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:	**28**
F1:	**16**
F2:	**16**
F3:	**15**
$\sum$:	**75**

Ellenőrző kérdések (28p)

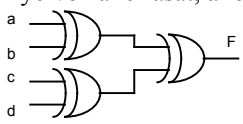
E1. Konvertálja az alábbi számokat a kért formátumra! (3p)

kiinduló formátum	konvertálandó szám	kért formátum	konvertált szám
4 bites 2-es komplemens	1101	8 bites 2-es komplemens	**11111101**
16 bites NBCD	0001100101010110	decimális	**1956**
8 bites ofszet kód	00101000	8 bits 2-es komplemens	**10101000**

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

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

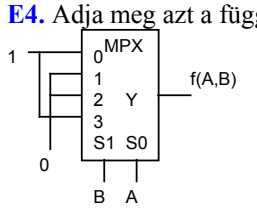
E3. Milyen speciális tulajdonságú az alábbi áramkör? (Segítségül egy szó: paritás). Adja meg Verilog nyelven a leírását, a lehető legegyszerűbben! (2p)



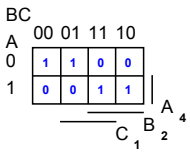
Tulajdonság ..... **páratlan paritást jelez**.....

assign F = ... **a ^b ^ c ^d**.....

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



f(A,B) = ... **AB + /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! (1p)

..... **/A/B + AB**.....

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

..... **assign f = ({A,B,C}<2) | ({A,B,C}>5)**.....

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 =.. **tc**..... d =....**4'h8**.....

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

assign ld =..... **tc | ldd**..... d =..... **ldd ? d_in : 4'h8**.....

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

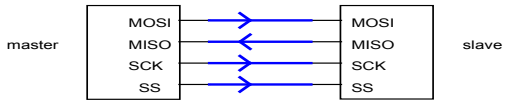
and .. **r0, r1**.....

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

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

funkcionális elem neve: **egyenlőség komparátor (engedélyezhető)**..

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



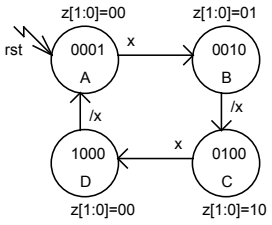
E10. Mely állítások igazak és melyek hamisak? Jelölje +-al az igaz, --al a hamis állításokat! (5p)

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 megváltoztathatják a flag-ek értékét.	+

Feladatok:

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

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



X	0	1	0
Z[1:0]	00	**01**	**10**

b. Milyen ismert állapotkódolást alkalmaztak? (1p)

..... **bit/state, (one hot encoding, n-ből 1)**...

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

module FSM (**input clk, rst, x, otput [1:0] z**)

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 =..**00010**..... , C = ... **0100**.... D = ... **1000**.....;

reg [3:0] s;

reg [3:0] next_state;

//Állapotregiszter (2p):

always@(**posedge clk**)

  if (rst) s <= **A;**

  else   s <= **next_state;**

//Next_state logika (5p):

```
always@(*)
  case (s)
        A: if(~x)  next_state <= A;
           else    next_state <= B;

        B: if(~x)  next_state <= C;
           else    next_state <= B;

        C: if(~x)  next_state <= C;
           else    next_state <= D;

        D: if(~x)  next_state <= A;
           else    next_state <= D;

        default:   next_state <= A;
```

//Kimeneti logika (2p):
assign z = **{2{(s ==A)}} & 2'b00 | {2{(s ==B)}} & 2'b01 | {2{(s ==C)}} &2'b10 | {2{(s ==D)}} & 2'b00;**
legegyszerűbb: assign z = **s[2:1];**

F2. 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 idő 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 van egy hasonló érzékelő (e2). Ennek jelzésekor 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 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. (16p)

a. Tervezzen meg egy 9 bites bináris ***felfele*** 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 din adatot. (2p)
c. Példányosítsa azámláló modult úgy, hogy számlálás üzemmódban 1/1000 másodpercenként lépjen (***sig_mili***)! Az rst és a /MEAS jel törölje, s ha nincs törölve, akkor a ***sig_mili*** jel léptesse ! (4p)

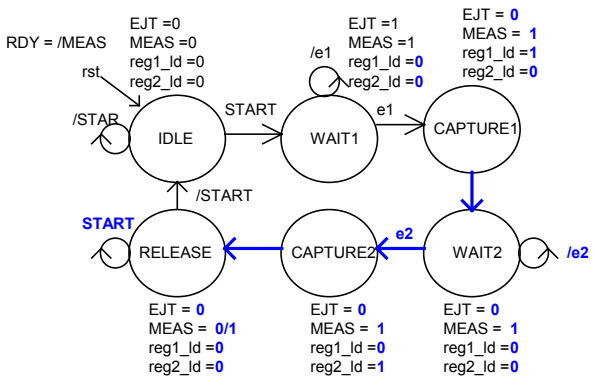
cnt512 meas_cnt(.clk(**clk**), .R(**rst | ~meas**), .E(***sig_mili***), .q(**q**));

a. //512-es számláló Verilog leírása:
module cnt512 (
input clk, R, E, output reg [8:0] q
)

always @(posedge clk)
if (R) **.q <= 9'h0;**.....
else
    if (E) ... **q <= q +9'h1;**.....
endmodule

b. Tölthető regiszter Verilog leírása:
module reg9 (input clk,ld, input [8:0]d, output reg [8:0] q)
always @(posedge clk)
  if (ld) q <= d;

d. Tervezze meg a vezérlő Moore állapotgráfját és rajzolja le! A rajzot elkezdtük, fejezze be! A kimeneteket minden állapotban adja meg! (6p)



F3. (15p) Egy MiniRISC buszra (A[7:0], Din[7:0], Dou[7:0], RD, WR, IRQ, clk, rst) illesztett soros port bemenete a parancsregiszter RXEN bitjével engedélyezhető. A státusregiszterének RXRDY bitjében jelzi, ha adat érkezett. Ekkor az adatregiszterből kiolvasható az adat. A státusregiszter 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
Státus 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 státus regiszter és az adatregiszter olvasását engedélyező status_rd és data_rd jeleit, továbbá az adatregiszter írását engedélyező data_wr jelet! (5p)

```
parameter base_addr = 8'he0;
        assign psel = (base_addr >> 2) == A[7:2];
        assign parancs_wr = psel & ({A[1],A[0]} == 2'b00) & WR;
        assign status_rd = psel & ({A[1],A[0]} == 2'b01) & RD;
        assign adat_rd = psel & ({A[1],A[0]} == 2'b10) & RD;
        assign adat_wr = psel & ({A[1],A[0]} == 2'b10) & WR;
```

b. A soros portot ASCII stringek fogadására használják, melyek hossza maximum 50 karakter. Írjon olyan program részletet, mely engedélyezi a perifériát, majd a státusát figyelve beolvassa belőle a karakterket, mindaddig, amíg 0 nem 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   0xe1
DEF adat   0xe2
DEF EN     0x01
DEF RXRDY  0x02

        DATA
in_string:
        DB 00

        CODE
start:
        mov r0, #EN
        mov par, r0             ;periféria engedélyezés
        mov r2, #in_string      ;adat pointer
stat_loop:
        mov r0, stat
        tst   r0, #RXRDY
        jz  stat_loop           ;karaktere várakozás
        mov r0, adat            ;karakter beolvasás
        mov (r2), r0            ;karakter a memóriába
        cmp r0,#0x00
        jz str_end              ;vége, ha 0x00 jött
        add r2, #1              ;adat pointer incr.
        jmp stat_loop           ;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, #0x00..         ;hossz kezdőérték
loop:   mov r3, (r1).....       ;karakter beolvasás
        cmp r3, #0x00.....      ;végjelzés figyelése
        jz end_length           ;kiurunk, ha vége
        add r1, #1              ;pointer inkrementálás
        add r0, #1              ;hossz növelés
        jmp loop                ;vissza a hurokba
end_length:
        rts...                  ;visszatérés a szubrutinból
```

Maximális pontszám: 75 pont
Rendelkezésre álló idő: 100 perc