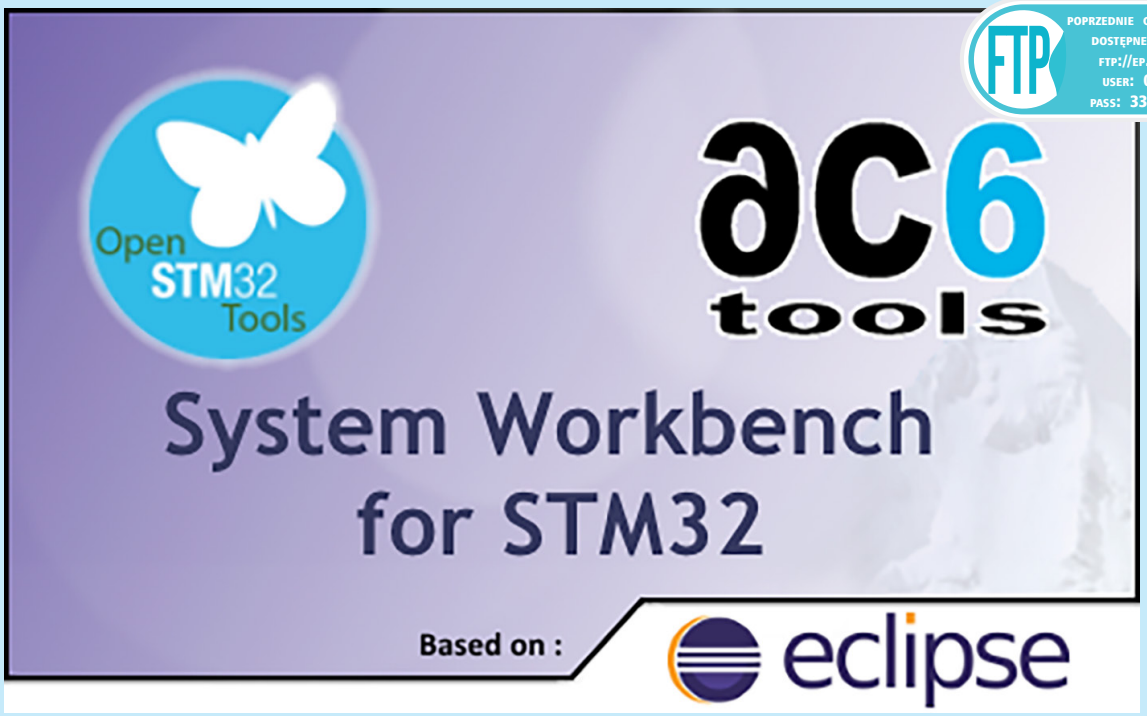


Środowisko programistyczne AC6 System Workbench dla mikrokontrolerów STM32 (2)

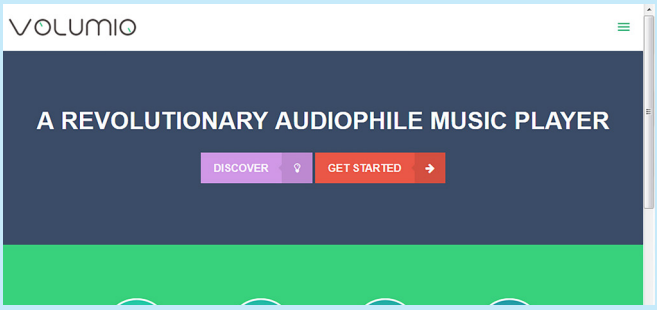
Rozpoczęcie pracy

W pierwszej części tego kursu przedstawiono w zarysie nowe środowisko programistyczne dla mikrokontrolerów STM32, jakim jest AC6 System Workbench. Pokazano ponadto jak je pobrać i zainstalować na komputerze. Teraz przyszedł czas na zaprezentowanie jak za pomocą tego narzędzia tworzyć aplikacje, począwszy od stworzenia projektu, poprzez edytowanie kodu źródłowego, na debugowaniu programu na platformie sprzętowej skończywszy.

Uruchomienie środowiska, tworzenie projektu

Każdorazowo po uruchomieniu programu AC6 System Workbench użytkownik zostanie poproszony o wskazanie ścieżki do przestrzeni roboczej z projektami programistycznymi, tak zwanego *workspace*'a (widok odpowiedniego okna pokazano na **rysunku 1**). Jest to adres miejsca na dysku twardym komputera, do którego AC6 System Workbench odwoła się i wczyta wszystkie obecne tam projekty programistyczne, ponadto nowo utworzone projekty również będą w tej lokalizacji zapisywane.

W AC6 System Workbench, podobnie jak w wielu innych środowiskach programistycznych, aplikacje tworzone są według modelu opartego o projekty programistyczne. Pojedynczy projekt programistyczny stanowi zbiór wzajemnie zależnych plików, na podstawie których można wygenerować plik z kodem wykonywalnym na daną platformę sprzętową (w tym przypadku mikrokontroler STM32). Trzon projektu programistycznego tworzą pliki z kodem źródłowym, w których zaimplementowano interfejs programistyczny do peryferiów oraz zadania realizowane przez aplikację. W przypadku AC6 System Workbench kod jest pisany przede wszystkim w języku C, zatem projekt zawiera pliki nagłówkowe z rozszerzeniem .h i źródłowe z rozszerzeniem .c.

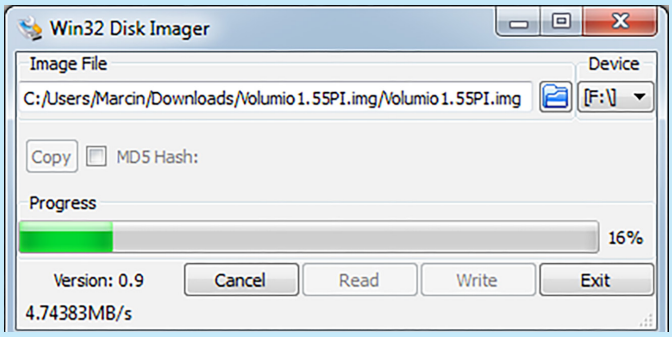


Rysunek 1. Okno ze ścieżką do przestrzeni roboczej AC6 System Workbench

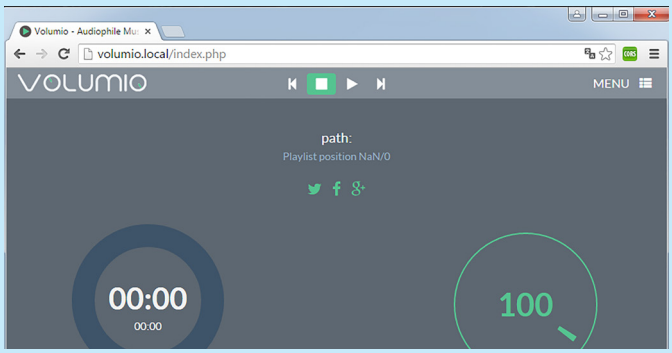
Istnieją dwa sposoby tworzenia projektów programistycznych dla AC6 System Workbench. Pierwszy z nich to użycie zintegrowanego w środowisku kreatora projektu. Drugie to natomiast wykorzystanie do tego celu zewnętrznego narzędzia, jakim jest generator kodu CubeMX, a następnie zaimportowanie projektu do środowiska programistycznego. Poniżej opisany został pierwszy sposób.

Aby wywołać kreator projektu należy z menu głównego środowiska AC6 System Workbench wybrać opcję *File->New->C Project*. W otworzonym w ten sposób oknie o nazwie *C Project* należy w polu *Project name* wpisać nazwę projektu, po czym wybrać rodzaj projektu. Właściwy wybór to *AC6 STM32 MCU Project*.

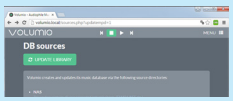
W drugim oknie kreatora (*Select Configuration*) użytkownik wybiera konfigurację projektu. Domyślnie zaznaczone są obie dostępne opcje: *Debug* oraz *Release*. Kolejne, trzecie okno kreatora (*MCU Configuration*) pozwala użytkownikowi na wskazanie platformy sprzętowej. Możliwy jest tu wybór płytki z oferty STMicroelectronics (z serii Discovery, Nucleo lub Eval boards), bądź zdefiniowanie własnej platformy. W czwartym i zarazem ostatnim już oknie kreatora (*Project Firmware configuration*) programista wskazuje, które biblioteki mają zostać dołączone do projektu. Obejmować mogą one sterowniki do peryferiów (starsze rozwiązanie zwane *Standard Peripheral Library*, jak też nowsze: HAL -*Hardware Abstraction Layer*), bibliotekę graficzną STemWin, bibliotekę do interfejsu dotykowego *Touch Sensing Library*, stos USB *host/device*, system plików FatFs oraz system operacyjny FreeRTOS. Po wciśnięciu przycisku *Finish* projekt zostanie stworzony oraz wczytany do AC6 System Workbench. Opisane kroki tworzenia projektu pokazano na **rysunku 2** i **rysunku 3**.



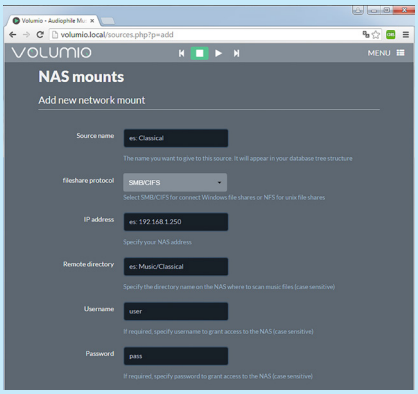
Rysunek 2. Pierwszy etap tworzenia projektu w środowisku AC6 System Workbench



Rysunek 3. Drugi etap tworzenia proejktu w środowisku AC6 System Workbench



Rysunek 4. Widok panelu zmiany perspektywy w środowisku AC6 System Workbench



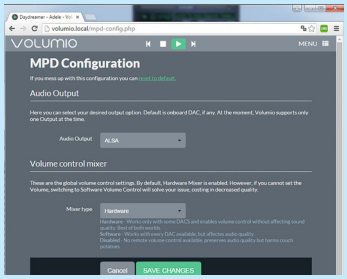
Rysunek 5. Widok perspektywy *C/C++* w środowiksu AC6 System Workbench

Tworzenie aplikacji – etap edycji kodu źródłowego igenerowaniapliku wynikowego

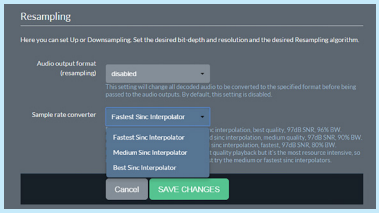
Aby ułatwić nieco programiście pracę, AC6 System Workbench potrafi dostosować swój wygląd indywidualnie do realizowanych przez programistę czynności. Do tego celu używany jest tak zwany mechanizm perspektyw. Perspektywa określa jakie okna narzędziowe są w danej chwili widoczne. Środowisko firmy AC6 System Workbench oferuje łącznie kilkanaście perspektyw. Najistotniejsze z nich to *C/C++* oraz *Debug*. Pierwsza służy do edycji kodu źródłowego oraz przetwarzania go do postaci pliku wynikowego. Druga używana jest do weryfikowania poprawności działania aplikacji w procesie debugowania jej na platformie sprzętowej. Wybranie perspektywy odbywa się przez naciśnięcie na odpowiadającą jej ikonę w panelu wyboru perspektywy. Umiejscowiony jest on w prawym górnym rogu okna środowiska AC6 System Workbench (**rysunek 4**).

Perspektywa *C/C++* (**rysunek 5**) udostępnia trzy okna: *Project Explorer*, edytor oraz *Console*. Okno *Project Explorer* służy do zarządzania projektami programistycznymi. Są w nim wyświetlone wszystkie projekty, które wczytane zostały spod adresu na dysku twardym podanego przez programistę jako obszar przestrzeni roboczej. Do przykładowych operacji realizowanych przy pomocy tego okna należy dodawanie nowych lub usuwanie istniejących projektów, importowanie projektów z innych lokalizacji, wywoływanie okna ustawień wskazanego projektu, modyfikacja nazw plików, wybieranie które z plików mają zostać otwarte do edycji itp.

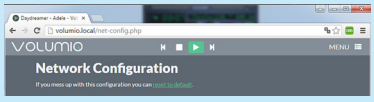
Okno *Project Explorer* stanowi punkt wyjścia dla drugiego narzędzia należącego do perspektywy *C/C++*, a mianowicie edytora, który jak sama nazwa wskazuje umożliwia edycję plików projektu programistycznego. Programista chcąc przejść do trybu edycji danego pliku wyszukuje go w oknie *Project Explorer*, a następnie dwukrotnie na niego klika. W efekcie plik zostanie otworzony do edycji w oknie edytora, które znajduje się w centralnej części programu AC6 System Workbench. W edytorze może być otwarty więcej niż jeden plik. Każdy z otwartych plików ma przyporządkowaną zakładkę, dzięki czemu można się między nimi w wygodny sposób przełączać. Samo narzędzie edytora ma standardową funkcjonalność. Otwarte w nim pliki mają postać tekstu, na którym można wykonywać typowe operacje np. dopisywania, usuwania, modyfikowania itp. Tekst w edytorze jest przeważnie kodem źródłowym napisanym w języku C. Programista używając elementów tego języka takich jak np. zmienne, tablice, struktury, wskaźniki, funkcje czy też instrukcje warunkowe może rozwijać kod źródłowy aplikacji. Edytor oferuje ponadto różne mechanizmy, które ułatwiają czytanie kodu. Są to między innymi: kolorowanie składni języka programowania, numerowanie linii kodu, zwijanie/rozwijanie bloków kodu, wskazywanie



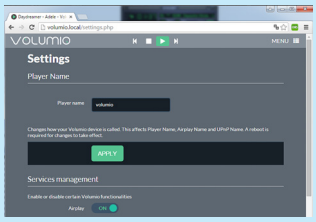
Rysunek 6. Widok okna Debug Configurations w środowisku AC6 System Workbench



Rysunek 7 Widok perspektywy *Debug* w środowiksu AC6 System Workbench



Rysunek 8. Widok okna *Breakpoint* w środowisku AC6 System Workbench



Rysunek 9. Dostęp do wybranych mechanizmów debugowania w środowisku AC6 System Workbench

Rysunek 10. Widok okna *Expressions* w środowisku AC6 System Workbench

klamry nawiasu otwierającego/zamykającego blok kodu (sparowanej z klamrą zaznaczoną).

Po zakończeniu pisania kodu aplikacji programista uruchamia proces, który przy użyciu narzędzi takich jak kompilator i linker przekształci pliki z kodem źródłowym w plik wykonywalny przez mikrokontroler. Tenże proces w środowisku AC6 System Workbench wywołać można z menu głównego, wybierając *Project*, a następnie *Build Project*. Informacje o przebiegu i wyniku tej operacji wyświetlane są w oknie *Console*. W przypadku gdy kompilator i linker nie natrafią na żadne problemy i zadanie tworzenia pliku wykonywalnego zakończy się powodzeniem, w oknie *Console* jako ostatnia wyświetlona zostanie wiadomość *Build Finished*. Jeśli jednak problemy zostaną wykryte, plik nie zostanie utworzony, a w oknie Console wyświetlone zostaną komunikaty o źródle problemów. Przeważnie są to błędy w składni kodu źródłowego.

Tworzenie aplikacji – konfigurowanie debugera

Przed pierwszym użyciem debugera środowiska AC6 System Workbench najpierw należy przygotować to narzędzie do pracy. Aby skonfigurować debuger programista wybiera z menu głównego programu AC6 System Workbench opcję *Run*, a potem *Debug Configurations...* W otworzonym w ten sposób oknie *Debug Configurations* należy wybrać debuger, który ma być używany. Właściwy wybór to *Ac6 STM32 Debugging*. Po dwukrotnym kliknięciu na tą nazwę stworzona zostanie nowa instancja debugera. Najważniejsze ustawienia dostępne są w zakładce *Main*. W polu *C/C++ Application* należy podać ścieżkę do pliku z rozszerzeniem .elf. Z kolei w pole *Project* należy wpisać nawzę projektu. Okno konfiguracji debugera z prawidłowymi ustawieniami pokazano na **rysunku 6**.

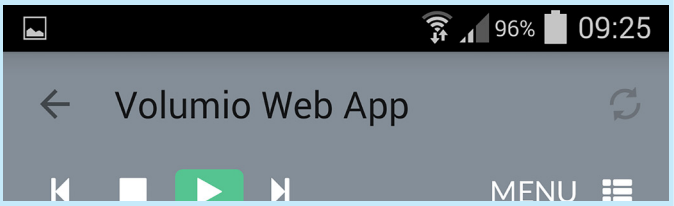
Tworzenie aplikacji – etap debugowania

W celu rozpoczęcia procesu debugowania aplikacjiz menu głównego programu AC6 System Workbench należy wybrać *Run*, a następnie *Debug*. W tym momencie środowisko programistyczne wymusi zapis pliku wykonywalnego aplikacji do pamięci mikrokontrolera. Następnie mikrokontroler zostanie zresetowany i przygotowany do rozpoczęcia wykonywania aplikacji. Jednocześnie w AC6 System Workbench perspektywa *C/C++* zostanie automatycznie zmieniona na *Debug* (**rysunek 7**). Dzięki temu programiście udostępnione zostaną okna narzędziowe przeznaczone do debugowania aplikacji.

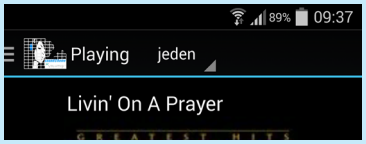
Powszechnie stosowane są dwie metody debugowania aplikacji. Pierwsza z nich to tak zwana praca krokowa. Nazwa dobrze oddaje tu zasady, według których odbywa się debugowanie. Kod źródłowy wykonywany jest etapami, a nawet linia po linii. Druga metoda debugowania opiera się na wykorzystaniu tak zwanych pułapek. Pułapka to wybrana przez programistę linia kodu źródłowego, po wykonaniu której działanie aplikacji zostanie wstrzymane. Pułapki stosowane są zazwyczaj w funkcjach obsługi przerwań lub w ciałach instrukcji warunkowych. W tym scenariuszu aplikacja działa w trybie pracy ciągłej, w trakcie której pułapki informują o wystąpieniu różnych zdarzeń. Aby ustawić pułapkę należy w edytorze najechać kursorem na lewy margines linii kodu, której wykonanie ma wyzwalać pułapkę, a następnie dwukrotnie kliknąć. W miejscu kliknięcia na potwierdzenie i oznaczenie ustawienia pułapki wyświetlona zostanie symbolizująca ją kropka koloru niebieskiego. Pułapki można włączać i wyłączać bez konieczności trwałego ich usuwania. Zarządzanie pułapkami realizowane jest z poziomu okna *Breakpoints* (**rysunek 8**).

Debugowanie zarówno metodą pracy krokowej, jak tez metodą z użyciem pułapek programista realizuje korzystając z dedykowanych do tego celu komend udostępnianych przez środowisko AC6 System Workbench. Najważniejsze komendy to: *Step Into* (wejście do wnętrza bloku kodu np. funkcji), *Step Over* (przejście do kolejnego, najbliższego bloku kodu), *Resume* (wznowienie działania aplikacji), *Suspend* (wstrzymanie działania aplikacji), *Terminate* (zatrzymanie działania aplikacji) oraz*Reset* (powrót do początku wykonywania aplikacji). Komendy te wywoływać można z poziomu menu AC6 System Workbench. Ponadto są one również częściowo dostępne w formie ikon (**rysunek 9**).

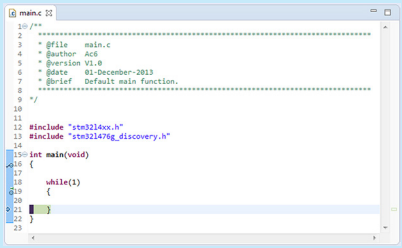
Debugowanie aplikacji poprzez pracę krokową, czy też za pomocą pułapek sprawia, że programista wstrzymuje pracę mikrokontrolera w wybranych przez siebie momentach działania aplikacji. W tych momentach programista zapoznaje się z informacjami o stanie systemu.



Rysunek 11. Widok okien *Registers* oraz *I/O Registers* w środowisku AC6 System Workbench



Rysunek 12. Widok okna *Memory* w środowisku AC6 System Workbench



Rysunek 13. Widok okna edytora w środowisku AC6 System Workbench

Z myślą o tej czynności AC6 System Workbench udostępnia okno *Expressions* (pokazano je na **rysunku 10**). Pozwala ono na wgląd w wartość danych aplikacji. Każda zmienna, tablica i struktura, którą programista chce obserwować, musi zostać dodana do tych okien ręcznie. Czynność tą wykonuje się poprzez zaznaczenie w edytorze wybranej danej, wywołanie prawym przyciskiem myszy menu kontekstowego oraz wybranie w wyświetlonej listy opcji *Add Watch Expression*… Alternatywnie można kliknąć na ikonę plusa w oknie *Expressions*, a następnie wpisać ręcznie nazwę danej.

Bardziej dogłębną informację o stanie systemu może dać analiza zawartości rejestrów mikrokontrolera. W AC6 System Workbench podgląd rejestrów układu umożliwiają okna *Registers* (rejestry rdzenia Cortex-M) oraz *I/O Registers* (rdzenie peryferiów), których widok pokazano na **rysunku 11**.

W określonych sytuacjach programista może mieć potrzebę wglądu w zawartość pamięci mikrokontrolera na poziomie bajtów. Taką informację można uzyskać korzystając z okna *Memory*. W lewej kolumnie należy wpisać adres pamięci mikrokontrolera. W prawej kolumnie wyświetlona zostanie zawartość obszaru pamięci począwszy od wskazanego adresu. Widok okna *Memory* pokazano na **rysunku 12**.

Ostatnim z okien debugowania w AC6 System Workbench jest edytor. Poprzez kolorowanie na zielono tła linii kodu i zaznaczanie jej dodatkowo strzałką edytor informuje programistę na jakim etapie wykonywania aplikacji jest w danej chwili mikrokontroler. Przykład widoku okna edytora w trybie debugowania zaprezentowano na **rysunku 13**.

Podsumowanie

Środowiska programistyczne oparte na Eclipse niejednokrotnie kojarzą są z koniecznością wykonania mało intuicyjnej i tym samym czasochłonnej konfiguracji tej platformy na różnych etapach pracy programisty. W przypadku AC6 System Workbench twórcy dołożyli starań, aby było ono w możliwie dużym stopniu przyjazne użytkownikowi. W efekcie środowisko po instalacji nie wymaga konfiguracji i jest gotowe od razu do użycia. Ponadto proces tworzenia aplikacji jest bardzo czytelny, począwszy od stworzenia kompletnego projektu programistycznego w zaledwie kilku krokach, poprzez edycję kodu źródłowego, kompilowanie go i debugowanie programu za pomocą typowych, dobrze znanych z innych środowisk narzędzi.

Szymon Panecki, EP

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 07643, pass: 332wwppm