|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| logoKM | logoKM | | POLITECHNIKA ŚLĄSKA  WYDZIAŁ ELEKTRYCZNY  KATEDRA MECHATRONIKI | logoKM |
| Instrukcja do ćwiczenia laboratoryjnego | | | | |
| Przedmiot: | | Przetwarzanie i Wizualizacja Danych Pomiarowych | |  |
| Symbol ćwiczenia: | | PiWDP4 | |  |
| Tytuł ćwiczenia: | | **Pętle w środowisku LabVIEW** | | |

**SPIS TREŚCI**

|  |  |  |
| --- | --- | --- |
|  | **SPIS RYSUNKÓW** | 2 |
| 1. | **CELE ĆWICZENIA** | 3 |
| 2. | **WPROWADZENIE** | 3 |
| 2.1. | **Pętla While** | 3 |
| 2.2. | **Pętla FOR** | 5 |
| 3. | **LABORATORYJNE STANOWISKO BADAWCZE** | 6 |
| 3.1. | **Obiekt badany** | 6 |
| 3.2. | **Urządzenia dodatkowe** | 6 |
| 3.3. | **Oprogramowanie** | 7 |
| 4. | **PROGRAM ĆWICZENIA – WYKAZ ZADAŃ DO REALIZACJI** | 7 |
| 5. | **PRZYKŁAD REALIZACJI ZADANIA – Przygotowanie symulowanej karty DAQ** | 8 |
| 6. | **RAPORT** | 9 |
| 7. | **PYTANIA** | 9 |
|  |  |  |

# SPIS RYSUNKÓW

|  |  |  |
| --- | --- | --- |
| 1. |  |  |
| 2. |  |  |
| 3. |  |  |
| 4. |  |  |
| 5. |  |  |
| 6. |  |  |
| 7. |  |  |
| 8. |  |  |
| 9. |  |  |

1. **CELE ĆWICZENIA**

Celem ćwiczenia jest zapoznanie się z działaniem i zastosowaniem pętli programowych w środowisku LabVIEW:

* Poznanie właściwości i zastosowania pętli While
* Poznanie właściwości i zastosowania pętli For
* Poznanie struktur sekwencyjnych
* Poznanie najważniejszych informacji nt. klastra błędu

1. **WPROWADZENIE**

Pętle w środowisku LabVIEW są zamkniętymi obszarami otaczającymi fragment kodu, który ma być wykonywany wielokrotnie. **Indeksowanie pętli (tak samo jak indeksowanie wektorów) zaczyna się od wartości 0!** Wejścia i wyjścia pętli nazywamy tunelami odpowiednio wejściowymi i wyjściowymi. Istnieją cztery typy tuneli:

- tunel zwykły (**Last Value**) – (wejściowy) wprowadza dane do pętli, we wszystkich iteracjach wprowadza tą samą wartość, (wyjściowy) wyprowadza do dalszego fragmentu kodu wartość z ostatniej iteracji,

- tunel autoindeksowany (**Indexing**) – służy do współpracy pętli z wektorami danych, (wejściowy) w kolejnych iteracjach pętli wprowadza dane z kolejnych komórek wektora – w 0 iteracji wprowadza wartość z 0 elementu wektora, w 1 iteracji wprowadza wartość z 1 elementu wektora itd. (wyjściowy) tworzy wektor zawierający kolejne wartości obliczone w kolejnych iteracjach pętli, rozmiar wektora jest równy liczbie iteracji pętli,

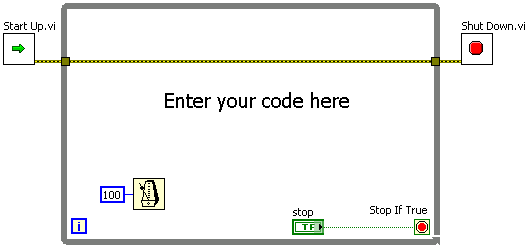
- tunel łączący (**Concatenating**) – służy do uzupełniania wektora kolejnymi komórkami, istnieje tylko jako tunel wyjściowy,

- rejestr przesuwny (**Shift Register**) – tunel przenoszący wartość z poprzedniej iteracji do aktualnej, może być inicjowany wprowadzając daną do pierwszej iteracji, w przypadku, gdy nie jest inicjowany wprowadza domyślną lub zapamiętaną wartość, jako jedyny pojawia się po obydwu stronach pętli, może zostać rozszerzony, żeby „pamiętać” wartości z więcej niż jednej iteracji.

W przypadku pierwszych trzech tuneli (wyjściowych) można dodatkowo uaktywnić opcję warunkową (Conditional). W tym przypadku tylko wartości, dla których boolowski warunek jest spełniony zostają wyprowadzone.

* 1. **Pętla While**

Petla **While** jest główną pętlą programu, wykonuje się aż do spełnienia warunku wyjścia. Warunek wyjścia jest typu boolowskiego i może być ustawiony jako „Stop if True” lub jako „Continue if True”. Pętla **While** wykonuje się zawsze co najmniej raz – warunek wyjścia sprawdzany jest na końcu. Na Przedstawiono wygląd pętli While w podstawowej strukturze programu.



1. Pętla While w podstawowej strukturze programu.

Pętlę **While** stosuje się najczęściej jako główną pętlę programu, warunek zatrzymania jest wywoływany za pomocą przycisku kończącego aplikację (*stop*), może także być zatrzymywana np. w wyniku błędu. Pętlę **While** z niezainicjowanym rejestrem przesuwnym oraz strukturą **case** stosuje się jako tzw. zmienną funkcyjną (więcej informacji na temat zmiennych funkcjonalnych będzie na ćwiczeniu ze struktur programistycznych).

* 1. **Pętla FOR**

Podstawowym zastosowaniem pętli **FOR** są operacje na wektorach, dlatego indeksowanie pętli **FOR** oraz wektorów jest takie samo (od zera). Pętla **FOR** może nie zostać wykonana ani razu, jeżeli na wejście licznikowe zostanie podana wartość „0” lub wektor podany na wejście autoindeksujące ma rozmiar „0”. Opcjonalnym elementem pętli jest terminal warunkowy uaktywniany **PPM** 🡪 **Conditional Terminal**.

|  |  |
| --- | --- |
| a)  forloopconditionalex.bmp | b) |
| 1. Pętla FOR: widok ogólny (a), generacja 10-cio elementowego wektora o losowych wartościach (b). | |

* 1. **Struktury sekwencyjne**

W LabVIEW istnieją dwie struktury sekwencyjne **Flat Sequence Structure** oraz **Stacked Sequence Structure**. Ich widok przypomina kolejne klatki filmu. Ich działanie jest identyczne, różnią się prezentacją graficzną w oknie diagramu. Struktura płaska (**Flat**) jest bardziej czytelna jednak wymaga więcej miejsca ponieważ widać wszystkie klatki. Struktura nakładana (**Stacked**) wyświetla widok tylko jednej klatki na raz, pozostałe są ukryte pod nią więc zajmuje mniej miejsca jednak kod pozostałych kratek jest ukryty, przez co jest mniej czytelny. Można bardzo szybko dokonać zmiany struktury z płaskiej na nakładaną i odwrotnie klikając **PPM** i wybierając odpowiednią opcję z menu podręcznego, struktury sekwencyjne można konwertować także na inne struktury.

* 1. **Klaster błędu**

1. **LABORATORYJNE STANOWISKO BADAWCZE**
   1. **Obiekt badany**

- Środowisko programistyczne LabVIEW,

* 1. **Urządzenia dodatkowe**

- brak,

* 1. **Oprogramowanie**
* LabVIEW 2013 lub nowszy

1. **PROGRAM ĆWICZENIA – WYKAZ ZADAŃ DO REALIZACJI**

Kolejne kroki do wykonania podczas zajęć:

- uruchomić nowy projekt „*System pomiarowy DAQ*”,

- w oknie projektu utworzyć nowy plik vi na poziomie *My Computer*,

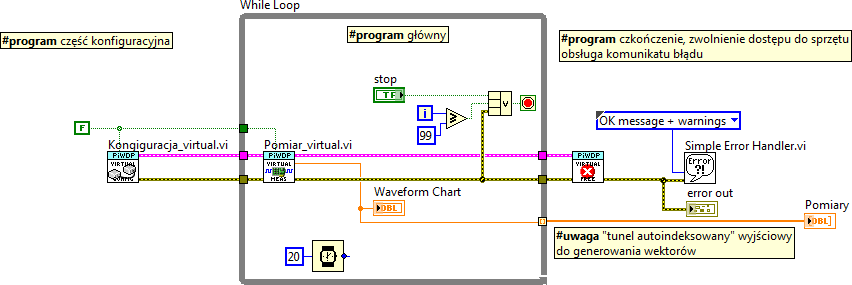
- zapisać plik pod nazwą „*main.vi*” w katalogu „*Stacja pogodowa/*”,

- uzupełnić plik zgodnie z rysunkiem,

- uzupełnić program główny, aby liczył średnią biegnącą z ostatnich 5 pomiarów,

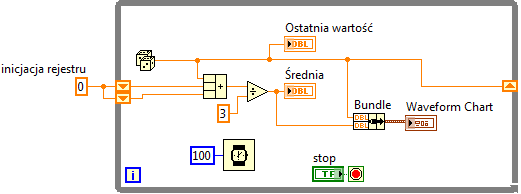
- zapisać plik,

- uzupełnić program w odpowiednim miejscu, o konwersję danych z karty na stopnie Celsjusza z zastosowaniem pliku „*temp konwersja.vi*”,

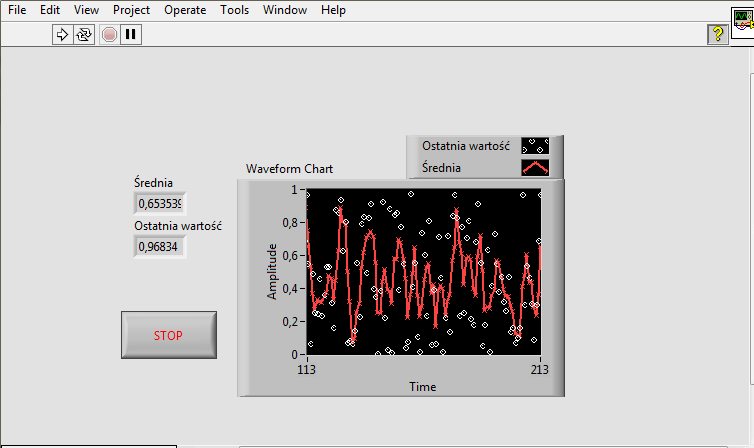


1. **PRZYKŁAD REALIZACJI ZADANIA – średnia biegnąca**

Realizacja zadania liczenia średniej biegnącej z 3 ostatnich pomiarów. Na (Rys. 3) przedstawiono przykład realizacji zadania liczenia średniej biegnącej dla trzech ostatnich wartości wylosowanych. Na (Rys. 4) przedstawiono przykładowy panel programu. Na wskaźniku typu Waveform Chart przedstawiono dwie wartości: Ostatnia wartość – w postaci białych punktów, Średnia – w postaci czerwonego przebiegu z zaznaczonymi punktami.



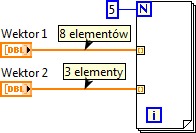
1. Program do liczenia średniej biegnącej.



1. Widok okna programu do liczenia średniej biegnącej
2. **RAPORT**

Raport z przeprowadzonego ćwiczenia laboratoryjnego powinien zawierać opis kolejnych czynności wykonywanych w trakcie realizacji ćwiczenia, zrzuty ekranu dokumentujące wykonane kroki oraz zanotowane parametry konfiguracyjne kart DAQ (mogą być zawarte  
w tabeli)

1. **PYTANIA**
2. Wymienić podobieństwa i różnice pętli **While** oraz **FOR**.
3. Ile razy zostanie wykonana pętla na poniższym rysunku?



# LITERATURA

1. LabVIEW Core 1 Course manual.
2. LabVIEW Core 1 Exercise book.
3. LabVIEW Core 2 Course manual.
4. LabVIEW Core 2 Exercise book.
5. Nota katalogowa LM35

Opracowanie: Marek Kciuk

# ZADANIA DO REALIZACJI PRZED PRZYSTĄPIENIEM DO ĆWICZENIA

- brak

# ZADANIA DODATKOWE PO WYKONANIU ĆWICZENIA

- poprawić estetycznie okno programu „*main.vi*”

- poprawić wskaźnik **Waveform Chart** aby prezentował dane w taki sam sposób, jak to zostało przedstawione na Rys. 4.

# ZAŁĄCZNIKI

## Linkowanie tuneli wejściowych z wyjściowymi

Często zdarza się,…

## Pliki polimorficzne

Pliki polimorficzne składają się z wielu