FIȘA DISCIPLINEI

Arhitectura sistemelor de calcul / Computer systems architecture

Anul universitar 2025-2026

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea Babeș-Bolyai
1.2. Facultatea	Matematică și Informatică
1.3. Departamentul	Informatică
1.4. Domeniul de studii	Informatică
1.5. Ciclul de studii	Licența
1.6. Programul de studii / Calificarea	Informatică
1.7. Forma de învățământ	Cu frecvență

2. Date despre disciplină

2.1. Denumirea disciplinei	**Arhitectura sistemelor de calcul / Computer systems architecture**					Codul disciplinei	**MLRE004**
2.2. Titularul activităților de curs	Conf. univ. dr. ing. Mihai SUCIU						
2.3. Titularul activităților de seminar	Conf. univ. dr. ing. Mihai SUCIU						
2.4. Anul de studiu	1	2.5. Semestrul	1	2.6. Tipul de evaluare	E	2.7. Regimul disciplinei	Obligatoriu

3. Timpul total estimat (ore pe semestru al activităților didactice)

3.1. Număr de ore pe săptămână	**5**	din care: 3.2. curs	**2**	3.3. seminar/ laborator/proiect	**1/2 /0**
3.4. Total ore din planul de învățământ	70	din care: 3.5. curs	28	3.6 seminar/laborator/proiect	42
Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)					**ore**
Studiul după manual, suport de curs, bibliografie și notițe (AI)					20
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					10
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					20
Tutoriat (consiliere profesională)					10
Examinări					20
Alte activități					
3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)				**80**	
3.8. Total ore pe semestru				**150**	
3.9. Numărul de credite				**6**	

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	-
4.2. de competențe	-

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	sala de curs dotata cu tablă și videoproiector
5.2. de desfășurare a seminarului/ laboratorului	laborator cu calculatoare

6.1. Competențele specifice acumulate[1]

Competențe profesionale/esențiale	• dezvoltarea și întreținerea aplicațiilor informatice • utilizarea instrumentelor informatice în context interdisciplinar
Competențe transversale	• aplicarea regulilor de muncă organizată și eficientă, a unor atitudini responsabile față de domeniul didactic-științific, pentru valorificarea creativă a propriului potențial, cu respectarea principiilor și a normelor de etică profesională • desfășurarea eficientă a activităților organizate într-un grup interdisciplinar și dezvoltarea capacităților empatice de comunicare interpersonală, de relaționare și colaborare cu grupuri diverse

6.2. Rezultatele învățării

Cunoștințe	Studentul are cunoștințe necesare pentru utilizarea calculatoarelor, dezvoltarea programelor și aplicațiilor software, procesarea informațiilor. Absolventul are cunoștințe legate de programare, matematică, inginerie și tehnologie si are abilitățile necesare pentru a le folosi in crearea de sisteme informatice complexe.
Aptitudini	Studentul este capabil să aplice reguli generale unor probleme specifice și de a produce soluții relevante. Absolventul are abilitatea de a dezvolta, proiecta și crea noi aplicații, sisteme sau produse folosind bunele practici din domeniu.
Responsabilități și autonomie	Studentul are capacitatea de a lucra independent pentru a identifica probleme complexe și a examina probleme conexe pentru a dezvolta opțiuni de rezolvare și implementa soluții. Absolventul este capabil să combine informații diverse pentru a formula soluții și genera idei de dezvoltare pentru noi produse și aplicații.

[1] Se poate opta pentru competențe sau pentru rezultatele învățării, respectiv pentru ambele. În cazul în care se alege o singură variantă, se va șterge tabelul aferent celeilalte opțiuni, iar opțiunea păstrată va fi numerotată cu 6.

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	• Cunoașterea modelelor arhitecturale ale calculatoarelor, funcționarea procesorului, utilizarea sistemelor de reprezentare a informației in calculator.
7.2 Obiectivele specifice	• Însușirea de către studenți a modelelor arhitecturale ale calculatoarelor, funcționarea procesorului, a utilizării sistemelor de reprezentare a informației in calculator. • Inițiere în programarea în limbaj de asamblare, ceea ce asigura înțelegerea arhitecturii și funcționarii unui microprocesor. • Înțelegerea impactului arhitecturii procesoarelor 80x86 asupra sistemului de operare Windows si asupra limitărilor sale. • Conștientizarea triadei arhitectura – sistem de operare – limbaje de programare si a interacțiunilor dintre acestea drept nucleu de baza a informaticii. • Conștientizarea influentei pe care principiile funcționale de baza ale arhitecturii von Neumann le au asupra modului de implementare a limbajelor de programare de nivel înalt; • Conștientizarea impactului arhitectural asupra tehnicilor de proiectare si implementare a limbajelor de programare de nivel înalt.

8. Conținuturi

8.1 Curs	Metode de predare	Observații
1. Reprezentarea datelor: Tipuri de date elementare, reprezentari binare si ordini de plasare, organizarea si memorarea datelor	Expunerea, conversația, dezbaterea, problematizarea, descoperirea	
2. Codificarea caracterelor, codificarea numerelor întregi, convenție cu semn si fara semn, bitul de semn, cod complementar, operatii aritmetice, conceptul de depasire, conversia la o locație de alte dimensiuni		
3. Performantele unui SC, arhitectura microprocesorului 80x86 – structura, registrii, calculul de adresa, moduri de adresare, adrese FAR si NEAR		
4. Unitatea executiva (EU) a microprocesorului 80x86: rolul si functiile registrilor si al flagurilor. Clasificare (Registrii si Flaguri) si studii de caz.		
5. Unitatea BIU a microprocesorului 80x86: registrii de adresa, registrii de segment, reprezentarea instrucțiunilor. Formula de specificare offset pe 32 biți si formula de specificare offset pe 16 biți. Aritmetica de pointeri		
6. Elementele limbajului de asamblare: formatul unei linii sursa, expresii, tipuri de accesare a operanzilor, operatori aritmetici, operatori pe biti, operatori de tip si tipuri de date asociate operanzilor, contorul de locații. Conversii nondistructive (si operatorii specifici)		
7. Directive standard pentru definirea segmentelor. Directive pentru definirea datelor. Directive de generare a datelor. Directivele EQU și INCLUDE		
8. Instructiuni ale limbajului de asamblare:		

instructiuni de transfer, conversii, operatii aritmetice cu semn si fara semn, operatii de deplasare si rotire de biti, operatii logice pe biti		
9. Impactul reprezentării little endian asupra accesării datelor. Constante de tip string. Reprezentare în memorie și utilizare în cadrul unor instrucțiuni de transfer.		
10. Instructiuni de salt conditionat si neconditionat, instructiuni de ciclare, instructiuni pe siruri. Conceptul de depășire și modul în care arhitectura 80x86 reactioneaza la aparitia acestei situatii		
11. Reprezentarea instrucțiunilor mașină. Formatul intern al unei instrucțiuni. Prefixe de instrucțiuni.		
12. Programarea multimodul ASM-ASM: directivele de import-export GLOBAL si EXTERN. Legarea de module scrise in limbaj de asamblare si modul de comunicare intre acestea		
13. Implementarea apelului de subprograme. Convenții de apel: CDECL și STDCALL, cod de apel, cod de intrare, cod de iesire la nivelul limbajelor de nivel inalt vs. limbaj de asamblare		
14. Legarea de module NASM cu module scrise în limbaje de nivel înalt (studiu de caz – programarea C). Exemple și discuții pentru apeluri recursive		

Bibliografie
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 – materiale postate pe pagina cursului
12. Cursuri si materiale suport postate pe site-ul cursului

8.2 Seminar / laborator	Metode de predare	Observații
S1: Introducere în limbajul de asamblare IA-32. Conversia numerelor între bazele de numerație 2, 10, 16. Reprezentarea numerelor întregi în memoria calculatorului. Reprezentarea numerelor cu semn și fără semn. Elementele limbajului de asamblare IA-32. Primul program in limbaj de asamblare		
S2: Obtinerea offsetului / valorii unei variabile. Ordinea de plasare a octetilor in memorie. Reprezentare little-endian. Instrucțiuni pentru numere cu semn și fără semn. Instrucțiuni aritmetice (inmulțiri și împărțiri). Conversii		

pentru numerele cu semn și conversii pentru numerele fără semn. Instructiuni aritmetice care tin cont de transport. Instructiuni de lucru cu stiva.		
S3: Instructiuni de comparare. Salturi condiționate și necondiționate. Instructiuni de ciclare. Operații cu șiruri.		
S4: Instrucțiuni pe șiruri. Probleme complexe cu șiruri.		
S5: Apeluri de funcții si operare cu fisiere de tip text (printf, scanf, fread, fscanf, fprintf, fclose).		
S6: Programare multi-modul folosind limbajul de asamblare.		
S7: Pregătire pentru examene: discuții și studii de caz.		
L1: Conversia între diferite baze de numerație. Conceptul de bit. Bit de semn. Cod complementar. Reprezentarea numerelor întregi cu semn. Instrumente pentru laboratoare. Structura unui program in limbaj de asamblare. Editare, asamblare, link-editare si depanare fisiere.		
L2: Expresii aritmetice simple: adunări, scaderi, inmulțiri și împărțiri.		
L3: Expresii aritmetice complexe (little-endian, conversii in reprezentarea cu semn și fără semn, declararea de variabile și constante).		
L4: instrucțiuni pe biți (operații logice, operații de deplasare și de rotire).		
L5: Operații simple cu șiruri (instrucțiuni pentru comparații, salturi condiționale și instrucțiuni repetitive).		
L6: Operațiuni complexe cu șiruri (instrucțiuni specifice limbajului de asamblare pentru lucrul la șiruri de octeți / cuvinte / dublu cuvinte / quadwords).		
L7: Apeluri de funcții. Biblioteci. Utilizarea funcțiilor externe. Convenții de apel, Apelarea unei funcții de sistem. Funcții standard msvcrt.		
L8: test		
L9: Operații cu fișiere text (deschidere, scriere, citire si închidere).		
L10: Discuții, analiză și evaluare a lucrărilor de laborator. Predarea ultimelor teme date		
L11: Programare multimodul (asm + asm)		
L12: Programare multimodul (asm + C)		
L13: Pregatire pentru examenele practice: discuții și studii de caz		
L14: test		

Bibliografie

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 – materiale postate pe pagina cursului
12. Cursuri si materiale suport postate pe site-ul cursului

9. Coroborarea conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatori reprezentativi din domeniul aferent programului

- Acest curs exista in programul de studiu al tuturor universităților importante din Romania si străinătate.
- Acest curs asigura cunoștințele de baza pe care orice programator trebuie sa la aibă.

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	Cunoasterea principiilor de baza ale domeniului	Examen scris	60%
	Verificarea înțelegerii conceptelor limbajului de asamblare		
10.5 Seminar/laborator	Rezolvarea de probleme prin aplicarea principiilor programarii pe 32 biți in limbaj de asamblare	Media notelor obținute pe teme de laborator predate	10%
	Dezvoltarea si implementarea unei soluții in limbaj de asamblare pentru o problema data	Media testelor de la laborator	30%
10.6 Standard minim de performanță			

- Pentru promovare este necesara obținerea notei minim 5 la fiecare dintre probele de evaluare.
- Prezențe: 75% prezență la activitățile de seminar, 90% prezență la activitățile de laborator.
- Studenții cu mai mult de 2 absențe nemotivate la seminar/laborator nu vor putea susține examenul în sesiunea normală și/sau în sesiunea de restanțe (activitățile de seminar/laborator sunt activități care se desfășoară pe principiul „activitate pe parcursul semestrului", și nu pot fi recuperate sau repetate pentru o eventuală reluare a

examenului (acești studenți vor trebui să repete acest curs în următorul an universitar)). Studenții cu certificate medicale pentru fiecare dintre absențe sunt exceptați de la această regulă.

11. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)[2]

Nu se aplică.

Data completării:	Semnătura titularului de curs	Semnătura titularului de seminar
	Conf. univ. dr. Mihai SUCIU	Conf. univ. dr. Mihai SUCIU
Data avizării în departament:		Semnătura directorului de departament
		Conf.dr. Adrian STERCA

[2] Păstrați doar etichetele care, în conformitate cu *Procedura de aplicare a etichetelor ODD în procesul academic*, se potrivesc disciplinei și ștergeți-le pe celelalte, inclusiv eticheta generală pentru *Dezvoltare durabilă* - dacă nu se aplică. Dacă nicio etichetă nu descrie disciplina, ștergeți-le pe toate și scrieți "*Nu se aplică.*".