Od Autora

Do mikrokontrolerów wszyscy już się przyzwyczailiśmy – obecnie są one nieodłączną częścią każdego, nawet najprostszego, urządzenia elektronicznego. Ze względu na ich możliwości, atrakcyjną cenę i łatwość stosowania trudno sobie bez nich wyobrazić zbudowanie nawet najprostszego urządzenia. Pomimo że od długiego czasu na rynku komputerów osobistych są wykorzystywane rozwiązania 32- oraz 64-bitowe, w świecie mikrokontrolerów do tej pory królowały mikrokontrolery 8-bitowe, które w większości zastosowań okazywały się wystarczające. Jednak od pewnego czasu jesteśmy świadkami ARM-owej „rewolucji", zapoczątkowanej przez firmę NXP wprowadzeniem na rynek 32-bitowych mikrokontrolerów z rodziny LPC2000. Dziś już nikogo nie dziwi wydajny mikrokontroler z 32-bitowym rdzeniem ARM, który kosztuje tyle samo co jego 8-bitowy starszy brat. Obecnie coraz więcej firm zamiast ponosić olbrzymie koszty na opracowanie i spopularyzowanie własnych jednostek centralnych mikrokontrolerów, produkuje układy z rdzeniem zakupionym od firmy ARM. Dzięki temu teraz mikrokontrolery z rdzeniem ARM są już standardem, tak jak to dawniej miało miejsce w przypadku mikrokontrolerów z rodziny 8051. Rozwiązanie takie jest korzystne zarówno dla producenta, ponieważ nie musi on tworzyć od podstaw wszystkich narzędzi dla własnej rodziny (na przykład kompilatorów C), jak i dla użytkownika, uwalniając go od zakupu oddzielnych narzędzi dla każdego typu mikrokontrolera.

W przypadku narzędzi programowych do mikrokontrolerów również można mówić o rewolucji. Kilkanaście lat temu dobry jakościowo kompilator języka C kosztował gigantyczne kwoty, które były nie do zaakceptowania przez małe firmy, nie mówiąc już o hobbystach. Obecnie, dzięki idei otwartego oprogramowania (*Open Source*) oraz dużej pamięci programu współczesnych mikrokontrolerów, możemy bez ponoszenia dodatkowych kosztów wykorzystywać kompilatory C/C++ (GCC). Narzędzia te stały się na tyle dobre, że jeśli chodzi o ARM-y producenci w większości przypadków nie zadają sobie już trudu, aby napisać własny kompilator, tylko wykorzystują GCC, opakowując go w atrakcyjne środowisko graficzne IDE oraz dodając trochę narzędzi autorskich (na przykład do obsługi JTAG-a) i sprzedają je następnie za duże pieniądze. Jeżeli jesteśmy w stanie zrezygnować z kilku „wodotrysków", to zupełnie za darmo możemy zbudować sobie środowisko do programowania mikrokontrolerów.

Dopiero co zdążyliśmy się przyzwyczaić do dostępności mikrokontrolerów z rdzeniem ARM7TDMI-S, a producenci zaczęli umieszczać w mikrokontrolerach wydajniejsze rdzenie ARM9, które do tej pory były zarezerwowane tylko dla dużych maszyn. Tak więc elektronicy mają do dyspozycji coraz większą moc obliczeniową. Oprócz niezaprzeczalnych zalet rodzi to też istotne obawy, które już od dłuższego czasu można zaobserwować w przypadku oprogramowania do komputerów PC – mianowicie programiści, mając do dyspozycji coraz potężniejsze „maszyny", coraz mniej myślą o optymalizacji swoich programów, bo wydajny komputer mający gigabajty pamięci i procesor o częstotliwości taktowania mierzonej w GHz i tak sobie świetnie poradzi, więc po co się męczyć nad optymalnymi algorytmami? Czyżby to samo czekało nas w przypadku oprogramowania dla mikrokontrolerów? Wydaje

się, że niestety tak i proces ten właśnie teraz odbywa się na naszych oczach, ponieważ coraz wydajniejsze mikrokontrolery zachęcają nas do takich poczynań.

Bohaterem niniejszej książki jest właśnie taki wydajny mikrokontroler produkowany przez firmę STMicroelectronics. STR912 jest wyposażony w rdzeń ARM966E-S pracujący z częstotliwością do 96 MHz, co umożliwia uruchamianie bardzo wydajnych i ciekawych aplikacji. Dodatkowym atutem, jak przystało na mikrokontroler z rdzeniem ARM9, są nowoczesne układy peryferyjne, z których szczególnej uwagi wymagają interfejsy USB Device oraz Ethernet. Dziś trudno sobie wyobrazić nawet proste urządzenie, pozbawione możliwości dołączenia do sieci czy do PC-ta za pomocą nowoczesnego interfejsu USB. Pisząc tę książkę, starałem się odejść od klasycznej koncepcji skrupulatnego opisywania wszystkich peryferii bit po bicie, rejestr po rejestrze, a skupić się na jak największej liczbie przykładów i konkretnych rozwiązań układowych.

Książka ta jest przeznaczona dla Czytelników mających już pewne doświadczenie w technice mikroprocesorowej oraz umiejętność programowania chociażby dowolnej rodziny mikrokontrolerów (na przykład 8-bitowych AVR). Książka składa się z 5 rozdziałów przechodzących od aspektów teoretycznych do coraz bardziej zaawansowanych przykładów praktycznych. W pierwszym rozdziale przedstawiłem opis budowy architektury rdzenia ARM966E-S, nawiązując przy tym do architektury innych mikrokontrolerów 8-bitowych oraz rdzenia ARM7TDMI-S. W rozdziale drugim opisałem budowę wewnętrznych układów peryferyjnych mikrokontrolera ze szczególnym uwzględnieniem mechanizmów pozwalających na przyspieszenie wykonywania programu z pamięci Flash mikrokontrolera. W kolejnym rozdziale zapoznaję Czytelnika z wybraną platformą sprzętową (ZL24ARM – moduł z mikrokontrolerem STR912 i ZL25ARM – płyta bazowa), na której będą uruchamiane przykłady zawarte w książce oraz przygotujemy środowisko programistyczne umożliwiające kompilowanie i debugowanie programów z wykorzystaniem środowiska *Eclipse* IDE. Postaram się również przy tej okazji nie zaniedbać użytkowników systemu Linux, wspominając, jak przygotować w tym systemie środowisko o identycznej funkcjonalności. W kolejnych rozdziałach przechodzę już do praktycznych zagadnień, pokazując, w jaki sposób można okiełznać poszczególne układy peryferyjne mikrokontrolera na przykładzie bardzo konkretnych problemów spotykanych w praktyce. W ostatnim rozdziale, niejako na deser, podaję najciekawsze zagadnienia związane z siecią Ethernet oraz interfejsem USB. Wbrew pozorom nie będzie to trudny temat, gdyż zostaną wykorzystane biblioteki dostarczane przez producenta mikrokontrolerów ST91x.

Na wszelkie uwagi i opinie związane z niniejszą książką czekam pod adresem:

lucjan.bryndza@ep.com.pl