Лаб. Упражнение No: 2

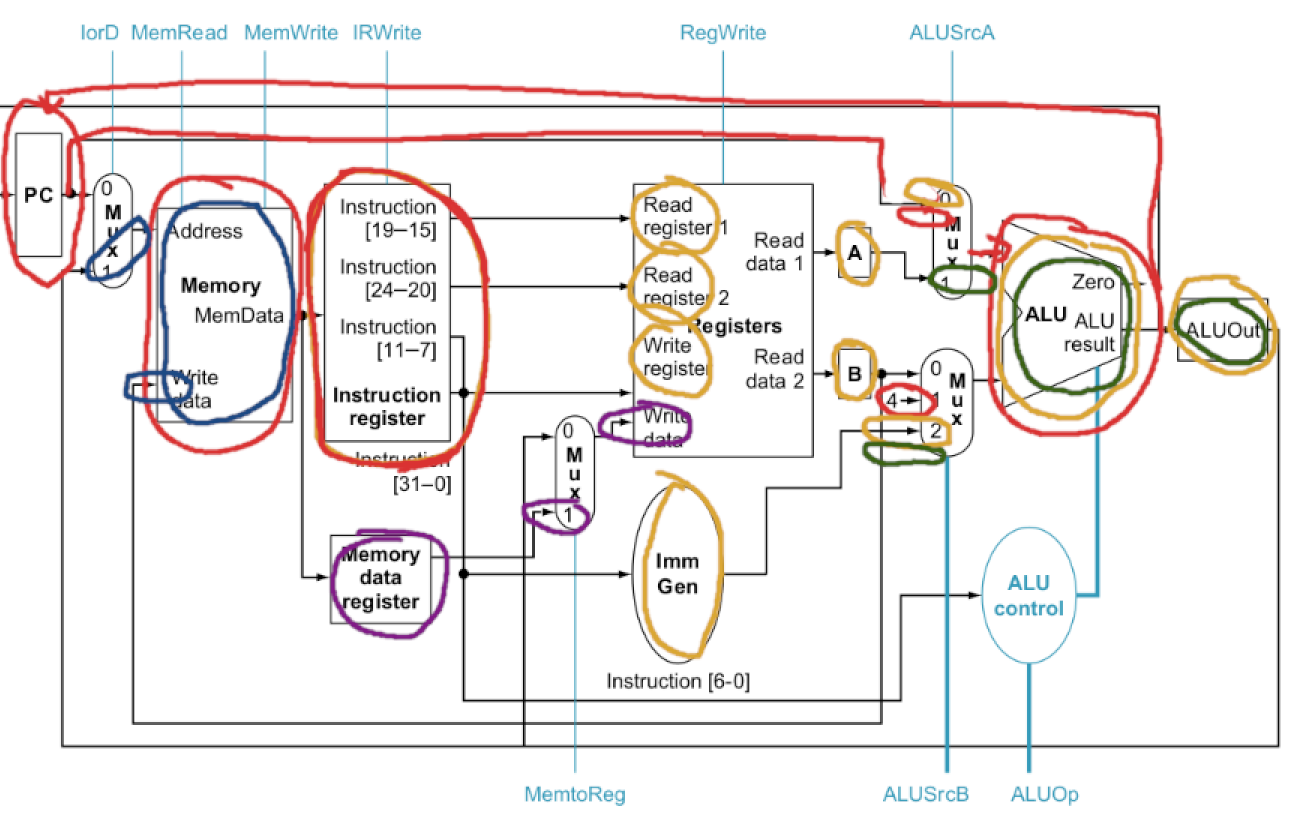
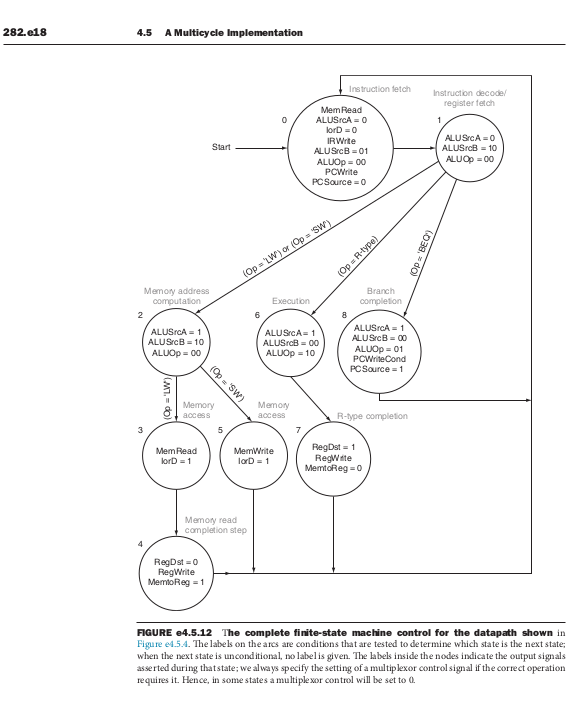
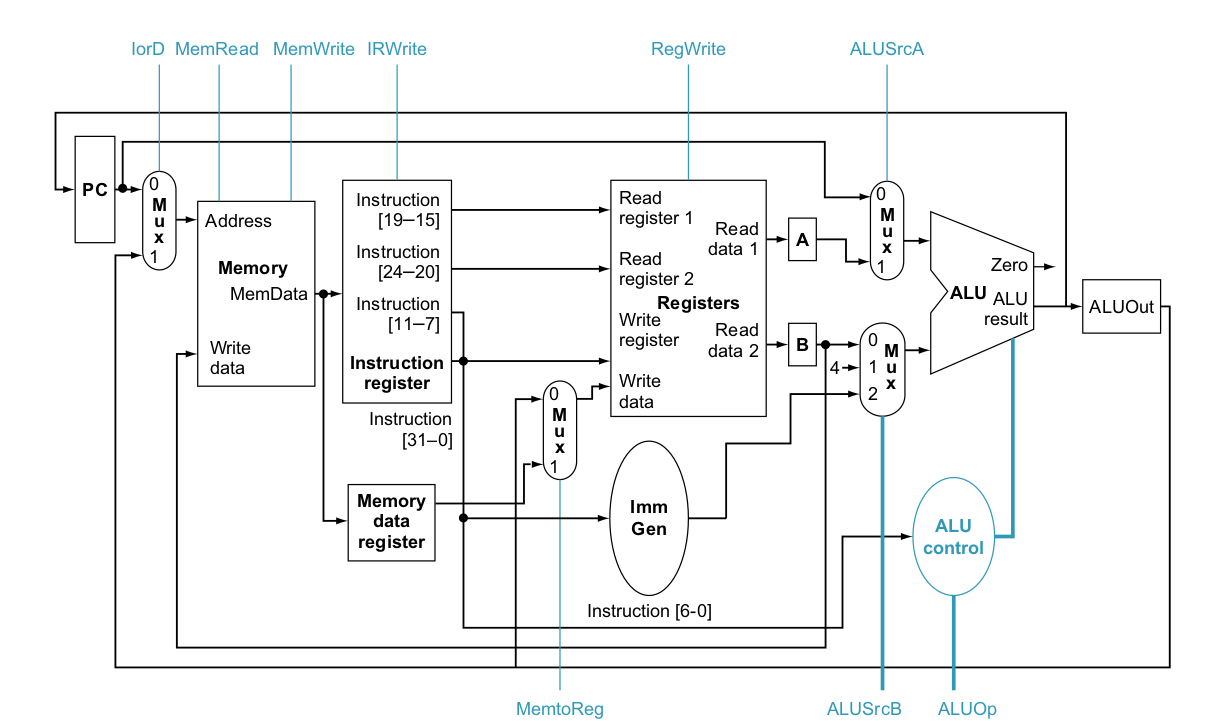
Дисциплина: Компютърни Архитектури

Асистент: Иван Янчев

Тема: Многотактова и конвейерна имплементация

========================================================================

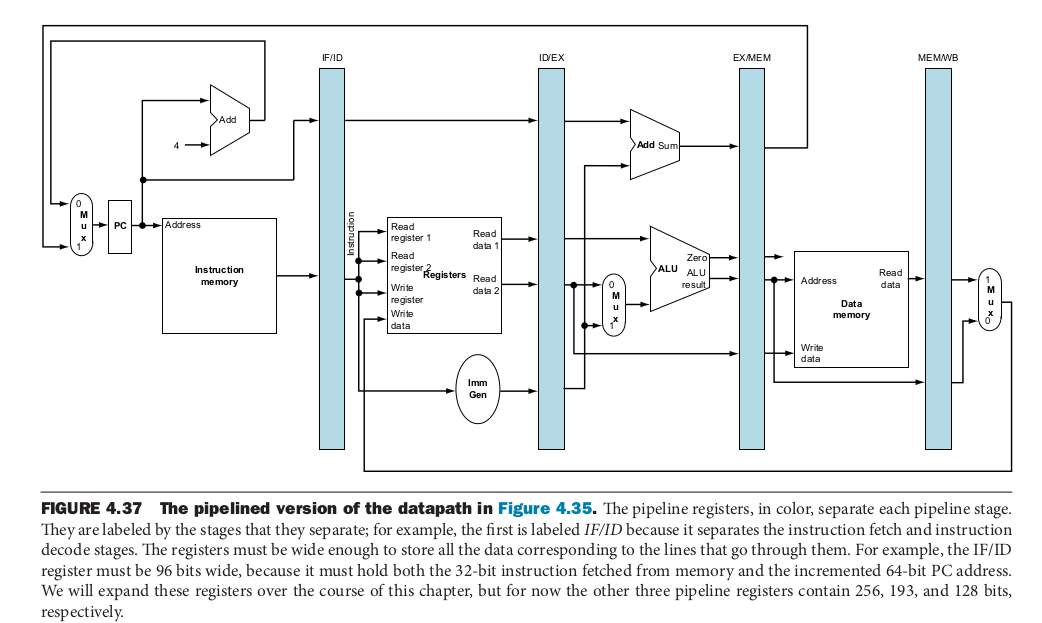
1. **Теория:**До сегабеше разглеждана само **еднотактовата имплементация**. Тя е неефективна и не се използва в съвременните процесори. При нея **на всеки такт от часовника се изпълнява една инструкция**, което води до **време за изпълнение на иструкциите равно на най-бавната** (от до сега разгледаните това е load защото използва и четирите основни елемента, а регистровият файл – два пъти, при четене и при запис). Това за прости процесори е приемливо, но когато се добавят и специализираните инструкции( напр. Плаваща запетая), времето за изпълнение се влошава значително. Решение на този проблем са многотактовата(Multicycle) и конвейерната(Pipeline) имплементации.  
     
     
   Многотактова имплементация datapath: (фиг 1).  
     
     
   фиг 1.  
   Тя ще бъде семпло разгледана, защото е по-непроизводителна от конвейерната.  
   При нея една иструкция се изпълнява за няколко цикъла, които са значително по-къси като времетраене от еднотактовата. Тук понеже на различните цикли се извършва различна операция, хардуерът може да бъде намален. От схемата може да се забележи, че двата суматора(add) са съкратени, както и инструкционната и данновата памети са обединение в една(Memory). Това е защото четенето от инструкционната памет и четенето от данновата памет могат да се случват в различни цикли и следователно не са ни нужни два елемента. Аналогично за суматорите, ALU-то може да извършва събиранията в няколко цикъла, различни от този в който си извършва неговата зададена операция.  
     
   На схемата са добавени и няколко допълнителни междинни регистри (MD register, A и B, ALUOut). Те запазват изхода на определени елементи. Понеже множество елементи отговарят за множество различни неща, техните входни данни и операции ще се променят в процес на изпълняване на инструкцията. Ако старите данни не се запазят някъде (в случая регистър, но може и памет, регистров файл, програмен брояч), те ще се изгубят.  
     
   Управляващите сигнали също се променят в процес на изпълнение на инструкцията. При еднотактовата имплементация разглеждахме управляващото устройство(Control) като таблици на истиност – подава се някакъв вход(битове 0-6), който отговаря на някакви управляващи сигнали. При многотактовата имплементация управляващото устройство е по-скоро машина с крайни състояния, като управляващите сигнали преминават от едно състояние в друго и се сменят на всеки такт. (фиг 2)  
     
   Инструкциите могат да бъдат разделени на следните етапи, всеки етап отнема по един цикъл на часовника.  
     
   1. Извличане на инструкцията(Instruction Fetch [IF])  
   - Това е извличането на текущата инструкция от паметта и изчисляването на следващия адрес PC+4. (тези операции, както и много други в следващи фази, могат да бъдат изпълнени едновременно, понеже са на различни пътища и нямат конфликт помежду си(използват различни у-ва)).  
     
   2. Декодиране и извличане на данни от регистровия файл(Instruction Decode [ID])  
   - Read data 1 и Read data 2 се запазват в съответните регистри (A и B), Изчислява се адрес за условен преход (PC + immediate).  
     
   3. Изпълнение, изчисление на target адрес за паметта или условие за преход(Instruction Execute[EX]).   
   - В тази фаза основен елемент е ALU-то и изпълнението на неговите операции.  
     
   4. Достъп до паметта, или запазване на инструкциите в регистър при R-format (Memory Access [MEM]). Тази фаза е за достъп до паметта, ако тя не е нужна се пропуска.  
     
   5. Запис в регистрите (Writeback [WB]), Тази фаза е основно за load инструкцията и се изразява в запазване на прочетената памет в регистрите. (фиг 3).  
      
   За повече информация: стр: (282.e1 до 283) от учебника.  
     
     
     
     
     
   фиг 2.



Фиг 3.

Дизайнерите се стремят времената за изпълнение на всяка от фазите да са максимално идентични, но все пак тактовете на часовника могат да бъдат с различно времетраене.

**Конвейерно изпълнение:**Конвейерното изпълнение е подобно на многотактовото от към това, че изпълнението на инструкцията е на няколко такта и е подобно на еднотактовото от към това, че елементи с една ѝ съща функция трябва да бъдат дублирани.Datapath-а също е подобен на еднотактовото (фиг 4).   
  
Между различните фази са поместени регистри пазещи състоянието на инструкцията за момента (данните необходими за да се изпълнят следващите етапи). Тези регистри са IF/ID, ID/EX, EX/MEM, MEM/WB. Имената им идват от това между кои фази се намират.



За повече информация стр. 283+ в учебника. Пропуснете всички hazards за сега.

**2. Допълнителна теория:**



1. **За протокола**стр 374 - 4.16, 4.15, 4.7, 4.8, 4.9, 4.10  
   Бонус към предишното упражнение: 4.11, 4.12, 4.13
2. **Линкове**

**Упражнения**: [https://github.com/tu-iyan/Computer-Architecture-2024](упражнения)

**Учебник:** [http://library.lol/main/373C67B0C5E22C9B92B1D8FACDC47E68](учебник)