ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1999
Seria: INFORMATYKA z. 36 Nr kol. 1414

Ryszard PAWŁOWSKI
Politechnika Śląska, Instytut Informatyki

AUTONOMIA PRACY PODSIECI W HIERARCHICZNEJ SIECI POLOWEJ STEROWANEJ ROZKAZAMI JĘZYKA SCPI*)

Streszczenie. W publikacji przedstawiono mechanizmy autonomicznej pracy podsieci w ramach hierarchicznej sieci polowej sterowanej rozkazami języka SCPI. Rolę stacji sterującej wyzwalaniem pomiarów pełni stacja master lokalny. W stacji tej zaimplementowano mechanizm zewnętrznej zmiennej stanu będącej podstawą programu cyklicznego odpytywania stacji slave. Wewnętrzne zmienne stanu umożliwiają dostęp do podstawowych parametrów sterujących pracą stacji master.

AUTONOMOUS SUBNET OPERATION IN THE HIERARCHICAL FIELDBUS CONTROLLED BY SCPI COMMANDS

Summary. In the paper the algorithms of the autonomous subnet operation in the hierarchical fieldbus controlled by SCPI commands are presented. The master station controls release of the measurments. In this station the mechanizm of the external state variable was implemented. These variables are the base of a cyclic data reading from the slave stations. Internal state variables control the basic parameters of the master station.

1. Wprowadzenie

Niniejsza publikacja przedstawia mechanizmy autonomicznej pracy podsieci w hierarchicznej sieci polowej sterowanej rozkazami języka SCPI (Standard Commands for Programmable Instruments) [4]. Koncepcję pracy sieci hierarchicznej, przedstawioną w publikacji [1], rozwinięto w pracach dotyczących stacji slave [2] i stacji master [3]. Stacja

*) Praca finansowana przez KBN w ramach projektu badawczego 302/T11/97/12

master lokalny, będąca kluczowym elementem umożliwiającym komunikację w hierarchicznej sieci polowej, stała się także najważniejszym elementem mechanizmu umożliwiającego realizację autonomicznej pracy podsieci.

2. Zmienna stanu

Zmienna stanu jest obiektem, z którym związane są *wartość* oraz *akcja*. Pierwotnie mechanizm zmiennej stanu planowano zaimplementować przede wszystkim w stacji slave. Pozwalał on "zasłonić" przed aplikacją sterującą pracą całej sieci rzeczywisty przyrząd pomiarowy z jego własnym językiem sterowania. Aplikacja sterująca odwołuje się do wartości mierzonej przez przyrząd pomiarowy poprzez odwołanie do abstrakcyjnego obiektu, jakim jest zmienna stanu zdefiniowana w stacji slave. Realizowane jest to poprzez rozkaz SCPI ":var". W przypadku operacji odczytu wartości zmiennej stanu rozkazem ":var<numer_zmiennej>?" (np. ":var123?") stacja slave wykonuje akcję związaną z daną zmienną. Może nią być np. szereg poleceń wysłanych do przyrządu pomiarowego (w jego własnym języku programowania) powodujący wyzwolenie pomiaru określonej wielkości fizycznej i odesłanie wyniku do stacji slave. Odpowiedź przyrządu pomiarowego powinna zostać przekonwertowana do postaci ciągu znaków ASCII rozprezentujących wynik pomiaru, a następnie zapisana w polu przeznaczonym na wartość związaną z daną zmienną. Ostatecznie zawartość tego pola powinna zostać odesłana jako wynik wykonania polecenia ":var". W przypadku operacji zapisu wartości do zmiennej stanu realizowanej rozkazem ":var<numer_zmiennej> <wartość_zmiennej>" (np. ":var123 24.5") ciąg znaków reprezentujący nową wartość zmiennej jest zapisywany do pola przeznaczonego na wartość danej zmiennej, a następnie stacja slave wykonuje akcję związaną z tą zmienną. W tym przypadku zazwyczaj będzie to konwersja wartości zmiennej do formatu akceptowanego przez obsługiwany przyrząd pomiarowy oraz wysłanie poleceń do przyrządu pomiarowego zmieniających np. zakres pomiarowy zgodnie z nową wartością zmiennej stanu.

Jednak już w pierwszej publikacji dotyczącej hierarchicznej sieci polowej [1] zasygnalizowano możliwość rozszerzenia zestawu operacji dotyczących tych zmiennych o operację wyzwolenia pomiaru. Zgodnie z tym pomysłem przedstawiona powyżej operacja odczytu wartości zmiennej stanu mogłaby zostać zrealizowana w dwóch krokach: wyzwolenie pomiaru danej zmiennej (":var123") i właściwy odczyt wartości (":var123?"). Wyzwolenie pomiaru wartości zmiennej wiązałoby się tylko z realizacją akcji związanej z tą zmienną i zapisaniem wyniku pomiaru otrzymanego od przyrządu pomiarowego do pola przeznaczonego na wartość danej zmiennej stanu. Z kolei operacja odczytu wartości zmiennej nie wyzwalałaby akcji, a tylko odczytywała przygotowaną wartość, która następnie powinna

zostać odesłana jako wynik wykonania rozkazu SCPI. Ta możliwość oczywiście nie wyklucza implementacji zmiennej stanu według algorytmu opisanego poprzednio, tzn. istnienia zmiennych, dla których operacja wyzwolenia pomiaru byłaby ignorowana, natomiast operacja odczytu wiązałaby się z realizacją akcji i odesłaniem wartości zmiennej w pojedynczej transakcji. Jak łatwo zauważyć, różnice pomiędzy tymi realizacjami są NIEWIDOCZNE dla stacji na wyższych poziomach hierarchii, a także możliwe jest istnienie (nawet w ramach jednej stacji) zmiennych stanu zrealizowanych według obu algorytmów.

Analizując algorytm transmisji rozkazu SCPI, a następnie odpowiedzi pomiędzy stacjami master i slave, która odbywa się poprzez realizację szeregu transakcji w protokole ModBUS [1], łatwo zauważyć, że transmisja rozkazu blokuje zaangażowane w nią stacje: stację master będącą źródłem rozkazu, stację slave będącą odbiorcą rozkazu oraz wszystkie stacje master będące pośrednikami przy transmisji danego rozkazu SCPI. W przypadku długotrwałego pomiaru przez przyrząd pomiarowy wartości fizycznej (np. częstotliwości) czas realizacji pomiaru mógłby powodować niebezpieczne w skutkach zablokowanie fragmentu sieci. Realizacja dostępu do wartości zmiennych stanu reprezentujących wielkości fizyczne o długim czasie pomiaru poprzez zastosowanie dwustopniowego dostępu zlikwiduje ten problem. Jedna, długotrwała transakcja zostanie wtedy zastąpiona dwoma niezależnymi transakcjami, których czas trwania jest o wiele krótszy. Pomiędzy pierwszą transakcją, w której wysłany zostanie rozkaz wyzwalający pomiar, a drugą, w której nastąpi odczyt wartości zmiennej, może upłynąć teoretycznie dowolnie długi okres czasu, w którym przyrząd pomiarowy będzie realizował pomiar określonej wielkości fizycznej, a cała sieć niezablokowana będzie pracowała dalej, przesyłając kolejne rozkazy.

3. Autonomia pracy podsieci

Z realizacją operacji wyzwolenia pomiaru nierozerwalnie związany jest problem autonomii pracy podsieci i zmiennych stanu zdefiniowanych w stacji master. Jak łatwo zauważyć, pozytywne cechy dwuetapowej (wyzwolenie pomiaru, odczyt gotowego wyniku) operacji odczytu wartości zmiennej stanu można wykorzystać wtedy, gdy stacja wysyłająca rozkaz SCPI zna czas pomiaru określonej wielkości fizycznej. Trudno sobie wyobrazić, aby wszystkie te dane (wartości zmiennych stanu i czasy ich pomiaru) były centralnie gromadzone tylko w stacji master systemowy zarządzającej pracą całej sieci - o wiele efektywniejsze jest wyodrębnienie pewnych fragmentów sieci oraz powierzenie sterowania pomiarami w tych podsieciach stacjom lokalnym. Stacją, której powierzono rolę lokalnego zarządcy, jest stacja master lokalny. Zgodnie z pomysłem przedstawionym w pracy [1] zrealizowano w stacji

master lokalny moduł pracy lokalnej, którego najważniejsze elementy i algorytmy pracy przedstawiono w kolejnych punktach niniejszej pracy.

4. Zmienne stanu w stacji master

Zmienne stanu zdefiniowane w stacji master lokalny, mimo zewnętrznego podobieństwa do zmiennych stanu w stacji slave opisanych wyżej (identyczne mechanizmy dostępu, jednolite oznaczanie), różnią się sposobem realizacji poszczególnych operacji.

W związku w istnieniem mechanizmu autonomicznej pracy podsieci część tych zmiennych będzie "przechowywała" wartość odczytaną z podległej stacji slave, udostępniając ją w sposób ciągły i zasłaniając (przed innymi stacjami master na wyższych poziomach hierarchii) jej rzeczywiste źródło. Realizacja cyklicznych odczytów ze stacji slave wartości zdefiniowanej tam zmiennej może dostosować się do znanego czasu pomiaru danej zmiennej i zastosować opisany wcześniej dwuetapowy cykl dostępu do wartości zmiennej. Oczywiście, istnienie zmiennych cyklicznych nie wyklucza możliwości implementacji także zmiennych aperiodycznych, których pomiar wyzwalany jest na żądanie. Możliwe jest także istnienie zmiennych przeznaczonych "do zapisu". Mogą one być związane ze zmiennymi w stacjach slave, które są przeznaczone do sterowania parametrami przyrządu pomiarowego (np. zakres pomiarowy). Reasumując, mamy tutaj do czynienia ze zmiennymi reprezentującymi obiekty (zmienne) w podległych stacjach slave (lub master), a więc z obiektami zewnętrznymi - w dalszej części opracowania będzie używane określenie *zmienna zewnętrzna*. Tego rodzaju zmienne będą wymagały określenia czasu cyklu, czasu pomiaru, kierunku transmisji danych (do lub ze stacji slave) oraz polecenia SCPI umożliwiającego dostęp do zmiennej zdefiniowanej w stacji slave.

Oprócz przedstawionych powyżej zmiennych zewnętrznych w stacji master zaimplementowano *zmienne wewnętrzne*. Mechanizm związany z tymi zmiennymi umożliwia dostęp w zunifikowany sposób do pewnych parametrów pracy stacji master. Jako przykład można podać zmienną stanu, która zawiera czas przeterminowania (timeoutu) dla transakcji ModBUS'a realizowanych przez moduł TRAN. Ze zmienną tego typu może być związana wartość (np. zawartość określonej komórki pamięci stacji master) lub wartość ta może być obliczana "w locie" - w momencie dostępu do zmiennej. Przykładami mogą tu być zmienna reprezentująca timeout transakcji ModBUS'a i zmienna udostępniająca typ zmiennej stanu zdefiniowanej w stacji master o określonym numerze. Dla tego typu zmiennych należy określić tylko akcję związaną z daną zmienną.

5. Parametry charakteryzujące zmienne stanu

Jak zaznaczono wyżej, w stacji master możemy spotkać dwa rodzaje zmiennych: zmienne zewnętrzne reprezentujące obiekty (zmienne stanu) w podległych stacjach slave lub master oraz zmienne wewnętrzne reprezentujące obiekty wewnętrzne stacji master. Dla zmiennych wewnętrznych podstawowe parametry to:

numer_zmiennej numer zmiennej będący częścią rozkazu ":var<numer_zmiennej>";
stala1 ilość żądanej pamięci RAM (do 255 bajtów);
stala2 wartość stała (np. adres komórki pamięci) przekazywana procedurze wykonawczej;
kod_procedury kod maszynowy procedury dostępu do wartości zmiennej.

Podczas tworzenia aplikacji okazało się, że wiele zmiennych wewnętrznych jest podobnych. Przykładem mogą tu być zmienne reprezentujące czasy przeterminowania (timeouty) dla transakcji ModBUS'a z poleceniem SCPI i dla transakcji ModBUS'a typu PING (pakiet typu ECHO pozwalający stwierdzić, czy pod danym adresem ModBUS'a jest jakaś stacja). Jak łatwo zauważyć, są to dwie zmienne, które różnią się jedynie adresem komórki pamięci RAM przechowującej wartość. Aby ułatwić i uprościć tworzenie oprogramowania realizującego operacje na zmiennych, jako parametr stala2 można umieścić adres komórki RAM z wartością tej zmiennej. Procedura wykonawcza związana ze zmienną jest uniwersalna i umożliwia dostęp do komórki pamięci RAM o adresie wskazanym właśnie poprzez parametr stala2. Dotychczas przygotowano procedury realizujące dostęp do następujących zmiennych wewnętrznych:

- kod ostatniego błędu;
- timeout dla transakcji ModBUS'a typu PING;
- timeout dla transakcji ModBUS'a z poleceniem SCPI;
- wersja oprogramowania stacji master;
- nazwa stacji (Master_Lokalny_SCPI);
- zmienna umożliwiająca odczyt i modyfikację bajtu flag dla zmiennych zewnętrznych;
- operacje na fladze blokady dla dowolnej zmiennej zewnętrznej;
- odczyt typu zmiennej (wewnętrzna, zewnętrzna).

Dla zmiennych zewnętrznych podstawowe parametry opisujące daną zmienną to:

numer_zmiennej numer zmiennej będący częścią rozkazu ":var<numer_zmiennej>";
cykl_sek,cykl_min czas cyklu dla danej zmiennej: zapis <n,0> oznacza, że czas cyklu wynosi <n> sekund, zapis <0,n> oznacza czas cyklu <n> minut, natomiast <0,0>

oznacza zmienną asynchroniczą, dla której wyzwolenie pomiaru następuje na żądanie, dopuszczalne okresy wynoszą od 1 do 255 sekund (4 minuty i 15 sekund) dla zmiennych sekundowych i od 1 do 255 minut (4 godziny i 15 minut) dla zmiennych minutowych;

flagi — początkowe ustawienie flag:

<7> wyzwolenie pomiaru lub wymuszenie przepisania wartości do stacji slave;

<6> zmienna typu READ ONLY - próba zapisu wywołuje błąd;

<5> kierunek przesyłania wartości: 0 odczyt ze stacji slave, 1 zapis do stacji slave;

<4> flaga wystąpienia błędu komunikacyjnego podczas realizowania operacji ze stacją slave;

<0> blokada zmiennej;

(bity 1, 2 i 3 są niewykorzystane);

czas_pomiaru — czas pomiaru danej zmiennej w stacji podrzędnej, wartość 0 oznacza zmienną, której wartość dostępna jest natychmiast (transakcja w jednym kroku), wartość <n> oznacza, że dostęp do wartości zmiennej odbywa się w dwóch cyklach, pomiędzy którymi należy odczekać <n> sekund;

adres — adres w protokole ModBUS stacji podrzędnej, do której należy wysłać polecenie SCPI dostępu do wartości zmiennej stanu;

kod_rozkazu_SCPI — rozkaz dostępu do rzeczywistej wartości zmiennej stanu zapisany w postaci ciągu znaków ASCII reprezentujących rozkaz SCPI;

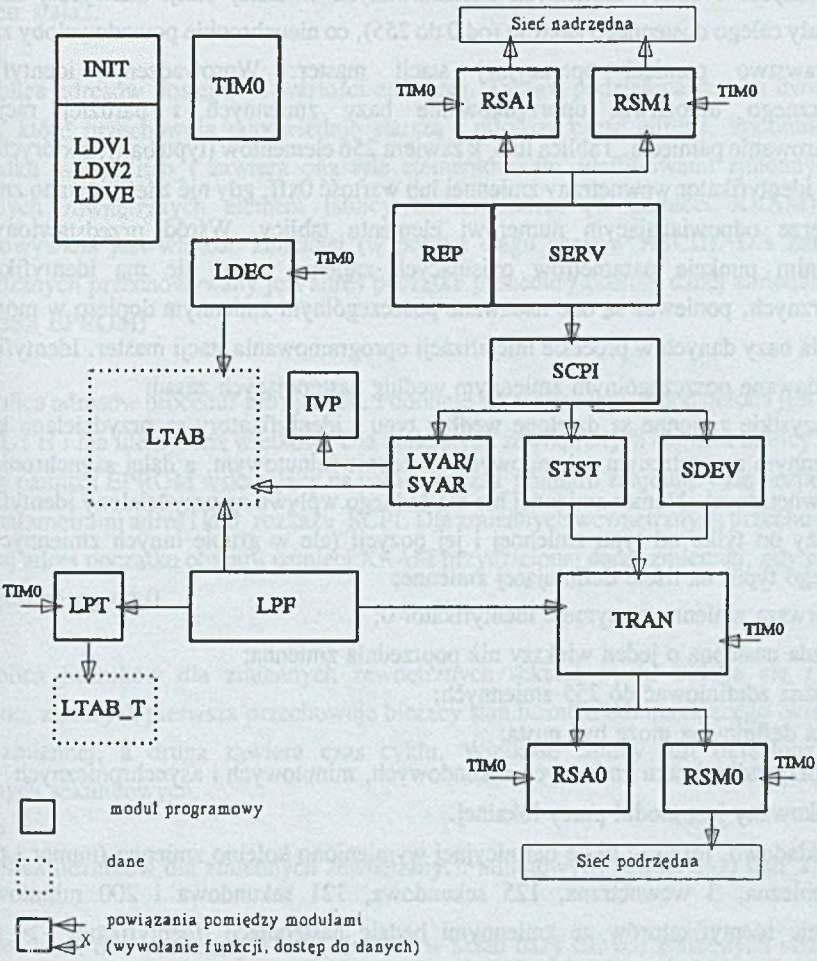
wzorzec — wzorzec wartości zmiennej umożliwiający ustalenie wartości początkowej zmiennej.

Parametry określające czas cyklu stanowią podstawę do podziału zmiennych zewnętrznych na: sekundowe, minutowe i asynchroniczne.

Opisane powyżej parametry należy określić dla każdej deklarowanej zmiennej, tworząc w ten sposób listę definiującą zmienne stanu dla określonej stacji master - jest to swego rodzaju program, według którego pracować będzie stacja master, odpytując stacje podległe o wartości poszczególnych zmiennych stanu.

6. Implementacja bazy danych zmiennych stanu w stacji master

Przedstawione w poprzednim punkcie parametry opisujące zmienne stanu zdefiniowane w stacji master stanowią podstawę do utworzenia bazy danych zmiennych stanu (rys. 1, obiekt LTAB). Znajduje się ona w pamięci zewnętrznej RAM mikrokontrolera 80C320



Rys. 1. Schemat blokowy oprogramowania stacji master
Fig. 1. Block diagram of the master station software

(pamięć XRAM). Baza ta tworzona jest w momencie inicjalizacji oprogramowania stacji master i składa się z kilku niezależnych tablic. Tablice te zostały wyodrębnione w ten sposób,

aby uprościć i przyspieszyć działanie modułów programowych operujących na zmiennych stanu. Najważniejsze tablice wchodzące w skład bazy to:

ltab_k

Tablica konwersji numeru zmiennej (element kodu rozkazu ":var" np. ":var124" oznacza zmienną o numerze 124) na identyfikator wykorzystywany przez oprogramowanie do celów wewnętrznych. Numery zmiennych zdefiniowanych w danej stacji zazwyczaj nie będą pokrywały całego dostępnego zakresu (od 0 do 255), co nieuchronnie powodowałoby znaczne marnotrawstwo pamięci operacyjnej stacji master. Wprowadzenie identyfikatora wewnętrznego umożliwia uporządkowanie bazy zmiennych i bardziej racjonalne gospodarowanie pamięcią. Tablica ltab_k zawiera 256 elementów (typu bajt), z których każdy zawiera identyfikator wewnętrzny zmiennej lub wartość 0xff, gdy nie zdefiniowano zmiennej o numerze odpowiadającym numerowi elementu tablicy. Wśród przedstawionych w poprzednim punkcie parametrów opisujących zmienne stanu nie ma identyfikatorów wewnętrznych, ponieważ są one nadawane poszczególnym zmiennym dopiero w momencie tworzenia bazy danych w procesie inicjalizacji oprogramowania stacji master. Identyfikatory te są nadawane poszczególnym zmiennym według następujących zasad:

- wszystkie zmienne są dzielone według typu i identyfikatory są przydzielane kolejno zmiennym zewnętrznym sekundowym, następnie minutowym, a dalej asynchronicznym i wewnętrznym. Numer zmiennej nie ma żadnego wpływu na przydzielony identyfikator. Zależy on tylko od typu zmiennej i jej pozycji (ale w grupie innych zmiennych tego samego typu) na liście definiującej zmienne;
- pierwsza zmienna otrzymuje identyfikator 0;
- każda następna o jeden większy niż poprzednia zmienna;
- można zdefiniować do 255 zmiennych;
- lista definicyjna może być pusta;
- w przypadku braku zmiennych sekundowych, minutowych i asynchronicznych blokowany jest moduł pracy lokalnej.

Przykładowo, jeżeli w liście definicyjnej wymieniono kolejno zmienne (numer i typ): 5 asynchroniczna, 3 wewnętrzna, 125 sekundowa, 121 sekundowa i 200 minutowa, to powiązanie identyfikatorów ze zmiennymi będzie następujące (identyfikator -> numer zmiennej): 0->125 1->121 2->200 3->5 4->3. Pozostałe elementy tablicy ltab_k zawierają wartość 0xff (brak definicji zmiennej o danym numerze).

Identyfikatory wewnętrzne zmiennych są używane w pozostałych tablicach jako INDEKSY przy dostępie do poszczególnych elementów tych tablic.

ltab_f

Tablica flag zmiennych uporządkowanych według identyfikatorów. Wielkość tablicy jest uzależniona od sumarycznej liczby wszystkich zmiennych zdefiniowanych w danej stacji master. Dla zmiennych zewnętrznych elementy tablicy zawierają bajt z flagami tych zmiennych, natomiast w przypadku zmiennych wewnętrznych przechowywany jest w niej parametr stala2.

ltab_d

Tablica adresów dostępu do wartości zmiennej. Tablica podzielona jest na dwie części H i L, które przechowują odpowiednio starszą i młodszą część adresu. Podobnie jak w przypadku tablicy ltab_f zawiera ona tyle elementów, ile zdefiniowano zmiennych. Dla zmiennych zewnętrznych element tablicy zawiera adres (w pamięci XRAM), gdzie przechowywana jest wartość zmiennej (w postaci ciągu znaków ASCII). Dla zmiennych wewnętrznych przechowywany jest adres początku procedury obsługi danej zmiennej (adres w pamięci EPROM)

ltab_p

Tablica adresów procedur sterujących. Podobnie jak tablica ltab_d podzielona jest na dwie części L i H i ma identyczną wielkość. Dla zmiennych zewnętrznych element tablicy zawiera adres w pamięci EPROM wskazujący na parametr czas_pomiaru znajdujący się bezpośrednio przed parametrami adres i kod_rozkazu_SCPI. Dla zmiennych wewnętrznych przechowywany jest tutaj adres początku obszaru pamięci XRAM przydzielonej danej zmiennej, gdy parametr stala1 jest różny od 0.

ltab_s

Tablica liczników dla zmiennych zewnętrznych sekundowych. Składa się z dwóch podtablic, z których pierwsza przechowuje bieżący stan licznika odmierzającego okres cyklu danej zmiennej, a druga zawiera czas cyklu. Wielkość tablicy jest określona ilością zmiennych sekundowych.

ltab_m

Tablica liczników dla zmiennych zewnętrznych minutowych podobna do ltab_s.

Elementem nie wchodzącym bezpośrednio w skład bazy danych zmiennych stanu stacji master jest tablica **ltab_t**. Umożliwia ona realizację operacji odpytywania o wartość zmiennych stanu w stacjach podrzędnych w dwóch krokach z odmierzaniem czasu pomiędzy wysłaniem polecenia SCPI wyzwalającego pomiar (pierwsza transakcja) i polecenia odczytującego zmierzoną wartość (druga transakcja). Czas pomiędzy tymi transakcjami określony jest przez parametr czas_pomiaru i wyrażony w sekundach. W przypadku odczytu wartości zmiennej, dla której czas pomiaru jest różny od zera, po pierwszym kroku zapamiętany zostaje identyfikator zmiennej oraz czas pomiaru w tablicy ltab_t. Liczniki czasu

w tablicy ltab_t są dekrementowane cyklicznie co 1 sekundę, a fakt wyzerowania któregokolwiek z liczników jest wykrywany i wyzwala realizację drugiego kroku operacji, tj. wysłanie polecenia SCPI odczytującego zmierzoną wartość.

7. Podstawowe moduły stacji master lokalny

Na rysunku 1 przedstawiono podstawowe moduły wchodzące w skład oprogramowania stacji master. Strzałkami zaznaczono powiązania pomiędzy modułami (klient i serwer) oraz dostęp modułów do ważniejszych struktur danych. Moduły te można podzielić na dwie grupy: opracowane wcześniej moduły odpowiedzialne za funkcje routera w hierarchicznej sieci polowej opisane dokładniej w pracy [3] oraz zestaw modułów odpowiedzialnych za sterowanie autonomiczną pracą podsieci.

7.1. Grupa modułów związana z funkcjami routera

RSA1/RSM1 to moduły komunikacyjne realizujące transmisje ramek dla łącza RS1 (łącze "do góry") w protokołach ASCII i ModBUS. Zgodnie z konwencją zaproponowaną w pracy [3] pełnią one rolę serwerów. Kolejnym modułem jest procedura SERV realizująca funkcje serwera protokołu ModBUS dla łącza RS1. Inicjuje ona odbiór ramki z rozkazem (z wyższego poziomu sieci hierarchicznej) i wstępnie analizuje zawartość ramki. W przypadku otrzymania ramki z rozkazem SCPI jest ona przekazywana do modułu SCPI, który jest dekoderem języka SCPI. Moduł ten po rozpoznaniu rozkazu przekazuje go do realizacji jednemu z modułów: SDEV, STST lub SVAR. Moduły STST (realizacja rozkazu ":tst<n>?") i SDEV (realizacja rozkazu ":dev<n>:comm") wykonują określony rozkaz wykorzystując do komunikacji na sieci niższego poziomu (łącze RS0) funkcje oferowane przez moduł TRAN. Moduł ten jest serwerem realizującym transakcje w protokole ModBUS na sieci niższego poziomu. Korzysta on do realizacji transmisji z modułu RSM0 (podobnego do modułu RSM1). Wynik realizacji transakcji w protokole ModBUS (np. ramka z danymi) jest przekazywany z modułu TRAN poprzez jeden z modułów realizujących rozkazy SCPI (SDEV lub STST) do modułu SERV, który jest odpowiedzialny za przygotowanie odpowiedzi i odesłanie jej korzystając z jednego z modułów komunikacyjnych (RSA1 lub RSM1). Wspomniany wyżej moduł SVAR był odpowiedzialny za wykonanie rozkazu SCPI ":var<n>". Należy w tym miejscu wspomnieć jeszcze o modułach: INIT, który jest odpowiedzialny za inicjalizację pracy stacji master, oraz module TIM0, który pełni rolę serwera czasowego odpowiedzialnego za odmierzanie czasów przeterminowania (timeout)

ważniejszych operacji. Z usług tego ostatniego korzystają m.in. moduły komunikacyjne RSA1, RSM1 i RSM0 (timeout znakowy) oraz TRAN (timeout transakcji ModBUS'a).

7.2. Moduły autonomicznej pracy podsieci

Moduły odpowiedzialne za sterowanie pracą autonomicznej podsieci skupione są wokół bazy danych zmiennych stanu zdefiniowanych w stacji master. Jej strukturę opisano w poprzednim punkcie. Poszczególne moduły, działając niezależnie, modyfikują zawartość bazy danych, wpływając na działanie innych modułów programowych.

Omawianie poszczególnych modułów rozpoczniemy od modułów inicjalizacji LDV1, LDV2 i LDVE. Sa one odpowiedzialne za przeanalizowanie listy definiującej zmienne stanu (LDV1) i wygenerowanie na jej podstawie tablic wchodzących w skład bazy danych (LDV2). W przypadku wykrycia błędu tworzona jest uproszczona baza danych umożliwiająca pracę stacji master (LDVE). Moduły te zostały połączone z istniejącym poprzednio modułem INIT.

Kolejnym modułem jest LDEC. Jego zadaniem jest cykliczne dekrementowanie liczników bieżących zmiennych sekundowych (ltab_s) lub minutowych (ltab_m). Moduł ten wywoływany jest przez serwer czasowy TIM0 co 1 sekundę i dodatkowo co 1 minutę. W przypadku wyzerowania jakiegoś licznika zapalana jest flaga <7> (wyzwolenie pomiaru) w odpowiadającym danej zmiennej bajcie flag (ltab_f) i dodatkowo licznik bieżący jest na nowo ładowany wartością okresu cyklu.

Moduł LVAR zastąpił istniejący poprzednio moduł SVAR. Jego zadaniem jest realizacja dostępu do wartości zmiennych stanu dla potrzeb realizacji rozkazów :var przekazywanych ze stacji master na wyższych poziomach hierarchii. W tym celu moduł analizuje treść rozkazu SCPI wyodrębniając numer zmiennej i ewentualne parametry oraz znak '?'. Numer zmiennej jest konwertowany na identyfikator wewnętrzny (ltab_k), który jest podstawą do określenia typu zmiennej, a także będzie używany w charakterze indeksu przy dostępie do innych tablic wchodzących w skład bazy danych. W przypadku zmiennej wewnętrznej wywoływana jest procedura o adresie odczytanym z tablicy ltab_d. Dla zmiennych zewnętrznych zależnie od operacji (odczyt lub zapis) następuje odczytanie lub zapis wartości zmiennej w pamięci XRAM według adresu z tablicy ltab_d. Dla operacji wyzwolenia następuje zapalenie flagi <7> w odpowiednim elemencie tablicy ltab_f.

Moduł IVP zawiera zestaw typowych procedur dostępu do zmiennych wewnętrznych standardowo definiowanych w każdej stacji master.

Moduł LPT zawiera procedury operujące na tablicy ltab_t przechowującej parametry (identyfikator wewnętrzny i czas pomiaru) zmiennych, dla których realizowany jest dwuetapowy algorytm dostępu do wartości i zawiera:

- procedurę cyklicznego dekrementowania liczników (wywoływana co 1 sekundę z modułu serwera czasowego TIM0);
- procedurę inicjowania tablicy ltab_t (wywoływana przez moduł INIT);
- procedurę przeglądania tablicy ltab_t, wyszukującej element, dla którego licznik czasu pomiaru jest wyzerowany. Po znalezieniu takiego elementu zwracany jest identyfikator zmiennej;
- procedurę wstawiającą nowy element (identyfikator zmiennej i czas pomiaru) do tablicy.

Dwie ostatnie procedury wykorzystywane są przez moduł LPF, który jest głównym modułem pracy lokalnej. Moduł ten pracujący w oparciu o mechanizm maszyny stanów (szeroko stosowanej w modułach opisanych w pracy [3]) stanowi część pętli programu głównego razem z modułem SERV. Moduł LPF, wykorzystując procedurę z modułu LPT, przeszukuje tablicę ltab_t w poszukiwaniu zmiennej, dla której wysłano już rozkaz wyzwolenia pomiaru (pierwszy etap transakcji dostępu do wartości zmiennej) i dla której upłynął już czas pomiaru (wyzerowany licznik). Po znalezieniu takiej zmiennej przygotowywany jest rozkaz odczytu wartości zmiennej i inicjowana jest transakcja ModBUS'a przy użyciu modułu TRAN. Po jej poprawnym zakończeniu wartość zwrócona przez stację podrzędną jest przepisywana do pola z wartością danej zmiennej (adres w tablicy ltab_d), a dodatkowo gaszona jest flaga wyzwolenia pomiaru danej zmiennej. W przypadku gdy moduł LPF nie znalazł w tablicy ltab_t zmiennej, dla której należałoby dokończyć dwuetapową transakcję dostępu do wartości, przeszukiwana jest tablica ltab_f. Po znalezieniu zmiennej z zapaloną flagą wyzwolenia pomiaru inicjowana jest dla niej transakcja odczytu jej wartości ze stacji podrzędnej. W przypadku zmiennej z czasem pomiaru równym zero realizowana jest pojedyncza transakcja (wyzwolenie i odczyt wartości poleceniem SCPI według adresu z ltab_p), a wartość zwrócona ze stacji podrzędnej jest wpisywana do pola z wartością danej zmiennej (adres z ltab_d). Dla zmiennych, dla których czas pomiaru jest różny od zera, inicjowana jest w pierwszym kroku transakcja wyzwolenia pomiaru (polecenie SCPI wskazywane jest przez wpis w tablicy ltab_p). Po pozytywnym potwierdzeniu przez stację podrzędną przyjęcia rozkazu i rozpoczęcia pomiaru moduł LPF dopisuje do tablicy ltab_t informację o kolejnej zainicjowanej transakcji dwuetapowej. Maksymalnie można jednocześnie realizować cztery transakcje dwuetapowe.

W nowej wersji oprogramowania stacji master zawierającej moduły pracy lokalnej nieznacznie zmodyfikowano także wcześniej istniejące moduły. Oprócz poprawienia kilku drobnych błędów wyodrębniono z modułu SERV wszystkie procedury przygotowujące i odsyłające odpowiedzi SCPI i utworzono z nich osobny moduł REP.

8. Podsumowanie

Przedstawione oprogramowanie rozbudowuje opracowane wcześniej moduły programowe stacji mastera lokalnego o dodatkową funkcję związaną z autonomią pracy podsieci. Stacja mastera lokalnego odgrywa w takiej wyodrębnionej podsieci rolę stacji sterującej wyzwalaniem pomiarów według opracowanego wcześniej programu odpytywania podrzędnych stacji o wartości wybranych zmiennych stanu zdefiniowanych w tych stacjach. Istniejące mechanizmy pozwalają na realizację odczytu wartości zmiennej w jednym (wykonanie pomiaru i zwrócenie wartości) lub w dwóch krokach (wyzwolenie, a po określonym parametrem czasie odczytanie wartości w ramach osobnej transakcji). Czasy cyklu odpytywania mogą być wyrażane w sekundach lub minutach, dopuszczalny zakres obejmuje czasy od 1 sekundy do 4 godzin i 15 minut. Możliwe jest także odpytywanie o wartość zmiennej na żądanie (zmienne asynchroniczne). Dodatkowe znaczniki związane ze zmiennymi umożliwiają implementację zmiennych do zapisu, dla których wartość zapisana do stacji master jest następnie przepisywana do właściwej stacji slave oraz implementację zmiennych tylko do odczytu. Mechanizm zmiennych wewnętrznych umożliwia ujednolicony dostęp do podstawowych parametrów pracy stacji master (podobny mechanizm zostanie w przyszłości zrealizowany dla stacji slave). Poszczególne moduły programowe charakteryzują się dużą niezależnością (przypominają koncepcją demony w dużych systemach operacyjnych), wiele z nich zrealizowano w oparciu o mechanizm maszyny stanów (co umożliwia wielozadaniowe działanie całego oprogramowania). Współpraca wielu modułów oparta jest na mechanizmie klient-serwer. Oprogramowanie zostało opracowane w języku asemblera dla procesora 8051 i uruchomione na stacji master zrealizowanej w oparciu o mikrokontroler Dallas 80C320. Stacja wyposażona została w 8kB pamięci EPROM (program) oraz w 8kB pamięci RAM (pamięć danych). Przeprowadzone testy wykazały poprawne działanie autonomicznej podsieci sterowanej przez stację master.

LITERATURA

1. Mielczarek W., Pawłowski R.: Hierarchiczna sieć polowa sterowana rozkazami języka SCPI. ZN Pol. Śl. s. Informatyka z. 33, Gliwice 1997.
2. Mielczarek W.: Stacja slave hierarchicznej sieci polowej sterowanej rozkazami języka SCPI. ZN Pol. Śl. s. Informatyka z. 33, Gliwice 1997.
3. Pawłowski R.: Stacja master hierarchicznej sieci polowej sterowanej rozkazami języka SCPI. ZN Pol. Śl. s. Informatyka z. 34, Gliwice 1998.
4. IEEE Std. 488.2-1987 "IEEE Standard Codes, Formats, Protocols, and Common Commands" i jej rozszerzenie SCPI (Standard Commands for Programmable Instruments) version 1993.

Recenzent: Dr inż. Ryszard Winiarczyk

Wpłynęło do Redakcji 12 kwietnia 1999 r.

Abstract

This publication is a second paper about the local master station of the hierarchical fieldbus controlled by SCPI commands. The paper describes the software controlled autonomous subnet operation. The software was built as a set of independent program modules (fig. 1). The local master station controls release of the measurments in the hierarchical fieldbus. In this station two types of state variables: external and internal were implemented. State variable is an object with attributes like: value, action and severals additional parameters. External state variables preserved value that was read from the slave stations. Each external state variable is described by the parameters like: time between measurments, measurement time, binary flags and SCPI command. Internal state variables control the basic parameters of the master station like: communication timeouts or error flags. Uniformed access to a value of a state variable (external and internal) is possible by the SCPI command :var<variable_number>. A value may be read by the command ":var123?" or write by the command ":var123 24.5". The software contrlling the master operation was coded in 8051 assambler language and compiled and tested in lab system with Dallas 80C320 microcontroller.