TMS320VC5416 z Nokii 5510

Zestaw ewaluacyjny dla procesora sygnałowego (1)

Procesory DSP odważnie wkraczające we współczesny świat elektroniki, dając dobry kompromis pomiędzy ceną a niebywałymi możliwościami obliczeniowymi, są coraz częściej stosowane przez projektantów. Można je znaleźć w wielu przenośnych urządzeniach codziennego użytku, jak np. w odtwarzaczach MP3 oraz telefonach.

Rekomendacje:
jak każdy zestaw ewaluacyjny, tak i ten na pewno będzie nieodzownym wyposażeniem konstruktorów zajmujących się układami DSP, tym bardziej, że ze względu na cenę nie są one łatwo dostępne dla amatorów.

Jednym ze światowych liderów wśród producentów układów DSP jest Texas Intruments. Z ciekawszych pozycji w bogatej ofercie tej firmy można wyróżnić bardzo wydajne układy serii C6000 lub nieco mniej efektywne, jednak cenione ze względu na energooszczędność, procesory serii C5000. Niestety ceny zestawów ewaluacyjnych dla tych układów są dosyć wysokie, przez co nie każdy elektronik amator może sobie na nie pozwolić. Nie ułatwia to poznania tych jakże interesujących układów. Dobrą alternatywą może być własnoręczne zbudowanie zestawu startowego, który dawałby możliwość eksperymentowania z obróbką dźwięku, czy też grafiką. Jedyny problem może stanowić dostępność podzespołów bazowych do takiego projektu.

AVT–5174

W ofercie AVT:
AVT–5174A – płytka drukowana

PODSTAWOWE PARAMETRY

- Płytka o wymiarach 119×98 mm
- Zasilanie 6...15 V
- Gniazda: PHONES, LINE OUT, LINE IN, MIC, USB, PS2, DSP JTAG, CPLD JTAG, MSP430 JTAG, wielokanałowy buforowany port szeregowy McBSP, interfejs równoległy HPI8 (adres i dane multipleksowane na 8-bitowym porcie)
- Pamięć SRAM: 512 kB×16
- Pamięć NAND Flash: 64 MB×8
- Wyświetlacz LCD: matryca 640x480 16-bit
- Częstotliwość próbkowania sygnałów audio: 8...96 kHz

Tab. 1. Zestawienie elementów wraz z oznaczeniami

Symbol	Oznaczenie	Opis
C66	22 µF	Kondensator tantalowy
C67	22 µF	Kondensator tantalowy
C68	22 µF	Kondensator tantalowy
D1	LED	Zielona dioda
D1	LED	Zielona dioda
U2	LK112M16	Stabilizator 1,6 V
U3	MSP430F135	Mikrokontroler
U4	LP3985 3.1	Stabilizator 3,1 V
U6	K9K1208U0A	NAND Flash 64 MB
U12	TMS320VC5416	DSP
U13	PDIUSBD12	Interfejs USB
U14	SN65220	Zabezpieczenie ESD
U15	TLV320AIC23	Kodek audio

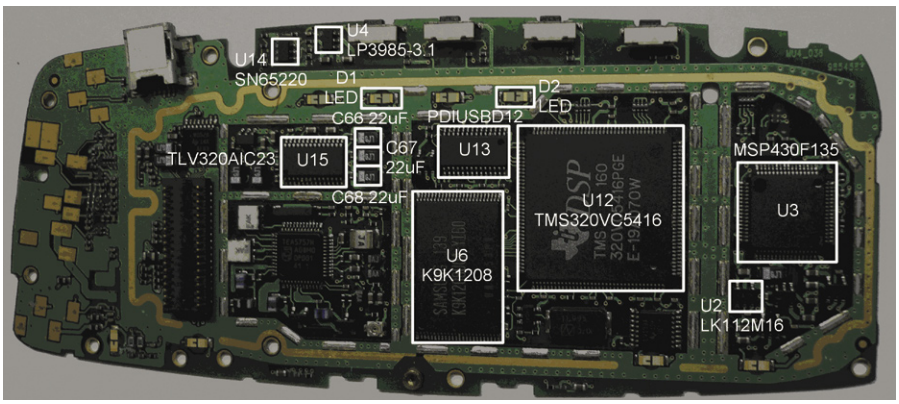
PROJEKTY POKREWNE wymienione artykuły są w całości dostępne na CD

Tytuł artykułu	Nr EP/EdW	Kit
Zestaw startowy dla mikrokontrolerów PsoC	EP 4/2006	AVT-926
Zestaw startowy dla mikrokontrolerów ST7FLITE2x	EP 7-8/2006	AVT-939
Zestaw uruchomieniowy dla procesorów 89CX051 i AVR	EP 3/2000	AVT-854
Emulator procesorów 89CX051	EdW 3/2000	AVT-2501

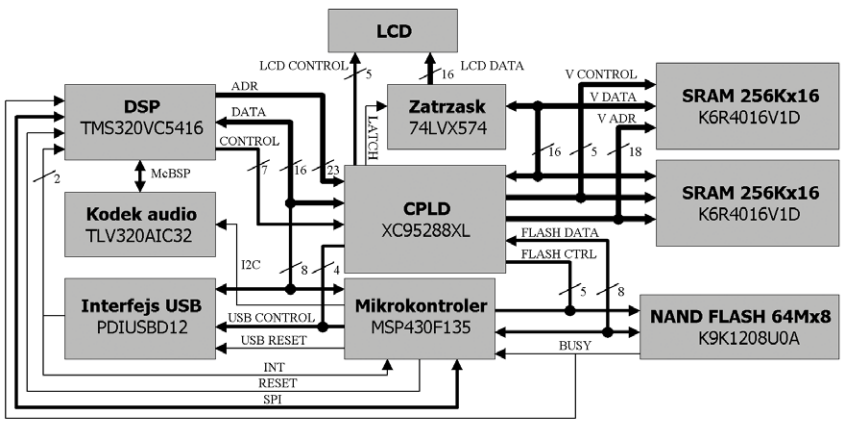
Postanowiłem zatem rozejrzeć się za jakimś niedrogim, a zarazem łatwo dostępnym źródłem elementów. Moją uwagę zwrócił telefon Nokia 5510, który ma już parę dobrych lat, przez co jest tani i dosyć łatwo dostępny. W jego wnętrzu można znaleźć wiele interesujących elementów, takich jak pamięć Flash o pojemności 64 MB, kodek audio, interfejs USB no i co najważniejsze procesor TMS320VC5416 taktowany zegarem 160 MHz. Przydałby się jeszcze jakiś interfejs wizualny do komunikacji z użytkownikiem. Postanowiłem zatem wyposażyć zestaw w prostą kartę graficzną w oparciu o układ logiki programowalnej współpracujący z pamięcią SRAM.

Nokia 5510 – co w środku?

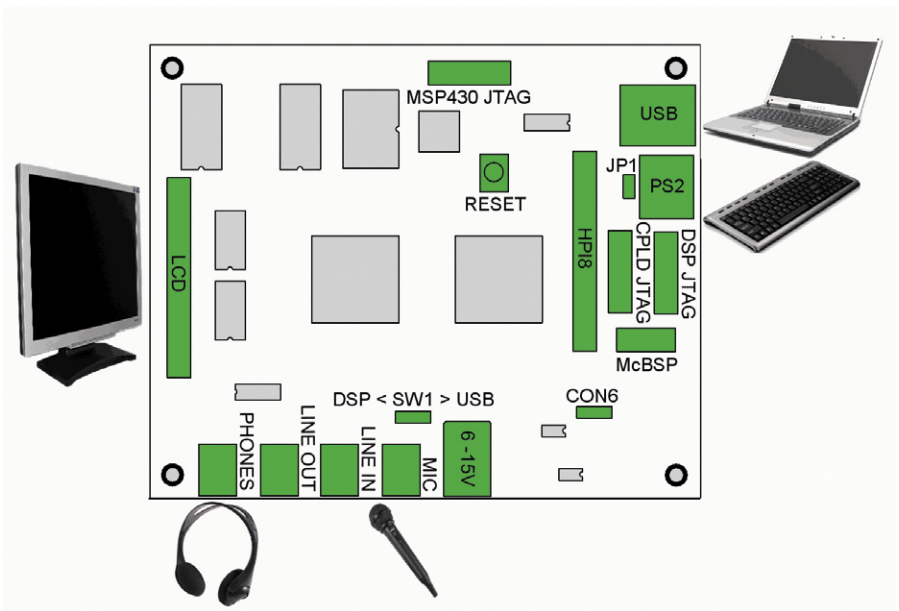
Po zdjęciu obudowy oraz wyświetlacza z klawiaturą, w telefonie znajdujemy dwie płytki. Na jednej z nich elementy są przylutowane w technologii BGA z bardzo drobnym rastrem, co niestety stanowi sporą przeszkodę w ponownym ich wykorzystaniu. Znacznie bardziej przyjazna elektronikowi amatorowi jest druga z płytek. Elementy tam umieszczone są w obudowach SMD, przez co łatwo można je wylutować. Potocznie płytka ta jest przez niektórych nazywana „modułem muzycznym" (**fot. 1**). Odpowiada między innymi za odtwarzanie MP3, radio i magazynowanie plików. To właśnie ona posłuży nam jako „dawca" elementów do na-



Fot. 1. „Modułu muzyczny" z telefonu Nokia 5510



Rys. 2. Uproszczony schemat blokowy zestawu



Rys. 3. Rozmieszczenie wyprowadzeń oraz przycisków na płycie układu

WYKAZ ELEMENTÓW

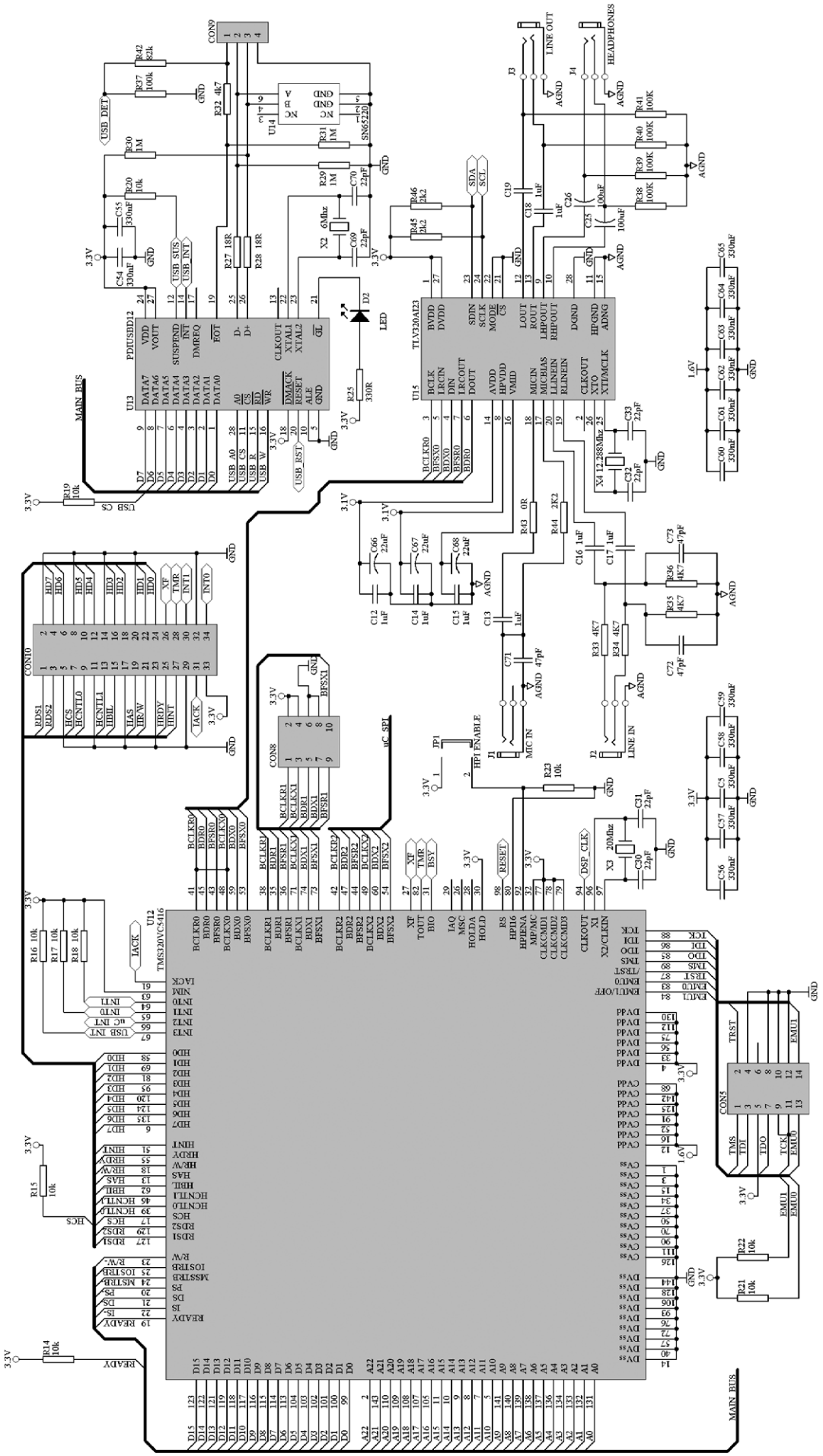
Rezystory
R1, R2: 0,1 Ω
R3...R23, R26: 10 kΩ (0603)
R24, R25: 330 Ω (0603)
R27, R28: 18 Ω (0805)
R29, R30, R31: 1 MΩ (0805)
R32...R36: 4,7 kΩ 1% (0603)
R37...R41: 100 kΩ 1% (0603)
R42: 82 kΩ (0603)
R43: 0 Ω (0603)
R44, R45, R46: 2,2 kΩ (0603)

Kondensatory
C1...C4: 100 nF (0805)
C6...C19: 1 μF/16 V (0805)
C20...C26: 100 μF/16 V (6032) tantal
C27: 100 μF/16 V
C28...C33, C69, C70: 22 pF (0805)
C5,C34...C65: 330 nF (0603)
C66, C67, C68: 22 μF (0805) tantal
C71, C72, C73: 47 pF (0805)

Półprzewodniki
D1, D2: LED (1206)
D3: 1N5819
T1: IRF7416 (SO–8)
U1: MAX1626 (SO–8)
U2: LK112M16 (SOT–25)
U3: MSP430F135 (PQFP–64)
U4: LP3985 3.1 (SOT–25)
U5: 7805 (TO–252)
U6: K9K1208U0A (TSOP–48)
U7, U8: K6R4016V1D (TSOP–44)
U9, U10: 74LVX574 (SOL–20)
U11: XC95288XL (PQFP–144)
U12: TMS320VC5416 PQFP–144
U13: PDIUSBD12 (TSSOP–28)
U14: SN65220 (SOT–26)
U15: TLV320AIC23 (TSSOP–28)
U16: 74LCX125 (TSSOP–14)

Inne
X1: kwarc 8 MHz
X2: kwarc 6 MHz
X3: kwarc 20 MHz
X4: kwarc 12,288 MHz
G1: generator kwarcowy 40 MHz
L1: dławik 100 μH
JP1: goldpin 1x2 + zworka
J1, J2, J3, J4: gniazdo minijack 3,5 mm
CON1: gniazdo NS39–W2K
CON2: gniazdo mini–DIN 6
CON3, CON4, CON5: goldpin 7x2
CON6: Molex 22–27–2031
CON8: goldpin 5x2
CON9: złącze USB–B
CON7, CON10: goldpin 17x2

Na CD karty katalogowe i noty aplikacyjne elementów oznaczonych na Wykazie Elementów kolorem czerwonym



Rys. 4. Schemat elektryczny części z DSP oraz jego peryferii

Tab. 2. Struktura pakietu Command Block Wrapper

bajt/bit	7	6	5	4	3	2	1	0
0...3	dCBWSignature							
4...7	dCBWTag							
8...11	dCBWDataTransferLength							
12	bmCBWFlags							
13	Reserved (0)				bCBWLUN			
14	Reserved (0)		bCBWCBLength					
15...30	CBWCB							

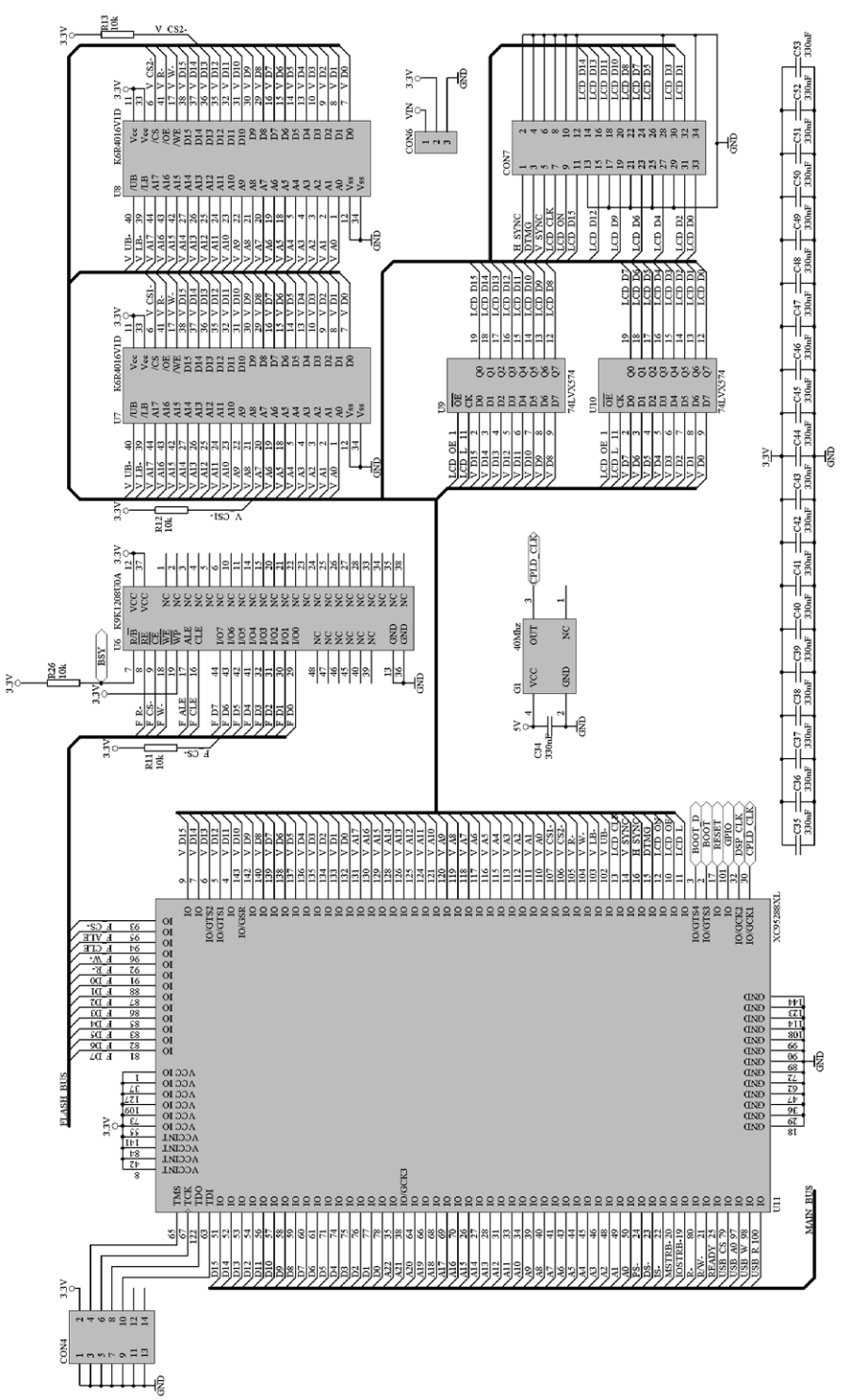
szego zestawu. Wykorzystywane elementy są zaznaczone obwódką oraz symbolem odpowiadającym oznaczeniu na schemacie elektrycznym zestawu. W **tab. 1** przedstawiono zestawienie elementów, oznaczenie i krótki opis.

Opis układu

Na **rys. 2** przedstawiono uproszczony schemat blokowy zestawu, ilustrujący połączenia pomiędzy elementami. Jak widać, sporą jego część zajmuje układ logiki programowalnej. Pełni on rolę niejako mostka pomiędzy głównym procesorem a częścią peryferiów (SRAM, Flash). Podlegają mu 2 banki pamięci po 512 kB, których zawartość jest wyświetlana na matrycy LCD. W celu zredukowania używanych linii I/O układu CPLD zastosowany został zatrzask przekazujący dane do matrycy bezpośrednio z wyjścia danych pamięci. Dodatkowo logika programowalna angażowana jest podczas dostępu do Flash'a przez DSP. Bezpośredni dostęp do pamięci NAND ma również mikrokontroler odpowiadający za umożliwienie zapisu/odczytu danych poprzez interfejs USB, bez potrzeby angażowania głównego procesora. Na barkach MSP430F135 spoczywa także zerowanie układów, obsługa klawiatury oraz konfiguracja wewnętrznych rejestrów kodeka audio poprzez interfejs I²C. Komunikacja pomiędzy mikrokontrolerem a DSP odbywa się przez interfejs SPI. MSP430F135 pracuje w trybie slave, więc gdy chce on przesłać informację o naciśnięciu klawisza przez użytkownika, musi to zasygnalizować masterowi generując przerwanie. Przesyłanie danych pomiędzy kodekiem audio a procesorem jest realizowane poprzez dwukierunkowy port szeregowy McBSP. Na **rys. 3** przedstawiono wyprowadzenia oraz istotniejsze elementy na płytce układu. Schemat elektryczny składa się z trzech części. Schemat ideowy układu DSP wraz z peryferiami przedstawiono na **rys. 4**, na **rys. 5** cześć odpowiedzialną za wyświetlanie informacji na LCD, czyli CPLD i pamięci SRAM, na **rys. 6** zaś mikrokontroler wraz z elementami zasilania.

Interfejs USB

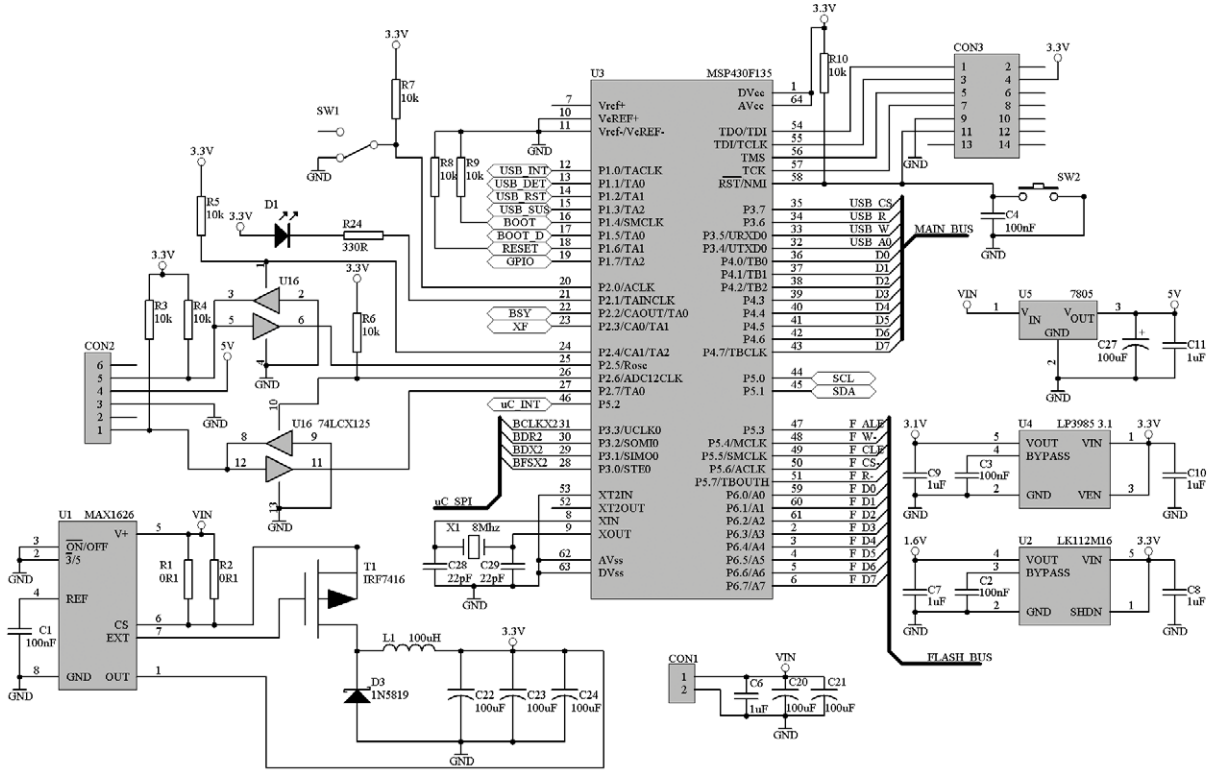
Zapis do pamięci NAND Flash na zewnątrz układu jest możliwy dzięki interfejsowi USB. Pozwala on na zapisanie w niej danych, które mogą być później wykorzystywane przez system. Ponadto w pamięci tej jest także zapisany



Rys. 5. Schemat elektryczny części odpowiadającej za obsługę LCD

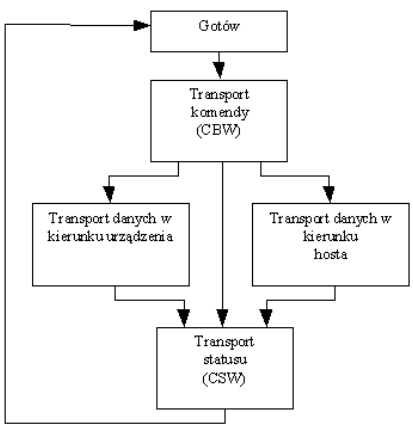
program procesora DSP. W celu uruchomienia układu w trybie USB przesuwamy przełącznik SW1 w kierunku przetwornicy. DSP jest w stanie zerowania, jego wyjścia danych są w stanie wysokiej impedancji, kontrolę nad główna szyną przejmuje MSP430F135, mając tym samym wyłączny dostęp do PDIUSBD12. Mikrokontroler jest odpowiedzialny za obsługę komend SCSI poprzez magistralę USB. Teraz zestaw, po podłączeniu do komputera kablem USB, jest widziany przez system operacyjny jako dysk wymienny (**rys. 7**).

Przyjrzyjmy się teraz nieco bliżej działaniu magistrali USB. Komputer komunikuje się z urządzeniami na końcach szyny USB poprzez wymianę pakietów pomiędzy tak zwanymi punktami końcowymi (EP – Endpoint). W skład takiego pakietu wchodzi między innymi adres urządzenia docelowego, numer punktu końcowego oraz kierunek (In, Out). Punkt końcowy można potraktować jako część urządzenia uczestniczącą w dialogu z komputerem. Maksymalnie na jedno urządzenie może przypadać 16 punktów końcowych, numerowanych od 0 do 15. Wymagane jest, aby każde urządzenie obsługiwało EP0 (Endpoint 0) odpowiadający za kontrolę i status. Poprzez niego, po podłączeniu urządzenia do magistrali USB, przydzielany jest odpowiedni adres



Rys. 6. Mikrokontroler pomocniczy wraz z elementami zasilania

Rys. 7. Zestaw widziany przez Windows XP po podłączeniu przez port USB



Rys. 8. Schemat przepływu komend, danych i statusu

podczas wczesnej fazy identyfikacji, następnie przesyłane są struktury dotyczące ogólnych informacji (DeviceDescriptor) oraz konfiguracji (ConfigurationDescriptor). W przypadku naszego układu istotna jest struktura InterfaceDescriptor wchodzącą w skład ConfigurationDescriptor, opisująca sposób komunikacji. Pole bInterfaceClass=08h oznacza klasę urządzenia magazynującego, bInterfaceSubClass=06h implikuje zgodność z SCSI Primary Commands–2, natomiast bInterfaceProtocol=50h masowy tryb transmisji danych (bulk–only transport). Według tej konfiguracji nasz zestaw poza standardowa obsługą EP0 musi implementować także EP1 w trybie masowym, Jest to tryb asynchroniczny, pozwalający na zmienne odstępy czasowe pomiędzy przesyłaniem kolejnych pakietów. Został on opracowany z myślą o transferze dużych porcji danych. Wykorzystuje się go w urządzeniach takich jak: pamięci masowe, skanery, drukarki. Szczegółową jego specyfikację oraz specyfikację samej magistrali USB można znaleźć w dokumentach „Universal Serial Bus Mass Storage Class Bulk–Only Transport" i „USB in a Nutshell".

Tab. 3. Blok o rozmiarze 6 bajtów opisujący komendę SCIS

bajt/bit	7	6	5	4	3	2	1	0
0	Kod operacji							
1	Zarezerwowane			(MSB)				
2	Logiczny adres sektora							
3								(LSB)
4	Liczba sektorów							
5	Kontrola							

Tab. 4. Blok o rozmiarze 10 bajtów opisujący komendę SCIS

bajt/bit	7	6	5	4	3	2	1	0
0	Kod operacji							
1	Zarezerwowane			Flagi				
2	(MSB)							
3	Logiczny adres sektora							
4								
5								(LSB)
6	Zarezerwowane							
7	(MSB)	Liczba sektorów						
8								(LSB)
9	Kontrola							

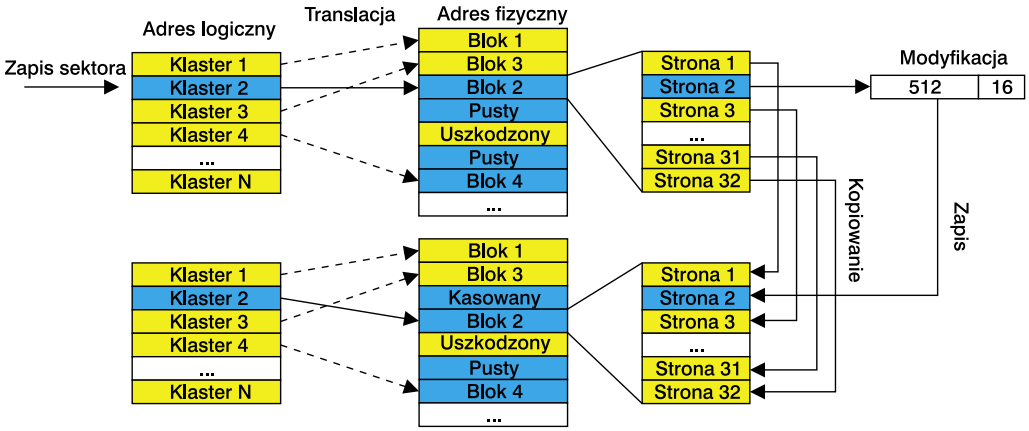
Na **rys. 8** przedstawiono schemat przepływu komend, danych i statusu dla transferów masowych. Przesyłanie właściwych danych w protokole jest poprzedzone pakietem zawierającym komendę, a kończone pakietem ze statusem operacji.

Schemat transferu omówię na przykładzie komendy zapisu. Początkowo urządzenie znajduje się w stanie gotowości.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
LSN0	LSN1	LSN2	RSV	RSV	BI	ECC0	ECC1	ECC2	SECC0	SECC1	RSV	RSV	RSV	RSV	RSV

LSNx Adres logiczny strony
ECCx Korekcja i kontrola błędów dla całej strony
SECCx Korekcja i kontrola błędów dla LSN
BI Inna wartość niż FFh oznacza uszkodzony blok
RSV Zarezerwowane/nieużywane

Rys. 9. Proponowany przez producentów sposób zapisu informacji o stronie



Rys. 10. Proces zapisu sektora pamięci Flash (NAND)

Komputer macierzysty chcąc dokonać zapisu danych w sektorze przesyła pakiet o długości 31 bajtów, który opakowuje właściwą komendę SCSI (CBW – Command Block Wrapper). Jego strukturę opisuje **tab. 2**.

Pole dCBWSignature ma zawsze wartość 43425355h („USBC"). Ułatwia to identyfikację pakietu. dCBWTag zawiera liczbę wygenerowaną przez hosta. Do poprawnego zakończenia operacji należy ją zwrócić w polu o tej samej nazwie w pakiecie statusu. dCBWDataTransferLength niesie informację o tym, jakiej ilości danych oczekuje host, 7 bit w polu bmCBWFlags mówiący o kierunku ich transferu (0 – od komputera do urządzenia, 1 – w przeciwnym kierunku). Właściwa operacja SCIS (zapisu) mieści się w tablicy CBWCB, gdzie bCBWCBLength określa jej rozmiar w bajtach. Uogólniony wygląd bloku opisującego komendę SCIS 6– i 10–bajtową przedstawiają **tab. 3** i **4**.

Jak widać, pierwszy bajt bloku (CBWCB[0]) zawsze identyfikuje operację. Komendy obsługiwane przez zestaw z krótkim opisem przedstawiono w **tab. 5**.

Wracając do naszego przykładowego zapisu sektora, po wartości pierwszego bajtu w polu CBWCB równej 2Ah identyfikujemy komendę WRITE. Wartość typu DWORD z offsetem 2, reprezentuje logiczny adres sektora początkowego, wartość typu WORD z offsetem 7, to liczba sektorów do zapisu. Teraz trzeba wykonać translację adresu logicznego na fizyczny w pamięci NAND Flash i zapisu sektorów otrzymywanych w kolejnych pakietach danych. W tym celu należy się przyjrzeć budowie pamięci K9K1208U0A. Składa się ona z szeregu stron o rozmiarze 512 bajtów plus 16 bajtów na dodatkowe informacje. Poza tym, pamięć jest zorganizowana w bloki składające się z zestawów 32 kolejnych stron. Operacje odczytu i programowania są wykonywane na stronach, kasowania natomiast na całych blokach. Zatem, gdy chcemy zastąpić dane na jednej stronie innymi danymi, musimy dokonać kasowania wszystkich 32 stron w bloku. Zachodzi potrzeba skopiowania niemodyfikowanych informacji do innego pustego bloku, przez co zmienia się ich fizyczny adres. Pamięci NAND w celu zwiększenia wydajności oferują możliwość bezpośredniego kopiowania stron wewnątrz układu. Pomijane więc są cykle operacji odczytu i zapisu wchodzące w skład przenoszenia danych przez procesor sterujący. Jednak powstaje problem identyfikacji sektora, do którego chcemy się odwołać mając jego adres logiczny. Informacja ta jest umieszczana w dodatkowych 16 bajtach na stronie. Na **rys. 9** przedstawiono proponowany przez producentów sposób zapisu tych danych. Pozostaje jeszcze jedno istotne zjawisko – zużywanie bloków. Pamięci Flash mają ograniczoną liczbę cykli kasowania. W przypadku błędu podczas czyszczenia bloku, należy go zaznaczyć jako uszkodzony, by uniknąć w przyszłości utraty informacji. Na **rys. 10** przedstawiono cały omówiony proces programowania sektora. Dla poprawy czytelności przyjęto, że w jednym bloku mieści się jeden klaster.

Uwaga! Może się zdarzyć, że w krytycznym momencie zapisu, zostanie przerwana praca układu (poprzez zmianę trybu pracy lub odcięcie zasilania). W wyniku tego informacje o adresach logicznych zostaną utracone. W takim przypadku należy sformatować pamięć Flash. Dokonujemy tego uruchamiając zestaw w trybie USB i wciskając jednocześnie klawisze Ctrl+Alt+Esc (układ ma być odłączony od komputera).

Po wykonaniu zapisu musimy powiadomić hosta o statusie wykonania operacji. Robimy to wysyłając 13–bajtowy pakiet (CSW – Command Status Wrapper). Jego wygląd przedstawiono w **tab. 6**. Pola dCBWSignature i dCBWTag mają to samo znaczenie i wartość, co w pakiecie z komendą. dCSWDataResidue to różnica między dCBWDataTransferLength, a liczbą przesłanych danych. bCSWStatus zawiera status wykonania komendy, wartość 00h oznacza pomyślne zakończenie operacji, 01h to błąd lub nie rozpoznanie komendy.

Zagadnienie pamięci masowych to bardzo obszerny temat, a powyższy opis stanowi tylko podstawowy zarys zasad ich działania. Zainteresowanych zachęcam do lektury dokumentu „USB Mass Storage Device Using a PIC MCU".

Marcin Ostopinko
C5416@wp.pl

Tab. 5. Obsługiwane komendy SCSI

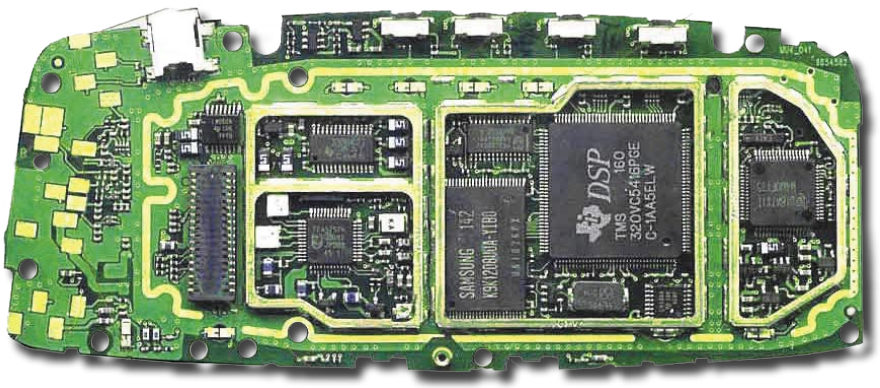
Nazwa	Kod	Wejście	Wyjście	Opis
INQUIRY	12h	–	Struktura 36-bajtowa zawierająca informacje o urządzeniu	Odczyt podstawowych informacji o urządzeniu
READ CAPACITY	25h	–	Rozmiar oraz adres LBA ostatniego sektora	Odczyt informacji o pojemności nośnika danych
READ (10)	28h	Adres LBA pierwszego sektora oraz ich liczba do odczytu	Odczytane dane	Odczyt sektora (sektorów)
WRITE (10)	2Ah	Adres LBA pierwszego sektora oraz ich liczba do zapisu. Dane do zapisu	–	Zapis sektora (sektorów)
MODE SENSE (6)	1Ah	–	Zwraca 4 bajty odpowiedzi	Odczyt trybu pracy urządzenia
REQUEST SENSE (6)	03h	–	Struktura 18–bajtowa zawierająca informacje o błędzie urządzenia.	Odczyt informacji o stanie urządzenia
PREVENT ALLOW MEDIUM REMOVAL	1Eh	–	Zawsze zwraca poprawne wykonanie (nie implementowane)	Zabronienie/zezwolenie usunięcia nośnika danych
TEST UNIT READY	00h	–	Zawsze zwraca poprawne wykonanie	Zbadanie stanu gotowości urządzenia
VERIFY (10)	2Fh	–	Zawsze zwraca poprawne wykonanie (nie implementowane)	Weryfikacja sektora (sektorów)
START/STOP	1Bh	–	Zawsze zwraca poprawne wykonanie (nie implementowane)	Zarządzanie stanem nośnika danych

Tab. 6. Command Status Wrapper

bajt/bit	7	6	5	4	3	2	1	0
0...3	dCBWSignature							
4...7	dCBWTag							
8...11	dCSWDataResidue							
12	bCSWStatus							

TMS320VC5416 z Nokii 5510
Zestaw ewaluacyjny dla procesora sygnałowego (2)

Przedstawiamy drugą część opisu wykonania zestawu ewaluacyjnego przeznaczonego dla procesorów DSP. Podano w nim informacje uzupełniające na temat montażu i uruchomienia płytki, a także bootloadera, układów pamięci i procesora DSP.



Bootloader

Procesor TMS320VC5416 posiada wewnętrzną pamięć ROM, w której umieszczony jest *bootloader*. W telefonie Nokia 5510 ładuje on kod bezpośrednio z pamięci NAND Flash do RAM–u. Aby go uruchomić, należy podczas zerowania układu podać stan niski na wejście MP/!MC. Wówczas wewnętrzny ROM jest mapowany w przestrzeni adresowej pamięci programu. Niestety, nie udało mi się znaleźć dokładniejszych informacji dotyczących działania tego *bootloadera*, dlatego nie pozostaje nic innego, jak opracować własny program kopiujący dane do wewnętrznej pamięci. Bezpośrednie wykonywanie rozkazów z pamięci NAND ze względu na jej budowę nie jest możliwe. Trzeba skorzystać z innego sposobu przekazywania instrukcji, mianowicie podczas zerowania DSP na linii BOOT mikrokontroler ustawia stan wysoki, sygnalizując tym samym układowi logiki programowalnej, że od tego momentu wszelkie odczyty DSP z zewnętrznej pamięci dotyczyć będą procesu bootowania. Po wyzerowaniu TMS320VC5416 do licznika rozkazów PC ładowany jest adres 0xFF80. Jeśli na nóżce MP/!MC jest stan wysoki, to zamiast ROM–u mapowana jest w tym miejscu pamięć zewnętrzna. W momencie, gdy procesor odwoła się do niej w celu odczytania pierwszego rozkazu (sygnały sterujące !PS=!MSTRB=0 R/!W=1), CPLD ustawia na linii READY stan niski, wymuszając w ten sposób dodatkowe stany opóźnienia odczytu (**rys. 11**). Teraz inicjatywę przejmuje mikrokontroler MSP430F135. Sprawdza on istnienie pliku *boot.dat* w systemie FAT32 wewnątrz Flasha, a następnie zaczyna „podrzucać" procesorowi podczas kolejnych odczytów z pamięci, rozkazy zapisujące kolejne bajty pliku do pamięci RAM. Pamięć Flash posiada 8–bitowe wyjście danych, dlatego wymagane jest wykorzystanie zatrzasków wewnątrz układu logiki programowalnej. Wartości są zatrzaskiwane parami, na opadających zboczach linii BOOT_D, a następnie kierowane na 16–bitowa szynę danych.

Sam format pliku *boot.dat* jest taki sam, jak w przypadku fabrycznego *bootloadera* Texas Instruments dla standardowego 8–bitowego trybu szeregowego (**tab. 7**) opisanego w dokumencie *„TMS320VC5402A/VC5409A/VC5410A/VC5416 Bootloader"*. Składa się on z bloków danych wraz z docelowym adresem i rozmiarem. W przypadku braku lub nieprawidłowości w strukturze *boot.dat*, błąd jest sygnalizowany migającą diodą D1.

Po transporcie wszystkich części danych mikrokontroler chcąc odczytać rozmiar kolejnego bloku natrafia na wartość 0000h, oznaczającą

AVT–5174

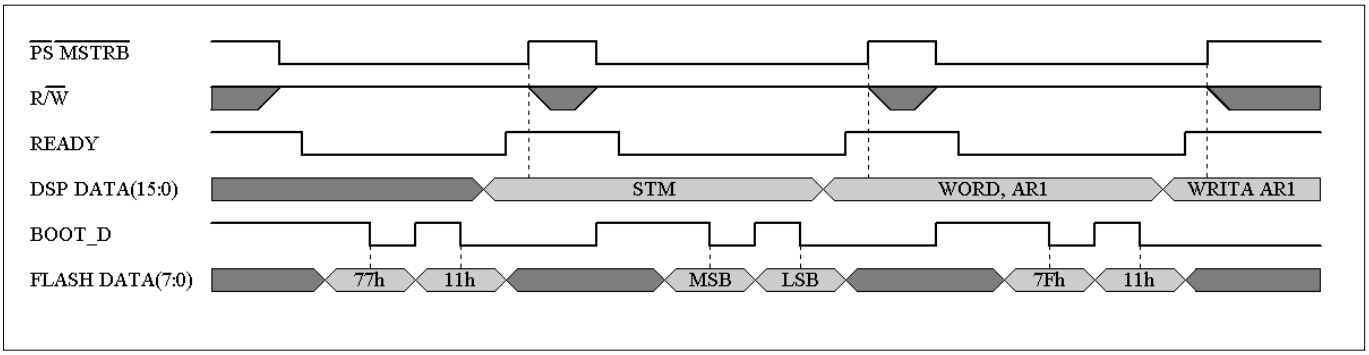
W ofercie AVT:
AVT–5174A – płytka drukowana

PODSTAWOWE PARAMETRY

- Płytka o wymiarach 119×98 mm
- Zasilanie 6...15 V
- Gniazda: PHONES, LINE OUT, LINE IN, MIC, USB, PS2, DSP JTAG, CPLD JTAG, MSP430 JTAG, wielokanałowy buforowany port szeregowy McBSP, interfejs równoległy HPI8 (adres i dane multipleksowane na 8-bitowym porcie)
- Pamięć SRAM: 512 kB×16
- Pamięć NAND Flash: 64 MB×8
- Wyświetlacz LCD: matryca 640x480 16-bit
- Częstotliwość próbkowania sygnałów audio: 8...96 kHz

PROJEKTY POKREWNE wymienione artykuły są w całości dostępne na CD

Tytuł artykułu	Nr EP/EdW	Kit
Zestaw startowy dla mikrokontrolerów PsoC	EP 4/2006	AVT-926
Zestaw startowy dla mikrokontrolerów ST7FLITE2x	EP 7-8/2006	AVT-939
Zestaw uruchomieniowy dla procesorów 89CX051 i AVR	EP 3/2000	AVT-854
Emulator procesorów 89CX051	EdW 3/2000	AVT-2501



Rys. 11. Zależności pomiędzy sygnałami podczas przekazywania rozkazów *bootloadera* do DSP

koniec pliku. Proces bootowania kończony jest ustawieniem stanu niskiego na linii BOOT. Od tego momentu wszystkie odczyty i zapisy do pamięci są swobodnie wykonywane przez DSP.

PDIUSBD12 i K9K1208U0A

Jak pokazano na schemacie blokowym (rys. 2), pomiędzy pamięcią Flash a DSP znajduje się układ logiki programowalnej. Pracuje on w roli dekodera adresu oraz pośredniczy w wymianie danych. Pamięć K9K1208U0A ma jedną 8–bitową szynę, na której są multipleksowane dane, adres oraz komendy. Poprzez wejścia ALE i CLE możemy dokonać wyboru, do której z wymienionych lokalizacji chcemy mieć dostęp. Ze względu na potrzebę zapewnienia odpowiednich sekwencji sygnałów sterujących przy operacjach na pamięci Flash, DSP manipuluje nimi niezależnie poprzez zapisy do odpowiednich portów I/O. W **tab. 8** przedstawiono ich rozmieszczenie w przestrzeni adresowej I/O procesora i ich opis. W przypadku PDIUSBD12 nie jest wymagany tak skomplikowany interfejs dostępu. Posiada on dwa porty. Poprzez pierwszy z nich dokonywany jest zapis lub odczyt danych do/z wewnętrznego bufora. Drugi port służy do konfiguracji układu i wysyłania komend.

Tab. 7. Format pliku boot.dat

Bajt	Wartość	Opis	
0	08h	Sygnatura	
1	AAh		
2	X	Ignorowane	
3	X		
4	X		
5	X		
6	X		
7	X		
8	X		
9	X		
10	MSB	Numer strony początku programu, młodsze 7 bitów ładowane do XPC	
11	LSB		
12	MSB	Adres początku programu, ładowany do PC	
13	LSB		
14	MSB	Rozmiar (X) bloku nr 0 wyrażony w dwubajtowych słowach	Blok nr 0
15	LSB		
16	MSB	Numer docelowej strony bloku (młodsze 7 bitów)	
17	LSB		
18	MSB	Adres docelowy bloku	
19	LSB		
20	MSB	WORD[1]	
21	LSB		
20+2X-2	MSB	WORD[X]	
20+2X-1	LSB		
20+2X	MSB	Rozmiar (Y) bloku nr 1 wyrażony w dwubajtowych słowach	Blok nr 1
21+2X	LSB		
22+2X	MSB	Numer docelowej strony bloku (młodsze 7 bitów)	
23+2X	LSB		
24+2X	MSB	Adres docelowy bloku	
25+2X	LSB		
26+2X	MSB	WORD[1]	
27+2X	LSB		
26+2(X+Y)-2	MSB	WORD[Y]	
26+2(X+Y)-1	LSB		
2N-2	00h	Oznaczenie końca sekcji bloków	
2N-1	00h		

Interfejs LCD

Podobne konstrukcje oparte na logice programowalnej były już opisane na łamach czasopisma Elektronika Praktyczna, dlatego nie będę się zagłębiać w szczegółową zasadę działania wyświetlaczy i kontrolerów graficznych. W prezentowanym układzie wykorzystywana jest matryca 640x480 16–bit. Układ taktowany jest zegarem 40 MHz pochodzącym z generatora G1, następnie częstotliwość ta jest dzielona przez 2 i podawana do wyświetlacza (złącze CON7). Na płytce znajdują się dwie kości SRAM o organizacji 16–bitowej, których łączna pojemność wynosi 1 MB. Jeden piksel na ekranie odpowiada jednej lokalizacji w pamięci. Kolejno wyświetlane komórki pamięci tworzą obraz. Niestety w przestrzeni adresowej DSP nie ma wystarczająco dużo miejsca, aby zmieścić cały bufor wideo w sposób liniowy. Pamięć programu TMS320VC5416 jest podzielona na 128 stron po 64 k słów każda. Począwszy od strony o numerze 4, obszar o adresie XX8000h–XXFFFFh odwołuje się do pamięci zewnętrznej. Poprzez prostą translację adresów w układzie logiki programowalnej, SRAM został podzielony na sekcje po 32 k i umieszczony na każdej stronie. Widać to na **rys. 12**. Funkcje

`put_pixel()`, `area_fill()`, `print_str()` oraz `area_copy()` w module `graphics.h`, pokazują w jaki sposób można się odwołać do pamięci wideo.

Audio

Kodek TLV320AIC23 umożliwia nagrywanie (mikrofon – J1, line in – J2) oraz odtwarzanie (line out – J3, słuchawki – J4) dźwięku w szerokim zakresie częstotliwości próbkowania od 8 do 96 kHz. Zapisem do jego rejestrów konfiguracyjnych, poprzez magistralę I²C, zajmuje się układ MSP430F135. Służy do tego funkcja AIC23write(). Gdy procesor DSP chce zmienić jakąś wartość w rejestrze konfiguracji kodeka, musi powiadomić o tym mikrokontroler poprzez interfejs SPI. Przesyłanie strumienia danych audio między samym kodekiem a procesorem realizowane jest poprzez dwukierunkowy port szeregowy McBSP0. Kodek pracuje w trybie master, co oznacza, że to on generuje sygnały zegarowe oraz ramki synchronizacyjne. DSP ma do dyspozycji 6 niezależnych kanałów DMA. Daje to możliwość transferu danych z pamięci TMS320VC5416 bez potrzeby angażowania CPU.

Klawiatura

Klawiatura podłączana jest do portu PS2. Układ 74LCX125 pełni rolę izolacji pomiędzy mikrokontrolerem zasilanym napięciem 3,3 V a poziomami o wartości 5 V pochodzącymi z klawiatury. W momencie, kiedy MSP430F135 odczyta kod naciśniętego klawisza za pomocą funkcji `KeyboardRead()`, zapisuje go w buforze transmisji SPI. Następnie sygnalizuje procesorowi DSP przerwaniem INT2 o gotowość do wysłania informacji o naciśniętym klawiszu.

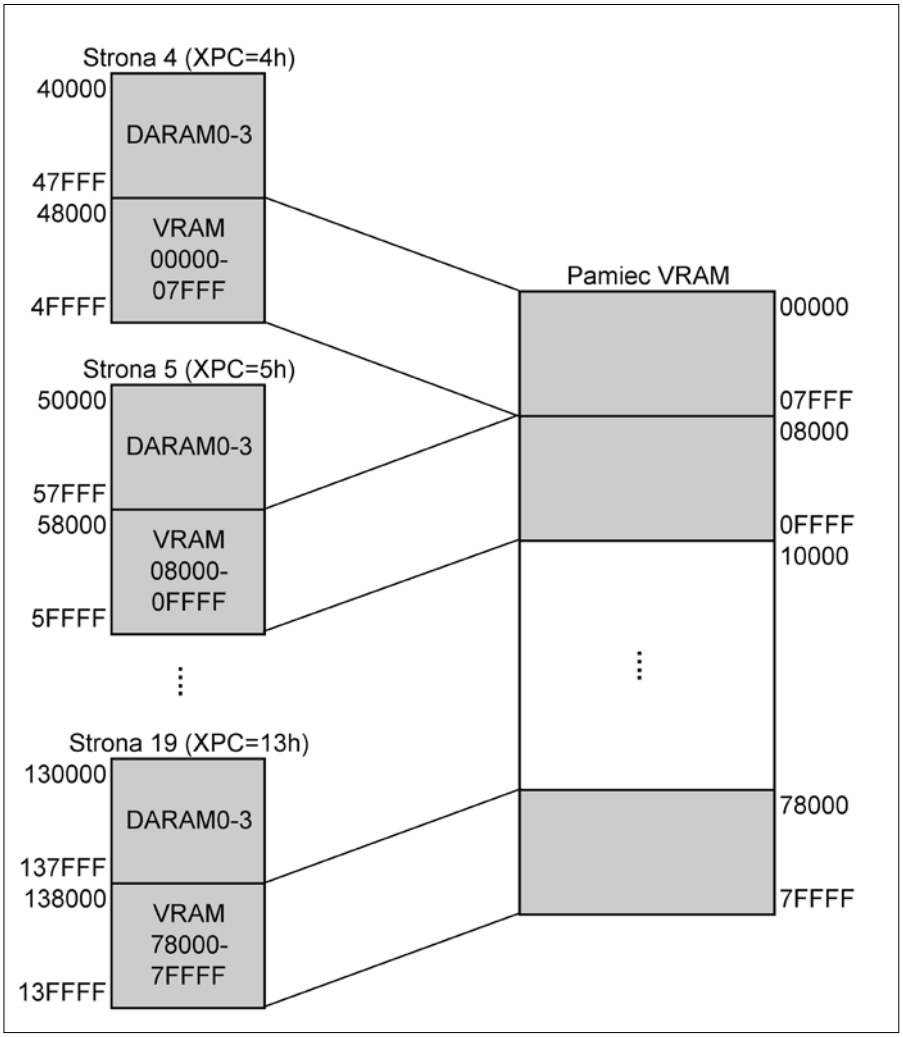
Wyprowadzenia

Mając na uwadze możliwość dalszej rozbudowy systemu, na płytce umieszczono dwa złącza (rys. 3). Pierwsze to wielokanałowy buforowany port szeregowy McBSP1. Drugie złącze wyprowadza równoległy interfejs HPI8, w którym adres i dane są multipleksowane na jednym 8–bitowym porcie. Możliwy jest przez niego dostęp do wewnętrznej pamięci TMS320VC5416. Obydwa interfejsy można skonfigurować jako GPIO. Dodatkowo zworka JP1 pozwala na aktywowanie HPI8 sprzętowo, tuż po zerowaniu procesora.

Poza samymi złączami do wymiany danych, na płytce znajdują się także porty JTAG dla XC95288XL, MSP430F135 i TMS320VC5416 pozwalające na programowanie w systemie.

Zasilanie

Głównym żywicielem zestawu jest przetwornica MAX1626. Jej maksymalna wydajność prądowa wynosi ponad 2 A. Układy w zestawie nie mają wprawdzie takiego apetytu, jednak trzeba uwzględnić możliwość podłączenia ekranu LCD. Zasilanie wyprowadzono na złączu CON6, gdzie oprócz napięcia 3,3 V bezpośrednio dostępne jest napięcie wejściowe układu. Procesor



Rys. 12. Organizacja pamięci wideo w przestrzeni adresowej DSP

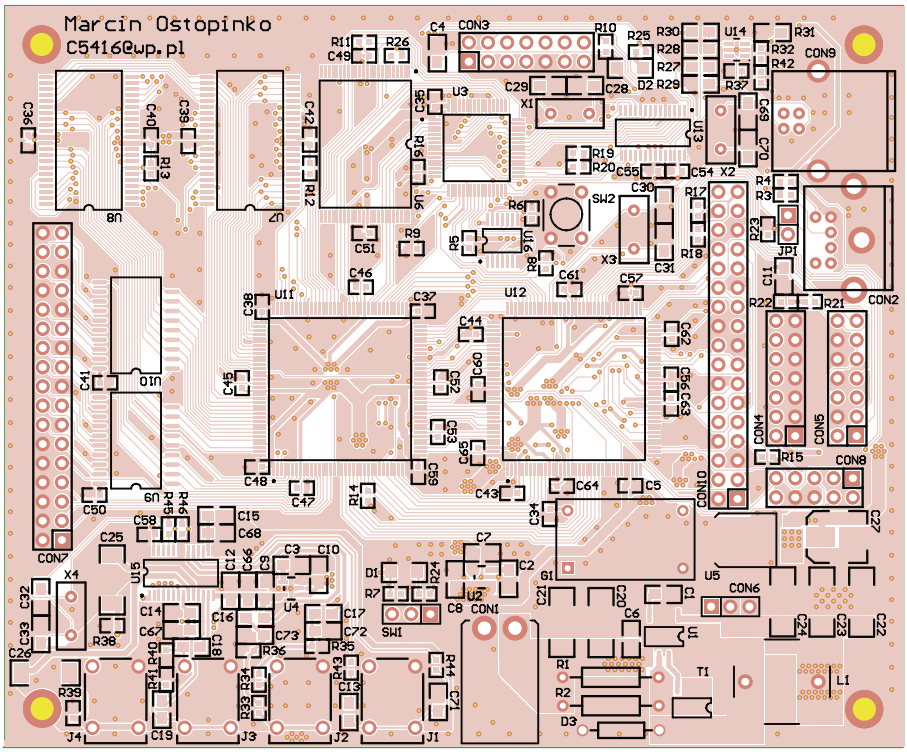
TMS320VC5416 wymaga dodatkowo napięcia o wartości 1,6 V do zasilania rdzenia. Służy do tego stabilizator LK112M16. W celu uzyskania odpowiedniej jakości dźwięku część analogową kodeka TLV320AIC23 zasila stabilizator LP3985.

Montaż i uruchomienie

Schemat montażowy płytki drukowanej pokazano na **rys. 13**. Zmontowanie układu wymaga sporych umiejętności w lutowaniu SMD. Układy TMS320VC5416 i XC95288XL mają dosyć drobny raster, a dodatkowo blisko nich umieszczone są kondensatory w obudowach 0603. W pierwszej kolejności lutujemy przetwornicę MAX1626 wraz z elementami potrzebnymi do jej pracy (R1, R2,C1, C6, C20...24, L1, D3, T1) oraz mikrokontroler MSP430F135. Poprzez JTAG–a na złączu CON3 zastępujemy stary program z Nokii nowym, właściwym dla naszego systemu. Można tego dokonać za

Tab. 8. Mapa portów w przestrzeni adresowej I/O

Adres	Dostęp	Opis
0000h	RW	Interfejs USB – zapis/odczyt danych
0001h	RW	Interfejs USB – zapis/odczyt komend
0003h	RW	Flash – zapis(W=0)/odczyt(R=0) danych(CS=ALE=CLE=0), adresu (ALE=1 CS=CLE=0), komendy (CS=ALE=0 CLE=1)
0004h	W	Flash – zapis do tego portu ustawia CS=0
0005h	W	Flash – zapis do tego portu ustawia CS=1
0006h	W	Flash – zapis do tego portu ustawia ALE=1
0007h	W	Flash – zapis do tego portu ustawia ALE=0
0008h	W	Flash – zapis do tego portu ustawia CLE=1
0009h	W	Flash – zapis do tego portu ustawia CLE=0
000Ah	W	Flash – zapis do tego portu ustawia R=0
000Bh	W	Flash – zapis do tego portu ustawia R=1
000Ch	W	Flash – zapis do tego portu ustawia W=0
000Dh	W	Flash – zapis do tego portu ustawia W=1

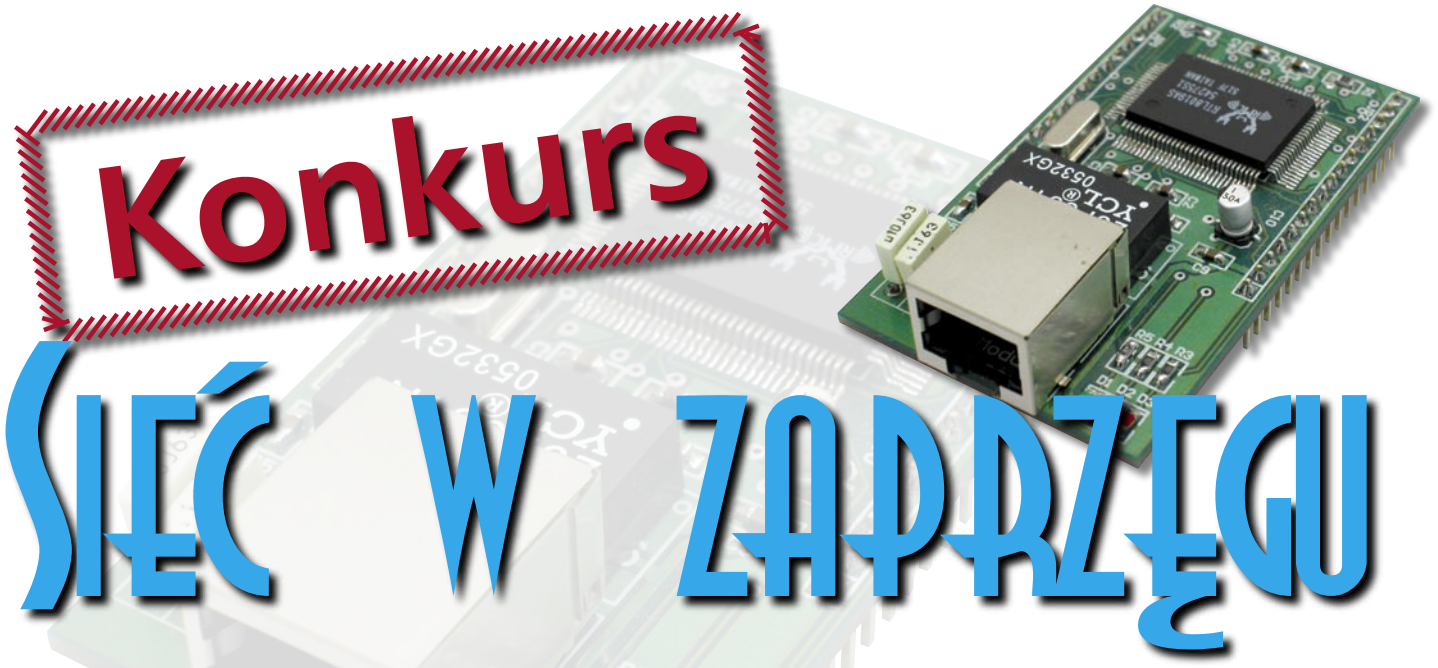


Rys. 13. Schemat montażowy

pomocą JTAG–a opisanego w EP 3/2005 (AVT–1409). Programowanie układu w późniejszej fazie montażu może spowodować nieoczekiwane efekty, ponieważ w wyniku działania starego programu mogą wystąpić konflikty na liniach I/O z innymi układami. Następnie można dokonać montażu reszty elementów na PCB poza procesorem TMS320VC5416. Przed wykonaniem tej czynności trzeba zaprogramować układ XC95288XL, także przez JTAG znajdujący się na złączu CON4. Gdy już mamy poprawnie zmontowany układ, podłączamy klawiaturę i przesuwamy przełącznik SW1 w kierunku przetwornicy. Po podłączeniu zasilania (6...15 V) jednocześnie wciskamy klawisze Ctrl+Alt+Esc. W tym momencie jest „formatowana" pamięć NAND Flash, czyli kasowane są w niej dane i przygotowywana jest odpowiednia struktura w celu obsługi systemu plików. Po odczekaniu kilku sekund podłączamy zestaw poprzez port USB do komputera. System Windows powinien wykryć układ jako dysk wymienny, na który kopiujemy program demonstracyjny (*boot.dat*). W celu uruchomienia programu przesuwamy przełącznik SW1 w kierunku kodeka audio. Na ekranie LCD jest widoczny system plików w pamięci NAND Flash. Nawigacja po nim odbywa się kursorami oraz klawiszami Enter i Esc. Możliwy jest podgląd plików graficznych BMP oraz odgrywanie dźwięku w formacie WAV. Program ten to oczywiście zaledwie ułamek możliwości zestawu. Przykładowo, podłączając przetwornik ADC można w prosty sposób zbudować oscyloskop. Natomiast poprzez zastosowanie odbiornika GPS możemy wyświetlać pozycję na ekranie LCD z użyciem map zapisanych w pamięci Flash. Przy wykorzystaniu mostka PCI-2040 podłączanego do interfejsu HPI8, zestaw może nawet współpracować z magistralą PCI.

Marcin Ostopinko
C5416@wp.pl

R E K L A M A



Sieci internetowe nie są zarezerwowane wyłącznie dla komputerów. W prostych aplikacjach doskonałym rozwiązaniem jest zastosowanie (zamiast PC) urządzenia opartego na mikrokontrolerze. Sprawdza się to doskonale w przypadku zdalnego sterowania lub pomiarów. Mikrokontroler jest znacznie mniejszy (od komputera), zużywa mniej prądu i jest mniej podatny na wszelkie „wpadki" programowo-obsługowe.

Aby mikrokontroler współpracował z Internetem potrzebny jest układ sprzęgający – **interfejs ethernetowy**. Oto moduł (AVT 1443) uniwersalnego interfejsu ethernetowego, który zawiera wszystkie elementy niezbędne do budowy toru Ethernet-procesor.

Możliwości zastosowań tego modułu do różnorodnych aplikacji, korzystających z Internetu, są praktycznie nieograniczone.

Zadanie konkursowe

Prześlij do redakcji EP (e-mail redakcja@ep.com.pl) opis projektu (ze schematem) układu, w którym chcesz wykorzystać moduł AVT 1443. Wszystkim autorom oryginalnych, poprawnych propozycji wyślemy za darmo moduł AVT 1443. Jesteśmy też zainteresowani opublikowaniem wykonanych i sprawdzonych układów z zastosowaniem interfejsu Ethernet.