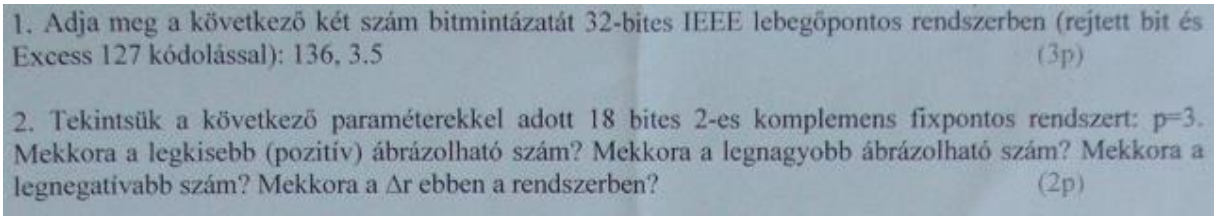
ZH TÍPUSFELADATOK

Tartalom

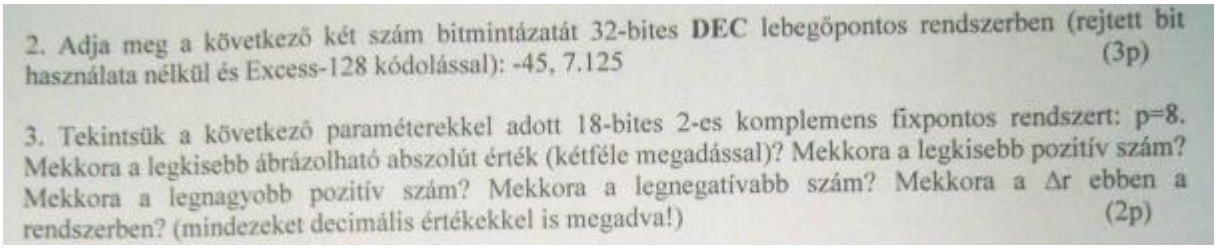
>>> 1. számátalakítós 1
>>> HAMMING 3
>>> RANDOM KÓDOK A VÉGÉN (VHDL) 3
>>> LEBEGŐPONTOS FOS 7
>>> FOLYAMAToS FOSMANÓ 7
>>> SZINTÉZIS 8
>>> MEMÓRIA FOS 8
>>> FA, RCA 9
>>> LACA, időszükséglet 9
>>> CÍMŰ SZÁMÍTÓ BLOKKDIAGRAM 10
>>> DIREKT FOS 10
>>> ELMÉLET 11
>>> RISC CISC 11
>>> HÁLÓZATOK 11
>>>REGISZTERES D TÁROLÓS FOS TAVALYRÓL 11
>>> HARVARD, NEUMANN 11
>>> LOGIKAI FÜGGVÉNYEK 12
>>> ALU 12
>>> REGISZTER ABLAKOZÁS 12
>>> EGYÉB ELMÉLET 12

>>> 1. számátalakítós



1. Adja meg a következő két szám bitmintázatát 32-bites IEEE lebegőpontos rendszerben (rejtett bit és Excess 127 kódolással): 136, 3.5 (3p)

2. Tekintsük a következő paraméterekkel adott 18 bites 2-es komplemens fixpontos rendszert: p=3. Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)



2. Adja meg a következő két szám bitmintázatát 32-bites **DEC** lebegőpontos rendszerben (rejtett bit használata nélkül és Excess-128 kódolással): -45, 7.125 (3p)

3. Tekintsük a következő paraméterekkel adott 18-bites 2-es komplemens fixpontos rendszert: p=8. Mekkora a legkisebb ábrázolható abszolút érték (kétféle megadással)? Mekkora a legkisebb pozitív szám? Mekkora a legnagyobb pozitív szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (mindezeket decimális értékekkel is megadva!) (2p)

2. Adja meg a következő két szám bitmintázatát 32-bites **DEC** lebegőpontos rendszerben (rejtett bit használata nélkül és Excess-128 kódolással): 14, -130.75 (3p)

3. Tekintsük a következő paraméterekkel adott 16-bites 2-es komplemens fixpontos rendszert: p=7, Mekkora a legkisebb ábrázolható abszolút érték (kétféle megadással)? Mekkora a legkisebb pozitív szám? Mekkora a legnagyobb pozitív szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (mindezeket decimális értékekkel is megadva!) (2p)

3. Adja meg a DEC 32 bites, normalizált lebegőpontos rendszer jellemző paramétereit (V_M, V_E, V_{FPN}, NLM, NLE, NLM)! Adottak a következő értékek: r_b=2, r_e=2, e=8, Excess 128, m=p=24. (3p)

3. Adja meg az IBM 32 bites, normalizált lebegőpontos rendszer jellemző paramétereit (V_M, V_E, V_{FPN}, NLM, NLE, NLM)! Adottak a következő értékek: r_b=16, r_e=2, e=7, Excess 64, m=p=6. (3p)

2. Tekintsük a következő paraméterekkel adott 15 bites 2-es komplemens fixpontos rendszert: p=7, Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

2. Tekintsük a következő paraméterekkel adott 16 bites 2-es komplemens fixpontos rendszert: p=5, Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

2. Tekintsük a következő paraméterekkel adott 16 bites 2-es komplemens fixpontos rendszert: p=5, Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

2. Tekintsük a következő paraméterekkel adott 17 bites 2-es komplemens fixpontos rendszert: p=6, Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

2. Tekintsük a következő paraméterekkel adott 17 bites 2-es komplemens fixpontos rendszert: p=6, Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

2. 2. Adott egy normalizált lebegőpontos rendszer a következő értékekkel: r_b=2, r_e=2, e=6 (Excess-32), m=9, p=9 (HB beállítva!). Számítsa ki a rendszer jellemző paramétereit (V_M, V_E, V_{FPN}, NLM, NLE, NRV)! (3p)

2. Adott egy normalizált lebegőpontos rendszer a következő értékekkel: r_b=2, r_e=2, e=6 (Excess-32), m=10, p=9 (HB beállítva!). Számítsa ki a rendszer jellemző paramétereit (V_M, V_E, V_{FPN}, NLM, NLE, NRV)! (3p)

2. Tekintsük a következő paraméterekkel adott 15 bites 2-es komplemens fixpontos rendszert: p=7, Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

2. Tekintsük a következő paraméterekkel adott 16 bites 2-es komplemens fixpontos rendszert: p=5. Mekkora a legkisebb (pozitív) ábrázolható szám? Mekkora a legnagyobb ábrázolható szám? Mekkora a legnegatívabb szám? Mekkora a Δr ebben a rendszerben? (2p)

>>> HAMMING

4. Tervezze meg és rajzolja fel a 7-bites Hamming-kódú kódszó hibajavító áramkörét! (Milyen egységeket kell felhasználni, hogyan kell a paritásbit-csoportokat képezni az adat és kódbitek megfelelő pozícióinak felírásával?) (3p)

(3p)
4. Tervezze meg és rajzolja fel a 7-bites Hamming-kódú kódszó hibajavító áramkörét! (Milyen egységeket kell felhasználni, hogyan kell a paritásbit-csoportokat képezni az adat és kódbitek megfelelő pozícióinak felírásával?) (3p)

4. Adja meg a 8-bites Hamming-kód esetén, az egyszeres (C_i) és kettős hibajelző bittel (DEB) kiegészített bináris bitmintázatot a következő bináris adatra: 10010110. Használjon páratlan paritást! A felírás során hogyan képezzük a paritásbit-csoportokat az adat- és kódbitek helyzetének figyelembe vételével? (3p)

3. Tervezze meg és rajzolja fel a 4-bites adatbusz Hamming kódos hibajavító áramkörét! (Milyen dekódert kell használni, hogyan kell a paritásbitcsoportokat képezni az adat és kódbitek felírásával.) (3p)

3. Tervezze meg és rajzolja fel a 11-bites adatbusz Hamming kódos hibajavító áramkörét! (Milyen dekódert kell használni, hogyan kell a paritásbitcsoportokat képezni az adat és kódbitek felírásával.) (3p)

>>> RANDOM KÓDOK A VÉGÉN (VHDL)

10. Adja meg az alábbi VHDL szubrutin DFG (adatfolyam) gráfját:

```
LED_PROC : process (Bus2IP_Clk) is
begin
  if Bus2IP_Clk'event and Bus2IP_Clk='1' then
      if Bus2IP_Reset='1' then
          LED_i<="0000";
      else
          if Bus2IP_WrCE(0)='1' then
              LED_i<=Bus2IP_Data(0 to 3);
          else
              LED_i<=LED_i;
          end if;
      end if;
  end if;
end process LED_PROC;
```

(2p)

10. Adja meg az alábbi VHDL szubrutin CFG (vezérlésfolyam) gráfját:

```
TemplAddr : process (clk,state)
begin
  if state = Waiting then
      TemplAddrReg <= 0;
  else
      if clk'event and clk = '1' then
          if ce = '1' then
              TemplAddrReg <= TemplAddrRegNext;
          else
              TemplAddrReg <= TemplAddrReg;
          end if;
      end if;
  end if;
end process;
```

10. Adja meg az alábbi VHDL szubrutin DFG-gráfját. Rajzolja is fel, milyen áramkört definiál az alábbi leírás?

```
architecture behavioral of MODULE is
begin
      p1: process(a, b, cin)
      variable vsum : std_logic_vector(6 downto 1);
      variable carry : std_logic;
      begin
            carry := cin;
            for i in 1 to 6 loop
                  vsum(i) := (a(i) xor b(i)) xor carry;
                  carry := (a(i) and b(i)) or (carry and (a(i) or b(i)));
            end loop;
            sum <= vsum;
            cout <= carry;
      end process p1;
end behavioral;
```

(3p)

10. Adja meg az alábbi VHDL szubrutin CFG-gráfját. Rajzolja is fel, milyen áramkört definiál az alábbi leírás?

```
architecture behavioral of MODULE is
begin
      p1: process(a, b, cin)
      variable vsum : std_logic_vector(3 downto 0);
      variable carry : std_logic;
      begin
            carry := cin;
            for i in 0 to 3 loop
                  vsum(i) := (a(i) xor b(i)) xor carry;
                  carry := (a(i) and b(i)) or (carry and (a(i) or b(i)));
            end loop;
            sum <= vsum;
            cout <= carry;
      end process p1;
end behavioral;
```

(3p)

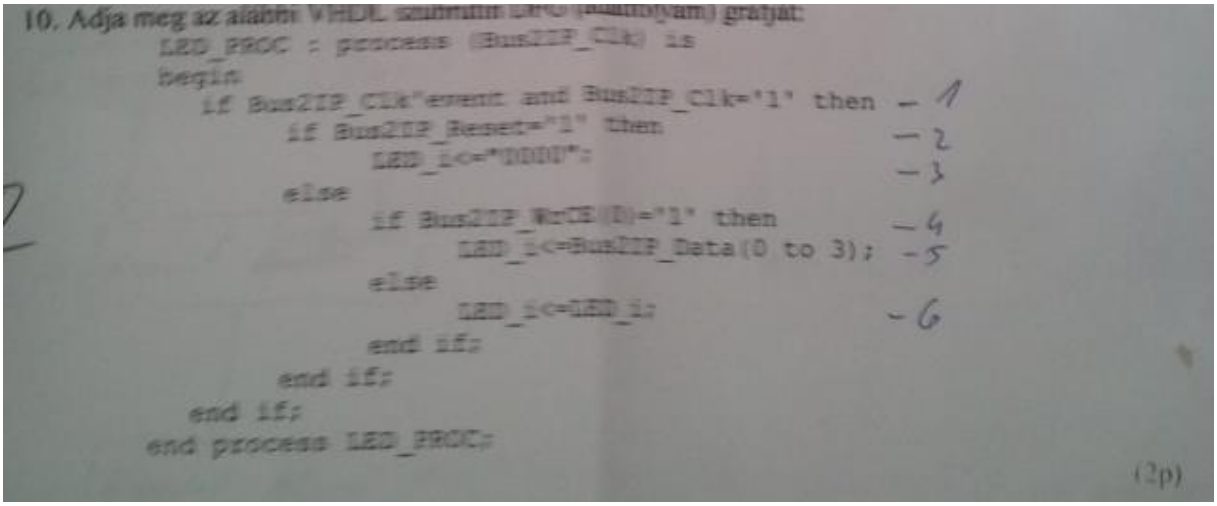
10. Adja meg az alábbi VHDL szubrutin DFG (adatfolyam) gráfját:

```
TemplAddr : process (clk,state)
begin
  if state = Waiting then
        TemplAddrReg <= 0;
  else
        if clk'event and clk = '1' then
           if ce = '1' then
                 TemplAddrReg <= TemplAddrRegNext;
           else
                 TemplAddrReg <= TemplAddrReg;
           end if;
        end if;
  end if;
end process;
```

(2p)

10. Adja meg az alábbi VHDL szubrutin DFG (adatfolyam) gráfját:

```
LED_PROC : process (Bus2IP_Clk) is
begin
 1 if Bus2IP_Clk'event and Bus2IP_Clk='1' then
 2      if Bus2IP_Reset='1' then
 3           LED_i<="0000";
 4      else
 5           if Bus2IP_WrCE(0)='1' then
 6                LED_i<=Bus2IP_Data(0 to 3);
 7           else
 8                LED_i<=LED_i;
 9           end if;
10      end if;
11 end if;
end process LED_PROC;
```



10. Adja meg az alábbi VHDL szubrutin CFG (vezérlésfolyam) gráfját:

```
TemplAddr : process (clk,state)
begin
  if state = Waiting then
       TemplAddrReg <= 0;
  else
       if clk'event and clk = '1' then
            if ce = '1' then
                 TemplAddrReg <= TemplAddrRegNext;
            else
                 TemplAddrReg <= TemplAddrReg;
            end if;
       end if;
  end if;
end process;
```

(2p)

11. Adja meg az alábbi VHDL szubrutin DFG-gráfját:

```
-- behavioral implementation of the LED_proc
architecture behavioral of LED is
begin
    LED_PROC: process (Bus2IP_Clk) is
          begin
             if Bus2IP_Clk'event and Bus2IP_Clk='1' then
                   if Bus2IP_Reset='1' then
                         LED_i<="0000";
                   else
                         if Bus2IP_WrCE(0)='1' then
                               LED_i<=Bus2IP_Data(0 to 3);
                         else
                               LED_i<=LED_i;
                         end if;
                   end if;
             end if;
          end process LED_PROC;
    end process LED_PROC;
end behavioral;
```

(2p)

11. Adja meg az alábbi VHDL szubrutin CFG (vezérlésfolyam) gráfját:

```
-- behavioral implementation of the N-bit adder
architecture behavioral of adderN is
begin
    p1: process(a, b, cin)
      variable vsum : std_logic_vector(N downto 1);
      variable carry : std_logic;
    begin
      carry := cin;
      for i in 1 to N loop
          vsum(i) := (a(i) xor b(i)) xor carry;
          carry := (a(i) and b(i)) or (carry and (a(i) or b(i)));
      end loop;
      sum <= vsum;
      cout <= carry;
    end process p1;
end behavioral;
```

(2p)

11. Adja meg az alábbi VHDL szubrutin DFG (adatfolyam) gráfját:

```
-- behavioral implementation of the N-bit adder
architecture behavioral of adderN is
begin
    p1: process(a, b, cin)
      variable vsum : std_logic_vector(N downto 1);
      variable carry : std_logic;
    begin
      carry := cin;
      for i in 1 to N loop
          vsum(i) := (a(i) xor b(i)) xor carry;
          carry := (a(i) and b(i)) or (carry and (a(i) or b(i)));
      end loop;
      sum <= vsum;
      cout <= carry;
    end process p1;
end behavioral;
```

(2p)

10. Adja meg az alábbi VHDL szubrutin CFG (vezérlésfolyam) gráfját:

```
LED_PROC : process ( Bus2IP_Clk) is
begin
  if Bus2IP_Clk'event and Bus2IP_Clk='1' then
       if Bus2IP_Reset='1' then
            LED_i<="0000";
       else
            if Bus2IP_WrCE(0)='1' then
                 LED_i<=Bus2IP_Data(0 to 3);
            else
                 LED_i<=LED_i;
            end if;
       end if;
  end if;
end process LED_PROC;
```

(2p)

>>> LEBEGŐPONTOS FOS

4. Adja meg a lebegőpontos összeadó blokkdiagrammját, az egyes blokkok funkciójával együtt! (2p)

5. Adja meg a lebegőpontos kivonó blokkdiagramját, az egyes blokkok funkciójával együtt! (2p)

5. Adja meg a lebegőpontos osztó blokkdiagramját, az egyes blokkok funkciójával együtt! (2p)

5. Adja meg a lebegőpontos szorzó blokkdiagramját, az egyes blokkok funkciójával együtt! (2p)

5. Adja meg a lebegőpontos kivonás esetén a műveletvégzés képletét, a kivonás blokkdiagramját, valamint az egyes blokkok funkcióját! (2p)

>>> FOLYAMATOS FOSMANÓ

6. Adja meg egy 16*16 bites iterativ szorzóalgoritmus folyamatábráját, és ennek áramköri megvalósitását, továbbá az algoritmus végrehajtási idejét! (3p)

7. Adja meg egy 16*16-bites iterativ, Shift&Add módszerű szorzóalgoritmus folyamatábráját, és ennek áramköri megvalósítását, továbbá az algoritmus végrehajtási időszükségletét! (3p)

7. Adja meg egy 16*16-bites iterativ, fordított sorrendű szorzóalgoritmus folyamatábráját, és ennek áramköri megvalósítását, továbbá az algoritmus végrehajtási időszükségletét! (3p)

7. Adja meg egy 16*16-bites iterativ, fordított sorrendű szorzóalgoritmus folyamatábráját, és ennek áramköri megvalósítását, továbbá az algoritmus végrehajtási időszükségletét! (3p)

7. Adja meg egy 16*16-bites hagyományos osztó-algoritmus folyamatábráját, és ennek áramköri megvalósítását, továbbá adja meg az osztás paramétereit és tulajdonságát! (3p)

6. Adja meg egy 16*16 bites iterativ szorzóalgoritmus folyamatábráját, és ennek áramköri megvalósitását, továbbá az algoritmus végrehajtási idejét! (3p)

6. Adja meg egy 16*16 bites iterativ szorzóalgoritmus folyamatábráját, és ennek áramköri megvalósitását, továbbá az algoritmus végrehajtási idejét! (3p)

6. Az iteratív osztási művelet egyik közvetlen, gyors módszere a következő:

$$Q = \frac{D_D \times f_0 \times f_1 \times f_2 \ldots}{D_S \times f_0 \times f_1 \times f_2 \ldots}$$

Mi az algoritmus végrehajtásának menete, illetve leállásának a feltétele? Rajzolja fel a működést biztosító áramkör kapcsolási rajzát! (2p)

>>> SZINTÉZIS

8. Definiálja az allokáció fogalmát (allocation) magas-szintű szintézis esetén! (1p)

8. Adja meg a magas-szintű szintézis (HLS) elméleti alaptételét! (1p)

8. Adja meg magas-szintű szintézis (HLS) alapját meghatározó sejtést! (2p)

8. Mit jelent a szintézis, illetve milyen főbb algoritmikus lépésekből áll a magaszintű szintézis. (2p)

8. Adja meg a magas-szintű szintézis (HLS) elméleti alaptételét! (1p)

8. Definiálja az allokáció fogalmát (allocation) magas-szintű szintézis esetén! (1p)

9. Definiálja a DFG-t magas-szintű szintézis esetén! (1p)

9. Definiálja a CFG-t magas-szintű szintézis esetén! (1p)

9. Definiálja az allokáció (allocation) és az ütemezés (scheduling) fogalmát magas-szintű szintézis esetén! (1p)

>>> MEMÓRIA FOS

9. Adott : memória hozzáférés ideje 20ns, regiszterből regiszterbe másolás 4 ns, kivonás 6 ns, Adja meg a ADD3 **X, Y, Z** RTL leírását és időszükségletét! (3p)

9. Adott : memória hozzáférés ideje 20ns, regiszterből regiszterbe másolás 4 ns, kivonás 6 ns, Adja meg a ADD2 ***X, *Y** RTL leírását és időszükségletét! (3p)

9. Adottak: memória hozzáférés ideje 20ns, regiszterből-regiszterbe másolás 4ns, aritmetikai művelet pedig 6ns. Adja meg a SUB3 ***X, Y, Z** RTL leírását és pontos időszükségletét! (3p)

9. Adottak: memória hozzáférés ideje 30ns, regiszterből-regiszterbe másolás 4ns, aritmetikai művelet pedig 5ns. Adja meg a DIV3 $*R_X, R_Y, R_Z$ RTL leírását és pontos időszükségletét! (3p)

9. Adott : memória hozzáférés ideje 10ns, regiszterből regiszterbe másolás 2 ns, kivonás 3 ns, Adja meg a SUB3 **X, RY, Z** RTL leírását és időszükségletét! (3p)

9. Adott : memória hozzáférés ideje 10ns, regiszterből regiszterbe másolás 2 ns, összeadás 3 ns, Adja meg a ADD3 **RX, Y, Z** utasítás RTL leírását és időszükségletét! (3p)

9. Adott : memória hozzáférés ideje 20ns, regiszterből regiszterbe másolás 4 ns, kivonás 6 ns, Adja meg a ADD3 **X, Y, Z** RTL leírását és időszükségletét! (3p)

9. Adott : memória hozzáférés ideje 20ns, regiszterből regiszterbe másolás 4 ns, kivonás 6 ns, Adja meg a ADD2 *X, *Y RTL leirását és időszükségletét! (3p)

10. Adott : memória hozzáférés ideje 10ns, regiszterből regiszterbe másolás 4ns, szorzás 6ns. Adja meg a MUL2 **A,*B** RTL leírását és pontos időszükségletét! (3p)

10. Adott: memória hozzáférés ideje 20ns, regiszterből regiszterbe másolás 3ns, osztás 5ns. Adja meg a DIV2 *X, Y RTL leírását és pontos időszükségletét! (3p)

9. Adott : memória hozzáférés ideje 10ns, regiszterből regiszterbe másolás 2 ns, kivonás 3 ns, Adja meg a SUB2 *X, *Y RTL leirását és időszükségletét! (3p)

>>> FA, RCA

7. Adja meg egy egybites Full Adder (FA) igazságtáblázatát, Karnough tábláit, kapuszintű kapcsolási rajzát, továbbá a 8-bites RCA (Riple Carry Adder) kapcsolási rajzát és számítási időszükségletét! (3p)

8. Adja meg az egybites Full Substractor (FS) igazságtáblázatát, Karnaugh tábláit, kapuszintű kapcsolási rajzát és számítási időszükségletét! (3p)

8. Adja meg egy 8-bites RCA (Ripple-Carry Adder) egység blokkdiagramját és számítási időszükségletét! Adja meg blokkjának belső felépítését, kimeneti függvényeit, és igazságtáblázatát! (3p)

7. Adja meg egy egybites Full Substractor (FS) igazságtáblázatát, Karnough tábláit, kapuszintű kapcsolási rajzát, továbbá a 6-bites RCA (Riple Carry Adder) kapcsolási rajzát és számítási időszükségletét! (3p)

7. Adja meg egy 16*16-bites iterativ, Shift&Add módszerű szorzóalgoritmus folyamatábráját, és ennek áramkőri megvalósítását, továbbá az algoritmus végrehajtási időszükségletét! (3p)

7. Adja meg egy egybites Full Substractor (FS) igazságtáblázatát, Karnough tábláit, kapuszintű kapcsolási rajzát, továbbá a 6-bites RCA (Riple Carry Adder) kapcsolási rajzát és számítási időszükségletét! (3p)

>>> LACA, időszükséglet

7. Adja meg egy 12 bites Riple Carry Adder összeadó blokkdiagramját, és adja meg a 12 bites összeadás időszükségletét, ha egy kapu késleltetése G. (3p)

7. Adja meg egy 4×4-bites direkt (közvetlen) szorzó rendszer blokkdiagramját Full Adder-ek felhasználásával (P=A×B). Adja meg egy szorzás sebességét ha egy kapu késleltetése G. (3p)

7. Adja meg egy 4×4 bites direkt (közvetlen) szorzó rendszer blokkdiagramját Carry Save Adder-ek felhasználásával (P=A×B). Adja meg egy szorzás sebességét ha egy kapu késleltetése G. (3p)

7. Adja meg egy 16 bites LACA (Look-ahead Carry Adder) összeadó blokkdiagramját, és adja meg a 16 bites összeadás időszükségletét (4 bites LACG generátort feltételezve). (3p)

6. Adja meg egy 23*23 bites szorzó rendszer blokkdiagrammját (7-3) és (3-2) sorcsökkentők egységek felhasználásával, ahol az utolsó szint egy FA. Adja meg a szorzás időszükségletét, ha egy kapu késleltetése G. (3p)

7. Adja meg egy 16 bites LACA (Look-ahead Carry Adder) összeadó blokkdiagramját, és adja meg a 16 bites összeadás időszükségletét (4 bites LACG generátort feltételezve). (3p)

>>> CÍMŰ SZÁMÍTÓ BLOKKDIAGRAM

5. Rajzolja fel a három-című számítógép blokkdiagrammját! Definiálja az egyes blokkok funkcióját is! (2p)

6. Adja meg a 16-bites zéró-című gép blokkdiagramját! Definiálja az egyes blokkok funkcióját is! (2p)

6. Rajzolja fel a három-című számítógép blokkdiagramját! Definiálja az egyes blokkok funkcióját is! (2p)

6. Rajzolja fel a regiszter nélküli három-című számítógép blokkdiagramját! Definiálja az egyes blokkok funkcióját is! (2p)

6. Rajzolja fel a regiszteres három-című számítógép blokkdiagramját! Definiálja az egyes blokkok funkcióját is! (2p)

5) Rajzolja fel a kétcímű számítógép blokkdiagrammját! Definiálja az egyes blokkok funkcióját is! (2p)

5. Rajzolja fel a egycímű számítógép blokkdiagrammját! Definiálja az egyes blokkok funkcióját is! (2p)

5. Rajzolja fel a zéró-című számítógép blokkdiagrammját! Definiálja az egyes blokkok funkcióját is! (2p)

5. Rajzolja fel a kétcímű számítógép blokkdiagrammját! Definiálja az egyes blokkok funkcióját is! (2p)

5. Rajzolja fel a egycímű számítógép blokkdiagrammját! Definiálja az egyes blokkok funkcióját is! (2p)

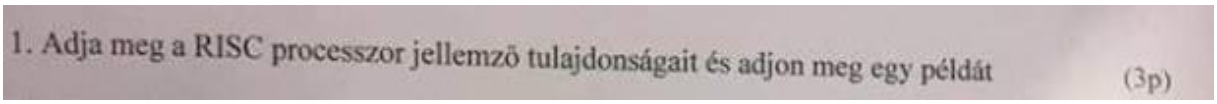
>>> DIREKT FOS

6. Adja meg egy direkt osztó blokkdiagramját és adja meg a végrehajtás időszükségletét, ha az osztó 6 bites és az osztandó12 bites. (3p)

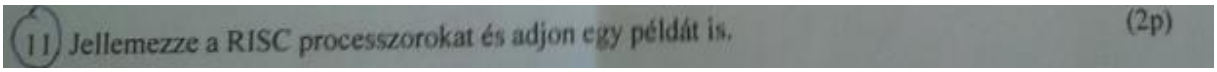
6. A direkt osztó blockdiagramját, határozza meg a végrahajtás időszükségleltét és a leállás feltételét. (2p)

>>> ELMÉLET

>>> RISC CISC


1. Adja meg a RISC processzor jellemző tulajdonságait és adjon meg egy példát (3p)

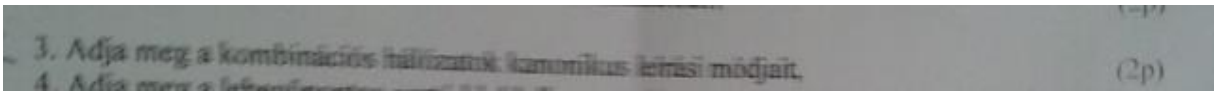
3. Adja meg a RISC architektúrák jellemzőit és a soroljon fel néhány tipikus RISC processzort (3p)


11) Jellemezze a RISC processzorokat és adjon egy példát is. (2p)

1. CISC processzorok jellemzői, tulajdonságai (előnyök, hátrányok)! Soroljon fel néhány példát CISC processzorra! (2p)

1. RISC processzorok jellemzői, tulajdonságai (előnyök, hátrányok)! Soroljon fel néhány példát RISC processzorra! (2p)

>>> HÁLÓZATOK

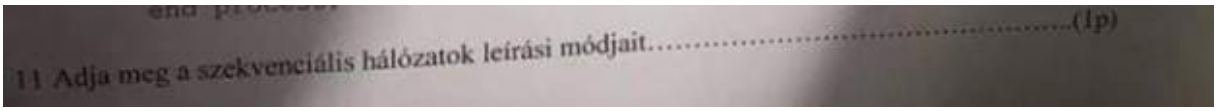

3. Adja meg a kombinációs hálózatok kanonikus leírási módjait. (2p)

11. Sorolja fel a kombinációs hálózatok leírási módjait.

1. Adja meg a kombinációs hálózatok megadásának módjait,
Milyen kombinációs hálózat opitmalizásáokat ismer és mik ennek a korlátai (3p)

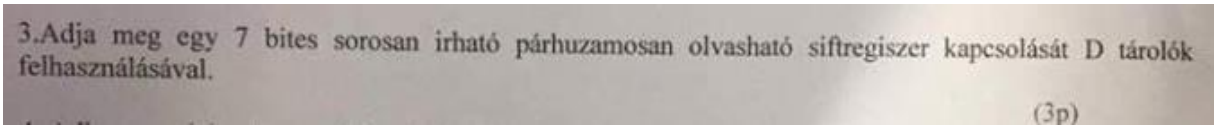
1. 1. Adja meg a szekvenciális hálózatok megadásának módjait,
Milyen fajta hazárd jelenségeket ismer és hogyan lehet ezeket megszüntetni? (3p)

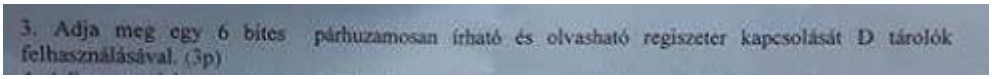
3. Adja meg a szekvenciális hálózatok definicióját, sorolja fel a megadásának módjait. Definiálja hazárd fogalmát. (3p)


11 Adja meg a szekvenciális hálózatok leírási módjait..... (1p)


11. Sorolja fel a kombinációs hálózatok leírási módjait. (1p)

>>>REGISZTERES D TÁROLÓS FOS TAVALYRÓL


3.Adja meg egy 7 bites sorosan irható párhuzamosan olvasható siftregiszer kapcsolását D tárolók felhasználásával. (3p)


3. Adja meg egy 6 bites párhuzamosan írható és olvasható regiszeter kapcsolását D tárolók felhasználásával. (3p)

>>> HARVARD, NEUMANN

2. Mi a Neumann architektúra definíciója? (1p)

2. Mi a Harvard architektúra definíciója? (1p)

>>> LOGIKAI FÜGGVÉNYEK

1. Definiálja a logikai függvények diszjunktív normálformáját! (képzési szabályok, képlet) (1p)

1. Definiálja a logikai függvények konjunktív normálformáját! (képzési szabályok, képlet) (1p)

>>> ALU

10. Mi az ALU? Milyen a felépítése (pl.: 4-bites ALU)? Milyen függvényekkel működik (működését leíró függvénytáblázat)? (3p)

1. Rajzolja fel egy 4 bites ALU blokk diagramját és ismertesse a funkcionális egységeket (2p)

>>> REGISZTER ABLAKOZÁS

1. Magyarázza meg a regiszter ablakozás működését és alkalmazását (3p)

1. Magyarázza meg a regiszter ablakozás működését és alkalmazását (3p)

>>> EGYÉB ELMÉLET

1. Rajzolja fel egy 9 bites buszhoz a paritásbitet generáló kapcsolást. (2p)

3. Adja meg a közvetlen gyors osztóműködési elvét és ennek architektúrális megvalósítását. Hogyan lehet a számítási pontosságot figyelembe venni? (3p)

4. Adott a következő komplex matematikai kifejezés: $V = [S*P - Q*(R / X)] - T$.
Zéró-című gép használatával értékeljük ki a kifejezést, előről – hátra, illetve hátulról – előre felé haladva. Melyik módszer lesz az optimálisabb (ha a hagyományos zéró-című adatkezelési-, illetve aritmetikai műveleteket használhatjuk)? (3p)