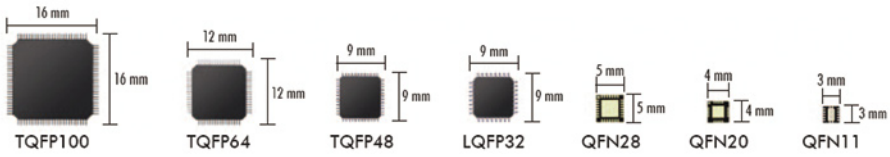
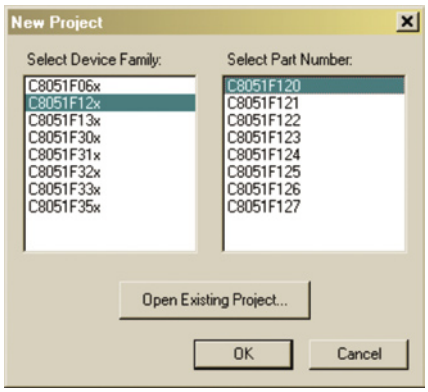
Configuration Wizard

Wszyscy kibice piłki kopanej zapewne wiedzą, co to są stałe fragmenty gry. Występują one zresztą nie tylko w tej dyscyplinie, ale są najczęściej z nią właśnie utożsamiane. Język sportowy często przenika do mowy potocznej, zdarza się więc, że tak też nazywamy powtarzające się, rutynowe czynności życia codziennego.

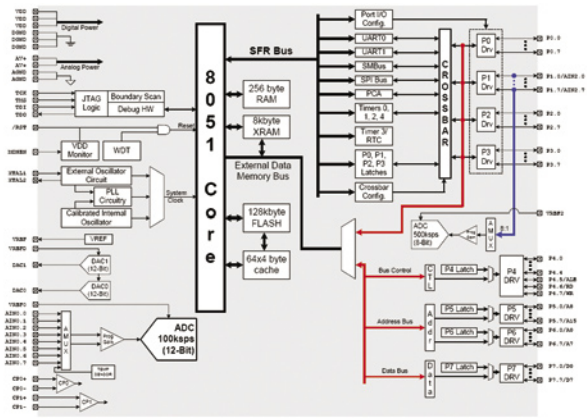


Rys. 1. Obudowy mikrokontrolerów C8051Fxxx

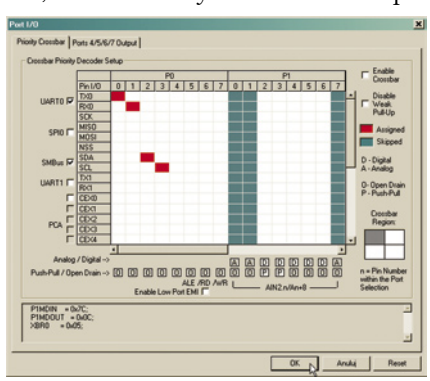


Rys. 3. Okno wyboru mikrokontrolera programu Configuration Wizard

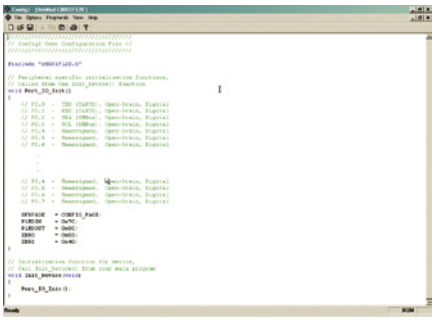
Każdy program informatyczny, czy to dla dużego superkomputera, czy malutkiego mikroprocesora zawiera pewien fragment, którego celem jest zainicjowanie zmiennych systemowych oraz wszelkich urządzeń peryferyjnych istniejących w systemie. Dotyczy to zarówno wewnętrznych bloków funkcjonalnych procesora, jak i urządzeń zewnętrznych. Od indywidualnych przyzwyczajeń programisty zależy, w jaki sposób zadanie to zrealizuje praktycznie. Dobrym zwyczajem wydaje się być zawarcie wszystkich, a przynajmniej większości instrukcji inicjujących w osobnych podprogramach (jednym lub kilku, np. wydzielonych dla określonych typów peryferiali). Taka metoda, przynajmniej teoretycznie, powinna ułatwiać tworzenie aplikacji, wystarczy wszak raz przygotować odpowiednią procedurę, zapisać ją w odrębnym pliku, a później tylko stosowną dyrektywą dołączać do projektu. Doceniając „powagę" problemu, niektórzy producenci narzędzi dla programistów oferują nie tylko tak potężne pomoce, jak środowiska IDE, ale i całkiem małe, ale bardzo przydatne *gadgety* programistyczne. Przykładem może być *Configuration Wizard* firmy Silicon Laboratories. Jest to prosty programik, służący do automatycznego generowania kodu inicjalizującego dla mikrokontrolerów F00x, F01x, F02x, F04x (wersja 1) oraz F2xx F06x, F12x, F13x, F30x, F31x, F32x, F33x, F35x (wersja 2 oprogramowania). Są to raczej mało znane rodzimym konstruktorom układy, a szkoda, bo są niezwykle interesujące. Fakt, że ich rdzeń bazuje na prehistorycznej już – powiedzieliby szydercy – 51––ce, w konfrontacji z uzyskiwanymi osiągami i cechami użytkowymi, nie powinien mieć najmniejszego znaczenia. Mikrokontrolery C8051Fxxx należą do jednych z najszybszych układów z tym rdzeniem. W ofercie można znaleźć grupy układów specjalizowanych do określonych zadań lub charakteryzujące się zbliżonymi parametrami. Większość z nich wyposażono w jeden lub dwa wielokanałowe przetworniki analogowo – cyfrowe oraz przetworniki cyfrowo – analogowe. Są więc one predestynowane do precyzyjnego przetwarzania sygnałów. Wyróżniono grupę mikrokontrolerów wyposażonych w interfejs USB 2.0, interfejs CAN, a także układy w miniaturowych i subminiaturowych obudowach (**rys. 1**) oraz mikrokontrolery ogólnego zastosowania. Zestawienie parametrów przedstawiono w **tab. 1**. Mimo 8–bitowego rdzenia, osiągane moce obliczeniowe od 20 do 100 MIPS muszą robić wrażenie. Także 128 MB pamięci Flash „na pokładzie" świadczy, że układy te nadają się do tworzenia poważnych aplikacji. Potwierdzają to również niestety niemałe ceny, co w takich przypadkach nie zawsze stanowi problem. Na uwagę zasługuje bardzo dobre wsparcie techniczne ze strony producenta. Do każdego typu mikrokontrolera można dobrać odpowiedni zestaw ewaluacyjny zawierający płytkę prototypową, adapter umożliwiający programowanie układów, środowisko IDE oraz zestaw narzędziowy *Configuration Wizard*. Poprzez Internet dostępne są również bardzo obszerne noty katalogowe i aplikacyjne. Można tą drogą dokonywać zakupów wszystkich produktów firmy Silicon Laboratories, a zwolennicy kontaktów bezpo-



Rys. 2. Schemat blokowy rdzenia mikrokontrolera C8051F120



Rys. 4. Okno konfiguracji portów we/wy

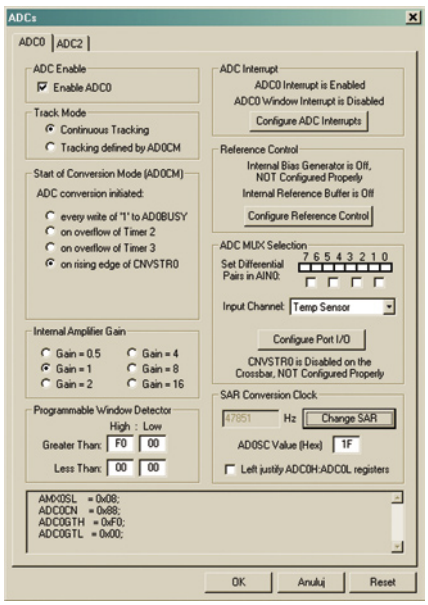


Rys. 5. Okno kodu programu otrzymanego w wyniku czynności konfiguracyjnych

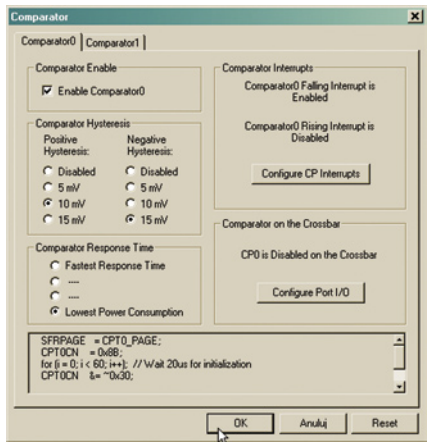
średnich mogą skorzystać z pomocy polskich dystrybutorów.

Bogactwo peryferiali

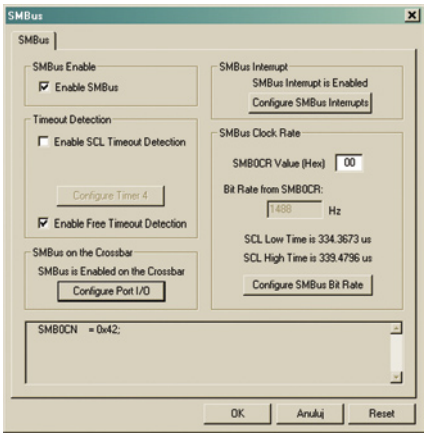
Na **rys. 2**. przedstawiono schemat blokowy jednego z najmocniejszych mikrokontrolerów firmy Silicon Laboratories – C8051F120. Jak widać, w ekstremalnym przypadku konstruktor będzie miał do obsługi sporo wewnętrznych bloków funkcjonalnych – pamięć zewnętrzną, interfejsy: *I²C*, *SPI*, dwa *UART*–y, a także 5 timerów, *PCA* (*Programable Counter Array*), 64 porty I/O, dwa przetworniki *ADC*, przetwornik *DAC*, komparator analogowy. Zważywszy, że peryferiale te dysponują różnymi trybami pracy, już samo ich zainicjowanie może być dość kłopotliwe. Pomocą będzie w tym na pewno program *Configuration Wizard*. Program ten jest dostępny np. na stronie *http://www.silabs.com/tgwWebApp/appmanager/tgw/tgwHome?_nfpb=true&_pageLabel=GenericContentPage&contentObjectId=/public/web_content/products/Microcontrollers/en/mcu_configwiz.htm* bez żadnych opłat. Pobranie go, w przeciwieństwie do większości innych pomocy internetowych firmy Silicon Laboratories, nie wymaga żadnej rejestracji. Instalacja przebiega w typowy sposób. Zaraz po jej zakończeniu, narzędzie jest gotowe do użycia. Na dysku są zapisywane dwie wersje oprogramowania (jeśli zostanie to zaznaczone podczas instalacji). Każda z nich obsługuje inną grupę mikrokontrolerów, warto więc zainstalować zarówno *Configuration Wizard*, jak i *Configuration Wizard 2*. Pierwszą czynnością po uruchomieniu programu jest określenie typu mikrokontrolera, dla którego tworzony będzie kod inicjujący (**rys. 3**). *Configuration Wizard* stanowi wsparcie dla programów pisanych w asemblerze oraz w języku C, wygodne więc będzie ustawienie odpowiedniej opcji już na samym początku pracy. Podejmowanie tej decyzji nie jest jednak krytyczne i bez obawy o utratę wykonanej pracy można to będzie zmienić w dowolnym momencie. Następnie przystępujemy już do zasadniczej pracy, która z punktu widzenia operacji manualnych będzie dość prosta, natomiast ze względu na liczne możliwości konfiguracyjne obsługiwanych mikrokontrolerów na pewno będzie wymagała posiadania wiadomości na ich temat. Tworzenie kodu inicjującego będzie polegało na kolejnym wybieraniu używanych w aplikacji peryferiali (z menu *Peripherals*), a następnie zaznaczaniu odpowiednich opcji, tudzież określaniu wartości liczbowych niektórych parametrów. Poszczególne okna konfiguracyjne są zaprojektowane bardzo intuicyjne, często nie będzie nawet potrzebne odwoływanie się do not katalogowych. W gruncie rzeczy po to właśnie powstało takie narzędzie, jak *Configuration Wizard*. Przykład konfiguracji portów we/wy mikrokontrolera C8051F120 pokazano na **rys. 4**. W tym przypadku zostały zadeklarowane interfejsy *UART0* i *SMBus* (*I²C*). Porty P1.0, P1.1 i P1.7 jako wejścia przetwornika A/C. Porty P1.2 i P1.3 pracują jako cyfrowe typu *Push – Pull*, natomiast P1.4... P1.6 jako cyfrowe typu *Open – Drain*. W dolnej części ekranu widoczne jest okno, w którym są wyświetlane przetłumaczone na wybrany język rozkazy mikrokontrolera odpowiadające zaznaczonym opcjom. Dla doświadczonego programisty będą stanowiły zapewne potwierdzenie poprawności przeprowadzonych działań. Po zaakceptowaniu zmian, w oknie głównym, pojawi się kompletny fragment funkcji inicjujących (**rys. 5**). Na **rys. 6...9** pokazano okna konfiguracji kilku innych przykładowych peryferiali mikrokontrolera C8051F120.



Rys. 6. Okno konfiguracji przetworników A/C

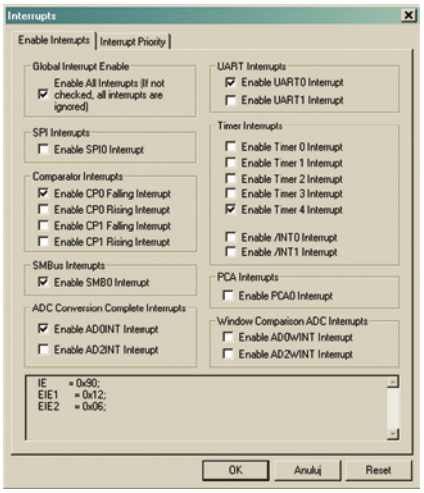


Rys. 7. Okno konfiguracji komparatorów



Rys. 8. Okno konfiguracji interfejsu SMBus (I²C)

Okno prezentacji kodu (rys. 5), choć do złudzenia przypomina typowe okno edytorów tekstowych nie-



Rys. 9. Okno konfiguracji systemu przerwań

Tab. 1. Zestawienie parametrów wybranych mikrokontrolerów C8051Fxxx firmy Silicon Laboratories

Typ	Pamięć Flash [kB]	Pamięć RAM [B]	Liczba linii I/O	Interfejsy szeregowe	Timery (16-bit)	Liczba kanałów PCA	A/C1	A/C2	C/A	Wbudowany czujnik temperatury	Źródło napięcia referencyjnego	Komparatory	Inne	Obudowa
C8051F060	64	4352	59	CAN2.0B, 2 UART, SMBus, SPI	5	6	16-bit, 2 kan.	10-bit, 8 kan.	12-bit, 2 kan.	+	+	3	DMA	TQFP100
C8051F061	64	4352	24	CAN2.0B, 2 UART, SMBus, SPI	5	6	16-bit, 2 kan.	10-bit, 8 kan.	12-bit, 2 kan.	+	+	3	DMA	TQFP64
C8051F062	64	4352	59	CAN2.0B, 2 UART, SMBus, SPI	5	6	16-bit, 2 kan.	10-bit, 8 kan.	12-bit, 2 kan.	+	+	3	DMA	TQFP100
C8051F063	64	4352	24	CAN2.0B, 2 UART, SMBus, SPI	5	6	16-bit, 2 kan.	10-bit, 8 kan.	12-bit, 2 kan.	+	+	3	DMA	TQFP64
C8051F120	128	8448	64	2 UART, SMBus, SPI	5	6	12-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	16x16 MAC	TQFP100
C8051F123	128	8448	32	2 UART, SMBus, SPI	5	6	10-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	16x16 MAC	TQFP64
C8051F124	128	8448	64	2 UART, SMBus, SPI	5	6	12-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	-	TQFP100
C8051F125	128	8448	32	2 UART, SMBus, SPI	5	6	12-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	-	TQFP64
C8051F126	128	8448	64	2 UART, SMBus, SPI	5	6	10-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	-	TQFP100
C8051F127	128	8448	32	2 UART, SMBus, SPI	5	6	10-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	-	TQFP64
C8051F064	64	4352	59	2 UART, SMBus, SPI	5	6	16-bit, 2 kan.	-	-	-	+	3	DMA	TQFP100
C8051F065	64	4352	24	2 UART, SMBus, SPI	5	6	16-bit, 2 kan.	-	-	-	+	3	DMA	TQFP64
C8051F040	64	4352	64	CAN2.0B, 2 UART, SMBus, SPI	5	6	12-bit, 13 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	3	±60V PGA	TQFP100
C8051F041	64	4352	32	CAN2.0B, 2 UART, SMBus, SPI	5	6	12-bit, 13 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	3	±60V PGA	TQFP64
C8051F042	64	4352	64	CAN2.0B, 2 UART, SMBus, SPI	5	6	10-bit, 13 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	3	±60V PGA	TQFP100
C8051F043	64	4352	32	CAN2.0B, 2 UART, SMBus, SPI	5	6	10-bit, 13 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	3	±60V PGA	TQFP64
C8051F020	64	4352	64	2 UART, SMBus, SPI	5	5	12-bit, 8 kan.	8-bit, 8 kan.	12-bit, 2 kan.	+	+	2	-	TQFP100
C8051F000	32	256	32	UART, SMBus, SPI	4	5	12-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP64
C8051F001	32	256	16	UART, SMBus, SPI	4	5	12-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP48
C8051F002	32	256	8	UART, SMBus, SPI	4	5	12-bit, 4 kan.	-	12-bit, 2 kan.	+	+	1	-	LQFP32
C8051F005	32	2304	32	UART, SMBus, SPI	4	5	12-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP64
C8051F006	32	2304	16	UART, SMBus, SPI	4	5	12-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP48
C8051F007	32	2304	8	UART, SMBus, SPI	4	5	12-bit, 4 kan.	-	12-bit, 2 kan.	+	+	1	-	LQFP32
C8051F010	32	256	32	UART, SMBus, SPI	4	5	10-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP64
C8051F011	32	256	16	UART, SMBus, SPI	4	5	10-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP48
C8051F012	32	256	8	UART, SMBus, SPI	4	5	10-bit, 4 kan.	-	12-bit, 2 kan.	+	+	1	-	LQFP32
C8051F015	32	2304	32	UART, SMBus, SPI	4	5	10-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP64
C8051F016	32	2304	16	UART, SMBus, SPI	4	5	10-bit, 8 kan.	-	12-bit, 2 kan.	+	+	2	-	TQFP48
C8051F017	32	2304	8	UART, SMBus, SPI	4	5	10-bit, 4 kan.	-	12-bit, 2 kan.	+	+	1	-	LQFP32
C8051F018	16	1280	32	UART, SMBus, SPI	4	5	10-bit, 8 kan.	-	-	+	+	2	-	TQFP64
C8051F019	16	1280	16	UART, SMBus, SPI	4	5	10-bit, 8 kan.	-	-	+	+	2	-	TQFP48
C8051F044	64	4352	64	CAN2.0B, 2 UART, SMBus, SPI	5	6	10-bit, 13 kan.	-	-	+	+	3	±60V PGA	TQFP100
C8051F045	64	4352	32	CAN2.0B, 2 UART, SMBus, SPI	5	6	10-bit, 13 kan.	-	-	+	+	3	±60V PGA	TQFP64
C8051F046	32	4352	64	CAN2.0B, 2 UART, SMBus, SPI	5	6	10-bit, 13 kan.	-	-	+	+	3	±60V PGA	TQFP100
C8051F047	32	4352	32	CAN2.0B, 2 UART, SMBus, SPI	5	6	10-bit, 13 kan.	-	-	+	+	3	±60V PGA	TQFP64
C8051F350	8	768	17	UART, SMBus, SPI	4	3	24-bit, 8 kan.	-	8-bit, 2 kan.	+	-	1	-	LQFP32
C8051F351	8	768	17	UART, SMBus, SPI	4	3	24-bit, 8 kan.	-	8-bit, 2 kan.	+	-	1	-	MLP28

Typ	Pamięć Flash [kB]	Pamięć RAM [B]	Liczba linii I/O	Interfejsy szeregowe	Timery (16-bit)	Liczba kanałów PCA	A/C1	A/C2	C/A	Wbudowany czujnik temperatury	Źródło napięcia referencyjnego	Komparatory	Inne	Obudowa
C8051F352	8	768	17	UART, SMBus, SPI	4	3	16-bit, 8 kan.	-	8-bit, 2 kan.	+	-	1	-	LQFP32
C8051F353	8	768	17	UART, SMBus, SPI	4	3	16-bit, 8 kan.	-	8-bit, 2 kan.	+	-	1	-	MLP28
C8051F320	16	2304	25	USB 2.0, UART, SMBus, SPI	4	5	10-bit, 17 kan.	-	-	+	+	2	-	LQFP32
C8051F321	16	2304	21	USB 2.0, UART, SMBus, SPI	4	5	10-bit, 13 kan.	-	-	+	+	2	-	MLP28
C8051F206	8	1280	32	UART, SPI	3	-	12-bit, 32 kan.	-	-	-	-	2	-	TQFP48
C8051F220	8	256	32	UART, SPI	3	-	8-bit, 32 kan.	-	-	-	-	2	-	TQFP48
C8051F221	8	256	22	UART, SPI	3	-	8-bit, 32 kan.	-	-	-	-	2	-	LQFP32
C8051F226	8	1280	32	UART, SPI	3	-	8-bit, 32 kan.	-	-	-	-	2	-	TQFP48
C8051F230	8	256	32	UART, SPI	3	-	-	-	-	-	-	2	-	TQFP48
C8051F231	8	256	22	UART, SPI	3	-	-	-	-	-	-	2	-	LQFP32
C8051F236	8	1280	32	UART, SPI	3	-	-	-	-	-	-	2	-	TQFP48
C8051F310	16	1280	29	UART, SMBus, SPI	4	5	10-bit, 21 kan.	-	-	+	-	2	-	LQFP32
C8051F311	16	1280	25	UART, SMBus, SPI	4	5	10-bit, 17 kan.	-	-	+	-	2	-	MLP28
C8051F312	8	1280	29	UART, SMBus, SPI	4	5	10-bit, 21 kan.	-	-	+	-	2	-	LQFP32
C8051F313	8	1280	25	UART, SMBus, SPI	4	5	10-bit, 17 kan.	-	-	+	-	2	-	MLP28
C8051F314	8	1280	29	UART, SMBus, SPI	4	5	-	-	-	+	-	2	-	LQFP32
C8051F315	8	1280	25	UART, SMBus, SPI	4	5	-	-	-	+	-	2	-	MLP28
C8051F330	8	768	17	UART, SMBus, SPI	4	3	10-bit, 16 kan.	-	10-bit, 1 kan.	+	+	1	-	MLP20
C8051F330D	8	768	17	UART, SMBus, SPI	4	3	10-bit, 16 kan.	-	10-bit, 1 kan.	+	+	1	-	PDIP20
C8051F331	8	768	17	UART, SMBus, SPI	4	3	-	-	-	-	-	1	-	MLP20
C8051F332	4	768	17	UART, SMBus, SPI	4	3	10-bit, 16 kan.	-	-	+	+	1	-	MLP20
C8051F333	4	768	17	UART, SMBus, SPI	4	3	-	-	-	-	-	1	-	MLP20
C8051F334	2	768	17	UART, SMBus, SPI	4	3	10-bit, 16 kan.	-	-	+	+	1	-	MLP20
C8051F335	2	768	17	UART, SMBus, SPI	4	3	-	-	-	-	-	1	-	MLP20
C8051F300	8	256	8	UART, SMBus	3	3	8-bit, 8 kan.	-	-	+	-	1	-	MLP11
C8051F301	8	256	8	UART, SMBus	3	3	-	-	-	-	-	1	-	MLP11
C8051F302	8	256	8	UART, SMBus	3	3	8-bit, 8 kan.	-	-	+	-	1	-	MLP11
C8051F303	8	256	8	UART, SMBus	3	3	-	-	-	-	-	1	-	MLP11
C8051F304	4	256	8	UART, SMBus	3	3	-	-	-	-	-	1	-	MLP11
C8051F305	2	256	8	UART, SMBus	3	3	-	-	-	-	-	1	-	MLP11

stety nie daje możliwości ręcznego modyfikowania programu. Ewentualne poprawki należy wykonywać poprzez ponowne wywołanie odpowiedniej opcji z menu *Peripherals*. Wprowadzone nowe ustawienia spowodują naczytanie poprzednich rezultatów. W razie konieczności można w desperackim geście wybrać opcję *Reset All* i ponownie przystąpić od początku do pracy. Dobrym zwyczajem będzie zapisywanie projektów w wewnętrznym formacie *Configuration Wizarda* (pliki z rozszerzeniem CWG). Pozwoli to na ewentualną późniejszą modyfikację kodu w tym programie, jako, że nie potrafi on zaimportować danych tekstowych. Z kolei do dalszego wykorzystania otrzymanego z *Wizarda* kodu np. w środowisku IDE lub dowolnym edytorze, konieczny będzie eksport rezultatu pracy w formacie tekstowym. Uzyskujemy to przez wybranie polecenia *File–>Save Source As...* Dane są zapisywane jako plik tekstowy z rozszerzeniem C lub ASM, zależnie od ustawienia opcji *Options–>Code Format*.

Podsumowanie

Programy takie, jak *Configuration Wizard* stanowią istotną pomoc szczególnie dla konstruktorów wykorzystujących wiele różnych typów mikrokontrolerów. Choć większość z nich bazuje na podobnych mechanizmach, to jednak różnią się między sobą nierzadko drobnymi, ale istotnymi szczegółami. Zapamiętanie wszystkich niuansów jest dość trudne i chyba nawet niepotrzebne. Zawsze można oprzeć się na nocie katalogowej, albo skorzystać z takich dobrodziejstw, jak *Configuration Wizard*, koncentrując cały intelekt na optymalizacji stosowanych algorytmów i zwalczaniu problemów hardware'owych. Procedury inicjalizacyjne są przecież tylko „stałym fragmentem gry" w każdej aplikacji mikroprocesorowej. Wiadomo jednak, że ze stałych fragmentów gry padają gole. Nieprawidłowo zainicjowany peryferial nie będzie pracował zgodnie z oczekiwaniami i wykrycie tego faktu może zabrać niekiedy sporo czasu konstruktorowi.

Jarosław Doliński, EP
jaroslaw.dolinski@ep.com.pl