**Raport MIPS Pipeline**

**Serban Alexandra**

**Grupa 30225**

**Prof. Coord. Mircea Paul Muresan**

1. **Descrierea programului ales**

Programul pe care l-am ales presupune parcurgerea unui vector, gasirea minimului si maximului din sir si apoi verificarea daca cele 2 sunt divizibile. Descrierea programului in C este urmatoarea:

int a[]= {3,4,6,7,5,200,7,1};

int min=999, max=-999;

int i=0;

while(i<8)

{

if(min > a[i] )

min=a[i];

if(max < a[i] )

max=a[i];

i++;

}

while(max>0){

max=max-min;

}

Rezultatul final se va gasi stocat in max. La inceput sunt initializate valorile vectorului, contorul i si valorile min si max, de preferat min cu o valoare cat mai mare si max cu o valoare cat mai mica. Vom presupune ca valoarea vectorului este mereu de 8. In primul while se afla valorile min si max si in cel de al 2-lea while se calculeaza daca min si max sunt divizibile.

1. **Descrierea in mips 16 cu hazardurile rezolvate**

addi $0, $0, 0

addi $1, $0, 99

addi $2, $0, -99

addi $3, $0, 8

addi $4, $0, 0

addi $6, $0, 0

lw $5,4($6)

beq $0,$3,19

addi $6,$0,0

addi $6,$0,0

addi $6,$0,0

bge $5,$1,4

addi $6,$0,0

addi $6,$0,0

addi $6,$0,0

lw $1,4($6)

bge $5,$2,4

addi $6,$0,0

addi $6,$0,0

addi $6,$0,0

lw $2,4($6)

addi $0,$0,1

addi $6,$6,1

addi $0,$0,0

addi $0,$0,0

addi $0,$0,0

lw $5,4($6)

j 7

addi $6,$0,0

bge $4,$2,5

addi $6,$0,0

addi $6,$0,0

addi $6,$0,0

sub $2,$2,$1

j 29

addi $6,$0,0

1. **Tabel hazarduri**

****

1. **Codificare cod masina instructiuni**

B"001\_000\_000\_0000000", -- 0. 2000 ADDI $0, $0, 0

B"001\_000\_001\_1100011", -- 1. 20E3 ADDI $1, $0, 99

B"001\_000\_010\_0011101", -- 2. 211D ADDI $2, $0, -99

B"001\_000\_011\_0001000", -- 3. 2188 ADDI $3, $0, 8

B"001\_000\_100\_0000000", -- 4. 2200 ADDI $4, $0, 0

B"001\_000\_110\_0000000", -- 5. 2300 ADDI $6, $0, 0

B"010\_110\_101\_0000100", -- 6. 5A84 LW $5,4($6)

B"100\_000\_011\_0010011", -- 7. 8193 BEQ $0,$3,19

B"001\_000\_110\_0000000", -- 8. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", -- 9. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --10. 2300 ADDI $6,$0,0

B"110\_101\_001\_0000100", --11. D484 BGE $5,$1,4

B"001\_000\_110\_0000000", --12. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --13. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --14. 2300 ADDI $6,$0,0

B"010\_110\_001\_0000100", --15. 5884 LW $1,4($6)

B"110\_010\_101\_0000100", --16. CA84 BGE $2,$5,4

B"001\_000\_110\_0000000", --17. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --18. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --19. 2300 ADDI $6,$0,0

B"010\_110\_010\_0000100", --20. 5904 LW $2,4($6)

B"001\_000\_000\_0000001", --21. 2001 ADDI $0,$0,1

B"001\_110\_110\_0000001", --22. 3B01 ADDI $6,$6,1

B"001\_000\_000\_0000000", --23. 2000 ADDI $0,$0,0

B"001\_000\_000\_0000000", --24. 2000 ADDI $0,$0,0

B"001\_000\_000\_0000000", --25. 2000 ADDI $0,$0,0

B"010\_110\_101\_0000100", --26. 5A84 LW $5,4($6)

B"111\_0000000000111", --27. E007 J 7

B"001\_000\_110\_0000000", --28. 2300 ADDI $6,$0,0

B"110\_100\_010\_0000101", --29. D105 BGE $4,$2,5

B"001\_000\_110\_0000000", --30. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --31. 2300 ADDI $6,$0,0

B"001\_000\_110\_0000000", --32. 2300 ADDI $6,$0,0

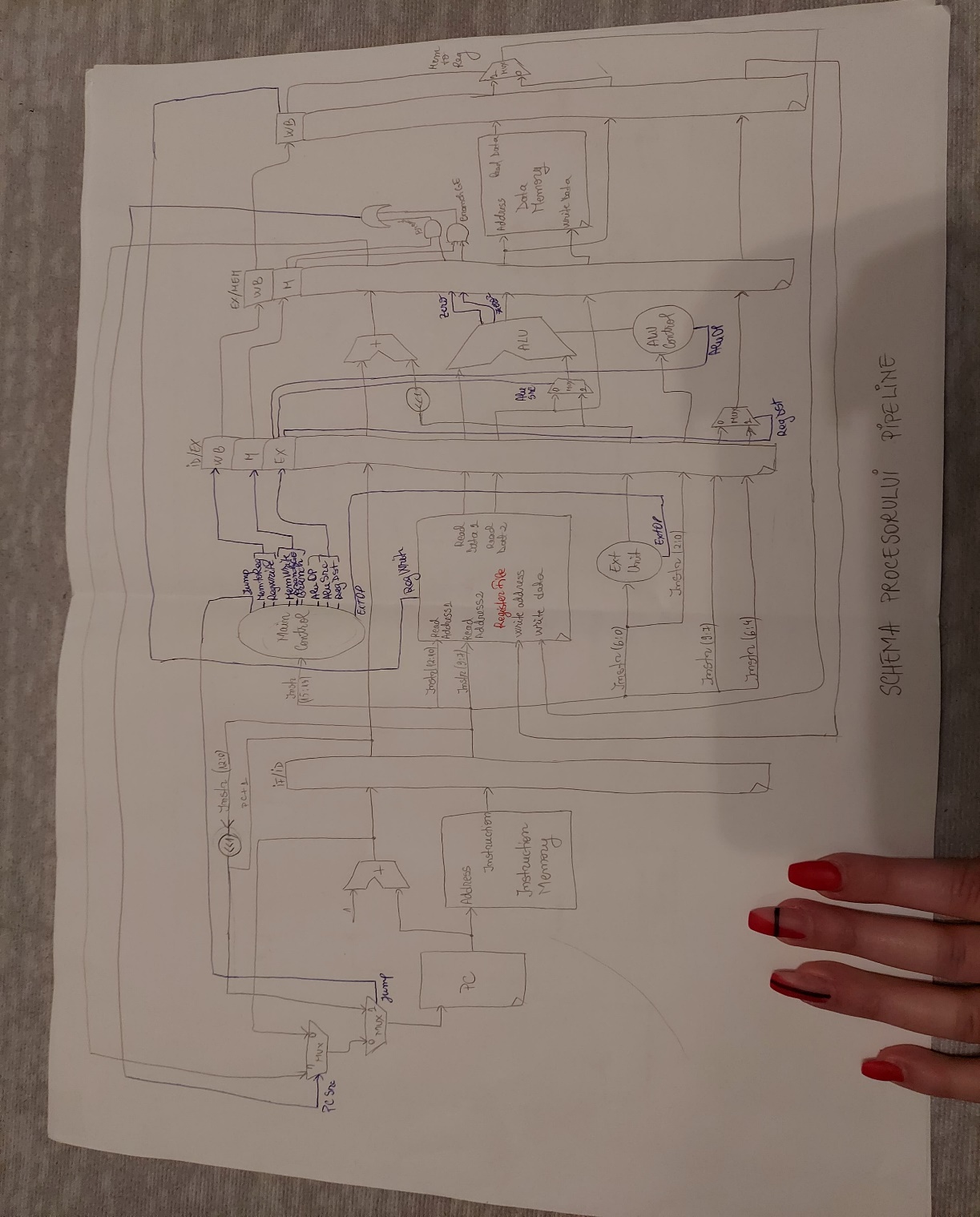
B"000\_010\_001\_010\_0\_001", --33. 08A1 SUB $2,$2,$1

B"111\_0000000011101", --34. E01D J 29

B"001\_000\_110\_0000000", --35. 2300 ADDI $6,$0,0

others => B"000\_010\_010\_010\_0\_101"); -- 0005 OR $2, $2, $2

1. **Schema Mips Pipeline**



1. **Corectitudinea in VHDL**

Programul respecta structura descrisa in laboratoare, genereaza bitstreamul insa nu l-am testat pe placuta. In cod exista toate elementele din laboratoare. Am si cateva warninguri care sper ca nu afecteaza corectitudinea programului.

1. **Tabelul cu descrierea registrilor pipeline**

|  |  |  |  |
| --- | --- | --- | --- |
| REG\_IF\_ID ( 31 – 0 ) | REG\_ID\_EX ( 82 – 0 ) | REG\_EX\_MEM ( 57 – 0 ) | REG\_MEM\_WB ( 36 – 0 ) |
| PC+1 ( 31 – 16 ) | RegDst(82) | B\_adress ( 57 – 42 ) | MemData ( 36 – 21 ) |
| Instruction ( 15 – 0 ) | Instr(9 downto 7)  (81 downto 79) | AluRes ( 41 – 26) | AluRes\_out ( 20 – 5) |
|  | RD1 ( 78 – 63 ) | RD2 ( 25 – 10) | WriteAdressReg ( 4 – 2) |
|  | RD2 ( 62 – 47) | Zero1 ( 9 ) | MemReg ( 1 ) |
|  | ExtImm ( 46 – 31 ) | Zero ( 8 ) | RegWrite ( 0 ) |
|  | PC+1 (30 – 15 ) | WriteAddressReg ( 7 – 5 ) |  |
|  | Func ( 14 – 12 ) | MemReg ( 4 ) |  |
|  | Sa ( 11 ) | RegWrite ( 3 ) |  |
|  | Instr(6 downto 4)  (10 – 8 ) | MemWrite ( 2 ) |  |
|  | RegWrite ( 7 ) | BranchLE ( 1 ) |  |
|  | AluSrc ( 6 ) | Branch ( 0 ) |  |
|  | AluOp ( 5 – 4 ) |  |  |
|  | MemWrite ( 3 ) |  |  |
|  | MemReg ( 2 ) |  |  |
|  | BranchLE ( 1 ) |  |  |
|  | Branch ( 0 ) |  |  |

1. **Testarea pe placuta**

Nu am testat pe placuta mips pipeline la momentul predarii.