# FIŞA DISCIPLINEI

| | | | | | |
|---|---|---|---|---|---|
| Denumirea disciplinei : | | BAZELE LOGICE ALE AUTOMATELOR | | | |
| Codul disciplinei: | | | | | |
| Programul de studii: | | **INGINERIA CALCULATOARELOR ÎN APLICAŢII INDUSTRIALE** | | | |
| Catedra: | | CALCULATOARE SI AUTOMATIZARI | | | |
| Facultatea: | | DE INGINERIE „Hermann Oberth” | | | |
| Universitatea: | | „Lucian Blaga” Sibiu | | | |
| Anul de studiu**:** | I | Semestrul | I | Tipul de evaluare finală | C |
| Regimul disciplinei (DI=obligatorie/ DO=opțională/DF=liber aleasă): | | | DI | Numărul de credite: | 5 |
| Categoria formativă a disciplinei (DF=fundamentală.; DI=inginereşti; DS=specialitate; DC=complementară) | | | | | DF |
| Total ore din planul de învăţământ | | 28 | | Total ore pe semestru: | 28 |
| Titularul disciplinei: Conf.univ.dr.ing. Adrian FLOREA | | | | | |

| **Numărul total de ore (pe semestru) din planul de învăţământ** | | | | | |
|---|---|---|---|---|---|
| Total ore/ semestru | **C** | **S** | **L** | **P** | **Total** |
| | 14 | | 14 | | 28 |

| | |
|---|---|
| **Obiective:** | Acest curs reprezintă o introducere axată pe **înţelegerea principiilor fundamentale ale ştiinţei şi ingineriei calculatoarelor**. Abordare ***bottom-up*** centrată în principal pe două componente:<br>• **Structura de bază a calculatoarelor** folosind simulatorul LC-3 (*Little Computer* – www.ece.utexas.edu/~ambler/ee306/Software&Doc/**LC3**WinGuide.pdf<br>• **Exemplificarea la nivelul limbajului C şi implicaţiile în hardware a structurilor de control, a diverselor tehnici de alocare a memoriei, lucrul cu stiva, apel de funcţii şi revenire, stivele de date aferente funcţiilor, transmiterea parametrilor în cazul apelului, recursivitate,** pointeri, tablouri şi structuri elementare de date.<br>Aspectul novator al abordării constă în prezentarea **interfeţei arhitectură (ISA, limbaj de asamblare specific) – aplicaţie de nivel înalt** (incluzând **compilatorul ca şi generator de cod pentru respectiva arhitectură).**<br><br>Relativ la **interfata Hardware – Software, cu toate ca putem fi specialisti intr-una din cele doua, trebuie intelese capabilitatile si limitarile ambelor componente** ale sistemului de calcul. |

| | |
|---|---|
| **Competenţe specifice disciplinei** | **1. Cunoaştere şi înţelegere:**<br>• Întelegerea principiilor de bază ale structurii sistemelor de calcul<br>• Proprietatile algoritmilor, Reprezentarea acestora prin *scheme logice* si prin *pseudocod*, Respectarea principiilor programării structurate în procesul de elaborare a algoritmilor<br>• Identificarea tipurilor de informaţii reprezentate într-un calculator; reprezentarea informaţiilor; conversii de valori între sisteme de numeraţie; aritmetică binară; operaţii logice, legile lui De Morgan<br>• Identificarea circuitelor digitale combinaţionale si secvenţiale care compun o microarhitectură generică (LC-3): tranzistoare MOS, porţi logice (fundamentale), bistabili, structuri logice (Multiplexoare, Decodificatoare, Sumatoare), registre, memorii.<br>• Înţelegerea modelului de execuţie ***von Neumann***, (componentele modelului şi ciclul instrucţiunii) aferent unei arhitecturi simple LC-3, a limbajului de asamblare propriu. Cunoaşterea interfeţei cu tastatura şi monitorul, a apelurilor sistem (întreruperi software reprezentând servicii ale sistemului de operare prin instrucţiuni dedicate), modurile de adresare pentru instrucţiunile cu referire la memorie (load / store), mecanismul de apel şi revenire din subrutină.<br>• Exemplificarea la nivelul limbajului C şi implicaţiile în hardware a structurilor de control, a diverselor tehnici de alocare a memoriei, lucrul cu stiva, apel de funcţii şi revenire, stivele de date aferente funcţiilor, transmiterea parametrilor în cazul apelului, recursivitate, pointeri, tablouri şi structuri elementare de date. |
| | **2. Explicare şi interpretare:**<br>• Cunoaşterea şi utilizarea adecvată a noţiunilor specifice ştiinţei calculatoarelor, explicarea şi interpretarea conţinuturilor teoretice şi practice ale disciplinei<br>• Explicarea unor noţiuni din discipline sau domenii nestudiate încă şi care vor fi introduse la un nivel sumar<br>• Explicaţii la întrebările studenţilor<br>• Indicaţii pentru întocmirea referatelor şi temelor de casă |
| | **3. Instrumental – aplicative:**<br>• Scrierea de programe in limbaj de asamblare LC-3, realizarea de scheme logice / pseudocod pentru orice problemă propusă.<br>• Rezolvarea unor probleme de proiectare folosind circuite de memorie, regiştrii, automate programabile. |
| | **4. Atitudinale:**<br>♦ Exprimarea unui mod de gândire creativ, în structurarea şi rezolvarea problemelor<br>♦ Conştientizarea impactului social, economic al ştiinţei calculatoarelor<br>♦ Formarea obişnuinţelor de a recurge la concepte şi metode informatice de tip algoritmic specifice în abordarea unei varietăţi de probleme.<br>♦ Manifestarea unor atitudini favorabile faţă de ştiinţă şi de cunoaştere în general<br>♦ Manifestarea unei atitudini responsabile şi pozitive faţă de profesia didactică |

| Conţinutul tematic (descriptori) | **TEMATICA CURSURILOR** | | |
|---|---|---|---|
| | Nr. crt. | Denumirea temei | Nr. ore |
| | 1. | Introducere în sistemele de procesare a informaţiei. Calculatorul – dispozitiv universal de calcul. Structură. Nivele de transformare (abstractizare). | 2 |
| | 2. | Reprezentarea informaţiilor. Entropia informaţională. Exemplificare diferenţa *dată – informaţie*. Sisteme de numeraţie. Reprezentarea numerelor întregi: fără semn, cu semn (MS, C1, C2). Conversii de valori între sisteme de numeraţie. Reprezentarea numerelor fracţionare: standardul IEEE 754. Coduri pentru detectarea şi corectarea erorilor. Operaţii aritmetico-logice în binar. Depăşirea domeniului de reprezentare. Extensia semnului. Coduri ASCII şi Unicode. Conversia din caracter în numeric şi invers. | 2 |
| | 3. | Algoritmi. Caracteristicile algoritmilor. Reprezentarea algoritmilor prin scheme logice si prin pseudocod. | 2 |
| | 4. | Modelul arhitectural *von Neumann*. Componente de bază. Principiile procesării instrucţiunilor. Ciclul instrucţiunii Ciclul instrucţiunii văzut ca un automat cu număr finit de stări. Tipuri de instrucţiuni. Ceasul procesorului. Oprirea sistemului de calcul. Modelul de procesare „*von Neumann*”. Codificarea instrucţiunilor. Determinarea numărului de instrucţiuni executate pe un anumit microprocesor într-o secundă. Determinarea numărului de accese la memorie pentru fiecare tip de instrucţiune. | 2 |
| | 5. | Limbajul de asamblare aferent arhitecturii LC-3. Asamblorul. Etapele generării codului maşină. Tabela de simboluri. Stiva – structură. Principiu de funcţionare. Operaţii aferente (*Push & Pop*). Implementare hardware şi software. Modul de lucru prin întreruperi hardware. Operaţii aritmetice folosind stiva. Rolul stivei in tratarea intreruperilor hardware imbricate. | 2 |
| | 6. | Generarea codului obiect pentru o arhitectură dată. Implementarea în hardware a funcţiilor din programele de nivel înalt. Stiva de date aferentă funcţiilor. Transferul parametrilor în cazul apelului. Translatarea codului de la nivel *high (C)* la nivel *low (asamblare – LC3)*. | 2 |
| | 7. | Introducere în *recursivitate*. Recursivitate vs. Recurenţă. Rolul stivei în implementarea recursivităţii. Comparaţia dintre *recursiv* şi *iterativ* în alegerea algoritmului de rezolvare a problemelor. Avantaje / dezavantaje. Implementarea recursivităţii la nivelul stivei de date. Eroarea *Stack Overflow*. Tipuri de funcţii recursive. Eliminarea recursivităţii. Pointeri. Tablouri. Transferul parametrilor prin adresa. Pointeri spre functii. Legătura dintre nivelul *high* şi *low* văzută prin intermediul modurilor de adresare. Alocarea şi accesarea indirectă de variabile. | 2 |

<table>
<tr><td></td><td colspan="3">TEMATICA SEMINARIILOR/LABORATOARELOR/PROIECTULUI</td></tr>
<tr><td></td><td>1.</td><td>Reprezentarea numerelor întregi: fără semn, cu semn (MS, C1, C2). Conversii din diverse baze. Reprezentarea numerelor fracţionare: standardul IEEE 754. Operaţii aritmetico-logice în binar. Coduri ASCII şi Unicode. Conversia din caracter în numeric şi invers.</td><td>2</td></tr>
<tr><td></td><td>2.</td><td>Realizarea de scheme logice si pseudocod pentru diverse aplicatii. Apeluri de subrutine – directe şi indirecte. Reveniri. Întreruperi software la nivel low. Salvarea şi restaurarea regiştriilor. Strategiile “caller-save” respectiv “callee save”. Aplicaţii în LC-3 cu întreruperi software şi apeluri imbricate de subrutine.</td><td>2</td></tr>
<tr><td></td><td>3.</td><td>Aplicaţii cu stiva şi operaţiile Push / Pop. Codificarea ASCII to Binary şi Binary to ASCII în LC-3.</td><td>2</td></tr>
<tr><td></td><td>4.</td><td>Examinare parţială din materia parcursă (accent pe scheme logice si pseudocod si pe modelul arhitectural von Neumann: componente de bază si principiile procesării instrucţiunilor).</td><td>2</td></tr>
<tr><td></td><td>5.</td><td>Aplicaţii scrise recursiv şi iterativ. Avantaje / dezavantaje. Implementarea recursivităţii la nivelul stivei de date. Evidenţierea erorii Stack Overflow.</td><td>2</td></tr>
<tr><td></td><td>6.</td><td>Aplicaţii cu pointeri şi tablouri. Transferul parametrilor prin adresa. Pointeri spre funcţii. Legătura dintre nivelul high şi low văzută prin intermediul modurilor de adresare. Alocarea şi accesarea indirectă de variabile.</td><td>2</td></tr>
<tr><td></td><td>7</td><td>Ghid de utilizare LC-3 Edit şi LC-3 Simulate. Implementarea software a aplicaţiilor anterior prezentate la curs şi laborator. Depanare si simularea folosind cele doua utilitare.</td><td>2</td></tr>
<tr><td>Metode de predare / seminarizare</td><td colspan="3">Metodele de predare la curs se bazeaza pe folosirea videoproiectorului, dar in anumite cazuri in care se impune se foloseste creta si tabla pentru diverse demonstratii sau rezolvari de exemple / probleme.<br>La seminar sunt rezolvate o serie de probleme la tabla, se propun si se rezolva (fie ca tema, fie ca test, fie ca pure exercitii) teste grila. De asemenea, sunt folosite calculatoarele pentru rezolvarea unor probleme de programare in limbaj de asamblare folosind simulatorul LC-3 si memory (vezi mai jos).<br>Simulatorul LC-3 (for Little Computer 3) – disponibil atât sub sistem de operare Windows cât şi Linux. LC-3 descrie funcţionarea unei arhitecturi pe 16 biţi; arhitectura înglobează cele mai importante caracteristici ale procesoarelor Intel 8088 – folosit în primul sistem IBM PC în 1981, Motorola 68000 – folosit la Apple Macintosh 1984, sau Intel Pentium III integrat în sistemele anilor 2000.<br>Simulatorul arhitecturii LC-3 permite testare / depanare a programelor scrise în limbaj de asamblare, stabilire de puncte de întrerupere. De la adresa http://users.ece.utexas.edu/~ambler/ee306/software&doc.htm pot fi descărcate în mod gratuit pentru utilizare în scop didactic atât simulatorul LC-3 (LC301.exe), un simulator realizat în Macromedia Flash Player versiunea 6, care descrie operaţiile de citire – scriere dintr-o memorie cu 4 locaţii având 3 biţi fiecare (memory.exe), precum şi ghidul de utilizare al simulatorului LC-3.</td></tr>
</table>

| | | |
|---|---|---|
| Stabilirea notei finale (procentaje)[*)] | - răspunsurile la examen/colocviu(evaluare finală) | 60 |
| | - teste pe parcursul semestrului | 20 |
| | - răspunsurile finale la lucrările practice de laborator | |
| | - activitaţi gen teme/referate/eseuri/traduceri/proiecte etc. | 20 |
| | - teme de control | |
| | - alte activităti*(precizaţi)*…… …….. | |
| | - TOTAL | 100% |

[*)] Condiţii de promovare: minim nota 5 pe toate componentele (Examen / Teste din timpul semestrului / Teme de casa).

| | |
|---|---|
| Descrieţi modalitatea practică de evaluare finală, E/V ( de exemplu: lucrare scrisă (descriptive şi/sau test grilă şi/sau probleme etc.), examinare orală cu bilete, colocviu individual ori în grup, proiect etc)<br>Evaluarea finală va cuprinde lucrare scrisă compusă din (partial) teste grilă si probleme. | |
| **Cerinţe minime pentru nota 5**<br>• Cunoaşterea, înţelegerea şi explicarea noţiunilor elementare specifice ştiinţei şi ingineriei calculatoarelor.<br>• Interes constant manifestat pentru însuşirea disciplinei.<br>• Îndeplinirea condiţiilor minime obligatorii în privinţa temelor de casă, referatelor şi a testelor date pe parcursul semestrului. | **Cerinţe pentru nota 10**<br>• Cunoaşterea, înţelegerea şi explicarea noţiunilor specifice ştiinţei şi ingineriei calculatoarelor.<br>• Punctaj maxim pentru activitatea din timpul semestrului (seminar) care include temele de casă, referatele şi a testele parţiale date.<br>• Punctaj maxim pentru lucrarea scrisă la examenul final.<br>• Utilizarea logică şi creativă a noţiunilor disciplinei. |
| **TOTAL ore studiu individual (pe semestru) =2 ore x 14 sapt. = 28 ore** | |

| | |
|---|---|
| **Bibliografia** | **Minimală obligatorie**:<br>1. Patt Yale, Patel Sanjay – „*Introduction to Computing Systems: from bits & gates to C & beyond*", McGraw-Hill Higher Education, 2001.<br>2. Patterson David, Hennessy John – *Computer Organisation and Design*: *The Hardware/Software Interface*, Third Edition, Elsevier, 2005, ISBN: 1-55860-604.<br>3. Cormen, T. H., Leiserson, C. E., Rivest, R. L. – *Introduction to Algorithms*. McGraw-Hill, New York, 1990.<br>4. Pop Vasile – *Analiza şi sinteza dispozitivelor numerice*. Curs litografiat, Institutul Politehnic Timişoara, 1986, vol. I şi II.<br>5. Vinţan Lucian – *Organizarea şi proiectarea microarhitecturilor*, http://webspace.ulbsibiu.ro/lucian.vintan/html/Organizarea.pdf.<br>6. Mârşanu Radu – „*Sisteme de calcul*", Manual pentru licee de informatică, cls. IX-a, Editura Didactică şi Pedagogică, Bucureşti, 1996.<br>7. Florea Adrian – *Introducere în Ştiinţa şi Ingineria Calculatoarelor. Interfaţa Hardware – Software*, Editura Matrix ROM, Bucureşti, ISBN 978-973-755-264-8, 2007 (313 pg.).<br>**Complementară:**<br>8. Zaharia Mihai, Leon Florin – „*Limbajul C de la Zero la Student*", Editura Politehnium Iasi, 2004.<br>9. Dana Lica, ş.a. – *Fundamentele programării*, Editura L&S Soft, Bucureşti.<br>10. Florea Adrian – *Predicţia dinamică a valorilor în microprocesoarele generaţiei următoare*, Editura MatrixROM, 2005.<br>11. Golomety A. – *Proiectarea Translatoarelor*, Editura Universităţii "Lucian Blaga", Sibiu, 1997.<br>12. Gorgan D., Sebestyen-Pal Gh., *Computer Design*, Editura albastra, Cluj-Napoca,. ISBN 973-650-123-X, 2005.<br>13. Katz R. H., Borriello G., *Contemporary Logic Design (2nd Edition)*, Prentice Hall, 2005. |
| Lista materialelor didactice utilizate în procesul de predare:<br>VideoProiector / RetroProiector, Calculatoare (simulatoare software), Tabla + Creta | |

| Coordonator de Disciplină | Grad didactic, titlul, prenume, numele | Semnătura |
|---|---|---|
| | Conf.univ.dr.ing. Adrian FLOREA | |