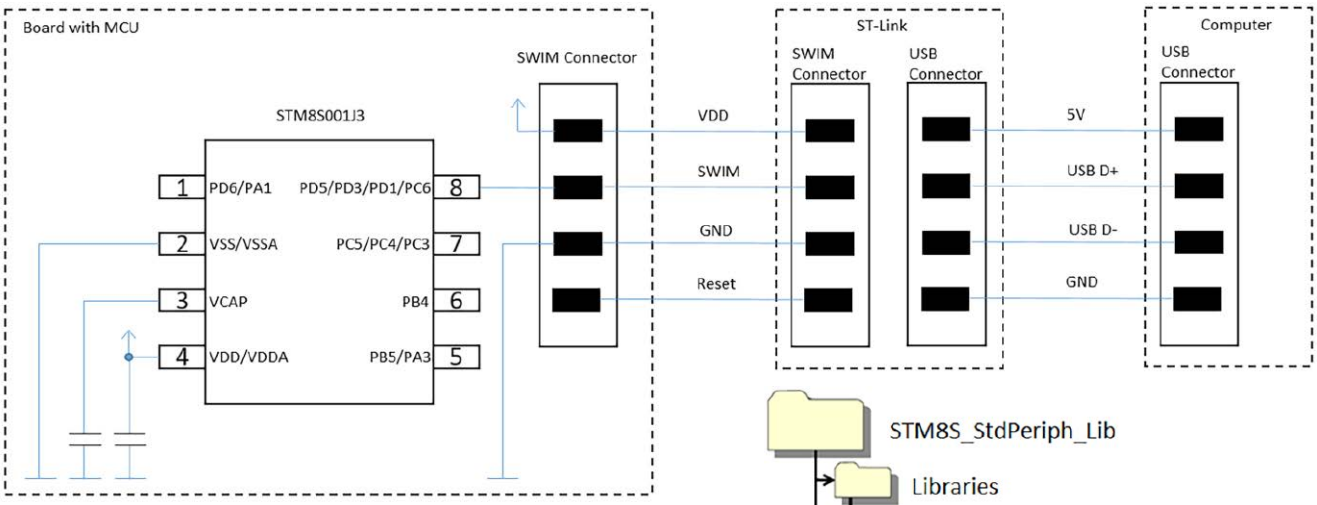
STM8S001J3 (3)

Przegląd narzędzi i rozpoczęcie pracy mikrokontrolerem

Kolejny, trzeci już artykuł z serii dotyczącej 8-pinowego mikrokontrolera STM8S001J3 przybliża temat rozpoczęcia pracy z tym układem. Artykuł podzielono na trzy części: przygotowanie platformy sprzętowej, opis dostępnych narzędzi programowych oraz wykonanie pierwszego projektu za pomocą wybranych przez autora narzędzi programistycznych.

W warstwie sprzętowej należy zadbać o to, aby możliwa była komunikacja między mikrokontrolerem i komputerem w celu przesłania stworzonej przez programistę aplikacji do pamięci mikrokontrolera. Na **rysunku 1** pokazano odpowiedni schemat, który pomaga w zrozumieniu podstawowej konfiguracji sprzętowej wymaganej do tego celu. Analizę tego schematu warto zacząć od samego układu STM8S001J3.



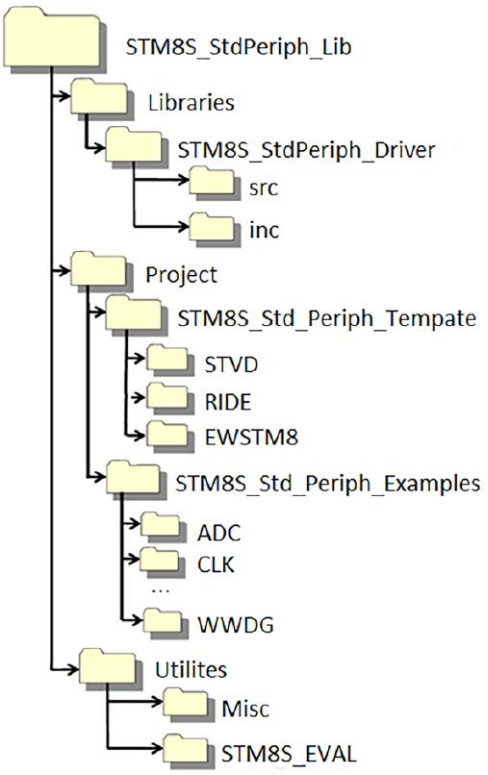
Rysunek 1. Schemat przedstawiający podstawową konfigurację sprzętową pozwalającą na programowanie mikrokontrolera STM8S001J3

Aby zapewnić poprawną pracę mikrokontrolera należy doprowadzić napięcie zasilania (dodatni potencjał do nóżki VDD/VDDA i ujemny do nóżki VSS/VSSA), jak również dołączyć dwa kondensatory (1 µF między nóżką VCAP i ujemnym potencjałem napięcia zasilania oraz 100 nF między nóżką VDD/VDDA i ujemnym potencjałem napięcia zasilania). Następnie należy pomyśleć o interfejsie programowania i debugowania. Nosi on nazwę SWIM i wymaga użycia zaledwie jednej linii sygnałowej, a więc też jednego pinu mikrokontrolera, który oznaczono „PD5/PD3/PD1/PC6". Standardowe złącze programowania/debugowania powinno udostępniać cztery sygnały: dodatni potencjał napięcia zasilania, linię SWIM, ujemny potencjał napięcia zasilania oraz linię reset (tej ostatniej brak w mikrokontrolerze STM8S001J3). Sygnały z takiego złącza za pomocą przewodów są połączone z programatorem/debugerem. Najbardziej popularnym z nich jest ST-Link firmy STMicroelectronics. Urządzenie to jest dostępne w dwóch formach: samodzielnej (ST-Link jako płytka zamknięta w obudowie z wyprowadzonymi złączami) oraz zintegrowanej (ST-Link jako element płytki ewaluacyjnej z mikrokontrolerem – Discovery lub *evaluation board*). Następnie, ST-Link musi być połączony z komputerem. Do tego celu służy interfejs USB. Przewód USB dostarcza również napięcie zasilania dla ST-Linka.

Narzędzia programistyczne

Programiści chcący tworzyć aplikację dla mikrokontrolerów STM8 mogą wybierać spośród szerokiej gamy narzędzi bezpłatnych oraz komercyjnych (krótki wykaz zawarto w **tabeli 1**).

Wśród kompilatorów dostępnych dla układów STM8 warto zwrócić uwagę na produkt firmy Cosmic. Nosi on nazwę CXSTM8 i zawiera kompilator ANSI-C, linker, generator plików .hex i narzędzie do sterowania za pomocą linii komend. Na mocy porozumienia pomiędzy firmami Cosmic i STMicroelectronics, od lutego 2016 r. kompilator CXSTM8 jest oferowany bezpłatnie bez żadnych ograniczeń. Licencja jest przydzielana na rok z możliwością jej odnowienia.



Rysunek 2. Struktura bibliotek SPL

Tabela 1. Wykaz wybranych narzędzi programistycznych dla mikrokontrolerów STM8

Środowisko programistyczne	ST Visual Develop	IAR Systems Embedded Workbsnech	Raisonance Ride7
Możliwy do użycia kompilator	Kompilator Cosmic CXSTM8 lub kompilator firmy Raisonance	Kompilator firmy IAR Systems	Kompilator firmy Raisonance
Debuger sprzętowy	ST-Link (cena: ok. 20 USD)		RLink (cena: 99 Euro)
Wersja bezpłatna	Pakiet CXSTM8 + STVD dostępny bez ograniczeń wielkości kodu i czasu użytkowania	Wersja z ograniczeniem wielkości kodu (8kB) lub czasu użytkowania (30-dni)	Wersja z ograniczeniem wielkości kodu (2kB) lub czasu użytkowania (30-dni)

Aby uzyskać kompletną platformę do tworzenia aplikacji dla mikrokontrolerów STM8, kompilator CXSTM8 należy uzupełnić o środowisko programistyczne (IDE). Takim jest STVD (ST Visual Develop) firmy STMicroelectronics. Zawiera ono narzędzia, takie jak: generator projektu z wyborem mikrokontrolera, edytor kodu z funkcjami kolorowania składni i autouzupełniania, symulator, debuger ze wsparciem dla pracy krokowej, obsługą pułapek i podglądem stanu systemu (zawartości zmiennych i rejestrów) oraz programator pamięci Flash. Środowisko STVD **jest dostępne bezpłatnie**. Dzięki temu, że CXSTM8 i STVD są profesjonalnymi, bezpłatnymi narzędziami, ten zestaw jest chętnie wybierany przez osoby pracujące z mikrokontrolerami STM8.

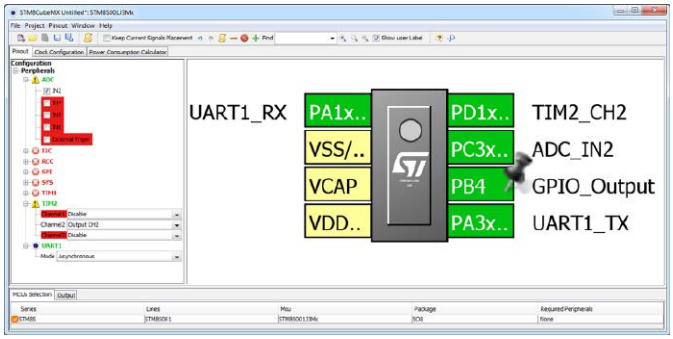
Rozwiązaniem alternatywnym może być środowisko Embedded Workbench firmy IAR Systems. Ten szwedzki producent oferuje środowiska programistyczne dla wielu rodzin mikrokontrolerów (8-, 16- oraz 32-bitowych, w tym bazujących na rdzeniu ARM Cortex-M) z oferty wiodących producentów: STMicroelectronics, Renesas, Texas Instruments, NXP, Atmel, SIlicon Labs, Cypress i Toshiba. Jedną z odmian środowiska Embedded Workbench jest wariant dla układów STM8. Wybór tego środowiska ma kilka istotnych zalet. Pierwszą zaletą jest fakt, iż Embedded Workbench jest kompletnym zestawem narzędzi, a więc nie ma tu konieczności oddzielnego pobierania, instalowania i konfigurowania kilku narzędzi, jak miało to miejsce w poprzednim scenariuszu (CXSTM8+STVD). Drugą zaletą jest pewność, że narzędzia są stabilne i funkcjonalne, po prostu wykonane i testowane przez jednego producenta. Produkt IAR Systems jest obecny na rynku od wielu lat i wyrobił sobie bardzo dobrą markę. Trzecią z zalet jest kompatybilność z różnymi rodzinami mikrokontrolerów. Zatem dla programistów, którzy nie znali dotąd rodziny STM8, ale pracowali już ze środowiskiem Embedded Workbench, rozpoczęcie pracy z układami STM8 będzie znacznie ułatwione. Embedded Workbench for STM8 dostępny jest bezpłatnie w pełnej wersji na czas 30 dni lub bez ograniczenia czasowego, jednak z ograniczeniem wielkości kodu do 8 kB. Pełna wersja bez ograniczeń wymaga zakupu odpowiedniej licencji.

Innym rozwiązaniem komercyjnym jest oprogramowanie oferowane przez firmę Raisonance. W jego skład wchodzi sprzętowy programator/debuger RLink, pakiet Rkit dla układów STM8 (w tym kompilator) oraz środowisko programistyczne Ride7. Produkt firmy Raisonance reprezentuje model biznesowy podobny do opisanego wcześniej rozwiązania firmy IAR Systems. Oznacza to możliwość korzystania z nieodpłatnej wersji produktu z ograniczeniem czasu użytkowania (30 dni) lub wielkości kodu (2 kB) oraz konieczność zakupu licencji na pełną, pozbawioną ograniczeń wersję produktu.

Dodatkowe narzędzia programowe

Kompilatory i środowiska programistyczne wymienione w poprzedniej części artykułu to nie jedyne narzędzia programistyczne, z których można korzystać przy pracy z mikrokontrolerem STM8S001J3. Warto pamiętać również o innych narzędziach uzupełniających ekosystem: Standard Peripheral Library, STM8CubeMX, STM Studio oraz ST Visual Programmer.

Standard Peripheral Library (SPL) to pakiet bibliotek programistycznych będących prostym w użyciu API (*Application Programming Interface*) dla zasobów mikrokontrolera (bloku zegarowego, portów I/O, interfejsów komunikacyjnych, przetwornika A/C itp.). Dzięki ich użyciu programista może uniknąć czasochłonnego studiowania dokumentacji technicznej w celu zrozumienia, jak sterować peryferiami mikrokontrolera oraz równie czasochłonnego pisania kodu w oparciu o modyfikowanie rejestrów. W zamian wystarczy wywołać kilka intuicyjnych i łatwych w użyciu funkcji, które pozwolą na uzyskanie pożądanej konfiguracji i sposobu działania peryferiów. Dzięki temu tworzenie aplikacji staje się znacznie łatwiejsze i tym samym szybsze. Typowo, dla każdego zasobu mikrokontrolera są dostępne dwa pliki biblioteczne: nagłówkowy i źródłowy. Przykładowo, dla przetwornika A/C numer 1 są to pliki *stm8s_adc1.h* oraz *stm8s_adc1.c*. Co istotne, SPL to nie tylko biblioteki dla zasobów mikrokontrolera, ale również szereg przykładowych aplikacji oraz szablony projektów dla trzech środowisk programistycznych: STVD, Embedded Workbench oraz Ride7. Strukturę pakietu SPL pokazano na **rysunku 2**.



Rysunek 3. Widok zakładki *Pinout* w programie STM8CubeMX w warunkach, gdy wybrany jest mikrokontroler STM8S001J3

STM8CubeMX to program komputerowy, który ułatwia tworzenie systemu w warstwie sprzętowej i programowej. Zakładka *Pinout* umożliwia przypisanie peryferiów do wyprowadzeń mikrokontrolera za pomocą graficznego interfejsu użytkownika. Czynność ta pozwala na jednoznaczne stwierdzenie czy peryferia wymagane przez aplikację mogą pracować jednocześnie (czy nie ma konfliktów). Ponadto, schemat ułatwia późniejsze projektowanie schematu elektrycznego i optymalizowanie layoutu płytki PCB. Zakładka *Clock Configuration* prezentuje w zrozumiały sposób całe drzewo sygnałów zegarowych mikrokontrolera. Dzięki schematowi graficznemu i możliwości jego edytowania znalezienie optymalnej konfiguracji z punktu widzenia wymagań danej aplikacji staje się łatwe. Trzecia i zarazem ostatnia zakładka *Power Consumption Calculator* daje możliwość oszacowania, jaki będzie pobór prądu mikrokontrolera w warunkach zbliżonych do tych tworzonych przez aplikację. Po wybraniu wartości napięcia zasilania, listy aktywnych peryferiów oraz konfiguracji zegara systemowego jest wyświetlana wartość poboru prądu. Widok okna programu STM8CubeMX pokazano na **rysunku 3**.

STM Studio to program komputerowy, który umożliwia nieinwazyjne analizowanie zawartości zmiennych w kodzie aplikacji, gdy jest ona wykonywana przez mikrokontroler. W tym celu programator/debuger ST-Link odczytuje przez interfejs SWIM zawartość zmiennych spod wskazanych adresów w pamięci SRAM i przesyła je do komputera, gdzie program STM Studio prezentuje te dane w czasie

rzeczywistym w sposób graficzny (np. w formie wykresu w funkcji czasu).

ST Visual Programmer (STVP) to program komputerowy będący programatorem mikrokontrolerów STM8. Wygenerowany wcześniej przez środowisko programistyczne plik aplikacji w formacie *.hex* lub *.s19* jest wczytywany przez program ST Visual Programmer i przesyłany do pamięci mikrokontrolera za pomocą programatora/debugera ST-Link.

Środowisko ST Visual Develop, kompilator Cosmic oraz biblioteki SPL – pierwsze kroki

W tej części artykułu pokazane zostanie jak rozpocząć pracę z bezpłatnym pakietem narzędzi dla mikrokontrolerów STM8: kompilatorem Cosmic CXSTM8, środowiskiem programistycznym STVD oraz bibliotekami SPL. Dla zwiększenia czytelności poszczególne czynności zostały wypunktowane.

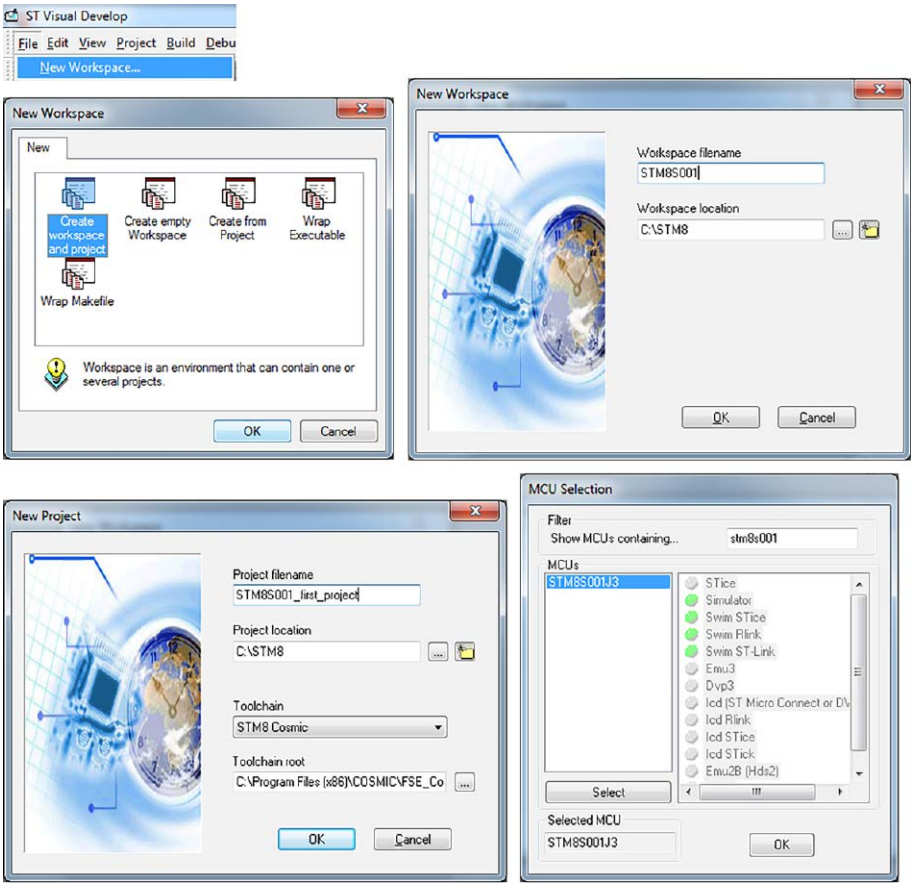
1. Instalowanie

Naturalnie, pierwszą czynnością jest zainstalowanie narzędzi na dysku twardym komputera. Kompilator CXSTM8 można pobrać spod adresu ***http://cosmicsoftware.com/download_stm8_free.php***. Po wypełnieniu formularza strona internetowa udostępni link do pliku instalacyjnego. Pobrany plik należy uruchomić, co pozwoli na zainstalowanie kompilatora. Po zainstalowaniu CXSTM8 będzie wymagane dokonanie rejestracji produktu. W tym celu instalator poprosi o wysłanie wiadomości e-mail do firmy Cosmic zawierającej numer seryjny kompilatora oraz informacje identyfikujące komputer. W odpowiedzi producent przyśle plik z licencją, za której pomocą można aktywować kompilator wskazując mu ścieżkę do pliku licencji zapisanego na dysku twardym. Gdy kompilator CXSTM8 jest już zainstalowany i aktywny, należy przystąpić do zainstalowania środowiska programistycznego STVD. Można je pobrać spod adresu ***http://www.st.com*** – w polu wyszukiwania wpisać nazwę narzędzia „STVD-STM8". Na samym dole strony produktu znajduje się dział *GET SOFTWARE*, a w nim przycisk o tej samej nazwie. Jego naciśnięcie pozwoli na pobranie pliku instalacyjnego najnowszej wersji środowiska programistycznego STVD. Pobrany plik należy uruchomić, co pozwoli na zainstalowanie STVD.

Gdy ten etap jest już zakończony, czas na ostatni krok, a więc biblioteki SPL. Ponownie należy użyć adresu ***http://www.st.com*** wpisując tym razem w polu wyszukiwania nazwę „STSW-STM8069". Użytkownik zostanie przekierowany na stronę bibliotek SPL dla mikrokontrolerów STM8S/A. Analogicznie do poprzedniego narzędzia, należy skierować się na dół strony do działu *GET SOFTWARE*, gdzie przyciśnięcie przycisku o takiej samej nazwie pozwoli na pobranie skompresowanego pliku z bibliotekami i zapisanie go na dysku twardym komputera. Pobrany plik należy rozpakować. Jego instalowanie nie jest potrzebne.

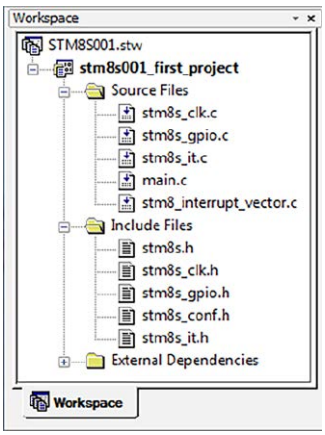
2. Tworzenie nowego projektu w STVD

Po zainstalowaniu wszystkich narzędzi użytkownik może przystąpić do pracy. Rozpoczynamy od uruchomienia środowiska programistycznego STVD. Tworzenie nowego projektu jest realizowane za pomocą kreatora (**rysunek 4**), który jest wywoływany z poziomu menu *File → New Workspace…* W otwartym w ten sposób pierwszym oknie kreatora należy wybrać *Create Workspace and project*. Wybór jest zatwierdzany przyciskiem *OK*. Drugie okno kreatora pozwala użytkownikowi na wpisanie nazwy nowej przestrzeni roboczej oraz



Rysunek 4. Kolejne kroki tworzenia projektu w STVD

ścieżkę na dysku, gdzie zostanie ona zapisana. Kontynuować można po wciśnięciu przycisku *OK*. Kolejne, trzecie już okno kreatora daje możliwość wpisania nazwy projektu (wraz z jego ścieżką na dysku) oraz wybrania kompilatora. W drugim polu należy wskazać *STM8 Cosmic*. Analogicznie do poprzednich kroków wybory akceptowane są przyciskiem *OK*. W ostatnim oknie kreatora jest wybierany mikrokontroler. Na liście należy odszukać „STM8S001J3" i kliknąć przycisk *Select*, a następnie *OK*.

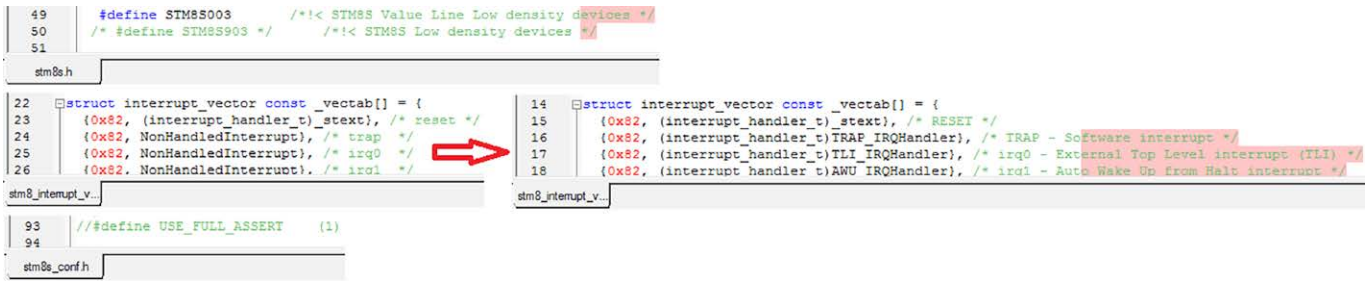


Rysunek 5. Widok drzewa projektu STVD po dodaniu plików SPL

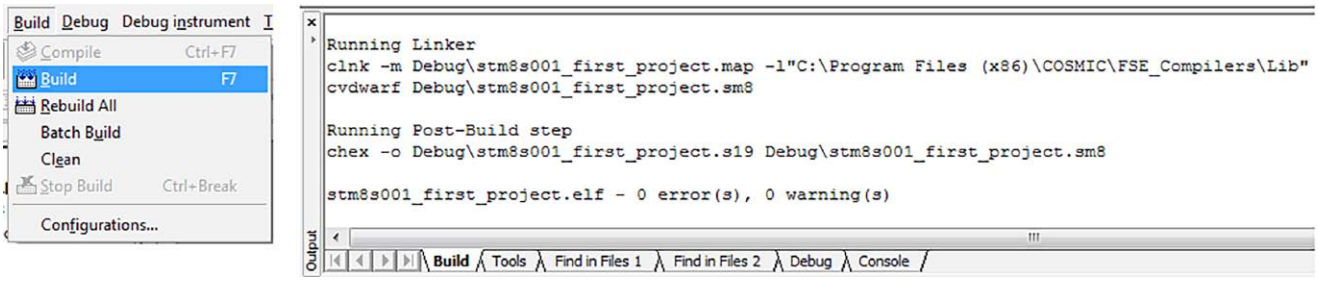
Nowy projekt zostanie utworzony, zapisany na dysku i wczytany do przestrzeni roboczej środowiska STVD. Projekt zawiera trzy katalogi: *Source Files*, *Include Files* oraz *External Dependencies*. Domyślnie w ktalogu *Source Files* znajdują się dla pliki: *main.c* z funkcją *main* i nieskończoną pętlą *while* oraz *stm8_interrupt_vector.c* ze strukturą dla wektorów przerwań. Katatog *Include Files* jest pusty, natomiast katalog *External Dependencies* zawiera jeden plik o nazwie *mods0.h*, w którym zdefiniowany jest model stosu mikrokontrolera.

3. Dodawanie plików bibliotecznych SPL

Projekt należy uzupełnić o dodatkowe pliki bibliotek SPL. Aby całość projektu znajdowała się w jednym miejscu, można skopiować cały katalog SPL do tego samego miejsca, gdzie znajduje się projekt. Zacznijmy od dodania plików źródłowych (z rozszerzeniem .c). W tym celu katalog *Source Files* w drzewie projektu środowiska STVD należy kliknąć prawym przyciskiem myszy i z menu wybrać *Add Files to Folder…* W oknie dodawania plików kolejno należy wskazać katalog SPL (domyślna nazwa



Rysunek 6. Efekt edycji plików *stm8s.h, stm8_interrupt_vector.c oraz stm8s_conf.h*



Rysunek 7. Kompilowanie projektu STVD

to *STM8S_StdPeriph_Lib*), podkatalog *Libraries*, podkatalog *STM8S_StdPeriph_Driver*, podkatalog *src*. Z listy plików należy wybrać te, których nazwy są skojarzone z zasobami mikrokontrolera używanymi przez aplikację. Przykładowo dla prostej aplikacji, która konfiguruje blok sygnałów zegarowych oraz używa portów I/O, należy wybrać pliki *stm8s_clk.c* oraz *stm8s_gpio.c*. Po zaznaczeniu tych plików należy kliknąć przycisk *Open*, dzięki czemu pliki zostaną dodane do projektu STVD.

Analogicznie postąpić należy w przypadku plików nagłówkowych. Katalog *Include Files* w drzewie projektu środowiska STVD należy kliknąć prawym przyciskiem myszy i z menu wybrać *Add Files to Folder…* W oknie dodawania plików należy odnieść się do katalogu SPL i rozklikać podkatalogi w sposób podobny jak poprzednio z tą różnicą, że ostatni podkatalog ma nosić nazwę *inc* zamiast *src*. Teraz zaznaczyć należy pliki związane z zasobami mikrokontrolera (a więc konsekwentnie *stm8s_clk.h* i *stm8s_gpio.h*) oraz dodatkowo wybrać plik *stm8s.h* (z definicjami rejestrów mikrokontrolerów STM8 i bitów tych rejestrów). Kliknięcie przycisku *Open* zakończy proces dodawania plików do projektu STVD.

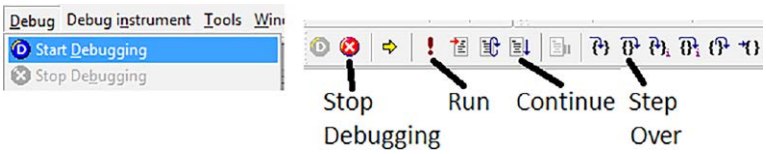
W projekcie brakuje jeszcze trzech plików: pliku *stm8s_conf.h*, który odpowiada za wybór bibliotek SPL dołączanych do projektu STVD oraz plików *stm8s_it.c* i *stm8s_it.h*, w których zdefiniowano handlery do obsługi przerwań. Pliki te należy dodać do projektu tak jak to pokazano w przypadku wcześniejszych plików źródłowych i nagłówkowych. Pliki o wspomnianych nazwach można znaleźć w katalogu SPL wybierając kolejno podkatalogi *Project* i *STM8S_StdPeriph_Template*. Widok drzewa projektu STVD ze wszystkimi wymaganymi plikami pokazano na **rysunku 5**.

4. Edycja plików bibliotecznych SPL

Plik *stm8s.h*. należy sprofilować pod kątem użycia mikrokontrolera STM8S001J3. W tym celu należy go otworzyć w edytorze STVD i w linii kodu numer 49 usunąć komentarz przy definicji *„#define STM8S003”* (biblioteka nie została zaktualizowana o układ STM8S001, jednak układ STM8S003 jest do niego bliźniaczo podobny, stąd istnieje możliwość takiego zastępstwa).

Plik *stm8_interrupt_vector.c* również wymaga edycji. Zdefiniowane w tym pliku wektory przerwań nie mają nazw. Stąd też należy sięgnąć do pliku o tej samej nazwie w podkatalogu *Project*, a następnie *STM8S_StdPeriph_Template*, skopiować zawartość pliku i wkleić ją do pliku w edytorze STVD.

Ostatnim edytowanym plikiem jest *stm8s_conf.h*. Aby uniknąć potencjalnych błędów w kompilacji warto zakomentować linię numer 93 zawierającą kod *#define USE_FULL_ASSERT (1)*. Sposób edycji plików w sposób graficzny pokazano na **rysunku 6**.



Rysunek 8. Narzędzia do debugowania w STVD

5. Kompilowanie projektu, wgrywanie pliku wykonywalnego do pamięci mikrokontrolera, debugowanie aplikacji

Praca nad projektem została zakończona. W tym momencie może on zostać pomyślnie skompilowany. Proces kompilowania (**rysunek 7**) można zainicjować z poziomu menu (*Build* → *Build*) lub przez skrót na klawiaturze (F7). Proces i finalny efekt kompilacji obserwować można w oknie *Output*. Oczekiwany komunikat końcowy to informacja o wygenerowaniu pliku wynikowego o nazwie projektu z rozszerzeniem .elf oraz brak błędów i ostrzeżeń.

W tym momencie środowisko STVD może połączyć się z mikrokontrolerem, aby wgrać do jego pamięci plik wynikowy i rozpocząć debugowanie (**rysunek 8**). W tym celu z poziomu menu wybrać należy *Debug* → *Start debugging*. Środowisko STVD zmieni nieco wygląd udostępniając narzędzia do debugowania, które używać można za pomocą ikon, skrótów klawiatury lub z poziomu menu (opcja *Debug*). Podstawowe z nich to: *Run* (CTRL+F5), *Continue* (F5), *Stop Debugging* i *Step Over* (F10).

Podsumowanie

W artykule skoncentrowano się na narzędziach dla mikrokontrolerów STM8 z wyróżnieniem 8-pinowego układu STM8S001J3. Pierwsza część artykułu dotyczy warstwy sprzętowej, w której pokazano podstawowy schemat elektryczny zapewniający komunikację między narzędziami programowymi/sprzętowymi i mikrokontrolerem. Druga część artykułu przybliża ofertę narzędzi programowych (bezpłatne oraz komercyjne kompilatory i środowiska programistyczne) oraz dodatkowe narzędzia, które ułatwiają pracę z mikrokontrolerem. W ostatniej części artykułu pokazano jak krok po kroku rozpocząć pracę z wybranymi, bezpłatnymi narzędziami: kompilatorem firmy Cosmic, oraz środowiskiem programistycznym i bibliotekami firmy STMicroelectronics. Projekt, którego stworzenie zostało opisane w artykule stanowi szablon dla przyszłych aplikacji, które przedstawione zostaną w kolejnych artykułach w ramach tego cyklu.

Szymon Panecki
szymon.panecki@st.com