****

**MIPS Processor Project**

**Under Supervision of:**

**Prof\Ashraf Salem**

**Eng\Ahmed Fathy**

**Made by G10**

**Contents:**

**Sec A: Introduction To our Project**

**A.1-Description of MIPS project**

**A.2-Description of Design**

**A.3-supported Instructions**

**A.4-Software Description**

**Sec B: Role of each member**

**B.1-Data of Members**

**B.2-Role of Each**

**Sec C: Test Cases**

**C.1-Tests in C and assembly**

**C.2-Expected output for each test**

**Sec A**

**A.1**

**MIPS project is an Implementation of MIPS processor in Verilog hardware description- language . the designed processor implementation supports different operations in Assembly such as:add,and,or,lw,sw,addi,ori,andi,xori,Jr,Jal,J,beq,etc… .**

**our project is synthesized**

**Sucessfully and without Any Warnings! .**

**A.2**

**Our design consists of :**

**1-RegisterFile :has 32 Registers two of them has special purpose :$sp,$ra ,$zero**

**And the others are general purpose registers**

**2-Instruction Memory: consists of 8192 instruction spaces 32 bits each ,capable of storing 32 bit instructions**

**3-Data memory :it's 32KB consists of 8192 word spaces each word is 32bits and is designed as follows: whenever memWirte is enabled it is then accessed and whenever memRead is enabled Data memory reveals the proposed address stored value ,otherwise data(memory output is zero)**

**4-ALU : it's designed to do arithmetic and logic operations such as Add,Xor, And ,Or ,Sll,Srl/a ..etc.. .**

**5-Control unit: we separated Alu control and Control unit so to make it easier while designing , we implemented it so that also func segment of the instruction is connected to support rare instructions such as jr**

**6-alu control: it's used to control the alu in it's different modes , its inputs are both Control unit and func segment**

**7-pc counter and muxes: we designed muxes as modules to make it easy when implementing CPU module to handle jump operations easily , pc counter monitors the current address of instruction and calculates the next pc address with an external ALU (adder)**

**A.3**

**Our MIPS processor supports all operations in assembly of Types I,R,J**

**I type such as:lw sw, addi,andi ,ori ,xori ,lui**

**R type :add,and,or xor,jr ,sll,sra,srl**

**J type: Jumps**

**Also MIPS supports Sign extension and zero extension .**

**A.4**

**We used python to write our software That's used for testing also we provided GUI so as to make it user friendly , we used tkinker as**

**Sec B**

**B.1 )Our team members:-**

**1-حازم محمد محمود النشار 1600471**

**2-جون عصام حلمي 1600460**

**3-عاصم كامل احمد 1600719**

**4-خالد عاطف عبد العزيز 1600504**

**5-حازم محمد زكي 1600469**

**6-طه محمد طه 1600714**

**7-احمد طه فكري طه 1600108**

**B.2)Role of each member:-**

**1-Hazem Nashar:-**

**Contributed in software program and GUI and also contributed in solving bugs in Verilog code**

**2-John Essam:-**

**Contributed in Verilog code implementation by implementing ALU and RegisterFile along side with testing and debugging the code**

**3-Khaled Atef:**

**Contributed in Verilog design by implementing Memory and Control unit along side with contribution in synthesis.**

**4-Assem Kamal:-**

**Contributed in Verilog design by implementing other modules :instruction memory ,alu control unit**

**And tested them (unit testing)**

**5-Hazem M.Zaki:-**

**Contributed in overall debugging of the whole project and added some features that solved all bugs present in code ,he contributed in synthesis.**

**6-Taha Mohammed :-**

**Contributed in Verilog design by implementing muxes and some other features inside mipsCPU module and contributed in unit testing for memory and ALU**

**7-Ahmed Taha Fekry:-**

**Contributed in writing testcases finding out expected outputs of each test case, Wrote the report .**

**Sec C**

**Test Case 1:**

**Mips code:-**

**addi $s0,$0,1**

**addi $s1,$0,$0**

**addi $t0,$0,128**

**while:beq $s0,$t0,done**

**sll $s0,$s0,1**

**addi $s1,$s1,1**

**j while**

**$t0->128#**

**$s0->x#**

**$s1->y#**

**C code:-**

**Int x=1,y=0;**

**While(x!=128){**

**y=y+1;**

**x=x\*2;**

**}**

**Test Case 2**

**Mips Code:-**

**Addi $s2,$0,1**

**Add $s1,$0,$0**

**Addi $s3,$0,10**

**For:slt $t0,$s3,$s2**

**Bne $t0,$0,EXIT**

**Sll $t1,$s2,2**

**Add $s1,$s1,$t1**

**Addi $s2,$s2,1**

**J for**

**EXIT:**

**C code equivalent:-**

**$s2->x**

**$s1->y**

**Int x,y=0;**

**For(x=1;x<10;x++){**

**y=y+2\*x;**

**y=y+1;**

**}**

**Test Case 3**

**Main:addi $a1,$0,5**

**Addi $t0,$0,36**

**Jal func**

**Addi $a0,$s4,$0**

**Addi $s5,$a0,400**

**J exit**

**Func:addi $t1,$0,6**

**Add $s0,$t0,$0**

**Loop:slt $t5,$0,$s0**

**Beq $t5,$0,return**

**Sub $s0,$s0,$t1**

**Add $s1,$s1,$a1**

**J loop**

**Return:add $s4,$s1,$0**

**Jr $ra**

**Exit:**

**C equivalent code:**

**Int func(int x){**

**Int f=6,y=x,j=0;**

**While(y>0)**

**{**

**Y=y-f;**

**J=j+y;**

**}**

**Return j;**

**}**