ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1997
Seria: ELEKTRONIKA z.10 Nr kol.1380

Witold BARAN
Instytut Elektroniki, Politechnika Śląska

OPROGRAMOWANIE 16-BITOWEGO SYSTEMU WIELOPROCESOROWEGO

Streszczenie. W pracy opisano oprogramowanie 16-bitowego systemu wieloprocesorowego zbudowanego na podstawie mikroprocesorów rodziny Intel iAPX86. Przedstawiono podstawowe elementy oprogramowania oraz wnioski dotyczące ważniejszych problemów związanych z programowaniem systemów wieloprocesorowych.

THE SOFTWARE OF A 16-BIT MULTIPROCESSOR SYSTEM

Summary. The article presents the software of the multiprocessor system based on the Intel iAPX86 microprocessors. The basic elements of the software are presented as well as the most important conclusions concerning with programming multiprocessor system.

1. Wstęp

W pracy zaprezentowano model laboratoryjny systemu wieloprocesorowego z wykorzystaniem mikroprocesorów 16-bitowych należących do rodziny iAPX86. Mikroprocesory te posiadają wewnętrzną architekturę umożliwiającą pracę wieloprocesorową. Podstawowe własności tych mikroprocesorów polegają na tym, że w czasie pracy istnieją okresy czasowe, w których mikroprocesor nie korzysta z magistrali systemowej (BUS IDLE). Jest to skutkiem istnienia wewnętrznej kolejki rozkazów, stanowiącej pamięć notatnikową dla programów w formie zalążkowej (cache memory). Te swobodne okresy czasu mogą być wykorzystane przez inne procesory przyłączone do wspólnej magistrali w celu wymiany informacji z urządzeniami również do niej przyłączonymi.

Jako podstawowe założenie przyjmuje się, że wspólnym zasobem w projektowanym systemie jest wyłącznie pamięć operacyjna. W dalszym ciągu pamięć ta określana jest terminem pamięci globalnej, a magistrala łącząca - magistralą globalną. Z punktu widzenia każdego pro-

cesora przestrzeń adresowa podzielona jest logicznie na dwie części: pamięć globalną i pamięć lokalną. Do pamięci lokalnej posiada dostęp tylko jeden procesor bez angażowania magistrali globalnej. Rozróżnienie rodzaju pamięci odbywa się przez adresy.

Drugie założenie dotyczy liczby procesorów. Przyjmuje się, że w systemie może pracować do 8 procesorów. Konstrukcyjnie wszystkie procesory są identyczne, jednak funkcjonalnie jeden z nich jest wyróżniony i w dalszym ciągu nazywany będzie procesorem MASTER. Główna funkcja tego procesora polega na planowaniu i zlecaniu zadań częściowych dla pozostałych procesorów zwanych procesorami SLAVE.

Trzecim założeniem podstawowym jest to, że projektowany układ nie jest samodzielny operatorsko, tzn. nie zawiera systemu operacyjnego i urządzeń do komunikacji z człowiekiem. Konieczna jest w związku z tym współpraca z nadrzędnym systemem komputerowym, którym może być dowolny komputer klasy PC. Uniwersalne oprogramowanie narzędziowe takich komputerów wykorzystywane jest do przygotowywania zadań dla systemu wieloprocesorowego oraz śledzenia ich realizacji.

Kolejne założenie podstawowe polega na tym, że każdy z procesorów wyposażony jest w bibliotekę procedur arytmetycznych wielokrotnej precyzji (dla liczb stało- i zmiennoprzecinkowych), której kopia umieszczona jest w pamięci stałej ROM, w zasobach lokalnych. Dzięki temu uzyskuje się dalsze zmniejszenie obciążenia magistrali globalnej i w efekcie zwiększenie prędkości działania systemu.

Strukturalny schemat systemu przedstawiony jest na rys. 1.

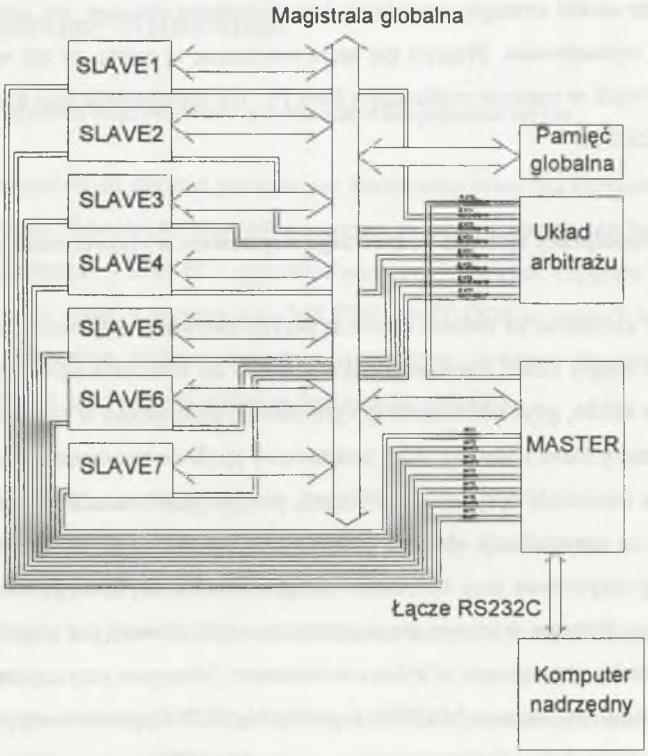
Dalsza część pracy zawiera opis BIOS'ów procesorów Master i Slave oraz oprogramowania komputera nadrzędnego dla systemu wieloprocesorowego

W opracowaniu wykorzystano zestaw procedur arytmetyki wielokrotnej precyzji dla mikroprocesorów 16-bitowych. Założenia do tej biblioteki sformułowane zostały na podstawie eksperymentów wykonanych w zrealizowanym wcześniej systemie z procesorami 8-bitowymi. Biblioteka zawiera 44 funkcje obejmujące m.in. pełny zestaw operacji arytmetycznych stało- i zmiennoprzecinkowych zgodnych ze standardem IEEE 754. Istnienie tych procedur w każdym procesorze zmniejszy liczbę cykli interfejsowych przy dostępie do pamięci globalnej i zwiększy efektywność obliczeniową całego systemu.

Ze względu na zastosowanie, do konstrukcji systemu wieloprocesorowego, standardowych płyt głównych z komputerów zgodnych z IBM PC/AT, koncepcja oprogramowania standardowego oparta została na maksymalnym wykorzystaniu istniejącego oprogramowania BIOS zawartego w pamięci EPROM. Spowodowało to konieczność modyfikacji programu

BIOS poprzez dostosowanie go do działania bez systemu operacyjnego (np. MS DOS) i bez standardowych urządzeń peryferyjnych. Jako jedyne urządzenie peryferyjne (umożliwiające komunikację z systemem nadzorczym) pozostawiono możliwość obsługi interfejsu RS-232C. Wprawdzie jest on tylko wykorzystywany w procesorze Master, lecz pozostawienie go w pozostałych procesorach pozwoliło zachować maksymalną jednorodność oprogramowania wszystkich procesorów.

Przyjmując powyższe założenia ze standardowego programu BIOS, pozostają jedynie procedury testujące podzespoły jednostki centralnej, inicjujące kontrolery bezpośrednio z tą jednostką współpracujące oraz przenoszące sterowanie do programu obsługi działania systemu wieloprocesorowego.



Rys. 1. Struktura systemu wieloprocesorowego
Fig. 1. Structure of the multiprocessor system

2. Oprogramowanie systemu wieloprocesorowego

Jako system nadzorczy został wybrany komputer klasy IBM PC, natomiast łącze zrealizowano przy użyciu łącza szeregowego RS232C. Decyzja o użyciu w urządzeniu łącza szeregowego, dużo wolniejszego w stosunku do równoległego, wynikała z faktu projektowania użycia systemu wieloprocesorowego do obliczeń numerycznych, co wiąże się ze stosunkowo małą częstotliwością wymiany informacji między jego poszczególnymi elementami. Przy obliczeniach, przeważnie, czas przesyłania danych jest dużo mniejszy niż czas wykonywania obliczeń. Widać tu oczywiście kolejną trudność wyłaniającą się przed użytkownikiem realizującym obliczenia. Musi on nie tylko dobrze opracować rozdział zadań w systemie wieloprocesorowym, ale także ustalić strategię przesyłania danych podczas obliczeń, np. przez odpowiednią kolejność ich wykonywania. Przyjęty typ łącza ma jeszcze tę zaletę, że nie wymaga żadnych zmian sprzętowych w systemie nadzorczym IBM PC. Na wyposażeniu tego komputera znajduje się złącze RS232C.

2.1. Zasady współpracy systemu nadzorczego z systemem wieloprocesorowym

Przyjęcie założenia, że zadania i dane są przygotowywane w systemie nadzorczym, spowodowało, że mogły zostać użyte standardowe środki do tworzenia oprogramowania na mikroprocesorze 80286, gdyż taki właśnie typ procesora wykorzystano w systemie wieloprocesorowym (procesory Slave i Master). Jako podstawowy język do tworzenia aplikacji w systemie, ze względu na możliwość optymalizacji obliczeń, przyjęto język asembler 80x86. Jako podstawowe kryterium optymalizacji obliczeń przejęto czas ich realizacji. Wynika stąd następująca metodologia postępowania przy tworzeniu oprogramowania użytkowego dla systemu wieloprocesorowego. Program źródłowy dla asemblera przygotowywany jest przy użyciu dowolnego edytora tekstów, pracującego w trybie nie-dokument. Następnie przy użyciu standardowych asemblerów i linkerów systemu MS-DOS (np. TASM, TLINK) generowany jest program wynikowy, który może być wykonywany w procesorze MASTER systemu wieloprocesorowego. Podobnie przygotowywane są zbiory danych. Ponieważ żaden program nie jest wolny od błędów, uruchomienie programu może nastąpić przy użyciu programów uruchomieniowych, np. TD lub AFD. Tak przygotowane zbiory będą następnie transmitowane do systemu wieloprocesorowego i tam wykonane.

Oprogramowanie opracowane w ramach tej pracy podzielić można na dwa rodzaje :

- oprogramowanie systemu wieloprocesorowego,
- oprogramowanie systemu nadzorczego.

Zadaniem pierwszego jest realizacja konkretnych zadań obliczeniowych i wstępne testowanie sprzętu (procesory SLAVE i MASTER). Procesor MASTER realizuje także komunikację z systemem nadzorczym i daje możliwość ładowania zadania i danych oraz odczyt wyników. Programy te zawarte są w pamięciach stałych poszczególnych procesorów.

Zadaniem oprogramowania systemu nadzorczego jest stworzenie środowiska pozwalającego na uruchamianie wcześniej napisanych aplikacji w systemie wieloprocesorowym.

3. Program sterujący Master-Multi

3.1. Miejsce programu Master-Multi w systemie wieloprocesorowym

Program Master-Multi stanowi podstawowe środowisko pracy dla użytkownika systemu wieloprocesorowego. Stanowi on nadrzędny program w całym systemie, jest łącznikiem między systemem sterującym - IBM PC i systemem wieloprocesorowym. Program Master- Multi działa pod nadzorem systemu operacyjnego MS DOS lub PC DOS w wersji 3.30 lub wyższej. Umożliwia on transmisję bloków danych i programów zarówno do pamięci lokalnej (pojemność - 640kB), jak i pamięci globalnej (pojemność - 320kB).

3.2. Opis programu Master-Multi

Podstawowe funkcje programu Master-Multi

1. komunikacja z użytkownikiem w otoczeniu systemu operacyjnego komputera nadzorczego;
2. komunikacja z procesorem Master i procesorami Slave poprzez łącze szeregowe;
3. operowanie zbiorami z wykorzystaniem zasobów komputera nadzorczego.

Uproszczony schemat blokowy programu

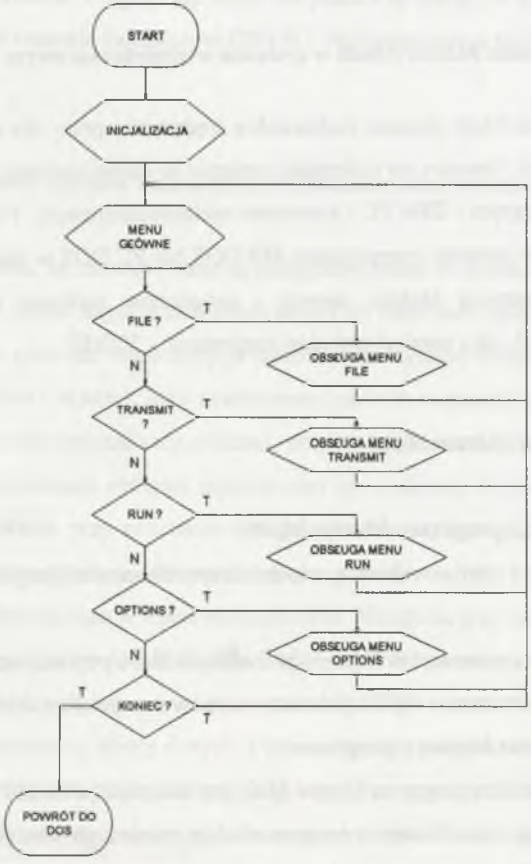
Pierwszą czynnością programu Master-Multi jest inicjalizacja kanału transmisji szeregowej oraz zmiennych programu. Następnie program obsługuje menu główne programu. Po wyborze dowolnej z opcji wykonywana jest jej obsługa, a po jej zakończeniu następuje powrót do menu

głównego. Z menu głównego możliwy jest także powrót do systemu operacyjnego komputera nadzorczego. Ogólny schemat blokowy programu przedstawia rys. 2.

Zasady współpracy programów Master-Multi i Master będą zamieszczone w punkcie 4.3. W tym punkcie główny nacisk został położony na przedstawienie programu Master-Multi od strony użytkownika i jego współpracy z systemem MS DOS.

Program sterujący całym systemem wieloprocesorowym został napisany z zastosowaniem techniki rozwijanego menu okienkowego. Każde z okien odpowiada jednej funkcji lub ich grupie. Po poprawnej inicjalizacji programu (brak sygnalizacji błędów) mamy do dyspozycji menu główne, a w nim cztery opcje. Są to:

File Transmit Run Options



Rys. 2. Ogólny schemat blokowy programu Master-Multi
Fig. 2. General block diagram of the Master-Multi programme

Zasadą przyjętą przy tworzeniu tego programu jest, że wyboru danej opcji dokonać można przez naciśnięcie pierwszej litery (małej lub dużej) słowa opisującego opcję lub przez naprowadzenie podświetlonego prostokąta na właściwą opcję i naciśnięcie klawisza Enter. Wycofanie się z realizacji danej opcji następuje przez naciśnięcie klawisza Esc. Akceptacja opcji lub parametrów ma miejsce przez naciśnięcie Enter. W przypadku sytuacji wyboru jako domniemany jest przypadek dający ewentualne najmniejsze szkody (jest on wyświetlany jako podświetlony lub w nawiasach kwadratowych "[ ]"). Zakończenie wykonywania programu z poziomu menu głównego następuje po naciśnięciu kombinacji klawiszy Alt-X.

Program zawsze proponuje standardowe rozszerzenia zbiorów:

cfg - zbiór zawierający dane konfiguracyjne;

dat - zbiór z danymi;

pro - zbiór z programem;

res - zbiór z wynikami;

Opcja File

Opcja ta zapewnia obsługę zbiorów dyskowych w systemie operacyjnym MS DOS. Możliwe są tu następujące opcje:

Open daje możliwość odczytu zbioru z danymi lub programem do bufora w pamięci komputera sterującego. **Save** pozwala zapamiętać w zbiorze dyskowym wyniki obliczeń przetransmitowane wcześniej do bufora w pamięci systemu nadzorczego.

Opcja **Change dir** zapewnia możliwość zmiany ścieżki dostępu do zbiorów na dysku.

Dos shell umożliwia wykonanie operacji w systemie operacyjnym. Powrót z systemu operacyjnego do programu następuje po wydaniu polecenia <**Exit**>.

Opcja **Exit** umożliwia zakończenie wykonywania programu Master-Multi i powrót do systemu operacyjnego.

Opcja Transmit

Opcja ta pozwala na transmisję programu (**Program**) i danych (**Data**) do procesora Master oraz rezultatów (**Results**) z tego procesora. W transmisji wykorzystywane są dane zapamiętane w buforze pamięci komputera nadzorczego.

Przy transmisji każdego z bloków program wymaga podania adresu początku i końca transmitowanego bloku. Każdy z adresów składa się z dwóch części: adresu bloku (segment) i adresu komórki w segmencie (offset). Szczególnie ważne jest właściwe podanie adresów bloku

z programem. W przypadku błędu grozi to utratą kontroli programowej nad systemem wieloprocesorowym. System nie posiada niestety żadnej ochrony sprzętowej pamięci.

Opcja Run

W tej opcji istnieje możliwość uruchomienia programu aplikacyjnego od zadanego adresu. Podobnie jak dla transmisji podawany jest on jako segment i offset. Należy zwrócić szczególną uwagę na deklarowany adres startu programu. Dodatkową możliwością tej opcji jest pomiar czasu wykonywania programu. Pomiar ten jest wykonywany z dokładnością do 55 ms, gdyż z taką dokładnością jest liczony czas systemowy w komputerach IBM PC.

Opcja Options

Najważniejszą z podopcji tego menu jest możliwość odczytania stanu systemu wieloprocesorowego. W jego rezultacie dostajemy listę aktywnych procesorów w systemie.

Pozostałe możliwości tego menu to ustawienie opcji automatycznego zapisu zbioru z wynikami przy kończeniu pracy z programem Master-Multi. Dodatkowo możliwe jest zapisanie lub odczytanie zbioru z konfiguracją. W zbiorze tym przechowywane są m.in. nazwy zbiorów, ścieżka dostępu oraz adresy bloków i adres startu. Jest to szczególnie istotna opcja przy kilku użytkownikach lub większej liczbie programów obliczeniowych.

4. Program Master

4.1. Miejsce programu w systemie

Program Master nazwać można jądrem programowym systemu wieloprocesorowego. Pozwala on na komunikację systemu nadzorczego z procesorami Slave. Zawiera również standardowe procedury arytmetyczne (UINT) umożliwiające wykorzystanie procesora Master jako jeszcze jednego procesora obliczeniowego.

4.2. Opis programu Master

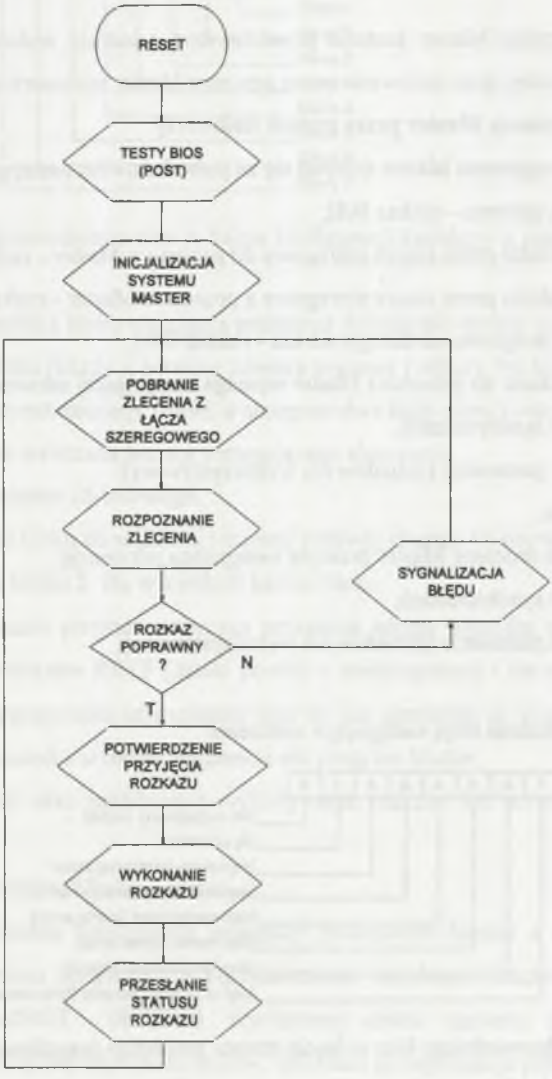
Podstawowe funkcje programu

Funkcje programu Master można podzielić na dwie grupy:

1. komunikacja z systemem nadzorczym i procesorami Slave,
2. funkcje arytmetyczne.

O ile drugą grupę funkcji można uznać za oczywistą, to pierwsza wymaga nieco komentarza. Po impulsie RESET (i poprawnym przejściu testów BIOS'u) zadaniem procesora Master

jest zainicjowanie procesorów Slave, łącza szeregowego oraz podstawowych zmiennych. Następnie procesor oczekuje na zlecenia z systemu nadzorczego. Po odebraniu zlecenia Master rozpoznaje je i wykonuje. Zakończenie wykonania zadania polega na przesłaniu przez procesor Master informacji do systemu nadzorczego. Informacja ta jest jednocześnie sygnałem, że Master oczekuje na kolejne zadanie.



Rys. 3. Uproszczony schemat blokowy programu Master
Fig. 3. Simplified block diagram of the Master programme

Uproszczony schemat blokowy programu Master

Schemat blokowy programu Master przedstawiono na rys. 3. Realizuje on wszystkie funkcje opisane wcześniej.

4.3. Obsługa programu Master

Obsługa programu Master zostanie przedstawiona z punktu widzenia jego sterowania przez system nadzorczy oraz sterowania przez procesor Master procesorów Slave.

Sterowanie programem Master przez system nadzorczy

Sterowanie programem Master odbywa się za pośrednictwem następujących rozkazów:

1. inicjalizacja systemu - rozkaz 00H;
2. przesłanie bloku przez złącze szeregowe do procesora Master - rozkaz 01H;
3. przesłanie bloku przez złącze szeregowe z procesora Master - rozkaz 02H;
4. wykonanie programu od danego adresu - rozkaz 03H.

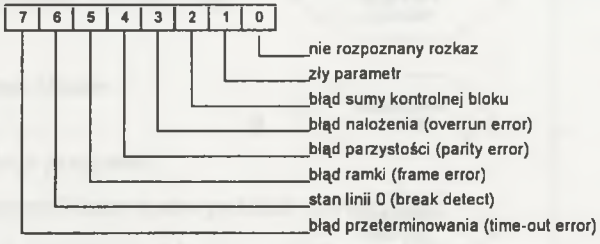
Przesłanie rozkazu do procesora Master wymaga następującej sekwencji:

1. 0E6H - bajt synchronizacji;
2. 0FFH - bajt parametru (aktualnie nie wykorzystywany);
3. bajt rozkazu.

W odpowiedzi procesor Master przesyła następującą sekwencję:

1. 0E6H - bajt synchronizacji;
2. 0FFH - bajt parametru (aktualnie nie wykorzystywany);
3. bajt status.

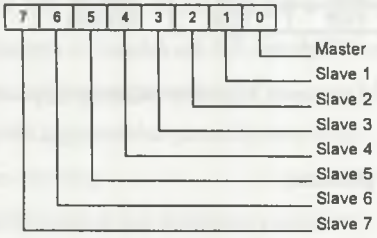
Bity w bajcie statusu mają następujące znaczenie:



Ustawienie odpowiedniego bitu w bajcie statusu powoduje sygnalizację błędu. Status 00H świadczy o braku błędów.

Wykonanie rozkazu inicjalizacji systemu polega na wyzerowaniu obszaru tablic parametrów procesorów Slave, a następnie stwierdzeniu, które z procesorów Slave są podłączone do systemu. Jako rezultat inicjalizacji Master przesyła bajt konfiguracji systemu oraz bajt zawierający liczbę procesorów w systemie.

Znaczenie bitów w bajcie konfiguracji jest następujące:



Ustawienie odpowiedniego bitu w bajcie konfiguracji świadczy o podłączeniu procesora do systemu.

Rozkazy przesyłania bloku wymagają przesłania dodatkowo dwóch parametrów, adresów początku i końca bloku (każdy z adresów zawiera segment i offset). Na końcu każdego bloku przesyłany jest bajt synchronizacji, 0E6H, a następnie dwa bajty sumy kontrolnej.

Suma kontrolna wyliczana jest wg następującego algorytmu:

1: zerowanie rejestru 16-bitowego;

2. dodanie bajtu bloku do młodszej i starszej połówki rejestru 16-bitowego;

3. powtórzenie kroku 2. dla wszystkich bajtów bloku.

Rozkaz wykonania programu wymaga przesłania adresu początku programu. Program musi kończyć się rozkazem RETF (daleki powrót z podprogramu) i nie może ustawiać własnego stosu. W tym przypadku adresowanie typu far jest niezbędne ze względu na umieszczenie programu użytkownika w innym segmencie niż program Master.

Każdy parametr oraz zakończenie wykonywania rozkazu jest potwierdzane sekwencją statusu.

Współpraca procesorów Master i Slave

Podstawową zasadę komunikacji pomiędzy procesorem Master a danym procesorem Slave stanowi wymiana informacji za pośrednictwem wspólnego obszaru pamięci globalnej (adresy liniowe 0A000H - 0EFFFH). Wyróżniony obszar nazwano tablicą komunikacji (TACOM). Długość tablicy wynosi 10 bajtów, natomiast jej organizacja jest następująca:

TACOM + 14	rezerwa	rezerwa	TACOM + 15
TACOM + 12	DI		TACOM + 13
TACOM + 10	SI		TACOM + 11
TACOM + 8	BL	BH	TACOM + 9
TACOM + 6	CL	CH	TACOM + 7
TACOM + 4	DL	DH	TACOM + 5
TACOM + 2	AL.	AH	TACOM + 3
TACOM + 0	PAR	FUN	TACOM + 1

Za pośrednictwem tablicy TACOM procesor Master przekazuje odpowiedniemu procesorowi Slave następujące dane:

1. adresy argumentów w pamięci globalnej;
2. długości argumentów;
3. kod operacji, jaka ma zostać wykonana;
4. parametr określający ewentualne dodatkowe czynności.

W trakcie realizacji zadania Slave umieszcza w odpowiadającej mu tablicy TACOM następujące informacje:

1. znacznik zakończenia obliczeń;
2. adres wyniku;
3. status wyniku (znak, ewentualny nadmiar itp.).

Polecenie rozpoczęcia wykonywania zadania jest przekazywane przez Master drogą programową. Wzbudza on odpowiedni bit w rejestrze przerwań. Operację tę realizuje następująca sekwencja rozkazów:

```
MOV     DX,INTR
MOV     AL,MASKA
OUT     DX,AL
```

Ustawienie w rejestrze przerwań bitu *i* powoduje wysłanie sygnału przerwania do *i-tego* procesora. Jak widać, możliwe jest w ten sposób równoczesne pobudzenie wszystkich procesorów Slave w systemie. Po przyjęciu przerwania dany procesor podrzędny wysyła sygnał potwierdzenia, kasujący odpowiadające mu zgłoszenie w rejestrze przerwań. Rejestr umieszczony jest w przestrzeni adresowej globalnych urządzeń we/wy. Tak więc, oprócz modułu Master, każdy procesor Slave posiada możliwości programowej zmiany jego zawartości. Dany Slave mógłby więc kasować programowo odpowiadające mu zgłoszenie przerwania. Ponieważ jednak nie ma możliwości odczytu zawartości rejestru, niemożliwe jest wyzerowanie określonego bitu bez zmiany bitów pozostałych. Przyjęto więc zasadę, że programowy dostęp do rejestru

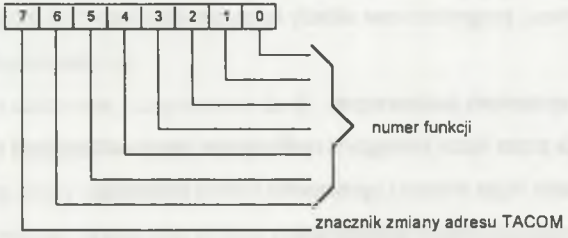
przerwań wykorzystuje jedynie procesor Master podczas zlecania zadań, natomiast procesory Slave kasują określone bity rejestru sprzętowo.

Format tablicy TACOM

Poszczególne pola tablicy mają następujące znaczenie:

1. BX - adres pierwszego argumentu; ustalono, że w polu pierwszego argumentu umieszczany jest wynik; w sytuacji gdy z pewnych względów nie spełnia tej reguły, Slave umieszcza w komórkach BX zmodyfikowany adres wyniku;
2. CX - adres drugiego argumentu;
3. DX - długość argumentów pomniejszona o 1;
4. AX - status wyniku;
5. SI - wskaźnik danych lub dodatkowy parametr;
6. DI - wskaźnik danych lub dodatkowy parametr;
6. PAR - definiuje czynności dodatkowe procesora Slave;
7. FUN - określa kod zadania zleconego Slave'owi.

Format bajtu FUN ma postać:



Kody funkcji wykonywanych przez procesory Slave:

0- potwierdzenie sprawności procesora;

1- pobranie i wykonanie programu;

2- bezpośrednie wykonanie funkcji wywoływanej przez INT60;

3- funkcja testowa (INT 61H) - odtwarzająca prostą sekwencję dźwięków.

Program załadowany przez system nadzorczy może korzystać z procedur standardowych programu Master i Slave (funkcje wywoływane przez INT 60H).

4.4. Opis techniczny programu Master

Program Master realizuje następujące funkcje:

1. inicjalizację systemu;
2. komunikację z systemem nadzorczym;
3. komunikację z procesorami Slave;
4. działania arytmetyczne na liczbach wielokrotnej precyzji.

Organizacja pamięci

Procesor Master posiada 64 kB pamięci EPROM i 640 kB pamięci RAM. Do realizacji podstawowych funkcji programu Master przeznaczony został obszar zmiennych roboczych pod adresem 0040:0000 do 0040:FFFF. Obszar ten zawiera zmienne systemowe i komórki robocze procedur arytmetycznych. Pozostały obszar pamięci RAM może być wykorzystany do ładowania programów użytkownika.

Inicjalizacja systemu

Po impulsie RESET wykonywane są standardowe testy BIOS (POST), następnie ustawiany jest stos programu, programowane układy łącza szeregowego oraz inicjalizowane procesory Slave.

Komunikacja z systemem nadzorczym

Komunikacja przez łącze szeregowe realizowana jest w następującej sekwencji:

1. wyzerowanie bajtu statusu i opróżnienie bufora transmisji;
2. pobranie rozkazu z łącza szeregowego;
3. rozpoznanie rozkazu;
4. przygotowanie do wykonania procedury realizującej rozkaz;
5. przesłanie statusu do systemu nadzorczego - potwierdzenie odebrania rozkazu;
6. wykonanie rozkazu;
7. przesłanie statusu wykonania rozkazu.

Komunikacja z procesorami Slave

Komunikacja z procesorami Slave realizowana jest za pośrednictwem tablic parametrów TACOM. Program wczytywany przez łącze szeregowe może wywoływać procedury standardowe programu Master, procedury wywoływane przez INT 60H.

Procedury arytmetyczne (INT 60H)

Procedura obsługi przerwania INT60h zawiera szereg podprogramów wykonujących operacje arytmetyczne i logiczne na liczbach dowolnej długości. Mogą to być zarówno liczby całkowite bez znaku, jak i ze znakiem oraz liczby rzeczywiste w zapisie stało- i zmiennoprzecinkowym. Aby wykonać żądaną operację, należy umieścić w odpowiednich rejestrach procesora adresy argumentów, ich długość, w niektórych przypadkach adres, pod który należy przesłać wynik. Po wykonaniu tych czynności należy w rejestrze AH umieścić kod operacji, a następnie wykonać rozkaz **int 60h.** Podprogram ten realizuje następujące operacje:

- przenoszenie argumentu w inne miejsce pamięci
- zerowanie argumentu
- negacja poszczególnych bitów argumentu
- dodawanie dwóch liczb całkowitych
- odejmowanie dwóch liczb całkowitych
- mnożenie dwóch liczb całkowitych
- dzielenie liczby całkowitej przez liczbę całkowitą (wynik: iloraz całkowity, reszta całkowita)
- dzielenie stałoprzecinkowe
- negacja liczby całkowitej (uzupełnienie do 2)
- inkrementacja liczby całkowitej
- dekrementacja liczby całkowitej
- przesunięcie jednego argumentu w lewo o liczbę bitów zawartą w drugim argumencie
- przesunięcie jednego argumentu w prawo o liczbę bitów zawartą w drugim argumencie
- dodawanie dwóch liczb zmiennoprzecinkowych
- odejmowanie dwóch liczb zmiennoprzecinkowych
- mnożenie dwóch liczb zmiennoprzecinkowych
- dzielenie dwóch liczb zmiennoprzecinkowych
- inkrementacja liczby zmiennoprzecinkowej
- dekrementacja liczby zmiennoprzecinkowej
- pierwiastek liczby zmiennoprzecinkowej.

5. Program Slave

5.1. Miejsce programu w systemie

Program Slave jest najniższym w hierarchii programem systemu. Zawiera standardowe procedury numeryczne. Wymaga dostarczenia danych przez procesor Master. Pewną "autonomię" zachowuje po włączeniu zasilania podczas testów BIOS (POST).

5.2. Opis programu Slave

Podstawowe funkcje programu

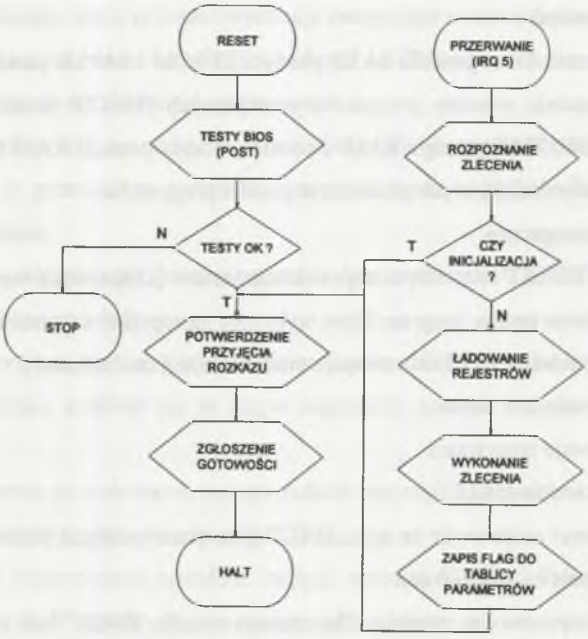
Funkcje programu Slave można podzielić na dwie grupy:

1. komunikacja z procesorem Master,
2. funkcje arytmetyczne.

Po odebraniu zlecenia procesor Slave rozpoznaje je, ładuje dane do odpowiednich rejestrów i rozpoczyna wykonanie zleconego zadania. Po wykonaniu zadania procesor Slave informuje procesor Master o wykonaniu zadania i oczekuje na kolejne zadanie.

Uproszczony schemat blokowy programu

Po włączeniu zasilania generowany jest sygnał RESET, procesor przeprowadza testy (BIOS - POST). W przypadku pozytywnego zakończenia testów ustawiane są zmienne programu, zgłaszana jest gotowość procesora Slave do wykonania zadania. Następnie procesor oczekuje na następne zadanie, przechodząc w stan HALT. Procesor Master informuje procesor Slave o nowym zadaniu, generując przerwanie 5. Program obsługi tego przerwania jest umieszczony w 1kB pamięci w tablicy, która jest inicjalizowana przez BIOS podczas testów. Procedura obsługi przerwania w pierwszym rzędzie rozpoznaje zlecenie. W przypadku wykrycia zlecenia inicjalizacji przeprowadzane jest działanie jak po poprawnym przejściu testów BIOS'u. Po rozpoznaniu innego polecenia następuje ładowanie rejestrów procesora zawartością tablicy parametrów i wykonanie polecenia. Po zakończeniu zadania flagi procesora zapamiętywane są w tablicy parametrów. Następnie ma miejsce ustawienie zmiennych oraz zgłoszenie gotowości procesora do wykonania następnego zadania. Procesor przechodzi w stan oczekiwania na następne zadanie.



Rys. 4. Uproszczony schemat blokowy programu Slave
Fig. 4. Simplified block diagram of the Slave programme

5.3. Obsługa programu Slave

Program Slave korzysta z danych umieszczonych w tablicy parametrów, której adres przekazywany jest przez procesor Master jako pierwsze zlecenie dla procesora Slave. Format tablicy parametrów (TACOM) zamieszczono w punkcie 4.3.

5.4. Opis techniczny programu Slave

Program Slave realizuje następujące funkcje:

1. testy (BIOS - POST)
2. inicjalizację wewnętrzną
2. inicjalizację zewnętrzną
4. pobranie tablicy TACOM
5. działania arytmetyczne na liczbach wielokrotnej precyzji.

Organizacja pamięci

Każdy procesor Slave posiada 64 kB pamięci EPROM i 640 kB pamięci RAM. Zmienne systemowe i komórki robocze procedur arytmetycznych (UINT) zawarte są w segmencie 0040:0000 do 0040:FFFF pamięci RAM. Pozostały obszar pamięci RAM może być wykorzystany do ładowania dodatkowych procedur w postaci programów.

Inicjalizacja wewnętrzna

Po sygnale RESET rozpoczyna się realizacja testów (adaptowana wersja BIOS). Po pomyślnym wykonaniu testów program Slave wykonuje następujące czynności:

1. zeruje komórkę FIRST stanowiącą semafor inicjacji zewnętrznej
2. ustawia wskaźnik stosu
3. odblokowuje przerwania
4. wchodzi w stan HALT.

Mikroprocesor może wyjść ze stanu HALT tylko przez przyjęcie przerwania lub RESET.

Inicjalizacja zerowa - zewnętrzna

Po przyjęciu przerwania procesor Slave testuje semafor FIRST. Jeśli ma on wartość 00H, rozpoczyna inicjację zerową. Polega ona na pobraniu adresu tablicy TACOM przygotowanego przez procesor Master i przepisaniu go do zmiennej systemowej ATAC. Adres tablicy znajduje się zawsze w ustalonym miejscu - są to dwie pierwsze komórki pamięci globalnej (0A000h:0000h). Od tej chwili zmiana adresu tablicy może być dokonana jedynie specjalnym formatem bajtu FUN. Następnie procesor zapisuje wartość 0FFH do zmiennej FIRST, zamykając semafor inicjacji zerowej. Dodatkowo wpisuje wartość 0FFH w polu FUN przydzielonej tablicy TACOM. Jest znacznik zakończenia pracy dla Master'a. W końcu zostaje odtworzona wartość wskaźnika stosu, odblokowane przerwania i procesor wchodzi w stan HALT.

Realizacja zadania

Każde następne przerwanie, po inicjacji zerowej, traktowane jest przez Slave jako zlecenie zadania. Wykonanie zadania składa się z następujących etapów:

1. czynności wstępne
2. wywołanie odpowiedniej procedury
3. odblokowanie przerwań i wejście w stan HALT.

Podczas pierwszego etapu procesor wykonuje następujące czynności:

1. Ustawia wskaźnik stosu na początek tablicy TACOM, pobiera dane z komórek PAR i FUN i przepisuje do zmiennych systemowych.
2. Analizuje zawartość komórki FUN. Jeśli jest to funkcja zerowa, następuje inicjalizacja procesora, w przeciwnym przypadku przygotowywane są zmienne dla wykonania zleconego zadania.
3. Pobiera do właściwych rejestrów pozostałe dane z tablicy TACOM. Następnie odtwarza stos w pamięci lokalnej.

Wywołany program (zadanie) kończy się rozkazem dalekiego powrotu z podprogramu (program użytkownika znajduje się w innym segmencie pamięci niż procedury programu Slave).

Obsługa końcowa po wykonaniu danego zadania obejmuje następujące czynności:

1. zapisanie statusu wyniku do tablicy TACOM;
2. opcjonalne wyzerowanie semafora inicjacji zerowej FIRST, jeśli bit 7 komórki FUM był równy 1. Pozwala to na zmianę przez Master adresu przydzielonej tablicy TACOM;
3. ustawienie znacznika końca pracy w polu FUN (0FFH);
4. reinicjalizacja wskaźnika stosu, odblokowanie przerwań, przejście w stan HALT.

6. Wnioski

Przy tworzeniu oprogramowania dla systemu wieloprocesorowego autor starał się śledzić aktualne tendencje w tworzeniu oprogramowania wspomagającego programowanie systemów wieloprocesorowych. Wydają się one dążyć do symetrycznej wieloprocesorowości bazującej na wieloprocesorowej odmianie systemu UNIX. Z punktu widzenia realizowanych obliczeń przedstawiony system realizuje zadania z wykorzystaniem symetrycznej wieloprocesorowości (SMP), chociaż jako całość można go traktować jako asymetryczny (ASMP).

Zalety takiego rozwiązania to duża wydajność, szczególnie przy zastosowaniu najnowszych wersji procesora firmy Intel - Pentium. Powszechność zastosowania procesorów tej firmy powoduje znaczną redukcję kosztów - istnieje możliwość zastosowania standardowych modułów. Komputer może być stosunkowo łatwo rozbudowywalny, co również ma niebagatelne znaczenie. Ważnym atutem takiego rozwiązania jest również uniwersalność systemu UNIX. Bogata biblioteka programów i narzędzi programistycznych powoduje, że rozwiązanie

to może stać się standardem w przyszłości. Świadczyć może o tym również fakt pojawienia się wiosną 1995 systemu UNIX firmy Novell (handlowa nazwa systemu - UNIXWare) będącej jednym z największych twórców oprogramowania, szczególnie sieciowego i systemowego. W 1996 pojawiła się wersja wieloprocesorowa systemu NetWare 4.1x firmy Novell i nowa wersja systemu Windows NT 4.0 firmy Microsoft. Obydwa systemy pozwalają na pracę z wykorzystaniem symetrycznej wieloprocesorowości.

Interesujące jest też podejście do sieci komputerowych (szczególnie tych wykorzystujących szybkie protokoły) jako do rozproszonych systemów, wieloprocesorowych. Użycie tych systemów, w sytuacjach gdy ilość transmitowanej informacji nie jest zbyt duża, może być korzystne. Zaletą tych systemów jest oprócz ich uniwersalności i bardzo łatwej konfigurowalności prosta (tania) konstrukcja. W większości są to jednoprocesorowe komputery. Ponieważ systemy te są stosunkowo luźno ze sobą powiązane, wzrasta ich niezawodność. Dodatkowo możliwe jest wykorzystanie w jednym wirtualnym systemie wieloprocesorowym komputerów pracujących na innych typach procesorów i wykorzystujących różne systemy operacyjne. Możliwe staje się więc zbudowanie prawdziwie heterogenicznego systemu wieloprocesorowego.

Oprogramowanie komputera nadrzędnego systemu wieloprocesorowego powinno zatem umożliwiać stosunkowo łatwe przejście całego systemu na docelowy system operacyjny. Jedynym wyborem pozostaje więc napisane oprogramowanie realizujące środowisko programowe do programowania całego systemu z możliwością nadzoru pracy całości, jak i tworzenia własnych algorytmów sterowania przydziałem zadań dla poszczególnych zadań. Ze względu na możliwość przejścia w przyszłości na system operacyjny pochodny UNIX naturalne wydaje się wybranie języka C do stworzenia tego oprogramowania.

Autor przedstawił aktualny stan swoich prac nad oprogramowaniem systemu wieloprocesorowego. Dalszy kierunek badań to próba implementacji licznych algorytmów szeregowania zadań z uwzględnieniem specyfiki tego systemu wieloprocesorowego.

LITERATURA

1. Borzemski L.: *Wybrane problemy równoważenia obciążenia w systemach rozproszonych.* Wydawnictwo Politechniki Wrocławskiej, 1989.
2. Enslow P.H. Jr., (red.): *Systemy cyfrowe wieloprocesorowe.* WNT.Warszawa 1978.
3. Fisher M.J., Griffith N.D., Lunch N.A: *Global states of a distributed systems* IEEE Trans. on Software Eng., Vol. SE-8, No 3, May, 198-202 1982.
4. Fountain T.J., Shute M.J.: *Multiprocessor Computer Architecture.* Elsevier Science Publishers B.V., 1990.
5. Flynn M.J.:*Very high-speed computing systems Proc.* IEEE, Vol. 54, No 12 1966.
6.Gelenbe E.:*Multiprocessor Performance.* John Wiley & Sons, 1989.
7. Kaniewski M., Wieremiejczyk K.: *Aplikacje rozproszone w praktyce.* Unix Forum 2/94.
8. Kaniewski M., Wieremiejczyk K.: *Mainframe a systemy otwarte.* Unix Forum 1/94.
9. Kruskal C.P., Madej T., Rudolph L.: *Parallel prefix on fully connected direct connection machines Proc. Int. Conf.* Parallel Processing, IEEE 1986.
10. Kucera L.: *Parallel computation and conflicts in memory access Inf.* Proc. Letters, Vol. 14, No 2, April, 93-96 1982.
11. Lewis G.T., El Rewini H.: *Introduction to parallel Computing.* Prentince-Hall, 1992.
12. Liebowitz B.H., Carson J.H.: *Multiple processor systems for real-time applications.* Prentice-Hall, 1985.
13. Milter R.K.: *Parallel processing.* The Fairmont Press, Inc., 1990.
14. Paker Y.: *Multi-microprocessor systems.* Academic Press, Inc., 1983.
15. Quinn M.J.: *Designing Efficient Algorithms for Parallel Computers McGraw-Holl.* Singapore 1988.
16. Santosh G.A.: *Reducing interprocessor communication in parallel architectures: system configuration and assignment, center of supercomputing research and development Rpt.* No 726, University of Illinois at Urbana Champaign, January 1988.
17. Sysło M.M.: *Maszyny MIMD i SIMD.* Informatyka, nr 11-12 1988.
18. Virhkin U.: *Implementation of simultaneous memory address access in models that forbit if Journal of Algorithms* 4, 45-50 1983.

Recenzent: Prof.dr hab.inż. Adam Mrózek

Wpłynęło do Redakcji 5.06.1997 r.

Abstract

The article presents software of the multiprocessor system with the modular construction based on the Intel 20286 microprocessor. The software is divided hierarchically into three levels: the software of the supervising system, Master-Multi, the software of the distinguished processor - Master and the other processors - Slave. The structure of consecutive software blocks is discussed. The technical description of Master and Slave programs are presented as well as the list of the used mathematical and numerical procedures. The article presents also the most important conclusions concerning with programming multiprocessor system. The described solutions have been designed and implemented at the Division of Digital Equipment and Microprocessors of the Institute of Electronics of the Silesian Technical University in Gliwice.