Od pomysłu do prototypu, czyli narzędzia programisty

Każdy program rozpoczyna istnienie jako pomysł, pomysł rozwiązania jakiegoś problemu czy sposób realizacji jakiegoś zadania. Powstają wtedy główne założenia jakie musi spełniać oraz zarys zadań które musi wykonać. Następnie program trzeba zapisać ale z zachowaniem pewnych zasad. Określają one strukturę i składnię programu, dają pewien zasób funkcji i poleceń.

Środowisko programistyczne

Do pisania programu potrzebne jest środowisko programistyczne tzw. IDE. Jest to zbiór narzędzi, które pomagają w pisaniu programów. Te narzędzia to w rzeczywistości także programy, stworzone przez programistów by ułatwić pracę programistom.

Bohaterem naszego cyklu nauki programowania jest mikrokontroler AVR, zatem musimy wybrać środowisko stworzone dla tej rodziny układów. Producent mikrokontrolera, firma Atmel, udostępnia darmowe środowisko ATMEL Studio. Oczywiście ma ono tyle samo przeciwników, co zwolenników, a największym konkurentem jest środowisko Eclipse. Dla naszych celów ważne jest to, że ATMEL Studio jest łatwe w obsłudze i instalacji.

Pakiet instalacyjny należy pobrać ze strony: *http://www.atmel.com/tools/ATMELSTUDIO.aspx*. Do wyboru jest wersja *web installer*, która ma niewielki rozmiar ale w trakcie instalacji wymaga połączenia z internetem. Druga wersja to *offline* – pełny pakiet instalacyjny (>700MB). W przypadku starszego komputera lub systemu operacyjnego czy w przypadku problemów z instalacją lub działaniem najnowszej wersji, można pobrać którąś z wcześniejszych wersji np. 6.2, ze strony *http://www.atmel.com/tools/STUDIOARCHIVE.aspx*. W ten sposób również otrzymamy pełnowartościowe środowisko dla mikrokontrolerów AVR różniące się tylko nieznacznie od wersji najnowszej. Pobranie każdej z powyższych wersji wymaga zarejestrowania się na stronie Atmela, należy wypełnić kilka pól oraz podać adres e-mail, na który otrzymamy wiadomość weryfikującą.

Instalacja środowiska zgodnie ze wskazówkaminie powinna sprawić problemów.

Mikrokontroler

Załóżmy, że mamy stworzony projekt i napisany program. W takiej postaci, jak widzimy go na ekranie monitora jest dla mikrokotrolera... bezużyteczny. Musi zostać przetłumaczony na język maszynowy, w uproszczeniu chodzi o wygenerowanie pliku z rozszerzeniem hex, jak poniżej.

Program w języku C:

```
while (i < RADIO_READ_WORDS) {
        I2c_read(&reg, I2C_ACK);
        radio.buff[RADIO_READ_ADR + i] = (reg << 8);

        if (i == (RADIO_READ_WORDS - 1)) {
                I2c_read(&reg, I2C_NO_ACK);
        } else {
```

Plik hex:

```
:100110008C6185B987B18F6087B984B18C6184B9EE
:100120008FE19EE40197F1F700C000002A988FE369
:100130009CE90197F1F700C0000083E00E945300A2
:100140008FE19EE40197F1F700C0000083E00E9478
:1001500053008FE99FE00197F1F700C0000083E0B2
:100160000E9453008FE99FE00197F1F700C0000063
:1001700082E00E9453008FE99FE00197F1F700C0F1
```

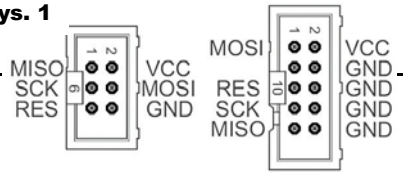
Taki proces przetwarzania programu, nazywany kompilowaniem, wykonywany jest przez środowisko programistyczne, np. w ATMEL Studio po naciśnięciu przycisku F7. Dopiero po skompilowaniu program nadaje się do umieszczenia w pamięci flash mikrokontrolera.

Programator

Przeniesienie skompilowanego programu do pamięci mikrokontrolera wymaga interfejsu i programatora. Interfejs to specjalne złącze w mikrokontrolerze, służące do przesyłania pliku hex wprost do pamięci flash. W mikrokontrolerach AVR najczęściej wykorzystuje się szeregowy interfejs SPI (*Serial peripheral interface*). Wymaga on dołączenia sygnałów oznaczonych MISO, MOSI oraz SCK. Potrzebny jest też dodatkowy sygnał RES oraz linie zasilania VCC i GND. Ten komplet sygnałów tworzy złącze programowania, które często znajduje się na płytkach z mikrokontrolerami AVR. Takie rozwiązanie określane jest terminem ISP – *In system programming* czyli programowanie w docelowym systemie. Spotykamy dwa standardy złącz programowania, 6-szpilkowe ATMEL-ISP oraz 10-szpilkowe KANDA - szczegóły przedstawia **rysunek 1**.

Programowanie przy pomocy ISP nie jest jedynym sposobem, część mikrokontrolerów AVR wyposażona jest w interfejs JTAG, który oprócz programowania, pozwala na debugowanie, czyli usuwanie błędów dzięki dokładnemu bieżącemu kontrolowaniu realizacji programu. Można także programować poprzez bootloader (np w Arduino), co z grubsza biorąc polega na tym, że procesor sam „wciąga" program. Przy „klasycznym" programowaniu, programator potrzebny jest, aby odebrać dane z komputera (plik hex) i wysłać go do mikrokontrolera poprzez interfejs – złącze programowania. Oto dwa najpopularniejsze USB-ASP i AVR ISP MK II, oba podłączane do komputera przez złącze USB.

Rys. 1

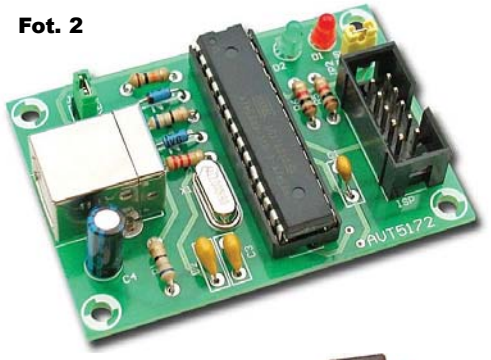


USB-ASP

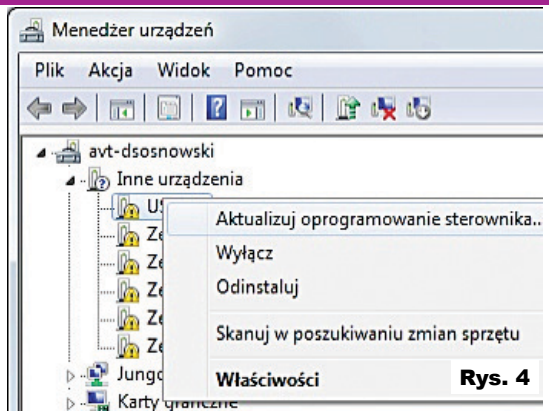
Według wielu opinii, USB-ASP (USBasp) to idealny wybór na początek przygody z programowaniem, bardzo popularny w sieci, obsługiwany przez wiele programów. Zaletą jest bardzo prosta konstrukcja. Wadą to, że do działania wymaga już zaprogramowanego procesora ATmega8. W przypadku samodzielnej budowy programatora „od zera", trzeba dysponować zaprogramowanym układem lub sprawnym programatorem (problemu tego nie ma przy zakupie gotowego programatora). W zasadzie programator obsługiwany jest poprzez oprogramowanie AVRDUDE z linii poleceń, czyli w dość niewygodny sposób, ale powstało wiele programów ułatwiających to zadanie, tzw. GUI np. AVRDUDE-GUI, AVR BurnOmat, itp. Ponadto USB-ASP obsługują takie środowiska jak BASCOM czy ARDUINO. Programatory USB-ASP dostępne w sklepie AVT to np. AVTPROG4, AVT5172 i AVT5325 (**fotografie 2** i **3**).

Niestety, wykorzystywany w ramach kursu C ATMEL Studio nie współpracuje bezpośrednio z programatorem USB-ASP. Nic jednak nie stoi na przeszkodzie, aby także w ramach kursu C z wykorzystaniem płytki testowej lub stykowej, tworzyć programy w ATMEL Studio, a uzyskane pliki hex przenosić do mikrokontrolera przy pomocy innej aplikacji – jest to powszechna praktyka. Oto szczegóły.

Fot. 2



Fot. 3



Rys. 4

Instalacja sterowników USB ASP

Po podłączeniu programatora do komputera należy zainstalować sterowniki. Najnowsze sterowniki dla programatora można pobrać ze strony domowej projektu *http://www.fischl.de/usbasp/,* dostępne w dziale D*ownload.* Bezpośredni link do ostatniej wersji: *http://www.fischl.de/usbasp/usbasp.2011-05-28.tar.gz.* Pobrane archiwum należy rozpakować. Teraz należy uruchomić menedżer urządzeń systemu Windows, odnaleźć w drzewie urządzeń USBasp i wybrać aktualizacje sterowników – **rysunek 4**.

W oknie, które się pojawi, wybieramy jak na **rysunku 5**.

Klikamy na *przeglądaj* i odnajdujemy rozpakowane wcześniej pliki, wybieramy katalog *bin / win-driver / libusb_0.1.12.1* jak na **rysunku 6** i klikamy *Dalej*.

Na ostrzeżenie które się pojawi odpowiadamy jak na **rysunku 7**.

Po chwili powinniśmy zobaczyć okienko informujące o pomyślnym zakończeniu instalacji, a w menedżerze urządzeń programator będzie widoczny jak na **rysunku 8**.

AVRDUDE

Instalacja. Programator USB-ASP obsługiwany jest bezpośrednio poprzez oprogramowanie AVRDUDE, które, oprócz sterowników, również musi znajdować się na komputerze. Te pliki również można pobrać ze strony *http://www.fischl.de/usbasp/,* dostępne w dziale S*oftware*. Bezpośredni link do ostatniej wersji: *http://download.savannah.gnu.org/releases/avrdude/*. Pobrane pliki należy tylko rozpakować, otrzymamy folder o nazwie *avrdude-6.2-mingw32* a w nim dwa pliki o nazwie *avrdude.exe* i *avrdude.conf*.

Instalacja okienka – nakładki. AVRDUDE jest programem konsolowym tzn, że jest obsługiwany z linii komend – jest to sposób mało wygodny. Dlatego zainstalujemy interfejs graficzny dla AVRDUDE tzw. GUI. Powstało wiele takich okienek, wybierzemy jedno z popularniejszych które jest bardzo proste w obsłudze – BurnOmat. Strona domowa projektu: *http://avr8-burn-o-mat.aaabbb.de/avr8_burn_o_mat_avrdude_gui_en.php*,

oraz bezpośredni link: *http://avr8-burn-o-mat.aaabbb.de/AVR8_Burn-O-Mat_2_1_2_setup.exe*.

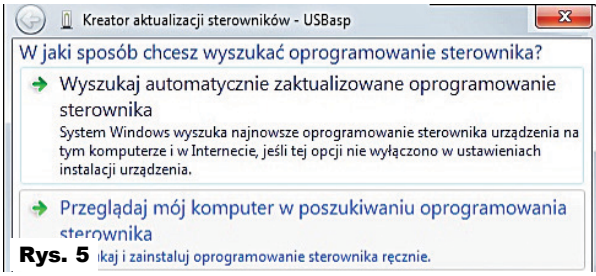
Na początku przeprowadzamy prostą konfigurację – klikamy na *Settings* i wybieramy *AVRDUDE*. Pokaże się okienko, w którym musimy wskazać lokalizację plików AVRDUDE, wybrać rodzaj programatora i port tak na **rysunku 9**. Na koniec klikamy *Apply* i mamy narzędzie gotowe do pracy.

Mając tak zainstalowany pakiet narzędzi do USBasp oraz środowisko ATMEL Studio, możemy już programować mikrokontrolery. W tym celu w ATMEL Studio tworzymy nowy projekt, wpisujemy kod programu w języku C i kompilujemy go wybierając z paska poleceń na górze, *Build* a następnie *Build Solution* lub wciskając na klawiaturze F7. Po zakończeniu kompilowania, w folderze naszego projektu zostanie utworzony folder *Release*, a w nim plik z nazwą projektu i rozszerzeniem *hex*. Teraz musimy przejść do okna programu BurnOmat, upewnić się, że w polu *AVR type* ustawiony jest właściwy typ mikrokontrolera, a następnie kliknąć przycisk *File* i wskazać utworzony plik *hex*. Po kliknięciu na przycisk *Write* mikrokontroler zostanie zaprogramowany wskazanym plikiem, o czym poinformują komunikaty w dolnej części okna.

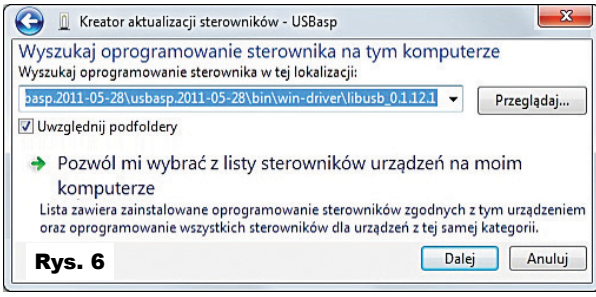
AVR ISP MKII

Interesującym programatorem, który współpracuje bezpośrednio z ATMEL Studio jest AVR ISP MKII. Oprócz wersji oryginalnej dostępne są tzw. klony tego programatora (**fotografie 10** i **11**) – generalnie działają prawidłowo, ale zwykle występują problemy ze współpracą z kolejnymi najświeższymi wersjami ATMEL Studio. Na szczęście programator posiada funkcję aktualizacji firmware'u (funkcja nie działa z poziomu Atmel Studio). W ofercie AVT dostępne są AVT5388 oraz AVTPROG, które mają uaktualniony program, który obecnie działa do wersji **Atmel Studio 7.0 build 594**. Gdy pojawią się nowsze wersje ATMEL Studio, zapewne trzeba będzie uaktualniać oprogramowanie programatora.

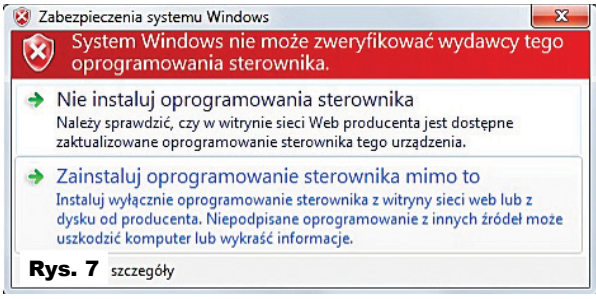
KS
ksavt@wp.pl



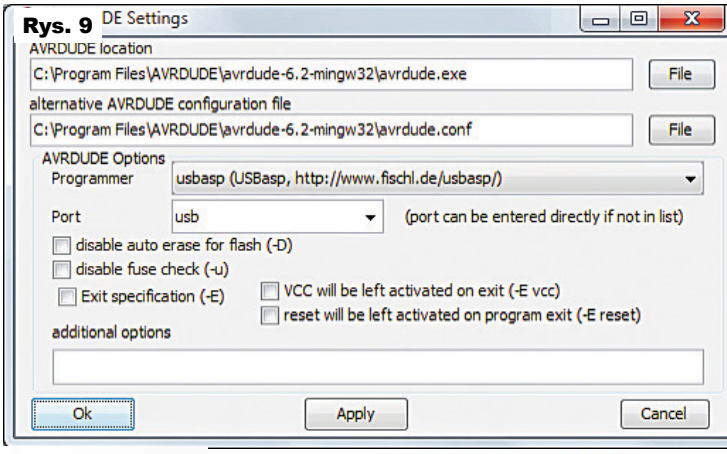
Rys. 5



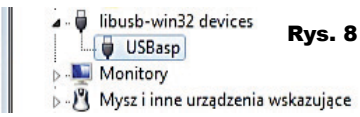
Rys. 6



Rys. 7 szczegóły

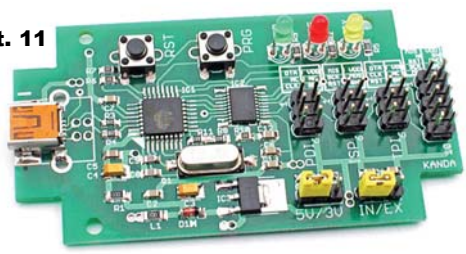


Rys. 9



Rys. 8


Fot. 10


Fot. 11