20…..év …hó …nap

NÉV:.....neptun kód:..…… Kurzus:……

A feladatokat önállóan, meg nem engedett segédeszközök használata nélkül oldottam meg:

Olvasható aláírás:.....……

Kedves Kolléga! ***A kitöltést a dátum, név és aláírás rovatokkal kezdje!*** Az alábbi kérdésekre a válaszokat - ahol lehet - mindig a feladatlapon oldja meg! A feladatok megoldása során a részletes kidolgozást nagyfeladatonként külön papíron végezze, (egyértelműen jelölje, hogy melyik lap melyik feladathoz tartozik, a papírra már a kezdetkor írja rá a nevét és neptun kódját) és ezeket a papírokat is adja be a dolgozatával! A kérdésekre a táblázatok vagy a pontozott vonalak értelemszerű kitöltésével válaszoljon, hacsak külön másként nem kérjük. ***Mindenütt a legegyszerűbb megoldás éri a legtöbb pontot.*** Jó munkát!

E:
F1:
F2:
F3:
$\sum$:

Ellenőrző kérdések (25p)

E1. Konvertálja az alábbi számokat a kért formátumra! (2p)

kiinduló formátum	konvertálandó szám	kért formátum	konvertált szám
decimális	20	6 bites előjel nélküli bináris	**010100**
8 bites 2-es komplemens	11111101	decimális	**-3**

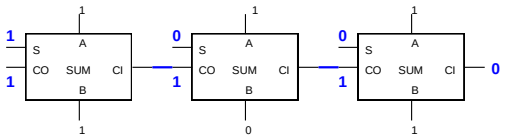
E2. Mi az előnye a 2-es komplemens számábrázolásnak? Húzza alá a legkedvezőbb választ! (1p)

a. A előjeles számok viszonyának (kisebb/nagyobb) eldöntésénél előnyös.
b. Előjeles számok összeadásánál előnyös.
c. Előjeles számos 2 hatvánnyal való szorzásánál előnyös.

E3. Egyszerűsítse az alábbi Boole algebai kifejezést, Boole algebrai átalakításokkal! A részeredményeket is írja le! (1p)

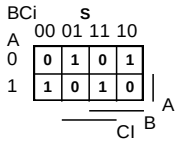
A + /A.B = ...**(A+/A)(A + B) = A + B**.....

E4. ***Kaszkádosítsa*** az alábbi, 1 bites összeadókat a rajz kiegészítésével úgy, hogy azok 2 db 3 bites számot helyesen adjanak össze. (CI értékét ennek megfelelően írja be). ***Adja meg a kimeneteiken megjelenő logikai értékeket is a beírt adatok esetére!*** (2p)



E5. Megadtuk a fenti összedadó S kimenetének igazságtábláját Karnaugh táblás alakban. (3p)

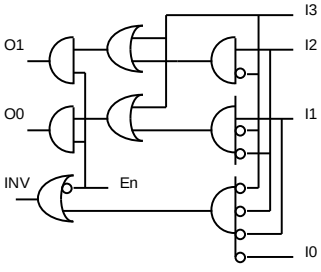
a. Írja le a függvényt megvalósító kombinációs hálózatot ***Verilog nyelven***, egyszerűsítetlen ***SOP*** alakban!



.. assign S = ~A&~B&CI | ~A&B&~CI | A&~B&~CI | A&B&CI; .

b. Lehet-e egyszerűsíteni a függvényt, ha a megvalósításhoz csak AND, OR, INV logikai operátorokat használhatunk? Húzza alá a megfelelő választ! igen nem

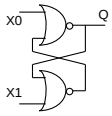
E6. Az alábbi kapcsolási rajz egy ismert funkcionális elem belső felépítését mutatja. Adja meg a nevét, majd folytassa az ***O kimenetének*** Verilog leírását! (3p)



neve:.....**prioritás enkóder**.....

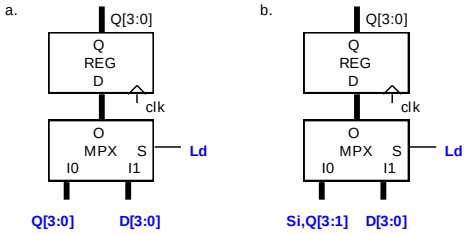
```
always (*)
if(En)
      casex (I)
            4'b0001: O = 2'b00;
                  4'b001x: O = 2'b01;
                  4'b01xx: O = 2'b10;
                  4'b1xxx: O = 2'b11;
                  default:   O = 2'b00;
            endcase
```

E7. **a.** Milyen ismert áramkör látható az alábbi kapcsolási rajzon? (1p) .....**SR flip-flop**.....
b. Adja meg, milyen válaszsorozatot ad az alábbi bemeneti sorozatra az áramkör! A megoldást elkezdtük, folytassa! (2p)



X0	0	0	1	0
X1	1	0	0	0
Q	1	1	**0**	**0**

E8. Egészítse ki az alábbi rajzot, az kérdéseknek megfelelően!



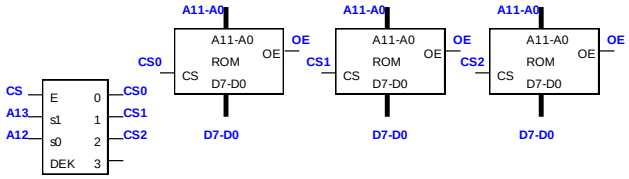
a. Egy engedélyezhető regisztert valósítson meg! (Ld = 1 tölt) (1p)
b. Egy jobbra shiftelő tölthető shiftregisztert valósítson meg! (Ld = 1 tölt, egyébként shiftel) (1p)

E9. Milyen funkcionális elemet adtunk meg az alábbi Verilog leírással? (1p)

```
wire [3:0] in;
assign ou = |(({3'b000,en} << sel) & in);
```

funkcionális elem neve: .**multiplexer (4 bit, en)**.

E10. **a.** Készítsen az alábbi ROM-okból 3-szoros kapacitású egységet! (Egészítse kia a rajzot!) (1p)
Az elkészítendő egység jelei: A13-A0, D7-D0, OE, CS
b. Mekkora egy ROM modul kapacitása? (1p) ……**4**…kibyte

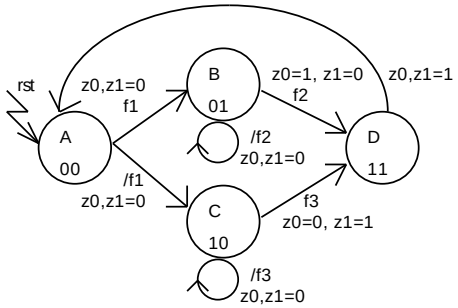


E11. Mely állítások igazak és melyek hamisak? Jelölje **+**-al az igaz, **-**-al a hamis állításokat! (5p)

1.	Az always blokkban wire és reg típusú változóknak lehet értéket adni.	**-**
2.	A bináris számlálók bármely adat kimenetén a magas és alacsony szint névlegesen ugyanannyi ideig tart.	**+**
3.	Az EXNOR függvény páratlan számú 1-es esetén ad 1-et.	**-**
4.	Minden szinkron sorrendi hálózat kimenete csak az órajel aktív élének hatására változhat meg.	**-**
5.	Ha két ugyanazon órajelre működő regiszter között egy kombinációs hálózat van, akkor az ideális órajel periódusidejére (Tclk) igaz, hogy Tclk > tcqmax + tpmax + tsu	**+**

Feladatok:

F1. (11p) Adott egy FSM az alábbi állapotgráffal. Állapotok: A,B,C,D,E. Bemenetek: rst, Start, f1, f2, f3. Kimenetek: z0,z1. Az állapotkódolást megadtuk. Készítse le a Verilog leírását külön az ***állapotregiszter***, külön a ***next_state logika*** és külön a ***kimeneti logika*** megadásával! A next_state logikánál az always-t a kimeneti logikánál pedig assign-t használja! A leírást elkezdtük, fejezze be!



Milyen modell szerint működik? Húzza alá a megfelelőt! (1p)

Mealy Moore

//Konstans és változó deklarációk, minhárom egységhez tartozó deklarációkat ide írja! (2p):

```
localparam [1:0] A = 2'b00,
B = 2'b01,
C = 2'b10,
D = 2'b11;
reg [1:0] state;
reg [1:0] next_state;
```

//Állapotregiszter (2p):

```
always@( posedge clk)
  if(rst) state <= A;
  else state <= next_state;
```

//Next_state logika (4p):

```
always@( *  )
  case (state)
          A: if(f1) next_state <= B;
             else next_state <= C;
          B: if(f2) next_state <= D;
             else next_state <= B;
          C: if(f3) next_state <= D;
             else next_state <= C;
          D: next_state <= A;
      default  next_state <= A;
  endcase
```

//Kimeneti logika (2p):

```
assign z0 = f2 & (state == B) | (state == D);
assign z1 = f3 & (state == C) | (state == D);
```

F2. Decimális számláló modul tervezés, kaszkádosítás, időmultiplexált kijelzés. (12p)

a. Adja meg egy decimális számláló (dec_cnt) Verilog modulját. A számláló tölthető (***ld***, ***d[3:0]***) kétirányú (***dir*** = 1 fel), engedélyezhető (***en***), kimenetei: ***q[3:0]*** és ***tc***. (6p)

```
module dec_cnt(input clk, input ld, input dir, input en, input [3:0] d, output [3:0] q, output tc)
assign q0 = q == 4'd0;
assign q9 = q == 4'd9;
assign tc = en & (dir ? q9  : q0);

always @(posedge clk)
begin
  if(ld) q <= d;
    else if(en)
            if(dir)
               if(q9) q <= 0;
               else q <= q+1;
            else
               if(q0) q <= 9;
               else  q <= q-1;
end
```

b. Kaszkádosítson a fenti dec_cnt modulból 2 példányt (***dec_cntH***, ***dec_cntL***. Utóbbi adja az LSB-t)! A kaszkádosítást annak Verilog leírásával végezze! (4p)

```
dec_cnt dec_cntL(.clk(clk), .ld(ld), .dir(dir), .en(en), .q(qL), .tc(tcL));
dec_cnt dec_cntH(.clk(clk), .ld(ld), .dir(dir), .en(tcL), .q(qH), .tc(tcH));
```

c. ***Egészítse ki a b-pontbeli leírást (alább vagy külön lapon), hogy a decimális számlálók kimenete egy 2 digites időmultiplexált kijelzőn jelenjen meg!*** A 7 szegmenses kijelző digitjeinek engedélyezését a ***dig0, dig1*** jelek végzik, az adat bemenete ***dat[3:0]***. A 7 szegmenses kijelző BCD kódú adatokat vár (a BCD_7 szegmenses dekóder bele van építve). Az időmultiplexálás vezérléséhez rendelkezésére áll egy 2 bites gyűrűs számláló modul: ***ring(input rst, input clk, output [1:0] Ou)***. (3p)

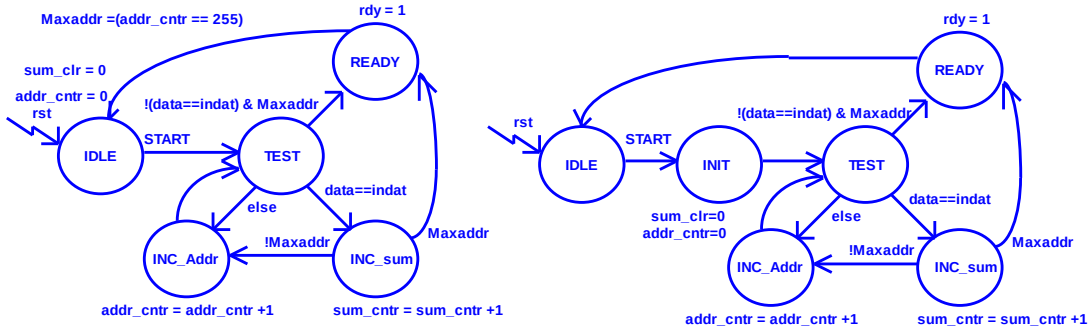
```
wire dig1,dig0;
wire [3:0] dat;
ring ring(.rst(rst), .clk(clk), .Ou({dig1,dig0}));  // gyűrűs számláló példányosítása

always @(*)                          //adatok multiplexálása
 case ({dig1,dig0})
          2'b01:   dat <= qL;
          2'b10:    dat <= qH;
          default:
 endcase
```

F3. ***Egy 256x8-as ROM-ban meg kell számolni a külső din bemeneten lévő adattal egyező adott adatok számát!*** A számolást a ***start*** bemenetre adott impulzus indítja. A feladat elkészültét az ***rdy*** kimenetén egy pulzussal jelzi, ekkor ***data_count*** kimenetén az adatok száma található. (12p)

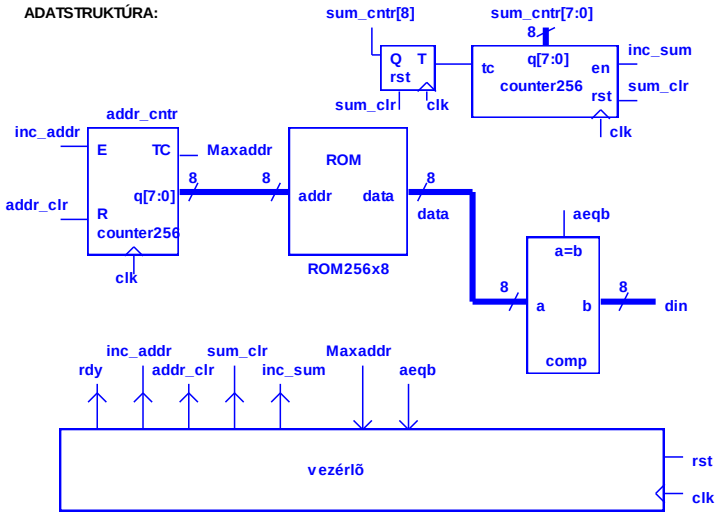
a. Röviden fogalmazza meg az algoritmust! (2p)
A start előtt az adat számlálót és cím számlálót 0-ázzuk. A start után minden címen tesztelni kell, hogy a ROM adat egyezik-e a bemenetivel. Egyezés esetén inkrementálni kell az adat számlálót. Tesztelés után növelni kell a címszámlálót. A legnagyobb memória címen való tesztelés után rdy jelzés, majd vissza az elejére.

b. Rajzolja le az adatstruktúrát vezérlő HLSM állapotdiagramját! (4p)
Az baloldali verzióban az eredmény csak az rdy alatt érvényes, a baloldaliban az új START-ig.



c. Rajzolja le a feladat adatstruktúrájának blokkvázlatát. A feladat elkészítéséhez az alábbi előre megadott modulokat használja fel (akár több példányban is). (6p)

ROM256x8(input [7:0] addr, output [7:0] data); //ROM
counter256(input clk, input rst, input en, output [7:0] q, output tc); //szinkron rst-ű felfele számláló
comp(input [7:0] a, input [7:0] b, output eq); // egyenlőség komparátor



A számlálót 9-bitesre kell kiegészíteni, mert ha minden adat egyezik, akkor a sum_cntr = 256 (0x100).

Maximális pontszám: 60 pont
Rendelkezésre álló idő: 100 perc