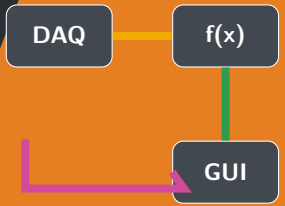
Dane i logika programu

Tablice, Case, pętle, tunele i wykresy



Wprowadzenie do LabVIEW i układów kontrolno-pomiarowych ▪ Wykład 2
dr inż. Angelika Tefelska

Mapa dzisiejszego wykładu



Cel: po tym wykładzie mamy umieć przejść od pojedynczych operacji do programu, który **przetwarza serię danych, podejmuje decyzje i czytelnie prezentuje wyniki pomiarów**.

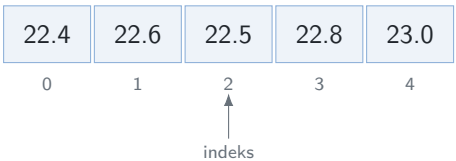
1. Tablice

Jak przechowywać i przetwarzać serię pomiarów?



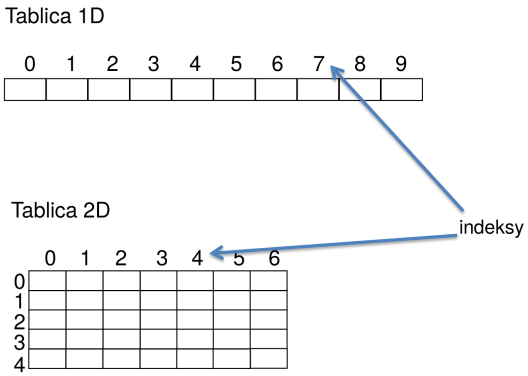
Po co nam tablice?

- jeden pomiar temperatury: 22.4 °C,
- 100 pomiarów temperatury: potrzebujemy **jednej uporządkowanej struktury**,
- tablica pozwala wykonać tę samą operację na wielu elementach,
- indeks wskazuje, który element aktualnie odczytujemy.



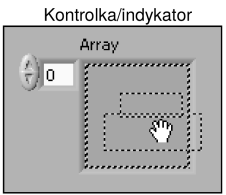
Tablica – najważniejsze zasady

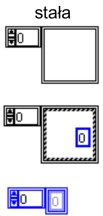
- zawiera elementy **tego samego typu**,
- może mieć jeden lub więcej wymiarów,
- elementy odczytujemy przez **indeks**,
- pierwszy indeks ma wartość **0**,
- w LabVIEW nie istnieje **tablica tablic**; używamy kolejnego wymiaru lub np. tablicy klastrów.

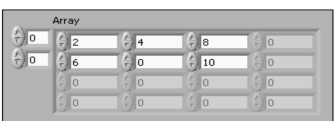


Jak utworzyć tablicę?

- na Front Panel wybieramy kontener **Array**,
- do środka wkładamy typ elementu: Numeric, Boolean, String, Enum itd.,
- typ pierwszego elementu określa typ całej tablicy,
- można tworzyć także stałe tablicowe na Block Diagramie.



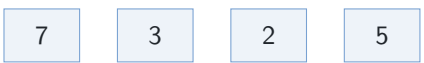




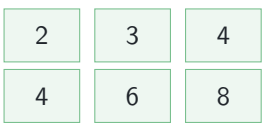
Część elementów w tablicy może zostać niezainicjalizowana. W takim wypadku są one zamalowane na szaro

Tablice 1D i 2D

1D – seria pomiarów

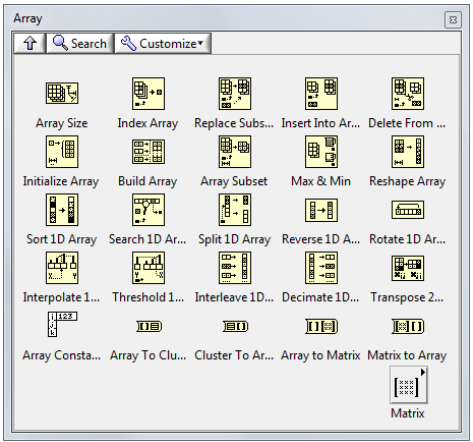


2D – tabela danych



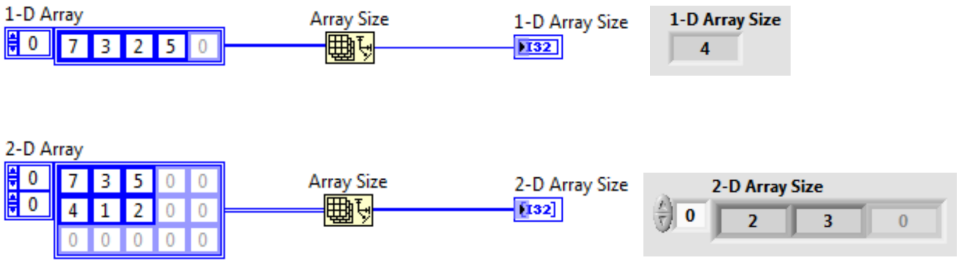
- 1D: przebieg czasowy, lista wyników, wartości z kolejnych kanałów,
- 2D: wiele kanałów $\times$ wiele próbek, tabela wyników eksperymentu.

Paleta Array



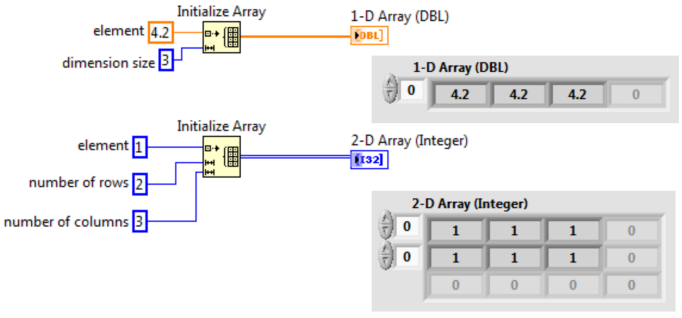
- ważniejsze od zapamiętania ikon jest rozumienie **operacji**,
- najczęściej używane: Array Size, Index Array, Array Subset, Initialize Array, Build Array,
- kolejne funkcje łatwo znaleźć przez Quick Drop lub Search.

Array Size – ile danych mamy?



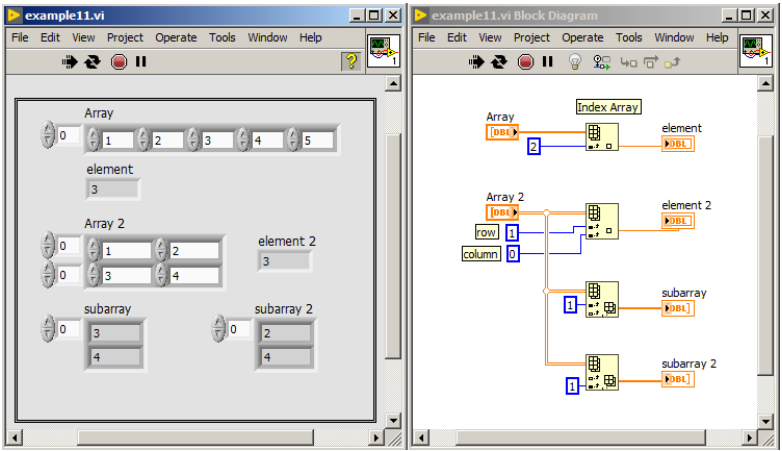
- dla tablicy 1D otrzymujemy jedną liczbę,
- dla tablicy wielowymiarowej wynik jest tablicą rozmiarów poszczególnych wymiarów, gdzie 0-wy element to liczba wierszy a 1-wy element to liczba kolumn.

Initialize Array – tablica o zadanym rozmiarze



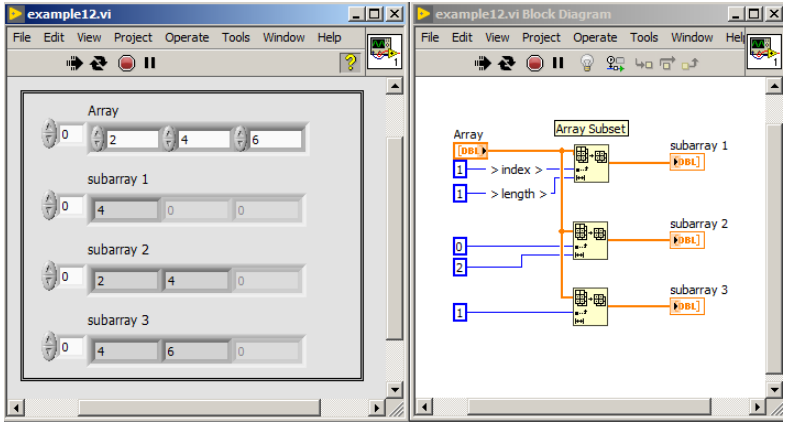
- podajemy wartość elementu,
- podajemy rozmiar wymiaru lub wymiarów,
- powstaje tablica wypełniona wybranym elementem.

Index Array – dostęp do elementu



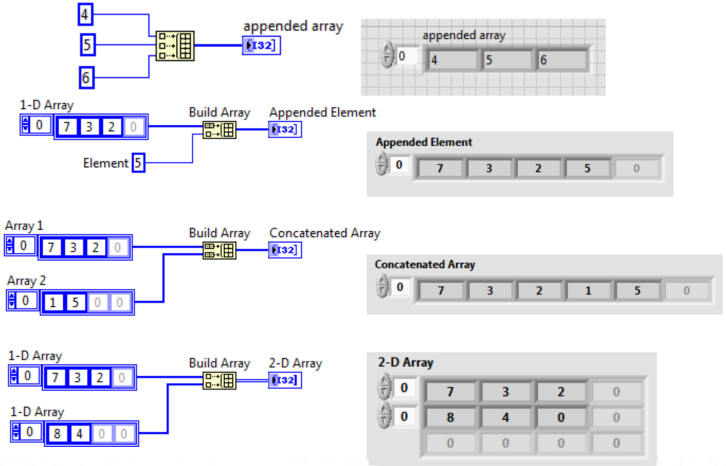
- 1D: jeden indeks,
- 2D: wiersz i kolumna,
- można rozszerzyć węzeł i pobierać kilka kolejnych elementów.

Array Subset – fragment tablicy



- **index** – od którego elementu zaczynamy,
- **length** – ile elementów pobieramy,
- typowe zastosowanie: analiza wybranego okna próbek.

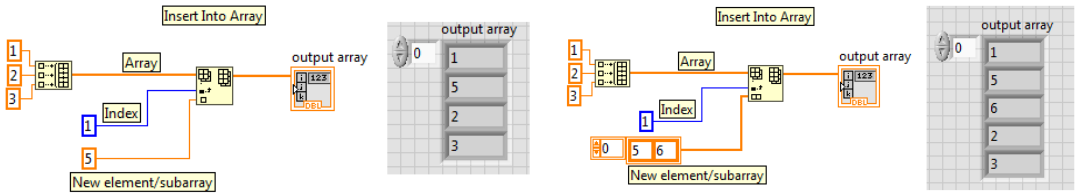
Build Array – składanie danych



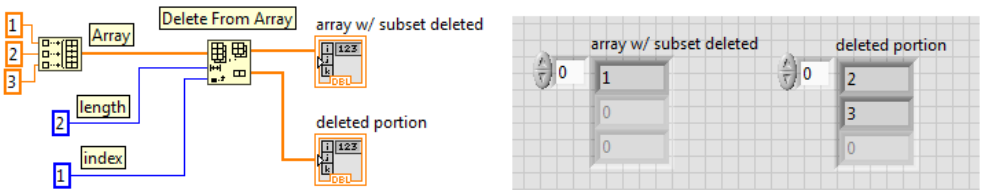
Zwróć uwagę na dwie różne operacje: **dodanie elementu / utworzenie wyższego wymiaru** oraz **konkatenację**. Warto zawsze spojrzeć na typ przewodu na wyjściu.

Insert Into Array i Delete From Array

Wstawienie elementu lub fragmentu:

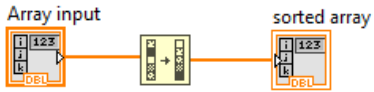


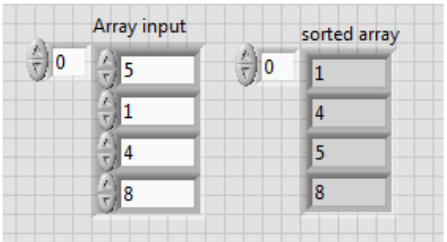
Usunięcie wybranych elementów i zwrot usuniętego fragmentu:

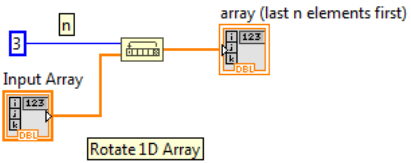


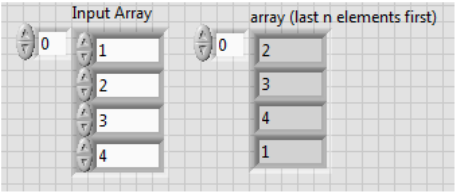
Program1.vi

Sort i Rotate – przestawianie elementów



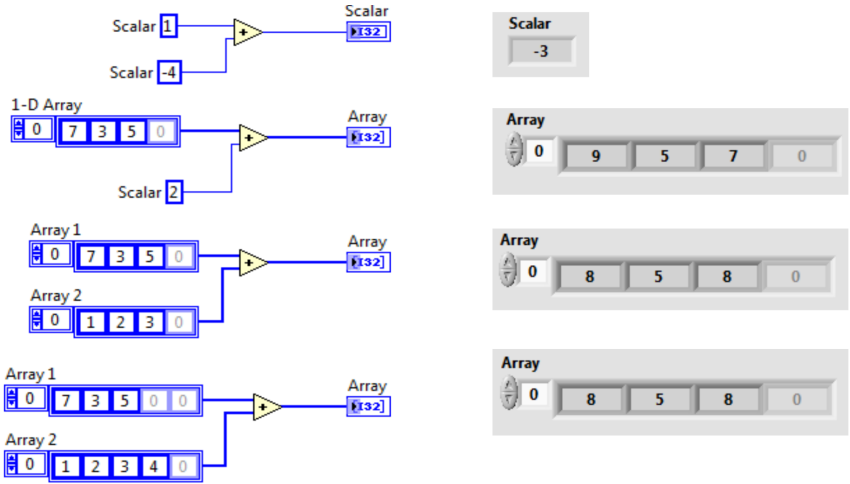






Program2.vi

Polimorfizm – ta sama funkcja, różne dane



Wiele funkcji LabVIEW dopasowuje działanie do typu wejścia: skalar, tablica 1D lub tablica 2D. Nie oznacza to jednak, że każda kombinacja typów będzie poprawna.

Najczęstsze pułapki z tablicami

- indeksowanie zaczyna się od **0**,
- szare pola na Front Panelu nie oznaczają kolejnych realnych elementów tablicy,
- Build Array może zmienić **wymiar** wyniku, jeśli nie ustawimy Concatenate Inputs,
- funkcja **Index Array** dla indeksu spoza zakresu zwraca **wartość domyślną typu**: dla Numeric jest to **0**, dla Boolean – **FALSE**, a dla String – pusty łańcuch,
- jeśli z góry znamy rozmiar dużej tablicy, zamiast rozbudowywać ją element po elemencie warto utworzyć ją funkcją **Initialize Array**, a następnie uzupełniać wartości.

Uwaga: **-1 nie jest** domyślną wartością zwracaną dla tablicy numerycznej.

Czas na podsumowanie tej części wykładu!

Zamiast tekstu – seria pytań kontrolnych.

SPRAWDŹMY, CO JUŻ UMIEMY



Pytanie kontrolne

Tablica 2D ma 4 wiersze i 7 kolumn. Co zwróci Array Size?

(a) liczbę 28
(b) tablicę {4,7}
(c) tablicę {7,4}
(d) dwie osobne liczby 4 i 7

tablice

Pytanie kontrolne

Tablica 2D ma 4 wiersze i 7 kolumn. Co zwróci Array Size?

(a) liczbę 28
(b) tablicę {7,4}
(c) **tablicę {4,7}**
(d) dwie osobne liczby 4 i 7

tablice

Pytanie kontrolne

Który indeks wskazuje pierwszy element tablicy 1D w LabVIEW?

(a) 0
(b) 1
(c) -1
(d) zależy od typu danych

tablice

Pytanie kontrolne

Który indeks wskazuje pierwszy element tablicy 1D w LabVIEW?

(a) **0**
(b) 1
(c) -1
(d) zależy od typu danych

tablice

Pytanie kontrolne

Chcesz pobrać 5 kolejnych próbek, zaczynając od elementu o indeksie 20. Która funkcja pasuje najlepiej?

(a) Array Size
(b) Index Array
(c) Array Subset
(d) Initialize Array

tablice

Pytanie kontrolne

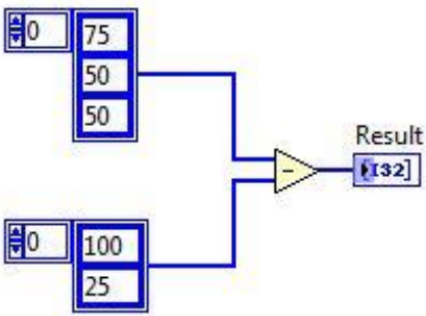
Chcesz pobrać 5 kolejnych próbek, zaczynając od elementu o indeksie 20. Która funkcja pasuje najlepiej?

(a) Array Size
(b) Index Array
(c) **Array Subset**
(d) Initialize Array

tablice

Podsumowanie

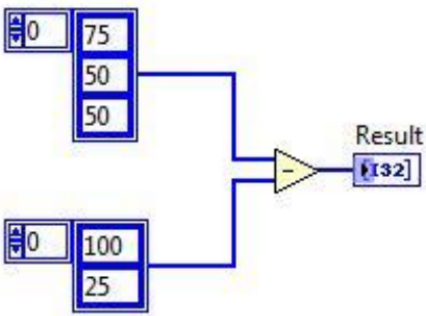
Jaka wartość zostanie wyświetlona w zmiennej Result?



a) tablica jednowymiarowa {-25, 25, 50}
b) tablica jednowymiarowa: {-25,25}
c) tablica dwuwymiarowa: {{-25,-50,-50}, {50,25,25}}
d) tablica jednowymiarowa: {75, -50, 25}

Podsumowanie

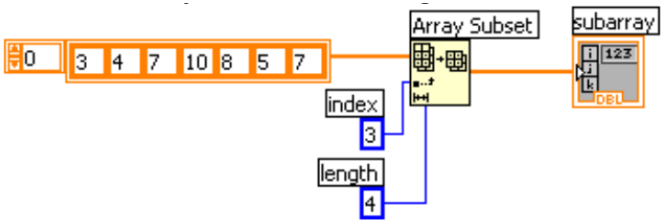
Jaka wartość zostanie wyświetlona w zmiennej Result?



a) tablica jednowymiarowa {-25, 25, 50}
b) **tablica jednowymiarowa: {-25,25}**
c) tablica dwuwymiarowa: {{-25,-50,-50}, {50,25,25}}
d) tablica jednowymiarowa: {75, -50, 25}

Podsumowanie

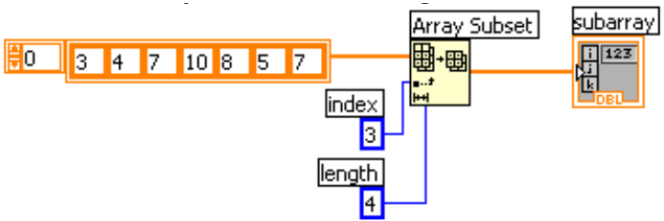
Jaki wynik będzie wyświetlony w zmiennej subarray?



a) tablica jednowymiarowa: {8,5,7}
b) tablica jednowymiarowa: {10,8,5}
c) tablica jednowymiarowa: {10,8,5,7}
d) tablica jednowymiarowa: {7,10,8,5}

Podsumowanie

Jaki wynik będzie wyświetlony w zmiennej subarray?



a) tablica jednowymiarowa: {8,5,7}
b) tablica jednowymiarowa: {10,8,5}
c) **tablica jednowymiarowa: {10,8,5,7}**
d) tablica jednowymiarowa: {7,10,8,5}

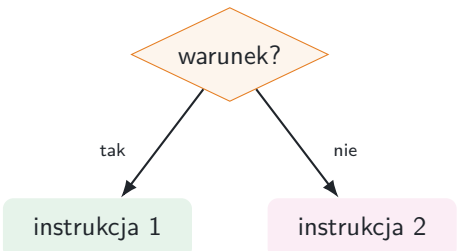
2. Case i Select

Program ma nie tylko liczyć – ma też podejmować decyzje

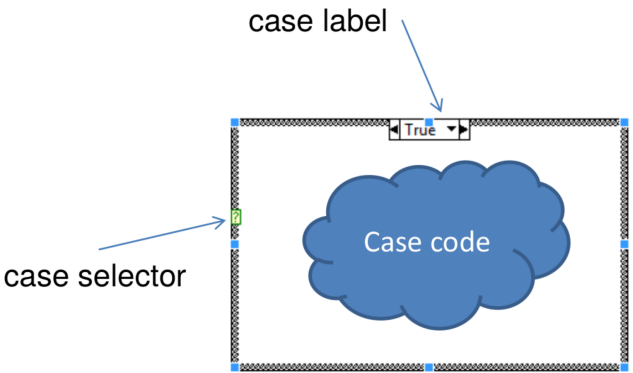


Sterowanie przepływem w języku dataflow

- dataflow mówi **kiedy** węzeł może się wykonać,
- struktury programistyczne mówią **który fragment** ma się wykonać,
- najprostsza decyzja: Case Structure.



Anatomia Case Structure



- **Case selector**: wartość wybierająca przypadek,
- **Case label**: nazwa lub zakres aktywnej ramki,
- wykonywana jest tylko **jedna** ramka.

Case selector i Case labels

Do selektora Case pasują m.in.:

- Boolean, integer, enum, string,
- error cluster – jako specjalny przypadek.

Dlaczego uważać na DBL?

Liczby zmiennoprzecinkowe nie zawsze mają dokładną reprezentację binarną. Dlatego dwa wyniki, które na ekranie wyglądają identycznie, mogą różnić się w ostatnich bitach, np.

$$0.1 + 0.2 \neq 0.3$$

przy porównaniu dokładnej równości.

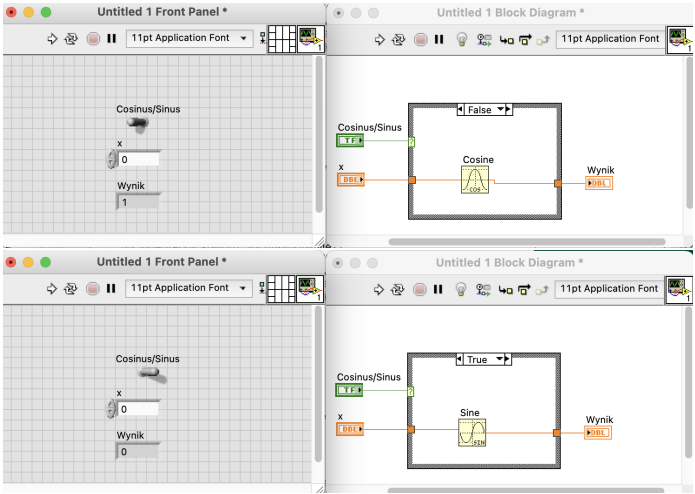
Case label może opisywać:

- pojedynczą wartość: `5`,
- listę: `5, 6, 7`,
- zakres: `7..10`,
- String: `Red`, `Green`,
- przypadek `Default`.

Po podłączeniu **DBL** do selektora, LabVIEW wykonuje konwersję do typu całkowitego (pojawia się coercion dot).

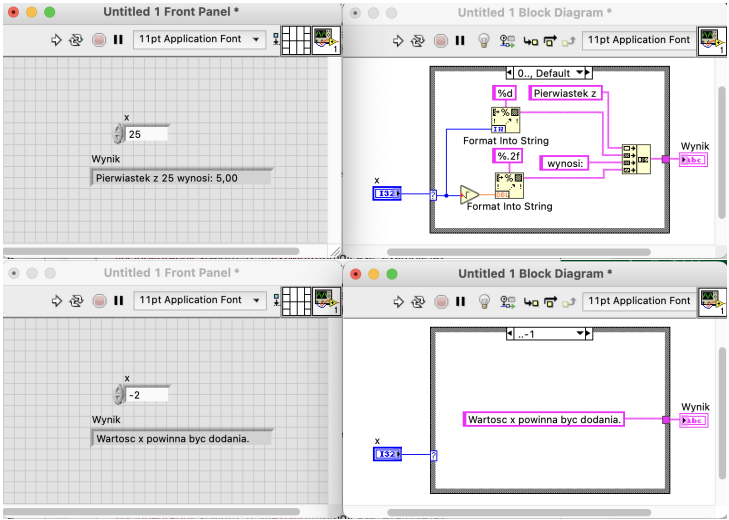
Dobra praktyka: logikę Case opieraj na Boolean, Integer lub Enum, a nie na dokładnym porównaniu wartości DBL.

Przykład: sin czy cos?



Boolean wybiera jedną z dwóch operacji. Zwróć uwagę, że tylko kod w aktywnym przypadku jest wykonywany.

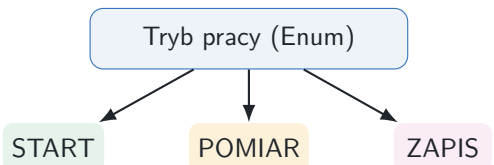
Przykład 2



Program3.vi

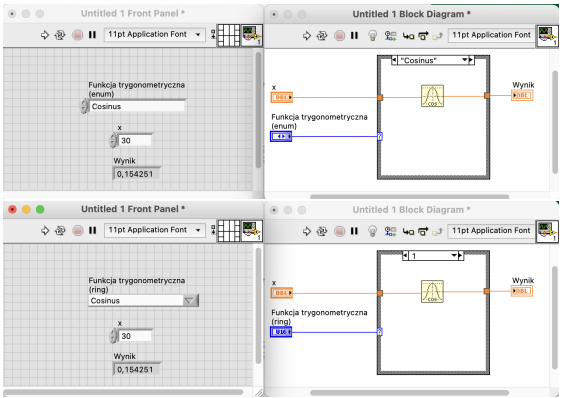
Dlaczego enum pasuje do Case?

- nazwa ramki odpowiada nazwie wartości enuma,
- kod czytamy jako np. *START / POMIAR / ZAPIS*, a nie *0 / 1 / 2*,
- połączenie **Enum + Case** jest fundamentem prostej maszyny stanów – wrócimy do tego na kolejnych wykładach.



Przykład 3

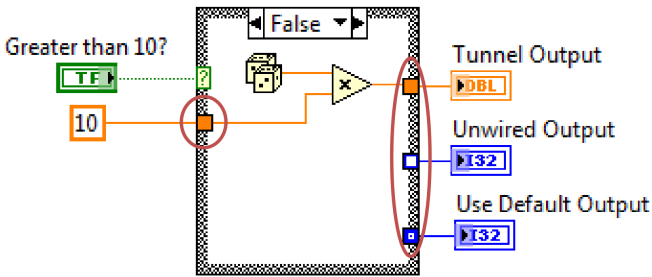
- Podłączenie typu *enum* do *case selector* jest wygodne gdyż w case label automatycznie będziemy mieli te nazwy opcji, co widzi użytkownik.
- Natomiast w przypadku *ring*, będziemy operowali na wartościach liczbowych przypisanych do danych opcji.



Program1_1.vi & program2.vi

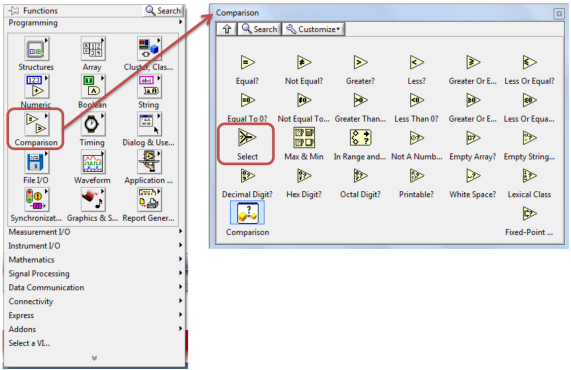
Tunele w Case Structure

Tunele to wejścia/wyjścia drutów ze struktur



- tunel przenosi dane przez granicę struktury,
- klasycznie każde wyjście powinno mieć wartość w każdym przypadku,
- opcja *Use Default if Unwired* spowoduje zwrócenie wartości domyślnej gdy nic nie podłączymy, wygodna ale może ukryć błąd logiczny.

Select – decyzja bez dużej struktury



- wybiera jedną z dwóch **wartości**,
- warunek Boolean trafia do terminala ?,
- przydatne dla prostych wyborów, gdy nie potrzebujemy dwóch osobnych fragmentów kodu.

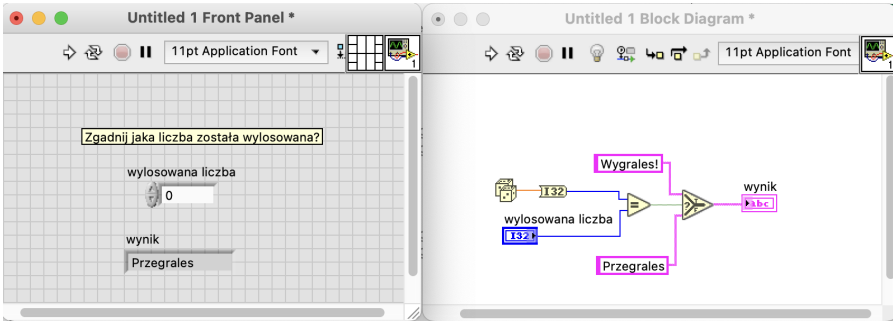
Select a Case – kluczowa różnica

Select

Oba wejściowe fragmenty danych są obliczane, a funkcja wybiera **wynik**.

Case Structure

Wykonuje się tylko kod znajdujący się w **wybranej ramce**.



Pytanie kontrolne

Którego typu nie można podłączyć bezpośrednio do Case selector bez konwersji?

(a) Boolean
(b) Enum
(c) String
(d) Double (DBL)

Case

Pytanie kontrolne

Którego typu nie można podłączyć bezpośrednio do Case selector bez konwersji?

(a) Boolean
(b) Enum
(c) String
(d) **Double (DBL)**

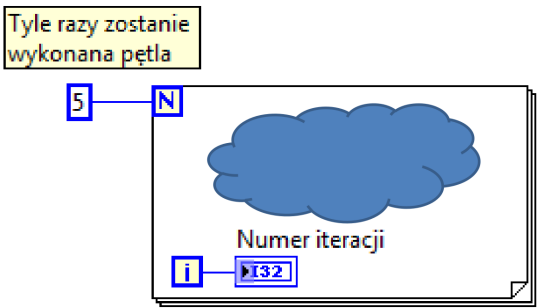
Case

3. Pętle i tunele

Jak wykonać tę samą operację wiele razy?



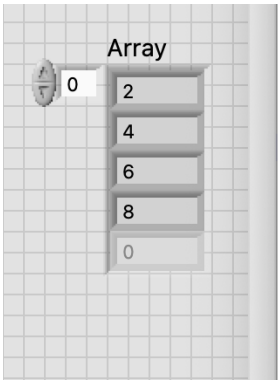
For Loop – znamy liczbę iteracji

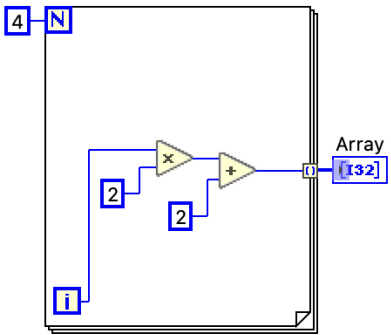


- Pętla *for* powtarza danych fragment kodu *N* razy.
- `N`: maksymalna liczba iteracji,
- `i`: numer bieżącej iteracji,
- `i` zaczyna się od **0**,
- dla `N=5`: `i=0,1,2,3,4`.

Pętle iterują się od 0!

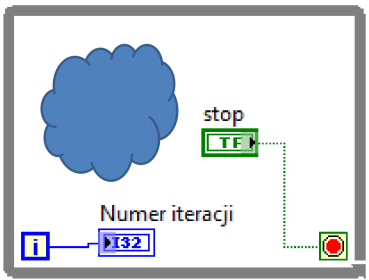
For Loop – przykład





Program4.vi

While Loop – kończymy po spełnieniu warunku

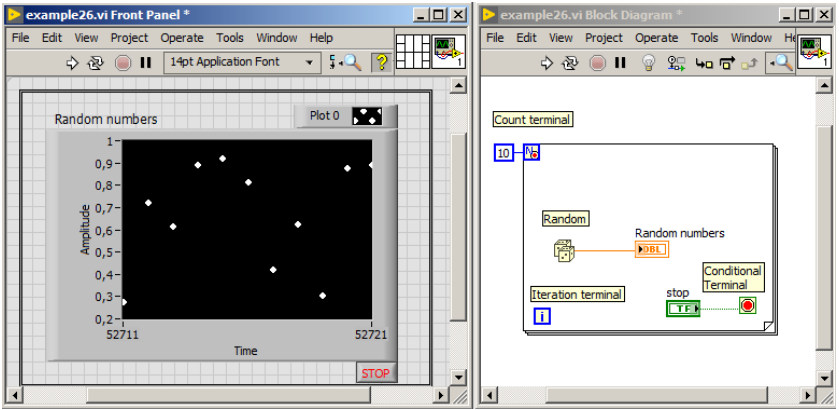



Stop if TRUE


Continue if TRUE

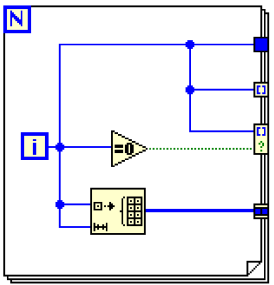
- wykonuje się co najmniej raz,
- kończy się na podstawie terminala warunkowego,
- tryb: *Stop if True* lub *Continue if True*.

For Loop też może zakończyć się wcześniej



- `N` podaje limit,
- terminal warunkowy może zakończyć pętlę przed osiągnięciem `N`,
- przydatne gdy np.: szukamy pierwszej próbki spełniającej warunek.

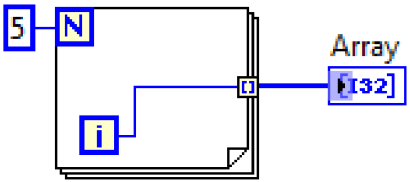
Tunele pętli – co przechodzi przez granicę?

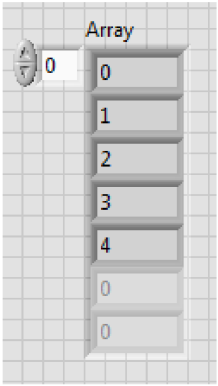


W pętli for domyślnie tunele wstawiane są w trybie indexing a w pętli while w trybie last value

- Last value - zwraca ostatnią wartość.
- Indexing - gromadzi wszystkie wartości podczas działania pętli i po jej zakończeniu zwraca je w formie tablicy.
- Conditional - jak indexing z tą różnicą, że można podać warunek, które dane gromadzić a które nie.
- Concatenating - dodaje do siebie tablice i zwraca jedną długą tablicę.

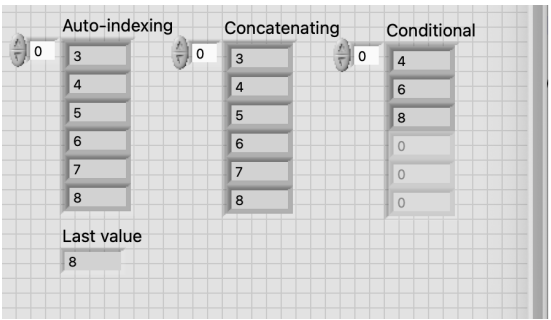
Auto-indexing na wyjściu – pętla buduje tablicę

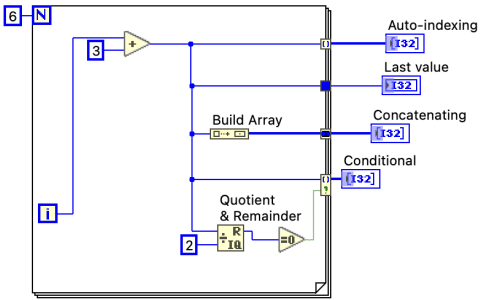




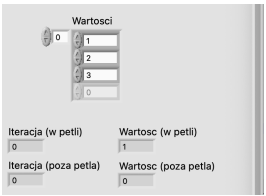
- każda iteracja produkuje jeden element,
- po zakończeniu pętli powstaje tablica,
- stanowi to wygodny sposób na stworzenie tablicy wypełnionej różnymi elementami.

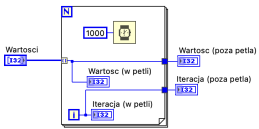
Tunele pętli w praktyce

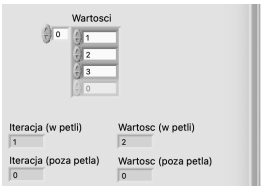


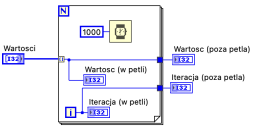


Auto-indexing na wejściu – pętla bierze element po elemencie

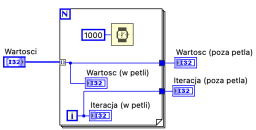


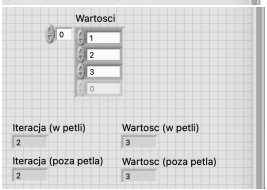


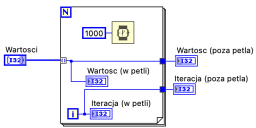






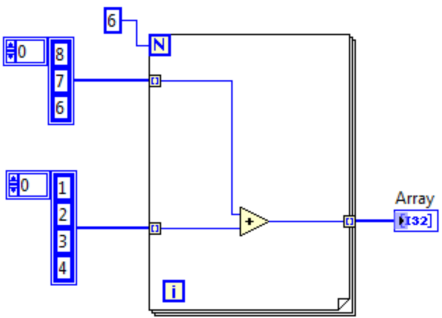


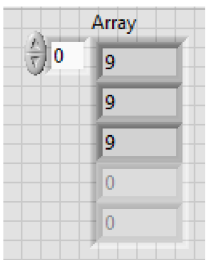




- do wnętrza trafia pojedynczy element,
- jeśli `N` nie jest podłączone, *For Loop* wykonuje się tyle razy ile jest elementów w tablicy,
- nie musimy wcześniej wywoływać *Array Size* i znać rozmiaru tablicy.

Kilka tablic na wejściu – kto ustala liczbę iteracji?



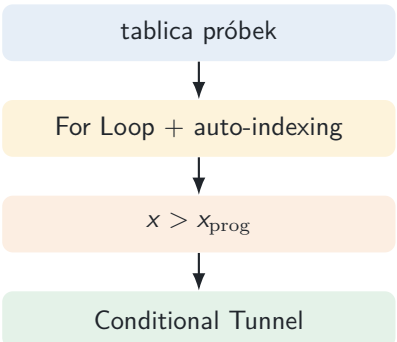


- *For Loop* wykona się tyle razy ile elementów jest w najkrótszej tablicy,
- jeśli podłączymy też `N`, obowiązuje najmniejsze ograniczenie,
- wygodny sposób odczytywania elementu po elemencie z tablicy.

For czy While?

	For Loop	While Loop
Kiedy używać?	znamy liczbę elementów / iteracji	nie znamy z góry liczby iteracji
Czy może wykonać się 0 razy?	tak	nie
Domyślny tunel	auto-indexing	last value / indexing wyłączony
Typowy przykład	obróbka całej tablicy	aplikacja działająca do naciśnięcia STOP

Przykład: Jak filtrować serię pomiarów



- pętla dostaje próbki pojedynczo,
- warunek testuje każdą próbkę,
- na wyjściu otrzymujemy krótszą tablicę.

Czas na podsumowanie tej części wykładu!

Zamiast tekstu – seria pytań kontrolnych.

SPRAWDŹMY, CO JUŻ UMIEMY



Pytanie kontrolne

Do For Loop podłączono tablicę zawierającą 8 elementów. Terminal N nie jest podłączony. Ile iteracji wykona pętla?

(a) 7
(b) 8
(c) 9
(d) nie można określić

pętle

Pytanie kontrolne

Do For Loop podłączono tablicę zawierającą 8 elementów. Terminal N nie jest podłączony. Ile iteracji wykona pętla?

(a) 7
(b) **8**
(c) 9
(d) nie można określić

pętle

Pytanie kontrolne

Który tunel pozwala zbudować tablicę zawierającą tylko elementy spełniające warunek?

(a) Last Value
(b) Indexing
(c) Conditional
(d) Concatenating

tunele

Pytanie kontrolne

Który tunel pozwala zbudować tablicę zawierającą tylko elementy spełniające warunek?

(a) Last Value
(b) Indexing
(c) **Conditional**
(d) Concatenating

tunele

Pytanie kontrolne

Która pętla w typowej konfiguracji musi wykonać się co najmniej raz?

(a) For Loop
(b) While Loop
(c) obie
(d) żadna

pętle

Pytanie kontrolne

Która pętla w typowej konfiguracji musi wykonać się co najmniej raz?

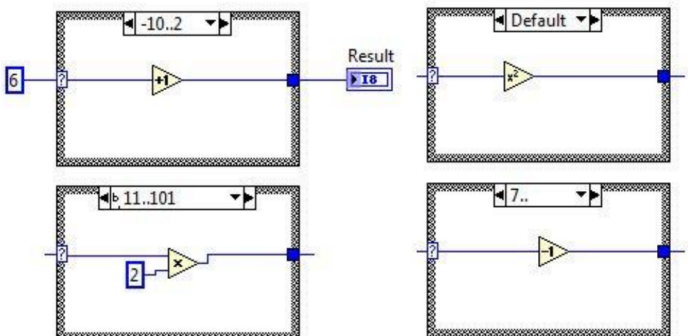
(a) For Loop
(b) **While Loop**
(c) obie
(d) żadna

pętle

Pytanie kontrolne

Poniżej przedstawiono wszystkie stany struktury Case. Jaka wartość zostanie wyświetlona w zmiennej *Result*?

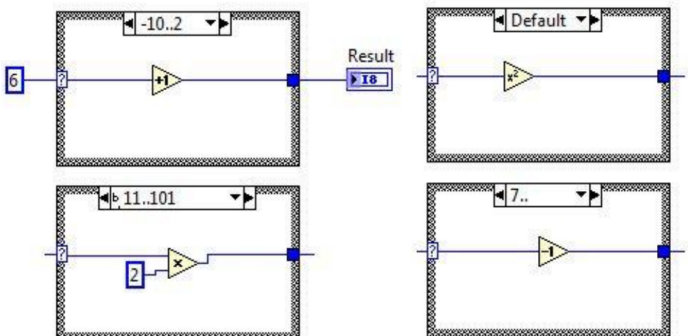
a) 5
b) 7
c) 12
d) 36



Pytanie kontrolne

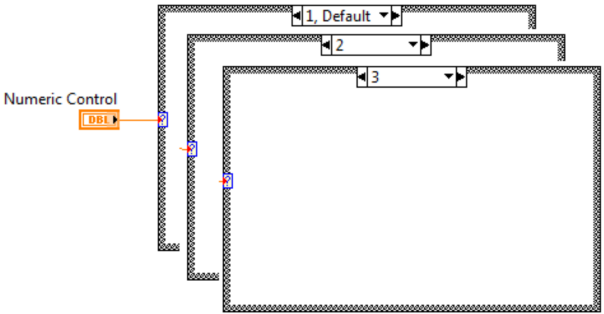
Poniżej przedstawiono wszystkie stany struktury Case. Jaka wartość zostanie wyświetlona w zmiennej *Result*?

a) 5
b) 7
c) 12
d) **36**



Pytanie kontrolne

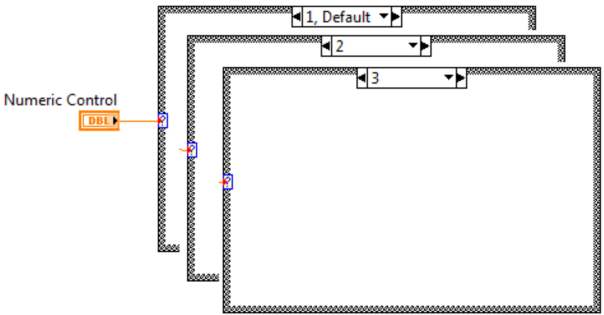
Która ramka struktury case wykona się, jeśli *Numeric Control* przyjmuje wartość 2.5?



(a) 1, Default
(b) 2
(c) 3
(d) żadna, gdyż *Numeric Control* jest typu double

Pytanie kontrolne

Która ramka struktury case wykona się, jeśli *Numeric Control* przyjmuje wartość 2.5?

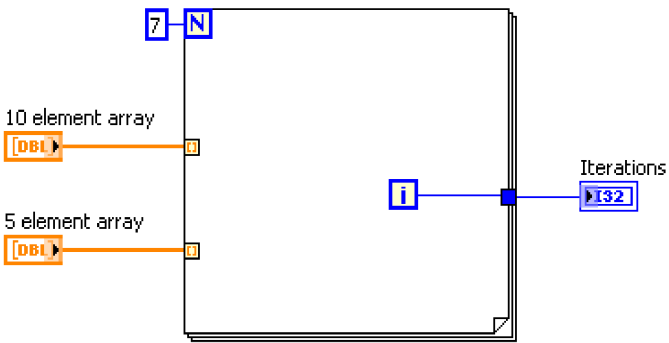


(a) 1, Default
(b) **2** - bo przy konwersji *Numeric Control* z double na int otrzymamy wartość 2 (patrz wykład 1 temat konwersji liczb).
(c) 3
(d) żadna, gdyż *Numeric Control* jest typu double

Pytanie kontrolne

Jaką wartość zwróci *Iterations* po wykonaniu pętli?

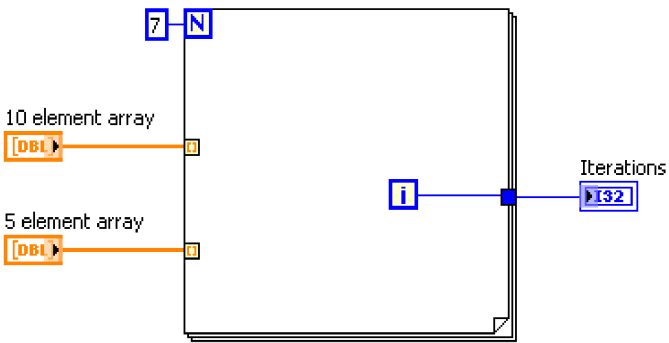
a) 4
b) 6
c) 7
d) 9



Pytanie kontrolne

Jaką wartość zwróci *Iterations* po wykonaniu pętli?

a) **4**
b) 6
c) 7
d) 9

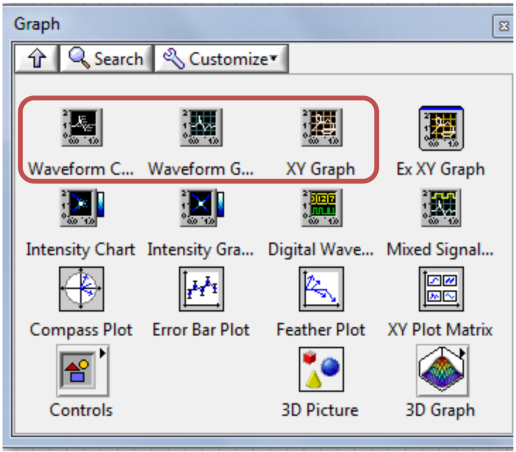


4. Wykresy

Jak dobrać sposób prezentacji danych do rodzaju pomiaru?

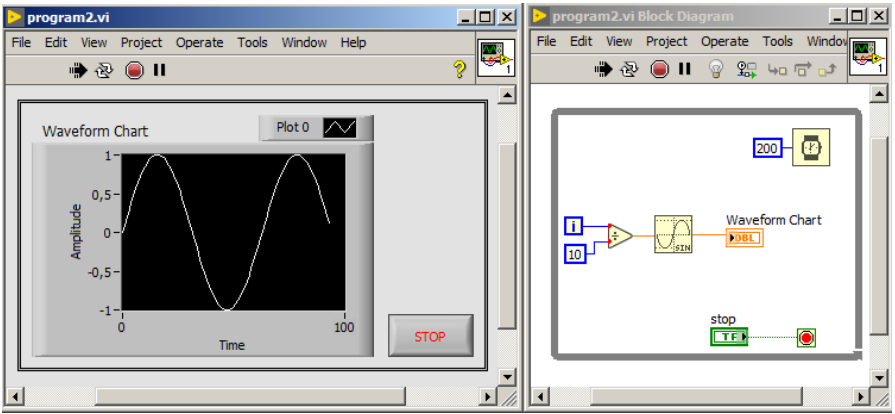


Gdzie znaleźć wykresy?



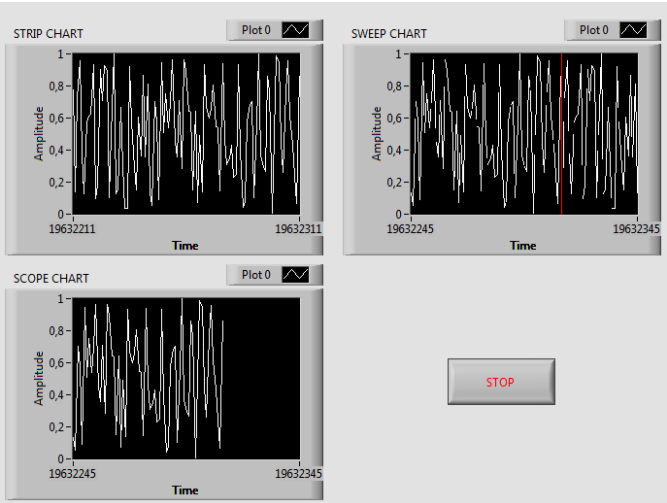
- **Waveform Chart** – przebieg aktualizowany na bieżąco,
- **Waveform Graph** – seria próbek o równych odstępach na osi X,
- **XY Graph** – sami podajemy zarówno X, jak i Y.

Waveform Chart – punkt po punkcie



- każda iteracja może dostarczyć jedną nową próbkę,
- Chart zachowuje poprzednie punkty w historii,
- punkty są rozmieszczone równomiernie na osi X.

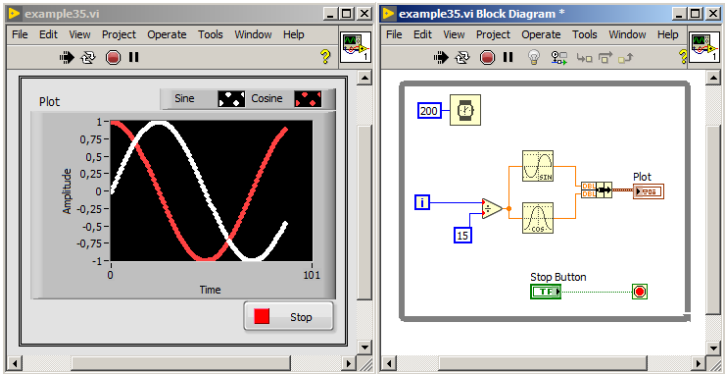
Waveform Chart – trzy tryby wyświetlania



- **Strip Chart**: dane przesuwają się wraz z napływem nowych próbek,
- **Scope Chart**: po zapełnieniu obszaru wykres zaczyna rysowanie od początku,
- **Sweep Chart**: nowa i stara część są rozdzielane pionową linią.

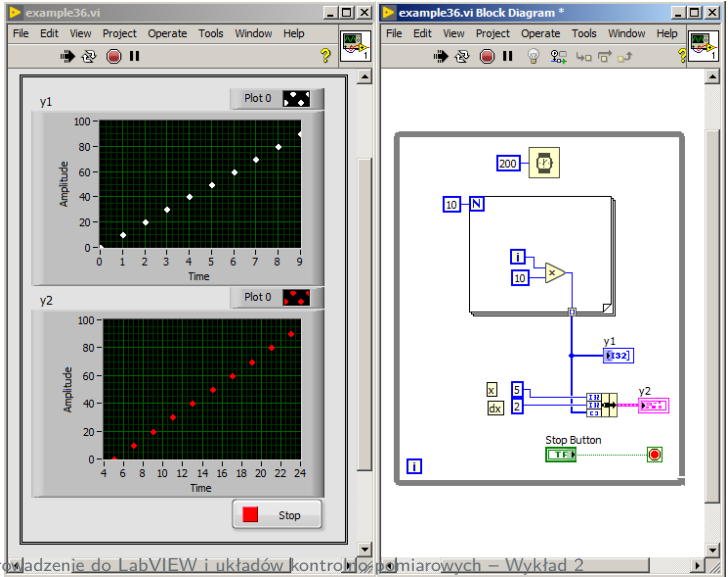
Program5.vi

Kilka przebiegów na Waveform Chart



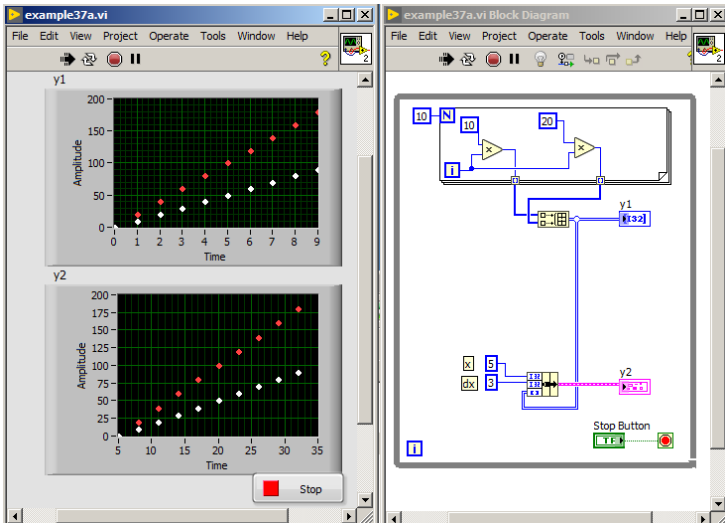
- wiele sygnałów można wyświetlić na jednej osi,
- dane dla kilku przebiegów łączymy w klaster przed podaniem na Chart,
- przydatne np. do porównania dwóch kanałów pomiarowych w czasie.

Waveform Graph – cała seria naraz



- Wyświetla całą serię punktów zamiast dodawać punkt po punkcie,
- na wejście trafia najczęściej tablica wartości Y,
- kolejne elementy tablicy odpowiadają kolejnym punktom osi X,
- istnieje możliwość narysowania wykresu zaczynającego się od wybranej wartości na osi x i o innej Δx niż 1. Wtedy należy użyć klastra.

Kilka serii na Waveform Graph



- dla serii o wspólnej osi X możemy użyć m.in. tablicy 2D,
- w przypadku wykresu zaczynającego się od innej wartości x_0, należy skorzystać z klastra podając jako trzeci element tablicę 2D.

Wyświetlenie kilku serii pomiarowych na Waveform Graph

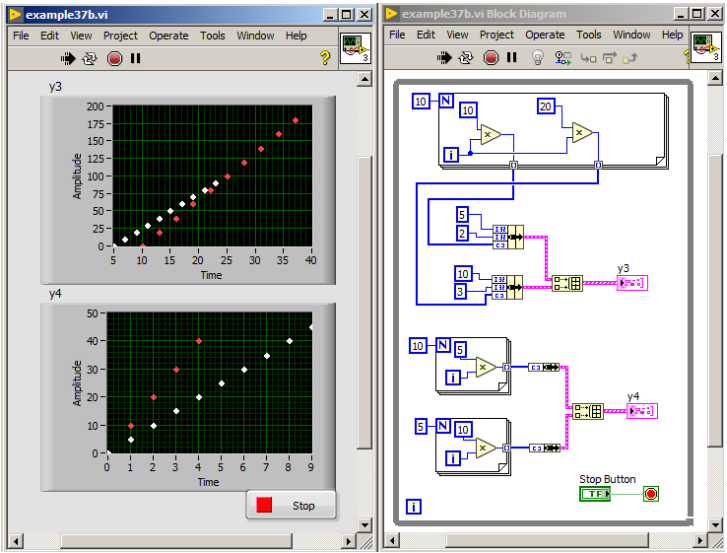


Chart czy Graph? – najważniejsza różnica

Waveform Chart

- dostaje kolejne próbki **punkt po punkcie**,
- sam przechowuje historię,
- naturalny wybór dla pomiaru wykonywanego w pętli.

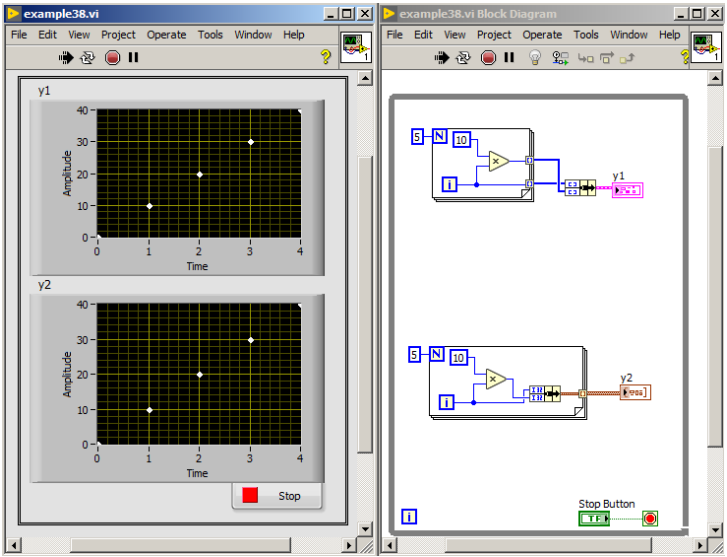
Chart = strumień próbek

Waveform Graph

- dostaje **całą serię** danych jednocześnie,
- typowo podłączamy tablicę próbek,
- dobry do prezentacji gotowego wyniku pomiaru.

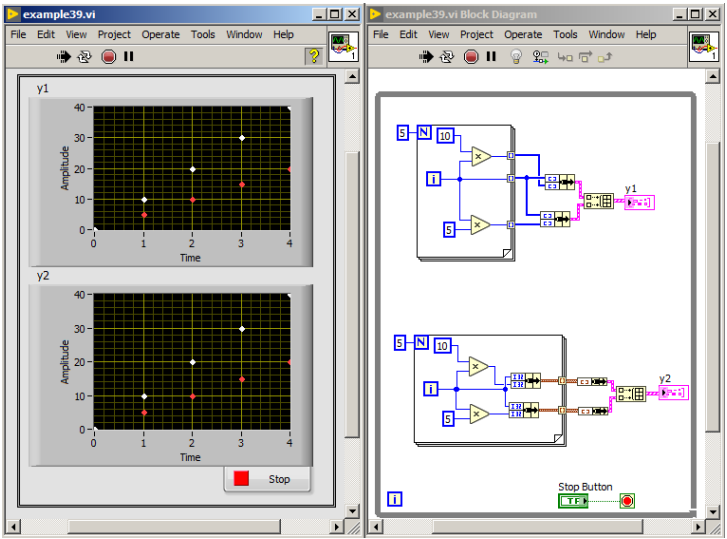
Graph = gotowa seria

XY Graph – gdy X nie jest równomierne



- przekazujemy osobno wartości **X** i **Y**,
- odstępy pomiędzy punktami X nie muszą być jednakowe,
- naturalny wybór dla charakterystyk $y = f(x)$, np. prąd–napięcie.

Kilka serii na XY Graph



- każda seria ma własną parę tablic X i Y,
- pojedynczą serię opisujemy klastrem {X array, Y array},
- wiele serii można przekazać jako tablicę takich klastrów.

Który wykres wybrać?

	Waveform Chart	Waveform Graph	XY Graph
Dane wejściowe	kolejne próbki	cała seria Y	pary X,Y
Oś X	równomierna	równomierna	dowolna
Historia	przechowywana przez Chart	nadpisywany przebieg co wywołanie	nadpisywany przebieg co wywołanie
Typowy przykład	temperatura na żywo	zapisany przebieg	charakterystyka np. I(U)

Czas na podsumowanie tej części wykładu!

Zamiast tekstu – seria pytań kontrolnych.

SPRAWDŹMY, CO JUŻ UMIEMY



Pytanie kontrolne

Mierzysz temperaturę raz na sekundę i chcesz na bieżąco dopisywać każdy nowy punkt do przebiegu. Co wybierzesz?

(a) Waveform Chart
(b) Waveform Graph
(c) XY Graph
(d) Array Indicator

wykresy

Pytanie kontrolne

Mierzysz temperaturę raz na sekundę i chcesz na bieżąco dopisywać każdy nowy punkt do przebiegu. Co wybierzesz?

(a) **Waveform Chart**
(b) Waveform Graph
(c) XY Graph
(d) Array Indicator

wykresy

Pytanie kontrolne

Chcesz narysować charakterystykę prąd–napięcie, a kolejne wartości napięcia nie są rozmieszczone równomiernie. Który wykres jest najlepszy?

(a) Waveform Chart
(b) Waveform Graph
(c) XY Graph
(d) Intensity Graph

wykresy

Pytanie kontrolne

Chcesz narysować charakterystykę prąd–napięcie, a kolejne wartości napięcia nie są rozmieszczone równomiernie. Który wykres jest najlepszy?

(a) Waveform Chart
(b) Waveform Graph
(c) **XY Graph**
(d) Intensity Graph

wykresy

Pytanie kontrolne

Który tryb Waveform Chart oddziela stare i aktualnie rysowane dane pionową linią, podobnie jak w monitorowaniu EKG?

(a) Strip Chart
(b) Scope Chart
(c) Sweep Chart
(d) Step Chart

wykresy

Pytanie kontrolne

Który tryb Waveform Chart oddziela stare i aktualnie rysowane dane pionową linią, podobnie jak w monitorowaniu EKG?

(a) Strip Chart
(b) Scope Chart
(c) **Sweep Chart**
(d) Step Chart

wykresy

Dziękuję za uwagę!

Następnie: pamięć programu, SubVI, Event Structure i zapis do pliku