Przemysłowy Instytut Automatyki i Pomiarów
Al. Jerozolimskie 202, 02-486 Warszawa

440

ZESPÓŁ AUTOMATYKI ELEKTRONICZNEJ
PRACOWNIA TESTERÓW ELEKTRONICZNYCH

BE10

Nazwa ONB/ZNB

Główny wykonawca mgr inż. Tadeusz Goszczyński

Wykonawcy: mgr inż. Jarosław Kowalski

Przenośny zestaw wystawowy urządzeń Lon Works

Etap 2:

Opracowanie założeń, oprogramowanie i uruchomienie zestawu /dokumentacja z etapu 1 oraz instrukcja obsługi i wydruki oprogramowania z komentarzami

(Tytuł pracy, numer i tytuł etapu)

Zleceniodawca Praca statutowa PIAP

Kierownik Pracowni

mgr inż. T.Goszczyński

Z-ca Dyr.d/s
Bad.-Rozwojow.

dr inż. J.Jabłkowski

Kierownik Zespołu

doc.dr inż. J.Korytkowski

Pracę zakończono dnia 28.02.1995 r.

Nr arch. 7196 **Nr zlecenia** S1526

Analiza deskryptorowa

SYSTEM LON WORKS + ZBIERANIE DANYCH + STEROWANIE

Abstrakt

Dokumentacja zawiera instrukcję obsługi stanowiska wystawowego Lon Works i wydruki oprogramowania konfiguracyjnego.

Tytuły poprzednich sprawozdań

Opracowanie materiałów firmy Echelon. Nr rej.7063

Zestaw modelowy urządzeń Lon Works dla systemu zbierania danych o mediach energetycznych. Nr rej.7160.

Zestaw Lon Works do sterowania urządzeniami, współpracujący z systemem zbierania danych o mediach energetycznych. Etap 3.: Oprogramowanie PC do sterowania i zbierania danych o pracy zestawu /dokumentacja w postaci wydruku oprogramowania z komentarzami/. Nr rej. 7195.

Rozdzielnik

Egz. 1. OIN

Egz. 2. ZAE

Egz. 3. ZAE

I. INSTRUKCJA OBSŁUGI STANOWISKA WYSTAWOWEGO

1. Sprawdzić wzrokowo poprawność konstrukcji stanowiska.
2. Włączyć kabel zasilania do gniazda 220V - gniazdo to zgodnie wymaganiami normy powinno mieć bolec fazowy po prawej stronie bolca uziemienia.

Uwaga: lampy powinny zapalić się na chwilę i zgasnąć.

3. Natychmiast po włączeniu kabla wcisnąć przycisk z rysunkiem dzwonka - startujący silnik S1.
4. Od tej chwili można (lecz nie ma konieczności) podłączyć komputer zgodnie z p.II niniejszej instrukcji.
5. Sprawdzić sterowanie przyciskami:

PRZYCISK PANELOWY - BUDYNEK - STEROWANIE SILNIKIEM
klawiszami + i - ZAŁ i WYŁ

- ręczne sterowanie ściemniaczem światła w budynku oraz obrotami dużej tarczy z napisem PIAP

PRZYCISK PANELOWY - FABRYKA - STEROWANIE OŚWIETLENIEM
klawiszami + i - ZAŁ i WYŁ

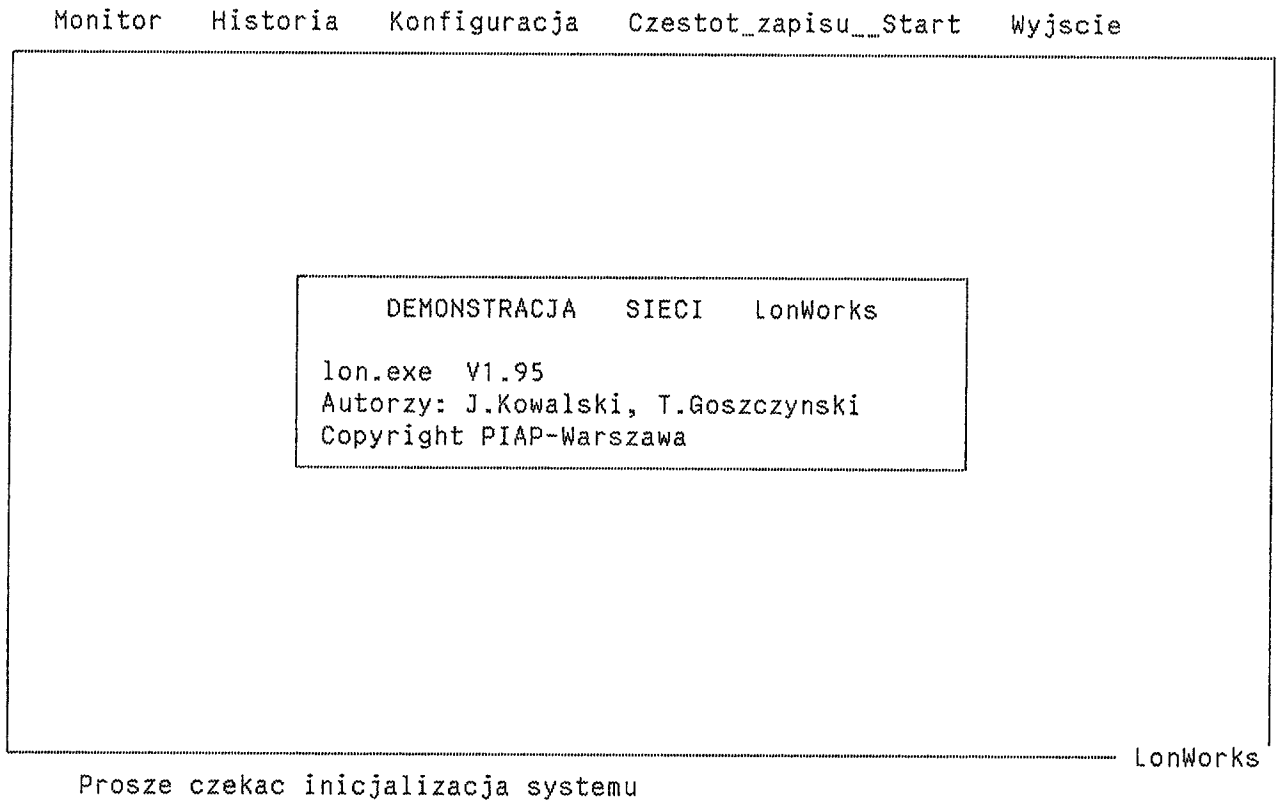
- ręczne sterowanie prędkością obrotową silnika w fabryce

PRZYCISK PANELOWY FABRYKA AH1 - AHLSTROM

- ręczne załączanie i wyłączanie oświetlenia fabryki

II.. INSTRUKCJA OBSŁUGI PROGRAMU KOPUTEROWEGO - OBSŁUGI STANOWISKA WYSTAWOWEGO

1. Sprawdzić czy kabel sieci Lon jest odłączony od Adaptera SLTA.
2. Wywołać program lon znajdujący się w gałęzi c:\WYST
3 Dołączyć kabel sieci Lon do Adaptera SLTA.
4. Ekran komputera po ukazaniu winiety tytułowej przez kilka sekund przyjmuje wygląd przedstawiony na rys.1.
5. Menu przedstawione w górnej linii ekranu ma aktywne jedynie pole MONITOR. (pozostałe opcje są wyłączone dla uproszczenia obsługi - jedynie sygnalizują możliwości osiągalne w typowych programach zbierania danych)
6. Należy wcisnąć klawisz Enter gdy podświetlone jest pole Monitor.
7. Ekran komputera przyjmuje wygląd przedstawiony na rys.2


Monitor Historia Konfiguracja Czestot_zapisu__Start Wyjscie
DEMONSTRACJA SIECI LonWorks
lon.exe V1.95
Autorzy: J.Kowalski, T.Goszczynski
Copyright PIAP-Warszawa
LonWorks
Prosze czekac inicjalizacja systemu

4

Monitor Historia Konfiguracja Czestot_zapisu__Start Wyjscie

ZBIERANIE DANYCH Z SIECI LONWORKS

Data: 01.03.1995 TEMPERATURA: 22.2 °C OBROTY: 2580 obr/min
Godzina: 13:32:44 OSWIETLENIE: 1456 lux NAPIECIE: 8.7 V

Licznik energii cieplnej LC: 14.595 MW 0.950 GJ
Licznik energii elektrycznej LE1: 54.915 kW 3.600 kWh
Licznik energii elektrycznej LE2: 284.211 kW 18.500 kWh

STEROWANIE URZADZENIAMI POPRZEZ SIEC LONWORKS

Nastawa jasnosci swiecenia lamp: 30.0 %
Zalaczanie oswietlenia w budynku : OFF

Nastawa predkosci obrotowej silnika: 80.0 %
Zalaczanie oswietlenia w fabryce : ON

Zalaczanie ogrzewania wody w budynku : ON
Zalaczanie klimatyzacji w budynku : OFF

LonWorks

Esc - powrot do menu

8. W górnej części ekranu pod tytułem ZBIERANIE DANYCH Z SIECI LONWORKS odczytywane są automatycznie wartości danych zbieranych. z urządzeń pomiarowych na tablicy.

Ta część ekranu nie wymaga obsługi użytkownika.

9. W dolnej części ekranu pod tytułem STEROWANIE URZADZENIAMI POPRZEZ SIEC LONWORKS

znajduje się 6 pól, których wartości wyświetlane w okienkach po prawej stronie tekstu są programowane

jednym z trzech sposobów:

poprzez komputer - automatycznie

przez obsługę (lub klienta) z klawiatury komputera przy pomocy klawiszy + - oraz strzałek

przez obsługę (lub klienta) z klawiatury na stoisku przy pomocy klawiszy + - oraz ZAL WYL

III. FUNKCJE URZĄDZEŃ NA TABLICY LONWORKS

1 WYŚWIETLACZ DMN-240 SYSMIK przedstawia na przemian wartości pomiarów:

- temperaturę w hali fabrycznej - z czujnika IMT Sysmik
- obroty silnika w fabryce - z czujnika IMC Sysmik

2 MODUŁ 8 WYJŚĆ CYFROWYCH - 824450 WEIDMULLER

wysterowuje lampki teletechniczne 24V :

- 2 lampki wskazujące jaką wielkość fizyczną wskazuje wyświetlacz DMN-240
- 1 lampkę sygnalizującą pracę systemu klimatyzacji w budynku
- 1 lampkę sygnalizującą pracę podgrzewacza wody w budynku

3 MODUŁ 4 WEJŚĆ CYFROWYCH - 824446 WEIDMULLER

przyjmuje sygnały z :

- sygnał z symulatora licznika ciepła fabryki
- sygnał z symulatora licznika elektrycznego fabryki
- sygnał z symulatora licznika elektrycznego z budynku

4 GNIAZDO DO PODŁĄCZENIA KOMPUTERA

zawiera gniazdo do kabla sieci komputerowej, do którego można podłączyć komputer z programem wyst.exe opisanym w dalszej części instrukcji. Do gniazda należy wetknąć kabel od interfejsu Adapter SLTA/2, który połączony jest ze złączem COM komputera.

5 ROUTER ŁĄCZĄCY SIEĆ TPT78 i Link Power

wykonany w PIAP z elementów dostarczonych przez Echelon (RTR-10 i inne) umożliwia transmisję pomiędzy urządzeniami o różnych mediach transmisji: skrętką - kabel biało-czerwony oraz kablem zasilania 40 V - kabel niebiesko-czerwony

6 MODUŁ WE JŚĆ ANALOGOWY CH 825779 WEIDMULLER

- mierzy napięcie sterujące prędkością obrotów silnika w fabryce

7 MODUŁ IMC SYSMIK

- pomiar prędkości obrotów silnika w fabryce

8 MODUŁ IMT - SYSMIK

- pomiar temperatury w hali fabrycznej

9 MODUŁ AH-ALCM0 - AHLSTROM - wyjście analogowe 1..10V i przekaźnik

- sterowanie silnika tarczy z napisem PIAP
- ściemnianie światła w budynku
- załączanie oświetlenia zewnętrznego budynku

10 MODUŁ ŚCIEMNNIACZA AH- ATD500R AHLSTROM -

sterowany poprzez moduł AH-ALCM0 reguluje jasność lamp 220V

11 MODUŁ ALCM0 - AHLSTROM

- sterowanie silnika w fabryce
- załączanie oświetlenia głównego fabryki

12 PRZYCISK PANELOWY AH4 - AHLSTROM

klawiszami + i - ZAŁ i WYŁ

- ręczne sterowanie ściemniaczem światła w budynku oraz obrotami dużej tarczy z napisem PIAP

13 PRZYCISK PANELOWY AH4 - AHLSTROM

klawiszami + i - ZAŁ i WYŁ

- ręczne sterowanie prędkością obrotową silnika w fabryce

14 PRZYCISK PANELOWY AH1 - AHLSTROM

- ręczne załączanie i wyłączanie oświetlenia fabryki

15 CZUJNIK FOTO (AHLSTROM)

- pomiar natężenia światła (do wyświetlacza)

16 ZASILACZ LINK POWER (AHLSTROM)

- zasilanie modułów Ahlstrom

17 ZASILACZ STABILIZOWANY 24V DC

- zasilanie modułów Weidmuller i Sysmik
- zasilanie symulatorów liczników
- zasilanie lampek teletechnicznych

18 ZASILACZ NIESTABILIZOWANY 12V DC

- zasilanie silników
- zasilanie Routera

19 KOMPUTER + SLTA /2

- monitorowanie i sterowanie całości

```
File: WYST1.XIF generated by APC Revision 1.73, XIF Version 2
Copyright (c) 1990, 1992  by Echelon Corporation
All Rights Reserved.  Run on Wed Feb 10 11:33:45 1993

48:6F:73:74:41:70:70:6C
2 15 1 13 0 3 3 3 3 4 6 11 11 11 11 0 30 8
0 4 4 1 4 13 18 1331 0 15 5 3 196
*

VAR dim_out 0 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"dimmer_out
0 *   1
3 0 0 0 0
VAR dim_in 1 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
"dimmer_in
0 *   1
3 0 0 0 0
VAR maly_out 2 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"maly_out
0 *   1
3 0 0 0 0
VAR maly_in 3 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
"maly_in
0 *   1
3 0 0 0 0
VAR out_Or3 4 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"s1
22 *   1
1 0 0 1 0
VAR out_Or4 5 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"s2
22 *   1
1 0 0 1 0
VAR T2 6 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
"temp
39 *   1
2 0 0 0 0
VAR in_lux 7 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"luksy
8 *   1
2 0 0 0 0
VAR in_RPM_1 8 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"obroty
8 *   1
2 0 0 0 0
VAR in_v_fl 9 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
"napiecie
66 *   5
3 0 1 0 0
3 1 7 0 0
3 0 1 0 0
3 1 7 0 0
2 0 0 0 0
VAR in_PeerLevel3 10 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"cieplo
```

```
22 *   1
1 0 0 1 0
VAR in_PeerLevel1 11 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"energia1
22 *   1
1 0 0 1 0
VAR in_PeerLevel2 12 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"energia2
22 *   1
1 0 0 1 0
```

```
File: IMT_D.XIF generated by APC Revision 2.22, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Wed Jun 15 16:49:03 1994

54:32:53:45:4E:53:5F:35
2 15 0 4 0 3 3 3 3 3 3 11 9 2 4 0 0 2
0 5 4 13 18 1395 0 15 5 3 118
1 2 1 1 4 4 4 15 200 0
78130 0 0 0 0 42 0 0 0 0 0 0
29 0 240 0 10 0 10 50 0 5 7 4 4 14 15
*
"Temperature

VAR T1 0 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
39 *   1
2 0 0 0 0
VAR T2 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
39 *   1
2 0 0 0 0
VAR error 2 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   1
1 0 0 0 0
VAR cycle_ms 3 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 1
*
8 *   1
2 0 0 0 0
```

```
File: IMC.XIF generated by APC Revision 2.22, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Thu Dec 15 11:17:25 1994

49:4D:43:00:00:00:00:00
2 15 0 13 0 3 3 3 3 3 3 11 9 2 4 0 0 8
0 5 4 13 18 1369 0 15 5 3 145
1 2 1 1 4 4 4 15 200 0
78130 0 0 0 0 42 0 0 0 0 0 0 0
29 0 240 0 10 0 10 50 0 5 7 4 4 14 15
*
"IMC - Input Module Counter / Universal Counter

VAR nvoCount_1 0 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvoCount_2 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvoFreqHz_1 2 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
76 *   1
2 0 0 0 0
VAR nvoFreqHz_2 3 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
76 *   1
2 0 0 0 0
VAR nvoRPM_1 4 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvoRPM_2 5 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvoError 6 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   1
1 0 0 0 0
VAR nvicSample_ms_1 7 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvicSample_ms_2 8 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvicSend_ms_1 9 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvicSend_ms_2 10 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
```

```
8 *   1
2 0 0 0 0
VAR nvicPPR_1 11 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
VAR nvicPPR_2 12 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
8 *   1
2 0 0 0 0
```

```
File: ALCM.XIF generated by LONBUILDER Release 2.20, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Thu Dec 22 20:24:31 1994

41:43:54:55:4C:50:30:37
2 15 0 6 0 3 3 3 3 3 3 11 9 4 2 0 0 0
0 4 4 0 0 0 0 0 0 0 0
1 7 1 0 4 4 4 15 200 0
78125 0 0 0 0 0 0 0 0 0 0 0
90 0 240 0 0 0 40 40 0 5 8 5 12 14 15
*
*

VAR nvi00Request 0 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
92 *   2
2 0 0 0 0
1 0 0 1 0
VAR nvo00Status 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
93 *  20
2 0 0 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 7 0 0
3 0 8 0 0
VAR nvi01Value 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
95 *   2
1 0 0 0 0
1 0 0 0 0
VAR nvo01ValueFb 3 0 0 0
0 1 63 1 2 1 0 1 0 1 0 0 0
*
95 *   2
1 0 0 0 0
1 0 0 0 0
VAR nvi_minlevel 4 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
21 *   1
1 0 0 0 0
VAR nvi_maxlevel 5 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
21 *   1
1 0 0 0 0
```

```
File: AHRV2.XIF generated by APC Revision 2.22, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Tue Dec 13 10:25:46 1994

50:4F:54:31:41:00:00:00
2 7 0 4 0 3 2 2 2 2 3 11 9 2 4 0 0 4
8 4 3 13 18 540 0 15 5 3 78
0 0 1 0 4 4 4 15 100 0
312500 0 0 0 0 0 0 0 0 0 0 0
26 0 240 0 10 0 20 40 0 5 7 4 8 14 15
*
"@0,2.

VAR nvi00Request 0 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"@0|1.
92 *   2
2 0 0 0 0
1 0 0 1 0
VAR nvo00Status 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"@0|2.
93 *  20
2 0 0 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 7 0 0
3 0 8 0 0
VAR nvi01ValueFb 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"@1|1.
95 *   2
1 0 0 0 0
1 0 0 0 0
VAR nvo01Value 3 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"@1|2.
95 *   2
1 0 0 0 0
1 0 0 0 0
```

```
File: AHPHOTO.XIF generated by LONBUILDER Release 2.20, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Thu Dec 22 20:23:21 1994

53:45:4E:53:4C:53:50:44
2 8 0 4 0 3 2 2 2 3 2 11 9 4 2 0 0 0
8 4 3 0 0 0 0 0 0 0 0
1 7 1 0 4 4 4 15 200 0
78125 0 0 0 0 0 0 0 0 0 0 0
90 0 240 0 0 0 40 40 0 5 8 5 12 14 15
*
*

VAR nvoLuxvalue 0 0 0 0
0 1 63 1 0 1 0 1 0 1 1 0 0
*
79 *   1
2 0 0 0 0
VAR nvoOccupied 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   1
5 0 0 0 0
VAR nciMinSendT 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
5 0 0 0 0
VAR nciStartupD 3 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
5 0 0 0 0
```

```
File: AH4V2.XIF generated by APC Revision 2.22, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Wed Sep 21 19:57:16 1994

50:42:34:43:41:00:00:00
2 8 0 4 0 3 2 2 2 2 3 11 9 2 4 0 0 4
8 4 3 13 18 548 0 15 5 3 83
0 0 1 0 4 4 4 15 100 0
312500 0 0 0 0 0 0 0 0 0 0 0
26 0 240 0 10 0 20 40 0 5 7 4 8 14 15
*
"@0,2.

VAR nvi00Request 0 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"@0|1.
92 *   2
2 0 0 0 0
1 0 0 1 0
VAR nvo00Status 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"@0|2.
93 *  20
2 0 0 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 7 0 0
3 0 8 0 0
VAR nvi01ValueFb 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"@1|1.
95 *   2
1 0 0 0 0
1 0 0 0 0
VAR nvo01Value 3 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"@1|2.
95 *   2
1 0 0 0 0
1 0 0 0 0
```

```
File: AH1V2.XIF generated by APC Revision 2.22, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Wed Sep 21 19:56:59 1994

50:42:53:57:31:42:00:00
2 15 0 5 0 3 2 2 2 2 3 11 9 2 4 0 0 4
8 4 3 13 18 555 0 15 5 3 121
0 0 1 0 4 4 4 15 100 0
312500 0 0 0 0 0 0 0 0 0 0 0
26 0 240 0 10 0 20 40 0 5 7 4 8 14 15
*
"@0,2.

VAR nvi00Request 0 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"@0|1.
92 *   2
2 0 0 0 0
1 0 0 1 0
VAR nvo00Status 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"@0|2.
93 *  20
2 0 0 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 7 0 0
3 0 8 0 0
VAR nvi01ValueFb 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
"@1|1.
95 *   2
1 0 0 0 0
1 0 0 0 0
VAR nvo01Value 3 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
"@1|2.
95 *   2
1 0 0 0 0
1 0 0 0 0
VAR reverse_ind 4 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 1
"@1|6.
0 *   1
1 0 0 1 0
```

```
File: 824450.XIF generated by APC Revision 2.21, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Mon Sep 26 14:28:02 1994

57:49:38:32:34:34:35:30
2 15 0 59 0 3 3 3 3 3 3 11 11 9 9 0 0 16
0 4 4 13 18 892 0 15 5 3 283
0 2 1 1 4 0 3 15 2000 0
78125 0 0 0 0 42 0 0 0 0 0 0 0
15 0 240 0 0 0 10 50 0 5 0 0 0 0 0
*
"Weidmueller Interface 824450   8 Channel DC out 0.5A not pro
"tected

VAR nvVersion 0 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *  10
1 0 0 1 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0  6
0 0 0 0  8
VAR nvNodeStatus 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *  19
0 0 0 0 0
0 0 0 0  6
0 0 0 0  8
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
VAR nvStatusHb 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   2
2 0 0 0 0
2 0 0 0 0
VAR nvCfg1 3 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp1 4 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
```


18

```
1 0 0 1 0
VAR nvAnd1 5 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr1 6 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp1 7 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint1 8 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat1 9 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvCfg2 10 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp2 11 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd2 12 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr2 13 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp2 14 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint2 15 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat2 16 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
```

```
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvCfg3 17 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp3 18 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd3 19 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr3 20 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp3 21 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint3 22 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat3 23 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvCfg4 24 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp4 25 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd4 26 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr4 27 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp4 28 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
```

```
4 0 3 0 0
VAR nvUniSetpoint4 29 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat4 30 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvCfg5 31 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp5 32 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd5 33 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr5 34 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp5 35 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint5 36 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat5 37 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvCfg6 38 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp6 39 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd6 40 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
```

```
*
22 *   1
1 0 0 1 0
VAR nvOr6 41 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp6 42 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint6 43 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat6 44 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvCfg7 45 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp7 46 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd7 47 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr7 48 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp7 49 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint7 50 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat7 51 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
```

```
VAR nvCfg8 52 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 19 0 0
VAR nvInp8 53 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvAnd8 54 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvOr8 55 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvUniInp8 56 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvUniSetpoint8 57 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
0 *   1
4 0 3 0 0
VAR nvChannelStat8 58 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
```

```
File: 824448.XIF generated by APC Revision 2.21, XIF Version 3
Copyright (c) 1990, 1993  by Echelon Corporation
All Rights Reserved.  Run on Mon Sep 26 14:28:01 1994

57:49:38:32:34:34:34:38
2 15 0 32 0 3 4 3 3 4 3 11 10 9 10 0 0 10
0 4 4 13 18 864 0 15 5 3 202
0 2 1 1 4 0 3 15 2000 0
78125 0 0 0 0 42 0 0 0 0 0 0
15 0 240 0 0 0 10 50 0 5 0 0 0 0 0
*
"Weidmueller Interface 824448   2 Channel Input 48...250 Vuc
"isolated

VAR nvVersion 0 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *  10
1 0 0 1 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0  6
0 0 0 0  8
VAR nvNodeStatus 1 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *  19
0 0 0 0 0
0 0 0 0  6
0 0 0 0  8
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
1 0 0 1 0
0 0 0 0 0
VAR nvStatusHb 2 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   2
2 0 0 0 0
2 0 0 0 0
VAR nvPeerLevel1 3 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
22 *   1
1 0 0 1 0
VAR nvHostLevel1 4 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
22 *   1
```

```
1 0 0 1 0
VAR nvPeerTxRate1 5 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   2
2 0 0 0 0
2 0 0 0 0
VAR nvHostTxRate1 6 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   2
2 0 0 0 0
2 0 0 0 0
VAR nvPreload1 7 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
2 0 0 0 0
VAR nvCounter1 8 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   1
2 0 0 0 0
VAR nvCfg1 9 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 15 0 0
VAR nvChannelStat1 10 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvPeerLevel2 11 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
22 *   1
1 0 0 1 0
VAR nvHostLevel2 12 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
22 *   1
1 0 0 1 0
VAR nvPeerTxRate2 13 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   2
2 0 0 0 0
2 0 0 0 0
VAR nvHostTxRate2 14 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   2
2 0 0 0 0
2 0 0 0 0
VAR nvPreload2 15 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
2 0 0 0 0
VAR nvCounter2 16 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
```

```
*
0 *   1
2 0 0 0 0
VAR nvCfg2 17 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   1
4 0 15 0 0
VAR nvChannelStat2 18 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvGpip1 19 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvGpCfg1 20 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   6
3 0 1 0 0
3 1 1 0 0
3 2 6 0 0
0 0 0 0 0
1 0 0 1 0
2 0 0 0 0
VAR nvGpipStat1 21 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvGpip2 22 0 0 0
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvGpCfg2 23 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   6
3 0 1 0 0
3 1 1 0 0
3 2 6 0 0
0 0 0 0 0
1 0 0 1 0
2 0 0 0 0
VAR nvGpipStat2 24 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvGpip3 25 0 0 0
```

```
0 1 63 0 0 1 0 1 0 1 1 0 0
*
22 *   1
1 0 0 1 0
VAR nvGpCfg3 26 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   6
3 0 1 0 0
3 1 1 0 0
3 2 6 0 0
0 0 0 0 0
1 0 0 1 0
2 0 0 0 0
VAR nvGpipStat3 27 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvGpip4 28 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
22 *   1
1 0 0 1 0
VAR nvGpCfg4 29 0 0 0
0 1 63 0 0 1 0 1 0 1 0 0 0
*
0 *   6
3 0 1 0 0
3 1 1 0 0
3 2 6 0 0
0 0 0 0 0
1 0 0 1 0
2 0 0 0 0
VAR nvGpipStat4 30 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   5
3 0 1 0 0
1 0 0 1 0
0 0 0 0 0
0 0 0 0  4
0 0 0 0  6
VAR nvVersionAction 31 0 0 0
0 1 63 1 0 1 0 1 0 1 0 0 0
*
0 *   6
1 0 0 1 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
```

```
File: XF78LP1.RIF generated by RIFMAKER Revision 1.5, RIF Version 1
Copyright (c) 1992, 1994, Echelon Corporation
All rights reserved.  Run on Mon Jan 30 16:16:21 1995

FLEXIBLE
80:00:01:01:01:01:01:21
80:00:01:01:01:01:04:21
2 0 0 0 0 3 8 3 0 2 3 11 11 9 9
2 0 0 0 0 3 8 3 0 2 3 11 11 9 9
0 5 4 1 3 1254 0
0 5 4 4 3 1254 0
"router TPTt8 / Link Power do modulow Ahlstroma 10 MHz
```

```
/* przerwanie zegarowe 0x1c  - 18 razy na sekunde */
void interrupt  far przerw_time(void)
{
   if(licz++>18*CZAS_MINUTY) {
     licz=0;
     MINuta=1;
   }
}

/*****************************************************/
int minut30(void)
{


      if(MIN30==2) {  /* pierwszy raz */
       p_30 = czas();    MIN30=0;
       for(pom_k=1; pom_k<=IL_KAN; ++pom_k ) {
        enpoll_p30[pom_k] = zapytaj();
        energ_st30[pom_k] = 0;
       }
      } /* koniec ini */

      if(MIN30==0) {
       k_30 = czas();
       if(( r_30=k_30-p_30)<0)
        r_30 += MAX_ZEGAR;

       if((double)r_30 > INTERWAL30) {
        MIN30=1;
        p_30=k_30;
        ile_jeszcze = IL_KAN;  /* bo bedzie liczyl 1moc/1pomiarek() */
        for(pom_k=1; pom_k<=IL_KAN; ++pom_k )
        NIEliczony[pom_k]=1;
        NIEliczony[0]=0;  /* zawsze wypada */
       }
      }
   if(MIN30) {
    if(NIEliczony[pom_kanal]) { /* najpierw zglaszajacy imp kanal */
      NIEliczony[pom_kanal]=0;

      enpoll_k30[pom_kanal] = zapytaj(); /* zczytana z wezla po 30 min */
                            /* przyrost 30 min w wezle zapisany */
      enpoll30[pom_kanal]  = enpoll_k30[pom_kanal] - enpoll_p30[pom_kanal];
      enpoll_p30[pom_kanal] = enpoll_k30[pom_kanal]; /* do nastepnych 30 min */
      /* dodac przed pom_energ ++energ_imp[pom_kanal]  */
      /* obliczmy przyrost zliczania biezacego impulsow w 30 min */
      enzlicz30[pom_kanal] = energ_imp[pom_kanal] - energ_st30[pom_kanal];
      energ_st30[pom_kanal] = energ_imp[pom_kanal]; /* zapamietaj do nast 30 min */
       /* zaliczymy do sumy wiekszy z tych przyrostow ( gdzies brak zasilania) */
      if(enzlicz30[pom_kanal] > enpoll30[pom_kanal])
       enpoll30[pom_kanal] = enzlicz30[pom_kanal];      /* wiekszy przyrost */

      en_co30[pom_kanal] += enpoll30[pom_kanal];  /* zapamietaj na 30 min */
      energ_imp[pom_kanal] = en_co30[pom_kanal];   /* uaktualnij do wyswietlania bie
go */

      moc_30[pom_kanal]=  3600.0 * enpoll30[pom_kanal] * kanpar[pom_kanal].kan_wzmoc
                    kanpar[pom_kanal].kan_alarm_przyr/(INTERWAL30/18.0);
//printf("kanal= %d  enpoll=%d  enzlicz=%d ",pom_kanal, enpoll30[pom_kanal], enzlicz30[p
om_kanal]);
      pom_energ[pom_kanal]= energ_imp[pom_kanal] * kanpar[pom_kanal].kan_wzmoc ;
    } /* if NIEliczony */
    else {
     for(pom_ka=1; pom_ka<=IL_KAN; ++pom_ka ) {
```

```
      if(NIEliczony[pom_ka]) { /* pozostale ile_jeszcze po jednym na pomiarek() */
      NIEliczony[pom_ka]=0;
      enpoll_k30[pom_ka] = zapytaj(); /* zczytana z wezla po 30 min */
                           /* przyrost 30 min w wezle zapisany */
      enpoll30[pom_ka]  = enpoll_k30[pom_ka] - enpoll_p30[pom_ka];
      enpoll_p30[pom_ka] = enpoll_k30[pom_ka]; /* do nastepnych 30 min */
      /* dodac przed pom_energ ++energ_imp[pom_ka]  */
      /* obliczmy przyrost zliczania biezacego impulsow w 30 min */
      enzlicz30[pom_ka] = energ_imp[pom_ka] - energ_st30[pom_ka];
      energ_st30[pom_ka] = energ_imp[pom_ka]; /* zapamietaj do nast 30 min */
       /* zaliczymy do sumy wiekszy z tych przyrostow ( gdzies brak zasilania) */
      if(enzlicz30[pom_ka] > enpoll30[pom_ka])
       enpoll30[pom_ka] = enzlicz30[pom_ka];

      en_co30[pom_ka] += enpoll30[pom_ka];  /* zapamietaj na 30 min */
      energ_imp[pom_ka] = en_co30[pom_ka];   /* uaktualnij do wyswietlania biezacego */
/*moc_30[pom_ka] */
      pom_wynik[pom_ka]=  3600.0 * energ_imp[pom_ka] * kanpar[pom_ka].kan_wzmoc *
                    kanpar[pom_ka].kan_alarm_przyr/INTERWAL30;

      pom_energ[pom_ka]= energ_imp[pom_ka] * kanpar[pom_ka].kan_wzmoc ;
//printf("wpisalem kanal         %d ",pom_ka);
      break;  /* tylko jeden przebieg petli */
      } /* if NIEliczony */
      if(--ile_jeszcze==0)  {
//printf("           MIN30 = 0");
       MIN30=0;
      }
     } // for pom_ka
    } // else
   } // ifMIN30

  return(0);
}

/*****************************************************/



unsigned int  zapytaj(void)
{
symul_poll += 1;
return(symul_poll);
}

/*****************************************************/
 void prin_temp( byte * data ) {   // print a value of type SNVT_xxx_f
    union {
        unsigned int  temp;
        byte    dane[ 2 ];
    } swapper;
    int   i;
    for( i = 1; i >= 0; i-- ) {
        swapper.dane[ i ] = * data++;
//printf("     bajt = %x ",swapper.dane[ i ]);
    }

   pom_energ[pom_kanal]=(double)(swapper.temp-2740)/10.0;
//      printf("     temp = %u ", swapper.temp);
//      printf("  t1= %.1lf", (double)(swapper.temp-2740)/10.0);
}
```

```
 /*************************************************************/

int pisz(                   MsgData     * in_data,
                            ServiceType   service,
                            int           in_length,
                            boolean       in_auth ) {
unsigned int a;
//memcpy(&pom_energ[pom_kanal],&in_data.unv.data,2);
//a=in_data->unv.data;

//  printf("in_data = %u ", a);
prin_temp(in_data->unv.data);
return(0);
}
/************************************************/

/* wywolywany kiedy sie da - sprawdza czy trzeba zapisac dane na dysk
   sprawdza MINuta ustawiane w przerwaniu zegarowym
   wywoluje czytaj_dane() -do wyswietlenia lub alarmow
   sprawdza czy roznica_zeg > INTERWAL_ZAPISU wtedy zapis */


int pomiarek(pierwszy)
int pierwszy;
{


    NI_Code          ni_error;
   ServiceType      service;        /* ACKD, UNACKD_RPT, UNACKD, REQUEST */
   RcvAddrDtl       in_addr;        /* address of incoming msg */
   boolean          in_auth;        /* if incoming was authenticated */
   MsgData          in_data;        /* data of incoming msg */
  int              in_length;      /* length of incoming msg */
  //boolean          prompt_flag;


unsigned int a;

             MINuta=0;

        ni_error = ni_receive_msg(     // Check for network activity
            & service,        /* handle incoming message to this node */
            & in_addr,
            & in_data,
            & in_length,
            & in_auth );
//    printf("jest nv error= %d ",ni_error);
          if( ni_error == NI_TIMEOUT)  {
//    printf("timeout ");
          return (0);
          }
          if( ni_error == NI_OK ){
          process_msg( service, &in_addr, &in_data, in_length, in_auth);
      /* handle_netvar_msg przydziela pom_kanal=nv_index */
 /* printf("pom_kanal= %d ", pom_kanal);*/
  if(pom_kanal<=5) {
    /* dla pozostalych funkcje print w applmsg.c przypisuja wartosci */
          if(++pom_licznik[pom_kanal]%2==0)
           return(0); // drugie zbocze impulsu
```

```
      ++energ_imp[pom_kanal];
      pom_energ[pom_kanal]+=  kanpar[pom_kanal].kan_wzmoc;

      zegar_new[pom_kanal] = (double)czas();
      if(zegar_old[pom_kanal]==0) {
       zegar_old[pom_kanal] = zegar_new[pom_kanal];
       return(0);
      }
      roznica_zeg=zegar_new[pom_kanal]-zegar_old[pom_kanal];
      zegar_old[pom_kanal] = zegar_new[pom_kanal];

      if(roznica_zeg==0)
       return(0);
       if(roznica_zeg<0)
        roznica_zeg+=MAX_ZEGAR;

  pom_wynik[pom_kanal] = 3600.0 * kanpar[pom_kanal].kan_wzmoc *
                      kanpar[pom_kanal].kan_alarm_przyr/(roznica_zeg/18.0);

      if( (pom_wynik[pom_kanal] < kanpar[pom_kanal].kan_alarm_high)
          && ( ALZap[pom_kanal]==1) ){  /* zanik alarmu */
       ALArm[pom_kanal]=0;
       ALZap[pom_kanal]=0;
       }
      if( (pom_wynik[pom_kanal] > kanpar[pom_kanal].kan_alarm_high)
          && ( ALZap[pom_kanal]==0) ){
       ALArm[pom_kanal]=1;
       ALZap[pom_kanal]=1;
       ZAPis=1;
      }
      if( (pom_wynik[pom_kanal] > kanpar[pom_kanal].kan_alarm_low )
          && ( ALZap[pom_kanal]==2) ){  /* zanik alarmu */
       ALArm[pom_kanal]=0;
       ALZap[pom_kanal]=0;
      }
      if( (pom_wynik[pom_kanal] < kanpar[pom_kanal].kan_alarm_low )
          && ( ALZap[pom_kanal]==0) ){
       ALArm[pom_kanal]=2;
       ALZap[pom_kanal]=2;
       ZAPis=1;
      }
    minut30();

    } // if  pom_kanal <=4
     } // if NI ERROR

    sterowanie();

     if(ZAPis==0) {  /* jesli nie bylo polecenia zapisu */
       zegar_odczytany=czas();
       roznica_zeg=(double)(zegar_odczytany-zegar_zapisu);
       if(roznica_zeg<0.0)
        roznica_zeg+=MAX_ZEGAR;
      if((unsigned int)roznica_zeg<interwal)
      {
       MINuta=0;

       return(0);
      }
     }      // if zapis
     /* byl ZAPis lub minal INTERWAL_ZAPIS        U */
  //printf("              ZAPis= %d", ZAPis);
```

```
//      printf("interwal= %d     roznica= %.2lf ",interwal, roznica_zeg);
   zegar_zapisu=czas();
   history_day();
//   lotus_day();
   ZAPis=0;  /* zapisano wynik na dysk*/
   MINuta=0;

 return(a);
}
/*******************************/

/*****************************************************************/
int ini_pomiary()
{
 int i;
 unsigned char j;


        message("CZEKAJ_INICJALIZACJA");
        clrscr();
        czytaj_config();
        interwal = 18 * okres_zapisu;
        /* inicjalizacja calego systemu */
/*      if(ini())  { */    /* blad w ini */
/*QA*/   /* koncz_pomiary();*/
/*       return(1);
        }               */
         /* inicjalizacja tabeli pomiary */

         for(i=1;i<=IL_KAN;++i)  {
          pom_licznik[i]=0;
          pom_energ[i] = 0;
          ALArm[i]=0;
          ALZap[i]=0;
         }
        /* czekaj chwile */
/*QA     j=100;
        i=5;
        while(i>0){
         gettime(&godz_zapis);
         if (godz_zapis.ti_sec != j){
          j=godz_zapis.ti_sec;
          cprintf("\rCZEKAJ %d sek.",--i);
         }
        }
*/
        ilosc_odczytow=0;
        rs_bufor_wskaz=0;
        il_error_czytania=0;
        il_error_ciszy=0;
  /*    zapytaj_o_dane(); */
        getdate(&data_zapis);
        data_zapis.da_year-=1900;

/*QA     ZAPis=1; */    /* pierwszy pomiar bedzie zapisany */
        MINuta=0; /* pomiarek() czyta dane po MINuta */
return(0);
}

 /*---------------------------------------*/
int koncz_pomiary(void)
{
```

```
 disable();
/* setvect(IRQ,oldfunc); */ /* wpisz adres starego progr pzerwania */
 setvect(IRQ_TIME,oldfunc_time);
 enable();
 return(0);
}
/*-------------------------*/
int ini_zegar(void)
{
 disable();
 oldfunc_time=getvect(IRQ_TIME);
 setvect(IRQ_TIME,przerw_time);
  enable();
     interwal = 18 * okres_zapisu;

 return(0);
}
/*-----------------------------------------------------------*/

/*-----------------------------------------------------*/


/*-------------------------------------*/
/* korzysta z daty ze structury data_zapis wywolywanej w pomiarek()
   zapis danych na dysk */

int history_day()
{
int x;
/*char nazwa_zbioru[13];*/
FILE * fpp;

  /* szuka czy jest zbior jesli nie ma to otwiera nowy, jesli jest to
     dopisuje do starego */

   gettime(&godz_zapis);
   getdate(&data_zapis);
   data_zapis.da_year-=1900;
     sprintf(nazwa_zbioru,"d%02d%02d%02d.lst",
     data_zapis.da_year,data_zapis.da_mon,data_zapis.da_day);

     if((fpp=fopen(nazwa_zbioru,"a"))==NULL)
      return(1);

     zegarek();
     for(pom_kanal=1;pom_kanal<=IL_KAN;++pom_kanal)  {
      fprintf(fpp,"\n%-3d    ",pom_kanal);
      fprintf(fpp,"%02d-%02d-%02d   ",(int)data_zapis.da_day,(int)data_zapis.da_mon,
_zapis.da_year);
      fprintf(fpp,"%s    ",tekst_zegarka);
      fprintf(fpp,"%d                  ",pom_kanal);

      fprintf(fpp,"%.2lf",pom_wynik[pom_kanal] );
     }
 fclose(fpp);
 return(0);
}
/*-----------------------------------------------------------*/

int lotus_day()
{
int x;
/*char nazwa_zbioru[13];*/
```

```
FILE * fpp;

  /* szuka czy jest zbior jesli nie ma to otwiera nowy, jesli jest to
     dopisuje do starego */

   gettime(&godz_zapis);
   getdate(&data_zapis);
   data_zapis.da_year-=1900;
     sprintf(nazwa_zbioru,"t%02d%02d%02d.txt",
     data_zapis.da_year,data_zapis.da_mon,data_zapis.da_day);

     if((fpp=fopen(nazwa_zbioru,"a"))==NULL)
      return(1);
     for(pom_kanal=1;pom_kanal<=IL_KAN;++pom_kanal)  {
      fprintf(fpp,"\n%d ",pom_kanal);
      fprintf(fpp,"%.2lf",pom_wynik[pom_kanal] );
     }
 fclose(fpp);
 return(0);
}
/*******************************************************/

/*-------------------------------------------------------------*/
/* pisze text w oknie high */
/* moze sie pokazac wszedzie - znika po wcisnieciu klawisza */
  int error_lotny(text)
  char* text;
  {
   int spalte;
   spalte = ((80-strlen(text))/2)+1; /* centrowanie okna */
   open_window(10,spalte,3,strlen(text)+1,ERROR_FG, ERROR_BG);
   wprint_at(1,2,text);  /* wskazanie na tekst */
   wprint_at(3,2,"    Nacisnij ...");
   spalte=get_znak();        /* i czekaj */
   close_window();
   return(spalte);
  }
/*-------------------------------------------------------------*/
/* pisze text w oknie high */
/* moze sie pokazac wszedzie - znika po 3 sek
   nie zostawiajac message */

  void error_lotny1(text)
  char* text;
  {
   int spalte;
   spalte = ((80-strlen(text))/2)+1; /* centrowanie okna */
   open_window(10,spalte,3,strlen(text)+1,ERROR_FG, ERROR_BG);
   wprint_at(1,2,text);  /* wskazanie na tekst */
   sleep(2);
   close_window();
  }

/*******************************************************/

/*-----------------------------------------------------*/
/*
char *bios_data(void)
{
char far  *zr;
char cs[10];
zr=(char far *) ROMdata;
```

```
strcpy(cs,zr);
return(zr);
}
*/
/*-------------------------------------------------------*/
/*
haslo(void)
{
int j;
char *dat;
char data[10];

dat=bios_data();
message(HASLO);
for(j=0; j<3; j++) {
cscanf("    %s",data);
if(strcmpi(dat,data))      message(HASLO_ZLE);
 else  break;
 }
if(j<3) {
 message(HASLO_OK);
 ilkan=IL_KAN;
 }
else
{ ilkan=2;
  message(DEMO);
}
}
*/
/*-------------------------------------------------------------*/

long int czas(void)
{
long int far  *zegar;
unsigned long int czas;

zegar=(long int far *) tuczas;
czas=*zegar;
return(czas);
}
/*-------------------------------------------------------*/
zegarek(void)
{
unsigned int godz,min,sek;
unsigned long int cza;
    cza=czas();
    godz = cza / 65543;
    min  = (cza-godz*65543)/1092;
    sek  = (cza-godz*65543-min*1092) /18;
    if(sek==60) {sek=0; ++min; }
    sprintf(tekst_zegarka,"%02u:%02u:%02u",godz,min,sek);
    return(0);
 }

/*---------------------------------------*/
int data1_data2(data1,data2)
struct tm*  data1;
struct tm*  data2;
{
int a;
  a=512*data1->tm_year+32*data1->tm_mon+data1->tm_mday -
  (512*data2->tm_year+32*data2->tm_mon+data2->tm_mday);
```

```
  return(a);
}
/************************************************/
int czas1_czas2(czas1,czas2)
struct tm*  czas1;
struct tm*  czas2;
{
int a;
     a=64*czas1->tm_hour+czas1->tm_min-
       (64*czas2->tm_hour+czas2->tm_min);
 return(a);
}
/************************************************/
int do_his()
{
int a,kanal,i,j;
int czas_pop, data_pop , data_nas, czas_nas;
int day,mon,year,hour,min,kan;
double wyn;
char    godzina[12], data[12];
 if((fhis=fopen(nazwa_zbioru,"r"))==NULL){
  error(nazwa_zbioru);
  return(4);
  }
       do {        /* czytaj i zamazuj do daty poczatku */
        if((fscanf(fhis,"%*d%2d%*c%2d%*c%2d%2d%*c%2d%*c%*d%d%lf"
                  ,&data_his.tm_mday, &data_his.tm_mon, &data_his.tm_year,
                 &data_his.tm_hour,&data_his.tm_min,&kan,&wyn))<7) {
         fclose(fhis);
         return(1);
         }
       } while( (data_pop=data1_data2(&data_his,&data_po))<0 ||
        ( data_pop==0  &&  czas1_czas2(&data_his,&data_po) <0  )  ) ;
         kanal=kan;
         pomiar[kanal]=wyn;  /* kanal nr 0 zeby nie zamazac */
                                    /* od tad dobra data */
       while(1) { /* PRACUJE az gdy data za duza to else break */

        if((fscanf(fhis,"%*d%2d%*c%2d%*c%2d%2d%*c%2d%*c%*d%d%lf"
                  ,&data_his.tm_mday, &data_his.tm_mon, &data_his.tm_year,
                  &data_his.tm_hour,&data_his.tm_min,&kan,&wyn))<7) {
         fclose(fhis);
         return(2);
        }
           kanal=kan;
           pomiar[kanal]=wyn;
           data_pop=data1_data2(&data_his,&data_ko);
           czas_pop=czas1_czas2(&data_his,&data_ko);
        if( data_pop <0 ||( data_pop==0  &&  czas_pop<0 )  ) {
                       /* czy mniejszy od konca i zapamietaj
                         wartosc do porownania kiedy Nowy Czas */
             data_pom.tm_year= data_his.tm_year;
             data_pom.tm_mon = data_his.tm_mon;
             data_pom.tm_mday = data_his.tm_mday;
             data_pom.tm_hour=data_his.tm_hour;
             data_pom.tm_min =data_his.tm_min;
                          /* zapamietaj do wpisu przed zamazaniem */
         while(1) { /* CZYTA 40 az gdy nowy czas to  break */
          if((fscanf(fhis,"%*d%2d%*c%2d%*c%2d%2d%*c%2d%*c%*d%d%lf"
                    ,&data_his.tm_mday, &data_his.tm_mon, &data_his.tm_year,
                    &data_his.tm_hour,&data_his.tm_min,&kan,&wyn))<7) {
                    /* niepelna linia lub EOF */
```

```
/*          cprintf("\n\r BRAK DANYCH W ZBIORZE 3"); */
            fclose(fhis);
            zapis(); /* gdy koniec zbioru czytanego to zapisz*/
            return(1);
          }
/*
          if((data_pop<=data1_data2(&data_his,&data_ko) ) ||
           (czas_pop<=czas1_czas2(&data_his,&data_ko))) {
*/     /* data nowa wieksza LUB czas nowy wiekszy niz poprzednio */
      /* zmiany bo dwa zapisy w tym samym czasie wywalaly */

          if(kan>1) {   /* nr 1 byl czytany w poprzednim scanfie */
                        /* nie nowy pomiar */
           kanal=kan;
           pomiar[kanal]=wyn;  /* 40 odczytow */
          }
          else {   /* nowy czas pomiaru */
           if(kbhit()) {
            getch();
            if(tak_nie("  Przerwac  historie ? "))
             return(1);
           }
           zapis();  /* a potem dopiero */
           kanal=kan;   /* kanal 1 juz do next tabeli */
           pomiar[kanal]=wyn;
           break;  /* z petli  CZYTA 40 */
          }
         } /* while CZYTA 40 */
        } /* if czas w zakresie */
        else /* czas poza zakresem wydruku */
         break;    /* z petli PRACUJE */
       }  /* PRACUJE */
       return(0);
}

/*************************************************************/
int zapis()
{
int n_s,j,typ;
char godzina[20], data[20];
FILE *flab, *fopen();

  flab=fopen("lab.his","a");
  sprintf(godzina,"%02d:%02d",data_pom.tm_hour,data_pom.tm_min);
  sprintf(data,"%02d-%02d-%02d",data_pom.tm_mday,data_pom.tm_mon,data_pom.tm_year);

  for(n_s=1; n_s<=IL_STACJI; n_s++) {

    if(strcmp(data,stara_data))  {
     strcpy(stara_data,data);
     fprintf(flab,"\n|--------------------|-------------------------------------------
---------|");
     fprintf(flab,"\n| DATA: %10s  |     Stacja  Zbierania  Danych
   |",data);
     fprintf(flab,"\n|--------------------|------|------|------|------|------|------|----
--|------|");
     fprintf(flab,"\n|  Punkt pomiarowy   |   1  |   2  |   3  |   4  |   5  |   6  |   7
  |   8  |");
     fprintf(flab,"\n|--------------------|------|------|------|------|------|------|----
--|------|");
/*
     fprintf(flab,"\n|  Jednostka         | %s | %s | %s | %s | %s | %s | %s | %s |",
       kan_jednostka_opis[17], kan_jednostka_opis[17], kan_jednostka_opis[17],
       kan_jednostka_opis[17], kan_jednostka_opis[17], kan_jednostka_opis[17],
       kan_jednostka_opis[0], kan_jednostka_opis[29]);
```

```
     fprintf(flab,"\n|  Jednostka        |");
     for(j=1; j<6; j++)  {
      typ = 1+kanpar[j].kan_jednostka;
      fprintf(flab," %s |",kan_jednostka_opis[typ]);
     }
     fprintf(flab,"\n|------------------|------|------|------|------|------|------|---
--|------|");
    }


     fprintf(flab,"\n|  Godzina    %s |",godzina);
     for(j=1; j<6; j++)
     if( kanpar[j].kan_rodzaj==0)  /* w konfiguracji kanal=0 nieistniejacy */
      fprintf(flab," off  |");
     else
      if(fabs(pomiar[stacja[n_s].tem[j]])<MAX_TEMP)
       fprintf(flab,"%6.2lf|",pomiar[stacja[n_s].tem[j]]);
      else
       fprintf(flab," err%2d|",(int)(pomiar[stacja[n_s].tem[j]]/1.0E5));
     fprintf(flab,"\n|------------------|------|------|------|------|------|------|---
--|------|");

  }
       fclose(flab);
       return(0);
}
/*--  ANALIZA   ------------------------------------*/
int do_anal()
{
int kanal,i,j;
int czas_pop, data_pop , data_nas, czas_nas;
int day,mon,year,hour,min,kan;
double wyn;
char    godzina[12], data[12];
/*
FILE  *fhis , *fopen();
FILE *flab;
*/
 if((fhis=fopen(nazwa_zbioru,"r"))==NULL){
  error(nazwa_zbioru);
  return(4);
  }


      do {        /* czytaj i zamazuj do daty poczatku */
       if((fscanf(fhis,"%*d%2d%*c%2d%*c%2d%2d%*c%2d%*c%*d%d%lf"
                 ,&data_his.tm_mday, &data_his.tm_mon, &data_his.tm_year,
                 &data_his.tm_hour,&data_his.tm_min,&kan,&wyn))<7) {

/*       cprintf("\n\r BRAK DANYCH W ZBIORZE 1");*/
         fclose(fhis);
/*       error(" BRAK DANYCH #1 ! ");*/
         return(1);
         }
       } while( (data_pop=data1_data2(&data_his,&data_po))<0 ||
        ( data_pop==0  &&  czas1_czas2(&data_his,&data_po) <0  )  ) ;
         kanal=kan;
         pomiar[kanal]=wyn;  /* kanal nr 0 zeby nie zamazac */
                             /* od tad dobra data */
       while(1) { /* PRACUJE az gdy data za duza to else break */
        if((fscanf(fhis,"%*d%2d%*c%2d%*c%2d%2d%*c%2d%*c%*d%d%lf"
                  ,&data_his.tm_mday, &data_his.tm_mon, &data_his.tm_year,
                  &data_his.tm_hour,&data_his.tm_min,&kan,&wyn))<7) {
         fclose(fhis);
```

```
          oblicz();
          return(0);

        }
        kanal=kan;
        pomiar[kanal]=wyn;
        data_pop=data1_data2(&data_his,&data_ko);
        czas_pop=czas1_czas2(&data_his,&data_ko);
        if( data_pop <0 ||( data_pop==0  &&  czas_pop<0 )  ) {
                        /* czy mniejszy od konca i zapamietaj
                          wartosc do porownania kiedy Nowy Czas */
         while(1) { /* CZYTA 40 az gdy nowy czas to  break */
          if((fscanf(fhis,"%*d%2d%*c%2d%*c%2d%2d%*c%2d%*c%*d%d%lf"
                    ,&data_his.tm_mday, &data_his.tm_mon, &data_his.tm_year,
                    &data_his.tm_hour,&data_his.tm_min,&kan,&wyn))<7) {
                                        /* niepelna linia lub EOF */
             fclose(fhis);
             oblicz(); /* zbior sie skonczyl nie ma nastepnych */
             return(0);         /* zapisz_anal() przeniesione do do_analiza()*/
          }
/*
          if((data_pop<=data1_data2(&data_his,&data_ko) ) ||
           (czas_pop<=czas1_czas2(&data_his,&data_ko))) {
*/      /* nowa roznica do konca mniej ujemna */

          if(kan>1) {   /* nr 1 byl czytany w poprzednim scanfie */
                        /* nie nowy pomiar */
            kanal=kan;
            pomiar[kanal]=wyn;  /* 40 odczytow */
          }
          else {   /* nowy nr kanalu =1    */
          if(kbhit()) {
           getch();
           if(tak_nie("  Przerwac  analize ? "))
            return(1);
           }
          oblicz();  /* a potem dopiero */
          kanal=kan;   /* kanal zero juz do next tabeli */
          pomiar[kanal]=wyn;
          break;  /* z petli  CZYTA 40 */
          }
         } /* while CZYTA 40 */
        } /* if czas w zakresie */
        else /* czas poza zakresem wydruku */
         break;    /* z petli PRACUJE */
       }  /* PRACUJE */

      return(0);
}
/*************************************************************/

int zapisz_anal()
{
int j,typ;
double maxzakres, minzakres;
int przekrocz[8];
char godz_pocz[20], data_pocz[20];
char godz_konc[20], data_konc[20];

FILE *flab, *fopen();

  flab=fopen("lab.his","a");
```

```
  sprintf(godz_pocz,"%02d:%02d",data_po.tm_hour,data_po.tm_min);
  sprintf(data_pocz,"%02d-%02d-%02d",data_po.tm_mday,data_po.tm_mon,data_po.tm_year);
  sprintf(godz_konc,"%02d:%02d",data_ko.tm_hour,data_ko.tm_min);
  sprintf(data_konc,"%02d-%02d-%02d",data_ko.tm_mday,data_ko.tm_mon,data_ko.tm_year);

/*-----------------------------------------*/
/* stacja nr %d */

fprintf(flab,"\n|-------------------------------------------------------------------------------
----|");
fprintf(flab,"\n|  DATA pocz  %12s   %12s                                          |",data_pocz
,godz_pocz);
fprintf(flab,"\n|  DATA konc  %12s   %12s                                          |",data_k. .c
,godz_konc);
fprintf(flab,"\n|-------------------|-----------------------------------------------
----|");
fprintf(flab,"\n|                   |            Stacja Zbierania Danych
    |");
fprintf(flab,"\n|                   |------|------|------|------|------|------|------|--
----|");
fprintf(flab,"\n|                   |   1  |   2  |   3  |   4  |   5  |   6  |   7
 8  |");
fprintf(flab,"\n|-------------------|------|------|------|------|------|------|------|--
----|");
     fprintf(flab,"\n|  Jednostka        |");
        maxzakres = MAX_TEMP; minzakres = MIN_TEMP;
     for(j=1; j<6; j++)  {
         if(j>7)
          maxzakres = MAX_WILG; minzakres = MIN_WILG;
      przekrocz[j] = 0;   /* potem bada czy wartosc poza zakresem */
      typ = 1+ kanpar[j].kan_jednostka;
      fprintf(flab,"  %s|",kan_jednostka_opis[typ]);
     }
     fprintf(flab,"\n|-------------------|------|------|------|------|------|------|------|----
--|------|");
fprintf(flab,"\n|  WART MAX         |");
for(j=1; j<6; j++)
     if( kanpar[j].kan_rodzaj==0)
      fprintf(flab," off  |");
     else {
      if(maxp[stacja[numer_stacji].tem[j]] <= maxzakres  &&
         maxp[stacja[numer_stacji].tem[j]] >= minzakres )
       fprintf(flab,"%6.2lf|",maxp[stacja[numer_stacji].tem[j]]);
      else    {
       przekrocz[j] = 1;
       fprintf(flab," err%2d|",(int)(maxp[stacja[numer_stacji].tem[j]]/1.0E5));
      }
     }
fprintf(flab,"\n|-------------------|------|------|------|------|------|------|------|--
----|");
fprintf(flab,"\n|  WART MIN         |");
for(j=1; j<6; j++)
     if( kanpar[j].kan_rodzaj==0)
      fprintf(flab," off  |");
     else
      if(minp[stacja[numer_stacji].tem[j]] <= maxzakres  &&
         minp[stacja[numer_stacji].tem[j]] >= minzakres  && przekrocz[j]==0)
       fprintf(flab,"%6.2lf|",minp[stacja[numer_stacji].tem[j]]);
      else    {
       przekrocz[j] = 1;
       fprintf(flab," err%2d|",(int)(minp[stacja[numer_stacji].tem[j]]/1.0E5));
      }
fprintf(flab,"\n|-------------------|------|------|------|------|------|------|------
----|");
fprintf(flab,"\n|  WART SRED        |");
for(j=1; j<6; j++)
```


41

```
  char text2[]=" Podaj czas pomiedzy zapisami w sekundach";
  char bufor[80];
  int stac_nr=1;          /* 1 - 64 */
          char linia[69];
          int i;
          int pion;

          wcolor(EKRAN_FG,EKRAN_BG);
          for(i=0;i<68;i++)
           linia[i]=(char)196;
          linia[68]='\0';
          wprint_at(0,2,linia);

       clrscr();
       kierunek=1;
    while(kierunek){

       message("                          ESC-zakonczenie  ");
       kierunek=1;
       while(kierunek!=0)
       {
         gotoxy(12,6);
         sprintf(bufor,"%s",text2);
        show_int(bufor, okres_zapisu,5);
          gotoxy(wherex()-5,6);
        kierunek=get_int2( &okres_zapisu,5,5,32000);
       }
    }
    /* tu zapis parametrow tabeli str_par na dysk */
    if(tak_nie(" Zapamietac do dalszej pracy ?")){
     interwal = 18 * okres_zapisu;
     pisz_config();
    }
return(0);

}
```

```
 flab=fopen("lab.his","w");
 fclose(flab);
 long_pocz= 10000L*data_po.tm_year+100L*data_po.tm_mon+data_po.tm_mday;
 done=0;
 FIRst=1;
 sprintf(nazwa_zbioru,"BRAK_ZBIORU");
 n=czytaj_zbiory();
 sort(tytuly_zbiorow,n,strcmp,swap); /* sort(v,n,comp,exch) */
 while(sscanf(tytuly_zbiorow[i++],"%s",nazwa_zbioru)>0) {
  if((sscanf(nazwa_zbioru,"%*c%ld",&long_data)==0)) {
/*  error("  BRAK_ZBIORU *.LST "); */ /* tylko brak 1 zbioru to error */
  return(0);
  }
  if(long_data>=long_pocz)
  if(!do_his())   /* zwrocil zero his lub anal wykonana do konca czasu */
   break;
 }
 return(0);
}
/*****************************************************************/
do_analiza()
{
struct ffblk ffblk;
long int long_data, long_pocz;
int n, i=0;
message(" ANALIZA:          PROSZE  CZEKAC  -  CZYTAM ZBIORY *.LST ");
 flab=fopen("lab.his","w");
 fclose(flab);
 ini_obl();
long_pocz= 10000L*data_po.tm_year+100L*data_po.tm_mon+data_po.tm_mday;
done=0;
FIRst=1;

 sprintf(nazwa_zbioru,"BRAK_ZBIORU");
 n=czytaj_zbiory();
 sort(tytuly_zbiorow,n,strcmp,swap); /* sort(v,n,comp,exch) */
 while(sscanf(tytuly_zbiorow[i++],"%s",nazwa_zbioru)>0) {
  if((sscanf(nazwa_zbioru,"%*c%ld",&long_data)==0)) {
/*   error("  BRAK_ZBIORU *.LST ");*/
   return(1);
  }
  if(long_data>=long_pocz)
  if(!do_anal())   /* zwrocil zero his lub anal wykonana do konca czasu */
    break;
 }
/*
  if(pilosc>1)
   zapisz_anal();
*/
  return(0);
}
/*****************************************************************/

/*****************************************************************/
int wybierz_dysk(void)
{
return(0);
}
/*****************************************************************/
int wybierz_czas(void)
{
int kierunek ;
```

```
char *p;
void *malloc();
done=0;
FIRst=1;

   n_zbiorow = 0;
  while(!done) {       /* wyskoczy gdy zabraknie zbioru do czytania */
   if(n_zbiorow > MAX_ZBIOROW)
    return(-1);
   if((p =malloc(13)) == NULL)    /* max dlug nazwy zbioru DOS */
    return(-1);
   if(FIRst){
    done = findfirst("*.lst",&ffblk,0);
    FIRst=0;
   }
   else
    done = findnext(&ffblk);
   if(!done) {
    sprintf(p,"%s",ffblk.ff_name);   /* str 125 */
    tytuly_zbiorow[n_zbiorow++] = p;
   }
  }
   return(n_zbiorow);
}
/*************************************************/

sort(v,n,comp,exch)
char *v[];
int n;
int (*comp)(), (*exch)();
{
 int gap,i,j;
      for(gap=n/2; gap>0; gap/=2)
       for(i=gap; i<n; i++)
        for(j=i-gap; j>=0; j-=gap)  {
         if((*comp)(v[j],v[j+gap])<=0)
          break;
         (*exch)(&v[j],&v[j+gap]);
        }
return(0);
}
/************************************************/
swap(px,py)
char *px[], *py[];
{
  char *temp;

  temp= *px;
  *px = *py;
  *py = temp;
return(0);
}
/*************************************************************/
do_history()
{
struct ffblk ffblk;
long int long_pocz, long_data;
int n, i=0;

strcpy(stara_data,"00-00-00" );

message(" HISTORIA:     PROSZE  CZEKAC  -  CZYTAM ZBIORY *.LST ");
```

```
        przekrocz[j] = 1;
        fprintf(flab," err%2d|",(int)(sredp[stacja[numer_stacji].tem[j]]/1.0E5));
       }
fprintf(flab,"\n|------------------|------|------|------|------|------|------|------|
----|");
/*fprintf(flab,"\n|  ODCH. STAND      |");
for(j=1; j<6; j++)
   if( kanpar[j].kan_rodzaj==0)
      fprintf(flab," off  |");
     else
      if(SL[stacja[numer_stacji].tem[j]] < maxzakres  &&
         SL[stacja[numer_stacji].tem[j]] > minzakres  && przekrocz[j]==0)
       fprintf(flab,"%6.2lf|",SL[stacja[numer_stacji].tem[j]]);
      else    {
       przekrocz[j] = 1;
       fprintf(flab," err%2d|",(int)(SL[stacja[numer_stacji].wil[j]]/1.0E5));
      }
fprintf(flab,"\n|------------------|------|------|------|------|------|------|------
----|");
*/
        fclose(flab);
        return(0);
}

/*********************************************/
oblicz()
{
int kanal;
   ++pilosc;
   for(kanal=1; kanal<=IL_KAN; kanal++) {
    maxp[kanal] = max( maxp[kanal],pomiar[kanal] );
    minp[kanal] = min( minp[kanal],pomiar[kanal] );
    if(sumap[kanal]>1.0E8)   {
      error("ZA DLUGI ZBIOR  - nie liczy srednich ");
      return(1);  /* da wynik czesciowy bo wyskoczy*/
    }
    sumap[kanal] += pomiar[kanal];

    sredp[kanal] = sumap[kanal] / pilosc;
   }
 return(0);
}

y*****************/

/*****************/
ini_obl()
{
int kanal;
   for(kanal=1; kanal<=IL_KAN; kanal++) {
    maxp[kanal] = -1.0E6;
    minp[kanal] = +1.0E6;
    sumap[kanal]=0.0;
    sredp[kanal]=0.0;
//    Es[kanal]=0.0;
   }
   pilosc=0.0;
return(0);
}
/*************************************************/
int czytaj_zbiory()
{
int  n_zbiorow;
struct ffblk ffblk;
```

```
if( kanpar[j].kan_rodzaj==0)
  fprintf(flab," off  |");
 else
  if(sredp[stacja[numer_stacji].tem[j]] <= maxzakres  &&
     sredp[stacja[numer_stacji].tem[j]] >= minzakres  && przekrocz[j]==0)
   fprintf(flab,"%6.2lf|",sredp[stacja[numer_stacji].tem[j]]);
  else    {
```

```
/* PIAP  userfun.c       15.02.95       */

       #include <stdio.h>
       #include <string.h>
       #include "window3.h"
       #include "menue3.h"
       #include "stfun.h"

       #include <conio.h>
       #include <ctype.h>
       #include <alloc.h>
       #include <dos.h>
       #include <dir.h>
       #include <math.h>
       #include <time.h>
       #include <bios.h>
       #include <io.h>
       #include <fcntl.h>
       #include <errno.h>
       #include "userfun.h"
       #include "komun.h"
void zalacz1(void);
void wylacz1(void);
void zalacz2(void);
void wylacz2(void);

        void show_float_barwa1(char*,double,int,int);
        void show_int_barwa(char*,int,int,int);
int stan_dimmer, stan_zew, stan_maly, stan_fab, stan_s1 , stan_s2;
int nastawa_dimmer, zal_zew, nastawa_maly, zal_fab , zal_s1 , zal_s2;
float  napiecie;
int obroty, luksy;
int nr_danej, znak;
int kolory[17];      //tabela potrzebna do obslugi ekranu
float te1=12.2,te2=33.0;
float zad_te1=23.4,zad_te2=25.0;
int czas1=44,zad_czas1=55,czas2=0,zad_czas2=10;
int piec1=0,piec2=1;

int auto1=1,  grzeje1, czas_wl1, czas_grz1, il_grzej=0;
int auto2=1,  grzeje2, czas_wl2, czas_grz2;
#define  MAX_GRZEJ  1

/*-- Stale lokalne  ------*/
#define   ZAKONCZ "  klawisze: + -  i  strzalki    Esc - powrot do menu  "
#define   GORA  372
#define   DOL   380


       /*----------------------------prototypy funkcji w  programie */
void nadaj_zmienna(int,int);

       void jakie_kanaly(void);
       int get_int2(int*,int,int,int);
       PARAM* gmem(unsigned);
       void free_mem(PARAM*);
       int pisz_config(void);
       int czytaj_config(void);
       int tak_nie(char*);
       void jakie_stacje(void);
       STACJA* gmem_s(unsigned);
```

```
        void free_mem_s(STACJA*);
        int video_mode(void);
        void CursorNorm(void);
        void CursorBlock(void);
        void CursorOff(void);
        void ekran_pomiarow(void);
        int obsluga_ekranu(void);
        void jakie_konwersje(void);
        KONW* gmem_k(unsigned);
        void free_mem_k(KONW*);
        void historia_pom(void);
        void analiza_wynikow(void);
        void czytaj_z_b(void);
        void zapisz_na_b(void);

/*-------------------------------------------------------------*/
/*- Zmienne globalne ------------------------------------------*/

        PARAM* kanpar;          /* tabela struktur , pamiec przydzielana
                                   przez  kanpar=gmem(IL_KAN+1)
                                   odwolanie kanpar[i].xxxxxx
                                   gzie i=1...IL_KAN
                                   */
        STACJA* stacja;
        KONW* konwer;

     /* p[rzekazywane do komun.c  do historii i analizy */
     struct tm  data_po;
     struct tm  data_ko;
        int numer_stacji=1;
        /* wpisywane przy analizie danej stacji tylko do wydruku */
        float disp_rms[IL_POZ_STACJI+1],accel_rms[IL_POZ_STACJI+1],
              level_emi[IL_POZ_STACJI+1],level_noise[IL_POZ_STACJI+1];


     /* te sa w komun.c  */
     extern double moc_30[IL_KAN+1];

        extern int nr_COM,okres_zapisu;
        extern int ZAPis,MINuta;
        extern int ALArm[IL_KAN+1];
        extern double pom_wynik[IL_KAN+1];
        extern double pom_energ[IL_KAN+1];  /* LON */
        int do_history(void);   /* w komun.c */
        int do_analiza(void);
        int pomiarek(int);
        extern int ilosc_odczytow;
        float te1,te2;


/*-- Zmienne i stale lokalne w userfun.c ------*/

        int stac_bie;           /* 1 - 3 */

        #define KONS   27

        char system_config[13]="configxx.pro";

       char kan_par_opis[IL_POZ][KONS]={
        "Nazwa kanalu            : ",
        "Rodzaj kanalu   [0...24]: ",
        "Jednostka       [0...64]: ",
```

```
"Rodz.konwersji  [0....4]: ",
"Wartosc jednostki energ.: ",
"Przesuniecie            : ",
"Alarm dolny             : ",
"Alarm gorny             : ",
"Wspolcz. przel. mocy    : "
};

int kan_rodzaj_tab[25]={0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,
                        30,31,32,33,34,65};
/* kan_rodzaj = kan_rodzaj_tab[kan_rodzaj_nr] */
/* bo numery nie kolejne */
char kan_rodzaj_opis[25][21]={

/* ANALOG */
"NIEAKTYWNY          ",
"mikrovolty          ",   /* 1 */
"milivolty           ",
"miliampery          ",
"przetwornik 4-20 mA ",
"termoelement typ K  ",
"termoelement typ J  ",
"termoelement typ T  ",
"termoelement typ S  ",
"termoelement typ E  ",
"termoelement typ R  ",
"termoelement typ B  ",
"termoelement typ N  ",
"kompensacja CJ      ",
"czujnik Pt100       ",
"rezystancja         ",
"Full Bridge Str.G.  ",
"Half Bridge Str.G.  ",
"Quart.Bridge Str.G. ",  /* 18 */
"dwustanowe ON>1.5V  ",  /* 30 */
"kontakt ON=zwarty   ",  /* 31 */

/* DIGITAL */
"cyfrowe ON=1 OFF=0  ",  /* 32 */
"cyfrowe ON=0 OFF=1  ",  /* 33 */
"zliczanie zmian 0/1 ",  /* 34 */
"wyjscie cyfrowe     "   /* 65 */
};

char kan_jednostka_opis[65][5]={
"    ","kV  ","V   ","mV  ","uV  ","kA  ","A   ","mA  ",
"VA  ","Mom ","kom ","om  ","mom ","MW  ","kW  ","W   ",
"mW  ","°C  ","°F  ","°K  ","MHz ","kHz ","Hz  ","mHz ",
"rpm ","s   ","ms  ","ms  ","ppm ","%   ","kN  ","N   ",
"kg  ","g   ","mg  ","pH  ","F   ","mF  ","uF  ","nF  ",
"pF  ","bar ","mbar","Pa  ","hPa ","C   ","S   ","MT  ",
"MWh ","MW  ","kWh ","kW  ","m3  ","m3/h","GJ  ","MW  ",
"uWh ","H   ","mH  ","uH  ","km  ","m   ","cm  ","mm  ",
"um  "
};

char kan_konwersja_opis[5][21]={
"bez konwersji       ",
"y=ax3 + bx2 + cx + d",
"y=c*sqrt(ax + b) + d",
"y=c / (ax + b) + d  ",
"Yn = aXn + (1-a)Yn-1"
```

```
        };


/*--------------------------------*/


/*------------------------*/
 void jakie_kanaly()
 {
  int kierunek;
  int nr_danej;
  char bufor[80];
  int kan_nr=1;          /* 1 - 256 */
          char linia[69];
          int i;

          wcolor(EKRAN_FG,EKRAN_BG);
          for(i=0;i<68;i++)
           linia[i]=(char)196;
          linia[68]='\0';
          wprint_at(0,2,linia);

       clrscr();
       kierunek=1;
    while(kierunek){
       if((kierunek==4)&&(kan_nr>1)) kan_nr--;
       if((kierunek==5)&&(kan_nr<IL_KAN)) kan_nr++;
       sprintf(bufor,"%3d",kan_nr);
       wprint_at(0,18," WPROWADZANIE PARAMETROW KANALU Nr:     ");

       wcolor(INVERS_FG,INVERS_BG);
       wprint_at(0,54,bufor);

       wcolor(EKRAN_FG,EKRAN_BG);
       gotoxy(1,1);
       show_string(kan_par_opis[0],kanpar[kan_nr].kan_nazwa,DL_NAZWY+1);
       gotoxy(1,3);
       show_int(kan_par_opis[1],kanpar[kan_nr].kan_rodzaj_nr,3);
       show_string(" ",kan_rodzaj_opis[kanpar[kan_nr].kan_rodzaj_nr],20);
       gotoxy(1,5);
       show_int(kan_par_opis[2],kanpar[kan_nr].kan_jednostka,3);
       show_string(" ",kan_jednostka_opis[kanpar[kan_nr].kan_jednostka],20);
       gotoxy(1,7);
       show_int(kan_par_opis[3],kanpar[kan_nr].kan_konwersja,3);
       show_string(" ",kan_konwersja_opis[ konwer[kanpar[kan_nr].kan_konwersja].konw_ty
p],20);
       gotoxy(1,9);
       show_float(kan_par_opis[4],kanpar[kan_nr].kan_wzmoc,7);
       gotoxy(1,11);
       show_float(kan_par_opis[5],kanpar[kan_nr].kan_offset,7);
       gotoxy(1,13);
       show_float(kan_par_opis[6],kanpar[kan_nr].kan_alarm_low,7);
       gotoxy(1,15);
       show_float(kan_par_opis[7],kanpar[kan_nr].kan_alarm_high,7);
       gotoxy(1,17);
       show_float(kan_par_opis[8],kanpar[kan_nr].kan_alarm_przyr,7);

       message(" \30\31-wybor ENTER-potwierdzenie ESC-zakonczenie PgUp-poprzedni PgDn-n
astæpny ");
       nr_danej=1;
       kierunek=1;

       while((kierunek!=0)&&(kierunek!=4)&&(kierunek!=5))
```

```
        {
         switch(nr_danej)
         {
         case 1:
          gotoxy(KONS,1);
          kierunek=get_string1(kanpar[kan_nr].kan_nazwa,DL_NAZWY);
         break;
         case 2:
          pokaz_rodzaje();
          gotoxy(KONS,3);
          kierunek=get_int2(&kanpar[kan_nr].kan_rodzaj_nr,2,0,24);
          kanpar[kan_nr].kan_rodzaj =
           kan_rodzaj_tab[kanpar[kan_nr].kan_rodzaj_nr];
          kasuj_pokaz();
          gotoxy(KONS+3,3);
          show_string(" ",kan_rodzaj_opis[kanpar[kan_nr].kan_rodzaj_nr],20);
         break;
         case 3:
          pokaz_jednostki();
          gotoxy(KONS,5);
          kierunek=get_int2(&kanpar[kan_nr].kan_jednostka,2,0,64);
          kasuj_pokaz();
          gotoxy(KONS+3,5);
          show_string(" ",kan_jednostka_opis[kanpar[kan_nr].kan_jednostka],20);
         break;
         case 4:
          pokaz_konwersja();
          gotoxy(KONS,7);
          kierunek=get_int2(&kanpar[kan_nr].kan_konwersja,1,0,4);
          kasuj_pokaz();
          gotoxy(KONS+3,7);
          show_string(" ",kan_konwersja_opis[ konwer[kanpar[kan_nr].kan_konwersja].konw_
typ],20);
/*        show_string(" ",kan_konwersja_opis[kanpar[kan_nr].kan_konwersja],20);*/

         break;
         case 5:
          gotoxy(KONS,9);
          kierunek=get_float0(&kanpar[kan_nr].kan_wzmoc,6);
         break;
         case 6:
          gotoxy(KONS,11);
          kierunek=get_float0(&kanpar[kan_nr].kan_offset,6);
         break;
         case 7:
          gotoxy(KONS,13);
          kierunek=get_float0(&kanpar[kan_nr].kan_alarm_low,6);
         break;
         case 8:
          gotoxy(KONS,15);
          kierunek=get_float0(&kanpar[kan_nr].kan_alarm_high,6);
         break;
         case 9:
          gotoxy(KONS,17);
          kierunek=get_float0(&kanpar[kan_nr].kan_alarm_przyr,6);
         break;
         }

         if((kierunek==1)&&(nr_danej>1))
          nr_danej--;
         else
          if((kierunek>1)&&(nr_danej<IL_POZ))
           nr_danej++;
```

```
        }
      }
     /* tu zapis parametrow ze struktur kanpar[i] na dysk */
     if(tak_nie("Zapisac parametry kanalow na dysk ?"))
      pisz_config();
     else
     czytaj_config();
}
/*=============== FUNKCJE LOKALNE DLA jakie_kanaly */
/*-----------------------*/
int pokaz_rodzaje()
{
int i;
 open_window(6,KONS+6,13,46,HELP_FG,HELP_BG);
 wprint_at(0,3," Rodzaje kanalow ");
 for(i=0;i<12;i++){
  gotoxy(1,1+i);
  cprintf("%2d.%s",2*i,kan_rodzaj_opis[2*i]);
  cprintf(" %2d.%s",2*i+1,kan_rodzaj_opis[2*i+1]);
 }
 gotoxy(1,1+i);
 cprintf("%2d.%s",2*i,kan_rodzaj_opis[2*i]);
 window(2,3,79,23);
 return(0);
}

/*-----------------------*/
int pokaz_jednostki()
{
int i;
 open_window(8,KONS+6,13,42,HELP_FG,HELP_BG);
 wprint_at(0,3," Rodzaje jednostek ");
 for(i=0;i<13;i++){
  gotoxy(1,1+i);
  cprintf(" %2d.%s",5*i,kan_jednostka_opis[5*i]);
  cprintf(" %2d.%s",5*i+1,kan_jednostka_opis[5*i+1]);
  cprintf(" %2d.%s",5*i+2,kan_jednostka_opis[5*i+2]);
  cprintf(" %2d.%s",5*i+3,kan_jednostka_opis[5*i+3]);
  cprintf(" %2d.%s",5*i+4,kan_jednostka_opis[5*i+4]);
 }
 window(2,3,79,23);
 return(0);
}

/*-----------------------*/
int pokaz_konwersja()
{
int i;
 open_window(10,KONS+6,5,26,HELP_FG,HELP_BG);
 wprint_at(0,3," Rodzaje konwersji ");
 for(i=0;i<5;i++){
  gotoxy(1,1+i);
  cprintf(" %d. %s",i,kan_konwersja_opis[konwer[i].konw_typ]);
 }
 window(2,3,79,23);
 return(0);
}

/*--------------------*/
int kasuj_pokaz()
{
```

```
  close_window();
  return(0);
 }
/*========================== */

/*---------------------------------*/
/* modyfikacje do profi JK 22.10.92  PgUp PgDn, min, max */
        /* pobierz liczbe int o dlugosci   w zakresie min,max */
        /*przy wywolaniu &nazwa             */
        /* wez z nazwa oddaj do nazwa       */
        /*wynik:0-zakonczono ESC
                1-zakonczono Up
                2-zakonczono Down
                3-zakonczono ENTER
                4-zakonczono PgUp
                5-zakonczono PgDn */

        int get_int2(nazwa,dlugosc,min,max)
         int *nazwa;
         int dlugosc;
         int min;
         int max;
         {
          char bufor[80];
          int wynik,a,b,gotowe=1;
          int nazwax;

          a=wherex();
          b=wherey();
          while(gotowe)
          {
           gotoxy(a,b);
           nazwax=*nazwa;
           sprintf(bufor,"%d",nazwax);
           wynik=get_string1(bufor,dlugosc);
           nazwax=atoi(bufor);
            if((nazwax>=min)&&(nazwax<=max))
             {
              *nazwa=nazwax;
              gotoxy(a,b);
              wcolor(INVERS_FG,INVERS_BG);
              cprintf("%-*d",dlugosc,nazwax);
              wcolor(EKRAN_FG,EKRAN_BG);
              gotowe=0;
             }
          }
          return(wynik);
         }
/*---------------------------------*/
/*---------------------------------*/
/* modyfikacje do scanlab JK 04.93 Left Right */
/* to samo co get_int2 tyko uzywa get_string2 */

        int get_int3(nazwa,dlugosc,min,max)
         int *nazwa;
         int dlugosc;
         int min;
         int max;
         {
          char bufor[80];
          int wynik,a,b,gotowe=1;
          int nazwax;
```

```
          a=wherex();
          b=wherey();
          while(gotowe)
          {
           gotoxy(a,b);
           nazwax=*nazwa;
           sprintf(bufor,"%d",nazwax);
           wynik=get_string2(bufor,dlugosc);
           nazwax=atoi(bufor);
            if((nazwax>=min)&&(nazwax<=max))
             {
              *nazwa=nazwax;
              gotoxy(a,b);
              wcolor(INVERS_FG,INVERS_BG);
              cprintf("%-*d",dlugosc,nazwax);
              wcolor(EKRAN_FG,EKRAN_BG);
              gotowe=0;
             }
          }
          return(wynik);
         }

/*---------------------------------*/
/* modyfikacje do scanlab JK 04.93 Left Right */
/* to samo co get_float0 tyko uzywa get_string2 */

        int get_float03(nazwa,dlugosc)
         float *nazwa;
         int dlugosc;
         {
          char bufor[80];
          int wynik,a,b;
          float nazwax;

          a=wherex();
          b=wherey();
           nazwax=*nazwa;
           sprintf(bufor,"%-*.3f",dlugosc,nazwax);    /* zamienic liczbe na string */
           bufor[dlugosc]='\0';
           wynik=get_string2(bufor,dlugosc);

            nazwax=(float)atof(bufor);    /* zamienic string na liczbe */
              *nazwa=nazwax;
              gotoxy(a,b);       /*likwidacja nie przyjetych znakow */
              wcolor(INVERS_FG,INVERS_BG);
              sprintf(bufor,"%-*.3f",dlugosc,nazwax);
              bufor[dlugosc]='\0';
              cprintf("%s",bufor);
              wcolor(EKRAN_FG,EKRAN_BG);
          return(wynik);
         }


/*---------------------------------*/
/*---------------------------------*/
PARAM* gmem(number)
unsigned number;
{
   PARAM* cp;
   if((cp=(PARAM*)calloc(number,sizeof(PARAM)))==NULL){
    error("  ZA MALO MIEJSCA W PAMIECI !!");
```

```
    return(NULL);
   }
   return (cp);
}

/*-------------------------------------*/
void free_mem(obj)
 PARAM* obj;
 {
 free(obj);
 }

/*---------------------------------------------*/
/* zapamietaj opisy w zbiorze system_config */
  int pisz_config()
{
 int i,j;
 FILE *f;
    message("                          CZEKAJ... ");
   f=fopen(system_config,"w");
 if (f==NULL)
  { error("  Nie mozna zapisac konfiguracji na dysk !!! ");
    return(1);
  }

   for(i=1;i<IL_KAN+1;i++){
    fprintf(f,"%-*s\n",DL_NAZWY,kanpar[i].kan_nazwa);
    fprintf(f,"%d %d %d %d %f %f %f %f %f\n",
        kanpar[i].kan_rodzaj_nr,
        kanpar[i].kan_rodzaj,
        kanpar[i].kan_jednostka,
        kanpar[i].kan_konwersja,
        kanpar[i].kan_wzmoc,
        kanpar[i].kan_offset,
        kanpar[i].kan_alarm_low,
        kanpar[i].kan_alarm_high,
        kanpar[i].kan_alarm_przyr);
   }

/*

             for(i=1;i<IL_STACJI+1;i++)  {
              fprintf(f,"%-*s\n",DL_OPISU_STACJI,stacja[i].opis_stacji);
              for(j=1;j<IL_POZ_STACJI;j++){
               fprintf(f,"%d %d %d ",
               stacja[i].tem[j],stacja[i].wil[j],stacja[i].cis[j]);
              }
              fprintf(f,"%d %d %d\n",
               stacja[i].tem[j],stacja[i].wil[j],stacja[i].cis[j]);
             }
*/

       for(i=1;i<5;i++){
        fprintf(f,"%d %f %f %f %f\n",
        konwer[i].konw_typ,
        konwer[i].konw_a,
        konwer[i].konw_b,
        konwer[i].konw_c,
        konwer[i].konw_d);
       }

/*QA*/
```

```
        fprintf(f,"%d %d \n",nr_COM,okres_zapisu);
 fclose(f);
 return(0);
}

/*-----------------------------------------*/
/*czyta opisy z dysku i umieszcza w strukturach
  jesli nie ma zbioru 1, jesli ok 0  */

  int czytaj_config()
  {
 int i,j,zle;
 FILE *f;
 char buf1[80];
    message("                          CZEKAJ... ");
   f=fopen(system_config,"r");
 if (f==NULL)
  {
    error(" Nie ma zbioru konfiguracyjnego !!! ");
    return(1);
  }

    zle=0;
    for(i=1;i<IL_KAN+1;i++){
     if( fgets(buf1,DL_NAZWY,f)==NULL){
      zle=1;
      break;
     }
     sprintf(kanpar[i].kan_nazwa,buf1);
     if( fscanf(f,"%d%d%d%d%f%f%f%f%f\n",
        &(kanpar[i].kan_rodzaj_nr),
        &(kanpar[i].kan_rodzaj),
        &(kanpar[i].kan_jednostka),
        &(kanpar[i].kan_konwersja),
        &(kanpar[i].kan_wzmoc),
        &(kanpar[i].kan_offset),
        &(kanpar[i].kan_alarm_low),
        &(kanpar[i].kan_alarm_high),
        &(kanpar[i].kan_alarm_przyr) )==-1){
        zle=1;
        break;
     }
   }
/*
   if(!zle){
    for(i=1;i<IL_STACJI+1;i++){
     if( fgets(buf1,DL_OPISU_STACJI,f)==NULL){
      zle=1;
      break;
     }
     sprintf(stacja[i].opis_stacji,buf1);

     for(j=1;j<IL_POZ_STACJI;j++)
      if( fscanf(f,"%d%d%d",&(stacja[i].tem[j]),
                            &(stacja[i].wil[j]),
                            &(stacja[i].cis[j]) )==-1){
       zle=1;
       break;
      }
     if(!zle){
      if( fscanf(f,"%d%d%d\n",&(stacja[i].tem[j]),
                              &(stacja[i].wil[j]),
```

```
                                &(stacja[i].cis[j]) )==-1){
          zle=1;
          break;
         }
        }
       }
      }
   */
      if(!zle){
       for(i=1;i<5;i++){
        if( fscanf(f,"%d%f%f%f%f\n",
           &(konwer[i].konw_typ),
           &(konwer[i].konw_a),
           &(konwer[i].konw_b),
           &(konwer[i].konw_c),
           &(konwer[i].konw_d) )==-1){
           zle=1;
           break;
        }
       }
      }
      if(!zle){
       if(fscanf(f,"%d %d \n",&nr_COM,&okres_zapisu)==-1)
        zle=1;
      }
    fclose(f);
    if (zle) {
     error(" Zly format zbioru konfiguracyjnego !!! ");
     return(1);
    }
    return(0);
   }
   /*-----------------------------------------------------------------*/
   /* pisze text w oknie high
      pyta czy tak czy nie, zwraca  tak=1 nie=0 */

     int tak_nie(text)
     char* text;
     {
      int spalte;
      int kl;
      spalte = ((80-strlen(text))/2)+1; /* centrowanie okna */
      open_window(10,spalte,2,strlen(text)+3,MESSAGE_FG,MESSAGE_BG);
      message(" TAK: Nacisnij klawisz T     NIE: Dowolny inny klawisz ");
      gotoxy(2,1);
      cprintf("%s",text);
      gotoxy((strlen(text)+3)/2 - 5,2);
      cprintf("TAK / NIE ?");

      kl=get_znak();        /* i czekaj */
      close_window();
         if((kl=='t')||(kl=='T'))
          return(1);
         else
          return(0);
     }

     /*--------------------------------*/

   /*--------------------------------*/
   STACJA* gmem_s(number)
   unsigned number;
```

```
{
   STACJA* cp;
   if((cp=(STACJA*)calloc(number,sizeof(STACJA)))==NULL){
    error("  ZA MALO MIEJSCA W PAMIECI !!");
    return(NULL);
   }
   return (cp);
}

/*-------------------------------------*/
void free_mem_s(obj)
 STACJA* obj;
 {
 free(obj);
 }

 /*------------------------------------------------ ---------*/
int video_mode()   /* dopasuj sie do rodzaju monitora */
{
    union REGS r;

   r.h.ah = 0x0F;      /*  GET_VIDEO_MODE    */
   int86(0x10,&r,&r);  /*  VIDEO_INT  */
   return(r.h.al);
}
/*------ ------------------------------------------------------*/
void CursorNorm()
{
   union REGS r;
   if( video_mode()== 7)
    r.x.cx = 0x0B0C;
   else
    r.x.cx= 0x0607;
   r.x.ax = 0x0100;
   int86(0x10,&r,&r);  /*  VIDEO_INT  */

}
/*-------------------------------------------------------------*/
void CursorBlock()
{
   union REGS r;
   if( video_mode()== 7)
    r.x.cx = 0x000C;
   else
    r.x.cx= 0x0007;
   r.x.ax = 0x0100;
   int86(0x10,&r,&r);  /*  VIDEO_INT  */

}
/*-------------------------- ---  ----------------------------*/
void CursorOff()
{
   union REGS r;
   if( video_mode()== 7)
    r.x.cx = 0x3000;
   else
    r.x.cx= 0x2000;
   r.x.ax = 0x0100;
   int86(0x10,&r,&r);  /*  VIDEO_INT  */

}
 /*------ ------------------------------------------------*/
```

```
  void ekran_pomiarow()
  {
    char bufor[80];
    int i;
  int polecenie_obslugi;
        CursorOff();
        clrscr();
        bufor[0]=' ';              /*czyszczenie gornej linii*/
        for(i=1;i<68;i++)
         bufor[i]=(char)196;
        bufor[68]='\0';
        wprint_at(0,2,bufor);
        wprint_at(0,3," POMIARY ");
/*      wprint_at(0,46," Stacja    "); */
        stac_bie=1;
        message(ZAKONCZ);
        while(1){
         if(pomiarek(0)==27)
          break;
/*QA*/ /*        if(ilosc_odczytow !=0 )*/
         polecenie_obslugi=obsluga_ekranu();
         if(polecenie_obslugi==2){
          MINuta=1;
          ZAPis=1;
         }
         if(polecenie_obslugi==1)
          break;
         if(polecenie_obslugi==3) {
          exit_func();
          break;
         }
        }
        CursorNorm();
  }

  /*-------------------------------------------------------*/
/*
     zwraca 1 jesli ESC
            2 jesli ENTER */

  int obsluga_ekranu(void)
  {
  char bufor[80];
  struct date dzisiaj;
  struct time godzina;
  int kierunek,i,bie_kan,x;
  int pom_wysw;
  int barwa;


   gettime(&godzina);
        getdate(&dzisiaj);
/* sprintf(bufor,"%2d",stac_bie);
        wcolor(INVERS_FG,INVERS_BG);
        wprint_at(0,53,bufor);
*/
        wcolor(EKRAN_FG,EKRAN_BG);


        gotoxy(1,1);
        cprintf(" Data: %02d.%02d.%d       Godzina: %02d:%02d:%02d\n",
        (int)dzisiaj.da_day,(int)dzisiaj.da_mon,dzisiaj.da_year,
```

```
        (int)godzina.ti_hour,(int)godzina.ti_min,(int)godzina.ti_sec);

        gotoxy(1,2);
        show_string(" ",stacja[stac_bie].opis_stacji,DL_OPISU_STACJI+1);
        /*&&&&&&&&&&*/
//cprintf("\n\r ilosc odczytow danych od poczatku pomiaru  %d ",ilosc_odczytow);
        /*&&&&&&&&&&*/
        gotoxy(DL_NAZWY+5,4);  cprintf("w. chwilowa");
        gotoxy(45,4);  cprintf("w. srednia");
        gotoxy(60,4);  cprintf("w. zliczona");

        for(i=1;i<IL_KAN+1;i++){
         gotoxy(1,2*i+4);
         cprintf("%d",i);
         show_string(" ",kanpar[i].kan_nazwa,DL_NAZWY+1);
         if(kanpar[i].kan_rodzaj>0){
          show_float_barwa("  ",pom_wynik[i],8,ALArm[i]);

         }
         else
          show_string(" "," off",8);

         sprintf(bufor," %s",kan_jednostka_opis[kanpar[i].kan_jednostka + 1]);
         cprintf("%s",bufor);
         gotoxy(45,2*i+4);  /*LON */
         show_float_barwa(" ",moc_30[i],8,ALArm[i]);
         sprintf(bufor," %s",kan_jednostka_opis[kanpar[i].kan_jednostka +1]);
         cprintf("%s",bufor);

         gotoxy(60,2*i+4);  /*LON */
         show_float_barwa(" ",pom_energ[i],8,ALArm[i]);
         sprintf(bufor," %s",kan_jednostka_opis[kanpar[i].kan_jednostka]);
         cprintf("%s",bufor);
        }
        if(kbhit()){
         kierunek=get_znak();
         if(kierunek==27) return(1);
         if(kierunek==13) return(2);
         if(kierunek==113) return(3);  /* q wyjscie z apisem LonWorka config*/

/*       if((kierunek==373)&&(stac_bie>1)) stac_bie--;
         if((kierunek==381)&&(stac_bie<IL_STACJI)) stac_bie++;
*/
        }
        return(0);

  }
/*-------------------------------------*/
KONW* gmem_k(number)
unsigned number;
{
   KONW* cp;
   if((cp=(KONW*)calloc(number,sizeof(KONW)))==NULL){
    error("  ZA MALO MIEJSCA W PAMIECI !!");
    return(NULL);
   }
   return (cp);
}

/*-------------------------------------*/
void free_mem_k(obj)
 KONW* obj;
```

```
 {
 free(obj);
 }
/*-------------------------- ----- -----*/

void jakie_konwersje()
{
  int kierunek;
  int nr_danej;
  char bufor[80];
  int konw_nr=1;          /* 1 - 4 */
          char linia[69];
          int i;

          wcolor(EKRAN_FG,EKRAN_BG);
          for(i=0;i<68;i++)
           linia[i]=(char)196;
          linia[68]='\0 ;
          wprint_at(0,2,linia);

        clrscr();
        kierunek=1;
      while(kierunek){
        if((kierunek==4)&&(konw_nr>1)) konw_nr-- ;
        if((kierunek==5)&&(konw_nr<4)) konw_nr++;
        sprintf(bufor,"%3d",konw_nr);
        wprint_at(0,18," WPROWADZANIE PARAMETROW KONWERSJI Nr:      ");
        wcolor(INVERS_FG,INVERS_BG);
        wprint_at(0,57,bufor);

        wcolor(EKRAN_FG,EKRAN_BG);
        gotoxy(1,1);
        show_int("Rodz.konwersji  [0....4]: ",konwer[konw_nr].konw_typ,3);
        gotoxy(1,3);
        show_float("Parametr konwersji a    : ",konwer[konw_nr].konw_a,7);
        gotoxy(1,5);
        show_float("Parametr konwersji b    : ",konwer[konw_nr].konw_b,7);
        gotoxy(1,7);
        show_float("Parametr konwersji c    : ",konwer[konw_nr].konw_c,7);
        gotoxy(1,9);
        show_float("Parametr konwersji d    : ",konwer[konw_nr].konw_d,7);

        message(" \30\31-wybor ENTER-potwierdzenie ESC-zakonczenie PgUp-poprzedni PgDn n
astæpny ");
        nr_danej=1;
        kierunek=1;

        while((kierunek!=0)&&(kierunek!=4)&&(kierunek!=5))
        {
         switch(nr_danej)
         {
         case 1:
          pokaz_konwersja1();
          gotoxy(KONS,1);
          kierunek=get_int2(&konwer[konw_nr].konw_typ,1,0,4);
          kasuj_pokaz();
          gotoxy(KONS+3,1);
          show_string(" ",kan_konwersja_opis[konwer[konw_nr].konw_typ],20);
         break;
         case 2:
          gotoxy(KONS,3);
          kierunek=get_float0(&konwer[konw_nr].konw_a,6);
         break;
```

```
          case 3:
           gotoxy(KONS,5);
           kierunek=get_float0(&konwer[konw_nr].konw_b,6);
          break;
          case 4:
           gotoxy(KONS,7);
           kierunek=get_float0(&konwer[konw_nr].konw_c,6);
          break;
          case 5:
           gotoxy(KONS,9);
           kierunek=get_float0(&konwer[konw_nr].konw_d,6);
          break;
          }


          if((kierunek==1)&&(nr_danej>1))
           nr_danej--;
          else
           if((kierunek>1)&&(nr_danej<5))
            nr_danej++;

         }
       }
      /* tu zapis parametrow ze struktur konwer[i] na dysk */
      if(tak_nie("Zapisac parametry konwersji na dysk ?"))
       pisz_config();
      else
       czytaj_config();
}
 /*----------------------*/
 int pokaz_konwersja1()
 {
 int i;
  open_window(4,KONS+6,5,26,HELP_FG,HELP_BG);
  wprint_at(0,3," Rodzaje konwersji ");
  for(i=0;i<5;i++){
   gotoxy(1,1+i);
   cprintf(" %d. %s",i,kan_konwersja_opis[i]);
  }
  window(2,3,79,23);
  return(0);
 }

   /*-----------------------------------------------------*/
int display_tabele()
{
 FILE *f;
 char bufor[80];
 char* linia;
 int err=0,znak,i,j,k,status;
  clrscr();
  message("                          CZEKAJ... ");
/*  system("copy lab.his temp_____.wyn > NUL");*/

  clrscr();
  f=fopen("lab.his","r");
  if (f==NULL)
  {
    error(" Nie ma zbioru wynikow !!! ");
    return(1);
  }
```

```
  message(" ESC - zakonczenie  D   drukowanie   Dowolny inny klawisz - dalej");
  while(1){
   for(i=1;i<21;i++){
    linia=fgets(bufor,80,f);
    if(linia==NULL)
     break;
   bufor[strlen(bufor)-1]=0;
    cprintf("%s",bufor);
    if (wherex()>1)
        cprintf("\n\r");

   }
   znak=get_znak();
   if((znak==27)|||(znak=='D')|||(znak=='d'))  /* nacisnieto ESC lub d */
    break;
  }
  fclose(f);
  if((znak=='D')|||(znak=='d'))
   if(tak_nie("CZY DRUKARKA OK ?"))  {
        message(" Esc - przerwac        PROSZE  CZEKAC    -  DRUKOWANIE  WYNIKOW" );
                    /* system("copy temp_____.wyn PRN > NUL");*/
/**TG**/
     if((f=fopen("lab.his","r"))==NULL){
      error("BRAK lab_his");
      return(4);
     }

     status = biosprint(2,0,0);    /* nie wlaczona */
      if (status &0x08)
       error_lotny("    Sprawdz zasilanie drukarki   ");
     status = biosprint(2,0,0);  /* jesli byla wlaczona bez papieru */
      if (!(status &0x80))       /* i wlozonoi papier to brak ON LINE */
       error_lotny("    Sprawdz przycisk   ON LINE    drukarki ");

     while( fgets(bufor,80,f) ) {
           if(kbhit()) {
            getch();
            if(tak_nie("    Przerwac  drukowanie ? ")) {
             for(j=0; j<600; j++)
              message("       PRZERWANO  DRUKOWANIE  -  BUFOR DRUKARKI JESZCZE PELEN",,
               /*system("del temp_____.wyn > NUL");*/
             return(0);
            }
           }


           i=0;
      while(1) {
       status = biosprint(2,0,0);
       err=0;
       if (status &0x01)
        err=error_lotny("        Uszkodzona   drukarka  ?      ");
       if (status &0x08)
        err=error_lotny("    Sprawdz  zasilanie  drukarki    ");
       if (status &0x20)  {
        err=error_lotny("  Brak papieru w drukarce  lub  odlaczony kabel");
        status = biosprint(2,0,0);  /* jesli skonczyl sie papier */
        if(!(status &0x80))       /* i wlozonoi papier to brak ON LINE */
         error_lotny("  Sprawdz przycisk    ON LINE   drukarki ");
       }
       if(err==0)  /* nie ma bledu */
       break;
```

```
        if(err==27)
        return(1);
       } /* while 1  badanie statusu */

       while((znak=(int)bufor[i++])!=0)  {
        while(1) {
         status = biosprint(2,0,0);
         err=0;
         if (status &0x01)
          err=error_lotny("   Uszkodzona   drukarka   ?       ");
         if (status &0x08)
          err=error_lotny("    Sprawdz   zasilanie   drukarki   ");
         if (status &0x20) {
          err=error_lotny("  Brak papieru w drukarce  lub  odlaczony kabel");
          status = biosprint(2,0,0);  /* jesli skonczyl sie papier */
          if(!(status &0x80))       /* i wlozonoi papier to brak ON LINE */
           error_lotny("  Sprawdz przycisk    ON LINE    drukarki ");
         }
         if(err==0)  /* nie ma bledu */
          break;
         if(err==27)
          return(1);
        }
        biosprint(0,znak, 0);
       }
      }   /* while fgets nie zwroci EOF */
       while((znak=(int)bufor[i++])!=0)  /* ostatnia linia */
        biosprint(0,znak, 0);
    } /* if DRUKARKA TAK */
//LON  system("del temp_____.wyn > NUL");
  return(0);

}

/*--------------------------------*/
/* pyta o date i czas - bieze ze struktury dp  i tam oddaje*/
/* dp  wskaznik do struktury */
/* lokalna dla historia_pom i analiza_pom */

int daj_date_czas(dp,tekst)
struct tm* dp;
char* tekst;
{
  int kierunek;
  int nr_danej;
  int d1,d2,d3;
  int t1,t2;
        open_window(6,30,10,18,EKRAN_FG,EKRAN_BG);
        message(" ENTER - potwierdzenie  PgDn - zakonczenie  ESC - anuluj");
        wprint_at(0,3,tekst);
        gotoxy(1,2);
        show_int("Rok    :",dp->tm_year,2);
        gotoxy(1,4);
        show_int("Miesiac:",dp->tm_mon,2);
        gotoxy(1,6);
        show_int("Dzien  :",dp->tm_mday,2);
        gotoxy(1,8);
        show_int("Godzina:",dp->tm_hour,2);
        gotoxy(1,10);
        show_int("Minuta :",dp->tm_min,2);

        nr_danej=1;
```

```
        kierunek=1;

        d1=dp->tm_year;
        d2=dp->tm_mon;
        d3=dp->tm_mday;
        t1=dp->tm_hour;
        t2=dp->tm_min;

        while(kierunek!=5)  /* PgDn */
        {
         switch(nr_danej)
         {
         case 1:
          gotoxy(9,2);
          kierunek=get_int2(&d1,2,90,99);
         break;
         case 2:
          gotoxy(9,4);
          kierunek=get_int2(&d2,2,1,12);
         break;
         case 3:
          gotoxy(9,6);
          kierunek=get_int2(&d3,2,1,31);
         break;
         case 4:
          gotoxy(9,8);
          kierunek=get_int2(&t1,2,0,23);
         break;
         case 5:
         gotoxy(9,10);
          kierunek=get_int2(&t2,2,0,59);
         break;
         }
         if(kierunek==0)
          break;

         if((kierunek==1)&&(nr_danej>1))
          nr_danej--;
         else
          if((kierunek>1)&&(nr_danej<5))
           nr_danej++;

        }
        dp->tm_year=d1;
        dp->tm_mon=d2;
        dp->tm_mday=d3;
        dp->tm_hour=t1;
        dp->tm_min=t2;
        close_window();

        if(kierunek==0)
         return(1);
        else
         return(0);
}
/*-------------------------------------*/
/* pyta o date - bieze ze struktury dp i tam oddaje*/
/* dp - wskaznik do struktury */

int daj_date(dp,tekst)
struct date* dp;
char* tekst;
```

```
{
  int kierunek;
  int nr_danej;
  int d1,d2,d3;
        open_window(6,30,7,18,EKRAN_FG,EKRAN_BG);
        message(" ENTER - potwierdzenie  PgDn - zakonczenie  ESC - anuluj");
        wprint_at(0,3,tekst);
        gotoxy(1,2);
        show_int("Rok    :",dp->da_year,2);
        gotoxy(1,4);
        show_int("Miesiac:",(int)dp->da_mon,2);
        gotoxy(1,6);
        show_int("Dzien  :",(int)dp->da_day,2);

        nr_danej=1;
        kierunek=1;

        d1=dp->da_year;
        d2=(int)dp->da_mon;
        d3=(int)dp->da_day;

        while(kierunek!=5)
        {
         switch(nr_danej)
         {
         case 1:
          gotoxy(9,2);
          kierunek=get_int2(&d1,2,90,99);
         break;
         case 2:
          gotoxy(9,4);
          kierunek=get_int2(&d2,2,1,12);
         break;
         case 3:
          gotoxy(9,6);
          kierunek=get_int2(&d3,2,1,31);
         break;
         }

         if(kierunek==0)
          break;

         if((kierunek==1)&&(nr_danej>1))
          nr_danej--;
         else
          if((kierunek>1)&&(nr_danej<3))
           nr_danej++;

        }
        dp->da_year=d1;
        dp->da_mon=(char)d2;
        dp->da_day=(char)d3;
        close_window();
        if(kierunek==0)
         return(1);
        else
         return(0);
}
/*--------------------------*/
int wez_czas(teraz)
 struct tm *teraz;
{
```

```
  time_t t;
  struct tm *gmt;
        t=time(NULL);
        gmt=localtime(&t);
        teraz->tm_year=gmt->tm_year;
        teraz->tm_mon=(gmt->tm_mon)+1;
        teraz->tm_mday=gmt->tm_mday;
        teraz->tm_hour=gmt->tm_hour;
        teraz->tm_min=gmt->tm_min;
}

/*--------------------------------*/
/* pyta o poczatek i koniec, wywoluje do_histo i wyswietla
   zbior wynikowy */

void historia_pom()
{
  char bufor[80];
  int i;

          for(i=0;i<68;i++)
            bufor[i]=(char)196;
          bufor[68]='\0';
          wprint_at(0,2,bufor);
          wprint_at(0,3," WYNIKI ");
          clrscr();

        data_po.tm_year=94;
        data_po.tm_mon=12;
        data_po.tm_mday=1;
        data_po.tm_hour=0;
        data_po.tm_min=0;

        wez_czas(&data_ko);

        if(daj_date_czas(&data_po,"DATA POCZATKOWA"))
        {
         clrscr();
         return;
        }
        gotoxy(1,2);
        cprintf(" DATA POCZATKOWA:  %02d.%02d.%02dr.\n\r",
        data_po.tm_mday,data_po.tm_mon,data_po.tm_year);
        cprintf(" GODZINA          :  %02d:%02d",
        data_po.tm_hour,data_po.tm_min);

        if(daj_date_czas(&data_ko,"DATA KONCOWA"))
        {
         clrscr();
         return;
        }
        gotoxy(1,5);
        cprintf(" DATA KONCOWA    :  %02d.%02d.%02dr.\n\r",
        data_ko.tm_mday,data_ko.tm_mon,data_ko.tm_year);
        cprintf(" GODZINA          :  %02d:%02d",
        data_ko.tm_hour,data_ko.tm_min);

        if(tak_nie("Podac historie pomiarow ?")){

         /*funkcja robiaca lab.his - historia */
         if(do_history()==0)
          display_tabele();
```

```
        }
         clrscr();

}
/*-------------------------------------*/

/*---------------------------------*/
/* pyta o poczatek, koniec  i numer stacji, wywoluje do_anali i wyswietla
   zbior wynikowy */
void analiza_pom()
{
  char bufor[80];
  int i;
          for(i=0;i<68;i++)
            bufor[i]=(char)196;
          bufor[68]='\0';
          wprint_at(0,2,bufor);
          wprint_at(0,3," ANALIZA POMIAROW ");
          clrscr();

        data_po.tm_year=94;
        data_po.tm_mon=12;
        data_po.tm_mday=1;
        data_po.tm_hour=0;
        data_po.tm_min=0;

        wez_czas(&data_ko);

        if(daj_date_czas(&data_po,"DATA POCZATKOWA"))
        {
         clrscr();
         return;
        }
        gotoxy(1,2);
        cprintf(" DATA POCZATKOWA:  %02d.%02d.%02dr.\n\r",
        data_po.tm_mday,data_po.tm_mon,data_po.tm_year);
        cprintf(" GODZINA          :  %02d:%02d",
        data_po.tm_hour,data_po.tm_min);

        if(daj_date_czas(&data_ko,"DATA KONCOWA"))
        {
         clrscr();
         return;
        }
        gotoxy(1,5);
        cprintf(" DATA KONCOWA     :  %02d.%02d.%02dr.\n\r",
        data_ko.tm_mday,data_ko.tm_mon,data_ko.tm_year);
        cprintf(" GODZINA          :  %02d:%02d",
        data_ko.tm_hour,data_ko.tm_min);

/*
        open_window(10,30,3,18,EKRAN_FG,EKRAN_BG);
        message(" ENTER - potwierdzenie  ESC - zakonczenie ");
        wprint_at(0,3,"STACJA");
        gotoxy(1,2);
        show_int("Nr stacji     :",numer_stacji,2);
        gotoxy(15,2);
        get_int2(&numer_stacji,2,1,3);
        close_window();
        gotoxy(1,8);
        cprintf(" STACJA NUMER    : %d",numer_stacji);
*/
```

```
        /* dodatkowe_dane(); */

        if(tak_nie("Wykonac analize pomiarow ?")){

         /*funkcja robiaca lab.his - analiza */
                if(do_analiza()==0)
          display_tabele();
        }
         clrscr();

}
/*------------------------------------------- -----*/

  void display_maske1()
  {
   int i;
        for(i=0;i<10;i++)
         kolory[i]=2;

        nr_danej=1;
        znak=0;
        kolory[1]=3;

        wcolor(INVERS_FG,INVERS_BG);
        gotoxy(2,1);
        cputs(" STEROWANIE PIECAMI  ");
        wcolor(EKRAN_FG,EKRAN_BG);

        gotoxy(1,3);
        cprintf(" Data:            Godzina:");

        // modul IMT Sysmik
        gotoxy(15,6);
        show_float_barwa1("Zadana temp 1: ",zad_te1,5,kolory[1]);
        cputs(" °C");
        gotoxy(45,6);
        show_float_barwa1("Temperatura1: ",te1,5,kolory[0]);
        cputs(" °C");
        gotoxy(15,8);
        show_int_barwa("Max czas pracy ",zad_czas1,5,kolory[2]);
        cputs(" sek");
        gotoxy(45,8);
        show_int_barwa(" czas pracy   ",czas_grz1,5,kolory[2]);
        cputs("  sek");
        gotoxy(30,10);
        show_int_barwa("Automatyka     pieca     ",auto1,3,kolory[3]);

        //druga czesc
        gotoxy(15,16);
        show_float_barwa1("ZADANA TEMP 2: ",zad_te2,5,kolory[4]);
        cputs(" °C");
        gotoxy(45,16);
        show_float_barwa1("Temperatura2: ",te2,5,kolory[0]);
        cputs(" °C");
        gotoxy(15,18);
        show_int_barwa("Max czas pracy ",zad_czas2,5,kolory[5]);
        cputs("sek");
        gotoxy(45,18);
        show_int_barwa(" czas pracy   ",czas_grz2,5,kolory[5]);
        cputs("sek");
        gotoxy(30,20);
        show_int_barwa("Automatyka     pieca     ",auto2,3,kolory[6]);
```

```
        //modul IMC Sysmik
/*      gotoxy(52,3);
          show_int_barwa("OBROTY:   ",obroty,5,1);
        cputs(" obr/min");
        // modul we analog Weidm.
        gotoxy(52,4);
        show_float_barwa1("NAPIECIE: ",napiecie,5,2);
        cputs(" V");
        // modul 4 we Weidm.
        gotoxy(2,6);
        show_float_barwa("Licznik energii cieplnej LC:      ",moc_cieplna,8,0);
        cputs(" MW");
        gotoxy(50,6);
        show_float_barwa("",cieplo,8,0);
        cputs(" GJ");

        gotoxy(2,7);
        show_float_barwa("Licznik energii elektrycznej LE1: ",moc1,8,1);
        cputs(" kW");
        gotoxy(50,7);
        show_float_barwa("",energia1,8,1);
        cputs(" kWh");

        gotoxy(2,8);
        show_float_barwa("Licznik energii elektrycznej LE2: ",moc2,8,0);
        cputs(" kW");
        gotoxy(50,8);
        show_float_barwa("",energia2,8,2);
        cputs(" kWh");

        wcolor(INVERS_FG,INVERS_BG);
        gotoxy(2,11);
        cputs(" STEROWANIE URZADZENIAMI POPRZEZ SIEC LONWORKS ");
        wcolor(EKRAN_FG,EKRAN_BG);
*/
        //modul ALCM1 Ahlstrom
/*  gotoxy(2,13);
   show_float_barwa1("Nastawa natezenia swiecenia lamp:        ",(float)stan_dimmer/2
7,kolory[1]);
        cputs(" %");
        gotoxy(2,14);
   show_int_barwa("Zalaczanie oswietlenia zewnetrznego:     ",stan_zew,3,kolory[2]);
        //modul ALCM2 Ahlstrom
        gotoxy(2,16);
   show_float_barwa1("Nastawa predkosci obrotowej silnika:     ",(float)stan_maly/2.0,,
kolory[3]);
        cputs(" %");
        gotoxy(2,17);
   show_int_barwa("Zalaczanie glownego oswietlenia fabryki: ",stan_fab,3,kolory[4]);
        //modul 8wy Weidm.
        gotoxy(2,19);
*/
/* stan_dimmer, stan_zew, stan_maly, stan_fab  */
/*
   show_int_barwa("Zalaczanie dodatkowego oswietlenia S1:   ",stan_s1,3,kolory[5]);
        gotoxy(2,20);
   show_int_barwa("Zalaczanie dodatkowego oswietlenia S2:   ",stan_s2,3,kolory[6]);
*/
  }

/*-------------------------------------------------*/
/*
     zwraca 1 jesli ESC
```

```
            2 jesli ENTER
            3 jesli Q */

int obsluga_ekranu1(void)
{
char bufor[80];
struct date dzisiaj;
struct time godzina;
int kierunek,i;
int pom_wysw;
int barwa;


      gettime(&godzina);
      getdate(&dzisiaj);


      wcolor(EKRAN_FG,EKRAN_BG);

      gotoxy(8,3);    //data
      cprintf("%02d.%02d.%d",
      (int)dzisiaj.da_day,(int)dzisiaj.da_mon,dzisiaj.da_year);

      gotoxy(29,3);   //godzina
      cprintf("%02d:%02d:%02d",
      (int)godzina.ti_hour,(int)godzina.ti_min,(int)godzina.ti_sec);


      /*      show_float(kan_par_opis[8],kanpar[kan_nr].kan_alarm_przyr,7);
        kierunek=get_int2(&kanpar[kan_nr].kan_konwersja,1,0,4); */


      gotoxy(30,6);
      show_float_barwa1("",zad_te1,5,kolory[1]);
      gotoxy(59,6);
      show_float_barwa1("",te1,5,kolory[0]);
      gotoxy(30,8);
      show_int_barwa("",zad_czas1,5,kolory[2]);
      gotoxy(59,8);
      show_int_barwa("",czas_grz1,5,kolory[7]);
      gotoxy(55,10);
      show_int_barwa("",auto1,3,kolory[3]);

      gotoxy(30,16);
      show_float_barwa1("",zad_te2,5,kolory[4]);
      gotoxy(59,16);  //temperatura
      show_float_barwa1("",te2,5,kolory[0]);
      gotoxy(30,18);
      show_int_barwa("",zad_czas2,5,kolory[5]);
      gotoxy(59,18);
      show_int_barwa("",czas_grz2,5,kolory[8]);
      gotoxy(55,20);
      show_int_barwa("",auto2,3,kolory[6]);
//    show_stan_barwa("",piec2,3,kolory[6]);
/*
      gotoxy(62,3);   //obroty
      show_int_barwa("",obroty,5,1);
      gotoxy(62,4);   //napiecie
      show_float_barwa1("",napiecie,5,2);

      gotoxy(36,6);
      show_float_barwa("",moc_cieplna,8,0);
```

```
        gotoxy(50,6);
        show_float_barwa("",cieplo,8,0);
        gotoxy(36,7);
        show_float_barwa("",moc1,8,1);
        gotoxy(50,7);
        show_float_barwa("",energia1,8,2);
        gotoxy(36,8);
        show_float_barwa("",moc2,8,0);
        gotoxy(50,8);
        show_float_barwa("",energia2,8,0);
*/
/*      gotoxy(43,13);
        show_float_barwa1("",(float)stan_dimmer/2.0,7,kolory[1]);
        gotoxy(43,14);
        show_stan_barwa("",stan_zew,3,kolory[2]);
        gotoxy(43,16);
        show_float_barwa1("",(float)stan_maly/2.0,7,kolory[3]);
        gotoxy(43,17);
        show_stan_barwa("",stan_fab,3,kolory[4]);
*/
/*
        gotoxy(43,19);
        show_stan_barwa("",stan_s1,3,kolory[5]);
        gotoxy(43,20);
        show_stan_barwa("",stan_s2,3,kolory[6]);

        if(AUTO)
          automat();
  */
      // obsluga klawiatury
        if(kbhit()){
         kierunek=get_znak();
/* debug*/          i=1;
        if(kierunek==27) return(1);  //Esc
        if(kierunek==13) return(2);  //Enter
        if(kierunek==113) return(3); //q
/*
        if(kierunek=='a')  { AUTO=1;
        message("      praca automatyczna     wcisniecie klawisza   r - reczna");
        ciemnosci(); nadaj_auto(); }
        if(kierunek=='r') {
         AUTO=0;
        message("      reczne sterowanie      wcisniecie klawisza   a - automat ");
                    }
        if(AUTO==0) {
*/
/*
        if((kierunek==43)||(kierunek==45)) //plus lub minus
        {
         if (kierunek==43)
         {
           znak=1;    //plus lub ON
         }
         else
         {
           znak=0;    //minus lub OFF
         }
         // tutaj funkcja wysylajaca zmienna w siec LonWorks
         nadaj_zmienna(nr_danej,znak);
        }
*/
        if((kierunek==GORA)&&(nr_danej>1)){
```

```
          kolory[nr_danej]=2;
          nr_danej--;
          kolory[nr_danej]=3;
         }
         else
          if((kierunek==DOL)&&(nr_danej<6))
          {
           kolory[nr_danej]=2;
           nr_danej++;
           kolory[nr_danej]=3;
          }

       if(kierunek==8) //<-kasuj
       {
          switch(nr_danej)
          {
          case 1:
           gotoxy(30,6);
           kierunek=get_float03(&zad_te1,5);
           break;
          case 2:
           gotoxy(30,8);
           kierunek=get_int2(&zad_czas1,5,5,60);
           break;
          case 3:
           gotoxy(55,10);
           kierunek=get_int2(&auto1,3,0,1);
           break;
          case 4:
           gotoxy(30,16);
           kierunek=get_float03(&zad_te2,5);
           break;
          case 5:
           gotoxy(30,18);
           kierunek=get_int2(&zad_czas2,5,5,60);
           break;
          case 6:
           gotoxy(55,20);
           kierunek=get_int2(&auto2,3,0,1);
           break;
          } // switch
        } // if <-

   //     } //if AUTO
        } // if kbhit
// if((int)godzina.ti_sec%4==0)
// NV_poll();

        return(0);
  }
/*************************************************************/


  void ekran_sterowan(void)
  {
    char bufor[80];
    int i;
  int polecenie_obslugi;
        CursorOff();
        clrscr();
        bufor[0]='   ';              /*czyszczenie gornej linii*/
        for(i=1;i<68;i++)
```

```
         bufor[i]=(char)196;
        bufor[68]='\0';
        wprint_at(0,2,bufor);
        wprint_at(0,3," STEROWANIE  ");
/*      wprint_at(0,46," Stacja    "); */
        stac_bie=1;
        display_maske1();
        message(ZAKONCZ);
        while(1){
         if(pomiarek(0)==27)
          break;
/*QA*/ /*        if(ilosc_odczytow !=0 )*/
         polecenie_obslugi=obsluga_ekranu1();
         if(polecenie_obslugi==2){
          MINuta=1;
          ZAPis=1;
         }
         if(polecenie_obslugi==1)
          break;
         if(polecenie_obslugi==3) {
          exit_func();
          break;
         }
        }
        CursorNorm();
  }

/*---------------------------*/


void sterowanie(void)
{
 if(auto1 && grzeje1==0) {
  if( il_grzej < MAX_GRZEJ && te1 < zad_te1 ) {
   czas_wl1= oblczas();
   grzeje1=1;
   NV_update(7);
   kolory[7] = 1;
   ++ il_grzej;
  }
 }
 if(auto1 && grzeje1) {
    czas_grz1 = oblczas() - czas_wl1;
    if(czas_grz1<0)    czas_grz1+=3600;
   if(te1 > zad_te1 || czas_grz1 > zad_czas1) {
    grzeje1=0;
    czas_grz1=0;
   NV_update(7);
    kolory[7] = 0;
    -- il_grzej;
   }
 }
 if(auto2 && grzeje2==0) {
  if( il_grzej < MAX_GRZEJ && te2 < zad_te2 ) {
   czas_wl2= oblczas();
   grzeje2=1;
   NV_update(8);
   kolory[8] = 1;
   ++ il_grzej;
  }
 }
```

```
 if(auto2 && grzeje2) {
  czas_grz2 = oblczas() - czas_wl2;
   if(czas_grz2<0)    czas_grz2+=3600;
   if(te2 > zad_te2 || czas_grz2 > zad_czas2) {
    grzeje2=0;
    czas_grz2=0;
   NV_update(8);
    kolory[8] = 0;
    -- il_grzej;
   }
 }

}
/*---------------------------------- ----------*/
int oblczas(void)
{
int czasgrzania;
struct time godzina;
        gettime(&godzina);
        czasgrzania=(int)(godzina.ti_min*60+godzina.ti_sec);

return (czasgrzania);
}


/*---------------------------*/
void zalacz1(void)
{
}
/*---------------------------*/
void wylacz1(void)
{
}
/*---------------------------*/
void zalacz2(void)
{
}
/*---------------------------*/
void wylacz2(void)
{
}
/*************************************************/
void nadaj_zmienna(nr_danej,znak)
int nr_danej, znak;
{
 switch(nr_danej)
          {
          case 1:
          if(znak==0){
           if(stan_dimmer>0)
            nastawa_dimmer=stan_dimmer-1;
          }
          else
          {
          if(stan_dimmer<200)
           nastawa_dimmer=stan_dimmer+1;
          }
           //zamiast tego nizej wyslanie w siec
//   NV_update(0);  /* zmienic w applmsg NV_update(void) */
  stan_dimmer=nastawa_dimmer;
         break;
         case 2:
```

```
          if(znak==0){
           if(stan_zew==1)   // ON zmiana na OFF
           zal_zew=stan_zew-1;
          }
          else
          {
     if(stan_zew==0)   //OFF zmiana na ON
           zal_zew=stan_zew+1;
          }
           //zamiast tego nizej wyslanie w siec
//   NV_update(0);     /* tez dimmer */
  stan_zew=zal_zew;
//printf("poll");
//NV_poll();
         break;
         case 3:
          if(znak==0){
           if(stan_maly>0)
            nastawa_maly=stan_maly-1;
          }
          else
          {
           if(stan_maly<200)
            nastawa_maly=stan_maly+1;
          }
           //zamiast tego nizej wyslanie w siec
//   NV_update(2);  /* maly_out */
  stan_maly=nastawa_maly;
         break;
         case 4:
          if(znak==0){
           if(stan_fab==1)   // ON zmiana na OFF
           zal_fab=stan_fab-1;
          }
          else
          {
           if(stan_fab==0)   //OFF zmiana na ON
           zal_fab=stan_fab+1;
          }
           //zamiast tego nizej wyslanie w siec
//   NV_update(2);  /* tez maly */
  stan_fab=zal_fab;
         break;
/*       case 5:
          if(znak==0){
           if(stan_s1==1)   // ON zmiana na OFF
           zal_s1=stan_s1-1;
          }
          else
          {
           if(stan_s1==0)   //OFF zmiana na ON
           zal_s1=stan_s1+1;
          }
//printf("\n stan_s1 = %d zal= %d ", stan_s1, zal_s1);
//      NV_update(4);
      stan_s1=zal_s1;

         break;
         case 6:
          if(znak==0){
           if(stan_s2==1)   // ON zmiana na OFF
           zal_s2=stan_s2-1;
```

```
            }
            else
            {
             if(stan_s2==0)   //OFF zmiana na ON
             zal_s2=stan_s2+1;
            }
 //   NV_update(5);
              stan_s2=zal_s2;
           break;
*/
           default:
           break;
           }
}
/*---------- koniec USERFUN.C */
```

```
        /* PIAP   stfun3.c
        **
        **
        */
        #include <stdio.h>
        #include <stdlib.h>
        #include <string.h>
        #include <conio.h>
        #include <ctype.h>
        #include <math.h>
        #include <alloc.h>
        #include <dos.h>
        #include "window3.h"
        #include "menue3.h"
        #include "stfun3.h"


/*------------------------------------  prototypy funkcji
        void error(char*);
        void show_float(char*,double,int);
        void show_float_barwa(char*,double,int,int);
        int get_float0(float*,int);
        void show_int(char*,int,int);
        int get_int0(int*,int);
        int get_int1(int*,int);
        void show_string(char*,char*,int);
        int get_string1(char*,int);
        int get_znak(void);
        int video_mode(void);
        void CursorNorm(void);
        void CursorBlock(void);
        void CursorOff(void);

*/
        int pomiarek(int);

/*-----------------------------------------------------------------*/
/* pisze text w oknie high */

  void error(text)
  char* text;
  {
   int spalte;
   spalte = ((80-strlen(text))/2)+1; /* centrowanie okna */
   open_window(10,spalte,3,strlen(text)+1,ERROR_FG, ERROR_BG);
   wprint_at(1,2,text);  /* wskazanie na tekst */
   message("               Nacisnij dowolny klawisz...");
   (void)get_znak();        /* i czekaj */
   close_window();
  }

     /*--------------------------------------*/
     /* pokaz tekst i liczbe float w polu o dlugosci */

        void show_float(tekst,nazwa,dlugosc)
         char*tekst;
         double nazwa;
         int dlugosc;
         {
```

```
        char bufor[80];

        sprintf(bufor,"%-*.3f",dlugosc,nazwa);
        bufor[dlugosc]='\0';
        cputs(tekst);
        wcolor(INVERS_FG,INVERS_BG);
        cprintf("%s",bufor);
        wcolor(EKRAN_FG,EKRAN_BG);
        }
    /*---------------------------------------*/
    /* pokaz tekst i liczbe float w polu o dlugosci
       i kolorze: 0
                  1
                  2
    */

       void show_float_barwa(tekst,nazwa,dlugosc,barwa)
        char*tekst;
        double nazwa;
        int dlugosc;
        int barwa;
        {
        char bufor[80];

        sprintf(bufor,"%-*.3f",dlugosc,nazwa);  /*LON*/
        bufor[dlugosc]='\0';
        cputs(tekst);
        if(barwa==0)
         wcolor(HELP_FG,HELP_BG);
        if(barwa==1)
         wcolor(HIGH_FG,HIGH_BG);
        if(barwa==2)
         wcolor(LOW_FG,LOW_BG);
        cprintf("%s",bufor);
        wcolor(EKRAN_FG,EKRAN_BG);
        }
/*---------------------------------*/
 /* pokaz tekst i liczbe float z jednym miejscem po przecinku
    w polu o dlugosci
        i kolorze: 0
                   1
                   2
                   3 na jasnym tle tak jak show_float
  Dodane dla potrzeb LONDEMO
 */

        void show_float_barwa1(tekst,nazwa,dlugosc,barwa)
         char*tekst;
         double nazwa;
         int dlugosc;
         int barwa;
         {
         char bufor[80];

         sprintf(bufor,"%-*.1f",dlugosc,nazwa);
         bufor[dlugosc]='\0';
         cputs(tekst);
         if(barwa==0)
          wcolor(HELP_FG,HELP_BG);
         if(barwa==1)
          wcolor(HIGH_FG,HIGH_BG);
         if(barwa==2)
```

```
          wcolor(LOW_FG,LOW_BG);
         if(barwa==3)
          wcolor(INVERS_FG,INVERS_BG);
         cprintf("%s",bufor);
         wcolor(EKRAN_FG,EKRAN_BG);
         }


        /*---------------------------------*/
        /* pobierz liczbe float o dlugosci  jesli glupota to liczba=0 */
        /* wez z nazwa oddaj do nazwa       */
        /*przy wywolaniu &nazwa             */
        /*wynik:0-zakonczono ESC
                1-zakonczono Up
                3-zakonczono ENTER           */

        int get_float0(nazwa,dlugosc)
         float *nazwa;
         int dlugosc;
         {
          char bufor[80];
          int wynik,a,b;
          float nazwax;

          a=wherex();
          b=wherey();
           nazwax=*nazwa;
           sprintf(bufor,"%-*.3f",dlugosc,nazwax);    /* zamienic liczbe na string */
           bufor[dlugosc]='\0';
           wynik=get_string1(bufor,dlugosc);

            nazwax=(float)atof(bufor);    /* zamienic string na liczbe */
              *nazwa=nazwax;
              gotoxy(a,b);       /*likwidacja nie przyjetych znakow */
              wcolor(INVERS_FG,INVERS_BG);
              sprintf(bufor,"%-*.3f",dlugosc,nazwax);
              bufor[dlugosc]='\0';
              cprintf("%s",bufor);
              wcolor(EKRAN_FG,EKRAN_BG);
          return(wynik);
         }


/*---------------------------------*/
/* pokaz tekst i liczbe int w polu o dlugosci */

        void show_int(tekst,nazwa,dlugosc)
         char*tekst;
         int nazwa;
         int dlugosc;
         {
         cputs(tekst);
         wcolor(INVERS_FG,INVERS_BG);
         cprintf("%-*d",dlugosc,nazwa);
         wcolor(EKRAN_FG,EKRAN_BG);
         }
/*---------------------------------*/
/* pokaz tekst i liczbe int w polu o dlugosci
        i barwie : 0
                   1
                   2
                   3 na jasnym tle tak jak show_float
```

```
   Dodane dla potrzeb LONDEMO
*/

        void show_int_barwa(tekst,nazwa,dlugosc,barwa)
         char*tekst;
         int nazwa;
         int dlugosc;
         int barwa;
         {
         cputs(tekst);
         if(barwa==0)
          wcolor(HELP_FG,HELP_BG);
         if(barwa==1)
          wcolor(HIGH_FG,HIGH_BG);
         if(barwa==2)
          wcolor(LOW_FG,LOW_BG);
         if(barwa==3)
          wcolor(INVERS_FG,INVERS_BG);
         cprintf("%-*d",dlugosc,nazwa);
         wcolor(EKRAN_FG,EKRAN_BG);
         }

/*---------------------------------*/
/* pokaz tekst i liczbe int w polu o dlugosci
        jesli >0 to ON
        jesli 0 to OFF
        i barwie : 0
                   1
                   2
                   3 na jasnym tle tak jak show_float
   Dodane dla potrzeb LONDEMO
*/

        void show_stan_barwa(tekst,nazwa,dlugosc,barwa)
         char*tekst;
         int nazwa;
         int dlugosc;
         int barwa;
         {
         cputs(tekst);
         if(barwa==0)
          wcolor(HELP_FG,HELP_BG);
         if(barwa==1)
          wcolor(HIGH_FG,HIGH_BG);
         if(barwa==2)
          wcolor(LOW_FG,LOW_BG);
         if(barwa==3)
          wcolor(INVERS_FG,INVERS_BG);
          if(nazwa==0)
           cprintf("%-*s",dlugosc,"OFF");
          else
           cprintf("%-*s",dlugosc,"ON");
         wcolor(EKRAN_FG,EKRAN_BG);
         }


/*---------------------------------*/
/* pobierz liczbe int o dlugosci       z zerem   */
/* jesli glupota to zero             */
/* przy wywolaniu &nazwa             */
/* wez z nazwa oddaj do nazwa       */
/*        wynik:0-zakonczono ESC
```

```
        1-zakonczono Up
        2-zakonczono Down
        3-zakonczono ENTER          */

int get_int0(nazwa,dlugosc)
 int *nazwa;
 int dlugosc;
 {
  char bufor[80];
  int wynik,a,b;
  int nazwax;

  a=wherex();
  b=wherey();
   nazwax=*nazwa;
   sprintf(bufor,"%d",nazwax);
   wynik=get_string1(bufor,dlugosc);

   nazwax=atoi(bufor);
      *nazwa=nazwax;
      gotoxy(a,b);
      wcolor(INVERS_FG,INVERS_BG);
      cprintf("%-*d",dlugosc,nazwax);
      wcolor(EKRAN_FG,EKRAN_BG);
  return(wynik);
 }


/*---------------------------------*/
/* pobierz liczbe int o dlugosci   tylko > 0 */
/*przy wywolaniu &nazwa             */
/* wez z nazwa oddaj do nazwa       */
/*wynik:0-zakonczono ESC
        1-zakonczono Up
        2-zakonczono Down
        3-zakonczono ENTER           */

int get_int1(nazwa,dlugosc)
 int *nazwa;
 int dlugosc;
 {
  char bufor[80];
  int wynik,a,b,gotowe=1;
  int nazwax;

  a=wherex();
  b=wherey();
  while(gotowe)
  {
   gotoxy(a,b);
   nazwax=*nazwa;
   sprintf(bufor,"%d",nazwax);
   wynik=get_string1(bufor,dlugosc);

    if((nazwax=atoi(bufor))>0)
     {
      *nazwa=nazwax;
      gotoxy(a,b);
      wcolor(INVERS_FG,INVERS_BG);
      cprintf("%-*d",dlugosc,nazwax);
      wcolor(EKRAN_FG,EKRAN_BG);
      gotowe=0;
```

```
            }
          }
          return(wynik);
         }

/*-----------------------------------*/
/* pokaz tekst i string w polu o dlugosci */

        void show_string(tekst,nazwa,dlugosc)
         char*tekst;
         char*nazwa;
         int dlugosc;
         {
         cputs(tekst);
         wcolor(INVERS_FG,INVERS_BG);
         cprintf("%-*s",dlugosc,nazwa);
         wcolor(EKRAN_FG,EKRAN_BG);
         }
/*-----------------------------------*/
/* modyfikacje do profi JK 22.10.92  PgUp PgDn*/
/* pobierz string o dlugosci */
/* wez z nazwa oddaj do nazwa       */
/* wynik:0-zakonczono ESC
                1-zakonczono Up
                2-zakonczono Down
                3-zakonczono ENTER
                4-zakonczono PgUp
                5-zakonczono PgDn          */

        int get_string1(nazwa,dlugosc)
         char*nazwa;
         int dlugosc;

         {
         int i,a,b,position,znak,wynik;
         char bufor[80];
         int dalej=1;

          a=wherex();                    /*zapamietaj pozycje w oknie*/
          b=wherey();
          sprintf(bufor,"%s",nazwa);
          position=strlen(bufor);        /*pozycja w buforze*/
          wcolor(INVERS_FG,INVERS_BG);
          for(i=0;i<dlugosc;i++)         /*czysc linie*/
           putch(' ');
          gotoxy(a,b);

          cputs(bufor);                  /* wpisz string */
          a=wherex();
          /* wycofanie do ostatniej nie-spacji */
          while((position>0)&&(bufor[position-1]==' '))
             {
              position--;
              a--;
             }
          bufor[position]='\0';
          gotoxy(a,b);

          while(dalej)                   /*wyjscie ESC,Up,Enter */
          {
           znak=get_znak();
           switch(znak)
```

```
            {
             case 377:    /*right*/
              break;
             case 13:  /*  \r' */
             case 380:    /*down*/
             case 372:  /* Up */
             case 27:   /* ESC */
             case 373:   /*PgUp*/
             case 381:   /*PgDn */
              sprintf(nazwa,"%s",bufor);
              dalej=0;
              wynik=3;
              if(znak==380)
              wynik=2;
              if(znak==27)
              wynik=0;
              if(znak==372)
              wynik=1;
              if(znak==373)
              wynik=4;
              if(znak==381)
              wynik=5;

              break;
             case 8:     /* '\b' */
             case 375:  /* Left */
              if(position>0)
              {
               position--;
               bufor[position]='\0';
               gotoxy(a-1,b);
               putch(' ');
               a--;
               gotoxy(a,b);
              }
              break;
             default:                        /* dopisz znaki na koncu */
              if(position<dlugosc)
              {
              bufor[position++]=(char)znak;
              bufor[position]='\0';
              putch((char)znak);
              a=wherex();
              }
             break;
            }
           }
            wcolor(EKRAN_FG,EKRAN_BG);
            return(wynik);
          }
/*--------------------------------*/
/* modyfikacje do scanlab 04.93 JK   Left Right */
/* modyfikacje do profi JK 22.10.92  PgUp PgDn*/
/* pobierz string o dlugosci */
/* wez z nazwa oddaj do nazwa       */
/* wynik:0-zakonczono ESC
                1-zakonczono Up
                2-zakonczono Down
                3-zakonczono ENTER
                4-zakonczono PgUp
                5-zakonczono PgDn
                6- zakonczono Left
```

```
        7- zakonczono Right
           */

int get_string2(nazwa,dlugosc)
 char*nazwa;
 int dlugosc;

 {
 int i,a,b,position,znak,wynik;
 char bufor[80];
 int dalej=1;

  a=wherex();                    /*zapamietaj pozycje w oknie*/
  b=wherey();
  sprintf(bufor,"%s",nazwa);
  position=strlen(bufor);        /*pozycja w buforze*/
  wcolor(INVERS_FG,INVERS_BG);
  for(i=0;i<dlugosc;i++)         /*czysc linie*/
   putch(' ');
  gotoxy(a,b);

  cputs(bufor);                  /* wpisz string */
  a=wherex();
  /* wycofanie do ostatniej nie-spacji */
  while((position>0)&&(bufor[position-1]==' '))
     {
      position--;
      a--;
     }
  bufor[position]='\0';
  gotoxy(a,b);

  while(dalej)                  /*wyjscie ESC,Up,Enter */
  {
   znak=get_znak();
   switch(znak)
   {
    case 13:  /* '\r' */
    case 380:    /*down*/
    case 372:  /* Up */
    case 27:   /* ESC */
    case 373:   /*PgUp*/
    case 381:   /*PgDn */
    case 375:  /* Left */
    case 377:    /*right*/
     sprintf(nazwa,"%s",bufor);
     dalej=0;
     wynik=3;
     if(znak==380)
     wynik=2;
     if(znak==27)
     wynik=0;
     if(znak==372)
     wynik=1;
     if(znak==373)
     wynik=4;
     if(znak==381)
     wynik=5;
     if(znak==375)
     wynik=6;
     if(znak==377)
     wynik=7;
```

```
            break;
           case 8:     /* '\b' */

            if(position>0)
            {
             position--;
             bufor[position]='\0';
             gotoxy(a-1,b);
             putch(' ');
             a--;
             gotoxy(a,b);
            }
            break;
           default:                        /* dopisz znaki na koncu */
            if(position<dlugosc)
            {
            bufor[position++]=(char)znak;
            bufor[position]='\0';
            putch((char)znak);
            a=wherex();
            }
            break;
          }
         }
          wcolor(EKRAN_FG,EKRAN_BG);
          return(wynik);
        }
/*-----------------------------------------*/
/* modyfikacje do profi JK 22.10.92  PgUp PgDn*/
/* przyjmuje znaki do pisania oraz Backspace,Esc,Enter */
/* zwraca kod znaku lub kod rozszerzony+300 jesli strzalki */

   int get_znak()
   {
   int a;
   int done=1;
    while(done)
    {
     while(1){
      pomiarek(0);
      if(kbhit())
       break;
     }
     a=getch();
     if(a==0)                /*dla znakow o rozszerzonym kodzie*/
      {
       a=getch();
       switch(a){
       case 72:       /* Up */
       case 80:       /* Down */
       case 75:       /* Left */
       case 77:       /* Right */
       case 73:          /* PgUp */
       case 81:          /* PgDn */
        done=0;
        a+=300;
        break;
       default:
        break;
       }
      }
```

```
     else
      switch(a){
       case 134:      /* polskie znaki - kody Mazovii klawpoly/5  */
       case 141:
       case 143:
       case 144:
       case 145:
       case 146:
       case 149:
       case 152:
       case 156:
       case 158:
       case 161:
       case 162:
       case 163:
       case 164:
       case 165:
       case 166:
       case 167:
        done=0;
        break;
       default:
        if(isprint(a)||(a==8)||(a==27)||(a==13))
         done=0;
        break;
       }

    }
    return(a);
   }
 /*-------------------------------------------------------*/
/*-------------------------------------------------------------*/
/* GET_KEY - odczytanie wcisnietego klawisza  */
int get_key()
{
    int taste;
    while(1){
     pomiarek(0);
     if(kbhit())
      break;
    }
    taste = getch();
    if (taste == 0) {
        taste = getch();
        taste |= 0x100;
    }
    return(taste);
 }

  /*-------------------------------------------------------*/


  /*-------------------------------------------------------*/
/* koniec programu stfun3.c */
```