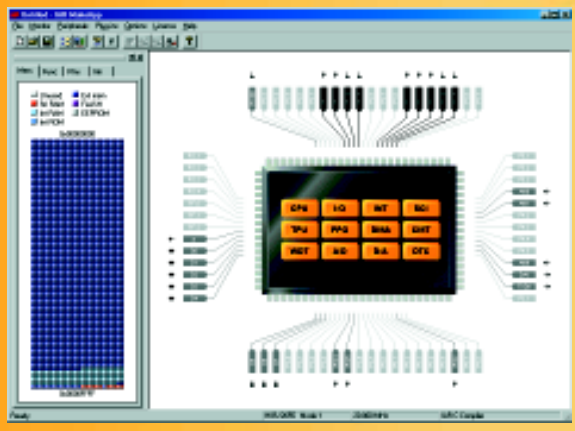
# IAR Make App™

***Coraz większa skala integracji, coraz więcej tranzystorów, coraz większa moc obliczeniowa, coraz więcej wbudowanych w mikrokontrolery układów - tak przez ostatnie lata wygląda rozwój mikrokomputerów jednoukładowych. Z jednej strony umożliwia to znaczne zmniejszenie wymiarów urządzeń, gdyż pojedynczy układ scalony może realizować bardzo różnorodne funkcje, z drugiej jednak projektant musi poświęcić bardzo dużo czasu na poznanie możliwości tak rozbudowanego układu. Często dochodzi do dziwnych sytuacji, gdy obniżenie kosztów produkcji związane z zastosowaniem nowoczesnego układu nie powoduje obniżenia ceny końcowej urządzenia, gdyż producent musi uwzględnić w kalkulacji produktu koszt opracowania urządzenia (a czas pracy konstruktora nie jest tani). W niektórych sytuacjach z pomocą może przyjść aplikacja MakeApp firmy IAR, znanej chyba każdemu elektronikowi z produkcji wysokiej jakości kompilatorów języka C dla wielu rodzin mikroprocesorów.***

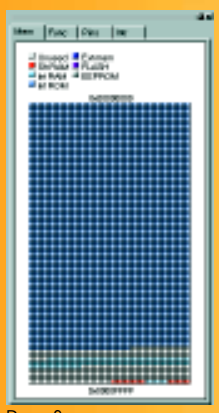
Większość elektroników zaakceptowała fakt wykorzystywania języków wysokiego poziomu w fazie tworzenia oprogramowania dla mikrokontrolerów stosowanych w projektowanych urządzeniach. Oczywiście, nie da się dobrze oprogramować mikrokontrolera bez znajomości asemblera i architektury stosowanego procesora, ale języki wysokiego poziomu umożliwiają przełamanie barier psychicznych, często występujących na początkowym etapie przygody z programowaniem układów mikroprocesorowych. Poznanie asemblera i dalsze poznawanie tajników działania procesorów przychodzi z czasem, gdy wzrasta liczba projektów zrealizowanych z wykorzystaniem mikroprocesorów. Zwykle po zrealizowaniu kilku, kilkunastu projektów z poznanym procesorem projektant dochodzi do wniosku, że w niektórych sytuacjach zastosowanie innego typu procesora byłoby, delikatnie mówiąc, zręczniejsze. Dzięki wykorzystaniu języka wysokiego poziomu możliwe jest niemal bezbolesne przeniesienie programu do innego typu procesora. Niemal bezbolesne, gdyż każdy program realizuje pewien algorytm w oparciu o dane lub sygnały wejściowe. Najczęściej zastosowany algorytm nie jest w sposób krytyczny powiązany z typem procesora (w projekcie zegara można wykorzystać ten sam algorytm obliczania roku przestępnego niezależnie od tego czy zastosujemy prosty procesor 8-bitowy, czy szybki, 32-bitowy procesor RISC). Natomiast zwykle okazuje się, że pewne kłopoty są związane z funkcjami wejścia/wyjścia oraz wykorzystaniem takich samych układów peryferyjnych wbudowanych w różne typy mikrokontrolerów. Wyobraźmy sobie następującą sytuację: na zlecenie piszemy program do czytnika kart magnetycznych, który połączony z komputerem pracuje w ramach systemu kontroli czasu pracy. Zadanie można podzielić na dwa etapy: odczytanie karty magnetycznej oraz przesłanie odczytanych i zaszyfrowanych (dla bezpieczeństwa) danych do komputera. Realizacja zlecenia przebiega bez zakłóceń do momentu, gdy oba moduły testujemy oddzielnie. Niestety w trakcie testów niemal gotowego urządzenia okazuje się, że czas potrzebny na zaszyfrowanie danych jest tak długi, że odczytanie następnej karty jest możliwe dopiero po 20 sekundach. Jest to sytuacja nie do przyjęcia w dużym zakładzie pracy, a algorytm szyfrowania został dostarczony przez zleceniodawcę, więc nie można go zmienić. W grę nie wchodzi rozbudowa urządzenia o dodatkową pamięć (do kolejkowania odczytanych danych) ani dołożenie drugiego procesora, gdyż w obudowie nie ma na to miejsca. Po rozpaczliwym poszukiwaniu wyjścia z tej sytuacji okazuje się, że najlepszym rozwiązaniem jest zastosowanie innego, silniejszego procesora, ale zbliża się termin oddania pracy, a zastosowany w projekcie procesor nie ma silniejszego (kompatybilnego) brata. Wówczas, przy konieczności zmiany procesora (z innej rodziny), programista realizujący to zlecenie w asemblerze popełni samobójstwo, w lepszej sytuacji będzie ten, kto program pisał w języku wysokiego poziomu, a we względnie komfortowej sytuacji będzie użytkownik *MakeApp*.

Rys. 1.

*MakeApp* jest narzędziem programistycznym zdejmującym z programisty konieczność samodzielnego tworzenia funkcji obsługi układów peryferyjnych wbudowanych w mikrokomputery jednoukładowe. Dodatkowo aplikacja kontroluje poprawność wybranych ustawień dla tych układów, dzięki czemu programista uniknie sytuacji wykorzystania tych samych zasobów do obsługi różnych zadań. *MakeApp* generuje kod źródłowy w języku C, przygotowany dla kompilatora IAR lub kompilatora GNU C. Aplikacja może wygenerować kod z funkcjami bibliotecznymi dla najpopularniejszych procesorów firm: Atmel, Hitachi, Mitsubishi oraz Toshiba. Niestety, *MakeApp* jest przygotowywany oddzielnie dla każdej z wymienionych rodzin procesorów. Oznacza to, że chcąc zapewnić przenośność programów między różnymi rodzinami mikroprocesorów należy kupić kilka licencji programu. Wersja demonstracyjna generuje kod

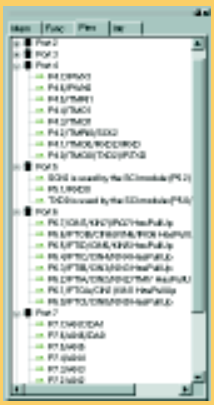


Rys. 2.



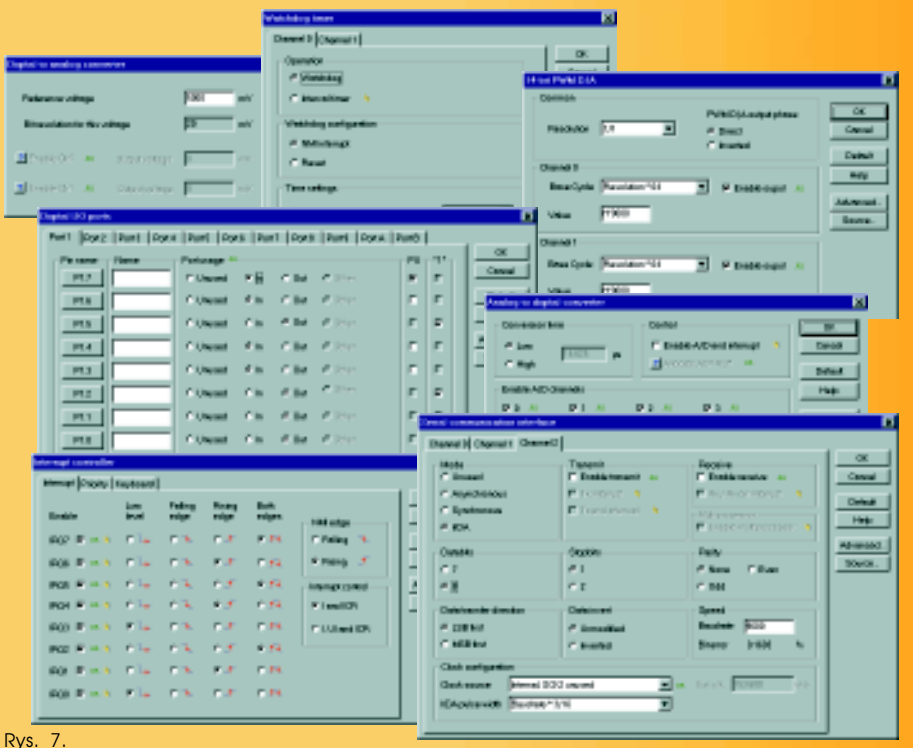
Rys. 3.

Rys. 4.



Rys. 5.

Rys. 6.



Rys. 7.

źródłowy dla wirtualnego (nieistniejącego) procesora o nazwie „Viking“, którego układy peryferyjne są na tyle rozbudowane, że z łatwością można ocenić przydatność aplikacji.

Po uruchomieniu *MakeApp* należy wybrać typ układu docelowego (**rys. 1**). Okno aplikacji jest podzielone na dwie części (**rys. 2**). Z lewej strony znajdują się cztery zakładki:

- w zakładce *Mem* (**rys. 3**) jest wyświetlany aktualny stan pamięci mikroprocesora;
- w zakładce *Func* (**rys. 4**) są wyświetlane nazwy funkcji, które zostaną wygenerowane dla aktualnie wybranych ustawień układów peryferyjnych mikroprocesora;
- w zakładce *Pins* (**rys. 5**) są wyświetlane zadania przypisane pinom mikroprocesora dla wybranych ustawień układów peryferyjnych;
- w zakładce *Intr* (**rys. 6**) są pokazane uaktywnione źródła przerwań.

Prawa strona okna aplikacji zawiera graficzną reprezentację mikroprocesora z symbolami układów peryferyjnych (rys. 2). Oczywiście liczba i rodzaje dostępnych układów peryferyjnych zależą od typu wybranego mikroprocesora. W wersji demonstracyjnej „procesor“ „Viking 1“ udostępnia między innymi układ watchdoga, przetworniki A/C i C/A, interfejs do komunikacji szeregowej, układ PWM. Niewykorzystywane przez funkcje biblioteczne piny mikroprocesora są wyświetlane w kolorze jasnoszarym, natomiast piny wykorzystywane przez jakąkolwiek z funkcji są wyświetlane w kolorze ciemnoszarym. Przy każdym wykorzystywanym pinie znajduje się strzałka pokazująca kierunek przepływu danych. Dodatkowo, po najechaniu myszą na symbol układu peryferyjnego, piny związane z tym układem zmieniają kolor na czarny. Kliknięcie na symbolu układu peryferyjnego powoduje wyświetlenie okna konfiguracyjnego danego układu (**rys. 7**). Zmiana parametrów w oknie konfiguracyjnym może powodować uaktywnianie poszczególnych układów oraz zmieniać ich sposób pracy. Dla każdego układu peryferyjnego można sprawdzić wykorzystywane zasoby, czyli pamięć, rejestry, przerwania itp. Próba wykorzystania zasobów (np. timera lub portu mikroprocesora) zajętych już przez inny układ peryferyjny będzie uniemożliwiona i zasygnalizowana odpowiednim komunikatem (**rys. 8**). Po ustawieniu konfiguracji można wygenerować kod z funkcjami bibliotecznymi dla wybranych układów peryferyjnych (**rys. 9**). Kod źródłowy wygenerowany dla procesora „Viking 1“ zawierał około 3000 linii programu.

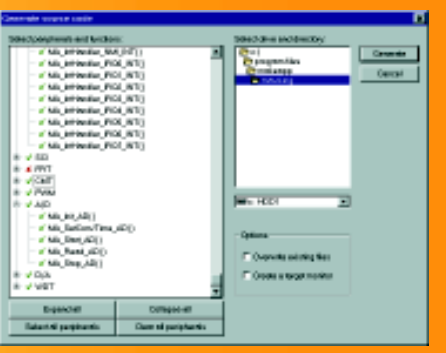
Rys. 8.

Korzystanie z *MakeApp* nie jest konieczne. Każdą z funkcji generowanych przez aplikację średnio doświadczony programista może napisać samodzielnie. Przy większym doświadczeniu programista może również utworzyć odpowiednie biblioteki dla wielu rodzin procesorów. Niewątpliwą zaletą takiego rozwiązania jest doświadczenie zdobyte w trakcie eksperymentów (bo przecież większość nowych funkcji bibliotecznych należy sprawdzić w działaniu), ale każdy powinien odpowiedzieć sobie na pytanie czy nie lepiej w tym czasie realizować kolejny projekt, korzystając z funkcji opracowanych i przetestowanych przez profesjonalistów. Wbudowana w *MakeApp* kontrola poprawności zadanych ustawień dla układów peryferyjnych z pewnością zostanie doceniona w trakcie realizacji bardziej skomplikowanych projektów.

**Paweł Zbysiński**

*Wersja demonstracyjna IAR MakeApp jest dostępna w Internecie pod adresem http://www.iar.com/FTP/pub/MakeApp/maviking.zip.*

*Więcej informacji o programie można uzyskać w firmie RK-System, tel. (0-22) 755-69-83*



Rys. 9.