Szakmai szigorlat

TANTÁRGYKÓD
NAISS1SAND

SZAK
Mérnök informatikus BSc

OKTATÓ(K)
Dr. Hajnal Éva, Dr. Orosz Tamás

TAGOZAT
Nappali

FÉLÉV
2016-17

TEMATIKA

Szakmai alapozó mérnöki ismeretek

Felkészülést segítő témakörök

1. Kombinációs hálózatok tervezése és vizsgálata: logikai érték, logikai függvények, kanonikus konjunktív és diszjunktív függvények. Boole-algebrai azonosságok, tulajdonságok. Kétváltozós logikai függvények elnevezése. Kombinációs hálózatok leírási módjai: logikai függvények, igazságtáblázat, logikai kapcsolási rajz, minterm és maxterm alakok. Univerzális műveletek: NAND és NOR kapuk. Egyszerűsítési szabályok: Karnaugh tábla, számjegyes minimalizálási eljárás, szimmetrikus függvények.
2. Ideális és valódi építőelemek, a valódi építőelemek jellemzői: A nemidealitások okai, jelterjedési késési idő, kombinációs hálózatok hazárdjai. Hazárdok típusai, kiküszöbölésük.
3. Sorrendi hálózat fogalma, alapmodelljei (Mealy és Moore), sorrendi hálózatok csoportosítása, szinkron és aszinkron működés. Szinkron hálózatok tervezése és vizsgálata: Tároló alapelemek, flip-flop típusok és ezek alkalmazástechnikája, kapukból és tároló elemekből álló hálózat tervezése.
4. Szinkron hálózatok vizsgálata, állapottáblázat, állapotgráf, karakterisztikus egyenlet. Szinkron és aszinkron hálózatok tervezési módszerei. Tipikus szinkron hálózatok: Számlálók, regiszterek, összetett szinkron rendszerek. Aszinkron és szinkron hálózatok állapotkódolása: szomszédos kódolás, önfüggő csoportok, kritikus versenyhelyzet, átvezető állapotok.
5. Fontosabb logikai áramkörcsaládok alapáramkörei, jellemzői RTL, DTL, TTL, CMOS. Tároló alapáramkörök, tároló cellák tulajdonságai, működési elvek.
6. Digitális áramkörök statikus és dinamikus jellemzői, digitális jelek fel-lefutási és késleltetési jellemzői, alapkapuk transzfer karakterisztikái, statikus és dinamikus teljesítményfelvéte!
7. Programozható logikai áramkörök felépítése, legfontosabb részegységei, és programozása VHDL-ben. Digitális rendszerek viselkedés és strukturális leírási módszerei VHDL-ben, konkurens utasítások, szekvenciális utasítás típusok, strukturális leírás, deklaráció és példányosítás.

Programozási és szoftvertechnológiai ismeretek

Felkészülést segítő témakörök

Az egyes témaköröknél szükséges az adott téma általános bemutatása, példák bemutatása, az algoritmusok ismertetése pszeudokóddal, az algoritmusok szemléltetése konkrét példán keresztül, az algoritmusok hatékonyságának elemzése, valamint a vizsgáztató kérésének megfelelően az algoritmust megvalósító C# kód megadása.

A szigorlaton – a szigorlati jelleg folytán – olyan kérdések is várhatók, amik több témakör együttes ismeretét feltételezik, akár több tantárgy anyagán átívelve. (Például: Ismertesse és hasonlítsa össze a beillesztéses, quicksort és kupac-rendezéseket!)

1. Programozási tételek

Sorozatszámítás, eldöntés, kiválasztás, lineáris keresés, megszámlálás, maximumkiválasztás. Másolás, kiválogatás, szétválogatás,metszet, egyesítés, összefuttatás.

2. Programozási tételek egymásra építése

Másolás és sorozatszámítás; másolás és maximumkiválasztás. Megszámolás és keresés. Maximumkiválasztás és kiválogatás. Kiválogatás és maximumkiválasztás; kiválogatás és másolás.

3. Rendezések I.

Egyszerű cserés rendezés, minimumkiválasztásos rendezés, buborékos rendezés, javított buborékos rendezés, beillesztéses rendezés, Shell rendezés.

4. Keresések

Lineáris keresés rendezett sorozatban, logaritmikus keresés. Programozási tételek megvalósítása rendezett sorozatok esetén.

5. Halmazok

Halmazreprezentáció, rendezett sorozatból a többször előforduló elemek elhagyása, egy rendezett sorozat halmaz tulajdonságának vizsgálata, tartalmazás, részhalmaz, halmazműveletek (unió, metszet, különbség, komplementer, szimmetrikus differencia).

6. Rekurzió I.

Rekurzív algoritmusok jellemzői. Példák rekurzióra: faktoriális, Fibonacci számok, binomiális együtthatók.Rekurzív algoritmusok jellemzői. Példák rekurzióra: szöveg megfordítás, palindrom számok, hatványozás, Hanoi tornyai.

7. Rekurzió II.

Rekurzív algoritmusok jellemzői, keresések rekurzív megvalósítása. Quicksort, őrszem elem kiválasztásának módjai.

8. Oszd meg és uralkodj elvű algoritmusok

Oszd meg és uralkodj elv, maximumkiválasztás, k-adik legkisebb elem meghatározása, a Quicksort algoritmus őrszem elemének kiválasztása.

9. Érték és referencia típusú változók

Érték és referencia típusú változók, metódusok paramétereinek érték és referencia szerinti átadása.

10. Programozási Paradigmák rövid összefoglalása

Strukturált és OO paradigma, OOP kialakulásának oka, a strukturált és objektumorientált program bemutatása.

11. OOP modellezés és programkészítés

Programozás mint modellkészítés, modellezés objektumokkal, objektumorientált program és készítésének folyamata.

12. OOP paradigma I.

Az OO paradigma alapelvei, osztály részei és a példányok, objektumreferencia, kódújrafelhasználás.Öröklődés. Polimorfizmus.Interfészek szerepe, felépítése.Komponens alapú programozás.

13. Eseménykezelés

Eseménykezelés elvi háttere. Hagyományos megoldások.Alapvető eseménykezelési módszerek áttekintése (származtatás, interfészek, metódusreferenciák).

14. Tesztelés, hibakeresés

Tesztelési és hibakeresési technikák. Kivételkezelés előnyei és hátrányai. Saját kivétel készítése. Többrétegű architektúrák.

15. Rekurzió III.

Rekurzió technikai háttere (verem, lokális változók, hívás módja, visszatérési értékek kezelése).Backtrack algoritmusok.

16. Rendezések folytatás

További összehasonlító rendezések: Shell rendezés, kupacrendezés. Nem összehasonlító renzedések: radix-, edényrendezés.

17. Adatszerkezetek

Láncolt listák, egyszerű láncolt lista felépítése, műveletei. Rendezett láncolt lista. Egyéb speciális listák.

18. Fa adatszerkezetek I.

Bináris fa, bináris keresőfa. Beszúrás, keresés és törlés.

19. Fa adatszerkezetek II.

B-fa felépítése. AVL-fa. Piros-fekete fák. Keresés.

20. Hasító táblázatok

Hasító függvények. Kulcsütközések kezelése.

21. Gráfok

Irányított és irányítatlan gráfok. Gráf adatstruktúra. Feszítőfák, Dijkstra algoritmus.

22. Gráfbejárások.

Útkeresés, összefüggő komponensek keresése, topológiai rendezés.

23. A szoftverfejlesztés problémái – a szoftver, mint termék jellemzői

A fejlesztés szakaszainak költségarányai, a szoftverkrízis, a szoftvertermék jellemzői

24. Hagyományos szoftver életciklus modellek

Modellezési koncepció, a modellek csoportosítása, vízesés modell, „V” modell, evolúciós modell, formális rendszerfejlesztés.

25. Újrafelhasználás-orientált és inkrementális életciklus modellek

Újrafelhasználás-orientált fejlesztés, inkrementális fejlesztés, RAD modell, Boehm-féle modell.

26. Agilis szoftverfejlesztés

Agilis szoftverfejlesztés, Extrém programozás, SCRUM módszer.

27. A RationalUnifiedProcess

Hagyományos vs. agilis módszerek, RUP, szoftverfejlesztés más utakon.

28. Szoftver projekt menedzsment

A menedzsment fő területei, jellemzői, feladata, folyamatai, a projekt előkészítése, tervezése.

29. Szoftver követelmény analízis és tervezés

Követelmények meghatározása, követelmények tervezése, a tervezés, tevékenységei, modellje,

tervezési koncepciók: absztrakció, modellalkotás.

30. Szoftver verifikáció, validáció, tesztelés

Verifikáció, validáció, tesztelés – az OO rendszerek tesztelése nélkül.

31. Szoftver tesztelés és átvizsgálás

OO rendszerek tesztelése, átvizsgálások, automatizált elemzés, szoftverevolúció.

32. Az UML eszközei – kezdeti modellezés

Bevezetés, diagramtípusok, UseCase diagram, profil diagram Csomag diagram, telepítési diagram.

33. Szerkezetmodellezés az UML eszközeivel

Osztálydiagram, objektum diagram, komponens diagram, kompozit szerkezet diagram.

34. Viselkedés modellezés UML segítségével

A viselkedés modellezés diagramjainak áttekintése, állapotgép diagram, időzítés diagram.

35. Interakció és kommunikáció modellezése az UML eszközeivel

Szekvencia diagram, kommunikációs diagram, aktivitás diagram, interakció áttekintő diagram.

ÉRDEMJEGY KIALAKÍTÁSA

A szigorlat a szakmai törzsanyag két területére épül:

- rendszertechnikai ismeretek (szakmai alapozó mérnöki ismeretek), valamint
- programozási és szoftvertechnológiai ismeretek.

A vizsga két részből áll:
Első rész (60 perc): írásbeli beugró a rendszertechnikai ismeretek részből (30 perc); valamint a programozási és szoftvertechnológiai ismeretekből (30 perc).
Második rész (75 perc, szünet, majd újabb 75 perc): Részletes kifejtést igénylő kérdések megválaszolása mindkét terület témáiból. Rendszertechnikai ismeretekből feladatmegoldás, míg programozás és szoftvertechnológia területből elmélet/feladatmegoldás.
Az első rész mindkét területének minimum elégségesre teljesítése előfeltétele a második részre bocsátásnak. Az első rész csak szűrő jellegű, sikertelen teljesítése esetén a szigorlat elégtelen, a sikeres teljesítés a második rész megírásának jogát jelenti, további következménnyel nem bír.

A két terület kérdéseire adott válaszok eredményei 50-50%-os megosztásban számítanak a végső értékelésben.
A szakmai szigorlat sikeres, ha mindkét vizsgarész mindkét tematikájából legalább elégséges osztályzatot szerez a hallgató.
Részben sikerült vizsga (pl. egyik terület sikeres) átvitele másik vizsgaalkalomra nem lehetséges! Szintén nem lehetséges az első rész elfogadtatása egy következő vizsgaalkalommal.
A szigorlat kijavítása után lehetőséget biztosítunk a hallgatóknak a dolgozat javításának megtekintésére.
A második rész mindkét részét az alábbi százalékos skála szerint értékeljük: 0-50 elégtelen; 51-63 elégséges; 64-75 közepes; 76-87 jó; 88-100 jeles.
A kapott két jegy átlaga adja a szigorlat végső jegyét a kerekítés szabályi szerint.
A hallgató korábban nyújtott kiemelkedő teljesítménye alapján megajánlott jegyet kaphat - amit nem köteles elfogadni - a következők szerint:
a D tanterv szerint teljesített szigorlati tárgyak (Digitális technika, Digitális rendszerek, Programozás I., Programozás II., Szoftvertechnológia I.) jegyei között nem lehet közepesnél rosszabb;

- ha az 5 tárgy átlaga >= 4,60 és <= 5,00 --> a szigorlat osztályzata jeles(5);

VIZSGA MÓDJA

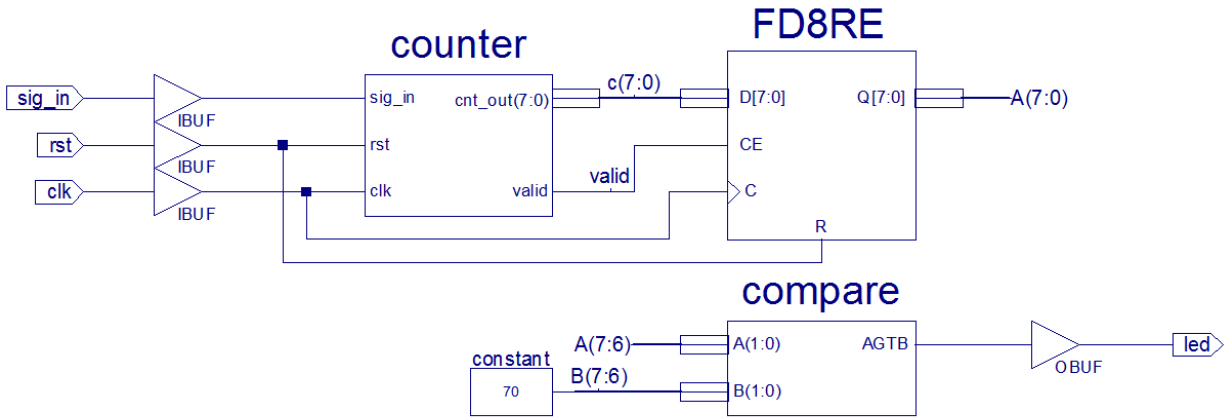
Rendszertechnikai ismeretek (minta)

Beugró rész (minta)

1. Mi az alapvető különbség a kombinációs és sorrendi logikai hálózatok között? (5 pont)
2. Hogyan lehet megakadályozni a nem ideális kombinációs logikai hálózatokban kialakuló hazárdjelenségeket? (5 pont)
3. Mi a jelentősége logikai hálózatok tervezésekor a logikai függvények egyszerűsítésének?(5 pont)
4. Írja fel a De-Morgan azonosságokat! (5 pont)
5. Miért alacsonyabb a CMOS logikai áramkörök statikus teljesítményfelvétele a TTL áramkörökéhez képest? (5 pont)
6. Minimum hány tároló-elemmel valósítható meg egy N állapotú bináris számláló? (5 pont)
7. Mi a legfőbb előnye a CMOS logikának a TTL-el szemben? (5 pont)
8. A hardverleíró nyelvekben mi a különbség a szekvenciális és a konkurens utasítások között? (5 pont)

Írásbeli rész (minta)

Az alábbiakban egy periódusidő mérésére alkalmas kapcsolás blokkvázlatát láthatjuk. A **sig_in** bemenő jel egy 50%-os kitöltési tényezőjű négyszögjel. A 8-bites **counter** blokk a bemenő jel magas értéke alatt számol, és a **valid** kimenetén jelzi a hozzá csatlakozó 8-bites regiszter (**FD8RE**) felé, hogy a számérték eltárolható. A **compare**blokk a regiszter kimeneti jelét összehasonlítja a megadott **B** konstanssal (a két legmagasabb helyiértéken) és a kimenetén jelzi, ha a regiszterben tárolt (a periódusidővel arányos) számérték meghaladja **B**-t.



1. Rajzolja fel a logikai bemenetek CMOS buffer-ének (**IBUF**) kapcsolási rajzát! Tüntesse fel a védődiódákat is! (2 pont)
2. Adja meg a **counter** blokk VHDL kódját. A használt könyvtárakat nem kell feltüntetni. (5pont)
3. Adja meg a **compare** blokk kapuszintű kapcsolási rajzát. (5pont)
4. Milyen hexadecimális számértéket tárolna a számláló 1 MHz-es (**clk**) órajel és 5 kHz-es bemeneti (**sig_in**) jel esetén. (2 pont)
5. Rajzolja le, milyen tranzisztoros kapcsolással lehetne a **led** kimeneti jel felhasználásával egy LED-et meghajtani. (2 pont)

Szoftver feladatsor (minta)

Beugró rész (minta)

1. Írja le röviden az „Oszd meg és uralkodj” elvű algoritmusok jellemzőit! (6 pont)
2. Írja le pszeudokóddal a logaritmikus keresés rekurzív megvalósítását! Adja meg a bemeneti és kimeneti változókat is! (21 pont)
3. OOP paradigma esetében mit értünk „absztrakt osztály” alatt? (6 pont)
4. Milyen feladatok esetében használható hatékonyan a visszalépéses keresés? (6 pont)
5. Írja le pszeudokóddal a bináris keresőfa „preorder” bejárásának algoritmusát! (15 pont)
6. Sorolja fel a SCRUM-ban értelmezett szerepköröket! (5 pont)
7. Sorolja fel a tervezési folyamat tevékenységeit! (6 pont)
8. Sorolja fel az UML viselkedés diagramjait! (7 pont)

Írásbeli rész (minta)

1. Rendező algoritmusok
   a. Írja le, hogy mi a rendező algoritmusok feladata, milyen feltételek teljesülése esetén használhatók! (3 pont)
   b. Csoportosítsa a megismert rendező algoritmusokat! (6 pont)
   c. Ismertesse az alábbi rendező algoritmusokat a megismert pszeudokód formátum használatával:
      i. Javított beillesztéses rendezés (12 pont)
      ii. Kupacrendezés (20 pont)
   d. A megismert rendező algoritmusok (nem csak a fenti kettő) közül milyen esetben melyiket érdemes alkalmazni? Indokolja is a válaszát kitérve az egyes algoritmusok erősségeire és gyengeségeire is! (13 pont)
2. Ismertesse a lényeget kiemelve, (max. 2 oldal) terjedelemben az újrafelhasználás-orientált és inkrementális életciklus modelleket! (18 pont)