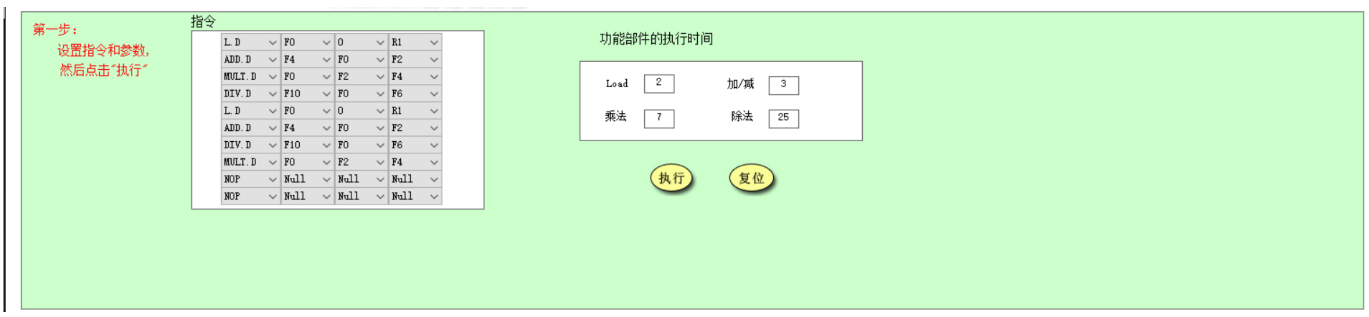
计算机体系结构lab4

19281030-张云鹏

Tomasulo算法

输入指令程序, 配置指令耗时


第一步：
设置指令和参数，
然后点击"执行"

指令

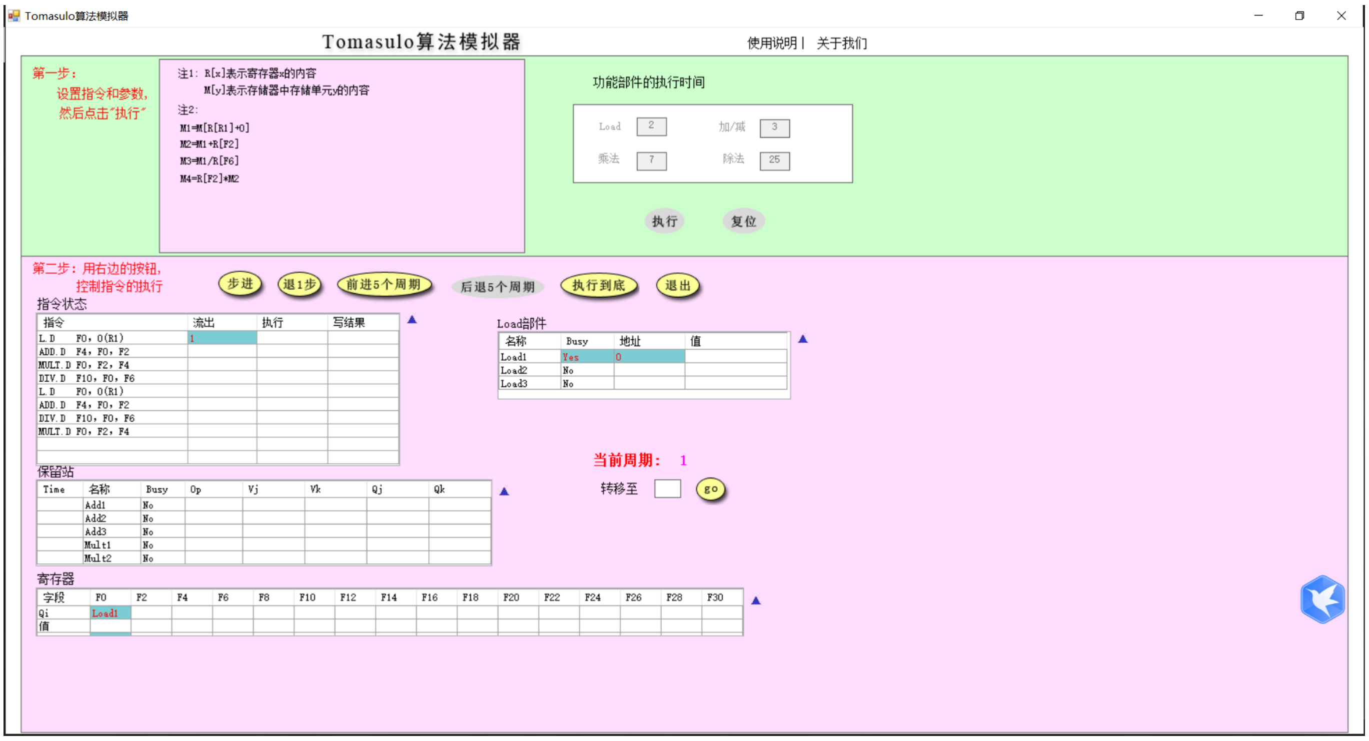
L.D	F0	0	R1
ADD.D	F4	F0	F2
MULT.D	F0	F2	F4
DIV.D	F10	F0	F6
L.D	F0	0	R1
ADD.D	F4	F0	F2
DIV.D	F10	F0	F6
MULT.D	F0	F2	F4
NOP	Null	Null	Null
NOP	Null	Null	Null

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

指令执行周期跟踪

1. `Cycle 1`
    - 取出第一条指令`L.D F0, 0(R1)`
    - `Load1`设为Busy
    - `Load1`名存入F0



第一步：
设置指令和参数，
然后点击"执行"

注1: R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]
M3=M1/R[F6]
M4=R[F2]*M2

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，
控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1		
ADD.D F4，F0，F2			
MULT.D F0，F2，F4			
DIV.D F10，F0，F6			
L.D F0，0(R1)			
ADD.D F4，F0，F2			
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	Yes	0	
Load2	No		
Load3	No		

当前周期： 1

转移至

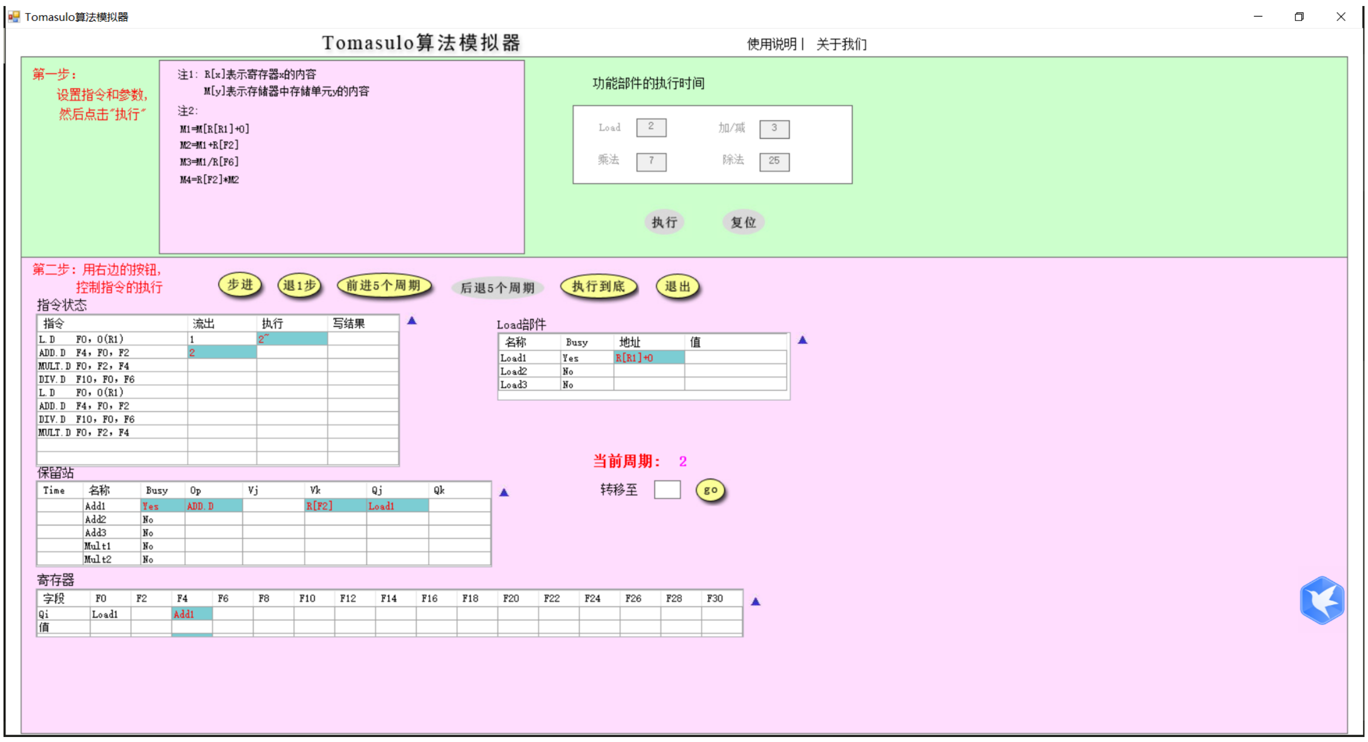
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1															
值																

2. `Cycle 2`
    - 取出第二条指令 `ADD.D F4, F0, F2`
    - `Qj`存入`Load1`
    - `Add1`设为Busy
    - `Add1`名存入 `F4`
    - `L.D`指令执行



Tomasulo算法模拟器

第一步：
设置指令和参数，
然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2：
M1=M[R[R1]+0]
M2=M1+R[F2]
M3=M1/R[F6]
M4=R[F2]*M2

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，
控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~	
ADD.D F4，F0，F2	2		
MULT.D F0，F2，F4			
DIV.D F10，F0，F6			
L.D F0，0(R1)			
ADD.D F4，F0，F2			
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	Yes	R[R1]+0	
Load2	No		
Load3	No		

当前周期： 2

转移至

保留站

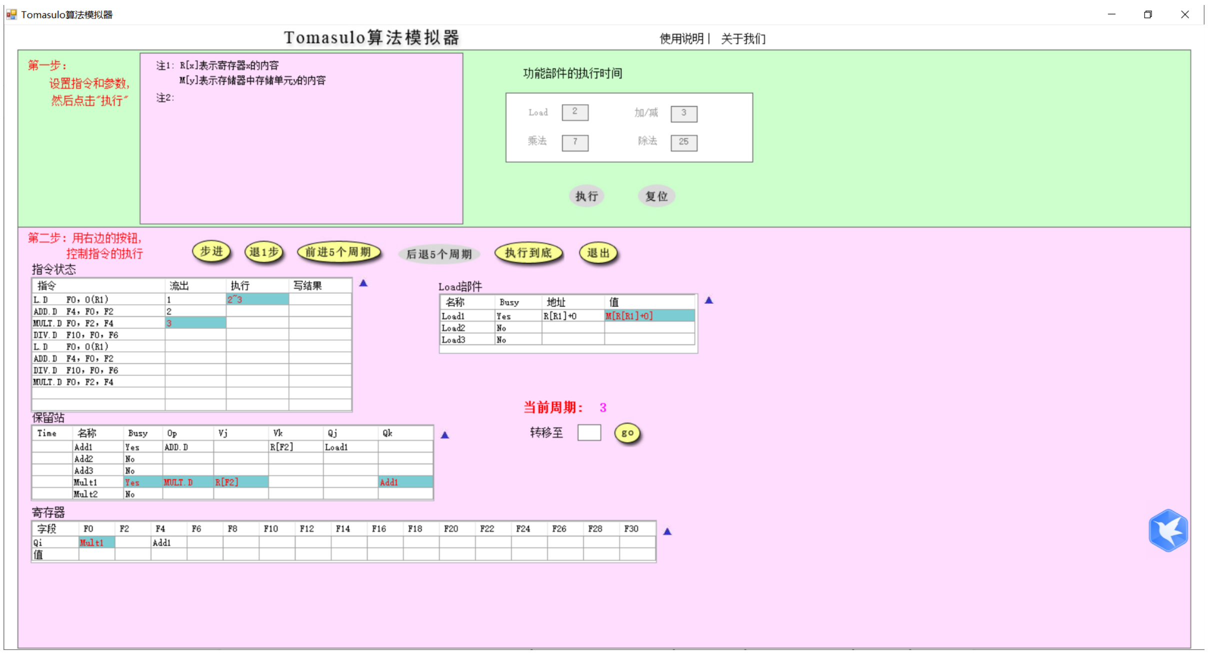
Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	Yes	ADD.D		R[F2]	Load1	
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add1													
值																

3. `Cycle 3`

   - `Load`指令执行完毕
   - 取出第三条指令 `MULT.D F0, F2, F4`
   - Mult1.Busy=Yes
   - F0.Qi=Mult1
   - Mult1.Qk=Add1



Tomasulo算法模拟器

第一步：
设置指令和参数，
然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2：

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，
控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	
ADD.D F4，F0，F2	2		
MULT.D F0，F2，F4	3		
DIV.D F10，F0，F6			
L.D F0，0(R1)			
ADD.D F4，F0，F2			
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	Yes	R[R1]+0	M[R[R1]+0]
Load2	No		
Load3	No		

当前周期： 3

转移至

保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	Yes	ADD.D		R[F2]	Load1	
	Add2	No					
	Add3	No					
	Mult1	Yes	MULT.D	R[F2]			Add1
	Mult2	No					

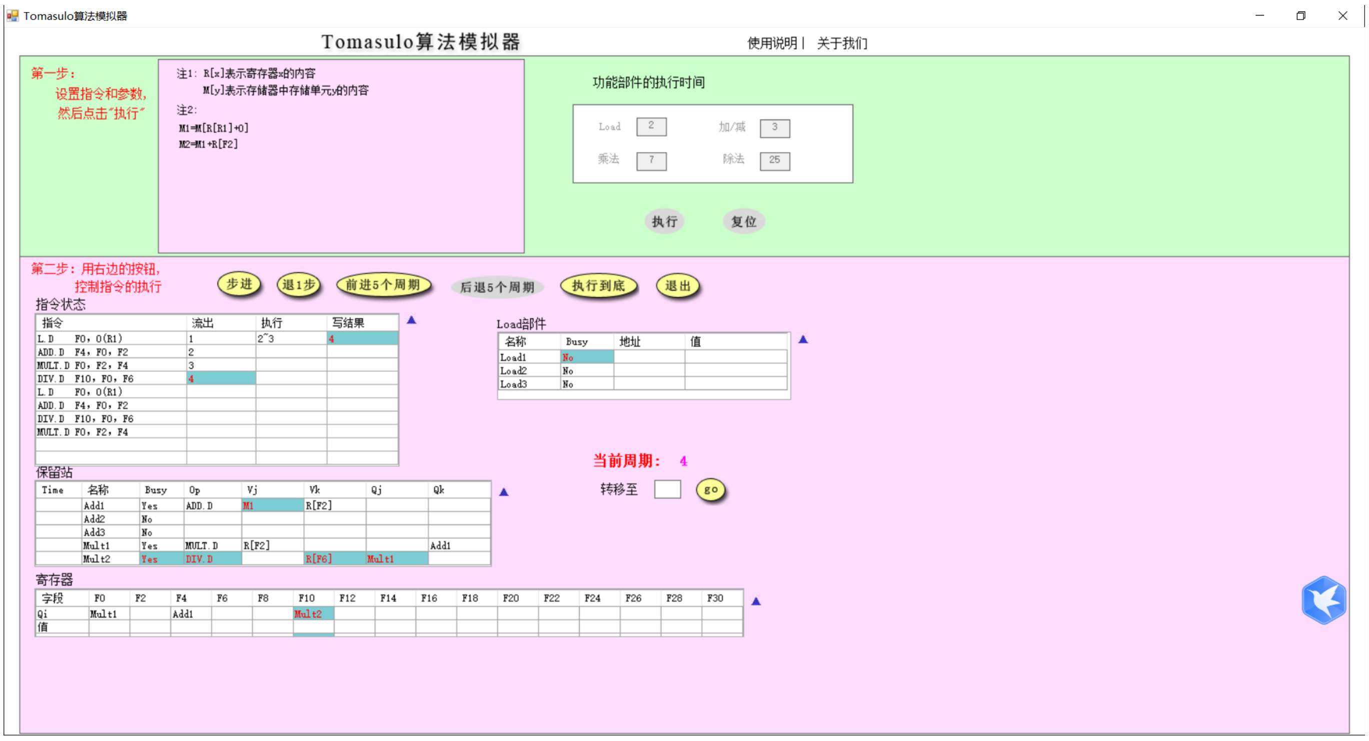
寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Mult1		Add1													
值																

4. Cycle 4

   - Load1 Write Result
   - Load1.Busy =No
   - Add1.Vj=M1
   - Fetch 4th instruction
   - Mult2.Busy = Yes

- Mult2.Qj = Mult1



Tomasulo算法模拟器

第一步：设置指令和参数，然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2		
MULT.D F0，F2，F4	3		
DIV.D F10，F0，F6	4		
L.D F0，0(R1)			
ADD.D F4，F0，F2			
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 4

转移至

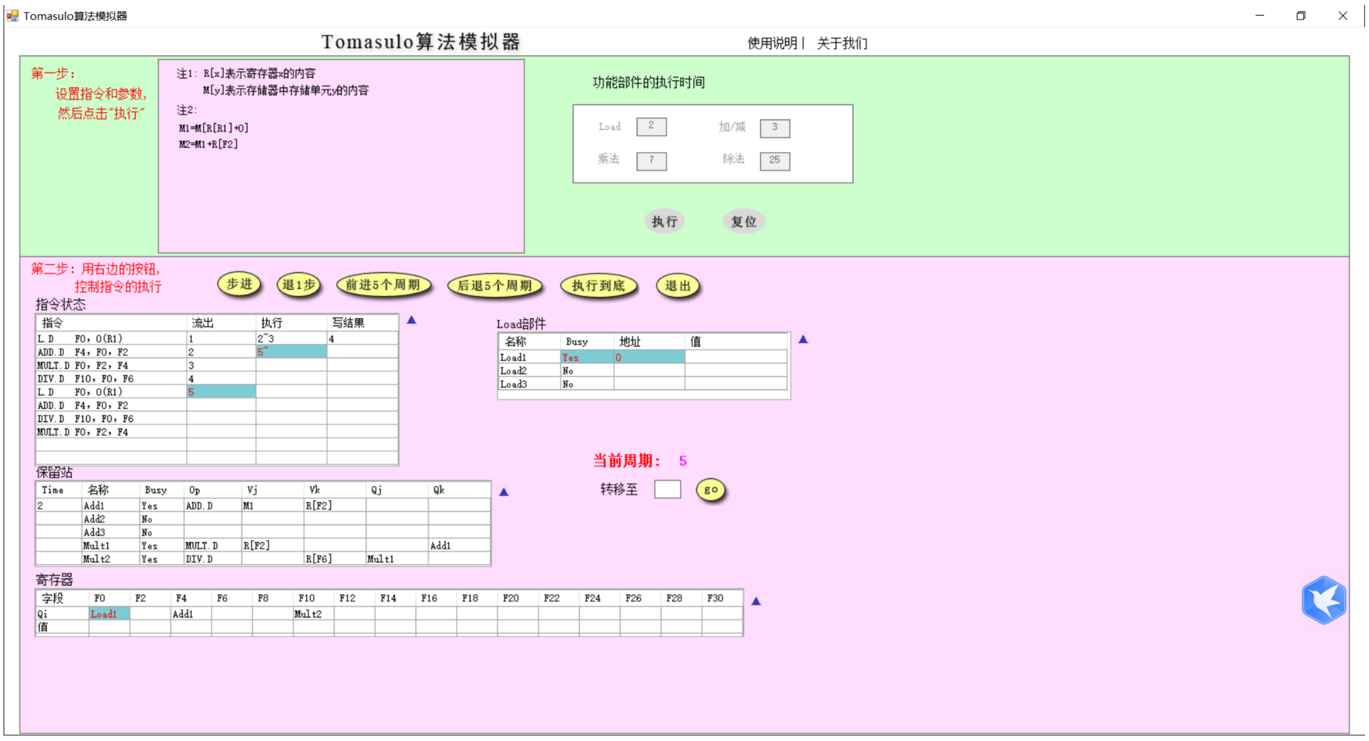
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	Yes	ADD.D	M1	R[F2]		
	Add2	No					
	Add3	No					
	Mult1	Yes	MULT.D	R[F2]			Add1
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Mult1		Add1			Mult2										
值																

5. Cycle 5
    - 2nd instruction start to execute
    - fetch 5th instruction
    - Load1.Busy = Yes
    - F0.Qi = Load1



Tomasulo算法模拟器

第一步：设置指令和参数，然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~	
MULT.D F0，F2，F4	3		
DIV.D F10，F0，F6	4		
L.D F0，0(R1)	5		
ADD.D F4，F0，F2			
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	Yes	0	
Load2	No		
Load3	No		

当前周期： 5

转移至

保留站

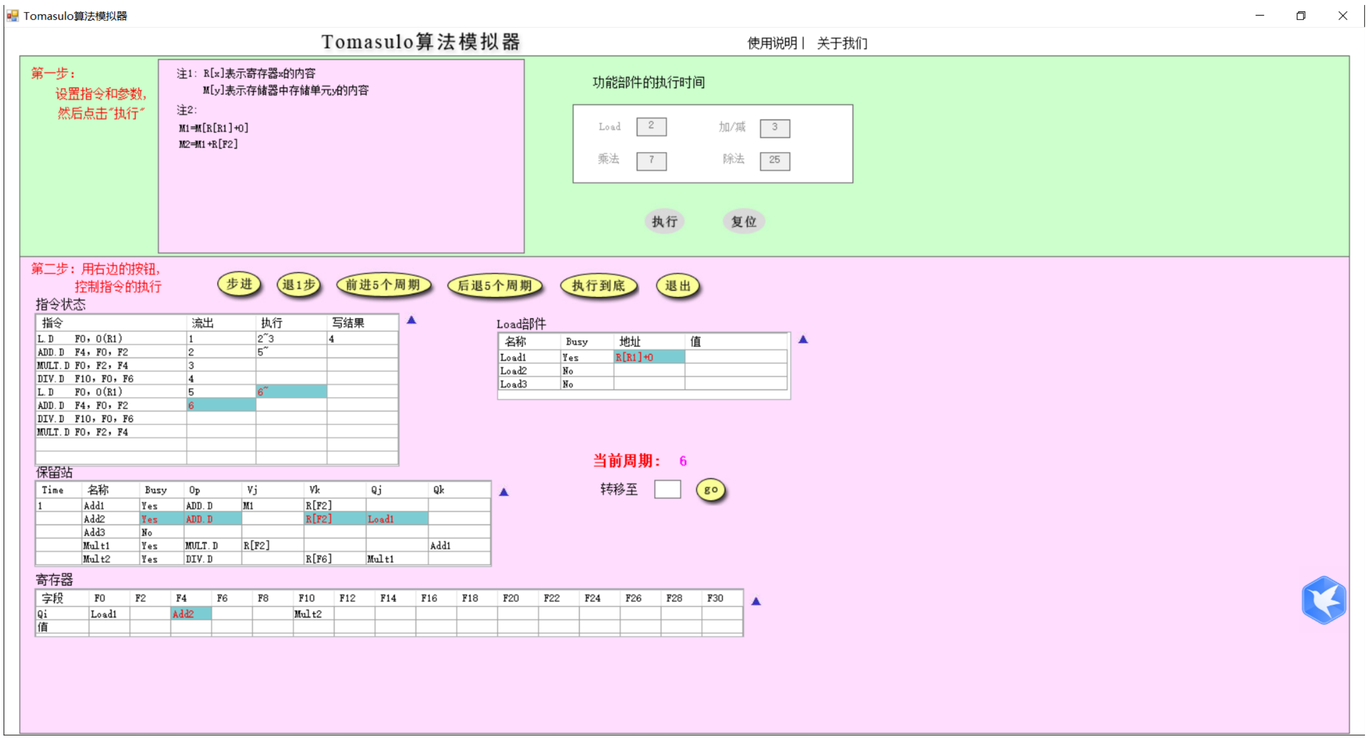
Time	名称	Busy	Op	Vj	Vk	Qj	Qk
2	Add1	Yes	ADD.D	M1	R[F2]		
	Add2	No					
	Add3	No					
	Mult1	Yes	MULT.D	R[F2]			Add1
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add1			Mult2										
值																

6. Cycle 6
    - Load1 start to execute
    - Fetch 6th instruction
    - Add2.Busy = Yes

- Add2.Qj = Load1



Tomasulo算法模拟器

使用说明 | 关于我们

第一步：设置指令和参数，然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~	
MULT.D F0，F2，F4	3		
DIV.D F10，F0，F6	4		
L.D F0，0(R1)	5	6~	
ADD.D F4，F0，F2	6		
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	Yes	R[R1]+0	
Load2	No		
Load3	No		

当前周期：6

转移至

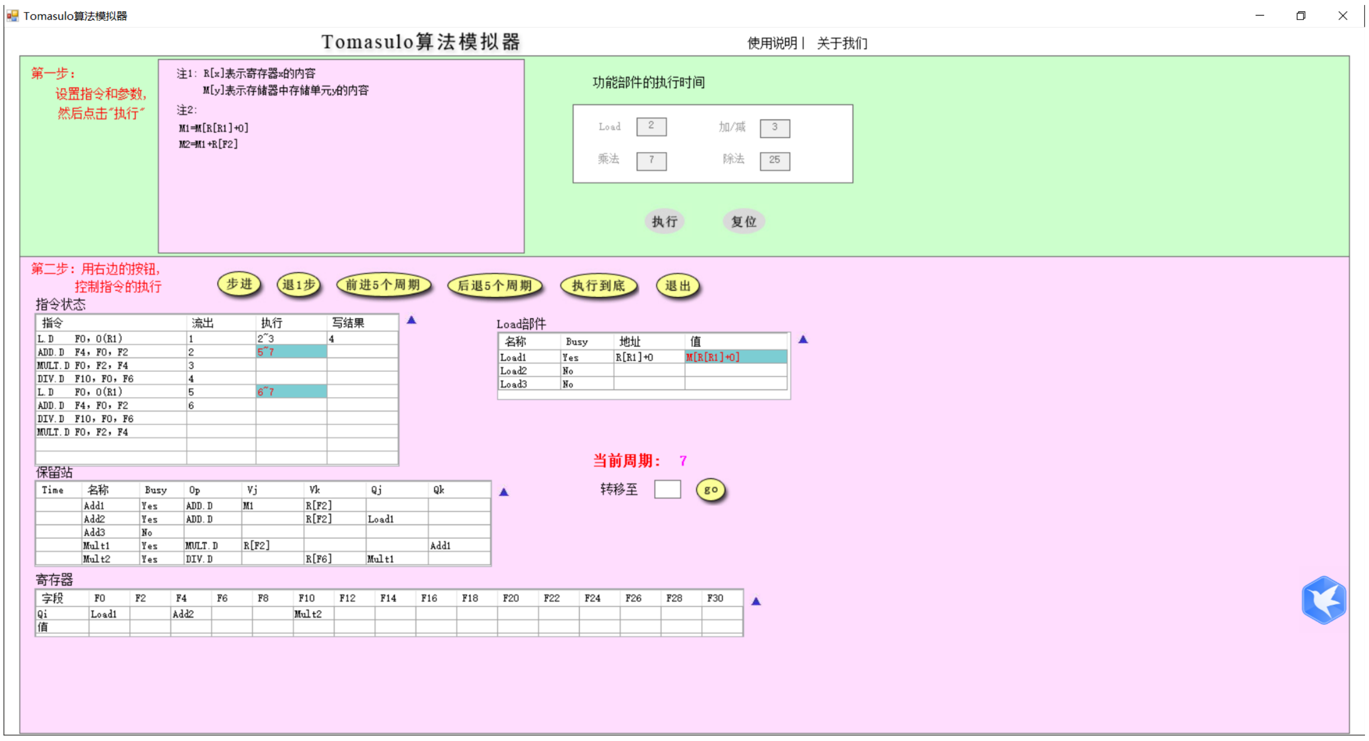
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
1	Add1	Yes	ADD.D	M1	R[F2]		
	Add2	Yes	ADD.D		R[F2]	Load1	
	Add3	No					
	Mult1	Yes	MULT.D	R[F2]			Add1
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult2										
值																

7. Cycle 7
   - 2nd and 5th instruction complete



Tomasulo算法模拟器

使用说明 | 关于我们

第一步：设置指令和参数，然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~7	
MULT.D F0，F2，F4	3		
DIV.D F10，F0，F6	4		
L.D F0，0(R1)	5	6~7	
ADD.D F4，F0，F2	6		
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	Yes	R[R1]+0	M[R[R1]+0]
Load2	No		
Load3	No		

当前周期：7

转移至

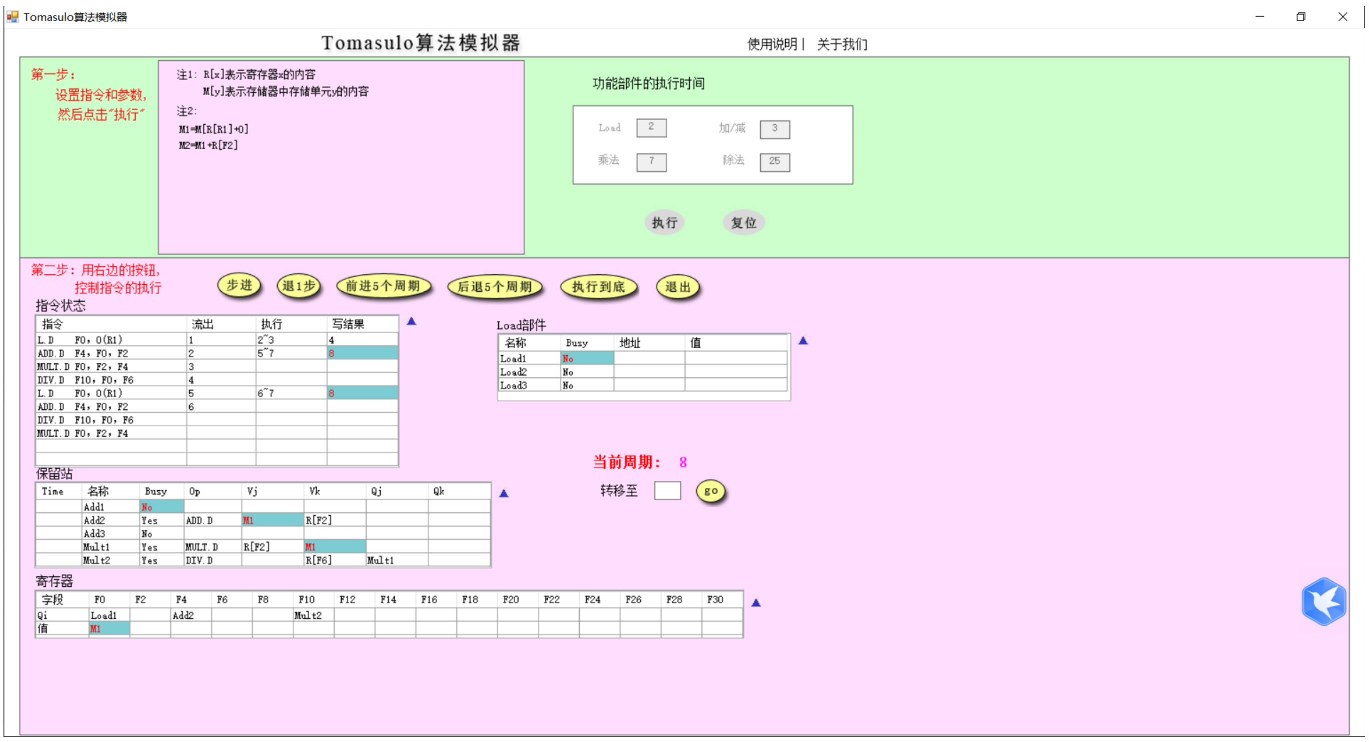
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	Yes	ADD.D	M1	R[F2]		
	Add2	Yes	ADD.D		R[F2]	Load1	
	Add3	No					
	Mult1	Yes	MULT.D	R[F2]			Add1
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult2										
值																

8. Cycle 8
   - 2nd and 5th instruction write result
   - F0.Value = M1
   - Add2.Vj = M1
   - Mult1.Vk = M1



注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

Load	2	加/减	3
乘法	7	除法	25

指令状态

指令	流出	执行	写结果
L.D F0, 0(R1)	1	2~3	4
ADD.D F4, F0, F2	2	5~7	8
MULT.D F0, F2, F4	3		
DIV.D F10, F0, F6	4		
L.D F0, 0(R1)	5	6~7	8
ADD.D F4, F0, F2	6		
DIV.D F10, F0, F6			
MULT.D F0, F2, F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 8

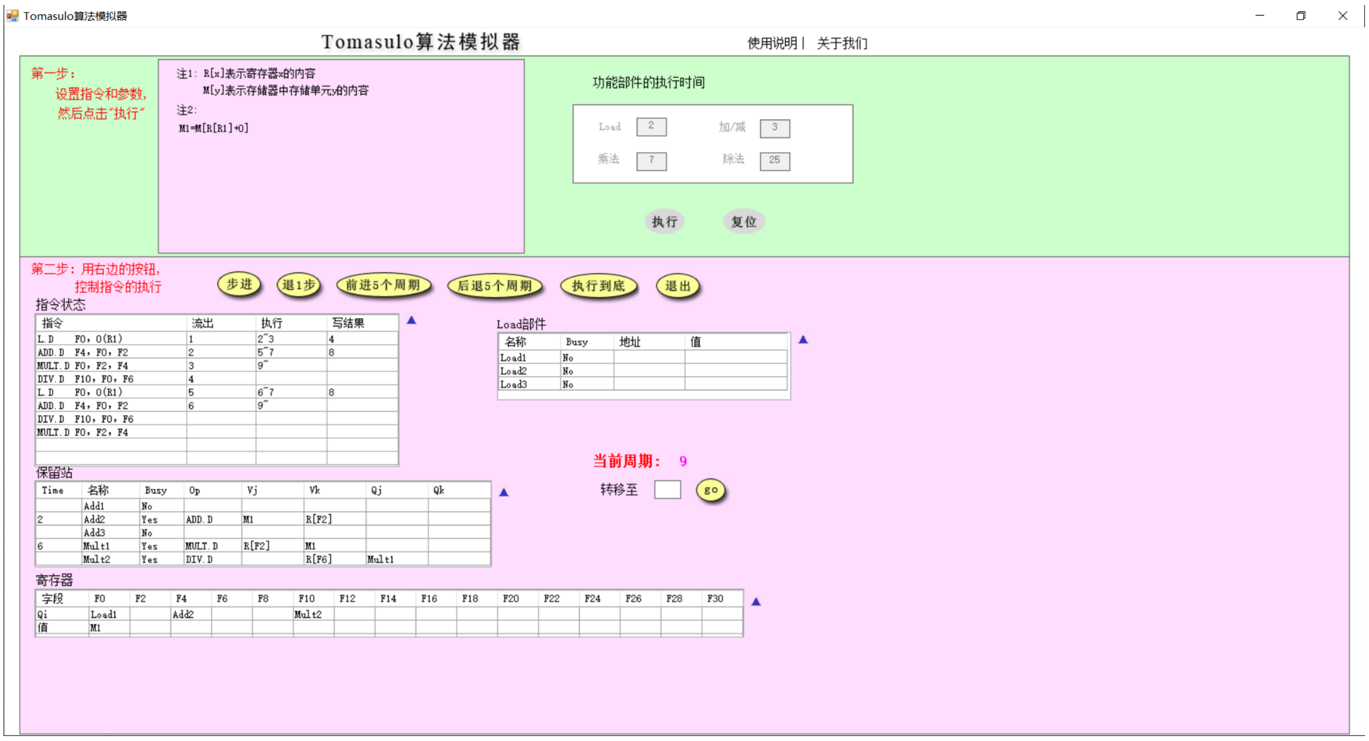
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	Yes	ADD.D	M1	R[F2]		
	Add3	No					
	Mult1	Yes	MULT.D	R[F2]	M1		
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult2										
值	M1															

9. Cycle 9
    - Mult2 and Add2 start to execute



注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]

Load	2	加/减	3
乘法	7	除法	25

指令状态

指令	流出	执行	写结果
L.D F0, 0(R1)	1	2~3	4
ADD.D F4, F0, F2	2	5~7	8
MULT.D F0, F2, F4	3	9~	
DIV.D F10, F0, F6	4		
L.D F0, 0(R1)	5	6~7	8
ADD.D F4, F0, F2	6	9~	
DIV.D F10, F0, F6			
MULT.D F0, F2, F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 9

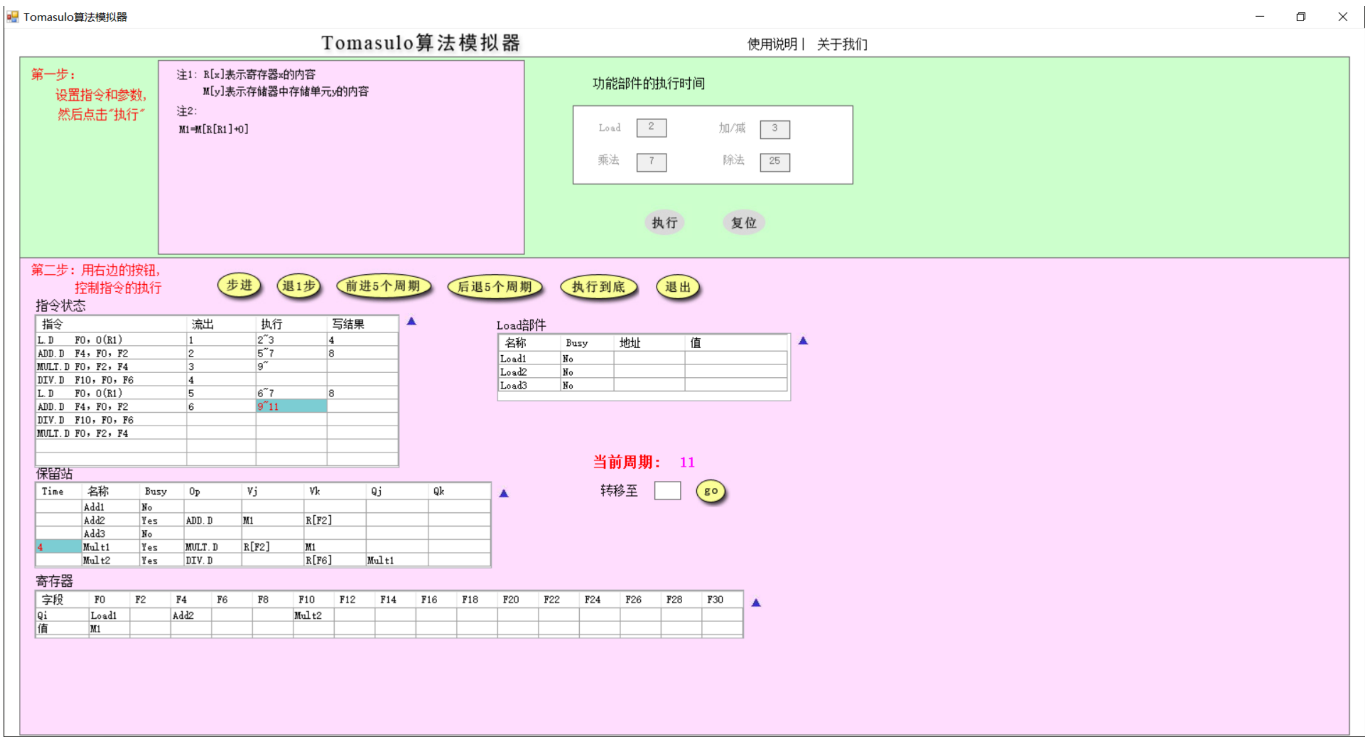
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
2	Add2	Yes	ADD.D	M1	R[F2]		
	Add3	No					
6	Mult1	Yes	MULT.D	R[F2]	M1		
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult2										
值	M1															

10. Skip Cycle 10
11. Cycle 11
    - Add2 Complete



Tomasulo算法模拟器

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]

当前周期： 11

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~7	8
MULT.D F0，F2，F4	3	9~	
DIV.D F10，F0，F6	4		
L.D F0，0(R1)	5	6~7	8
ADD.D F4，F0，F2	6	9~11	
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

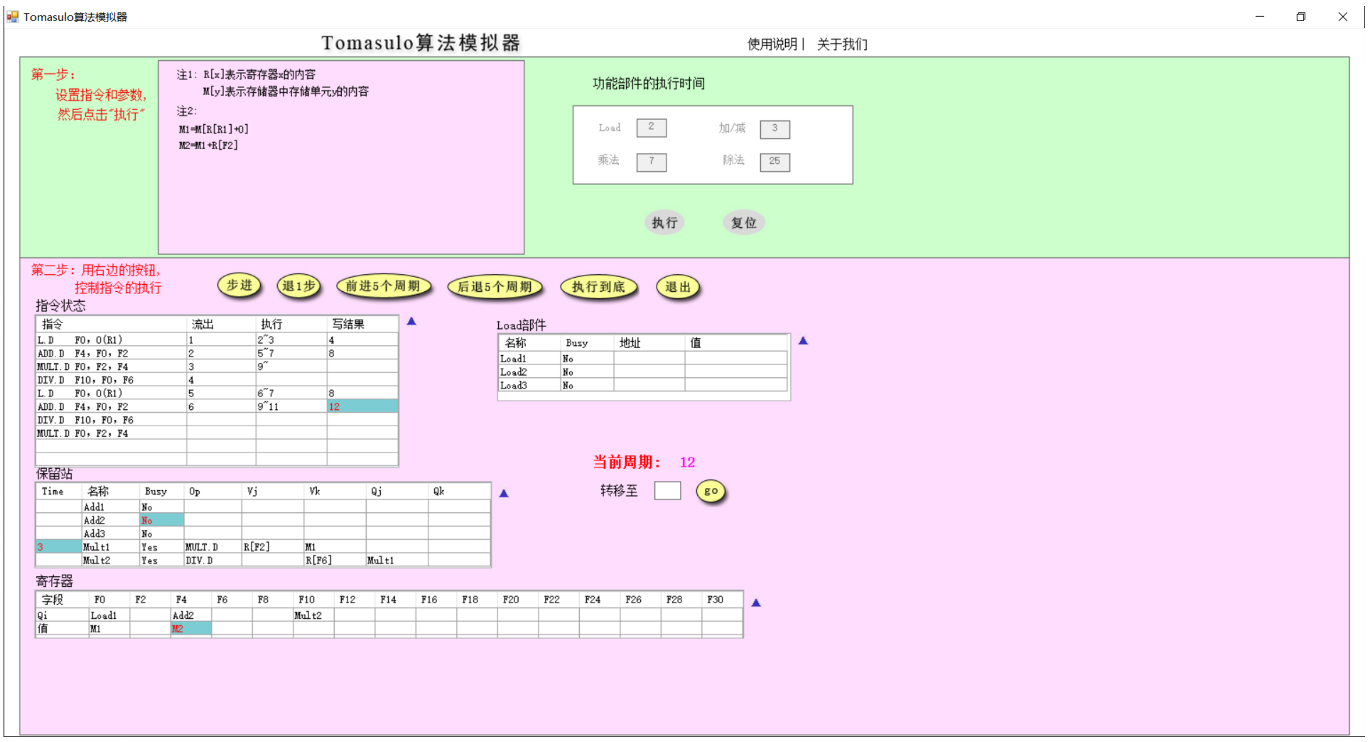
保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	Yes	ADD.D	M1	R[F2]		
	Add3	No					
4	Mult1	Yes	MULT.D	R[F2]	M1		
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult2										
值	M1															

12. Cycle 12
    - 6th write to F4
    - F4.Value = M2
    - Add2.Busy = No



Tomasulo算法模拟器

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

当前周期： 12

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~7	8
MULT.D F0，F2，F4	3	9~	
DIV.D F10，F0，F6	4		
L.D F0，0(R1)	5	6~7	8
ADD.D F4，F0，F2	6	9~11	12
DIV.D F10，F0，F6			
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
3	Mult1	Yes	MULT.D	R[F2]	M1		
	Mult2	Yes	DIV.D		R[F6]	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult2										
值	M1		M2													

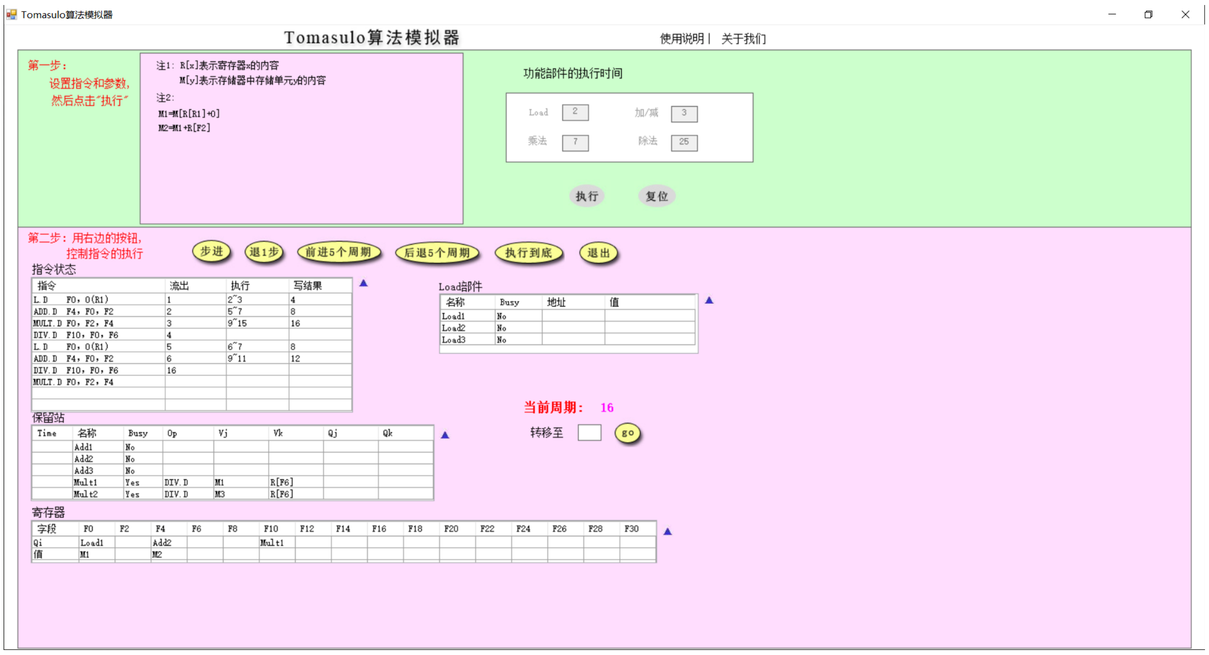
13. Cycle 13 skip

14. Cycle 14 skip

15. Cycle 15 skip

16. Cycle 16

- 3rd instruction write to F10
- Fetch 7th instruction
- F10.Qi = Mult1
- Mult2.Vj = M3



Tomasulo算法模拟器

第一步：设置指令和参数，然后点击"执行"

注1: R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~7	8
MULT.D F0，F2，F4	3	9~15	16
DIV.D F10，F0，F6	4		
L.D F0，0(R1)	5	6~7	8
ADD.D F4，F0，F2	6	9~11	12
DIV.D F10，F0，F6	16		
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 16

保留站

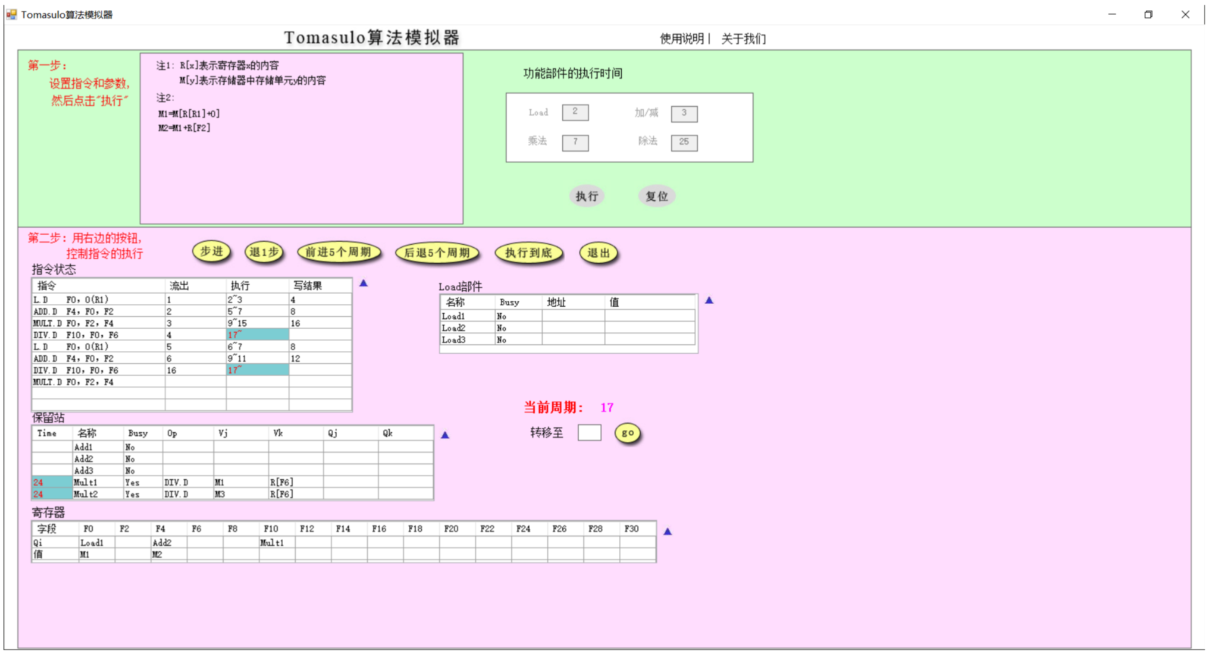
Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	Yes	DIV.D	M1	R[F6]		
	Mult2	Yes	DIV.D	M3	R[F6]		

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult1										
值	M1		M2													

17. Cycle 17

- 4th and 7th instruction start to execute



Tomasulo算法模拟器

第一步：设置指令和参数，然后点击"执行"

注1: R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容
注2:
M1=M[R[R1]+0]
M2=M1+R[F2]

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~7	8
MULT.D F0，F2，F4	3	9~15	16
DIV.D F10，F0，F6	4	17~	
L.D F0，0(R1)	5	6~7	8
ADD.D F4，F0，F2	6	9~11	12
DIV.D F10，F0，F6	16	17~	
MULT.D F0，F2，F4			

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 17

保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
24	Mult1	Yes	DIV.D	M1	R[F6]		
24	Mult2	Yes	DIV.D	M3	R[F6]		

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Load1		Add2			Mult1										
值	M1		M2													

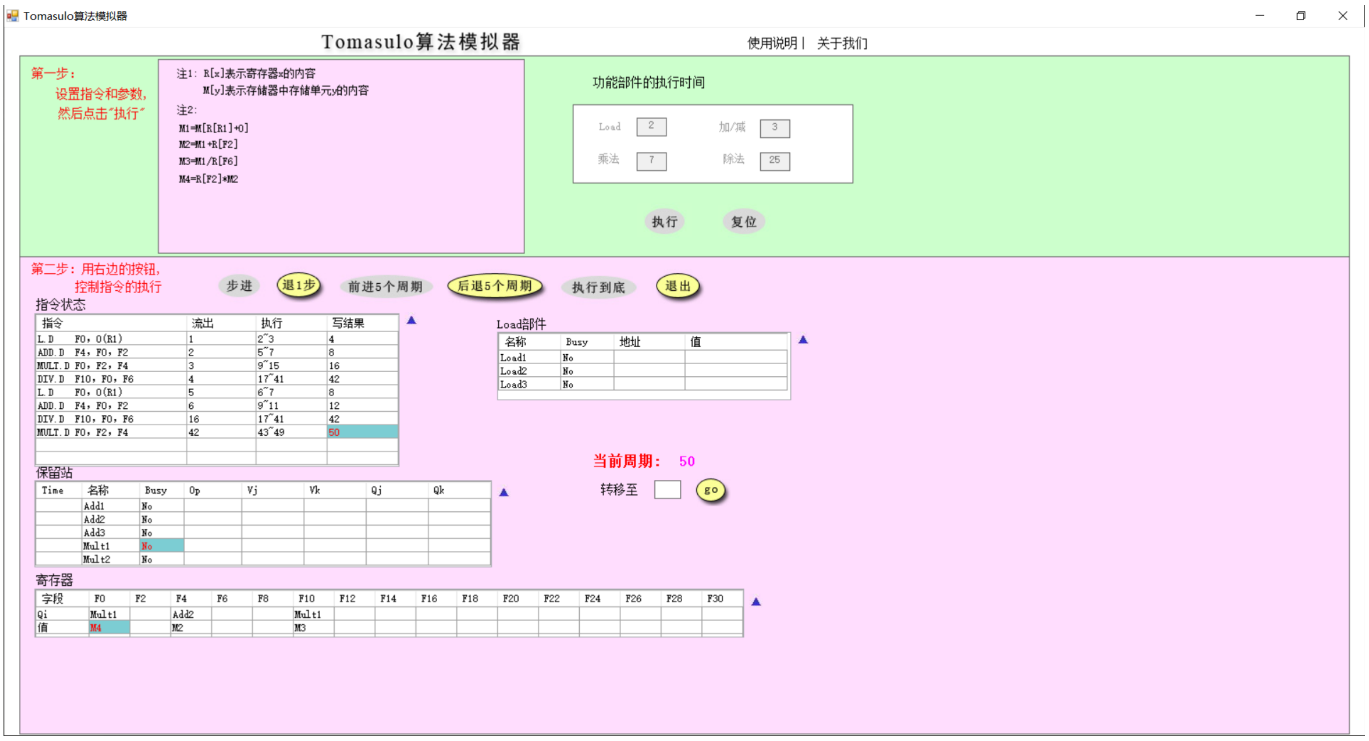
18. Cycle 18 ~ Cycle 41 skip

19. Cycle 42

- 4th and 7th instruction write results
- F10.Value = M3
- Fetch 8th instruction
- F0.Qi = Mult1

20. Cycle 43 ~ Cycle 49 skip

21. Cycle 50
    - 8th write to F0
    - F0.Value = M4



Tomasulo算法模拟器

第一步：设置指令和参数，然后点击"执行"

注1：R[x]表示寄存器x的内容
M[y]表示存储器中存储单元y的内容

注2：
M1=M[R[R1]+0]
M2=M1+R[F2]
M3=M1/R[F6]
M4=R[F2]*M2

功能部件的执行时间

Load	2	加/减	3
乘法	7	除法	25

第二步：用右边的按钮，控制指令的执行

指令状态

指令	流出	执行	写结果
L.D F0，0(R1)	1	2~3	4
ADD.D F4，F0，F2	2	5~7	8
MULT.D F0，F2，F4	3	9~15	16
DIV.D F10，F0，F6	4	17~41	42
L.D F0，0(R1)	5	6~7	8
ADD.D F4，F0，F2	6	9~11	12
DIV.D F10，F0，F6	16	17~41	42
MULT.D F0，F2，F4	42	43~49	50

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 50

转移至

保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Mult1		Add2			Mult1										
值	M4		M2			M3										

Tomasulo 原理分析

RAW(写后读):

1. L.D F0, (0)R1
2. ADD.D F1, F0

Tomasulo算法会先将Load后面依赖F0的, 使用Load1. 相当于通过保留站的表映射, 缓存依赖关系, 等到数据准备好之后通过CDB broadcast, 所以后面的指令无需等待停顿, 从而提高了效率.

WAR(读后写)

1. ADD.D F2, F1
2. L.D F1, (0)R1

Tomasulo重命名读的寄存器为Load1, 之前的Add1并不会受到影响.

WAW(写后写)

1. L.D F1, (0)R1
2. L.D F1, (0)R2 Tomasulo流出后即进行写预约, 后面的可以覆盖前面的, 所以保证最后结果是最后的输出