PikoCNC PLC manual

wersja 1.2

2016.07.28

PLC (*Programmable Logic Controller*)

„Podstawową zasadą pracy sterowników jest praca cykliczna, w której sterownik wykonuje kolejno po sobie pojedyncze rozkazy programu w takiej kolejności, w jakiej są one zapisane w programie. Na początku każdego cyklu program odczytuje "obraz" stanu wejść sterownika i zapisuje ich stany (obraz wejść procesu). Po wykonaniu wszystkich rozkazów i określeniu (wyliczeniu) aktualnego dla danej sytuacji stanu wyjść, sterownik wpisuje stany wyjść do pamięci będącej obrazem wyjść procesu a system operacyjny wysterowuje odpowiednie wyjścia sterujące elementami wykonawczymi. Tak więc wszystkie połączenia sygnałowe spotykają się w układach (modułach) wejściowych sterownika, a program śledzi ich obraz i reaguje zmianą stanów wyjść w zależności od algorytmu.” - Wikipedia

Wstęp

Procesor PLC w kontrolerze umożliwia dowolne łączenie sygnałów, operacje logiczne i czasowe między nimi – czyli o wiele bardziej elastyczne podejście niż „na stałe” ustalone możliwości i przypisane zasoby.

Procesor PLC wykonuje program cyklicznie 1000 razy/sek. Praca każdego cyklu przebiega według następującego algorytmu:

1. Odczyt wejść, stanów kontrolera typu R (read)
2. Wykonanie programu
3. Zapis wyjść oraz stanów kontrolera typu W (write)

PLC rozpoczyna pracę od razu po nawiązaniu połączenia z kontrolerem. Przetwarzaniem programu zajmuje się wyłącznie kontroler i nie podejmie pracy jeśli nie dostanie programu dla PLC lub wystąpił jakiś błąd z nim związany – w takiej sytuacji wystąpi E_STOP z odpowiednim komunikatem.

Rozmiar maksymalny programu PLC zależy od modelu płyty. Dla płyt typu A,B jest to 120 instrukcji, dla płyt typu E 150 instrukcji.

Programowanie

Program jest wykonywany linia po linii, jedna linia tekstu może zawierać tylko jedną instrukcję.

Podstawowym, bardzo często używanym w dalszej części pojęciem jest „bufor linii” (w skrócie BL). Jest to zmienna, która jest swoistym pośrednikiem we wszystkich działaniach. Zmienna jest typu bool czyli może przyjmować dwie wartości „1” lub „0”.

Generalnie instrukcje dzielą się na następujące grupy:

- Zapis bitu z rejestru do BL (tabela 1).
- Zapis BL do bitu w rejestrze (tabela 2).
- Operacje logiczne na BL (tabela 3).
- Instrukcje sterujące przebiegiem programu w zależności od stanu BL (IF, ELSE, ENDIF) (tabela 4).
- Instrukcje dla trybu drabinkowego (tabela 5).

Przykład: przepisanie stanu wejścia numer 0 na wyjście numer 5.

```
<< IN 0       // instrukcja przepisuje stan wejścia numer 0 do BL
>> OUT 5     // instrukcja przepisuje stan BL na wyjście nr 5
```

BL utrzymuje bieżący stan do momentu, aż go nie zmienimy.

Przykład: przepisanie stanu wejścia numer 0 na wyjścia numer 5,6. Na wyjście 7 podawana jest zanegowany stan wejścia 0.

```
<< IN 0       // instrukcja przepisuje stan wejścia numer 0 do BL
>> OUT 5    // przepisanie stanu BL na wyjście nr 5
>> OUT 6    // przepisanie stanu BL na wyjście nr 6
!> OUT 7     // przepisanie zanegowanego stanu BL na wyjście nr 7
```

Tabela 1. Operacje zapisu do bufora linii

Skróty: SRC – adres źródłowy. BL – bufor linii.

Składnia	Opis
<< SRC	Przepisanie SRC do BL
!< SRC	Przepisanie zanegowanej wartości SRC do BL
+< SRC	Wpisanie „1” do BL jeśli wystąpiło narastające zbocze w SRC, inaczej wpisanie „0”. Detekcja zbocza to porównanie bieżącego stanu bitu oraz stanu tego bitu na końcu poprzedniego cyklu.
-< SRC	Wpisanie „1” do BL jeśli wystąpiło opadające zbocze w SRC, inaczej wpisanie „0”
^< SRC	Wpisanie „1” do BL jeśli wystąpiło opadające lub narastające zbocze w SRC, inaczej wpisanie „0”
=< SRC	Wpisanie „1” do BL jeśli nie wystąpiło opadające lub narastające zbocze w SRC, inaczej wpisanie „0”

Tabela 2. Operacje zapisu do rejestrów

Skróty: DST – adres przeznaczenia. BL – bufor linii.

Składnia	Opis
>> DST	Bezpośrednie przepisanie BL do DST
!> DST	Przypisanie zanegowanej wartości BL do DST
S> DST	Wpisanie „1” do DST jeśli BL=1, inaczej DST nie zostanie zmienione.
R> DST	Wpisanie „0” do DST jeśli BL=1, inaczej DST nie zostanie zmienione.
T> DST	Zmiana stanu DST na przeciwny jeśli BL=1, inaczej DST nie zostanie zmieniony.
H> DST	Wpisanie „1” do DST.
L> DST	Wpisanie „0” do DST

Tabela 3. Operacje logiczne na buforze linii

Skróty: SRC – adres źródłowy. BL – bufor linii.

Składnia	Opis
NOT	Negacja BL<= NOT BL.
AND SRC	Wpisanie do BL wyniku operacji BL<=BL AND SRC
AND! SRC	Wpisanie do BL wyniku operacji BL<=BL AND (NOT SRC)
OR SRC	Wpisanie do BL wyniku operacji BL<=BL OR SRC
OR! SRC	Wpisanie do BL wyniku operacji BL<=BL OR (NOT SRC)
XOR SRC	Wpisanie do BL wyniku operacji BL<=BL XOR SRC
XOR! SRC	Wpisanie do BL wyniku operacji BL<=BL XOR (NOT SRC)

Tabela 4. Składnia instrukcji skoków warunkowych

Skróty: BL – bufor linii.

Składnia	Opis
IF_TRUE_BEGIN	Kolejne instrukcje będą wykonywane jeśli BL=1, w innym przypadku nastąpi skok do ELSE_BEGIN lub do ENDIF jeśli brak ELSE_BEGIN. Instrukcje IF..ENDIF mogą być dowolnie zagnieżdżone - czyli jeden warunek IF ENDIF może zawierać w sobie kolejny warunek IF..ENDIF.
IF_FALSE_BEGIN	Kolejne instrukcje będą wykonywane jeśli BL=0, w innym przypadku nastąpi skok do ELSE_BEGIN lub ENDIF jeśli brak ELSE_BEGIN.
ENDIF	Kończy sekwencję IF.. ELSE..
ELSE_BEGIN	Jw.

Tabela 5. Instrukcje dla trybu drabinkowego

Skróty: BL – bufor linii. BLD – indeks głębokości bufora linii

Składnia	Opis
BEG	(BEGIN) Rozpoczęcie trybu drabinkowego BL<=1, BLD <=0
END	Zakończenie gałęzi.
PAR	(PARALLEL) Otwarcie gałęzi równolegle połączonych elementów.
SER	(SERIAL) Otwarcie gałęzi szeregowo połączonych elementów.

Dyrektywy kompilatora

Dyrektywy same w sobie nie są programem natomiast spełniają rolę pomocniczą. Zawsze rozpoczynają się znakiem „#”.

Składnia	Opis
#NAME_I Index = Name	Nadaje nazwę wejściu o podanym indeksie (Index 0-31). Nadawanie nazw ma dwojaką funkcję: • Nazwa jest wyświetlana w oknie kontrolek • Możemy odwoływać się w adresie do nazwy a nie numeru bitu, co jest zdecydowanie lepszym rozwiązaniem. Nadanie nazwy dwukrotnie jest traktowane jako błąd.
#NAME_I_N Index = Name	Jw. z tą różnicą, że wejście ma zanegowany stan aktywny – co widoczne jest także w kontrolkach.
#NAME_O Index = Name	Nadaje nazwę wyjściu o podanym indeksie. (Index 0-31)
#NAME_M Index = Name	Nadaje nazwę bitowi w rejestrze M. (Index 0-31)
#SET_OPTION Index = Value	Nadanie wartości rejestrowi opcji kompilacji o danym indeksie.
#IF_OPTION Index = Value	Sprawdza czy rejestr opcji zawiera daną wartość. Jeśli nie zawiera podanej wartości kolejne instrukcje aż do #END_IF nie będą kompilowane – czyli nie znajdą się w programie wysłanym do kontrolera. Także inne dyrektywy znajdujące się w obrębie #IF.. #END nie będą uwzględniane.
#END_OPTION	Kończy #IF_OPTION
#TXT_MSG Index = Text	Ustawienie tekstu powiadomienia jakie pojawi się na ekranie po aktywowaniu w rejestrze MEMO bitu C_MSG_% - gdzie % to numer indeksu.
#TXT_ESTOP Index = Text	Ustawienie tekstu alarmu jaki pojawi się na ekranie po wystąpieniu E_STOP o numerze równym Index.
#SET_M Index = Value	Nadanie wartości początkową bitu o danym indeksie w rejestrze M. Jest to wartość wyjściowa po uruchomieniu PLC.
#SET_TIMER Index = Time	Ustawienie w sekundach czasu timera o danym indeksie. (Index 0-7), (Time 0.001 – 65 sek)
#MACRO_NAME Index = Macro	Ustawienie makra jakie zostanie wykonane po aktywowaniu w rejestrze MEMO lub IN bitu o nazwie C_MACRO_% - gdzie % to numer indeksu. Indeks może przyjmować wartości w zakresie 0-4. np. Nadajemy nazwę makra wraz z argumentami #MACRO_NAME 0 = G0 X0 Y0 #MACRO_NAME 1 = G1 X100 F1000 Następnie bitowi w rejestrze IN lub MEMO nadajemy nazwę: #NAME_I 8 = C_MACRO_0 #NAME_M 14 = C_MACRO_1

Komentarze

Możemy stosować dwa rodzaje komentarzy dla pojedynczej linii „//”

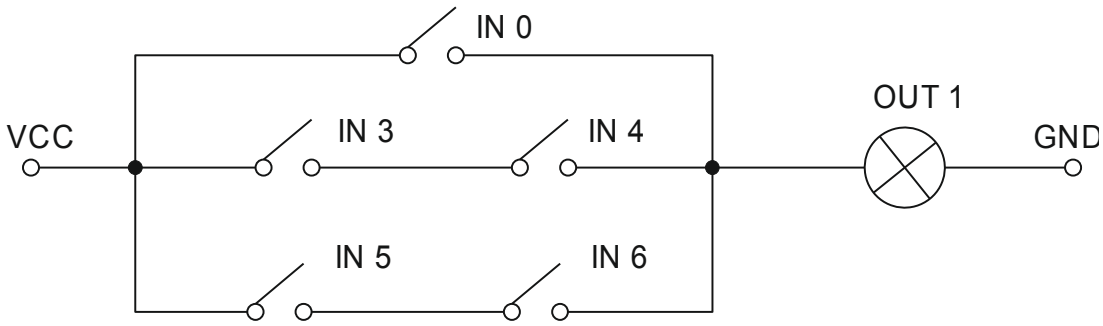
lub dla większej ilości linii nawiasy klamrowe: „{„ oraz „}”. Znaki klamry muszą być jednak jako pierwsze znaki w linii nie licząc spacji i tabulacji. Natomiast komentarz typu „//” może być w dowolnym miejscu linii tekstu.

Tryb drabinkowy

W trybie drabinkowym (tzn. między słowami kluczowymi PAR..END lub SER..END) operacje zapisu do bufora linii funkcjonują nieco inaczej – stan aktywny „1” bitu źródłowego oznacza klucz załączony a „0” wyłączony. Słowo kluczowe „BEG” Ustawia BL na „1” oraz zeruje indeks zagnieżdżenia drabinki. W trybie drabinkowym nie można stosować warunków IF.. ELSE.. ENDIF.

```
BEG
PAR
<< IN 0
    SER
        << IN 3
        << IN 4
    END
    SER
        << IN 5
        << IN 6
    END
END
>> OUT 1
```

Powyższy zapis odpowiada schematowi jak na rysunku.



Rejestry

Skróty: R - read, W - write.

Rodzaj	Typ	Rejestry
Fizyczne wejścia	R	IN
Fizyczne wyjścia	W	OUT
Logiczne wejścia	R	CONTROL1, CONTROL2, 1/2 TIMER
Logiczne wyjścia	W	STATUS, 1/2 TIMER
Pamięć	R/W	MEMO

Jednym z zadań programu w sterowniku PLC jest połączyć te elementy które chcemy wykorzystać np. wejścia fizyczne IN z wejściami logicznymi HOME, LIMIT etc. Kontroler „sam z siebie” nie obsługuje żadnych I/O nawet wyjście „enable” czy „e-stop” musimy oprogramować w PLC.

Rejestry zależnie od rodzaju są typu R lub W. Jednak nie oznacza to że nie można zapisać rejestru typu R lub odczytać rejestru typu W – należy tylko mieć świadomość, że przy rozpoczęciu kolejnej pętli programu rejestry R są inicjowane danymi wartościami np. stanem wejść fizycznych, natomiast rejestry typu W są zerowane. Jedynie rejestr MEMO pamięta swoją zawartość.

Rejestr MEMO

Rejestr ten stanowi podręczną pamięć PLC. Może go zapisywać zarówno program sterujący jak i PLC w kontrolerze. Do bitów rejestru program sterujący może się odwoływać przez ich numer lub nazwę nadaną dyrektywą #NAME_M. Pewne nazwy są zarezerwowane do z góry ustalonych celów:

Komendy zdefiniowane na stałe:

Nazwa	Opis
C_M3	Bit ustawiany gdy w programie wystąpi komenda M3, a gaszony kiedy wystąpi M5
C_M4	Bit ustawiany gdy w programie wystąpi komenda M4, a gaszony kiedy wystąpi M5
C_M7	Bit ustawiany gdy w programie wystąpi komenda M7, a gaszony kiedy wystąpi M9
C_M8	Bit ustawiany gdy w programie wystąpi komenda M8, a gaszony kiedy wystąpi M9
C_M10	Bit ustawiany gdy w programie wystąpi komenda M10, a gaszony kiedy wystąpi M11

Komendy definiowane przez użytkownika:

Nazwa	Opis
C_KEY_%	Gdzie % to liczba w zakresie 0-9. Bit ustawiany kiedy zostanie naciśnięty odpowiedni klawisz z zakresu 0 do 9 na klawiaturze numerycznej. Bit jest gaszony kiedy puścimy. klawisz.

Nazwa	Opis
C_KEY_S%	Bit ustawiany kiedy naciśnięty jest klawisz o kodzie % np. #NAME_M 12 = C_KEY_S72 // klawisz "H" #NAME_M 13 = C_KEY_SS50 // klawisz "2" + SHIFT Aby uzyskać kod klawisza w menu podręcznym okna edytora mamy pozycję „Pobierz do schowka kod klawisza”. Kiedy okienko się otworzy naciskamy dany klawisz, zamykamy okno a następnie przez „CTRL+V” wklejamy kod klawisza.
C_KEY_SS%	Jw. ale razem z klawiszem SHIFT.
C_KEY_SC%	Jw. ale razem z klawiszem CTRL.
C_KEY_SHIFT	Bit ustawiany kiedy zostanie naciśnięty klawisz SHIFT
C_KEY_CTRL	Bit ustawiany kiedy zostanie naciśnięty klawisz CTRL
MSG_TXT_%	Kiedy bit o tej nazwie jest aktywny na ekranie wyświetlany jest komunikat informacyjny o numerze % zdefiniowany dyrektywą #TXT_MSG.
C_MACRO_%	Kiedy bit o tej nazwie zostanie aktywowany wykonane zostanie makro o numerze % zdefiniowane dyrektywą #MACRO_NAME.

Jak widać z pierwszej tabeli program sterujący po wystąpieniu w g-kodzie kodu M3 nie wysteruje bezpośrednio jakiegoś wyjścia, ale zapala flagę C_M3 w rejestrze MEMO. To, co z tym zostanie zrobione dalej zależy wyłącznie od programu w PLC. W najprostszym wariancie może być np. :

<< M C_M3
>> OUT 1

Czyli bezpośrednie przekierowanie na OUT 1.

Stan początkowy rejestru możemy zmieniać dyrektywą #SET_M ustawiającą dane bity w momencie nawiązania połączenia. W monecie wystąpienia E_STOP bity 0-15 są automatycznie zerowane natomiast pozostałe zachowują wartość. Jeśli chcemy wyzerować górne 16 bitów (16-31) musimy ustawić bit CLR_MEM_H (rejestr CONTROL 2).

Rejestr TIMER

Kontroler posiada 8 timerów każdy może odmierzać czas od 0.001 do 65 sekund. Każdy timer ma wyjście stanu T# oraz dwa wejścia sterujące: ustawiające T#_S oraz kasujące T#_R. Ustawienie „1” na wejściu T#_S powoduje, że do licznika timera jest przepisywana wartość czasu która jest przypisana timerowi. Załóżmy, że przypisano timerowi czas 1s zatem do licznika jest wpisywana wartość 1000, która to, co 1/1000 sek. jest zmniejszana o 1. Wyjście T# stanu timera przyjmuje wartość „1” tak długo jak długo stan licznika timera jest większy od zera. Ustawienie „1” na wejściu T#_R zeruje licznik timera – czyli natychmiast przerywa jego pracę.
Wejścia T#_S oraz T#_R są automatycznie zerowane przed wykonaniem cyklu programu.

Przykład:

```
#SET_TIMER 0 = 1.0        // Ustawiamy czas timera numer zero  na 1 sek.

+< IN 0          // BL=1 jeśli narastające zbocze na IN 0
>> T0_S         // Jeśli BL=1 to startujemy timer 0
// Podpięcie wyjścia stanu timera do OUT 1
<< T0            // BL=T0
>> OUT 1        // OUT1 = BL
```

Zatem jeśli zamkniemy obwód na IN 0 na wyjściu OUT 1 pojawi się 1 sek. impuls. Jeśli zmodyfikujemy powyższy przykład i zamiast „+< IN 0” zrobimy „<< IN 0” uzyskamy timer z podtrzymaniem.

Tablice rejestrów

Rejestr CONTROL 1

Bit	Nazwa	Typ	Opis
0	E_STOP_0	W	Wejścia zgłaszania E_STOP
1	E_STOP_1	W	
2	E_STOP_2	W	
3	E_STOP_3	W	
4	E_STOP_4	W	
5	RESET	W	Wejście przycisku RESET.
6	START	W	Wejście przycisku START.
7	STOP	W	Wejście przycisku STOP.
8	PAUSE	W	Wejście przycisku PAUZA.
9	REF	W	Wejście przycisku cyklu REF (jazda referencyjna)
10	MAT_REF	W	Wejście przycisku cyklu pomiaru wysokości materiału.
11	WH_PULSE	W	Wejście sygnału PULSE impulsatora (enkodera).
12	WH_DIR	W	Wejście sygnału DIR impulsatora.
13	WH_MD0	W	Wejścia sterujące trybem pracy impulsatora: 0 – nieaktywny (000) 1 - Zmiana F (001) 2 - Zmiana S (010) 3 - Zmiana położenia osi 0 (011) 4 - Zmiana położenia osi 1 (100) 5 - Zmiana położenia osi 2 (101) 6 - Zmiana położenia osi 3 (110) 7 - Zmiana położenia osi 4 (111)
14	WH_MD1	W	
15	WH_MD2	W	
16	JOG_L0	W	Wejście Jog w lewo
17	JOG_L1	W	
18	JOG_L2	W	
19	JOG_L3	W	
20	JOG_L4	W	
21	JOG_L5	W	
22	JOG_R0	W	Wejście Jog w prawo
23	JOG_R1	W	
24	JOG_R2	W	
25	JOG_R3	W	
26	JOG_R4	W	
27	JOG_R5	W	
28	JOG_FAST	W	Tryby szybki dla JOG i impulsatora
29	WH_JOG_L	W	Wejście Jog w lewo oraz w prawo. Numer osi wyznaczają bity trybu pracy impulsatora (WH_MD0 - WH_MD2)
30	WH_JOG_R	W	
31	WH_ZERO	W	Zerowanie parametru wyznaczonego przez bity trybu pracy

Bit	Nazwa	Typ	Opis
			impulsatora. Osie są zerowane, natomiast parametry F, S ustawiane na 100%.

Rejestr CONTROL 2

Bit	Nazwa	Typ	Opis
0	LIMIT_L0	W	Wejścia LIMT_L (lewy) dla osi.
1	LIMIT_L1	W	
2	LIMIT_L2	W	
3	LIMIT_L3	W	
4	LIMIT_L4	W	
5	LIMIT_L5	W	
6	LIMIT_R0	W	Wejścia LIMT_R (prawy) dla osi.
7	LIMIT_R1	W	
8	LIMIT_R2	W	
9	LIMIT_R3	W	
10	LIMIT_R4	W	
11	LIMIT_R5	W	
12	HOME_0	W	Wejścia HOME osi.
13	HOME_1	W	
14	HOME_2	W	
15	HOME_3	W	
16	HOME_4	W	
17	HOME_5	W	
18	INDEX_0	W	Wejścia indeks osi.
19	INDEX_1	W	
20	INDEX_2	W	
21	INDEX_3	W	
22	INDEX_4	W	
23	INDEX_5	W	
24	FIFO_WAIT	W	Wejście zatrzymujące wykonywanie komend z kolejki FIFO tak długo jak długo wejście jest w stanie aktywnym.
25	N_WAIT_G0	W	Jeżeli wejście jest w stanie aktywnym to FIFO_WAIT nie dotyczy komendy G0.
26	PROBE	W	Wejście wykorzystywane do pomiaru długości narzędzia, wysokości materiału.
27	CLR_MEM_H	W	Ustawienie bitu czyści (zeruje) górne 16 bitów rejestru MEM
28	TA_PROBE	W	Wejście HOME osi technicznej
29	TA_JOG_R	W	Jog osi technicznej (prawo)
30	TA_JOG_L	W	Jog osi technicznej (lewo)
31	PARK_0	W	Wejście przycisku jazdy do pozycji PARK_0

Rejestr STATE

Bit	Nazwa	Typ	Opis
0	IS_MOVE_0	R	Bity informują, że dana oś jest aktualnie w ruchu.
1	IS_MOVE_1	R	
2	IS_MOVE_2	R	
3	IS_MOVE_3	R	
4	IS_MOVE_4	R	
5	IS_MOVE_5	R	
6	DIR_0	R	Bity informują o aktualnym kierunku jazdy osi.
7	DIR_1	R	
8	DIR_2	R	
9	DIR_3	R	
10	DIR_4	R	
11	DIR_5	R	
12	CORNER	R	Sygnał informuje, że ruch wyhamowuje lub nie osiągnął jeszcze prędkości zadanej.
13	IS_JOG	R	Informuje, że jest wykonywany jog.
14	TA_DIR	R	Informuje o kierunku jazdy osi technicznej
15	TA_REF	R	Informuje, że oś techniczna wykonuje jazdę referencyjną.
16	TA_MOVE	R	Informuje, że oś techniczna jest w ruchu.
17	F_IS_100	R	Informuje, że procentowe wzmocnienie F jest 100%
18	IS_G1	R	Wykonywany jest ruch G1
19	IS_G0	R	Wykonywany jest ruch G0
20	REF_0	R	Jazda referencyjna do momenty wystąpienia stanu aktywnego na wejściu HOME osi
21	REF_1	R	Jazda referencyjna do momenty zaniku stanu aktywnego na wejściu HOME osi.
22	REF_2	R	Jazda referencyjna do momenty wystąpienia stanu aktywnego na wejściu INDX osi.
23	REF_3	R	Jazda referencyjna do momenty wystąpienia stanu aktywnego na wejściu PROBE
24	REF_4	R	Jazda referencyjna do momenty zaniku stanu aktywnego na wejściu PROBE.
25	IS_EXE	R	Informuje, że wykonywany jest program
26	IS_PAUSE	R	Program w stanie pauzy.
27	IS_AUTO	R	Ruch w trybie automatycznym.
28	RUN	R	Kontroler w stanie „RUN”
29	S_IS_100	R	Informuje, że procentowe wzmocnienie S jest 100%
30	E_STOP	R	Wystąpiło zatrzymanie przez E_STOP
31	MOVE_INIT	R	Bit ustawiany każdorazowo po zainicjowaniu ruchu G1. Kasowany automatycznie po wykonaniu pętli programu przez PLC

Rejestr TIMER

Bit	Nazwa	Typ	Opis
0	T0	R	Zawierają aktualny stan timera.
1	T1	R	
2	T2	R	
3	T3	R	
4	T4	R	
5	T5	R	
6	T6	R	
7	T7	R	
8	T0_S	W	Wejście ustawiające timer.
9	T1_S	W	
10	T2_S	W	
11	T3_S	W	
12	T4_S	W	
13	T5_S	W	
14	T6_S	W	
15	T7_S	W	
16	T0_R	W	Wejście kasujące timer.
17	T1_R	W	
18	T2_R	W	
19	T3_R	W	
20	T4_R	W	
21	T5_R	W	
22	T6_R	W	
23	T7_R	W	
24	PULSE_0	R	Przebieg o częstotliwości 50 Hz
25	PULSE_1	R	Przebieg o częstotliwości 25 Hz
26	PULSE_2	R	Przebieg o częstotliwości 12,5Hz
27	PULSE_3	R	Przebieg o częstotliwości 6,25 Hz
28	PULSE_4	R	Przebieg o częstotliwości 3,12 Hz
29	PULSE_5	R	Przebieg o częstotliwości 1,56 Hz
30	PULSE_6	R	Przebieg o częstotliwości 0,78 Hz
31	REMOTE_IN	R	Wejście modułu zdalnych wejść.

Rejestr MEMO

Bit	Nazwa	Typ	Opis
0-31	M	R/W	Rejestr pomocniczy – pamięć podręczna.

Rejestr IN

Bit	Nazwa	Typ	Opis
0-31	IN	R	Rejestr wejść fizycznych. Liczba rzeczywistych dostępnych bitów zależy typu kontrolera.

Rejestr OUT

Bit	Nazwa	Typ	Opis
0-31	OUT	W	Rejestr wyjść fizycznych. Liczba rzeczywistych dostępnych bitów zależy typu kontrolera.

Funkcje zapisu/odczytu rejestrów PLC z poziomu makr.

num – numer bitu (0-31).
devname – nazwa bitu zdefiniowana w PLC.
st – docelowy stan bitu.

Rejestr	R/W	Funkcja / procedura
MEMO	W	procedure SetMemo(num:cardinal; st:boolean)
	W	procedure SetMemoN(devname:string; st:boolean)
	R	function Memo(num:cardinal):boolean
	R	function MemoN(devname:string):boolean
IN	R	function Input(num:cardinal):boolean
	R	function InputN(devname:string):boolean
OUT	R	function Output(num:cardinal):boolean
	R	function OutputN(devname:string):boolean
TIMER	W	procedure SetTimer(num:cardinal; time:extended) Procedura ustala czas timera gdzie: num – numer timera (0-7), time – czas timera w sekundach (0.001 – 65 sek.)

Okno PLC.

Ikony w zakładce „Program” w oknie PLC

1 2 3 4 5

Ikona	Opis	Funkcja
1	Przywróć bieżący	Otwiera w edytorze plik bieżący czyli plik o nawie PLC.pdt z katalogu aktualnego profilu - zazwyczaj jest to katalog ProfilDef.
2	Zapisz bieżący	Zapisuje program jako plik PLC.pdt w katalogu aktualnego profilu. Program przed zapisem jest zawsze kompilowany dlatego nie da się zapisać programu który ma np. błędy składniowe.
3	Otwórz dowolny	Otwiera plik z programem z dowolnej lokalizacji
4	Zapisz jako	Zapisuje plik w dowolnej lokalizacji.
5	Kompiluj	Kompilacja programu. Umożliwia na bieżąco sprawdzanie poprawności programu.

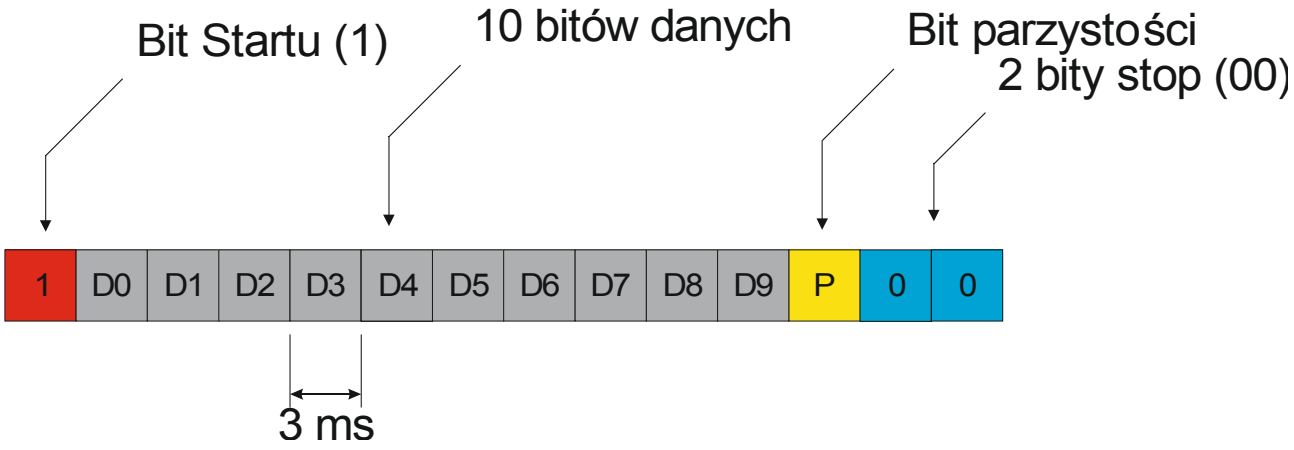
Pierwsze uruchomienie po instalacji.

Jeśli program jest instalowany pierwszy raz (lub pierwszy raz w wersji z PLC) plik bieżący PLC.pdt nie istnieje – nie jest instalowany przez instalator. W takim przypadku przy otwarciu okna PLC dostaniemy komunikat czy otworzyć plik PLC_def.txt z lokalizacji InitFiles. Możemy kliknąć na „Tak” następnie należy zmodyfikować program do własnych celów i ikonką (2) zapisać jako bieżący plik. W domyślnym pliku wejścia HOME, oraz wejścia przycisków, a także wyjścia O1, O2 są skonfigurowane tak aby było jak we wcześniejszych wersjach. Należy skonfigurować tylko przeznaczenie wejść LIMIT oraz ustawić czasy dla wyjść np. rozpędzanie / hamowanie dla O1. Należy zwrócić uwagę też na to, że nie ma już jog-a osi technicznej z klawiatury (wbudowanego w program) – jeśli takowy chcemy trzeba go zrobić w PLC.

Moduł zdalnych wejść.

Z myślą o zwiększeniu liczby wejść na takie potrzeby jak JOG czy przyciski sterujące - PLC obsługuje prosty protokół komunikacyjny na wejściu REMOTE_IN (rejestr TIMER).

Budowa ramki danych.



Jak widać ramka składa się z jednego bitu startu (=1), 10-ciu bitów danych, bitu parzystości oraz dwóch bitów stopu (=00). Bit parzystości ustawiany jest gdy w polu danych (D0-D9) ustawiona jest nieparzysta liczba bitów. Ramki muszą być nadawane non-stop jedna za drugą, ale co 10 ramkę należy zrobić przerwę w nadawaniu na czas trwania jednej ramki (42 ms). Przerwa ma na celu dać gwarancję synchronizacji transmisji. Przerwa w nadawaniu powyżej 70 ms jest traktowana jako zakończenie transmisji i zerowane są wtedy odpowiednie bity w rejestrze IN. Czas trwania jednego bitu to 3 ms.

Obsługa na poziomie PLC.

Dane odebrane przez PLC widoczne są w rejestrze IN jako bity 22-31. Tak więc bit D0 widoczny jest jako IN 22 i kolejno aż do D9, które jest pod IN 31. Dalej pod te wejścia podpinamy odpowiednie funkcje.

Przykład podpięcia modułu oraz kilku funkcji:

```
<< IN 5              // Moduł zdalnych wejść podpięty pod wejście 5
>> REMOTE_IN

<< IN 22             // Podpięcie do modułu JOG XL,XR oraz wejścia trybu szybkiego
>> JOG_L0

<< IN 23
>> JOG_R0

<< IN 30
>> JOG_FAST
```

PPHU ELCOSIMO
Andrzej Woźniak
ul. Zielona 1B
62-110 Damasławek