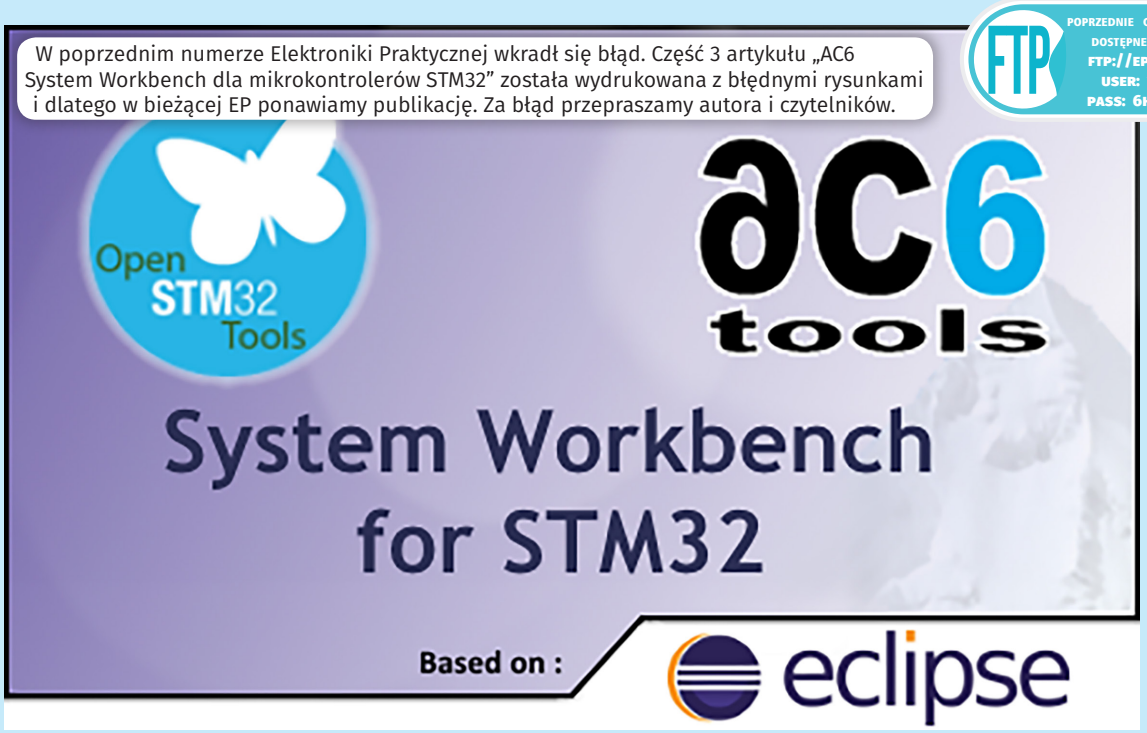
W poprzednim numerze Elektroniki Praktycznej wkradł się błąd. Część 3 artykułu „AC6 System Workbench dla mikrokontrolerów STM32" została wydrukowana z błędnymi rysunkami i dlatego w bieżącej EP ponawiamy publikację. Za błąd przepraszamy autora i czytelników.






# Środowisko programistyczne AC6 System Workbench dla mikrokontrolerów STM32 (3)

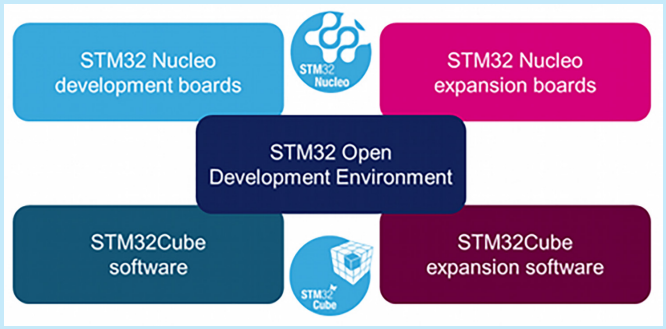
## Współpraca z ekosystemem narzędzi Open Development Environment

**Korzystanie z nowego środowiska programistycznego AC6 System Workbench wiąże się z różnymi udogodnieniami dla programistów tworzących systemy oparte na mikrokontrolerach z rodziny STM32. Jednym z nich jest możliwość współpracy z ekosystemem narzędzi STM32 ODE (Open Development Environment). W artykule pokazano krok po kroku jak przy ich wykorzystaniu stworzyć kompletną aplikację.**

Firma STMicrolelectronics oferuje atrakcyjny ekosystem narzędzi dla mikrokontrolerów z rodziny STM32 o nazwie STM32 ODE (*STM32 Open Development Environment* – **rysunek 1**). Składają się na niego zarówno narzędzia sprzętowe, jak i programowe. Podstawę oferty narzędzi sprzętowych stanowią bardzo przystępne cenowo płytki z serii STM32 Nucleo. Wyposażenie każdej z płytek STM32 Nucleo obejmuje głównie mikrokontroler, programator/debuger ST-Link oraz gniazda rozszerzeniowe, w tym gniazda Arduino.

Uzupełnieniem oferty narzędzi sprzętowych są płytki rozszerzeniowe. Pojedyncza płytka może być wyposażona np. w czujniki, kontroler silnika, podzespoły komunikacyjne itp. Poprzez łączenie płytek STM32 Nucleo z płytkami rozszerzeniowymi można uzyskać rozbudowane platformy, sprofilowane pod kątem konkretnych aplikacji.

Narzędzia programowe noszą wspólną nazwę Cube i są oferowane bezpłatnie przez producenta. Najważniejszym ich elementem jest zestaw bibliotek HAL (*Hardware Abstraction Layer*), które stanowią wygodny interfejs programistyczny pozwalający na kontrolowanie peryferiów mikrokontrolera. Cube to również oprogramowanie middleware, w tym stosy protokołów, system operacyjny czasu rzeczywistego oraz system plików FAT. Istotną częścią narzędzi programowych jest CubeMX. Jest to program komputerowy, który na bazie wspomnianych bibliotek HAL i oprogramowania middleware generuje kod źródłowy dla mikrokontrolera wraz z projektem programistycznym(dla różnych środowisk: IAR




**Rysunek 1. Elementy składowe ekosystemu narzędzi STM32 Open Development Environment**

EWARM, Keil MDK-ARM, Atollic TRUEStudio lub AC6 System Workbench).

## Koncepcja tworzenia systemu z użyciem STM32 ODE i AC6 System Workbench

Tworzenie systemu w oparciu o narzędzia z ekosystemu STM32 ODE oraz przy wykorzystaniu środowiska programistycznego AC6 System Workbench można podzielić na kilka etapów (pokazano je na **rysunku 2**). Poniżej opisano w skrócie każdy z nich.

W pierwszym etapie programista używa programu CubeMX. Przy pomocy graficznego interfejsu użytkownika wybierany jest model mikrokontrolera, po czym konfigurowane są jego zasoby: wyprowadzenia, peryferie oraz sygnały zegarowe. Następnie CubeMX pozwala stworzyć projekt programistyczny z kodem źródłowym odpowiadającym wskazanej konfiguracji.

W drugim kroku wygenerowany przez CubeMX projekt należy zaimportować do środowiska programistycznego AC6 System Workbench. W środowisku tym programista wykonuje pracę polegającą na dodaniu kodu źródłowego implementującego zadania przewidziane do realizacji przez aplikację.

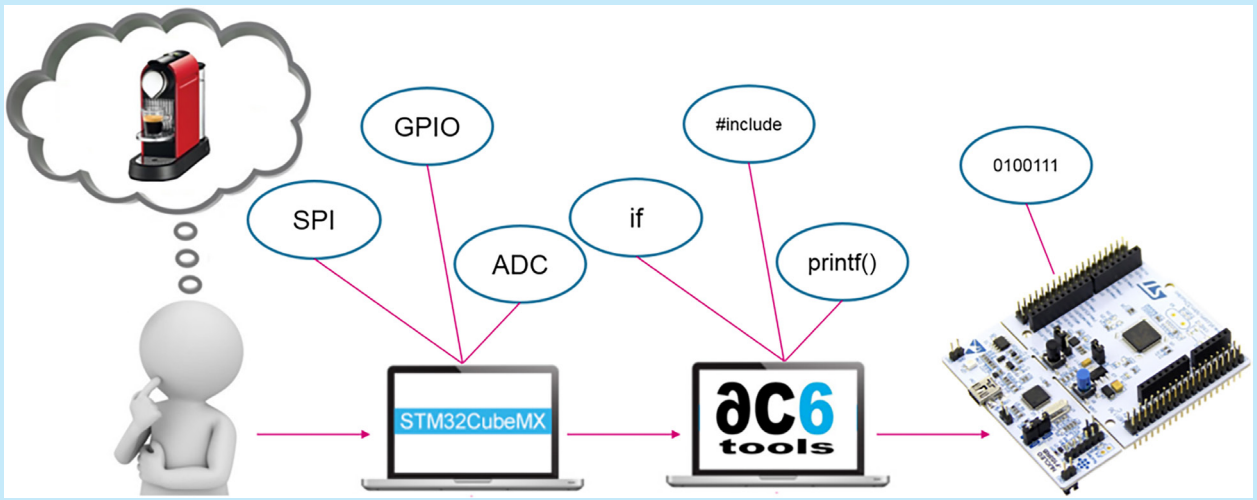
W trzeciej fazie kod źródłowy przekształcany jest na plik wykonywalny i wgrywany do pamięci mikrokontrolera. Po wykonaniu tej czynności aplikacja działa na platformie sprzętowej, zatem etap ten kończy pracę nad systemem.

Aby pokazać jak przedstawiony model funkcjonuje w praktyce, posłużono się przykładem. Jest nim prosty system będący interfejsem wejścia/wyjścia dla użytkownika (odczytywanie stanu przycisku oraz zmienianie stanu diody LED). W dalszej części artykułu opisano jak go zrealizować. Każdemu z trzech etapów tworzenia systemu poświęcono osobny rozdział. Jako platformę sprzętową wykorzystano płytkę NUCLEO-L476RG z mikrokontrolerem z rodziny STM32 L4. Narzędzia programowe to CubeMX w wersji 4.10 oraz AC6 System Workbench w wersji 1.3.

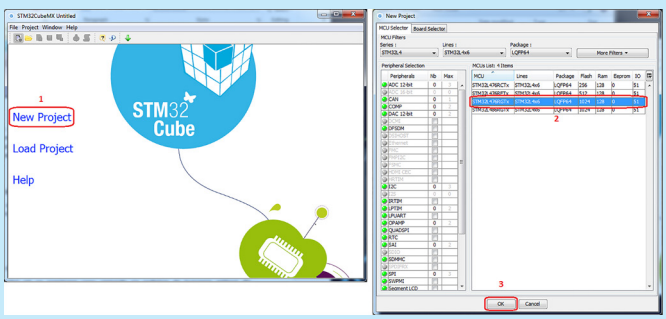
## Projekt w programie CubeMX

Pracę z narzędziem CubeMX programista rozpoczyna od stworzenia nowego projektu. W tym celu należy kliknąć na widoczny w oknie głównym programu napis *New Project* lub alternatywnie wybrać z menu głównego *File –> New Project...* Otworzone zostanie okno, które jest kreatorem nowego projektu. Należy w nim wskazać model mikrokontrolera, dla którego realizowana będzie aplikacja. Jako, że na płytce NUCLEO-L476RG zastosowany został układ STM32L476RGT6, to właśnie ten model musi zostać wybrany. Aby łatwiej go odnaleźć, korzystnie jest zawęzić listę mikrokontrolerów, która domyślnie zawiera pełny wykaz układów z rodziny STM32. Listę można skrócić poprzez zastosowanie filtrów, takich jak podgrupa rodziny STM32, linia w danej podgrupie, obudowa itp. Następnie należy zaakceptować wybór klikając przycisk *OK* (**rysunek 3**).

Po utworzeniu projektu program CubeMX udostępnia kilka zakładek, które pozwalają programiście na graficzną konfigurację mikrokontrolera. W pierwszej zakładce, która nosi nazwę *Pinout* (**rysunek 4**), wybierane są zasoby, z których układ STM32L476RGT6 będzie korzystał. Po lewej stronie okna programu, w postaci listy, widoczne są peryferie. Po prawej stronie okna programu, w formie obudowy układu, widoczne są piny. Na potrzeby przykładowej aplikacji należy




**Rysunek 2. Koncepcja tworzenia systemu przy użyciu narzędzi: STM32 ODE i AC6 System Workbench**




**Rysunek 3. Tworzenie nowego projektu w programie CubeMX**

skonfigurować dwa wyprowadzenia. Pierwsze z nich to PC13, które połączone jest z przyciskiem. Drugie natomiast to PA5, które połączone jest z diodą LED. Aby wykonać konfigurację należy najechać na schemat obudowy i kliknąć na element symbolizujący wyprowadzenie (kolejno PC13 i PA5), po czym z otworzonego w ten sposób menu kontekstowego należy wybrać tryb pracy wyprowadzenia (odpowiednio *GPIO_Input* oraz *GPIO_Output*).

Dalsze konfigurowanie mikrokontrolera może być realizowane przy użyciu zakładek *Clock Configuration* oraz *Configuration* (**rysunek 5**). Zakładka *Clock Configuration* pozwala na wybranie źródła sygnału zegarowego (HSI, MSI, LSI, HSE, LSE, PLL) i wskazanie

jego częstotliwości. Możliwe jest ponadto ustawienie innych elementów mechanizmu zegarowego mikrokontrolera (multipleksery, mnożniki, dzielniki), które decydują o częstotliwości sygnału taktującego poszczególne peryferie. Z kolei zakładka *Configuration* udostępnia bardziej zaawansowane opcje konfiguracyjne peryferiów, które zostały zaznaczone przez użytkownika w zakładce *Pinout*. W kontekście aplikacji przykładowej nie ma potrzeby dokonywania zmian w ustawieniach domyślnych zakładek *Clock Configuration* oraz *Configuration*.

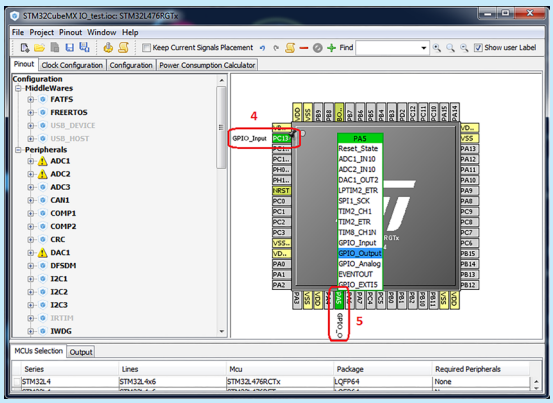
W tym momencie praca z programem CubeMX dobiega końca. Ostatnią czynnością związaną z tym narzędziem jest wygenerowanie kodu źródłowego wraz z projektem programistycznym. W tym celu należy wybrać z menu głównego programu *Project –> Generate Code* lub kliknąć na skojarzoną z tą operacją ikonę. W otworzonym w ten sposób oknie należy wpisać nazwę projektu (pole *Project Name*), wskazać ścieżkę na dysku twardym, gdzie kod ma zostać wygenerowany, wybrać środowisko programistyczne (pole *Toolchain/IDE*) i kliknąć przycisk *OK* (**rysunek 6**).

## Projekt w środowisku AC6 System Workbench

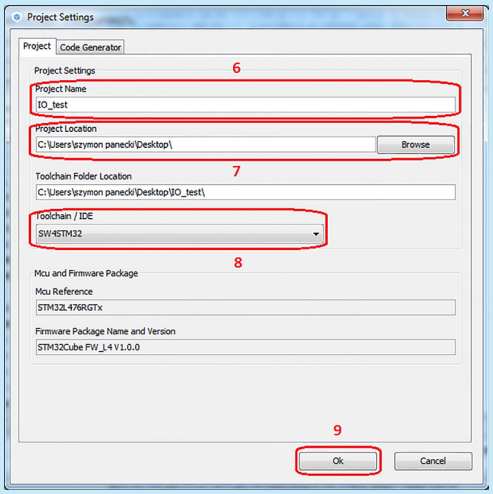
Środowisko AC6 System Workbench tuż po włączeniu wyświetla okno, w którym wskazać należy ścieżkę do przestrzeni roboczej. Może to być dowolnie wybrana przez programistę lokalizacja na dysku twardym. Korzystnie jest jednak wskazać ścieżkę, pod którą umieszczony został wygenerowany projekt. Aby przejść dalej, konieczne jest zaakceptowanie wyboru poprzez kliknięcie na przycisk *OK* (**rysunek 7**).

Nowo-stworzona przestrzeń robocza jest domyślnie pusta, dlatego za pierwszym razem konieczne jest zaimportowanie projektu do tejże przestrzeni. Narzędzie do importowania projektu programista wywołuje z menu głównego, przez wybranie *File –> Import...* W otworzonym w ten sposób oknie należy wskazać typ importowanego elementu. W tym wypadku należy wybrać z listy najpierw *General*, a potem *Existing Projects into Workspace*. Przejście dalej sygnalizowane jest kliknięciem przycisku *Next*. W drugim kroku należy wskazać ścieżkę na dysku twardym, pod którą znajduje się projekt. Proces importowania projektu kończy kliknięcie przycisku *Finish* (**rysunek 8**).W tym momencie pliki projektowe zostają wczytane do środowiska programistycznego i są udostępniane użytkownikowi w postaci drzewa.

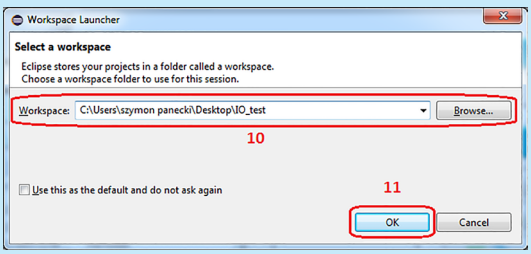
Dalsza praca nad aplikacją wymaga edycji pliku main.c. Aby otworzyć ten plik w edytorze, należy odnaleźć go w drzewie projektu (*Application –> User –> main.c*) i dwukrotnie na niego kliknąć. Kod źródłowy należy dodawać w sekcjach *USER CODE*, które nie podlegają skasowaniu po regeneracji kodu w programie CubeMX. Jako, że przewidziane w aplikacji przykładowej zadanie odczytywania stanu przycisku i zmiany stanu diody LED ma charakter cykliczny, kod źródłowy trzeba umieścić w sekcji *USER CODE 3*, gdyż znajduje się ona w nieskończonej pętli (*while (1)*). Do odczytu stanu przycisku użyć należy funkcji *HAL_GPIO_ReadPin()*, natomiast zmianę stanu diody LED można uzyskać przez wykorzystanie funkcji *HAL_GPIO_WritePin()*. Całość dodanego do pliku main.c kodu źródłowego pokazano na **rysunku 9**.



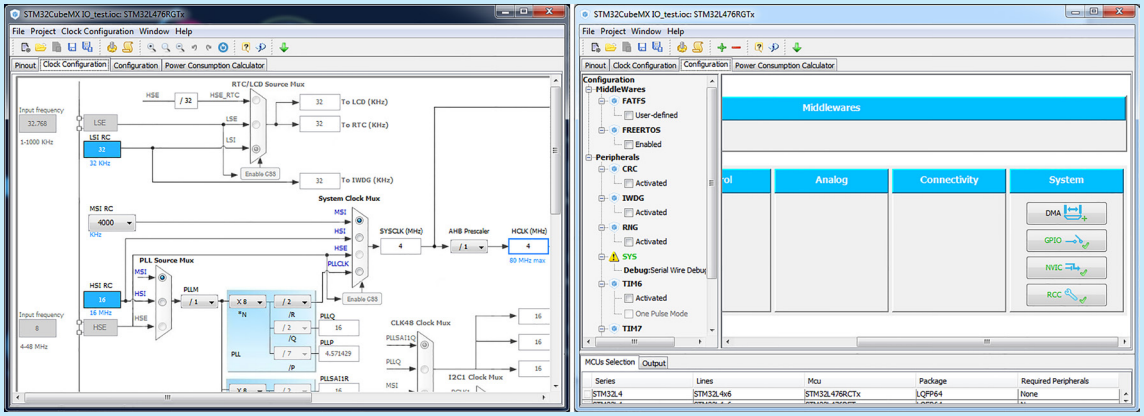

**Rysunek 4. Konfiguracja wyprowadzeń mikrokontrolera w programie CubeMX**



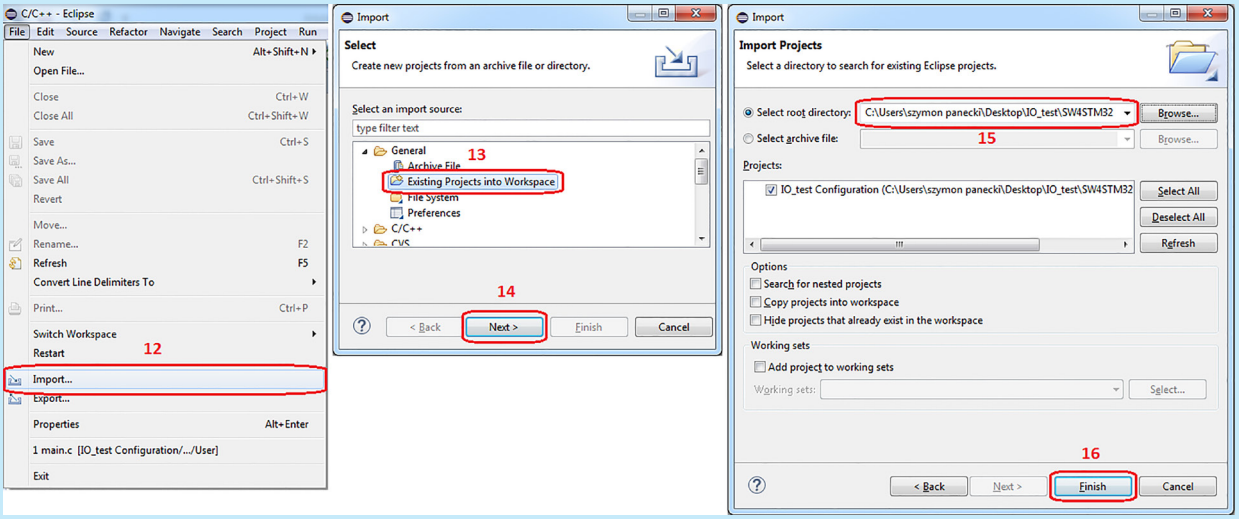

**Rysunek 6. Wygenerowanie kodu źródłowego wraz z projektem programistyznym w programie CubeMX**




**Rysunek 7. Wybór przestrzeni roboczej w środowisku AC6 System Workbench**




**Rysunek 5. Konfiguracja sygnałów zegarowych oraz peryferiów mikrokontrolera w programie CubeMX**

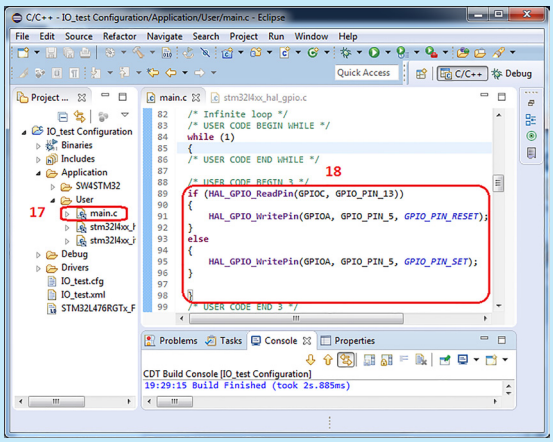



**Rysunek 8. Importowanie projektu wygenerowanego w CubeMX do środowiska AC6 System Workbench**

## Uruchamianie aplikacji na platformie sprzętowej

Po zakończeniu pracy nad kodem źródłowym programista przekształca go do postaci pliku wykonywalnego. W tym celu należy z menu głównego wybrać *Project –> Build All*. Jeśli operacja ta zakończy się sukcesem, w oknie *Console* wyświetlony zostanie komunikat *Build Finished*. Aby zaprogramować mikrokontroler i uruchomić aplikację na mikrokontrolerze, należy wywołać narzędzie debugera. Aby to zrobić, należy z menu głównego należy wybrać *Run –> Debug* lub kliknąć ikonę skojarzoną z tą czynnością. Przy pierwszej próbie debugowania środowisko AC6 System Workbench otworzy okno konfiguracji debugera. Dwa pola muszą zostać w nim uzupełnione ręcznie: *Debug device* oraz *Debug interface*. Jej poprawne wartości to odpowiednio *ST-LinkV2-1* oraz *SWD*. Zmiany zaakceptować należy klikając przycisk *OK* (**rysunek 10**).

W tym momencie środowisko AC6 System Workbench nawiąże komunikację z programatorem/debugerem ST-Link i plik wykonywalny zostanie wgrany do pamięci mikrokontrolera STM32L476RGT6. Jednocześnie programista uzyska dostęp do narzędzi i okien debugowania. Najbardziej podstawową czynnością, jaką można wykonać, jest uruchomienie aplikacji w trybie ciągłym. Programista realizuję ją przez wybranie z menu głównego *Run –> Resume* lub kliknięcie ikony (**rysunek 11**). Działanie aplikacji można przetestować na płytce NUCLEO-L476RG. Efektem powinno być zaświecenie lub zgaszenie diody LED w warunkach naciśniętego lub zwolnionego przycisku.

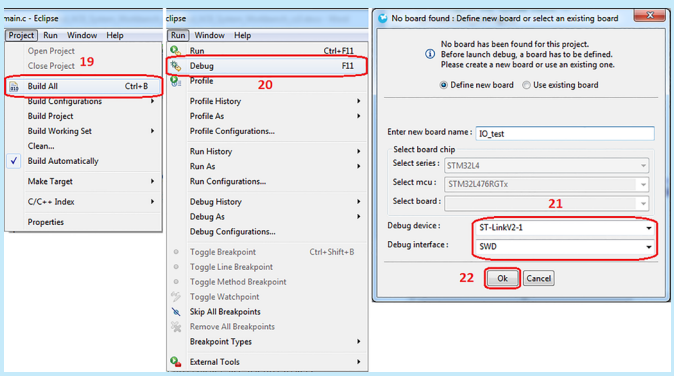



**Rysunek 9. Plik main.c z kodem źródłowym dodanym w celu odczytu przycisku i sterowania diodą LED**

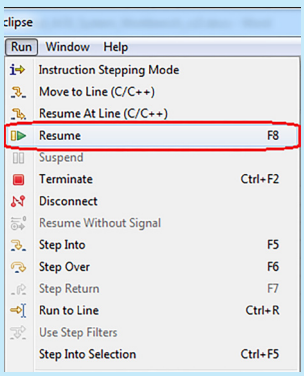
## Podsumowanie

Możliwość współpracy środowiska AC6 System Workbench z STM32 ODE daje duże korzyści osobom, które zdecydują się używać tych narzędzi do tworzenia systemów opartych na mikrokontrolerach z rodziny STM32. Po pierwsze, dlatego, że użytkownicy otrzymują do dyspozycji zestaw niedrogich narzędzi projektowych (środowisko AC6 System Workbench, jak też biblioteki HAL i program CubeMX są udostępniane bezpłatnie, płytki STM32 Nucleo są bardzo przystępne cenowo). Po drugie, dlatego, że zestaw wymienionych narzędzi tworzy spójną całość, gdyż rozwiązania te pozwalają na przejście od pomysłu, poprzez implementację, do działającego systemu włącznie. Po trzecie, warto zwrócić uwagę, że użycie tych narzędzi ułatwia tworzenia aplikacji (CubeMX pozwala na graficzne generowanie kodu, AC6 System Workbench może importować projekty generowane przez CubeMX, Nucleo jest natomiast gotową do użycia platformą sprzętową), dzięki czemu czas pracy nad systemem znaczącą się skraca.

**Szymon Panecki, EP**




**Rysunek 10. Stworzenie pliku wykonywalnego oraz uruchomienie debugera w środowisku AC6 System Workbench**




**Rysunek 11. Uruchomienie aplikacji w trybie działania ciągłego w środowisku AC6 System Workbench**