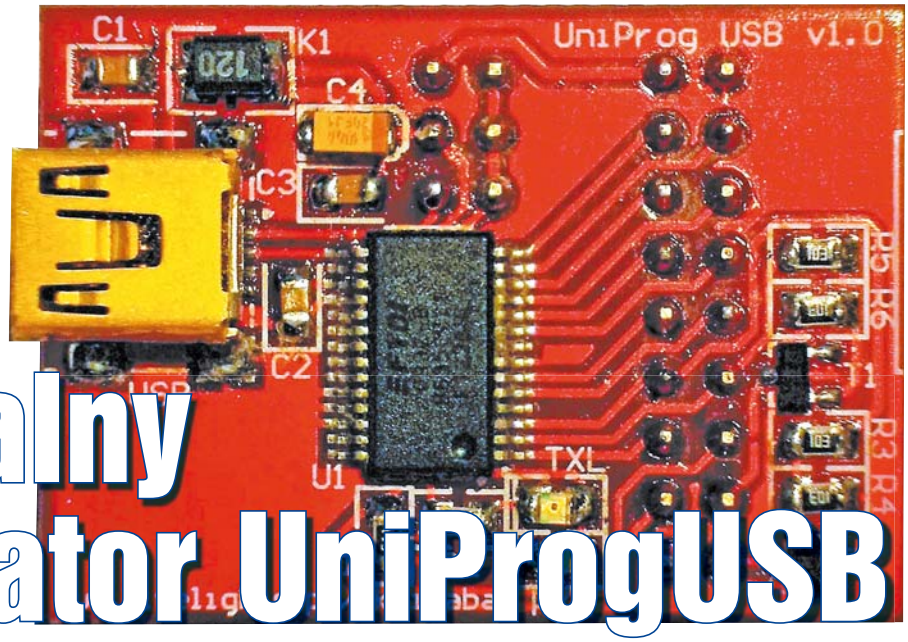
kit 2994 AVT



Uniwersalny programator UniProgUSB

Do czego to służy?

Na rynku dostępnych jest wiele programatorów i debuggerów do najróżniejszych rodzin mikrokontrolerów. Podstawową wadą większości z nich jest możliwość zastosowania tylko do jednego rodzaju układów, a w przypadku niektórych, istotną wadą jest cena, w szczególności w wersjach z interfejsem USB. Opisany poniżej układ powstał głównie z myślą o użytkownikach uniwersalnej makiety dydaktycznej Akai Kaba (*www.livelights.pl/akaikaba*) i ich zapotrzebowaniu na programator sterowany przez USB, a to ze względu na coraz częstszy brak interfejsu LPT w komputerach stacjonarnych, nie wspominając o laptopach. Dzięki swej prostocie i niskim kosztom wykonania jest to konstrukcja przydatna wszystkim osobom, programującym dowolne mikrokontrolery z rodzin AVR, MSP430 i ARM.

Jak to działa?

Pod względem konstrukcyjnym urządzenie jest bardzo proste i dzięki temu także tanie. Schemat elektryczny można zobaczyć na **rysunku 1**. Zapewne wprawne oko wielu Czytelników rozpozna na nim najzwyklejszy konwerter USB-UART, zbudowany z wykorzystaniem bardzo popularnego układu FT232 firmy FTDI Chip. Układ ten pozwala na bardzo proste stworzenie wirtualnego portu COM w komputerze PC, co dodaje do urządzenia dodatkową funkcję właśnie w postaci konwertera USB-UART i to ze wszystkimi liniami tego interfejsu.

Poza wirtualnym COM-em, układ FT232 ma rzadziej wykorzystywaną możliwość pracy w tak zwanym trybie Bit Bang. W tym trybie 8 linii układu, normalnie wykorzystywanych jako linie RX, TX i wszystkie pozostałe sygnały UART-a, staje się 8-bitowym portem dwukierunkowym. Innymi słowy, otrzymujemy możliwość odczytu lub wystawiania stanu logicznego z 8 niezależnych pinów i to ze znaczną prędkością – o wiele większą niż w przypadku mozolnego sterowania liniami zwykłego interfejsu RS232.

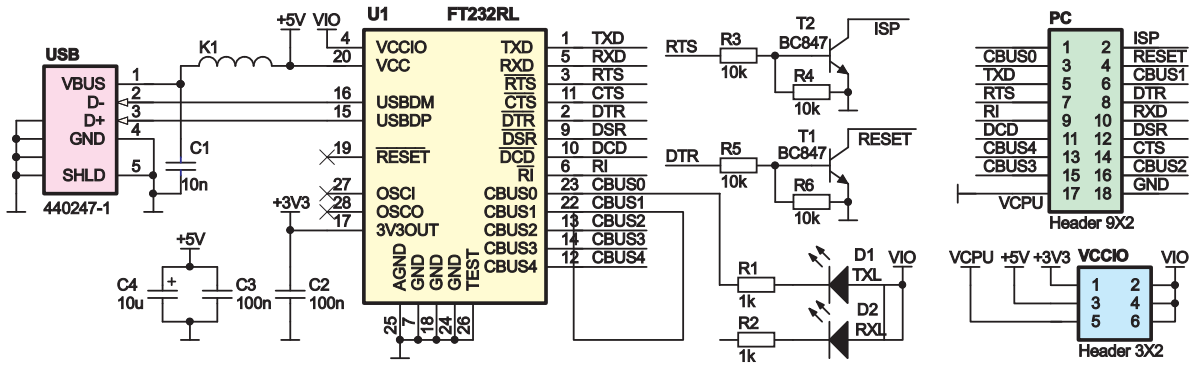
Ponadto sterowniki układu pozwalają na kontrolowanie pięciu dodatkowych linii, oznaczonych jako CBUSx. Domyślnie pełnią one takie funkcje, jak np. sterowanie diodami sygnalizacyjnymi TXL i RXL czy przełączanie kierunku transmisji RS485. Jednak można je wykorzystać także w roli GPIO lub jako wyjście zegara 6, 12, 24 lub 48MHz. Dzięki wyprowadzeniu wszystkich 13 pinów I/O na złącze oznaczone PC, układ poza rolą programatora, może także pełnić wiele innych interesujących funkcji.

Na płytce, poza układem U1 i wspomnianym złączem PC, znajduje się jeszcze garstka innych elementów, w tym oczywiście złącze interfejsu USB w popularnym standardzie mini USB-B. Zasilanie układu U1 pobierane jest bezpośrednio z portu USB i filtrowane przez elementy C1, K1, C3 i C4. Element K1 jest najzwyklejszym koralikiem ferrytowym, jednak na płytce przewidziano na niego sporo miejsca (obudowa 1210), aby można było zastosować egzemplarz o znacznym prądzie. Dzięki temu otrzymujemy możliwość zasilania zewnętrznego układu napięciem +5V i prądem do 500mA (po odpowiedniej konfiguracji interfejsu USB).

Napięcie zasilające piny IO układu można wybrać zworką VCCIO. Pozwala ona na wybranie napięcia +5V, +3,3V z wbudowanego w FT232 stabilizatora (o wydajności do 50mA) lub napięcia z programowanego układu. Dzięki temu, że wybrane napięcie obecne będzie jednocześnie na pinach 2, 4 i 6 VCCIO, można jeden z tych pinów wykorzystać także do zasilania zewnętrznego układu.

Wyjaśnienia wymaga jeszcze obecność tranzystorów T1 i T2. Otóż „klasyczne" programatory, np. procesorów ARM7 firmy NXP, są oparte na porcie szeregowym RS232, a konkretnie o jego linii RX, TX oraz RTS i DTR, służących do wprowadzenia procesora

Rys. 1 Schemat ideowy



w tryb ISP. Aby nie stosować dodatkowej konwersji napięć, najczęściej wykorzystuje się na tych liniach tranzystory w układzie otwartego kolektora, co automatycznie oznacza negację stanów logicznych na tych liniach. Dla ułatwienia współpracy z niektórymi programami, wspierającymi ten sposób programowania, linie DTR i RTS dostępne są zarówno wprost, jak i w opcji zanegowanej właśnie przez tranzystory T1 i T2.

Montaż i uruchomienie

Cały układ programatora mieści się na prostej jednostronnej płytce, której wzór pokazano na **rysunku 2**. Dzięki jednej warstwie płytka jest tania i możliwa do wykonania w warunkach domowych. W przypadku produkcji fabrycznej warto dodać dwustronną soldermaskę i opis – za opcję taką nie trzeba wiele dopłacać, a znacząco ułatwia identyfikację poszczególnych pinów złącza PC i położenie zworki VCCIO. Niestety, w układzie prototypowym firma wykonująca obwód drukowany co prawda naniosła dwustronnie maskę, ale zapomniała o opisie. No cóż, zdarza się.

Montaż najlepiej rozpocząć od układu U1 i złącza USB, następnie wlutować elementy bierne, tranzystory i diody LED, a na koniec goldpiny.

Uruchomienie układu sprowadza się do ściągnięcia i zainstalowania odpowiednich sterowników ze strony:
www.ftdichip.com.
Na stronie dostępne jest także narzędzie do konfigurowania specjalnych funkcji układu FT232. Po zainstalowaniu sterowników na komputerze PC powinien pojawić się nowy port COM. Kontroli działania układu można dokonać, wysyłając dowolne znaki z terminalu i obserwując diody LED. Przy zwarciu linii RX i TX, powinniśmy zobaczyć echo wysyłanych znaków. Po tej operacji możemy przystąpić do rozpakowania i uruchomienia najnowszej wersji mojego programu ***UniProgUSB GUI***, którą można ściągnąć ze strony *www.livelights.pl/akaikaba*.

Program

Do tej pory nie pisałem prawie nic na temat możliwości układu w jego głównej roli, czyli jako programatora. Jak już się przekonaliśmy, pod względem konstrukcyjnym układ jest bardzo prosty, a cały jego potencjał wiąże się z oprogramowaniem, zainstalowanym na PC. Program mojego autorstwa, pod nazwą ***UniProgUSB GUI***, w obecnej wersji umożliwia współpracę programatora z większością mikrokontrolerów AVR, dużą częścią układów MSP430, a także z mikrokontrolerami z rdzeniem ARM7 i Cortex firmy NXP.

Program, który można zobaczyć na zrzutach, na rysunku 3, jest w istocie utworzoną przeze mnie nakładką graficzną na narzędzia do programowania wyżej wymienionych układów, działającą z linii komend.

ARM

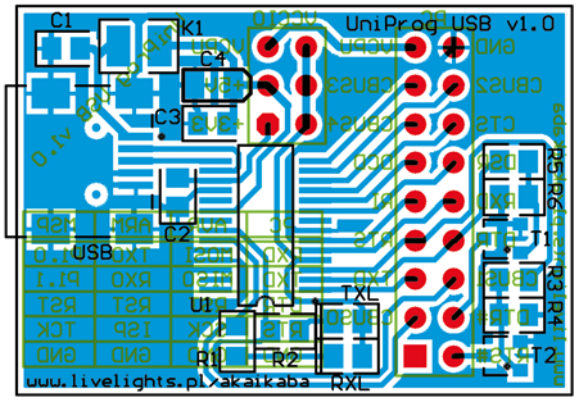
Dla mikroprocesorów ARM jest to program ***lpc21isp*** z bardzo drobną kosmetyką. Program ten wykorzystuje port COM i wbudowany w mikrokontroler bootloader, stąd w okienku nakładki graficznej opcja wyboru portu COM (**rysunek 3a – górny**). Mamy także możliwość wyboru prędkości programowania – w praktyce układy powinny pewnie pracować z prędkością 38400bps, aczkolwiek niejednokrotnie udawało mi się pracować z wyższymi prędkościami, np. 115200 lub 230400bps. Program wymaga także podania (przynajmniej w przybliżeniu) wartości kwarcu, z jakim pracuje mikrokontroler.

Odpowiedni wsad dla mikrokontrolera wybieramy, klikając na przycisk z symbolem folderu i zieloną strzałką. Obok pojawi się nazwa pliku i jego rozmiar, przy czym należy pamiętać, że jest to rozmiar pliku hex, a nie ilość zajętego miejsca w pamięci mikrokontrolera, która w praktyce będzie około 3 razy mniejsza. Po wybraniu plik jest kopiowany i dalsze operacje przeprowadzane są dla bezpieczeństwa na tej kopii. Klikając przycisk z czerwoną strzałką skierowaną do góry, rozpoczynamy proces programowania. Pojawi się wywołany w ten sposób konsolowy program *upusbarm* (*lpc21isp*), w którego oknie zobaczymy dalszy przebieg programowania. Pewną drobną wadą tego podprogramu jest brak możliwości odczytania danych z pamięci procesora ARM.

AVR

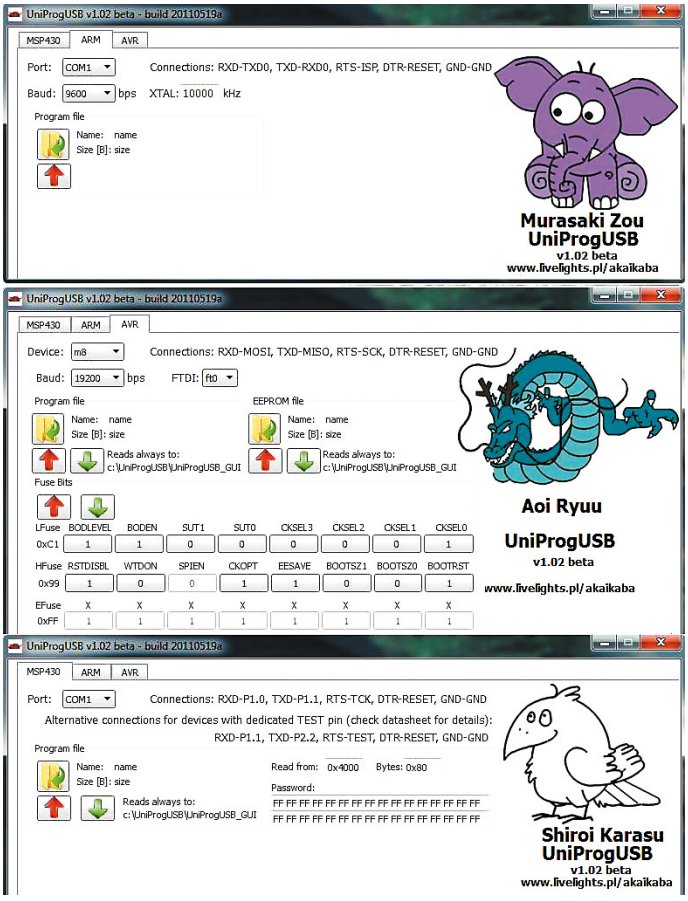
Do programowania AVR-ów UniProgUSB wykorzystuje specjalną kompilację programu AVRDUDE, pozwalającą na wykorzystanie układu FT232 w trybie Bit Bang, dzięki czemu programator jest naprawdę szybki. Nie jest to najnowsza wersja tego programu, gdyż plug-in obsługujący tryb Bit Bang nie wchodzi w skład standardowej rewizji, a jego wkompilowanie okazało się na tyle skomplikowane, że chwilowo zarzuciłem ten pomysł. Niemniej jednak wersja ta obsługuje większość dostępnych obecnie 8-bitowych AVR-ów, mających możliwość programowania przez ISP.

Ponieważ w tej konfiguracji układ FT232 nie tworzy wirtualnego portu COM, w przypadku, gdyby do komputera podłączonych było więcej układów tego typu, zamiast COM wybieramy numer układu, np. ft0, ft1 itd. według **rysunku 3b – środkowy**. Tutaj również możemy wybrać prędkość programowania, a największa dostępna to 230400 i powinna ona działać bez zarzutu. Dodam, że jest to realna prędkość komunikacji z mikrokontrolerem, a nie prędkość układu



Rys. 2 Płytka drukowana (skala 200%)

Rys. 3 Okna programu UniProgUSB GUI



FT232, która w rzeczywistości jest czterokrotnie większa.

Przed zaprogramowaniem należy jeszcze wybrać typ mikrokontrolera w polu *Device*. Jeżeli naszego AVR-ka nie ma na liście, możemy go dodać przez stworzenie odpowiedniego pliku w folderze *avr_devices*, gdzie znajdują się definicje *fusebitów*. Dla ułatwienia, można skorzystać z dodatkowego programiku *avrStudio_2_UniProgUSB.exe*, który konwertuje wybrany plik xml z opisem procesora ze środowiska AVR Studio, do prościutkiego formatu wykorzystywanego przez *UniProgUSB GUI*. Po tej konwersji należy jeszcze dodać nowy mikrokontroler do listy w pliku *avr_list.txt*.

Wczytywanie pliku i programowanie jest analogiczne jak procesorów ARM. Pojawia się tu także dodatkowy przycisk z zieloną strzałką, skierowaną w dół. Przycisk ten służy oczywiście do odczytu informacji z pamięci mikrokontrolera. Ponieważ AVR-y mają dodatkowo pamięć EEPROM, obok części do programowania i odczytu pamięci FLASH, zamieszczony jest identyczny zestaw przycisków dla pamięci EEPROM.

Poniżej znajduje się pole kontroli *fusebitów*. Przyjęto tu filozofię zgodną z opisem w datasheecie, tzn. 1 oznacza bit niezaprogramowany, a 0 zaprogramowany. Nazwy bitów powinny być zgodne z dokumentacją. Kliknięcie na przycisk dowolnego bitu powoduje zmianę jego stanu na przeciwny. Po lewej stronie każdej grupy bitów znajduje się wartość, jaka zostanie zaprogramowana w formacie heksadecymalnym. Wszystkie nieużywane bity są nieaktywne i oznaczone jako „X”. Dla bezpieczeństwa zawsze nieaktywnym będzie także bit SPIEN, dzięki czemu ogromnie zmniejsza się prawdopodobieństwo przypadkowego zablokowania możliwości programowania mikrokontrolera przez ISP.

MSP430

Na **rysunku 3c – dolny** przedstawiona jest zakładka przeznaczona dla mikrokontrolerów MSP430. Być może nie są one jeszcze zbyt dobrze znane Czytelnikom EdW, ale na pewno można znaleźć o nich sporo informacji na łamach siostrzanej EP. Niewiele osób, przynajmniej z mojego otoczenia, ma świadomość, że duża część tych mikrokontrolerów posiada wbudowany bootloader, podobnie jak ma to miejsce w procesorach ARM. Zwykle jedynym znanym tanim sposobem programowania tych układów jest dość prosty debugger, którego wadą jest interfejs LPT. Prawie nie słyszy się o podobnych konstrukcjach na USB, poza drogim rozwiązaniem producenta układów, czyli firmy Texas Instruments.

Tu, dzięki wykorzystaniu wbudowanego bootloadera, otrzymujemy bardzo tani sposób programowania MSP430 przez USB. Po wybraniu portu COM postępujemy analogicznie jak dla procesorów ARM. Podprogramem używanym do programowania tych układów jest zmodyfikowana wersja programu ze strony TI, pod nazwą BSLDEMO. Modyfikacja polegała na zmianie timingów stanów logicznych na liniach wprowadzających procesor do bootloadera (zwanego w dokumentacji BSL). Oryginalne przebiegi wydawały się niezgodne z dokumentacją, a i ta zawierała drobny błąd. Metodą prób i błędów udało mi się jednak ustalić odpowiednią sekwencję działań na tych liniach.

Odczyt z pamięci mikrokontrolera MSP430 następuje w dość specyficzny sposób. W polu *Read from* należy podać adres w pamięci, od którego ma się rozpocząć odczyt, a w polu *Bytes* liczbę bajtów do odczytania. Obie wartości podajemy heksadecymalnie. Ponadto, aby odczytać pamięć, potrzebne jest hasło. Jest nim tablica wektorów, czyli ostanie 32 bajty programu, jakim został zaprogramowany mikrokontroler i można je odnaleźć w pliku *.txt* z programem wgranym uprzednio do mikrokontrolera.

Programowany procesor należy połączyć do odpowiednich pinów programatora za pomocą kilku standardowych kabelków. W każdej zakładce programu jest opis najczęściej stosowanego połączenia. Dla poszczególnych układów MSP430 może on się nieco różnić, zależnie od konkretnego typu i dlatego warto poszukać odpowiednich informacji w datasheecie lub manualu. Dla ułatwienia pracy z wieloma rodzinami mikrokontrolerów, we wszystkich przypadkach wykorzystuje się te same linie programatora, tj. RX, TX, RTS i DTR oraz oczywiście masę i opcjonalnie zasilanie.

Możliwości zmian

Nigdy nie uważałem się za wybitnego programistę, a w szczególności jeśli idzie o programy na komputery PC. Dlatego też program *UniProgUSB GUI* jest napisany po linii najmniejszego oporu. Stąd np. konieczność umieszczenia go w konkretnym katalogu na dysku C i odczyt z pamięci procesora zawsze do konkretnego pliku w tym katalogu. Jeżeli ktoś jest chętny wspomóc rozwój programu, to proszę o kontakt. Dodam, że powstał on w środowisku QT.

Przypominam, że najnowszą wersję programu *UniProgUSB GUI* można ściągnąć ze strony *www.livelights.pl/akaikaba.*

Jeśli chodzi o obecnie wspierane mikrokontrolery, to myślę, że można by rozbudować jeszcze zakładkę AVR o modyfikację lock bitów. Jeżeli znajdzie się czas, a może przede wszystkim zainteresowanie, chciałbym rozbudować program o możliwość programowania mikrokontrolerów ARM z firmy ST, czyli STM32. Firma ta do niedawna udostępniała odpowiednie konsolowe narzędzie, lecz potem usunęła je ze strony, więc będzie się to wiązało z pewnymi poszukiwaniami. Oczywiście do tego i w zasadzie dowolnego mikrokontrolera można użyć osobnych programów, lecz ideą tego projektu było zebranie wszystkich podstawowych narzędzi w jednym miejscu. Niemniej jednak, jeżeli GUI nie posiada wystarczającej funkcjonalności do jakiegoś zastosowania, a posiada ją zastosowany podprogram, to jak najbardziej zachęcam do jego wykorzystania, choćby za pośrednictwem plików wsadowych – przykłady w odpowiednich dedykowanych katalogach pakietu, ściągniętego ze strony *www.livelights.pl/akaikaba,*
np. plik **programuj MSP.bat** znajduje się w folderze UniProgUSB/msp, a plik **programuj ARM.bat** w folderze UniProgUSB/arm.

Jeżeli ktoś byłby zainteresowany dodaniem możliwości programowania innych rodzin mikrokontrolerów czy np. układów programowalnych, to proszę o kontakt. Proszę o kontakt również w wypadku zauważenia jakichś znaczących nieprawidłowości, sugestii, pomysłów. Proszę pisać w wypadku sukcesu, bądź porażki, wykorzystania programatora z mikrokontrolerem, którego jeszcze nie ma na liście w pliku *tested devices list*.

Na koniec jeszcze słowo o obrazkach pojawiających się po prawej stronie w każdej z zakładek programu (jeżeli ktoś nie chce ich oglądać, może zmniejszyć okno) i ich tajemniczo brzmiących nazwach. Otóż są to obrazy, odpowiadające tym zamieszczonym na modułach do makiety Akai Kaba, a ich nazwy zaczerpnięte są z języka japońskiego i są połączeniem zwierzęcia i koloru. I tak przykładowo „Shiroi Karasu” to nic innego jak „Biały Kruk”, „Aoi Ryuu” to „Zielony Smok” (przy czym słowo „Aoi” można też rozumieć jako niebieski), „Murasaki Zou” to „Fioletowy Słoń” – w tym miejscu Czytelnicy znający ten język mogą doszukać się pewnego błędu gramatycznego, za który przepraszam, ale zauważony został już po produkcji pierwszej partii modułu i ludzie zdążyli przyzwyczaić się do nazwy. Nazwa makiety „Akai Kaba” natomiast znaczy „Czerwony Hipopotam”. Ot taki ciekawszy sposób nazewnictwa urządzeń elektronicznych ;-).

Filip Rus
filip.rus@livelights.pl

Wykaz elementów

R1,R2	470...1kΩ SMD
R3-R6	10kΩ SMD
C1	10nF SMD
C2,C3	100nF ceram.
C4	10uF/10V SMD
K1	dławik10...100uH SMD
D1,D2	LED SMD
T1,T2	BC847
U1	FT232RL (SO28)
USB	gniazdo USB
VCCIO	goldpin 3X2
PC	goldpin 9X2

Komplet podzespołów z płytką jest dostępny w sieci handlowej AVT jako kit szkolny AVT-2994.