ЛР № 8. Конкурентне виконання машинних інструкцій

Мета: Опанування технікою розпаралелення виконання машинних інструкцій на рівні апаратури..

Завдання: Засобами архітектурного симулятора WinMIPS64 дослідити на прикладі створених студентом фрагментів програмного коду можливості паралельного опрацювання машинних інструкцій на рівні апаратури та позитивні ефекти, що при цьому досягаються. За результатами проведених лабораторних досліджень оформити звіт та захистити його..

Методика виконання лабораторної роботи

Ми дозволимо випускати на виконання інструкції рухомої коми зі сходинки ID тоді, коли це стає можливим. Така інструкція може продовжити виконання у власному конвеєрі виконання операції рухомої коми або призупинитися через неготовність її операндів. Обрана нами стратегія дозволяє продемонструвати переваги невпорядкованого завершення виконання (out-of- order completion), але вона також може спричинити небезпеку WAR конвеєрного виконання. Тут може допомогти техніка переназв регістрів (register renaming), яку треба досконально розуміти. Наприклад, таке може статися в наступному фрагменті коду:

```
.text
add.d f7,f7,f3add.d f7,f7,f4
mul.d f4,f5,f6          ; WAR через спільний регістр f4
```

Коли випустити mul.d, тоді ця інструкція (за певних умов) «пережене» другу інструкцію add.d і першої запише до f4. Отже, mul.d мусимо затримати на ID. Structural hazards arise at the MEM stage bottleneck, as instructions attempt to exit more than one of the execute stage piplelines at the same time. Просте правило запобігання небезпек звучить так: довгі інструкції виконують першими.

Нехай маємо фрагмент коду:

```
;************************************
;*** winMIPS64 //hazard3.s//          *****
;*** (c) 2003 CA226, DCU              *****
;************************************
.text
div.d f7,f9,f10mul.d f2,f4,f3 sub.d f7,f7,f4 ld      r1,78(r0)add.d f4,f5,f6 halt
```

Результати його синтаксичного контролю.

```
Pass 1 completed with 0 errors
;************************************
;*** winMIPS64 //hazard3.s//                          *****
;*** (c) 2005 CA226, DCU                              *****
;************************************
00000000              .text
00000000 462a49c3        div.d f7,f9,f1000000004 46232082     mul.d f2,f4,f3
00000008 462439c1        sub.d f7,f7,f4
0000000c dc01004e        ld  r1,78(r0)
00000010 46262900        add.d f4,f5,f6   ; WAR on f4
00000014 04000000        halt

Pass 2 completed with 0 errorsCode Symbol Table
Data Symbol Table
```

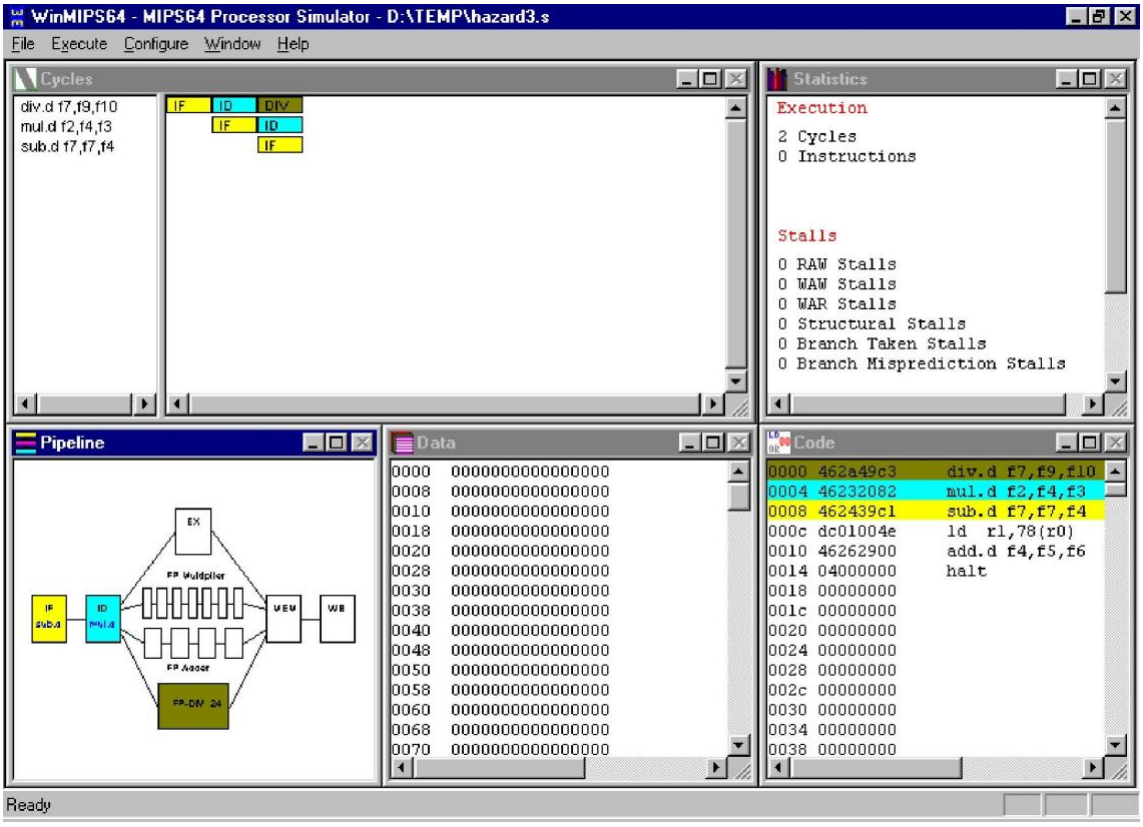

Рис. 8.1. Неконвеєрний повільний виконавчий вузол ділення з рухомою комою розпочав роботу першим

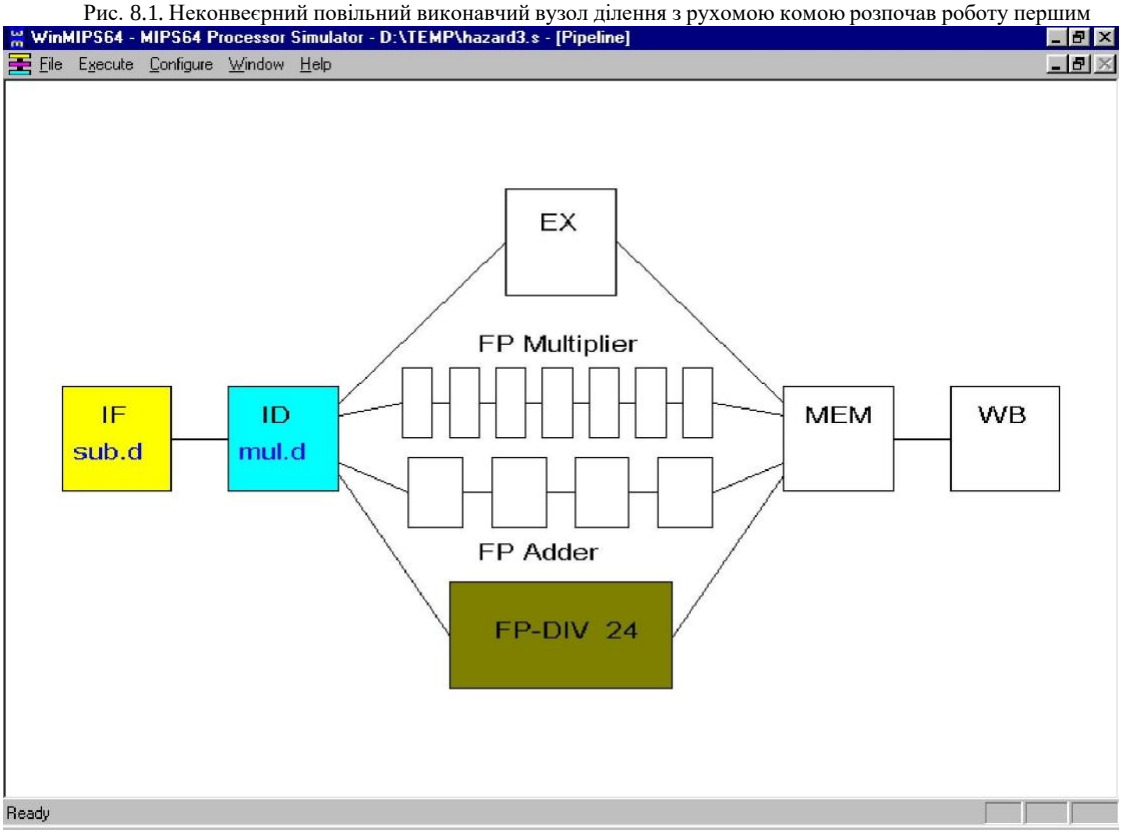

Рис. 8.2. Неконвеєрний виконавчий вузол ділення рухомої коми (24 такти затримки) завантажено

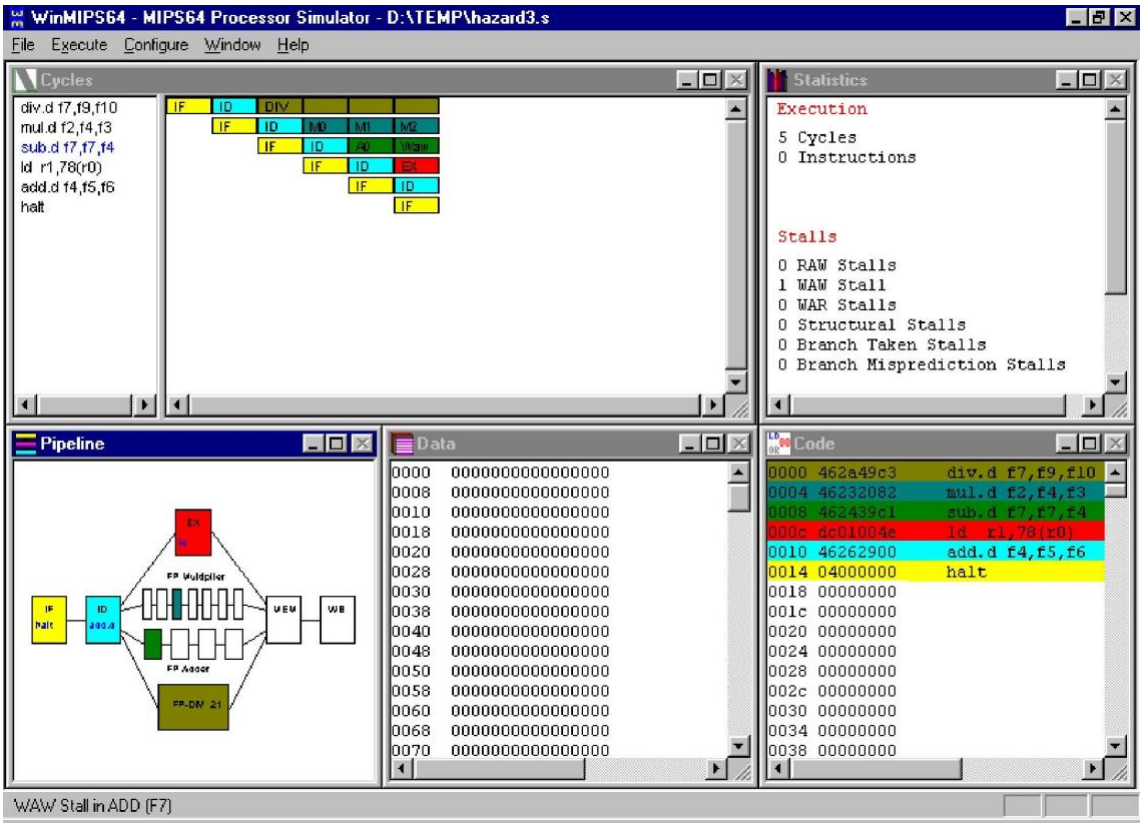

Рис. 8.3. Паралельно виконуються чотири інструкції (ясно, що з невпорядкованим завершенням)

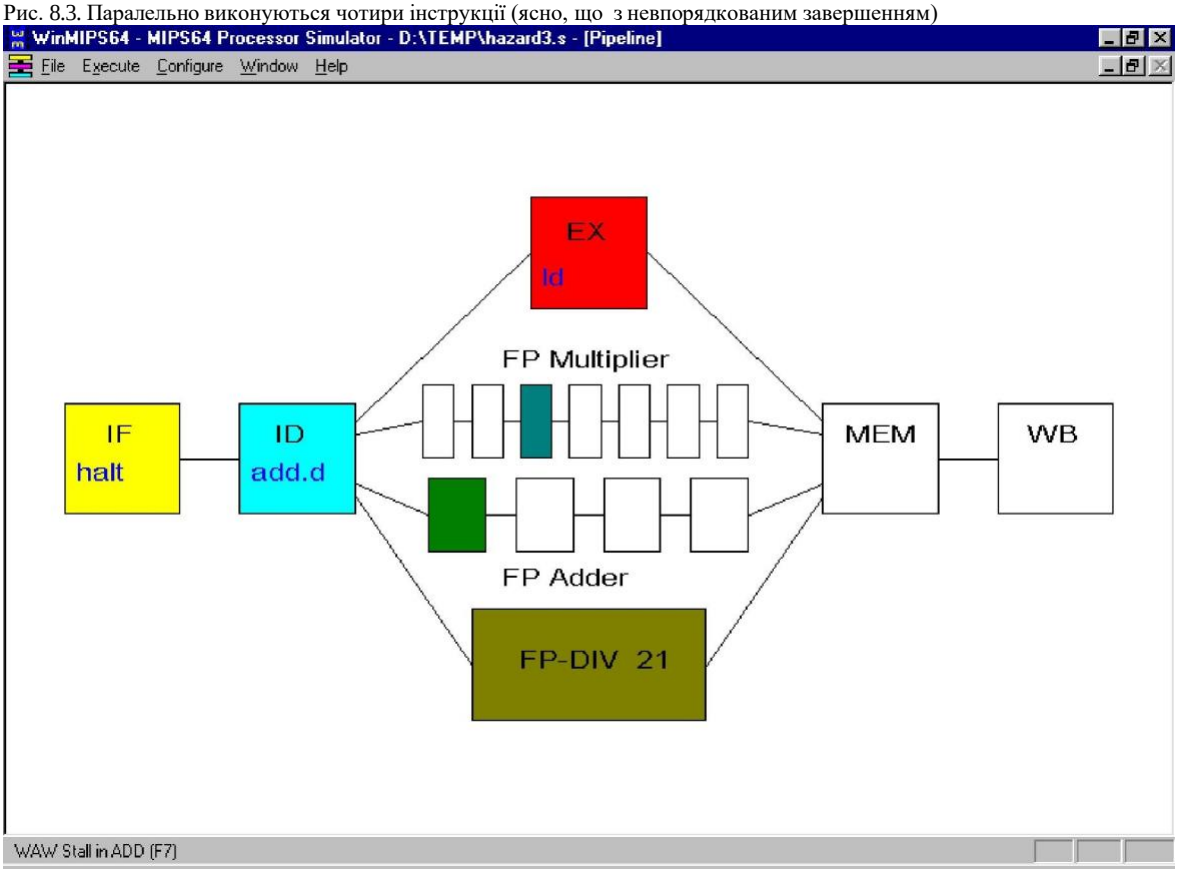

Рис. 8.4. Паралельне виконання продовжується, сходинки MEM жодна інструкція ще не досягла

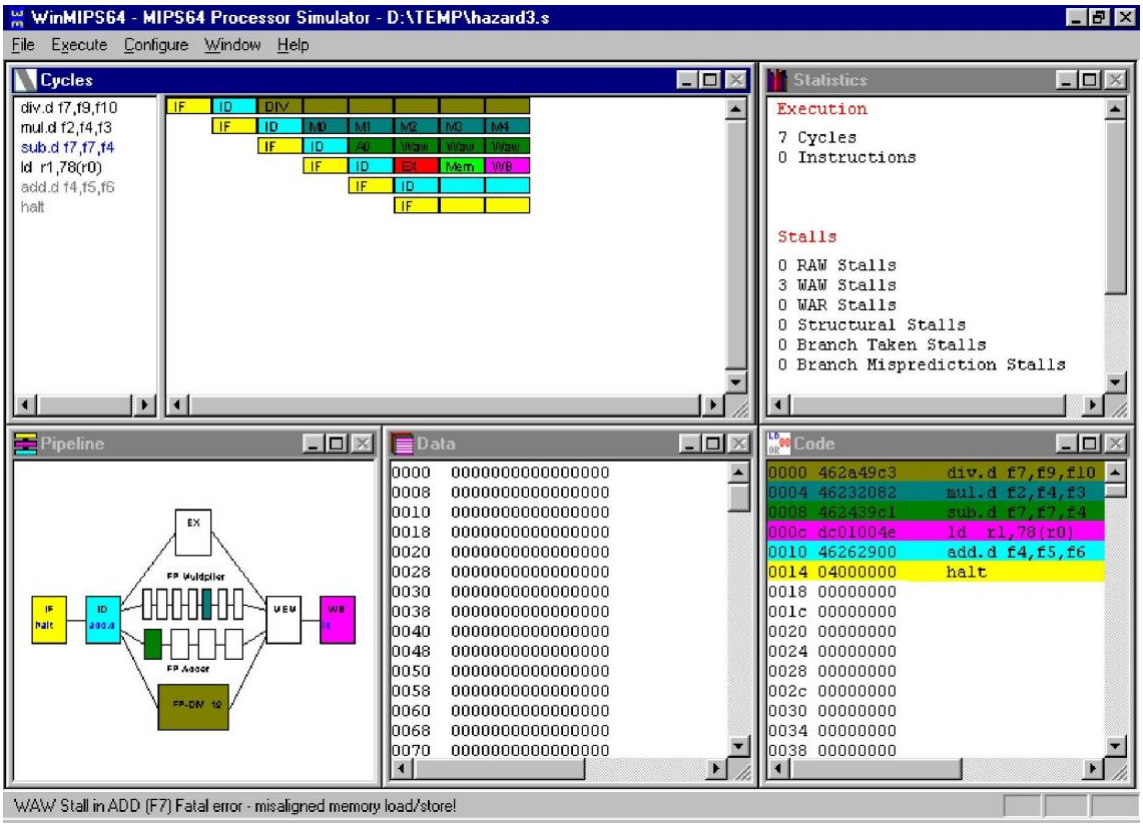

Рис.8.5. Перша інструкція ділення ще виконується, а третя від кінця ld майже завершилася

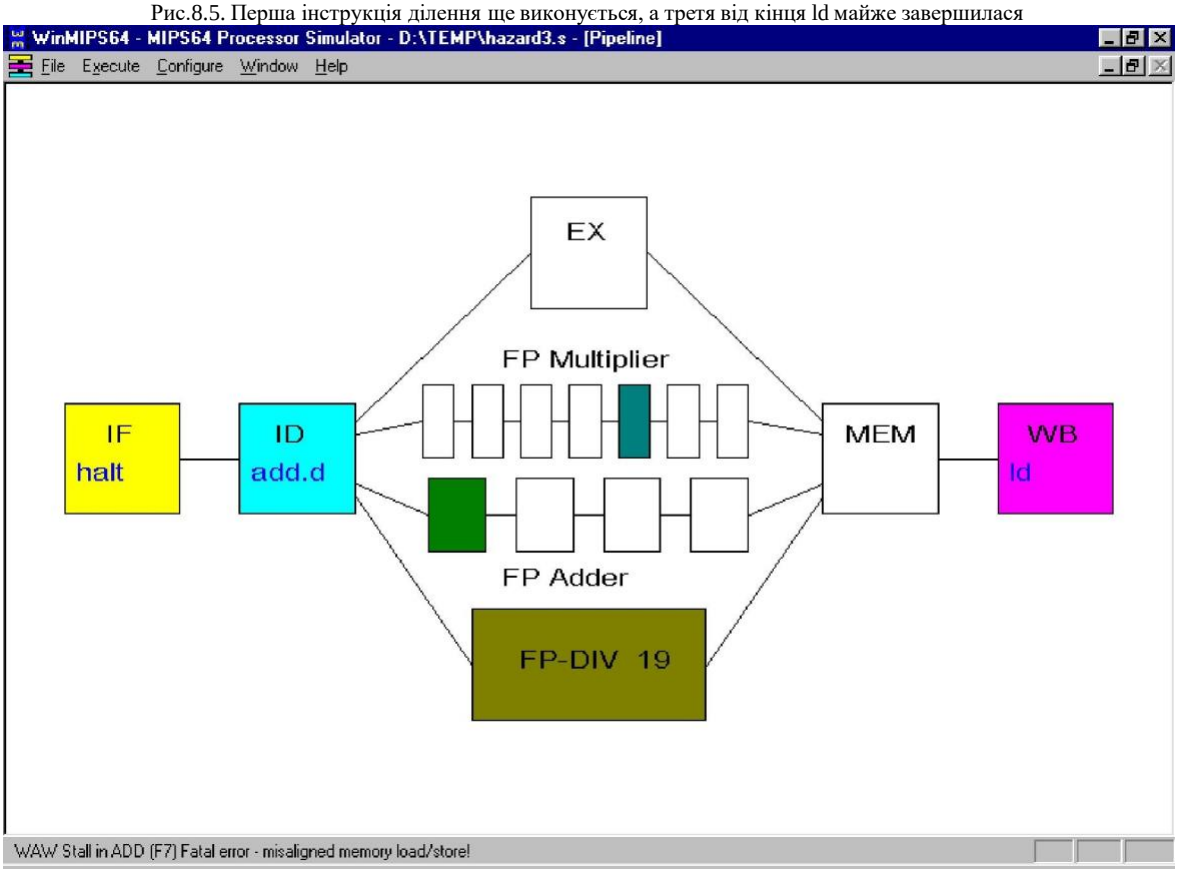

Рис. 8.6. Невпорядковане (кажуть, хаотичне) завершення виконання інструкцій потоку

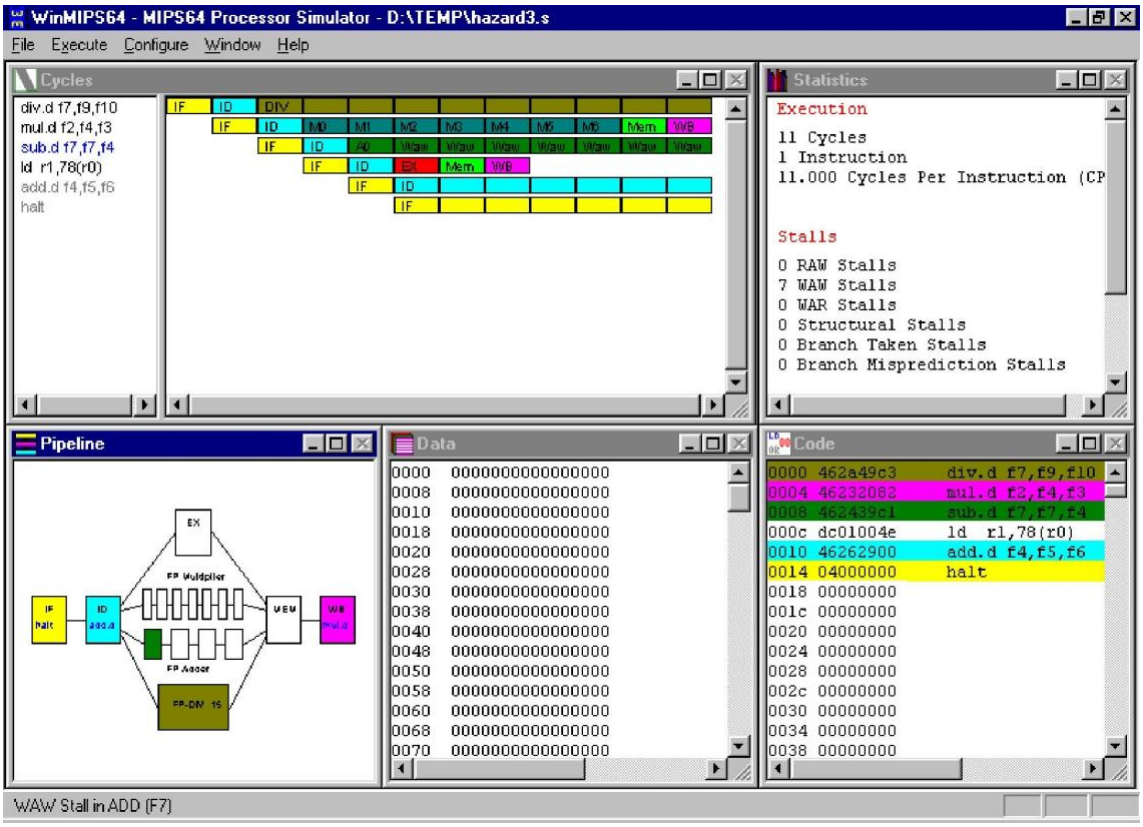

Рис. 8.7. sub.d має ту саму мету, що і div.d. Утворилася ситуація WAW, тому пригальмовані add.d та halt

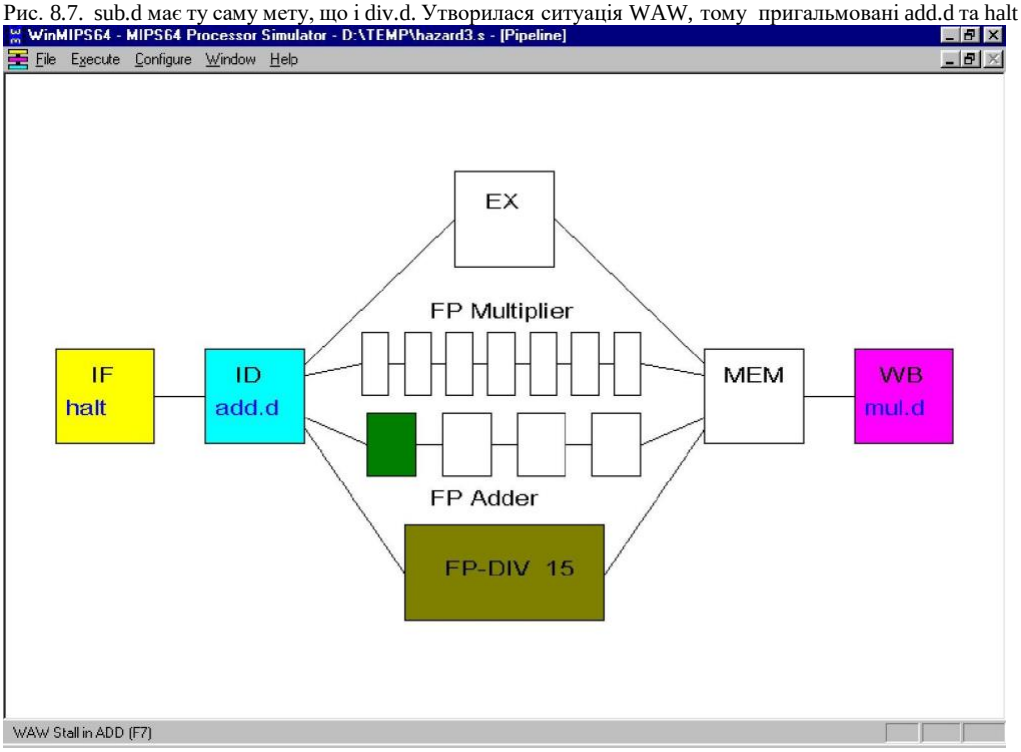

Рис.8.8. Пригальмована на першій з чотирьох виконавчих сходинок sub.d гальмує наступнуза неюadd.d

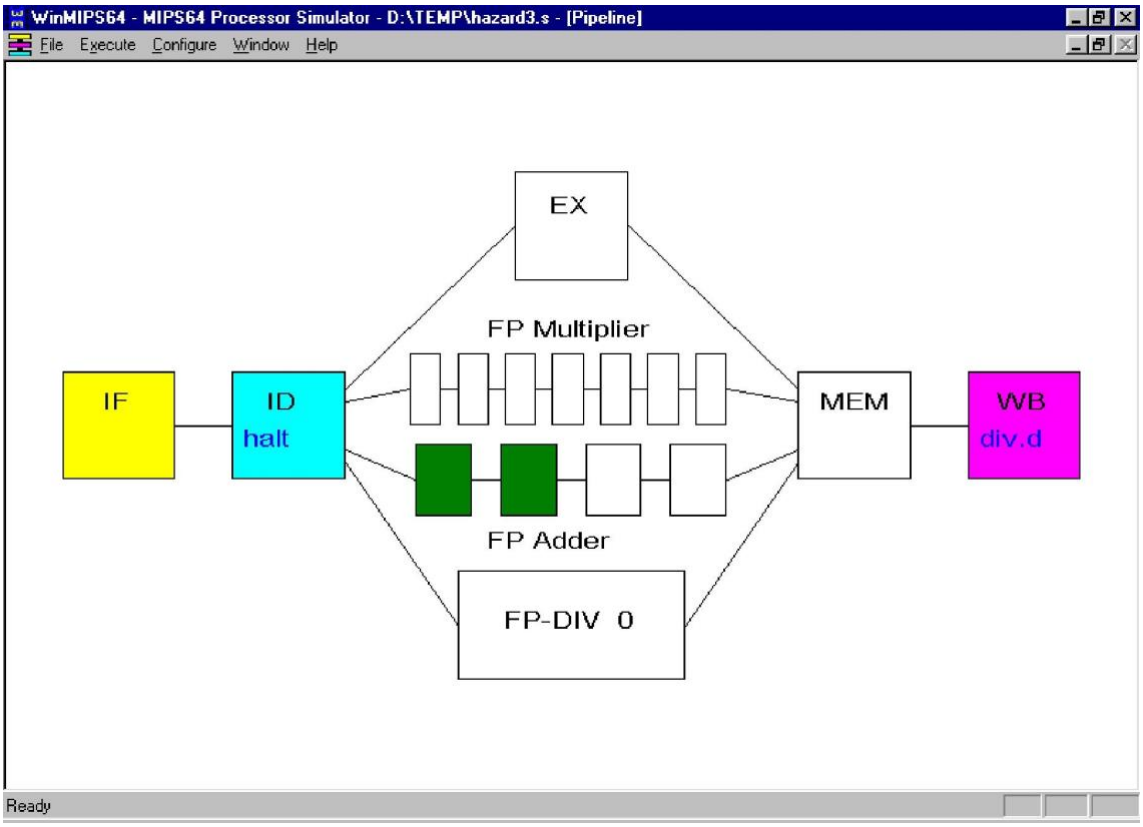

Рис. 8.9 Завершення div.d розблокувало конвеєр. На виконанні знаходяться віднімання і додавання

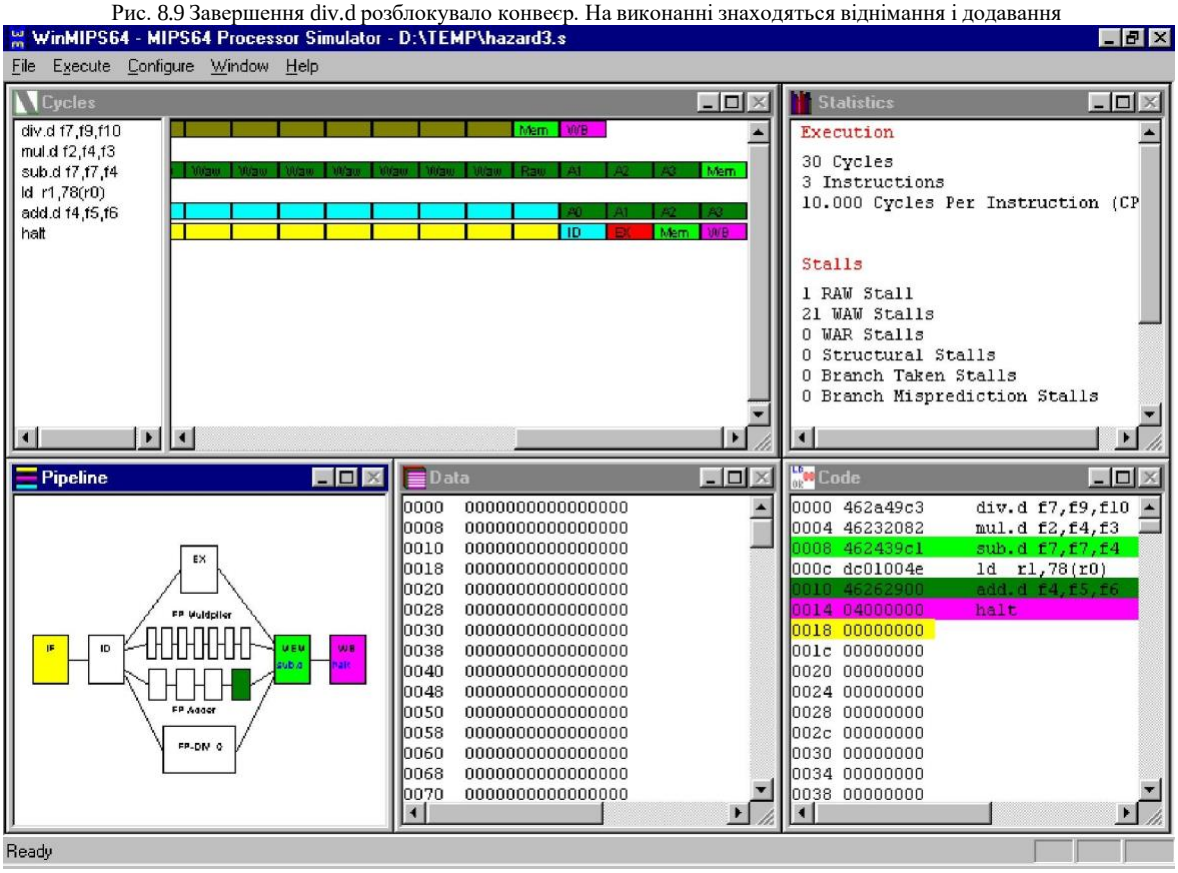

Рис. 8.10 Хаотично завершуються спочатку halt, потім рухомі віднімання та додавання

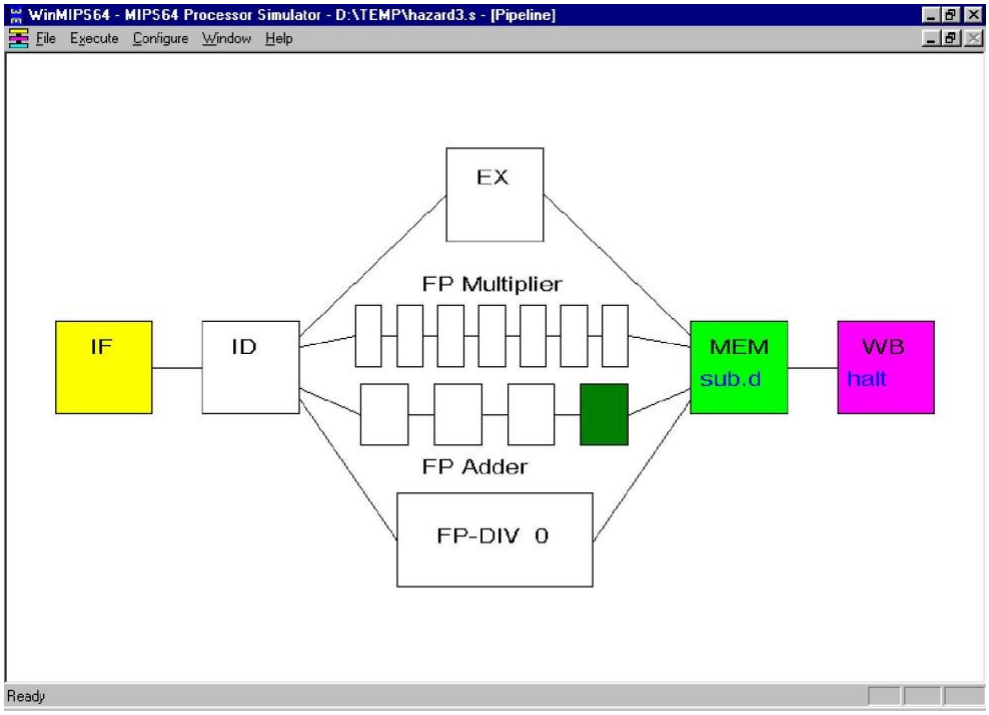

Рис. 8.11 Хаотичне завершення виконання інструкцій потоку (передтеча методу динамічного виконання)

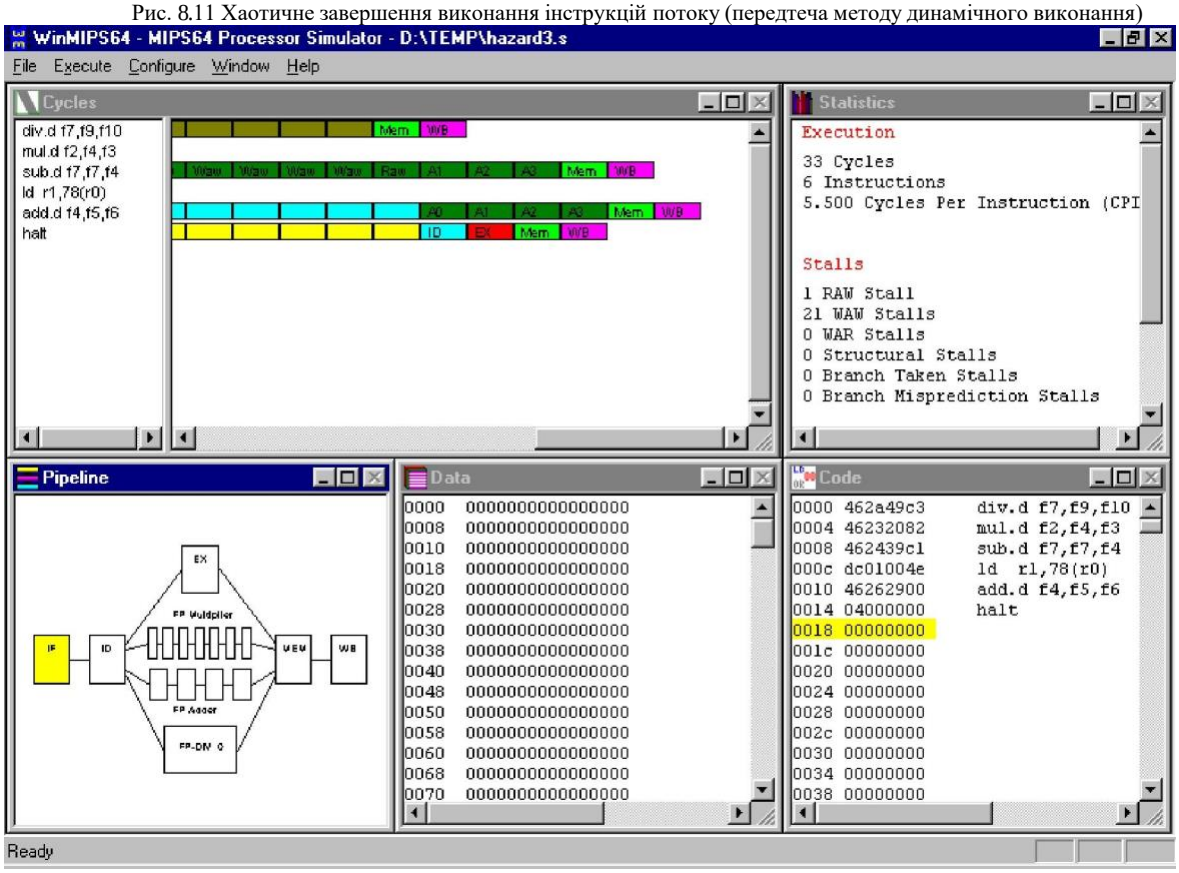

Рис. 8.12. Програму виконано, отримано програмні статистики

Варіанти завдань

Персональні варіанти завдань знайдете у табличці з варіантами.