



Katedra Informatyki i Automatyki Politechniki Rzeszowskiej

Sterowanie procesami dyskretnymi

Instrukcja I

"CODESYS Symulacja"

Cel ćwiczenia

Celem ćwiczenia jest zapoznanie studentów z podstawami programowania sterowników PLC w środowisku CODESYS. Studenci nauczą się tworzyć nowy projekt, dodawać elementy sterujące, programować w języku ST oraz tworzyć prostą wizualizację.

1. Tworzenie nowego projektu

- 1. Uruchamiamy program CODESYS.
- 2. Klikamy "New Project" (Nowy projekt).
- 3. Wybieramy "Standard project".
- 4. Ustawiamy nazwę projektu oraz wybieramy lokalizację do zapisania pliku.
- 5. W oknie wyboru:
 - Wybieramy target: Berghof MX6 Control.

Standard Project					
	You are about to create a new standard project. This wizard will create the following objects within this project: - One programmable device as specified below - A program PLC_PRG in the language specified below - A cyclic task which calls PLC_PRG - A reference to the newest version of the Standard library currently installed.				
	Device PLC_PRG in	Berghof MX6 Control (Berghof Automation GmbH) Berghof MX6 Control (Berghof Automation GmbH) Berghof MX6 SoftWotion Control (Berghof Automation GmbH) CODESYS Control RTE V3 (3S - Smart Software Solutions GmbH) CODESYS Control Win V3 (3S - Smart Software Solutions GmbH) CODESYS Control Win V3 x64 (3S - Smart Software Solutions GmbH) CODESYS Control Win V3 x64 (3S - Smart Software Solutions GmbH) CODESYS Control Win V3 x64 (3S - Smart Software Solutions GmbH) CODESYS SoftMotion RTE V3 (3S - Smart Software Solutions GmbH) CODESYS SoftMotion RTE V3 (3S - Smart Software Solutions GmbH) CODESYS SoftMotion RTE V3 (3S - Smart Software Solutions GmbH) CODESYS SoftMotion Win V3 x64 (3S - Smart Software Solutions GmbH) CODESYS SoftMotion Win V3 x64 (3S - Smart Software Solutions GmbH)	~		

• Wybieramy język programowania: SFC (Sequential Function Chart).

'ou are abou' bjects withir One progran A program P A cyclic task A reference f	t to create a new standard project. This wizard will create the following n this project: nmable device as specified below ² LC_PRG in the language specified below c which calls PLC_PRG to the newest version of the Standard library currently installed.	
One program A program P A cyclic task A reference t	nmable device as specified below LC_PRG in the language specified below c which calls PLC_PRG to the newest version of the Standard library currently installed.	
)evice	Berghof MX6 Control (Berghof Automation GmbH)	\sim
LC_PRG in	Sequential Function Chart (SFC)	\sim
	Continuous Function Chart (CFC) Continuous Function Chart (CFC) - page-oriented Function Block Diagram (FBD) Ladder Diagram (LD) Ladder Logic Diagram (LD) Sequential Function Chart (SFC)	
		Continuous Function Chart (CFC) Continuous Function Chart (CFC) - page-oriented Function Block Diagram (FBD) Ladder Diagram (LD) Ladder Logic Diagram (LD) Sequential Function Chart (SFC) Structured Text (ST)

6. Po dokonaniu wyboru klikamy "OK", co spowoduje utworzenie nowego projektu.

Uwaga: Jeśli pracujemy na rzeczywistym sterowniku PLC, należy kliknąć **"Device"**, wybrać **Gateway** oraz wyszukać sterownik, klikając **"Scan Network"**.

munication Settings	Scan Network	Sateway 👻 Device	• •				
ations		_			_	_	
up and Restore							
				•		•	
		Ga	Gai teway-1	:eway	DESKTOP-8R06H	56	~
Settings		IP-/	Address:		Press ENTER to s	et active path	
C Shell		Por	ainost t:				
ers and Groups		121	.7				
ass Rights							
nbol Rights							
ensed Software Metrics							
Objects							
sk Deployment							
tus							
ormation							
Select Device						;	<
Select the Network Path	to the Controller						
💑 🖕 Gateway-1				Device Name: Gateway-1		Scan Network	
				Driver: TCP/IP		Wink	4
				IP-Address:			
				Port:			
				1217			
				,			_
Hide non-matching o	devices, filter by T	arget ID			ОК	Cancel]

Jeśli korzystamy z symulacji, pomijamy ten krok i aktywujemy tryb symulacji: klikamy **"Online**", a następnie zaznaczamy **"Simulation**".



2. Dodawanie elementów sterowania w programie

1. Po prawej stronie znajduje się panel **"Toolbox"**, skąd wybieramy elementy do programu.



2. Dodajemy kolejne kroki do programu. Przykładowo tworząc graf jak na rysuku poniżej.



3. Po kliknięciu na dodany krok, wybieramy opcję "Copy implementation", a następnie wybieramy język ST.

CODESYS	×					
Please select the duplication mode for step actions.						
Copy reference: A new step calls the same actions						
Copy implementation: New action objects are created for a new step						
Set as default. This setting can be changed in SFC editor options						
OK Cancel						

mplementation language
Structured Text (ST)
Continuous Function Chart (CFC) Continuous Function Chart (CFC) - page-oriented Function Block Diagram (FBD) Ladder Diagram (LD) Ladder Logic Diagram (LD) Sequential Function Chart (SFC)

4. Powtarzamy proces dla kolejnych kroków.

3. Tworzenie wizualizacji

- 1. W drzewie projektu znajdujemy "Application".
- 2. Klikamy na nią prawym przyciskiem myszy i wybieramy "Add Object", a następnie "Visualization".



3. W nowo otwartym oknie zaznaczamy "VisuSymbols" i klikamy OK – dodaje to obiekt wizualizacji do projektu.

Add Visualization			×		
Creates a visualization ob	ject				
Name:					
Visualization					
Symbol libraries	Active				
~ 🕘 VisuSymbols (System)					
A visualization symbol library is a CODESYS library with graphics and graphical objects. If the visualization symbol library is assigned the library is added into the POUs library manager. The graphics and graphical objects are shown in the toolbox when a visualization editor is the active editor.					
I	Add	Cancel			

4. Efektem powinno być otwarcie okna wizualizacji.



5. Po prawej stronie pojawi się **"Visualization Toolbox"**, z którego wybieramy potrzebne elementy wizualizacji.



- 6. Aby dodać element do wizualizacji, przeciągamy go do okna projektu za pomocą myszy.
- 7. Po dodaniu elementu klikamy na niego, a po prawej stronie pojawi się okno parametrów, w którym możemy zmieniać właściwości (np. kolor, rozmiar).

Properties	- ₽ X
🍸 Filter 🔹 💕	Sort by 👻
₿↓ Sort order 🝷	✓ Advanced
Property	Value ^
x	432
Y	302
Width	137
Height	74
Angle	0
Center	
x	500
Y	339
Colors	
Normal	
⊟ F	Bas
T	255
□ Fi	XY
T	255
Alarm	
Use gradi	
Gradient s	linear,
± Appearance	
± Texts	
± Text prope	
Absolute m	
= Movem	
X	
Y	
Rotation	~

Grupowanie elementów

Jeśli chcemy, aby kilka elementów było traktowanych jako jedna całość:

- 1. Zaznaczamy wszystkie elementy.
- 2. Klikamy opcję "Group".



Animacja ruchu elementów

Aby elementy poruszały się w wizualizacji:

- 1. Przypisujemy do nich odpowiednie zmienne, np. Movement, X.
- 2. Wprowadzamy odpowiednie wartości sterujące ruchem.



Obsługa lampek kontrolnych

Aby elementy poruszały się w wizualizacji:

1. Do lampek należy przypisać odpowiednie zmienne.

	Property	Value			
	Element name	GenElemInst_54			
	Type of element	Lamp			
	Position				
	X	1192			
	Y	194			
	Width	42			
	Height	49			
	Variable	PLC_PRG.Czujnik2L			
	Image settings				
• •	± Texts				
	 State variables 				
	😑 Center				
	X	1213			
	Y	218			
	 Absolute movement 				
	Animation duration	0			
	Bring to foreground				
	Background				

4. Programowanie w języku ST

1. W oknie edytora kodu deklarujemy zmienne sterujące.

```
WagoniklVel: INT; // Prędkość wagonika 1
Wagonik2Vel: INT; // Prędkość wagonika 2
Wagonik3Vel: INT; // Prędkość wagonika 3
Czujnik1L, Czujnik1P: BOOL; // Czujniki krańcowe wagonika 1 (L - lewa, P - prawa)
Czujnik2L, Czujnik2P: BOOL; // Czujniki krańcowe wagonika 2
Czujnik3L, Czujnik3P: BOOL; // Czujniki krańcowe wagonika 3
position1, position2, position3 : REAL;
```

2. Należy w kodzie kroku 0 ustawić początkowe wartości zmiennych zdefiniowanych podczas budowy wizualizacji.

```
1
          position1 := 992;
     2
          position2 := 992;
          position3 := 992;
     3
4
           CzujniklP := TRUE;
           Czujnik2P := TRUE;
           Czujnik3P := TRUE;
           CzujniklL := FALSE;
      9
           Czujnik2L := FALSE;
    10
           Czujnik3L := FALSE;
```

3. Przykładowe linijki kodu w języku ST ustawiające prędkość wagonu oraz zmieniające czujniki krańcowe w zależności od położenia wagonika

```
WagoniklVel:=1000;
positionl := positionl - 0.001*WagoniklVel;
IF positionl <= 12 THEN positionl:= 12; czujniklL:= TRUE;
END_IF
IF positionl <992 THEN czujniklP := FALSE;
END_IF
IF</pre>
```

5. Testowanie symulacji

- 1. Klikamy "Online" \rightarrow "Login", aby połączyć się z symulatorem.
- 2. Uruchamiamy program za pomocą "Start".
- 3. Obserwujemy działanie programu i wizualizacji.
- 4. W razie potrzeby wprowadzamy poprawki do kodu lub elementów wizualizacji.
- 5. Po zakończeniu testów klikamy "Stop", a następnie "Logout".

Zadania

1. Zadaniem zespołu studentów jest opracowanie wizualizacji zawierającej 3 wagoniki poruszające się w prostej linii wraz z kontrolkami sygnalizującymi osiągnięcie stanów skrajnych położenia wagoników. Przykładowy wygląd wizualizacji:



2. Należy utworzyć sterowanie, które z różnymi prędkościami wystartuje wagoniki z lewej strony do prawej, a następnie każdy z nich niezależnie po osiągnięciu skrajnej pozycji po prawej stronie wróci z tą samą prędkością na stronę lewą, gdzie poczeka aż wszystkie wagoniki znajdą się po lewej stronie. Wtedy program wykona się ponownie.