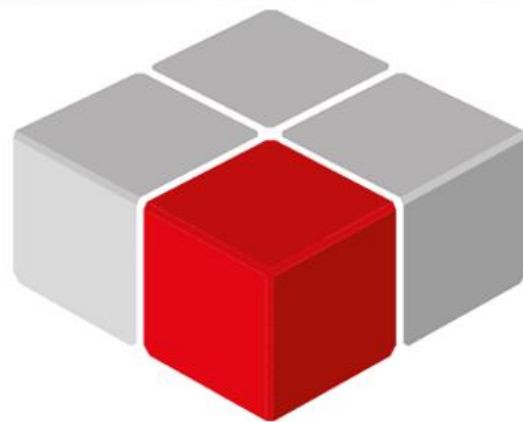


**CODESYS**

# Kurs programowania **CODESYS** w języku ST

**CONTROL**  **byte**®



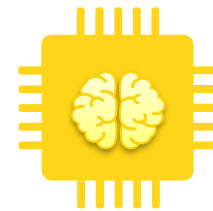
**CODESYS**

## 2) Zmienne



# Plan

- Podstawowe jednostki pamięci w Codesys
- Definicja zmiennej oraz stałej
- Typy zmiennych
- Deklarowanie zmiennych
- Konwersja typów
- Tablice oraz struktury
- Problematiczne zagadnienia



# Podstawowe jednostki pamięci w Codesys



Bit – najmniejsza jednostka pamięci



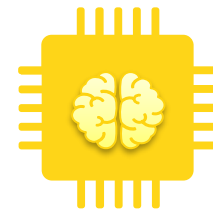
Byte – składa się z 8 bitów



Word czyli 2 Byte'y



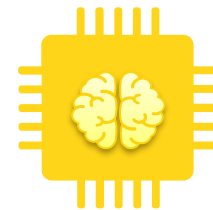
DoubleWord  
czyli dwa Wordy



# Podstawowe jednostki pamięci w Codesys

| Double Word | DW2          |     |              |    | DW1          |    |              |    | DW0          |    |              |    |
|-------------|--------------|-----|--------------|----|--------------|----|--------------|----|--------------|----|--------------|----|
| Word        | W5           |     | W4           |    | W3           |    | W2           |    | W1           |    | W0           |    |
| Byte        | B11          | B10 | B9           | B8 | B7           | B6 | B5           | B4 | B3           | B2 | B1           | B0 |
| Bit         | 5.15-<br>5.0 |     | 4.15-<br>4.0 |    | 3.15-<br>3.0 |    | 2.15-<br>2.0 |    | 1.15-<br>1.0 |    | 0.15-<br>0.0 |    |

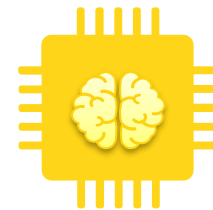
W celu dostania się do konkretnego bitu danego słowa należy odnieść się do numeru słowa i określić numer bitu np.: W1.5



# Tryb adresowania dla urządzeń

| Double Word | DW2        |     |            |    | DW1        |    |            |    | DW0        |    |            |    |
|-------------|------------|-----|------------|----|------------|----|------------|----|------------|----|------------|----|
| Word        | W5         |     | W4         |    | W3         |    | W2         |    | W1         |    | W0         |    |
| Byte        | B11        | B10 | B9         | B8 | B7         | B6 | B5         | B4 | B3         | B2 | B1         | B0 |
| Bit         | 5.0 - 5.15 |     | 4.0 - 4.15 |    | 3.0 - 3.15 |    | 2.0 - 2.15 |    | 1.0 - 1.15 |    | 0.0 - 0.15 |    |

| Double Word | DW2        |    |            |    | DW2        |    |            |    | DW0        |    |            |    |
|-------------|------------|----|------------|----|------------|----|------------|----|------------|----|------------|----|
| Word        | W3         |    | W2         |    | W2         |    | W1         |    | W1         |    | W0         |    |
| Byte        | B6         | B5 | B5         | B4 | B4         | B3 | B3         | B2 | B2         | B1 | B1         | B0 |
| Bit         | 5.0 - 5.15 |    | 4.0 - 4.15 |    | 3.0 - 3.15 |    | 2.0 - 2.15 |    | 1.0 - 1.15 |    | 0.0 - 0.15 |    |



# Pamięć fizyczna

Każdy sterownik PLC programowany w Codesys posiada pamięć rezerwowaną dla fizycznych wejść **%I** , fizycznych wyjść **%Q**, a także pamięć pomocniczą **%M**

| Prefix dla zakresu pamięci | Opis                                              | Przykłady adresowania                                                                                             |
|----------------------------|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| I                          | Zakres pamięci zarezerwowany dla wejść fizycznych | %IW25 – adres wejściowy dla 25 Worda<br>%IX7.5 – dostęp konkretnego bitu<br>%IB10 – dostęp do konkretnego Byte'u  |
| Q                          | Zakres pamięci zarezerwowany dla wyjść fizycznych | %QW50 – adres wyjściowy dla 50 Worda<br>%QX10.5 – dostęp konkretnego bitu<br>%QB15 – dostęp do konkretnego Byte'u |
| M                          | Zakres dla pamięci markerów (pamięć pomocnicza)   | %MD12 – adres dla double word w pamięci markerów<br>%MX30.1- bit w Wordzie 30, pamięć markerów                    |



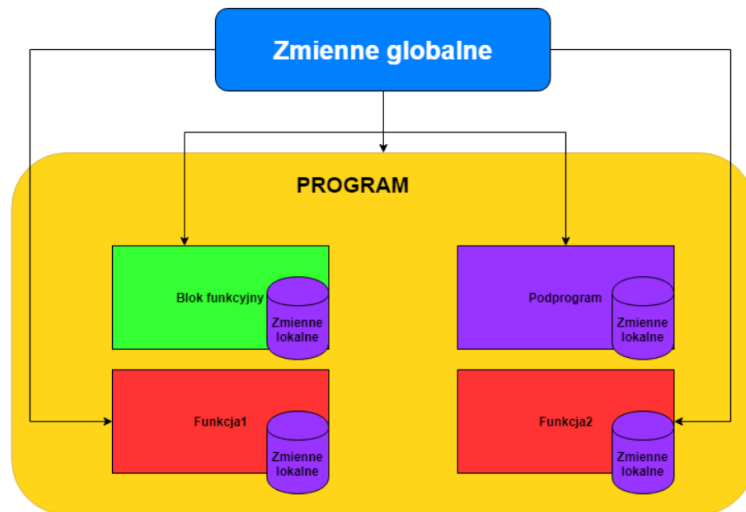
# Zmienna - **variable**

- Przechowuje dane
- Posiada nazwę nadaną przez programistę
- Posiada typ danych
- Rezerwuje miejsce w pamięci sterownika PLC
- Zmienia wartość podczas wykonywania programu

# Zmienne globalne/lokalne

Zmienne globalne - dostęp z każdego miejsca programu użytkownika

Zmienne lokalne - dostęp tylko wewnątrz FB, FC, PRG, gdzie zostały zadeklarowane





# Stała - constant

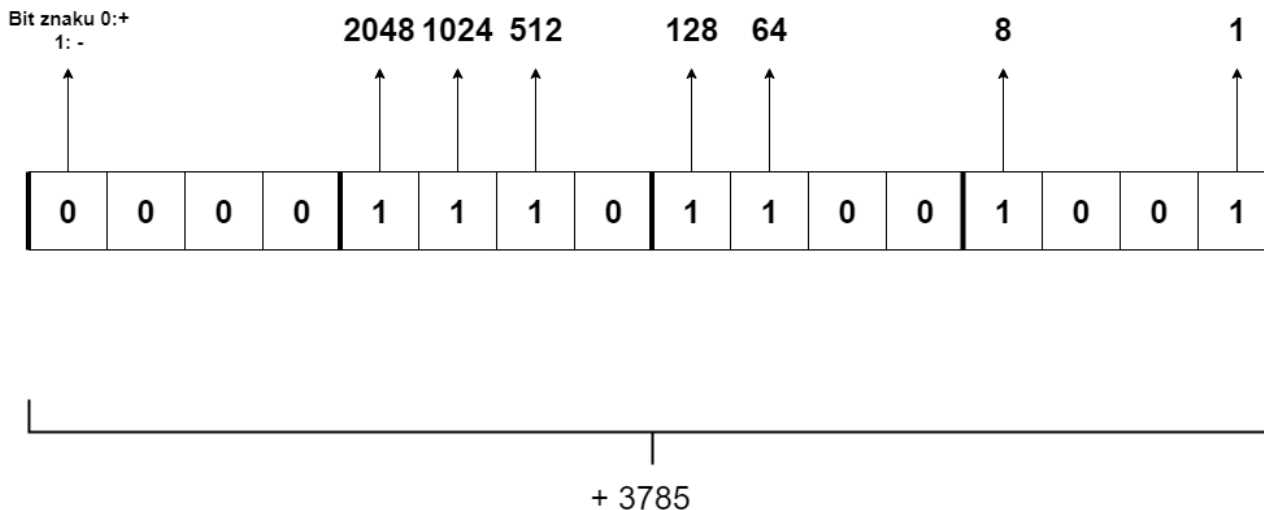
- Przechowuje dane
- Posiada nazwę
- Posiada typ danych
- Rezerwuje miejsce w pamięci sterownika PLC
- **Nie zmienia** wartości podczas wykonywania programu

# T y p y z m i e n n y c h

| Nazwa         | Prefiks | Rozmiar |       | Zakres                                                                 | Przykład                                                |
|---------------|---------|---------|-------|------------------------------------------------------------------------|---------------------------------------------------------|
|               |         | bity    | bajty |                                                                        |                                                         |
| BOOL          | x       | 8       | –     | 0, 1                                                                   | TRUE                                                    |
| BYTE          | b       | 8       | 1     | 0...255                                                                | 16#F                                                    |
| WORD          | w       | 16      | 2     | 0...65535                                                              | 16#F0F0                                                 |
| DWORD         | dw      | 32      | 4     | 0... 4294967295                                                        | 16#FFFFFFFF                                             |
| LWORD         | lw      | 64      | 8     | 2 <sup>64</sup> -1                                                     | -                                                       |
| USINT         | usi     | 8       | 1     | 0... 255                                                               | 78                                                      |
| UINT          | ui      | 16      | 2     | 0... 65 535                                                            | 65865                                                   |
| UDINT         | udi     | 32      | 4     | 0... 4294967295                                                        | 24891924                                                |
| SINT          | si      | 8       | 1     | -128... 127                                                            | -50                                                     |
| INT           | i       | 16      | 2     | -32 768... 32 767                                                      | 23562                                                   |
| DINT          | di      | 32      | 4     | -2 147 483 648...2 147 483,647                                         | -2131754992                                             |
| REAL          | r       | 32      | 4     | -3.402823e+38... -1.175 495e-38, ±0,<br>+1.175 495e-38...+3.402823e+38 | 123.456 -3.4                                            |
| LREAL         | lr      | 64      | 8     | -                                                                      | 123.456 -3.4                                            |
| STRING        | s       | -       | 255   | „...”ciąg znaków                                                       | ‘Kurs programowania’                                    |
| DATE          | date    | 16      | 2     | D#1970-1-1...D#2106-02-07                                              | D#2019-02-10                                            |
| TIME          | tim     | 32      | 4     | T#0d0h0m0s0ms ...<br>T#49d17h2m47s295ms                                | T#5m_30s T#1d_2h_15m_30s_45ms<br>TIME#10d20h30m20s630ms |
| TIME_OF_DAY   | tod     | 32      | 4     | TOD#0:0:0.0 ...TOD#23:59:59.999                                        | TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400               |
| DATE_AND_TIME | dt      | 32      | 4     | DT#1979-1-1-00:00:00 ...<br>DT#2106-2-7-6:28:15                        | DT#2019-2-7-6:28:15                                     |



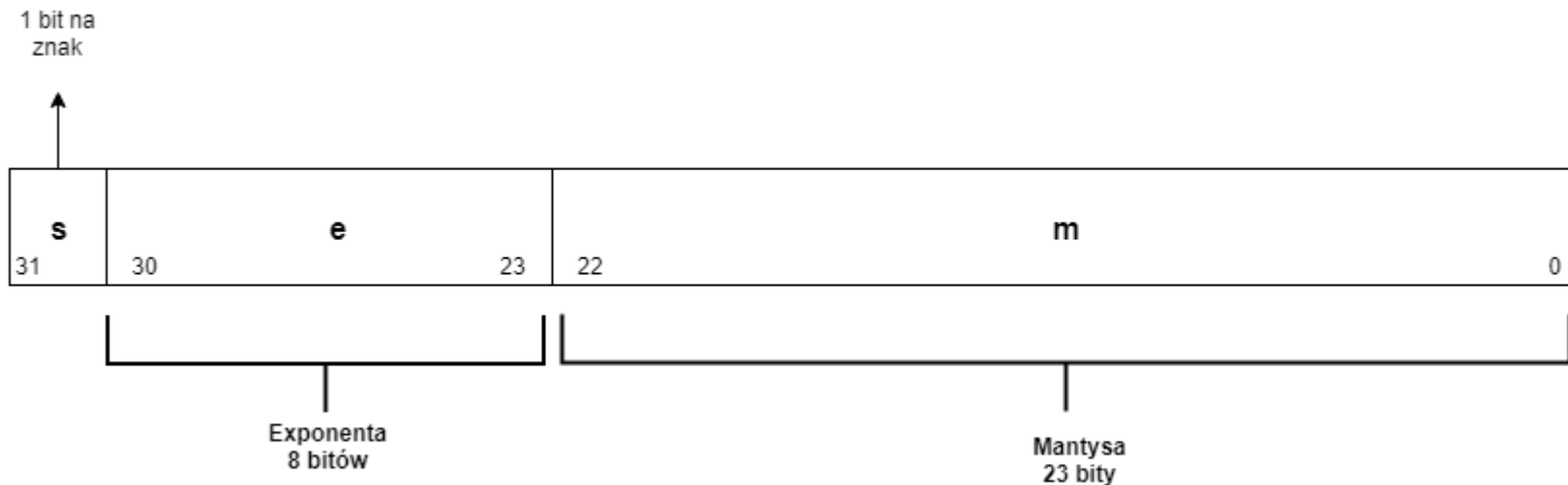
# Typ danych integer



Reprezentacja liczby dodatniej 3785 na 16 bitach typu INT. 14 bitów zaczynając od prawej reprezentuje dane. Bit 15 określa znak. 0 – określa dodatni znak, 1 – określa negatywny znak



# Liczby zmiennoprzecinkowe



Zmienna typu REAL składa się z 32 bitów, jest to liczba zmiennoprzecinkowa, składa się z następujących komponentów: bit 31 reprezentuje znak liczby, exponenta – 8 bitów odpowiada za rząd wielkości, mantysa – 23 bity czyli część liczby po przecinku

# Deklarowanie zmiennych

| Pole                  | Opis                                                                                                                                                                  |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Name</i>           | nazwa zmiennej, która musi być unikalna dla danej jednostki organizacyjnej oprogramowania. Nazwy zmiennych mogą się powtarzać w różnych POU.                          |
| <i>Type</i>           | określa typ zmiennej                                                                                                                                                  |
| <i>Initialization</i> | wpisana wartość inicjalizuje zmienną podczas rozpoczęcia wykonywania programu                                                                                         |
| <i>Address</i>        | Opcja deklaracji odpowiedniego adresu pamięci w sterowniku PLC                                                                                                        |
| <i>Comment</i>        | Dodawanie komentarza przy deklaracji danej zmiennej                                                                                                                   |
| <i>Scope</i>          | Należy określić czy jest to zmienna wejściowa/wyjściowa VAR_INPUT, VAR_OUTPUT etc. Ważne przy działaniu w danym POU.                                                  |
| <i>Object</i>         | Określenie w jakiej jednostce organizacyjnej oprogramowania ma być zadeklarowana zmienna                                                                              |
| <i>Flags</i>          | Dostępne są trzy pola do zaznaczenia: CONSTANT – deklaracja stałej, RETAIN – zmienna nieulotna z pamięci PLC, PERSISTENT- zmienna odporna na ponowne wgranie programu |

# Deklarowanie zmiennych/stałych

Auto Declare

Scope: VAR

Name: Zmienna

Type: INT

Object: PLC\_PRG [Application]

Initialization: 100

Address: %MW0

Flags:

- ☐ CONSTANT
- ☐ RETAIN
- ☐ PERSISTENT

Comment: zmienna odpowiedzialna za zliczanie

OK Cancel

# Konwersja

```
ZmiennaWyj := <typ zmiennej wejściowej>_TO_<typ zmiennej wyjściowej>(IN := ZmiennaWej);
```

## Konwersja typów

BOOL\_TO\_BYTE

BOOL\_TO\_WORD

BOOL\_TO\_DWORD

BYTE\_TO\_WORD

DINT\_TO\_REAL

INT\_TO\_DINT

INT\_TO\_REAL

WORD\_TO\_DWORD



Uwaga!!!

Konwersja typu może spowodować częściową utratę wartości lub znaku zmiennej.

# Notacja w systemach liczbowych

```
1.iZmienna := 2#00010011; // zapis dwójkowy lub binarny  
2.iZmienna := 8#23; // zapis ósemkowy  
3.iZmienna := 19; // zapis dziesiętny  
4.iZmienna := 16#13 // zapis szesnastkowy
```

# Dane typu Array - tablice

Array to struktura składająca się ze zmiennych tego samego typu.

Można ją zastosować np. do działań matematycznych, zapisywania receptur, śledzenia materiałów w kolejce itp.

Prefiks dla tablic: a np. aGniazdaMaszyny

Deklaracja tablicy w Codesys:

aMojaNazwaTablicy: array[dzakres..gzakres] of type

Przykład: **aPompy[1..10] of BOOL**

| aTablica      |  | ARRAY [0..10] OF INT |     |
|---------------|--|----------------------|-----|
| ◆ aTablica[0] |  | INT                  | 10  |
| ◆ aTablica[1] |  | INT                  | 20  |
| ◆ aTablica[2] |  | INT                  | 50  |
| ◆ aTablica[3] |  | INT                  | 60  |
| ◆ aTablica[4] |  | INT                  | 80  |
| ◆ aTablica[5] |  | INT                  | 100 |

# Dane typu STRUCT oraz typy danych PLC

STRUCT to struktura składająca się z danych różnego typu.

Zadeklarowanie struktury odbywa się na poziomie danego POU lub w zmiennych globalnych.

Typy danych PLC określone są dla całego sterownika. Wszelkie zmiany zostaną wprowadzone dla wszystkich danych tego typu w obrębie programu użytkownika. Deklaracja struktur w Codesys odbywa się w drzewie projektu klikając opcję DUT (Data User Type) typ danych użytkownika

Prefiks dla struktur: typ  
Np. typSilnik

```
TYPE typSilnik :  
STRUCT  
END_STRUCT  
    arrValves: ARRAY[0..5] OF BOOL;  
    bSpeed_Min: BYTE;  
    bSpeed_Max: BYTE;  
    Temperature: WORD;  
    arrAxes: ARRAY[0..4] OF BOOL;  
    bOil: BYTE  
END_TYPE
```

# Enumeracja

Enumeracja jest typem danych definiowanym przez użytkownika, składającym się z serii oddzielonych przecinkami komponentów (wartości wyliczeniowych). Wartości wyliczeniowe można używać globalnie w całym projekcie i traktować je jako stałe.

Nie ma określonego prefixu, nazywamy dowolnie

```
TYPE SygnalizacjaSwietlna :  
(  
    zielone := 0,  
    zolte   := 1,  
    czerwone := 2  
);  
END_TYPE
```