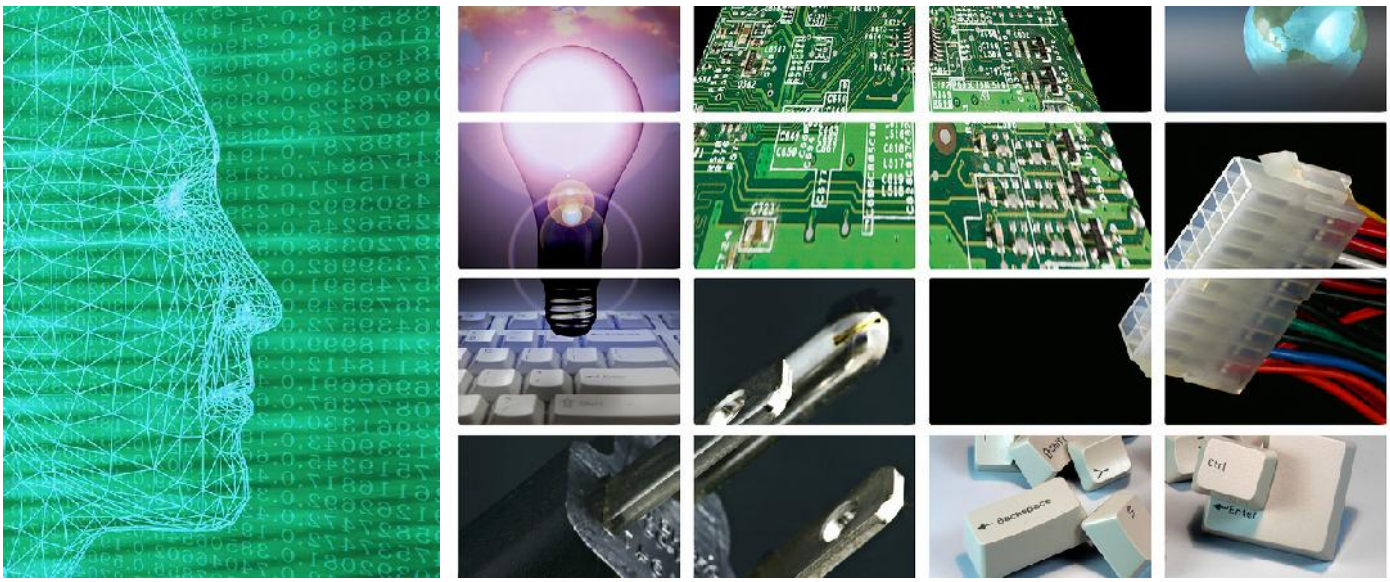
ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - ВАРНА
TECHNICAL UNIVERSITY OF VARNA

Година XII, Брой 2/2014

КОМПЮТЪРНИ НАУКИ И ТЕХНОЛОГИИ



Bulgaria Communications Chapter

Faculty of Computing & Automation

COMPUTER SCIENCE AND TECHNOLOGIES

Втора научна конференция с международно участие
"Компютърни науки и технологии"
26-27 септември, 2014 г. Варна, България

Year XII, Number 2/2014

Компютърни науки и технологии

Издание
на Факултета по изчислителна техника и автоматизация
Технически университет - Варна

Редактор: гл. ас. д-р Ю. Петкова
Гл. редактор: доц. д-р П. Антонов

Печат: ТУ-Варна

За контакти:

Технически университет - Варна
ФИТА
ул. „Студентска” 1, 9010 Варна,
България
тел./факс: (052) 383 320
e-mail: peter.antonov@ieee.org
yulka.petkova@tu-varna.bg

ISSN 1312-3335

Computer Science and Technologies

Publication
of Computing and Automation Faculty
Technical University of Varna

Editor: Ass. Prof. Y. Petkova, PhD
Chief Editor: Assoc. Prof. P. Antonov, PhD

Printing: TU-Varna

For contacts:

Technical University of Varna
Faculty of Computing and Automation
1, Studentska Str., 9010 Varna,
Bulgaria
Tel/Fax: (+359) 52 383 320
e-mail: peter.antonov@ieee.org
yulka.petkova@tu-varna.bg

ISSN 1312-3335

25 ГОДИНИ ФАКУЛТЕТ ПО ИЗЧИСЛИТЕЛНА ТЕХНИКА И АВТОМАТИЗАЦИЯ

Факултетът по изчислителна техника и автоматизация (ФИТА) при ТУ – Варна е създаден с Решение № 195 на Министерския съвет на Република България от 27.12.1989 г. В неговия състав са включени четири катедри: „Автоматизация на производството”, „Изчислителна техника”, която през 2002 г. е преименувана на „Компютърни науки и технологии”, „Математика” и „Физика”. 25-годишния си юбилей Факултетът посреща като едно от водещите учебни и научни звена на университета с утвърден авторитет в страната и в чужбина.

Към момента в състава на Факултета са включени две профилиращи катедри: „Автоматизация на производството” (АП) и „Компютърни науки и технологии” (КНТ).

Катедра АП е създадена през 1970 г. и понастоящем отговаря за обучението по специалностите „Автоматика, информационни и управляващи компютърни системи” и „Роботика и мехатроника” за всички образователно-квалификационни степени, както и за придобиването на ОНС „Доктор” по специалностите: „Теория на автоматичното управление” и „Автоматизация на производството”. През 2010 г. катедрата тържествено отпразнува своята 40-годишнина.

Катедра КНТ отговаря за обучението по специалности „Компютърни системи и технологии” и „Софтуерни и Интернет технологии” за ОКС „Бакалавър” и „Магистър”, за общообразователното компютърно обучение в университета и за обучението за придобиване на ОНС „Доктор” по специалностите „Системно програмиране”, „Компютърни системи, комплекси и мрежи” и „Автоматизирани системи за обработка на информация и управление”. През 2013 г. катедрата отбеляза три важни годишнини: 45 години от създаването си, 45 години опит в базовото компютърно обучение и 30 години опит в подготовката на компютърни инженери.

Катедрите на факултета разполагат с богата материално-техническа база, която постоянно се усъвършенства с участието и на водещи международни фирми. Академичният състав на факултета включва 50 щатни преподаватели, от които 21 (1 професор и 20 доценти) са хабилитирани, а други 10 притежават ОНС „Доктор”. Преподавателите и докторантите участват активно в национални и международни проекти, както и в програмите за международно сътрудничество, към които се привличат и най-добрите студенти. Студентите във факултета са с висока успеваемост, участват активно в културно-масовите и спортните мероприятия. Болшинството от завършилите досега над 5000 компютърни инженери и инженери по автоматика се реализират много успешно както в страната, така и в чужбина.

ЧЕСТИТ ЮБИЛЕЙ НА ВСИЧКИ СТУДЕНТИ, ПРЕПОДАВАТЕЛИ И СЛУЖИТЕЛИ ВЪВ ФАКУЛТЕТА. ЗДРАВЕ И НОВИ УСПЕХИ.

Доц. д-р инж. Петър Антонов
Декан на ФИТА

25 YEARS FACULTY OF COMPUTING AND AUTOMATION

Faculty of Computing and Automation (FCA) in the Technical University of Varna was established by Decision № 195 of the Council of Ministers of the Republic of Bulgaria at 27^{-th} Dec. 1989. At the beginning it included four departments: "Automation", "Computing" which in 2002 was renamed "Computer science and engineering", "Mathematics" and "Physics". Its 25^{-th} anniversary the Faculty hails as one of the leading teaching and research units of the university with recognised prestige both in the country and abroad.

Currently, the Faculty includes two profiling Departments: "Automation", and "Computer Science and Engineering" (CSE).

The Automation Department was established in 1970 and is currently responsible for teaching in "Automation, Information and Control Systems" and "Robotics and Mechatronics" for all educational and qualification degrees as well as training to acquire the PhD (doctor) educational and scientific degrees in the specialties "Control Theory" and "Automation". In 2010 the Department officially celebrated its 40^{-th} anniversary.

CSE Department is responsible for teaching in "Computer Systems and Technologies" and "Software and Internet Technologies" for the "Bachelor" and "Master" degree for the mainstream computer training at the university and training for acquiring scientific degree "Doctor" in the specialties "System Programming", "Computer systems, complexes and networks" and "Automated Information Processing and Control". In 2013 the Department noted three important anniversaries: 45 years since its establishment, 45 years of experience in basic computer training and 30 years of experience in the training of computer engineers.

The departments of the Faculty have rich material and technical facilities available, which are constantly being updated with the participation of leading international companies. The academic staff of the Faculty includes 50 full-time lecturers, of which 21 (1 professor and 20 associated professors) with academic rank, and other 10 have "Doctor" (PhD) degree. Lecturers and doctoral students actively participate in national and international projects as well as in the programs for international cooperation, to which the best students are attracted. Students of the Faculty have a high success rate; they actively participate in the mass cultural and sporting events. The majority of graduates, more than 5,000 computer and automation engineers, realise very successful both within the country and abroad.

HAPPY ANNIVERSARY TO ALL THE STUDENTS, LECTURERS AND STAFF OF THE FACULTY. WISHING YOU HEALTH AND NEW SUCCESSES!

Assoc. Prof. Peter Antonov, PhD
Dean of FCA

Уважаеми колеги,

От името на Организационния и Програмния комитети на конференцията „Компютърни науки и технологии’2014” Ви благодаря за участието в нея и изнесените интересни доклади по актуални теми, свързани с развитието и приложението на Информационните и компютърни технологии. Много от докладите са посветени на интердисциплинарни изследвания, което е свидетелство за активното навлизане на ИТК технологиите в различни аспекти на изследователската дейност, особено в свързаните с автоматизация и управление. Радостен е фактът, че редом с утвърдили се изследователи и преподаватели в научните секции представиха свои идеи и разработки и много студенти и докторанти.

Надявам се с Вас да се срещаме и общуваме и на следващите издания на конференцията, която катедра „Компютърни науки и технологии” се стреми да превърне в ежегоден научен и научно-практически форум, чиято цел е да развие и укрепи сътрудничеството и обмяната на опит и идеи, както в научната, така и в преподавателската и образователната дейности.

С пожелание за здраве, щастие и успехи! До нови срещи на конференция „Компютърни науки и технологии’2015”!

Септември, 2014 г.

доц. д-р инж. Надежда Рускова,
Ръководител катедра КНТ

Dear colleagues,

On behalf of the Organizing and Program Committees of the Conference "Computer Science and Technologies'2014" thank you for your participation and the presented reports on interesting topics related to the development and the application of the information and computer technologies. Many of the reports have been devoted to interdisciplinary research, which is a testament to the active uptake of ICT technology in different aspects of research, especially related to automation and control. It’s joyful the fact that along with reputed researchers and lecturers many undergraduate and graduate students presented their ideas and developments in scientific sections.

Hope we meet and communicate at subsequent editions of the conference, that the Computer Science and Engineering Department aims to make an annual scientific and practical forum whose purpose is to develop and strengthen cooperation and exchange of experience and ideas, both in scientific, teaching and educational activities.

Wishing you health, happiness and success! See you soon at the "Computer Science and Technologies'2015" conference!

September, 2014

Assoc. Prof. Nadezhda Ruskova, PhD
Head of Computer Science and Engineering Department

СПОНСОРИ НА КОНФЕРЕНЦИЯТА

http://www.telecoms.bg	**ТЕЛЕКОМС ООД** е телекомуникационен оператор, предоставящ услуги на регионални и локални интернет доставчици.
http://www.eurorisksystems.com	**EuroRisk Системи ООД** предоставя софтуерни системи и услуги за банки, застрахователни и финансови институции и други доставчици на финансови услуги.
http://www.asicdepot.com	**ASIC Depot** предоставя разширени услуги по проектиране , дизайн и верификация на мрежови процесори, мултимедия, контролери и периферия, мобилни приложения, софтуер за проверка и др.
http://www.bg.adastragrp.com	**АДАСТРА** предлага информационен мениджмънт - пълна гама услуги, насочени към рационализация и ускоряване на бизнес процеси или обновяване на фирмена технологична инфраструктура.
http://www.researchmetrics.com	The complete ecosystem of tools and services for your research business™
www.zariba.com	Based on free-to-play or pay-per-download models, using advanced social and connected features, localized in 12 languages and available in many countries, Zariba's games address a wide user base.
http://www.158ltd.com	Проектиране и изграждане на софтуерни приложения както за клиенти на фирмата, така и модерни и атрактивни продукти – създадени и администрирани от **158ltd.**
http://ciscoacademy.tu-varna.bg	Локалната **Cisco Академия** при ТУ-Варна провежда курсове по най-новите учебни програми на Cisco Networking Academy - Cisco CCNA Exploration и Cisco CCNA Security на български и английски език.
http://msacademy.tu-varna.bg	Академията предлага модерно обучение и практически опит. Обучението се провежда от компетентни инструктори, сертифицирани по актуалните технологии на Microsoft.

СБОРНИК ДОКЛАДИ

Втора научна конференция с международно участие
"Компютърни науки и технологии"
26-27 септември, 2014 г.
Варна, България

Second Scientific International Conference
Computer Sciences and Engineering
26-27 September, 2014
Varna, Bulgaria

PROCEEDINGS

Организационен комитет	Organising committee
Председател:	**Chairman:**
Надежда Рускова – ръководител на катедра „Компютърни науки и технологии”	**Nadezhda Ruskova** – Head of Computer Science and Engineering Department
Членове:	**Members:**
Слава Йорданова	Slava Yordanova
Христо Вълчанов	Hristo Valchanov
Виолета Божикова	Violeta Bozhikova
Христо Ненов	Hristo Nenov
Марияна Стоева	Mariana Stoeva
Венета Алексиева	Veneta Aleksieva
Милена Карова	Milena Karova
Ивайло Пенев	Ivaylo Penev
Юлка Петкова	Yulka Petkova
Деница Радева	Denitsa Radeva

Програмен комитет	Program committee
Председател:	**Chairman:**
Петър Антонов – декан на Факултета по изчислителна техника и автоматизация	**Peter Antonov** – Dean of Computing and Automation Faculty
Членове:	**Members:**
Елена Рачева	Elena Razcheva
Димитър Тянев	Dimitar Tyanev
Анатолий Антонов	Anatoliy Antonov
Митко Митев	Mitko Mitev
Трифон Русков	Trifon Ruskov
Михаил Шестопалов	Mikhail Shestopalov
Розалина Димова	Rosalina Dimova
Емил Маринов	Emil Marinov
Радослав Вробел	Radoslav Vrobel
Пламенка Боровска	Plamenka Borovska

СЪДЪРЖАНИЕ **CONTENTS**

СЕКЦИЯ 3
АВТОМАТИКА И КОМПЮТЪРНИ СИСТЕМИ ЗА УПРАВЛЕНИЕ

Втора научна конференция с международно участие
"Компютърни науки и технологии"
26-27 септември, 2014 г.
Варна, България

Second Scientific International Conference
Computer Sciences and Engineering
26-27 September, 2014
Varna, Bulgaria

SECTION 3
AUTOMATION AND COMPUTER CONTROL SYSTEMS

16 КАНАЛЕН АЦП С ВИРТУАЛНО ПОВИШАВАНЕ НА РАЗРЯДНОСТТА

Илия Ив. Хаджидимов, Кръстин Кр. Йорданов

Резюме: В настоящата разработка е демонстрирана микропроцесорна система за аналогоцифрово преобразуване с 16 канала посредством мултиплексор/демултиплексор (MUX/DEMUX). За постигане на по-висока точност на измерване се използва метод на фирма ATMEL за повишаване на резолюцията чрез „oversampling", което представлява виртуално увеличаване на базовата 10 битова разрядност чрез обработка на повече данни от АЦП.
Ключови думи: АЦП, мултиплексор, oversampling

16-channel virtually increased ADC resolution

Iliya I. Hadzhidimov, Krystin K. Yordanov

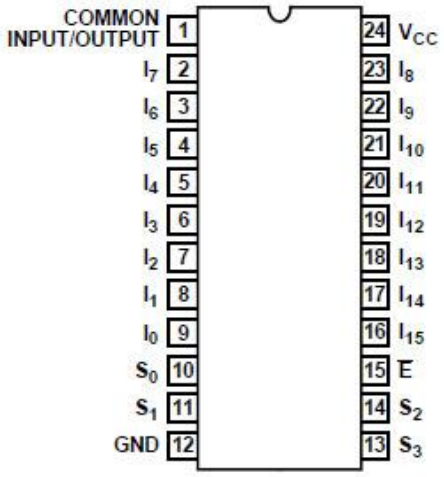
Abstract: A 16-channel MUX/DEMUX ADC microprocessor measurement system with oversampling, based on ATMEL technology has been performed. The oversampling causes enhancing the resolution from 10 bits to 16 bits.
Keywords: ADC, MUX/DEMUX, oversampling

1. Увод

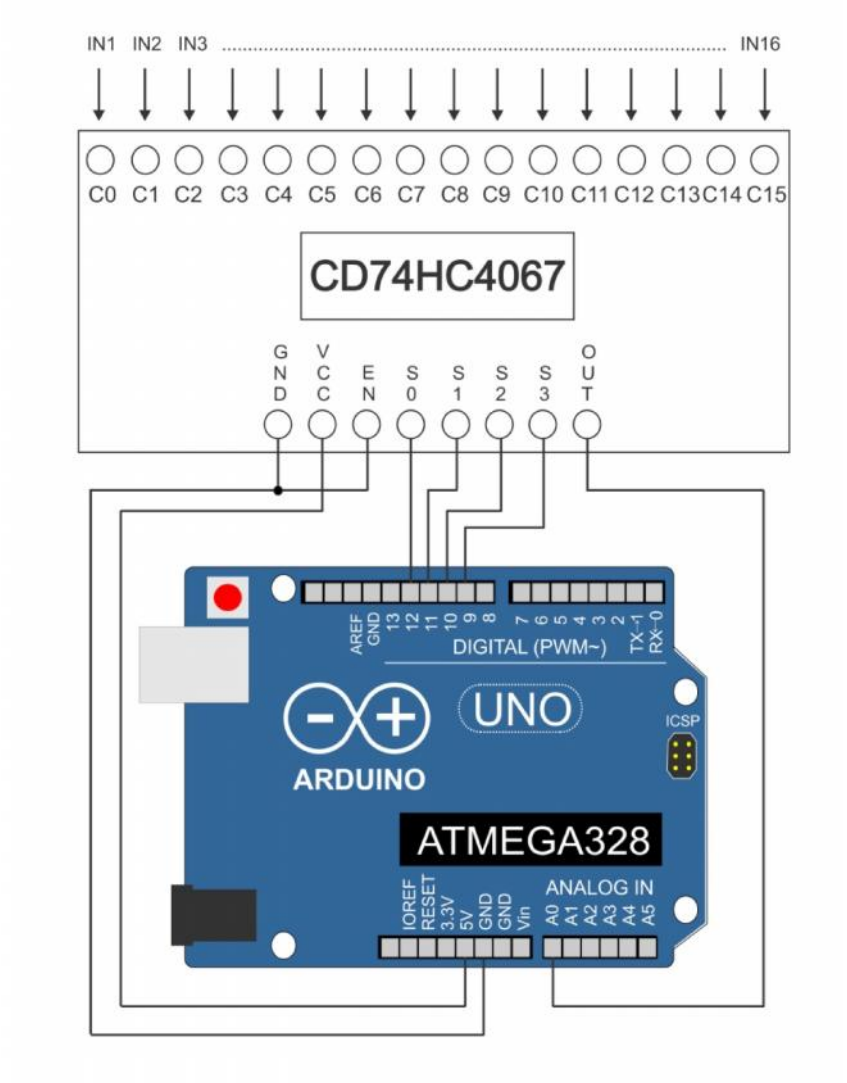
Разработката представлява микропроцесорна система на базата на ARDUINO. При необходимост от измерване на повече от 5 аналогови величини, ATMEGA328 не може да бъде използван самостоятелно. Поради тази причина е приложен 16 канален мултиплексор/демултиплексор CD74HC4067 на Texas Instruments в комбинация със софтуерна технология на фирмата ATMEL, която позволява да бъде увеличена базовата разрядност на аналогоцифровото преобразуване чрез увеличаване броя на показанията от АЦП.

2. АЦП в комбинация с мултиплексор/демултиплексор

При необходимост от измерване на повече аналогови сигнали от наличния брой аналогови портове на микропроцесорната система, обикновено се използват аналогови мултиплексори. В случая, поради необходимостта от измерване на повече аналогови величини е избран CD74HC4067, фигура 1, [1]. Той представлява цифрово управляван превключвател с възможност за използване на входните аналогови сигнали и като изходни за следващо устройство в схемата, например усилвател или микропроцесорен модул (MCU). Управлението му се извършва посредством 4 цифрови канала на ARDUINO с нули и единици в съответната комбинация за 16-те канала, чрез което се постига режим "LOW", "HIGH". Захранването е в границите от 2V до 6V и може да се осъществи от ARDUINO, фигура 2. В разработката е използвана схема на „Geeetech" [4] за CD74HC4067.



Фиг. 1. Изглед отгоре с означение на пиновете на CD74HC4067, [1].



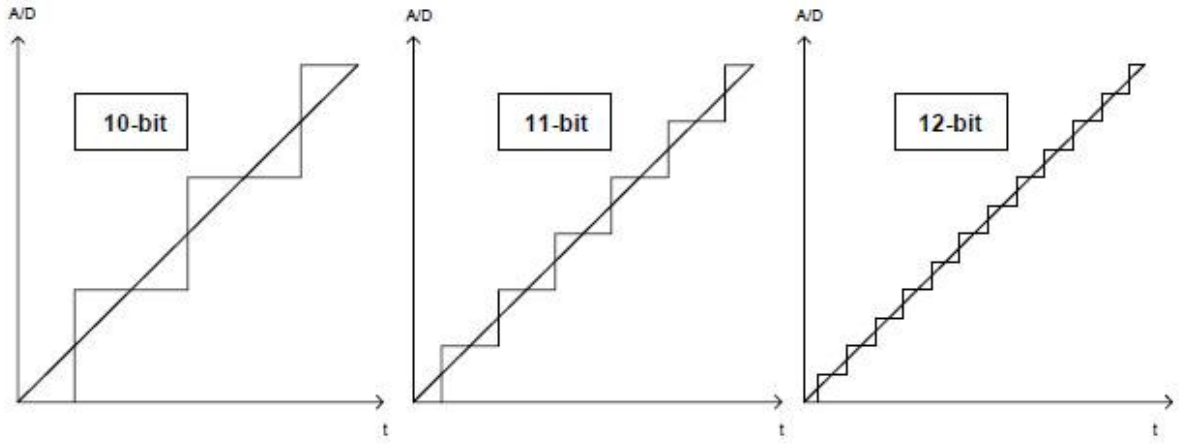
Фиг.2. Схема на микропроцесорната система.

3. Създаване на виртуално АЦП с по-висока от базовата 10-битова разрядност

Платформата ARDUINO на базата на ATNEGA328 е с 10 битово аналогоцифрово преобразуване, което дава възможност за измерване до 5V с точност около 5 mV. В редица

случаи тази точност не е достатъчна, поради естеството на измерваните величини. Този недостатък може да се избегне посредством усилване на измерваните аналогови величини, което от своя страна прави микропроцесорната система по-скъпа и ненадеждна. За преодоляване на това, в настоящата разработка е приложена идея, разработена от ATMEL – AVR121 Enhancing ADC resolution by oversampling, [2]. AVR контролерите на ATMEL са също както ATMEGA328 с 10 битово аналогоцифрово преобразуване. Методът се базира на използване на повече показания, получени от АЦП, с цел постигане на по-точно регистриране на аналоговата величина за сметка на динамичните характеристики на АЦП. По този начин честотата на семплиране се поставя във функция на прескалиране на "CPU clock".

Недостатък на този метод е невъзможността за компенсиране на интегралната нелинейност при измерването. На фигура 3 е представена графично идеята на метода за 10, 11 и 12 битово преобразуване.



Фиг. 3. Графична интерпретация на идеята за "oversampling", [2]

Методът на ATMEL е прилаган широко в практиката. Той дава възможност за създаване на виртуално АЦП с по-висока базова разрядност от тази, с която функционира MCU.

Gabriel Staples [3] представя готова библиотека за ARDUINO, реализираща идеята за "oversampling". В таблица 1 са представени характеристиките за различните резолюции, от 10 до 16 битово преобразуване.

Таблица 1. Зависимост на точността и времето за измерване от резолюцията, [3]

Резолюция, bit	**Допълнителни битове**	**Максимално показание на ADC**	**Точност на измерване, mV**	**Брой показания за постигане на резолюцията**	**Максимален "Sample rate" Hz**	**Време за прочитане на тази резолюция, sec**
10	0	1023	4.8876	1	8300	0.000120482
11	1	2046	2.4438	4	2075	0.000481928
12	2	4092	1.2219	16	518.75	0.001927711
13	3	8184	0.6109	64	129.6875	0.007710843
14	4	16368	0.3055	256	32.421875	0.030843373
15	5	32736	0.1527	1024	8.10546875	0.123373494
16	6	65472	0.0764	4096	2.026367188	0.493493976

Както се вижда от таблица 1, разликата между 10 и 16 битово преобразуване по време е около половин секунда, докато чувствителността на измерваната величина се подобрява от около 5 mV за 10 бита до около 0,1 mVза 16 бита. На тази база при процеси, измерваните параметри на които не се променят с голяма скорост във времето, този метод може да се прилага успешно, още повече в [3] е предвидена и процедура за елементарна статистическа обработка със средноаритметична стойност за времето на обработка на сигнала.

4. Тестови процедури

Методиката е тествана за измерване на един канал на АЦП със стабилизирано напрежение 1127 mV +/- 0,5 mV при различни разрядности от 10 до 16 бита и различен брой обработени показания, 1, 10 и 25, като за обработено напрежение е прието средното аритметично. В таблици 2, 3 и 4 са представени данните от измерванията при различните условия за виртуалната разрядност и броя на показанията, използвани за получаване на средноаритметичната стойност, колона 3.

Таблица 2. Стойности при едно показание

Разрядност bit	Брой показания	Напрежение, V
10	1	1.12903225
11	1	1.12903225
12	1	1.12781035
13	1	1.12781035
14	1	1.12750482
15	1	1.12735211
16	1	1.12742853

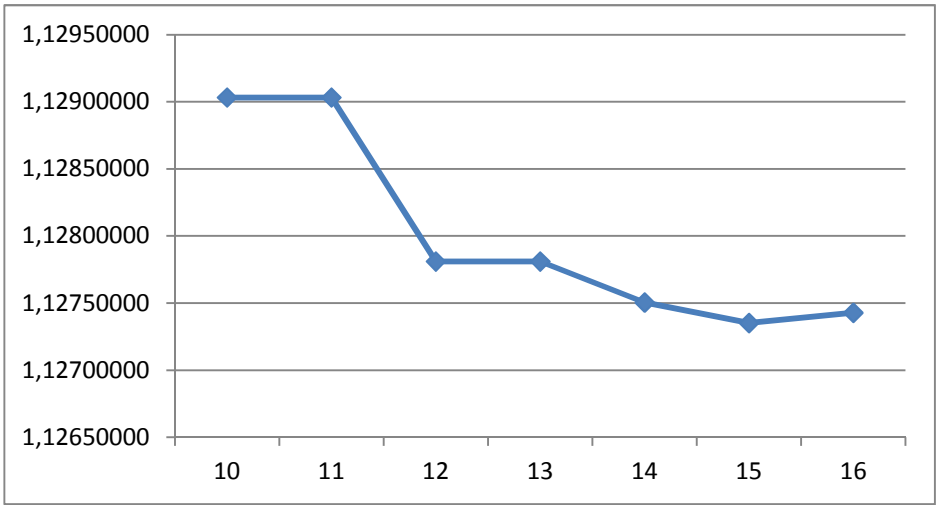
Таблица 3. Стойности при 10 показания

Разрядност bit	Брой показания	Напрежение, V
10	10	1.12756597
11	10	1.12878787
12	10	1.12866568
13	10	1.12756597
14	10	1.12726044
15	10	1.12732160
16	10	1.12730634

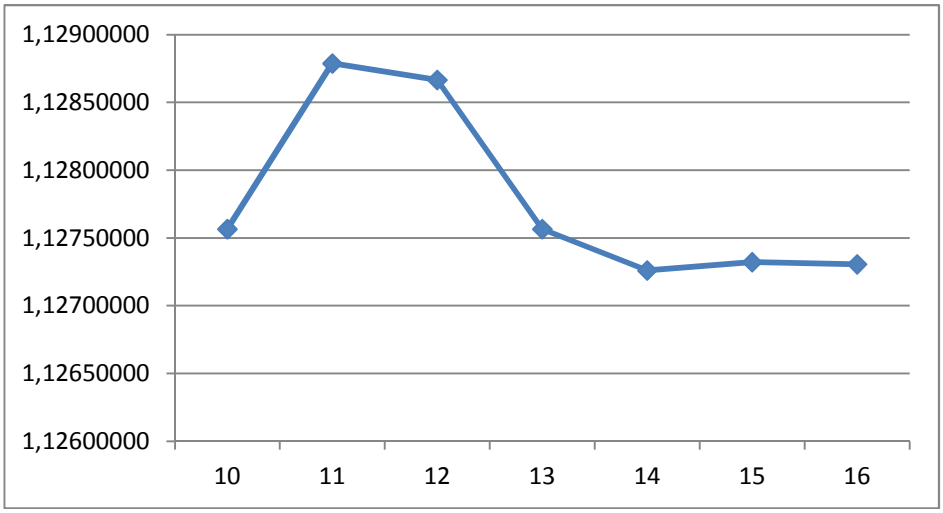
Таблица 4. Стойности при 25 показания

Разрядност bit	Брой показания	Напрежение, V
10	25	1.12883675
11	25	1.12932550
12	25	1.12834799
13	25	1.12781035
14	25	1.12755370
15	25	1.12730944
16	25	1.12741625

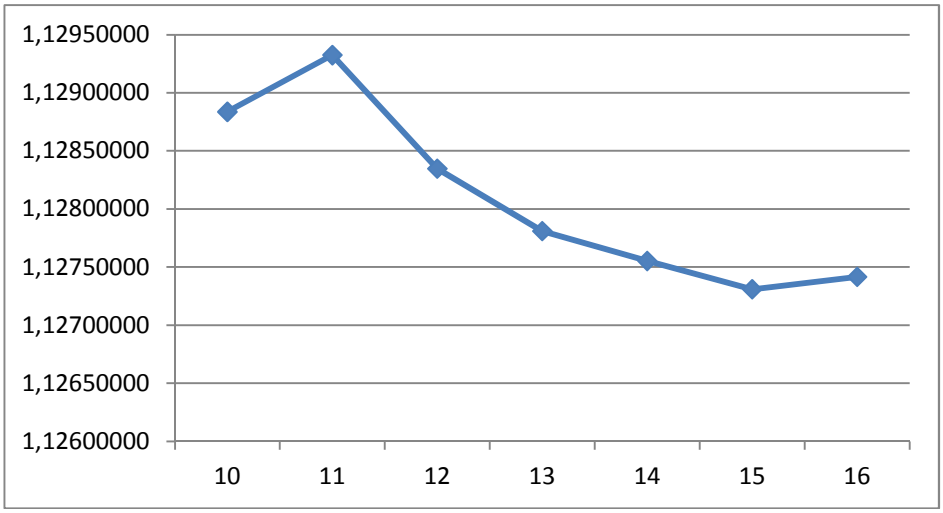
Резултатите са представени в графичен вид на фигури 4а, б, в. На абсцисната ос е разрядността, а на ординатната полученото напрежение от аналогоцифровото преобразуване.



Фиг. 4а. Резултати от аналогоцифровото преобразуване с едно показание



Фиг. 4б. Резултати от аналогоцифровото преобразуване с 10 показания



Фиг. 4в. Резултати от аналогоцифровото преобразуване с 25 показания.

5. Изводи

От тестовите примери се вижда, че максимална точност се постига при 14, 15 и 16 битово преобразуване. Това показва, че методът, предложен от ATMEL, може да се прилага за измерване на аналогови величини при процеси със сравнително ниски честоти, тъй като точността се повишава за сметка на скоростта на преобразуване на аналоговите сигнали.

Друго предимство е възможността за процесорна статистическа обработка на данните от типа на цитираните тестови случаи по-горе. Така могат да бъдат компенсирани апаратни грешки и случайни флуктуации, предизвикани от електромагнитни полета, към които аналоговите сензори са чувствителни.

Литература

[1]. High-Speed CMOS Logic 16-Channel Analog Multiplexer/Demultiplexer. CD74HC4067, CD74HCT4067, Texas Instruments, July 2003.

[2]. AVR121: Enhancing ADC resolution by oversampling. Atmel Coporation, 2005.

[3]. Staples G., Using the Arduino Uno's built-in 10-bit to 21-bit ADC (Analog to Digital Converter), May 2014.

[4]. Geeetech CD74HC4067 Analog/Digital MUX Breakout compatible with Arduino/iduino.

За контакти:
доц. д-р Илия Иванов Хаджидимов,
катедра „Топлотехника",
Технически университет – Варна,
e-mail: i_hadzhidimov@tu-varna.bg.

РЕТРОСПЕКТИВНА ИДЕНТИФИКАЦИЯ НА КОРАБЕН ДВИГАТЕЛ

Иван Е. Иванов, Ивайло Д. Бакалов, Димитър Г. Генов

Резюме: Разглежда се идентификация на динамичното поведение на дизелов корабен двигател Д 3900А. Чрез математическо моделиране се изследва влиянието на обикновено дизелово гориво и биодизел В100 върху динамиката на обекта. Идентификацията се извършва с прилагане на графоаналитични методи в средата на MATLAB. Въвеждат се два критерия за оценка на динамичните показатели на двигателя и влиянието на вида на горивото: максимална относителна грешка и интегрален критерий.
Ключови думи: идентификация, дизелов двигател, биодизел.

Retrospective Identification of Marine Diesel Engines

Ivan E. Ivanov, Ivaylo D. Bakalov, Dimitar G. Genov

Abstract: Identification and dynamic of the marine diesel engine D 3900 A. Through mathematical modeling to examine the impact of regular diesel fuel and B100 on the dynamics of the object. Identification is performed by application of graphic-analytical methods in the MATLAB. Introduces two criteria for evaluating the dynamic performance of the engine and the influence of the type of fuel: maximum relative error and integral criterion.
Keywords: Identification, diesel, biodiesel.

1. Увод

Теорията на идентификацията се прилага широко в изследователската практика при получаване на модели с цел прогнозиране на състоянията на обекти, при оптимизация или целенасочени изследвания. Особено ефективен този подход е при оценка на динамичното поведение на обекта на изследване за набиране на подходяща информация за неговото управление. Така математичният модел се явява единствено ефективно средство за предаване на целенасочена информация на управляващото устройство с цел неговото оптимално управление.

В практиката на корабостроенето математичният модел е полезно средство при синтез на предавателните функции на елементите на системата за адаптивно управление на курса на кораба.

Навлизането на биогоривата като добавка към основното гориво дава съществени предимства по отношение на качеството на горивните процеси и основно намаляване на CO2 в атмосферата. Затова усилията на много изследователи са насочени към оценка на положителните и отрицателни въздействия и ефекти от тази добавка.

2. Изложение

В [2] се разглеждат множество положителни моменти от прилагането на биодизел (Б100), отнасящи се до качеството на горивните процеси:

- биодизелът е с изключително ниско съдържание на сяра и е с висока мазителна способност и бърза биоразградимост;
- при В100 азотните окиси намаляват с около 10 %;

- липсата на сяра позволява прилагането на технологии, контролиращи азотните окиси, които не биха могли да се приложат при производството на конвенционалните горива;

- при изгарянето на биодизел се отделя приблизително толкова CO2 в атмосферата, колкото растенията биха приели по време на растежа си.

- при оптимизация на впръскването – биодизелут намалява емисиите и разхода на гориво [1].

В литературата липсват данни за изследвания на влиянието на B100 върху динамиката на дизеловите корабни двигатели.

Настоящата статия е посветена на едно изследване от прилагане на стандартното дизелово гориво и биодизел B100 върху динамиката на двигател Д 3900А, характеристиките на дизеловия двигател са дадени в таблица 1. Отсъствието на обективен критерий за оценка на това влияние се компенсира с въвеждането на оценка на максималната относителна грешка между реакциите на двигателя на двата типа гориво и интегрален критерий, формиран по подходящ начин. Връзка между хода и ъгълът на отклонение на рейката е дадена в таблица 2.

Таблица 1. Технически данни - дизелов двигател Д 3900А

№	Наименование	Димeнсия	Стойност
1.	2	3	4
2.	Диаметър на цилиндъра	mm	98,475
3.	Ход на буталото	mm	127
4.	Работен обем	dm^3	3.9
5.	Ред на работа на цилиндрите		1-3-4-2
6.	Номинална мощност по BSAU141a	kW	59
7.	Честота на въртене на коляновия вал при номинална мошност	min	2500
8.	Максимален въртящ момент	Nm	265
9.	Честота на въртене на коляновия вал при максимален въртящ момент	min	1500
10.	Регулировъчно налягане на дюзите	MPa	19.5
11.	Температура на охлаждащата течност	^0C	83÷95°
12.	Минимална честота на въртене на коляновия вал на празен ход	min	550÷650

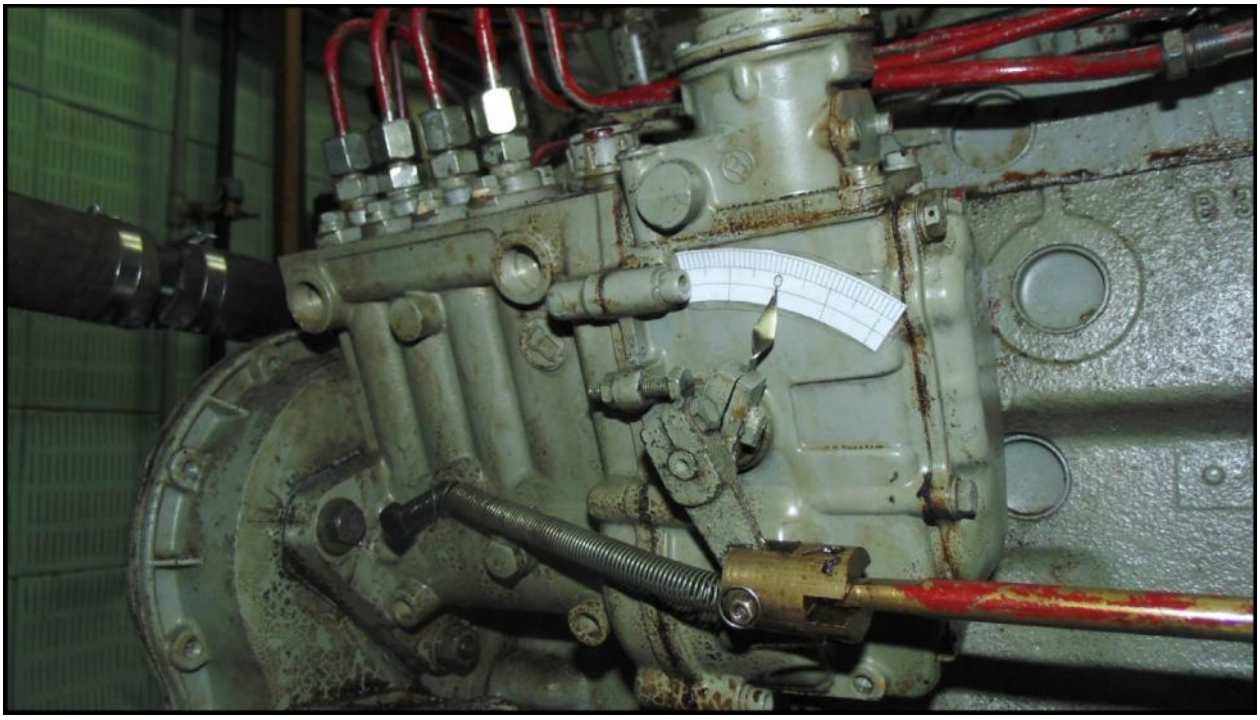
Таблица 2. Зависимост между хода и ъгъла на рейката на горивната апаратура

α	0	1^0	2^0	3^0
x	0	2,5 мм	5 мм	7,5 мм

Като технология на изследванията се прилага планиран експеримент по експериментално снемане на динамичните характеристики на обекта с последващо обработване на резултатите чрез методите на идентификация.

Набиране на информация за целите на идентификацията.

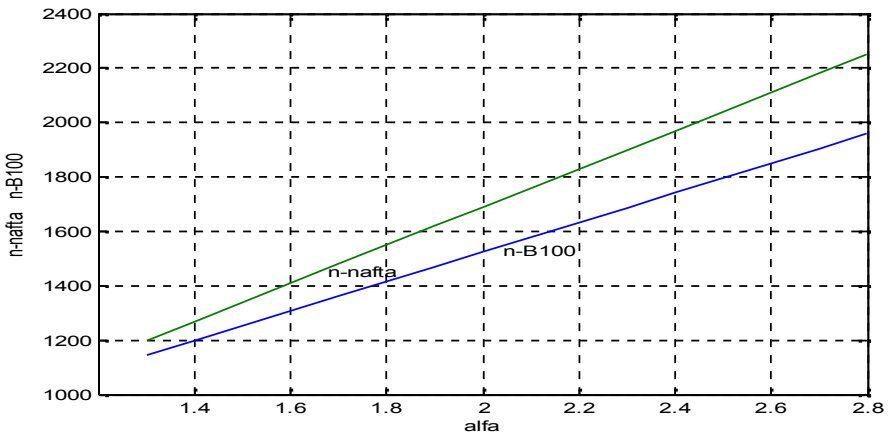
За целите на изследването интерес представлява изменението на честотата на въртене в преходен режим. Използвана е установката, дадена на следната фигура:



Фиг. 1.

2.1 Статични характеристики на дизелов двигател Д 3900А

Характерът на зависимостта на честотата на въртене от положението на рейката е даден на фигура 1. В работния диапазон 1200-2200 оборота/мин зависимостта е линейна. Този факт позволява при изследване на динамиката на двигателя да се избере смущаващо въздействие с максималма амплитуда, в случая от 1,3 до 2,7 grad. на рейката, фигура1.

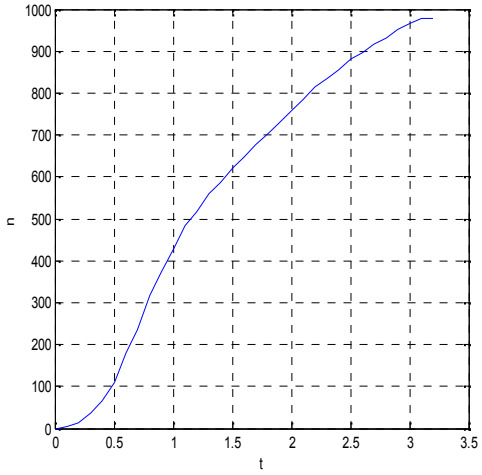


Фиг. 2.

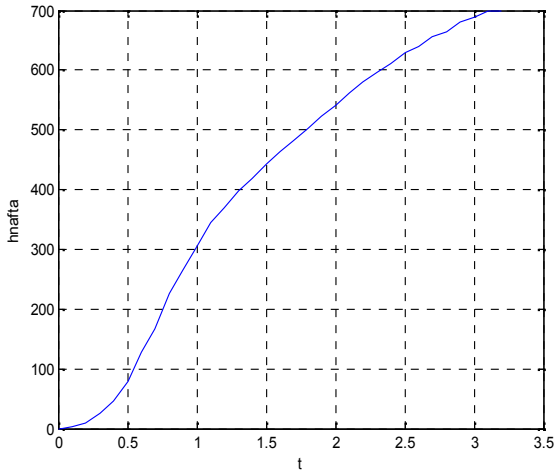
2.2 Идентификация на динамиката на двигател

2.2.1 Идентификация на динамиката на двигателя, работещ с традиционно дизелово гориво (нафта)

При избрана амплитуда на смущението от 1,4 grad е снета експериментално реакцията на двигателя, показана на фигура 3. На фигура 4 е дадена преходната характеристика, а нейната нормирана - на фигура 5.



Фиг. 3.



Фиг. 4.

От нормираната преходна функция са отчетени стойности за т. A(0,7;t_7), t_7=1.7 s. Съгласно алгоритъма на графоаналитичен метод, даден в [3], за t_4= t_7/3=0.57 s, и от фигура 5 е отчетена стойност 0,174 < 0,191, което е основание за избор на апроксимиращ модел от типа

$$T^2\frac{d^2n_n}{dt^2}+2T\frac{dn_n}{dt}+n_n=K_n\alpha(t-\tau) \qquad (1)$$

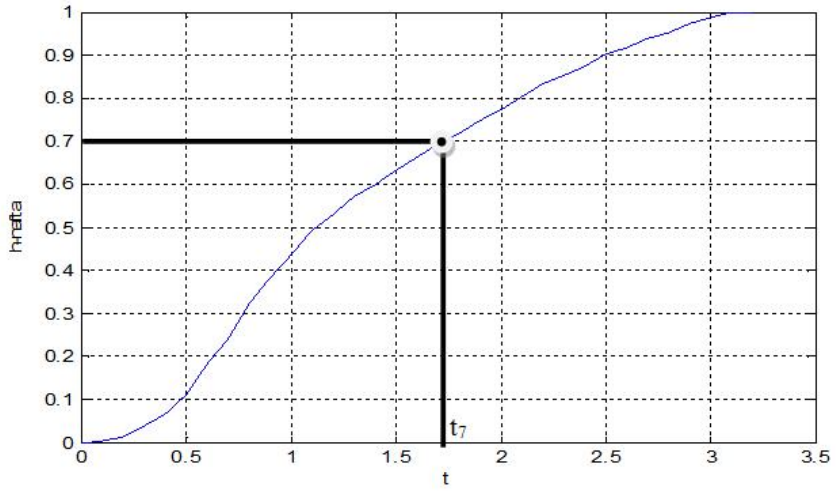
От нормираната преходна характеристика, фигура 5 за стойност 0.191 е отчетено t_4'' =0.6 s. По изразите: $\tau=0.5(3t_4''-t_7)$ =0.05 s $T=\frac{t_7-\tau}{2.4}$ = 0.69 са определени параметроте на модела (1), т.е.

$$0{,}47\frac{d^2n_n}{dt^2}+1{,}38\frac{dn_n}{dt}+n_n=699{,}35\alpha(t-0{,}05) \qquad (2)$$

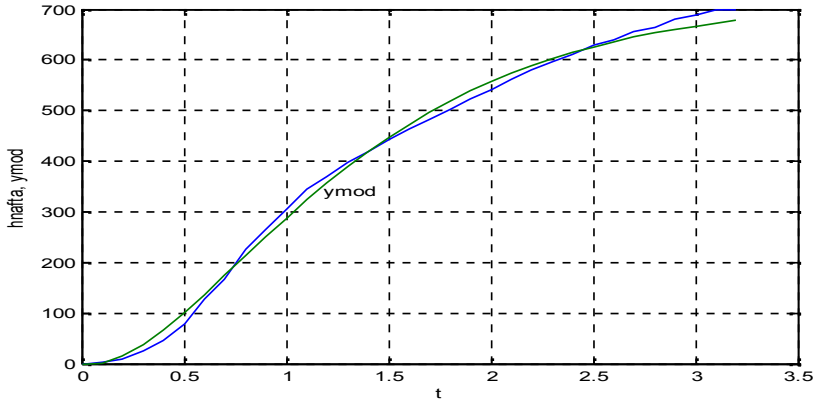
Неговото решение, при α =1 има вида

$$h_n(t)=699{,}35[(1-\exp(-\frac{t-0{,}05}{0{,}69})-\frac{t}{0{,}69}\exp(-\frac{t-0{,}05}{0{,}69})] \qquad (3)$$

На фигура 6 са показани преходната характеристика от фигура 4 и нейната апроксимация с модела (3). Моделът (2), апроксимира експерименталните данни с грешка под 4%.



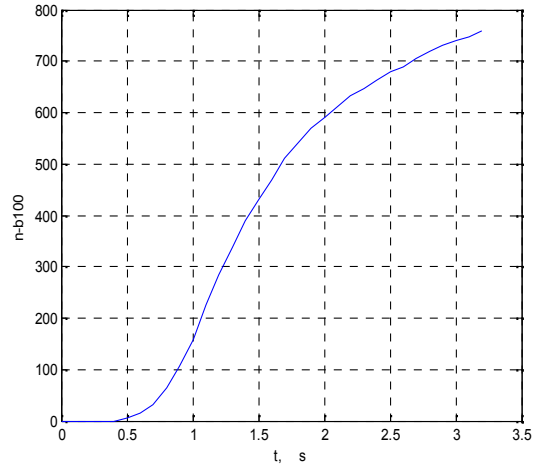
Фиг.5



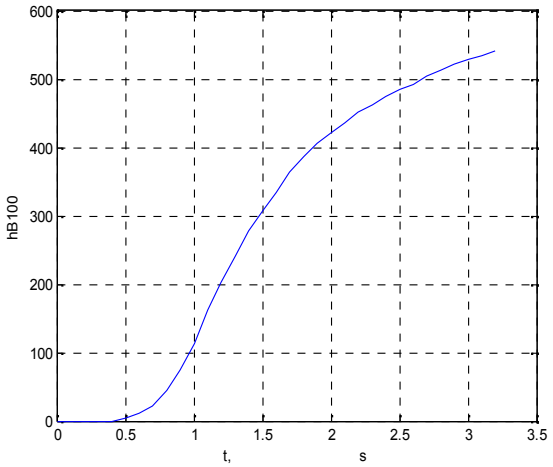
Фиг. 6

2.2.2 Идентификация на динамиката на двигателя, работещ с гориво B100

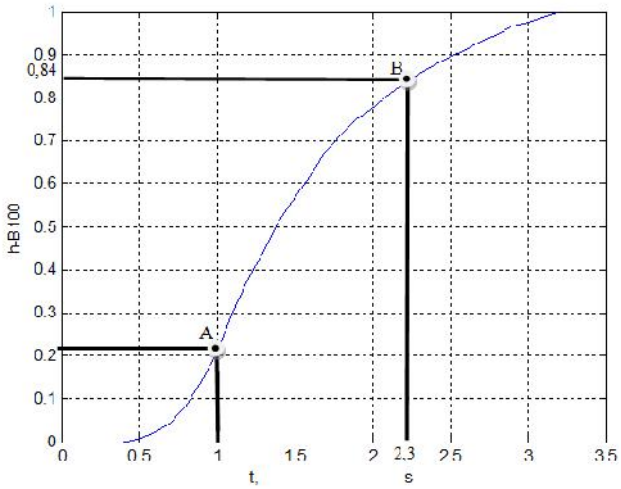
Условията при провеждане на експеримента с гориво B100 са аналогични на тези с обикновеното дизелово гориво. Снетата експериментално реакция на двигателя е показана на фигура 7. На фигура 8 е дадена преходната характеристика, а нейната нормирана - на фигура 9.



Фиг. 7



Фиг. 8



Фиг. 9

Поради специфичния характер на реакцията n_B100, се приема апроксимиращ модел от вида

$$T_B \frac{dn_B(t)}{dt} + n_B(t) = K_B \alpha(t-\tau) \quad (4)$$

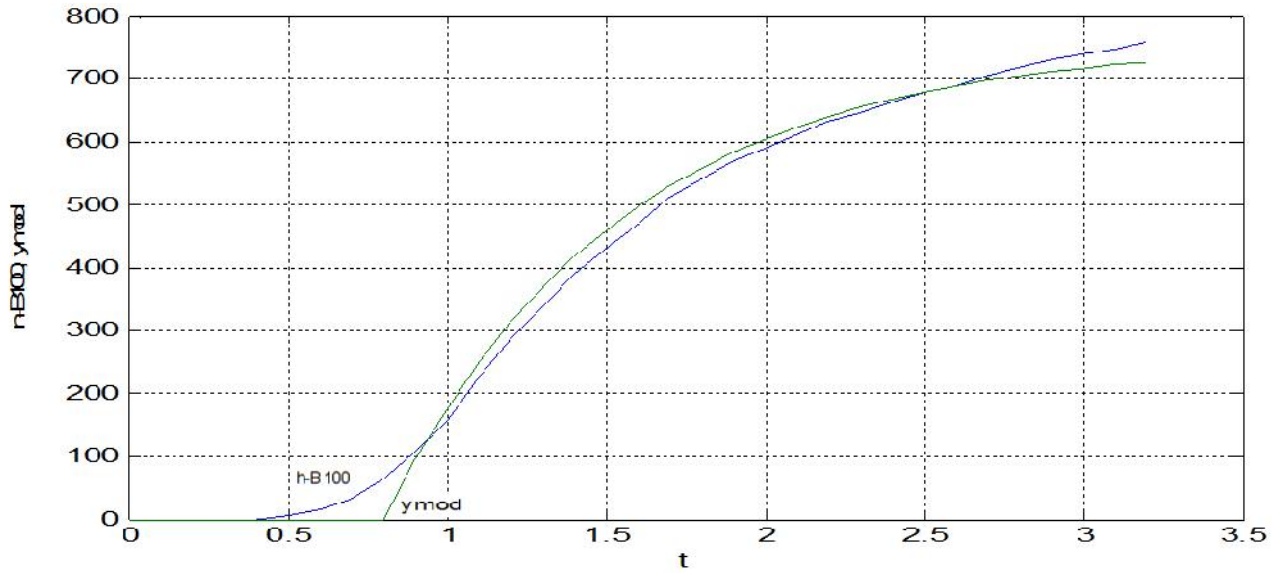
От нормираната преходна характеристика, фигура 9 по графичен метод, даден в [3], са определени т.A(t_A,h_A)=A(1;0.208) и т.B(t_B,h_B)=B(2,3;0,85), откъдето се пресмятат параметрите на модела $\tau = [t_B \ln(1-h_A) - t_A(1-h_B)]/[\ln(1-h_A) - \ln(1-h_B)] = 0{,}8$ s и $T_B = -(t_A - \tau)/\ln(1-h_A) = 0{,}77$ s. Моделът (4) добива вида

$$0{,}77 \frac{dn_B(t)}{dt} + n_B(t) = 541{,}4\alpha(t-0{,}8) \quad (5)$$

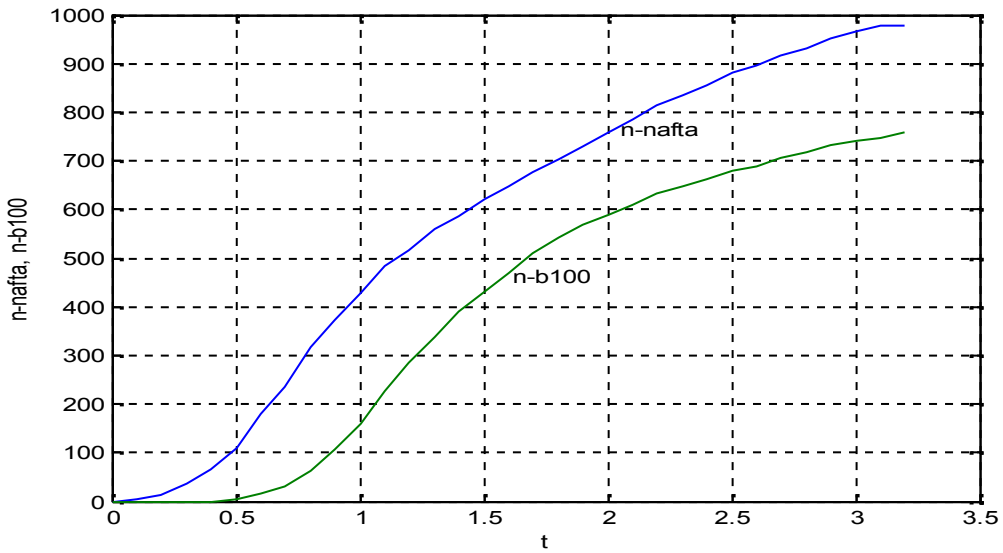
а преходната характеристика

$$h_{nB} = 541{,}4[1 - \exp(-\frac{t-0{,}8}{0{,}77})],\ t > 0{,}8s; \quad h_{nB} = 0,\ 0 < t < 0{,}8s.$$

На фигура 10 са дадени реакцията на обекта и нейната апроксимация с модел (5).



Фиг. 10



Фиг. 11

На фигура 11 са показани реакциите на обекта при двата вида гориво.

За обективиране влиянието на вида на горивото върху динамиката на процесите ще бъде въведена функция на грешката между двата процеса,

$$E(t)=n_n(t)- n_B(t) \quad (6)$$

Максималната относителна грешка е

$$E_{max} = \max[E(t)]/\max[n_n(t)].100=23,8\% \quad (7)$$

Като допълнение към E_{max} за по-пълна характеристика на влиянието на качеството на горивото ще бъде въведена интегралната грешка. За целта се въвеждат следните функции $S_E = \int_0^\infty E(t)dt$; $Sn_n = \int_0^\infty n_n(t)dt$; $S_{nB} = \int_0^\infty n_B(t)dt$. Числените стойности за конкретният случай са: S_E=578,24; Sn_n=1883,4; Sn_B=1305,2.

Относителната интегрална грешка при стандартното гориво е E_{E-n} =578,24/1883,4.100=30,7% а при B100 E_{E-nB} = 578,24/1305,2.100=44,3%.

3. Изводи

1) Динамиката на дизеловият двигател се различава съществено при използване на стандартно дизелово гориво (нафта) и B100.
2) При използване на B100 се наблюдава закъснение в реакцията на двигателя след подаване рязко на стъпаловидно смущение.
3) При подаване на еднакви смущаващи въздействия се наблюдават по-ниски стойности на честотата на въртене при установените режими на n-B100, фиг 11. В този случай относителната грешка е

Еуст=[$n_n(\infty)$-$n_{nB}(\infty)$]/$n_n(\infty)$=220,93/979,09.100=22,6%

Литература

[1]. Иванов И., И. Бакалов, Р. Стоянов, „Методи и средства за намаляване на емисиите на вредните газове на корабните дизелови двигатели.“ Международна конференция MATTEX 2012, 22 - 24 ноември 2012г., (под печат).

[2]. Стоянов, Р. И. Бакалов, „Изследване продуктите на горене на дизелов двигател при използване на дизелово гориво и биодизел”, Международна конференция „ЕКО Варна 2014” (под печат).

[3]. Генов, Д. „Идентификация на системи” , ТУ-Варна, 2007, 380 с.

За контакти:
доц. д-р инж. Иван Енчев Иванов
катедра „Корабни силови уредби“
ВВМУ „Никола Вапцаров” - Варна
E-mail: ivanenchev@abv.bg

ас. инж. Ивайло Данков Бакалов
катедра „Корабни силови уредби“
ВВМУ „Никола Вапцаров” - Варна
E-mail: bakalov_vvmu@abv.bg

доц. д-р инж. Димитър Генчев Генов
E-mail: dggenov43@gmail.com

ИДЕНТИФИКАЦИЯ ДИНАМИЧЕСКОГО ОБЪЕКТА РЕГРЕССИОННЫМ МЕТОДОВ В ВЕСОИЗМЕРИТЕЛЬНЫХ СИСТЕМАХ

Евгений Касьяненко, Валерий Ситников

Резюме: Рассмотрена возможность определения постоянной составляющей сигналов, полученных при измерении параметров динамических систем. Приведено аналитическое описание метода. Метод позволяет получить искомы значения для каждого конкретного сигнала. Предложена работоспособная модель и отражены полученные в процессе моделирования результаты.
Ключевые слова: динамическая система, регрессионный метод, дифференциальные уравнения, тензометрический датчик, фазовые переменные.

IDENTIFICATION OF DYNAMIC OBJECTS REGRESSION METHOD IN WEIGHING SYSTEMS

Yevhen Kasianenko, Valeriy Sytnikov

Summary: The possibility of determining the constant component of the signals obtained in the measurement of parameters of dynamic systems. The analytical description of the method. The method allows to obtain the desired value for each signal. Propose a workable model and received reflected in the simulation results.
Keywords: dynamical system, regression method, differential equations, strain gauge, the phase variables.

Введение

При взвешивании движущихся объектов на тензометрических весах проблема повышения точности взвешивания обусловлена динамическими процессами в весоизмерительной системе. Электромеханическая часть весоизмерительной системы состоит из движущегося обьекта, весовой платформы и тензометрических датчиков.

Сигналы, полученные с тензометрических датчиков, являются зашумленными или искаженными, что в большинстве случаев не позволяет получить результат с требуемой точностью [1].

Рассматривается возможность и различные аспекты регрессионного метода идентификации при динамическом взвешивании для повышения точности определения веса. Результатом использования регрессионного метода идентификации динамических систем является согласованная по параметрам и структуре модель весоизмерительной системы, позволяющая сформировать искомую информативную составляющую сигнала.

Задача идентификации ставится как нахождение таких оценок неизвестных параметров a_t, $i = 0,1,...m$, при которых заданная уравнением аналитическая зависимость будет наилучшим образом аппроксимировать экспериментальные данные [2].

Совокупность соотношений образует систему уравнений относительно оценок $m+1$ коэффициентов уравнения регрессии, решение которой определяет искомые коэффициенты. Для выбора полинома используются общие графические представления о характере экспериментальной зависимости, а также дополнительные косвенные соображения. Однако, универсальные методики обоснования выбора вида и порядка полинома m отсутствуют.

Моделирование регрессионного метода идентификации

Предложенный в работе метод идентификации составляющих сигналов динамических систем опробуем на электрической модели [3], поскольку электрические динамические системы доступны к моделированию, просты к пониманию специалистам электротехнических специальностей и весьма распространены в современной технике и быту.

Если проводить аналогию между электрической и механической системами, то электрическая емкость соответствует массе, индуктивность – жесткости, электрическое сопротивление – потерям на трение, напряжение – скорости, а электрический ток – механической силе. Аналогично заменяются и все остальные электрические величины механическими [4]. Также электрические законы и формулы по своей сути вполне соответствуют механическим, да и не только механическим, а повторяются во всех областях физики, в экономике, природе и т.д. Следовательно, методы применимые к моделям электрических систем, будут применимы и к системам других видов.

Исходя из практической задачи динамического взвешивания, в качестве исследуемой модели возьмем груз на пружинном подвесе, заменяемый RLC–цепочкой в электрической модели [5]. Источник тока будет имитировать силу тяжести, конденсатор – массу, индуктивность –жесткость пружины, резистор – трение. Для моделирования интерес представляют системы 2-го и 4-го порядков, так как груз на подвесе является системой 2-го порядка, а динамическое взвешивание катящихся объектов в грубом приближении – системой 4-го порядка. На рисунку 1 приведена RLC–модель 2-го порядка, построенная.

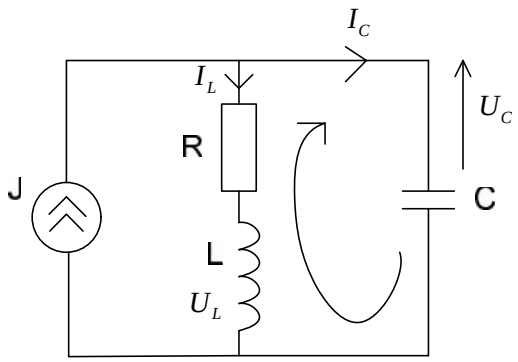


Рисунок 1. RLC- модель 2-го порядка

Для того чтобы применить регрессионный метод идентификации и определить все необходимые информативные составляющие и погрешность нужно проделать ряд преобразований. На рисунке 2 представлена структурная схема решения задачи.

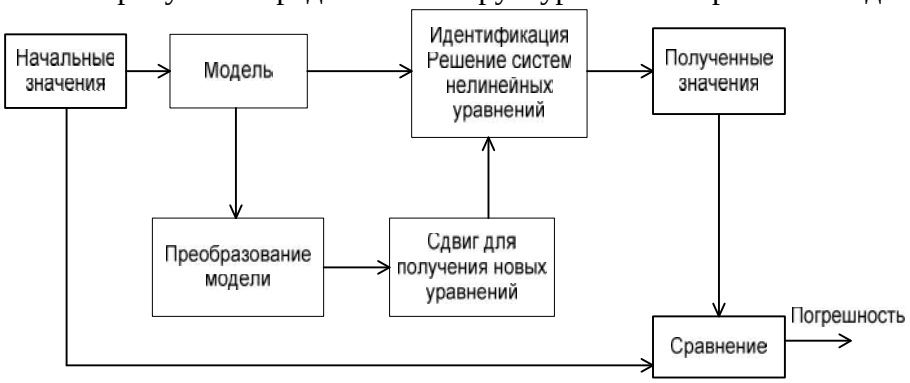


Рисунок 2. Структурная схема решения задачи идентификации

Составим систему уравнений по первому и второму закону Кирхгофа:

$$\begin{cases} J - I_L - I_C = 0 \\ U_R + U_L - U_c = 0 \end{cases} \quad (1)$$

Выразим из первого уравнения ток через конденсатор, а из второго – напряжение на катушке и запишем систему уравнений в дифференциальной форме для электрической цепи второго порядка:

$$\begin{cases} \dfrac{dU_C}{dt} = \dfrac{J}{C} - \dfrac{I_L}{C} \\ \dfrac{dI_L}{dt} = \dfrac{U_C}{L} - \dfrac{RI_L}{L} \end{cases} \quad (2)$$

Перейдем от аналоговой формы записи к цифровой и для того чтобы применить итерационные методы нужно систему дифференциальных уравнений преобразовать к системе разностных уравнений :

$$\begin{cases} U_{C,i+1} = U_{C,i} + h\dfrac{J}{C} - h\dfrac{I_{L,i}}{C} \\ I_{L,i+1} = I_{L,i} + h\dfrac{U_{C,i}}{L} - h\dfrac{RI_{L,i}}{L} \end{cases} \quad (3)$$

Для цепи второго порядка максимальная величина шага интегрирования ограничена величиной $h \le 2\tau_{\min}$, где τ - минимальная постоянная времени цепи. Так как цепь устойчива ($\alpha > 0$), то выбераем $\lambda 1$ и тогда $h \le 0.463$ с.

После моделирования можем посмотреть зависимости тока и напряжения от времени.

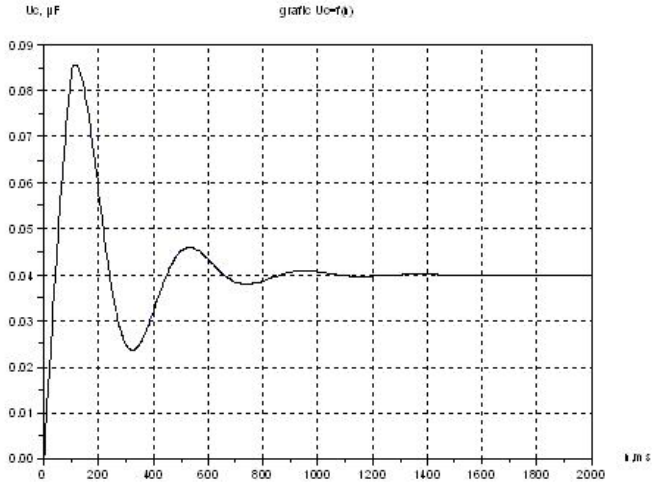


Рисунок 3. График зависимости напряжения на конденсаторе времени для шага 0,001

Для того чтобы решить задачу идентификации нужно преобразовать систему дифференциальных уравнений с целью исключения лишних фазовых переменных. Для этого выразим из второго уравнения формулы 3 напряжение на конденсаторе в момент времени i.

$$U_{C.i} = \frac{L}{h} I_{L.i+1} - h\frac{I_{L.i}}{L} + RI_{L,i} \quad (4)$$

Далее произведем сдвиг во времени в формуле 4.

$$U_{C,i+1} = \frac{L}{h} I_{L,i+2} - h\frac{I_{L,i+1}}{L} + RI_{L,i+1} \quad (5)$$

Теперь подставим формулу 4 и 5 в формулу 3 вместо соответственно напряжения на конденсаторе в момент времени i и i+1. Произведем все необходимые преобразования и выразим ток через катушку в момент времени i+2.

Так как решение задачи сводиться к решению системы нелинейных уравнений с 4-мя неизвестными (C, R, L, J), то необходима система аналогичного порядка :

$$\begin{cases} I_{L,i+2} = \dfrac{-CL + CRh - h^2}{CL} I_{L,i} + \dfrac{2L - Rh}{L} I_{L,i+1} + \dfrac{h^2 J}{CL} \\ I_{L,i+3} = \dfrac{-CL + CRh - h^2}{CL} I_{L,i+1} + \dfrac{2L - Rh}{L} I_{L,i+2} + \dfrac{h^2 J}{CL} \\ I_{L,i+4} = \dfrac{-CL + CRh - h^2}{CL} I_{L,i+2} + \dfrac{2L - Rh}{L} I_{L,i+3} + \dfrac{h^2 J}{CL} \\ I_{L,i+5} = \dfrac{-CL + CRh - h^2}{CL} I_{L,i+3} + \dfrac{2L - Rh}{L} I_{L,i+4} + \dfrac{h^2 J}{CL} \end{cases} \quad (6)$$

Шаг интегрирования следует выбрать значительно больше чем при наблюдаемой величины. Так как при решении первой задачи шаг интегрирования был выбран 0,001, то при идентификации неизвестных возьмем 0,005 и 0,01. После решения системы уравнений (6) и нахождения всех неизвестных (C, R, L, J) нужно сравнить полученные коефициенты с заданными первоначально.

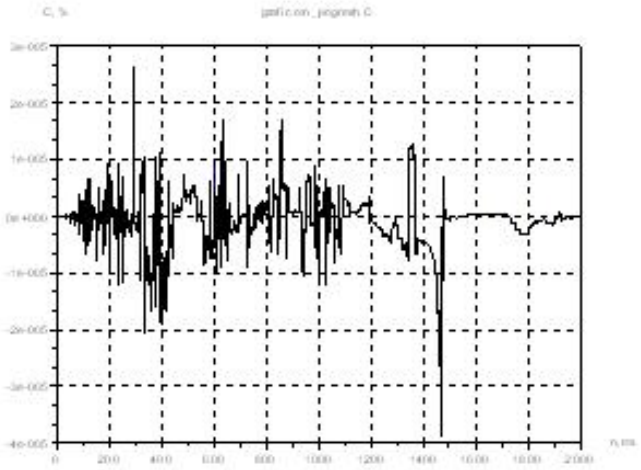


Рисунок 4. График относительной значения C между первым и вторым случаем 0,005

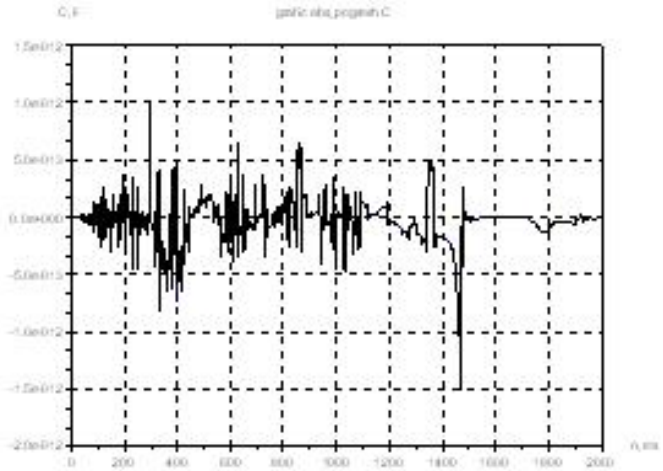


Рисунок 5. График и абсолютной погрешности значения C между первым и вторым случаем 0,005

Исходя из полученных значений параметров системы во второй части и сравнив их с ранее заданными, можем сделать вывод, что метод идентификации сработал правильно, погрешности параметров удовлетворяют требованиям.

Заключение

Данный метод может применяться при определении параметров динамических систем. Как правило, определяемые параметры на некотором промежутке времени являются постоянными, но измерить их точно не удается из-за помех различного рода, вызванных например колебаниями всего объекта или его частей. Решить данную задачу можно измерив на определенном промежутке времени искомый параметр, а затем полученную последовательность данных обработать предложенным методом. Такие задачи, как уже было сказано, возникают при измерении веса: движущихся объектов, на движущихся объектах, цистерн с колеблющейся жидкостью, подвешенных грузов и т.д.

Литература

[1]. Солонина А.И., Основы цифровой обработки сигналов: Курс лекций/ А.И. Солонина, Д.А. Улахович, С.М. Арбузов и др. – СПб.: БХВ-Петербург, 2003.– 615 с.

[2]. Сергиенко А.Б., Цифровая обработка сигналов: Учебник для вузов – СПб.: Питер, 2002.– 608 с.

[3] Уйдров Б., Стирнз С., Адаптивная обработка сигналов: пер. с англ. – М.: Радио и связь, 1989. – 440 с.

[4]. Сергиенко А.Б., Алгоритмы адаптивной фильтрации: особенности реализации в MATLAB. //Математика в приложениях. – 2003. – №1(1). – С.18-28.

[5]. Айфичер Э., Джервис Б., Цифровая обработка сигналов: Практический подход, 2-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2008. – 992 с.

For contacts:
Yevhen Kasianenko
Institute of Computer Systems, Department of Computer Systems
Odessa National Polytechnic University
E-mail: Leon.mail.ru@mail.ru
Doctor of Science (Eng.), prof., Valeriy, Sytnikov, Head of Computer System Department
Institute of Computer Systems, Department of Computer Systems
Odessa National Polytechnic University
E-mail: sitnvs@mail.ru

АНАЛИЗ УСТОЙЧИВОСТИ ЦИФРОВОГО ФИЛЬТРА ТРАКТА ОБРАБОТКИ ПРИ ПЕРЕСТРОЙКЕ ЕГО ПАРАМЕТРОВ

Анна Ухина, Валерий Ситников

Резюме: Существуют вычислительные задачи, граф вычисления которых не помещается в память. Также существует задача наилучшего разбиения для конкретной задачи. В работе представлен алгоритм разбиения графа на подграфы, основанный на существующий алгоритмах, а так же структура которая позволяет увеличить скорость работы с частями графа.
Ключевые слова: разбиение графа на подграфы, алгоритмы разбиения графа.

DIGITAL FILTER STABILITY ANALYSIS PROCESSING PATHS DURING ITS PARAMETERS ADJUSTMENT

Hanna Ukhina, Valeriy Sytnikov

Summary: There are computational problems that computation graph cannot fit in memory. There is also a problem of the best partition for a specific task. The article presents an algorithm for graph partitioning into sub-graphs, based on existing algorithms, as well as the structure, which allows increasing the speed of the graph parts.
Keywords: decomposition of a graph into sub-graphs, graph-partitioning algorithms.

1. Введение

Автономные мобильные платформы все активнее входят в нашу повседневную жизнь. Это и варианты беспилотных летательных аппаратов, наземных платформ, а также надводных и подводных аппаратов. В большинстве случаев это сложные специализированные компьютерные системы, которые имеют специфические цели и задачи, связанные с управлением движения, связи с координирующим сервером, сбор, предварительная обработка и передача информации с датчиков.

При создании таких автономных мобильных платформ возникает задача перестройки работы трактов обработки информации при изменении условий работы или при доопределении каких-либо параметров. Например, в квадрокоптерах имеются системы определения безопасной высоты, а у наземных платформ – устройства определения расстояния до препятствия.

В этих условиях компоненты, входящие в тракт обработки, должны иметь возможность перестраивать свои параметры (в узком или широком диапазонах) без потери устойчивости.

2. Описание тракта обработки и анализ устойчивости фильтра

Тракт первичной обработки измерительной информации обычно имеет цифровой фильтр для выделения информационного сигнала на фоне помех. Однако, например, акустический зондирующий импульс может изменять свою частоту излучения для доопределения расстояния до препятствия. В этом случае цифровой фильтр должен отслеживать основную частоту излучения и перестраивать свои параметры для наилучшей фильтрации помех.

Автономные платформы имеют ограничения на энергоресурсы, что приводит к необходимости ограничивать разрядность представления коэффициентов фильтра, а соответственно разрядность аппаратной части.

В связи с тем, что реализация фильтров высокого порядка строиться обычно на фильтрах первого и второго порядков [1]. Поэтому рассмотрим устойчивость цифрового фильтра при перестройке его параметров на примере фильтра второго порядка вида

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}}. \quad (1)$$

где a_0, a_1, a_2 – действительные коэффициенты числителя; b_1, b_2 – действительные коэффициенты знаменателя.

Для фильтрации отраженного сигнала на фоне помех используется полосовой фильтр. Особенность этого фильтра состоит в необходимости перестройки его центральной частоты $\overline{\omega_0}$, а возможно и полосы пропускания Δ в соответствии с частотой излучения передатчика.

Устойчивость цифрового фильтра второго порядка описывается треугольником устойчивости в координатах коэффициентов знаменателя передаточной функции b1 и b2 [2]. Для полосового и режекторного фильтров треугольник устойчивости приведен на рис. 1. Кружочками показаны соответствующие центральные частоты, которые увеличиваются с низу в верх, изменяясь от 0 до 1.

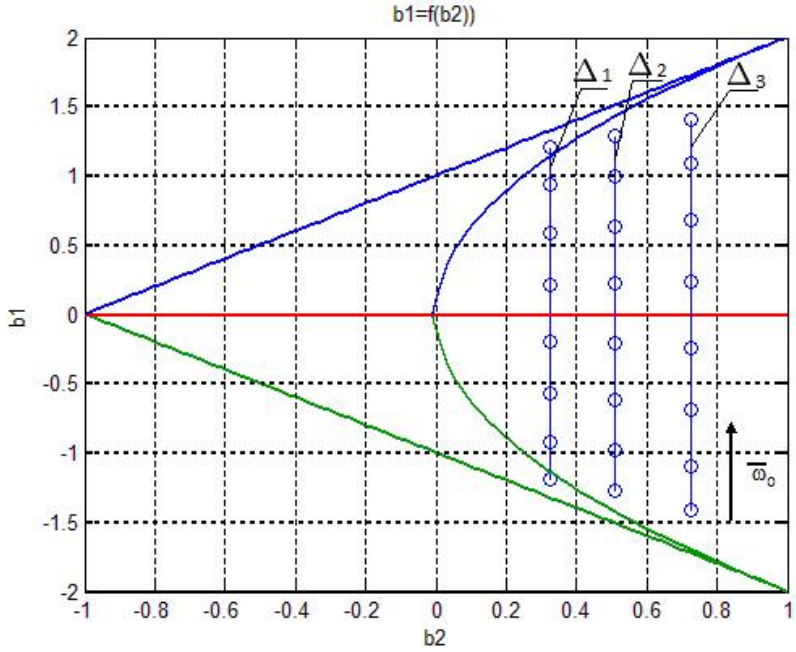


Рис. 1. Треугольник устойчивости для цифровых полосового и режекторного фильтров Баттерворта второго порядка, где полоса пропускания Δ1 >Δ2>Δ3

Анализ устойчивости полосового и режекторного фильтров Баттерворта второго порядка показал, что:

1. При уменьшении полосы пропускания Δ коэффициент знаменателя b2 смещается в сторону увеличения до 1. При увеличении полосы пропускания Δ коэффициент b2 стремится к 0. Поэтому при изменении полосы пропускания Δ возможен выход на границу устойчивости или выход за область устойчивости фильтра.

2. При изменении разрядности представления коэффициентов знаменателя фильтра исследования устойчивости показали, что коэффициент знаменателя b1 имеет больший разброс, чем коэффициент знаменателя b2. Чем меньше величина коэффициента b1 тем больше разброс, который может достигать 12% относительной погрешности.

Следует отметить, что включение режекторного фильтра в обратную связь позволяет получить соответствующий полосовой фильтр. С другой стороны полосовой фильтр можно получить за счет последовательного включения фильтров нижних и верхних частот.

Аналогичный анализ для фильтров Баттерворта нижних и верхних частот, рис.2., показал, что:

1. При увеличении частоты среза $\overline{\omega}_c$ коэффициент знаменателя b1 меняется от -2 до 2.

2. При изменении разрядности представления коэффициентов знаменателя фильтра исследования показали, что коэффициент знаменателя b1 имеет больший разброс, чем коэффициент знаменателя b2. Чем меньше величина коэффициента b1 тем больше разброс, который может достигать 12% относительной погрешности.

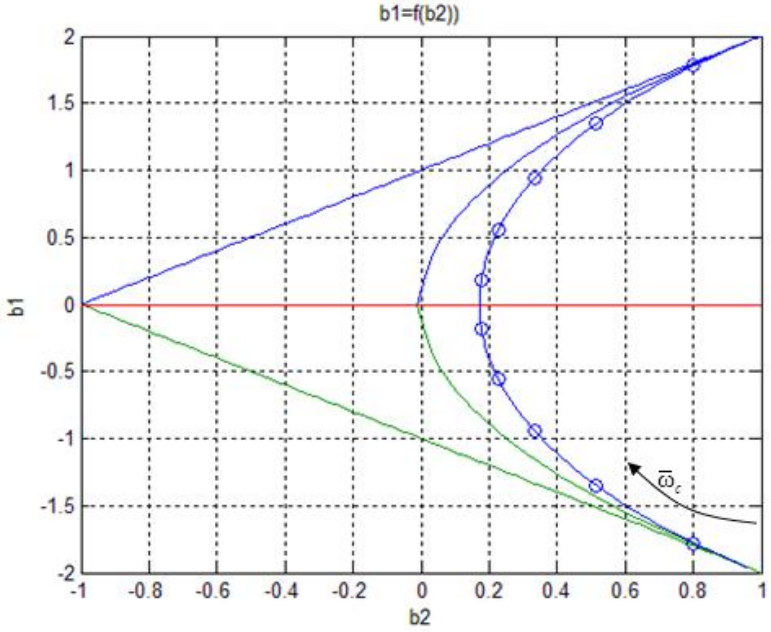


Рис. 2. Треугольник устойчивости для цифровых фильтров Баттерворта второго порядка нижних и верхних частот, где ω_c – относительная частота среза фильтра

Таким образом, исследования устойчивости показали, что при изменении разрядности представления коэффициентов знаменателя фильтра возможен выход за пределы устойчивости, что обуславливает необходимость проверки устойчивости фильтра при заданной разрядности представления коэффициентов и заданном диапазоне перестройки параметров на этапе разработки фильтра.

Литература

[1] Сергиенко А.Б., Цифровая обработка сигналов: 2-е изд. – СПб: Питер, 2005. – 751 с.

[2] Ситников В.С., Ступень П.В., Проверка устойчивости цифровых фильтров – Труды Одесского политехнического университета. – Одесса, 1999. – Вып.1 – С.160-162.

For contacts:
Bachelor, Hanna, Ukhina
Institute of Computer Systems, Department of Computer Systems
Odessa National Polytechnic University
E-mail: anyuta.uhina@inbox.ru
Doctor of Science (Eng.), prof., Valeriy, Sytnikov, Head of Computer System Department
Institute of Computer Systems, Department of Computer Systems
Odessa National Polytechnic University
E-mail: sitnvs@mail.ru

РАЗБИЕНИЕ ГРАФА НА ПОДГРАФЫ В УСЛОВИЯХ ОГРАНИЧЕНОГО ОБЪЕМА ОПЕРАТИВНОЙ ПАМЯТИ

Тихон Ситников, Анатолий Биленко

Резюме: Рассмотрена задача разбиения графа вычислений большим количеством вершин на подграфы в условиях ограниченного объема оперативной памяти. Что позволило увеличить быстродействие вычислений за счет рационального соотношения между частями графа вычислений между оперативной памятью и частями хранящимися на жестком диске. В работе также представлен алгоритм разбиения графа на подграфы, основанный на сочетании известных алгоритмах, а так же структура которая позволяет увеличить скорость работы с частями графа.
Ключевые слова: граф вычисления с большим количеством вершин, разбиение графа на подграфы, алгоритмы разбиения графа.

GRAPH PARTITIONING IN SYSTEMS WITH LIMITED AMOUNT OF RAM

Tykhon Sytnikov, Anatoly Bilenko

Abstract: Graph partitioning task for graphs with big number of vertexes, performed in systems with limited amount of RAM is being reviewed. The solution increased partitioning performance by means of optimal partition size and ratio between in-memory and on-disk partitions. Graph partitioning algorithm based on well-known algorithms and modified graph data structures, allowing to increase partitioning performance, is presented.
Keywords: graph of computation with big number of vertexes, graph partitioning algorithms.

1. Введение

Разбиения графа обычно рассматривается при задачах, таких как раскраска графа, определение числа и компонентов связанного графа, проектирование топологии локальной сети, трассировка соединений на печатной плате, разбиение задачи на блоки для вычисления на многопроцессорной системе и т.д. В перечисленных задачах встречаются графы с различными характеристиками и размерами. При большом количестве вершин графа вычислений возникает ситуация, когда в оперативную память вычислительной машины можно поместить только часть данных, которые отображают часть вершин графа вычисления.

Поэтому необходимо разработать алгоритм, который управляет процессом хранения частей на внешнем носителе и при необходимости подгружать следующие вершины для продолжения вычислений. Поскольку время обращения к данным на внешнем носителе на много больше, чем время обращения к оперативной памяти, то необходимо найти рациональное соотношение между повышением быстродействия алгоритма вычисления и обращением между внешним носителем и оперативной памятью.

2. Описание способа реализации

Для реализации алгоритма обращения к внешнему носителю необходимо выделить узкие места вычислительной системы. На рис. 1., показана структурная схема работы с частями графа вычисления, на которой показаны основные структурные элементы алгоритма работы программы разбиения. Из структурной схемы следует что Оптимизатор доступа и КЭШ чтения лежит в основе обмена данными между внешним носителем и оперативной

памятью. Это обусловлено тем, что скорость обработки запросов к внешнему носителю относительно мала и КЭШ предоставленный ОС работает не наилучшим образом.

КЭШ чтения в связке с оптимизатором доступа уменьшает количество обращений к внешнему носителю, на котором сохранены части графа в виде файлов. При этом уменьшается количества запросов к внешнему носителю, что позволяет ускорить подгрузку частей графа.

Некоторые алгоритмы разбиения используют дополнительную информацию, которая помогает быстрее выполнить качественное разбиение. Под качественным разбиением понимают такое разбиение, которое описано системой уравнений (1).

Для повышения скорости разбиения предлагается использовать метаданные как дополнительную информацию о вершинах графа, которые вводятся в части на этапе «принудительное разбиение». Метаданные используются при дальнейшем обращении к вершинам, при извлечении с внешнего носителя.

$$\begin{cases} V = \bigcup_{i=1}^{k} V_i \ , \ V_i \cap V_j = \varnothing, \forall i, j, i \neq j \\ \mu_j = \sum_{v_i \in V_j} w_v(v_i) \ , \ \mu_p - \mu_q \to \min \\ p, q \in \{1,2...k\} \quad p \neq q \\ \mu_e = \sum_{e_{ij} \in E, v_i \in V_p, v_j \in V_q, q \neq p} w_e(e_{ij}) \to \min \end{cases} \tag{1}$$

Следует отметить, что учет параметров внешнего носителя данных, позволяет обеспечить быстродействие при проектировании алгоритма разбиения:

Random access time (RAt): ~20ms
IO operations per second (IOPS): 100
IO read/write (IORW): 20 Mb/s

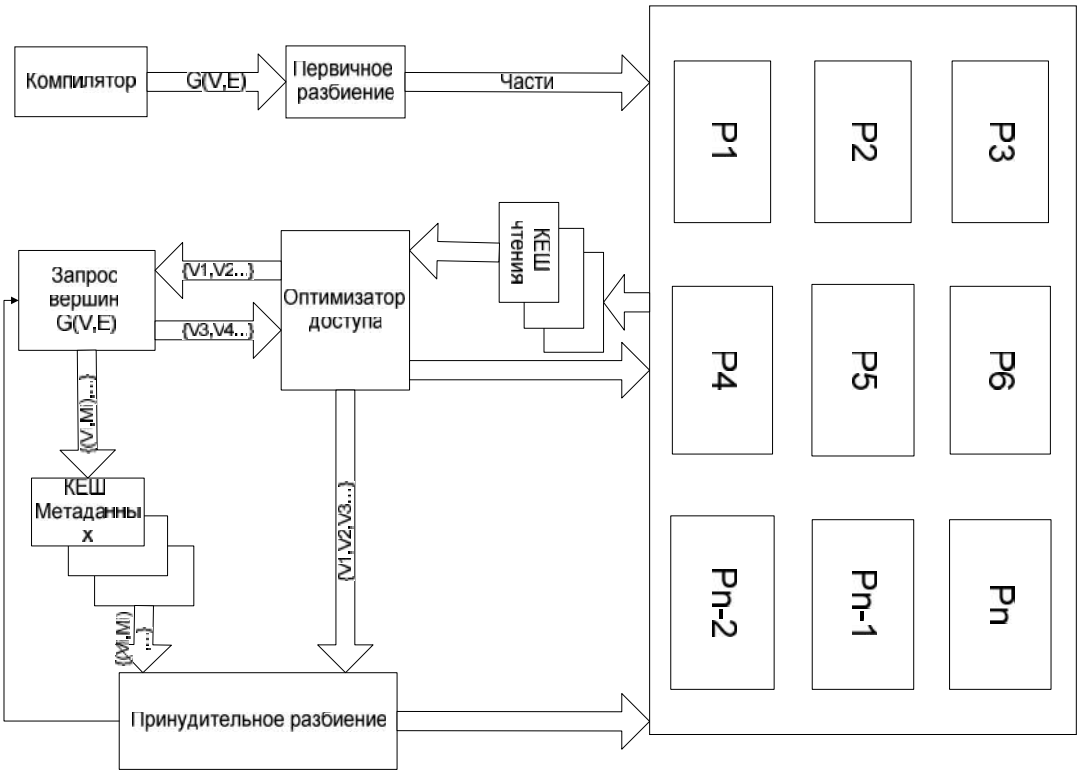


Рис. 1. Структура системы работы с частями графа

Введем предположение, что для разбиения графа нужно хранить в памяти всего Q частей на один программный поток. Количество Q связано с размерами частей S, так что Q * S ≈ Uram, где Uram - объем физической памяти, который доступен пользователю (минус память ОС и память сервисов ОС). Это предположение следует из рассмотрения графа в ярусно-параллельной форме, при условии что все вершины с которыми работает программа в данный момент находятся в соседних ярусах графа. Примером может быть {Група1} и {Группы 3 и 4} на рис. 2.

Ведем ещё одно предположение. Пусть в граф, который передается алгоритму разбиения, полученный из абстрактного синтаксического дерева программы, путем удаления всех дубликатов поддеревьев из него, добавим к информации о вершине на этапе "Первичная разбивка" (рис. 1) информацию (поле структуры) о количестве вершин множества предков (predcessor-ов). Используя эту информацию можно выбрать часть как множество всех множеств предков нескольких близких вершин и размер части в "хранилище частей" (рис. 1) как iN (sizeof (GNi) * predcessor (GNi)), где GNi - текущая вершина графа, который обходится в конкретный момент.

Один из возможных алгоритмов разбиения графа, множество вершин которого нельзя полностью загрузить в память может состоять в следующем: Граф должен быть представленв виде списков смежности [2].

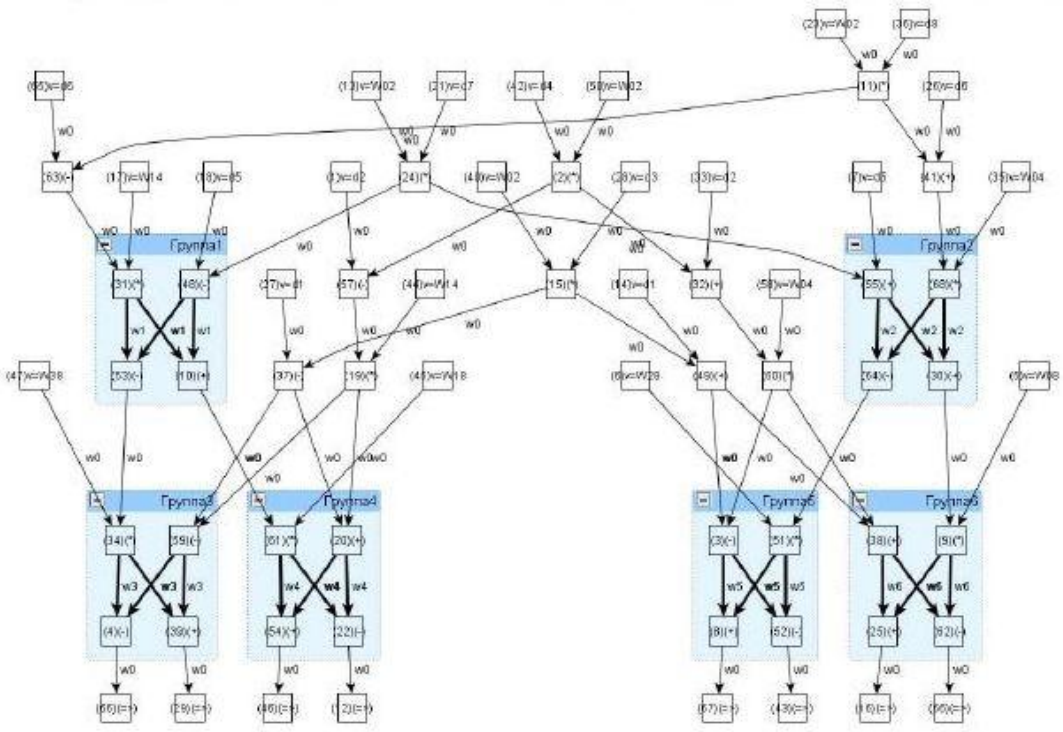


Рис 2. Подгружаемые вершины

1. Разместить каждый список смежности на внешнем хранителе, организовать MRU-кеш[2] для доступа к элементам этого списка.

2. Назначить каждому списку смежности и каждому элементу в нем уникальный идентификатор [3].

3. Если вершина графа находится в двух списках смежности i и j, назначить этой вершине идентификатор, равный $min(id(vi), id(vj))$, где id – это функция, которая возвращает идентификатор вершины.

4. Обойти все списки смежности и назначить им идентификатор равный $min(id(vi))$, где vi – вершины, которые принадлежат текущему списку смежности.

5. Воспользоваться известными алгоритмами разбиения графа на подграфы, например Кернигана - Лина, представляющих списки смежности с одинаковыми идентификаторами[1].

На рис. 1. представлена общая структура программы. В части, под названием «компилятор», создается абстрактное синтаксическое дерево (АСД). Необходимо отметить, что граф полученный из АСД предполагает, что граф не имеет циклов. Далее в следующем блоке, происходит первичное разбиение графа на части, пропорциональное размеру памяти на вычислительной системе. В нашем случае это размер примерно в Q раз меньше общего объема оперативной памяти. Такое соотношение обусловлено размерами физической оперативной памяти, которую потребляет ОС, для сохранения работоспособности системы и требованием программы разбиения.

Первичный алгоритм разбиения должен разбивать граф на равные части, рис. 1. Первичное разбиение можно упростить благодаря тому, что компилятор дополняет в каждую вершину графа количеством предшественников этой вершины. Это возможно сделать при условии того, что граф не имеет циклов. Добавление метаданных позволяет быстро находить часть требуемого объема [4].

Блок «Запрос вершин» (рис 1), реализует запрос и подгрузку вершин с жесткого диска. Это происходит благодаря блокам «Оптимизатор доступа» и блока «КЭШ метаданных». Блок «Оптимизатор доступа» оптимизирует запросы путем уменьшения обращения программы к жесткому диску. Частое обращение к диску негативно влияет на работу системы благодаря тому, что тратится время RAt на обращение к диску. Это обусловлено тем что позиционирование головки на считываемый/записываемый цилиндр. При таком запросе ОС использует системные запросы read и write (ssize_t write(int fd, const void *buf, size_t count)). Жесткий диск за одну секунду может обработать IOPS запросов на чтение или на запись, которые примерно соответствуют системным вызовам read и write. Уменьшение запросов происходит благодаря учету структуры графа и планированию этих запросов с учетом перечисленных параметров. Также отметим, что ограничивающим фактором при размере части является IORW. Необходимо учитывать этот параметр, так чтобы успеть запланировать загрузку необходимой части графа в память с учетом того, что можно прочитать только ограниченное количество данных в секунду (IORW).

При ответе на запрос, в LRU кэш память блока «КЭШ память» (рис 1) подгружается набор вершин с жесткого диска. Графически части для подгрузки изображены на рис 2.

Оптимизация заключается в том, что части графа, которые подгружаются с диска в кэш память, загружаются заранее. Планирование происходит с учетом вышеуказанных параметров. При такой работе обращение к части графа, которая находится на диске, исключается. Это ускоряет работу программы, благодаря уменьшению числа запросов к жесткому диску и оптимизации доступа к структурам данных.

Количество метаданных в блоке «КЭШ метаданных» пропорционально количеству вершин, которые находятся в КЭШ памяти.

Каждая вершина имеет уникальный идентификатор, если вершина находится в двух списках смежности, то этим вершинам назначается новый идентификатор, что позволяет упростить алгоритм принудительного разбиения в блоке «принудительное разбиение» [4].

По условию предложенной схемы в блоке «принудительное разбиение» (рис. 1) возможно использовать любой алгоритм разбиения на подграфы. Предложенная схема позволяет сохранять части графа на жестком диске в соответствии с вышеуказанными параметрами.

В результате использования алгоритма можно получить результат, представленный на рис. 3., в котором использовался сильносвязанный граф из 50 вершин, который разбит на 2 равные по весу части.

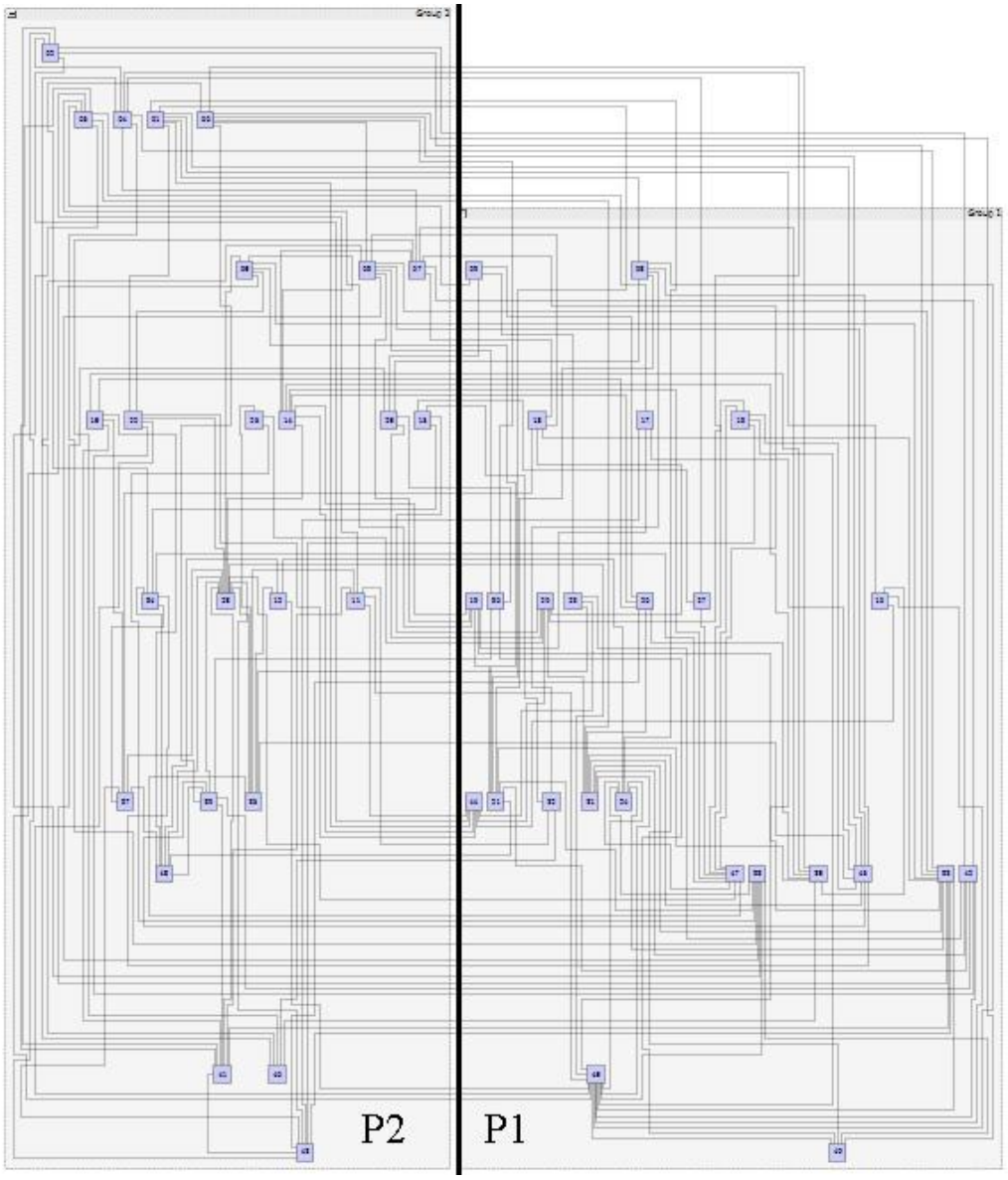


Рис. 3. Результат использования предлагаемой структуры

Выводы:

Обсуждаемая тема является востребованной и в общем виде не может быть решена на данный момент. Использование на практике предложеных структур и алгоритмов, может существенно повлиять на производительность вычислений и, в зависимости от ресурсов системы, дать ощутимый прирост быстродействия алгоритма разбиения графов. Использование методанных в свою очередь дает возможность ускорить процесс вторичного разбиения графа на части.

Литература

[1]. Bilenko A. Graph partitioning methods for computation in reconfigurable systems - Electrotechnic and computer systems. - No 05(81), 2012 -, pp 181-183

[2]. Cormen H., Leiserson E., Rivest L.,Stein C., Introduction to algorithms, second edition - September 1, 2001, The Mit Press, Cambrigde, Massachusetts London, England

[3]. Cache algorithms - http://en.wikipedia.org/wiki/Cache_algori%thms# Most_Recently_Used

[4]. Ситников Т., Биленко А. Разбиение графов на подграфы для реконфигурируемых систем в условиях ограниченного объема оперативной памяти. - System Analysis and Information Technologies 16-th International Conference SAIT 2014 Kyiv, Ukraine, May 26-30, 2014, pp. 445

For contacts:
Bachelor, Tykhon, Sytnikov
Institute of Computer Systems,
Department of Computer Systems
Odessa National Polytechnic University
E-mail: tykhon.sytnikov@yandex.ru

Ph(D), asst., Anatoly, Bilenko
Institute of Computer Systems,
Department of Computer Systems
Odessa National Polytechnic University
E-mail: b_@ukr.net

INVESTIGATION OF ELECTROMAGNETIC PHENOMENA PREDICTING EARTHQUAKE BY NEW FET SENSOR

Metin Saltik, Suzan C. Mustafa

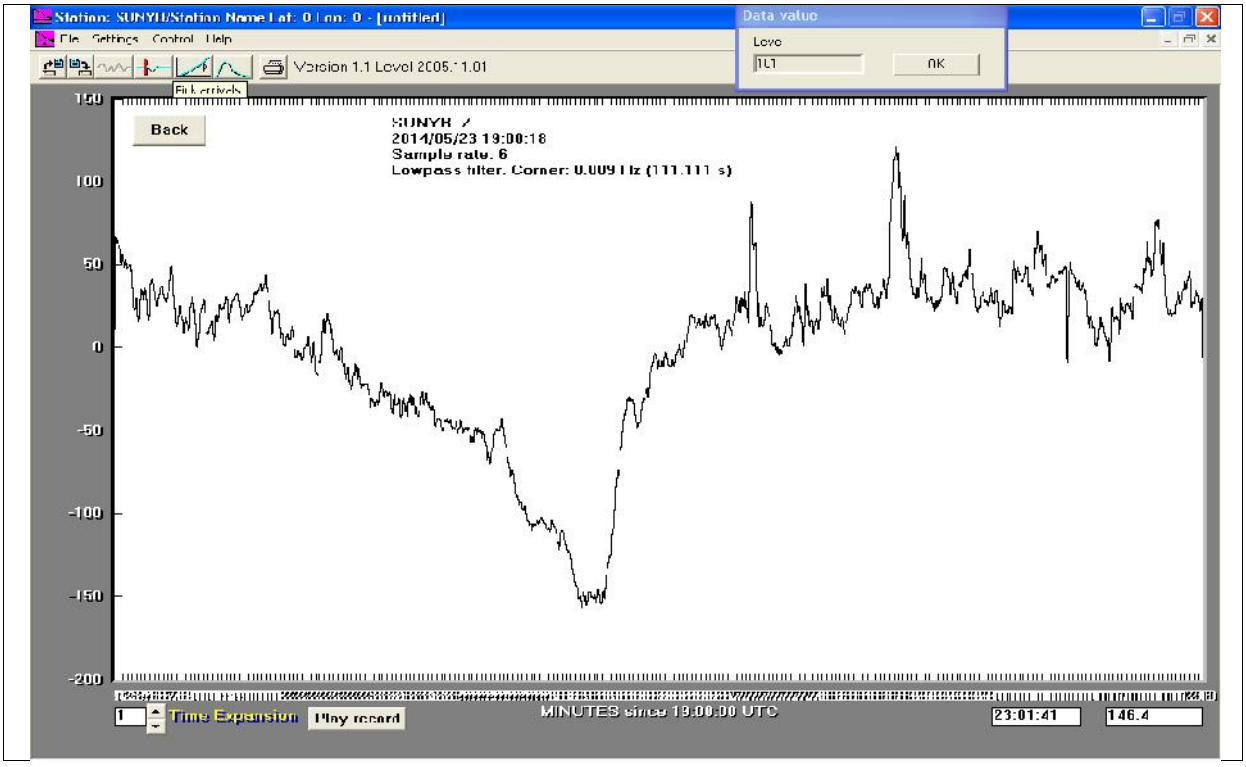
Abstract: Seismo-electromagnetics is the study of electromagnetic phenomena associated with seismic activity such as earthquake and volcanoes and also the use electromagnetic methods in seismology. It has been reported that electromagnetic phenomena take place in a wide frequency range prior to an earthquake.It has been observed that electromagnetic disturbances happen during the days that precede an earthquake. These disturbances happen when crystalline rocks are deformed by the slow grinding of the earth that occurs just before an earthquake. The cracking creates tremendous electric currents in the ground which to the travel surface and into the air. These currents alter the magnetic field surrounding the earthquake zone and these electromagnetic effects can easily be detected. Just before a large earthquake strikes electrical disrurbances can be detected at the edge of space in the ionosphere of the earth's atmosphere. The scientific expolanation is that rocks in the earth's crust begin to compress in the run up to an earthquake. When rocks compress they generate an electric current between the ground and atmosphere. The World as a result of the interaction between the atmosphere causes the formation of waves in the ULF frequency. Therefore, these results can be used in earthquake prediction.
ULF(Ultra-Low-Frequency) electromagnetic emission is recently recognized as one of the most promising candidates for short-therm earthquake prediction. This paper reviews previous convincing evidence on the presence of ULF emisions before a few large earthquake in Turkey and besides Turkey.
Keywords: ULF emission, Earthquake prediction, FET sensors

1. INTRODUCTION

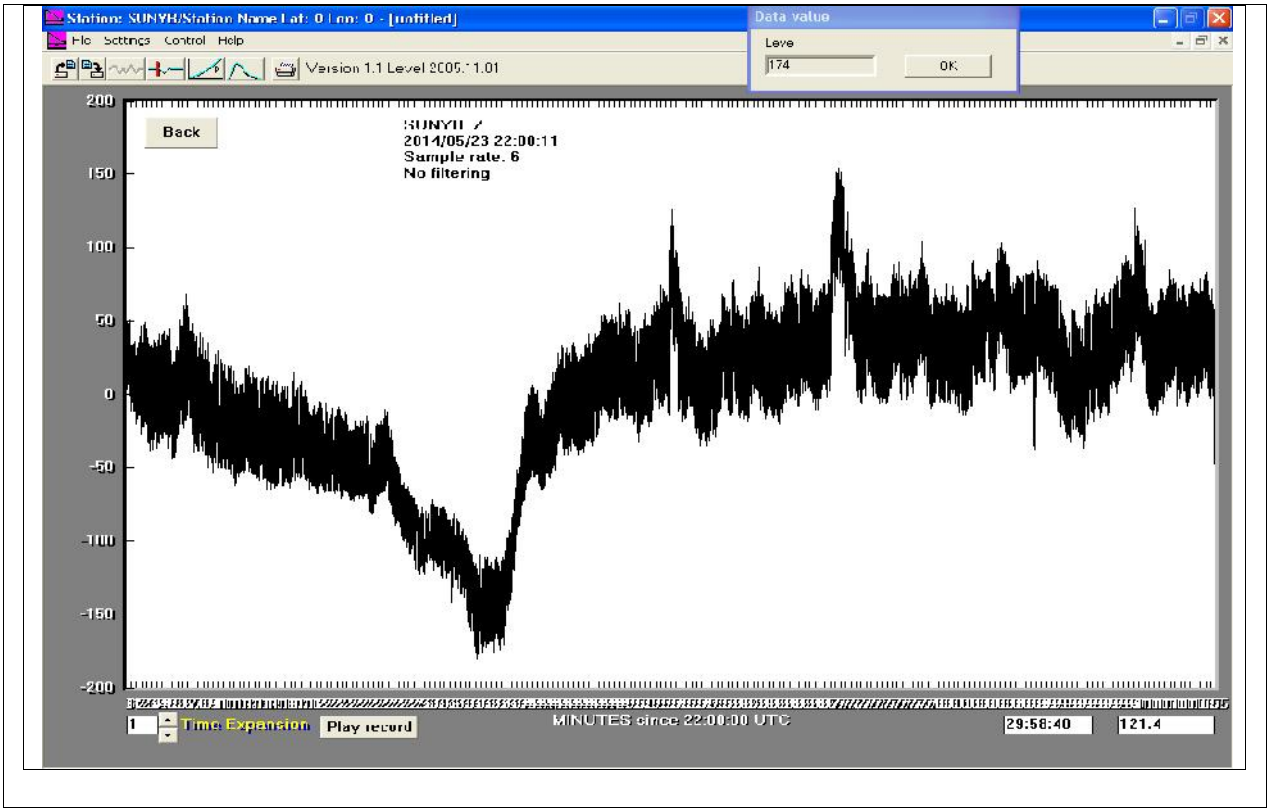
It has been recently reported that electromagnetic phenomena take place in a wide frequency range prior to an earthquake and these precursory seismo-electromagnetic effect are expected to be useful for the mitigation of earthquake hazards. Basically there are two principal methods of observation of earthquake signatures. The first is the direct observation of electromagnetic emissions from the lithosphere and the second is to detect indirectly the seismic effect taken place in a form propagation anomaly of the pre-existing transmitter signals. The first method is based on the idea that natural emissions are radiated from the hypocenter of earthquakes due to some tectonic effect during their preparation phase. The second is based on the idea that there take place the anamolies in the atmosphere and ionosphere due to the seismicity leading to the generation of propagation anomaly on the preexisting transmitter signal characteristics. This experimental study deals with the ULF electric field variations belonging to the first category. The study on seismogenic ULF emissions started in the early 1990S. Even though the radio emissions are generated as a pulse in the earthquake hypocenter higher frequency components can not propagate over long distances in the lithosphere due to severe attenuation, but ULF waves can propagate up to an observation point near the earth's surface small attenuation. This is the most important advantage of seismogenic emissions.

There have been reported three reliable events from the ULF electric field variations prior to the earthquake: (1) Armenia, Spitak earthquake (1988 December 8, Magnitude = 6.9) , (2) USA, California, Loma Prieta earthquake (1989 October 18, M = 7.1) , (3) Guam earthquake (1993 August 8, M = 8.0) and (4)Turkey earthquake(1999 Agust 17, Magnitude = 7.4), (5) Gulf of Saros earthquake, Turkey (2014 May 25, M = 6.5). The Gulf of Saros earthquake happened very close to the observing station, so that it is better for use the indicate the result for this earthquake. Fig. 1 illustrates the temporal evolution of ULF electric field (frequency = 0.009 Hz). It indicates that the electric field increases for about one hour before the earthquake,

followed by a quiet period and a sharp increase approximately one hour before the earthquake. Very significant changes in ULFelectric field were also observed for other earthquakes.



(A)



(B)

Fig. 1 Temporal evolution electric field variation for the Gulf of SAROS earthquake. (Turkey (M = 6.5, f = 0.009 Hz, A:Lowpass filter, B = No filtering)

2. HISTORICAL DEVELOPMENT

Considering the historical pain, some animal behavior before earthquakes, changes that have been recorded in historical sources. Animal behavior prior to earthquakes exchange of views is not too old to start. After 1950, changes in animal behavior prior to earthquakes have started to examine the cause. As a result of experimental studies, animals exposed to electric and magnetic fields, behavior were examined, were found to be similar to those resulting behavior before earthquakes. These experimental findings after researchers mechanism of earthquake events play an important role in the electric and magnetic is concluded. Turkey after the 1999 Marmara earthquake, conducted field observations, exchange of animal movements have been recorded in an obvious way. In vivo exposure to electric and magnetic fields it was observed a large increase in the hormone serotonin. A group of researchers from Istanbul University Faculty of Medicine in their experimental studies, passing us in determining seismic, electromagnetic events and have provided our views.

Below, some of the majo earthquakes have occurred on earth before the earthquake, the changes in electric and magnetic fields are observed. The cause of the abnormal behavio of animals before the earthquake, are abnormalities in electric and magnetic fields.

The Kobe Earthquake (M7.3) that occurred at 5:47 a.m., January 17, 1995, was caused by the movement of the Nojima fault on Awaji Island. Total official casualties were 6433, and over 40,000 were injured. The 1519 statements (1711 cases) on precursors collected by Wadatsumi (1995) through the mass media were mostly from the surrounding areas and can be classified thus:

Unusual animal behavior 872 (51%)

Sky and atmosphere 490 (29%)

Sea and land phenomena 189 (11%)

Electric appliances 149 (9%)

The Kansai Science Forum collected 173 statements from the epicenter (which Wadatsumi was unable to collect in the immediate post-quake confusion), but they were essentially the same as those he had already collected and similar to those described in Japanese legends and proverbs. As mentioned, the unusual behavior of home-electric appliances was a new feature.

The Izmit Earthquake in 1999 (M7.4)

The Izmit Earthquake (M7.4) at 3:02 a.m., August 17, 1999, destroyed Izmit and Adapazari on the North Anatolian fault, one of the world's longest and most active strike-slip (horizontal motion) faults. The 1999 event is the 11th quake of M> 6.7 since records were first kept. Local soil conditions under buildings also affected the degree of shaking and ground failures. Dr U. Ulusoy (1999), who returned to Turkey after eight months in our laboratory as a post-doctoral fellow studying Electron Spin Resonance (ESR) of geological fault materials, asked citizens to report anything unusual they noticed before the earthquake. She collected 880 statements by 348 witnesses (male: 198, female: 150) by letter (105), fax (114), email (86) and phone (43). However, there were few reports from the epicenter area. So we visited the epicenter areas, Adapazari and Izmit one month after the earthquake and collected 137 statements directly from witnesses then living in tents and nearby villages.

3. ELECTRIC FIELD SENSORS AND OBSERVATION SYSTEM

Before the earthquake, to detect changes in the electric field, the sensor is designed FET. In this sensor, which is used to FET 10 to the minus 12 pico amps have sensitivity. Thus, before the earthquake stuck in the rocks formed by the electric loads are available to detect. These small changes in the electric field of the load, the prediction of earthquakes is one of the most important parameters. Before the earthquake, some animals it detects these changes are available in the literature. On 05/25/2014, occurring approximately 2 hours prior to the earthquake in the Gulf of

Saros an obvious change was observed in the electric field Fig.1. A and B. We have achieved these results, is an important result for predicting earthquakes in the future.

The general structure of the measuring system can be summarized as follows:

1. Fet sensor
2. Filter
3. Amplifier
4. A/D converter 5. PC

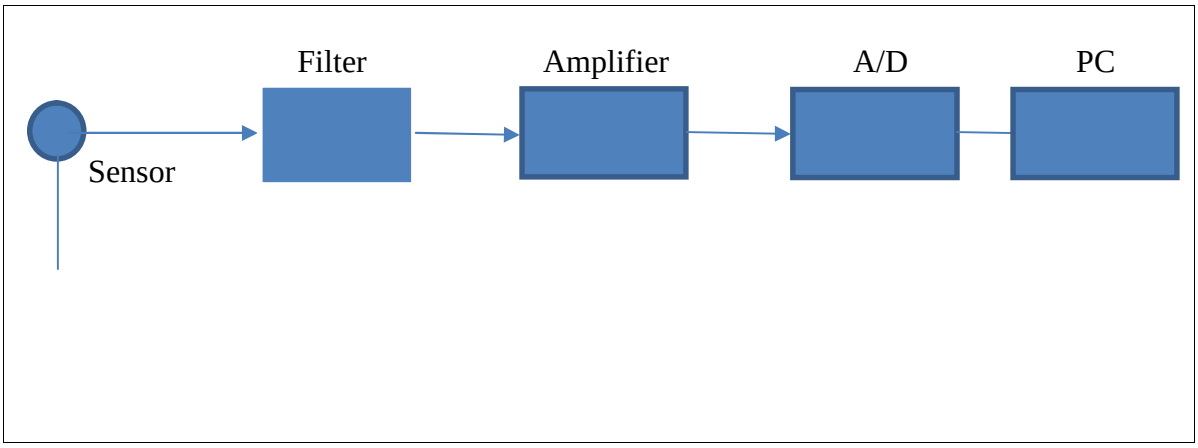


Fig. 2 General structure of the measuring system

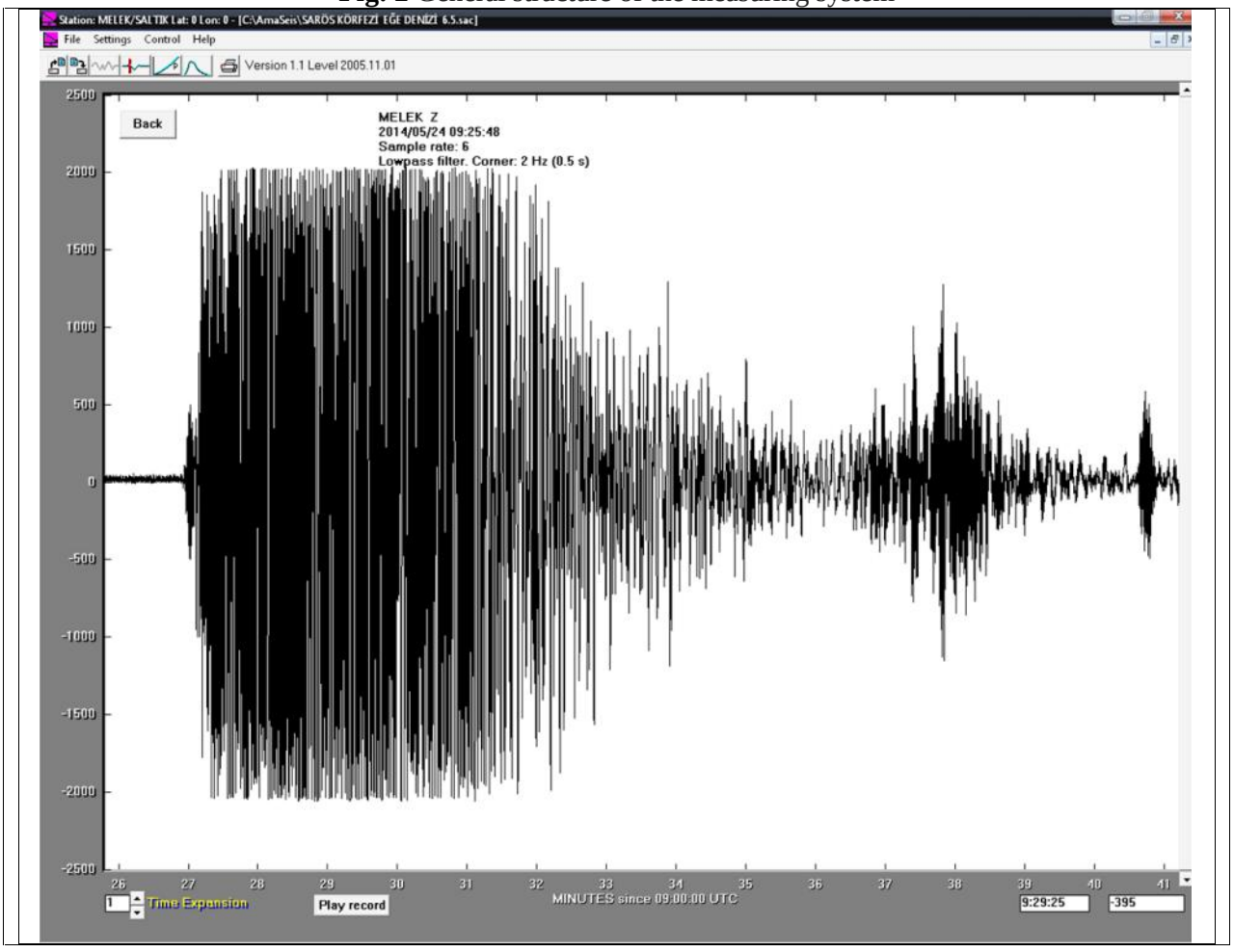


Fig. 3. 05.25.2014 Gulf of Saros Turkey earthquak seismometers record

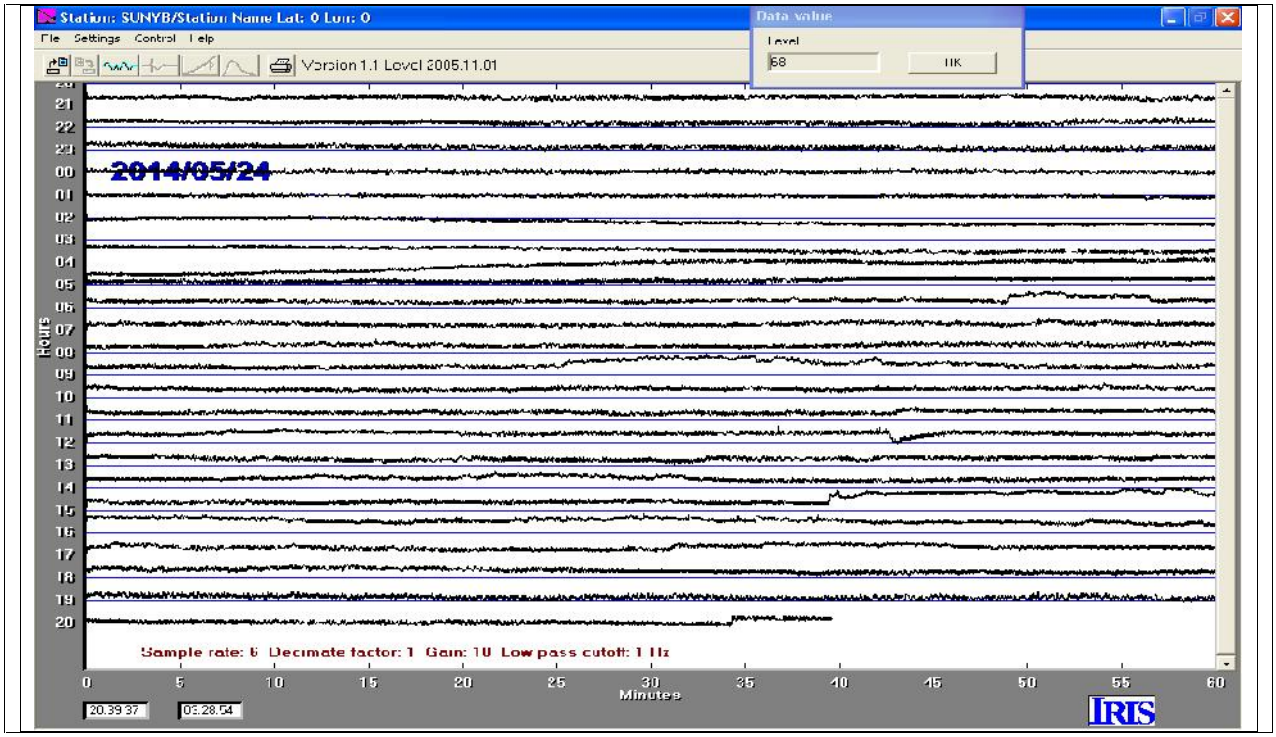


Fig. 4. The overall appearance of Records

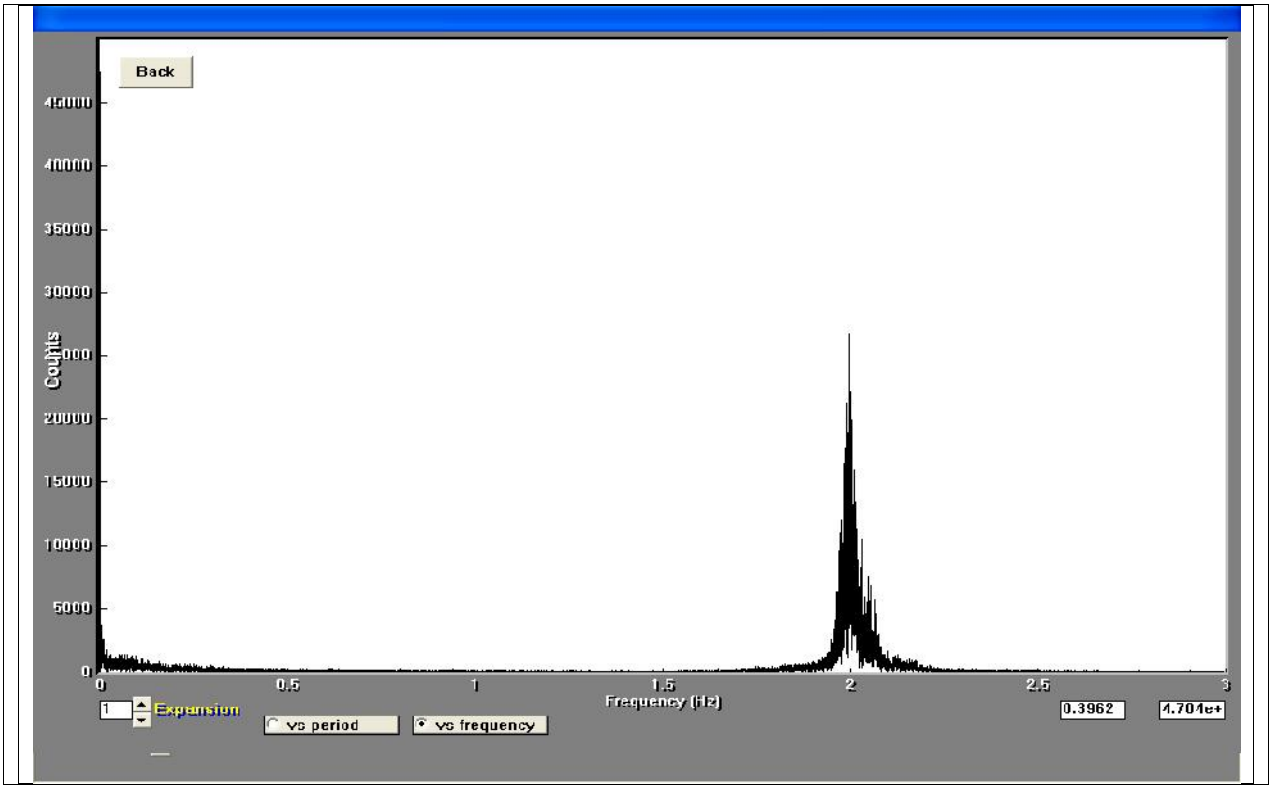


Fig. 5. Fourier transform of the electric field changes

DISCUSSION AND CONCLUSIONS

In this study we examined the electrical parameters that affect the global atmospheric circuit and the results were recorded graphically. The resulting graphic is observed from environmental sources of electricity or load column resistance and thus affects the global circuit.

In particular, high-energy particles from the sun and sunny outdoors were found to be quite dominant. Others are lightning in a dominant parameter. As you know, lightning has very high electrical voltage and current events. The effects of lightning registry system was seen as a sudden order, and this spark other effects according to the values of the voltage is quite high. Lightning in a short time and high global circuit with a voltage-current effects were observed. Although the effects of lightning short megawatts of power is the degree.

Other effects are the earthquake, as a result of the pressure occurring before earthquakes electric charges graphic effects have been registered as a global circuit. Before earthquakes occurring due to pressure loads, can be used in earthquake prediction.

In this study, the previous modification of the earthquakes and then are stored and defined as earthquakes.

In conclusion, this study examined the global atmospheric electrical phenomena been activated outside, the above results have been obtained, it is business.

REFERENCES

[1]. Ismaguilov, V. S.; Kopytenko, Yu. A.; Hattori, K.; Hayakawa, M. Variations of phase velocity and gradient values of ULF geomagnetic disturbances connected with the Izu strong earthquakes. Natural Hazards and Earth System Sci. 2002, 20, 1–5.

[2]. Ohta, K.; Watanabe, N.; Hayakawa, M. The observation of ULF emissions at Nakatsugawa, Japan, in possible association with the Sumatra earthquake. Int'l J. Remote Sensing 2007, 28, 3121–3131.

[3]. Ohta, K.; Watanabe, N.; Hayakawa, M. The observation of ULF emissions at Nakatsugawa in possible association with the 2004 Mid Niigata Prefecture earthquake. Earth Planets Space 2005, 57, 1003–1008.

[4]. Molchanov, O. A.; Hayakawa, M. Generation of ULF electromagnetic emissions by microfracturing.Geophys. Res. Lett. 1995, 22, 3091–3094.

[5]. Molchanov, O. A.; Hayakawa, M.; Rafalsky, V. A. Penetration characteristics of electromagnetic emissions from an underground seismic source into the atmosphere, ionosphere and magnetosphere. J. Geophys. Res. 1995, 100, 1691–1712.

CONTACTS:
Metin Saltik
Sakarya University, Turkey
SMYO - Adapazari
e-mail: saltik@sakarya.edu.tr

Eng. Suzan C. Mustafa, PhD student
Technical University of Varna
e-mail: suzan.chetinova@abv.bg

СЕКЦИЯ 4
ЕЛЕКТРОННИ ОБРАЗОВАТЕЛНИ ТЕХНОЛОГИИ

Втора научна конференция с международно участие
"Компютърни науки и технологии"
26-27 септември, 2014 г.
Варна, България

Second Scientific International Conference
Computer Sciences and Engineering
26-27 September, 2014
Varna, Bulgaria

SECTION 4
ELECTRONIC EDUCATIONAL TECHNOLOGIES

INTERACTIVE MULTIMEDIA TOOLS FOR ONLINE EDUCATION IN POWER ELECTRONICS

Angel St. Marinov

Abstract: The current paper suggests a set of interactive multimedia tools for presentation of complex educational material in power electronics. The tool set is developed in flash based application and was implemented for both lecture presentations and e-learning. The necessity of such tools is explained by defining basic problems and issues related to presenting materials in power electronics. The means to resolving these issues which can be accomplished through the use of interactive software, flash based animations are presented. An example with a dedicated application for power electronics is given.
Keywords: power electronics, electronic education, interactive tools.

Интерактивни мултимедийни приложения за електронно обучение по преобразователна техника

Ангел Ст. Маринов

Резюме: В настоящата статия са представени набор от интерактивни интернет базирани приложения. Тези приложения могат да бъдат използвани като презентации в част от лекционен курс или като инструмент за дистанционно обучение. Дискутирани са основните проблеми при представянето на материали за електронно обучение. Представени са основните средства за реализирането на тези приложения като интерактивен софтуер и флаш базирани анимации. Представен е пример на готово приложение.
Ключови думи: силова електроника, електронно обучение, интерактивни приложения.

1. Introduction

The term Power Electronics refers to a specific branch of the electronic technologies. This branch involves the conversion of electrical energy through the utilization of solid state semiconductor switches [1]. Power electronics are an important feature of every modern device powered by electrical energy. Some major application that can be mentioned include: (i) power supplies - basically for all modern electronic devices; (ii) control of energy generation - mostly in smart energy systems and renewable energy generation; (iii) energy conversion for industrial processes - induction heating, welding, electrical motor control, etc..[2].That defines power electronics into a required subject within higher education programs and curricula – under one form or another - for each engineer that specializes in the professional field of Power engineering, Electronics and Automation (professional field given defined by Bulgarian legislation [3]).

Currently with the improvement of computer technologies – online and remote studies become a popular option for higher education in many fields, including engineering. That poses various challenges in presenting and adapting knowledge and information for remote users. This is especially true for the subject of power electronics where complex information and data is presented through various block and circuit diagrams, waveforms, equations and text.

The current paper: (i) addresses some of those challenges by presenting issues and problems that were noted through analysis and derived from practical experience in online education in power electronics (Section 2. Presentation and adaptation issues for online education in the subject of power electronics); (ii) gives motivated suggestion for solutions of the noted issues (Section 3. Presented solutions for materials for online education in the subject of power electronics); (iii) gives

an example of the presented solutions (Section 4. Example application); gives a summary of the presented work (Section 5. Conclusion)

2. Presentation and adaptation issues for online education in the subject of power electronics

By analyzing the content of the initial source material for power electronics and its specifics, several basic types of issues can be distinguished. Those issues were defined based on a dedicated literature review on online education [4, 5, 6] and the experience gained through the conduction of two projects within the Technical University of Varna [7, 8, 9]. The projects include elements of online education and have courses relevant to power electronics. The issues can be summarized as follows into a three main groups:

a. *Difficulties in presenting of information in a comprehensive way that explains the correct processes in the circuit.*
   The purpose of each element in a power electronic circuit is to direct the electrical flows in order to convert energy for the use of specific electrical device. The main difficulty in explaining the operation and mechanics behind power electronic circuits lies in properly presenting the change of current or voltage and relation between the used component, defined formulas and obtained waveforms. This is especially problematic in online education, where the user has to discover the relation between circuitries, waveforms and equations by himself.
b. *Difficulties of providing simultaneous display of the equivalent circuits formed by the commutation of the electronic switches.*
   Other significant problem in e-learning materials for power electronics is to properly and clearly define the path of the current for a specific state of the components in a circuit and the change of path when related to a specific state of some components. In conventional teaching materials those relations are presented trough equivalent circuits, where each state of the main circuit is presented by a separate equivalent circuit. In online education this however requires multiple figures and further explanation, which would require multiple presentation screens that can confuse the user.
c. *Difficulties of representing complete information while maintaining a scroll free screen.*
   When presenting information for online education it is important to hold the attention span of the user. When various information including figures, diagrams, circuits and text is presented, multiple computer screens are required. This can be obtained either by scrolling or by switching between screens. If the material cannot be divided into small sections, this could distract the user, since for the same section text and figures have to be scrolled up and down or backtracked trough different screens. This is especially true for power electronics where the explanation of a single circuit could hold numerous figures, equations and large paragraphs of text.

3. Presented solutions for materials for online education in the subject of power electronics

Based on the issues mentioned above several approaches that can resolve them were developed and are suggested. The approaches include the utilization of modern computer based multimedia that can integrate animated features and user interactivity. In the case with examples that follow in the next section the computer multimedia is based on flash, HTML5 or other popular technologies can be used as well. These approaches can be summarized as follows into three defined solutions:

a. *Synchronization between waveforms, equivalent circuits transitions and explanations*
   The presented solution includes a simultaneous presentation of equivalent circuits, waveforms and process description. In order for the solution to work all of the components of the information have to be synchronously presented through animation. An especially

important part is the synchronism between the waveform animation and the change rate of the equivalent circuit. This allow the user to see the relation between the equivalent circuits (defined by the state of the electronic switches) and the change of the voltages and currents within the topology. The combination of all the components and their simultaneous presentation also allows the user to concentrate on the given application – information is presented in a single screen, scrolling and switching between screens can be avoided.

b. Presentation through multiple tabs on a single screen

The multiple components in a power electronics schematic could be presented on several subplot graphics with color code in order to easily present the path of the current, the change of voltage and certain switching state. This again leads to concentration of the information and a more comprehensive presentation that will not require the user to scroll or backtrack through different screens.

c. Comprehensive control of parameters that affect the circuit functionality

In every interactive presentation there should be a number of functional buttons, which control several important circuit parameters. The control of the parameters allows viewing different modes of operation within a single application. It will also allow the user to interact with the circuit adding an exploration element.

4. Example application

The presented in the previous section solutions are given as a summary in an example application. The example application involves a short course for controlled rectifiers. It combines three different types of rectifiers – single phase controlled bridge rectifier, single phase half wave controller rectifier and single phase controlled rectifier with centered – tapped transformer. The different types can be selected through invective buttons - fig.1. By clicking on the chosen schematic another slide opens as presented in fig.2 containing the related schematic (1) and two other areas, one for the parameters, equations and descriptions (2) and the other for the waveforms in defined test points of the schematic (3).

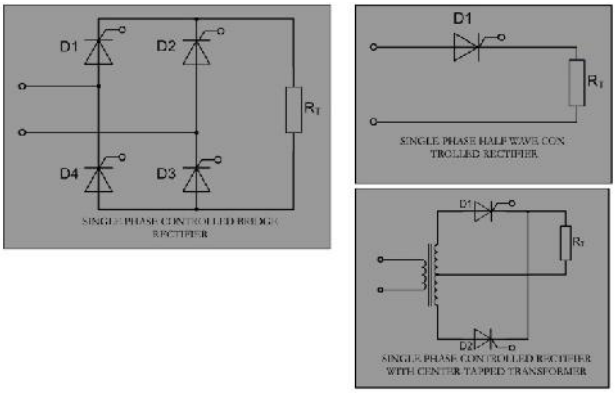


Fig. 1 Main view of the interactive application

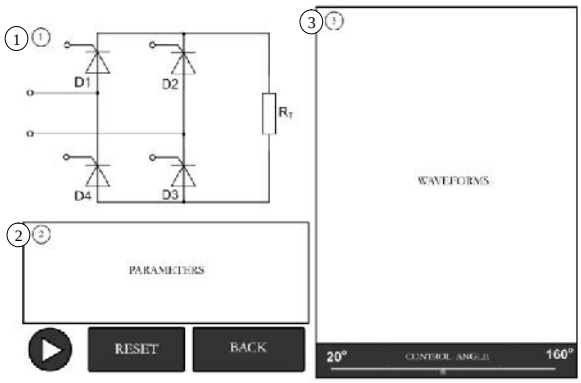


Fig. 2 Main view of single phase controlled bridge rectifier before simulation

The presentation is started by pressing an interactive button (fig.3). The graphics and parameters are color coded in order to be easily distinguished and associated with one another by the user. For example the voltage on Silicon Controlled Rectifier (SCR) D1 and SCR D3 is defined with blue. Thus it will be easily associated with the blue wave in the diagrams on the right and furthermore the resulted signals can be compared with one another.

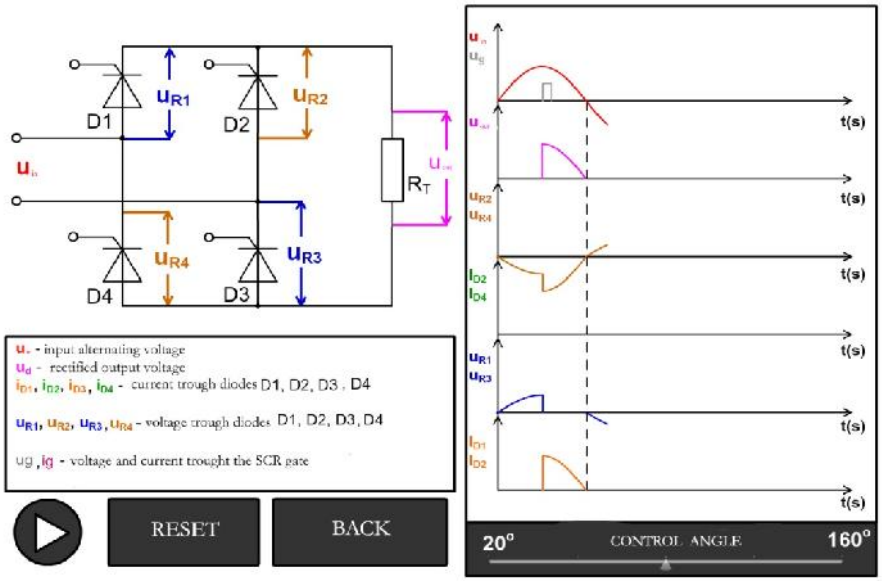


Fig. 3 Color coded example of running processes in the schematic

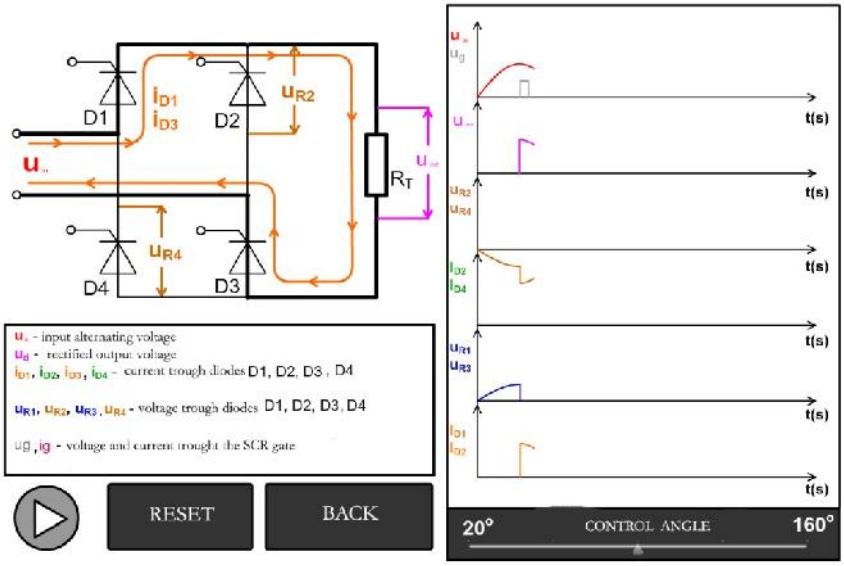


Fig. 4 Presents the current through D1 and D3

When a specific state of the components in the schematic is reached, this is easily discerned (fig.4 and fig.5). For example at first the current passes through SCRs D1 and D3 which is shown with orange on fig. 4. The green color defines the path of the current trough SCRs D2 and D4, which is also shown with the same color on the waveforms area on the right (fig. 5). Thus the change in state of the components, paths of current are clearly defined providing a scroll free screen and comprehensive definitions for the user. The different equivalent circuits are distinguished from the main topology through lines with higher width – combining in a single tab the main circuit and its equivalent branches.

An important parameter for the presented in the current paper schematic is the control angle. The control angle in this case can reconfigure the circuit and its parameters. Its effect however is difficult to explain through static images and waveforms. So in this case it is chosen that the control angle is to be defined by a slide control button. This is shown in fig. 6. Thus sliding the button to 20^o results in change of the control voltage on the SCRs, the current through them and consequently to a change of the output voltage of the rectifier. All this is implemented through a script file, implemented in the presentation where the theoretical formulas concerning the specified features

are defined. Thus an accurate and full description of the processes in current schematic allows the user to easily work and comprehend the main dependencies and work states without concerns.

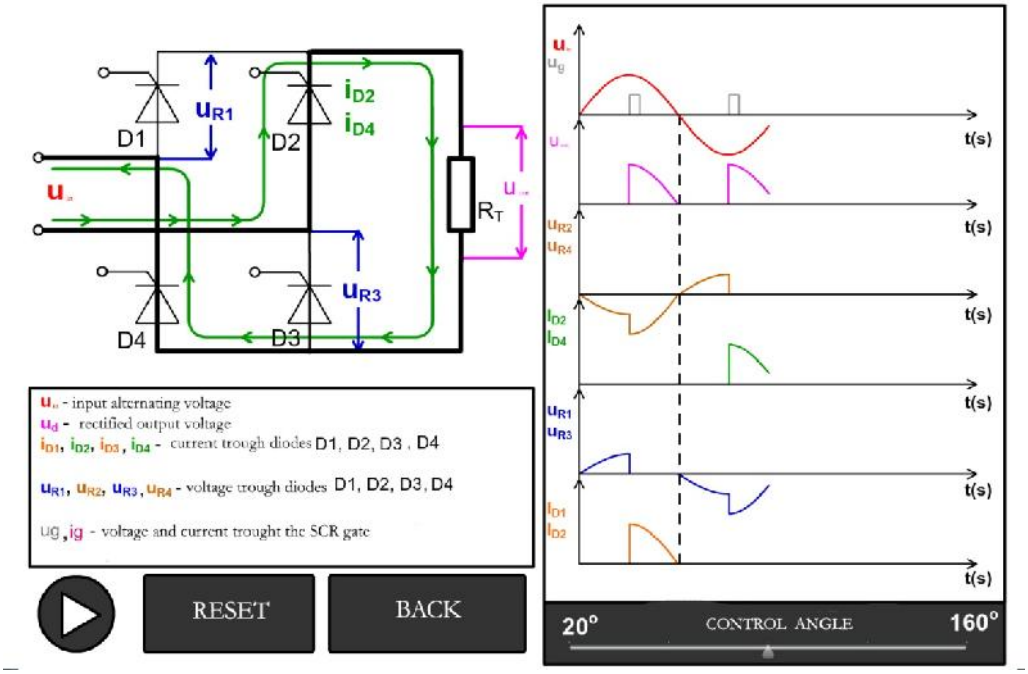


Fig. 5 Presents the current through D2 and D4

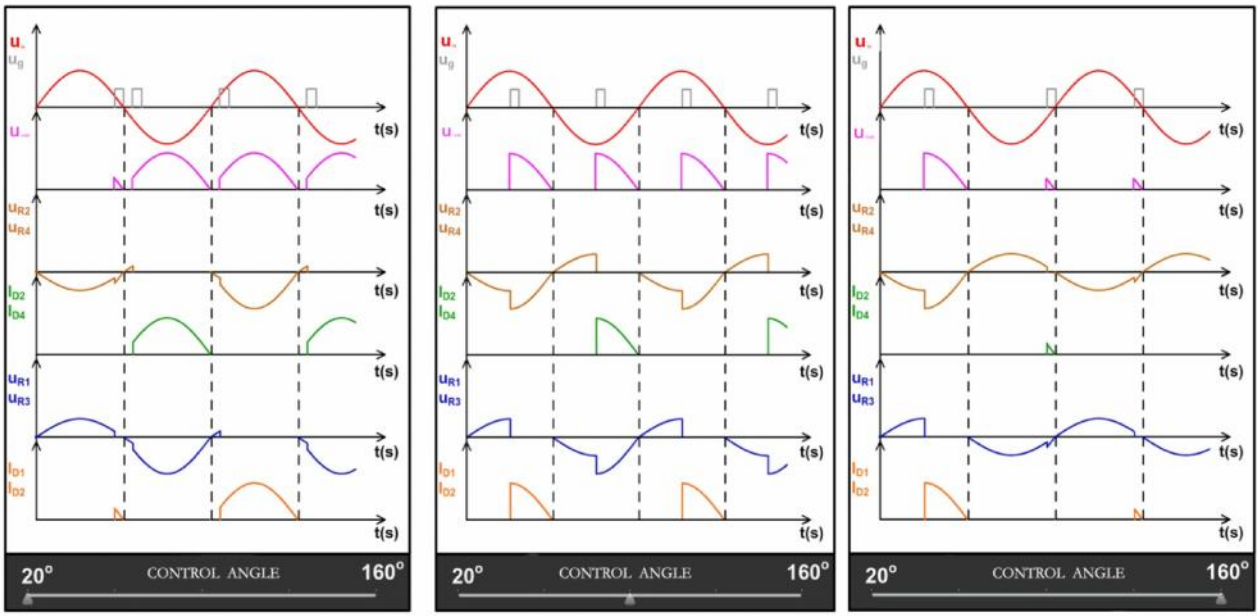


Fig. 6 Changing the control angle – from left to right waveforms for 20^o, 90^o and 160^o

The presented example is developed in flash based software, where: the circuit and the text are developed through animation with multiple frames; the waveforms and the control buttons are developed through the utilization of action script 3.0. The example and other materials are currently being developed in HTML5 in order to increase the compatibility to additional hardware platforms – mobile phones, tablets, etc. The current example, its source, along with other specialized media for online education can be viewed at the web page of the Department of Electronics and Microelectronics of the Technical University of Varna (http://www.tu-varna.bg/tu-varnaetm/)

5. Conclusion

Based on a specialized literature review and experience gained through the development of two international projects three major issues related to the online education in power electronics were noted. The issues included difficulties in presenting complex information in a comprehensive way that will not distract the user and can keep his attention. Solution to deal with those issues were presented. The solution include the utilization of modern media techniques. An example of the presented solutions is included in the paper. The example along with other similar appliactions is available and can be used free of charge. It was also extensively used and tested in lectures of power electronics in the Technical University of Varna.

Reference

[1]. M.H.Rashid, Power Electronics Circuits, Devices and Applications, 3rd Ed. Upper saddle, NJ: Pearson Prentice Hall. 2006.

[2]. Mohammed S., A. Moamen, B. Hasanin, A Review of the State-Of-The-Art of Power Electronics For Power System Applications, Quest Journals, Journal of Electronics and Communication Engineering Research, Vol. 1, Issue 1 (2013) pp: 43-52, ISSN(Online) :2321-5941

[3]. Класификатор на областите на висше образование и професионалните направления

[4]. Zhu J., Hybrid online-education strategy for delivering engineering and technology courses, 2nd International Conference on Networking and Digital Society (ICNDS), 2010, Volume:2, pages: 448 – 451; ISBN: 978-1-4244-5162-3

[5]. Drofenik, U., A. Musing; J. Kolar, Novel online simulator for education of power electronics and electrical engineering, International Power Electronics Conference (IPEC), 2010, Page(s): 1105 – 1111, ISBN: 978-1-4244-5394-8

[6]. Country report: Bulgaria, project: “BSUN Joint Master Degree Program on the Management of Renewable Energy Sources – ARGOS”, 2011

[7]. V. Valchev (2013), Interactive Multimedia Tools and Applications in Vocational Education in Wind Energy Technologies, International Jubilee Conference: 50-th Anniversary Department ETET, 2013, Varna, Bulgaria, 2013.

[8]. “Vocational Training in Wind Energy Technologies – Good Practices”; project “Transfer of Innovative VET System In Wind Energy Technologies” – TrainWIND, 2013

За контакти:
ас. д-р Ангел Станимиров Маринов
катедра „Електронна Техника и Микроелектроника”
Технически Университет Варна
E-mail: a.marinov@tu-varna.bg

ALGORITHM FOR IMAGE RECOGNITION AND PROCESSING FOR STUDENT EXAMINATION IN ELECTRONIC BASED EDUCATION

Mariana Iv. Shotova, Hristo B. Nenov, Angel St. Marinov

Abstract: This paper suggests an algorithm for image recognition and processing that can be used during student examinations and self-testing as a part of a system for electronic education. The algorithm compares user images containing graphs and waveforms to a preset defined image and on the basis of a preset threshold evaluates if the image correct or not. This allows to broaden examination techniques beyond simple tests and expand problem complexity. Such testing could be very beneficial for technical studies where many process are described and quantified graphically. The suggested algorithm is tested for various graphs and waveforms. In the paper one of those tests is presented where as an example a Volt-Ampere characteristic of a semiconductor rectification diode is used.
Keywords: Image recognition, image processing, electronic based education, Euclidian distance, edge detection

Алгоритъм за разпознаване и обработка на изображения с приложение при полагане на изпити чрез системи за електронно обучение

Мариана Ив. Шотова, Христо Ненов, Ангел Ст. Маринов

Резюме: Настоящата статия представя алгоритъм за разпознаване и обработка на изображения, който може да бъде използван при изпитване или самооценка на студенти, участващи в електронно обучение. Алгоритъмът позволява сравняването на изображения – графики, времедиаграми – разработени от потребителя, с предварително зададени такива. На база на дефинирани прагове се определя дали изображението на потребителя е правилно, постоянно или не. Това дава възможност за разширяване на спектъра на техниките за изпитване, като потенциалният принос би бил най-голям за техническите науки. Алгоритъм е тестван при използване на различни графики, като за настоящата статия е представен пример, базиран на волт-амперна характеристика на полупроводников диод.
Ключови думи: Разпознаване на изображения, обработка на изображения, електронно обучение, евклидово разстояние

1. Introduction

The development of modern computer and internet technologies changes in various and profound ways the means of human communication and social interaction. One of the areas where such changes are progressively taking part is education. Electronic and distance learning are already a verified and well applied concept taken by many Universities and colleges. This also includes the Technical University of Varna where in the recent years several projects related to electronic education were completed or are currently being finalized, with students underway to take part in the developed courses [1], [2], [3], [4].

For compliance with the aforementioned demand, new computer and software technologies are being developed, where various online platforms (Moodle, EFront, SEKOIA) that can host courses exist. Such platforms allow the authors, during the course development, to apply and use different multimedia tools and techniques [5], [6] (flash, HTML5, etc.) - presenting broad possibilities and ways to introduce materials or tests for the students.

Despite this already rich environment due to the diverse nature of possible areas of education new tools are always required. For example, one area where standardized testing does not allow for

a complete objective grading are the technical studies. In the technical studies often the courses include various graphs and waveforms that describe and/or quantify process and effects. In some cases students are required not only to pick and recognize the correct graph or waveform but to be able to draw it. For this cases few tools that allow computer based examination and self-testing exist. In line with this, the current paper suggests a computer based algorithm that allows the comparison between user drawn graphs and waveforms and predefined ones. In this way the system can evaluate whether students have acquired the correct knowledge or not.

2. Suggested algorithm

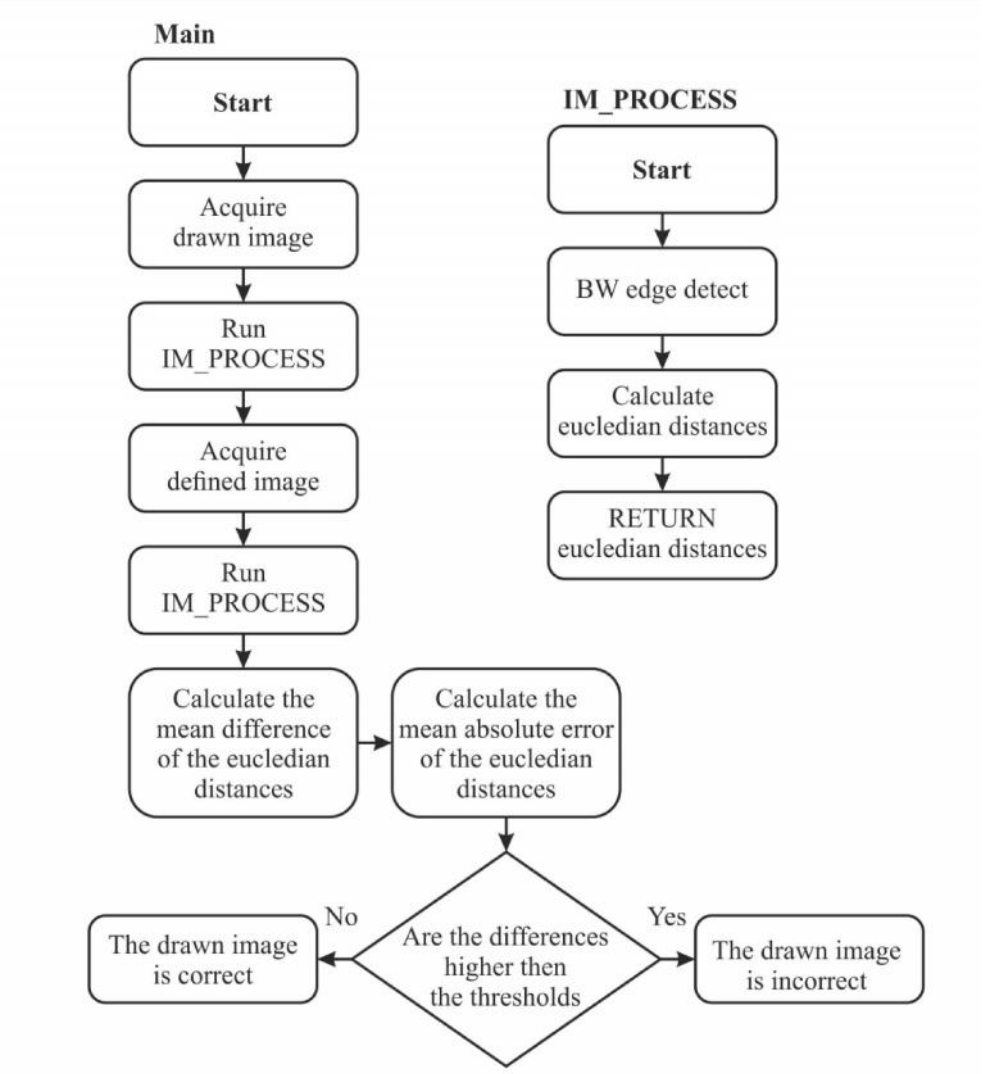


Fig. 1. Suggested algorithm flowchart

The flowchart of the presented algorithm is shown at figure 1. The algorithm can be sub-divided into two defined programs: main and IM_PROCESS (image processing). The algorithm flow can be described as follows:

- The image drawn by the user is loaded. This can be done in a separated GUI based environment where plot axis and a drawing window are already available. Additional complexity to the examination can be added if the user has to set the values of the axes as well.
- The loaded image is processed by calling IM_PROCESS, where the following is executed:

- A black and white edge detection based on Prewitt is used [7], [8], [9]. If the user is required to draw several graphs on the same axes, with different colors. Then a color segmentation can be applied before this function.
- The Euclidian distance for each pixel in a row is calculated.
- The calculated values are returned upon completion

• The defined image is loaded. This is the original correct image of the graph or waveform. This image has to be with the same size and drawn relatively to the same axes as the image dawn by the user.

• The loaded defined image is processed by calling IM_PROCESS, the same software code as described above is executed. The calculated Euclidian distances are returned in a different variable.

• A mean difference and a mean absolute error are calculated.

• Based on a predefined thresholds a decision on whether the graph is correct or not is taken.

For the described algorithm a dedicated software was developed. The software for initial testing was utilized as a MATLAB script. Some examples of its application are presented below.

3. Example algorithm application

The suggested algorithm and the developed software code was tested for various graphs and waveforms – mainly in the field of electrical and electronics engineering. An example of one of those tests, for the means of verification is presented below. The example includes a test with a Volt-Ampere characteristic of semiconductor diode – figure 2 – where six test curves were drawn by hand using a mouse interface – figure 3. The algorithm was tested with different interfaces as well – touch screen, drawing pad – but the mouse interface was harder to work with and provided distorted graphs, this is way it is used in the current example.

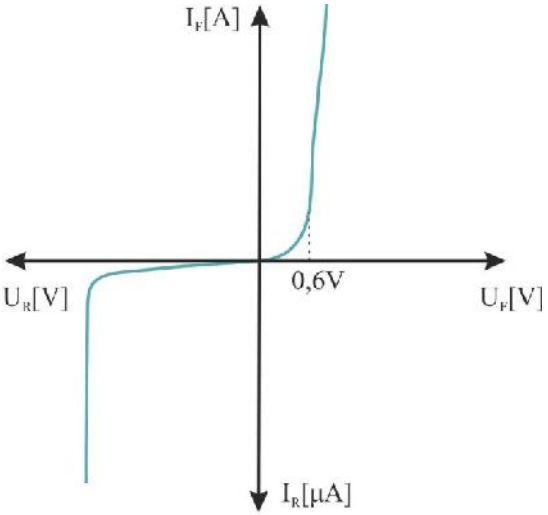


Fig. 2. Define graph example

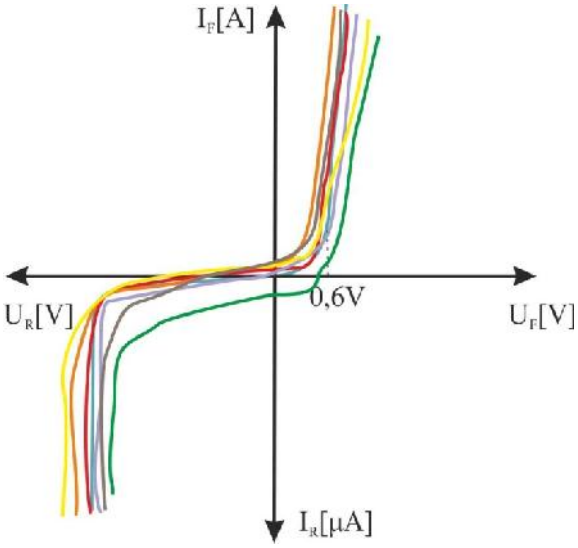


Fig. 3. Define graph and user drawn graphs

The complete set of graphs is presented at figure 4, where: Original – is the defined graph; 1÷6 are user graphs that are similar to the required graph but can be either correct or incorrect (the graphs from figure 3); 7 and 8 are completely incorrect graphs. The set was prepared in order to test if the algorithm is capable to differentiate between graphs that are similar to the required as well as between graphs that are completely different. In the set colors are used only for the reader to differentiate between the graphs easily. The actual graphs for the testing are drawn on black and white.

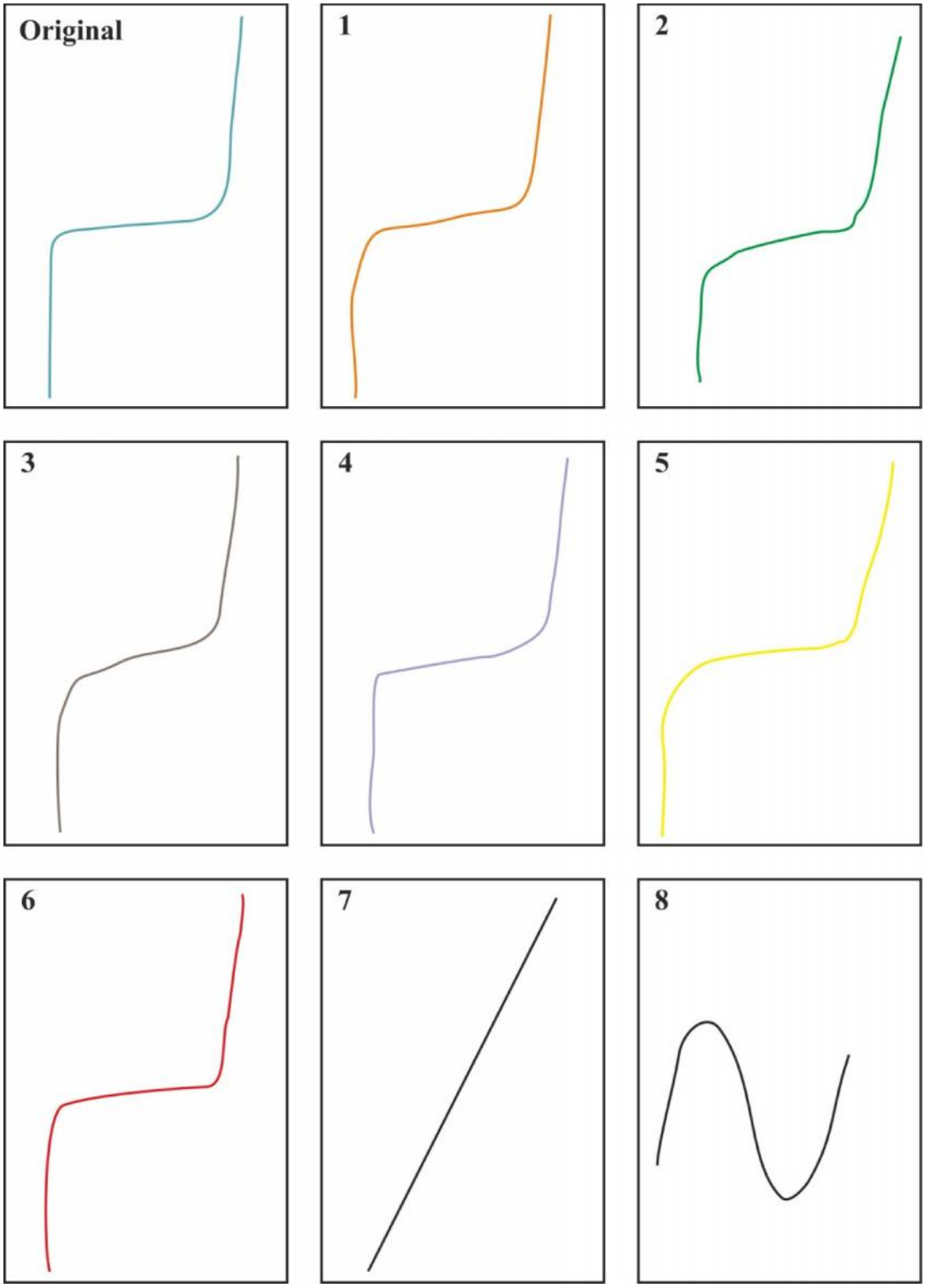


Fig. 4. Complete graph set

The results from the software application are presented at table 1, where images 1÷8 correspond to those in figure 4. The table includes: the calculate difference in the Euclidian distance (EDD) for the defined figure and for the drawn figure; the mean absolute error between the Euclidian distances (MEA); and the program decision – whether the drawn graph is correct or incorrect on the basis of the predefined graph.

For this test the comparison threshold for the drawn graph to be correct was set as EDD greater than 0 and MEA lower or equal to 100. Complying with this threshold it can be seen that

images 1, 4 and 6 are correct while 2,3,5,7 and 8 are not. 1, 4 and 6 are closest to the original, not only in shape but also in the way they are placed relevant to the axes, while 2,3,5,7 and 8 are either with the wrong shape or are not correctly placed relevant to the axes (cross different quadrants, are not correctly scaled).

The example shows that the algorithm is functional and operational, but also that it requires fine pre-usage adjustment. The thresholds have to be correctly set. For the current example, from table 1 it can be seen that if the MEA threshold was a little bit higher than 100 image 5 could also qualify – while generally correct this image has some inconsistence that cannot be accepted (crossing in the wrong quadrant.)

Table 1. Experimental results from the application of the suggested algorithm

Image	Euclidian distance difference (EDD)	Mean absolute error (MEA)	Program decision
1	74.6763	13.6377	Correct
2	1.3116	161.5024	Not correct
3	-20.1099	54.8998	Not correct
4	10.9662	57.6184	Correct
5	97.9988	111.2935	Not correct
6	15.1184	17.6208	Correct
7	-1.4191	168.2742	Not correct
8	-44.3430	237.6570	Not correct

4. Conclusion

An algorithm for image recognition and processing relevant to student evaluation in electronic education is presented. The algorithm is tested using various images, where example is given in the current paper. Based on the example it can be concluded:

- The Suggested algorithm is functional and operational
- Different graphs can be recognized compared to an original, with various degree of precision
- The algorithm can be applied to evaluation of students during courses in technical sciences, where the students are required to be able to draw graphs and waveforms
- Before using the algorithm for a specific graph thresholds have to be set so the required sensibility can be obtained.

Currently the algorithm is coded as a script in MATLAB, but due to its simple nature an application for different environments can easily be made.

Sample software code and additional example can be provided upon contacting the authors.

References

[1]. Country report: Bulgaria, project: "BSUN Joint Master Degree Program on the Management of Renewable Energy Sources – ARGOS", 2011

[2]. "Vocational Training in Wind Energy Technologies – Good Practices"; project "Transfer of Innovative VET System In Wind Energy Technologies" –TrainWIND, 2013

[3]. "New forms for electronic education in the Technical University of Varna" – project financed by the Bulgarian human resource development program.

[4]. EUTEMPE-RX: European Training and Education for Medical Physics Experts in Radiology, in FP7 Fission-2013-5.1.1: EURATOM Fission Training Schemes (EFTS) in 'Nuclear Fission, Safety and Radiation Protection', Project Number: 60529801.08.2013 - 31.07.2016.

[5]. A. Marinov, K. Bliznakova1, I. Buliev, H. Bosmans, N. Van Peteghem, R. Verraest, R. Padovani, S. Christofides and C. Caruana. “Application of advanced techniques for online presentation of educational material for education and training developed within the EUTEMPE-RX project”, MBEC2014, Dubrovnik, Republic of Croatia, 2014
[6]. V. Valchev (2013). „Interactive Multimedia Tools and Applications in Vocational Education in Wind Energy Technologies”, International Jubilee Conference: 50-th Anniversary Department ETET, 2013, Varna, Bulgaria, 2013.
[7]. J. Canny. “A computational approach to edge detection”, IEEE Transaction on pattern analysis and machine intelligence, Vol. PAMI-8, No 6, 1986, pp. 679-698
[8]. J. Lim, Two-Dimensional signal and image processing, Englewood Cliffs, NJ, Prentice Hall, 1990, pp. 478-488
[9]. J. Parker, Algorithms for image processing and computer vision, New York, John Wiley & Sons, Inc., 1997, pp. 23-29

За контакти:
ас. д-р Ангел Станимиров Маринов
катедра „Електронна техника и микроелектроника”
Технически университет Варна
E-mail: a.marinov@tu-varna.bg

Bulgaria Communications Chapter

КОНЦЕПЦИЯ ЗА АКТИВНО УЧЕНЕ С ЕЛЕКТРОННИ ОБРАЗОВАТЕЛНИ РЕСУРСИ

Марияна И. Николова

Резюме: Целта на статията е да се представят основните дейности за прилагане на активните методи на учене в образователния процес и използването на електронните ресурси за повишаване на неговата ефективност. Показани са предимствата на този стил на обучение и ролята на участниците в този процес. Разгледани са дидактическите и технологичните аспекти при проектиране на процеса на активното учене.
Ключови думи: пирамида на учене, активен стил на учене, електронни средства за активно учене, електронни образователни ресурси.

Concept for Acrive Learning with Electronic Educational Resources

Mariyana I. Nikolova

Abstract: The aim of this report is to describe, present the main activities for the application of active learning methods and to determine their purpose and role in the e-learning process in order the e-learning effectiveness to be increased. There are shown the advantages of this style of training and the role of participants of this process by using the electronic technology. Report examines didactic and technological aspects in the design process of active learning.
Keywords: pyramid of learning, active learning style, electronic means of active learning, electronic learning resources.

1. ВЪВЕДЕНИЕ

Активно учене е стил на обучение чрез разнообразни учебни дейности - учене чрез действие, учене чрез трупане на опит, учене чрез практика. Дидактическата трактовка разглежда активните техники като ефекивен прийом, чрез който от една страна се достига до знанието, а от друга се стимулира креативното начало на това познание. Тази интерпретация се основава на два основни факта – познанието е творчески акт на взаимодействие между субекта и обекта на познанието и действието като характеристика на мисленето. Проблемността и активността като методически елемент не се отнася само до формиране на различни знания и компетенции, но и за изработване на модел за търсене, откриване и решаване на проблеми в ситуации от различен тип.

В настоящата статия се разглежда спецификата и учебните дейности в процеса на активното учене. Целта е да се покажат предимствата на този стил на обучение и ролята на всички участници в образователния процес. Електронните образователни ресурси (е-ресурси) са сред най-важните компоненти на този процес.

Основни електронни образователни ресурси (е-ресурсите) са: електронни учебници и книги; компютърни игри и симулации, моделиране на обекти и явления, тестови системи; информационно-търсещи справочни системи; средства за имитационно моделиране; средства за автоматизация на професионални дейности; интерфейси към лаборатории за отдалечен достъп, виртуални лабораторни практикуми; комплексни обучаващи програми и др.

Съвременното обучение все повече се основава на активни методи, при които обучаемият играе основна роля, а в е-ресурсите е концентрирана съдържателната съставляваща на учебния процес. Ролята на е-ресурсите в учебния процес е съществено по-голяма, отколкото на традиционните книжни формати, защото новите образователни

технологии предполагат промяна на контактите ‘преподавател-обучавани’ и увеличаване на времето за самостоятелна подготовка [7]. Към е-ресурсите са насочени допълнителни очаквания - да осигурят поддръжка на обучението в онази негова част, която се използва за създаване на нещо реално, за симулация на практически дейности и за защитаване и обосноваване на тези и позиции. Тогава говорим за проектиране на образователния процес върху основата на активното учене чрез използване на електронни образователни ресурси.

Наблюдава се трайна тенденция в посока на сериозно увеличаване на виртуалната образователна информация, включително и съхраняване на цели обучаващи системи като е-ресурси. Основен проблем в това отношение е, че голяма част от тези области в образователното виртуално пространство не са свързани.

2. ПИРАМИДАТА НА УЧЕНЕ И СТИЛЪТ НА УЧЕНЕ

„Пирамидата на учене” [5] е един модел, който обяснява каква е ефективността на учене в зависимост от различните начини, с които се подхожда към него. Известна е на света още от първата половина на XX век и определя склонността на човек да помни определена част от информацията. Пирамидалният модел е разработен от Брус Хайланд в средата на миналия век като Конус на ученето– фигура 1. Моделът прави разлика между пасивното и активното учене. По-доброто запаметяване, а и по-доброто разбиране на нещата е в резултат на активно учене, докато при пасивното учене също има запаметяване и разбиране на наученото, но с големи “загуби”.

Пирамидалният модел недвусмислено показва, че съществуват разнообразни начини за усвояване на знанията, т.е. хората учат успешно по повече от един или два начина. Според Пирамидата на ученето от фигура 1 се вижда, че ако се запомня едва 10% от това, което се чете, то трайността на знанията са 90% от това, което се казва и се реализира чрез правене на нещо реално или чрез симулация на създаване на нещо реално.

Фиг.1. Пирамида на ученето

Следвайки пирамидата става ясно, че най-малко се научава от лекция и четене. Дискусията и практиката са далеч по-ефективни подходи, защото човек участва в процеса активно, с творчество, с въображение и с действия. Логическата обосновка за това е, че при слушане работи само една репрезентативна система – слуховата, а когато се включат повече системи, нещата стават много по-лесни.

Съвременните образователни модели и практики, базирайки се на педагогическите теории, безспорно доказват, че много подходящи за успешното преподаване и надграждане на знанията и компетенциите са методите на активно учене с помощта на е-ресурси.

Системата на обучение може да се определи като конструиране на образователния процес [1], [3], чиято цел е учащите да постигат по-добри резултати чрез подходящи средства и методи. Съответствието с индивидуалния стил на учене е от особено значение, който в общата класификация от [2] и [4] е: зрителен стил – учене чрез наблюдение; слухов стил – учене чрез слушане и тактилен стил – учене чрез движение, действие и правене. Когато „системата" открие как се обработва информацията най-добре, може да се учи по-ефективно и с по малко време като се разширят стратегиите, използвани за обучение и изследване.

3. ХАРАКТЕРИСТИКА И СПЕЦИФИКА НА МЕТОДИТЕ ЗА АКТИВНОТО УЧЕНЕ

Активното учене включва следните ключови дейности:

- добре подготвени разнообразни учебни дейности – ролеви дейности, учебни казуси, групови проекти и семинари;
- учене чрез действие, изпълнение и представяне на резултат от страна на обучаемите;
- учене при трупане на опит чрез практика;
- дискусии, решаване на проблеми, анализ и оценяване;
- систематичен процес на осмисляне на дейността, така че тя да води до натрупване на умения и компетенции.

Основна теза на съвременното разбиране на активното учене е намиране на един разумен баланс между познанията и прагматичните умения. Това привлича много образователни системи, защото обучаемите научават къде и как ще могат да използват знанията. За добрия резултат са важни факти като потребности от обучение, вътрешната мотивация на обучаващите се, контекст на провеждане на обучението и уменията на преподавателя да прилага принципите на активното учене.

Концепцията за активно учене включва основно проблемно-базирано обучение и обучение чрез проекти.

Проблемно-базираното обучение е такава организация на учебния процес, при която преподавателят създава проблемна ситуация и под негово ръководство обучаемите самостоятелно или с малко помощ и творческа работа решават задача. Когато има наличие на проблемна ситуация [6]:

- ✓ Определят се необходимите знания за решаване на проблема.
- ✓ Търсят се "знания" от допълнителни източници.
- ✓ Реализират се активни действия от страна на обучаемите.
- ✓ Стимулират се мисловните процеси и се развиват познавателните умения.

Предимствата на проблемно-базирано обучение са показани на фигура 2.

Фиг.2. Предимства на проблемно-базирано обучение

Проектът като образователна технология представлява съвкупност от различни дейности – планиране, изпълнение на заданието, решаване на текуши проблеми, вземане на решения, оформяне на документация, защита на проекта. Предимствата са показани на фигура 3. Това води до създаване на продукт или изследване на проблем от реалния живот от обучаемите под ръководството на преподавател. Обучаемите се научават да използват и прилагат знания и умения от различни областта с помощта на средствата на информационните и комуникационни технологии (ИКТ), да работят в екип, да представят завършен информационен продукт като придобият опит и увеличат познанията си [9]. Проектът е процес, ограничен в рамките на определено време и ресурси и обикновено изисква работа в екип, налагащо разпределяне и координиране на задачи и отговорности като се отчитат личните качества и компетенции на всеки.

Тази образователна технология развива креативност, формира определени личностни качества, като създава култура на поведение и мислене в конкретна ситуация. Обучаемият действа ангажирано в реална ситуация, поема отговорност за това, учи чрез натрупване на опит и чрез преживяване на случващото се.

Фиг. 3. Предимства на проектите като образователна технология

4. ПРИЛОЖЕНИЕ НА АКТИВНОТО УЧЕНЕ

Разгледаните методи за активно учене намират приложение в практиката на много преподаватели. Успехът при използване на методите на активното учене е в тяхното правилно прилагане. Ролята и задачите на преподавателите могат да се обединят така:

- ✓ Насочване и мотивиране на обучаемите към подходящи учебните дейности - посочване на значението на проблема или темата на проекта, предизвикване и поддържане интереса за практическото решаване;
- ✓ Акцентиране и насочване към знанието за решаване на проблема или реализиране на проекта чрез въвеждане на ново учебно съдържание или припомняне на знания посредством дискусии;
- ✓ Обяснение чрез предоставяне на примери и контра примери;
- ✓ Предоставяне на възможност за изпълнение на дейности и проверка на адекватността на наученото;
- ✓ Обсъждане на решенията и тяхното съответствие в различни ситуации.

За съжаление резултатите не винаги постигат реално поставените цели. При разработване на съдържателната компонента на учебния материал и заданията за проектите преподавателите не отчитат и не се съобразяват със спецификата на целевата група от една

страна и специалните дидактически подходи при прилагане на активните образователни методи. Ролята на преподавателите е важна и не бива да се ограничи само с поставяне на задание и няколко кратки указания, защото тогава проектите се отчитат формално, като не рядко те „са копирани” и раелния резултат не решава поставените дидактически задачи.

Използването на подходящия електронен образователен ресурс и съответните технологични средства често изисква допълнителна подготовка за това.

Образователните цели като резултат от прилагане на методите за активно учене са свързани с придобиване и усъвършенстване на различни умения. За успешното реализиране на обучаемите на пазара на труда е важно у всеки да се развиват освен технически и „нетехнически” умения като решаване на проблеми, работа по проект, работа в екип, информационни умения, умения за представяне на информация [8].

Прилагането на методите за активно учене са добра база за изграждане на комплекс от ключови умения за работа и представяне на информация. *Информационните умения* са свързани със събиране и обработване на информация по подходящ начин за постигане на определена цел и включват:

- определяне на даден информационен проблем;
- определяне дали дадени информационни ресурси са подходящи за конкретна цел;
- систематично търсене чрез прилагане на подходящи техники;
- ефективно обработване на намерената информация за постигане на определена цел.

Уменията за работа по проект са свързани с дейности като:

- определяне и формулиране на задачи и подзадачи и планиране;
- общуване в екипа;
- проследяване и отчитане на развитието на процеса и обединяване на резултатите;
- отчитане на резултатите.

Учебните дейности, свързани с отчитане, документиране и защита на проекта спомагат за изграждането на *умения за представяне на информация* като:

- подбор, подреждане и структуриране на информацията по представянето;
- използване на подходящи средства, стил и език;
- оформяне на представянето.

5. ЗАКЛЮЧЕНИЕ

С настоящия доклад се насочва вниманието към активния стил на обучение, който е един от ефективните пътища за постигане на нов тип въздействие, създаващо условия за трайни знания и компетенции и пълно разгръщане и използване на личностния потенциал на обучаемите.

Потребностите на информационното общество в периода на глобална комуникация са свързани с достигане на определено ниво на различни компетенции. Уменията да се използват предимствата на ИКТ - достъп, обработка, интегриране, оценката, създаване, търсене, разпространение на информация са ключови в контекстта на различни предметни области. Подходите, целящи изграждане на професионални компетентности, се основават на „динамично използване и интегриране” на:

- знанията, които се прилагат за различни дейности, а не само като вид усвоена информация;
- уменията, чрез които се прехвърля въздействието върху различни обекти;
- навиците, осигуряващи осъзнато и интуитивно действие, дори когато ситуацията е нестандартна и не толкова позната.

Проблемно-базираното обучение е в основата на проектното обучение, свързано с активни действия от страна на обучаемите и учене чрез правене. Проектът създава условия за самостоятелна работа, за използване на различни техники и интеграция на знания и умения от

различни области, като творчески се използват информационните и електронните технологии в различните предметни области.

Съвременното образование все повече се основава на активните методи на обучение и няма учебна дейност, която да не е свързана с използване на електронните образователни ресурси и виртуалната информационна и комуникационна среда.

ИЗПОЛЗВАНА ЛИТЕРАТУРА

[1]. Димов П., Електронно обучение чрез Интернет. София, Сиела-Софт Паблишинг, 2004.
[2]. Михова М., Преподаването и ученето. Теории. Стилове. Модели, 2005
[3]. Николова М., Проблемно-базираният и проекктният поход в обучението по информационни технологии - необходимост и възможности. Издателство «Абагар» Велоко Търново, 2012.
[4]. Павлов Д., Образователните информационни технологии. Модул 1. Университетски курс, изд. Даниела Убенова, С. 2001
[5]. Христов Т., Пирамида на ученето, Сита Мениджмънт Консулт ООД, (http://www.novavizia.com/10864.html), София, 2011
[6]. TREE – Teaching and Research in Engineering in Europe. Special Interest Group B5 "Problem based and project oriented learning", 2007.
[7]. http://bglog.net/ClientFiles/ab85a094-b9ad-4621-b158-2fca1ef64ca7/G_Totkov.pdf
[8]. http://research.uni-sofia.bg/bitstream/10506/198/1/original_dipl_rab.pdf
[9]. http://www.rusedu.info/Article982.html, Применение проблемного обучения при изучении информатики.

За контакти:
Доц. д-р Марияна Николова
Катедра: „Компютърни системи и технологии”
ВТУ „Св.Св.Кирил и Методий”, В. Търново
E-mail: mnikolova_vt@abv.bg

СИМУЛАЦИЯ НА ОБЛАКОВИ ИЗЧИСЛИТЕЛНИ СРЕДИ (CLOUDSIM)

Деян П. Атанасов, Трифон И. Русков

Резюме: Облаковите изчисления са съвременно средство, при което ИТ инфраструктура и приложения се предоставят като услуги на крайни клиенти. Целта на този доклад е да опише и оцени CloudSim - нов, обобщен и разширяем инструмент, позволяващ безпроблемна симулация и експериментиране върху нововъзникващите облакови инфраструктури и услуги.
Ключови думи: облакови изчисления, център за данни, симулация, управление на ресурси.

Simulation of Cloud Computing Environments with CloudSim

Deyan P. Atanasov, Trifon I. Ruskov

Abstract: Cloud computing is a recent advancement wherein IT infrastructure and applications are provided as ‘services’ to end-users under a usage-based payment model. The aim of this paper is to describe and evaluate CloudSim - a new generalized and extensible simulation framework that enables seamless simulation, and experimentation of emerging Cloud computing infrastructures and management services.
Keywords: cloud computing, data center, simulation, resource management.

1. Въведение

Активното развитие на разпределената обработка на информацията, базирана на широко разнообразие на изчислителни среди и платформи, постави основата за развитие на нови, по-гъвкави архитектурни модели и концепции като Грид и облакови изчисления. Облаковите изчисления се базират основно на виртуализацията и скриване от потребителите на ресурсните детайли, удовлетворявайки изискването за удобен достъп, надеждност и сигурност на предоставяните услуги [1]. От друга страна облаковите изчисления представляват и динамичен бизнес модел, при който доставчик на дадена услуга може да продава или препродава тази услуга на своите клиенти. Динамиката на модела се определя от факта, че клиентът ползва услугата само когато е необходима и съответно заплаща само за периода на нейното използване. От това произтича и необходимостта за поддържане на високо качество на предоставените услуги.

2. Облаково базирани услуги

Облаково базираните услуги се класифицират в три основни класа [1]:

- Софтуер като услуга (SaaS) – предлага възможност за наемане на приложение от доставчик, работещо в облаковата среда. За достъп до приложението е необходим единствено уеб браузър. SaaS услугите са предназначени за крайните потребители. При тях потребителите не управляват облаковата инфраструктура, (мрежи, сървъри, операционни системи, съхранение на данните), а само някои потребителски настройки на самото приложение. Пример за SaaS услуги са електронна поща, социални мрежи, уеб хостинг, симулационен софтуер, обработка на данни в реално време и др.;
- Платформа като услуга (PaaS). Услугите PaaS са предназначени за разработчици на услуги. Чрез тях се предоставя достъп до среда за разработване. Клиентът на

услугата има възможност за изграждане на собствени приложения, работещи върху наетата среда. Към PaaS спадат услуги за разработване, тестване и интеграция на софтуер. Потребителят на PaaS също не управлява облаковата инфраструктура;

- Инфраструктура като услуга (IaaS). Предлага възможност за наемане на изчислителни мощности и съхранение на данни като стандартна услуга. Услугите IaaS са предназначени за системни инженери. Използват се за създаване на платформи за услуги. На клиента се предоставя възможност за използване на сървъри, операционни системи, системи за съхранение на данни, мрежово оборудване и др. Той няма контрол единствено върху цялостната облакова инфраструктура.

Върху качеството на предоставяните облакови услуги съществено влияние оказват производителността на реализация на услугите и политиката за осигуряване на необходимите ресурси. Количествената оценка на производителността и политиката в реална облакова среда (каквито са Amazon EC2, Microsoft Azure, Google App Engine) за различните класове услуги се постига изключително трудно поради следните факти:

- различните облакови структури имат различни изисквания към модели за доставки, размери на системата и ресурсите и не могат да бъдат унифицирани и параметризирани;
- потребителите имат разнородни изисквания по отношение на различни аспекти на качеството на услугите (QoS);
- изпълнението на приложенията силно зависи от предоставяната от облаковата структура производителност, моментната натовареност и възможността за мащабиране на средата. Условията за изпълнение са динамични и трудно могат да се предвидят за точната оценка на производителността.

За определяне и оценка на производителността на дадено приложение или услуга не е удачно да се използват реални облакови структури като Amazon EC2 или Microsoft Azure, тъй като разходите за подобно нещо в една мащабна облакова среда биха били огромни. Освен това, отнема значително време, когато трябва да се преконфигурират параметрите, по които се извършва оценяването в облаковата среда. Така че не е възможно да се направи сравнителен анализ и експерименти на производителността в повторяеми, надеждни и скалируеми среди, използвайки реални облакови среди.

Алтернативата е използването на инструменти за симулация. Тези инструменти дават възможност за оценяване на хипотетични изследвания в контролирана среда, където резултатите могат лесно да се възпроизведат.

3. Симулатор на облакови среди CloudSim

Най-известната симулираща система на облакови среди е симулаторът CloudSim. Той е обобщена и разширяема система за симулация на разнообразни облакови инфраструктури и оценка на алгоритмите за разпределяне на ресурсите. С помощта на CloudSim изследователи и разработчици могат да извършват тестове за производителност на разработените от тях услуги в контролирана и лесна за настройка среда [2], [3]. Към достойнствата на симулатора могат да се отнесат времевата ефективност, изразяваща се във високо бързодействие при изпълнение на симулационните модели, гъвкавост и приложимост към различни облакови концепции. Тестването на разработените модели може от своя страна да се изпълни в реални облакови среди. CloudSim предлага специфични към облакови среди и услуги функционални възможности, по-важните от които са:

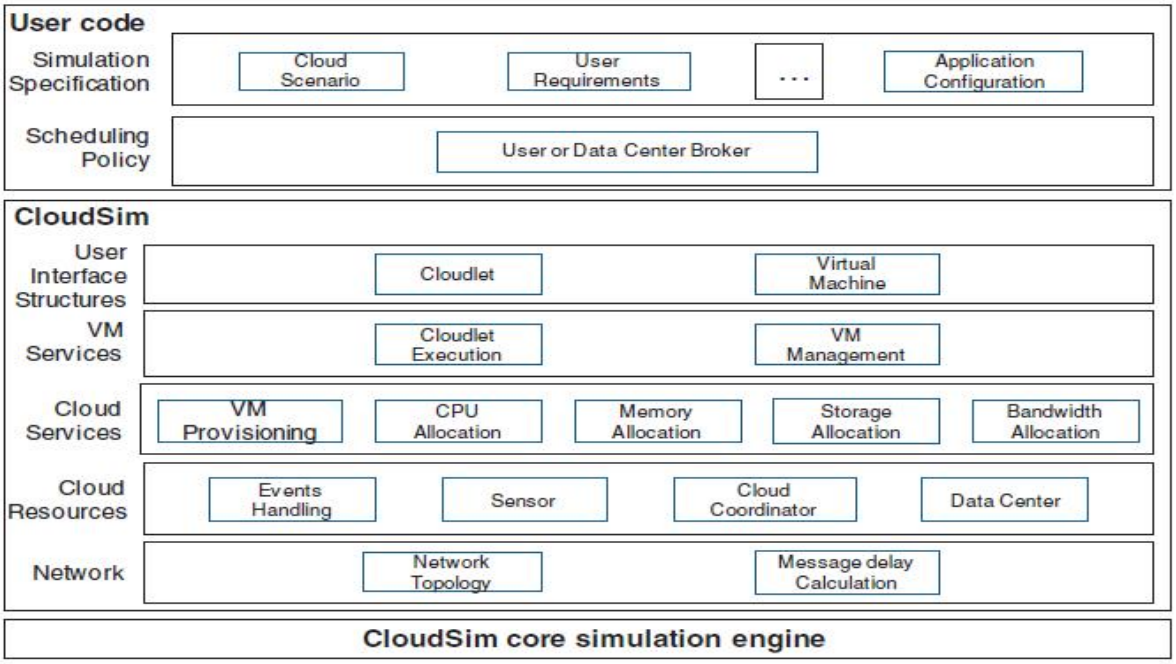
- възможност за симулация на мащабни облакови среди, включително и центрове за данни (datacenters) върху един физически самостоятелен изчислителен възел;

- възможност за симулация на различни по пропускателна способност мрежови връзки;
- възможност за симулация на обединение от облакови среди (федерации от облаци);
- възможност за управление на виртуализацията с цел подпомагане създаването и управлението на множество независими виртуализирани услуги, работещи върху отделни центрове за данни.

4. Архитектура на CloudSim

Симулаторът CloudSim има йерархична структура (фиг. 1), състояща се от три нива [2]:

- Ядро, отговарящо за обработката на събития, както и за създаването на cloudsim обекти като хостове, виртуални машини, центрове за данни, брокери и др.
- Симулационно ниво, осигуряващо възможност за моделиране и симулация на облаково базиран център за данни. Нивото включва специализирани интерфейси за осигуряването на виртуалните машини с ресурси като: процесор, RAM памет, съхранение на данни и пропускателната способност. Този слой управлява изпълнението на приложенията и следи състоянието на системата. Нивото е изградено от 5 йерархични слоя:
  - управление на мрежата – представя топологията на мрежата, както и закъсненията на съобщенията (message delay);
  - управление на облаковите ресурси – съдържа компоненти за: моделиране на услугите на ниво инфраструктура (Datacenter), наблюдение на вътрешното състояние на ресурсите в центъра за данни, въз основа на което взема решения за балансиране на натоварването (CloudCoordinator);
  - управление на облаковите услуги – съдържа компоненти за: осигуряване на хостовете с виртуални машини; разпределяне на ресурси като процесор, памет, пропускателна способност. Тук е възможна реализация на техники и различни алгоритми за осигуряване и разпределяне на ресурси;
  - управление на виртуалните машини – включва класове за управление на виртуалните машини и потребителските задачи (cloudlets);
  - потребителски интерфейс – предоставя интерфейс за използване на конфигурираните виртуални машини и изпълнение на задачите.
- Потребителско ниво - дава възможност за промяна на параметрите на основните обекти: хостове (брой на сървърите и техните характеристики), виртуални машини, потребители, политики за планиране. Потребителското ниво предоставя на потребителите възможности за: генериране на нови конфигурации на приложенията; изпълняване на тестове върху облаковата среда, базирани на потребителски конфигурации; сравнение и оценка на техники за разпределяне на ресурси за облаци и обединени облакови среди.



Фиг. 1. Структура на симулатора CloudSim [2], [3]

Услугите на ниво инфраструктура (IaaS) могат да бъдат симулирани чрез обекта DataCenter (център за данни) от симулационния слой на CloudSim. Този обект управлява други обекти, наречени хостове. Към хостовете са назначени една или повече виртуални машини (ВМ) на основа на политиката за разпределяне на хардуерни ресурси. В CloudSim обектите представляват инстанции на компоненти. Един CloudSim компонент може да бъде клас или набор от класове, представляващи CloudSim модел (център за данни, хост). Хостът е CloudSim компонент, който представя физически сървър в даден облак. Хост компонентът изпълнява интерфейси, позволяващи симулация на разпределени системи.

Разпределянето на ресурсите на ВМ (VM allocation) е процес на създаване на инстанции на виртуалната машина върху хостове, които отговарят на характеристиките, конфигурациите и изискванията на доставчиците на облаковата услуга с цел постигане на определени нива на качество на услугата (QoS). За заделянето на ресурсите е отговорен компонентът VmAllocationPolicy. По подразбиране VmAllocationPolicy прилага метод за разпределяне First-Come-First-Serve (FCFS), при който първият пристигнал клиент се обслужва пръв.

5. Използване на CloudSim и получени резултати

Симулаторът CloudSim е използван за анализ на услугите в разпределена система, изграждана от катедра КНТ, за поддържане на учебния процес и предоставяне на различни други образователни услуги. Разпределената система се състои от блейд сървъри, като всеки от тях има следните характеристики:

- 2x CPU Intel XEON E5-2600v2 6 core 2 GHz;
- 32GB оперативна памет;
- 2 x HDD 146GB 15K rpm;
- 10Gb мрежов адаптер.

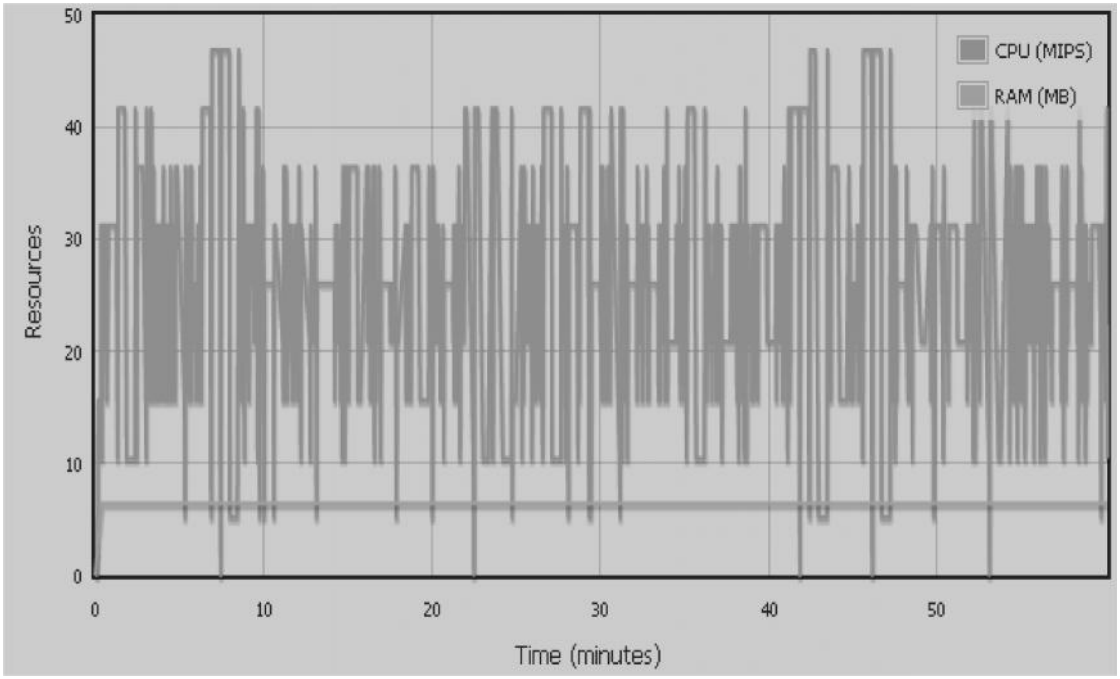
Тези характеристики на реалната инфраструктура трябва да се задават като параметри в симулационната среда. По този начин се получава възможност за анализ и сравнение на получените резултати с реалната разпределена система. За целта симулационният модел се базира на конфигурация от два центъра за данни, всеки от които съдържа два сървъра

(хостове) с параметрите на реалните блейд сървъри. Центровете за данни се използват от клиент (симулиращ работата на студент от произволна учебна лаборатория на катедра КНТ), който изпраща за изпълнение задачи (cloudlets) към виртуалните машини, изпълнявани от тях. Тези задачи представляват модел на използване на облаково-базирана приложна услуга и се характеризират с предварително определени размер, количество данни за трансфер и дължина на инструкциите (mips). В CloudSim задачите могат да използват изчислителните ресурси на виртуалните машини по една от съществуващите три политики за планиране. Това са:

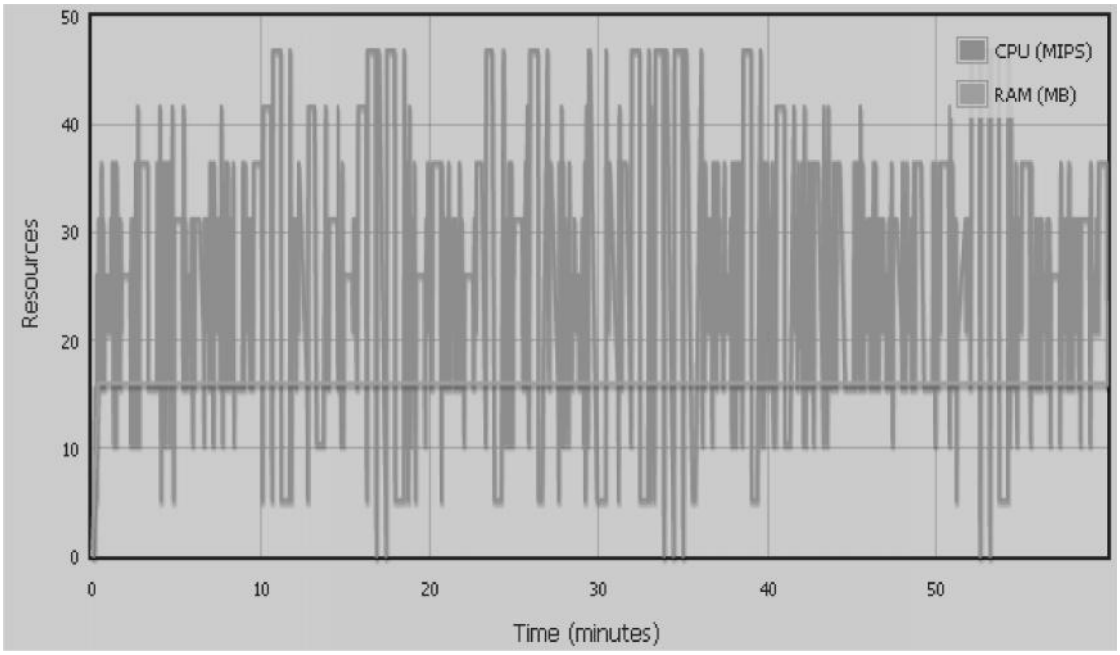
- space-shared – виртуалната машина изпълнява задачите една след друга, като при изпълнението на дадена задача останалите са в режим на изчакване;
- time-shared – виртуалната машина изпълнява няколко задачи едновременно, при което всяка задача използва виртуалната машина за определен период от време, след което достъп до ресурси се дава на следващата задача. Докато дадена задача използва ВМ, останалите са в режим на изчакване;
- dynamic workload – динамично използване на ресурсите на виртуалните машини. При тази политика на всяка задача се предлага ВМ с налични свободни ресурси.

В двата центъра за данни са стартирани общо 20 виртуални машини, конфигурирани с едноядрени процесори, производителност 1000 mips и 512MB оперативна памет. Симулационният тест е с продължителност 60 мин. За целите на симулацията са използвани и трите политики за планиране на ресурсите, като динамичното планиране (dynamic workload) даде най-добър резултат по-отношение на равномерното използване на ресурсите и натоварване на виртуалните машини.

Получените резултати за използването от двата дейта центъра на ресурси са показани на фигура 2 и фигура 3. За извеждането на резултатите е използван инструментът CloudReports. Това е графичен инструмент с отворен код, който симулира разпределени изчислителни среди, използващ ядрото на CloudSim.



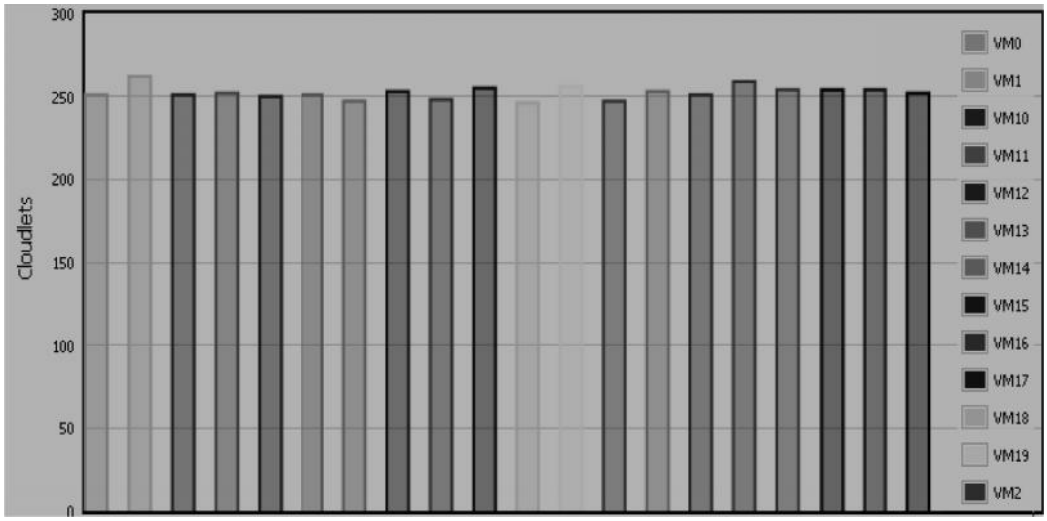
Фиг. 2. Използване на ресурсите в Datacenter1



Фиг. 3. Използване на ресурсите в Datacenter1

Получените резултати показват следното:

- за 60 минути ВМ със зададените характеристики са успели да обработят над 5000 задачи (cloudlets), при динамично планиране на ресурсите на виртуалните машини;
- политиката за динамично планиране на ресурсите дава най-добри резултати по отношение на равномерното натоварване на сървърите и ВМ;
- разпределението на обработените задачи показва изпълнение средно по 250 задачи от една ВМ, което е доказателство за ефективното разпределяне на задачите от страна на дейта центровете (фигура 4);
- ресурсите, използвани от двата дейта центъра, са натоварени равномерно;
- симулираната инфраструктура има достатъчно свободни ресурси и може да обработва заявки и задачи от много по-голям брой клиенти, както локални така и отдалечени.



Фиг. 4. Разпределение на задачите по виртуални машини

6. Заключение

С увеличаването на популярността и значението на облаковите изчисления изследователи от различни университети и компании доразвиват CloudSim, като създават собствени механизми за предоставяне на ресурси и доставка на услуги и го използват за:

- оценка на алгоритми за разпределение на ресурсите;
- анализ на енергийната ефективност на управлението на центрове за данни;
- оптимизация на облакови среди.

Проведените в настоящата работа тестове показват приложимостта на CloudSim за симулация, свързана с изследване на ефективността на обработка на заявки и задачи от локални клиенти. Като насока за бъдеща работа по симулатора може да се посочи разширяване на функционалността му, свързана с предоставяне на възможност за отчитане на влиянието на реални мрежови връзки и техните параметри.

Литература:

[1]. Mell P., and T. Grance, The NIST Definition of Cloud Computing, NIST Special Publication 800-145, 2011.

[2]. Calheiros R., R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, Software: Practice and Experience (SPE), Volume 41, Number 1, pp: 23-50, ISSN: 0038-0644, Wiley Press, New York, USA, January, 2011.

[3]. Calheiros, R., R. Ranjan, C. A. F. DeRose, and R. Buyya, Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services, 2009.

За контакти:
докторант инж. Деян П. Атанасов
катедра ”Компютърни науки и технологии”
Технически университет – Варна
dido.paraskevov@tu-varna.bg

доц. д-р инж. Трифон И. Русков
катедра ”Компютърни науки и технологии”
Технически университет – Варна
ruskov@tu-varna.bg

СЕКЦИЯ 5
СТУДЕНТСКА СЕСИЯ

тора научна конференция с международно участие
"Компютърни науки и технологии"
26-27 септември, 2014 г.
Варна, България

Second Scientific International Conference
Computer Sciences and Engineering
26-27 September, 2014
Varna, Bulgaria

SECTION 5
STUDENT SESSION

СИНТЕЗ НА КАСКАДНА ЛОГИЧЕСКА СХЕМА ЗА ОПРЕДЕЛЯНЕ БРОЯ НА СТАРШИТЕ НЕЗНАЧЕЩИ ЦИФРИ В РАЗРЯДНА МРЕЖА С ПРОИЗВОЛНА ДЪЛЖИНА

Симона С. Стоянова, Димитър С. Тянев

Резюме: Синтезирана е логическа схема, която определя броя на старшите незначещи цифри на число, представено в разрядна мрежа с произволна дължина. Съдържанието на разрядната мрежа може да бъде интерпретирано по различни начини – като число със знак в прав или в допълнителен код и като дробно число. Това дава възможност схемата да бъде използвана както в устройства за работа с фиксирана запетая, така и с плаваща запетая. Броят на старшите незначещи цифри в числото е необходим за изпълнение на следващо високо производително еднотактно изместване наляво. Тази микрооперация има място в алгоритмите на различни машинни операции, изпълнявани в цифровия процесор. Независимостта на схемата от дължината на разрядната мрежа конструктивно се постига въз основа на принципа на каскадността. Синтезираният градивен елемент решава същата задача и има минимална дължина от 3 бита.

Ключови думи: изместване наляво, незначеща цифра, разрядна мрежа, число.

Syntesis of Cascade Logic Scheme for Number of Senior Non-significant Digits in Bit-set with Common Length Determination

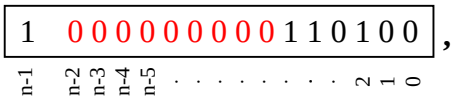
Simona S. Stoyanova, Dimitar S. Tyanev

Abstract: The syntesized logic scheme is capable of determining the number of senior non-significant digits of a binary number represented in bitset with arbitrary length. The content of the bitset can be interpreted in different ways - as signed number, represented in SM, OC and TC or as fractal signed binary number. This allows the scheme to be used in devices with fixed-point or floating point. The number of senior non-significant digits in the number is required to perform the following high throughput single-ended shift left. This microoperation takes a place in the algorithms for various machine commands implemented in the digital processor. The independence of the scheme of the length of the bitset is achieved based on the principle of cascadity. The synthesized building block solves the same problem and have a minimum length of 3 bits.

Keywords: shift left, non-significant digit, bit-set, number.

1. Въведение

Една от възможностите за повишаване на производителността на цифровите устройства на микрооперационно ниво се съдържа в микрооперация изместване. В алгоритмите за изпълнение на различни аритметически и логически операции се налага да се извършва изместване на съдържанието на разрядната мрежа. В алгоритъма на операция деление например се налага да се извършва лява нормализация на делимото и делителя [1], [2]. В алгоритмите за обработка на числа с плаваща запетая също се налага да се изпълнява изместване наляво, с цел нормализация. Каквато и да е причината за лява нормалност на съдържанието на разрядната мрежа, става дума за изхвърляне на старшите незначещи цифри, допълващи числото отляво. Например, нека разрядната мрежа е 16-битова (n=16) и в нея да е представено в прав код цялото число (-52). Тогава двоичното съдържание на разрядната мрежа ще бъде следното



Както се вижда, между знаковия бит и най-старшата цифра на модула на числото, разрядната мрежа е запълнена с 9 нули, които са незначещи за числото цифри. Лявата нормализация на числото (необходима например в алгоритъма за деление) може да бъде постигната след 9 едноразрядни измествания наляво

1 1101000000000000

Достигането на този вид на числото чрез 9 последователни едноразрядни измествания наляво е достатъчно бавен процес, което е причина за търсене на високо скоростно еднотактно изместване. Броят k на изместванията в общия случай, за n-битова разрядна мрежа, съдържаща число със знак, се променя в интервала

$$k \in [0, (n\text{-}2)] . \qquad (1)$$

За горния пример k=9.

Еднотактно изместване наляво може да се реализира чрез програмируема изместваща матрица [1]. Параметърът, с който тази схема се инициализира (програмира), е числото k. Така възниква задачата за автоматично формиране на стойността на този параметър. Предвид липсата в публичното пространство на литературни източници за решаване на тази задача, тя се явява актуална и нейното решение е цел на настоящото изследване.

2. Съображения относно постановката за синтез на логическата схема

Логическият синтез на схема, която ще формира стойността на числото k, следва да преодолее следните начални проблеми:

1. Стойността на числото k варира в интервала (1). В общият случай k следва да се получи като m-битово двоично число, където

$$m = \lfloor \log_2 (n-2) \rfloor . \qquad (2)$$

Тъй като разрядността m на числото k е функция от дължината на разрядната мрежа n, то следва, че логическата схема за формиране на числото k може да бъде синтезирана само за постоянна дължина на разрядната мрежа. Всяко изменение в дължината на разрядната мрежа, или на формàта за представяне на числото, ще изисква синтез на нова логическа схема.

2. Лявата нормализация е изискване, което се налага върху различни по вид и форма числа, явяващи се в разрядната мрежа – цели без знак, цели със знак, дробни със знак, представени в прав или в допълнителен код. Известно е, че в инверсните машинни кодове (обратен и допълнителен) старшите незначещи цифри съвпадат със знаковата цифра. Известно е още, че старшата значеща цифра в модула на отрицателните числа, представени в обратен и в допълнителен код, може да бъде както нула така и единица [1].

Основният извод, който се налага след изложеното по-горе е, че крайно решение за логическата схема трябва да се търси чрез мултиплициране на една градивна логическа схема, синтезирана за част от разрядната мрежа с минимална дължина. При наличието на такъв градивен елемент с възможност за каскадно свързване, може да бъде реализирана произволна дължина за входния аргумент.

На този етап от анализа на проблема може да бъде приета за минимална дължината от три бита. Дължината от три незначещи цифри ще доведе до оползотворяването на всички възможни комбинации на 2-битовия резултат (k=0,1,2,3), който може да бъде формиран за

нея. Това означава, че анализът на двойно по-дълга част от разрядната мрежа ще изисква конкатениране на два градивни елемента. При това общият резултат ще бъде сумата

$$k = k_i + k_{i-1} \, . \tag{3}$$

където с i е означен номерът на текущия 3-битов градивен елемент.

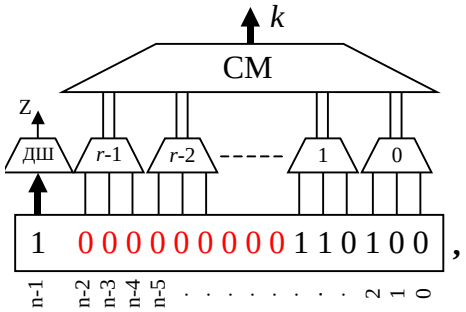
Така окончателната стойност за k може в общия случай да се изрази така

$$k = \sum_{i=0}^{r-1} k_i \, . \tag{4}$$

където с r е означен броят на градивните елементи, покриващи дължината от (n-1) бита

$$r = \left\lceil \frac{n-1}{3} \right\rceil . \tag{5}$$

Имайки предвид, че старшата значеща цифра на числото може да се намира във всяка от позициите вдясно от знаковата, то следва че първият градивен елемент трябва да обработва позициите с номера (n-2), (n-3) и (n-4) (при числа със знак). При такова позициониране знаковият бит не е включен в анализа. От това следва, че всеки следващ градивен елемент трябва да допълва отдясно вече поставените. С други думи, има се предвид, че номерата на градивните елементи ще се променят в низходящ ред отляво надясно (r-1), (r-2), … 2, 1, 0, както е показано на фигура 1.



Фиг. 1. Логическа структура

В логическата структура се вижда, че е включен и двоичен суматор СМ, който реализира сумата (4).

Що се отнася до изменчивата природа на незначещите цифри в двоична бройна система, този проблем може да бъде разрешен от знаковия бит, позовавайки се на изказаната вече зависимост, според която незначещите старши цифри съвпадат със знаковата. Разбира се, необходимо е да се определя мястото на старшата значеща цифра, при това следва да се изключи наличието на пълна нула в разрядната мрежа, тъй като тогава всякакво изместване на съдържанието й, е безсмислено. Нулево съдържание в разрядната мрежа може лесно да бъде дешифрирано и формираният при това признак Z (zero) да преопредели числото k=0, ако е необходимо.

В следствие на описаното по-горе позициониране на градивните елементи на схемата се изявява проблем с най-младшия елемент. Този елемент с №0, както всеки от останалите, има три входа. Възможността обаче те да бъдат реално подключени не е гарантирана, тъй като общата дължина от разрядната мрежа, която трябва да покрие схемата, може да не бъде кратна на 3. При това са възможни три ситуации – остава един бит, два бита или три бита. Последният случай не е проблемен.

Случаят, когато е останал неподключен само един бит, е особен с това, че ако числото в разрядната мрежа е нула, то числото k е без смисъл, а ако съдържанието на разрядната мрежа е различно от нула, то числото k е вече формирано в схемата от предходните градивни

елементи. Ако обаче съдържанието на разрядната мрежа е единица, то следва, че единствената значеща цифра се намира в най-младшия разряд. В този случай за тази група се получава стойността k_0=0. С други думи, в този случай, включването на градивен елемент върху група от един бит, не е необходимо, защото резултатът от нейното функциониране е предопределен. Така общият брой градивни елементи ще бъде (r-1).

В случаят, когато са останали два младши бита, за групата са възможни два резултата:

1. k_0=0, ако в тези два бита има комбинация 10 или 11;
2. k_0=1, ако в тези два бита има комбинация 01 или 00. В последната ситуация признакът Z следва да игнорира изчисляването на числото k. Така реален остава само резултатът k_0=1, съответстващ на комбинация 01. В крайна сметка, логиката на описаните възможности се изразява чрез уравнението

$$k_0 = \overline{b_1}, \tag{6}$$

Тъй като числата със знак се представят в допълнителен или в обратен код, ,. Ако казаното по-горе може да се отнесе до положителните числа, както и до тези със знак представени в прав код, то следва да бъдат разгледани отрицателните, представени в обратен или в допълнителен код. Както е пояснено в пункт 2 на предходния раздел в тези случаи старшите незначещи цифри на числото в разрядната мрежа се определят от знаковата, т.е. те са единици. При позициониране на последния градивен елемент са възможни описаните по-горе ситуации. Ако е останал неподключен само един бит (b_0) (виж фиг.1), то той може да съдържа незначеща или значеща цифра. Анализът на този случай води до аналогичен резултат, формулиран за положителните числа, т.е. k_0=0.

В случаят, когато са останали два младши бита, за групата са възможни следните резултати, които зависят от кода.

Ако кодът е обратен, то

1. k_0=0, ако в тези два бита има комбинация 01 или 00;
2. k_0=1, ако в тези два бита има комбинация 10 или 11. В последната ситуация признакът Z следва да игнорира изчисляването на числото k. Така реален остава само резултатът k_0=1, съответстващ на комбинация 10. Така логиката на описаните възможности се изразява чрез уравнението

$$k_0 = b_1, \tag{7}$$

Ако кодът е допълнителен, то

1) k_0=0, ако в тези два бита има комбинация 10 или 00;
2) k_0=1, ако в тези два бита има комбинация 01 или 11.

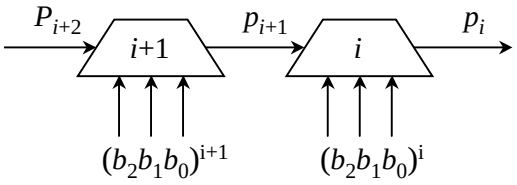
Тук логиката за числото k може да се изрази така

$$k_0 = b_0, \tag{8}$$

3. Синтез на логическата схема на градивния елемент

Както вече беше решено, ще бъде търсено решение, при което логическата схема на градивния елемент ще има три входни аргумента ($b_2b_1b_0$) – битове на i-тата група. Групите от по 3 бита ще се формират както е показано на фиг.1, т.е. отляво надясно, започвайки от бит №(n-2). Покриването на разрядна мрежа с произволна дължина ще става с последователно включване на градивния елемент, както вече беше предложено. При тази постановка всеки елемент има за задача да “открие” старшата значеща единица на съдържащото се в разрядната мрежа число. Ако даден елемент постигне това, работата на стоящите вдясно от него елементи се обезсмисля. Техните резултати трябва да бъдат нула.

Последното означава, че между градивните елементи има функционални връзки, които следва да бъдат изяснени. Всеки градивен елемент има смисъл да анализира групата от разряди, към които е подключен, ако предходния елемент не е открил старша значеща цифра в числото. Така става ясно, че два съседни елемента трябва да бъдат свързани със сигнал *p* (*permission*), който генерира ляво стоящият и се използва от дясно стоящия в качеството на разрешение. Казаното е илюстрирано на фигура 2.



Фиг. 2. Сигнал разрешение

Логиката на сигнала *p* се дефинира като изходна функция на всеки градивен елемент така

$$p_i = \begin{cases} 1, & ako \ \{(b_2b_1b_0) = \bar{s} \cap (000) \ \cup \ (b_2b_1b_0) = s \cap (111)\} \cap p_{i+1}; \\ 0, & ako \ (b_2b_1b_0) \neq (000) \ \cup \ (b_2b_1b_0) \neq (111) \cup \overline{p_{i+1}}. \end{cases}, \quad i \in [(r-1), 0], \qquad (9)$$

при това, когато един градивен елемент получи на вход стойност $p_{i+1} = 0$, той задължително следва да генерира същата изходна стойност $p_i = 0$.

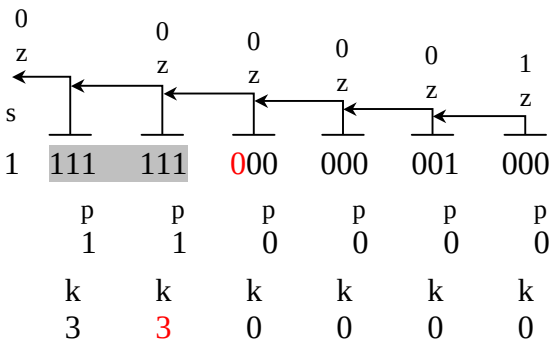
Както се вижда в определението (9), за текущата тройка входни битове са включени две от възможните комбинации - (000) и (111). Става дума за старши незначещи цифри на числа представени в допълнителен код, когато незначещите цифри съвпадат със знаковата.

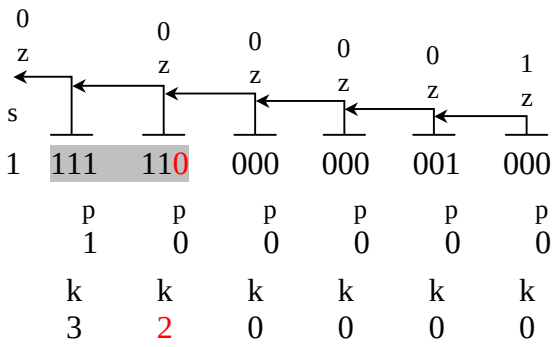
За преодоляване на изключението, което създават отрицателните числа, свързано с променящата се стойност на старшата значеща цифра, е направен следният анализ.

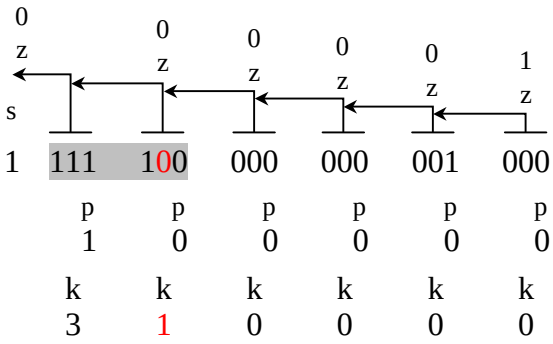
Разгледани са примери с отрицателни числа в допълнителен код, със старша значеща цифра нула и със страша значеща цифра единица, при това намираща се в различни позиции на текущата тройка битове. Примерите показват, че разпознаването на старшата значеща цифра като нула или като единица, е функция от комбинацията на дясно стоящите битове. За целта за всяка тройка битове е въведена функцията *z* (*zero*). Стойностите на тези функции се умножават логически отдясно наляво. В представените примери са показани стойностите на функциите *z*, *p* и *k*, генерирани за всяка тройка битове от примерното съдържание на разрядната мрежа. Първите три примера илюстрират отрицателно число в допълнителен код със старша значеща цифра нула

(1 1111110000000001000),
(1 1111100000000001000),
(1 1111000000000001000),

намираща се в различни позиции спрямо текущата тройка битове.

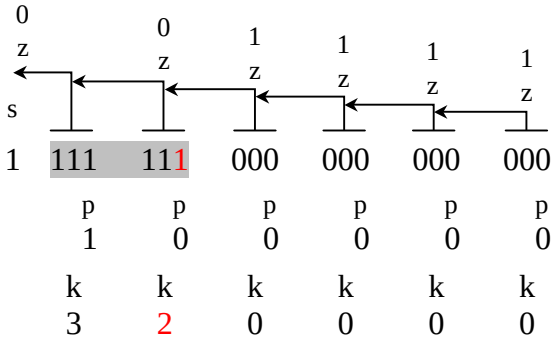


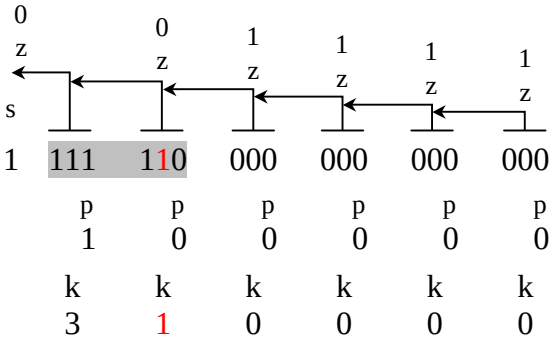


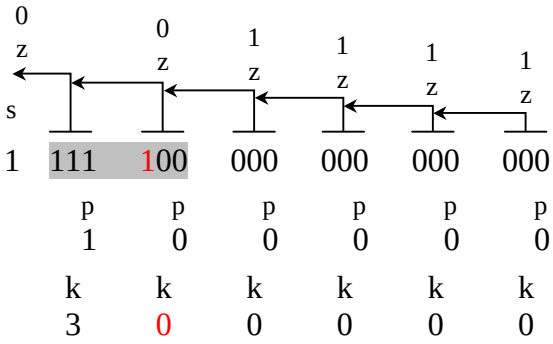


Следващите три примера илюстрират отрицателно число в допълнителен код със старша значеща цифра единица, намираща се в различни позиции на текущата тройка битове.

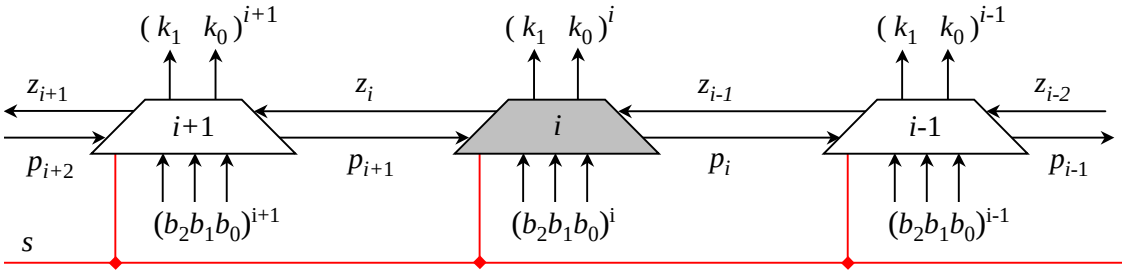
(1 111111000000000000),
(1 111110000000000000),
(1 111100000000000000),







За разпознаване на двойствеността на старшата значеща цифра в отрицателни числа спомага функция *z*, която дешифрира нулева комбинация в текущата 3-битова група. Последователното в посока към старшите разряди каскадно умножение на стойностите на функциите *z* достига със стойност нула групата, в която се намира старшата значеща нула, и достига съответно със стойност единица групата, в която се намира старшата значеща единица. В резултат на този анализ следва, че всяка градивна схема се “атакува” отдясно от функция *z*, а отляво от функция *p*. Освен това всяка група се инициализира от функцията на знака на числото *s*. Така логическите аргументи на 3-битовата градивна схема могат да се илюстрират чрез структурата от фигура 3.



Фиг. 3. Аргументи на градивната схема

След така изяснената функционалност логическата схема на градивния елемент може да бъде синтезирана от съставената за целта непълна таблица на истинност.

Таблица 1. Таблица на истинност

	Вход елемент №(i)						Изход елемент №(i)			
№	p_{i+1}	s	b_2	b_1	b_0	z_{i-1}	k_1	k_0	z_i	p_i
1	0	*	*	*	*	*	0	0	0	0
2	1	0	0	0	0	*	1	1	0	1
3	1	0	0	0	1	*	1	0	0	0
4	1	0	0	1	*	*	0	1	0	0
5	1	0	1	*	*	*	0	0	0	0
6	1	1	0	0	0	1	0	0	1	0
7	1	1	0	*	*	0	0	0	0	0
8	1	1	1	0	*	0	0	1	0	0
9	1	1	1	0	*	1	0	0	0	0
10	1	1	1	1	0	0	1	0	0	0
11	1	1	1	1	0	1	0	1	0	0
12	1	1	1	1	1	0	1	1	0	1
13	1	1	1	1	1	1	1	0	0	1

Тъй като таблицата на истинност е непълна, тя се нуждае от следните пояснения:

1. Когато към схемата на градивния елемент постъпва сигнал p_{i+1}=0 (забрана за функциониране), на всички изходи схемата генерира логическа нула (p_i=0, z_i=0, k_1=k_0=0). Всички подобни случаи са обобщени в ред №1 на таблицата. Следващите редове в таблицата пояснявят случаите, в които към схемата на градивния елемент постъпва сигнал разрешение p_{i+1}=1.
2. Редове №2, 3, 4 и 5 обобщават случаите с разрешение (p_{i+1}=1) при положителни числа (s=0). В ред №2 е представена входна комбинация $b_2b_1b_0$ без значеща цифра (000), а в следващите три ред – комбинации със старша значеща цифра в три различни позиции на групата (001, 01*, 1**). Необходимо е да се поясни, че при положителни числа, функциите z_i=0, тъй като нямат съответния смисъл.
3. От ред №6 до ред №13 са обобщени всички случаи на отрицателни числа. За тези числа функциите z_i имат описания по-горе смисъл. Ако входната комбинация е (000) (ред №6) и z_{i-1}=0, това означава, че отдясно на текущата тройка има само нули и тази група следва да генерира само z_i=1.
4. На изход p_i се генерира единица само в два случая:
   - При положителни числа (s=0) при входна комбинация (000), съответстваща на три незначещи цифри, без значение на стойността на z_{i-1};

- При отрицателни числа (s=1), при входна комбинация (111), съответстваща на три незначещи цифри, без значение на стойността на z_{i-1}.

По-горе представената в табличен вид логика се изразява чрез следната система логически функции:

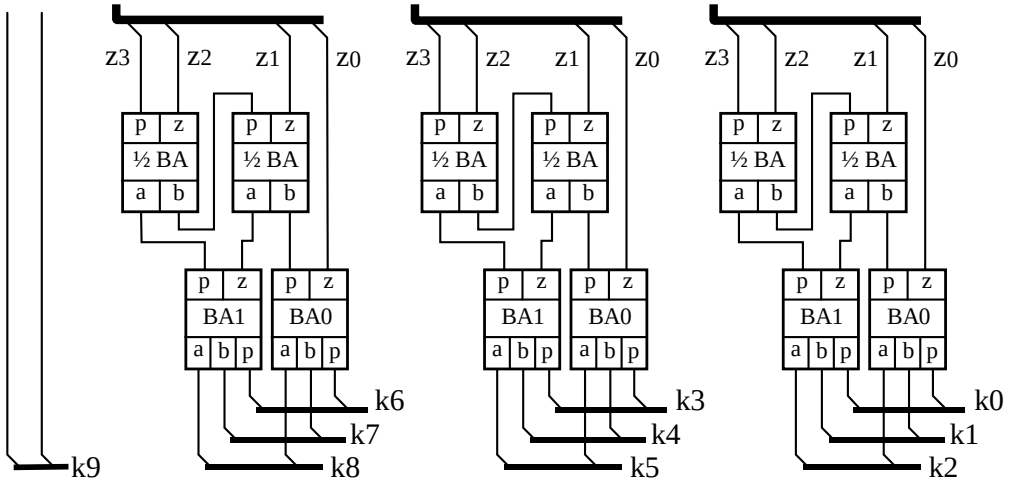
$$\begin{cases} p_i = p_{i+1} \cap \left((\bar{s} \cap \overline{b_2} \cap \overline{b_1} \cap \overline{b_0}) \ \cup \ (s \cap b_2 \cap b_1 \cap b_0) \right) ; \\ z_i = s \cap p_{i+1} \cap z_{i-1} \cap \left(\overline{b_2} \cap \overline{b_1} \cap \overline{b_0} \right) ; \\ k_0 = p_{i+1} \cap \left[\bar{s} \cap \overline{b_2} \cap \left(b_1 \cup \overline{b_0} \right) \cup \ s \cap b_2 \cap \left((\overline{b_1} \cup b_0) \cap \overline{z_{i-1}} \ \cup \ b_1 \cap \overline{b_0} \cap z_{i-1} \right) \right] ; \\ k_1 = p_{i+1} \cap \left[\bar{s} \cap \overline{b_2} \cap \overline{b_1} \ \cup \ s \cap b_2 \cap b_1 \cap (\overline{z_{i-1}} \cup b_0) \right] . \end{cases} \quad (10)$$

4. Каскадност на суматора

Според фигура 1 резултатът, който схемата трябва да получи, е сумата (4). Събираемите k_i в нея са 2-битови числа. Броят r на тези събираеми се определя според (5). С цел минимална латентност на изчислението на числото k, и предвид едновременното формиране на отделните събираеми k_i, събирането следва да се осъществи паралелно.

Изследването на задачата за паралелно събиране на повече от две цели числа е представено в [4], а в [5] са представени допълненията към задачата и нейното обобщение. Логическата структура на суматорът (фигура 1) е йерархична и се синтезира въз основа на тривходови концентратори. Йерархичната структура се определя от дължината на крайния резултат. Максимално възможната дължина на сумата се дефинира от обща теорема, доказана в [5]. Ако за числата бъде приета типичната дължина от 32 бита, тогава числото k ще има максимална дължина от 5 бита. Върху разрядите от №(n-2) до №(1) следва да се разположат 10 градивни елемента, чиято логическа схема реализира системата (10). Тези елементи трябва да се свържат по начина, показан на фигура 3. Всяка от тези логически схеми ще формира 2-битово число k_i. Десетте двубитови числа подлежат на събиране.

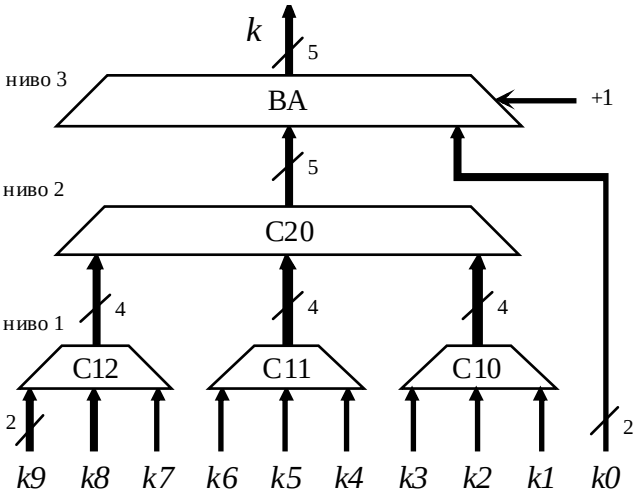
Първото ниво на суматора (за предложената примерна дължина на разрядната мрежа) се реализира чрез логическата схема, показана на фигура 4.



Фиг. 4. Първо ниво на суматора (3 3-входови концентратора за 2-битови числа)

Както се вижда, всеки от тривходовите концентратори е реализиран от два пълни суматора BA (*Binari Adder*) и два полусуматора ½BA. При това разпределение остава едо от числата k_9, което следва да се подключи към по-високите нива на суматора. Всеки

концентратор изчислява 4-битова сума ($z_3z_2z_1z_0$). Методиката, по която се изграждат следващите нива на паралелния суматор, изисква разпределение на тези 3 числа плюс числото k_9, към 3 тривходови 4-битови концентратора. Логическата схема на тези конценратори е представена в [4] и тук не се коментира. Концентраторите на второто ниво ще получат 5-битова сума. Към получената 5-битова сума двоичният суматор ВА добавя останалото число k_9. Така се получава логическата структура на паралелния суматор, която е илюстрирана на фигура 5. За удобство на схемата оставеното число е k_0.



Фиг. 5. Логическа структура за паралелното събиране

На трето ниво в структурата е поставен двоичен суматор. Необходимо е да се каже, че неговата логическа схема мпже да бъде значително опростена, ако се отчете фактът, че към 5-битовата сума от второ ниво, той трябва да прибави едно 2-битово число. Това позволява старшите три бита да бъда реализирани чрез полусуматори, както това е направено в схемите от ниво 1.

5. Заключение

Със синтезираната логическа схема на 3-битовия градивен елемент (10) и на хоризонталния паралелен концентратор (фиг.5), както и с методиката за произволни дължини на разрядната мрежа, представена в раздел 2, поставената тук цел се приема за постигната. Разбира се, отделните решения не са единствени и обект на изследване могат да бъдат и други различни предложения. В крайна сметка получената схема определя броя на старшите незначещи цифри в цяло число, представено в прав или в допълнителен код. В случай, че се иска числото в разрядната мрежа да се интерпретира като число без знак, това означава, че възможните изместванията наляво могат да се увеличат с единица. Този случай може да се реализира чрез управляващ сигнал,

$$УС(+1) = \bar{s} \ , \tag{11}$$

където с s е означена двоичната цифра в най-старшия бит. Така той ще прибави единица към числото k. Сигналът $УС(+1)$ се подава по линията на преноса в най-младшия бит на изходния суматор, както е показано на фиг.5.

В случай, че схемата се прилага за лява нормализация на мантисата в разрядна мрежа с плаваща запетая, след изместване на мантисата на k бита наляво, е необходимо да се коригира стойността на характеристиката [1]. Корекцията трябва да реализира следната операция

$$p_k = p - k \tag{12}$$

Полученото число k се използва в програмируемите изместващи матрици, които по този начин реализират еднотактно изместване на съответния брой разряди.

На въпроса, колко все пак се ускорява изчислителният процес след прилагане на описаното еднотактно изместване наляво, може да се отговори по следния начин. Появата на незначещи цифри в старшата част на числата е обективно явление. Броят на тези цифри обаче е непредвидим, което означава, че числото k има случаен характер. За да може да се даде приблизителна числена оценка на поставения по-горе въпрос, трябва да се знае закона за разпределение на плътността на тази случайна величина. Тук се предполага, че в първо приближение е възможно да се приеме за явяващите се в разрядната мрежа числа равномерен закон на разпределение. Тъй като числата всъщност изразяват дължината на интервала, който заемат в разрядната мрежа, това означава, че той също е случайна величина. Дължината на числото в битове варира в интервала [0, (n-2)] като не се имат предвид знаковия и най-младшия битове. Така най-вероятната дължина на числото в разрядната мрежа се оценява както следва

$$m = \frac{(n-2)-0}{2} = \frac{n-2}{2} \quad . \tag{13}$$

В същия интервал варира и числото k. Тогава неговата най-вероятна дължина се оценява аналогично за интервала [0, $\frac{n-2}{2}$] както следва

$$m_k = \frac{\frac{n-2}{2}-0}{2} = \frac{n-2}{4} \quad . \tag{14}$$

За 32 битова разрядна мрежа тази оценка означава, че средният брой измствания наляво ще бъде 7 бита. От своя страна тази стойност позволява да се твърди, че схемата за еднотактно изместване, води най-вероятно до седемкратно ускоряване на тази микрооперация.

На практика законът за разпределение на дължината на числата, явяващи се в разрядната мрежа, а оттам и на дължината на интервала, запълнен с незначещи цифри, може да бъде различен. Това зависи от вида на операциите, в резултат на които се получават числата по време на изчисления.

Литература

[1]. Тянев, Д.С., Организация на компютъра, том 1 и том 2, ISBN: 978-954-20-0412-7, ISBN 978-954-20-0413-4, ТУ - Варна, 2008 год.

[2]. Тянев, Д.С., Организация на компютъра – упражнения, ISBN 954:-20-0258-0, ТУ - Варна, 2007 год.

[3]. Тянев, Д.С., Организация на компютъра – проектиране на логически структури, ISBN: 954-20-0259-9, ТУ - Варна, 2004 год.

[4]. Tyanev D.S., Nikolov N.N., Popova S.I., Synthesis and comparative analysis of multiple inputs parallel adders, Fourth International Bulgarian-Greek Conference – “Computer Science’2008”, 18-19 September 2008, Kavala, Greece. Vol. 1, ISBN: 978-954-580-255-3, pp. 270-278.

[5]. Тянев, Д.С., Синтез на асинхронни микроконвейерни системи с обща структура, Дисертация за научната степен “доктор на науките”, Глава 4, IX.2012.

За контакти:

Симона С. Стоянова, ТУ-Варна, спец. “СИТ”
e-mail: simonita_s_s@abv.bg

Димитър С. Тянев, доц. д.н. инж.,
ТУ-Варна, www.tyanev.com .

РАЗРАБОТВАНЕ И ИЗСЛЕДВАНЕ НА ХИБРИДНА КРИПТОСИСТЕМА ЗА ЗАЩИТА НА КРАТКИ СЪОБЩЕНИЯ

Мюслюм И. Вели, Юлка П. Петкова

Резюме: Сигурността е една от най-важните характеристики на всяка съвременна компютърно-комуникационна система. Защитата на процеса на предаване на данни изисква най-сериозно внимание, тъй като тя обхваща най-уязвимите и достъпни за нарушения точки. В този доклад се предлага една хибридна схема за защита от несанкциониран достъп върху предаваните по комуникационен канал кратки съобщения.
Ключови думи: сигурност, защита на данни, хибридна криптосистема, криптиране, изпращане на криптирани съобщения, сигурен криптопротокол, автентикация, шифриране, дешифриране.

Research and Development of Hybrid Cryptosystem for Protection of Short Messages

Myuslyum I. Veli, Yulka P. Petkova

Abstract: Security is one of the most important features of any modern computer-communication system. Protection of the process of data transmission requires the serious attention, as it covers the most vulnerable and accessible violations points. This report proposes a hybrid scheme to protect against unauthorized access to the transmitted over a communication channel short messages.
Keywords: security, data protection, hybrid cryptosystem, encryption, decryption, send encrypted data, secure crypto protocol, authentication, ciphering, deciphering.

1. Въведение

С бързото развитие и навлизане на компютърните комуникации във всички сфери на живота, мерките за защита, както срещу непреднамерени грешки, така и срещу преднамерените целенасочени въздействия стават все по-съществени, в следствие на инкасираните щети.

Обезпечаването на достатъчно сигурна защита срещу несанкциониран достъп в компютърните комуникации е сложна и широкообхватна задача, която се решава с помощта на комплекс от мерки с организационен и програмно-технически характер, които могат да бъдат обособени в следните основни групи:

- ✓ организация на изграждането, управлението и контрола на системата за защита;
- ✓ физическа и техническа защита на съоръженията и ресурсите на компютърно-комуникационните системи;
- ✓ защита на процеса на предаване на данни;
- ✓ автентикация на субектите (оператори и процеси);
- ✓ управление на достъпа до ресурсите на системите.

В случая преднамерените действия на евентуалните нарушители могат да бъдат както пасивни, така и активни.

При пасивни действия нарушителят само следи за предаваните данни, без да нарушава самия процес на предаване, като се старае да разбере съдържанието на данните (release of data contents) или да разкрие данновия поток (release of data stream) [1], [3].

Разкриването на съдържанието на данните е най-често срещаното преднамерено въздействие, което по същество води до нарушаване на анонимността на данновия обмен.

Активният нарушител може да осъществи различни преобразувания върху данните, например: модифициране, забавяне за определено време, дублиране, унищожаване, вмъкване на лъжливи данни, промяна на реда на следване. Тези преобразувания се обозначават общо като въздействия върху данновия поток или нарушение на целостта на потока [1], [3].

За да бъде защитена предаваната информация по комуникационен канал, както и да се гарантира за самоличността на източника на съобщението, е необходимо данните да бъдат закодирани (маскирани, криптирани) преди тяхното изпращане, впоследствие след получаването им да бъдат възстановени.

Защитата на процеса на предаване на данни, заедно с процеса на автентикация изискват най-сериозно внимание, тъй като те обхващат най-уязвимите и достъпни за нарушения точки в една компютърно-комуникационна система.

Прилагането дори и на най-сигурните криптографски алгоритми може да се окаже недостатъчно за осигуряване на конфиденциалност на разговора, ако същите не се приложат правилно, в строго определена последователност и/или неправилен избор на криптографските алгоритми.

Съвременните криптографски алгоритми са достатъчно сигурни, осигуряващи висока степен на конфиденциалност, което означава, че надеждността на криптосистемата ще зависи изцяло от начина, по който са проектирани криптографските алгоритми и най-вече от криптопротокола на системата.

За да бъдат максимално ефективни и в същото време с висока степен на надеждност, се прилагат криптопротоколи, обединяващи възможностите на няколко криптографски алгоритъма в единна криптосистема – хибридна [1], [2], [3], [4].

В настоящия доклад се предлага един вариант на хибридна криптосистема за надеждна защита на съобщения, предавани по комуникационен канал, както и изследване на начините за противодействие срещу нарушителите.

2. Предложение за сигурен криптопротокол в хибридна криптосистема

За да се осигури конфиденциалност на една криптосистема, използваща симетрични алгоритми за криптиране на открито (незащитено) съобщение, е необходим защитен канал за предаване на ключа, като единственото изискване е ключът да се пази в тайна. Именно това се явява и основният недостатък на симетричната криптография.

От друга страна публичните (асиметрични) криптографски алгоритми използват двойка от ключове (публичен и частен) за процедурите криптиране и декриптиране, както и за електронен подпис. Предимството на асиметричната криптография е, че не се изисква източникът и получателят да имат споделена тайна помежду си (таен, секретен ключ), за да си комуникират секретно. На практика асиметричните алгоритми решават основния недостатък на симетричните алгортми. Недостатъкът обаче на публичната криптография е, че алгоритмите се базират на математически изчисления с висока сложност, поради много дългите ключове, с които се извършват изчисленията, а това ги прави ресурсоемки.

Комбинирайки положителните качества на симетричните и асиметричните алгоритми, а именно ефективността на алгоритмите с таен ключ и удобството на публичните алгоритми, като резултат се получава комбинирана криптосистема, наричана още хибридна криптографска схема.

При хибридните криптографски системи данните (откритият текст) се криптират със симетричен алгоритъм, а тайният ключ – с асиметричен алгоритъм. На този принцип са създадени протоколи като HTTPS в комбинация с SSL/TLS, IPSec, Skype и криптосистеми като PGP, TIS/PEM и RIPEM.

Най-разпространените криптопротоколи са стандартните Encrypt-then-Sign и Sign-then-Encrypt. Първият от тях използва механизъм, при който първоначално се шифрират данните

с някакъв симетричен алгоритъм, след което се прилага електронен подпис с помощта на асиметричен алгоритъм, като секретният ключ се шифрира също с асиметричен алгоритъм. Вторият механизъм за осигуряване на защита на данните първоначално извършва операцията подписване, впоследствие и криптиране на данните и секретният ключ [3].

Стандартните криптопротоколи, без значение от използваните за реализацията им криптографски алгоритми, въпреки че поставят електронен подпис, не могат да гарантират самоличността на източника, поради това, че алгоритъмът за електронен подпис сам по себе си не може да гарантира за едно от най-важните качества – автентичност на подателя. Стандартните криптопротоколи също така са уязвими и на атаки от вида „нелегално препращане на съобщението“ и „замяна на електронен подпис“. Въпреки тези недостатъци се продължава да се разчита на тези криптопротоколи.

За да се избегнат горепосочените криптоатаки и недостатъци на стандартните криптопротоколи, в настоящият доклад се предлага хибридна криптосистема със семантично сигурен криптопротокол – Sign-then-Encrypt-then-Sign, като това не е за сметка на производителността на системата.

Този криптопротокол в хибридната криптосистема изисква задължително следните действия, в строго определена последователност, без значение от използваните криптографски алгоритми и начина им на реализация:

1. Подписване на съобщението от страна на източника чрез асиметричен алгоритъм;
2. Криптиране на откритото съобщение с помощта на симетричен алгоритъм;
3. Криптиране на сесийният(временен) ключ чрез асиметричен алгоритъм;
4. Подписване на криптирания сесиен ключ отново чрез асиметричен алгоритъм.

Вътрешният подпис осигурява невъзможност за отричане на авторство (стъпка 1) и интегритет на съобщението, криптирането на съобщението и на сесийния секретен ключ (стъпки 2 и 3) осигуряват конфиденциалност на разговора, а външният подпис (стъпка 4) – предоставя автентичност на подателя.

В качеството на криптографски алгоритми при проектиране на хибридната криптосистема със семантично сигурен криптопротокол са реализирани издържливи на криптоатаки алгоритми. За шифриране на открития текст (съобщение) е предназначен симетричният алгоритъм AES (Advanced Encryption Standard). Криптирането на ключовете на симетричният алгоритъм се извършва от RSA (Rivest, Shamir и Adleman), а за нанасяне на електронен подпис върху съобщението и криптирания симетричен ключ е реализиран алгоритъмът за електронни подписи DSA (Digital Signature Algorithm).

AES е бърз и компактен алгоритъм, с проста математическа структура, благодарение на която той се оказва лесен за анализ при оценката на нивата на защита.

Алгоритъмът демонстрира много добра устойчивост на атаки, при които хакерът се опитва да декодира шифрираното съобщение, като анализира външните му проявления, включително и време за изпълнение. В стандарта AES е определена дължината на блока от данни, равна на 128. Един 128-битов (16-байтов) ключ осигурява $340*10^{36}$ възможни комбинации, а 256-битовият (32-байтов) ключ увеличава това число на $11*10^{76}$. Промеждутъчните (междинните) резултати от преобразуванията, които се изпълняват в рамките на криптоалгоритъма, се наричат състояния (State). Състоянието може да се представи във вид на матрица от байтове. При размер на блока, равен на 128-бита, този 16-байтов масив е показан в таблица 1 [1], [4].

Ключът също се представя във вид на матрица, което до голяма степен улеснява проектирането и същевременно повишава производителността. При AES, за да се криптира един блок от данни, се извършват няколко операции: размяна на байтовете (субституция), разбъркване по редове, разбъркване по колони и добавяне на ключа (сума по модул 2), като при това тези операции се прилагат между 10 и 12 пъти, което внася изключително високо ниво на разсейване на входния блок и получаване на напълно случайни резултати след

криптиране. Дешифрирането представлява прилагане на същите операции при криптиране, но в обратен ред.

Таблица 1. Състояния при AES алгоритъм

S_{00}	S_{01}	S_{02}	S_{03}
S_{10}	S_{11}	S_{12}	S_{13}
S_{20}	s_{21}	S_{22}	S_{23}
S_{30}	s_{31}	S_{32}	S_{33}

Реализираният асиметричен алгоритъм RSA е предназначен за защитено предаване на симетричния ключ от източника към получателя. RSA разчита на една трудна за решаване математическа задача – намирането на прости множители на едно голямо число.

Описанието на RSA е много просто. Първоначално всеки получател си избира две достатъчно големи прости числа "a" и "b" (с над 100 десетични цифри), след което се изчисляват следните произведения [1], [2]:

$$n = a * b \tag{1}$$

$$\phi(n) = (a-1) * (b-1) \tag{2}$$

Където с формула (2) е обозначена функцията на Ойлер. Публичният ключ $\mathbf{K_P}$ и секретният ключ $\mathbf{K_S}$ от двойката "публичен/секретен" ключове на дадения получател се определят от съотношенията [1], [2]:

$$\text{НОД}\,[K_P, \Phi(n)] = 1 \tag{3}$$

$$(K_S * K_P) \bmod \Phi(n) = 1 \tag{4}$$

С НОД във формула (3) е обозначена функцията най-голям общ делител на числата в скобите. Обикновено публичният ключ е случайно генериран и на негова база се изчислява и секретният ключ. По-нататък за шифриране се използва публичният ключ заедно с произведението от (1), а за дешифриране – частният ключ заедно с произведението от формула (1). След генериране на двойката ключове се унищожават данните за генерираните числа "a" и "b", както и стойността на функцията на Ойлер - формула (2).

Електронен подпис – представлява математическа схема за автентификация на изпратено електронно съобщение, удостоверява самоличността на изпратилия съобщението (non-repudiation), както и гарантира целостта на данните (информацията не е била променяна между изпращането и получаването ѝ). За подписването се използват хеш-функции и алгоритъм с публичен ключ. Като подписващият използва своя частен ключ от двойката ключове, за да се подпише, а верификацията се извършва с публичния ключ на източника [1], [5].

В настоящата криптосистема за подписване е реализиран алгоритъмът за цифрови подписи – DSA. Той не е предназначен за шифриране на открити съобщения, нито пък за обмен на сесийни ключове, а само за подписване на предаваните данни. Използва се за осигуряване на контрол за цялостност на съобщенията и за автентичност на техните източници (използването само на този алгоритъм не гарантира автентичност).

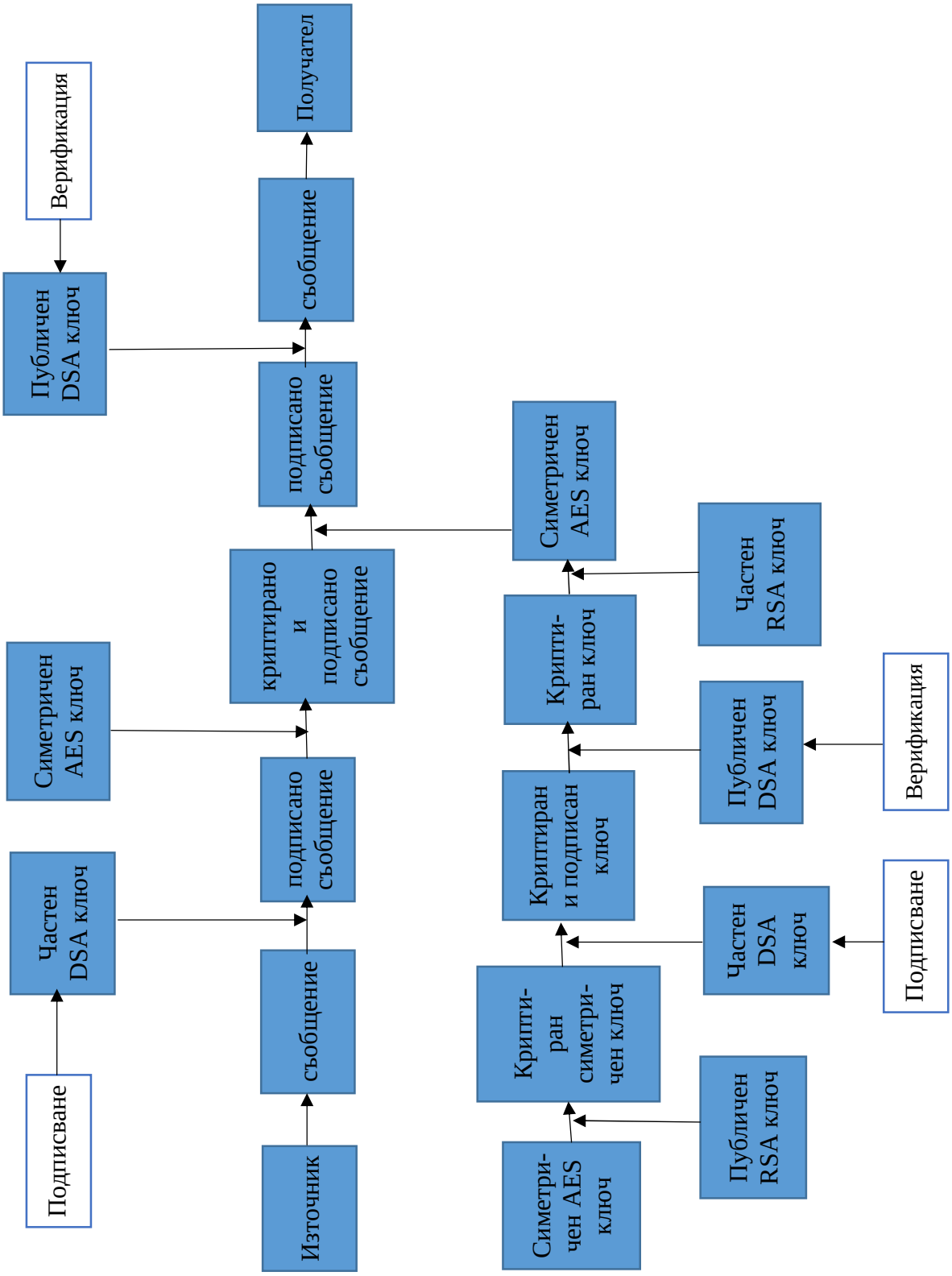
Процедура за електронно подписване:

- изчисляване на хеш-стойност (изображение), с помощта на сигурна еднопосочна хешираща функция на входното съобщение;
- шифриране на получената хеш-стойност (message digest) чрез използване на частния ключ на източника. Полученият резултат представлява цифровият подпис.

Процедура за верификация на електронен подпис:

- изчисляване на хеш-стойност на полученото открито съобщение;

- дешифриране на получения електронен подпис на съобщението с публичния ключ на източника;
- проверка за равенство между изчислената и дешифрираната хеш-стойности.



Фиг. 1. Блокова схема на Sign-then-Encrypt-then-Sign криптопротокол на хибридната криптосистема

Семантично сигурният криптопротокол в хибридната система е визуализиран в структурната схема, показана на фигура 1.

Изпълняват се следните действия от страна на подателя на съобщението:

- подателят образува съобщение за изпращане;
- подписва съобщението със собствения частен ключ на DSA алгоритъма;
- генерира случаен сесиен ключ;
- шифрира съобщението с генерирания ключ;
- шифрира сесийния ключ с публичния ключ на RSA алгоритъма на получателя;
- подписва криптирания ключ със собствения частен ключ на DSA алгоритъма;
- изпраща шифротекста и криптираният ключ;

Действия от страна на получателя при получено шифрирано съобщение:

- верификация на подписа на криптирания сесиен ключ с публичния ключ на DSA алгоритъма на източника;
- декриптиране на ключа със собствения частен ключ на RSA алгоритъма;
- дешифриране на шифротекста със сесийния ключ;
- верификация на подписа на съобщението с публичния ключ на DSA алгоритъма на източника;
- прочитане на съобщението.

Забележка: За да може да си комуникират защитено, източникът и получателят предварително е необходимо да са разменили помежду си публичните си ключове. Сесийният ключ се генерира и унищожава автоматично от системата. Частните ключове по никакъв начин не трябва да бъдат предоставени на друг.

Ключовете на отделните алгоритми са избрани спрямо препоръките на RFC 3766 за дължини на ключове. Симетричният алгоритъм AES използва секретни ключове с дължина от 128 бита. За защита на сесийните ключове се използват двойка от ключове на RSA алгоритъма (публичен и частен) с дължини от 2048 бита. Дължините на двойката от ключове при DSA са 1024 бита. При това двойките ключове на публичните алгоритми се съхраняват в кодирани формати, като публичните ключове се поддържат в кодиран X509 формат [4], [5], а частните – в кодиран PKCS#8 формат [4], [5]. Тези формати са стандартизирани и се използват от услугите за сертификация. Това дава възможност при бъдещи разширения на криптосистемата лесно да се приложат и сертификати към съответните ключове.

3. Резултати

За да бъде възможно тестването на работоспособността на хибридната криптосистема, е проектирано мобилно приложение, което позволява изпращане на кратки съобщения от мобилното устройство към сървърна машина (компютър). Съобщенията преминават през механизмите на криптиране и подписване автоматично, без намесата на потребител и се дешифрират и верифицат от компютър. За целта устройствата трябва да имат интернет свързаност.

В следващите таблици е показано времетраенето на отделните етапи от шифрирането и дешифрирането, както и от подписване и верификация. Резултатите са постигнати на конвенционален компютър, като тестовете са извършени многократно и са изведени средностатистическите стойности, показани в таблици 2, 3 и 4.

Таблица 2. Времена при алгоритъм AES

AES		
Генериране на ключ	Криптиране (100 симв.)	Декриптиране
32ms	261ms	225ms

Таблица 3. Времена при алгоритъм RSA

RSA		
Генериране на ключове	Криптиране	Декриптиране
4.7s	180ms	95ms

Таблица 4. Времена при алгоритъм DSA

DSA		
Генериране на ключове	Подписване	Верификация
190ms	10ms	12ms

Генерирането на двойките ключове за публичните алгоритми се извършва само веднъж, при първоначално стартиране, а сесийният ключ се генерира при всяко изпращано съобщение.

4. Заключение

В този доклад е предложена хибридна криптосистема със семантично сигурен криптопротокол за защита на кратки съобщения от външни, злонамерени атаки, както и изграждане на прост комуникационен канал, осигуряващ възможност за обмен на информация между мобилно устройство и компютър през незащитен комуникационен канал, използвайки проектираната хибридна система. Тя осигурява следните качества на защита: конфиденциалност (confidentiality) на разговора, цялост, интегритет (integrity) на съобщението и сесийния ключ, удостоверяване самоличността на написалия съобщението (non-repudiation), гарантиране за източника на съобщението, автентикация (authentication).

Постигнати са висока степен на сигурност и издържливост срещу евентуални криптоатаки чрез реализацията на сигурен криптопротокол, издържливи на атаки алгоритми и препоръчителни дължини и формати на ключовете.

Следва да се отбележи, че за гарантиране на висока степен на сигурност все пак се изисква допълнително време, като това време не е безкрайно малко. Един пълен цикъл (шифриране, подписване, дешифриране и верификация) изисква около 840 ms или малко по-малко от 1 секунда, т.е. всяко от устройствата внася допълнително време от около 420ms, което може да се каже, че е приемливо на фона на цялостната защита, която се осигурява. При бъдещо развитие на системата усилията ще бъдат насочени върху намаляването на тези времена, както и към изграждане на реална система за изпращане на съобщения, предназначена за крайните потребители.

Литература

[1]. Петър Антонов, Симеон Малчев, Криптография в компютърните комуникации, 2000г.
[2]. Steve Burnett, Stephen Paine, RSA Security's Official Guide to Cryptography, 2001г.
[3]. Christof Paar, Jan Pelzl, Bart Preneel, Understanding Cryptography: A Textbook for Students and Practitioners, 2011г.
[4]. Joan Daemen, Vincent Rijmen, The Design of RijndaeL: AES - The Advanced Encryption Standard, 2002г.
[5]. Jonathan Katz, Digital Signatures, 2010г.

За контакти:
инж. Мюслюм И. Вели
катедра „Компютърни науки и технологии"
ТУ - Варна
e-mail: buzzblister@gmail.com

ВЪЗМОЖНОСТИ ЗА КОРПОРАТИВНО ТЪРСЕНЕ НА ГОЛЯМ ПО МАЩАБ НАБОР ОТ ДАННИ В WEB

Ивелин И. Янев

Резюме: В настоящата публикация е разгледан проблемът с бързодействието на операции, които се извършват от самите Бази от данни [3], и по-конкретно операцията търсене. Това е проблем, който смущава всеки един софтуерен разработчик, тъй като в съвременния технологичен свят използването на интернет ресурси е залегнало във всеки един аспект от работата и ежедневието на човека. Точно тук възниква проблемът с търсенето в този голям обем от информация, защото единица данни N биват достъпни до краен брой потребители M. Но това отношение предполага по-специални изисквания към възможността за търсене и обработка на данни от сървърите и Базите от данни. Със същия казус са се сблъскали преди години разработчиците на сайта CNET и за решаването му са написали платформата Apache SOLR [1], която е с отворен код.
Ключови думи: full text search, Solr, търсене в текст, База от данни, facets.

Possibilities of Corporate Search of a Large Number of Data on the Web

Ivelin I. Yanev

Abstract: The following publication describes the problem with fast processing operations which are being processed by the databases themselves and more specifically – the operation search. This is a problem which disturbs almost every software developer. The modern technology world uses internet resources in every aspect of humans' life. Searching in this large number of data is a problem, which occurs here, because a unit of data N can be accessed by end number of clients M. The developers of CNET site faced the same problem a few years ago and they wrote the open source code platform Apache SOLR as a solution to the problem.
Keywords: full text search, Solr, searching, Data bases, facets.

1. Увод

След появането на големите онлайн системи, в които търсенето на информация от страна на потребители стои в основата на тяхната функционалност, е трябвало да се създаде нов подход при действието "търсене" [2]. За целта се появяват доста инуитивни решения на този казус, като: Solr, ElasticSearch, Whoosh и др. Всичките те работят на един и същ принцип - индексиране на документи. Но с времето Solr се е сдобил с повече популярност сред разработчиците, което е довело и до големия брой отделни функционални възможности и предимства пред останалите. Solr е написан на Java, но е достъпен през HTTP, което дава възможност за работата му с много голям брой програмни езици. Solr е приложение - обикновено сървлет като Tomcat, Glassfish или JBoss. Може да се манипулира, както казахме по HTTP, използвайки XML, JSON, CSV или двоичен формат [1]. Но самата библиотека осигурява и ниво на абстракция, което предоставя възможността да се извикват native методи на Solr.

В основата на Solr стои индексиране на документи. Тези документи представляват обекти от реалния свят: книги, къща, стая и др. Чрез тях Solr представя т. нар. "fulltext search" и ако съпоставим с SQL, това се постига с WHERE клаузи и LIKE оператора [3]. Към тези документи може да се прикрепи и допълнителна информация без значение от големината ѝ.

Едни от най-полезните функции на Solr са Facets (фасети). С няколко думи, с фасетите можете да изградите филтри, по които да се търси (групиране на резултатите по дадени критерии «тип/марка/модел/категория») [1].

Solr работи на инстанция, като една инстанция може да има едно или повече ядра. А самото ядро е колекция от конфигурирани документи със съответна схема.

Как се инсталира Solr?

Apache Solr се инсталира много лесно в рамките на следните няколко действия:

1. Сваля се съответната версия на Solr от сайта http://lucene.apache.org/ или друго хранилище на файлове;
2. Разархивира се архива и се поставя в избрана от нас директория на файловата система;
3. Стартира се start.jar файл.

Ще покажем горните действия, като за целта използваме ОС Ubuntu 14.04.

1. wget http://apache.cbox.biz/lucene/solr/4.9.0/solr-4.9.0-src.tgz
2. tar -zxvf solr-4.9.0-src.tgz
3. cd solr-4.9.0/solr/example
4. java -jar start.jar

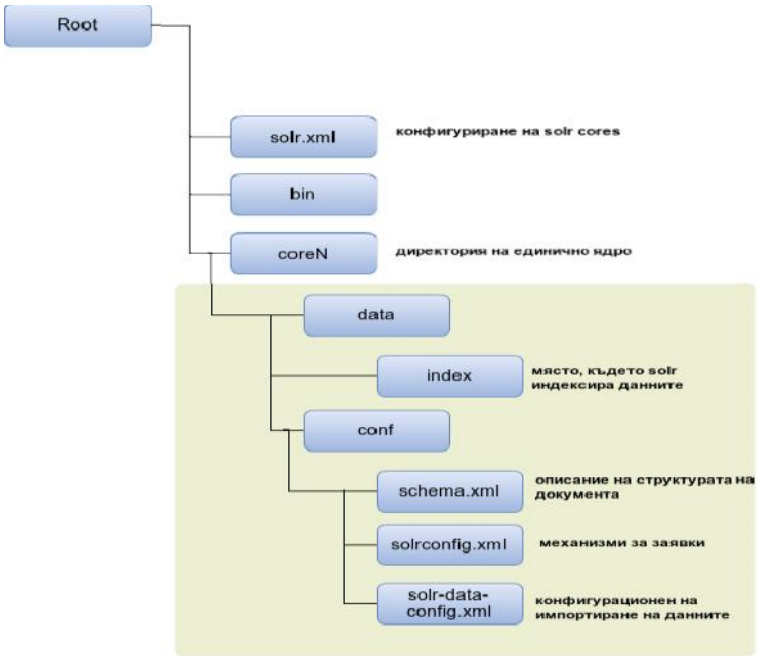
След като сме извършили горните действия, имаме вече инсталиран и пуснат Solr. Можем да използваме интернет интерфейса, за да работим с него. За целта трябва да посетим следния адрес: http://localhost:8983/solr/.

2. Конфигуриране на Solr

Конфигурирането на на Solr не е лесна задача и никога не се прави веднъж за винаги за дадено ядро. За конфигурирането на нова инстанция са необходими следните няколко стъпки:

- създаване на ново ядро (solr.xml);
- създаване схемата на документа (schema.xml);
- създаване на механизма за заявки (solrconfig.xml)

За онагледаване на тези стъпки на фигура 1 е показана структурата на конфигурационните файлове [1].



Фиг. 1. Файловата структура на Solr

Създаване на ново ядро (индекс)

В този документ се записват всички данни, които искаме да бъдат индексирани в нашия Solr документ. Всеки един документ може да има само една схема, която е съобразена с нужните входни данни и с това, как и какви търсения ще искаме да направим. Трябва да споменем, че всяко ново ядро има отделна директория, а името бива избрано от самия разработчик. За конфигуриране на ядро, трябва да се запишат следните тагове:

```
<solr persistent="true" sharedlib="lib">
<cores adminpath="/admin/cores" host="${host:}" hostcontext="${hostContext:}" hostport="${jetty.port:}"
zkclienttimeout="${zkClientTimeout:15000}">
<core default="true" instancedir="auctions" name="auctions">
</core></cores>
</solr>
```

Обикновено конфигурационният solr.xml, който се намира в /example/, е достатъчен, но няма да има проблем и ако решим да си конфигурираме наш файл. Трябва да споменем, че в новите версии от 4.3 на Apache Solr има нов синтаксис за solr.xml, който трябва да спазваме, ако искаме да работим със Solr Core конфигурацията им.

Схема

Solr схемата описва основната единица на информация - документ. Всяко едно ядро може да има само една схема, която се състои от известен брой полета. На всяко поле бива зададен тип, който се определя в зависимост от анализа на текста, който искаме да индексираме. Пример за една такава схема е schema.xml:

В следния xml са показани полетата и примерни данни, които трябва да приемат тези полета.

```
<doc>
<field name="auction_id">123</field>
<field name="title">Acer IPS Monitor</field>
<field name="category">monitors</field>
<field name="current_bid">95</field>
<field name="end_date">2014-07-06T09:26:04.18Z</field>
<field name="feature">IPS</field>
<field name="feature">Swivel</field>
</doc>
```

Ето и самата схема, на която елементите ще бъдат описани по-долу:

```
<schema name="example" version="1.5">
          <fields>
          <field name="_version_" type="long" indexed="true" stored="true" required="true"/>
          <field name="auction_id" type="string" indexed="true" stored="true" required="true"
 multiValued="false" />
     <field name="title" type="text_en" indexed="true" stored="true" required="true" multiValued="false" />
     <field name="category" type="string" indexed="true" stored="true" required="true" multiValued="false"
/>
     <field name="current_bid" type="currency" indexed="true" stored="true" required="true"
multiValued="false" />
     <field name="end_date" type="date" indexed="true" stored="true" required="true" multiValued="false"
/>
     <field name="feature" type="string" indexed="true" stored="true" required="false" multiValued="true"
/>
  </fields>
```

```
 <uniqueKey>auction_id</uniqueKey>
 <types>
    <fieldType name="string" class="solr.StrField" sortMissingLast="true" />
    <fieldType name="date" class="solr.TrieDateField" precisionStep="0" positionIncrementGap="0"/>
    <fieldType name="long" class="solr.TrieLongField" precisionStep="0" positionIncrementGap="0"/>
<fieldType name="text_en" class="solr.TextField" positionIncrementGap="100">
    </fieldType>
    <fieldType name="currency" class="solr.CurrencyField" precisionStep="8" defaultCurrency="USD"
currencyConfig="currency.xml" />
 </types>
</schema>
```

Както казахме, полето описва парче информация, което се използва от Java за предаване на данните, независимо дали те ще се съхраняват и изискват във всеки един документ. Има два типа полета “copyField” и “dynamicField” (да не се бърка с параметризиран тип type="string").

“CopyField” позволяват да се индексират полета по повече от един начин, както и позволява различни типове полета “text_bg” или текст.

“DynamicField” са обратни на горепосочените, като позволяват да се обработват няколко полета по един и същи начин. А общата употреба на динамични полета представлява улавяне на всички полета в документа, в който не трябва да се индексира. Това е необходимо, тъй като когато полета са индексирани, всички полета по документи трябва да бъдат обработени.

Как се ползват, е показано на следните редове:

shmena.xml

```
<schema name="example" version="1.5">
 <fields>
    <field name="title" type="text_en" indexed="true" stored="true" required="true" multiValued="false" />
    <field name="category" type="string" indexed="true" stored="true" required="true" multiValued="false"
/>
    <field name="feature" type="string" indexed="true" stored="true" required="false" multiValued="true"
/>
    <field name="allText" type="text_en" indexed="true" stored="false" required="true"
multiValued="true" />
 </fields>
 <copyField source="title" dest="allText" />
 <copyField source="category" dest="allText" />
 <copyField source="feature" dest="allText" />
 <dynamicField name="*" type="ignored" multiValued="true" />
</schema>
```

Analysers, Tokenisers и Filters

Analysers (Анализатор): конвертира текста от полетата в индексиран текст. Всеки анализатор се състои от един или повече tokenizer или филтри. Анализаторите нямат име, а биват идентифицирани от типа на полето, където са дефинирани.

Tokenisers: разбива поток от текст на отделни части (tokens). Например, имаме текста “Харесвам кюфтета с картофки”. Можем да го подложим на филтър, който да го разбие на следните части (tokens): (Харесвам, кюфтета, с, картофки) или пък да се преобразува на още по-малки части (Х, а, р, е, с, в, а, м, к, ю, ф, т, е, т, а, с, к, а, р, т, о, ф, к, и).

Filters (филтри): вземат частите (tokens) и ги транформират, като това действие може да бъде отхвърлено. Примерно, филтър, който преобразува всички букви в малки.

Ето как се определят и отбелязват споменатите компоненти в Solr.

```
<fieldType name="text_en" class="solr.TextField" positionIncrementGap="100">
<analyzer type="index">
  <tokenizer class="solr.StandardTokenizerFactory"/>
  <filter class="solr.StopFilterFactory" ignoreCase="true" words="lang/stopwords_en.txt"
enablePositionIncrements="true"/>
  <filter class="solr.LowerCaseFilterFactory"/>
<filter class="solr.EnglishPossessiveFilterFactory"/>
  <filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt"/>
  <filter class="solr.PorterStemFilterFactory"/>
</analyzer>
<analyzer type="query">
  <tokenizer class="solr.StandardTokenizerFactory"/>
  <filter class="solr.SynonymFilterFactory" synonyms="synonyms.txt"  ignoreCase="true" expand="true"/>
  <filter class="solr.StopFilterFactory" ignoreCase="true" words="lang/stopwords_en.txt"
enablePositionIncrements="true" />
  <filter class="solr.LowerCaseFilterFactory"/>
<filter class="solr.EnglishPossessiveFilterFactory"/>
  <filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt"/>
  <filter class="solr.PorterStemFilterFactory"/>
</analyzer>
</fieldType>
```

Multivalued Fields(многозначни полета):

Тези полета позволяват да имаме две стойности в документа на единично поле. Примерно, документът, показан по-долу, има само една стойност на поле. Но в примера за multivalued field полето “feature” има две стойности или с други думи Solr позволява да имаме няколко стойности в едно поле. Но важното е да разберем, че когато използваме multivalued за поле, данните биват “сплескани”.

XML, когато полетата приемат само една стойност:

```
<auction>
  <title>Desktop PC</title>
  <feature>
     <name>RAM</name>
     <value>16 GB</value>
  </feature>
  <feature>
     <name>CPU Frequency</name>
     <value>4.5 GHz</value>
  </feature>
</auction>
```

Как изглежда документът в Solr:

```
<doc>
  <field name="title">Desktop PC</field>
  <field name="feature_name">RAM</field>
  <field name="feature_value">16 GB</field>
  <field name="feature_name">CPU Frequency</field>
  <field name="feature_value">4.5 GHz</field>
</doc>
```

И полученият ефект:

```
<doc>
  <field name="title">Desktop PC</field>
```

```
  <field name="feature_name">RAM CPU Frequency</field>
  <field name="feature_value">16 GB 4.5 GHz</field>
</doc>
```

Solr спазва този начин на индексиране и е ясно, че връзката между името и стойността се губи. Или с други думи, в Solr не се поддържа отношението «един към много» в едно ядро.

solrconfig.xml

solrconfig.xml се използва за конфигуриране на множество аспекти на експлоатацията на Solr, например използва се за конфигуриране на:

- заявките;
- слушателите (слушат за заявки);
- администраторен интерфейс;
- репликация и дублиране.

Обикновено единствените промени, които трябва да бъдат направени за solrconfig.xml, са: добави или промени търсенето по заявките. Тези два примера ще бъдат обхванати в секциите съответно за индексиране и търсене.

3. Индексиране на данни

Има два начина, които се използват за добавяне и индексиране на данни [3]:

XML заявка или импортиране от БД, като всички индекси се съхраняват в БД. Но по-добрият вариант е следният: първо се извършва импортиране на данните. Когато се извърши това импортиране чрез изпращане на документа във формат на XML чрез HTTP заявка към Solr.

Импортирането на данни се извършва с помощта на Data Import Handler(DIH). За използването му трябва да се създаде конфигурационен файл, но трябва и да се допълни solrconfig.xml за използването на DIH. Повече подробности за конфигуриране на DIH може да се научи на адрес: http://wiki.apache.org/solr/DataImportHandler. След като импортираме данните, новите документи се добавят, като се публикуват със Solr манипулатор. За целта сме показали следния ред, който трябва да присъства в solrconfig.xml

```
<requestHandler name="/update" class="solr.UpdateRequestHandler" />
```

По този начин, чрез изпращане на XML към URL http://localhost:8983/solr/coreName/update Solr ще добави документа към индекс-а. В случай, че има вече индексирани XML данни, които искаме да индексираме, то най- вероятно няма да са в формата, в който Solr очаква.

Оригинален XML:

```
<auction>
  <auction_id>234</auction_id>
  <title> IPS Monitor</title>
  <category>monitors</category>
  <current_bid>2.95</current_bid>
</auction>
```

Solr формат:

```
<doc>
  <field name="auction_id">54432834</field>
```

```
  <field name="title">Dell M2012 24" IPS Monitor</field>
  <field name="category">monitors</field>
  <field name="current_bid">279.95</field>
</doc>
```

4. Търсене

Търсенето представлява заявка към Solr, в която описваме какво търсим, какви критерии и какви резултати да ни върне, ако има индексирани данни. За илюстриране е следният пример:

```
<requestHandler name="/broadQuery" class="solr.SearchHandler">
  <lst name="defaults">
//метод на парсване при търсенето
    <str name="defType">edismax</str>
//в какъв вид(формат) да е върнатия резултат
    <str name="wt">xml</str>
//полетата с резултати
    <str name="fl">auction_id title</str>
//полета в които се търси и тяхната тежест
    <str name="qf">Title^2 Feature</str>
//броя на резултатите, които се връщат
    <str name="rows">100</str>
    <str name="pf">Title^4 Feature^2</str>
    <str name="ps">0</str>
    <str name="echoParams">all</str>
  </lst>
</requestHandler>
```

Няколко уточнения на част от по-горните редове:

defType - определя какъв парсер за търсене ще се използва. Има няколко парсера, които са ни на разположение: Standard Search, DisMax и DisMax. eDismax съчетава функциите на Standard Search и DisMax. eDisMax поддържа пълна свобода на синтаксис и е по-толерантен към синтактични грешки в заявката.

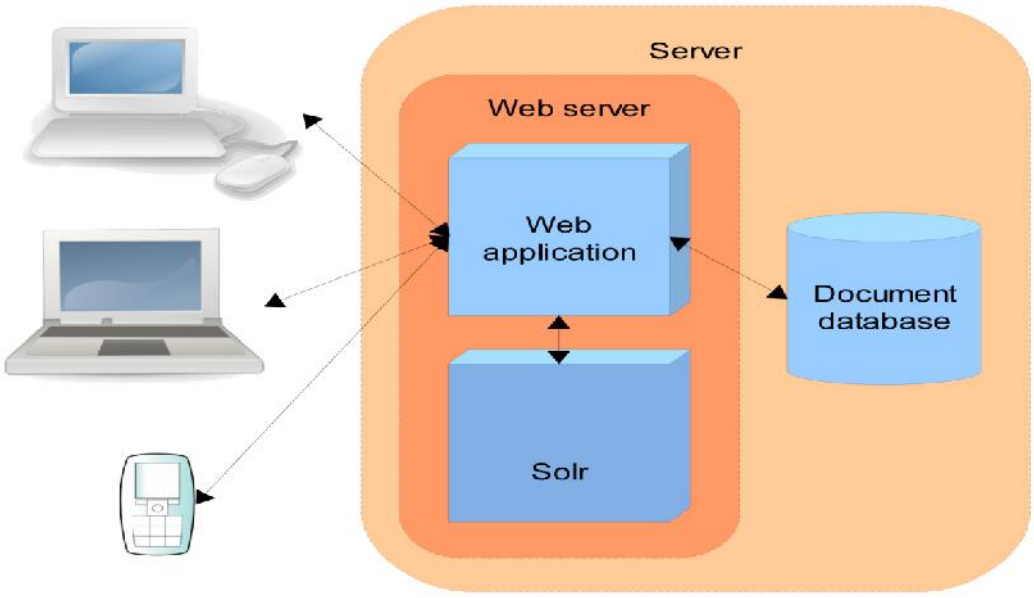
Pf - уточнява кои полета трябва да бъдат проверени за съвпадения. Ако съвпаденията са достатъчно близки, те може да се считат за намерен резултат. Може да се определя и тежестта на съвпадения, като по този начин може да се управлява колко е чувствително търсенето.

ps - уточнява колко термини могат да бъдат между съответни условия.

Ще разгледаме един експериментален пример за разликата между Solr и Mysql: нека приемем, че имаме набор от данни около 400 000 записа, които съдържат търговки имена и адреси. На Mysql сме задали Full Text Index върху тези полета. За по-добра гъвкавост при използване на Solr създаваме следната API структура, която ще извлича данните при търсене (фигура 2).

От направените тестове се получиха следните резултати при извличане на резулати:

- Solr беше най-бърз в 6100 търсения при общо 6300 възможни или, обърнато във време, на Solr му трябваха 350 сек., докато на Solr - 2900 сек. Въпреки че тези числа изглеждат впечатляващи, това е, защото голям фактор е самото API, с което работи Solr [2].



Фиг. 2. API интерфейс за работа с Solr

5. Заключение

От гледна точка на База от данни [3], Solr може да се разглежда като една таблица с много бързо изпълнение на заявки към нея. От направените примери можем да заключим, че конфигурирането на ядро в Solr не е лесно, но само използване на възможностите на технологията позволява да се управляват данни, които биха прекалено сложни и в някои случаи невъзможни, ако се управляват от База от данни. Трябва да подчертаем, че Solr не е База от данни, а само индексирани редове от документи в XML формат. Solr бива използван от най- голямите компания в света, като: AOL, AT&T, CISCO, Ebay, Reddit и още много други.

Литература

[1] Apache Solr Reference Guide [Онлайн]:
https://cwiki.apache.org/confluence/display/solr/Apache+Solr+Reference+Guide
[2] Martin Reddy, Rule Of API Desing
[3] Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems

За контакти:
инж. Ивелин И. Янев
катедра „Компютърни науки и технологии”
Технически университет - Варна
E-mail:qnev89@gmail.com

ПРИЛАГАНЕ НА RSA АЛГОРИТЪМ ПРИ КРИПТИРАНЕ НА СЪОБЩЕНИЯ В ЗАЩИТЕНА КОМУНИКАЦИОННА СРЕДА

Цеко И. Цеков, Никола С. Обретенов, Милена Н. Карова

Резюме: Докладът дискутира разработка, представляваща система за сигурност, която служи за удостоверяване на лични данни, с цел навлизане в сигурна виртуална среда. Сигурността (по-специално криптиране на съобщения) се обезпечава от вариант на RSA алгоритъм за криптиране на информацията. За да се улесни възможния достъп до защитена виртуална среда от страна на потребителите, защитената структура включва в себе си смартфон и персонален компютър. Приложението осигурява удобен и сигурен вход в комуникационната среда като повишава сигурността на системата.
Ключови думи: авторизация, криптиране, RSA алгоритъм, парола, client phone, main server, чат-стая

RSA Implementation in Secure Communication Environment

Tseko I. Tsekov, Nikola S. Obretenov, Milena N. Karova

Summary: The report discusses development, representing a security system that serves to verify the personal data in order to enter in a secure virtual environment. Security (especially encryption of messages) is provided by a variant of RSA encryption algorithm information. To facilitate the possible access to a protected virtual environment by consumers, protected structure includes a smartphone and personal computer. The application provides a convenient and safe entrance in communication environment by increasing the system security.
Keywords: authorization, encryption, RSA algorithm, password, client phone, main server, chat room

1. Въведение

Криптирането е техника, която използва математика, за да трансформира информация по начин, който я прави нечетима за всеки друг, освен за този със специални познания, обикновено отбелязвани като „ключ”. Има много приложения на криптирането, но някои от най-важните помагат да се защити сигурността и поверителността на файловете на вашия компютър, информация, преминаваща през интернет или оставена да седи на файл на нечий друг компютър. Ако криптирането се използва както трябва, информацията би трябвало да е четима само за вас и за хората, които получават ключ от вас. Криптирането осигурява силна техническа защита срещу много видове заплахи и тази защита често е лесно да се получи.

Криптирането играе важна роля в смекчаването на риска, свързан с множеството заплахи. Ако информация с висока степен на поверителност е криптирана, ще трябва да се вземе тайният ключ, за да се декодира. Когато се криптира информация с висока степен на поверителност и тя завършва записана от други в курса на комуникацията онлайн, криптирането пречи на тези без таен ключ да узнаят съдържанието на съобщението.

Модерното криптиране е много трудно за пробиване, използва сложна математика, за да обърка информацията и да се убеди, че само един човек, притежаващ правилния ключ може да я подреди. Още повече, модерните компютри са достатъчно мощни, за да може да бъдат насочени да криптират комуникациите и информацията рутинно, а не просто криптирането да се пази за специални случаи или за информация с особено голяма степен на поверителност.

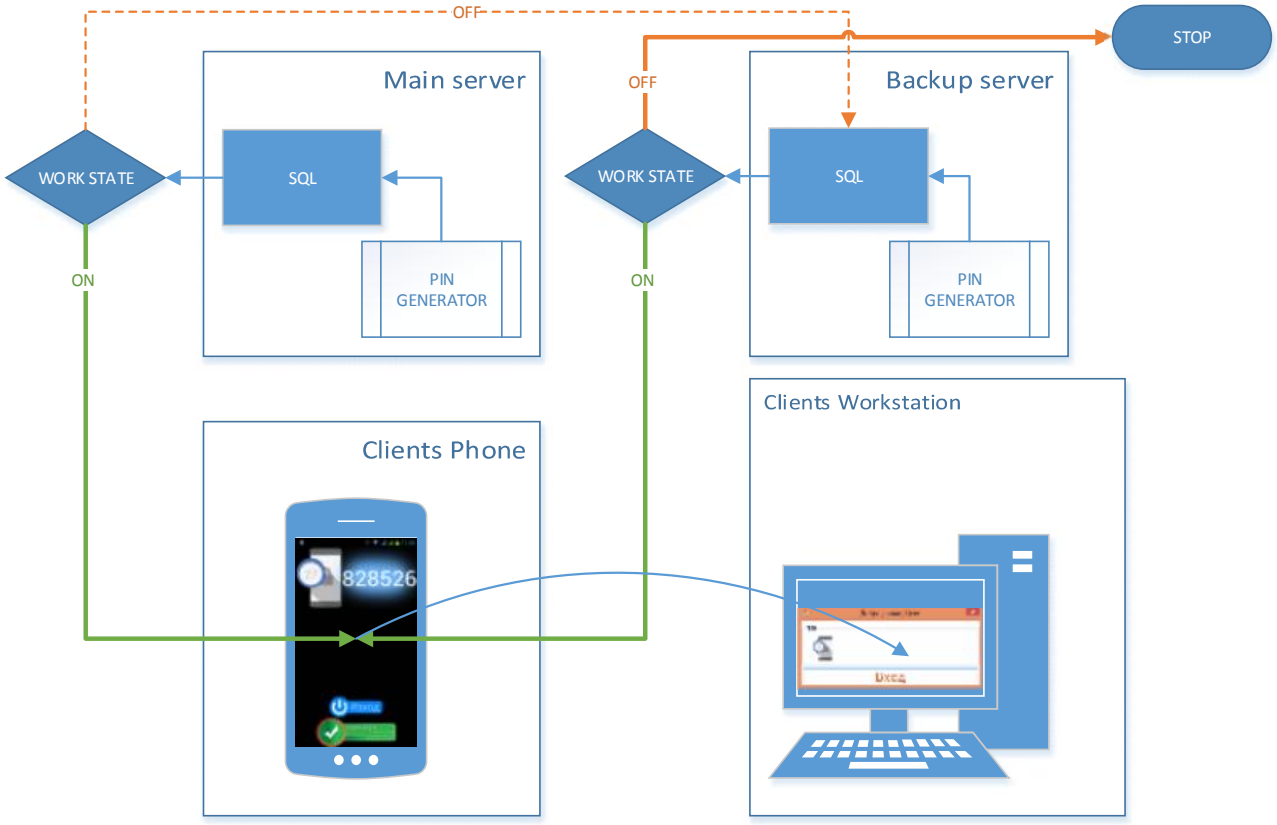
В разработената системата криптирането на съобщения е част от обезпечаване на работата в „чат-стая“, която хардуерно се обезпечава от група компютри, обслужвани от два

сървъра - един главен и един резервен (backup). Сървърите поддържат базата от данни, в която са записите на потребителите. Тези данни служат за вход в чат-стая, която е десктоп приложение. Веднъж регистрани, чрез своите смартфони, потребителите използват за вход генерираните към смартфона кодове (пароли). След успешно въведени данни, достъпът до системата е осигурен. В тази виртуална среда комуникацията става чрез криптиращи алгоритми, осигуряващи сигурната кореспонденция между потребителите.

2. Изложение

2.1 Общо описание на системата

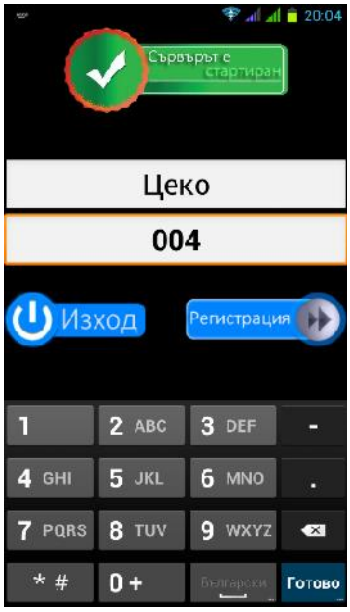
Общият вид на системата, включваща комуникация между приложенията и сървърите е представена на фигура 1.



Фиг. 1. Блок-схема на комуникационната система

Системата, както е описано в схемата, се поддържа благодарение на сървърите, т.е. те са главният фактор за работата на другата част от устройствата.

В сървъра се съхраняват данните за вход в десктоп приложението, които постъпват от смартфона, служещ за регистрация. Информацията, която се изпраща от смартфона, е кодът на компютъра, от където ще се влиза в чат-стаята и потребителско име за вход в чат-стаята. Тук се включва една от скритите за потребителя „идеи“ на програмата – заедно с въведените от него данни, към сървъра се изпраща и IMEI номерът на устройството (фигура 2).



Фиг. 2. Регистрация

Кодът, който се попълва, трябва да бъде задължително съвпадащ с този на работното място. Самият код на работното място се изписва на монитора, след като е стартирано десктоп приложението.

Таблица 1 от базата от данни се попълва след успешно изпълнена регистрация.

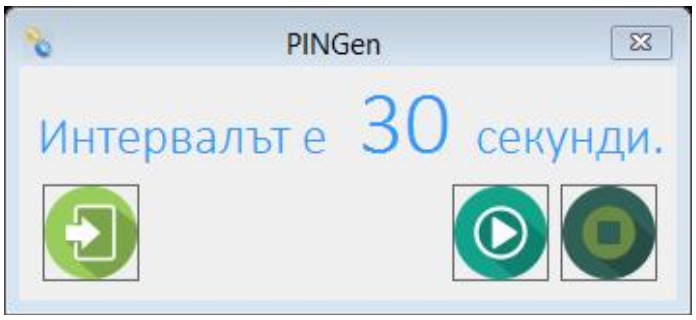
Таблица 1. Съдържание - база данни

Id	IMEI_device	IP_device	Username	PIN_employee
1	356876030250990	192.168.1.4	User	741163
2	492019943230168	192.168.1.12	User2	889325

Структурата на БД е показана на фигура 3. Тя е реализирана на **Microsoft SQL Server 2008 R2**. Към описаните в Таблица 1 полета се включва и едно ново ниво на сигурност, нужно за вход в системата – **ПИН код**. Той се генерира от сървъра, на който му е зададен интервал от време, през което да се сменя. Трябва да се отбележи, че Таблица 1, показва данните в некриптиран вид (всички те се изпращат в криптирано състояние). Алгоритмите за криптиране се използват готови и работата на функциите е описана в т. 2.2. Програмата, която генерира тези кодове, е инсталирана на сървърната машина (фигура 5). По подразбиране, времето за смяна на ПИН кода е 30 секунди (фигура 4).

Column Name	Data Type	Allow Nulls
id	int	☐
IMEI_device	nvarchar(50)	☐
IP_device	nvarchar(50)	☐
Name_employee	nvarchar(50)	☐
PIN_employee	int	☐

Фиг. 3. Структура на таблица от БД



Фиг. 4. ПИН-генератор

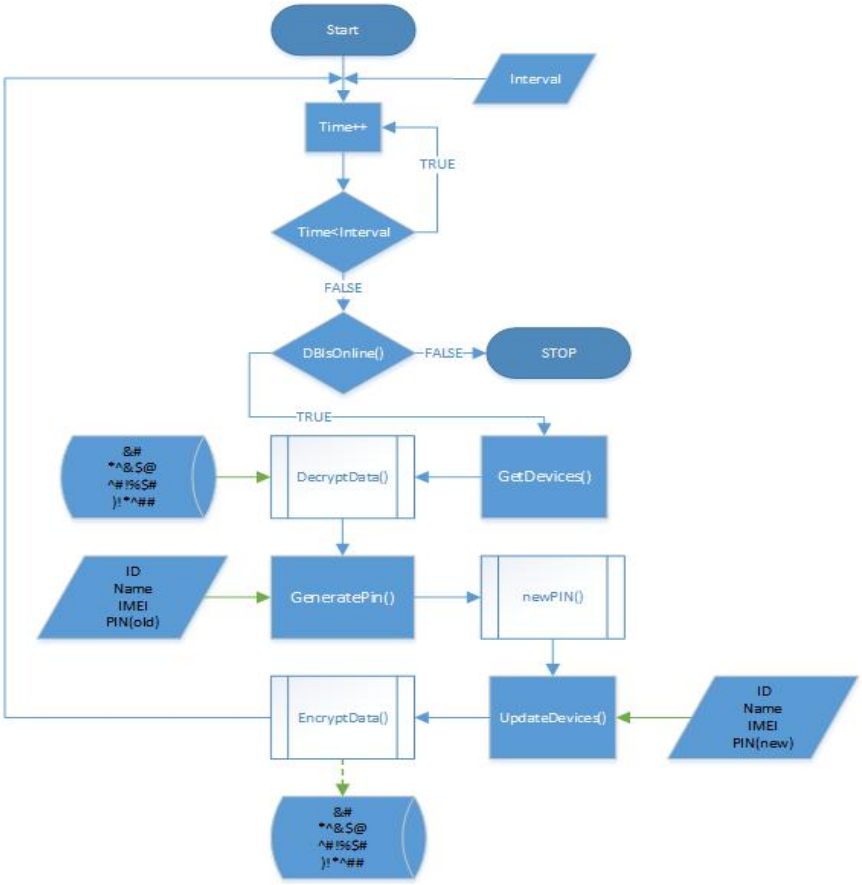
2.2 Програма PIN генератор

В началото на програмата се задава като параметър интервал, през който ще се генерира нов пин за потребителя. При достигане на зададения интервал се прави проба за установяване на връзка със сървъра. При неосъществена комуникация със сървъра изпълнението спира.

При осъществяване на връзка с базата от данни се изпълнява програмата GetDevices(). Тя осигурява достъп до базата от данни със заявка, която връща информация за всички потребители. Информацията за потребителите постъпва в криптиран вид и затова се прекарва съответно през модул за декриптиране.

GeneratePin() генерира се нов ПИН на базата на вече приведената в разбираем вид информация, която се обработва за всеки потребител поотделно.

UpdateDevices() криптира данните за потребителите с ново генерирани ПИН кодове (за по-голяма сигурност) и след това те се изпращат обратно към базата от данни.



Фиг. 5. Блок – схема на програмата за генериране на кодове

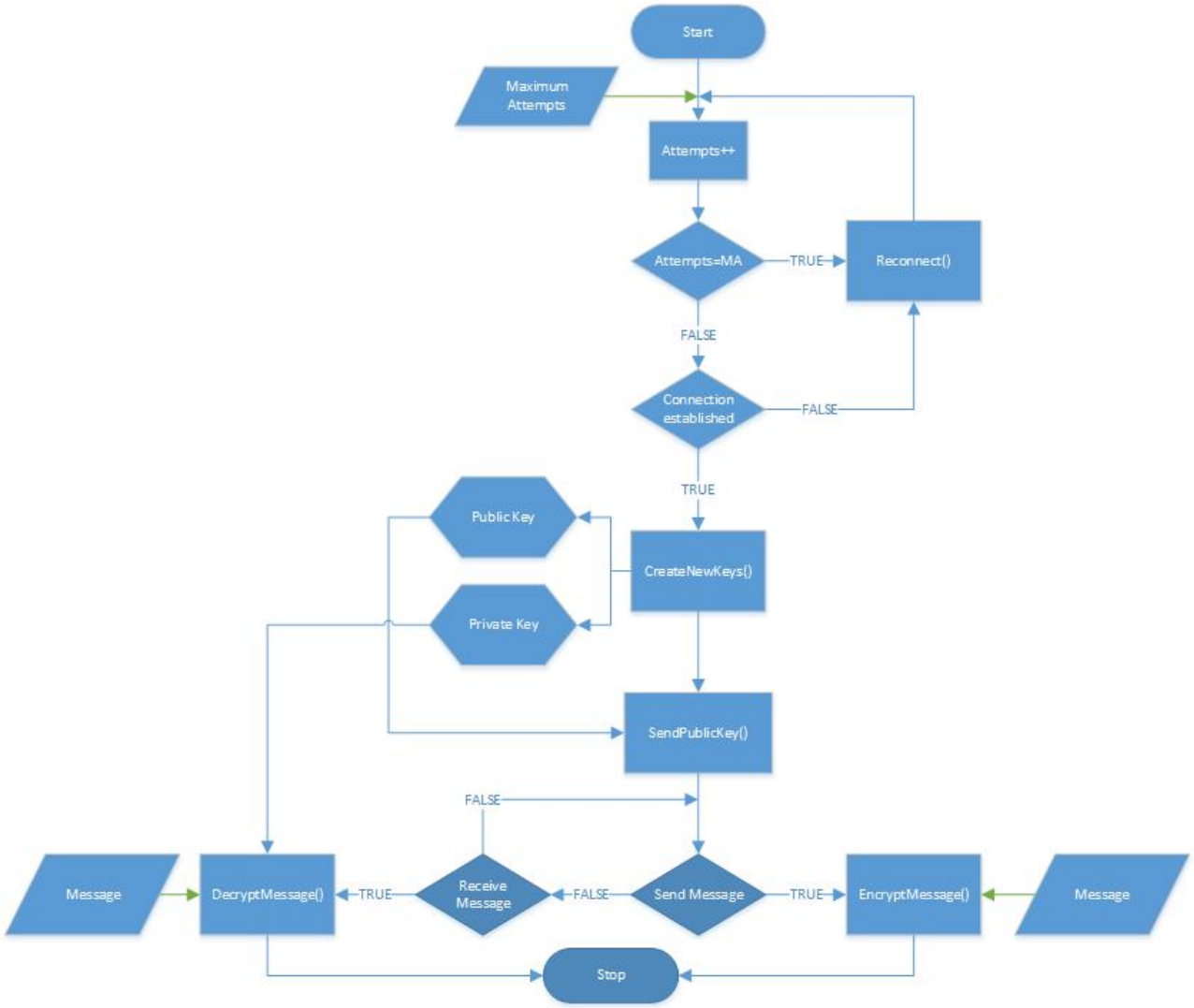
2.3 Програма за предаване на криптирани съобщения в „чат-стая“

Начинът на криптиране на съобщенията, осъществяването на връзка и предаването на събщенията е даден на фигура 6.

Като параметър се задава брой опити (Maximum Attempts – MA), като след достигането им се зарежда блок „Reconnect()”, който отваря нова връзка с избрания участник от чата.

Ако връзката е установена преди достигане на “MA”, се генерират ключовете, с които ще се осъществи сигурната комуникация между двете страни. Генерираните ключове са съответно личен и публичен. Публичният ключ веднага се изпраща към събеседника. По този начин той ще праща криптирани съобщения с един и същи публичен ключ.

При осъществена реална комуникация между двете страни се „слуша“ за операцията, която ще предприеме потребителят, т.е. дали ще се изпраща или ще се получава съобщение. При приемане на съобщение то трябва да се декриптира, а при изпращане на съобщение то трябва да се криптира съответно и да се подпише (за да се знае от кого е изпратено) Signing messages [1].



Фиг. 6. Блок – схема на програмата за криптиране на съобщения чрез RSA алгоритъм и осъществяване на връзка в защитена среда

Ключовете се генерират чрез използване на готови функции от библиотеката на Visual Basic - Security.Cryptography (фигура 7). Тези библиотеки използват RSA алгоритми за криптиране.

```
Public Shared Function CreateNewKeys() As Keypair
        Try
            Using RSA As New RSACryptoServiceProvider(2048) 'битов ключ
                Dim Keys As New Keypair

                Keys.Privatekey = RSA.ToXmlString(True)
                Keys.Publickey = RSA.ToXmlString(False)
                Return Keys
            End Using
        Catch ex As Exception
            Throw New Exception("Keypair.CreateNewKeys():" & ex.Message, ex)
        End Try
End Function
```

Фиг. 7. Програмен фрагмент за генериране на ключове

Съответно видът на ключовете е в XML формат. Примерно, публичният ключ изглежда така:

```
<RSAKeyValue>
</Modulus>
        vP5mMXMeiG6bPyCjx0+RX6FHIPxub7HfGi9jF2QWbZl6eckm2b3flMEcw6QGY
        7yR41sZ72wiPL5bD2D9jZDtsTEW879XBbFizpWyeRj06ohjIusbGBSuxE4CKKW
        JnLkFEePqghqv3YqMfAaptC01N57Y2nJaDW6w2QjmgcikUzs=
</Modulus>
<Exponent>
        AQAB
</Exponent>
</RSAKeyValue>
```

За криптиране и декриптиране се използват готови библиотеки с RSA криптиране Cryptography.RSACryptoServiceProvider. Авторите не променят RSA параметри като случайните цели числа p и q, числото на Ойлер, формули за изчисление на ключовете и др. Използва се готов клас с RSA компоненти RSA Class [2].

```
Public Shared Function Encrypt(ByVal Data As String, ByVal Publickey As String) As RSAResult
        Try
            Dim ByteConverter As New UnicodeEncoding()
            Return Encrypt(ByteConverter.GetBytes(Data), Publickey)
        Catch ex As Exception
            Throw New Exception("Encrypt(String): " & ex.Message, ex)
        End Try
End Function

Public Shared Function Encrypt(ByVal Data() As Byte, ByVal Publickey As String) As RSAResult
        Try
                    Dim RSA As System.Security.Cryptography.RSACryptoServiceProvider = New
       System.Security.Cryptography.RSACryptoServiceProvider()
            RSA.FromXmlString(Publickey)
            Return New RSAResult(RSAEncrypt(Data, RSA.ExportParameters(False), False))
        Catch ex As Exception
            Throw New Exception("Encrypt(Bytes): " & ex.Message, ex)
        End Try
End Function
```

```
Public Shared Function Decrypt(ByVal Data() As Byte, ByVal Privatekey As String) As
RSAResult
        Try
            Dim RSA As System.Security.Cryptography.RSACryptoServiceProvider = New
System.Security.Cryptography.RSACryptoServiceProvider()
            RSA.FromXmlString(Privatekey)
            Dim Result As New RSAResult(RSADecrypt(Data, RSA.ExportParameters(True), False))
            Return Result
        Catch ex As Exception
            Throw New Exception("Decrypt(): " & ex.Message, ex)
        End Try
End Function

Private Shared Function RSADecrypt(ByVal DataToDecrypt() As Byte, ByVal RSAKeyInfo As
RSAParameters, ByVal DoOAEPPadding As Boolean) As Byte()
        Try
            Dim decryptedData() As Byte
            Using RSA As New RSACryptoServiceProvider
                RSA.ImportParameters(RSAKeyInfo)
                decryptedData = RSA.Decrypt(DataToDecrypt, DoOAEPPadding)
            End Using
            Return decryptedData
        Catch e As CryptographicException
            Throw New Exception("RSADecrypt(): " & e.Message, e)
        End Try
End Function
```

Фиг. 8. Програмни фрагменти на програмите за криптиране и декриптиране на текста

3. Тестване на система

3.1 Спецификации

Мобилното приложение е проектирано и тествано на операционната система Android, като достъпните версии за работа са от “Android 4.1 Jelly Bean” и нагоре. Хардуерните характеристики на смартфона, от който е тествано приложението, са:

- CPU: MTK6589 Quad Core 1.2GHz
- RAM: 2GB

Десктоп приложението е създадено и тествано на операционна система “Windows”, като версиите за работа са съответно от “XP/Vista/7/8/8.1”. Спецификациите на машината, върху която е тествано десктоп приложението, са:

- CPU: Intel® Core™ i5-2430M 2.4GHz
- RAM: 8GB

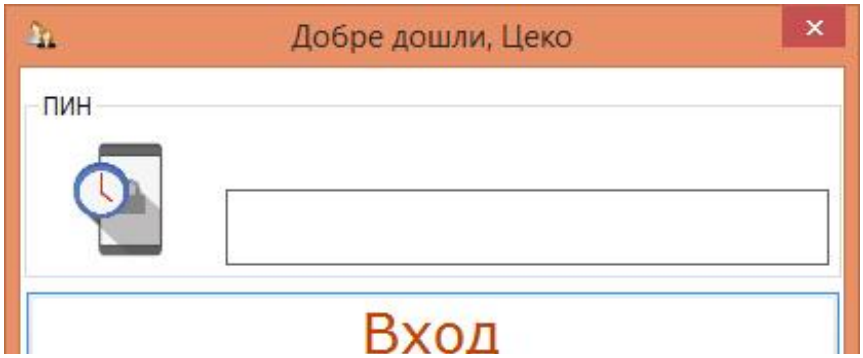
Сървърната машина е тествана със спецификации, посочени по-горе (десктоп приложението), като операционната система, с която е тестван SQL сървърът, е Windows Server 2012.

3.2 По-важни стъпки на приложението

При липса на регистрация десктоп приложението изписва кода на работното място, с който трябва да го регистрирате в мобилното приложение (базата от данни). Стъпките, през които минава потребителят преди регистрация, са описани на фигура 9.

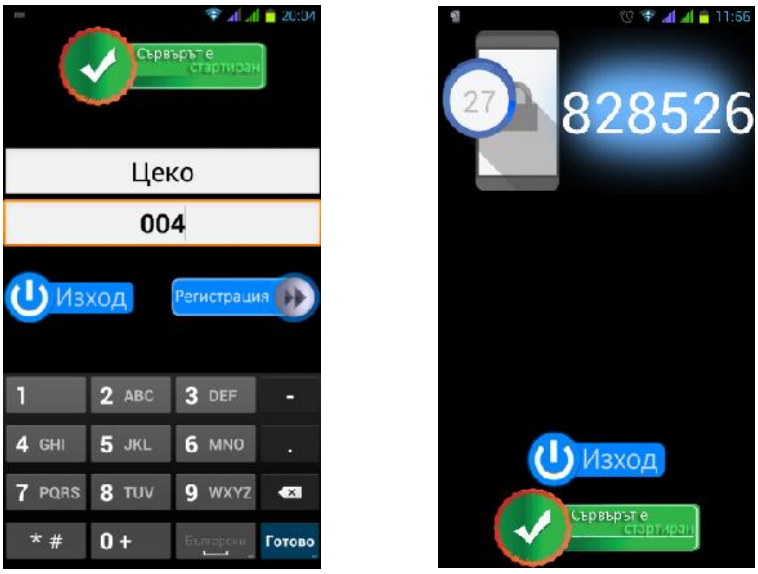
Фиг. 9. Код нужен при регистриране на работното място

Формата за вход на главното десктоп приложение изисква ПИН (фигура 10). ПИН кодът, нужен за успешен вход в системата, се визуализира на личните телефони на потребителите. По този начин устройството, което те най-често използват, се превръща в ключ за тяхната работа. Форми за използване на кода при регистрацията на работното място и при презареждане на приложението след регистрацията са дадени на фигура 11.



Фиг. 10. Login – Форма в десктоп приложението

При правилно разчитане на ПИН кода от телефона, той се въвежда в login формата спомената по-горе, и потребителят има достъп до защитената виртуална среда („чат-стая“). При успешно влизане в стаята, всяко изпратено съобщение може да се види от всички потребители, които са в стаята.



Фиг. 11. Етапи при регистрането през смартфона.

3. Заключение

Приложението, което е създадено, е много удобно и осигурява нужната сигурност при конфиденциално общуване. Може да обхване всякакъв тип фирмена или корпоративна дейност, където изключително важно е преносът на информация да не бъде достояние на

трета страна. Бъдещите разработки могат да се насочат към разширяване на самото приложение. В бъдещ план на разработки се предполага да се покрият и по-ниски версии на андроид системата. Като научно-приложна бъдеща задача може да се формулира оценката на сигурността на предаване на съобщения в защитената виртуална среда.

Литература

[1]. Signing messages, http://en.wikipedia.org/wiki/RSA_(cryptosystem)
[2]. RSA Class, http://www.tma.dk/rsa/

За контакти:
Студент, Цеко Ивайлов Цеков
катедра „Компютърни науки и Технологии”
Технически университет - Варна
E-mail: t.tsekov@sintex-group.com

Студент, Никола Стоянов Обретенов
катедра „Компютърни науки и Технологии”
Технически университет - Варна
E-mail: nikola_obretenov@mail.bg

гл. ас. д-р Милена Николова Карова
катедра „Компютърни науки и технологии”
Технически университет - Варна
E-mail: milena.karova@gmail.com

ПОДХОД ЗА РАЗРАБОТВАНЕ НА БОТНЕТ

Юлия А. Алексиева

Резюме: Ботнет са една от най-развиващите се заплахи за сигурността в Интернет, а скритата им IRC технология за размяна на файлове, атака към сървъри и DDoS атаки, ги прави изключително опасни, трудни за засичане и предотвратяване. Затова ботнет са нарастваща опасност за Интернет сигурността и стабилността и трябва да се познават в детайли, за да могат да бъдат спрени. Този доклад предлага разработка на IRC ботнет чрез програмата Windrop, като са показани основните конфигурационни настройки. Представя се и един начин за предотвратяване на недоброномерен windrop процес върху определен сървър.
Ключови думи: ботнет, ботнет атаки, IRC мрежа, хъб бот

An Approach for Developing a Botnet

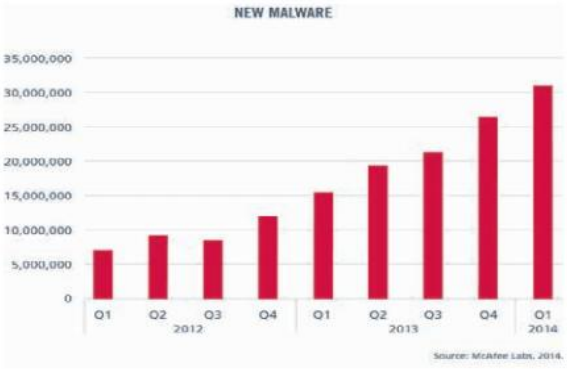
Yulia A. Aleksieva

Abstract: Botnets are one of the most advancing security threats on the Internet and their hidden IRC technology to exchange files, server and DDoS attacks makes them extremely dangerous, difficult to detect and prevent. Therefore botnets are a growing threat to Internet security and stability and should be known in detail, so that they can be stopped. This report proposes the development of IRC botnet through the Windrop program, by showing the basic configuration settings. A way of preventing malicious windrop process on a particular server is presented.
Keywords: botnet, botnet attacks, IRC network, hub bot

1. Въведение

При използването на Интернет технологии в ежедневните комуникации, потребителите са изложени на редица заплахи, свързани със сигурността на: тяхната самоличност, предавани и получавани данни и всякаква лична, поверителна информация. Кибер-престъпниците вече са способни да осъществят усъвършенствани атаки към мрежовата инфраструктура чрез няколко отдалечени хостове, като действията им са мотивирани от финансови и политически цели [5].

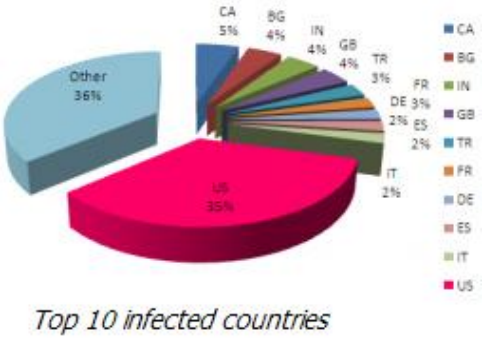
В глобален мащаб нови неизвестни атаки от зловредни кодове, в частност ботнет инфекции и заплахи, съгласно [6], са се увеличили рязко през последната четвъртина на 2013 и първата четвъртина на 2014, покачвайки внушителните си размери с над 32%, както е показано на фигура 1.



Фиг. 1. Инфекции от нов зловреден код - McAfee Threat Labs [6]

Според проучване на Symantec и Bkav Security Labs [7], в момента една от най-големите известни ботнет заплахи е бързо разпространяващият се Zeus ботнет. Световните мащабни статистики показват, че България е пряко засегната от зловредната заплаха, бивайки трета по процент на заразени компютри от Zeus ботнет (към август 2014), както е показано на фигура 2.

No	Countries	Infected machines	Percentage
1	United States	6505	34.70%
2	Canada	985	5.30%
3	Bulgaria	799	4.30%
4	India	785	4.20%
5	United Kingdom	741	4.00%
6	Turkey	529	2.80%
7	France	435	2.30%
8	Germany	400	2.10%
9	Spain	399	2.10%
10	Italy	398	2.10%



Фиг. 2. Топ 10 заразени страни от Zeus Ботнет – Bkav Security Labs [7]

Нарастването на Ботнет заплахите влияе негативно на сигурната комуникация в мрежите и изисква непрестанно усилие да се осигуряват техники, засичащи Ботнет, които са достатъчно всеобхватни. Това е от изключително значение за съвременния компютърно базиран бизнес. За да бъде изпълнена качествено тази задача и да бъдат създадени адекватни техники за откриването и евентуалното прекратяване на този вид заплахи, се налага нуждата от подробното им познаване. Вникването в същността на ботнета и разбиране на дейността му ще доведе до ефикасна и разумна техника за засичане на ботнет. Налага се необходимостта от създаване на качествени специализирани приложения за симулации на Ботнет атаки, като за момента е известен един единствен такъв продукт – Bonesi [9].

2. Проблемът Ботнет

Във времето Ботнет са претърпяли развитие от прости генератори на спам до основно средство за масови криминални операции. Ботнет са замесени в кражби на самоличности и номера на кредитни карти, различни форми на шпионаж, DoS (отказ на обслужване) атаки, атаки към несигурни продукти от новия дигитален свят – поставяйки в опасност модерното общество и развиващата се и разширяваща икономика.

Бот е машина, на която зловреден компютърен софтуер е бил инсталиран, като той може да се контролира самостоятелно и автоматично. Ботнет представлява група от ботове (съвкупност от заразени компютри, познати като зомби компютри), изпълняващи зловреден софтуер, бивайки контролирани от един хакер – общоизвестен като botmaster (главен бот), чрез централизирана инфраструктура за командване и контрол (C&C center) на атаката [1]. Инфраструктурата C&C, контролираща ботовете, е най-често от злонамерен характер и може нелегално да контролира изчислителните ресурси на ботовете. Ботнет използва и набира компютри, превръщайки ги в армия за кибер атака, за да бъдат използвани за спам, фалшиви уеб сайтове, DDoS атаки, вируси, червеи, задни врати, събиране на информация чрез фишинг и измами.

Според уебсайта SearchSecurity.com, поддържан от руската Kaspersky Laboratory [8], Ботнет понастоящем представлява най-голямата заплаха за Интернет. Проучванията твърдят, че 11% от свързаните към Интернет над 650 милиона компютри са ботове.

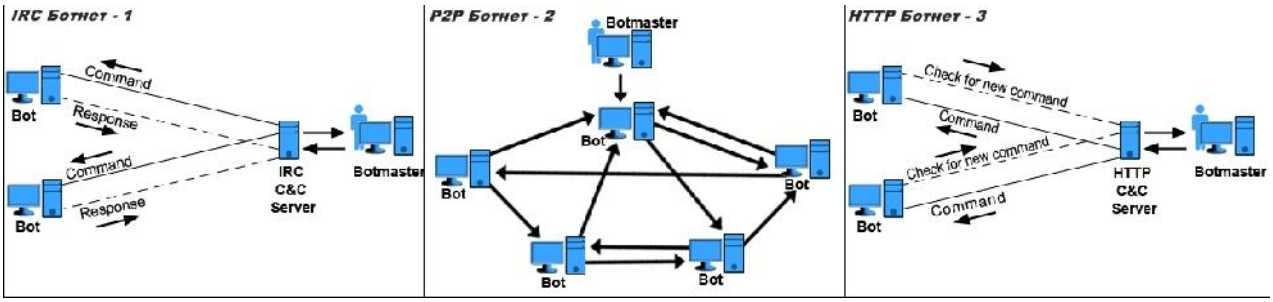
Комбинацията на Ботнет със съвременни технологии като IRC, HTTP и P2P (peer-to-peer) дава възможност безшумно да организират скритата си тактика в легални потребителски приложения. Това дава основание на изследователите да разделят условно

ботнет в три главни разновидности спрямо структурата на C&C сървъра, който използват и системите, които заразяват – IRC ботнет (централизирана архитектура), HTTP ботнет (централизирана архитектура) и P2P ботнет (децентрализирана архитектура) [3]. Чрез всяка от трите разновидности има възможност да се извършат различни атаки. Модерните ботнет технологии имат предимството да съчетават тези разновидности, превръщайки засичането на ботнет атаките много по-сложно.

IRC ботнет – първо поколение ботнет. Internet Relay Chat (IRC) се използва като инфраструктурен C&C протокол, чрез който главния бот разпространява команди към своята бот армия. IRC е чат система, която осигурява едно-към-едно и едно-към-много мигновени съобщения през Интернет. Нападателят може да използвате или публичен IRC сървър, или да изгради свой собствен сървър за целите на C&C. Ботовете са конфигурирани да се свържат обратно към C&C сървъра, очаквайки от който и да е конфигуриран порт да получат команди и да я изпълнят. Основната слабост на този тип C&C структура е лесната възможност да бъде идентифициран сървърът и местоположението му, а оттам съответно да се разстрои цялата ботнет мрежа. Действието на IRC ботнета е показано на фиг. 3.1.

P2P ботнет – второ поколение ботнет. P2P ботнет има повече предимства от IRC ботнета, тъй като командващият и контролният център C&C е разположен също като при P2P устройствата, които споделят файлове по Интернет. P2P Ботнет предполага всяка участваща компрометирана машина да действа като партньор (peer) за останалите и командите се изпращат помежду машините, а не всяка да очаква команда от C&C сървъра, както при IRC ботнета. Действието на P2P ботнета е показано на фигура 3.2.

HTTP ботнет – трето поколение ботнет. Протоколът HTTP се използва за публикуване на команди на определени уеб сървъри. Вместо да остават в свързания режим, HTTP ботовете периодично посещават определени уеб сървъри, за да получат актуализации или нови команди. Главните ботове използват HTTP протокола, зад който да скрият своята дейност сред нормалните уеб потоци и лесно да избегнат сегашните методи за откриване, като защитни стени. Действието на HTTP ботнета е показано на фигура 3.3.



Фиг. 3. Действие на 3-те вида ботнет

3. Разработване на ботнет

A. Съвременни разработки

Ботнет имат богата история, въпреки че представляват най-скоро появилата се Интернет заплаха, но за сметка на това и най-развиваща, усложняваща и все по-широко разпространена. Произходът на ботовете се базира на IRC мрежите. IRC ботнет са забелязани точно преди началото на новото хилядолетие. Въвежда се концепцията на заразена машина, свързваща се към Internet Relay Chat (IRC) канал, за да „слуша" за злонамерени команди. Бързо започва развитието им и се програмират да изпълняват доброжелателни функции като улесняване на администрирането на канал или играенето на онлайн игри, но повечето били заразени ботове изпращащи спам съобщения, DoS атаки. През 2013 Bernhard Waldecker [3] дава ясно класификация на техниките, засичащи IRC ботнет – хост-базирани, мрежово-базирани и комбинирани. По късно тезата му е доразвита от Bahari Belaton [4], като

проучването твърди, че предложеният модел може да открие всички заразени IRC бот елементи, да посочи тяхното настоящо състояние на атака, да филтрира злонамерени IRC съобщения и да разкрие поведението на ботнет независимо от ботнет комуникационния протокол с известен процент фалшив положителен сигнал.

B. Eggdrop ботнет и Windrop ботнет

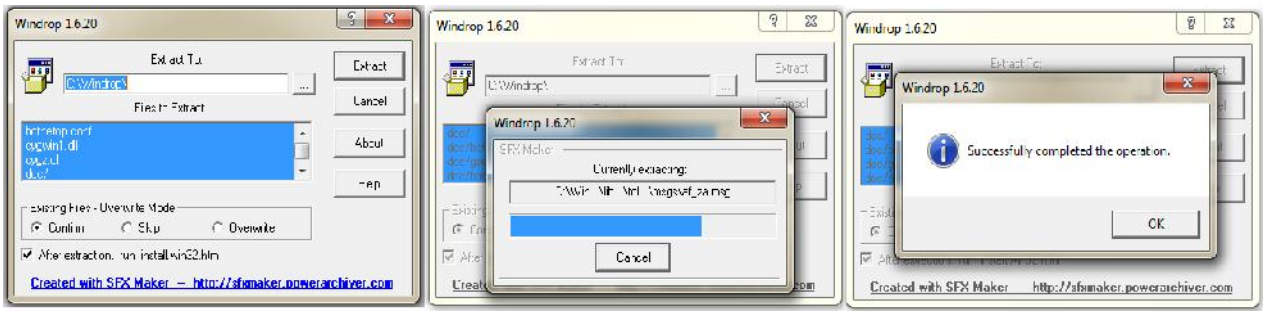
IRC ботнет е най-старият известен ботнет, но въпреки това той все още се проучва, за да се предотврати неговото действие. На практика IRC ботнет е най-лесно да бъде засечен, но на теория той остава най-голямата ботнет заплаха, тъй като лесно и бързо се разпространява.

В днешно време Интернет е пълен с различни tool kit-ове за разработване на различни ботнети – някои платени, други не. Това само показва колко сериозна е заплахата от ботнет, тъй като всеки, който има известни компютърни умения може да разработи собствен уникален ботнет. Най-известното средство за създаване и разпространяване на IRC ботнет е Eggdrop. Тази програма дава възможност за разработване на IRC ботове. Eggdrop работи под операционна система Linux, но има своя дъщерна програма – Windrop, абсолютно идентична с Eggdrop, която работи под Windows ОС.

Един IRC бот е програма, която стои в IRC канал и извършва автоматизирано задачи, докато изглежда точно като нормален потребител на канала. Една от основните опции на Windrop е свързването между ботовете, което позволява да се създаде собствен ботнет. Един ботнет може да се състои от минимум 2 свързани бота до хиляди свързани ботове. Най-важното свойство на свързаните ботове е споделянето на потребителска база данни (userfile.user). С тази особеност, ботовете могат да поддържат техните потребителски бази данни, списъците за отхвърляне и списъците с игнорирането на бота, синхронизирани помежду си.

C. Ботнет

Експерименталната разработка е извършена на конфигурация с процесор Intel i3M330, 2.13 GHz, DDR RAM 3GB, Windows 7 32-bit операционна система. Използвани са следните програми: Windrop, mIRC, WireShark. Създаденият за целите на експеримента бот YuliaBot не е зловреден.



Фиг. 4. Инсталационен файл на Windrop

След стартирането на инсталационния файл на Windrop (фигура 4), от папката, в която е инсталиран бота, се отваря конфигурационният файл windrop.conf с текстов редактор. Файлът трябва да бъде внимателно прочетен, за да могат да бъдат направени необходимите настройки. Тъй като настройките са няколко на брой, са показани в таблица 1– първоначалният им вид (колона 1) и съответстващите промени (колона 2). Тези действия се извършват, за да бъде създаден главния бот (hub bot) и той да може да бъде стартиран. В конфигурационния файл има над 1000 различни настройки за бота, но тук са показани само основните. Записва се пълният път до изпълнимия файл на windrop. Избира се име на бота, съответстващият му администратор, IRC мрежата, към която се свързва и името на потребителския файл (1-5). Записва се външният IP адрес, а не адресът от вътрешната мрежа (6). Задава се потребителското име на собственика на бота – за да се подсигури ботът да

отговаря само на неговите команди (7). Само притежателят на бота има пълен контрол върху него. Избира се име на chan файла и на канала, към който собственика иска да присъедини бота си (9-10). В случая е създаден отделен IRC канал (към съответната мрежа) само за изследователска цел - #MyFirstBot. Избира се към коя IRC мрежа от изброените ще се свърже бота, иначе ботът няма да се свърже към съответния IRC сървър (11). Избира се потребителско име на бота, също така и резервно, ако първото се окаже недостъпно (12,13). Задава се името на сървъра и съответния порт, към който да се закачи бота (15). В IRC мрежата съществуват така наречените IRCops (IRC проследяващи), които проверяват за наличието на ботове, най-често с командата hello, за това настройките на бота се променят и той не отговаря на hello, а на новата дума – EHO (16).

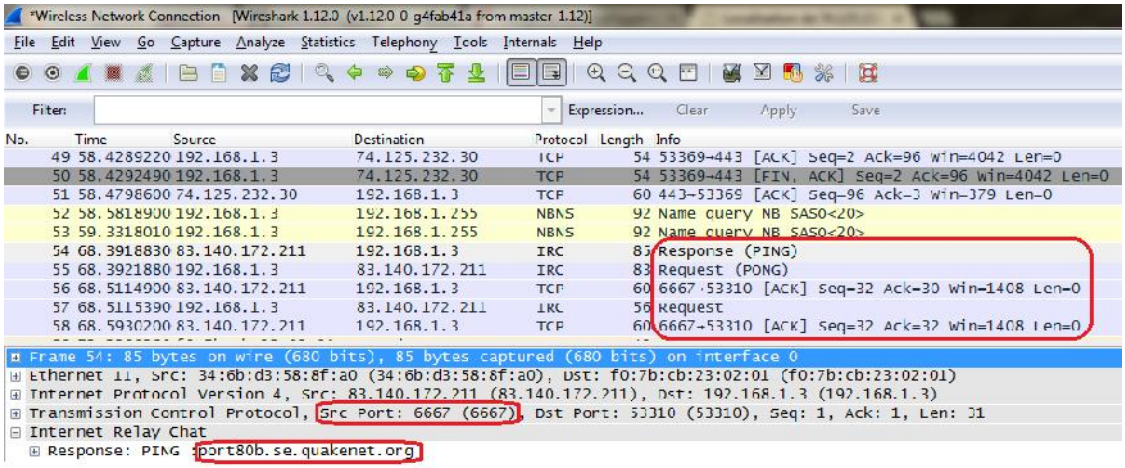
Таблица 1. Настройки в windrop.conf

№	windrop.conf	botYulia.conf
1	/path/to/executable/eggdrop	C:\Windrop\eggdrop.exe
2	set username "lamest"	set username "Yulia"
3	set admin "Lamer <email: lamer@lamest.lame.org>"	set admin "Yulia <email: iulia93@abv.bg>"
4	set network "I.didn't.edit.my.config.file.net"	set network "Quakenet"
5	set userfile "LamestBot.user"	set userfile "YuliaBot.user"
6	#set nat-ip "127.0.0.1"	set nat-ip " 46.10.19.24"
7	#set owner "MrLame, MrsLame"	set owner "yuli"
8	die "Please make sure you edit your config file completely."	-
9	set chanfile "LamestBot.chan"	set chanfile "YuliaBot.chan"
10	#channel add #lamest	#channel add #MyFirstBot
11	set net-type 0	set net-type 5
12	set nick "Lamestbot"	set nick "YuliaBot"
13	set altnick "Llamab?t"	set altnick "YuliaBot?"
14	set realname "/msg LamestBot hello"	set realname "/msg YuliaBot hello"
15	set servers { you.need.to.change.this:6667 another.example.com:7000:password }	set servers { port80a.se.quakenet.org:6667 port80b.se.quakenet.org:6667 port80c.se.quakenet.org:6667 }
16	#unbind msg - hello *msg:hello #bind msg - myword *msg:hello	unbind msg - hello *msg:hello bind msg - EHO *msg:hello

Фиг. 5. Генериране на бота

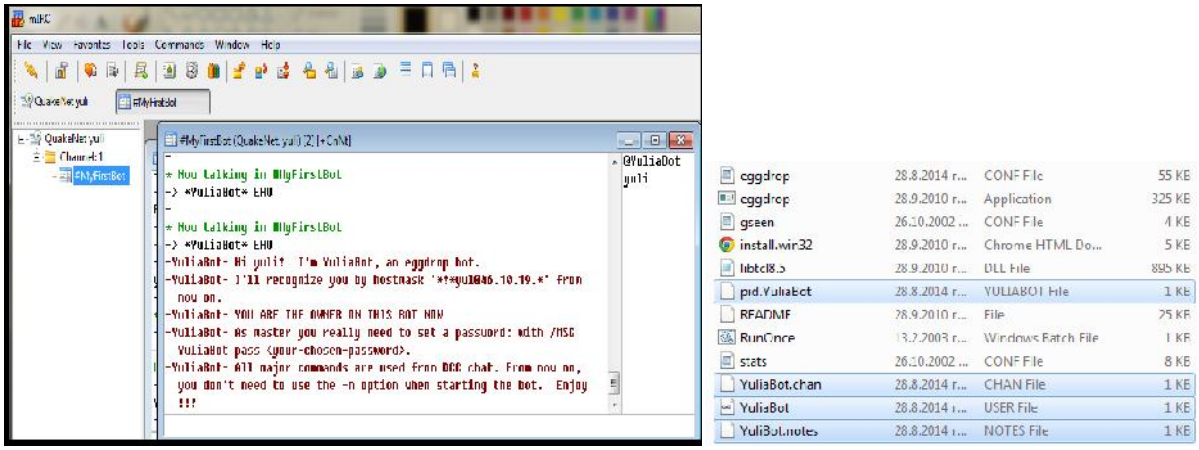
След като в конфигурационния файл са попълнени настройките на бота, той трябва да бъде стартиран. Първото стартиране става от RunOnce.bat или с команда в командния промпт (фигура 5). Ако ботът е активиран, той започва комуникация. На фигура 6 може да се види,

че сървърът, към който е свързан компютърът, е port80c.se.quakenet.org, а портът на комуникацията е 6667, съгласно направените настройки от таблица 1. Ако в конфигурационния файл няма допуснати грешки, потребителят влиза в упоменатия канал с упоменатото си потребителско име – като притежател на бота и изпраща команда /msg YuliaBot EHO и ако ботът е правилно свързан, отговаря (фигура 7- вляво).



Фиг. 6. Действащ бот

Ботът може да се подсигури с парола, за да отговаря само на притежателя си. След присвояването на бота се създават всички нужни файлове (фигура 7- вдясно).



Фиг. 7. Идентифициране на бота пред неговият собственик и файловете на бота

След създаването на главния бот може да се създаде и цялата ботнет мрежа. Хъб бот е главният бот, към който всички останали ботове в ботнета (leaf bots – ботове листа) се свързват. Хъб бот играе ролята на C&C сървър. За да се изгради ботнет мрежа, трябва да се добави всеки бот в потребителската база данни на всеки от останалите, след това да се зададат различни флагове за всеки от тях, които ще определят кой кой е и каква дейност трябва да извършва.

Първо, в конфигурационния файл на бота се записват всички канали, в които ботовете да синхронизират информацията помежду си (напр. потребителски флагове, банове и т.н). Флаговете на бота определят правата му. Стъпките за конфигуриране са представени в таблица 2. Избира се име на ботнет (1). Избира се порт (2), на който бота да „слуша” за команда, като порта трябва да е между 1025 и 65535. В конфигурационния файл е посочен порт 3333, но ясно е упоменато, че този порт не трябва да се използва, т.к. в мрежовото общество е отбелязан като порт за вируси. Включва се режим за споделяне (3). Настройка (4) е с цел при споделяне на потребителски файлове, да не се правят промени в самия файл, като

се предлагат и други конфигурации, но тази е най-сигурна за хъб бота. Включва се режим за компресия, за да може да се запазва голямо количество данни в потребителските файлове (5). Задава се режим за възможност за изпращане на данни (най-елементарните ботове просто изпращат Троянски коне в каналите, към които са свързани чрез тази настройка) – 6,7.

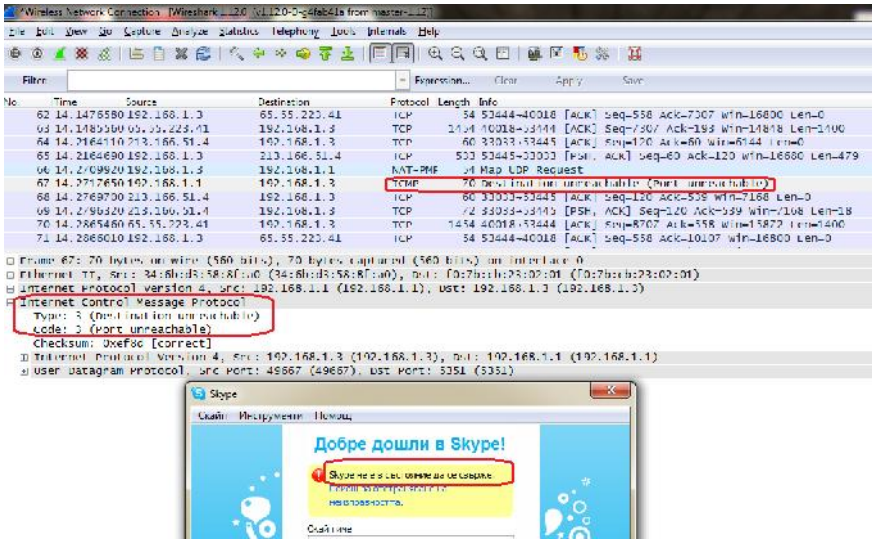
Таблица 2. Настройки в windrop.conf за ботнет

№	windrop.conf	botYulia.conf
1	`#set botnet-nick "LlamaBot"`	`#set botnet-nick "YuliaBotNet"`
2	`#listen 3333 all`	`#listen 7271 all`
3	`#loadmodule share`	`loadmodule share`
4	`#set private-user 0`	`set private-user 0`
5	`#loadmodule compress`	`loadmodule compress`
6	`#loadmodule filesys`	`loadmodule filesys`
7	`#loadmodule transfer`	`loadmodule transfer`

За свързването на два бота е нужно да се знаят адресите и портовете на съответните ботове. Нов бот се добавя с командата '.+bot YuliaBot irc.org:7271'.

D. Поведение на системата при наличие на бот

След генерирането на бота, той остава активен, дори и неговият собственик да се е изключил от IRC мрежата. Например стартирането на Skype става невъзможно, тъй като използва едни и същи портове с бота. Това може да се наблюдава със снифера Wireshark, както е представено на фиг.8. Съобщението за грешка (на ICMP протокола) се дължи на недостижим порт 5351, който вече е зает от бота. Това налага спирането му.



Фиг. 8. Блокиране на други приложения от бота

E. Премахване на бот от сървър след зараза

Най-честата употреба на този вид IRC ботове е да чакат команда, зададена от главния бот, или от друг оторизиран потребител в даден IRC канал. Когато бота получи тази команда, той и съответно останалите ботове, част от ботнета, атакуват отдалечен сървър, като постоянно се свързват към него и така сървърът не може да функционира правилно – ботнета извършва DDoS (Distributed Denial of Service) атака. Заразената машина се счита за дрон (drone).

За подсигуряването на заразен сървър са нужни няколко стъпки. Първо, windrop процесът трябва да бъде спрян. След това се спират и процесите, свързани с windrop, като се има предвид, че голяма част от процесите са маскирани. Следващата стъпка е да се изтрият съответните програми от сървъра, защото windrop има функционалността да се рестартира след като веднъж бъде спрян.

След като windrop бъде ограничен, трябва да се вземат мерки, за да не бъде заразен отново сървърът. Подсигуряването се извършва по следния начин: софтуерът на сървъра трябва да бъде обновен с най-новите версии, ненужните услуги трябва да бъдат спрени и премахнати и да се инсталира качествена защитна стена.

4. Заключение

По подразбиране комуникациите през Интернет са отворени и неконтролирани. Този факт е в конфликт с нуждите на съвременния компютърно-базиран бизнес, който изисква конфиденциалност на предаваните данни. Нарастването на Ботнет заплахите влияие негативно на сигурната комуникация в мрежите и изисква непрестанно усилие да се осигуряват техники, засичащи Ботнет, които са достатъчно всеобхватни. Затова познаването на този вид заплахи е от ключово значение за създаването на адекватни техники за откриването им.

В този доклад е представено разбработването на ботнет и е създадена безопасна IRC ботнет мрежа чрез програмата Windrop. Също така е представен и начин за предотвратяване на дейността на windrop ботнет върху заразен сървър.

Като цел на бъдеща работа се предвижда разработването на HTTP ботнет и P2P ботнет, сравнение между трите разновидности и анализ на ефективността на заразяване при симулации на различни ботнет атаки. Като по-далечна цел е създаване на приложение за симулиране на различни Ботнет атаки.

Литература

[1]. Agarwal, S., Performance Analysis of Peer-To-Peer Botnets using “The Storm Botnet" as an Exemplar, University of Victoria, 2010, стр. 2-7, 71,72

[2]. Bégin, F., BYOB: Build Your Own Botnet and learn how to mitigate the threat posed by botnets, SANS Institute, The SANS Institute, 2011, стр. 1-4, 32-35

[3]. Waldecker, B., A Review on IRC Botnet Detection and Defence, Austria, St. Poelten University of Applied Sciences, 2013, стр. 1-9

[4]. Belaton, B., A. H. R. Awadi, Multi-phase IRC Botnet and Botnet Behavior Detection Model. // International Journal of Computer Applications (0975 – 8887), March 2013, Volume 66– No.15, стр. 2,10

[5]. Алексиева, Ю. А., Критерии за техники, откриващи Ботнет. В: Годишник на ТУ-Варна, Студентска научна сесия – 2014, Университетско изд., 2014

[6]. McAfee, http://www.mcafee.com/uk/resources/reports/rp-quarterly-threat-q1-2014.pdf

[7]. Le Minh Hung, Tracking Zeus botnet which updates like Conficker, http://security.bkav.com/home/-/blogs/tracking-zeus-botnet-which-updates-like-conficker/normal

[8]. Search Security, http://searchsecurity.techtarget.com

[9]. Bonesi, the DDoS Botnet Simulator, https://code.google.com/p/bonesi/

За контакти:
Студент, Юлия А. Алексиева
катедра „Компютърни науки и технологии”
Технически университет Варна
E-mail: iulia93@abv.bg

СИСТЕМА ЗА МОНИТОРИНГ НА УСКОРИТЕЛНИЯ КОМПЛЕКС В ЦЕРН

Митко Д. Митев, Ивайло П. Пенев

Резюме: Докладът запознава със системата DIAMON, която изпълнява мониторинг на ускорителния комплекс в ЦЕРН. Разгледана е общата структура на системата и е направено функционално описание на някои от най-важните компоненти. Представени са резултати от тестване на производителността на ускорителния процес, получени чрез системата.
Ключови думи: Мониторинг, DIAMON, ЦЕРН.

System for Monitoring the Acceleration Complex at CERN

Mitko D. Mitev, Ivaylo P. Penev

Abstract: The paper presents the system DIAMON, performing monitoring of the acceleration complex at CERN. The common structure of the system is clarified and functional description of the most important components is made. Results from tests of the performance of the acceleration process, obtained by the system, are presented.
Keywords: Monitoring, DIAMON, CERN.

1. Въведение

Европейската организация за ядрени изследвания (ЦЕРН) е най-голямата лаборатория за физика на елементарните частици в света. Тя е основана през 1953 година и през 60 годишната и история са направени множество революционни открития като: Хигс бозона, PET скенерите, супер проводимите кабели, Интернетът и грид архитектурата [2].

ЦЕРН е пионер в областта на физиката, но това не би било възможно без технологичния бум в края на миналия век. В текущия момент в най-големия ускорител LHC се достигат енергии от 7 TeV, като след като завърши модернизацията се очакват 14 TeV. Детекторите достигат маса 12500 тона и габарити от 26x16 метра. Работи се при температура близка до абсолютната нула (-273.15) в радиоактивни условия. Използваната техника е една от най-добрите съществуващи в момента. Гарантирането на безопасността е от изключителна важност и компромиси в тази сфера не биват допускани. Спазват се стриктни протоколи, планове за действие и са съставени специални групи, грижещи се за безопасността, всичко е тествано и проверено, всяко устройство е подсигурено при отказ, средата е изолирана от кибер атаки, комплексът е построен на 100 метра под земята и е гарантирано, че няма да бъде оказана вреда на еко системата.

Както в повечето високо рискови индустриални предприятия и в ЦЕРН има множество контролни системи. Това са група от устройства (хардуер и софтуер), грижещи се за управлението и регулацията на друга група от устройства или системи. Стремежът е да бъде автоматизиран максимално всеки процес и да бъде гарантирана неговата безпроблемна работа. Поради изключителната голяма сложност и високия риск е изграден контролен център, където 24/7 на смени работят оператори. Те управляват комплекса от ускорители и следят за възникването на всякакви аномалии и дефекти. Ако бъде локализирано нестандартно събитие, е налична и 24 часова телефонна поддръжка за всяка от системите. За да могат операторите да извършват безпроблемно своята работа, са налични алармена система (LASER) и система за мониторинг (DIAMON).

При разпределените системи мониторингът е процес на събиране и съхранение на данните за текущото състояние на устройствата в ускорителя. Събраната информация е от изключително значение за следене на критични процеси, анализиране на причините, довели до повреди и бедствия, оценка на текущите системи и процеси и тяхното подобрение. Преди да се имплементира мониторингът, трябва да се вземат предвид следните фактори:

- Гарантиране, че той няма да доведе до загуба на производителността на системата, която се наблюдава. Процесът, който публикува данните за състоянието на системата, трябва да е оптимизиран и да оказва минимално влияние върху основните и функции.
- Осигуряване на надежден пренос на данните с подходящи протоколи, съответстващи на изискванията за сигурност, проводимост и бързодействие.
- Трябва да бъде извършен детайлен анализ за определяне на процесите, които трябва да бъдат следени и да се изготвят конфигурации за тях от съответните експерти.
- Трябва да бъде избрана подходяща архитектура и да се гарантира, че информацията може да бъде съхранена и анализирана.
- Трябва да бъде предвидена възможност за подобрение и лесно внедряване на нови модули. По този начин ще бъде гарантирана надеждната работа в бъдеще и ще могат да бъдат задоволени нови изисквания към системата.

2. Изложение

При разработването на DIAMON са взети под внимание всички горе споменати фактори. След внимателен анализ на всички изисквания и актуални концепции за дизайн е взето решение да се обединят усилията на DIAMON екипа с тези на TIM (Technical Infrastructure Monitoring) и да бъде създаден C^{2}MON сървърът. Той покрива всички нужди на двата проекта и позволява създаването на високо надеждни, мащабируеми и гъвкави системи за контрол.

2.1. Дизайн на системата

Използвана е традиционната трислойна архитектура, представена на фигура 1.

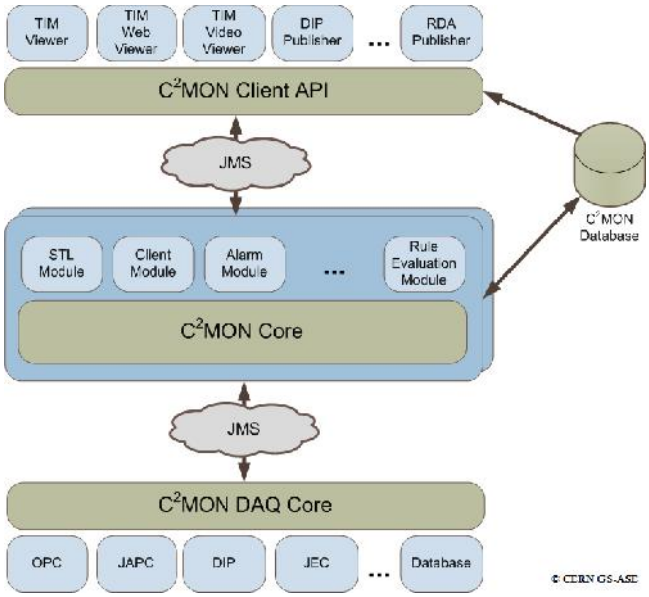
- Слоят за събиране на информация DAQ (Data Acquisition layer) е съставен от множество модули, с помощта на които се осъществява връзка между C^{2}MON и наблюдаваната системата. Разработчиците имат възможност да използват някои от съществуващите модули в репозиторито или да бъде създаден нов, в случаите когато наличните не съответстват с изискванията (различен протокол или устройства).
- Сървърната архитектура използва Java Spring базирани контейнери. Тя може да бъде стартирана в клъстер режим, като инстанциите могат да бъдат инсталирани на различни хардуерни системи за по-голяма надеждност. Вътрешно сървърът е разделен на различни модули, част от които са незадължителни. Разработчиците могат да активират само тези, от които има нужда и в последствие да разширят функционалността при промяна на изискванията.
- Интерфейсният слой служи за достъп на сървърната функционалност (Client API). Той е съставен от множество процедури за достъп, които могат да бъдат използвани от клиентските приложения.

Освен C^{2}MON сървъра DIAMON системата включва още:

- База от данни, където се съхраняват текущите конфигурации за наблюдаваните системи.
- APEX приложение, което позволява лесно модифициране на конфигурациите от страна на потребителя и синхронизацията им със сървъра. Обикновено това се извършва от лицето, което е отговорно за домейна. Поддържа се асинхронен режим,

след всяка модификация или да бъде делегирано на ежеседмичната процедура за ъпдейт.

- DIAMON GUI е клиентско приложение, визуализиращо информацията, пристигаща от сървъра.
- DIAMON Viewer е също клиентско приложение, което дава различен поглед над информацията. То има няколко удобни опции за тестване при разработване. Също така предоставя възможност за интегриране на информацията от двата DIAMON и TIM в бъдеще.
- DIAMON Dashboard Editor е специализирано клиентско приложение, насочено към разработчиците на табла и други специализирани графични репрезентации на информацията. Таблата могат да бъдат интегрирани с DIAMON Viewer и DIAMON GUI.



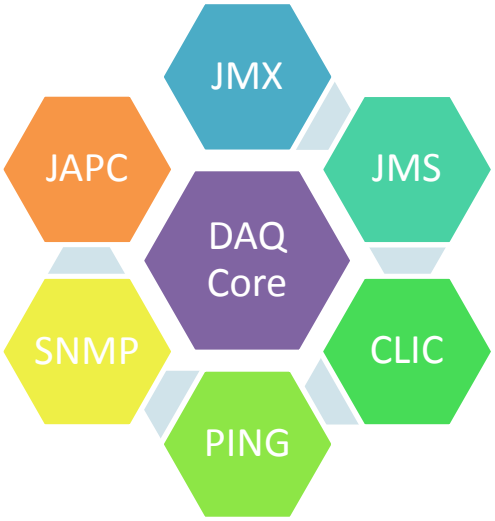
Фиг. 1. C^{2}MON Архитектура

2.2. Модул за събиране на информация - Data Acquisition module (DAQ)

Едно от основните изисквания към сървъра е да предоставя възможност за интегриране на информация, събрана от множество разнообразни източници. Различията може да бъдат в хардуера, софтуера, протоколите за комуникация и други. C^{2}MON е изграден по начин, гарантиращ лесно и бързо интегриране на информацията от всякакви видове устройства.

DAQ модулите се грижат за осъществяване на връзката с устройствата и комуникацията със сървъра (фигура 2). Също така данните биват конвертирани, валидирани и филтрирани, за да се гарантира, че сървърът разполага само актуална и надеждна информация и не се хаби изчислителна мощ за обработка на погрешни данни.

Фиг. 2. Диаграма на комуникацията между сървъра, DAQs и устройствата в C^{2}MON архитектурата



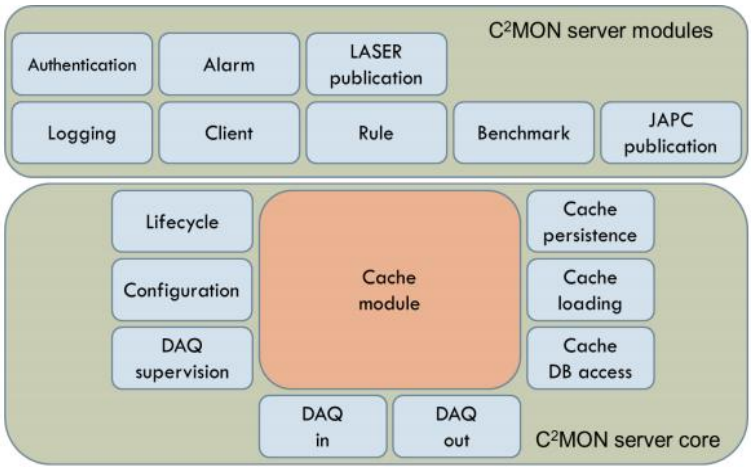
Фиг. 3. Архитектура на модулите за събиране на информация
(DAQ Architecture)

Модулите за събиране на информация се състоят от ядро (DAQ Core), предоставящо основната функционалност и специфични модули (DAQ Flavors), осъществяващи връзката между сървъра и устройствата от съответен тип (фигура 3). В ядрото е имплементирана основната функционалност на всички DAQ модули: комуникация, конфигуриране, валидация, конвертиране, филтриране, complex event processing (CEP). Специфични модули са например Oracle, SSH [4], SNMP и др.

2.3. Сървър

C^{2}MON сървърът е в основата на DIAMON [3]. Благодарение на добрата си архитектура осигурява надеждността и бързодействието, необходимо за мониторинга на комплекс с размерите на този в ЦЕРН.

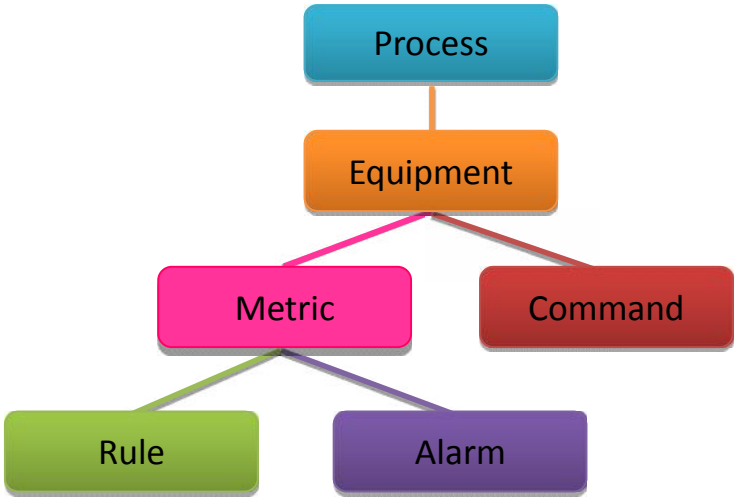
Сървърът е разработен на модулен принцип (фигура 4). Част от компонентите са задължителни за гарантиране на функционалността, а останалите са избирателни според проблема, пред който екипът е изправен.



Фиг. 4. C^{2}MON Сървърна архитектура

Съобразно добрите практики при разработване на системи за мониторин и контрол, е създадена йерархия от обекти. Те универсализират начина на представяне на следените

системи и устройства, техните параметри, команди и асоциираните правила и аларми (фигура 5).



Фиг. 5. Йерархия на елементите в C^{2}MON

2.5. Клиентски приложения

Клиентските приложения са всички програми, използващи C^{2}MON Client API. Те могат да бъдат разделени в две категории:

- Обслужващи програми (услуги);
- Графични приложения.

2.6. Услуги

Услугите са сравнително прости приложения, чиято цел е да предоставят достъп на други приложения до информацията в DIAMON. Типичен пример са публикуващите програми. Те използват клиентския интерфейс за абониране към метриките, представляващи интерес. При всеки ъпдейт или през определен времеви интервал данните се извличат, конвертират и препращат към другите системи [1].

2.7. Графични приложения

Графичните приложения са потребителски интерфейси към информацията, налична в DIAMON. Те имат различни характеристики, функции и биват използвани от различни групи от потребители:

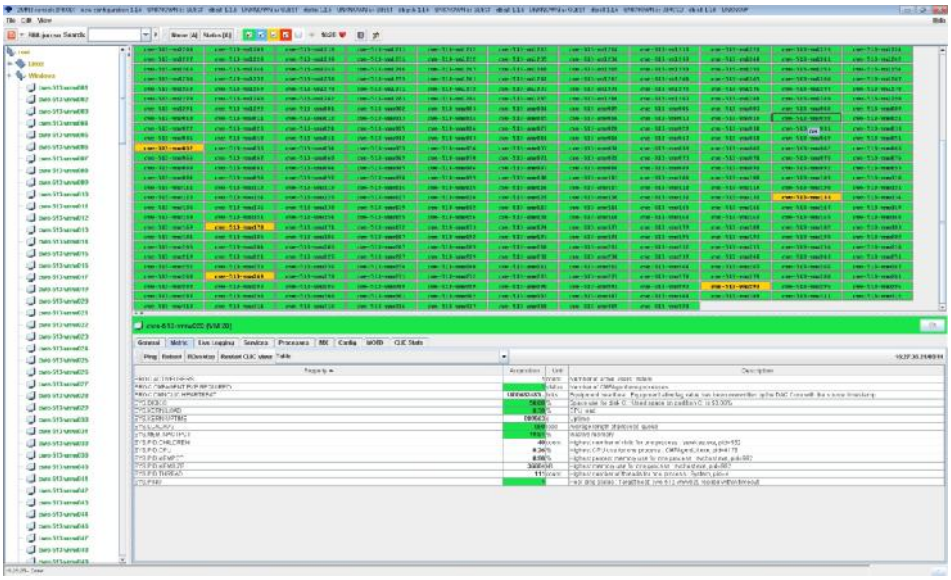
- C2MON Web Config Viewer;
- DIAMON Dashboard Editor & DIAMON Symbol Editor;
- DIAMON Viewer;
- DIAMON GUI;
- LASER.

2.8. DIAMON GUI

Това е основното графично приложение на DIAMON. Интерфейсът се състои от четири основни зони (фигура 6).

- Меню лента, където се намират бутоните за автентикация, статус бар, опции за сортиране, филтрация и менюто за историческа навигация;

- Дървовидна структура от където може да бъде селектирана желаната конфигурация или хост;
- Основен прозорец, визуализиращ текущия статус на всички хостове (или текущо филтрираните);
- Прозорец с множество табове, съдържащ детайлна информация за селектирания хост. Всеки от табовете представлява отделен плъгин и техният брой зависи от текущата конфигурация и тип на устройството.



Фиг. 6. Основен прозорец на DAIMON GUI

Налични са два режима на работа:

- Нормален: Използва се от операторите от контролната зала на ускорителния комплекс и показва текущото състояние на хостовете.
- Исторически: Използва се от експертите, които анализират изминали събития с цел подобряване на процесите или анализ на възникнали грешки.

3. Резултати

В настоящия момент се извършва мониторинг на приблизително 5000 устройства с 70000 метрики и 30000 правила. Успешното имплементиране на подхода за филтриране значително намалява натоварването на сървъра. В резултат от внедряване на описаната архитектура ъпдейтите са намалени от 500 в секунда до 200 в секунда. Средното натоварване на процесора е 4-5%, а използваната памет 500MB. Тези нива са много ниски в сравнение с реалния капацитет на приложението, измерен по време на тестовете за производителност. Това гарантира, че DIAMON ще може безпроблемно да се справи с натоварването при нормално опериране на LHC.

4. Заключение

Мониторинг системата е високо производителна, надеждна и сигурна. Тя предоставя възможност за следене на голям брой устройства, хисторизиране и анализиране на информацията. Комбинацията от модерни технологии, бързодействие и иновативен интерфейс улеснява операторите и им дава възможност да извършват ежедневните си задачи по оптимален и лесен начин. По този начин се гарантира безопастността на комплекса от

ускорители и се помага за постигане на мисиите на ЦЕРН. Някои от основните предимства на DIAMON са:

- Гарантиране на надеждната работа 24/7.
- Възможност да бъде инсталирана на множество видове хардуер. Позволява да се подменят сървърните компоненти и операционната система.
- Динамично реконфигуриране (без да се налага рестартиране). По този начин е предоставена възможност за лесно добавяне на нови източници на информация, която трябва да бъде следена.
- Пълна съвместимост с технологиите и системите, текущо използвани в ЦЕРН.

Литература

[1]. Eckel, B. Thinking in Java: The Definitive Introduction to Object-Oriented Programming in the Language of the World-Wide Web. Prentice Hall, 2006, ISBN-10: 0131872486
[2]. http://home.web.cern.ch/
[3]. http://c2mon.web.cern.ch/
[4]. http://www.ssh.com/

За контакти:
Инж. Митко Деянов Митев
Катедра „Компютърни науки и технологии“
Технически университет - Варна
E-mail: mitkodm@abv.bg

Ас. д-р инж. Ивайло Пламенов Пенев
Катедра „Компютърни науки и технологии“
Технически университет - Варна
E-mail: ivailo.penev@tu-varna.bg

РАЗРАБОТКА НА КРИПТОГРАФСКИ ПРОТОКОЛ ИЗПОЛЗВАЩ ПРЕДЛОЖЕН БЛОКОВ ШИФЪР И АПЕРИОДИЧНА СМЯНА НА КЛЮЧ

Сиво В. Даскалов

Резюме: Разработеният криптографски протокол прилага шифър с дължина на блока данни и текущия ключ 2^n бита. Алгоритъмът за криптиране за всеки блок се състои в размяната и инвертирането на отделни сегменти от некриптирания блок в зависимост от използвания ключ. Всеки ключ се използва привидно случаен брой пъти между 0 и 15, след което следващият генериран ключ се изпраща по мрежата криптиран, използвайки предишния. Първоначалният ключ се установява чрез използването на публичния ключ на реципиента.
Ключови думи: Апериодичен, блоков шифър, криптография, сегментиране, симетричен ключ

Cryptographic Protocol Using a Proposed Block Cipher and Aperiodic Key Replacement

Sivo V. Daskalov

Abstract: The proposed cryptographic protocol implements a sipher with block and key length of 2^n bits. The encryption algorithm consists of several stages each of which swaps or inverts different segments of the plaintext according to the current key. Each key is used a seemingly random number of times between 0 and 15, afterwards the next generated key is encrypted and transmitted using the previous one. The initial key is established using Public key encryption.
Keywords: Aperiodic, block cipher, cryptography, segmentation, symmetric key

1. Въведение

В настоящата ера криптографията е намерила широко приложение извън сферата на разузнаването. Тя се използва за защита на лични и корпоративни данни, внасяйки сигурност в електронния свят и особено в мрежата. В развитието на този раздел от математиката и информатиката се инвестират много усилия и финанси, в следствие на което са разработени много и ефективни криптографски протоколи. Най-използваните сред тях са RSA, SSL (Secure Sockets Layer), DES (Data Encryption Standard) и AES (Advanced Encryption Standard) [1]. Съвременната криптография цели да постигне четири неща. Първото е конфиденциалността, с други думи, информацията да бъде разбрана само от този, за когото е предназначена и от никой друг. Второто е целостта - недопускане на възможност данните да бъдат променени по някакъв начин при пренасянето им от подателя до получателя, без това да бъде открито. Третото е осигуряване на ясни механизми на свързване на шифрираните данни с получателя и подателя, така че никой от тях да не може на по-късен етап да отрече участието си в комуникацията. От съществена важност са и методите за идентификация или иначе казано подсигуряването, че кодираните данни ще попаднат точно при получателя си [3]. Целите на криптографията се постигат в общия случай чрез използването на алгоритъм, който трансформира изпращаните данни по такъв начин, че единствено планираният им получател да може да възстанови първоначалните данни. При симетричните алгоритми за криптиране като предложеният в този доклад, на кодиращата функция от страната на изпращащия и декодиращата функция от страната на получателя се подава еднакъв „ключ”, който е известен на двете страни в комуникацията [2]. Между участниците се изпраща кодирана информация, невъзможна за прочитане от трета страна.

2. Изложение

Описание на блоковия шифър и неговия алгоритъм на действие

Принципът на действие на шифъра е приложим върху блокове с различен размер, но равен на степен на числото 2. Въпреки това за демонстрация на действието на алгоритъма и по-нататъшни изследвания е избрана дължина на блока данни и ключа 64 бита.

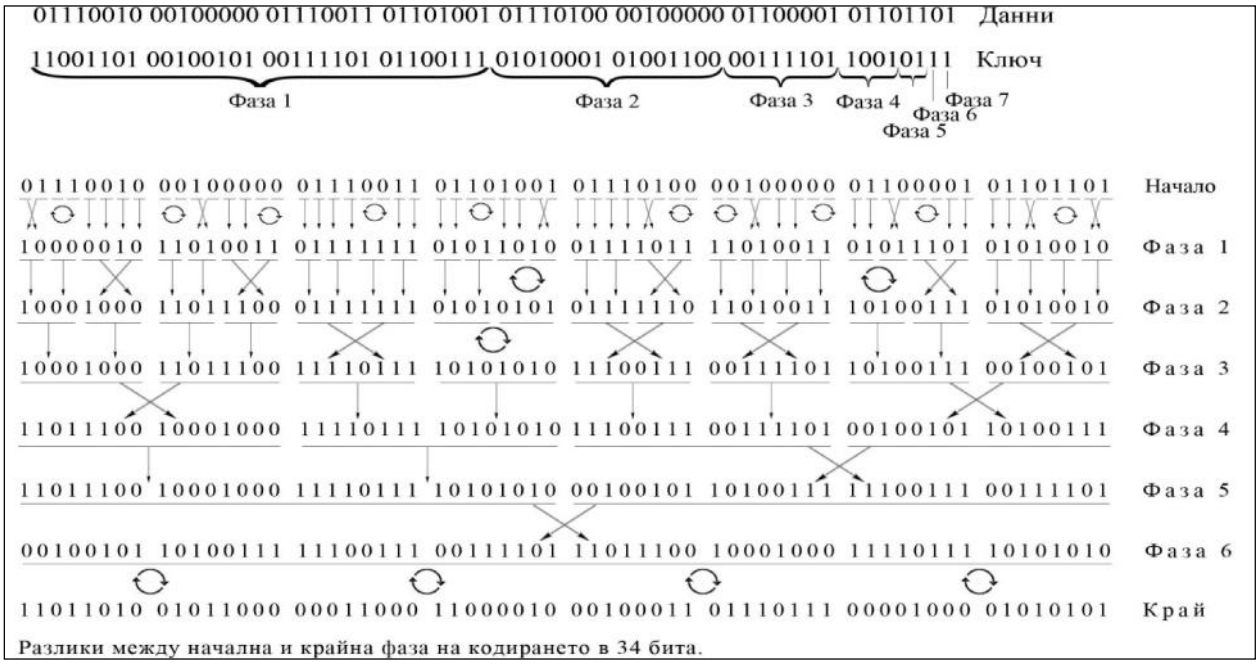
Действието на криптиращата функция се разделя на 6+1 фази. В шестте фази блока от данни се разделя на сегменти с дължина равна на 2^{n-1}, където n-номер на фазата. Итеративно се обхожда блока с данни и се обработват по двойки съседните сегменти. Обработката на всяка двойка сегменти определя един сегмент от по-горен ранг, такъв с двоен размер и заемащ определената от обработваните сегменти позиция в блока данни. Обработката на всяка двойка сегменти се управлява от определен бит от ключа, като са възможни три изхода от обработката:

a) Стойността на текущия бит на ключа е 0: Тогава между двата сегмента не се извършват операции и всеки бит от тях запазва мястото и стойността си.
b) Стойността на текущия бит на ключа е 1 и двата сегмента не са идентични: В този случай двата сегмента разменят местата си в блока от данни.
c) Стойността на текущия бит на ключа е 1 и двата сегмента са идентични: В този случай всеки бит от двата сегмента получава инверсната си стойност.

Последният бит от ключа се използва за финално инвертиране на всички битове от блока, ако стойността му е 1.

Таблица 1. Разпределение на използването на битовете от ключа по фази на криптирането

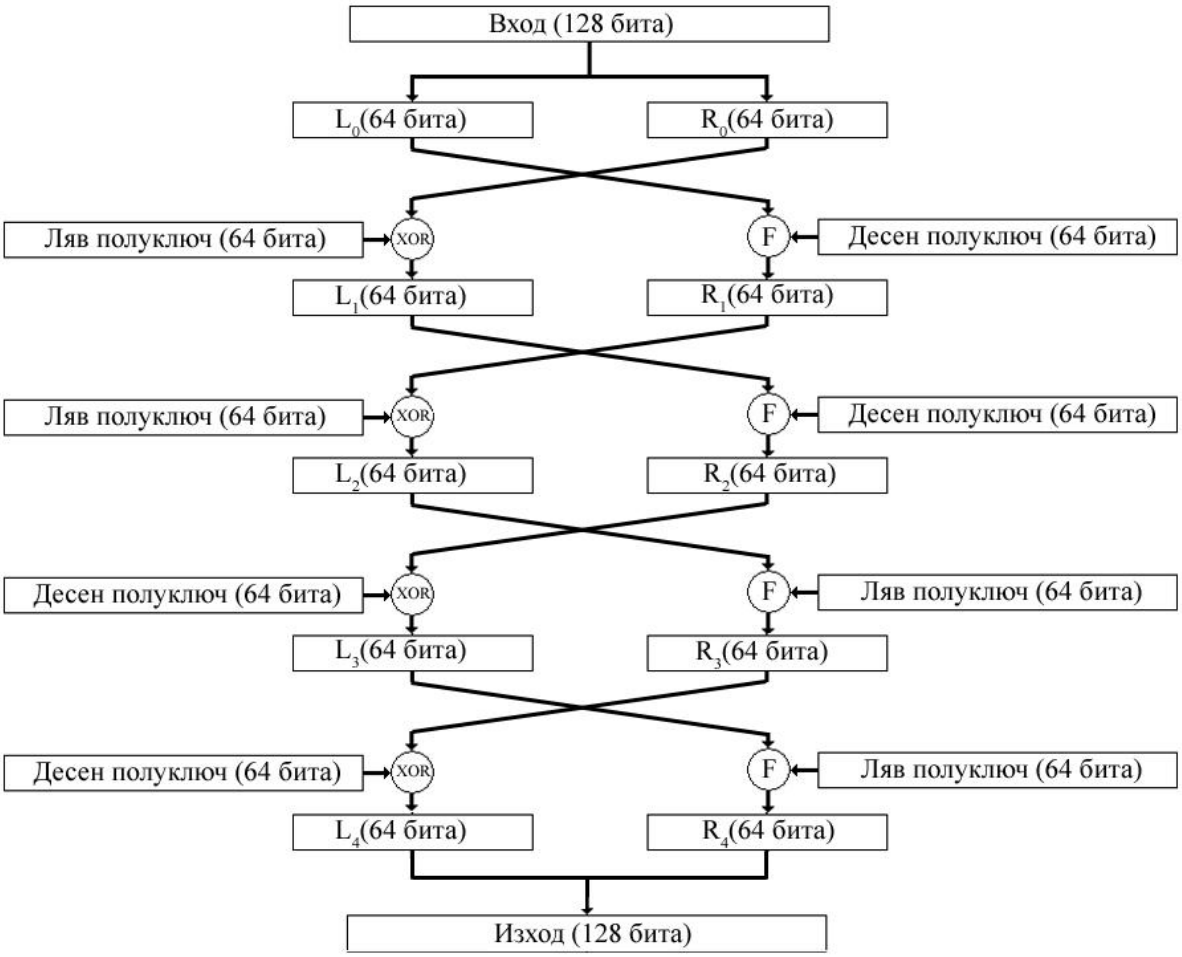
Номер на фаза	Размер на сегмент	Брой използвани битове от ключ	Номер на битове
1	1	32	1-32
2	2	16	33-48
3	4	8	49-56
4	8	4	57-60
5	16	2	61-62
6	32	1	63



Фиг. 1. Демонстрация на различните фази от криптиращия алгоритъм

Разширен многостепенен шифър с удвоена дължина на блока

Гореописаният блоков шифър за всяка фаза на кодирането се прилага върху половината от удвоения блок, използвайки половината от текущия ключ. Върху необработената от шифъра блокова половина се прилага операцията XOR, използвайки втората половина от текущия ключ. Тази фаза от генерирането на крипотекста се повтаря четири пъти, до пълното изчерпване на комбинациите полуключ – полублок, както е показано на фигура 2. Броят на повторенията при изпълнението на гореописаните фази е произволен и позволява модификация на поредността и характера на приложените стъпки.



Фиг. 2. Демонстриране на многостепенното криптиране

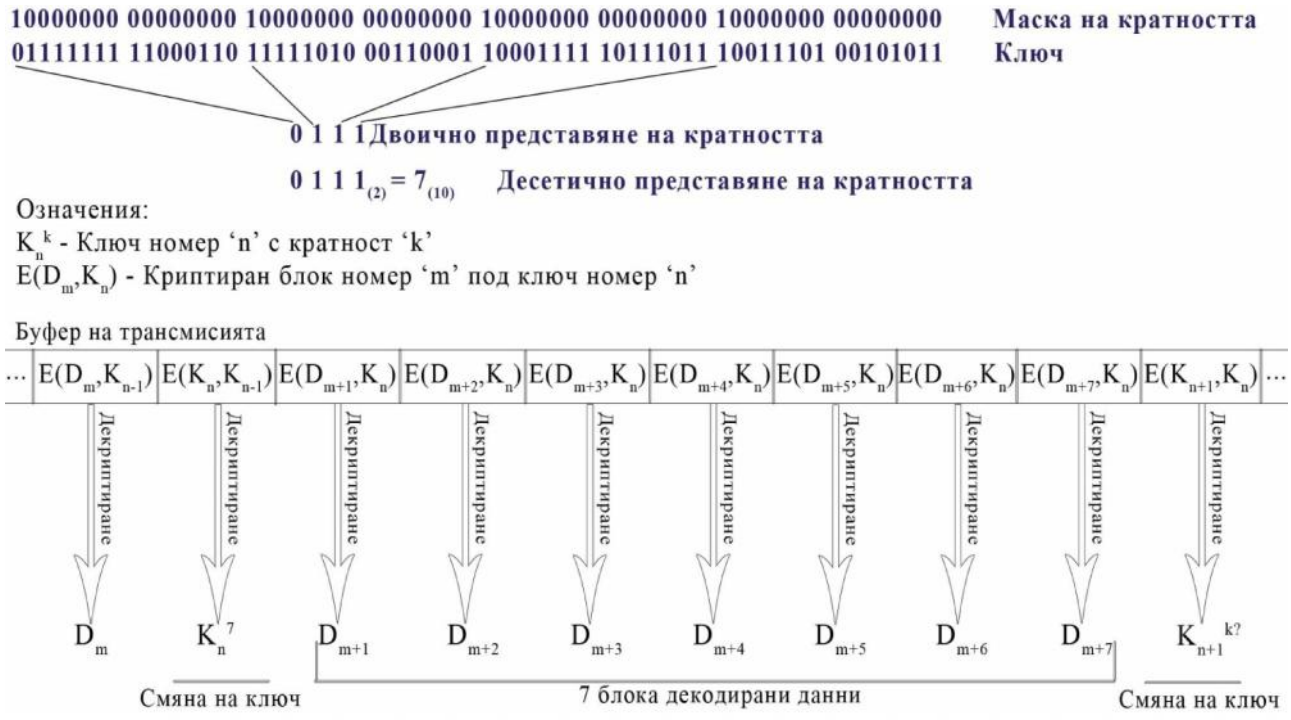
Генериране на първоначален ключ

Проблемът с генерирането на първоначален ключ може да бъде решен чрез имплементацията на асиметрични криптографски похвати като използването на публичен и частен ключ. След първоначалното установяване на ключ безопасна трансмисия на всеки следващ генериран ключ може да се извърши по данновия канал, както следва да се покаже в следващата част от този доклад.

Апериодична смяна на ключовете

От всеки ключ са избрани четири бита, които разглеждани като двоично число определят за кодирането на колко блока от данни ще се използва текущият ключ. Този параметър на ключа се дефинира като негова кратност и е в границите [0,15]. Позицията на

тези битове е известна и на двете страни и след прехвърлянето на определения от текущия ключ брой блокове по канала предстои изпращане на нов ключ. Той е генериран от активната страна и за подобряване на сигурността всеки негов бит има 50% вероятност да заема една от двете логически стойности. Резултантните ключове са с неопределен брой единици вариращ най-често между 28 и 36, но с математическо очакване за тези характеристика равно на 32 (половината от битовете на ключа). След изпращането на определения от кратността на ключа брой блокове следва смяна на ключа. Текущият ключ се използва за последен път при кодирането на следващия генериран ключ за изпращането му към получателя. Кратността на новия ключ е определена и преди генериране на следващ ключ се изпращат блокове данни, равни по брой на тази кратност. Те са кодирани чрез използването на новия ключ и могат да бъдат успешно декодирани с него. Въпреки че кратността на ключовете е в границите [0,15], те се използват за криптирането и декриптирането на [1,16] блока, последният от които представлява заменящият ключ. Този процес продължава до приключване на предаването на данни, а случайния характер на кратността на блоковете увеличава сигурността чрез подобряване на непредвидимостта на характера на предаваните блокове. Трета страна не би различила криптиран ключ от криптирани данни, което допълнително затруднява криптоанализа на протокола.



Фиг. 3. Демонстрация на апериодичната смяна на ключовете и потока от криптирани данни

3. Резултати

A) Постановка на изследванията

Алгоритъмът е реализиран на езика C++. На една машина се симулира криптирането и декриптирането на потока от данни. Параметри на изчислителя:

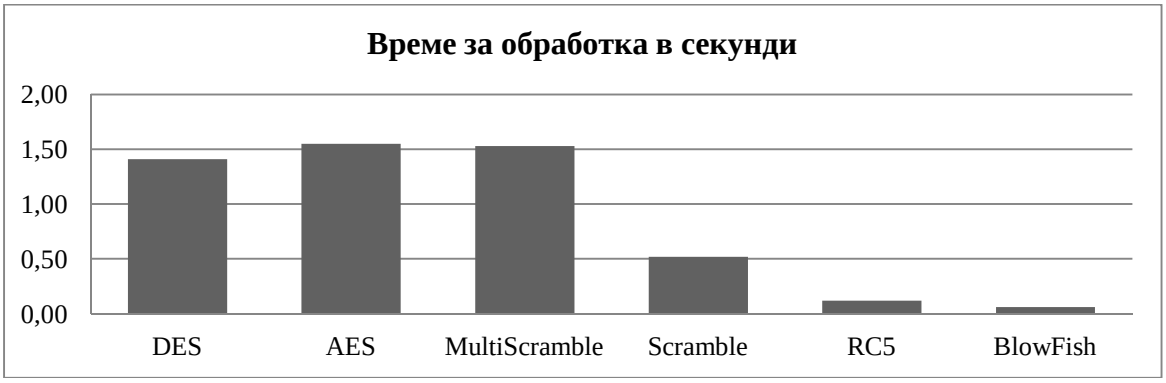
a) Процесор: Intel Core i5-3570 3.4GHz Quad-Core Processor
b) Памет: GeIL EVO Corsa DDR3 2133MHz 8GB

B) Сравнителна скорост на трансмисия

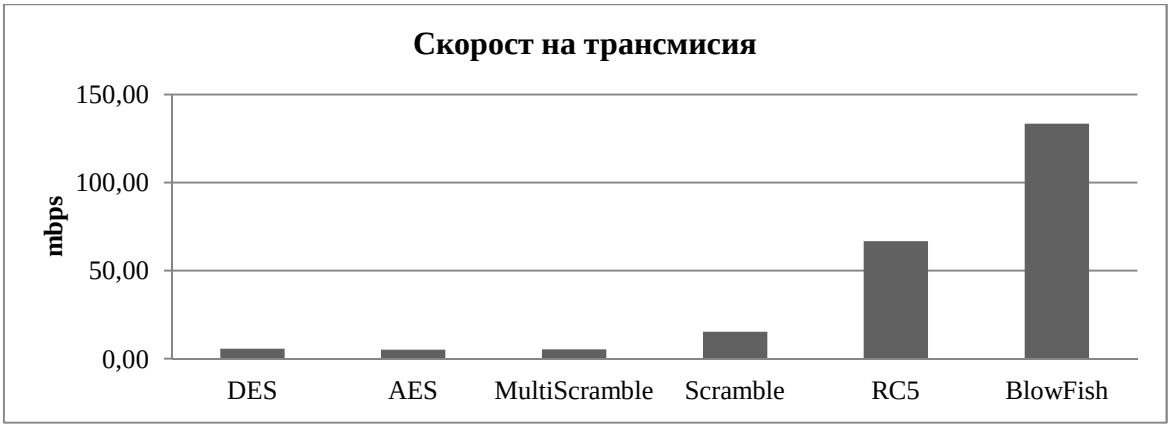
Изследвана е скоростта на трансмисия на предложения шифър при размер на блока данни 128 бита и многостепенния му вариант (означени съответно като Scramble и MultiScramble) в сравнение с най-разпространените блокови шифри в световен мащаб. Резултатите от изследването са представени в Таблица 2 и фигури 4 и 5.

Таблица 2. Сравнение на скоростта на шифрите при обработка на 1 млн. символа

	Време в секунди за 1 млн. символа	**Скорост на трансмисия в [mbps]**
DES	1.41	5.67
AES	1.55	5.16
MultiScramble	1.52	5.23
Scramble	0.51	15.38
RC5	0.12	66.67
BlowFish	0.06	133.33



Фиг. 4. Време за обработката на 1 милион символа



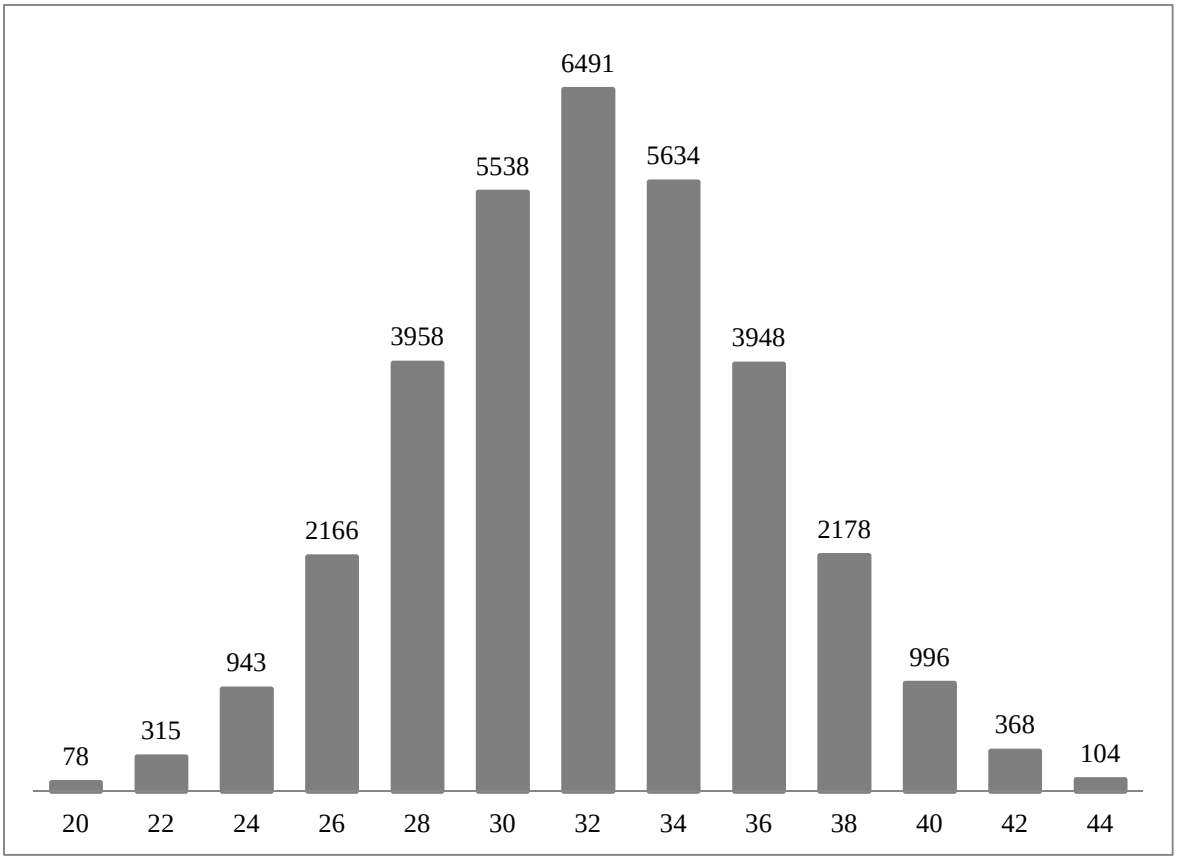
Фиг. 5. Скорост на трансмисия / количество данни, обработени за една секунда / в [mbps]

C) Анализ на броя разлики между криптиран и некриптиран вариант на блок данни

Проследени са разликите между некодираните данни и кодирания им вариант за 32 768 блока при размер на блока 64 бита. Математическото очакване за тези разлики е 32 (половината битове). Резултатите от изследването са представени в Таблица 3 и фигура 6.

Таблица 3. Разделяне на блоковете по брой разлики на кодирането им с оригинала

Брой разлики	Брой блокове	Вероятност
20	78	0.2%
22	315	1%
24	943	2.9%
26	2166	6.6%
28	3958	12.1%
30	5538	16.9%
32	6491	19.9%
34	5634	17.2%
36	3948	12.1%
38	2178	6.7%
40	996	32%
42	368	1.1%
44	104	0.3%
Общо:	32 678	100%



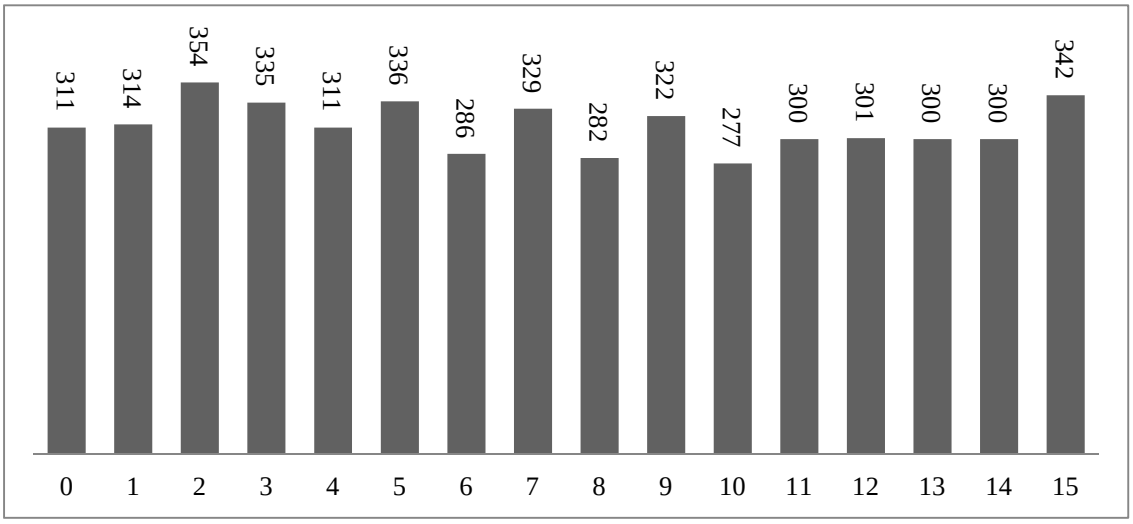
Фиг. 6. Брой блокове с указания брой разлики между кодирани и некодирани данни

D) Изследване на кратността на ключовете

Чрез извадка от 5000 ключа е изследвана случайността при генерирането на ключове. Размерът на отделните групи ключове според кратността е близък, което доказва случайния характер на параметъра (виж Таблица 4 и фигура 7).

Таблица 4. Кратност на ключове

Кратност	Брой	Вероятност
0	311	6.22%
1	314	6.28%
2	354	7.08%
3	335	6.7%
4	311	6.22%
5	336	6.72%
6	286	5.72%
7	329	6.58%
8	282	5.64%
9	322	6.44%
10	277	5.54%
11	300	6%
12	301	6.02%
13	300	6%
14	300	6%
15	342	6.84%
Общо ключове:	5000	100%



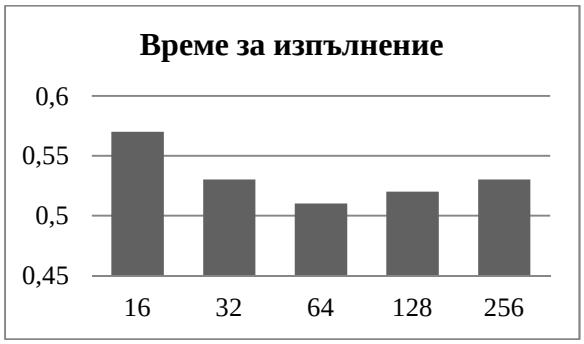
Фиг. 7. Брой ключове според кратността

Е) Изследване на време за кодиране и декодиране

Изследването на времето за кодиране и декодиране на поток от данни с размер 1 милион символа довежда до извода, че скоростта на обработка слабо зависи от избрания размер на блока. Въпреки това, 64 битов блок е най-подходящият размер за данните (виж Таблица 5 и фигура 8).

Таблица 5. Време за изпълнение при вариращ размер на блоковете

Размер на блоковете	Брой на блоковете	Време за изпълнение
16	567708	0.57 сек.
32	284402	0.53 сек.
64	142805	0.51 сек.
128	71463	0.52 сек.
256	36284	0.53 сек.



Фиг. 8. Време за изпълнение при различни размери на данновите блокове

3. Изводи

Предстои задълбочено изследване на криптографския протокол, за което е наложително подлагането му на разнотипни криптоаналитични атаки, които да определят степента му на сигурност. На този етап пт разработките направените изследвания показват ефективността и качествата на предложения алгоритъм. Възможна насока на работа е определянето на оптимален метод за установяване на първоначалния ключ на трансмисията.

Литература

[1]. https://www.cryptochallenge.com/home/importance
[2]. http://en.wikipedia.org/wiki/Block_cipher
[3]. http://en.wikipedia.org/wiki/Cryptographic_protocol

За контакти:
Сиво В. Даскалов
студент в катедра КНТ
ТУ- Варна
E-mail: sivodaskalov@gmail.com;

Научни ръководители:
гл. ас. д-р инж. Милена Карова, e-mail: milena.karova@gmail.com
гл. ас. д-р Златка Матева, e-mail: mateva@tu-varna.bg

ПРИЛОЖЕНИЯ НА OPENCV ЗА РАЗПОЗНАВАНЕ И СЛЕДЕНЕ НА ОБЕКТИ В РЕАЛНО ВРЕМЕ

Красимир Д. Димитров, Сиво В. Даскалов

Резюме: Кратък обзор на Open-Source библиотеката OpenCV и демонстрация на някои от възможностите и за разпознаване и следене на обекти в реално време. За целите на доклада сме представили няколко примера, реализирани на езика C++.
Ключови думи: Компютърно зрение, OpenCV, разпознаване на обекти, следене на обекти

Applications of OpenCV for Object Recognition and Movement Tracking in Real-time Systems

Krasimir D. Dimitrov, Sivo V. Daskalov

Abstract: Review of the open-source library OpenCV and its capabiliries for real-time object recognition and movement tracking. For the purpose of this paper we've gathered some examples of different recognition and tracking methods, written in C++.
Keywords: Computer vision, OpenCV, Object recognition, Object tracking.

1. Въведение

Компютърното зрение [1,2] е термин, с който се означават група изследвания в областта на компютърните науки, насочени към автоматична обработка на изображения от реалния свят, с цел да се извлече и интерпретира визуалната информация в тях. Като научна дисциплина, компютърното зрение разглежда теорията зад изображенията, използвайки информация, получена механично. В областта на технологиите, компютърно зрение обикновено се отнася до процеса на интегриране на автоматизирания анализ на изображенията с други методи и технологии, например при проектиране на роботи и други индустриални приложения.

За целите на доклада, като система за компютърно зрение, ще използваме Open-Source библиотеката OpenCV. OpenCV е библиотека, състояща се от различни функции, позволяващи работа с графични данни. Тя съдържа съвкупност от инструменти, позволяващи обработка на изображения (филтриране на шум, изрязване и др. графични операции), работа с данни от камера в реално време, представяне на изображения като матрици и др.

2. Метод за разпознаване по цвят на обекта

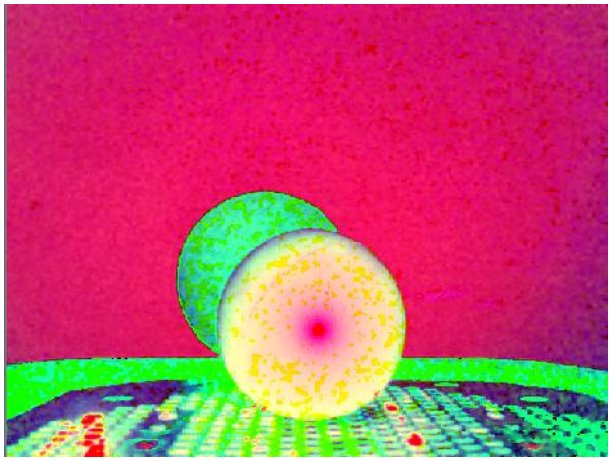
Този метод е удобен, ако обектът, който търсим на изображението, е едноцветен и отличаващ се от фона. Методът [3] може да бъде разделен на няколко основни стъпки:

1) Конвертиране на изображението от RGB (фигура 1) в HSV (фигура 2). Това се извършва, защото HSV отделя интензивността на цвета от самия цвят. По този начин цветът може да бъде установен с точност. В RGB цветовия модел е възможно цветът да се сгреши с друг, поради наличието на светлосенки и др. Например, при знаците „Стоп!“ основният цвят от RGB модела не винаги е Red компонентата.

В OpenCV това се извършва чрез функцията cvtColor с дефиниция:

void cvtColor(InputArray src, OutputArray dst, int code, int dstCn=0)

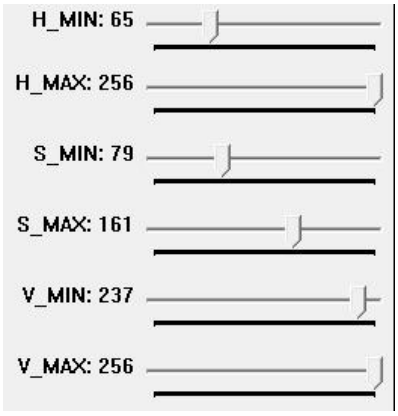
Фиг. 1. Оригинално изображение в RGB схема



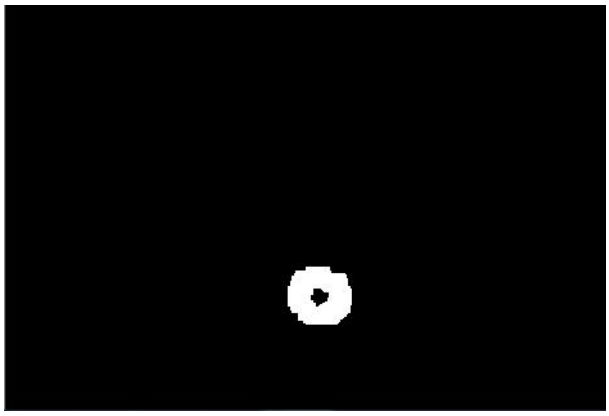
Фиг. 2. Изображението в HSV цветова схема

2) Филтриране на търсения цвят, използвайки допустими граници на компонентите H, S и V (фигура 3). В OpenCV това се извършва чрез функцията inRange с дефиниция:

void inRange(InputArray src, InputArray lowerb, InputArray upperb, OutputArray dst)



Фиг. 3. Параметри за филтриране



Фиг. 4. Резултантно бинарно изображение

3) Резултатът от стъпки 1) и 2) е бинарно изображение на обекта и остатъчен шум. За изчистването му се използват морфологичните операции Ерозия и Дилатация в горепосочения ред. След прилагането на първия оператор шумът е премахнат, но части от търсения обект могат да бъдат загубени. Дилатацията възвръща премахнатите части от обекта и възстановява неговата цялост. След обработката на матрицата остава бинарно изображение на обекта (фигура 4) при подбиране на подходящи параметри на използваните функции в OpenCV, а именно:

erode(src, erosion_dst, element);
dilate(src, dilation_dst, element);

4) Последната стъпка е намирането на контура на търсения обект. За целта използваме функцията FindContours на OpenCV, която връща вектор с всички намерени контури в изображението. С помощта на „Moments" метода намираме най-големия контур от вектора чрез неговата вътрешна част. Това позволява успешно разпознаване дори и при наличие на остатъчен шум от предходната стъпка.

void findContours(InputOutputArray image, OutputArrayOfArrays contours, int mode, int method, Point offset=Point())

5) След изпълнение на алгоритъма е получен контурът на търсения обект, който може да бъде подложен на допълнителна обработка според изискванията на задачата.

3. Метод за следене на обект чрез изследване на последователни изображения

Този метод [3] използва две последователни изображения (фигура 5) от камерата и открива несъвпадащите пиксели между тях. При преместване на обект фонът ще бъде запазен, а разлика между двата кадъра ще има само в предходната и настоящата позиция на обекта.

Фиг. 5. Два съседни кадъра от видеоматериала

За намиране на разликата между двата кадъра чрез OpenCV използваме функцията absdiff с дефиниция:

void absdiff(InputArray src1, InputArray src2, OutputArray dst);

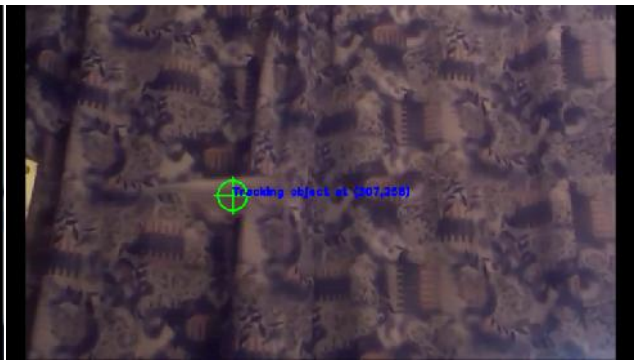
Тази функция връща изображение в сивата гама (фигура 6), което следва да се бинаризира чрез функцията threshold с дефиниция:

double threshold(InputArray src, OutputArray dst, double thresh, double maxval, int type)

Фиг. 6. Образ на разликите между съседните изображения

След като имаме бинарно изображение използваме замъгляваща функцията, за да премахнем шумовете и да слеем разликите между двете изображения. След прилагане на blur функцията отново използваме threshold, за да получим крайното бинарно изображение, илюстриращо движението във видеоматериала (фигура 7). За получаването на контура на движението можем да използваме стъпка 4) от предходния метод (фигура 8).

Фиг. 7. Бинаризирано движение



Фиг. 8. Резултат от проследяването

4. Изводи

Предимства на метода за разпознаване по цвят:

- Методът позволява използването на камера с ниско качество.
- Методът е лесен за използване и имплементиране в различни програми.
- Възможно е разпознаването по снимка.

Недостатъци на метода за разпознаване по цвят:

- Следеният обект трябва да е едноцветен или поне всички цветове върху него да са нюанси на един и същ цвят.
- При голяма скорост обектът се замазва и не винаги може да се открие.
- Фонът не трябва да съдържа от цвета на търсения обект.

Предимства на метода за следене на обект чрез изследване на последователни изображения:

- Методът следи добре обекти, движещи се с голяма скорост (при условие че и кадрите се снемат достатъчно бързо).
- Цветът на обекта и на фона няма значение при използването на този метод.

Недостатъци на метода за следене на обект чрез изследване на последователни изображения:

- Изисква по-добра камера.
- Кадрите трябва да бъдат заснети по-бързо (за да няма голямо изменение в местоположението на обекта).
- Ако все пак има голямо изменение в местоположението на обекта, между двата поредни кадъра е възможно да се получат неточни данни за текущото местоположение на обекта, дължащи се на апроксимацията в намирането му.
- Нужно е фонът на заснемане да е неподвижен.

Литература

[1]. http://docs.opencv.org/
[2]. http://en.wikipedia.org/wiki/Computer_vision
[3]. https://www.youtube.com/channel/UCJ2b0kP6Hwc_R8ebv8P2f9w

За контакти:
Красимир Д. Димитров
E-mail: kr.dimitrov93@gmail.com
Сиво В. Даскалов
E-mail: sivodaskalov@gmail.com
Студенти в Катедра КНТ на ТУ- Варна

СИСТЕМА ЗА ВИЗУАЛНО КОНФИГУРИРАНЕ НА ПЕРИФЕРИЯТА НА МИКРОКОНТРОЛЕРИ НА ATMEL

Десислав В. Мичев, Трифон И. Русков

Резюме: В статията се разглеждат особеностите и изискванията към мултиплатформени системи за конфигуриране предназначението и функционалността на изводите на едночиповите микроконтролери, използвани в различни цифрови устройства. Представена е мултиплатформена система за конфигуриране на периферия за контролери Atmel, чиято цел е ускоряване на разработката на проекти на базата на микроконтролерите Atmel, генериране на оптимален по обем и бързодействие на изпълнение код за управление на периферните изводи, както и използване за практическо обучение на студенти.
Ключови думи: периферия, микроконтролери, atmel

System for Visual Peripheral Configuration of Atmel Microcontrollers

Desislav V. Michev, Trifon I. Ruskov

Abstract: In this article we discuss features and requirements for portable systems which allow easy setting the functionality of the single chip microcontroller pins used in various digital devices. We introduced a portable system for Atmel microcontrollers peripheral configuration. The system gives an opportunity for generating an optimal in size and performance code for peripheral control. It may be used also as teaching tool for students in subjects on microcontrollers and microcontrollers based design.
Keywords: peripheral, microcontrollers, atmel

1. Увод

Съвременните електронни устройства в управлението и бита се изграждат на базата на микроконтролери. Повсеместното им използване изисква включване на възможности за подключване и управление на разнообразни и често уникални външни компоненти като датчици, преобразователи на информацията, електромеханични елементи и др. Това изисква специфични познания и умения от проектантите и разработчиците на електронна апаратура на базата на контролери. За облекчаване на техния труд и съкращаване на сроковете за разработка и внедряване на нови изделия, което е от съществено значение при съвременното динамично развитие на електронните технологии, производителите на микроконтролери предоставят инструментални средства за автоматизация на проектирането. Тези средства основно се базират на мултиплатформени библиотеки с възможност за използване в средата на различни операционни системи. Най-известните среди за конфигуриране на микроконтролери в настоящия момент са: MPLAB Code configurator [2] за контролери на фирмата Microchip, STM32Cube [3] за контролери на фирмата ST и др. Същевременно за контролерите на една от водещите фирми, производителки на микроконтролери, Atmel [1] не е известна такава мултиплатформена инструментална среда. Целта на настоящата работа е разработка на мултиплатформена система за конфигуриране на периферия за контролери Atmel, която значително да ускори разработката на проекти на базата на тези микроконтролери, както и да се използва за практическо обучение на студенти.

2. Изисквания към мултиплаформената система за конфигуриране на периферия

За да се отговори максимално на нуждите на разработчиците на електронно оборудване на базата на микроконтролерите Atmel, разработваната система трябва да притежава като минимум следните функционални възможности:

- Графично представяне на избрания микроконтролер. Фирма Atmel предлага богата гама микриконтролери с различен брой, разположение и предназначение на изводите. Графичното представяне спомага потребителите лесно да се ориентират при избор на най-подходящите изводи за нужното приложение, като също така му предлага мащабируемост на изображението и подходящо оцветяване, според избраното предназначение на изводите. Допълнително удобство може да се постигне чрез добавяне на интерактивност, чрез която да се избира коя от наличните мултиплексирани функционалности да се използва за дадения извод.
- Интерфейс за конфигуриране на периферия. Потребителският интерфейс трябва да бъде гъвкав и интуитивен и да подпомага максимално проектанта при вземане на най-подходящото решение за конфигуриране на нужната периферия.
- Генериране на оптимален по бързодействие и заеман обем памет изходен програмен код. За постигане на това изискване най-подходящ е изходен код на език C, без допълнителни библиотеки.
- Мултиплатформеност. Системата трябва да бъде лесно преносима и да работи еднакво добре в средата на различни операционни системи – Windows, Linux, MAC OS и др.
- Разширяемост. Лесното добавяне на нови микроконтролери от същата серия без необходимост от прекомпилиране.

За постигане на нужната мултиплатформеност, малък размер и високо бързодействие системата е разработена на език C++ с използване на мултиплатформените библиотеки wxWidgets [4] и SQLite3 [5].

3. Основна структура на системата

Следвайки поставените изисквания, разработената инструментална система CWAM (Configuration Wizard for Atmel Microcontrollers) има структура (фигура 1), състояща се от следните компоненти:

- Прозорец за създаване на нов проект, който предоставя на клиента възможност да избере нужните данни за създаване на нов проект. Поради големия наличен брой микроконтролери, произвеждани от фирма Atmel, към прозореца за създаване на нов проект са добавени и два филтъра, даващи възможност за филтрация на наличната периферия и наличните за системата контролери. Предоставя се също така възможност за филтриране на микроконтролери по наличната в тях периферия.
- Панел за визуализация на контролер. Панелът фигурира винаги в централната горна част на системата и представя визуализация на корпуса на целевия контролер. Използва се само в проект. Визуализира корпуса на избрания микороконтролер, както и неговите изводи, като ги оцветява по подходящ начин. Позволява движение и скалиране на изображението с цел по–голямо удобство при работа.
- Панел за визуализация на модули. Този панел представлява табуларен графичен контрол, който позволява към него да бъдат добавяни много на брой прозорци. В

текущата реализация контролата позволява показване само на един прозорец - модул.

Фиг. 1. Структура на системата

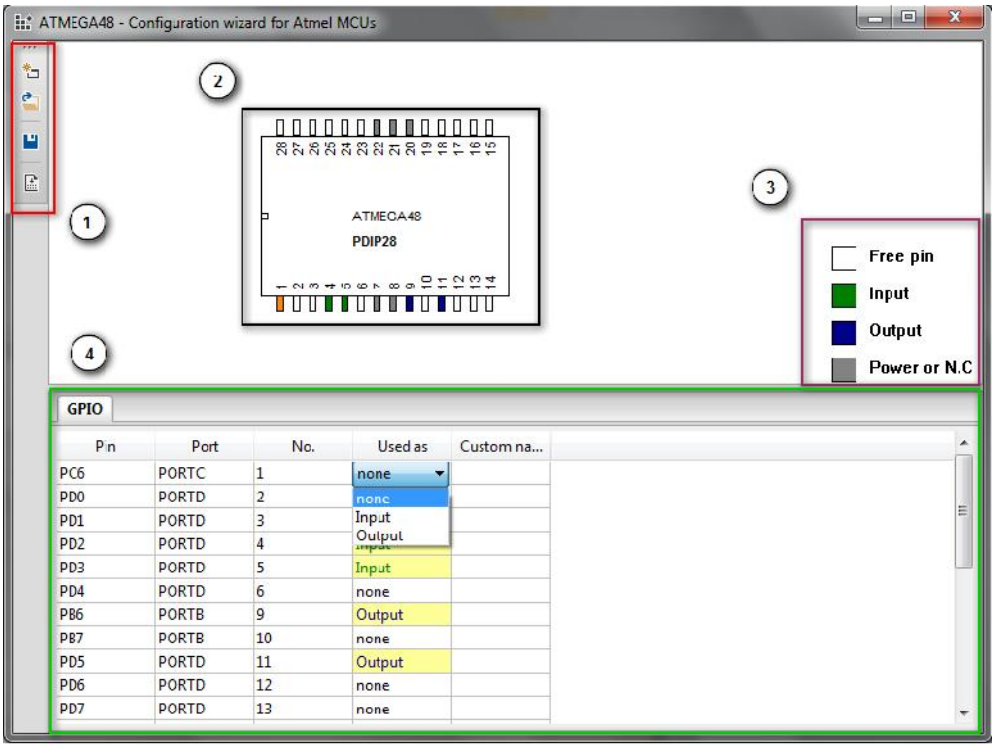
- Проект. Чрез този блок се реализира функционалността за менажиране на проект, която включва: създаване на нов проект, отваряне на съществуващ проект, запис на проект, генериране на изходен код, създаване и инициализиране на модулите на целевия контролер, генератор на корпуси. Генерирането на графичното изображение корпус на конкретен микроконтролер се извършва автоматично по наименованието на корпуса.
- Модули. Модулите предоставят графичен интерфейс за генериране на конкретна конфигурация на изводите на микроконтролера и/или на един или повече регистри от него. Всяка периферна конфигурация се генерира от отделен модул. При това всеки един модул генерира собствен изходен код и го предава към създадения проект, който съответно го записва във файлове.
- Източник на данни. Източникът на данни има за цел да осигури на системата набор от микроконтролери, модули и зависимости, които ще бъдат използвани при създаване на нов проект. Физически данните се съхраняват в SQLite 3 база от данни и използват към момента седем таблици. Описанието на микроконтролерите включва: име, корпус, изводи и тяхното именоване към всеки корпус на микроконтролер, периферия и нейните изводи към корпус на микроконтролер. Всичко това дава възможност на системата да бъде разширена чрез наличния огромен инструментариум, предоставен за SQLite, без системата да има нужда от прекомпилиране при промяна на данните.

3. Работа с програмна система CWAM

Работният екран на системата CWAM е показан на фигура 2. Той се стартира чрез избор на нов или съществуващ проект и предоставя интуитивност и удобство за работа. Основните елементи на основния работен екран са:

1. Лента с инструменти - предоставя възможност за създаване на нов, отваряне на съществуващ и запис на текущ проект. Тя предоставя също така и бутон, чрез който се генерира изходен C код, базиран на текущата конфигурация.
2. Изглед на избрания микроконтролер – изобразява корпуса и изводите на избрания микроконтролер, като оцветява по подходящ начин всеки извод.
3. Легенда с цветове, указваща текущото състояние на даден извод.
4. Списък с наличните за избрания микроконтролер модули за конфигуриране на периферия

За да се конфигурира дадена периферия, трябва да се избере съответният модул от поле 4 на работния екран, при което се визуализира форма за задаване на възможните параметри. След като се зададат всички нужни параметри в избраните модули се генерира C изходен код чрез натискане на бутон „Generate source code” от лентата с инструменти.



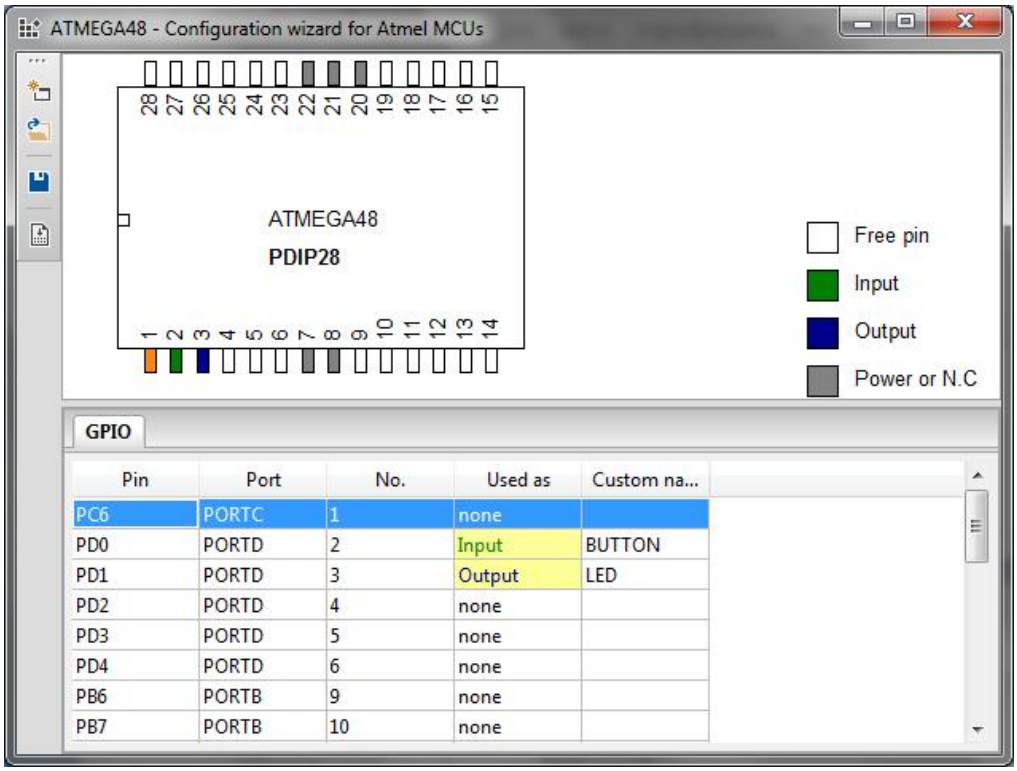
Фиг. 2. Работен екран

Като краен продукт на системата се получава „C” програмен код, който може да бъде вложен в рзработвана система на базата на съответния микроконтролер на фирмата Atmel. Чрез него периферията на целевия микроконтролер се конфигурира според настройките, които са били зададени чрез графичният интерфейс на системата. Изходният код не използва библиотеки и има една единствена зависимост – хедър файловете за микроконтролерите, предоставени от Atmel. Избягването на библиотечни функции има за цел да предостави на студенти и разработчици допълнителна възможност за по–детайлно усвояване на архитектурата на микроконтролерите и съответно за генериране на по–малък по обем изпълним код (firmware).

Пример за конфигуриране на периферия с програмна система CWAM:

- Установяване на извод PD1 като изход;
- Именоване на извод PD1 като „LED”;
- Установяване на извод PD0 като вход;
- Именоване на извод PD0 като „BUTTON”.

Графичното представяне на тази конфигурация може да се види на фигура 3.



Фиг. 3. Примерна конфигурация

В резултат на представената конфигурация, системата генерира следните два С файла – periphrals.h и peripherals.c

Файл peripherals.h

```
#ifndef _PERIPHERALS_H_
#define _PERIPHERALS_H_
#include <stdint.h>
#include <avr/io.h>
#define BUTTON_PORT PORTB
#define BUTTON_PIN PINB
#define LED PD1
#define LED_DDR DDRB
#define LED_PORT PORTB
#define LED_PIN PINB
#define BUTTON_DDR DDRB
#define BUTTON PD0
void Init_periperals();

#endif
```

Файл peripherals.c

```
#include <stdint.h>
#include <avr/io.h>
#include "peripherals.h"

void Init_peripherals()
{

	/* Set pin direction for PORTD */
	/* Inputs:  PD0 */
	/* Outputs:  PD1 */
	DDRD = (1<<PD1);
}
```

Разработчикът на микроконтролерната система може да използва тези файлове като ги вложи в свой проект на език С и извика функция **Init_pripherals** в началото на своя програмен код. Това обаче трябва да бъде преди основния цикъл.

Основната функционалност на системата приключва с инициализация на избраната периферия. На базата на тази конфигурация, разработчикът може да проектира детайлно алгоритъма на работа на целевата микроконтролерна система. Работата на разработчика допълнително се улеснява от именуването използваните изводи от модул “GPIO”, които е генерирала системата. В конкретния пример това са LED и BUTTON. За тях системата е генерирала следните четири дефиниции за всяко именуване:

- LED_PORT – използвания от LED порт;

- LED_PIN – използвания от LED порт, който е валиден само при операция **четене;**
- LED_DDR – използваният от LED регистър за определяне на посоката на даден порт;
- LED – който представлява номер от озползвания порт. За BUTTON дефинициите са аналогични

4. Заключение

Разработената система CWAM удовлетворява поставените изисквания и предоставя удобен инструмент както за разработка на микроконтролерни системи, така и за обучение на студенти и разботчици. Като перспективи за бъдещо развитие се планира добавяне на допълнителни модули за конфигуриране на периферия. За улеснение на потребителите е необходимо да бъде добавена функционалност, даваща възможност за интерактивно конфигуриране на периферия през визуалното представяне на целевия контролер.

Литература

[1].http://www.atmel.com/
[2].http://www.microchip.com/pagehandler/en_us/devtools/code_configurator/home.html
[3].http://www.st.com/stm32cube-pr2
[4].http://wxwidgets.org/
[5].http://www.sqlite.org/

За контакти:
Десислав В. Мичев
Катедра „Компютърни науки и технологии”
Технически университет - Варна
E-mail: d.michev@gmail.com

доц. д-р инж. Трифон И. Русков
Катедра „Компютърни науки и технологии”
Технически университет - Варна
E-mail: ruskov@tu-varna.bg

ИЗИСКВАНИЯ ЗА ОФОРМЯНЕ НА СТАТИИТЕ ЗА СПИСАНИЕ "КОМПЮТЪРНИ НАУКИ И ТЕХНОЛОГИИ”

I. Статиите се представят разпечатани в два екземпляра (оригинал и копие) в размер до 6 страници, формат А4 на адрес: Технически университет – Варна, ФИТА, ул. „Студентска” 1, 9010 Варна, както и в електронен вид на имейл адреси: peter.antonov@ieee.bg или jppet@abv.bg.

II. Текстът на статията трябва да включва: УВОД (поставяне на задачата), ИЗЛОЖЕНИЕ (изпълнение на задачата), ЗАКЛЮЧЕНИЕ (получени резултати), БЛАГОДАРНОСТИ към сътрудниците, които не са съавтори на ръкописа (ако има такива), ЛИТЕРАТУРА и информация за контакти, включваща: научно звание и степен, име, организация, поделение (катедра), e-mail адрес.

III. Всички математически формули трябва да са написани ясно и четливо (препоръчва се използуване на Microsoft Equation).

IV. Текстът трябва да бъде въведен във файл във формат WinWord 2000/2003 с шрифт Times New Roman. Форматирането трябва да бъде както следва:

1. Размер на листа - А4, полета: ляво - 20мм, дясно - 20мм, горно - 15мм, долно - 35мм, Header 12.5мм, Footer 12.5мм (1.25см).
2. Заглавие на български език - размер на шрифта 16, удебелен, главни букви.
3. Един празен ред - размер на шрифта 14, нормален.
4. Имена на авторите - име, инициали на презиме, фамилия, без звания и научни степени - размер на шрифта 14, нормален.
5. Два празни реда - размер на шрифта 14, нормален.
6. Резюме и ключови думи на български език, до 8 реда - размер на шрифта 11, нормален.
7. Заглавие на английски език - размер на шрифта 12, удебелен.
8. Един празен ред - размер на шрифта 11, нормален.
9. Имена на авторите на английски език - размер на шрифта 11, нормален.
10. Един празен ред - размер на шрифта 11, нормален.
11. Резюме и ключови думи на английски език, до 8 реда - размер на шрифта 11, нормален.
12. Основните раздели на статията (Увод, Изложение, Заключение, Благодарности, Литература) се форматират в едноколонен текст както следва:
    a. Наименование на раздел или на подраздел - размер на шрифта 12, удебелен, центриран, един празен ред преди наименованието и един празен ред след него - размер на шрифта 12, нормален;
    b. Текст - размер на шрифта 12, нормален, отстъп на първи ред на параграф – 10 мм; разстояние от параграф до съседните (Before и After) за целия текст – 0.
    c. Цитиране на литературен източник - номер на източника от списъка в квадратни скоби;
    d. Текстът на формулите се позиционира в средата на реда. Номерация на формулите - дясно подравнена, в кръгли скоби.
    e. Фигури - центрирани, разположение спрямо текста: “Layout: In line with text”. Номер и наименование на фигурата - размер на шрифта 11, нормален, центриран. Отстояние от съседните параграфи – 6 pt.
    f. Литература – всеки литературен източник се представя с: номер в квадратни скоби и точка, списък на авторите (първият автор започва с фамилия, останалите – с име), заглавие, издателство, град, година на издаване, страници.
    g. За контакти: научно звание и степен, име, презиме (инициали), фамилия, организация, поделение (катедра), e-mail адрес, с шрифт 11, дясно подравнено.

Образец за форматиране можете да изтеглите от адрес http://cs.tu-varna.bg/ - Списание КНТ, Spisanie_Obrazec.zip.

Този брой от списанието се отпечатва със средства от научно-изследователски проект, финансиран целево от държавния бюджет.