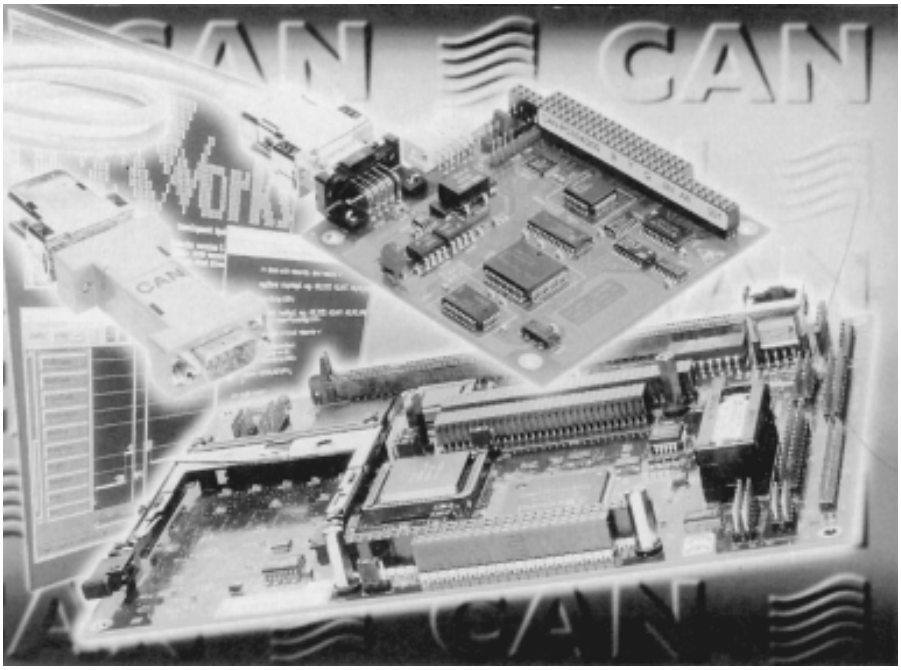
Magistrala CAN, część 5

Oprogramowanie interfejsu

Sieć CAN składa się oczywiście nie tylko z opisanego w poprzedniej części artykułu interfejsu magistrali. Interfejs jest po prostu łącznikiem pomiędzy mikrosterownikiem czy komputerem a właściwą magistralą CAN. Układ, szczegółowo opisany przed miesiącem, do prawidłowego działania wymaga oprogramowania sterującego i to właśnie jest tematem tej części publikacji.



Każda stacja lub węzeł systemu magistrali CAN wymaga, poza interfejsem magistrali, mikrosterownika lub komputera z odpowiednim programem. Do uruchomienia stacji, jej sprawdzenia i obsługiwania potrzeba dwóch zestawów programów: oprogramowania operacyjnego i oprogramowania aplikacyjnego.

Program operacyjny wprowadza cały system w ruch i zapewnia jego działanie, a także testuje interfejs wraz z mikrosterownikiem lub komputerem. Taki test wykazuje, czy sterowanie przez mikrosterownik/komputer działa właściwie, tak z punktu widzenia sprzętowego, jak i programowego, oraz czy dane są poprawnie przesyłane do magistrali CAN. Test więc pomaga w ustanowieniu prostej ścieżki łączności pomiędzy dwoma lub kilkoma węzłami.

Oprogramowanie aplikacyjne jest związane ze szczególną rolą mikrosterownika/komputera w sieci. Od niego zależy konkretne zastosowanie stacji: rejestracja pomiarów, sterowanie wyświetlaczem, transmisja czasu i daty czy jeszcze inne cele.

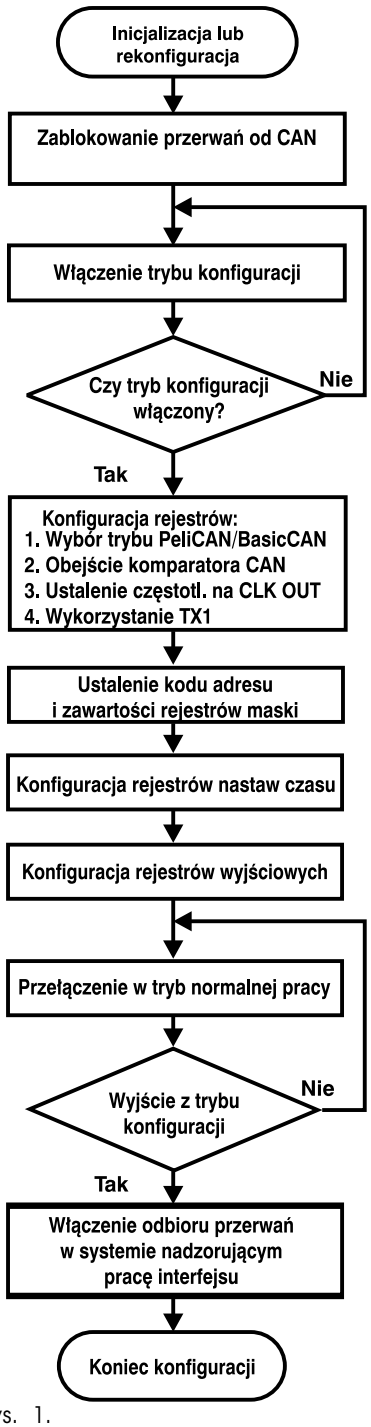
Do działania każdego węzła jest więc potrzebny szczególny program, odpowiedni do jego funkcji. Na działanie sieci składa się suma wszystkich funkcji wykonywanych przez poszczególne stacje. Innymi słowy, w celu osiągnięcia oczekiwanych rezultatów przestrzennie rozmieszczona sieć może być sterowana i monitorowana jako całość.

Program operacyjny

Programowanie sterownika CAN podlega tym samym ogólnym zasadom, jak zewnętrznych urządzeń peryferyjnych:

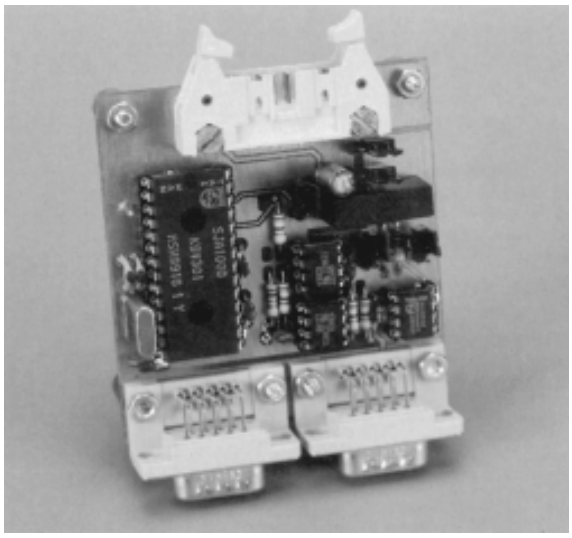
- Funkcja sterownika jest stała albo ustawiana (zaprogramowana) przez zestawy danych zapisywane w Rejestrach Funkcji Specjalnych (SFR).
- Wewnętrzne SFR-y są przez mikrosterownik/komputer interpretowane jako normalne adresy pamięci w zakresie zewnętrznej RAM, pod którymi dane mogą być wpisywane lub odczytywane. Oznacza to, że mikrosterow-

nik/komputer nie wie, że działa wraz ze sterownikiem CAN. Istotny z jego punktu widzenia i dla oprogramowania aplikacyjnego jest jedynie dostęp do określonych miejsc w pamięci. Zatem przy tworzeniu oprogramowania aplikacyjnego dla sterownika IC3 należy wykonać następujące zadania:
- ustalić podstawowy adres selekcyjny dla sterownika SJA1000,
- zinterpretować wewnętrzne ustawienia struktury SFR-ów w sterowniku,
- stworzyć procedurę podstawowej inicjalizacji sterownika,
- stworzyć procedurę dostarczania danych do magistrali,
- stworzyć procedurę odbioru danych z magistrali CAN.



Rys. 1.

Opis realizacji tych zadań dla podstawowego trybu CAN sterownika zostanie przedstawiony pokrótce w następnych paragrafach. Obszerniejsze i bardziej szczegółowe informacje można znaleźć w danych technicznych i notach aplikacyjnych sterownika.



Ustalanie podstawowego adresu selekcyjnego

Dostęp do układu jest możliwy poprzez podstawowy adres selekcyjny (wyboru układu). Ponieważ sterownik IC3, w podstawowym trybie CAN, wymaga spójnego 32-bajtowego zakresu adresów zewnętrznych, a w trybie PeliCAN jednego ze 128 bajtów, maksymalny zakres został ustalony na 128 bajtów, aby nie wykluczać możliwości użycia w przyszłości trybu PeliCAN.

Stan niski na końcówce CS (3) jest dla sterownika SJA1000 sygnałem zezwalającym. Oznacza to, że mikrosterownik/komputer musi kodować adres układu w taki sposób, aby w spójnym zakresie adresowym, nie mniejszym od 128 bajtów, wywołać sygnał stanu niskiego na końcówce 8 złącza K3, co umożliwi sterownikowi CAN przesłanie danych. Pierwszy taki adres staje się tak zwanym podstawowym adresem wyboru chipu sterownika. Gdy mikrosterownik lub komputer sięgnie do mieszczącej się w tym zakresie adresowym pozycji w RAM, uzyskuje bajt zawarty w SFR sterownika lub może do SFR wpisać nowy bajt. Przyjęto, że podstawowym adresem wyboru układu sterownika SJ1000 jest F000H.

Struktura wewnętrzna SFR

Najważniejsze SFR-y sterownika IC3 do pracy w podstawowym trybie CAN zestawiono w tab. 6. Znaczenie poszczególnych kolumn tabeli jest następujące:

1. W pierwszej kolumnie, adresów CAN, znajdują się wewnętrzne adresy odpowiednich SFR-ów, do których trzeba tylko dodać podstawowy adres wyboru układu. Jeżeli, na przykład, potrzebny jest dostęp do rejestru stanu sterownika, to do wewnętrznego adresu SFR-a, wynoszącego 2, trzeba dodać F000H. Jeżeli zatem wykonywana jest na tym rejestrze operacja odczytu lub zapisu, program musi udostępnić adres F002H w zewnętrznej RAM. Od tego momentu rejestr dzielnika sygnału zegarowego będzie dostępny pod adresem F001FH (=F000H+31D=F000H+1FH - uwaga, użyto dwóch różnych systemów liczbowych).

2. Druga kolumna pokazuje podział SFR-ów na trzy różne grupy: grupę sterowania, grupę bufora nadawania i grupę bufora odbioru.

3. Sterownik funkcjonuje w dwóch, sterowanych programem, trybach:
- w trybie działania, będącym normalnym trybem pracy,
- w trybie kasowania, będący trybem IC3 w czasie kasowania sprzętowego albo gdy bit kasowania rejestru sterowania jest ustawiony. Sterownik powraca wtedy do normalnego trybu działania.

Tryb kasowania jest potrzebny do inicjalizacji sterownika, tylko w tym trybie bowiem mogą zostać ustawione niektóre parametry

działania. Zostaje wówczas ustawiony bit kasowania (sterownik ustawia swój normalny tryb działania), po czym odpowiednie parametry mogą zostać zmienione i bit kasowania jest wyłączony. Sterownik podejmuje wtedy na nowo działanie ze zmienionymi parametrami.

4. W trzeciej i czwartej kolumnie pokazano:
- funkcje rejestru,
- znaczenie wpisywanej w trybie działania zawartości rejestru,
- znaczenie odczytywanej zawartości rejestru.

5. W piątej i szóstej kolumnie są pokazane odnośne dane rejestru w trybie kasowania.

Przykład opisu SFR o adresie 4

Tryb działania (normalne funkcjonowanie sterownika):
- odczyt - chociaż odczyt tego rejestru jest możliwy, wyniki odczytu nie są użyteczne, ponieważ zawsze odczytuje się wartość FFH.
- zapis - do rejestru nie można niczego wpisać.

Tryb kasowania (sterownik jest w trybie kasowania):
- odczyt - wynikiem odczytu z rejestru jest kod akceptacji,
- zapis - do rejestru można wpisać nowy kod akceptacji.

Z tego przykładu widać, że w czasie normalnego działania sterownika ten SFR nie ma specjalnej funkcji. Trzeba jednak zwrócić uwagę na to, że w trybie kasowania kod akceptacji, z którym sterownik funkcjonuje w czasie normalnego działania, jest ustawiony.

Tworzenie procedury podstawowej inicjalizacji

Przed rozpoczęciem pracy nad tą procedurą niezbędne jest zapoznanie się z notą aplikacyjną sterownika SJA1000 (AN97076 - dostępna w Internecie). Na 23. stronie tego dokumentu jest przytoczony schemat działań ze szczegółowymi komentarzami o sposobie inicjalizacji sterownika. Trzeba też zapoznać się dokładnie z opisem pojedynczego rejestru, co pozwoli bez trudu dobierać parametry zgodnie z indywidualnymi potrzebami.

Tworzenie procedury przesyłania danych

Jak już wspomniano, większość zadań, potrzebnych przy przesyłaniu danych, przejmuje sterownik SJA1000. Wysłanie bajtu danych do magistrali CAN wymaga jedynie czterech czynności:
- dostarczenia sterownikowi identyfikatora (ID) ramki, która ma zostać wysłana,
- wskazania, ile należy wysłać bajtów (0..8) danych,
- określenia, czy ramka jest ramką zdalnego żądania transmisji (RTR),
- wpisania wymaganych bajtów danych do bufora nadawania w sterowniku.

I to wszystko! Reszta procesu jest automatycznie wykonywana przez sterownik CAN:
- zestawianie ramki,
- obliczanie sumy CRC (cyklicznej kontroli nadmiarowej),
- dołączenie do ramki pozostałych pól,
- uzyskanie dostępu do magistrali,
- wysłanie ramki,
- sprawdzenie błędów.

Za pośrednictwem rejestru stanu użytkownik otrzymuje komunikaty o powodzeniu lub niepowodzeniu transmisji.

Tab. 6. Wewnętrzne SFR-y sterownika, używane w podstawowym trybie działania interfejsu CAN.

ADRES CAN	SEGMENT	TRYB DZIAŁANIA		TRYB KASOWANIA	
		WPIS	ODCZYT	WPIS	ODCZYT
0	sterowanie	sterowanie	sterowanie	sterowanie	sterowanie
1		(FFH)	rozkaz	(FFH)	(FFH)
2		stan	-	stan	-
3		przerwanie	-	przerwanie	-
4		(FFH)	-	kod akceptacji	kod akceptacji
5		(FFH)	-	maska akceptacji	maska akceptacji
6		(FFH)	-	takt magistrali 0	takt magistrali 0
7		(FFH)	-	takt magistrali 1	takt magistrali 1
8		(FFH)	-	sterowanie wyj.	sterowanie wyj.
9		test	test; uwaga 2	test	test; uwaga 2
10	bufor nadawania	identyfikator (10 do 3)	identyfikator (10 do 3)	(FFH)	-
11		identyfikator (2 do 0) RTR i DLC	identyfikator (2 do 0) RTR i DLC	(FFH)	-
12		bajt danych 1	bajt danych 1	(FFH)	-
13		bajt danych 2	bajt danych 2	(FFH)	-
14		bajt danych 3	bajt danych 3	(FFH)	-
15		bajt danych 4	bajt danych 4	(FFH)	-
16		bajt danych 5	bajt danych 5	(FFH)	-
17		bajt danych 6	bajt danych 6	(FFH)	-
18		bajt danych 7	bajt danych 7	(FFH)	-
19		bajt danych 8	bajt danych 8	(FFH)	-
20	bufor odbioru	identyfikator (10 do 3)	identyfikator (10 do 3)	identyfikator (10 do 3)	identyfikator (10 do 3)
21		identyfikator (2 do 0) RTR i DLC	identyfikator (2 do 0) RTR i DLC	identyfikator (2 do 0) RTR i DLC	identyfikator (2 do 0) RTR i DLC
22		bajt danych 1	bajt danych 1	bajt danych 1	bajt danych 1
23		bajt danych 2	bajt danych 2	bajt danych 2	bajt danych 2
24		bajt danych 3	bajt danych 3	bajt danych 3	bajt danych 3
25		bajt danych 4	bajt danych 4	bajt danych 4	bajt danych 4
26		bajt danych 5	bajt danych 5	bajt danych 5	bajt danych 5
27		bajt danych 6	bajt danych 6	bajt danych 6	bajt danych 6
28		bajt danych 7	bajt danych 7	bajt danych 7	bajt danych 7
29		bajt danych 8	bajt danych 8	bajt danych 8	bajt danych 8
30		(FFH)	-	(FFH)	-
31		dzielnik sygnału zegarowego	dzielnik sygnału zegarowego	dzielnik sygnału zegarowego	dzielnik sygnału zegarowego

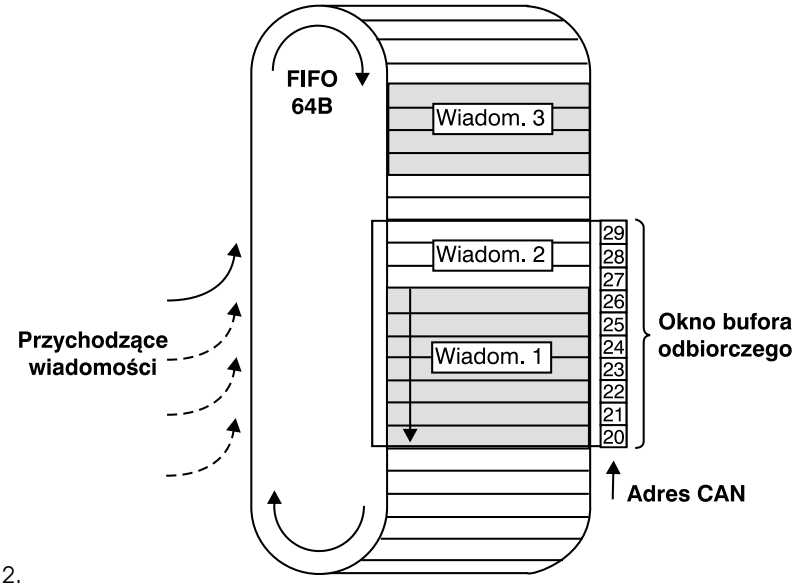
Tworzenie procedury odbioru danych

Przy odbiorze danych sterownik SJA1000 także przejmuje większość zadań, czyli odbiera dane niemal całkowicie automatycznie. Sterownik przetwarza odebrane ramki i wpisuje zawarte w nich potrzebne informacje do sekcji wykrywania błędów i filtru akceptacji w jego RXFIFO. Jeżeli filtr akceptacji jest wyłączony, oceniana jest każda odebrana ramka. W RXFIFO są zapisywane kolejne dane z każdej ramki (tab. 6, adresy 20..29):

- identyfikator ramki,
- bit zdalnego żądania transmisji (RTR),
- kod długości danych (DLC),
- bajty danych użytecznych.

Pojemność wewnętrznej pamięci odbiorczej FIFO w IC3 wynosi dokładnie 64 bajty. Liczba ramek, które mogą być przechowywane w pamięci interfejsu, zależy od rozmiaru ramki, a przede wszystkim od kodu długości danych. Okienkiem bufora odbioru (tab. 6, adresy 20..29), które może być czytane przez użytkownika, jest to, co zostaje przesunięte do okienka przez RXFIFO. Składa się z właśnie odebranego zespołu danych (ramka komunikatu), które użytkownik może przetwarzać za pomocą programu. Łączność pomiędzy SJA1000 a mikrosterownikiem/komputerem w trybie odbioru może przybierać dwie formy:

- *Sterowanie przerwaniami.* Gdy sterownik otrzyma pozbawioną błędów, kompletną ramkę, inicjuje przerwanie w mikrosterowniku poprzez jego końcówkę 16 (INT). Wywołuje to natychmiastową reakcję mikrosterownika/komputera na otrzymany komunikat, który może niezwłocznie zostać odczytany za pośrednictwem sterownika.
- *Operacja odpytywania (polling).* Przy tym rodzaju operacji bit stanu bufora odbioru w rejestrze stanu sterownika jest nieustannie odpytywany przez mikrosterownik/komputer. Gdy bit ten jest ustawiony, co sygnalizuje poprawne odebranie przez sterownik co najmniej jednego komunikatu, program odczytuje ramkę i stosownie ją przetwarza.



Rys. 2.

Po odczytaniu komunikatu program aplikacyjny ponownie udziela zezwolenia okienku bufora odbioru, potwierdzając, że otrzymany komunikat został odebrany i przetworzony. Okienko odbiorcze jest więc gotowe do odbioru z RXFIFO następnej ramki. W ten sposób program aplikacyjny jest informowany o przetworzeniu kolejnych ramek. Trzeba jeszcze wspomnieć o dwóch sprawach:

- Niezwłocznie po odczytaniu i przetworzeniu ramki (komunikatu) okienko bufora odbioru musi zostać zwolnione przez „rozkaz zwolnienia bufora odbioru“, aby sterownik mógł do okienka przesunąć następny komunikat. Jeżeli rozkaz ten nie zostanie wydany, to ten sam komunikat będzie nieustannie przetwarzany powodując przepełnienie RXFIFO, ponieważ nie będą przesuwane następne odbierane ramki.
- Gdy częstość ramek jest duża, i wiele ramek jest wysyłanych jedna za drugą, a kolejne rozkazy mogą być przesuwane niedostatecznie sprawnie, powstaje ryzyko szybkiego przepełnienia RXFIFO. Wobec takiego niebezpieczeństwa, trzeba użyć odpowiednio szybkiego mikrosterownika/komputera z wysokiej jakości oprogramowaniem. Przepełnienie RXFIFO jest sygnalizowane przez sterownik ustawieniem bitu błędu, czyli bitu przepełnienia danych w rejestrze stanu. Komunikat, który miał zostać przesunięty do RXFIFO (i który wywołał przepełnienie), zostaje wtedy skasowany i stracony.

EE