Ellenőrző kérdések (28p)

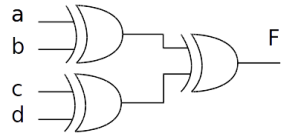
E1. Konvertálja az alábbi számokat a kért formátumra! (3p)

kiinduló formátum	**konvertálandó szám**	**kért formátum**	**konvertált szám**
4 bites 2-es komplemens	1101	8 bites 2-es komplemens	
16 bites NBCD	0001100101010110	decimális	
8 bites ofszet kód	00101000	8 bites 2-es komplemens	

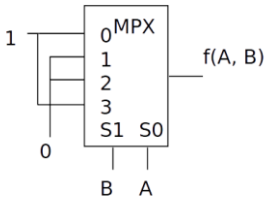
E2. ***Egyszerűsítse az alábbi Boole algebrai kifejezéseket,*** Boole algebrai átalakításokkal! (2p)

/(A + /B) + AB = …… A + /AB = ……

E3. Milyen speciális tulajdonságú az alábbi áramkör? (Segítségül egy szó: *paritás*). Adja meg Verilog nyelven a leírását, a lehető legegyszerűbben! (2p)



Tulajdonság: ……
assign F = ……

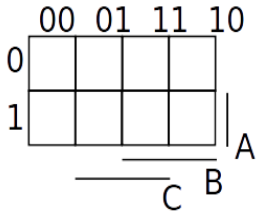


E4. Adja meg azt a függvényt SOP alakban, amit az alábbi multiplexer megvalósít! (2p)

f(A, B) = ……

E5. Egy olyan függvényt kell megvalósítani, ami az ABC 3 bites számok közül a 0, 1, 6, 7 esetén jelez.

a. Töltse ki a függvény Karnaugh tábláját! (2p)



b. Adja meg a függvény legegyszerűbb SOP alakját!

……

c. Írja le a függvényt megvalósító kombinációs hálózatot ***Verilog nyelven, a megadott relációs operátorok (<, >) használatával!*** (1p)

……

E6. Adott egy 16-os modulusú lefele számláló, down_cnt16(input clk, rst, ld, input [3:0], d, output tc, output reg [3:0] q). Tölthető (ld), végérték jelzéssel rendelkezik (tc).

a. Készítsen egy példányából 9-es modulusú számlálót (**8**,7,6,5,4,3,2,1,0,**8**, …)! Adja meg a szükséges bekötéseket assign-al. (3p)

assign ld = …… d = ……

b. Módosítsa úgy, hogy a kialakított 9-es modulusú számláló egy ldd jellel tölthető maradjon! (2p)

assign ld = …… d = ……

E7. A ***2 regisztercímes processzor architektúra*** esetén hogyan néz ki ki egy logikai AND utasítás mnemonikja? A megadását elkezdtük, egészítse ki a regiszterek megadásával! (1p)

and ……

E8. Milyen funkcionális elemet adtunk meg az alábbi Verilog leírással? (2p)

```
wire e, ao;
wire [3:0], Ia, Ib;
assign ou = (Ia == Ib) & e;
```

funkcionális elem neve: ……

E9. Rajzolja le hogyan kell összekötni egy SPI mastert egy SPI slave-vel! Rajzolja be a jelek irányát is nyíllal! (2p)

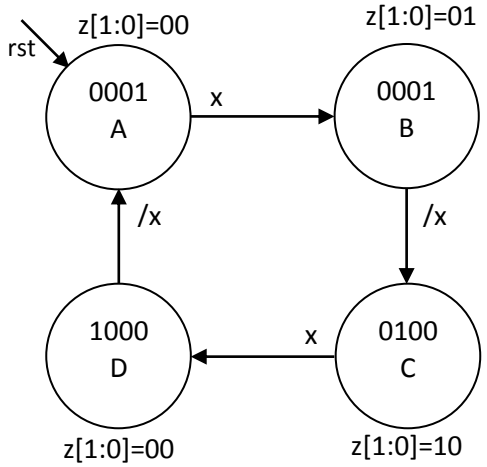
E10. Mely állítások igazak és melyek hamisak? Jelölje +-al az igaz, --al a hamis állításokat! (5p)

1.	Egy 8 cím bemenetű ROM-mal tetszőleges 8 változós függvény megvalósítható.	
2.	Egy csak D és clk bemenetekkel rendelkező D flip-flop kimenete csak az órajel élénél (kicsit utána) változhat meg.	
3.	NOR kaput inverterként nem lehet használni.	
4.	Az interrupt hatására biztosan megváltozik a STACK tartalma, mielőtt az IT rutin első utasítása elkezd végrehajtódni.	
5.	A shiftelő utasítások megváltoztatják a flag-ek értékét.	

Feladatok:

F1. (16p) Adott egy FSM az alábbi kódolt állapotgráfjával. (A nem jelölt x-re marad.)

a. Az A állapotból kiindulva milyen kimenetet ad, a következő bemeneti sorozatra? (2p)



X	0	1	1
Z[1:0]			

b. Milyen ismert állapotkódolást alkalmaztak? (1p)

……

b. Készítse el az FSM modul deklarációját! (2p)

module FSM (……...)

d. ***Készítse el a Verilog leírását*** külön az ***állapotregiszter***, külön a ***next_state logika*** és külön a ***kimeneti logika*** megadásával! A leírást elkezdtük, fejezze be!

Verilog leírás.

// **A konstans és változó deklarációk** (2p)

```
localparam [3:0] A = 4'b0001, B = ……, C = ……, D = ……;
reg [3:0] s;
reg [3:0] next_state;

// Állapotregiszter (2p)
always @ (                )
   if (rst) s <=
   else     s <=
```

// **Next_state logika** (5p)
always @ ( )
  case (s)
    A: if(~x)
       else

    B: if(~x)
       else

    C: if(~x)
       else

    D: if(~x)
       else
    default:
// **Kimeneti logika** (2p)
assign z =

F2. (20p) A fizika tantárgyhoz a gravitációs gyorsulás mérését megkönnyítő segédeszközt kell tervezni. A működés: A START pergésmentes nyomógomb lenyomásakor egy vasgolyót elenged egy elektromágnes az EJT impulzus hatására. Ekkor elindul egy 9 bites számláló, mely 1/1000 sec-onként lép. A golyó 0,5m megtételekor elhalad egy e1 érzékelő előtt, mely 1 órajel hosszú impulzussal jelzi az áthaladást. Ekkor az idő számláló értékét át kell írni a reg1 regiszterbe, a reg1_ld vezérlő jellel. 1m-nél szintén vagy egy hasonló érzékelő (e2). Ennek jelzéskor az idő számláló értékét át kell írni egy reg2 regiszterbe a reg2_ld vezérlő jellel és az RDY jelet ki kell adni, Ez egy LED-en jelzi a mérés végét. A mért adatok alapján kiszámítható a g értéke. Rendelkezésre áll egy előosztóról jövő **sig_mili** jel, mely 1/1000 másodpercenként 1 órajelnyi ideig aktív (ezt nem kell előállítani). Az áramkör órajele egy néhány MHz-es **clk** jel (pontos értéke itt lényegtelen). A feladat megoldását részfeladatokra bontottuk.

a. Tervezzen meg egy 9 bites bináris ***felfelé*** számlálót! (Adja meg a Verilog leírását!) Legyen szinkron törölhető (R), engedélyezhető (E)! A leírást alább elkezdtük, fejezze be ! (4p)
b. Tervezze meg a tölthető regisztert! A regiszter az ld jel hatására szinkron töltse be a din adatot. (2p)
c. Példányosítsa a számláló modult úgy, hogy a számlálás üzemmódban 1/1000 másodpercenként lépjen (***sig_milli***)!
Az rst és a /MEAS jel törölje, s ha nincs törölve, akkor a ***sig_milli*** jel léptesse! (4p)

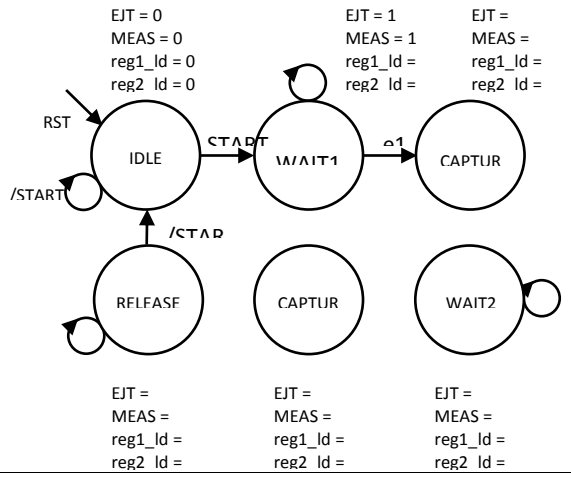
cnt512 means_cnt(.clk(…….), R(……..), E(…….), q(…….));

a. 9 bites számláló Verilog leírása:
module cnt512 (
  input clk, R, E, output reg [8:0] q
)

always @ (posedge clk)
  if(R) …….
  else
    if(E) …….
endmodule

b. Tölthető regiszter Verilog leírása:

d. Tervezze meg a vezérlő Moore állapotgráfját és rajzolja le! A rajzot elkezdtük, fejezze be! A kimeneteket minden állapotban adja meg! (10p)



F3. (21p) Egy MiniRISC buszra (A [7:0], Dou [7:0], RD, WR, IRQ, clk, rst) illesztett soros port bemenetre a parancsregiszter RXEN bitjével engedélyezhető. A státuszregiszterének RXRDY bitjében jelzi ha adat érkezett. Ekkor az adatregiszterekből kiolvasható az adat. A státuszregiszter RXRDY bitje az adat olvasása után automatikusan törlődik. A periféria báziscíme 0xE0. A programozói felülete a következő:

funkció	**cím**	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**	**olvasható/írható**
Parancs regiszter	Báziscím + 0	X	X	X	X	X	X	X	EN	W
Státusz regiszter	Báziscím + 1	X	X	X	X	X	X	RXRDY	EN	R
Adat regiszter	Báziscím + 2	D7	D6	D5	D4	D3	D2	D1	D0	R/W

a. Tervezze meg Verilog nyelven a periféria parancs regiszterbe írást engedélyező parancs_wr, a státuszregiszter és az adatregiszter olvasását engedélyező status_rw és data_rd jeleit, továbbá az adatregiszter írását engedélyező data_wr jelet! (7p)

b. A soros portot ASCII stringek fogadására használják, melyek hossza maximum 50 karakter. Írjon olyan program részletet, mely engedélyezi a perifériát, majd a státuszát figyelve beolvassa belőle a karaktereket, mindaddig, amíg nem 0 érkezik. A stringet leteszi az in_string memória kezdőcímtől kezdve. (Tételezze fel, hogy biztosan nem jön 50 karakternél hosszabb string.) (7p)

```
DEF par 0xE0
DEF stat ……...
DEF adat ……..
DEF EN ……....
DEF RXRDY……

      DATA
in_string:
      DB 00
      …….
start:
      mov r0, #…….
      mov par, ……..          ; periféria engedélyezése
      mov r2, ……             ; adat pointer
stat_loop:
      mov r0, ……
      tst …….
      …….                    ; várakozás adatra
      ……..                   ; karakter beolvasás
      ……..                   ; karakter a memóriába
      cmp r0, …….
      …….                    ; vége, ha 0x00 jött
      …….                    ; adat pointer iner.
      …….                    ; vissza a hurokba
str_end:
```

c. Írjon szubrutint, mely meghatározza az r1 regiszterében megkapott címen lévő 0-val végződő string hosszát és azt visszaadja az r0 regiszterben. (Tételezze fel, hogy a string soha nem hosszabb 50 karakternél.) (7p)

```
str_length:
    mov r0, …….          ; hossz kezdőérték
loop:
    mov r3, …….          ; karakter beolvasás
    cmp ……..             ; végjelzés figyelése
    …….                  ; kiugrunk, ha vége
    …….                  ; pointer inkrementálás
    …….                  ; hossz növelése
    …….                  ; vissza a hurokba
end_length:
    …….                  ; visszatérés szubrutinból
```

Maximális pontszám: 75 pontszám
Rendelkezésre álló idő: 100 perc