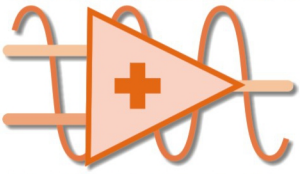
Dariusz Świsulski

# Przykłady cyfrowego przetwarzania sygnałów w LabVIEW



Wydawnictwo Politechniki Gdańskiej

Dariusz Świsulski

# Przykłady cyfrowego przetwarzania sygnałów w LabVIEW

Gdańsk 2014

PRZEWODNICZĄCY KOMITETU REDAKCYJNEGO
WYDAWNICTWA POLITECHNIKI GDAŃSKIEJ
*Janusz T. Cieśliński*

RECENZENT
*Wiesław Kiciński*

PROJEKT OKŁADKI
*Katarzyna Olszonowicz*

Wydanie I – 2012

Wydano za zgodą
Rektora Politechniki Gdańskiej

Publikacja dostępna tylko w wersji elektronicznej –
Pomorska Biblioteka Cyfrowa http://pbc.gda.pl

Oferta wydawnicza Politechniki Gdańskiej jest dostępna pod adresem
http://www.pg.edu.pl/wydawnictwo/katalog
zamówienia prosimy kierować na adres wydaw@pg.gda.pl

© Copyright by Wydawnictwo Politechniki Gdańskiej
Gdańsk 2014

Utwór nie może być powielany i rozpowszechniany, w jakiejkolwiek formie
i w jakikolwiek sposób, bez pisemnej zgody wydawcy

ISBN 978-83-7348-540-2

WYDAWNICTWO POLITECHNIKI GDAŃSKIEJ

Wydanie II. Ark. wyd. 5,7, ark. druku 5,25, 1048/820

# Spis treści

# Cyfrowego przetwarzania sygnałów w środowisku LabVIEW

Zintegrowane środowisko LabVIEW firmy National Instruments służy do przygotowania oprogramowania systemów pomiarowych, sterowania itp. W odróżnieniu od klasycznych języków programowania, w których program złożony jest z poleceń umieszczonych w kolejnych wierszach, LabVIEW wykorzystuje graficzny język programowania. Program tworzony jest w postaci diagramu, na którym poszczególne operacje, przedstawiane w postaci symboli, łączone są zgodnie z kierunkiem przepływu sygnałów. Programy utworzone przy użyciu LabVIEW nazywane są przyrządami wirtualnymi (*Virtual Instruments - VIs*). W przyrządach tych realizacja sprzętowa pewnych funkcji zastąpiona jest odpowiednim oprogramowaniem wykonywanym przez komputer ogólnego przeznaczenia. Dotyczy to głównie obsługi przyrządu oraz realizacji algorytmów przetwarzania sygnałów. Pozwala to na łatwą modyfikację takiego przyrządu i szybkie dostosowywanie go do wymagań użytkownika. Można w prosty sposób rozbudowywać algorytmy przetwarzania i analizy sygnałów oraz sposoby prezentacji wyników pomiarów.

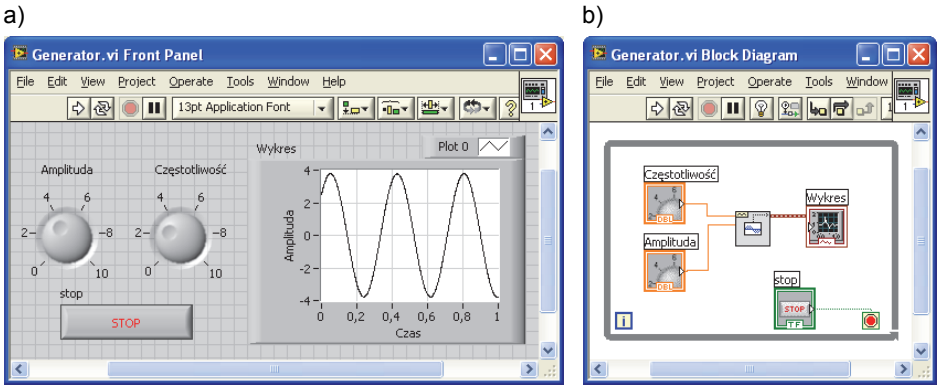
Programy pisane w LabVIEW przygotowywane są w sposób hierarchiczny, tzn. dany przyrząd wirtualny może być użyty w innym przyrządzie jako podprogram. Ponieważ każdy podprogram może być uruchamiany oddzielnie, modułowa struktura ułatwia usuwanie błędów i uruchamianie całego programu.

## 1.1. Praca z LabVIEW

Przygotowując aplikację przy użyciu LabVIEW, mamy do dyspozycji dwa główne okna: panel i diagram (rys. 1.1).

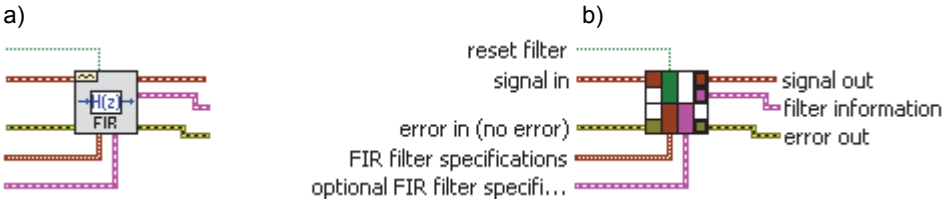
Panel pełni funkcję interaktywnego interfejsu z użytkownikiem. Symuluje on płytę czołową rzeczywistego przyrządu. Umieszczone są na nim elementy służące do wprowadzania danych do programu (*Controls*), np.: przełączniki, pokrętła, i służące do wyprowadzania danych z programu do użytkownika (*Indicators*), np.: wskaźniki, wyświetlacze. Do obsługi elementów na panelu wykorzystywana jest mysz lub klawiatura.

Diagram zawiera program źródłowy aplikacji w języku graficznym.




Rys. 1.1. Okna środowiska LabVIEW: a) panel, b) diagram

Z przyrządem wirtualnym związana jest też ikona (*Icon*), umożliwiająca umieszczenie danego przyrządu wirtualnego w innej aplikacji jako podprogramu, a także zaciski (*Connector*), pozwalające na wprowadzanie danych do podprogramu i wyprowadzanie wyników (rys. 1.2).




Rys. 1.2. Ikona (a) i zaciski (b) przyrządu wirtualnego *Digital FIR Filter VI*

Jeżeli aktywnym oknem jest okno panelu, to w każdej chwili możemy przejść do diagramu, wybierając z opcji *Window* menu polecenie *Show Block Diagram* lub naciskając klawisze Ctrl-E. Podobnie z diagramu przechodzimy do panelu, wybierając z opcji *Window* menu polecenie *Show Front Panel*, lub naciskając klawisze Ctrl-E.

W górnej części okna panelu znajduje się pasek przycisków narzędziowych (rys. 1.3). Podobny pasek, rozbudowany o obsługę debuggera, znajduje się w górnej części okna diagramu.


Rys. 1.3. Pasek przycisków narzędziowych w LabVIEW

Znaczenie poszczególnych przycisków jest następujące:

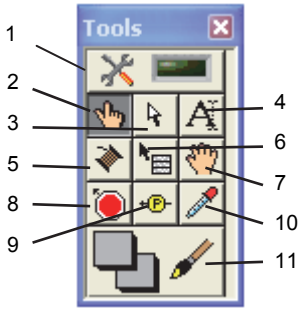
1 – przycisk do uruchamiania przyrządów wirtualnych, po uruchomieniu strzałka zmienia kolor na czarny,

2 – przycisk do uruchomienia przyrządu wirtualnego w sposób ciągły, po wykonaniu programu rozpoczyna się jego wykonywanie od początku (przycisk używany przy testowaniu programu),

3 – przycisk do przerwania wykonywania programu,

4 – przycisk do wstrzymania lub wznowienia wykonywania programu,
5 – wybór rodzaju czcionki, jej wielkości, koloru i sposobu justowania,
6 – wyrównywanie położenia elementów,
7 – wyrównywanie odległości między elementami,
8 – zmiana wymiarów grupy obiektów,
9 – zmiana kolejności nakładania się wybranych elementów,
10 – pokazanie lub ukrycie okna pomocy.

Wybierając z opcji *View* menu LabVIEW polecenie *Tools Palette*, otworzymy okno, w którym dostępne są narzędzia do edycji i uruchamiania przyrządów wirtualnych (rys. 1.4). Okno to otworzy się również po równoczesnym naciśnięciu klawisza *Shift* i prawego przycisku myszy.




Rys. 1.4. Narzędzia w oknie *Tools Palette*

Mamy tutaj do dyspozycji następujące narzędzia:
1 – automatyczny dobór narzędzia,
2 – narzędzie do zmiany nastaw obiektów na panelu,
3 – narzędzie do zmiany położenia, wymiarów i wyboru obiektów,
4 – narzędzie do edycji tekstów i tworzenia napisów,
5 – narzędzie do łączenia obiektów na diagramie,
6 – narzędzie do rozwijania menu z dostępnymi obiektami,
7 – narzędzie do przewijania zawartości okna,
8 – narzędzie do ustawiania punktów przerwania programu,
9 – narzędzie do ustawiania sondy pokazującej wartość sygnału w wybranym punkcie programu,
10 – narzędzie do kopiowania wybranego koloru, w celu użycia przez następne narzędzie,
11 – narzędzie do zmiany kolorów.

Przy wciśniętym pierwszym przycisku odpowiednie narzędzie dobierane jest automatycznie. Szybką zmianę między pierwszymi czterema narzędziami można uzyskać, używając klawisza tabulacji, każde naciśnięcie przełącza na następne narzędzie.

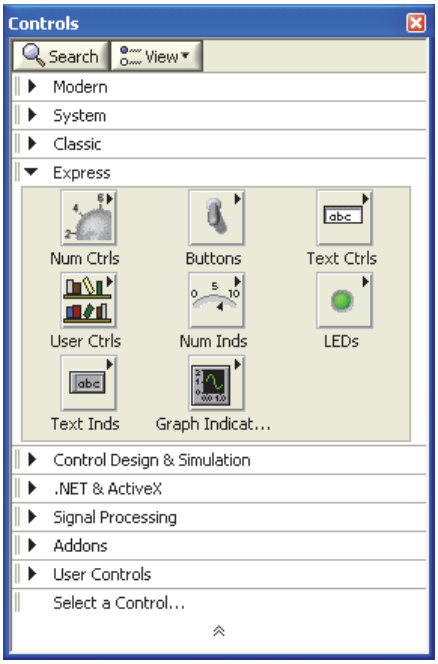
Jeżeli chcemy umieścić obiekt na panelu, należy otworzyć okno *Controls*, wybierając z opcji *View* menu polecenie *Controls Palette* lub, przesuwając kursor na okno panelu i naciskając prawy przycisk myszy. Otwiera się okno pokazane na rysunku 1.5.

Dwa przyciski w górnej części pozwalają na poszukiwanie obiektu według nazwy (*Search*) i zmianę sposobu przedstawiania zawartości palety (*View*).

Elementy umieszczone są w grupach podzielonych na kilka zbiorów, m.in.:
*Modern* – styl nowoczesny,
*System* – styl systemowy,
*Classic* – stylu klasyczny,
*Express* – dostęp do podstawowych grup elementów.

*User Controls* pozwala na szybki dostęp do własnych elementów użytkownika, umieszczonych w katalogu User.lib. *Select a Control* pozwala na dostęp do elementów zapisanych w pliku umieszczonym w dowolnym miejscu na dysku.

Zawartość każdego zbioru można rozwinąć, klikając na odpowiednią strzałkę z lewej strony lub jego nazwę.




Rys. 1.5. Okno *Controls*

Na rysunku 1.5 pokazano rozwinięty zbiór *Express* zawierający osiem grup:

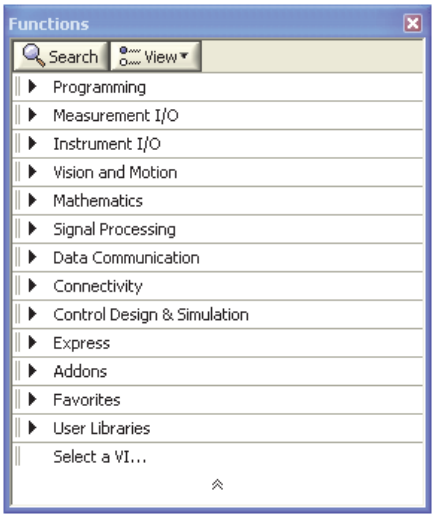
*Num Ctrls* – elementy do wprowadzania do programu danych liczbowych (np. pokrętła, suwaki),
*Buttons* – elementy do wprowadzania do programu danych typu logicznego (np. przełączniki, przyciski),
*Text Ctrls* – elementy do wprowadzania do programu danych w postaci tekstu lub ścieżki dostępu do pliku,
*User Ctrls* – elementy utworzone przez użytkownika,
*Num Inds* – elementy do odczytywania z programu danych liczbowych (np. wskaźniki cyfrowe i wskazówkowe),
*LEDs* – elementy do odczytywania z programu danych typu logicznego (np. wskaźniki w postaci diody LED),

*Text Inds* – elementy do odczytywania z programu danych w postaci tekstu lub ścieżki dostępu do pliku,
*Graph Indicat…* – elementy do przedstawiania w programie informacji w postaci graficznej na wykresach.

W celu wybrania obiektu należy kliknąć na odpowiednią grupę, a następnie, po pokazaniu się zawartości danej grupy, wybrać odpowiedni obiekt.

Podobnie jak kontrolki na panelu, na diagramie umieszczamy ikony funkcji. Gdy chcemy umieścić ikonę w oknie diagramu, należy otworzyć okno *Functions*, wybierając z opcji *Window* menu polecenie *Functions Palette* lub przesunąć kursor na okno diagramu i nacisnąć prawy przycisk myszy. Okno *Functions* pokazane jest na rysunku 1.6.

Przyciski w górnej części okna służą do poszukiwania funkcji według nazwy (*Search*) i zmiany sposobu przedstawiania zawartości okna (*View*). Polecenie *Select a VI...* pozwala na wyszukiwanie przyrządów wirtualnych zapisanych na dysku.




Rys. 1.6. Okno *Functions*

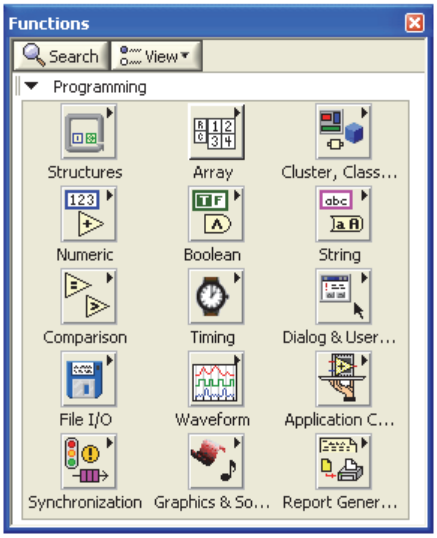
Ikony funkcji podzielone są na zbiory. Zawartość każdego zbioru można rozwinąć, klikając na odpowiednią strzałkę z lewej strony lub jego nazwę.

Zestaw *Programming* zawiera zestaw podstawowych funkcji wykorzystywanych przy przygotowaniu programu (rys. 1.7).

W poszczególnych grupach znajdują się następujące bloki funkcjonalne:

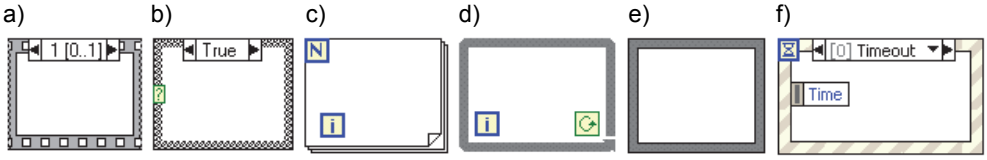
*Structures* – sekwencje, instrukcje wyboru, pętle, struktura do wpisywania wzorów, zmienne globalne i lokalne,
*Array* – operacje na macierzach,
*Cluster, Class, & Variant* – operacje na złożonych typach danych,
*Numeric* – operacje na liczbach,
*Boolean* – operacje na zmiennych typu logicznego,
*String* – operacje na łańcuchach znaków,
*Comparison* – operacje porównania,
*Timing* – funkcje związane z czasem,
*Dialog & User Interface* – funkcje związane z tworzeniem okien dialogowych,

*File I/O* – funkcje do obsługi plików,
*Waveform* – operacje na przebiegach sygnałów,
*Application Control* – funkcje do sterowania działaniem aplikacji,
*Synchronization* – funkcje do synchronizacji działania programu,
*Graphics & Sound* – funkcje związane z tworzeniem grafiki i dźwięków,
*Report Generation* – funkcje do tworzenia raportów.




Rys. 1.7. Grupy funkcji w zbiorze *Programming*

Szczególne znaczenie mają struktury umieszczone w grupie *Structures* (rys. 1.8). Umożliwiają one programowanie strukturalne, podobnie jak w klasycznych językach programowania.




Rys. 1.8. Struktury w grupie *Structures* okna *Functions*

Poszczególne struktury mają następujące zadania:

a) *Sequence* – umożliwia dokładne określenie kolejności wykonywania poszczególnych operacji (np. przy wykorzystaniu przyrządu pomiarowego najpierw należy zaprogramować jego funkcje, a dopiero potem odczytać wynik). W pierwszej kolejności wykonywane są operacje z ramki 0, następnie ramki 1 itd. Dodanie nowej ramki jest realizowane przez ustawienie kursora na krawędzi sekwencji, naciśnięcie prawego przycisku myszy i wybranie z menu polecenia *Add Frame After* (dodanie ramki za bieżącą) lub polecenia *Add Frame Before* (dodanie ramki przed bieżącą). Dostępne są dwa rodzaje tej struktury: *Stacked Sequence Structure* (wszystkie ramki w tym samym oknie, a w

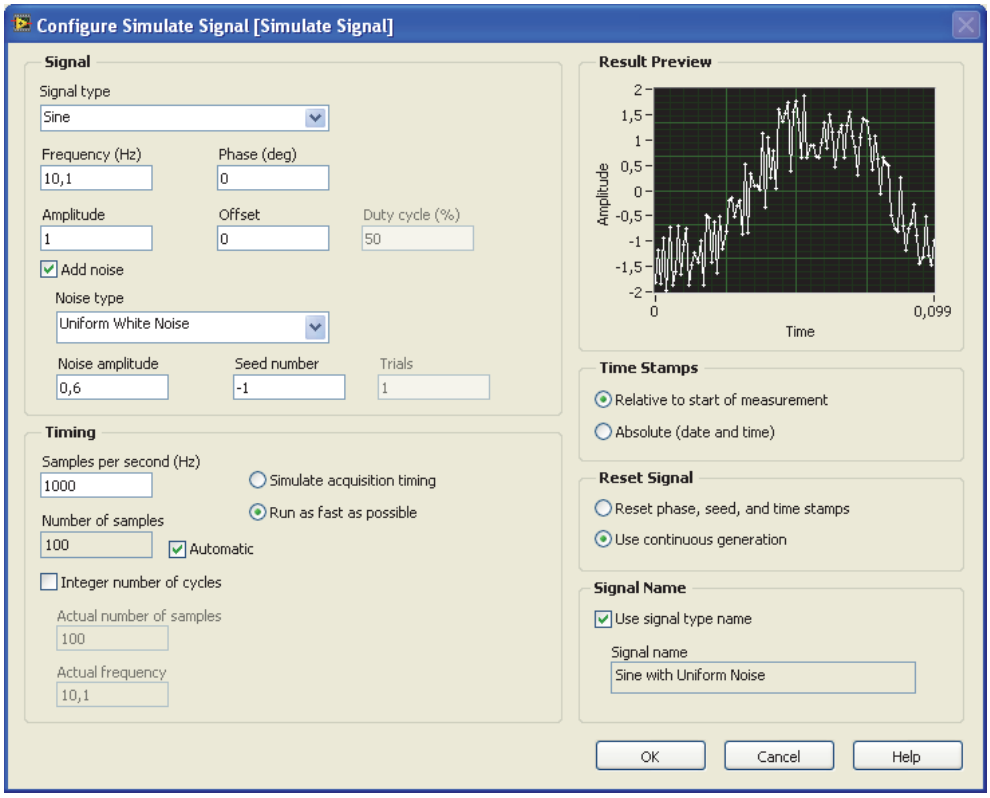
danej chwili pokazana tylko jedna ramka o wybranym numerze) i *Flat Sequence Structure* (kolejne ramki pokazane obok siebie).

b) *Case* – w zależności od wartości zmiennej dołączonej do zacisku w postaci znaku zapytania na krawędzi, wykonywany jest fragment programu umieszczony w jednej, wybranej ramce.
c) *For Loop* – umieszczona w tej ramce część programu jest wykonywana tyle razy, ile wynosi liczba dołączona do zacisku z literą „N" w lewym górnym rogu ramki. Sygnał dołączony do zacisku z literą „i" informuje o numerze aktualnie wykonywanej iteracji.
d) *While Loop* – umieszczona w tej ramce część programu jest wykonywana tak długo, aż zmienna dołączona do zacisku w postaci zakrzywionej strzałki przybierze wartość *False*. Znaczenie zacisku „i" jest takie samo jak w pętli *For Loop*. Ustawienie kursora na krawędzi ramki i naciśnięcie prawego przycisku myszy rozwija menu, w którym zamiast *Continue If True* można wybrać *Stop If True*. W tym przypadku część programu w ramce jest wykonywana tak długo, aż zmienna dołączona do zacisku w postaci czerwonego kółka przybierze wartość *True*.
e) *Formula Node* – blok służy do wprowadzania wzorów matematycznych, w przypadku, gdy nie chcemy do tego używać ikon z okna *Functions*. W celu wprowadzenia zmiennych wejściowych i wyjściowych, na krawędzi ramki należy umieścić zaciski przez ustawienie na niej kursora, naciśnięcie prawego przycisku myszy i wybranie z menu pozycji *Add Input* lub *Add Output*. Do utworzonego w ten sposób zacisku należy wpisać nazwę zmiennej, używaną we wzorach umieszczonych wewnątrz ramki.
f) *Event* – oczekiwanie na zdarzenie, które nastąpi na panelu programu, a następnie, w zależności od zdarzenia wykonywany jest fragment programu z odpowiedniej ramki (podobnie jak *Case*, tylko wykonanie nie zależy od wartości zmiennej, lecz od zdarzeń związanych z obsługą programu przez użytkownika). Struktura używana do synchronizacji wydarzenia na panelu z diagramem.

Złożone aplikacje mogą być przygotowane w prostszy sposób przez zastąpienie połączeń na diagramie wpisywaniem odpowiednich wartości w specjalnych oknach konfiguracji. Okna konfiguracji (nazywane jako *Express VI*) dostępne są w palecie *Functions*. Różnią się od klasycznych przyrządów wirtualnych niebieskim tłem.

Przykładowe okno do konfiguracji parametrów generowanego sygnału *Configure Simulate Signal* jest pokazane na rysunku 1.9. W oknie tym możemy wybrać kształt przebiegu, częstotliwość, fazę, amplitudę, offset, dodać szum, określić liczbę próbek na sekundę itd. Można też zobaczyć, jak wygląda wygenerowany przebieg.

Po zamknięciu okna przyciskiem OK, na diagramie otrzymamy symbol procedury z zaciskiem wyjściowym. Klikając dwa razy lewym przyciskiem myszy na symbol procedury, zostanie ponownie otwarte okno konfiguratora, umożliwiające zmianę zadanych parametrów.



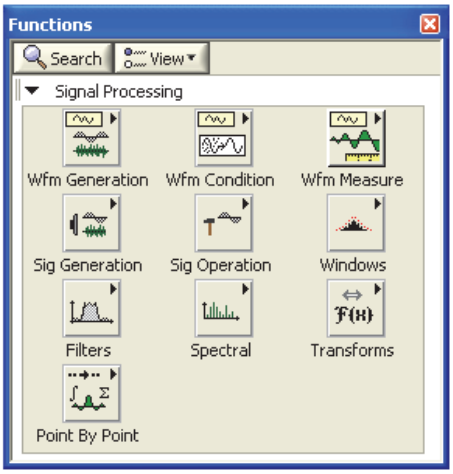

Rys. 1.9. Konfiguracja parametrów sygnału w oknie *Configure Simulate Signal*

## 1.2. Funkcje cyfrowego przetwarzania sygnałów dostępne w LabVIEW

Funkcje cyfrowego przetwarzania sygnałów dostępne w LabVIEW, znajdują się w zbiorze *Signal Processing* palety *Functions* (rys. 1.10).

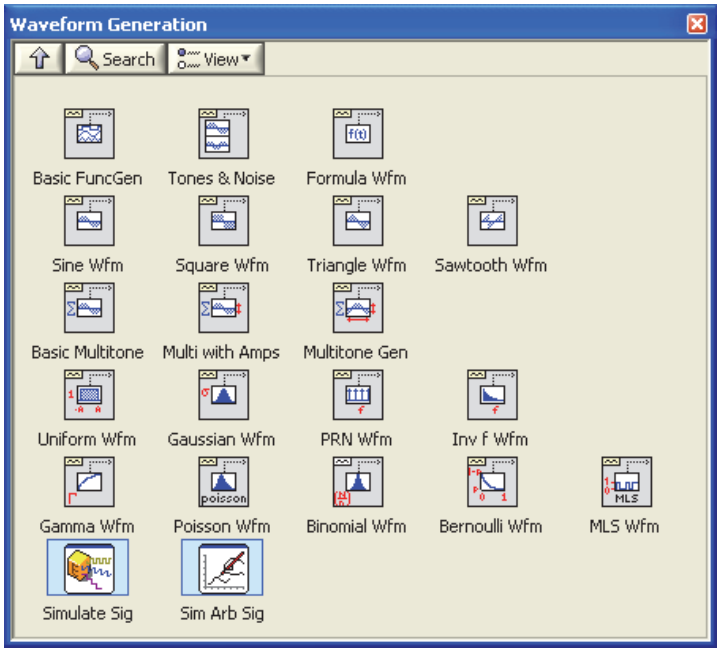
Podzielone są one na dziesięć grup:

*Waveform Generation* – generowanie sygnałów różnych rodzajów i kształtów (rys. 1.11),
*Waveform Conditioning* – kondycjonowanie danych (np. filtracja cyfrowa, skalowanie) (rys. 1.12),
*Waveform Measurements* – podstawowe funkcje przetwarzania przebiegów w dziedzinie czasu i częstotliwości (rys. 1.13),
*Signal Generation* – generowanie jednowymiarowych tablic z przebiegami o różnych kształtach (rys. 1.14),
*Signal Operation* – operacje na sygnałach (np. splot, korelacja) (rys. 1.15),
*Windows* – okna wygładzające (rys. 1.16),
*Filters* – filtracja cyfrowa (rys. 1.17),
*Spectral* – analiza widmowa (rys. 1.18),
*Transforms* – transformaty (m.in. Fouriera, Hilberta, Laplace'a) (rys. 1.19),
*Point By Point* – analiza sygnałów punkt po punkcie (rys. 1.20).

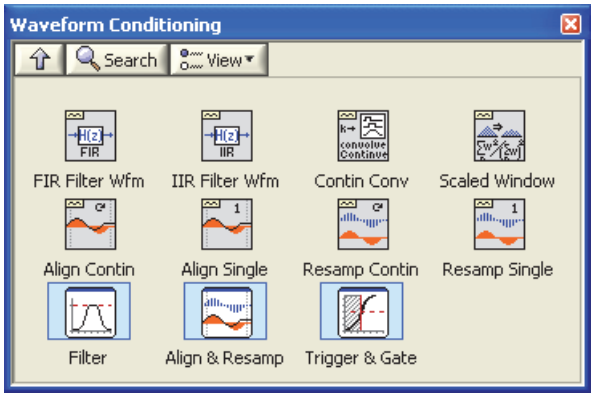



Rys. 1.10. Grupy funkcji w zbiorze *Signal Processing*

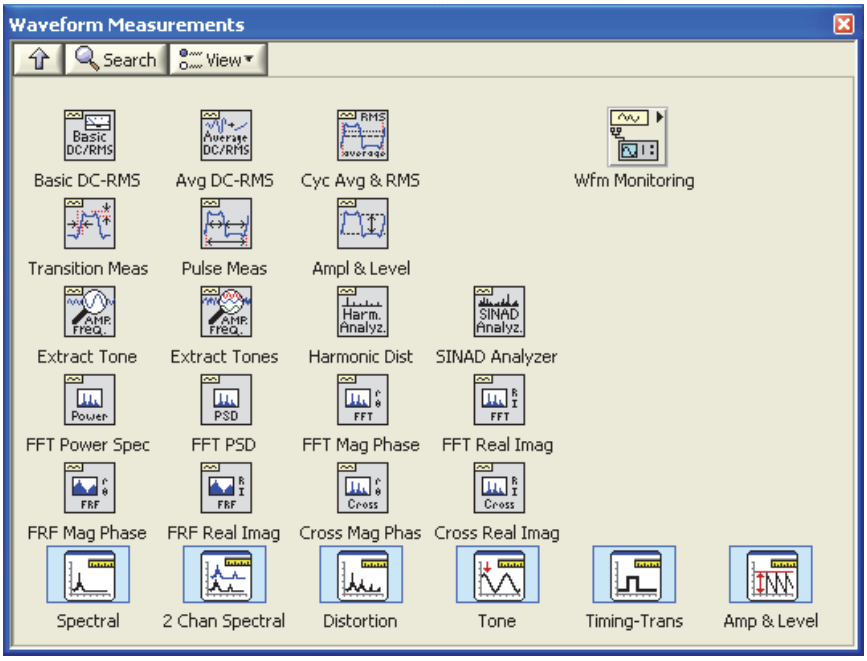
Funkcje w grupie *Point By Point* są odpowiednikami funkcji w pozostałych grupach przetwarzania sygnałów. Wykorzystujemy je, gdy przetwarzanie wykonywane jest na bieżąco, w trakcie wykonywania pomiarów, czyli gdy w kolejnych krokach programu dochodzą pojedyncze próbki. Pozostałe funkcje wymagają całego bloku danych podlegającego przetworzeniu, czyli można je wykorzystać, gdy mamy już wszystkie wyniki po zakończonych pomiarach. Ikony funkcji *Point By Point* mają dodatkowe oznaczenie w postaci czerwonego symbolu ●→●.



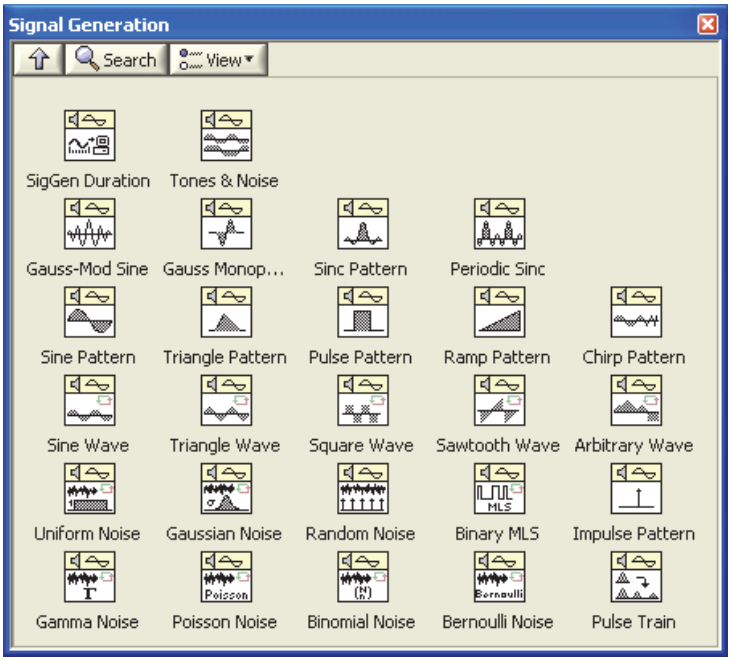

Rys. 1.11. Funkcje *Waveform Generation*



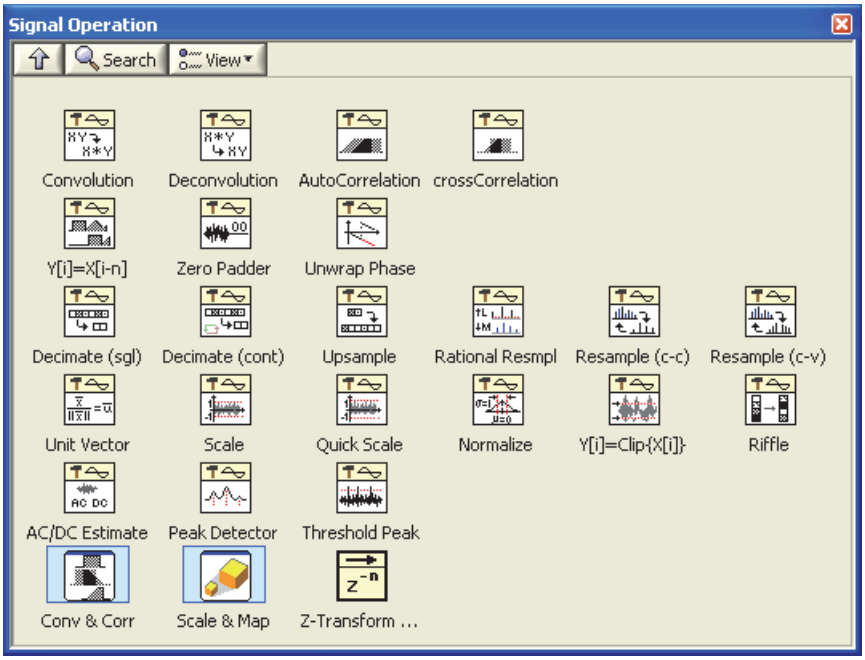

Rys. 1.12. Funkcje *Waveform Conditioning*



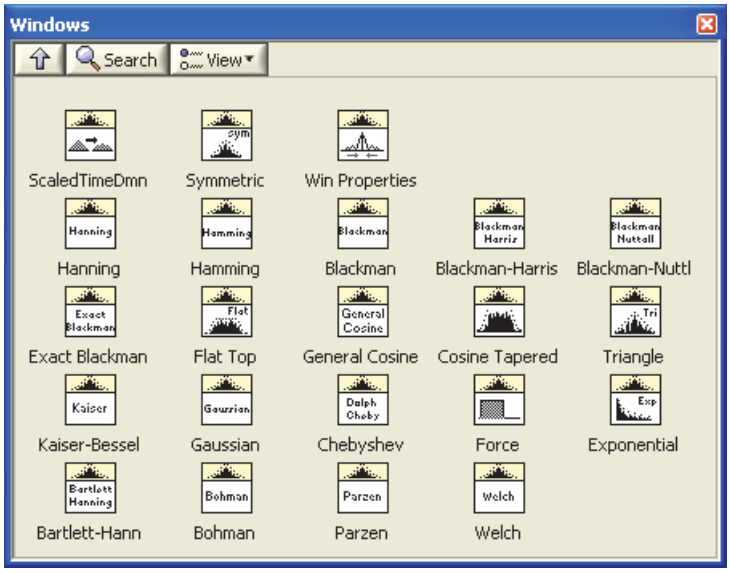

Rys. 1.13. Funkcje *Waveform Measurements*



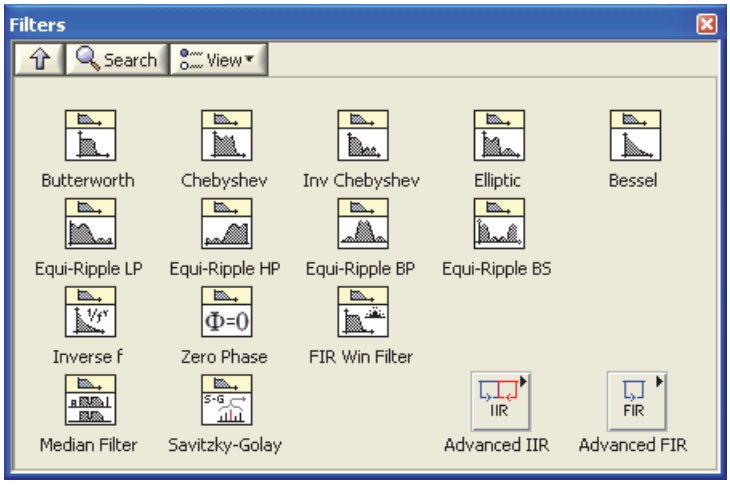

Rys. 1.14. Funkcje *Signal Generation*



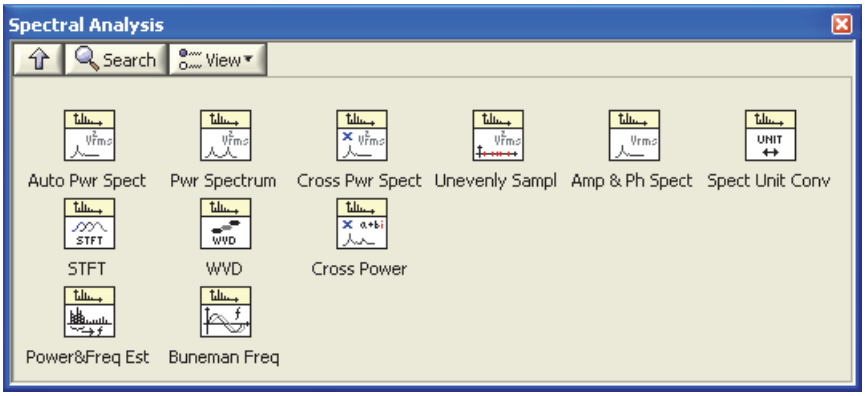

Rys. 1.15. Funkcje *Signal Operation*



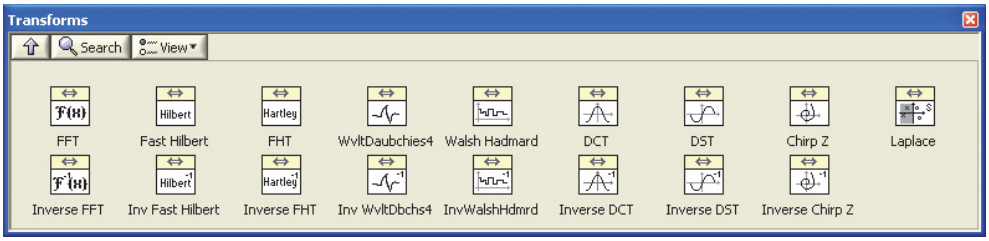

Rys. 1.16. Funkcje *Windows*



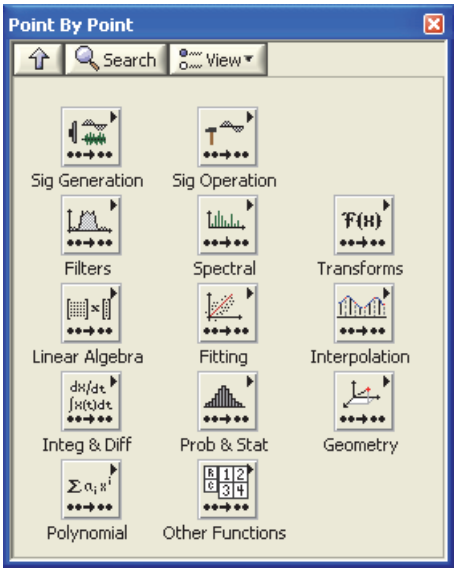

Rys. 1.17. Funkcje *Filters*




Rys. 1.18. Funkcje *Spectral Analysis*




Rys. 1.19. Funkcje *Transforms*




Rys. 1.20. Funkcje *Point By Point*

## 1.3. Konfiguracja elementów regulacyjnych i wskaźników

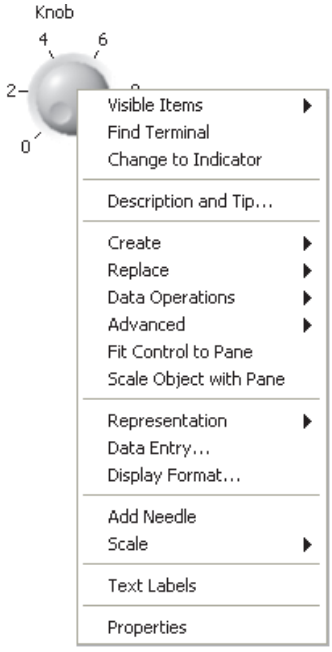
Każdy element umieszczony na panelu możemy zmienić i skonfigurować odpowiednio do naszych potrzeb.

Jeżeli chcemy zmienić wielkość obiektu znajdującego się na panelu, należy w oknie *Tools* wybrać narzędzie w postaci strzałki. Strzałkę ustawiamy w jednym z czterech rogów obiektu, a następnie po naciśnięciu lewego przycisku myszy zmieniamy położenie tak, by obiekt osiągnął wymaganą wielkość.

Jeżeli chcemy zmienić kolor obiektu, z okna *Tools* wybieramy pędzelek. Ustawiamy pędzelek na wybranym obiekcie i naciskamy prawy przycisk myszy. Następnie z palety kolorów należy wybrać odpowiednią barwę. Jeżeli chcemy zmienić kolor obiektu na taki, jaki jest w kwadraciku w dolnej części okna *Tools*, to po ustawieniu pędzelka na ten obiekt wystarczy nacisnąć lewy przycisk myszy.

Jeżeli umieszczonym na panelu obiektem jest np. pokrętło, to domyślnie umożliwia ono zmianę danej wejściowej w zakresie od 0 do 10. W celu zmiany tego zakresu, z okna *Tools* należy wybrać kursor w postaci ręki z wyciągniętym palcem lub litery A, kursorem tym zaznaczyć liczbę na początku lub końcu zakresu i wpisać z klawiatury nową wartość. Zmiana zostanie wykonana po ustawieniu kursora na panelu poza obiektem i naciśnięciu lewego przycisku myszy lub naciśnięciu przycisku *Enter* na pasku przycisków.

Zmiany w konfiguracji obiektu można wprowadzić po ustawieniu kursora na tym obiekcie i naciśnięciu prawego przycisku myszy. Pojawia się menu mogące się różnić dla różnych obiektów. Jako przykład na rysunku 1.21 przedstawiono menu dla pokrętła.




Rys. 1.21. Menu konfiguracji pokrętła

Z menu możemy wybrać odpowiednią pozycję:

*Visible Items* – do pokazania lub ukrycia elementów związanych z pokrętłem, np. etykiety obiektu lub dodatkowego wskaźnika cyfrowego,
*Find Terminal* – odszukuje na diagramie zacisk danego obiektu z panelu,
*Change to Idicator* – zamiana pokrętła z elementu wejściowego na wyjściowy (wskaźnik),
*Description and Tip* – utworzenie opisu obiektu, pokazywanego w oknie *Context Help*,
*Create* – umieszczenie na diagramie dodatkowych zacisków związanych z obiektem, np. zmiennej lokalnej lub „*Property Node*" umożliwiającego zmianę właściwości danego obiektu w czasie wykonywania programu,
*Replace* – zamiana obiektu na inny, wybrany z okna *Controls*,
*Data Operations* – umożliwia przyjęcie aktualnego ustawienia jako początkowego, zmianę nastawienia na początkowe itd.,
*Advanced* – ustawienia zaawansowane, np. wybór klawisza, na naciśnięcie którego będzie reagował dany obiekt, ukrycie obiektu itd.,
*Fit Control to Pane* – dopasowanie obiektu do wielkości okna,
*Scale Object with Pane* – pokazanie linii wyznaczających położenie obiektu,
*Representation* – zmiana reprezentacji liczby,
*Data Entry* – zmiana zakresu danych,
*Display Format* – zmiana formatu i liczby miejsc po przecinku,
*Add Needle* – dodanie nowej wskazówki do pokrętła,
*Scale* – zmiany skali pokrętła,
*Text Labels* – zmiana skali z liczbowej na tekstową,
*Properties* – ustawienie wyglądu obiektu.

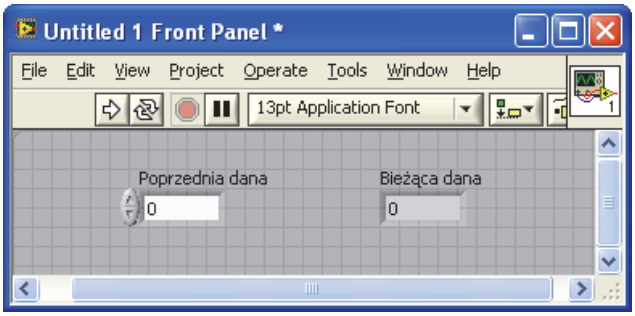
## 1.4. Tworzenie aplikacji przy wykorzystaniu LabVIEW

Wykorzystanie LabVIEW zostanie omówione na przykładzie przyrządu wirtualnego realizującego następujące funkcje:

— generowanie danych pomiarowych,
— filtracja dolnopasmowa danych pomiarowych,
— graficzne zobrazowanie danych na wejściu i wyjściu filtra dolnoprzepustowego.

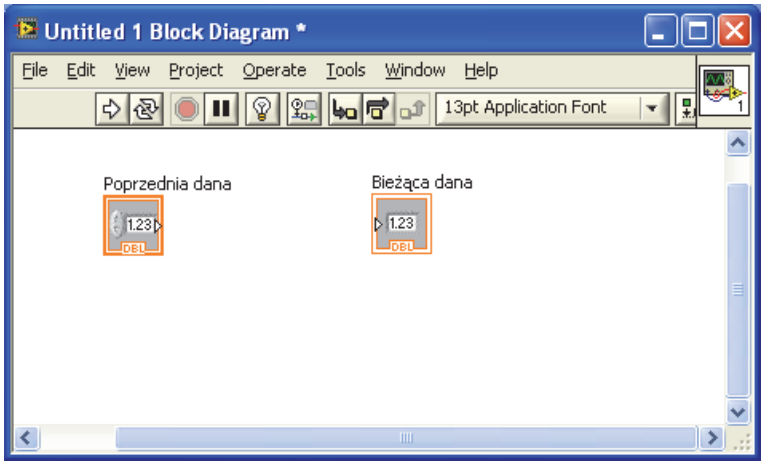
Do generowania danych pomiarowych zostanie wykorzystany generator liczb pseudolosowych o rozkładzie równomiernym. Generowane liczby będą normalizowane do przedziału wartości od –0,5 do 0,5. Kolejny element ciągu danych pomiarowych będzie tworzony jako suma poprzedniego elementu ciągu i unormowanej liczby pseudolosowej.

Na panelu podprogramu generowania danych umieszczamy dwa obiekty ze zbioru *Express* okna *Controls*: *Numeric Control* z grupy *Numeric Controls* do zobrazowania poprzedniej danej pomiarowej oraz *Numeric Indicator* z grupy *Numeric Indicators* do wyświetlenia bieżącej danej pomiarowej. W polu etykiety obu obiektów wpisujemy odpowiednie nazwy. Panel przyrządu wirtualnego powinien wyglądać jak na rys. 1.22.




Rys. 1.22. Panel przyrządu wirtualnego do generowania danych

Umieszczenie obiektu na panelu powoduje automatyczne umieszczenie ikony symbolizującej zacisk tego obiektu na diagramie (rys. 1.23).

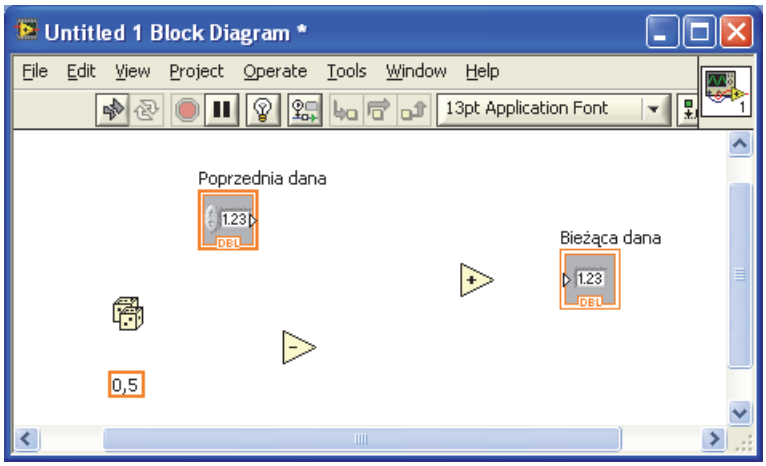



Rys. 1.23. Diagram przyrządu wirtualnego do generowania danych po umieszczeniu obiektów na panelu

Obiekty służące do wprowadzania danych do programu (*Controls*) mają ikonę z pogrubioną zewnętrzną ramką i strzałką po prawej stronie, obiekty do odbioru danych z programu (*Indicators*) z cienką zewnętrzną ramką i strzałką po lewej stronie. Kolor ramki informuje o typie zmiennej, np. pomarańczowy dla liczb rzeczywistych, granatowy dla całkowitych, zielony dla zmiennej typu logicznego, różowy dla zmiennej tekstowej. Rysunek wewnątrz zacisku informuje o rodzaju obiektu na panelu, natomiast znajdujący się poniżej symbol o reprezentacji zmiennej (DBL oznacza liczbę podwójnej precyzji). Ustawiając kursor na ikonie, naciskając prawy przycisk myszy i odznaczając pozycję *View as Icon,* można przekształcić ikonę w zacisk o wyglądzie małego prostokąta. Grubość zewnętrznej ramki tego prostokąta, kolor oraz znajdujący się we wnętrzu symbol mają analogiczne znaczenie jak dla ikony.

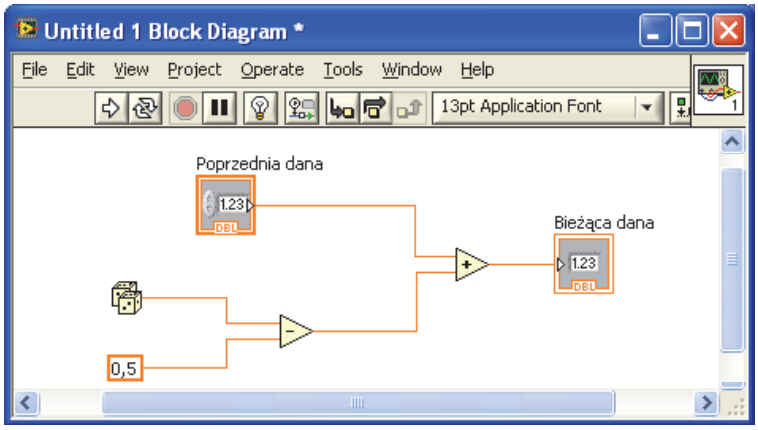
Do generowania liczb pseudolosowych wykorzystamy funkcję *Random Number*, znajdującą się w zbiorze *Express* okna *Functions*, w grupie *Arithmetic & Comparison* → *Express Numeric*. Ikona funkcji *Random Number* przedstawia dwie kostki do gry. Funkcja ta generuje liczby z przedziału od 0 do 1. W celu normalizacji wartości generowanych liczb

do przedziału od –0,5 do 0,5, od liczby wygenerowanej przez *Random Number* należy odjąć 0,5. Do operacji odejmowania (normalizacja liczb pseudolosowych) i dodawania (kolejna liczba ciągu danych pomiarowych) można wykorzystać funkcje *Substract* oraz *Add* z okna *Functions*. Natomiast stałą 0,5 uzyskamy, wybierając *Numeric Constant* i wpisując odpowiednią wartość. Po umieszczeniu funkcji diagram powinien wyglądać jak na rysunku 1.24.




Rys. 1.24. Diagram przyrządu wirtualnego do generowania danych po umieszczeniu na nim pozostałych funkcji

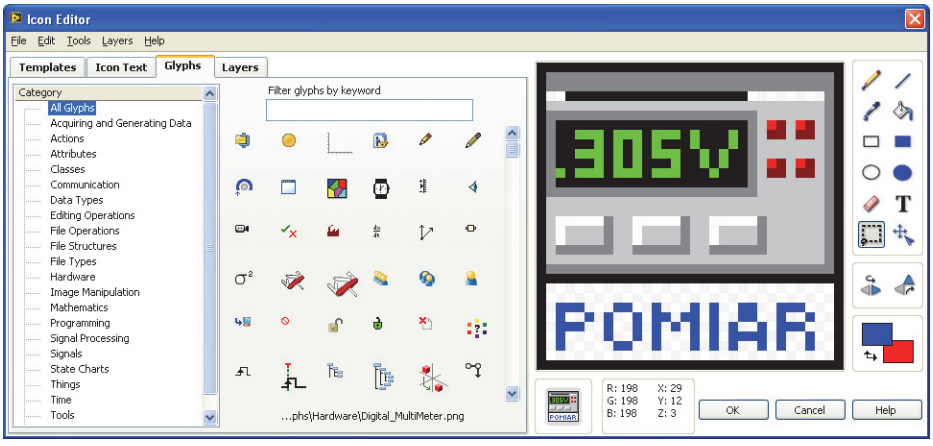
Poszczególne elementy na diagramie można połączyć, wybierając z okna *Tools* narzędzie w postaci szpulki. Wystający koniec nitki kierujemy na element, który chcemy połączyć. Jeżeli jest to ikona obiektu na panelu, powinna migać jej ramka, jeżeli jest to funkcja lub przyrząd wirtualny, powinny pokazać się krótkie odcinki przewodów symbolizujące zaciski, a migać będzie część elementu związana z zaciskiem, na który skierowaliśmy szpulkę. Następnie naciskamy i zwalniamy lewy przycisk myszy i kierujemy kursor na zacisk, do którego chcemy podłączyć element wybrany wcześniej. Połączenie będzie wykonane po naciśnięciu i zwolnieniu lewego przycisku myszy. Kolor i grubość linii informują o typie przesyłanych przez nią danych (np. pomarańczowy dla liczb rzeczywistych, granatowy dla całkowitych, zielony dla zmiennych typu logicznego, różowy dla zmiennych tekstowych, linia pogrubiona w przypadku tablicy). Jeżeli połączenie zostało wykonane błędnie, zaznaczone jest linią przerywaną. Wszystkie błędne połączenia można usunąć, wybierając polecenie *Remove Broken Wires* z opcji *Edit* menu lub naciskając klawisze Ctrl-B. Po wykonaniu wszystkich połączeń diagram powinien wyglądać jak na rysunku 1.25.




Rys. 1.25. Diagram przyrządu wirtualnego do generowania danych po wykonaniu połączeń

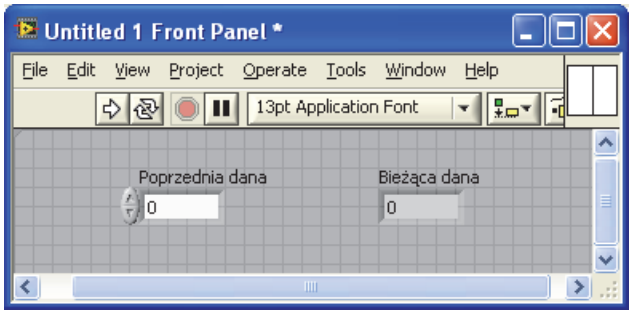
Jeżeli w diagramie jest błąd, np. nie jest podłączony sygnał na wejście, które wymaga połączenia, strzałka na przycisku do uruchomienia programu jest złamana (patrz rys. 1.24). Jeżeli mimo to naciśniemy ten przycisk, wprawdzie program nie będzie wykonany, ale otrzymamy wykaz błędów występujących w diagramie. Przez ustawienie kursora na wybranym błędzie i naciśnięcie przycisku myszy podany zostanie opis błędu i zaznaczony błędny element na diagramie. Pomoże to przy usuwaniu błędów.

Dla przyrządu wirtualnego, który będzie wykorzystany jako podprogram, należy utworzyć ikonę. W tym celu ustawiamy kursor na kwadracie w prawym górnym rogu i wciskamy prawy przycisk myszy. Z rozwiniętego menu wybieramy pozycję *Edit Icon...*, a po zwolnieniu przycisku myszy powinno otworzyć się okno do edycji ikony przyrządu wirtualnego (*Icon Editor*). W prawej części tego okna znajduje się kwadrat z domyślną ikoną. Po jej skasowaniu, korzystając z prostego programu graficznego, możemy utworzyć własną ikonę. Do tego celu można skorzystać z gotowych elementów graficznych. Na rysunku 1.26 pokazane jest okno *Icon Editor* z przygotowywaną ikoną. Po zakończeniu rysowania naciskamy *OK*, zamykając okno i powracając do panelu. Utworzony rysunek powinien pojawić się w kwadracie w prawym górnym rogu.




Rys. 1.26. Edycja ikony przyrządu wirtualnego

Można przystąpić teraz do zdefiniowania zacisków. W tym celu kursor ponownie ustawiamy na kwadracie w prawym górnym rogu panelu i naciskamy prawy przycisk myszy. Po wybraniu pozycji *Show Connector* zwalniamy przycisk myszy. W miejscu ikony pojawia się kwadrat podzielony na pola. Liczba pól zależy od liczby parametrów wejściowych i wyjściowych przyrządu wirtualnego. Zwykle do zacisków z lewej strony podłączane są parametry wejściowe, do zacisków z prawej strony – wyjściowe. Jeżeli chcemy zmienić liczbę zacisków lub ich rozkład, należy ustawić kursor na zaciskach, nacisnąć prawy przycisk myszy, z menu wybrać pozycję *Patterns*, a następnie wybrać odpowiedni wzór zacisków. Pojedynczy zacisk można dodać przez polecenie *Add Terminal*, a usunąć przez polecenie *Remove Terminal*. Zaciski można również obrócić (*Rotate 90 Degrees*) lub wykonać ich lustrzane odbicie (*Flip Horizontal*, *Flip Vertical*).




Rys. 1.27. Panel przyrządu wirtualnego przygotowany do zdefiniowania połączeń

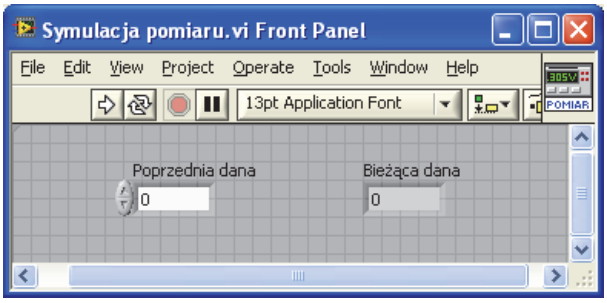
W naszym przykładzie występuje jeden parametr wejściowy i jeden wyjściowy, dlatego wybieramy zacisk podzielony na dwa pola (rys. 1.27). Aby przypisać obiekty na panelu poszczególnym polom, należy kliknąć na wybrane pole (białe pole powinno zmienić swój kolor na czarny), następnie kliknąć na obiekcie na panelu, który odpowiada polu z kwadratu. Czarne pole powinno zmienić swój kolor na odpowiadający typowi zmiennej, np. pomarańczowy dla liczb rzeczywistych. Czynności te należy powtórzyć dla każdego pola.

Do pola z lewej strony przypisujemy element wejściowy „Poprzednia dana", do pola z prawej strony przypisujemy element wyjściowy „Bieżąca dana". Po zdefiniowaniu wszystkich zacisków możemy ponownie ustawić kursor na kwadracie i po naciśnięciu prawego przycisku myszy wybrać pozycję *Show Icon*.

Ostatnim etapem jest zapisanie przyrządu wirtualnego do pliku na dysku. Jeżeli zapisujemy po raz pierwszy, z opcji menu *File* wybieramy polecenie *Save As...*, w otwartym oknie wybieramy folder. Nowa nazwa powinna pojawić się na pasku tytułu w górnej części okna panelu i diagramu.

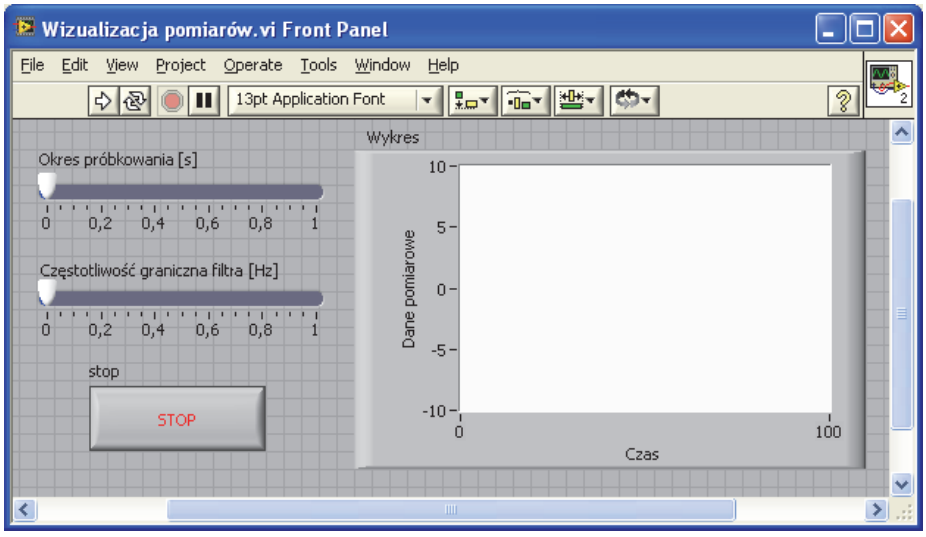
Jeżeli plik zapisaliśmy jako „Symulacja pomiaru", okno panelu powinno wyglądać jak na rysunku 1.28. W przyszłości, gdy będziemy zapisywali program do pliku bez zmiany jego nazwy, z opcji *File* należy wybrać polecenie *Save* lub nacisnąć klawisze Ctrl-S.

Poprawność działania programu można sprawdzić przez wpisanie dowolnej liczby w polu „Poprzednia dana" i kilkukrotne wciśnięcie strzałki do uruchomienia programu. Liczba podawana jako „Bieżąca dana" może się różnić od liczby w polu „Poprzednia dana" o nie więcej niż 0,5. Gotową, sprawdzoną i zapisaną do pliku procedurę zamykamy poleceniem *Close* z grupy *File* lub przez naciśniecie przycisku zamykania x w prawym górnym rogu okna panelu.




Rys. 1.28. Panel przyrządu wirtualnego po zapisaniu do pliku pod nową nazwą

Można teraz przystąpić do przygotowania programu głównego. W tym celu otwieramy okno nowego panelu przez wybranie z opcji *File* menu polecenia *New VI* lub przez polecenie *New → BlankVI* w oknie *Getting Started*. Na panelu umieszczamy wykres *Waveform Chart* do przedstawienia zmian mierzonej wielkości, dwa suwaki *Pointer Slide* do nastawienia czasu między kolejnymi pomiarami i częstotliwości granicznej filtru oraz przycisk STOP do zatrzymania wykonywania programu. Dla każdego obiektu należy wpisać jego etykietę. Zakres zmian wartości nastawianej suwakiem i zakres zmian wielkości przedstawianej na wykresie możemy zmienić tak, jak już opisano w rozdziale 1.3, ustawiając kursor (w postaci wyciągniętego palca lub litery A) na wybranej liczbie, naciskając lewy przycisk myszy i wpisując nową wartość. Panel przyrządu wirtualnego po zapisaniu do pliku pod nazwą „Wizualizacja pomiarów" powinien wyglądać podobnie jak na rysunku 1.29.




Rys. 1.29. Panel przyrządu wirtualnego do graficznej prezentacji danych pomiarowych

Można teraz przystąpić do edycji diagramu. Ponieważ program powinien być wykonywany w pętli, skorzystamy z opisanej wcześniej pętli *While Loop*. Ramkę pętli należy umieścić na diagramie tak, by w jej wnętrzu znalazły się będące już na diagramie zaciski. Jeżeli chcemy, by program umieszczony wewnątrz ramki po uruchomieniu był wykonywany w pętli do chwili naciśnięcia przycisku STOP, w konfiguracji pętli należy wybrać pozy-

cję *Stop If True*. Zacisk odpowiadający przyciskowi STOP łączymy z zaciskiem w postaci czerwonego kółka w prawym dolnym rogu ramki.

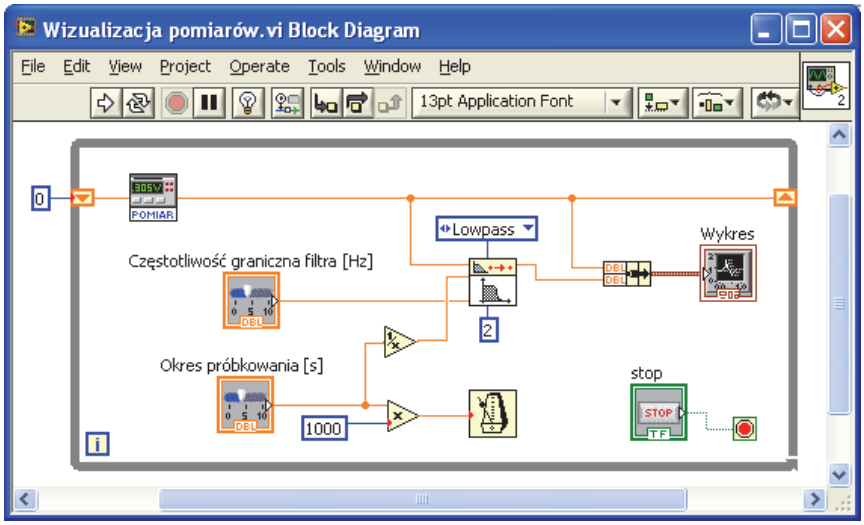
Przesuwając kursor na lewą lub prawą krawędź ramki pętli, naciskając prawy przycisk myszy i wybierając z menu pozycję *Add Shift Register*, możemy umieścić na ramce zaciski rejestrów wejściowego i wyjściowego pętli. Wartość przekazana do zacisku wyjściowego pętli (z symbolem ▲, przy prawej krawędzi ramki) zostanie przekazana do zacisku wejściowego (z symbolem ▼, przy lewej krawędzi ramki) przy następnym wykonywaniu pętli. Dzięki temu aktualnie zmierzoną wartość możemy przy następnym wykonywaniu pętli wykorzystać jako wartość poprzednią. W trakcie pierwszego wykonywania pętli do rejestru wejściowego zostaje pobrana wartość z zacisku na zewnątrz ramki, dlatego możemy podłączyć do niego wartość inicjującą, np. 0.

W ramce pętli należy umieścić ikonę przygotowanego wcześniej przyrządu wirtualnego „Symulacja pomiaru.vi". Można to zrobić przez wybranie z okna *Functions* pozycji *Select a VI...*, wyszukanie folderu, do którego zapisaliśmy przyrząd wirtualny i wybranie odpowiedniego pliku. Wyjście wstawionej ikony łączymy z zaciskiem wyjściowym *Shift Register*. Jej wejście łączymy z zaciskiem wejściowym *Shift Register*.

Do filtracji możemy wykorzystać filtr Butterwortha, znajdujący się w *Functions* → *Signal Processing* → *Point by Point* → *Filters*. Do wejścia *x* filtra dołączamy sygnał wyjściowy z bloku *Symulacja pomiaru.vi*. Następnie, łącząc odpowiednie ikony z zaciskami filtra, wybieramy jego rodzaj (filtr dolnoprzepustowy) oraz rząd (np. 2). Zacisk częstotliwości granicznej filtra łączymy z ikoną zacisku suwaka, zaś zacisk częstotliwości próbkowania (*sampling frequency*) poprzez blok inwersji łączymy z ikoną zacisku suwaka regulacji okresu próbkowania.

W celu równoczesnego zobrazowania dwóch przebiegów danych pomiarowych, tj. przed i po ich filtracji należy użyć funkcję *Bundle* (*Functions* → *Programming* → *Cluster, Class, & Variant*).

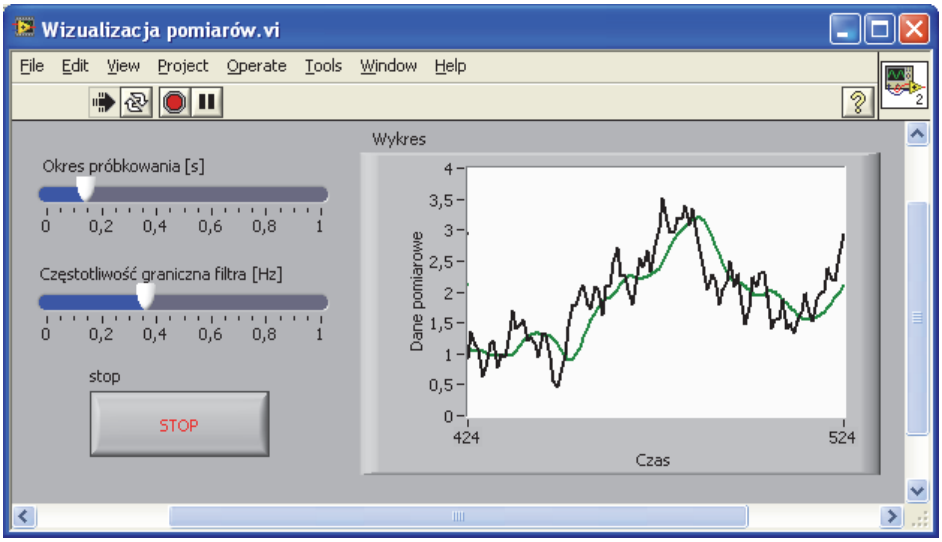
Spowolnienie działania pętli jest możliwe przez wykorzystanie funkcji opóźnienia *Wait Until Next ms Multiple* (*Functions* → *Programming* → *Timing*). Na wejście tej funkcji należy podać okres czasu między kolejnymi wykonaniami pętli w milisekundach. Jeżeli pokrętłem na panelu ustawiamy czas w sekundach, mnożymy go przez 1000.




Rys. 1.30. Diagram przyrządu wirtualnego do graficznej prezentacji danych pomiarowych

Po wykonaniu opisanych połączeń diagram powinien wyglądać jak na rys. 1.30. Program możemy teraz zapisać do pliku i uruchomić.

Panel programu po uruchomieniu przedstawiony jest na rysunku 1.31. Program można wykorzystać do sprawdzenia, jaki wpływ na przebieg odfiltrowanego sygnału ma zadana częstotliwość graniczna. Częstotliwość ta musi być nastawiona przed uruchomieniem programu. Po każdorazowej zmianie częstotliwości granicznej należy zatrzymać program i ponownie uruchomić, w przeciwnym wypadku nowa wartość nie zostanie uwzględniona.



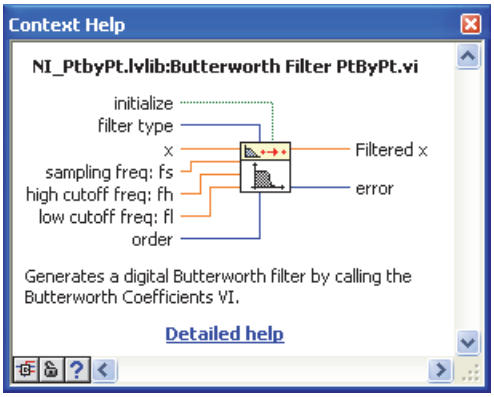

Rys. 1.31. Panel programu po uruchomieniu

Przygotowany program można rozbudować o możliwość nastawiania wartości maksymalnej i minimalnej, przekroczenie których sygnalizowane będzie zapaleniem wskaźników. Wartości maksymalna i minimalna mogą być nastawiane pokrętłami, a ich wartości przedstawione na tym samym wykresie, co zmiany wartości mierzonej.

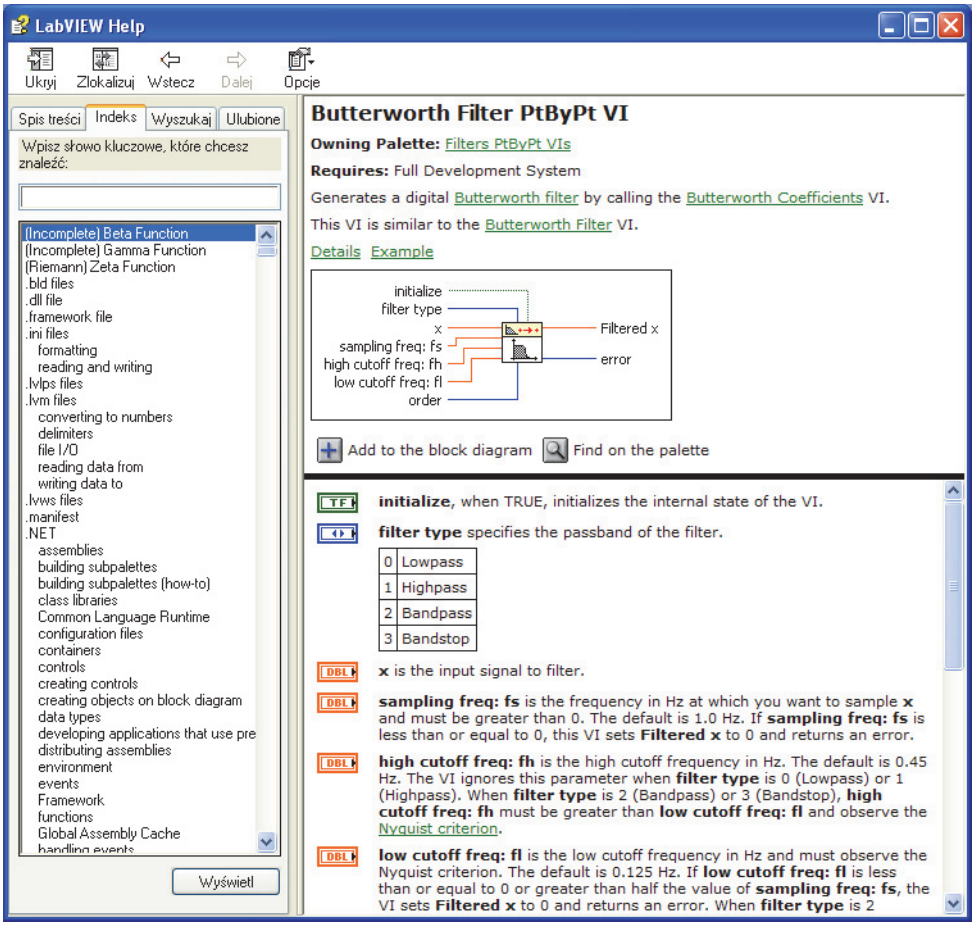
## 1.5. Dodatkowa pomoc przy pisaniu i uruchamianiu programu w LabVIEW

W czasie pracy z LabVIEW można korzystać z rozbudowanego systemu pomocy. Jeżeli z opcji *Help* menu wybierzemy polecenie *Show Context Help*, naciśniemy klawisze Ctrl-H lub na pasku przycisków narzędziowych naciśniemy przycisk ze znakiem zapytania, to otworzy się okno pomocy kontekstowej.

Jeżeli będziemy w oknie diagramu i ustawimy kursor na umieszczonym tam elemencie, w oknie pomocy pojawi się opis tego elementu (rys. 1.32). Jest to użyteczne szczególnie przy wykorzystaniu przyrządów wirtualnych, gdy w oknie pomocy znajduje się opis zacisków tego przyrządu.



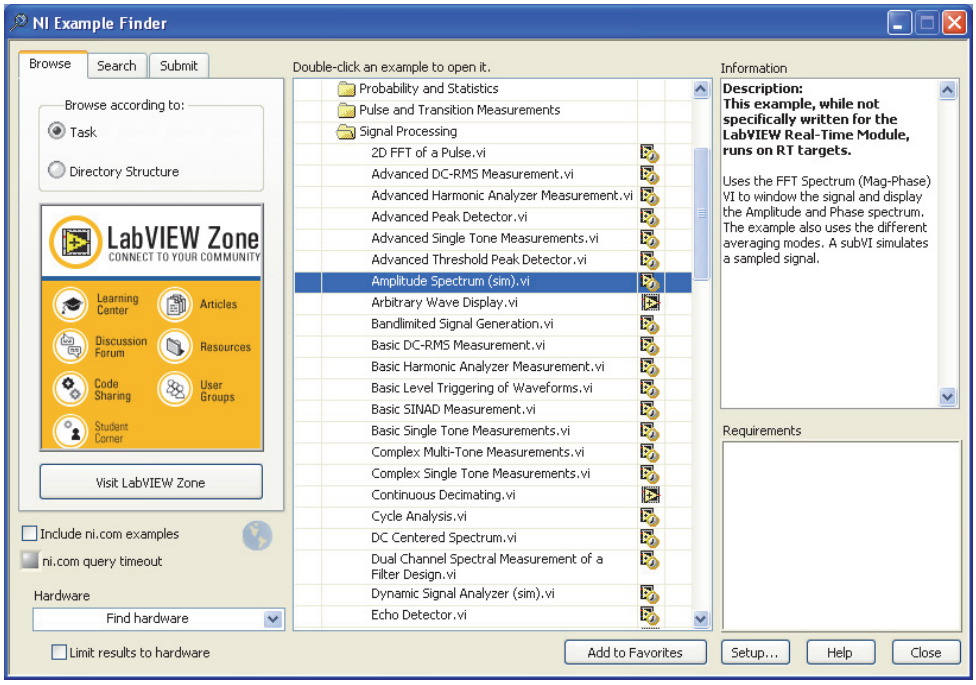

Rys. 1.32. Przykładowe okno pomocy kontekstowej dla funkcji *Butterworth Filter PtByPt*




Rys. 1.33. *LabVIEW Help* z opisem funkcji *Butterworth Filter PtByPt*

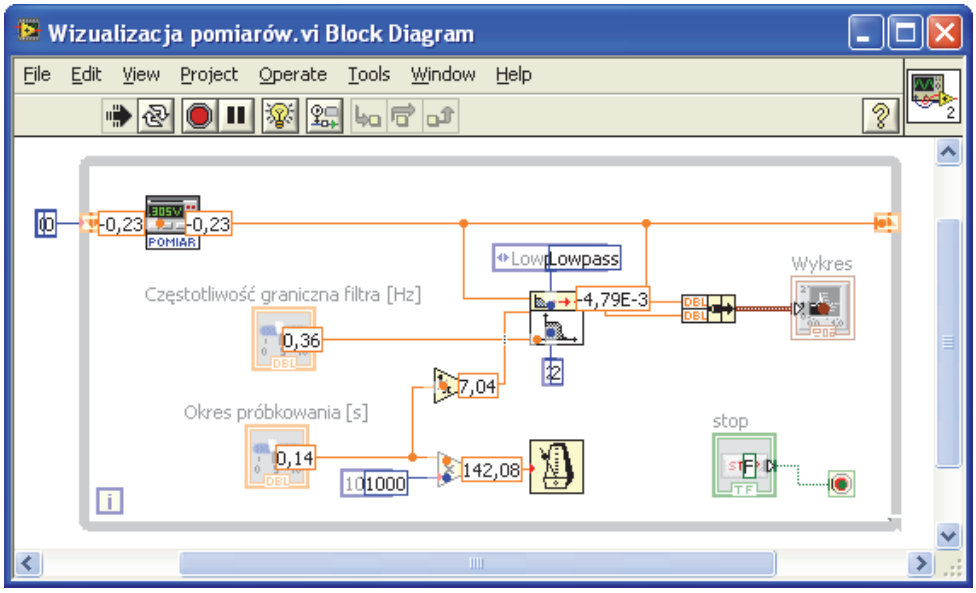
Jeżeli z opcji *Help* menu wybierzemy polecenie *Search the LabVIEW Help...* lub naciśniemy klawisze Ctrl-?, to otworzy się okno *LabVIEW Help*, zawierające instrukcję do programu LabVIEW. Naciskając przycisk ze znakiem zapytania w dolnej części okna *Context Help*, również przejdziemy do podręcznika LabVIEW, ale bezpośrednio do miejsca opisującego przyrząd wirtualny znajdujący się w oknie *Context Help* (rys. 1.33).

Wybierając z opcji *Help* menu polecenie *Find Examples*, można skorzystać z przykładów gotowych programów przygotowanych przy użyciu LabVIEW. Przykłady podzielone są na grupy tematyczne, dzięki czemu łatwo znaleźć potrzebne w danej chwili (rys. 1.34). Przykłady pokazują wykorzystanie przyrządów wirtualnych, a fragmenty przykładów można skopiować bezpośrednio do naszego programu. Przygotowując bardziej zaawansowane aplikacje, warto tam zaglądać, ponieważ nie ma potrzeby tracić czasu na wymyślanie rozwiązań, które zostały już zrobione przez innych.




Rys. 1.34. Okno z listą przykładowych programów przetwarzania sygnałów

Z szeregu ułatwień można również korzystać przy uruchamianiu programu w LabVIEW. Naciskając na pasku przycisków narzędziowych diagramu przycisk z żarówką, możemy uruchomić program z animacją przedstawiającą aktualnie wykonywaną operację. Pokazane jest to w postaci przesuwających się kropek, a na wyjściu poszczególnych elementów na diagramie podana jest aktualna wartość sygnału (rys. 1.35).




Rys. 1.35. Diagram po uruchomieniu z zaznaczeniem aktualnie wykonywanej operacji

Uruchamiany program może być wykonywany w pojedynczych krokach. W tym celu na pasku przycisków narzędziowych diagramu należy nacisnąć przycisk *Start Single Stepping*. W zależności od tego, który przycisk na pasku zostanie naciśnięty w kolejnych krokach, przy napotkaniu przyrządu wirtualnego przejdziemy do tego przyrządu lub jego program zostanie wykonany w pojedynczym kroku.

Wybierając z okna *Tools* narzędzie *Set/Clear Breakpoint,* można w dowolnym miejscu diagramu ustawić punkty przerwania, tzn. punkty, po dojściu do których wykonywanie programu jest zatrzymane. Po zatrzymaniu wykonywanie programu może być przerwane lub dalej kontynuowane.

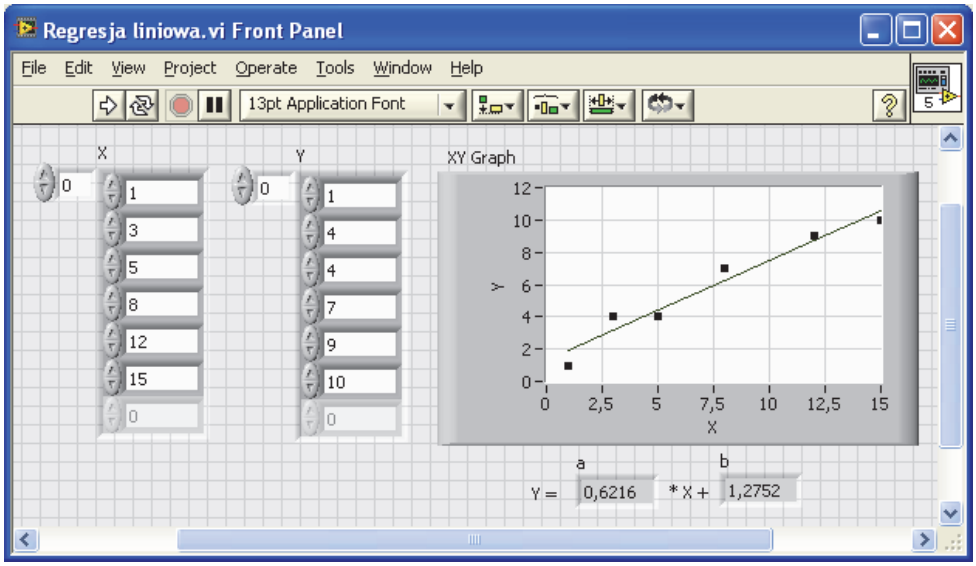
Wybierając z okna *Tools* narzędzie *Probe Data*, można w dowolnym miejscu diagramu ustawić sondę podającą wartość sygnału w tym punkcie, w czasie wykonywania programu. Okno *Probe* jest widoczne, gdy aktywnym oknem jest nie tylko diagram, ale także panel.

## 1.6. Przykłady programów

Programy do cyfrowego przetwarzania sygnałów ze szczegółowym opisem przedstawione są w kolejnych rozdziałach. Poniżej zamieszczono opis czterech prostych aplikacji.

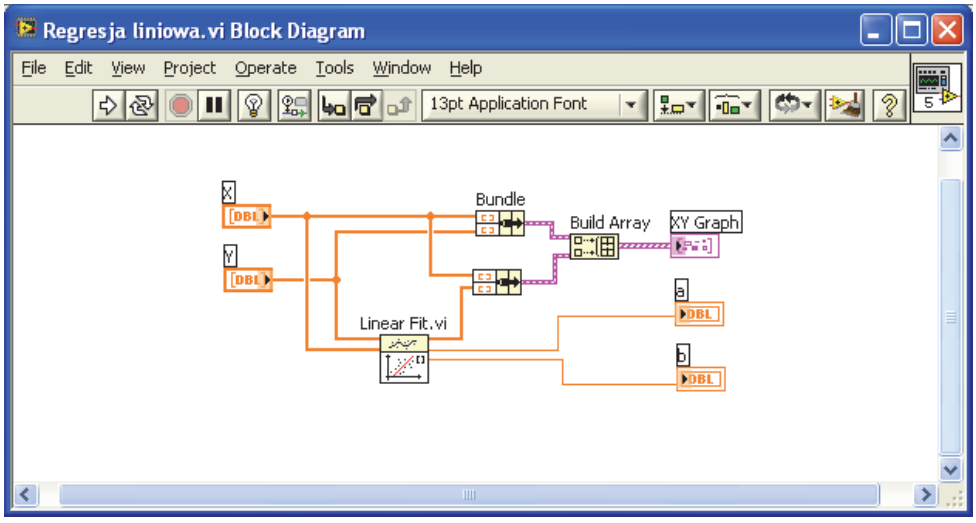
Pierwsza z nich służy do regresji liniowej, czyli wyznaczenia współczynników $a$ i $b$ opisujących prostą $y = a \cdot x + b$, dopasowaną do $n$ punktów $\{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$.

Panel programu przedstawiony jest na rysunku 1.36, a jego diagram na rysunku 1.37.




Rys. 1.36. Panel programu regresji liniowej

W celu umieszczenia tablicy na panelu, z grupy *Array, Matrix & Cluster* palety *Controls* należy wybrać element *Array*. Następnie wybrać typ elementów, jakimi będzie wypełniona tablica (w naszym programie jest to element *Numeric Control*). Wybrany z palety *Controls* element umieszczamy w kwadracie tablicy tak, by na krawędzi kwadratu pojawiła się przerywana linia.




Rys. 1.37. Diagram programu regresji liniowej

W programie wykorzystano funkcję *Linear Fit* (*Functions* → *Mathematics* → *Fitting*). Oprócz wartości współczynników *a* i *b* wyniki przedstawiane są na wykresie XY.

Kolejny program służy do obliczania zawartości harmonicznych w przebiegu odkształconym, składającym się z pierwszej i trzeciej harmonicznej o zadanej amplitudzie i częstotliwości oraz przesunięciu fazowym między tymi składowymi.

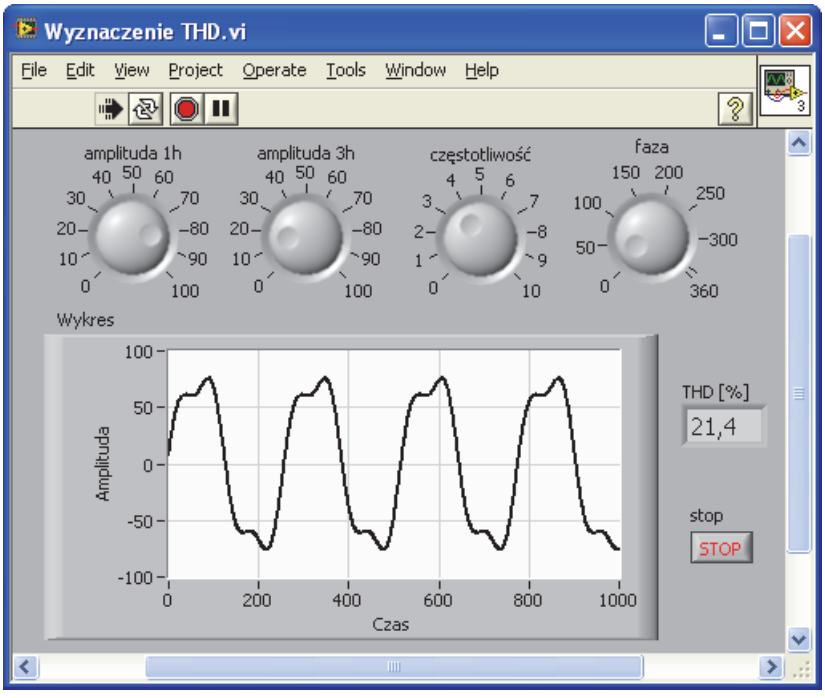
Współczynnik zawartości harmonicznych THD (*Total Harmonic Distortion*) definiowany jest jako stosunek wartości skutecznej wyższych harmonicznych sygnału, do wartości skutecznej składowej podstawowej $U_1$:

$$THD = \frac{\sqrt{\sum_{i=2}^{n} U_i^2}}{U_1} \tag{1.1}$$

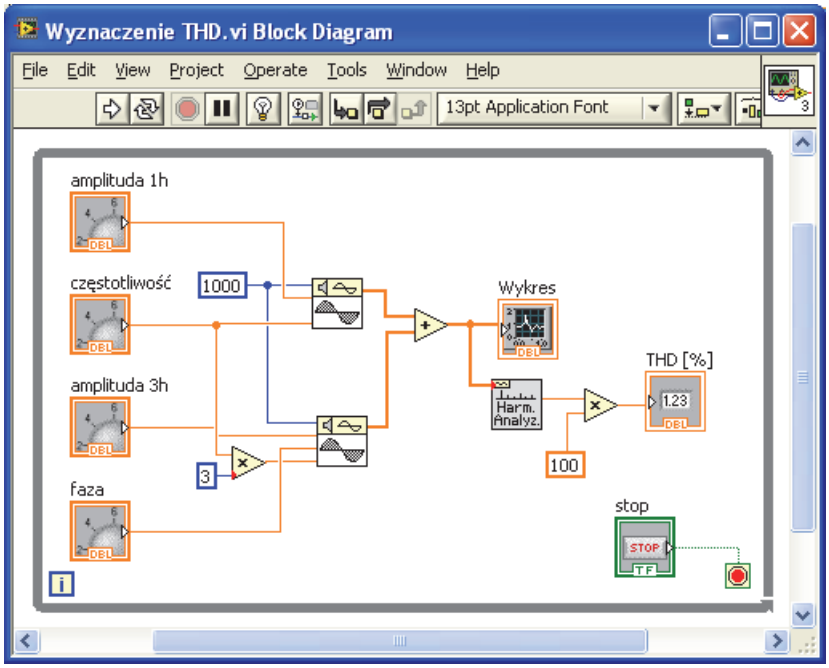
Panel programu przedstawiony jest na rysunku 1.38, a jego diagram na rysunku 1.39.

W trzecim przykładzie wyszukiwane są wartości szczytowe w analizowanym przebiegu. Do wyszukania szczytów wykorzystano funkcję *Waveform Peak Detection*. Na jej wejścia można podać próg (*threshold*), powyżej którego będą wyszukiwane wartości szczytowe i ich minimalną szerokość (*width*). Na wyjściu otrzymujemy położenie wartości szczytowych i ich poziom.

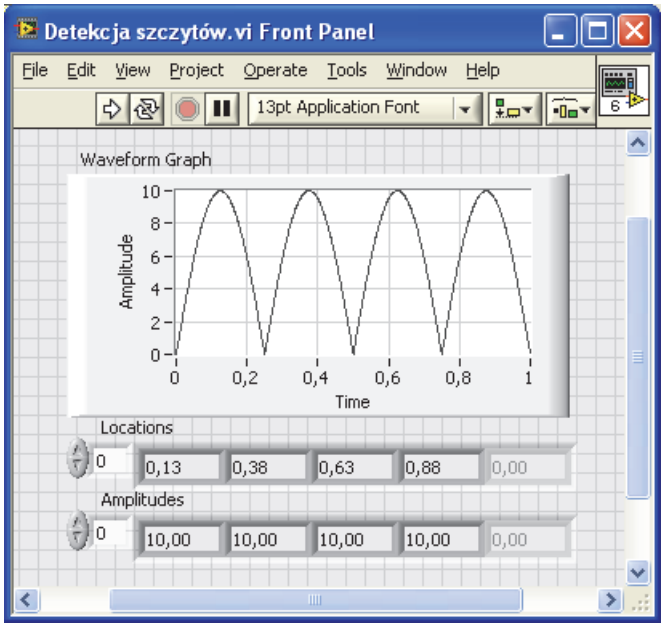
W programie, którego panel przedstawiony jest na rysunku 1.40, a diagram na rysunku 1.41, wartości szczytowe wyszukiwane są w przebiegu uzyskanym jako wartość bezwzględna przebiegu sinusoidalnego.



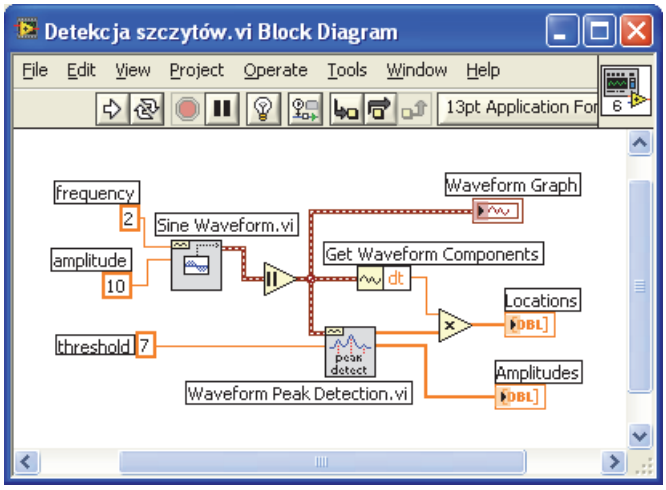

Rys. 1.38. Panel programu do wyznaczenia współczynnika THD




Rys. 1.39. Diagram programu do wyznaczenia współczynnika THD

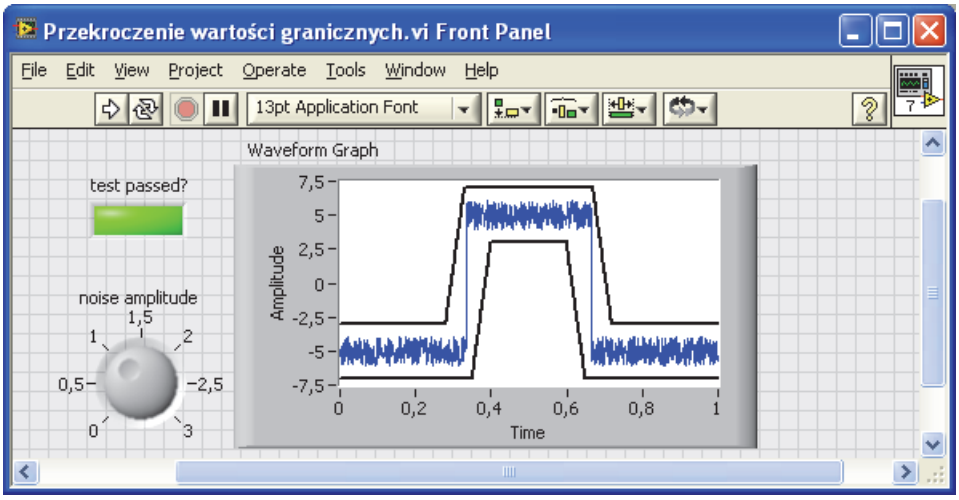



Rys. 1.40. Panel programu do wykrywania wartości szczytowych

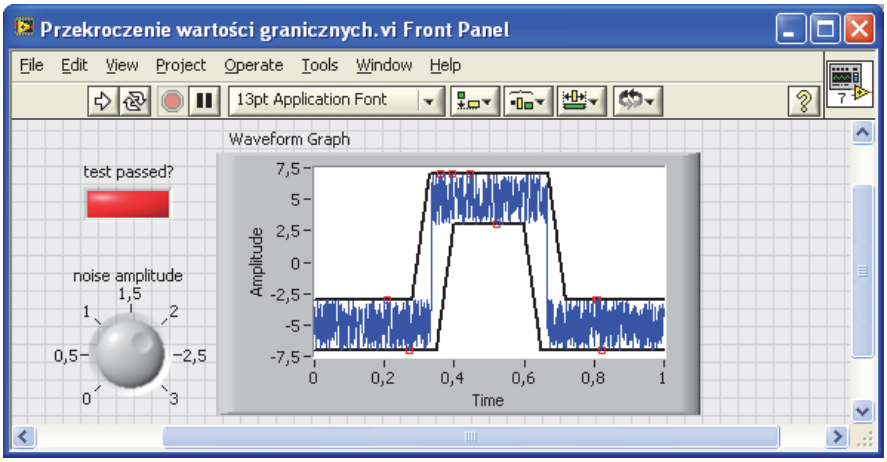



Rys. 1.41. Diagram programu do wykrywania wartości szczytowych

W kolejnym przykładzie zadawane są wartości graniczne sygnału, a sygnalizowane jest ich przekroczenie przez sygnał pomiarowy. Panel programu dla pozytywnego wyniku testu pokazany jest na rysunku 1.42, dla wyniku negatywnego na rysunku 1.43, natomiast jego diagram na rysunku 1.44.



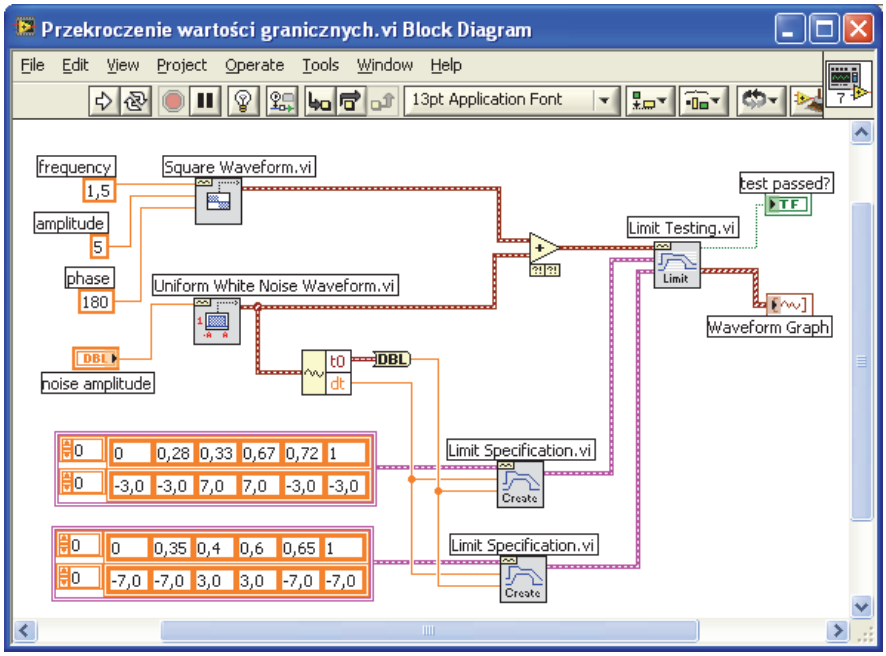

Rys. 1.42. Panel programu do testowania przekroczenia wartości granicznych – wynik testu pozytywny




Rys. 1.43. Panel programu do testowania przekroczenia wartości granicznych – wynik testu negatywny

Analizowany sygnał został zasymulowany jako suma przebiegu prostokątnego o stałej amplitudzie i stałym okresie (funkcja *Square Waveform*) oraz szumu (funkcja *Uniform White Noise Waveform*).

Do zadania wartości granicznych wykorzystano funkcje *Limit Specification*. Na ich wejścia *Specification Cluster* podawane są klastry zawierające tablice ze współrzędnymi punktów charakterystyki granicznej. Do sprawdzenia przekroczenia wartości granicznych służy funkcja *Limit Testing*.




Rys. 1.44. Diagram programu do testowania przekroczenia wartości granicznych

# Próbkowanie, kwantyzacja, aliasing, analiza widmowa

## 2.1. Przetwarzanie analogowo-cyfrowe

Przetwarzanie analogowo-cyfrowe jest procesem, w którym sygnał analogowy $x(t)$, określony w każdej chwili czasowej w przedziale obserwacji, zostaje przetworzony na ciąg $N$ liczb $\{x[0], x[1], x[2], \ldots, x[N-1]\}$ odpowiadających wartościom chwilowym sygnału analogowego w stałych odstępach czasu. Przetwarzanie analogowo-cyfrowe obejmuje trzy etapy: próbkowanie, kwantyzację i kodowanie.

Próbkowanie sygnału ciągłego $x(t)$ polega na akwizycji w dyskretnych momentach czasu $n \cdot \Delta t$ próbek $x(n \cdot \Delta t)$. W wyniku próbkowania otrzymuje się ciąg próbek sygnału $\{x(t_0), x(t_1), x(t_2), \ldots, x(t_{N-1})\}$ w dyskretnych chwilach $t_i$.

Kwantyzacja sygnału ciągłego $x(t)$ polega na przypisaniu jego próbkom wartości dyskretnych ze skończonego zbioru przedziałów kwantowania.

Zakres zmian sygnału podzielony zostaje na skończoną liczbę przedziałów kwantowania. Zwykle liczba tych przedziałów jest potęgą liczby 2, np. $2^{12} = 4096$, $2^{14} = 16384$. Każda próbka zostaje odwzorowana wartością przypisaną do danego przedziału kwantowania. W zależności od przyjętego sposobu kwantyzacji może to być górna lub dolna granica przedziału, najczęściej jednak jest to wartość w środku przedziału.

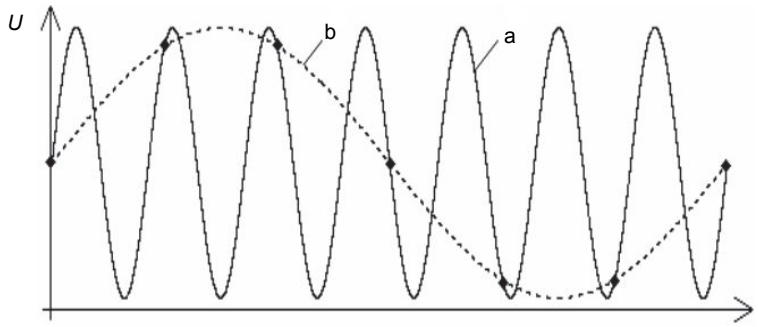
W trakcie kodowania, każdej próbce zostaje przypisana odpowiadająca jej liczba. Do kodowania nie jest wykorzystywany oddzielny układ, ale dokonuje się to automatycznie, w sposób zależny od typu przetwornika analogowo-cyfrowego.

## 2.2. Częstotliwość próbkowania

Ważnym parametrem próbkowania jest jego częstotliwość (*sampling rate*). Obecnie produkowane przetworniki analogowo-cyfrowe posiadają maksymalną częstotliwość próbkowania od kilkudziesięciu kiloherców do kilku megaherców. Częstotliwość próbkowania określa, ile próbek jest pobieranych w jednostce czasu.

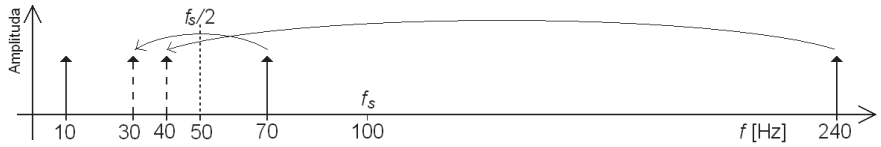
Jeżeli w próbkowanym sygnale występują składowe o częstotliwości większej od połowy częstotliwości próbkowania $f_s$, to część mocy z pasma powyżej $f_s/2$ zostaje „przesunięta" do zakresu częstotliwości mniejszych od $f_s/2$. W rezultacie składowe te nałożą się na składowe występujące wcześniej w tym zakresie, powodując deformację kształtu mierzonego sygnału.

Jeżeli w wyniku próbkowania z częstotliwością $f_s$ częstotliwość sygnału została zinterpretowana jako $f_p$ ($0 < f_p < f_s/2$), to przy nieodpowiednio dobranej częstotliwości próbkowania nie jesteśmy w stanie określić, czy mierzony sygnał miał rzeczywiście częstotliwość $f_p$, czy częstotliwość większą: $f_s \pm f_p$, $2f_s \pm f_p$, ... , $nf_s \pm f_p$ (gdzie $n$ jest liczbą naturalną) (rys. 2.1).




Rys. 2.1. Zjawisko nakładania się widm przy próbkowaniu z częstotliwością $f_s$ = 6 kHz:
a – sygnał o częstotliwości 7 kHz, b – sygnał odtworzony o błędnie zinterpretowanej częstotliwości $f_p$ =1 kHz

Przykład nakładania się widm (ang. *aliasing*) w dziedzinie częstotliwości przedstawia rysunek 2.2. Załóżmy, że częstotliwość próbkowania wynosi $f_s$ = 100 Hz, zaś w sygnale zawarte są składowe o częstotliwościach 10, 70 i 240 Hz.




Rys. 2.2. Błędna interpretacja składowych w widmie sygnału na skutek aliasingu

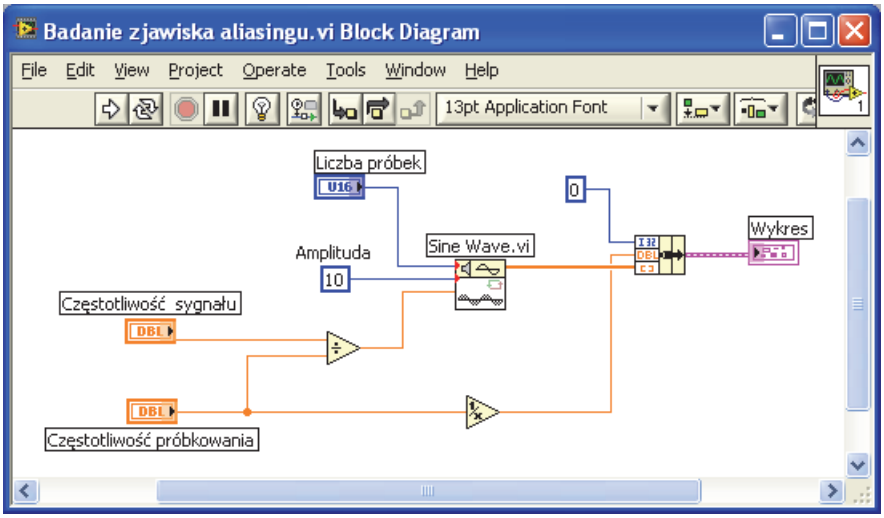
Połowa częstotliwości próbkowania wynosi $f_s/2$ = 50 Hz. Dlatego tylko najniższa składowa o częstotliwości 10 Hz została zinterpretowana prawidłowo. Składowa o częstotliwości 70 Hz jest widoczna jako 30 Hz, natomiast składowa o częstotliwości 240 Hz jest widoczna jako 40 Hz.

Zgodnie z twierdzeniem Shannona-Kotielnikowa, dla uniknięcia zjawiska aliasingu częstotliwość próbkowania musi być co najmniej dwa razy większa od maksymalnej częstotliwości występującej w paśmie sygnału.

Jeżeli częstotliwości składowych występujących w sygnale są większe od połowy częstotliwości próbkowania, należy zastosować filtr dolnoprzepustowy.

## 2.3. Program do badania wpływu aliasingu

Do zademonstrowania wpływu aliasingu można wykorzystać program przygotowany w LabVIEW, którego diagram pokazany jest na rysunku 2.3. Program umożliwia śledzenie składowych w widmie sygnału spróbkowanego ze stałą częstotliwością próbkowania.

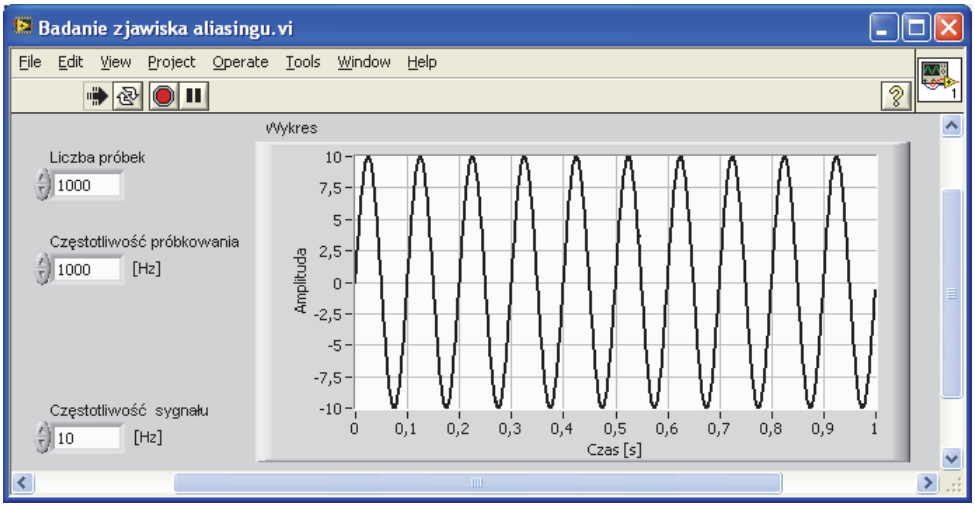



Rys. 2.3. Diagram programu do symulacji zjawiska aliasingu

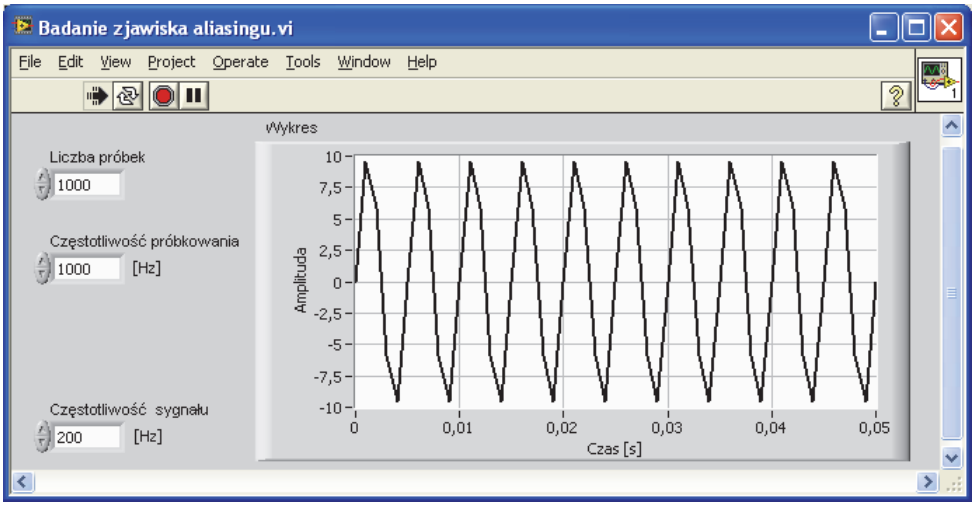
Symulację można wykonać dla różnych wartości zadanej częstotliwości sygnału $f$, częstotliwości próbkowania $f_s$ i liczby próbek $N$.

Częstotliwość $f_p$ określamy jako stosunek liczby $k$ okresów zliczonych na wykresie w przedziale czasu $\tau$ do szerokości tego przedziału: $f_p = k/\tau$. Czas $\tau$ odczytujemy z osi wykresu. Jeżeli liczba okresów jest tak duża, że niemożliwe jest ich policzenie, należy kliknąć na ostatnią liczbę na osi czasu i wpisać nową, mniejszą wartość.

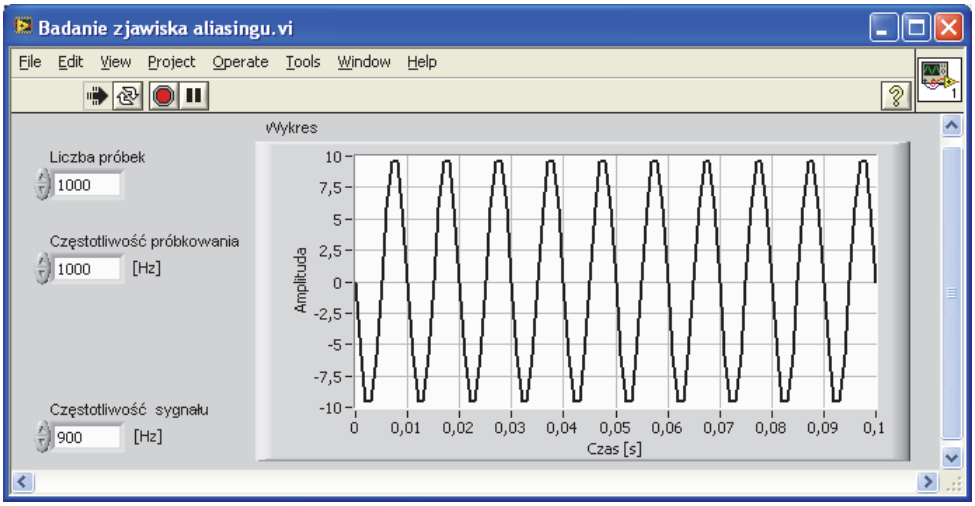
Przebiegi uzyskane dla przykładowych częstotliwości sygnału przedstawione są na panelu programu, na rysunkach 2.4–2.7.




Rys. 2.4. Panel programu z przebiegiem sygnału dla $f = 10$ Hz i $f_s = 1000$ Hz




Rys. 2.5. Panel programu z przebiegiem sygnału dla $f = 200$ Hz i $f_s = 1000$ Hz

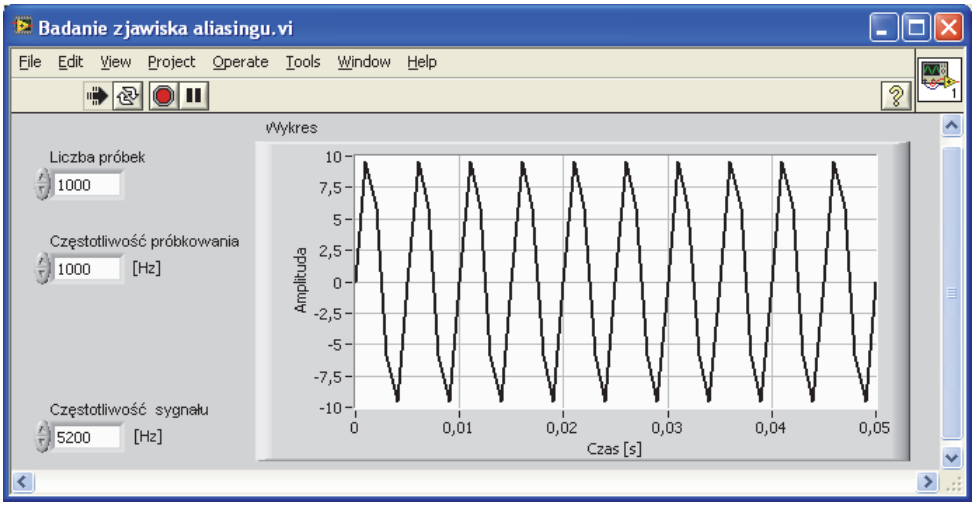



Rys. 2.6. Panel programu z przebiegiem sygnału dla $f = 900$ Hz i $f_s = 1000$ Hz

Przebieg na rysunku 2.4 został uzyskany dla $f = 10$ Hz i $f_s = 1000$ Hz. Z wykresu można odczytać $k = 10$ i $\tau = 1$ s. Stąd częstotliwość $f_p = k/\tau = 10/1$ 1/s = 10 Hz została odtworzona prawidłowo.

Przebieg na rysunku 2.5 został uzyskany dla $f = 200$ Hz i $f_s = 1000$ Hz. Z wykresu można odczytać $k = 10$ i $\tau = 0{,}05$ s. Stąd częstotliwość $f_p = k/\tau = 10/0{,}05$ 1/s = 200 Hz również została odtworzona prawidłowo.

Przebieg na rysunku 2.6 został uzyskany dla $f = 900$ Hz i $f_s = 1000$ Hz. Z wykresu można odczytać $k = 10$ i $\tau = 0{,}1$ s. Stąd częstotliwość $f_p = k/\tau = 10/0{,}1$ 1/s = 100 Hz została odtworzona nieprawidłowo. Wynika to z tego, że częstotliwość sygnału $f$ jest większa od połowy częstotliwości próbkowania $f_s/2 = 500$ Hz.

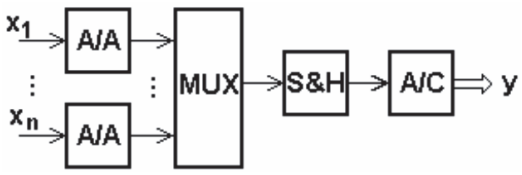



Rys. 2.7. Panel programu z przebiegiem sygnału dla $f = 5200$ Hz i $f_s = 1000$ Hz

Przebieg na rysunku 2.7 został uzyskany dla $f = 5200$ Hz i $f_s = 1000$ Hz. Z wykresu można odczytać $k = 10$ i $\tau = 0{,}05$ s. Stąd częstotliwość $f_p = k/\tau = 10/0{,}05$ 1/s = 200 Hz została odtworzona również nieprawidłowo.

## 2.4. Pomiary z wykorzystaniem bloku akwizycji sygnałów pomiarowych

Przedstawiony w rozdziale 2.3 program może być rozbudowany o pomiar rzeczywistego sygnału napięciowego, np. z generatora funkcyjnego. Do pomiaru można wykorzystać blok akwizycji sygnałów pomiarowych. Zadaniem takiego bloku jest zbieranie sygnałów pomiarowych i przetwarzanie ich na postać cyfrową. Na rysunku 2.8 przedstawiona jest podstawowa konfiguracja bloku akwizycji sygnałów pomiarowych. W jego skład wchodzą przełączniki kanałów, układy formujące, układy próbkująco-pamiętające i przetworniki analogowo-cyfrowe.




Rys. 2.8. Konfiguracja bloku akwizycji sygnałów pomiarowych;
A/A – układy formujące, MUX – przełącznik kanałów, S&H – układ próbkująco-pamiętający,
A/C – przetwornik analogowo-cyfrowy

Przełącznik kanałów (MUX) służy do przyłączenia do wyjścia jednego z kanałów wejściowych, w zależności od stanu sygnału sterującego. Zadaniem układu formującego (A/A) jest wstępna obróbka i normalizacja sygnału wejściowego (np. wzmocnienie).

Układ próbkująco-pamiętający (S&H) służy do pobrania próbki napięcia wejściowego i zapamiętania jej na określony czas. Układy S&H stosuje się, gdy szybkość zmian mierzo-

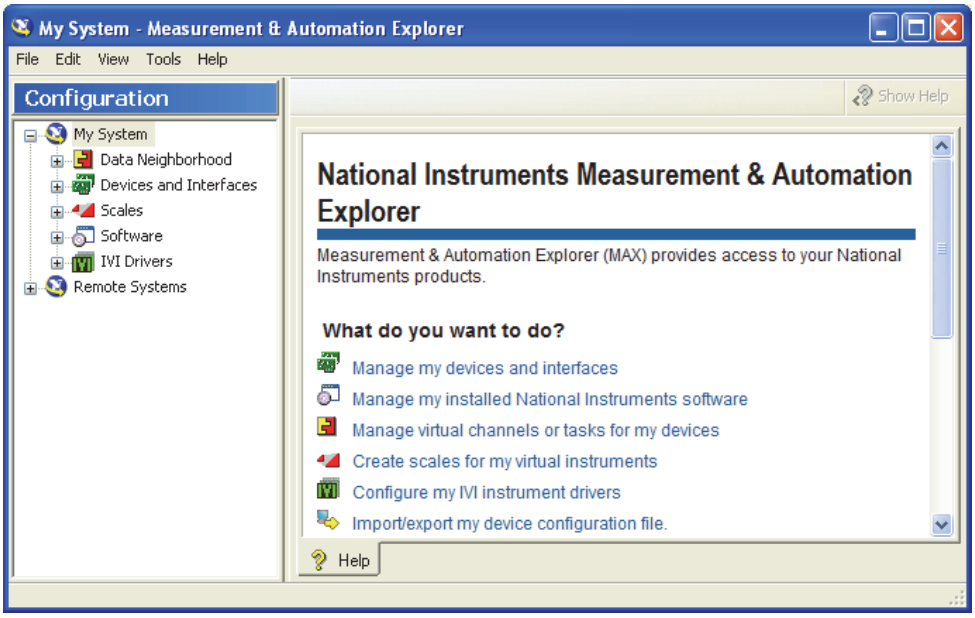
nego napięcia jest większa od dopuszczalnej dla przetwornika analogowo-cyfrowego. Układ zabezpiecza przed błędami, jakie mogą wystąpić, gdy w trakcie przetwarzania analogowo-cyfrowego zmienia się napięcie na wejściu przetwornika.

Głównym elementem układu akwizycji sygnałów pomiarowych jest przetwornik analogowo-cyfrowy. Zachodzi w nim konwersja sygnału analogowego na kod cyfrowy.

W przedstawionej konfiguracji stosowana jest akwizycja sygnałów z podziałem czasowym, która nie może być wykorzystana, kiedy wymagana jest akwizycja sygnałów z kilku kanałów równocześnie.

Układy akwizycji sygnałów pomiarowych wykonywane są najczęściej w postaci modułów (kart pomiarowych), które można umieścić bezpośrednio w komputerze (z magistralą PCI lub PCI Express) lub łączyć z komputerem za pomocą interfejsu USB (rys. 2.9).


Rys. 2.9. Przykładowy moduł akwizycji z interfejsem USB




Rys. 2.10. Okno programu Measurement & Automation Explorer

Po umieszczeniu karty pomiarowej w obudowie komputera lub połączeniu przez USB instalowane są sterowniki. Dla współczesnych modułów *Plug and Play* firmy National Instruments instalacja odbywa się automatycznie.

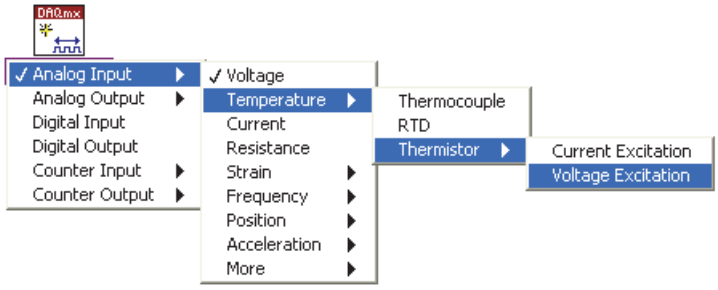
Prawidłowo zainstalowany sprzęt należy skonfigurować w programie *Measurement & Automation Explorer* (w skrócie MAX). Program ten można uruchomić niezależnie od LabVIEW lub bezpośrednio z menu LabVIEW, wybierając z opcji *Tools* polecenie *Measurement & Automation Explorer*. Po uruchomieniu programu pokazuje się okno jak na rys. 2.10.

W lewej części okna można wybrać interesujący nas sprzęt (lub dodać nowy), po prawej stronie podane są dodatkowe informacje. Przy użyciu programu MAX można testować połączony z komputerem sprzęt, wykonywać proste pomiary, tworzyć kanały wirtualne itp. Odświeżanie informacji o dołączonym sprzęcie może być wykonane przy każdorazowym uruchomieniu programu lub po wybraniu polecenia *Refresh* z opcji *View* menu.

Do obsługi modułów akwizycji firmy National Instruments wykorzystywane są sterowniki NI-DAQmx. Przygotowywane oprogramowanie oparte o kanały cechuje się przejrzystą strukturą. Dla sterowników DAQmx wprowadzono podział według zadań, dzięki czemu wszystkie typy pomiarów realizowane są za pomocą funkcji należących do jednej grupy. Zaletą sterowników NI-DAQmx jest również możliwość symulacji kart pomiarowych.

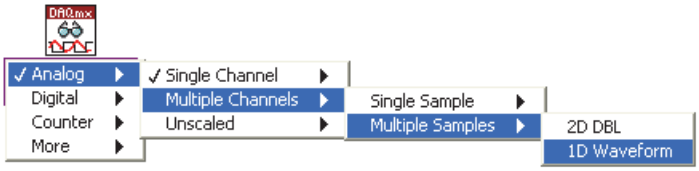
Większość zadań wykorzystujących sterowniki DAQmx może być wykonana z zastosowaniem dziesięciu podstawowych funkcji:

— *DAQmx Create Virtual Channel* – tworzy kanał wirtualny i dodaje go do zadania. Jeżeli zadanie nie jest wyspecyfikowane, zostaje utworzone nowe zadanie. Typ pomiaru zostaje wybrany przez ustawienie kursora na napisie pod ikoną i po naciśnięciu przycisku myszy zaznaczenie w rozwiniętym menu (rys. 2.11). W zależności od wybranego typu, funkcja posiada różne zaciski;



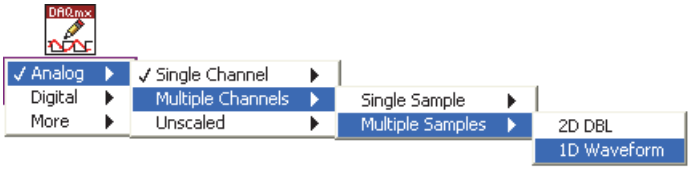

Rys. 2.11. Wybór typu pomiaru dla *DAQmx Create Virtual Channel*

— *DAQmx Read* – odczytuje próbki sygnału dla podanego zadania. Typ akwizycji, liczba kanałów, liczba próbek i typ danych wybierany jest z rozwiniętego menu po ustawieniu kursora na napisie pod ikoną i naciśnięciu przycisku myszy (rys. 2.12);



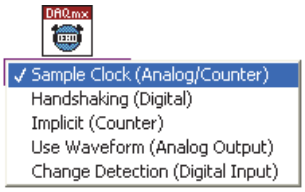

Rys. 2.12. Wybór ustawień dla *DAQmx Read*

— *DAQmx Write* – zapisuje próbki sygnału dla podanego zadania. Typ generacji (analogowa lub cyfrowa), liczba kanałów, liczba próbek i typ danych wybierany jest z rozwiniętego menu po ustawieniu kursora na napisie pod ikoną i naciśnięciu przycisku myszy (rys. 2.13);



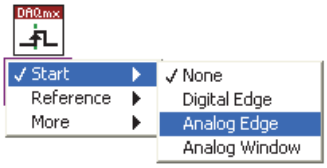

Rys. 2.13. Wybór ustawień dla *DAQmx Write*

— *DAQmx Wait Until Done* – czeka na zakończenie operacji akwizycji. Zapewnia zakończenie pomiarów lub generacji przed zatrzymaniem zadania;
— *DAQmx Timing* – określa uwarunkowania czasowe operacji akwizycji. Ustawiany parametr wybierany jest z rozwiniętego menu po ustawieniu kursora na napisie pod ikoną i naciśnięciu przycisku myszy (rys. 2.14). W zależności o wybranego typu, funkcja posiada różne zaciski;




Rys. 2.14. Wybór ustawień dla *DAQmx Timing*

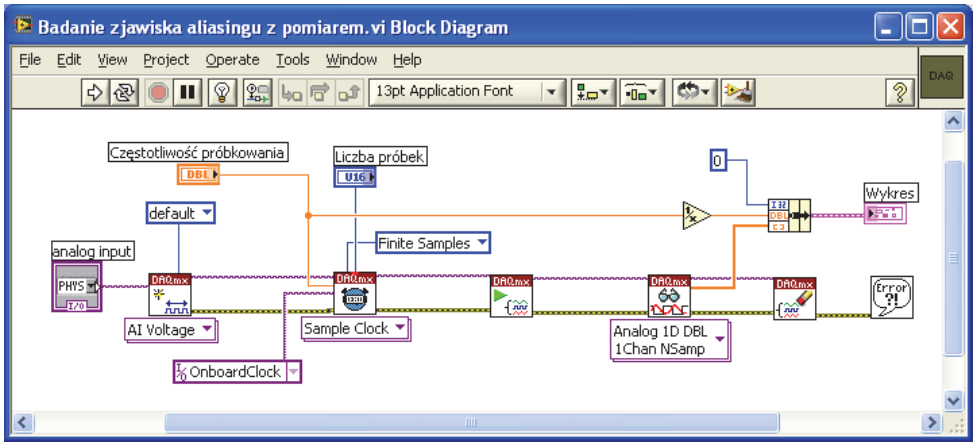
— *DAQmx Trigger* – określa warunki wyzwolenia pomiarów lub generacji. Rodzaj wyzwolenia wybierany jest z rozwiniętego menu po ustawieniu kursora na napisie pod ikoną i naciśnięciu przycisku myszy (rys. 2.15);




Rys. 2.15. Wybór ustawień dla *DAQmx Trigger*

— *DAQmx Start Task* – uruchamia wyspecyfikowane zadanie. Jeżeli funkcja *DAQmx Start Task* nie jest użyta, zadanie może zostać uruchomione funkcją *DAQmx Read* lub *DAQmx Write*. Funkcja *DAQmx Start Task* powinna być wykorzystana przy wielokrotnym wywoływaniu w pętli funkcji *DAQmx Read* lub *DAQmx Write*;
— *DAQmx Stop Task* – zatrzymuje wyspecyfikowane zadanie;
— *DAQmx Clear Task* – zamyka wyspecyfikowane zadanie. Jeżeli zadanie jest uruchomione, przed zamknięciem, zostaje zatrzymane;
— *DAQmx Property Node* – umożliwia odczyt i zmianę właściwości operacji związanych z akwizycją danych.

Na rysunku 2.16 przedstawiony jest diagram programu w LabVIEW do pobrania z zadanego wejścia analogowego modułu akwizycji określonej liczby próbek z zadaną częstotliwością.




Rys. 2.16. Program do wykonania pomiarów z wykorzystaniem modułu akwizycji sygnałów

Za pomocą funkcji *DAQmx Create Virtual Channel* zostaje utworzony kanał związany z zadanym wejściem analogowym karty pomiarowej. Funkcja *DAQmx Timing* określa źródło i częstotliwość sygnału taktującego oraz sposób próbkowania. Funkcja *DAQmx Start Task* uruchamia zadanie. Za pomocą funkcji *DAQmx Read* zrealizowany jest odczyt wartości próbek mierzonego napięcia. Po zakończeniu pomiarów zadanie zostaje zamknięte za pomocą funkcji *DAQmx Clear Task*.

## 2.5. Analiza Fouriera

Sygnały pomiarowe analizowane są zwykle w dziedzinie czasu lub w dziedzinie częstotliwości.

Sygnał okresowy spełniający warunki Dirichleta można przedstawić za pomocą szeregu Fouriera.

$$x(t) = \tfrac{1}{2}a_0 + \sum_{n=1}^{\infty}\left[a_n \cos(n\omega_0 t) + b_n \sin(n\omega_0 t)\right] \tag{2.1}$$

gdzie współczynniki składowych szeregu Fouriera są następujące:

$$a_0 = \frac{2}{T}\int_0^T x(t)\,dt \tag{2.2}$$

$$a_n = \frac{2}{T}\int_0^T x(t)\cos n\omega_0 t\,dt \tag{2.3}$$

$$b_n = \frac{2}{T}\int_0^T x(t)\sin n\omega_0 t\,dt \tag{2.4}$$

Dla sygnału ciągłego $x(t)$ spróbkowanego równomiernie z częstotliwością $f_s$, co najmniej dwa razy większą od częstotliwości maksymalnej składowej występującej w widmie sygnału, dyskretne przekształcenie Fouriera ma postać:

$$X\left(\frac{nf_s}{N}\right) = \sum_{k=0}^{N-1} x(kT_s) \cdot e^{-j2\pi nk/N} \tag{2.5}$$

gdzie: $n = 0, 1, 2, ..., N–1$; $f_s = 1/T_s$

Dyskretna transformata Fouriera DFT (*Discrete Fourier Transform*) przekształca ciąg próbek wartości chwilowych sygnału o długości $N$:

$$x(kT_s) = x(0), x(T_s), x(2T_s), ..., x((N-1)T_s) \tag{2.6}$$

na $N$ punktowy ciąg dyskretny w dziedzinie częstotliwości:

$$X\left(\frac{nf_s}{N}\right) = X(0), X\left(\frac{f_s}{N}\right), X\left(\frac{2f_s}{N}\right), ..., X\left(\frac{(N-1)f_s}{N}\right) \tag{2.7}$$

W latach sześćdziesiątych pojawił się algorytm szybkiej transformaty Fouriera FFT (*Fast Fourier Transform*). W stosunku do dyskretnej transformaty Fouriera DFT szybka transformata Fouriera FFT jest algorytmem umożliwiającym znaczne zmniejszenie liczby wykonywanych działań arytmetycznych, a więc skrócenie czasu obliczeń. Realizuje się to poprzez podział ciągu $N$ próbek na krótsze ciągi, dla których obliczana jest dyskretna transformata. Liczba próbek, dla których obliczana jest FFT, powinna być potęgą liczby 2. Do obliczenia DFT należy wykonać $N^2$ mnożeń, w przypadku FFT tylko $N \cdot \lg_2 N$. Przykładowo dla $N = 1024$ daje to przeszło 100-krotne zmniejszenie liczby wykonywanych mnożeń. Wynik uzyskany przy wykonywaniu obliczeń DFT i FFT jest taki sam.

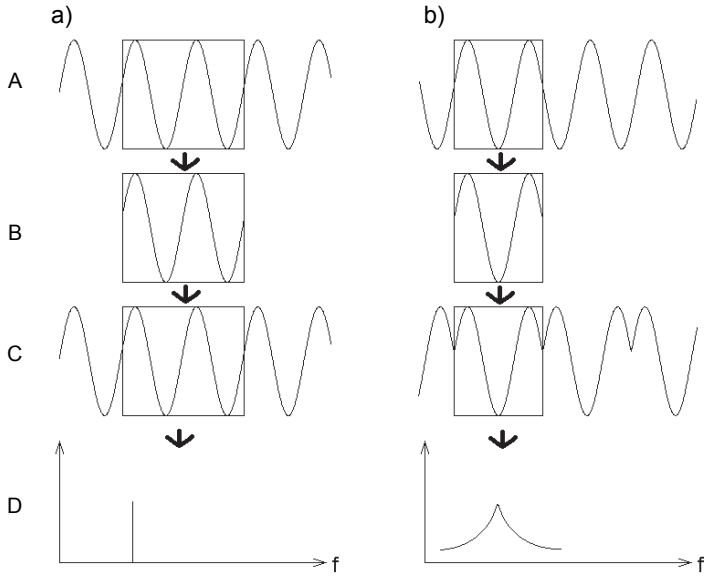
## 2.6. Okno czasowe prostokątne

Dyskretna transformata Fouriera jest obliczana dla skończonej długości ciągu $x(kT_s)$, co w praktyce jest realizowane przez ograniczenie ciągu próbek wartości chwilowych sygnału za pomocą okna czasowego. Efektem nałożenia na sygnał okna czasowego jest zniekształcenie jego widma zwane przenikaniem lub przeciekiem. Ponieważ kształt okna czasowego wpływa na strukturę widma analizowanego sygnału, są one często nazywane oknami widmowymi.

Okna czasowe przyjmują wartości niezerowe wyłącznie dla $n = 0, 1, 2, ..., N – 1$, gdzie $N$ jest długością okna.

Stopień zniekształcenia widma sygnału zależy od tego, czy ciąg próbek po operacji okienkowania zawiera całkowitą liczbę okresów składowych harmonicznych w nim zawartych (rys. 2.17).

Po operacji okienkowania otrzymujemy ciąg $N$ próbek, przy czym odstęp między sąsiednimi próbkami jest równy okresowi próbkowania $T_s = 1/f_s$. Długość okna czasowego wynosi $T_w = N \cdot T_s = N/f_s$.




Rys. 2.17. Wpływ okna czasowego prostokątnego na widmo sygnału: a) szerokość okna równa dwóm okresom analizowanego sygnału, b) szerokość okna równa niecałkowitej liczbie okresów sygnału, A – sygnał oryginalny, B – sygnał ograniczony oknem czasowym, C – sygnał po operacji okienkowania, D – widmo sygnału

Widmo częstotliwościowe sygnału charakteryzowane jest przez dwa parametry: rozdzielczość i szerokość widma. Rozdzielczość widma $f_w$ określa odległość między dwoma prążkami. Im mniejsza wartość $f_w$, tym gęstość prążków jest większa, co pozwala na dokładniejsze wyznaczenie składowych występujących w widmie. Rozdzielczość widma można wyznaczyć jako odwrotność długości okna czasowego:

$$f_w = \frac{1}{T_w} = \frac{f_s}{N} \tag{2.8}$$

Szerokość widma $f_m$ określona jest przez największą częstotliwość przedstawioną w widmie sygnału. Liczba składowych w widmie (bez składowej zerowej) wynosi $N/2$-1. Stąd szerokość widma można obliczyć jako iloczyn liczby składowych i rozdzielczości:

$$f_m = \left(\frac{N}{2} - 1\right) f_w = \left(\frac{N}{2} - 1\right) \frac{f_s}{N} = \frac{1}{2}\left(1 - \frac{2}{N}\right) f_s \tag{2.9}$$

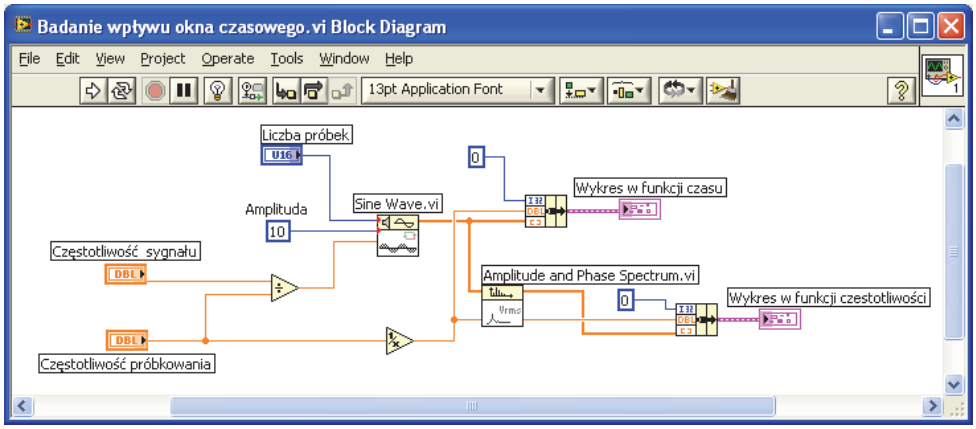
Dla dużej liczby próbek $N$ szerokość widma jest w przybliżeniu równa połowie częstotliwości próbkowania $f_m = f_s/2$.

Dobór okna czasowego ma ważne znaczenie dla rozdzielczości częstotliwościowej i amplitudowej w analizie częstotliwościowej sygnałów. Zbyt mała rozdzielczość nie pozwoli na rozróżnienie dwóch składowych o zbliżonych częstotliwościach lub dwóch składowych o większej różnicy amplitud.

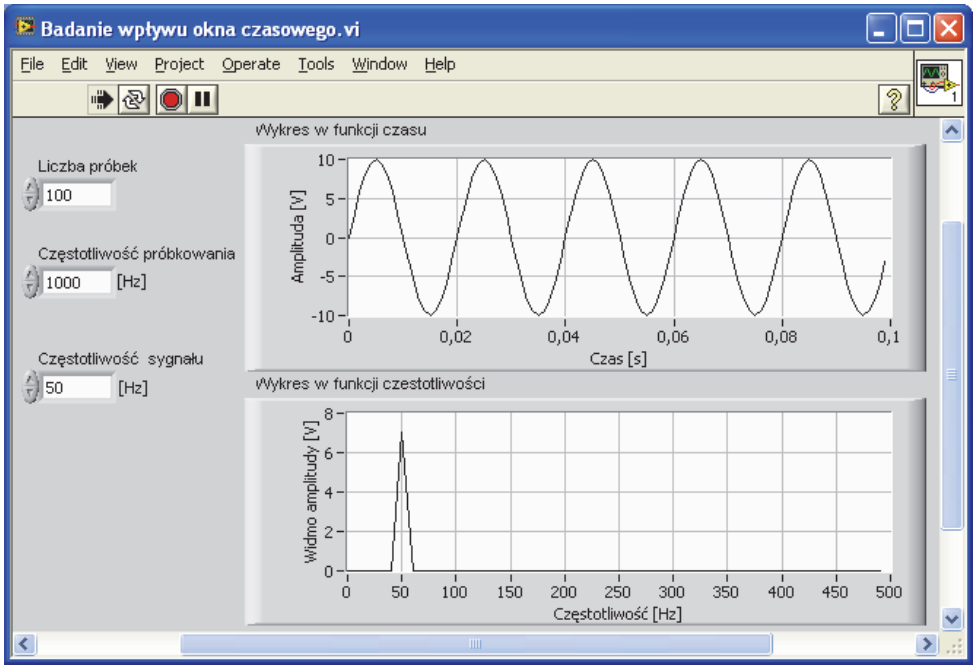
Na rysunku 2.18 przedstawiony jest diagram przygotowanego w LabVIEW programu, którego zadaniem jest zademonstrowanie wpływu okna czasowego na widmo sygnału.

Wyniki uzyskane dla przykładowych częstotliwości sygnału, częstotliwości próbkowania oraz liczby próbek okna wycinającego przedstawione są na panelu programu na rysunkach 2.19–2.22.

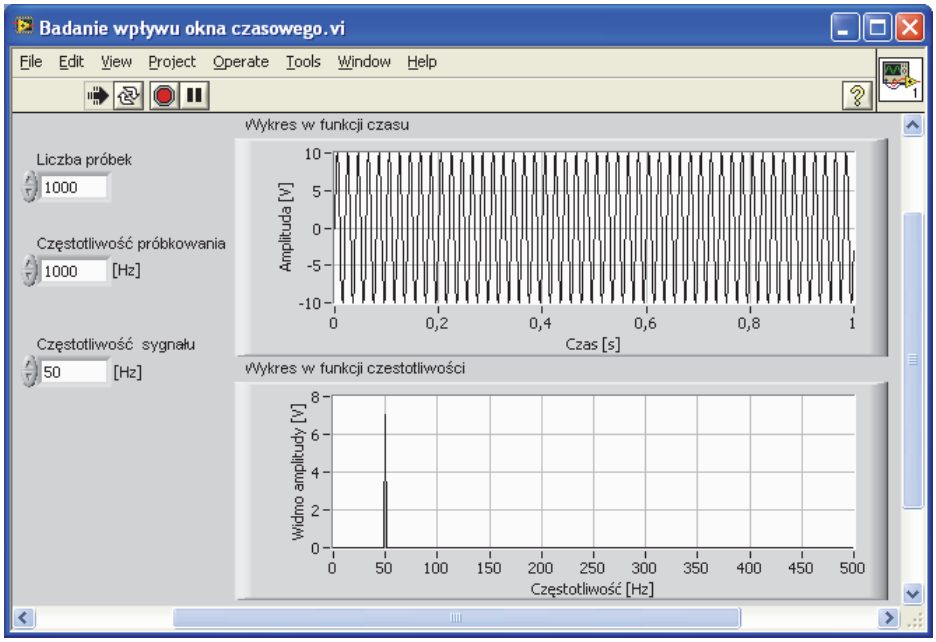
Wykresy na rysunku 2.19 zostały uzyskane dla $f_s = 1000$ Hz i $N = 100$. Rozdzielczość częstotliwości, jak widać na wykresie widma częstotliwościowego, nie jest najlepsza i wynosi $f_w = f_s/N = 1000$ Hz/100 = 10 Hz. Szerokość widma wynosi $f_m = f_s/2 = 1000$ Hz/2 = 500 Hz.



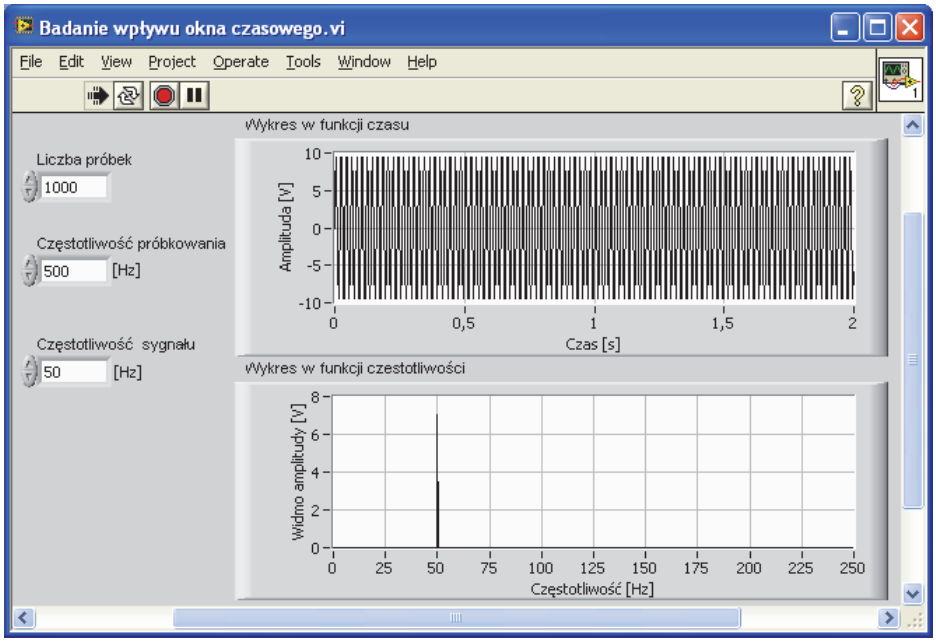

Rys. 2.18. Diagram programu do demonstracji wpływu okna czasowego




Rys. 2.19. Panel programu z przebiegiem sygnału w funkcji czasu oraz widmem częstotliwościowym dla $f = 50$ Hz, $f_s = 1000$ Hz i $N = 100$




Rys. 2.20. Panel programu z przebiegiem sygnału w funkcji czasu oraz widmem częstotliwościowym dla $f = 50$ Hz, $f_s = 1000$ Hz i $N = 1000$



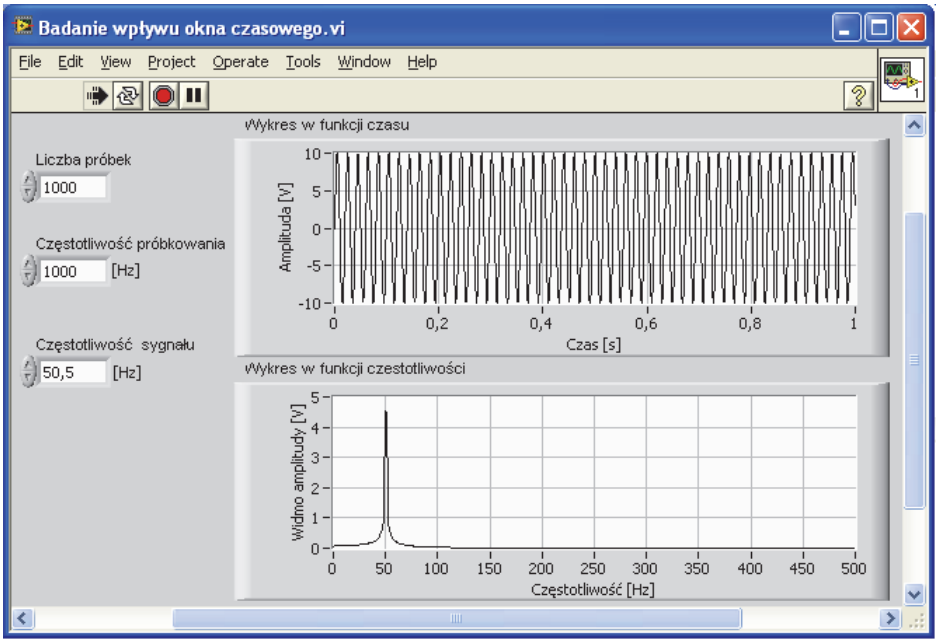

Rys. 2.21. Panel programu z przebiegiem sygnału w funkcji czasu oraz widmem częstotliwościowym dla $f = 50$ Hz, $f_s = 500$ Hz i $N = 1000$

Wykresy na rysunku 2.20 zostały uzyskane dla $f_s = 1000$ Hz i $N = 1000$. Rozdzielczość częstotliwości dzięki zwiększeniu liczby próbek wynosi $f_w = f_s/N = 1000$ Hz/1000 = 1 Hz. Szerokość widma nie zależy od liczby próbek, dlatego jak poprzednio wynosi $f_m = f_s/2 = 1000$ Hz/2 = 500 Hz.

Wykresy na rysunku 2.21 zostały uzyskane dla $f_s = 500$ Hz i $N = 1000$. Rozdzielczość częstotliwości, dzięki zmniejszeniu częstotliwości próbkowania wynosi $f_w = f_s/N =$ 500 Hz/1000 = 0,5 Hz. Spowodowało to jednak zmniejszenie szerokości widma, które wynosi $f_m = f_s/2 = 500$ Hz/2 = 250 Hz.

W praktyce można spotkać się z pytaniem, z jaką częstotliwością należy próbkować i ile próbek pobrać dla zadanego zakresu widma i rozdzielczości. Przykładowo, gdy chcemy analizować sygnał w zakresie do 25 kHz z rozdzielczością 0,5 Hz należy próbkować z częstotliwością $f_s = 2f_m = 2 \cdot 25$ kHz =50 kHz i należy pobrać $N = f_s/f_w = 50$ kHz/0,5 Hz = 100 000 próbek.

Przedstawiony program pozwala również na sprawdzenie, jak na kształt i wysokość prążka w widmie częstotliwościowym wpływa to, czy liczba okresów w oknie pomiarowym spróbkowanego sygnału jest całkowita.




Rys. 2.22. Panel programu z przebiegiem sygnału w funkcji czasu oraz widmem częstotliwościowym dla $f = 50{,}5$ Hz, $f_s = 1000$ Hz i $N = 1000$

Wykresy na rysunku 2.20 zostały uzyskane dla częstotliwości analizowanego sygnału równej $f = 50$ Hz. W obszarze okna czasowego znalazło się $N \cdot f/f_s = 1000 \cdot 50$ Hz/1000 Hz = 50 okresów sygnału. Liczba okresów jest całkowita, dzięki czemu w widmie otrzymujemy pojedynczy prążek.

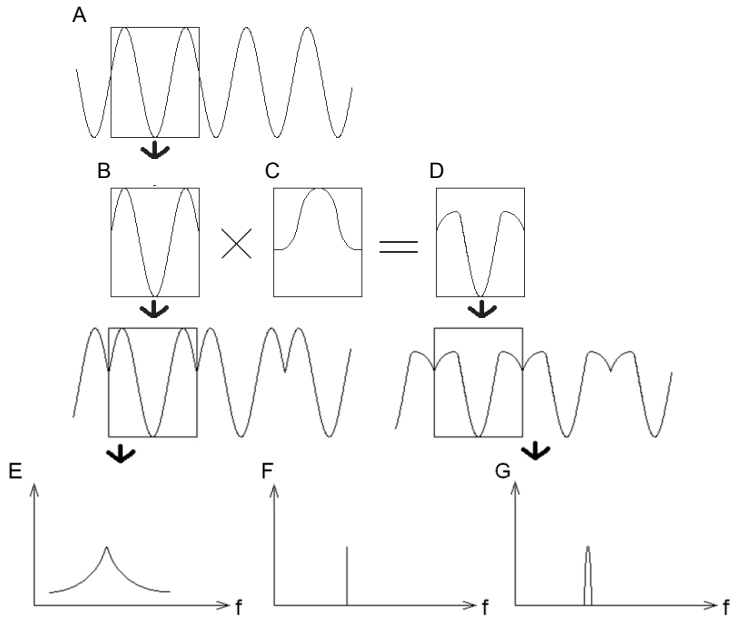
Wykresy na rysunku 2.22 zostały uzyskane dla częstotliwości sygnału równej $f = 50{,}5$ Hz. W obszarze okna czasowego znalazło się $N \cdot f/f_s = 1000 \cdot 50{,}5$ Hz/1000 Hz = 50,5

okresów sygnału. Liczba okresów nie jest całkowita, dlatego występuje zjawisko przenikania, objawiające się widocznym rozmyciem widma.

## 2.7. Okna czasowe wygładzające

Jak już wiadomo z poprzedniego rozdziału, zastosowanie okna czasowego powoduje deformację widma sygnału. Ten niekorzystny efekt może być ograniczony przez odkształcenie wycinka sygnału przez jego wytłumienie na krańcach przedziału.

Odkształcenie takie, a przez to zmniejszenie przenikania widma, można uzyskać przez zastosowanie okna czasowego o kształcie innym niż prostokątne (rys. 2.23).




Rys. 2.23. Wpływ okna wygładzającego na postać widma sygnału: A – sygnał oryginalny, B – sygnał czasowy ograniczony oknem czasowym, C – kształt okna wygładzającego, D – postać sygnału na wyjściu okna wygładzającego, E – widmo sygnału obliczone bez okna wygładzającego, F – teoretyczne widmo sygnału, G – widmo sygnału ograniczonego oknem wygładzającym

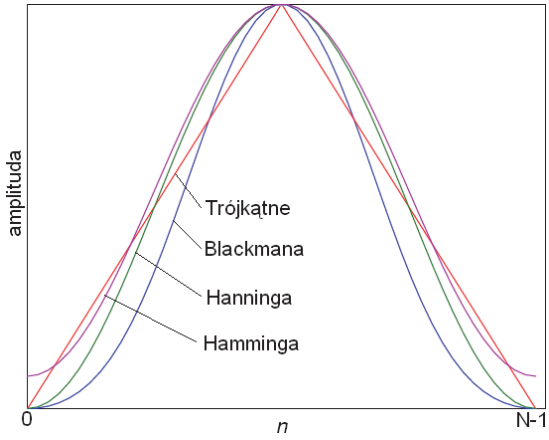
Znanych jest wiele różnych okien, różniących się charakterystyką (tab. 2.1, rys. 2.24). Wybór danego okna zależy od konkretnego zastosowania, np. okno prostokątne nadaje się bardziej do przebiegów nieustalonych, okno Hanninga do sygnałów ciągłych.

Poszczególne funkcje różnią się szerokością listka głównego i tłumieniem listków bocznych, co ma znaczenie w rozdzielczości analizy częstotliwościowej (rys. 2.25). Jeżeli szerokość listka głównego będzie większa od różnicy częstotliwości dwóch składowych, to odpowiadające im prążki zleją się w jeden. Jeżeli natomiast wysokość listków bocznych będzie porównywalna z amplitudą innych składowych, to te składowe mogą nie zostać wykryte.

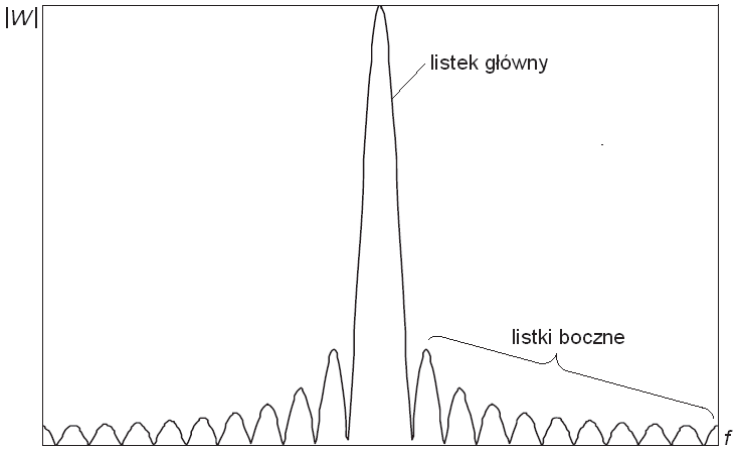
**Tabela 2.1**

Funkcje okien wygładzających

| Nazwa okna | Równanie |
|---|---|
| Prostokątne | $w(n) = 1,\ 0 \le n \le N-1$ |
| Trójkątne (Bartletta) | z zerowymi wartościami skrajnych elementów<br>$w(n) = [2/(N-1)]\,[(N-1)/2 - \lvert n-(N-1)/2\rvert],\ 0 \le n \le N-1$<br>z niezerowymi wartościami skrajnych elementów<br>$w(n) = [2/N\,(N/2 - \lvert n-(N-1)/2\rvert],\ 0 \le n \le N-1$ |
| Hanninga (Hanna) | $w(n) = 0{,}5\{1 - \cos[2\pi n/(N-1)]\},\ 0 \le n \le N-1$ |
| Hamminga | $w(n) = 0{,}54 - 0{,}46\cos[2\pi n/(N-1)],\ 0 \le n \le N-1$ |
| Blackmana | $w(n) = 0{,}42 - 0{,}5\cos[2\pi n/(N-1)] + 0{,}08\cos[4\pi n/(N-1)],\ 0 \le n \le N-1$ |




Rys. 2.24. Funkcje okien wygładzających




Rys. 2.25. Przykładowe widmo amplitudowe

Okno trójkątne w stosunku do okna prostokątnego lepiej tłumi listki boczne, ale ma szerszy listek główny. Okno Hanninga, ze względu na funkcję cosinus występującą w jego opisie ma łagodniejszy kształt od okna prostokątnego i lepiej tłumi listki boczne. Kształt okna Hamminga w porównaniu z oknem Hanninga powoduje zwiększenie tłumienia listków bocznych, nie wpływając na szerokość listka głównego. Okno Blackmana w porównaniu z oknami Hamminga i Hanninga lepiej tłumi listki boczne, ale ma szerszy listek główny. Porównanie okien wygładzających przedstawiono w tabeli 2.2.

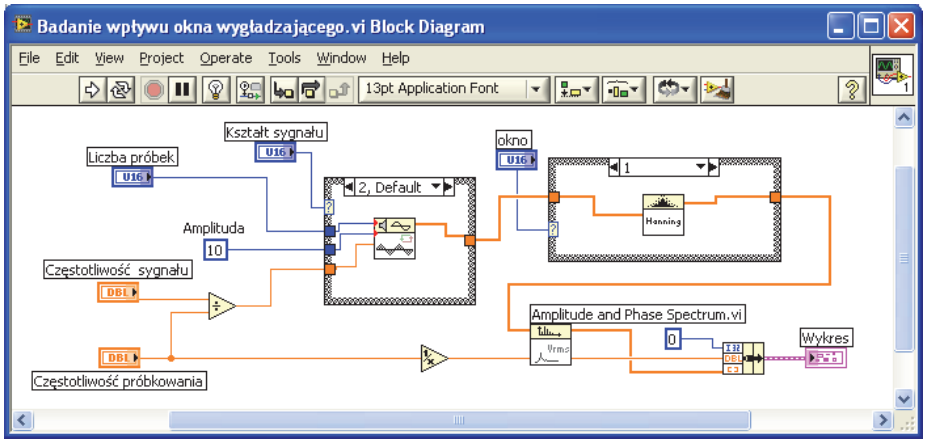
**Tabela 2.2**

Porównanie parametrów okien wygładzających

| Nazwa okna | Szerokość listka głównego | Względne tłumienie listków bocznych [dB] |
|---|---|---|
| Prostokątne | $2\pi/N$ | –13 |
| Trójkątne (Bartletta) | $4\pi/N$ | –25 |
| Hanninga (Hanna) | $4\pi/N$ | –31 |
| Hamminga | $4\pi/N$ | –41 |
| Blackmana | $6\pi/N$ | –57 |

Zastosowanie okien wygładzających pozwala na zmniejszenie wpływu przenikania widma, bez konieczności poszerzania okna czasowego. Należy jednak uważać, szczególnie w przypadku, gdy w analizowanym sygnale występują składowe o zbliżonych częstotliwościach, których prążki mogą zostać połączone w jeden. Rodzaj okna często dobierany jest doświadczalnie, jako kompromis między dążeniem do stłumienia listków bocznych i uzyskaniem wąskiego listka głównego. Problem przy doświadczalnym doborze okna wygładzającego wynika z tego, że nie znamy prawdziwego kształtu widma sygnału, dlatego nie wiemy, w jakim stopniu uzyskane widmo jest do niego zbliżone.

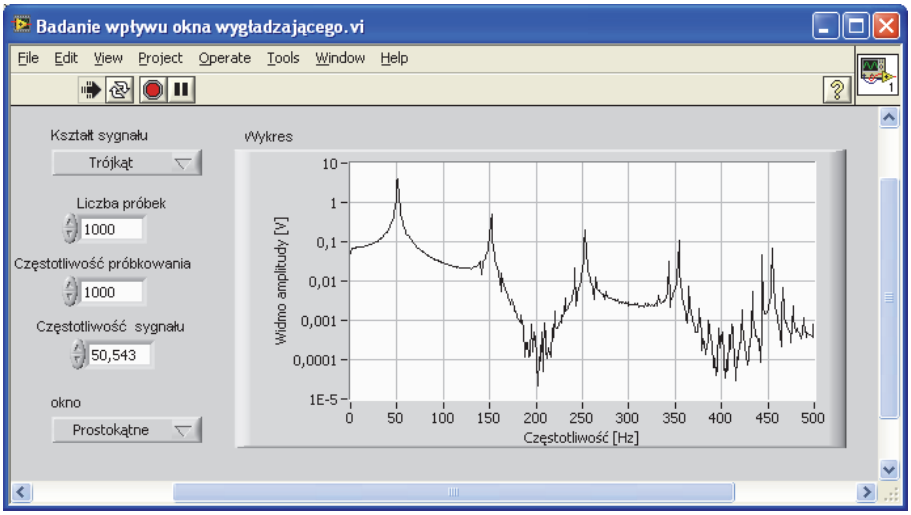
Na rysunku 2.26 przedstawiony jest diagram przygotowanego w LabVIEW programu, którego zadaniem jest zademonstrowanie wpływu zastosowanego okna wygładzającego na widmo amplitudy.



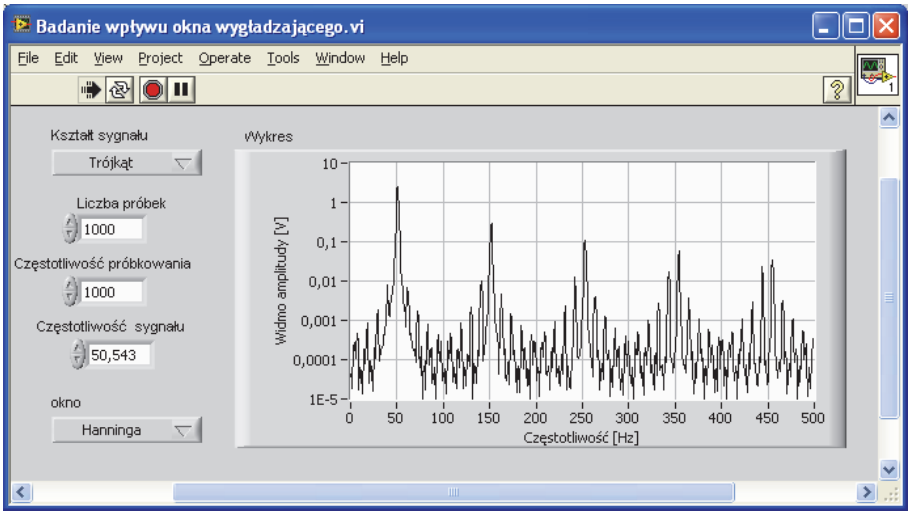

Rys. 2.26. Diagram programu do demonstracji wpływu okna wygładzającego

Program pozwala na porównanie widma uzyskanego przy zastosowaniu okna prostokątnego oraz okna trójkątnego (Bartletta), Hanninga (Hanna), Hamminga i Blackmana. Różnice są lepiej widoczne po przełączeniu wykresu z widmem na logarytmiczną skalę amplitudy (po ustawieniu kursora na wykresie i naciśnięciu prawego przycisku myszy wybrać: *Y Scale* → *Mapping* → *Logarithmic*). Program umożliwia przedstawienie widma sygnału sinusoidalnego, prostokątnego oraz trójkątnego o zadanej częstotliwości.

Do wyboru funkcji okna oraz kształtu sygnału wykorzystano funkcje *Case*. Kolejne ramki dla danej struktury dodajemy przez ustawienie kursora na krawędzi ramki i po naciśnięciu prawego przycisku myszy wybranie z menu polecenia *Add Case After*. W jednej z ramek, oprócz kolejnego numeru, należy wpisać *Default*.




Rys. 2.27. Panel programu z widmem sygnału trójkątnego bez okna wygładzającego




Rys. 2.28. Panel programu z widmem sygnału trójkątnego z oknem Hanninga

Do zacisku z symbolem „?” na ramce struktury *Case* łączymy zacisk umieszczonej na panelu kontrolki *Text Ring* z grupy *Ring & Enum*. Po ustawieniu kursora na kontrolce i naciśnięciu prawego przycisku myszy, możemy wybrać polecenie *Edit Items*, otwierające okno do wpisania kolejnych pozycji kontrolki. Należy zwrócić uwagę, by wartości przypisane do kolejnych pozycji odpowiadały wartościom w polu na górnej ramce struktury *Case*.

Widmo sygnału przebiegu o kształcie trójkątnym uzyskane dla przykładowej częstotliwości próbkowania oraz liczby próbek przedstawione jest na rysunkach 2.27 i 2.28.

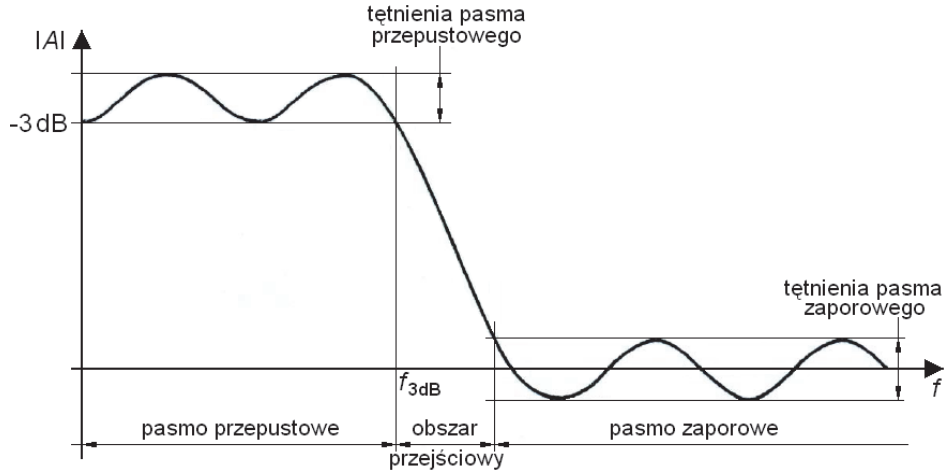
Porównując wykresy na obu rysunkach, widać, że zastosowanie okna wygładzającego spowodowało wyraźne zmniejszenie poziomu listków bocznych.

# Filtry o skończonej i nieskończonej odpowiedzi impulsowej

## 3.1. Charakterystyka filtrów

Celem filtracji jest usunięcie niepożądanych składowych z sygnału lub wydzielenie składowych pożądanych.

Ocena filtrów może być wykonana na podstawie ich charakterystyk. Na rysunku 3.1 przedstawiona jest typowa charakterystyka amplitudowa cyfrowego filtra dolnoprzepustowego.




Rys. 3.1. Charakterystyka amplitudowa cyfrowego filtra dolnoprzepustowego

Na charakterystyce możemy wyróżnić:

- pasmo przepustowe – zakres częstotliwości, w którym sygnał przechodzi przez układ osłabiony w niewielkim stopniu (do częstotliwości obcięcia $f_{3dB}$, przy której amplituda spada o 3 dB wartości nominalnej),
- obszar przejściowy – zakres częstotliwości między pasmem przepustowym i zaporowym, określający szybkość zmiany amplitudy w funkcji częstotliwości (stromość nachylenia charakterystyki),
- pasmo zaporowe – zakres częstotliwości, w którym sygnał spada poniżej założonego poziomu.

Możliwe jest zaprojektowanie filtra, w którym nie występują tętnienia w paśmie przepustowym, jednak odbywa się to kosztem stromości nachylenia charakterystyki. Oscylacje w paśmie zaporowym nie mają praktycznie znaczenia, ważne jest tylko, by amplituda nie przekroczyła założonego poziomu.

Kolejnym ważnym parametrem filtrów jest charakterystyka fazowa (przesunięcie fazowe w funkcji częstotliwości), związana z czasem opóźnienia sygnałów o różnych częstotliwościach. Jeżeli charakterystyka fazowa w paśmie przepustowym jest nieliniowa, sygnały o różnych częstotliwościach opóźnione są o różną wartość okresu, doprowadzając do zniekształcenia sygnału wyjściowego.

Oprócz charakterystyk częstotliwościowych, do opisu właściwości filtrów wykorzystuje się odpowiedź na funkcję skoku jednostkowego w dziedzinie czasu.

Możemy wyróżnić:

— czas narastania odpowiedzi – czas, w którym napięcie wyjściowe rośnie od 10% do 90% wartości końcowej,
— czas ustalania - czas, w jakim napięcie wyjściowe ustala się w zakresie 5% odchylenia od wartości końcowej.
— przerzut - maksymalna wartość, o jaką napięcie wyjściowe przewyższa chwilowo wartość końcową, po przekroczeniu czasu narastania,
— tętnienie - oscylacje wokół średniej z wartości końcowej.

Ze względu na charakterystykę amplitudową, filtry możemy podzielić na:

— dolnoprzepustowe,
— górnoprzepustowe,
— pasmowozaporowe,
— pasmowoprzepustowe.

Najczęściej spotykane są następujące realizacje filtrów: filtr Butterwortha, Czebyszewa, eliptyczny i Bessela.

Filtr Butterwortha charakteryzuje się płaską charakterystyką amplitudową w paśmie przepustowym i zaporowym (filtr maksymalnie płaski). Zostało to uzyskane kosztem małego nachylenia charakterystyki amplitudowej w obszarze przejściowym. W obszarze tym występuje również nieliniowość odpowiedzi fazowej. Odpowiedź amplitudowa filtra Butterwortha jest określona zależnością:

$$\left|\frac{V_{out}}{V_{in}}\right| = \frac{1}{\sqrt{1+\left(\frac{f}{f_{3db}}\right)^{2n}}} \tag{3.1}$$

gdzie: $f_{3db}$ – częstotliwość graniczna, przy której wzmocnienie spada o 3 dB,
$n$ – rząd filtra.

Zwiększenie rzędu filtra poprawia stromość charakterystyki amplitudowej w obszarze przejściowym, jednak wiąże się to ze zwiększeniem komplikacji układu lub zwiększeniem liczby obliczeń dla realizacji cyfrowej.

Ze względu na wymienione wady, filtr Butterwortha w praktyce stosowany jest dość rzadko.

Filtr Czebyszewa typu I charakteryzuje się niewielkimi tętnieniami w paśmie przepustowym, natomiast typu II tętnieniami w paśmie zaporowym. Filtr ten ma większe nachylenie charakterystyki amplitudowej w obszarze przejściowym niż w filtrze Butterwortha. Odpowiedź amplitudowa filtra Czebyszewa jest określona zależnością:

$$\left|\frac{V_{out}}{V_{in}}\right| = \frac{1}{\sqrt{1+\varepsilon^2 C_n{}^2\left(\frac{f}{f_{3db}}\right)}} \tag{3.2}$$

gdzie: $C_n$ – wielomian będący funkcją rzędu filtra,
$\varepsilon$ – stała określająca ilość tętnień w paśmie przepustowym.

Tętnienia w całym paśmie przepustowym są jednakowe, a ze zwiększeniem rzędu rośnie gęstość tętnień. Nieliniowość charakterystyki fazowej jest nawet większa niż filtra Butterwortha.

Filtr eliptyczny (filtr Cauera) charakteryzuje się największą stromością charakterystyki amplitudowej w obszarze przejściowym oraz tętnieniami, które występują zarówno w paśmie przepustowym, jak i zaporowym.

Ze względu na największą nieliniowość charakterystyki fazowej filtr eliptyczny może być stosowany, gdy nie stawia się wymagań w stosunku do tego parametru.

Filtr Bessela charakteryzuje się małą stromością charakterystyki amplitudowej w obszarze przejściowym, natomiast jego zaletą jest najbardziej liniowa charakterystyka fazowa.

## 3.2. Filtry cyfrowe

Filtr cyfrowy jest algorytmem lub procesem obliczeniowym powodującym transformację sygnału cyfrowego w inną sekwencję liczb. Filtry cyfrowe, w porównaniu z filtrami analogowymi, umożliwiają m.in. wyeliminowanie dryftu parametrów, wyeliminowanie problemów związanych z szumami elementów składowych, uzyskanie wysokiej dokładności, łatwość zmiany parametrów.

Algorytm realizujący filtrację cyfrową może być przedstawiony w postaci transmitancji:

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 - \sum_{k=1}^{N} a_k z^{-k}} = \frac{Y(z)}{X(z)} \tag{3.3}$$

gdzie: $z$ – zmienna zespolona,
$a_k$, $b_k$ – współczynniki filtra.

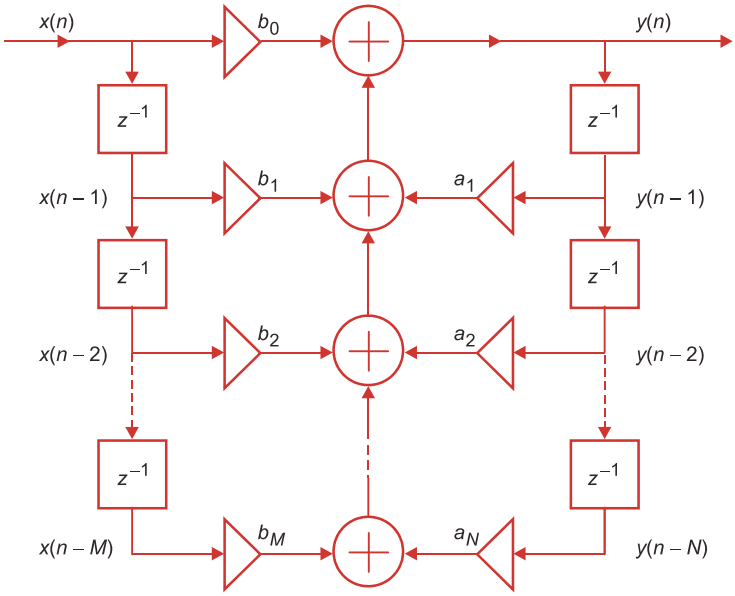
Dla dyskretnych układów liniowych związek pomiędzy dyskretnym sygnałem wejściowym (pobudzeniem) $x(n)$ a wyjściowym (odpowiedzią) $y(n)$ ma postać:

$$y(n) = \sum_{k=1}^{N} a_k y(n-k) + \sum_{k=0}^{M} b_k x(n-k) \tag{3.4}$$

Równanie (3.4) opisuje algorytm obliczeniowy, w którym opóźnione wyrazy pobudzenia mnoży się przez współczynniki $b_k$, a opóźnione wyrazy odpowiedzi przez współczynniki $a_k$, a następnie wszystkie iloczyny sumuje się.

Jeżeli przynajmniej jeden ze współczynników $a_k$ we wzorze (3.4) jest różny od zera, filtr nazywany jest rekursywnym lub filtrem o nieskończonej odpowiedzi impulsowej (IIR – *Infinite Impulse Response*). Spotyka się też polski skrót NOI.

Na rysunku 3.2 przedstawiony jest schemat blokowy układu dyskretnego, realizującego zależność 3.4. Blok oznaczony jako „$z^{-1}$” wprowadza opóźnienie o jedną próbkę.




Rys. 3.2. Schemat blokowy filtra cyfrowego IIR

Jeżeli wszystkie współczynniki $a_k$ w zależności (3.4) są równe zero, wyrażenie zostaje zredukowane do postaci:

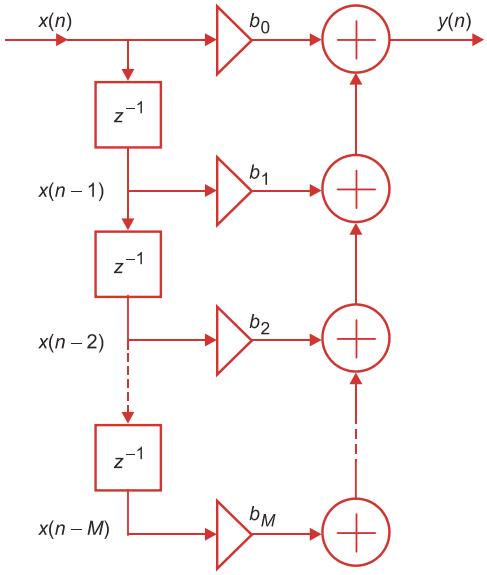
$$y(n) = \sum_{k=0}^{M} b_k x(n-k) \tag{3.5}$$

Wynika z tego, że filtr na pobudzenie pojedynczą próbką daje odpowiedź o ograniczonej długości. Jest to filtr nierekursywny, nazywany filtrem o ograniczonej odpowiedzi impulsowej (FIR – *Finite Impulse Response*). Spotyka się też polski skrót SOI.

Na rysunku 3.3 przedstawiony jest schemat blokowy układu dyskretnego, realizującego zależność 3.5.

Filtr FIR nie posiada sprzężenia zwrotnego, a każda próbka sygnału wyjściowego jest średnią ważoną próbek sygnału wejściowego.

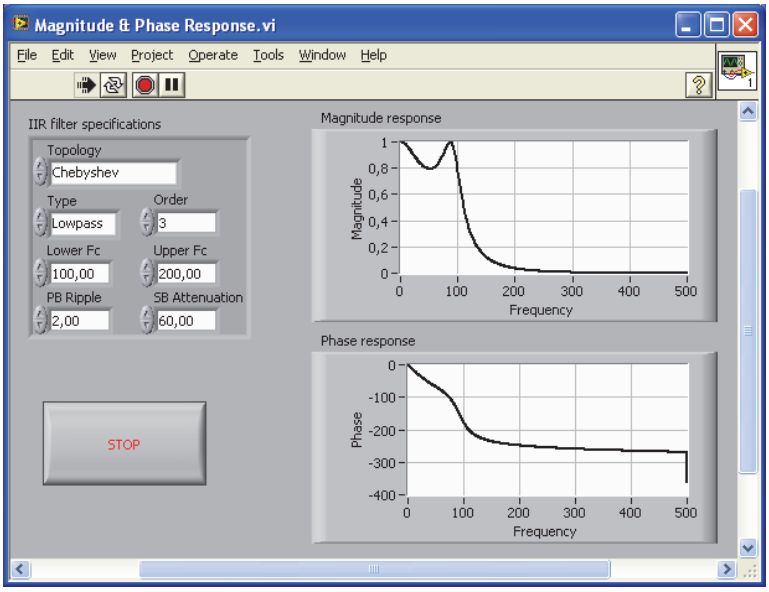
Zaletą filtrów FIR jest liniowa charakterystyka fazowa, jaka wymagana jest w niektórych zastosowaniach. Inną zaletą filtrów FIR jest ich stabilność – transmitancja filtrów FIR nie posiada biegunów, dlatego filtry te są zawsze stabilne. Wadą filtrów IIR jest nieliniowa charakterystyka fazowa. Filtr IIR może być niestabilny, zależy to od położenia biegunów. Zaletą filtrów IIR jest to, że dla uzyskania podobnego efektu filtrowania w porównaniu z filtrami FIR działają szybciej i nie wymagają specjalnej pamięci (operacja filtrowania wykonywana jest w tym samym obszarze pamięci).




Rys. 3.3. Schemat blokowy filtra cyfrowego FIR

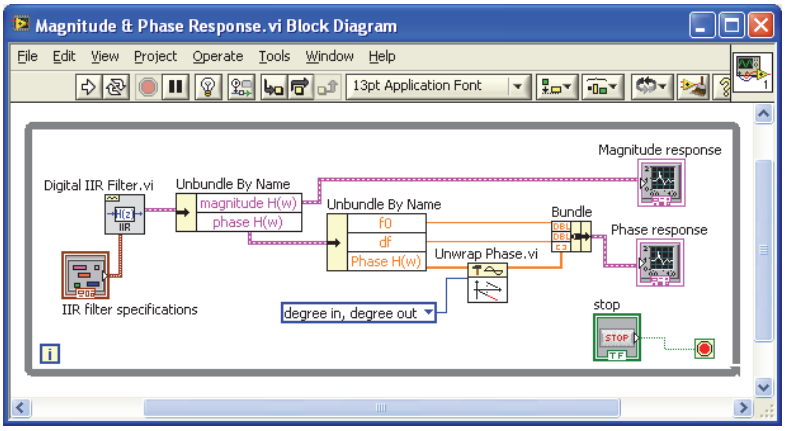
## 3.3. Programy do badania działania filtrów cyfrowych

Na rysunku 3.4 przedstawiony jest panel, a na rysunku 3.5 diagram programu w LabVIEW, przeznaczonego do sprawdzenia charakterystyk amplitudowych i fazowych filtra o nieskończonej odpowiedzi impulsowej IIR.




Rys. 3.4. Panel programu z charakterystyką amplitudową i fazową filtrów

Jedną z metod projektowania takich filtrów jest metoda biliniowa, polegająca na projektowaniu filtrów analogowych i ich transformacji do postaci cyfrowej. Przykładami filtrów cyfrowych o znanych z filtrów analogowych realizacjach są: filtr Butterwortha, Czebyszewa, eliptyczny i Bessela.

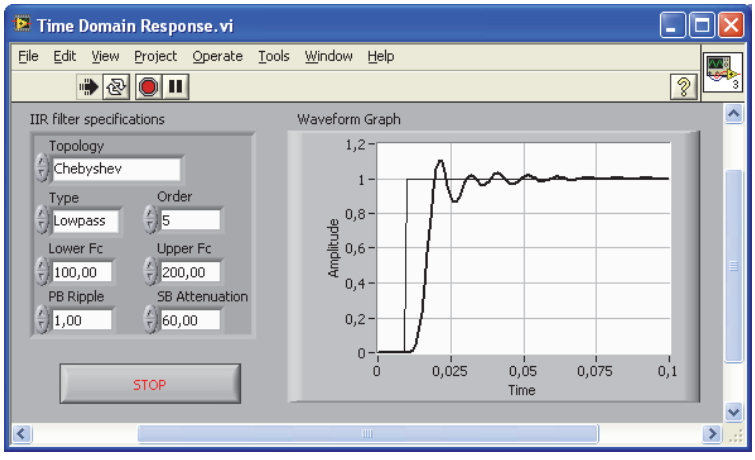



Rys. 3.5. Diagram programu z charakterystyką amplitudową i fazową filtrów

W programie wykorzystana została funkcja *Digital IIR Filter* (*Functions* → *Signal Processing* → *Waveform Conditioning*). Po ustawieniu narzędzia do łączenia elementów na zacisku *IIR filter specifications* należy nacisnąć prawy przycisk myszy i wybrać polecenie *Create* ► *Control*. Do rozdzielenia sygnału wyjściowego na charakterystykę amplitudową i fazową można wykorzystać funkcję *Unbundle by Name* (*Functions* → *Programming* → *Cluster, Class, & Variant*). Funkcja *Unwrap Phase* służy do usunięcia nieciągłości fazy.

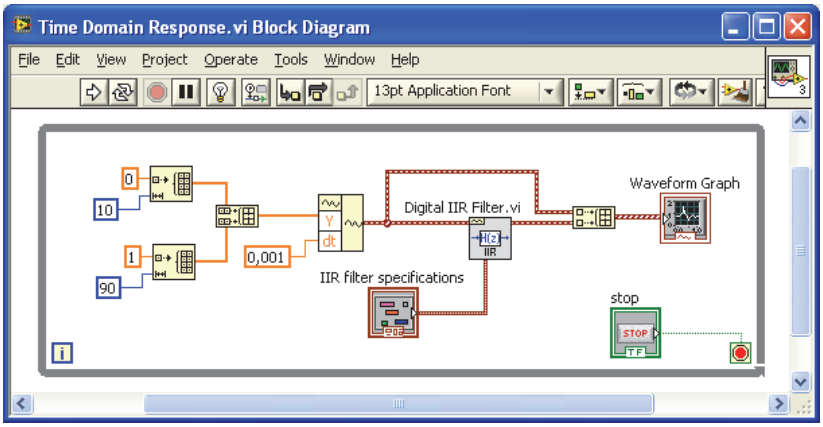
Program pozwala na zapoznanie się z charakterystykami filtrów dla różnych realizacji i różnych parametrów.

Kolejny program przedstawia odpowiedź filtra na funkcję skoku jednostkowego w dziedzinie czasu. Panel tego programu pokazany jest na rysunku 3.6, natomiast diagram na rysunku 3.7.

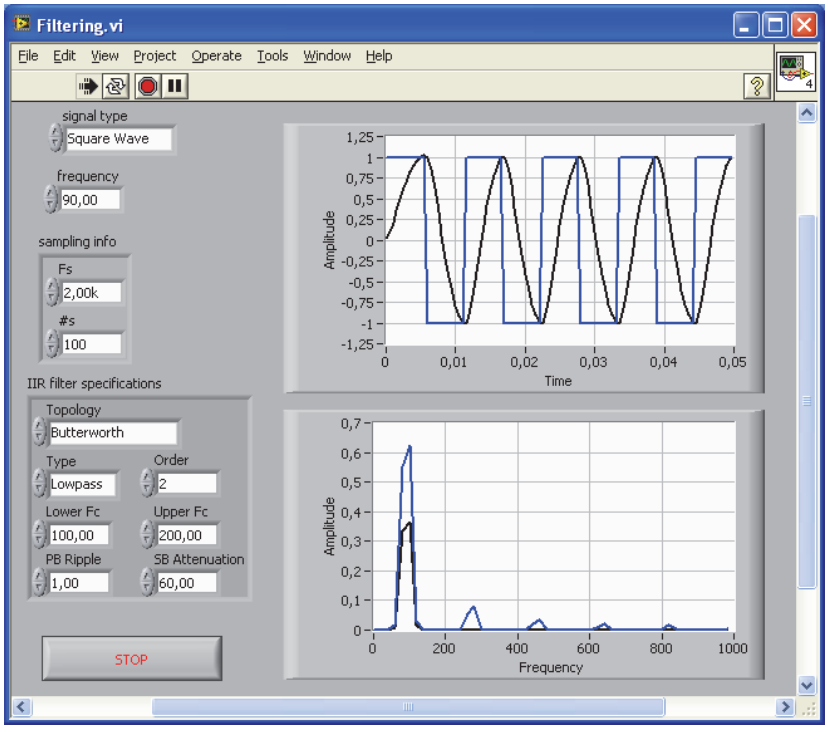



Rys. 3.6. Panel programu z odpowiedzią filtra na skok jednostkowy

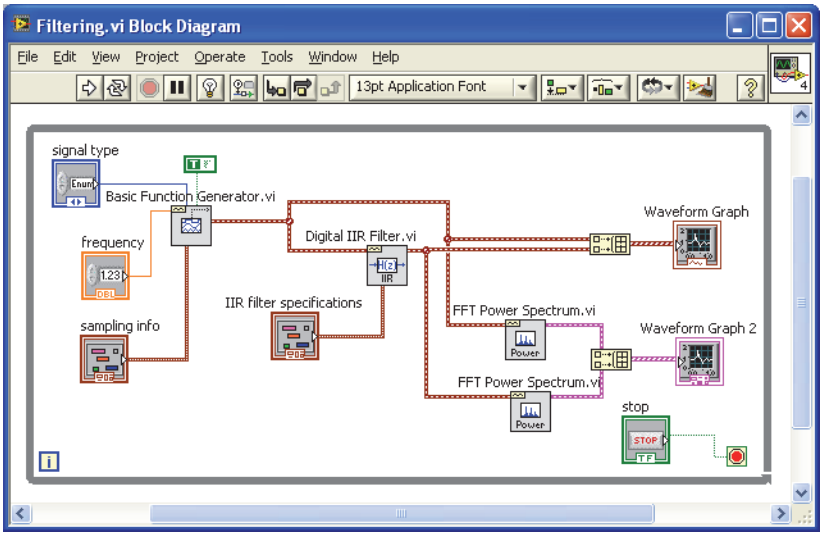
Program pozwala na wyznaczenie czasu narastania odpowiedzi, czasu ustalania, przerzutu i tętnienia dla różnych realizacji filtra i różnych parametrów.




Rys. 3.7. Diagram programu z odpowiedzią filtra na skok jednostkowy




Rys. 3.8. Panel programu do analizy działania filtrów




Rys. 3.9. Diagram programu do analizy działania filtrów

Następny program służy do zapoznania się z działaniem filtrów. Jego panel pokazany jest na rysunku 3.8, natomiast diagram na rysunku 3.9.

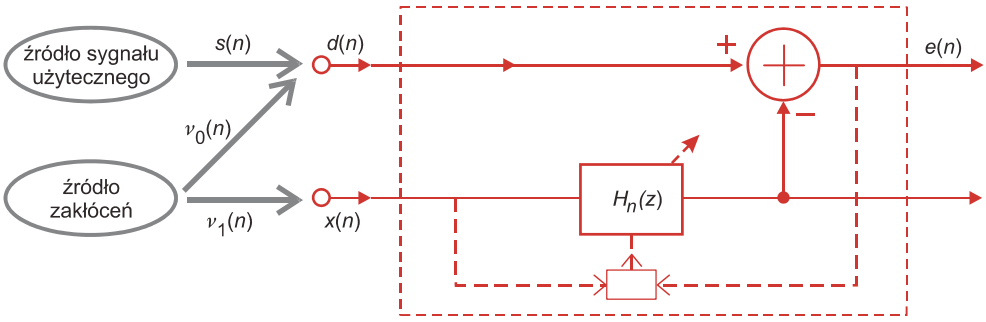
Wykresy na panelu programu przedstawiają przebieg sygnału w dziedzinie czasu na wejściu filtra i jego wyjściu oraz widmo częstotliwościowe sygnału również na wejściu i wyjściu filtra. Program pozwala na sprawdzenie działania różnych rodzajów filtrów (Butterwortha, Czebyszewa, eliptycznego, Bessela), różnych typów (dolnoprzepustowego, górnoprzepustowego, pasmowoprzepustowego, pasmowozaporowego), wpływu zadanego rzędu filtra oraz wpływu nastawionej wartości dolnej i górnej częstotliwości granicznej. W programie można zadawać różne kształty sygnału pobudzającego.

# Filtracja adaptacyjna

## 4.1. Wprowadzenie do filtracji adaptacyjnej

W rozdziale 3. przedstawiono filtry cyfrowe, których współczynniki są niezmienne w czasie. W technice filtracji znane są również filtry adaptacyjne, których parametry są automatycznie dostosowywane do zmian parametrów sygnału. Filtracja adaptacyjna może być stosowana w identyfikacji, predykcji sygnału lub przy usuwaniu szumu.

Przykładowa struktura blokowa filtra adaptacyjnego do usuwania zakłóceń pokazana jest na rysunku 4.1.




Rys. 4.1. Schemat blokowy filtra adaptacyjnego

Na pierwsze wejście doprowadzony jest sygnał filtrowany $d(n)$. Jest on sumą sygnału użytecznego $s(n)$ oraz sygnału zakłócającego $v_0(n)$.

$$d(n) = s(n) + v_0(n) \tag{4.1}$$

Na drugie wejście doprowadzony jest sygnał dodatkowy (odniesienia) $x(n)$, pochodzący ze źródła zakłóceń.

$$x(n) = v_1(n) \tag{4.2}$$

Przykładem może być wykorzystanie zestawu głośnomówiącego w trakcie jazdy samochodem. Źródłem sygnału filtrowanego jest mikrofon rejestrujący mowę, ale również hałas otoczenia (dźwięk pracy silnika). Źródłem sygnału odniesienia jest dodatkowy mikrofon zamontowany w innym miejscu, rejestrujący tylko odgłos pracy silnika. Sygnały $v_0(n)$ i $v_1(n)$ są ze sobą skorelowane, ale ze względu na różne umieszczenie czujników i właściwości toru pomiarowego mogą się różnić amplitudą i fazą. Dlatego nie można w prosty sposób wykorzystać sygnału odniesienia do wydzielenia zakłóceń z sygnału rejestrowanego $d(n)$.

Zadaniem filtra $H_n(z)$, którego transmitancja może zmieniać się w czasie, jest takie przetworzenie sygnału odniesienia $v_1(n)$, aby sygnał $y(n)$ jak najwierniej odzwierciedlał składową zakłócającą $v_0(n)$ (sumującą się z sygnałem użytecznym) w sensie przyjętego kryterium błędu. Następnie sygnał $y(n)$ jest odejmowany od sygnału $d(n)$.

W wyniku filtracji otrzymujemy wyjściowy sygnał błędu $e(n)$, odpowiadający sygnałowi mierzonemu po usunięciu zakłóceń.

$$e(n) = s(n) + v_0(n) - y(n) \tag{4.3}$$

Celem obliczeń jest wyznaczenie współczynników transmitancji $H_n(z)$. Dla filtra adaptacyjnego rekursywnego typu IIR mamy postać:

$$H_n(z) = \frac{b_0(n) + b_1(n)z^{-1} + b_2(n)z^{-2} + ... + b_M(n)z^{-M}}{1 + a_1(n)z^{-1} + a_2(n)z^{-2} + ... + a_N(n)z^{-N}} \tag{4.4}$$

Ponieważ filtr IIR posiada sprzężenie zwrotne i może być z tego powodu niestabilny, w praktyce częściej stosuje się w filtracji adaptacyjnej nierekursywny filtr FIR. Równanie (4.4) przyjmuje postać (parametry filtra oznaczono symbolem $h$):

$$H_n(z) = h_0(n) + h_1(n)z^{-1} + h_2(n)z^{-2} + ... + h_M(n)z^{-M} \tag{4.5}$$

Sygnał $y(n)$ na wyjściu filtra jest opisany zależnością:

$$y(n) = \sum_{k=0}^{M} h_k(n)x(n-k) \tag{4.6}$$

Błąd $e(n)$ dopasowania sygnału $y(n)$ do sygnałów $x(n)$ i $d(n)$ jest funkcją współczynników filtra.

Znane są różne algorytmy adaptacyjne. Wybór algorytmu uzależniony jest m.in. od prędkości zbieżności, niedopasowania, odporności (*robustness*), złożoności obliczeniowej.

Jednym z częściej stosowanych algorytmów adaptacyjnych, ze względu na łatwość implementacji, jest algorytm najmniejszej średniej kwadratowej LMS (*Least Mean Squares*). Znane są różne typy tego algorytmu, np. znormalizowany LMS, nieszczelny LMS, LMS z błędem znaku.

W filtrach adaptacyjnych LMS jest minimalizowana wartość chwilowa błędu:

$$J = e^2(n) \tag{4.7}$$

Dla tego filtra otrzymuje się równanie, z którego można wyznaczyć zmodyfikowany wektor współczynników filtra $\mathbf{h}(n+1)$ na podstawie poprzedniej wartości tego wektora oraz współczynnika skalującego $\mu$, sygnału błędu $e(n)$ oraz sygnału odniesienia $x(n)$:

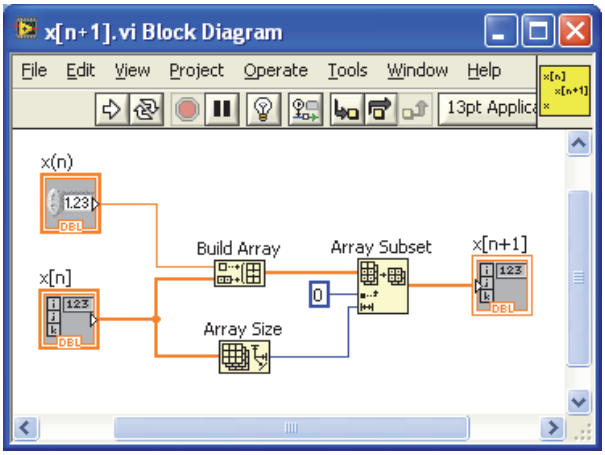
$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu \cdot e(n) \cdot \mathbf{x}(n) \tag{4.8}$$

Współczynnik skalujący $\mu$ ma wpływ na szybkość przestrajania. Duża wartość współczynnika umożliwia szybsze przestrajanie, ale powoduje niebezpieczeństwo „przeskoczenia" rozwiązania.

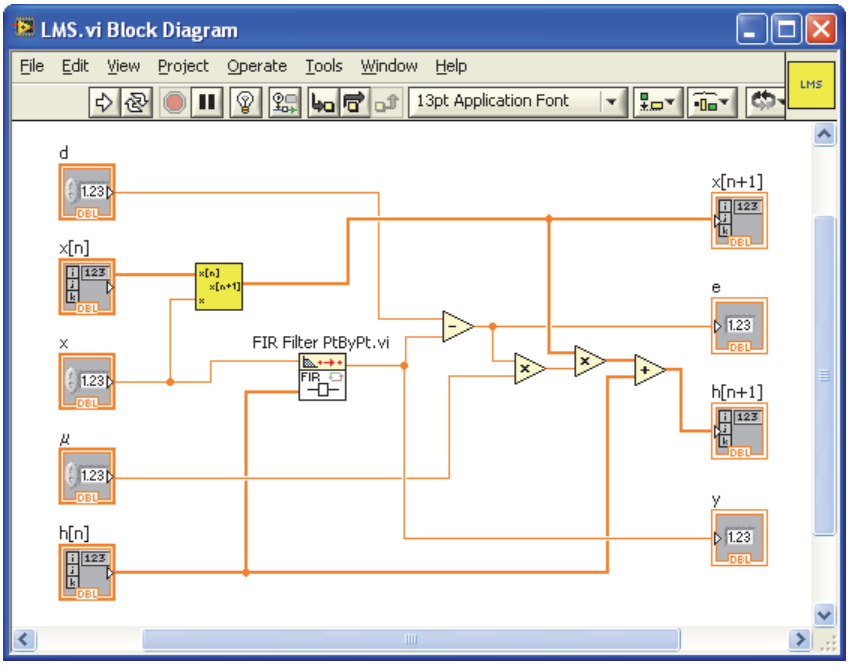
## 4.2. Program do filtracji adaptacyjnej

W przedstawionym programie przygotowanym w LabVIEW, do sygnału użytecznego o zadanej częstotliwości dodawane są zakłócenia o zmieniającym się losowo paśmie, filtrowane za pomocą filtra adaptacyjnego LMS.

Na rysunku 4.2 przedstawiony jest diagram podprogramu, który na początku tablicy wejściowej dodaje komórkę z nową wartością i jednocześnie usuwa ostatnią komórkę.

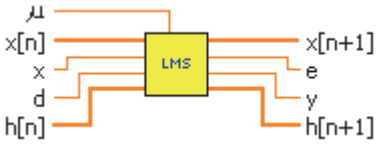



Rys. 4.2. Diagram podprogramu dodającego nowy element do tablicy




Rys. 4.3. Diagram podprogramu filtra LMS

Do nowego elementu dodajemy tablicę za pomocą funkcji *Build Array*. Ostatni element usuwamy za pomocą funkcji *Array Subset*, który podaje na wyjściu tablicę o liczbie elementów równej liczbie elementów tablicy wejściowej podprogramu (odczytanej za pomocą funkcji *Array Size*).




Rys. 4.4. Ikona i zaciski podprogramu filtra LMS

Diagram podprogramu filtra LMS pokazany jest na rysunku 4.3, a ikona z zaciskami na rysunku 4.4.

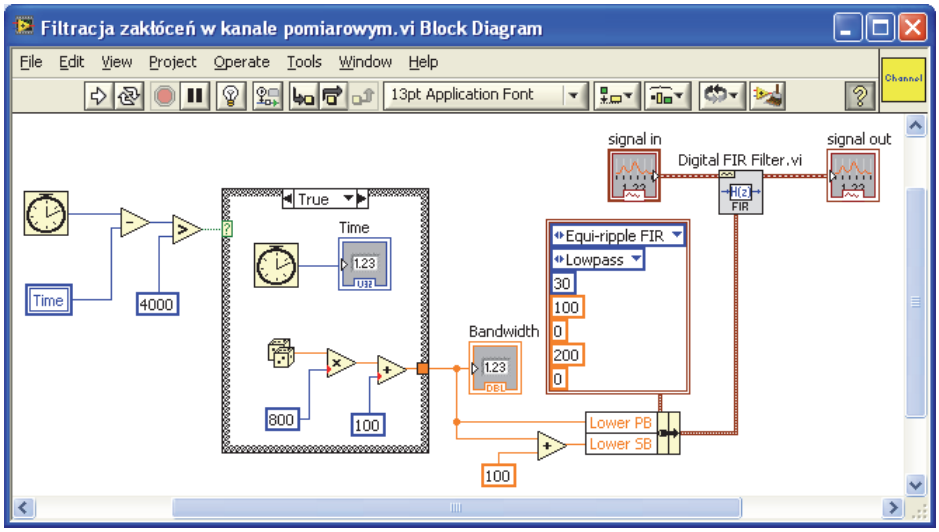
Zgodnie z rysunkiem 4.1 od składowej $d(n)$ odejmowana jest składowa $y(n)$, otrzymana przez odfiltrowanie składowej $x(n)$. Jak wcześniej podano, wykorzystujemy do tego celu filtr FIR (funkcja *FIR Filter PtByPt* z *Functions → Signal Processing → Point by Point → Filters*). W wyniku odejmowania otrzymujemy wyjściowy sygnał błędu $e(n)$.

Zgodnie z (4.8) po pomnożeniu sygnału błędu przez współczynnik skalujący $\mu$ oraz przez sygnał odniesienia $x(n)$ oraz po dodaniu wektora współczynników filtra $\mathbf{h}(n)$ z poprzedniego kroku, otrzymujemy zmodyfikowany wektor współczynników filtra $\mathbf{h}(n+1)$.

Na koniec należy utworzyć ikonę i zaciski (np. jak na rys. 4.4), by podprogram można było wykorzystać w programie głównym.

W głównym programie wykorzystano jeszcze jeden podprogram, służący do symulacji zniekształcenia zakłóceń w kanale pomiarowym.

Sygnał zakłócający $v_1(n)$ jest sumą dwóch przebiegów sinusoidalnych, ale załóżmy, że w wyniku działania toru pomiarowego, składowa zakłócająca $v_0(n)$, która dodaje się do sygnału użytecznego, ma odfiltrowane wyższe częstotliwości, z nieznaną częstotliwością graniczną. Diagram podprogramu symulującego tor pomiarowy pokazany jest na rysunku 4.5.




Rys. 4.5. Diagram podprogramu symulującego filtrację w torze pomiarowym

Sygnał zakłócający filtrowany jest za pomocą filtra dolnoprzepustowego FIR (*Digital FIR Filter* z palety *Functions* → *Signal Processing* → *Waveform Conditioning*), którego parametry zadawane są za pomocą stałej *Cluster*, dołączonej do zacisku *FIR filter specifications*. Za pomocą *Bundle By Name* (*Functions* → *Programming* → *Cluster, Class, & Variant*) zmieniane są częstotliwości graniczne obszaru przejściowego, przy jego niezmiennej szerokości 100 Hz.

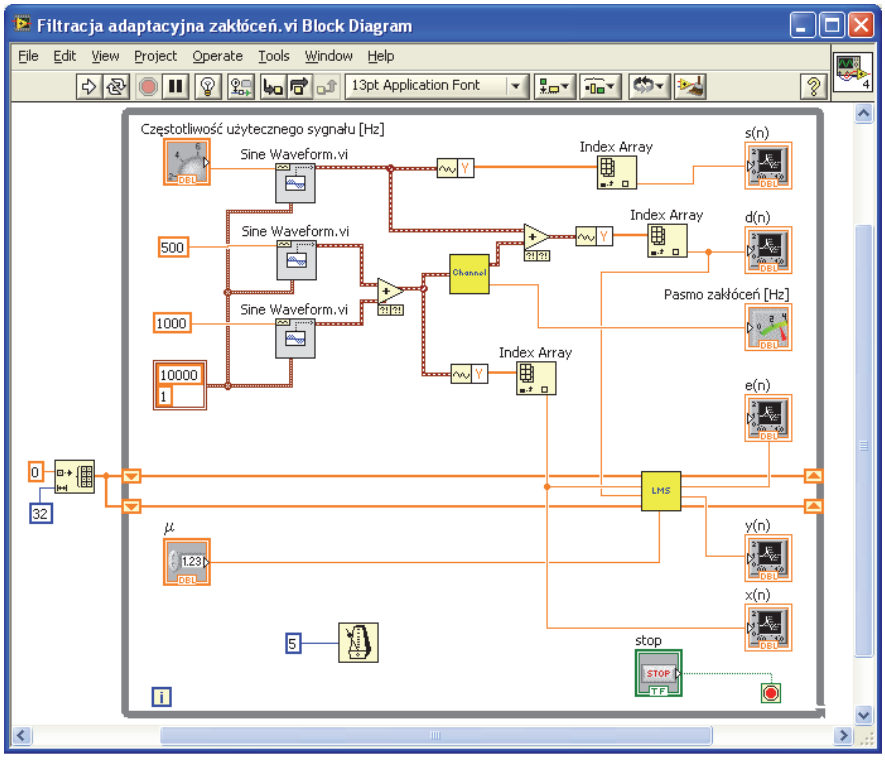
Częstotliwość graniczna filtra zmieniana jest losowo co 4 s. Realizuje się to przez odmierzanie czasu za pomocą funkcji *Tick Count* i sprawdzanie, czy od ostatniej zmiany częstotliwości upłynął zadany czas. Jeżeli tak, wykonywany jest fragment programu w ramce *True* struktury *Case*. Zapamiętywany jest tam nowy czas, od którego realizuje się jego upływ, a za pomocą funkcji generatora liczb pseudolosowych *Random Number*, po wymnożeniu przez pierwszą stałą i dodaniu drugiej, otrzymujemy nową częstotliwość graniczną filtra w zakresie 100–900 Hz.

W ramce *False* struktury *Case* należy jeszcze umieścić zmienną lokalną kontrolki *Bandwidth*. Zmienną lokalną tworzymy, ustawiając kursor na ikonie kontrolki, naciskając prawy przycisku myszy i wybierając polecenie *Create* ► *Local Variable*. Poleceniem *Change to Read* można zmienić tryb zmiennej lokalnej do odczytu. Utworzoną zmienną łączymy z zaciskiem na prawej ramce struktury *Case*.

W podobny sposób należy utworzyć zmienną lokalną kontrolki *Time*.

Dla gotowego podprogramu należy utworzyć zacisk wejściowy *signal in* oraz dwa zaciski wyjściowe: *signal out* oraz *Bandwidth*.

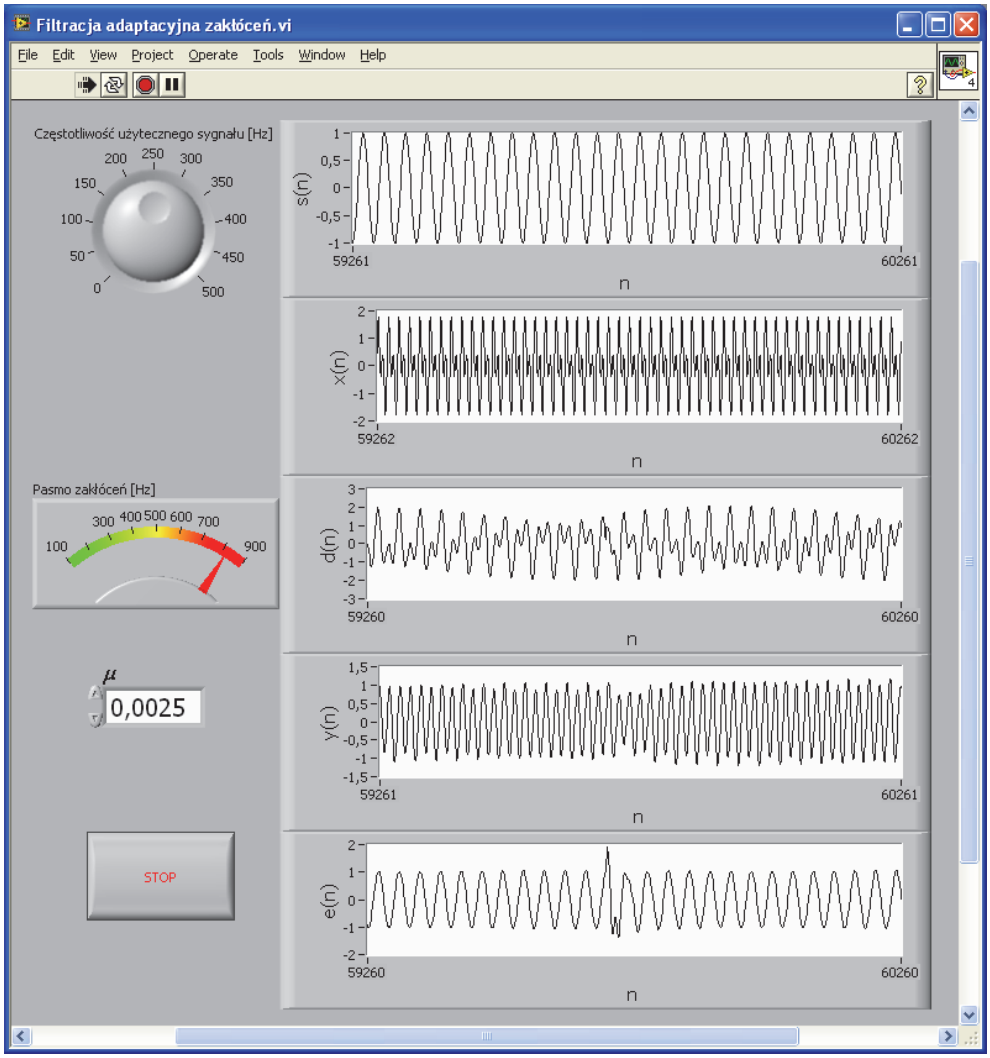
Po przygotowaniu podprogramów, można przystąpić do przygotowania programu głównego. Diagram tego programu pokazany jest na rysunku 4.6.



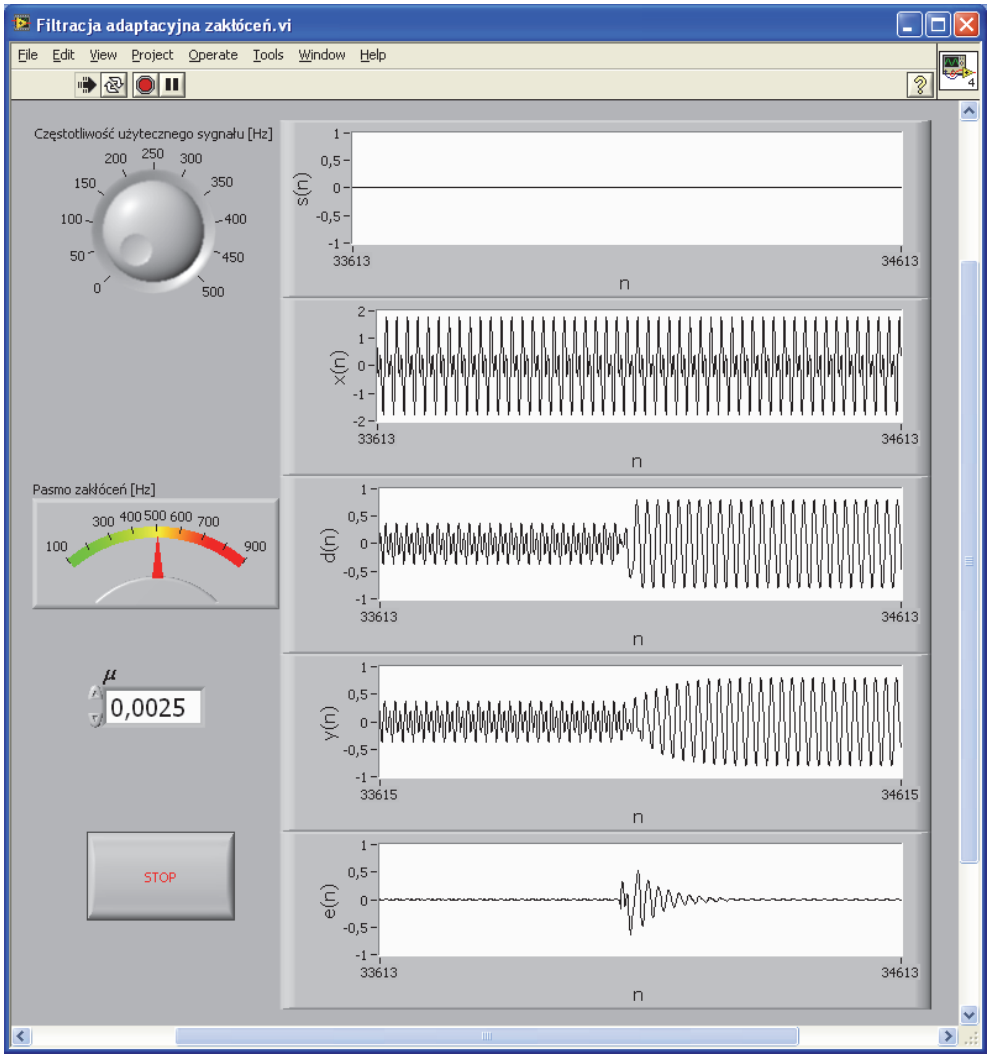

Rys. 4.6. Diagram programu adaptacyjnej filtracji zakłóceń

Sygnałem użytecznym $s(n)$ jest sygnał sinusoidalny z generatora *Sine Waveform* (*Functions* → *Signal Processing* → *Waveform Generation*) o zadawanej pokrętłem na panelu częstotliwości. Jak już wspomniano, sygnał zakłócający jest sumą dwóch przebiegów sinusoidalnych, o częstotliwościach 500 Hz i 1000 Hz. Do zacisku *sampling info* dołączamy klaster, w którym podajemy częstotliwość próbkowania oraz liczbę próbek generowanych przy jednym wykonaniu funkcji (1 próbka).

Do zasymulowania zmiany parametrów sygnału zakłóceń w torze pomiarowym wykorzystano opisany podprogram. Sygnał wyjściowy tego podprogramu zsumowany jest z sygnałem użytecznym, otrzymując w ten sposób sygnał mierzony $d(n)$.




Rys. 4.7. Panel programu adaptacyjnej filtracji zakłóceń




Rys. 4.8. Panel programu adaptacyjnej filtracji zakłóceń dla $s(n) = 0$

Ponieważ zmienne na wyjściu generatorów mają format *waveform*, w celu uzyskania pojedynczej próbki przy każdym wykonaniu pętli, sygnał ten łączymy z funkcją *Get Waveform Components* (*Functions* → *Programming* → *Waveform*) oraz *Index Array* (*Functions* → *Programming* → *Array*).

Do przedstawienia zmiany sygnałów w charakterystycznych punktach układu należy wykorzystać wykresy *Waveform Chart*.

Panel programu z przykładowymi wynikami pokazany jest na rysunku 4.7 i 4.8.

Kolejne wykresy przedstawiają:

$s(n)$ – sygnał użyteczny o zadawanej częstotliwości, który powinien być odtworzony w procesie filtracji adaptacyjnej,

$x(n)$ – sygnał zakłócający na wejściu odniesienia (suma dwóch przebiegów sinusoidalnych o stałych częstotliwościach),

$d(n)$ – mierzony sygnał zakłócony, zgodnie ze wzorem (4.1),
$y(n)$ – sygnał $x(n)$ odfiltrowany za pomocą filtra $H_n(z)$,
$e(n)$ – wyjściowy sygnał błędu, przy idealnej filtracji byłoby $e(n) = s(n)$.

W przedstawionym na rysunku 4.7 przykładzie można zobaczyć, że w chwili, gdy zmieniła się częstotliwość graniczna filtra symulującego kanał pomiarowy (zmiana przebiegu $d(n)$), przez krótką chwilę sygnał $e(n)$ odbiegał od $s(n)$. Po przestrojeniu filtra system działał dalej prawidłowo.

Jeżeli sygnał użyteczny jest równy zero ($s(n) = 0$), to wyjściowy sygnał błędu $e(n)$ jest równy różnicy między mierzonym sygnałem zakłóconym $d(n)$ oraz odfiltrowanym sygnałem odniesienia $y(n)$ (rys. 4.8).

Przedstawiony program pozwala na zapoznanie z działaniem filtracji adaptacyjnej przy różnych częstotliwościach sygnału użytecznego. Umożliwia również zbadanie działania algorytmu dla różnych wartości współczynnika skalującego $\mu$.

# Analiza czasowo-częstotliwościowa

Analiza częstotliwościowa umożliwia określenie składowych występujących w widmie częstotliwościowym. Nie pozwala natomiast na analizę sygnałów niestacjonarnych, w których częstotliwości lub amplitudy poszczególnych składowych zmieniają się w czasie. Do takiego celu można wykorzystać analizę czasowo-częstotliwościową *t|f* (*JTFA – Joint Time-Frequency Analysis*).

W cyfrowym przetwarzaniu sygnałów znanych jest wiele różnych transformacji *t|f*, różniących się właściwościami: transformacja Gabora, krótkookresowa transformacja Fouriera, transformacja falkowa, transformacja Wignera-Ville'a, rodzina transformacji klasy Cohena.

## 5.1. Krótkookresowa transformata Fouriera STFT

Zasada krótkookresowej transformaty Fouriera (STFT – *Short-Time Fourier Transform*) polega na podziale sygnału na segmenty, z których każdy niezależnie podlega analizie widmowej. Okno czasowe przesuwane jest o mały, stały odcinek czasu wzdłuż całego sygnału (rys. 5.1). Łącząc otrzymane w ten sposób widma, otrzymujemy wykres, na którego osiach jest czas i częstotliwość, trzecia oś, określana przez intensywność kolorów, podaje informację o amplitudzie.

Podobnie jak dla tradycyjnej analizy Fouriera, również tutaj stosuje się okna wygładzające, pozwalające wyeliminować ostre przejścia na krańcach przedziałów.

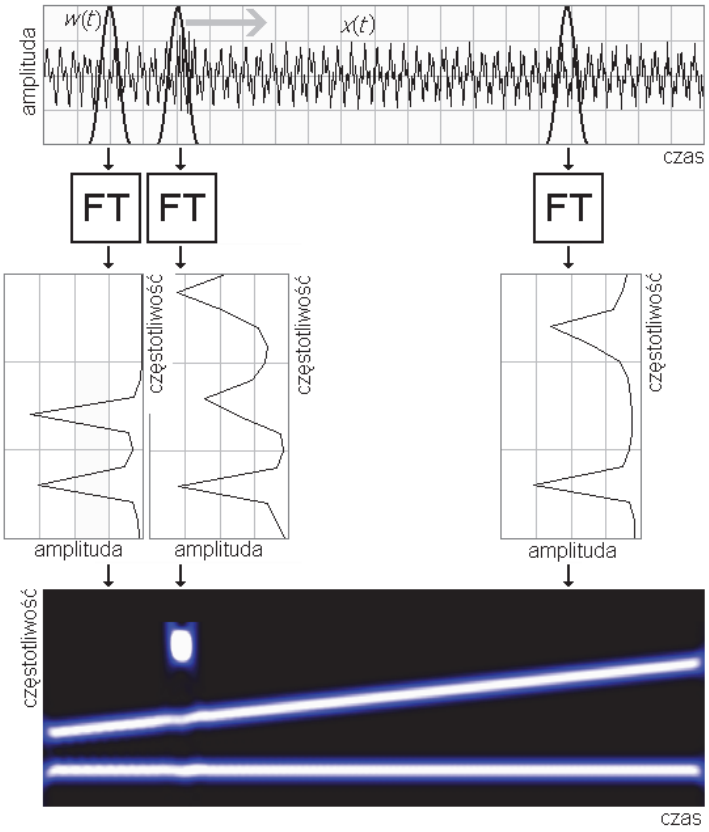
Krótkookresową transformatę Fouriera sygnału $x(t)$, przy użyciu okna $w(t)$ w pozycji $(\tau, \xi)$ można zdefiniować jako:

$$X(\tau,\xi) = \int_{-\infty}^{+\infty} x(t) \cdot w(t-\tau) \cdot e^{-j\xi t} dt \tag{5.1}$$

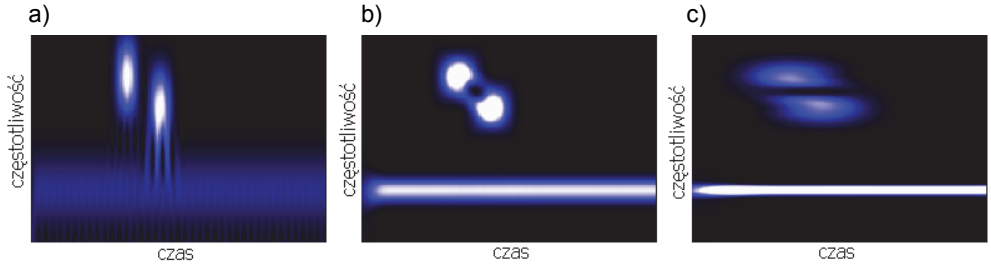
Z zasady nieoznaczoności wynika, że iloczyn szerokości pasma częstotliwościowego i czasu sygnału nie może być mniejszy od pewnej minimalnej wartości. Dlatego, zwiększając długość okna, zwiększamy rozdzielczość w dziedzinie częstotliwości, ale pogarszamy rozdzielczość w dziedzinie czasu. Przy małej długości okna rozdzielczość w dziedzinie czasu będzie większa, ale rozdzielczość w dziedzinie częstotliwości będzie gorsza.

Przykład wpływu długości okna na wynik analizy czasowo-częstotliwościowej przedstawiony jest na rysunku 5.2. Analizowany sygnał zawiera składową o stałej częstotliwości w całym analizowanym przedziale czasu oraz dwa zaburzenia o wyższych częstotliwościach przesunięte w czasie. Dla krótkiego okna na rysunku 5.2a można określić chwile

wystąpienia zaburzeń, ale nie można dokładnie oszacować ich częstotliwości. Dla dłuższego okna na rysunku 5.2c można określić częstotliwości wszystkich składowych, nie można natomiast dokładnie określić lokalizacji zaburzeń. Wynik analizy dla pośrednich warunków pokazany jest na rysunku 5.2b.




Rys. 5.1. Ilustracja krótkookresowej transformaty Fouriera

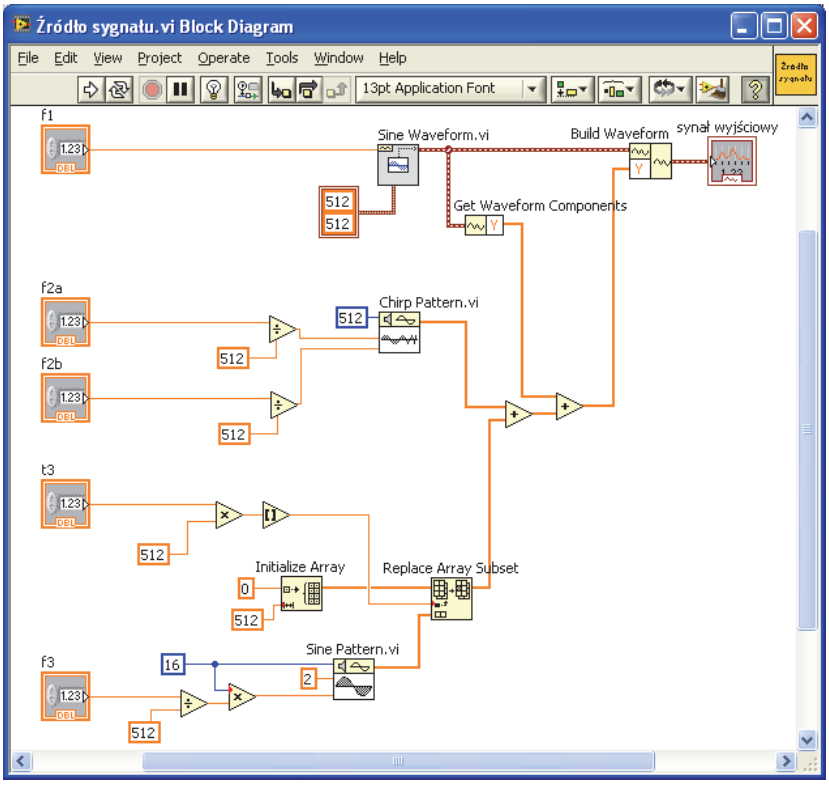



Rys. 5.2. Wpływ długości okna na wynik analizy czasowo-częstotliwościowej

## 5.2. Realizacja analizy STFT

W rozdziale tym przedstawiony jest program w LabVIEW, który pozwala na zapoznanie się z analizą STFT.

Rozpoczynamy od przygotowania podprogramu do generacji sygnału, który zostanie przetworzony za pomocą algorytmu STFT. Sygnał zawiera trzy składowe: o stałej częstotliwości $f_1$, o częstotliwości zmieniającej się liniowo od wartości $f_{2a}$ do $f_{2b}$ oraz krótkiego impulsu o częstotliwości $f_3$ w chwili $t_3$. Diagram podprogramu pokazany jest na rysunku 5.3.

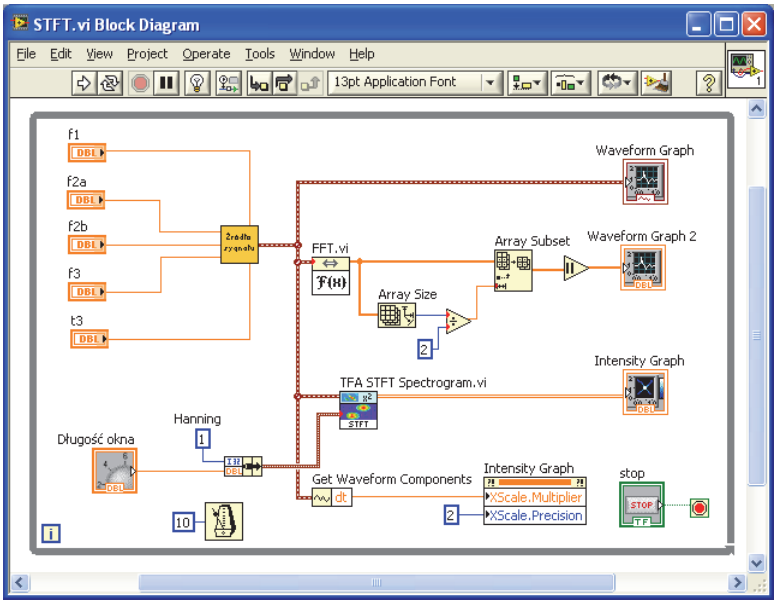



Rys. 5.3. Diagram podprogramu generującego sygnał do analizy STFT

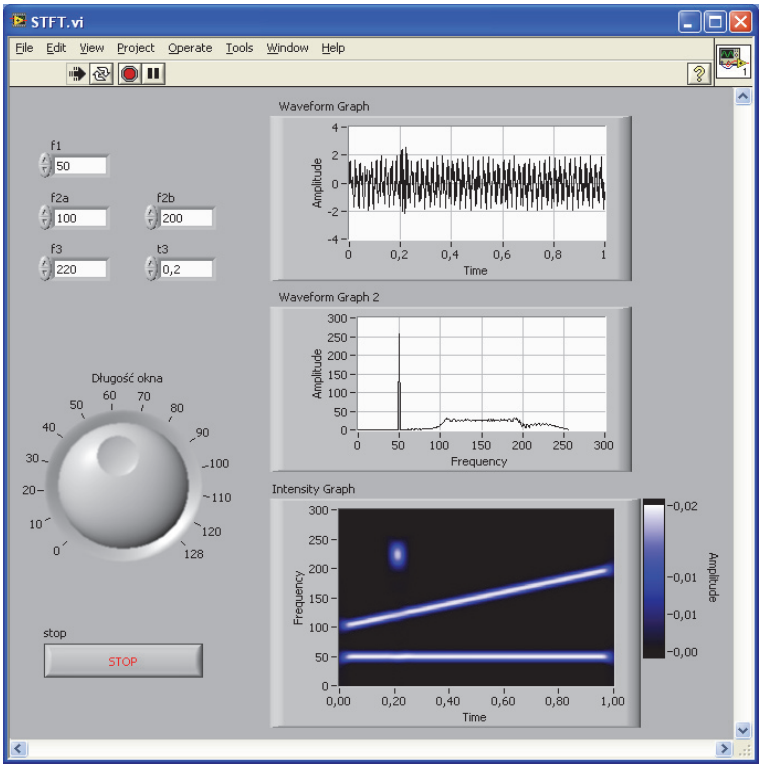
Przyjmijmy, że wygenerowany ciąg zawiera 512 próbek. Przebieg sinusoidalny generowany jest za pomocą funkcji *Sine Waveform*. W ustawieniach próbkowania (*sampling info*) podajemy częstotliwość próbkowania 512 Hz oraz liczbę próbek równą 512. Dzięki takim ustawieniom symulujemy długość okna czasowego równą 1 s.

Do wygenerowania przebiegu o zmiennej częstotliwości można wykorzystać funkcję *Chirp Pattern*.

Ostatnia składowa to impuls o czasie trwania (1 s)·16/(512 Hz) = 31,25 ms i zadanym położeniu. Pozostałe 512 – 16 = 496 próbki mają wartość zero. Tablica z elementami o wartości 0 generowana jest funkcją *Initialize Array*, a przy użyciu funkcji *Replace Array Subset* wymieniamy 16 próbek przebiegu sinusoidalnego.




Rys. 5.4. Diagram programu do analizy STFT

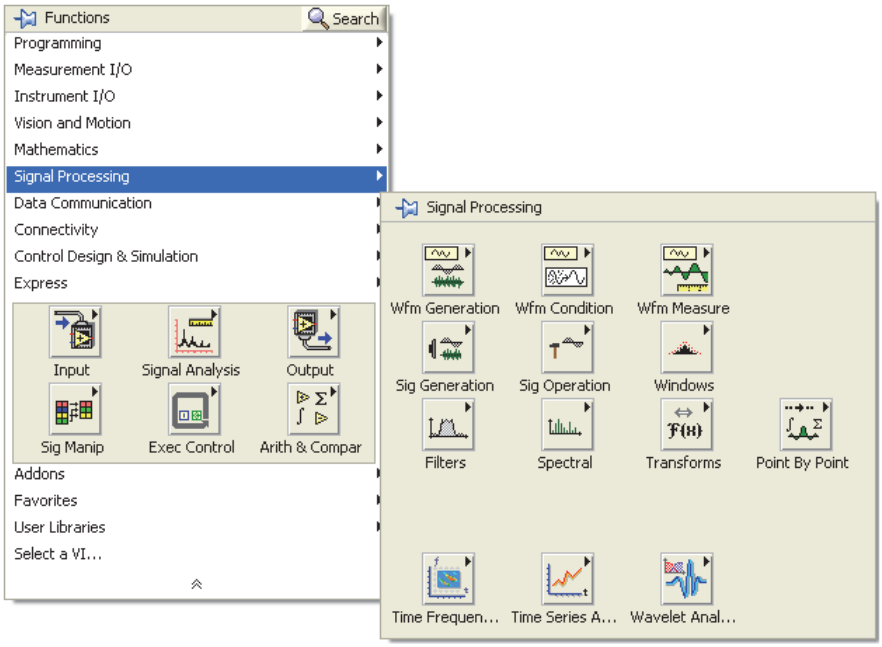



Rys. 5.5. Panel programu do analizy STFT

Po przygotowaniu podprogramu należy utworzyć 5 zacisków wejściowych do zadawania parametrów sygnału i jeden zacisk wyjściowy.

Dla wygenerowanego przebiegu za pomocą przedstawionego podprogramu zostanie wykonana analiza w dziedzinie czasu, częstotliwości oraz analiza czasowo-częstotliwościowa. Diagram służącego do tego celu programu pokazany jest na rysunku 5.4, natomiast panel na rysunku 5.5.

W programie wykorzystano funkcję *TFA STFT Spectrogram*, która dostępna jest po zainstalowaniu dodatkowej biblioteki *Advanced Signal Processing Toolkit*. Funkcje zawarte w tej bibliotece podzielone są na trzy grupy: *Time Frequency Analysis*, *Time Series Analysis* oraz *Wavelet Analysis* (rys. 5.6).




Rys. 5.6. Paleta *Functions* po zainstalowaniu biblioteki *Advanced Signal Processing Toolkit*

Do wejścia *signal* funkcji *TFA STFT Spectrogram* należy podłączyć przebieg z przygotowanej wcześniej funkcji. Wejście *window info* łączymy z wyjściem funkcji *Bundle*, na której zaciski podajemy numer wybranego okna wygładzającego (0 – bez okna wygładzającego, 1 – Hanning, 2 – Hamming, …) oraz liczbę próbek okna czasowego analizy STFT.

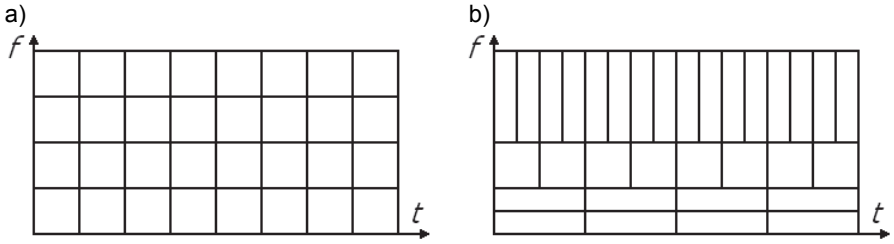
Przebiegi w dziedzinie czasu i częstotliwości zostaną przedstawione na wykresach *Waveform Graph*, natomiast wyniki analizy czasowo-częstotliwościowej na wykresie *Intensity Graph* (*Controls* → *Modern* → *Graph*). Do przeskalowania osi czasu tego wykresu można użyć *Property Node*. Po ustawieniu kursora na zacisku wykresu i naciśnięciu prawego przycisku myszy wybieramy polecenie *Create* ► *Property Node* ► *X Scale* ► *Offset and Multiplier* ► *Multiplier*. Poleceniem *Change To Write* zmieniamy kierunek wprowadzania informacji do zacisku. W podobny sposób tworzymy *Property Node* do zadania liczby miejsc po przecinku skali czasu (*X Scale* ► *Display Format* ► *Precision*). Po ustawieniu kursora na wykresie na panelu i naciśnięciu prawego przycisku myszy, wybieramy jeszcze polecenie automatycznego skalowania *Z Scale* ► *AutoScale Z*.

Po przygotowaniu i uruchomieniu programu można stwierdzić, że analiza częstotliwościowa nie daje zadowalających informacji o składowych, które zmieniają swoją częstotliwość w trakcie pomiaru. W takiej sytuacji rozwiązaniem jest właśnie zastosowanie analizy czasowo-częstotliwościowej.

Program pozwala na sprawdzenie dla różnych częstotliwości generowanego sygnału wpływu długości okna czasowego na wyniki analizy czasowo-częstotliwościowej.

## 5.3. Analiza falkowa

Problem rozdzielczości związany z wykorzystaniem STFT (rys. 5.7a) można ominąć przez zastosowanie analizy falkowej, w której dla różnych częstotliwości wykorzystywane są okna o różnej długości (rys. 5.7b). Dla wykrycia składowych o wyższej częstotliwości używane są wąskie okna, pozwalające uzyskać większą rozdzielczość w dziedzinie czasu. Dla wykrycia składowych o niższych częstotliwościach używane są szerokie okna, pozwalające uzyskać większą rozdzielczość w dziedzinie częstotliwości.




Rys. 5.7. Schematy dekompozycji czasowo-częstotliwościowej sygnałów:
a) krótkookresowa transformacja Fouriera, b) transformacja falkowa

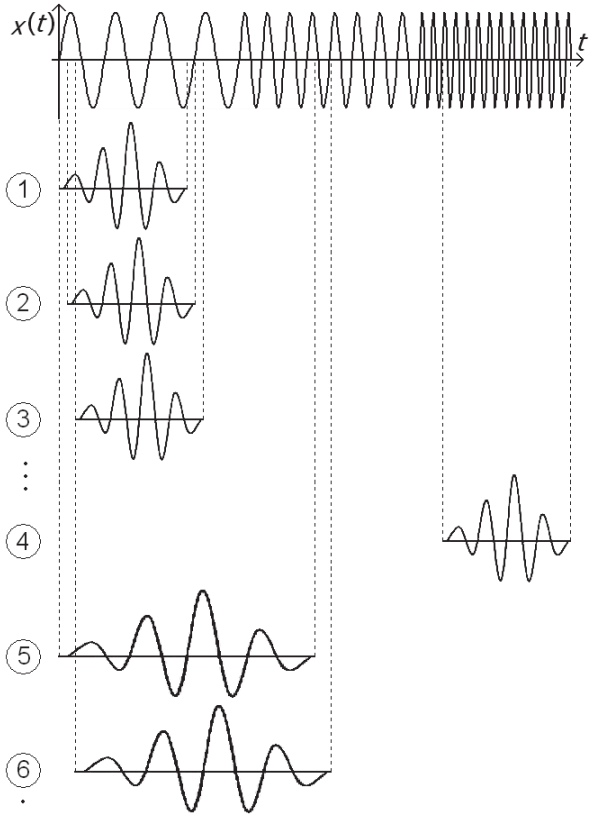
Ciągła transformata falkowa funkcji $x(t)$ określana jest wzorami:

$$W(\tau,\sigma) = \int_{-\infty}^{+\infty} x(t)\cdot\overline{\psi_{\tau,\sigma}(t)}\,dt$$
$$\psi_{\tau,\sigma}(t) = \frac{1}{\sqrt{\sigma}}\psi\left(\frac{t-\tau}{\sigma}\right) \tag{5.2}$$

Funkcja $\psi(t)$ jest falką o skończonym czasie oscylacji i wartości średniej równej zero. Przez zmianę parametru $\sigma$ dokonuje się zmiany skali (częstotliwości), a przez zmianę parametru $\tau$ przesunięcia, w rezultacie czego otrzymujemy z falki podstawowej rodzinę falek. Dzięki czynnikowi $1/\sqrt{\sigma}$ wszystkie falki rodziny mają taką samą energię, czyli pole powierzchni (całka) z wykresu funkcji falkowej jest stałe, dla wszystkich wartości skali (rozciągnięcia).

Ilustracja procesu dekompozycji falkowej funkcji $x(t)$ przedstawiona jest na rysunku 5.8. Rozpoczynamy od ustawienia wybranej falki na początku fragmentu analizowanego sygnału (1) i obliczamy współczynniki falkowe odpowiadające korelacji (podobieństwu) między falką o danej skali i pozycji, a odpowiednim segmentem sygnału $x(t)$. Następnie przesuwamy falkę o jeden cykl w prawo (zmiana $\tau$) (2) i powtarzamy obliczenia dla nowego segmentu sygnału. Procedurę powtarzamy dla kolejnych segmentów (3), aż do osiągnię-

cia końca analizowanego sygnału (4). Następnie przeskalowujemy falkę przez zmianę $\sigma$ (rozciągamy falkę) i powracamy na początek sygnału (5). Obliczamy współczynniki falkowe, ponownie przesuwamy falkę (6) i powtarzamy obliczenia. Przedstawione czynności powtarzane są do wyczerpania wszystkich skal.




Rys. 5.8. Ilustracja wykonywania dekompozycji falkowej

Mała wartość skali $\sigma$ pozwala na zidentyfikowanie szybkozmiennych detali występujących w analizowanym sygnale (składowe o wysokich częstotliwościach). Zwiększenie skali powoduje rozciągnięcie falki i w związku z tym falka porównywana jest z dłuższym segmentem sygnału, pozwalając na zidentyfikowanie składowych wolnozmiennych (o niższych częstotliwościach).

Wyniki analizy falkowej mogą zostać przedstawione na wykresie w funkcji czasu (oś pozioma) i skali (oś pionowa), natomiast intensywność koloru oznacza amplitudę współczynników. Inaczej niż w analizie STFT wynik zostaje uzyskany we współrzędnych czas-skala ($t/s$), a nie współrzędnych czas-częstotliwość ($t/f$). Jednak nie ma problemu, by dla konkretnego przypadku przeliczyć skalę na częstotliwość.

Dla ciągłej transformaty falkowej CWT (*Continuous Wavelet Transform*) ciągła zmiana parametrów $\sigma$ i $\tau$ powodowałaby bardzo dużą, nadmiarową liczbę współczynników falkowych, a w związku z tym duży czas obliczeń i wymagany duży obszar pamięci. Dlatego w praktyce ograniczamy się do mniejszej liczby skal i przesunięć.

Dyskretna transformata falkowa DWT (*Discrete Wavelet Transform*) bazuje na diadycznym podziale skali oraz przesunięcia:

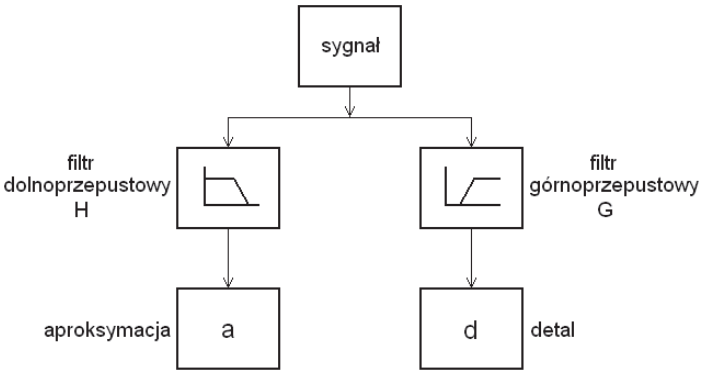
$$\sigma = 2^{-s}, \qquad \tau = 2^{-s} l \tag{5.3}$$

gdzie $l$ określa przesunięcie, a $s$ współczynnik skali ($l = 0, 1, 2,\ldots$; $s = 0, 1, 2,\ldots$).

Dyskretna transformata falkowa ma postać:

$$W(l,s) = \sum_n x(n) \cdot \psi\left(2^s n - l\right) \tag{5.4}$$

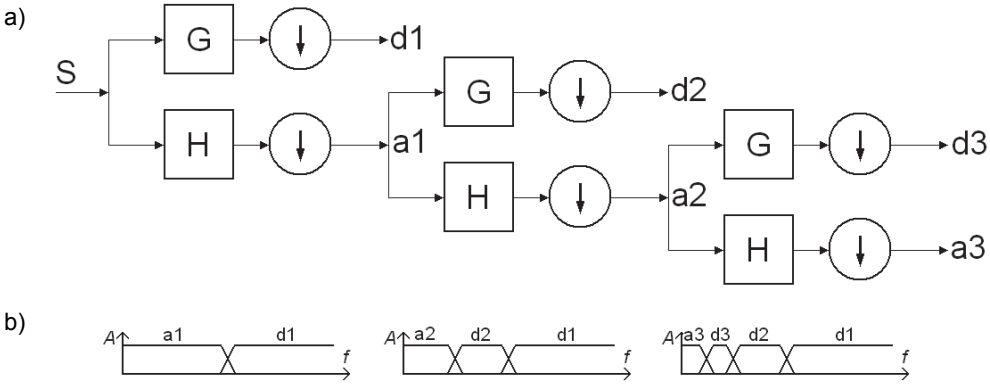
W praktyce analiza falkowa jest wykonywana za pomocą zespołu filtrów (rys. 5.9).




Rys. 5.9. Filtracyjna dekompozycja falkowa

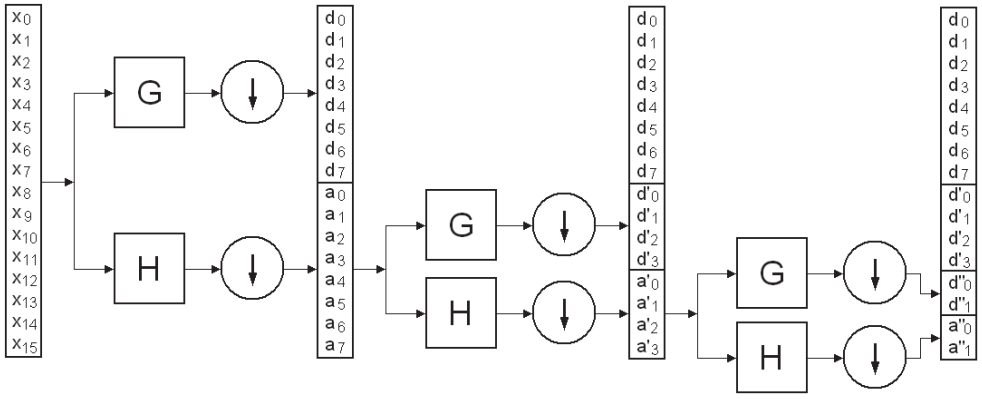
Na wyjściu filtra dolnoprzepustowego H otrzymuje się składowe wolnozmienne sygnału (tzw. aproksymacja), natomiast na wyjściu filtra górnoprzepustowego składowe odzwierciedlające szybkie zmiany w sygnale (tzw. detal).

Otrzymana w wyniku przedstawionego procesu aproksymacja może ponownie podlegać rozdzieleniu na dwie składowe. W ten sposób otrzymujemy tzw. drzewo dekompozycji falkowej (rys. 5.10).




Rys. 5.10. Drzewo dekompozycji falkowej dla 3-poziomowej analizy (a) i odpowiadające jej zakresy częstotliwości (b)

Na wyjściu filtrów, obie składowe: aproksymacji i detali, zawierają taką samą liczbę elementów, co sygnał wejściowy. Zatem sumaryczna liczba elementów jest dwukrotnie większa niż liczba elementów ciągu wejściowego. Dlatego na wyjściu filtrów realizowana jest operacja decymacji, polegająca na odrzuceniu co drugiego elementu w obu ciągach współczynników falkowych (na rysunku 5.10 oznaczona symbolem ↓). W rezultacie sumaryczna liczba elementów aproksymacji i detalu jest równa liczbie elementów wejściowych (rys. 5.11).

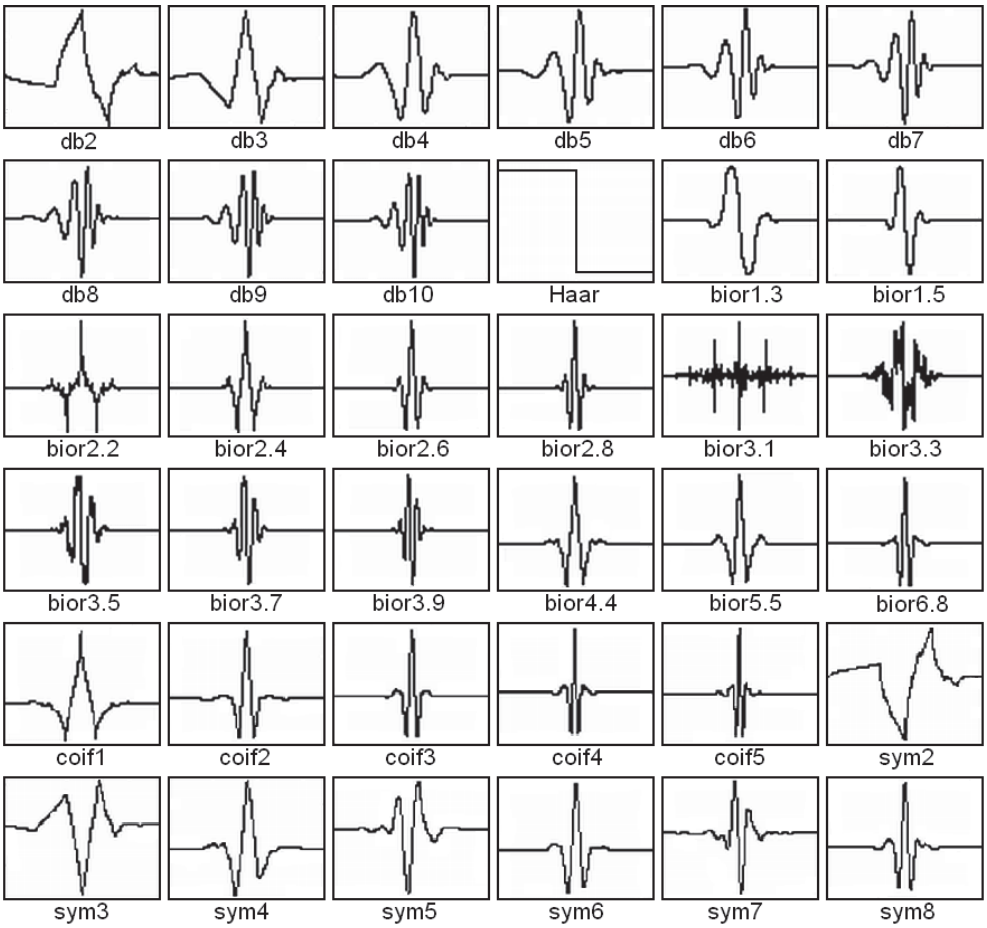



Rys. 5.11. Ilustracja procesu decymacji współczynników falkowych w kolejnych krokach dekompozycji falkowej sygnału

Proces dekompozycji falkowej może zawierać szereg kroków, ale musi zostać zakończony, gdy w kolejnym kroku detal będzie zawierał tylko jedną próbkę. Dlatego liczba kroków jest liczbą naturalną nie większą niż $\log_2 n$, gdzie $n$ jest liczbą próbek sygnału wejściowego.

Znanych jest wiele falek o różnym kształcie. Do najczęściej stosowanych należą falki Haara, Daubechies (skrócona nazwa **db**), Coiflets (skrócona nazwa **coif**), Symlets (skrócona nazwa **sym**), biortogonalne (skrócona nazwa **bior**). Kształt wybranych falek pokazany jest na rysunku 5.12.

Falka o kształcie funkcji Haara ma prosty kształt przypominający funkcję skoku, ale jej praktyczne zastosowanie jest ograniczone. Jednymi z częściej stosowanych są falki Daubechies. Numer w symbolu tych falek oznacza rząd. Falka Daubechies rzędu 1 jest taka sama, jak falka Haara. Falki biortogonalne charakteryzują się liniową fazą. Falki biortogonalne wykorzystywane do syntezy (składania sygnału) są ortogonalne względem pokazanych na rysunku falek wykorzystywanych do dekompozycji.



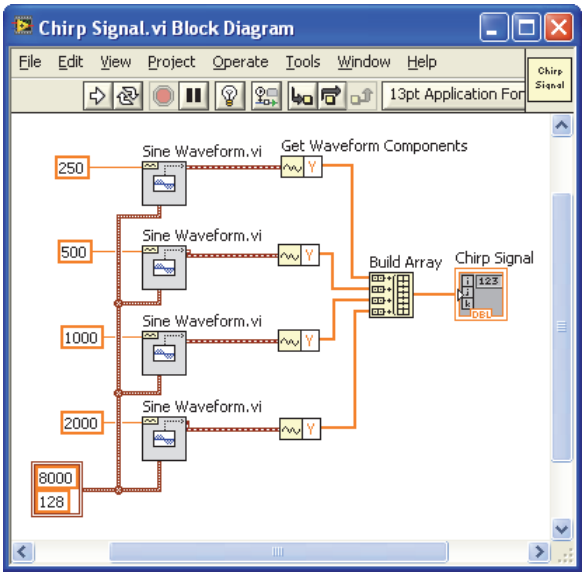

Rys. 5.12. Kształty przykładowych falek

## 5.4. Realizacja analizy falkowej

Analiza falkowa zostanie przedstawiona na przykładzie analizy sygnału, który skokowo zmienia swoją częstotliwość. Diagram podprogramu do generowania sygnału przedstawiony jest na rysunku 5.13.

W programie czterokrotnie wykorzystano funkcję *Sine Waveform*, generującą 128 próbek przebiegu sinusoidalnego, o częstotliwościach kolejno 250, 500, 1000 i 2000 Hz. Sygnały wyjściowe typu *waveform* za pomocą funkcji *Get Waveform Components* zostają przekonwertowane na tablice jednowymiarowe DBL, a następnie połączone w jeden ciąg próbek za pomocą funkcji *Build Array* (po ustawieniu kursora na tej funkcji i naciśnięciu prawego przycisku myszy, należy zaznaczyć pozycję *Concatenate Inputs*). W rezultacie działania tego podprogramu otrzymujemy przebieg zawierający $4 \cdot 128 = 512$ próbek.

Po zdefiniowaniu zacisku wyjściowego i przygotowaniu ikony podprogram należy zapisać na dysku.



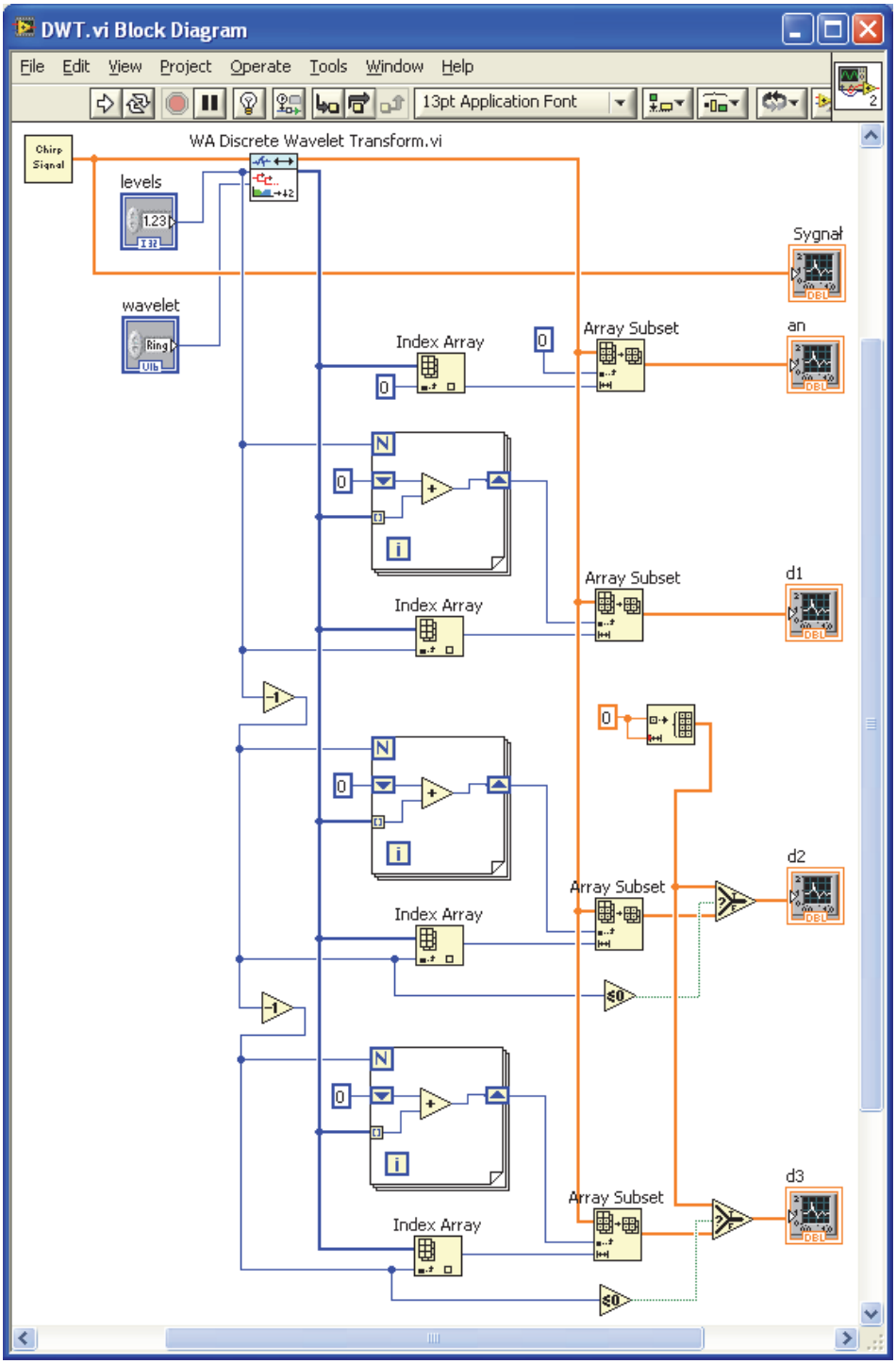

Rys. 5.13. Diagram podprogramu generującego sygnał do analizy falkowej

Podprogram jest wykorzystany w programie demonstrującym działanie analizy falkowej. Diagram programu pokazany jest na rysunku 5.14, a panel na rysunku 5.15.

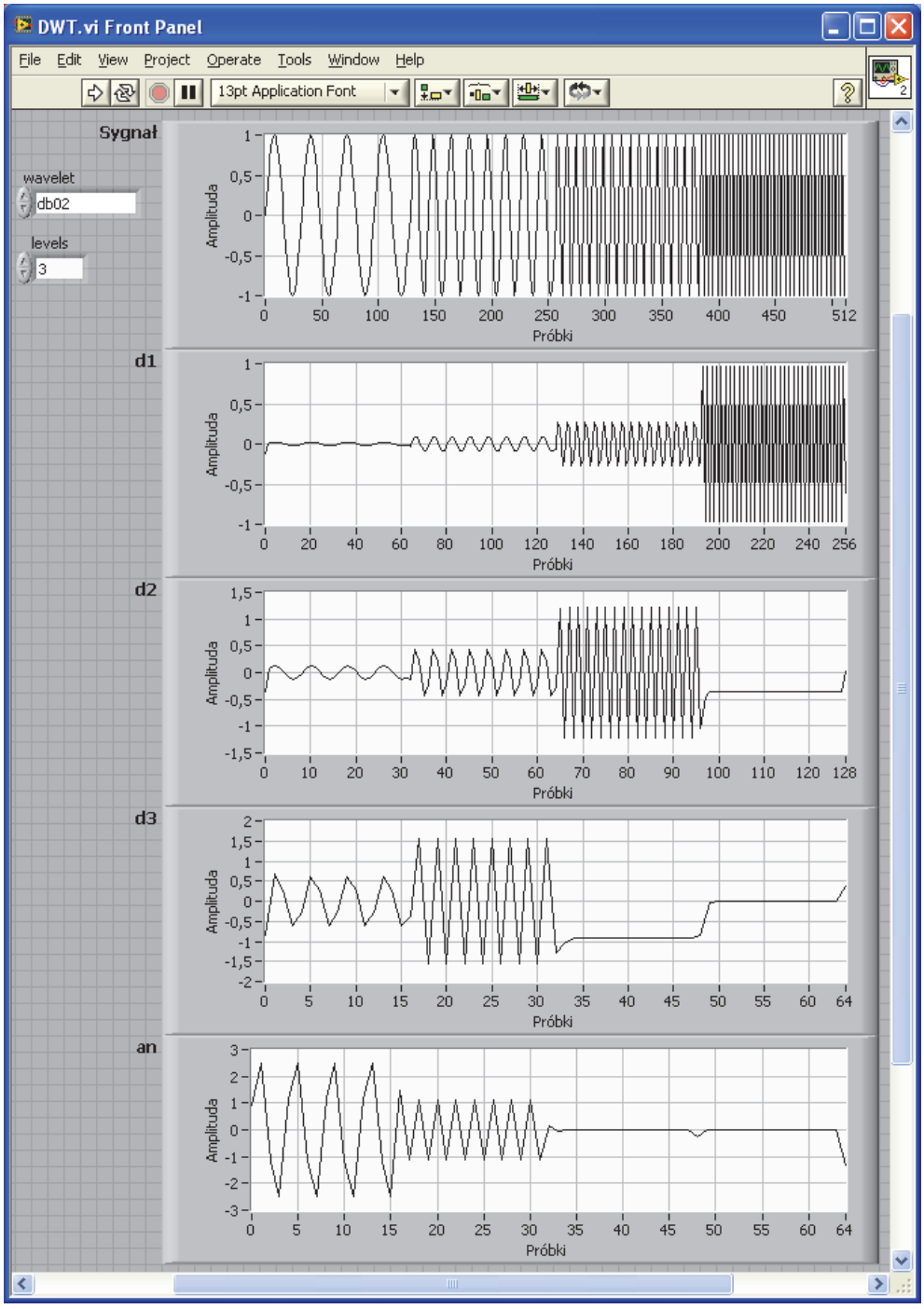
W programie wykorzystano funkcję *WA Discrete Wavelet Transform* ze wspomnianej już biblioteki *Advanced Signal Processing Toolkit*. Na wejście *signal* tej funkcji podajemy analizowany sygnał, na wejście *levels* liczbę poziomów analizy falkowej, natomiast zacisk wejściowy *wavelet* pozwala na wybór falki.

Z zacisku wyjściowego *DWT coef* można pobrać tablicę jednowymiarową, zawierającą współczynniki aproksymacji i detali poszczególnych poziomów (od ostatniego do pierwszego poziomu). Położenie poszczególnych danych można określić na podstawie zawartości jednowymiarowej tablicy wyjściowej *length*. Dla *N* poziomów analizy, tablica ta zawiera *N*+2 elementów: liczbę próbek aproksymacji *N*-tego poziomu, liczbę próbek detali od *N*-tego do pierwszego poziomu oraz liczbę próbek sygnału wejściowego.

Przedstawiony program pozwala na zapoznanie się z wynikami dla 1, 2 i 3 poziomów analizy falkowej. Umożliwia również sprawdzenie, jaki wpływ na wyniki analizy ma rodzaj wybranej falki.




Rys. 5.14. Diagram programu do analizy falkowej




Rys. 5.15. Panel programu do analizy falkowej

# Bibliografia


[1] Gonera M., Kiciński W., Opara T.: *Czasowo-częstotliwościowe metody analizy sygnałów*. Prace PIE 2004, nr 149, s. 7–37.

[2] Kehtarnavaz N., Kim N.: *Digital Signal Processing. System-Level Design Using LabVIEW*. Elsevier 2005.

[3] Marven C., Ewers G.: *Zarys cyfrowego przetwarzania sygnałów*. Warszawa: Wyd. Komunikacji i Łączności 1999.

[4] Rak R.J.: *Wirtualny przyrząd pomiarowy, realne narzędzie współczesnej metrologii*. Warszawa: Oficyna Wydawnicza Polit. Warszawskiej 2003.

[5] Świsulski D.: *Komputerowe przetwarzanie sygnałów pomiarowych. Metrologia*. Skrypt do laboratorium dla studentów kierunku elektrotechnika. Gdańsk: Polit. Gdańska, Wydz. Elektrotechniki i Automatyki 2009, s. 117–125.

[6] Świsulski D.: *Komputerowa technika pomiarowa. Oprogramowanie wirtualnych przyrządów pomiarowych w LabVIEW*. Warszawa: Agenda Wydawnicza PAK 2005.

[7] Świsulski D.: *Systemy Pomiarowe. Laboratorium*. Gdańsk: Wyd. Polit. Gdańskiej 2004.

[8] Zieliński T.P.: *Cyfrowe przetwarzanie sygnałów. Od teorii do zastosowań*. Warszawa: Wyd. Komunikacji i Łączności 2005.

[9] *Laboratorium cyfrowego przetwarzania sygnałów*. Polit. Częstochowska, Wydz. Elektryczny, Zakł. Automatyki. http://www.ztmapc.el.pcz.pl/stud/cpslab.html