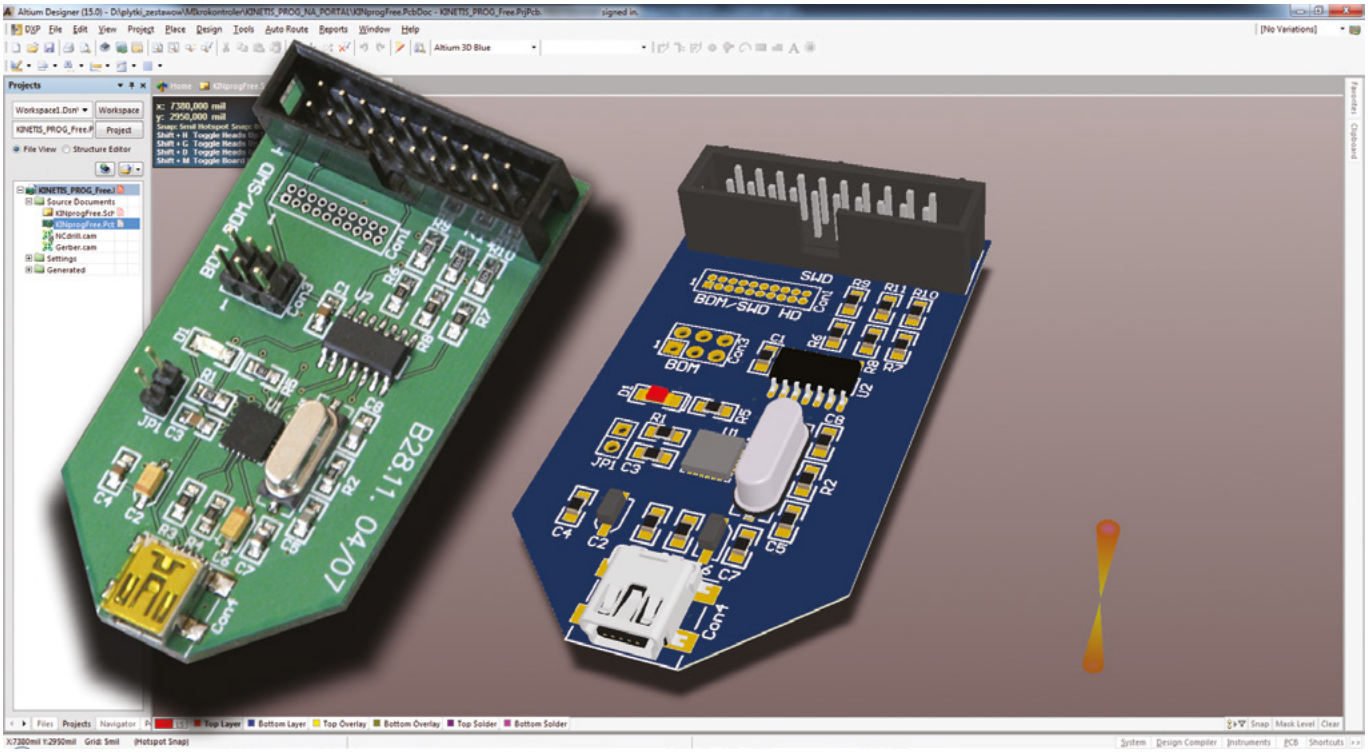
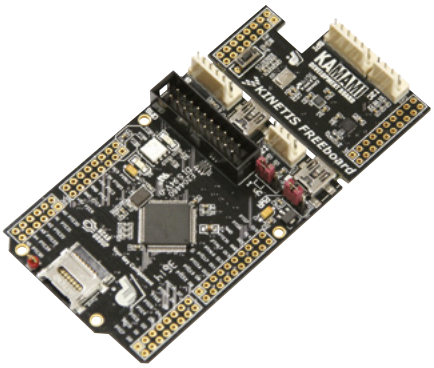


USBDM: programator SWD dla mikrokontrolerów Freescale KINETIS

USBDM to open-source'owy projekt programatora pozwalającego na programowanie i debugowanie pracy większości mikrokontrolerów firmy Freescale, w tym także mikrokontrolerów z rodziny Kinetis, które są „sercem" tanich zestawów startowych z rodziny FREEDOM oraz popularnego zestawu startowego FREEboard z mikrokontrolerem MKL25Z (Cortex-M0+), który do niedawna był dostępny bezpłatnie jako dodatek do książki „Mikrokontrolery KINETIS dla (bardzo) początkujących".

W artykule przedstawiamy projekt konstrukcji sprzętowej programatora USBDM oraz opis jego konfiguracji i sposobu użycia z bezpłatnymi środowiskami programistycznymi firmy Freescale.

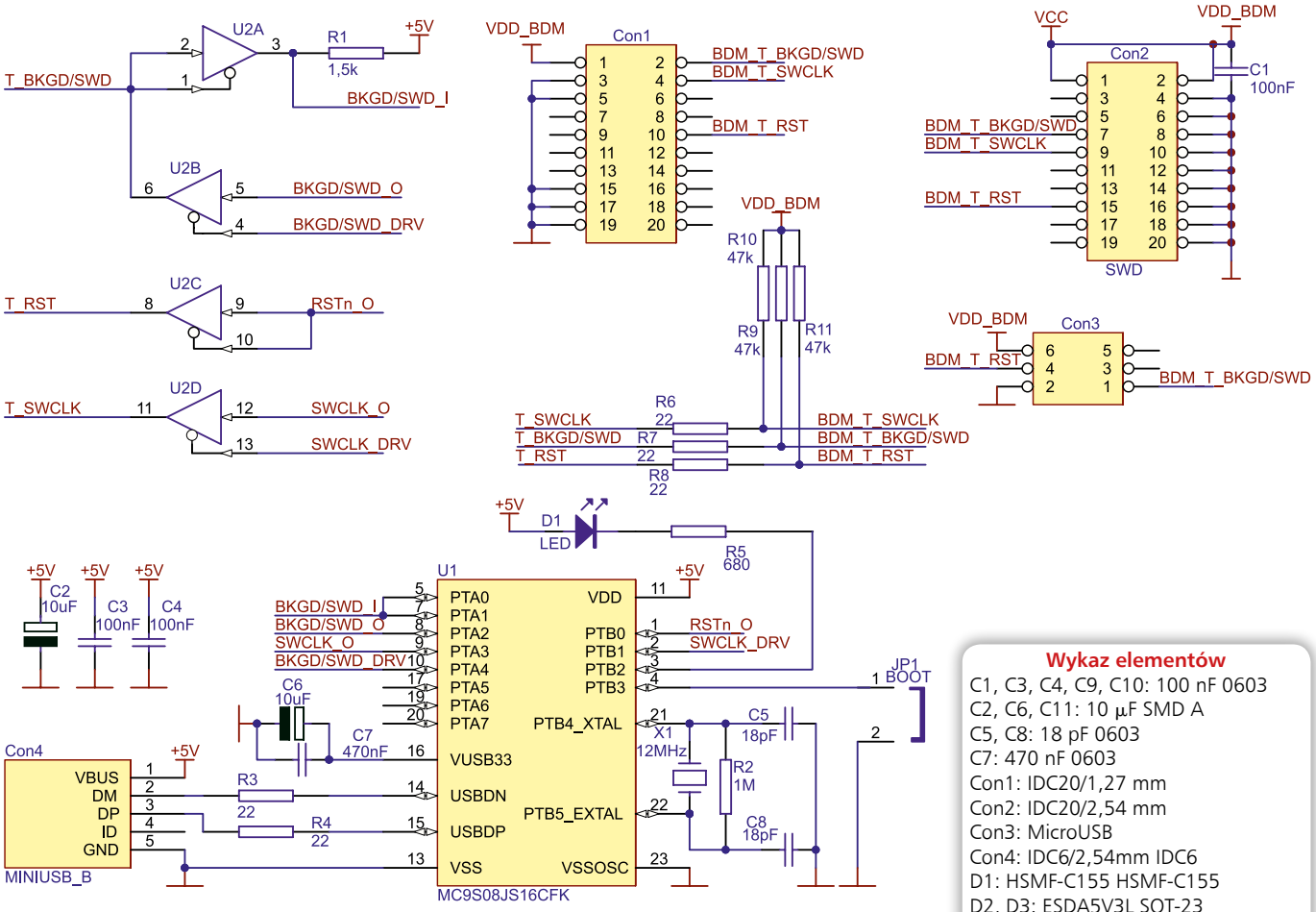


Fotografia 1. Wygląd zestawu FREEboard z mikrokontrolerem MKL25Z128, obsługiwanym przez programator opisany w artykule

W artykule przedstawiamy programator dla mikrokontrolerów produkowanych przez firmę Freescale, który jest rozwiązaniem *open-source*, rozwijanym przez grupę fanów rozwiązań oferowanych przez tego producenta, w tym układów z rodzin RS08, HCS08, HC12, Coldfire V1-4, MC56F800xx oraz – to efekt prac z ostatnich kilkunastu miesięcy – także KINETIS. Szczegółowe informacje o projekcie są dostępne w Internecie pod adresem *http://usbdm.sourceforge.net/*.

Aktywność firmy Freescale na naszym rynku skupia się na promocji mikrokontrolerów niskomocowych (Kinetis L), które jako pierwsze na świecie były wyposażone w rdzeń Cortex-M0+ firmy ARM. Mikrokontrolery z tej rodziny zastosowano w tanich zestawach startowych z rodziny FREEDOM, jeden z najpopularniejszych mikrokontrolerów – MKL25Z – użyto w zestawie FREEboard (**fotografia 1**), który na początku roku był bezpłatnie dodawany do popularnej książki „Mikrokontrolery KINETIS dla (bardzo) początkujących" poświęconej przybliżeniu mikrokontrolerów KINETIS L początkującym. Co interesujące, zestaw ten został wyprodukowany w Polsce (przez firmę KAMAMI) i zgodnie z informacjami, które uzyskaliśmy od producenta, do rąk zainteresowanych trafiło ponad 600 bezpłatnych zestawów tego typu.

Schemat elektryczny programatora USBDM pokazano na **rysunku 2**. Jego konstrukcja jest – jak widać – prosta, urządzenie bazuje na mikrokontrolerze MC9S08JS16 firmy Freescale, który wyposażono w wewnętrzny interfejs USB. Zastosowany



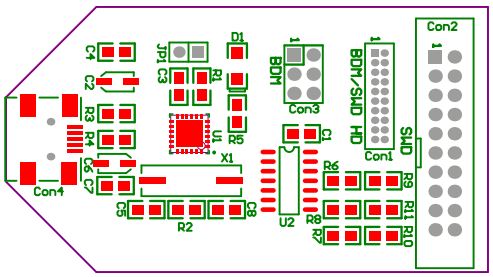
Rysunek 2. Schemat elektryczny programatora USBDM

Wykaz elementów

C1, C3, C4, C9, C10: 100 nF 0603
C2, C6, C11: 10 µF SMD A
C5, C8: 18 pF 0603
C7: 470 nF 0603
Con1: IDC20/1,27 mm
Con2: IDC20/2,54 mm
Con3: MicroUSB
Con4: IDC6/2,54mm IDC6
D1: HSMF-C155 HSMF-C155
D2, D3: ESDA5V3L SOT-23
F1: 500 mA/1206 1206
JP1: SIP-2 z jumperem
R1: 1,5 kΩ 0603
R2, R3: 100 Ω 0603
R4: 10 kΩ CAY10_8PIN
R5: 27 Ω CAY10_8PIN
R6: 1 MΩ 0603
R7, R8: 22 Ω 0603
U1, U2, U3, U4: 74AHC1G125 SOT23-5
U5: MC9S08JS16CFK QFN20_5x5
U6: MCP1700-3,3 SOT-23
X1: 12 MHz SMD 3,2x2,5 mm

mikrokontroler jest standardową platformą sprzętową używana do implementacji programatorów USBDM, dzięki czemu jego użytkownicy będą mieli łatwość dostępu do bezpłatnych aktualizacji firmware, które zapewniają nie tylko adaptację możliwości programatora do zmian na rynku, ale także jego prawidłową pracę z kolejnymi wersjami środowisk programistycznych.

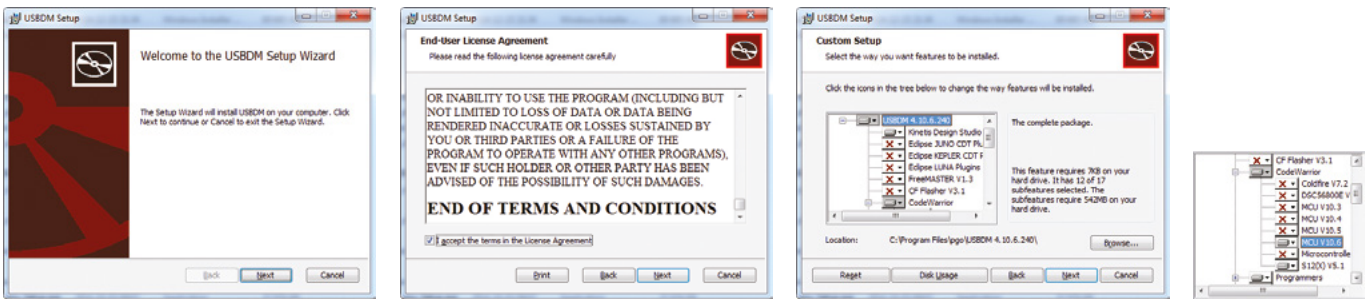
Wróćmy do omówienia budowy programatora: w założeniu ma on się komunikować z programowanym/debugowanym mikrokontrolerem za pomocą dwuliniowego interfejsu SWD. Każda z linii komunikacyjnych jest buforowana za pomocą bramek VHC125, zasilanych takim samym napięciem jak programowany/debugowany mikrokontroler, dzięki czemu można wygodnie programować mikrokontrolery zasilane także skrajnie niskimi napięciami (co jest możliwe, zwłaszcza w przypadku energooszczędnych mikrokontrolerów z rodziny KINETIS L). Układy 74VHC125 mogą pracować zasilane napięciem od 1,8 V lub 2,0 V w zależności od producenta układu.



Rysunek 3. Schemat montażowy programatora USBDM

Programator wyposażono w trzy złącza służące do programowania-debugowania:

- „gęste" Con1 zgodne ze standardem stosowanym przez firmę Freescale m.in. w zestawach Tower,
- 20-stykowe Con2 o rastrze wyprowadzeń 2,54 mm i rozmieszczeniu sygnałów zgodnym z JTAG/SWD (wyprowadzono wyłącznie linie SWD!),\



Rysunek 4. Kolejne etapy instalacji IDE

Projekt płytki drukowanej programatora publikujemy na płycie DVD-EP3/2015 w wersji dla Altium Designera (źródła) oraz w postaci plików Gerber + NC Drill.

- 6-stykowe Con3 o rastrze 2,54 mm, zgodne ze standardem BDM firmy Freescale (przydatne w przypadku niektórych rodzin mikrokontrolerów HC(s)08/ ColdFire).

Dioda świecąca D1 służy do sygnalizacji pracy programatora, miganiem informuje o aktywności interfejsu SWD.

Na **rysunku 3** pokazano schemat montażowy płytki programatora, poza złączami SWD i jumperem zastosowano wyłącznie elementy SMD. JP1 służy do uruchomienia bootloadera USB, który umożliwia zapisanie w pamięci Flash mikrokontrolera U1 firmware, co dokładnie opisujemy w dalszej części artykułu.

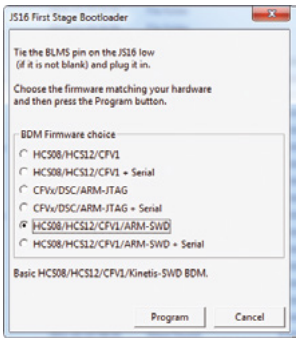
Oprogramowanie narzędziowe i firmware

O ile wykonanie programatora – ze względu na jego prosta konstrukcję – nie wymaga specjalnych zabiegów, to instalacja oprogramowania i skonfigurowanie go do współpracy z programatorem wymaga wykonania kilkunastu kroków, które przedstawiamy w punktach poniżej. Kolejno wykonujemy następujące czynności:

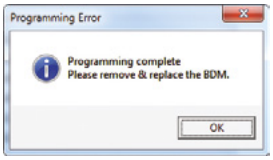
Instalacja zintegrowanego środowiska programistycznego, np. Kinetis Design Studio czy CodeWarrior 10.6 z narzędziem Processor Expert.

Instalacja sterowników USB do programatora USBDM. Są dostępne oficjalne ich wersje dla systemów Windows XP 32/64, Windows 7 32/64. Co do innych systemów operacyjnych należy sprawdzić, czy jest już dostępne odpowiednie wsparcie. Projekt USBDM jest ciągle rozwijany, często pojawiają się poprawki i aktualizacje, warto więc na bieżąco śledzić informacje publikowane na stronie projektu lub forach dyskusyjnych poświęconych temu programatorowi. Aktualne wersje sterowników to:

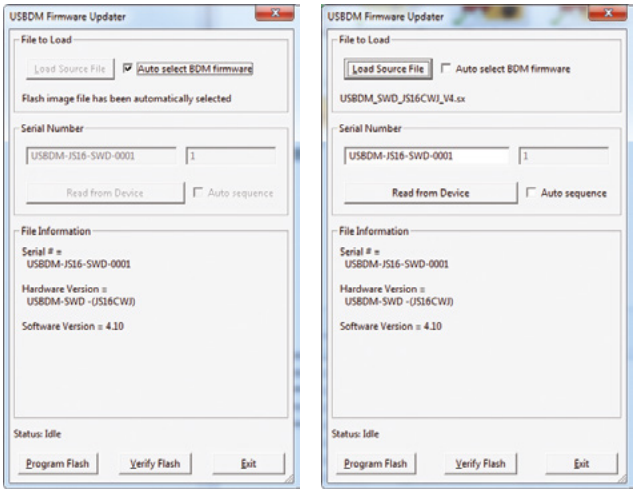
- USBDM_ Drivers_1_3_0_ WinXP_x64.msi
- USBDM_ Drivers_1_3_0_ WinXP_x32.msi
- USBDM_ Drivers_1_3_0_Win_ x64.msi
- USBDM_ Drivers_1_3_0_Win_ x32.msi



Rysunek 5. Okno programu ładującego bootloader USB



Rysunek 6. „Dziwny" komunikat o błędzie, którego treść jest optymistyczna



Rysunek 7. Okno programu ładującego firmware do mikrokontrolera JS16

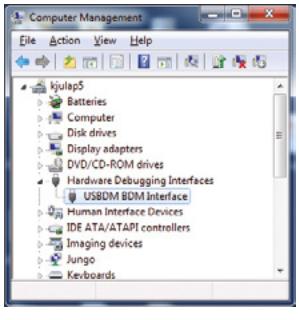
Oczywiście należy wybrać sterownik pasujący do posiadanej wersji systemu operacyjnego.

Instalacja oprogramowania będącego „pośrednikiem" między samym programatorem (sprzętem) a środowiskiem programistycznym. Aktualna wersja jest dostępna pod adresem *http://goo.gl/Gq4c2e* (obecnie jest to 4.10.6.240, ale tutaj sytuacja zmienia się dość dynamicznie i licznik wersji dość szybko rośnie). To oprogramowanie koniecznie należy zainstalować dopiero **po instalacji środowiska zintegrowanego (IDE)**, bo tak jest po prostu najwygodniej. Instalator pakietu USBDM sam wtedy znajdzie na dysku komputera różne wersje zainstalowanych i wspieranych środowisk programistycznych (np. CodeWarrior) i skopiuje potrzebne wtyczki (ang. *plugins*) i pliki konfiguracyjne do odpowiednich folderów. Można to samo zrobić ręcznie, ale zdecydowanie nie polecam tego sposobu.W trakcie instalacji nie trzeba zmieniać domyślnych opcji ustawień. Kolejne etapy instalacji wyglądają jak pokazano na **rysunku 4**.

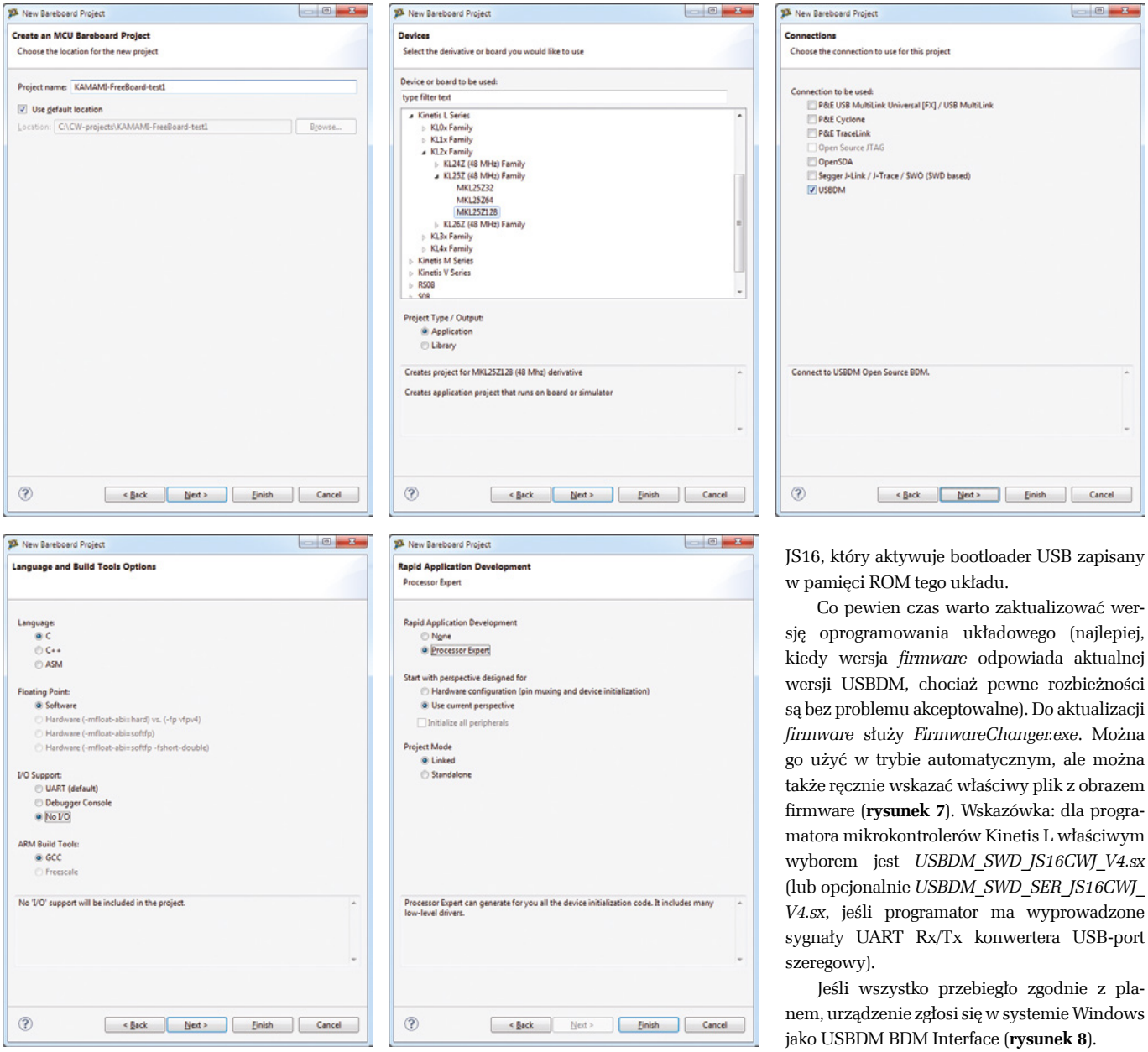
Należy zwrócić uwagę na to, czy została wykryta wcześniejsza instalacja środowiska CodeWarrior MCU V10.6. Jeśli nie, to prawdopodobnie nie została poprawnie zakończona jego instalacja. Można też samodzielnie wskazać lokalizację folderu, w którym zainstalowany został CodeWarrior 10.6.

Dołączenie programatora USBDM.

Jeśli rzeczywiście podłączasz programator po raz pierwszy (masz programator bez załadowanego *firmware*) lub chcesz wymienić oprogramowanie układowe (*firmware*) na nowe, to warto założyć najpierw zamontować zworkę JP1 i dopiero wtedy podłączyć układ do komputera. W trakcie podłączania do portu USB programator przejdzie w tryb ładowania oprogramowania układowego. Ten sam efekt można osiągnąć programowo, ale o tym później. Jeśli rzeczywiście podłączyłeś programator po raz pierwszy, lub po raz pierwszy użyty został konkretny port USB, może być konieczne wskazanie lokalizacji sterowników *bootloadera*. Typowo jest to folder *C:\Program Files\ pgo*. W 64-bitowych systemach Windows ścieżka może dodatkowo zawierać ciąg znaków x64. W zależności od wersji systemu i ustawionego poziomu zabezpieczeń instalacja tych sterowników odbędzie się całkowicie automatycznie lub będzie wymagała ręcznego wskazania ich lokalizacji. Po zainstalowaniu sterowników układ JS16 (JS16 to skrótowe oznaczenie modelu mikrokontrolera będącego „sercem" programatora USBDM) jest gotowy do załadowania oprogramowania układowego (*firmware*). Służy do tego specjalne narzędzie *JS16_Bootloader.exe* zlokalizowane w folderze *C:\Program Files\pgo\USBDM 4.10.6.240* (dokładna ścieżka zależy od wersji systemu operacyjnego i oprogramowania USBDM). Pojawi się następujące okienko jak na **rysunku 5**. Należy wybrać wskazaną na rysunku obok opcję „HCS08/HCS12/CFV1/ARM-SWD" i zaczekać kilkanaście sekund, w tym czasie trwa ładowanie oprogramowania układowego. W tym czasie program może sprawiać wrażenie „martwego", ale nie należy się tym przejmować tylko cierpliwie zaczekać na zakończenie tego procesu, co zostanie zasygnalizowane pesymistycznie brzmiącym komunikatem (**rysunek 6**). Tym razem też nie warto się za bardzo stresować, groźnie brzmiący tytuł „Programming Error" wcale nie oznacza,



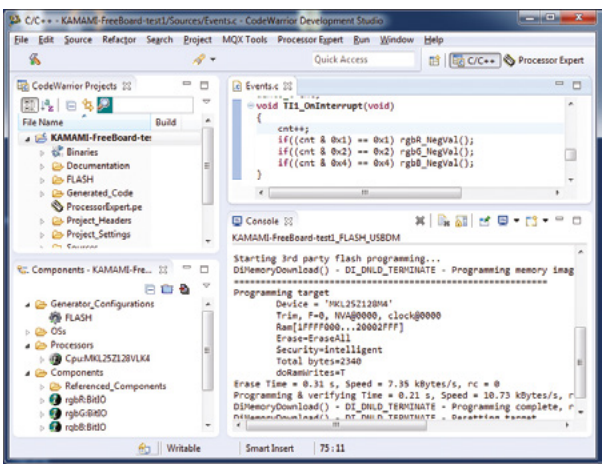
Rysunek 8. Programator jest widoczny w systemie Windows jako USBDM BDM Interface



Rysunek 9. Kolejne kroki zakładania projektu (w Code Warriorze) z zastosowanym programatorem USBDM

że miał miejsce błąd, tak po prostu programista zatytułował okienko z komunikatem sygnalizującym koniec programowania. Zgodnie z wyświetlonym poleceniem należy odłączyć i ponownie podłączyć do portu USB programator. Tym razem zainstalowane zostaną właściwe sterowniki USBDM i zapali się dioda świecąca D1 sygnalizująca aktywność programatora.

Jeśli programator ma już załadowane aktualne oprogramowanie układowe, to nic nie musisz zmieniać, układ jest gotowy do pracy (jest to typowa sytuacja). **Uwaga! W trakcie normalnego używania programatora USBDM zworka JP1 nie może być założona**, w przeciwnym razie układ przejdzie w tryb wymiany *firmware*. Jest to sprzętowy mechanizm wbudowany w układ JS16, który aktywuje bootloader USB zapisany w pamięci ROM tego układu.



Rysunek 10. Widok okna środowiska Code Warrior Development Studio z przykładowym projektem

Co pewien czas warto zaktualizować wersję oprogramowania układowego (najlepiej, kiedy wersja *firmware* odpowiada aktualnej wersji USBDM, chociaż pewne rozbieżności są bez problemu akceptowalne). Do aktualizacji *firmware* służy *FirmwareChanger.exe*. Można go użyć w trybie automatycznym, ale można także ręcznie wskazać właściwy plik z obrazem firmware (**rysunek 7**). Wskazówka: dla programatora mikrokontrolerów Kinetis L właściwym wyborem jest *USBDM_SWD_JS16CWJ_V4.sx* (lub opcjonalnie *USBDM_SWD_SER_JS16CWJ_V4.sx*, jeśli programator ma wyprowadzone sygnały UART Rx/Tx konwertera USB-port szeregowy).

Jeśli wszystko przebiegło zgodnie z planem, urządzenie zgłosi się w systemie Windows jako USBDM BDM Interface (**rysunek 8**).

Programator jest gotowy do pracy. Zakładając nowy projekt należy wybrać USBDM jako interfejs programatora.

Jak korzystać z programatora USBDM – przykład na zestawie FREEboard

W kolejny krokach nadajemy nazwę projektowi, wybieramy odpowiedni model mikrokontroler (w zestawie KINETIS FREEboard jest to MKL25Z128), wskazujemy posiadany programator (USBDM), wybieramy język programowania a w ostatnim okienku zaznaczamy opcję „Processor Expert" (**rysunek 9**).

Po skonfigurowaniu komponentu Cpu oraz dodaniu kilku komponentów takich jak TimerInt (przerwanie zegara) oraz 3 komponenty BitIO (3 sygnały sterujące składowymi *red*, *green*, *blue* diody świecącej RGB, która jest standardowym wyposażeniem zestawu FREEboard) można skompilować program i zaprogramować nim mikrokontroler (**rysunek 10**).

Krzysztof Urbański