EL-PIAST SP. Z O.O.

Macrocontrol

Opis standardowych bloków funkcyjnych

Ryszard Dworzecki

2016-04-20

Opis bloków funkcyjnych zawartych w standardowych bibliotekach oprogramowania Macrocontrol.

Spis treści

1. Wprowadzenie 7
2. Alarm 9
3. Logic 10
3.1. Negation 10
3.2. Exclusive or (Różnica symetryczna) 10
3.3. Left shift 8 bits (8-bitowe przesunięcie w lewo) 11
3.4. Right shift 8 bits (8-bitowe przesunięcie w prawo) 11
3.5. Left Rotate 8 bits (8-bitowa rotacja w lewo) 12
3.6. Right rotate 8 bits (8-bitowa rotacja w prawo) 13
3.7. Left shift analog (Przesunięcie w lewo wartości analogowej) 13
3.8. Right shift analog (Przesunięcie w prawo wartości analogowej) 14
3.9. Left rotate analog (Rotacja w lewo wartości analogowej) 14
3.10. Right rotate analog (Rotacja w prawo wartości analogowej) 14
3.11. Bit active no (Liczba aktywnych bitów) 15
4. Arithmetic 16
4.1. Minus 16
4.2. Integer (Wartość całkowita) 16
4.3. Sum (Suma) 16
4.4. Difference (Różnica) 17
4.5. Multiplication (Mnożenie) 17
4.6. Division (Dzielenie) 17
4.7. Modulo (Dzielenie modulo) 18
4.8. Maximum 18
4.9. Minimum 19
4.10. Absolute (Wartość bezwzględna) 19
4.11. Square root (Pierwiastek kwadratowy) 19
4.12. Power (Potęga) 20
4.13. Natural logarithm (Logarytm naturalny) 20
4.14. Logarithm 10 (Logarytm dziesiętny) 20
4.15. Exponent e (Potęga liczby Eulera) 21
4.16. Sinus 21

4.17. Cosinus ... 21
4.18. Tangens ... 21
4.19. Arcus sinus ... 22
4.20. Arcus cosinus ... 22
4.21. Arcus tangens ... 22
4.22. Deg2Rad (Konwersja stopni na radiany) ... 23
4.23. Rad2Deg (Konwersja radianów na stopnie) ... 23
4.24. Non-linear func 10 (10-punktowa nieliniowa funkcja) ... 24
4.25. Non-linear func 5 (5-punktowa nieliniowa funkcja) ... 25
4.26. Scale (Skalowanie) ... 25
4.27. Scale2 (Skalowanie wersja 2) ... 26
4.28. Random (Wartość losowa) ... 27
5. Triggers ... 29
5.1. Trigger (Przerzutnik) ... 29
5.2. D trigger (Przerzutnik D) ... 29
5.3. RS trigger (Asynchroniczny przerzutnik z priorytetem kasowania) ... 29
5.4. SR trigger (Asynchroniczny przerzutnik SR z priorytetem ustawiania) ... 30
5.5. JK trigger (Przerzutnik JK) ... 30
5.6. Analog Trigger (Przerzutnik analogowy) ... 31
6. Timeline ... 32
6.1. Delay 1 (Opóźnienie stanu 1) ... 32
6.2. Delay 0 (Opóźnienie stanu 0) ... 32
6.3. Delay 01 (Opóźnienie 0 i 1) ... 33
6.4. Impulse (Impuls) ... 33
6.5. Minimum impulse (Minimalny impuls) ... 34
6.6. Maximum impulse (Maksymalny impuls) ... 34
6.7. PWM (Modulator długości impulsu) ... 35
6.8. Generator ... 35
6.9. PID regulator (Regulator proporcjonalno – różniczkująco – całkujący) ... 36
6.10. PID2 regulator (Regulator proporcjonalno – różniczkująco – całkujący, wersja 2) ... 36
6.11. PID3 regulator (Regulator proporcjonalno – różniczkująco – całkujący, wersja 3) ... 37
6.12. Follower (Funkcja nadążająca) ... 38
6.13. Filter (Filtr) ... 38
6.14. Average (Średnia) ... 39

6.15. System time (Czas systemowy) ... 39
6.16. Time interval (Odstęp czasu) ... 40
6.17. Time compare (Porównanie czasów) ... 40
6.18. Time downcounter (Timer) ... 40
6.19. Max24 (Maksymalna wartość z ostatnich 24h) ... 41
6.20. Calendar Week (Kalendarz tygodniowy) ... 42
6.21. Calendar Vacation (Kalendarz wakacji) ... 42
7. Multiplex ... 44
7.1. Digital Switcher (Przełącznik cyfrowy) ... 44
7.2. Analog Switcher (Przełącznik analogowy) ... 44
7.3. Digital Multiplexer 4Ch (Multiplekser cyfrowy 4-kanałowy) ... 45
7.4. Digital Multiplexer 8Ch (Multiplekser cyfrowy 8-kanałowy) ... 45
7.5. Digital Demultiplexer 4Ch (Demultiplekser 4-kanałowy) ... 46
7.6. Digital Demultiplexer 8Ch (Demultiplekser 8-kanałowy) ... 46
7.7. Analog Multiplexer 4Ch (Multiplekser analogowy 4-kanałowy) ... 47
7.8. Analog Multiplexer 8Ch (Multiplekser analogowy 8-kanałowy) ... 47
7.9. Analog Demultiplexer 4Ch (Demultiplekser analogowy 4-kanałowy) ... 48
7.10. Analog Demultiplexer 8Ch (Demultiplekser analogowy 8-kanałowy) ... 48
8. Compare ... 50
8.1. Limitation ... 50
8.2. Signaling (Sygnalizacja z histerezą) ... 50
8.3. Equal (Równy) ... 51
8.4. Greater (Większy) ... 51
8.5. Greater or equal (Większy lub równy) ... 51
8.6. Lesser (Mniejszy) ... 52
8.7. Lesser or equal (Mniejszy lub równy) ... 52
9. Conversion ... 53
9.1. An to Dig8 (Rozbicie wartości analogowej na 8 bitów) ... 53
9.2. An to Dig16 (Rozbicie wartości analogowej na 16 bitów) ... 53
9.3. Dig8 to An (Scalenie 8 bitów w sygnał analogowy) ... 54
9.4. Dig16 to An (Scalenie 16 bitów w sygnał analogowy) ... 55
9.5. Conv2Time (Konwersja do czasu) ... 55
9.6. Conv2Date (Konwersja do daty) ... 56
9.7. ConvFromTime (Konwersja z czasu) ... 56

9.8. ConvFromDate (Konwersja z daty) 57
9.9. Sec2Time (Konwersja liczby sekund na czas) 57
9.10. Time2Sec (Konwersja czasu na liczbę sekund) 57
9.11. Date2Days (Konwersja daty na liczbę dni) 58
9.12. Days2Date (Konwersja liczby dni na datę) 58
9.13. Float2Fixed (Konwersja liczby z formatu float do fixed) 58
9.14. Fixed2Float (Konwersja liczby z formatu fixed do float) 59
10. Others 60
10.1. Counter (Licznik) 60
10.2. Stabilizer (Stabilizator sygnału) 60
10.3. FIFO (Kolejka FIFO 8 argumentów) 61
10.4. LIFO (Kolejka LIFO 8 argumentów) 61
10.5. Get Value 2 (Ostatnio zmieniona wartość z dwóch) 62
11. System 63
11.1. Temp test (Test pomiaru temperatury) 63
11.2. Ctrl sensor (Czujnik temperatury z panelu HMI) 63
11.3. Ctrl sensor 2 (Czujnik temperatury z HMI RS-485 Master) 63
11.4. Com detection (Wykrywanie komunikacji) 64
11.5. Calendar (Aktywacja kalendarza) 64
11.6. Alarm state (Stany alarmowe) 65
11.7. First loop (Pierwsza pętla programu) 65
11.8. Loop count (Licznik pętli) 65
11.9. Page deactive (Dezaktywacja strony menu) 66
11.10. An_in mode (Tryb pracy wejścia analogowego) 66
11.11. RTC (Dostęp do zegara RTC) 67
11.12. RTC2 (Dostęp do zegara RTC wersja 2) 68
11.13. BACnet Error (błąd BACnet) 68
11.14. Restore Default (Przywrócenie ustawień domyślnych) 68
11.15. Set Variable (Ustawienie zmiennej) 69
11.16. Keys (Klawisze HMI) 69
11.17. Language (Wersja językowa HMI) 69
12. Communication Basic 71
12.1. MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości) 71
12.2. MS VarRead 4 (Odczyt czterech 32-bitowych zmiennych) 72

12.3. MS VarRead 8 (Odczyt ośmiu 32-bitowych zmiennych) 73
12.4. MS VarWrite 1 (NE) (Zapis pojedynczej 32-bitowej zmiennej) 74
12.5. MS VarWrite 4 (NE) (Zapis czterech 32-bitowych zmiennych) 75
12.6. MS VarWrite 8 (NE) (Zapis ośmiu 32-bitowych zmiennych) 77
12.7. MB CoilRead 1 (Odczyt pojedynczej zmiennej binarnej Modbus RTU, komenda: 0x01). 78
12.8. MB CoilRead 4 (Odczyt czterech zmiennych binarnych Modbus RTU, komenda: 0x01).. 79
12.9. MB CoilRead 8 (Odczyt ośmiu zmiennych binarnych Modbus RTU, komenda: 0x01)..... 80
12.10. MB InputRead 1_2 (Odczyt jednego binarnego wejścia Modbus RTU, komenda: 0x02) . 81
12.11. MB InputRead 4_2 (Odczyt czterech binarnych wejść Modbus RTU, komenda: 0x02).... 82
12.12. MB InputRead 8_2 (Odczyt ośmiu binarnych wejść Modbus RTU, komenda: 0x02) 83
12.13. MB RegRead 1 (Odczyt pojedynczego rejestru Modbus RTU, komenda: 0x03)..... 84
12.14. MB RegRead 4 (Odczyt czterech rejestrów Modbus RTU, komenda: 0x03)..... 85
12.15. MB RegRead 8 (Odczyt ośmiu rejestrów Modbus RTU, komenda: 0x03) 86
12.16. MB RegRead 1_2 (Odczyt pojedynczego rejestru Modbus RTU, komenda: 0x03)..... 87
12.17. MB RegRead 4_2 (Odczyt czterech rejestrów Modbus RTU, komenda: 0x03)..... 88
12.18. MB RegRead 8_2 (Odczyt ośmiu rejestrów Modbus RTU, komenda: 0x03) 89
12.19. MB InRegRead 1_2 (Odczyt jednego rej. we. Modbus RTU, komenda: 0x04)..... 90
12.20. MB InRegRead 4_2 (Odczyt czterech rej. we. Modbus RTU, komenda: 0x04)..... 91
12.21. MB InRegRead 8_2 (Odczyt ośmiu rej. wej. Modbus RTU, komenda: 0x04)..... 92
12.22. MB CoilWrite 1 (Zapis pojedynczej binarnej zmiennej Modbus RTU, komenda: 0x05) ... 93
12.23. MB CoilWrite 4 (Zapis czterech binarnych zmiennych Modbus RTU, komenda: 0x0F).... 94
12.24. MB CoilWrite 8 (Zapis ośmiu binarnych zmiennych Modbus RTU, komenda: 0x0F)..... 95
12.25. MB RegWrite 1 (Zapis pojedynczego rejestru Modbus RTU, komenda: 0x06)..... 96
12.26. MB RegWrite 4 (Zapis czterech rejestrów Modbus RTU, komenda: 0x10) 97
12.27. MB RegWrite 8 (Zapis ośmiu rejestrów Modbus RTU, komenda: 0x10) 98
12.28. MB RegWrite 1_2 (Zapis pojedynczego rejestru Modbus RTU, komenda: 0x06)..... 99
12.29. MB RegWrite 4_2 (Zapis czterech rejestrów Modbus RTU, komenda: 0x10) 100
12.30. MB RegWrite 8_2 (Zapis ośmiu rejestrów Modbus RTU, komenda: 0x10) 101
13. Communication Devices 103
13.1. LG iC5/iG5 (Obsługa falownika LG iC5 / iG5) 103
13.2. SV iP5A (Obsługa falownika LG SV iP5A)..... 105
13.3. Danfoss FC (Obsługa falownika Danfoss)..... 107
13.4. AFC150 (Obsługa falownika TWERD)..... 109
13.5. AFC150 Config (Konfiguracja falownika TWERD)..... 111

13.6. HVinverter (Obsługa falownika HFinverter) 112
13.7. HFinverter Config (Konfiguracja falownika HFinverter) 114
13.8. Omron MX2 (Obsługa falownika Omron MX2) 114
13.9. EBMPAPST (Obsługa wentylatora EBMPAPST) 116
13.10. EBMPAPST Config (Konfiguracja wentylatora EBMPAPST) 118
13.11. PT1000 Module (Obsługa modułu PT1000) 119

1. Wprowadzenie

Niniejszy dokument opisuje wszystkie bloki funkcyjne zgromadzone w podstawowych bibliotekach oprogramowania *Macrocontrol*. Biblioteki pojawiają się w oknie bibliotek obiektów każdorazowo z otwieranym projektem aplikacji PLC.

Macrocontrol standardowo posiada 11 bibliotek podstawowych bloków funkcyjnych:

Logic – bloki funkcji logicznych

Arithmetic – bloki funkcji matematycznych

Triggers – bloki przerzutników i innych wyzwalaczy

Timeline – bloki opóźnień czasowych, regulatorów i innych funkcji związanych z czasem

Multiplex – bloki multiplekserów i demultiplekserów

Compare – bloki operacji porównujących

Conversion – bloki funkcji konwertujących wartości

Others – inne bloki nieskateryzowane wcześniej

System – bloki funkcji specjalnych dostępnych w zależności od platformy sterownika PLC

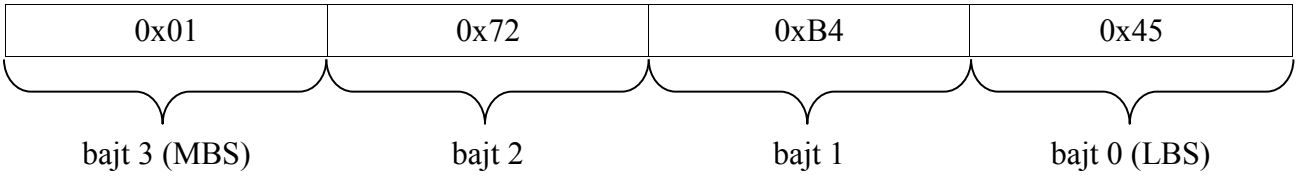
Com Basic – elementarne bloki komunikacyjne do wymiany zmiennych między sterownikami EL-Piast lub obcymi po protokole Modbus RTU

Com Devices – bloki służące do obsługi zewnętrznych urządzeń takich jak falowniki poprzez odpowiednie łącza komunikacyjne

Uwagi:

Funkcje przyjmujące wartości analogowe czasami w obliczeniach wykorzystują różne formaty reprezentacji tych wartości. Wartości analogowe w systemie *Macrocontrol* to liczby 32-dwu bitowe. W formacie *Fixed* 24 bardziej znaczące bity przeznaczone są na wartość całkowitą natomiast ostatnie najmniej znaczące 8 bitów na część ułamkową. Oznacza to że dokładność po przecinku wynosi 1/256. Te same liczby mogą być przedstawione jako całkowite 32-dwu bitowe wartości w formacie *Decimal* lub *Hex*. W przypadku formatu *Hex* liczba występuje w postaci heksadecymalnej.

Przykład:



Wartość stałoprzecinkowa: 94900,27

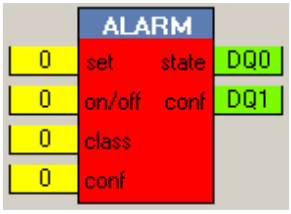
Wartość dziesiętna: 24294469

Wartość heksadecymalna: 0x0172B445

Wszystkie pomiary temperatury podawane są w stopniach Celsjusza.

Wszystkie pomiary czasu podawane są w sekundach chyba że jest napisane inaczej.

2. Alarm



Argumenty:

set – ustawienie stanu alarmowego (aktywacja)

on/off – aktywuje i dezaktywuje funkcję bloku alarmowego

class – klasa alarmu 0 – alarm blokujący (wymaga ręcznego kasowania), 1 – alarm z auto-resetem

conf – wejście potwierdzające alarm (kasuje alarm blokujący gdy *set* = 0)

Zwracane wartości:

state – stan alarmu. 1 – alarm aktywny

conf – potwierdzenie alarmu. 1 – alarm wciąż aktywny i potwierdzony

Opis:

Funkcja realizuje obsługę alarmu. Nazwa bloku jest nazwą alarmu, która potem widoczna jest w *Menu Alarmów* paneli HMI. Każdorazowe wystąpienie alarmu rejestrowane jest w systemie sterownika wraz z datą i godziną jego wystąpienia. Aktywacja alarmu wiążę się również z zbiorczą sygnalizacją stanów alarmowych najczęściej za pomocą czerwonej diody LED i podświetlenia wyświetlacza panelu HMI.

3. Logic

3.1. Negation

Argumenty:

arg – wartość cyfrowa

Zwracane wartości:

res – negacja argumentu *arg*

Opis:

Funkcja zwraca negacje logiczną argumentu *arg*. Gdy *arg* = 0 to *res = 1*, gdy *arg* = 1 to *res = 0*.

3.2. Exclusive or (Różnica symetryczna)

Argumenty:

arg1, *arg2* – wartość cyfrowa

Zwracane wartości:

res – wynik logicznej funkcji *XOR*

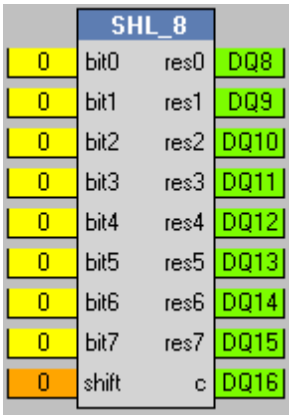
Opis:

Funkcja zwraca logiczny *XOR* między argumentami *arg1* i *arg2*.

Tabela prawdy funkcji *XOR:*

arg1	***arg2***	***res***
0	0	0
0	1	1
1	0	1
1	1	0

3.3. Left shift 8 bits (8-bitowe przesunięcie w lewo)



Argumenty:

bit0-7 – wartości dla kolejnych bitów

shift – liczba miejsc bitowych do przesunięcia. Wartość jest w formacie *Decimal* lub *Hex*

Zwracane wartości:

res0-7 – wyniki operacji dla kolejnych bitów

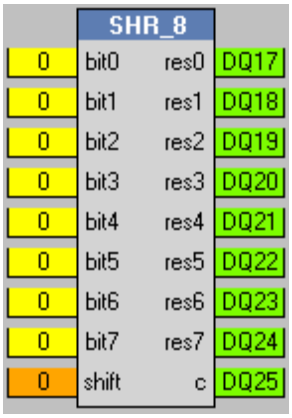
c – bit przeniesienia (carry)

Opis:

Funkcja realizuje bitowe przesunięcie w lewo dla 8 bitów. W *c* znajduje się bit który ostatnio był w *bit7*. Poniżej przedstawiony jest algorytm działania dla jednej iteracji przesunięcia (*shift* = 1):

res0 = 0, *res1* = *bit0*, *res2* = *bit1*, *res3* = *bit2*, *res4* = *bit3*, *res5* = *bit4*, *res6* = *bit5*, *res7* = *bit6*, *c* = *bit7*

3.4. Right shift 8 bits (8-bitowe przesunięcie w prawo)



Argumenty:

bit0-7 – wartości kolejnych bitów

shift – liczba miejsc bitowych do przesunięcia. Wartość jest w formacie *Decimal* lub *Hex*

Zwracane wartości:

res0-7 – wyniki operacji dla kolejnych bitów

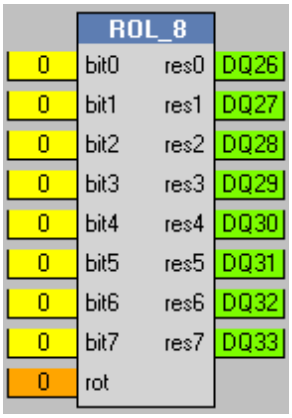
c – bit przeniesienia.

Opis:

Funkcja realizuje bitowe przesunięcie w prawo dla 8 bitów. W *c* znajduje się bit który ostatnio był w *bit0*. Poniżej przedstawiony jest algorytm działania dla jednej iteracji przesunięcia (*shift* = 1):

c = bit0, *res0 = bit1*, *res1 = bit2*, *res2 = bit3*, *res3 = bit4*, *res4 = bit5*, *res5 = bit6*, *res6 = bit7*, *res7* = 0

3.5. Left Rotate 8 bits (8-bitowa rotacja w lewo)



Argumenty:

bit0-7 – wartości kolejnych bitów

rot – liczba miejsc bitowych do rotacji. Wartość jest w formacie *Decimal* lub *Hex*.

Zwracane wartości:

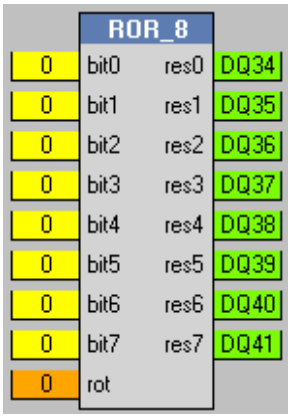
res0-7 – wyniki operacji dla kolejnych bitów

Opis:

Funkcja realizuje bitową rotacje w lewo dla 8 bitów. Poniżej przedstawiony jest algorytm działania dla jednej iteracji rotacji (*rot* = 1):

res0 = bit7, *res1 = bit0*, *res2 = bit1*, *res3 = bit2*, *res4 = bit3*, *res5 = bit4*, *res6 = bit5*, *res7 = bit6*

3.6. Right rotate 8 bits (8-bitowa rotacja w prawo)



Argumenty:

bit0-7 – wartości kolejnych bitów

rot – liczba miejsc bitowych do rotacji. Wartość jest w formacie *Decimal* lub *Hex*

Zwracane wartości:

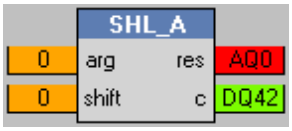
res0-7 – wyniki operacji dla kolejnych bitów

Opis:

Funkcja realizuje bitową rotacje w prawo dla 8 bitów. Poniżej przedstawiony jest algorytm działania dla jednej iteracji rotacji (*rot* = 1):

res0 = bit1, *res1 = bit2*, *res2 = bit3*, *res3 = bit4*, *res4 = bit5*, *res5 = bit6*, *res6 = bit7*, *res7 = bit0*

3.7. Left shift analog (Przesunięcie w lewo wartości analogowej)



Argumenty:

arg – wartości analogowa do przesunięcia (format: *Decimal* lub *Hex*)

shift – liczba miejsc bitowych do przesunięcia. Wartość jest w formacie *Decimal* lub *Hex*

Zwracane wartości:

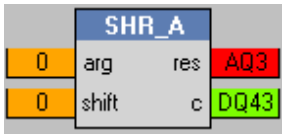
res – wynik operacji przesunięcia bitowego

c – bit przeniesienia

Opis:

Funkcja realizuje bitowe przesunięcie w lewo dla wartości analogowej *arg*. Operacja traktuje argument jak 32-bitową wartość całkowitą.

3.8. Right shift analog (Przesunięcie w prawo wartości analogowej)



Argumenty:

arg – wartości analogowa do przesunięcia (format: *Decimal* lub *Hex*)

shift – liczba miejsc bitowych do przesunięcia. Wartość jest w formacie *Decimal* lub *Hex*.

Zwracane wartości:

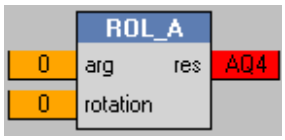
res – wynik operacji przesunięcia bitowego

c – bit przeniesienia

Opis:

Funkcja realizuje bitowe przesunięcie w prawo dla wartości analogowej *arg*. Operacja traktuje argument jak 32-bitową wartość całkowitą.

3.9. Left rotate analog (Rotacja w lewo wartości analogowej)



Argumenty:

arg – wartości analogowa do rotacji (format: *Decimal* lub *Hex*)

shift – liczba miejsc bitowych dla rotacji. Wartość jest w formacie *Decimal* lub *Hex*.

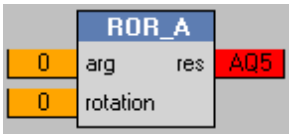
Zwracane wartości:

res – wynik rotacji bitowej

Opis:

Funkcja realizuje bitową rotację w lewo dla wartości analogowej *arg*. Operacja traktuje argument jak 32-bitową wartość całkowitą.

3.10. Right rotate analog (Rotacja w prawo wartości analogowej)



Argumenty:

arg – wartości analogowa do rotacji (format: *Decimal* lub *Hex*)

shift – liczba miejsc bitowych dla rotacji. Wartość jest w formacie *Decimal* lub *Hex*.

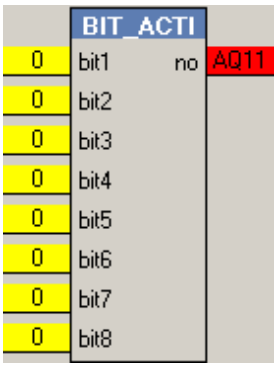
Zwracane wartości:

res – wynik rotacji bitowej

Opis:

Funkcja realizuje bitową rotację w prawo dla wartości analogowej *arg*. Operacja traktuje argument jak 32-bitową wartość całkowitą.

3.11. Bit active no (Liczba aktywnych bitów)



Argumenty:

bit1-7 – wartości kolejnych bitów

Zwracane wartości:

no – liczba aktywnych bitów (bitów równych 1). Jeżeli wszystkie bity są nieaktywne zwraca 0. Wartość jest w formacie *Decimal* lub *Hex*.

Opis:

Funkcja zlicza ilość aktywnych bitów na wejściach i zwraca ich ilość.

4. Arithmetic

4.1. Minus

Argumenty:

in – analogowy argument

Zwracane wartości:

out – wartość analogowa – *in*

Opis:

Funkcja zwraca ujemną (przeciwną) wartość argumentu *in*.

4.2. Integer (Wartość całkowita)

Argumenty:

arg – analogowy argument (format: *Fixed*)

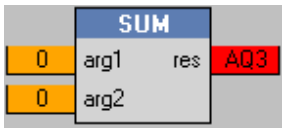
Zwracane wartości:

res – wartość analogowa int(*arg*) (format: *Fixed*)

Opis:

Funkcja zwraca część całkowitą argumentu *arg*. Przykładowo dla wartości *arg* = 45.231 wynikiem funkcji będzie *res* = 45.

4.3. Sum (Suma)



Argumenty:

arg1 – analogowy argument 1

arg2 – analogowy argument 2

Zwracane wartości:

res – suma *arg1* i *arg2* (*out* = *arg1* + *arg2*)

Opis:

Funkcja zwraca sumę argumentów *arg1* i *arg2*.

4.4. Difference (Różnica)

Argumenty:

arg1 – analogowy argument 1

arg2 – analogowy argument 2

Zwracane wartości:

res – różnica *arg1* i *arg2* (*out* = *arg1* – *arg2*)

Opis:

Funkcja zwraca różnice argumentów *arg1* i *arg2*.

4.5. Multiplication (Mnożenie)

Argumenty:

arg1 – analogowy argument 1 (format: *Fixed*)

arg2 – analogowy argument 2 (format: *Fixed*)

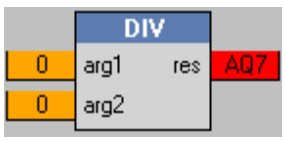
Zwracane wartości:

res – iloczyn *arg1* i *arg2* (*out* = *arg1* * *arg2*) (format: *Fixed*)

Opis:

Funkcja zwraca iloczyn argumentów *arg1* i *arg2*. Obliczenia dokonywane są przy założeniu że argumenty są w formacie *Fixed*.

4.6. Division (Dzielenie)



Argumenty:

arg1 – analogowy argument 1 (format: *Fixed*)

arg2 – analogowy argument 2 (format: *Fixed*)

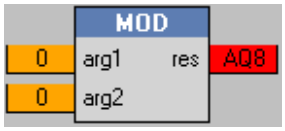
Zwracane wartości:

res – iloraz *arg1* i *arg2* (*out* = *arg1* / *arg2*) (format: *Fixed*)

Opis:

Funkcja zwraca iloraz argumentów *arg1* i *arg2*. Obliczenia dokonywane są przy założeniu że argumenty są w formacie *Fixed*. Jeżeli *arg2* = 0 (dzielenie przez 0) to *res* = 0.

4.7. Modulo (Dzielenie modulo)



Argumenty:

arg1 – analogowy argument 1

arg2 – analogowy argument 2

Zwracane wartości:

res – iloraz modulo *arg1* i *arg2* (*out* = *arg1* mod *arg2*)

Opis:

Funkcja zwraca iloraz modulo argumentów *arg1* i *arg2*.

4.8. Maximum

Argumenty:

arg1 – analogowy argument 1

arg2 – analogowy argument 2

Zwracane wartości:

res – maksimum z *arg1* i *arg2* (*out* = max {*arg1*, *arg2*})

Opis:

Funkcja zwraca maksymalną wartość z argumentów *arg1* i *arg2*.

4.9. Minimum

Argumenty:

arg1 – analogowy argument 1

arg2 – analogowy argument 2

Zwracane wartości:

res – minimum z *arg1* i *arg2* (*out* = min {*arg1*, *arg2*})

Opis:

Funkcja zwraca minimalną wartość z argumentów *arg1* i *arg2*.

4.10. Absolute (Wartość bezwzględna)

Argumenty:

in – analogowy argument

Zwracane wartości:

out – wartość bezwzględna argumentu *in*

Opis:

Funkcja zwraca dodatnią wartość argumentu *in*. Gdy $in < 0$ to $out = -in$, gdy $in \geq 0$ to $out = in$.

4.11. Square root (Pierwiastek kwadratowy)

Argumenty:

res – analogowy argument dla pierwiastkowania (format: *Fixed*)

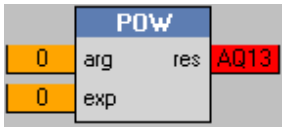
Zwracane wartości:

out – pierwiastek dla argumentu *arg* (format: *Fixed*)

Opis:

Funkcja zwraca pierwiastek kwadratowy dla argumentu *arg*.

4.12. Power (Potęga)



Argumenty:

arg – analogowy argument potęgowania (format: *Fixed*)

exp – analogowy wykładnik potęgowania (format: *Fixed*)

Zwracane wartości:

res – potęga *arg* o wykładniku *exp* (format: *Fixed*)

Opis:

Funkcja zwraca potęgę argumentu *arg* o wykładniku *exp*.

4.13. Natural logarithm (Logarytm naturalny)

Argumenty:

arg – analogowy argument logarytmowania (format: *Fixed*)

Zwracane wartości:

res – logarytm naturalny dla argumentu *arg* (format: *Fixed*)

Opis:

Funkcja zwraca logarytm naturalny dla argumentu *arg*.

4.14. Logarithm 10 (Logarytm dziesiętny)

Argumenty:

arg – analogowy argument logarytmowania dziesiętnego (format: *Fixed*)

Zwracane wartości:

res – logarytm dziesiętny dla argumentu *arg* (format: *Fixed*)

Opis:

Funkcja zwraca logarytm dziesiętny dla argumentu *arg*.

4.15. Exponent e (Potęga liczby Eulera)

Argumenty:

arg – analogowy argument potęgowania liczby Eulera (format: *Fixed*)

Zwracane wartości:

res – wynik potęgowania liczby Eulera dla argumentu *arg* (format: *Fixed*)

Opis:

Funkcja zwraca potęgę liczby Eulera dla wykładnika *arg*.

4.16. Sinus

Argumenty:

arg – kąt w radianach (format: *Fixed*)

Zwracane wartości:

res – sinus kąta *arg* (format: *Fixed*)

Opis:

Funkcja zwraca sinus kąta *arg* podanego w radianach.

4.17. Cosinus

Argumenty:

arg – kąt w radianach (format: *Fixed*)

Zwracane wartości:

res – cosinus kąta *arg* (format: *Fixed*)

Opis:

Funkcja zwraca cosinus kąta *arg* podanego w radianach.

4.18. Tangens

Argumenty:

arg – kąt w radianach (format: *Fixed*)

Zwracane wartości:

res – tangens kąta *arg* (format: *Fixed*)

Opis:

Funkcja zwraca tangens kąta *arg* podanego w radianach.

4.19. Arcus sinus

Argumenty:

arg – wartość sinusa (format: *Fixed*)

Zwracane wartości:

res – kąt w radianach dla wartości sinusa *arg* (format: *Fixed*)

Opis:

Funkcja zwraca arcus sinus w radianach dla wartości sinusa *arg*.

4.20. Arcus cosinus

Argumenty:

arg – wartość cosinusa (format: *Fixed*)

Zwracane wartości:

res – kąt w radianach dla wartości cosinusa *arg* (format: *Fixed*)

Opis:

Funkcja zwraca arcus cosinus w radianach dla wartości cosinusa *arg*.

4.21. Arcus tangens

Argumenty:

arg – wartość tangensa (format: *Fixed*)

Zwracane wartości:

res – kąt w radianach dla wartości tangensa *arg* (format: *Fixed*)

Opis:

Funkcja zwraca arcus tangens w radianach dla wartości tangensa *arg*.

4.22. Deg2Rad (Konwersja stopni na radiany)

Argumenty:

arg – wartość kąta w stopniach (format: *Fixed*)

Zwracane wartości:

res – wartość kąt w radianach odpowiadająca wartości kąta *arg* w stopniach (format: *Fixed*)

Opis:

Funkcja dokonuje konwersji stopni na radiany.

4.23. Rad2Deg (Konwersja radianów na stopnie)

Argumenty:

arg – wartość kąta w radianach (format: *Fixed*)

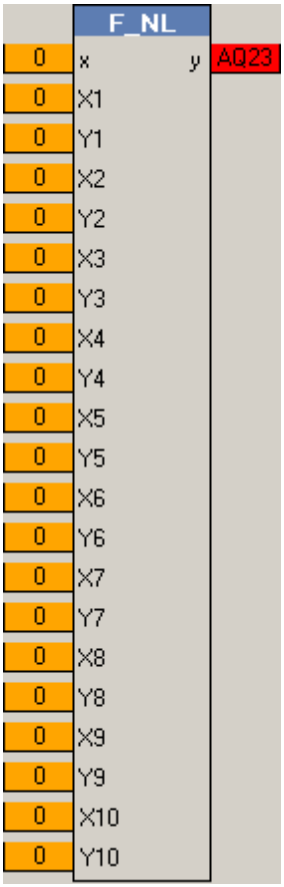
Zwracane wartości:

res – wartość kąt w stopniach odpowiadająca wartości kąta *arg* w radianach (format: *Fixed*)

Opis:

Funkcja dokonuje konwersji radianów na stopnie.

4.24. Non-linear func 10 (10-punktowa nieliniowa funkcja)



Argumenty:

x – argument funkcji dwuwymiarowej

X1-10 – odcięta dla punktów charakteryzujących funkcje

Y1-10 – rzędna dla punktów charakteryzujących funkcje

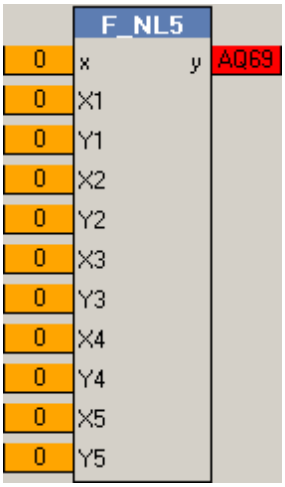
Zwracane wartości:

y – wartość funkcji dwuwymiarowej dla argumentu *x*

Opis:

Funkcja opisana przez 10 punktów P(*X*n,*Y*n) gdzie n = 1,2,...,10. Dla argumentu *x* zwracana jest wartość funkcji *y*. Funkcja jest interpolowana liniowo w kolejnych przedziałach punktów charakteryzujących funkcje.

4.25. Non-linear func 5 (5-punktowa nieliniowa funkcja)



Argumenty:

x – argument funkcji dwuwymiarowej

X1-5 – odcięta dla punktów charakteryzujących funkcje

Y1-5 – rzędna dla punktów charakteryzujących funkcje

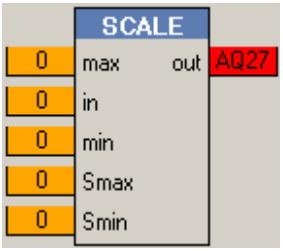
Zwracane wartości:

y – wartość funkcji dwuwymiarowej dla argumentu *x*

Opis:

Funkcja opisana przez 5 punktów P(*X*n,*Y*n) gdzie n = 1,2,...,5. Dla argumentu *x* zwracana jest wartość funkcji *y*. Funkcja jest interpolowana liniowo w kolejnych przedziałach punktów charakteryzujących funkcje.

4.26. Scale (Skalowanie)



Argumenty:

max – ograniczenie górne wejścia analogowego *in* (*max* > *min*)

in – wejście analogowe.

min – ograniczenie dolne wejścia analogowego *in* (*max* > *min*)

Smax – górny zakres przedziału skalowania (*Smax* > *Smin*)

Smin – dolny zakres przedziału skalowania (*Smax* > *Smin*)

Zwracane wartości:

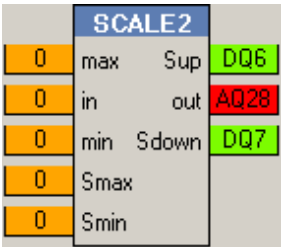
out – przeskalowana wartość *in*

Opis:

Funkcja skaluje wartość z wejścia *in* w taki sposób że zakres <*Smin*, *Smax*> odpowiada zakresowi <*min*, *max*>. Jeżeli wartość na wejściu *in* jest spoza <*min*, *max*> to jest przycinana do wartości brzegowych zakresu.

out = (*in* – *min*) * (*Smax* – *Smin*) / (*max* – *min*) + *Smin*

4.27. Scale2 (Skalowanie wersja 2)



Argumenty:

max – ograniczenie górne wejścia analogowego *in* (*max* > *min*)

in – wejście analogowe

min – ograniczenie dolne wejścia analogowego *in* (*max* > *min*)

Smax – górny zakres przedziału skalowania (*Smax* > *Smin*)

Smin – dolny zakres przedziału skalowania (*Smax* > *Smin*)

Zwracane wartości:

Sup - sygnalizacja przekroczenia górnego ograniczenia przez wartość wejściową (*in* > *max*)

out - przeskalowana wartość *in*

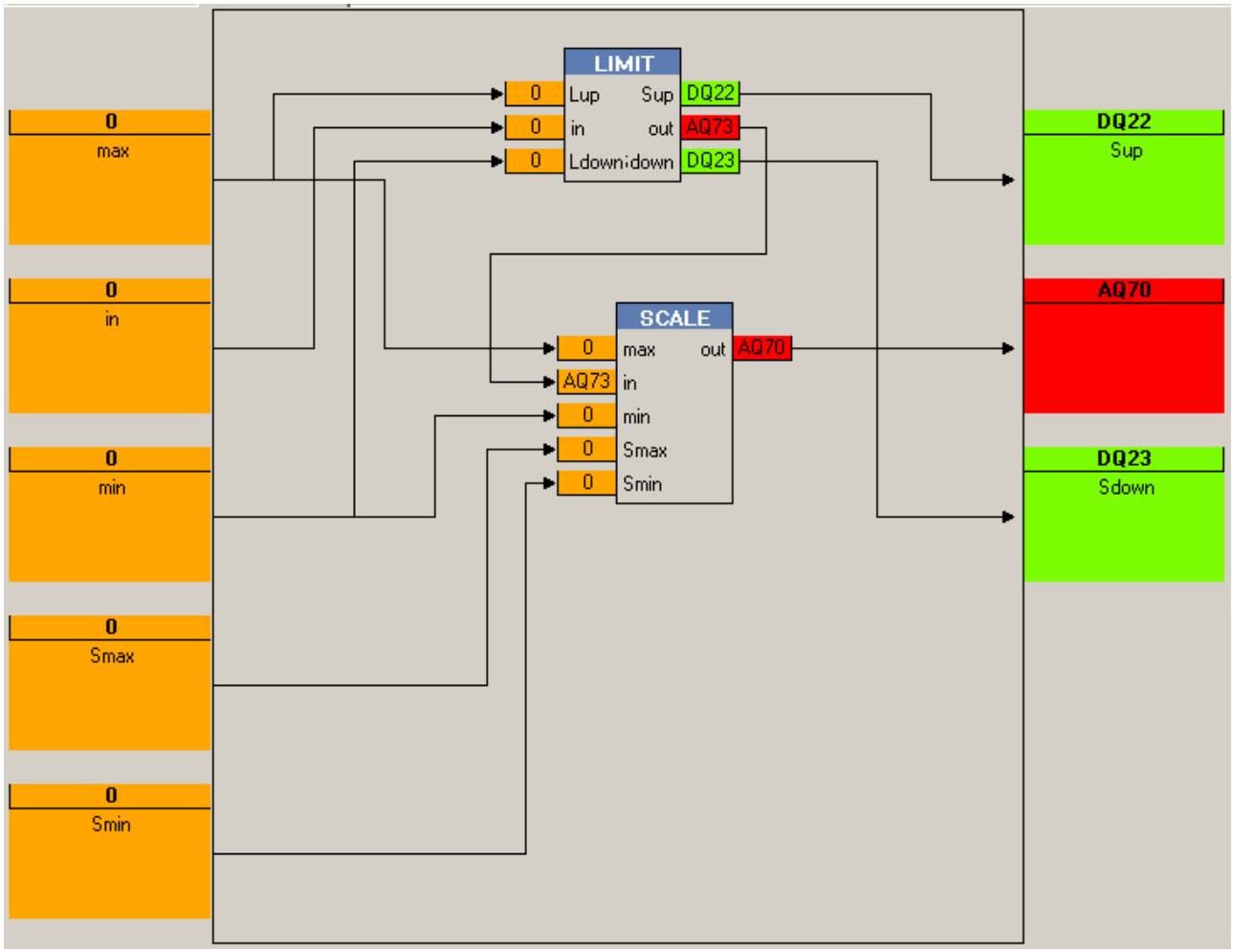
Sdown - sygnalizacja przekroczenia dolnego ograniczenia przez wartość wejściową (*in* < *min*)

Opis:

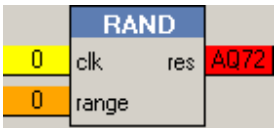
Funkcja skaluje wartość z wejścia *in* w taki sposób że zakres <*Smin*, *Smax*> odpowiada zakresowi <*min*, *max*>. Jeżeli wartość na wejściu *in* jest spoza <*min*, *max*> to jest przycinana do wartości brzegowych zakresu. Dodatkowo funkcja informuje o przekroczeniu zakresu przez wartość na wejściu *in*.

*out = (in – min) * (Smax – Smin) / (max – min) + Smin*

Działanie bloku funkcyjnego SCALE2 można porównać do makra składającego się z bloków LIMIT oraz SCALE



4.28. Random (Wartość losowa)



Argumenty:

clk – zbocze narastające generuje losową wartość na wyjściu *res*

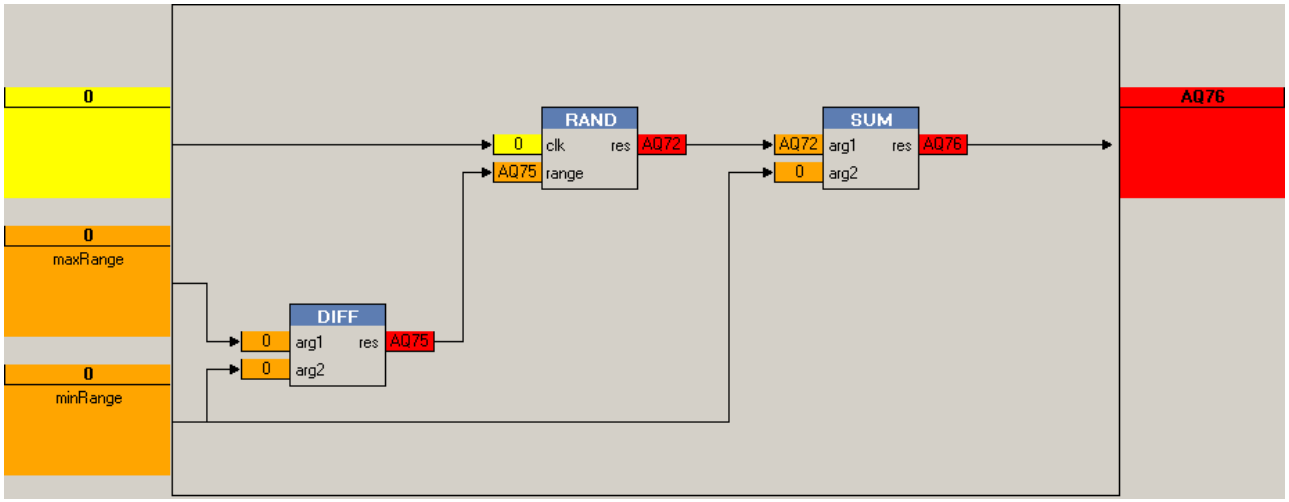
range – zakres generowanej liczby od 0 włącznie

Zwracane wartości:

res – wygenerowana wartość losowa z przedziału <0, *range*)

Opis:

Funkcja generuje pseudo-losową wartość *res* ≥ 0 i *res* < *range*. W celu wygenerowania liczby losowej z dowolnego zakresu można np. zbudować makro


0
0
maxRange
0
minRange
DIFF
0
arg1
res
AQ75
0
arg2
RAND
0
clk
res
AQ72
AQ75
range
SUM
AQ72
arg1
res
AQ76
0
arg2
AQ76

5. Triggers

5.1. Trigger (Przerzutnik)

Argumenty:

clk – impuls zegarowy (wyzwalanie zboczem narastającym)

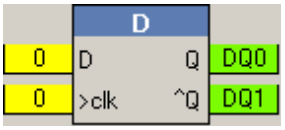
Zwracane wartości:

Q – stan przerzutnika

Opis:

Prosty przerzutnik. Gdy na wejściu *clk* pojawia się zbocze narastające wyjście *Q* przyjmuje wartość 1 przez jeden cykl algorytmu sterowania, a następnie automatycznie je kasuje.

5.2. D trigger (Przerzutnik D)



Argumenty:

D – argument przerzutnika

clk – impuls zegarowy (wyzwalanie zboczem narastającym)

Zwracane wartości:

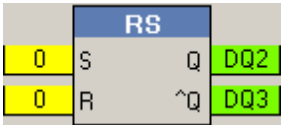
Q - stan przerzutnika

^*Q* - negacja stanu przerzutnika

Opis:

Blok realizuje funkcję przerzutnika D. Gdy na *clk* pojawi się zbocze narastające wartość z wejścia *D* jest zapamiętywana w przerzutniku i można ją odczytać na wyjściu *Q* lub jej negacje na wyjściu ^*Q*.

5.3. RS trigger (Asynchroniczny przerzutnik z priorytetem kasowania)



Argumenty:

S – argument przerzutnika

R – sygnał resetu przerzutnika

Zwracane wartości:

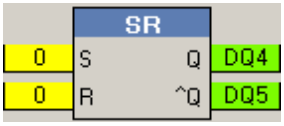
Q – stan przerzutnika

^*Q* – negacja stanu przerzutnika

Opis:

Blok realizuje funkcję przerzutnika RS. Jeżeli na wejściu *S* pojawi się wartość 1 zostanie zapamiętywana w przerzutniku i można ją odczytać na wyjściu *Q* lub jej negację na wyjściu ^*Q*. Zerowanie przerzutnika następuje po ustawieniu wejścia *R*. Wejście *R* ma priorytet wyższy niż wejście *S*.

5.4. SR trigger (Asynchroniczny przerzutnik SR z priorytetem ustawiania)



Argumenty:

S – argument przerzutnika

R – sygnał resetu przerzutnika

Zwracane wartości:

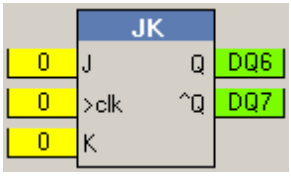
Q – stan przerzutnika

^*Q* – negacja stanu przerzutnika

Opis:

Blok realizuje funkcję przerzutnika SR. Gdy na *S* pojawi się wartość 1 jest ona zapamiętywana w przerzutniku i można ją odczytać na wyjściu *Q* lub jej negację na wyjściu ^*Q*. Zerowanie przerzutnika następuje po ustawieniu wejścia *R*. Wejście *S* ma priorytet wyższy niż wejście *R*.

5.5. JK trigger (Przerzutnik JK)



Argumenty:

J – argument J przerzutnika

clk – impuls zegarowy (wyzwalanie zboczem narastającym)

K – argument K przerzutnika

Zwracane wartości:

Q – stan przerzutnika

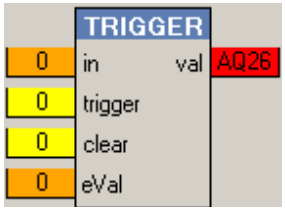
^*Q* – negacja stanu przerzutnika

Opis:

Blok realizuje funkcję przerzutnika JK. Zmiana stanu przerzutnika następuje po podaniu zbocza narastającego na wejście zegarowe *clk*. Przerzutnik działa według tabeli:

J	*K*	*Q*(n)	*Q*(n+1)	Opis
0	0	0 lub 1	0 lub 1	podtrzymanie
0	1	0 lub 1	0	przejście na 0
1	0	0 lub 1	1	przejście na 1
1	1	0 lub 1	1 lub 0	~Q (negacja)

5.6. Analog Trigger (Przerzutnik analogowy)



Argumenty:

in – argument analogowy

trigger – wyzwolenie zapamiętania wartości *in* (zbocze narastające)

clear – wyzwolenie skasowania ustawionej wartości i zapisania *eVal* (zbocze narastające)

eVal – wartość domyślna przerzutnika analogowego

Zwracane wartości:

val – aktualny stan przerzutnika analogowego

Opis:

Realizuje funkcję przerzutnika analogowego. W momencie pojawienia się zbocza narastającego na wejściu *trigger* w przerzutniku zapamiętywana jest wartość z wejścia *in*. Kasowanie odbywa się poprzez zbocze narastające na wejściu *clear*, wtedy przerzutnik przyjmuje wartość domyślną *eVal*.

6. Timeline

6.1. Delay 1 (Opóźnienie stanu 1)

Argumenty:

in – argument cyfrowy (reakcja na zbocze narastające)

delay – opóźnienie w sekundach (format: *Fixed*)

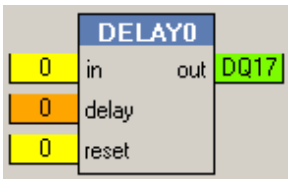
Zwracane wartości:

out – opóźnienie wartości 1 z wejścia *in* po czasie *delay*

Opis:

Funkcja opóźnia stan 1 wejścia *in* o czas *delay*. Jeżeli na wejściu *in* pojawi się stan 1 to na wyjściu *out* ten stan zostanie ustawiony po czasie *delay*. Wartość 0 na wejściu *in* powoduje natychmiastowe wyzerowanie wyjścia *out*.

6.2. Delay 0 (Opóźnienie stanu 0)



Argumenty:

in – argument cyfrowy (reakcja na zbocze opadające)

delay – opóźnienie w sekundach (format: *Fixed*)

reset – kasowanie opóźnienia (zbocze narastające) i ustawienie wartości 0 na wyjściu *out*

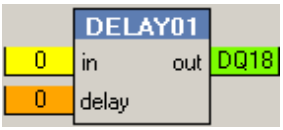
Zwracane wartości:

out – opóźnienie wartości 0 z wejścia *in* po czasie *delay*

Opis:

Funkcja opóźnia stan 0 wejścia *in* o czas *delay*. Jeżeli na wejściu *in* pojawi się stan 0 to na wyjściu *out* ten stan zostanie ustawiony po czasie *delay*. Wartość 1 na wejściu *in* powoduje natychmiastowe ustawienie wyjścia *out*. Podanie zbocza narastającego na wejście *reset* ustawia wartości 0 na wyjściu *out* (bez opóźnienia). Przy pierwszym wywołaniu funkcji (start algorytmu) dla argumentu *in* = 0 na wyjściu będzie obecny stan 0.

6.3. Delay 01 (Opóźnienie 0 i 1)



Argumenty:

in – argument cyfrowy (reakcja na dowolne zbocze)

delay – opóźnienie w sekundach (format: *Fixed*)

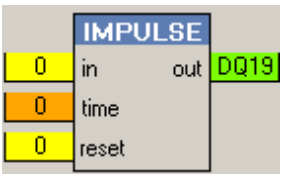
Zwracane wartości:

out – wartość z wejścia *in* po czasie *delay*.

Opis:

Funkcja opóźniająca stan z wejścia *in* o czas *delay*. Jeżeli na wejściu *in* pojawi się stan inny niż wcześniej to zostanie on przeniesiony na wyjście *out* po czasie *delay*. Przy pierwszym wywołaniu funkcji (start algorytmu) na wyjściu będzie obecny stan 0.

6.4. Impulse (Impuls)



Argumenty:

in – argument cyfrowy (reakcja na zbocze narastające)

time – czas trwania impulsu w sekundach (format: *Fixed*)

reset – kasowanie opóźnienia (zbocze narastające) i ustawienie wartości 0 na wyjściu *out*

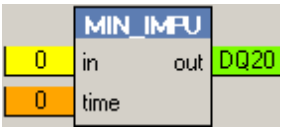
Zwracane wartości:

out – impuls

Opis:

Funkcja generuje impuls o stałym czasie trwania. Po pojawieniu się na wejściu *in* stanu 1 na wyjściu *out* będzie utrzymany stan 1 przez czas *time* nawet gdy na wejściu *in* pojawi się w międzyczasie stan 0. Po upływie zadanego czasu na wyjściu *out* pojawi się stan 0 nawet mimo że na wejściu *in* utrzymany jest stan 1. Podanie zbocza narastającego na wejście *reset* spowoduje natychmiastowe ustawienie wartości 0 na wyjściu, impuls zostaje przerwany i kolejne jego wywołanie wymaga zbocza narastającego na wejściu *in*.

6.5. Minimum impulse (Minimalny impuls)



Argumenty:

in – argument cyfrowy

time – czas trwania impulsu w sekundach (format: *Fixed*)

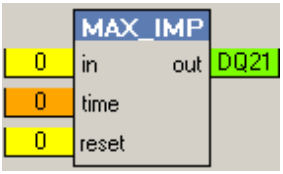
Zwracane wartości:

out – impuls

Opis:

Funkcja generuje impuls o czasie trwania *time* lub krótszym. Po pojawieniu się na wejściu *in* stanu 1 na wyjściu *out* zostanie ustawiony stan 1 i utrzymany przez czas *time* lub, jeżeli w międzyczasie na wejściu *in* pojawi się stan 0, impuls zostanie przerwany i wyjście *out* przyjmie stan 0. Po upływie zadanego czasu na wyjściu *out* pojawi się stan 0 nawet mimo że na wejściu *in* utrzymany jest stan 1.

6.6. Maximum impulse (Maksymalny impuls)



Argumenty:

in – argument cyfrowy

time – czas trwania impulsu w sekundach (format: *Fixed*)

reset – kasowanie opóźnienia (zbocze narastające) i ustawienie wartości 0 na wyjściu *out*.

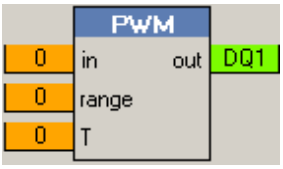
Zwracane wartości:

out – impuls

Opis:

Funkcja generuje impuls o czasie trwania *time* lub dłuższym. Po pojawieniu się na wejściu *in* stanu 1 na wyjściu *out* zostanie ustawiony stan 1 i utrzymany przez czas *time* lub, jeżeli w międzyczasie na wejściu *reset* pojawi się zbocze narastające, impuls zostanie przerwany i wyjście *out* przyjmie stan 0. Po upływie zadanego czasu, jeżeli na wejściu *in* nadal jest obecny stan 1, to wyjście *out* utrzyma stan 1 do chwili pojawienia się stanu 0 na wejściu *in*.

6.7. PWM (Modulator długości impulsu)



Argumenty:

in – wejście sterujące (wartości z zakresu 0 – *range*)

range – maksymalna wartość wejścia *in* oznaczająca 100% wysterowania PWM

T – czas trwania okresu PWM w sekundach (format: *Fixed*)

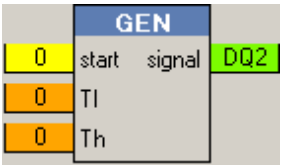
Zwracane wartości:

out – impuls PWM

Opis:

Funkcja PWM (Pulse Width Modulation). Na wyjściu generowane są impulsy o okresie *T* i wypełnieniu w zakresie 0 – 100% skalowanym według aktualnego stanu wejścia *in*, w zakresie od 0 do *range*.

6.8. Generator



Argumenty:

start – stan 1 uruchamia generator

Tl – czas trwania stanu niskiego na wyjściu *signal* w sekundach (format: *Fixed*)

Th – czas trwania stanu wysokiego na wyjściu *signal* w sekundach (format: *Fixed*)

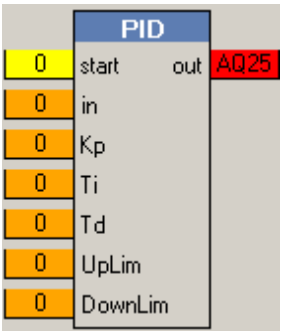
Zwracane wartości:

signal – generowany sygnał

Opis:

Funkcja generuje na wyjściu *signal* impulsy których stan niski utrzymany jest przez czas *Tl*, a stan wysoki przez czas *Th*. Po podaniu stanu 1 na wejście *start* generator rozpoczyna pracę od stanu niskiego.

6.9. PID regulator (Regulator proporcjonalno – różniczkująco – całkujący)



Argumenty:

start – uruchomienie regulatora PID

in – wejście sterujące regulatora (odchyłka)

Kp – współczynnik wzmocnienia (format: *Fixed*)

Ti – współczynnik całkowania w sekundach (format: *Fixed*)

Td – współczynnik różniczkowania w sekundach (format: *Fixed*)

UpLim – górne ograniczenie wartości wyjściowej

DownLim – dolne ograniczenie wartości wyjściowej

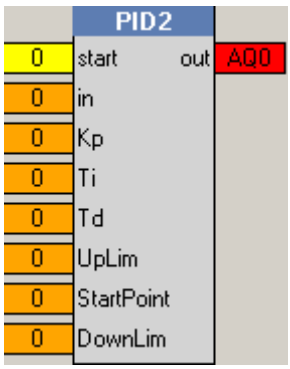
Zwracane wartości:

out – sygnał regulatora PID

Opis:

Funkcja realizuje regulację PID według zadanych parametrów. Z chwilą ustawienia stanu 1 na wejściu *start* regulator rozpoczyna pracę od zera. Regulator próbuje zniwelować uchyb z wejścia *in*. Gdy wyjście *out* osiągnie limit ustalony przez *UpLim* i *DownLim* regulator wstrzymuje całkowanie i różniczkowanie na ostatnio ustalonym poziomie.

6.10. PID2 regulator (Regulator proporcjonalno – różniczkująco – całkujący, wersja 2)



Argumenty:

start – uruchomienie regulatora PID

in – wejście sterujące regulatora (odchyłka)

Kp – współczynnik wzmocnienia (format: *Fixed*)

Ti – współczynnik całkowania w sekundach (format: *Fixed*)

Td – współczynnik różniczkowania w sekundach (format: *Fixed*)

UpLim – górne ograniczenie wartości wyjściowej

StartPoint – początek regulacji

DownLim – dolne ograniczenie wartości wyjściowej

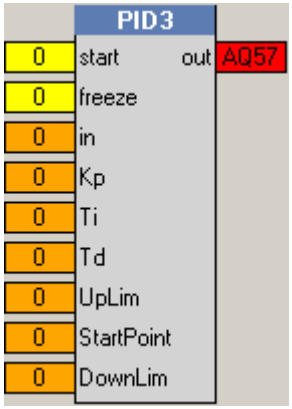
Zwracane wartości:

out – sygnał regulatora PID

Opis:

Funkcja realizuje regulację PID według zadanych parametrów. Gdy wejście *start* = 0 wyjście regulatora wskazuje wartość *StartPoint*. Z chwilą ustawienia stanu 1 na wejściu *start* regulator rozpoczyna pracę od wartości *StartPoint*. Regulator próbuje zniwelować uchyb z wejścia *in*. Gdy wyjście *out* osiągnie limit ustalony przez *UpLim* i *DownLim* regulator wstrzymuje całkowanie i różniczkowanie na ostatnio ustalonym poziomie.

6.11. PID3 regulator (Regulator proporcjonalno – różniczkująco – całkujący, wersja 3)



Argumenty:

start – uruchomienie regulatora PID

freeze – zablokowanie regulatora

in – wejście sterujące regulatora (odchyłka)

Kp – współczynnik wzmocnienia (format: *Fixed*)

Ti – współczynnik całkowania w sekundach (format: *Fixed*)

Td – współczynnik różniczkowania w sekundach (format: *Fixed*)

UpLim – górne ograniczenie wartości wyjściowej

StartPoint – początek regulacji

DownLim – dolne ograniczenie wartości wyjściowej

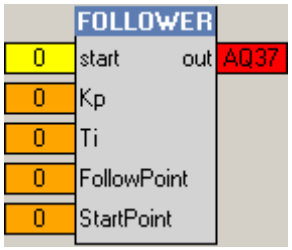
Zwracane wartości:

out – sygnał regulatora PID

Opis:

Funkcja realizuje regulację PID według zadanych parametrów. Gdy wejście *start* = 0 wyjście regulatora wskazuje wartość *StartPoint*. Z chwilą ustawienia stanu 1 na wejściu *start* regulator rozpoczyna pracę od wartości *StartPoint*. Regulator próbuje zniwelować uchyb z wejścia *in*. Gdy wyjście *out* osiągnie limit ustalony przez *UpLim* i *DownLim* regulator wstrzymuje całkowanie i różniczkowanie na ostatnio ustalonym poziomie. Gdy wejście *freeze* = 1 to regulator wstrzymuje swoją pracę, a wyjście *out* zachowuje ostatnią wartość.

6.12. Follower (Funkcja nadążająca)



Argumenty:

start – uruchomienie działania bloku

Kp – współczynnik wzmocnienia (format: *Fixed*)

Ti – współczynnik całkowania w sekundach (format: *Fixed*)

FollowPoint – punkt do którego funkcja ma dążyć

StartPoint – początek regulacji

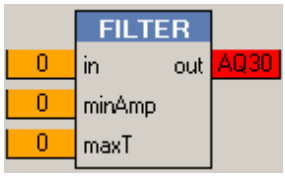
Zwracane wartości:

out – aktualny stan funkcji nadążającej w przedziale <*StartPoint*, *FollowPoint*>.

Opis:

Gdy wejście *start* = 0 wyjście *out* = *StartPoint*. W momencie gdy *start* = 1 funkcja nadąża za ustaloną wartością *FollowPoint* według ustalonych parametrów *Kp* i *Ti* tak jak w przypadku regulatora PI.

6.13. Filter (Filtr)



Argumenty:

in – wartość analogowa

minAmp – zakres amplitudy zmian sygnału wejściowego

maxT – czas trwania zmiany sygnału wejściowego w zakresie *minAmp* w sekundach (format: *Fixed*)

Zwracane wartości:

out – filtrowany sygnał wejściowy

Opis:

Funkcja implementuje prosty filtr sygnału analogowego. Filtr nie przepuszcza zmian sygnału analogowego poniżej ±*minAmp* w stosunku do wyjścia *out* i czasie trwania krótszym niż *maxT*. Jeżeli zmiana sygnału jest większa od *minAmp* lub czas jej trwania przekracza *maxT* zostaje ona uwzględniona i stan wejścia *in* zostanie przepisany na wyjście *out*.

6.14. Average (Średnia)

Argumenty:

in – wartość analogowa

span – łączny czas uśredniania ośmiu próbek (format: *Fixed*)

Zwracane wartości:

res – średnia

Opis:

Funkcja wyznacza średnią z ośmiu próbek zbieranych z wejścia *in* przez okres *span*. Każda próbka jest zapisywana po czasie *span* / 8.

6.15. System time (Czas systemowy)

Zwracane wartości:

time – aktualny czas systemowy w sekundach od początku uruchomienia sterownika PLC (włączenia zasilania) (format: *Fixed*).

Opis:

Funkcja podaje aktualny czas jaki upłynął od momentu włączenia urządzenia. Przydatne dla operacji związanych z czasem.

6.16. Time interval (Odstęp czasu)

Argumenty:

old_t – czas w sekundach (format: *Fixed*)

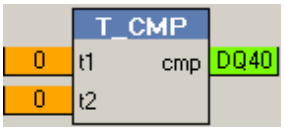
Zwracane wartości:

intrv – odstęp czasowy w sekundach pomiędzy aktualnym czasem systemowym (jaki upłynął od momentu uruchomienia sterownika PLC), a wartością na wejściu *old_t* (format: *Fixed*).

Opis:

Funkcja podaje odstęp pomiędzy aktualnym czasem systemowym, a wartością czasu z wejścia *old_t*.

6.17. Time compare (Porównanie czasów)



Argumenty:

t1 – czas 1 do porównania w sekundach (format: *Fixed*)

t2 – czas 2 do porównania w sekundach (format: *Fixed*)

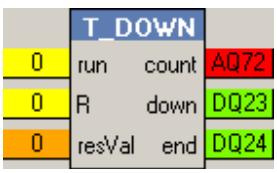
Zwracane wartości:

cmp – informacja czy wartość czasu *t1* jest mniejsza od *t2*. Jeżeli tak to *cmp* = 1, w przeciwnym wypadku *cmp* = 0

Opis:

Funkcja porównuje dwie wartości czasu. Czas obecny w systemie jest wartością 32-bitową w której 24 bity przeznaczone są na pomiar sekund. Oznacza to że po 194 dniach licznik czasu „przekręci się" i zacznie liczyć od nowa. Z tego względu wszystkie porównania czasów systemowych należy dokonywać tą funkcją gdyż uwzględnia ona ewentualność przepełnienia licznika czasu systemowego.

6.18. Time downcounter (Timer)



Argumenty:

run – uruchomienie timera

t2 – kasowanie timera (zbocze narastające) i ustawienie wartości początkowej *resVal*

resVal – wartość początkowa timera w sekundach (format: *Fixed*)

Zwracane wartości:

count – aktualny stan licznika timera (zmienia się co sekundę)

down – impuls pojawiający się co sekundę odliczania timera na jeden cykl algorytmu sterowania

end – impuls pojawiający się po zakończeniu odliczania na jeden cykl algorytmu sterowania

Opis:

Timer zlicza czas w dół do zera począwszy od wartości *resVal*. Zliczanie trwa tylko gdy na wejściu *run* jest stan 1. Kasowanie timera odbywa się przez podanie zbocza narastającego na wejście *R*. Aktualny stan licznika timera jest na wyjściu *count*. Dodatkowo wyjście *down* co sekundę działania timera generuje impuls trwający jeden cykl algorytmu sterowania. Gdy licznik osiągnie wartość 0 na wyjściu *end* pojawi się impuls trwający jeden cykl algorytmu sterowania i timer zakończy działanie.

6.19. Max24 (Maksymalna wartość z ostatnich 24h)

Argumenty:

in – wartość monitorowana

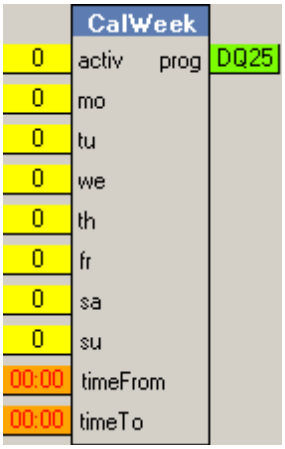
Zwracane wartości:

max – maksymalna wartość z wejścia *in*, która wystąpiła w ciągu ostatnich 24h.

Opis:

Funkcja monitoruje stan wejścia *in* i zapisuje maksymalną osiągniętą wartość z ostatniej godziny. Przechowywane są 24 próbki maksymalne z ostatnich pełnych godzin. Wartość *max* obliczana jest jako maksymalna wartość z 24 zapamiętanych maksymalnych próbek z każdej pełnej godziny.

6.20. Calendar Week (Kalendarz tygodniowy)



Argumenty:

ativ – aktywacja działania bloku kalendarza tygodniowego

mo, tu, we, th, fr, sa, su – aktywacja programu dla odpowiednich dni tygodnia

timeFrom – czas rozpoczęcia działania programu (format: *Time hh:mm.ss*)

timeTo – czas zakończenia działania programu (format: *Time hh:mm.ss*)

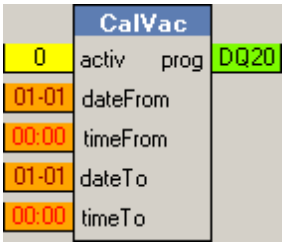
Zwracane wartości:

prog – informacja o działaniu programu kalendarza

Opis:

Blok kalendarza tygodniowego bada czy aktualny czas i dzień tygodnia zegara RTC sterownika PLC spełnia warunki programu. Jeżeli zegar RTC wskazuje dzień tygodnia zaznaczony na wejściu *mo – su* i aktualny czas *T* spełnia warunek $timerFrom \leq T < timeTo$ to wyjście *prog* przyjmie stan 1.

6.21. Calendar Vacation (Kalendarz wakacji)



Argumenty:

ativ – aktywacja działania bloku kalendarza

dateFrom – data rozpoczęcia wakacji (format: *Date DD-MM-YY*)

timeFrom – czas rozpoczęcia wakacji (format: *Time hh:mm.ss*)

dateTo – data zakończenia wakacji (format: *Date DD-MM-YY*)

timeTo – czas zakończenia wakacji (format: *Time hh:mm.ss*)

Zwracane wartości:

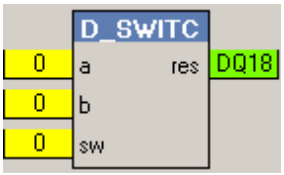
prog – informacja o działaniu programu kalendarza

Opis:

Blok kalendarza wakacji bada czy aktualny czas i data zegara RTC sterownika PLC zawiera się w przedziale ustalonym jako okres wakacji to wyjście *prog* przyjmie stan 1. W praktyce funkcja działa jednorazowo, a po okresie wakacji, wymaga ponownego nastawienia na następny rok.

7. Multiplex

7.1. Digital Switcher (Przełącznik cyfrowy)



Argumenty:

a – argument cyfrowy 1

b – argument cyfrowy 2

sw – przełącznik wejść

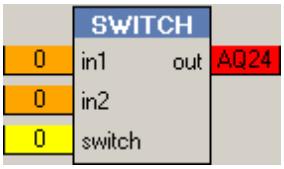
Zwracane wartości:

out – przeniesiony sygnał z odpowiedniego wejścia według stanu wejścia *sw*

Opis:

Przełącznik cyfrowy. Przełącza sygnał *a* i *b* na odpowiednie wyjście *res* w taki sposób że: jeżeli *sw* = 0 to *res* = *a*, w przeciwnym wypadku *res* = *b*.

7.2. Analog Switcher (Przełącznik analogowy)



Argumenty:

in1 – wejście analogowe 1

in2 – wejście analogowe 2

switch – przełącznik wejść

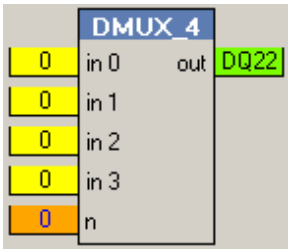
Zwracane wartości:

out - przeniesiony sygnał z odpowiedniego wejścia według stanu wejścia *switch*

Opis:

Przełącznik analogowy. Przełącza sygnał *in1* i *in2* na odpowiednie wyjście *out* w taki sposób że: jeżeli *switch* = 0 to *out* = *in1*, w przeciwnym wypadku *out* = *in2*.

7.3. Digital Multiplexer 4Ch (Multiplekser cyfrowy 4-kanałowy)



Argumenty:

in0-3 – argumenty cyfrowe

n – numer wejścia które ma być przełączone na wyjście *out* (wartości z zakresu 0 – 3 format: *Decimal* lub *Hex*)

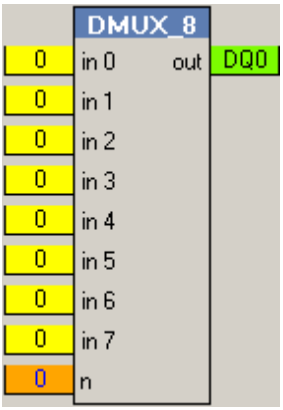
Zwracane wartości:

out – wartość z wejścia *in**n***

Opis:

Realizuje funkcję 4-kanałowego multipleksera cyfrowego. Wyjście *out* = *in**n*** gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*. Gdy *n* jest spoza zakresu to *out* = 0.

7.4. Digital Multiplexer 8Ch (Multiplekser cyfrowy 8-kanałowy)



Argumenty:

in 0-7 – argumenty cyfrowe

n – numer wejścia które ma być przełączone na wyjście *out* (wartości z zakresu 0 – 7 format: *Decimal* lub *Hex*)

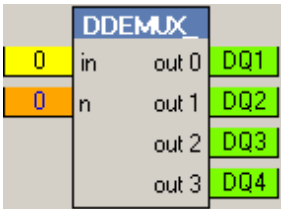
Zwracane wartości:

out - wartość z wejścia *in**n***

Opis:

Realizuje funkcję 8-kanałowego multipleksera cyfrowego. Wyjście *out* = *in**n*** gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*. Gdy *n* jest spoza zakresu to *out* = 0.

7.5. Digital Demultiplexer 4Ch (Demultiplekser 4-kanałowy)



Argumenty:

in – argument cyfrowy

n – numer wyjścia na które ma być przełączone wejście *in* (wartości z zakresu 0 – 3 format: *Decimal* lub *Hex*)

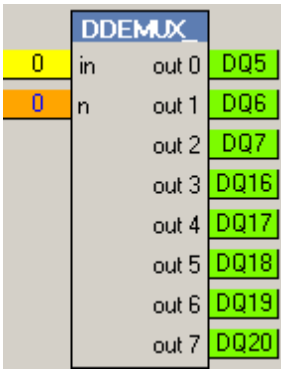
Zwracane wartości:

out 0-3 – na wyjściu *out**n*** jest obecna wartość z wejścia *in*, na pozostałych 0

Opis:

Realizuje funkcję 4-kanałowego demultipleksera cyfrowego. Wyjście *out**n*** = *in* gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*, na pozostałych wyjściach jest stan 0. Jeżeli *n* jest spoza zakresu to wszystkie wyjścia przyjmują stan 0.

7.6. Digital Demultiplexer 8Ch (Demultiplekser 8-kanałowy)



Argumenty:

in – argument cyfrowy

n – numer wyjścia na które ma być przełączone wejście *in* (wartości z zakresu 0 – 7 format: *Decimal* lub *Hex*)

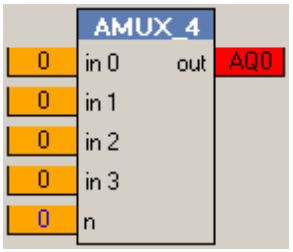
Zwracane wartości:

out 0-7 – na wyjściu *out**n*** jest obecna wartość z wejścia *in*, na pozostałych 0

Opis:

Realizuje funkcję 8-kanałowego demultipleksera cyfrowego. Wyjście *out**n*** = *in* gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*, na pozostałych wyjściach jest stan 0. Jeżeli *n* jest spoza zakresu to wszystkie wyjścia przyjmują stan 0.

7.7. Analog Multiplexer 4Ch (Multiplekser analogowy 4-kanałowy)



Argumenty:

in 0-3 – argumenty analogowe

n – numer wejścia które ma być przełączone na wyjście *out* (wartości z zakresu 0 – 3 format: *Decimal* lub *Hex*)

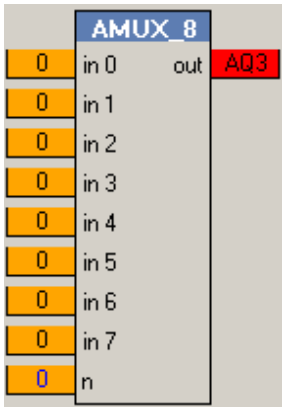
Zwracane wartości:

out – wartość z wejścia *in**n***.

Opis:

Realizuje funkcję 4-kanałowego multipleksera analogowego. Wyjście *out* = *in**n*** gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*. Jeżeli *n* jest spoza zakresu to *out* = 0.

7.8. Analog Multiplexer 8Ch (Multiplekser analogowy 8-kanałowy)



Argumenty:

in 0-7 – argumenty analogowe

n – numer wejścia które ma być przełączone na wyjście *out* (wartości z zakresu 0 – 7 format: *Decimal* lub *Hex*)

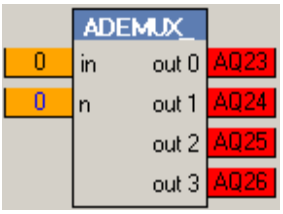
Zwracane wartości:

out – wartość z wejścia *in**n***

Opis:

Realizuje funkcję 8-kanałowego multipleksera analogowego. Wyjście *out* = *in**n*** gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*. Jeżeli *n* jest spoza zakresu to *out* = 0.

7.9. Analog Demultiplexer 4Ch (Demultiplekser analogowy 4-kanałowy)



Argumenty:

in – argument analogowy

n - numer wyjścia na które ma być przełączone wejście *in* (wartości z zakresu 0 – 3 format: *Decimal* lub *Hex*)

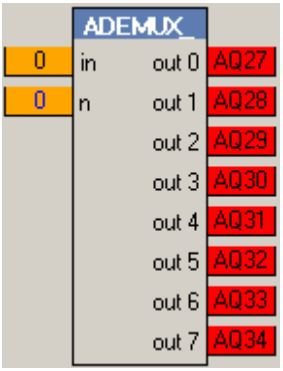
Zwracane wartości:

out 0-3 – na wyjściu *out**n*** jest obecna wartość z wejścia *in*, na pozostałych 0

Opis:

Realizuje funkcję 4-kanałowego demultipleksera analogowego. Wyjście *out**n*** = *in* gdzie *n* jest argumentem w formacie *Decimal* lub *Hex*, na pozostałych wyjściach jest stan 0. Jeżeli *n* jest spoza zakresu to wszystkie wyjścia przyjmują stan 0.

7.10. Analog Demultiplexer 8Ch (Demultiplekser analogowy 8-kanałowy)



Argumenty:

in – argument analogowy

n – numer wyjścia na które ma być przełączone wejście *in* (wartości z zakresu 0 – 7 format: *Decimal* lub *Hex*)

Zwracane wartości:

out 0-7 – na wyjściu *out**n*** jest obecna wartość z wejścia *in*, na pozostałych 0.

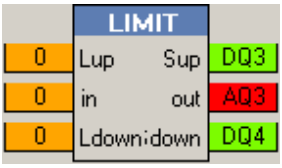
Opis:

Realizuje funkcję 8-kanałowego demultipleksera analogowego. Wyjście *out**n*** = *in* gdzie *n* jest

argumentem w formacie *Decimal* lub *Hex* na pozostałych wyjściach jest stan 0. Jeżeli *n* jest spoza zakresu to wszystkie wyjścia przyjmują stan 0.

8. Compare

8.1. Limitation



Argumenty:

Lup – limit górny

in – argument analogowy

Ldown – limit dolny

Zwracane wartości:

Sup – sygnalizacja przekroczenia górnego limitu

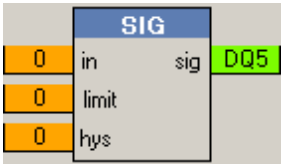
out – ograniczony sygnał z wejścia *in* zawierający się w zakresie *Lup* >= *out* >= *Ldown*

Sdown – sygnalizacja przekroczenia dolnego limitu

Opis:

Funkcja limituje sygnał z wejścia *in*. Po przekroczeniu wartości granicznych *Lup* lub *Ldown* sygnalizuje to odpowiednio na wyjściu *Sup* lub *Sdown*.

8.2. Signaling (Sygnalizacja z histerezą)



Argumenty:

in – wejście analogowe

limit – górna wartość graniczna

hys – dolna histereza sygnalizacji

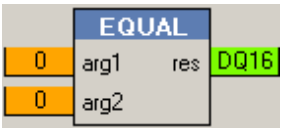
Zwracane wartości:

sig – sygnalizacja przekroczenia wartości granicznej *limit* z dolną histerezą

Opis:

Funkcja sygnalizująca przekroczenie na wejściu *in* wartości granicznej ustalonej przez wejście *limit*. Jeżeli *in* > *limit* to *sig* = 1. Aby wyście *sig* zmieniło stan spowrotem na 0 wartość na wejściu *in* musi spaść poniżej *limit* – *his*.

8.3. Equal (Równy)



Argumenty:

arg1 – argument analogowy 1

arg2 – argument analogowy 2

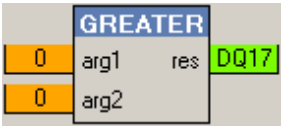
Zwracane wartości:

res – wynik porównania argumentów

Opis:

Funkcja porównuje argumenty *arg1* i *arg2*. Jeżeli są równe to *res* = 1, w przeciwnym wypadku *res* = 0.

8.4. Greater (Większy)



Argumenty:

arg1 – argument analogowy 1

arg2 – argument analogowy 2

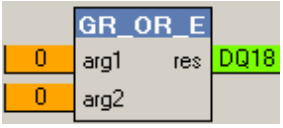
Zwracane wartości:

res – wynik porównania argumentów

Opis:

Funkcja porównuje argumenty *arg1* i *arg2*. Jeżeli *arg1* > *arg2* to *res* = 1, w przeciwnym wypadku *res* = 0.

8.5. Greater or equal (Większy lub równy)



Argumenty:

arg1 – argument analogowy 1

arg2 – argument analogowy 2

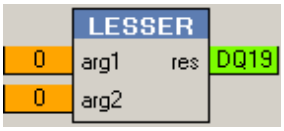
Zwracane wartości:

res – wynik porównania argumentów

Opis:

Funkcja porównuje argumenty *arg1* i *arg2*. Jeżeli *arg1* >= *arg2* to *res* = 1, w przeciwnym wypadku *res* = 0.

8.6. Lesser (Mniejszy)



Argumenty:

arg1 – argument analogowy 1

arg2 – argument analogowy 2

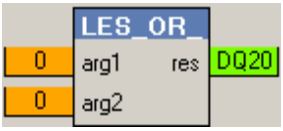
Zwracane wartości:

res – wynik porównania argumentów

Opis:

Funkcja porównuje argumenty *arg1* i *arg2*. Jeżeli *arg1* < *arg2* to *res* = 1, w przeciwnym wypadku *res* = 0.

8.7. Lesser or equal (Mniejszy lub równy)



Argumenty:

arg1 – argument analogowy 1

arg2 – argument analogowy 2

Zwracane wartości:

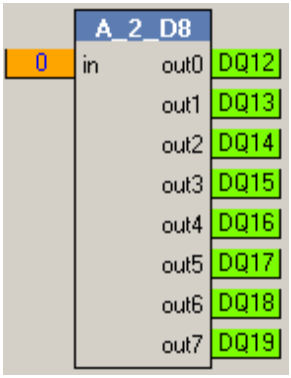
res – wynik porównania argumentów

Opis:

Funkcja porównuje argumenty *arg1* i *arg2*. Jeżeli *arg1* <= *arg2* to *res* = 1, w przeciwnym wypadku *res* = 0.

9. Conversion

9.1. An to Dig8 (Rozbicie wartości analogowej na 8 bitów)



Argumenty:

in – wejście analogowe (format: *Decimal* lub *Hex*)

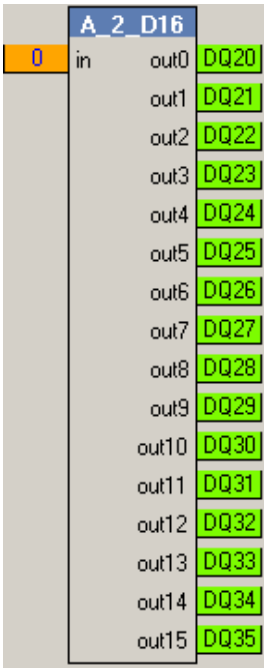
Zwracane wartości:

out0-7 – wyjścia cyfrowe przedstawiające 8 najmłodszych bitów analogowej wartości wejściowej

Opis:

Konwersja wartości analogowej na 8 bitów. Wyjścia zawierają 8 najmłodszych bitów wartości *in*.

9.2. An to Dig16 (Rozbicie wartości analogowej na 16 bitów)



Argumenty:

in – wejście analogowe (format: *Decimal* lub *Hex*)

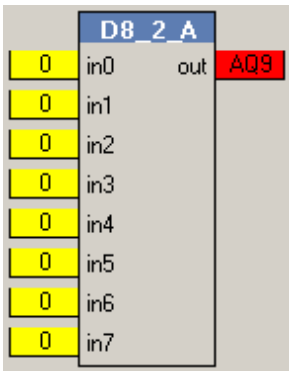
Zwracane wartości:

out0-15 – wyjścia cyfrowe przedstawiające 16 najmłodszych bitów analogowej wartości wejściowej

Opis:

Konwersja wartości analogowej na 16 bitów. Wyjścia zawierają 16 najmłodszych bitów wartości *in*.

9.3. Dig8 to An (Scalenie 8 bitów w sygnał analogowy)



Argumenty:

in0-7 – wejścia cyfrowe

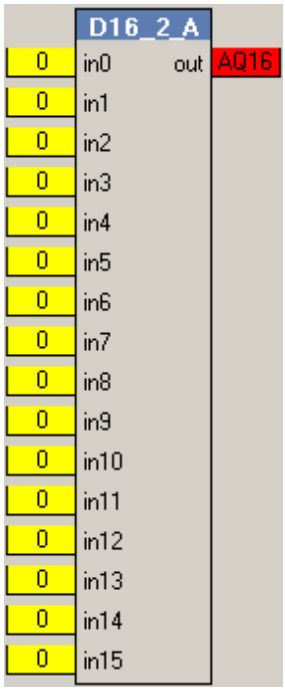
Zwracane wartości:

out – wyjście analogowe którego wartość jest liczbą złożoną z 8 bitów wartości wejściowych (format: *Decimal* lub *Hex*)

Opis:

Konwersja 8 wartości cyfrowych na wartość analogową.

9.4. Dig16 to An (Scalenie 16 bitów w sygnał analogowy)



Argumenty:

in0-15 – wejścia cyfrowe

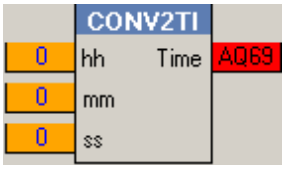
Zwracane wartości:

out – wyjście analogowe którego wartość jest liczbą złożoną z 16 bitów wartości wejściowych (format: *Decimal* lub *Hex*)

Opis:

Konwersja 16 wartości cyfrowych na wartość analogową.

9.5. Conv2Time (Konwersja do czasu)



Argumenty:

hh – godziny w formacie *Decimal* lub *Hex*

mm – minuty w formacie *Decimal* lub *Hex*

ss – sekundy w formacie *Decimal* lub *Hex*

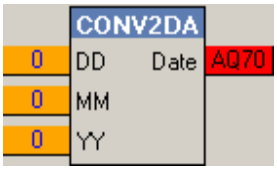
Zwracane wartości:

Time – Czas w formacie *Time (hh:mm.ss)*

Opis:

Konwersja czasu zapisanego oddzielnie na trzech wartościach do formatu *Time (hh:mm.ss)*.

9.6. Conv2Date (Konwersja do daty)



Argumenty:

DD – dni w formacie *Decimal* lub *Hex*

MM – miesiące w formacie *Decimal* lub *Hex*

YY – lata w formacie *Decimal* lub *Hex* (zakres: 0 - 99)

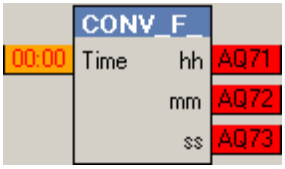
Zwracane wartości:

Date – Data w formacie *Date (DD-MM-YY)*

Opis:

Konwersja daty zapisanej oddzielnie na trzech wartościach do formatu *Date (DD-MM-YY)*.

9.7. ConvFromTime (Konwersja z czasu)



Argumenty:

Time – Czas w formacie *Time (hh:mm.ss)*

Zwracane wartości:

hh – godziny w formacie *Decimal* lub *Hex*

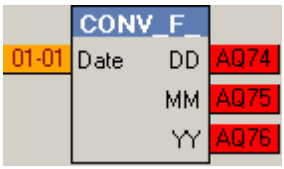
mm – minuty w formacie *Decimal* lub *Hex*

ss – sekundy w formacie *Decimal* lub *Hex*

Opis:

Konwersja czasu z formatu *Time (hh:mm.ss)* do zapisanego oddzielnie na trzech wartościach.

9.8. ConvFromDate (Konwersja z daty)



Argumenty:

Date – Data w formacie *Date (DD-MM-YY)*

Zwracane wartości:

DD – dni w formacie *Decimal* lub *Hex*

MM – miesiące w formacie *Decimal* lub *Hex*

YY – lata w formacie *Decimal* lub *Hex* (zakres: 0 - 99)

Opis:

Konwersja daty z formatu *Date (DD-MM-YY)* do zapisanej oddzielnie na trzech wartościach.

9.9. Sec2Time (Konwersja liczby sekund na czas)

Argumenty:

sec – liczba sekund w formacie *Fixed*

Zwracane wartości:

time – czas w formacie *Time (hh:mm.ss)*

Opis:

Konwersja liczby sekund jakie upłynęły od czasu 00:00.00 do formatu *Time (hh:mm.ss)*.

9.10. Time2Sec (Konwersja czasu na liczbę sekund)

Argumenty:

time – czas w formacie *Time (hh:mm.ss)*

Zwracane wartości:

sec – liczba sekund w formacie *Fixed*

Opis:

Konwersja czasu z formatu *Time (hh:mm.ss)* do liczby sekund jakie upłynęły od 00:00.00.

9.11. Date2Days (Konwersja daty na liczbę dni)

Argumenty:

date – data w formacie *Date (DD-MM-YY)*

Zwracane wartości:

days – liczba dni w formacie *Decimal* lub *Hex*

Opis:

Konwersja daty w formacie *Date (DD-MM-YY)* do liczby dni jakie upłynęły od daty 01-01-00.

9.12. Days2Date (Konwersja liczby dni na datę)

Argumenty:

days – liczba dni w formacie *Decimal* lub *Hex*

Zwracane wartości:

date – data w formacie *Date (DD-MM-YY)*

Opis:

Konwersja liczby dni jakie upłynęły od daty 01-01-00 na format *Date (DD-MM-YY)*.

9.13. Float2Fixed (Konwersja liczby z formatu float do fixed)

Argumenty:

float – wartość w standardowym 32-bitowym formacie float

Zwracane wartości:

fixed – wartość skonwertowana na typ stałopozycyjny fixed używany w sterownikach EL-Piast.

Opis:

Konwersja liczby zmiennoprzecinkowej typu float na wartość typu fixed.

9.14. Fixed2Float (Konwersja liczby z formatu fixed do float)

Argumenty:

fixed – stałopozycyjna wartość w formacie fixed używana w sterownikach EL-Piast

Zwracane wartości:

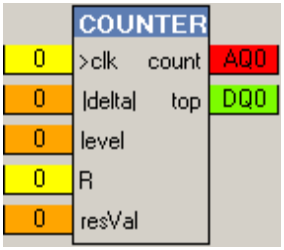
float –wartość skonwertowana na standardowy 32-bitowy typ float

Opis:

Konwersja liczby stałopozycyjnej typu fixed na zmiennopozycyjną typu float.

10. Others

10.1. Counter (Licznik)



Argumenty:

clk – impuls zegarowy dla licznika (wyzwalanie zboczem narastającym)

delta – bezwzględna wartość o jaką zwiększany jest licznik przy każdym impulsie zegarowym

level – maksymalna wartość licznika (szczyt)

R – reset licznika zboczem narastającym

resVal – wartość początkowa licznika

Zwracane wartości:

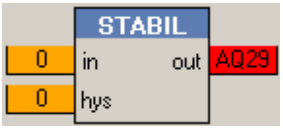
count – aktualny stan licznika

top – ustawiany gdy licznik osiągnął szczyt (wartość *level*)

Opis:

Blok realizuje funkcję licznika. Gdy na *clk* pojawi się zbocze narastające licznik począwszy od *resVal* zwiększa się o wartość *delta* do momentu osiągnięcia wartości maksymalnej *level*. Po osiągnięciu, bądź przekroczeniu wartości maksymalnej wyjście *top* ustawiane jest na stan 1. Po podaniu zbocza narastającego na wejście resetujące *R* licznik powraca do stanu początkowego *resVal*.

10.2. Stabilizer (Stabilizator sygnału)



Argumenty:

in – wartość analogowa

hys – histereza stabilizacji

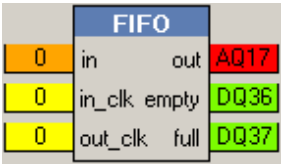
Zwracane wartości:

out – stabilizowany sygnał wejściowy według histerezy *hys*

Opis:

Funkcja realizuje proste stabilizowanie sygnału analogowego w taki sposób że ignoruje drobne zmiany sygnału poniżej +-*hys*. Negatywnym skutkiem działania funkcji jest to że sygnał na wyjściu przestaje być płynny i tworzy schodki z progami o wysokości nie mniejszej niż parametr *hys*.

10.3. FIFO (Kolejka FIFO 8 argumentów)



Argumenty:

in – analogowa wartość wejściowa do kolejki

in_clk – impuls zegarowy powodujący zapisanie wartości z wejścia *in* do kolejki (wywołanie zboczem narastającym). Jeżeli kolejka jest pełna wartość nie zostaje zapisana

out_clk – impuls zegarowy wyzwalający przepisanie kolejnej wartości na wyjście (wyjęcie elementu z kolejki). Jeżeli kolejka jest pusta to na wyjściu *out* ustawiana jest wartość 0

Zwracane wartości:

out – wartość z końca kolejki ostatnio wyjęta w trakcie impulsu *out_clk*. Jeżeli kolejka jest pusta 0

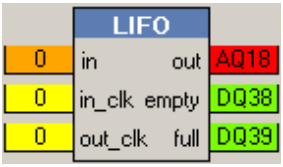
empty – informacja czy kolejka jest pusta

full – informacja czy kolejka jest pełna. Pojemność kolejki to 8 wartości analogowych

Opis:

Kolejka FIFO (First In First Out). Umożliwia kolejkowanie do 8 wartości analogowych.

10.4. LIFO (Kolejka LIFO 8 argumentów)



Argumenty:

in – analogowa wartość wejściowa do kolejki

in_clk – impuls zegarowy powodujący zapisanie wartości z wejścia *in* do kolejki (wywołanie zboczem narastającym). Jeżeli kolejka jest pełna wartość nie zostaje zapisana

out_clk – impuls zegarowy wyzwalający przepisanie kolejnej wartości na wyjście (wyjęcie elementu z kolejki). Jeżeli kolejka jest pusta to na wyjściu *out* ustawiana jest wartość 0

Zwracane wartości:

out – wartość z końca kolejki ostatnio wyjęta w trakcie impulsu *out_clk*

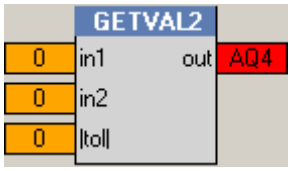
empty – informacja czy kolejka jest pusta

full – informacja czy kolejka jest pełna. Pojemność kolejki to 8 wartości analogowych.

Opis:

Kolejka LIFO (Last In First Out) / stos. Umożliwia kolejkowanie do 8 wartości analogowych.

10.5. Get Value 2 (Ostatnio zmieniona wartość z dwóch)



Argumenty:

in1 – pierwsza wartość

in2 – druga wartość

|*tol*| – minimalna tolerancja ± zmiany dla wartości jaka ma być uwzględniona

Zwracane wartości:

out – wybrana wartość z *in1* lub *in2*

Opis:

Funkcja przy pierwszym wywołaniu obiera na wyjściu *out* stan z wejścia *in1*. Gdy którekolwiek z wejść *in1* lub *in2* zmieni swoją wartość o więcej niż ± |*tol*| wtedy wartość ze zmienionego wejścia zostanie zapamiętana na wyjściu *out*. Taki stan się utrzyma do kolejnej zarejestrowanej zmiany któregokolwiek z wejść *in1* lub *in2*.

11. System

11.1. Temp test (Test pomiaru temperatury)

Argumenty:

in – wartość analogowa pomiaru temperatury (format: *Fixed*).

Zwracane wartości:

test – sygnalizacja poprawnego pomiaru temperatury z *in*

Opis:

Funkcja informuje czy wartość pomiaru temperatury na wejściu *in* jest poprawna. Jeżeli nie to znaczy że na wejściu nie ma elementu pomiarowego (czujnika) lub jest on uszkodzony. W praktyce funkcja sprawdza czy wartość jest różna od wartości nieustalonej 0x80000000.

11.2. Ctrl sensor (Czujnik temperatury z panelu HMI)

Argumenty:

time – czas bezwładności wyjścia pomiarowego temperatury w przypadku braku komunikacji z panelem HMI określany w sekundach (format: *Fixed*).

Zwracane wartości:

sensor – pomiar temperatury odczytywany z panelu HMI podłączonego do dedykowanego portu.

Opis:

Funkcja podaje wartość temperatury z czujnika umieszczonego w zewnętrznym panelu HMI podłączonym do dedykowanego portu panela HMI. Jeżeli komunikacja jest przerwana to po czasie *time* na wyjściu *sensor* pojawi się sygnał niewiarygodnego pomiaru (wartość nieustalona) co oznacza że czujnik jest nieaktywny (badane funkcją *Temp test*).

11.3. Ctrl sensor 2 (Czujnik temperatury z HMI RS-485 Master)

Argumenty:

time – czas bezwładności wyjścia pomiarowego temperatury w przypadku braku komunikacji z panelem HMI określany w sekundach (format: *Fixed*).

Zwracane wartości:

sensor – pomiar temperatury odczytywany z panelu HMI podłączonego do portu RS-485 Master

Opis:

Funkcja podaje wartość temperatury z czujnika umieszczonego w zewnętrznym panelu HMI podłączonym do portu RS-485 Master. Jeżeli komunikacja jest przerwana to po czasie *time* na wyjściu *sensor* pojawi się sygnał niewiarygodnego pomiaru (wartość nieustalona) co oznacza że czujnik jest nieaktywny (badane funkcją *Temp test*).

11.4. Com detection (Wykrywanie komunikacji)

Argumenty:

time – okres braku komunikacji w sekundach po jakim zostanie skasowane wyjście *detect* (format: *Fixed*)

Zwracane wartości:

detect – informacja o obecności komunikacji z urządzeniami nadrzędnymi na porcie RS-485 Master

Opis:

Funkcja informuje czy na porcie komunikacyjnym RS-485 Master jest obecne urządzenie nadrzędne. Jeżeli przez czas *time* na porcie nie ma żadnej aktywności wyjście *detect* ustawia się w stan 0 (kasuje się).

11.5. Calendar (Aktywacja kalendarza)

Argumenty:

act. – aktywacja funkcji kalendarza

Zwracane wartości:

is act. – informacja czy w danym momencie wykonywany jest któryś z programów kalendarza.

Opis:

Funkcja aktywuje działanie kalendarza. Jeżeli kalendarz jest aktywowany wejściem *act.* i w danym momencie, związanym z czasem i datą zegara RTC, jest wykonywany któryś z programów kalendarza to wyjście *is act.* ustawiane jest w stan 1. Aktywacja kalendarza nie oznacza że w danym czasie istnieje program kalendarza który może być wykonywany.

11.6. Alarm state (Stany alarmowe)

Argumenty:

hard – ustawienie twardego alarmu

soft – ustawienie miękkiego alarmu

Zwracane wartości:

hard – informuje czy twardy alarm jest aktywny

soft – informuje czy miękki alarm jest aktywny

Opis:

Funkcja ustawia i odczytuje stany alarmowe. Z reguły stany alarmowe zgłaszane i obsługiwane są blokami typu *Alarm* opisanym w rozdziale 2. . Funkcja *Alarm state* umożliwia zgłoszenie globalnego alarmu, a także odczyt czy w danym momencie został zgłoszony alarm przez blok *Alarm state* lub dowolny blok *Alarm*.

11.7. First loop (Pierwsza pętla programu)

Zwracane wartości:

check – informuje czy właśnie wykonywana jest pierwsza pętla programu po uruchomieniu bądź restarcie sterownika PLC

Opis:

Zwraca informację o tym czy wykonywana jest pierwsza pętla programu po wznowieniu działania sterownika PLC (włączenie do prądu, reset). Przydatne do wywołania wstępnej inicjacji pewnych parametrów algorytmu sterowania.

11.8. Loop count (Licznik pętli)

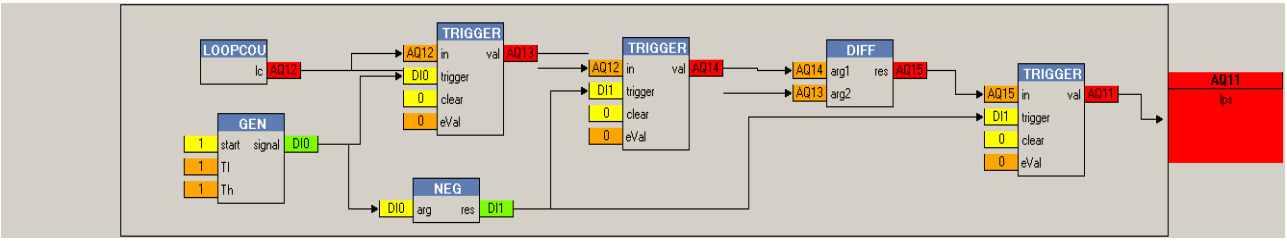
Zwracane wartości:

lc – licznik pętli programu od ostatniego uruchomienia sterownika PLC (format: *Decimal* lub *Hex*)

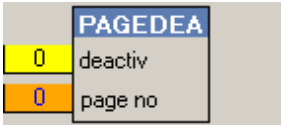
Opis:

Funkcja podaje ilość przebiegu pętli programu liczone od ostatniego uruchomienia sterownika

PLC. Funkcja przydatna np. przy badaniu z jaką szybkością działa algorytm sterowania. Można zbudować makro podające ilość iteracji (pętli) algorytmu sterowania na sekundę:



11.9. Page deactive (Dezaktywacja strony menu)



Argumenty:

deactiv – aktywacja wyłączenia strony nr *page no* menu HMI

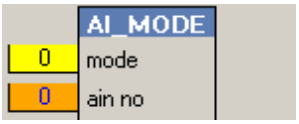
page no – numer strony menu HMI której dotyczy funkcja (format: *Decimal* lub *Hex*)

Opis:

Uwaga: funkcja jest przestarzała. Tą samą akcję można wywołać stosując opcję *Show from Variable* w edytorze HMI.

Funkcja umożliwia wyłączenie / ukrycie odpowiedniej karty z menu panela HMI. Każda strona menu ma swój numer widoczny w edytorze HMI (HMI Editor). Po ustawieniu na wejściu *deactiv* stanu 1 strona o numerze *page no* zostanie ukryta na panelu HMI podłączonym do sterownika PLC.

11.10. An_in mode (Tryb pracy wejścia analogowego)



Argumenty:

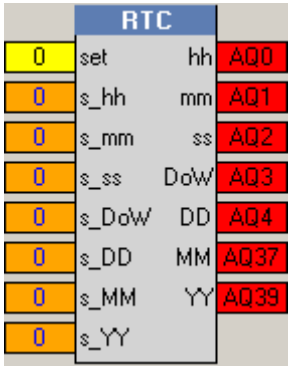
mode – wybór trybu działania wejścia analogowego nr *ain no*

ain no – numer wejścia analogowego (format: *Decimal* lub *Hex*)

Opis:

Funkcja umożliwia przełączenie trybu działania wejścia analogowego dla sterowników posiadających taką funkcję (model ELP10T32-V, ELP11R32-V). Jeżeli *mode=0* to wejście numer *ain no* działa jako wejście napięciowe w zakresie 0-10V. Jeżeli *mode=1* wejście działa jako wejście prądowe w zakresie 0-20mA. Jeżeli blok nie jest wykorzystywany to domyślnie każde wejście analogowe działa jako napięciowe w zakresie 0-10V.

11.11. RTC (Dostęp do zegara RTC)



Argumenty:

set – zapis do RTC wywoływany zboczem narastającym

s_hh – godzina do zapisu w formacie *Decimal*

s_mm – minuta do zapisu w formacie *Decimal*

s_ss – sekunda do zapisu w formacie *Decimal*

s_DoW – dzień tygodnia do zapisu w formacie *Decimal*

s_DD – dzień miesiąca do zapisu w formacie *Decimal*

s_MM – miesiąc do zapisu w formacie *Decimal*

s_YY – rok do zapisu w formacie *Decimal*

Zwracane wartości:

hh – aktualna godzina w formacie *Decimal*

mm – aktualna minuta w formacie *Decimal*

ss – aktualna sekunda w formacie *Decimal*

DoW – aktualny dzień tygodnia (od 1 do 7 gdzie 1 oznacza poniedziałek) formacie *Decimal*

DD – aktualny dzień miesiąca w formacie *Decimal*

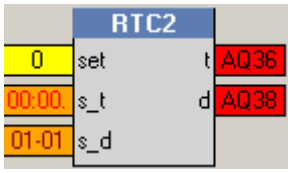
MM – aktualny miesiąc roku w formacie *Decimal*

YY – aktualny rok (w zakresie 0-99) w formacie *Decimal*

Opis:

Funkcja podaje aktualny stan podtrzymywanego baterią zegara czasu rzeczywistego wykorzystywanego głównie do pracy kalendarza i rejestrowania czasu wystąpienia alarmów. Umożliwia również zapis ustalonego na wejściach czasu i daty.

11.12. RTC2 (Dostęp do zegara RTC wersja 2)



Argumenty:

set – zapis do RTC wywoływany zboczem narastającym

s_t – czas do zapisu w formacie *Time (hh:mm.ss)*

s_d –data do zapisu w formacie *Date (DD-MM-YY)*

Zwracane wartości:

t – aktualny czas w formacie *Time (hh:mm.ss)*

d – aktualna data w formacie *Date (DD-MM-YY)*

Opis:

Funkcja podaje aktualny stan podtrzymywanego baterią zegara czasu rzeczywistego wykorzystywanego głównie do pracy kalendarza i rejestrowania czasu wystąpienia alarmów. Umożliwia również zapis ustalonego na wejściach czasu i daty.

11.13. BACnet Error (błąd BACnet)

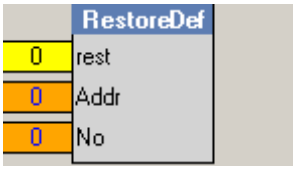
Zwracane wartości:

read_err – informacja czy jest błąd w odczycie obiektów po protokole BACnet

Opis:

Funkcja współpracuje ze sterownikami obsługującymi protokół BACnet. Gdy konfiguracja sterownika (BACnet Integrator) nadpisuje obiekty BACnet z innych urządzeń w sieci i występuje problem z komunikacja to wyjście *read_err* jest ustawione.

11.14. Restore Default (Przywrócenie ustawień domyślnych)



Argumenty:

rest – wywołanie przywracania ustawień domyślnych zboczem narastającym

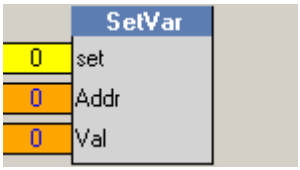
Addr – adres początku zmiennych do przywrócenia

No – ilość kolejnych zmiennych które mają zostać przywrócone do domyślnych stanów

Opis:

Funkcja wywołuje proces przywracania domyślnych nastaw w wybranym zakresie zmiennych aplikacji.

11.15. Set Variable (Ustawienie zmiennej)



Argumenty:

set – wywołanie ustawienia zmiennej zboczem narastającym

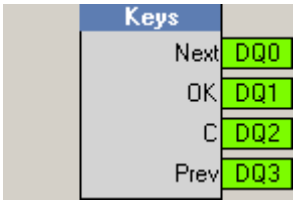
Addr – adres zmiennej do ustawienia

Val – wartość do ustawienia

Opis:

Funkcja ustawia wartość wybranej zmiennej.

11.16. Keys (Klawisze HMI)



Zwracane wartości:

Next – stan wciśnięcia klawisza strzałki *Next*

OK – stan wciśnięcia klawisza *OK*

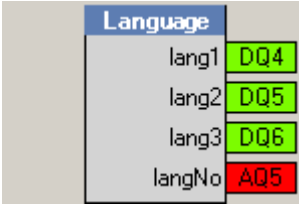
C – stan wciśnięcia klawisza *C*

Prev – stan wciśnięcia klawisza strzałki *Prev*

Opis:

Informacja o stanie klawiszy wbudowanego w sterownik HMI. Dzięki tej funkcji można, poprzez kombinację klawiszy, wywołać dowolną akcję w aplikacji PLC np. załączenie pompy po przytrzymaniu przez 3 sekundy jednocześnie klawiszy *Next* + *Prev*.

11.17. Language (Wersja językowa HMI)



Zwracane wartości:

lang1 – stan 1 oznacza że aktualnie wybrana jest pierwsza wersja językowa

lang2 – stan 1 oznacza że aktualnie wybrana jest druga wersja językowa

lang3 – stan 1 oznacza że aktualnie wybrana jest trzecia wersja językowa

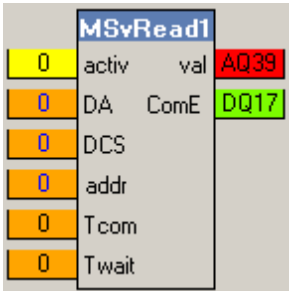
langNo – aktualna wersja językowa w formacie *Decimal* w zakresie (0-2)

Opis:

Informacja o aktualnie wybranej wersji językowej HMI.

12. Communication Basic

12.1. MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres sterownika podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*).

Możliwe nastawy:

0: 2k4 bps

1: 4k8 bps

2: 9k6 bps

3: 14k4 bps

4: 19k2 bps

5: 28k8 bps

6: 38k4 bps

7: 57k6 bps

8: 76k8 bps

9: 115k2 bps

addr – adres zmiennej, która ma być odczytana ze sterownika podrzędnego (format: *Decimal* lub *Hex*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

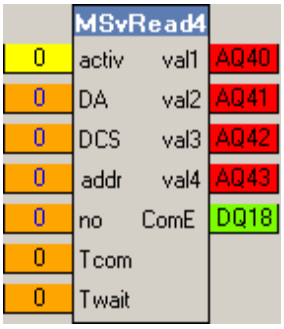
val – wartość zmiennej. Gdy brak komunikacji lub *activ = 0* to wartość jest równa 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji sieciowej między sterownikami firmy EL-Piast zgodny z protokołem Modbus RTU. Blok odczytuje jedną 32-bitową zmienną. Zaleca się stosowanie tej funkcji tylko do współpracy między sterownikami firmy EL-Piast.

12.2. MS VarRead 4 (Odczyt czterech 32-bitowych zmiennych)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres sterownika podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej, która ma być odczytana ze sterownika podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych do odczytu (maksymalnie 4)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

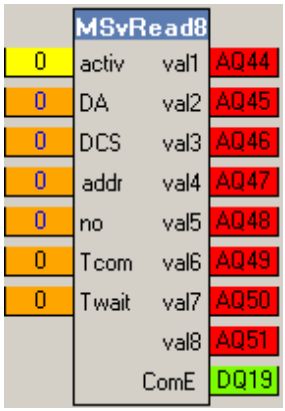
val1-4 – wartości kolejnych odczytanych zmiennych. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji sieciowej między sterownikami firmy EL-Piast zgodny z protokołem Modbus RTU. Blok odczytuje maksymalnie cztery 32-bitowe zmienne. Zaleca się stosowanie tej funkcji tylko do współpracy między sterownikami firmy EL-Piast.

12.3. MS VarRead 8 (Odczyt ośmiu 32-bitowych zmiennych)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres sterownika podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej, która ma być odczytana ze sterownika podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych do odczytu (maksymalnie 8)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

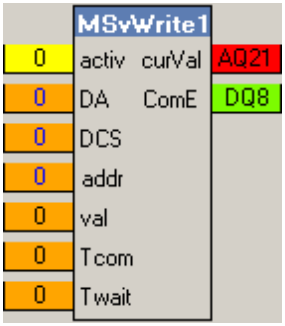
val1-8 – wartości kolejnych odczytanych zmiennych. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji sieciowej między sterownikami firmy EL-Piast zgodny z protokołem Modbus RTU. Blok odczytuje maksymalnie osiem 32-bitowych zmiennych. Zaleca się stosowanie tej funkcji tylko do współpracy między sterownikami firmy EL-Piast.

12.4. MS VarWrite 1 (NE) (Zapis pojedynczej 32-bitowej zmiennej)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres sterownika podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres zmiennej, która ma być zapisana / odczytana ze sterownika podrzędnego (format: *Decimal* lub *Hex*)

val – wartość do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

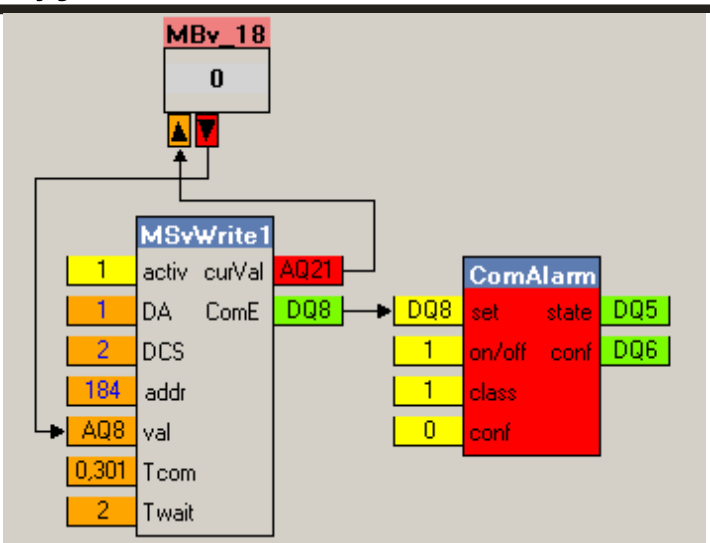
curVal – aktualna wartość zmiennej

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji sieciowej między sterownikami firmy EL-Piast zgodny z protokołem Modbus RTU. Blok zapisuje i odczytuje jedną 32-bitową zmienną. Zapis zmiennej odbywa się tylko gdy wejście *val* zmieni wartość, natomiast odczyt jest ciągły. Zaleca się stosowanie tej funkcji tylko do współpracy między sterownikami firmy EL-Piast. Wartość zmiennej zapisywana jest do nieulotnej pamięci EEPROM. Pamięć EEPROM ma ograniczoną ilość zapisów (około 100tyś.) dlatego należy ograniczyć liczbę zapisów do minimum. W przypadku często zmieniających się wartości, zamiast zapisywać wartość ze sterownika X na sterownik Y należy na sterowniku Y odczytywać wartość ze sterownika X, wtedy pamięć EEPROM nie jest w użyciu.

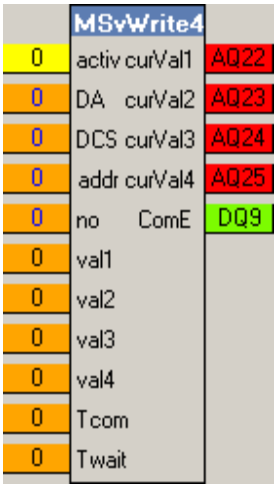
Przykład zastosowania bloku zapisu zmiennej:



Zmienna *MBv_18* przechowuje wartość zdalnej zmiennej o adresie 184 ze sterownika o adresie 1. Ręczna zmiana wartość zmiennej *MBv_18* wywoła procedurę zapisu nowej wartości do zdalnego sterownika. Należy tak ustalić priorytety połączeń aby zapis na wejście *val* był przed odczytem z wyjścia *curVal*. W razie problemów z komunikacją zostanie zgłoszony alarm *ComAlarm*.

Wersja bloku z rozszerzeniem NE oznacza że wartość nie będzie zapisywana do pamięci EEPROM co uchroni sterownik przed uszkodzeniem związanym z częstymi zapisami do nieulotnej pamięci.

12.5. MS VarWrite 4 (NE) (Zapis czterech 32-bitowych zmiennych)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres sterownika podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej, która ma być zapisana / odczytana ze sterownika podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych do zapisu (maksymalnie 4)

val1-4 – wartości kolejnych zmiennych do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

curVal1-4 – aktualne wartości kolejnych zmiennych

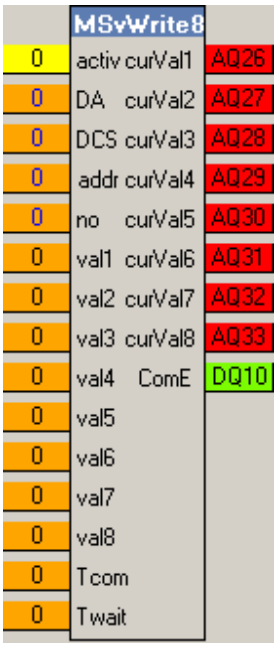
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji sieciowej między sterownikami firmy EL-Piast zgodny z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie cztery 32-bitowe zmienne. Zaleca się stosowanie tej funkcji tylko do współpracy między sterownikami firmy EL-Piast. Zapis wszystkich zmiennych odbywa się tylko gdy którekolwiek z wejść *val1-4* zmieni swoją wartość, natomiast odczyt jest ciągły. Wartości dla zmiennych zapisywane są do nieulotnej pamięci EEPROM. Pamięć EEPROM ma ograniczoną ilość zapisów (około 100tyś.) dlatego należy ograniczyć liczbę zapisów do minimum. W przypadku często zmieniających się wartości, zamiast zapisywać wartości ze sterownika X na sterownik Y należy na sterowniku Y odczytywać wartości ze sterownika X, wtedy pamięć EEPROM nie jest w użyciu.

Wersja bloku z rozszerzeniem NE oznacza że wartość nie będzie zapisywana do pamięci EEPROM co uchroni sterownik przed uszkodzeniem związanym z częstymi zapisami do nieulotnej pamięci.

12.6. MS VarWrite 8 (NE) (Zapis ośmiu 32-bitowych zmiennych)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres sterownika podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej, która ma być zapisana / odczytana ze sterownika podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych do zapisu (maksymalnie 8)

val1-8 – wartości kolejnych zmiennych do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

curVal1-8 – aktualne wartości kolejnych zmiennych

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to *ComE* = 1 (błąd komunikacji)

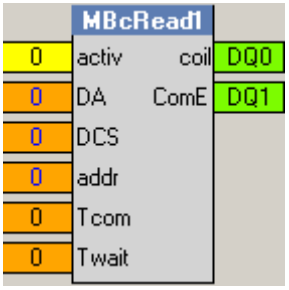
Opis:

Blok realizujący funkcję komunikacji sieciowej między sterownikami firmy EL-Piast zgodny z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie osiem 32-bitowych zmiennych. Zaleca się stosowanie tej funkcji tylko do współpracy między sterownikami firmy EL-Piast. Zapis wszystkich zmiennych odbywa się tylko gdy którekolwiek z wejść *val1-8* zmieni swoją wartość, natomiast odczyt jest ciągły. Wartości dla zmiennych zapisywane są do

nieulotnej pamięci EEPROM. Pamięć EEPROM ma ograniczoną ilość zapisów (około 100tyś.) dlatego należy ograniczyć liczbę zapisów do minimum. W przypadku często zmieniających się wartości, zamiast zapisywać wartości ze sterownika X na sterownik Y należy na sterowniku Y odczytywać wartości ze sterownika X, wtedy pamięć EEPROM nie jest w użyciu.

Wersja bloku z rozszerzeniem NE oznacza że wartość nie będzie zapisywana do pamięci EEPROM co uchroni sterownik przed uszkodzeniem związanym z częstymi zapisami do nieulotnej pamięci.

12.7. MB CoilRead 1 (Odczyt pojedynczej zmiennej binarnej Modbus RTU, komenda: 0x01)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres coil, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

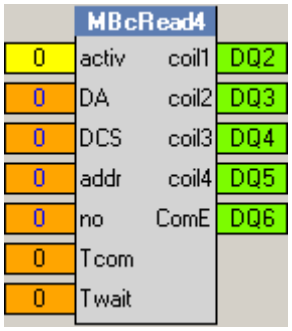
coil – wartość coil. Gdy brak komunikacji lub *activ = 0* to wartość jest równa 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje jedną zmienną typu *Coil*.

12.8. MB CoilRead 4 (Odczyt czterech zmiennych binarnych Modbus RTU, komenda: 0x01)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej typu *Coil*, która ma być odczytana z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych typu *Coil* do odczytu (maksymalnie 4)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

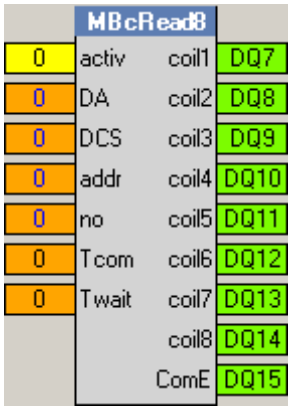
coil1-4 – wartości kolejnych odczytanych zmiennych binarnych typu *Coil*. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie cztery binarne zmienne typu *Coil*.

12.9. MB CoilRead 8 (Odczyt ośmiu zmiennych binarnych Modbus RTU, komenda: 0x01)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej typu *Coil*, która ma być odczytana z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych typu *Coil* do odczytu (maksymalnie 8)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

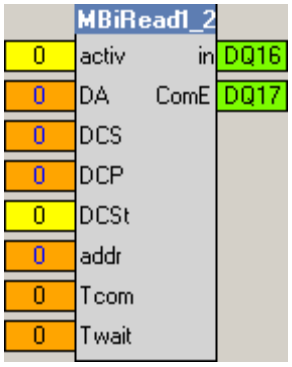
coil1-8 – wartości kolejnych odczytanych zmiennych binarnych typu *Coil*. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie osiem binarnych zmiennych typu *Coil*.

12.10. MB InputRead 1_2 (Odczyt jednego binarnego wejścia Modbus RTU, komenda: 0x02)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu***Błąd! Nie można dnaleźć źródła odwołania.***

addr – adres wejścia, które ma być odczytane z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

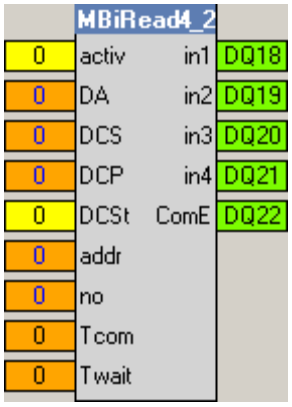
in – wartość wejścia. Gdy brak komunikacji lub *activ = 0* to wartość jest równa 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje jedną binarną zmienną typu *Input*.

12.11. MB InputRead 4_2 (Odczyt czterech binarnych wejść Modbus RTU, komenda: 0x02)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszej zmiennej typu *Coil*, która ma być odczytana z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych typu *Input* do odczytu (maksymalnie 4)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

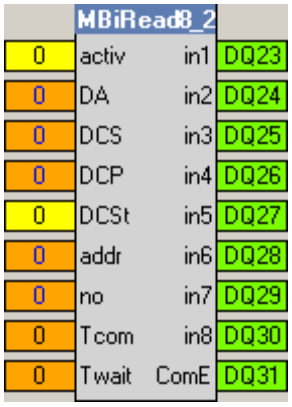
in1-4 – wartości kolejnych odczytanych zmiennych binarnych typu *Input*. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie cztery binarne zmienne typu *Input*.

12.12. MB InputRead 8_2 (Odczyt ośmiu binarnych wejść Modbus RTU, komenda: 0x02)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszej zmiennej typu *Coil*, która ma być odczytana z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych typu *Input* do odczytu (maksymalnie 8)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

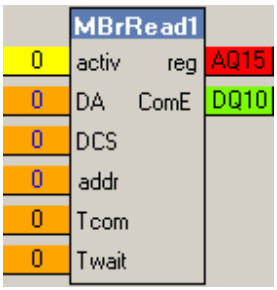
in1-8 – wartości kolejnych odczytanych zmiennych binarnych typu *Input*. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie osiem binarnych zmiennych typu *Input*.

12.13. MB RegRead 1 (Odczyt pojedynczego rejestru Modbus RTU, komenda: 0x03)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

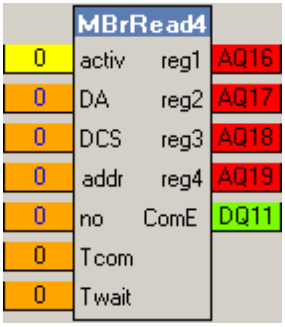
reg – wartość rejestru. Gdy brak komunikacji lub *activ = 0* to wartość jest równa 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje jeden 16-bitowy rejestr.

12.14. MB RegRead 4 (Odczyt czterech rejestrów Modbus RTU, komenda: 0x03)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszego rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do odczytu (maksymalnie 4)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

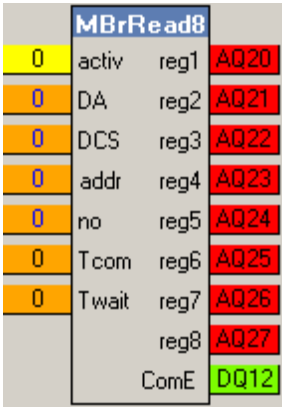
reg1-4 – wartości kolejnych odczytanych rejestrów. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie cztery 16-bitowe rejestry.

12.15. MB RegRead 8 (Odczyt ośmiu rejestrów Modbus RTU, komenda: 0x03)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzednego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszego rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do odczytu (maksymalnie 8)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

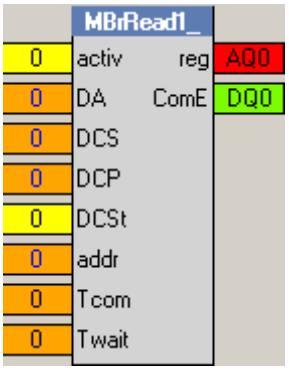
reg1-8 – wartości kolejnych odczytanych rejestrów. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie osiem 16-bitowych rejestrów.

12.16. MB RegRead 1_2 (Odczyt pojedynczego rejestru Modbus RTU, komenda: 0x03)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

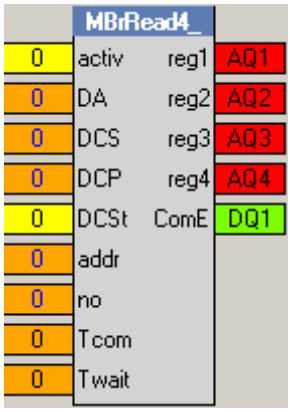
reg – wartość rejestru. Gdy brak komunikacji lub *activ = 0* to wartość jest równa 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje jeden 16-bitowy rejestr.

12.17. MB RegRead 4_2 (Odczyt czterech rejestrów Modbus RTU, komenda: 0x03)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszego rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do odczytu (maksymalnie 4)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

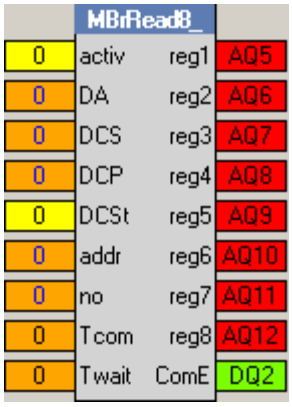
reg1-4 – wartości kolejnych odczytanych rejestrów. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie cztery 16-bitowe rejestry.

12.18. MB RegRead 8_2 (Odczyt ośmiu rejestrów Modbus RTU, komenda: 0x03)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzednego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszego rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do odczytu (maksymalnie 8)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

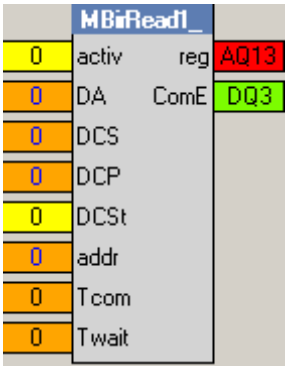
reg1-8 – wartości kolejnych odczytanych rejestrów. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie osiem 16-bitowych rejestrów.

12.19. MB InRegRead 1_2 (Odczyt jednego rej. we. Modbus RTU, komenda: 0x04)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

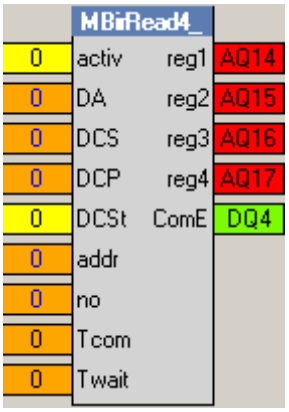
reg – wartość rejestru. Gdy brak komunikacji lub *activ = 0* to wartość jest równa 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do sterownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje jeden 16-bitowy rejestr wejściowy.

12.20. MB InRegRead 4_2 (Odczyt czterech rej. we. Modbus RTU, komenda: 0x04)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszego rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do odczytu (maksymalnie 4)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

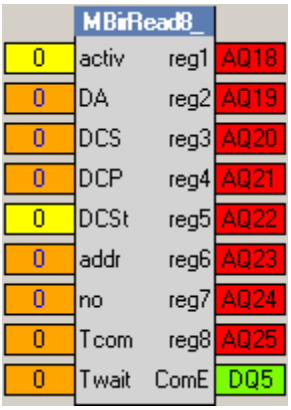
reg1-4 – wartości kolejnych odczytanych rejestrów. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie cztery 16-bitowe rejestry wejściowe.

12.21. MB InRegRead 8_2 (Odczyt ośmiu rej. wej. Modbus RTU, komenda: 0x04)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzednego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszego rejestru, który ma być odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do odczytu (maksymalnie 8)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

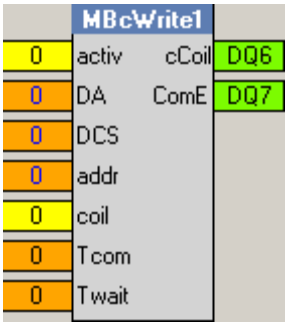
reg1-8 – wartości kolejnych odczytanych rejestrów. Gdy brak komunikacji lub *activ = 0* to wszystkie wartości równe 0

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok odczytuje maksymalnie osiem 16-bitowych rejestrów wejściowych.

12.22. MB CoilWrite 1 (Zapis pojedynczej binarnej zmiennej Modbus RTU, komenda: 0x05)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres zmiennej binarnej, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

coil – wartość do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

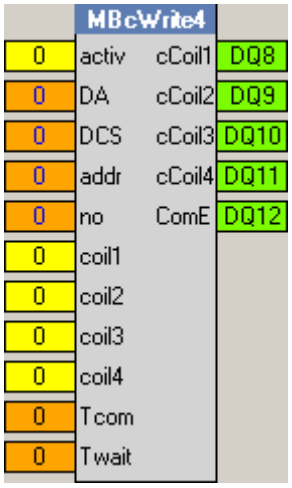
cCoil – aktualna wartość zmiennej binarnej

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje jedną binarną zmienną typu *Coil*. Zapis zmiennej odbywa się tylko gdy wejście *coil* zmieni wartość, natomiast odczyt jest ciągły. Jeżeli wartość zmiennej zapisywana jest w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.23. MB CoilWrite 4 (Zapis czterech binarnych zmiennych Modbus RTU, komenda: 0x0F)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej, która ma być zapisana / odczytana z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych do zapisu (maksymalnie 4)

coil1-4 – wartości kolejnych zmiennych do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

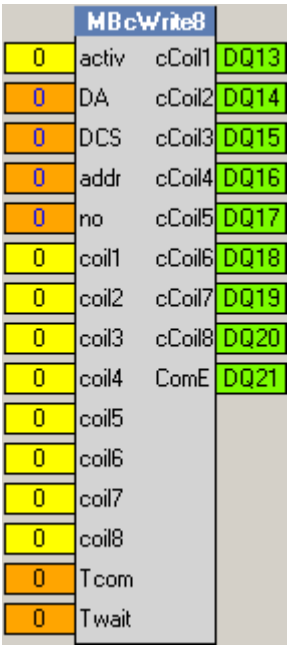
cCoil1-4 – aktualne wartości kolejnych zmiennych binarnych

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie cztery binarne zmienne typu *Coil*. Zapis wszystkich zmiennych odbywa się tylko gdy którekolwiek z wejść *coil1-4* zmieni swoją wartość, natomiast odczyt jest ciągły. Jeżeli wartości zmiennych zapisywane są w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.24. MB CoilWrite 8 (Zapis ośmiu binarnych zmiennych Modbus RTU, komenda: 0x0F)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszej zmiennej, która ma być zapisana / odczytana z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba zmiennych binarnych do zapisu (maksymalnie 8)

coil1-8 – wartości kolejnych zmiennych binarnych do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

cCoil1-8 – aktualne wartości kolejnych zmiennych binarnych

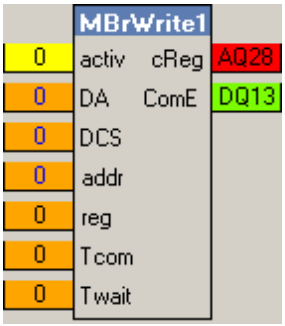
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie osiem zmiennych binarnych typu *Coil*. Zapis wszystkich zmiennych odbywa się tylko gdy którekolwiek z wejść *coil1-8* zmieni swoją wartość, natomiast odczyt jest ciągły. Jeżeli wartości rejestrów zapisywane są w urządzeniu podrzędnym do nieulotnej pamięci EEPROM

należy ograniczyć liczbę zapisów do minimum.

12.25. MB RegWrite 1 (Zapis pojedynczego rejestru Modbus RTU, komenda: 0x06)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres rejestru, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

reg – wartość do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

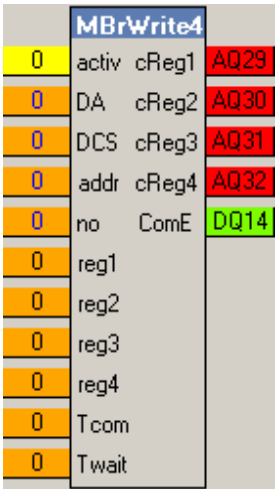
cReg – aktualna wartość rejestru

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje jeden 16-bitowy rejestr. Zapis rejestru odbywa się tylko gdy wejście *reg* zmieni wartość, natomiast odczyt jest ciągły. Jeżeli wartość rejestru zapisywana jest w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.26. MB RegWrite 4 (Zapis czterech rejestrów Modbus RTU, komenda: 0x10)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszego rejestru, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do zapisu (maksymalnie 4)

reg1-4 – wartości kolejnych rejestrów do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

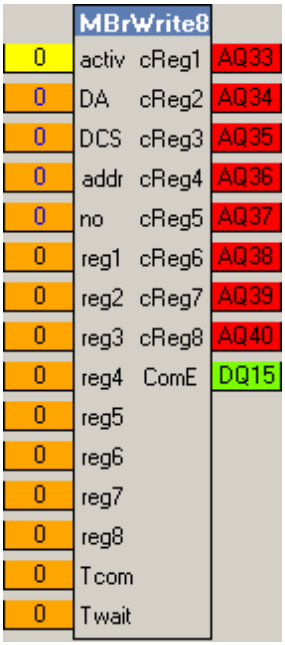
cReg1-4 – aktualne wartości kolejnych rejestrów

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie cztery 16-bitowe rejestry. Zapis wszystkich rejestrów odbywa się tylko gdy którekolwiek z wejść *reg1-4* zmieni swoją wartość, natomiast odczyt jest ciągły. Jeżeli wartości rejestrów zapisywane są w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.27. MB RegWrite 8 (Zapis ośmiu rejestrów Modbus RTU, komenda: 0x10)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

addr – adres pierwszego rejestru, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do zapisu (maksymalnie 8)

reg1-8 – wartości kolejnych rejestrów do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

cReg1-8 – aktualne wartości kolejnych rejestrów

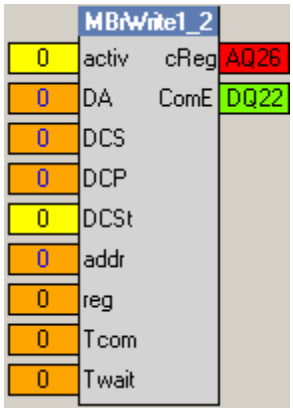
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie osiem 16-bitowych rejestrów. Zapis wszystkich rejestrów odbywa się tylko gdy którekolwiek z wejść *reg1-8* zmieni swoją wartość, natomiast odczyt jest ciągły. Jeżeli wartości rejestrów

zapisywane są w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.28. MB RegWrite 1_2 (Zapis pojedynczego rejestru Modbus RTU, komenda: 0x06)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres rejestru, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

reg – wartość do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

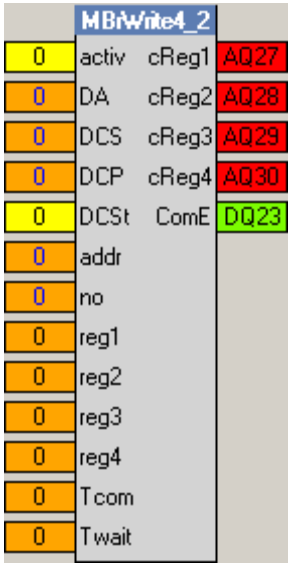
cReg – aktualna wartość rejestru

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to *ComE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje jeden 16-bitowy rejestr. Zapis rejestru odbywa się tylko gdy wejście *reg* zmieni wartość, natomiast odczyt jest ciągły. Jeżeli wartość rejestru zapisywana jest w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.29. MB RegWrite 4_2 (Zapis czterech rejestrów Modbus RTU, komenda: 0x10)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszego rejestru, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do zapisu (maksymalnie 4)

reg1-4 – wartości kolejnych rejestrów do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

cReg1-4 – aktualne wartości kolejnych rejestrów

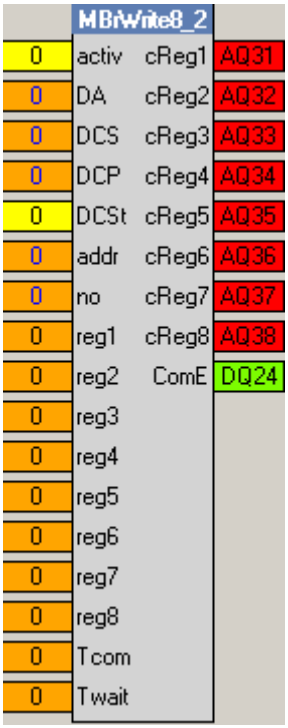
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie cztery 16-

bitowe rejestry. Zapis wszystkich rejestrów odbywa się tylko gdy którekolwiek z wejść *reg1-4* zmieni swoją wartość, natomiast odczyt jest ciągły. Jeżeli wartości rejestrów zapisywane są w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

12.30. MB RegWrite 8_2 (Zapis ośmiu rejestrów Modbus RTU, komenda: 0x10)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

DA – adres Modbus urządzenia podrzędnego (format: *Decimal* lub *Hex*)

DCS – prędkość komunikacji (format: *Decimal* lub *Hex*). Nastawy takie same jak w przypadku 12.1 MS VarRead 1 (Odczyt pojedynczej 32-bitowej wartości).

DCP – parzystość komunikacji. 0 – brak, 1 – parzyste, 2 – nieparzyste

DCSt – ilość bitów stopu komunikacji: 0 – 1 bit stopu, 1 – 2 bity stopu

addr – adres pierwszego rejestru, który ma być zapisany / odczytany z urządzenia podrzędnego (format: *Decimal* lub *Hex*)

no – liczba rejestrów do zapisu (maksymalnie 8)

reg1-8 – wartości kolejnych rejestrów do zapisu

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

cReg1-8 – aktualne wartości kolejnych rejestrów

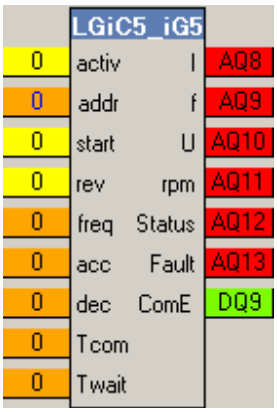
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do urządzenia podrzędnego to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji między sterownikiem PLC, a urządzeniem podrzędnym zgodnym z protokołem Modbus RTU. Blok zapisuje i odczytuje maksymalnie osiem 16-bitowych rejestrów. Zapis wszystkich rejestrów odbywa się tylko gdy którekolwiek z wejść *reg1-8* zmieni swoją wartość, natomiast odczyt jest ciągły. Jeżeli wartości rejestrów zapisywane są w urządzeniu podrzędnym do nieulotnej pamięci EEPROM należy ograniczyć liczbę zapisów do minimum.

13. Communication Devices

13.1. LG iC5/iG5 (Obsługa falownika LG iC5 / iG5)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

start – sygnał startu dla falownika

rev – sygnał do pracy silnika w rewersie

freq – częstotliwość zadana dla silnika z zakresu 0 – 60 Hz (format: *Fixed*)

acc – czas przyspieszania w sekundach od 0 do częstotliwości maksymalnej (format: *Fixed*)

dec – czas hamowania w sekundach od częstotliwości maksymalnej do 0 (format: *Fixed*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od falownika w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu falownika w amperach (format: *Fixed*)

f – aktualna częstotliwość pracy silnika w Hz (format: *Fixed*)

U – aktualne napięcie wyjściowe falownika w V (format: *Fixed*)

rpm – aktualna prędkość obrotowa silnika obr./min (format: *Fixed*)

Status – aktualny stan pracy falownika zapisany bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej statusu na podstawie dokumentacji LG:

bit 0: Stop

bit 1: Forward running

bit 2: Reverse running

bit 3: Fault (Trip)

bit 4: Accelerating

bit 5: Decelerating

bit 6: speed arrival

bit 7: DC Braking

bit 8: Stopping

bit 9: not Used

bit 10: Brake Open

bit 11: Forward run command

bit 12: Reverse run command

bit 13: REM. R/S

bit 14: REM. Freq

Fault – informacja o błędzie falownika. Wartość różna od zera oznacza błąd w działaniu falownika. Informacja o błędzie zapisana jest bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej informacji o błędzie na podstawie dokumentacji LG:

bit 0: OCT

bit 1: OVT

bit 2: EXT-A

bit 3: EST (BX)

bit 4: COL

bit 5: GFT (Ground Fault)

bit 6: OHT (Inverter overheat)

bit 7: ETH (Motor overheat)

bit 8: OLT (Overload trip)

bit 9: HW-Diag

bit 10: EXT-B

bit 11: EEP (Parameter Write Error)

bit 12: FAN (Lock & Open Error)

bit 13: PO (Phase Open)

bit 14: IOLT

bit 15: LVT

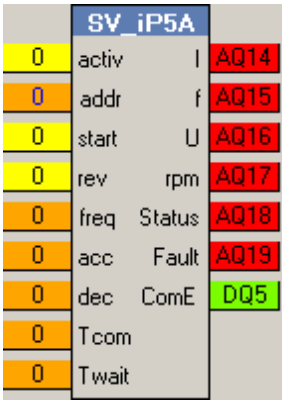
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z falownikiem LG serii iC5 lub iG5 z domyślną prędkością komunikacji 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji

producenta.

13.2. SV iP5A (Obsługa falownika LG SV iP5A)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

start – sygnał startu dla falownika

rev – sygnał do pracy silnika w rewersie

freq – częstotliwość zadana dla silnika z zakresu 0 – 60 Hz (format: *Fixed*)

acc – czas przyspieszania w sekundach od 0 do częstotliwości maksymalnej (format: *Fixed*)

dec – czas hamowania w sekundach od częstotliwości maksymalnej do 0 (format: *Fixed*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od falownika w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu falownika w amperach (format: *Fixed*)

f – aktualna częstotliwość pracy silnika w Hz (format: *Fixed*)

U – aktualne napięcie wyjściowe falownika w V (format: *Fixed*)

rpm – aktualna prędkość obrotowa silnika obr./min (format: *Fixed*)

Status – aktualny stan pracy falownika zapisany bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej statusu na podstawie dokumentacji LG:

bit 0: Stop

bit 1: Forward running

bit 2: Reverse running

bit 3: Fault (Trip)

bit 4: Accelerating

bit 5: Decelerating

bit 6: speed arrival

bit 7: DC Braking

bit 8: Stopping

bit 9: not Used

bit 10: Brake Open

bit 11: Forward run command

bit 12: Reverse run command

bit 13: REM. R/S

bit 14: REM. Freq

Fault – informacja o błędzie falownika. Wartość różna od zera oznacza błąd w działaniu falownika. Informacja o błędzie zapisana jest bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej informacji o błędzie na podstawie dokumentacji LG:

bit 0: OCT

bit 1: OVT

bit 2: EXT-A

bit 3: EST (BX)

bit 4: COL

bit 5: GFT (Ground Fault)

bit 6: OHT (Inverter overheat)

bit 7: ETH (Motor overheat)

bit 8: OLT (Overload trip)

bit 9: HW-Diag

bit 10: EXT-B

bit 11: EEP (Parameter Write Error)

bit 12: FAN (Lock & Open Error)

bit 13: PO (Phase Open)

bit 14: IOLT

bit 15: LVT

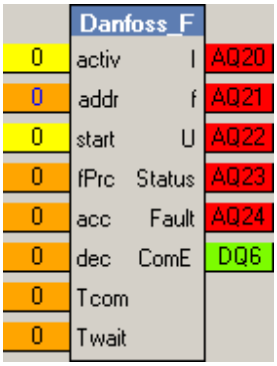
ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z falownikiem LG serii iP5A z domyślną prędkością komunikacji 9600 bps po wewnętrznym protokole LG. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji

producenta.

13.3. Danfoss FC (Obsługa falownika Danfoss)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

start – sygnał startu dla falownika

fPrc – zadany procent częstotliwości dla silnika z zakresu 0 – 100% (format: *Fixed*)

acc – czas przyspieszania w sekundach od 0 do częstotliwości maksymalnej (format: *Fixed*)

dec – czas hamowania w sekundach od częstotliwości maksymalnej do 0 (format: *Fixed*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od falownika w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu falownika w amperach (format: *Fixed*)

f – aktualna częstotliwość pracy silnika w Hz (format: *Fixed*)

U – aktualne napięcie wyjściowe falownika w V (format: *Fixed*)

Status – aktualny stan pracy falownika zapisany bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej statusu na podstawie dokumentacji Danfoss:

bit 0: Control not ready / Control ready

bit 1: Unit not ready / Unit ready

bit 2: Coasting / Enable

bit 3: No error / Error, trip

bit 4: No error / Error (no trip)

bit 5: Reserved

bit 6: Not trip locked / Trip locked

bit 7: No warning / Warning

bit 8: Speed != reference / Speed = reference

bit 9: Local operation / Bus control

bit 10: Out of frequency limit / Frequency limit OK

bit 11: Not running / Running

bit 12: No resistor brake fault / Resistor brake fault

bit 13: Voltage OK / Voltage exceeded

bit 14: Torque OK / Torque exceeded

bit 15: No thermal warning / Thermal warning

Fault – informacja o błędzie falownika. Wartość różna od zera oznacza błąd w działaniu falownika. Informacja o błędzie zapisana jest bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej informacji o błędzie na podstawie dokumentacji Danfoss:

bit 0: Brake check

bit 1: Pwr.card temp

bit 2: Earth Fault

bit 3: -

bit 4: Ctrl.word TO

bit 5: Over Current

bit 6: -

bit 7: Motor th over

bit 8: Motor ETR over

bit 9: Inverter overload

bit 10: DC under volt

bit 11: DC over volt

bit 12: Short Circuit

bit 13: -

bit 14: Mains ph. loss

bit 15: "AMT Not OK"

bit 16: Live zero error

bit 17: Internal fault

bit 18: -

bit 19: U phase loss

bit 20: V phase loss

bit 21: W phase loss

bit 22: -

bit 23: Control Voltage Fault

bit 24: -

bit 25: -

bit 26: Brake resistor shortcircuit

bit 27: Brake IGBT shortcircuit

bit 28: -

bit 29: Drive initialised

bit 30: -

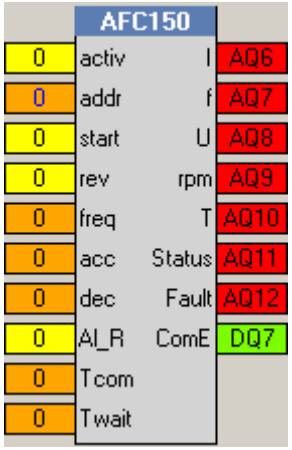
bit 31: Mech. brake low

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z falownikiem Danfoss (wszystkie modele) z domyślną prędkością komunikacji 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.4. AFC150 (Obsługa falownika TWERD)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

start – sygnał startu dla falownika

rev – sygnał do pracy silnika w rewersie

freq – częstotliwość zadana dla silnika z zakresu 0 – 100 Hz (format: *Fixed*)

acc – czas przyspieszania w sekundach od 0 do częstotliwości maksymalnej (format: *Fixed*)

dec – czas hamowania w sekundach od częstotliwości maksymalnej do 0 (format: *Fixed*)

Al_R – reset alarmów

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od falownika w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu falownika w amperach (format: *Fixed*)

f – aktualna częstotliwość pracy silnika w Hz (format: *Fixed*)

U – aktualne napięcie wyjściowe falownika w V (format: *Fixed*)

rpm – aktualna prędkość silnika

T – aktualna temperatura radiatora falownika w °C (format: *Fixed*)

Status – aktualny stan pracy falownika zapisany bitowo (format: *Decimal* lub *Hex*)

Opis zmiennej statusu na podstawie dokumentacji TWERD:

bit 0: Start info

bit 1: One of HMI is active

bit 2: The device is blocked

bit 3: Ready to restart

bit 4,5,6: Not used

bit 7: CRC error in EEPROM

bit 8,9,10,11,12,13 - Not used

bit 14: Engine work direction (0 - right, 1 - left)

bit 15: 1 - id gear (parameter 1.10)

Fault – informacja o kodzie błędu falownika. Wartość różna od zera oznacza błąd w działaniu falownika. Format *Decimal*.

Opis wartości zmiennej informacji o błędzie na podstawie dokumentacji TWERD:

1 - Heatsink temperature to high (higher then 85 degree)

3 - DC voltage to high

4 - DC voltage to low

5 - Shortcut on output

6 - Output current to high

7 - Engine thermal overload

13 - Heatsink temperature to low (lower then -10 degree)

21 - External fault

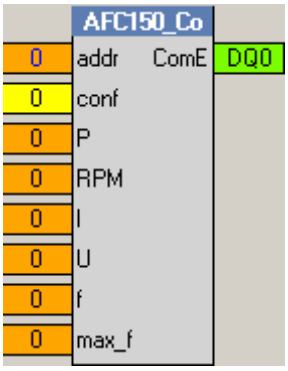
27 - Serial Modbus timeout

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z falownikiem TWERD model AFC150 z domyślną prędkością komunikacji 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.5. AFC150 Config (Konfiguracja falownika TWERD)



Argumenty:

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

conf – zbocze narastające wywołuje zapisanie konfiguracji na falowniku

P – znamionowa moc silnika w W (format: *Fixed*)

RPM – znamionowa prędkość obrotowa silnika (format: *Fixed*)

I – znamionowy prąd silnika (format: *Fixed*)

U – znamionowe napięcie silnika (format: *Fixed*)

f – znamionowa częstotliwość prądu silnika (format: *Fixed*)

max_f – maksymalna dopuszczalna częstotliwość prądu silnika (format: *Fixed*)

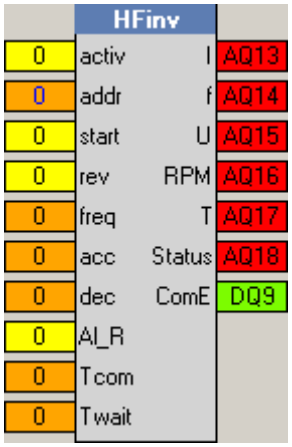
Zwracane wartości:

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji) i proces konfiguracji zostaje przerwany.

Opis:

Blok przeprowadza konfigurację falownika TWERD model AFC150. Konfiguracja odbywa się poprzez komunikację z falownikiem z domyślną prędkością 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.6. HVinverter (Obsługa falownika HFinverter)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

start – sygnał startu dla falownika

rev – sygnał do pracy silnika w rewersie

freq – częstotliwość zadana dla silnika z zakresu 0 – 100 Hz (format: *Fixed*)

acc – czas przyspieszania w sekundach od 0 do częstotliwości maksymalnej (format: *Fixed*)

dec – czas hamowania w sekundach od częstotliwości maksymalnej do 0 (format: *Fixed*)

Al_R – reset alarmów

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od falownika w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu falownika w amperach (format: *Fixed*)

f – aktualna częstotliwość pracy silnika w Hz (format: *Fixed*)

U – aktualne napięcie wyjściowe falownika w V (format: *Fixed*)

rpm – aktualna prędkość silnika

T – aktualna temperatura radiatora falownika w °C (format: *Fixed*)

Status – aktualny stan pracy falownika (format: *Decimal* lub *Hex*)

Opis zmiennej statusu na podstawie dokumentacji HFinverter:

0x00 - idle mode

0x01 - forward work

0x02 - reverse work

0x04 - current overload (OC error)

0x05 - DC voltage overload (OE error)

0x06 - wrong supply voltage parameters (PF1 error)

0x07 - FC overload (OL1 error)

0x08 - voltage to low (LU error)

0x09 - overheating (OH error)

0x0A - engine overload (OL1 error)

0x0B - distortion (ERR error)

0x0C - LL

0x0D - external error ESP

0x0E - ERR1

0x0F - ERR2

0x10 - ERR3

0x11 - ERR4

0x12 - OC1

0x13 - PFO

0x14 - analog signal error (Aerr)

0x15 - EP3

0x16 - load to low

0x17 - PP

0x18 - pressure out of limit (nP)

0x19 - wrong PID parameters (ERR5)

0x1A - wrong user (UERO)

0x1B - macro conflict at switching (UER2)

0x1C - ground error (GF)

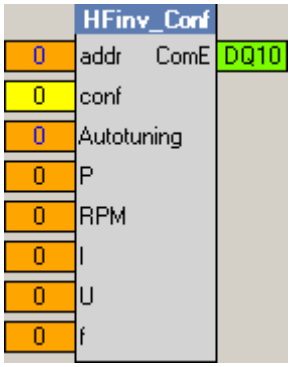
0x1D - PG card error (PG)

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z falownikiem HFinverter E-1000 / E-2000 Micro Drive z domyślną prędkością komunikacji 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.7. HFinverter Config (Konfiguracja falownika HFinverter)



Argumenty:

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

conf – zbocze narastające wywołuje zapisanie konfiguracji na falowniku

Autotuning – 0 – brak, 1 – dynamiczny, 2 - statyczny

P – znamionowa moc silnika w W (format: *Fixed*)

RPM – znamionowa prędkość obrotowa silnika (format: *Fixed*)

I – znamionowy prąd silnika (format: *Fixed*)

U – znamionowe napięcie silnika (format: *Fixed*)

f – znamionowa częstotliwość prądu silnika (format: *Fixed*)

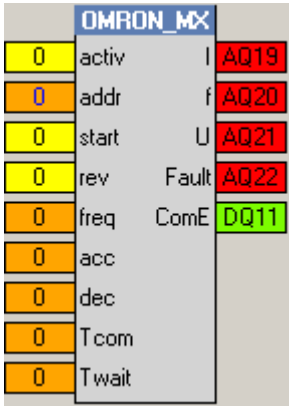
Zwracane wartości:

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji) i proces konfiguracji zostaje przerwany.

Opis:

Blok przeprowadza konfigurację falownika HFinverter E-1000 / E-2000 Micro Drive. Konfiguracja odbywa się poprzez komunikację z falownikiem z domyślną prędkością 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.8. Omron MX2 (Obsługa falownika Omron MX2)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus falownika (format: *Decimal* lub *Hex*)

start – sygnał startu dla falownika

rev – sygnał do pracy silnika w rewersie

freq – częstotliwość zadana dla silnika (format: *Fixed*)

acc – czas przyspieszania w sekundach od 0 do częstotliwości maksymalnej (format: *Fixed*)

dec – czas hamowania w sekundach od częstotliwości maksymalnej do 0 (format: *Fixed*)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od falownika w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu falownika w amperach (format: *Fixed*)

f – aktualna częstotliwość pracy silnika w Hz (format: *Fixed*)

U – aktualne napięcie wyjściowe falownika w V (format: *Fixed*)

Fault – informacja o kodzie błędu falownika. Wartość różna od zera oznacza błąd w działaniu falownika. Format *Decimal*.

Opis wartości zmiennej informacji o błędzie na podstawie dokumentacji Omron:

0: No trip

1: Over-current event while at constant speed

2: Over-current event during deceleration

3: Over-current event during acceleration

4: Over-current event during other conditions

5: Overload protection

6: Braking resistor overload protection

7: Overvoltage protection

8: EEPROM error

9: Undervoltage protection

10: Current detection error

11: CPU error

12: External trip

13: USP error

14: Ground-fault protection

15: Input overvoltage protection

21: Inverter thermal trip

22: CPU error

25: Main circuit error

30: Driver error

35: Thermistor error

36: Braking error

37: Safe Stop

38: Low-speed overload protection

40: Operator connection

41: Modbus communication error

43: Easy sequence error (invalid instruction)

44: Easy sequence error (invalid nesting count)

45: Easy sequence execution error 1

50-59: Easy sequence user trip 0 to 9

60-69: Option error 0 to 9

80: Encoder disconnection

81: Excessive speed

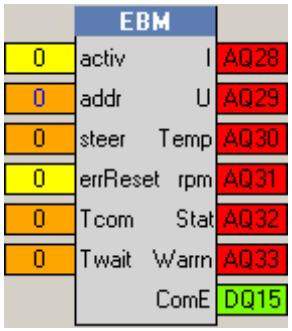
83: Position control range trip

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z falownikiem Omron MX2 z domyślną prędkością komunikacji 9600 bps po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.9. EBMPAPST (Obsługa wentylatora EBMPAPST)



Argumenty:

activ – aktywacja działania bloku. Powoduje wysyłanie komunikatów poprzez port RS-485 Slave

addr – adres Modbus wentylatora (format: *Decimal* lub *Hex*)

steer – sygnał sterujący dla wentylatora w zakresie 0-100% (format: *Fixed*)

errReset – reset błędów

Tcom – czas zwłoki w oczekiwaniu na odpowiedź Modbus od wentylatora w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

I – aktualny pomiar prądu wentylatora w amperach (format: *Fixed*)

U – aktualne napięcie wyjściowe wentylatora w V (format: *Fixed*)

Temp – aktualna temperatura modułu wentylatora (format: *Fixed*)

rpm – aktualna prędkość silnika wentylatora (format: *Fixed*)

Stat – aktualny stan pracy wentylatora (format: *Decimal* lub *Hex*)

Opis zmiennej statusu na podstawie dokumentacji EBMPAPST:

bit 0: Phase failure (3-phase devices) or mains undervoltage (1-phase devices)

bit 1: - Not used

bit 2: Power mod overheated

bit 3: Communication error between master controller and slave controller

bit 4: Fan bad (general error)

bit 5: Motor overheated

bit 6: Hall sensor error

bit 7: Locked motor

bit 8-11: - Not used

bit 12: DC-link undervoltage

bit 13-15: - Not used

Warrn – błędy wentylatora (format: *Decimal lub Hex*)

Opis zmiennej na podstawie dokumentacji EBMPAPST:

bit 0: Current limitation engaged

bit 1: Line impedance too high (DC-link voltage unstable)

bit 2: Power limitation engaged

bit 3: Output stage temperature high

bit 4: Motor temperature high

bit 5: Electronics interior temperature high

bit 6: DC-link voltage low

bit 7: Break mode: set if exterior drive is applied in opposite direction wih high speed for prolonged period

bit 8: - Not used

bit 9: Actual speed is less than limit speed for runing monitor

bit 10: Cable break at set value analogue input

bit 11: Heating enable

bit 12: DC-link voltage high

bit 13: - Not used

bit 14: Supply voltage high

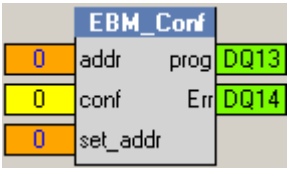
bit 15: Sheding function active

ComE – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji)

Opis:

Blok realizuje funkcję komunikacji z wentylatorem EBMPAPST z domyślnymi nastawami komunikacji 19200E1 po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.10. EBMPAPST Config (Konfiguracja wentylatora EBMPAPST)



Argumenty:

addr – adres Modbus wentylatora (format: *Decimal* lub *Hex*)

conf – zbocze narastające wywołuje zapisanie konfiguracji do wentylatora

set_addr – nowy adres wentylatora (format: *Decimal*)

Zwracane wartości:

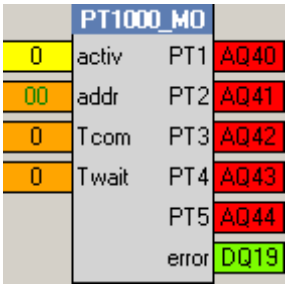
prog – stan 1 oznacza że trwa zapis konfiguracji do wentylatora

Err – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to C*omE* = 1 (błąd komunikacji) i proces konfiguracji zostaje przerwany.

Opis:

Blok dokonuje zmiany adresu Modbus RTU wentylatora EBMPAPST. Konfiguracja odbywa się poprzez komunikację z falownikiem z domyślnymi nastawami 19200E1 po protokole Modbus RTU. W celu zapewnienia poprawnej komunikacji konieczna jest zmiana pewnych nastaw falownika opisanych w dokumentacji producenta.

13.11. PT1000 Module (Obsługa modułu PT1000)



Argumenty:

activ – aktywacja działania bloku komunikacyjnego

addr – adres Modbus modułu PT1000 (format: *Decimal* lub *Hex*). Przesunięcie o 16 (0x10)

Tcom – czas zwłoki w oczekiwaniu na odpowiedź w sekundach (zalecane wartości w zakresie 0,1 – 0,5 sekundy) (format: *Fixed*)

Twait – odstęp czasu w sekundach pomiędzy kolejną serią komunikatów (zalecane wartości w zakresie 1 – 5 sekund) (format: *Fixed*)

Zwracane wartości:

PT1-5 – pomiary temperatur z poszczególnych czujników modułu PT1000

error – jeżeli brak odpowiedzi na trzy kolejne komunikaty wysłane do falownika to *error* = 1 (błąd komunikacji)

Opis:

Blok realizujący funkcję komunikacji z modułem pomiarowym PT1000 firmy EL-Piast. Adresacja Modbus modułu PT1000 zaczyna się od wartości 16 (0x10). Nastawa adresu na zworkach modułu na 0 oznacza faktyczny adres 16 (0x10) który należy wpisać na wejście *addr*.