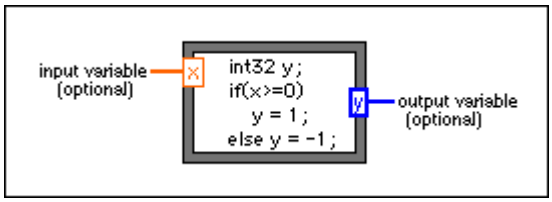
Budowa i oprogramowanie komputerowych systemów sterowania

Laboratorium 2

Wprowadzenie do Labview

Formula Node

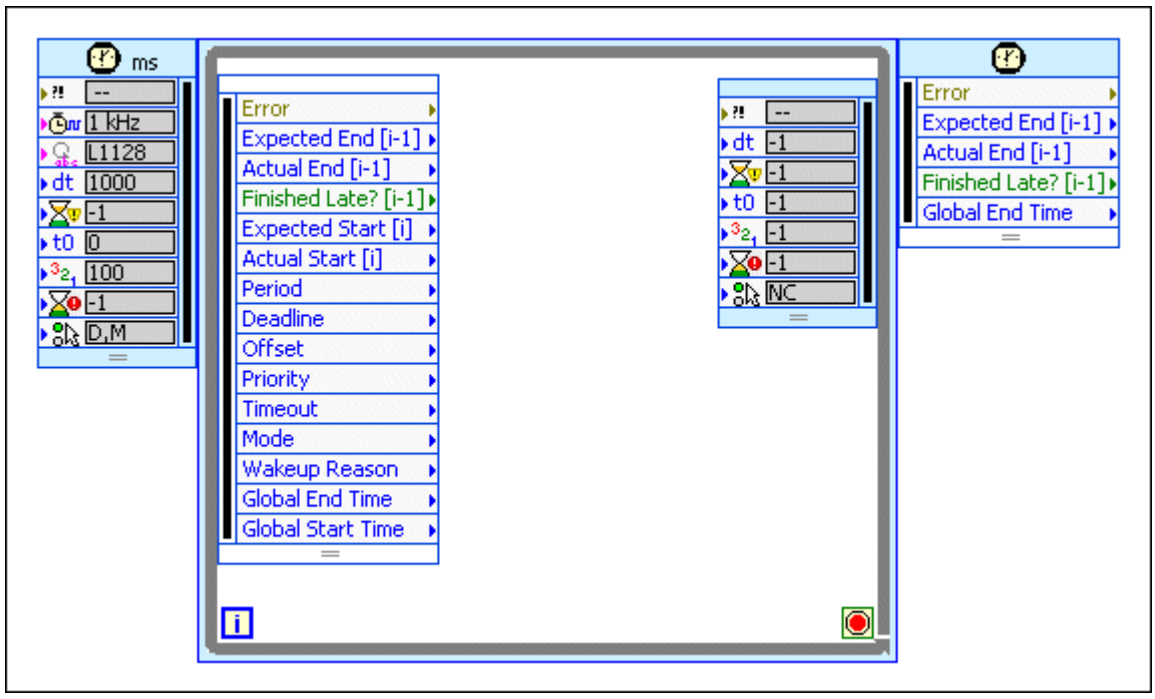
Oblicza matematyczne formuły i wyrażenia w sposób podobny do C w ramach schematu blokowego. Następujące funkcje są dozwolone w wyrażeniach abs, acos, acosh, asin, asinh, atan, atan2, atanh, ceil, cos, cosh, cot, csc, exp, expm1, floor, getexp, getman, int, intrz, ln, lnp1, log, log2, max, min, mod, pow, rand, rem, sec, sign, sin, sinc, sinh, sizeOfDim, sqrt, tan, tanh.



W celu utworzenia terminali wejściowych należy kliknąć prawym klawiszem na obramowaniu i wybrać Add Input z menu podręcznego. Wpisać nazwę zmiennej, możliwa jest późniejsza modyfikacja nazw zmiennych z wykorzystaniem narzędzi Labeling lub Operating.
Podobnie dla terminali wyjściowych należy kliknąć prawym klawiszem na obramowaniu i wybrać Add Output z menu podręcznego.
Terminale są czułe na wielkość znaków. Nie ma ograniczeń na liczbę wejśc, wyjść oraz równań. Obramowanie wejść jest cieńsze od wyjść. Domyślnym typem danych dla wyjść jest typ zmiennoprzecinkowy podwójnej precyzji. Zmianę typu możemy dokonać deklarując zmienną określonego typu w bloku Formula Node, np. int32 y.

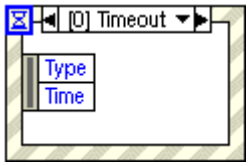
Timed Loop

Wykonuje jeden lub więcej subdiagramów lub ramek sekwencyjnie w każdej iteracji pętli w chwili określonej w ustawieniach pętli. Pętla Timed Loop jest wykorzystywana w aplikacjach wymagających precyzyjnego odmierzania czasu, lub różnych poziomów priorytetów wykonywania. Kliknięcie prawym klawiszem myszy na ramce pętli umożliwia dodawanie ramek.

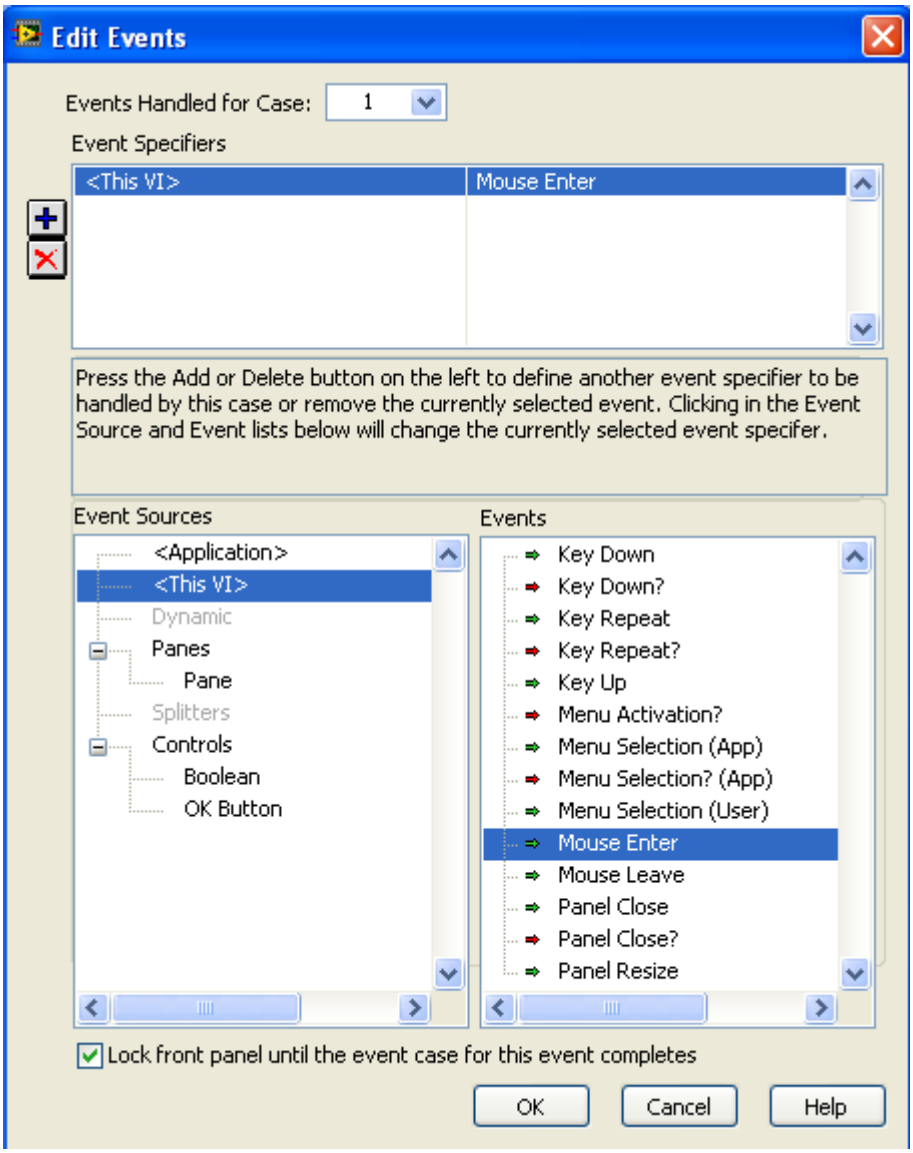


Event Structure

Struktura Event może mieć jeden lub wiele subdiagramów lub zdarzeń, tylko jeden z nich jest wykonywany kiedy struktura jest wykonywana. Struktura Event czeka na pojawienie się zdarzenia i następnie wykonuje odpowiadający temu zdarzeniu diagram.



Kliknięcie prawym klawiszem na ramce pozwala dodawać nowe zdarzenia i konfigurować które zdarzenia mają być obsługiwane.



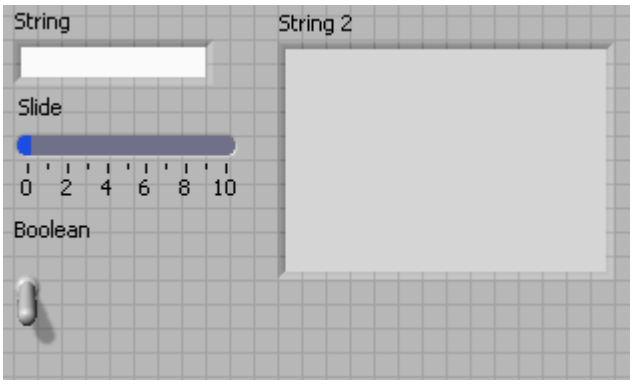
Podłączenie terminala w lewym górnym rogu pozwala określić liczbę milisekund jaką struktura ma czekać na wystąpienie zdarzenia. Domyślne jet -1 co oznacza, że czeka w nieskończoność.

LabVIEW - Strings

Łańcuchy (String) są w językach programowania typem danych, służącym do przechowywania ciągu znaków. Mogą służyć do przechowywania tekstów. Odgrywają również istotną rolę przy obsłudze plików - w nich przechowywane są dane zapisywane i odczytywane z plików. LabVIEW implementuje kontrolki i funkcje umożliwiające posługiwanie się łańcuchami.

Zupełnie podstawowymi elementami obsługi łańcuchów są kontrolki i wskaźniki (String Controls, String Indicators). Wyglądem są one zbliżone do kontrolek i wskaźników numerycznych (Numeric Controls, Numeric Indicators). Ich funkcjonowanie jest również podobne - pozwalają wprowadzać i wyyświetlać dane, jakimi są w tym przypadku teksty.
Funkcje zaimplementowane w LabVIEW pozwalają nam na różnego rodzaju operacje na tekscie. Do nich możemy zaliczyć m.in.:

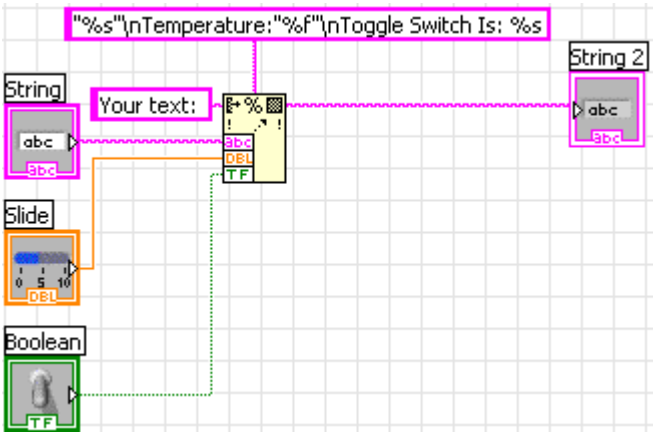
formułowanie (tworzenie) łańcuchów
funkcje konwersji łańcucha (na liczby i inne typy danych; oraz im odwrotne)
łączenie i dzielenie łańcuchów
wyszukiwanie fragmentów tekstu
formatowanie daty/czasu
inne
Poniższy przykład zapozna nas z tworzeniem łańcuchów. Panel frontowy w tym przypadku będzie jak pokazano poniżej.



Panel frontowy strings1.vi

Kontrolkę (String Control) znajdziemy w palecie kontrolek (Text Ctrls > String Ctr), na panelu frontowym.
Wskaźnik (String Indicator) odnajdziemy w tej samej palecie (Text Inds > String Ind).
Przełącznik - Buttons > Toggle Switch.
Suwak - Num Ctrls > Fill Slide.

Połączmy elementy w poniższy sposób:



Formatowanie stringa znajdziemy na palecie funkcji: All Functions > String > Format Into String. Stałą łańcuchową (String Constant) - odnajdziemy w All Functions > String > String Constant. Jedną z nich należy: ' "%s"\nTemperature:"%f"\nToggle Switch Is: %s' i podłączyć do terminala format string (jak wyżej). Drugą natomiast należy wypełnić: "Your text: "

Przed uruchomieniem - kliknijmy prawym przyciskiem na kontrolkę "String" na panelu frontowym prawym przyciskiem myszy. Wybierzmy Update Values while Typing. Zapewni to że wartości będą aktualizowane w trakcie modyfikacji pola.

Po uruchomieniu przykładu przyciskiem Run Continously spróbujmy wypełnić pola, zmienić wskaźnik, czy wreszcie przełączyć przełącznik. Powinniśmy zauważyć różnicę w wypisywanym tekscie.

Analiza

Wypisywany tekst jest formatowany przez funkcję Format Into String znajdującą się na diagramie blokowym. O sposobie formatowania decyduje łańcuch formatujący (Format String): ' "%s"\nTemperature:"%f"\nToggle Switch Is: %s' .

W miejsce znaków %s, %f,%s - wstawiane są sformatowane dane wejściowe (w kolejności podłączenia do terminalów wejściowych).
Odwoływanie się do zmiennych w łańcuchu formatującym odbywa się za pomocą poniższych znaków %. Wyjątek stanowi znak (%%), który nie odwołuje się do żadnej zmiennej.

Znak	**Funkcja**
%b	wypisuje liczbę binarny (1011)
%d	wypisuje liczbę całkowity (12)
%f	wypisuje liczbę zmiennoprzecinkowa (np. 12,345)
%g	wypisuje liczbę zmiennoprzecinkowy / lub z eksponentą (np. 12,345)
%o	wypisuje liczbę ósemkowy (701)
%s	wypisuje łańcuch (abc)
%x	wypisuje liczbę hex (B8)
%%	nie odwołuje się do zmiennej, lecz wypisuje znak %

Na samym początku łańcucha wyjściowego wstawiany jest łańcuch inicjujący (inital string). W naszym przypadku jest nim 'Your text: '.

Funkcja Format Into String zawiera rówież obsługę błędów. Są one przekazywane do terminala error out, po zsumowaniu z błędami wejściowymi error in.

Oprócz wyżej wymienonych znaków - istnieją inne znaki, takie jak znak przejścia do nowej linii (\n). Jest więcej takich znaków. Większość z nich została wymieniona w poniższej tabeli.

Znak	**Funkcja**
\n	wypisuje początek nowej linii
\t	wypisuje tabulator
\\	wypisuje znak \
\hex	wypisuje znak ASCII o numerze hex, np. \64 wypisze setny znak z tablicy ASCII (znak d) (100 dec. = 64hex)) d, zaś \20

LabVIEW - SubVI

SubVI - jest to funkcja (definiowana przez użytkownika), stosowana w LabVIEW. SubVI jest opisane w podobny sposób co VI. SubVI posiada wejścia i wyjścia. Posiada ikonę, która będzie reprezentować SubVI. VI może zawierać funkcje dostępna z palety (Functions Palette), jak również funkcje definiowane przez użytkownika (SubVI).

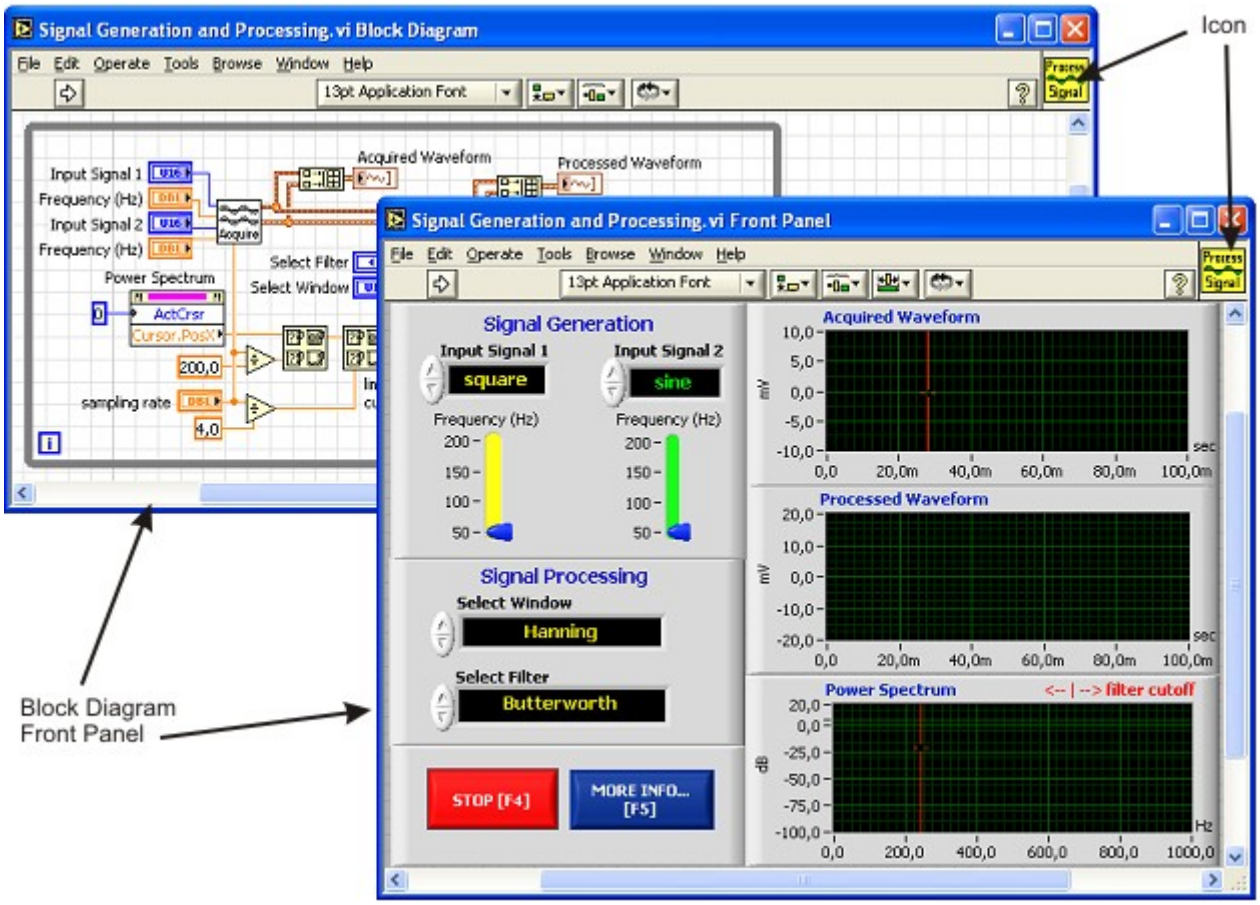
Użytkownik po stworzeniu VI - może zdecydować się że pewien fragment programu jest tak często używany, że warto go zgrupować i przedstawić w postaci czarnej skrzynki. Taką czarną skrzynką będzie SubVI. Będzie zawierać wejścia i wyjścia, a jego reprezentacją będzie ikona. SubVI stanowi osobny plik (.vi), który może istnieć niezależnie od wszystkich ikon, lub też może zostać włączony do biblioteki. SubVI to nic innego jak kolejny VI, który może istnieć niezależnie od wszelkich ikon. Może zostać również włączony do biblioteki.

Główne zalety SubVI:

- łatwość reprezentacji skomplikowanego kawałka programu w postaci ikony, oraz uzyskiwana czytelność programu
- osobny plik z SubVI pozwala na zastosowanie w różnych projektach

Wady SubVI:
- zewnętrzny plik może zapodziać się i spowodować że projekt VI nie będzie funkcjonować (jest to w prawdzie wina programisty, niemniej jednak nadmiar plików może być nieco kłopotliwy)
- niedokładnie zdefiniowane wartości początkowe SubVI mogą być przyczyną nieprawidłowego funkcjonowania algorytmu (należy zwracać uwagę np. na długość Array w SubVI)
SubVI jest niczym innym jak kolejnym VI. Sub składa się analogicznie z

- Panelu frontowego
- Diagramu blokowego
- Ikony



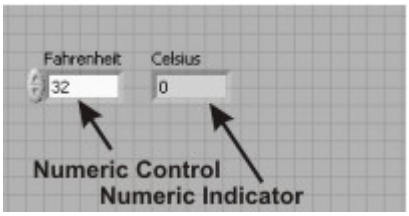
Tworzenie SubVI

tworzymy panel frontowy (Front Panel) i diagram blokowy (Block Diagram), ikonę (Icon)
sprawdzamy funkcjonowanie
przypisujemy kontrolki i wyjścia projektu - odpowiednim terminalom

Poniżej pokazano skrótowo tworzenie SubVI. Będzie on konwertować temperaturę Fahrenheit na Celsius, zgodnie ze wzorem:

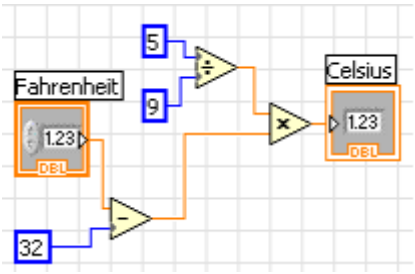
TempC = (5/9) * (TempF - 32)

Tworzymy poniższy panel frontowy.



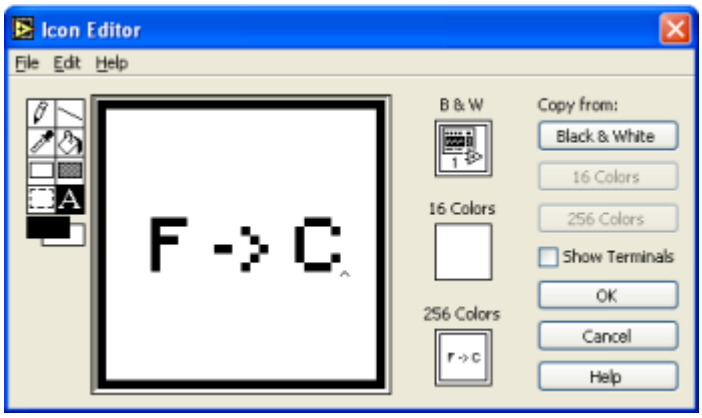
LabVIEW - SubVi - Panel frontowy

Tworzymy poniższy diagram blokowy.



LabVIEW - SubVi - Diagram Blokowy

Należy również wyedytować ikonę. W tym celu należy kliknąć prawym przyciskiem ikonę w oknie diagramu blokowego (lub panelu frontowgo). Z listy pop-up wybieramy Edit Icon i edytujemy ją.



LabVIEW - SubVi - Edycja ikony

Testujemy funkcjonowanie. Z palety narzędzi (Tool Palette) wybieramy Operate Value:

Operate Value - pozwala na zmianę wartości wybranego elementu

Modyfikujemy wartość temperatury Fahrenheit w panelu frontowym, a następnie klikamy przycisk Run, lub wybieramy z paska Menu Operate > Run w celu uruchomienia programu. Powinna pojawić się nowa wartość stopni Celsjusza.

Jeżeli mamy już gotowy VI, który nas satysfakcjonuje - należy przypisać terminale. W tym celu wybieramy z palety narzędzi (Tool Palette) Connect Wire.

Connect wire - łączy elementy

Będziemy łączyć terminale naszego SubVI z kontrolkami. Klikamy na ikonę panela frontowego prawym przyciskiem i wybieramy Show Connector. Pojawią się dwa prostokąty symbolizujące terminale. Klikamy ponownie - tym razem lewym przyciskiem na lewy prostokąt. Zmieni on kolor na czarny. Następnie klikamy na kontrolkę Fahrenheit. Prostokąt zmieni kolor na brązowy, co będzie oznaczało że jest przypisany. Następnie klikamy na drugi terminal (prawy prostokąt) lewym przyciskiem. Klikamy lewym przyciskiem na wskaźnik Celsius. Oba brązowe prostokąty oznaczają, że SubVI jest w pełni przypisany.

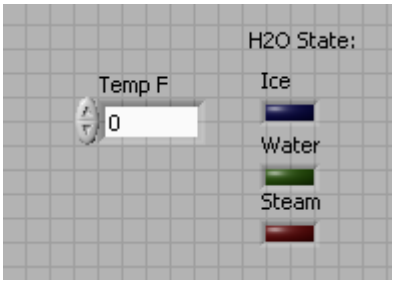
W celu powrotu do ikony - klikamy na dowolny prostokąt prawym przyciskiem myszy i wybieramy Show Icon. Zapisujemy nasz SubVI pod nazwa FtoC.vi.

W ten sposób stworzyliśmy nasz SubVI, który użyjemy do nowego projektu.

Korzystanie z SubVI

Tworzymy nowy projekt VI. Wykorzystamy w nim stworzone uprzednio SubVI.

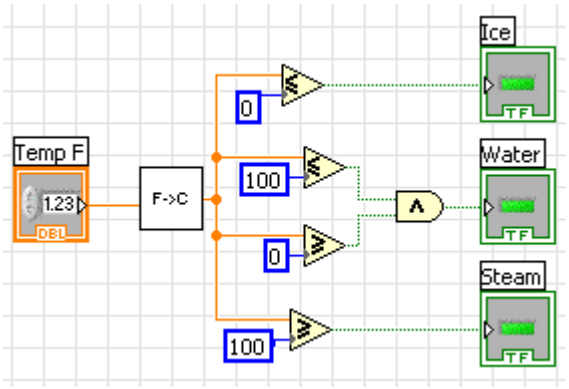
Tworzymy poniższy panel frontowy.



LabVIEW - VI - Panel frontowy

Diody Ice, Water, Steam wybieramy z palety kontrolek (Controls Palette): LEDs > Square LED
Kolor diod możemy zmienić. W tym celu należy kliknąć prawym przyciskiem na diodę i wybrać z Properities. Zakładka Appearance zawiera pole Colors. Klikając na kolory diodę zapaloną (On) i wyłączoną (Off). Możemy zmienić jej kolor.

Tworzymy poniższy diagram blokowy.



LabVIEW - SubVi - Diagram Blokowy

SubVI FtoC.vi wybieramy z palety funkcji. (Functions Palette). Klikamy na All Functions w palecie. W lewym dolnym rogu znaduje się Select a VI.... Klikamy, a następnie wybieramy interesujący nas VI. W tym przypadku będzie to FtoC.vi.
Funkcje porównania znajdziemy w palecie funkcji - Arith/Compare > Comparision.
Tam też możemy wybrać porównanie logiczne: Arith/Compare > Boolean.

Przechodzimy do panelu frontowego i klikamy Run Continously.
Zmieniamy temperaturę Temp F - tak, aby zapaliły się diody wskazujące na stan ciała (Ice, Water, Steam). Przykład ten nauczy nas przy jakiej temperaturze Fahrenheit zamarza i wrze woda.

LabVIEW - Files

Istnieje kilka sposobów obsługi plików w LabVIEW. Daje to dużą elastyczność programiście VI. W zależności od potrzeb może on użyć bardziej lub mniej skomplikowanych funkcji.

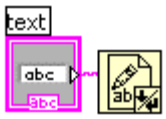
Wszystkie funkcje obsługi plików zostały zorganizowane w funkcje:

- Wysokiego poziomu VI (High-Level I/O VIs)
- Funkcje niskiego poziomu (Low-Level File I/O Functions)
- Funkcje zaawansowane (Advanced File I/O)

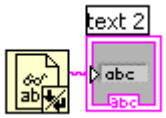
High-Level I/O VIs - są to funkcje w postaci SubVI; Te pliki VI wywołują funkcje niskiego poziomu (Low-Level File I/O); dzięki temu posiadają prosty interfejs (terminale) i są łatwiejsze, ale zarazem mniej elastyczne.

Przykład - zapis i odczyt ciągu znaków

Jako jeden z najprostszych przykładów możemy wziąć funkcję zapisu i odczytu ciągu znaków.



Najprostszy zapis

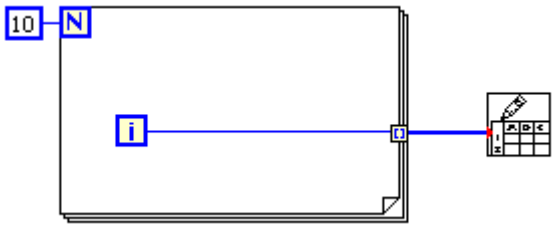


Najprostszy odczyt

Przy zapisie skorzystano z funkcji Write Text to File.vi . Wartością zapisywaną jest stała typu łacuchowego (String) o treści "Tekst". Przy odczycie skorzystano z funkcji Read Text from File.vi. Wynik funkcji zostaje przekazany do wskaźnika Text Indicator.

Przykład - zapis arkusza (Spreadsheet) 1D

Funkcje wysokopoziomowe ułatwiają szybkie i sprawne zapisywanie tablic. Jako przykład możemy posłużyć się poniższym diagramem blokowym VI. Przy zapisie skorzystano z funkcji Write To Spreadsheet.vi . W przykładzie zapisujemy tablicę wygenerowaną w pętli for.

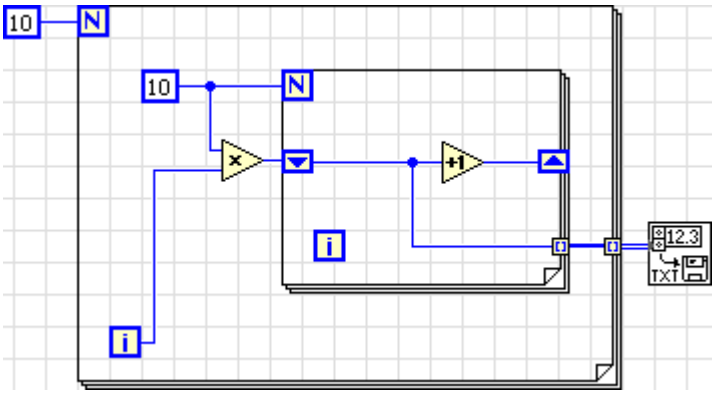


Zapis tablicy (Array) jednowymiarowej do pliku.

Przykład - zapis i odczyt arkuszy (Spreadsheet), 2D

Zapis 2D

Wykorzystując tę samą funkcję Write To Spreadsheet.vi - możemy zapisać również do pliku dane wielowymiarowe. Zostało to przedstawione na poniższym diagramie blokowym przykładu write2spreadsheet2d.vi. W pierwszym etapie działania - funckja generuje tablicę dwuwymiarową, korzystając z zagnieżdżonej pętli. Tablica (array) ta zostaje przekazana do funkcji wysokopoziomowego zapisu. Ponieważ funkcja Write To Spreadsheet.vi nie posiada zadeklarowanych żadnych innych informacji - toteż po uruchomieniu przykładu - LabVIEW spyta o nazwę pliku do zapisu.



Zapis tablicy (Array) dwuwymiarowej do pliku.

Low-Level File I/O Functions - funkcje niskiego poziomu posiadają nieco bardziej złożony interfrejs (więcej terminali), są bardziej elastyczne, ale zarazem wymagają większej ilości argumentów.

Advanced File I/O Functions - funkcje zaawansowane pozwalają na jeszcze bardziej skomplikowany dostęp do plików.

Oprócz podziału ze względu na rodzaje terminali - można dokonać podział ze względu na typ dostępu. Możliwe są dwie opcje:

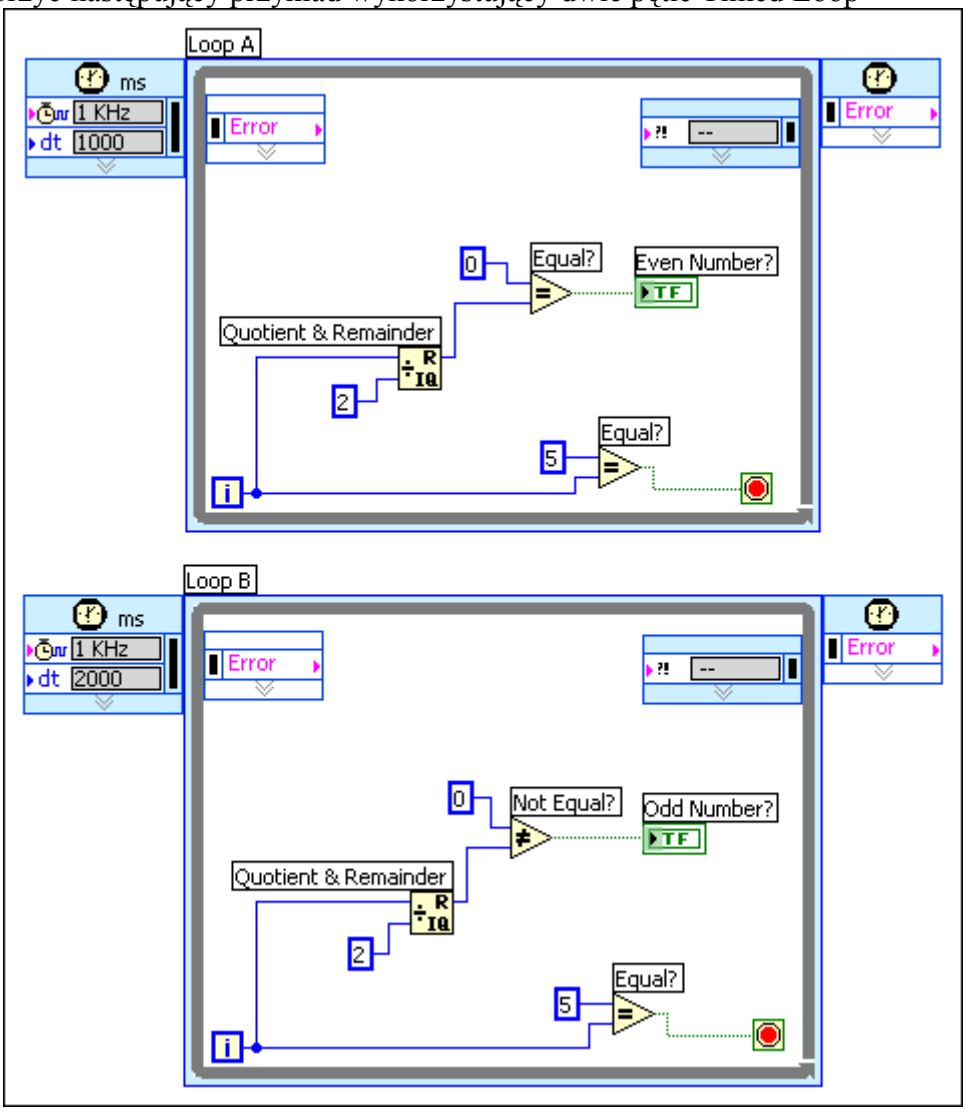
pliki tekstowe - mają przede wszystkim zastosowanie, gdy pliki przeznaczone są do odczytu dla człowieka; pliki tekstowe nie zawierają dziwnych znaków, powinny być bez przeszkód rozczytywalne przez człowieka; wygodnie jest stować pliki tekstowe, do których zapisujemy uprzednio sformatowane łańcuchy znaków (String). Takie formatowanie i zapis do pliku tekstowego jest rzeczą łatwą, szybką, oraz daje użytkownikowi gotowy, łatwy do zrozumienia plik.

pliki binarne - służą przede wszystkim do składowania danych w wygodny dla programisty sposób; pliki te najczęściej nie są czytelne (zrozumiałe) dla człowieka. Najlepiej stosować je do zapisu liczb lub innych danych.

LabVIEW obsługuje także pliki typu Datalog, umożliwiające zapisu lub odczyt struktur. Są one podobne budową do arkuszy (Spreadsheet) w których każdy wiersz zawiera rekord (strukturę). Wszystkie elementy które LabVIEW zapisuje do jednego pliku - muszą być takiego samego typu. Przypomina to umieszczanie danych w klastrach. Aby operacja powiodła się - muszą być tego samego typu.

Zadania

1. Utworzyć i uruchomić wszystkie przykłady znajdujące się w instrukcji.
2. Zapoznać się z elementami palety Programming>Array i napisać przykłady wykorzystujące następujące funkcje: Array Size, Index Array, Insert Into Array, Initialize Array, Array Max & Min, Sort 1D Array.
3. Utworzyć VI generujące tablicę 1D, a następnie przemnożyć przez siebie elementy w parach startując od 1 i 2, następnie 3 i 4 itd. Wykorzystać element Decimate 1D Array z palety Programming>Array.
4. Zapoznać się z elementami palety Programming>Strings i napisać przykłady wykorzystujące następujące funkcje: String Length, Concatenate Strings, String Subset, Search and Replace String, Match Pattern
5. Utworzyć VI wykorzystujące Formula Node do obliczenia równań:
   y1 = x3 + x2 + 5
   y2 = mx + b
6. Utworzyć następujący przykład wykorzystujący dwie pętle Timed Loop



7. Utworzyć program kalkulatora wykorzystujący Event Structure do obsługi zdarzeń generowanych przez klawisze.
8. Zapoznać się z przykładami obsługi plików dostępnymi w środowisku Labview: Read from Text File.vi, Write to Text File.vi, Write Binary File.vi, Read Binary File.vi. Dokładnie opisać działanie tych przykładów i wykorzystać do utworzenia własnych przykładów.