Marek KOŚMIDER, Mateusz CHMIELOWIEC, Ryszard KOWALIK, Emil BARTOSIEWICZ

Politechnika Warszawska, Instytut Elektroenergetyki

doi:10.15199/48.2016.07.02

Dydaktyczne zastosowanie modułu łącza Ethernet/IP w systemie mikroprocesorowym DSM-51

Streszczenie. *Artykuł opisuje przykładowe zastosowanie modułu łącza Ethernet/IP typu IIM7010A do współpracy z Dydaktycznym Systemem Mikroprocesorowym DSM-51, w sposób umożliwiający wykorzystanie tego modułu podczas zajęć laboratoryjnych.*

Abstract. *The article presents the example solution of cooperation between the IIM7010A Ethernet/IP module and Microprocessor Educational System DSM-51 designed to exploit this module during laboratory classes (**Didactic use of Ethernet/IP module for DSM-51 Microprocessor Educational System**).*

Słowa kluczowe: DSM-51, Ethernet/IP, IIM7010A, dydaktyka.
Keywords: DSM-51, Ethernet/IP, IIM7010A, education.

Wstęp

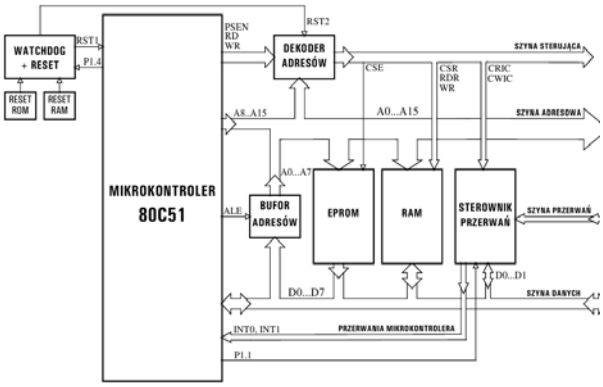
Dydaktyczny System Mikroprocesorowy DSM-51, opracowany przez firmę MicroMade, jest uniwersalnym sterownikiem wyposażonym w mikrokontroler 8051, rozbudowanym o szereg elementów zewnętrznych. Urządzenie to jest stosowane z powodzeniem, od wielu lat, w technikach o profilu elektrycznym i w wyższych uczeniach technicznych, jako pomoc dydaktyczna przy poznawaniu procesorów rodziny 51', asemblera tych procesorów oraz podstaw języka C. Również w Instytucie Elektroenergetyki Politechniki Warszawskiej prowadzone są ćwiczenia laboratoryjne z wykorzystaniem przedstawionego sterownika. Celowe staje się zatem ciągłe udoskonalanie i poszerzanie jego możliwości.

Na przestrzeni ostatnich kilku lat można zauważyć gwałtownie rosnący udział urządzeń mikroprocesorowych wykorzystujących standard Ethernet. Rynek urządzeń sieciowych w tym czasie bardzo się rozwinął, przez co wzrosła dostępność i spadły ceny urządzeń i kontrolerów umożliwiających przesyłanie danych w standardzie Ethernet. Co więcej, w zasadzie każdy aktualnie dostępny na rynku komputer osobisty typu PC czy przenośny (laptop, netbook) jest wyposażony w kartę sieciową tego standardu. Upowszechnienie, jak i relatywnie niska cena urządzeń sieciowych w stosunku do możliwości spowodowały, że coraz więcej przedmiotów codziennego użytku (np. RTV czy AGD) posiada możliwość podłączenia do sieci Ethernet.

Dydaktyczny System Mikroprocesorowy DSM-51

Dydaktyczny System Mikroprocesorowy DSM-51 został zaprojektowany w 1992 roku przez firmę MicroMade. Później przez lata był modernizowany aż do powstania (w 2004 roku) trzeciej wersji systemu, zgodnej z zasadniczymi wymaganiami dyrektywy Unii Europejskiej nr 89/336/EEC, dotyczącej kompatybilności elektromagnetycznej. W niniejszym opracowaniu wykorzystywano zestaw w wersji drugiej, który (w przeciwieństwie do wersji najnowszej) posiada złącze magistrali systemowej.

Schemat blokowy części mikroprocesorowej systemu DSM-51 został przedstawiony na rysunku 1. Na schemacie możemy wyszczególnić takie układy, jak: mikrokontroler, pamięci (RAM i EPROM), bufor i dekoder adresów, sterownik przerwań i układ watchdog. Główne magistrale komunikacyjne pomiędzy tymi układami tworzą: szyna adresowa, szyna danych i szyna sterująca. Mikrokontroler 80C51 posiada multipleksowaną szynę danych z szyną adresową, tzn. młodsza część adresu (A0..A7) jest wystawiana na szynę danych i wpisywana do rejestru BUFOR ADRESÓW (fizycznie układ HC573) sygnałem ALE. Bufor „przetrzymuje" na swoim wyjściu zatrzaśniętą dolną część adresu (bity od A0 do A7), aż do otrzymania następnego sygnału ALE. Wówczas następuje ponowne przepisane stanu jego wejść na wyjścia. W połączeniu ze starszą częścią adresu (A8..A15 - bezpośrednio z mikrokontrolera), zostaje uzyskana pełna 16-bitowa szyna adresowa. Dzięki temu szyna danych (D0..D7) może pełnić swoją funkcję docelową - tj. przesyłanie danych pomiędzy układami. O tym, który z układów zostanie uaktywniony po podaniu pełnego adresu, decyduje DEKODER ADRESÓW - w tym wypadku programowany układ logiczny PLD typu GAL16V8, wraz z demultiplekserem typu 74HC139.



Rys.1. Schemat blokowy systemu mikroprocesorowego DSM-51 w wersji 2, wg [1]

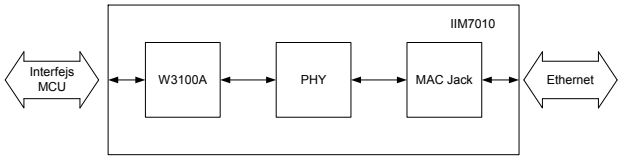
Mikrokontroler 8051 wykonany jest w mieszanej architekturze harwardzkiej, tzn. zachowane jest rozdzielenie pamięci programu i pamięci danych, a do komunikacji wykorzystane są te same szyny adresowa i danych. Podczas normalnej pracy systemu mikroprocesorowego, mikrokontroler odczytuje kolejne rozkazy z pamięci EPROM, natomiast pamięć RAM wykorzystywana jest tylko jako pamięć operacyjna. Również w systemie DSM-51 dostępny jest ten schemat działania, tzn. w przestrzeni adresowej pamięci programu dostępny jest układ EPROM o pojemności 32kB, a w przestrzeni adresowej pamięci danych układ RAM o takiej samej pojemności. Ze względu na fakt, że w systemie DSM-51 zachodzi potrzeba uruchamiania programów napisanych przez użytkowników w sposób szybki i prosty, dlatego do celów testowych pamięć RAM zostaje umieszczona w przestrzeni adresowej pamięci programu. Wyborem pamięci, z której odczytywany jest program, także zajmuje się dekoder adresów. W trybie

0 dekodera program odczytywany jest z pamięci EPROM, natomiast w trybie 1 z pamięci RAM. Dekoder adresów zajmuje się także uaktywnianiem pozostałych układów podłączonych do magistrali systemowej. Na wejście otrzymuje on sygnały PSEN, RD, WR oraz wybrane linie adresowe (A6, A7, A12..A15), a na wyjściu generuje sygnały CSE, CSR, RDR, WR oraz inne do pozostałych układów.

Moduł Ethernet/IP typu IIM7010A

Moduł IIM7010A został zaprojektowany przez firmę WIZNET [4] - wychodząc naprzeciw oczekiwaniom projektantów, którzy potrzebowali jak najprostszego urządzenia, umożliwiającego w łatwy sposób przystosowanie określonej aplikacji do współpracy z siecią Ethernet. Moduł jest zaprojektowany i wykonany w taki sposób, aby można go było bez większego problemu używać we własnych aplikacjach. Jest on wygodny w użyciu ze względu na bardzo zwartą i przemyślaną konstrukcję. Został wykorzystany druk dwustronny z powierzchniowym montażem elementów SMD. Moduł podłączany jest do układu mikroprocesorowego poprzez dwa złącza o wymiarach rastru 2 mm. W wyprowadzeniach tych uwzględniono sygnały 15-bitowej szyny adresowej, 8-bitowej szyny danych, a także sygnały sterujące pracą mikrokontrolera oraz sygnały informujące o statusie połączenia internetowego, poprzez dołączenie np. diod LED. Dodatkowo, zostały wyprowadzone sygnały linii interfejsu I^2C, dzięki któremu można połączyć moduł internetowy z mikrokontrolerem bez potrzeby wykorzystania portów P0 i P2. Sam moduł jest zasilany napięciem 3,3 V, ale jego wejścia/wyjścia cyfrowe są przystosowane do współpracy ze standardem TTL 5 V. Moduł IIM7010A samoczynnie wykrywa sieć Ethernet 10/100 Mb/s.

Schemat blokowy układu IIM7010A, przedstawiający sposób realizacji interfejsu pomiędzy siecią Ethernet a systemem mikroprocesorowym, został przedstawiony na rysunku 2.



Rys.2. Schemat blokowy modułu IIM7010A, wg [2]

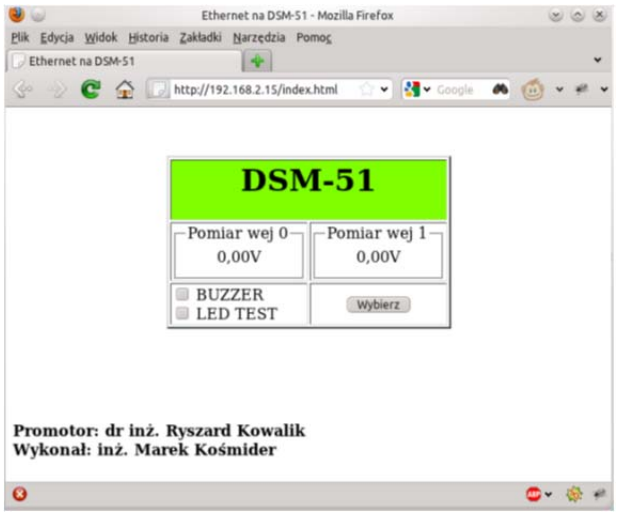
Moduł IIM7010A składa się z gniazda RJ-45 (bezpośrednio łączącego się z siecią Ethernet), układu PHY (kontrolera sieci w postaci RTL8201BL) oraz głównego układu W3100A. Układ ten odpowiedzialny jest za sprzętową realizację stosu TCP/IP. Zawiera on protokoły do obsługi sieci, w tym TCP/IP, UDP, ICMP, ARP, DLC, MAC, a także protokoły warstwy aplikacji, takie jak DHCP, HTTP, SMTP, PING.

Do połączenia z siecią Ethernet możliwe jest wykorzystanie czterech kanałów. Układ umożliwia wymianę danych w trybie full-duplex. Został wykonany w technologii CMOS 0,35 µm LSI i ma obudowę SMD. Dzięki użyciu układu W3100A, w zależności od aplikacji sterującej, można uzyskać prędkości transmisji w zakresie od 300 kb/s do 6 Mb/s.

Wraz z układem został przez producenta przygotowany zestaw procedur źródłowych interfejsu API, dzięki którym możliwe jest sterowanie układem W3100A. Sterowanie to polega na zapisywaniu danych pod określony adres układu W3100A, w wyniku czego układ realizuje ustalone zadania związane ze sprzętową implementacją stosu TCP/IP. Dane zapisane do bufora nadawczego są uzupełniane o nagłówek odpowiedniego protokołu i ostatecznie przekształcane na ramkę Ethernet, która (poprzez układ RTL8201BL) zostaje wysłana do sieci Ethernet. Adresy poszczególnych rejestrów oraz funkcje sterujące pracą układu W3100A zostały zapisane w dostarczonych przez producenta listingach plików źródłowych (pliki: SOCKET.C, SOCKET.H, TYPE.H). Dodatkowo, wraz z procedurami API, opracowano przykładowe programy umożliwiające, przy wykorzystaniu procesora 80C51 wraz z układem W3100A, uruchomienie aplikacji prostego serwera WWW, serwera TCP, klienta TCP, klienta DHCP czy klienta UDP.

Opis programu serwera WWW

Oprogramowanie, wykorzystujące możliwości modułu IIM7010A, zostało napisane w języku C i skompilowane na komputerze z zainstalowanym systemem Windows XP i kompilatorem Keil C51. Program ma funkcjonalność serwera WWW, który po nawiązaniu połączenia przez przeglądarkę internetową zwraca kod HTML, reprezentujący aktualne pomiary z kanału 1 i 2 przetwornika analogowo-cyfrowego ADC oraz pozwalający na załączenie diody LED (TEST) i brzęczyka (BUZZER) systemu DSM-51. Wygląd strony WWW, generowanej przez niniejszy program, przedstawiono na rysunku 3.

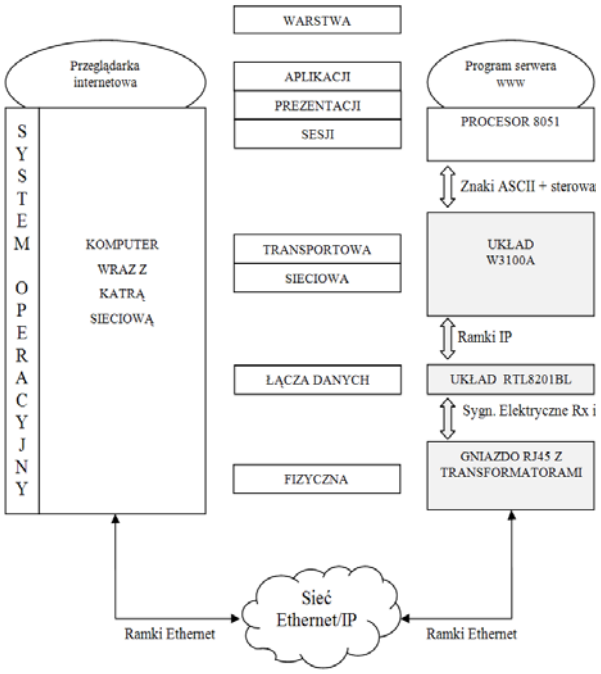


Rys.3. Wygląd strony WWW generowanej przez program

Wymiana danych pomiędzy przeglądarką internetową a serwerem WWW jest realizowana poprzez łącze Ethernet, z wykorzystaniem protokołów IP, TCP, HTTP. Na rysunku 4. pokazano tego rodzaju wymianę danych, w formie schematu blokowego zawierającego warstwę sprzętową oraz czynności wykonywane przez odpowiednie podprogramy.

Zgodnie z rysunkiem 4., wymiana danych w warstwie Ethernet/IP przebiega poprzez wymianę pakietów danych. W warstwie transportowej wykorzystywany jest protokół TCP. Wymiana danych w tym protokole przebiega w kilku krokach. Pierwszym z nich jest ustanowienie połączenia logicznego. Rozpatrując połączenie TCP z punktu widzenia funkcjonalności, można je potraktować jako wirtualne połączenie (kanał) pomiędzy warstwami TCP maszyny serwera i klienta. Charakterystyczne dla każdego kanału są numer portu serwera i klienta, pomiędzy którymi następuje wymiana danych. Kombinacja tych numerów portów, wraz z adresami sieciowymi, określa parę wirtualnych gniazd (ang. *socket*) tworzących połączenie.

Na poziomie warstwy aplikacji, wymiana danych jest wymianą znaków ASCII. Przebiega ona w następujący sposób: przeglądarka internetowa, po wpisaniu przykładowego adresu http://192.168.2.15/index.html, wysyła do serwera żądanie (ang. *request*) w postaci ciągu znaków przedstawionych w listingu 1. Warto zaznaczyć, iż zawartość zapytania różni się w zależności od systemu operacyjnego, zainstalowanej przeglądarki i jej ustawień, oraz adresu IP komputera, na którym przeglądarka jest uruchamiana.



Rys.4. Schemat blokowy, przedstawiający zasadę funkcjonowania serwera WWW, działającego na systemie DSM-51 z wykorzystaniem modułu IIM7010A (szare elementy)

Listing 1. Nagłówek zapytania

```
GET /index.html HTTP/1.1
Host: 192.168.2.15
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1)
Gecko/20100101 Firefox/4.0.1
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: pl
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

Serwer, po otrzymaniu takiego ciągu znaków, analizuje jego treść, a następnie podejmuje odpowiednią decyzję - w zależności od wyniku analizy. Po wykonaniu tych działań, serwer generuje odpowiedź (ang. *response*) - także w postaci ciągu znaków ASCII. Nagłówek odpowiedzi zamieszczony został na listingu 2.

Listing 2. Nagłówek odpowiedzi serwera

```
HTTP/1.1 200 OK
Server: Apache/1.3.9 (Unix)
```

Pierwsza linia listingu 2. zawiera kod odpowiedzi HTTP, który w tym wypadku oznacza zaakceptowanie zapytania i zwrócenie zawartość żądanego dokumentu w dalszej części odpowiedzi. Druga linia nagłówka informuje przeglądarkę o aplikacji serwera. Oczywiście w systemie DSM-51 nie jest zaimplementowany serwer Apache, ale jest on najbardziej rozpowszechnioną aplikacją serwera WWW i stąd taki wybór nazwy dla serwera w DSM-51. Dalsza część odpowiedzi to docelowa zawartość strony, zapisana w języku znaczników HTML, która zostaje wyświetlana w oknie przeglądarki. Jak wszystkie poprzednio przesyłane informacje, tak i ta jest przesyłana w postaci ciągu znaków. Zawartość strony, zapisana w języku HTML, została przedstawiona na listingu 3.

Listing 3. Odpowiedź serwera – zawartość strony

```
<html><head><title>Ethernet na DSM-51</title>
<meta http-equiv="Content-Type"
content="Text/html;charset=Windows-1250">
<meta http-equiv="refresh" content="5"></head>
<body bgcolor="white"><div align="center"><p><br></p>
<table border="2" width="300"><tr align="center">
<th colspan="2" bgcolor="chartreuse"><h1>DSM-
51</h1></th></tr>
<tr align="center" height="42" bgcolor="white">
<td width="50%"><fieldset><legend>Pomiar wej
0</legend><div>0,00V
</div></fieldset></form></td><td>
<fieldset><legend>Pomiar wej 1</legend><div>0,00V
</div></fieldset></form></td></tr><tr height="42">
<form name="form1" method="get" action = "LCD.CGI">
<td><input type="checkbox" name="led1" value="on"> BUZZER
<br>
<input type="checkbox" name="led2" value="on"> LED TEST</td>
<td align="center"><input type="submit" name="Submit"
value="Wybierz">
</td></form></tr></table></div><br><br><br><br>
<div align="left"><p><h4>Promotor: dr inż. Ryszard Kowalik
<br>Wykonał: inż. Marek Kośmider</h4>
</p></div></body></html>
```

Otrzymując taki kod, przeglądarka internetowa odpowiednio go interpretuje i wyświetla w postaci widocznej na rysunku 3. Strona odświeżana jest automatycznie co 5 sekund. Jej zawartość jest stała - oprócz dwóch liczb, informujących o napięciu podawanym na kanały nr 0 i 1 przetwornika analogowo-cyfrowego ADC. Wartości te generowane są dynamicznie przez program, na podstawie rzeczywistych wartości napięcia mierzonego na zaciskach wejścia przetwornika analogowo-cyfrowego systemu DSM-51.

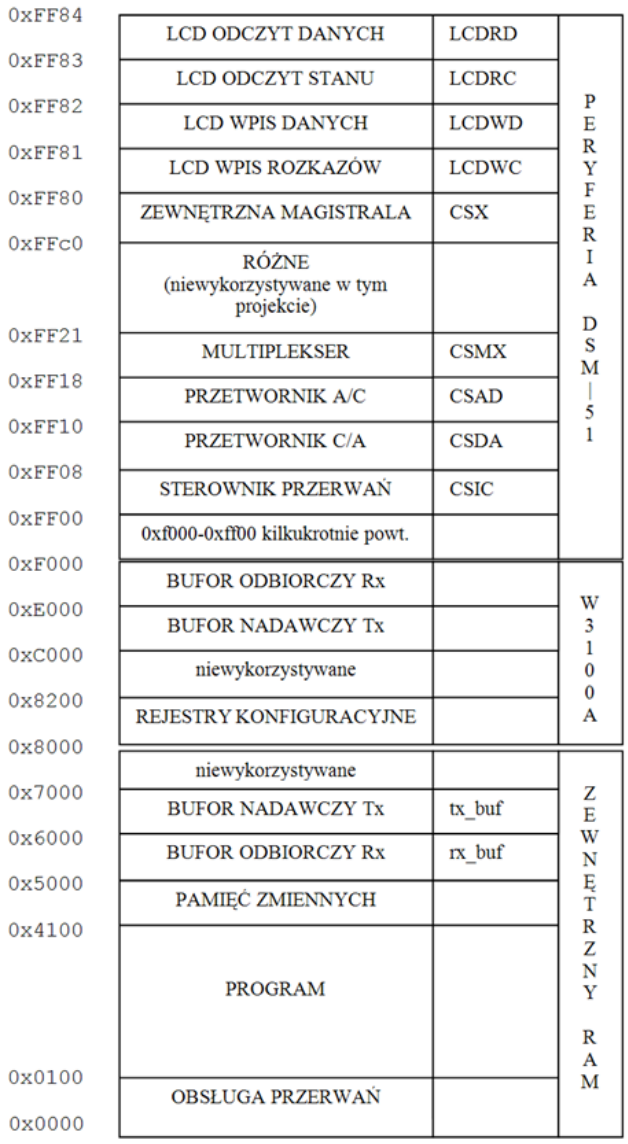
Opis źródeł programu

Aby uzyskać funkcjonalność, polegającą na poprawnej wymianie bloków danych w protokole HTTP, należało wykonać urządzenie zawierające elementy potrzebne do poprawnego podłączenia modułu IIM7010A do systemu DSM-51 (patrz rozdz. *Część sprzętowa stanowiska*) oraz napisać i wgrać do tego systemu oprogramowanie, mające funkcjonalność serwera WWW.

Wszystkie funkcje, które są wykorzystywane przez procesor 80C51 i polegające na konfiguracji układu W3100A oraz wymiany danych z tym układem i obsługą zapytań programu klienta WWW, zostały napisane w języku C i zaimplementowane w systemie DSM-51. W skład programu wchodzą zarówno procedury z (dostarczonej przez producenta układu W3100A) dedykowanej biblioteki, jak również funkcje napisane przez projektanta opisywanego układu.

Na rysunku 5. została przedstawiona mapa zewnętrznej pamięci danych procesora 80C51 systemu DSM-51 z podłączonym modułem zawierającym IIM7010A. Należy zwrócić szczególną uwagę na ostatni jej fragment (obszar 0xF000 – 0xFFFF), w którym umieszczono urządzenia wejść/wyjść systemu DSM-51. Z powodu niepełnego rozróżniania adresów przez dekoder adresów (brak sygnałów bitów adresowych A8..A15), obszar ten podzielony jest na szesnaście bloków jednakowych zestawów sygnałów wyboru urządzeń peryferyjnych. Dla

mikrokontrolera 80C51 pod adresami 0xF000 – 0xF0FF są widoczne dokładnie te same urządzenia co pod adresami 0xF100 – 0xF1FF, 0xF200 – 0xF2FF i tak dalej aż do 0xFF00 – 0xFFFF.



Rys.5. Mapa zewnętrznej pamięci (XRAM) procesora 80C51

W programie dla urządzeń I/O stosowane są adresy z zakresu 0xFF00 – 0xFFFF. W przestrzeni adresów od 0x0000 do 0x40FF dostępna jest pamięć programu, gdyż w systemie DSM-51 pamięć programu użytkownika również znajduje się w zewnętrznej pamięci RAM - w celu szybkiego wgrywania i testowania programów. W przestrzeni od adresu 0x4100 do 0x7FFF znajduje się w dalszym ciągu zewnętrzna pamięć RAM, pracująca tu jako pamięć zmiennych oraz buforów nadawczych i odbiorczych. Pod adresami z zakresu od 0x8000 do 0xEFFF widoczny jest układ W3100A modułu IIM7010A, zawierający rejestry konfiguracyjne oraz bufory nadawczy i odbiorczy dla każdego gniazda. Dla adresów z tego przedziału wytwarzany jest sygnał CSX, doprowadzony do złącza magistrali systemowej systemu DSM-51 - co umożliwia wybranie modułu IIM7010A.

Źródła programu składają się z kilku plików **.c* i **.h*, które są najpierw kompilowane przez pakiet Keil C51 do plików **.obj*, a następnie likowane do jednego pliku, z którego generowany jest kod maszynowy mikroprocesora 8051 w formie pliku **.hex*. Zestawienie plików źródłowych zawarte jest w tabeli 1.

Tabela 1. Zestawienie plików źródłowych, tworzących omawiane oprogramowanie

Nazwa pliku	Opis
DSM51.h	Zawiera definicje nazw urządzeń zewnętrznym systemu DSM-51, przypisujące im odpowiednie adresy z przestrzeni adresowej XRAM (np. CSAD, CSMX).
httpd.c	Zawiera funkcję obsługi przerwania od Timera 0, funkcję dekodującą zapytania przeglądarki i procedury tworzące ciąg znaków ASCII, będący zawartością strony WWW, a także podprogram wpisujący ten ciąg do bufora zewnętrznego modułu IIM7010A.
httpd.h	Zawiera deklaracje funkcji zdefiniowanych w pliku httpd.c.
lcd.c	Zawiera funkcje obsługi wyświetlacza LCD, w tym procedurę ładującą tablicę polskich znaków do pamięci wyświetlacza.
main.c	Zawiera główną funkcję *main()* programu, wraz z pętlą nieskończoną i definicjami funkcji inicjalizujących system DSM-51 i układ IIM7010A.
socket.c	Zawiera funkcje obsługujące gniazda, funkcje odczytu i zapisu danych do i z buforów oraz procedurę obsługi przerwania od układu W3100A i konfigurującą ten układ.
socket.h	Zawiera deklarację liczby gniazd, definicję rejestrów wewnętrznych układu W3100A oraz kodów statusów operacji i komend, a także deklaracje funkcji zdefiniowanych w pliku socket.c.
sockutil.c	Zawiera przydatne funkcje związane z obsługą układu W3100A, w opisywanym programie w większości niewykorzystywane.
sockutil.h	Zawiera deklaracje funkcji zdefiniowanych w pliku sockutil.c.
type.h	Zawiera definicję typów zmiennych tj. BYTE, INT, WORD.
util.c	Zawiera funkcje użytkowe, głównie do obsługi łańcuchów znaków (ang. *string*) tj. przeszukiwanie ciągów, zmiana wielkości liter, zamiana liczb na postać binarną i odwrotnie.
util.h	Zawiera deklaracje funkcji zdefiniowanych w pliku util.c.

Opis działania programu głównego

Na listingu 4. zostały przedstawione ważniejsze elementy programu głównego. Na początku programu znajdują się dyrektywy preprocesora, dotyczące dołączania plików nagłówkowych i definiowania stałych. Wywoływana jest także procedura inicjalizująca wartości rejestrów procesora 8051 – *Init8051()*, funkcja inicjalizująca układ W3100A – *initW3100A()* oraz funkcja *InitNetConfig()* - ustawiająca odpowiednie rejestry wewnętrzne wspomnianego układu tak, aby podczas wymiany danych w protokole Ethernet/IP używał on właściwych wartości: adresu IP, maski podsieci czy adresu MAC. Po tych czynnościach następuje inicjalizacja czterech gniazd (ang. *socket*), o numerach od 0 do 3.

Pod pojęciem gniazda należy rozumieć zestaw danych oraz czynności, które zapewniają poprawną wymianę danych między jednym klientem i serwerem. Można stwierdzić, że gniazdo pełni rolę interfejsu serwera „na zewnątrz”. Oznacza to, że przeglądarka, po wysłaniu żądania do serwera, wymienia dane z wykorzystaniem pierwszego wolnego gniazda.

W buforach danych nadawczych i odbiorczych, skojarzonych z tym gniazdem, pojawiają się ciągi znaków ASCII, wymieniane przez program serwera i klienta. W związku z powyższym, po przesłaniu danych z programu

klienta do serwera, serwer odczytuje dane z bufora odczytu, a następnie je analizuje i zwraca odpowiednie dane do bufora nadawczego tego samego gniazda - gdyż odpowiedź serwera również wysyłana jest do przeglądarki z wykorzystaniem tego samego gniazda.

Listing 4. Ważniejsze elementy programu głównego

```
#include <reg51.h>
#include "socket.h"
#include "httpd.h"
#include "sockutil.h"
#include "DSM51.h"
#include "lcd.c"
#define   MAX_BUF_SIZE    2048
[…]
u_char xdata * rx_buf = 0x5000;
u_char xdata * tx_buf = 0x6000;
[…]
void main()
{
    […]
    Init8051();
    initW3100A();
    InitNetConfig();
    for (i = 0; i < MAX_SOCK_NUM; i++) init_sock(i);
    while (1)
    {       […]
            for (i = 0; i < MAX_SOCK_NUM; i++)
            {
            state = select(i, SEL_CONTROL);

            switch(state)
                    {
                    case SOCK_ESTABLISHED :
                    […]
                    case SOCK_CLOSE_WAIT :
                    […]
                    case SOCK_CLOSED :
                    […]
                    }
            }
    }
}
```

Gniazdo prowadzi więc transmisję dwukierunkową. Po ustawieniu 4 gniazd (wszystkich w stan „nasłuchiwania" na porcie 80 na pojawiające się znaki ASCII), następuje wejście programu w pętlę nieskończoną. Pętla ta sprawdza naprzemiennie stan każdego z gniazd i, w zależności od tego stanu, podejmuje odpowiednie działania. Jeżeli gniazdo nasłuchuje, to nie są podejmowane żadne działania. Jeżeli gniazdo wskazuje na przerwanie połączenia, to jest ono zamykane. Jeżeli gniazdo jest zamknięte, to następuje ponowna jego inicjalizacja - to jest włączenie nasłuchiwania na porcie 80. Natomiast jeżeli gniazdo wykazuje, że jest ustanowione połączenie z klientem (przeglądarką), to następuje odczyt żądania, jego analiza i zwrócenie odpowiedzi do przeglądarki (ewentualnie zamiana stanu brzęczyka BUZZER i/lub diody TEST). Po zakończeniu połączenia, dane gniazdo jest zamykane i inicjalizowane na nowo.

Warto dodać, że w programie wykorzystywane są również dodatkowe elementy, nie będące częścią składową modułu IIM7010A i nie związane z wymianą danych poprzez protokół Ethernet/IP – przetwornik ADC i wyświetlacz LCD.

Początek pliku *main.c* zawiera dyrektywy *#include* załączania plików nagłówkowych (plików z rozszerzeniem **.h*). W plikach tych zawarte są deklaracje funkcji, definicje stałych oraz makra, pogrupowane według funkcjonalności. W pliku *reg51.h* zdefiniowane są adresy rejestrów mikrokontrolera 80C51, w *socket.c* oraz *socket.h* znajdują się wpisy dotyczące obsługi układu W3100A (w tym również obsługi gniazd), natomiast plik *httpd.h* zawiera deklaracje funkcji związanych z funkcjonalnością serwera www. Plik nagłówkowy *DSM51.h* określa adresy urządzeń zewnętrznych systemu DSM-51, takich jak np. przetworniki analogowo-cyfrowy ADC i cyfrowo-analogowy DAC, sterownik przerwań, czy wyświetlacz LCD. Funkcje do obsługi wyświetlacza zadeklarowane są zaś w pliku *lcd.h.*

W następnym kroku definiowana jest stała *MAX_BUF_SIZE* - jako wartość 2048. Reprezentuje ona maksymalną liczbę danych, która może się znaleźć w buforze dla danych odbiorczych. Każde z czterech gniazd ma bowiem przydzielone 2048 B dla bufora odbiorczego i 2048 B dla bufora nadawczego. Wspomniane bufory umieszczone są w przestrzeni adresowej układu W3100A, natomiast w programie zdefiniowane są dodatkowo bufory nadawczy i odbiorczy – odpowiednio *rx_buf* i *tx_buf,* o rozmiarach 4096 B, umieszczone w pamięci RAM systemu DSM-51. Bufor *rx_buf* zaczyna się od adresu *0x5000*, a bufor *tx_buf* od adresu *0x6000,* co widoczne jest na rysunku 5.

Część sprzętowa stanowiska

W celu umożliwienia współpracy pomiędzy modułem IIM7010A a systemem DSM-51, została zaprojektowana, a następnie wykonana, płytka wyposażona zarówno w gniazdo do podłączenia magistrali systemowej układu DSM-51, jak i gniazdo modułu IMM7010A (rysunek 6.).

Rys.6. Zdjęcie układu łączącego (przejściówka) moduł IIM7010A z systemem DSM-51, wraz z włożonym modułem IIM7010A

Główne zadania „przejściówki" to:

- dopasowanie napięcia zasilania,
- sygnalizacja stanu połączenia diodami LED,
- generowanie sygnałów *reset* i */reset,*
- połączenie elektryczne pomiędzy modułem a systemem DSM-51.

Dopasowanie napięcia zasilania

System DSM-51 zasilany jest z zasilacza wtyczkowego, dostarczającego stabilizowanego napięcia 9 V. Napięcie to wykorzystywane jest do zasilania wyświetlaczy siedmiosegmentowych oraz obniżane jest do wartości 5 V poprzez stabilizator 7805 - a następnie służy do zasilania części cyfrowej sterownika. Napięcie 9 V stosowane jest także do zasilania części analogowej, lecz z punktu widzenia podłączenia modułu IIM7010A jest to nieistotne. Gniazdo magistrali systemowej systemu DSM-51, na wyprowadzeniach nr 32 i 34, posiada dostępne napięcie stabilizowane 5 V. Moduł IIM7010A wymaga natomiast zasilania napięciem 3,3 V. W związku z tym, płytka łącząca moduł ethernetowy z systemem DSM-51 została wyposażona w stabilizator L4931C33 produkcji

STMicroelectronics, o wydajności prądowej 300 mA - co jest wystarczające dla tego zastosowania. Układ L4931C33 pracuje w swojej standardowej aplikacji. Wejścia/wyjścia modułu przystosowane są do pracy zarówno przy napięciu 3,3 V, jak i w standardzie TTL 5 V. Dzięki tej właściwości zbędne jest stosowanie konwertera poziomów logicznych z 3,3 V na 5 V i możliwe staje się bezpośrednie połączenie magistrali adresowej i danych do układu IIM7010A, co znaczenie obniża komplikację i koszt całego układu.

Na płytce zamontowane są również kondensatory przeciwzakłóceniowe 100 nF, filtrujące napięcie zasilania, oraz dioda LED, która świeci się w momencie pojawienia się stabilizowanego napięcia 3,3 V.

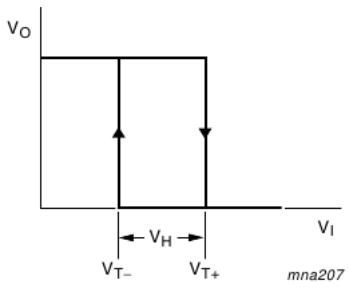
Sygnalizacja stanu pracy diodami LED

Na płytce zostały przewidziane diody LED do sygnalizacji pracy modułu Ethernet. Katody diod podłączone są do odpowiednich wyprowadzeń tego modułu. Opis znaczenia każdej diody został przedstawiony w tabeli 2.

Tabela 2. Diody sygnalizacyjne płytki

Nr wypr. IIM7010A	Oznaczenie	Nr diody LED	Co sygnalizuje
6	L_COL	1	świeci w momencie wystąpienia kolizji
8	L_100ACT	2	świeci, gdy jest połączenie w standardzie 100Base-TX i miga przy przesyłaniu danych
10	L_10ACT	3	świeci, gdy jest połączenie w standardzie 10Base-T i miga przy przesyłaniu danych
11	L_DUPX	4	świeci w trybie full-duplex
12	L_LINK	5	świeci, gdy jest nawiązane połączenie

Anoda każdej diody połączona jest poprzez rezystor o wartości 1 kΩ do napięcia +3,3 V. Rezystor ogranicza natężenie prądu. Stan niski na wyprowadzeniu modułu IIM7010A powoduje zaświecenie wybranej diody.



Rys.7. Charakterystyka transmitancji inwertera z układem Schmitta, wg [3]

Generator sygnałów reset i /reset

Osobną kwestią jest również generator sygnału *reset* i */reset*, zbudowany na układzie U2 typu 74HC14 firmy Philips (obecne NXP Semiconductors), zawierający w swej strukturze 6 inwerterów z układem Schmitta. Inwertery z układem Schmitta różnią się od standardowych posiadaniem układu progowego z histerezą. Na rysunku 7. została przedstawiona charakterystyka transmitancji pojedynczej bramki. Dla opisywanego układu, przy zasilaniu 3,3 V, wartości V_{T-}, V_{T+}, V_H są następujące: V_{T-} = 1,7 V, V_{T+} = 0,9 V, V_H = 0,8 V. W związku z powyższym, przełączania bramki Schmitta następuje przy przekraczaniu napięcia ok. 1,7 V (przy narastaniu sygnału wejściowego) i przy napięciu ok. 0,9 V (przy opadaniu tego sygnału). Wygenerowane sygnały *reset i /reset* doprowadzone są do modułu IIM7010A i służą odpowiednio: *reset* do resetowania układu W3100A, a sygnał */reset* - do resetowania układu RTL8201BL.

Połączenia magistrali systemowej i linii sterujących systemu DSM-51

Opisywany układ ma przede wszystkim łączyć magistralę danych, magistralę adresową oraz sygnały sterujące */wr*, */rd*, */int* i */csx* systemu DSM-51 z modułem IIM7010A.

Szyna danych wyprowadzona jest na złącze magistrali systemowej poprzez bufor. Bufor ten to układ typu 74HCT245, zawierający w swej strukturze osiem bramek z wyjściami trójstanowymi i sterowanym kierunkiem przesyłu danych. Bufory magistrali są zablokowane (ich wyjścia w stanie wysokiej impedancji) dopóty, dopóki linia EXB, podłączona do wyprowadzenia nr 19 (/G), nie zostanie spolaryzowana na zero (np. zwarta do masy). Zwarcie powoduje uaktywnienie buforów i możliwość wymiany danych.

Szyna adresowa została również wyprowadzona przez bufory (U26 i U27) - jednakże, ze względu na jeden kierunek przepływu danych, są to bufory typu 74HC541. Podobnie jak w przypadku układu 74HCT245, wyjścia buforów są w stanie wysokiej impedancji dopóki linia EXB, podłączona do wyprowadzeń nr 1 (/G1) i 19 (/G2), nie zostanie spolaryzowana na zero.

Zapis danych do urządzenia zewnętrznego przebiega następująco. W momencie, gdy na szynie adresowej mikrokontroler 80C51 wystawia adres z przedziału od 0x8000 do 0xF000, dekoder adresów wybiera układ podłączony do złącza magistrali systemowej poprzez ustawienie linii CSX w stan niski. Mikrokontroler następnie wystawia na magistrali danych wartość, która ma zostać zapisana do układu zewnętrznego. Dekoder adresów wytwarza sygnał RDX i CSX, zgodnie z równaniami (1) i (2).

$$RDX = \overline{A_{15} * (\overline{A_{14}} + \overline{A_{13}} + \overline{A_{12}} + A_7 * A_6) * \overline{RD}} \quad (1)$$

$$CSX = \overline{A_{15} * (\overline{A_{14}} + \overline{A_{13}} + \overline{A_{12}} + A_7 * A_6)} \quad (2)$$

Jak widać, w przypadku gdy RD=1 linia RDX jest w stanie wysokim. Linia ta, oprócz przekazywania sygnału *Read* do urządzenia, steruje również kierunkiem przepływu danych (linia DIR tego układu podłączona jest do linii RDX) przez układ U25 typu 74HCT245, będący dwukierunkowym buforem danych. W momencie, gdy linia DIR (RDX) jest w stanie wysokim, przepływ danych następuje z portu A do portu B, tj. z systemu DSM-51 do urządzenia zewnętrznego. Zatem po wystawieniu przez mikrokontroler sygnału WR=0 (należy zauważyć, że po przejściu przez bufor jest sygnałem WRX), następuje zapis danych do urządzenia zewnętrznego - ponieważ do tego układu został podany stan aktywny na sygnał wyboru CSX=0 oraz sygnał zapisu danych WRX=0.

W przypadku odczytu danych z urządzenia zewnętrznego, czyli również odwoływania się do adresów z przedziału 0x8000-0xF000, tak jak w przypadku zapisu zostaje wygenerowany sygnał CSX. Dodatkowo, gdy mikrokontroler wystawi aktywny stan sygnału RD (0 logiczne), dekoder adresu ustawia linię RDX w stan niski (zero logiczne = ON) i równocześnie linia DIR układu U25 przyjmuje stan niski (ON) - co powoduje, że kierunek

przepływ danych w układzie U25 zostaje ustalony na od portu B do portu A, czyli z systemu zewnętrznego do systemu DSM-51. Sygnał RDX trafia również do układu zewnętrznego jako sygnał RD - zatem układ zewnętrzny, po otrzymaniu sygnału CSX, adresu i sygnału RD, wystawia na swoją magistralę danych wartość z komórki o podanym adresie.

Należy zwrócić szczególną uwagę na podłączenie do urządzenia DSM-51 linii INT modułu IIM7010A, odpowiedzialnej za udostępnienie możliwości generowania przerwań od układu W3100A. Po stronie urządzenia DSM-51 linia ta oznaczona jest jako IX i podłączona jest do linii nr 6 układu U5 typu GAL16V8, pełniącego funkcję sterownika przerwań. Sterownik ten wykorzystywany jest w celu zwiększenia liczby przerwań zewnętrznych możliwych do obsługiwania przez mikrokontroler.

Do sterownika przerwań podłączone jest sześć sygnałów przerwań:

- IAD – z przetwornika analogowo-cyfrowego,
- IOI – z wejścia izolowanego galwanicznie,
- IPA – z układu 8255 (sterownika transmisji równoległej),
- IPB – z układu 8255 (sterownika transmisji równoległej),
- IX – ze złącza szyny systemowej,
- IRS – z kanału COM2.

Równania (wg [5]), odpowiedzialne za generowanie aktywnych stanów linii INT0 i INT1 w sterowniku przerwań, zostały zamieszczone poniżej. Równanie (3) dotyczy pamięci przerwania IAD; równanie (4) - pamięci przerwania IOI; równanie (5) - pamięci przerwania IPA; równanie (6) - pamięci przerwania IPB; równania (7) i (8) opisują sygnał przerwania INT0 procesora; równanie (9) - sygnał przerwania INT1 procesora; równania (10) i (11) charakteryzują sygnały wyjściowe (decydujące o priorytecie przerwań); równania (12) i (13) opisują sygnały sterujące trójstanowymi buforami wyjściowymi.

(3) $$O_0 = (\overline{D_0} * \overline{D_1} * \overline{WR} * \overline{CSIC} * IAD) + (O_0 * IAD)$$

(4) $$O_1 = (D_0 * \overline{D_1} * \overline{WR} * \overline{CSIC} * IOI) + (O_1 * IOI)$$

(5) $$O_2 = (\overline{D_0} * D_1 * \overline{WR} * \overline{CSIC} * \overline{IPA}) + (O_2 * \overline{IPA}) + PP1$$

(6)

$$O_3 = (D_0 * D_1 * \overline{WR} * \overline{CSIC} * IX * PP1) + (O_3 * IX * PP1) +$$
$$+ (D_0 * D_1 * \overline{WR} * \overline{CSIC} * \overline{IPB} * \overline{PP1}) + (O_3 * \overline{IPB} * \overline{PP1})$$

(7) $$INT0 = 0$$

(8) $$INT0.TRST = \overline{IRS}$$

(9) $$INT1 = O_0 * O_1 * O_2 * O_3$$

(10) $$D_0 = O_0 * \overline{O_1} + O_0 * O_2$$

(11) $$D_1 = O_0 * O_1$$

(12) $$D_0.TRST = \overline{RD} * \overline{CSIC}$$

(13) $$D_1.TRST = \overline{RD} * \overline{CSIC}$$

Od strony mikrokontrolera podłączona jest linia P1.1 (sygnał *PP1* w sterowniku przerwań), którą steruje mikrokontroler, linie INT0 i INT1 sterowane przez układ GAL oraz dwie linie danych D0 i D1. Układ wybierany jest sygnałem CSIC (aktywny stan = 0) generowanym przez dekoder U7 typu 74HC138 po wybraniu adresu 0xFF00.

(14) $$CSIO = \overline{A_{15} * A_{14} * A_{13} * A_{12} * \overline{A_7} * \overline{A_6}}$$

Sygnał CSIC generowany jest dwuetapowo. Do dekodera adresów U6 typu GAL16V8 doprowadzone są linie adresowe A6, A7 oraz od A12 do A15. Zgodnie z równaniem (14), dekoder adresów wystawia aktywny sygnał CSIO gdy na liniach adresowych pojawiają się adresy o wartościach z przedziału od 0xF000 do 0xFFBF. Sygnałem CSIO wybierany jest układ U7 typu 74HC138, do którego doprowadzone są adresy *A3, A4* i *A5*. Układ ten to dekoder wystawiający sygnał aktywny na jedno z ośmiu wyprowadzeń na podstawie trzybitowego adresu. Zatem, po pojawieniu się adresu 000B aktywne jest wyprowadzenie Y0, które oznaczone jest jako sygnał CSIC, dla adresu 001B aktywne jest wyprowadzenie Y1 oznaczone jako sygnał CSDA i tak dalej aż do adresu 111B, dla którego aktywne jest wyprowadzenie Y7 reprezentujące sygnał CSDB.

W sterowniku przerwań linia INT0 jest praktycznie bezpośrednio połączona z linią przerwania z kanału COM2, czyli z IRS. Natomiast inaczej jest w przypadku linii INT1. Pojawia się na niej sygnał przerwania zawsze, kiedy pojawi się przerwanie na którejś z podłączonych linii układów peryferyjnych. Natomiast żeby dowiedzieć się, od którego urządzenia nastąpiło przerwanie, należy odczytać dane z rejestru CSIC, czyli ze sterownika przerwań - widocznego pod adresem 0xFF00 zewnętrznej pamięci XRAM procesora 80C51. Tu pojawia się kolejny problem, gdyż układów podłączonych do sterownika jest 5 (6 razem z kanałem COM2 - ale ten pomijamy, ze względu na bezpośrednie połączenie z INT0), a linii danych tylko 2. Tu wykorzystywana jest wspomniana wcześniej linia P1.1. Jeżeli linia P1.1 jest w stanie wysokim, sterownik przerwań jest w trybie pracy, w którym numery przerwań przedstawiają się następująco (w kolejności od najwyższego do najniższego priorytetu przerwania, wg [6]):

- 0 – *IAD*,
- 1 – *IOI*,
- 3 – *IX*.

Natomiast gdy linia P1.1 jest w stanie niskim, numery przerwań są następujące (w kolejności od najwyższego do najniższego priorytetu przerwania, wg [6]):

- 0 – *IAD*,
- 1 – *IOI*,
- 2 – *IPA*,
- 3 – *IPB*.

Z powyższego wynika, że aby móc obsługiwać przerwania ze złącza magistrali systemowej, należy ustawić linię P1.1 w stan wysoki - co jest realizowane po resecie mikrokontrolera. Na listingu 5. przedstawiony jest sposób programowej obsługi przerwania od linii INT1 (port P3.3 mikrokontrolera 80C51).

W podprogramie obsługi tego przerwania, na początku wyłączane jest przerwanie od linii INT1 (instrukcja EX1=0) - żeby zapobiec nakładaniu się wywołania kolejnego przerwania, podczas gdy obecne nie zostało do końca obsłużone. Następnie, ze sterownika przerwań (układ U5, widoczny pod adresem 0xFF00) odczytywany jest numer urządzenia będącego źródłem przerwania INT1. Wartość odczytywana jest dwubitowo - z tego powodu w tej samej instrukcji pozostałe bity są ustawiane na 0 (iloczyn logiczny z wartością 0x03). W celu umożliwienia zgłoszenia następnego przerwania INT1, pod adres kontrolera przerwań zostaje zapisana odczytana wartość, co powoduje wyzerowanie pamięci tego sterownika. W przypadku, gdy wartość odczytana (będąca numerem urządzenia zgłaszającego przerwanie) nie została zapisana do sterownika, sterownik przerwań, po ponownym zgłoszeniu przerwania przez urządzenie, nie zmieniłby stanu linii INT1 - czyli nie zgłosiłby następnego przerwania.

Listing 5. Obsługa przerwania od modułu IIM7010A

```
void Int1(void) interrupt 2
{
    u_char status;
    unsigned char f;
    EX1 = 0;                                        // INT1 DISABLE
    f = CSIC&0x03;
    CSIC = f;
    if (f == 3){
        status = INT_REG;
        [...]
```

Odczytana wartość dwubitowa, będąca numerem urządzenia zgłaszającego przerwanie, jest porównywana z liczbą 3 - co odpowiada przerwaniu od linii *IX,* do której podłączona jest linia przerwania modułu IIM7010A. Jeżeli przerwanie nastąpiło faktycznie od złącza magistrali zewnętrznej, to uruchamiana jest właściwa procedura. W procedurze tej przepisywany jest status każdego z gniazd (od 0 do 3) do czteroelementowej tablicy I_STATUS[], przechowującej statusy gniazd i zlokalizowanej w obszarze *data*. Po obsłudze przerwania, ponownie uaktywniane jest przerwanie od linii INT1 i procedura kończy się.

Podsumowanie

Prace przeprowadzone w Instytucie Elektroenergetyki Politechniki Warszawskiej wskazują na możliwość skutecznego wykorzystania modułu łącza Ethernet/IP typu IIM7010A do współpracy z Dydaktycznym Systemem Mikroprocesorowym DSM-51 firmy MicroMade. Przedstawione w artykule rozwiązanie może zostać z powodzeniem wykorzystane w dydaktyce, rozwijając możliwości, powszechnie w niej wykorzystywanego, systemu DSM-51. Może ono stanowić bazę do przygotowania zajęć laboratoryjnych, poruszających nowe zagadnienia w procesie nauczania programowania mikrokontrolera 8051 i przedstawiających podstawowe aspekty działania najpopularniejszych protokołów sieciowych w ujęciu praktycznym. Niewielki koszt oraz poziom skomplikowania przedstawionego rozwiązania stanowią jego dodatkowe zalety w kontekście potencjalnego, szerszego zastosowania. Jedynym ograniczeniem jest przy tym posiadanie systemu DSM-51 w wersji, która posiada złącze magistrali systemowej.

Autorzy*: inż. Mateusz Chmielowiec, Politechnika Warszawska, Instytut Elektroenergetyki, E-mail: chmielom@ee.pw.edu.pl;*
inż. Marek Kośmider, Politechnika Warszawska, Instytut Elektroenergetyki;
dr inż. Ryszard Kowalik, Politechnika Warszawska, Instytut Elektroenergetyki, ul. Koszykowa 75, Gmach Mechaniki, 00-662 Warszawa, e-mail: Ryszard.Kowalik@ien.pw.edu.pl;
mgr inż. Emil Bartosiewicz, Politechnika Warszawska, Instytut Elektroenergetyki, ul. Koszykowa 75, Gmach Mechaniki, 00-662 Warszawa, E-mail: Emil.Bartosiewicz@ien.pw.edu.pl.

LITERATURA

[1] Gałka P., Gałka P.: *Podstawy programowania mikrokontrolera 8051*, wydanie IV, MIKOM, Warszawa 2005
[2] Doliński J.: Krzemowy stos TCP/IP, *Elektronika Praktyczna,* 2003, nr 10
[3] Dokumentacja: 74HC14
[4] Dokumentacja: IIM7010A
[5] Dokumentacja: DSM-51
[6] Gazarkiewicz R., Kowalik R.: *Dydaktyczny System Mikroprocesorowy DSM-51 - Ćwiczenia w języku C dla mikrokontrolera 8051 w praktyce*, wydanie I, MIKOM, Warszawa 2006