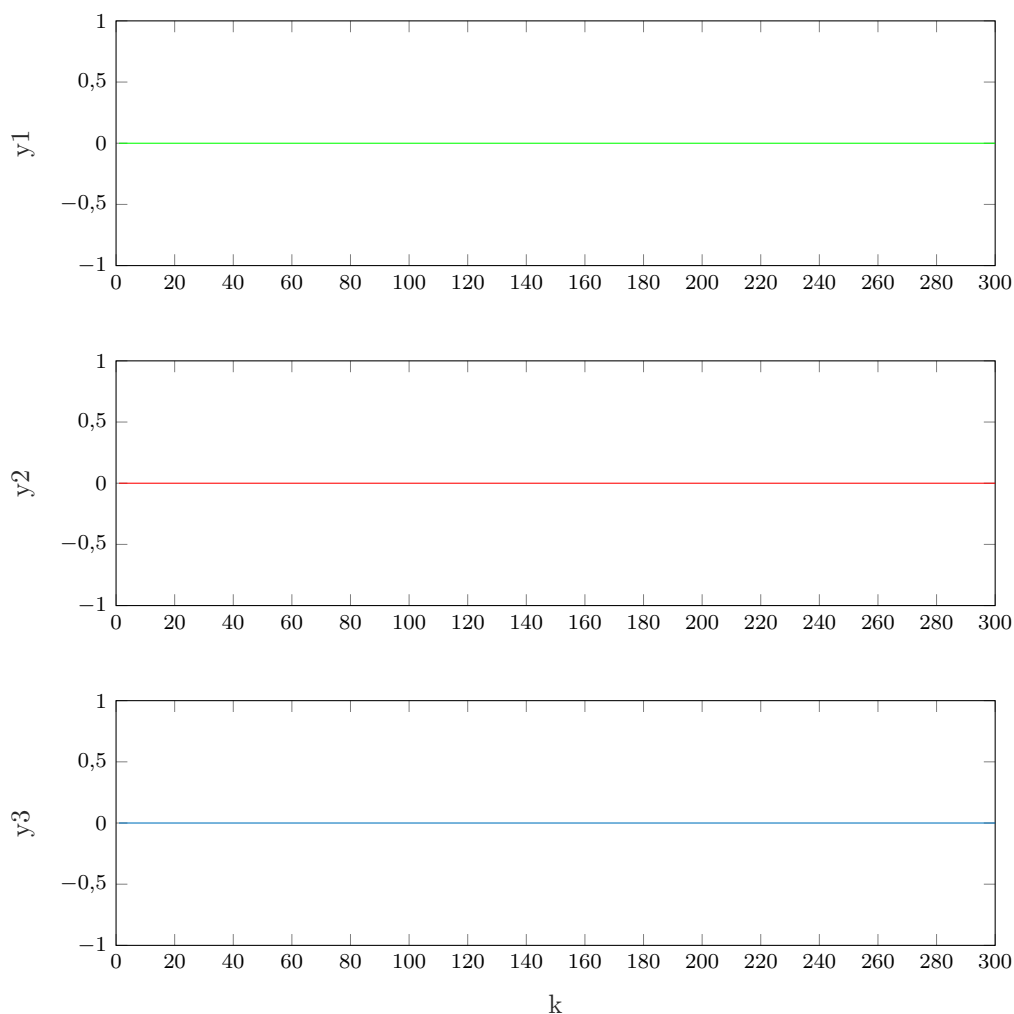
Warszawa, 2021

Spis treści

1. Sprawdzenie punktu pracy	2
2. Odpowiedzi skokowe poszczególnych torów	4
3. Regulatory	5
3.1. PID	5
3.2. DMC	5
4. Eksperymentalne wyznaczenie nastaw	7
4.1. PID	7
4.1.1. $u_1 - y_2; u_2 - y_3; u_3 = const = 0; u_4 - y_1$	7
4.1.2. $u_1 = const = 0; u_2 - y_3; u_3 - y_1; u_4 - y_2$	10
4.2. DMC - wyniki eksperymentów	14
4.3. DMC oszczędny - wyniki eksperymentów	19
4.4. Regulacja DMC - komentarz ogólny	23
5. Laboratorium	24
5.1. Określenie wartości pomiaru temperatury w punkcie pracy	24
5.2. Mechanizm zabezpieczający przed uszkodzeniem stanowiska	24
5.3. Regulator PID	25
5.4. Regulator DMC	29
5.5. Panel operatora	32
5.6. Automat stanów	32
6. Laboratorium: stanowisko TRAS	35
6.1. Zabezpieczenie przed uszkodzeniem stanowiska	35
6.2. Wyznaczenie charakterystyki statycznej	35
6.3. Regulator PID	35
6.4. Regulator PID z użyciem gotowej funkcji	36
6.5. Automat stanów	36
6.6. Wizualizacja procesu	37

1. Sprawdzenie punktu pracy

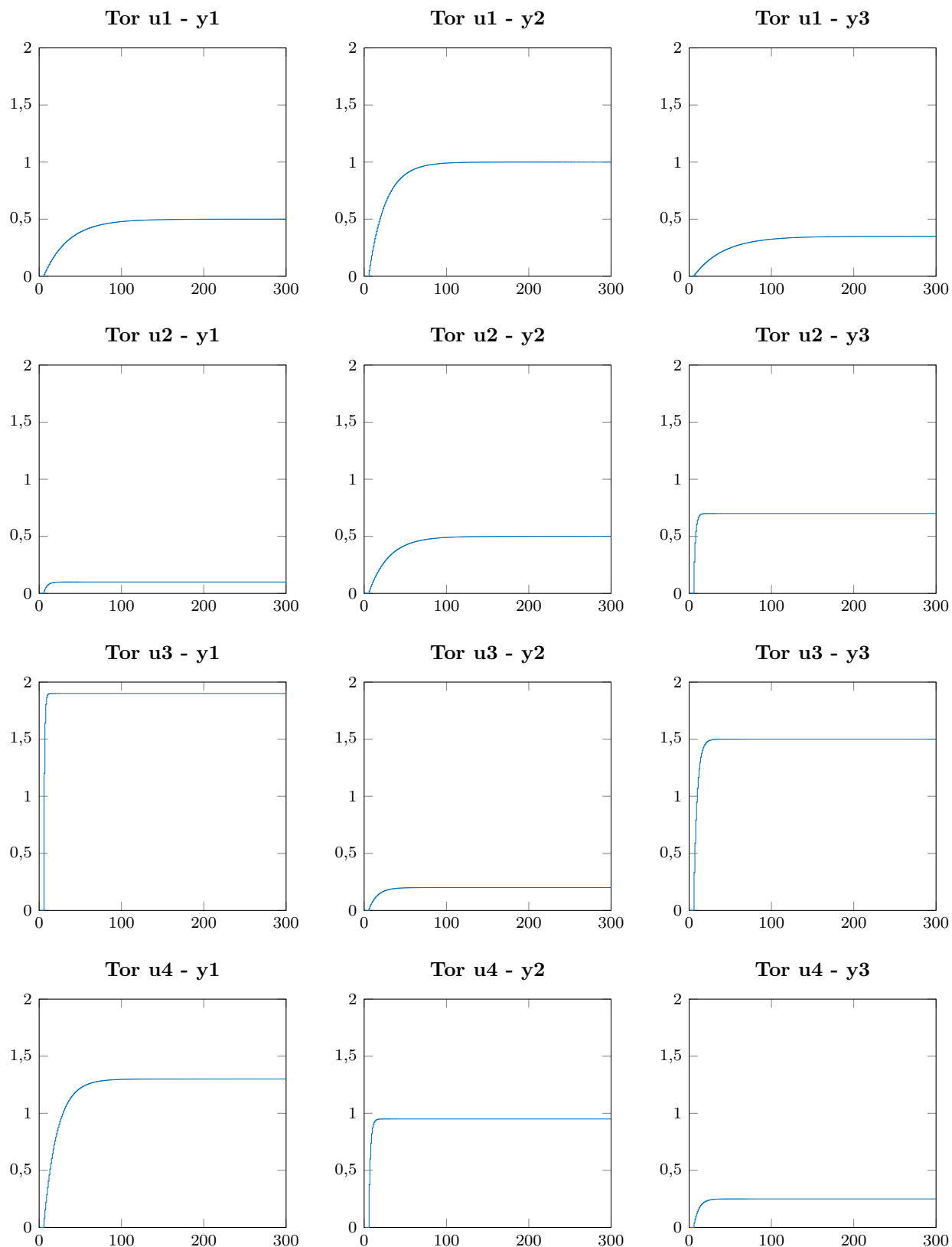
Podając za wejścia same zera, po 300 iteracjach dostajemy następujący przebieg wyjść:



Rys. 1.1. Przebieg wyjść obiektu przy stałych wejściach: $u_1 = 0, u_2 = 0, u_3 = 0$

Każde wyjście ustabilizowało się na wartości 0, więc podany w zadaniu punkt pracy jest zgodny z rzeczywistością.

2. Odpowiedzi skokowe poszczególnych torów

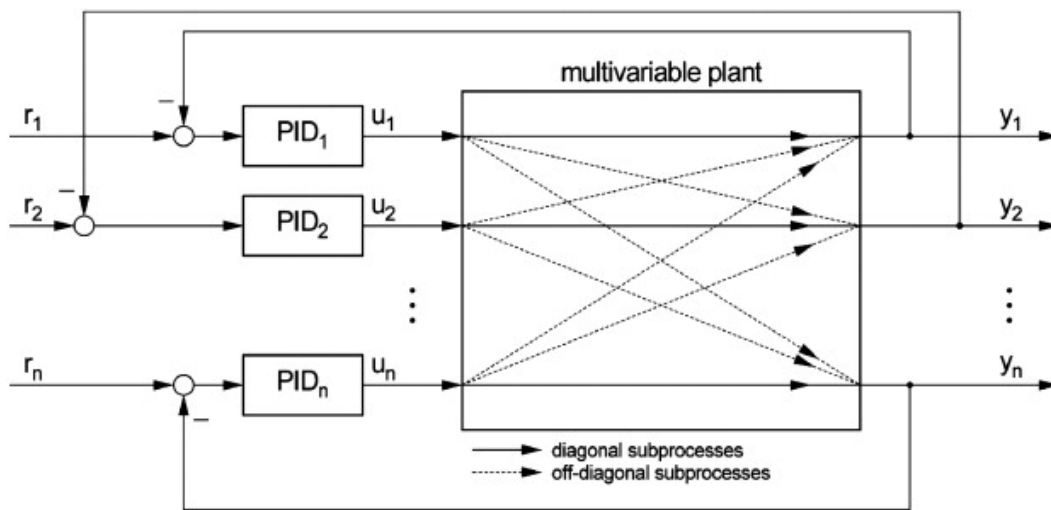


Rys. 2.1. Odpowiedzi poszczególnych torów dla skoku 0 - 1

3. Regulatory

3.1. PID

Ponieważ regulator PID to jest regulator o jednym wejściu i jednym wyjściu, w danym przypadku możemy użyć maksymalnie trzech regulatorów PID, jako że liczba wyjść procesu jest równa 3. Patrząc na odpowiedzi skokowe możemy ocenić które wejście ma największy wpływ na które wyjście i dla takich par $u - y$ nastroić regulatory PID. Na jedno wejście procesu w takim podejściu ustawiamy stałą.



3.2. DMC

Regulator DMC z natury jest wielowymiarowy. Prawo regulacji wielowymiarowego DMC w wersji klasycznej wygląda następująco:

$$\Delta U(k) = K(Y^{zad}(k) - Y^0(k))$$

Wektor przyrostów sterowania wyznacza przyrosty na całym horyzoncie sterowania i ma długość $n_u N_U$.

$$\Delta U(k) = \begin{bmatrix} \Delta u_1(k|k) \\ \vdots \\ \Delta u_{n_u}(k|k) \\ \vdots \\ \Delta u_1(k + N_u - 1|k) \\ \vdots \\ \Delta u_{n_u}(k + N_u - 1|k) \end{bmatrix}$$

Wektory $Y^{zad}(k)$ i $Y(k)$ są wektorami wyjść i wyjść zadanych o długości $n_y N$.

$$Y^{zad}(k) = \begin{bmatrix} y_1^{zad}(k) \\ \vdots \\ y_{n_y}^{zad}(k) \\ \vdots \\ y_1^{zad}(k) \\ \vdots \\ y_{n_y}^{zad}(k) \end{bmatrix}$$

$$Y(k) = \begin{bmatrix} y_1(k) \\ \vdots \\ y_{n_y}(k) \\ \vdots \\ y_1(k) \\ \vdots \\ y_{n_y}(k) \end{bmatrix}$$

K jest macierzą następującą:

$$K = (M^T \Psi M + \Lambda)^{-1} M^T \Psi$$

Y^0 jest wektorem wyjść modelu.

$$Y^0(k) = Y(k) + M^p \Delta U^p(k)$$

W którym $\Delta U^p(k)$ jest wektorem przeszłych przyrostów, natomiast M^p :

$$M^p = \begin{bmatrix} s_2 - s_1 & \cdots & s_D - s_{D-1} \\ s_3 - s_1 & \cdots & s_{D+1} - s_{D-1} \\ \vdots & \ddots & \vdots \\ s_{N+1} - s_1 & \cdots & s_{N+D-1} - s_{D-1} \end{bmatrix}$$

Przy czym każdy element s_i jest macierzą o rozmiarze $n_y \times n_u$.

$$s_i = \begin{bmatrix} s_i^{1,1} & \cdots & s_i^{1,n_u} \\ \vdots & \ddots & \vdots \\ s_i^{n_y,1} & \cdots & s_i^{n_y,n_u} \end{bmatrix}$$

Macierz M wygląda następująco:

$$M = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ s_2 & s_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N & s_{N-1} & \cdots & s_{N-N_u-1} \end{bmatrix}$$

Kierując się tą samą logiką wyznacza się macierz K_e w przypadku uproszczonej wersji DMC, czyli tzw. DMC oszczędnego.

4. Eksperymentalne wyznaczenie nastaw

4.1. PID

W przedstawionych poniżej eksperymentach została przyjęta następująca konwencja nazewnictwa:

PID_1 - pid, kontrolujący wyjście numer 1.

PID_2 - pid, kontrolujący wyjście numer 2.

PID_3 - pid, kontrolujący wyjście numer 3.

K^i, T_i^i, T_d^i - parametry pida i .

Patrząc na wykresy z poprzedniego zadania można zobaczyć że wejście 3 ma największy wpływ zarówno na wyjścia 1 i 3. Biorąc to pod uwagę najpierw spróbowaliśmy ustawić stałą (zero) na to wejście i nie podłączać do niego regulatora, żeby nie regulować jednym wejściem dwóch wyjść jednocześnie.

4.1.1. $u_1 - y_2; u_2 - y_3; u_3 = const = 0; u_4 - y_1$

Zacniemy dobór od następujących parametrów regulatorów:

$K_1 = 0, 1;$

$T_{i1} = 0, 1;$

$T_{d1} = 0, 1;$

$K_2 = 0, 01;$

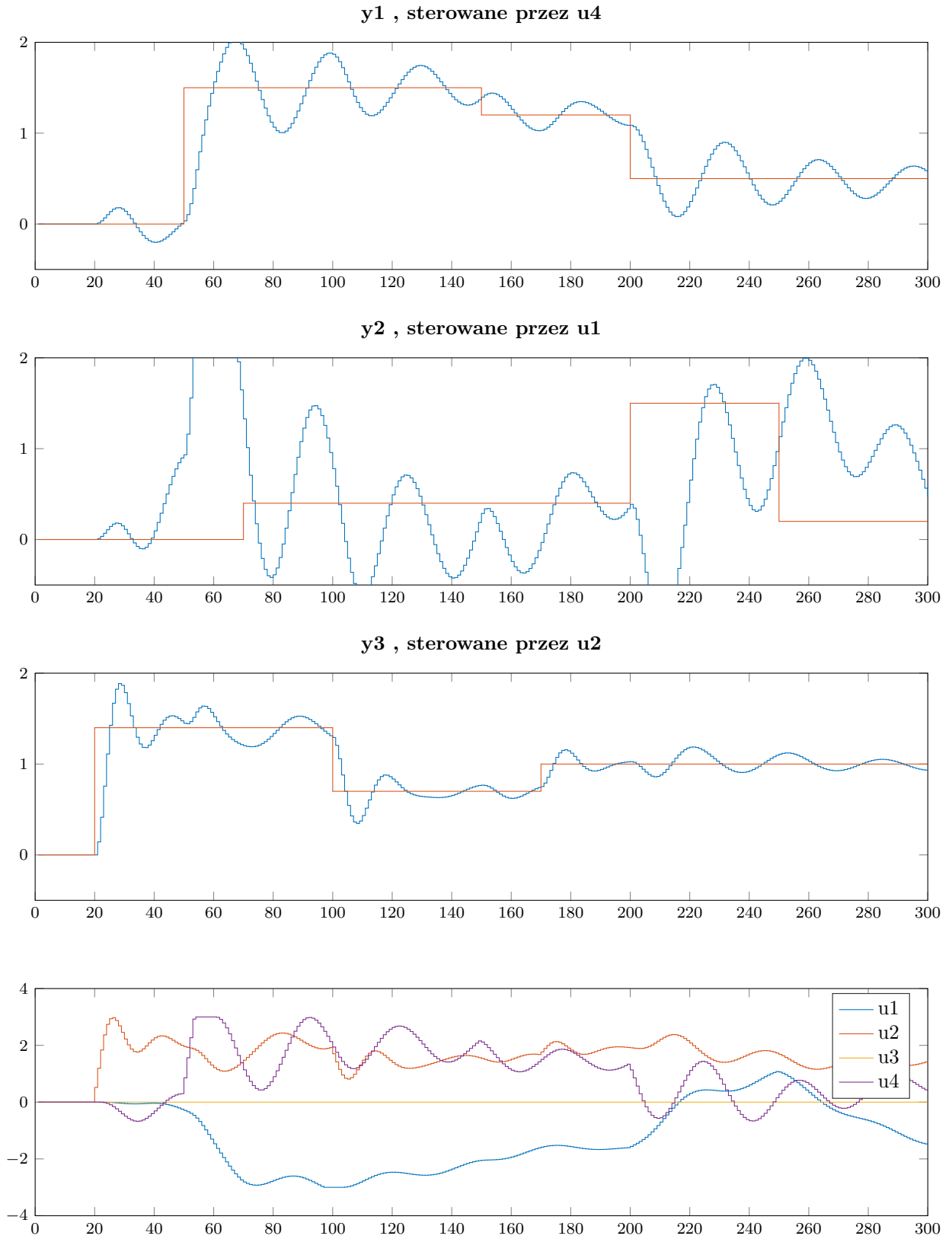
$T_{i2} = 0, 1;$

$T_{d2} = 0, 1;$

$K_3 = 0, 1;$

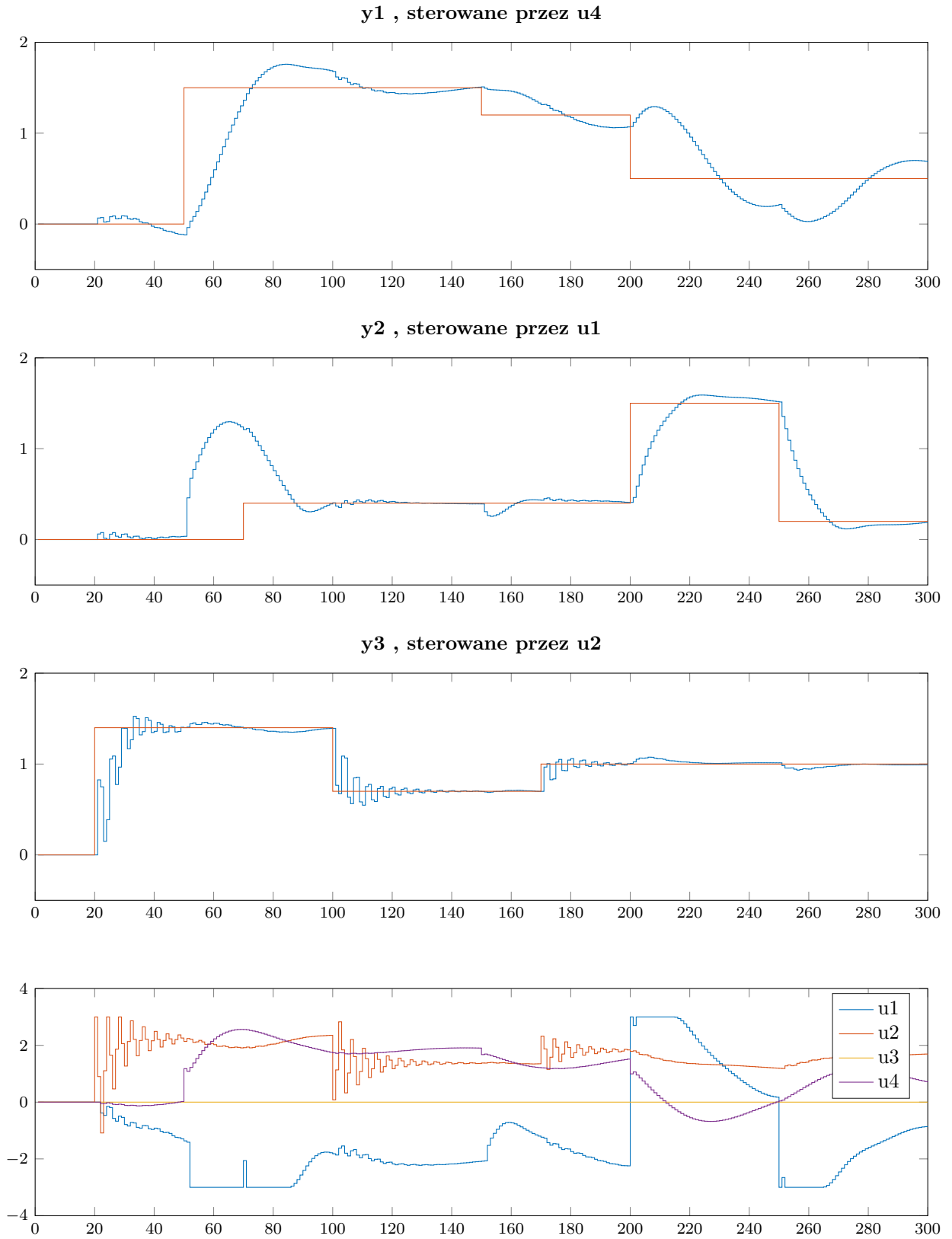
$T_{i3} = 0, 1;$

$T_{d3} = 0, 1.$



Rys. 4.1. $u_3 = 0; K_1 = 0,1; T_{i1} = 0,1; T_{d1} = 0,1; K_2 = 0,01; T_{i2} = 0,1; T_{d2} = 0,1; K_3 = 0,1; T_{i3} = 0,1; T_{d3} = 0,1$.

$$E = 370,8381 \quad (4.1)$$



Rys. 4.2. $u_3 = 0,00$; $K_1 = 0,5$; $Ti_1 = 2$; $Td_1 = 0,2$; $K_2 = 4$; $Ti_2 = 1,5$; $Td_2 = 0,2$; $K_3 = 1$; $Ti_3 = 1$; $Td_3 = 1$.

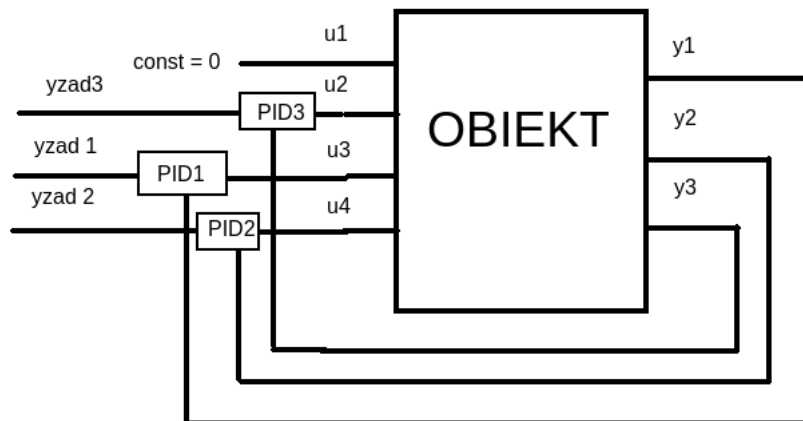
$$E = 96,3728$$

$$(4.2)$$

Jak widać na pierwszym zestawie wykresów, regulacja pierwszego i drugiego wyjścia jest zbyt "luźna". Więc warto było zwiększyć parametry całkowania w obu wypadkach a dodatkowo zwiększyć parametry różniczkowania żeby pozbyć się przesterowań. Przedstawiona wyżej regulacja jest lepsza od poprzedniej i jest to najlepsza regulacja PID którą dało się uzyskać przy takim podłączeniu regulatorów do wejść.

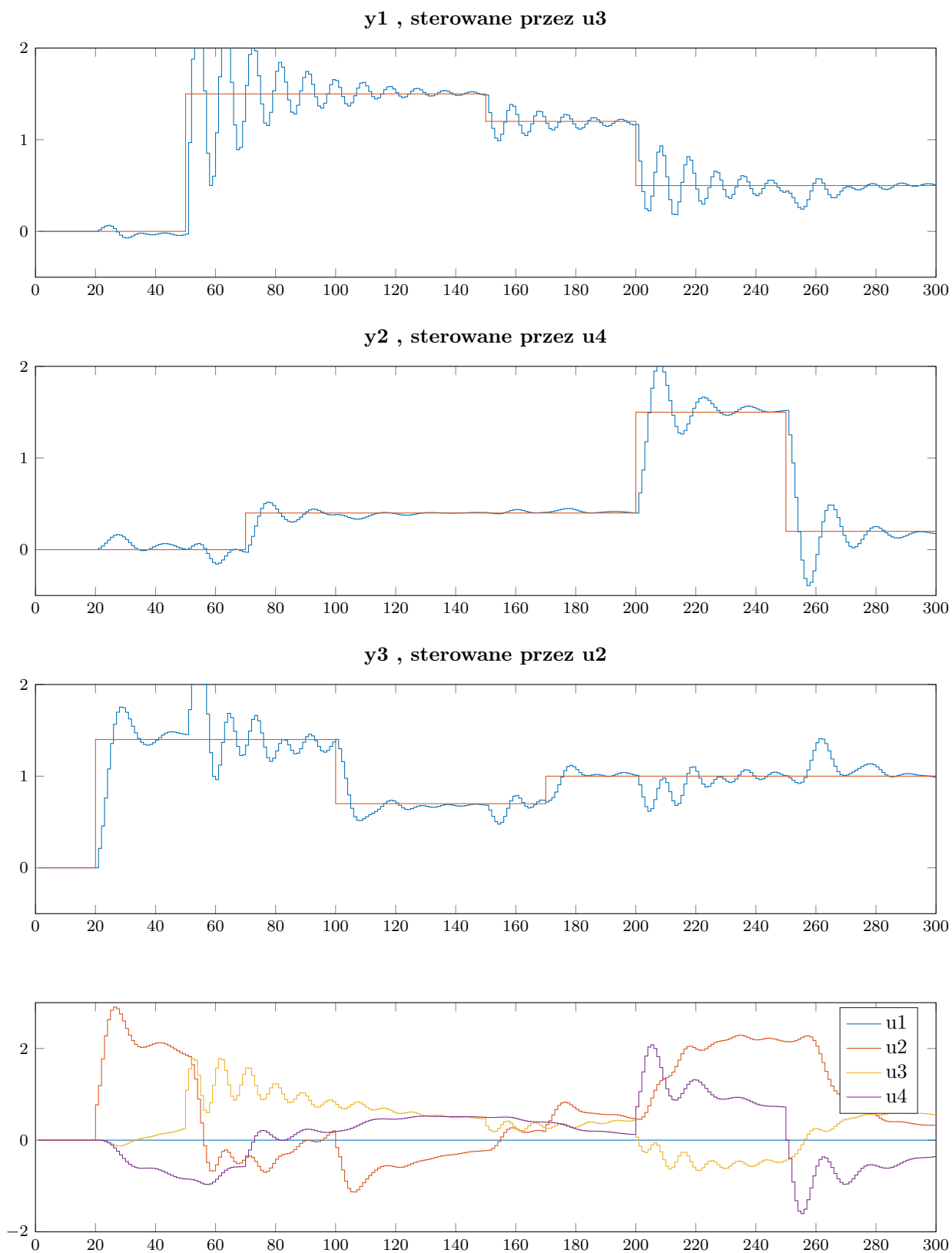
Jak można zauważyć, regulacja ta nie jest najładniejsza. Widać duży skok na drugim wyjściu spowodowany wzrostem wartości zadanej wyjścia pierwszego. W drugim podejściu spróbowaliśmy użyć do sterowania poszczególnymi wyjściami tych wejść, które rzeczywiście mają na nich największy wpływ. Wyjątkiem jest wyjście 3, które nie może być sterowane przez wejście 3, które ma jeszcze większy wpływ na wyjście pierwsze.

4.1.2. $u_1 = \text{const} = 0; u_2 = y_3; u_3 = y_1; u_4 = y_2$



Zacniemy od następujących domyślnych parametrów regulatorów:

$$\begin{aligned}
 K_1 &= 0, 1; \\
 T_{i1} &= 0, 1; \\
 T_{d1} &= 1; \\
 K_2 &= 0, 1; \\
 T_{i2} &= 0, 1; \\
 T_{d2} &= 1; \\
 K_3 &= 0, 1; \\
 T_{i3} &= 0, 1; \\
 T_{d3} &= 1.
 \end{aligned}$$



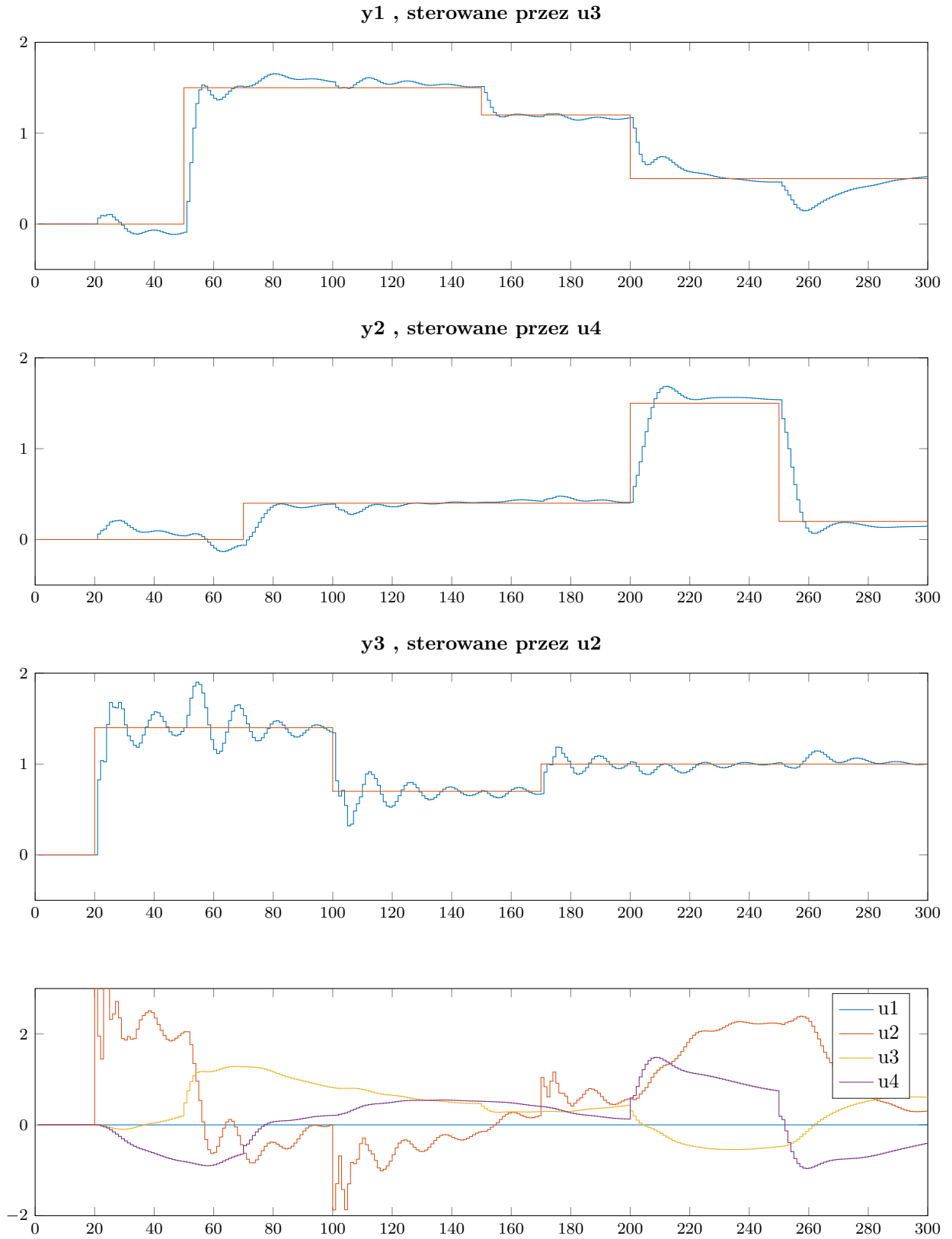
Rys. 4.3. $u_1 = 0,00; K_1 = 0,1; T_{i1} = 0,1; T_{d1} = 1; K_2 = 0,1; T_{i2} = 0,1; T_{d2} = 1; K_3 = 0,1; T_{i3} = 0,1; T_{d3} = 1$.

$$E = 48,2639$$

$$(4.3)$$

Ogólnie regulacja jest lepsza. Zostają kwestie oscylacji na wyjściach natomiast teraz już każde wyjście z mniejszą bądź większą dokładnością śledzi wartość zadaną.

Najlepsza regulacja którą dało się uzyskać przy danej konfiguracji jest zaprezentowana na następnej stronie:

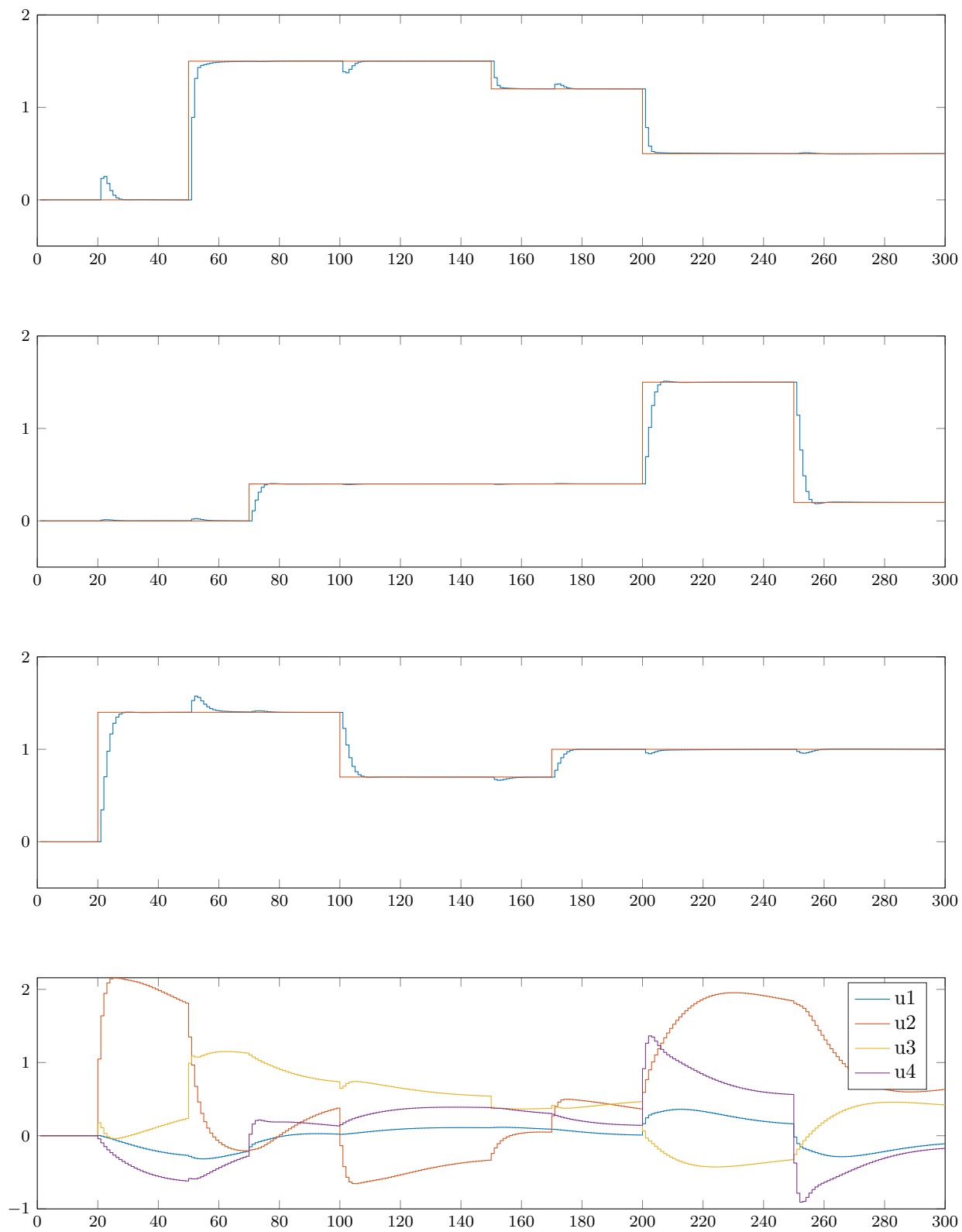


Rys. 4.4. $u_1 = 0,00; K_1 = 0,1; T_{i1} = 0,3; T_{d1} = 0; K_2 = 0,1; T_{i2} = 0,2; T_{d2} = 1; K_3 = 0,1; T_{i3} = 0,04; T_{d3} = 10$.

$$E = 34,9696$$

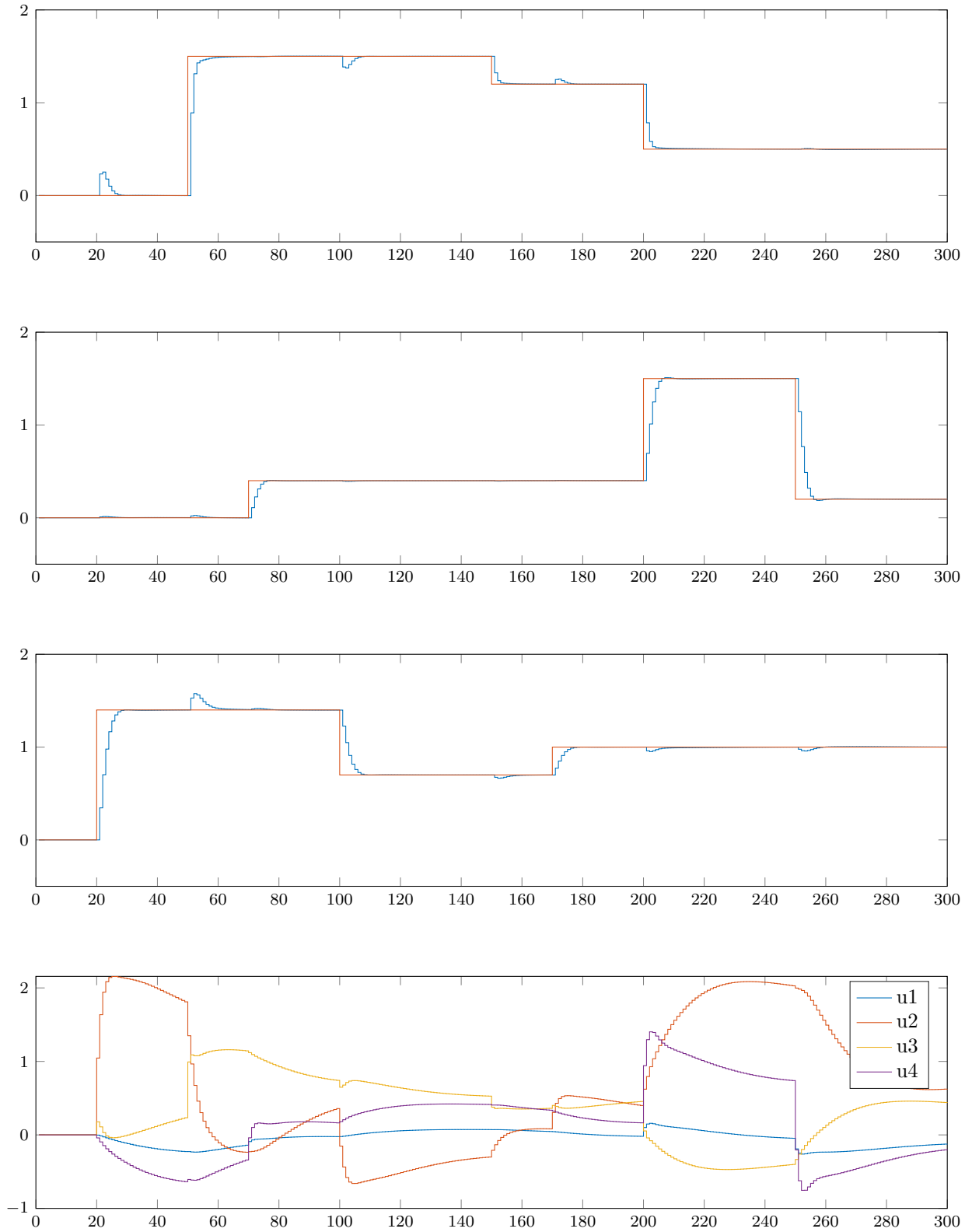
$$(4.4)$$

4.2. DMC - wyniki eksperymentów

Rys. 4.5. $DMC.N = 150; Nu = 50; \mu = [111]; \lambda = [1111]$.

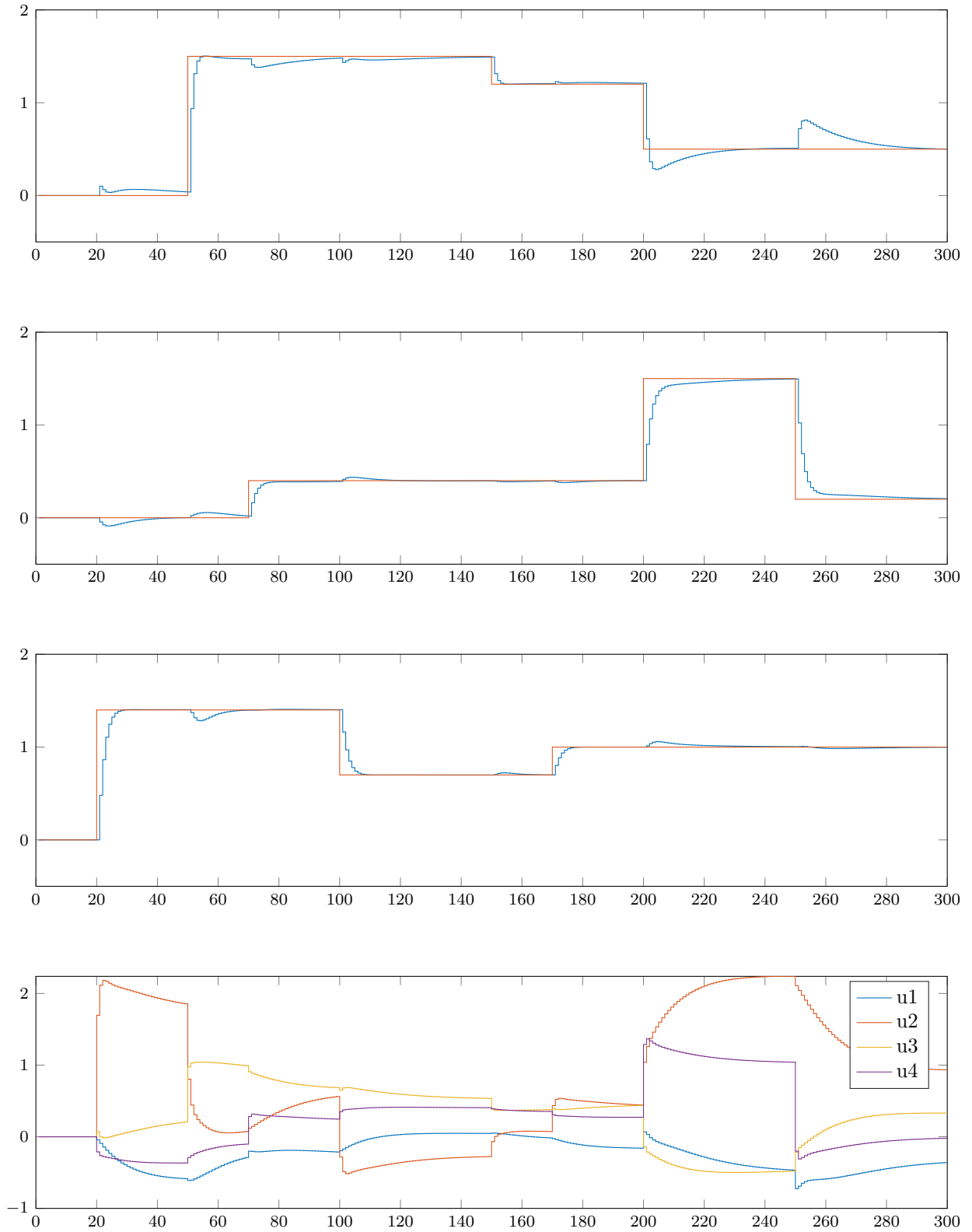
$$E = 14,071$$

$$(4.5)$$

Rys. 4.6. $DMC.N = 10; Nu = 10; \mu = [111]; \lambda = [1111]$.

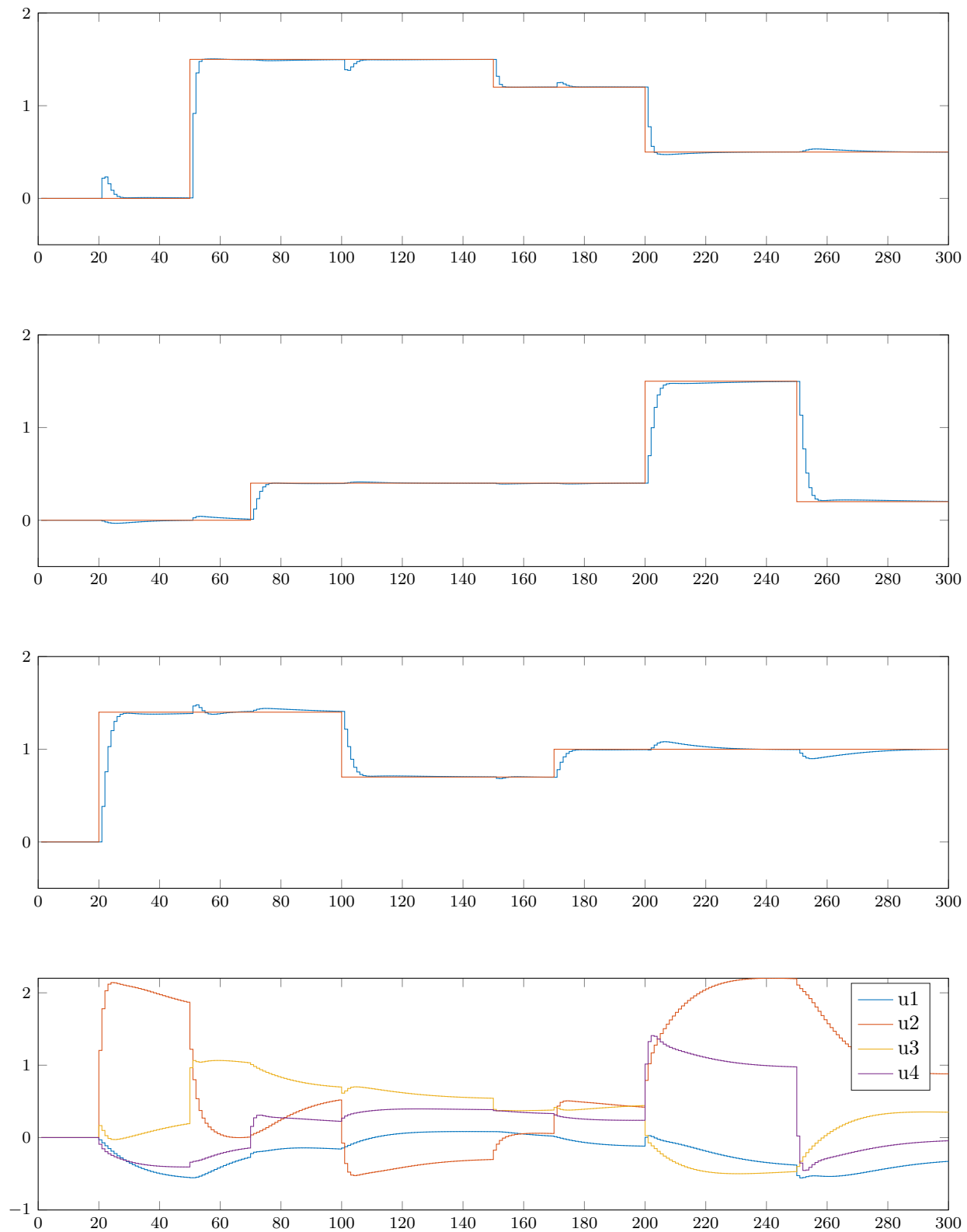
$$E = 14,0854$$

(4.6)

Rys. 4.7. $DMC.N = 10; Nu = 1; \mu = [111]; \lambda = [1111]$.

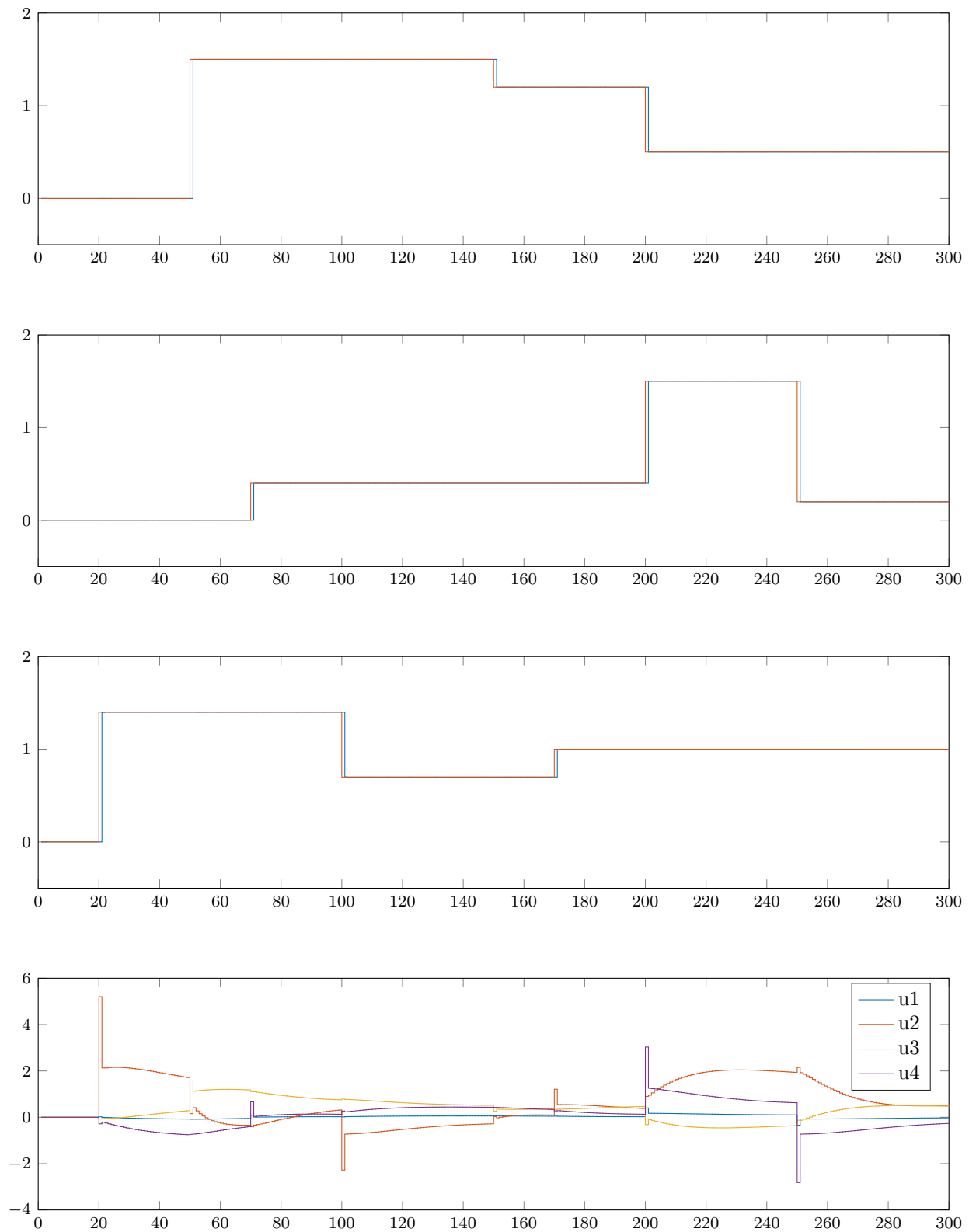
$$E = 14, 1638$$

(4.7)

Rys. 4.8. $DMC.N = 10; Nu = 2; \mu = [111]; \lambda = [1111]$.

$$E = 13,9256$$

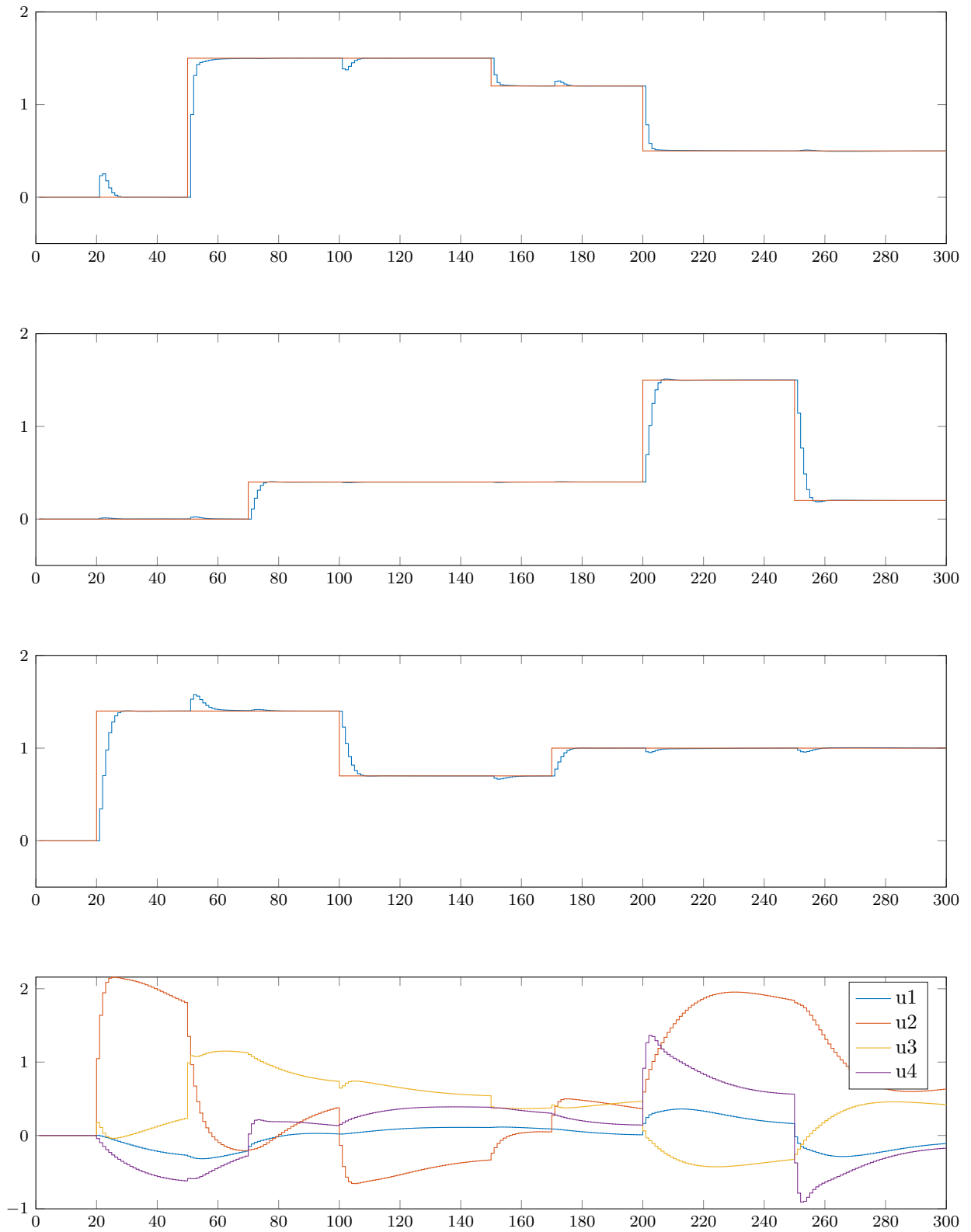
(4.8)

Rys. 4.9. $DMC.N = 1; Nu = 1; \mu = [101010]; \lambda = [0, 0010, 0010, 0010, 001]$.

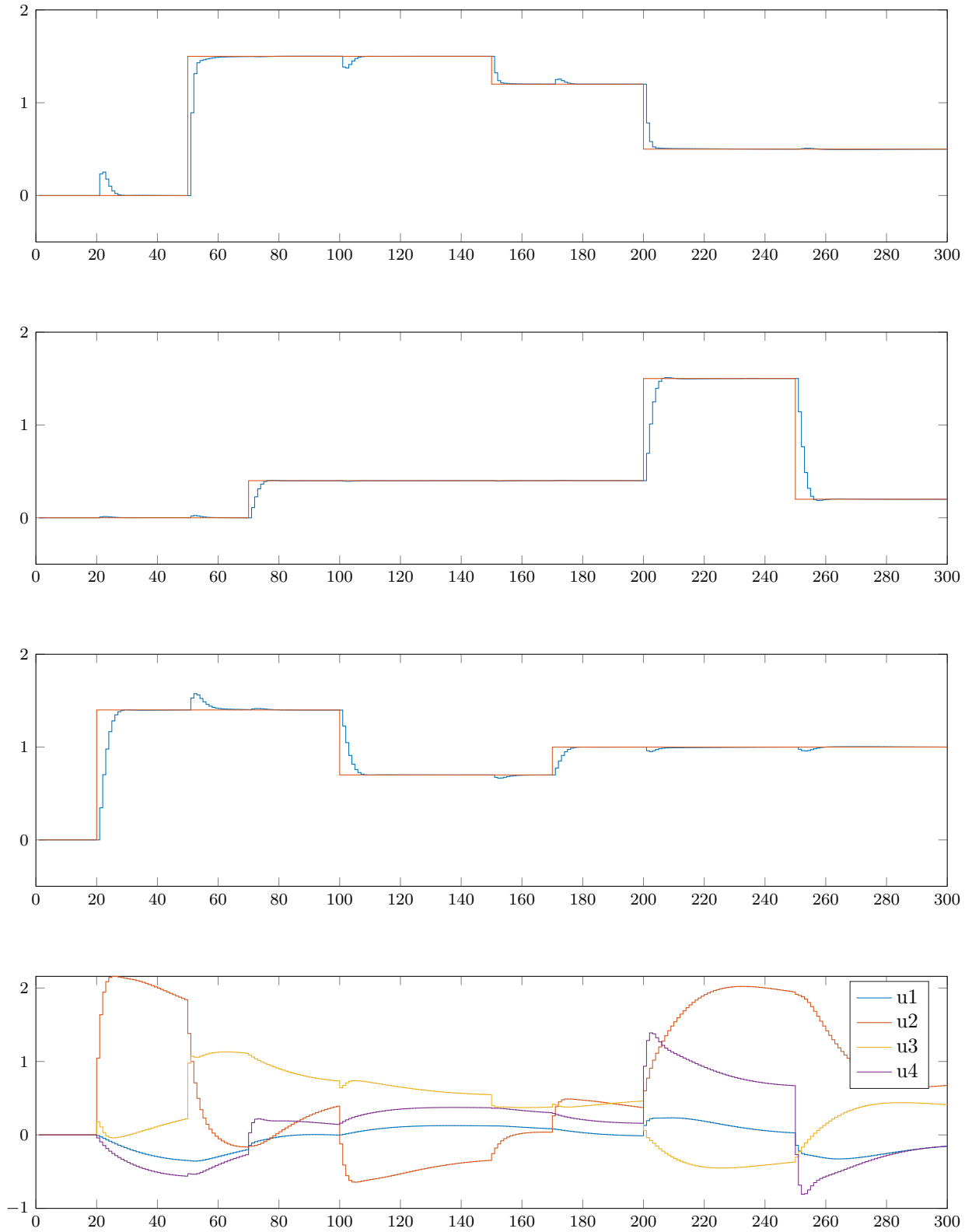
$$E = 8,43$$

$$(4.9)$$

4.3. DMC oszczędny - wyniki eksperymentów

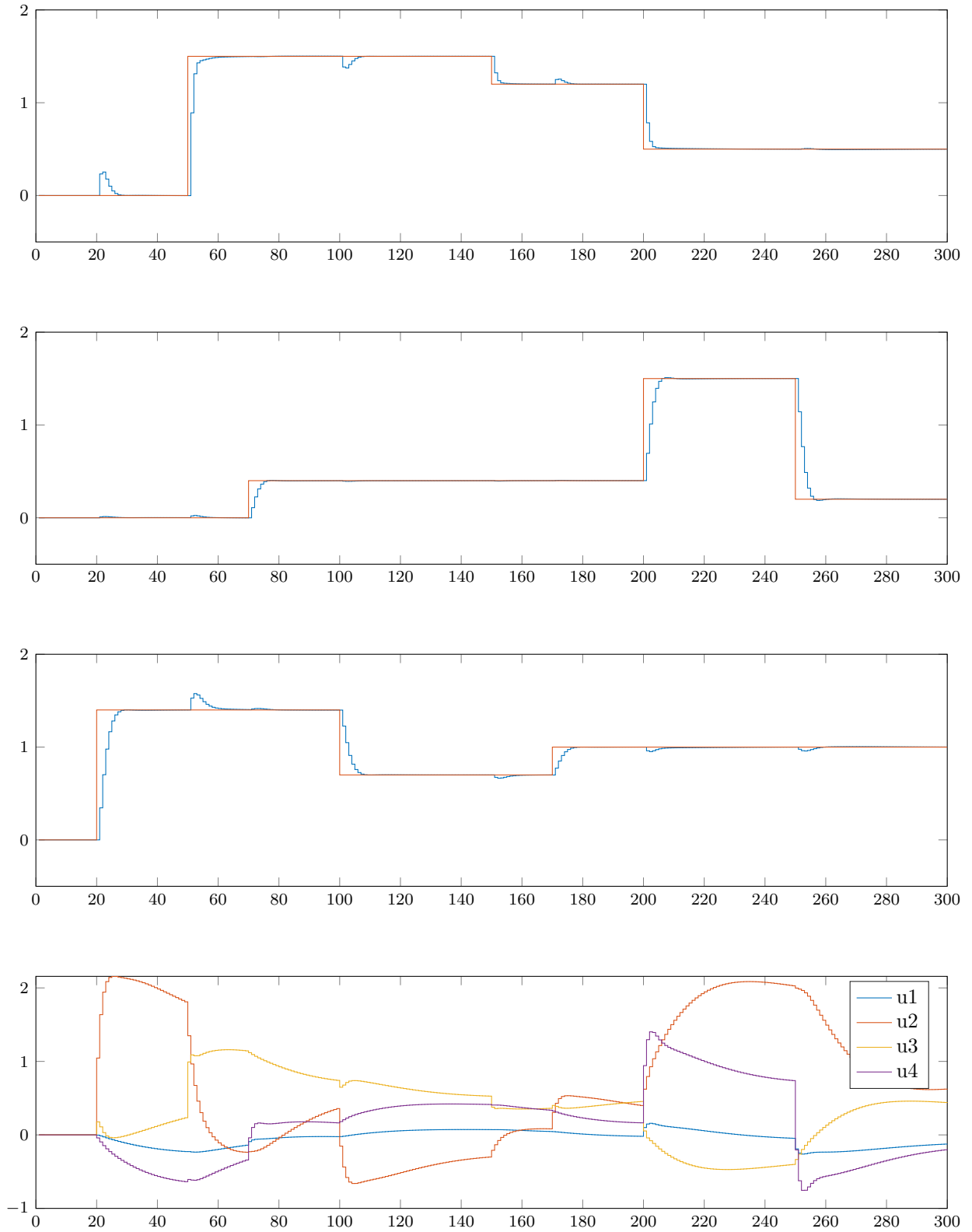
Rys. 4.10. DMC oszczędny. $N = 150$; $Nu = 50$; $\mu = [111]$; $\lambda = [1111]$.

$$E = 14,071 \quad (4.10)$$

Rys. 4.11. DMC oszczędny. $N = 100$; $Nu = 50$; $\mu = [111]$; $\lambda = [1111]$.

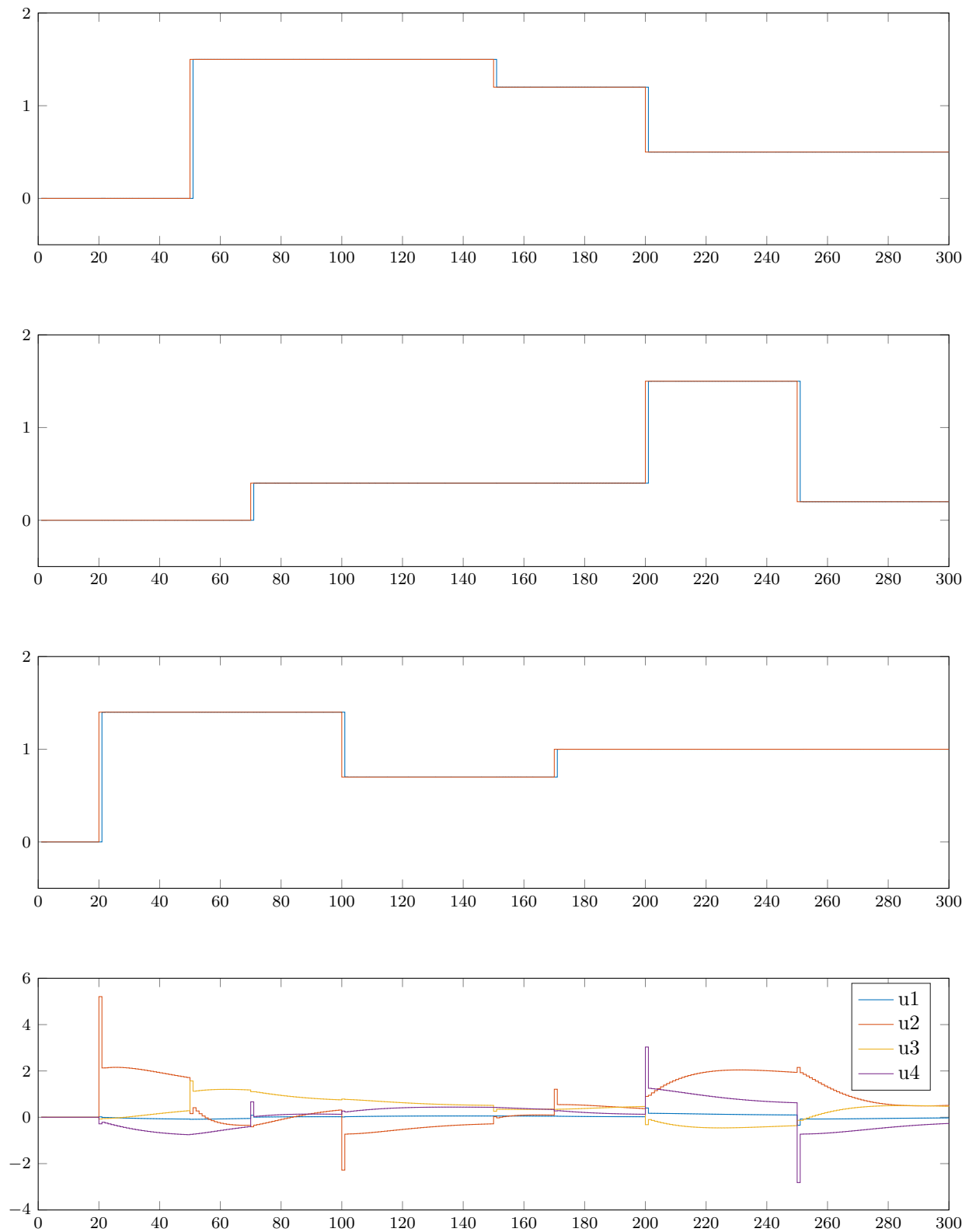
$$E = 14,0725$$

$$(4.11)$$

Rys. 4.12. DMC oszczędny. $N = 10$; $Nu = 10$; $\mu = [111]$; $\lambda = [1111]$.

$$E = 14,0854$$

(4.12)

Rys. 4.13. DMC oszczędny. $N = 1$; $Nu = 1$; $\mu = [101010]$; $\lambda = [0, 0010, 0010, 0010, 001]$.

$$E = 8,43 \quad (4.13)$$

4.4. Regulacja DMC - komentarz ogólny

Jak widać wyniki są nie do porównania z regulatorami PID. Regulacja jest szybsza i lepszej jakości, przy czym zmniejszając horyzonty do najmniejszych możliwych wartości oraz odpowiednio zmniejszając λ (pozwalając na gwałtowne skoki sterowania) i zwiększając ψ (zwiększając moc regulacji) udało się uzyskać idealnej matematycznie regulacji, w której wyjścia zmieniają się w każdej chwili i idealnie śledzą wartość zadaną. Tutaj warto zauważyć że było to możliwe dla tego że regulacja była bez ograniczeń, oraz dla tego że obiekt jest pewnym matematycznym modelem, nie mający odpowiedników w rzeczywistości. Regulacja klasycznego regulatora DMC nie różniła się od regulacji DMC oszczędnego. Więc w obu tych przypadkach wartość błędu optymalna jest równa:

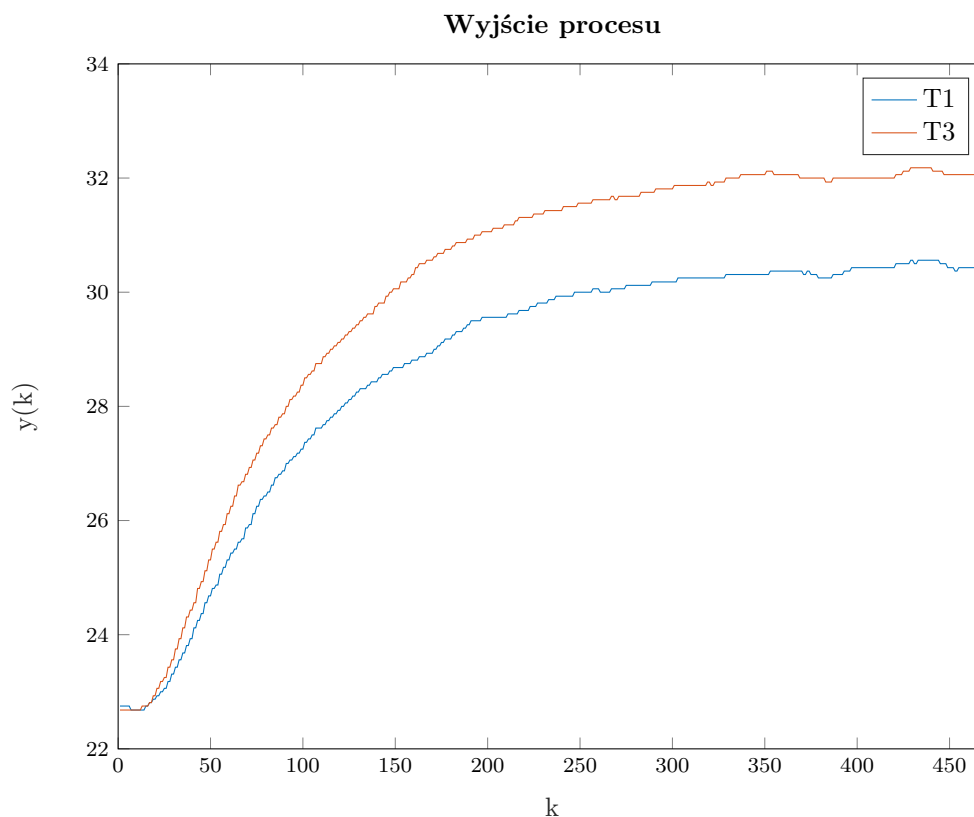
$$E = 8,43 \quad (4.14)$$

5. Laboratorium

5.1. Określenie wartości pomiaru temperatury w punkcie pracy

W celu określenia wartości pomiaru temperatury w punkcie pracy, ustawiono moc wentylatorów $W1 = W2 = 50\%$ oraz moc grzałek $G1 = 25\%$ $G2 = 30\%$. Po pewnym czasie temperatury, odczytywane przez czujniki zaczęły się stabilizować na poziomie: $T1 = 29,81^{\circ}C$ oraz $T3 = 31,43^{\circ}C$.

Niestety z powodu ciągłego ruchu powietrza związanego z przemieszczaniem się osób w sali i dużej ilości tych osób wpływających na temperaturę sali oraz czułość stanowiska pomiarowego temperatury odczytywane przez czujnik zaczęły lekko oscylować wokół temperatur w punkcie pracy.



Rys. 5.1. Pomiar temperatury w punkcie pracy

5.2. Mechanizm zabezpieczający przed uszkodzeniem stanowiska

Zadaniem tego mechanizmu jest wyłączanie grzałki w moncie, gdy czujnik przy tej grzałce przekroczy temperaturę $250^{\circ}C$. W mechanizmie w tym uwzględniono również ograniczenia sygnału sterującego tzn. moc grzałki może przyjmować wartości z zakresu 0-100% (w programie odpowiada to wartościom 0-1000). W programie wartości temperatur odpowiadają 100-krotnie większej wartości niż w rzeczywistości, dlatego też temperatura $250^{\circ}C$ zapisana jest jako 25000.

Implementacja przedstawia się w następujący sposób:

Listing 5.1. Mechanizm zabezpieczający

```
//mechanizm zabezpieczający
IF D100 >= 25000 THEN // T1>250
    D114 := 0; //G1=0%
END_IF;

IF D102 >= 25000 THEN // T3>250
    D115 := 0; //G2=0%
END_IF;

// zabezpieczenie sygnału sterującego
IF(D114>=1000) THEN //G1>100%
    D114 := 1000; //G1=100%
    ELSIF(D114<=0) THEN //G1<0%
        D114:=0; //G1=0%
    END_IF;

IF(D115>=1000) THEN //G2>100%
    D115 := 1000; //G2=100%
    ELSIF(D115<=0) THEN //G2<0%
        D115:=0; //G2=0%
    END_IF;
```

5.3. Regulator PID

W zadaniu tym użyto dwóch regulatorów PID (jeden ustala wartość sterowania grzałką G1, biorąc pod uwagę sygnał wyjściowy T1, natomiast drugi wyznacza G2, a bierze pod uwagę wartość T3). Dla każdego z regulatorów wartość sterowania jest liczona z wzoru:

$$u(k) = r_2 e(k-2) + r_1 e(k-1) + r_0 e(k) + u(k-1) \quad (5.1)$$

Współczynniki regulacji r_i są wyliczane ze standardowych wzorów.

Listing 5.2. Wyznaczenie współczynników regulacji

```
//Współczynniki K, Ti i Td dla regulatorow
K_1 := 6.0;
K_2 := 6.0;

TI_1 := 100.0;
TI_2 := 110.0;

TD_1 := 0.0;
TD_2 := 0.0;

TProb := 4.0; //czas probkowania

//Wartosci uchybow w poprzednich chwilach probkowania
E0_1 := 0;
E1_1 := 0;
E2_1 := 0;
E0_2 := 0;
E1_2 := 0;
E2_2 := 0;
```

```

//wyliczenie wspolczynnikow ri dla PID1
R0_1 := K_1 * (1 + (TProb/(2*TI_1))+TD_1/TProb);
R1_1 := K_1 * (TProb/(2*TI_1) - 2*TD_1/TProb - 1);
R2_1 := K_1 * TD_1 / TProb;

//wyliczenie wspolczynnikow ri dla PID2
R0_2 := K_2 * (1 + (TProb/(2*TI_2))+TD_2/TProb);
R1_2 := K_2 * (TProb/(2*TI_2) - 2*TD_2/TProb - 1);
R2_2 := K_2 * TD_2 / TProb;

```

Listing 5.3. Implementacja algorytmu PID

```

T1_act := D100; //pobrane aktualnej wartosci T1
T3_act := D102; //pobrane aktualnej wartosci T3

// PID 1
E2_1 := E1_1; //przesuniecie wartosci uchybow w chwili k-2, k-1
E1_1 := E0_1;
E0_1 := y_zad1 - T1_act; //wyliczenie aktualnego uchybu

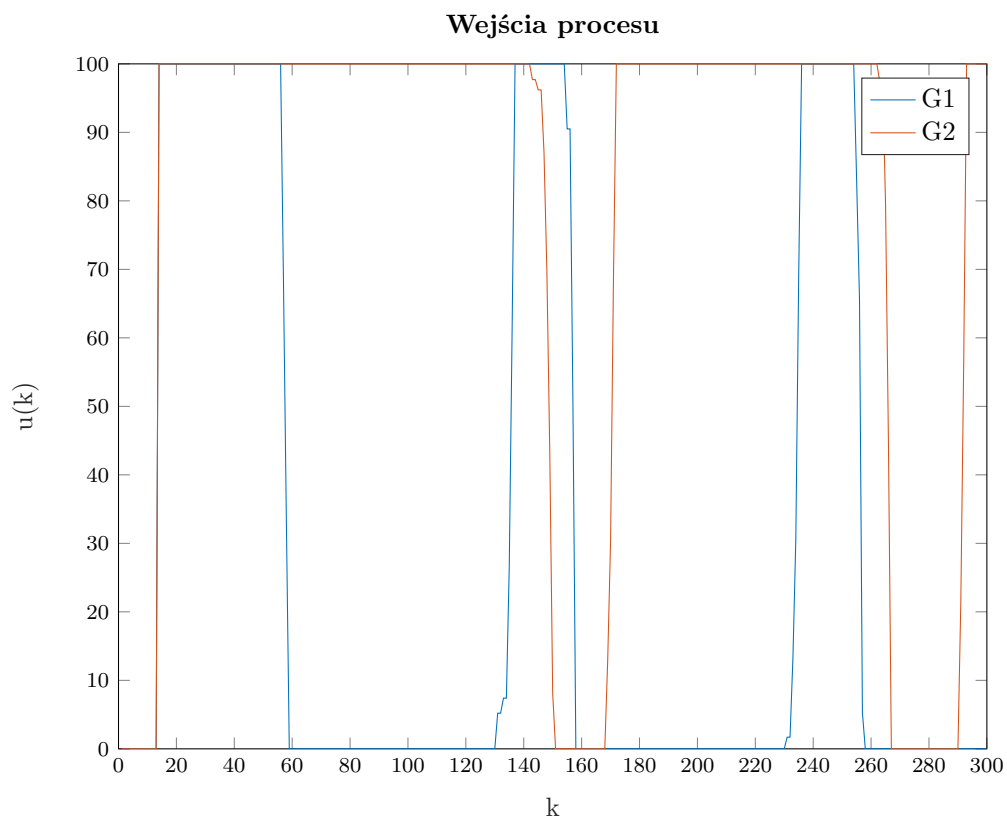
//wyliczenie nowego sterowania zgodnie ze wzorem
U_1 := R2_1*E2_1 + R1_1*E1_1 + R0_1*E0_1 + U_1;
D114 := REAL_TO_INT(U_1); //zadanie nowej wartosci sterowania

// PID 2
E2_2 := E1_2;
E1_2 := E0_2;
E0_2 := y_zad2 - T3_act;

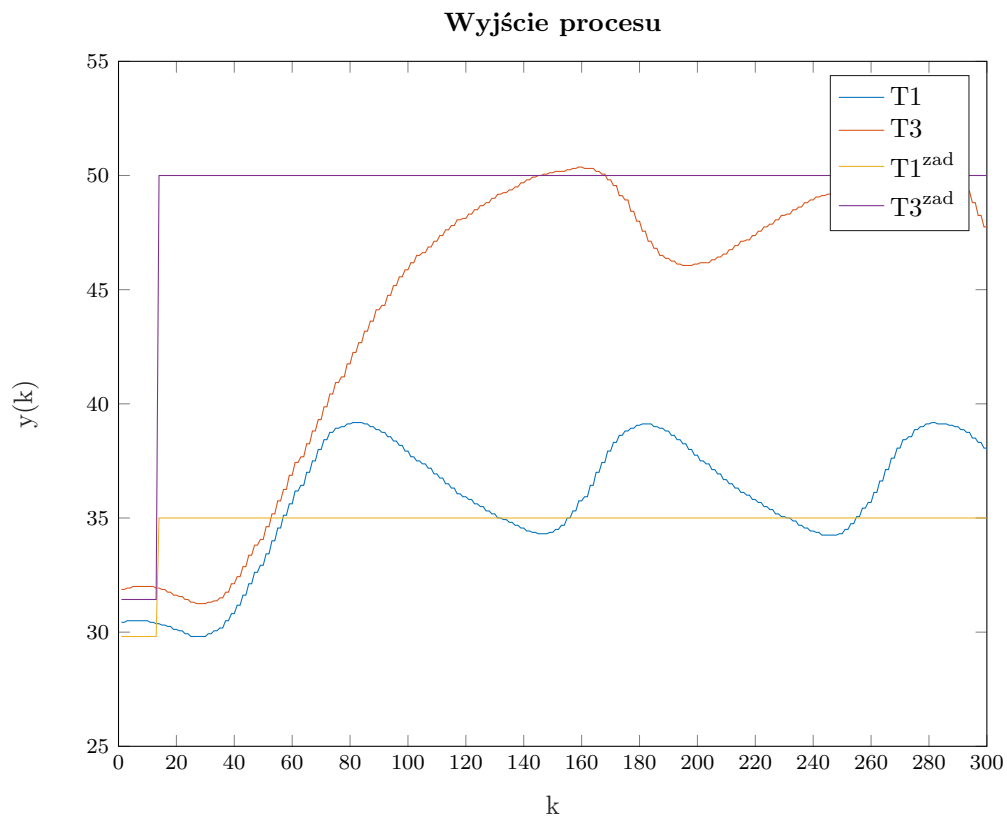
//wyliczenie nowego sterowania zgodnie ze wzorem
U_2 := R2_2*E2_2 + R1_2*E1_2 + R0_2*E0_2 + U_2;
D115 := REAL_TO_INT(U_2); //zadanie nowej wartosci sterowania

```

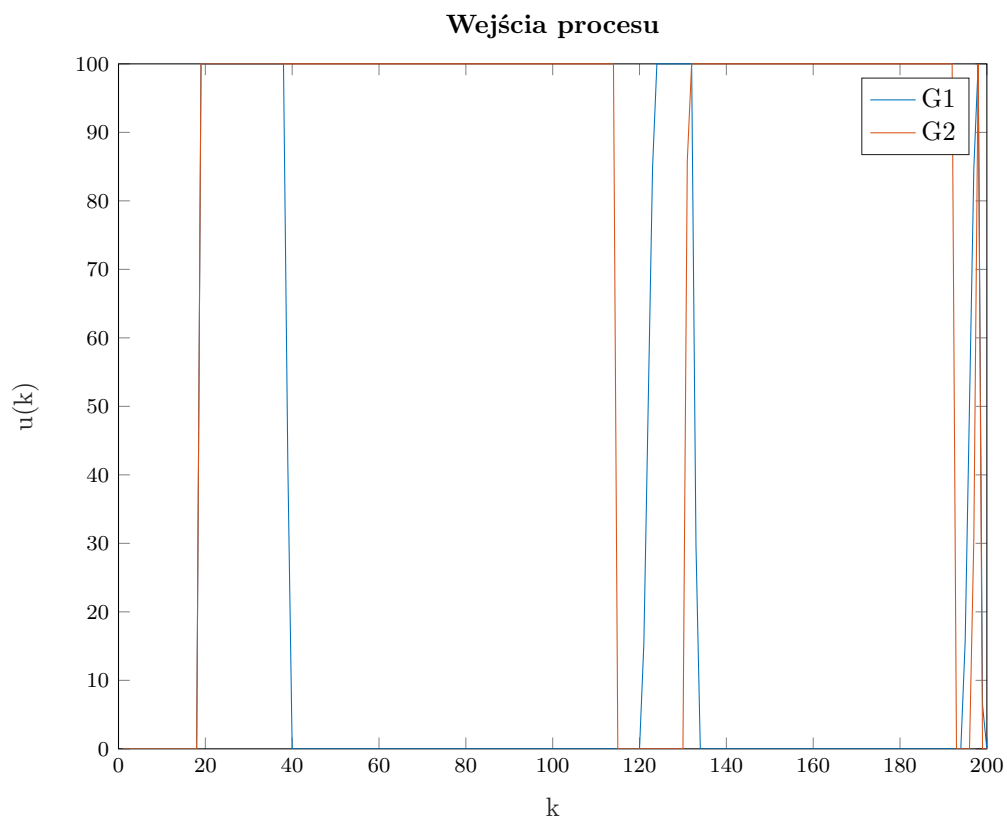
Po każdym wyliczeniu nowej wartości sterowania, wykonywany jest mechanizm zabezpieczający z punktu drugiego.



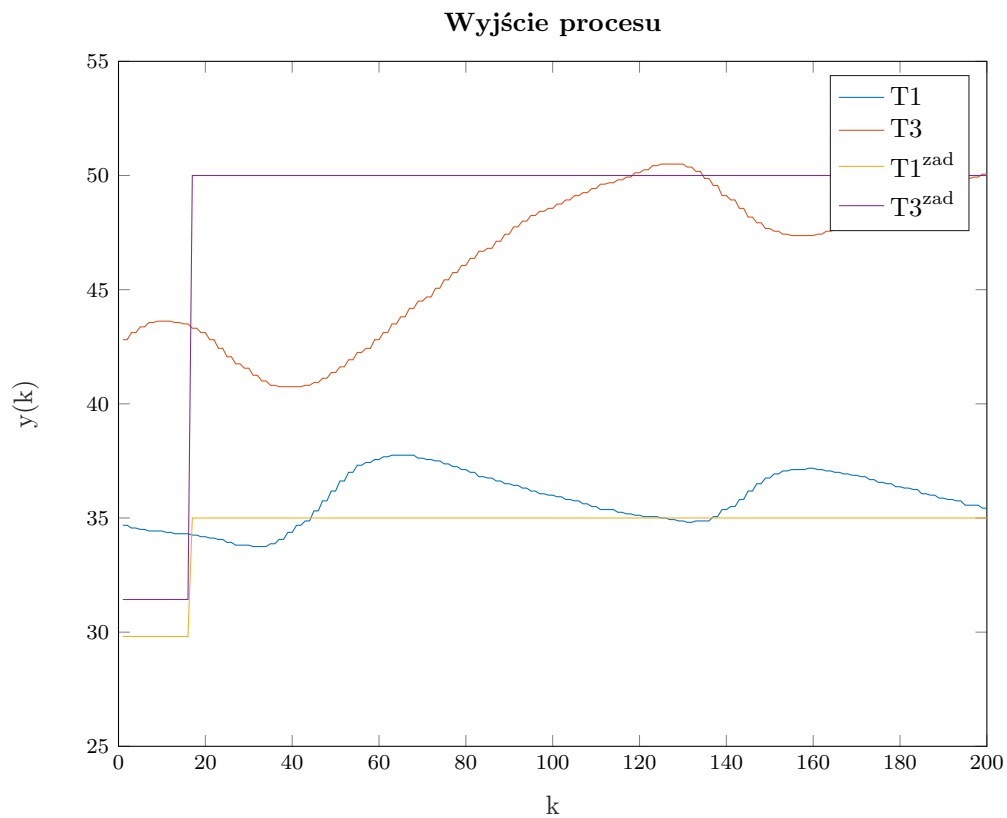
Rys. 5.2. Sygnały sterujące: PID1: $K=3$ $T_i=100$ $T_d=1.3$ PID2: $K=2$ $T_i=90$ $T_d=1$



Rys. 5.3. Wyjścia obiektu: PID1: $K=3$ $T_i=100$ $T_d=1.3$ PID2: $K=2$ $T_i=90$ $T_d=1$



Rys. 5.4. Sygnały sterujące: PID1: $K=1$ $T_i=10$ $T_d=3$ PID2: $K=3$ $T_i=9$ $T_d=2$



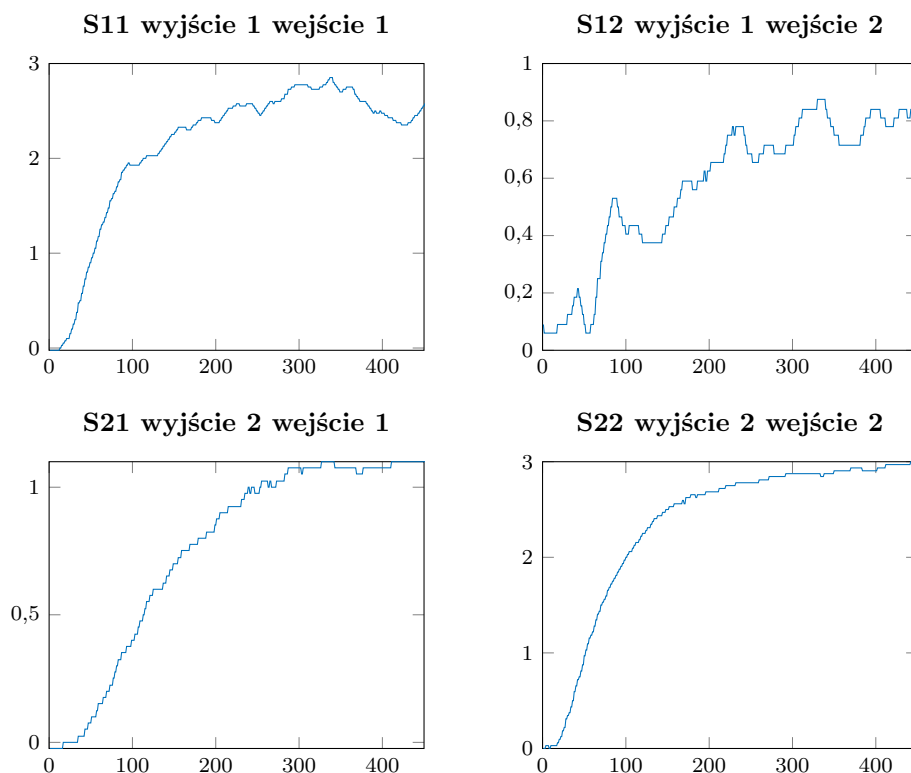
Rys. 5.5. Wyjścia obiektu: PID1: $K=1$ $T_i=10$ $T_d=3$ PID2: $K=3$ $T_i=9$ $T_d=2$

Żaden z powyższych zestawów regulatorów nie działa w sposób zadowalający. Sygnały wyjściowe zbliżają się do wartości zadanej, jednakże wartość ta ciągle oscyluje (oscylacje są dość duże). Rozwiązaniem tego problemu może być próba dostrojenia regulatorów.

5.4. Regulator DMC

Implementacja regulatora DMC w wersji 2x2 zaczęła się od wyznaczenia współczynników s . Do tego celu należy zebrać przebiegi wyjść na skok sygnału sterującego (kolejno skok na wejściu pierwszym G1 przy stałym G2 oraz skok na wejściu G2 przy stałym G1). Następnie dla każdego z przebiegów należy zastosować wzór:

$$S_i = \frac{Y(i) - Y_{pp}}{\Delta U}, \text{ dla } i = 1, 2 \dots D \quad (5.2)$$



Rys. 5.6. Odpowiedzi skokowe

Implementacja samego algorytmu regulacji DMC została wykonana w wersji oszczędnej, składającej się z dwóch części: 1. Wyliczenie K_e i K_u W Matlab:

```
clear all;
close all;
load('wspolczynniki.mat')
czas_symulacji = 100;
nu=2; %liczba wejsci
ny=2; %liczba wyjsc

%Macierz odpowiedzi skokowych , kazdy wspolczynnik si to macierz 2x2
for i=1:length(s11)
    S(i)=[s11(i) s12(i);...
```

```

        s21(i) s22(i)]];
end

%Parametry dobrane eksperymentalnie
D=480; N=15; Nu=10;
lambda1=1; lambda2=1;
psi1=1; psi2=1;

%Punkty pracy W1=W2=50 (500) G1=25 (250) G2=30 (300) T1= 29,81 T3=31,43
u1pp=250;
u2pp=300;

y1pp=2981;
y2pp=3143;

%Macierz Mp
for i=1:(D-1)
    for j=1:N
        if j+i>D
            Mp{j,i}=S{D}-S{i};
        else
            Mp{j,i}=S{j+i}-S{i};
        end
    end
end

%Macierz M
for i=1:Nu
    M(i:N,i)=S(1:N-i+1);
    M(1:i-1,i)=[0 0; 0 0];
end

%Macierz psi
for i=1:N
    for j=1:N
        if i == j
            Psi{i,j} =[psi1 0; 0 psi2];
        else
            Psi{i,j} =[0 0; 0 0];
        end
    end
end

%Macierz Lambd
for i=1:Nu
    for j=1:Nu

```

```

        if i == j
            Lambda{i,j} =[lambda1 0; 0 lambda2];
        else
            Lambda{i,j} =[0 0; 0 0];
        end

    end
end

%zamiana na macierze
Lambda_m = cell2mat(Lambda);
Mp_m = cell2mat(Mp);
M_m = cell2mat(M);
Psi_m = cell2mat(Psi);

%wyliczenie macierzy K
K=(M_m'*Psi_m*M_m+Lambda_m)^(-1)*M_m'*Psi_m;

%obliczenie Ku
K1 = K(1:ny,:);
Ku = K1*Mp_m;

%zapis do pliku tekstowego, tak aby latwo przekopiowac do GXWorks3
fileID = fopen('wiersz1.txt','w');
% petla do zapisu Ku1
for i=1:length(K1)
    fprintf( fileID, 'KU_1[%d] := %d;\n', i-1, Ku(1,i));
end
fclose(fileID);

fileID = fopen('wiersz2.txt','w');
% petla do zapisu Ku2
for i=1:length(K1)
    fprintf( fileID, 'KU_2[%d] := %d;\n', i-1, Ku(2,i));
end
fclose(fileID);

%obliczenie Ke (macierz rozmiaru 2x2)
Ke(1,1)=sum(K1(1,1:2:N*ny));
Ke(1,2)=sum(K1(1,2:2:N*ny));
Ke(2,1)=sum(K1(2,1:2:N*ny));
Ke(2,2)=sum(K1(2,2:2:N*ny));

```

2. Wyznaczanie na bieżąco wartości sterowania:

```

KE_11 := -0.171922435496501;
KE_21 := 0.679020850746672;
KE_12 := -0.318796410782815;
KE_22 := 0.202188351028979;

//wyznaczenie przyrostow (rozpisane dzialania macierzowe)

```



```

dU1 := KE_11*(y_zad1-T1_act)+KE_12*(y_zad2-T3_act);
dU2 := KE_21*(y_zad1-T1_act)+KE_22*(y_zad2-T3_act);

FOR IND := 0 TO 29 BY 1 DO
    dU1 := dU1 - KU_1[IND] *DUP[IND];
    dU2 := dU2 - KU_2[IND] *DUP[IND];
END_FOR;

// przesunięcie wartości w wektorze dup
FOR IND := 30 TO 2 BY -1 DO
    DUP[IND] := DUP[IND-2];
END_FOR;
DUP[0] := dU1;
DUP[1] := dU2;

U_1 := U_1 +dU1; //nowa wartość to stara plus przyrost
U_2 := U_2 +dU2; //nowa wartość to stara plus przyrost

D114 := REAL_TO_INT(U_1); //zadanie nowej wartości sterowania
D115 := REAL_TO_INT(U_2); //zadanie nowej wartości sterowania

//mechanizm zabezpieczający

```

Wartości K_e i K_u są przekopiowane z Matlaba. K_u jest przeniesione jako 2 wektory danych (KU_1 , KU_2).

Tutaj tak samo jak w implementacji regulatora PID, na sam koniec jest wykonywany mechanizm zabezpieczający. Niestety powyższego algorytmu nie udało się przetestować w realnych warunkach na stanowisku laboratoryjnym.

5.5. Panel operatora

Na interfejsie użytkownika po prawej stronie można zauważyć słupki, które obrazują wartości grzałek G1, G2 (od 0 do 100%). Po prawej stronie pod tytułem widnieją wartości zmierzonych temperatur (T_1 oraz T_3), pod nimi są wartości zadane tych wyjść. Na samym dole są 3 przyciski zmieniające stan automatu (a z boku widnieje napis mówiący, w którym stanie aktualnie się znajduje) opisanego w następnym podrozdziale.

5.6. Automat stanów

Listing 5.4. Implementacja automatu stanów

```

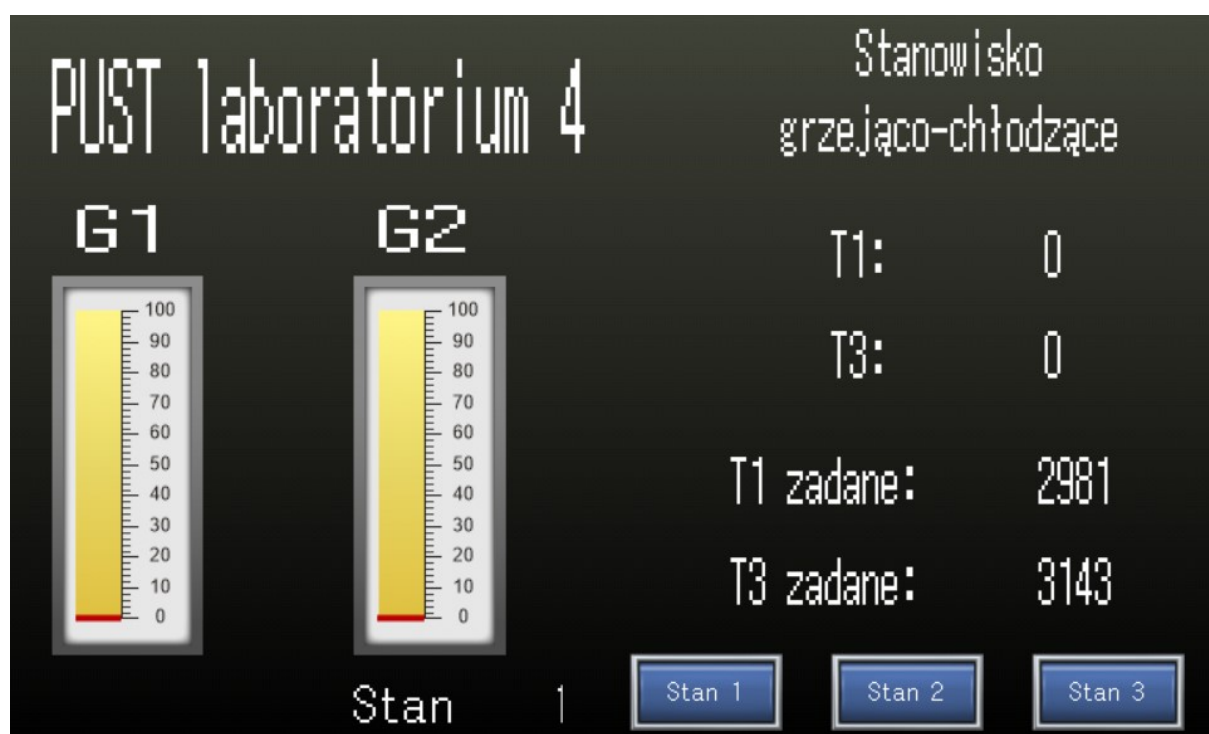
CASE Stan1 OF
    //M11 przycisk do zmiany na stan 1
    //M12 przycisk do zmiany na stan 2
    //M13 przycisk do zmiany na stan 3
    1:
        y_zad1 := 2981; //wartości zadane dla stanu 1
        y_zad2 := 3143;
        IF M12 THEN
            Stan1 := 2;
        END_IF;
        IF M13 THEN
            Stan1 := 3;
        END_IF;
    2:

```

```
y_zad1 := 3500; //wartosci zadane dla stanu 2
y_zad2 := 5000;
IF M11 THEN
    Stan1 := 1;
END_IF;
IF M13 THEN
    Stan1 := 3;
END_IF;
3:
y_zad1 := 4000; //wartosci zadane dla stanu 3
y_zad2 := 3500;
IF M12 THEN
    Stan1 := 2;
END_IF;
IF M11 THEN
    Stan1 := 1;
END_IF;
END_CASE;
```

Zmiana stanu automatu odbywa się po wciśnięciu przycisków (widocznych na panelu operatora). Każdy z tych przycisków posiada nazwę, która informuje na jaki stan zmienia się obecny stan. W każdym stanie zdefiniowane są inne wartości temperatur zadanych:

1. W stanie pierwszym: $T1^{zad} = 29,81^{\circ}C$ $T3^{zad} = 31,43^{\circ}C$
2. W stanie drugim: $T1^{zad} = 35^{\circ}C$ $T3^{zad} = 50^{\circ}C$
2. W stanie trzecim: $T1^{zad} = 40^{\circ}C$ $T3^{zad} = 35^{\circ}C$



Rys. 5.7. Wygląd interfejsu użytkownika

6. Laboratorium: stanowisko TRAS

6.1. Zabezpieczenie przed uszkodzeniem stanowiska

Założenia zabezpieczenia stanowiska występują na dwóch obszarach:

1. Zabezpieczenie wartości wypełnienia PWM- może ona przyjmować wartości od 0-1000
2. Ograniczenia wychylenia elementów wykonawczych- tzn wartości kątów obrotu enkoderów przyjmują ograniczone wartości (w poziomie od -180 do 180, a w pionie od -70 do 70), po przekroczeniu wartości brzegowych wypełnienie odpowiedniego sygnału PWM jest zmniejszane do 1.

6.2. Wyznaczenie charakterystyki statycznej

Charakterystykę statyczną można wyznaczyć poprzez zadawanie różnego wypełnienia sygnału PWM (sygnał sterujący) dla silników Pt oraz Az. Następnie należy stworzyć wykresy ilustrujące, jakie pozycje enkoderów uzyskano przy danych wartościach sterujących.

6.3. Regulator PID

Implementacja regulatora wygląda analogicznie do implementacji regulatora PID dla stanowiska z pierwszej części laboratorium. Znając parametry K , T_i , T_d oraz czas próbkowania, należy wyznaczyć wartości współczynników r_0 , r_1 i r_2 i podstawić do wzoru na nową wartość sterowania.

Te zadania należy wykonać dla regulatora pracującego dla położenia pionowego, jak i dla położenia poziomego. Poniżej zamieszczony jest kod dla jednego z regulatorów:

```
//SV zadana wartosc
//PV obecna wartosc
//MV sterowanie

//ustawienie wartosci mierzonej (kat na enkoderze)
wlasny_PID_Az.PV := Enc_Az;

//przesunięcia wartosci uchybow w chwili k-2, k-1
wlasny_PID_Az.Ep2 := vlasny_PID_Az.Ep1;
wlasny_PID_Az.Ep1 := vlasny_PID_Az.Ep0;
//wyliczenie aktualnego uchybu
wlasny_PID_Az.Ep0 := vlasny_PID_Az.SV - vlasny_PID_Az.PV; =
//wyliczenie nowego sterowania zgodnie ze wzorem
wlasny_PID_Az.MV:=wlasny_PID_Az.Rp2*wlasny_PID_Az.Ep2+wlasny_PID_Az.MV;
wlasny_PID_Az.MV:=wlasny_PID_Az.MV+wlasny_PID_Az.Rp1*wlasny_PID_Az.Ep1;
wlasny_PID_Az.MV:=wlasny_PID_Az.MV+wlasny_PID_Az.Rp0*wlasny_PID_Az.Ep0;

//zabezpieczenie sygnału sterującego
IF vlasny_PID_Az.MV >1000 THEN
    vlasny_PID_Az.MV := 1000;
END_IF;
IF vlasny_PID_Az.MV <-1000 THEN
```

```

        wlasny_PID_Az.MV := 0;
END_IF;

//Ustawienie nowych wartosci sterowania
IF wlasny_PID_Az.MV < 0 THEN
    Dir1_Az := 1;
    PWM_Az := ABS(REAL_TO_INT(wlasny_PID_Az.MV));
ELSE
    Dir1_Az := 0;
    PWM_Az := REAL_TO_INT(wlasny_PID_Az.MV);
END_IF;

```

Przy ustawieniu nowych wartości należy pamiętać, że silnik porusza się w dwie strony. Dzięki tej własności można otrzymywać wartości kąta enkodera (wartość ta wyznaczana jest za pomocą wejścia licznikowego) zarówno ujemne jak i dodatnie, dlatego też może się zdarzyć sytuacja, że wyliczona wartość nowego sterowania będzie ujemna. W takim przypadku jest ustawiane wyjście zmiany kierunku silnika, a wypełnienie sygnału PWM jest wartością bezwzględną wyliczonej wartości sterowania. Sygnał PWM ustawiony jest w taki sposób, że wypełnienie może przyjmować wartości od 0 do 1000.

Procedura doboru nastawów regulatorów polegałaby na wprowadzeniu układu w stan oscylacji krytycznych, a następnie wyznaczenie parametrów zgodnie z metodą Zieglera-Nicholsa. Gdyby te parametry nie przynosiłyby zadowalających rezultatów, należałoby dostroić regulator metodą inżynierską.

6.4. Regulator PID z użyciem gotowej funkcji

Użycie gotowej funkcji regulatora PID dla jednego z silników przedstawiono poniżej:

```

PID_Az.PV := Enc_Az; //ustawienie wartosci mierzonej (kata na enkoderze)
//funkcja wbudowana sterownika PLC
PID( PID_Az.Control_ON, PID_Az.SV, PID_Az.PV, PID_Az.params[0], PID_Az.MV );
IF PID_Az.MV < 0 THEN
    Dir1_Az := 1;
    PWM_Az := ABS(PID_Az.MV); //ustawienie nowego sterowania
ELSE
    Dir1_Az := 0;
    PWM_Az := PID_Az.MV; //ustawienie nowego sterowania
END_IF;

```

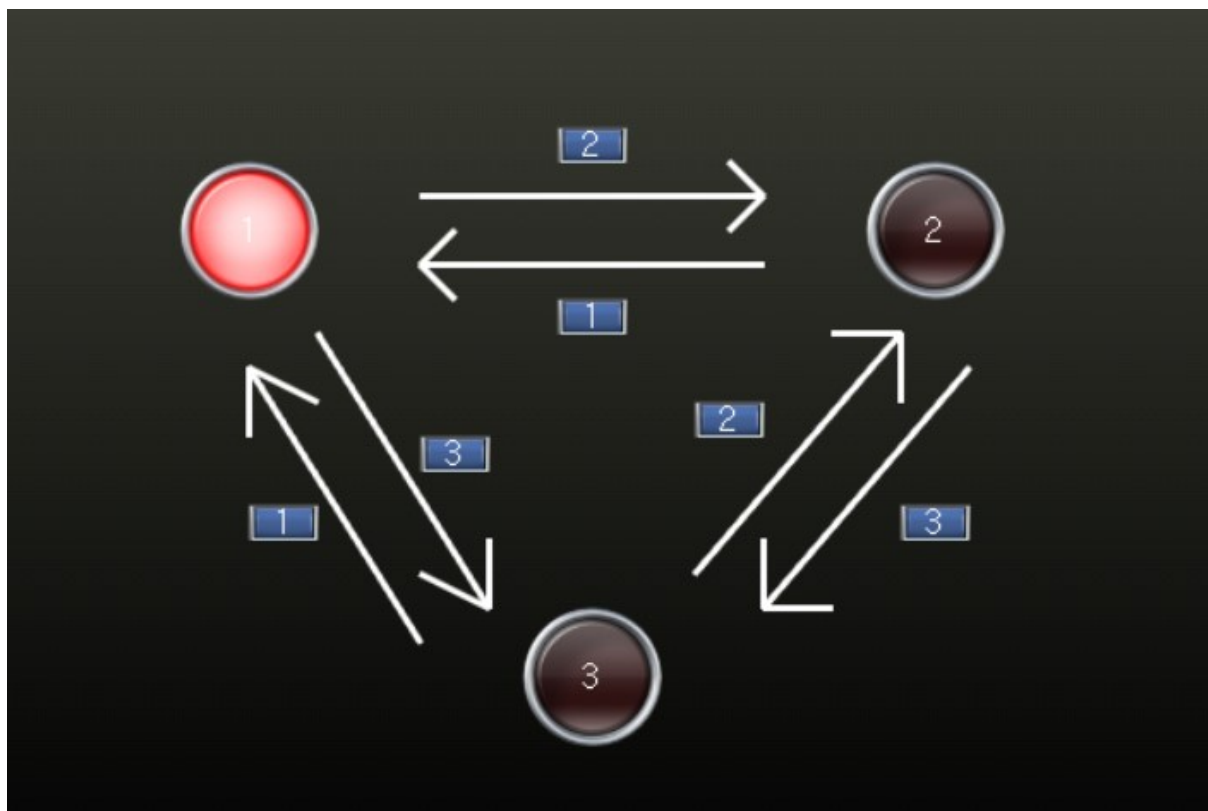
Oczywiście (tak jak w przypadku regulatora PID, który był własnoręcznie implementowany) najpierw należy zainicjalizować potrzebne parametry regulatora do których wchodzi m.in. K , T_i , T_d , czas próbkowania, ograniczenia. Proces strojenia jest analogiczny do procesu opisanego w poprzednim podpunkcie.

6.5. Automat stanów

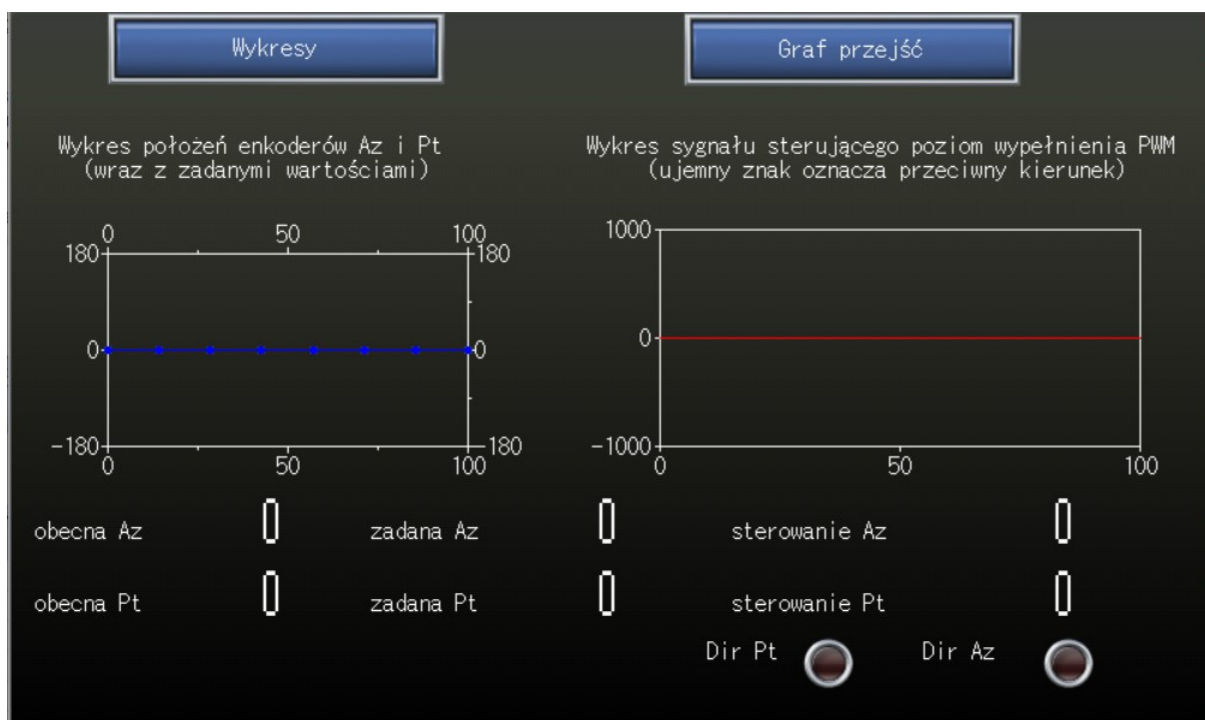
Idea automatu stanów jest analogiczna do w przypadku z części pierwszej laboratorium. Istnieje zmienna przechowująca numer stanu obecnego, a w każdym stanie określona są inne wartości zadane dla regulatorów z implementacją własną i użyciem gotowych funkcji. Przełączanie między stanami odbywa się po wciśnięciu przycisku z nazwą stanu, do którego automat ma się przełączyć.

6.6. Wizualizacja procesu

Wizualizacja składa się z wykresów: sygnałów zadanych i zmierzonych (po lewej stronie) oraz wartości sterowania (po prawej stronie). Oba te wykresy są odpowiednio podpisane. Dodatkowo na dole ekranu wypisywane są wartości aktualne. W prawym dolnym rogu można zauważyć dwie diody, które informują o zmianie kierunku ruchu silnika.



Rys. 6.1. Wygląd grafu stanów dla interfejsu użytkownika



Rys. 6.2. Wygląd interfejsu użytkownika (wizualizacja)