

**DOKUMENT WZORCOWY**

440

**OŚRODEK POMIARÓW RUCHU I CZASU**

(Nazwa ONB/ZNB)

BE10

**Główny wykonawca**  
**Wykonawcy**mgr inż. Rafał Więcko  
mgr inż. Rafał Więcko  
inż. Jerzy Niewiatowski„Opracowanie modułów pomiarowych dotykowego pomiaru grubości  
do systemu pomiarowego o architekturze rozproszonej”

praca jednoetapowa

(Tytuł pracy, numer i tytuł etapu)

**Zleceniodawca**

PIAP

Główny Wykonawca



mgr inż. Rafał Więcko

Z-ca Dyrektora d/s  
Badawczo Rozwojowych

dr inż. Jan Jabłkowski

Kierownik ORC



mgr inż. A. Cybulski

Pracę zakończono dnia ..10.02.98.....

Nr arch. 7540

Nr zlecenia S1766

## **Abstrakt**

## **Tytuły poprzednich sprawozdań**

## **Rozdzielnik**

Egz.1. OIN

Egz.2. ORC

Egz.3.

## Spis treści

1. Sprawy formalne
  - 1.1. Cel pracy
  - 1.2. Podstawa wykonania pracy
2. Wstęp
3. Moduł przetwornika do współpracy z indukcyjnym czujnikiem pomiaru przemieszczeń liniowych
4. Miernik tablicowy NGS96F1/ADC35
5. Badania modelu
6. Wnioski końcowe
7. Wykaz załączników

## **1. Sprawy formalne**

### **1.1. Cel pracy**

Celem pracy było sprawdzenie możliwości wykorzystania specjalizowanego układu scalonego do współpracy z czujnikami indukcyjnymi przemieszczeń liniowych i w przypadku pozytywnych wyników tej próby, opracowanie konstrukcji modelu miernika do systemu pomiarowego o architekturze rozproszonej.

### **1.2. Podstawa wykonania pracy**

Podstawą wykonania pracy jest zlecenie statutowe S1766 p.t. „Opracowanie modułów pomiarowych dotykowego pomiaru grubości do systemu pomiarowego o architekturze rozproszonej”

## **2. Wstęp**

W związku z zapytaniami ofertowymi o pomiar grubości folii pokrytych warstwą półprzezroczystą, transportowanych w procesie produkcji, nie mogąc zastosować bezdotkowych głowic pomiarowych, w ORC podjęto prace wyprzedzające zawarcie umów na wykonanie stanowisk pomiarowych, których celem było określenie przydatności dostępnych w handlu indukcyjnych czujników dotykowych (LVDT - Linear Variable Differential Transducer) i możliwości wykonania miernika współpracującego z tymi czujnikami. Z uwagi na podobieństwo pomiaru grubości folii i zrealizowanego uprzednio stanowiska bezdotkowego pomiaru grubości płyt wiórowych, zdecydowano się zachować rozproszoną architekturę stanowiska pomiarowego. Umożliwiało to zarówno wykorzystanie doświadczeń zdobytych przy realizacji pomiaru bezdotkowego, jak również opracowanych wówczas urządzeń (zespół obecności płyty i wyznaczania interwału skanowania, sterownik sieci głowic pomiarowych, oprogramowanie terminala wizualizacji pomiaru). Co więcej wykorzystane mogły być istotne partie oprogramowania głowic pomiarowych, zwłaszcza komunikacyjne. Przedmiotem sprawdzenia była możliwość wykorzystania gotowego układu scalonego AD698 firmy Analog Devices (USA), zapewniającego zintegrowaną obsługę specyficznego czujnika, jakim jest indukcyjny (transformatorowy) czujnik pomiaru przemieszczeń liniowych. Realizacja zadania w sposób dyskretny mijała się z celem. Z uwagi na swoje przeznaczenie (element systemu rozproszonego), za standard pakietów przyjęto płytki formatu pojedynczej eurokarty, moduł zaś pozbawiony miał być lokalnego wyświetlacza wyników pomiarów.

Niestety, już w trakcie realizacji tego zlecenia potencjalni kontrahenci wycofali się z rozmów na temat opracowania stanowisk pomiarowych, ze względu na własne trudności finansowe. Dalsza realizacja zadania w pierwotnej postaci miałyby jedynie znaczenie formalne. W tej sytuacji zrezygnowano z koncepcji wykonania modułu pomiarowego dostosowanego konstrukcyjnie do rozproszonej architektury stanowiska i sposobu mocowania do jego ramy nośnej, podejmując decyzję o wykonaniu modułu pomiarowego, jako miernika tablicowego w standardowej obudowie NGS96x96. Do momentu zmiany koncepcji wykonano schematy ideowe pakietu przetwornika AD698/ADS7809 wraz z wstępnym rozmieszczeniem

elementów na płycie drukowanej, schematy pakietu jednostki centralnej 68HC05 i schematy modułu zasilacza 220Vac (Załącznik Nr. 2). Z punktu widzenia istoty zadania (sprawdzenie parametrów układu AD698) nie miało to większego znaczenia. Wykonanie w postaci standardowego modułu tablicowego dawało potencjalne możliwości sprzedaży osobnych urządzeń, poza systemem, pozwalając również na ocenę rozwiązań konstrukcyjnych w obudowie tablicowej, istotnie odmiennych o stosowanych przy wykonaniu pakietów standardu pojedynczej eurokarty (160x100mm). Prace własne nad konstrukcją mierników tablicowych (96x96mm) prowadzone są w ORC od kilku już lat, tym niemniej nie nadano im dotychczas charakteru formalnego, mimo że w ocenie zespołu konstruktorów pożądane byłoby wprowadzenie tego typu urządzeń do produkcji. Opracowany model miernika tablicowego umożliwia wykonanie pomiarów m.in. wartości napięcia, prądu (4-20mA), szybkości obrotowych. Dlatego też zdecydowano o jego wykorzystaniu do sprawdzenia parametrów użytkowych przetwornika AD698 przyjmując, że połączenia elektryczne zostaną wykonane na uniwersalnej płycie drukowanej, a sam przetwornik znajdzie się poza obudową podstawowego przyrządu.

### **3. Moduł przetwornika do współpracy z indukcyjnym czujnikiem pomiaru przemieszczeń liniowych**

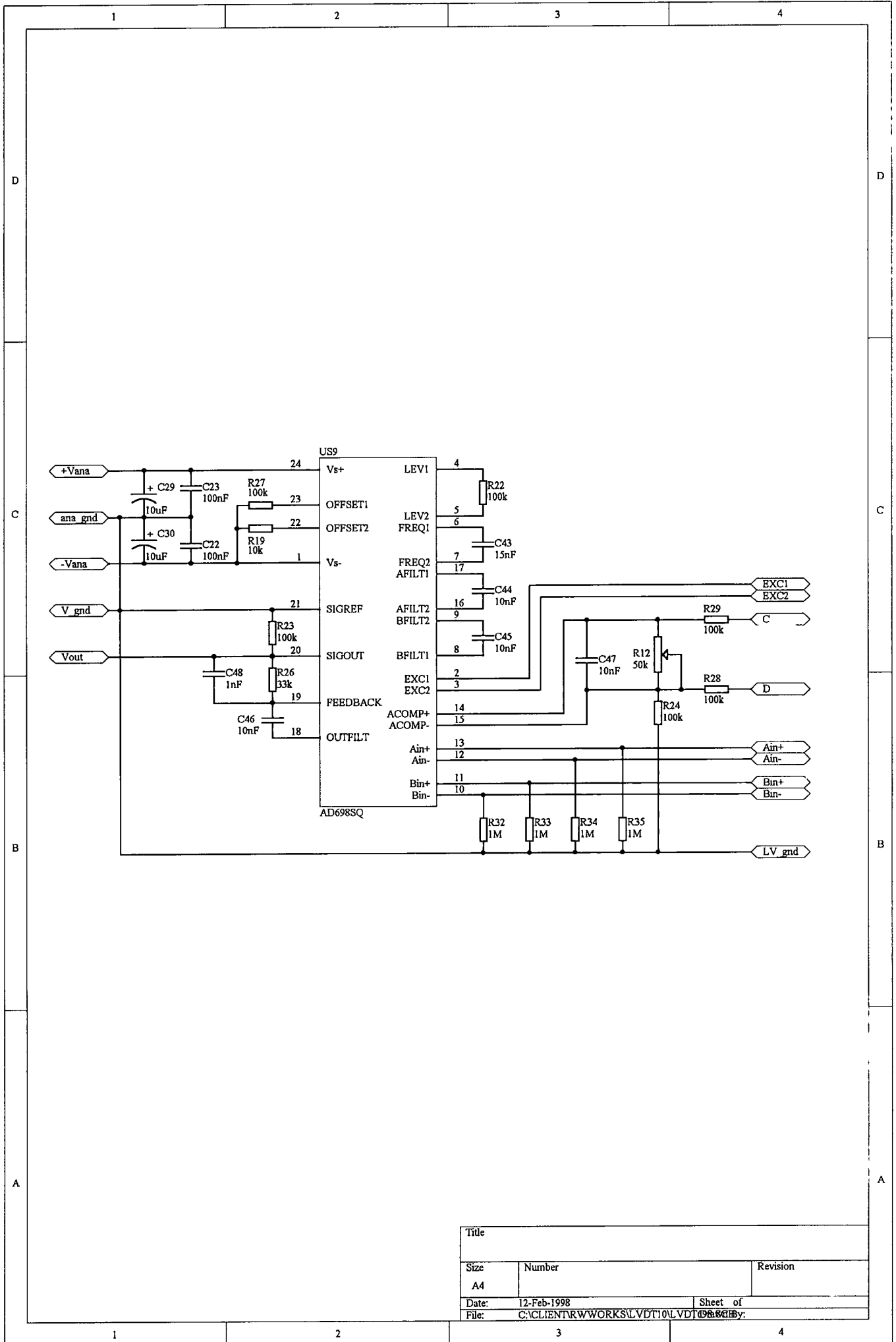
Schemat ideowy układu interfejsu czujnika indukcyjnego przedstawiono na rys.1. Do jego konstrukcji wykorzystano specjalizowany układ scalony AD698 firmy Analog Devices, pracujący w układzie mostkowym. Ten rodzaj pracy odpowiada wykorzystywanemu czujnikowi CL-70, produkcji firmy ZEP-WN. Konwerter AD698 generuje sygnały sterujące czujnika, zapewniając jednocześnie przekształcenie sygnałów wejściowych (A, B) na wartość napięcia wyjściowego (+/-10V), proporcjonalną do położenia rdzenia czujnika indukcyjnego. Przetwornik zasilany jest napięciem symetrycznym (+/-), doprowadzonym z zewnątrz. Układ wykonano na płycie uniwersalnej i zamknięto w plastikowej obudowie. Docelowo znajdzie się on wewnątrz modułu miernika tablicowego, na pakiecie przetwornika analogowo-cyfrowego.

### **4. Miernik tablicowy NGS96F1/ADC35**

Miernikowi tablicowemu nadano charakter modułowy. Składa się on z pięciu pakietów:

- jednostki centralnej 68HC11F1 (MPU-F1-10),
- przetwornika a/d 4.5 cyfry - ICL7135 i d/a 12 bitów - MAX543 (ANA-F1-10),
- zasilacza 220Vac (PWR-F1-10),
- wyświetlacza LCD 5x7 seg z klawiaturą 6-cio przyciskową (LCD-F1-10), albo
- 8-przyciskowej klawiatury funkcyjnej (KBD-F1-10), z odrębnym wyświetlaczem alfanumerycznym LCD 2x16 znaków.

Szczegółowy opis i schematy modułów zamieszczono w załączniku Nr.3. Pakiet przetwornika a/d ANA-F1-10 skonfigurowano do jednokanałowego pomiaru napięcia w zakresie +/-10V. Zainstalowane oprogramowanie adc35-10.src prezentuje na wyświetlaczu wartość średnią kolejnych ośmiu pomiarów. Wydruk oprogramowania aplikacyjnego zamieszczono w załączniku Nr.4.



Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\LVDT10\LVDT10.dwg	9 of 10

6

## 5. Badania modelu

W ramach pracy wykonano trzy rodzaje badań:

- sprawdzenie charakterystyki napięcia wyjściowego modułu konwertera AD698 w funkcji przemieszczenia liniowego ( $\pm 5\text{mm}$ ),
- sprawdzenie charakterystyki wskazań miernika NGS96F1/ADC35 w funkcji napięcia wejściowego ( $\pm 10\text{V}$ ),
- sprawdzenie charakterystyki wskazań miernika NGS96F1/ADC35 w funkcji przemieszczenia liniowego ( $\pm 5\text{mm}$ ).

Do sprawdzenia charakterystyki konwertera w funkcji przemieszczeń liniowych wykorzystano stolik do precyzyjnego zadawania przemieszczeń z panelem pomiarowym LH31A (Sony) i miernik cyfrowy Protek 506. Do sprawdzenia charakterystyki napięciowej miernika tablicowego wykorzystano regulowane źródło napięcia (0...+10V) i miernik cyfrowy Protek 506. Do sprawdzenia charakterystyki wskazań miernika tablicowego w funkcji przemieszczeń liniowych (cały tor pomiarowy - czujnik, moduł konwertera i miernik) wykorzystano stolik do precyzyjnego zadawania przemieszczeń z panelem pomiarowym LH31A (Sony) i miernik cyfrowy Protek 506. Wyniki badań modelu (Załącznik Nr.1) wskazują na liniowość toru pomiarowego. Szczególnie istotna jest stabilność wyników. Daje ona podstawy by oczekiwać, że realne jest uzyskanie zdecydowanie większej rozdzielczości od zakładanej, bliskiej teoretycznej (0.275  $\mu\text{m/imp.}$ ), a tym samym zwiększenie dokładności przyrządu do 0.005, a nawet 0.001 mm, z tym, że wymaga to przeprowadzenia dalszych badań, w tym temperaturowych, stabilności długoczasowej i dynamicznych.

## 6. Wnioski końcowe

Przeprowadzone próby dowodzą, że możliwe jest wykorzystanie układu scalonego AD698 do budowy własnych przyrządów do dotykowych pomiarów przemieszczeń liniowych za pomocą standardowych czujników. Zastrzeżenia budzi jednak konstrukcja mechaniczna zastosowanego czujnika produkcji krajowej. W momencie realizacji pracy był to jedyny czujnik, który można było zakupić osobno, bez urządzeń odczytowych. Jego cena jest porównywalna z cenami czujników produkcji zachodniej. W chwili obecnej sytuacja ta uległa diametralnej zmianie i przedstawicielstwa firm zachodnich (np. Sony) oferują również same czujniki, których konstrukcja mechaniczna nie budzi zastrzeżeń, a co więcej dostępna jest cała gama różnych rozwiązań styku z elementem mierzonym, m.in. rolki.

Przyjęte rozwiązanie (specjalizowany układ scalony, przetwornik integracyjny 4.5 cyfry) spełnia założenia (dokładność 0.01mm, dla zakresu 10mm) i umożliwia uzyskanie nawet lepszych parametrów (rozdzielczość ok. 1 $\mu\text{m}$ ) niż początkowo zakładano. W przypadku kontynuacji prac należy rozważyć możliwość wykorzystania innego, szybszego przetwornika integracyjnego, w pełni 16-bitowego (np. TSC850, TSC500A, Teledyne), albo ADS7809 (Burr-Brown), umożliwiających uzyskanie lepszych parametrów użytkowych (głównie czasu odpowiedzi).

Wykonanie miernika w postaci przyrządu tablicowego umożliwia sprzedaż zestawów do pomiaru przemieszczeń liniowych (jedno, albo dwukanałowych)

niezależnie od kompletnego stanowiska do pomiaru grubości. Jednocześnie zachowana została pełna kompatybilność z istniejącym systemem pomiarowym o architekturze rozproszonej - zależnie od zainstalowanego oprogramowania miernik może współpracować z sterownikiem sieci głowic bezdotykowego stanowiska pomiaru grubości. Ponadto uważamy za celowe kontynuowanie opracowania i uruchomienia produkcji własnej rodziny specjalizowanych przyrządów tablicowych (mierniki ciśnienia, prądu i napięcia, obrotomierze, regulatory), czemu sprzyja modularność zrealizowanego rozwiązania. Należy jednak pamiętać, że obecne wymagania odbiorców zdecydowanie wzrosły, zmianie uległa również oferta producentów. Obok rozwiązań standardowych z wyświetlaczami typu LED oferowane są moduły z graficznymi ekranami LCD (również kolorowymi), które wyznaczają nowy standard w konstrukcji tego typu urządzeń. W ORC przeprowadzono prace pilotażowe umożliwiające ocenę realności takiego rozwiązania, zakończone wynikiem pozytywnym. Do prób wykorzystano ekran monochromatyczny LCD typu LM238 (Hitachi). W przypadku przyrządów tablicowych konieczne jest jednak zastosowanie ekranu o mniejszych rozmiarach, odpowiadającego wymiarom obudowy.



## Wykaz załączników

1. Wyniki badań modelu
2. Dokumentacja modelu miernika do systemu rozproszonego
3. Dokumentacja modelu miernika tablicowego
4. Oprogramowanie modelu miernika tablicowego

**S1766**

„Opracowanie modułów pomiarowych dotykowego pomiaru grubości do systemu pomiarowego o architekturze rozproszonej”

**ZAŁĄCZNIK Nr 1.**

Wyniki badań modelu

**Tab.1. Tabela sprawdzenia charakterystyki napięciowej modułu konwertera AD698 w funkcji przemieszczeń liniowych (+/-5mm)**

L.p.	przemieszczenie [mm]	Uwy [V]
1.	5.5	9.77
2.	5.0	8.89
3.	4.5	8.00
4.	4.0	7.11
5.	3.5	6.22
6.	3.0	5.34
7.	2.5	4.44
8.	2.0	3.56
9.	1.5	2.67
10.	1.0	1.78
11.	0.5	0.89
12.	0.0	0.00
13.	-0.5	-0.89
14.	-1.0	-1.78
15.	-1.5	-2.66
16.	-2.0	-3.55
17.	-2.5	-4.44
18.	-3.0	-5.32
19.	-3.5	-6.21
20.	-4.0	-7.09
21.	-4.5	-7.98
22.	-5.0	-8.87
23.	-5.5	-9.75

SLP = 1.76  
ITC = 0.16  
COR = 0.99244

M

Tab.2. Tabela sprawdzenia charakterystyki napięciowej modułu konwertera AD698 w funkcji przemieszczeń liniowych (+/-1.5mm)

L.p.	przemieszczenie [mm]	Uwy [V]
1.	1.5	2.671
2.	1.4	2.493
3.	1.3	2.313
4.	1.2	2.132
5.	1.1	1.959
6.	1.0	1.783
7.	0.9	1.602
8.	0.8	1.426
9.	0.7	1.245
10.	0.6	1.068
11.	0.5	0.890
12.	0.4	0.712
13.	0.3	0.535
14.	0.2	0.356
15.	0.1	0.178
16.	0.0	0.001
17.	-0.1	-0.178
18.	-0.2	-0.356
19.	-0.3	-0.534
20.	-0.4	-0.711
21.	-0.5	-0.891
22.	-0.6	-1.068
23.	-0.7	-1.246
24.	-0.8	-1.424
25.	-0.9	-1.601
26.	-1.0	-1.711
27.	-1.1	-1.956
28.	-1.2	-2.122
29.	-1.3	-2.301
30.	-1.4	-2.483
31.	-1.5	-2.664

SLP = 1.78  
ITC = 0.04  
COR = 0.99997

**Tab.3. Tabela sprawdzenia charakterystyki napięciowej miernika tablicowego NGS96ADC35 - napięcia wejściowe dodatnie**

L.p.	Uwe(nom.) [V]	Uwe(rz.) [V]	odczyt [l.imp.]
1.	0	0.022	+78
2.	1	1.023	+2079
3.	2	2.02	+4070
4.	3	3.01	+6072
5.	4	3.97	+7999
6.	5	4.97	+9998
7.	6	5.97	+12002
8.	7	6.96	+13981
9.	8	7.95	+15944
10.	9	8.94	+17923
11.	10	9.92	+19878

**Tab.4. Tabela sprawdzenia charakterystyki napięciowej miernika tablicowego NGS96ADC35 - napięcia wejściowe ujemne**

L.p.	Uwe(nom.) [V]	Uwe(rz.) [V]	odczyt [l.imp.]
1.	0	-0.039	-112
2.	-1	-1.049	-2130
3.	-2	-2.03	-4122
4.	-3	-3.03	-6117
5.	-4	-4.01	-8070
6.	-5	-4.99	-10051
7.	-6	-5.99	-12041
8.	-7	-6.99	-14032
9.	-8	-7.98	-15994
10.	-9	-8.97	-17956
11.	-10	-9.94	-19867

SLP = 2006.01  
 ITC = 2.57  
 COR = 0.99999

Tab.5. Tabela sprawdzenia charakterystyki toru pomiarowego (czujnik, konwerter, miernik) w funkcji przemieszczeń liniowych (+/-5mm)

L.p.	przemieszczenie [mm]	odczyt [l.imp]
1.	5.5	19531
2.	5.0	17786
3.	4.5	16041
4.	4.0	14271
5.	3.5	12495
6.	3.0	10721
7.	2.5	8940
8.	2.0	7162
9.	1.5	5381
10.	1.0	3600
11.	0.5	1810
12.	0.0	-35
13.	-0.5	-1812
14.	-1.0	-3590
15.	-1.5	-5372
16.	-2.0	-7150
17.	-2.5	-8929
18.	-3.0	-10703
19.	-3.5	-12473
20.	-4.0	-14230
21.	-4.5	-15992
22.	-5.0	-17746
23.	-5.5	-19499

SLP = 3559.54  
ITC = 9  
COR = 0.99999

**Tab.6. Tabela sprawdzenia charakterystyki toru pomiarowego (czujnik, konwerter, miernik) w funkcji przemieszczeń liniowych (+/-1.5mm)**

L.p.	przemieszczenie [mm]	odczyt [l.imp]
1.	1.5	5381
2.	1.4	5025
3.	1.3	4667
4.	1.2	4310
5.	1.1	3953
6.	1.0	3600
7.	0.9	3237
8.	0.8	2885
9.	0.7	2523
10.	0.6	2167
11.	0.5	1810
12.	0.4	1454
13.	0.3	1097
14.	0.2	743
15.	0.1	386
16.	0.0	-35
17.	-0.1	-388
18.	-0.2	-744
19.	-0.3	-1099
20.	-0.4	-1454
21.	-0.5	-1812
22.	-0.6	-2166
23.	-0.7	-2524
24.	-0.8	-2880
25.	-0.9	-3235
26.	-1.0	-3590
27.	-1.1	-3947
28.	-1.2	-4302
29.	-1.3	-4659
30.	-1.4	-5015
31.	-1.5	-5372

SLP = 3614.58

ITC = 22.39

COR = 0.9923

**S1766**

„Opracowanie modułów pomiarowych dotykowego pomiaru grubości do systemu pomiarowego o architekturze rozproszonej”

**ZAŁĄCZNIK Nr 2.**

Dokumentacja modelu miernika do systemu o architekturze rozproszonej



1

2

3

4

D

D

# lvdt-10

C

C

## LVDT/ADC16 unit

B

B

A

A

Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RWWORKS\LVDT10\LVDT10.DWG	

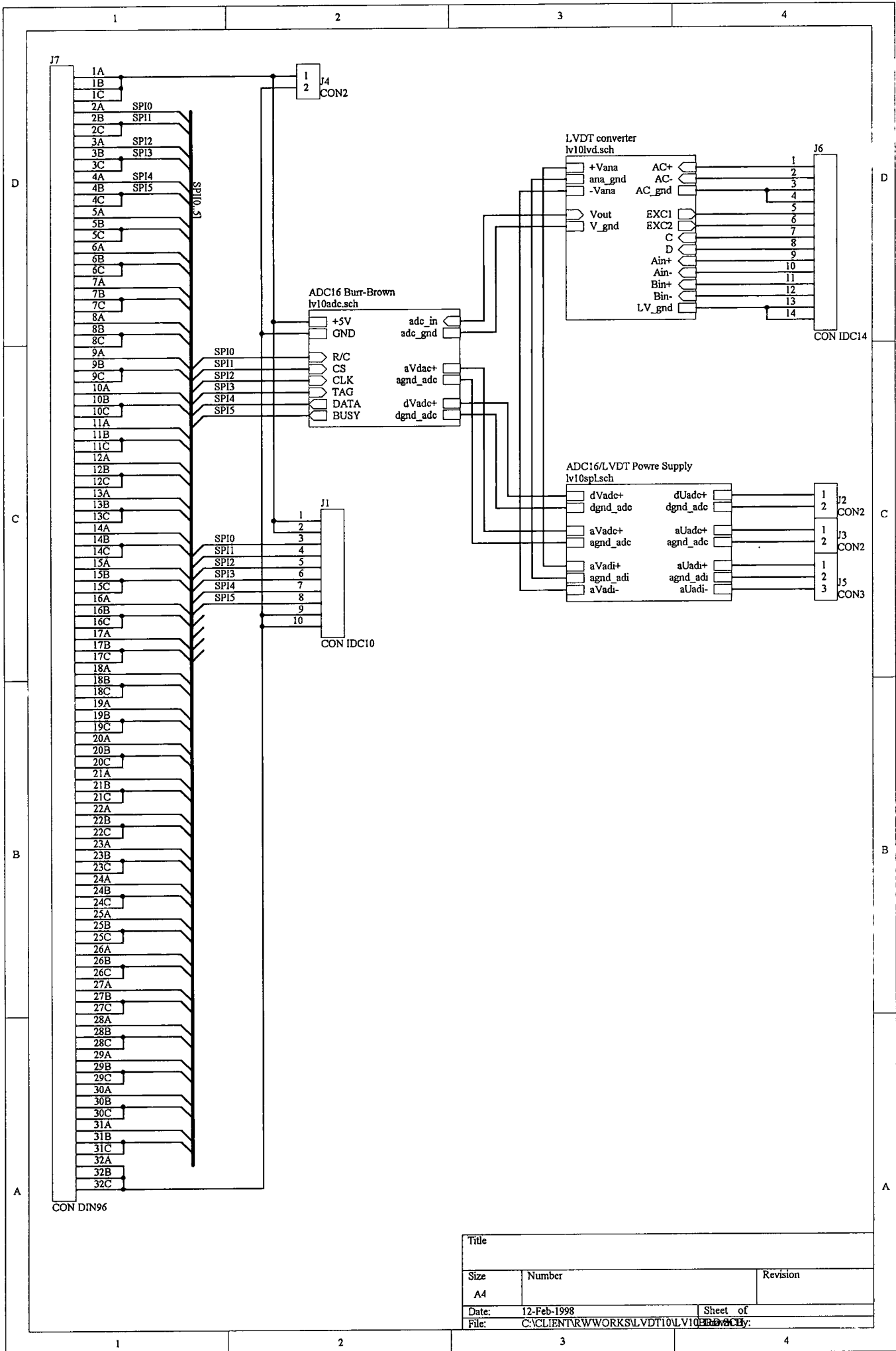
1

2

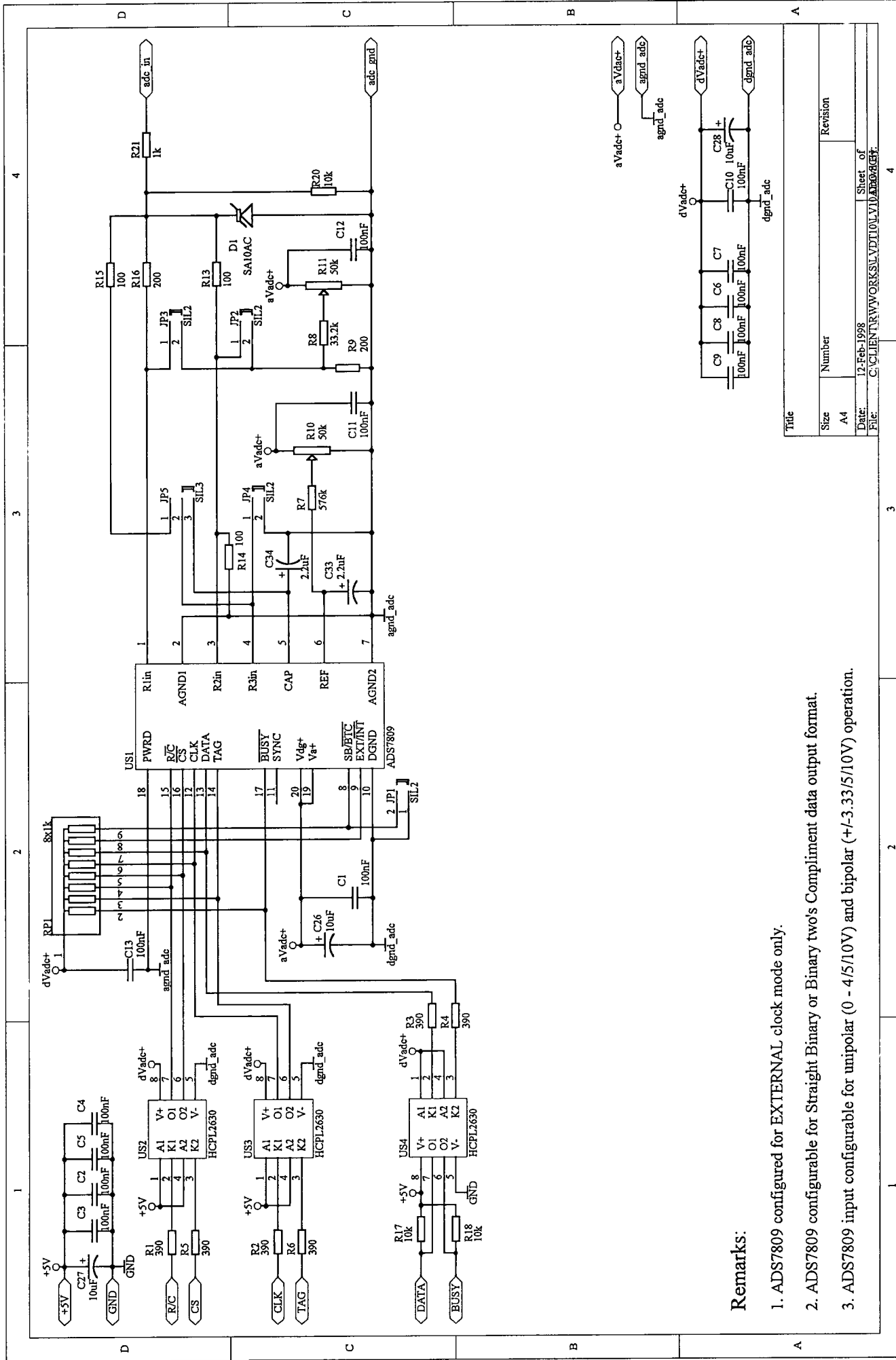
3

4

17



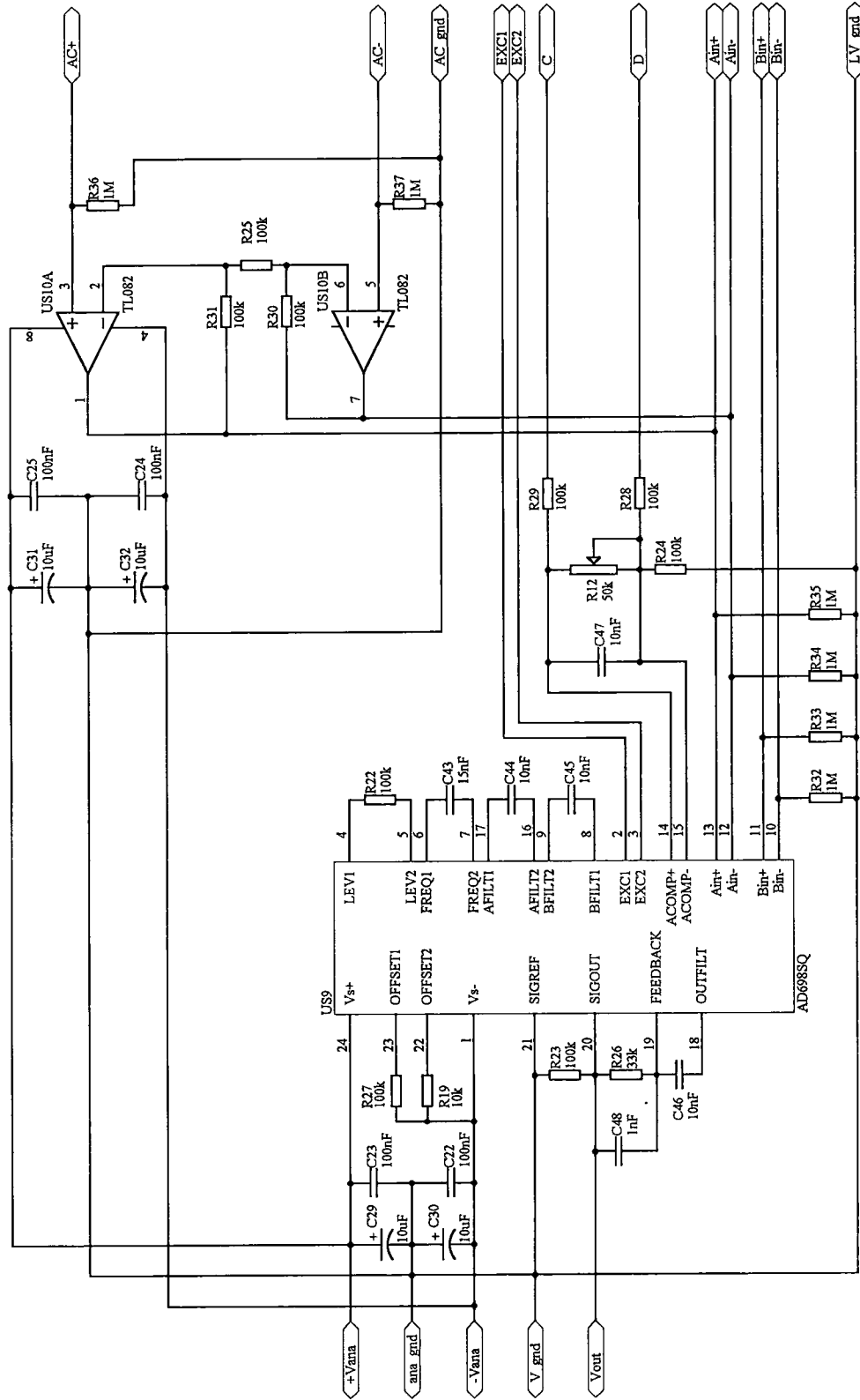
Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\LVDT10\LV10	Drawn by:



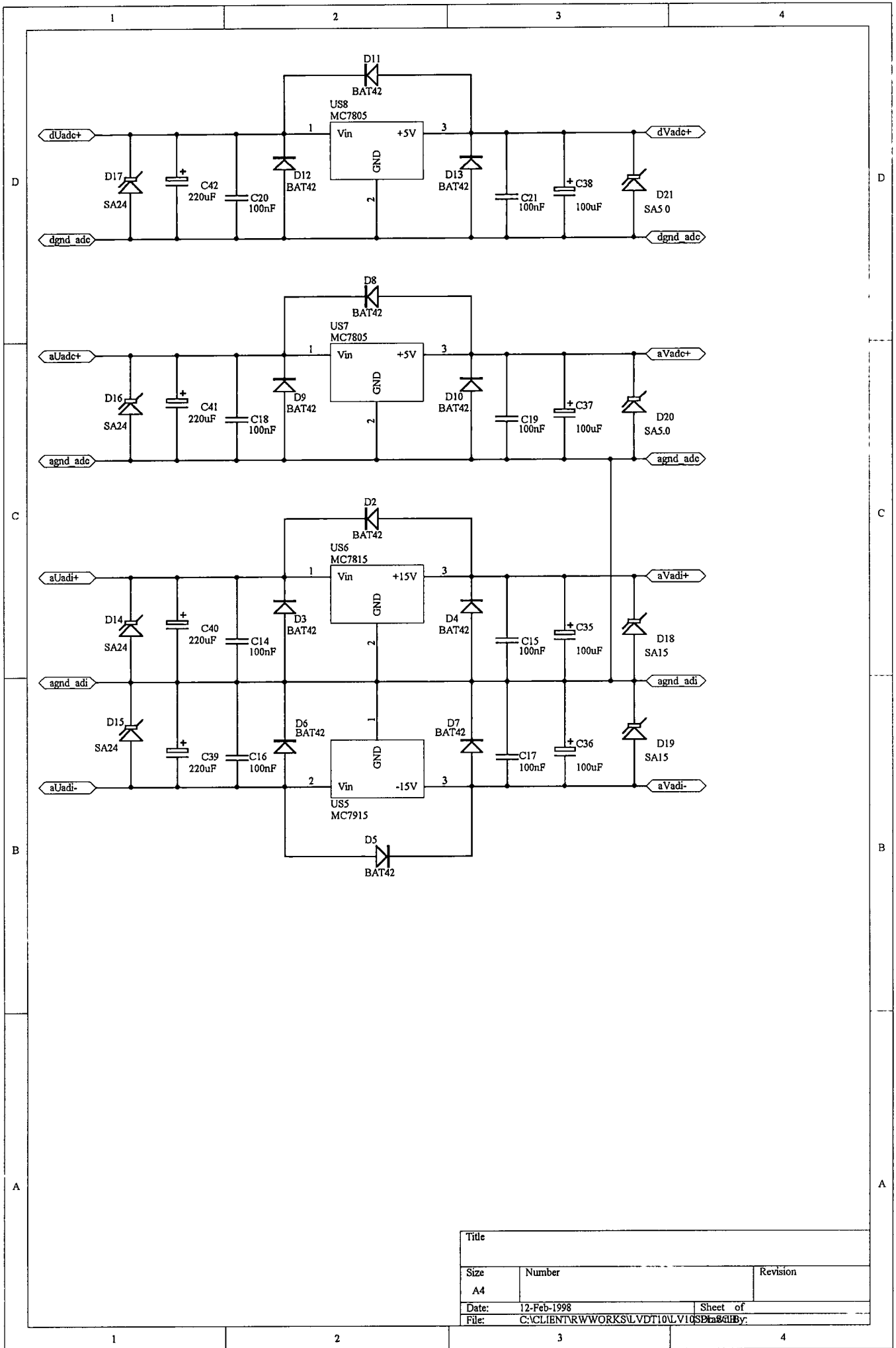
**Remarks:**

1. ADS7809 configured for EXTERNAL clock mode only.
2. ADS7809 configurable for Straight Binary or Binary two's Complement data output format.
3. ADS7809 input configurable for unipolar (0 - 4.5/10V) and bipolar (+/-3.33/5/10V) operation.

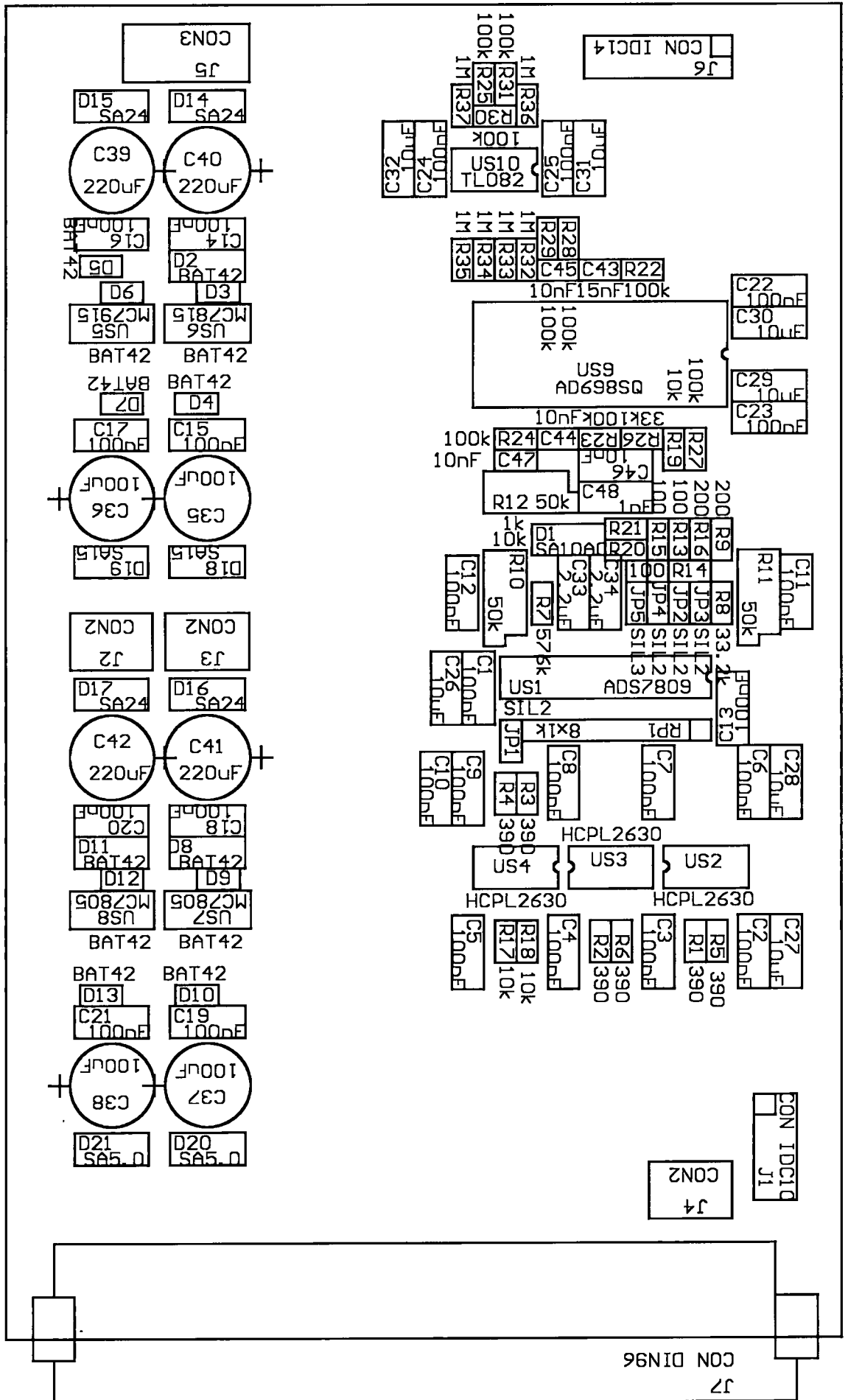
Title	
Size	Number
A4	Revision
Date:	Sheet of
File:	4



Title		Revision	
Size	Number		
A4			
Date:	12-Feb-1998	Sheet of	
File:	C:\CLIENT\WORK\SLV\DT10LV10\DT10V.A4	Sheet of	



Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\LVDT10\LV10SP1A.DWG	Drawn By:



1

2

3

4

D

D

# c8mpu10

C

C

## sbc05c8-10

B

B

A

A

Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\C8SBC10\C8FPD10.SCB	

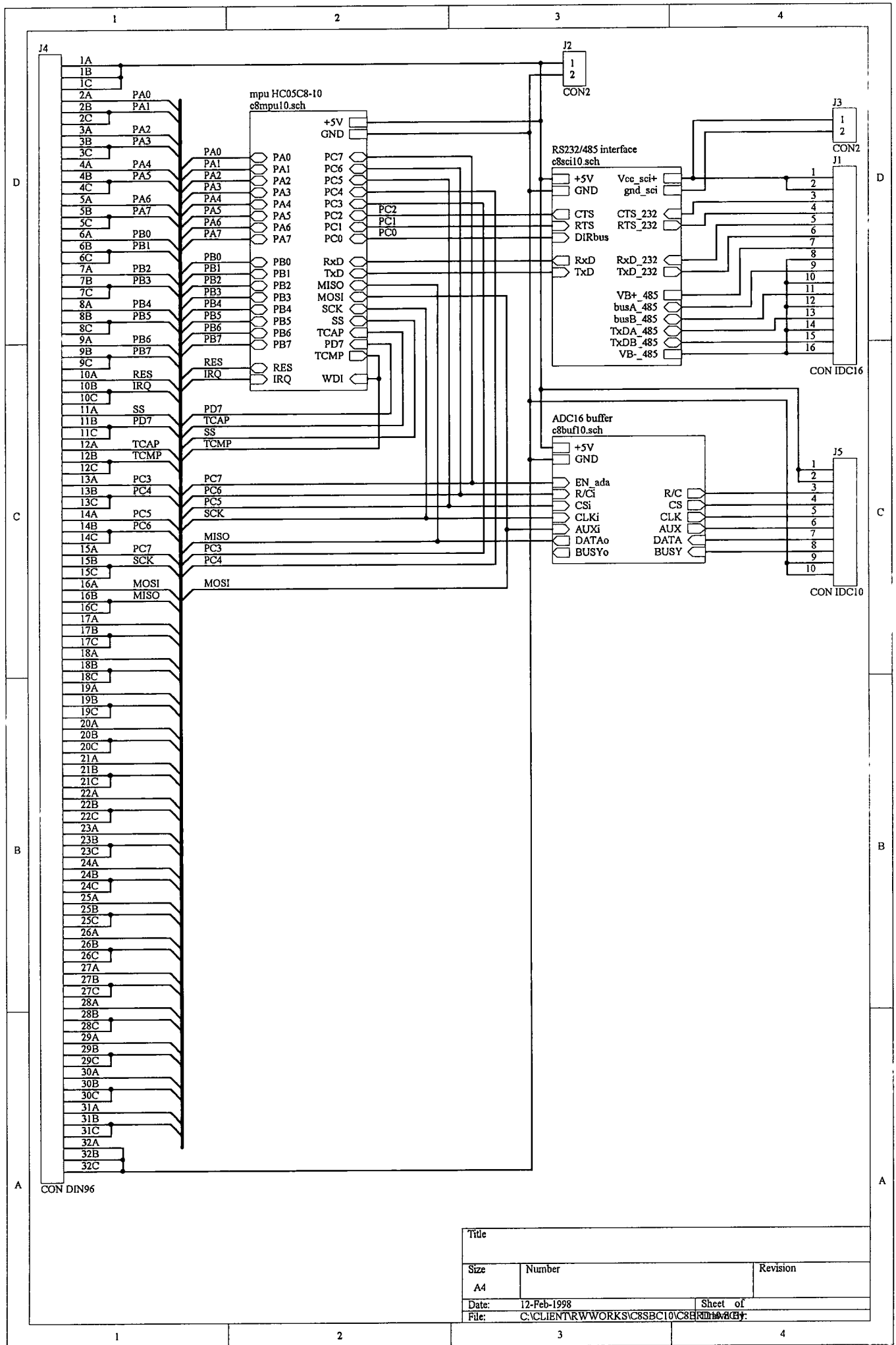
1

2

3

4

23

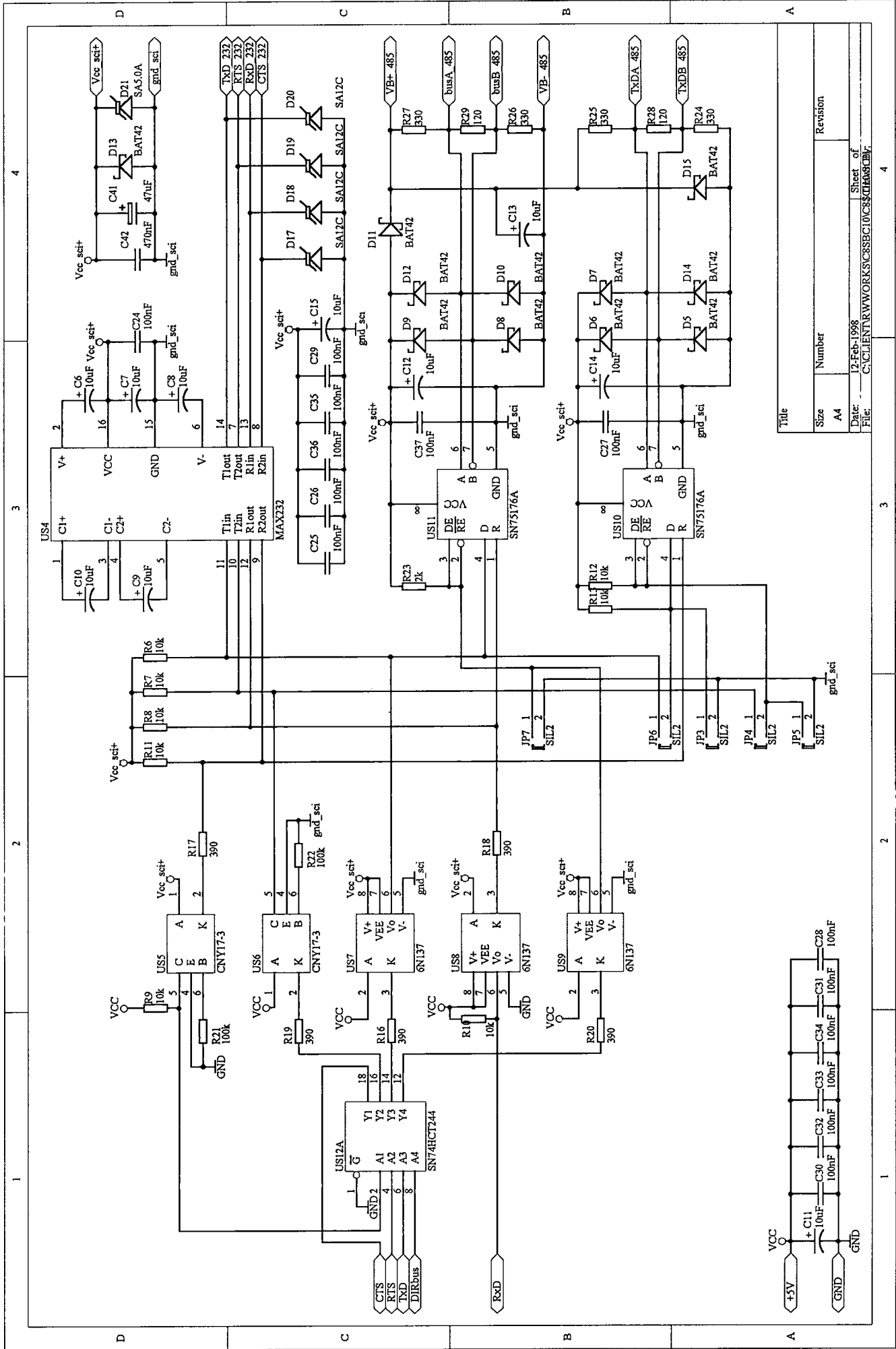


Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\C88BC10\C88B10.DWG	

24

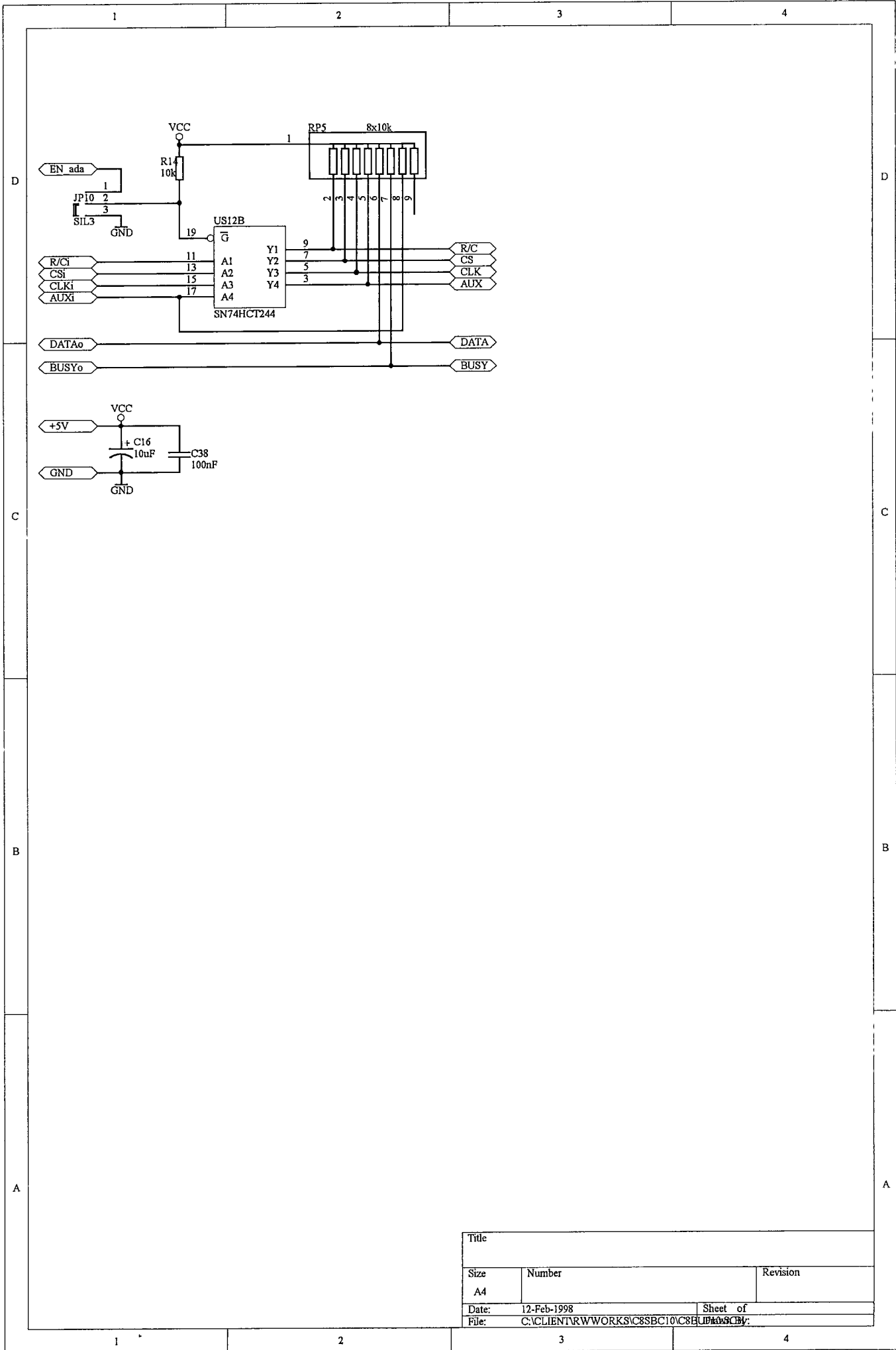






Title	
Size	Number
A4	
Date:	Revision
12-Feb-1998	
File:	Sheet of
C:\CLIENT\WORK\SCSSEC10\C35\DIRBUS.BDF	4

26



Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RWWORKS\C88BC10\C88BUP105C1	10 of 10

24

1

2

3

4

D

D

# lvdt-10

## power supply

C

C

B

B

A

A

Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\C8SBC10\C8FP10.C	

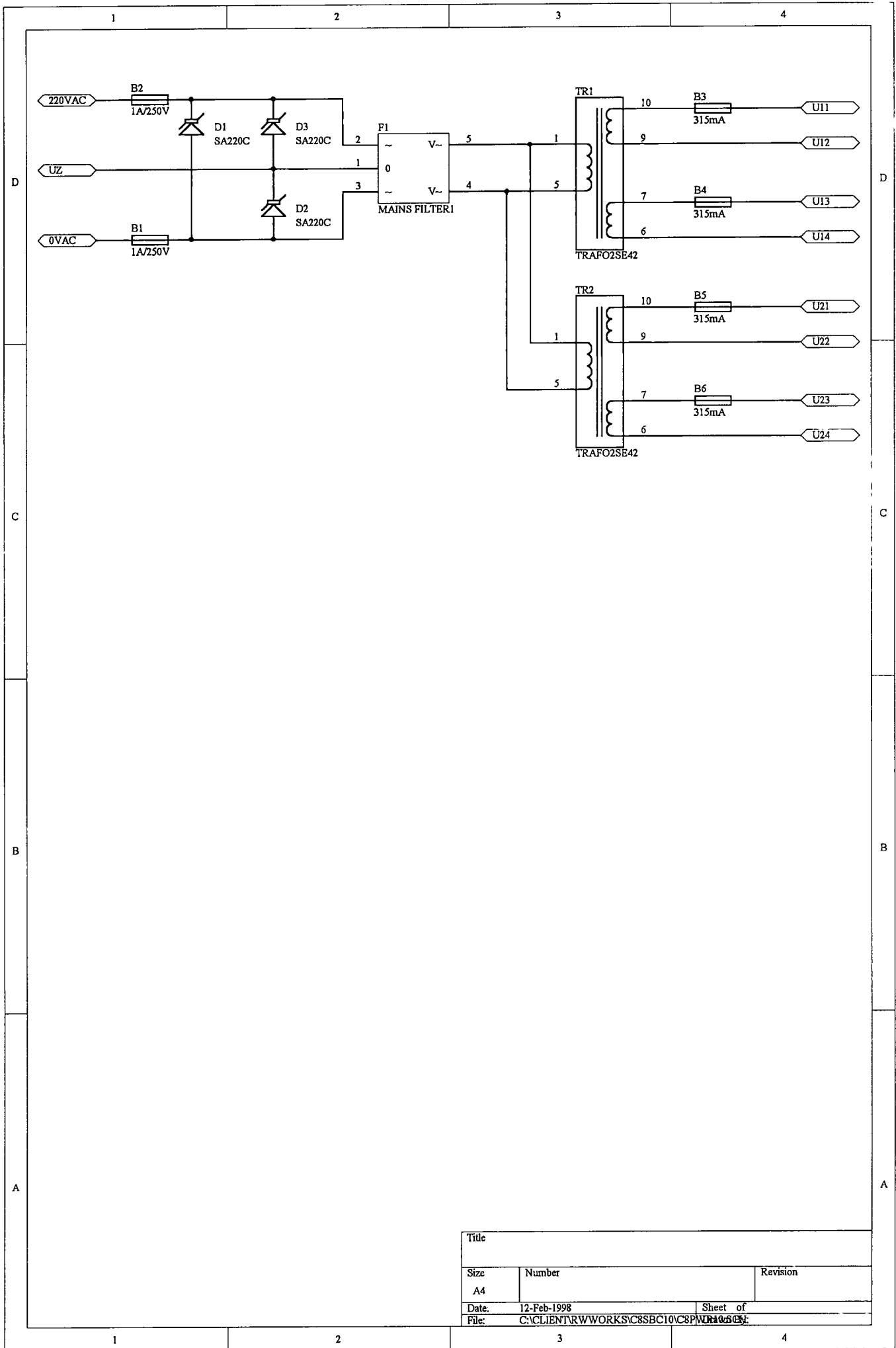
1

2

3

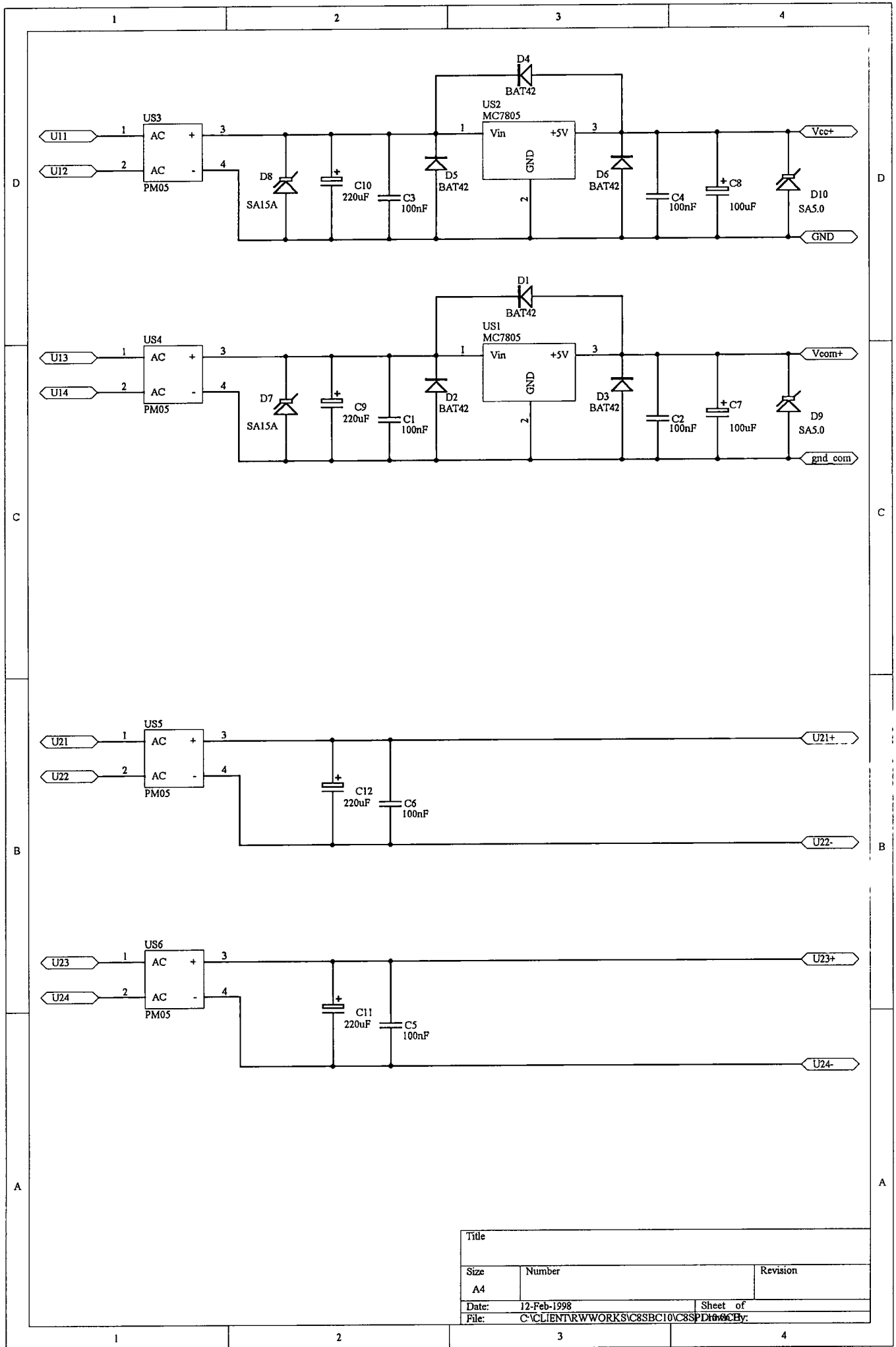
4

28



Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKSVC\8SBC10C8P\110605	11 of 11

29



Title		
Size	Number	Revision
A4		
Date:	12-Feb-1998	Sheet of
File:	C:\CLIENT\RW\WORKS\C8SBC10\C8SP1000.C	

## S1766

„Opracowanie modułów pomiarowych dotykowego pomiaru grubości do systemu pomiarowego o architekturze rozproszonej”

### ZAŁĄCZNIK Nr 3.

Dokumentacja modelu miernika tablicowego NGS96F1/ADC35

1. Moduł jednostki centralnej (MPU-F1-10)
2. Moduł wejść/wyjść analogowych (ANA-F1-10)
3. Moduł zasilacza (PWR-F1-10)
4. Moduł klawiatury funkcyjnej 1x8 (KBD-F1-10)
5. Moduł wyświetlacza LCD 5x7seg i klawiatury funkcyjnej 1x6

## MODUŁ JEDNOSTKI CENTRALNEJ

MPU-F1-10



## 1.CHARAKTERYSTYKA MODUŁU JEDNOSTKI CENTRALNEJ MPU-F1-10

Moduł jednostki centralnej jest podstawowym urządzeniem systemu F1-10, dostosowanym do obudów tablicowych typu NGS96 wg DIN43700. Obsługuje samodzielnie wybrane zadania pomiarowe w dziedzinie czasu i częstotliwości (np. pomiar szybkości obrotowych, zliczanie zdarzeń zewnętrznych itp.).

- jednostka centralna: MC68HC11F1FN
- zewnętrzny EPROM: 8/16/32/64kb
- wewnętrzny RAM: 1kb
- wewnętrzny EEPROM: 512b
- zegar czasu rzeczywistego: MC68HC68T1
- interfejs szeregowy: RS232C/422/485, izolowany
- 4 izolowane galwanicznie wejścia dwustanowe stałoprądowe
- podtrzymanie zawartości RTC, RAM\_MPU w warunkach zaniku napięcia zasilania (baterie na pakiecie)
- sygnalizacja akustyczna: beeper - CLK\_rtc
- interfejs układów wyświetlaczy alfanumerycznych LCD, z wbudowanym sterownikiem HD44780 (80x44mm max)
- wbudowany układ generacji napięcia kontrastu wyświetlacza LCD (TN/STN)
- możliwość rozszerzenia zasobów sprzętowych modułu poprzez dołączania dodatkowych pakietów systemu F1-10 (złącza J1, J2)

## 2. ARCHITEKTURA MODUŁU MPU-F1-10

### 2.1. Przestrzenie adresowe urządzeń modułu MPU-F1-10

Przestrzeń adresowa zewnętrznej pamięci programu EPROM konfigurowana jest programowo, za pomocą wewnętrznego układu generacji sygnałów dostępu i zależy wyłącznie od rodzaju zastosowanej pamięci i potrzeb aplikacji. Usytuowanie wewnętrznej pamięci RAM i EEPROM określone jest podobnie i zależy wyłącznie od potrzeb użytkownika. Sygnały dostępu do wyświetlacza LCD generowane są również za pomocą układu generacji sygnałów dostępu, i mogą być uaktywniane programowo.

Tabela 2.1. Przestrzeń adresowa sygnałów dostępu

CSx	funkcja	stan aktywny	przestrzeń adresowa
CSPROG\	ROM_CS\	0	0-64kb
CSGEN	PG6	-	-
CSIO1	E1\	0	2kb
CSIO2	E2\	0	2kb

## 2.2. Układ zegara czasu rzeczywistego RTC

Układ zegara czasu rzeczywistego (rtc - MC68HC68T1) dołączony jest do MPU za pomocą interfejsu szeregowego SPI.

Linie INT\, CPUR\, LINE i PSE wyprowadzone są na złącze ekspansji MPU (J3, Tab.3.3.) i muszą być dołączone do wybranych linii MPU (n.p. ICn), albo modułu zasilania PWR-F1-10.. Linia CLK przeznaczona jest do generacji sygnałów akustycznych (złącze BP). Układ RTC posiada niezależne źródło podtrzymania zasilania (BAT2), umieszczone na pakiecie.

Tabela 2.2. Interfejs zegara czasu rzeczywistego

MPU	RTC	Interfejs SPI MPU ustawiany jest programowo w tryb master. Linia SS\ wykorzystywana jest wówczas jako sygnał dostępu do RTC (stan aktywny wysoki). Układ RTC może być pominięty.
MOSI	MOSI	
MISO	MISO	
SCLK	SCK	
SS\	SS	

## 2.3. Interfejs wyświetlacza LCD

Wyświetlacz dołączony jest do modułu poprzez złącze J2 (Tab.3.2.). Sygnały dostępu (Enable - E1, E2) generowane są przy pomocy zanegowanych (HCT14) sygnałów dostępu do portów peryferyjnych CSIO1 i CSIO2, synchronizowanych wewnątrz sygnałem E MPU. Z uwagi na odwrócenie sygnałów CSIO1, CSIO2 (HCT14), wymagane stanem tych sygnałów po resecie, stanem aktywnym dostępu do układu wyświetlacza LCD z wbudowanym sterownikiem HD44780 jest stan niski (0). Układ interfejsu wyświetlacza LCD uzupełnia układ generacji napięcia kontrastu (ICL7660), regulowanego potencjometrem R16. Umożliwia to dołączanie wyświetlaczy typu TN i STN. W przypadku zastosowania wyświetlacza typu TN, układ generacji ujemnego napięcia (7660) może być pominięty.

## 2.4. Układ kontroli napięć zasilania modułu MPU-F1-10

Moduł jednostki centralnej posiada układ kontroli poprawności napięcia zasilania Vmpu (+5V), zapewniający podtrzymanie zawartości RAM, w warunkach zaniku napięcia zasilania (MAX694). Zmiany stanu napięcia zasilania mogą być sygnalizowane MPU poprzez przerwania. Podtrzymanie napięcia zasilania umożliwia niezależne źródło napięcia zasilania (BAT1 - 4V min), umieszczone na pakiecie. Projekt druku umożliwia zrealizowanie standardowego układu generacji sygnału RESET\, za pomocą elementów dyskretnych. Poprzez monitorowanie napięcia niestabilizowanego Umpu+ pakietu PWR-F1-10, możliwe jest wczesne wykrywanie awarii zasilania. Rozszerzoną kontrolę zasilania umożliwia wykorzystanie układu LINE SENSE RTC, monitorującego napięcie przemiennie pakietu zasilania (~Umpu).

Tabela 2.3. Funkcje systemowe układu kontroli poprawności napięcia zasilania

---

MAX694	funkcje systemowe
--------	-------------------

---

Vbat	+4V min, on board
Vo	RAM, MODB MPU
PFI	+5V/Vbat/Umpu+ (P), próg regulowany potencjometrem R8
PFO\	IRQ\ MPU, wyzwalane zboczem
RES\	RES\ MPU
WDI	n.c. może być dołączone do linii MPU

---

MC68HC68T1	
------------	--

---

LINE	~Umpu (PWR-F1-10), próg regulowany potencjometrem R8
------	------------------------------------------------------

---

## 2.5. Łącze szeregowe RS232C

Moduł MPU-F1-10 posiada wbudowane izolowane galwanicznie łącze szeregowe standardu RS232C(MAX232)/RS485(75176) (Tab.3.4.).

W przypadku wykorzystywania łącza szeregowego w standardzie RS422/485, konieczne jest dołączenie sygnału DIR do wybranej linii MPU (Tab.3.5.).

## 2.6. Izolowane galwanicznie wejścia dwustanowe stałoprądowe

Moduł MPU-F1-10 posiada 4 obwody wejść dwustanowych stałoprądowych (I0-3), izolowanych galwanicznie od masy systemu. Projekt druku umożliwia zastosowanie w trzech obwodach (I1,2,3) szybkich transoptorów 6N137, umożliwiających pomiary w dziedzinie czasu z pełną dokładnością timera MPU. Wejście I0 umożliwia dołączanie typowych sygnałów obiektowych (CNY17,  $t_{on/off} = 5\mu s$ ). Z uwagi na otwarty charakter modułu MPU-F1-10, sygnały wejściowe nie zostały dołączone do MPU. Użytkownik konfiguruje pakiet do potrzeb aplikacji (Tab.3.7.). Również złącze obiektowe INP (Tab.3.6.) być skonfigurowane zgodnie z wymaganiami zadania.

## 2.7. Rozszerzenie funkcji modułu PGM-F1-10

Zasoby sprzętowe modułu jednostki centralnej MPU-F1-10 mogą być rozszerzane poprzez dołączanie dodatkowych modułów (np. wejść/wyjść analogowych ANA-F1-10, klawiatury KBD-F1-10, wyświetlacza cyfrowego LCD-F1-10) do złączy ekspansji J1 (Tab.3.1) i J2(Tab3.2.).

## 2.8. Konfiguracja sprzętowa modułu MPU-F1-10 do potrzeb aplikacji

Modułowi jednostki centralnej MPU-F1-10 nadano otwarty charakter, wyposażając procesor w szereg urządzeń dodatkowych (rtc, 4 izolowane wejścia binarne, interfejs szeregowy RS232/485, interfejs wyświetlaczy alfanumerycznych, układ kontroli poprawności napięcia zasilania.), ale pozostawiając pewne sygnały niedołączone do MPU. Użytkownik może dzięki temu dowolnie skonfigurować pakiet do potrzeb zadania, uwzględniając również wymagania pakietów rozszerzających zasoby sprzętowe modułu MPU-F1-10. Wymaga to wykonania połączeń punktów konfiguracyjnych pakietu (Tab.3.9.) z wybranymi liniami MPU (złącza ekspansji J1 - Tab3.1., J2 - Tab3.2.). Również złącze obiektowe INP (Tab.3.6.) wymaga konfiguracji. Rozwiązanie to umożliwia opracowanie nowych urządzeń i wykonywanie pojedynczych egzemplarzy. W przypadku większych serii produkcyjnych wybranych urządzeń konieczne jest opracowanie nowych płytek drukowanych, dostosowanych do wytwarzanych urządzeń.

### 3. OPIS ZŁĄCZ MODUŁU MPU-F1-10

Tabela 3.1. Złącze ekspansji MPU (J1)

nr_a	sygnał	nr_b	sygnał
1a.	Vmpu+ (+5V)	1b.	Vmpu+ (+5V)
2a.	PA0 (IC3)	2b.	PA1 (IC2)
3a.	PA2 (IC1)	3b.	PA3 (IC4)
4a.	PA4 (OC4)	4b.	PA5 (OC3)
5a.	PA6 (OC2)	5b.	PA7 (PAI)
6a.	PD5 (SS\)	6b.	PD4 (SCK)
7a.	PD3 (MOSI)	7b.	PD2 (MISO)
8a.	PD1 (TxD)	8b.	PD0 (RxD)
9a.	PG0	9b.	PG2
10a.	PG1	10b.	PG3
11a.	PG4 (CSIO2)	11b.	PG5 (CSIO1)
12a.	PG6 (CSGEN)	12b.	IRQ\
13a.	XIRQ\	13b.	RES\
14a.	n.c. (OUTA)	14b.	PSE_rtc
15a.	n.c. (OUTB)	15b.	n.c. (PFI PWR-F1-10)
16a.	n.c.	16b.	n.c. (~Umpu PWR-F1-10)
17a.	GND	17b.	GND

Tabela 3.2. Złącze ekspansji MPU-F1-10 (J2)

nr_a	sygnał	nr_b	sygnał	
1a.	GND	1b.	GND	(COM)
2a.	Vmpu+ (+5V)	2b.	Vmpu+ (+5V)	
3a.	PE4	3b.	PE0	
4a.	PE5	4b.	PE1	
5a.	PE6	5b.	PE2	
6a.	PE7	6b.	PE3	
7a.	VRH	7b.	VRH	VRL - zwarte
8a.	GND	8b.	GND	do masy (GND)

Tabela 3.3. Złącze wyświetlacza alfanumerycznego LCD (J3)

nr	sygnał	funkcja
1.	GND	
2.	VDD	(Vmpu+)
3.	Vo	(napięcie kontrastu wyświetlacza LCD)
4.	RS	(A0 - MPU)
5.	R/W	(R/W - MPU)
6.	E1	(CSIO1\ - MPU)
7.	DB0	(D0 - MPU)
8.	DB1	(D1 - MPU)
9.	DB2	(D2 - MPU)
10.	DB3	(D3 - MPU)
11.	DB4	(D4 - MPU)
12.	DB5	(D5 - MPU)
13.	DB6	(D6 - MPU)
14.	DB7	(D7 - MPU)
15.	n.c.	
16.	E2	(CSIO2\ - MPU)



Tabela 3.4. Złącze obiektowe interfejsu szeregowego RS232C (COM)

ozn.	RS232C	RS485
COM		shield_485
+		Vbus_485+
R	RxD_232	A_485
T	TxD_232	B_485
-	gnd_232	gnd_485

Tabela 3.5. Przypisanie sygnałów RS232/485 liniom układu LS244

RS485	LS244	
RxD	1A2	
TxD	1Y1	
DIR	1Y4	(1A4 dołączyć do wybranej linii MPU)

Tabela 3.6. Złącze obiektowe wejść dwustanowych (INP)

nr.	sygnał	(opto)	
0.	Vin+		Złącze konfigurowane do potrzeb aplikacji.
1.	I0	(CNY17)	
2.	I1	(6N137)	
3.	I2	(6N137)	
4.	I3	(6N137)	
5.	Vin-		

Tabela 3.7. Przypisanie sygnałów wejść izolowanych liniom układu LS244

sygnał	LS244	
I0.	2A3	Wyjścia 2Yn układu LS244 dołączyć do wybranych linii MPU, wg potrzeb aplikacji.
I1.	2A2	
I2.	2A1	
I3.	2A4	

Tabela 3.8. Linie zasilania

sygnał	funkcja
Vin+/-	zasilanie czujników obwodu wejść dwustanowych
Vrs+/-	zasilanie układów interfejsu szeregowego
+/-5V	zasilanie MPU (Vmpu+, GND)

Tabela 3.9. Punkty konfiguracyjne pakietu MPU-F1-10

ozn.	funkcja	konfiguracja
F	PFO\ (MAX694)	pull_up, IRQ\ (edge), albo XIRQ\
X	XIRQ\ (MPU)	
I	IRQ\ (MPU)	
W	WDI (MAX694)	n.c., albo wybrana linia MPU
P	PFI (MAX694)	Vmpu+, BAT1+, P (PWR-F1-10)
V	Vmpu+ (+5V)	
BAT1+/-	STANDBY MPU (4V min)	przełączanie baterii MPU
L	LINE (MC68HC68T1)	n.c., albo ~Umpu (PWR-F1-10)
BAT2+/-	STANDBY RTC (3V)	przełączanie baterii RTC
A	1Y3 (LS244)	n.c., albo OUTA (PWR-F1-10)
B	2A2 (LS244)	I1, albo OUTB (PWR-F1-10)
C	2A3 (LS244)	DIR_RS485, albo OUTB (PWR-F1-10)
D	1Y4 (LS244)	I0, albo OUTB (PWR-F1-10)
BP	CLK (MC68HC68T1)	sygnał sterujący beepera
+	Vdd+ (MC68HC68T1)	alternatywne zasilanie beepera
-	GND (MC68HC68T1)	linia zasilania beepera

DMT 96-F1. Konfiguracja wyjści MPU-F1-10

$I_0 \rightarrow IC_4$  (N)

$I_1 \rightarrow IC_1$  (F)

$I_2 \rightarrow IC_2$  (F)

$I_3 \rightarrow IC_3$  (F)

$P6 \rightarrow DI2485$

# ANA96-F1. Konfiguracja współpracy z MPU-F1-10

dnA	-	PA <sub>0</sub>	(IC <sub>3</sub> )	}	adc 7135
dnB	-	PA <sub>1</sub>	(IC <sub>2</sub> )		
ead	-	IC <sub>1</sub>	(PA <sub>2</sub> )		
POL	-	PA <sub>3</sub>	(IC <sub>4</sub> )		
CLK	-	PAI	(PA <sub>7</sub> )		

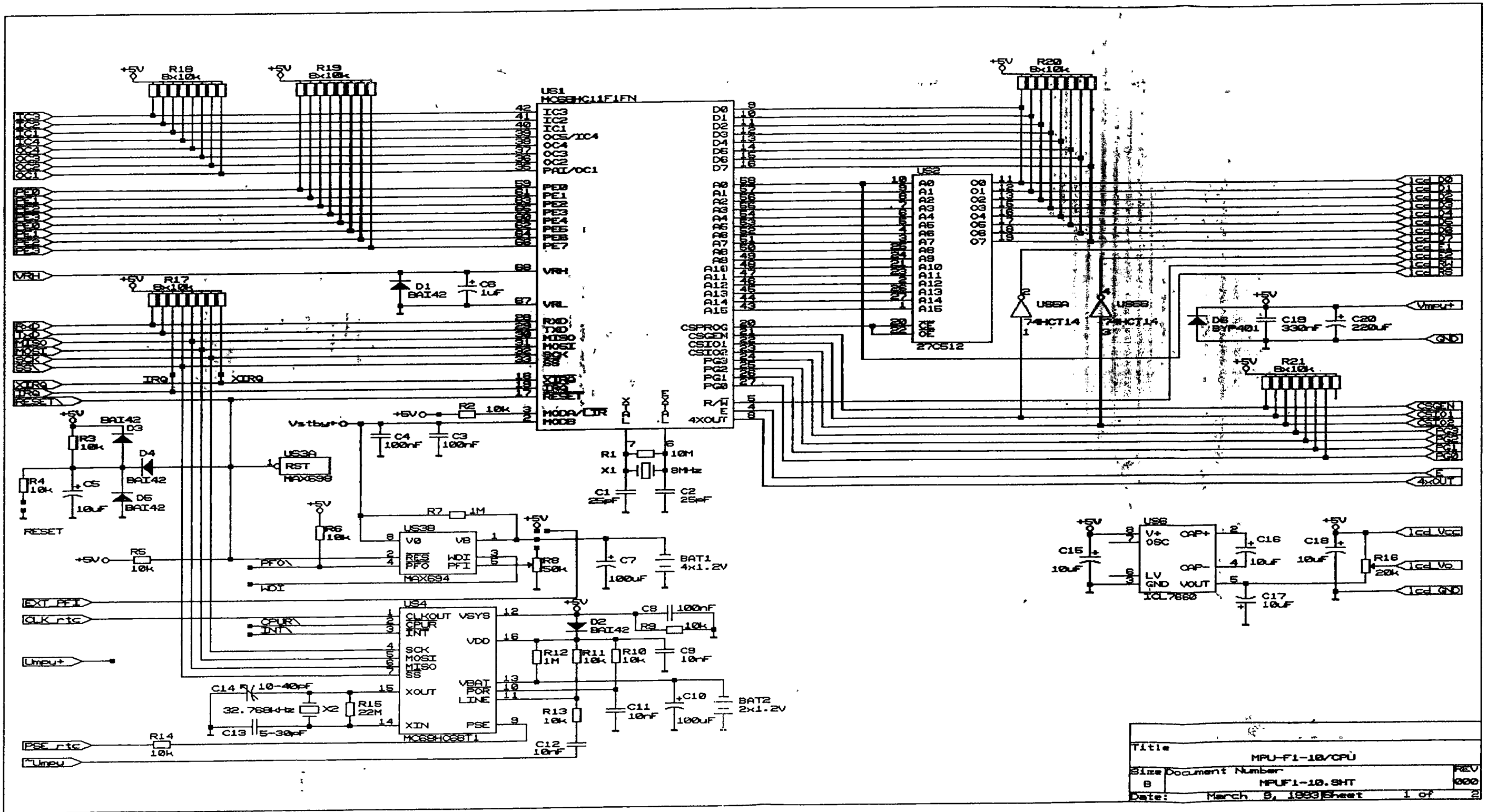
LOAD	-	OC2	}	doc 7543
CLK	-	SCK		
SEI	-	MOSI		

WYKAZ ELEMENTÓW MODELU MPU-F1-10

L.p.	Liczba	Oznaczenie	Typ
1.	1	U31	MC68HC11F1FN
2.	1	U32	27C512
3.	1	U33A	MAX698
4.	1	U33B	MAX694
5.	1	U34	MC68HC68T1
6.	1	U35	74HCT14
7.	1	U36	ICL7660
8.	1	U37	74HCT244
9.	6	U38, U39, U310, U314, U315, U316	6N137
10.	1	U311	75176
11.	1	U312	MAX232
12.	1	U313	CNY17-2
13.	1	X1	8MHz
14.	1	X2	32.768kHz
15.	5	D1, D2, D3, D4, D5	BAI42
16.	2	D6, D7	BYP401
17.	1	D8	BAI42
18.	4	D9, D11, D13, D15	6V8
19.	4	D10, D12, D14, D16	BAVP18
20.	1	R1	10M
21.	10	R2, R3, R4, R5, R6, R9, R10, R11, R13, R14	10k
22.	1	R8	50k
23.	2	R7, R12	1M
24.	1	R15	22M
25.	1	R16	20k
26.	5	R21, R17, R18, R19, R20	8x10k
27.	1	R22	8x10k
28.	7	R23, R24, R28, R37, R38, R39, R40	390
29.	3	R25, R26, R27	1k
30.	6	R29, R21, R33, R34, R35, R36	10k
31.	4	R41, R42, R43, R44	24V
32.	4	R46, R45, R47, R48	10
33.	1	R49	100k
34.	2	C1, C2	25pF
35.	3	C3, C4, C8	100nF
36.	5	C5, C15, C16, C17, C18	10uF
37.	1	C6	1uF
38.	2	C7, C10	100uF
39.	3	C9, C11, C12	10nF
40.	1	C13	5-30pF
41.	1	C14	10-40pF
42.	1	C19	330nF
43.	1	C20	220uF

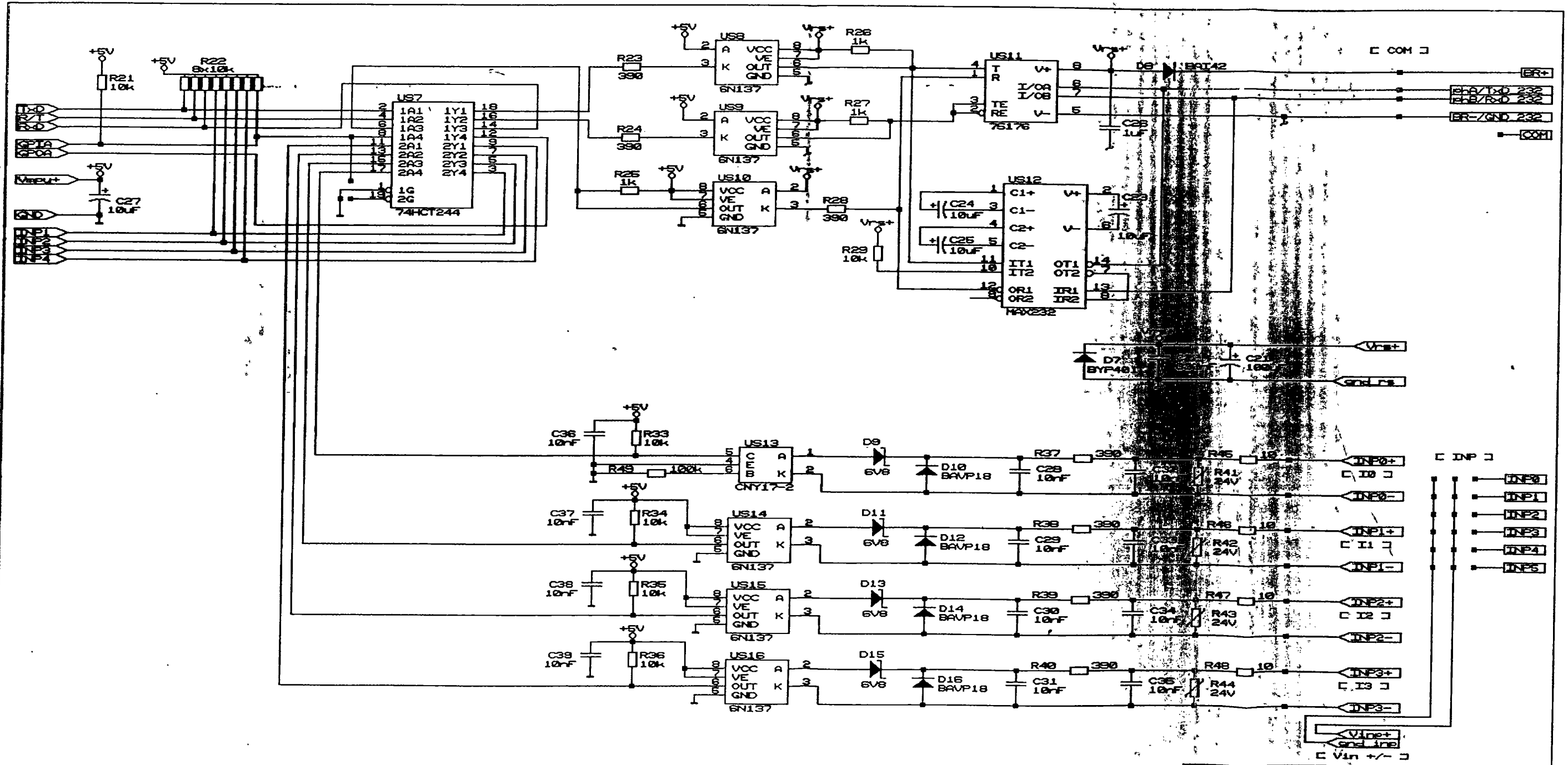
44.	1	C21	100uF
45.	1	C22	330nF
46.	4	C23, C24, C25, C27	10uF
47.	1	C26	1uF
48.	12	C28, C29, C30, C31, C32, C33, C34, C35, C36, C37, C38, C39	10nF
49.	1	J1	JDIL34
50.	1	J2	JDIL16
51.	1	J3	JSIL16
52.	1	BAT1	4V min
53.	1	BAT2	2.4V min

---



Title		MPU-F1-10/CPU	
Size Document Number		MPUF1-10.SHT	
B	REV	1 of	2
Date:		March 8, 1993	





Title		MPU-F1-16/INP.COM	
Size Document Number		MPUF1-11.SHT	
Date:	March 9, 1993	Sheet	2 of 2



**MODUŁ WEJŚĆ/WYJŚĆ ANALOGOWYCH**

**ANA-F1-10**

## 1. CHARAKTERYSTYKA MODUŁU WEJŚĆ/WYJŚĆ ANALOGOWYCH ANA-F1-10

Moduł zawiera układ 4 wejść analogowych (16 bitowych) standardu 4-20mA i jednokanałowe wyjście (12 bitowe) prądowe 4-20mA (albo napięciowe 0-1V).

Moduł dostosowano do obudów tablicowych NGS96 standardu DIN43700.

### 1.2. Charakterystyka wejść analogowych modułu ANA-F1-10

- 4 kanały standardu 4-20mA, automatycznie skanowane sygnałem BUSY\ przetwornika adc (co ok. 0.3s)
- zakres napięć wejściowych +/- 2V
- rezystancja pomiarowa 100 ohm (dzielona 2x50)
- możliwość kalibracji poziomu sygnałów wejściowych (symetryzacja pomiędzy kanałami)
- możliwość dołączenia ekranów przewodów obiektowych
- zabezpieczenie układów wejściowych przed przepięciami
- złącze obiektowe konfigurowane do potrzeb aplikacji
  
- 16 bitowy przetwornik integracyjny adc ICL7135
- możliwość zmiany częstotliwości pracy przetwornika (typ. 125kHz, czas konwersji ok.300ms)
- identyfikacja kanału przetwarzania
- sygnalizacja polaryzacji sygnału wejściowego
- wbudowany układ generacji ujemnego napięcia zasilania (-5V)
- rozdzielone napięcia zasilania toru analogowego i cyfrowego
- wbudowany układ zasilania obwodu czujników zewnętrznych (stabilizator +12Vdc)
- złącze ekspansji, umożliwiające zastąpienie układów wejściowych standardu 4-20mA, układami specjalizowanymi
- izolacja galwaniczna wejść analogowych od masy jednostki centralnej
- izolacja galwaniczna wejść analogowych od wyjścia analogowego

### 1.3. Charakterystyka wyjścia analogowego modułu ANA-F1-10

- jednokanałowe wyjście prądowe 4-20mA, albo napięciowe 0-1V
- możliwość zasilania układów wyjściowych z pętli 4-20mA, albo wbudowanego zasilacza (Vdac - PWR-F1-10)
- złącze obiektowe konfigurowane do potrzeb aplikacji
  
- 12 bitowy przetwornik dac MAX543
- wbudowany układ generacji ujemnego napięcia zasilania (-5V) i referencyjnego
- separacja układu przetwarzania od stopnia wyjściowego
- zabezpieczenie stopnia wyjściowego przed przepięciami i przeciążeniem
- izolacja galwaniczna wejść analogowych od masy jednostki centralnej
- izolacja galwaniczna wejść analogowych od wyjścia analogowego
- współpraca z modułem MPU-F1-10 poprzez łącze SPI

### 1.4. Opis funkcjonowania toru wejść analogowych

Tor analogowy zrealizowano w oparciu o dokładny przetwornik integracyjny ICL7135 (4.5 cyfry + znak). Układ cztero kanałowego multipleksera wejściowego (CD4051) odseparowano dokładnym wzmacniaczem ICL7650, pracującym w układzie wzmacniacza różnicowego. Kanały wejściowe skanowane są automatycznie sygnałem końca konwersji (BUSY\), wyznaczającym początek fazy autozerowania przetwornika. Stan wysoki sygnału BUSY\\_adc wyznacza bramkę dla impulsów zegarowych CLK\\_adc, zliczanych przez mikrokontroler. Numer kanału przetwarzanego dostępny jest MPU na wyjściach CHA\\_adc i CHB\\_adc.

### 1.5. Obsługa wyjścia analogowego

Przetwornik dac MAX543 współpracuje z MPU poprzez interfejs SPI.

## 2. ZŁĄCZA MODUŁU WEJŚĆ/WYJŚĆ ANALOGOWYCH ANA-F1-10

Tabela 2.1. Złącze współpracy z modułem jednostki centralnej MPU-F1-10 (J1)

nr_a.	sygnał	(MPU)	nr_b.	sygnał	(MPU)
1a.	Vmpu+	(+5V)	1b.	Vmpu+	(+5V)
2a.	CLK_dac	(SCLK)	2b.	SRI_dac	(MOSI)
3a.	GND		3b.	LOAD_dac	(PA/PGx)
4a.	CHA_adc	(PA/PGx)	4b.	GND	
5a.	BUSY_adc	(ICx)	5b.	POL_adc	(PA/PGx)
6a.	CLK_adc	(PAI)	6b.	CHB_adc	(PA/PGx)
7a.	GND		7b.	GND	
8a.	GND		8b.	GND	

Tabela 2.2. Złącze obiektowe toru analogowego

nr.	sygnał	
1.	VADC+	złącze obiektowe jest konfigurowane
2.	gnd_adc	do potrzeb aplikacji
3.	AN0	
4.	AN1	
5.	AN2	
6.	AN3	
7.	Vext+	

Tabela 2.3. Złącze aplikacyjne toru adc (J2)

nr.	sygnał	(konfiguracja)	
1.	Vadc-	(-5V)	(projekt druku umożliwia łatwe
2.	1y	(IN-, gnd_adc)	odłączenie wejść ny od masy
3.	1x	(IN+)	gnd_adc)
4.	3y	(IN-, gnd_adc)	
5.	3x	(IN+)	IN+/- - wejścia 7650
6.	2y	(IN-, gnd_adc)	
7.	2x	(IN+)	
8.	0y	(IN-, gnd_adc)	
9.	0x	(IN+)	
10.	gnd_adc		
11.	Vadc+	(+5Vadc)	
12.	Vadc+	(+5Vadc)	

Tabela 2.4. Przypisanie kanałów wejściowych układowi multipleksera (CD4051)

we	CH_BA	CD4051	
AN0	00	0x	wyjście x dołączono do IN+ 7650
AN1	01	1x	wyjście y dołączono do IN- 7650
AN2	10	2x	wejścia ny zwarte są do masy (gnd_adc)
AN3	11	3x	

Tabela 2.5. Złącze obiektowe toru dac

ozn.	4-20mA	(0-1V)	
+	V+	(Vdac+)	
T	shield	(Vout_dac)	sygnał konfigurowalny
-	V-	(gnd_dac)	

WYKAZ ELEMENTÓW MODELU ANA-F1-10

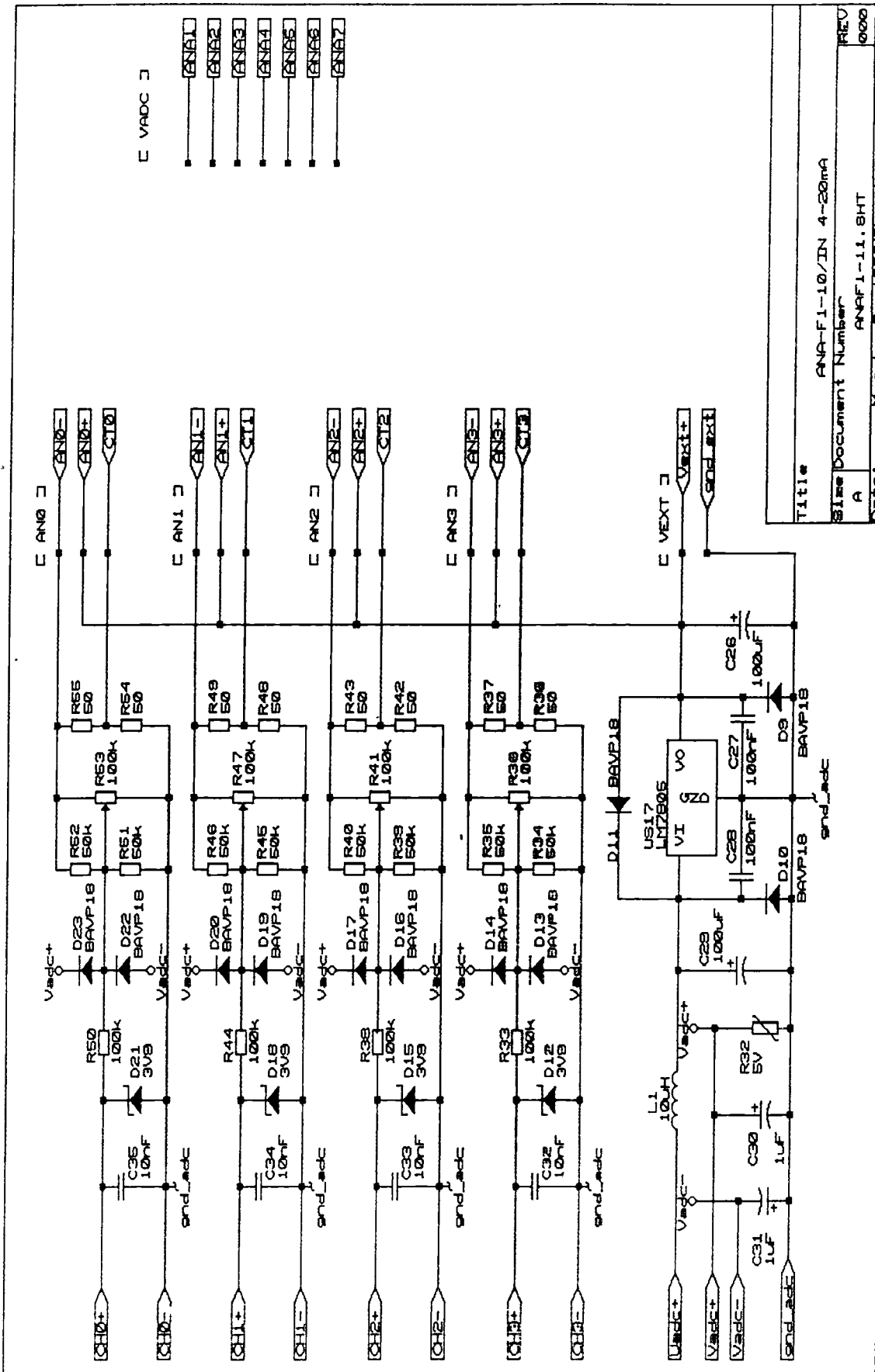
Lp.	Liczba	Oznaczenie	Typ
1.	2	US1, US10	74HCT244
2.	5	US2, US3, US4, US5, US6	CNY17-2
3.	2	US7, US9	74HCT393
4.	1	US8	74HCT00
5.	1	US11	ICL7135
6.	2	US12, US16	ICL7650
7.	1	US13	CD4052
8.	4	US14, US15, US17, US24	LM7805
9.	3	US18, US19, US20	6N137
10.	1	US21	ICL7660
11.	1	US22	MAX543
12.	1	US23	TL062
13.	1	T1	BC147
14.	1	T2	BC148
15.	1	X1	4.194MHz
16.	1	D1	BYP401
17.	1	D2	LM185-1.2V
18.	19	D3, D4, D5, D6, D7, D8, D9, D10, D11, D13, D14, D16, D17, D19, D20, D22, D23, D25, D26	BAVP18
19.	4	D12, D15, D18, D21	3V9
20.	1	D24	LM185-2.5
21.	20	R1, R2, R3, R4, R5, R16, R17, R18, R19, R27, R59, R60, R61, R62, R63, R64, R65, R67, R68, R69	10k
22.	14	R6, R7, R8, R9, R10, R20, R21, R22, R23, R24, R25, R26, R28, R31	100k
23.	8	R11, R12, R13, R14, R15, R56, R57, R58	390
24.	1	R29	25k
25.	1	R30	1k
26.	1	R32	5V
27.	8	R33, R36, R38, R41, R44, R47, R50, R53	100k
28.	8	R34, R35, R39, R40, R45, R46, R51, R52	50k
29.	8	R37, R38, R42, R43, R48, R49, R54, R55	50
30.	1	R70	100



31.	1	R71	30V
32.	7	C1, C3, C6, C8, C11 C26, C27	100uF
33.	5	C2, C19, C20 C31, C30	1uF
34.	9	C4, C5, C9, C10, C17, C18, C21 C27, C28 C43, C44	100nF
35.	9	C7, C12, C13, C14, C15, C16 C38, C39, C40	10uF
36.	1	C22	220nF
37.	1	C23	12pF
38.	5	C24 C32, C33, C34, C35	10nF
39.	1	C25	10pF
40.	3	C36, C42, C45	47uF
41.	1	C37	330nF
42.	1	C41	0.1uF
43.	1	L1	10uH
44.	1	J1	JDIL16

---

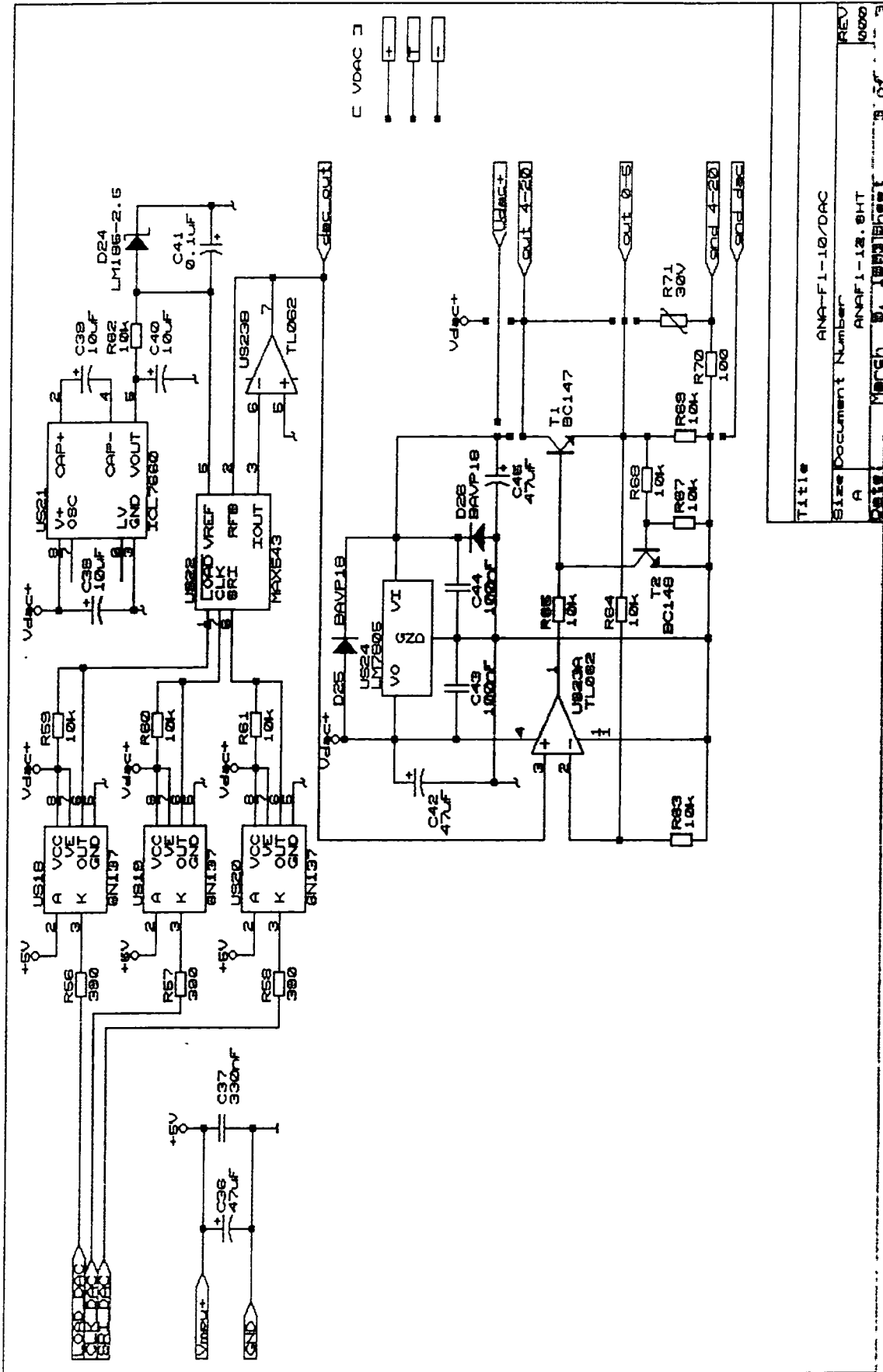




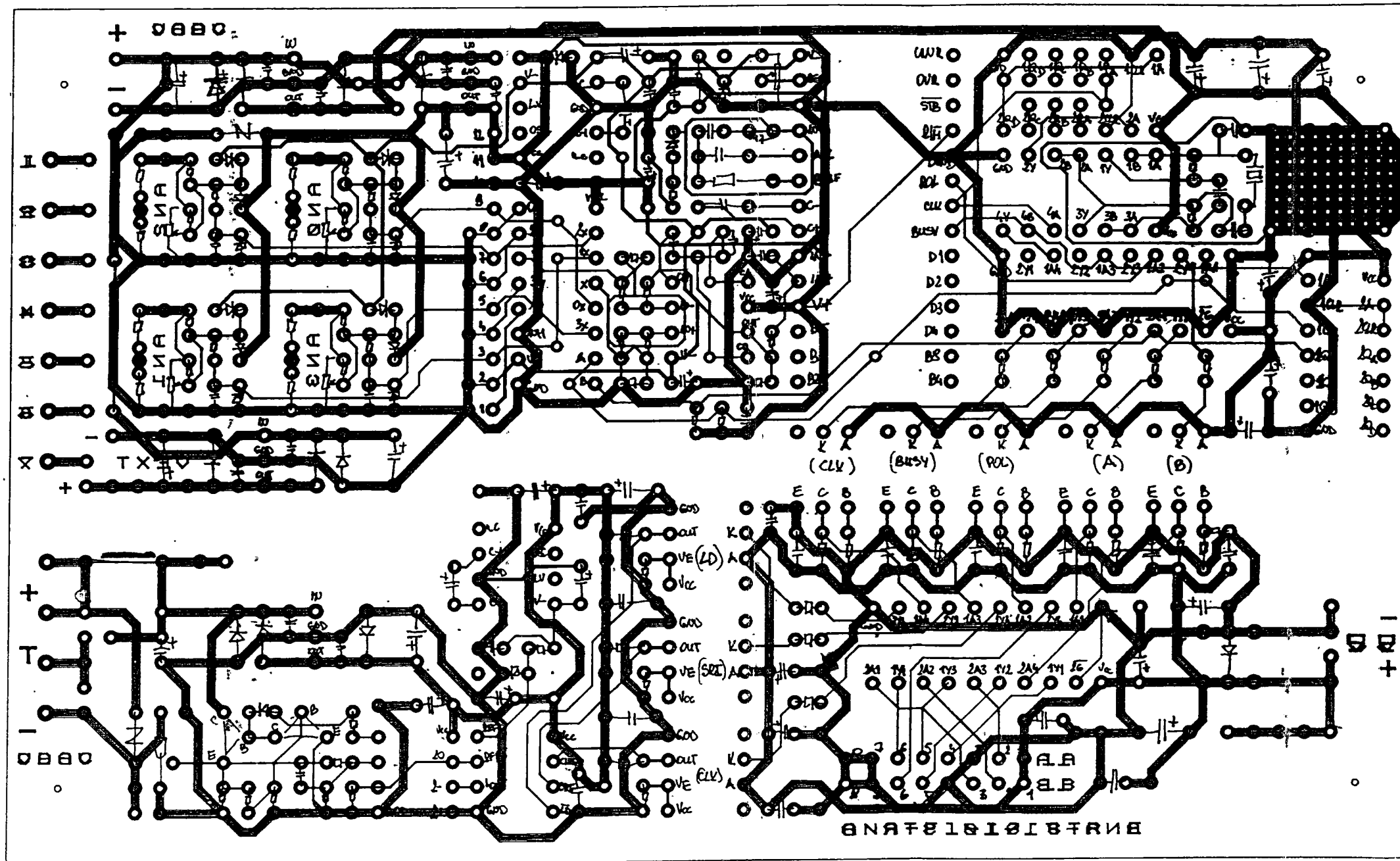
C VADC 0

- AN0
- AN1
- AN2
- AN3
- AN6
- AN7

Title	
ANAFI-10/IN 4-20mA	REV
Slave Document Number	000
A ANAFI-11.8HT	
Date: March 9, 1993	Sheet 2 of 3



Title	ANA-F1-10/DAC
Size	A
Document Number	ANAF1-12.8HT
REV	0000



01

**MODUŁ ZASILACZA**

**PWR-F1-10**

## 1. CHARAKTERYSTYKA MODUŁU ZASILACZA PWR-F1-10

Moduł zasilany jest zewnętrznym napięciem 220Vac. Na wejściu zastosowano filtr sieciowy (220Vac/2A). Projekt druku umożliwia montaż transformatorów typu EI38 (3VA) i EI42 (4.5VA).

Moduł zasilacza PWR-F1-10 generuje cztery izolowane napięcia zasilania:

- stabilizowane (Vmpu):           +5Vdc (170/220mA)     (EI38/EI42)
- stabilizowane (Vrs):           +5Vdc (170/220mA)
- niestabilizowane (Vadc):       +12Vdc (170/220mA)
- niestabilizowane (Vdac):       +12Vdc (170/220mA)

Moduł posiada dwa izolowane galwanicznie wyjścia przekaźnikowe 220Vac/10A (OUTA, OUTB) wraz z układami sterowania.

Konstrukcję mechaniczną dostosowano do obudów tablicowych NGS96 standardu DIN43700.

## 2. ZŁĄCZA MODUŁU ZASILANIA PWR-F1-10

Tabela 2.1. Złącze obiektowe napięcia zasilania 220Vac

ozn.	sygnał
U	220Vac
0	EARTH
U	0Vac

Tabela 2.2. Złącze obiektowe wyjść przekaźnikowych

ozn.	funkcja
OUTA	channel A
OUTB	channel B

(każdy kanał wyjściowy posiada dwa nieoznaczone wyprowadzenia styków przekaźnika)

Tabela 2.3. Złącze wejść sterujących wyjściami OUTA, OUTB

ozn.	funkcja	stan aktywny
A	channel A	0
B	channel B	0
+	Vmpu+	
-	GNDmpu	

Układ sterowania wyjściami przekaźnikowymi zasilany jest napięciem pakietowym Vdac, wykorzystywanym również do zasilania toru przetwornika dac modułu ANA-F1-10 (o ile pracuje jako wyjście napięciowe).



Tabela 2.4. Złącze zasilania toru dac modułu ANA-F1-10 (VDAC)

ozn.	funkcja
Vdac+	+12Vdc
Vdac-	gnd_dac

Tabela 2.5. Złącze zasilania toru adc modułu ANA-F1-10 (VADC)

ozn.	funkcja
Vadc+	+12Vdc
Vadc-	gnd_dac

Tabela 2.6. Złącze zasilania interfejsu szeregowego modułu MPU-F1-10 (VRS)

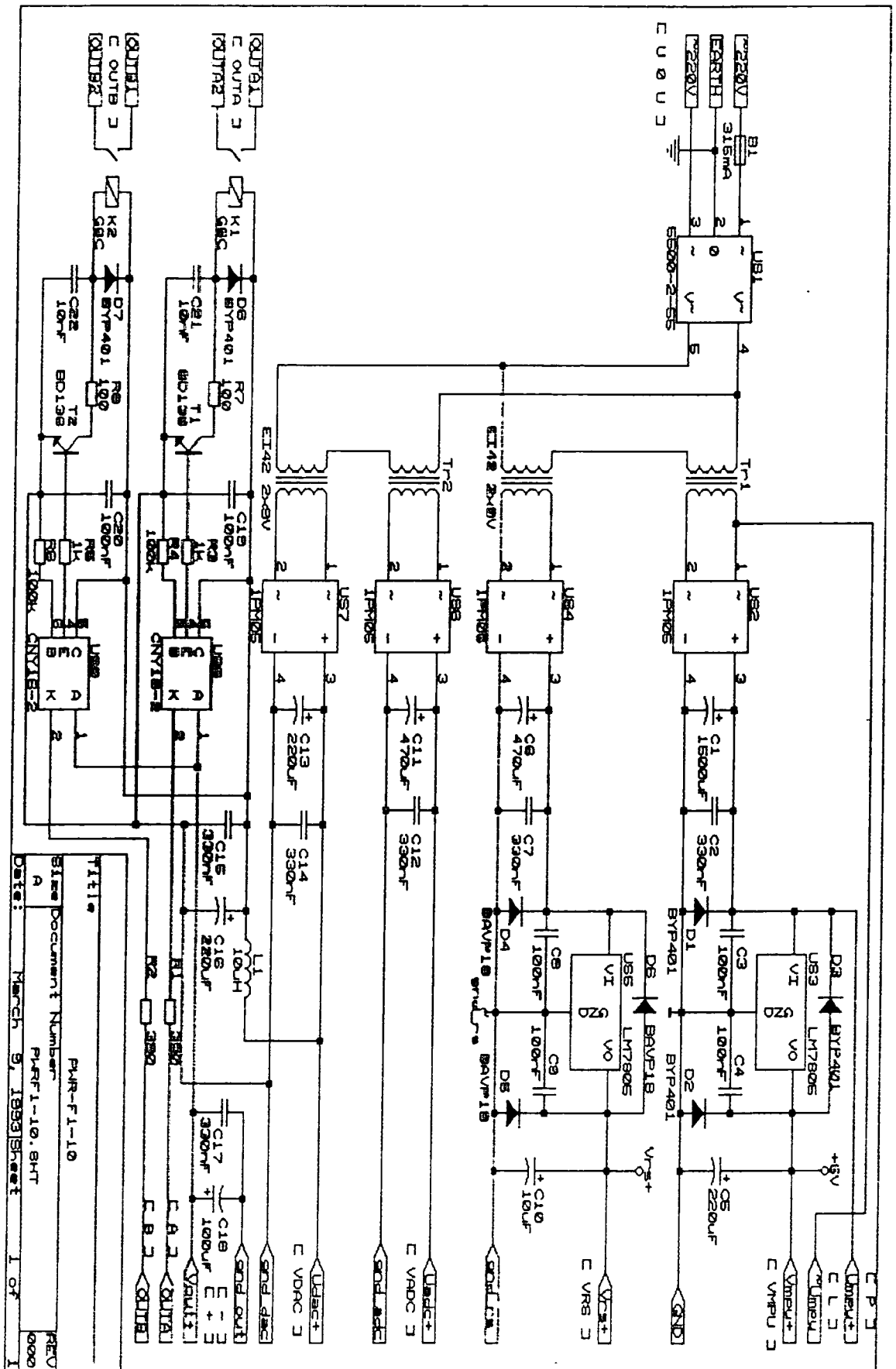
ozn.	funkcja
Vrs+	+5Vdc
Vrs-	gnd_rs

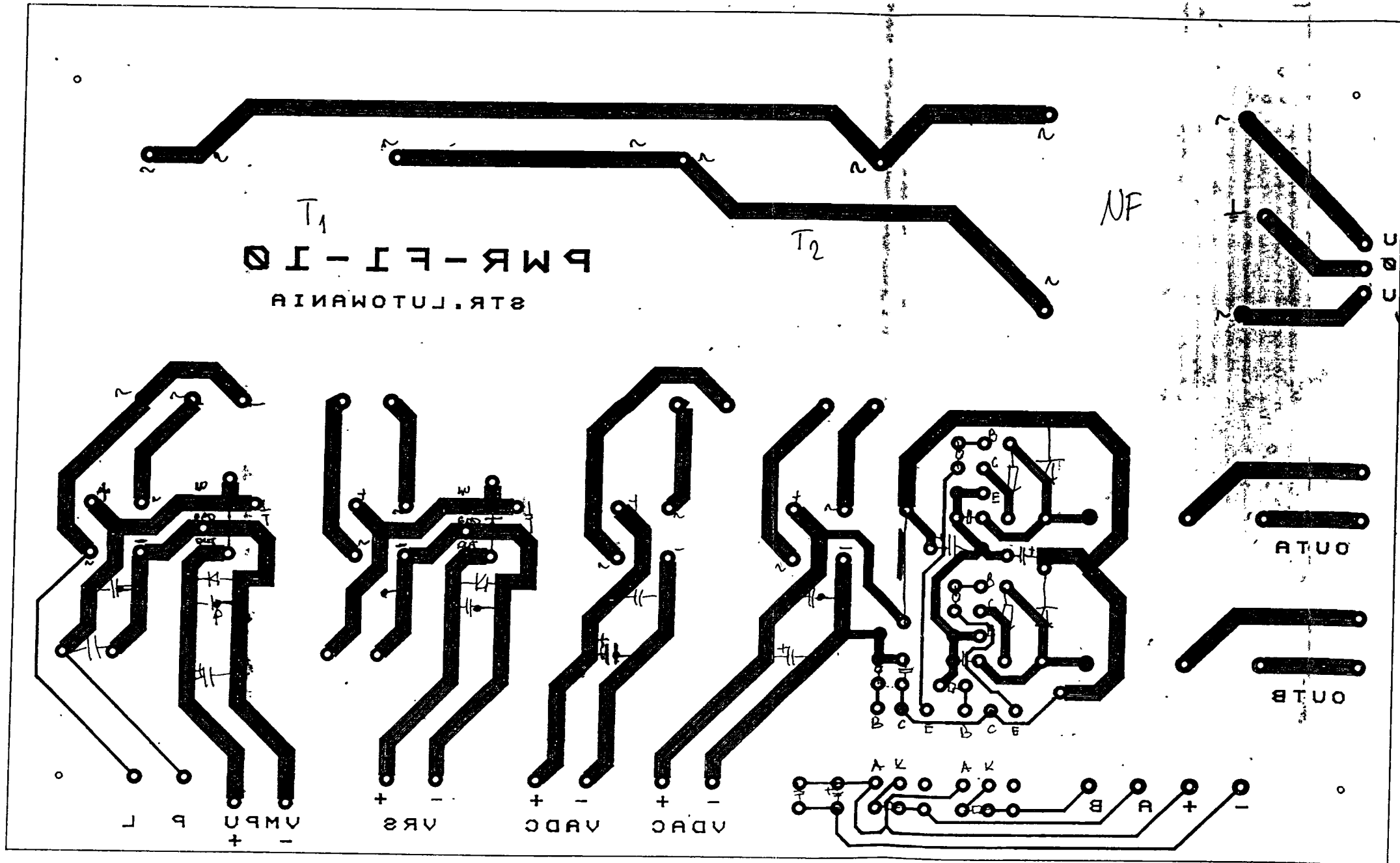
Tabela 2.7. Złącze zasilania jednostki centralnej modułu MPU-F1-10 (VMPU)

ozn.	funkcja
Vmpu+	+5Vdc
Vmpu-	GND
P	Umpu+ (PFI MAX694)
L	~Umpu (LINE MC68HC68T1)

WYKAZ ELEMENTÓW MODUŁU PWR-F1-10

L.p.	Liczba	Oznaczenie	Typ
1.	1	US1	5500-2-55
2.	4	US2, US4, US6, US7	1PM05
3.	2	US3, US5	LM7805
4.	2	US8, US9	CNY16-2
5.	2	T1, T2	BD136
6.	5	D1, D2, D3, D6, D7	BYP401
7.	3	D4, D5, D6	BAVP18
8.	2	R1, R2	390
9.	2	R3, R5	1k
10.	2	R4, R6	100k
11.	2	R7, R8	100
12.	1	C1	1500uF
13.	6	C2, C7, C12, C14, C15, C17	330nF
14.	6	C3, C4, C8, C9, C19, C20	100nF
15.	3	C5, C13, C16	220uF
16.	2	C6, C11	470uF
17.	1	C10	10uF
18.	1	C18	100uF
19.	2	C21, C22	10nF
20.	2	K1, K2	G6C
21.	1	L1	10uH
22.	2	Tr1, Tr2	EI38/EI42 2x9V
23.	1	B1	315mA





**MODUŁ KLAWIATURY FUNKCYJNEJ 1x8**

**KBD-F1-10**

## 1. CHARAKTERYSTYKA MODUŁU KLAWIATURY FUNKCYJNEJ KBD-F1-10

Moduł klawiatury funkcyjnej składa się z ośmiu przycisków monostabilnych o wspólnej linii (COM), zwartej do masy (GND). Klawiatura dostosowana jest do obudów tablicowych NGS96, standardu DIN43700. Umożliwia stosowanie wyświetlaczy alfanumerycznych LCD o wymiarach 84x44mm max.

Tabela 1.1. Układ klawiszy klawiatury KBD-F1-10

---

K7	K6	K5	K4	(widok od strony elementów, tzn. od frontu obudowy)
K0	K1	K2	K3	

---

Tabela 1.2. Złącze klawiatury KBD-F1-10 (J1)

---

nr_a.	sygnał	nr_b	sygnał
0a.	GND (COM)	0b.	GND (COM)
1a.	K7	1b.	GND (COM)
2a.	K5	2b.	K6
3a.	K3	3b.	K4
4a.	K1	4b.	K2
5a.	K0	5b.	GND (COM)
6a.	GND (COM)	6b.	GND (COM)

---

Tabela 1.3. Kody sprzętowe klawiatury KBD-F1-10

---

K0	0FEh	(PE0)	Kody podano przy założeniu, że linie zwrotne klawiatury (K0-7) dołączone są do portu PE0-7 MPU, a linia wspólna (COM) do masy MPU-F1-10 (GND).
K1	0FDh	(PE1)	
K2	0FBh	(PE2)	
K3	0F7h	(PE3)	
K4	0EFh	(PE4)	
K5	0DFh	(PE5)	
K6	0BFh	(PE6)	
K7	07Fh	(PE7)	

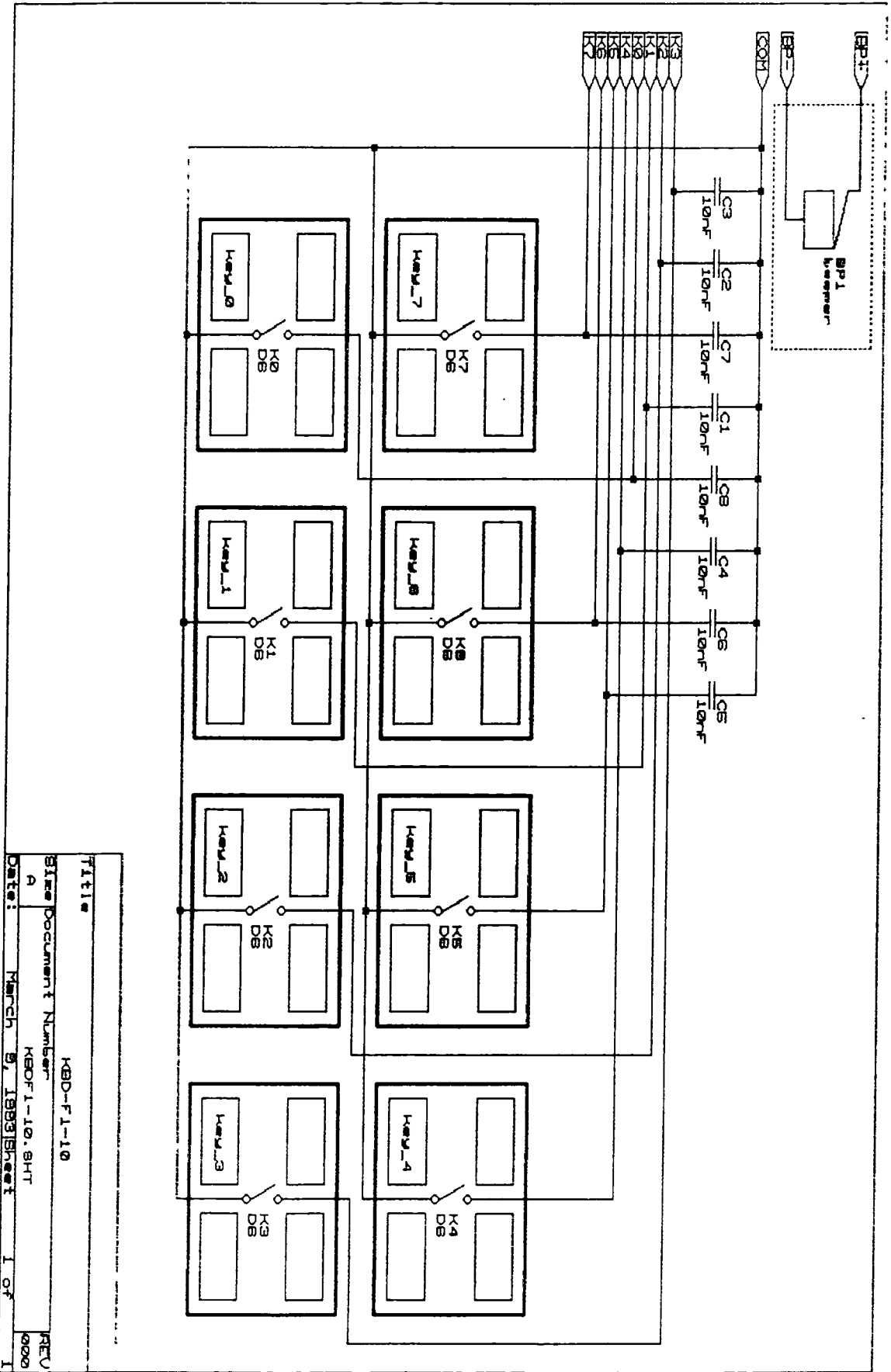
---

WYKAZ ELEMENTÓW MODUŁU PCF-F1-10

---

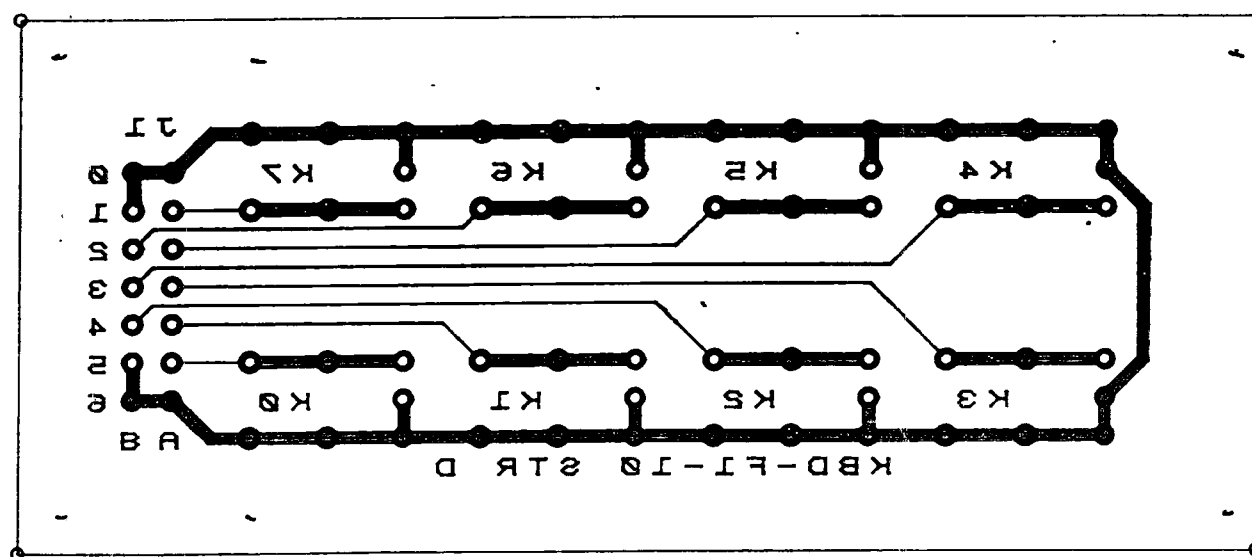
L.p.	Liczba	Opis	Typ
1.	8	C1, C2, C3, C4, C5, C6, C7, C8	10rF
2.	8	K0, K1, K2, K3, K4, K5, K6, K7	D6
3.	1	J1	JDIL16
4.	1	BP1	beeper

---



Title: KBD-F1-10  
 Size Document Number: KBOF1-10.SHT  
 A  
 Date: March 9, 1993 Sheet 1 of 1  
 REV 0000





35.5 -  
 max 30

**MODUŁ WYŚWIETLACZA LCD 5X7seg  
I KLAWIATURY FUNKCYJNEJ 1x6**

**LCD-F1-10**

## 1. CHARAKTERYSTYKA MODUŁU WYŚWIETLACZA LCD-F1-10

Moduł wyświetlacza ciekłokrystalicznego LCD posiada 7 segmentowy wyświetlacz LCD o sterowaniu bezpośrednim (3.5/4.5/5/8 cyfr) i sześć przycisków monostabilnych o wspólnej linii (COM), zwartej do masy (GND).

Moduł dostosowany jest do obudów tablicowych NGS96, standardu DIN43700.

### 1.1. Zespół wyświetlacza ciekłokrystalicznego

Projekt druku umożliwia zastosowanie dowolnego wyświetlacza LCD o sterowaniu bezpośrednim (od 3.5 do 8 cyfr). Wymagane jest dołączenie linii BP do właściwych odprowadzeń wyświetlacza, po rozcięciu ścieżki sterowania segmentem. Moduł zawiera dwa sterowniki wyświetlaczy ciekłokrystalicznych PCF2112, umożliwiające sterowanie 64 segmentami wyświetlacza i układ generacji napięcia LCD.

Tabela 1.1. Złącze wyświetlacza modułu LCD-F1-10 (J1)

nr	sygnał	
1.	Vdd (+5V)	upper - sterownik obsługujący
2.	GND	górną część segmentów
3.	DATA	wyświetlacza
4.	CLB	lower - sterownik obsługujący
5.	DLEN1 (upper)	dolną część segmentów
6.	DLEN2 (lower)	wyświetlacza

Wyświetlacz fizycznie dzieli się na dwie części, górną i dolną, zgodnie z wyprowadzeniami segmentów wyświetlacza. Każdą część obsługuje odrębny sterownik PCF2112. Przypisanie segmentów wyświetlacza i sterowników zależy sposobu wlutowania wyświetlacza.

## 1.2. Zespół klawiatury funkcyjnej modułu LCD-F1-10

Tabela 1.2. Układ klawiatury modułu LCD-F1-10

---

K5	K4	K3	K2	K1	K0	(widok od strony lutowania, tzn. od frontu obudowy)
----	----	----	----	----	----	-----------------------------------------------------

---

Tabela 1.3. Złącze klawiatury modułu LCD-F1-10 (J2)

---

nr_a.	sygnał	nr_b	sygnał
0a.	GND (COM)	0b.	GND (COM)
2a.	K5	2b.	K6
3a.	K3	3b.	K4
4a.	K1	4b.	K2

---

Tabela 1.4. Kody sprzętowe klawiatury KBD-F1-10 (ORAA #C0h)

---

K0	0FEh	(PE0)	Kody podano przy założeniu, że linie zwrotne klawiatury (K0-7) dołączone są do portu PE0-7 MPU, a linia wspólna (COM) do masy MPU-F1-10 (GND). Najstarsze bity portu klawiatury są maskowane poprzez ustawienie 1.
K1	0FDh	(PE1)	
K2	0FBh	(PE2)	
K3	0F7h	(PE3)	
K4	0EFh	(PE4)	
K5	0DFh	(PE5)	

---

Tabela 1.5. Kody sprzętowe klawiatury KBD-F1-10 (ANDA #3Fh)

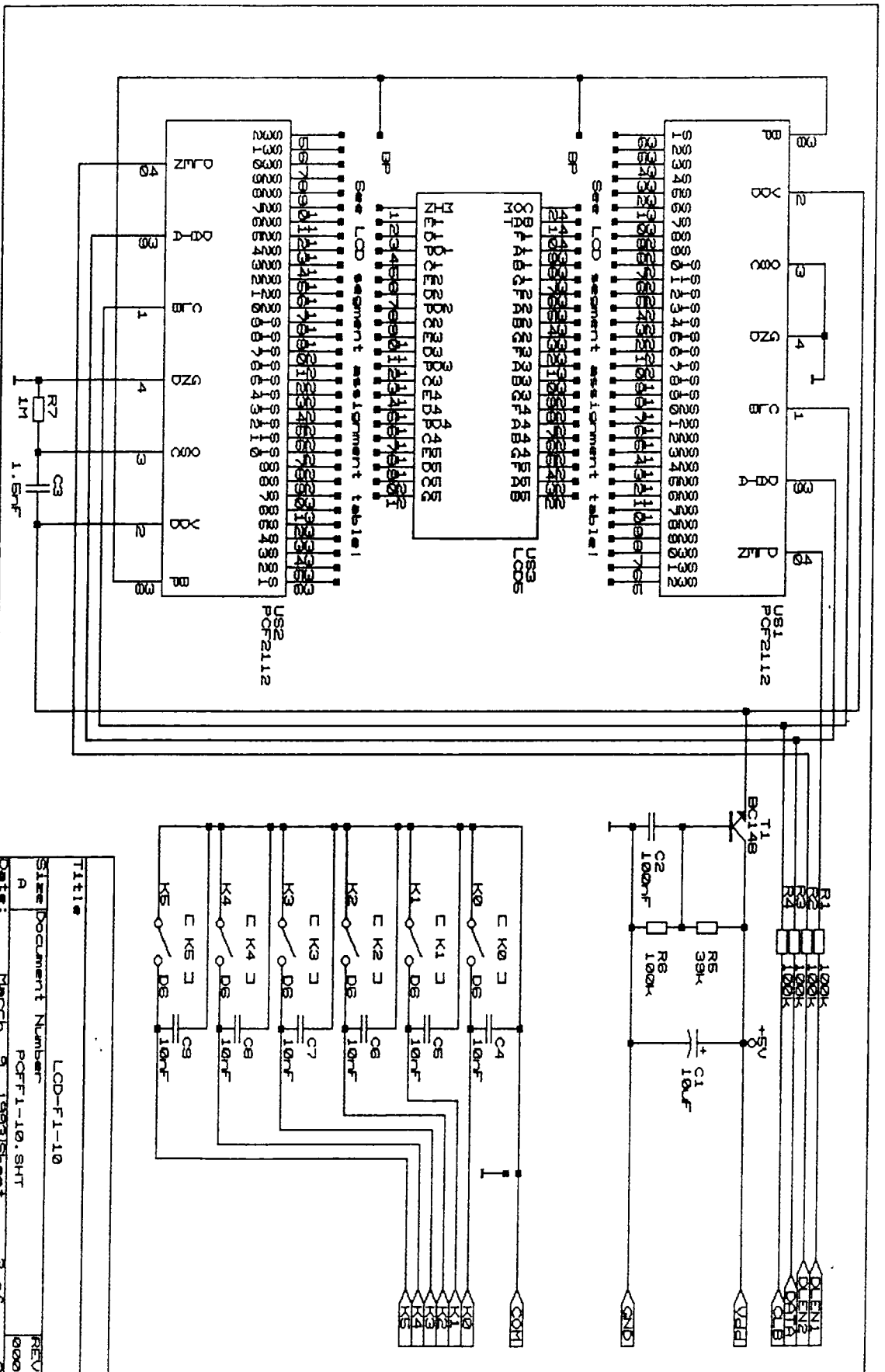
---

K0	0FEh	(PE0)	Kody podano przy założeniu, że linie zwrotne klawiatury (K0-7) dołączone są do portu PE0-7 MPU, a linia wspólna (COM) do masy MPU-F1-10 (GND). Najstarsze bity portu klawiatury są zerowane.
K1	0FDh	(PE1)	
K2	0FBh	(PE2)	
K3	0F7h	(PE3)	
K4	0EFh	(PE4)	
K5	0DFh	(PE5)	

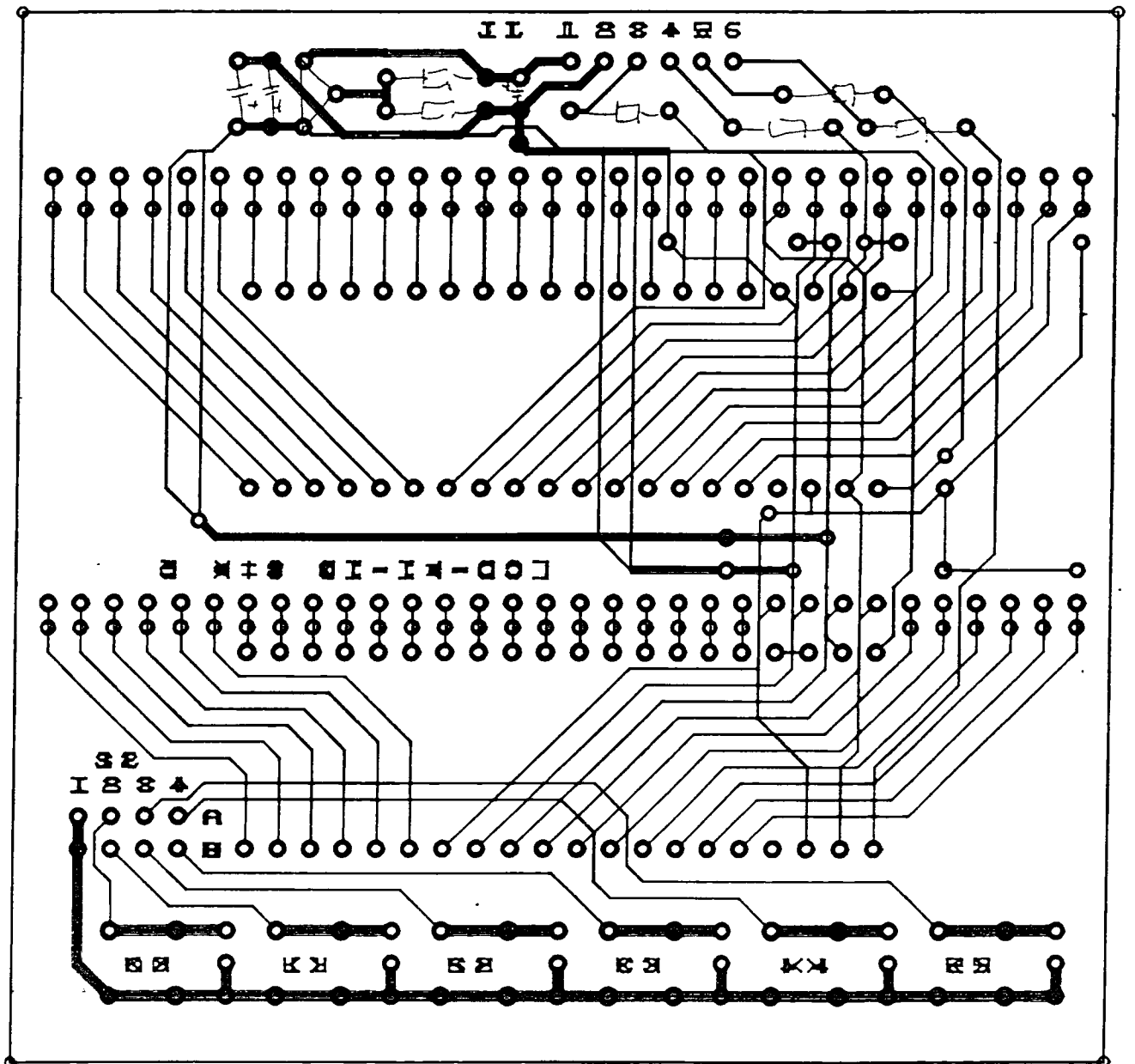
---

... AC ELEMENTOW MODULL PCF-F1-10

L.p.	Liczba	Opiszenie	Typ
1.	2	JS1, JS2	PCF2112
2.	1	LED	LCDS
3.	1	T1	BC148
4.	5	R1, R2, R3, R4, R6	100k
5.	1	R5	39k
6.	1	R7	1M
7.	1	C1	10uF
8.	1	C2	100nF
9.	1	C3	1.5nF
10.	6	C4, C5, C6, C7, C8, C9	10nF
11.	6	K0, K1, K2, K3, K4, K5	D6
12.	1	J1	J5IL6
13.	1	J2	JDIL14



Title: LOD-F1-10  
 Size Document Number: PCFF1-10.SHT  
 A  
 Date: March 9, 1993 Sheet 2 of 2  
 REV 0000



**S1766**

„Opracowanie modułów pomiarowych dotykowego pomiaru grubości do systemu pomiarowego o architekturze rozproszonej”

**ZAŁĄCZNIK Nr 4.**

Wydruk oprogramowania miernika tablicowego  
do pomiaru przemieszczeń liniowych



; file updated: 94-03-30

```
;  
;# MC68HC11F1 ##### MC68HC11F1 #  
;# MC68HC11F1 ##### MC68HC11F1 #  
;#  
;# MPU CONFIGURATION #  
;#  
;# hc11f1.mpu #  
;#  
;# MC68HC11F1 ##### MC68HC11F1 #  
;# MC68HC11F1 ##### MC68HC11F1 #
```

```
;  
;# MC68HC11F1 ##### MC68HC11F1 #  
;#  
;# SPECIAL FUNCTION REGISTERS ALLOCATION #  
;#  
;# MC68HC11F1 ##### MC68HC11F1 #
```

```
;  
; THE FOLLOWING STATEMENTS DEFINE THE SPECIAL FUNCTION REGISTERS, FOR C16  
; ASSEMBLER, IN ALPHABETICAL ORDER WHERE THEY ARE LOCATED AFTER A RESET.
```

```
;  
;# MC68HC11F1 ##### MC68HC11F1 #
```

```
PORTA: equ 1000h ; PORT A DATA REGISTER  
DDRA: equ 1001h ; DATA DIRECTION REG FOR PORT A  
PORTG: equ 1002h ; PORT G DATA REGISTER  
DDRG: equ 1003h ; DATA DIRECTION REG FOR PORT G  
PORTB: equ 1004h ; PORT B DATA REGISTER  
PORTF: equ 1005h ; PORT F DATA REGISTER  
PORTC: equ 1006h ; PORT C DATA REGISTER  
DDRC: equ 1007h ; DATA DIRECTION REG FOR PORT C  
PORTD: equ 1008h ; PORT D DATA REGISTER  
DDRD: equ 1009h ; DATA DIRECTION REG FOR PORT D  
PORTE: equ 100Ah ; PORT E DATA REGISTER  
  
CFORC: equ 100Bh ; TIMER COMPARE FORCE REGISTER  
OC1M: equ 100Ch ; OUTPUT COMPARE 1 MASK REGISTER  
OC1D: equ 100Dh ; OUTPUT COMPARE 1 DATA REGISTER  
TCNT: equ 100Eh ; TIMER COUNTER REGISTER (WORD)  
TIC1: equ 1010h ; TIMER INPUT CAPTURE REGISTER 1  
TIC2: equ 1012h ; TIMER INPUT CAPTURE REGISTER 2  
TIC3: equ 1014h ; TIMER INPUT CAPTURE REGISTER 3  
TOC1: equ 1016h ; TIMER OUTPUT COMPARE REG 1  
TOC2: equ 1018h ; TIMER OUTPUT COMPARE REG 2  
TOC3: equ 101Ah ; TIMER OUTPUT COMPARE REG 3  
TOC4: equ 101Ch ; TIMER OUTPUT COMPARE REG 4  
TOC5: equ 101Eh ; TIMER OUTPUT COMPARE REG 5  
TCTL1: equ 1020h ; TIMER CONTROL REGISTER 1  
TCTL2: equ 1021h ; TIMER CONTROL REGISTER 2  
TMSK1: equ 1022h ; MAIN TIMER INT MASK REGISTER 1  
TFLG1: equ 1023h ; MAIN TIMER INT. FLAG REG 1  
TMSK2: equ 1024h ; MAIN TIMER INT MASK REGISTER 2  
TFLG2: equ 1025h ; MAIN TIMER INT. FLAG REG 2  
PACTL: equ 1026h ; PULSE ACCUMULATOR CONTROL REG  
PACNT: equ 1027h ; PULSE ACCUMULATOR COUNT REG  
  
SPCR: equ 1028h ; SPI CONTROL REGISTER  
SPSR: equ 1029h ; SPI STATUS REGISTER  
SPDR: equ 102Ah ; SPI DATA REGISTER  
BAUD: equ 102Bh ; SCI BAUD RATE CONTROL REGISTER  
SCCR1: equ 102Ch ; SCI CONTROL REGISTER 1  
SCCR2: equ 102Dh ; SCI CONTROL REGISTER 2  
SCSR: equ 102Eh ; SCI STATUS REGISTER
```

```

SCDR:      equ      102Fh      ; SCI DATA REGISTER

ADCTL:    equ      1030h      ; A/D CONTROL/STATUS REGISTER
ADR1:     equ      1031h      ; A/D RESULT REGISTER 1
ADR2:     equ      1032h      ; A/D RESULT REGISTER 2
ADR3:     equ      1033h      ; A/D RESULT REGISTER 3
ADR4:     equ      1034h      ; A/D RESULT REGISTER 4

BPROT:    equ      1035h      ; EEPROM BLOCK PROTECT REGISTER
;         equ      1036h      ; RESERVED
;         equ      1037h      ; RESERVED
OPT2:     equ      1038h      ; SYSTEM CONFIGURATION OPTIONS 2 REGISTER
OPTION:   equ      1039h      ; SYSTEM CONFIGURATION OPTIONS
COPRST:   equ      103Ah      ; RESET COP TIMER CIRCUITRY
PPROG:    equ      103Bh      ; EEPROM PROGRAMMING REGISTER
HPRIO:    equ      103Ch      ; HIGHEST PRIORITY INTERRUPT
INIT:     equ      103Dh      ; RAM AND I/O MAPPING REGISTER
TEST1:    equ      103Eh      ; FACTORY TEST CONTROL REGISTER
CONFIG:   equ      103Fh      ; CONFIGURATION CONTROL REGISTER

;         equ      1040h      ; RESERVED
;         equ      105Dh      ; RESERVED

CSSTRH:   equ      105Ch      ; CHIP SELECT CLOCK STRECH REGISTER
CSCTL:    equ      105Dh      ; CHIP SELECT CONTROL REGISTER
CSGADR:   equ      105Eh      ; GENERAL-PURPOSE CHIP SELECT ADDRESS REGISTER
CSGSIZ:   equ      105Fh      ; GENERAL-PURPOSE CHIP SELECT SIZE REGISTER

;# MC68HC11F1 ##### MC68HC11F1 #
;#
;#          FLAGS IN SPECIAL FUNCTION REGISTERS
;#
;# MC68HC11F1 ##### MC68HC11F1 #

bit_0:    equ      01h      ; bit0   inside byte
bit_1:    equ      02h      ; bit1
bit_2:    equ      04h      ; bit2
bit_3:    equ      08h      ; bit3
bit_4:    equ      10h      ; bit4
bit_5:    equ      20h      ; bit5
bit_6:    equ      40h      ; bit6
bit_7:    equ      80h      ; bit7

PA0:      equ      01h      ; bit0   in PORTA
PA1:      equ      02h      ; bit1
PA2:      equ      04h      ; bit2
PA3:      equ      08h      ; bit3
PA4:      equ      10h      ; bit4
PA5:      equ      20h      ; bit5
PA6:      equ      40h      ; bit6
PA7:      equ      80h      ; bit7

PD0:      equ      01h      ; bit0   in PORTD
PD1:      equ      02h      ; bit1
PD2:      equ      04h      ; bit2
PD3:      equ      08h      ; bit3
PD4:      equ      10h      ; bit4
PD5:      equ      20h      ; bit5
; 0      equ      40h      ; bit6
; 0      equ      80h      ; bit7

PE0:      equ      01h      ; bit0   in PORTE
PE1:      equ      02h      ; bit1
PE2:      equ      04h      ; bit2

```

```

PE3:      equ  08h  ; bit3
PE4:      equ  10h  ; bit4
PE5:      equ  20h  ; bit5
PE6:      equ  40h  ; bit6
PE7:      equ  80h  ; bit7

PF0:      equ  01h  ; bit0  in PORTF
PF1:      equ  02h  ; bit1
PF2:      equ  04h  ; bit2
PF3:      equ  08h  ; bit3
PF4:      equ  10h  ; bit4
PF5:      equ  20h  ; bit5
PF6:      equ  40h  ; bit6
PF7:      equ  80h  ; bit7

PG0:      equ  01h  ; bit0  in PORTG
PG1:      equ  02h  ; bit1
PG2:      equ  04h  ; bit2
PG3:      equ  08h  ; bit3
PG4:      equ  10h  ; bit4
PG5:      equ  20h  ; bit5
PG6:      equ  40h  ; bit6
PG7:      equ  80h  ; bit7

FOC1:     equ  80h  ; flags in CFORC
FOC2:     equ  40h  ;
FOC3:     equ  20h  ;
FOC4:     equ  10h  ;
FOC5:     equ  08h  ;
; 0      equ  04h  ;
; 0      equ  02h  ;
; 0      equ  01h  ;

OC1M7:    equ  80h  ; flags in OC1M
OC1M6:    equ  40h  ;
OC1M5:    equ  20h  ;
OC1M4:    equ  10h  ;
OC1M3:    equ  08h  ;
; 0      equ  04h  ;
; 0      equ  02h  ;
; 0      equ  01h  ;

OC1D7:    equ  80h  ; flags in OC1D
OC1D6:    equ  40h  ;
OC1D5:    equ  20h  ;
OC1D4:    equ  10h  ;
OC1D3:    equ  08h  ;
; 0      equ  04h  ;
; 0      equ  02h  ;
; 0      equ  01h  ;

OM2:      equ  80h  ; flags in TCTL1
OL2:      equ  40h  ;
OM3:      equ  20h  ;
OL3:      equ  10h  ;
OM4:      equ  08h  ;
OL4:      equ  04h  ;
OM5:      equ  02h  ;
OL5:      equ  01h  ;

EDG4B:    equ  80h  ; flags in TCTL2
EDG4A:    equ  40h  ;
EDG1B:    equ  20h  ;
EDG1A:    equ  10h  ;

```

```

EDG2B:      equ  08h  ;
EDG2A:      equ  04h  ;
EDG3B:      equ  02h  ;
EDG3A:      equ  01h  ;

OC1I:      equ  80h  ; flags in TMSK1
OC2I:      equ  40h  ;
OC3I:      equ  20h  ;
OC4I:      equ  10h  ;
OC5I:      equ  08h  ;
IC4I:      equ  08h  ;
IC1I:      equ  04h  ;
IC2I:      equ  02h  ;
IC3I:      equ  01h  ;

OC1F:      equ  80h  ; flags in TFLG1
OC2F:      equ  40h  ;
OC3F:      equ  20h  ;
OC4F:      equ  10h  ;
OC5F:      equ  08h  ;
IC4F:      equ  08h  ;
IC1F:      equ  04h  ;
IC2F:      equ  02h  ;
IC3F:      equ  01h  ;

TOI:       equ  80h  ; flags in TMSK2
RTII:      equ  40h  ;
PAOVI:     equ  20h  ;
PAII:      equ  10h  ;
; 0       equ  80h  ;
; 0       equ  40h  ;
PR1:      equ  02h  ;
PR0:      equ  01h  ;

TOF:       equ  80h  ; flags in TFLG2
RTIF:      equ  40h  ;
PAOVF:     equ  20h  ;
PAIF:      equ  10h  ;
; 0       equ  08h  ;
; 0       equ  04h  ;
; 0       equ  02h  ;
; 0       equ  01h  ;

; 0       equ  80h  ; flags in PACTL
PAEN:      equ  40h  ;
PAMOD:     equ  20h  ;
PEDGE:     equ  10h  ;
DDRA3:     equ  08h  ;
I4O5:     equ  04h  ;
RTR1:     equ  02h  ;
RTR0:     equ  01h  ;

SPIE:      equ  80h  ; flags in SPCR
SPE:       equ  40h  ;
DWOM:      equ  20h  ;
MSTR:      equ  10h  ;
CPOL:      equ  08h  ;
CPHA:      equ  04h  ;
SPR1:      equ  02h  ;
SPR0:      equ  01h  ;

SPIF:      equ  80h  ; flags in SPSR
WCOL:      equ  40h  ;
; 0       equ  20h  ;

```

```

MODF:      equ 10h ;
; 0       equ 08h ;
; 0       equ 04h ;
; 0       equ 02h ;
; 0       equ 01h ;

TCLR:      equ 80h ; flags in BAUD
; 0       equ 40h ;
SCP1:      equ 20h ;
SCP0:      equ 10h ;
RCKB:      equ 08h ;
SCR2:      equ 04h ;
SCR1:      equ 02h ;
SCR0:      equ 01h ;

R8:        equ 80h ; flags in SCCR1
T8:        equ 40h ;
; 0       equ 20h ;
M:         equ 10h ;
WAKE:      equ 08h ;
; 0       equ 04h ;
; 0       equ 02h ;
;         equ 01h ;

TIE:       equ 80h ; flags in SCCR2
TCIE:      equ 40h ;
RIE:       equ 20h ;
ILIE:      equ 10h ;
TE:        equ 08h ;
RE:        equ 04h ;
RWU:       equ 02h ;
SBK:       equ 01h ;

TDRE:      equ 80h ; flags in SCSR
TC:        equ 40h ;
RDRF:      equ 20h ;
IDLE:      equ 10h ;
OR_f:      equ 08h ;
NF:        equ 04h ;
FE:        equ 02h ;
; 0       equ 01h ;

CU1:       equ 80h ; flags in ADCTL
; 0       equ 40h ;
SCAN:      equ 20h ;
MULT:      equ 10h ;
CD:        equ 08h ;
CC:        equ 04h ;
CB:        equ 02h ;
CA:        equ 01h ;

;         equ 80h ; flags in BPROT
;         equ 40h ;
;         equ 20h ;
PTCON:     equ 10h ;
BPRT3:     equ 08h ;
BPRT2:     equ 04h ;
BPRT1:     equ 02h ;
BPRT0:     equ 01h ;

GWOM:      equ 80h ; flags in OPT2
CWOM:      equ 40h ;
CLK4X:     equ 20h ;
; 0       equ 10h ;

```

```

; 0          equ 08h ;
; 0          equ 04h ;
; 0          equ 02h ;
; 0          equ 01h ;

ADPU:        equ 80h ; flags in OPTION
CSEL:        equ 40h ;
IRQE:        equ 20h ;
DLY:         equ 10h ;
CME:         equ 08h ;
FCME:        equ 04h ;
CR1:         equ 02h ;
CR0:         equ 01h ;

ODD:         equ 80h ; flags in PROG
EVEN:        equ 40h ;
; 0          equ 20h ;
BYTE:        equ 10h ;
ROW:         equ 08h ;
ERASE:       equ 04h ;
EELAT:       equ 02h ;
EEPGM:       equ 01h ;

REGUT:       equ 80h ; flags in HRPIO
SMOD:        equ 40h ;
MDA:         equ 20h ;
IRV:         equ 10h ;
PSEL3:       equ 08h ;
PSEL2:       equ 04h ;
PSEL1:       equ 02h ;
PSEL0:       equ 01h ;

RAM3:        equ 80h ; flags in INIT
RAM2:        equ 40h ;
RAM1:        equ 20h ;
RAM0:        equ 10h ;
REG3:        equ 08h ;
REG2:        equ 04h ;
REG1:        equ 02h ;
REG0:        equ 01h ;

T  AP:       equ 80h ; flags in TEST1
;           equ 40h ;
OCCR:       equ 20h ;
CBYP:       equ 10h ;
DISR:       equ 08h ;
FCM:        equ 04h ;
FCOP:       equ 02h ;
;           equ 01h ; TCON

EE3:        equ 80h ; flags in CONFIG
EE2:        equ 40h ;
EE1:        equ 20h ;
EE0:        equ 10h ;
; 1         equ 08h ; NOSEC
NOCOP:      equ 04h ;
; 1         equ 02h ; ROMON
EEON:       equ 01h ;

IO1SA:      equ 80h ; flags in CSSTRH
IO1SB:      equ 40h ;
IO2SA:      equ 20h ;
IOASB:      equ 10h ;
GSTHA:      equ 08h ;

```

```

GSTHB:      equ  04h  ;
PSTHA:      equ  02h  ;
PSTHB:      equ  01h  ;

IO1EN:      equ  80h  ; flags in CSCTL
IO1PL:      equ  40h  ;
IO2EN:      equ  20h  ;
IOAPL:      equ  10h  ;
GCSPR:      equ  08h  ;
PCSEN:      equ  04h  ;
PSIZA:      equ  02h  ;
PSIZB:      equ  01h  ;

GA15:      equ  80h  ; flags in CSGADR
GA14:      equ  40h  ;
GA13:      equ  20h  ;
GA12:      equ  10h  ;
GA11:      equ  08h  ;
GA10:      equ  04h  ;
;          equ  02h  ;
;          equ  01h  ;

ICMIV:      equ  80h  ; flags in CSGSIZ
ICMIV:      equ  40h  ;
;          equ  20h  ;
GNPOL:      equ  10h  ;
GNVLD:      equ  08h  ;
GSIZA:      equ  04h  ;
GSIZB:      equ  02h  ;
GSIZC:      equ  01h  ;

;# MC68HC11F1 ##### MC68HC11F1 #
;#
;#          SPECIAL FUNCTION REGISTERS CONFIGURATION CONSTANTS          #
;#
;# MC68HC11F1 ##### MC68HC11F1 #

;# TCTL1 OM/OLx CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

oml2_dis:   equ  000h ; output compare disabled (ORA) OM2 - output mode
oml2_tgl:   equ  040h ; toggle OC2 output line (ORA) OL2 - output level
oml2_res:   equ  080h ; clear OC2 line to zero (ORA)
oml2_set:   equ  0C0h ; set OC2 line to line (ORA)
oml2_clr:   equ  03Fh ; reset OC2 bits mask (AND)

oml3_dis:   equ  000h ; output compare disabled (ORA) OM3 - output mode
oml3_tgl:   equ  010h ; toggle OC3 output line (ORA) OL3 - output level
oml3_res:   equ  020h ; clear OC3 line to zero (ORA)
oml3_set:   equ  030h ; set OC3 line to line (ORA)
oml3_clr:   equ  0CFh ; reset OC3 bits mask (AND)

oml4_dis:   equ  000h ; output compare disabled (ORA) OM4 - output mode
oml4_tgl:   equ  004h ; toggle OC4 output line (ORA) OL4 - output level
oml4_res:   equ  008h ; clear OC4 line to zero (ORA)
oml4_set:   equ  00Ch ; set OC4 line to line (ORA)
oml4_clr:   equ  0F3h ; reset OC4 bits mask (AND)

oml5_dis:   equ  000h ; output compare disabled (ORA) OM5 - output mode
oml5_tgl:   equ  010h ; toggle OC5 output line (ORA) OL5 - output level
oml5_res:   equ  020h ; clear OC5 line to zero (ORA)
oml5_set:   equ  030h ; set OC5 line to line (ORA)
oml5_clr:   equ  0CFh ; reset OC5 bits mask (AND)

```

```

;# TCTL2 EDGxAB CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

edg1_clr:      equ  0CFh ; reset EDG1AB bits mask (AND)
edg1_dis:      equ  000h ; capture disabled (ORA)
edg1_up:       equ  010h ; capture on rising edges only (ORA)
edg1_dn:       equ  020h ; capture on falling edges only (ORA)
edg1_evr:      equ  030h ; capture on any (rising & falling) edges only (ORA)

edg2_clr:      equ  0F3h ; reset EDG2AB bits mask (ANDA)
edg2_dis:      equ  000h ; capture disabled (ORA)
edg2_up:       equ  004h ; capture on rising edges only (ORA)
edg2_dn:       equ  008h ; capture on falling edges only (ORA)
edg2_evr:      equ  00Ch ; capture on any (rising & falling) edges only (ORA)

edg3_clr:      equ  0FCh ; reset EDG3AB bits mask (ANDA)
edg3_dis:      equ  000h ; capture disabled (ORA)
edg3_up:       equ  001h ; capture on rising edges only (ORA)
edg3_dn:       equ  002h ; capture on falling edges only (ORA)
edg3_evr:      equ  003h ; capture on any (rising & falling) edges only (ORA)

edg4_clr:      equ  03Fh ; reset EDG4AB bits mask (ANDA)
edg4_dis:      equ  000h ; capture disabled (ORA)
edg4_up:       equ  040h ; capture on rising edges only (ORA)
edg4_dn:       equ  080h ; capture on falling edges only (ORA)
edg4_evr:      equ  0C0h ; capture on any (rising & falling) edges only (ORA)

;# PACTL MODE/EDGE CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

pai_cnt_dn:    equ  000h ; PAI falling edge increments the counter
pai_cnt_up:    equ  010h ; PAI rising edge increments the counter
pai_inh_lo:    equ  020h ; Low on PAI inhibits counting
pai_inh_hi:    equ  030h ; High on PAI inhibits counting

;# SCI CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

baud_125k:     equ  000h ; 8MHz clock baud rate constants, SCP=00b
baud_62500:    equ  001h
baud_31250:    equ  002h
baud_15625:    equ  003h

baud_9600:     equ  030h ; 8MHz clock baud rate constants, SCP=11b
baud_4800:     equ  031h
baud_2400:     equ  032h
baud_1200:     equ  033h
baud_600:      equ  034h
baud_300:      equ  035h
baud_150:      equ  036h
baud_75:       equ  037h

;# CSTL CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

psiz_64k:      equ  000h ; to be OR'ed with CSTL
psiz_32k:      equ  001h
psiz_16k:      equ  002h
psiz_8k:       equ  003h

io1pl_hi:      equ  040h ; to be OR'ed with CSTL
io1pl_lo:      equ  0BFh ; to be AND'ed with CSTL
io2pl_hi:      equ  010h ; to be OR'ed with CSTL
io2pl_lo:      equ  0EFh ; to be AND'ed with CSTL

```



```

gcspr_hi:      equ  080h ; to be OR'ed with CSTL
gcspr_lo:      equ  0F7h ; to be AND'ed with CSTL

;# CSSIZ CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

; psiz_64k:    equ  000h ; to be OR'ed with CSGSIZ
; psiz_32k:    equ  001h
; psiz_16k:    equ  002h
; psiz_8k:     equ  003h
psiz_4k:       equ  004h
psiz_2k:       equ  005h
psiz_1k:       equ  006h
psiz_0k:       equ  007h ; disabled

io1av_hi:      equ  080h ; to be OR'ed with CSGSIZ
io1av_lo:      equ  07Fh ; to be AND'ed with CSGSIZ
io2av_hi:      equ  040h ; to be OR'ed with CSGSIZ
io2av_lo:      equ  0BFh ; to be AND'ed with CSGSIZ

gnp01_hi:      equ  010h ; to be OR'ed with CSGSIZ
gnp01_lo:      equ  0EFh ; to be AND'ed with CSGSIZ
gavld_hi:      equ  008h ; to be OR'ed with CSGSIZ
gavld_lo:      equ  0F7h ; to be AND'ed with CSGSIZ

;# CSSTRH CONFIGURATION CONSTANTS
;# MC68HC11F1 ##### MC68HC11F1 #

io1_strh_0:    equ  000h ; to be OR'ed with CSSTRH
io1_strh_1:    equ  040h
io1_strh_2:    equ  080h
io1_strh_3:    equ  0C0h
io1_strh_clr:  equ  03Fh ; to be AND'ed with CSSTRH (CLEAR settings)

io2_strh_0:    equ  000h ; to be OR'ed with CSSTRH
io2_strh_1:    equ  010h
io2_strh_2:    equ  020h
io2_strh_3:    equ  030h
io2_strh_clr:  equ  0CFh ; to be AND'ed with CSSTRH (CLEAR settings)

gen_strh_0:    equ  000h ; to be OR'ed with CSSTRH
gen_strh_1:    equ  040h
gen_strh_2:    equ  080h
gen_strh_3:    equ  0C0h
gen_strh_clr:  equ  0F3h ; to be AND'ed with CSSTRH (CLEAR settings)

pgm_strh_0:    equ  000h ; to be OR'ed with CSSTRH
pgm_strh_1:    equ  001h
pgm_strh_2:    equ  002h
pgm_strh_3:    equ  003h
pgm_strh_clr:  equ  0FCh ; to be AND'ed with CSSTRH (CLEAR settings)

;# hc11f1.mpu - END
;# MC68HC11F1 ##### MC68HC11F1 #

```

; file updated: 98-02-05

```

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#           MPU HC11F1 SUPERVISORY ROUTINE
;#
;#           adc35-10.src
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

```

```

CPU      "6811RW.TBL"      ; LOAD 6811RW TABLE
HOF      "INT8"

```

```

;# HARDWARE ALLOCATION BASE CONSTANTS
;# MC68HC11 ##### MC68HC11 #

```

```

regs_base:    equ 1000h      ; registers allocation base
ram_base:     equ 0000h      ; RAM allocation base
rom_base:     equ 0E000h     ; ROM allocation base
eeprom_base:  equ 0B000h     ; EEPROM allocation base
ext_ram_base: equ 0400h      ; external RAM allocation base

```

```

;# SOFTWARE CONFIGURATION VARIABLES
;# MC68HC11 ##### MC68HC11 #

```

```

sft_cnfg:    equ 00000h      ; assembly flow variable

```

```

;# INCLUDE MPU DATA FILES
;# MC68HC11 ##### MC68HC11 #

```

```

;-----
incl      "work11\sys11\hc11f1.mpu"
;-----

```

```

;# TASK/ROUTINES VARIABLES ALLOCATION SCHEME
;# MC68HC11 ##### MC68HC11 #
;#

```

```

;# pg0_tsk_vars: equ pg0_vars+1    ; task variables area - page 0
;# pg0_vars:     set pg0_vars+nnh  ; update allocation pointer
;# "
;# int_tsk_vars: equ int_vars+1    ; task variables area - internal RAM
;# int_vars:     set int_vars+nnh  ; update allocation pointer
;#
;# ext_tsk_vars: equ ext_vars+1    ; task variables area - external RAM
;# ext_vars:     set ext_vars+nnh  ; update allocation pointer
;#
;# code_alloc:   set $              ; program code area
;#              org code_alloc     ; restore allocation pointer
;#
;# rom_tsk_data: equ rom_data+1    ; task data area - EPROM
;# rom_data:     set rom_data+nnh  ; update allocation pointer
;#
;# eep_tsk_data: equ eep_data+1    ; task data area - EPROM
;# eep_data:     set eep_data+nnh  ; update allocation pointer
;#

```

```

;# MC68HC11 ##### MC68HC11 #

```

```

code_base:    equ rom_base+00h    ; program code allocation base
pg0_base:     equ ram_base+00h    ; page 0 RAM variables allocation base
int_base:     equ ram_base+100h   ; internal RAM variables allocation base
eep_base:     equ eeprom_base+00h ; EEPROM data allocation base

```

```

pg0_vars:      set pg0_base+00h      ; page 0 RAM variables allocation pointer
int_vars:      set int_base+00h      ; internal RAM variables allocation pointer
ext_vars:      set ext_base+00h      ; external RAM variables allocation pointer
rom_data:      set rom_base+00h      ; ROM data allocation pointer
eep_data:      set eep_base+00h      ; EEPROM data allocation pointer
code_alloc:    set code_base+00h     ; program code allocation pointer

sp_ini:        equ 003F0h            ; stack pointer value
sp_top:        equ 003F2h            ; top of stack absolute value
sp_bottom:     equ 003C0h            ; bottom of stack absolute value

;# SYSTEM VARIABLES
;# MC68HC11 ##### MC68HC11 #

PA_bf:        equ pg0_vars+00h      ; PORT A buffer
PD_bf:        equ pg0_vars+01h      ; PORT D buffer
PE_bf:        equ pg0_vars+02h      ; PORT E buffer
PG_bf:        equ pg0_vars+03h      ; PORT G buffer
trnsf_cnt:    equ pg0_vars+04h      ; set/clr/fill_mem_w loop counter
mode:         equ pg0_vars+05h      ; device/dsp mode buffer

pg0_vars:      set pg0_vars+05h      ; update allocation pointer

;# SYSTEM CONSTANTS
;# MC68HC11 ##### MC68HC11 #

mode_rst:     equ 0FFh              ; restart mode
mode_sys:     equ 000h              ; system mode

dta_no_val:   equ 0FFh              ; unmodified data value

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#          MPU HARDWARE CONFIGURATION
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;# ROM 27C64      - CSPROG : E000h to FFFFh, active low, ON
;# RAM 6264      - CSGEN  : 0000h to 7FFFh, active low, OFF
;# EEPROM (built in) : B000h to B1FFh, present
;# LCD E (E1)    - CSIO1  : 1060h to 17FFh, active low, ON, strh3
;# LCD E2        - CSIO2  : 1800h to 1FFFh, active low, OFF
;#
;# MC68HC11 ##### MC68HC11 #

;# MPU F1 HARDWARE CONFIGURATION COSTANTS
;# MC68HC11 ##### MC68HC11 #

CSSTRH_ini:   equ 0C0h              ; CHIP SELECT configuration constants
CSGADR_ini:   equ 000h
CSGSIZ_ini:   equ 007h
CSCTL_ini:    equ 087h

BPROT_ini:    equ 01Fh              ; HARDWARE configuration constants (64E)
OPT2_ini:     equ 000h
OPTION_ini:   equ 013h              ; (64E)
PPROG_ini:    equ 000h
HPRIO_ini:    equ 007h
INIT_ini:     equ 001h
CONFIG_ini:   equ 0BFh

PORTA_ini:    equ 000h              ; IO PORT's configuration constants

```

```

DDRA_ini:    equ    000h
PORTG_ini:   equ    0A0h
DDRG_ini:    equ    0A0h
PORTB_ini:   equ    000h
PORTF_ini:   equ    000h
PORTC_ini:   equ    000h
DDRC_ini:    equ    000h
PORTD_ini:   equ    000h
DDRD_ini:    equ    000h

CFORC_ini:   equ    000h    ; TIMER configuration constants
OC1M_ini:    equ    000h
OC1D_ini:    equ    000h
TIC1_ini:    equ    000h
TIC2_ini:    equ    000h
TIC3_ini:    equ    000h
TOC1_ini:    equ    000h
TOC2_ini:    equ    000h
TOC3_ini:    equ    000h
TOC4_ini:    equ    000h
TOC5_ini:    equ    000h
TCTL1_ini:   equ    000h
TCTL2_ini:   equ    000h
TMR1_ini:    equ    000h
TFLG1_ini:   equ    000h
TMSK2_ini:   equ    040h    ; (64E)
TFLG2_ini:   equ    000h
PACTL_ini:   equ    004h    ; PULSE ACCUMULATOR configuration constants (64E)
PACNT_ini:   equ    000h

SPCR_ini:    equ    000h    ; SPI configuration constants
SPSR_ini:    equ    000h
SPDR_ini:    equ    000h

BAUD_ini:    equ    000h    ; SCI configuration constants
SCCR1_ini:   equ    000h
SCCR2_ini:   equ    000h
SCSR_ini:    equ    000h
SCDR_ini:    equ    000h

ADCTL_ini:   equ    000h    ; ADC configuration constants

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#          SUPERVISORY ROUTINES          #
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

org code_base

;# CONFIGURE MPU ON POWER ON
;# MC68HC11 ##### MC68HC11 #

pwr_on:      lds    #sp_ini          ; initialize stack pointer

;# CONFIGURE MPU
;# MC68HC11 ##### MC68HC11 #

cnfg_mpu:    nop                    ; configure MPU

; cnfg_stack: lds    #sp_ini          ; initialize stack pointer

```

```

cnfg_mpu_cs:  nop                ; CONFIGURE MPU CHIP SELECTS
              ldaa  #CSSTRH_ini
              staa  CSSTRH
              ldaa  #CSGADR_ini
              staa  CSGADR
              ldaa  #CSGSIZ_ini
              staa  CSGSIZ
              ldaa  #CSCTL_ini
              staa  CSCTL

cnfg_mpu_hdw: nop                ; CONFIGURE MPU HARDWARE OPTIONS
              ldaa  #BPROT_ini    ; (64E)
              staa  BPROT
              ldaa  #OPT2_ini    ; (64E)?
              staa  OPT2
              ldaa  #OPTION_ini  ; (64E)
              staa  OPTION
              ldaa  #HPRIO_ini
              staa  HPRIO
              ldaa  #INIT_ini    ; (64E)
              staa  INIT

              ldaa  #0FFh        ; DELAY TO ELAPSE 64E PERIOD
cnfg_mpu_64:  deca
              bne   cnfg_mpu_64

cnfg_mpu_io:  nop                ; CONFIGURE IO
              ldaa  #DDRA_ini    ; configure PD
              staa  DDRA
              ldaa  #PORTA_ini   ; configure PA
              staa  PORTA        ; initialize PA
              staa  PA_bf
              ldaa  #DDRD_ini    ; configure PD
              staa  DDRD
              ldaa  #PORTD_ini   ; initialize PD
              staa  PORTD
              staa  PD_bf
              ldaa  #DDRG_ini    ; configure PG
              staa  DDRG
              ldaa  #PORTG_ini   ; initialize PG
              staa  PORTG
              staa  PG_bf

; cnfg_mpu_tmr: nop                ; CONFIGURE MPU TIMER SECTION
;
;       ldaa  #CFORC_ini
;       staa  CFORC
;
;       ldaa  #OC1M_ini
;       staa  OC1M
;
;       ldaa  #OC1D_ini
;       staa  OC1D
;
;       ldd  #TIC1_ini
;       std  TIC1
;
;       ldd  #TIC2_ini
;       std  TIC2
;
;       ldd  #TIC3_ini
;       std  TIC3
;
;       ldd  #TOC1_ini
;       std  TOC1
;
;       ldd  #TOC2_ini
;       std  TOC2
;
;       ldd  #TOC3_ini
;       std  TOC3
;
;       ldd  #TOC4_ini
;       std  TOC4

```

```

;         ldd    #TOC5_ini
;         std    TOC5
;         ldaa  #TCTL1_ini
;         staa  TCTL1
;         ldaa  #TCTL2_ini
;         staa  TCTL2
;         ldaa  #TMSK1_ini
;         staa  TMSK1
;         ldaa  #TFLG1_ini
;         staa  TFLG1
;         ldaa  #TMSK2_ini          ; (64E)
;         staa  TMSK2
;         ldaa  #TFLG2_ini
;         staa  TFLG2
;         ldaa  #PACTL_ini
;         staa  PACTL
;         ldaa  #PACNT_ini
;         staa  PACNT

;cnfg_mpu_spi: nop                ; CONFIGURE MPU SPI SECTION
;         ldaa  SPSR
;         ldaa  SPDR
;         ldaa  #SPCR_ini
;         staa  SPCR

;cnfg_mpu_sci: nop                ; CONFIGURE MPU SCI SECTION
;         ldaa  #BAUD_ini
;         staa  BAUD
;         ldaa  #SCCR1_ini
;         staa  SCCR1
;         ldaa  #SCCR2_ini
;         staa  SCCR2
;         ldaa  SCSR                ; clear SCI receive flags
;         ldaa  SCDR
;         ldaa  SCSR                ; clear SCI transmit flags
;         ldaa  #SCDR_ini
;         staa  SCDR

;cnfg_mpu_adc: nop                ; CONFIGURE MPU ADC SECTION
;         ldaa  #ADCTL_ini
;         staa  ADCTL

;cnfg_mpu_end: nop

;# INITIALIZE SOFTWARE ENVIRONMENT
;# MC68HC11 ##### MC68HC11 #

cnfg_hdw_dev: ldaa  #mode_rst      ; initialize restart mode
;         staa  mode
;         jsr   kbd_cnfg_drv      ; initialize kbd buffers
;         jsr   lcd_cnfg_drv     ; initialize lcd controller
;         ldaa  #dsp_src0        ; initialize dsp_source pointer
;         staa  dsp_source
;         ldy   #lcd_rst_rem     ; display power on restart remark
;         jsr   dsp_rem_ldr
;         jsr   lcd_data_ldr
;         jsr   sft_delay32
;         ldy   #lcd_pas_rem     ; display restart passed remark
;         jsr   dsp_l1_ldr
;         jsr   lcd_data_ldr
;         jsr   sft_delay32
;         ldaa  #dsp_src0        ; initialize dsp_source pointer
;         bset  sts_lcd, dsp_rq_f

```

```

cnfg_sft_dev: jsr   spvr_env_ini   ; initialize supervisory environment
              ldaa  #mode_sys     ; status variables
              staa  mode
              bset  adc_sts, adc_sync_rq ; enable adc_sync
              cli   ; enable interrupts

;# EXECUTION ROUTINE
;# MC68HC11 ##### MC68HC11 #

exe_drv:      jsr   cop_rst_drv

exe_drv1:     brclr sts_kbd, kbd_rq_f, exe_drv2      ; service kbd
              jsr   kbd_sel_drv

exe_drv2:     brclr adc_sts, adc_sync_rq, exe_drv3   ; service adc_sync
              ldaa #dsp_src1     ; initialize dsp_source pointer
              staa dsp_source
              jsr   adc_sync

exe_drv3:     brclr adc_sts, adc_curr_upd, exe_drv4   ; service adc_avrg
              jsr   adc_avrg
              ldaa #dsp_src1     ; initialize dsp_source pointer
              staa dsp_source
              bset  sts_lcd, dsp_rq_f

exe_drv4:     brclr sts_lcd, dsp_rq_f, exe_sys        ; service dsp_bf
              jsr   dsp_sel_drv

exe_sys:      nop
exe_l0:       brclr sts_lcd, lcd_l0_rq_f, exe_l1      ; output lcd line 0
              jsr   lcd_data_ldr
exe_l1:       brclr sts_lcd, lcd_l1_rq_f, exe_kbd     ; output lcd line 1
              jsr   lcd_data_ldr
exe_kbd:      brclr sts_kbd, kbd_vd_f, exe_end        ; define kbd code
              jsr   kbd_drv

exe_end:      jmp   exe_drv

;# rti ROUTINE
;# M68HC11 ##### M68HC11 #

r   _drv:     brset sts_lcd, lcd_ld_f, rti_drv1
              brset sts_lcd, lcd_cmd_f, rti_drv1
              bra   rti_drv2
rti_drv1:     jmp   rti_drv_end
rti_drv2:     nop
              jsr   kbd_scan_drv

rti_drv_end:  ldaa  #RTIF
              staa TFLG2
              rti

;# cop RESET ROUTINE
;# M68HC11 ##### M68HC11 #

cop_rst_drv:  nop
              ldaa #055h
              staa coprst
              ldaa #0AAh
              staa coprst

cop_rst_end:  rts

;# M68HC11 ##### M68HC11 #
;# M68HC11 ##### M68HC11 #

```

```

;#
;#
;#
;# M68HC11 ##### M68HC11 #
;# M68HC11 ##### M68HC11 #

;# DELAY ROUTINES
;# M68HC11 ##### M68HC11 #

sft_delay32:  ldx    #0FFFFh      ; SOFTWARE DELAY 32 ( approx. 0.8s )
              ldy    #00004h
sft_delay32yx:nop
              dex
              bne    sft_delay32yx
              dey
              bne    sft_delay32yx
              rts

sft_delay16:  ldx    #0FFFFh      ; SOFTWARE DELAY 16 ( approx. 0.2s )
sft_delay16x:nop
              dex
              bne    sft_delay16x
              rts

;# RELOAD MEMORY BYTES
;# M68HC11 ##### M68HC11 #

rld_mem_b:   ldaa   0,X           ; reload (B) bytes from (IX) to (IY)
              staa  0,Y
              inx                    ; adjust pointers
              iny
              decb                    ; test if all bytes copied
              bne   rld_mem_b
              rts

;# RELOAD MEMORY WORDS
;# M68HC11 ##### M68HC11 #

rld_mem_w:   ldd    0,X           ; reload (trnsf_cnt) words from (IX)
              std    0,Y           ; to (IY), number of words in trnsf_cnt
              inx                    ; adjust pointers
              iny
              dec    trnsf_cnt       ; test if all words copied
              bne   rld_mem_w
              rts

;# CLEAR/FILL MEMORY BYTES
;# M68HC11 ##### M68HC11 #

set_mem_b:   ldd    #0FFh         ; set B bytes (0FFh in A)
              bra   fill_mem_b
clr_mem_b:   clra                    ; clear B bytes (00h in A)
fill_mem_b:  staa   0,X           ; fill B bytes from A to (IX)
              inx                    ; adjust pointer
              decb                    ; test if all bytes copied
              bne   fill_mem_b
              rts

;# FILL MEMORY WORDS
;# M68HC11 ##### M68HC11 #

set_mem_w:   ldd    #0FFFFh       ; set (trnsf_cnt) bytes (0FFFFh in D)
              bra   fill_mem_w
clr_mem_w:   ldd    #00h           ; clear (trnsf_cnt) bytes (00h in D)

```



```

fill_mem_w:  std    0,X          ; fill (trnsf_cnt) words from D to (IX)
              inx                ; adjust pointer
              dec    trnsf_cnt   ; test if all words copied
              bne   rld_mem_w
              rts

```

```

;# M68HC11 ##### M68HC11 #
;# M68HC11 ##### M68HC11 #
;#
;#          INCLUDE ENVIRONMENT FILES
;#
;# M68HC11 ##### M68HC11 #
;# M68HC11 ##### M68HC11 #

```

```

incl "work11\sys11\mth-f1.drv"
incl "work11\sys11\ascii-f1.drv"
incl "work11\sys11\lcd2-f1.drv"
incl "work11\sys11\kb1x8-f1.drv"

```

```

incl "work11\adc35-10.dsp"
incl "work11\adc35-10.kbd"
incl "work11\adc35-10.icl"
i'   "work11\adc35-10.spv"

```

```

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#          INTERRUPT VECTORS TABLE
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

```

```

org 0FFC0h
dwm  spvr_rsvd_spu    ; 0FFC0 - reserved
dwm  spvr_rsvd_spu    ; 0FFC2 - reserved
dwm  spvr_rsvd_spu    ; 0FFC4 - reserved
dwm  spvr_rsvd_spu    ; 0FFC6 - reserved
dwm  spvr_rsvd_spu    ; 0FFC8 - reserved
dwm  spvr_rsvd_spu    ; 0FFCA - reserved
dwm  spvr_rsvd_spu    ; 0FFCC - reserved
dwm  spvr_rsvd_spu    ; 0FFCE - reserved
dwm  spvr_rsvd_spu    ; 0FFD0 - reserved
dwm  spvr_rsvd_spu    ; 0FFD2 - reserved
dwm  spvr_rsvd_spu    ; 0FFD4 - reserved

```

```

org 0FFD6h
dwm  spvr_sci_spu     ; 0FFD6 - SCI Serial System
dwm  spvr_spi_spu     ; 0FFD8 - SPI Serial Transfer Complete
dwm  spvr_pae_spu     ; 0FFDA - Pulse Accumulator Input Edge
;dwm  spvr_pao_spu     ; 0FFDC - Pulse Accumulator Overflow
dwm  adc_ov_irq       ; 0FFDC - Pulse Accumulator Overflow    ADC_ov
dwm  spvr_tof_spu     ; 0FFDE - Timer Overflow
dwm  spvr_i4o5_spu    ; 0FFE0 - Timer I4O5
dwm  spvr_oc4_spu     ; 0FFE2 - Timer Output Compare 4
dwm  spvr_oc3_spu     ; 0FFE4 - Timer Output Compare 3
dwm  spvr_oc2_spu     ; 0FFE6 - Timer Output Compare 2
dwm  spvr_oc1_spu     ; 0FFE0 - Timer Output Compare 1
dwm  spvr_ic3_spu     ; 0FFEA - Timer Input Capture 3
dwm  spvr_ic2_spu     ; 0FFEC - Timer Input Capture 2
;dwm  spvr_ic1_spu     ; 0FFEE - Timer Input Capture 1
dwm  adc_eoc_irq      ; 0FFEE - Timer Input Capture 1 ADC_eoc
DWM  rti_drv          ; 0FFF0 - Real Time Interrupt
dwm  spvr_irq_spu     ; 0FFF2 - IRQ Pin or Parallel IO Interrupt

```

```
dwm spvr_xrq_spu ; 0FFF4 - XIRQ Pin Non-Maskable Interrupt
dwm spvr_swi_spu ; 0FFF6 - SWI
dwm spvr_ill_spu ; 0FFF8 - Illegal Opcode Trap
dwm spvr_cop_spu ; 0FFFA - COP Failure ( Reset )
dwm spvr_cmf_spu ; 0FFFC - Clock Monitor Fail ( Reset )
dwm pwr_on ; 0FFFE - Reset
```

```

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;# ngs96-f1.src #
;#
;# END #
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
```

END

; file updated: 98-02-05

```
;;# MC68HC11 ##### MC68HC11 #
;;# MC68HC11 ##### MC68HC11 #
;;#
;;#
;;#          adc35-10.icl
;;#
;;#
;;# MC68HC11 ##### MC68HC11 #
;;# MC68HC11 ##### MC68HC11 #
```

```
;;# INTERRUPT VECTORS SHOULD BE SET TO:
;;# MC68HC11 ##### MC68HC11 #
```

```
;;# dwm   adc_ov_irq      ; 0FFDC - Pulse Accumulator Overflow
;;# dwm   adc_eoc_irq     ; 0FFEA - Timer Input Capture 3
```

```
;;# ADC HARDWARE ASSIGNMENTS
;;# MC68HC11 ##### MC68HC11 #
```

```
ad_ eoc_p:          equ   PORTA          ; adc hardware devices constants
ac_ eoc_pin:        equ   PA2 ;7135_busy (IC1)
adc_POL_pin:        equ   PA3 ;7135_pol  (IC4)
adc_CLK_pin:        equ   PA7 ;7135_clk  (PAI)
adc_POL_bit:        equ   bit_7
```

```
adc_ov_msk:         equ   paovf
adc_ov_cnfg:        equ   pai_cnt_up ; for input AND gate (08)
adc_eoc_msk:        equ   ic1f
adc_eoc_edg_clr:    equ   edg1_clr
adc_eoc_edg_cnfg:   equ   edg1_dn ; for input AND gate (08)
```

```
;;# ADC VARIABLES
;;# MC68HC11 ##### MC68HC11 #
```

```
adc_vars:           equ   pg0_vars      ; adc variables area begin

adc_rslt0:          equ   pacnt         ; adc result lo
adc_rslt1:          equ   adc_vars+00h  ; adc result hi
adc_cyc_cnt:        equ   adc_vars+01h  ; adc cycle counter
adc_avrg_cnt:       equ   adc_vars+02h  ; adc average counter
adc_sts:            equ   adc_vars+03h  ; adc status
adc_curr3:          equ   adc_vars+04h  ; adc current result (adc_bf average)
adc_curr2:          equ   adc_vars+05h
adc_curr1:          equ   adc_vars+06h
adc_curr0:          equ   adc_vars+07h
adc_bf_ptr:         equ   adc_vars+08h  ; adc_bf pointer
adc_POL_bf0:        equ   adc_vars+0Ah  ; adc_POL buffer for adc_bf_0,7
adc_POL_bf1:        equ   adc_vars+0Bh  ; adc_POL buffer for adc_bf_8,F
adc_POL_ptr:        equ   adc_vars+0Ch  ; adc_POL_ptr temporary storage hi
adc_POL_ptr_lo:     equ   adc_vars+0Dh  ; adc_POL_ptr temporary storage lo

adc_bcd9:           equ   adc_vars+10h
adc_bcd8:           equ   adc_vars+11h
adc_bcd7:           equ   adc_vars+12h
adc_bcd6:           equ   adc_vars+13h
adc_bcd5:           equ   adc_vars+14h
adc_bcd4:           equ   adc_vars+15h
adc_bcd3:           equ   adc_vars+16h
adc_bcd2:           equ   adc_vars+17h
adc_bcd1:           equ   adc_vars+18h
adc_bcd0:           equ   adc_vars+19h
```

```

adc_bf_beg:      equ  adc_vars+1Fh  ; must be placed on xxx0h boundary
adc_bf_end:      equ  adc_vars+2Fh

pg0_vars:        set  adc_vars+33h  ; adc variables area end

;# ADC FLAGS
;# MC68HC11 ##### MC68HC11 #

adc_POL_pls:     equ  bit_0          ; flags in adc_sts
adc_POL_mns:     equ  bit_1
adc_POL_det:     equ  bit_2
adc_done:        equ  bit_3
adc_curr_cyc:    equ  bit_4
adc_curr_rti:    equ  bit_5
adc_sync_rq:     equ  bit_6
adc_curr_upd:    equ  bit_7

;# ADC CONSTANTS
;# MC68HC11 ##### MC68HC11 #

adc_intgr:       equ  2711h          ; adc constants
adc_bf_lght:     equ  0008h
adc_cyc_itv:     equ  0008h
adc_sync_itv:    equ  0010h

;# ADC TABLES
;# MC68HC11 ##### MC68HC11 #

adc_POL_tbl:     dfb  01h ; bit_0
                  dfb  02h ; bit_1
                  dfb  04h ; bit_2
                  dfb  08h ; bit_3
                  dfb  10h ; bit_4
                  dfb  20h ; bit_5
                  dfb  40h ; bit_6
                  dfb  80h ; bit_7

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#                               ADC35 ROUTINES
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

;# ADC SYNCHRONIZATION
;# MC68HC11 ##### MC68HC11 #

adc_sync:        nop
                  clra
                  staa  adc_sts

adc_sync_hi:     ldaa  #adc_eoc_pin
                  bita  adc_eoc_p
                  beq   adc_sync_hi
                  bita  adc_eoc_p
                  beq   adc_sync_hi
                  bita  adc_eoc_p
                  beq   adc_sync_hi
adc_sync_lo:     ldaa  #adc_eoc_pin
                  bita  adc_eoc_p
                  bne   adc_sync_lo
                  bita  adc_eoc_p

```

```

    bne    adc_sync_lo
    bita   adc_eoc_p
    bne    adc_sync_lo

```

```

;# ADC INTERRUPTS INITIALIZATION

```

```

;# MC68HC11 ##### MC68HC11 #

```

```

adc_irq_cnfg:  ldaa   #adc_sync_itv ; set adc_sync interval
               staa   adc_cyc_cnt
               clra
               staa   adc_rslt0    ; initialize adc counter
               staa   adc_rslt1
               staa   adc_POL_bf0  ; initialize adc_POL_bf
               staa   adc_POL_bf1  ;
               ldx    #adc_bf_beg  ; initialize adc_bf pointer
               stx    adc_bf_ptr
               ldaa   tctl2        ; configure adc_sts_pin interrupts
               anda   #adc_eoc_edg_clr
               oraa   #adc_eoc_edg_cnfg
               staa   tctl2
               ldaa   pact1        ; configure PAI system
               oraa   #adc_ov_cnfg
               staa   pact1
adc_irq_ini:   ldaa   #adc_eoc_msk  ; clear adc_sts_pin interrupt flag
               staa   tflg1
               ldaa   tmsk1        ; enable adc_sts_pin interrupts
               oraa   #adc_eoc_msk
               staa   tmsk1
               ldaa   pact1        ; enable PAI system
               oraa   #PAEN
               staa   pact1
               ldaa   #adc_ov_msk  ; clear adc_rslt1 overflow interrupt flag
               staa   tflg2
               ldaa   tmsk2        ; enable adc_rslt1 overflow interrupts
               oraa   #adc_ov_msk
               staa   tmsk2

               bset   adc_sts, adc_curr_cyc ; set adc cyclic update flag
               bclr   adc_sts, adc_sync_rq
               rts

```

```

; DC OV INTERRUPT

```

```

;# MC68HC11 ##### MC68HC11 #

```

```

adc_ov_irq:    nop
               inc    adc_rslt1
               ldaa   #adc_ov_msk
               staa   tflg2
               ldaa   tflg1
               bita   #adc_eoc_msk
               bne    adc_irq1
               rti

```

```

;# ADC EOC INTERRUPT

```

```

;# MC68HC11 ##### MC68HC11 #

```

```

adc_eoc_irq:   ldaa   tflg2        ; test if adc_ov_f pending
               bita   #adc_ov_msk
               beq    adc_irq1
               inc    adc_rslt1    ; update adc_rslt1
               ldaa   #adc_ov_msk
               staa   tflg2
adc_irq1:      ldd    adc_bf_ptr    ; update adc_POL_bf
               ldy    #adc_POL_tbl ;

```

```

lsrb
bitb #bit_3 ;
bne adc_irq3 ;
andb #07h ; update adc_POL_bf0
aby ;
ldaa adc_eoc_p ; test adc_POL
bita #adc_POL_pin ;
beq adc_irq2 ; for input AND gate (08) (bne for NAND)
ldaa 0, Y ; insert negative POL
oraa adc_POL_bf0 ;
staa adc_POL_bf0 ;
bra adc_irq5 ;
adc_irq2: ldaa 0, Y ; insert positive POL
coma ;
anda adc_POL_bf0 ;
staa adc_POL_bf0 ;
bra adc_irq5 ;
adc_irq3: andb #07h ; update adc_POL_bf1
aby ;
ldaa adc_eoc_p ; test adc_POL
bita #adc_POL_pin ;
beq adc_irq4 ; for input AND gate (08) (bne for NAND)
ldaa 0, Y ; insert negative POL
oraa adc_POL_bf1 ;
staa adc_POL_bf1 ;
bra adc_irq5 ;
adc_irq4: ldaa 0, Y ; insert positive POL
coma ;
anda adc_POL_bf1 ;
staa adc_POL_bf1 ;
adc_irq5: ldx adc_bf_ptr ; update adc_bf
ldaa adc_rslt1 ;
adc_irq6: staa 0,X ; save adc_rslt1
ldaa adc_rslt0 ; save adc_rslt0
staa 1,X ;
adc_irq7: inx ; update adc_bf_ptr
inx
cpx #adc_bf_end
bls adc_irq8
ldx #adc_bf_beg
adc_irq8: stx adc_bf_ptr ; save adc_bf_ptr
; _irq_cyc: brclr adc_sts, adc_curr_cyc, adc_irq_end ; test adc mode:
dec adc_cyc_cnt ; if cyclic test if data is to be dspld
bne adc_irq_end
ldaa #adc_cyc_itv
staa adc_cyc_cnt
bset adc_sts, adc_curr_upd
adc_irq_end: bset adc_sts, adc_done
clra ; restore adc_rslt for next conversion
staa adc_rslt0
staa adc_rslt1
ldaa #adc_eoc_msk
staa tflg1
rti

```

```

;# ADC BUFFER AVERAGE

```

```

;# MC68HC11 ##### MC68HC11 #

```

```

adc_avrg: bclr adc_sts, adc_curr_upd
clra
ldx #opr_03
staa 0, X ; POL_pls buffer - opr_0
staa 1, X
staa 2, X

```

```

        staa 3, X
        staa 4, X      ; POL_mns buffer - opr_1
        staa 5, X
        staa 6, X
        staa 7, X
        bclr adc_sts, adc_POL_pls
        bclr adc_sts, adc_POL_mns
        ldaa #adc_bf_lght
        staa adc_avrg_cnt
        ldx  adc_bf_ptr

adc_avrg1:  cpx  #adc_bf_beg
           bhi  adc_avrg2
           ldx  #adc_bf_end
           inx
           inx
adc_avrg2:  dex
           dex
           stx  adc_POL_ptr
           ldd  adc_POL_ptr
           lsr  b
           bitb bit_3
           bne  adc_avrg3
           ldaa adc_POL_bf0
           bra  adc_avrg4
adc_avrg3:  ldaa adc_POL_bf1
adc_avrg4:  ldy  #adc_POL_tbl
           andb #07h
           aby
           anda 0, Y
           bne  adc_avrg_mns
adc_avrg_pls: ldd 0, X
           bset adc_sts, adc_POL_pls
           subd #adc_intgr
           addd opr_01
           bcc  adc_avrg_pls1
           inc  opr_02
           bne  adc_avrg_pls1
           inc  opr_03
adc_avrg_pls1: std opr_01
           bra  adc_avrg5
adc_avrg_mns: ldd 0, X
           bset adc_sts, adc_POL_mns
           subd #adc_intgr
           addd opr_11
           bcc  adc_avrg_mns1
           inc  opr_12
           bne  adc_avrg_mns1
           inc  opr_13
adc_avrg_mns1: std opr_11
adc_avrg5:  dec  adc_avrg_cnt
           bne  adc_avrg1

adc_sum:    nop
           brclr adc_sts, adc_POL_pls, adc_sum_mns
           brclr adc_sts, adc_POL_mns, adc_sum_pls
adc_sum_pm_p: ldaa opr_00
           suba opr_10
           staa opr_20
           ldaa opr_01
           sbca opr_11
           staa opr_21
           ldaa opr_02
           sbca opr_12

```

```

        staa  opr_22
        ldaa  opr_03
        sbca  opr_13
        staa  opr_23
        bcs   adc_sum_pm_m
        bclr  adc_sts, adc_POL_det      ; mark POL plus
        ldd   opr_23
        std   opr_03
        ldd   opr_21
        std   opr_01
        bra   adc_avrg6
adc_sum_pm_m:  ldaa  opr_10
        suba  opr_00
        staa  opr_00
        ldaa  opr_11
        sbca  opr_01
        staa  opr_01
        ldaa  opr_12
        sbca  opr_02
        staa  opr_02
        ldaa  opr_13
        sbca  opr_03
        staa  opr_03
        bset  adc_sts, adc_POL_det      ; mark POL minus
        bra   adc_avrg6

adc_sum_mns:  bclr  adc_sts, adc_POL_pls
        ldd   opr_13
        std   opr_03
        ldd   opr_11
        std   opr_01
        bra   adc_avrg6

adc_sum_pls:  bclr  adc_sts, adc_POL_pls

adc_avrg6:   ldaa  #adc_bf_lght
        staa  opr_20
        clra
        staa  opr_21
        staa  opr_22
        staa  opr_23
        jsr   div32rm
        ldd   opr_03
        std   adc_curr3
        ldd   opr_01
        std   adc_curr1

adc_bcd_val:  ldx   #adc_curr3
        ldy   #adc_bcd9
        jsr   cvrt_bin_bcd
        rts

;# ana96-f1.adc - END
;# MC68HC11 ##### MC68HC11 #

```



```

;# file updated: 98-02-05

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#                USER KEYBOARD ROUTINES
;#
;#                adc35-10.kbd
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

;# KB1x8-f1.usr FLAGS
;# MC68HC11 ##### MC68HC11 #

spvr_kbd_f:    equ bit_1

;# KB1x8-f1.usr CONSTANTS
;# MC68HC11 ##### MC68HC11 #

key_K0:       equ    0EFh    ; K0, ..., K7 keys hardware codes
key_K1:       equ    0FEh
key_K2:       equ    0DFh
key_K3:       equ    0FDh
key_K4:       equ    0BFh
key_K5:       equ    0FBh
key_K6:       equ    07Fh
key_K7:       equ    0F7h

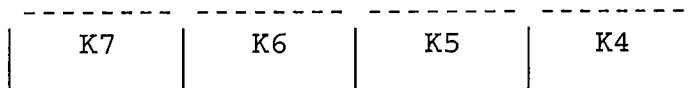
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#                KEYBOARD INTERPRETER ROUTINES
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#                THIS ROUTINES SHOULD BE INSERTED INTO KERF1-10.SRC
;#                AS THEY ARE TASK DEPENDENT
;#
;# MC68HC11 ##### MC68HC11 #

;# KBD INTERPRETER - SELECT FUNCTIONS TO BE SERVICED
;# MC68HC11 ##### MC68HC11 #

kbd_sel_drv:  bclr   sts_kbd, kbd_rq_f        ; clear kbd_rq_f flag
              ldab  mode
kbd_sel_sys:  cmpb  #mode_sys
              bne   kbd_sel_spvr
              jmp   kbd_sys_drv
kbd_sel_spvr: cmpb  #mode_spvr
              bne   kbd_sel_end
kbd_sel_spvr1: jmp  kbd_spvr_drv
kbd_sel_end:  rts

;# SYSTEM KEYBOARD
;# MC68HC11 ##### MC68HC11 #
;#
;#
;#
;#

```



```

;#
;#
;#
;#
;#
;#
;#
;# MC68HC11 ##### MC68HC11 #

```

K0	K1	K2	K3
swi->spv			

```

kbd_sys_drv:  nop
              ldaa  kbd_vd_cd
kbd_sys0:    cmpa  #key_K0
              bne   kbd_sys1
              ;swi                ; involve supervisory routine
              ;ldaa  #dsp_src0
              ;bra   kbd_sys_upd
              bne   kbd_sys_end
kbd_sys1:    cmpa  #key_K1
              bne   kbd_sys2
              ldaa  #dsp_src1
              bra   kbd_sys_upd
kbd_sys2:    cmpa  #key_K2
              bne   kbd_sys3
              ldaa  #dsp_src2
              bra   kbd_sys_upd
kbd_sys3:    cmpa  #key_K3
              bne   kbd_sys4
              ldaa  #dsp_src3
              bra   kbd_sys_upd
kbd_sys4:    cmpa  #key_K4
              bne   kbd_sys5
              ldaa  #dsp_src4
              bra   kbd_sys_upd
kbd_sys5:    cmpa  #key_K5
              bne   kbd_sys6
              ldaa  #dsp_src5
              bra   kbd_sys_upd
kbd_sys6:    cmpa  #key_K6
              bne   kbd_sys7
              ldaa  #dsp_src6
              bra   kbd_sys_upd
k   _sys7:   cmpa  #key_K7
              bne   kbd_sys_end
              ldaa  #dsp_src7

kbd_sys_upd: staa  dsp_source    ; quit kbd_sys_drv
              bset  sts_lcd, dsp_rq_f
kbd_sys_end: bclr  sts_kbd, kbd_rq_f
              rts

```

```

;# ngs96-f1.kbd - END
;# MC68HC11 ##### MC68HC11 #

```

;<# file updated: 98-02-05

;<# MC68HC11 ##### MC68HC11 #
;<# MC68HC11 ##### MC68HC11 #
;<#
;<# USER LCD 2x16 DISPLAY ROUTINES #
;<# #
;<# adc35-10.dsp #
;<# #
;<# MC68HC11 ##### MC68HC11 #
;<# MC68HC11 ##### MC68HC11 #

dsp\_source: equ pg0\_vars+01h ; dsp source pointer

pg0\_vars: set pg0\_vars+01h ; update allocation pointer

dsp\_src0: equ 00h
dsp\_src1: equ 01h
dsp\_src2: equ 02h
dsp\_src3: equ 03h
dsp\_src4: equ 04h
dsp\_src5: equ 05h
dsp\_src6: equ 06h
dsp\_src7: equ 07h

;<# MC68HC11 ##### MC68HC11 #
;<# MC68HC11 ##### MC68HC11 #
;<#
;<# SYSTEM REMARKS TABLES #
;<# #
;<# MC68HC11 ##### MC68HC11 #
;<# MC68HC11 ##### MC68HC11 #

;<# LINE FORMAT: | 0123456789ABCDEF | line\_0
;<# | | line\_1

lcd\_ver\_rem: DFB " PIAP-ORC " ; SYSTEM REM TABLE
DFB " MPU-F1-10 "
lcd\_rst\_rem: DFB "\* adc35-10.src \*" ; POWER ON RESTART REM TABLE
DFB " pwr\_on restart "
1 pas\_rem: DFB " restart passed " ; POWER ON PASSED REM TABLE
lcd\_sys\_rem: DFB "mpu-f1 > " ; SYSTEM REM TABLE
lcd\_K0\_rem: DFB " key K0 pressed " ; key K0 rem table
lcd\_K1\_rem: DFB " key K1 pressed " ; key K1 rem table
lcd\_K2\_rem: DFB " key K2 pressed " ; key K2 rem table
lcd\_K3\_rem: DFB " key K3 pressed " ; key K3 rem table
lcd\_K4\_rem: DFB " key K4 pressed " ; key K4 rem table
lcd\_K5\_rem: DFB " key K5 pressed " ; key K5 rem table
lcd\_K6\_rem: DFB " key K6 pressed " ; key K6 rem table
lcd\_K7\_rem: DFB " key K7 pressed " ; key K7 rem table
adc\_snc\_rem: DFB "adc7135 sync\_ip " ; adc35 rem table
adc\_msm\_rem: DFB "adc7135 enabled " ; adc35 rem table
adc\_cyc\_rem: DFB "cyc: + " ; adc35 rem table
adc\_val\_rem: DFB "val: + " ; adc35 rem table
lcd\_bld\_rem: DFB " " ; BLANK rem table
DFB " "

;<# MC68HC11 ##### MC68HC11 #
;<# MC68HC11 ##### MC68HC11 #
;<#

```

;#                DISPLAY TASK ROUTINES                #
;#                #
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;# DSP mode select DRIVERS
;# MC68HC11 ##### MC68HC11 #

dsp_sel_drv:  brset sts_lcd, dsp_rq_f, dsp_sel_drv1
              jmp   dsp_sel_end
dsp_sel_drv1: bclr  sts_lcd, dsp_rq_f
              ldab  mode                ; load mode into B
dsp_sel_sys:  cmpb  #mode_sys            ; test if mode_sys
              bne   dsp_sel_spvr
              jmp   dsp_sys_drv
dsp_sel_spvr: cmpb  #mode_spvr          ; test if mode_spvr
              bne   dsp_sel_end
dsp_sel_spvr1: jmp  dsp_spvr_drv
dsp_sel_end:  rts                        ; quit kbd_drv

;# DSP sys DRIVERS
;# MC68HC11 ##### MC68HC11 #

dsi_sys_drv:  ldaa  dsp_source          ; define display source

dsp_sys0:     cmpa  #dsp_src0
              bne  dsp_sys1
              ;ldy  #lcd_K0_rem        ; load IY with K0_rem address
              ;bra  dsp_sys_ldr
              jmp  adc_view_sync       ; adc services
dsp_sys1:     cmpa  #dsp_src1
              bne  dsp_sys2
              ;ldy  #lcd_K1_rem        ; standard service
              ;bra  dsp_sys_ldr       ; standard service
              jmp  adc_view            ; adc services
dsp_sys2:     cmpa  #dsp_src2
              bne  dsp_sys3
              ldy  #lcd_K2_rem
              bra  dsp_sys_ldr
dsp_sys3:     cmpa  #dsp_src3
              bne  dsp_sys4
              ldy  #lcd_K3_rem
              bra  dsp_sys_ldr
dsp_sys4:     cmpa  #dsp_src4
              bne  dsp_sys5
              ldy  #lcd_K4_rem
              bra  dsp_sys_ldr
dsp_sys5:     cmpa  #dsp_src5
              bne  dsp_sys6
              ldy  #lcd_K5_rem
              bra  dsp_sys_ldr
dsp_sys6:     cmpa  #dsp_src6
              bne  dsp_sys7
              ldy  #lcd_K6_rem
              bra  dsp_sys_ldr
dsp_sys7:     cmpa  #dsp_src7
              bne  dsp_sys_end
              ldy  #lcd_K7_rem

dsp_sys_ldr:  jsr   dsp_l1_ldr
dsp_sys_ldr0: ldy  #lcd_sys_rem        ; load IY with sys_rem address
              jsr   dsp_l0_ldr        ; reload remark table into dsp_bf
dsp_sys_end:  rts                        ; quit dsp_drv

```

```

;# ADC sys DRIVERS
;# MC68HC11 ##### MC68HC11 #

adc_view_sync:nop
        ldy    #adc_snc_rem
        jsr    dsp_l1_ldr
        rts

adc_view:    nop
adc_view_cyc: brclr  adc_sts, adc_curr_cyc, adc_view_val
        ldy    #adc_cyc_rem
        jsr    dsp_l1_ldr
        bra    adc_view_pol
adc_view_val: ldy    #adc_val_rem
        jsr    dsp_l1_ldr
adc_view_pol: brclr  adc_sts, adc_POL_det, adc_view_bcd
        ldy    #dsp_bf_15
        ldaa   #"-"
        staa   0, Y
adc_view_bcd: ldx    #adc_bcd9
        ldy    #dsp_bf_16
        ldab   #0Ah
        jsr    cvrt_bcd_asc
ac__view_msm: ldy    #adc_msm_rem
        jsr    dsp_l0_ldr
adc_view_end: rts                    ; quit dsp_drv

;# adc35-f1.dsp - END
;# MC68HC11 ##### MC68HC11 #

```

```

;# file updated: 98-02-05

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#          SUPERVISORY DISPLAY ROUTINES
;#
;#          adc35-10.spv
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

;# SUPERVISORY VARIABLES
;# MC68HC11 ##### MC68HC11 #

sts_spvr3:    equ pg0_vars+01h    ; supervisory error buffer
sts_spvr2:    equ pg0_vars+02h
sts_spvr1:    equ pg0_vars+03h
sts_spvr0:    equ pg0_vars+04h
spvr_CCR:     equ pg0_vars+05h    ; supervisory registers buffers
spvr_B:       equ pg0_vars+06h
spvr_A:       equ pg0_vars+07h
spvr_X:       equ pg0_vars+08h
spvr_Y:       equ pg0_vars+0Ah
spvr_PC:      equ pg0_vars+0Ch
spvr_SP:      equ pg0_vars+0Eh

dsp_spvr_reg: equ pg0_vars+10h    ; lcd regs pointer

pg0_vars:     set pg0_vars+10h    ; update allocation pointer

;# SUPERVISORY FLAGS
;# MC68HC11 ##### MC68HC11 #

spvr_irq_f:   equ bit_0          ; flags in sts_spvr0
spvr_xrq_f:   equ bit_1
spvr_swi_f:   equ bit_2
spvr_ill_f:   equ bit_3
spvr_cop_f:   equ bit_4
spvr_cmf_f:   equ bit_5
spvr_pwr_f:   equ bit_6
spvr_rst_f:   equ bit_7

spvr_sci_f:   equ bit_0          ; flags in sts_spvr1
spvr_spi_f:   equ bit_1
spvr_rti_f:   equ bit_2
spvr_pae_f:   equ bit_3
spvr_pao_f:   equ bit_4
spvr_tof_f:   equ bit_5
spvr_rsvd_f:  equ bit_6
;spvr_xxx_f:  equ bit_7

spvr_i4o5_f:  equ bit_0          ; flags in sts_spvr2
spvr_oc4_f:   equ bit_1
spvr_oc3_f:   equ bit_2
spvr_oc2_f:   equ bit_3
spvr_oc1_f:   equ bit_4
spvr_ic3_f:   equ bit_5
spvr_ic2_f:   equ bit_6
spvr_ic1_f:   equ bit_7

;# SUPERVISORY CONSTANTS
;# MC68HC11 ##### MC68HC11 #

```

110

```

sp_spvr:      equ 003FEh      ; supervisory stack pointer value
mode_spvr:    equ 0F0h        ; supervisory mode

spvr_rg0:     equ 00h         ; supervisory display register page0
spvr_rg1:     equ 01h
spvr_rg2:     equ 02h

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#
;#          SYSTEM REMARKS TABLES
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

;# LINE FORMAT:      | 0123456789ABCDEF | line_0
;#                   |                               | line_1

spvr_urc_rem: DFB  "*" UNRECOGNIZED  *" ; UNRECOGNIZED SPVR ENTRY
spvr_page_rem:DFB  "sts_spv:fffffff" ; SUPERVISORY STATUS TABLE
spvr_pcsp_rem:DFB  "PC:pppp SP:ssss"  ; SUPERVISORY REGS TABLES
spvr_ixy_rem:  DFB  "IX:xxxx IY:yyyy"
spvr_abc_rem:  DFB  "A:aa B:bb CCR:cc"

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#
;#          INITIALIZE SOFTWARE SUPERVISORY ENVIRONMENT
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

spvr_env_ini:  ldaa  #mode_spvr      ; introduce supervisory mode on restart
               staa  mode
               clra
               staa  sts_spvr0
               staa  sts_spvr1
               staa  sts_spvr2
               staa  sts_spvr3
               ldaa  #spvr_rg0
               staa  dsp_spvr_reg
               rts

;# SUPERVISORY cop ROUTINE
;# M68HC11 ##### M68HC11 #

spvr_cop_drv:  nop
               ldaa  #055h
               staa  coprst
               ldaa  #0AAh
               staa  coprst

spvr_cop_end:  rts

;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#
;#          SUPERVISORY SPURIOUS EXCEPTIONS DRIVERS
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

spvr_i4o5_spu: bset  sts_spvr2, spvr_i4o5_f
               bra   spvr_exc_spu

```

AAA

```

spvr_oc4_spu: bset sts_spvr2, spvr_oc4_f
               bra  spvr_exc_spu
spvr_oc3_spu: bset sts_spvr2, spvr_oc3_f
               bra  spvr_exc_spu
spvr_oc2_spu: bset sts_spvr2, spvr_oc2_f
               bra  spvr_exc_spu
spvr_oc1_spu: bset sts_spvr2, spvr_oc1_f
               bra  spvr_exc_spu
spvr_ic3_spu: bset sts_spvr2, spvr_ic3_f
               bra  spvr_exc_spu
spvr_ic2_spu: bset sts_spvr2, spvr_ic2_f
               bra  spvr_exc_spu
spvr_ic1_spu: bset sts_spvr2, spvr_ic1_f
               bra  spvr_exc_spu
spvr_sci_spu: bset sts_spvr1, spvr_sci_f
               bra  spvr_exc_spu
spvr_spi_spu: bset sts_spvr1, spvr_spi_f
               bra  spvr_exc_spu
spvr_rti_spu: bset sts_spvr1, spvr_rti_f
               bra  spvr_exc_spu
spvr_pae_spu: bset sts_spvr1, spvr_pae_f
               bra  spvr_exc_spu
spvr_pao_spu: bset sts_spvr1, spvr_pao_f
               bra  spvr_exc_spu
spvr_tof_spu: bset sts_spvr1, spvr_tof_f
               bra  spvr_exc_spu
spvr_rsvd_spu: bset sts_spvr1, spvr_rsvd_f
               bra  spvr_exc_spu
spvr_irq_spu: bset sts_spvr0, spvr_irq_f
               bra  spvr_exc_spu
spvr_xrq_spu: bset sts_spvr0, spvr_xrq_f
               bra  spvr_exc_spu
spvr_swi_spu: bset sts_spvr0, spvr_swi_f
               bra  spvr_exc_spu
spvr_ill_spu: bset sts_spvr0, spvr_ill_f
               bra  spvr_exc_spu
spvr_cop_spu: bset sts_spvr0, spvr_cop_f
               bra  spvr_res_spu
spvr_cmf_spu: bset sts_spvr0, spvr_cmf_f
               bra  spvr_res_spu
spvr_rst_spu: bset sts_spvr0, spvr_rst_f
               bra  spvr_res_spu
               nop

spvr_exc_spu: sei                ; SUPERVISORY SERVICES FOR EXCEPTIONS
               pula                ; restore exception regs from the stack
               staa spvr_CCR
               pula
               staa spvr_B
               pula
               staa spvr_A
               pulx
               stx  spvr_X
               pulx
               stx  spvr_Y
               pulx
               stx  spvr_PC
               tsx
               stx  spvr_SP
               bra  spvr_err_exit

spvr_res_spu: sei                ; SUPERVISORY SERVICES FOR RESETS
               ldaa #0FFh
               staa spvr_CCR      ; set supervisory regs

```



```

    staa spvr_B
    staa spvr_A
    ldx #0FFFFh
    stx spvr_X
    stx spvr_Y
    ldx #0FFFEh
    stx spvr_PC
    ldx #sp_spvr
    stx spvr_SP
    bra spvr_err_exit

```

```

spvr_sft_err: sei ; SUPERVISORY SERVICES FOR SOFTWARE ERRORS
              staa spvr_A ; copy software regs
              stab spvr_B
              tpa
              staa spvr_CCR
              stx spvr_X
              sty spvr_Y
              txs
              stx spvr_SP
              ldx 0,X
              stx spvr_PC

```

```

spvr_err_exit: lds #sp_spvr ; initialize supervisory stack pointer
              clra ; set task dependent parameters
              staa sts_lcd
              staa sts_kbd
              ldaa mode ; save current task mode
              staa sts_spvr3
              ldaa #mode_spvr ; set supervisory mode
              staa mode
              ldaa #spvr_rg0 ; set regs page0
              staa dsp_spvr_reg
              jsr dsp_spvr_drv ; display error remark
              jsr lcd_data_ldr

              lds #sp_ini ; restore initial stack pointer
              cli ; enable interrupts

              jmp exe_drv ; back to the main loop

```

```

MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;# LCD 2x16 SUPERVISORY DISPLAY ROUTINES #
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

```

```

;# SUPERVISORY DISPLAY: |0123456789ABCDEF| line
;# spvr status page -> |sts_spv:00000000| line_0
;# spvr registers page -> |PC:hhhh SP:hhhh| line_1

```

```

;# DSP spvr DRIVER
;# MC68HC11 ##### MC68HC11 #

```

```

dsp_spvr_drv: nop
dsp_spvr_pg: ldy #spvr_page_rem ; load IY with spvr_page rem address
            jsr dsp_l0_ldr ; reload remark table into dsp_bf
            ldy #dsp_bf_08
            ldx #sts_spvr3
            jsr cvrt_hex8_asc

dsp_spvr_rg: ldaa dsp_spvr_reg ; display spvr_regs
dsp_spvr_rg0: cmpa #spvr_rg0 ; display spvr_rg0 contents

```

A12

```

    bne    dsp_spvr_rg1
    ldy    #spvr_pcsp_rem
    jsr    dsp_l1_ldr
    ldy    #dsp_bf_13
    ldx    #spvr_PC
    jsr    cvrt_hex4_asc
    ldy    #dsp_bf_1C
    ldx    #spvr_SP
    jsr    cvrt_hex4_asc
    bra    dsp_spvr_ldr
dsp_spvr_rg1: cmpa    #spvr_rg1           ; display spvr_rg1 contents
    bne    dsp_spvr_rg2
    ldy    #spvr_ixy_rem
    jsr    dsp_l1_ldr
    ldy    #dsp_bf_13
    ldx    #spvr_X
    jsr    cvrt_hex4_asc
    ldy    #dsp_bf_1C
    ldx    #spvr_Y
    jsr    cvrt_hex4_asc
    bra    dsp_spvr_ldr
dsp_spvr_rg2: nop           ; display spvr_rg2 contents
    ldaa   #spvr_rg2
    staa   dsp_spvr_reg
    ldy    #spvr_abc_rem
    jsr    dsp_l1_ldr
    ldy    #dsp_bf_12
    ldx    #spvr_A
    jsr    cvrt_hex2_asc
    ldy    #dsp_bf_17
    ldx    #spvr_B
    jsr    cvrt_hex2_asc
    ldy    #dsp_bf_1E
    ldx    #spvr_CCR
    jsr    cvrt_hex2_asc
dsp_spvr_ldr: bset    sts_lcd, lcd_l0_rq_f
    bset    sts_lcd, lcd_l1_rq_f
dsp_spvr_end: rts           ; quit dsp_drv

```

```

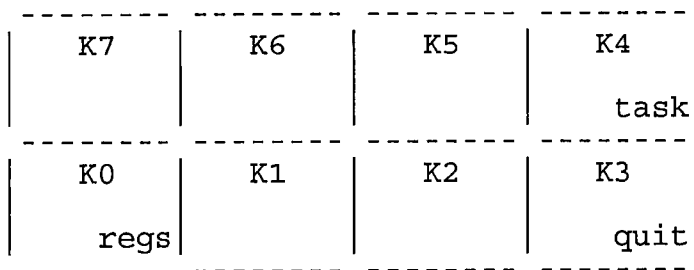
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #
;#
;#          SUPERVISORY KEYBOARD ROUTINES
;#
;# MC68HC11 ##### MC68HC11 #
;# MC68HC11 ##### MC68HC11 #

```

```

;# SUPERVISORY KEYBOARD
;# MC68HC11 ##### MC68HC11 #
;#
;#

```



```

;# MC68HC11 ##### MC68HC11 #

```

```

kbd_spvr_drv: nop
    ldaa   kbd_vd_cd

```

*MB*

```

kbd_spvr_K0:  cmpa  #key_K0
              bne  kbd_spvr_K3
              ldaa dsp_spvr_reg
              inca
              cmpa #spvr_rg2
              bls  kbd_spvr_K01
              ldaa #spvr_rg0
kbd_spvr_K01: staa  dsp_spvr_reg
              bra  kbd_spvr_upd
kbd_spvr_K3:  cmpa  #key_K3
              bne  kbd_spvr_K4
              ldaa #mode_sys           ; restore system mode
              staa mode
              bra  kbd_spvr_upd
kbd_spvr_K4:  cmpa  #key_K4
              bne  kbd_spvr_end
              ldaa sts_spvr3         ; restore task mode
              staa mode
kbd_spvr_upd: bset  sts_lcd, dsp_rq_f
kbd_spvr_end: bclr  sts_kbd, kbd_rq_f ; quit kbd_spvr_drv
              rts

```

```

;† :gs96-f1.spv - END

```

```

;# :C68HC11 ##### MC68HC11 #

```