|  |
| --- |
|  |
| 硬件课程设计报告 |
| RISC模型机设计 |
|  |

|  |
| --- |
|  |

目 录

1. 设计目的……..............................................................3
2. 设计过程. …….............................................................3

2.1指令集设计…….....................................................3

2.2模块设计……..........................................................3

2.2.1寄存器堆…….................................................3

2.2.2运算器…….....................................................5

2.2.3存储器…….....................................................7

2.2.4其他模块…….................................................8

2.2.5控制器…….....................................................9

2.3整机调试……..........................................................12

2.3.1测试程序结果…….........................................12

2.3.2仿真结果…….................................................13

2.3.3下载后执行结果…….....................................13

1. 实验总结心得体会……................................................14
2. 设计目的
3. 掌握时序的工作方式。
4. 掌握CPU的时序产生器。
5. 掌握ROM,RAM的设置以及.mif文件的使用。
6. 掌握PC加1计数方法。
7. 掌握指令寄存器(IR)的功能。

6、掌握微程序控制器的工作原理和具体实现方法。

7、仿照CISC模型机设计出RISC模型机并学习如何修改指令系统。

8、设计模型机并测试。

1. 设计过程

选定位数为8位，从底层开始设计数据寄存器，寄存器堆，ALU运算器，微程序控制器，RAM存储器，时序发生器，PC程序计数器并将它们封装成元器件用于顶层图的设计。

2.1指令集设计

指令格式：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位： | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 功能： | OP-CODE | | | | RS | | RD | |

指令集：

|  |  |  |
| --- | --- | --- |
| 助记符 | 机器指令码 | 功能说明 |
| IN0 | 10H | SW->R0 |
| IN1 | 24H | SW->R1 |
| ADD | 34H | R0 + R1 -> R0 |
| SUB | 44H | R0 + R1 -> R0 |
| OUT | 50H | R0 -> BUS |
| JMP | 40H | R0 -> PC |

2.2 模块设计

**2.2.1寄存器堆**

寄存器：

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

ENTITY REG IS

PORT(GATE:IN STD\_LOGIC;

CLK:IN STD\_LOGIC;

DIN: IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

DOUT:OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0));

END REG;

ARCHITECTURE BEHAV OF REG IS

BEGIN

PROCESS(GATE,DIN)

BEGIN

IF CLK'EVENT AND CLK='1' THEN

IF GATE='1' THEN DOUT<=DIN;

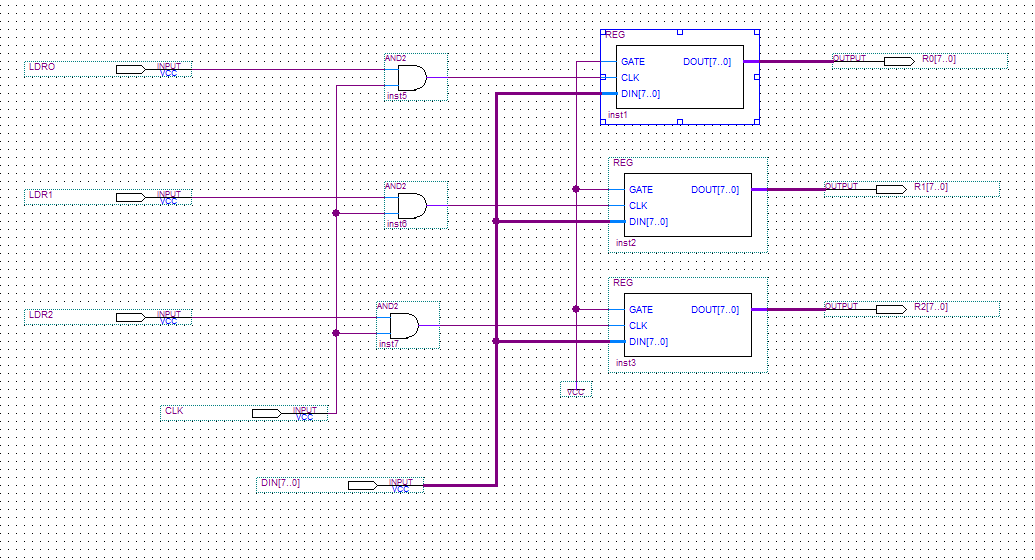
END IF;

END IF;

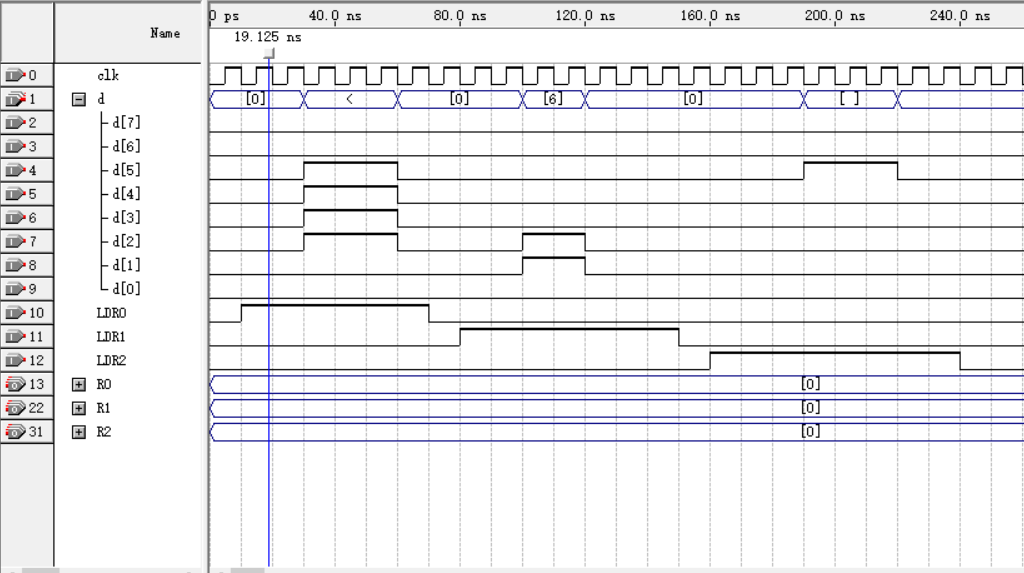
END PROCESS;

END BEHAV;

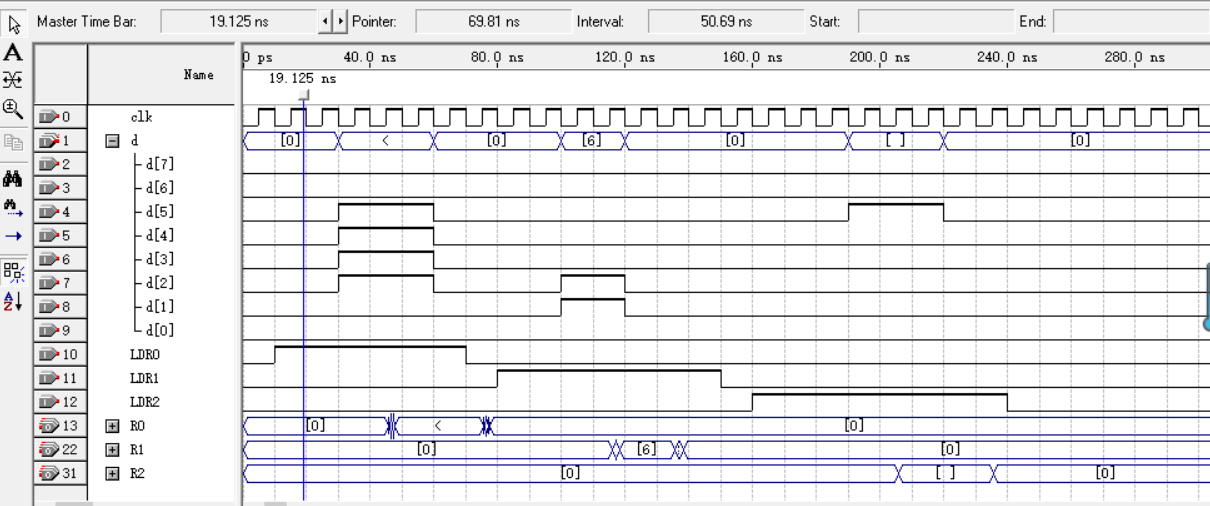
寄存器堆：



仿真文件：

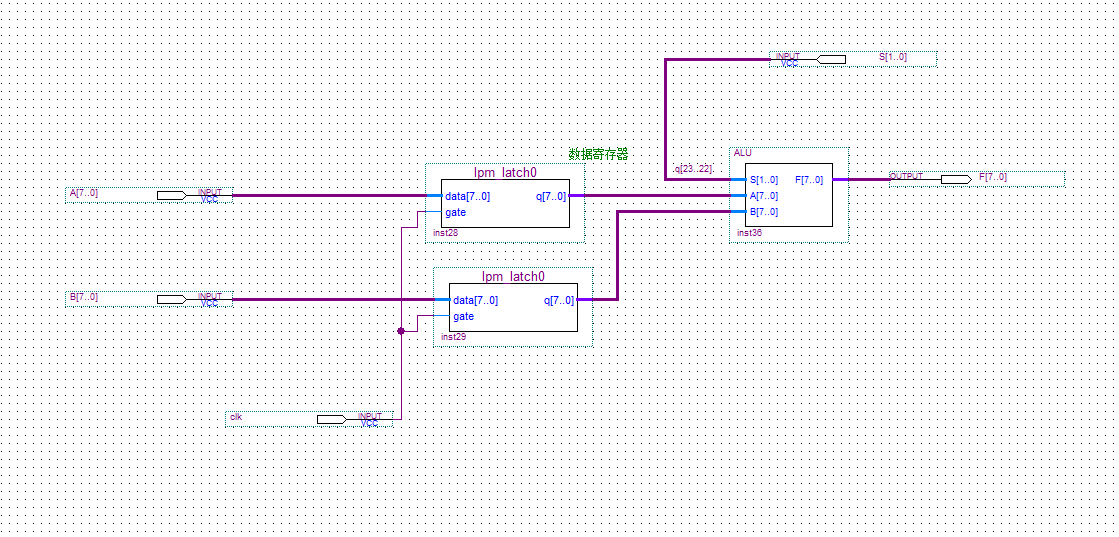


仿真结果：



**2.2.2 运算器模块**

DR+ALU模块：



ALU代码部分：

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ALU IS

PORT(S :IN STD\_LOGIC\_VECTOR(1 DOWNTO 0);

A,B :IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

F :OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0));

-- CO,FZ:OUT STD\_LOGIC);

END ALU;

ARCHITECTURE behave OF ALU IS

BEGIN

PROCESS(A,B)

BEGIN

CASE S IS

WHEN "00"=> F<=NOT A;

WHEN "01"=> F<=A+B;

WHEN "10"=> F<=A-B;

WHEN "11"=> F<=A+1;

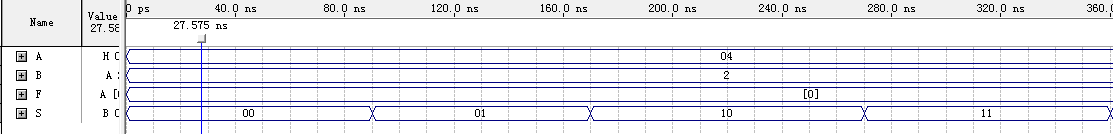
WHEN OTHERS=> F<="00000000";

END CASE;

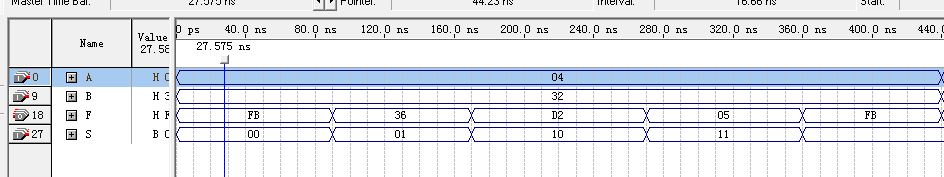
END PROCESS;

END behave;

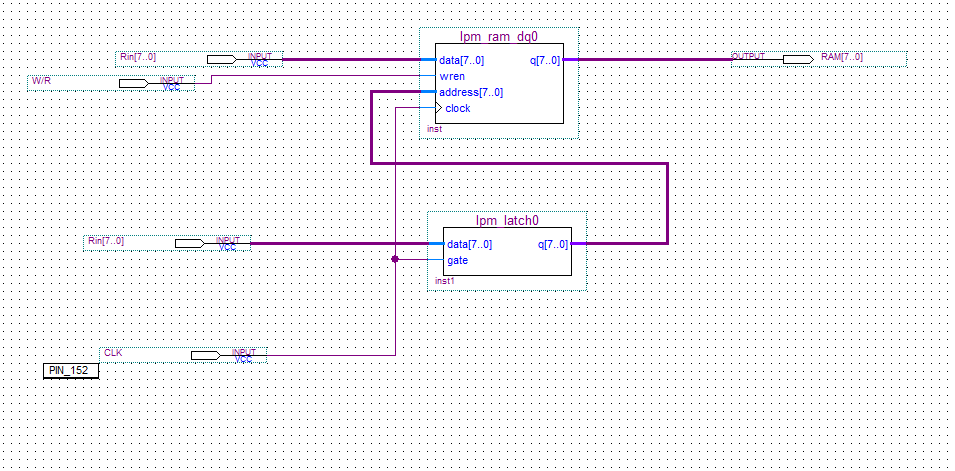
仿真文件：



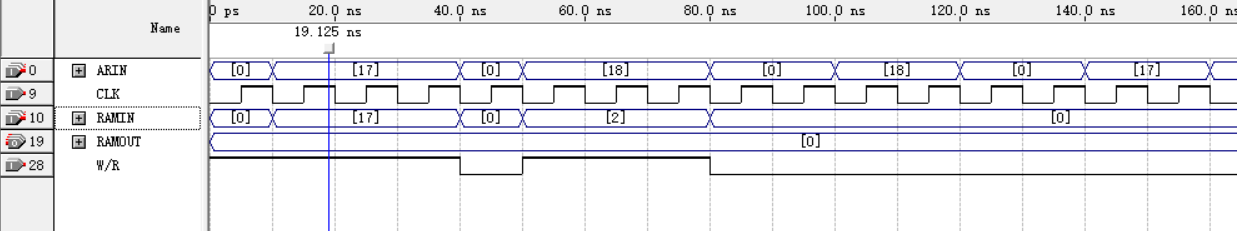
仿真结果：



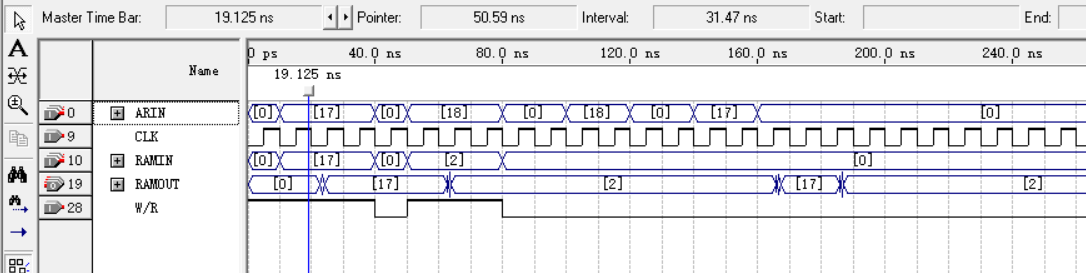
**2.2.3 存储器模块**



仿真文件：

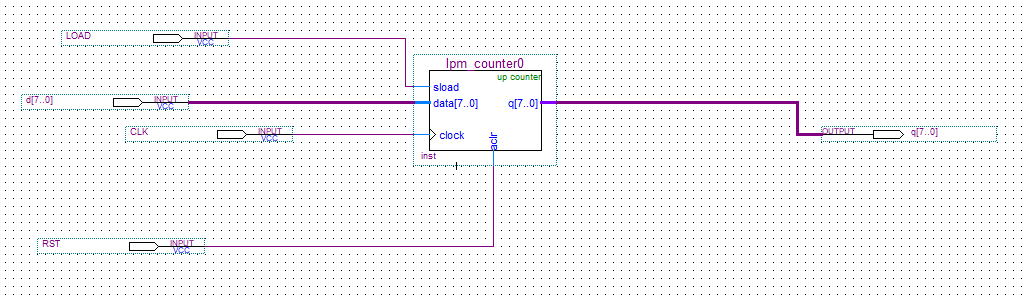


仿真结果：

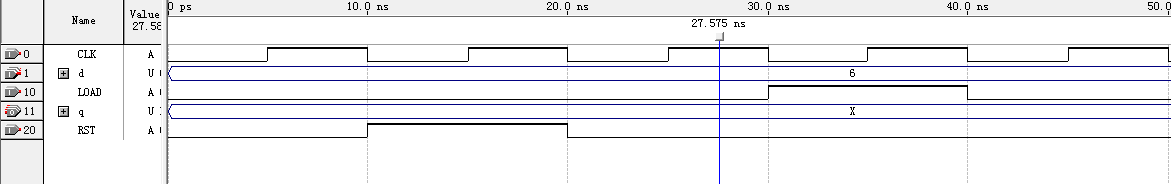


**2.2.4 其他模块**

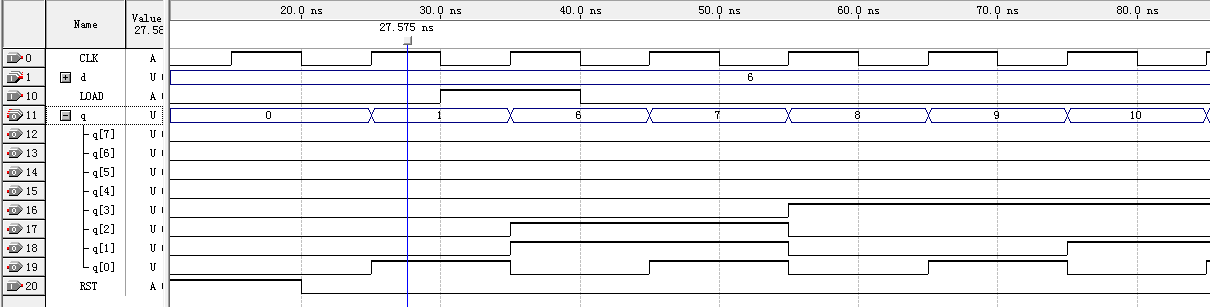
PC：



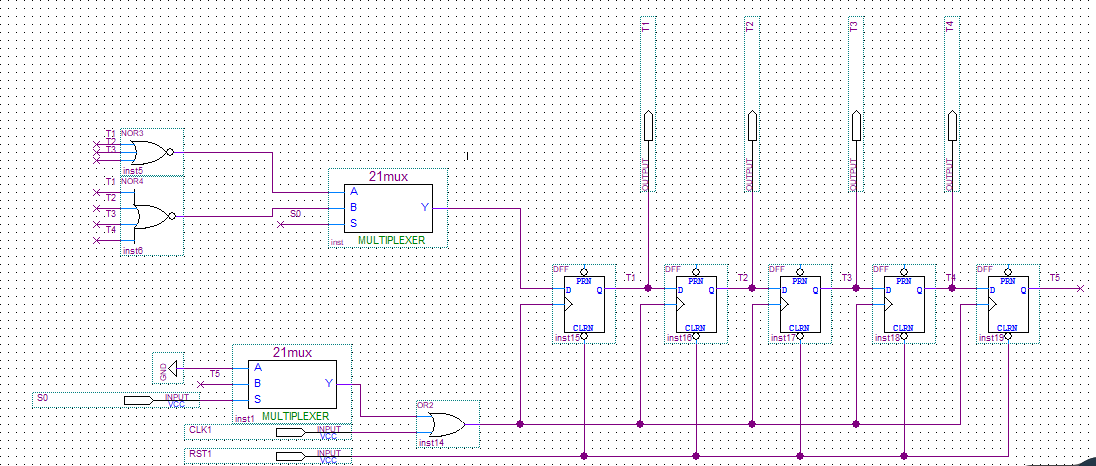
仿真文件：



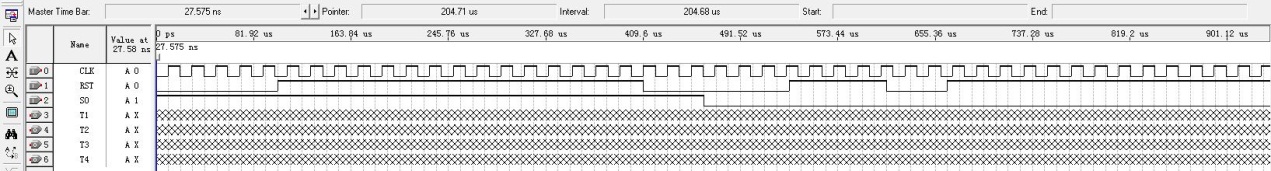
仿真结果：



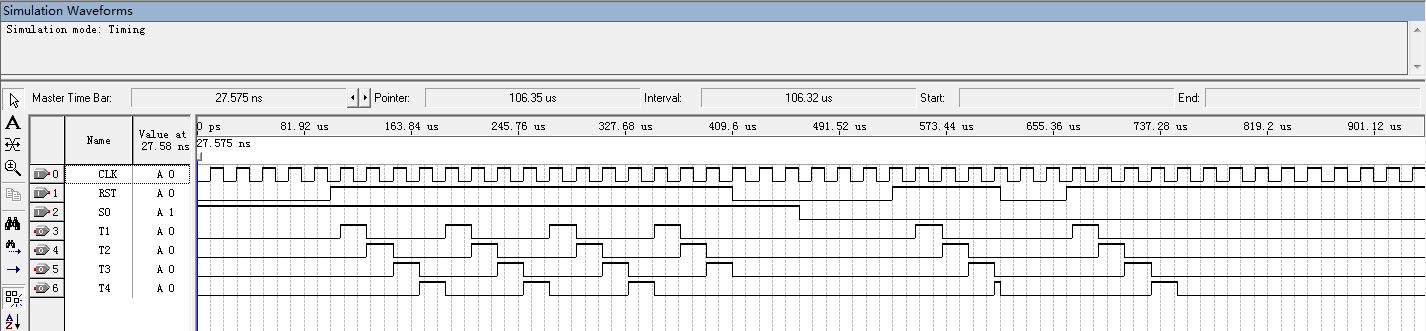
TIMER：



仿真文件：

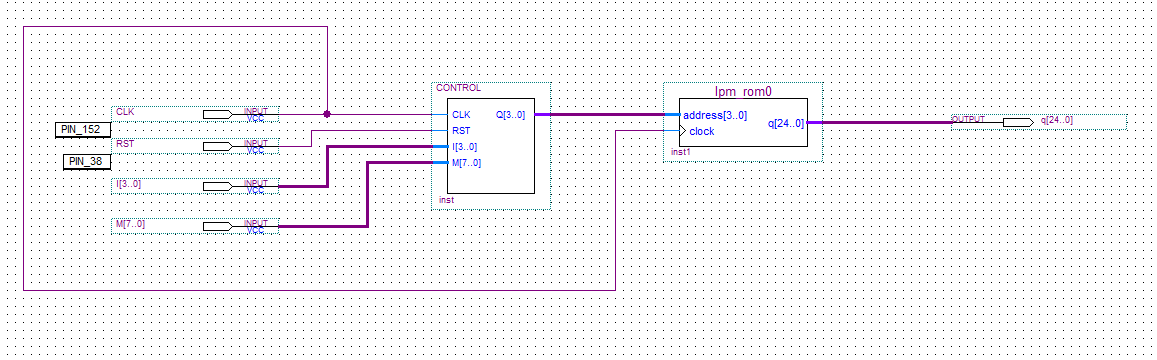


仿真结果：



**2.2.5 控制器模块**

控制器模块：



微控制器：

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY CONTROL IS

PORT ( CLK : IN STD\_LOGIC;

RST : IN STD\_LOGIC;

I : IN STD\_LOGIC\_VECTOR(3 DOWNTO 0);

M : IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

Q : OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0) );

END CONTROL;

ARCHITECTURE behave OF CONTROL IS

BEGIN

PROCESS(I,M,RST,CLK)

BEGIN

IF RST='1'

THEN

Q<="0000";

ELSE

IF clk'event and clk='1'

THEN

IF I="0010"

THEN

CASE M IS

WHEN "00010000"=> Q<="0010"; --IN0 02

WHEN "00100100"=> Q<="0011"; --IN1 03

WHEN "00110100"=> Q<="0100"; --ADD 04

WHEN "01000100"=> Q<="0111"; --SUB 07

WHEN "01010000"=> Q<="1010"; --OUT 12

WHEN "01000000"=> Q<="1111"; --jump 15

WHEN OTHERS=> Q<="0000";

END CASE;

ELSE

Q<=I;

END IF;

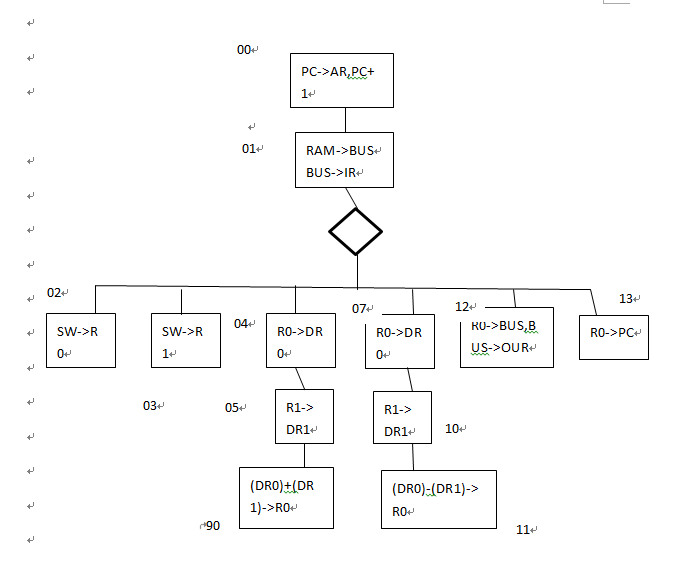
END IF;

END IF;

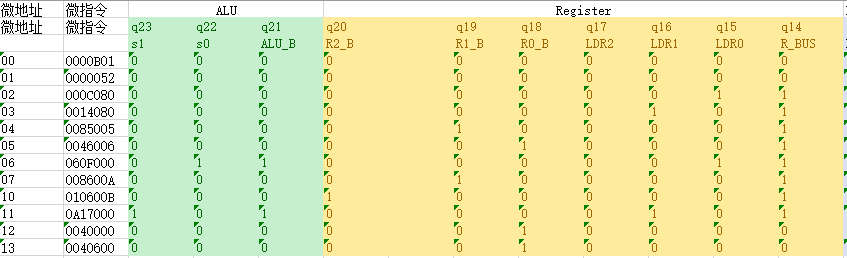
END PROCESS;

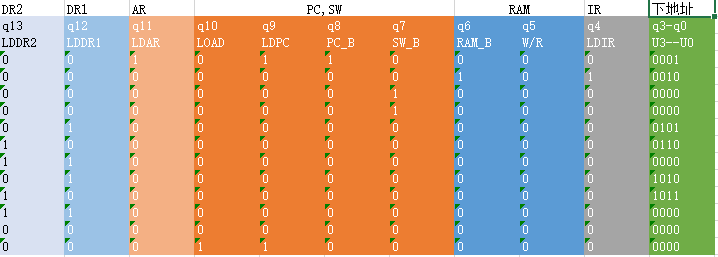
END behave;

流程图：



微指令：

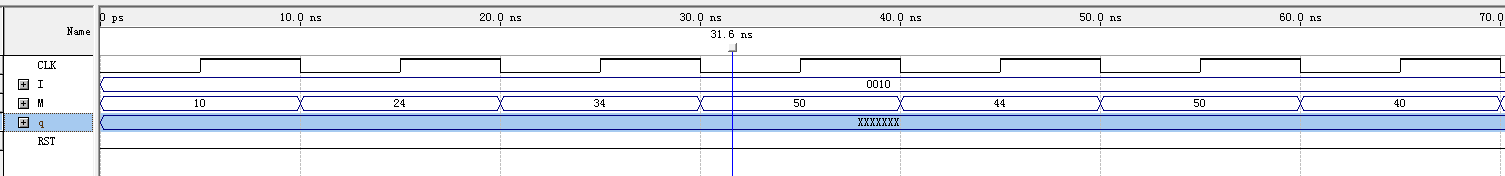




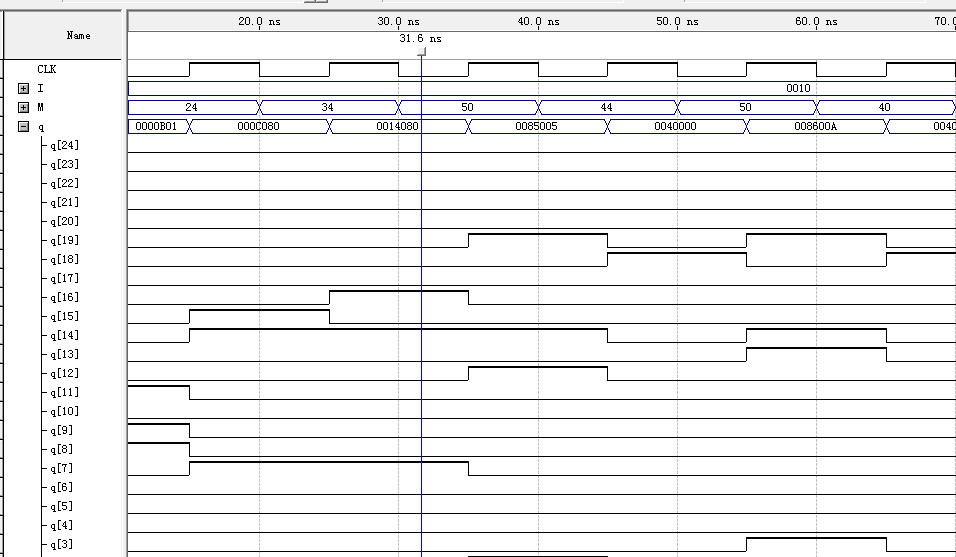
ROM1.mif:



仿真文件：



仿真结果：

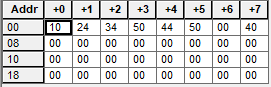


**2.3整机调试**

**2.3.1测试程序设计**

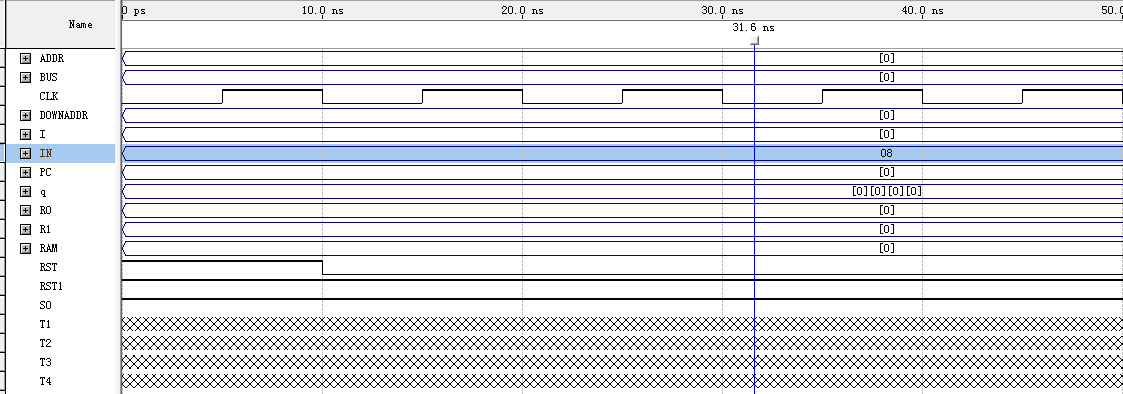
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **助记符** | **机器指令码** | **源操作数** | **目的操作数** | **功能** | **编码** |
| IN0 | 0001 | 00 | XX | SW→R0 | 10H |
| IN1 | 0010 | 01 | XX | SW->R1 | 24H |
| ADD R0，R1 | 0011 | 01 | 00 | R0+R1→R0 | 34 H |
| SUB R0， R1 | 0100 | 01 | 00 | R0-R1→R0 | 44 H |
| JMP R0 | 0100 | 00 | XX | R0→PC | 40 H |
| OUT | 0101 | 00 | XX | R0→BUS | 50 H |

RAM1.mif:

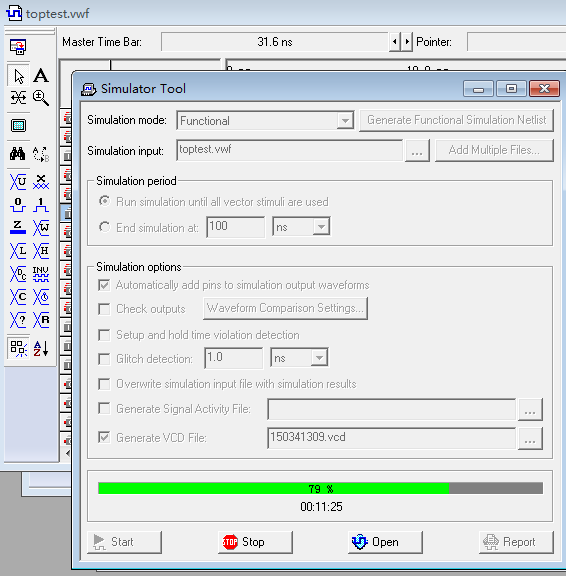


**2.3.2 仿真结果**

仿真文件:



仿真结果：

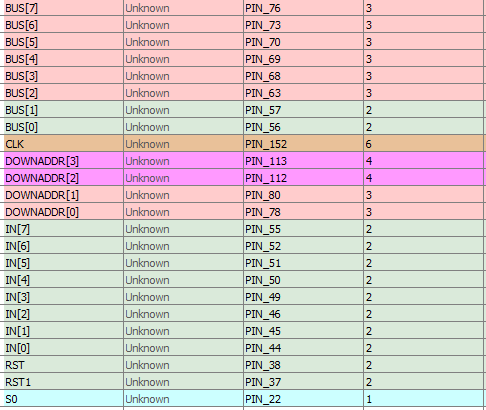


顶层图的仿真总是有的时候能出来有的时候出不来。

**2.3.3下载后的结果**

模式选择0。

管脚绑定：



输出结果说明：

键1和键2控制8位输入，键3控制时序一直为1，键4控制时序单步连续，键5控制整体复位。数码管1,2显示总线数据和微指令下地址。

DOWNADDR[3..0]为下地址晶码管输出显示。

IN [7..0]是8位数据输入。

RST是整体的重置。

RST1是时序发生器重置。

S0是时序单步连续选择。

1. 实验总结及心得体会

总结：RAM的时钟控制应该是INCLOCK而不是CLOCK,在下载的过程中因为设置错了CLOCK导致结果刚开始有些不对，其他地方并没有什么大的问题，就是有些单元模块仿真出了问题。

体会：通过半学期的实验比较彻底的了解了RICS指令系统，在老师和同学的帮助下对CPU是如何工作的有了更深的了解。