

### 1.3 Konfiguracja i parametryzacja

Do konfiguracji sterownika PSW służą alternatywnie: panel czołowy, symulator oraz program konfiguracji graficznej. Panel umożliwia konfigurację autonomiczną bez angażowania dodatkowych urządzeń, natomiast symulator i program konfiguracji graficznej wymagają komputera PC.

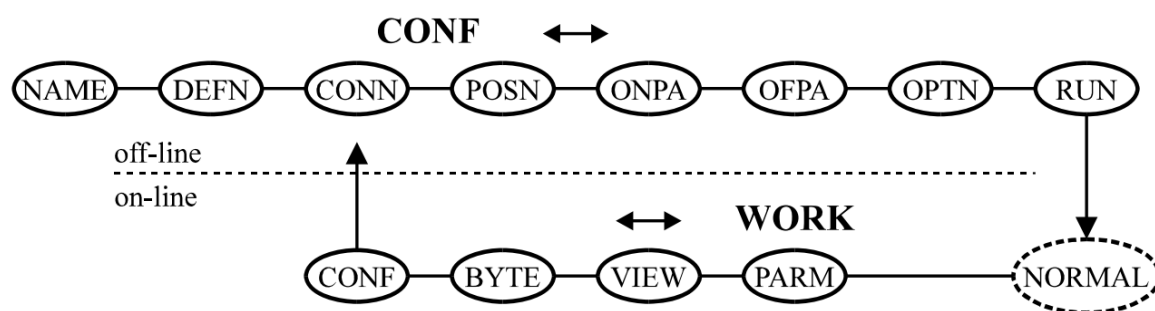
Konfigurację przeprowadza się w stanie *off-line* (inaczej CONF), w którym obsługa procesu jest wstrzymana. W stanie *on-line* (nazywanym WORK), sterownik prowadzi obsługę procesu. Dane konfiguracyjne i parametry zapisane są w nieulotnej pamięci EEPROM.

#### Fazy konfiguracji

W sterowniku PSW fazy konfiguracji są następujące:

- NAME** - nadanie nazwy konfigurowanemu układowi (numeru 001 ... 255);
- DEFN** - definiowanie bloków funkcyjnych (przydzielanie funkcji dla bloków);
- CONN** - łączenie wejść i wyjść bloków (zawsze łączymy wejście z wyjściem);
- POSN** - pozycjonowanie - określenie kolejności obliczeń bloków algorytmicznych;
- OFPA** - ustawianie końcowych wartości parametrów typu *off-line*;
- ONPA** - ustawianie początkowych wartości parametrów typu *online*;
- OPTN** - wybór układów opcjonalnych (*preset, downloading*);
- RUN** - analiza wprowadzonej konfiguracji i przejście do obsługi procesu.

Wyboru fazy dokonuje się z tzw. *nadrzędnego poziomu konfiguracji* przez naciskanie odpowiednich klawiszy. Przejścia pomiędzy poszczególnymi fazami pokazano poniżej.



Rys. 2 Fazy konfiguracji sterownika PSW.

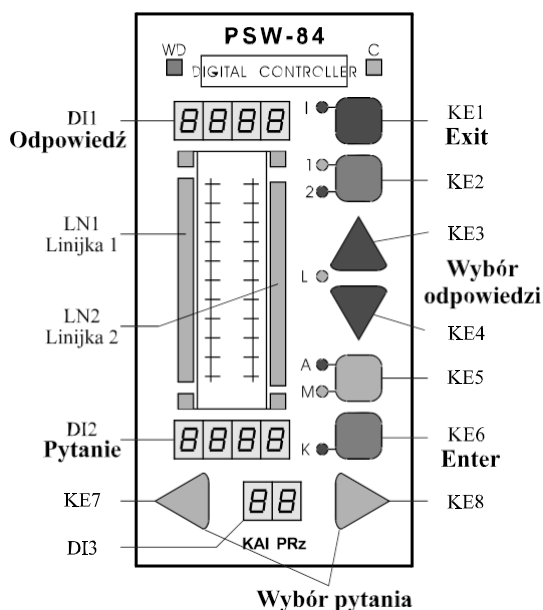
W ramach każdej fazy można określić tryb dostępu do pamięci EEPROM. Są to:

**read** - tylko odczyt danych, **set** - zapis lub odczyt danych konfiguracyjnych

Tryb *read* służy głównie do kontroli danych.

## Panel operatorski

Konfiguracja sterownika odbywa się w oparciu o serię pytań i odpowiedzi. Dolna para klawiszy KE7, KE8 oraz wskaźnik DI2 umożliwiają przechodzenie przez kolejne pytania, które stawia sterownik (w pytaniach o parametry wykorzystywany jest dodatkowo wskaźnik DI3). Wybór właściwej odpowiedzi, następuje przy pomocy bocznych klawiszy KE3, KE4 i górnego wskaźnika DI1. Utrzymywanie naciśniętego jednego klawisza danej pary powoduje, że kolejne pytania/odpowiedzi przewijają się cyklicznie. Proces ten można przyspieszyć, poprzez dodatkowe naciśnięcie drugiego klawisza (dłuższe przeskoki w pytaniach/odpowiedziach). Do wejścia i wyjścia z danej fazy służą klawisze KE6 - *Enter* i KE1 - *Exit*. Ich gotowość sygnalizują diody LED: K oraz I. Podczas konfiguracji na linijkach widnieje wzór paskowy („zebra”).



Rys. 3 Panel operatorski.

Poziom nadrzędny:

pytanie: **NAME, DEFN,..., RUN** -

faza odpowiedź: **read, set**

Aktualne dane można zmieniać tylko wówczas, gdy jesteśmy w trybie *set* i jeżeli sprzętowy przełącznik blokady konfiguracji CE znajduje się w pozycji *on* (zezwozenie).

## Nadanie nazwy - NAME

Nazwa konfiguracji ma znaczenie identyfikacyjne przy przechowywaniu jej w pliku na dysku. Wybierając na poziomie nadrzędnym fazę NAME i naciskając przycisk *Enter* (KE6) na wskaźnikach pytań i odpowiedzi mamy:

pytanie:	<b>NAME</b>	- faza (pytanie, wyjście przez KE1)
odpowiedź:	<b>none, 1, 2, ..., 255</b>	- numer (odpowiedź, zmiana KE3, KE4)

**none** oznacza brak nazwy i występuje przy wyzerowanej pamięci EEPROM. Bez ustawienia klawiszami KE3, KE4 konkretnej liczby nie można przejść do następnych faz. Po wybraniu nazwy naciskamy *Exit* (KE1). Zostaje ona zachowana i regulator wraca na poziom nadrzędny.

## Definiowanie funkcji bloków - DEFN

Zakłada się, że konfigurowany układ jest przedstawiony w formie schematu blokowego zbudowanego z odpowiednio połączonych bloków funkcyjnych. W fazie DEFN dokonuje się wyboru bloków do układu przyporządkowując im funkcje. Po

ustawieniu na wskaźnikach DEFN-set i naciśnięciu *Enter* (KE6) mamy:

pytanie:       **b01, b02, ..., c01, c02, ...**       - nazwa bloku (identyfikator, ID)  
odpowiedź:   **ndef, ABS, ADD, ..., SP, PID, ...**   - przyporządkowana mu funkcja

Definiowanie polega więc na ustawieniu właściwych par *blok-funkcja*.

**ndef.** Blok ma *status niezdefiniowany*. Status ten posiadają bloki, którym nie przypisano żadnej funkcji.

Tylko bloki zdefiniowane, którym przyporządkowano konkretne funkcje, uczestniczą w następnych fazach konfiguracji.

- Bloki i funkcje w fazie DEFN

Blok - pytanie	Funkcja – odpowiedzi	Uwagi
b01,..., b80	ndef, ABS, ADD,..., TOTI	bloki proste
KE1,..., KE8	KEY, KEYC	klawisze panelu
LN1, LN2	DISP, LED, LED0, LED1	linijka diodowa
AO	AO, AOC	wyjścia analogowe
BO	BO, BOC, BOT, BOL	wyjścia binarne
TRDV	ndef, YES	komunikacja pionowa
SGDH	ndef, YES	komunikacja pozioma
RTC	ndef, YES	zegar czasu rzeczywistego
c01,...,c40	ndef, 3AMT,..., SERV, SP	bloki złożone

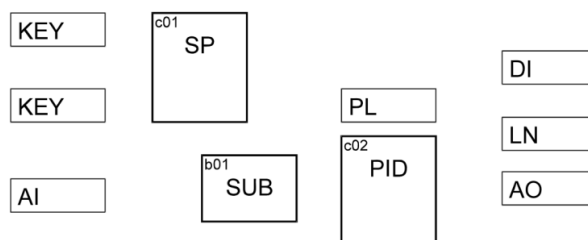
- Bloki nie występujące w DEFN (mają stałe funkcje)

w wejścia	AI, BI	komunikacja	VERT, HORI
alarmy	AL	parametry	PL, PD, PB
diody LED	LD1, ..., LD6	stałe	CNST

### Bloki złożone

Bloki złożone muszą być definiowane *sukcesywnie*, tzn. najpierw c01, potem c02, c03 itd. (c - *complex*). Pamięć rezerwowana jest dynamicznie stosownie do wymagań funkcji. Zmiana funkcji bloku złożonego lub jego eliminacja poprzez *ndef* jest możliwa tylko wówczas, gdy jest on ostatnim w łańcuchu bloków złożonych. Chcąc zmienić funkcję bloku wcześniejszego należy w pierwszej kolejności nadać status *ndef* wszystkim blokom następnym (postępując od końca), a następnie zmienić funkcję tego bloku i ponownie odtworzyć funkcje bloków następnych.

Po DEFN konfigurowany schemat stanowi zbiór osobnych (nie połączonych ze sobą) bloków funkcyjnych (zob. Rys. 4).



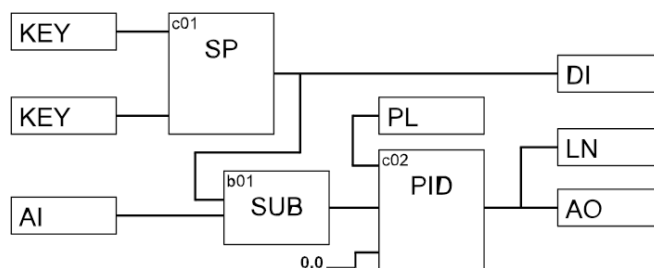
Rys. 4 Faza definiowania (DEFN)

### Łączenie wejść i wyjść - CONN

Zdefiniowane bloki należy połączyć zgodnie ze schematem blokowym. Po ustawieniu CONN-set i naciśnięciu *Enter*, stan wskaźników wygląda następująco (przykład):

pytanie:	<b>KE1.L,</b>	<b>b01.1 [EXOR]</b>	- wejście
odpowiedź:	<b>b01.B [EXOR]</b>	<b>KE1.1</b>	- wyjście, stała, <i>nc</i>

Podstawowe oznaczenie składa się z nazwy bloku i numeru/symbolu wejścia/wyjścia jak na schemacie funkcji. W przypadku bloków prostych i złożonych oznaczenie uzupełnia się o nazwę funkcji (w nawiasie kwadratowym). Jest ono wyświetlane na zmianę z funkcją, w cyklu 3 + 1 sekund. Czyli z pełnego oznaczenia b01.1 [EXOR] część podstawowa b01.1 pojawia się przez 3 sekundy, a EXOR przez 1 sekundę. *nc* oznacza, że wejście nie jest połączone (*not connected*). Zbiór możliwych wyjść (odpowiedzi), które regulator przedstawia do wyboru, obejmuje wyjścia tego samego rodzaju, co wejście wskazane w pytaniu. Na wskaźnikach pojawiają się więc tylko pary *analogowe-analogowe* lub *binarno-binarne*. Każde wejście może być połączone tylko z jednym wyjściem, ale kilka wejść może jednocześnie korzystać z jednego wyjścia. W fazie CONN bloki funkcyjne, które po DEFN występowały osobno, zostają połączone zgodnie z wprowadzonym schematem.



Rys. 5 Faza łączenia (CONN)

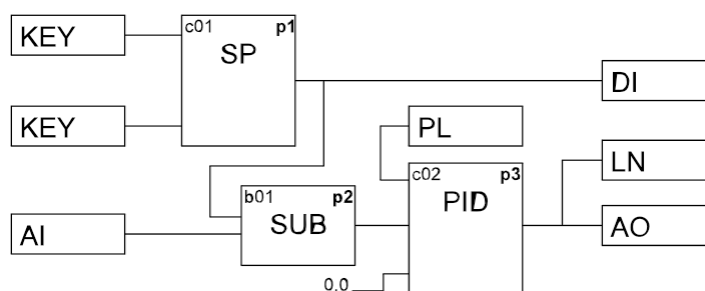
### Pozycjonowanie - POSN

W fazie tej ustala się kolejność obliczeń zdefiniowanych bloków algorytmicznych prostych i złożonych, czyli ich pozycje. Pozycji jest tyle ile razem bloków prostych i złożonych, tzn.  $80 + 40 = 120$ . Są one oznaczane przez p01, p02 itd. Po POSN-set i *Enter* mamy (przykład):

pytanie:	<b>p1,</b>	<b>p2</b>	- pozycja
odpowiedź:	<b>b01 [EXOR]</b>	<b>c01 [PID], npos, inst, delt</b>	- blok [funkcja]

Początkowo wszystkie pozycje mają status niezajętych - *npos* (*not positioned*). Pierwszym stanem wskaźników będzie więc *p01-npos*. Na *p01* i następnych pozycjach należy poumieszczać właściwe bloki.

Blok można umieścić tylko na *jednej pozycji*. Znika on potem z listy odpowiedzi. Sterownik będzie realizował obsługę bloków na kolejnych pozycjach, aż do napotkania pierwszej niezajętej - *npos*. Ewentualne dalsze bloki za *npos* nie są brane pod uwagę. W fazie POSN zostaje określona kolejność obsługi bloków prostych i złożonych (definiowalnych). Kolejność obsługi bloków wejściowych i wyjściowych jest z góry ustalona.



Rys. 6 Faza pozycjonowania (POSN)

#### Edycja - *npos*, *inst*, *deli*

***npos*.** - przywrócenie statusu *npos* dotychczas zajętej pozycji zwalnia umieszczony na niej blok. Powraca on do listy odpowiedzi i można go wykorzystać ponownie.

***inst*.** - służy do wstawiania *npos* na odpowiednią pozycję (*insert*). O ile na pytanie o pozycję  $p_i$  wybierze się odpowiedź *inst* i naciśnie *Enter* (KE6), to  $p_i$  otrzyma status *npos*, a dotychczasowe bloki na pozycjach począwszy od  $p_i$  zostaną przesunięte o jedną pozycję w prawo (z  $p_i$  na  $p_{i+1}$  itd.). Stanem wskaźników stanie się  $p_i-npos$ . Można teraz umieścić na  $p_i$  jeden z wolnych bloków (lub pozostawić *npos*, jeżeli zamierza się testować tylko część układu).

***delt*.** - usuwa blok lub *npos* ze wskazanej pozycji (*delete*). Po ustawieniu  $p_j-delt$  i *Enter*, bloki za  $p_j$  zostają dosunięte o jedną pozycję w lewo. Blok z  $p_{j+1}$  zajmie pozycję  $p_j$  i jego nazwa pojawi się jako odpowiedź. Usunięty blok z  $p_j$  powraca do listy odpowiedzi.

Uwagi:

- Jeżeli w układzie jest przewidywane sprzężenie zwrotne z bloków na dalszych pozycjach, to do obliczeń zostaną wzięte wartości z poprzedniego cyklu. Ma to przede wszystkim znaczenie w układach logicznych, gdzie istotna jest kolejność obliczeń bloków.
- Bloki wejściowe są obsługiwane na początku cyklu, a wyjściowe na końcu. W ramach bloków wejściowych pierwsze są wejścia analogowe, potem binarne, klawisze, RTC oraz odbiorniki komunikacyjne. Podobnie jest z blokami wyjściowymi.
- Jeżeli w fazie POSN nastąpił powrót do DEFN i funkcję pewnego bloku algorytmicznego zastąpiono inną, to jego dotychczasowa pozycja zostaje zwolniona - *npos*, jak również kasowane są dotychczasowe połączenia do tego bloku. Podobnie, jeśli blok został wyeliminowany przez *ndef*, to jest on również usunięty ze swej pozycji - *npos* i skasowane są wszystkie połączenia do niego. Eliminację tak

wprowadzonych *npos* realizuje *delt*.

### Ustawianie parametrów - OFPA, ONPA

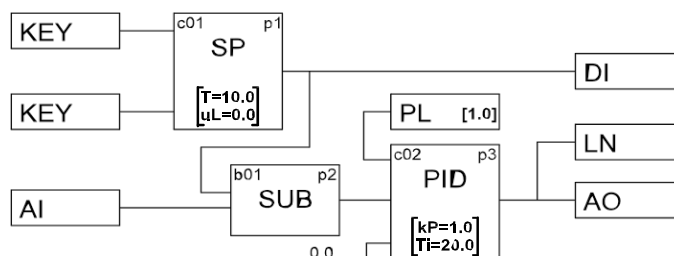
Parametry zdefiniowanych bloków funkcyjnych można ustawiać w fazach OFPA i ONPA. Faza OFPA dotyczy parametrów typu *off-line*, które podczas obsługi procesu pozostają stałe. Na schematach identyfikuje je symbol [C] (CONF). W fazie ONPA są ustawiane wstępne wartości parametrów *on-line* oznaczanych na schematach przez [W] (WORK). Można je potem zmieniać nie przerywając obsługi procesu.

W fazach OFPA, ONPA uczestniczy dodatkowo dolny wskaźnik DI3, którym wyświetla symbol ustawianego parametru. Dla przykładu stan wskaźników może być następujący:

pytanie:	<b>DI1.1 - yL,</b>	<b>c03 [PID] - KP</b>	- blok [funkcja] - parametr	DI2 - DI3
odpowiedź:	<b>10</b>	<b>5.0</b>	- wartość parametru	DI1

Symbole parametrów są takie, jak na schematach funkcji. Ewentualne różnice wynikają z ograniczonych możliwości wyświetlaczy 7-segmentowych. W kilku przypadkach istnieją pewne odstępstwa od powyższego formatu, ale nie powinny one budzić wątpliwości.

Parametry należą do określonych przedziałów, co uwzględnia zbiór odpowiedzi. Nowo zdefiniowany blok otrzymuje określone wstępne wartości parametrów. Są one pierwszymi odpowiedziami, które pojawiają się na wskaźniku. Wskaźniki cyfrowe i linijki diodowe mają odrębne zestawy parametrów dla obydwu torów (wejść). Rozróżnia się je dodając ".1" lub ".2" po nazwie bloku. W przypadku parametrów uniwersalnych PL, PD i PB symbol parametru pojawia się wprost na wskaźniku DI2.



Rys. 7 Faza parametryzacji (OFPA, ONPA)

### Ładowanie układów opcjonalnych - OPTN

W tej fazie istnieje możliwość przepisania z komputera PC do pamięci EEPROM danych konfiguracyjnych wcześniej przygotowanej struktury (z pliku). W pamięci można także ustawić dane początkowe (*preset*). Po OPTN-set i naciśnięciu *Enter* (KE6) mamy:

pytanie:	<b>OPTN</b>
odpowiedź:	<b>no, PRST, LOAD</b>

**PRST.** Dane początkowe dla kolejnych faz są następujące:

NAME - *none*, brak nazwy (numeru) dla konfiguracji  
 DEFN - *ndef* dla bloków algorytmicznych, pierwotne dla bloków we/wy  
 CONN - *nc* lub stałe CNST zależnie od funkcji

POSN - *npos* na wszystkich pozycjach

OFPA, ONPA - wstępne wartości parametrów

Gdyby teraz nadać jakąkolwiek nazwę (NAME) i przejść do obsługi procesu (RUN), to wejścia analogowe i binarne zostaną wyzerowane, a wskaźniki, linijki i diody LED zgaszone.

**LOAD.** Sterownik oczekuje teraz na dane konfiguracyjne, które w grupie 16 komunikatów powinny być przekazane przez komputer PC. Po stronie komputera służy do tego program konfiguracji graficznej (opcjonalny) lub symulator z programem TRANS. Dane konfiguracji NAME-*nnn*, gdzie *nnn* oznacza liczbę 001, 002 itd., mieszczą się w pliku *p84\_conf.nnn* symulatora. Po odebraniu konfiguracji sterownik przechodzi automatycznie do READ-*name*. Przepisanie gotowych danych z dysku jest stosowane dla powtarzalnych (typowych) układów sterowania.

### Analiza konfiguracji

Po ustawieniu RUN na poziomie nadrzędnym pierwsze naciśnięcie *Enter* (KE6) wyświetla żądanie potwierdzenia *YES*. Po ponownym potwierdzeniu *Enter*, PSW przeprowadza analizę danych konfiguracyjnych. Na wskaźnikach jest wtedy sygnalizowane *MEM-BUSY*, a w tym czasie wykonywane jest:

- sprawdzenie kompletności połączeń wejść wszystkich bloków;
- badanie, czy połączenia następują wyłącznie między blokami obsługiwanymi;
- kontrola, czy bloki algorytmiczne mają dołączone wyjścia (są wykorzystane);
- kontrola, czy wszystkie zdefiniowane bloki algorytmiczne są obsługiwane.

Blokami obsługiwanymi są bloki wejściowe, wyjściowe oraz te z bloków algorytmicznych, które umieszczono na pozycjach p01, p02, ..., itd. aż do pierwszego wystąpienia *npos*.

Podczas analizy mogą pojawić się następujące komunikaty (więcej informacji w [1]):

- **nCON-err** - błąd, brak połączenia lub błędne połączenie;
- **nUSE-warn** - ostrzeżenie, blok niewykorzystany;
- **nPOS-warn** - ostrzeżenie, blok nieobsługiwany.

Po *Enter* sterownik kontynuuje analizę konfiguracji. W przypadku bezbłędnej konfiguracji PSW rozpoczyna obsługę procesu. Ze wskaźników znika *MEM-BUSY*, a pojawiają się wartości sygnałów (zależnie od wprowadzonej konfiguracji i wykorzystania elementów wizualnych panelu).

### Parametryzacja on-line

W fazie WORK sterownik PSW prowadzi ciągłą obsługę procesu. Równolegle z nią istnieje możliwość realizacji następujących operacji:

- zmiana wartości parametrów typu *on-line* (oznaczonych [W]),
- przeglądanie wartości zmiennych na wejściach bloków,
- kontrola poszczególnych bajtów w pamięci (tylko dla serwisu).

Można również powrócić do stanu konfiguracji dla ew. modyfikacji układu. Obsługa procesu zostanie wtedy jednak przerwana.

### Tryb obsługi panelu

Jest niemal taki sam jak podczas konfiguracji, z tym że teraz tylko jeden klawisz (KE6) rozpoczyna i kończy daną operację (*Enter/Exit*). Tryb obsługi jest następujący:

1. Nacisnąć KE6 i przytrzymać przez ok. 10 sekund, aż na wskaźniku DI3 pojawi się pulsujący napis *PV*. Sygnalizuje to gotowość do operacji *on-line* lub przejścia do konfiguracji. Po zwolnieniu KE6 napis *PV* przestaje migać.
2. Jednocześnie dwa główne wskaźniki przyjmują jeden z następujących stanów:
  - PARM - *set/read* - parametryzacja *on-line*
  - VIEW - *read* - przeglądanie zmiennych
  - BYTE - *set/read* - kontrola bajtów w pamięci
  - CONF - *set/read* - przejście do konfiguracji.

Wyboru PARM, VIEW, BYTE lub CONF dokonuje się klawiszami KE3, KE4 (bocznymi). Napis *set/read* sygnalizuje zamknięcie lub otwarcie przełącznika sprzętowego CE. Zmiana *set* na *read* przez naciskanie klawiszy nie jest więc możliwa. Jeżeli konfiguracja została zablokowana, CONF-*set/read* nie pojawi się.

3. Przejście do wybranej operacji następuje przez naciśnięcie klawisza KE7 lub KE8. Czynności w ramach PARM, VIEW, BYTE opisano niżej. W przypadku CONF po pierwszym naciśnięciu KE7 lub KE8 zapala się *MODE-CONF*, a po drugim (potwierdzenie) sterownik przechodzi na nadrzędny poziom konfiguracji do NAME-*set/read*. Napis *MODE-CONF* sygnalizuje, że obsługa procesu zostanie przerwana (*off-line*).
4. Zakończenie operacji PARM, VIEW, BYTE następuje przez ponowne naciśnięcie KE6. Jeżeli jednak żadnego z klawiszy nie naciśnięto przez 20 sekund, to zakończenie następuje automatycznie. Na wskaźnikach pojawiają się wartości zmiennych, a klawiszom przywrócone zostają funkcje zależne od wprowadzonej konfiguracji.

Linijki diodowe, diody indywidualne i pozostałe klawisze panelu nie biorące udziału w operacjach *on-line* zachowują funkcje przewidziane w konfiguracji. Nadal więc można (przynajmniej częściowo) obserwować proces oraz ingerować w pracę sterownika.

### Ustawianie parametrów *on-line* - PARM

Parametry *on-line* występują na schematach funkcji z symbolem [W]. Przeznaczenie wskaźników i klawiszy podczas parametryzacji jest takie samo jak w fazie konfiguracji (za wyjątkiem *Enter/Exit*). Na wskaźnikach dolnych pojawia się pytanie, czyli *blok [funkcja]* - *symbol parametru*, a na górnym odpowiedź - *wartość*. Zmiana wartości jest możliwa, gdy parametryzację rozpoczęto od PARM-*set* i zamknięty jest przełącznik PE.

### Przeglądanie wejść - VIEW

Przeglądanie jest ograniczone tylko do wejść bloków. VIEW jest przydatne do testowania układu oraz analizy sytuacji nietypowych. Po wybraniu VIEW-*read* i naciśnięciu KE7 lub KE8 wskaźniki przyjmują stan następujący (przykład):

pytanie:	<b>b01.2 [COMP]</b>	<b>LD1.1</b>	- wejście bloku,
odpowiedź:	<b>0.123</b>	<b>Lo</b>	- aktualna wartość

Wyboru wejścia dokonuje się za pomocą klawiszy KE7, KE8. Powrót do normalnej



# 1 Sterowniki wielofunkcyjne PSW

Programowalny sterownik wielofunkcyjny (PSW) to wspólna nazwa kilku urządzeń wykorzystujących oprogramowanie regulatora wielofunkcyjnego RWF [1,3]. Każde z tych urządzeń posiada kilka odmian, wynikających z odmiennej realizacji sprzętowej lub z innej liczby sygnałów obiektowych. Jednak idea pracy, tryb konfiguracji i potencjalne możliwości są wspólną cechą tych urządzeń. Instrukcja zawiera jedynie opis sterownika PSW-84 w wersji standardowej, która jest najbardziej reprezentatywna.

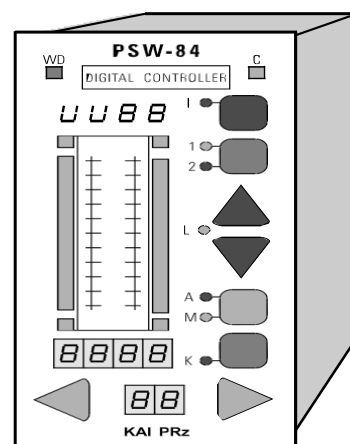
## 1.1 Charakterystyka ogólna

Sterownik PSW jest wielofunkcyjnym konfigurowalnym urządzeniem automatyki umożliwiającym realizację większości zadań jakie są wymagane w sterowaniu procesami technologicznymi. Do zadań tych należy: regulacja PID, sterowanie logiczne, różnorodne obliczenia pomocnicze, sterowanie maszyn i urządzeń, pomiary. PSW może funkcjonować jako programator zegarowy i sekwenster, prowadzić przełączanie, porównywanie, zliczanie, sygnalizację alarmów, rejestrować maksima/minima. Może pracować autonomicznie lub jako element rozproszonego systemu automatyki. Jest przeznaczony do automatyzacji zarówno prostych obiektów jak i złożonych instalacji technologicznych.

**Zastosowania.** Typowe zastosowania sterownika PSW to:

- regulator PID
- sterownik logiczno-sekwencyjny
- kalkulator-wskaźnik

Wszystkie te zadania mogą być realizowane jednocześnie, stąd można w łatwy sposób zaadaptować sterownik do rozmaitych zastosowań.



Rys. 1 Sterownik PSW-84.

**Sygnały obiektowe.** To one w znacznej mierze decydują o funkcjonalności i możliwościach sterownika. PSW-84 w wersji standardowej posiada:

- 8 wejść + 8 wyjść analogowych (prądowych, w zakresie 0/4...20mA)
- 20 wejść + 16 (18) wyjść binarnych (z izolacją galwaniczną, 0/24V).

Pozwala to prowadzić zarówno regulację ciągłą jak i sterowanie logiczne w porównywalnym zakresie.

**Bloki funkcyjne.** Oprogramowanie PSW składa się z bloków funkcyjnych. W PSW-84 jest ich 120 i można im przyporządkować 85 rozmaitych funkcji. Są wśród nich są funkcje arytmetyczne i logiczne, przełączniki, komparatory, itp. Szczegółowy wykaz dostępnych funkcji można znaleźć w pozycji [1]. Podczas programowania, inaczej konfiguracji, bloki funkcyjne zostają połączone w odpowiedni układ, analogicznie jak elementy automatyki konwencjonalnej.

**Konfiguracja.** Konfiguracja sterownika może zostać zrealizowana przy pomocy:

- panelu czołowego - konfiguracja polega na przejściu przez serię pytań i odpowiedzi. Dane konfiguracyjne i parametry zapisywane są w wewnętrznej pamięci EEPROM;
- symulatora - komputer PC symuluje sterownik przechowując dane konfiguracyjne w pliku dyskowym, skąd mogą one zostać przesłane do sterownika (łączem szeregowym);
- programu konfiguracji graficznej - w tym przypadku układ jest budowany myszką na ekranie komputera w postaci graficznej, a następnie plik wynikowy jest transmitowany do sterownika.

**Komunikacja.** PSW posiada 2 kanały komunikacyjne: pionowy (RS-232C/RS-485) do komputera nadrzędnego i poziomy (RS-485) do komunikacji w sieci z innymi sterownikami PSW. Komunikacja pionowa może być prowadzona z wykorzystaniem protokołów:

- MODBUS - odpowiada wszystkim znanym pakietom wizualizacji i sterowania nadrzędnego (m. in. FIX, InTouch, WIZCON);
- TRANS - protokół opracowany dla sterownika PSW.

## 1.2 Bloki funkcyjne

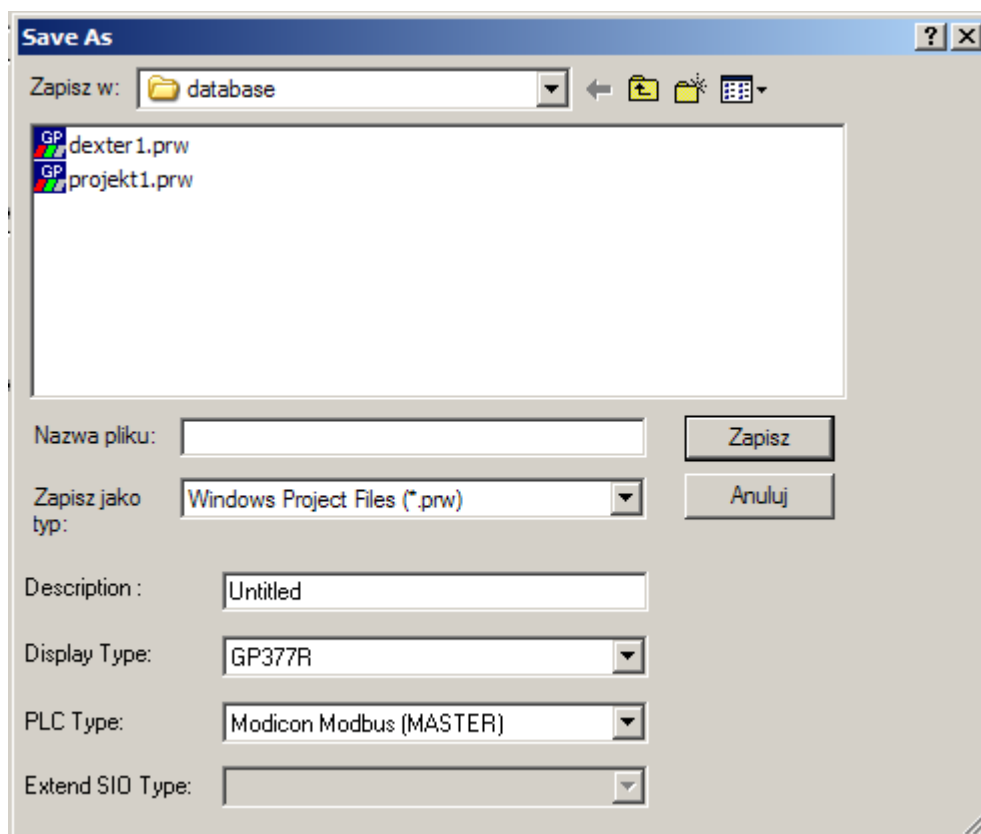
Sterownik PSW-84 zawiera bloki wejściowe, algorytmiczne i wyjściowe.

*Bloki wejściowe* obsługują wejścia obiektowe, klawisze panelu, zegar RTC oraz odbiorniki komunikacyjne. Wytwarzają wewnętrzne zmienne analogowe i binarne, stanowiące jednocześnie dane dla innych bloków.

*Bloki wyjściowe* dokonują przetworzenia zmiennych wewnętrznych na wyjścia obiektowe, stany elementów wizualnych panelu oraz dane wysyłane przez nadajniki komunikacyjne. Zakresy wyjść analogowych odpowiadają przedziałowi 0.0...1.0 zmiennych wewnętrznych.

*Bloki złożone.* Sterownik PSW-84 zawiera:

- 80 bloków prostych, z których każdemu można przyporządkować dowolną z 48 funkcji (zob. Tab. 1);
- 40 bloków złożonych, którym kolejno przyporządkowuje się dowolną z 37 funkcji złożonych (zob. Tab. 2);
- 32 parametry uniwersalne („potencjometry”), 16 parametrów binarnych, stałe, alarmy.



Rysunek 8: Okno zapisu projektu

W polu „Nazwa pliku” nadajemy nazwę całemu projektowi jak i pliku, w którym będzie on zapisany. Jest jeszcze przez zapisem możliwość zmiany typu ekranu dla którego będzie zapisany projekt oraz protokół komunikacyjny ze sterownikiem. Następnie klikamy „Zapisz” i cały plik zostaje zapisany w jednym pliku.

### 3. Przedstawienie użycia wybranych elementów w panelu



Ta część projektu przedstawia przykład użycia wybranych elementów oraz ich własności. Wybrane elementy mogą zostać użyte przy realizacji skomplikowanej wizualizacji pracy różnych procesów technologicznych.

Tworzenie ekranu projektu odbywa się poprzez umieszczenie predefiniowanych elementów na polu rysowniczym, a następnie konfigurowaniu własności ich parametrów. Wszystkie elementy możliwe do zastosowania w projekcie są umieszczone na listwie. Po najechaniu na ikonę elementu wyświetlana jest jego nazwa.

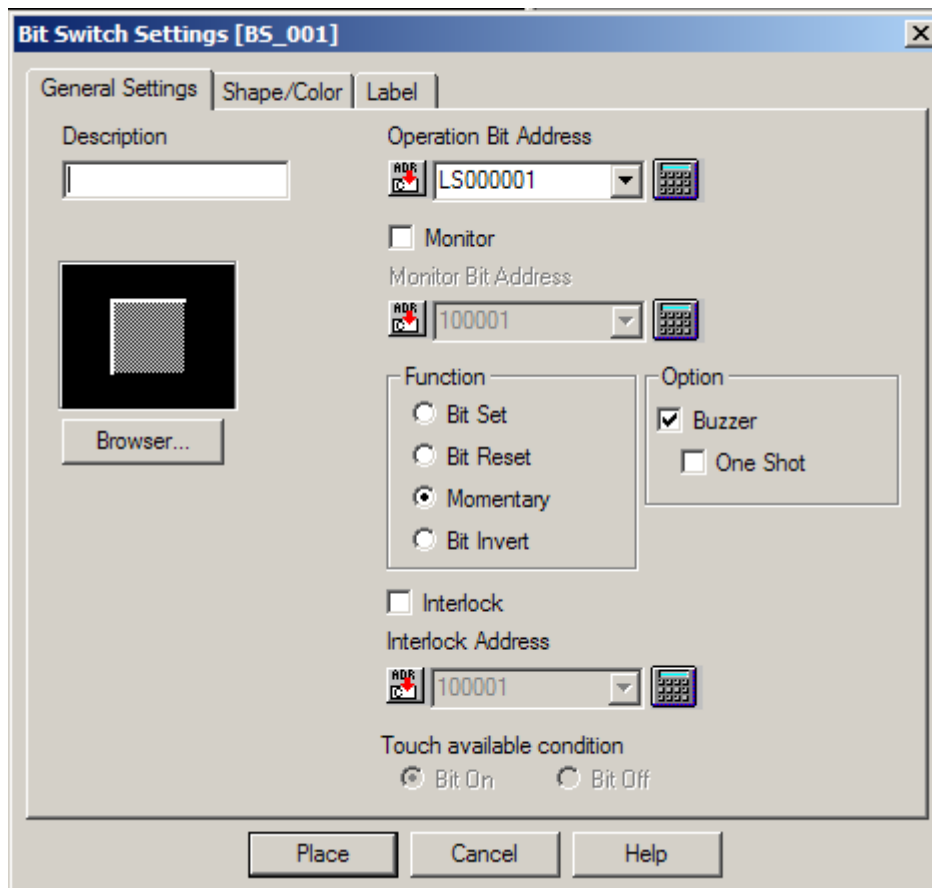


Rysunek 9: Elementy na pasku narzędziowym, możliwe do użycia

## a) Dodanie elementów typu „Bit Switch” oraz „Lamp”

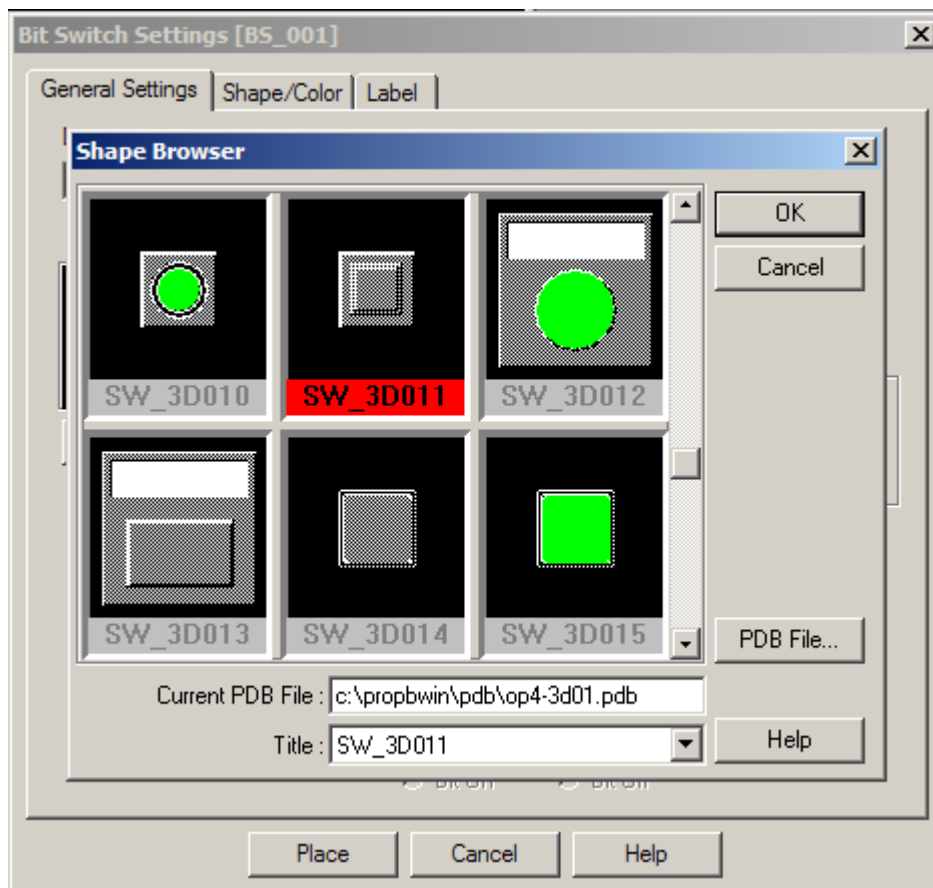
Pierwszym dodanym elementem będzie „Bit Switch” . Używany będzie tutaj z włączoną opcją „momentary” co oznacza, że w czasie trzymania tego elementu pod adresem wpisanym w jego ustawieniach będzie zapisywana „1” logiczna. Dodatkowo dodamy element „Lamp” , dzięki któremu będzie wskazywał jaka wartość logiczna jest wpisana pod danym adresem. Oczywiście adres będzie ten sam co przy Bit Switch aby zaobserwować czy przycisk zmienia wartość pod adresem.

Po wybraniu z paska elementu otwiera się okno ze szczegółowymi ustawieniami wybranego elementu:




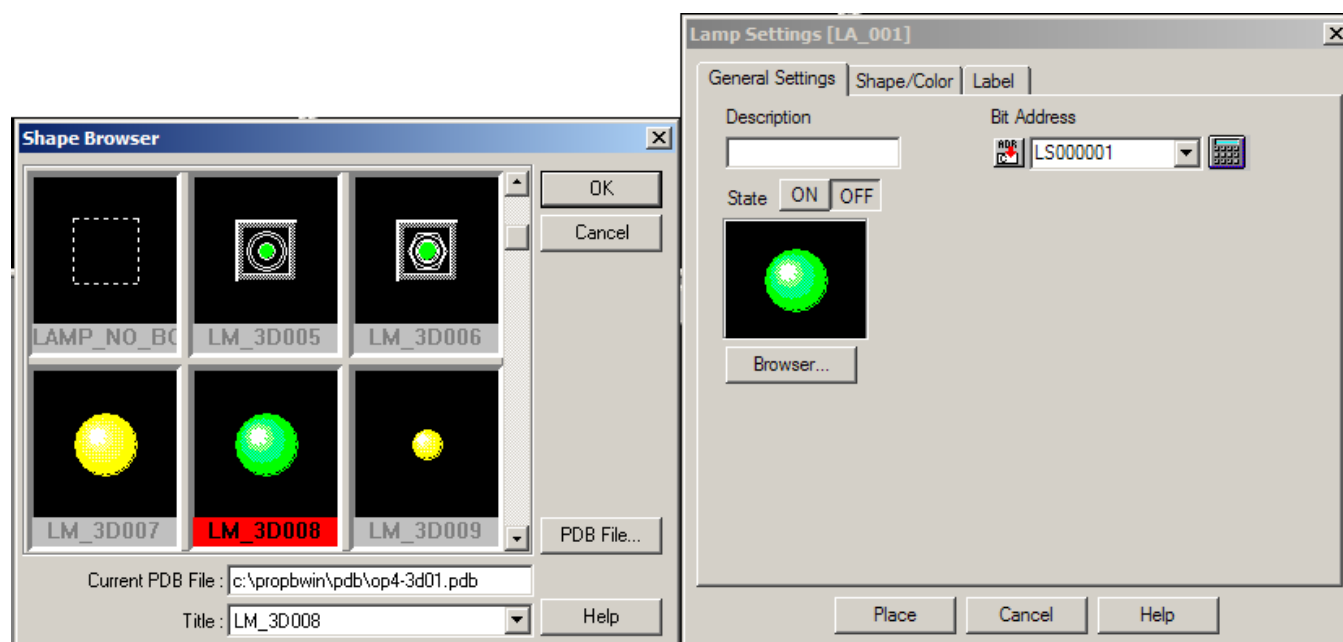
Rysunek 10: Okno ustawień elementu Bit Switch

W polu „Description” można wpisać nazwę elementu. Po naciśnięciu przycisku „Browser” można ustawić ikonę, która będzie wyświetlana jako element Bit Switch. W polu „Operation Bit Address” wpisujemy adres bitu, który ma być zmieniany po naciśnięciu elementu.



Rysunek 11: Wybór ikony symbolizującej element Bit Switch

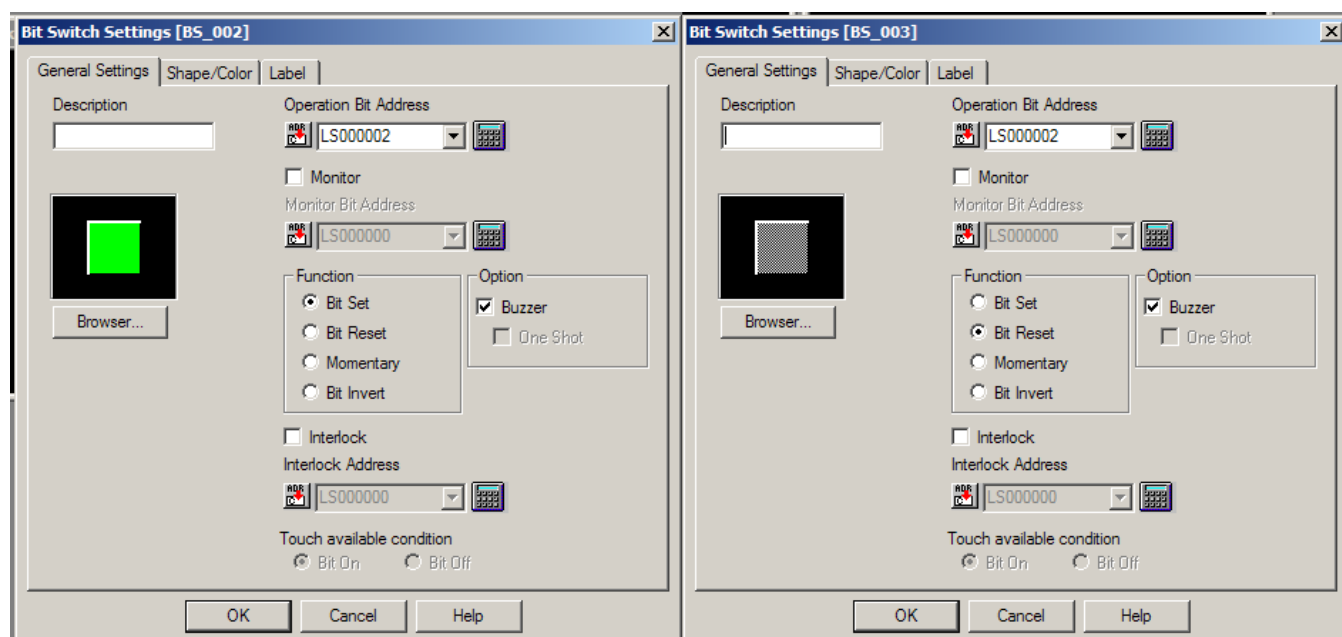
Następnie dodajmy element „Lamp”  aby można na bieżąco monitorować stan wcześniej zmienianego bitu. Po wybraniu elementu należy ustawić w polu Bit Address ten sam adres jak przy bit switch oraz wybrać ikonę symbolizującą ten element.



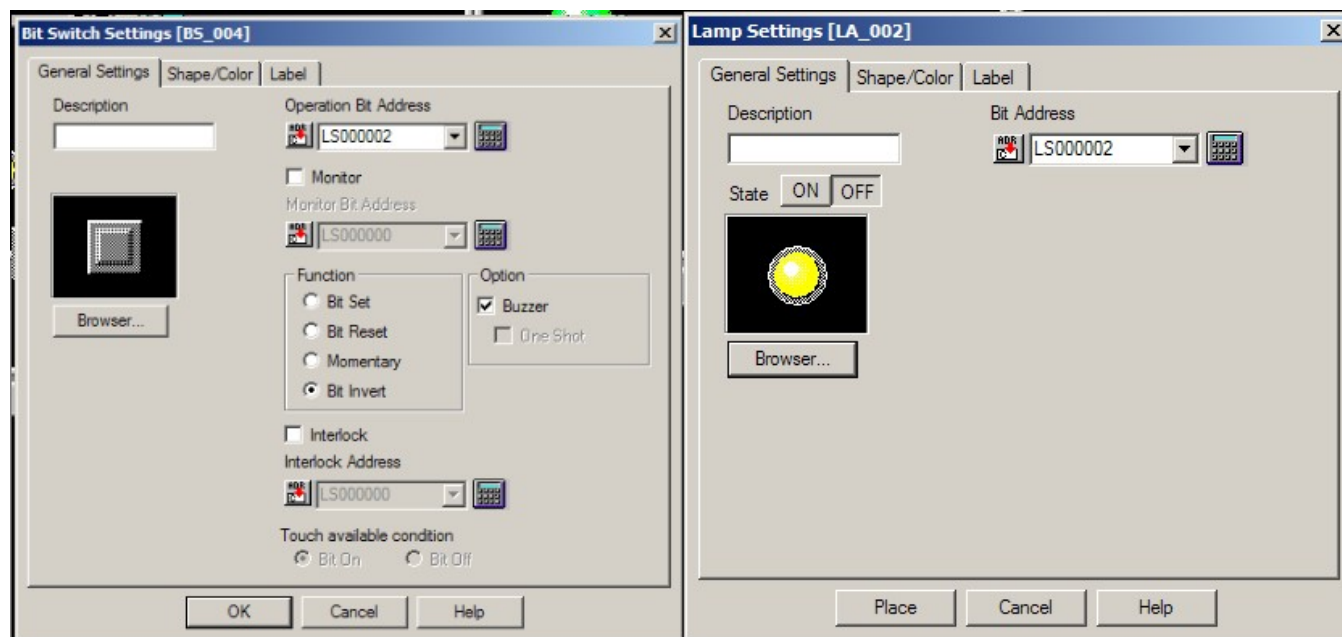
Rysunek 12: Wybór ustawień dla elementu „Lamp”

Następnie dodamy trzy przyciski typu Bit Switch tylko każdy pracujący w innym trybie: pierwszy będzie ustawiał dany bit „Bit Set”, drugi zerował bit „Bit Reset”, a trzeci ustawiał na wartość

przeciwnej niż jest zapisana wartość „Bit Invert”. Na końcu dodamy kolejny element Lamp aby można obserwować działanie klawiszy. Oczywiście wszystkie klawisze będą zmieniały stan bitu pod tym samym adresem jak i element Lamp też będzie „monitorować” ten sam adres.




Rysunek 13: Ustawienie dla dwóch przycisków: w trybie Bit Set i Bit Reset

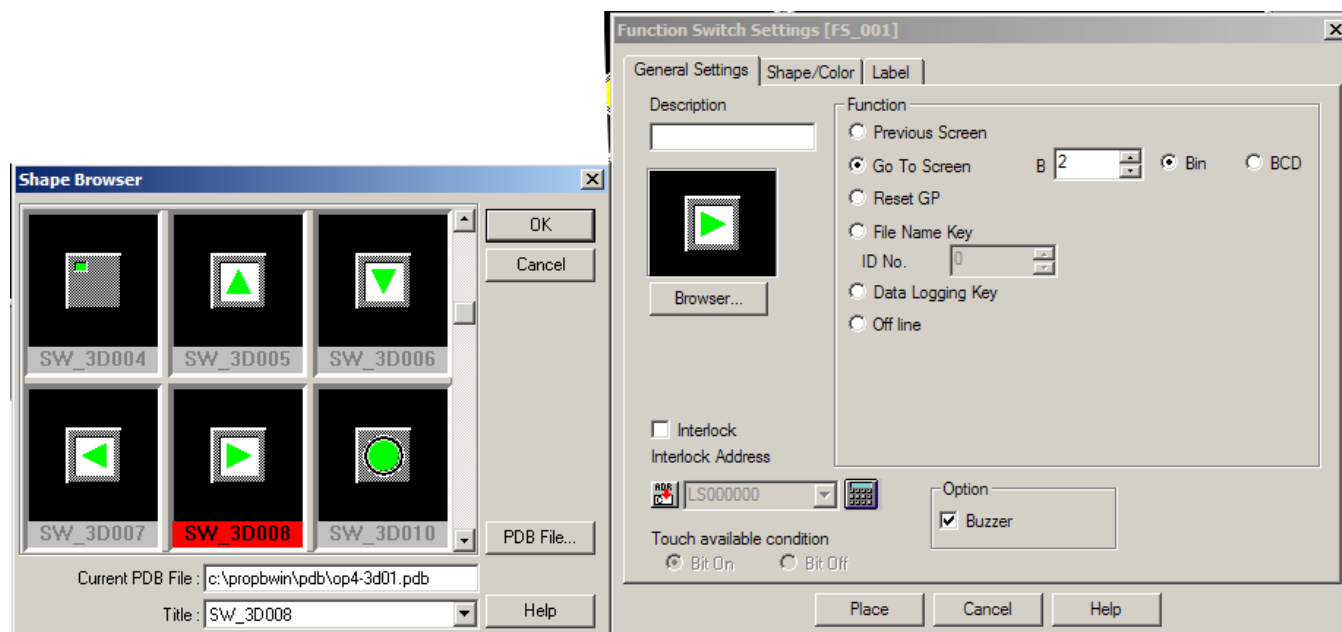


Rysunek 14: Ustawienie dla przycisku w trybie Bit Invert oraz elementu Lamp.

## b) Dodanie elementu „Function SW”

Podczas wizualizacji procesu można tworzyć więcej niż jeden ekran i dowolnie poruszać się pomiędzy nimi. Przechodzenie między ekranami można zrealizować za pomocą elementu „Function SW”.

Aby dodać element typu Function SW należy wcisnąć ikonę . Następnie ukazuje się okno konfiguracyjne tego elementu.

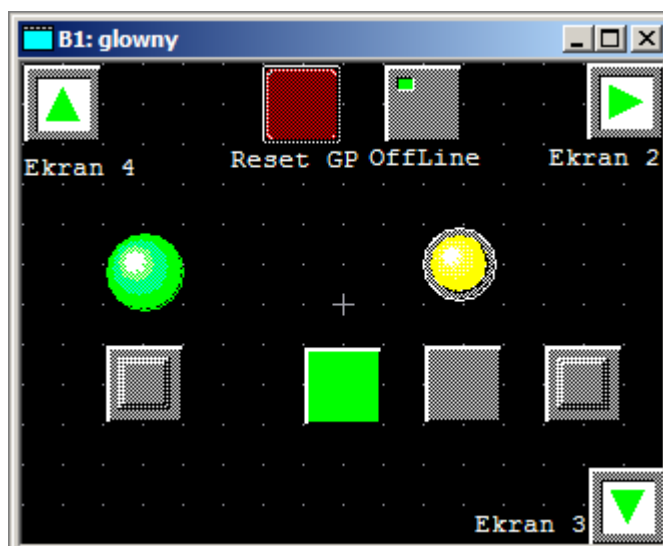


Rysunek 15: Konfiguracja elementu Function SW.


Należy wybrać opcję „Go to Screen” i wprowadzić wartość 2. Oznacza to, że po naciśnięciu tego elementu na wyświetlaczu panelu pojawi się ekran numer dwa. Należy jeszcze dodać do tego ekranu 3 takie elementy z tym, że powinny one prowadzić do ekranów nr. 3, 4.

Element Function SW pozwala na realizacji jeszcze takich funkcji jak: przechodzenie do wcześniejszego ekranu, resetowanie panelu, odłączanie się od sterownika oraz pracę z plikami.

Dodatkowo dodamy dwa takie elementy z funkcją „Reset GP” oraz „Off Line”.




Rysunek 16: Wygląd ekranu głównego po dodaniu wszystkich elementów

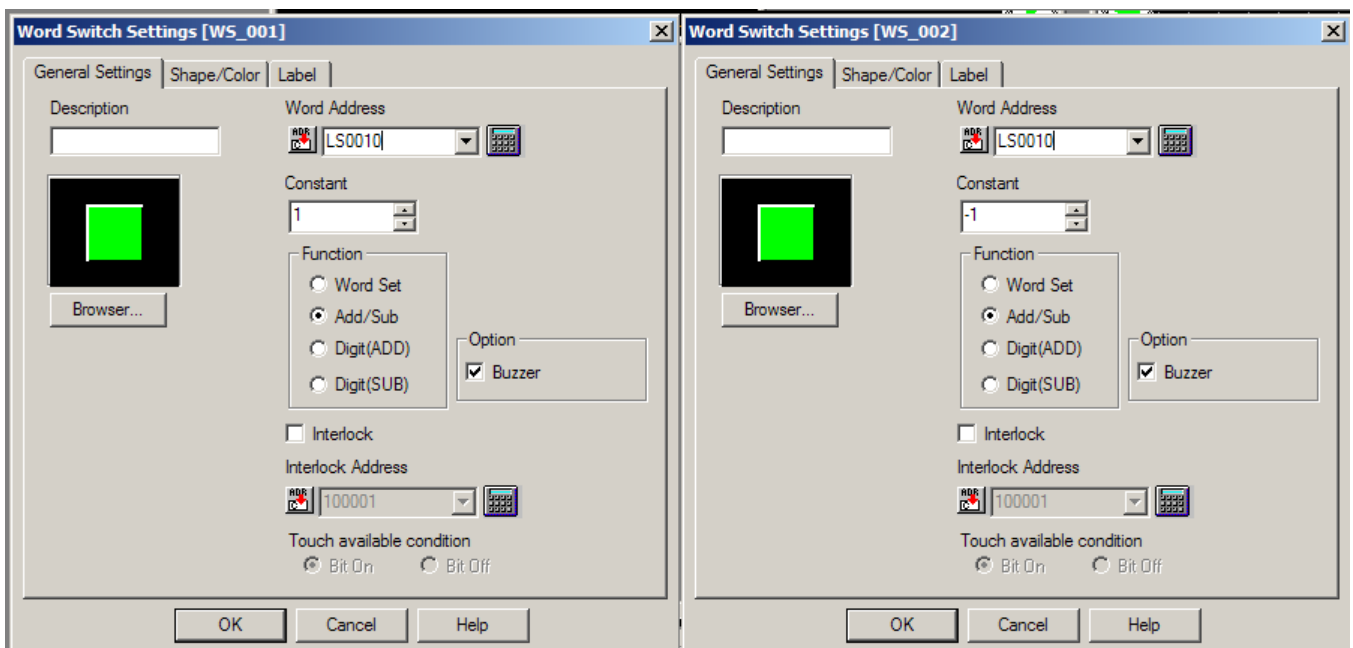
Aby podpisać dany element można posłużyć się narzędziem „Text”. Aby dodać nowy napis należy kliknąć ikonę  na pasku roboczym w głównym oknie. W otwartym oknie wpisujemy tekst i klikamy na ekranie gdzie ma się pojawić.

### c) Dodanie do ekranu elementu „Word SW”


Kolejnym krokiem będzie dodanie elementów typu „Word SW”, „Value Display” oraz „Free Bar Graph”. Na ekranie także należy dodać elementy Function SW aby można przechodzić pomiędzy poszczególnymi ekranami.

Aby dodać element typu Word SW należy wcisnąć ikonę . Po kliknięciu ukazują się okno konfiguracyjne tego elementu. Zadaniem tego elementu jest działanie na adresie słowa pod którym zapisana jest zmienna.

W pierwszym elemencie wybieramy adres LS0010 oraz funkcję Add/Sub o wartość 1. W drugim elemencie ten sam adres i funkcję, natomiast wartość zmiany o -1. Czyli będzie wartość zmiennej zwiększana lub zmniejszana o jedną jednostkę zależnie od kliknięcia klawisza.

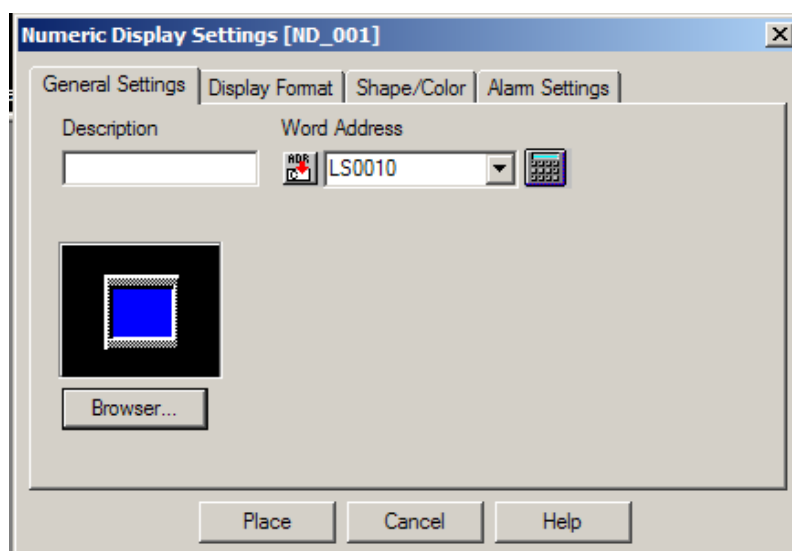


Rysunek 17: Ustawienia elementów typu Word Switch


Aby dodać element „Value Display” należy wybrać . Element służy do wyświetlania wartości zmiennej pod danym adresem. Po wybraniu ikony ukazuje się okno konfiguracyjne.

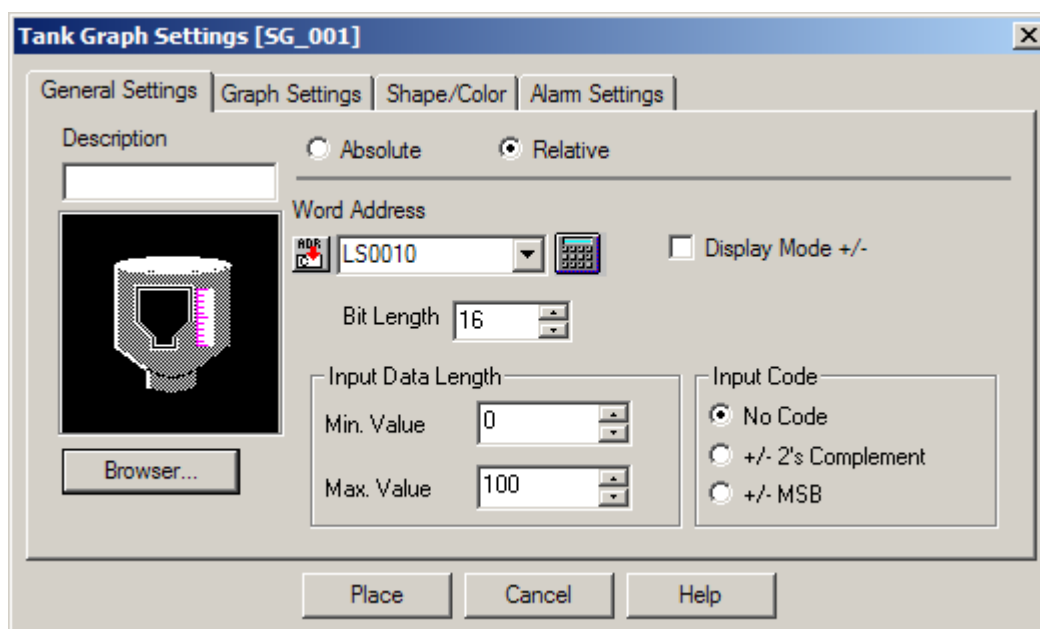
Należy wybrać adres z którego będzie czytana wartość oraz ewentualnie wybrać inny wygląd elementu.





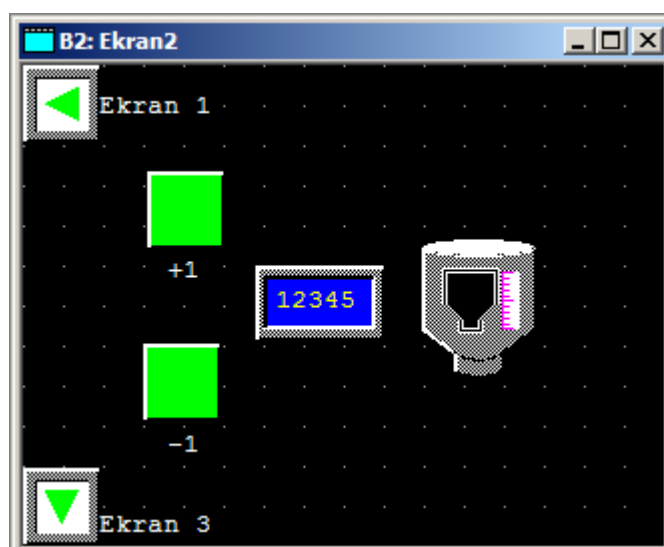
Rysunek 18: Okno konfiguracyjne elementu Value Display

Kolejnym elementem będzie „Free Bar Graph”. Może on służyć do wizualnego zobrazowania stopnia napełnienia jakiegoś zbiornika ale także ustalenie granic w jakich może zmieniać się dana zmienna. Aby wybrać ten element klikamy na ikonę . Pojawia się okno konfiguracyjne. Wybieramy adres zmiennej taki jak dla wcześniejszych elementów na tym ekranie oraz sposób wyświetlania wartości „Relative” co oznacza względny czyli należy jeszcze ustawić zakresy w jakich wartości odczytywanej zmiennej powinny się zmieniać. Dajemy od 0 do 100.



Rysunek 19: Okno konfiguracyjne elementu Free Bar Graph


Do aktualnie tworzone ekranu należy dodać jeszcze 2 przyciski przejścia do ekranu pierwszego oraz ekranu trzeciego. Przydatne będzie także dodać podpisy do tych przycisków aby wiadomo było do którego ekranu przejdzie się po naciśnięciu przycisku.



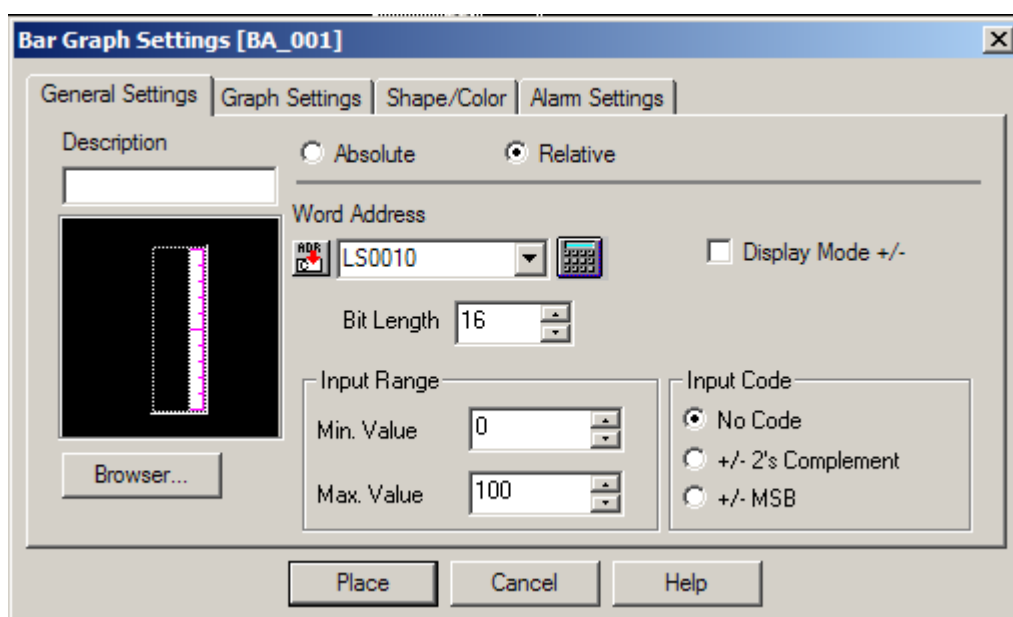
Rysunek 20: skonfigurowany ekran drugi

#### d) Dodanie elementu „Bar Graph”, „Pie Graph”, „Half Pie Graph”, „Meter Graph”


Na początku tworzymy nowy bazowy ekran  i nadajemy mu numer 3.

Następnie dodajemy element typu „Bar Graph” . Po kliknięciu na ikonę otwiera nam się okno konfiguracyjne tego elementu.

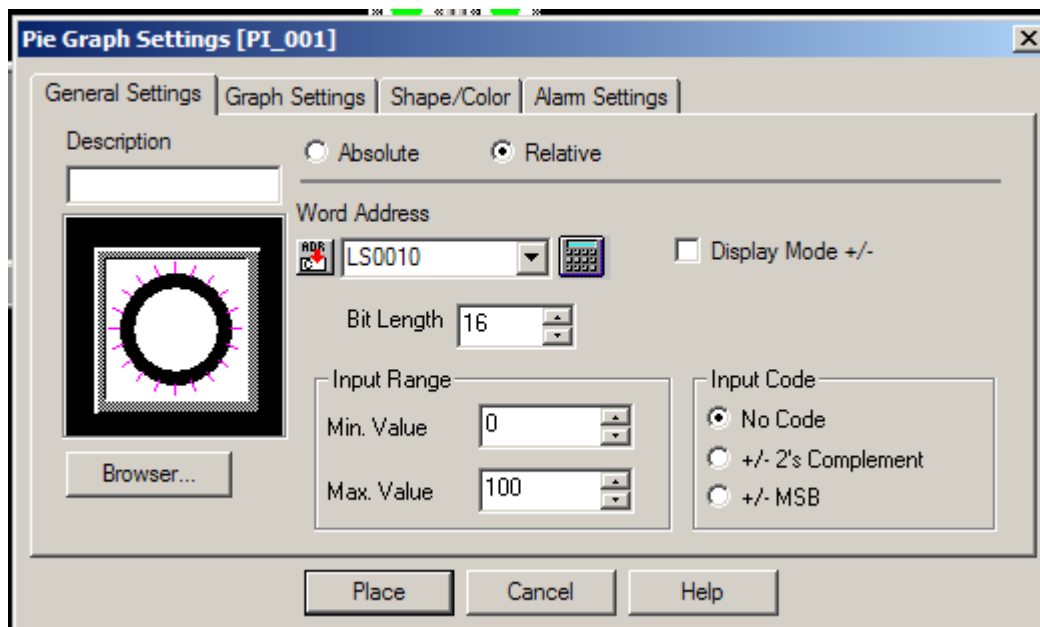
Ustawiamy adres słowa LS0010, który będzie odczytywany a jego wartość rysowana na słupku. Dajemy przedział wartości od 0 do 100 i umieszczamy go na trzecim ekranie.



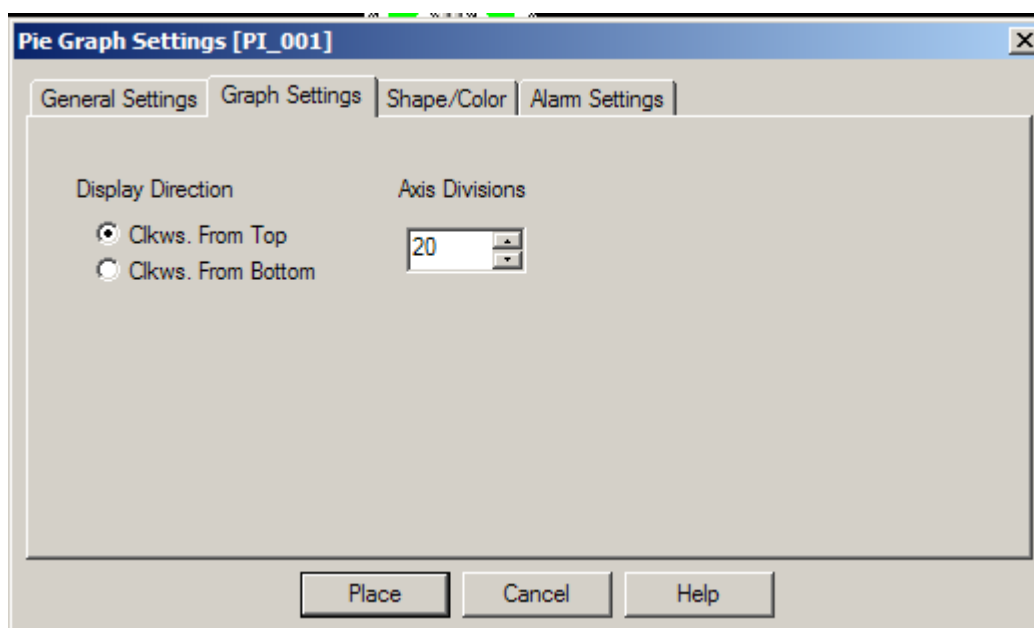
Rysunek 21: Okno konfiguracyjne elementu Bar Graph

Kolejnym elementem dodanym będzie „Pie Graph”. Jest to wykres kołowy, wypełniany zależnie od wartości wskazanej zmiennej w pamięci. Aby dodać ten element klikamy ikonę . Następnie pojawia się okno konfiguracyjne tego elementu.


Ustawiamy adres na LS0010 oraz zakres mierzonej zmiennej od 0 do 100. W drugiej zakładce „Graph Settings” można zmienić parametr „Display Direction” określający pozycję od której będzie początek wypełniania wykresu kołowego oraz „Axis Divisions” który określa ilość kresek dzielących cały wykres kołowy. Po skonfigurowaniu można dodać element do okna trzeciego.

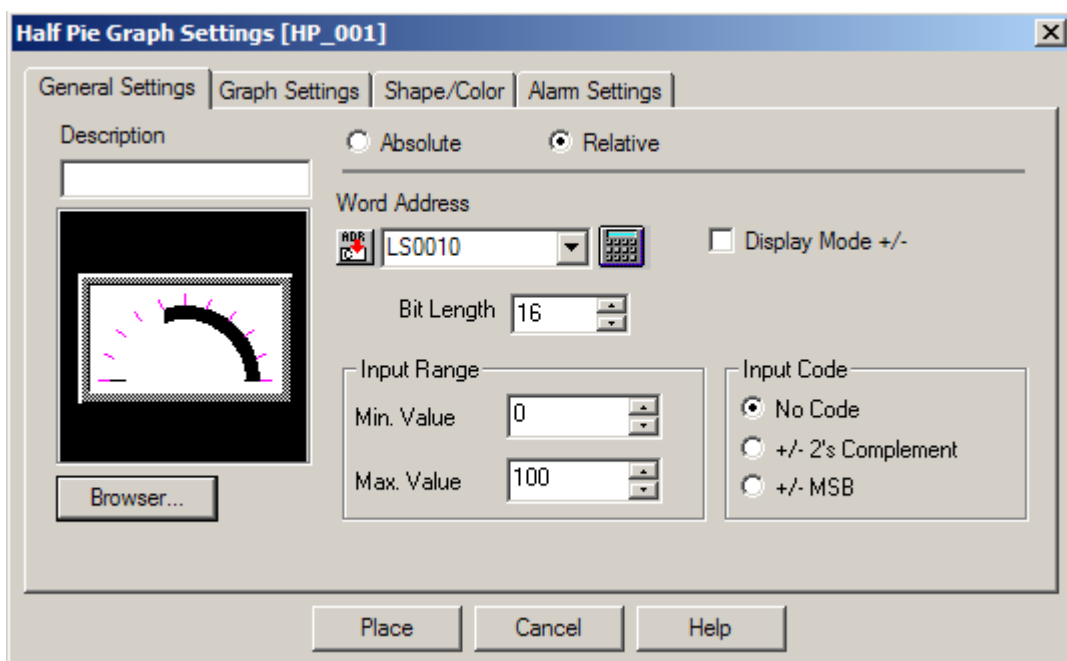


Rysunek 22: Okno konfiguracyjne Pie Graph




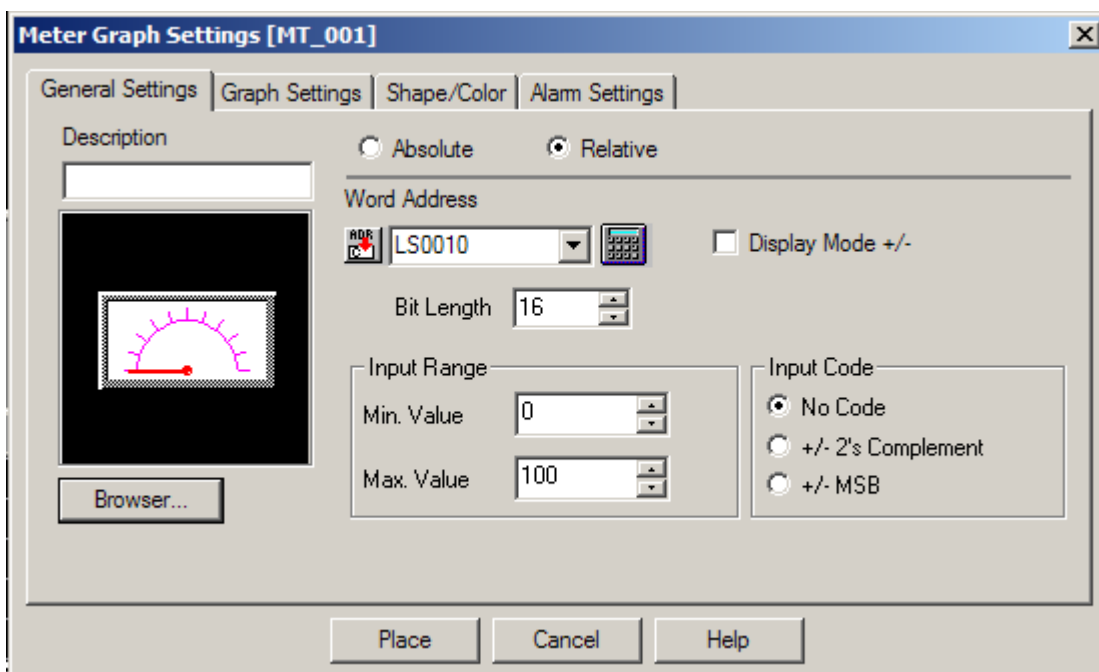
Rysunek 23: Okno konfiguracyjne Pie Graph zakładka druga.

Następnie wybieramy element „Half Pie Graph” jest to wykres kołowy ale połówkowy. Aby go wybrać należy wybrać ikonę . Po kliknięciu pojawia się okno konfiguracyjne. Parametry ustawiamy analogicznie do ustawień Pie Graph i umieszczamy na ekranie nr 3.




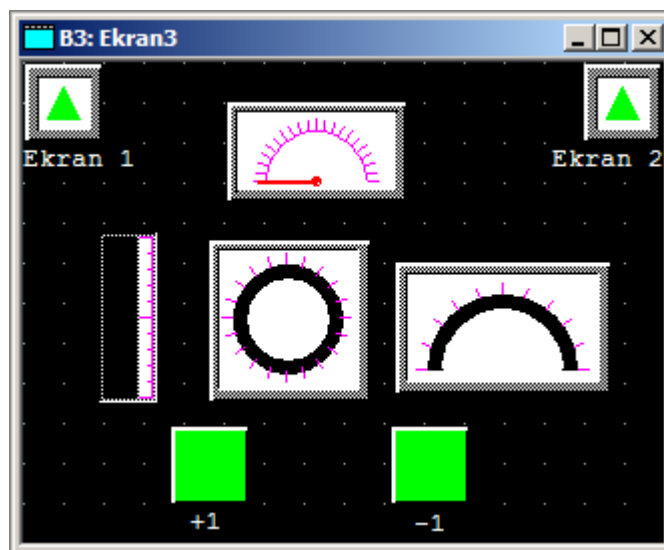
Rysunek 24: Okno konfiguracyjne Half Pie Graph.

Następnie dodajemy element typu „Metere Graph”. Aby dodać ten element należy kliknąć ikonę . Po kliknięciu pojawia się okno konfiguracyjne elementu. Wpisujemy adres słowa LS0010, zakres wartości od 0 do 100. Można dopasować wyświetlany obrazek klikając przycisk „Browser...”. W drugiej zakładce „Graph Settings” można wybrać z której strony będzie wartość zerowa oraz ilość kresk na podziałce.



Rysunek 25: Okno konfiguracyjne elementu Meter Graph.


Do ekranu trzeciego należy dodać jeszcze dwa przyciski typu Word SW aby można zmieniać wartości pod adresem LS0010 oraz dwa przyciski typu Function SW prowadzące do ekranu pierwszego i drugiego. Dodatkowo można podpisać te wszystkie przyciski narzędziem Text .

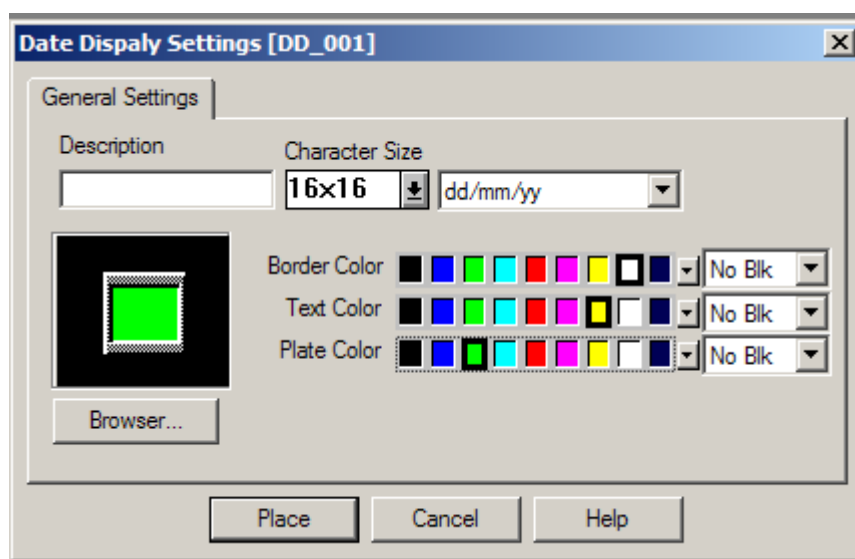


Rysunek 26: W pełni skonfigurowany ekran numer 3.


#### e) Dodanie elementów typu „Date Display” i „Time Display”

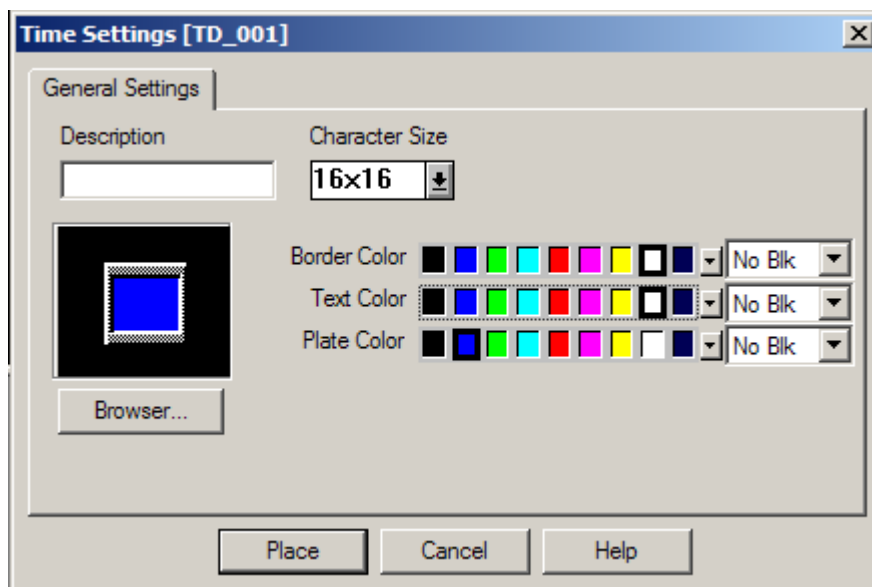
Pomocnymi elementami są Date i Time Display, służące do wyświetlania czasu i daty ze sterownika. Aby dodać te elementy należy dodać nowe okno i nadać mu numer 4.

Na początku klikamy na ikonę  aby dodać wyświetlanie daty. W otwartym oknie można skonfigurować format daty, wielkość znaków oraz wygląd ramki oraz kolor wypełnienia. Po skonfigurowaniu należy dodać element Date Display do okna nr 4.



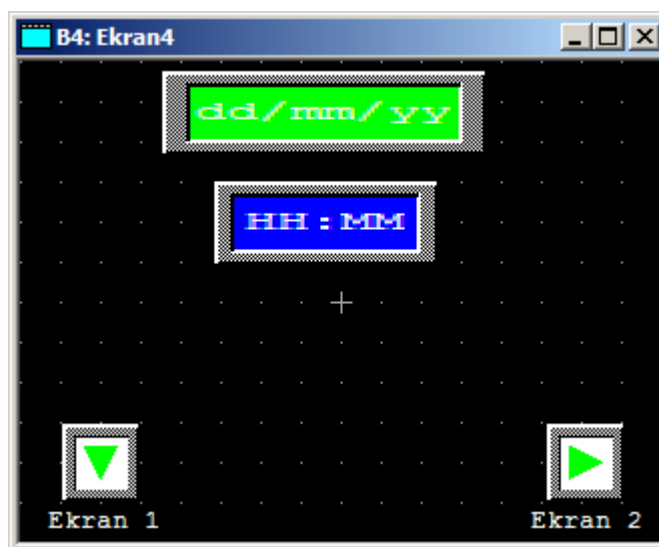
Rysunek 27: Okno konfiguracyjne elementu Data Display

Następnie dodajemy element typu Time Display poprzez kliknięcie ikony . Po kliknięciu ukazują się okno konfiguracyjne. Możemy w nim skonfigurować wielkość znaków wyświetlanej godziny, typ ramki oraz kolory wypełnienia jej oraz kolor znaków.



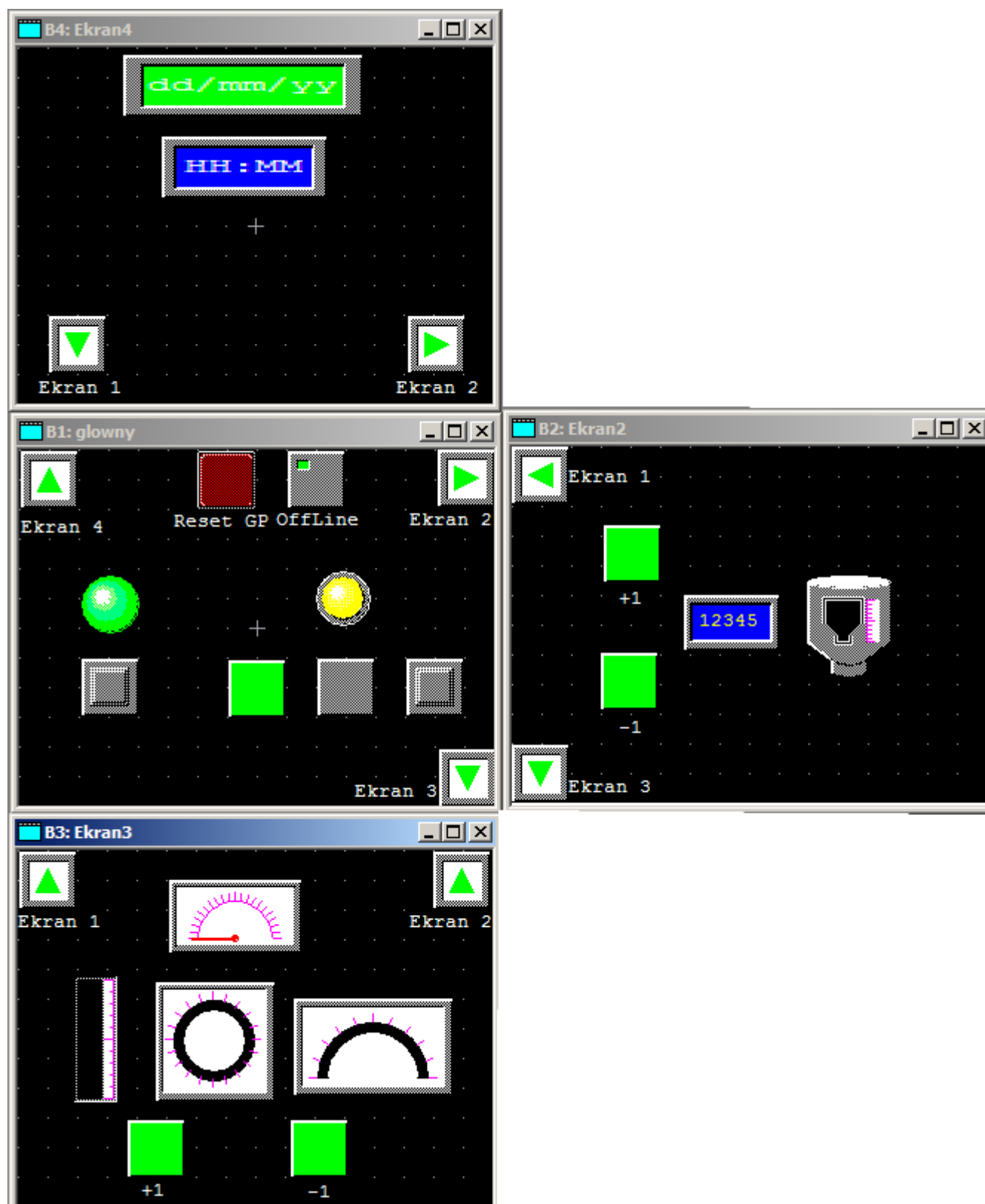
Rysunek 28: Okno konfiguracyjne elementu Time Display

Po dodaniu elementu Time Display należy dodać jeszcze 2 przyciski typu Function SW prowadzące do ekranów nr 1 i 2. Można dodać podpisanie do tych przycisków. Gotowy ekran nr 4 powinien wyglądać tak jak na poniższym rysunku.



Rysunek 29: Gotowy ekran numer 4

Należy zapisać zmiany we wszystkich oknach i można zamknąć okno edytora. Na kolejnej stronie jest rysunek ze wszystkimi skonfigurowanymi oknami.



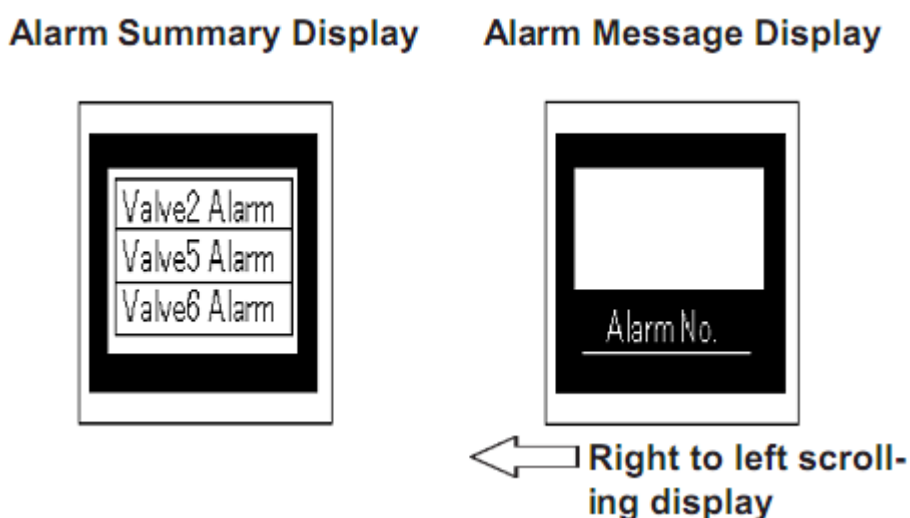
Rysunek 30: Skonfigurowane wszystkie cztery ekrany.

## 4. Konfiguracja alarmów dla zmiennych bitowych oraz rejestrów

Panel umożliwia wprowadzanie oraz edycje alarmów dla zmiennych bitowych jak i rejestrów. Alarmy są wyświetlane gdy zajdzie pewna sytuacja, wartość w monitorowanej zmiennej.

Wyróżnia się dwa sposoby wyświetlania alarmów:

- jako „Alarm Sumary” wyświetlane jako „a-Tag” ( forma skrócona wyświetlana w lewym rogu ekranu) lub „Q-Tag” (forma pełna na środku ekranu)
- jako „Alarm Message” – alarm jest wyświetlany od prawej strony do lewej w przesuwanym pasku.



Rysunek 31: Sposoby wyświetlania alarmów

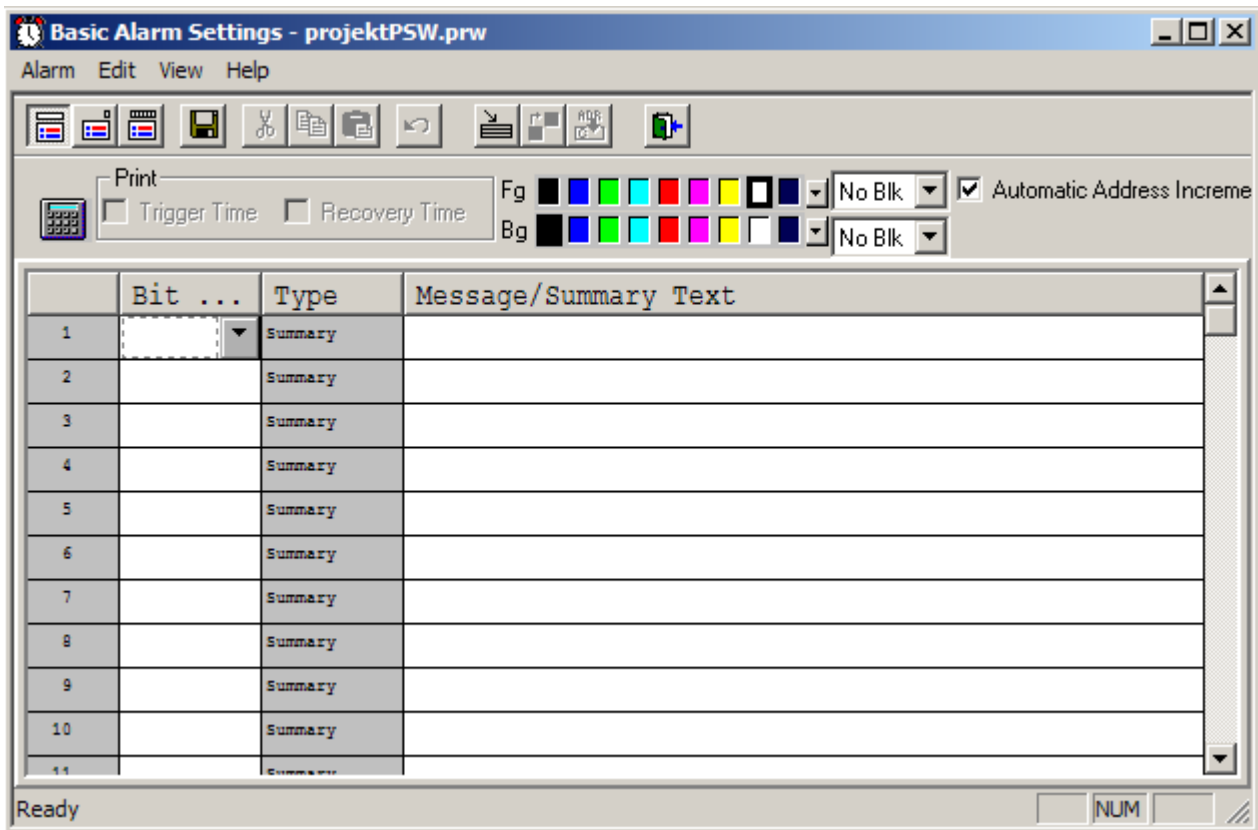
Aby móc tworzyć lub edytować nowe ekrany należy kliknąć w głównym oknie programu GP-PRO/PBIII for Windows w ikonę „ALARM”



Rysunek 32: Ikona wejścia do edycji alarmów



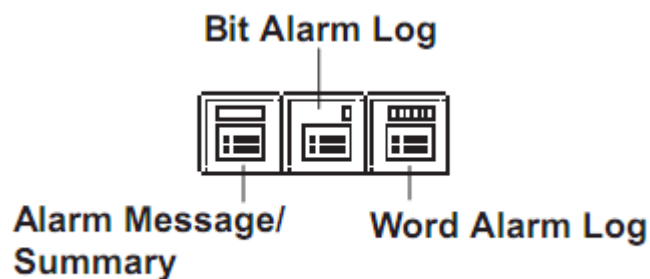
Po kliknięciu ukazują się główne okno edycji alarmów.



Rysunek 33: Okno edycji alarmów

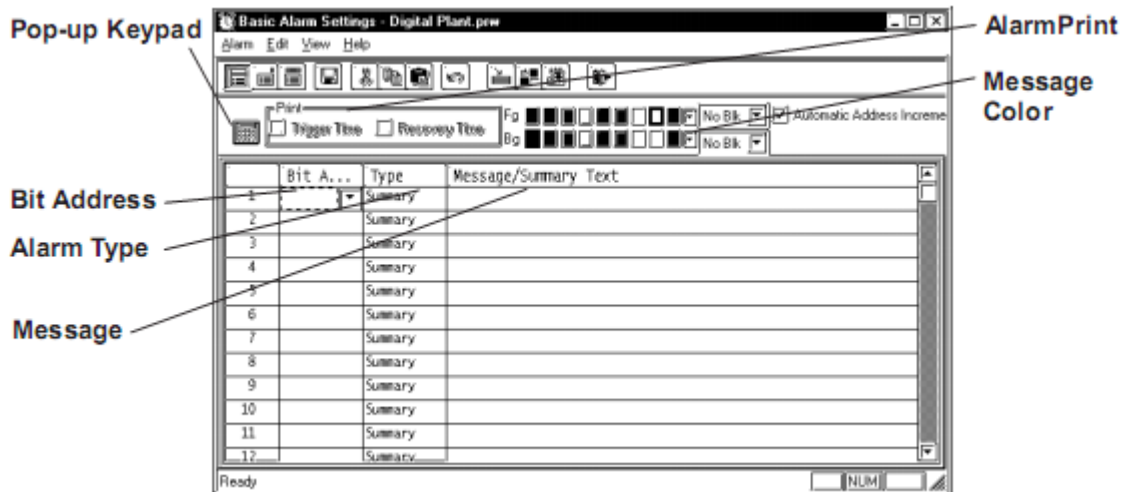
Wyróżnia się trzy typy alarmów:

- Alarm Message/Summary (opis dla alarmów w tabelce)
- Bit Alarm Log (alarm dla bitów)
- Word Alarm Log (alarm dla rejestrów)



Rysunek 34: Wybór typu alarmu na belce okna głównego

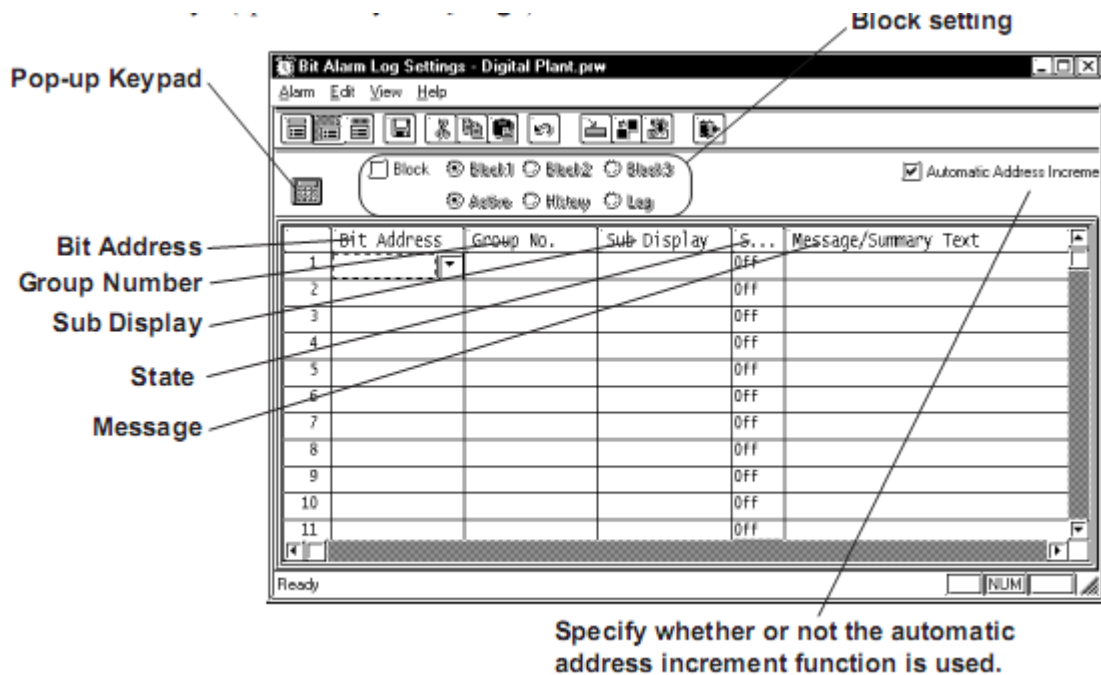
## a) Alarm Message/Summary



Rysunek 35: Opis okna dla Alarm Message/Summary

- Bit Address – miejsce na adres bitu
- Type – sposób wyświetlania alarmu summary – tabelka informacyjna, message – przesuwający się napis
- Message/Summary Text – wypisywana wiadomość
- Alarm Print – wyświetlane dodatkowe informacje
  - \* Trigger Time: czas startu alarmu
  - \* Recovery Time: czas zakończenia alarmu

## b) Bit Alarm Log

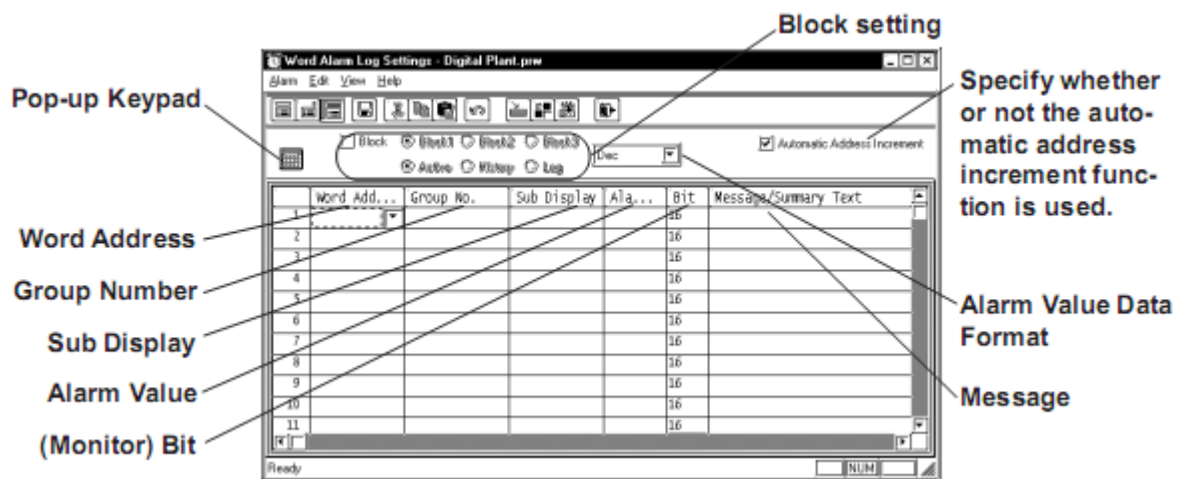


Rysunek 36: Opis okna dla Bit Alarm Log

- Bit Address – miejsce na adres bitu
- Gropu No. – numer grupy dla której będzie zliczany alarm, grupy są dla tej przestrzeni adresów
- Sub Display – określa numer ekran na którym będzie wyświetlana wiadomość alarmu
- State – On/Off określa status bit monitora dla zmiennej, dla jakiego stanu będzie aktywny alarm.

### c) Word Alarm Log

Wyświetla alarm jako Alarm Summary dla rejestrów.



Rysunek 37: Opis okna dla Word Alarm Log

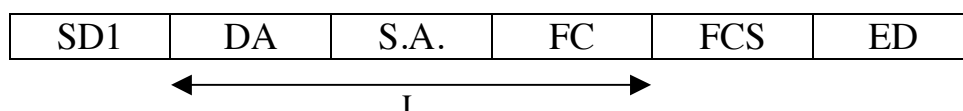
- Word Address – adres 16 bitowego rejestru
- Alarm Value – określenie wartości dla jakiej będzie wywołany alarm

## Dodatek

### Ramki w protokole PROFIBUS.

W PROFIBUS istnieją trzy podstawowe rodzaje ramek:

- Telegramy wywołania i potwierdzenia ze stałą długością pola informacyjnego bez danych:



Gdzie:

SD1 – bajt startu o kodzie 10H

DA – adres docelowy

SA – adres źródła

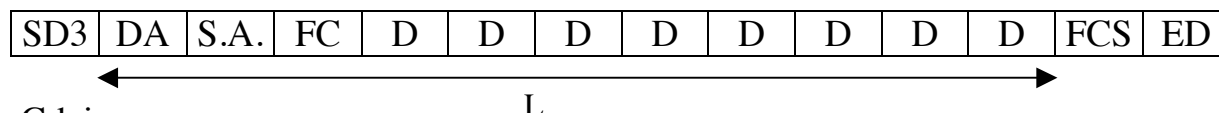
FC – bajt sterujący

FCS – bajt kontrolny

ED – bajt końca o kodzie 16H

L – liczba bajtów w FCS

- Telegramy wywołania i potwierdzenia ze stałą długością pola informacyjnego z danymi:



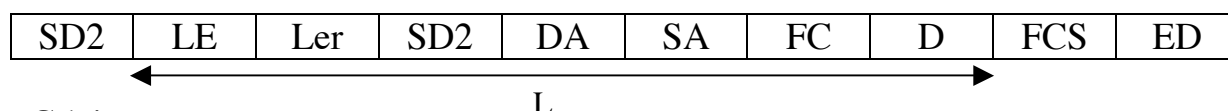
Gdzie:

SD3 – bajt startu o kodzie A2H

D – dane

L – liczba bajtów w FCS = 11

- Telegramy wywołania i potwierdzenia ze stałą długością pola informacyjnego z zmienną długością danych:



Gdzie:

SD2 – bajt startu o kodzie 68H

LE – długość jednostki (liczba wszystkich ramek minus 6)

Ler – długość jednostki, powtórzona

D – dane (1-74)

L – liczba bajtów w FCS = 11

Przykładowe ramki:

Wszystkie wartości w postaci szesnastkowej.

Test komunikacji:

10 66 E6 01 4D 16

Żądanie powtórzenia ostatniego komunikatu:

10 66 E6 12 5E 16

Zatrzymanie DIGITRIC-a:

A2 66 E6 0F 58 00 00 00 00 00 00 00 B3 16

Wznowienie pracy przez Digitrica:

A2 66 E6 0F 58 01 00 00 00 00 00 00 B4 16

### **Suma kontrolna:**

Sumę kontrolną FCS jest bardzo łatwo policzyć. Jest to zwykła suma arytmetyczna z tą różnicą, że do ramki wstawiany jest tylko najmłodszy bajt sumy. Najlepiej liczy się ją dla liczb w postaci szesnastkowej.

Przykład:

$$\begin{array}{r} \text{E4} \\ 43 \\ 12 \\ 3D \\ + 90 \\ \hline 206 \end{array}$$

Zostawiamy najmłodszą część a resztę odrzucamy, czyli w naszym przypadku suma wynosi 06hex.

# 1. Modbus TCP/IP - zasada działania

Modbus TCP/IP (również Modbus TCP) jest po prostu protokołem Modbus RTU z interfejsem TCP, który działa w sieci Ethernetowej.

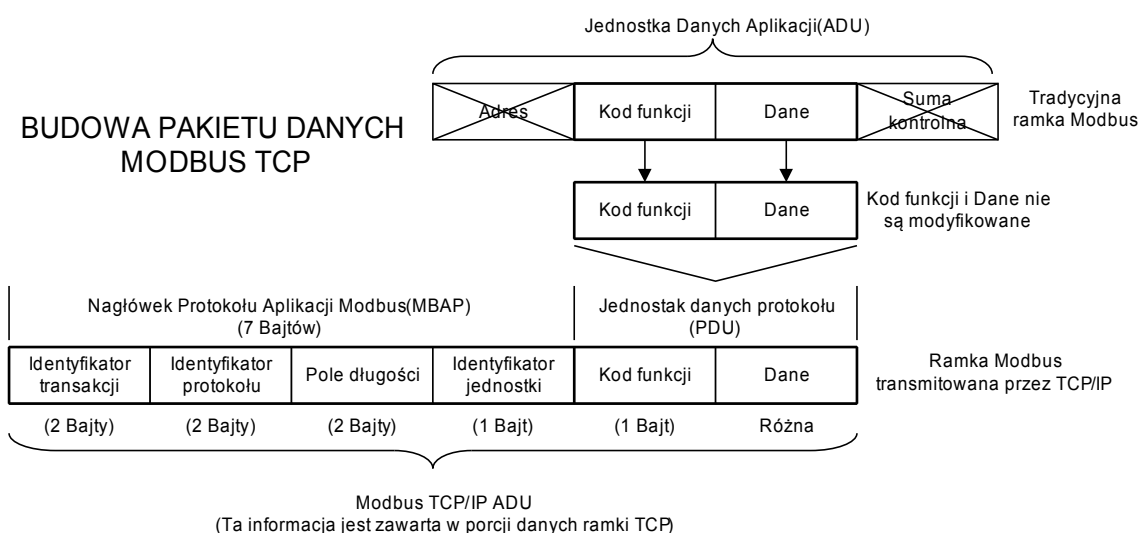
Strukturą wiadomości Modbus jest protokół aplikacji, który definiuje organizację i interpretację danych niezależnych od medium transmisyjnego.

TCP/IP odnosi się do Transmission Control Protocol and Internet Protocol, który zapewnia medium transmisyjne dla transferu danych Modbus TCP/IP.

Prosto podany, TCP/IP pozwala wymieniać bloki danych binarnych pomiędzy komputerami. Jest to poza tym ogólnosiwiatowy standard który służy jako podstawa dla World Wide Web. Główną funkcją TCP jest zapewnienie poprawnego przesłania wszystkich pakietów danych., podczas gdy IP upewnia nas, że wiadomość jest właściwie zaadresowana i zroutowana. Trzeba podkreślić, że kombinacja TCP/IP jest zaledwie protokołem transportowym i nie definiuje znaczenia danych i jak te dane mają być interpretowane (to jest zadanie protokołu aplikacji, w tym wypadku Modbus).

Podsumowując, Modbus TCP/IP używa TCP/IP i sieci Ethernet do transportu danych o strukturze wiadomości Modbus pomiędzy kompatybilnymi urządzeniami. Modbus TCP/IP łączy w sobie sieć fizyczną (Ethernet) ze standardem sieciowym (TCP/IP) i standardową metodą reprezentowania danych (Modbus jako protokół aplikacyjny). Istotnie, wiadomość Modbus TCP/IP jest po prostu komunikatem Modbus enkapsulowanym w opakowanie Ethernet TCP/IP.

W praktyce Modbus TCP osadza standardową ramkę danych Modbus w ramkę TCP bez sumy kontrolnej Modbus, jak pokazano na poniższym diagramie.



Na rysunku widzimy, że kod funkcji i pole danych są wchłaniane w ich oryginalnej formie.

Application Data Unit (ramka ADU) Modbus TCP/IP przyjmuje 7 bajtowy nagłówek (identyfikator transakcji + identyfikator protokołu + długość pola + identyfikator jednostki), a jednostka danych protokołu (kod funkcji + dane). Nagłówek protokołu aplikacji Modbus (MBAP) ma długość 7 bajtów i zawiera poszczególne pola:

- **Identyfikator transakcji/inwokacji (2 Bajty):** to pole identyfikatora jest używane do sparowania transakcji gdy zostało wysłanych kolejno wiele wiadomości w jednym połączeniu TCP przez klienta bez czekania na uprzednią odpowiedź.

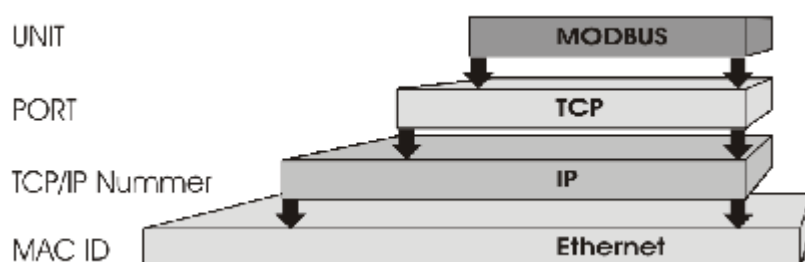
- **Identyfikator protokołu (2 Bajty):** to pole jest zawsze równe 0 dla komunikatów Modbus i innych wartości są zarezerwowane dla przyszłego rozszerzenia.
- **Długość pola (2 Bajty):** to pole zawiera liczebność kolejnych pól włącznie z identyfikatorem jednostki, kodem funkcji i polem danych.
- **Identyfikator jednostki (1 Bajt):** to pole używa się w celu identyfikacji zdalnego serwera sieci nie TCP/IP (dal szeregowego mostkowania). W typowym serwerze aplikacji Modbus TCP/IP, identyfikator jednostki jest ustawiony na 00 lub FF, ignorowane przez serwer i po prostu odpowiadane z powrotem w odpowiedzi.

Kompletny ramka Modbus TCP/IP ADU jest osadzona w pole danych standardowej ramki TCP a wysyłana przez TCP do systemowego portu 502, który jest zarezerwowany dla aplikacji Modbus. Klienci i serwery TCP/IP nasłuchują i odpowiadają dane Modbus przez port 502.

## 2. Modbus TCP/IP – dla sterownika BX9000

Protokół Ethernet jest adresowany na podstawie numeru MAC urządzenia. Użytkownik zazwyczaj nie potrzebuje być niepokojony tym adresem. Adres IP ma długość 4 bajtów i musi być sparametryzowany przez użytkownika na Szynie Łącza i w aplikacji. W Modbus TCP numer portu komunikacji TCP to 502. Jednostka (UNIT) może być dowolnie wybrana pod Modbus TCP i nie musi być konfigurowalna przez użytkownika.

Numer portu TCP dla Modbus TCP został zstandaryzowany jako 502.



### 2.1 Protokół ModbusTCP - ramka komunikatu

Ramka danych protokołu Modbus TCP dla sterownika BX9000 jest zbudowana tak samo jak standardowa ramka przedstawiona w punkcie 1.

Byte	Name	Description
0	Identyfikator Transakcji	Jest zwracane przez urządzenie Slave
1	Identyfikator Transakcji	Jest zwracane przez urządzenie Slave
2	Identyfikator protokołu	Zawsze 0
3	Identyfikator protokołu	Zawsze 0
4	Pole długości	0 (Jeśli wiadomość mniejsza niż 256 Bajtów)
5	Pole długości	Liczba kolejnych bajtów
6	Identyfikator Jednostki	Jest zwracane przez urządzenie Slave

7	Modbus	Protokół Modbus z kodem funkcji i polem danych
---	--------	--

## 2.2 Interfejs Modbus TCP

Address		Description
0x0000		Process data interface
0x00FF		Inputs
0x0800		Process data interface
0x08FF		Outputs
0x1000	Read only	Bus Coupler identification
0x1006		
0x100A		2 byte PLC interface
0x100B		Bus terminal diagnosis
0x100C		Bus Coupler status
0x1010		Process image length in bits, analog outputs (without PLC variables)
0x1011		Process image length in bits, analog inputs (without PLC variables)
0x1012		Process image length in bits, digital outputs
0x1013		Process image length in bits, digital inputs
0x1020		Watchdog, current time in [ms]
0x110A		2 byte PLC interface
0x110B	Read/Write	Bus terminal diagnosis
0x1120		Watchdog, pre-defined time in [ms] (Default value: 1000)
0x1121		Watchdog Reset Register
0x1122		Type of watchdog
		1 Telegram watchdog (default)
		0 Write telegram watchdog
0x1123		ModbusTCP mode**
		1 Fast Modbus
		0 Normal Modbus (default)
0x4000		Flags area (%MB..)*
0x47FF		

## 2.3 Błędne odpowiedzi urządzenia slave w ModbusTCP

Kiedy użytkownik wysyła do urządzenia podrzędnego (slave) żądanie lub informację, której ten nie rozumie, ten odpowiada z raportem błędu. Ta odpowiedź zawiera numer funkcji i kod błędu. 0x80 jest dodawane do zwracanej wartości funkcji.

Kod	Nazwa	Znaczenie
1	NIEPOPRAWNA FUNKCJA	Funkcja Modbus nie zaimplementowana
2	NIEPOPRAWNY ADRES DANYCH	Niepoprawny adres lub długość
3	NIELEGALNA WARTOŚĆ DANYCH	Niepoprawny parametr - Funkcja diagnostyczna - Niepoprawny rejestr
4	BŁĄD URZĄDZENIA	Błąd Watchdog lub K-Bus



	SLAVE	
6	URZĄDZENIE SLAVE ZAJĘTE	Dane wyjściowe zostały już dostarczone z innego IP urządzenia.

## 2.4 Funkcje ModbusTCP obsługiwane przez sterownik BX9000

### Odczyt rejestrów (Funkcja 3)

Odczyt rejestrów binarnych może odczytywać słowa wejściowe i wyjściowe a także rejestry. Wejścia mają offset 0 - 0xFF a wyjścia 0x800 - 0x8FF.

W danym przykładzie czytane są dwa pierwsze wyjścia analogowe. Analogowe wyjścia są na offsecie 0x800. Długość zawiera liczbę kanałów do przeczytania.

#### Zapytanie

Nazwa bajtu	Przykład
Kod funkcji	3
Adres startu (starszy bajt)	8
Adres startu (młodszy bajt)	0
Liczebność (starszy bajt)	0
Liczebność (młodszy bajt)	2

Urządzenie slave odpowiada liczbą bajtów równą 4, są to 4 bajty danych. Żądanie zwrócone było o dwa kanały analogowe, a te zostały podzielone na dwa słowa. In the analog output process image, the first channel has the value 0x3FFF, while the second channel has the value 0x0.

#### Odpowiedź

Nazwa bajtu	Przykład
Kod funkcji	3
Liczba bajtów	4
Dane 1 (starszy bajt)	63
Dane 1 (młodszy bajt)	255
Dane 2 (starszy bajt)	0
Dane 2 (młodszy bajt)	0

### Odczyt rejestrów wejściowych (Funkcja 4)

Funkcja odczytu rejestrów wejściowych czyta wejścia analogowe.

W tym przykładzie pierwsze dwa wejścia analogowe urządzenia slave są czytane. Wejścia analogowe zaczynają się od offsetu 0x0000. Długość zawiera liczbę słów które będą czytane. Moduł KL 3002 ma dwa słowa danych wejściowych, dlatego wartość *Liczebności* (Count low) jest podana jako 2.

## Zapytanie

Nazwa bajtu	Przykład
Kod funkcji	4
Adres startu (starszy bajt)	0
Adres startu (młodszy bajt)	0
Liczebność (starszy bajt)	0
Liczebność (młodszy bajt)	2

Urządzenie slave odpowiada liczbą bajtów równą 4, są to 4 bajty danych. Żądanie zwrócone było o dwa kanały analogowe, a te zostały podzielone na dwa słowa.

## Odpowiedź

Nazwa bajtu	Przykład
Kod funkcji	4
Liczba bajtów	4
Dane 1 (starszy bajt)	0
Dane 1 (młodszy bajt)	56
Dane 2 (starszy bajt)	63
Dane 2 (młodszy bajt)	11

## Zapisz pojedynczy rejestr (Funkcja 6)

Funkcja zapisu pojedynczego rejestru może być użyta do dostępu do wyjściowego process image i interfejsu.

Pierwsze analogowe wyjście jest zapisane za pomocą funkcji 6. Analogowe wyjścia rozpoczynają się od offsetu 0x0800. W tym wypadku offset zawsze opisuje słowo. To oznacza że offset 0x0003 odnosi się do czwartego słowa w wyjściowym process image.

## Zapytanie

Nazwa bajtu	Przykład
Kod funkcji	6
Adres startu (starszy bajt)	8
Adres startu (młodszy bajt)	0
Dane (starszy bajt)	63
Dane (młodszy bajt)	255

Urządzenie slave odpowiada tym samym komunikatem i potwierdza dostarczenie wartości.

## Odpowiedź

Nazwa bajtu	Przykład
Kod funkcji	6
Adres startu (starszy bajt)	8
Adres startu (młodszy bajt)	0
Dane (starszy bajt)	63

Dane (młodszy bajt)	255
---------------------	-----

## Zapisz wiele rejestrów (Funkcja 16)

Funkcja zapisz wiele rejestrów może być użyta do zapisu określonej liczby analogowych wyjść. Pierwsze dwa słowa wyjść analogowych są zapisane w tym przykładzie. Analogowe wyjścia zaczynają się w offsecie 0x0800. W tym wypadku offset zawsze opisuje słowo. To oznacza że offset 0x0003 odnosi się do czwartego słowa w wyjściowym process image. Długość określa liczbę słów, a Liczba bajtów jest określona przez sumę wszystkich bajtów, które będą zapisane.

Przykład: 4 słowa – odpowiada to liczbie bajtów 8

Bajty danych zawierają wartości analogowych wyjść. W danym przykładzie dwa słowa będą zapisane. Pierwsze słowo do nadania to 0x7FFF a drugie 0x3FFF.

### Zapytanie

Nazwa bajtu	Przykład
Kod funkcji	16
Adres startu (starszy bajt)	8
Adres startu (młodszy bajt)	0
Długość (starszy bajt)	0
Długość (młodszy bajt)	2
Liczba bajtów	4
Dane 1 bajt 1	127
Dane 1 bajt 2	255
Dane 2 bajt 1	63
Dane 2 bajt 1	255

### Odpowiedź

Urządzenie odpowiada adresem startu i długością transmitowanych słów

Nazwa bajtu	Przykład
Kod funkcji	16
Adres startu (starszy bajt)	8
Adres startu (młodszy bajt)	0
Długość (starszy bajt)	0
Długość (młodszy bajt)	2

## Odczytaj lub zapisz rejestry (Funkcja 23)

Kilka analogowych wyjść może być zapisanych a także analogowych wyjść odczytanych za pomocą jednego komunikatu za pomocą funkcji odczytaj/zapisz rejestry. W tym przykładzie dwa pierwsze analogowe wyjścia są zapisane i dwa pierwsze analogowe wejścia odczytane. Analogowe wyjścia zaczynają się od offsetu 0x0800 a wejścia od 0x0000. Offset zawsze opisuje słowo. Offset 0x0003 zapisuje do czwartego słowa w wyjściowym process image. Długość określa liczbę słów, a Liczba bajtów jest określona przez sumę wszystkich bajtów, które będą zapisane.

Przykład: 4 słowa – odpowiada to liczbie bajtów 8

Bajty danych zawierają wartości analogowych wyjść. W danym przykładzie dwa słowa będą zapisane. Pierwsze słowo do nadania to 0x3FFF a drugie 0x7FFF.

### Zapytanie

Nazwa bajtu	Przykład
Kod funkcji	23
Adres startu odczytu (starszy bajt)	0
Adres startu odczytu (młodszy bajt)	0
Długość odczytu (starszy bajt)	0
Długość odczytu (młodszy bajt)	2
Adres startu zapisu (starszy bajt)	8
Adres startu zapisu (młodszy bajt)	0
Długość zapisu (starszy bajt)	0
Długość zapisu (młodszy bajt)	2
Liczba bajtów	4
Dane 1 (starszy bajt)	63
Dane 1 (młodszy bajt)	255
Dane 2 (starszy bajt)	127
Dane 2 (młodszy bajt)	255

### Odpowiedź

Urządzenie slave odpowiada adresem startu i liczbą bajtów transmitowanych w *Liczbie bajtów*. Dane informacyjne zawierają w danym przykładzie pierwsze słowo 0x0038 a drugie 0x3F0B.

Nazwa bajtu	Przykład
Kod funkcji	23
Liczba bajtów	4
Dane 1 (starszy bajt)	0
Dane 1 (młodszy bajt)	56
Dane 2 (starszy bajt)	63
Dane 2 (młodszy bajt)	11

# **I. WPROWADZENIE**

## **1. Charakterystyka protokołu CAN**

CAN (Controller Area Network) jest szeregową magistralą, która znalazła szerokie zastosowanie w automatyce przemysłowej. Zaprojektowana została pierwotnie dla przemysłu motoryzacyjnego z myślą o szybkiej i niezawodnej komunikacji wykorzystującej krótkie wiadomości. Umożliwia pracę w czasie rzeczywistym. Wbudowany sposób zabezpieczenia przed błędami gwarantuje, że przez 1000 lat pracy tylko jeden błąd nie będzie wykryty. Ze względu na zastosowane mechanizmy ograniczania i wykrywania błędów sieć CAN jest odporna nawet w krytycznych warunkach zakłóceńowych.

Magistralę CAN określa specyfikacja firmy BOSCH. Wersja 2.0 wyróżnia dwa standardy:

- Standard CAN -standardowy CAN, wersja 2.0A, 11-bitowy identyfikator,
- Extended CAN -rozszerzony CAN, wersja 2.0B, 29-bitowy identyfikator.

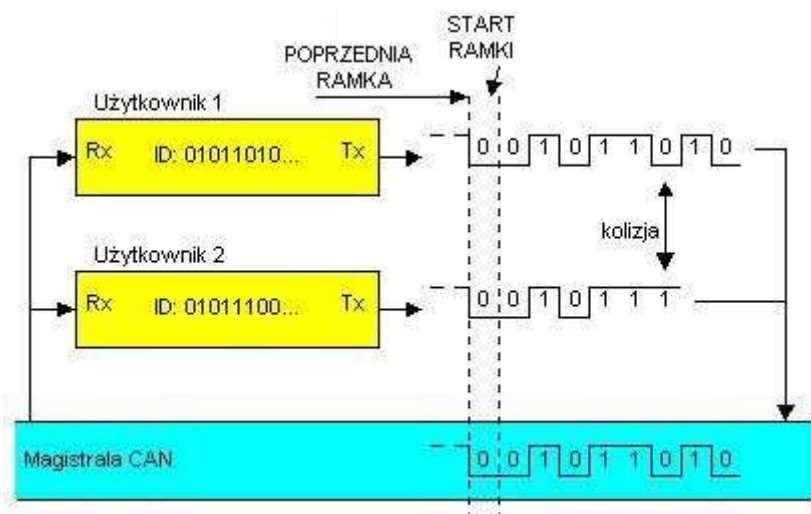
W sieci CAN rozróżniamy dwa poziomy logiczne:

- dominujący (ang. „dominant”) - na magistralę zapisywane jest logiczne 0,
- recesywny (ang. „recessive”) - na magistralę zapisywane jest logiczne 1.

Magistrala CAN jest dwuprzewodową, pół-duplexową siecią wykorzystującą CSMA/CA (dostęp z unikaniem kolizji) jako sposób dostępu do medium transmisyjnego. Zapewnia on jednakowe prawa dla poszczególnych stacji oraz uniezależnia sieć od awarii któregośkolwiek z urządzeń. Wszystkie stacje nasłuchują, czy w sieci nie znajduje się sygnał. Ponadto wszystkie mają jednakowy dostęp do wspólnego medium transmisyjnego.

Przed rozpoczęciem nadawania każda stacja CAN sprawdza stan sieci. Używa również mechanizmu wykrywania kolizji, podobnie jak sieć Ethernet. Należy podkreślić, że w sieci Ethernet, jeśli nastąpi kolizja, obydwa nadające urządzenia zaprzestają transmisji na losowy przedział czasu, by ponownie próbować wysłać dane. W konsekwencji Ethernet jest podatny na przeciążenia magistrali. CAN natomiast stosuje arbitraż, rozwiązując skutecznie problem kolizji.

Jeżeli stacja nie stwierdzi trwania transmisji pochodzącej od innych stacji, wysyła określony sygnał oznaczający chęć nadawania. Po tym fakcie odczekuje przedział czasu, aby sygnał dotarł do wszystkich urządzeń i rozpoczyna transmisję danych. Każdy węzeł, który w trakcie nadawania otrzymał zgłoszenie nadawania od innego węzła, zaprzestaje transmisji. W ten sposób określono zasady zachowania poszczególnych stacji w przypadku kolizji. Ważne jest,

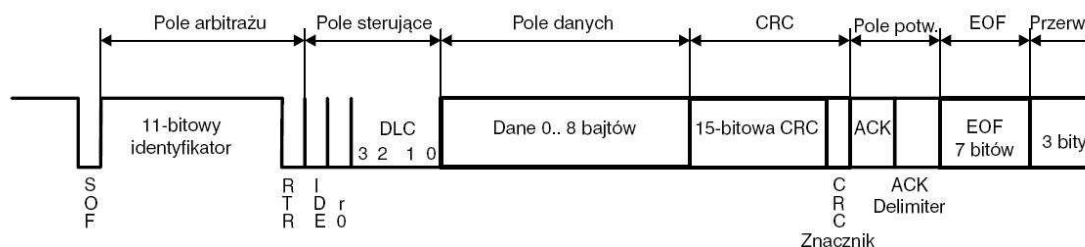


że po fakcie kolizji w sieci CAN, w przeciwieństwie do sieci Ethernet (CSMA/CD), o prawo dostępu do medium transmisyjnego starają się wyłącznie stacje biorące udział w kolizji. Wiadomość CAN jest wysyłana przez węzeł, dopiero wtedy gdy ten stwierdzi, że magistrala jest wolna.

Pierwszym polem komunikatu jest identyfikator (11-bitowy lub 29-bitowy). Jeżeli jednocześnie będą nadawać dwa moduły, to identyfikatory wiadomości "zsumują się". Podczas jednoczesnej próby zapisania na magistralę logicznego 0 (poziom dominujący) i logicznego 1 (poziom ustępujący) przez dwa różne urządzenia, na magistrali pozostanie poziom dominujący. Poziom 0 ma wyższy priorytet niż poziom 1.

Dzięki temu stacja, która nadała 1 (komunikat o wyższym identyfikatorze – niższym priorytecie) wykrywa kolizję i przerywa nadawanie. Natomiast węzeł o niższym identyfikatorze nadaje resztę komunikatu bez przeszkód

Struktura ramki CAN:



- pole startowe SOF (Start of Frame) – bit ten jest zawsze dominujący i oznacza początek ramki wiadomości,

- pole arbitrażu – zawiera dane na podstawie których określany jest dostęp do medium transmisyjnego:
  - \* identyfikator – zawiera logiczny adres i priorytet komunikatu, im niższy identyfikator tym wyższy priorytet,
  - \* bit RTR (Remote Transmission Request) - poziom logiczny 0 oznacza, że wiadomość jest ramką danych (Data Frame),
- pole sterujące – 6 bitów informacji o budowie ramki:
  - \* bity r0 i r1 są zarezerwowane dla przyszłych zastosowań,
  - \* bity DLC (Data Length Code) określają rozmiar pola danych w bajtach w zakresie 0-8,
- pole danych (Data Field) zawierające od zera do ośmiu bajtów transmitowanych danych,
- pole CRC – pole o długości 15 bitów informacji zabezpieczających,
- pole potwierdzenia (ACKnowledge field) – zawiera 2-bitowe potwierdzenie poprawnego odebrania ramek,
- pole zakończenia ramki EOF (End Of Frame ) - zawiera siedem recesywnych bitów.