**計算機結構程式專題二報告**

學校: 國立臺南大學

學生: 蔡易霖

**如何執行**

此專案使用Java程式語言撰寫，提供三種執行方式:

1. 使用 javac 編譯並執行
2. 使用包裝檔 jar 執行
3. 使用 .exe 執行

**使用 javac 編譯並執行**

在 tomasulo.java 檔案目錄下執行以下指令來編譯 .java 檔案。

#編譯

javac -encoding UTF-8 tomasulo.java

執行指令時給予fileName檔名。

#執行

java tomasulo ./test/test1.txt

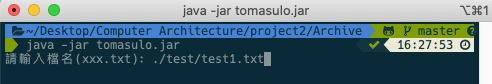
檔案位置填入相對路徑



**使用包裝檔 jar 執行**

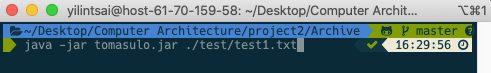
程式碼已經預先打包成 jar 檔案，可以直接執行指令，並依照提示給予參數。

java -jar tomasulo.jar



再者可以直接在執行指令時給予參數，fileName。

java -jar tomasulo.jar ./test/test1.txt



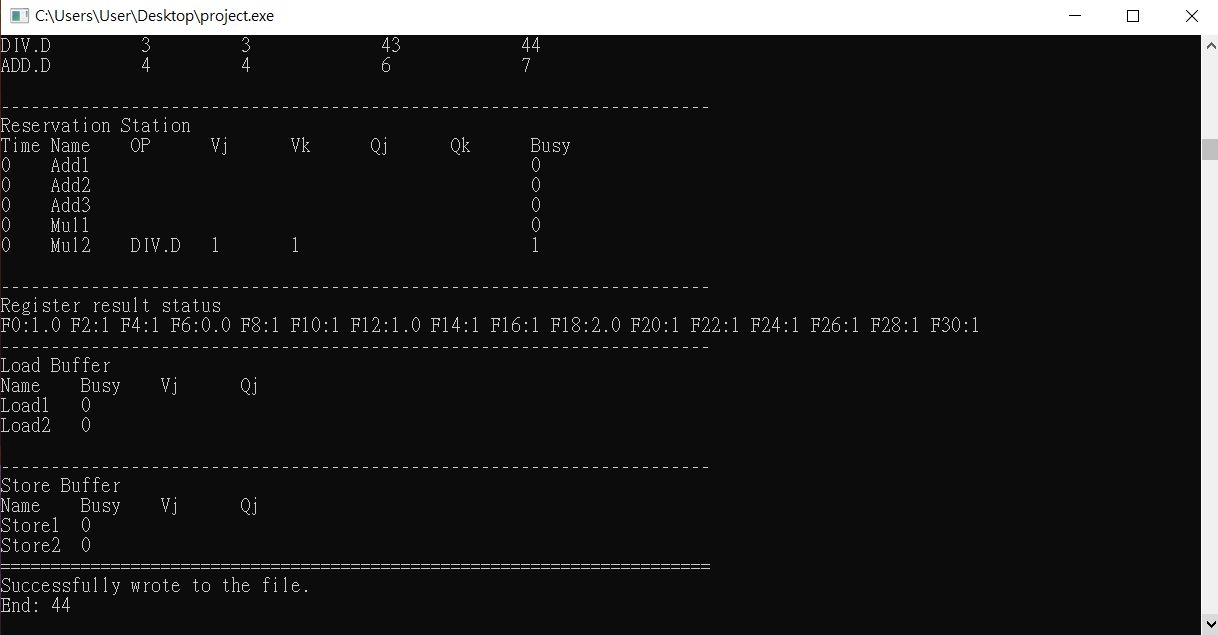
test1.txt 需與 .jar 檔放置於同一個資料夾位置，或是使用相對路徑

**使用 .exe 執行**

在 windows 環境下打開執行檔，依據提示輸入指定參數。



執行結果將會印在終端機上，且會產生出一個output.txt檔案在該資料夾下





Output.txt 內紀錄了每一個週期的Instruction Status包含指令類型、Issue、開始執行週期、執行結束週期、寫回週期。第二個Reservation Station則記錄了每一個資源的佔用狀況。第三個為浮點數暫存器的Register result status。第四個為Load/Store Buffer。

## 程式架構說明

此專案自定義了兩個 Class ，分別為 Instruction指令格式的儲存以及ReservationStation保留站的控管。讀檔函式負責處理檔案每一行執行指令分別有opcode rd, rs, rt。此函式透過讀檔解析每一行的指令並依序放入 instructionList 中。程式的主流程為讀檔→執行tomasulo演算法。由於程式碼過多說明文件就不逐行說明，程式碼內有完整的程式註解。以下簡單描述我專案二實作的tomasulo演算法執行流程。

#### **Issue:**

1. 檢查目前指令是否可以被 Issue
2. 檢查該指令是否有 data dependence(RAW、WAR、WAW) 狀況發生
3. 判斷目前是否有保留站可以被使用

#### **Execution:**

1. 檢查所有的保留站，能立即執行的就開始執行
2. 檢查指令是否可以結束執行

#### **WriteResult**

1. 檢查全部的保留站，是否已經可以寫入結果
2. 若已經執行結束的指令，將會廣播更新保留站的Qj或Qk並寫入結果
3. 清空已經執行完畢的保留站並釋放空間

#### **程式相關參數說明:**

* Reservation Station  
  有loadBuffer、storeBuffer、adder、multiplier，分別有2、2、3、2個資源。
* Register Status

1. 浮點數暫存器有16個，編號為F0、F2、F4、…、F30，初始值為1
2. 整數暫存器有32個，編號為R0、R1、…、R31，除R1的初始值為16外，其餘整數暫存器初始值為0

* 指令執行週期  
  L.D=2、S.D=1、MUL.D=10、ADD.D=2、DIV.D=40

在此專案中Load實作不考慮load-store queue為第一個才開始執行

## 測試結果

test1.txt (20%) no dependence

MUL.D F0, F2, F4

SUB.D F6, F8, F10

DIV.D F12, F14, F16

ADD.D F18, F20, F22

|  |  |  |
| --- | --- | --- |
| 1 | 11 | 12 |
| 2 | 4 | 5 |
| 3 | 43 | 44 |
| 4 | 6 | 7 |

test2.txt (20%) true dependence

L.D F6, 8(R1)

ADD.D F0, F2, F6

DIV.D F8, F0, F4

SUB.D F10, F6, F8

MUL.D F12, F10, F30

S.D F12, 40(R1)

|  |  |  |
| --- | --- | --- |
| 1 | 3 | 4 |
| 2 | 6 | 7 |
| 3 | 47 | 48 |
| 4 | 50 | 51 |
| 5 | 61 | 62 |
| 6 | 7 | 63 |

test3.txt (20%) true dependence + anti dependence

L.D F6, 8(R2)

L.D F2, 40(R3)

ADD.D F4, F2, F6

DIV.D F8, F0, F4

SUB.D F0, F6, F2

MUL.D F12, F6, F4

S.D F12, 40(R3)

|  |  |  |
| --- | --- | --- |
| 1 | 3 | 4 |
| 2 | 4 | 5 |
| 3 | 7 | 8 |
| 4 | 48 | 49 |
| 5 | 7 | 8 |
| 6 | 18 | 19 |
| 7 | 8 | 20 |

test4.txt (20%) true dependence + anti dependence + output dependence

L.D F0, 0(R1)

L.D F2, 0(R1)

ADD.D F2, F2, F2

MUL.D F12, F6, F4

DIV.D F8, F12, F4

SUB.D F8, F6, F2

S.D F8, 16(R1)

S.D F12, 8(R1)

|  |  |  |
| --- | --- | --- |
| 1 | 3 | 4 |
| 2 | 4 | 5 |
| 3 | 7 | 8 |
| 4 | 14 | 15 |
| 5 | 55 | 56 |
| 6 | 10 | 11 |
| 7 | 8 | 12 |
| 8 | 9 | 16 |

test5.txt (20%) true dependence + anti dependence + output dependence

L.D F0, 0(R1)

L.D F2, 0(R0)

DIV.D F4, F0, F2

DIV.D F8, F6, F4

DIV.D F10, F8, F4

SUB.D F8, F0, F2

|  |  |  |
| --- | --- | --- |
| 1 | 3 | 4 |
| 2 | 4 | 5 |
| 3 | 45 | 46 |
| 4 | 86 | 87 |
| 47 | 127 | 128 |
| 48 | 50 | 51 |