ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1994
Seria: ELEKTRYKA z. 136 Nr kol. 1252

Jerzy JAKUBIEC

# WIELOZADANIOWA PRACA PRZYRZĄDU MIKROPROCESOROWEGO POD KONTROLĄ PROGRAMU DYSTRYBUCJI ZADAŃ

Streszczenie. W artykule opisano ogólną koncepcję budowy wielozadaniowego oprogramowania przeznaczonego dla przyrządów mikroprocesorowych. Podstawowym elementem oprogramowania jest program nazywany dystrybutorem zadań, który nadzoruje przydzielanie procesora do realizacji zadań zgodnie z ich priorytetem.

# MULTITASKING OPERATION OF INTELLIGENT INSTRUMENT UNDER CONTROL OF TASK DISTRIBUTION PROGRAM

Summary. A general conception of a multitasking software designed for an intelligent instrument is described in the paper. The basic part of the software is a program called task distributor which controls assigment of processor to realizing task accordingly with their priority.

# МНОГОЗАДАЧНАЯ РАБОТА МИКРОПРОЦЕССОРНОГО ПРИБОРА УПРАВЛЯЕМОГО ПРОГРАММОЙ РАСПРЕДЕЛЕНИЯ ЗАДАЧ

Резюме. В статье описана общая концепция построения многозадачного программного обеспечения предназначенного для микропроцессорных приборов. Главным элементом программного обеспечения является программа называемая дистрибутором (паспределителем) задач, которая управляет наделением процессора к реализации отдельных задач согласно их приоритету.

## 1. WSTĘP

Wśród współcześnie budowanych przyrządów mikroprocesorowych dużą grupę stanowią przyrządy pracujące w trybie ciągłym na bieżąco (nazywanym często pracą w czasie rzeczywistym). Przyrządy takie wykonują pomiary z określoną częstotliwością, zależną od kontrolowanego procesu, wykorzystując je do dalszych działań lub transmitując do innych urządzeń. Przyrządy tego rodzaju stosowane są w systemach pomiarowych, automatyzacyjnych, w robotach przemysłowych, zabezpieczeniach energetycznych i wielu innych. W ostatnim okresie szczególnie intesywnie rozwijane są tzw. przetworniki inteligentne, scalające w jeden element czujniki pomiarowe, układy przetwarzania analogowo-cyfrowego i mikroprocesor, które stają się swojego rodzaju standardem w budowie przyrządów mikroprocesorowych [1].

Cechą charakterystyczną przyrządów pracujących na bieżąco jest wykonywanie wielu zadań, takich jak obsługa procesu pomiarów, wstępna obróbka danych pomiarowych, korekcja błędów wyników, obsługa procedur transmisji, reakcja na zdarzenia zewnętrzne i innych. Istotne jest przy tym, że zadania te wykonują z reguły stosunkowo proste mikroprocesory 8-bitowe o relatywnie małej szybkości działania. Pomimo tego ich właściwości na ogół zapewniają wystarczająco sprawną obsługę wszystkich zadań w przeciętnych warunkach. Poszczególne zadania są realizowane wówczas kolejno w sposób narzucony przez programistę z ewentualnym zastosowaniem przerwań w działaniach wymagających szybkiej reakcji. Jednak w sytuacjach, gdy występuje spiętrzenie zadań - przykładowo na skutek ingerencji operatora systemu, wystąpienia stanu awaryjnego - taki sposób obsługi zadań może się okazać niewystarczający. W wyniku tego mogą się pojawić zakłócenia w pracy przyrządu będące efektem opóźnionej realizacji jednych zadań na skutek zbyt długiej realizacji dużej liczby zadań poprzednich.

Powyższe trudności na ogół udaje się pokonać stosując konstrukcje programowe pozwalające na wielozadaniową pracę przyrządu mikroprocesorowego, czyli równoległą (współbieżną) realizację wielu zadań przez jeden procesor w wymaganym

tempie, również w warunkach spiętrzenia zadań. Praca wielozadaniowa realizowana jest na ogół dwoma sposobami [3]:

- przez wykorzystanie systemu przerwań, wówczas część zadań jest obsługiwana przez program główny, nazywany w takich sytuacjach programem tła, pozostałe - za pomocą programów obsługi przerwań,
- przez cykliczne przydzielanie kolejnych przedziałów czasu pracy procesora kolejnym programom realizującym zadania (tzw. praca z podziałem czasu).

Wymienione dwa sposoby pracy wielozadaniowej nie zawsze są wystarczająco efektywne, czego główną przyczyną są bardzo ograniczone możliwości zmiany struktury zadań w trakcie ich wykonywania, a tym samym dostosowania działania przyrządu do zmiennych warunków zewnętrznych. Ten brak elastyczności jest spowodowany w pierwszym przypadku głównie przez sprzętową (a więc słabo poddającą się adaptacji) organizację systemu przerwań, w drugim - przez sztywne przydzielanie jednakowych odcinków czasu procesora kolejnym zadaniom.

Opisana sytuacja stanowi podstawową przyczynę poszukiwań nowych rozwiązań organizujących wielozadaniową pracę przyrządu mikroprocesorowego. Jednak nie jedyną. Można wskazać także i drugi powód, dla którego warto zajmować się tego rodzaju konstrukcjami. Jest nim potrzeba pewnej unifikacji procesu tworzenia oprogramowania różnych wersji przyrządów mikroprocesorowych. Aktualnie na ogół oprogramowanie każdego nowo konstruowanego przyrządu budowane jest od podstaw przy ograniczonym wykorzystywaniu przez programistę własnych uprzednio zbudowanych podprogramów. Tymczasem można wskazać wiele zadań realizowanych w taki sam lub bardzo podobny sposób w różnych konstrukcjach przyrządów. Przykładowo można wymienić tutaj pomiar czasu astronomicznego, obsługę typowych układów wejścia/wyjścia, realizację protokołu transmisji i wiele innych. Stworzenie nadrzędnej, wielozadaniowej struktury programowej pozwalającej na proste dołączanie do niej programów poszczególnych zadań, w tym dużej grupy zadań standardowych, może w istotnym stopniu uprościć proces dochodzenia do końcowej postaci programu użytkowego.

Dalsza część artykułu poświęcona jest głównie opisowi rozwiązania zaproponowanego w pracy [2], które może stanowić uniwersalny szkielet konstrukcyjny wielozadaniowej pracy przyrządów mikroprocesorowych o rożnorakich przeznaczeniach, a zarazem umożliwia adaptacyjną zmianę struktury zadań w trakcie pracy przyrządu. Podstawą tego sposobu jest podział poszczególnych zadań przyrządu na zadania cząstkowe i przydział procesora do ich realizacji zgodnie z przyjętą hierarchią ważności - stąd program organizujący pracę nazwano programem dystrybucji zadań lub w skrócie dystrybutorem. Główne zasady budowy dystrybutora są podobne do stosowanych w konstrukcjach wielozadaniowych systemów operacyjnych przeznaczonych do zastosowań w sterownikach systemów pomiarowych i automatyzacyjnych [4].

W kolejnych dwóch punktach omówiono podstawowe założenia budowy dystrybutorów, ich ogólną strukturę, a następnie opisano charakterystyczne rozwiązania programowe na przykładzie dystrybutora przeznaczonego dla mikrokontrolera INTEL 8051. Przedstawione rozważania, pomimo ich ukierunkowania na zagadnienia budowy dystrybutorów, można w dużej mierze odnosić do problemów wspólnych dla wielozadaniowej pracy jednoprocesorowych przyrządów pomiarowych.

## 2. OGÓLNE ZAŁOZENIA BUDOWY PROGRAMU DYSTRYBUCJI ZADAŃ

Można wyróżnić dwie podstawowe zasady organizacji pracy wielozadaniowej w strukturach jednoprocesorowych - są one stosowane przy budowie oprogramowania zarówno sterowników dużych systemów [4], jak i prostych przyrządów mikroprocesorowych. W skrócie można je określić jako:

- zasadę podziału zadań,
- zasadę priorytetu zadań.

Zasady te wynikają z dość oczywistego faktu, że w danym okresie procesor może wykonywać jeden program, czyli realizować jedno z zadań. Zatem, aby zewnętrzny efekt pracy procesora był odbierany jako równoległe wykonywanie wielu zadań, muszą być

one podzielone na wiele zadań cząstkowych i realizowane naprzemiennie zgodnie z określonym priorytetem wskazującym kolejność ich wykonywania.

Dla programu dystrybucji przyjęto, że kryterium podziału zadań na zadania cząstkowe stanowi maksymalny dopuszczalny czas reakcji procesora na inicjację jednego zadania w trakcie wykonywania innego. Wynika stąd zasada, że każde z zadań należy podzielić na taką liczbę zadań cząstkowych (nazywanych dalej również w skrócie zadaniami), aby czas realizacji każdego z nich przez procesor nie przekraczał przyjętej wartości dopuszczalnej.

Wielość zadań realizowanych przez procesor narzuca z kolei potrzebę określenia kolejności ich wykonywania, innymi słowy - nadania określonego priorytetu każdemu z zadań. Można wskazać dwa zasadnicze rodzaje priorytetu. Pierwszy określa kolejność wykonywania zadań w sytuacji, gdy kilka z nich jednocześnie żąda obsługi przez procesor. Drugi rodzaj priorytetu wskazuje, czy realizacja danego zadania może być przerwana przez inne - czyli ogólnie określa warunki wzajemnego przerywania realizacji jednych zadań przez inne. Relacje priorytetowe między poszczególnymi zadaniami tworzą strukturę priorytetów. Struktura ta może się zmieniać, między innymi możliwe są jej adaptacyjne zmiany w zależności od warunków pracy przyrządu lub systemu. Przykładowe możliwości w tym zakresie pokazano w punkcie 4.

## 3. OGÓLNA STRUKTURA DYSTRYBUTORA ZADAŃ

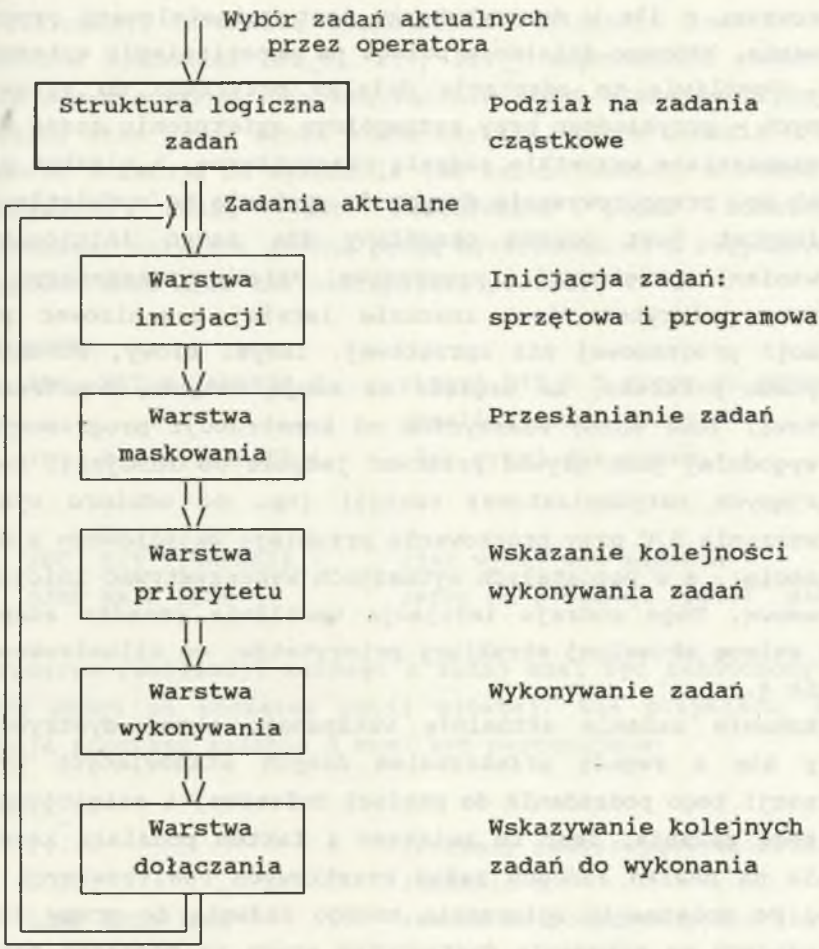
W strukturze oprogramowania wielozadaniowego można wyróżnić dwie podstawowe części: jądro i kompleks zadań. Jądro stanowią te programy, które organizują i nadzorują wykonywanie zadań przez procesor. Kompleks zadań tworzą dwojakiego rodzaju programy. Przede wszystkim są to programy użytkowe realizujące zadania właściwe dla danego przyrządu. Drugą grupę stanowią programy pomocnicze, wśród których można wymienić programy testujące, program zarządzający MONITOR i podobne.

Jądro w wielozadaniowych systemach operacyjnych stanowi dość złożoną konstrukcję programową, szczególnie w systemach wieloprocesorowych [4]. Wynika to ze specyfiki zadań realizowanych przez taki system operacyjny. Inaczej to wygląda w przyrządach mikroprocesorowych. Tutaj można uzyskiwać bardzo proste, tym niemniej efektywne, konstrukcje praktycznie nie wpływające na szybkość realizacji zadań, co ilustrują przykładowe rozwiązania opisane w punkcie 4.

Dystrybutory zadań mogą być budowane w różny sposób - zależy to od wymagań stawianych przyrządowi i od inwencji programisty. Można jednak wskazać pewne ogólne zasady ich budowy będące wynikiem zarówno możliwości, jakie stwarzają współczesne mikroprocesory w tym zakresie, jak i ogólnych założeń budowy dystrybutorów. Analiza ogólnej struktury dystrybutorów zadań, pokazanej na rys.1, pozwala na ocenę zakresu ich możliwości jako programów zarządzających wielozadaniową pracą przyrządów mikroprocesorowych.

Diagram przedstawiony na rys.1 ma strukturę warstwową. Każda z warstw oznacza spójny zespół czynności wykonywanych kolejno na zbiorze zadań aktualnych. Wszystkie zadania, które potencjalnie może realizować procesor, tworzą pewną strukturę logiczną. Struktura ta obejmuje podział na zadania cząstkowe, lokalizację w pamięci programu odpowiednich podprogramów realizujących tu zadania, określa przydział pamięci, sposoby przekazywania danych i temu podobne. W danym okresie tylko część zadań jest realizowana przez procesor - są to zadania aktualne. Wyboru tych zadań dokonuje operator za pomocą zespołu przycisków (klawiatury przyrządu) lub przy użyciu komputera nadrzędnego. Przykładowo operator może zarządzić testowanie bądź autokalibrację przyrządu, po czym wprowadza przyrząd w tryb pomiarowy.

Ze zbiorem zadań aktualnych związana jest pewna struktura logiczna, na której wykonywane są kolejne działania przedstawione na rys.1 w sposób warstwowy. I tak warstwa inicjacji określa sposób aktywizowania zadań, który może być sprzętowy (przy wykorzystaniu sygnałów przerwań) lub programowy - w tym drugim przypadku należy określić procedurę inicjacji.




Rys.1. Ogólna struktura dystrybutora zadań nadzorującego wieloprogramową pracę przyrządu mikroprocesorowego

Fig.1. General structure of the task distributor

Przykładowo w mikrokontrolerze INTEL 8051 można w tym celu wykorzystać pamięć danych adresowaną bitowo: ustawienie w stan 1 określonego bitu oznacza inicjację zadania, wyzerowanie - zadanie zostało wykonane lub przyjęte do wykonania.

Zadania zainicjowane (zgłoszone do wykonania) mogą nie być realizowane, o ile w dystrybutorze jest zainstalowany program maskowania, którego działanie polega na przesłanianiu wybranych zadań. Umożliwia to adaptację działań przyrządu do sytuacji skrajnych - przykładowo przy szczególnym spiętrzeniu zadań mogą być przesłaniane wszystkie zadania czasochłonne, a niezbyt pilne, jak np. przygotowywanie danych do wysłania na wyświetlacz.

Priorytet jest osobno określany dla zadań inicjowanych przerwaniami sprzętowymi i programowo. Należy tu zaznaczyć, że struktury priorytetu jest znacznie łatwiej organizować przy inicjacji programowej niż sprzętowej. Innymi słowy, struktura priorytetu przerwań, ze względu na swoją sztywną konstrukcję sprzętową, jest mniej elastyczna od konstrukcji programowych. Stąd wygodniej jest używać przerwań jedynie do inicjacji zadań wymagających natychmiastowej reakcji (np. do odbioru wyniku przetwarzania A/C przy próbkowaniu przebiegu wejściowego z dużą szybkością), a w pozostałych sytuacjach wykorzystywać inicjację programową. Tego rodzaju inicjacja umożliwia ponadto adaptacyjną zmianę aktualnej struktury priorytetów, co zilustrowano w punkcie 4.

Wykonanie zadania aktualnie wskazanego przez dystrybutor kończy się z reguły przekazaniem danych stanowiących wynik realizacji tego podzadania do pamięci buforowej i zainicjowanie kolejnego zadania. Jest to związane z faktem podziału każdego zadania na pewien łańcuch zadań cząstkowych realizowanych kolejno. Po dołączeniu zgłoszenia nowego zadania do grupy zadań oczekujących na wykonanie dystrybutor wraca na początek procedury przeglądania zadań zgłaszanych do wykonania.

## 4. PRZYKŁADOWE ROZWIĄZANIA DYSTRYBUTORA ZADAŃ MIKROKONTROLERA INTEL 8051

Załóżmy, że mikrokontroler 8051 wykonuje 8 zadań oznaczonych jako ZADANIE_0, ..., ZADANIE_7, uszeregowanych w kolejności od najbardziej do najmniej pilnego do wykonania. Każdemu zadaniu odpowiada 1 bit w pamięci adresowanej bitowo, który jest zapa-

lany (ustawiany w stan 1), gdy zadanie ma być wykonane i gaszony (zerowany) z chwilą rozpoczęcia realizacji zadania. Bity oznaczone symbolami BIT_0, ..., BIT_7 odpowiadają zadaniom o takim samym numerze i tworzą łącznie słowo stanu dystrybutora. Aktualny stan bitów słowa stanu określa, które zadania w danym momencie oczekują na wykonanie (są zainicjowane), a które nie. Wykonywanie zadań jest nadzorowane przez konstrukcję programową, nazywaną główną pętlą dystrybutora. W rozpatrywanym przypadku może mieć ona następującą postać:

```
MAIN_LOOP:
    JBC  BIT_0,ZADANIE_0      ;kasuj BIT_0 i skocz do programu
                              ;realizującego zadanie 0
    JBC  BIT_1,ZADANIE_1      ;jak wyżej dla zadania 1
          :
    JBC  BIT_7,ZADANIE_7      ;jak wyżej dla zadania 7
    SJMP MAIN_LOOP            ;wróć na początek pętli głównej
```

Program realizacji każdego z zadań musi być zakończony rozkazem skoku na początek pętli głównej. Dla przykładu, konstrukcja programu zadania 5 musi być następująca:

```
ZADANIE_5:
    .....                     ;rozkazy realizujące działania
    .....                     ;zadania 5
    LJMP MAIN_LOOP            ;skocz na początek pętli głównej
```

Powyższa konstrukcja programów obsługi zadań powoduje, że program po wykonaniu każdego z nich wraca na początek pętli głównej. Zatem następnym wykonywanym zadaniem będzie to spośród zgłoszonych, które jest usytuowane najbliżej początku pętli głównej. Taka kolejność wykonywania zadań powoduje, że im zadanie jest ulokowane bliżej początku pętli głównej, tym częściej jest przeglądany jego bit rejestru stanu i ma ono wyższy priorytet (pierwszeństwo) wykonywania.

Każdemu bitowi słowa stanu dystrybutora może towarzyszyć bit maski służący do ewentualnego przesłaniania (maskowania) zada-

nia. Zadanie przesłonione nie jest wykonywane, nawet jeśli zostanie zainicjowane. Gdy oznaczymy bity maski jako MASK_0, ..., MASK_7, gdzie numery 0,...,7 odpowiadają bitom słowa stanu, pętla główna uwzględniająca stan tych bitów przy aktywizowaniu zadań ma następującą postać:

```
MAIN_LOOP:
     JNB  MASK_0,ML01          ;skocz gdy bit maski wyzerowany
     JBC  BIT_0,ZADANIE_0      ;wykonaj zad. 0 gdy zainicjowane
ML01:
     JNB  MASK_1,ML02          ;skocz gdy bit maski wyzerowany
     JBC  BIT_1,ZADANIE_1      ;wykonaj zad. 1 gdy zainicjowane
          :
          :
ML07:
     JNB  MASK_7,MAIN_LOOP     ;skocz na początek gdy zero
     JBC  BIT_7,ZADANIE_7      ;wykonaj zad. 7 gdy zainicjowane
     SJMP MAIN_LOOP            ;skocz na początek pętli głównej
```

Dotychczas zakładano, że zadania są realizowane w kolejności od zadania 0 do 7 (o ile kilka z nich zostało zainicjowanych), co oznacza zarazem, że najwyższy priorytet ma zadanie 0 - najniższy 7. Możliwe jest jednak praktycznie dowolne inne uszeregowanie zadań. Umożliwia to adaptację kolejności realizacji zadań do warunków pracy przyrządu, jednak wymagane jest tworzenie nowych postaci pętli głównej dostosowanych do konkretnej kolejności. Załóżmy, że dla innej kolejności zadań główna pętla nosi nazwę MAIN_LOOP_1, dla jeszcze innej: MAIN_LOOP_2, itd. Wybór odpowiedniej pętli głównej , a tym samym zmiana priorytetów zadań, realizowany jest za pomocą poniższego programu. Przed jego wywołaniem należy umieścić w akumulatorze numer aktywizowanej pętli głównej: 0 dla pętli podstawowej, 1, 2,... dla kolejnych wersji pętli.

```
LOOP_SELECT:
     MOV  DPTR,#LOOP_TABLE     ;wprowadź do wskaźnika danych
                               ;adres tablicy adresów pętli
     RL   A                    ;mnóż przez 2
     JMP  @A+DPTR              ;skocz do wybranej pętli
```

```
LOOP_TABLE:
     AJMP MAIN_LOOP          ;pętla podstawowa
     AJMP MAIN_LOOP_1        ;pętla nr 1
          :
          :
```

Dystrybutor wykorzystuje dwojakiego rodzaju sposoby inicjacji zadań: programowy i sprzętowy (sygnałowy). Sposób programowy polega w tym przypadku na tym, że zakończenie wykonywania jednego zadania powoduje zapalenie bitu stanu zadania kolejnego w łańcuchu. Przykładowo przetworzenie wyniku pomiaru do postaci właściwej do wyświetlenia powoduje zapalenie bitu zadania realizującego przesłanie wyniku na wyświetlacz. Drugi sposób polega na wykorzystaniu sygnałów przerwań. Trzeba tu jednak zaznaczyć, że podprogramy obsługi przerwań powinny być zredukowane do niezbędnego minimum. Wynika to stąd, że sygnały przerwań mogą się pojawiać w dowolnych momentach i w pewnych sytuacjach może się pojawić ich kilka. W takich przypadkach zbyt długi czas realizacji podprogramów obsługi przerwań może spowodować przekroczenie dopuszczalnego czasu realizacji zadania bieżącego (przerwania wydłużają czas realizacji zadania o czas wykonania obsługi przerwania). Stąd można powiedzieć, że właściwie rola podprogramów obsługi przerwań sprowadza się do zapalenia odpowiedniego bitu rejestru stanu inicjującego już w sposób programowy wykonywanie zadania związanego z danym przerwaniem. Przykładowo, gdy koniec pomiaru sygnalizowany jest przerwaniem z przetwornika A/C, to w podprogramie obsługi realizowane jest jedynie przesłanie wyniku do określonej komórki pamięci i zapalenie odpowiedniego bitu rejestru stanu, co sygnalizuje uzyskanie wyniku dla celów dalszej obróbki. Pozostała część zadania, związana z przetwarzaniem wyniku, realizowana jest pod kontrolą pętli głównej.

## 5. UWAGI KOŃCOWE

Opisany program dystrybucji zadań w przyrządzie mikroprocesorowym konstruowany jest bardzo prostymi środkami, tym nie-

mniej daje duże możliwości realizacji wieloprogramowej pracy przyrządu w pełni zaspokajając potrzeby szerokiej klasy przyrządów pomiarowych. Prostota budowy dystrybutora ma wiele zalet. Przede wszystkim czas realizacji działań związanych z samą obsługą pętli głównej jest znikomo krótki w porównaniu z czasem realizacji zadań - nie wpływa zatem na szybkość działania przyrządu. Prostota pętli głównej pozwala na uzyskanie dużej przejrzystości oprogramowania, a tym samym ułatwia kontrolę realizacji zadań przez przyrząd, co jest bardzo istotne w fazie uruchamiania nowych programów. Możliwe jest dołączanie dużej liczby zadań, maskowanie części z nich w pewnych sytuacjach, co daje możliwość tworzenia uniwersalnych struktur programowych wykorzystywanych przez programistę w zależności od konkretnego przeznaczenia przyrządu. Możliwa jest ponadto adaptacja działania przyrządu do konkretnych sytuacji pomiarowych bezpośrednio w trakcie wykonywania innych działań. Własności te pozwalają na stwierdzenie, że na bazie oprogramowania, o strukturze podobnej jak w przypadku opisanego dystrybutora zadań, możliwe jest podjęcie próby budowania wirtualnych przyrządów mikroprocesorowych - czyli przyrządów o uniwersalnej konstrukcji sprzętowej i programowej dostosowywanych w sposób programowy do realizacji konkretnych zadań pomiarowych.

LITERATURA

1. Brignell J.E.: Smart Sensors, in T.Grandtke and W.H.Ko (eds), Sensors; A Comprehensive Surevey, Vol.1, Fundamentals and General Aspects, VCH Weinheim, 1989, Ch.12.
2. Jakubiec J.: Program dystrybucji zadań w mikroprocesorowym przyrządzie pomiarowym, ZN Pol. Śl. ser. Elektryka nr 134, Gliwice 1994.
3. Misiurewicz P.: Podstawy techniki mikroprocesorowej. WNT, Warszawa 1991.
4. Stevens P.: Środowisko oprogramowania aplikacji przemysłowych. Elektronizacja, 1993, nr 12.

Recenzent: prof. dr hab. inż. Michał Szyper

Wpłynęło do Redakcji 15 marca 1994

**Abstract**

One of the way of construction of multitasking software designed for intelligent instruments consists in dividing the tasks realized by the instrument to a number of partial tasks and then executing them by turns taking into account priority of the tasks. Number of the partial tasks has to be established in this manner so that execution time each of them does not exceed a permissible value. The paper presents a general structure of a program called tasks distributor developed accordingly with the conception described above. Activity of the distributor is characterized by the block-diagram shown in fig.1 where respective element represents operation which are realized by the distributor on the set of tasks. Some of these operations are illustrated by programs written in the assembler of the microcontroller INTEL 8051.