SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babeş Bolyai University**
1.2 Faculty	**Faculty of Mathematics and Computer Science**
1.3 Department	**Department of Computer Science**
1.4 Field of study	**Computer Science**
1.5 Study cycle	**Bachelor**
1.6 Study programme / Qualification	**Computer Science (in english)**

2. Information regarding the discipline

2.1 Name of the discipline				**Computer Systems Architecture**			
2.2 Course coordinator				**Lect. Dr. Vancea Alexandru-Ioan**			
2.3 Seminar coordinator				**Lect. Dr. Vancea Alexandru-Ioan**			
2.4. Year of study	**1**	2.5 Semester	**1**	2.6. Type of evaluation	**E**	2.7 Type of discipline	**Compulsory**

3. Total estimated time ((hours/semester of didactic activities)

3.1 Hours per week	5	Of which: 3.2 course	2	3.3 seminar/laboratory	1 sem + 2 lab
3.4 Total hours in the curriculum	70	Of which: 3.5 course	28	3.6 seminar/laboratory	42
Time allotment:			hours		
Learning using manual, course support, bibliography, course notes			20		
Additional documentation (in libraries, on electronic platforms, field documentation)			10		
Preparation for seminars/labs, homework, papers, portfolios and essays			20		
Tutorship			10		
Evaluations			20		
Other activities:					
3.7 Total individual study hours			80		
3.8 Total hours per semester			150		
3.9 Number of ECTS credits			6		

4. Prerequisites (if necessary)	
4.1. curriculum	
4.2. competencies	

5. Conditions (if necessary)

5.1. for the course	□ projector
5.2. for the seminar /lab	□ Laboratory with computers

6. Specific competencies acquired

6.1 Professional competencies	C6.1 Identification of basic concepts and models for computer systems and computer networks. C6.2 Identification and description of the basic architectures for the organization and management of systems and networks.
6.2 Transversal competencies	CT1 Application of organized and efficient work rules, of responsible attitudes towards the didactic and scientific domain, for the creative exploitation of their own potential according to the principles and rules of professional ethics CT3 Use of effective methods and techniques of learning, information, research and development of the capacity to exploit knowledge, to adapt to the requirements of a dynamic society and communication in Romanian language and in a foreign language

7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	Knowledge of the computer architecture models, processor functioning, computer information representation usage
7.2 Specific objective of the discipline	Understanding by the students of the computer architecture models, processor functioning, computer information representation usage Initiation in assembler language programming, which will assure the comprehension of the microprocessor architecture and functioning Understanding the basic functions of a computer's architectural components and its native low-level workflow. Awareness of the architectural impact on designing and implementing high level programming languages. Understanding the impact of the 80x86 processor architecture on Windows functioning and limitations. Awareness of the triade computer architecture – operating systems – programming languages and their interactions as the basic core of Computer Science.

8. Contents

8.1 Course		Teaching methods	Remarks
1	Data representation: elementary data, binary representation and placement orders, data organizing and storing	Exposure, description, explanation, examples, discussion of case studies	
2	Character coding, signed and unsigned representation, complementary code, conversions, the concept of mathematical overflow.		
3	Computing systems architecture: organization of a CS, the central processing unit, the system clock, computer on n bits, the storage, peripheral devices.		
4	CS performances, the 80x86 microprocessor's architecture – general view of its structure. The address computation mechanism, addressing modes, far addresses and near addresses.		
5	The Executive Unit (EU) of the 80x86 microprocessor: role and		

	functions of the general EU registers and the flags. Classifications (Registers and Flags) and case studies.		
6	The Bus Interface Unit (BIU) of the 80x86 microprocessor: the address registers, segment registers, machine instructions representation. The offset specification formula on 32 bits vs. on 16 bits.		
7	Assembly language elements: the source line format, expressions, accessing the operands, operators. Temporary non-distructive conversions (and specific operators).		
8	Directives for defining segments, data definition directives, directives EQU and INCLUDE, macros.		
9	Assembly language instructions: transfer instructions, signed and unsigned distructive conversions, signed and unsigned arithmetic operations, bitwise shifting and rotating, logical bitwise operations.		
10	Conditional and unconditional jump instructions, looping instructions, string instructions. Overflow analysis: how the 80x86 architecture reacts to it.		
11	Subprograms call implementation and multimodule programming: CDECL and STDCALL calling conventions, call code, entry code, exit code, the import-export directives EXTRN and GLOBAL.		
12	Linking NASM modules with modules written in high-level programming languages (case study – C programming language). Recursive call extensive example discussion.		
13	Windows static and dynamic libraries: LIB vs. DLL. NASM output object file formats and their library support. Win32 system libraries: file management examples, process management, memory management. Implementing user libraries.		
14	Real address mode vs.protected mode code execution environment. The interaction between user programs and the OS kernel. The virtual memory concept. Overview of the segmentation and paging process. Protection setup sample code: real-mode to protected-mode transition and 32-bit segments.		

Bibliography

1. Al. Vancea, F. Boian, D. Bufnea, A. Andreica, A. Darabant, A. Navroschi – Arhitectura calculatoarelor. Limbajul de asamblare 80x86., Editura Risoprint, Cluj-Napoca, 2014.
2. Al. Vancea, F. Boian, D. Bufnea, A. Gog, A. Darabant, A. Sabau – Arhitectura calculatoarelor. Limbajul de asamblare 80x86., Editura Risoprint, Cluj-Napoca, 2005.
3. A. Gog, A. Sabau, D. Bufnea, A. Sterca, A. Darabant, Al. Vancea – Programarea în limbaj de asamblare 80x86. Exemple si aplicatii., Editura Risoprint, Cluj-Napoca, 2005.
4. Randal Hyde – The Art of Assembly Programming, No Starch Press, 2003.
(http://homepage.mac.com/randyhyde/webster.cs.ucr.edu/www.artofasm.com/DOS/index.html)
5. Boian F.M. Vancea A. Arhitectura calculatoarelor, suport de curs. Facultatea de Matematica si Informatica, Centrul de Formare Continua si Invatamânt la Distanta,. Ed. Centrului de Formare Continua si Invatamânt la Distanta, Cluj, 2002
6. Irvine, K.R., 2015. *Assembly language for x86 processors*.

7. Kusswurm, D., 2014. *Modern X86 Assembly Language Programming*. Springer.
8. Carter, P.A., 2004. *PC Assembly Language*. Github: (http://pacman128.github.io/static/pcasm-book.pdf)
9. Cavanagh, J., 2013. *X86 Assembly Language and C Fundamentals*. CRC Press.
10. Guide, P., 2011. Intel® 64 and ia-32 architectures software developer's manual. *Volume 3B: System programming Guide, Part, 2*, p.11. (http://www.facweb.iitkgp.ac.in/~goutam/compiler/readingMaterial/intelXeon/253665.pdf)
11. BitDefender internal documentations – posted slides.

8.2 Seminar/Laboratory	Teaching methods	Remarks
Seminars: S1: Introduction to the IA-32 assembly language. Converting numbers between number bases 2, 10, 16. Representation of integer numbers in the computer's memory. Signed and unsigned instructions. S2: Signed and unsigned instructions. Arithmetic instructions (multiplications and divisions). Signed and unsigned conversions. S3: Little-endian representation of numbers in the memory. Conditional and unconditional jumps. String operations. S4: String instructions. Complex string problems. S5: Library functions call (printf, scanf, fread, fscanf, fprintf, fclose). S6: Multi-module programming using the assembly language. S7: Topics review and exam preparation by case studies. **Laboratories** L1: Converting between different nummeration bases. Bit. Sign bit. Complementary code. Representing signed integers. Tools for laboratories. Assembly language program structure. L2: Simple arithmetic expressions based on additions, substractions, multiplications and divisions. L3: Complex arithmetic expressions (little-endian, signed and unsigned conversions, declaring variables / constants).	Exposure, description, explanation, examples, discussion of case studies Practical projects	

L4: Bitwise instructions (Bitwise logical operations, Shift and rotate operations). L5: Simple String operations (Instructions for comparisons, conditional jumps and repetitive loops). L6: Complex String operations (Specific assembly language instructions for working on strings of bytes/words/doublewords/quadwords). L7: Function calls: Function libraries, Using external functions. Call conventions, Calling a system function. Standard msvcrt functions. L8: Moodle midterm test L9: Text file operations (open, write, read, close). L10: Plenary discussions, analysis and evaluation of up-to-date work. Catch up time with the last given homeworks. Concluding the assembly language stand alone part. L11: Multi-module programming (asm+asm) L12: Multi-module programming (asm+C) L13: Topics review and practical exam preparation by case studies L14: Practical exam		

Bibliography

1. Al. Vancea, F. Boian, D. Bufnea, A. Andreica, A. Darabant, A. Navroschi – Arhitectura calculatoarelor. Limbajul de asamblare 80x86., Editura Risoprint, Cluj-Napoca, 2014.
2. Al. Vancea, F. Boian, D. Bufnea, A. Gog, A. Darabant, A. Sabau – Arhitectura calculatoarelor. Limbajul de asamblare 80x86., Editura Risoprint, Cluj-Napoca, 2005.
3. A. Gog, A. Sabau, D. Bufnea, A. Sterca, A. Darabant, Al. Vancea – Programarea în limbaj de asamblare 80x86. Exemple si aplicatii., Editura Risoprint, Cluj-Napoca, 2005.
4. Randal Hyde – The Art of Assembly Programming, No Starch Press, 2003. (http://homepage.mac.com/randyhyde/webster.cs.ucr.edu/www.artofasm.com/DOS/index.html)
5. Boian F.M. Vancea A. Arhitectura calculatoarelor, suport de curs. Facultatea de Matematica si Informatica, Centrul de Formare Continua si Invatamânt la Distanta,. Ed. Centrului de Formare Continua si Invatamânt la Distanta, Cluj, 2002
6. Irvine, K.R., 2015. *Assembly language for x86 processors*.
7. Kusswurm, D., 2014. *Modern X86 Assembly Language Programming*. Springer.
8. Carter, P.A., 2004. *PC Assembly Language*. Github: (http://pacman128.github.io/static/pcasm-

book.pdf)
9. Cavanagh, J., 2013. *X86 Assembly Language and C Fundamentals*. CRC Press.
10. Guide, P., 2011. Intel® 64 and ia-32 architectures software developer's manual. *Volume 3B: System programming Guide, Part, 2*, p.11. (http://www.facweb.iitkgp.ac.in/~goutam/compiler/readingMaterial/intelXeon/253665.pdf)
11. BitDefender internal documentations – posted slides.
12. Computer system architecture course homepage – posted support materials and homeworks for lab preparation.

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

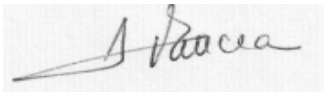
The course exists in the studying program of all major universities in Romania and abroad;
 The content of the course is considered by the software companies as important for average programming skills

10. Evaluation

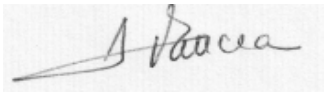
Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	Testing the basic principles of the domain and their interactions	Written exam	55 %
	Verifying the understanding of the assembly language basic operations and mechanisms	Moodle midterm multiple choice test	15 %
	Application of the 32 bits assembly language principles for problem solving;	Average grade received for the laboratory work	15 %
10.5 Lab/Seminar activities	Developing and implementing an assembly language code solution for a given problem	Practical exam	15 %
10.6 Minimum performance standards	At least grade 5 at each of the evaluation methods.		

Data completării 14.04.2020

Titular de curs
Lect. Dr. Alexandru VANCEA



Titular de seminar
Lect. Dr. Alexandru VANCEA



Data avizării în departament
_____

Director de departament
Prof. Dr. Anca ANDREICA