Java i STM32 – idealna para

Obecnie nikogo nie powinny już dziwić serwery webowe i aplikacje wyposażone w bogate, graficzne interfejsy użytkownika działające nawet na najprostszych i najtańszych mikroprocesorach. W przenoszeniu aplikacji między platformami pomagają coraz liczniejsze systemy operacyjne, a współczesne kompilatory pozwoliły na zastąpienie asemblera językiem C, a nawet językami obiektowymi, jak Java czy C++. Właśnie w tym kierunku podążyła firma IS2T dostarczając Butterfly OS, czyli zestawu narzędzi umożliwiających połączenie języków Java i C w mikrokontrolerach rodziny STM32.

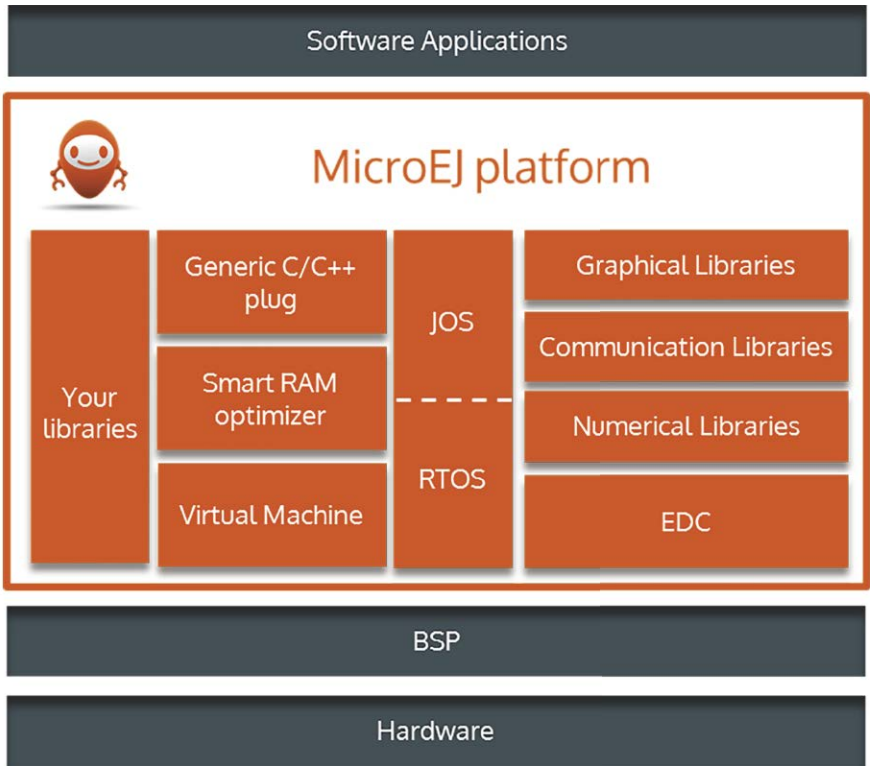
Dodatkowe informacje:
Dystrybutorem środowiska MicroEJ w Polsce jest firma Masters (*http://www.masters.com.pl/*)

Świat mikrokontrolerów ewoluuje nieustannie. Zmiany dotyczą nie tylko sprzętu, ale też programowania i dostępnych narzędzi. Mikrokontrolery 32-bitowe z rdzeniami Cortex znajdują się już w ofercie wszystkich czołowych producentów oraz dystrybutorów i skutecznie konkurują z mikrokontrolerami 8-bitowymi pod względem ceny i możliwości. Programiści systemów wbudowanych mają coraz więcej narzędzi zarówno płatnych, jak i darmowych, dzięki którym ich praca staje się bardziej efektywna, a oprogramowanie może powstawać znacznie szybciej.

Środowisko programistyczne

Butterfly OS zapewnia szereg narzędzi programistycznych oraz bibliotek ułatwiających tworzenie złożonych aplikacji dla systemów wbudowanych opartych na mikrokontrolerach rodziny STM32. Wśród nich wymienić warto biblioteki do komunikacji (m.in. Wi-Fi, Bluetooth, USB, itp.), biblioteki i narzędzia wspomagające tworzenie zaawansowanych interfejsów użytkownika opartych o wyświetlacze graficzne, panele dotykowe i przyciski, a także biblioteki matematyczne wspomagające obliczenia.

Aplikacje podzielone są na dwie warstwy. Pierwsza z nich napisana jest w języku C i obejmuje sterowniki peryferiów i urządzeń zewnętrznych, obsługę przerwań oraz opcjonalny system czasu rzeczywistego (obsługiwane są między innymi FreeRTOS, RTX, µc/OS). Wyższa warstwa obejmuje wspomniane biblioteki i główną aplikację napisaną w Javie, działającą na maszynie wirtualnej stworzonej przez firmę IS2T.



Rysunek 1. Schemat platformy MicroEJ

Embedded Java

Do uruchomienia aplikacji napisanej w Javie niezbędna jest maszyna wirtualna (JVM). Stanowi ona środowisko uruchomieniowe, w którym działają aplikacje kompilowane do odpowiedniego kodu bajtowego. Dzięki takiemu rozwiązaniu aplikacje napisane w tym języku mogą być uruchamiane na każdym urządzeniu z zainstalowaną maszyną wirtualną. Niestety, wymagania sprzętowe oficjalnej JVM dostarczanej przez Oracle przekraczają możliwości prostych mikrokontrolerów. Programiści z firmy IS2T znaleźli na to sposób implementując własną maszynę wirtualną Javy.

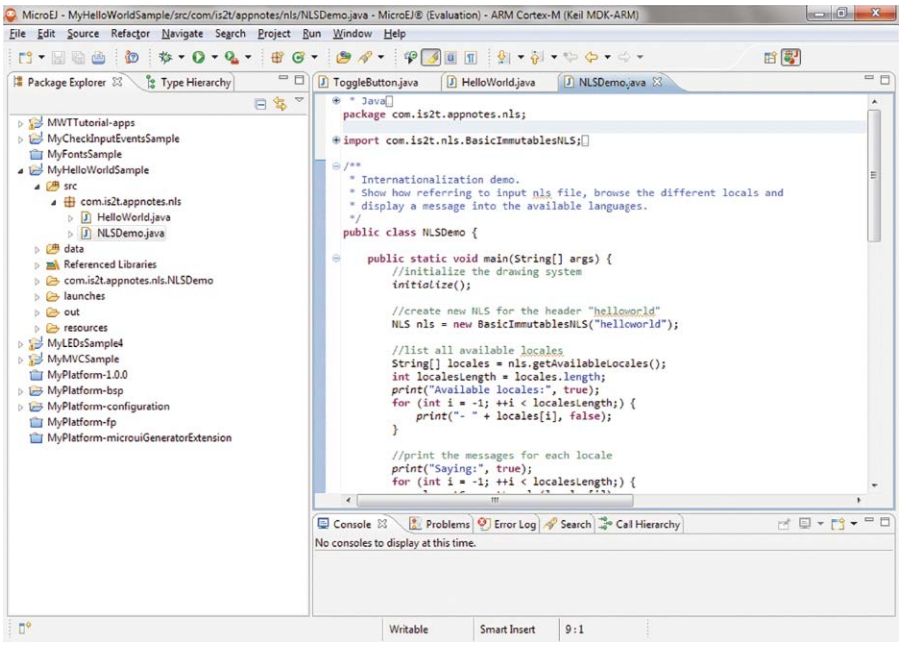
Platforma dostarczana przez IS2T wymaga zaledwie 28 kB pamięci Flash i 1 kB pamięci RAM. Zawiera maszynę wirtualną, biblioteki standardowe, matematyczne, graficzne oraz komunikacyjne. Schemat platformy pokazano na **rysunku 1**.

Ogromną zaletą platformy MicroEJ jest jej dostępność dla wielu różnych architektur, systemów operacyjnych (w tym Linuksa, Androida i kilku systemów czasu rzeczywistego) i kompilatorów, zarówno płatnych (Keil, IAR), jak i darmowego GCC. Dzięki temu raz napisane aplikacje mogą być uruchamiane na wielu urządzeniach, co gwarantuje pełną elastyczność w tworzeniu oprogramowania i zapewnia komfort pracy programistom systemów embedded zbliżony do pracy nad aplikacjami wysokopoziomowymi. Pełną listę wspieranych urządzeń i systemów znaleźć można na stronie producenta *http://goo.gl/uIcDM4* (**rysunek 2**). W chwili pisania artykułu, platforma była już także dostępna dla mikrokontrolerów z rdzeniami Cortex-M0 i Cortex-M0+.

Pisząc nowe aplikacje można wykorzystać stary kod napisany w Javie. Implementacja MicroEJ jest zgodna z J2SE i zawiera podstawowe biblioteki: *java.io*, *java.lang*, *java.util*, *java.lang.annotation*, *java.lang.ref* oraz *java.lang.reflect*, natomiast kod bajtowy generowany jest przy użyciu standardowego kompilatora Javy. Oczywistym ograniczeniem są zasoby użytego mikrokontrolera, przez co nie każda aplikacja przygotowana dla komputerów

	MCU	MPU	Mobile
Cores	ARM7, ARM9, Cortex-M3/M4, V850, RX, MIPS32/64	ARM9, Cortex-A, MIPS x86, Ikanos	ARM
RTOS	µc/OS, ThreadX, FreeRTOS, RTX, Osek, Keil OS, no RTOS	Linux 2.4+, Integrity, VxWorks	Android-3.2+, iOS-6.0+
RTOS Integration	GreenThread	GreenThread, NativeThread	NativeThread
C compilers	Keil, GCC, IAR	GCC, GreenHills	GCC, LLVM

Rysunek 2: Architektury wspierane przez MicroEJ (źródło *http://goo.gl/WmcRLM*)



Rysunek 3: Okno główne środowiska MicroEJ

PC będzie mogła być uruchomiona na platformie MicroEJ.

Pomimo tego, że programowanie aplikacji odbywa się głównie w języku Java, nic nie stoi na przeszkodzie, aby włączyć do projektu wcześniej napisany kod C. W języku tym są także napisane niskopoziomowe sterowniki urządzeń i kod specyficzny dla mikrokontrolera. Dołączanie kodu C jest możliwe dzięki bibliotece SNI (*Simple Native Interface*), która jest odpowiednikiem JNI (*Java Native Interface*) dostępnej w standardowej maszynie wirtualnej Java znanej z komputerów PC. Mogłoby się wydawać, że aplikacje napisane w C będą działać znacznie szybciej, jednakże producent MicroEJ zapewnia, że wydajność programów napisanych w obu językach jest porównywalna. Jako przykład można podać aplikację graficzną z odświeżaniem 50 fps na wyświetlaczu 480×272 z 32-bitową głębią kolorów wykorzystującą tylko 20% mocy obliczeniowej mikrokontrolera STM32F429NIH6. Więcej przykładów można zobaczyć na stronie IS2T (*http://www.is2t.com/videos/*). Dodatkowo, aplikacje napisane w Javie mogą korzystać z udogodnień znanych z programowania wysokopoziomowego, takich jak dynamiczna alokacja pamięci, automatyczne zwalnianie zasobów (*garbage collection*) i pełne wsparcie języka dla programowania obiektowego. Programiści tworzący aplikacje z graficznym interfejsem użytkownika z pewnością docenią możliwości biblioteki graficznej z obsługą ekranów, paneli dotykowych, a także czcionek i tłumaczeń.

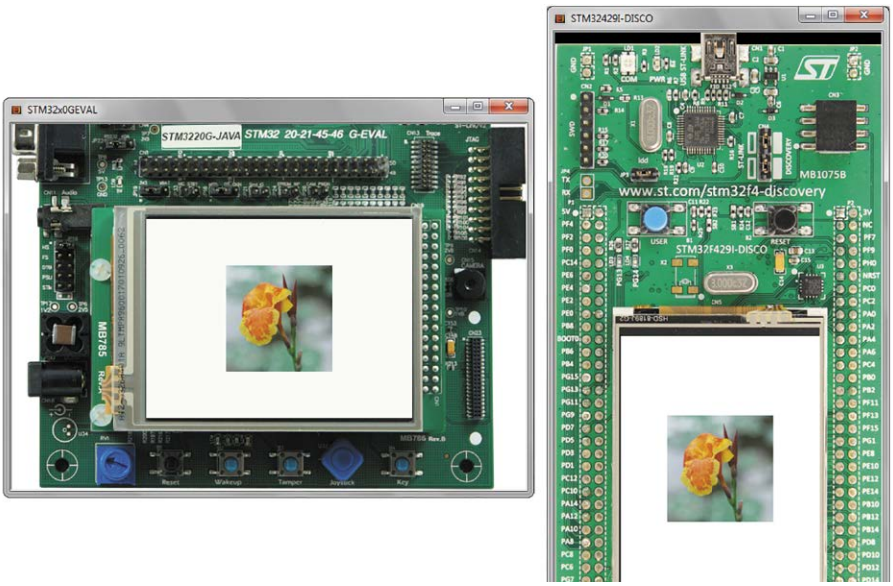
Środowisko programistyczne

IS2T oferuje środowisko programistyczne dla aplikacji pisanych na platformę MicroEJ (o tej samej nazwie). Jest ono zbudowane na popularnym i darmowym IDE – Eclipse (**rysunek 3**).

Środowisko to umożliwia pisanie i kompilację kodu Javy oraz przygotowanie platformy MicroEJ. Niestety na chwilę obecną nie jest możliwa kompilacja kodu C. Do tego celu należy użyć zewnętrznego kompilatora, niedostarczanego przez IS2T. Na szczęście jest możliwe wygenerowanie plików projektu dla kilku środowisk (m.in. Keil, Atollic), dzięki czemu pracę z kodem C można rozpocząć bez konieczności konfigurowania kolejnego projektu. Wsparcie dla GCC umożliwia także pracę z otwartymi toolchainami (np. GNU Tools for ARM + OpenOCD). Istotną informacją może być także dostępność MicroEJ zarówno dla systemu Linux, jak i Windows. Informacje o instalacji dla poszczególnych systemów znaleźć można na stronie IS2T.

Niewątpliwie jedną z najciekawszych cech środowiska MicroEJ jest możliwość symulacji kodu Javy na PC bez użycia dodatkowego sprzętu (**rysunek 4**). Symulacje można przeprowadzać dla kilku dostępnych zestawów ewaluacyjnych (także z serii STM32 Discovery), a także definiowanych przez siebie układów zawierających wyświetlacze graficzne, przyciski, diody LED oraz inne peryferia. Pozwala to na testowanie i debugowanie aplikacji bez konieczności programowania mikrokontrolera.

Warto również wspomnieć o przygotowanych platformach MicroEJ z opcją KickStart. Umożliwiają tworzenie aplikacji w Javie i ich uruchamianie na wybranych

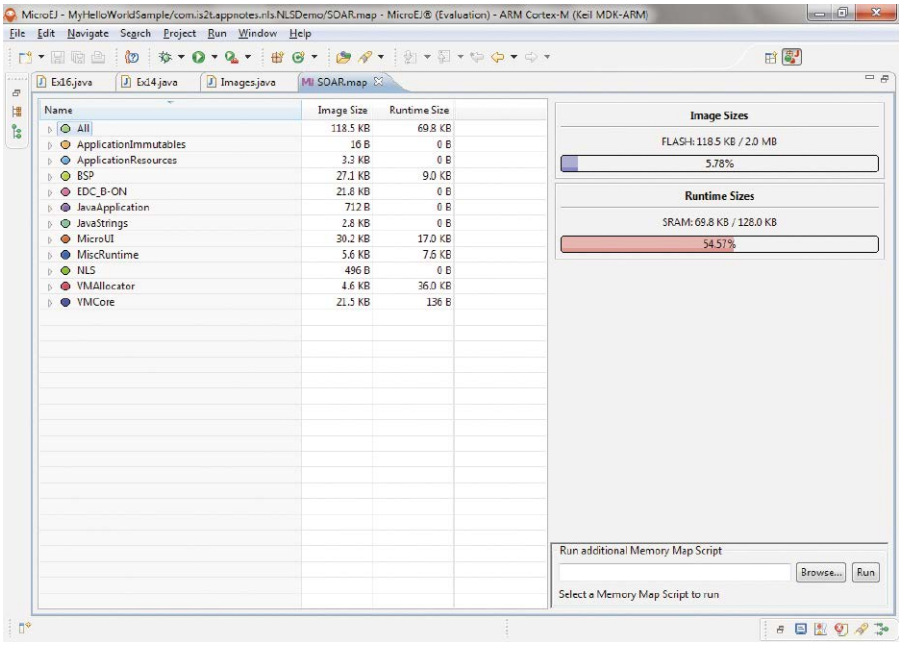


Rysunek 4: Możliwość symulacji aplikacji na różnych zestawach ewaluacyjnych

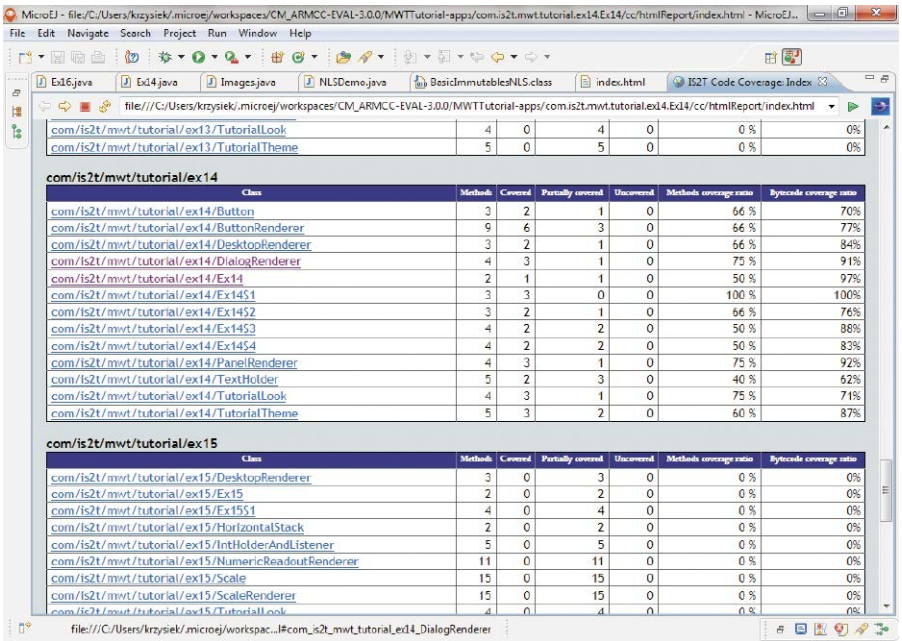
zestawach ewaluacyjnych bez potrzeby użycia kompilatora C i dodatkowego oprogramowania, poza środowiskiem MicroEJ. W tym przypadku używany jest kompilator Javy i linker stworzony przez IS2T, dzięki któremu możliwe jest uruchomienie aplikacji na maszynie wirtualnej.

Środowisko MicroEJ zawiera także narzędzia, które mogą być szczególnie pomocne przy testowaniu aplikacji. Pierwszym z nich jest *Memory Map Analyzer*, dzięki któremu można sprawdzić wykorzystanie pamięci RAM oraz Flash przez poszczególne komponenty platformy MicroEJ, aplikacje użytkownika oraz wykorzystane w nich obiekty (**rysunek 5**).

Kolejnym narzędziem jest *Stack Trace Descriptor* przydatny przy odczytywaniu komunikatów błędów zwracanych przez wyjątki w maszynie wirtualnej. Trzecim narzędziem jest *Code Coverage Analyzer*, który umożliwia sprawdzenie częstości wykonywania poszczególnych kawałków kodu. Wynikiem analizy jest raport HTML pozwalający zapoznać się z wnioskami (**rysunek 6**). *Heap Dumper* z kolei pozwala na zapisywanie chwilowego stanu sterty, w której znajdują się zaalokowane obiekty. Narzędzie to pomaga w wyszukiwaniu wycieków pamięci, przeglądaniu stanu obiektów i optymalizowaniu wykorzystania sterty aplikacji. Ostatnim z opisanych narzędzi jest *Test Suite*. Umożliwia ono sprawdzanie poprawności aplikacji udostępniając mechanizmy przeprowadzania testów jednostkowych i generując raporty do analizy błędów. Dalsze plany obejmują także kreator GUI dla osób, które nie programują na co dzień w Javie, a zajmują się graficzną stroną aplikacji.



Rysunek 5: Memory Map Analizer



Rysunek 6 Code Coverage Analyzer

Podsumowanie

Możliwość pisania aplikacji na mikrokontrolery w Javie niesie ze sobą z pewnością wiele korzyści płynących zarówno z samego języka, jak i dostępnych narzędzi. Java jest językiem czysto obiektowym, a przy tym stosunkowo prostym. Warto więc przyswoić sobie jego podstawy, aby czerpać korzyści z jego użycia, zwłaszcza w aplikacjach graficznych. Dostępność licznych bibliotek w środowisku MicroEJ znacznie przyspiesza produkcję oprogramowania i udostępnia wiele narzędzi wspomagających testowanie i debugowanie aplikacji. Niezaprzeczalną korzyścią jest możliwość tworzenia oprogramowania niezależnego od sprzętu, które dodatkowo można testować w symulatorze.

Niestety, środowisko oferowane przez firmę IS2T wymaga użycia dodatkowego oprogramowania do kompilowania kodu napisanego w C. Konieczne jest więc posiadanie kompilatora C – płatnego (Keil, IAR) lub darmowego (GCC). W pierwszym wypadku podnosi to koszty projektu, w drugim wymaga pracy związanej z konfigurowaniem środowiska. Na szczęście, jak zapewniają twórcy MicroEJ, niebawem pojawi się wersja środowiska z kompilatorem C, dzięki czemu cały projekt będzie można wykonać w obrębie jednego IDE.

Póki co środowisko MicroEJ nie wspiera debugowania aplikacji na platformie docelowej. Symulator daje ogromne możliwości i wygodę testowania, ale nie ma możliwość sprawdzenia, jak kod zachowuje się w warunkach rzeczywistych. Autorzy MicroEJ zapewniają jednak, że środowisko w przyszłości będzie wspierało standard JTAG, co umożliwi swobodne debugowanie kodu także na mikrokontrolerach.

Na zakończenie warto wspomnieć, z jakimi kosztami trzeba się liczyć, aby użyć MicroEJ w swoim projekcie. Cena rocznej licencji środowiska to ok. 2000 EUR dla mikrokontrolerów produkowanych przez ST. W cenę wliczono wsparcie techniczne ze strony tej firmy. W przypadku licencji na środowisko ze wsparciem dla wszystkich mikrokontrolerów wymienionych na stronie IS2T, cena wynosi ok. 4500 EUR. Wsparcie techniczne zapewnia wówczas firma IS2T. Dodatkowo, należy liczyć się z kosztami użycia platformy MicroEJ na każdym urządzeniu. Dla mikrokontrolerów produkowanych przez ST opłata ta wynosi ok. 15% ceny mikrokontrolera i jest już wliczona w cenę urządzeń z oznaczeniami J oraz U. Ich dokładną listę można znaleźć na stronie producenta (*http://goo.gl/U0WV7E*).

Krzysztof Chojnowski