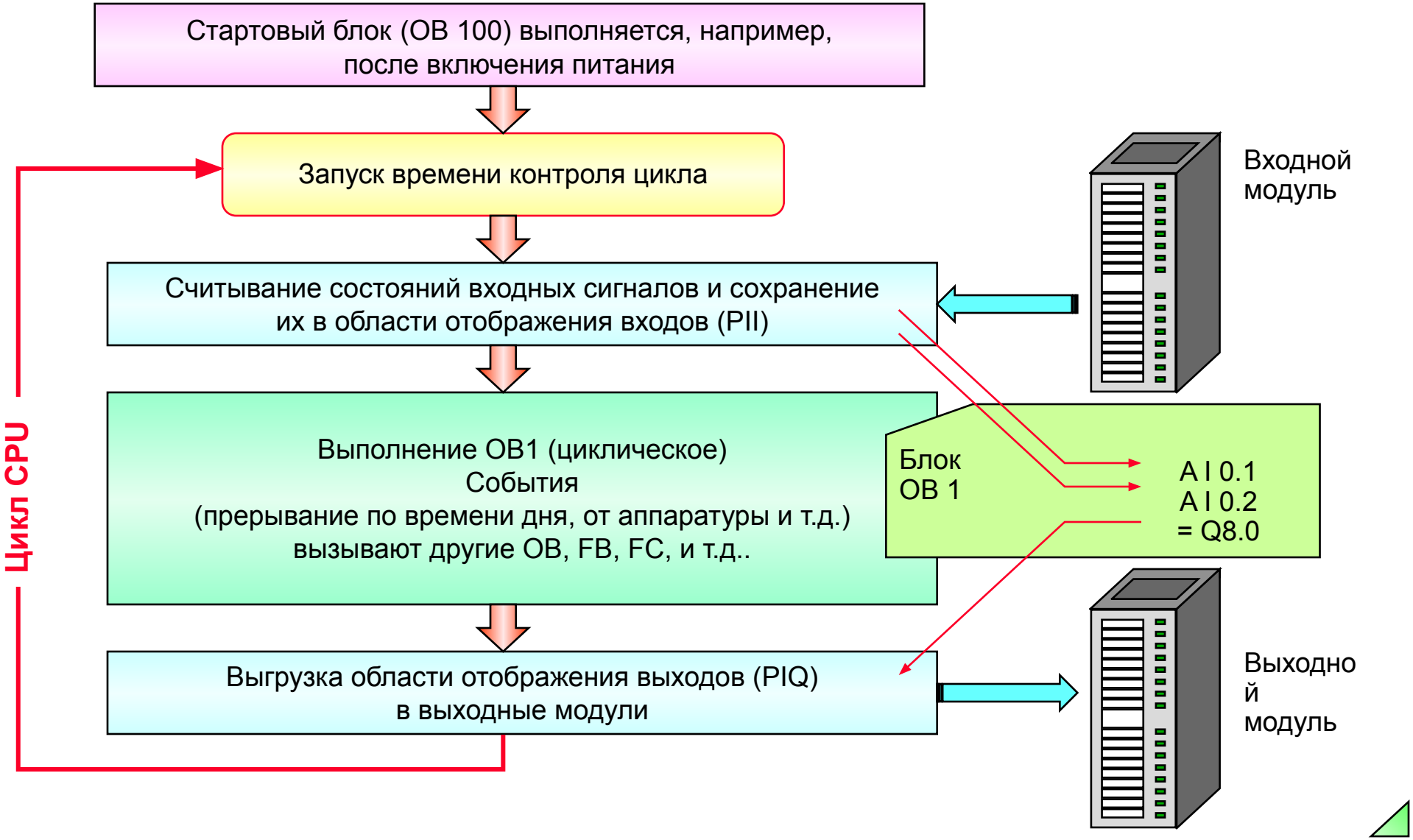
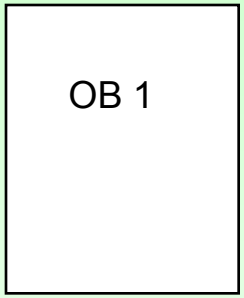
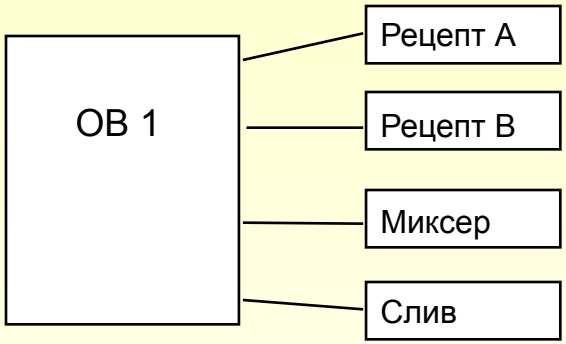
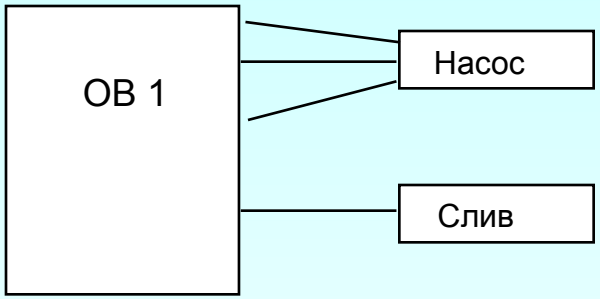
Понятие основного цикла
Переменные и типы данных
STEP 7

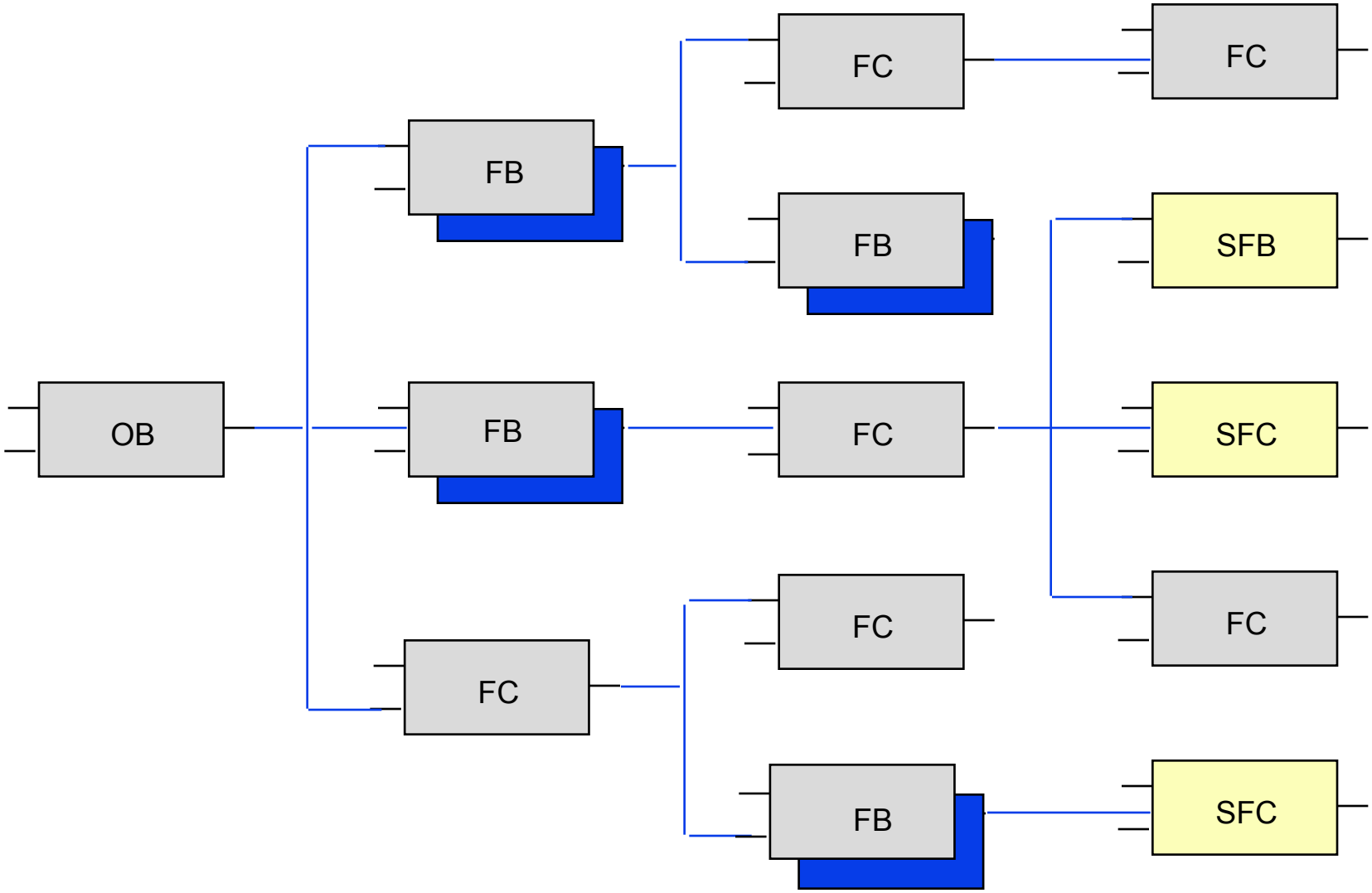
Циклическое выполнение программы



Структура программы

Линейная программа	Составная программа:	Структурированная программа
		
: Все инструкции находятся в одном блоке (обычно, в OB1)	Каждая функция программы реализуется в одном блоке. OB1 последовательно вызывает все блоки.	Универсальные функции реализуются в отдельных блоках. OB1 (или другие блоки) вызывает эти блоки и и передает им фактические данные. .

Структура программы



Значения переменных и типы данных

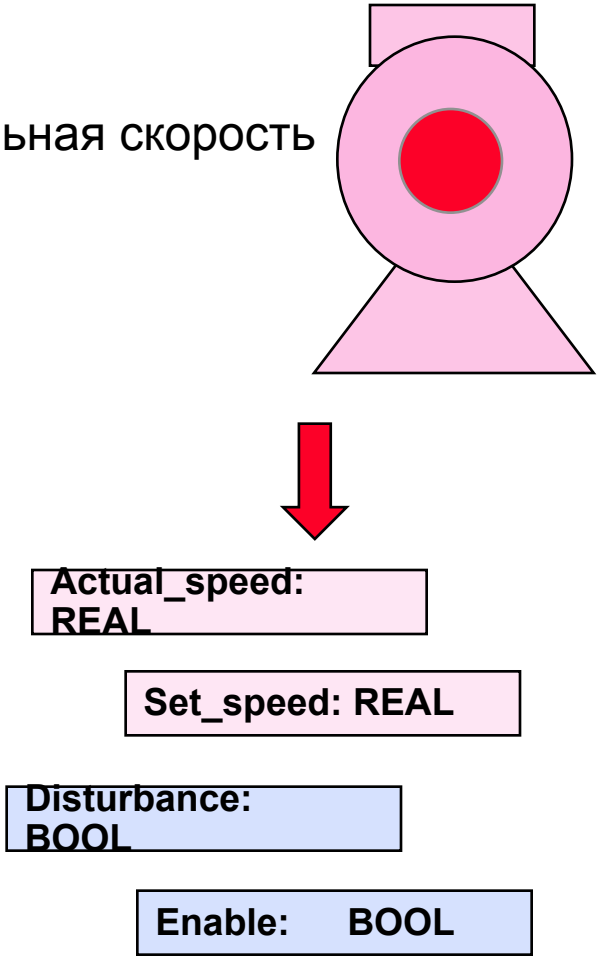
Тип данных характеризует основные свойства данных

- непрерывная область значений: напр., актуальная скорость
- "yes/no" - свойство: напр., вибрации

Тип данных устанавливает:

- доступный диапазон значений (INT: -32 368 ... +32 367, и т.д.)
- допустимые операции (арифметические инструкции: +, -, и т.д.)
- структуру объектов данного типа, т.е. расположение битов в памяти

Переменные разрешают Вам сохранять и позже продолжать обработку величин



Обзор типов данных в STEP 7

Элементарные типы данных (до 32 бит)

- **Битовые типы данных (BOOL, BYTE, WORD, DWORD, CHAR)**
- **Математические типы данных (INT, DINT, REAL)**
- **Временн*ые* типы (S5TIME, TIME, DATE, TIME_OF_DAY)**

Сложные типы данных (больше, чем 32 бита)

- **Дата и время (DATE_AND_TIME)**
- **Массив (ARRAY)**
- **Структура (STRUCT)**
- **Строка (STRING)**

Типы данных, определенные пользователем (больше, чем 32 бита)

Тип данных UDT (**U**ser **D**efined **T**ype)

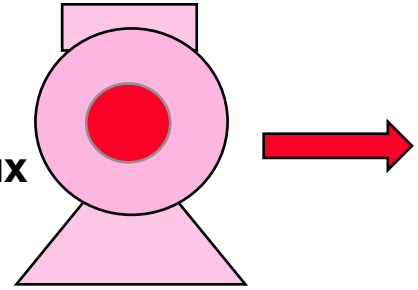
Элементарные типы данных в STEP 7

Ключевое слово	Длина (в битах)	Пример константы
BOOL BYTE WORD DWORD CHAR	1 8 16 32 8	1 or 0 B#16#A9 W#16#12AF DW#16#ADAC1EF5 'w'
S5TIME	16	S5T#5s_200ms
INT DINT REAL	16 32 32	123 65539 or L#-1 1.2 or 34.5E-12
TIME DATE TIME-OF-DAY	32 16 32	T#2D_1H_3M_45S_12MS D#1999-06-14 TOD#12:23:45.12

Важность сложных типов данных

"Лучшее" структурирование данных:

- адаптирует их к задаче
- создает “корректный" тип данных



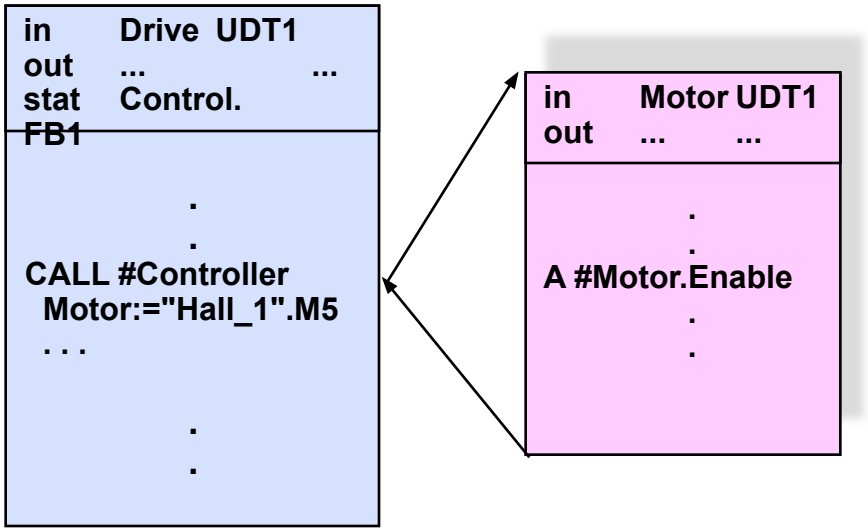
Motor: STRUCT

Set_speed:	REAL
Actual_speed: REAL	
Enable:	BOOL
Disturbance: BOOL	

END_STRUCT

Компактная форма передачи данных при вызове блока:

- “много“ элементов данных могут быть переданы в одном параметре
- делает возможным структкрированное программирование
- блоки “связываются" только через параметры
- программное обеспечение многократного пользования



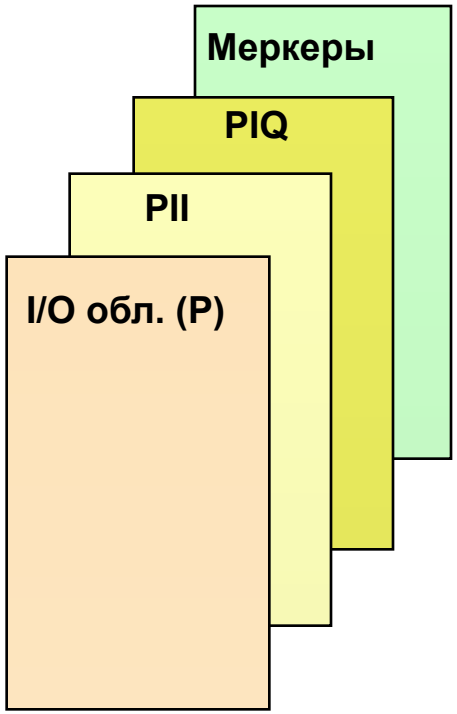
Сложные типы данных в STEP 7

Ключевое слово	Длина (в битах)	Пример	
DATE_AND_TIME (Дата и время)	**64**	**DT#99-06-14-12:14:55.0**	
STRING (Строка символов с max. 254 символами)	**8 * (число символов +2)**	**´This is a string´** **´SIEMENS´**	
ARRAY (Группа элементов одного и того же типа данных)	**Определяется пользователем**	**Meas_vals: ARRAY[1..20]** **INT**	
STRUCT (Структура, группа элементов разных типов данных)	**Определяется пользователем**	**Motor: STRUCT** **Speed : INT** **Current : REAL** **END_STRUCT**	
UDT (**U**ser **D**efined Data **T**ype = “Шаблон“, состоящий из элементарных и/или сложных типов данных)	**Определяется пользователем**	**UDT как блок**	**UDT как элемент массива**
		STRUCT **Speed : INT** **Current : REAL** **END_STRUCT**	**Drive: ARRAY[1..4]** **UDT1**

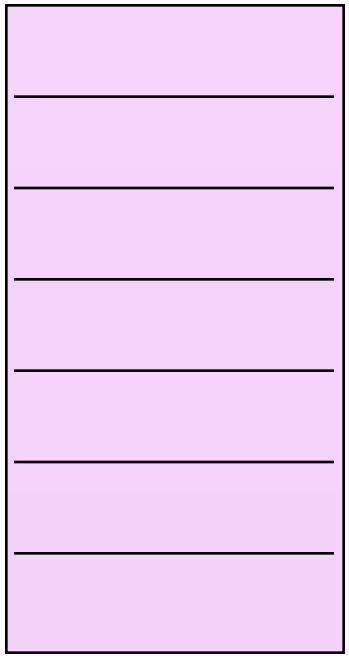
Параметрические типы в STEP 7

Ключевое слово	Длина (в битах)	Пример
TIMER	16	Contact time: TIMER . SI #Contact_time
COUNTER	16	NoCompParts: COUNTER . LC #No_Comp_Parts
BLOCK_FB BLOCK_FC BLOCK_DB BLOCK_SDB	16	Recall: BLOCK_FB . UC #Recall
Pointer	48	Measure: POINTER . L P##Measure .
ANY	80	Measured Values: ANY . L P##Meas_Values .

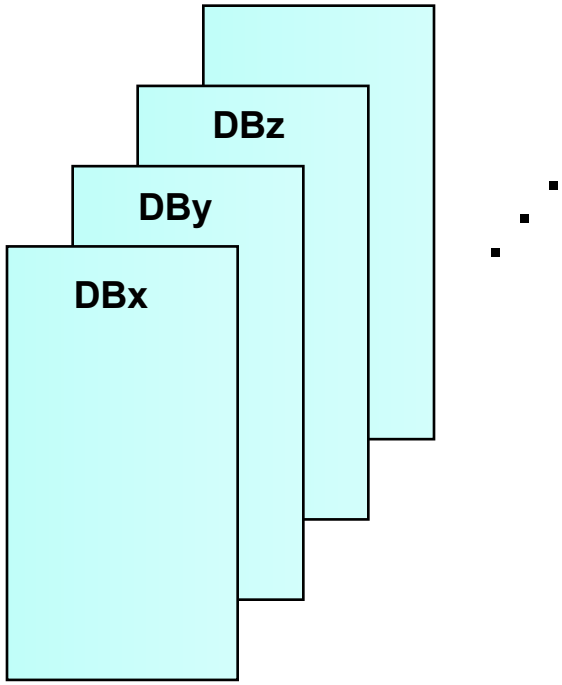
Области для хранения переменных



“Классические"
области PLC



Локальный
стек данных

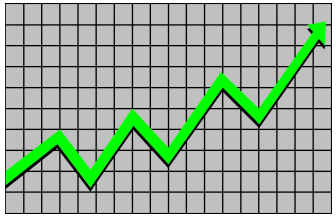


Блоки данных

Тип данных: ARRAY

ARRAY (массив):

- Группа компонентов с одинаковым типом данных
- Объявление:
  - одномерный:
    Имя_массива:ARRAY[*minИндекс*..*maxИндекс*] OF *Тип_данных;*
  - многомерный:
    Имя_массива :ARRAY[*minИндекс 1*.. *maxИндекс 1*, *minИндекс 2*.. *maxИндекс 2*,...] OF *Тип_данных;*
    Index: тип данных - INT (-32768...32767)



Meas_value: ARRAY[1..10]

Meas_value[1]:	**Real**
Meas_value[2]:	**Real**
Meas_value[3]:	**Real**
.	
.	
Meas_value[10]:	**Real**

Примеры:

- Объявление переменных:
  - одномерная: *Meas_value*: ARRAY[*1*..*10*] OF *REAL;*
  - многомерная:*Position*: ARRAY[*1*..*5*,*2*..*8*,...] OF *INT;*

- Доступ к переменным:
  - L #Meas_value[5] // Зазрузить 5-й элемент массива // Meas_value в ACCU1
  - T #Result[10,5]

Объявление и инициализация массивов

DB5 "Declaration View"

Address	Name	Type	Initial Value
0.0		STRUCT	
+0.0	sequence	ARRAY[1..10]	5 (2.730000e+002) , 3 (1.000000e+001)
*4.0		REAL	
+40.0	result	ARRAY[1..5,3..7]	10 (
*2.0		INT	
=90.0		END STRUCT	

Press F1 for help.

DB5 "Data View"

LAD/STL/FBD - [test\ARRAYs\...\DB5 - <Offline>]

File Edit Insert PLC Debug View Options Window Help

Address	Name	Type	Initial Value	Actual Value
0.0	sequence[1]	REAL	2.730000e+002	2.730000e+002
4.0	sequence[2]	REAL	2.730000e+002	2.730000e+002
8.0	sequence[3]	REAL	2.730000e+002	2.730000e+002
12.0	sequence[4]	REAL	2.730000e+002	2.730000e+002
16.0	sequence[5]	REAL	2.730000e+002	2.730000e+002
20.0	sequence[6]	REAL	1.000000e+001	1.000000e+001
24.0	sequence[7]	REAL	1.000000e+001	1.000000e+001
28.0	sequence[8]	REAL	1.000000e+001	1.000000e+001
32.0	sequence[9]	REAL	0.000000e+000	1.000000e+001
36.0	sequence[10]	REAL	0.000000e+000	1.000000e+001
40.0	result[1, 3]	INT	5	5
42.0	result[1, 4]	INT	5	5

Press F1 for help. | Stat. Data:90 | Dyn. Data:0 | Insert

Хранение переменных типа ARRAY в памяти

одномерный массив

- **Тип данных BOOL**

	7	6	5	4	3	2	1	0
Байт n[1)]	8	7	6	5	4	3	2	1
Байт n+1		и т.д.			12	11	10	9

- **Тип данных BYTE, CHAR**

Байт n[1)]	Байт 1
Байт n+1	Байт 2
Байт n+2	Байт 3

⋮

- **Тип данных WORD, DWORD,...**

Байт n[1)]	Слово 1
Байт n+1	
Байт n+2	Слово 2
Байт n+2	

⋮

[1)] n = четное

многомерный массив

- **Пример.**
  ARRAY[1..2,1..3,1..2] OF BYTE

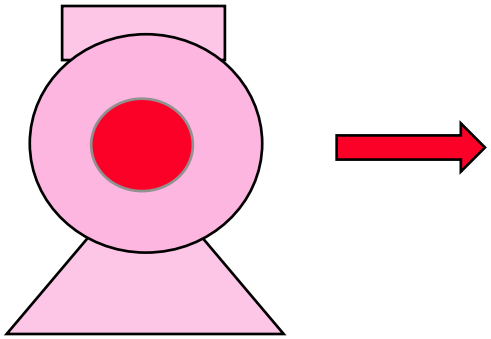
Байт n[1)]	Байт 1.1.1
Байт n+1	Байт 1.1.2
Байт n+2	Байт 1.2.1
⋮	Байт 1.2.2
	Байт 1.3.1
	Байт 1.3.2
	Байт 2.1.1
	Байт 2.1.2
	Байт 2.2.1
	Байт 2.2.2
	Байт 2.3.1
	Байт 2.3.2

Тип данных: STRUCT

STRUCT (Структура):

- Группа компонентов различных типов данных
- Объявление:

StructName: STRUCT
Имя_комп1: *Тип_данных;*
Имя_комп2: *Тип_данных;*
...
END_STRUCT



Motor: STRUCT

Set_Speed: REAL
Actual_Speed: REAL
Enable: BOOL
Disturbance: BOOL

END_STRUCT

Пример:

- Объявление переменных:
  - MotorControl : STRUCT
    ON : BOOL;
    OFF : BOOL;
    SetSpeed : INT;
    ActualSpeed : INT;
    END_STRUCT;

Доступ к переменным

S #MotorControl.ON
L #MotorControl.ActualSpeed
T #MotorControl.SetSpeed
...

Объявление структур

Пример: Объявление массива структур с полями типа ARRAY

DB6 "Declaration View"

LAD/STL/FBD - [DB6 -- test\SIMATIC 40

File Edit Insert PLC Debug View Options Window Help

Address	Name	Type	Initial Value	Comment
0.0		STRUCT		
+0.0	Axis	ARRAY[1..4]		
*0.0		STRUCT		
+0.0	Start	BOOL	FALS	
+0.1	Stop	BOOL	TRUE	
+2.0	Position	ARRAY[1..10]		
*0.0		STRUCT		
+0.0	Cutoffpoint_fron	REAL	0.00	
+4.0	Cutoffpoint_back	REAL	0.00	
+8.0	Stoppingpoint	REAL	0.00	
=12.0		END_STRUCT		
=122.0		END_STRUCT		
=488.0		END_STRUCT		

1: Errors 2: Info

Press F1 for help. Offline

DB6 "Data View"

LAD/STL/FBD - [DB6 -- test\SIMATIC 400\CPU 414-1]

File Edit Insert PLC Debug View Options Window Help

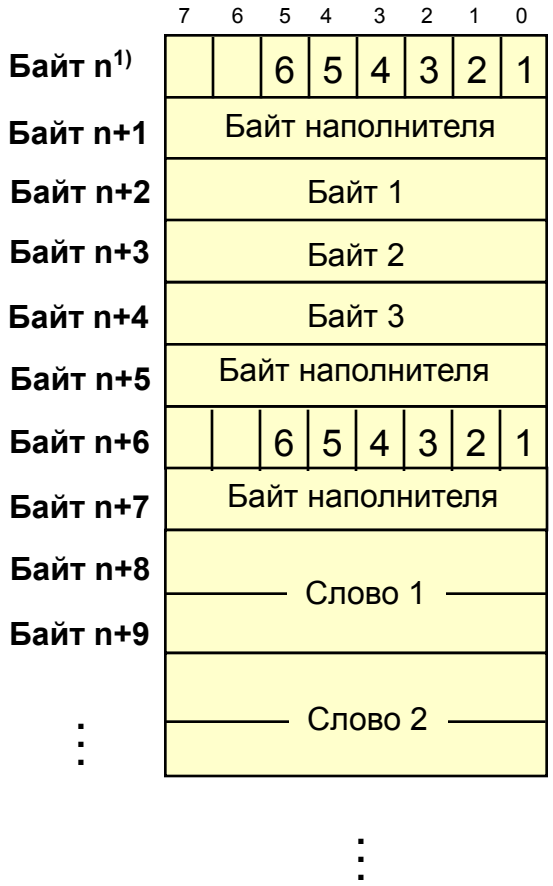
Address	Name	Type	Initial Value	Actual Value
0.0	Axis[1].Start	BOOL	FALSE	FALSE
0.1	Axis[1].Stop	BOOL	TRUE	TRUE
2.0	Axis[1].Position[1].Cutoffpoint_front	REAL	0.000000e+000	0.000000e+000
6.0	Axis[1].Position[1].Cutoffpoint_back	REAL	0.000000e+000	0.000000e+000
10.0	Axis[1].Position[1].Stoppingpoint	REAL	0.000000e+000	0.000000e+000
14.0	Axis[1].Position[2].Cutoffpoint_front	REAL	0.000000e+000	0.000000e+000
18.0	Axis[1].Position[2].Cutoffpoint_back	REAL	0.000000e+000	0.000000e+000
22.0	Axis[1].Position[2].Stoppingpoint	REAL	0.000000e+000	0.000000e+000
26.0	Axis[1].Position[3].Cutoffpoint_front	REAL	0.000000e+000	0.000000e+000
30.0	Axis[1].Position[3].Cutoffpoint_back	REAL	0.000000e+000	0.000000e+000
34.0	Axis[1].Position[3].Stoppingpoint	REAL	0.000000e+000	0.000000e+000
38.0	Axis[1].Position[4].Cutoffpoint_front	REAL	0.000000e+000	0.000000e+000
42.0	Axis[1].Position[4].Cutoffpoint_back	REAL	0.000000e+000	0.000000e+000
46.0	Axis[1].Position[4].Stoppingpoint	REAL	0.000000e+000	0.000000e+000
50.0	Axis[1].Position[5].Cutoffpoint_front	REAL	0.000000e+000	0.000000e+000
54.0	Axis[1].Position[5].Cutoffpoint_back	REAL	0.000000e+000	0.000000e+000
58.0	Axis[1].Position[5].Stoppingpoint	REAL	0.000000e+000	0.000000e+000

1: Errors 2: Info

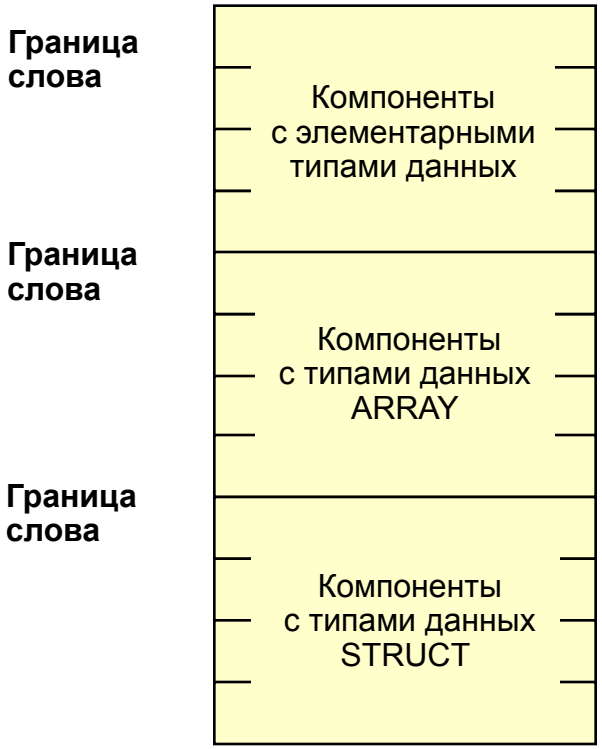
Press F1 for help. Offline Abs INS MOD

Хранение переменных типа STRUCT в памяти

Структуры с элементарными типами данных



Структуры со сложными типами данных

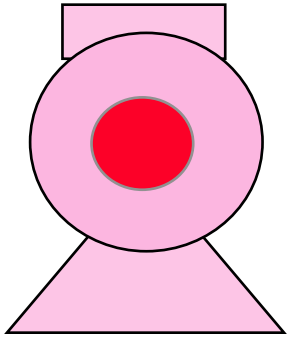


[1)] n = четное

Тип данных, определенный пользователем: UDT

UDT- тип данных, определенный пользователем:

- **создается шаблон для дальнейшего использования в объявлениях**
- **доступен для всех блоков из программной папки**

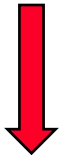


Пример:

- **Определение нового типа данных (структуры):**

```
UDT1    STRUCT
        SetSpeed : REAL;                    ...
        ActualSpeed    : REAL;
        Enable     : BOOL;
        Disturbance     : BOOL;
    END_STRUCT;
```

- **Объявление переменных:**
  - Motor_1: UDT1;
    Motor_2: UDT1;
- **Доступ к переменной:**
  - L #Motor_1.ActualSpeed



UDT1: STRUCT

Set_Speed: REAL
Actual_Speed: REAL
Enable: BOOL
Disturbance: BOOL

END_STRUCT

Тип данных: DATE_AND_TIME

Структура:

Байт n[1)]	Год (90 ... 89)		Месяц (01 ... 12)	Байт n+1
Байт n+2	День (01 ... 31)		Часы (00 ... 23)	Байт n+3
Байт n+4	Минуты (00 ... 59)		Секунды (00 ... 59)	Байт n+5
Байт n+6	Миллисекунды (000 ... 999)			Байт n+7
			День недели (1..7)	

1=Воскресенье
2=Попедельник
3=Вторник
4=Среда
5=Четверг
6=Пятница
7=Суббота

- Все значения храняться в BCD формате
- Предустановки переменных:
  DT#Год-Месяц-День-Часы:Минуты:Секунды.[Миллисекунды]
  Пример: DT#1998-03-21-17:23:00:00
- Работа через функции IEC-библиотеки

[1)] n = четное

Тип данных: STRING

Переменная типа STRING (строка) :

- **Тип данных STRING - строка символов до 254 символов длиной**
- **Применение: обработка текстовых сообщений**
- **Объявление:**
  - *Имя_Строки: STRING[maxNo]:* *‘ Текст_инициализации'*
    (Строка максимум из maxNo символов, maxNo: 0... 254)
  - *Имя_Строки:* STRING: *' Текст_инициализации'*
    (Строка максимум из 254 символов)

Примеры:

- **Объявление переменной:**
  - Fault_signal : STRING 'Motor_ failure_4'
    (Переменная *Fault signal* инициализируется указанным текстом)
  - Warning : STRING[50] ' '
    (“пустая" переменная *Warning,* может содержать до 50 символов)
- **Обработка:**
  - элементарный доступ:
    L #Fault_signal[5] (загрузить 5 -й символ из *Fault_ signal*)
  - Обработка посредством FC из IEC- библиотеки

Хранение строковых переменных в памяти

Пример:

- **Объявление с инициализацией**
  - Given_ name: STRING[8]: 'OTTO'
- **Хранение строковой переменной "Given_ name"**

Байт n[1)]	max длина= 8
Байт n+1	Текущая длина= 4
Байт n+2	1-й символ= 'O'
Байт n+3	2-й символ = 'T'
Байт n+4	3-й символ = 'T'
Байт n+5	4-й символ = 'O'
Байт e n+6	B#16#00
Байт n+7	B#16#00
Байт n+8	B#16#00
Байт n+9	B#16#00

⋮

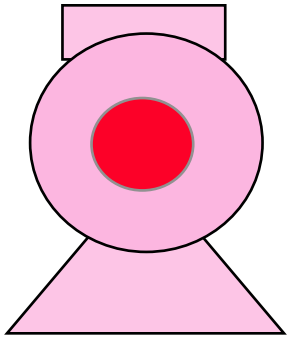
→ **Определяет максимальное число сохраняемых знаков, то есть длина, указанная в декларации**

→ **Определяет число знаков в настоящее время сохраненных в переменной типа STRING**

▯ **Информация относительно максимального числа сохраняемых знаков или относительно текущей длины может быть оценена функциями IEC-библиотеки.**

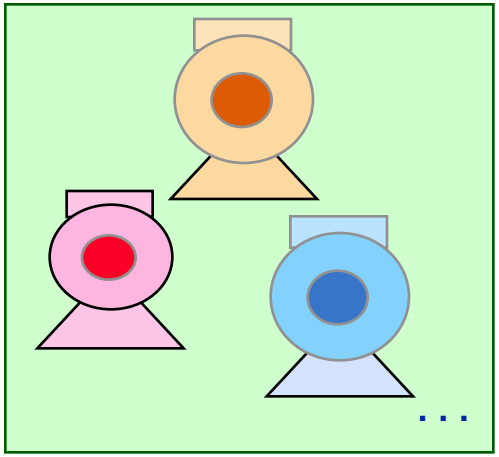
[1)] **n = четное**

Упражнение 5.1: Использование сложных типов данных



UDT99 "Motor"

Address	Name	Type	Initial Value
0.0		STRUCT	
+0.0	SetSpeed	REAL	0.000000e+000
+4.0	ActualSpeed	REAL	0.000000e+000
+8.0	SetActDiffMax	REAL	5.000000e-002
+12.0	Enable	BOOL	FALSE
+12.1	Disturbance	BOOL	TRUE
=14.0		END_STRUCT	



Hall_1

DB51 "Conv_area_Motors"

Address	Name	Type	Initial Value
0.0		STRUCT	
+0.0	ConvArea_1_Motor	ARRAY[1..20]	
*14.0		UDT99	
+280.0	ConvArea_2_Motor	ARRAY[1..20]	
*14.0		UDT99	
=560.0		END_STRUCT	