实验二：托马斯路算法模拟与分析 实验报告

计 71 庞博予 2017011315

一、实验概述

本次实验中，本系统实现对 NEL 汇编指令的解释执行，并基于此实现了一种非流水线的基本算法作为参考实验，以及托马斯路算法和基于托马斯路的 JUMP 语句分支预测算法。

此外，设计了 2 组样例，分别是冒泡排序和素数检测，并在自测样例和标准样例上分别测试 tomasulo 算法的正确性和其分支预测算法带来的性能提升。

二、程序框架与设计

1．基本框架与非流水线模拟算法

为了能够方便和系统地完成实验，首先需要搭建可以实现托马斯路算法的运行环境。首先就是要将字符串格式的汇编指令转化成可以用程序处理的信息流。因此设计一个类 Instr 来表示一条汇编指令，其记录了汇编指令的操作类型和三个操作数。汇编代码可以看作由汇编指令类 Instr 构成的列表。作为输入，系统中实现了从文件中逐行读取指令并解析成汇编指令列表并加载到模拟器中。

实验中的所有常数均写在 config.h 中，包括用于区分不同指令类型的指令编号，和保留站大小、运算器个数等。

作为实验中的参考模拟器，本系统中设计了一种很简单粗暴的模拟器，按照每次读取一条指令后执行该指令执行完毕后写回，再读取下一条指令的次序，一条条顺序执行指令。用该模拟器执行汇编程序所需要的周期可以作为托马斯路算法的执行周期数的参考。其执行过程和结束时的寄存器的值也可以作为托马斯路算法正确性的参考。

2．基础托马斯路算法

托马斯路算法中每一个周期内要做的事情包括：发射指令、执行指令、写回。

其中，发射指令的条件是待发射指令所需要的功能部件至少有一个空余。发射指令时需要检查该指令设计的操作数是否有被重命名，若重命名则需要用其别名（该寄存器对应的保留站编号）代替该寄存器否则用寄存器数值直接填入；也需要检查该指令的目标寄存器和保留站是否要在当前周期内写回，若是，则将涉及该保留站的写回操作优先完成，否则会使新发射的指令重命名混乱。

发射指令如果立即就绪也需要在下一个周期内执行，为了实现这一点，在每个周期开始时检查是否有排队中的就绪指令可以送入功能部件执行，之后再发射新的指令，然后才是检查指令是否就绪。这样一条指令最快也是在第一周期内发射并就绪，第二周期送入功能部件执行。

同样，第一周期内执行完成的指令要在第二周期内完成写回，不能在指令执行结束的当前周期将该运行结果广播到保留站而需要用记录下来在下一个周期内的功能部件执行并产生新的写回需求之前进行写回更新。

而为了同时满足以上的次序，本系统中每个周期中执行各步骤的次序是：

1) 将就绪的指令按 FIFO 顺序送入空闲功能部件
2) 检查是否可以发射新的指令，可以则发射
3) 执行上一周期内记录的写回操作并清空记录
4) 检查是否有指令进入就绪状态，有则进入等待执行队列
5) 功能部件执行一个周期，如果执行结束则将待写回结果记录

实现功能部件时为了方便处理，只有一个类，通过设置不同的参数来初始化成特定功能，调用 tick(int & ret)可以使其“执行”一个周期，若未运行完成则返回 false，运行完成则返回 true 同时 ret 中记录运行结果。功能部件类也有一个 public 的 busy 成员变量用于实时检查该部件是否还在执行状态。

保留站除了记录指令类型、源寄存器重命名、操作数值、目标寄存器，同时也记录了该保留站是否正处于忙碌状态、正由哪个功能部件执行、在源程序中的行号、就绪的周期（用于就绪时排队等待执行）。

每一条“写回”也用一个结构体记录，分别记录了写回的目标寄存器、写回操作来源的保留站编号、以及写回的值。

此外需要注意的一点是，算法运行的结束条件并不是将所有指令发射，而是保留站所有行为空，此时所有指令全部执行完成，才是真正执行完成的状态。在指令全部发射而未执行完成期间，要防止非法取指令导致的错误。

3．JUMP 实现与分支预测

最基础的 JUMP 指令实现便是引入一个 block 变量记录当前是否出现了发射但还未运行完的 JUMP 指令，如果有则视为阻塞、不再发射后续指令直到该 JUMP 指令对应的功能部件执行完成，根据执行结果更新下一条指令的地址并取消阻塞继续执行。

具体为在发射指令时检查指令类型，若为 JUMP 指令，则在将指令填入保留站后立即将 block 置为 true（阻塞）。注意发射指令时要根据指令类型决定下一条指令的位置，如果是非 JUMP 语句则下一条执行的指令即为当前指令行数+1 的指令，但如果使用分支预测算法就不一定了。

进行取指时增加一条判断，若当前处在阻塞状态，和功能部件满额同样不能发射指令。

在指令执行结束时也要加一条判断，如果指令类型是 JUMP，不进行写回（也没有可以写回的寄存器）只修改下一条指令地址。

在基础的 JUMP 实现上可以进一步实现分支预测。分支预测的大体思路是在遇到跳转语句时根据“经验”预测一个分支进行执行，若后来验证该预测正确则继续执行否则回退恢复现场。而这一思想实现成为代码时就产生了许多细节。

其一是不能连续两次分支预测，虽然理论上可以但是会产生很多不可预料的麻烦；所以当已经进行过一次分支预测后，程序的执行状态就需要从“真实”态切换到“预测”态，预测态时再遇到跳转指令类似基础方法执行阻塞，直到上一条跳转指令得到结果，此时解除预测态和阻塞态，提交现场或恢复现场。

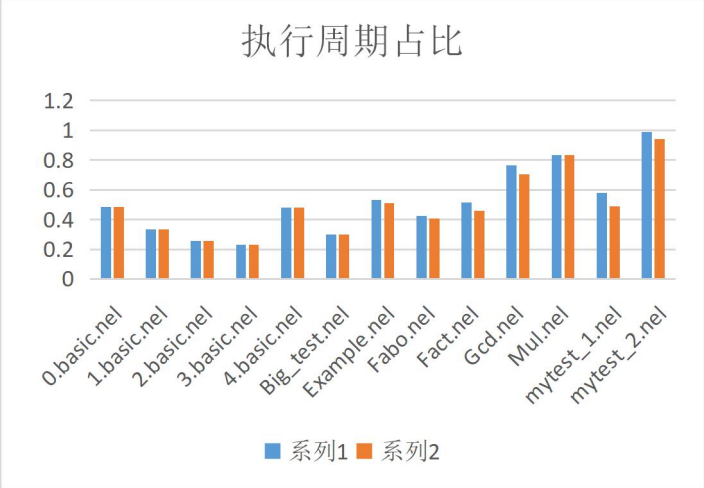
其二是进行分支预测后，之前未执行完的语句到底是更新被暂停的真实态寄存器还是更新继续执行的预测态寄存器还是两边一起更新；分支预测之后执行完的语句又该如何？经过思考可得出结论：“真实态”语句可以同时更新真实态和预测态寄存器、而“预测态”语句只能更新预测态寄存器。

其三是提交现场时，用预测态寄存器覆盖真实态寄存器，并将保留站中所有预测态语句修改为真实态语句，运行状态不变；而恢复现场时，保留真实态寄存器，放弃预测态寄存器，清除保留站中所有的预测态语句，继续执行。因为真实态语句无论是否执行完都会更新到真实态寄存器中，所以不论预测态如何，真实态仍然是时刻保持正确性的。

为了实现上述细节，需要对保留站增加一维信息表示该站中的指令是在真实态发射的还是在预测态发射的；在写回结构体中也需要增加一维来记录该写回是一条真实态写回还是一条预测态写回。同时也要开辟为预测态准备的临时寄存器和重命名空间。

三、运行结果与分析

样例	非流水线周期数	Tomasulo 周期数	分支预测周期数	预测准确率
0.basic.nel	33	16	16	
1.basic.nel	171	57	57	
2.basic.nel	148	38	38	
3.basic.nel	147	34	34	
4.basic.nel	131	63	63	
Big_test.nel	5000001	1500003	1500003	
Example.nel	47	25	24	33.3%
Fabo.nel	113	48	46	60.0%
Fact.nel	537	276	247	96.7%
Gcd.nel	83155840	63465740	58571713	85.5%
Mul.nel	6107	5086	5086	
mytest_1.nel	281	163	137	47.6%
mytest_2.nel	4057	4010	3812	99.0%



上图中比例表示该方法在该测例上执行周期数与非流水线参考算法执行周期数之比

四、样例设计与分析

1、冒泡排序（正数）

//mytest_1.nel

```
LD,R1,0x3
LD,R2,0x4
LD,R3,0x1
LD,R4,0x5
LD,R5,0x2
LD,R0,0x0
LD,R9,0x1
LD,R8,0x0
DIV,R6,R1,R2
JUMP,0x0,R6,0x5
ADD,R7,R0,R2
ADD,R2,R1,R0
ADD,R1,R7,R0
ADD,R8,R0,R9
DIV,R6,R2,R3
JUMP,0x0,R6,0x5
ADD,R7,R0,R3
ADD,R3,R2,R0
ADD,R2,R7,R0
ADD,R8,R0,R9
DIV,R6,R3,R4
JUMP,0x0,R6,0x5
```

```
ADD,R7,R0,R4
ADD,R4,R3,R0
ADD,R3,R7,R0
ADD,R8,R0,R9
DIV,R6,R4,R5
JUMP,0x0,R6,0x5
ADD,R7,R0,R5
ADD,R5,R4,R0
ADD,R4,R7,R0
ADD,R8,R0,R9
JUMP,0x1,R8,0xFFFFFFE7
JUMP,0x0,R10,0x1
```

功能：在 r1~r5 中放入五个正数，采用冒泡法对这五个数进行排序。

非流水线：281 周期

Tomasulo：163 周期

分支预测：137 周期，预测正确率 47.619%。

由于该程序中执行的跳转语句基本上均为相邻元素比较，故正确率为 50%左右是正常的。

2、检测素数

//mytest_2.nel

```
LD,R1,0x28A3
LD,R2,0x2
LD,R3,0x0
LD,R4,0x1
LD,R5,0x0
DIV,R6,R1,R2
MUL,R7,R2,R6
SUB,R6,R1,R7
JUMP,0x0,R6,0x2
JUMP,0x0,R5,0x4
ADD,R3,R5,R4
ADD,R8,R5,R2
DIV,R9,R1,R8
ADD,R2,R2,R4
MUL,R6,R2,R2
ADD,R7,R1,R6
DIV,R6,R7,R1
JUMP,0x2,R6,0x2
JUMP,0x1,R6,0xFFFFFFF3
ADD,R0,R5,R3
JUMP,0x0,R5,0x1
```

功能：在 R1 中放入一个正数，运行程序后 R0=0 表示该数是素数，R0=1 表示是合数，此时 R8 和 R9 是该合数的一个分解。

非流水线：4057 周期

Tomasulo：4010 周期

分支预测：3812 周期，预测正确率 98.997%。

五、运行方法

编译：将路径设到/src/目录下，输入 make 编译。

运行：测例都放在/input/目录下，且与约定的名称一致；保留可/input/中原有的自测样例。在/src/路径下./main -test 即可以测试模式执行所有测例。

同时程序也支持多种运行参数：

./main -detail 可以实时显示每一个周期的运行细节；

./main -a/-b/-t/-e/-p [-r/-p]中，第一个参数表示用哪一类测例进行测试，其中-a 表示全部，-b 表示基础测例，-t 表示使用自测样例，-e 表示用拓展测例，-p 表示用性能测例。第二个参数-r 表示用非流水线参照算法，-p 表示用分支预测 tomasulo 算法，不加表示用默认 tomasulo 算法。结果将放在/log/下*.log 文件中。