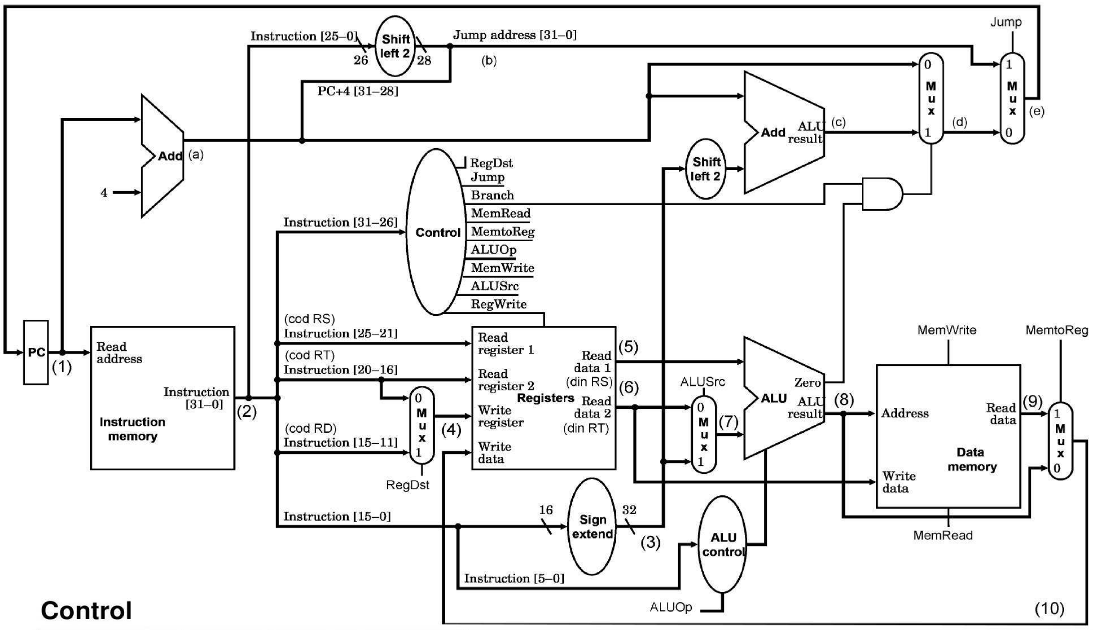
Implementarea cu un ciclu pe instructiune



Control

	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	X	0	0	1	0	0	0	0
0x4	beq	X	0	X	0	0	0	1	0	0	1
0x2	j	X	X	X	0	0	0	0	1	X	X

ALU Control

	ALUOp		Camp functie						Operatie	
	$ALUOp_1$	$ALUOp_0$	F5	F4	F3	F2	F1	F0		
lw/sw	0	0	X	X	X	X	X	X	010	(+)
beq	X	1	X	X	X	X	X	X	110	(-)
add	1	X	X	X	0	0	0	0	010	(+)
sub	1	X	X	X	0	0	1	0	110	(-)
and	1	X	X	X	0	1	0	0	000	(and)
R- or	1	X	X	X	0	1	0	1	001	(or)
format slt	1	X	X	X	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

```
add/sub/slt rd,rs,rt # rd := rs+rt, rd := rs-rt, rd := (rs<rt)?1:0
# |   0   |  rs   |  rt   |   rd  |    0   | 0x20/0x22/0x2a |
# ----------------------------------------------------------------
#  31-26 25-21 20-16 15-11  10-6   5----------0
#    6 b     5 b     5 b     5 b     5 b       6 b
```

```
lw/sw rt,imm(rs) # rt :=/=: mem[(rs)+imm]
 # | 0x23/0x2b  |  rs   |   rt   |   imm    |
 # ------------------------------------------------
 #   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
```

```
j et
# goto et
# PC:=(PC+4) & 0xf0000000 + imm*4
# |  0x2  |     imm     |
# -------------------------
#  31-26  25-------0
```

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