

Moduł mikrokontrolera MCU161eth

- SAB-C161O-L16M
 - Wydajność 8 MIPS
 - UART+SPI (SSC) z niezależnymi BRG
 - 2x IN + 10... 23 I/O
 - Watchdog Timer (od 32us do 523 ms @16MHz)
 - Programowalna charakterystyka wyjść CS (logika sprzęgająca na zewnątrz CPU nie jest potrzebna)
 - 2kB On-chip RAM
- 2x UART + LPT (TL16C452 wymienny z TL16C552)
- RTC 62423
- do 256kB RAM w module mikrokontrolera z podtrzymaniem zasilania
- flash 256kB (AT29C020)
- kontroler ethernet 10Mbps/20Mbps zgodny z NE2000 z 16 kB buforem ramek,
 - automatyczna korekcja polaryzacji UTP,
 - programowanie pamięci konfiguracji 93C46 „in-system”,
 - 16 – bitowy dostęp do bufora ramek,
- 4 wyjścia CS ogólnego przeznaczenia (dane 8 lub 16 bitów)
- 104 – stykowe złącze systemowe oraz dodatkowe złącza kontrolera ethernet
- możliwość wymiany oprogramowania „in-system”
- pojedyncze zasilanie 5V, typ. 120mA (max. 150mA), w stanie „idle” typowo 40mA
- 4-ro warstwowe PCB
- wymiary 102x57 mm



1. Charakterystyka jednostki centralnej.

Uniwersalny moduł mikrokontrolera wyposażony jest w 16-bitową jednostkę RISC SAB-C1610 o wydajności 8 MIPS wyprodukowaną przez firmę Infineon. Mikrokontroler ten zapewnia 16 MB przestrzeni adresowej w architekturze von Neumanna oraz dostęp do zewnętrznych urządzeń poprzez zarówno 8-bitową magistralę danych jak i 16-bitową. Sterowanie dostępem do urządzeń zewnętrznych odbywa się przez 4 wyjścia CS o programowanej (przez odpowiednie rejestry) charakterystyce, na którą składają się:

- parametry elektryczne : czas trwania CS, czas zwolnienia magistrali, czas opóźnienia RD/WR względem CS
- logiczne : tryb pracy magistrali danych 8/16 bitów, szerokość okna przypisanego do CSx oraz adres startu okna w całkowitej wewnętrznej przestrzeni adresowej C161.

Parametry CS0 oraz rozmiar zewnętrznej magistrali adresowej konfigurowane są przez zewnętrzne rezystory R3-R23, które wymuszają odpowiednie stany logiczne na magistrali danych zapamiętywane przez mikrokontroler w cyklu reset.

Okna przypisane do wyjść CSx mogą nakładać się na siebie oraz mogą być większe niż zewnętrzna przestrzeń adresowa, której rozmiar zależy od szerokości magistrali adresowej na zewnątrz C161 i może wynosić 16, 18, 20 lub 22 bity, natomiast wewnętrzna magistrala ma zawsze szerokość 24 bitów.

Mikrokontroler, ten poza wydajną jednostką centralną z 4-poziomą kolejką instrukcji, posiada także osobny moduł mnożenia 16x16 i dzielenia 32-/16 oraz „barrel-shifter”. Jednostka centralna poprzez wewnętrzne magistrale podłączona jest do następujących układów peryferyjnych:

- synchroniczny kanał szeregowy SPI o wydajności do 4Mbaud
- asynchroniczny kanał szeregowy o wydajności do 500kbaud (2Mbaud w trybie synchronicznym)
- dwa zespoły 16-bitowych timerów - łącznie 3+2 timery
- wewnętrzny RAM o pojemności 2kB na zbiór 16 -rejestrów roboczych GPR (o programowalnym adresie) , stos maszynowy, wskaźniki PEC, dane użytkownika.

Osobny moduł stanowi rozbudowany system przerwania, umożliwiający przydzielenie każdego z 20 przerwania do jednej z 16 grup priorytetów. Istotną właściwością, z punktu widzenia wymiany danych przez łącza szeregowy, jest tryb obsługi przerwania PEC, który zamiast standardowej obsługi przerwania tj. zapisania kontekstu jednostki centralnej na stosie i uruchomieniu odpowiedniego programu obsługi przerwania, wykonuje pojedynczy transfer bajtu lub słowa z adresu źródła do adresu celu. Wskaźniki celu i źródła mogą być automatycznie zmieniane (tylko inkrementowane). Transfer PEC minimalnie obciąża jednostkę centralną, co znacznie podnosi efektywność działania mikrokontrolera. Ten rodzaj obsługi przerwania nadaje się szczególnie do obsługi portów szeregowych.

Całości obrazu dopełnia tryb BSL, który umożliwia załadowanie i uruchomienie w wewnętrznej pamięci RAM dowolnego oprogramowania, którego rozmiar nie przekracza rozmiaru wewnętrznej pamięci RAM.

2. Właściwości modułu MCU-161

2.1 Urządzenia peryferyjne

W celu osiągnięcia optymalnej prędkości mikrokontroler pracuje z niemultipleksowaną magistralą adresową, która podłączona jest przez port P1 i część lub całość portu P4. Port P0 wyprowadza magistralę danych. Jeśli wszystkie urządzenia wpięte do mikrokontrolera pracują z 8-bitową magistralą danych, wtedy port P0H (starsza połowa portu P0) może służyć jako port I/O.

Do mikrokontrolera zostały podłączone następujące elementy:

- 256kB pamięci flash (AT29C020) do CS0 – z adresu 0x00'0000 pobierana jest pierwsza instrukcja programu po resecie. CS0 musi zawsze pracować w trybie 8-mio bitowym.
- 64kB, 128kB lub 256kB pamięci statycznej RAM do CS1. Pamięć RAM może być skonfigurowana w następujący sposób:

Organizacja	Tryb pracy CS1	Zwory ZW1 i ZW2	Układy
32k x8 x2	16 bitów	Rozwarte	U7A=62256, U8A=62256
128k x8	8 bitów	ZW1	U7=681000
128k x8 x2	16 bitów	ZW2	U7=681000, U8=681000

Przy 16-bitowej magistrali danych port P3.12 (BHE/WRH#) musi pracować w trybie „WRH#”. Rozmiar zewnętrznej przestrzeni adresowej nie powinien być mniejszy niż pojemność zastosowanych pamięci. (=> rozdział 2.2)

- układy peryferyjne do CS2:

Adres (A10, A11, A12)	Urządzenie	Przerwanie	Nr wektora przerwania
0xXX00XX	TL16C452- UART0	EX1IN (P2.9)	0x19
0xXX04XX	TL16C452- UART1	EX2IN (P2.10)	0x1A
0xXX08XX	TL16C452- LPT	EX3IN (P2.11)	0x1B
0xXX0CXX	RTC 62423		
0xXX10XX	CS2A		
0xXX14XX	CS2B		
0xXX18XX	CS2C		
0xXX1CXX	RTL8019	EX5IN (P2.13)	0x1D

Tabela adresów bazowych urządzeń wpiętych do CS2

Tryb pracy CS2: 8 lub 16 bitów. Linie adresowe TL16C452 i 62423 zostały podłączone, począwszy od linii A1 CPU, w celu możliwości zaadresowania rejestrów tych układów (przy dostępie typu słowo), jeśli magistrala danych dla CS2 pracuje w trybie 16 bitowym. Linie CS2A,B,C są wyprowadzone do złącza systemowego ZL1.

Adres bazowy i rozmiar okna dla CS2 jest ustalany w rejestrach ADDRSEL2 i BUSCON2 mikrokontrolera.

- Dowlone urządzenie zewnętrzne poza modulem sterowane przez CS3 – przestrzeń adresowa jest ograniczona przez szerokość zewnętrznej magistrali adresowej i może wynosić 64kB, 256kB, 1MB lub 4MB przy szerokości magistrali danych 8 lub 16 bitów.

Układ wejścia/wyjścia 16452

Adresy względne rejestrów TL16C452

UART 0 i 1	Adres bazowy
RBR/THR /DLL	+0
IER /DLM	+2
IIR	+4
LCR	+6
MCR	+8
LSR	+10
MSR	+12
SCR	+14

Centronix	Adres bazowy
Data reg. (R/W)	+0
Status (R)	+2
Control reg.(R/W)	+4

Port P2.11 jest dostępny również na złączu systemowym ZL1 i może służyć jako port I/O (lub wejście przerwania) jeśli trójstanowe wyjście przerwania INT2 układu 16452 nie jest wykorzystane (pozostaje w stanie Hi-Z).

Watch-dog MAX691

Układ wytwarza sygnał resetu dla CPU i kontrolera ethernet oraz podtrzymuje zasilanie pamięci RAM i RTC, dodatkowo układ może sterować przerwaniem NMI CPU po zalutowaniu zwory ZW3. W sytuacji, kiedy zwora ZW3 jest zwarta, obowiązkowe jest podłączenie wejścia PFI dostępnego na złączu ZL1 przez odpowiedni dzielnik rezystorowy do punktu pomiaru napięcia zasilania.

Kontroler ethernet RTL8019

Układ RTL8019 firmy Realtek jest wysoce zintegrowanym kontrolerem ethernet zgodnym ze standardem NE2000. Umożliwia transfer nie tylko z prędkością 10Mbps ale także 20Mbps w trybie full-duplex. Kontroler posiada wbudowany układ auto-detekcji rodzaju medium transmisyjnego : 10BaseT, BNC lub AUI oraz 8 wyjść przerwania IRQ i 16 adresów bazowych do wyboru.

W mikromodule MCU161eth kontroler podłączony jest do CPU przez 16-bitową magistralę danych a magistrala adresowa kontrolera jest skonfigurowana tak, by układ pracował z adresem bazowym 0x260. Z punktu widzenia przestrzeni adresowej CPU rejestry RTL8019 występują pod adresem przypisanym do CS2 i z offsetm 0x1C00 oraz nie są widoczne jako ciągły obszar ponieważ występują tylko pod parzystymi adresami podobnie jak w przypadku układu 16452.

Wszystkie rejestry konfiguracyjne mają szerokość 8-bitów niezależnie od ustawienia bitu WTS w DCR, od którego zależy tylko szerokość rejestru „Remote DMA”. Rejestry kontrolera ethernet oraz 16452 zostały zdefiniowane w pliku „161eth.h” dołączonym do mikromodułu.

Wyjście przerwania INT3 kontrolera jest podłączone na „stałe” do CPU, mimo to istnieje możliwość zmiany konfiguracji (rejestr CONFIG1 pole IRQS) i wykorzystania linii INT4 – 7, które są wyprowadzone przez złącze ZL4. Na liniach przerwania RTL8019, które nie są aktywnym wyjściem przerwania, utrzymywany jest stan wysoki przez wewnętrzne rezystory podciągające. Stan wszystkich linii może być odczytany przez rejestr INTR na stronie 3 palety rejestrów RTL8019.

Kontroler ethernet udostępnia 3 wyjścia statusowe Led: LED0, LED1 i LED2, które zależnie od ustawienia rejestru CONFIG3 mogą mieć przypisane następujące funkcje:

- LED0 – LED_COL lub LED_LINK
- LED1 – LED_RX lub LED_CRS
- LED2 – LED_TX lub LED_MCSB.

Wyjścia Led informujące o statusie kontrolera są dostępne przez złącze ZL4 i umożliwiają bezpośrednie sterowanie diodami (rezystory 330R do ograniczenia prądu są zamontowane w mikromodule). Wyjścia Led aktywne są w stanie niskim. Złącze ZL5 służy do podłączenia interfejsu 10Base-2 (BNC) – układu 8092/8392 poza mikromodulem.

Po zresetowaniu mikromodułu program nie powinien odwoływać się do rejestrów kontrolera przez około 2ms zanim kontroler nie odczyta konfiguracji zapisanej w pamięci szeregowej 93C46. Kontroler ethernet podłączony jest do mikrokontrolera C1610 w taki sposób, że z wykorzystaniem trybu PnP można zapisać nową konfigurację do pamięci szeregowej nawet wtedy, gdy zawartość pamięci jest nieprawidłowa (np. kiedy uległa uszkodzeniu) bez konieczności demontowania pamięci 93C46 z płyty mikromodułu. Do programowania pamięci 93C46 w trybie BSL służy aplikacja „pnp-c46.bsl”

2.2 Rezystory konfiguracyjne (R3-R23)

Do zmiany konfiguracji mikrokontrolera służą następujące rezystory:

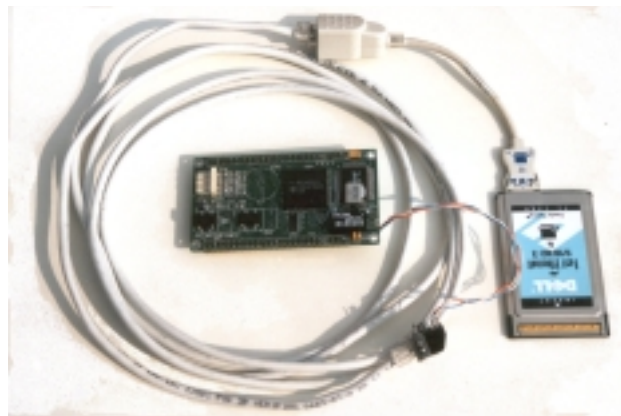
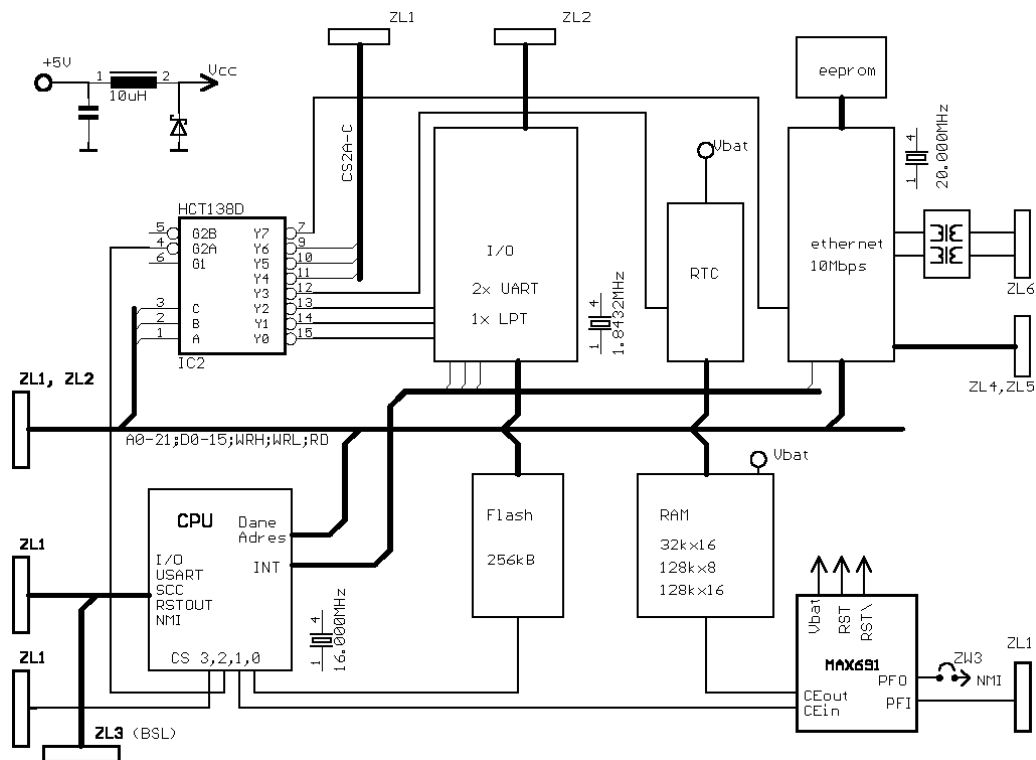
Rezystor do Vcc/Gnd	„1” (Vcc)	„0” (Gnd)
---------------------	-----------	-----------

R13/R12	Fxtal=2.. 32MHz („/2”) (GQ1/Q1)	Fxtal=1.. 16MHz („1:1”)
R16/R17 (msb) R19/R18 (lsb)	Konfiguracja SALSEL (linie adresowe segmentu w porcie P4): „1 1” – 2 linie adresowe A16, A17. Przestrzeń zewnętrzna 256kB „1 0” – 6 linii – A16.. A21. Przestrzeń zewnętrzna 4MB „0 1” – brak. Przestrzeń zewnętrzna 64kB „0 0” – 4 linie adresowe A16.. A19. Przestrzeń zewnętrzna 1MB	
R22/R23	P3.12 w magistrali 16 bitowej pracuje jako BHE#	P3.12 w magistrali 16 bitowej pracuje jako WRH#

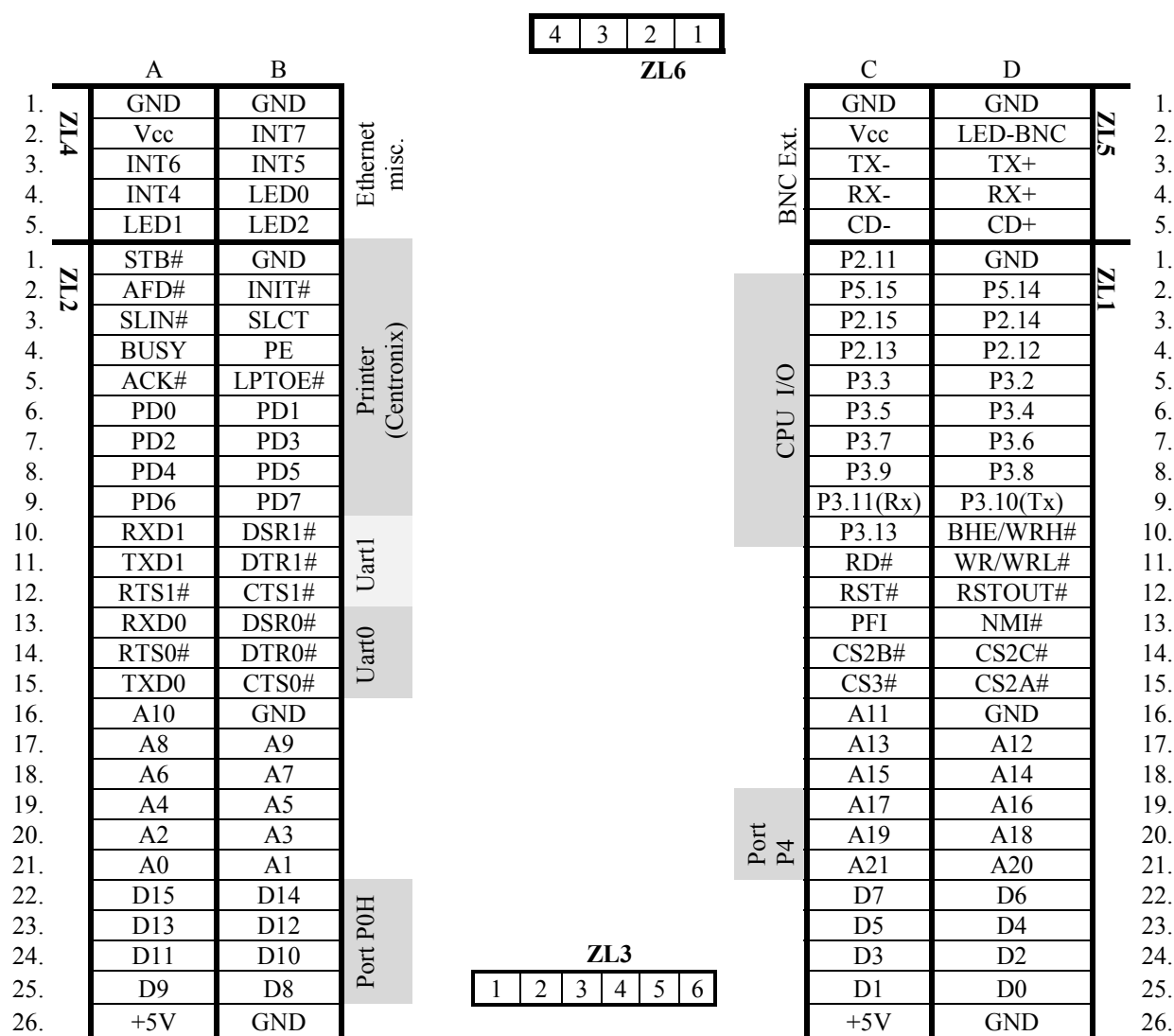
W płycie mikromodułu powinien być wlutowany tylko jeden z pary rezystorów podanych w powyższej tabeli.

Rezystory łączone z masą (R12, 17, 18, 23) powinny mieć wartość 6.8k lub mniejszą w zależności od całkowitego obciążenia magistrali danych przez sumaryczny prąd upływu wszystkich urządzeń wpiętych w tę magistralę. Dotyczy to także rezystorów R3, R4 i R6.

Nie wykorzystane linie adresowe portu P4, podobnie jak port POH, mogą pracować jako linie I/O.



Widok „z góry” złączy modułu MCU161eth



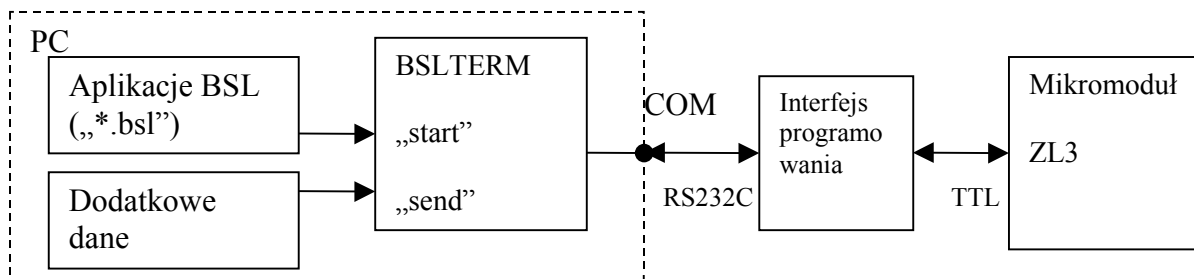
		Znaczenie
ZL3 Złącze BSL	1	Vcc
	2	Gnd
	3	Wejście danych (RxD)
	4	Wyjście danych (TxD)
	5	Wejście BSL#.
	6	Wejście RESET # (podłączać wyłącznie do wyjść typu „otwarty kolektor”)
ZL6 Złącze Ethernet – UTP	1	TPO+ (pin 1 RJ-45)
	2	TPO- (pin 2 RJ-45)
	3	TPI+ (pin 3 RJ-45)
	4	TPI- (pin 6 RJ-45)

INT4-7 – wyjścia przerwań kontrolera ethernet (opcjonalnie wejścia binarne)

Zasilanie powinno być podawane przez złącza ZL1/ZL2 (pin „+5V”).

3. Programowanie „in-system” modułu MCU-161

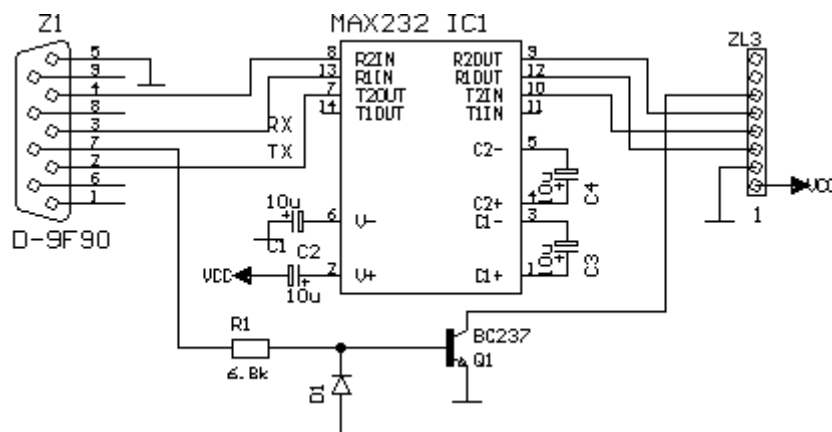
Programowanie mikromodułu „w systemie” tzn. bez użycia zewnętrznego programatora pamięci Flash jest możliwe dzięki specjalnej budowie zastosowanego mikrokontrolera SAB-C161O-L16., który umożliwia na żądanie, w tzw. trybie BSL, uruchomienie dowolnego oprogramowania przesłanego przez wbudowany w ten mikrokontroler port szeregowy. Realizacją protokołu wymiany danych w trybie BSL po stronie komputera PC zajmuje się program BSLTERM. Program ten służy do uruchomienia w mikromodule jednej z dostarczonych aplikacji, które są zapisane w plikach z rozszerzeniem „*.bsl”. Aplikacje te są programami narzędziowymi dostosowanymi do budowy mikromodułu z mikrokontrolerem C161O, które są przeznaczone m.in. do testowania oraz programowania pamięci Flash. Przed rozpoczęciem pracy z programem BSLTERM należy podłączyć złącze ZL3 mikromodułu przez adapter (interfejs programowania) do odpowiedniego portu szeregowego komputera. Program terminala BSLTERM umożliwia także wysłanie dowolnego pliku przez port szeregowy w trybie binarnym.



Idea programu terminala

Wprowadzenia złącza ZL3

ZL3	Znaczenie
1	Vcc
2	Gnd
3	Wejście danych (RxD)
4	Wyjście danych (TxD)
5	Przełącznik BSL. Stan niski na tym wejściu w połączeniu z impulsem reset aktywuje tryb BSL w mikrokontrolerze.
6	Wejście RESET (podłączać wyłącznie do wyjść typu „otwarty kolektor”)



Schemat uproszczonego interfejsu programowania

3.1 Przeznaczenie i obsługa programu BSLTERM

Wymagania :

- a) DOS +EMM386, WIN 95/98, NT 4.0, optymalny komputer: Pentium MMX ... II <500MHz
- b) Jeden port szeregowy (np. COM 1)

W skład programu wchodzi następujące pliki:

1. bsilterm.exe,
2. bsilterm.ini,
3. rtm.exe,
4. dpml16bi.ovl,

które powinny znaleźć się w tym samym katalogu oraz pliki dodatkowe:

1. flonline.bsl, f2online.bsl,
2. r16test.bsl, r8test.bsl, h16ld161.bsl, h8ld161.bsl

Pliki z grupy dodatkowych są aplikacjami uruchamianymi w mikromodule przy pomocy programu terminala „BSLTERM”. Plik bsilterm.ini określa konfigurację startową programu terminala i składa się z dwóch sekcji: APP i shortcuts podobnie do typowych plików „ini”. W sekcji APP zawarta jest konfiguracja portu komunikacyjnego. Parametr „loader” powinien mieć wartość 32. W sekcji „shortcuts” do klawiszy funkcyjnych przyporządkowane są komendy. Ze względu na wygodę obsługi programu jeden z klawiszy funkcyjnych (np. F6) powinien mieć przypisaną komendę „start <ścieżka>*.bsl”, gdzie ścieżka oznacza katalog, w którym znajdują się pliki typowych aplikacji BSL jak : flonline.bsl i ram161o.bsl. Przykład pliku konfiguracyjnego „BSLTERM.INI”:

```
[APP]                                <- sekcja „APP”
baud=28800                            <- prędkość portu szeregowego
bslappsdefdir=e:\164\bslapps         <- domyślny katalog aplikacji BSL
loader=32
port=2
ttymode=0

[shortcuts]                            <- sekcja skrótów
F1=start *
F10=start n:\164\hexld\hexld.bsl
F2=start n:\164\memtst\rttest.bsl
F3=start n:\164\rdback\rdback.bsl
F4=send n:\ormosk.164\telegram\sysinfo.bin
F5=send n:\ormosk.164\telegram\*.bin
F6=start e:\164\bslapps\*.bsl
```

Po uruchomieniu "bsilterm" , w oknie terminala wyświetlana jest informacja o rozpoznanym porcie "COMx : <prędkość>; INT = przerwanie; typ portu". Jeśli wskazywane przerwanie wynosi 0, wtedy program nie będzie mógł komunikować się z otoczeniem. W środowiskach windows przyczyną może być aktywność innego programu wykorzystującego ten sam port szeregowy (np. sesja DOS). Może zdarzyć się, że program nie rozpozna typu portu (zwykle jest to 16550), ale nie jest to szczególnie istotna informacja, o ile rozpoznane jest przerwanie. Do zmiany portu służy opcja "Port" w głównym menu (Alt-O). Prędkość transmisji danych zmienia się komendą „baud” wpisywaną w oknie terminala. Terminal

wyświetla skrócony opis komend po wpisaniu słowa „help”. Aby program mógł komunikować się z podłączoną płytą musi być wybrany odpowiedni port, taki, do którego podłączony jest interfejs programowania .

Uruchomienie aplikacji w mikromodule rozpoczyna się od wydania w oknie terminala komendy „start” z parametrem „*.bsl” – wtedy otworzy się okno wyboru bądź z nazwą konkretnego pliku aplikacji. Komendę można również wprowadzić przy pomocy zdefiniowanych w pliku bsfterm.ini klawiszy skrótów (np. F6).

W przypadku gdy aplikację wybiera się z okna wyboru plików, należy po zatwierdzeniu pliku aplikacji jeszcze raz klawiszem „enter” potwierdzić komendę w oknie terminala. W wyniku wpisania/ potwierdzenia komendy „start” terminal rozpoczyna proces ładowania i uruchamiania aplikacji wskazanej w parametrze komendy „start”. Proces ten przebiega następująco:

1. Nawiązanie komunikacji z mikromodulem, terminal wyświetla "CPU id: "
2. Jeśli mikrokontroler w mikromodule jest sprawny wraz z jego podstawowym otoczeniem (układ resetu, oscylator, rezystory konfiguracyjne i złącze ZL3) oraz jest sprawny adapter (interfejs programowania), wtedy terminal wyświetla bajt identyfikujący mikrokontroler, który w przypadku SAB-C1610 wynosi B5h lub D5h. Inne wartości bajtu identyfikacji świadczą o nieprawidłowej prędkości transmisji lub uszkodzeniu interfejsu programowania. Jeśli terminal otrzyma bajt odpowiedzi, wtedy przechodzi do uruchomienia aplikacji. Dalsza akcja zależy od rodzaju przesłanej aplikacji do mikrokontrolera.
3. Jeśli po czasie ok. 2 sek pojawi się napis "timeout" w miejscu bajtu identyfikacji, wtedy komunikacja z mikrokontrolerem nie powiodła się.. Przyczyną może być niewłaściwa prędkość komunikacji, niewłaściwy port lub przyczyny opisane w punkcie 2.

Kombinacje klawiszy Alt-R i Alt-B służą do wygenerowania sygnału reset i sygnału reset z aktywacją trybu BSL. Dyrektywa „start” aktywuje tryb BSL niezależnie od skrótu Alt-B.

Do wysłania dowolnego pliku przez port szeregowy służy komenda „send” z parametrem zawierającym nazwę pliku lub „*”. Plik jest wysyłany w trybie binarnym.

Program BSLTERM umożliwia automatyczne uruchomienie aplikacji BSL jeśli nazwa tej aplikacji zostanie przekazana jako pierwszy parametr w linii poleceń. Ścieżka do aplikacji BSL pobierana jest z parametru o nazwie „bslappsdefdir” w pliku konfiguracyjnym „BSLTERM.INI”. Jeśli w linii parametrów pojawi się drugi parametr zawierający nazwę pliku (opcjonalnie z pełną ścieżką dostępu) wtedy plik ten jest automatycznie wysyłany komendą „send” w momencie gdy uruchomina, w mikromodule, aplikacja BSL wyśle do terminala słowo „send ”.

3.2 Aplikacja FxONLINE.BSL

Do programowania pamięci flash typu AT29C010 służy "flonline.bsl". Uruchomiona aplikacja w module MCU161 wysyła do programu terminala tekst „CS0:AT29C010 On Line download utility © KB 2000” oraz w następnej linii „Send *.c10”. Po potwierdzeniu tej linii (enter) otwiera się okno wyboru, w którym należy wybrać odpowiedni plik do programowania. Po zatwierdzeniu tego pliku aplikacja rozpoczyna proces programowania pamięci flash. Stan procesu programowania wyświetlany jest w oknie terminala w postaci wierszy numerów zaprogramowanych sektorów pamięci. Numery są wyświetlane w systemie szesnastkowym z towarzyszącym znacznikiem po symbolu „:”. Znacznikiem może być jeden z następujących znaków:

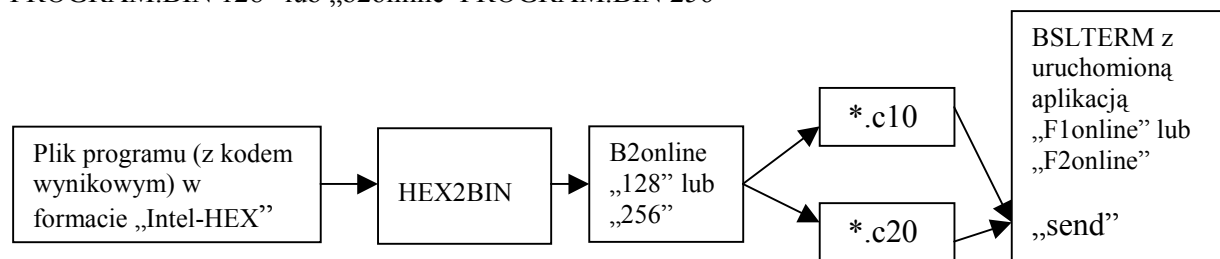
- „ ” (spacja) - sektor zaprogramowany został prawidłowo,

- „V” - oznacza błąd weryfikacji sektora - problemy z zewnętrznymi magistralami danych/adresu lub pamięcią flash
- „C” - błąd sumy kontrolnej sektora - problemy z transmisją danych PC <-> interfejs <-> mikromoduł

Pamięć Flash jest zaprogramowana prawidłowo, jeśli nigdzie nie wystąpią znaczniki „V” lub „C”. Zakończenie programowania sygnalizowane jest przez aplikację tekstem „press reset to start program”. Aby uruchomić program zapisany w pamięci Flash, można wyłączyć/ włączyć zasilanie modułu lub z poziomu terminala kombinacją klawiszy Alt-R wygenerować sygnał resetu.

Aplikacja "f2online.bsl" przeznaczona jest do programowania pamięci Flash typu AT29C020 jak i AT29C040, w których sektor ma rozmiar 256 bajtów. Zasada działania aplikacji jest taka sama jak w przypadku "f1online.bsl". Różnicą jest tekst zgłoszenia, gdzie występuje pamięć AT29C020 oraz rozszerzenie pliku do programowania – „*.c20” zamiast „*.c10”.

Pliki z rozszerzeniem „*.c10” i „*.c20” są generowane przez program „b2online.exe”, który w linii argumentów pobiera nazwę pliku binarnego oraz rozmiar sektora (128 lub 256), np. „b2online PROGRAM.BIN 128” lub „b2online PROGRAM.BIN 256”



Aplikacja	Błąd	Potencjalna przyczyna
„Fxonline”	„V”	Błąd weryfikacji sektora – uszkodzona pamięć Flash, zwarcie/przerwa w magistrali danych/adresów
	„C”	Błąd sumy kontrolnej sektora – uszkodzony interfejs programowania, błędy transmisji (np. niedopasowanie prędkości transmisji > 2.5 %)

3.3 Aplikacja R16TEST.BSL/ R8TEST.BSL

Aplikacje są przeznaczone do przetestowania zainstalowanej w mikromodule pamięci statycznej RAM. Pierwsza z nich służy do testowania pamięci o organizacji 16-to bitowej. Po uruchomieniu w mikromodule program aplikacji wykorzystuje zainicjowany wcześniej port szeregowy do prezentacji wyników testu pamięci, który składa się z trzech faz: „checkboard1”, „checkboard2”, „EX-OR”. W każdej z faz zapisywany jest wzór testowy do całej dostępnej pamięci, która wynosi 256kB x16 w przypadku R16TEST.BSL i 128kB x8 dla R8TEST.BSL, następnie program weryfikuje poprawność zapisu w blokach po 16 kB od najwyższego adresu 0x7FFFF. Przy każdym z adresów może pojawić się „OK” lub informacja o adresie błędu i wartości odczytanej z magistrali danych oraz wartości prawidłowej. Pierwsze dwie fazy służą do sprawdzenia poprawności magistrali danych. Ostatnia faza testu polega na zapisie wzoru, który zależy od miejsca zapisu do pamięci co umożliwi określenie uszkodzenia w magistrali adresowej po analizie adresu błędu weryfikacji komórki pamięci.

W przypadku mikromodułu z pamięcią o rozmiarze 64kB (x16) pierwsze dwie fazy testu nie powinny wykazywać żadnego błędu natomiast w ostatniej fazie testu (EX-OR) błąd będzie zasygnalizowany przy przekroczeniu granicy 64kB tj. przy adresie 0x6FFFF.

Poprawny wynik testu pamięci jest podstawą prawidłowego działania mikromodułu.

3.4 Aplikacja H16LD161.BSL/ H8LD161.BSL

Aplikacje HxxLD161.bsl są przeznaczone do ładowania plików w formacie Intel-Hex do pamięci RAM mikromodułu. Najwyższy adres w pliku nie powinien przekraczać 0x3FFFF. Aplikacje funkcjonują na podobnej zasadzie jak FxONLINE.BSL. Po uruchomieniu mikromodułu odsyła linię zawierającą komendę „send *.h86”.

3.5 Aplikacja PNP-C46.BSL

Aplikacja służy do programowania pamięci szeregowej 93C46 przy kontrolerze ethernet RTL8019 z wykorzystaniem mechanizmu PnP. Po załadowaniu aplikacji do mikromodułu może pojawić się napis:

„NIC not found” oznaczający uszkodzenie kontrolera ethernet lub „RTL8019 ready” i „send *.c46” oznaczający gotowość do zaprogramowania pamięci plikiem z rozszerzeniem C46. Po wybraniu odpowiedniego pliku następuje programowanie i odczytanie zawartości pamięci z wyświetleniem jej w oknie terminala. Z mikromodułem dostarczone są dwa pliki C46: „blank.c46” zawierający tylko bajty 0xFF oraz „def_cfg.c46”, który ustawia następujące rejestry RTL8019:

CONFIG 1:	0xB7	(IRQEN=1, INT3, 0x260)
CONFIG 2:	0x20	(BROM disabled)
CONFIG 3:	0x00	(LED0=LED_COL, LED1=LED_RX, LED2=LED_TX)
CONFIG 4:	0x00	(IOMS=0)

3.6 Programy RAMBOOT.ASM i RAMAPPL.ASM

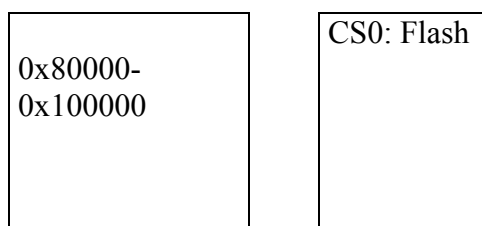
Powyższe programy, umieszczone w katalogu „develop.161”, dają możliwość uruchamiania oprogramowania użytkownika na etapie opracowywania przez przeładowanie do pamięci RAM mikromodułu bez konieczności ciągłego programowania pamięci flash, co jest wolniejszym procesem. Program RAMBOOT musi zostać zaprogramowany do pamięci nieulotnej ponieważ jest uruchamiany w obsłudze sprzętowego przerwania reset i wykonuje następujące czynności:

- konfiguruje External Bus Controller dla CS1 i CS0
- inicjuje port szeregowy (57600bps @Q=16.000MHz)
- wykonuje skok do pamięci RAM pod adres „targetaddr” (0x2000)

Program RAMBOOT dodatkowo zawiera szkielet obsługi przerwania NMI, przerwań stosu maszynowego, pułapek klasy B. Procedura generacji pliku wynikowego do programowania pamięci flash - „ramboot.c20” przy pomocy aplikacji „f2online.bsl” znajduje się w pliku „makefile”.

Program RAMAPPL jest przykładowym programem, który po skompilowaniu, można przeładować do pamięci RAM przy pomocy aplikacji „H16LD161.BSL” (lub „H8” dla pamięci 128kx8) i uruchomić przez wygenerowanie sygnału „reset”. Sygnał „reset” może zostać wysłany z programu terminala „bslterm” kombinacją klawiszy Alt-R lub w związku z wyłączeniem zasilania. W ostatnim przypadku należy liczyć się z utratą zawartości pamięci RAM (jeśli moduł nie ma baterii).

Mapa pamięci tworzona przez program RAMBOOT



0x7FFFF	CS1: CS1: 256kBx16 RAM
0x3FFFF	
0x0	

W obszarze o adresach 0x40000-0x7FFFF występuje kopia pamięci RAM. Program RAMBOOT musi być dostosowany do konfiguracji pamięci RAM w mikromodule (szerokość magistrali danych).

3.7 Dostosowanie μ Vision 2 do programu Bslterm

Program terminala można automatycznie uruchomić z właściwymi parametrami ze środowiska μ Vision 2 jeśli zostanie dodany do opcji „tools” w następujący sposób:

1. wybrać z opcji „tools”: „customize tools menu”
2. w oknie o tym samym tytule wybrać ikonę „insert”, następnie wprowadzić nazwę pod jaką będzie występował program w oknie „tools”
3. w linii „command” powinna znaleźć się nazwa programu terminala ze ścieżką np.: „C:\TOOLS\BSLTERM.EXE”
4. w linii „arguments” powinna znaleźć się nazwa aplikacji i np.: #H. Zmienna bslappsdefdir pliku BSLTERM.INI powinna wskazywać właściwy katalog z aplikacjami BSL.

Przykład:

```
Command:      C:\TOOLS\BSLTERM.EXE
Arguments:    h161d161 #H
```

Powyższa konfiguracja spowoduje automatyczne uruchomienie aplikacji ładującej do pamięci plik w formacie Intel-HEX, którego nazwa wraz ze ścieżką przekazana będzie w drugim parametrze. Środowisko μ Vision pod parametr „H” wstawi nazwę pliku „hex”, który powstaje w wyniku kompilacji otwartego projektu „uv2”.

Wpis :

```
Command:      C:\WINDOWS\Pulpit\BSLTERM.PIF
Arguments:    flonline $H@H.c10
```

powoduje uruchomienie terminala z aplikacją programownia pamięci Flash AT29C010 i automatyczne załadowanie pliku o nazwie projektu i rozszerzeniu „C10” z katalogu projektu. Dokładny opis słów kluczowych, które mogą wystąpić w parametrach komendy znajdują się w książce „ μ Vision2 getting started” w rozdziale 4.

Inną metodą jest utworzenie skrótu do programu terminala (jak w drugim przykładzie) , w którym można zapisać ustawienia dotyczące np. ekranu i podanie (wpisanie) pełnej ścieżki do tego skrótu w linii „command”.