**计算机组成原理**

**课程设计报告**

**学 号\_\_\_\_\_\_\_\_\_18021006\_\_\_\_\_\_\_\_\_\_\_**

**姓 名\_\_\_\_\_\_\_\_\_\_吴天鹤\_\_\_\_\_\_\_\_\_\_\_\_**

**指导教师\_\_\_\_\_\_\_\_\_\_朱文军\_\_\_\_\_\_\_\_\_\_\_\_**

**提交日期\_\_\_\_\_\_\_\_\_\_2020.7.8\_\_\_\_\_\_\_\_\_\_**

**成绩评价表**

|  |  |  |
| --- | --- | --- |
| **报告内容** | **报告结构** | **报告最终成绩** |
| **□丰富正确**  **□基本正确**  **□有一些问题**  **□问题很大** | **□完全符合要求**  **□基本符合要求**  **□有比较多的缺陷**  **□完全不符合要求** |  |
| **报告与Project功能一致性** | **报告图表** | **总体评价** |
| **□完全一致**  **□基本一致**  **□基本不一致** | **□符合规范**  **□基本符合规范**  **□有一些错误**  **□完全不正确** |  |

**教师签字:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**目录**

[1. 总体数据通路结构设计图 3](#_Toc45130610)

[2. 各个模块定义 3](#_Toc45130611)

[2.1 PC模块定义 3](#_Toc45130612)

[2.2 GPR模块定义 4](#_Toc45130613)

[2.3 ALU模块定义 5](#_Toc45130614)

[2.4 EXT模块定义 5](#_Toc45130615)

[2.5 DM模块定义 6](#_Toc45130616)

[2.6 Controller模块定义 6](#_Toc45130617)

[2.7 NPC模块定义 7](#_Toc45130618)

[2.8 IM模块定义 8](#_Toc45130619)

[2.9 IR模块定义 9](#_Toc45130620)

[2.10 A，B模块定义 9](#_Toc45130621)

[2.11 ALUOut模块定义 9](#_Toc45130622)

[2.12 DR模块定义 10](#_Toc45130623)

[2.13 SB\_module模块定义 11](#_Toc45130624)

[2.14 LB\_module模块定义 11](#_Toc45130625)

[3. 机器指令描述 12](#_Toc45130626)

[ADDU 12](#_Toc45130627)

[SUBU 12](#_Toc45130628)

[ORI 12](#_Toc45130629)

[LW 13](#_Toc45130630)

[SW 13](#_Toc45130631)

[LUI 13](#_Toc45130632)

[BEQ 13](#_Toc45130633)

[J: 13](#_Toc45130634)

[SLT： 13](#_Toc45130635)

[Addi: 13](#_Toc45130636)

[ADDIU: 13](#_Toc45130637)

[JAL: 13](#_Toc45130638)

[JR: 13](#_Toc45130639)

[SB： 14](#_Toc45130640)

[LB： 14](#_Toc45130641)

[4. 测试程序 14](#_Toc45130642)

[5. 测试结果与说明 16](#_Toc45130643)

[6. 收获、体会、总结 20](#_Toc45130644)

# 总体数据通路结构设计图

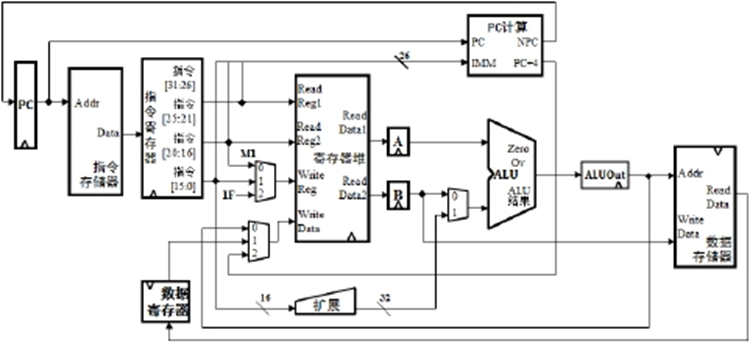


图1 总体数据通路结构设计图

# 各个模块定义

## 2.1 PC模块定义

1. 基本描述

PC主要功能是完成取指令功能。PC内部包括一个32位寄存器，根据时钟沿的上升进行PC+4.

1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| npc\_in | I | 从npc输入下一条指令的地址 |
| pc\_out | O | 输出下一条指令的地址 |
| clk | I | 时钟信号 |
| reset | I | 复位信号。  1：复位  0：无效 |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 输出下一条指令的地址 | 当复位信号有效时，PC被设置为0x00003000。输出由NPC传进来的下一条指令。 |
| 2 | 取指令 | 根据PC从NPC |

## 2.2 GPR模块定义

1. 基本描述

GPR是通用寄存器堆，里面有32个寄存器用来存储值，还有相应的DXM进行rt和rd寄存器的选取。主要功能是向特定的寄存器内写入具体的值，从特定的寄存器中取出值，传递给ALU，进行相应的指令功能的操作。

1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| ra | I | 接收当前指令的25-21位，选择对应的寄存器，取出对应的值 |
| rb | I | 接收当前指令的20-16位，选择对应的寄存器，取出对应的值 |
| rw | I | 选择对应的寄存器，作为结果存储的寄存器 |
| clk | I | 时钟信号 |
| reset | I | 复位信号。  1：复位  0：无效 |
| busW | I | 接收通过ALU之后的值存入由rw决定的寄存器中 |
| busA | O | 输出对应的寄存器内部的值 |
| busB | O | 输出对应的寄存器内部的值 |
| Overflow | O | 输出是否溢出信号 |
| Clk | I | 时钟信号 |
| Rst | I | 复位信号 |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 取值 | 根据rs, rt从特定寄存器取值 |
| 2 | 存值 | 根据rt或rd向特定寄存器写入值 |
| 3 | 向30号寄存器写1 | Addi指令溢出判断 |

## 2.3 ALU模块定义

(1) 基本描述

ALU是算术逻辑单元，在ALU中可以实现加、减，或运算。对应addu, subu, ori指令，同时能判定相减后是否为0，如果为0且指令为beq指令，则根据beq后的指令进行跳转。

(2) 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| busA | I | 输入第一个值busA |
| busB | I | 输入第二个值busB |
| ALUctr | I | 选择信号，对应的只输出busB(lui指令)，输出两个值相加，输出两个数相减，输出与立即数或运算 |
| zero | O | 判断两数相减是否为0 |
| out | O | 输出运算后的结果 |
| Overflow | O | 输出是否溢出信号 |
| Addi\_sel | I | Addi检测标识 |

(3) 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 相加 | 实现传入两数相加（要经过ALUSrc选择） |
| 2 | 相减 | 实现传入两数相减（要经过ALUSrc选择） |
| 3 | 与立即数或 | 实现与32位立即数相或（要经过ALUSrc选择） |

## 2.4 EXT模块定义

(1) 基本描述

EXT是扩展模块。该模块里能够实现16位到32位的0扩展，符号扩展，还有低16位0的扩展，高位为该立即数的功能。

(2) 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Imm16 | I | 输入16位立即数 |
| Extop | I | 选择信号，选择进行高位0扩展，符号扩展，低16位0扩展高16位为立即数即lui指令 |
| Output32 | O | 输出扩展后的结果 |

(3) 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 高位0扩展 | 将16位立即数进行高位0扩展 |
| 2 | 符号扩展 | 将16位立即数进行符号扩展 |
| 3 | 高位为立即数，低16位为0 | lui指令操作 |

## 2.5 DM模块定义

(1) 基本描述

DM是数据存储器模块。该模块是向具体的数据段地址存储相应的数据，和从相应的地址取出数值。

(2) 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| din | I | 输入rt对应寄存器的值 |
| Addr | I | 输入对应的地址 |
| dout | O | 输出从寄存器取出的值 |
| clk | I | 时钟信号 |
| we | I | 存储使能信号 |

(3) 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 输入地址 | 选择数据存储器具体的地址 |
| 2 | 存储数据 | 向相应的地址里写入数据 |
| 3 | 取出数据 | 从相应的地址里取出数据 |

## 2.6 Controller模块定义

(1) 基本描述

Controller是控制器模块。该模块是控制数据通路里选择信号的一个模块。通过对opcode和funct进行与或操作来实现一条指令输入，哪些选择信号打开的功能。

(2) 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| RegDst | O | 对rt和rd寄存器进行选择 |
| ALUSrc | O | 选择busB和16位立即数扩展后的值 |
| MemtoReg | O | 选择DM和ALU计算后的值 |
| RegWrite | O | 寄存器堆写使能信号 |
| MemWrite | O | 数据存储器写使能 |
| nPC\_sel | O | 检测Beq指令 |
| Extop | O | 选择扩展的方式 |
| jsel | O | J指令使能信号 |
| ALUctr | O | ALU执行操作选择信号 |
| Opcode\_in | I | 输入指令的opcode码 |
| Funct\_in | I | 输入指令的funct码 |
| Jr\_sel | O | 判断jr |
| Jal\_sel | O | 判断JAL |
| Addi\_sel | O | 判断ADDI |
| PCWr | O | PC写使能 |
| IRWr | O | IR写使能 |
| LB\_sel | O | 根据current判断LB |
| Sb\_sel | O | 根据current判断SB |

(3) 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 选择 | 选择相应的功能操作，当某些指令执行时相应的选择信号进行选择 |
| 2 | 使能信号 | 对相应的模块的功能进行使能判断，即什么时候该模块执行相应操作 |

## NPC模块定义

(1)基本描述

NPC主要功能是完成计算下一条指令功能。其中可以判断beq和j和jr指令，进行相应计算并选择输出。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| pc\_in | I | 从PC传进来值 |
| jal\_sel | I | Jal信号 |
| jr\_val | I | Jr的值 |
| zero | I | 传进BEQ的等于信号 |
| j\_sel | I | J指令信号 |
| j26\_in | I | J26值 |
| jr\_sel | I | Jr信号 |
| pc4\_out | O | Pc+4\_out，针对Jal指令 |
| npc\_out | O | Npc\_out |
| imm\_in | I | 26位输入 |
| npc\_sel | I | 输出下一条指令的地址 |
| clk | I | 时钟信号 |
| reset | I | 复位信号。  1：复位  0：无效 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 计算 | 计算下一条指令 |
| 2 | 取指令 | 根据PC从NPC |

## IM模块定义

(1)基本描述

IM用来存储指令的信息。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Addr | I | 从PC传进来值 |
| Dout | O | Jal信号 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储测试程序的指令 |
| 2 | 输出指令 | 输出指令 |

## IR模块定义

(1)基本描述

IR用来存做IM的锁存器来分开5个阶段。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Im\_in | I | 从PC传进来值 |
| IRWr | I | 写使能 |
| Ir\_out | O | Jal信号 |
| Clk | I | 时钟信号 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储测试程序的指令 |
| 2 | 输出指令 | 输出指令 |

## 2.10 A，B模块定义

(1)基本描述

A用来存做busA的锁存器来分开5个阶段。B用来存做busB的锁存器来分开5个阶段。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| read\_data1\_in | I | 从GPR传来rs / rt的值 |
| read\_data1\_out | O | 传出值 |
| Clk | I | 时钟信号 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储GPR传来rs / rt的值 |
| 2 | 输出值 | 输出GPR传来rs / rt的值 |

## 2.11 ALUOut模块定义

(1)基本描述

ALUOut用来存做ALU的值的锁存器来分开5个阶段。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| read\_data1\_in | I | 从GPR传来rs / rt的值 |
| read\_data1\_out | O | 传出值 |
| Clk | I | 时钟信号 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储ALU传进来的值 |
| 2 | 输出值 | 输出ALU传进来的值 |

## 2.12 DR模块定义

(1)基本描述

DR用来存做LB模块取出的值的锁存器来分开5个阶段。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| dm\_in | I | 存LB模块出来的值 |
| dr\_out | O | 传出值 |
| Clk | I | 时钟信号 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储LB模块传进来的值 |
| 2 | 输出值 | 输出LB模块传进来的值 |

## 2.13 SB\_module模块定义

(1)基本描述

SB\_module用来实现sb指令，向dm中写一个字节。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| dm\_in | I | 输入DM模块出来的值 |
| B\_reg\_in | I | 输入B寄存器传的值 |
| sb\_sel | I | Sb使能 |
| sb\_out | O | 输出值 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储DM模块传进来的值 |
| 2 | 输出值 | 输出拼接后的值 |

## 2.14 LB\_module模块定义

(1)基本描述

SB\_module用来实现sb指令，向dm中写一个字节。

(2)模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| dm\_in | I | 输入DM模块出来的值 |
| LB\_sel | I | LB使能 |
| LB\_out | O | 输出值 |

(3)功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储 | 存储DM模块传进来的值 |
| 2 | 输出值 | 输出选择并符号拓展后的值 |

# 机器指令描述

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 助记符 | Opcode | rs | rt | Rd | shamt | Funct |
| **ADDU** | 000000 | 5位 | 5位 | 5位 | 00000 | 100001 |
| **SUBU** | 000000 | 5位 | 5位 | 5位 | 00000 | 100011 |
| **SLT** | 000000 | 5位 | 5位 | 5位 | 00000 | 101010 |
| **JR** | 000000 | 5位 | 00000 | 00000 | Hint | 001000 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 助记符 | Opcode | rs | rt | immediate |
| **ORI** | 001101 | 5位 | 5位 | 16位 |
| **LW** | 100011 | 5位 | 5位 | 16位 |
| **SW** | 101011 | 5位 | 5位 | 16位 |
| **BEQ** | 000100 | 5位 | 5位 | 16位 |
| **LUI** | 001111 | 5位 | 5位 | 16位 |
| **ADDI** | 001000 | 5位 | 5位 | 16位 |
| **Addiu** | 001001 | 5位 | 5位 | 16位 |
| **LB** | 100000 | 5位 | 5位 | 16位 |
| **SB** | 101000 | 5位 | 5位 | 16位 |
| **J** | 000010 | 26位instr\_index | | |
| **JAL** | 000011 | Insrt\_index | | |

ADDU**:**

实现两个32位数相加，不考虑溢出

GPR[rd] <- GPR[rs] + GPR[rt]

SUBU**:**

实现两个32位数相减，不考虑溢出

GPR[rd] <- GPR[rs] - GPR[rt]

ORI**：**

与一个16位立即数做完高位0扩展后进行或操作

GPR[rt] <- GPR[rs] or immediate

LW**:**

从数据存储器内提取相应的值存入寄存器中

GPR[rt] <- memory[GPR[base] + offset]

SW**:**

向数据存储器内相应的地址写入值

Memory[GPR[base] + offset] <- GPR[rt]

LUI**:**

将一个16位常量赋值给寄存器中的高16位，低16位清0

GPR[rt] <- immediate || 016

BEQ**:**

比较rs对应寄存器和rt对应寄存器的内容，如果相等，则跳转至标号位置，不相等则继续执行PC + 4

If GPR[rs] = GPR[rt] then branch

J:

无条件跳转至256MB的区间内的任意指令位置

SLT：

GPR[rd] 🡨 (GPR[rs] < GPR[rt])

去记录是否小于

Addi:

GPR[rt] = GPR[rs] + immeditate

ADDIU:

GPR[rt] = GPR[rs] + immediate

JAL:

跳转并链接，把PC+4的值传入$ra

JR:

根据$ra的值进行跳转

SB：

向dm中写入一个字节

LB：

从dm中提取一个字节

# 测试程序

ori $16, $0, 1

ori $17, $0, 3

ori $8, $0, 1

ori $12, $0,0xabab

lui $13, 10

start:addu $4, $0,$16

addu $5, $0,$8

jal newadd

addu $16, $0, $2

subu $17,$17,$8

beq $16, $17, start

ori $8, $0,4

addiu $24,$0,0x7fffffff

addiu $9,$24,3

addiu $10,$24,5

addu $0,$0,$0

#addi $22,$24,6

start2:sw $9, -4($8)

sw $1, 0($8)

lb $14, 3($8)

sb $12,7($8)

lw $15,4($8)

sb $4, -3($8)

lb $18, -1($8)

addu $4,$0,$8

addu $5,$0,$9

jal newadd

slt $25,$10,$8

beq $25, $0,end2

slt $20,$12,$4

beq $20, $0, end1

lui $12, 65535

end1:ori $0, $0,1

lui $19, 0xefef

addiu $3,$0,0xababcdcd

start3:addiu $4, $3, 2

addi $23, $3, 5

jal newadd

addu $8, $0, $2

addu $4, $0, $8

addu $5, $0, $9

jal newadd

addu $9, $0, $2

addu $9, $8, $0

lui $10, 0x69

beq $8, $9, start4

beq $0, $0, start3

start4:j end

newadd:addu $2, $4, $5

addi $0,$12,0x1234

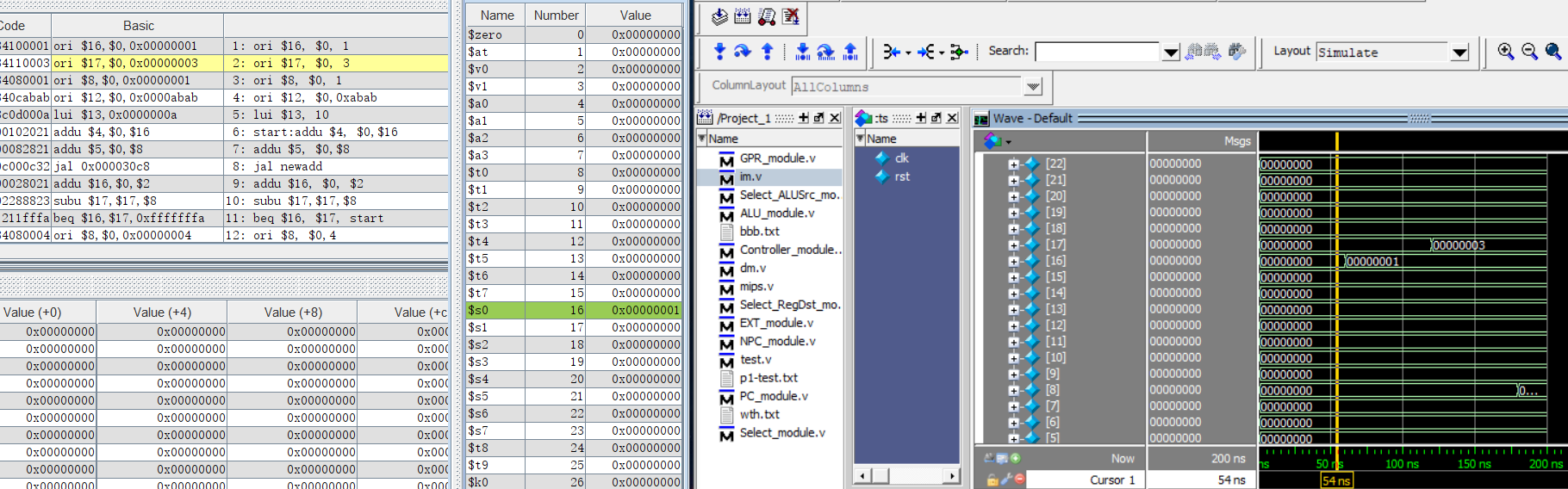
jr $31

end2:addi $26,$0,0x5678

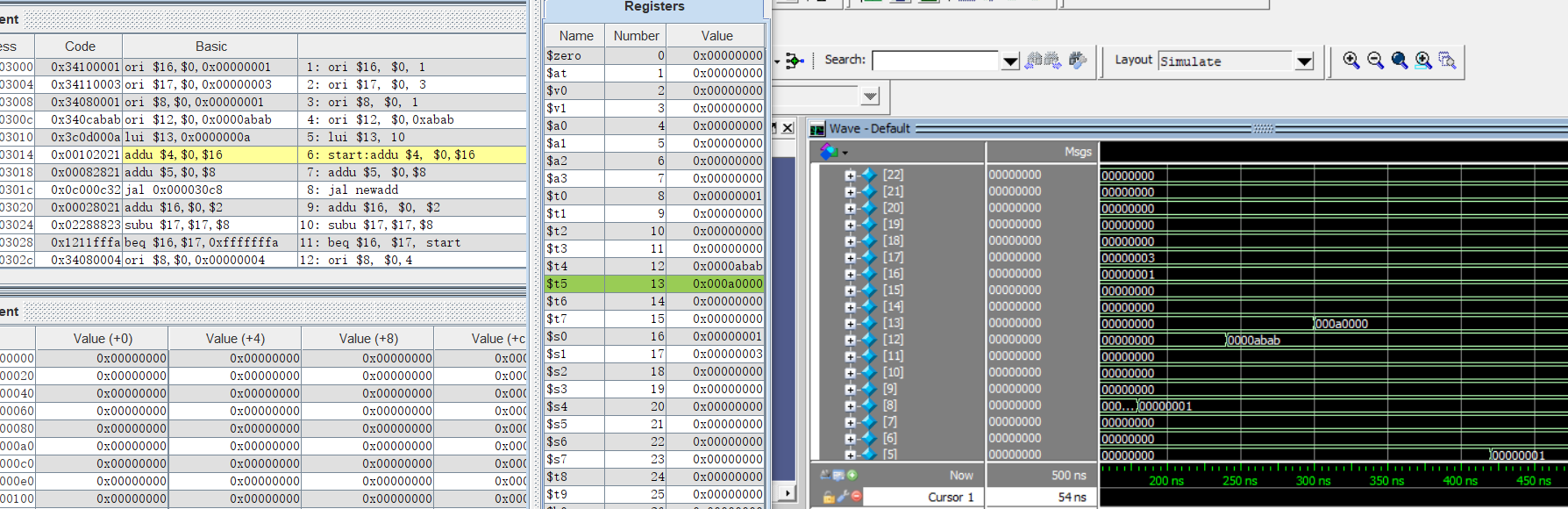
end:

# 测试结果与说明

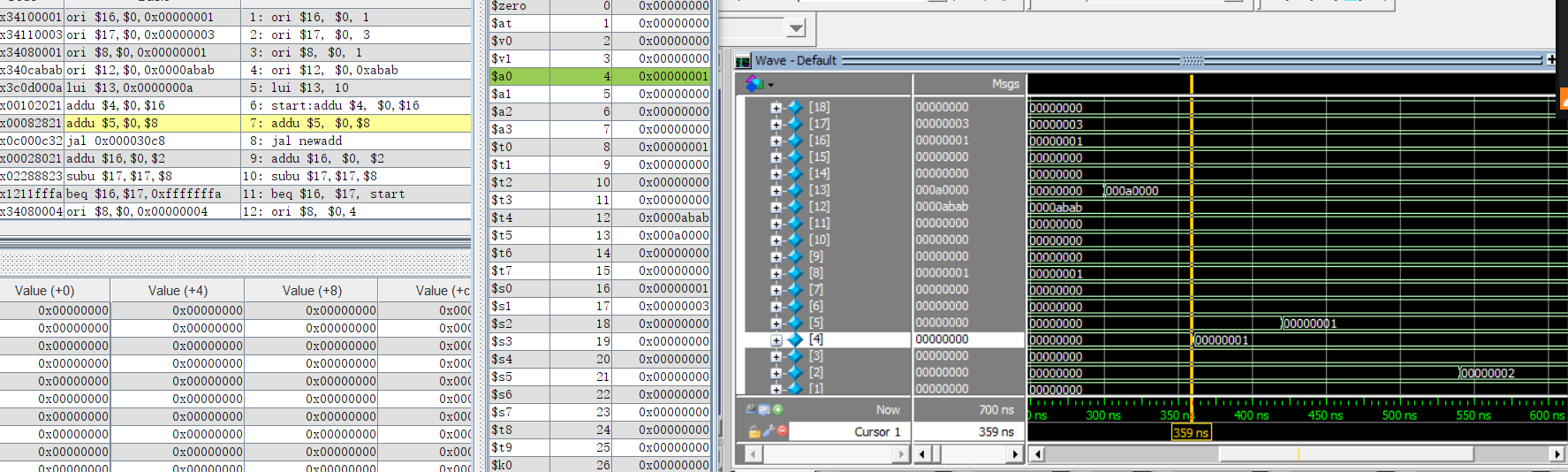
**Ori:**



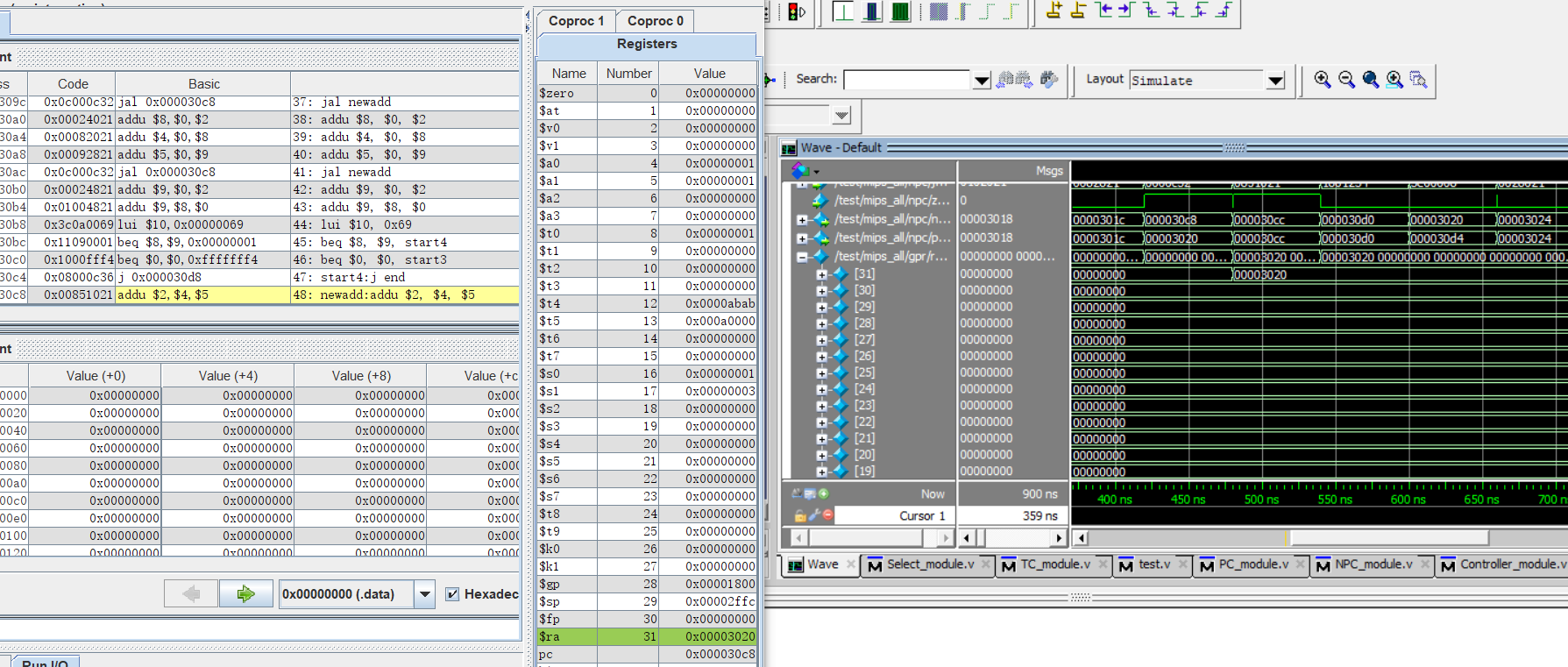
**Lui:**



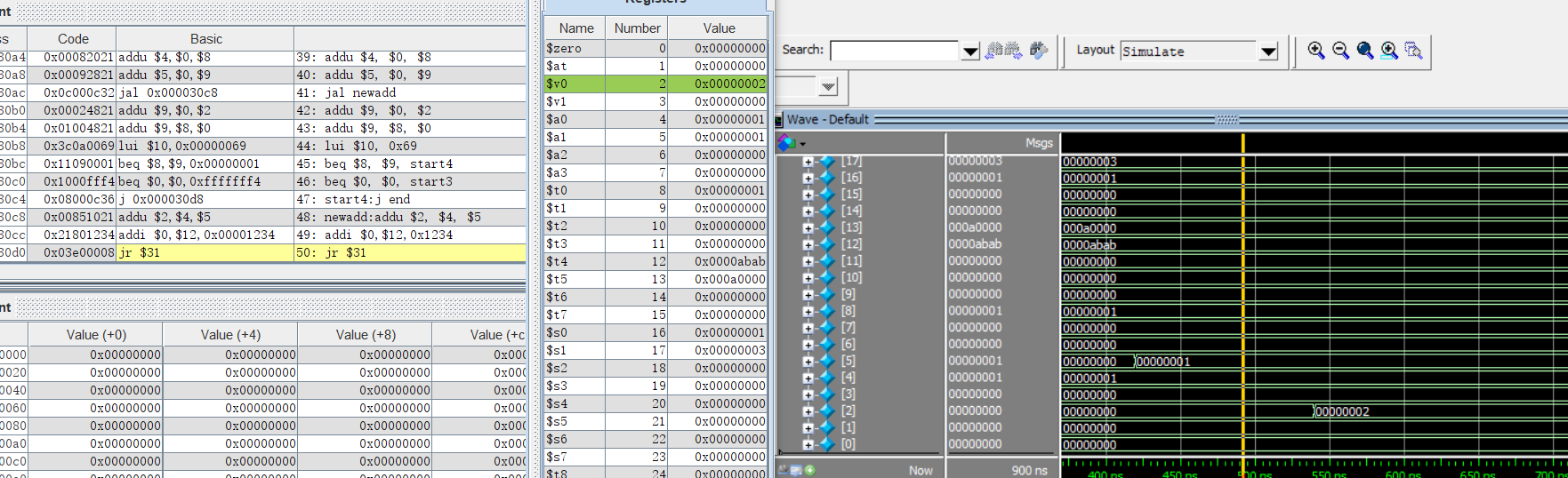
**Addu:**



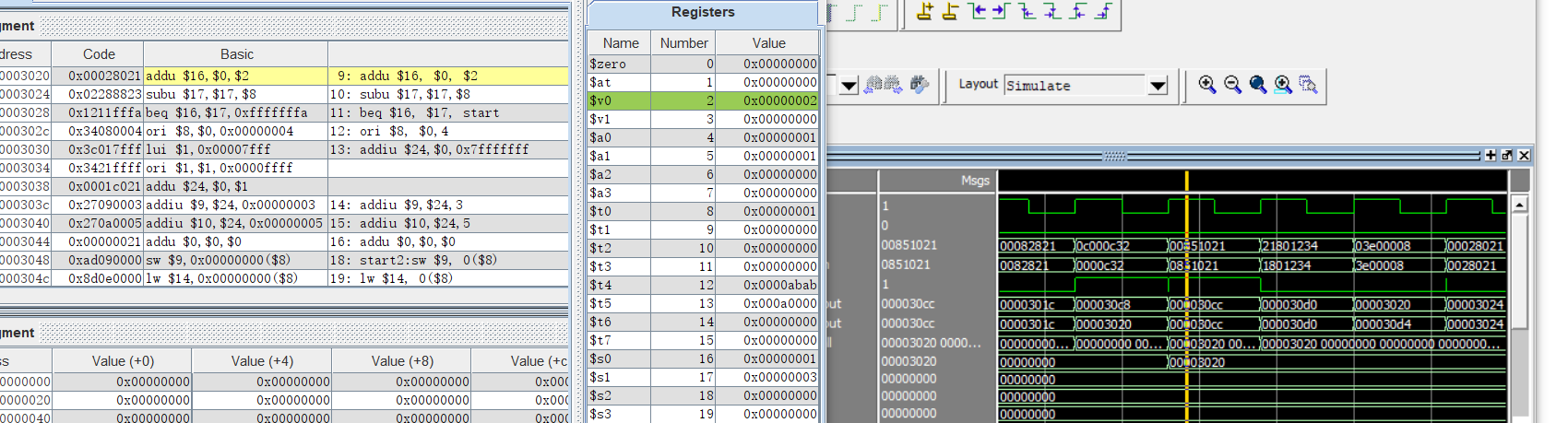
**JAL:**



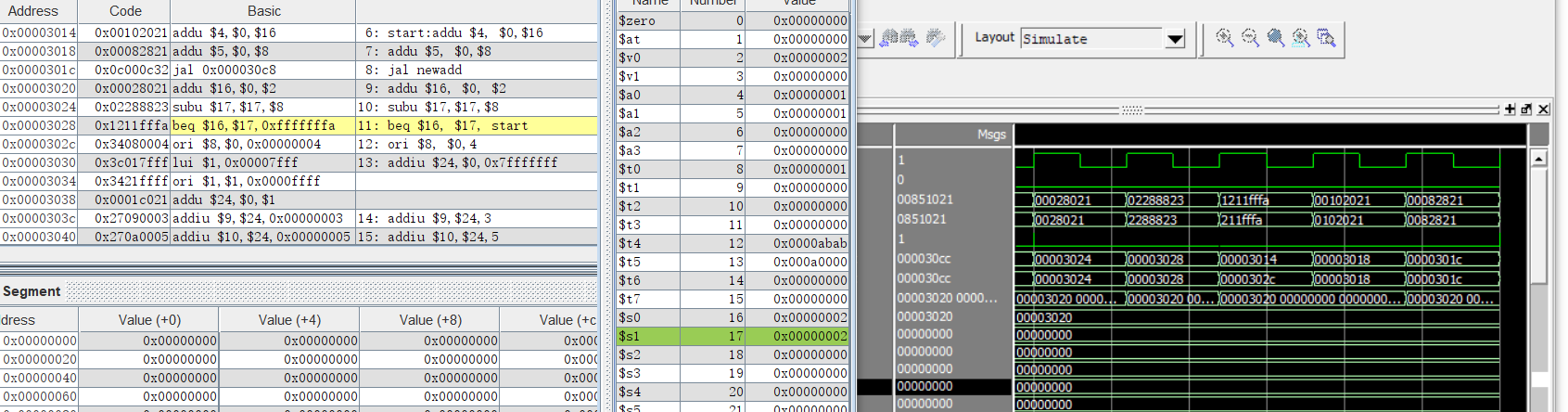
**Addi:**



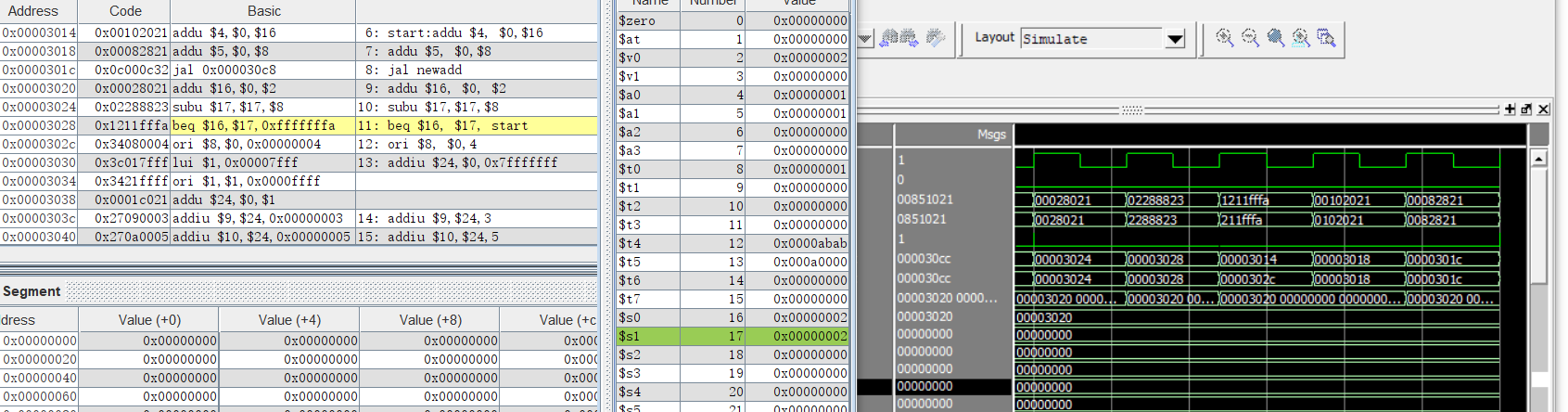
**JR:**

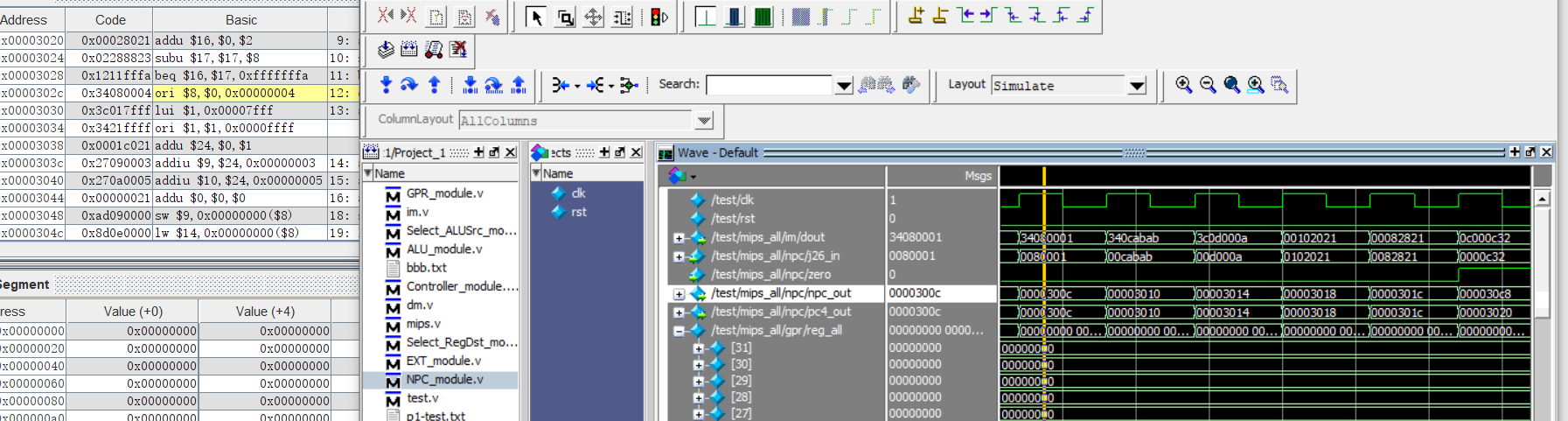


**Subu:**

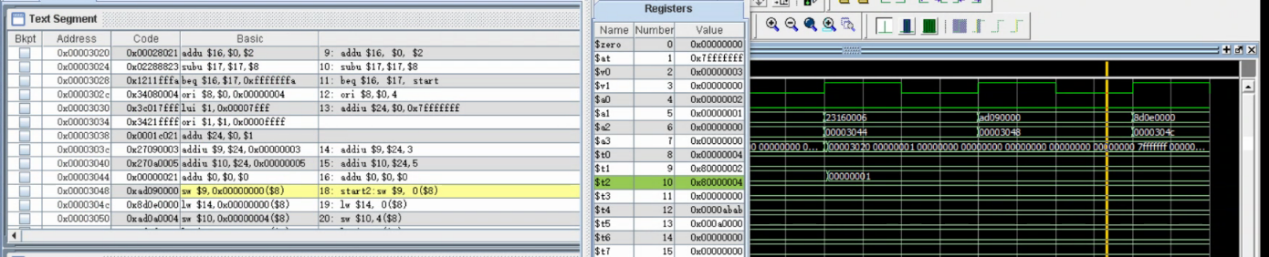


**Beq:**

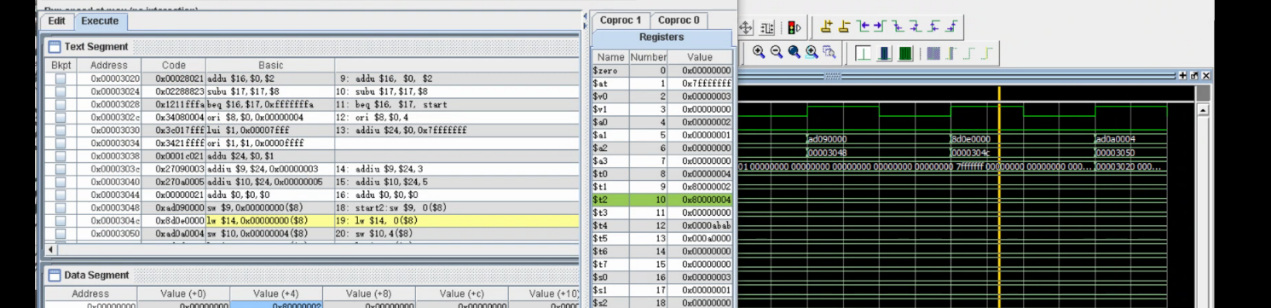




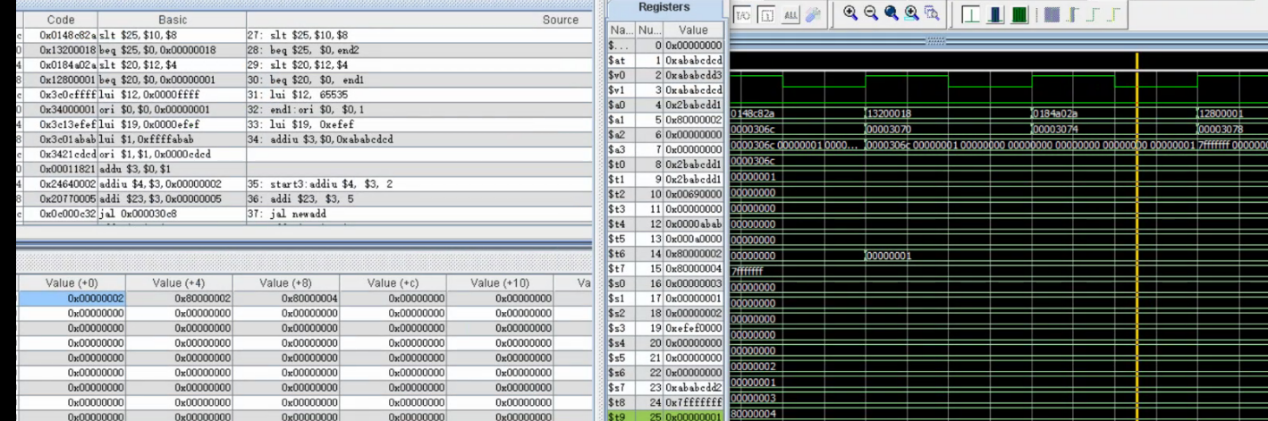
**Sw:**



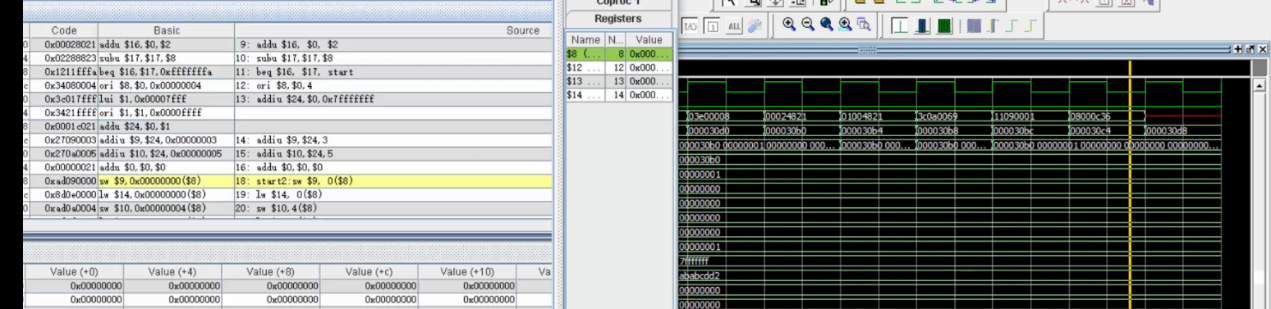
**Lw:**



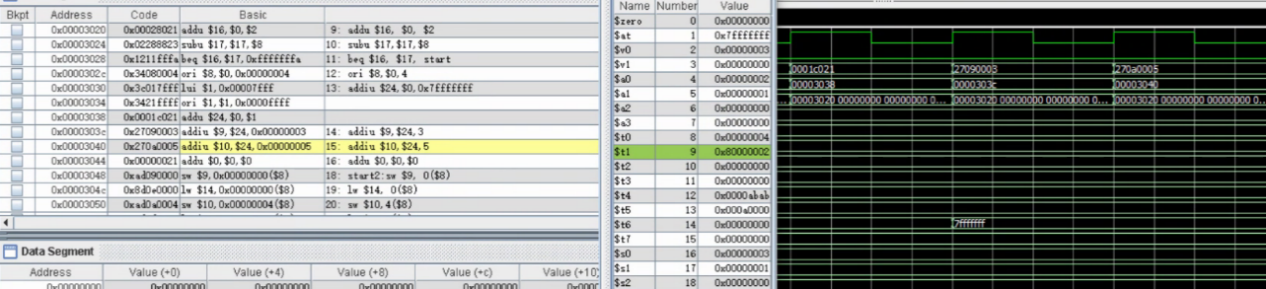
**Slt:**



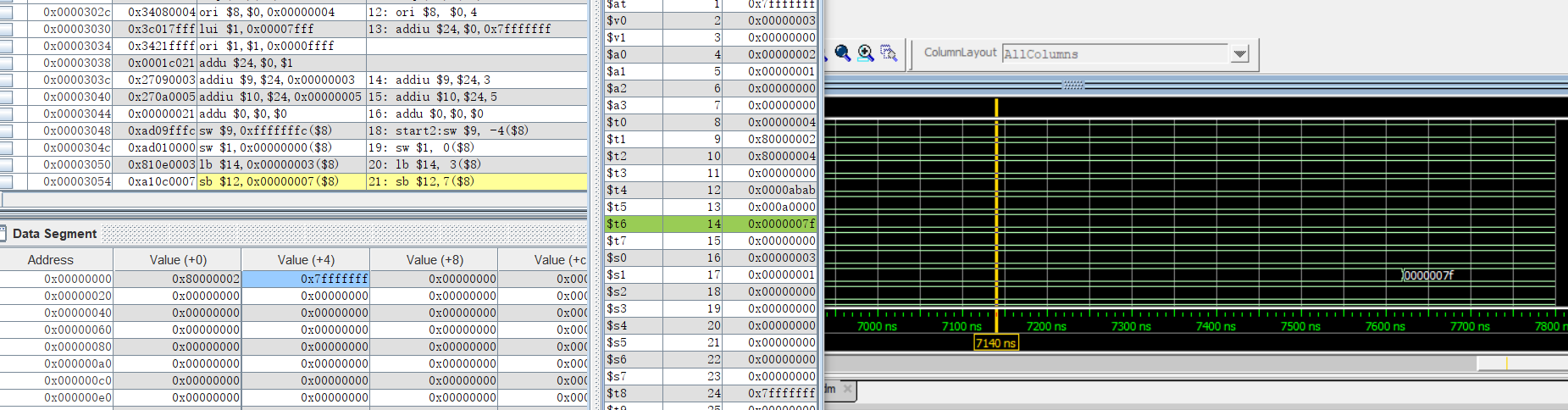
**J:**



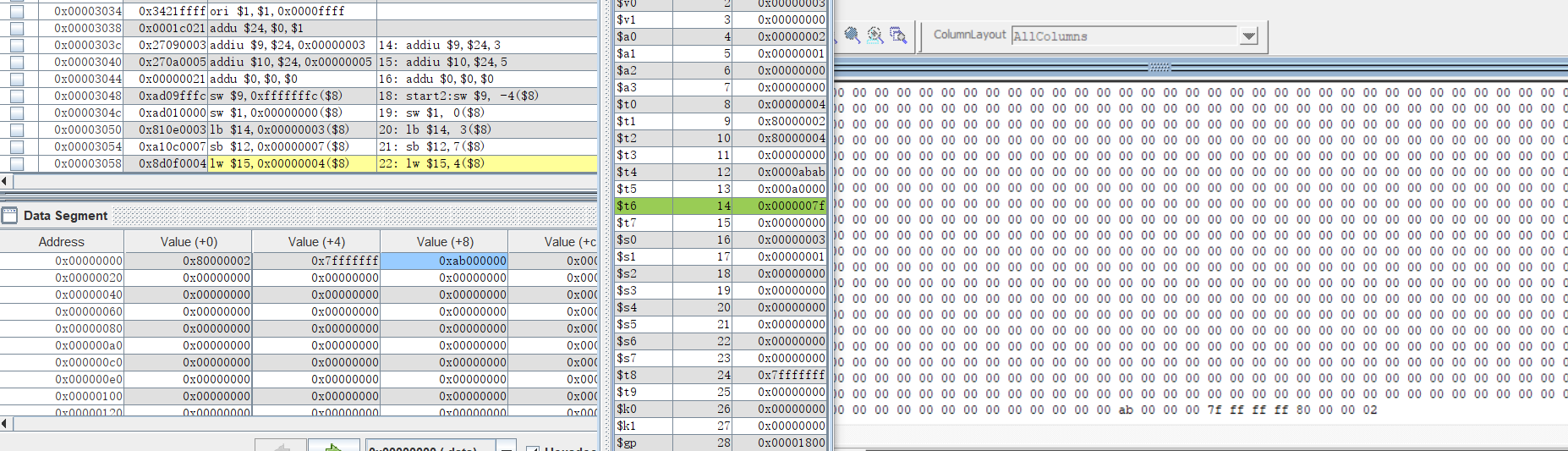
**ADDIU:**



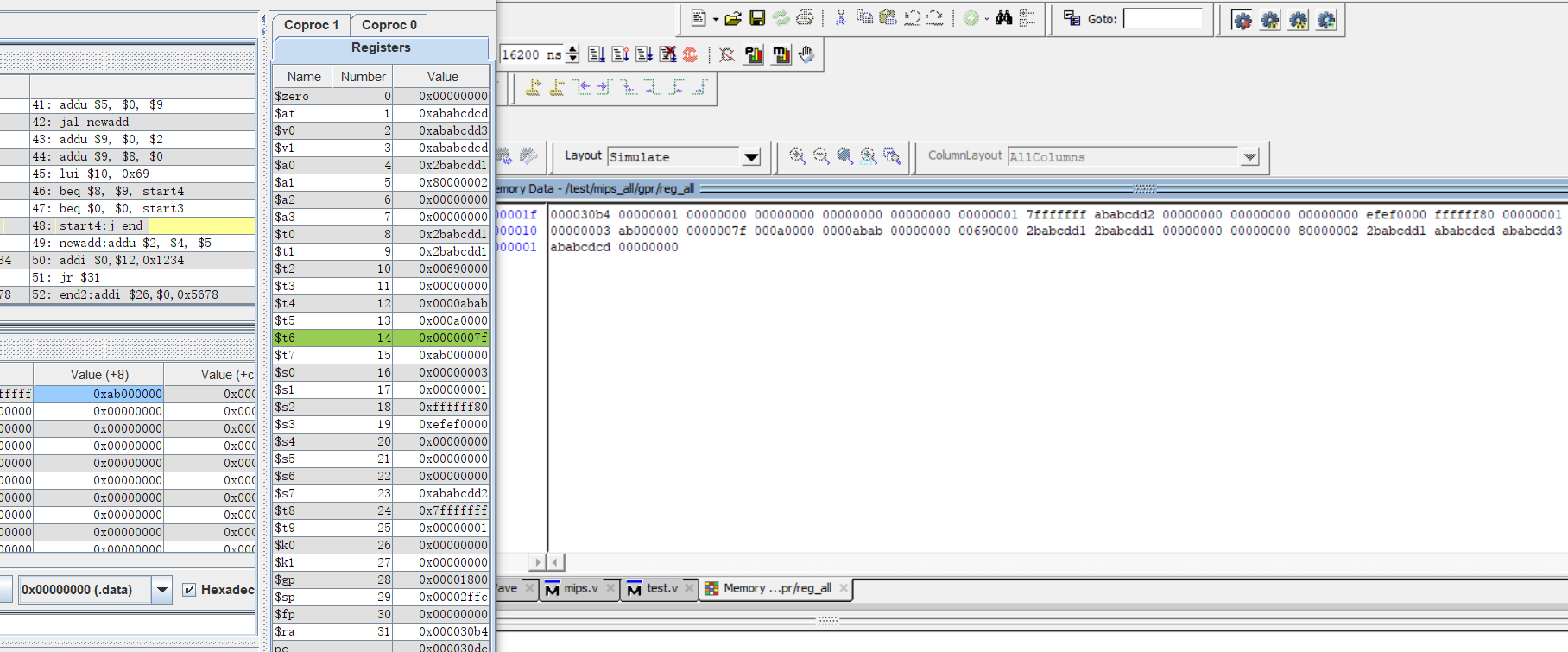
**Lb:**



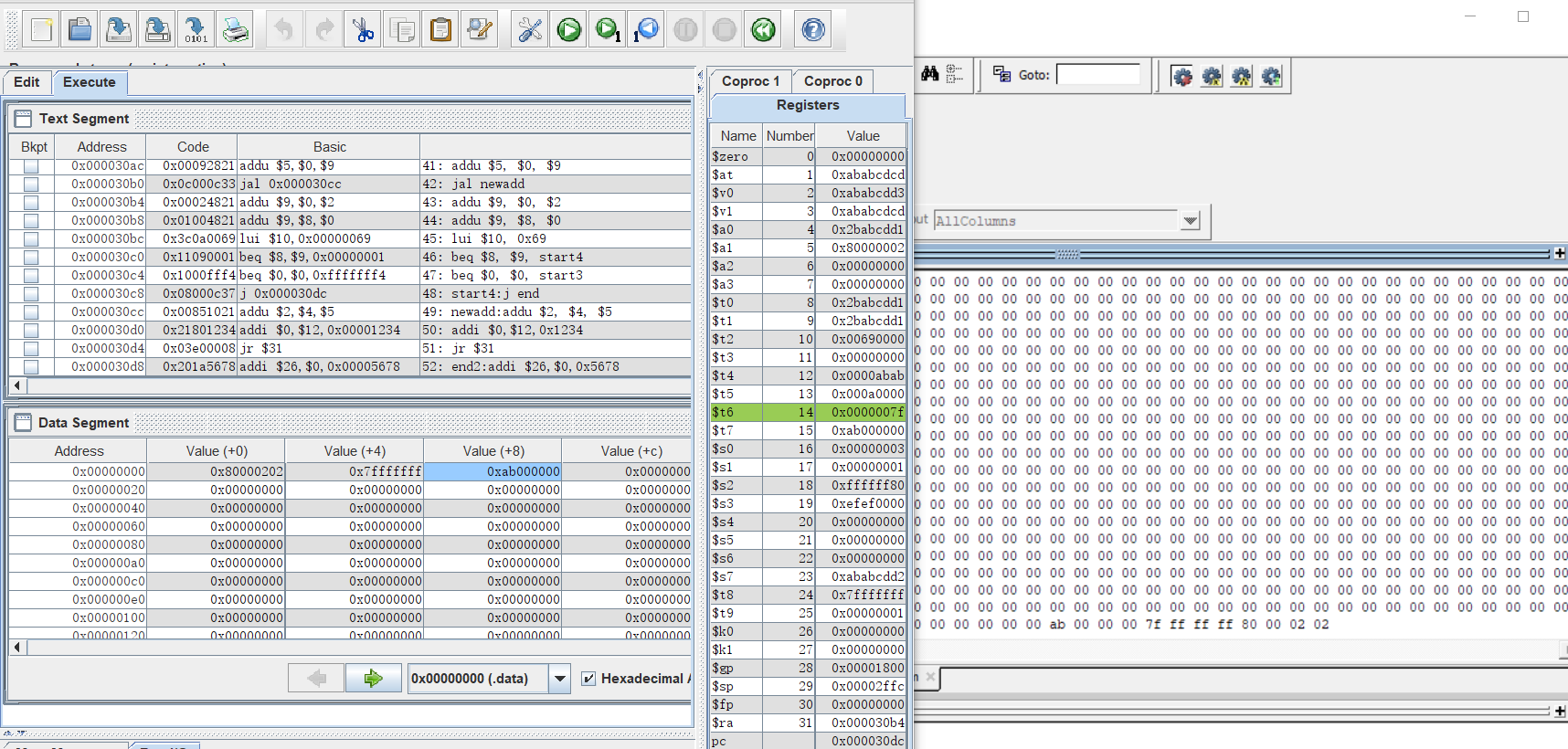
**Sb:**

****

**GPR：**

****

**DM：**

****

**每一项都与MARS中对应结果相吻合，故程序正确。**

# 收获、体会、总结

这次多周期的课设，让我增加了对多周期更深刻的认识，为P3课设打下了基础。感谢朱老师熬夜帮我调试，也感谢我的刘文星熬到4点多帮我调试，请老师酌情给刘文星加分。