

# 說明文件

## ★ Input:

```
ADDI F1, F2, 1
SUB F1, F3, F4
DIV F1, F2, F3
MUL F2, F3, F4
ADD F2, F4, F2
ADDI F4, F1, 2
MUL F5, F5, F5
ADD F1, F4, F4
```

## ★ 我自己的手算模擬:

Inst.	Ts	Ex	Wr	
ADDI F1, F2, 1	1	2	4	SUB ADD : 2
SUB F1, F3, F4	2	4	6	MUL : 1D
DIV F1, F2, F3	3	4	44	DIV : 4D
MUL F2, F3, F4	4	44	54	
ADD F2, F4, F2	5	55	51	
ADDI F4, F1, 2	6	45	47	
MUL F5, F5, F5	44	54	64	
ADD F1, F4, F4	45	48	50	

## ★ Output 範例有另外附上。

- ★ 我有三個 struct 分別用來宣告我的 input ( data ) 、 RS 、 buffer 。

```
8     typedef struct A//data
9     {
10         string Reg;
11         int Fa;
12         int Fb;
13         int Fc;
14     }A;
15
16
17     typedef struct B//rs
18     {
19         char Operator;
20         int b;
21         int c;
22         bool ready_all = false;//true means ready
23         bool ready_b = false;
24         bool ready_c = false;
25         bool empty = true;//true means empty
26     }B;
27
28     typedef struct C//buffer
29     {
30         int rs;
31         int b;
32         char oper;
33         int c;
34         int cycl;//紀錄run幾個cycle後可以wr
35     }C;
36
37     int RF[6] = { -1,0,2,4,6,8 };//第一個-1不要用
38     vector<int> RAT[5] = {};//始終指每項只能有一個數字
39     //vector<stack<int>>RAT[5];
40     B RS_add[3] = {};
41     B RS_mul[2] = {};
42     int cycle = 0;
43     const int add_cycle = 2, mul_cycle = 10, div_cycle = 40;
44     bool unit_add = true;//ALU unit//true means empty
45     bool unit_mul = true;
46     C buffer_add = {};
47     C buffer_mul = {};
48     bool out = false;//決定output與否
49     int RS_posa, RS_posm;//紀錄現在正在unit執行的RS為RS幾
50     //防止遞迴呼叫時多做
51     bool checkdis_a = false;//true時表示再次呼叫 但總共只讓他做一次
52     bool checkdis_m = false;
```

- ★ Main 裡面讀檔後將資料放入 data queue ，然後就呼叫 Tomasulo 開始模擬 。

```
505             Tomasulo(Data);
506
507             infile.close();
508             return 0;
509 }
```

## ★ Tomasulo:

```
391 void Tomasulo(queue<A>& data)
392 {
393     bool first = true; //第一次的話如果可以 dispatch要在下一個cycle才可以做
394
395     while (!data.empty())
396     {
397         cycle++;
398
399         if (!first) dispatch();
400         else first = false;
401         A u = data.front();
402         //cout << u.Reg << " " << u.Fa << "," << u.Fb << "," << u.Fc << endl;
403         if (issue(u))
404         {
405             data.pop(); //issue sucessful
406             out = true; //out有變化就要output
407         }
408
409         output();
410         out = false;
411     }
412
413     while (!unit_add || !unit_mul) //unit還有東西所以要繼續算下去
414     {
415         cycle++;
416         dispatch();
417         output();
418         out = false;
419     }
420 }
421 }
```

第一次進來的話直接 issue 就好，第一次以外的我會先做 dispatch 再 issue 因為我 capture 會在 dispatch 裡呼叫 write\_back 做，然後 issue 成功的時候 inst. 會 pop 掉。這整組的循環會做到 data 為空為止，即為沒東西可以 issue 時，我再接續著做 dispatch 直到所有 operator unit 都閒置下來，結束。

## ★ Issue:

```
55 bool issue(A inst)
56 {
57     bool find_empty = false;
58     int whrs; //where is rs
59
60     //RS有空位才可以進
61     if (inst.Reg == "ADDI" || inst.Reg == "ADD" || inst.Reg == "SUB")
62     {
63         for (int i = 0; i < 3; i++) //從第一個開始找有空的RS
64         {
65             if (RS_add[i].empty()) // RSi ----> ex:i=0 ,表RS1//要加1
66             {
67                 RS_add[i].empty = false; //更新為不為空
68                 find_empty = true; //RS有位子可以issue
69                 whrs = i + 1;
70
71                 // operator: +
72                 if (inst.Reg == "ADDI" || inst.Reg == "ADD") RS_add[i].operator_ = '+';
73                 // operator: -
74                 else RS_add[i].operator_ = '-';
75
76                 //check RAT
77                 //*****check inst.b 在RAT中有無東西*****
78                 if (RAT[inst.Fb - 1].empty()) //RAT沒東西
79                 {
80                     RS_add[i].b = RF[inst.Fb]; //則Fx值為RF裡的值
81                     //RAT[inst.Fa - 1].push_back(i + 1); //RAT給予對應的RSi//inst.Fa表示為Fx//ex: inst.Fa=1 表F1
82                     RS_add[i].ready_b = true; //RS.b is already
83                 }
84                 else //RAT不為空//要等待別人write back
85                 {
86                     //RS_add is not ready
87                     RS_add[i].ready_b = false;
88                     RS_add[i].b = RAT[inst.Fb - 1][0]; //給予RS編號
89                 }
90             }
91         }
92     }
93 }
```

```

1093319-project3 (主文件夹)
88     RS_add[i].b = RAT[inst.Fb - 1][0];//給予RS編號
89     //RAT[inst.Fa - 1].push_back(i + 1);//RAT給予對應的RSi//inst.Fa表示為Fx//ex: inst.Fa=1 表F1
90   }
91
92   /*****check inst.c 在RAT中有無東西*****
93   if (inst.Reg == "ADDI")
94   {
95     RS_add[i].c = inst.Fc;//addi , so inst.Fc is a number//ready is already true
96     RS_add[i].ready_c = true;
97   }
98   else//add && sub
99   {
100    if (RAT[inst.Fc - 1].empty())//RAT沒東西
101    {
102      RS_add[i].c = RF[inst.Fc];//則Fx值為RF裡的值
103      RS_add[i].ready_c = true;
104    }
105    else//RAT不為空//要等待別人write back
106    {
107      //RS_add is not ready
108      RS_add[i].ready_c = false;//就算RS.b is true但這邊RS.c還是false
109      RS_add[i].c = RAT[inst.Fc - 1][0];//給予RS編號
110    }
111  }
112
113  /*****inst.a 在RAT中有無RS*****
114  if (!RAT[inst.Fa - 1].empty())RAT[inst.Fa - 1].clear();//若RAT已有RS則先清掉在新增
115  RAT[inst.Fa - 1].push_back(i + 1);//RAT給予對應的RSi//inst.Fa表示為Fx//ex: inst.Fa=1 表F1
116
117  /**更新整體ready state**/
118  if (RS_add[i].ready_b && RS_add[i].ready_c)RS_add[i].ready_all = true;//already for dispatch
119  break;
120 }
121 }

1093319-project3 (主文件夹)
121
122 }
123 else//MUL DIV ----> RS編號+3
124 {
125   for (int i = 0; i < 2; i++)//從第一個開始找有空的RS
126   {
127     if (RS_mul[i].empty)
128     {
129       RS_mul[i].empty = false;//更新為不為空
130       find_empty = true;//RS有位子可以issue
131       whrs = i + 1;
132
133       // operator: +
134       if (inst.Reg == "MUL")RS_mul[i].Operator = '*';
135       // operator: /
136       else RS_mul[i].Operator = '/';
137
138       //check RAT
139       /*****check inst.b 在RAT中有無東西*****
140       if (RAT[inst.Fb - 1].empty())//RAT沒東西
141       {
142         RS_mul[i].b = RF[inst.Fb];//則Fx值為RF裡的值
143         RS_mul[i].ready_b = true;//RS.b is already
144       }
145       else//RAT不為空//要等待別人write back
146       {
147         RS_mul[i].ready_b = false;
148         RS_mul[i].b = RAT[inst.Fb - 1][0];//給予RS編號
149       }
150
151       /*****check inst.c 在RAT中有無東西*****
152       if (RAT[inst.Fc - 1].empty())//RAT沒東西
153       {
154         RS_mul[i].c = RF[inst.Fc];//則Fx值為RF裡的值
155       }
156     }
157   }
158 }

```

```

154     RS_mul[i].c = RF[inst.Fc];//則Fx值為RF裡的值
155     RS_mul[i].ready_c = true;
156   }
157   else//RAT不為空//要等待別人write back
158   {
159     RS_mul[i].ready_c = false;//就算RS.b is true但這邊RS.c還是false
160     RS_mul[i].c = RAT[inst.Fc - 1][0];//給予RS編號
161   }
162
163   //*****inst.a 在RAT中有無RS*****
164   if (!RAT[inst.Fa - 1].empty())RAT[inst.Fa - 1].clear();//若RAT已有RS則先清掉在新增
165   RAT[inst.Fa - 1].push_back(i + 4);//RAT給予對應的RSi//inst.Fa表示為Fx//ex: inst.Fa=1 表F1
166
167   /****更新整體ready state***/
168   if (RS_mul[i].ready_b && RS_mul[i].ready_c)RS_mul[i].ready_all = true;//already for dispatch
169   break;
170 }
171
172 }
173
174 return find_empty;
175

```

主要想法就是：

- 1.IQ 進 instruction 到 RS。
- 2.check RAT 中有無紀錄，沒有的話值就是 RF 的值，有的話就是 RS 未知數。
- 3.RAT 裡對應的 F 紿予 RS 編號，如果已有編號->取代
- 4.IQ pop

細部操作請洽註解，然後我這會回傳一個 bool 值，true 代表 issue 成功，所以 IQ (就是我的 data ) 會 pop( )，false 則相反。

### ★ Dispatch:

分兩個 unit 做，分別是 ADD 的與 MUL 的，然後我只要 unit 是空的，我就會去 RS 裡面找 ready 的做，然後放入 buffer 裡，並給予對應運算所要花的 cycle 是多少。在我 unit 裡面有東西的時候我先檢查他的剩餘 cycle，如果不為 0 則表這個還沒算完，所以繼續 -1；為 0 的時候，代表運算完成，所以可以做 write back 了，並釋放 RS 與 unit 。

```

237 void dispatch()
238 {
239   if (unit_add && checkdis_a)checkdis_a = false;
240   else if (unit_add)//add unit is empty
241   {
242     for (int i = 0; i < 3; i++)
243     {
244       if (RS_add[i].ready_all)//選一個ready進入buffer
245       {
246         out = true;
247         RS_posa = i;
248         buffer_add.rs = i;
249         unit_add = false;
250         buffer_add.oper = RS_add[i].Operator;
251         buffer_add.b = RS_add[i].b;
252         buffer_add.c = RS_add[i].c;
253         buffer_add.cycl = 2 - 1;//假設ADD ADDi SUB 均要做2 cycle
254         break;
255       }
256     }
257   }
258   else
259   {
260     if (buffer_add.cycl == 0)//可以wr
261     {
262       out = true;
263       int result;
264       if (buffer_add.oper == '+')result = buffer_add.b + buffer_add.c;//count the result
265       else result = buffer_add.b - buffer_add.c;

```

```

265     else result = buffer_add.b - buffer_add.c;
266
267     RS_add[RS_posa] = {};//RS釋放
268     write_back(buffer_add.rs, result);//wr to that rs
269     buffer_add = {};//buffer become empty
270     unit_add = true;//become empty
271     checkdis_m = true;
272     dispatch();//unit空間釋出 所以可以找一個ready的作dispatch//直接地回會多檢一次1
273 }
274 else buffer_add.cycl--;
275 }
276
277 ****
278 if (checkdis_m)
279 {
280     checkdis_m = false;
281     return;
282 }
283 else if (unit_mul && checkdis_m)checkdis_m = false;
284 else if (unit_mul)//mul unit is empty
285 {
286     for (int i = 0; i < 2; i++)
287     {
288         if (RS_mul[i].ready_all)//選一個ready進入buffer
289         {
290             out = true;
291             RS_posm = i;
292             buffer_mul.rs = i;
293             unit_mul = false;
294             buffer_mul.opert = RS_mul[i].Operator;
295             buffer_mul.b = RS_mul[i].b;

```

```

296             buffer_mul.b = RS_mul[i].b;
297             buffer_mul.c = RS_mul[i].c;
298             //假設MUL DIV個別需要10/40個cycle
299             if (buffer_mul.opert == '/')buffer_mul.cycl = 40 - 1;
300             else buffer_mul.cycl = 10 - 1;
301             break;
302         }
303     }
304     else
305     {
306         if (buffer_mul.cycl == 0)//可以wr
307         {
308             out = true;
309             int result;
310             if (buffer_mul.opert == '/')result = buffer_mul.b / buffer_mul.c;
311             else result = buffer_mul.b * buffer_mul.c;
312
313             RS_mul[RS_posm] = {};
314             write_back(buffer_mul.rs + 3, result);//wr to that rs
315             buffer_mul = {};//buffer become empty
316             unit_mul = true;//become empty
317             //checkdis_a = true;
318             dispatch();//unit空間釋出 所以可以找一個ready的作dispatch
319         }
320     }
321 }
322 }
323

```

## ★ Write back:

```
177 void write_back(int rs, int result)//RS=rs+1
178 {
179     //capture
180     for (auto& u : RS_add)//RS中的未知數給值
181     {
182         if (u.ready_all)continue;
183         else if (u.empty)continue;
184
185         if (!u.ready_b && u.b == rs + 1)
186         {
187             u.b = result;
188             u.ready_b = true;//get value ----> become ready
189         }
190         if (!u.ready_c && u.c == rs + 1)
191         {
192             u.c = result;
193             u.ready_c = true;//get value ----> become ready
194         }
195         if (u.ready_b && u.ready_c)
196         {
197             u.ready_all = true;
198             checkdis_a = true;
199         }
200
201     for (auto& u : RS_mul)//RS中的未知數給值
202     {
203         if (u.ready_all)continue;
204         else if (u.empty)continue;
205
206         if (!u.ready_b && u.b == rs + 1)
207         {
208             u.b = result;
209             u.ready_b = true;//get value ----> become ready
210         }
211         if (!u.ready_c && u.c == rs + 1)
212         {
213             u.c = result;
214             u.ready_c = true;//get value ----> become ready
215         }
216         if (u.ready_b && u.ready_c)
217         {
218             u.ready_all = true;
219             checkdis_m = true;
220         }
221
222     //Update value of RS in RAT
223     bool wr_RF = false;
224     for (int i = 0; i < 5; i++)
225     {
226         if (!RAT[i].empty() && RAT[i][0] == rs + 1)
227         {
228             RAT[i].clear();//釋放RAT
229             wr_RF = true;//update value in RF
230             if (wr_RF)RF[i + 1] = result;
231         }
232     }
233 }
234
235 }
```

我會先做 capture，將運算結果給到對應的 RS 中，給值完後更新個別 RS 的狀態，如果 ready 就表示可以 dispatch 了，最後做 RAT 釋放以及 RF 更新。

- ★ 之後呼叫 output 與 cout\_RS 輸出結果，並且有一個 bool 值會紀錄是否有變化，有的話輸出，沒有的話則進入下一個 cycle。