Grzegorz LENTKA, Mariusz CHILMON
POLITECHNIKA GDAŃSKA,
ul. Narutowicza 11/12, 80-233 Gdańsk

Energooszczędny kontroler systemu pomiarowego na bazie sieci Ethernet

Dr inż. Grzegorz LENTKA

Ukończył studia na Wydziale ETI Politechniki Gdańskiej w 1996 r. uzyskując dyplom magistra inżyniera o specjalności Systemy Pomiarowe. W tym samym roku podjął pracę w Katedrze Miernictwa Elektronicznego PG. W 2003 obronił pracę doktorską. Zajmuje się projektowaniem systemów pomiarowo-diagnostycznych układów elektronicz-nych i obiektów technicznych oraz wykorzystaniem DSP w pomiarach impedancji. Autor i współautor ponad 70 publikacji i 2 patentów.

e-mail: lentka@eti.pg.gda.pl

Mgr inż. Mariusz CHILMON

Ukończył studia na Wydziale ETI Politechniki Gdańskiej w 2010 r. uzyskując dyplom magistra inżyniera o specjalności Komputerowe Systemy Elektroniczne. Zainteresowania obejmują konstrukcje systemów wbudowanych, dedykowane dystrybucje systemu Linux, aplikacje bazodanowe, wykorzystanie języków skryptowych. Zawodowo zajmuje się tworzeniem oprogramowania na systemy wbudowane. Obecnie pracuje nad rozwijaniem central alarmowych.

e-mail: vmario@vmario.org

Streszczenie

Współpraca wielu urządzeń pomiarowych czy długotrwała akwizycja danych napotyka problemy: brak jednolitego interfejsu umożliwiającego użytkownikowi proste zarządzanie zbieraniem danych i przejrzystą ekspozycję danych, zużycie energii komputera nadzorującego ciągłą pracę sieci przyrządów pomiarowych. Rozwiązaniem może być zastosowanie autonomicznego kontrolera procesu akwizycji danych umożliwiającego konfigurację parametrów pracy i odczyt wyników.

Słowa kluczowe: rozproszone systemy pomiarowe.

Low-power controller for measurement systems based on Ethernet

Abstract

Multiple measurement instruments cooperation as well as long-lasting data acquisition can arise problems due to lack of unified interface for data collection management and data visualization and also energy consumption of computer controlling instruments network. These problems can be solved using autonomous acquisition controller allowing to configure parameters and to view collected data. The device is based on single-board computer with ARM9 processor working under Linux operating system. The application running on the controller allows performing multiple acquisition schedules and uses database approach for data collection. The controller can be managed using web-based user interface, which gives high flexibility and is operating system independent. The main advantage of this solution is low power consumption: the device uses below 1.5W.

Keywords: distributed measurement systems.

1. Wprowadzenie

Współczesne przyrządy pomiarowe na ogół wyposażone są w interfejs umożliwiający komunikację z komputerem. Zwiększa to funkcjonalność urządzenia, dając możliwość automatycznej akwizycji danych, archiwizowania wyników pomiarów i poddawania ich złożonej obróbce. Mimo upływu lat wciąż używane są interfejsy IEEE-488 i RS-232, jednak coraz częściej zastępowane są przez USB. Wadą wymienionych interfejsów jest mały zasięg i trudność łączenia przyrządów pomiarowych w sieć.

Rozwiązaniem jest skorzystanie z technologii Ethernet. Urządzenia obsługujące ten standard są interesujące dla projektanta — można je w prosty sposób łączyć w sieć i obsługiwać zdalnie z praktycznie każdego miejsca z dostępem do sieci Internet. Mimo złożoności protokołu TCP/IP nie napotyka się na duże trudności programistyczne, gdyż istnieje wiele gotowych implementacji, a praktycznie każdy system operacyjny udostępnia API ułatwiające pisanie aplikacji sieciowych. Nie ma także problemu z nadzorem takiej sieci przez komputery PC, gdyż są one zazwyczaj wyposażone w złącze RJ45, a coraz częściej w bezprzewodowy interfejs Wi-Fi.

W przypadku współpracy wielu urządzeń pomiarowych różnych producentów czy długotrwałej akwizycji danych napotyka się jednak na pewne problemy: brak ujednoliconego interfejsu graficznego, który umożliwiałby użytkownikowi proste zarządzanie harmonogramem zbierania danych i przejrzystą ekspozycję danych oraz zużycie energii i koszt komputera, który pracuje w trybie ciągłym, nadzorując pracę sieci przyrządów.

Dla autonomicznego kontrolera systemu pomiarowego zdefiniowano następujące wymagania:
- Praca w sieci Ethernet - dostęp do przyrządów odbywa się w sieci LAN/WAN rozpoznawanych po adresie IP;
- Użytkownik ma możliwość zdalnego dostępu do interfejsu sterownika przez sieć Internet;
- Wykorzystanie serwera HTTP po stronie kontrolera, a przeglądarki internetowej po stronie użytkownika upraszcza sterowanie, a ponadto umożliwia korzystania z urządzenia bez potrzeby instalacji sterowników i oprogramowania. Uniezależnia to użytkownika od systemu operacyjnego,
- Łatwość definiowania sterowników przyrządów pomiarowych - użytkownik musi mieć możliwość łatwego tworzenia własnych sterowników;
- Tworzenie harmonogramów akwizycji - każdy z przyrządów może mieć przypisany indywidualny harmonogram automatycznego dokonywania pomiarów;
- Prezentacja danych — możliwe jest przeglądanie danych pomiarowych przez interfejs webowy, a także pobranie ich w wygodnym formacie np. CSV (*Comma Separated Values*);
- Kontroler powinien mieć małe wymiary i niski pobór mocy.

2. Dobór technologii realizacji kontrolera

Obecnie na rynku istnieje wiele rozwiązań, pozwalających budować urządzenia korzystające z sieci Ethernet. Nawet 8-bitowe mikrokontrolery nierzadko zapewniają moc obliczeniową wystarczającą do implementacji stosu TCP/IP, a niektóre z nich są wzbogacane o sprzętowe moduły MAC (*Media Access Control*) czy zarówno MAC, jak i PHY (*Physical Layer*) [1]. Wraz ze wzrostem funkcjonalności technologii i wygody jej użycia, rosną koszty, wymiary i masa urządzenia oraz zapotrzebowanie na energie, zatem ostateczny wybór musi być kompromisem między funkcjonalnością i wygodą, a kosztem urządzenia.

Niewątpliwą zaletą mikrokontrolerów 8-bitowych jest stosunkowo niska cena i małe zapotrzebowanie na energię. Mikrokontroler ATmega128, którego koszt wynosi około 20 PLN, przy taktowaniu zegarem 16 MHz zużywa około 30 mA prądu. Najtańsze mikrokontrolery nie posiadają modułów MAC ani PHY, ale po podłączeniu zewnętrznego kontrolera i zainstalowaniu systemu RTOS ze wbudowanym stosem TCP/IP, łatwo je wykorzystać w sieci Ethernet. Umożliwia to np. system Contiki, przeznaczony przede wszystkim do sieci sensorowych, także bezprzewodowych [2]. Interesującym konkurentem Contiki jest Nut/OS, który oprócz protokołów takich jak ARP, TCP/IP i DNS dostarcza gotowe

funkcje serwerów FTP i HTTP. Ponadto system obsługuje karty pamięci z systemami plików FAT16 i FAT32 [3]. Mimo stosunkowo dużych możliwości urządzenie oparte na mikrokontrolerze 8-bitowym jest słabo skalowalne i należy liczyć się z trudnościami w przypadku obsługi urządzeń peryferyjnych takich jak dyski twarde czy karty Wi-Fi USB. Także implementacja skomplikowanych protokołów, np. kryptologicznych, jak TLS (*Transport Layer Security*), może sprawić problemy.

Wykorzystanie mikrokontrolera 32-bitowego daje znacznie większe możliwości, w szczególności rodzina rdzeni ARM, oprócz wydajnego CPU, oferuje peryferia obsługujące USB, Ethernet czy karty pamięci SD. Oprócz sprzętowego wsparcia dla wymienionych w założeniach technologii, architektura ARM jest interesująca także z powodu dostępności rozbudowanych systemów operacyjnych, jak GNU/Linux czy BSD. Systemy te posiadają biblioteki i sterowniki obsługujące urządzenia Wi-Fi i modemy oraz nośniki danych, jak dyski twarde IDE czy karty pamięci SD, a także możliwość wykorzystania języków skryptowych. W zamian za znacznie większe zasoby sprzętowe i bogatsze oprogramowanie należy liczyć się z większą ceną i zapotrzebowaniem na energię. Mikrokontroler AT91SAM9260 przy taktowaniu 180 MHz zużywa 130 mA [4], a koszt to około 70 PLN.

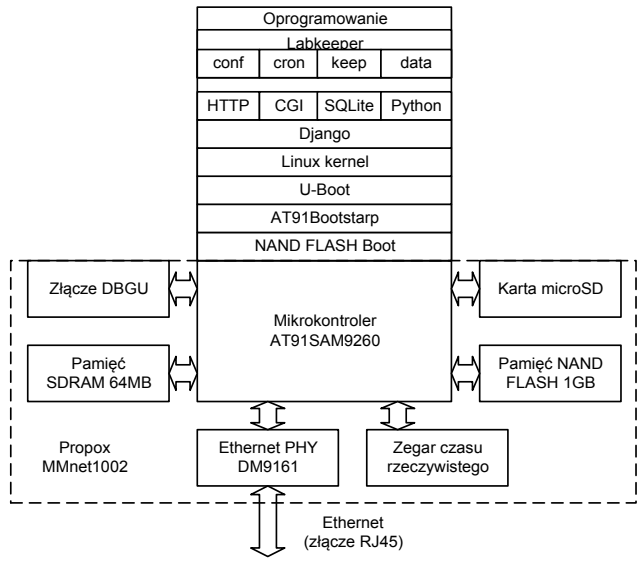
Z punktu widzenia wygody interfejsu użytkownika końcowego i łatwości tworzenia oprogramowania, najlepszym rozwiązaniem byłoby użycie mikrokomputera typu embedded PC. Mikrokomputer umożliwia wykorzystanie języków programowania wysokiego poziomu, w tym języków skryptowych takich jak PHP czy Python, znakomicie ułatwiających pisanie aplikacji z interfejsem webowym. Ponadto dostarcza bardzo funkcjonalną platformę sprzętową — w przypadku komputerów jednopłytkowych Mini-ITX, umożliwiającą podłączenie urządzeń peryferyjnych, takich jak dysk twardy, monitor, klawiatura czy drukarka. Poważnymi wadami komputerów jednopłytkowych są: cena i zapotrzebowanie na energię. Mikrokomputer WAFER-945GSE w standardzie 3,5" wyposażony w procesor Intel Atom N270 1,60 GHz, zintegrowaną kartę graficzną, złącze kart pamięci CF, złącza Gigabit Ethernet, USB, SATA, 1 GB pamięci RAM DDR2, pobiera 2,94 A przy zasilaniu 5 V, co daje blisko 15 W mocy [5], a w połączeniu z wysoką ceną około 300 USD stanowi czynnik dyskwalifikujący.

Biorąc pod uwagę przedstawiony przegląd rozwiązań i postawione założenia, wybrano mikrokontroler 32-bitowy ARM9 z dedykowanym systemem GNU/Linux. Zrezygnowano z mikrokontrolerów 8-bitowych ze względu na trudniejszą realizację zadań programistycznych i mniejsze możliwości dalszego rozwoju systemu. Biorąc pod uwagę wygodę rozbudowy oprogramowania przez użytkownika, odrzucono systemy RTOS na rzecz jądra Linux z oprogramowaniem GNU. Mikrokomputer jednopłytkowy uznano za opcję zbyt drogą, której możliwości zostałyby wykorzystane tylko w niewielkim stopniu.

3. Architektura sprzętowa i programowa proponowanego kontrolera

Uproszczony schemat blokowy proponowanego kontrolera pokazano na rys. 1. Dla uproszczenia prototyp sterownika oparto na module MMnet 1002 firmy Propox [6], zawierającym wszystkie niezbędne podzespoły sprzętowe m.in.: mikrokontroler AT91SAM9260, 64 MB pamięci SDRAM, 1 GB pamięci NAND Flash, interfejs Ethernet PHY, zegar RTC z podtrzymaniem.

Dzięki wykorzystaniu gotowej platformy sprzętowej możliwe było skupienie się na warstwie programowej. Po włączeniu zasilania kontrolę nad mikrokontrolerem sprawuje umieszczony w pamięci ROM tzw. Boot Program, który inicjalizuje podstawowe interfejsy oraz uruchamia program NAND Flash Boot, poszukujący aplikacji rozruchowej w pamięci NAND Flash. Podczas normalnej pracy w pamięci Flash znajduje się bootloader pierwszego stopnia AT91Bootstrap udostępniony przez Atmel. Jego podstawowym zadaniem jest konfiguracja generatora sygnału zegarowego, inicjalizacja GPIO oraz skonfigurowanie kontrolera pamięci SDRAM i załadowanie do niej bootloadera drugiego stopnia, którym na ogół jest U-Boot: otwarty, multiplatformowy bootloader dla systemów wbudowanych. Najważniejszą funkcją U-Boota jest wczytanie jądra systemu i uruchomienie go z odpowiednimi parametrami rozruchowymi. U-Boot'a wykorzystano nie tylko do uruchamiania jądra, ale także do uaktualniania poszczególnych bootloaderów i jądra systemu, co pozwala podczas rozwoju aplikacji na wprowadzanie poprawek do oprogramowania przy pomocy połączenia LAN i serwera TFTP.



Rys. 1. Uproszczona architektura sprzętowo-programowa energooszczędnego kontrolera systemu pomiarowego
Fig. 1. Simplified hardware-software architecture of the low-power controller of the measurement system

System operacyjny Linux został zbudowany z użyciem narzędzia Buildroot. Jako bibliotekę standardową języka C wykorzystano uClibc, stworzoną specjalnie z myślą o systemach wbudowanych i znacznie mniejszą od popularnej GNU C Library (znanej też jako glibc). Wspiera ona zarówno tradycyjne jądra Linux, jak i jego wersje bez MMU (Memory Management Unit), obsługuje wiele architektur, np. AMD64, ARM, Blackfin, MIPS/MIPSEL, PowerPC czy SPARC, udostępnia biblioteki współdzielone i watki, a jej twórcy zapewniają, iż da się z jej pomocą uruchomić zdecydowana większość aplikacji pisanych dla GNU C Library.

Spośród framework'ów do budowy aplikacji webowych zdecydowano się na Django, biorąc pod uwagę bogatą dokumentację, dużą liczbę gotowych funkcji, a przede wszystkim wykorzystanie języka Python. Ze względu na oszczędność pamięci, jako serwer bazy danych wykorzystano system SQLite, implementowany jako biblioteka języka C i przechowujący dane w pojedynczym pliku.

W prezentowanym rozwiązaniu użyto serwera lighttpd, który wykorzystuje zaledwie 3% pamięci operacyjnej. W zamian otrzymuje się możliwość nasłuchu pod wieloma adresami IP, zwiększenie bezpieczeństwa i wydajności serwowania plików statycznych. Serwer HTTP obsługuje samodzielnie pliki statyczne - dla pozostałych dane serwowane są przez Django za pomocą interfejsu FastCGI.

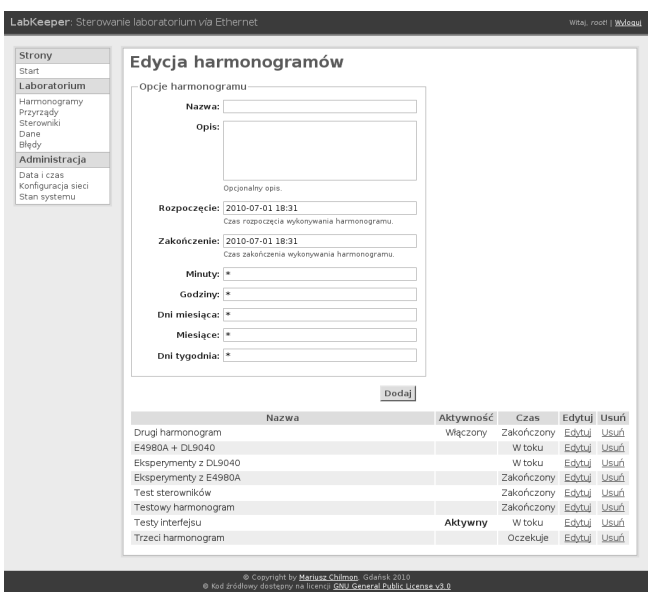
Aplikacja sterująca Labkeeper składa się z kilku modułów zrealizowanych z wykorzystaniem Django:
- **labkeeper** - moduł główny, zawierający pliki konfiguracyjne i administracyjne;
- **configurator** – pozwala na ustawienie parametrów zegara, sieci oraz udostępnia informacje o systemie i zasobach,
- **data** – wyświetla dane z akwizycji,
- **driver** - zawiera sterowniki urządzeń;
- **logger** – obsługuje komunikaty (np. błędów),
- **cron** - wywoływany cyklicznie co minutę sprawdza, które z aktywnych harmonogramów powinny być wykonane w danej chwili i tworzy wątki dla umieszczonych w nich zadań;

- **scheduler** - pozwala użytkownikowi zarządzać harmonogramami, urządzeniami i sterownikami
- **media i templates** – zawierają arkusze styli i szablony stron www.

Sterowniki urządzeń mają formę modułów języka Python, dzięki czemu nie wymagają kompilacji, są czytelne i dają użytkownikowi dostęp do bogatych zasobów funkcji standardowych tego języka. Implementacja prostych protokołów, opartych na przesyłaniu ciągów ASCII (np. SCPI) jest łatwa i intuicyjna. Opracowano dwa przykładowe sterowniki urządzeń SCPI: miernika Agilent E4980A i oscyloskopu Yokogawa DL9040, bazujące na metodzie *send_scpi*, przyjmującej jako argument i wysyłającej do przyrządu komendy SCPI.

W aplikacji Labkeeper harmonogramem nazywany jest zestaw komend, czyli zadań akwizycyjnych, które wysyłane są do określonych przyrządów we wskazanych momentach czasu. Cykliczne wykonywanie harmonogramów jest realizowane z użyciem demona cron. Wywołuje on co minutę główną funkcję cron() modułu cron/views.py. Funkcja ta tworzy listę wszystkich włączonych harmonogramów, dla których aktualny czas zawiera się miedzy czasem rozpoczęcia a zakończenia, i sprawdza je kolejno pod katem reguł czasu wykonania. Jeżeli dany zestaw komend ma być wykonany w chwili wywołania funkcji cron(), dla każdej z komend wywoływany jest watek DAQ i rozpoczyna się akwizycja danych. Dzięki pracy wątkowej możliwe jest równoległe wysyłanie zapytań do wielu przyrządów. Nie należy jednak dopuszczać do sytuacji, gdy jeden przyrząd otrzyma kilka różnych zapytań jednocześnie, ponieważ mogło by to skutkować wprowadzeniem go w nieprawidłowy stan lub błędna interpretacją odebranych przez aplikację danych — w celu zachowania prostoty, sterowniki pisane są z założeniem, że każdy jednorazowy proces akwizycji danych jest samodzielnym zadaniem. Zastosowano zatem kontrolę dostępu do przyrządów z użyciem krotki obiektów, zwracanych przez funkcje Lock() pakietu *threading*. Jest to odpowiednik tablicy muteksów, z których w tym przypadku każdy odpowiada za ograniczenie dostępu do przyrządów — w danej chwili tylko jeden z wątków może uzyskać dostęp do określonego przyrządu.

Wygląd strony www z zakładką Harmonogramy pozwalającą na zarządzanie harmonogramami akwizycji pokazano na rys. 2.



Rys. 2. Konfiguracja harmonogramów akwizycji aplikacji Labkeeper
Fig. 2. Acquisition schedules configuration in Labkeeper application

Stan wszystkich harmonogramów można odczytać w tabeli u dołu zakładki Harmonogramy (rys. 2). Kolumna „Czas" określa, czy harmonogram może być wykonywany („W toku"), czy oczekuje na rozpoczęcie („Oczekuje"), czy też został już zakończony („Zakończony"). W kolumnie „Aktywność" jest wyświetlany status „Włączony", jeżeli harmonogram jest włączony, ale jeszcze oczekuje na rozpoczęcie lub został już zakończony. Status „Aktywny" oznacza, ze dany harmonogram jest włączony i może być wykonywany — tylko tak oznaczone harmonogramy realizują swoje zadania, tj. wysyłają polecenia do urządzeń.

4. Podsumowanie

Zrealizowano założenia projektowe dla energooszczędnego, autonomicznego kontrolera systemu pomiarowego na bazie sieci Ethernet. Wykorzystano moduł MMnet1002, a oryginalny system operacyjny zastąpiono dedykowaną dystrybucją GNU/Linux zoptymalizowaną pod kątem minimalizacji zużycia zasobów. Efektem jest w pełni funkcjonalny sterownik z przejrzystym interfejsem użytkownika w postaci strony WWW. Reguły wykonywania dla harmonogramów są bardzo elastyczne i umożliwiają tworzenie złożonych zadań dla akwizycji.

Urządzenie testowano z dwoma przyrządami laboratoryjnymi: miernikiem Agilent LCR E4980A i oscyloskopem Yokogawa DL9040, dla których opracowano sterowniki. Eksperymenty wykazały dwa podstawowe problemy, jakie może napotkać użytkownik podczas wykorzystania sterownika: braki w dokumentacji protokołów przyrządów i nieprawidłowa komunikacja w przypadku szybko następujących po sobie połączeń TCP. Przyjęte rozwiązania programistyczne - sterowniki w postaci skryptów języka Python - umożliwiają prostą implementację nawet złożonych protokołów oraz zapewniają wysoką elastyczność i łatwość modyfikacji.

Pobór prądu przez kontroler przy zasilaniu 9 V jest na poziomie 150 mA, co oznacza pobór mocy poniżej 1,5 W. Sterownik jest więc rozwiązaniem bardzo energooszczędnym w stosunku do komputera PC.

Wadą urządzenia jest jego powolna praca: start systemu zajmuje 35 s, z czego ponad 15 s to uruchamianie aplikacji Labkeeper. Jednak za poważniejszy mankament należy uznać czasochłonne zapytania do bazy danych, które zauważalnie spowalniają interfejs użytkownika podczas przeglądania dużej ilości danych. Problem ten został zminimalizowany dzięki optymalizacji zapytań.

Warto zaznaczyć, że sterownik został zrealizowany wyłącznie z użyciem oprogramowania FLOSS (Free Libre/Open Source Software) na dystrybucjach systemu GNU/Linux. Wykazano w ten sposób praktyczną przydatność Wolnego i Otwartego Oprogramowania w realizacji systemów pomiarowych.

5. Literatura

[1] Embedded-System.Net: PIC18F97J60 Ethernet Microcontroller with MAC & PHY and Free TCP/IP Stack. http://embedded-system.net/pic18f97j60.html
[2] Dunkels A., Grönvall B., Voigt T.: Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors, I IEEE Workshop on Embedded Networked Sens., 2004, Tampa, USA
[3] egnite Software GmbH: NutApi. http://www.ethernut.de/api/index.html, 2007.
[4] Atmel Corporation: AT91SAM9260 Datasheet, 2009.
[5] IEI Technology Corp: WAFER-945GSE User Manual, 2009.
[6] Propox: MMnet1000 Podręcznik użytkownika. System Linux, 2009.

otrzymano / received: 15.07.2014
przyjęto do druku / accepted: 01.10.2014 *artykuł recenzowany / revised paper*