

MIKROKOMPUTER DYDAKTYCZNY ZD537

Instrukcja laboratoryjna

Kazimierz Kapłon
Jacek Majewski
Jarosław Sugier

Część I: Opis sprzętu

Część II: Środowisko programowe

Dodatki: 1. Monitor ZD537

2. Program testowy

3. Przykładowy program dydaktyczny

4. Schematy

Katedra Informatyki Technicznej
Politechnika Wrocławska

CZEŚĆ I OPIS SPRZĘTU

1. Skrócony opis zestawu dydaktycznego ZD537

Zestaw dydaktyczny ZD537 przeznaczony jest do realizacji ćwiczeń laboratoryjnych mających na celu:

- zapoznanie z architekturą prostych 8 bitowych systemów mikroprocesorowych,
- naukę programowania w języku assemblera mikrokontrolerów rodziny '51,
- poznanie podstawowych zasad wykorzystywania mikrokontrolerów rodziny '51.

Zestaw ZD537 składa się z następujących modułów i elementów:

- płyta główna zawierająca mikrokomputer jednokładowy Infineon / Siemens 80C537,
- płyta dodatkowa, która zawiera: diody LED, klawisze sterujące, buzzer, generator przebiegu prostokątnego oraz stabilizator napięcia zasilania,
- klawiatura 16. klawiszowa,
- okablowanie umożliwiające przyłączenie ZD537 poprzez porty szeregowo μC '537 do komputera klasy IBM PC,
- zasilacz niestabilizowany ($7 \div 9$) V.

Schematy logiczne i ideowe płyty głównej oraz płyty dodatkowej i klawiatury przedstawiono w dodatku 4 na stronach 20 i 21. Schematy montażowe obu płyt, umożliwiające lokalizację elementów i złączy zestawu dydaktycznego, pokazano na stronie 19.

2. Płyta główna ZD537

Płyta główna zestawu zawiera:

- Mikrokomputer jednokładowy 80C537; układ μC 80C537 (U1) jest rozbudowaną wersją μC 8051. Mikrokomputer 80C537 zawiera między innymi:
 - 9 portów równoległych I/O oznaczonych jako P0 .. P8,
 - 2 porty szeregowo SIO0 i SIO1,
 - 8 bitowy przetwornik A/D,
 - układy czasowo-licznikowe T0, T1 i T2.

Charakterystykę i pełny opis budowy i działania poszczególnych modułów μC '537 można znaleźć między innymi w [6] i [7].

- Pamięć ROM: (układ 27512, U3, [9])
 - zawiera podstawowy program zarządzający MONITOR ZD537, wygenerowany przez pakiet oprogramowania firmy KEIL (patrz dodatek 1).

- Pamięć RAM (układ 431000, U4, [8])
 - przechowuje program użytkownika przesłany łączem szeregowym, zawartość po wyłączeniu zasilania podtrzymywana baterią litową na płycie głównej.
- Złącze portów szeregowych SIO0 i SIO1
 - porty szeregowe SIO0 i SIO1 μ C '537 są dołączone do złącza DB9 poprzez konwerter poziomów napięć MAX232 (U9, [13]) – transmisja w standardzie RS232. Wybór portu SIO0/SIO1 dokonuje się przełącznikiem obok złącza DB9. Transmisję SIO1 można również zrealizować w standardzie RS485. Wymaga to włożenia układu U10 ([15]) oraz zdjęcia zwory Z10. Ponadto, transmisja RS485 na odległość powyżej 10m (jednak ≤ 1200 m) wymaga użycia terminatorów dopasowujących.
- RTC (*Real Time Clock*) - układ zegara/kalendarza (RTC-72421, U7, [11])
 - rejestry układu RTC dekodowane w przestrzeni adresowej pamięci XRAM pod adresami FF0xh (rys. 1). Działanie układu RTC jest podtrzymywane baterią litową .
- Układ MAX691 (U8, [12]) - dozór napięcia zasilania (generowanie sygnału RESET po włączeniu zasilania) oraz przełączania podtrzymania bateryjnego.
- Przycisk SW2: RESET systemu.
- Moduł wyświetlacza LCD
 - 2 wiersze po 16 znaków,
 - wyświetlacz LCD mocowany na płycie głównej i dołączony do niej przez złącze LCD1,
 - rejestry układu sterującego wyświetlaczem LCD dostępne są w przestrzeni adresowej pamięci XRAM pod adresami FF2xh (rys. 1, opis układu sterującego HD44780 - [10]).
- Klawiatura matrycowa 16-klawiszowa
 - dołączona poprzez złącze JP1,
 - obsługa możliwa metodą skaningową,
 - wywoływanie przerwań od klawiatury wymaga użycia dodatkowego układu U11.
- Moduł 8 przełączników SW1
 - dołączony do linii portu P7.

Uwaga: Ponieważ port P7 jest również używany w obsłudze klawiatury, zatem aby uniknąć konfliktu należy pamiętać, że gdy używana jest klawiatura, bity przełącznika SW1 muszą być w stanie OFF!
- Dodatkowe elementy płyty głównej dostępne pod adresami portu P6:
 - P6.4 – przetwornik piezoelektryczny z generatorem o częstotliwości ok. 1kHz,
 - P6.0 – dioda LED w kolorze czerwonym (D3),
 - P6.6 – uzwojenie przekaźnika (styki przekaźnika dostępne są na złączach ZS11, ZS12).
- Złącze JP9/JP11 - umożliwia dołączenie urządzenia zewnętrznego sterowanego z portu P4.

- Przetwornik A/D (8 kanałów)
 - może być obsługiwany przez przełączanie styków SW1 (napięcia 0V lub 5V); płynna zmiana w zakresie 0-5V możliwa m.in. przez dołączenie potencjometrów do złącza JP10.
- Układy GAL (U5, U6 typu 16V8)
 - dekodowanie adresów pamięci i urządzeń w przestrzeni XRAM,
 - układy dekodowania zapewniają **trzy tryby pracy płyty głównej**:
 - a) **tryb MONITOR**: Z7=OFF Z8=OFF (górną pozycją przełącznika) - działa program MONITOR 537;
 - b) **tryb RAM**: Z7=OFF Z8=ON (dolną pozycją przełącznika) - działa program użytkownika zapisany w pamięci RAM;
 - c) **tryb ROM**: Z7=ON Z8=OFF/ON (bez znaczenia) - działa program zapisany w pamięci ROM.
- Złącza JP5, JP6 – m. in. zasilanie płyty głównej z płyty dodatkowej.

3. Płyta dodatkowa ZD537

Płyta dodatkowa zasilana jest z zewnętrznego zasilacza niestabilizowanego o napięciu (7÷9)V. Na płycie umieszczono stabilizator 7805 (IC1). Płyta główna zestawu ZD537 zasilana jest napięciem stabilizowanym poprzez złącze JP5&JP6. Poprzez to samo złącze przyłączonych jest 8 diod LED, które umożliwiają obserwację stanów portu P1 μ C '537.

Do linii P1.7 dołączono generator przebiegu prostokątnego o regulowanej częstotliwości - układ NE555 (IC2). Zakres regulowanych częstotliwości wynosi (1÷30) Hz. Generowany przez układ NE555 przebieg można odłączyć od linii P1.7 zdejmując zworę J1. Jednoczesne generowanie sygnałów przez układ NE555 i μ C'537 na linii P1.7 jest układowo zabezpieczone opornikiem R16.

Do czterech linii portu P3 (P3.2, P3.3, P3.4 i P3.5) dołączone są diody LED oraz przyciski umożliwiające zadawanie sygnałów, np. generowanie przerwań. Pozostałe bity portu P3 są zajęte do obsługi modułów płyty głównej – sygnały \RD i \WR oraz RxD i TxD transmisji szeregowej – i nie są dostępne dla użytkownika zestawu ZD537.

Do bitu P3.2 dołączony jest piezoelektryczny przetwornik elektroakustyczny umożliwiający generowanie sygnałów dźwiękowych metodą programową.

4. Klawiatura

Klawiatura matrycowa jest dołączona do obwodów płyty głównej poprzez złącze JP1. Składa się z czterech wierszy, każdy po cztery klawisze (dodatek 4, schemat str. 21). Obsługa

klawiatury metodą skaningową wymaga wysterowania linii portu P5 (P5.4 ÷ P5.7) wartością zera logicznego, a następnie odczytaniu bitów P7.3 do P7.0.

Włożenie układu U11 umożliwia wywoływanie przerwań od klawiatury na linii P1.4 (dodatkowe przerwania procesora '537). Gdy używana jest klawiatura bity przełącznika SW1 muszą być w stanie wyłączenia OFF !

5. Okablowanie złącza transmisji szeregowej

Sygnały portów transmisji szeregowej SIO0 i SIO1 dostępne są na złączu JP2 i 9 stykowym złączu DB9 (schemat str. 21). Przełącznik dwustanowy (obok złącza DB9 na ścianie tylnej zestawu) włączony pomiędzy JP2 a złącze DB9 pozwala wybrać, który z portów procesora '537 SIO0 lub SIO1 zostanie przyłączony do złącza DB9 i komunikuje się z μ C PC. Ustawienie przełącznika w kierunku kropki oznacza przyłączenie do złącza DB9 kanału SIO0.

6. Mapa portów ZD537

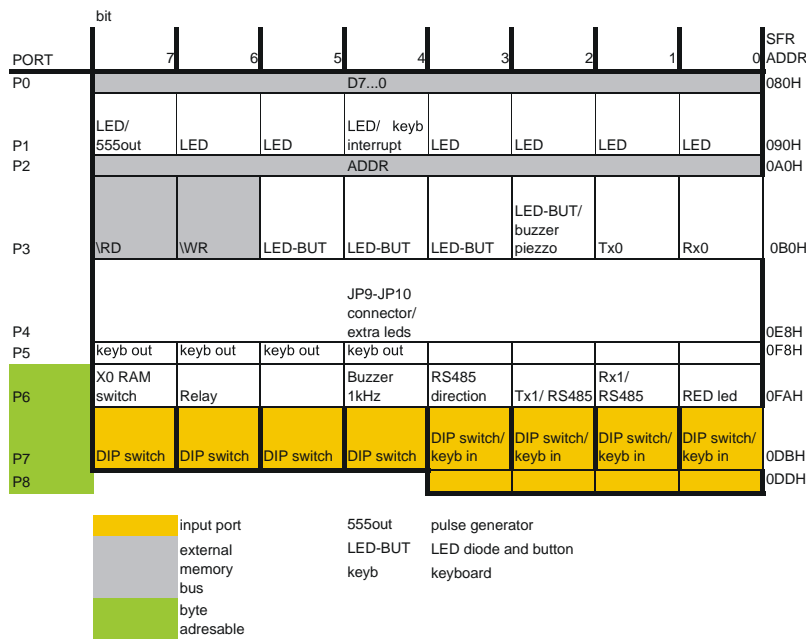
Na rysunku 1 przedstawiono mapę portów zestawu ZD537.

- Porty P0, P2 oraz częściowo P3 (linie P3.6 i P3.7) pełnią funkcję magistral wewnętrznych systemu '51 i są niedostępne dla użytkownika zestawu ZD537.
- Do portu P1 dołączono diody LED. Dodatkowo do portu P1.7 można dołączyć zworą J1 układ NE555 generatora sygnału prostokątnego. Na linii P1.4 może być generowany sygnał przerwania od klawiatury pod warunkiem włożenia układu U11.
- Do linii P3.5 ... P3.2 dołączono diody LED oraz przyciski. Ponadto, do linii P3.2 jest dołączony brzęczyk (*buzzer*) piezoelektryczny.
- Bity portu P4 wyprowadzone są na złącze JP9/JP11 i dostępne są dla użytkownika.
- Do starszej części portu P5 dołączono wyprowadzenia klawiatury. Sterowanie wierszami klawiatury odbywa się poprzez młodszą część portu P7 (*keyb in*). Jednocześnie do wszystkich linii portu P7 dołączone są przełączniki DIP SWITCH SW1. W przypadku wykorzystywania klawiatury, przełączniki SW1 muszą być wyłączone (OFF).
- Port P6 steruje czerwoną diodą D3 (P6.0) na płycie głównej, przekaźnikiem REL1 oraz brzęczykiem (*PC1-buzzer*) generującym stały ton o częstotliwości ok. 1 kHz. Bit P6.6 jest odpowiedzialny za kierunek transmisji złącza RS485. Linie RS485 są współdzielone ze złączem SIO1. Bit P6.7 przełącza banki pamięci XRAM.

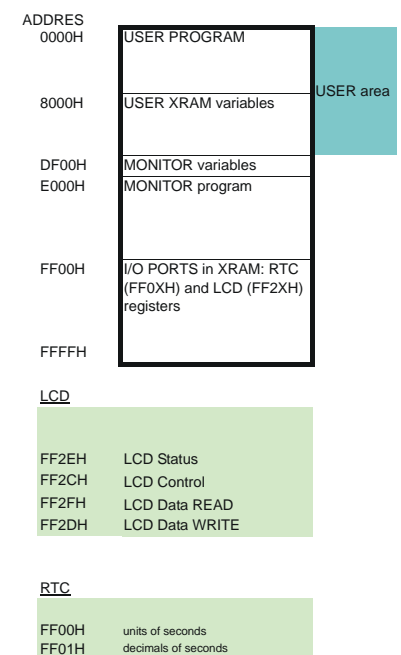
UWAGA:

- | | |
|------------------------|------------------------------------------------------|
| – porty P6, P7 i P8 | – adresowane bajtowo – brak dostępu do bitów portów, |
| – porty P1, P3, P4, P5 | – adresowane bajtowo lub/i bitowo, |
| – porty P7 i P8 | – można jedynie odczytywać (porty przetwornika A/D). |

ZD537 I/O PORTS



ZD537 XRAM & CODE MEMORY MAP



Rysunek 1: Mapa portów oraz przestrzeni adresowych zestawu ZD537.

7. Mapa przestrzeni pamięci XRAM i CODE

Mapę przestrzeni adresowych pamięci pokazano na rysunku 1.

- Pod kontrolą programu MONITOR przesyłany program użytkownika umieszczany jest w przestrzeni adresowej CODE od adresu 0000h (fizycznie pamięć RAM, U4).
- Zmienne użytkownika w przestrzeni XRAM powinny być alokowane od adresu 8000h, zaś jeśli potrzeba większego obszaru na zmienne, można je umieścić zaraz po programie użytkownika.
- Obszar XRAM od adresu DF00h jest zajęty przez lokalne zmienne programu MONITOR (fizycznie pamięć RAM, U4).
- Od adresu E000h w przestrzeni CODE znajduje się kod programu MONITOR ZD537 (fizycznie pamięć ROM, U3).
- W obszarze XRAM od adresu FF00h umieszczone są porty I/O:
 - Od adresu FF00h umieszczono 16 rejestrów układu RTC (zegar – kalendarz). Dane przekazywane są w kodzie BCD jako cyfry czterobitowe: pod adresem FF00h dostępne są jednostki sekund, pod adresem FF01h widoczne są dziesiątki sekund itd. ([11]).
 - Rejestry wyświetlacza LCD widoczne są od adresu FF2Xh ([10]).

W przypadku wygenerowania programu użytkownika i zaprogramowaniu go w pamięci ROM (tryb pracy ROM) cała pamięć XRAM (64kB) dostępna jest dla użytkownika. Zmienne użytkownika mogą więc zacząć się od adresu 0000H. Położenie portów I/O w przestrzeni XRAM pozostaje bez zmian. W takiej sytuacji można przełączać banki pamięci XRAM bitem P6.7. Przełączanie banków pamięci XRAM pad kontrolą monitora jest niecelowe bo zostanie utracony dostęp do zmiennych MONITORA ZD537.

CZEŚĆ II ŚRODOWISKO PROGRAMOWE

8. Komunikacja z komputerem PC

Współpraca zestawu ZD 537 z komputerem IBM PC wykorzystuje oprogramowanie firmy Keil Software GmbH (<http://www.keil.com>): zintegrowane środowisko uruchomieniowe μ Vision lub uproszczony monitor tekstowy mon51.exe. W obydwu przypadkach komunikacja odbywa się poprzez złącze szeregowo. W celu nawiązania transmisji po stronie zestawu należy:

- a) przełączyć gniazdo DB9 w tryb SIO0 (przełącznik w pozycji oznaczonej kropką),
- b) zainicjalizować zestaw (RESET) w trybie MONITOR (przełącznik Z8 zwolniony, zestaw zgłasza się komunikatem MONITOR ZD 537_ na wyświetlaczu LCD).

Komunikację z komputerem po stronie zestawu obsługuje monitor wygenerowany przy użyciu narzędzi Keil i zapisany w pamięci EPROM (dodatek 1, [14]).

9. Środowisko μ Vision 2

Aplikacja μ Vision to zintegrowane środowisko uruchomieniowe (*Integrated Development Environment*, IDE), które łączy w sobie wszystkie narzędzia używane w procesie tworzenia oprogramowania mikrokomputera '51. Umożliwia w jednej aplikacji edycję kodu źródłowego, kompilację, konsolidację (linkowanie), przesłanie kodu do pamięci zestawu oraz uruchomienie programu (w tym śledzenie krokowe, zakładanie pułapek, obserwowanie stanu portów itp.).

Używana w laboratorium wersja demonstracyjna posiada ograniczenie nie pozwalające wygenerować więcej niż 2 kB kodu wynikowego.

INFORMACJE O PROGRAMIE

Nie jest celem niniejszej instrukcji omówienie wszystkich możliwości μ Vision (które są bardzo szerokie!), gdyż duża ich część jest związana z pisanem programów w języku C i w ogóle nie jest wykorzystywana na laboratorium. Wyczerpujące informacje można znaleźć w

pomocy programu, która jest zgrupowana w panelu Books (otwierany poleceniem Help → Open Books Window). W szczególności na potrzeby laboratorium użyteczne mogą być:

- plik pomocy Windows dostępny pod tytułem „uVision User's Guide”;
- „Getting Started with μ Vision2”, plik GS51.PDF;
- „Macro Assembler and Utilities”, plik A51.PDF.

Dokumenty PDF są przechowywane w folderze Keil\C51\HLP\.

Poniżej omówiono główne kroki utworzenia, edycji oraz uruchomienia przykładowego projektu. Jeśli ustawienia pewnych zmiennych pominięto, oznacza to że są nieistotne albo należy je pozostawić w ustawieniach domyślnych.

UTWORZENIE NOWEGO PROJEKTU

1) Polecenie menu: Project → New project...

- Pliki projektu mają rozszerzenie .uv2.
- Zaleca się utworzenie projektu w osobnym katalogu (w oknie dialogowym „Create new project” można tworzyć nowe foldery).
- W oknie Select device for target ‘Target1’ należy odszukać typ układu (Infineon SAB 80C537).

2) Ustawienie opcji projektu: Project → Options for target ‘Target1’ (patrz rysunek 2)

- Zakładka Target: podać częstotliwość Xtal 12 MHz.
- Zakładka Output: sprawdzić czy jest zaznaczone pole Debug information.
- Zakładka Debug: zaznaczyć Use: Keil Monitor-51 Driver oraz Load application at startup.

Uwaga: możliwe jest uruchamianie programu w trybie symulacji programowej (bez dołączonego zestawu ZD-537), należy wówczas w zakładce Debug (rys. 2a) zaznaczyć Use Simulator.

- Klawisz Settings (rys. 2b): Port Com 2 (lub inny używany po stronie PC do komunikacji z zestawem), Baudrate 9600.

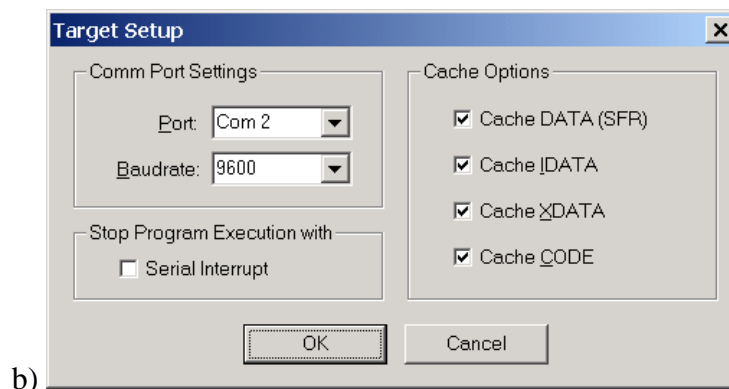
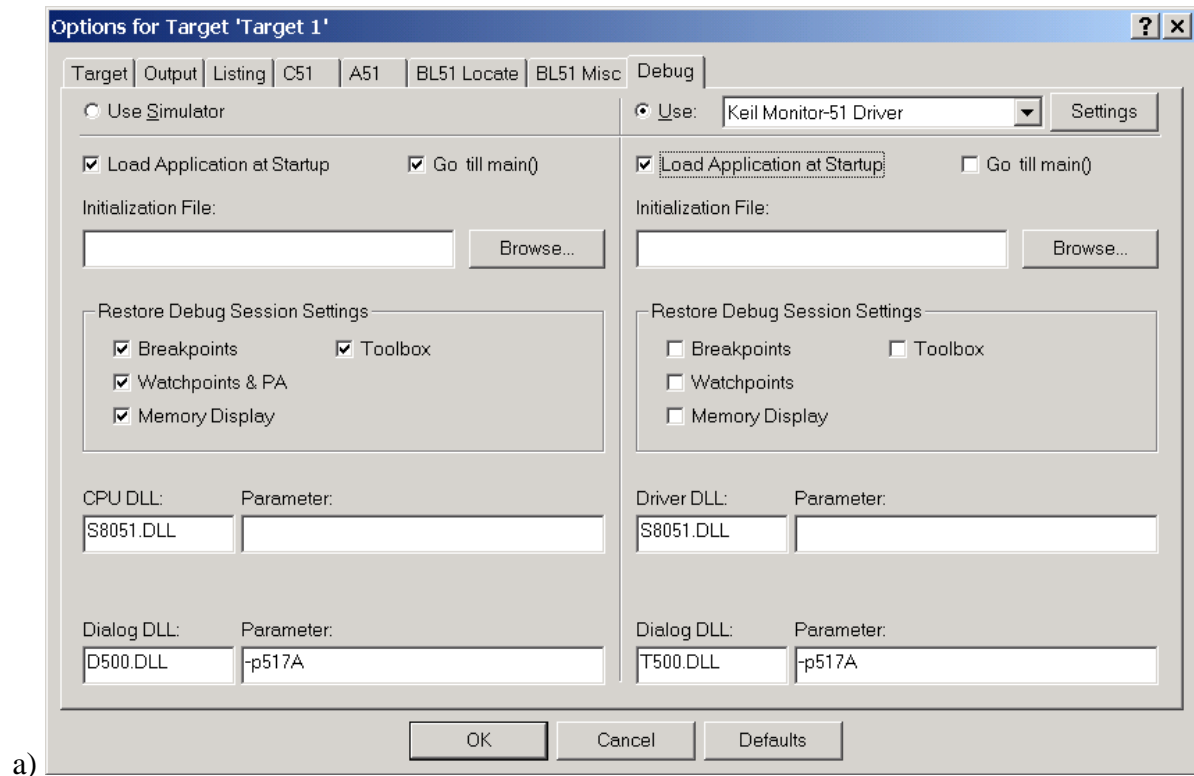
Uwaga: w przypadku uruchamiania programów, które używają do własnych celów kanału SIO0 mogą występować konflikty z transmisją monitora Keil; celowe wówczas może być wyczyszczenie wszystkich opcji Cache w ustawieniach z rysunku 2b.

3) Utworzenie pliku z kodem źródłowym

- Polecenie File → New, przy zapisie jako rozszerzenie nazwy pliku z kodem assemblerowym należy stosować a51 lub asm.
- Dodanie pliku do projektu: polecenie Project → Targets, Groups, Files..., zakładka Groups / Add files, zaznaczyć Source Group 1, przycisk Add files to group..., wskazać utworzony plik źródłowy.

KOMPILACJA I ŁĄCZENIE

- Polecenie: Project → Build target (F7) lub Project → Rebuild all targets.



Rysunek 2: Ustawianie opcji projektu.

- W przypadku wystąpienia błędów dwukrotne kliknięcie na linię z komunikatem błędu (wyświetlana w panelu Build) rozpoczyna edycję odpowiedniej linii źródłowej.

URUCHAMIANIE

- Polecenie: Debug → Start/stop debug session (Ctrl+F5).

Uwaga: jeśli zaznaczono w opcjach projektu Load application at startup, kod programu powinien automatycznie zostać przetransmitowany łączem szeregowym (postęp transmisji widoczny na pasku statusu w lewym dolnym rogu okna) i debugger winien zgłosić kursorem ustawionym na pierwszym poleceniu programu (żółta strzałka z lewej strony pierwszej linii kodu programu). Jeśli tak nie jest, najprawdopodobniej podczas tworzenia programu wystąpiły błędy (kompilacji, łączenia, transmisji...), które przed dalszą pracą należy usunąć.

- Śledzenie krokowe wykonania programu:
Debug → Go (F5)
Debug → Step (F11)
Debug → Step over (F10)
Debug → Run to cursor line (Ctrl+F10)
- Śledzenie wartości zmiennych (np. zdefiniowanych w pamięci IRAM poleceniem asemblera DATA): polecenie View → Watch & call stack window.
- Śledzenie zawartości obszarów pamięci: polecenie View → Memory window.
- Odwołania do różnych obszarów adresowych '51 uzyskuje się podając odpowiedni przedrostek adresu:
B:0xxx - pamięć IRAM adresowana bitowo
C:0xxxxx - pamięć CODE
D:0xxx - pamięć IRAM adresowana bezpośrednio (*direct*)
I:0xxx - pamięć IRAM adresowana pośrednio (*indirect*)
X:0xxxxx - pamięć XRAM

10. Monitor tekstowy MON51.EXE

Monitor MON51.EXE jest prostym tekstowym programem monitorującym działanie zestawów '51 przystosowanych do komunikacji z komputerem PC wg protokołu firmy Keil. Uruchomienie:

```
mon51.exe 2
```

gdzie parametr 2 podaje numer portu *Com*. Zakończenie działania - klawisz F1.

Przykładową sesję pracy z monitorem ilustruje rysunek 3. Symbol '#' oznacza *prompt* (zaproszenie do pisania). Użyte jako pierwsze polecenie `help` podaje repertuar możliwości monitora. Rozkaz `load tes01.hex` powoduje załadowanie przykładowego programu testowego. Plik `test01.hex` (utworzony np. w środowisku μ Vision) powinien znajdować się w tym samym katalogu co monitor `mon51.exe`. Polecenie `g` (*go*) uruchamia załadowany program. Tekst `PROCESS TERMINATED AT` jest wynikiem użycia przycisku RESET w trakcie działania. Polecenie `dc` (*display code*) wyświetla bajtowo zawartość pamięci CODE, natomiast `u` (*unassemble*) – zawartość tego samego obszaru po deasemblacji.

Przykładowy program testowy, który może być użyty do szybkiego sprawdzenia sprawności elementów zestawu, opisany jest w dodatku 2.

```

C:\WINNT\System32\cmd.exe - mon51 2
Copyright (c) 1995 KEIL SOFTWARE, INC. All rights reserved.

INSTALLED FOR PC/XT/AT (COM LINE 2) USING HARDWARE INTERRUPT SERVICE
*** MONITOR MODE ***
#AUDRATE: 9600 (DEFAULT)
#
#help
memory display modify fill utility
bit: >DB range >EB address >FILLB range value >A address - assemble
code: >DC range >EC address >FILLC range value >U range - disassemble
data: >DD range >ED address >FILLD range value >X [register] - disp/change
idata: >DI range >EI address >FILLI range value
xdata: >DX range >EX address >FILLX range value
pdata: >DP range >EP address >FILLP range value

program execution breakpoint(s) program load/save
>G [address] [,breakadd] - go >BD bp - disable >LOAD file - load hex/obj
>T [count] - trace step >BE bp - enable >SAVE file range - save hex
>P [count] - procedure step >BK bp - kill >LS file - load symbols
>BL - list
>HELP - display menu >BS address - set
#load test01.hex
....
#g

PROCESSING TERMINATED AT 0000
#dc
C:0000: 02 01 A8 30 99 FD C2 99 8F 99 22 E5 9B 30 E1 FB : ...0.....0..
C:0010: 53 9B FD 8F 9C 22 75 09 20 E4 F5 08 63 90 FF E4 : $....u....c...
C:0020: F5 0A F5 0B 05 0B E5 0B 70 02 05 0A E5 0B B4 FF : .....p.....
C:0030: F3 E5 0A B4 1F EE E5 FA 30 E4 0B 53 FA EF 53 FA : .....0..$.$.
#u 0
0000H LJMP 01A8H
0003H JNB TI(99H),0003H
0006H CLR TI(99H)
0008H MOV SBUF(99H),R7
000AH RET
000BH MOV A,9BH
000DH JNB 0E1H,000BH
0010H ANL 9BH,#0FDH
0013H MOV 9CH,R7
0015H RET
#

```

Rysunek 3: Praca z monitorem tekstowym mon51.exe.

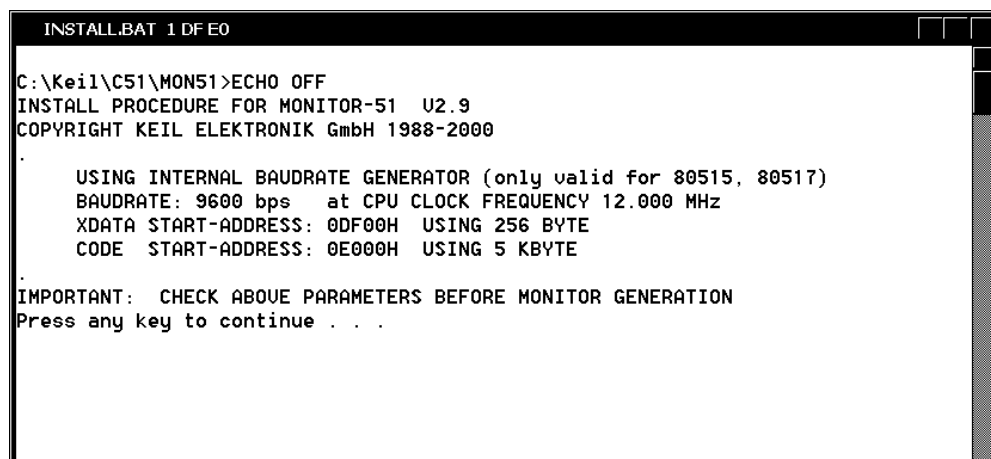
Dodatek 1: Monitor ZD537

Po stronie zestawu odpowiedzialnym za komunikację ze środowiskiem PC jest monitor (podstawowy program nadzorczy) zapisany w pamięci EPROM. Został wygenerowany przy użyciu narzędzi firmy Keil dołączonych do środowiska μ Vision w folderze \Keil\C51\MON51\.

Polecenie generacji monitora miało postać ([14]):

```
install.bat 1 DF E0
```

gdzie parametry 1 DF E0 oznaczają, odpowiednio, szybkość transmisji (9600 bps, *internal baudrate generator*) oraz numery stron pamięci XDATA i CODE, które mają być zajęte przez program (por. rysunek 4). W wyniku powstał zbiór mon51.hex, do którego dodano instrukcje generujące napis MONITOR ZD537_ na wyświetlaczu LCD oraz inicjowanie diody RED LED (P6.0, zapalenie) i BUZZER (P6.4, wyłączenie).



```
INSTALL.BAT 1 DF E0

C:\Keil\C51\MON51>ECHO OFF
INSTALL PROCEDURE FOR MONITOR-51  U2.9
COPYRIGHT KEIL ELEKTRONIK GmbH 1988-2000
.
  USING INTERNAL BAUDRATE GENERATOR (only valid for 80515, 80517)
  BAUDRATE: 9600 bps   at CPU CLOCK FREQUENCY 12.000 MHz
  XDATA START-ADDRESS: 0DF00H   USING 256 BYTE
  CODE  START-ADDRESS: 0E000H   USING 5 KBYTE
.
IMPORTANT:  CHECK ABOVE PARAMETERS BEFORE MONITOR GENERATION
Press any key to continue . . .
```

Rysunek 4: Generacja monitora ZD537.

Uwaga: Ponieważ złącze SIO0 jest zajęte na cele komunikacyjne monitora nie jest możliwa jego obserwacja w programie użytkownika. Aby było to możliwe można wygenerować program monitora komunikujący się przez port SIO1 (install.bat 3 DF E0) oraz podłączyć 2 kable do komputera IBM PC (*Com1* i *Com2*) – wówczas będzie można śledzić pracę łącza SIO0 używając do tego celu kanału SIO1.

Dodatek 2: Program testowy TEST01.HEX

Zadaniem programu TEST01.HEX jest sprawdzenie poprawności działania wszystkich elementów zestawu ZD537. I tak po jego uruchomieniu na wyświetlaczu LCD pojawia się napis:

<p>p 45 Change SW1</p>

Zmiana stanu przełącznika SW 1 powoduje zmianę znaku wyświetlanego na wyświetlaczu LCD (na rysunku znak ‘p’). Dwie kolejne cyfry są wynikiem działania układu RTC i pokazują upływający czas w sekundach. Kod znaku ustawiony na przełączniku SW1 jest jednocześnie wysyłany poprzez złącze SIO0. W kanale SIO1 jest transmitowany kod znaku o jeden większy. I tak jeśli kanał SIO0 transmituje literę **A** to kanał SIO1 będzie transmitował literę **B**, o czym można się przekonać przełączając przełącznik transmisji SIO (z tyłu zestawu).

Ponadto, program testowy zapala kolejno diody portu P1 i P4 (licznik pierścieniowy z krążącą wartością 0), a przetwornik piezoelektryczny (P3.2) wydaje terkoczący dźwięk. Przyciśnięcie przycisku P3.2 powoduje zaprzestanie wydawania dźwięku. Naciśnięcie przycisku P3.3 powoduje unieruchomienie brzęczyka generującego ton. Przycisk P3.4 jest odpowiedzialny za włączanie przekaźnika.

Jeśli przełącznik SW1 jest w pozycji OFF można testować sprawność działania klawiatury. Naciśnięcie przycisku na klawiaturze objawia się cyklicznym wyświetlaniem znaków specjalnych na wyświetlaczu i na złączu SIO0.

ZAWARTOŚĆ ZBIORU TEST01.HEX

```
:0B019D004368616E6765205357310016
:080003003099FDC2998F99228A
:0B000B00E59B30E1FB539BFD8F9C2226
:10001600750920E4F5086390FFE4F50AF50B050B76
:10002600E50B7002050AE50BB4FFF3E50AB41FEE13
:10003600E5FA30E40B53FAEF53FABF53FAFE8009A0
:1000460043FA1043FA4043FA010508E508B406C628
:1000560043FA1075985243D880758921758BF375CC
:100066008DF3D28E759BB2759DD990FF2EE020E759
:10007600F990FF2C7438F090FF2EE020E7F990FFFE
:100086002C7401F090FF2EE020E7F990FF2C740EFF
:10009600F090FF2EE020E7F990FF2C74C0F07BFF74
:1000A6007A01799D12016D90FF017401F090FF00B5
:1000B600F075E8FE7509FE30B20543FA0180035378
:1000C600FAFE30B30553FAEF800343FA1030B40555
:1000D60053FABF800343FA4085DB0890FF2EE020E9
:1000E600E7F990FF2C7480F090FF2EE020E7F9905E
:1000F600FF2DE508F090FF2EE020E7F990FF2D7424
:1001060020F090FF2EE020E7F990FF01E0540F2445
:100116003090FF2DF090FF2EE020E7F990FF00E0F1
:10012600540F243090FF2DF0AF097801EF088001BD
:1001360023D8FDF509FFF5E88FF88F90AF08120078
:1001460003E50804FF12000BE4F50AF50BB2B2054D
:100156000BE50B7002050AE50BB4FFF1E50AB40FD7
:07016600ECD2B20200BD2241
:10016D008B0C8A0D890EAB0CAA0DA90E1201B46071
:10017D001E90FF2EE020E7F9AB0C050EE50EAA0D43
:10018D007002050D14F91201B490FF2DF080D722E5
:030000000201A852
:0C01A800787FE4F6D8FD75810E02001689
:1001B400BB010689828A83E0225002E722BBFE0249
:0901C400E32289828A83E493227C
:00000001FF
```

Dodatek 3: Przykładowy program dydaktyczny

Ten dokument przedstawia przykładowy program testowy TEST dla zestawu dydaktycznego ZD537. Wszystkie moduły tego przykładu napisane zostały w języku assembler i są wzorowane na załączonym przez firmę KEIL programie template.a51 (patrz katalog \keil62\c51\asm\template.a51). Jest to polecany i obowiązujący styl pisania programów w assemblerze. Moduły programu pozwalają zobaczyć jak obsłużyć złącze transmisji szeregowej SIO0 i SIO1, wyświetlacz LCD, oraz zegar/kalendarz RTC.

Przedstawione moduły mają charakter przykładowy i nie są skończone. Zadanie dla studentów polega na użyciu przedstawionego przykładu i pisaniu swoich programów w podobnym stylu. Wskazane jest pisanie wielomodułowych programów z użyciem segmentów, makrodefinicji itp. Zadaniem laboratorium jest nauka nie tylko instrukcji procesora typu '51 lecz także poznanie pseudoinstrukcji języka assemblera.

Poniżej przedstawiono moduły przykładu wraz z komentarzami. Zamieszczone komentarze dotyczą numerów wierszy poszczególnych modułów i oznaczone są znakiem #nn, gdzie nn oznacza numer wiersza.

MODUŁ TEST.A51

```

1  $NOMOD51
2  NAME      TEST          ; ZD537 ASM tutorial
3
4  $NOLIST
5  #include <reg517.h>      // C-style  include definition file (for example, 80517)
6  ;$INCLUDE(reg517.inc)    ; asm-style include definition file (for example, 80517)
7  $INCLUDE(ZD537.inc)      ; definition file for ZD537 board
8  $LIST
9
10 EXTRN      CODE      (putcharSIO0, putcharSIO1, initSIO0, initSIO1) ; SIO functions
11 EXTRN      CODE      (putcharLCD, putstrLCD, putctrlLCD, initLCD)   ; LCD functions
12 EXTRN      CODE      (disp_time   )                               ; RTC functions
13
14 ?STACK      SEGMENT IDATA          ; ?STACK goes into IDATA RAM.
15              RSEG      ?STACK      ; switch to ?STACK segment.
16              DS        50          ; reserve your stack space
17              ; 50 bytes in this example.
18
19              CSEG      AT          0 ; absolute Segment at Address 0
20              LJMP      start        ; reset location (jump to start)
21
22 PROG        SEGMENT CODE
23              RSEG      PROG
24              USING     0            ; state register_bank used
25              ; for the following program code.
26 Start:      MOV        SP,#?STACK-1 ; assign stack at beginning
27
28              BUZZER_OFF;            ; Buzzer 1kHz off
29              REDLED_ON;             ; optional instruction
30              call initLCD ;
31              mov  A,#HOME2
32              call putctrlLCD         ; put LCD cursor to second line
33              MOV    DPTR,#text
34              call putstrLCD          ; display string
35
36              call initSIO0;
37              call initSIO1;
38
39 ?C01:
40              mov  A,#'A'
41              call putcharSIO0        ; SIO0 <-- 'A' (send charactear)
42              mov  A,#'B'
43              call putcharSIO1        ; SIO0 <-- 'B' (send charactear)
44              inc  P1                 ; binary counter on P1 LEDs
45              cpl  BUZZpiezzo         ; make sound on piezzo buzzer
46              mov  A,#HOME
47              call putctrlLCD
48              call disp_time
49
50              SJMP  ?C01               ; while(1);
51              RET
52
53 ?CO?TEXT     SEGMENT CODE
54              RSEG      ?CO?TEXT
55 text:        DB  "ZD537 test",00 ; text located in CODE memory
56
57              END                    ; END OF main
58

```

- #1 Wylączenie standardowych nazw rejestrów procesora '51 bo program dotyczy procesora typu '517.
- #4 Zabronienie listowania plików typu *header (include)*. Obowiązkowe jest poznanie zawartości zbioru *reg517.h* i *reg517.inc* !!!
- #5,6 Możliwe jest przygotowanie i włączenie plików definiujących na dwa sposoby: zgodny z językiem C i zgodny z językiem asemblera.
- #10-12 Definicje podprogramów zewnętrznych, w innych modułach.
- #14 Definicja segmentu stosu. Znak ? jest literą, która może wystąpić w nazwie.

- #19,20 Segment stały. Jest to pierwsza instrukcja po operacji RESET.
- #22 Segment relokowalny, umieszczony po obszarze wektorów przerwań.
- #26... Program testowy. Najpierw pojawia się tekst na wyświetlaczu LCD, a następnie w pętli wyprowadza się znaki A i B poprzez złącza SIO0 i SIO1 oraz czas na wyświetlaczu LCD.
- #53... Segment zawierający tekst, umieszczony w obszarze CODE.

MODUŁ: ZD537.INC

```

1 ; ZD537 BOARD: macros & definitions -----
2
3 NE555      BIT   P1.7
4 BUZZpiezzo BIT   P3.2
5
6 ;P6 port bit definitions
7 BUZZ1kHz   equ   00010000B
8 REDLED     equ   00000001B
9 RELAY      equ   01000000B
10
11 BUZZER_OFF  MACRO
12             ANL   P6,# (NOT BUZZ1kHz)
13             ENDM
14
15 BUZZER_ON   MACRO
16             ORL   P6,# BUZZ1kHz
17             ENDM
18
19 BUZZER_TOGGLE MACRO
20             XRL   P6,# BUZZ1kHz
21             ENDM
22
23 REDLED_ON   MACRO
24             ANL   P6,# (NOT REDLED)
25             ENDM
26
27
28 REDLED_OFF  MACRO
29             ORL   P6,#REDLED
30             ENDM
31
32 REDLED_TOGGLE MACRO
33             XRL   P6,#REDLED
34             ENDM
35
36 RELAY_OFF   MACRO                                ; assembler-style macrodefinition
37             ANL   P6,# (NOT RELAY)
38             ENDM
39
40
41 #define RELAY_ON                                // C-style macrodefinition
42             ORL   P6,# RELAY
43
44 ; LCD registers -----
45 LCDstatus   equ 0FF2EH
46 LCDcontrol  equ 0FF2CH
47 LCDdataWR   equ 0FF2DH
48 LCDdataRD   equ 0FF2FH
49
50 // LCD control bytes -----
51 #define HOME      0x80    // put curcor to second line
52 #define INITDISP  0x38    // LCD init (8-bit mode)
53 #define HOME2     0xc0    // put curcor to second line
54 #define LCDON     0x0e    // LCD nn, cursor off, blinking off
55 #define CLEAR     0x01    // LCD display clear
56
57
58 ; firts two RTC registers -----
59 RTCxs       equ 0FF00H    ; seconds
60 RTCsx       equ 0FF01H
61 RTCxm       equ 0FF02H    ; minutes
62 RTCmx       equ 0FF03H
63 RTCxh       equ 0FF04H    ; hours
64 RTChx       equ 0FF05H
65
66 RTCpd       equ 0FF0DH
67

```

- #3,4 Definicje bitów portów P1 i P3. Do pozostałych bitów można się odwołać używając notacji np. P1.0 , co oznacza bit 0 portu 1.
- #6-43 Do portu P6 nie ma dostępu bitowego. Dlatego zdefiniowano szereg makrodefinicji ułatwiających sterowanie urządzeniami dołączonymi do portu P6.
- #41,42 Jest to makrodefinicja zapisana w standardzie język C.
- #44-48 Rejestry LCD umieszczone są w przestrzeni XDATA pod wskazanymi adresami. Podobna sytuacja odnosi się do rejestrów RTC.

MODUŁ SIO.A51

```

1 $NOMOD51
2 NAME          SIO_CHAR_IO          ; basic procedures for serial comunication on SIO0 and SIO1
3
4 $NOLIST
5 //#include <reg517.h>                // include CPU definition file (for example, 80517)
6 $INCLUDE(reg517.inc)
7 $LIST
8
9 PUBLIC          putcharSIO0, putcharSIO1, initSIO0, initSIO1
10
11 SIO_CHAR_ROUTINES  SEGMENT  CODE
12                      RSEG    SIO_CHAR_ROUTINES
13
14
15 ;-----
16 ; Initialize serial interface
17 ; Using TIMER 1 to Generate Baud Rates
18 ; Oscillator frequency = 12MHz
19 initSIO0:
20     MOV     TMOD,#00100001B ;C/T = 0, Mode = 2
21     MOV     TH1,#-13
22     MOV     TL1,TH1
23     SETB    TR1
24     MOV     S0CON,#01010010B
25     ANL     ADCON0,#80H      ;12MHz 9600bps
26     RET
27 //#define initSIO_0 {S0CON=0x52; ADCON0|=0x80; TMOD=0x21;TH1=TL1=-13;TR1=1;} //9600 8-n-1
28
29 ;-----
30 ; Initialize serial interface 1
31 ; Oscillator frequency = 12MHz
32 initSIO1:
33     MOV     S1REL,#-39          ;12MHz 9600bps
34     MOV     S1CON,#0B2H
35     RET
36
37 //#define initSIO_1 { S1CON=0xB2; S1REL = -39; } //9600 8-n-1
38
39 ;-----
40 ; This routine outputs a single character through SIO0 to console.
41 ; The character is given in A.
42 putcharSIO0:
43     JNB     TI,$
44     CLR     TI
45     MOV     S0BUF,A
46     RET
47
48 ;-----
49 ; This routine outputs a single character throught SIO1 to console.
50 ; The character is given in A.
51 putcharSIO1:
52     PUSH    ACC
53     MOV     A,S1CON
54     JNB     ACC.1,putcharSIO1
55     ANL     S1CON,#0FDH
56     POP     ACC
57     MOV     S1BUF,A
58     RET
59
60     END
61
62 ; this module is not finished (lack of getchar, getstring ...)

```


Moduł obsługi transmisji szeregowej SIO0 i SIO1. Moduł nie jest skończony: brakuje procedur typu *getchar*, *putstring*, *getstring* itd.

MODUŁ LCD.A51

```

1  $NOMOD51
2  NAME      LCD_CHAR      ; LCD display procedures
3  $NOLIST
4  #include <reg517.h>      // include CPU definition file (for example, 80517)
5  ;$INCLUDE(reg517.inc)
6  $LIST
7
8  PUBLIC      putcharLCD, putstrLCD, initLCD, putctrlLCD
9  ; LCD registers      -----
10 LCDstatus equ 0FF2EH
11 LCDcontrol equ 0FF2CH
12 LCDdataWR equ 0FF2DH
13 LCDdataRD equ 0FF2FH
14
15 // LCD control bytes      -----
16 #define HOME      0x80      // put curcor to second line
17 #define INITDISP  0x38      // LCD init (8-bit mode)
18 #define HOM2      0xc0      // put curcor to second line
19 #define LCDON     0x0e      // LCD nn, cursor off, blinking off
20 #define CLEAR     0x01      // LCD display clear
21
22
23 LCDcntrlWR MACRO x
24     LOCAL loop
25 loop:
26     MOV     DPTR,#LCDstatus
27     MOVX    A,@DPTR
28     JB      ACC.7,loop      ; check if LCD busy
29
30     MOV     DPTR,#LCDcontrol ; write to LCD control
31     MOV     A, x
32     MOVX    @DPTR,A
33     ENDM
34
35 LCDcharWR MACRO
36     LOCAL loop1,loop2
37
38     PUSH    ACC
39 loop1:    MOV     DPTR,#LCDstatus
40     MOVX    A,@DPTR
41     JB      ACC.7,loop1      ; check if LCD busy
42
43 loop2:    MOV     DPTR,#LCDdataWR      ; write data to LCD
44     POP     ACC
45     MOVX    @DPTR,A
46     ENDM
47
48 init_LCD MACRO
49     LCDcntrlWR #INITDISP
50     LCDcntrlWR #CLEAR
51     LCDcntrlWR #LCDON
52     ENDM
53
54 LCD_CHAR ROUTINES SEGMENT CODE
55     RSEG LCD_CHAR_ROUTINES
56
57 ;-----
58 ; Initialize serial interface
59 initLCD:
60     init_LCD
61     RET
62 ;-----
63 ; This routine outputs a single character to LCD.
64 ; The character is given in A.
65 putcharLCD:
66     LCDcharWR
67     RET
68 ;-----
69 ; This routine outputs a control character to LCD.
70 ; The character is given in A.

```

```

71 putctrlLCD:
72     xch A, R2
73     LCDcntlWR R2
74     xch A, R2
75     RET
76 ;-----
77 ; This routine outputs a string to LCD. String is terminated by 00H.
78 ; The string in CODE memory is pointed by DPTR.
79 putstrLCD:
80     CLR    A
81     MOVC   A,@A+DPTR
82     JZ     ?EXIT      ; check if end of string
83     push   DPH
84     push   DPL
85     CALL   putcharLCD ; put char to LCD
86     pop    DPL
87     pop    DPH
88     INC    DPTR
89     SJMP   putstrLCD
90 ?EXIT: RET
91
92     END
93 ; this module is not finished (lack of polish characters ...)
94

```

Moduł obsługi wyświetlacza LCD demonstruje użycie makrodefinicji. Nie jest skończony: brakuje między innymi obsługi „polskich liter”.

MODUŁ RTC.A51

```

1  $NOMOD51
2  NAME      RTC          ; display time (minutes & seconds) on LCD
3
4  $NOLIST
5  #include <reg517.h>      // include CPU definition file (for example, 80517)
6  ;$INCLUDE(reg517.inc)
7  $LIST
8
9  ; firts two RTC registers -----
10 RTCxs equ 0FF00H        ; seconds
11 RTCsx equ 0FF01H
12 RTCxm equ 0FF02H        ; minutes
13 RTCmx equ 0FF03H
14 RTCxh equ 0FF04H        ; hours
15 RTChx equ 0FF05H
16
17 RTCpd equ 0FF0DH
18
19 PUBLIC disp_time
20
21 disp_nibble MACRO
22     movx A,@DPTR
23     anl  A,#0Fh          ; select 4-bits
24     orl  A,#30H          ; change to ASCII
25     call putcharLCD;
26     ENDM
27
28 EXTRN     CODE          (putcharLCD, putstrLCD, putctrlLCD, initLCD) ; LCD functions
29
30
31 RTC_PROC  SEGMENT CODE
32          RSEG   RTC_PROC
33
34 ;-----
35 ; get time and it dispaly on LCD
36 disp_time:
37     mov DPTR,#RTChx      ; get hours from RTC (higher nibble)
38     disp_nibble
39     mov DPTR,#RTCxh      ; get hours from RTC (lower nibble)
40     disp_nibble
41     mov A,#':'
42     call putcharLCD;
43     mov DPTR,#RTCmx      ; get minutes from RTC (higher nibble)
44     disp_nibble
45     mov DPTR,#RTCxm      ; get minutes from RTC (lower nibble)

```

```
46      disp_nibble
47      mov A,#': '
48      call putcharLCD;
49      mov DPTR,#RTCSx      ; get seconds from RTC (higher nibble)
50      disp_nibble
51      mov DPTR,#RTCxs      ; get seconds from RTC (lower nibble)
52      disp_nibble
53      RET
54
55
56      END                  ; END OF RTC
57 ; this module is not finished (lack of set time, write date/time as string ...)
58
59
```

Moduł obsługi układu zegar/kalendarz RTC demonstruje także użycie makrodefinicji. Nie jest skończony: brakuje procedur ustawiania daty/czasu oraz pisania daty i czasu do obszaru XDATA lub IDATA.

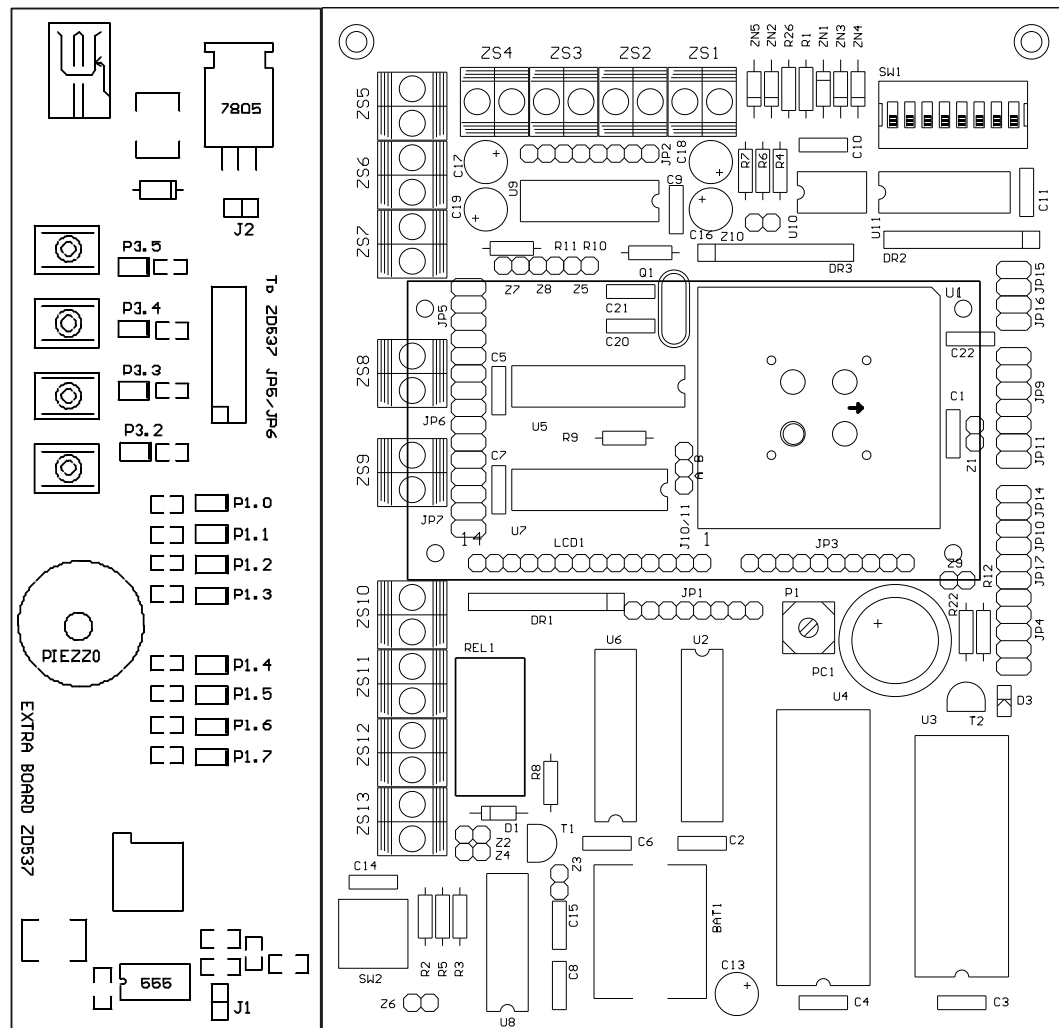
Dodatek 4: Schematy

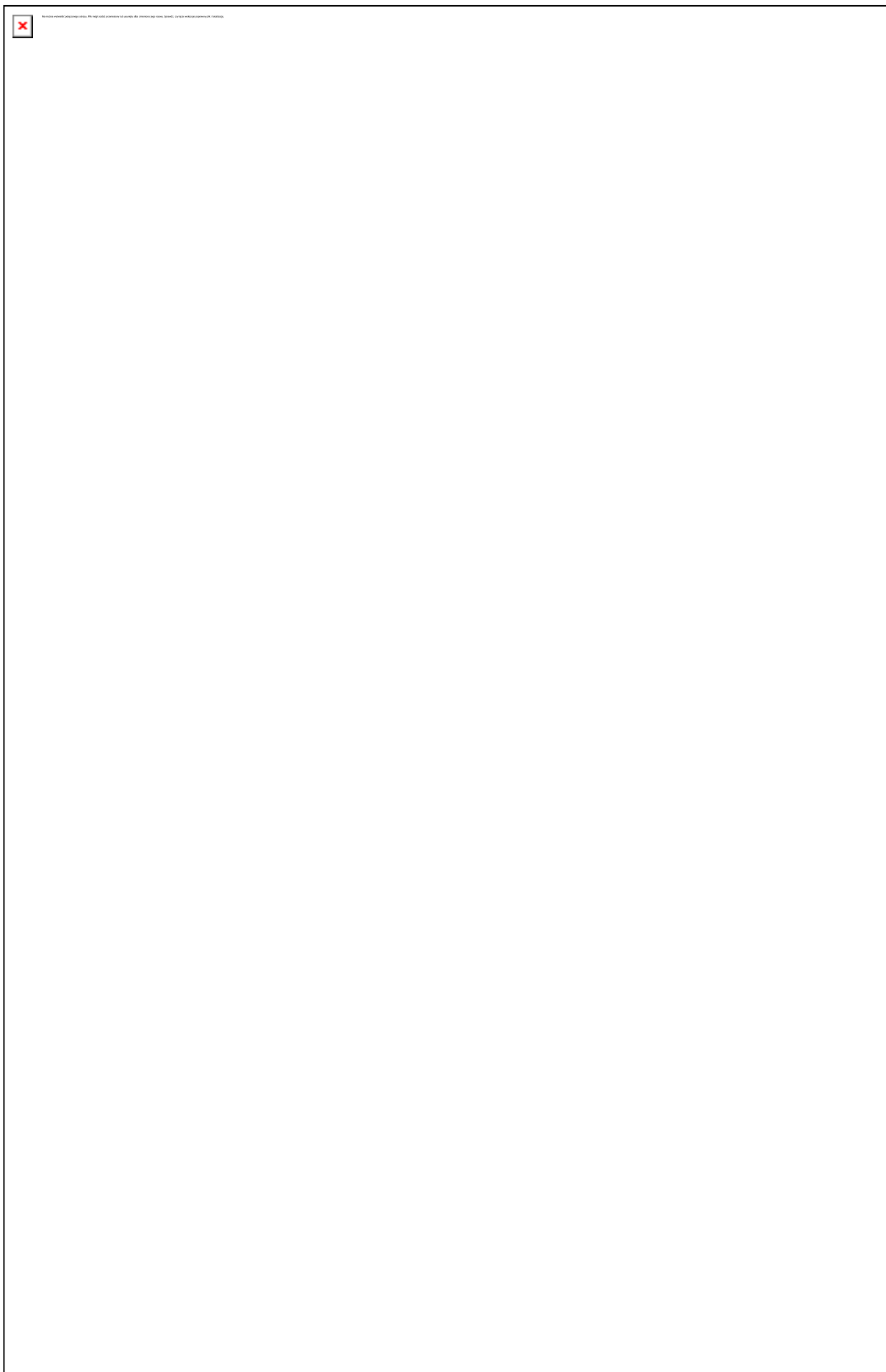
Str. 19 – Schemat montażowy

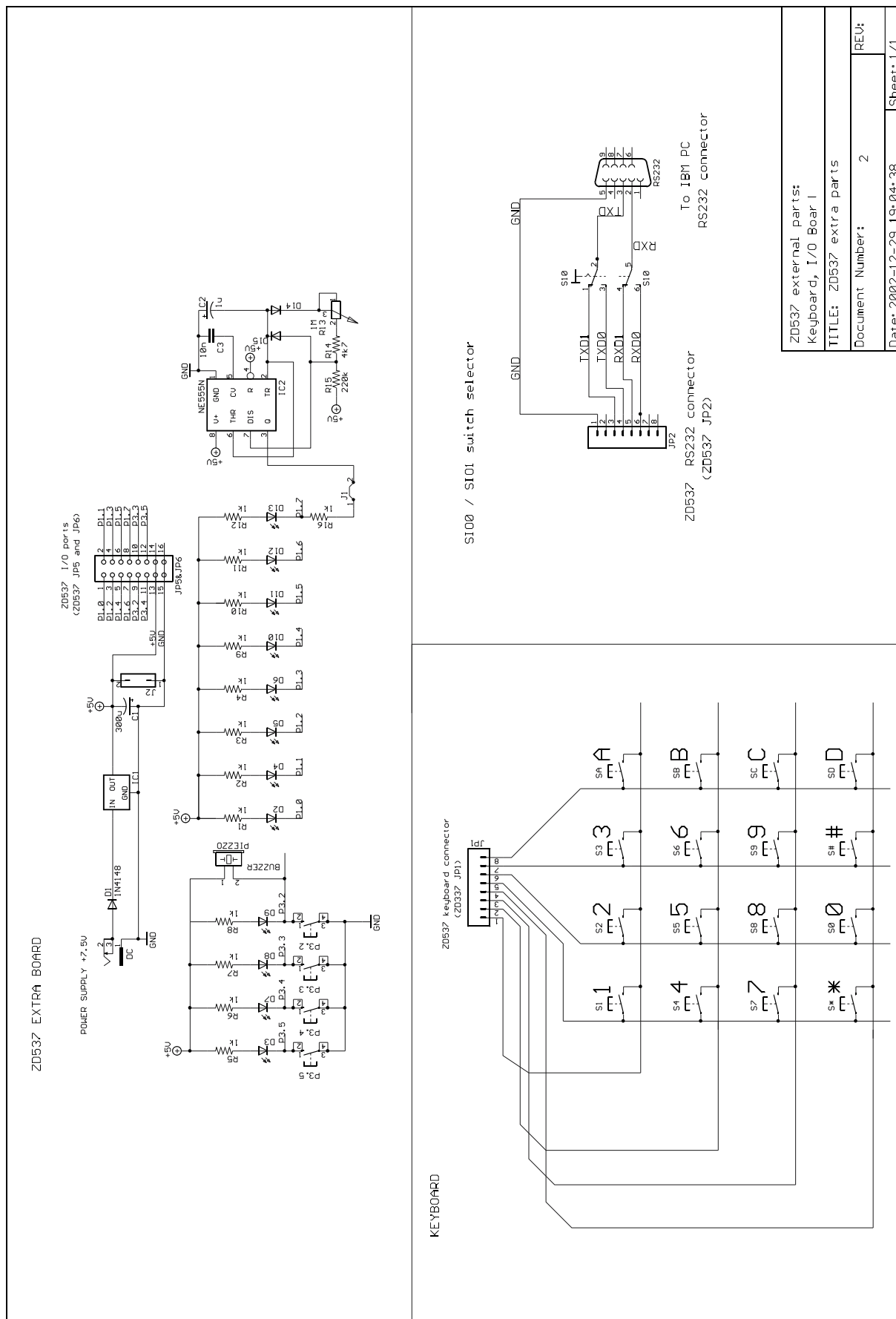
Str. 20 – Schemat logiczny płyty głównej ZD537

Str. 21 – Schematy układów dodatkowych (płyta dodatkowa, klawiatura, złącze DB9)

Power supply 7.5 V







Literatura

- [1] Janusz Janiczek, Andrzej Stępień: Systemy Mikroprocesorowe. Mikrokontrolery. Wydawnictwo Centrum Kształcenia Praktycznego, Wrocław 1997.
- [2] Janusz Janiczek, Andrzej Stępień: Systemy Mikroprocesorowe. Mikrokontroler 80(C)51/52. Wydawnictwo Elektronicznych Zakładów Naukowych, Wrocław 1995.
- [3] Andrzej Rydzewski: Mikrokomputery Jednookładowe Rodziny MCS-51. Wydawnictwo Naukowo – Techniczne, Warszawa 1995.
- [4] Jacek Majewski, Krzysztof Kardach: Programowanie Mikrokontrolerów z Serii 8x51 w Języku C (książka z płytą CD). Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2002.
- [5] Piotr Gałka, Paweł Gałka: Podstawy Programowania Mikrokontrolera 8051. Warszawa 1995.
- [6] Tomasz Starecki: Mikrokomputery Jednookładowe Rodziny 51. Wyd. NOZOMI, Warszawa 1996.

Dokumentacja PDF:

- [7] SAB 80C517/80C537, 8-Bit CMOS Single-Chip Microcontroller: User's Manual. Siemens Semiconductor Group, plik: 80517_USERMAN.PDF.
- [8] 1M-bit CMOS Static RAM, MOS Integrated Circuit μ PD431000A: Data Sheet. NEC Corp., plik: RAM_431000.PDF.
- [9] NMOS 512K (64K x 8) UV EPROM M27512. SGS-THOMSON Microelectronics, plik: 27512.PDF.
- [10] Dot Matrix Liquid Crystal Display Controller/Driver, HD44780U. Hitachi Ltd., plik: HD44780U.PDF.
- [11] Real Time Clock Module, RTC-72421/72423: Application Manual. SEIKO EPSON Corp., plik: RTC72421_APPMAN.PDF.
- [12] Microprocessor Supervisory Circuits: MAX691. Maxim Integrated Products, plik: MAX691A-MAX800M.PDF.
- [13] Precision, Single-Supply SPST Analog Switches: MAX323. Maxim Integrated Products, plik: MAX323-MAX325.PDF.
- [14] Application Note 152: Installing and Using Keil Monitor-51. Keil Elektronik GmbH, plik: MON51.PDF.
- [15] SN65176B, SN75176B: Differential Bus Transceivers. Texas Instruments Inc., plik: 75176.PDF.
- [16] GAL 16V8: High Performance E²CMOS PLD Generic Array Logic™. Lattice Semiconductor Corp., plik: 16V8.PDF.
- [17] 74HC/HCT573: Octal D-Type Transparent Latch; 3-State. Philips Semiconductors, plik: 74HC573.PDF.