

Wykład 4

22.03.2024 r.

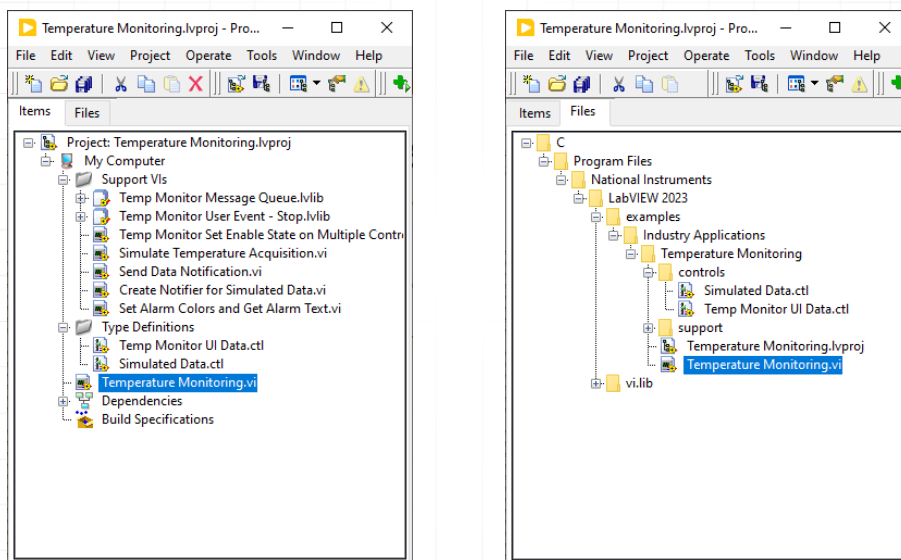
Dokumentacja i wizualizacja

Adrian Kaim



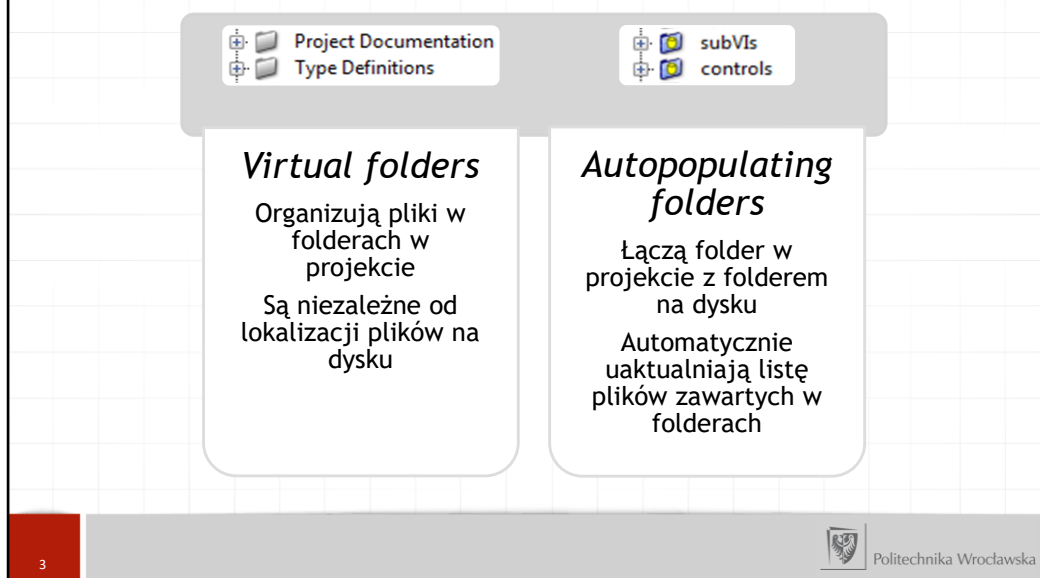
Politechnika Wrocławska

Project Explorer



Programy w LabVIEW mogą być organizowane w projekty, zawierające pliki konieczne do ich działania (pliki kodu, biblioteki, niestandardowe kontrolki, pliki tekstowe, pliki graficzne, etc.). W ramach projektu pliki te mogą zostać posegregowane w folderach. Podstawowym narzędziem służącym do obsługi zawartości projektu jest *Project Explorer*, którego okno pojawia się w momencie otwarcia/stworzenia projektu. Zawartość projektu można przeglądać jako listę jego elementów lub jako listę plików na dysku komputera.

Foldery w projekcie



Istnieją dwa główne typy folderów, jakie dostępne są w projektach w LabVIEW: wirtualne (*Virtual Folders*) oraz uzupełniające się automatycznie (*Autopopulating Folders*).

Aby dodać folder wirtualny należy kliknąć prawym przyciskiem myszy na poziom w którym chcemy go stworzyć i wybrać *New>Virtual Folder*. Aby dodać pliki do takiego folderu należy kliknąć na niego prawym przyciskiem myszy i wybrać *Add>File*. Można także stworzyć folder i jednocześnie (jednorazowo) automatycznie dodać jego zawartość - kliknąć prawym przyciskiem myszy na poziom w którym chcemy go stworzyć i wybrać *Add>Folder (Snapshot)*....

Aby stworzyć folder typu *Autopopulating*, należy kliknąć prawym przyciskiem myszy na poziom w którym chcemy go stworzyć i wybrać *Add>Folder (Autopopulating)*....

IV Properties

The screenshot shows the LabVIEW VI Properties dialog box with two tabs visible. The 'Documentation' tab is active, showing a text area for the VI description, a 'Local Help File' dropdown, and fields for 'Help tag' and 'Help path'. The 'Window Appearance' tab is also shown, with options for 'Window title', 'Same as VI name' checkbox, and radio buttons for 'Top-level application window', 'Dialog', 'Default', and 'Custom'. A 'Customize...' button is present. A small thumbnail image of a window with a red question mark is shown next to the 'Custom' option.

Documentation
Pozwala tworzyć opis programu, który wyświetla się w *Quick Help*. Dodatkowo pozwala dodawać odnośniki do pomocy.

Window Appearance
Pozwala zmieniać sposób wyświetlania *Front Panelu* działającego programu.

Aby wyświetlić okno własności programu (*VI Properties*), należy wybrać *File > VI Properties*.

Zasady projektowania *Front Panelu*

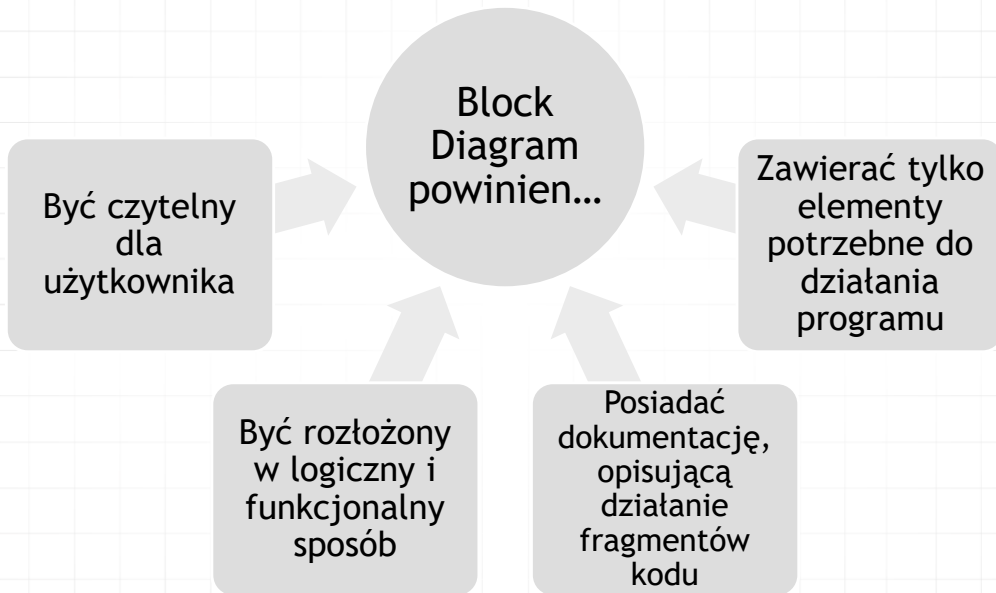


Zasady projektowania *Front Panelu*

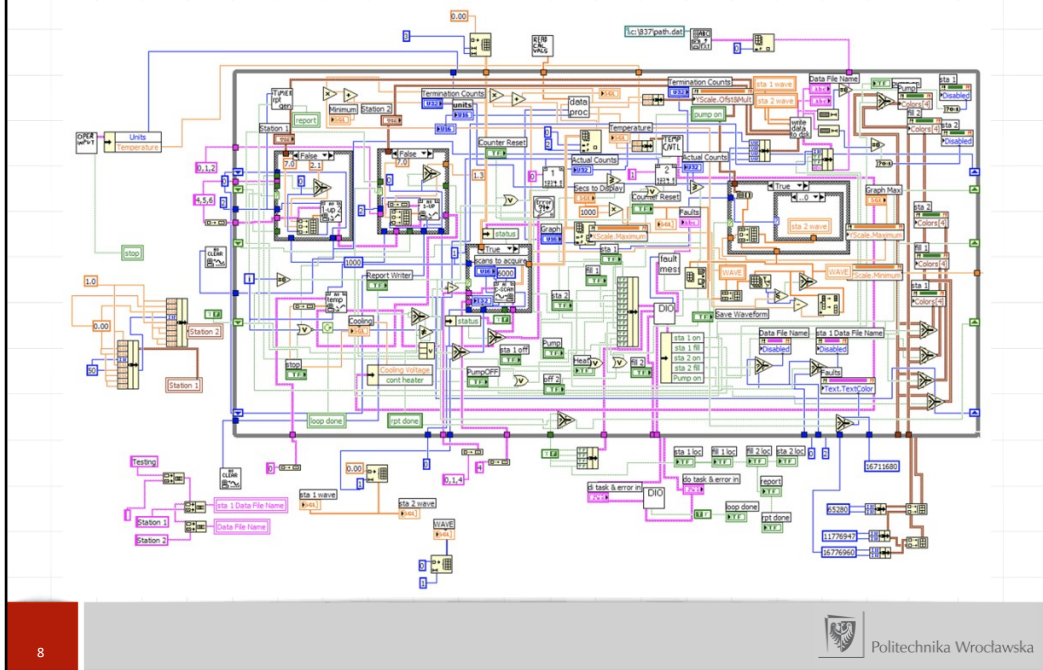


Przykłady dobrze i źle zaprojektowanych *Front Paneli*.

Zasady projektowania *Block Diagramu*



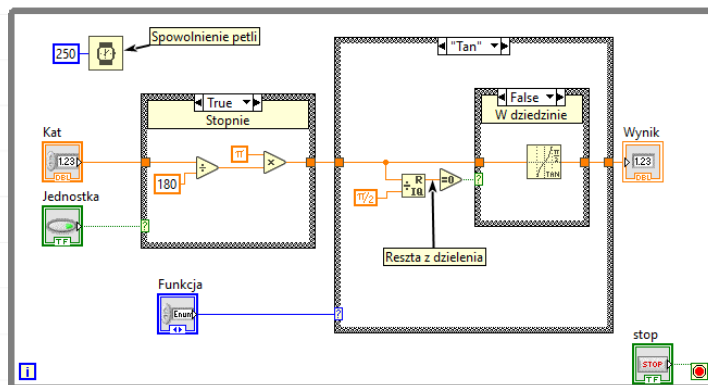
Block Diagram „Spaghetti”



Przykład źle zaprojektowanego *Block Diagramu*.

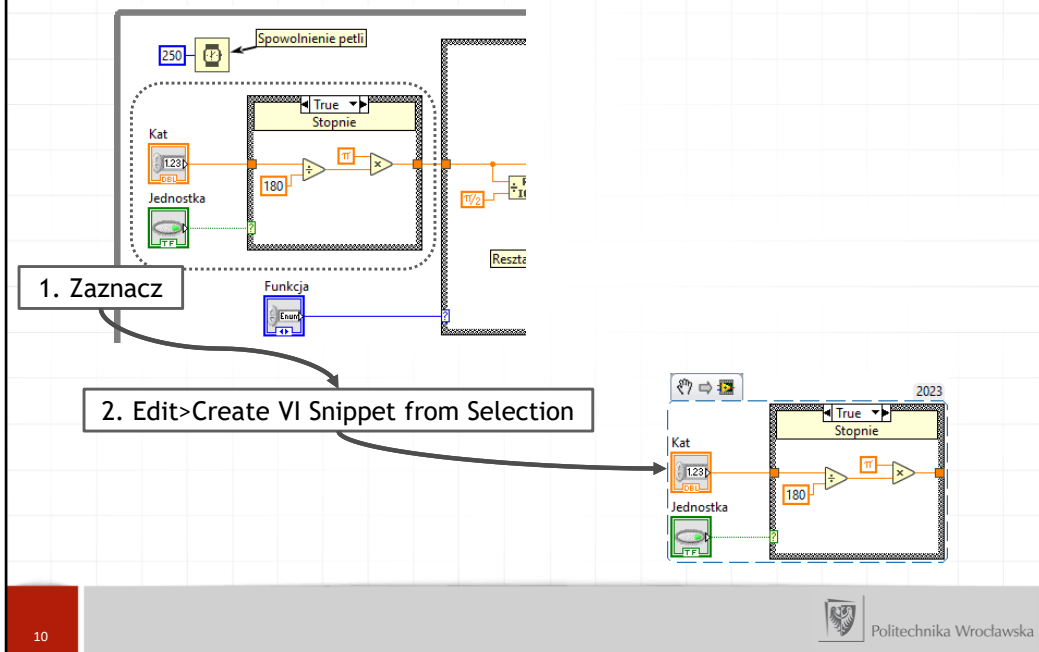
Etykiety (*Label*)

Wykład 2/ Zadanie 5
Adrian Kaim 2023



Do opisywania *Block Diagramu* oraz *Front Panelu* można wykorzystać etykiety. Aby dodać swobodną etykietę należy dwukrotnie kliknąć lewym przyciskiem myszy w wolnym miejscu. Swobodne etykiety mogą zostać zaczepione do elementów *Block Diagramu* przez połączenie ich strzałką, pojawiającą się po najechaniu kursorem na etykietę. Możliwe jest także stworzenie etykiet poddiagramów w strukturach – aby to zrobić, należy kliknąć prawym przyciskiem myszy na daną strukturę i wybrać *Visible Items>Subdiagram Label*.

Wycinki kodu (*Snippet*)



Pomocne w dokumentacji kodu (np. na forach internetowych) bywają tzw. wycinki kodu (*Snippet*). Wycinek taki jest graficzną reprezentacją fragmentu Block Diagramu, zapisaną w formacie PNG. W metadanych tego pliku graficznego zapisywany jest cały kod reprezentowany przez wycinek, co pozwala na proste uruchomienie go w środowisku LabVIEW.

Aby stworzyć wycinek należy zaznaczyć wybrany fragment kodu i wybrać *Edit>Create VI Snippet from Selection*, po czym należy go zapisać. Zapisanego wycinka nie można edytować w innych programach graficznych, gdyż usuwają one zawarty w metadanych kod.

Aby umieścić wycinek na Block Diagramie, wystarczy przeciągnąć na niego plik z wycinkiem (w przypadku gdy wycinek pochodzi z Internetu, należy go najpierw zapisać na komputerze).

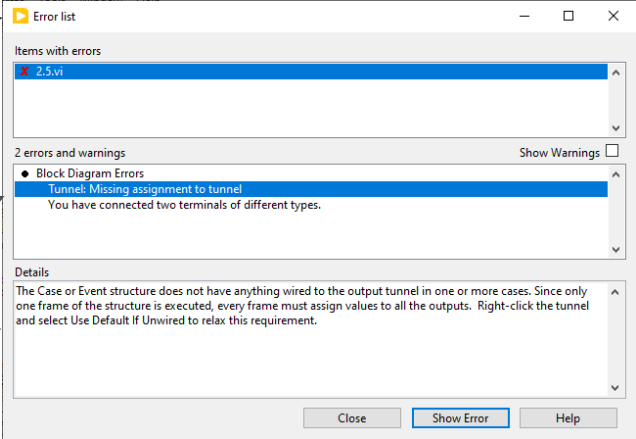
Debuging

Złamana strzałka oznacza błąd kompilacji.

LPM

Lista błędów uniemożliwiających kompilację programu.

Szczegóły na dotyczące zaznaczonego błędu.



Items with errors

- 2.5.vi

2 errors and warnings ☐ Show Warnings

- Block Diagram Errors
 - Tunnel: Missing assignment to tunnel
You have connected two terminals of different types.

Details

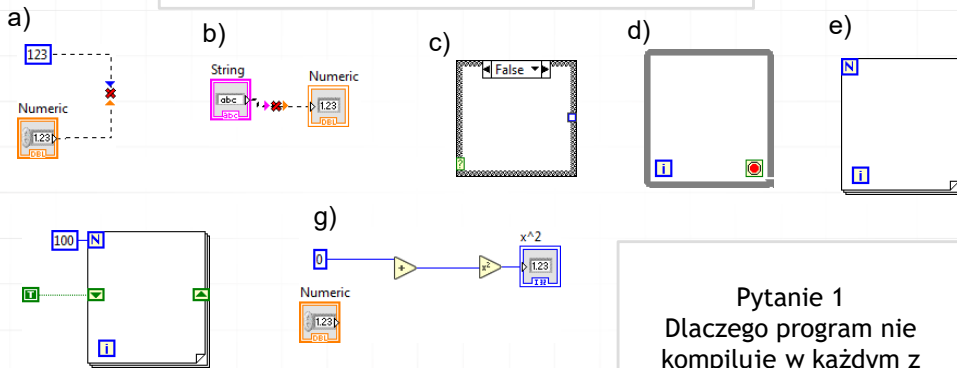
The Case or Event structure does not have anything wired to the output tunnel in one or more cases. Since only one frame of the structure is executed, every frame must assign values to all the outputs. Right-click the tunnel and select Use Default If Unwired to relax this requirement.

Close Show Error Help

Błędy kompilacji sygnalizowane są złącaną strzałką uruchamiania programu. Po jej kliknięciu uzyskuje się dostęp do listy błędów, które uniemożliwiają kompilację i ich szczegółowych opisów.

Debuging

Częste przyczyny błędu kompilacji:



Pytanie 1
Dlaczego program nie
kompiluje w każdym z
przypadków?

Narzędzia do debugowania



Wykonywanie programu jest spowolnione, a przepływające dane są podświetlane - pozwala to na kontrolę przepływu danych



Wykonywanie Vi krok po kroku, pozwala na kontrolę działania poszczególnych węzłów



Próbkowanie pozwala na podglądanie wartości przepływających w drutach

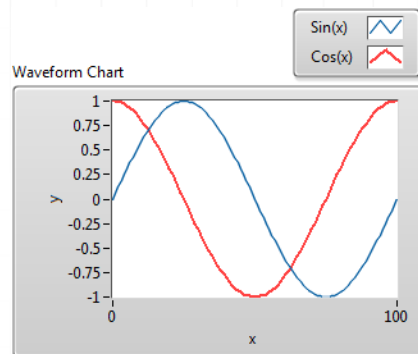
Wykres typu *Waveform Chart*

Indykator numeryczny

Może wyświetlać wiele wykresów

Na osi x kolejne próbki

Wykorzystywany do prezentacji danych
„w czasie rzeczywistym”



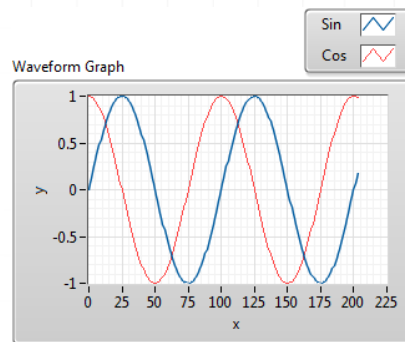
Wykres typu *Waveform Graph*

Indykator numeryczny

Może wyświetlać wiele wykresów o stałym próbkowaniu

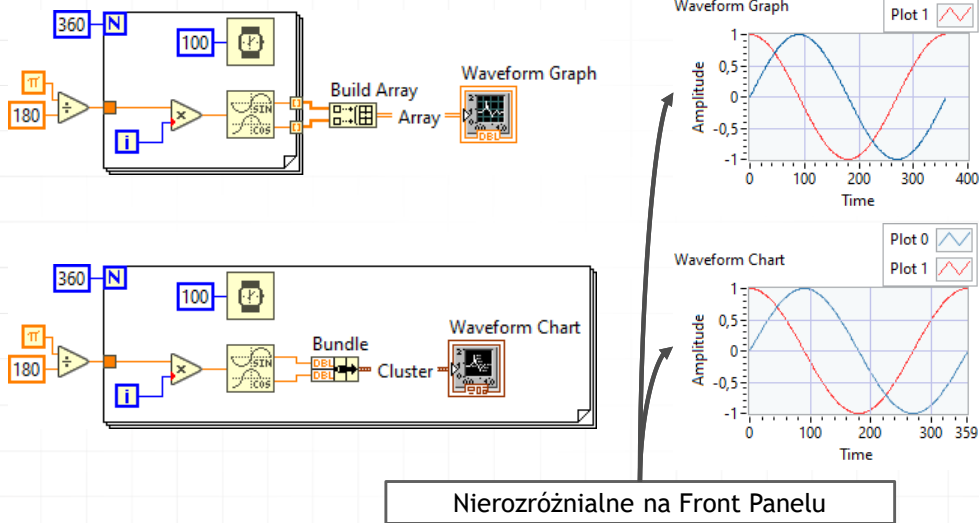
Na osi x kolejne próbki

Wykorzystywany do prezentacji wcześniej wykreowanych danych



Sprawdź też: [Przykład 1](#)

Chart vs Graph



Waveform Graph oczekuje gotowego zestawu danych w postaci tablicy 1D. Aby wyświetlić wiele sygnałów na jednym wykresie, należy złączyć je w tablicę 2D.

Waveform Chart oczekuje w każdym okresie pętli jednego punktu (jednej wartości). Aby wyświetlić kilka sygnałów na jednym wykresie, należy zebrać je w klaster.

Wykres typu *Graph XY*

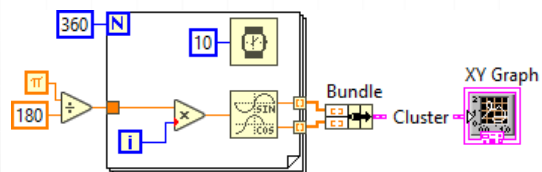
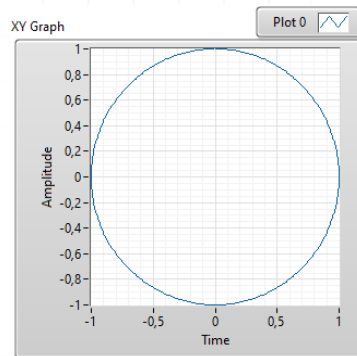
Graficzne przedstawienie danych

Wyświetla zależność $x(y)$

Jako danych wejściowych używa dwóch tablic połączonych w klastę

Można przyłączyć wiele klastrów

Wykorzystywany do prezentacji wcześniej wykreowanych danych



Wykres typu XY Graph oczekuje klastra złożonego z tablicy 1D współrzędnych X oraz tablicy 1D współrzędnych Y. Aby wyświetlić wiele sygnałów na takim wykresie, należy zebrać klastry w tablicę.

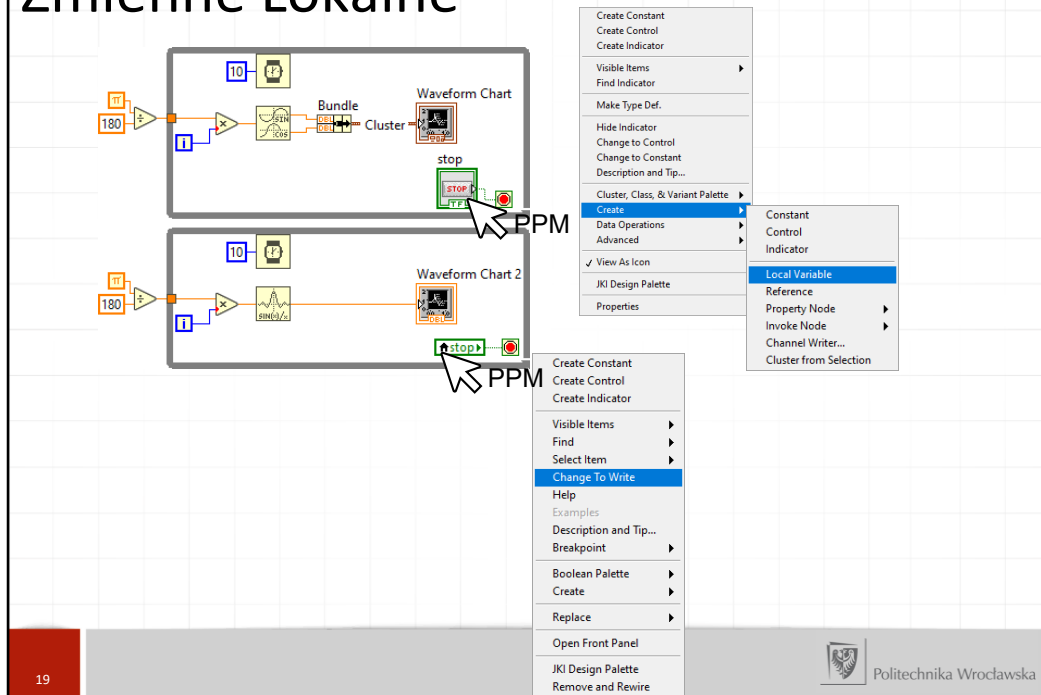
Property Node

The screenshot illustrates the process of adding a Property Node to an XY Graph in the JKI Design Palette. The main window shows the 'JKI Design Palette' on the left, with the 'Property Node' option selected under the 'Cluster, Class, & Variant Palette' section. The 'XY Graph' component is highlighted in the 'Value' section of the 'Property Node' menu. A callout box points to the 'XY Graph' component, stating 'Wartość odczytywana z Property Node' (Value read from Property Node). Another callout box points to the 'Value' property of the 'XY Graph' component, stating 'Wartość wpisywana do Property Node' (Value written to Property Node). The 'XY Graph' component is shown with a 'Value' property set to 'Plot.Color'. The 'JKI Design Palette' is also visible on the right side of the screen.

Aby stworzyć *Property Node* należy kliknąć prawym przyciskiem myszy na kontrolkę/indykator i wybrać *Create>Property Node*, a następnie wskazać własność, której ma dotyczyć. Jeden *Property Node* może dawać dostęp do więcej niż jednej własności (graficzne rozciągnięcie lub PPM>*Add Element*). Jeśli korzysta się z dostępu do wielu własności w ramach jednego *Property Node*, wpisywanie/odczytywanie odbywa się kolejno z góry na dół.

Aby przełączyć własność pomiędzy trybem odczytywania i wpisywania należy kliknąć na nią prawym przyciskiem myszy i wybrać *Change To Read/Write*.

Zmienne Lokalne

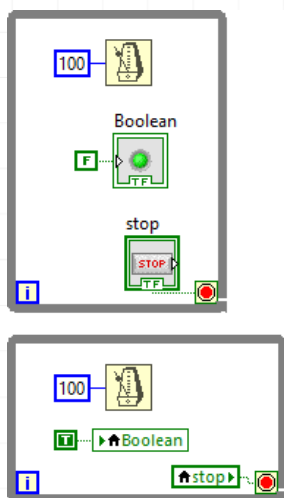


Zmienne lokalne pozwalają uzyskać dostęp do kontrolki lub indykatora w wielu miejscach na Block Diagramie.

Aby stworzyć zmienną lokalną należy kliknąć prawym przyciskiem myszy na kontrolkę/indikator i wybrać *Create>Local Variable*.

Możliwe jest przełączanie trybu działania zmiennej z wpisywania na odczytywanie wartości, w tym celu należy kliknąć na nią prawym przyciskiem myszy i wybrać *Change To Read/Write*.

Zmienne Lokalne



Należy uważać, aby nie nadpisywać zmiennych lokalnych w dwóch miejscach programu, a w szczególności by nie robić tego jednocześnie. W przypadku takim nie ma pewności które nadpisanie wykona się pierwsze, przez co program może działać w nieoczekiwany i niestabilny sposób.

Dioda w pokazanym powyżej przykładzie migać będzie w losowy sposób.

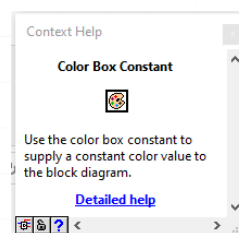
Zadanie 1

Pobierz program ze strony www.

Zlokalizuj błędy w programie. Napraw je. Oznacz
naprawione błędy etykietami na Block

Diagramie albo Front Panelu.

Wskazówka: typ
danych *Color Box
Constant*



Zadanie 2

Zaprojektuj etykietę nagłówkową, zawierającą miejsce na:

- nazwę programu lub funkcji,
- dane autora,
- numer wersji kodu oraz datę stworzenia,
- opis funkcjonalności.

Zapisz etykietę jako wycinek (*snippet*).

Zadanie 3

Napisz program do rysowania prostej zadanej przez współczynnik kierunkowy oraz wyraz wolny.

Podpowiedź: do utworzenia tablic wykorzystaj pętle *For* oraz *Auto-indexing*.

Zadanie 4

Napisz program do symulacji sygnału sinusoidalnego, który będzie wyświetlany na wykresie typu *Waveform Chart*. Wykorzystaj *Property Nodes* kontrolki *Waveform Chart*, aby umożliwić użytkownikowi zmianę koloru tła, wykresu, siatki. (Podpowiedź: wykorzystaj kontrolkę *Color box*).

Zadanie 5

Napisz program symulujący pracę sygnalizatora drogowego.



Zadanie 6

Na wykresie typu *Graph XY* narysuj okrąg o zadanym przez użytkownika promieniu.

Zadanie 7

Napisz program do tworzenia krzywych Lissajous.

