Wykorzystanie komunikacji szeregowej w aparaturze kontrolno - pomiarowej

Celem ćwiczenia jest opanowanie umiejętności komunikowania się z urządzeniem pomiarowym poprzez port szeregowy RS-232 komputera w środowisku LabView. Komunikacja ta ma zostać wykorzystana do stworzenia aplikacji sterującej procesem wzrostu temperatury elementu grzejącego (moduł Peltiera) oraz stabilizacji tej temperatury przy zadanej jej wartości.

1. Podstawowe informacje o komunikacji szeregowej i standardzie VISA

Komunikacja szeregowa (*serial communication*) jest popularnym sposobem przesyłania danych (bit po bicie) pomiędzy dwoma urządzeniami (np. komputerem i instrumentem pomiarowym). Ponieważ w danej chwili przesyłany jest tylko jeden bit poprzez pojedynczą linię sygnałową łączącą nadajnik z odbiornikiem, komunikację szeregową stosuje się do przesyłania niewielkich ilości danych, lub w przypadku przesyłania danych na duże odległości. Przesłanie każdego znaku odbywa się jako zestaw, który zawiera kolejno: bit startowy (dodatnie napięcie), właściwe bity kodujące znak, opcjonalny bit parzystości oraz bity stopu (napięcie ujemne, zazwyczaj 1, 1.5 lub 2 bity). Maksymalna liczba znaków, jaka może być przesłana w ciągu sekundy, jest równa szybkości transmisji (*baud rate*, liczba bodów: bitów na sekundę) podzielonej przez liczbę bitów w pojedynczym zestawie. Najbardziej popularnym standardem w komunikacji szeregowej jest standard RS-232 z 9 stykowym złączem. Poza trzema podstawowymi liniami przesyłowymi (dane nadawane, dane odbierane i masa) zawiera on także inne linie (żądanie nadawania (RTS), gotowość nadawania (CTS), gotowość terminalu (DTR), gotowość urządzenia transmisyjnego (DSR)), które umożliwiają transmisję asynchroniczną, wymagającą wymiany sygnałów sterujących (w odróżnieniu od rzadziej stosowanej transmisji synchronicznej, w której kolejne bity są przesyłane zgodnie z impulsami zegara nadajnika).

Każdy z protokołów komunikacji wykorzystywanych w naukowych i inżynieryjnych zastosowaniach (szeregowy, GPIB, PXI, Ethernet, VXI i inne) różni się znacząco od pozostałych, co powoduje wzajemną niekompatybilność. Aby temu zaradzić stworzono standard nazwany VISA (*Virtual Instrument Software Architecture*), umożliwiający

jednakowy z punktu widzenia programisty sposób komunikowania się z urządzeniami poprzez poszczególne protokoły i upraszczający przez to sposób pisania programów dostosowanych do różnych urządzeń. Funkcje VISA kontrolują urządzenia podłączone za pomocą różnych protokołów komunikacyjnych poprzez odpowiednie odwołania uwzględniające rodzaj protokołu, czego nie musi już wykonywać osoba programująca określone urządzenie.

Do obsługi komunikacji szeregowej w środowisku LabView wystarczą 4 funkcje, dostępne z palety **Functions > Instrument I/O > Serial**:

,

które służą odpowiednio do: konfiguracji sposobu transmisji szeregowej (**VISA Configure Serial Port**), wysyłania polecenia do urządzenia poprzez port szeregowy (**VISA Write**), odczytania informacji przesłanej przez urządzenie (**VISA Read**) oraz zamknięcia sesji komunikacyjnej (**VISA Close**). Wszystkie wymagają podania, jako jeden z parametrów wejściowych, identyfikatora sesji VISA (*VISA resource name*), który dla portu szeregowego ma postać:

ASRL[numer portu szeregowego]::INSTR

2. Komunikacja z modułem ćwiczeniowym

Pierwszym etapem ćwiczenia jest napisanie elementarnej aplikacji wysyłającej i odbierającej dane poprzez port szeregowy oraz zapoznanie się ze sposobem komunikowania się z „Modułem ćwiczeniowym z interfejsem RS232". Moduł ten w dalszej części ćwiczenia posłuży jako urządzenie pomiarowe sterujące elementem grzejącym i odczytujące jego temperaturę. Posiada on również możliwość zapalania diod kontrolnych, odczytywania wartości napięcia, częstotliwości, stanu wejścia TTL oraz ustawiania stanu wyjścia TTL, które można przetestować po podłączeniu zasilacza, generatora lub miernika do odpowiednich zacisków na przednim panelu modułu. Napisana aplikacja przydatna będzie również w dalszej części ćwiczenia do awaryjnej komunikacji z modułem, jeśli np. program sterujący temperaturą zawiesi się podczas nagrzewania, które trzeba będzie wyłączyć.

Przy ustalaniu konfiguracji komunikacji szeregowej (funkcja **VISA Configure Serial Port**, patrz powyżej) wystarczy podać jako parametry wejściowe tylko identyfikator sesji VISA (*VISA resource name*, najwygodniej poprzez polecenie *Create constant* przy tym złączu) oraz szybkość transmisji szeregowej 38400 (*baund rate*), gdyż pozostałe parametry

pracy modułu są takie same jak domyślne parametry funkcji konfiguracyjnej, czyli: 8 bitów danych (*data bits*), brak bitu parzystości (*parity*), 1 bit stopu (*stop bits*) i brak kontroli przepływu (*flow control*).

Pełny zestaw rozkazów, które można wysyłać do modułu ćwiczeniowego, jest następujący:

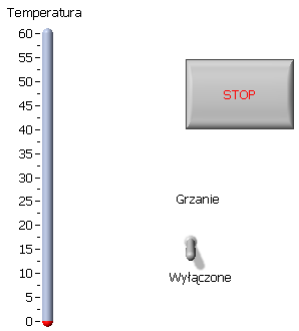
TTLIN : odczytuje stan z TTL WE
TTL1 : ustawia wysoki stan na TTL WY
TTL0 : ustawia niski stan na TTL WY
PELT1 : włącza zasilanie PELTIER
PELT0 : wyłącza zasilanie PELTIER
ACIN : odczytuje wartość napięcia z AC WE
ACIN1 : odczytuje wartość napięcia z AC WE i uśrednia
TEMP : odczytuje temperaturę
OMEG : odczytuje częstotliwość z COUNTER WE (1s)
OMEG1 : odczytuje częstotliwość z COUNTER WE (100ms)
LED1 : włącza/wyłącza LED1
LED2 : włącza/wyłącza LED2
LED3 : włącza/wyłącza LED3
LED4 : włącza/wyłącza LED4
DEMO : sekwencyjne miganie diod
HELP : wyświetla pomoc

Przy wysyłaniu rozkazu funkcją **VISA Write** należy pamiętać o dodaniu na końcu wysyłanego ciągu znaków (*write buffer*) znaku powrotu karetki (). Moduł ćwiczeniowy zwraca w postaci ciągu znakowego wartości odczytywanych wielkości (np. temperatury w stopniach Celsjusza) lub ciąg znaków „OK”, jeśli rozkaz nie polegał na odczycie (wszystkie ciągi zakończone są znakiem powrotu karetki). Odczytanie tych ciągów wykonuje się poprzez złącze *read buffer* funkcji **VISA Read**, podając liczbę znaków do odczytu (*byte count*). Należy zwrócić uwagę, aby liczba znaków do odczytu nie była mniejsza niż faktycznie wysłana przez moduł liczba znaków, gdyż spowoduje to błędny odczyt, a do następnego odczytu będą dodane znaki poprzednio nie odczytane. Z kolei zbyt duża liczba znaków do odczytu spowoduje niepotrzebne spowolnienie pracy. Odpowiednią liczbę znaków wysyłanych przez moduł po każdym z rozkazów można określić kontrolując wartość złącza *return count*, zwracającego liczbę faktycznie odczytanych znaków. Liczba ta zależy od

wysłanego rozkazu. Przed zakończeniem działania aplikacji należy zamknąć komunikację VISA funkcją **VISA Close**.

3. Pomiar temperatury i ręczne sterowanie elementem grzejącym

Następnym elementem ćwiczenia jest napisanie aplikacji, która będzie w sposób ciągły podawała temperaturę elementu grzejącego oraz umożliwiała jego włączanie i wyłączanie za pomocą przycisku na panelu frontowym LabView. Przykład takiego panelu frontowego znajduje się na rys.1.



Rys. 1. Przykład panelu frontowego aplikacji służącej do ręcznego sterowania elementem grzejącym

Najpierw należy podłączyć element grzejący, którym jest moduł Peltiera, do odpowiednich zacisków znajdujących się na tylnym panelu modułu ćwiczeniowego. Nagrzewanie się jednej powierzchni modułu Peltiera (i jednoczesne chłodzenie drugiej powierzchni) następuję dzięki zjawisku Peltiera. Jest to zjawisko termoelektryczne polegające na pochłanianiu ciepła na jednym ze spojeń dwóch różnych przewodników i wydzielaniu go na drugim spojeniu, gdy w obwodzie złożonym z tych przewodników płynie prąd stały. Ilość wydzielanego lub pochłoniętego ciepła jest proporcjonalna do natężenia prądu i temperatury bezwzględnej styku i zależy od rodzaju materiałów, z których jest zrobione spojenie. Zasilanie modułu Peltiera odbywa się z niezależnego zasilacza, który również należy podłączyć do odpowiednich zacisków znajdujących się na tylnym panelu modułu ćwiczeniowego. Im większe napięcie, tym moduł Peltiera będzie się szybciej nagrzewał. Nie należy przekraczać napięcia 4 V.

Czujnik temperatury (którym jest komercyjnie dostępny półprzewodnikowy czujnik LM335 firmy National Semiconductor) należy zetknąć z nagrzewającą się powierzchnią modułu Peltiera.

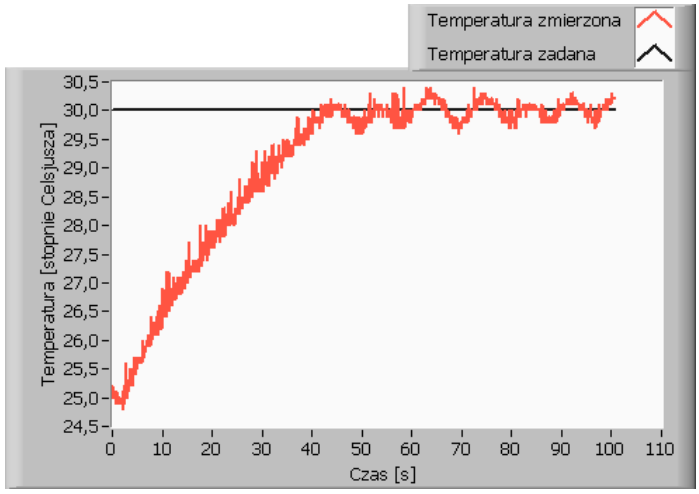
Do działania tej (jak i następnych) aplikacji wystarczą 3 rozkazy przesyłane przez port szeregowy: TEMP, PELT1 i PELT0. Dodatkowo, aplikację można wzbogacić o sygnalizację nagrzewania poprzez zapalenie się diody w module (rozkaz LED1).

4. Automatyczna skokowa stabilizacja temperatury

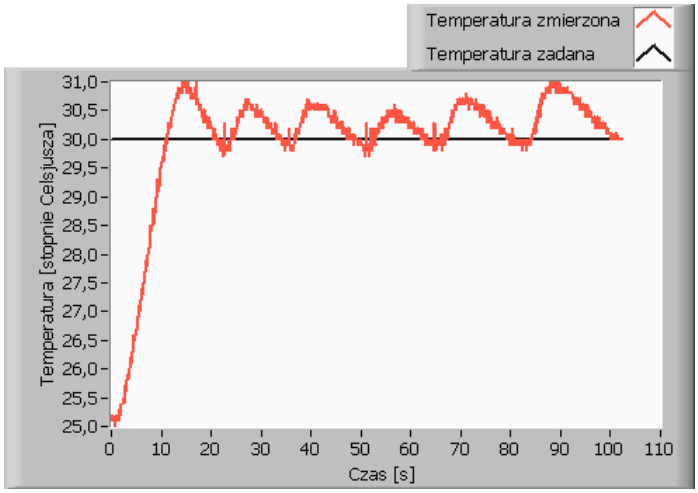
Najprostszym sposobem stabilizacji temperatury przy określonej wartości jest włączanie elementu grzejącego, jeśli temperatura jest niższa od zadanej, a wyłączaniu go, jeśli temperatura jest wyższa. W tym punkcie ćwiczenia należy stworzyć tak właśnie działającą aplikację. Parametrami określanymi przez użytkownika mają być zadana temperatura T_z (do której ma dojść mierzona temperatura T na powierzchni modułu Peltiera) oraz wartość tolerancji ΔT (czasami podwojona wartość ΔT nazywana jest w technice grzewczej histerezą). Napięcie na module Peltiera ma być załączane, jeśli $T<T_z-\Delta T$, a wyłączane, jeśli $T>T_z+\Delta T$. Dodatkowo wykreślany ma być wykres prezentujący proces dochodzenia w czasie do zadanej temperatury. Na rys. 2 pokazano przykładowe wykresy otrzymane dla dwóch różnych napięć zasilających moduł Peltiera.

Jak widać z przedstawionych wykresów, stabilizowana temperatura wykazuje oscylacje wokół zadanej wartości, które w przypadku wyższego napięcia są bardzo znaczące i wyraźnie przekraczają wartość ΔT. Zmniejszenie wartości tolerancji zmniejsza nieco wahania temperatury, ale nie eliminuje ich całkowicie. Dlatego konieczne jest zastosowanie bardziej złożonego algorytmu stabilizacji temperatury.

A)



B)



Rys. 2. Efekt działania skokowej stabilizacji temperatury na powierzchni modułu Peltiera dla T_z=30 °C i ΔT=0.1 °C przy napięciu na module Peltiera równym: A) 1.5 V (prąd 0.6 A); B) 3 V (prąd 1.5 A).

5. Automatyczna stabilizacja temperatury w oparciu o algorytm PID

Algorytm PID jest sposobem regulacji wartości rozmaitych wielkości, bardzo często stosowanym w technice przemysłowej i laboratoryjnej. Dla potrzeb tego ćwiczenia przedstawiona zostanie poniżej jedna z jego podstawowych postaci. Przyjmijmy następujące oznaczenia:

S –sygnał sterujący mocą elementu grzejącego [% maksymalnej mocy];

B – błąd (B = temperatura zadana – temperatura zmierzona);

W – wzmocnienie; C - czynnik przy całce; R – czynnik przy różniczce; t- czas.

Ogólny wzór na sygnał sterujący w algorytmie PID ma postać:

$$S(t) = W\left(B(t) + C\int B(t)\,dt + R\,dB(t)/dt \right). \qquad (1)$$

Poszczególne człony noszą nazwę: P – proporcjonalny, I –całkowy (*Integral*), D – różniczkowy (*Derivative*). Stosując tylko niektóre z członów otrzymujemy uproszczone algorytmy (np. PI).

Człon proporcjonalny wpływa na zmniejszenie błędu, ale samodzielnie nigdy nie zniweluje go do zera. Dla małego wzmocnienia występuje stosunkowo długi czas narastania temperatury i ostatecznie jej wartość nigdy nie dochodzi do wartości zadanej. Dla dużego wzmocnienia można spodziewać się odpowiedzi oscylacyjnej o niegasnących oscylacjach. Człon całkowy redukuje błąd do zera, ale może wprowadzić pewne przeregulowania aż do wystąpienia oscylacji włącznie (gasnących). Dodanie członu różniczkowego może zminimalizować niepożądane oscylacje temperatury wokół wartości zadanej i uzyskać optymalny wynik regulacji.

Po zróżniczkowaniu równania (1) dostajemy następujące wyrażenie:

$$dS/dt = W (dB/dt + C B + R d^2B/dt^2). \qquad (2)$$

Ostatni wzór można w prosty sposób wykorzystać w procedurze iteracyjnej. W pętli dla n-tego kroku (jeśli odstęp czasowy pomiędzy kolejnymi krokami jest odpowiednio mały i wynosi dt) ma on postać:

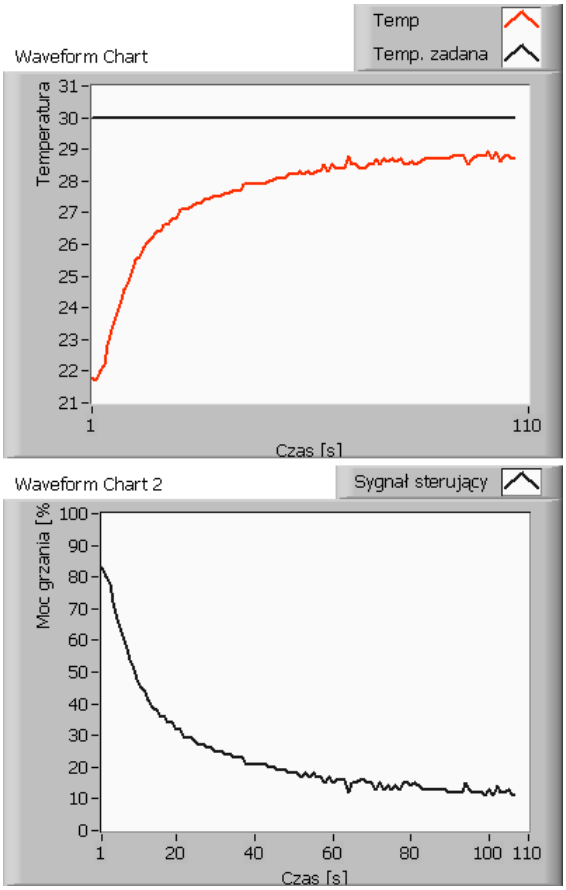
$$S_n = S_{n-1} + W \{ (B_n - B_{n-1}) + C B_n + R [(B_n - B_{n-1}) - (B_{n-1} - B_{n-2})] \}, \qquad (3)$$

gdzie C jest wyrażone w jednostkach 1/dt, a R jest wyrażone w jednostkach dt.

Jak widać, moc elementu grzejącego w algorytmie PID zmienia się w sposób ciągły. W przypadku tego ćwiczenia mamy możliwość jedynie skokowego załączania lub wyłączania określonego z góry napięcia (ustalanego na zewnętrznym zasilaczu), powodującego przepływ prądu przez moduł Peltiera. Aby zastosować algorytm PID należy zatem opracować „sztuczną" metodę ciągłej regulacji mocy elementu grzejącego. Najprostszym rozwiązaniem jest wybranie określonego przedziału czasu (tzw. okresu impulsowania) i w jego obrębie załączenie modułu Peltiera na taki procent czasu, który odpowiada procentowi maksymalnej mocy. Na przykład, jeśli wartość sygnału sterujący (obliczonego z algorytmu PID) wynosi 20 %, to przy okresie impulsowania 1 s moduł Peltiera powinien być włączony przez 200 ms (20 % z 1 s), a przez pozostałe 800 ms wyłączony. W technice grzewczej metoda taka nosi czasami nazwę modulacji szerokości impulsu (PWM, *Pulse Width Modulation*).

Mając tak przygotowaną funkcję sterującą nagrzewaniem modułu Peltiera w sposób pseudo-ciągły (najlepiej w postaci podprogramu, *subVI*), należy stworzyć zasadniczą dla tego ćwiczenia aplikację kontrolującą temperaturę i sterującą procesem nagrzewania w oparciu o

algorytm PID dany wzorem (3). Aby uniknąć sytuacji, gdy $S < 0\ \%$ lub $S > 100\%$ należy dodać odpowiednie limity na wartość sygnału sterującego S. Użytkownik takiej aplikacji powinien podać parametry algorytmu PID (wartości W, C i R), zadaną temperaturę oraz okres impulsowania. Proces dochodzenia w czasie do zadanej temperatury powinien być zobrazowany na wykresie, podobnie jak zmiany w czasie sygnału sterującego S. Przykład takiego wykresu znajduje się na rys. 3. Dodatkowo, aplikacja może być rozszerzona o obsługę ewentualnych błędów i zapis danych o przebiegach na dysku.



Rys. 3. Efekt działania stabilizacji temperatury na powierzchni modułu Peltiera w oparciu o algorytm PID dla parametrów równych: W=10, C=0, R=0, przy napięciu na module 3 V i okresie impulsowania 1 s.

6. Dobór optymalnych parametrów w algorytmie PID

Kluczową sprawą w algorytmie PID jest określenie optymalnych wartości parametrów W, C i R. Zależą one od wielu czynników, w przypadku tego ćwiczenia np. od wartości napięcia zasilającego moduł Peltiera (wpływa na szybkość nagrzewania się układu), temperatury otoczenia (wpływa na szybkość studzenia układu), bezwładności cieplnej układu.

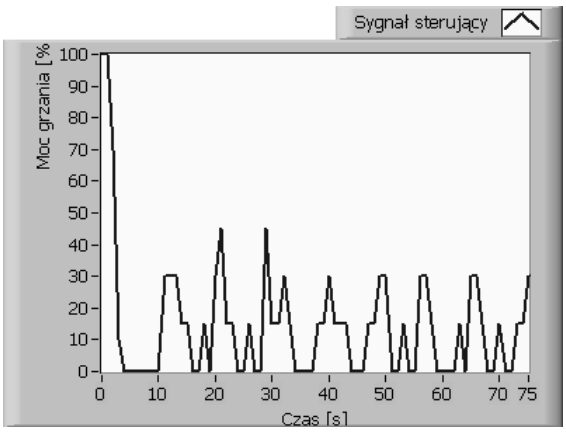
Optymalne wartości bardzo często trzeba dobierać ręcznie, istnieje jednak kilka metod pozwalających na wstępne oszacowanie wartości W, C i R.

Jedną z nich jest oscylacyjna metoda Ziegler'a – Nichols'a, której schemat postępowania przedstawia się następująco:

a) Ustalamy małe W oraz C=R=0.
b) Zwiększamy stopniowo W, aż uzyskamy oscylacje sygnału sterującego mocą grzejnika (oscylacje powinny mieć stałą amplitudę).
c) Notujemy krytyczną wartość wzmocnienia W_k, przy którym zaczęły się oscylacje, oraz mierzymy ich okres T.
d) Parametry kontrolera PID wyliczamy z poniższej tabelki:

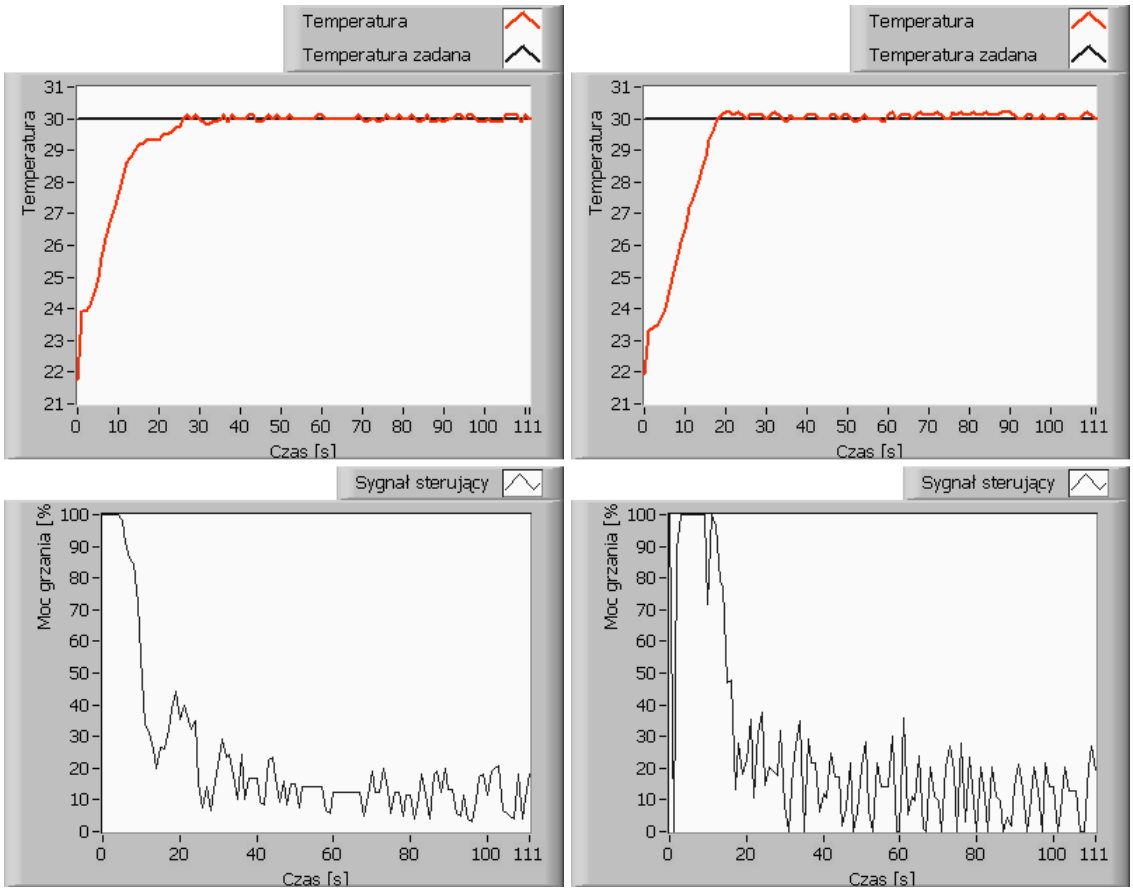
	W	C	R
P	0.5 W_k	—	—
PI	0.45 W_k	1.2/T	—
PID	0.6 W_k	2/T	T/8

Na przykład, dla napięcia 3 V otrzymano oscylacje sygnału sterującego przy wzmocnieniu Wk=150, których okres wynosił ok. T=9 s (patrz rys. 4). Obliczone więc, na podstawie powyższej tabelki, szacunkowe optymalne wartości wynoszą W=67, C=0,13 i R=0 (dla algorytmu PI) lub W=90, C=0,22 i R=1,12 (dla algorytmu PID).



Rys. 4. Oscylacje sygnału sterującego przy parametrach algorytmu PID: W=150, C=0, R=0, okresie impulsowania 1 s i napięciu 3 V

Zastosowanie tak oszacowanych parametrów do kontroli temperatury przedstawia rys. 5. Jak widać, stabilizacja temperatury jest znacznie lepsza niż w przypadku algorytmu skokowego, przedstawionego dla tego samego napięcia 3 V na rys. 2B. W zasadzie algorytm PI (bez członu różniczkowego) daje w tym wypadku lepsze wyniki niż PID, który przejawia czasami tendencje do niespodziewanych wahań temperatury. Być może wynika to z faktu nie do końca optymalnego dobrania parametrów w przypadku PID.



(A) (B)

Rys. 5. Efekt działania stabilizacji temperatury na powierzchni modułu Peltiera w oparciu o algorytm PID dla parametrów równych: (A) W=67, C=0,13 i R=0; (B) W=90, C=0,22 i R=1,12; napięcie na module wynosiło 3 V a okres impulsowania 1 s.