由于作业要求里只要求**"显示程序成功完成了“input1.txt”和“input2.txt”中指令下的动态调度，并回答第三节的附加问题"**，因此实验报告仅包括**新增假设、运行结果和附加问题**三部分。

新增假设

在实验要求的原有假设之下，增加如下假设：

- ROB的大小与指令集的指令条数相同
- Load和Store指令的执行时间为2个周期，也就是说在条件允许的情况下，可以在它们发射周期的下一个周期进入执行状态
- Load和Store指令的偏移量均是常数，没有需要从寄存器读值的

运行结果

详细运行结果可以查阅两个输出文件**output1.txt**和**output2.txt**，下面只放每条指令提交周期和最终执行情况表。

样例1

- 每条指令的提交周期：

```
Cycle_5;
Entry1: No,LD F6 34+ R2,Commit,F6,Mem[34+Regs[R2]];
Entry2: Yes,LD F2 45+ R3,Write Result,F2,Mem[45+Regs[R3]];
Entry3: Yes,MULTD F0 F2 F4,Execute,F0,#2*Regs[F4];
Entry4: Yes,SUBD F8 F6 F2,Execute,F8,#1-#2;
Entry5: Yes,DIVD F10 F0 F6,Issue,F10,#3/#1;
Entry6: No,,,,;
Load1: No,,,,,,;
Load2: No,fld,45,Regs[R3],,,#2;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: Yes,fsub.d,Regs[F6],Regs[F2],,,#4;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: Yes,fmul.d,Regs[F2],Regs[F2],,,#3;
Mult2: Yes,fdiv.d,,Regs[F0],#3,,#5;
Reorder#: F0:3;F1:;F2:2;F3:;F4:;F5:;F6:;F7:;F8:4;F9:;F10:5;
Busy: F0:Yes;F1:No;F2:Yes;F3:No;F4:No;F5:No;F6:No;F7:No;F8:Yes;F9:No;F10:Yes;
```

```
Cycle_6;
Entry1: No,LD F6 34+ R2,Commit,F6,Mem[34+Regs[R2]];
Entry2: No,LD F2 45+ R3,Commit,F2,Mem[45+Regs[R3]];
Entry3: Yes,MULTD F0 F2 F4,Execute,F0,#2*Regs[F4];
Entry4: Yes,SUBD F8 F6 F2,Execute,F8,#1-#2;
Entry5: Yes,DIVD F10 F0 F6,Issue,F10,#3/#1;
Entry6: Yes,ADDD F6 F8 F2,Issue,F6,#4+#2;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: Yes,fsub.d,Regs[F6],Regs[F2],,,#4;
Add2: Yes,fadd.d,,Regs[F8],#4,,#6;
Add3: No,,,,,,;
Mult1: Yes,fmul.d,Regs[F2],Regs[F2],,,#3;
Mult2: Yes,fdiv.d,,Regs[F0],#3,,#5;
Reorder#: F0:3;F1:;F2:;F3:;F4:;F5:;F6:6;F7:;F8:4;F9:;F10:5;
Busy: F0:Yes;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:Yes;F9:No;F10:Yes;
```

```
Cycle_16;
Entry1: No,LD F6 34+ R2,Commit,F6,Mem[34+Regs[R2]];
Entry2: No,LD F2 45+ R3,Commit,F2,Mem[45+Regs[R3]];
Entry3: No,MULTD F0 F2 F4,Commit,F0,#2*Regs[F4];
Entry4: Yes,SUBD F8 F6 F2,Write Result,F8,#1-#2;
Entry5: Yes,DIVD F10 F0 F6,Execute,F10,#3/#1;
Entry6: Yes,ADDD F6 F8 F2,Write Result,F6,#4+#2;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,fsub.d,Regs[F6],Regs[F2],,,#4;
Add2: No,fadd.d,Regs[F8],Regs[F8],,,#6;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: Yes,fdiv.d,Regs[F0],Regs[F0],,,#5;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:6;F7:;F8:4;F9:;F10:5;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:Yes;F9:No;F10:Yes;
```

```
Cycle_17;
Entry1: No,LD F6 34+ R2,Commit,F6,Mem[34+Regs[R2]];
Entry2: No,LD F2 45+ R3,Commit,F2,Mem[45+Regs[R3]];
Entry3: No,MULTD F0 F2 F4,Commit,F0,#2*Regs[F4];
Entry4: No,SUBD F8 F6 F2,Commit,F8,#1-#2;
Entry5: Yes,DIVD F10 F0 F6,Execute,F10,#3/#1;
Entry6: Yes,ADDD F6 F8 F2,Write Result,F6,#4+#2;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,fadd.d,Regs[F8],Regs[F8],,,#6;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: Yes,fdiv.d,Regs[F0],Regs[F0],,,#5;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:6;F7:;F8:;F9:;F10:5;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:Yes;
```

```
Cycle_36;
Entry1: No,LD F6 34+ R2,Commit,F6,Mem[34+Regs[R2]];
Entry2: No,LD F2 45+ R3,Commit,F2,Mem[45+Regs[R3]];
Entry3: No,MULTD F0 F2 F4,Commit,F0,#2*Regs[F4];
Entry4: No,SUBD F8 F6 F2,Commit,F8,#1-#2;
Entry5: No,DIVD F10 F0 F6,Commit,F10,#3/#1;
Entry6: Yes,ADDD F6 F8 F2,Write Result,F6,#4+#2;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,fadd.d,Regs[F8],Regs[F8],,,#6;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: No,,,,,,;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:6;F7:;F8:;F9:;F10:;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_37;
Entry1: No,LD F6 34+ R2,Commit,F6,Mem[34+Regs[R2]];
Entry2: No,LD F2 45+ R3,Commit,F2,Mem[45+Regs[R3]];
Entry3: No,MULTD F0 F2 F4,Commit,F0,#2*Regs[F4];
Entry4: No,SUBD F8 F6 F2,Commit,F8,#1-#2;
Entry5: No,DIVD F10 F0 F6,Commit,F10,#3/#1;
Entry6: No,ADDD F6 F8 F2,Commit,F6,#4+#2;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: No,,,,,,;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:;F7:;F8:;F9:;F10:;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:No;F7:No;F8:No;F9:No;F10:No;
```

- 最终执行情况表：

```
Final execution table:
Instruction     Issue cycle    Exec comp cycle  Write result cycle    Commit cycle
LD F6 34+ R2    1              2                4                     5
LD F2 45+ R3    2              3                5                     6
MULTD F0 F2 F4  3              5                15                    16
SUBD F8 F6 F2   4              5                7                     17
DIVD F10 F0 F6  5              15               35                    36
ADDD F6 F8 F2   6              7                9                     37
```

样例2

- 每条指令的提交周期：

```
Cycle_5;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: Yes,LD F4 0 R3,Write Result,F4,Mem[Regs[R3]];
Entry3: Yes,DIVD F0 F4 F2,Execute,F0,#2/#1;
Entry4: Yes,MULTD F6 F0 F2,Issue,F6,#3*#1;
Entry5: No,,,,;
Entry6: No,,,,;
Entry7: No,,,,;
Entry8: No,,,,;
Load1: No,,,,,,;
Load2: No,fld,0,Regs[R3],,,#2;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: Yes,fdiv.d,Regs[F4],Regs[F2],,,#3;
Mult2: Yes,fmul.d,,Regs[F2],#3,,#4;
Reorder#: F0:3;F1:;F2:;F3:;F4:2;F5:;F6:4;F7:;F8:;F9:;F10:;
Busy: F0:Yes;F1:No;F2:No;F3:No;F4:Yes;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_6;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: Yes,DIVD F0 F4 F2,Execute,F0,#2/#1;
Entry4: Yes,MULTD F6 F0 F2,Issue,F6,#3*#1;
Entry5: No,,,,;
Entry6: No,,,,;
Entry7: No,,,,;
Entry8: No,,,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: Yes,fdiv.d,Regs[F4],Regs[F2],,,#3;
Mult2: Yes,fmul.d,,Regs[F2],#3,,#4;
Reorder#: F0:3;F1:;F2:;F3:;F4:;F5:;F6:4;F7:;F8:;F9:;F10:;
Busy: F0:Yes;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_26;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: No,DIVD F0 F4 F2,Commit,F0,#2/#1;
Entry4: Yes,MULTD F6 F0 F2,Execute,F6,#3*#1;
Entry5: Yes,ADDD F0 F4 F2,Issue,F0,#2+#1;
Entry6: No,,,,;
Entry7: No,,,,;
Entry8: No,,,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: Yes,fadd.d,Regs[F4],Regs[F4],,,#5;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: Yes,fmul.d,Regs[F0],Regs[F2],,,#4;
Reorder#: F0:5;F1:;F2:;F3:;F4:;F5:;F6:4;F7:;F8:;F9:;F10:;
Busy: F0:Yes;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_36;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: No,DIVD F0 F4 F2,Commit,F0,#2/#1;
Entry4: No,MULTD F6 F0 F2,Commit,F6,#3*#1;
Entry5: Yes,ADDD F0 F4 F2,Write Result,F0,#2+#1;
Entry6: Yes,SD F6 0 R3,Issue,,;
Entry7: No,,,,;
Entry8: No,,,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: Yes,fstp,0,Regs[R3],,,#6;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,fadd.d,Regs[F4],Regs[F4],,,#5;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: No,,,,,,;
Reorder#: F0:5;F1:;F2:;F3:;F4:;F5:;F6:;F7:;F8:;F9:;F10:;
Busy: F0:Yes;F1:No;F2:No;F3:No;F4:No;F5:No;F6:No;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_37;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: No,DIVD F0 F4 F2,Commit,F0,#2/#1;
Entry4: No,MULTD F6 F0 F2,Commit,F6,#3*#1;
Entry5: No,ADDD F0 F4 F2,Commit,F0,#2+#1;
Entry6: Yes,SD F6 0 R3,Execute,,;
Entry7: Yes,MULTD F6 F0 F2,Issue,F6,#5*#1;
Entry8: No,,,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: Yes,fstp,0,Regs[R3],,,#6;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: Yes,fmul.d,Regs[F0],Regs[F0],,,#7;
Mult2: No,,,,,,;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:7;F7:;F8:;F9:;F10:;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_40;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: No,DIVD F0 F4 F2,Commit,F0,#2/#1;
Entry4: No,MULTD F6 F0 F2,Commit,F6,#3*#1;
Entry5: No,ADDD F0 F4 F2,Commit,F0,#2+#1;
Entry6: No,SD F6 0 R3,Commit,,;
Entry7: Yes,MULTD F6 F0 F2,Execute,F6,#5*#1;
Entry8: No,,,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: Yes,fmul.d,Regs[F0],Regs[F0],,,#7;
Mult2: No,,,,,,;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:7;F7:;F8:;F9:;F10:;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:Yes;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_49;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: No,DIVD F0 F4 F2,Commit,F0,#2/#1;
Entry4: No,MULTD F6 F0 F2,Commit,F6,#3*#1;
Entry5: No,ADDD F0 F4 F2,Commit,F0,#2+#1;
Entry6: No,SD F6 0 R3,Commit,,;
Entry7: No,MULTD F6 F0 F2,Commit,F6,#5*#1;
Entry8: Yes,SD F6 0 R1,Issue,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: Yes,fstp,0,Regs[R1],,,#8;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: No,,,,,,;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:;F7:;F8:;F9:;F10:;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:No;F7:No;F8:No;F9:No;F10:No;
```

```
Cycle_53;
Entry1: No,LD F2 0 R2,Commit,F2,Mem[Regs[R2]];
Entry2: No,LD F4 0 R3,Commit,F4,Mem[Regs[R3]];
Entry3: No,DIVD F0 F4 F2,Commit,F0,#2/#1;
Entry4: No,MULTD F6 F0 F2,Commit,F6,#3*#1;
Entry5: No,ADDD F0 F4 F2,Commit,F0,#2+#1;
Entry6: No,SD F6 0 R3,Commit,,;
Entry7: No,MULTD F6 F0 F2,Commit,F6,#5*#1;
Entry8: No,SD F6 0 R1,Commit,,;
Load1: No,,,,,,;
Load2: No,,,,,,;
Load3: No,,,,,,;
Store1: No,,,,,,;
Store2: No,,,,,,;
Store3: No,,,,,,;
Add1: No,,,,,,;
Add2: No,,,,,,;
Add3: No,,,,,,;
Mult1: No,,,,,,;
Mult2: No,,,,,,;
Reorder#: F0:;F1:;F2:;F3:;F4:;F5:;F6:;F7:;F8:;F9:;F10:;
Busy: F0:No;F1:No;F2:No;F3:No;F4:No;F5:No;F6:No;F7:No;F8:No;F9:No;F10:No;
```

- 最终执行情况表：

```
Final execution table:
Instruction     Issue cycle    Exec comp cycle  Write result cycle    Commit cycle
LD F2 0 R2      1              2                4                     5
LD F4 0 R3      2              3                5                     6
DIVD F0 F4 F2   3              5                25                    26
MULTD F6 F0 F2  4              25               35                    36
ADDD F0 F4 F2   26             27               29                    37
SD F6 0 R3      36             37               39                    40
MULTD F6 F0 F2  37             38               48                    49
SD F6 0 R1      49             50               52                    53
```

附加问题

(1)Tomasulo 算法相对于 Scoreboard 算法的优点？同时简述Tomasulo 存在的缺点。

Tomasulo算法相对于Scoreboard算法的优点有：

1. Tomasulo 算法通过使用保留站来实现动态调度，允许指令乱序执行。这使得算法更好地适应指令流的变化，提高了并行度。
2. Tomasulo 算法通过使用寄存器重命名和保留站来处理数据相关性，允许指令在等待数据就绪时继续执行，而不必等待之前的指令执行完成。这有助于提高指令流的效率。
3. Tomasulo 算法在不同的功能单元中使用多个保留站，这使得它能够有效地支持多种不同类型的指令，包括浮点运算、加载/存储等。

Tomasulo存在的缺点：

1. 实现 Tomasulo 算法需要更多的硬件支持，包括多个保留站和重命名寄存器等。这增加了硬件的复杂性和成本。
2. 在竞争激烈的情况下，多个指令可能竞争相同的保留站或功能单元，导致资源争用。这可能会导致性能下降。
3. Tomasulo 算法可能由于指令流水线的长度而导致一些延迟。这在一些特定的工作负载下可能成为性能瓶颈。

(2)简要介绍引入重排序改进 Tomasulo 的原理。

引入重排序的目的是解决Tomasulo算法乱序提交的问题。通过引入一个重排序缓存(ROB，Reorder Buffer)来记录指令在程序中的顺序，一条指令在执行完毕并写回结果之后并不会立马提交(此处的提交指的是修改处理器状态如修改逻辑寄存器堆)，而是先在ROB中等待，直到其前面的指令都提交完毕，才可以提交结果到逻辑寄存器堆。

(3)请分析重排序缓存的缺点。

1. ROB 的实现相对复杂，需要大量的硬件支持，且其维护成本较高，功耗相对较高。
2. ROB 的大小和设计会直接影响到指令的提交和写回的延迟。较大的 ROB 可能需要更多的时间来完成提交和写回操作，从而增加了整体延迟，会对性能造成一定的影响。
3. ROB 的实现可能引入一些资源争用和冲突。多个指令同时试图访问 ROB 中的条目时，可能引发竞争条件，需要额外的硬件来解决这些冲突。冲突和争用可能导致一些指令被延迟，降低了整体的效率。
4. ROB 在处理 Store 指令时需要将结果写回内存。对于大规模的 ROB，这可能导致对内存带宽的需求增加，特别是在存在大量 Store 指令的情况下。

5. 在异常或分支预测错误的情况下，ROB 的清理可能变得复杂。这可能导致一些指令被错误地提交或者被错误地清理，引入了时序上的错误行为。