**Raport MIPS Ciclu Unic**

**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. **Trasarea programului in assembly mips 16**

B"100\_000\_011\_0001000", -- 0. 8188 BEQ $0,$3,8

B"110\_101\_001\_0000001", -- 1. D481 BGE $5,$1,1

B"010\_110\_001\_0000100", -- 2. 5884 LW $1,4($6)

B"110\_101\_010\_0000001", -- 3. D501 BGE $5,$2,1

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

B"001\_000\_000\_0000001", -- 5. 2001 ADDI $0,$0,1

B"001\_110\_110\_0000100", -- 6. 3B04 ADDI $6,$6,4

B"010\_110\_101\_0000100", -- 7. 5A84 LW $5,4($6)

B"111\_0000000000000", -- 8. E000 J 0

B"110\_100\_010\_0000010", -- 9. D102 BGE $4,$2,2

B"000\_010\_001\_010\_0\_001", --10. 08A1 SUB $2,$2,$1

B"111\_0000000001001", --11. E009 J 9

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

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. **Descrierea testarii pe placuta Basys3**

Codul ar trebui sa afiseze urmatoarele in functie de ce switchuri sunt active:

• sw(7:5) = 000 - instructiunea

• sw(7:5) = 001 - următoare valoare a lui PC (PC + 1)

• sw(7:5) = 010 - read\_data\_1

• sw(7:5) = 011 – read\_data\_2

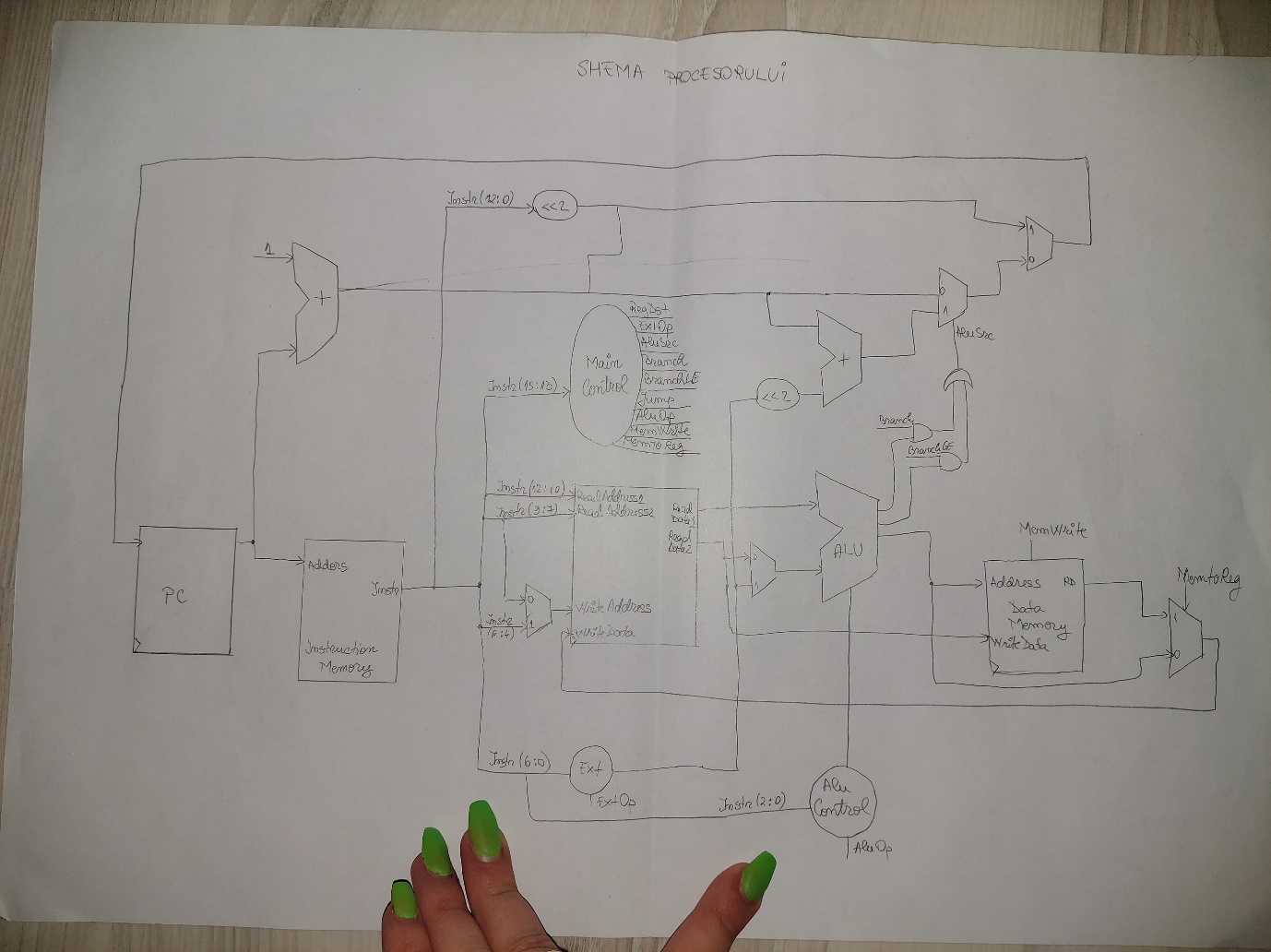
• sw(7:5) = 100 - Ext\_Imm

• sw(7:5) = 101 - ALURes

• sw(7:5) = 110 - MemData

• sw(7:5) = 111 – write\_data

1. **Schema microprocesorului**



1. **Tabelul de valori**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instructiune | Func Instr (2:0) | Opcode Instr (15:13) | ALUOp (1:0) | ALUCtrl (2:0) | RegDst | ExtOp | AluSrc | Branch | BranchGE | Jump | Mem Write | Memto Reg | Reg Write |
| Add | 000 | 000 | 00 | 000 | 1 | x | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Sub | 001 | 000 | 00 | 001 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Sll | 010 | 000 | 00 | 010 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Srl | 011 | 000 | 00 | 011 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| And | 100 | 000 | 00 | 100 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Or | 101 | 000 | 00 | 101 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Xor | 110 | 000 | 00 | 110 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Sra | 111 | 000 | 00 | 111 | 1 | x | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Addi | - | 001 | 01 | 000 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Lw | - | 010 | 01 | 000 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| Sw | - | 011 | 01 | 000 | X | 1 | 1 | 0 | 0 | 0 | 1 | X | 0 |
| Beq | - | 100 | 10 | 001 | X | 1 | 0 | 1 | 0 | 0 | 0 | X | 0 |
| Ori | - | 101 | 11 | 101 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Bge | - | 110 | 10 | 001 | X | X | 0 | 0 | 1 | 0 | 0 | X | 0 |
| J | - | 111 | xx | XXX | x | 1 | X | 0 | 0 | 1 | 0 | X | 1 |