MOKSLAS – LIETUVOS ATEITIS
SCIENCE – FUTURE OF LITHUANIA

ISSN 2029-2341 print / ISSN 2029-2252 online
2011 3(1): 82–86 doi: 10.3846/mla.2011.017

Telekomunikacijų inžinerija T 180
Telecommunication Engineering T 180

Elektronika ir elektrotechnika
Electronics and Electrical Engineering

AŠTUONIŲ SKILČIŲ MIKROVALDIKLIŲ PANAUDOJIMO GALIMYBIŲ TYRIMAS *ETHERNET* TINKLO ĮRENGINIUOSE

Lech Gulbinovič

Vilniaus Gedimino technikos universitetas
El. paštas: gulbinovic@gmail.com

Santrauka. Išnagrinėtos aštuonių skilčių mikrovaldiklių panaudojimo galimybės 10 Mbps *Ethernet* tinklo įrenginiuose. Suprojektuotas ir eksperimentiškai ištirtas įrenginys, kuris sudarytas iš aštuonių skilčių mikrovaldiklio *ATmega32* ir 10 Mbps *Ethernet* valdiklio *ENC28J60*. Nustatytos įrenginio duomenų apdorojimo spartos priklausomybės nuo paketų dydžio, protokolo, programos kodo kompiliavimo režimo. Įrenginio darbas modeliuotas kaip vieno kanalo masinio aptarnavimo sistema su apribota eile. Gauti eksperimentinio tyrimo rezultatai palyginti su modeliavimo rezultatais. Atlikti mikrovaldiklio programos kodo optimizavimo tyrimai. Parodyta, kad mikrovaldiklio duomenų paketų apdorojimo sparta priklauso nuo paketų dydžio. Paketų dydžiui kintant nuo 64 B iki 1514 B, gauti tokie maksimalios paketų apdorojimo spartos rezultatai: ICMP protokolo paketus mikrovaldiklis apdoroja 1,4–2,1 Mbps sparta, UDP protokolo paketus mikrovaldiklis apdoroja 1,3–1,8 Mbps sparta. Paketų vėlinimas taip pat priklauso nuo paketų dydžio. Paketų dydžiui kintant nuo 64 B iki 1514 B, gauti tokie maksimalios paketų apdorojimo spartos rezultatai: ICMP protokolo paketų vėlinimas kinta 0,8–8,4 ms, UDP protokolo paketų vėlinimas kinta 1,1–9,6 ms. Priklausomai nuo parinkto kompiliatoriaus režimo galima padidinti įrenginio veikimo spartą. Geriausi rezultatai palyginus su –Os optimizavimo lygiu gauti parinkus –O3 optimizavimo lygį, įrenginio greitaveika padidėjo ~5–6 %, vėlinimas sumažėjo ~3 %, tačiau programos kodo dydis padidėjo 68 %.

Reikšminiai žodžiai: *Ethernet*, mikrovaldiklis, vienkanalė masinio aptarnavimo sistema, kompiliatorius, paketų apdorojimo sparta.

Įvadas

Internetas yra labai lanksti, patogi ir efektyvi duomenų perdavimo priemonė. Beveik visa duomenų perdavimo technologija internetu remiasi *Ethernet* protokolais. Kartu su *Ethernet* protokolais tinkle naudojamas TCP/IP protokolų rinkinys, garantuojantis vientisą duomenų perdavimą tinkle. Šiuo metu yra labai daug įrenginių, kurie yra prijungti prie globalaus interneto tinklo. Tai gali būti įvairias funkcijas vykdantys kompiuteriai, specialūs įrenginiai, duomenų serveriai ir pan. Duomenų ir programų naudotojams serverio funkciją gali atlikti ne tik 32/64 skilčių mikroprocesoriai, bet ir daug mažesni ir pigesni 8 arba 16 bitų mikrovaldikliai.

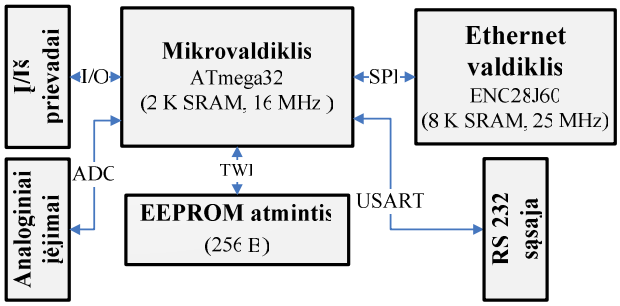
Darbo tikslas yra tinklo įrenginių su 8 skilčių mikrovaldikliais tyrimo metodikos sudarymas ir tyrimo priemonių parinkimas. Kadangi tokių įrenginių duomenų apdorojimo sparta nėra didelė, todėl svarbu nustatyti, kokio tipo uždavinius galima spręsti su 8 skilčių mikrovaldikliais *Ethernet* tinkluose, ir ar yra jų veikimo spartos padidinimo galimybės. Ištyrus tokių įrenginių charakteristikas, būtų galima apibrėžti ribas, kokio tipo uždavinius galima būtų spręsti panaudojus 8 skilčių mikrovaldiklius *Ethernet* tinkluose.

Įrenginio projektavimas

Tyrimams buvo suprojektuotas įrenginys, kuris skirtas kitų įrenginių valdymui, *Ethernet* sąsajos konvertavimui į RS-232 sąsają ir jutiklių duomenų parodymų persiuntimui *Ethernet* tinklu. Įrenginys valdomas per interneto naršyklę. Jis leidžia keisti programos kodą, aparatinės dalies sudėtį bei funkcijų rinkinį.

Suprojektuoto įrenginio svarbiausia sudedamoji dalis yra AVR šeimos aštuonių skilčių mikrovaldiklis *ATmega32*. Jo pagrindiniai parametrai: 32 KB programų atmintis, 2 KB operatyvioji atmintis, 16 MIPS veikimo sparta, aparatiškai įgyvendintos SPI, I2C ir USART sąsajos. *Ethernet* valdikliu buvo pasirinktas *Microchip ENC28J60* valdiklis, kurio pagrindiniai parametrai: PHY ir MAC OSI lygmenų aparatinis apdorojimas, 8 KB buferio dydis, 10*Base-T Ethernet* standartas, SPI sąsaja. Įrenginio struktūrinė schema parodyta 1 pav. Visus įrenginio komponentus apjungia ir valdo mikrovaldiklis.

Programa mikrovaldikliui buvo parašyta C kalba. Programos kodo kompiliavimui buvo naudojamas AVR-GCC nemokamas atvirojo kodo kompiliatorius ir AVR Studio 4 mikrovaldiklių gamintojo programa (Barnett 2006; AVR-GCC library 2010; Furman 2007).



1 pav. Įrenginio struktūrinė schema
Fig. 1. Device block scheme

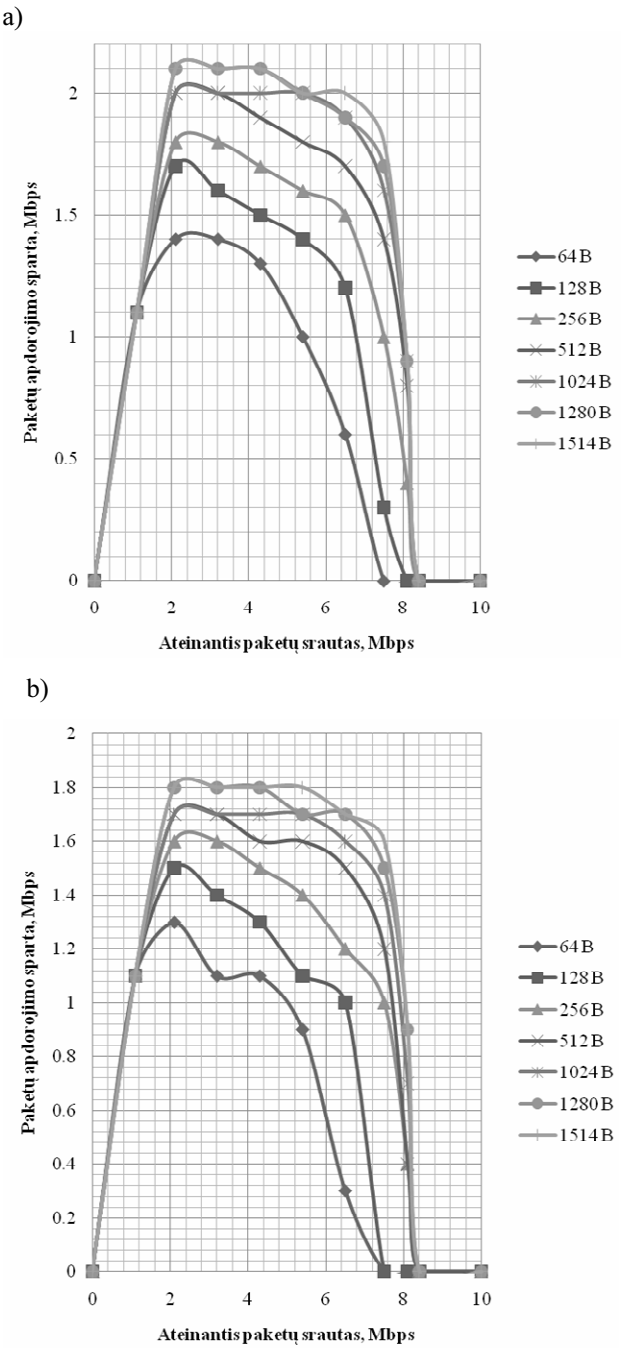
Tyrimas

Pirmiausiai reikėjo ištirti suprojektuoto įrenginio charakteristikas, išanalizuoti įrenginio veikimo spartos didinimo galimybes, nustatyti atsparumo DoS (angl. *Denial of Service)* atakoms lygį, sudaryti įrenginio matematinį modelį bei palyginti modeliavimo ir eksperimentinio tyrimo rezultatus.

Mikrovaldiklio duomenų apdorojimo spartos tyrimui buvo naudojamos Linux operacinės sistemos priemonės: srauto generatoriai *hping3*, *ping*, *tcpreplay* ir paketų analizės priemonė *Wireshark* (Sobell 2007; Sander 2009). Tinklo paketų srauto greitis buvo didinamas nuo 1 iki 10 Mbps, paketų dydžiai buvo parinkti pagal RFC 2544 specifikaciją (64, 128, 256, 512, 1024, 1280, 1514 B) (Bradner *et al.* 1999). Tyrimui panaudoti du protokolai ICMP (angl. *Internet Control Message Protocol*) – 3-čiojo OSI lygmens protokolas ir UDP (angl. *User Datagram Protocol*) – 4-tojo OSI lygmens protokolas (Goralski 2009; Plėštys *et al.* 2008). Šie protokolai priklauso skirtingiems OSI lygmenims ir yra plačiai naudojami. Tyrimams buvo pasirinkti skirtingų lygmenų protokolai tam, kad būtų palyginta paketų apdorojimo sparta skirtinguose OSI lygmenyse. Eksperimentiškai nustatytos paketų apdorojimo priklausomybės nuo ateinančio paketų srauto pateiktos 2 pav.

Iš grafikų matome, kad paketų apdorojimo sparta priklauso ne tik nuo ateinančio paketų srauto, bet ir nuo paketų tipo bei nuo paketų dydžio. Maksimali paketų apdorojimo sparta ICMP paketams kinta nuo 1,4 iki 2,1 Mbps, o UDP paketams nuo 1,3 iki 1,8 Mbps. Iš gautų rezultatų paaiškėja, kad pasirinktus įrenginius galima naudoti tokiose *Ethernet* tinklo vietose, kur duomenų srautai neviršija 2 Mbps. Prie didesnės duomenų srauto spartos įrenginys pradės prarasti duomenų paketus.

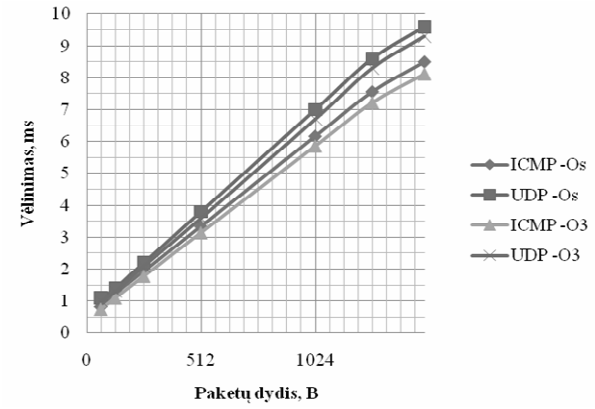
Paketų vėlinimas priklauso nuo paketų dydžio. Paketų vėlinimo priklausomybė nuo paketų dydžio parodyta 3 pav.



2 pav. Paketų apdorojimo priklausomybė nuo ateinančio paketų srauto: (a) ICMP paketai; (b) UDP paketai
Fig. 2. Packet processing characteristics: (a) ICMP packets; (b) UDP packets

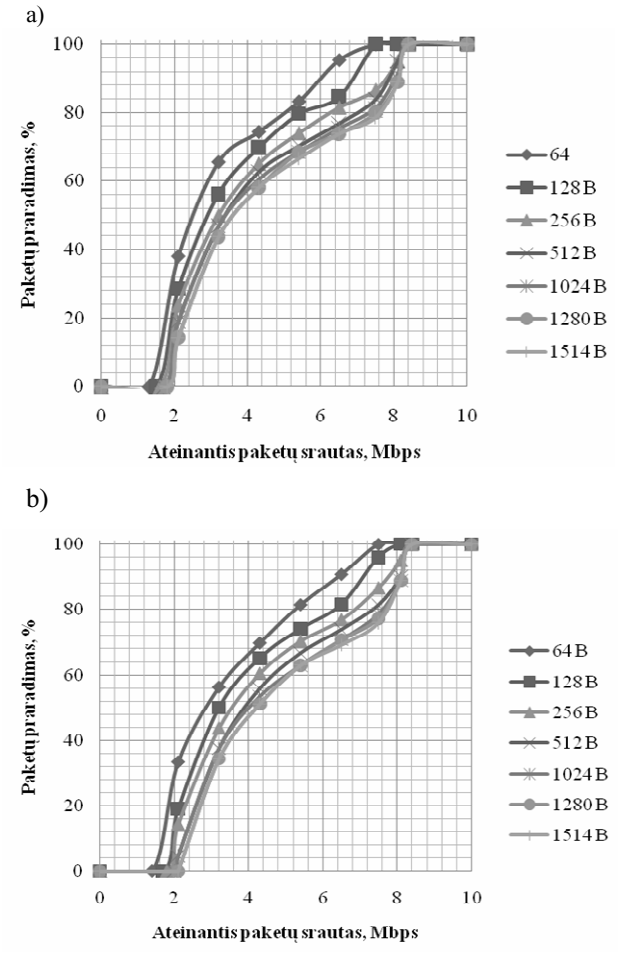
Iš gautų rezultatų matyti, kad paketų vėlinimo trukmė didėja tada, kai didėja paketų dydis. Ši priklausomybė yra beveik tiesinė.

Iš 2 pav. pateiktų grafikų apskaičiuotos paketų praradimo tikimybių priklausomybės nuo ateinančio paketų srauto pateiktos (4 pav., a, b).



3 pav. Paketų vėlinimo priklausomybė nuo paketų dydžio ir nuo programos kodo optimizavimo lygio: ICMP paketai, UDP paketai

Fig. 3. Packet latency characteristic, ICMP packets, UDP packets



4 pav. Paketų praradimo priklausomybės nuo ateinančio paketų srauto: a) ICMP paketai; b) UDP paketai

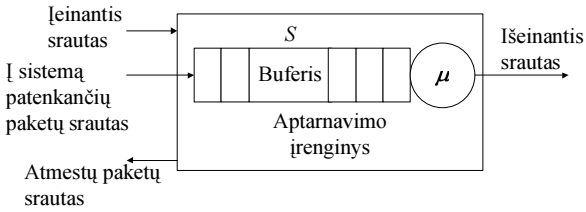
Fig. 4. Packet lost characteristics, (a) ICMP packets, (b) UDP packets

Iš grafikų matyti, kad paketų praradimo tikimybė priklauso nuo sistemos panaudojimo faktoriaus ρ (1).

Kol panaudojimo faktorius <1, praradimo tikimybė lygi 0 %. Didinant sistemos apkrovimą, kai panaudojimo faktorius >1, paketų praradimo tikimybė didėja eksponentiniu dėsniu. Pasiekus DoS atakos atsparumo lygį ~8 Mbps, paketų praradimo tikimybė išauga iki 100 %. DoS atakos metu perkraunami įrenginio resursai.

Matematinis modelis

Suprojektuotas įrenginys buvo modeliuojamas vieno kanalo masinio aptarnavimo sistema (MAS) su apribota eile (Gross *et al.* 1998; Plėštys *et al.* 2008). Vieno kanalo masinio aptarnavimo sistemos modelis pateiktas (5 pav.).



5 pav. Vieno kanalo MAS su apribota eile: S – buferio dydis; μ – paketų aptarnavimo sparta

Fig. 5. One service queuing system with limited ticket counter, S – buffer size; μ – packets processing speed

Pagrindinės sistemos darbą apibrėžiančios charakteristikos:

– sistemos panaudojimo faktorius:

$$\rho = \frac{\lambda}{\mu}, \tag{1}$$

čia: μ – paketo aptarnavimo sparta; λ – ateinančių paketų intensyvumas;

– paketų atmetimo tikimybė:

$$P_s = \frac{(1-\rho)\rho^S}{1-\rho^{S+1}}, \tag{2}$$

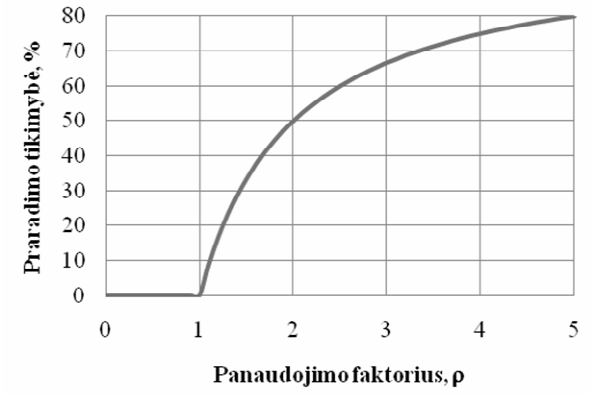
čia: S – buferio dydis;

– vidutinis eilėje laukiančių paketų skaičius:

$$N_q = \frac{\rho^2}{1-\rho} - \frac{\rho(S+\rho)}{1-\rho} \cdot P_S. \tag{3}$$

Paketų atmetimo tikimybė nepriklauso nuo buferio ir paketų dydžio, tačiau priklauso nuo panaudojimo faktoriaus. Teoriškai apskaičiuota paketų praradimo priklausomybė nuo panaudojimo faktoriaus prie skirtingų buferio dydžių n = 50 ir n = 100 pateikta 6 pav.

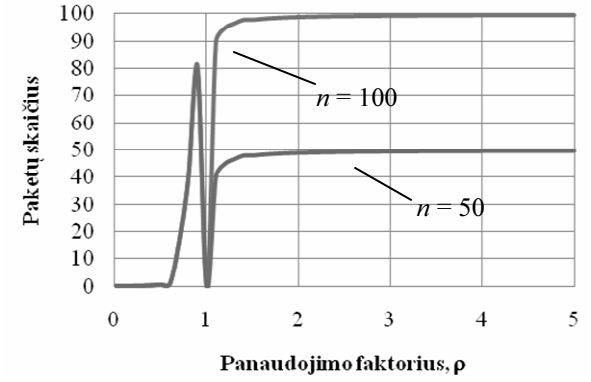
Gauti eksperimentiniai ir modeliavimo rezultatai kokybiškai sutampa. Teoriškai apskaičiuota paketų praradimo tikimybės priklausomybė (6 pav.) yra tokio pat pobūdžio kaip ir eksperimentiškai gautos (4 pav.).



6 pav. Paketų praradimo tikimybė, kai buferio dydis $n = 50$ ir $n = 100$ (grafikai sutampa)

Fig. 6. Packet lost with ticket booking counter $n = 50$ and $n = 100$

Buferio užpildymo priklausomybė nuo sistemos panaudojimo faktoriaus parodyta (7 pav.).



7 pav. Vidutinis paketų skaičius buferyje, kai buferio dydis $n = 100$ ir $n = 50$

Fig. 7. Average packet sum in buffer, then buffer size is $n = 100$ and $n = 50$

Vidutinis paketų skaičius eilėje priklauso nuo sistemos panaudojimo faktoriaus, kai panaudojimo faktorius > 1 buferis staigiai persipildo, o persipildymo greitis mažai priklauso nuo buferio dydžio (Gross *et al.* 1998).

Optimizavimas

Programos veikimo sparta priklauso nuo: programos algoritmo, programavimo kalbos, programos kodo kompiliatoriaus. Programos rašymui buvo pasirinkta C# programavimo kalba, sudarytas minimalus programos funkcijų sąrašas. Pasirinktas C kalbos kompiliatorius mikrovaldikliams AVR-GCC (Barnett 2006). Tam, kad būtų optimizuotas programos kodas, buvo keičiami kompiliatoriaus nustatymai. Priklausomybės nuo pasirinkto kompiliatoriaus kompiliavimo lygio ir programos kodo apimties bei operatyviosios atminties yra pateiktos 1 lentelėje.

1 lentelė. Programos kodo kompiliavimo režimų palyginimas

Table 1. Source code compilation mode comparison

Kompiliavimo lygis	Duomenų atmintis, B	Operatyvioji atmintis, B	Užimta duomenų atmintis, %	Užimta duomenų operatyvioji atmintis, %
–O0	37792	1918	115,3	93,7
–O1	21116	1908	64,4	93,2
–O2	20398	1908	62,2	93,2
–O3	33216	1904	101,4	93,0
–Os	19734	1908	60,2	93,2

Lyginant mikrovaldiklio duomenų apdorojimo spartą, kuri gauta naudojant –O0, –O1, –O2 ir –O3 programos kodo optimizavimo lygius, jokių programos veikimo spartos pokyčių nepastebėta. Didžiausias skirtumas mikrovaldiklio duomenų apdorojimo spartoje, yra užfiksuotas naudojant –O3 ir –Os programos kodo optimizavimo lygius. Lygis –Os geriausiai suspaudžia programos kodą dėl vėlinimo ciklų optimizavimo. Lygis –O3 mažiau suspaudžia programos kodą. Geriausias rezultatas gaunamas parinkus –O3 optimizavimo lygį, tuomet įrenginio greitaveika padidėja ~6 %, o vėlinimas sumažėja ~3 %, palyginus su –Os optimizavimo lygiu, tačiau programos kodo apimtis šiuo atveju padidėja 68 %.

Išvados

1. Mikrovaldiklio duomenų paketų apdorojimo sparta priklauso nuo paketų dydžio. Paketų dydžiui kintant nuo 64 B iki 1514 B gauti tokie maksimalios paketų apdorojimo spartos rezultatai: ICMP protokolo paketus aštuonių skilčių mikrovaldiklis apdoroja 1,4–2,1 Mbps, UDP protokolo paketus mikrovaldiklis apdoroja 1,3–1,8 Mbps sparta. Paketų vėlinimas taip pat priklauso nuo paketų dydžio. Paketų dydžiui kintant nuo 64 B iki 1514 B gauti tokie maksimalaus paketų vėlinimo rezultatai: ICMP protokolo paketų vėlinimas kinta 0,8–8,4 ms, UDP protokolo paketų vėlinimas kinta 1,1–9,6 ms.
2. Parenkant kompiliatoriaus nustatymus, galima padidinti įrenginio veikimo spartą. Geriausias rezultatas gaunamas parinkus –O3 optimizavimo lygį. Palyginus su –Os optimizavimo lygiu, įrenginio greitaveika padidėja ~5–6 %, vėlinimas sumažėja ~3 %, tačiau programos kodo apimtis šiuo atveju padidėja 68 %.

Literatūra

AVR-GCC library. 2010 [interaktyvus], [žiūrėta 2010 10 27]. Prieiga per internetą: <http://www.nongnu.org/avr-libc/>.

Bradner, S.; McQuaid, J. 1999. *Benchmarking Methodology for Network Interconnect Devices* [interaktyvus]. Network Working Group, Harvard University, NetScout Systems [žiūrėta 2010 10 27]. Prieiga per internetą: http://www.ietf.org/rfc/rfc2544.txt

Furman, B. J. 2007. *How to use AVR Studio 4 to program the ATmega128. ME 106 Intro to Mechatronics* [interaktyvus], [žiūrėta 2010 10 27]. Prieiga per internetą: <http://www.engr.sjsu.edu/bjfurman/courses/ME106/ME106pdf/UsingAVRStudio4.pdf>.

Goralski, W. 2009. *The Illustrated Network. How TCP/IP Works In A Modern Network.* London: Morgan Kaufmann. 829 p.

Gross, D.; Harris, C. M. 1998. *Fundamentals of Queueing Theory.* Hangover: Wiley-Interscience. 464 p.

Plėštys, R.; Kavaliūnas, R.; Vilutis, G.; Lagzdynytė, I.; Liutkauskas, V. 2008. *Kompiuterių tinklai.* Kaunas: Technologija. 152 p.

Barnett, R. H. 2006. *Embedded C Programming And The Atmel AVR.* New York: Delmar Cengage Learning. 560 p.

Sobell, M. G. 2007. *A Practical Guide to Ubuntu Linux.* Massachusetts: Prentice Hall. 1141 p.

Sander, V. 2009. *Beginning the Linux Command Line.* New York: Apress. 381 p.

FEASIBILITY STUDY OF 8-BIT MICROCONTROLLER APPLICATIONS FOR ETHERNET

L. Gulbinovič

Abstract

Feasibility study of 8-bit microcontroller applications for Ethernet is presented. Designed device is based on *ATmega32* microcontroller and 10 Mbps Ethernet controller *ENC28J60*. Device is simulated as mass queuing theoretical model with ticket booking counter. Practical explorations are accomplished and characteristics are determined. Practical results are compared to theoretical ones. Program code and device packet processing speed optimization are discussed. Microcontroller packet processing speed and packet latency depend on packet size. For ICMP protocol packet processing speed varies 1.4–2.1 Mbps, latency – 0.8–8.4 ms. UDP protocol packet processing speed varies 1.3–1.8 Mbps, latency – 1.1–9.6 ms. Packet processing speed depends on compilation settings and program code compression level. Best results are reached on optimization level -O3, then speed increased ~3% but program code size increased 68% comparing to –Os optimization level.

Keywords: Ethernet, microcontroller, mass queuing theory model, compilation, packet processing speed, optimization.