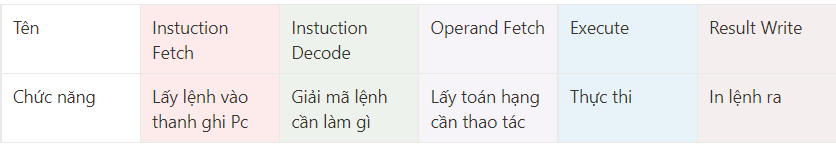
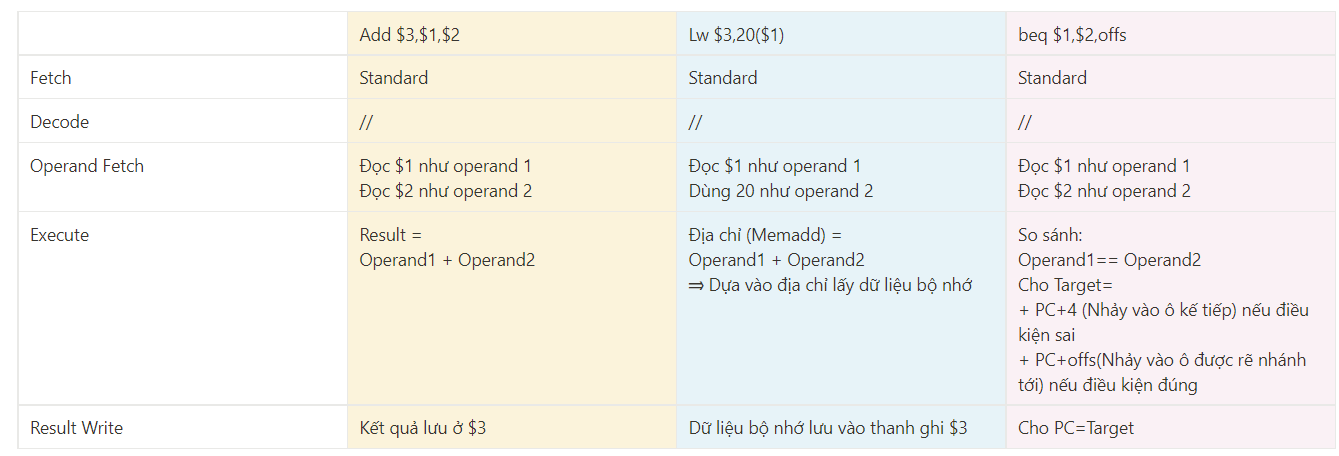
# **DataPath**

## *I.Chu Kì thực thi lệnh*



Ví dụ:



*II. 5 bước thực thi lệnh trong MIPS*

\* quy luật:

+ Gom Decode với operand Fetch lại với nhau

+ Chia Execute thành Alu ( bước tính toán) và Memory Access ( Truy xuất bộ nhớ)

## *III.Tìm hiểu kĩ các bước trong 5 chu kì thực thi lệnh trong MIPS:*

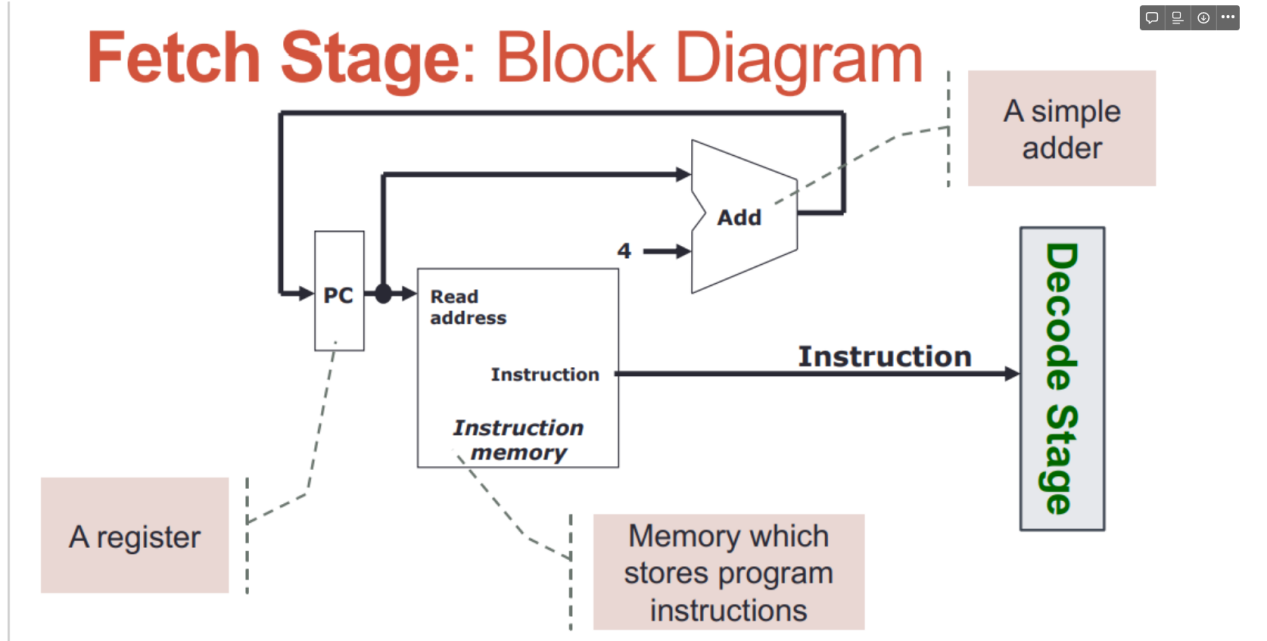
### *1.Fetch stage*

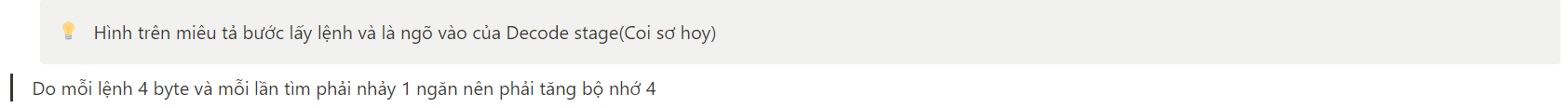
+ Dùng thanh ghi Program Counter(PC) để lấy lệnh bộ nhớ

+ Tăng bộ nhớ lên 4 để lấy lệnh tiếp theo

+ Output đưa tới bước decode

\* Sơ đồ của Fetch Stage:





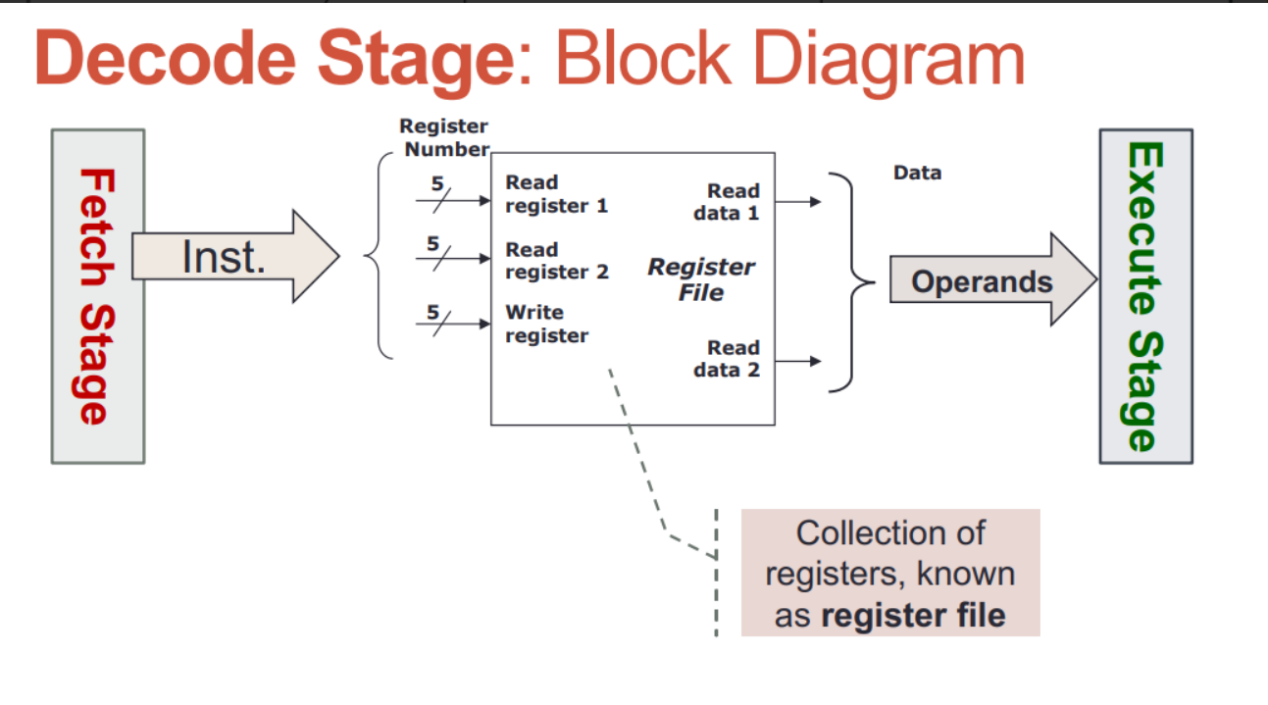
### *2.Decode stage*

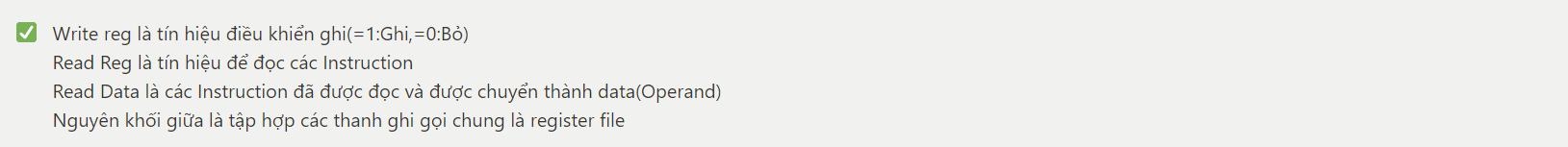
+ Đọc opcode xác định lệnh

+ Đọc các giá trị thanh ghi để tìm Operand

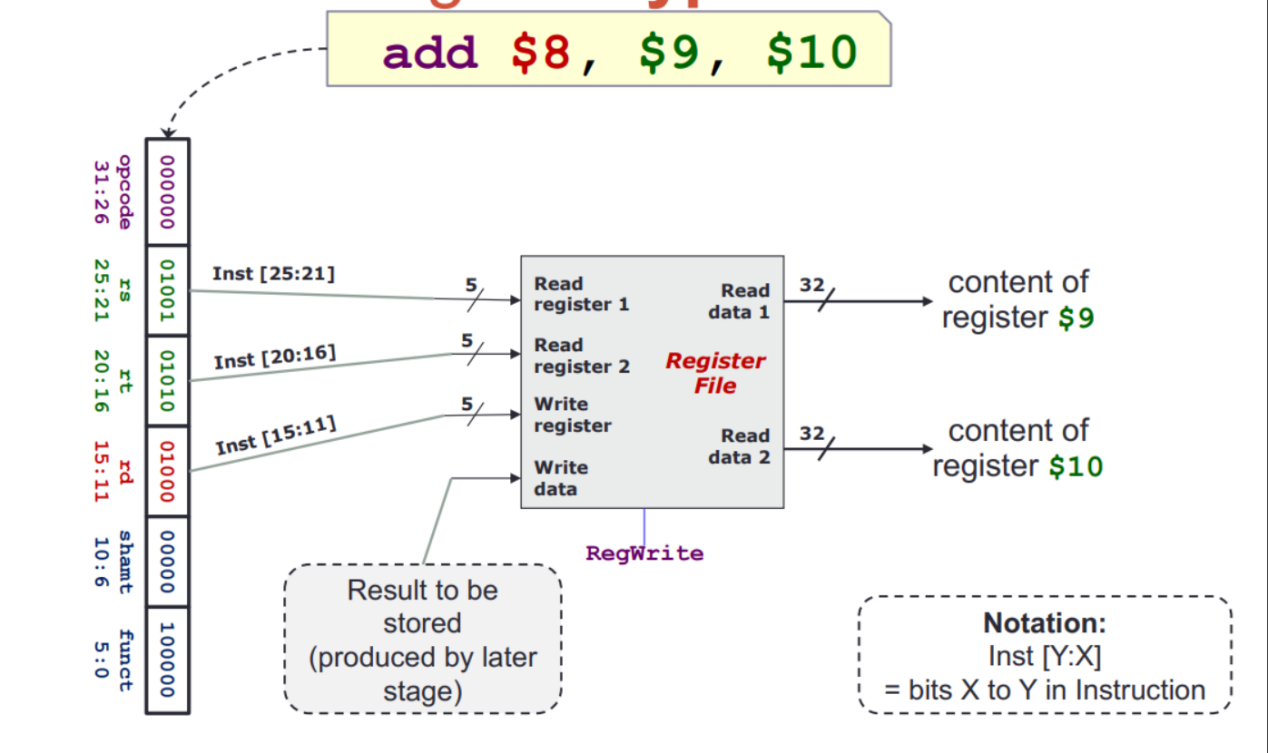
+ Đầu ra tới ALU Stage

\*Sơ đồ sơ khai:

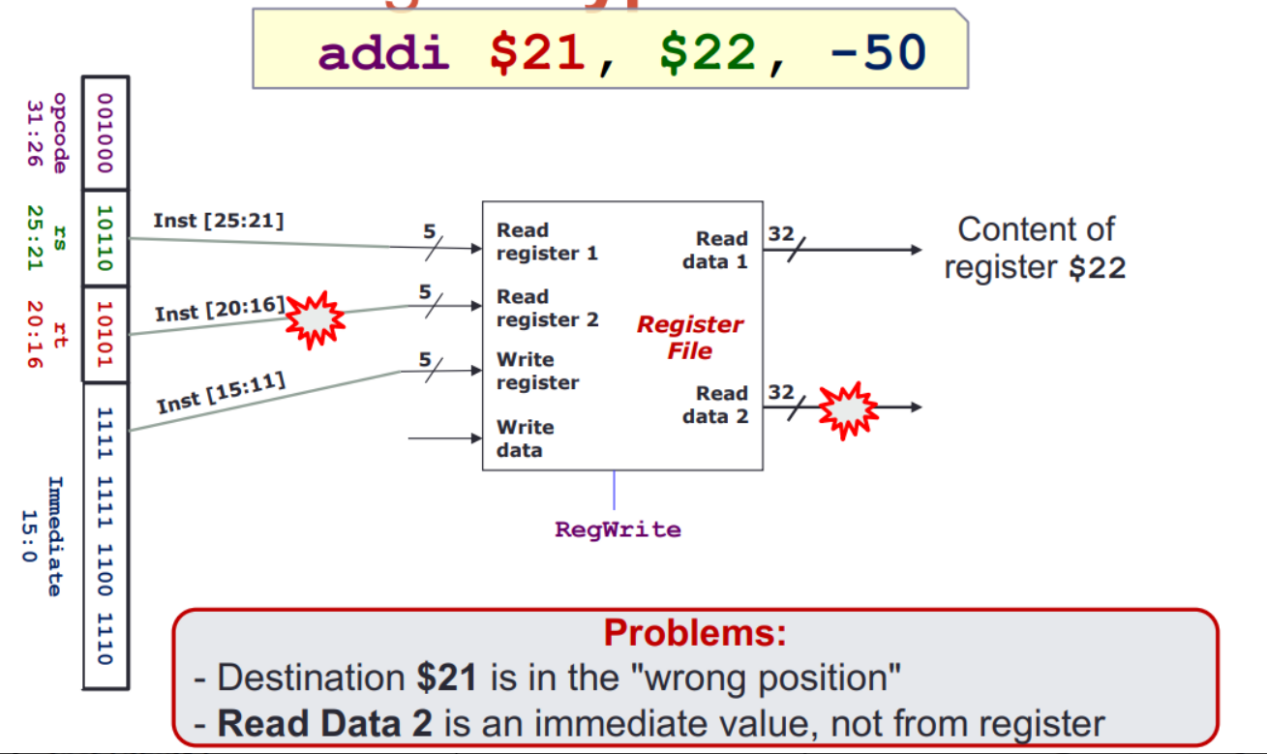




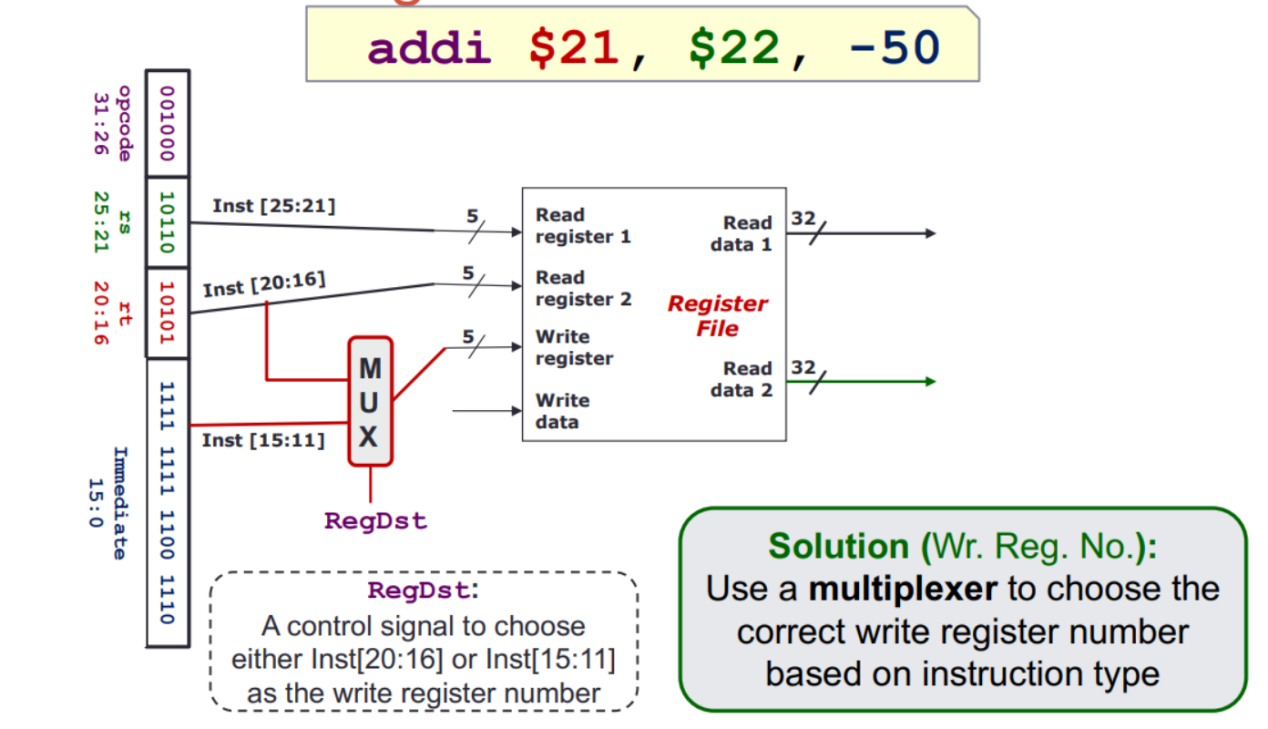
\* Sơ đồ hoàn chỉnh cho lệnh add:



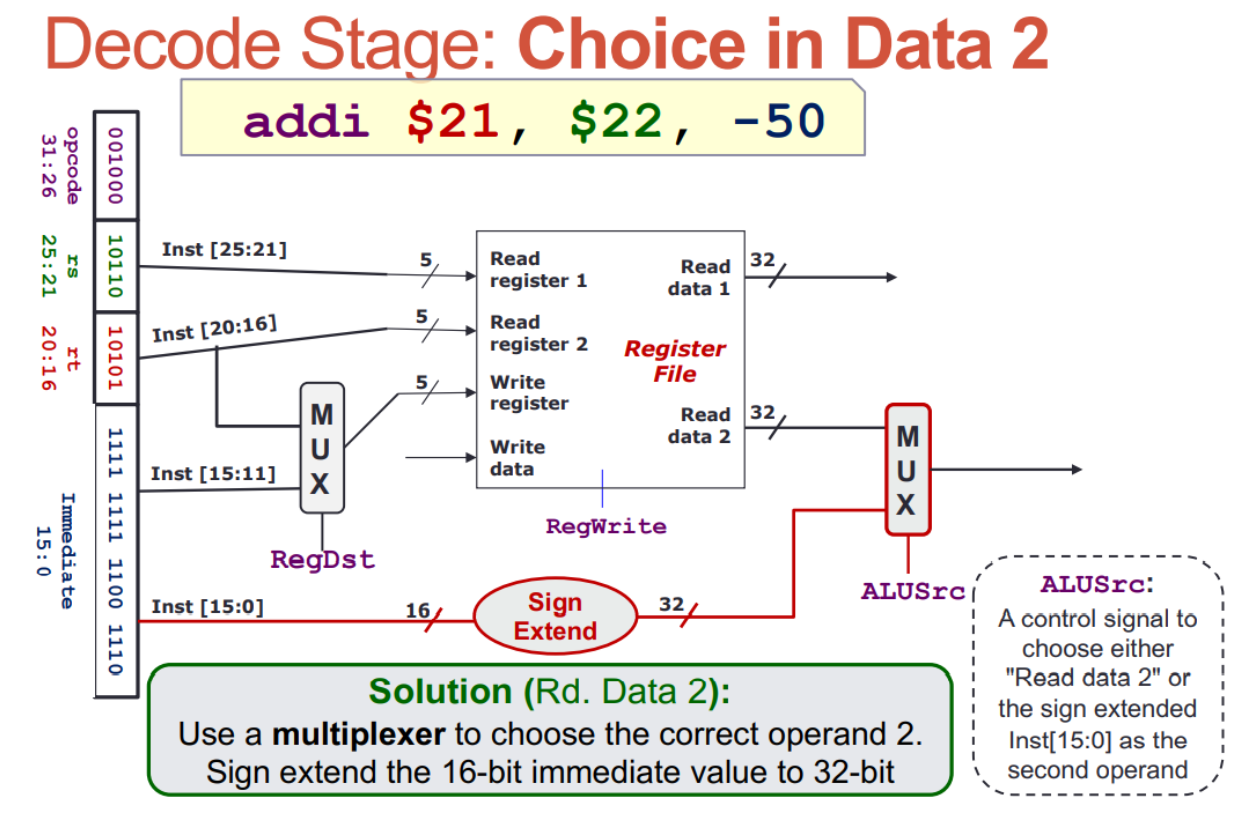
\* Tuy vậy sơ đồ hoàn chỉnh trên không thể dùng addi



\* Sơ đồ sơ khai cho tất cả các tập lệnh add(Dùng thêm một bộ mux):



\* Sơ đồ hoàn chỉnh ( Bao quát) cho Decode stage.



1. ALU stage

Tính toán số học và logic gồm:

+ Thao tác số học(add...)

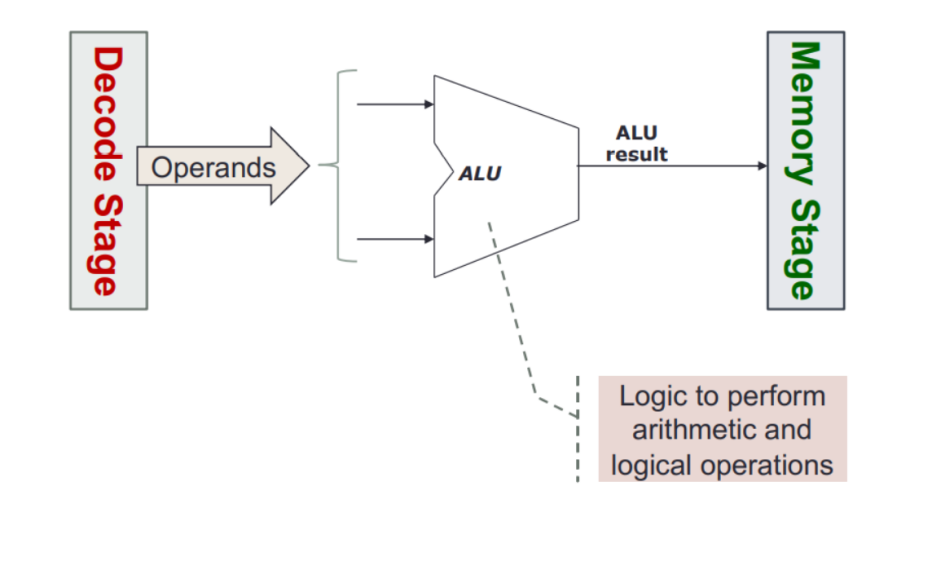
+ Thao tác logic(and,or...)

+ Dịch qua trái phải(sll...)

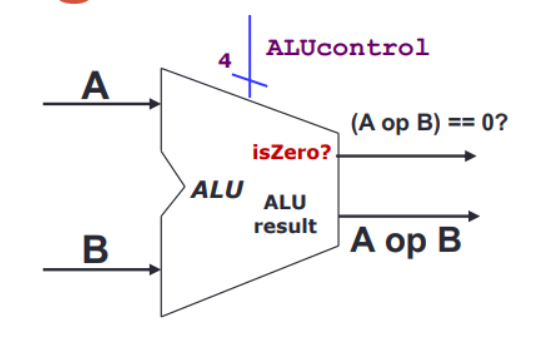
+ Thao tác địa chỉ(LW,SW)

+ Thao tác rẽ nhánh(bne,beq)

\* Sơ đồ tổng quát của ALU:



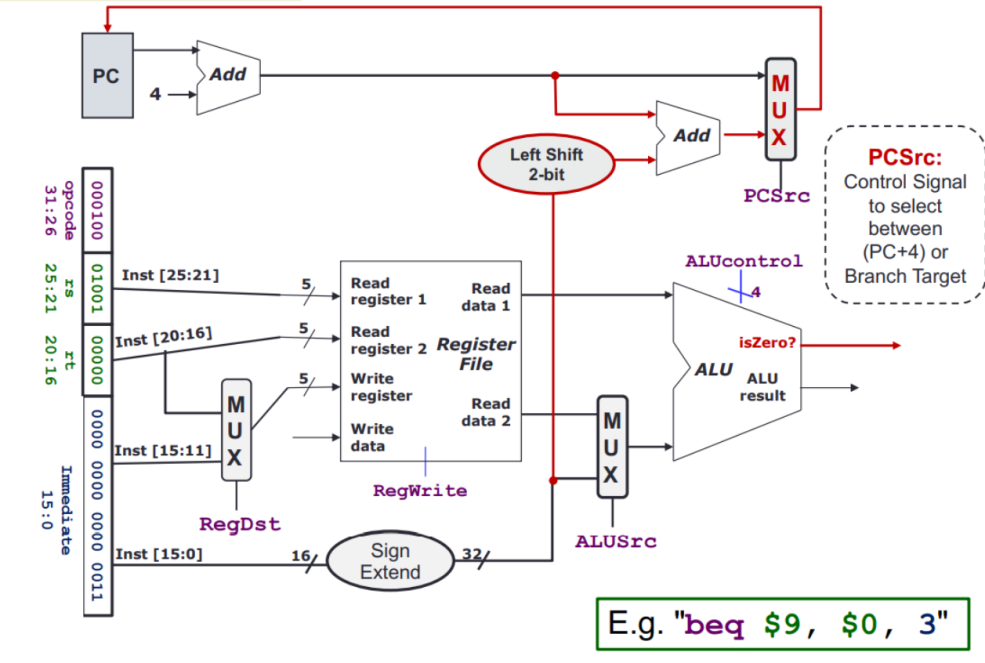
\* Sơ đồ tổng quát đầy đủ của ALU:



Nếu như A=B thì sẽ đi theo đường ở trên còn nếu A != B thì sẽ đi đường dưới

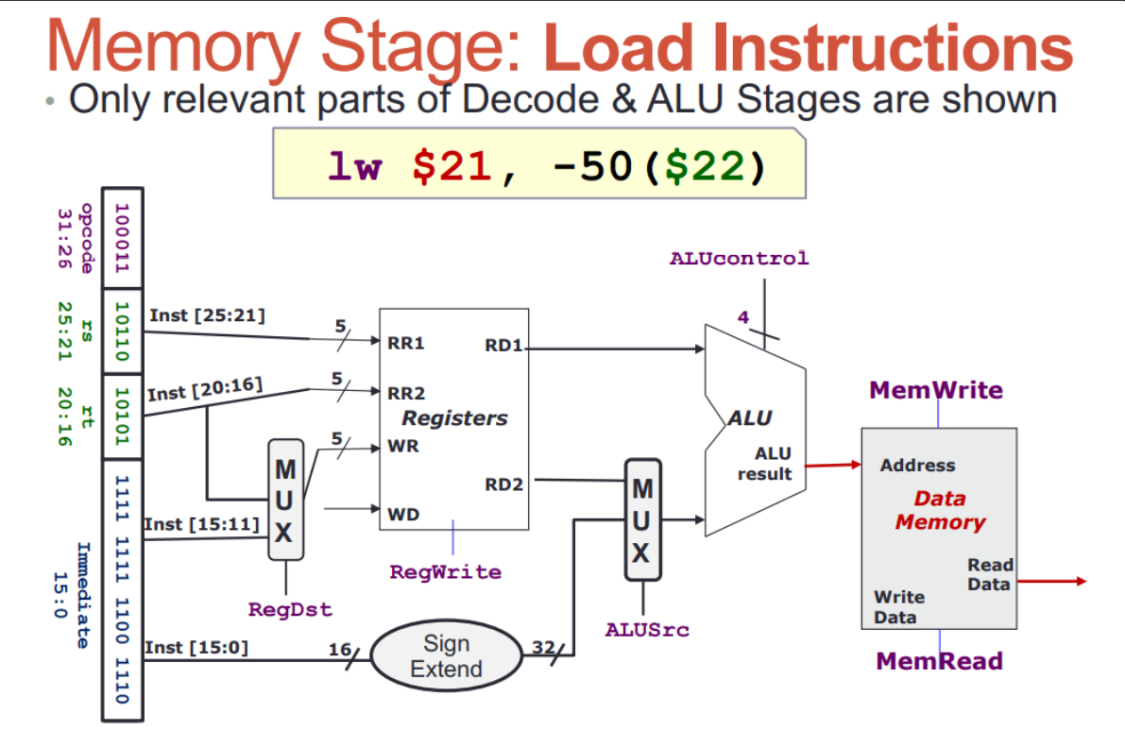
=> IsZero chỉ xuất hiện trong lệnh rẽ nhánh

\* Sơ đồ hoàn chỉnh tới bước ALU (Đối với lệnh rẽ nhánh):

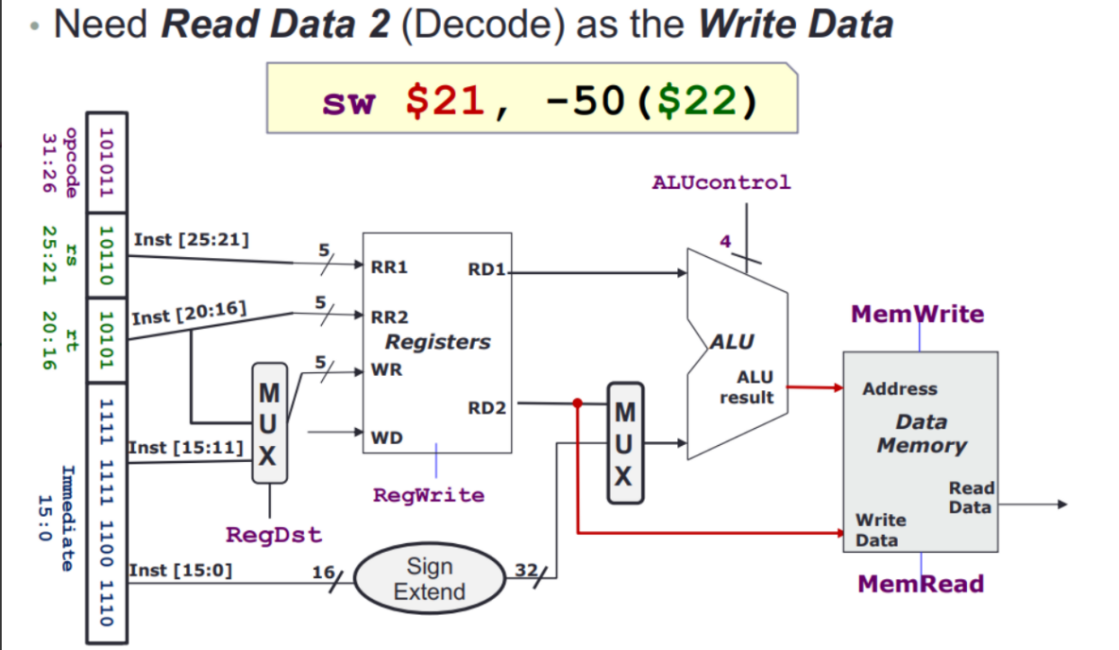


1. *Memory Access*

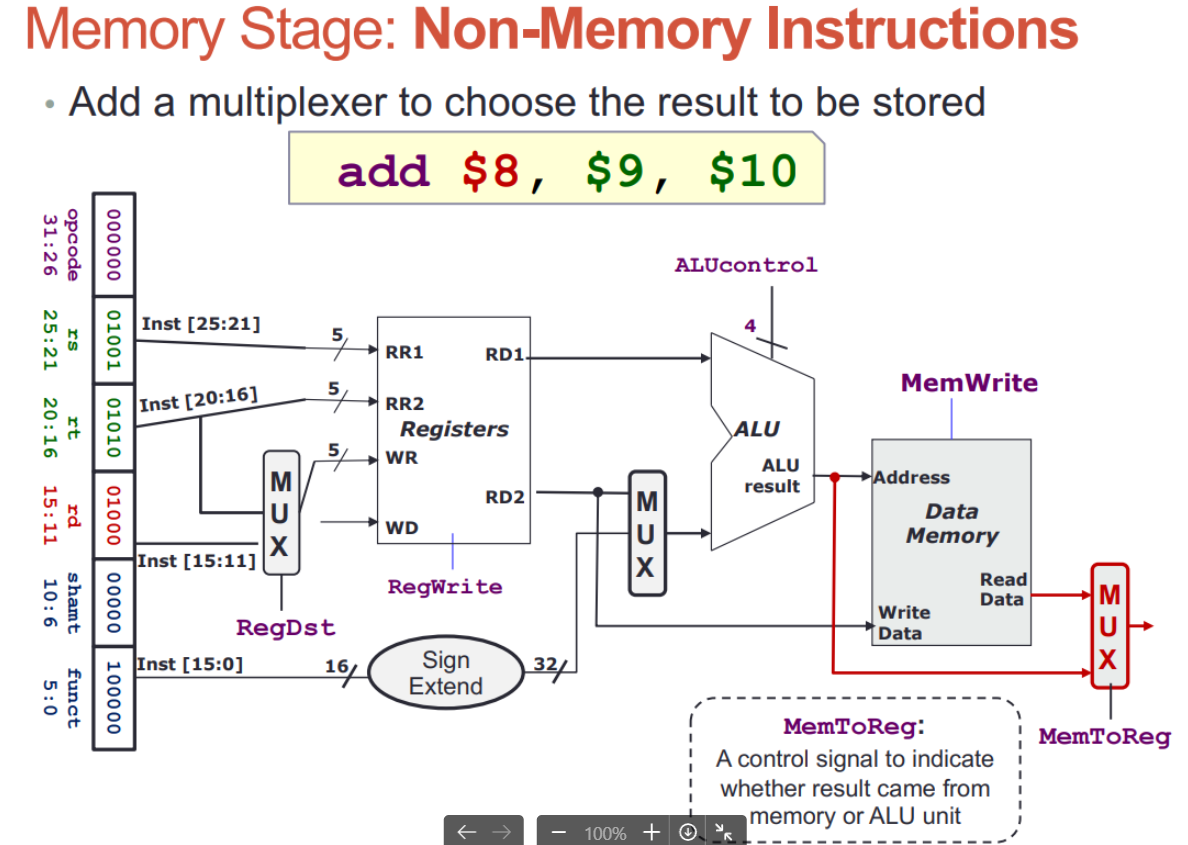
\* Sơ đồ cho lw:



\* Sơ đồ cho sw:

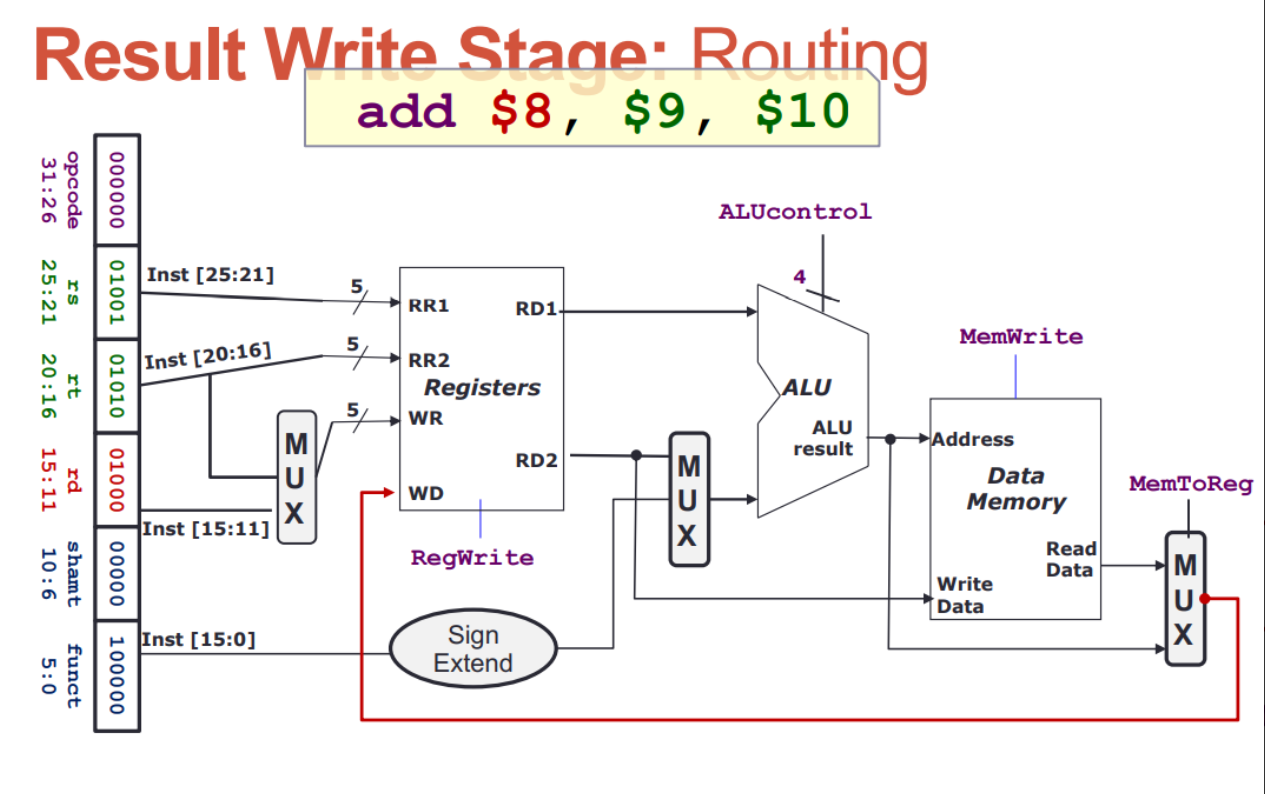


\* Sơ đồ cho lệnh add:



1. *Result Write*

\* Hình tổng thể Result Write cho lệnh add:



=>> Tổng hợp Datapath 