

2021 évhónap

NÉV:..... Neptun kód:..... GYAK 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:
Σ :

Ellenőrző kérdések (27p)

E1. (3p) Végezze el a megadott számokon az előírt átalakításokat!

binárisá alakítás 2 digites hexadecimális számból	AB	
2 digites decimálissá alakítás BCD kódolású számból	10000110	
Decimális számból előjel nélküli 4 bites fixpontos binárisá alakítás 2 bit egész rész, 2 bit tört rész	2.75	

E2. (2p) Írja le a Boole algebrai **disztributívias** kétféle alakját!

E3. (1p) Egyszerűsítse az alábbi Boole algebrai kifejezést!

$AB + /A.B + /BC = \dots\dots\dots$

E4. (2p) Az alábbi Verilog leírás egy ismert funkcionális elemet ír le. Adja meg a funkcionális elem **nevét** és jeleinek **funkcióját!**

always@(*)

neve:.....

if(e)

e funkciója:

case(s)

s funkciója:

1'b0: out <= I0;

1'b1: out <= I1;

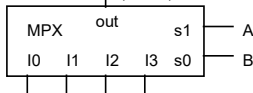
endcase

else out <= 2'b0;

out funkciója:.....

E5. (3p) Valósítsa meg az alábbi multiplexerrel az $f(A,B,C) = AB/C + A/BC + /A/BC + /A/B/C$ függvényt. Rendeljen megfelelő (0,1) konstansokat, és (C, /C) jeleket a bemeneteihez a táblázat kitöltésével!

$f(A,B,C) = AB/C + A/BC + /A/BC + /A/B/C$



I0	I1	I2	I3

E6. (2p) Megadtuk az **f** függvény definícióját igazságtáblával.

a. (1p) Írja le a függvényt megvalósító kombinációs hálózatot Verilog nyelven, **minimalizálatlan SOP alakban**, közvetlenül a mintermeket specifikálva!

A(4)	B(2)	C(1)	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

assign f =.....

b. (1p) Az ABC bemenetre 3 bites előjel nélküli bináris számokat adunk {A, B, C}, 'A' az MSB. Adja meg a függvényt Verilog formában, ha csak a <, >, & operátorokat használhatja!

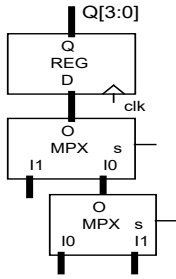
assign f =.....

E7. (2p) Megadtuk egy flip-flop kódolt állapotábráját (bemenet: d, aktuális állapot: q). Rajzolja le az állapotgráfját!

	d = 0	d = 1
q = 0	0	1
q = 1	0	1

Állapotgráf:

E8. (3p) Egészítse ki az alábbi rajzot úgy, hogy a megadott specifikációnak megfelelően működő áramkört valósítson meg! *Írja a megfelelő jelneveket a bemenetekhez ill. egészítse ki a megfelelő funkcionális egységgel, ahol szükséges!*



Egy 4 bites tölthető, engedélyezhető shiftregisztert valósítson meg!
A működés az Ld, E bitekkel így legyen állítható:

Ld, E	1x	01	00
funkció	betölti D[3:0]-at	balra shiftel (belépő bit SI)	tart

E9. (2p) Készítsen a fenti shiftregiszterből olyan áramkört, amely a **rst** jel után a következő 4 kimenetet adja ciklikusan: **0001**, 0010, 0100, 1000, **0001**,... Adja meg, hogy mit kell kötni az alább megadott bemeneteire!

Ld:..... D[3:0] = SI:

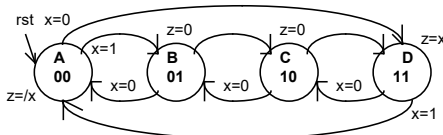
E10. (1p) Egészítse ki az alábbi Verilog leírást *feltételes értékadással*, hogy az egy engedélyezhető 2/1-es buszmultiplexert valósítson meg!

```
wire sel, en;
wire [3:0] ou, in1, in0;
assign ou = .....
```

E11. (6p) Mely állítások igazak és melyek hamisak? Jelölje **+ -al az igaz, -al a hamis** állításokat!

1.	Ha egy 2 bemenetű exor kapu egyik bemenetére 1-t adunk, a kimenetén a másik bemenet invertáltja jelenik meg.	
2.	Egy 3 címbemenetű (8/1-es) multiplexerrel és a 0, 1 konstansokkal tetszőleges 3 változós logikai függvény megvalósítható.	
3.	Tetszőleges logikai függvény előállítható csak AND és OR kapuk segítségével.	
4.	2-es komplement képzésnél a bitenkénti invertálás és az 1 hozzáadás tetszőleges sorrendben végezhető.	
5.	A FIFO-ból a legfrissebb adat olvasható ki először.	
6.	Az ASM leírásban egy feltétel doboz 2 fele ágazhat.	

F1. (13p) Adott egy FSM az alábbi *kódolt állapotgráffal*. Állapotok: A,B,C,D. Bemenetek: clk, rst (hatására az A-ba megy), x. Kimenet: z. Az állapotkódolást és z-t megadtuk. *Készítse el az FSM Verilog leírását* külön az *állapotregiszter*, külön a *next state logika* és külön a *kimeneti logika* megadásával! A leírást elkezdjük, fejezze be!



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

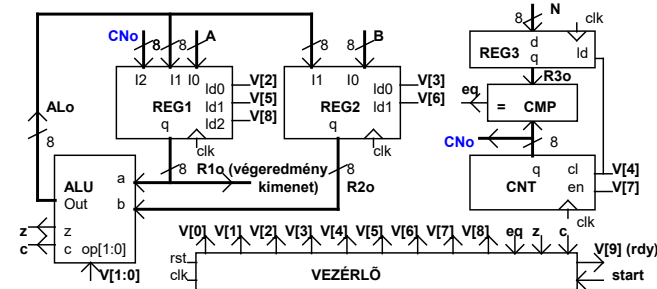
Verilog leírás:

A konstans és változó deklarációk. (2p)

```
localparam [1:0] A = 2'b00, B = ....., C = ....., D = .....;
reg [1:0] s, next_state;
wire z;
```

<pre>//Állapotregiszter (3p): always@(.....) if (rst) s <= else <= //Kimeneti logika (3p): Adja meg a legegyszerűbb Boole algebrai alakot! (2p) assign z = Adja meg az == operátort használó legegyszerűbb alakban (1p) assign z =</pre>	<pre>//Next_state logika (6p): always@(.....) begin case (s) A: if(...) next_state <=; else; B: if(...) next_state <=; else; C: if(...) next_state <=; else; D: if(...) next_state <=; else; default:; endcase end Milyen funkcionális elemet valósít meg az FSM? (1p)</pre>
---	--

F2. (14p) Funkcionális blokkvázlatával (adatstruktúra + vezérlő) adott egy a bemenő adatokkal többféle művelet elvégzésére alkalmas számító modul. Az adatstruktúra **3 db 8 bites adat bemenettel** rendelkezik: **A, B, N** (előjel nélküli számok). Az aritmetikai logikai egység (**ALU**) 4 műveletet tud elvégezni az **a (R1o)** és **b (R2o)** opreandusok között, az alább megadott táblázat szerint. Az **elvégezendő művelet** az ALU **op[1:0]** bemenetén állítható be. **Shiftelés esetén a belépő bit 0.** A **művelet eredményét** az Out (**ALo**) kimenet adja. Az ALU **z** kimenet 1 értéke jelzi, ha a **művelet eredménye 0**. A **c** kimenet **összeadás esetén a túlsordulást, kivonás esetén a negatív eredményt, shiftelés esetén a kilépő bit** értékét jelzi. A REG1 (**R1o**) 3 forrásból, a REG2 (**R2o**) 2 forrásból tölthető **multifunkciós regiszter**. Ezek az **órajel felfutó élére az ld nevébe kódolt sorszámú In bemenet értékét** töltik be (pl. ld0 In0-at). A REG3 egy tölthető (ld) regiszter. A CNT (**CNo**) egy töröhető (cl), engedélyezhető (en) **felfele számláló (CNo)**. Ehhez kapcsolódik a **CMP** komparátor, melynek másik adatát a Reg3 (**R3o**) tölthető (ld) regiszter adja, **eq=(R3o == CNo)**. Az **adatstruktúra kimenete az R1o, egy elvégzett számítási feladat végeredményét itt kell megjelentetni**. A vezérlőt az egy órajel hosszú **start** jel indítja. A vezérlő az adatstruktúrát a **V[8:0] vezérlő jelekkel** működteti. A működése közben figyelni képes az adatstruktúrából jövő z, c és eq feltétel jeleket. Egy számítási feladat elkészültét 1 órajel hosszú **rdy (V[9])** jellel kell jeleznie. Az eredménynek meg kell maradnia a következő start jelig.



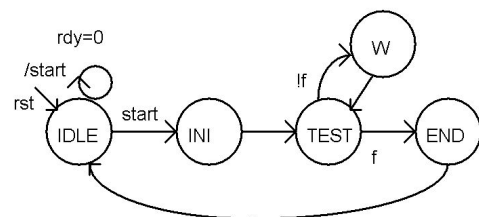
ALU funkció vezérlés: op[1:0]	ALU out (ALo)	z = 1, ha	c = 1, ha
00	a + b	ALo == 0	átvitel van
01	a - b	ALo == 0	a - b < 0
10	a << 1	ALo == 0	a[7]
11	b - a	ALo == 0	b - a < 0

A számoló egységgel a bemeneti adatokon a **4A-B** műveletet kell elvégezni. Feltételezzük, hogy a bemenő adatok megadásánál biztosított, hogy nem lesz túlsordulás.)

A képletet átalakítjuk az adatstruktúrán is elvégezhető műveletké: **(A << 2) - B**

Az A 2-szeri balra shifteléséhez felhasználjuk az adatstruktúra számlálóból és komparátorból álló részét is.

a. (4p) Felrajzoltuk a feladat megoldására alkalmas Moore típusú HLSM gráfot. Rendelje a megadott műveleteket a gráf felsorolt állapotaihoz!
R1o=A, R2o=B, R3o=2, CNo=0, R1o=R1o<<1, R1o=R1o-R2o, CNo=CNo+1, rdy=1



INI:
W:
END:

b. (1p) Válassza ki, a gráfban jelölt f feltétel bitet a felsoroltak közül: c!, c, z, !z, eq, !eq

f =

c. (5p) A fenti HLSM állapotgráf állapotaihoz rendelt műveletek alapján adja meg a vezérlő FSM Moore jellegű állapotgráfjának megadott állapotokban kiadandó V[8:0] vezérlőjelek értékét 9 bites vektorként! (V[9] értékét megadjuk, az csak az END állapotban 1 értékű.)

V[8:0]	8	7	6	5	4	3	2	1	0
INI									
W									
END									

d. (2p) Adja meg a REG2 (R2o) Verilog modul leírását! Legyen az ld0 prioritása nagyobb, mint ld1-é.
module REG2(input clk, ld0, ld1, input [7:0] I0, I1, output reg [7:0] q)
always@(.....)

if(.....) q <=

.....;

e. (2p) Példányosítsa a REG2 modult REG2_i néven és fenti blokkvázlaton szereplő megfelelő jelneveket használva kösse össze a blokkvázlat hozzá kapcsolódó elemeivel!

.....

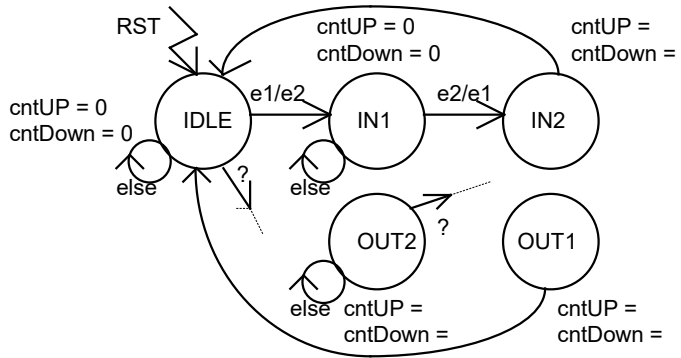
F2. (14p) A feladat egy teremben tartózkodók számának (max. 40 fő) kijelzése. A teremnek egyetlen ajtaja van. Az ajtóban 2 optikai érzékelő (e1, e2) van elhelyezve egymás után olyan távolságban, hogy a belépő nem tudja egyszerre mindkettőt eltakarni. Belépéskor e1, majd e2 ad ki 1 órajel hosszú impulzust, az órajellel szinkronban. Kilépéskor e2, majd e1. (Ha egyszer valaki elindult be vagy ki, akkor nem fordulhat vissza.) Belépéskor egy 2 dekádos BCD fel/le számlálóláncot inkrementálni kell, kilépéskor pedig dekrementálni. A számláló dekádok törlő (Cl), felfele számlálás (Up) és lefele számlálás (Down) vezérlő bemenetekkel rendelkeznek, melyek közül az utóbbi kettő egy vezérlőhöz kapcsolódik. A számlálóknak a kaszkádosításhoz engedélyező bemenetük (E) és végérték jelző kimenetük (TC) is van. A vezérlő figyeli az e1 és e2 érzékelőket, és a jelek sorrendjétől függően inkrementálja vagy dekrementálja a számlálót. Egy külső RST jellel lehet alaphelyzetbe hozni a számlálót és a vezérlőt. Az áramkör órajele egy néhány MHz-es clk jel (pontos értéke itt lényegtelen). A feladat megoldását részfeladatokra bontottuk.

- (5p) Tervezze meg a fenti 10-es modulusú BCD fel/le számlálót! (Adja meg a Verilog leírását!) A leírást alább elkezdjük, fejezze be!
- (4p) Készítsen 2 db számlálóból egy 2 dekádos számlálót! A számlálót kapcsolja össze a vezérlővel! A vezérlő jelei (CntUp, CntDown). A feladatot az alábbi 2 példány hiányzó jeleinek beírásával végezze el!

```
BCDcnt Ones( .clk( ... ), .Cl( ... ), .Up( ... ), .Down( ... ), .E( 1'b1), .TC(TC1), .q( qOnes ));
BCDcnt Tens( .clk( ... ), .Cl(... ), .Up( ... ), .Down( ... ), .E( ... ), .TC( ), .q( qTens ));
```

```
//10-es számláló Verilog leírása:
module BCDcnt( input clk, Cl, Up, Down,
E, output TC, output reg [3:0] q);
assign TC = E & (Up & (q == ) |
Down & (q == ));
always @(posedge clk)
if(Cl) q <=
else
if( )
begin
if( Up )
if( TC ) q <=
else q <=
else
if( )
if( ) q <=
else q <=
end
endmodule
```

c. (4p) Tervezze meg a vezérlő Moore állapotgráfját! A rajzot elkezdjük, fejezze be! Rajzolja be a hiányzó állapot átmeneteket (nyilakat)! Adja meg a hiányzó kimeneteket és állapot átmeneti feltételeket!



Milyen módon oldaná meg az aktuális létszám numerikus kijelzését? Milyen funkcionális egység kell ehhez? Egy rövid mondatban válaszoljon! (1p)

.....

Maximális pontszám: 68 pont
Rendelkezésre álló idő: 100 perc