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

Mikrokomputer Raspberry Pi jako sterownik systemu pomiarowego

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 artykule przedstawiono system pomiarowy, w którym nadrzędną rolę sprawuje mały, cieszący się coraz większą popularnością, mikrokomputer edukacyjny Raspberry Pi. System zaprojektowany został do wizualizacji rozkładu temperatury wewnątrz struktury układu reprogramowalnego FPGA, na podstawie dokonanych pomiarów częstotliwości oscylatorów pierścieniowych zaimplementowanych wewnątrz układu. Sterowanie procesem pomiarowym, akwizycja danych i prezentacja wyników nadzorowana jest przez mikrokomputer Raspberry Pi.

Słowa kluczowe: Raspberry Pi, oscylator pierścieniowy, FPGA.

Raspberry Pi as a measurement system control unit

Abstract

In this paper the system based on Raspberry Pi, a popular educational microcomputer [1] is described. In this system, a programmable FPGA Spartan-3 XC3S200 [5] device was tested. The Raspberry Pi worked as a control unit for the whole system (Fig. 1). A part of the system was implemented inside the tested structure (Fig. 2). It was an array of ring oscillators (Fig. 3), as temperature sensors, with a structure for controlling the ring oscillators. Simple ring oscillators are often implemented in FPGA devices. They are used both as a single element or an array of sensors for measuring the chip temperature [2, 3, 4]. The frequency of the activated sensor was measured outside by an oscilloscope (SCPI command was used). The frequency was dependent on temperature. The sensors can be located in different areas of a chip [6, 7]. In case of the tested device 36 sensors were used, but generally it depends on a tested device [8, 9]. The Raspberry Pi controlled the measurement process via an SPI serial interface. The results were collected from the oscilloscope via a UART/RS232 serial interface. The relation between frequency and temperature (Fig. 4) as well as 2D visualizations (Fig. 5) were made using Gnuplot and Scilab. The results should visualize the temperature distribution inside the device, but first right calibration of sensors should be made. The location of elements inside the FPGA sensor is of great significance [10], so in the case of an array of sensors, each ring oscillator should be analyzed and calibrated independently.

Keywords: Raspberry Pi, ring oscillator, FPGA.

1. Wprowadzenie

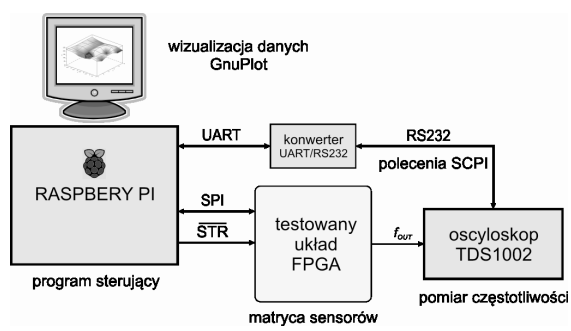
Raspberry Pi - mały, jednopłytowy moduł komputerowy o wymiarach 86 x 56 mm i wadze 45 g., wielkością zbliżony do rozmiaru karty kredytowej - został zaprojektowany, jako relatywnie tani komputer edukacyjny [1]. Opracowany przez Raspberry Pi Foundation (przy wsparciu University of Cambridge Laboratory i Broadcom Corporation), ma na celu przede wszystkim pomóc w promowaniu nauczania technik komputerowych i programowania w szkołach.

Stosunkowo niska cena i relatywnie duże możliwości obliczeniowe jednostki centralnej sprawiają, że Raspberry Pi jest coraz częściej wykorzystywany nie tylko w celach edukacyjnych, ale również (pojedynczo lub w zestawach) do sterowania procesami czy nadzorowania systemów pomiarowych. Postęp technologiczny i coraz powszechniejsze zastosowanie urządzeń z procesorami ARM skłania do przypuszczenia, że idea szerokiego wykorzystywania małych, łatwo dostępnych i dysponujących coraz większą mocą obliczeniową modułów komputerowych będzie kontynuowana. Ich potwierdzona już przydatność w wielu dziedzinach będzie rozwijana a obszar zastosowań i droga rozwoju wydaje się być podobna, jak w przypadku dotychczasowego rozwoju i zastosowań komputerów PC.

Moduł mikrokomputera Raspberry Pi oparty jest na układzie Broadcom BCM2835 typ SoC. Ogólnie pojęciem SoC (*ang. System-on-a-chip*) określa się struktury zawierające kompletny system mikroprocesorowy, a więc jednostkę centralną CPU, bloki pamięci FLASH lub ROM, EEPROM i RAM, układy cyfrowe i analogowe, a także przetworniki analogowo-cyfrowe i cyfrowo-analogowe oraz interfejsy szeregowe. W odróżnieniu od mikrokontrolerów, układy SoC są wyposażone w jednostkę centralną o stosunkowo dużej mocy obliczeniowej, umożliwiając uruchamianie systemów operacyjnych. Układ Broadcom BCM2835 zawiera procesor ARM1176JZFS, układ graficzny VideoCore IV GPU oraz 256 MB (Model A) lub 512 MB (Model B) pamięci RAM. Raspberry Pi posiada złącze dla kart pamięci SD. Na karcie pamięci instalowany jest system operacyjny, jest ona również używana do przechowywania danych.

2. Opis systemu pomiarowego

Schemat blokowy proponowanego układu pomiarowego przedstawiono na rysunku 1. Obiektem badanym jest układ reprogramowalny FPGA, w którego strukturze zaimplementowano matrycę czujników temperatury. Mikrokomputer Raspberry Pi nadzoruje pracą całego systemu pomiarowego. Jako czujniki temperatury zastosowano oscylatory pierścieniowe zbudowane z łańcucha nieparzystej liczby inwerterów. Ich parametry dynamiczne, a co za tym idzie częstotliwość pracy oscylatorów, zależne są m.in. od temperatury. W pracach [2, 3, 4] podawane są przykłady wyników badań zależności częstotliwości oscylatorów pierścieniowych od temperatury.



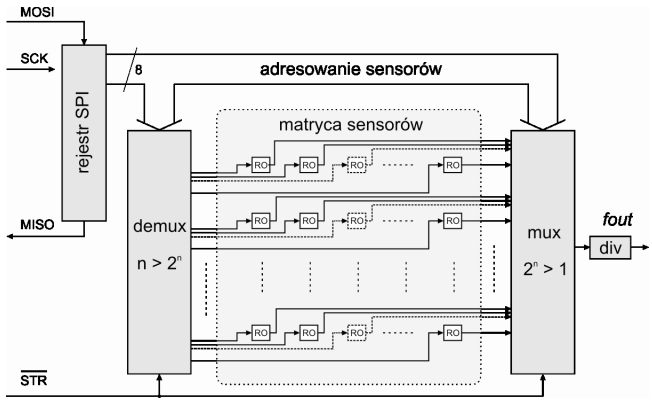
Rys. 1. Schemat blokowy systemu pomiarowego
Fig. 1. Block diagram of the measurement system

Moduł Raspberry Pi, poprzez interfejs SPI, steruje matrycą czujników, a poprzez interfejs szeregowy RS232 nadzoruje pracę oscyloskopu cyfrowego, którego zadaniem jest pomiar częstotliwości. Testy funkcjonalne systemu przeprowadzono z układem Spartan-3A XC3S200 (Xilinx) [5].

Schemat blokowy struktury pomiarowej zaimplementowanej wewnątrz układu XC3S200 przedstawiono na rysunku 2. W układzie zaprogramowano 36 czujników temperatury, jako matrycę o wymiarach 6x6, rozmieszczonych równomiernie na całym obszarze badanego układu. Liczba i miejsca umieszczenia poszczególnych oscylatorów zależne są przede wszystkim od zasobów

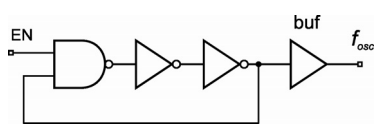
badanego układu oraz od wielkości obszaru struktury wybranej do analizy. Reprogramowalność układów FPGA umożliwia zmianę sposobu rozmieszczenia oscylatorów i ograniczenie badań tylko do wybranego fragmentu. Na etapie testowania systemu zdecydowano się na obrazowanie całego obszaru badanego układu FPGA.

Zwiększanie liczby zaimplementowanych oscylatorów, w celu wierniejszego odwzorowania warunków cieplnych wewnątrz układu, nie zawsze jest wskazane. Działanie oscylatorów pierścieniowych powoduje wystąpienie zjawiska samopodgrzewania, zatem raczej poszukuje się takich rozwiązań, które pozwalają na minimalizowanie liczby sensorów i optymalizację ich rozmieszczenia na obszarze badanej struktury [6, 7].



Rys. 2. Struktura pomiarowa implementowana w układzie FPGA
Fig. 2. Measurement structure implemented inside the FPGA device

W testowanym systemie pomiarowym mikrokomputer Raspberry Pi steruje matrycą czujników poprzez interfejs szeregowy SPI jako układ *master*. W badanej strukturze FPGA zaimplementowano uproszczony sterownik SPI pracujący w trybie *slave*. Jednostka *master* wystawia *n*-bitowe słowa sterujące na linii MOSI oraz takty zegara na linii SCK. Jednocześnie odczyt sygnału MISO umożliwia weryfikację wcześniej wysyłanych bajtów, co pozwala na kontrolę poprawności przesyłania słów sterujących. Przesłanie całego bajtu potwierdzane jest sygnałem /STR, co dodatkowo zmniejsza ryzyko błędnego przesłania bajtu i niewłaściwej interpretacji słowa sterującego w przypadku wystąpienia zakłóceń na linii danych lub utraty sygnału zegarowego.



Rys. 3. Układ oscylatora pierścieniowego z wejściem inicjującym
Fig. 3. A ring oscillator circuit with enable input

Uaktywnienie wybranego czujnika odbywa się poprzez wysłanie słowa adresującego o długości *n*. Demultiplekser (DEMUX) uaktywnia jedną z 2^n linii wyjściowych, aktywując wejście EN wybranego oscylatora (rys. 3) na czas określony przez długość trwania impulsu /STR. Sygnał f_{osc} z aktywnego czujnika, poprzez adresowany multiplekser (MUX), podawany jest przez dzielnik częstotliwości na wyjście układu jako sygnał f_{out}.

W obecnej wersji systemu sterowanie odbywa się słowem 8-bitowym, co umożliwia zaadresowanie maksymalnie 2^8=256 oscylatorów i wydaje się, że jest to liczba wystarczająca nawet w przypadkach analizy układów programowalnych o znacznie większych zasobach niż układ XC3S200 [8, 9]. Zwiększenie liczby czujników można uzyskać poprzez modyfikację układu multipleksera i demultipleksera, a struktura interfejsu SPI jako przesuwnego rejestru szeregowego umożliwia stosunkowo proste zwiększenie liczby *n* bitów słowa adresowego. Przy większej liczbie czujników warto rozważyć wierszowo-kolumowe aktywowanie matrycy. Do pomiaru częstotliwości wybranego oscylatora pierścieniowego zastosowano oscyloskop TDS1002 Tektronix wyposażony w interfejs szeregowy RS232.

Mikrokomputer Raspbery Pi wyposażony został w UART w bardzo okrojonej wersji (tzw. mini UART). Zubożona wersja mini UART umożliwia przesyłanie ramki złożonej z 7 lub 8 bitów danych, 1 bitu startu i 1 bitu stopu, bez bitu parzystości. Uproszczona wersja interfejsu nie umożliwia generowania sygnałów sterujących DCD, DSR, DTR i RI, generowania bitu parzystości i wykrywania błędów ramki, a rejestr nadawczo odbiorczy nie jest w pełni kompatybilny z układem 16550. Dla użytkownika dostępne są jedynie sygnały RxD i TxD, co pozwala na realizację prostej komunikacji asynchronicznej i sterowanie pracą zewnętrznego urządzenia pomiarowego (oscyloskopu, częstościomierza) wyposażonego w interfejs RS232.

Jednostka centralna mikrokomputera Raspberry Pi zasilana jest napięciem 3,3 V, również porty I/O, w tym UART, kompatybilne są z poziomami napięć 3,3 V. Aby zapewnić zgodność ze standardem RS232 zastosowano typowy konwerter poziomów UART/RS232 (układ MAX3232 dedykowany do pracy z poziomami napięć 3,3 V).

Sterowanie pracą oscyloskopu realizowane jest poprzez komendy sterujące języka SCPI przesyłane przez port szeregowy RS232C. Należy pamiętać, że choć polecenia języka SCPI stworzone zostały jako standard obsługi urządzeń pomiarowych, to jednak urządzenia poszczególnych producentów mogą wymagać nieco odmiennej składni poleceń sterujących. Przykładowe komendy SCPI sterujące pracą oscyloskopu, umożliwiające dokonanie odczytu mierzonej częstotliwości, przedstawiono w tabeli 1 (składnia poleceń dla oscyloskopu Tektronix).

Tab. 1. Polecenia SCPI dla oscyloskopu TDS1002
Tab. 1. SCPI commands for the TDS1002 oscilloscope

/identyfikacja urządzenia – zapytanie/ ***IDN?**\n
/odpowiedź oscyloskopu/ **TEKTRONIX, TDS 1002,0, CF.99.1.CT FV.v2.12 TDS2CM:CMV:v1.04**
/wybór trybu pomiaru częstotliwości/ **MEASU**rement:**IMM**ed:**TYP**e **FREQ**ency\n
/zapytanie o wynik/ **MEASU**rement:**IMM**ed:**VAL**ue?\n
/odpowiedź oscyloskopu – wynik pomiaru częstotliwości/ **3.1289112E7**\n

3. Program sterujący pracą systemu

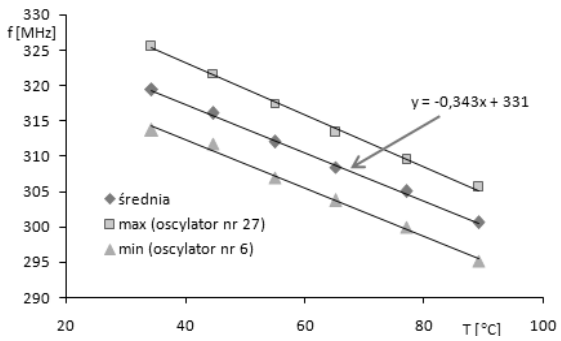
Nadzór nad pracą systemu sprawuje aplikacja napisana w języku C. Domyślnie, w mikrokomputerze Raspberry Pi pracującym pod kontrolą systemu Linux, port szeregowy przypisany jest do obsługi konsoli terminalowej. Aby bez konfliktów wykorzystywać zasoby UART, konieczna była modyfikacja domyślnych ustawień w plikach konfiguracyjnych. Obsługę programową portów I/O, w tym linii SPI i UART, ułatwiają biblioteki dedykowane dla jednostki centralnej BMC2835 (np. biblioteki WiringPi, BMC2835). Przykładowo, w programie testowym wykorzystano funkcje biblioteki BMC2835 do obsługi magistrali SPI.

Moduł Raspberry Pi pracuje pod kontrolą powszechnie dostępnego (*ang. Open Source*) systemu operacyjnego Linux (Rasbian). Zgodnie z tą ideą program w języku C, sterujący pracą systemu pomiarowego, napisano w ogólnodostępnym środowisku uruchomieniowym (edytor Geany), a do wizualizacji wyników pomiarów wykorzystano dostępne w zasobach wolnego oprogramowania środowisko Gnuplot. Program sterujący realizuje następujące funkcje: inicjalizacja magistrali SPI, inicjalizacja linii UART, adresowanie/aktywowanie kolejnych czujników matrycy sensorów, pomiar częstotliwości, wyświetlanie aktualnych pomiarów na konsoli terminalowej, zapis wyników pomiarów w formacie akceptowalnym przez program Gnuplot, wywołanie środowiska Gnuplot i wizualizacja wyników w formie wykresu 3D.

W programie uwzględniono możliwość zbierania danych w określonych, zadanych programowo, odstępach czasowych, celem generowania sekwencji plików graficznych, które w dalszej części są podstawą tworzenia animacji obrazujących zmiany mierzonej wielkości w czasie trwania procesu pomiarowego.

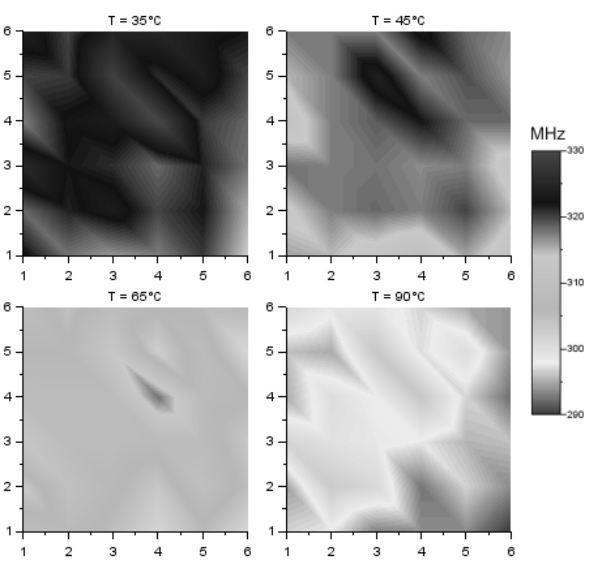
4. Testy systemu i wizualizacja wyników

Testy funkcjonalne systemu i pomiary częstotliwości matrycy oscylatorów przeprowadzono dla przypadku, gdy w badanym układzie FPGA zaimplementowana została tylko struktura pomiarowa (rys. 2). Nie zostały zaprogramowane inne elementy powodujące, jako efekt uboczny lub celowy, dodatkowe podgrzewanie układu. Podczas testów sprawdzono poprawność implementacji matrycy oscylatorów i działania interfejsu SPI. Kolejno testowano rozkazy sterujące aktywacją wybranego oscylatora i pozyskiwania wyników do mikrokomputera Raspberry Pi.



Rys. 4. Częstotliwości oscylatorów w funkcji zmian temperatury układu
Fig. 4. Ring oscillator frequency as a function of chip temperature

Dla każdego z 36 czujników zarejestrowano zmiany częstotliwości przy ogrzewaniu układu (nagrzewanie strumieniem gorącego powietrza). Temperaturę obudowy układu kontrolowano termometrem IR (NC100). Na rysunku 4 przedstawiono zależność zmian częstotliwości od temperatury dla trzech przypadków: oscylatorów o najwyższej i najniższej częstotliwości, oraz wyznaczoną średnią wartość dla wszystkich 36 czujników. Na jej podstawie określono przykładową, uśrednioną wartość współczynnika zmian temperaturowych: -0,343 MHz/°C. Wizualizacje 2D interpolowanego rozkładu wyników pomiarów częstotliwości dla kilku wybranych temperatur pokazano na rys. 5.



Rys. 5. Interpolowany rozkład częstotliwości oscylatorów - wizualizacja 2D
Fig. 5. Interpolated frequency of the ring oscillators - 2D visualization

Do wizualizacji wyników pomiarowych wykorzystano, wywoływane z poziomu programu C, środowisko Gnuplot. Jest to środowisko przeznaczone do graficznej prezentacji danych pomiarowych i choć posiada wbudowane podstawowe funkcje matematyczne, to nie jest dedykowanym środowiskiem obliczeniowym. Dlatego dalszą analizę wyników przeprowadzono w środowisku obliczeniowym Scilab.

5. Podsumowanie

Na etapie wstępnych testów systemu nie można uzyskanych wyników pomiarów częstotliwości wprost interpretować jako wyniki pomiarów temperatury, co pozwoliłoby na zobrazowanie rozkładu temperatury wewnątrz struktury. Mimo, że zastosowane oscylatory posiadają taką samą strukturę (złożone są z takiej samej, nieparzystej liczby inwerterów), implementowane są w różnych miejscach układu FPGA, co powoduje, że posiadają odmienne parametry dynamiczne [10]. Konieczne jest, dla każdego czujnika indywidualnie, wyznaczenie zależności zmian częstotliwości od temperatury a następnie przeprowadzenie korekcji wyników, co w założeniu autora jest celem dalszych zaplanowanych prac. Do celów kalibracji przewiduje się podgrzewanie/chłodzenie układu z wykorzystaniem ogniwa Peltiera. Po opracowaniu procedury kalibracji system będzie wykorzystywany do badań właściwości termicznych struktur reprogramowalnych.

Zastosowanie modułu Raspberry Pi, jako urządzenia nadzorującego pracę systemu pomiarowego, jest interesującą alternatywą dla systemów pomiarowych nadzorowanych tradycyjnie przez komputery klasy PC. Mniejsze (choć, nie tak małe) możliwości obliczeniowe jednostki centralnej BMC2835 oraz ograniczone zasoby pamięci, nie pozwalają w pełni konkurować z zawansowanymi komputerami klasy PC. Jednak cechy takie jak: możliwości pracy pod jedną z dystrybucji uniwersalnego systemu operacyjnego Linux, zdolność komunikowania się z urządzeniami pomiarowymi nie tylko poprzez tradycyjne interfejsy komunikacyjne, ale również poprzez linie I/O i magistrale szeregowe SPI i I2C, dostępne programowo zarówno z poziomu języków wysokiego poziomu (C, Python), jak i z poziomu asemblera, pozwalają na zastosowanie modułu jako elementu pomiarowych systemów wbudowanych, współpracującego z wyodrębnionymi podzespołami układów mikroprocesorowych i reprogramowalnych.

6. Literatura

[1] Upton E., Halfacree G.: Meet the Raspberry Pi. Wiley, 2012.
[2] Michalak S.: Badanie właściwości oscylatora pierścieniowego w temperaturze 77K. XV Konferencja Naukowa Reprogramowalne Układy Cyfrowe – RUC 2012, Szczecin, Pomiary Automatyka Kontrola, vol. 58, nr 7, s. 681-683, 2012.
[3] Lopez-Buedo S., Garrido J., Boemo E.: Thermal testing on programmable logic devices., Proceedings of the ISCAS'98 IEEE International Symposium on Circuits and Systems, vol. 2, pp. 240-243, 1998.
[4] Franco J.J.L., Boemo E., Castillo E., Parrilla L.: Ring oscillators as thermal sensors in FPGAs: Experiments in low voltage. In Programmable Logic Conference (SPL 2010), Southern, pp. 133-137, 2010.
[5] Spartan-3 Generation FPGA User Guide, UG331 (v1.8) June 13, 2011, www.xilinx.com
[6] Memik S.,O., Mukherjee R., Ni M., Long J.: Optimizing thermal sensor allocation for microprocessors, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 27, no. 3, pp. 516-527, 2008.
[7] Mukherjee R., Mondal S., Memik S.O.: Thermal sensor allocation and placement for reconfigurable system. Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design, pp. 437-442, 2006.
[8] Happe M., Agne A., Plessl C.: Measuring and predicting temperature distributions on FPGAs at run-time. International Conference on Reconfigurable Computing and FPGAs (ReConFig), pp. 55-60, 2011.
[9] Antola A., Piuri V., Sami M.: On-line diagnosis and reconfiguration of FPGA systems. Proceedings of The First IEEE International Workshop on Electronic Design, Test and Applications, pp. 291-296, 2002.
[10] Kwiatkowski P., Szymanowski R., Szplet R.: Identyfikacja parametrów dynamicznych linii szybkich przeniesień oraz globalnych linii zegarowych w układach programowalnych Spartan-6, Pomiary Automatyka Kontrola, vol. 59, nr 8, s.757-759, 2013.

otrzymano / received: 15.05.2014
przyjęto do druku / accepted: 01.07.2014 *artykuł recenzowany / revised paper*