UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

PROCESORUL MIPS CICLU UNIC- 16 BITI

Student: Hădărău Ioana

Grupa:30223

a) Cele 4 instructiuni suplimentare

1. Tipul R

- xor $rd, $rs, $rt RTL: RF[rd] ←RF[rs] ^ RF[rt] //Exclusive OR

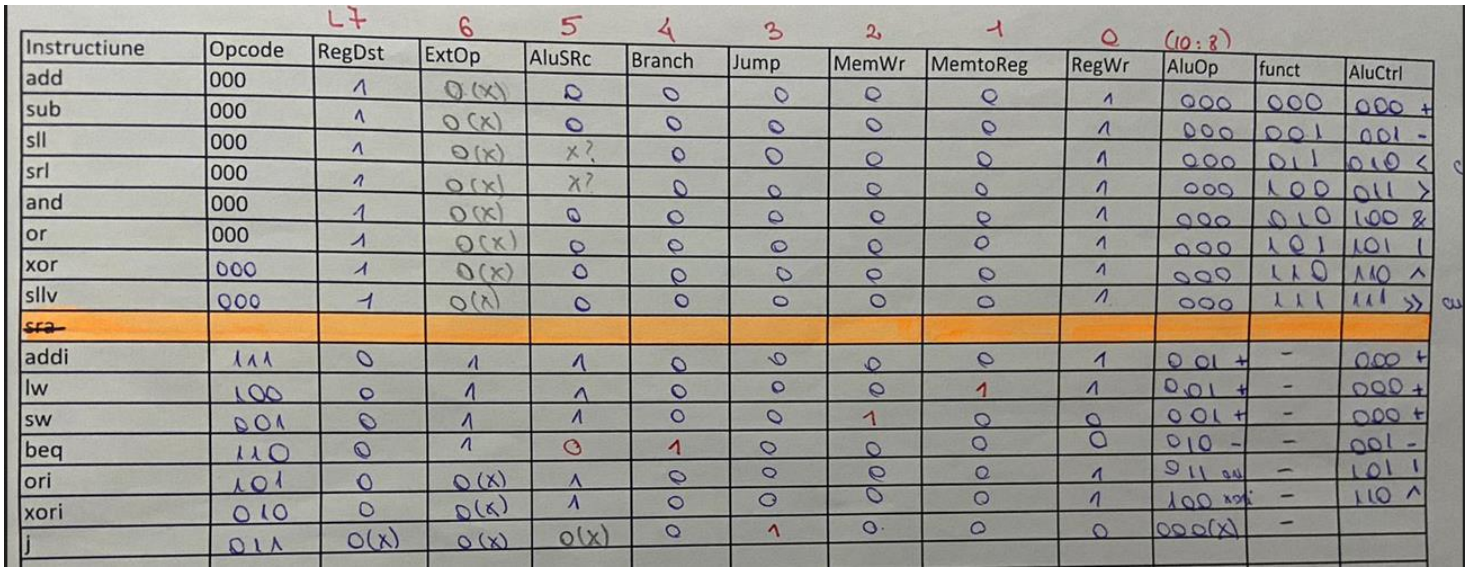
- sllv $rd, $rs, $rt RTL: RF[rd] ←RF[rs] << RF[rt] //Shift left logical variable

2. Tipul I

- ori $rt, $rs, imm RTL: : RF[rt] ←RF[rs] | Z_Ext(imm) // Logical OR unsigned constant

- xori $rt, $rs, imm RTL: RF[rt] ←RF[rs] & Z_Ext(imm) // Logical AND unsigned constant

b) Semnale control MIPS16 pentru Anexa 5



	7	6	5	4	3	2	1	0	(10:8)		
Instructiune	Opcode	RegDst	ExtOp	AluSRc	Branch	Jump	MemWr	MemtoReg	RegWr	AluOp	funct	AluCtrl
add	000	1	0 (x)	0	0	0	0	0	1	000	000	000 +
sub	000	1	0 (x)	0	0	0	0	0	1	000	001	001 -
sll	000	1	0 (x)	x?	0	0	0	0	1	000	011	010 <
srl	000	1	0 (x)	x?	0	0	0	0	1	000	100	011 >
and	000	1	0 (x)	0	0	0	0	0	1	000	010	100 &
or	000	1	0 (x)	0	0	0	0	0	1	000	101	101 |
xor	000	1	0 (x)	0	0	0	0	0	1	000	110	110 ^
sllv	000	1	0 (x)	0	0	0	0	0	1	000	111	111 >>
~~sra~~												
addi	111	0	1	1	0	0	0	0	1	0 01 +	-	000 +
lw	100	0	1	1	0	0	0	1	1	0 01 +	-	000 +
sw	001	0	1	1	0	0	1	0	0	001 +	-	000 +
beq	110	0	1	0	1	0	0	0	0	010 -	-	001 -
ori	101	0	0 (x)	1	0	0	0	0	1	011 or	-	101 |
xori	010	0	0 (x)	1	0	0	0	0	1	100 xor	-	110 ^
j	011	0 (x)	0 (x)	0 (x)	0	1	0	0	0	000 (x)	-	

c) Program- memoria ROM

Programul implementat pe procesor are scopul de a face suma tuturor numerelor de 2 cifre divizibile cu 3. Pentru a usura vizualizarea programului pe placuta am modificat capatul din dreapta al intervalului de insumare, astfel programul calculeaza in intervalul [12,24]

Programul în C

```
int s=0;
int i=9;
while(i<24)
        {
                i=i+3;
               s=s+i;
        }
```

Programul in cod masina - Programul în limbaj de asamblare

b"111_000_001_000_1001", adi r1,r0,9

b"111_000_100_001_1000", adi r4,r0,24

b"111_000_101_000_0000", adi r5,r0,0

b"110_100_001_000_0011", beq r1,r4,2

b"111_001_001_000_0011", addi r1.r.3

b"000_101_001_101_0000", add r5,r5,r1

b"011_000_000_000_0011", j 3

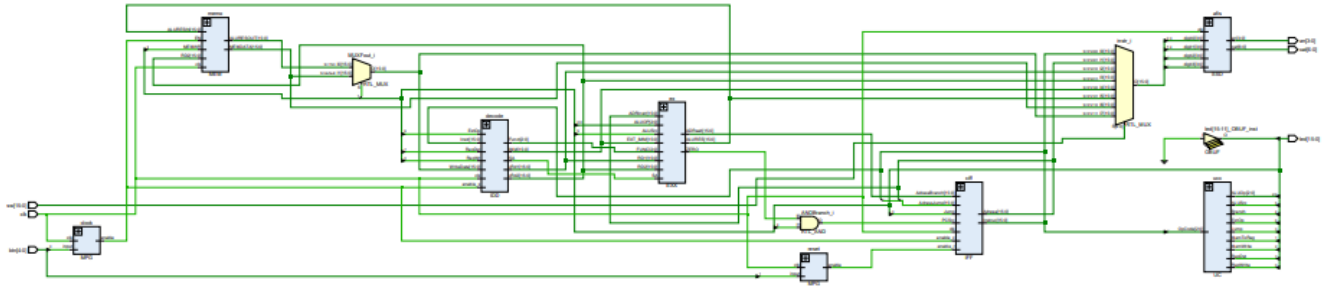
b"001_000_101_000_1010", sw r5,10(r0)

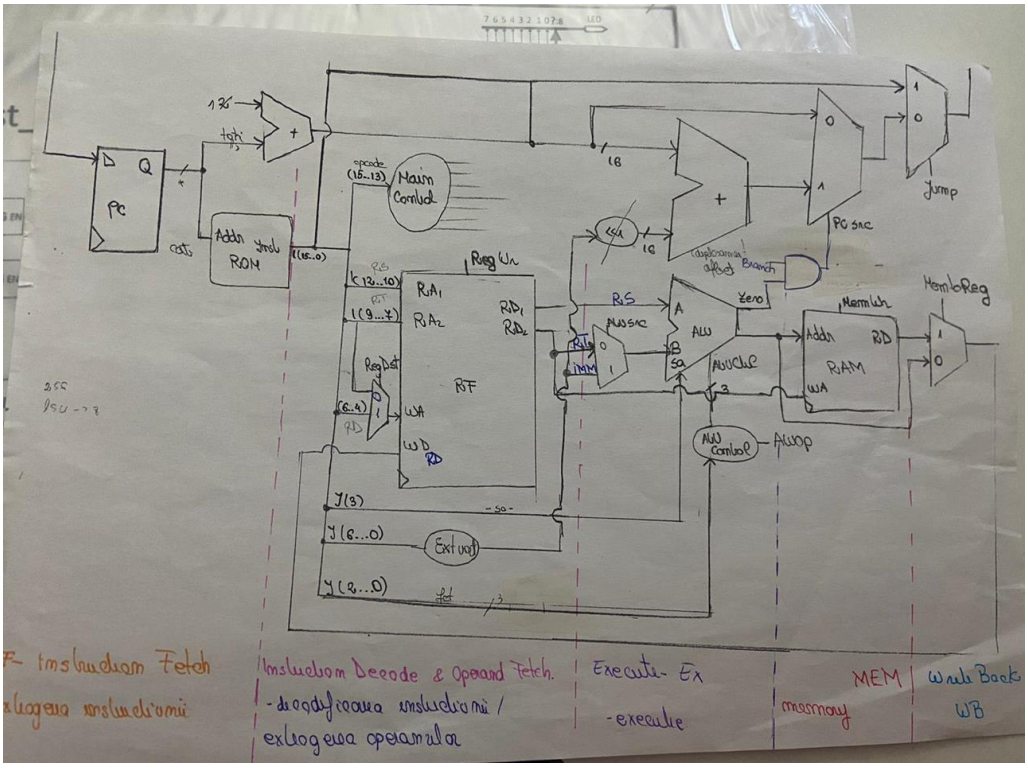
Trasarea executiei programului

Pas	SW(7:5)	"000"	"001"	"010"	"011"	"100"	"101"	"110"	"111"	De completat numai pentru instrucțiuni de salt	
	Instr (*în asamblare*)	Instr (*hexa*)	PC+1	RD1	RD2	Ext_Imm	ALURes	MemData	WD	BranchAddr	JumpAddr
0	ADDI ...$1, $0, 9	x"E089"	x"0001"	x"0000"	x"0000"	x"0009"	x"0009"	x"0000"	x"0009"		
1	ADDI $4, $0, 24	x"E218"	x"0002"	x"0000"	x"0000"	x"0018"	x"0018"	x"0000"	x"0018"		
2	ADDI $5, $0, 0	x"E280"	x"0003"	x"0000"	x"0000"	x"0000"	x"0000"	x"0000"	x"0000"		
3	BEQ $1, $4, 3	x"D083"	x"0004"	x"0018"	x"0009"	x"0003"	x"0007"	x"0000"	x"0007"	x"0007"	
4	ADDI $1, $1, 3	x"E483"	x"0005"	x"0009"	x"0009"	x"0003"	x"000C"	x"0000"	x"000C"		
5	ADDI $5, $5, $1	x"1A50"	x"0006"	x"0000"	x"000C"	x"0050"	x"000C"	x"0000"	x"000C"		
6	j 3	x"8003"	x"0007"	x"0000"	x"0000"	x"0008"	x"0000"	x"0000"	x"0000"		x"0003"
7	SW $5, 10($0)	x"2A8A"	x"0008"	x"0000"	x"005A"	x"000A"	x"000A"	x"005A"	x"000A"		
8	// instrucțiunea SW se execută la finalul programului; după ce se calculează										
9	suma										

d) Nu există părți din processor incomplete.

e) Nu există erori la implementarea în VHDL.
Schema corespunzătoare procesorului MIPS ciclu unic pe 16 biți cu instrucțiunile suplimentare adăugate este următoarea:





f) Programul a fost testat pe plăcuță FPGA BASYS 3 și funcționează conform așteptărilor.