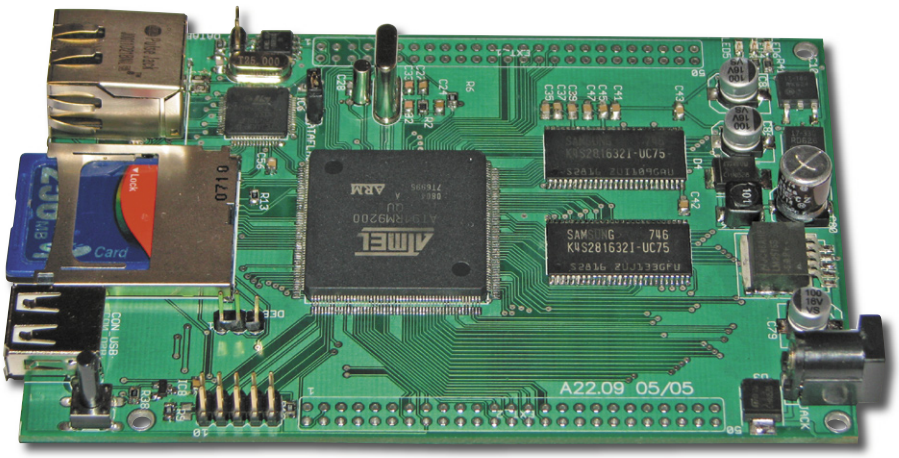
SARGE

Jednoukładowy komputer 32-bitowy z procesorem ARM9

Dzięki znacznemu spadkowi cen mikrokontrolery z rdzeniem ARM stały się bardzo popularne wśród elektroników-konstruktorów. Wbudowane peryferia pozwoliły na miniaturyzację obwodów PCB, przy jednoczesnym zachowaniu dużej funkcjonalności. Zastosowanie 32-bitowej architektury oraz jednostki MMU do zarządzania pamięcią (Memory Management Unit), pozwoliło na uruchamianie 32-bitowych systemów operacyjnych takich, jak Windows CE lub Linux. Niniejszy artykuł przedstawia mikrokomputer bazujący na układzie AT91RM9200 firmy Atmel, pracujący pod kontrolą systemu operacyjnego Linux.

Rekomendacje:
projekt dedykujemy wszystkim zainteresowanym wykorzystaniem mikrokontrolerów z rdzeniem ARM.



Mikrokontroler AT91RM9200

Układ AT91RM9200 ma wydajność 200 MIPS przy częstotliwości zegara 180 MHz, 16 kB wbudowanej pamięci SRAM oraz 16 kB pamięci cache dla instrukcji i danych. W pamięci ROM umieszczono bootloader pierwszego poziomu, wspomagający start głównego systemu z pamięci typu DataFlash, NAND-Flash, I²C EEPROM, Bootloader umożliwia także załadowanie programu przez port szeregowy RS232 za pomocą protokołu XMODEM lub przez złącze USB przy pomocy protokołu DFU.

Wśród podstawowych peryferii zastosowanego mikrokontrolera ARM należy wymienić: kontroler pamięci SDRAM, kontroler kart MMC/SD/SDIO, kontroler Flash oraz NAND-Flash, układ Ethernet MAC 10/100 Base-T, klient i host USB 2.0, 3 synchroniczne kontrolery szeregowe SSC (*Synchronous Serial Controler*), 4 USART-y, 1 master/4 slave SPI, 2 trójkanałowe liczniki/zegary, interfejs TWI (funkcjonalny odpowiednik I²C), interfejs JTAG, magistralę zgodną z EBI.

Większość wbudowanych peryferii jest dostępna przez dobrze znany z mikrokontrolerów AVR selektor portów wejścia/wyjścia PIO. Narzucone w ten sposób ograniczenie uniemożliwia użycie wszystkich peryferii w projekcie. Najczęściej jednak nie jest to potrzebne. Wszystkie możliwości mikrokontrolera opisane są w dokumentacji producenta, jednak jest to ponad 600 stron lektury [1].

Schemat blokowy układu przedstawiono na **rys. 1**. Możemy na nim wyróżnić poszczególne bloki funkcjonalne wymienione wcześniej. Na szczególną uwagę zasługuje jednostka debugowania DBGU. W normalnym trybie pracy umożliwia analizę działającego programu tak na poziomie języka maszynowego, jak i wysokiego poziomu, gdyż fizycznie jest to port szeregowy, przez który w wybranych miejscach programu można wysyłać komunikaty testowe. Ta sama jednostka może posłużyć do załadowania programu w trakcie realizacji programu bootloadera zaszytego wewnątrz pamięci ROM mikrokontrolera.

Blok zasilania

Układ AT91RM9200 wymaga dwóch napięć zasilających. Do zasilania rdzenia procesora używane jest napięcie 1,8 V, natomiast do komunikacji układu z pozostałymi elementami systemu (np. pamięć SDRAM lub kontroler warstwy PHY sieci Ethernet) wymagane jest napięcie o wartości zależnej od akceptowanych przez te układy. W przypadku opisywanego projektu jest to 3,3 V. Zastosowane zasilacze

AVT–5175

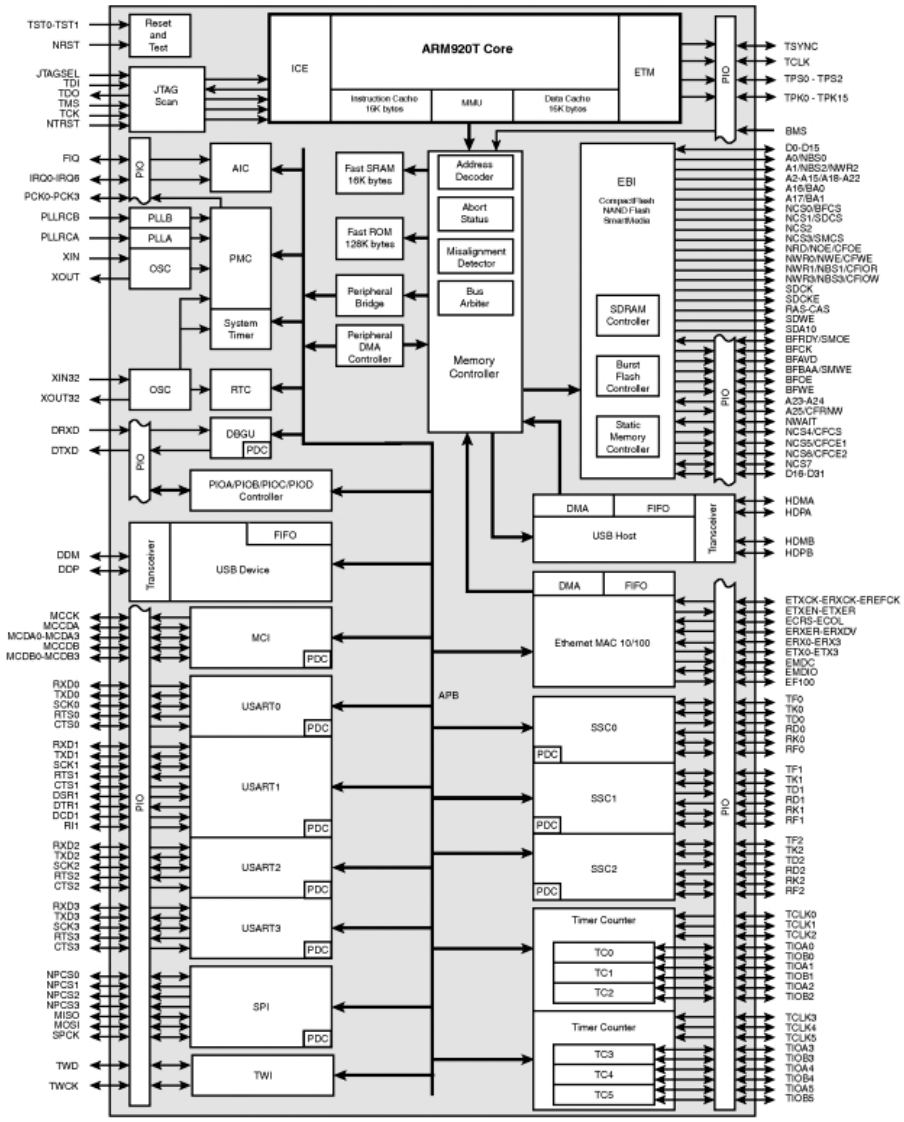
W ofercie AVT:
AVT–5175A – płytka drukowana

PODSTAWOWE PARAMETRY

- Pojedyncza, dwustronna płytka drukowana o wymiarach (126×80) mm
- Napięcie zasilania 9…12 VDC, pobór prądu około 0,1 A
- Praca pod kontrolą systemu Linux
- Interfejsy: Ethernet, USB, czytnik kart SD, złącze do układów zewnętrznych
- Procesor AT91RM9200 taktowany zegarem 18,432 MHz
- Pamięć RAM 16 MB, pamięć Flash do 16 MB

PROJEKTY POKREWNE wymienione artykuły są w całości dostępne na CD

Tytuł artykułu	Nr EP/EdW	Kit
EpFlashLink – programator JTAG dla mikrokontrolerów STR9 (rdzeń ARM9)	EP 9/2006	AVT-947
ARM na DIP-ie	EP 4/2005	AVT-1411
ARMputer z mikrokontrolerem LPC22xx i dodatkowymi Pamięciami SRAM/Flash	EP 6-7/2007	AVT-989
BF100 – linuksowy ARMputer	EP 3-4/2008	AVT-5128



Rys. 1. Schemat blokowy układu mikrokomputera

LDO zostały dobrane tak, aby zapewnić odpowiednią wydajność prądową zarówno dla rdzenia mikrokontrolera (wg danych producenta maks. 500 mA), jak i dla pozostałych układów zasilanych napięciem 3,3 V.

Stabilizatory napięcia IC11 i IC10 zasilane są z zasilacza impulsowego IC9 na układzie LM2575 o wydajności prądowej do 3 A i napięciu wyjściowym 5 V.

Kondensatory C79, C80, C82, C84 służą do filtracji napięć wyjściowych ze stabilizatorów, natomiast C81, C83 i C85 filtrują sygnały o wysokich częstotliwościach, które mogą się pojawić w obwodzie zasilania wskutek działania układów cyfrowych. Diody LED4...LED6 sygnalizują obecność poszczególnych napięć zasilania.

Blok mikrokontrolera AT91RM9200

Na **rys. 2** przedstawiono główny blok układu zawierający mikrokontroler AT91RM9200. Napięcie zasilające rdzeń o wartości 1,8 V doprowadzone jest do wejść VDDCORE0...4 i filtrowane przez dławik L1 oraz kondensatory C1 i C2. Dodatkowo każde wejście zasilania VDDCORE ma dołączony kondensator filtrujący o pojemności 100 nF (C9, C10, C12, C14, C15). Napięcie zasilające magistrale komunikacyjne I/O o wartości 3,3 V jest doprowadzone do wejść VDDIOP0...5 oraz VDDIOM0...6 i filtrowane za pomocą kondensatorów C11 i C13. Podobnie jak poprzednio, do zasilania podłączono dodatkowe kondensatory o pojemności 100 nF (C3...C8, C16...C22).

Do pracy procesora AT91 wymagane są dwa sygnały zegarowe. Sygnał zegarowy niskiej częstotliwości generowany przy użyciu kwarcu o częstotliwości 32768 Hz jest doprowadzony do XIN32/XOUT32. Generator pracuje w typowej konfiguracji Pierce'a, więc dodatkowo wymaga jeszcze dołączenia pojemności C28 i C31. Zegar niskiej częstotliwości jest używany do uruchomienia bootloadera drugiego poziomu, zaimplementowanego przez użytkownika, oraz steruje pracą wbudowanych w strukturę generatorów PLL (PLLA, PLLB) taktujących wewnętrzne bloki funkcjonalne o wiele wyższą częstotliwością.

Sygnał zegarowy wysokiej częstotliwości jest generowany z użyciem zewnętrznego kwarcu o częstotliwości 18,432 MHz, podłączonego do XIN/XOUT. Układ wykorzystuje kondensatory C32 i C33.

Filtry PLL stanowią odpowiednio elementy: R1, C23, C26 (PLLA), R2, C24, C27 (PLLB). Elementy dobrano tak, aby dla częstotliwości 18,432 MHz uzyskać wewnętrzny sygnał zegarowy MCK o częstotliwości 180 MHz dla rdzenia procesora oraz PLLA i PLLB, 48 MHz dla kontrolera USB (zarówno hosta, jak i urządzenia). Wartości elementów zostały obliczone przy pomocy kalkulatora filtra PLL [2] dostępnego na stronie internetowej producenta.

Zastosowany mikrokontroler jest wyposażony w interfejs JTAG, co znacznie ułatwia proces debugowania programów, jak i działania samego systemu przy użyciu odpowiednich narzędzi, takich jak OpenOCD [3] (*On-Chip debugging*) współdziałających z debuggerem gdb będącym składową środowiska kompilatora gcc. Złącze JTAG przedstawiono na **rys. 2**. Rezystory R7...R10 ustalają poziom sygnałów przy niepodłączonym interfejsie (rezystory R7 i R8 mogą zostać pominięte przy zastosowaniu standardowego interfejsu JTAG np. Wiggler). Na tym samym rysunku przedstawiono złącza rozszerzeń oznaczone jako EXT_1 oraz EXT_2, umożliwiające wykorzystanie komputera jako modułu. Na złącza wyprowadzono: sygnały danych, adresowe, sterujące magistralą, UART, I²C, SPI oraz I²S. Wszystkie wyprowadzone sygnały nie mają driverów, co wymaga zachowania uwagi przy dołączaniu modułów rozszerzających funkcje układu.

Pamięć SDRAM

Mikrokontroler posiada wbudowaną pamięć SRAM o pojemności 16 kB i może funkcjonować korzystając tylko z niej, jednak marnotrawstwem byłoby nie wykorzystanie pełnego jego potencjału i nie podłączenie większej ilości pamięci, wykorzystując w tym celu wbudowany kontroler SDRAM.

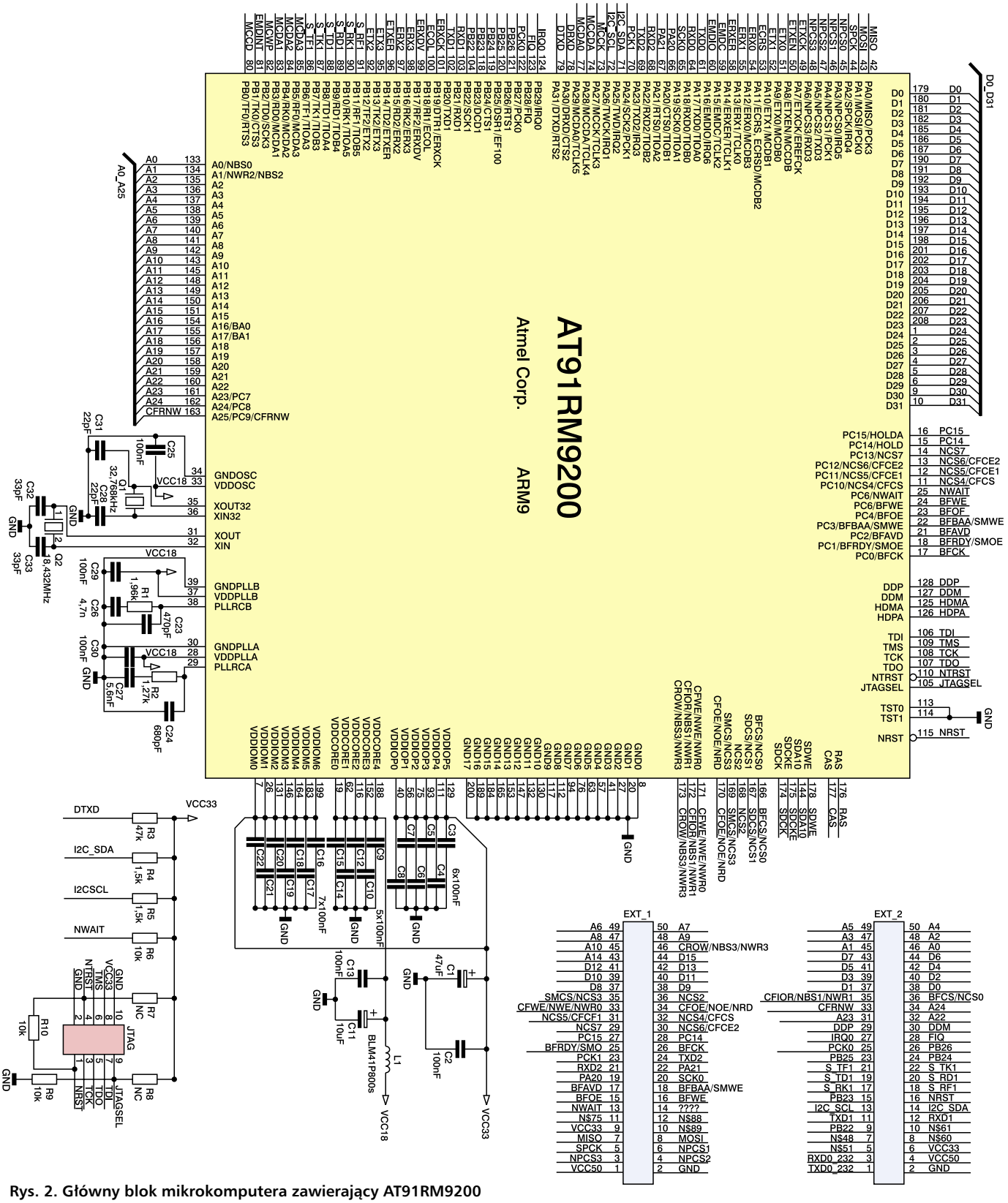
Wspomniany kontroler potrafi obsłużyć większość spotykanych pamięci SDRAM. W urządzeniu zastosowano dwa układy SDRAM K4S561632H-UC75000 mające czas odświeżania 75 ns, o 16-bitowej szynie danych. Połączenie pamięci w parę umożliwiło rozszerzenie szyny danych do 32 bitów. Ma to wpływ na wydajność podczas transferu bloków danych, znacznie go przyśpieszając. Istotne jest również znacznie szybsze adresowanie

pamięci, choć realizuje to kontroler sprzętowy bez udziału programisty. Pamięci SDRAM przedstawiono na **rys. 3**.

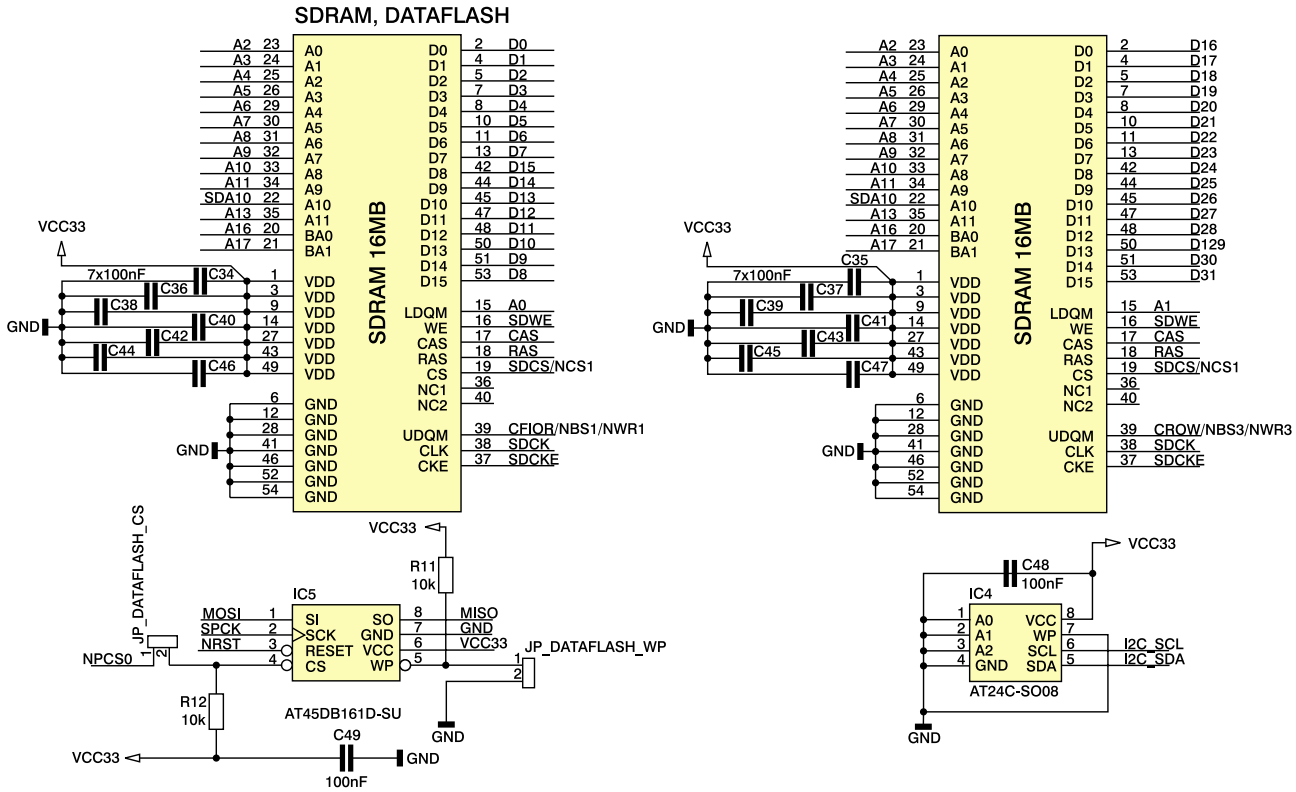
Układy pamięci IC2 i IC3 są zasilane napięciem 3,3 V. Kondensatory C34...C47 o pojemności 100 nF filtrują napięcie zasilające.

Sygnały SDWE, SDCS, SDCK, SCKE, CAS, RAS generowane są przez wewnętrzny kontroler SDRAM i doprowadzone do obu układów. Magistrala adresowa jest tutaj wykorzystana nietypowo z punktu adresowania liniowego zwykłych pamięci SRAM – linie adresowe A2...A10, SDA10, A13 odpowiadają adresowi kolumn, natomiast A0...A12 oraz A16 i A17 adresowi bloku BA1...BA2. Linie adresu fizycznego A0, A1 w połączeniu z sygnałem UDQM (CRW/NBS1/NWR1 dla IC2, CRW/NBS3/NWR3 dla IC3) w tym przypadku służą do wybierania układu pamięci, na którym odbywa się w danym momencie operacja zapisu/odczytu (lub inna przewidziana przez kontroler SDRAM – np. odświeżanie stanu danej kolumny pamięci).

Mikrokontroler AT91RM9200 do poprawnej pracy z pamięciami SDRAM wymaga odpowiedniego skonfigurowania kontrolera pamięci – odbywa się to najczęściej w bootloaderze drugiego poziomu.



Rys. 2. Główny blok mikrokomputera zawierający AT91RM9200



Rys. 3. Schemat połączeń pamięci SDRAM

Pamięć DataFlash

Wbudowany kontroler SPI pracując w trybie master obsługuje cztery urządzenia poprzez sterowanie sygnałami wyboru: NPCS0...NPCS3. Bootloader zapisany w pamięci ROM, podczas rozruchu mikrokontrolera umożliwia odczyt pamięci DataFlash aktywowanej sygnałem NPCS0. Na rys. 3 przedstawiono pamięć AT45DB161D (IC5) o pojemności 16 Mb. Istnieje możliwość zastosowania innych układów, jednak ich pojemność nie powinna być mniejsza ze względu na wymagania systemu operacyjnego Linux (1,5 MB). Zwierając JP_DATAFLASH_WP można zabezpieczyć pamięć IC5 przed zapisem. Aby uniknąć problemów w sytuacji, gdy do pamięci została zapisana błędna zawartość programu bootloadera drugiego lub trzeciego poziomu, wprowadzono dodatkową możliwość odłączenia pamięci DataFlash poprzez rozwarcie zwory JP_DATAFLASH_CS.

Niewykrycie przez bootloader z pamięci ROM bootowalnego programu w żadnej z pamięci zewnętrznych powoduje próbę pobrania programu przez RS232 za pomocą protokołu XMODEM z parametrami transmisji: 115200,8,n,1. Na podłączonym terminalu można zaobserwować wtedy charakterystyczną sekwencję znaków „CCCCCCCCCC", sygnalizującą próbę nawiązania połączenia.

W mikrokomputerze wykorzystano również szeregową pamięć EEPROM z interfejsem I²C. Na rys. 3 jest to układ IC4. Linie interfejsu wymagają użycia rezystorów podciągających. Rolę tę pełnią R4 i R5 (rys. 2). Zastosowano pamięć typu AT24C08 o pojemności 1 kB. Jest ona używana m.in. do zapamiętania adresu MAC karty sieciowej.

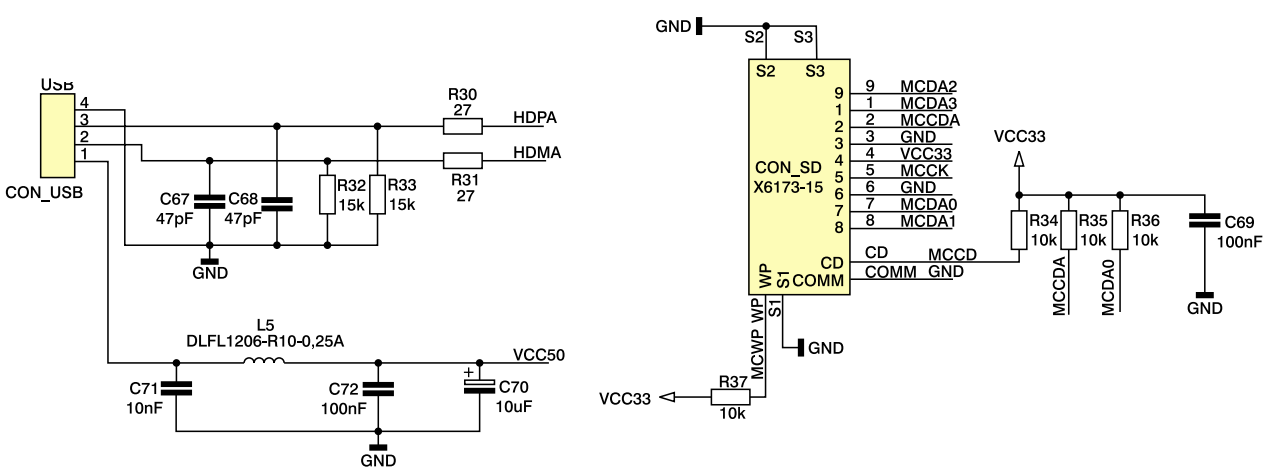
Host USB

Do niedawna host USB był rzadkością w mikrokontrolerach, a oprogramowanie go, jak i urządzenia podłączanego do hosta wymagało sporo wysiłku. Zastosowanie systemu operacyjnego, który od wielu lat wspiera standard USB oraz posiada sterowniki do wielu urządzeń, pozwala skupić się na innych problemach i uruchomić urządzenia, takie jak: dyski USB, klucze Bluetooth i inne. Na **rys. 4** przedstawiono złącze hosta USB CON_USB.

Rezystory R30 oraz R31 ograniczają prąd, a R32 oraz R33 wyznaczają impedancję linii sygnałów D+, D-. Kondensatory C67, C68 służą do odfiltrowania sygnałów wysokich częstotliwości. Kondensatory C70...C72 i cewka L5 filtrują napięcie zasilające dołączane urządzenie z interfejsem USB.

Kontroler kart MMC/SD

Sprzętowy kontroler kart DC ułatwia ich obsługę. Prezentowany mikrokomputer używa karty jako pamięci masowej. Znajduje się na niej



Rys. 4. Złącze hosta USB (CON_USB)

główny system plików, w skład którego – poza systemem operacyjnym – wchodzą także aplikacje i dane użytkownika.

Jako gniazdo kart użyto złącza X6173-15 (rys. 4). Do złącza doprowadzono wszystkie sygnały danych (MCDA0...MCDA3, MCDDA) i sygnał zegarowy MCCK. Dzięki sygnałowi MCCD można wykryć fakt obecności karty, natomiast MCWP sygnalizuje jej zabezpieczenie przed zapisem.

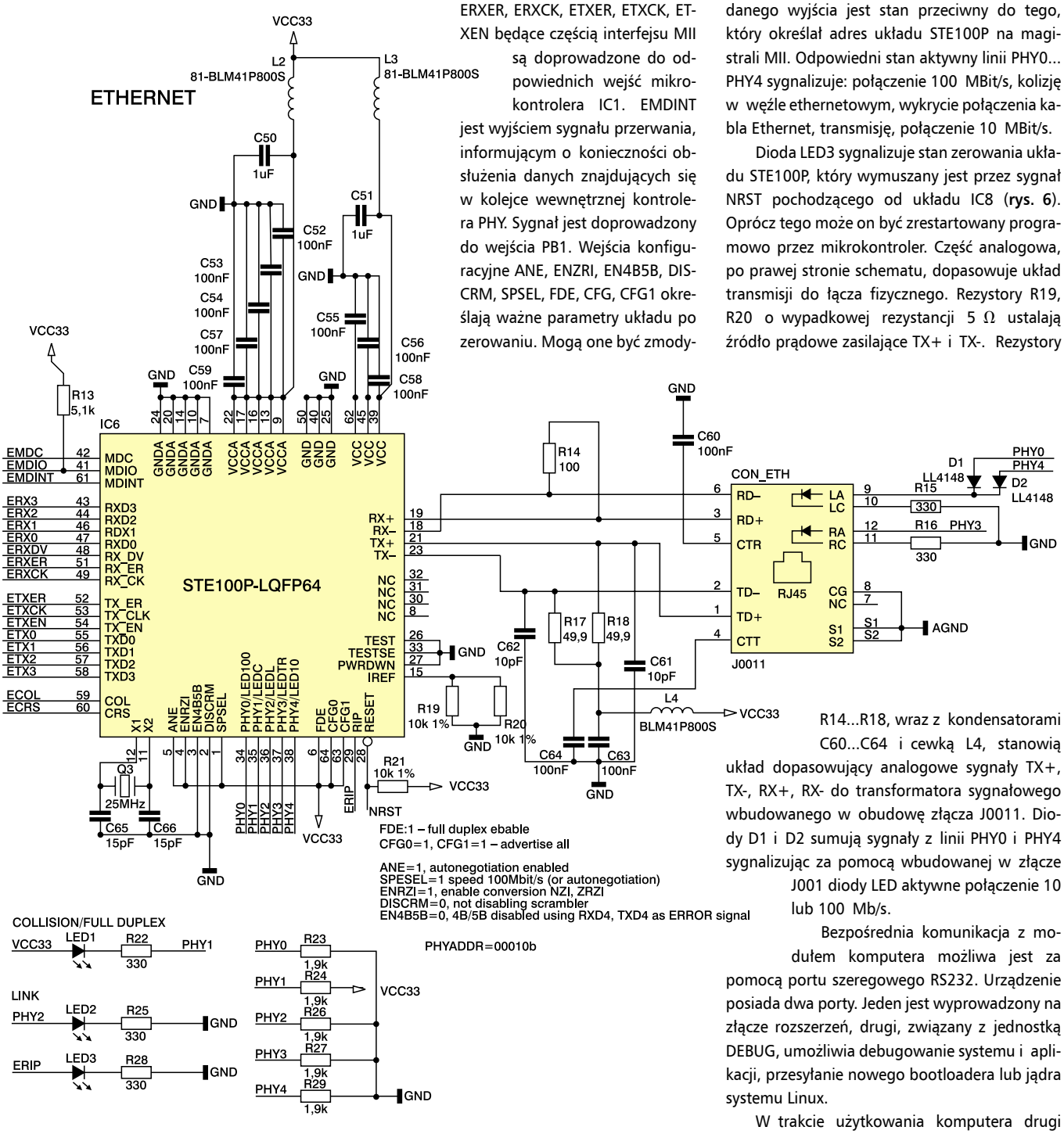
Ethernet

Mikrokontroler AT91RM9200 jest wyposażony w kontroler sieci Ethernet 10/100 Mbit/s. Są to interfejsy MII (*Media Independent Interface*) oraz RMII (*Reduced Media Independent Interface*) pozwalające na podłączenie zewnętrznego kontrolera fizycznej warstwy sieci PHY. Producenci układów PHY zachowują zgodność ze standardem, i dlatego użycie dowolnego układu zmusza jedynie do drobnej modyfikacji programu. Do mikrokomputera, ze względu na możliwość łatwego zakupu, wybrano układ STE100P, mogący pracować z prędkościami 10 i 100 Mb/s, zarówno z użyciem interfejsu MII, jak i RMII.

Na **rys. 5** przedstawiono typową aplikację układu STE100P. Układ jest zasilany napięciem 3,3 V. Część analogowa jest zasilana przez filtr L2-C50, natomiast część cyfrowa L3-C51. Układ wymaga odrębnego sygnału zegarowego o częstotliwości 25 MHz. Jest on generowany przy pomocy rezonatora kwarcowego Q3. Sygnały ETX0...3, ERX0...3, EMDC, EMDIO, ERXDV, ERXER, ERXCK, ETXER, ETXCK, ETXEN będące częścią interfejsu MII są doprowadzone do odpowiednich wejść mikrokontrolera IC1. EMDINT jest wyjściem sygnału przerwania, informującym o konieczności obsłużenia danych znajdujących się w kolejce wewnętrznej kontrolera PHY. Sygnał jest doprowadzony do wejścia PB1. Wejścia konfiguracyjne ANE, ENZRI, EN4B5B, DISCRM, SPSEL, FDE, CFG, CFG1 określają ważne parametry układu po zerowaniu. Mogą one być zmodyfikowane w trakcie działania kontrolera PHY poprzez wysłanie odpowiednich wartości do określonych rejestrów (robi to także mikrokontroler z poziomu systemu operacyjnego w trakcie uruchamiania sterownika karty sieciowej). Wejścia PHY0...4 ustalają adres układu STE100P na magistrali MII. Można do niej podłączyć do 31 układów PHY. Adres 0x00 jest zarezerwowany.

Wejścia adresowe PHY0...4 mają podwójną funkcję – w trakcie zerowania określają adres urządzenia na magistrali MII, w tym przypadku jest to adres 0x02. Adres jest ustalany przez stany logiczne na wejściach PHY0...4. Druga funkcja tych wejść objawia sie już po zerowaniu. Są one wykorzystywane do sygnalizacji stanu kontrolera Ethernet, przy czym stanem aktywnym danego wyjścia jest stan przeciwny do tego, który określał adres układu STE100P na magistrali MII. Odpowiedni stan aktywny linii PHY0... PHY4 sygnalizuje: połączenie 100 MBit/s, kolizję w węźle ethernetowym, wykrycie połączenia kabla Ethernet, transmisję, połączenie 10 MBit/s.

Dioda LED3 sygnalizuje stan zerowania układu STE100P, który wymuszany jest przez sygnał NRST pochodzącego od układu IC8 (**rys. 6**). Oprócz tego może on być zrestartowany programowo przez mikrokontroler. Część analogowa, po prawej stronie schematu, dopasowuje układ transmisji do łącza fizycznego. Rezystory R19, R20 o wypadkowej rezystancji 5 Ω ustalają źródło prądowe zasilające TX+ i TX-. Rezystory R14...R18, wraz z kondensatorami C60...C64 i cewką L4, stanowią układ dopasowujący analogowe sygnały TX+, TX-, RX+, RX- do transformatora sygnałowego wbudowanego w obudowę złącza J0011. Diody D1 i D2 sumują sygnały z linii PHY0 i PHY4 sygnalizując za pomocą wbudowanej w złącze J001 diody LED aktywne połączenie 10 lub 100 Mb/s.



Rys. 5. Typowa aplikacja układu STE100P

Bezpośrednia komunikacja z modułem komputera możliwa jest za pomocą portu szeregowego RS232. Urządzenie posiada dwa porty. Jeden jest wyprowadzony na złącze rozszerzeń, drugi, związany z jednostką DEBUG, umożliwia debugowanie systemu i aplikacji, przesyłanie nowego bootloadera lub jądra systemu Linux.

W trakcie użytkowania komputera drugi port można wykorzystać jako zwykły port szere-

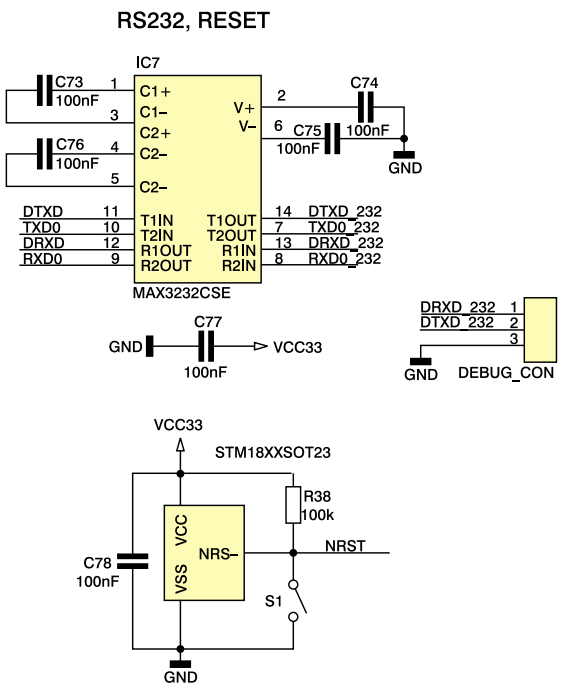
Tab. 1. Struktura wektora przerwań nr 6

31 ... 17	16 ... 13	12 ... 8	7 ... 0
Rozmiar strony DataFlash	Liczba stron	Zarezerwo-wane	Liczba bloków 512 b

gowy. Na rys. 6 przedstawiono układ IC7 będący konwerterem poziomów TTL/RS232. Na tym samym schemacie przedstawiono układ nadzoru napięcia zasilania STM18XX (IC8) generujący sygnał NRST (reset) przed ustaleniem się wartości napięcia zasilania 3,3 V. Sygnał jest doprowadzony do wszystkich układów wymagających zerowania.

Bootloader

AT91RM9200 (jak cała seria AT91) posiada zapisany w pamięci ROM bootloader pierwszego poziomu, który jest wywoływany po zerowaniu mikrokontrolera. Funkcjonalność bootloadera sprowadza się do wstępnej inicjalizacji mikrokontrolera, zegarów, układów PLL itd. W dalszej części jego działania następuje wyszukanie programu startowego w różnych obsługiwanych przez mikrokontroler pamięciach: DataFlash, NAND-Flash, I²C-EEPROM, a następnie podejmowana jest próba odebrania programu przez port szeregowy DEBUG z użyciem protokołu XMODEM i na końcu przy pomocy portu USB protokołem DFU.



Rys. 6. Schemat połączeń układu drivera RS232 i kontrolera napięcia zasilania IC8

Aby pamięć została uznana za zawierającą program startowy, musi zawierać na swoim początku odpowiednią strukturę. Pamięć musi zawierać 6 procedur obsługi przerwań/wyjątków sprzętowych zaczynających się skokiem do określonego adresu lub rozkazem LDR. Szósty wektor, który normalnie nie jest używany, zawiera informacje na temat rozmiaru programu startowego w pamięci DataFlash. W **tab. 1** przedstawiono wymaganą postać 32-bitowego fragmentu pamięci reprezentującej 6. wektor obsługi przerwań.

Pierwsze osiem bitów zawiera liczbę bloków, jakie należy przepisać z pamięci DataFlash do SRAM. Pojedynczy blok ma rozmiar 512 bajtów. Bity 13...16 określają numer strony pamięci DataFlash, od której należy rozpocząć odczyt. Bity 17...31 określają rozmiar strony pamięci DataFlash. Rozmiar strony zależy od zastosowanego typu układu AT45. W **tab. 2** przedstawiono parametry pamięci w zależności od typu.

Odpowiednio przeliczoną wartość 32-bitową należy zapisać w miejscu szóstego wektora przerwań. Tak przygotowana pamięć DataFlash zostanie rozpoznana przez bootloader pierwszego poziomu jako program do załadowania. W przedstawionym projekcie, głównym programem wykonywanym z pamięci DataFlash jest zmodyfikowany bootloader autorstwa Darrella Harmona [3].

Bootloader 2 poziomu

Program stanowiący bootloader drugiego poziomu w swojej głównej części inicjalizuje kontroler SDRAM i konfiguruje jego rejestry ustalając parametry zastosowanych pamięci. W dalszej części, w trybie terminalowym udostępnia funkcje, takie jak:

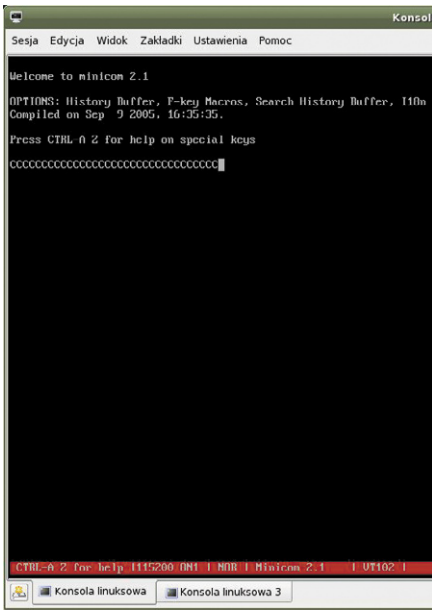
- zapisanie bootloadera drugiego poziomu do pamięci Flash,
- zapisanie bootloadera trzeciego poziomu (w tym przypadku U-boot, zostanie omówiony w dalszej części artykułu),
- zapisanie do wydzielonego obszaru jądra systemu Linux,
- wykonanie bootloadera trzeciego poziomu,
- przeprowadzenie testu pamięci SDRAM.

Pierwszym krokiem po uruchomieniu mikrokomputera jest przesłanie bootloadera drugiego poziomu. Na **rys. 7** przedstawiono zrzut ekranu z terminala podłączonego przy pomocy portu szeregowego do portu DEBUG. Można zauważyć charakterystyczny dla protokołu XMODEM ciąg znaków wywoławczych ‚C'.

W trakcie transmisji ciągu znaków wywoławczych przez mikrokomputer należy z poziomu programu terminalowego wysłać plik bootloadera *loader.bin* za pomocą wcześniej wspomnianego protokołu. Plik jest dostępny na płycie oraz (wraz z resztą plików) na stronie autora [4] w dziale „Downloads".

Po wczytaniu bootloadera powinno pojawić się menu z dostępnymi opcjami (**rys. 8**).

Ponieważ bootloader został wczytany do pamięci SRAM i został natychmiast wykonany, nie jest możliwe jego bezpośrednie skopiowanie do pamięci DataFlash. W celu jego zapisania należy

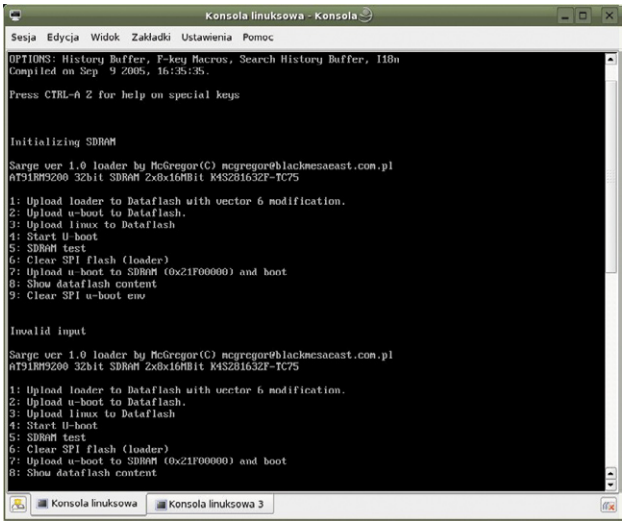


Rys. 7.

Tab. 2. Konfiguracja pamięci w zależności od użytych układów

Urządzenie	Pojemność	Rozmiar strony (bajty)	Liczba stron
AT45DB011B	1 Mb	264	512
AT45DB021B	2 Mb	264	1024
AT45DB041B	4 Mb	264	2048
AT45DB081B	8 Mb	264	4096
AT45DB161B	16 Mb	528	4096
AT45DB321B	32 Mb	528	8192
AT45DB642	64 Mb	1056	8192
AT45DB1282	128 Mb	1056	16384

czytaj na stronie 102



Rys. 8.

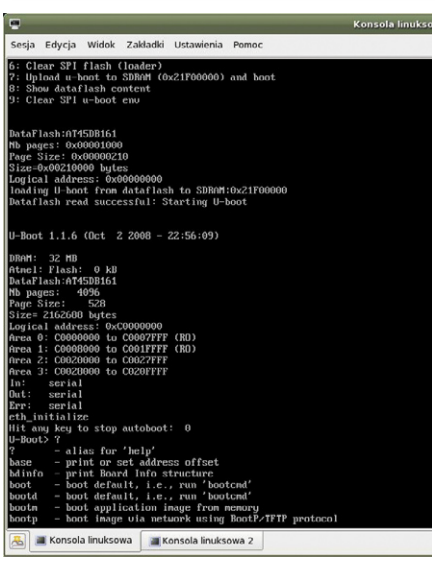
ponownie przesłać go za pomocą terminala. Procedurę rozpoczynamy od wyboru opcji 1 z menu, a następnie w programie terminalowym korzystamy z funkcji przesyłania pliku za pomocą protokołu XMODEM. Po przesłaniu pliku, na ekranie terminala pojawi się odpowiednia informacja o poprawnym lub błędnym zakończeniu operacji. Jeżeli operacja zakończyła się poprawnie, to od tego momentu możemy swobodnie zerować komputer, ponieważ bootloader po każdym starcie zostanie załadowany z pamięci DataFlash.

Bootloader U-boot

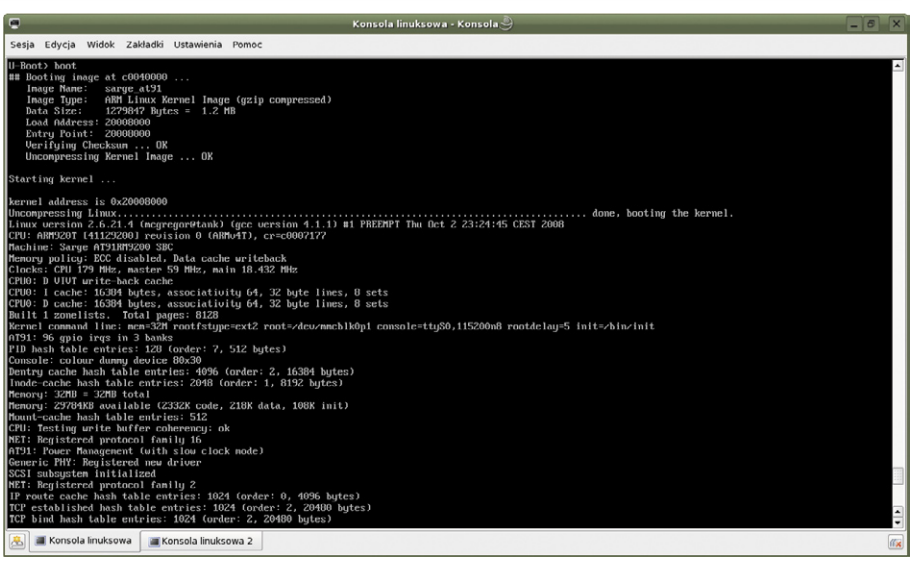
Kolejnym etapem jest zainstalowanie bootloadera 3 poziomu. Jest on bardziej zaawansowany, ładowany i uruchamiany przez znacznie prostszy i jednocześnie rozmiarami mniejszy bootloader 2 poziomu. U-boot – znany inaczej jako uniwersalny bootloader, jest rozwijany przez wielu programistów na świecie od kilku lat i dostępny jako OpenSource. Obsługuje kilkadziesiąt znanych platform sprzętowych i sporą grupę peryferii. Umożliwia między innymi bootowanie przez sieć Ethernet, czy modyfikację stanu pamięci NAND-Flash, DataFlash, EEPROM. Budowa modularna umożliwia łatwe dodanie kolejnych funkcji. W przyjętym zastosowaniu domyślna rola bootloadera U-boot polega na załadowaniu jądra systemu Linux z pamięci DataFlash do pamięci SDRAM, rozpakowanie (jeżeli jądro było skompresowane), a następnie przekazanie parametrów startowych systemu odczytanych ze specjalnego obszaru pamięci DataFlash i uruchomienie samego systemu. W dalszej części sterowanie programem przekazane jest systemowi operacyjnemu Linux, i jest to proces na tyle skomplikowany, że omówienie go wybiega poza ramy artykułu.

Zapisanie programu u-boot odbywa się przez wykorzystanie opcji nr 2 w bootloaderze pierwszego poziomu. Po wciśnięciu klawisza 2 należy przesłać plik *u-boot.bin* przy pomocy protokołu XMODEM. Po zapisaniu programu do pa-

WYKAZ ELEMENTÓW
Rezystory SMD 0805
R1: 1,96 kΩ
R2: 1,27 kΩ
R3: 47 kΩ
R4, R5: 1,5 kΩ
R6, R34, R35, R37, R9...R12: 10 kΩ
R7, R8: NC
R13: 5,1 kΩ
R14: 100 Ω
R15, R16, R22, R25, R28: 330 Ω
R17, R18: 49,9 Ω
R19...R21: 10 kΩ 1%
R23, R24, R26, R27, R29: 1,9 kΩ
R30, R31: 27 Ω
R32, R33: 15 kΩ
R36: 10 kΩ/NC
R38: 100 kΩ
R39: 220 Ω
R40: 510 Ω
R41: 680 Ω
Kondensatory SMD
C2...C10, C12...C22, C25, C29, C30, C34... C49, C52...C60, C63, C64, C69, C72...C78, C81, C83, C85: 100 nF 805
C23: 470 pF 805
C24: 680 pF 805
C26: 4,7 nF 805
C27: 5,6 nF 805
C28, C31: 22 pF 805
C32...C33: 33 pF 805
C50, C51: 1 μF 805
C61, C62: 10 pF 805
C65, C66: 15 pF 805
C67, C68: 47 pF 805
C71: 10 nF 805
C1: 47 μF/16 V
C11, C70: 10 μF/16 V
C79, C82, C84: 100 μF
C80: 1000 μF
Półprzewodniki SMD
D1, D2: LL4148
D3, D4: STPS340S
IC1: AT91RM9200
IC2, IC3: K4S281632I-UC75000
IC4: AT24C-SO08
IC5: AT45DB161D-SU
IC6: STE100P
IC7: MAX3232CSE
IC8: STM18XX
IC9: LM2575HVTO
IC10: SPX1117
IC11: NCP1117DT33
LED1, LED2, LED4: 0805 SMD YELLOW
LED3, LED6: 0805 SMD GREEN
LED5: 0805 SMD RED
Inne
CON_ETH: J0011D21BNL Gniazdko RJ-45 zintegrowane z transformatorem i diodami LED
CON_SD: Gniazdo SDCARD
CON_USB: złącze USB A do druku
DEBUG_CON: goldpin 1×3
EXT_1, EXT_2: goldpin 2×25
JP_DATAFLASH_CS: goldpin 1×2 plu jumper
JP_DATAFLASH_WP: goldpin 1×2
JTAG: goldpin 2×5
POWER_JACK: gniazdo zasilania
S1: mikroswitch
L1, L5: EMIRHW35X3 Koralik ferrytowy SMD
L2...L4: EMISMB403025 Koralik ferrytowy SMD
L6: 100 μH SMD
Q1: kwarc 32,768 kHz
Q2: kwarc 18,432 MHz
Q3: kwarc 25 MHz



Rys. 9.



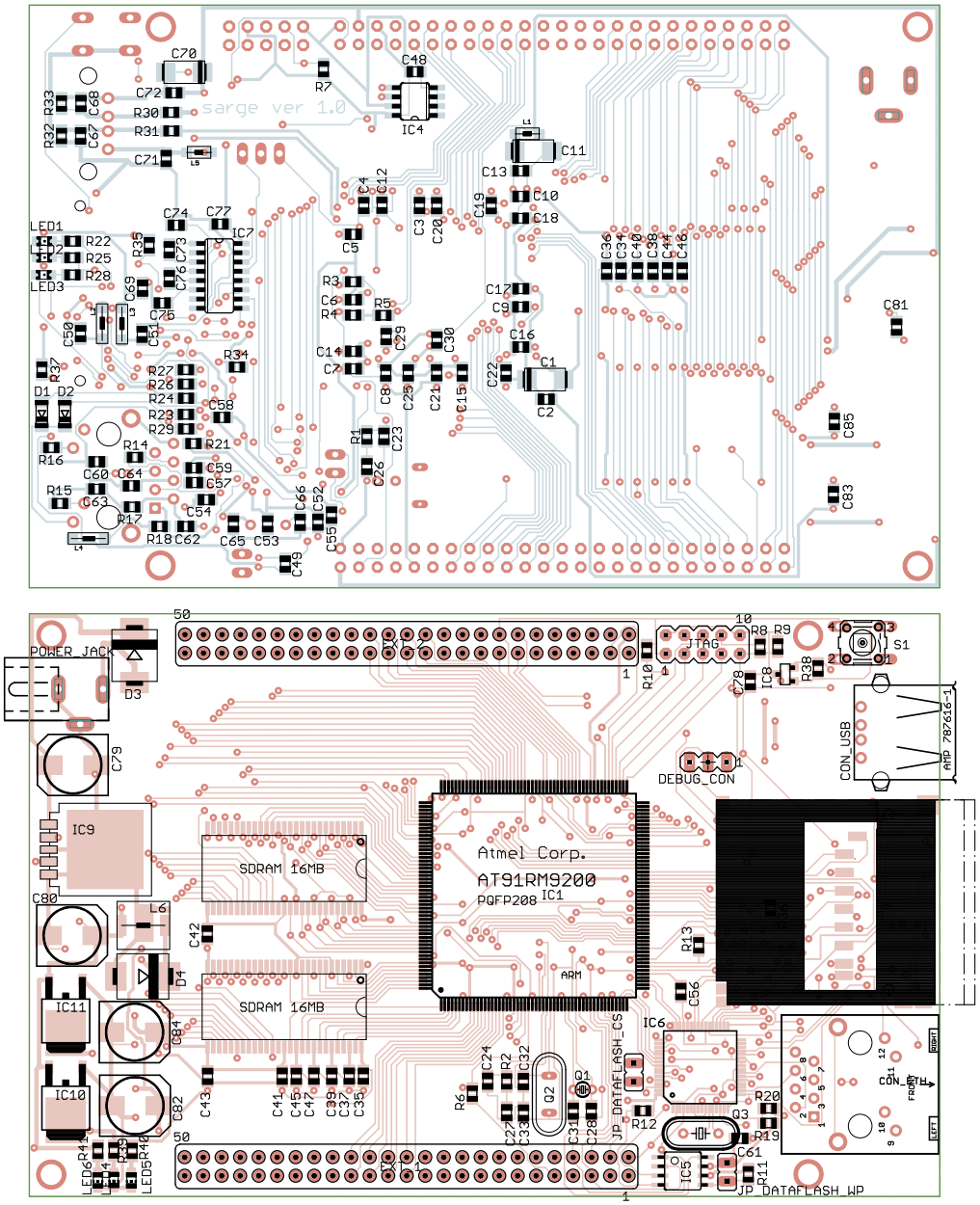
Rys. 10.

mięci, korzystając z opcji 4, można załadować program u-boot do pamięci RAM i go wykonać. Odbywa się to automatycznie, po 5 sekundach po wyzerowaniu komputera.

Działanie programu u-boot zaraz po załadowaniu zależne jest od zmiennych środowiskowych konfigurujących program. Domyślnie jest to załadowanie obrazu systemu Linux z pamięci DataFlash spod wskazanego adresu i jego uruchomienie. Zmienne środowiskowe można przeglądać/modyfikować przy pomocy polecenia printenv oraz setenv. Przykładowa zawartość zmiennych środowiskowych została przedstawiona niżej i zaleca się ich nie modyfikować:

```
bootcmd= bootm 0xC0040000
bootdelay=3
baudrate=115200
ethaddr=00:01:20:38:00:5b
ipaddr=192.168.0.212
serverip=192.168.0.200
rootpath="/tftp/at91/
rootfs"
netmask=255.255.255.0
bootfile="sarge_at91.img"
loadaddr=0x21000000
bootargs=mem=32M
rootfstype=ext2
root=/dev/mmcblk0p1
console=ttyS0,115200n8
rootdelay=5 init=/bin/init
stdin=serial
stdout=serial
stderr=serial
```

Należy zwrócić uwagę na zmienną `ethaddr`, która zawiera adres MAC karty sieciowej. Układ PHY nie ma przypisanego na stałe adresu MAC. Jego przypisanie odbywa się właśnie w bootloaderze drugiego poziomu podczas wstępnej konfiguracji PHY. Zmienne `ipaddr` określają adres IP komputera, a `serverip` adres IP serwera, z którego może zostać pobrany obraz systemu w przypadku bootowania przez sieć z użyciem protokołu BOOTP/TFTP. Aby takie bootowanie było możliwe, należy posiadać skonfigurowany serwer BOOTP/TFTP, który może wysłać na żądanie do komputera plik *sarge_at91.img* zawierający jądro systemu Linux, przygotowane specjalnie na potrzeby minikomputera. Domyślnie jądro systemu Linux umieszczone jest w pamięci DataFlash, i aby je uruchomić, należy wykonać polecenie `boot` w programie u-boot. Odbywa się to automatycznie po uruchomieniu tego programu i kilkusekundowej bezczynności. Jednak wcześniej należy je zapisać w pamięci. W tym celu w bootloaderze drugiego poziomu należy wcześniej skorzystać z opcji nr 3 i używając protokołu XMODEM przesłać plik *sarge_at91-uImage* lub *sarge_at91-zImage*.



Rys. 11. Schemat montażowy komputera

Linux

Jądro systemu umieszczone w pamięci DataFlash nie jest wszystkim, co jest potrzebne aby uruchomić system. Dlatego musimy przygotować system plików, który w tym przypadku znajduje się na karcie SD. Proponowany rozmiar karty to 256 MB dla wszystkich aplikacji kompilowanych domyślnie w pliku konfiguracyjnym, dla środowiska cross-kompilacyjnego OpenEmbedded. Kartę należy przygotować tak, aby miała system plików EXT2 na pierwszej partycji. Najlepiej to zrobić pod kontrolą sytemu Linux, używając programu *fdisk*. System plików należy przygotować system plików: `mkfs.ext2 /dev/mmcblk0p1`.

`/dev/mmcblk0p1` to nazwa urządzenia skojarzona z pierwszą partycją na karcie SD (uwaga, w różnych dystrybucjach Linuxa urządzenie może mieć różne nazwy, np. jako urządzenie SCSI, jeżeli korzystamy z zewnętrznego czytnika; najczęściej będzie to jednak `/dev/sdX1`, gdzie X to a, b, c...). Następnie na kartę należy skopiować system plików z aplikacjami podstawowymi i dodatkowymi przygotowanymi w środowisku OpenEmbedded, zawartymi w pliku *rootfs.tgz*. Tak przygotowaną kartę SD należy umieścić w złączu na płycie minikomputera. Praca z komputerem jest możliwa domyślnie poprzez port szeregowy modułu DEBUG oraz przez usługi takie, jak SSH i FTP w przypadku, gdy zostanie on podłączony do sieci TCP/IP przez Ethernet.

mgr inż. Grzegorz Rajtar
mcgregor@blackmesaeast.com.pl

Literatura

[1] http://www.atmel.com/dyn/resources/prod_documents/doc1768.pdf
[2] http://www.atmel.com/dyn/resources/prod_documents/PLL_LFT_filter_CALCULATOR_AT91.zip
[3] http://dlharmon.com/
[4] http://www.blackmesaeast.com.pl/linux/downloads/