STUDIA INFORMATICA 2003
Volume 24 Number 2A (53)

Krzysztof TOKARZ, Bartłomiej ZIELIŃSKI
Politechnika Śląska, Instytut Informatyki

PORÓWNANIE METOD IMPLEMENTACJI PROTOKOŁU IrDA W ŚRODOWISKU MIKROKOMPUTERA JEDNOUKŁADOWEGO

Streszczenie. W artykule przedstawiono porównanie dwóch metod implementacji protokołu IrDA w środowisku mikrokomputera jednoukładowego. Jedna z metod, polegająca na przetwarzaniu pojedynczych bajtów, osiąga mniejsze prędkości transmisji, ma jednak mniejsze wymagania sprzętowe. Druga metoda kosztem bardziej rozbudowanej części sprzętowej pozwala na osiągnięcie lepszych parametrów transmisji. W artykule dokonano porównania tych metod pod kątem podstawowych parametrów transmisji danych.

Słowa kluczowe: IrDA, implementacja protokołu.

COMPARISON OF IrDA PROTOCOL STACK IMPLEMENTATION IN MICROCONTROLLER'S ENVIRONMENT

Summary. This paper describes two methods of IrDA standard implementation in microcontroller's environment. First method using less complicated hardware allows slower transmission rate. Second method offers faster transmission but works only on more complex hardware. In this paper the comparison of those methods is made.

Keywords: IrDA, protocol implementation.

1. Wstęp

Szybki rozwój technologii komputerowej spowodował powstanie zminiaturyzowanych komputerów przenośnych o bardzo dużych możliwościach obliczeniowych. Komputery te, dla wygody użytkowania, wyposażone są w bezprzewodowe łącza transmisyjne wykorzystujące zarówno fale radiowe, jak i łącza optyczne. Najpopularniejszymi standardami bezprzewodowej transmisji danych za pośrednictwem fal radiowych są

Bluetooth oraz 802.11b, natomiast wykorzystującym fale świetlne z zakresu podczerwieni jest IrDA [1]. Łącze zgodne ze standardem IrDA występuje prawie w każdym komputerze przenośnym oraz w innych urządzeniach, jak na przykład w telefonie komórkowym. Prędkości transmisji do 115,2 kb/s zapewnione w podstawowej wersji protokołu IrDA, kompatybilnej z łączem szeregowym, są w zupełności wystarczające do przesyłania niewielkich porcji danych. Łącze IrDA w podstawowej [3] lub okrojonej wersji IrDA-Lite [4] może być z powodzeniem zastosowane w urządzeniach wyposażonych w mikroprocesor o niewielkiej mocy obliczeniowej. Jako przykład takiego wykorzystania łącz bezprzewodowych można przytoczyć system złożony ze stacji pomiarowych, z których akwizycji danych dokonuje pracownik obsługi wyposażony w komputer przenośny.

W niniejszej pracy przedstawiono opis dwóch implementacji protokołu IrDA w środowisku mikrokomputera jednoukładowego zgodnego z 8051. Dokonano analizy i porównania tych implementacji pod kątem podstawowych parametrów, jak maksymalna prędkość transmisji i maksymalny rozmiar przesyłanej ramki.

2. Charakterystyka środowiska

Mikrokomputery jednoukładowe zgodne z 8051, mimo stosunkowo niewielkiej mocy obliczeniowej, nie przekraczającej kilku milionów instrukcji na sekundę i ośmiobitowej architektury, są nadal bardzo szeroko wykorzystywane w urządzeniach elektronicznych opartych na technice mikroprocesorowej. Powstało i nadal powstaje wiele ciekawych wersji tych mikrokomputerów. Niektóre z nich charakteryzują się większą szybkością działania uzyskaną przez reorganizację architektury wewnętrznej i zmniejszenie liczby taktów zegarowych potrzebnych na wykonanie instrukcji. Inne posiadają wbudowane specjalizowane moduły obliczeniowe czy komunikacyjne. Do większości z nich można podłączyć zewnętrzne pamięci programu i danych pozwalające na tworzenie bardziej złożonych aplikacji. Opisane w pracy implementacje protokołu IrDA oparto na konkretnych konfiguracjach sprzętowych, nie ograniczają się jednak one tylko do takich konfiguracji.

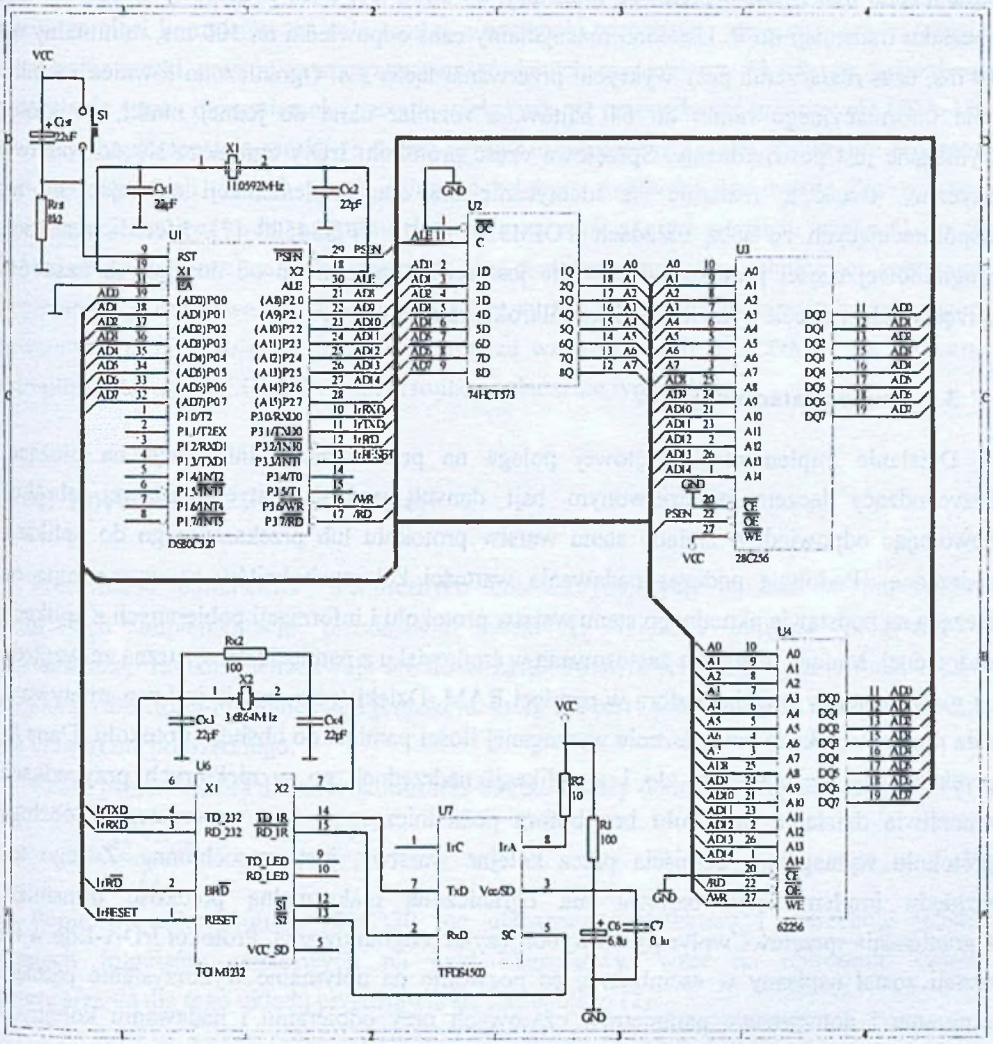
2.1. Środowisko z pamięcią wewnętrzną

Projett środowiska sprzętowego z pamięcią wewnętrzną oparto na mikrokomputerze jednoukładowym AT89C52 firmy Atmel [5]. Mikrokomputer ten posiada 8 kB pamięci programu oraz 256 komórek pamięci RAM. Maksymalna częstotliwość pracy zegara taktującego wynosi 24 MHz, co daje średnią prędkość 1,5 milionów instrukcji na sekundę, przy założeniu występowania w programie jednakowej ilości instrukcji wykonywanych w

jednym i w dwóch taktach maszynowych. Dokładny opis środowiska z wewnętrzną pamięcią programu oraz danych jest zawarty w pracy [2].

2.2. Środowisko z pamięcią zewnętrzną

Projekt środowiska z pamięcią zewnętrzną oparto na mikrokomputerze jednoukładowym DS80C320 firmy Dallas [8]. Mikrokomputer ten nie posiada wewnętrznej pamięci programu i musi mieć podłączoną pamięć zewnętrzną. Program jest zapisany w pamięci typu EEPROM o rozmiarze 32 kB. Rozmiar wewnętrznej pamięci RAM wynosi 256 bajtów. Dodatkowo w układzie zastosowano zewnętrzną pamięć danych o rozmiarze 32 kB służącą buforowaniu przychodzących i wychodzących ramek. Schemat elektryczny tej wersji przedstawia rysunek 1.



Rys. 1. Schemat układu z zewnętrzną pamięcią programu i danych
Fig. 1. Schema of the device with external program and data memory

Maksymalna częstotliwość pracy zegara taktującego wynosi 33 MHz, a dzięki zastosowaniu zmodyfikowanej architektury wewnętrznej mikrokomputera uzyskano dodatkowo 2,5-krotne średnie przyspieszenie wykonywania programu w odniesieniu do wersji standardowej.

3. Metody implementacji protokołu IrDA

Standard IrDA definiuje wersję uproszczoną IrDA-Lite [4] do zastosowania w sprzęcie o niewielkiej mocy obliczeniowej. IrDA-Lite doskonale nadaje się do implementacji w środowisku mikrokomputera jednoukładowego serii 8051. W tej wersji protokołu ograniczono maksymalną prędkość transmisji do 9600 b/s, ilość dodatkowych znaczników początku transmisji do 0. Ustalono maksymalny czas odpowiedzi na 500 ms, minimalny na 10 ms, czas rozłączenia przy wykryciu przerwania łącza 3 s. Ograniczono również rozmiar pola informacyjnego ramki do 64 bajtów, a rozmiar okna do jednej ramki, po której wymagane jest potwierdzenie. Sprzętowa część protokołu IrDA ogranicza się do warstwy fizycznej. Część tę realizuje się identycznie dla obu implementacji opierając się na współpracujących ze sobą układach TOIM3232 [6] i TFDS4500 [7]. Metoda realizacji programowej części protokołu IrDA-Lite jest ściśle uzależniona od dostępnych zasobów sprzętowych w docelowym środowisku mikrokomputerowym.

3.1. Implementacja bajtowa

Działanie implementacji bajtowej polega na przetwarzaniu informacji na bieżąco. Przychodzący łączem podczerwonym bajt danych podlega natychmiastowej obróbce powodując odpowiednie zmiany stanu warstw protokołu lub przekazanie go do aplikacji nadrzędnej. Podobnie podczas nadawania wartości kolejnych bajtów są wyznaczane na bieżąco na podstawie aktualnego stanu warstw protokołu i informacji pobieranych z aplikacji nadrzędnej. Metoda ta została zastosowana w środowisku z pamięcią wewnętrzną ze względu na mały dostępny rozmiar bufora w pamięci RAM. Dzięki temu, że nie jest zapamiętywana cała ramka, uzyskano zmniejszenie wymaganej ilości pamięci do obsługi protokołu. Dane są przekazywane bezpośrednio do i z aplikacji nadrzędnej, co w niektórych przypadkach umożliwia działanie protokołu bez bufora pośredniczącego. Jest oczywiste, że obsługa protokołu wymagająca przejścia przez kolejne warstwy, jest czasochłonna. Z tego też względu implementacja bajtowa ma ograniczoną maksymalną prędkość transmisji. Ograniczenia sprzętowe wpłynęły na wybór języka programowania. Protokół IrDA-Lite w tej wersji został napisany w asemblerze, co pozwoliło na optymalne wykorzystanie pamięci programu i dotrzymanie parametrów czasowych przy odbieraniu i nadawaniu kolejnych

bajtów ramki. Połączeniem z aplikacją nadrzędną zajmuje się warstwa emulacji łącza szeregowego IrCOMM w wersji 3-Wire RAW obsługiwana bezpośrednio z warstwy IrLMP.

3.2. Implementacja ramkowa

Implementacja ramkowa polega na przetwarzaniu danych jako kompletnych ramek informacyjnych. Odbierane dane są obrabiane tylko na poziomie podstawowym przez podwarstwę obsługi ramek (*framing layer*). Po odebraniu bajtu jest on zapamiętywany w dwubajtowej kolejce FIFO służącej wyliczaniu sumy kontrolnej CRC. W tej części jest również zapewniane zachowanie przezroczystości protokołu. Ostatnią czynnością jest zapamiętanie przychodzących danych w pamięci buforowej, której zawartość jest przetwarzana przez wyższe warstwy po odebraniu całej ramki. Podczas nadawania najpierw jest w pamięci kompletowana cała ramka, a następnie rozpoczyna się proces jej wysyłania. Jako bufor ramki pracuje zewnętrzna pamięć danych o rozmiarze 32 kB, co pozwala na przesyłanie ramek o rozmiarach znacznie większych niż przewidziane w protokole IrDA-Lite 64 bajty. Implementacja ramkowa dzięki ograniczonemu do minimum obciążeniu procesora podczas transmisji danych może osiągnąć większe prędkości transmisji. Zastosowanie zewnętrznej pamięci kodu programu pozwoliło na napisanie tej wersji w języku C, co ma znaczenie przy ewentualnym przeniesieniu protokołu na inną platformę sprzętową. Implementacja ramkowa ze względu na większe możliwości pamięciowe została wzbogacona o warstwę IrTTP jako pośredniczącą pomiędzu warstwą IrLMP i IrCOMM. W tej wersji zaimplementowano IrCOMM wykorzystujący połączenie typu 9-Wire.

4. Porównanie implementacji

Porównanie parametrów technicznych charakteryzujących opisane w poprzednich rozdziałach implementacje przedstawia tabela 1. Większa objętość programu w implementacji ramkowej ma swoje źródło w zastosowanym języku programowania oraz w większej złożoności implementacji protokołu, który nie jest ograniczony do wersji IrDA-Lite dla urządzenia podrzędnego.

Czasy przetwarzania dla mikrokontrolera 89C52 zostały obliczone na postawie wzoru (1).

$$t = \frac{12 * n_{cykl}}{f_{xtal}} \tag{1}$$

Ponieważ mikrokontroler 80C320 ma ulepszoną architekturę i potrzebuje jedynie czterech impulsów zegarowych na cykl maszynowy, wzór na obliczenie czasów przetwarzania dla tego układu przedstawia się następująco (2).

$$t = \frac{4 * n_{cykl}}{f_{xtal}} \tag{2}$$

W obu wzorach f_{xtal} to częstotliwość rezonatora kwarcowego taktującego mikrokontroler, a n_{cykl} to ilość cykli maszynowych, które są niezbędne do wykonania procedur protokołu.

Tabela 1

Parametry implementacji

Implementacja	Bajtowa	Ramkowa
Środowisko	Z pamięcią wewnętrzną	Z pamięcią zewnętrzną
Mikrokomputer	89C52	80C320
Język programowania	Asembler	C
Rozmiar kodu wynikowego	3,5 kB	23 kB
Zajętość pamięci RAM	69 B	126 B
Rozmiar bufora	62 B	16 kB
Częstotliwość zegara (f_{xtal})	18,432 MHz	11,0592 MHz
Wersja IrCOMM	3-Wire	9-Wire
Czas przetwarzania bajtu	195 us	85 us
Czas przetwarzania ramki	135 us	300-700 us
Maks. Prędkość transmisji	38400 b/s	115200 b/s

Czas przetwarzania pojedynczego bajtu odgrywa kluczową rolę dla osiąganych prędkości transmisji. Czas ten obejmuje procedurę obsługi przerwania portu szeregowego oraz wszystkich procedur wywoływanych przy odebraniu lub nadawaniu pojedynczego bajtu. W ten czas jest również wliczona obsługa przerwania układu licznikowego obsługującego timery warstwy IrLAP. Nie są wliczone procedury warstwy aplikacyjnej, ponieważ nie zaliczają się do struktury protokołu. Mimo szybszego zegara taktującego, czas obsługi pojedynczego bajtu jest ponad dwukrotnie dłuższy w wersji bajtowej, przez co osiągane przez tę implementację prędkości transmisji są niższe. Maksymalną prędkość transmisji osiąganą przez każdą z implementacji można wyliczyć na podstawie wzoru (3).

$$S_{\max} = \frac{11}{t_{byte}} \tag{3}$$

Implementacja bajtowa może osiągnąć teoretycznie prędkość 56410 b/s, ramkowa 129411 b/s. Po zestawieniu obliczonych prędkości z prędkościami zdefiniowanymi przez standard IrDA osiągnąć można 38400 b/s dla wersji bajtowej i 115200 b/s dla wersji ramkowej.

Podany w tabeli czas przetwarzania ramki obejmuje czas, który upłynął od odebrania ostatniego bajtu w ramce do momentu rozpoczęcia nadawania ramki odpowiedzi. W tym przypadku również podane wartości uwzględniają tylko czas wykonania procedur samego

protokołu, nie obejmując czasu działania aplikacji nadrzędnej. Zgodnie z oczekiwaniami, czas przetwarzania ramki w wersji bajtowej jest znacznie krótszy niż w wersji ramkowej. Wynika to z faktu, iż przetwarzanie protokołu nastąpiło wcześniej, już przy odbiorze kolejnych bajtów. Po zakończeniu odbierania ramki pozostało jedynie sprawdzenie poprawności sumy kontrolnej CRC i poinformowanie warstwy aplikacji o poprawności lub błędzie transmisji. Dłuższy czas przetwarzania ramki dla implementacji ramkowej wynika również z faktu większego skomplikowania procedur protokołu. Czas przetwarzania ramki wpływa na wartość minimalnego czasu odpowiedzi (*min. turnaround time*). W protokole IrDA czas ten jest negocjowany przy nawiązywaniu połączenia IrLAP i może przyjmować wartości z zakresu od 0 do 10 ms. Jak widać z wyliczonych wartości, obie implementacje osiągają czas przetwarzania co najmniej o rząd wielkości krótszy od maksymalnego dopuszczonego przez standard.

Minimalna wielkość pola informacyjnego protokołu IrDA-Lite wynosi 62 bajty. Taką wielkość przyjęto przy realizacji protokołu w wersji bajtowej. Dla aplikacji nadrzędnej pozostaje około 60 bajtów pamięci wewnętrznej, co dla większości zastosowań jest ilością wystarczającą. W wersji ramkowej przyjęto maksymalną wielkość pola informacyjnego 1024 bajty. Wersja ta obsługuje okno transmisyjne o rozmiarze od 1 do 7 ramek, dzięki czemu możliwe jest przesłanie 7 kB danych bez zmiany kierunku transmisji. Dostępna pamięć o rozmiarze 32 kB wystarcza na niezależny bufor nadawczy i odbiorczy.

5. Podsumowanie

Analiza obu implementacji pokazała, że wyliczone parametry transmisji znacznie przewyższają minimalne parametry zdefiniowane przez wersję standardu IrDA-Lite. Przy odpowiednio napisanych programach aplikacyjnych parametry te nie powinny ulec znacznemu pogorszeniu.

W toku dalszych prac planuje się dokonanie badania rzeczywistych osiąganych przez obie wersje parametrów transmisyjnych i dokonanie modyfikacji prowadzących do ich poprawy, lub do poszerzenia funkcjonalności implementacji protokołu.

LITERATURA

1. Zieliński B., Tokarz K.: Transmisja bezprzewodowa w standardzie IrDA. ZN Pol. Śl. s. Informatyka z. 36, Gliwice 1999.
2. Tokarz K.: Implementacja protokołu IrDA w środowisku mikrokomputera jednoukładowego. Studia Informatica Volume 22, Number 1 (43), Gliwice 2001.

3. Standards. http://www.irda.org/standards/pubs/IrData.zip .
4. Minimal IrDA Protocol Implementation (IrDA Lite). Version 1.0. Infrared Data Association, 07.11.1996. http://www.irda.org/standards/pubs/litever10.pdf .
5. Dokumentacja techniczna. Atmel 8-bit microcontroller with 8K Bytes Flash AT89C52. http://www.atmel.com/atmel/acrobat/doc0313.pdf .
6. Dokumentacja techniczna. Vishay Telefunken TOIM3232. Rev. 1, 01.04.1999. http://www.vishay.com/docs/toim3232.pdf .
7. Dokumentacja techniczna. Vishay Telefunken TFDS4500. Rev. A1.1, 09.07.1999. http://www.vishay.com/docs/tfd_4.pdf .
8. Dokumentacja techniczna. Dallas DS80C320/DS80C323 High-Speed/Low-Power Micro. http://pdfserv.maxim-ic.com/arpdf/DS80C320-DS80C323.pdf .

Recenzent: Dr inż. Ryszard Winiarczyk

Wpłynęło do Redakcji 26 marca 2003 r.

Abstract

The scope of this paper is the presentation of two possible versions of IrDA protocol implementation in microcontroller's environment. First version, described in detail in [2], is byte oriented. It executes whole protocol stack for every byte received or transmitted. Second version, described in this paper, executes only framing layer procedures for every byte. The rest of protocol layers are executed after whole frame is received. This version has been written in C language and implemented on more complex hardware. Schema of the hardware is presented on fig.1. Chapter 4 of this paper presents comparison of those implementations paying attention to execution time and maximum speed of the transmission. Results of theoretical calculations are presented in the table 1.

Adresy

Krzysztof TOKARZ: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, tato@zeus.polsl.gliwice.pl .

Bartłomiej ZIELIŃSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, bmw@zeus.polsl.gliwice.pl .