MIKROKOMPUTER DYDAKTYCZNY ZD537

Instrukcja laboratoryjna

Kazimierz Kapłon
Jacek Majewski
Jarosław Sugier

Instytut Cybernetyki Technicznej
Politechnika Wrocławska 2003

CZĘŚĆ 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 asemblera mikrokontrolerów rodziny ’51,
– poznanie podstawowych zasad wykorzystywania mikrokontrolerów rodziny ‘51.

Zestaw ZD537 składa się z następujących modułów i elementów:

- płyta główna zawierająca mikrokomputer jednoukładowy 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 szeregowe μC ’537 do komputera klasy IBM PC,
- zasilacz niestabilizowany (7 ÷ 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 jednoukładowy 80C537; układ μC 80C537 (U1) jest rozbudowaną wersją μC 8051. Mikrokomputer 80C537 zawiera miedzy innymi:
  – 9 portów równoległych I/O oznaczonych jako P0 .. P8,
  – 2 porty szeregowe 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 μC '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 ≤ 1200m) 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 wyświetlacza 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 (górna pozycja przełącznika) - działa program MONITOR 537;
    b) **tryb RAM**: Z7=OFF Z8=ON (dolna pozycja 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 ZD537zasilana jest napięciem stabilizowanym poprzez złącze JP5&JP6. Poprzez to samo złącze przyłączonych jest 8 diod LED, które umożliwiają obserwację stanów portu P1 μC ‘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 μC’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 ściance 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 μC 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.

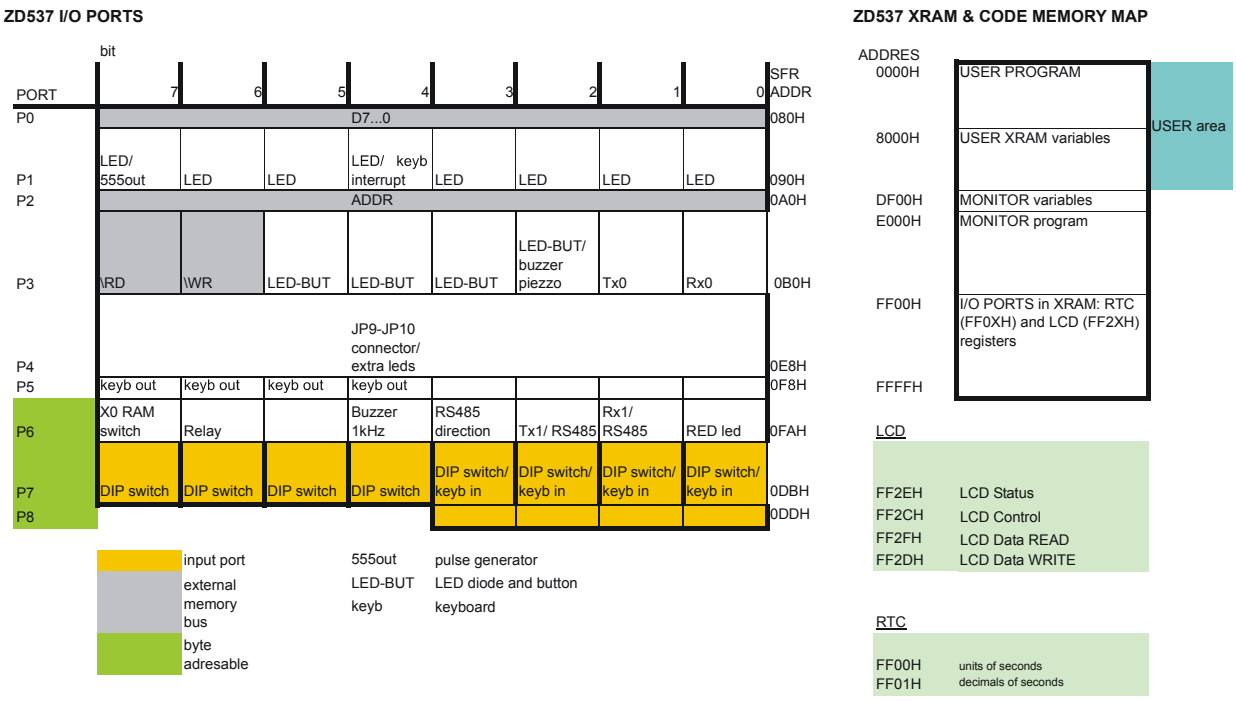
– Bity portu P4 wyprowadzone są na złącze JP9/JP11 i dostępne są dla użytkownika.

– 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).



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.

CZĘŚĆ 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 µVision lub uproszczony monitor tekstowy mon51.exe. W obydwu przypadkach komunikacja odbywa się poprzez złącze szeregowe. 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 µVision 2

Aplikacja µVision 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 µVision (które są bardzo szerokie!), gdyż duża ich część jest związana z pisaniem 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 µVision2", 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.

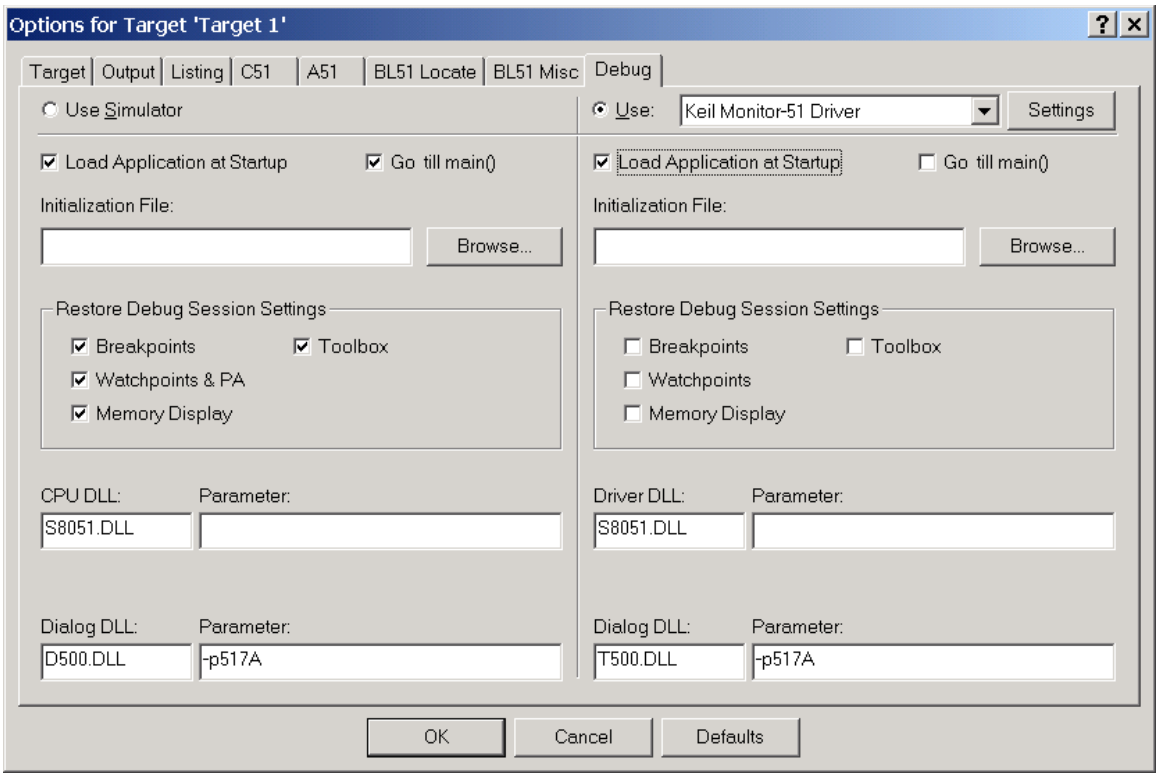
3) Utworzenie pliku z kodem źródłowym

- Polecenie File → New, przy zapisie jako rozszerzenie nazwy pliku z kodem asemblerowym 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.

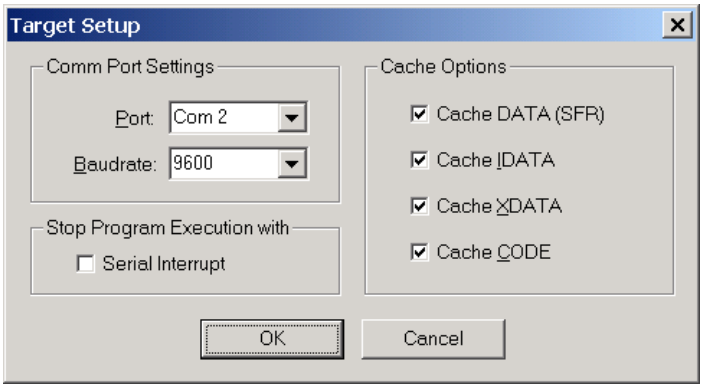
KOMPILACJA I ŁĄCZENIE

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

a)



b)



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 debuger winien zgłosić się 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

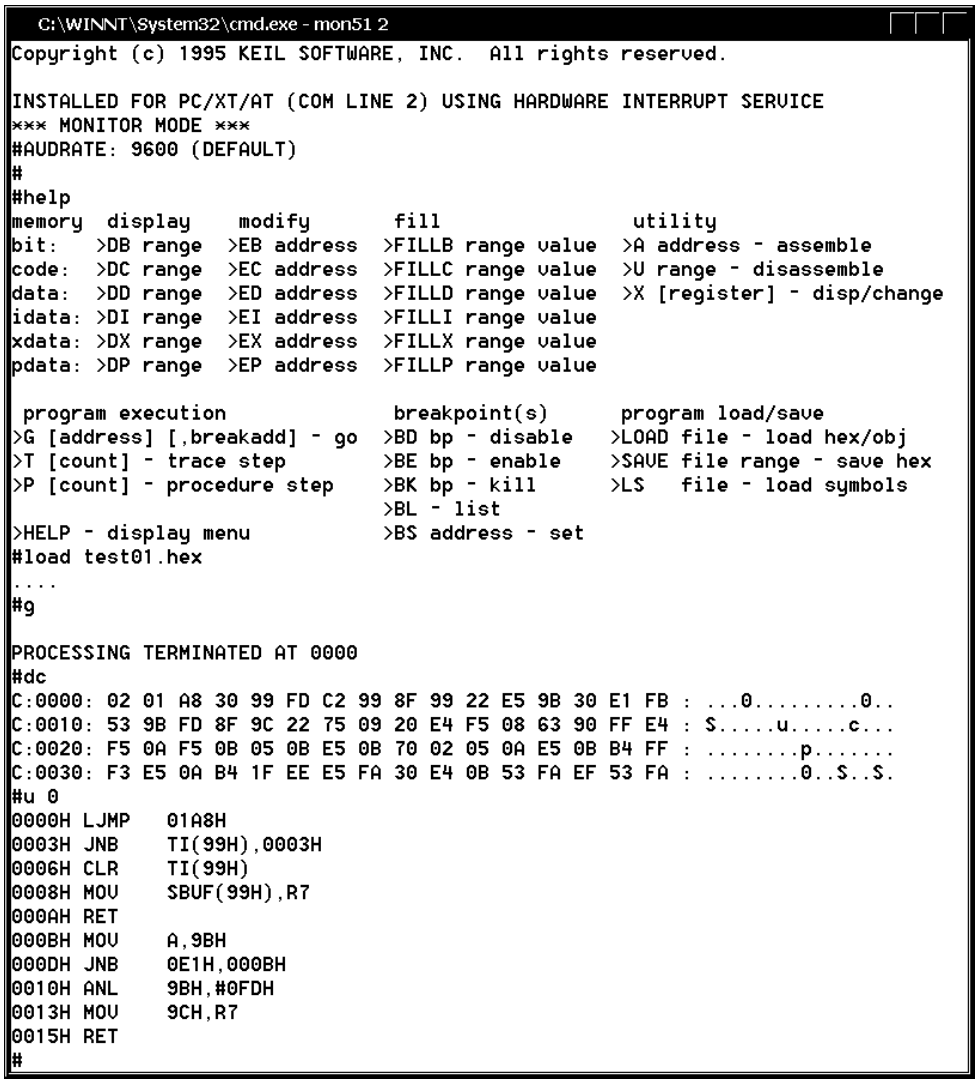
Monitor MON51.EXE jest prostym tekstowym programem monitorującym działanie zestawów ’51 przystosowanych do komunikacji z komputerem PC wg protokołu firmy Keil. Uruchomienie:

```
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 µVision) 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.



Rysunek 3: Praca z monitorem tekstowym mon51.exe.

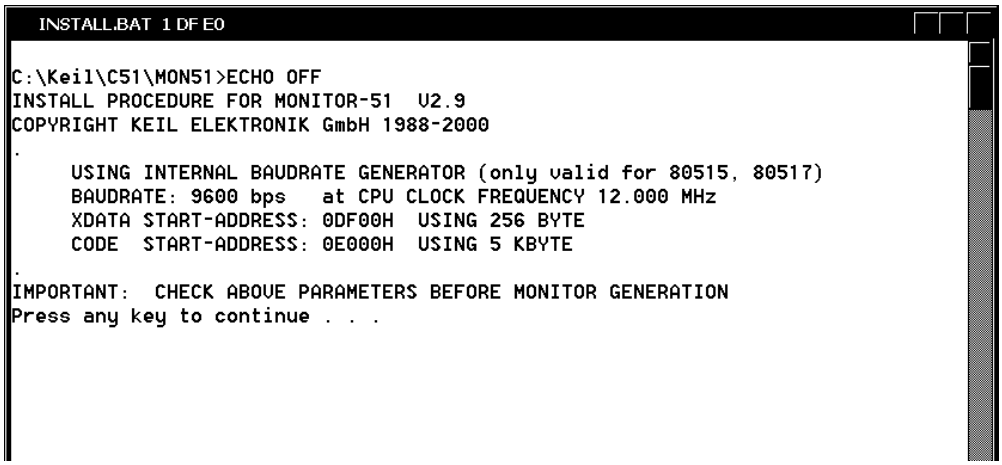
Dodatek 1: Monitor ZD537

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 µVision 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).



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 45
Change SW1
```

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 asembler 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 asemblerze. 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 asemblera.

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 counteron on P1 LEDs
45         cpl  BUZZpiezzo       ; make sound on piezzo buzzer
46         mov  A,#HOME          ; display time at fist line on LCD
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 Wyłączenie standardowych nazw rejestrów procesora ’51 bo program dotyczy procesora typu‘517.

#4 Zabronienie listowania plików typu *header (include)*.Obowiązkowe jest poznanie zawartości zbioru *reg571.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

```
; ZD537 BOARD: macros & definitions ----------------------

NE555      BIT  P1.7
BUZZpiezzo BIT  P3.2

;P6 port bit definitions
BUZZ1kHz   equ  00010000B
REDLED     equ  00000001B
RELAY      equ  01000000B

BUZZER_OFF    MACRO
               ANL  P6,#(NOT BUZZ1kHz)
              ENDM

BUZZER_ON     MACRO
               ORL  P6,# BUZZ1kHz
              ENDM

BUZZER_TOGGLE MACRO
               XRL  P6,# BUZZ1kHz
              ENDM

REDLED_ON     MACRO
               ANL  P6,#(NOT REDLED)
              ENDM


REDLED_OFF    MACRO
               ORL  P6,#REDLED
              ENDM

REDLED_TOGGLE MACRO
               XRL  P6,#REDLED
              ENDM

RELAY_OFF     MACRO                   ; asembler-style macrodefinition
               ANL  P6,#(NOT RELAY)
              ENDM


#define RELAY_ON                      // C-style macrodefinition
               ORL  P6,# RELAY

; LCD registers           ----------------------------------
LCDstatus  equ 0FF2EH
LCDcontrol equ 0FF2CH
LCDdataWR  equ 0FF2DH
LCDdataRD  equ 0FF2FH

// LCD control bytes       ---------------------------------
#define  HOME     0x80    // put curcor to second line
#define  INITDISP 0x38    // LCD init (8-bit mode)
#define  HOME2    0xc0    // put curcor to second line
#define  LCDON    0x0e    // LCD nn, cursor off, blinking off
#define  CLEAR    0x01    // LCD display clear


; firts two RTC registers  ---------------------------------
RTCxs  equ 0FF00H          ; seconds
RTCsx  equ 0FF01H
RTCxm  equ 0FF02H          ; minutes
RTCmx  equ 0FF03H
RTCxh  equ 0FF04H          ; hours
RTChx  equ 0FF05H

RTCpd  equ 0FF0DH

```

#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

```
$NOMOD51
NAME        SIO_CHAR_IO       ; basic procedures for serial comunication on SIO0 and SIO1

$NOLIST
//#include <reg517.h>          // include CPU definition file (for example, 80517)
$INCLUDE(reg517.inc)
$LIST

PUBLIC       putcharSIO0, putcharSIO1, initSIO0, initSIO1

SIO_CHAR_ROUTINES  SEGMENT  CODE
                   RSEG     SIO_CHAR_ROUTINES


;-------------------------------------------------------------------------------
; Initialize serial interface
; Using TIMER 1 to Generate Baud Rates
; Oscillator frequency = 12MHz
initSIO0:
       MOV   TMOD,#00100001B ;C/T = 0, Mode = 2
       MOV   TH1,#-13
       MOV   TL1,TH1
       SETB  TR1
       MOV   S0CON,#01010010B
       ANL   ADCON0,#80H     ;12MHz 9600bps
       RET
//#define initSIO_0 {S0CON=0x52; ADCON0|=0x80; TMOD=0x21;TH1=TL1=-13;TR1=1;} //9600 8-n-1

;-------------------------------------------------------------------------------
; Initialize serial interface 1
; Oscillator frequency = 12MHz
initSIO1:
       MOV   S1REL,#-39         ;12MHz 9600bps
       MOV   S1CON,#0B2H
       RET

//#define initSIO_1   {  S1CON=0xB2;  S1REL = -39; } //9600 8-n-1

;-------------------------------------------------------------------------------
; This routine outputs a single character through SIO0  to console.
; The character is given in A.
putcharSIO0:
          JNB  TI,$
          CLR  TI
          MOV  S0BUF,A
          RET

;-------------------------------------------------------------------------------
; This routine outputs a single character throught SIO1 to console.
; The character is given in A.
putcharSIO1:
          PUSH    ACC
          MOV     A,S1CON
          JNB     ACC.1,putcharSIO1
          ANL     S1CON,#0FDH
          POP     ACC
          MOV     S1BUF,A
          RET

       END

; 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

```
$NOMOD51
NAME        LCD_CHAR         ; LCD display procedures
$NOLIST
#include <reg517.h>      // include CPU definition file (for example, 80517)
;$INCLUDE(reg517.inc)
$LIST

PUBLIC       putcharLCD, putstrLCD, initLCD, putctrlLCD
; LCD registers           ---------------------------------
LCDstatus  equ 0FF2EH
LCDcontrol equ 0FF2CH
LCDdataWR  equ 0FF2DH
LCDdataRD  equ 0FF2FH

// LCD control bytes       ---------------------------------
#define  HOME     0x80    // put curcor to second line
#define  INITDISP 0x38    // LCD init (8-bit mode)
#define  HOM2     0xc0    // put curcor to second line
#define  LCDON    0x0e    // LCD nn, cursor off, blinking off
#define  CLEAR    0x01    // LCD display clear


LCDcntrlWR MACRO x
           LOCAL loop
loop:
           MOV    DPTR,#LCDstatus
           MOVX   A,@DPTR
           JB     ACC.7,loop       ; check if LCD busy

           MOV    DPTR,#LCDcontrol ; write to LCD control
           MOV    A, x
           MOVX   @DPTR,A
          ENDM

LCDcharWR MACRO
          LOCAL loop1,loop2

          PUSH    ACC
loop1:    MOV     DPTR,#LCDstatus
          MOVX    A,@DPTR
          JB      ACC.7,loop1            ; check if LCD busy

loop2:    MOV     DPTR,#LCDdataWR        ; write data to LCD
          POP     ACC
          MOVX    @DPTR,A
          ENDM

init_LCD  MACRO
           LCDcntrlWR #INITDISP
           LCDcntrlWR #CLEAR
           LCDcntrlWR #LCDON
          ENDM

LCD_CHAR_ROUTINES  SEGMENT  CODE
       RSEG  LCD_CHAR_ROUTINES

;-------------------------------------------------------------------------------
; Initialize serial interface
initLCD:
        init_LCD
        RET
;-------------------------------------------------------------------------------
; This routine outputs a single character to LCD.
; The character is given in A.
putcharLCD:
       LCDcharWR
       RET
;-------------------------------------------------------------------------------
; This routine outputs a control character to LCD.
; The character is given in A.
```

```
putctrlLCD:
        xch A, R2
        LCDcntrlWR R2
        xch A, R2
        RET
;-------------------------------------------------------------------------
; This routine outputs a string to LCD. String is terminated by 00H.
; The string in CODE memory is pointed by DPTR.
putstrLCD:
        CLR   A
        MOVC  A,@A+DPTR
        JZ    ?EXIT      ; check if end of string
        push  DPH
        push  DPL
        CALL  putcharLCD ; put char to LCD
        pop   DPL
        pop   DPH
        INC   DPTR
        SJMP  putstrLCD
?EXIT:  RET

       END
; this module is not finished (lack of polish characters ...)

```

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

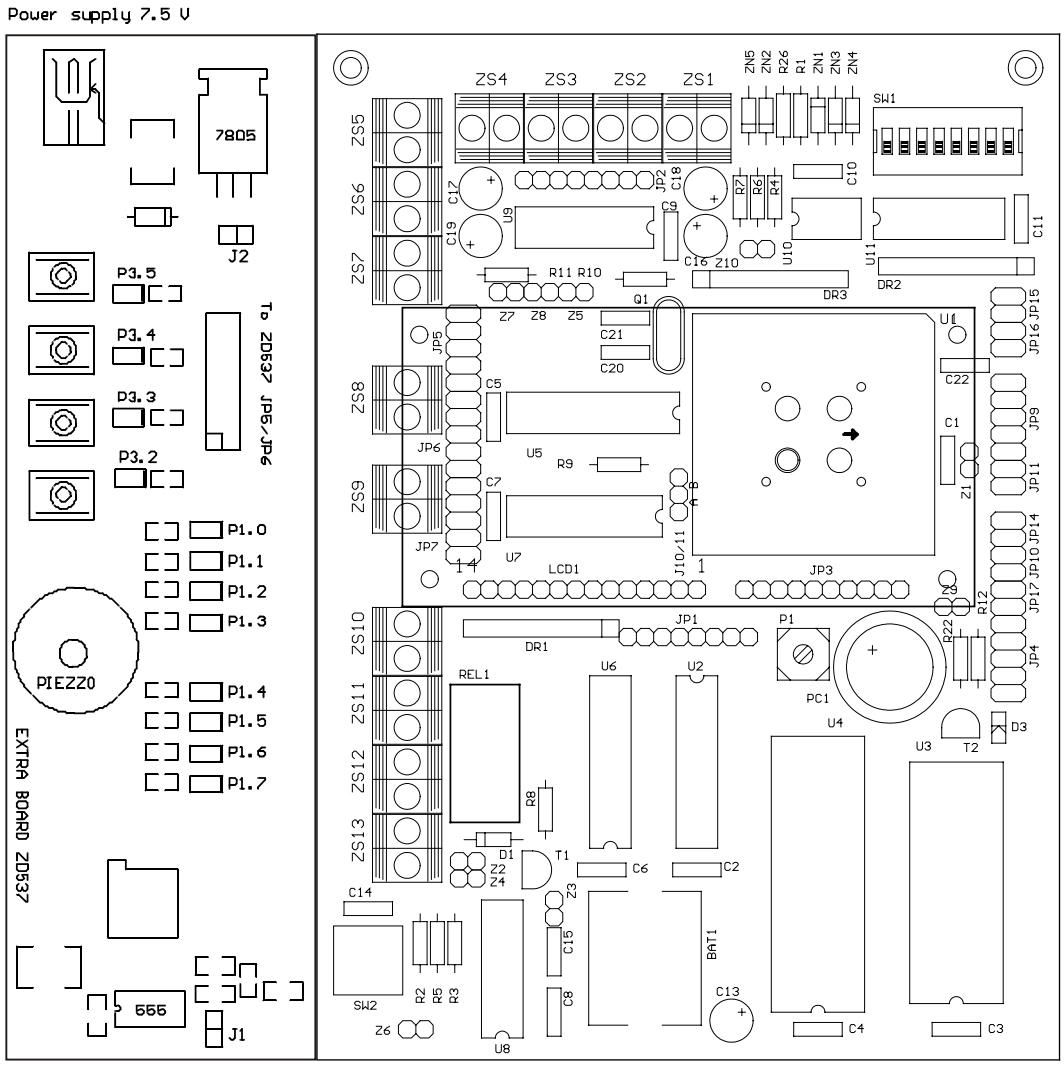
MODUŁ RTC.A51

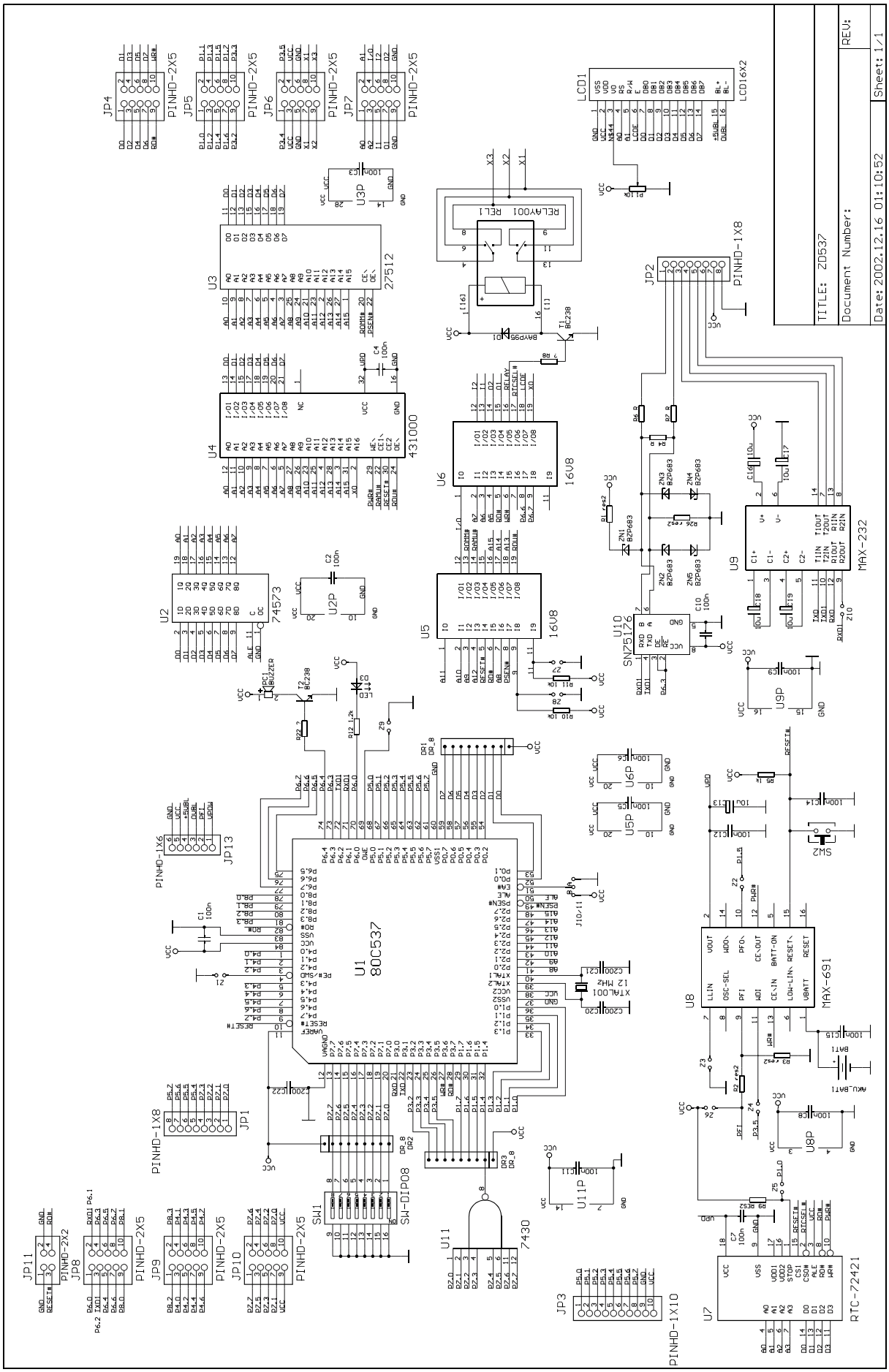
```
$NOMOD51
NAME       RTC              ; display time (minutes & seconds) on LCD

$NOLIST
#include <reg517.h>         // include CPU definition file (for example, 80517)
;$INCLUDE(reg517.inc)
$LIST

; firts two RTC registers  ----------------------------------
RTCxs  equ 0FF00H          ; seconds
RTCsx  equ 0FF01H
RTCxm  equ 0FF02H          ; minutes
RTCmx  equ 0FF03H
RTCxh  equ 0FF04H          ; hours
RTChx  equ 0FF05H

RTCpd  equ 0FF0DH

PUBLIC disp_time

disp_nibble MACRO
              movx A,@DPTR
              anl  A,#0Fh          ; select 4-bits
              orl  A,#30H          ; change to ASCII
              call putcharLCD;
            ENDM

EXTRN     CODE    (putcharLCD, putstrLCD, putctrlLCD, initLCD)   ; LCD functions


RTC_PROC  SEGMENT CODE
          RSEG  RTC_PROC

;-------------------------------------------------------------------------
; get time and it dispaly on LCD
disp_time:
        mov DPTR,#RTChx      ; get hours from RTC (higher nibble)
        disp_nibble
        mov DPTR,#RTCxh      ; get hours from RTC (lower nibble)
        disp_nibble
        mov A,#':'
        call putcharLCD;
        mov DPTR,#RTCmx      ; get minutes from RTC (higher nibble)
        disp_nibble
        mov DPTR,#RTCxm      ; get minutes from RTC (lower nibble)
```

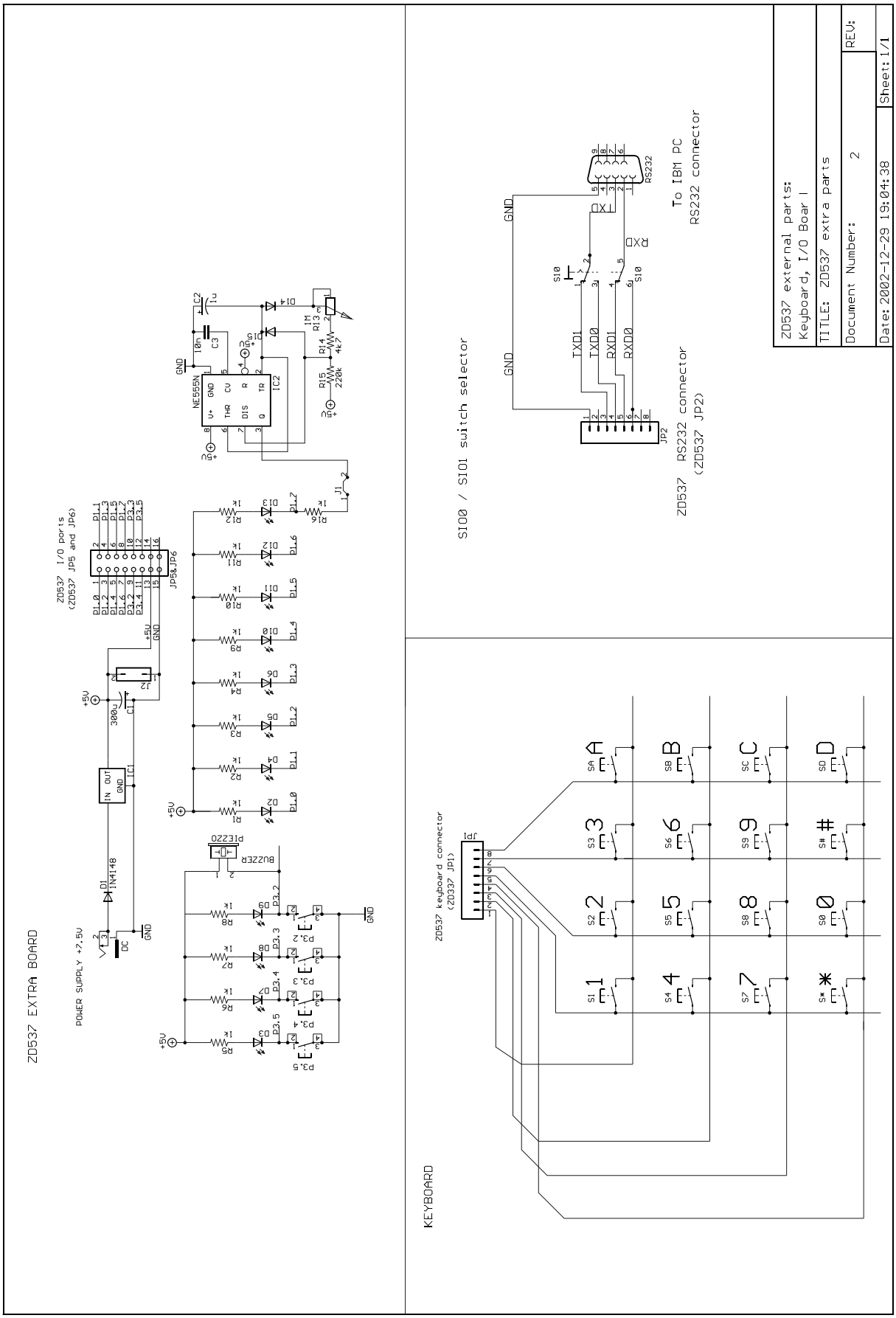
```
46          disp_nibble
47          mov A,#':'
48          call putcharLCD;
49          mov DPTR,#RTCsx      ; get seconds from RTC (higher nibble)
50          disp_nibble
51          mov DPTR,#RTCxs      ; get seconds from RTC (lower nibble)
52          disp_nibble
53          RET
54
55
56        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


Power supply 7.5 V
7805
J2
P3.5
P3.4
P3.3
P3.2
To ZD537 JP5/JP6
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
PIEZZO
EXTRA BOARD ZD537
555
J1
ZS1
ZS2
ZS3
ZS4
ZS5
ZS6
ZS7
ZS8
ZS9
ZS10
ZS11
ZS12
ZS13
SW1
SW2
LCD1
REL1
PC1
BAT1
DR1
DR2
DR3
JP1
JP2
JP3
JP4
JP5
JP6
JP7
JP9
JP10
JP11
JP14
JP15
JP16
JP17
U1
U2
U3
U4
U5
U6
U7
U8
U9
U10
U11
T1
T2
Q1
D1
D3
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C13
C14
C15
C16
C17
C18
C19
C20
C21
C22
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R22
R26
ZN1
ZN2
ZN3
ZN4
ZN5
Z1
Z2
Z3
Z4
Z5
Z6
Z7
Z8
Z10


TITLE: ZD537
Document Number:
REV:
Date: 2002.12.16 01:10:52
Sheet: 1/1
U1 80C537
U2 74573
U3 27512
U4 431000
U5 16V8
U6 16V8
U7 RTC-72421
U8 MAX-691
U9 MAX-232
U10 SN75176
U11 7430
LCD1 LCD16X2
JP1 PINHD-1X8
JP2 PINHD-1X8
JP3 PINHD-1X10
JP4 PINHD-2X5
JP5 PINHD-2X5
JP6 PINHD-2X5
JP7 PINHD-2X5
JP8 PINHD-2X5
JP9 PINHD-2X5
JP10 PINHD-2X5
JP11 PINHD-2X2
JP13 PINHD-1X6
SW1 SW-DIP08
X1 RELAY001
X2 RELAY001
X3 RELAY001
T1 BC238
T2 BC238
D1 BAYP95
D2 LED
D3 LED
BUZ1 BUZZER
BAT1 AKU_BAT1
X1 12 MHz
XTAL001


ZD537 EXTRA BOARD
POWER SUPPLY +7.5V
ZD537 I/O ports
(ZD537 JP5 and JP6)
JP5&JP6
NE555N
BUZZER
SIO0 / SIO1 switch selector
ZD537 RS232 connector
(ZD537 JP2)
To IBM PC
RS232 connector
RS232
ZD537 keyboard connector
(ZD337 JP1)
KEYBOARD
ZD537 external parts:
Keyboard, I/O Board
TITLE: ZD537 extra parts
Document Number: 2
REV:
Date: 2002-12-29 19:04:38
Sheet: 1/1

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 Jednoukładowe 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 Jednoukładowe 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 μPD431000A: 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 E^2CMOS 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.