Krzysztof ARNOLD, Sławomir MICHALAK
POLITECHNIKA POZNAŃSKA, WYDZIAŁ ELEKTRONIKI I TELEKOMUNIKACJI,
ul. Polanka 3, 60-965 Poznań

Czasowe uwarunkowania współpracy szeregowo-równoległego kontrolera CPLD z mikrokomputerem Raspberry PI i podsystemem PPI

Dr inż. Krzysztof ARNOLD

Absolwent Wydziału Elektroniki Politechniki Gdańskiej. Pracuje jako adiunkt w Katedrze Systemów Telekomunikacyjnych i Optoelektroniki na Wydziale Elektroniki i Telekomunikacji Politechniki Poznańskiej. W pracy naukowej zajmuje się problemami pomiarów charakterystyk i parametrów sygnałów stochastycznych, tematyką akwizycji danych w systemach pomiarowych oraz zagadnieniami projektowania, diagnostyki i rozwoju mikroprocesorowych systemów pomiarowych.

e-mail: karnold@et.put.poznan.pl

Dr inż. Sławomir MICHALAK

Pracuje, jako adiunkt w Katedrze Systemów Telekomunikacyjnych i Optoelektroniki na Wydziale Elektroniki i Telekomunikacji Politechniki Poznańskiej. W pracy naukowo-dydaktycznej zajmuje się zagadnieniami komputerowego wspomagania projektowania, symulacji układów elektronicznych, programowaniem układów mikroprocesorowych i układów programowalnych. Zajmuje się tematyką pozyskiwania informacji z inteligentnych czujników pomiarowych.

e-mail: michalak@et.put.poznan.pl

Streszczenie

W pracy omówiono właściwości komputerów Raspberry PI. Opisano system z Raspberry PI i rozszerzeniem portów równoległych, korzystający z łącza SPI. Przedyskutowano wymagania czasowe dla komunikacji kontrolera CPLD z komputerem Raspberry i podsystemem PPI. Wyznaczono czasy cykli zapisu i odczytu danych, realizowanych przez Raspberry PI podczas komunikacji z układem CPLD. Przedstawiono wyniki badań, pozwalające na oszacowanie szybkości transferu danych w systemie i wskazanie ograniczeń.

Słowa kluczowe: Warunki czasowe, Raspberry PI, CPLD, układy PPI.

Time conditions of cooperation of a serial-parallel CPLD controller, a Raspberry PI microcomputer and a PPI subsystem

Abstract

In this paper the hardware and software relations in data transfer between Raspberry Pi and peripheral PPI devices, via a CPLD controller are discussed. The necessity of increasing parallel I/O lines in a microprocessor system based on the Raspberry PI, a popular educational microcomputer module, is shown. An example of the system with the Raspberry PI, the SPI/PPI controller and programmable peripheral interface devices 82C55A is presented (Fig. 1). The time requirements for communication between the Raspberry PI central unit (BCM2835), the SPI/PPI controller and PPI devices are discussed and the examples of timing for 82C55A PPI read and write cycles are shown (Figs. 2 and 3). The software (based on C language and libraries) procedure of time T measurement, for sending of two bytes via SPI (SPI of BCM2835 works in standard master mode) and confirming /STR signal is presented (Fig. 4). The value of this time is not constant. It depends on a few components, also on the delays produced by the operating system. The experimental calculations were carried out for the mode value of T determined on 10000 samples (Fig. 5). The value of f_{SCK} (frequency clock for SPI) and for t_{SPI} (SPI time for one byte) as a function of a different SPI divider were analyzed (Figs. 6 and 7). Finally, the coefficient $2 \cdot t_{SPI}/T$ as a function of the SPI divider was determined and presented (Fig. 8). The obtained results showed the communication speed limitation and enabled us to choose the right SPI clock divider as well as to estimate the time of data transfer via the SPI interface implemented in the CPLD controller.

Keywords: time conditions, Raspberry PI, CPLD, PPI devices.

1. Wstęp

Jednopłytowy moduł komputerowy Raspberry PI, wielkością zbliżony do rozmiarów karty kredytowej, powstał z myślą o promowaniu nauczania technik komputerowych i programowania w szkołach. Zgodnie z tym przesłaniem Raspberry PI został zaprojektowany jako relatywnie tani komputer edukacyjny, mający cechy komputera osobistego [1]. Tak jak powszechnie teraz stosowane komputery PC, Raspberry PI może być w przyszłości wykorzystywany nie tylko do nauki programowania, ale i do pełnienia funkcji kontrolnych, sterujących i pomiarowych. Słuszność tej tezy potwierdza duża popularność, jaką urządzenie zdążyło już zdobyć. Powodzenie przedsięwzięcia stało się możliwe głównie dzięki połączeniu małych rozmiarów modułu komputerowego ze znaczną mocą obliczeniową. To, co jest zaletą projektu, niesie też jednak z sobą określone ograniczenia. Z założenia niewielkie rozmiary płyty komputera zapewniają jego mobilność, ale z drugiej strony utrudniają rozszerzanie wbudowanych zasobów i rozmieszczanie złącz. Z tego względu pojawiły się moduły towarzyszące, które zwiększają liczbę wejść i wyjść mikrokomputera, a także rozwiązania układowe, pozwalające na przyłączanie wybranych urządzeń zewnętrznych [2]. Komunikacja między nimi a jednostką centralną ma podstawowe znaczenie dla dalszych perspektyw rozwojowych systemów, konfigurowanych w oparciu o moduł Raspberry PI.

Możliwości komunikacji komputera Raspberry PI z otoczeniem są określane przez warunki sprzętowe i programowe. W warstwie sprzętowej Raspberry PI dysponuje 17 typowymi liniami I/O, podczas gdy programowa obsługa portu jest pośrednio zależna od wielozadaniowego systemu operacyjnego, zarządzającego pracą komputera. Powoduje to, że w systemach z Raspberry PI, pośredniczącym układem CPLD i zewnętrznym podsystemem portów równoległych [3], o szybkości obsługi urządzeń zewnętrznych może decydować nie warstwa sprzętowa, ale wydajność obliczeniowa jednostki centralnej. Konieczne jest zatem przeprowadzenie analizy, pozwalającej na wyznaczenie przepływności poleceń i danych pomiędzy komputerem Raspberry PI i kontrolerem CPLD oraz określenie szybkości obsługi dołączanych urządzeń zewnętrznych.

2. Właściwości sprzętowe i programowe komputera Raspberry PI

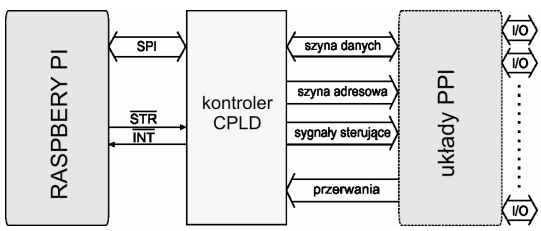
Architektura komputera Raspberry PI została oparta na układzie Broadcom BCM2835 typu SoC (*ang. System-on-a-chip*). Zawiera on kompletny system mikroprocesorowy, w którego skład wchodzi procesor ARM1176JZF-S taktowany zegarem 700 MHz, układ graficzny VideoCore IV GPU i 256 MB (Model A) lub 512 MB (Model B) pamięci RAM [1]. Raspberry PI posiada złącze dla kart pamięci SD. Na karcie pamięci instalowany jest system operacyjny oraz przechowywane są programy i dane. W zależności od wersji komputer wyposażono w jeden port USB (Model A), lub w dwa porty USB i kartę 10/100 Ethernet (Model B). Porty USB pozwalają na dołączenie urządzeń zewnętrznych, takich jak klawiatura, mysz, pamięci FLASH i twarde dyski. Karta Ethernet umożliwia dołączenie do internetu. Urządzenie posiada wyjście sygnału wideo HDMI oraz Composite Video, a także wyjście sygnału audio.

Raspberry PI dysponuje jednostką centralną o relatywnie dużej mocy obliczeniowej, dzięki czemu pracuje pod jedną z wersji systemu operacyjnego Linux (Debian, Fedora, Arch Linux), może też działać pod kontrolą systemów RISC OS i Android.

Moduł ma 17 linii GPIO (*General Purpose Input/Output*), wyprowadzonych bezpośrednio z układu BCM2835. Większość tych linii pełni również alternatywne funkcje, w tym odnoszące się do komunikacji z otoczeniem (interfejsy SPI, I^2C i UART).

3. System z komputerem Raspberry PI i rozszerzeniem portów równoległych

Charakterystyki systemowe komputerów Raspberry PI można znacznie poprawić, przyłączając do nich mikroprocesorowe układy transmisji równoległej PPI (*Peripheral Parallel Interface*). Architektura układów PPI pozwala na w pełni sprzętowe wsparcie komunikacji systemu z urządzeniami zewnętrznymi i programowe odciążenie jednostki centralnej, z którą współpracują za pośrednictwem magistrali [4]. Ponieważ linie magistrali obejmują szynę adresową, 8-bitową szynę danych i szynę sterującą, pomiędzy portem I/O komputera Raspberry a podsystemem PPI powinien znaleźć się dodatkowy układ (rys. 1), sterujący magistralą i pełniący funkcję kontrolera układów PPI [3].



Rys. 1. Schemat blokowy systemu z komputerem Raspberry PI i implementowanym kontrolerem układów PPI
Fig. 1. General block diagram of the microcomputer system with Raspberry PI and the implemented PPI's controller

Z punktu widzenia zajętości portu I/O Raspberry PI wskazane są rozwiązania, pozwalające na przyłączanie urządzeń zewnętrznych z wykorzystaniem szybkiej transmisji szeregowej. Kontrolery układów PPI i moduły transmisji, komunikujące się szeregowo z Raspberry PI, mogą być z powodzeniem implementowane w układach programowalnych.

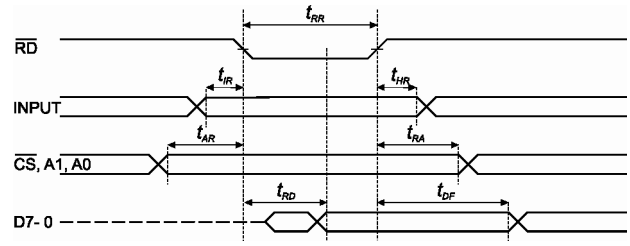
Jednostka centralna komunikuje się wówczas z programowalnym kontrolerem układów PPI przez interfejs SPI i odpowiada za obsługę tylko jednej linii przerwania zewnętrznego /INT (rys. 1). Praca kontrolera jest synchronizowana sygnałem /STR. Kontroler zaimplementowany w strukturze XC9572XL może sterować czterema układami 82C55A PPI, co odpowiada rozszerzeniu systemu o osiem 8-bitowych portów prowadzących transmisję równoległą z potwierdzeniem lub dwanaście portów w przypadku transmisji bez potwierdzenia. W trybie transmisji z potwierdzeniem każdy z portów układu PPI dostosowuje się indywidualnie do szybkości urządzenia zewnętrznego.

Połączenie modułu Raspberry Pi z kontrolerem CPLD angażuje tylko 5 linii I/O komputera.

4. Komunikacja kontrolera CPLD z podsystemem PPI

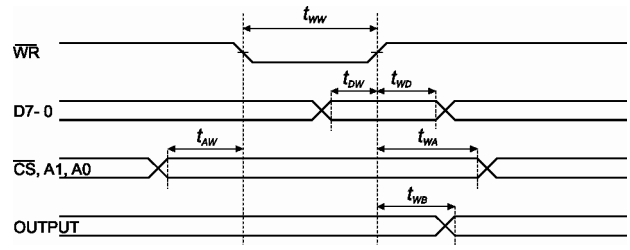
Układy 82C55A PPI przyjmują dane od urządzeń zewnętrznych lub przesyłają je na zewnątrz za pośrednictwem 8-bitowych portów, konfigurowanych w trzech trybach pracy [4]. Kontroler może być informowany o przebiegu transmisji przez przerwania i na tej podstawie generuje przerwanie /INT do komputera (rys. 1). Raspberry PI przekazuje do kontrolera polecenia odczytu danej wejściowej, przyjętej przez port układu PPI (rys. 2), lub zapisu danej wyjściowej, przeznaczonej do wysłania (rys. 3).

W cyklu odczytu kontroler adresuje żądający obsługi układ PPI (sygnały /CS, A1, A0), generuje sygnał odczytu /RD i przyjmuje bajt INPUT z magistrali danych D7-0. O długości cyklu decyduje suma czasów t_{AR}, t_{RR} i t_{DF} (rys. 2), która wynosi około 225 ns [4]. W cyklu zapisu kontroler adresuje układ PPI, wystawia wysyłany bajt na magistralę danych i generuje sygnał zapisu /WR. O długości cyklu decyduje suma czasów t_{AW}, t_{WW} i t_{WD} lub t_{WA} (rys. 3), wynosząca około 130 ns.



Rys. 2. Wykres czasowy cyklu odczytu portu w trybie 0
Fig. 2. Read cycle timing for mode 0

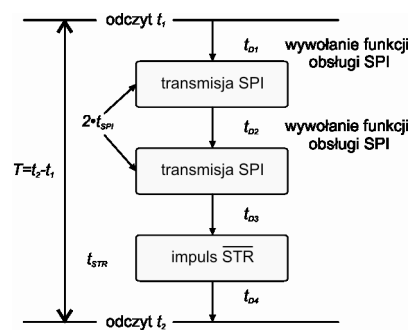
Ponieważ odstępy pomiędzy sygnałami /RD i /WR powinny przekraczać 300 ns, a czas t_{WB}, istotny dla urządzeń zewnętrznych, sięga 350 ns, częstotliwość generacji sekwencji odczytu i zapisu danych w trybie 0 nie powinna być większa niż 2 MHz (okres 0,5 µs). Wartość ta może być nieco różna tak dla wersji układu, jak i trybu jego pracy.



Rys. 3. Wykres czasowy cyklu zapisu portu w trybie 0
Fig. 3. Write cycle timing for mode 0

5. Komunikacja kontrolera CPLD z komputerem Raspberry PI

Układ BCM2835 wykorzystuje jako podstawę taktowania modułu SPI częstotliwość 250 MHz, dzieloną przez programowany współczynnik podziału 2^n, o minimalnej nastawie równej 2 [5]. W zestawieniu z częstotliwością zegarową układów CPLD, która dla struktury XC9572XL sięga 178 MHz [6], stwarza to perspektywę szeregowego transferu danych między Raspberry PI i kontrolerem, z szybkością znacznie przekraczającą wymagania obsługi rozbudowanego podsystemu PPI. Utrzymanie takiej szybkości jest jednak możliwe tylko w przedziale czasowym t_{SPI} (rys. 4), wyznaczonym przez taktowanie transmisji pojedynczego bajtu.



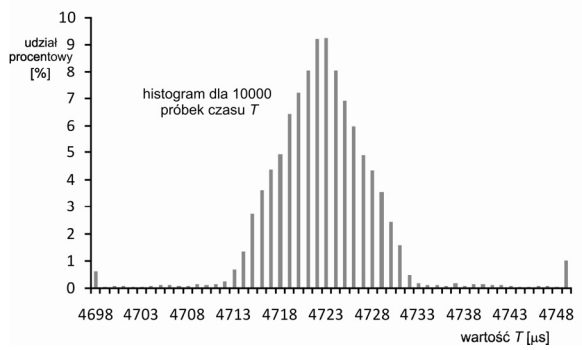
Rys. 4. Diagram czasowy sekwencji poleceń dla kontrolera CPLD
Fig. 4. Timing diagram of the instruction sequences for the CPLD controller

Pakiety transmisji, przeznaczone dla kontrolera CPLD, obejmują przesłanie dwóch bajtów i generację impulsu strobującego /STR. Oznacza to, że przesłaniom kolejnych bajtów towarzyszą przerwy czasowe t_{Di} w transmisji, wynikające z działania systemu operacyjnego i programu użytkownika (rys. 4).

Czas trwania sygnału /STR, wyznaczany programowo, nie powinien być krótszy od żadnego z czasów t_{RR} i t_{WW} (rys. 2, 3), a zatem od 150 ns [4]. Sumę czasów $2t_{SPI}$ i t_{STR} można więc teoretycznie zmniejszać do poziomu 278 ns. Rzeczywisty czas T nadawania pakietu danych, przeznaczonych dla kontrolera CPLD, może okazać się znacznie dłuższy (rys. 4).

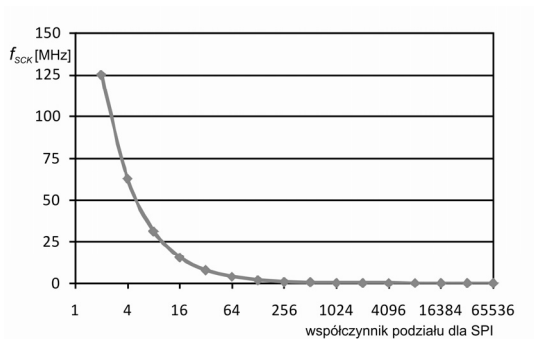
6. Wyniki badań parametrów czasowych pracy interfejsu SPI dla komputera Raspberry PI

Czas T nadawania pakietu danych wyznaczono jako różnicę znaczników czasowych t_2 i t_1, odczytanych z rozdzielczością 1µs z wykorzystaniem zegara programowego Raspberry PI (rys. 4). Czas T może się zmieniać, zwykle mieści się jednak w przedziale najczęściej występujących wartości (rys. 5).

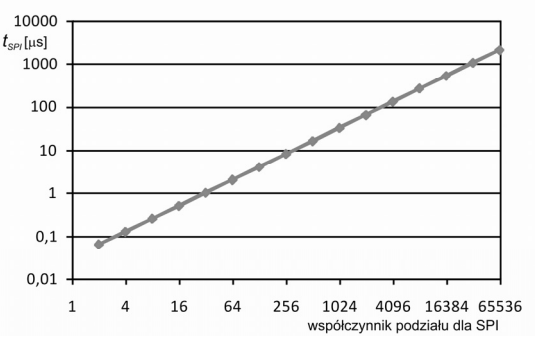


Rys. 5. Przykładowy rozrzut czasów nadawania pakietu danych
Fig. 5. An example of distribution for the data packet sending time

Interfejs SPI jest najszybszym łączem szeregowym komputera Raspberry PI. Maksymalna częstotliwość zegara transmisji f_{SCK}, dla podzielnika 2, wynosi 125 MHz (rys. 6), a odpowiadający jej czas t_{SPI} transmisji bajtu (8 taktów zegarowych) jest równy 64 ns (rys. 7). Transmisja w takich warunkach jest już w praktyce zależna od pojemności pasożytniczych i długości linii przesyłowej.

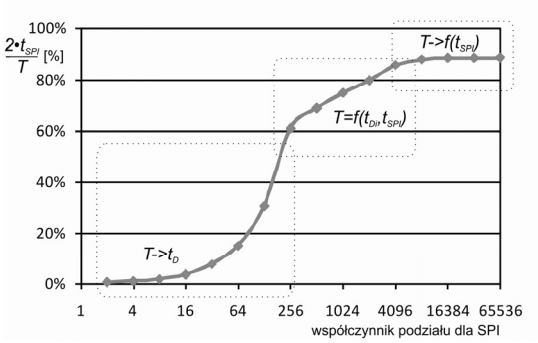


Rys. 6. Częstotliwość zegarowa SPI w funkcji współczynnika podziału SPI
Fig. 6. Serial clock frequency vs. SPI divider



Rys. 7. Zależność czasu transmisji bajtu od współczynnika podziału SPI
Fig. 7. Single byte transfer time vs. SPI divider

Udział czasu t_{SPI} w łącznym czasie nadawania pakietu danych T (dominanta) zależy od kilku czynników, w tym od współczynnika podziału, programującego częstotliwość zegarową na linii SCK interfejsu SPI (rys. 8). Dla współczynnika równego 64 czas T wynosi 27 µs, a dalsze zwiększanie częstotliwości f_{SCK} nie poprawia już znacząco sytuacji.



Rys. 8. Wskaźnik $2 \cdot t_{SPI}/T$ w funkcji współczynnika podziału SPI
Fig. 8. The coefficient $2 \cdot t_{SPI}/T$ vs. SPI divider

7. Podsumowanie

Funkcjonalność systemową komputera Raspberry Pi mogą zwiększyć programowalne kontrolery, sterujące magistralą zewnętrznych układów PPI i komunikujące się z Raspberry PI z wykorzystaniem szybkiego interfejsu SPI. Szybkość obsługi urządzeń współpracujących z Raspberry PI zależy jednak nie tylko od częstotliwości zegara interfejsu SPI, ale również od opóźnień wprowadzanych przez system operacyjny komputera oraz od wykorzystywanych funkcji i bibliotek programowych. Z tego względu właściwości układów FPGA i CPLD [6, 7], obiecujące uzyskanie dużej szybkości wymiany danych z Raspberry PI z jednej, a urządzeniami zewnętrznymi z drugiej strony, mogą być wykorzystane do granicy, wyznaczonej przez działanie oprogramowania zainstalowanego w komputerze.

Dla środowiska systemu operacyjnego Linux i języka programowania C zmniejszanie współczynnika podziału poniżej wartości 64 przynosi znaczny wzrost częstotliwości zegara transmisji, ale nie przekłada się istotnie na zwiększenie szybkości obsługi urządzeń zewnętrznych. Czas transmisji bajtu wynosi wówczas 2,048 µs, natomiast rzeczywisty czas nadawania pakietu danych zajmuje aż 27 µs. Ogranicza to szybkość dostępu do urządzeń zewnętrznych do około 38 tysięcy cykli zapisu lub odczytu na sekundę. Wykorzystanie możliwości kontrolerów CPLD w systemach z Raspberry PI wymaga więc zdecydowanie poszukiwania rozwiązań również w warstwie programowej.

Praca realizowana w ramach tematu 08/83/DSPB/4709.

8. Literatura

[1] Upton E., Halfacree G.: Meet the Raspberry Pi. Wiley, 2012.
[2] Johnson G.D., Krusienski D.J.: A Low-Cost Configurable Multichannel Cortical Stimulator Prototype. 6th Annual International IEEE EMBS Conference on Neural Engineering. San Diego, California, 6–8 November, 2013, p. 641-644.
[3] Arnold K., Michalak S.: Programowalny kontroler mikroprocesorowych układów transmisji równoległej z interfejsem SPI. Pomiary Automatyka Kontrola, vol.59, nr 8/2013, s.803-805.
[4] 82C55A Data Sheet FN2969.10. Intersil 2006.
[5] BCM 2835 ARM Peripheral. Product Specification, Broadcom Europe Ltd. 2012.
[6] XC9572XL High Performance CPLD. Product Specification. Xilinx 2007.
[7] Maśnicki R., Hallmann D.: Konwersja danych pomiędzy interfejsami Link Port i SPI. Pomiary Automatyka Kontrola, vol. 57, nr 12/2011, s.1466-1468.

otrzymano / received: 12.04.2014
przyjęto do druku / accepted: 02.06.2014 *artykuł recenzowany / revised paper*