

20.....év ...hó ...nap

NÉV:..... **MINTA ZH2**..... Neptun kód:..... gyak/lab kurzus:

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

Olvasható aláírás:.....

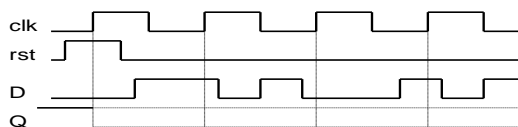
Kedves Kolléga! **A kitöltést a dátum, név és aláírás rovatokkal kezdje!** Az alábbi kérdésekre a válaszokat - ahol lehet - mindig a feladatlapon oldja meg! A feladatok megoldása során a részletes kidolgozást nagyfeladatonként külön papíron végezze, (egyértelműen jelölje, hogy melyik lap melyik feladathoz 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.** Ha több a feladat, mint a pont, a feladat pontszámát csökkentjük minden hiba esetén (azonban negatív pontot nem adunk). Jó munkát!

E:
F1:
F2:
Σ :
IMSC:

Ellenőrző kérdések (25p)

E1. (4p) a. Egy felfutó él érzékeny szinkron törölhető (rst) D flip-flop az alábbi jeleket kapja. Rajzolja le a Q kimenet idődiagramját! **b.** Adja meg egy törölő (rst) bemenettel rendelkező D flip-flop Verilog viselkedési leírását! Elkezdjük, folytassa!

```
always@(.....)
if(.....) .....
```

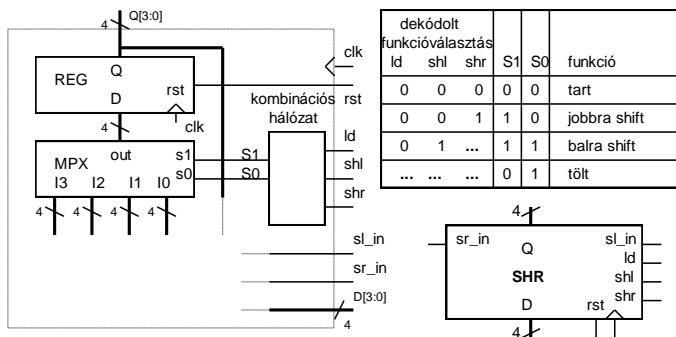


E2. (3p) Adja meg egy 2 db 8 bites forrásból betölteni képes **multifunkciós regiszter** Verilog leírását! **Ha ld0=1, q-ba töltse be In0-at, ha ld1=1, q-ba töltse be In1-et! Az ld0 legyen nagyobb prioritású!** (Most nem kell alaphelyzetbe állítás rst-re.) Elkezdjük, folytassa!
A hiányzó változó definícióktól eltekintünk.

```
always@(.....)
if(.....) .....
```

E3. (4p) Adott egy 2 irányú shiftregiszter funkcionális blokkvázlata és működési táblázata.

a. Adja meg, bitsorrend helyesen, hogy milyen jelek kapcsolódnak az MPX multiplexer egyes bemeitire! Ügyeljen arra, hogy a rajzon látható jel neveket használja!



I0:.....
I1:.....
I2:.....
I3:.....

b. Töltse ki a táblázat hiányzó részeit, ha a prioritás sorrendje a legerősebb jellel kezdve a következő: betöltés (ld), balra shift (shl), jobbra shift (shr)!

E4. (5p) Adja meg egy 4 bites 10-es modulusú fel le számláló Verilog kódját! A számláló rst-re törölődik, ld-ra betölti d[3:0]-at, cnt_up=1 esetén felfele számol, cnt_down=1 esetén lefele számol, egyébként nem változik a q kimenete. A vezérlőjelek prioritása a legerősebb jellel kezdve a következő: rst, ld, cnt_up, cnt_down. A jelek definiálásától most eltekintünk.

<pre>..... tc = always@(.....) if(.....) q <= else if(.....) begin if(.....) q <= end</pre>	<pre>else if(.....) begin q <= end</pre>
---------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------

E5. (3p) Adott 3 db számláló modul példányosítva, azonban az összekötésük hiányzik. Az első 7-es modulusú, a második 3-os modulusú, a harmadik 5-es modulusú.

a. Kaszkádosítsa a felsorolás szerinti sorrendben őket megadott wire típusú jelek megfelelő felhasználásával úgy, hogy a teljes számláncot a külső **EN** jel engedélyezze!

wire EN, OUT;
wire [1:0] tc_i;

cnt7 cnt7_1(.clk(.....), .rst(.....),.en(.....), .q(q10), .tc(.....));

cnt3 cnt3_1(.clk(.....), .rst(.....),.en(.....), .q(q10), .tc(.....));

cnt5 cnt5_2(.clk(.....), .rst(.....),.en(.....), .q(q10), .tc(OUT));

b. Hány órajelenként jelenik meg impulzus az **OUT** kimeneten?

E6. (6p) Mely állítások igazak és melyek hamisak? Jelölje **+-al az igaz, --al a hamis** állításokat!

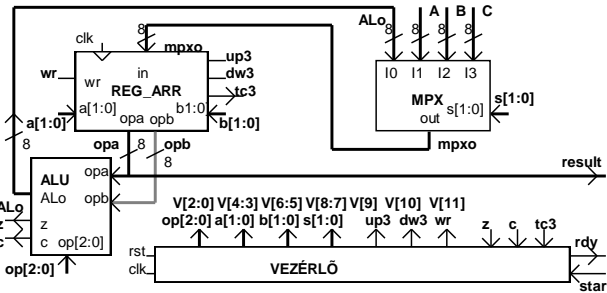
1.	Egy sorrendi hálózat (FSM) állapotgráfjának állapotszáma biztosan nem csökkenthető úgy, hogy a működése az állapotcsökkentés előtti változatával azonos maradjon.	
2.	Ha egy sorrendi hálózat kimenetét az állapotregiszter valamely bitjei adják, akkor az biztosan Moore modell szerint működik.	
3.	A FIFO -ba bármikor lehet adatot írni.	
4.	Egy FSM vezérlő állapotgráfjába írt állapotátmeneti feltétel lehet $A == B$.	
5.	Az ASM (algoritmikus állapotvezérlő) leírás egyetlen feltétel doboza csak 2-felé ágazást tesz lehetővé.	
6.	3 regiszter című processzor architektúra esetén egy művelet eredménye nem rontja el szükségyszerűen az egyik operandust.	

F1. (15p) Adott egy FSM az alábbi **kódotl állapotgráfjával**. A kimenete az állapotkód és **z**. Végezze el az alábbi feladatokat! **Az állapot kódok helyett mindenütt állapot neveket használjon!**

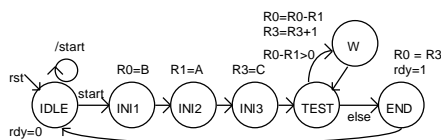
<p>bemenet: x[1:0] kimenet: s[2:0], z</p> <p>a. (3p) Adja meg az állapotregiszter Verilog leírását! //A konstans és változó deklarációk localparam [2:0] A = 3'b000, B = 3'b001, C = 3'b010, D = 3'b011, E = 3'b100; reg [2:0] s, next_s; wire clk, rst; wire [1:0] x; //Állapotregiszter always @ (.....) if (rst) s <=; else s <=</p>	<p>b. (9p) Következő állapot logika //Next_state logika always @ (.....) case (s) A: case(.....) 2'b01: next_s <=; 2'b10: next_s <=; 2'b11: next_s <=; default: next_s <=; endcase B: if(x == 2'b00) next_s <=; else next_s <= C; C: if(.....) next_s <=; else next_s <=; D: if(.....) next_s <=; else next_s <=; E: if(.....) next_s <=; else next_s <=; default: next_s <=; endcase</p>
<p>c. (1p) Milyen modell szerint működik? (Húzza alá a megfelelőt) Mealy Moore</p>	
<p>d. (2p) Adja meg a z kimenet logikai függvényét Verilog-ban! Ha csak az s-t, az állapot neveket, az == operátort és műveletet használhatja: assign z = A legegyszerűbb alakban: assign z =</p>	

F2. (20p) 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**) 8 műveletet tud elvégezni az **opa** és **opb** operandusokkal az alább megadott táblázat szerint. Az **elvégezendő művelet** az **ALU op[2:0]** bemenetén állítható be. **Shiftelés esetén a belépő bit 0. A művelet eredményét az ALo kimenet adják.** Az **ALU z** kimenet 1 értéke jelzi, ha a **művelet eredménye 0**. A **c** kimenet összeadás esetén a **túlszordulást**, kivonás esetén a **negatív eredményt**, shiftelés esetén a **kilépő bit értékét** jelzi. A **REG_ARR** egy **4 regisztert tartalmazó regiszter tömb**. Ennek **opa** kimenetén az **a[1:0]-val kiválasztott sorszámú regiszter tartalma** jelenik meg, ha **a[1:0]=i**, akkor **Ri**. Az adat

beírását $wr=1$ engedélyezi és az in bemenetere kiválasztott adat (mpxo) az a[1:0]-val kiválasztott regiszterbe íródik az órajel felfutó élére. Az opb kimenetén a b[1:0]-val kiválasztott regiszter tartalma jelenik meg, ha $b[1:0]=i$, akkor Ri. A REG_ARR bemenetere kerülő adat (mpxo) az MPX multiplexer s[1:0] bemenetével választható ki. A regisztertömb R3 regisztere speciális, mert fel/le számlálóként is funkcionál. Normál regiszterként írható (a[1:0]=2*b11 és $wr = 1$ esetén beíródik R3-ba az MPX-el kiválasztott érték) és normál regiszterként olvasható. Azonban, ha éppen nem írják, akkor $dw3=1$ esetén lefele számol ($R3=R3-1$), $up3 = 1$ esetén pedig felfele számol ($R3=R3+1$), ha megjön a clk aktív éle. Az írás erősebb, mint a számlálás engedélyezés. Az R3 regiszter fel vagy le számláltatásával egyidőben bármely más regiszterbe lehetséges írni. A tc3 kimenet akkor jelez, ha $R3=0$. Az adatstruktúra kimenete az opa (result), egy elvégzett számítási feladat végeredményét itt kell megjelentetni. A vezérlő egy kívülről jövő, 1 órajel hosszú start parancs indítja. A vezérlő az adatstruktúrát a V[11:0] vezérlő jelekkel működteti. A működése közben figyelni képes az adatstruktúrából jövő z, c és tc feltétel jeleket. Egy számítási feladat elkészültét 1 órajel hosszú rdy státusz jellel kell jeleznie. Az eredménynek meg kell maradnia a következő start jelig. Az aktuális művelet szempontjából érdektelen vezérlő jelek értékét 0-nak kell választani. Így, ha az eredmény az R0 regiszterben van, akkor a vezérlő regiszterművelet nélküli állapotaiban az opa (eredmény kimenet) automatikusan az R0 tartalma lesz.



a. (10p) Az alábbi Moore jellegű HLSM állapot diagram által leírt feladatot a fenti adatstruktúrával kell megvalósítani.



(A feladat a B/A egész részét állítja elő, C=0.) Adja meg a

ALU funkció vezérlés: op[2:0]	ALU out (Alo)	z =1, ha	c = 1, ha
000	opa + opb	Alo==0	átvitel van
001	opa - opb	Alo==0	a-b < 0
010	opa << 1	Alo==0	opa[7]
011	opb >> 1	Alo==0	opb[0]
100	opa & opb	Alo==0	Soha (c=0)
101	opa opb	Alo==0	Soha (c=0)
110	opa ^ opb	Alo==0	Soha (c=0)
111	opb	Alo==0	Soha (c=0)

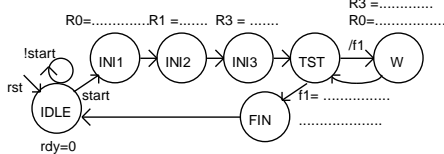
vezérlő állapotgráfiának megadott állapotaiban kiadandó vezérlőjeleket a V[11:0] bitek értékének (0, 1) alábbi táblázatba való beírásával! (Segítségképp néhány értéket megadtunk. A rdy-t most nem kérjük.)

	wr	dw3	up3	s[1:0]	b[1:0]	a[1:0]	op[2:0]
V[11:0]	11	10	9	8 7	6 5	4 3	2 1 0
INI1					0 0		0 0 0
INI2					0 0		0 0 0
INI3					0 0		0 0 0
TEST				0 0			
W				0 0			
END				0 0			

b. (1p) A vezérlő blokkvázlat mely feltétel jele képes jelezni az $R0-R1 > 0$ feltételt?
(Segítő kérdés: Melyik műveletet kell beállítani ehhez a TEST állapotban?)

c. (7p) A számító modullal a $C*B$ szorzatot kell kiszámítani (Az A bemenetet most nem használjuk).

A megvalósítás: A 0 kezdőértékű R0 regiszterhez C-szer hozzáadjuk B értékét. A hozzáadások számát a speciálisan számlálóként is használható R3 regiszterrel számoljuk. Csak az R0, R1 és R3 regisztereket használhatja. Rendelje a számítás elvégzéséhez szükséges műveleteket a Moore jellegű HLSM állapotaihoz. Az alább felsorolt műveletekből válasszon (szükségtelenek is vannak közöttük). Írja az elvégzendő művelet(ek) sorszámát a megfelelő állapot neve mellé! Nem biztos, hogy van elvégzendő művelet, ilyenkor írjon x-et! (A HLSM gráf segítő információkat tartalmaz. A gráf pontozott részzeit nem kell kitölteni, de az is segítő információ.) A többlet és hiányzó művelet sorszámokért pontlevonás jár!



1: $R0 = R0+R1$	2: $R0 = A$	3: $R0 = R0 \oplus R0$	4: $R0 = C$
5: $R1 = C$	6: $R1 = A$	7: $R1 = B$	8: $R1 = R0+R1$
9: $R3 = R3+1$	10: $R3 = C$	11: $R0 = R0+R1, R3 = R3-1$	
12: $R0 = R0+R1, R3 = R3+1$	13: $rdy = 1$		
IDLE:.....	INI1:.....	INI2:.....	INI3:.....
TST:.....	W:.....	FIN:.....	

d. (2p) Adja meg a kért feltétel(ek)et magasztintú feltétel megadással!

f1:

Adja meg a kért feltéte(lek)et blokkvázlaton szereplő azon jel(ek) megnevezésével, amely(ek) jelzi(k) a feltétel teljesülését. A parancs és feltétel bitek negáltját is fel lehet használni.

f1:

e. (5p) Adja meg az adatstruktúra REG_ARR regiszter tömbjének Verilog leírását a bevezetőben megadott információk alapján! Az R3 speciális számláló funkciójának megvalósításától most eltekintünk. A leírást elkezdjük, egészítse ki a hiányzó részekkel!

<pre>module REG_ARR (input clk, wr, input [1:0] a, b, input [7:0] in; output reg [7:0] opa, opb);</pre>	
<pre>// beírás megvalósítása reg [7:0] arr[3:0]; always@(.....) if(.....) <=;</pre>	<pre>always@(.....) // opa elő- // állítása <=;</pre>
<pre>always@(.....) // opb elő- // állítása <=;</pre>	
<pre>endmodule</pre>	

IMSC1. (3p)

Tervezzon **programozható modulusú 4 bites számlálót!** A számláló modulusa 2-től 16-ig legyen változtatható a 4 bites bemenetére adott szám függvényében. Adja meg az egység Verilog kódját!

IMSC2. (4p)

Készítsen az F2 feladat adatstruktúrájához egy olyan HLSM diagram részletet, amely **a regisztertömb R0 és R1 regiszterének értékét cseréli fel , a legkevesebb művelettel.** Milyen műveletet kell beállítani az op[2:0]-on?

Maximális pontszám: 60 pont (IMSC: 7p) Rendelkezésre álló idő: 100 perc