HALL2002

Przeszukując zasoby Internetu, można znaleźć tysiące urządzeń peryferyjnych do komputerów PC, będących podstawą dla dalszych konstrukcji tak amatorskich, jak i profesjonalnych. Karty we/wy, moduły mikroprocesorowe, różnego rodzaju przetworniki itd., itp. Zawsze liczymy na to, że do opisywanych projektów będzie dołączone oprogramowanie pozwalające na stworzenie pełnej aplikacji, albo przynajmniej pomagające w jej stworzeniu. Jednak tak nie jest, najczęściej wszystko co znajdziemy to kilka linii kodu np. w Pascalu. ***Rekomendacje:*** *interesujące narzędzie dla miłośników automatyki, stanowiące rozwiązanie pośrednie między dedykowanym systemem mikroprocesorowym a uniwersalnym sterownikiem PLC. Zainteresuje zarówno amatorów, jak i profesjonalistów.*

Wydawałoby się, że w dobie współczesnych narzędzi, choćby takich jak Delphi, pisanie programów jest łatwe, szybkie i przyjemne. Poniekąd jest to prawda. W ciągu jednego wieczoru można napisać mały, prosty program, ale już napisanie bardziej złożonego oprogramowania wymaga czasu, poza wiedzą i doświadczeniem oczywiście. A jak to mówią: czas to pieniądz. Dlatego też oprogramowanie oferowane z urządzeniami peryferyjnymi, jeżeli w ogóle jest dołączone, to stanowi najczęściej uproszczone do maksimum wersje służące tylko do testów i konfiguracji. Pozostaje zatem samodzielne napisanie programu. I znowu okazuje się, że to nie taka łatwa sprawa. Co innego stworzyć prostą aplikację, a co innego duży, wykończony w szczegółach i dobrze przetestowany program.

HALL2002 - zestaw programów narzędziowych

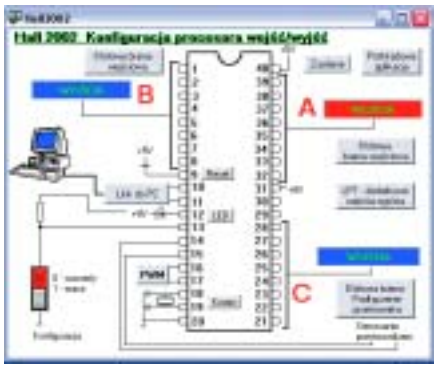
HALL2002 to próba rozwiązania opisanego wyżej problemu poprzez oddanie w ręce użytkownika kompletnego oprogramowania, z dołączonym mikrokontrolerem do obsługi wejść/wyjść, przekazujące na niego wykonanie sprzętowych zadań aplikacji. Pakiet programów narzędziowych pozwala na zbudowanie ekranu z różnymi przełącznikami, lampkami, wskaźnikami. Do dyspozycji mamy przekaźniki czasowe, liczniki, tabele alarmów, programatory czasowe i inne urządzenia logiczne. Aby wszystko to zebrać w całość i tchnąć w to życie, na potrzeby HALL-a powstał specjalny, prosty język programowania podobny do języków stosowanych w sterownikach PLC.

Można więc powiedzieć że HALL jest swego rodzaju sterownikiem Soft-PLC. Sterownikiem, który można wykorzystać do sterowania na przykład pracą dyplomową lub jakimiś urządzeniami w domu. Pomimo swego amatorskiego charakteru, HALL2002 może znaleźć wiele zastosowań profesjonalnych. Oczywiście obowiązują tu pewne ograniczenia - nie wszystkim można bezpiecznie sterować za pomocą komputera z systemem Windows. Nie mniej HALL poradzi sobie ze sterowaniem wielu urządzeń, choć do jego kuzyna - komputera HALL9000 sterującego statkiem kosmicznym z kultowego filmu Stanleya Kubricka - *Odyseja kosmiczna 2001*, trochę mu brakuje.

Procesor

Do oprogramowania dołączony jest procesor realizujący funkcje we/wy, komunikujący się z komputerem poprzez łącze szeregowe RS232. Procesor to popularny ATMEL 89C51 oprogramowany tak, aby mógł obsłużyć osiem sygnałów wejściowych, osiem wyjściowych oraz jedną uniwersalną bramę mogącą pracować jako wejścia lub wyjścia. Dodatkowo do dyspozycji są dwa wyjścia PWM oraz sygnały sterujące do ewentualnego podłączenia przetwornika ADC. Oczywiście jest też układ dopasowujący elektrycznie RS232 do standardu PC, np. MAX232. Tak przygotowany procesor stanowi podstawę dla dowolnej konstrukcji wykonanej przez użytkownika.

Obowiązują tu identyczne zasady jak przy tworzeniu dowolnej aplikacji z tym procesorem. Jednym z programów systemu HALL2002 jest program USCONFIG.EXE, wspomagający konfigurację procesora (**rys. 1**). Oprócz wejść/wyjść procesora możemy użyć 8 wyjść i 5 wejść z portu LPT. W oprogramowanie wbudowano symulator wejść/wyjść pozwalający na uruchamianie i testowanie projektów bez przygotowanego urządzenia. Przykładowy schemat sterownika opartego na procesorze HALL2002 przedstawia **rys. 2**. Brama wyjściowa za pośrednictwem układu ULN2803 steruje przekaźnikami. Do bramy wejściowej podłączono 8 wejść z optoizolacją. Brama uniwersalna została użyta do podłączenia przetwornika ADC0804. Procesor wystawia dwa sygnały w celu podłączenia przetwornika. Pierwszy z nich (wyprowadzenie 14) to sygnał inicjujący przetwornik, a drugi (wyprowadzenie 15) steruje multiplekserem analogowym (tutaj 4052). Jeżeli nie damy multiplek-



Rys. 1. Okno programu USCONFIG.EXE

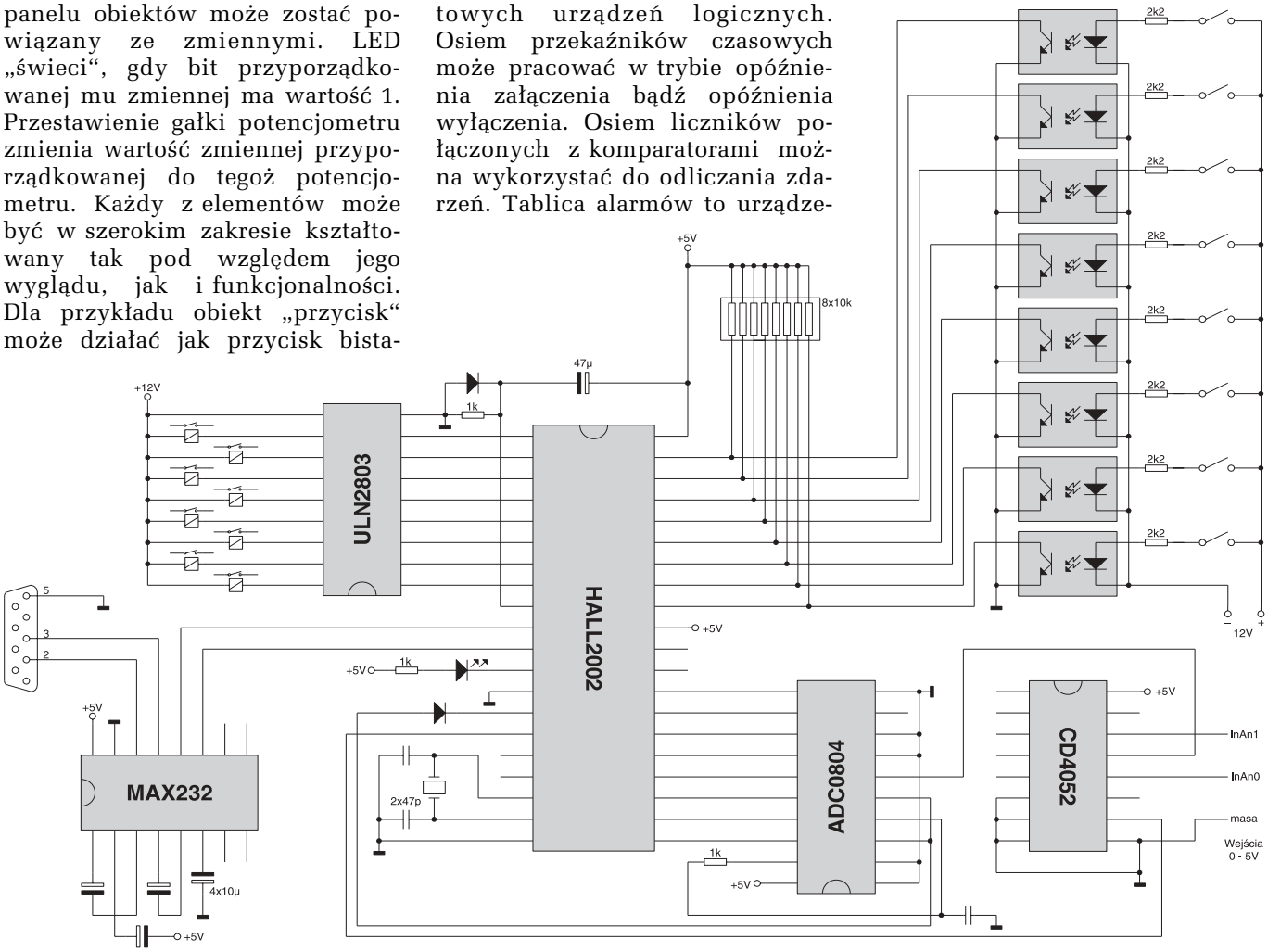
sera, to obydwa wejścia analogowe będą miały taką samą wartość.

Panel operatorski

Podstawą systemu jest panel operatorski, czyli okno, które będzie widoczne po uruchomieniu stworzonego przez użytkownika projektu. Panel operatorski można skomponować z wielu elementów, takich jak wirtualne diody LED, mierniki, przyciski, potencjometry. Każdy z umieszczonych na panelu obiektów może zostać powiązany ze zmiennymi. LED „świeci“, gdy bit przyporządkowanej mu zmiennej ma wartość 1. Przestawienie gałki potencjometru zmienia wartość zmiennej przyporządkowanej do tegoż potencjometru. Każdy z elementów może być w szerokim zakresie kształtowany tak pod względem jego wyglądu, jak i funkcjonalności. Dla przykładu obiekt „przycisk“ może działać jak przycisk bistabilny lub monostabilny, wpływający na stan przyporządkowanej zmiennej bitowej, może zwiększać lub zmniejszać wartość zmiennej typu word lub można za jego pomocą inicjować takie funkcje jak wyłączenie programu, otwieranie okien z programatorami czasowymi, wykresami, alarmami itp. Można kształtować wygląd przycisku, kolor, gradient, tekst dla przyciśniętego i puszczonego klawisza. Tłem panelu operatorskiego może być dowolna grafika. Możliwe jest podmienianie grafik w zależności od stanu zmiennych, co pozwala na stworzenie atrakcyjnie wyglądającego panelu operatorskiego. Nie jest to bez znaczenia np. w przypadku prac dyplomowych. Na **rys. 3** widzimy panel jednego z przykładowych projektów, którego zadaniem jest zaprezentowanie możliwości poszczególnych obiektów.

Urządzenia logiczne

Do dyspozycji mamy kilka gotowych urządzeń logicznych. Osiem przekaźników czasowych może pracować w trybie opóźnienia załączenia bądź opóźnienia wyłączenia. Osiem liczników połączonych z komparatorami można wykorzystać do odliczania zdarzeń. Tablica alarmów to urządzenie obsługujące listę alarmów. Urządzenie pozwala na przygotowanie ośmiu komunikatów dopisywanych do listy alarmów po zmianie przyporządkowanych zmiennych. Bez problemu można przygotować np. rejestr z wpisami o otwarciu bramy wjazdowej, gdzie każdy z komunikatów opatrzony będzie datą i godziną. Jeżeli chcemy, żeby urządzenie wydawało dźwięki, wystarczy wykorzystać zmienne Wav0...Wav15. Działają one tak, że każde zbocze narastające bitu WavX powoduje odtworzenie przyporządkowanego pliku WAV. Jeśli chcemy zlecić podlewanie ogródka np. we wtorek o godzinie 16, to pomoże nam w tym programator tygodniowy. Programator pozwala na dopisanie dowolnej liczby poleceń typu *[dzień_tygodnia; godzina; nr_wyjścia; załącz(wyłącz)]*. Jeśli jednak zapragniemy podlać kwiatki za trzy godziny, to do dyspozycji mamy dwa timery wzorowa-



Rys. 2. Schemat elektryczny przykładowego sterownika

Rys. 3. Widok przykładowego panelu aplikacji

ne na starych poczciwych minutnikach do jajek. Kontynuując ogródkowy wątek - gdybyśmy chcieli, aby podczas podlewania klombu zrobiła się na nim mała dyskoteka, to w sterowaniu lamp pomoże nam urządzenie logiczne zwane sekwencerem, przypominające działaniem programator mechaniczny - taki z krzywkami. Dwie zmienne - np. temperatury z naszego przykładowego ogródka pozyskane za pośrednictwem podłączonych przetworników ADC możemy wykreślić na wykresie trendów.

Ostatnim z wbudowanych urządzeń logicznych jest urządzenie pozwalające na uruchomienie zewnętrznych plików. Plików, a nie programów! Skutek jego działania jest identyczny jak w wyniku kliknięcia myszą w menedżerze Windows na dowolny plik. Zdarzenie aktywuje się narastającym zboczem zmiennych *Run0...Run3*. Uruchamiany jest wtedy plik, którego ścieżka dostępu znajduje się w pliku INI. Jeśli dla zmiennej *Run1* przyporządkujemy plik *c:\muzyka\autobiografia.mp3*, to zostanie uruchomiony program skojarzony z plikami *mp3* np. Winamp. Możemy więc przy podlewaniu kwiatków słuchać ulubionych „kawałków".

Język programowania

Mamy więc panel operatorski, urządzenia logiczne i procesor obsługujący nasze urządzenie. Aby to wszystko działało i to działało zgodnie z naszymi oczekiwaniami, musimy napisać program sterujący. Dla potrzeb HALL-a zaprojektowano specjalny język. Język programowania systemu to połączenie prostoty i funkcjonalności. Dziewięć rozkazów pozwala zarówno na proste powiązanie wbudowanych komponentów z wejściami i wyjściami, jak również na budowanie skomplikowanych algorytmów sterowania. Program składa się z 64 bloków. Każdy blok może składać się z 16 instrukcji. Rozkazy wykonywane są jeden za drugim - tak jak w typowym sterowniku PLC. Po zakończeniu bloku realizowany jest następny blok. Wykonanie bloku można przerwać poleceniem BREAK. Następuje wtedy przejście do następnego bloku. W połączeniu z rozkazami testującymi warunki rozkaz BREAK pozwala pominąć realizację części lub całego bloku, jeśli nie spełniono jakiegoś warunku. Taka konstrukcja może zdziwić osoby mające doświadczenie w pisaniu programów dla mikroprocesorów, a które nie pisały programów dla sterowników PLC. Program realizowany w kółko, bez skoków warunkowych, podprogramów i podobnych rozwiązań jest w pierwszym momencie nieco szokujący, ale z drugiej strony napisanie np. w asemblerze programu, który wykonuje kilka czynności równolegle, jest zajęciem iście karkołomnym, tym bardziej że stosowanie przerwań też ma swoje granice.

Z kolei prosta konstrukcja realizacji następujących po sobie rozkazów pozwala na odwzorowanie dowolnej sieci działań logicznych (tłumaczenie układu z podstawowych bramek na program). Teoretycznie w takim języku można oprogramować wszystko (język STD ze sterowników PLC). Stanowi to jednak niemałą sztukę.

Podział programu na bloki, o których realizacji nie możemy w dowolny sposób decydować, przybliża do siebie obie techniki, pozwalając na łatwą algorytmizację programu. Program operuje na zmiennych. Zmienna może być jednym bitem (wejście, wejście timera, wyjście timera) lub słowem (wejście analogowe, czas timera, stan licznika). W systemie HALL zmienne mają długość 16 bitów. Jeśli odczytywana wartość jest 8-bitowa - tak jak brama wejść cyfrowych lub przetwornik analogowo-cyfrowy, to starsze 8 bitów 16-bitowego słowa ma wartość zero. Wyjątkiem jest tu wyjście analogowe PWM, które jest 7-bitowe (0...127). Jeśli jego wartość jest większa, pozostałe bity zostają zignorowane. Poza zmiennymi opisującymi stan wejść/wyjść oraz zmiennymi skojarzonymi z urządzeniami zewnętrznymi mamy do dyspozycji akumulatory i markery. Akumulatory są zmiennymi tymczasowymi - tak binarnymi, jak i słowami służącymi jako podstawowe argumenty wykonywanych operacji. Akumulatory mogą być lokalne - odnoszące się do konkretnego bloku oraz globalne - dla całego programu. Markery, poza przechowywaniem zmiennych podczas przetwarzania programu, są odpowiedzialne za komunikację pomiędzy panelem operatorskim a programem. Każdy bit w systemie ma swój odpowiednik, który jest ustawiany, gdy bit podstawowy zmienia wartość z 0 na 1. Bit ten jest czytany zamiast bitu podstawowego, jeżeli w kodzie programu poprzedza go klucz ^. Dzięki takiemu rozwiązaniu przypisanie bitu do zmiennej z wejścia (lub markera) nastąpi tylko jeden raz w momencie podniesienia stanu wejścia z 0 na 1. Mówiąc inaczej, możemy pewne działania programu uzależnić nie od stanu zmiennej - np. wejścia, a od zbocza narastającego tej zmiennej. Edytor zbudowano tak, że prowadzi on za rękę tworzącego program - praktycznie nic tu nie piszemy, a tylko krok po kroku wybieramy z list nazwy rozkazów i zmiennych. Po pewnym czasie może być to trochę denerwujące, ale na początku odpada nam konieczność pamiętania składni i nazw zmiennych oraz wynikające z tego błędy (**rys. 4**).

Rozpatrzmy teraz pewien przykład. Załóżmy, że chcemy napisać prosty program, który będzie wyjście pierwsze (OUT0.0) załączał wejściem 1 (IN0.0), a wejściem drugim (IN0.1) będzie je wyłączał. Drugie wyjście (OUT0.1) ma zadziałać 5 sekund po załączeniu pierwszego wyjścia. Dodatkowo

Rys. 4. Okno edytora programu obsługującego aplikację

każde załączenie pierwszego wyjścia ma spowodować odtworzenie pliku dźwiękowego *bum.wav*. Tworzymy nowy projekt - nazwijmy go EPL. Do podkatalogu „pliki" dodamy plik *bum.wav*, a następnie wybieramy ustawienia urządzeń logicznych - moduł Wav i kojarzymy plik *bum.wav* ze zmienną wav0. W pierwszym bloku piszemy program:

```
01  IF  IN0.0    = Bit 1   //jeśli we0
02  Mov B OUT0.0 := Bit 1//załacz wy0
03  IF  IN0.1    = Bit 1   //jeśli we1
04  Mov B OUT0.0 :=Bit 0//wyłącz wy0
05  Mov B Wav0   := ^OUT0.0
          //jeśli zał. wy0 odtw. wav1
06  Mov B SetTi3 :=OUT0.0
          //wy0 do we ti3
07  Mov B OUT0.1 :=Ti3 //wy ti3 do wy1
```

Musimy jeszcze skonfigurować odpowiednio timer Ti3 - wykorzystywany w naszym programie. Ustawiamy go w tryb opóźnienia załączenia oraz wpisujemy 5 sekund. Teraz możemy uruchomić edytor panelu operatorskiego i dodać cztery kontrolki LED: dwie skojarzymy z wejściami IN0.0 i IN0.1, a dwie następne z wyjściami OUT0.0 i OUT0.1.

Jeśli coś się może zepsuć, na pewno się zepsuje

Jak już wspomniano, HALL2002 został stworzony jako system z pogranicza zastosowań amatorskich i profesjonalnych. Chcąc jednak zapędzić go do zastosowań profesjonalnych, trzeba pamiętać o kilku ważnych ograniczeniach związanych tak z samym systemem, jak i zastosowaniem komputera PC z systemem Windows do celów sterowniczych. Czas obróbki programu, czyli czas reakcji od załączenia wejścia do załączenia wyjścia (oczywiście dla programu, który przenosi bezpośrednio stan wejścia na wyjście) wynosi około 0,05 sekundy. Jednak system potrafi „płatać figle" i może (np. w sytuacji, gdy jakiś program próbuje uzyskać dostęp do nieistniejącej lub uszkodzonej dyskietki) nastąpić dłuższa przerwa w obróbce sygnałów. Trzeba więc poważnie przemyśleć konsekwencje chwilowego zatrzymania programu oraz konsekwencje zawieszenia się komputera. Ryzyko co prawda nie jest wielkie, szczególnie gdy zainstalujemy program na świeżo skonfigurowanym komputerze, ale zakładanie, że nic się nie stanie, byłoby objawem nieodpowiedzialności. Najlepszym wyjściem jest przeniesienie pewnych mechanizmów zabezpieczających na poziom sterowanych urządzeń. Posłużę się przykładem. Mamy wózek napędzany silnikiem o zmienianym kierunku obrotów i dwa wyłączniki krańcowe na końcach drogi, po której się porusza. Program ma za zadanie tak sterować wózkiem, aby jeździł on od wyłącznika do wyłącznika. Możemy to rozwiązać na dwa sposoby. Pierwszy jest taki, że podłączamy wyłączniki krańcowe do wejść, a styczniki lewo/prawo do wyjść. Jednak stosując takie rozwiązanie, ryzykujemy tym, że w przypadku zatrzymania programu jadący właśnie wózek nie zostanie zatrzymany i ulegnie uszkodzeniu. Innym wyjściem jest zbudowanie klasycznego układu nawrotnego z przyciskami prawo/lewo i wyłącznikami krańcowymi w obwodzie sterowania stycznikami, a następnie zastąpienie przycisków prawo/lewo przekaźnikami sterowanymi przez komputer. Jeżeli komputer załączy jazdę w jakimś kierunku i się zawiesi, to wózek dojedzie do wyłącznika krańcowego i się zatrzyma.

HALL i edukacja

Pomimo wielu różnic między HALL-em a sterownikami PLC, HALL2002 ma z nimi wiele wspólnego, podobnie jak z systemami wizualizacji i akwizycji danych (SCADA). Opanowanie sztuki pisania programów sterujących we wbudowanym w nim języku zaprocentuje z pewnością przy nauce jakiegokolwiek języka stosowanego w sterownikach programowalnych. W końcu wszystkie te języki próbują w jakiś sposób opisywać sieć działań logicznych, a u podstaw ich stworzenia leżał pomysł zamiany schematu elektrycznego lub schematu bloków logicznych (technika charakterystyczna dla Siemensa) na ciąg instrukcji programu.

Wojciech Mazurek
www.neuron.com.pl