Arhitectura	_:_41		1 1
Arhitectura	sistemel	or de	calcul

Numele:		
Data:	Grupa:	E-mail:

- 1. Fie x = 112 si y = 37.
- a) Convertiți x și y în baza 16.
- b) Convertiți mai departe x și y din baza 16 în baza 2 și din baza 2 în baza 8, direct (fără a trece prin baza 10).
- c) Calculați x y lucrând în baza 8.
- d) Convertiți rezultatul din baza 8 în baza 10.
- e) Calculați z=x-y folosind reprezentarea în complement față de 2 pe 8 biți. Se vor explicita reprezentările lui x și y, complementul față de 1 și cel față de 2 al reprezentării lui y, obținerea reprezentării lui z, interpretarea acesteia ca număr în baza 10.
 - f) Interpretați ca număr în baza 10 reprezentarea internă hexa în format single OxC1C80000.
 - 2. Fie $f: B_2^3 \longrightarrow B_2^2$, $f(x, y, z) = (f_1(x, y, z), f_2(x, y, z))$, unde: $f_1(x, y, z) = (x \oplus \bar{y}) (y \oplus \bar{z})$,

 $f_2(x,y,z)=1$ d.d. secvenţa $x,\,y,\,z$ este crescătoare (adică $x\leq y\leq z$).

- a) Construiți tabelul de valori al lui f și scrieți f_1 , f_2 în FND și FNC.
- b) Implementați f printr-un PROM.
- c) Implementați f printr-un codificator.
- d) Implementați f printr-un circuit cu două multiplexoare (câte unul pentru f_1, f_2) cu aceiași selectori x, y, z.
- e) Implementați f printr-un circuit care conține doar multiplexori elementari; apoi, reduceți la maximum numărul multiplexorilor elementari (nu este permisă adăugarea de porți NOT).
- f) Construiți un circuit 1-DS care citește unul câte unul o secvență de biți și, de fiecare dată, scoate 1 / 0, după cum $f_2(x, y, z) = 1 / 0$, unde x, y, z sunt antepenultimul, penultimul, respectiv ultimul bit citit.
 - g) Implementați porta NXOR printr-un circuit care conține doar porți NOR.
 - 3. Considerăm implementarea procesorului MIPS cu 1 ciclu / instrucțiune (vezi verso). Fie fragmentul de program: .data la \$t3, x

x: .word 5 et:
.text lw \$t4, 0(\$t3)
main: sub \$t2, \$t2, \$t4
li \$t2, 5 beq \$t2, \$0, et

Presupunem că în memorie instrucțiunea lw din program are adresa α , iar variabila x are adresa β .

- a) Pentru instrucțiunile lw și beq din program scrieți câmpurile din reprezentarea lor internă (ex: op/rs/rt/imm, valorile se scriu hexa); pentru beq din program scrieți și reprezentările ei binară (32 biți) și hexa (8 cifre hexa).
- b) Completați tabelul următor cu valorile obținute la prima executare a instrucțiunilor lw și beq din program; valorile se vor scrie hexa/formulă, iar dacă valoarea este necunoscută/nedefinită se va nota "?"; în coloanele PC și \$t2 se vor trece valorile noi, de la sfârșitul executării instrucțiunilor respective:

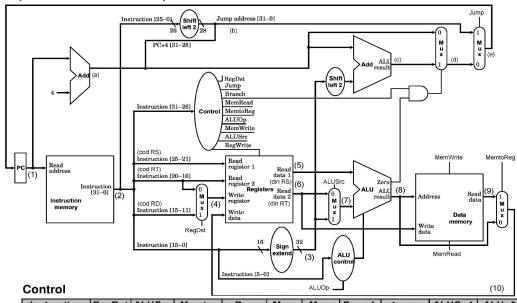
	1	4	5	7	8	ALU zero	10	(d)	(e)	Branch	Mem To Reg	ALU Op (2b)	ALU Ctrl (3b)	Reg Write	PC	\$t2
Iniţial															α	5
lw	α															
beq																

c) Adăugați procesorului implementarea instrucțiunii: dlw rd, rs, rt care încarcă în registrul rd valoarea aflată în memorie la adresa care se obține adunând valorile din registrii rs și rt (adică efectuează rd := mem[rs + rt]).

Pentru implementare, este suficientă adăugarea unei linii tabelului "Control" (codul op = - este o valoare nouă, nerelevantă). Completați această linie:

Instruction	RegDst	ALU	Mem	Reg	Mem	Mem	Branch	Jump	ALU	ALU
[31 - 26]		Src	То	Write	Read	Write			Op1	Op0
			Reg							

Implementarea cu un ciclu pe instructiune



	Instruction	RegDst	ALUSrc	Memto- Reg	Reg Write	Mem Read	Mem Write	Branch	Jump	ALUOp1	ALUp0
0x0	R-format	1	0	0	1	0	0	0	0	1	0
0x23	lw	0	1	1	1	1	0	0	0	0	0
0x2b	sw	X	1	Х	0	0	1	0	0	0	0
0x4	beq	X	0	Х	0	0	0	1	0	0	1
0x2	i	Х	Х	Χ	0	0	0	0	1	Х	Х

ΛI	11	Cor	tro
AL	.U	COL	шo

31-26 25-----0

Registri: \$t0 (8) - \$t7 (15)

		ALUOp			Ca	mp f	unct	Operatie			
		ALUOp ₁	ALUOp ₀	F5	F4	F3	F2	F1	F0		
l	N/SW		0	Х	Х	Х	Х	Х	Х	010	(+)
	beq		1	Х	Х	Х	Х	Х	Х	110	(-)
	add	1	Х	Х	Х	0	0	0	0	010	(+)
	sub	1	X	Х	Х	0	0	1	0	110	(-)
	and	1	Х	Х	Х	0	1	0	0	000	(and)
R-	or	1	Х	Х	Х	0	1	0	1	001	(or)
forma	alslt	1	Х	Х	Х	1	0	1	0	111	(slt)

ALU Operation

ALU control input	Function
000	and
001	or
010	add
110	subtract
111	set on less than

```
\verb"add/sub/slt" rd,rs,rt" \# rd := rs+rt, rd := rs-rt, rd := (rs<rt)?1:0
\# \mid 0 \mid rs \mid rt \mid rd \mid 0 \mid 0x20/0x22/0x2a \mid
# 31-26 25-21 20-16 15-11 10-6 5-----0
# 6b 5b 5b 5b 5b 6b
                                                                      # 31-----26 25-21 20-16 15---0
# 6 b 5b 5b 16 b
beq rs,rt,et
# if rs=rt then goto et
# if rs=rt then PC:=PC+4+imm*4 else PC:=PC+4
# | 0x4 | rs | rt | imm= (et-PC-4)/4 |
# 31-26 25-21 20-16 15-----0
# 6 b 5 b 5 b 16 b
# goto et
# PC:=(PC+4) & 0xf0000000 + imm*4
# | 0x2 | imm |
# ------
```