

LabVIEW

PLATFORMA EDUKACYJNA

Lekcja 1

Pierwsze kroki w środowisku LabVIEW

Przygotowali: Paulina Grela, Sylwia Jabłońska,
Kamil Rychlewicz, Arkadiusz Szczech

1. Tworzenie nowego projektu


- a. Aby utworzyć nowy projekt należy włączyć środowisko LabVIEW i wybrać przycisk Create Project.
- b. Następnie otworzy nam się okno dialogowe z wyborem rodzaju projektu. Wybieramy opcję Blank Project, która utworzy nam pusty projekt i zatwierdzamy wybór przyciskiem Finish.
- c. Po utworzeniu projektu musimy dodać do niego nowy plik VI. Aby to zrobić wybieramy opcję File, a następnie New VI.
- d. Zostaną otwarte dwa okna programowe – Block Diagram oraz Front Panel, w których będzie tworzony nasz program.
- e. Możemy użyć skrótu klawiszowego Ctrl + T, w celu domyślnego rozmieszczenia okienek programowych.
- f. Po utworzeniu nowego VI powinny nam się pojawić także opcje wyboru bloków tworzących program – Functions Palette/Controls Palette. Jeżeli nie otworzyły nam się one domyślnie możemy je otworzyć wybierając View, a następnie odpowiednią paletę funkcji.


2. Uruchamianie i debugowanie programu


W oknach Block Diagram oraz Front Panel w sekcji Operate można dostrzec następujące funkcje:

Run (Ctrl+R) powoduje natychmiastowe wykonanie programu. Jeśli program nie zawiera pętli głównej wykona się on bardzo szybko my zobaczymy tylko efekt końcowy.


Gotowość do wykonania programu sygnalizuje biała strzałka: .



Natomiast jeśli biała strzałka zmieniła się na przelatującą  to program sygnalizuje, że wykonanie kodu w tym momencie jest niemożliwe, ponieważ zawiera błąd. Przykładem błędu jest niepodłączone wejście funkcji, za sprawą którego nie będzie możliwe przekazanie do niej żadnych parametrów spowoduje to wykrycie błędu, ponieważ program po każdej operacji dokonuje sprawdzenia, czy jest on poprawnie skonstruowany.

Po dwukrotnym naciśnięciu na  pojawi się okno Error list, w którym będzie można przeczytać jakie błędy zostały wykryte w naszym programie.




Run Continuously dostępny jest pod przyciskiem . Po wybraniu tej opcji program zostanie wykonany z pełną prędkością i będzie rozpoczął się od nowa tak jakby znajdował się w pętli.

Możliwe jest wstrzymanie  i wznowienie  wykonania kodu w dowolnym miejscu korzystając z przycisku Pause.

Aby zakończyć wykonywanie programu należy wcisnąć przycisk Abort Execution (Ctrl+.) .

Aby włączyć/ wyłączyć animację przepływu danych w naszym programie należy wcisnąć przycisk Highlight Execution  / . Najlepiej jest włączyć najpierw animację, a następnie Run wówczas program wykonuje się znacznie wolniej, a na diagramach widoczne są przepływające po połączeniach "kulki" symbolizujące przepływ danych pomiędzy elementami diagramu. Można również zaobserwować wartości, które są przekazywane. Jest to pomocne do sprawdzenia poprawności napisanego programu.

LabVIEW umożliwia również testowanie aplikacji krok po kroku. Do tego celu służą trzy przyciski.

Za pomocą przycisku Step Into (Ctrl + Down)  można uruchomić program. Wówczas zostanie wykonana jedna instrukcja. Migająca w czarnym kwadracie funkcja będzie wykonana jako następna. Gdy miga cała ramka diagramu/programu możliwe jest za pomocą przycisku Abort Execution lub Step Out (Ctrl +Up)  zakończenie pracy krokowej. Korzystając z funkcji Step Into możliwe jest również prześledzenie nawet zagnieżdżonych fragmentów kodu. W momencie dojścia do funkcji zagnieżdżonej LabVIEW przełączy śledzenie na diagram tej funkcji. Aby powrócić na wyższy poziom, należy użyć przycisku Step Out. Jeśli zamierzamy debugować tylko aktualny diagram lub pominąć debugowanie zagnieżdżonej procedury, należy skorzystać z przycisku Step Over (Ctrl + Right) . Funkcje podrzędne wykonywane są w całości, bez otwierania diagramu i przechodzenia przez wszystkie jego funkcje.

LabVIEW pozwala również na ustawianie w dowolnym miejscu programu pułapek. Aby ustawić pułapkę klikamy prawym przyciskiem myszy w miejscu, gdzie chcemy ją umieścić i wybieramy z listy Breakpoints, a następnie Set Breakpoints. Można umieścić pułapkę bezpośrednio na linii łączącej bloczki, wówczas obserwować można na niej czerwone kółko, lub umieścić całą funkcję (bloczek) w pułapce, w tym przypadku funkcja ta będzie w czerwonej ramce. Gdy zostanie ustawiony Breakpoint to można przystąpić do uruchomienia programu. W tym celu wciskamy przycisk Run, wówczas zostanie wykonana część programu tylko do momentu napotkania pułapki. Miejsce zatrzymania programu jest sygnalizowane migającym czarnym prostokątem. Aby przejść dalej, należy wcisnąć przycisk Pause, co spowoduje wykonanie programu do końca lub do następnej pułapki. Usunięcie pułapki odbywa się poprzez kliknięcie na niej prawym przyciskiem myszy i wybranie Breakpoints, a następnie Clear Breakpoints.

W środowisku LabVIEW istnieje również możliwość podglądu zmiennych przy pomocy sondy. Aby ustawić sondę klikamy prawym przyciskiem myszy na połączeniu między funkcjami i wybieramy Probe. Na połączeniu pojawi się numer sondy oraz otworzy się dodatkowe okno (Probe Watch Windows), w którym będzie wyświetlana aktualna wartość zmiennej.

3. Pierwsze ćwiczenia

3.1. Zadanie 1: Realizacja funkcji logicznych z użyciem minimalizacji Mapami Karnaugh w środowisku LabVIEW

Naszym zadaniem jest zminimalizowanie funkcji logicznej przedstawionej za pomocą poniższej tabeli prawdy, minimalizacja jej za pomocą mapy Karnaugh oraz realizacja jej na podstawowych blokach środowiska LabVIEW.

Tabela prawdy funkcji prezentuje się następująco:

X1	X2	X3	X4	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

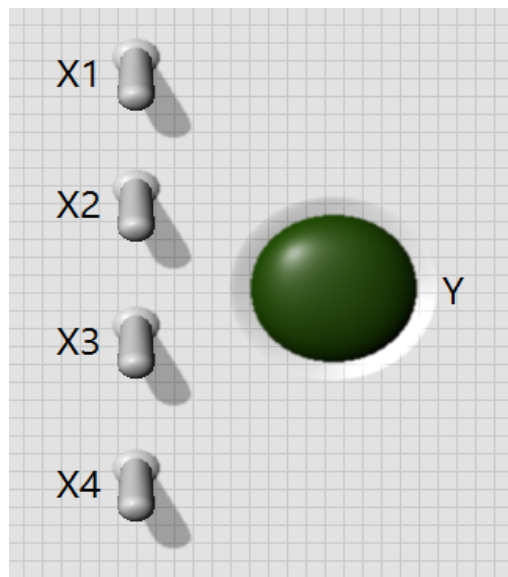
Daną funkcję możemy zminimalizować:

X1 X2 X3 X4	00	01	11	10
00	1	0	0	0
01	1	0	1	0
11	1	0	1	1
10	1	0	1	1

Z powyższej tabeli Karnaugh możemy wyznaczyć funkcję zminimalizowaną:

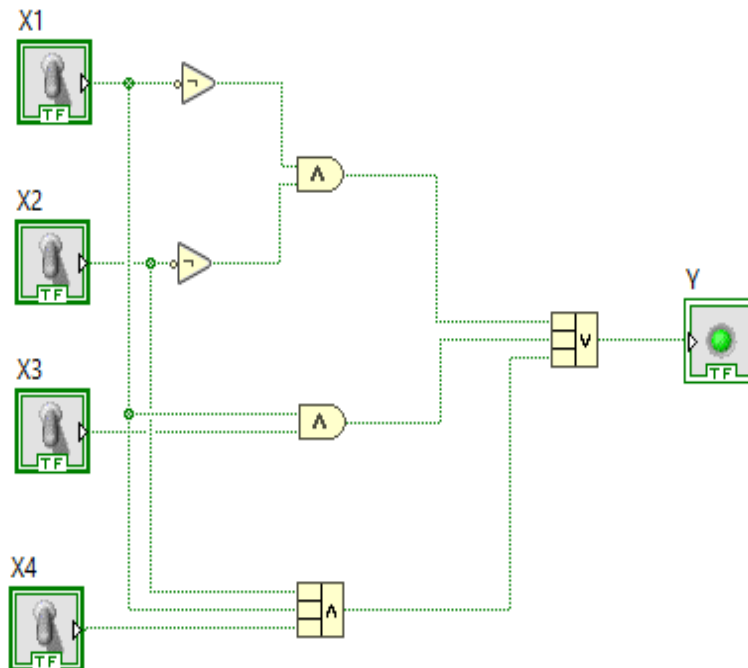
$$Y = \overline{X1X2} + X1X3 + X1X2X4$$

Na podstawie tego równania możemy zrealizować funkcję w środowisku LabVIEW. Aby to zrobić musimy w panelu frontowym dodać cztery przełączniki z kategorii Boolean. W programie użyte zostały Vertical Toggle Switch. Potrzebny będzie także element wizualizujący stan wyjścia. W tym celu do programu dodany zostanie element Round LED z kategorii Boolean. Po dodaniu tych elementów możemy zmienić ich rozmiar i nazwy. Finalny efekt dotychczasowej pracy powinien wyglądać następująco:



Następnie musimy zaprogramować odpowiednie działanie funkcji w oknie Block Diagram. Do tego celu potrzebne nam będą bramki logiczne AND, OR oraz NOT. Znajdziemy je w kategorii Programming – Boolean w Functions Palette. Wszystkie podstawowe bramki logiczne takie jak AND czy OR mają domyślnie 2 wejścia. W celu uzyskania tych bramek z większą ilością wejść należy użyć bloku Compound Arithmetic. Po jego dodaniu możemy zmieniają jego rozmiar sterować liczbą wejść, natomiast typ operacji logicznej zmieniamy klikając prawym przyciskiem myszy na dany element, a następnie nadać mu odpowiednią funkcję za pomocą opcji Change Mode.

Finalnie realizacja naszej funkcji powinna wyglądać następująco:



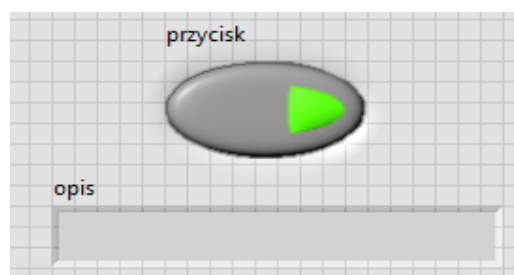
Po zrealizowaniu funkcji możemy sprawdzić jej działanie zgodnie z tabelą prawdy.

3.2. Zadanie 2: Użycie bloku Case do wyświetlania danych w zależności od wprowadzonej zmiennej.

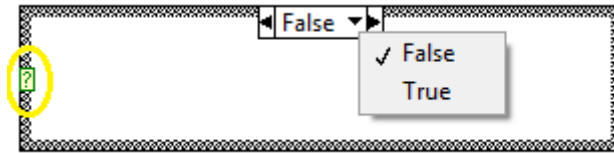
Następnym krokiem naszego kursu będzie zapoznanie z instrukcją wyboru case.

W tym celu stworzymy prosty program, który będzie wyświetlał, jaki jest stan naszego przycisku. W tym celu musimy stworzyć w oknie Front Panel przycisk: Push Button z kategorii Boolean oraz z kategorii String & Path wybrać String Indicator.

Po dodaniu tych elementów możemy zmienić ich rozmiar i nazwy. Finalny efekt dotychczasowej pracy powinien wyglądać następująco:



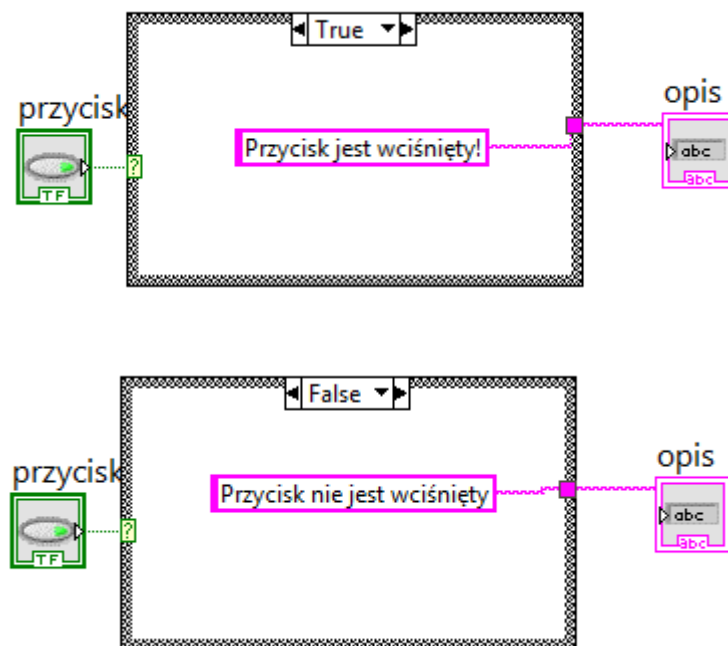
Następnie przechodzimy do zaprogramowania odpowiedniego działania funkcji w oknie Block Diagram. Mamy już tam dwa bloki: nazwane przycisk i opis. Musimy jeszcze dodać instrukcję warunkową case. W tym celu z kategorii Programming wybieramy Structures a następnie Case Structures. Powiększamy okno Case tak, aby struktura naszego programu zmieściła się w jego wnętrzu.



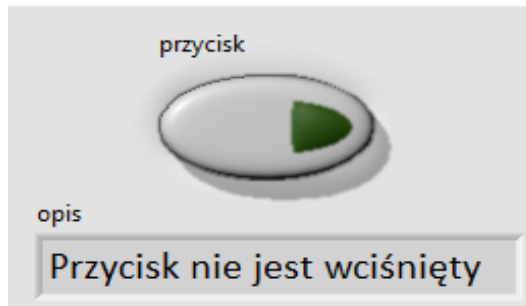
Powyżej został zamieszczony widok struktury Case. Zaznaczony na żółto element to Case Selector. Do niego podłączyć musimy element, od którego będzie zależała zmiana wyświetlanego przez nas opisu. Elementem tym jest nasz przycisk. Następnie widzimy, że instrukcja warunkowa pozwala nam na edycję programu w swoim wnętrzu w zależności od tego czy mamy na wejściu (w Case Selector) False czy True.

W momencie, gdy przycisk nie zostanie wciśnięty chcemy, aby opis wyświetlił nam, komentarz: "Przycisk nie jest wciśnięty!", zatem w strukturze Case ustawionej na False dodajemy String Constant dostępny w podkategorii String i wpisujemy tam nasz komentarz. Następnie łączymy go z naszym bloczkiem opis, który jest już poza Casem. Teraz przechodzimy do struktury ustawionej na True i postępujemy analogicznie jak wyżej, tylko wpisujemy komentarz: "Przycisk jest wciśnięty!".

Program powinien wyglądać następująco:

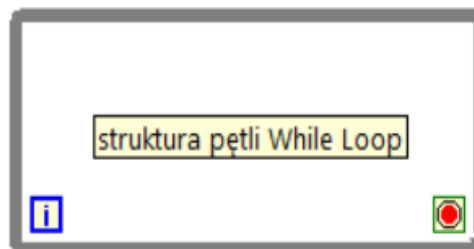


Po uruchomieniu programu, możemy sprawdzić jego poprawność.

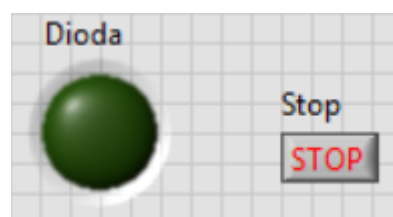


3.3. Zadanie 3: Zapoznanie z rejestrem przesuwным.

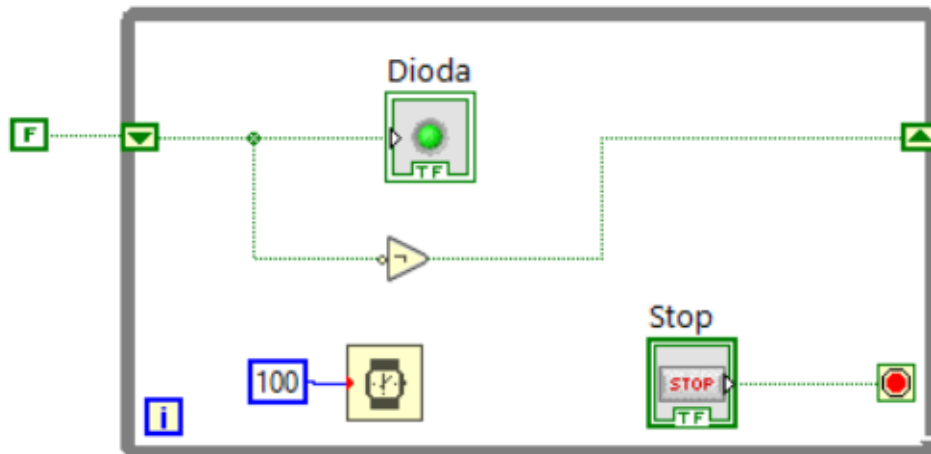
W tym punkcie kursu zapoznamy się z rejestrem przesuwным, wykonanym przy pomocy pętli While Loop.



W oknie Block Diagram dodajemy pętlę While Loop (z kategorii Programming wybieramy Structures, a następnie While Loop) nadając jej odpowiednią wielkość. W oknie Front Panel umieszczamy Round LED, która zapali się, gdy na wejściu rejestru przesuwного wartość logiczna będzie True oraz Stop Button z kategorii Boolean. Elementy automatycznie zostaną dodane do okna Block Diagram wewnątrz pętli While Loop.

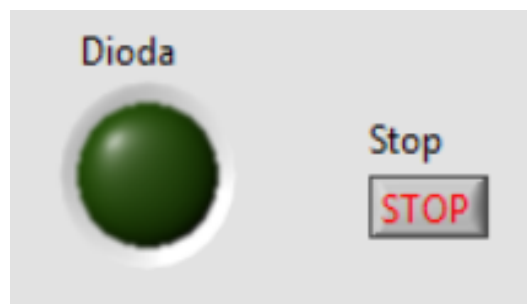


Wracamy do okna Block Diagram dodając wejście/wyjście rejestru, klikając prawym przyciskiem myszy na obwód pętli While Loop i wybieramy Add Shift Register. Z kategorii Boolean wybieramy negację Not oraz stałą False Constant, natomiast z kategorii Timing wybieramy zegar Wait(ms) podłączając do niego stałą Numeric Constant z kategorii Numeric, ustawiając na nim wartość np. 100. Program powinien wyglądać następująco:



Na wejściu rejestru przesuwne podana jest wartość logiczna False, przy kolejnej iteracji wartość logiczna zmienia się na wartość True, dzięki negacji Not. Zegar Wait(ms) daje nam możliwość sterowania czasem iteracji, tym samym czasem świecenia się diody, zmieniając stałą przy zegarze.

Po uruchomieniu programu, dioda mruga z częstotliwością 5Hz.



4. Ćwiczenia do samodzielnej realizacji

Zadanie – Zminimalizuj i zrealizuj funkcję opisaną następującą tabelą w programie LabVIEW:

X1	X2	X3	X4	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0