Міністерство освіти і науки України

Державний вищий навчальний заклад

“Національний гірничий університет”

Кафедра електропривода

Авторизований навчальний центр “Schneider Electric”

Розробив: Яланський О.А., доцент кафедри електропривода

м. Дніпропетровськ

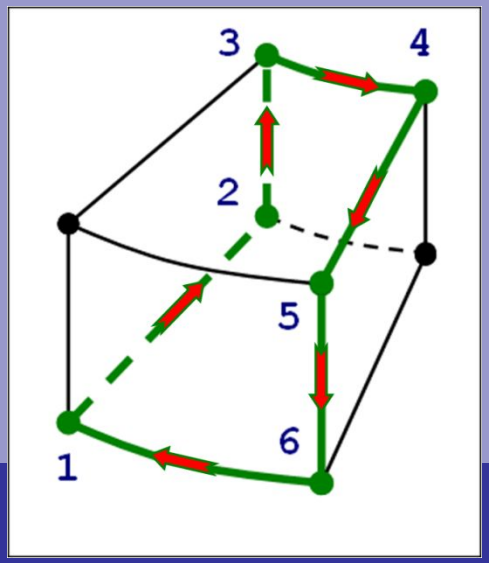
2012 - 2013

Цикловое управление манипулятором МП-9с с помощью ТМ-238

Цель работы: Рассмотреть различные подходы к решению одной типовой задачи циклового управления с использованием разных языков программирования и различных типов входных/выходных данных.

Цикловое управление манипулятором МП-9с с помощью ТМ-238

- Обеспечить движение схвата манипулятора по замкнутой траектории;
- Начальная установка руки и схвата выполняется вручную.



Глобальные переменные

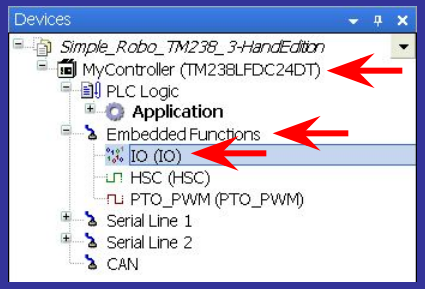
- **i_** – входные переменные (магнитоуправляемые грекноны манипулятора);
- **q_** – выходные переменные (электроклапаны манипулятора);
- **x** – логическая переменная (**BOOL**);
- **w** – переменная типа «Слово» (**WORD**).

```
1  VAR_GLOBAL
2      i_xDown, i_xUp, i_xLeft, i_xRight, i_xFrwrd, i_xBack: BOOL;
3      q_xDown, q_xUp, q_xLeft, q_xRight, q_xFrwrd, q_xBack: BOOL;
4      q_xPresure, q_xTake: BOOL;
5      i_wWord, q_wWord: INT;
6  END_VAR
```

Базовые варианты реализации

- **Используются глобальные переменные входов/выходов типа BOOL;**
- Локальные вспомогательные переменные не требуются;
- Определив текущее положение схвата, можно задать движение по одной степени свободы к следующей точке траектории.

Карта входов/выходов



I/O Configuration | I/O Mapping

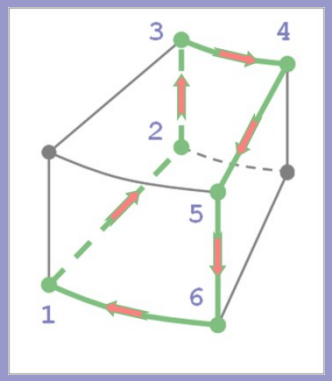
Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
Inputs							
		IW0	%IW0	WORD			
		I0	%IX0.0	BOOL			Fast Input
		I1	%IX0.1	BOOL			Fast Input
		I2	%IX0.2	BOOL			Fast Input
		I3	%IX0.3	BOOL			Fast Input
		I4	%IX0.4	BOOL			Fast Input
		I5	%IX0.5	BOOL			Fast Input
		I6	%IX0.6	BOOL			Fast Input
		I7	%IX0.7	BOOL			Fast Input
Application.GVL.i_xLeft		I8	~~%IX1.0~~	BOOL			
Application.GVL.i_xRight		I9	~~%IX1.1~~	BOOL			
Application.GVL.i_xFrwrd		I10	~~%IX1.2~~	BOOL			
Application.GVL.i_xBack		I11	~~%IX1.3~~	BOOL			
Application.GVL.i_xUp		I12	~~%IX1.4~~	BOOL			
Application.GVL.i_xDown		I13	~~%IX1.5~~	BOOL			
Outputs							
		QW0	%QW0	WORD			
		Q0	%QX0.0	BOOL			Fast Output
		Q1	%QX0.1	BOOL			Fast Output
Application.GVL.q_xPresure		Q2	~~%QX0.2~~	BOOL			Fast Output
Application.GVL.q_xTake		Q3	~~%QX0.3~~	BOOL			Fast Output
Application.GVL.q_xLeft		Q4	~~%QX0.4~~	BOOL			
Application.GVL.q_xRight		Q5	~~%QX0.5~~	BOOL			
Application.GVL.q_xFrwrd		Q6	~~%QX0.6~~	BOOL			
Application.GVL.q_xBack		Q7	~~%QX0.7~~	BOOL			
Application.GVL.q_xUp		Q8	~~%QX1.0~~	BOOL			
Application.GVL.q_xDown		Q9	~~%QX1.1~~	BOOL			

Reset mapping | Always update variables

= Create new variable = Map to existing variable

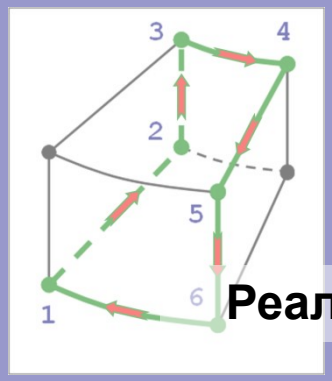
```
PROGRAM POU_Robo_Control_LD
VAR
END_VAR
```



Реализация на языке лестничных диаграмм

LD

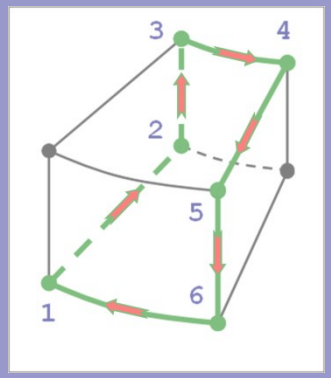
```
PROGRAM POU_Robo_Control_FBD
VAR
END_VAR
```



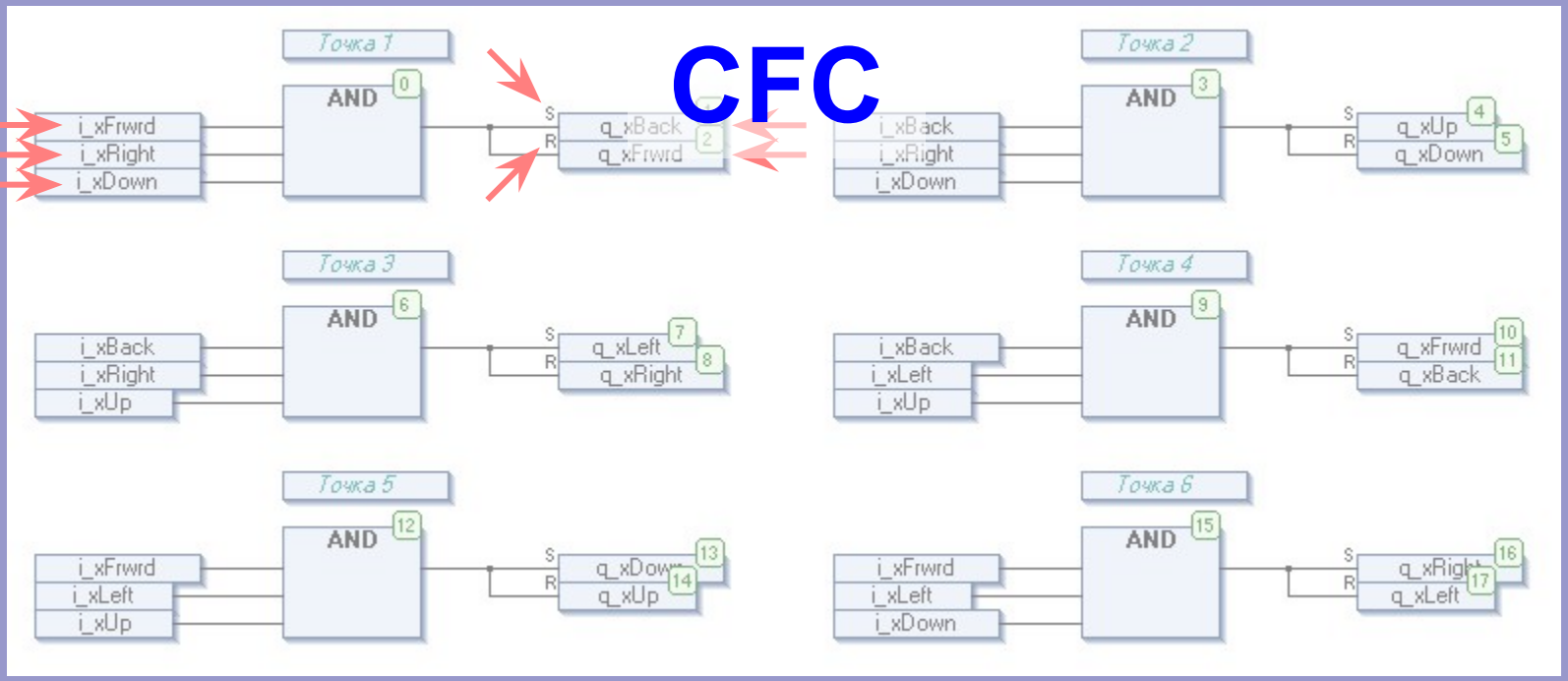
Реализация на языке диаграмм функциональных блоков

FBD

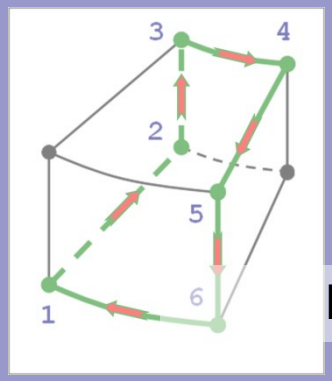
```
1   PROGRAM POU_Robo_Control_CFC
2   VAR
3   END_VAR
```



Реализация на языке непрерывных функциональных схем



```
PROGRAM POU_Robo_Control_ST
VAR
END_VAR
```

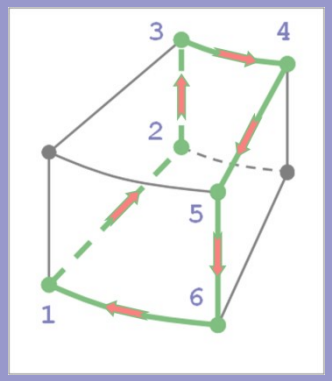


Реализация на языке структурированного текста

ST

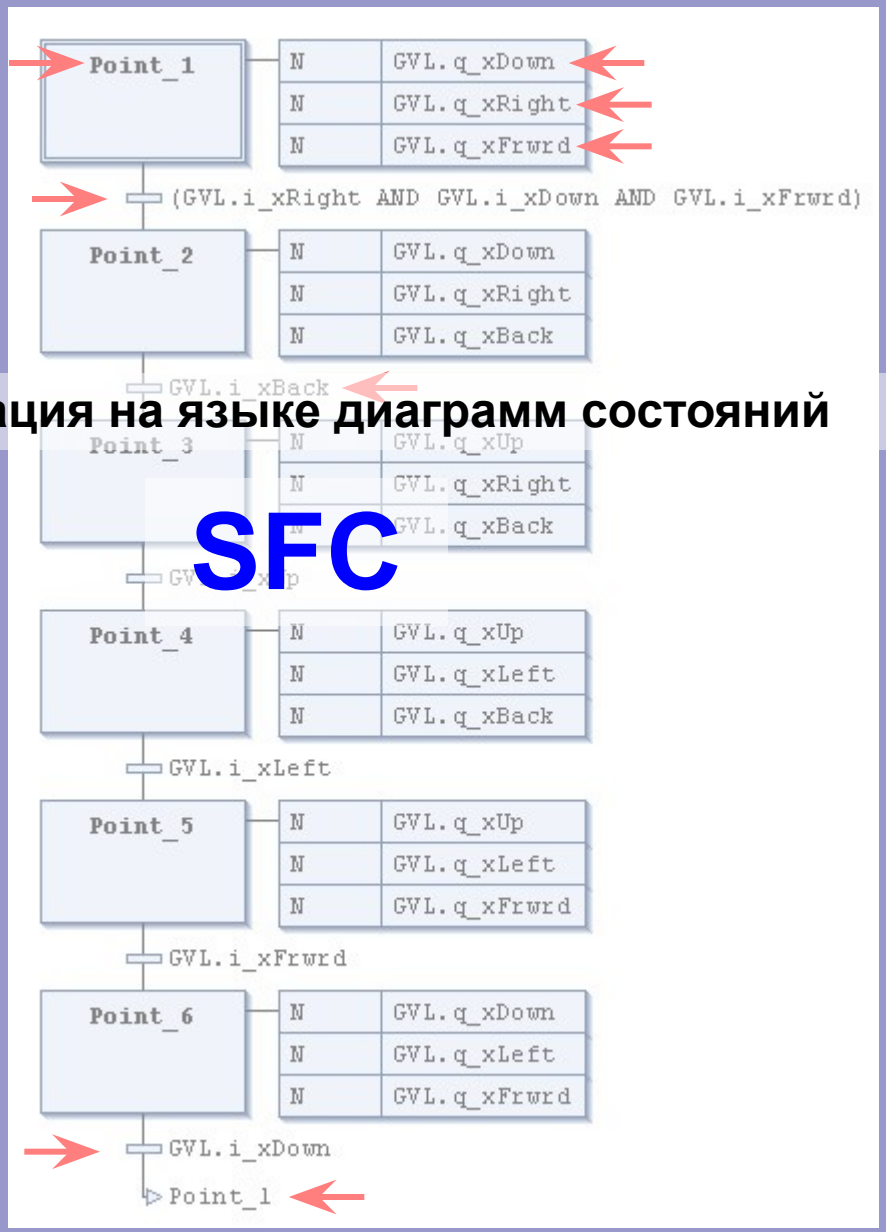
```
// Точка 1:
IF GVL.i_xFrwrd AND GVL.i_xRight AND GVL.i_xDown THEN
    GVL.q_xBack := TRUE;        //Включить Назад
    GVL.q_xFrwrd := FALSE;      //Выключить Вперед

//Точка 2:
ELSIF GVL.i_xBack AND GVL.i_xRight AND GVL.i_xDown THEN
    GVL.q_xUp := TRUE;          //Движение Вверх
    GVL.q_xDown := FALSE;

//Точка 3:
ELSIF GVL.i_xBack AND GVL.i_xRight AND GVL.i_xUp THEN
    GVL.q_xLeft := TRUE;        //Движение Влево
    GVL.q_xRight := FALSE;

//Точка 4:
ELSIF GVL.i_xBack AND GVL.i_xLeft AND GVL.i_xUp THEN
    GVL.q_xFrwrd := TRUE;       //Движение Вперед
    GVL.q_xBack := FALSE;

//Точка 5:
ELSIF GVL.i_xFrwrd AND GVL.i_xLeft AND GVL.i_xUp THEN
    GVL.q_xDown := TRUE;        //Движение Вниз
    GVL.q_xUp := FALSE;

//Точка 6:
ELSIF GVL.i_xFrwrd AND GVL.i_xLeft AND GVL.i_xDown THEN
    GVL.q_xRight := TRUE;       //Движение Вправо
    GVL.q_xLeft := FALSE;

END_IF;
```

```
PROGRAM POU_Robo_Control_SFC
VAR
END_VAR
```



Реализация на языке диаграмм состояний

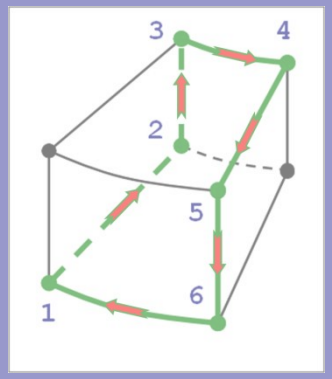
SFC

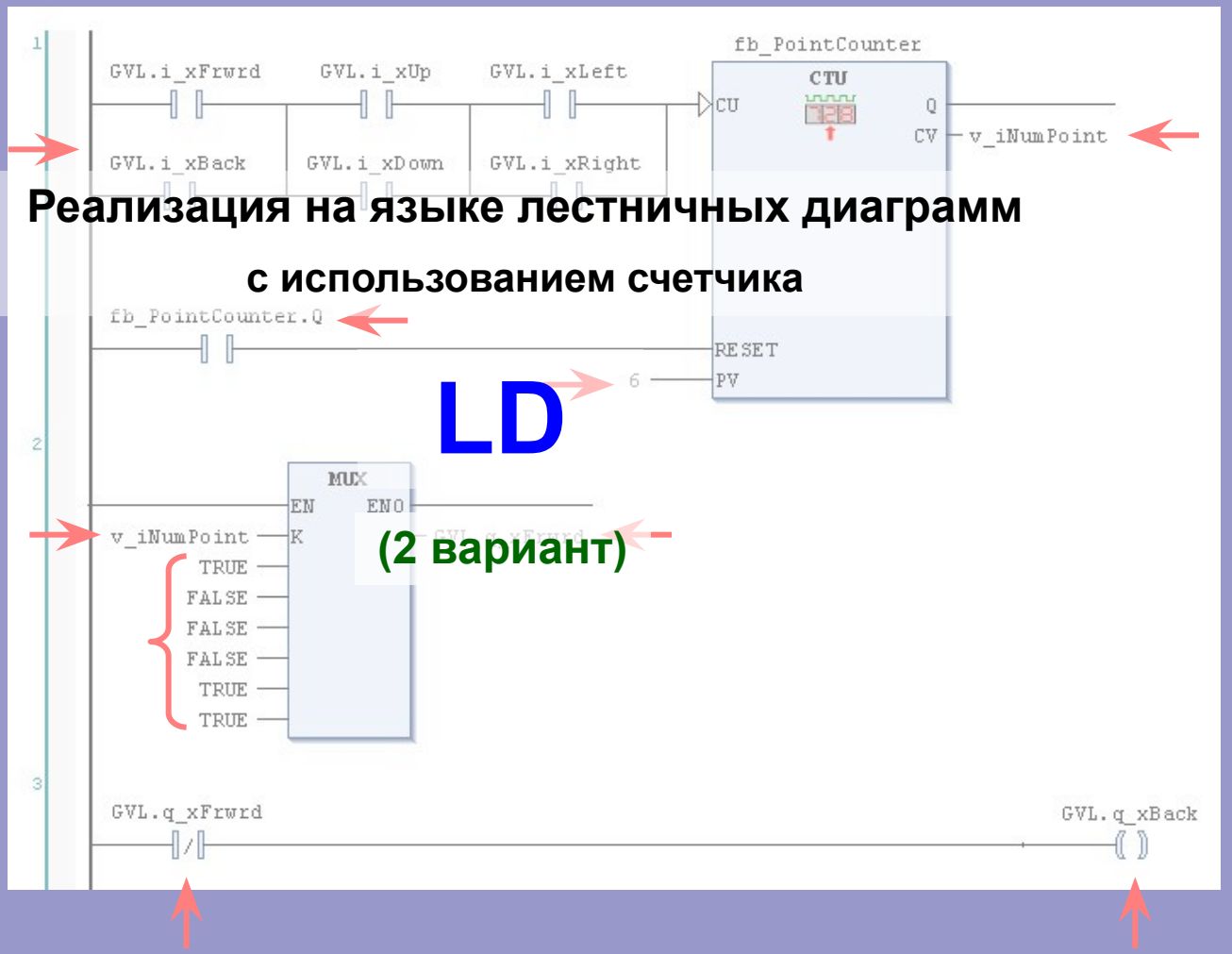


Дополнительные варианты реализации

- Возможно использование вспомогательных локальных переменных и функциональных блоков;
- При попадании в точку траектории увеличиваем значение счетчика номера текущей точки;
- В соответствии с номером текущей точки задаем движение к следующей точке траектории.

```
PROGRAM POU_Robo_Control_LD_2
VAR
    v_iNumPoint: INT;
    fb_PointCounter: CTU;
END_VAR
```





Реализация на языке лестничных диаграмм

с использованием счетчика

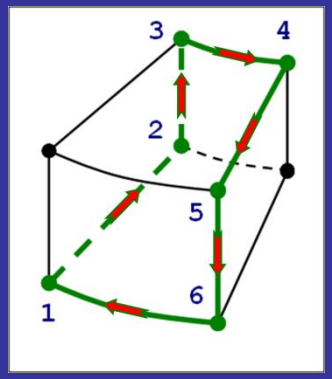
LD

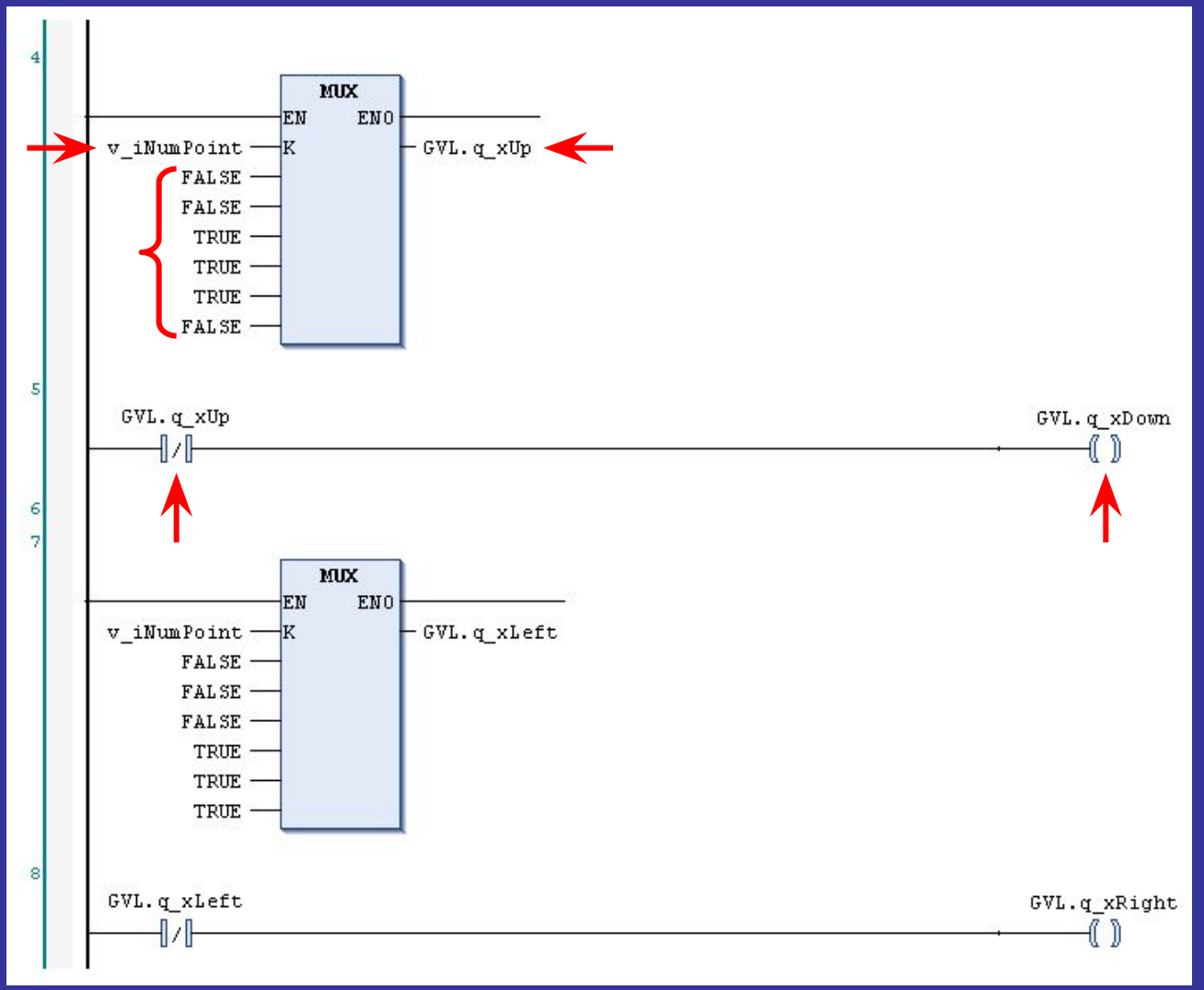
(2 вариант)

См. далее…

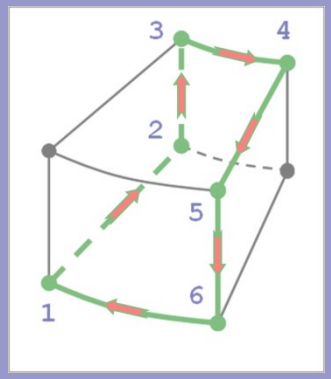
```
1  PROGRAM POU_Robo_Control_LD_2
2  VAR
3      v_iNumPoint: INT;
4      fb_PointCounter: CTU;
5  END_VAR
```

...Окончание



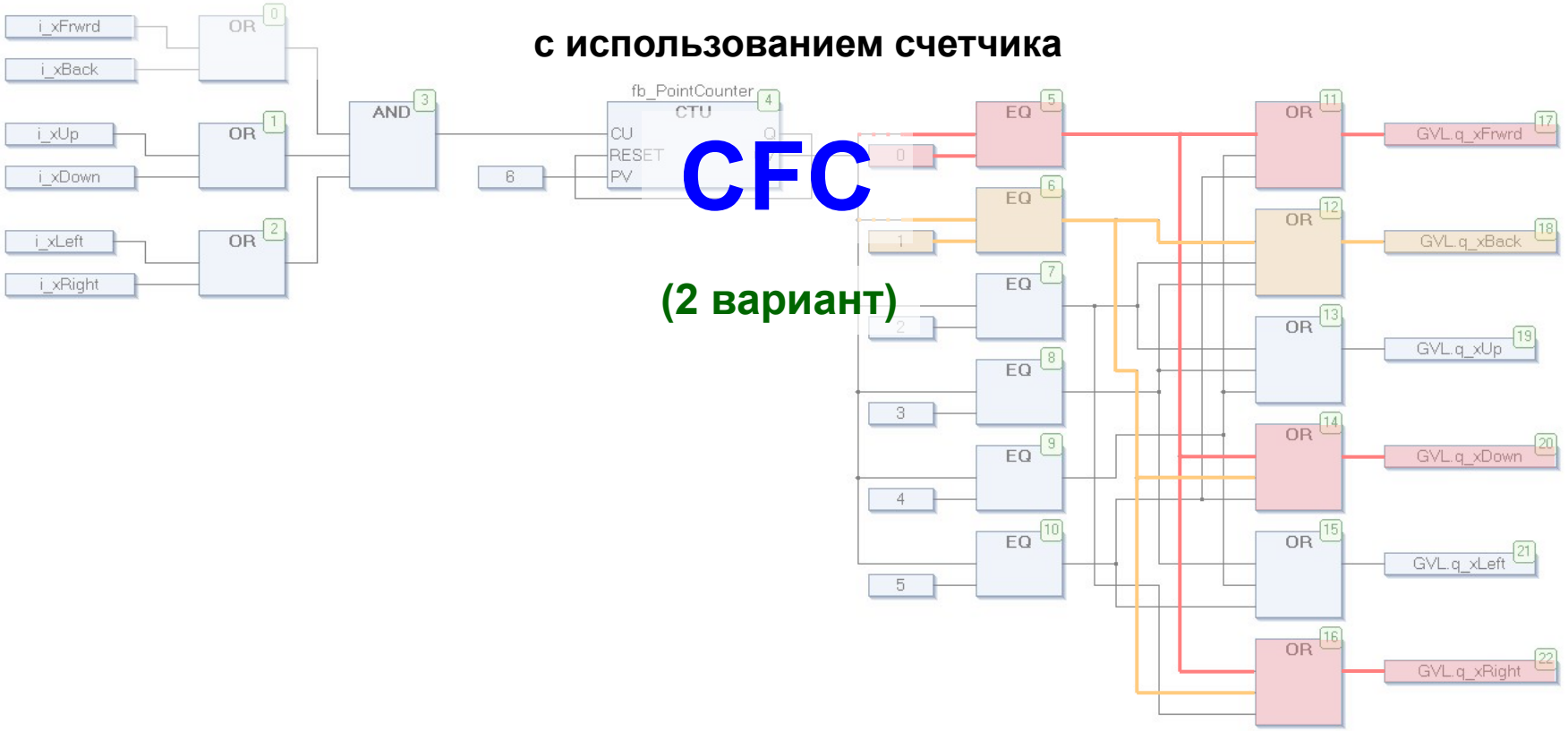


```
1  PROGRAM POU_Robo_Control_CFC_2
2  VAR
3      fb_PointCounter: CTU;
4  END_VAR
```



Реализация на языке непрерывных функциональных схем

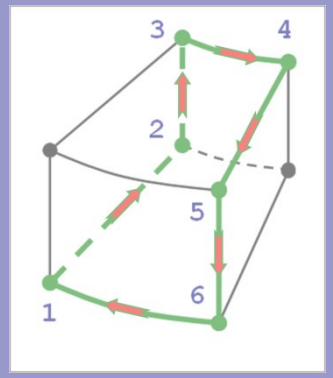
с использованием счетчика



CFC

(2 вариант)

```
PROGRAM POU_Robo_Control_ST_2
VAR
    v_xVar1, v_xVar2: BOOL;
    v_xMem: BOOL := FALSE;
    v_iNumPoint : INT;
END_VAR
```



Реализация на языке структурированного текста

с использованием программного счетчика

ST

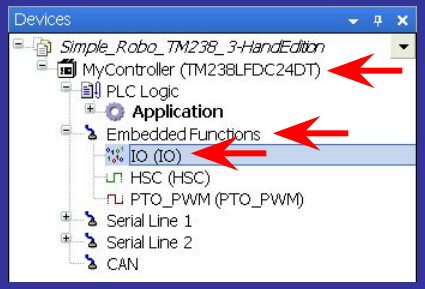
(2 вариант)

```
//Находится ли манипулятор в произвольной крайней точке?:
v_xVar1 := (GVL.i_xFrwrd OR GVL.i_xBack) AND (GVL.i_xRight OR GVL.i_xLeft) AND (GVL.i_xUp OR GVL.i_xDown);
//Детектор фронта сигнала v_xVar1:
v_xVar2 := v_xVar1 AND NOT v_xMem;
v_xMem := v_xVar1;
//Вычисление номера точки:
IF v_xVar2 THEN
    v_iNumPoint := v_iNumPoint + 1;
        IF v_iNumPoint > 6 THEN v_iNumPoint := 1; END_IF
END_IF;
//Установка манипулятора в требуемую точку, определяемую номером текущей точки:
CASE v_iNumPoint OF
1:  GVL.q_xBack := TRUE; GVL.q_xFrwrd := FALSE; GVL.q_xDown := TRUE; GVL.q_xUp := FALSE; GVL.q_xRight := TRUE; GVL.q_xLeft := FALSE;
2:  GVL.q_xBack := TRUE; GVL.q_xFrwrd := FALSE; GVL.q_xDown := FALSE; GVL.q_xUp := TRUE; GVL.q_xRight := TRUE; GVL.q_xLeft := FALSE;
3:  GVL.q_xBack := TRUE; GVL.q_xFrwrd := FALSE; GVL.q_xDown := FALSE; GVL.q_xUp := TRUE; GVL.q_xRight := FALSE; GVL.q_xLeft := TRUE;
4:  GVL.q_xBack := FALSE; GVL.q_xFrwrd := TRUE; GVL.q_xDown := FALSE; GVL.q_xUp := TRUE; GVL.q_xRight := FALSE; GVL.q_xLeft := TRUE;
5:  GVL.q_xBack := FALSE; GVL.q_xFrwrd := TRUE; GVL.q_xDown := TRUE; GVL.q_xUp := FALSE; GVL.q_xRight := FALSE; GVL.q_xLeft := TRUE;
6:  GVL.q_xBack := FALSE; GVL.q_xFrwrd := TRUE; GVL.q_xDown := TRUE; GVL.q_xUp := FALSE; GVL.q_xRight := TRUE; GVL.q_xLeft := FALSE;
END_CASE;
```

Дополнительные варианты реализации

- **Используются глобальные переменные входов/выходов типа WORD;**
- Возможно использование вспомогательных локальных переменных и функциональных блоков;
- При попадании в точку траектории увеличиваем значение счетчика номера текущей точки;
- В соответствии с номером текущей точки задаем движение к следующей точке траектории.

Карта входов/выходов



I/O Configuration | I/O Mapping

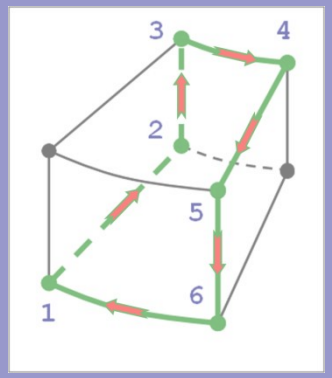
Channels

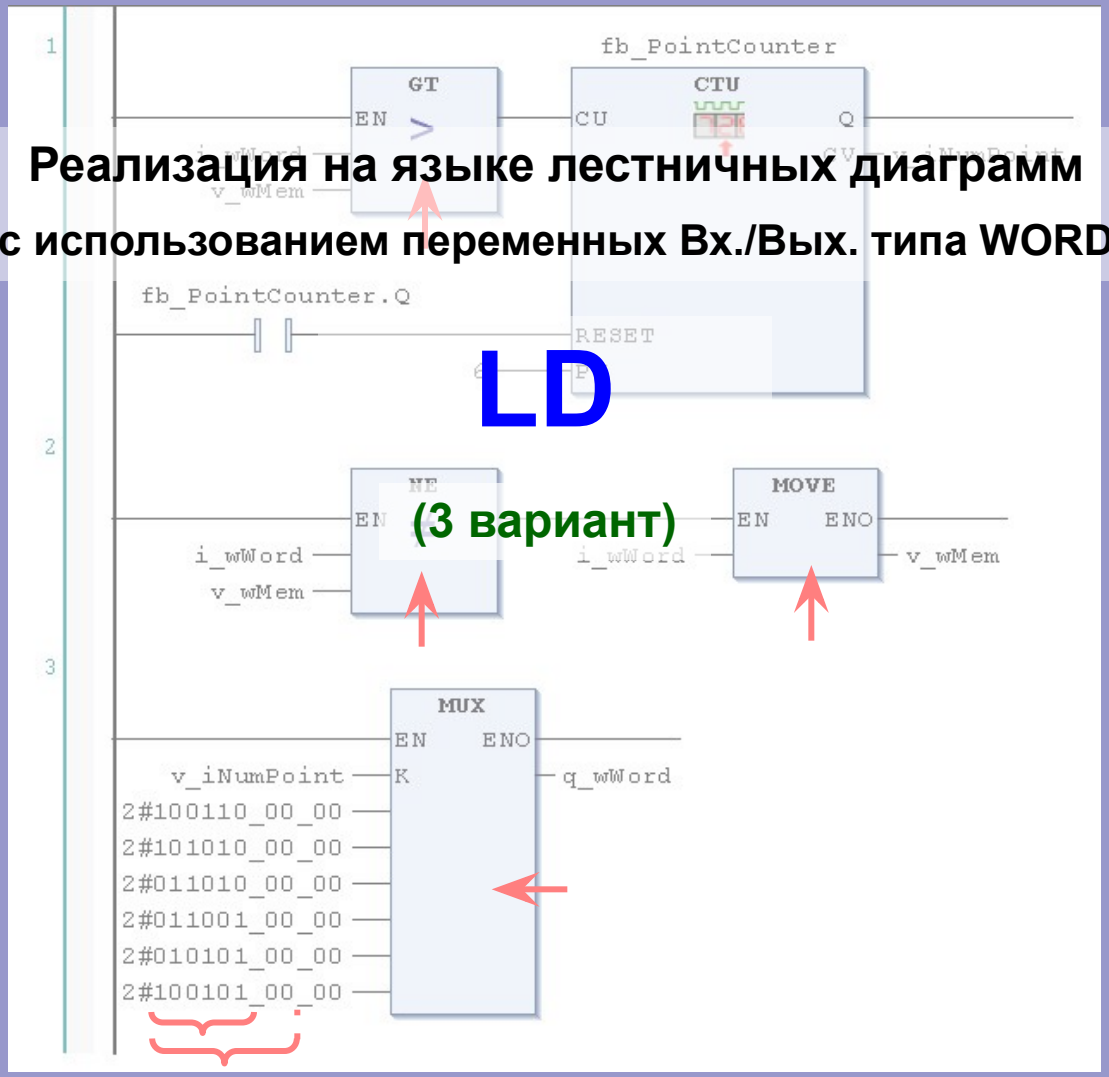
Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
Inputs							
Application.GVL.i_wWord		IW0	%IW0	WORD			
		I0	%IX0.0	BOOL			Fast Input
		I1	%IX0.1	BOOL			Fast Input
		I2	%IX0.2	BOOL			Fast Input
		I3	%IX0.3	BOOL			Fast Input
		I4	%IX0.4	BOOL			Fast Input
		I5	%IX0.5	BOOL			Fast Input
		I6	%IX0.6	BOOL			Fast Input
		I7	%IX0.7	BOOL			Fast Input
Application.GVL.i_xLeft		I8	%IX1.0	BOOL			
Application.GVL.i_xRight		I9	%IX1.1	BOOL			
Application.GVL.i_xFrwrd		I10	%IX1.2	BOOL			
Application.GVL.i_xBack		I11	%IX1.3	BOOL			
Application.GVL.i_xUp		I12	%IX1.4	BOOL			
Application.GVL.i_xDown		I13	%IX1.5	BOOL			
Outputs							
Application.GVL.q_wWord		QW0	%QW0	WORD			
		Q0	%QX0.0	BOOL			Fast Output
		Q1	%QX0.1	BOOL			Fast Output
		Q2	%QX0.2	BOOL			Fast Output
		Q3	%QX0.3	BOOL			Fast Output
		Q4	%QX0.4	BOOL			
		Q5	%QX0.5	BOOL			
		Q6	%QX0.6	BOOL			
		Q7	%QX0.7	BOOL			
		Q8	%QX1.0	BOOL			
		Q9	%QX1.1	BOOL			

Reset mapping | Always update variables

= Create new variable = Map to existing variable

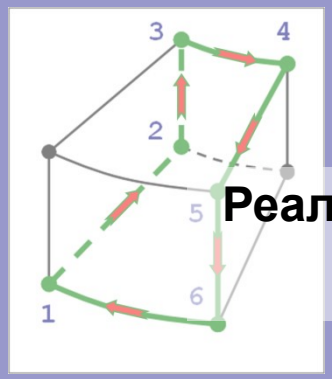
```
PROGRAM POU_Robo_Control_LD_3
VAR
    v_wMem: WORD;
    fb_PointCounter: CTU;
    v_iNumPoint: INT;
END_VAR
```





Реализация на языке лестничных диаграмм с использованием переменных Вх./Вых. типа WORD

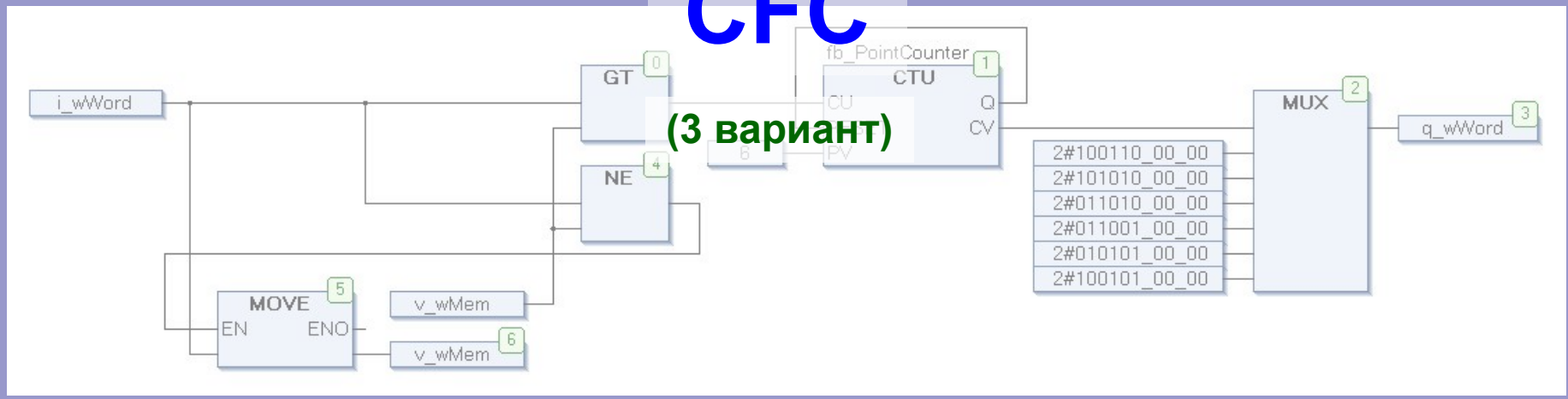
```
1  PROGRAM POU_Robo_Control_CFC_3
2  VAR
3      v_wMem: WORD;
4      fb_PointCounter: CTU;
5  END_VAR
```



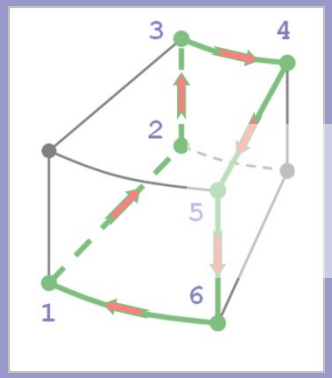
Реализация на языке непрерывных функциональных схем с использованием переменных Вх./Вых. типа WORD

CFC

(3 вариант)



```
PROGRAM POU_Robo_Control_ST_3
VAR
    v_xVar: BOOL;
    v_wMem: WORD;
    v_iNumPoint: INT;
END_VAR
```



Реализация на языке структурированного текста

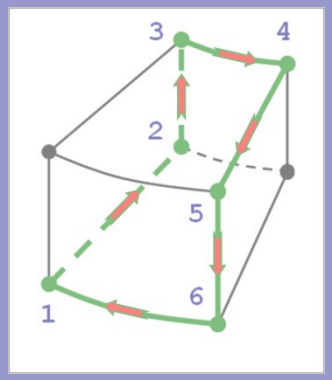
с использованием переменных Вх./Вых. типа WORD

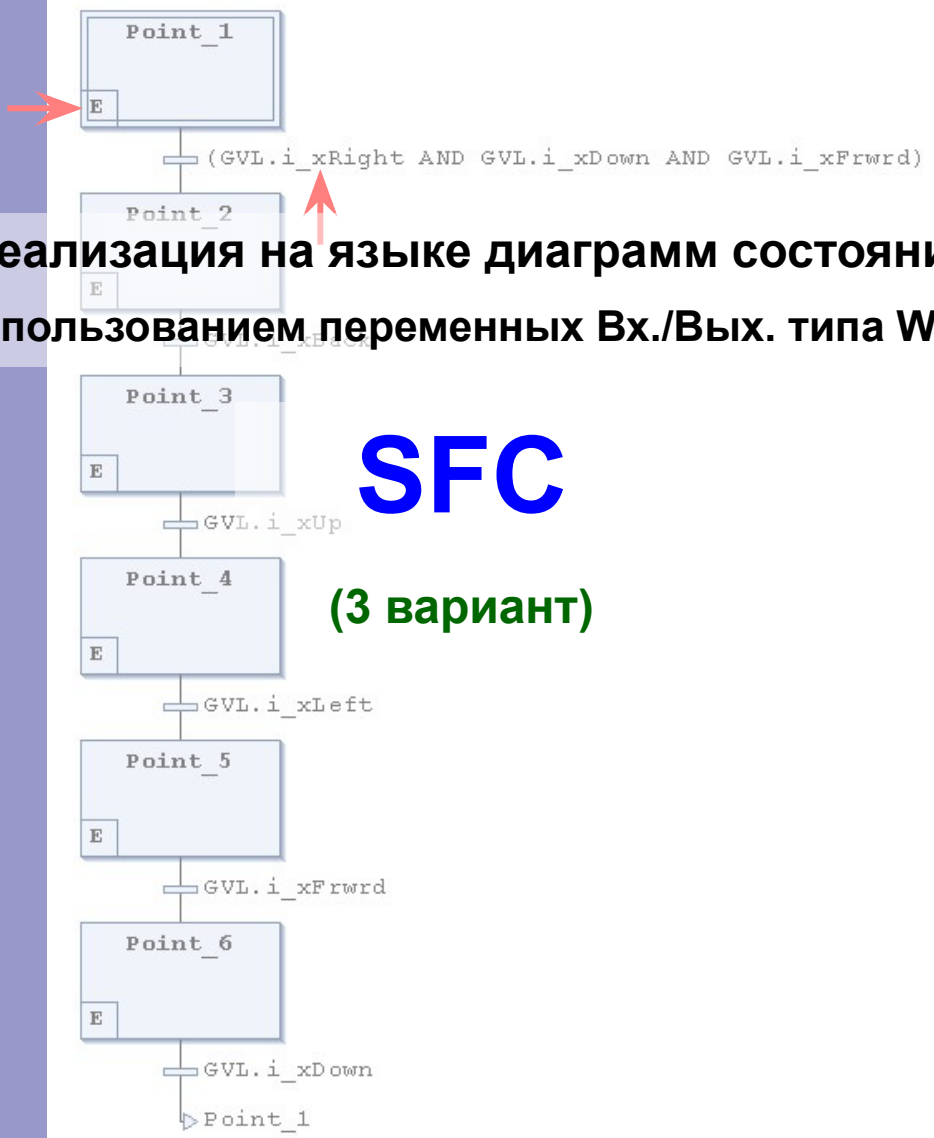
ST

(3 вариант)

```
//Детектор замыкания любого геркона (с выявлением фронта):
IF GVL.i_wWord > v_wMem THEN v_xVar := TRUE; ELSE v_xVar := FALSE; END_IF;
v_wMem := GVL.i_wWord;
//Вычисление номера точки:
IF v_xVar THEN
    v_iNumPoint := v_iNumPoint + 1;
END_IF;
//Установка манипулятора в требуемую точку, определяемую номером текущей точки:
CASE v_iNumPoint OF
//  Q9    Q8    Q7    Q6    Q5    Q4    Q3        Q2        Q1        Q0
// Down | Up | Back | Forw | Right | Left _ Take / Presure _ не исп. | не исп.
    1: GVL.q_wWord := 2#100110_00_00;
    2: GVL.q_wWord := 2#101010_00_00;
    3: GVL.q_wWord := 2#011010_00_00;
    4: GVL.q_wWord := 2#011001_00_00;
    5: GVL.q_wWord := 2#010101_00_00;
    6: GVL.q_wWord := 2#100101_00_00;
END_CASE;
// ВНИМАНИЕ! В случае необходимости использования выходов Q1 и Q0 их значения
// необходимо задавать, используя функцию OR, например:
//1: GVL.q_wWord := 2#100110_00_00 OR 2#01_00;
//2: GVL.q_wWord := 2#101010_00_00 OR 2#10_00;
//3: GVL.q_wWord := 2#011010_00_00 OR 2#01_00;
```

```
1  PROGRAM POU_Robo_Control_SFC_3
2  VAR
3  END_VAR
```





Реализация на языке диаграмм состояний

с использованием переменных Вх./Вых. типа WORD

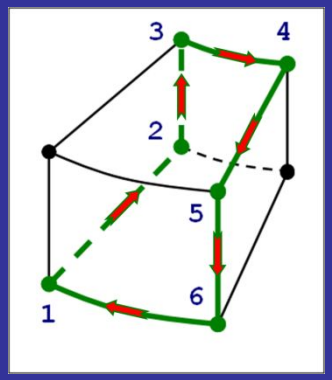
SFC

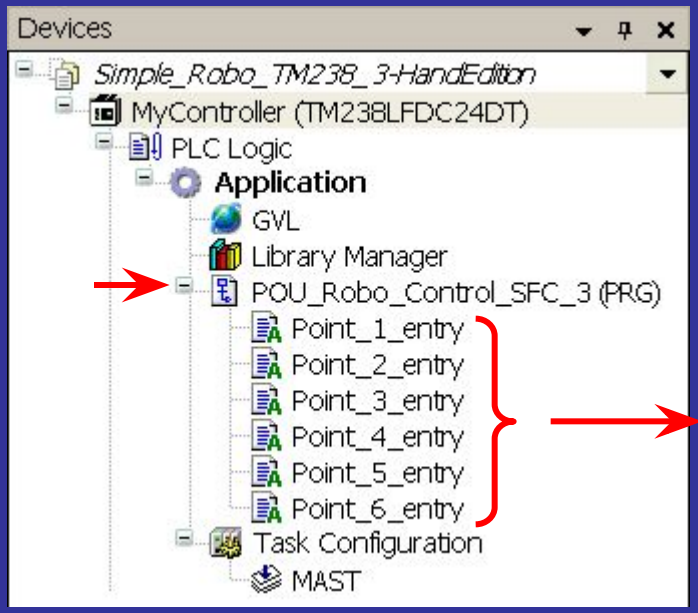
(3 вариант)

См. далее…

```
1  PROGRAM POU_Robo_Control_SFC_3
2  VAR
3  END_VAR
```

...Окончание





Point_1_entry
```
1  GVL.q_wWord := 2#100110_00_00;
```

Point_2_entry
```
1  GVL.q_wWord := 2#101010_00_00;
```

Point_3_entry
```
1  GVL.q_wWord := 2#011010_00_00;
```

Point_4_entry
```
1  GVL.q_wWord := 2#011001_00_00;
```

Point_5_entry
```
1  GVL.q_wWord := 2#010101_00_00;
```

Point_6_entry
```
1  GVL.q_wWord := 2#100101_00_00;
```

Задание

- Составить программы для обеспечения движения схвата манипулятора по заданной траектории;
- По требованию преподавателя реализовать движение схвата по разветвленной траектории;
- По требованию преподавателя реализовать возвратно-поступательное движение схвата по разомкнутой траектории.