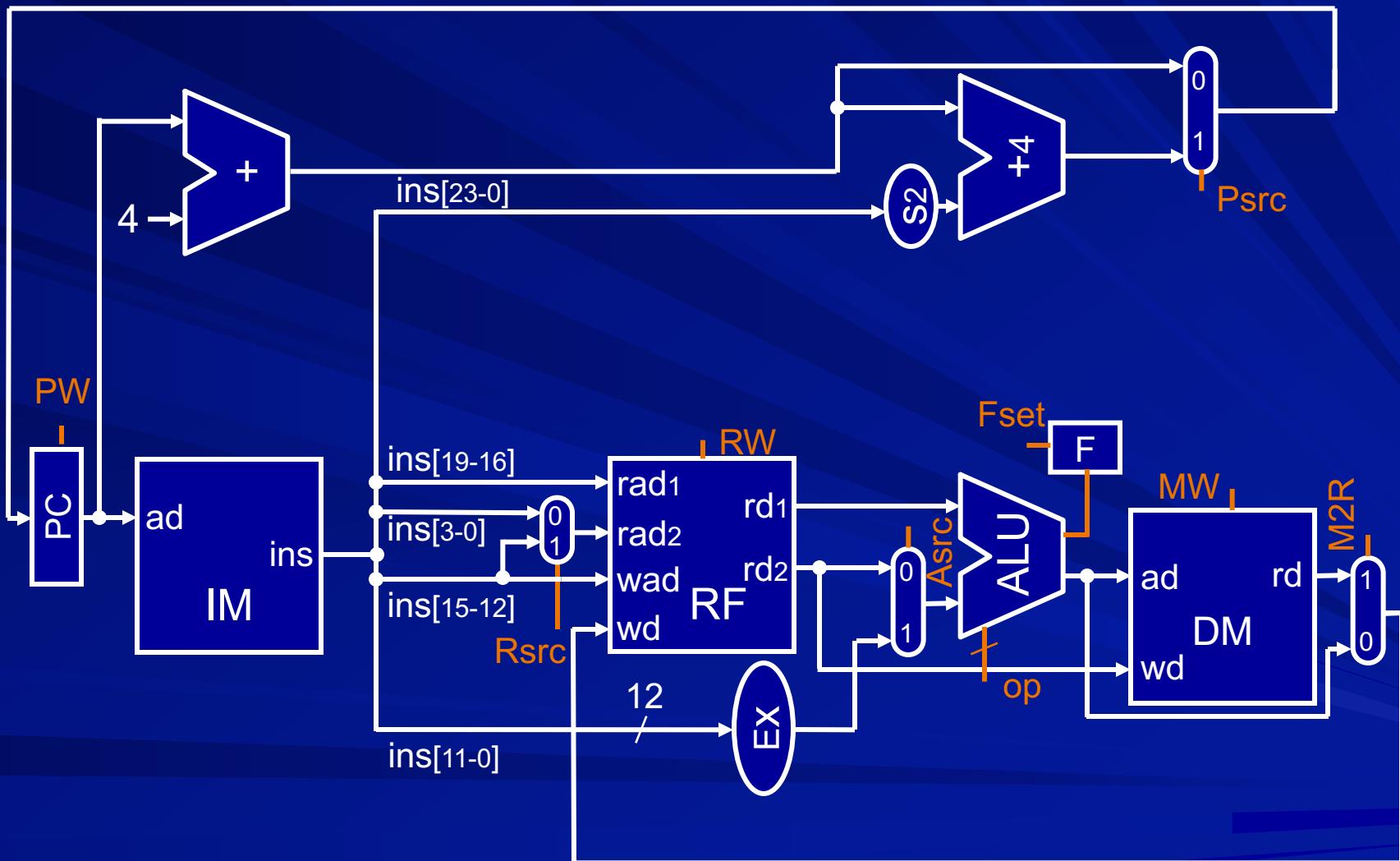


COL216

Computer Architecture

Design a processor -
Multi-cycle design approach
5th February, 2022

Single cycle Datapath



Problems with single cycle design

- Slowest instruction pulls down the clock frequency
- Resource utilization is poor
- There are some instructions which are impossible to be implemented in this manner

Analyzing performance

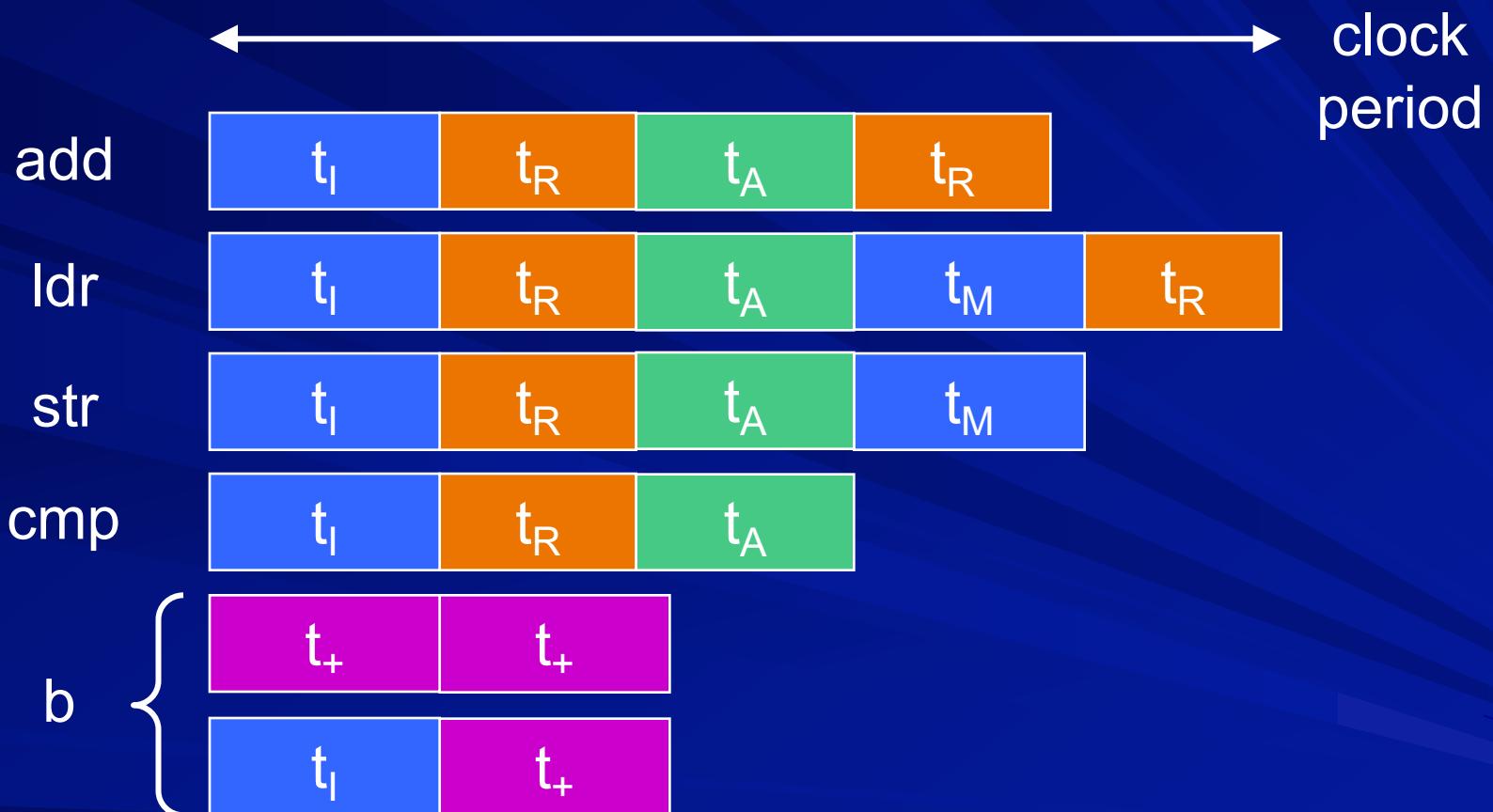
Component delays

■ Register	0
■ Adder	t_+
■ ALU	t_A
■ Multiplexer	0
■ Register file	t_R
■ Program memory	t_I
■ Data memory	t_M
■ Bit manipulation components	0

Overall clock period

$$\max \left\{ \begin{array}{ll} t_+, & t_I + t_R + t_A + t_R \\ t_+, & t_I + t_R + t_A \\ t_+, & t_I + t_R + t_A + t_M \\ t_+, & \overbrace{t_I + t_R + t_A + t_M + t_R} \\ t_I + t_+, & \overbrace{t_+ + t_+} ? \end{array} \right.$$

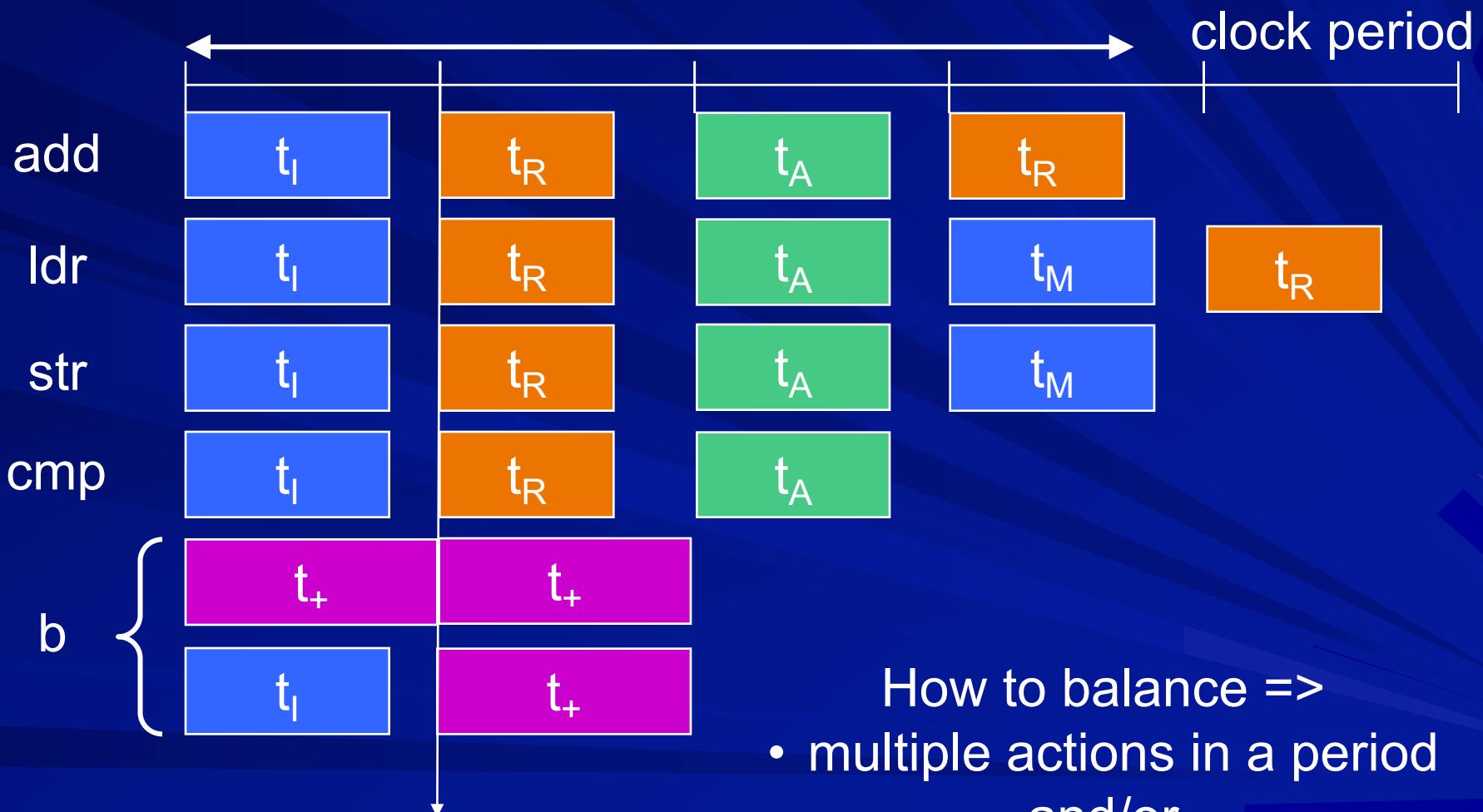
Clock period in single cycle design



Split the cycle into multiple cycles



Unbalanced delays



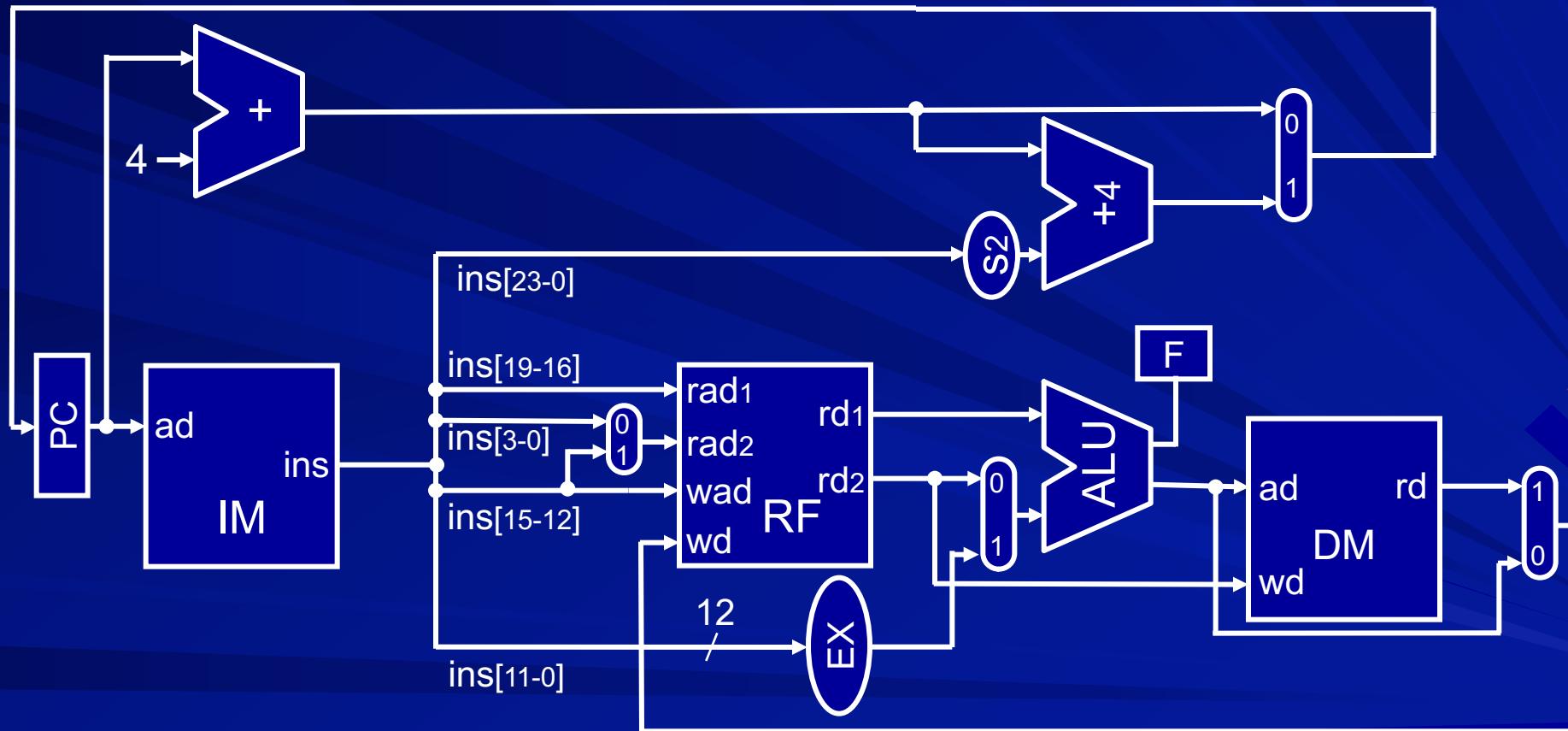
How to balance =>

- multiple actions in a period and/or
- multiple periods for an action

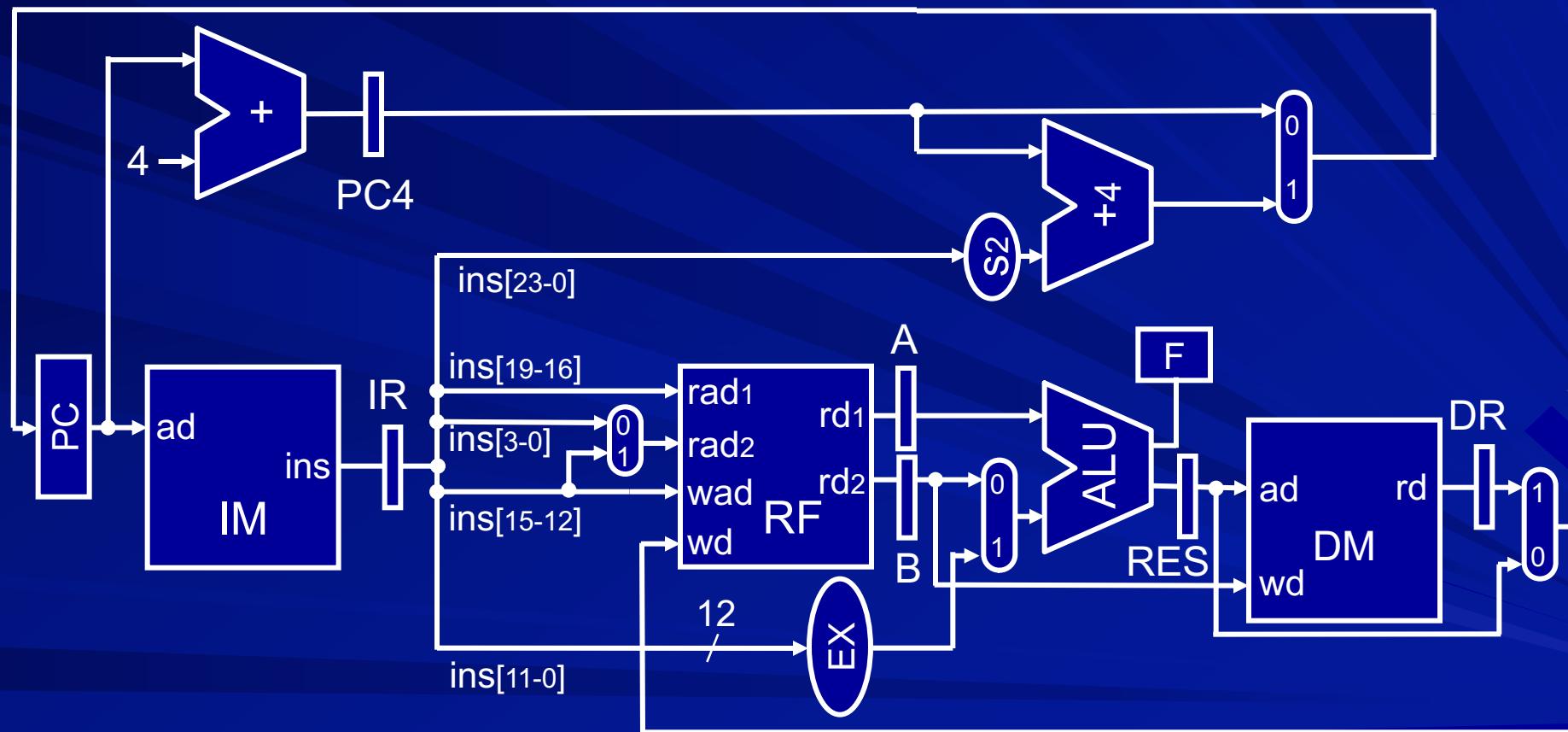
Improving resource utilization

- Can we eliminate two adders?
- How to share (or reuse) a resource (say ALU) in different clock cycles?
- Store results in registers.
- Of course, more multiplexing may be required!
- Resources in this design: RF, ALU, MEM.

Single Cycle Datapath

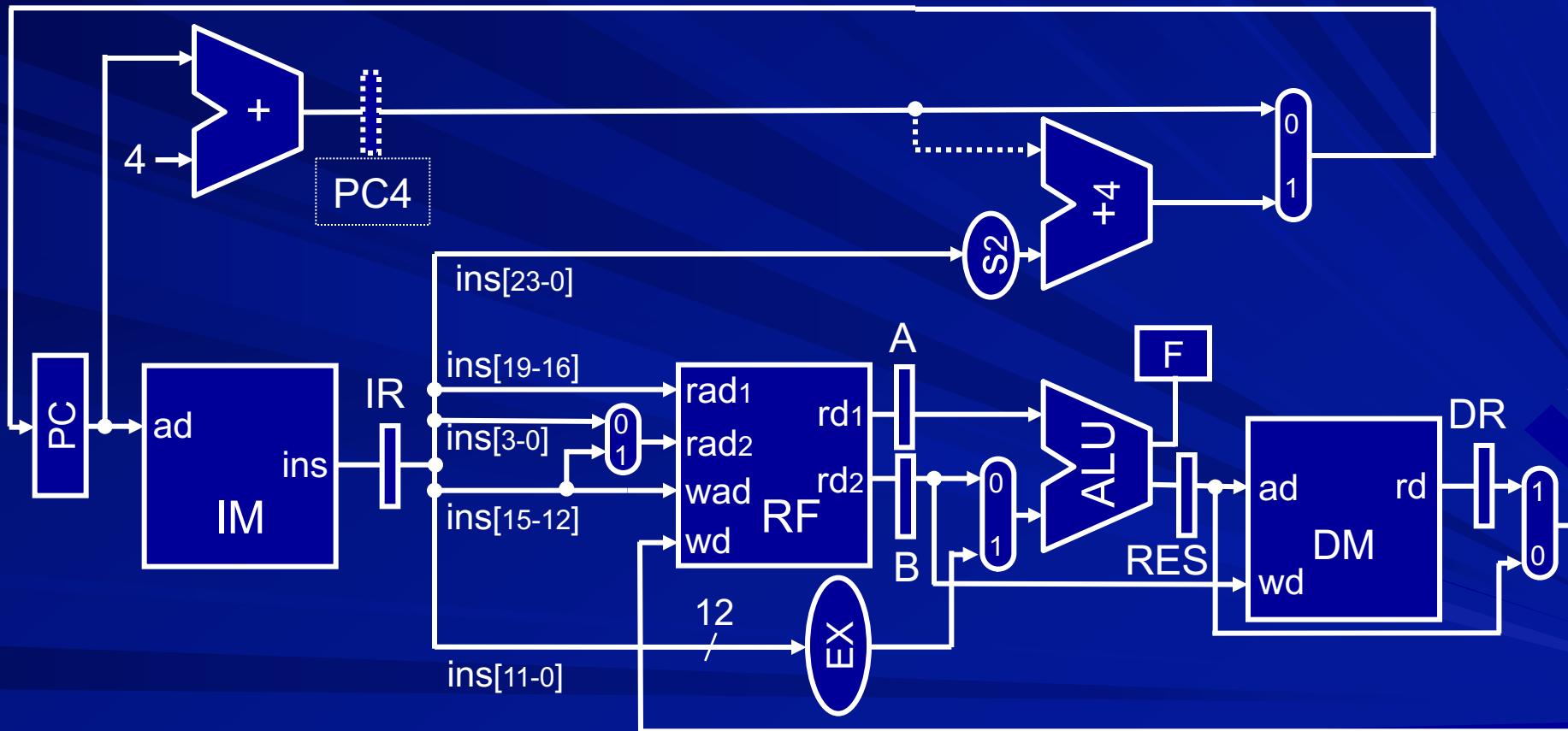


Introducing registers

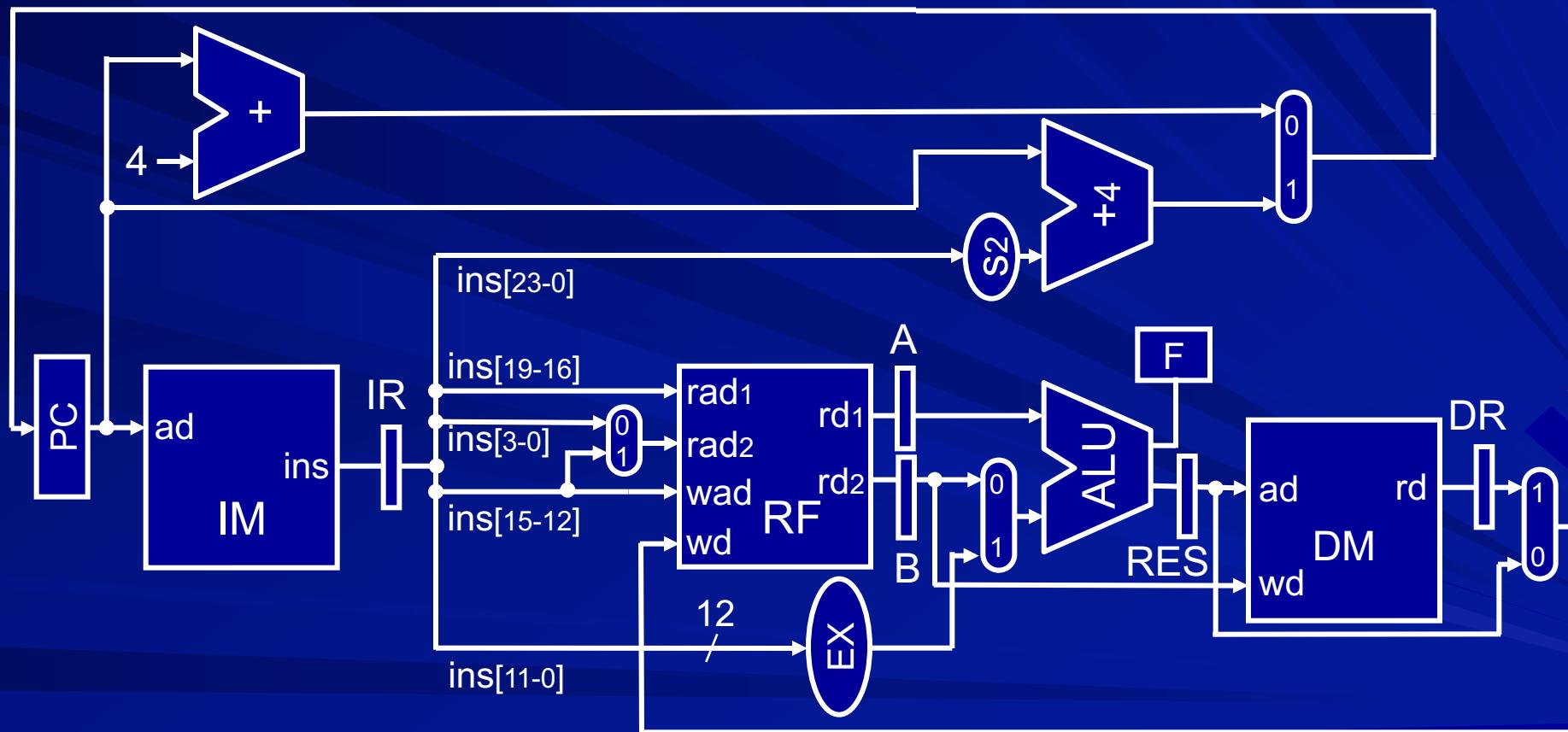


PC4 can be eliminated

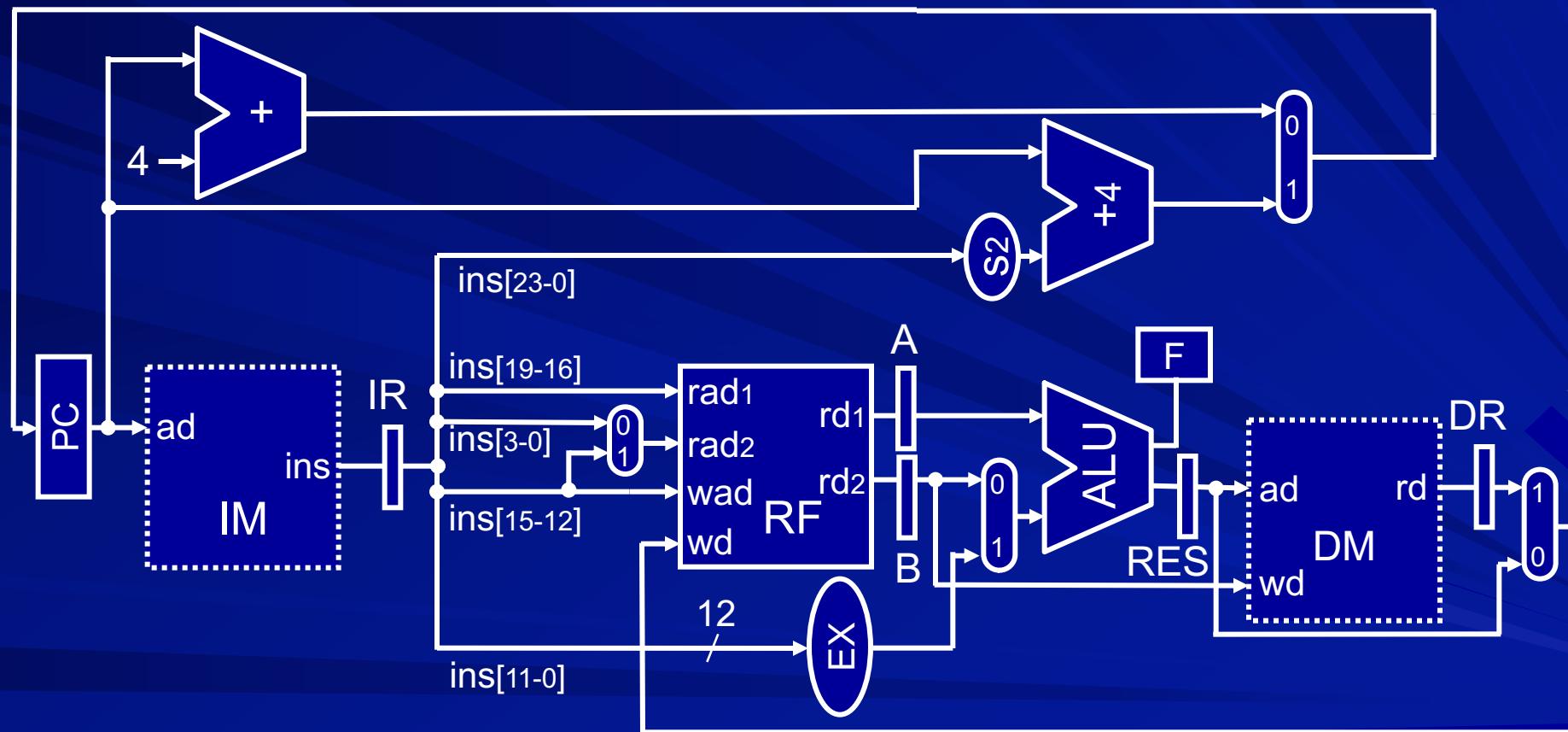
Store PC + 4 in PC itself



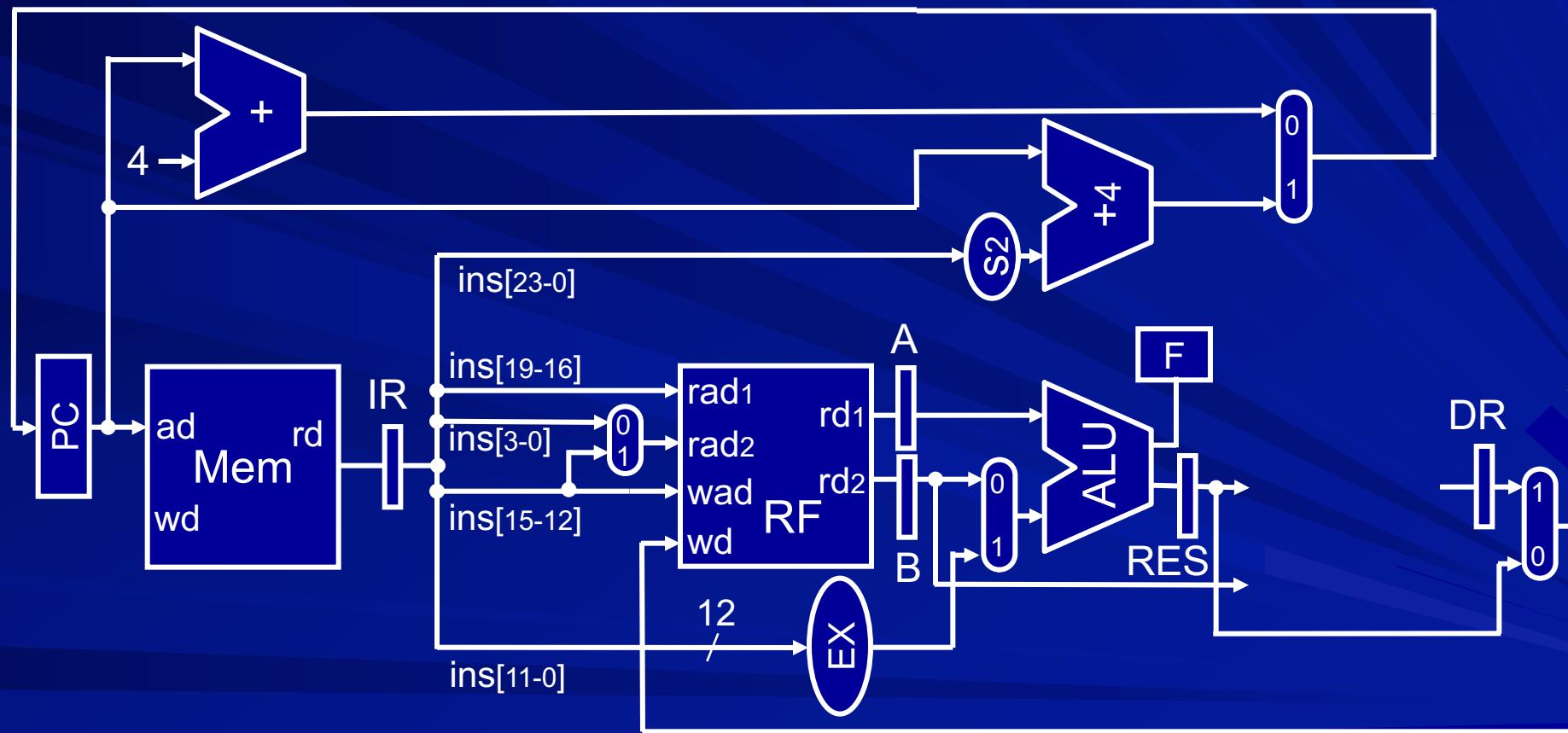
PC4 can be eliminated



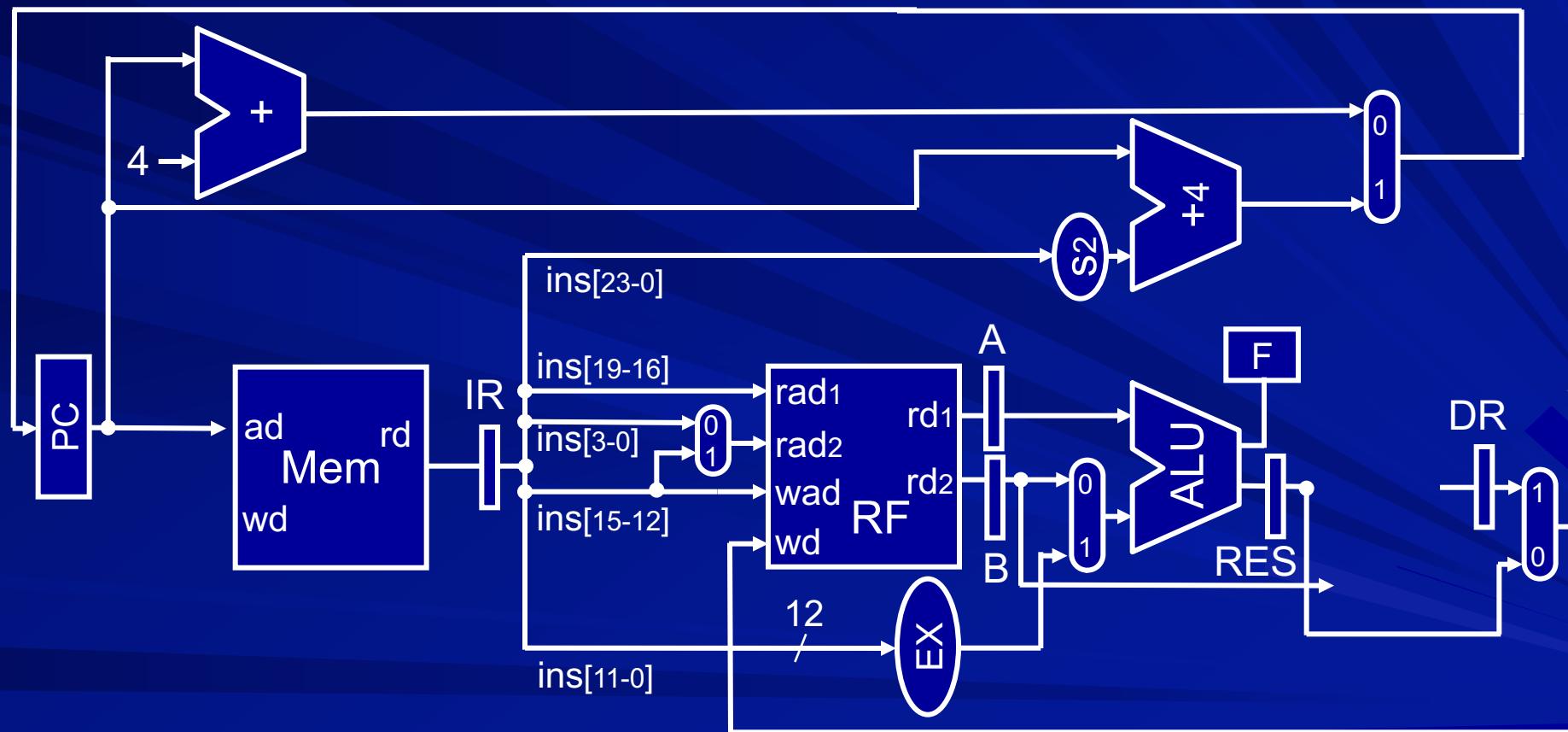
Merge IM and DM



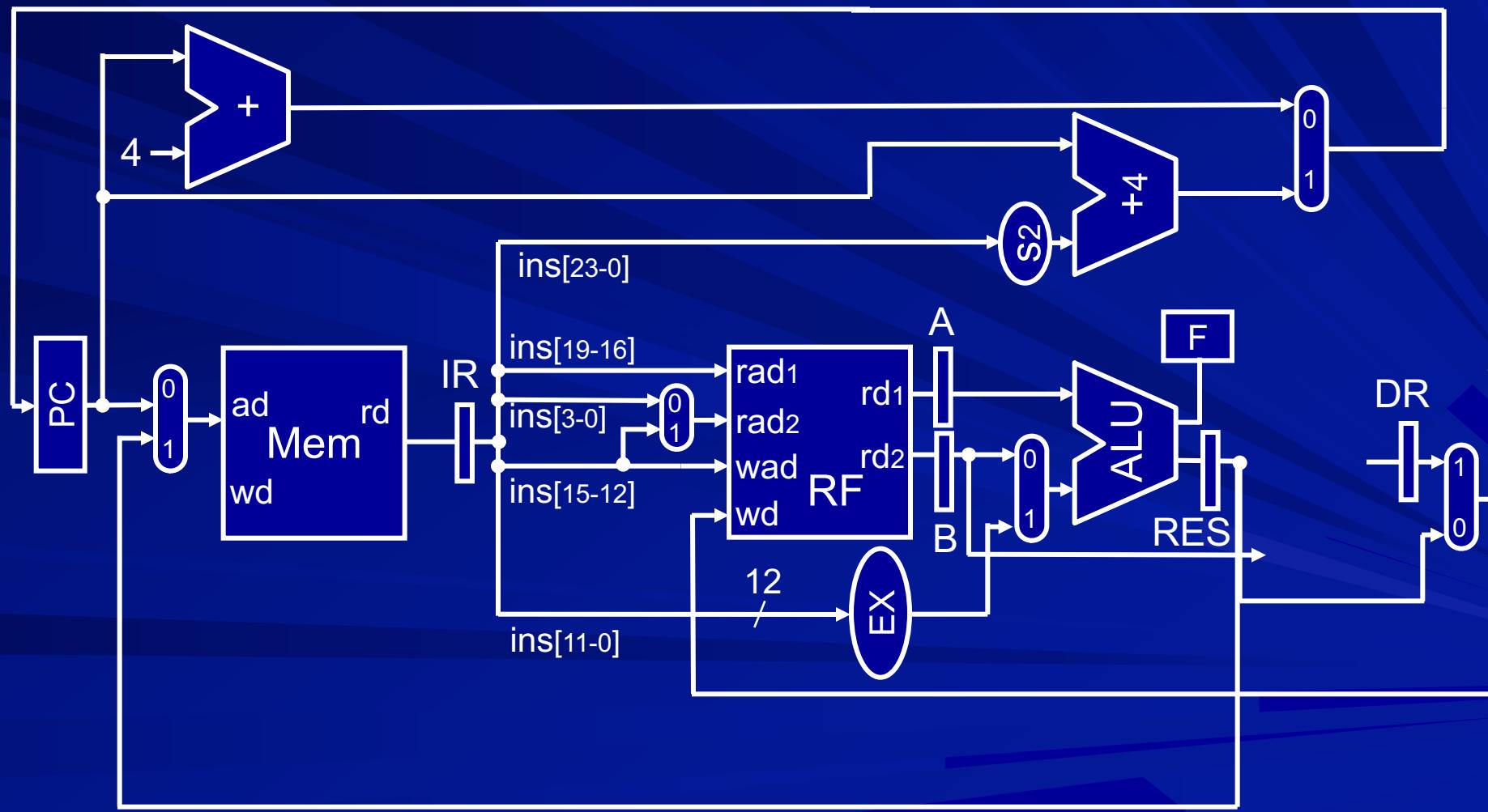
Merge IM and DM



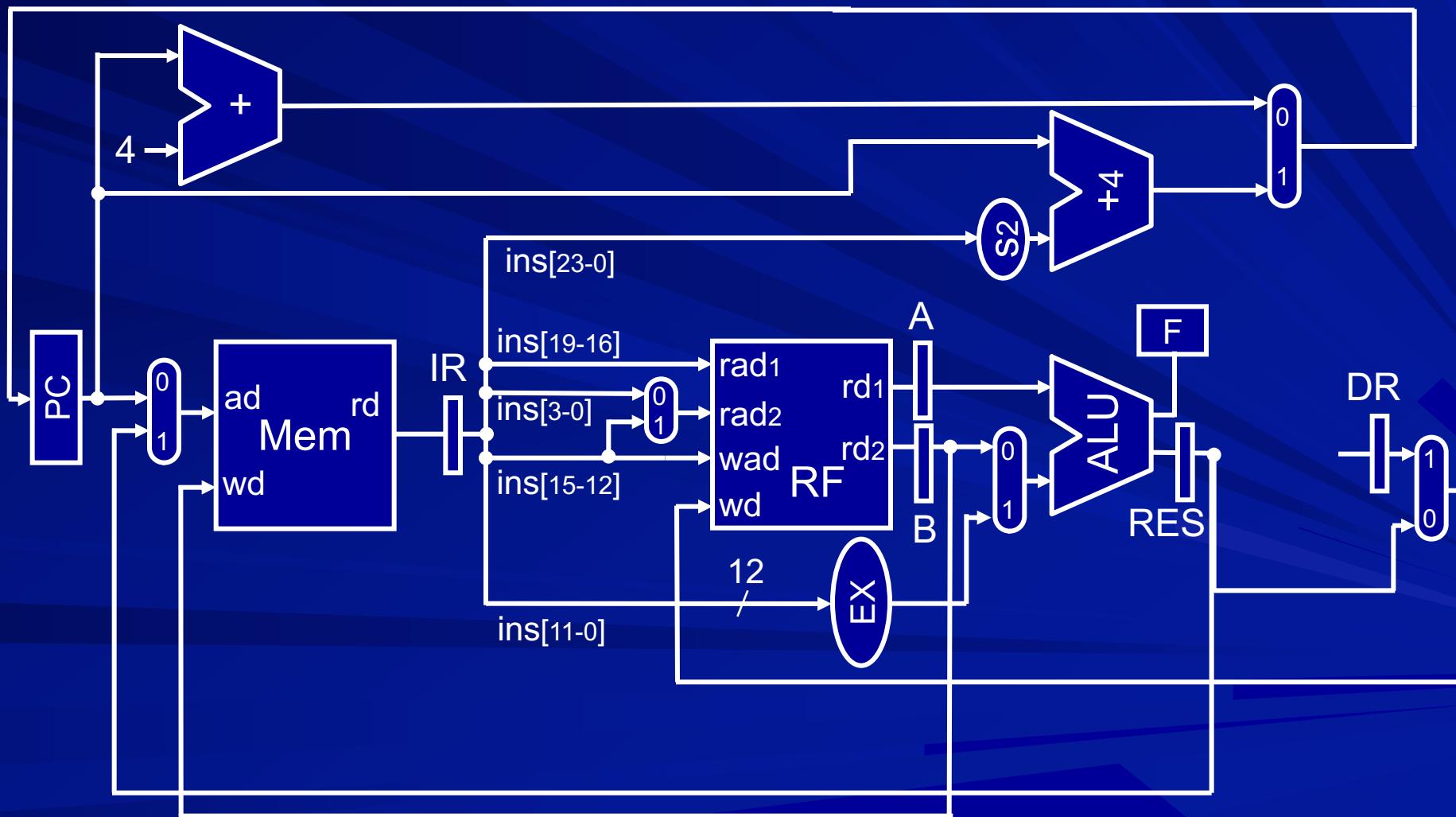
Merge IM and DM



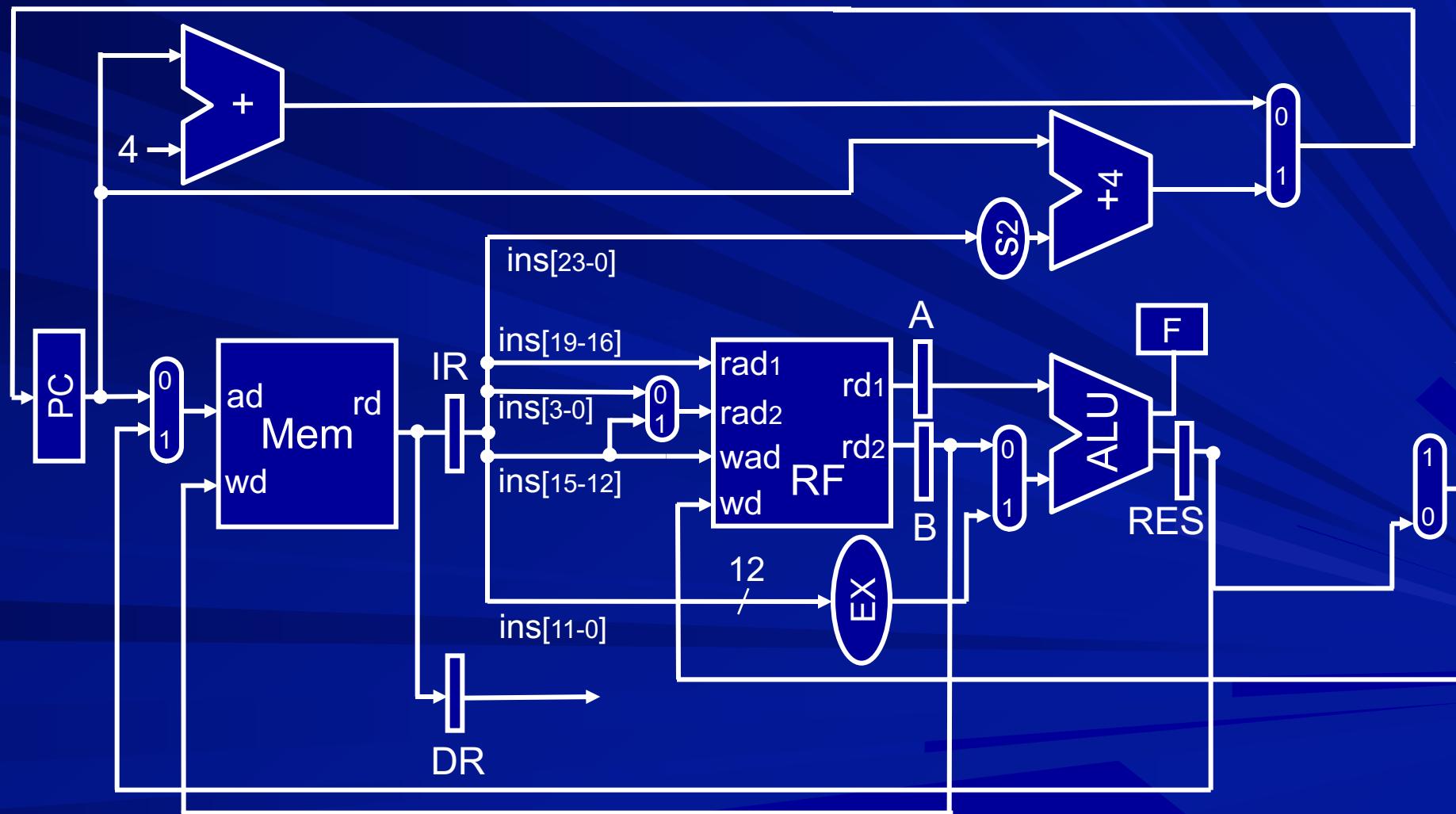
Merge IM and DM



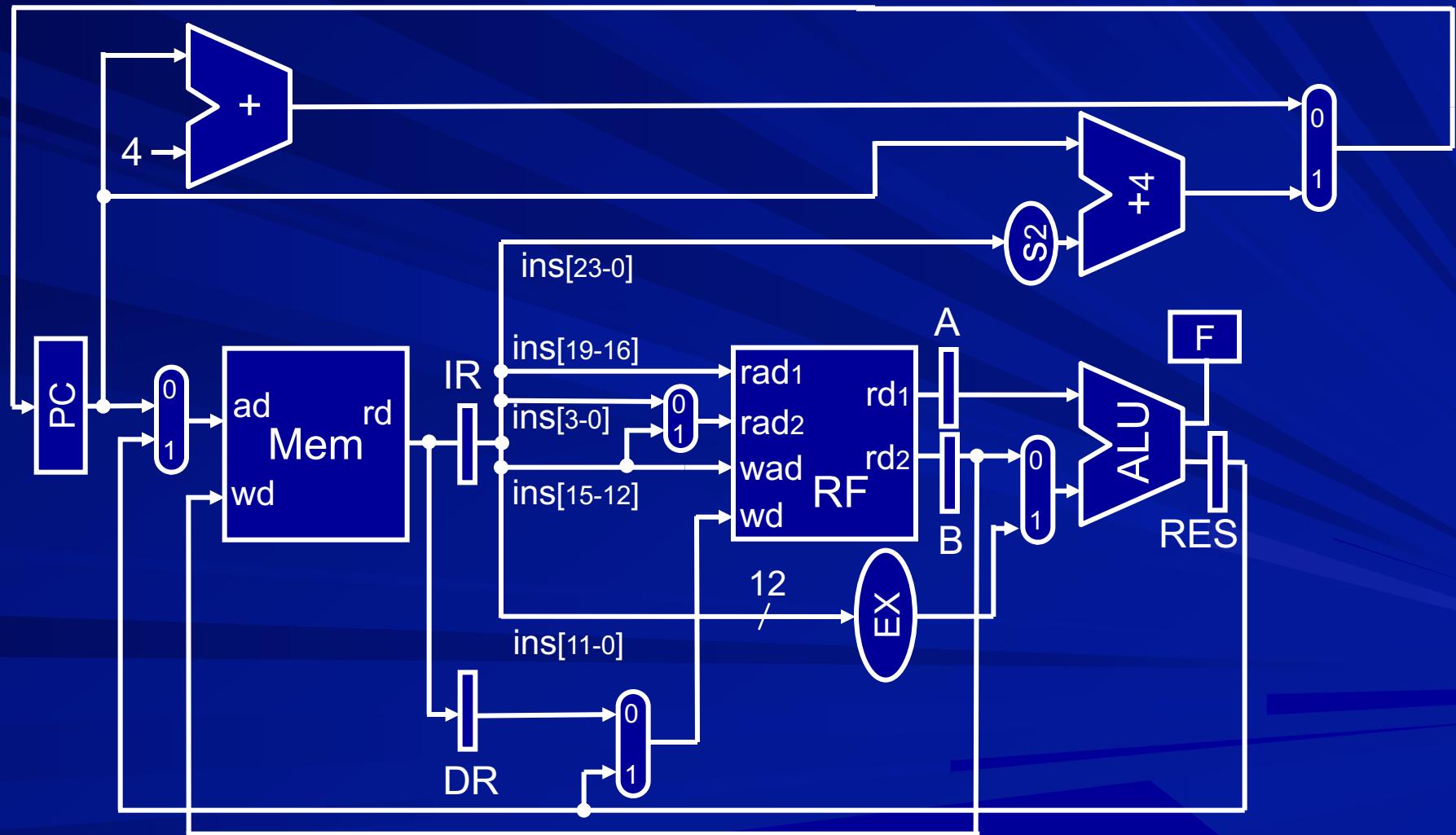
Merge IM and DM



Merge IM and DM

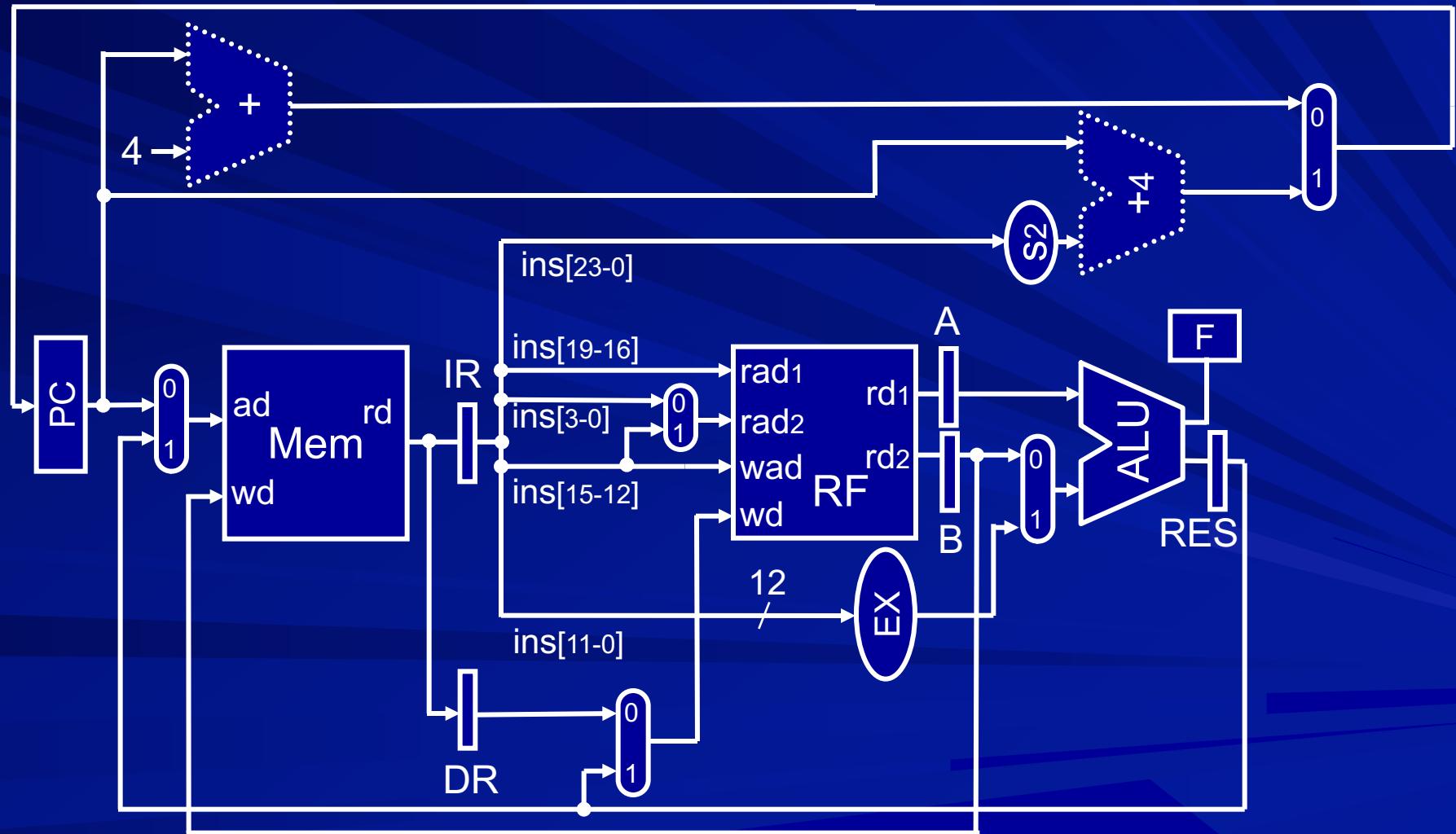


Merge IM and DM

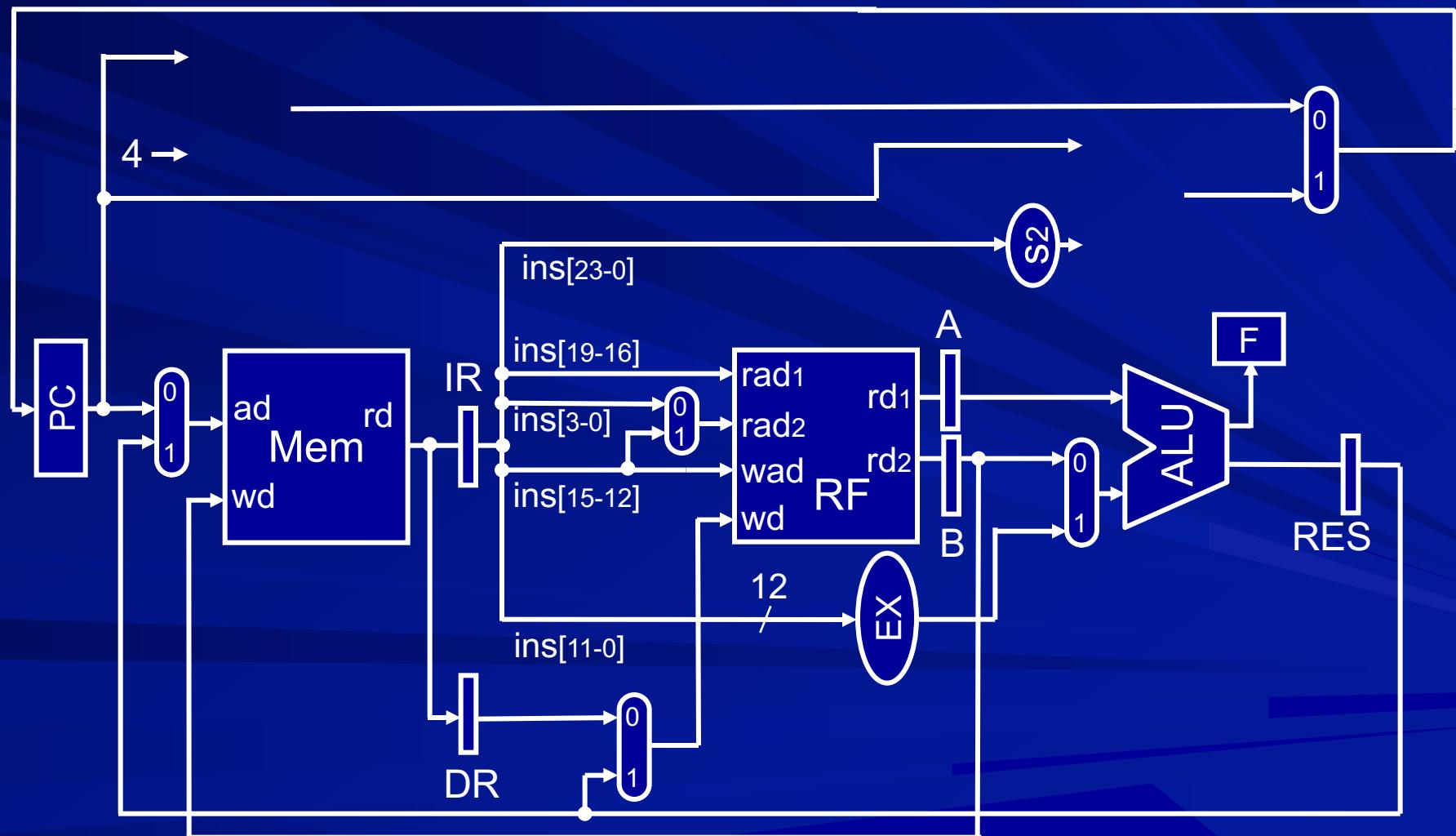


Eliminate adders

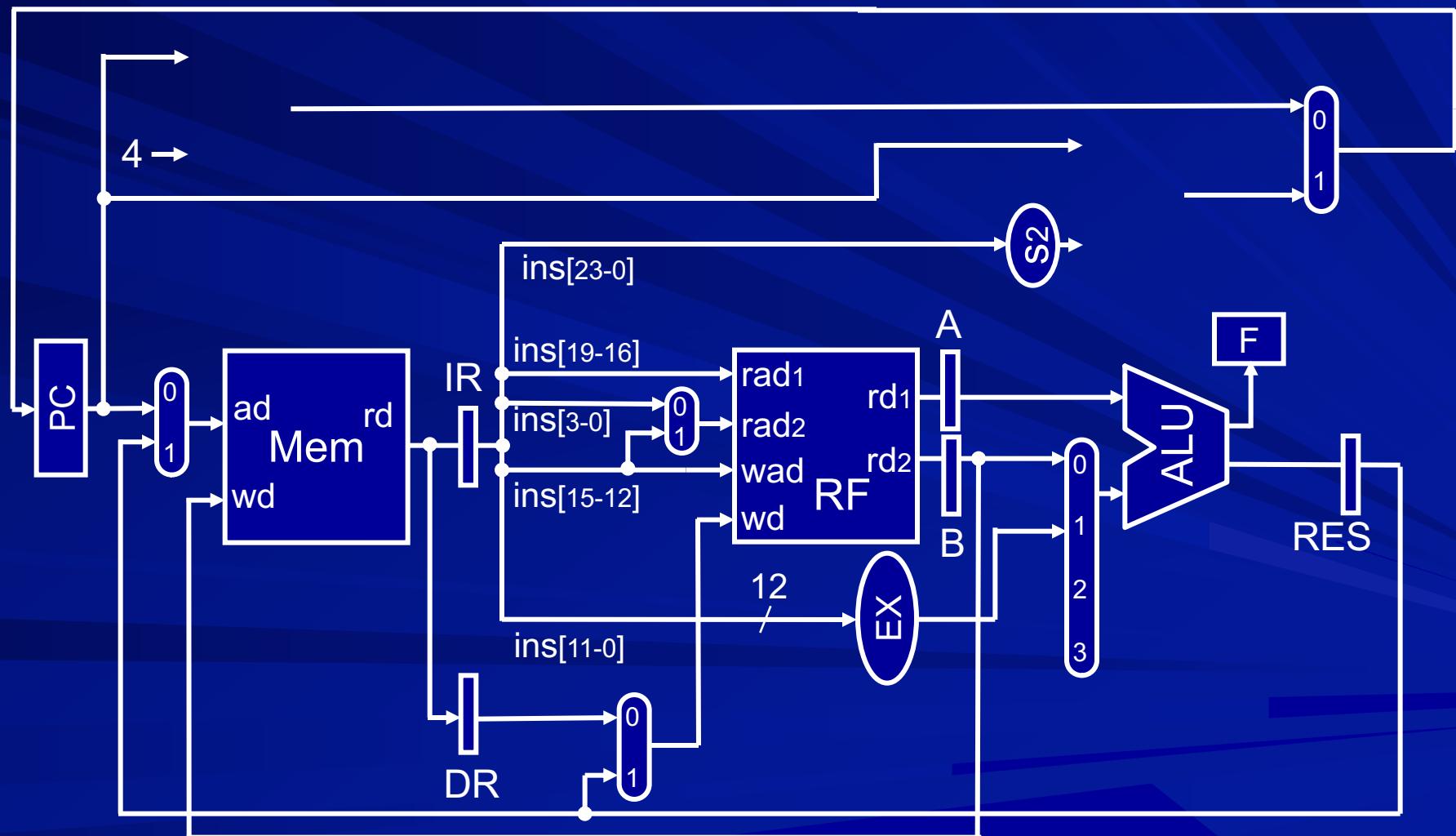
Use ALU



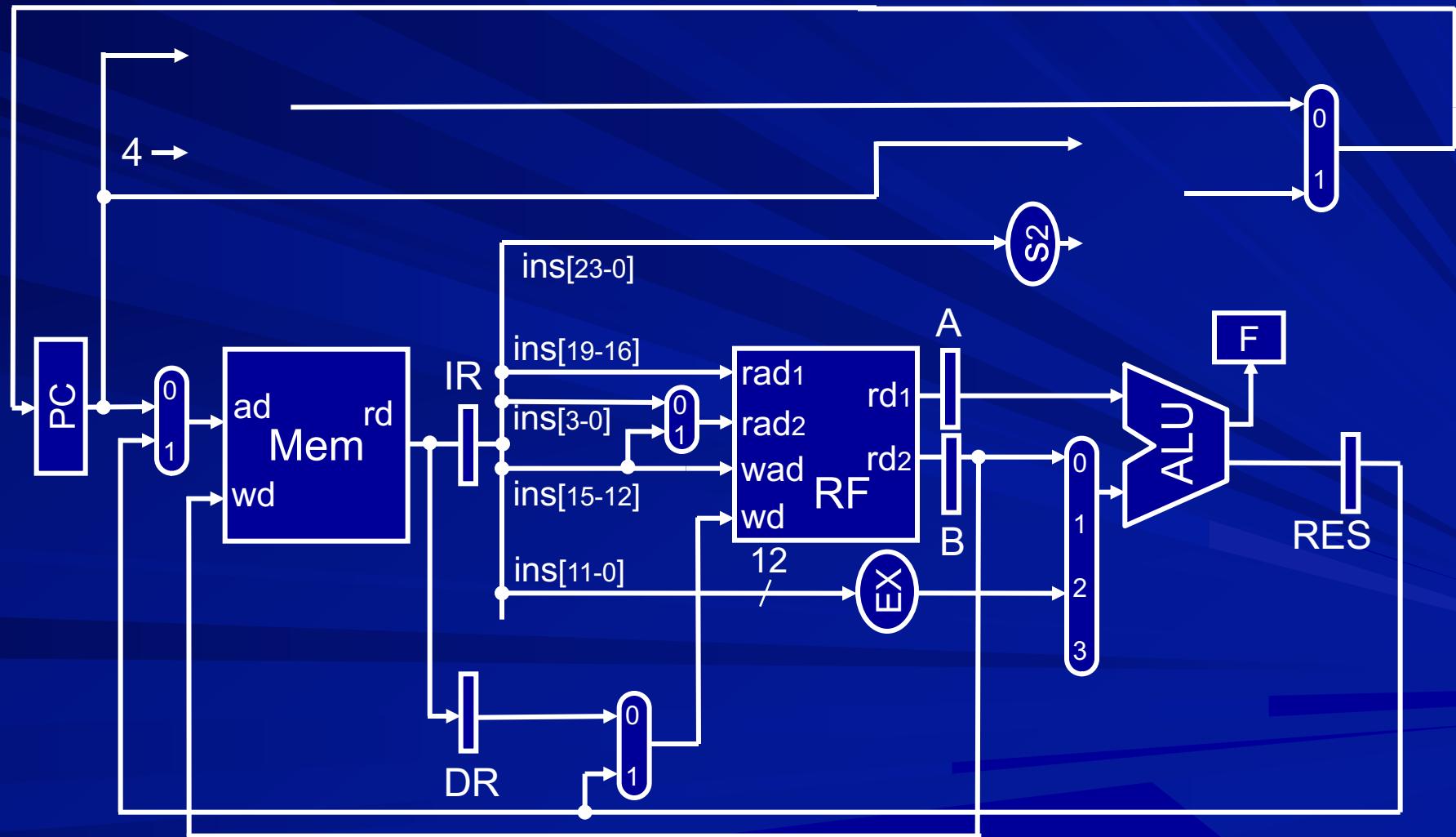
Eliminate adders



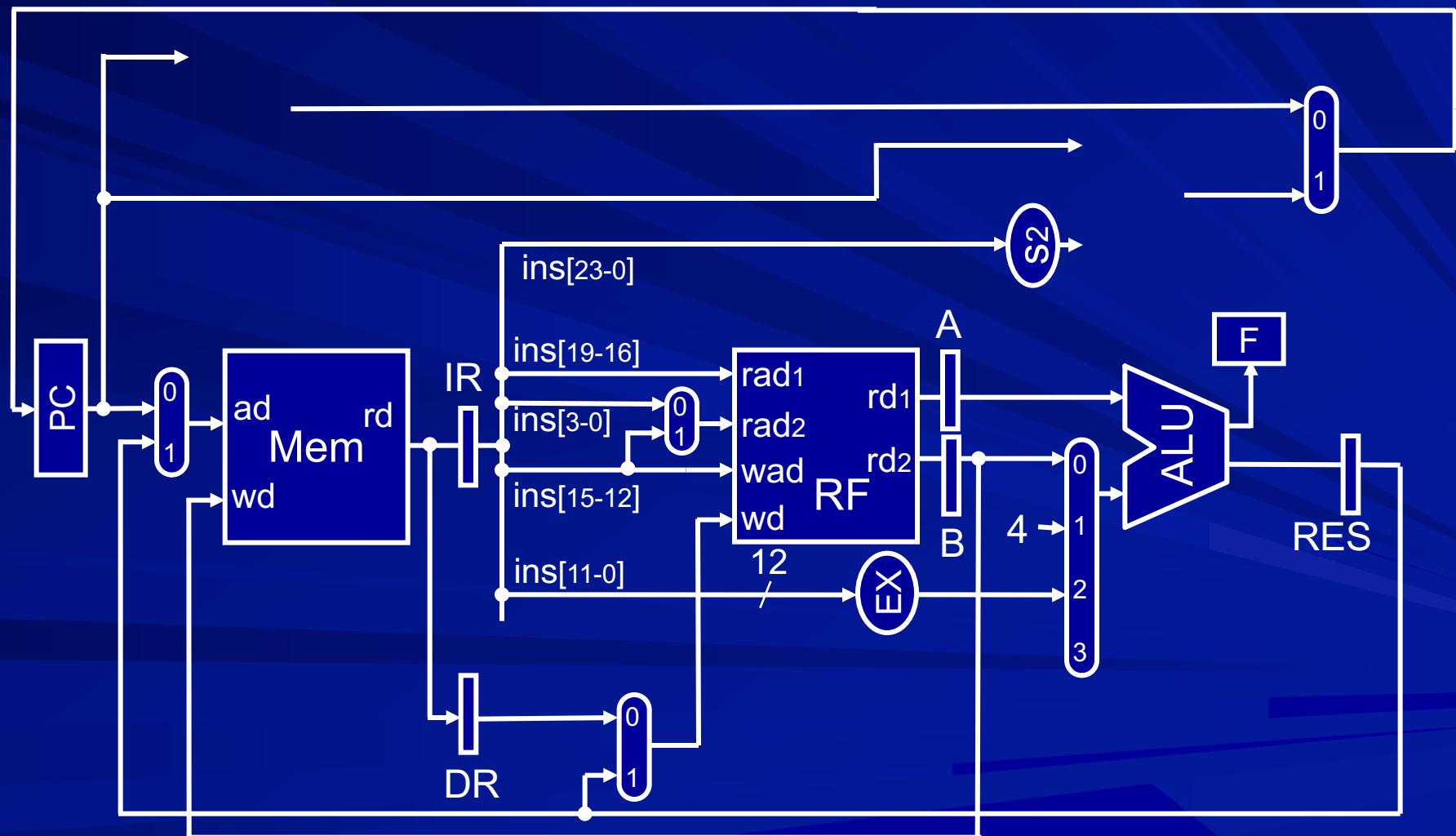
Eliminate adders



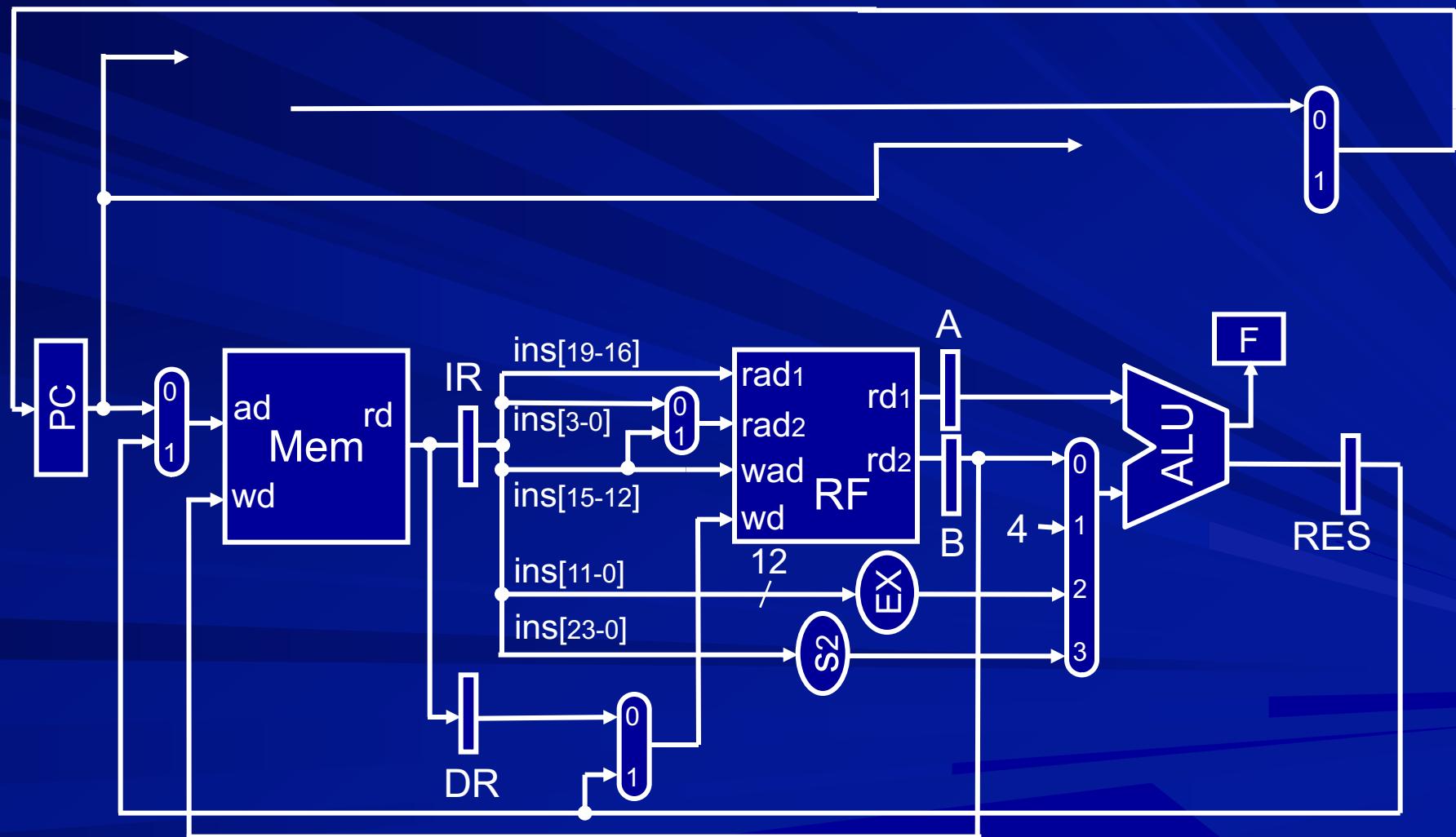
Eliminate adders



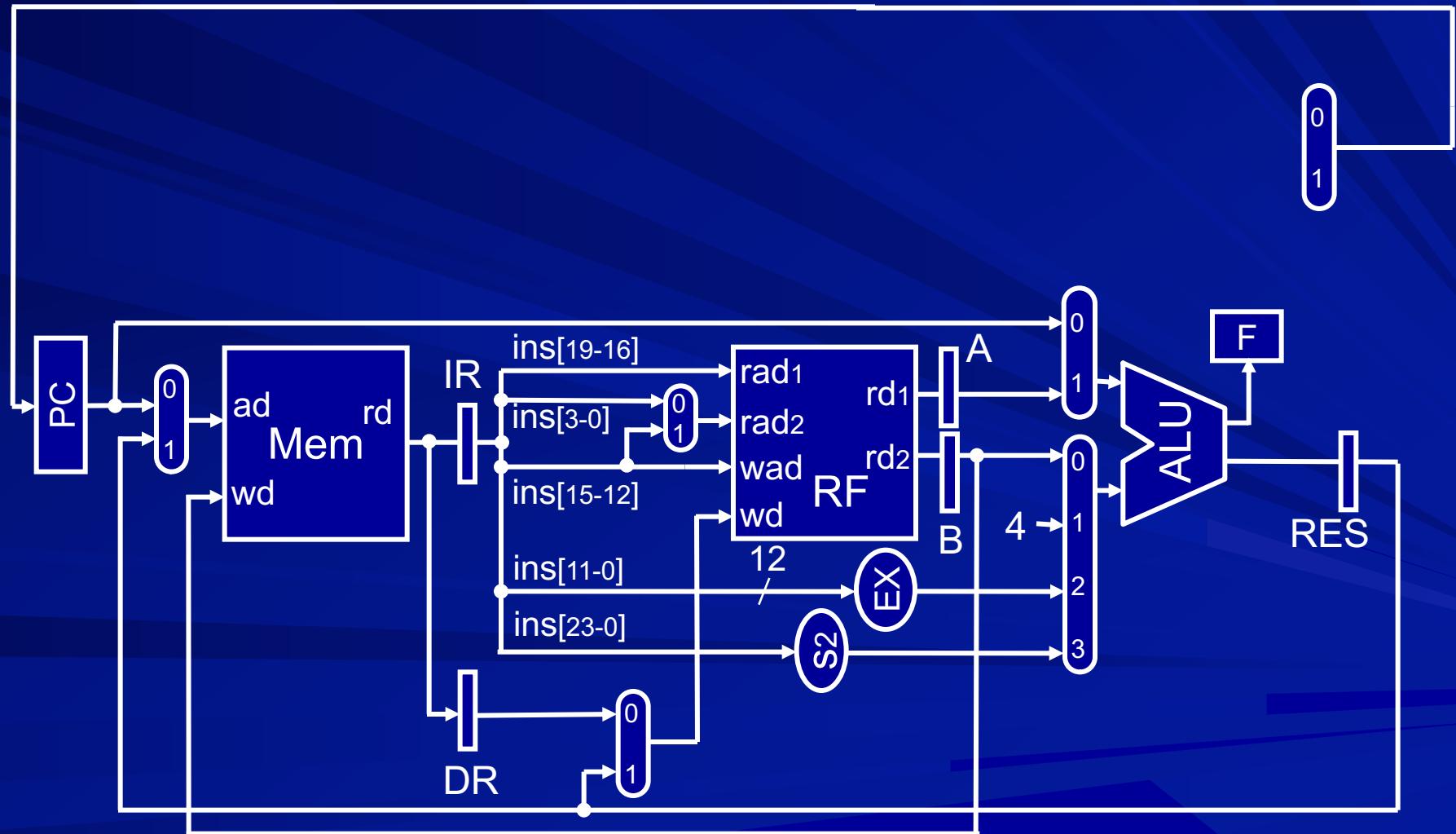
Eliminate adders



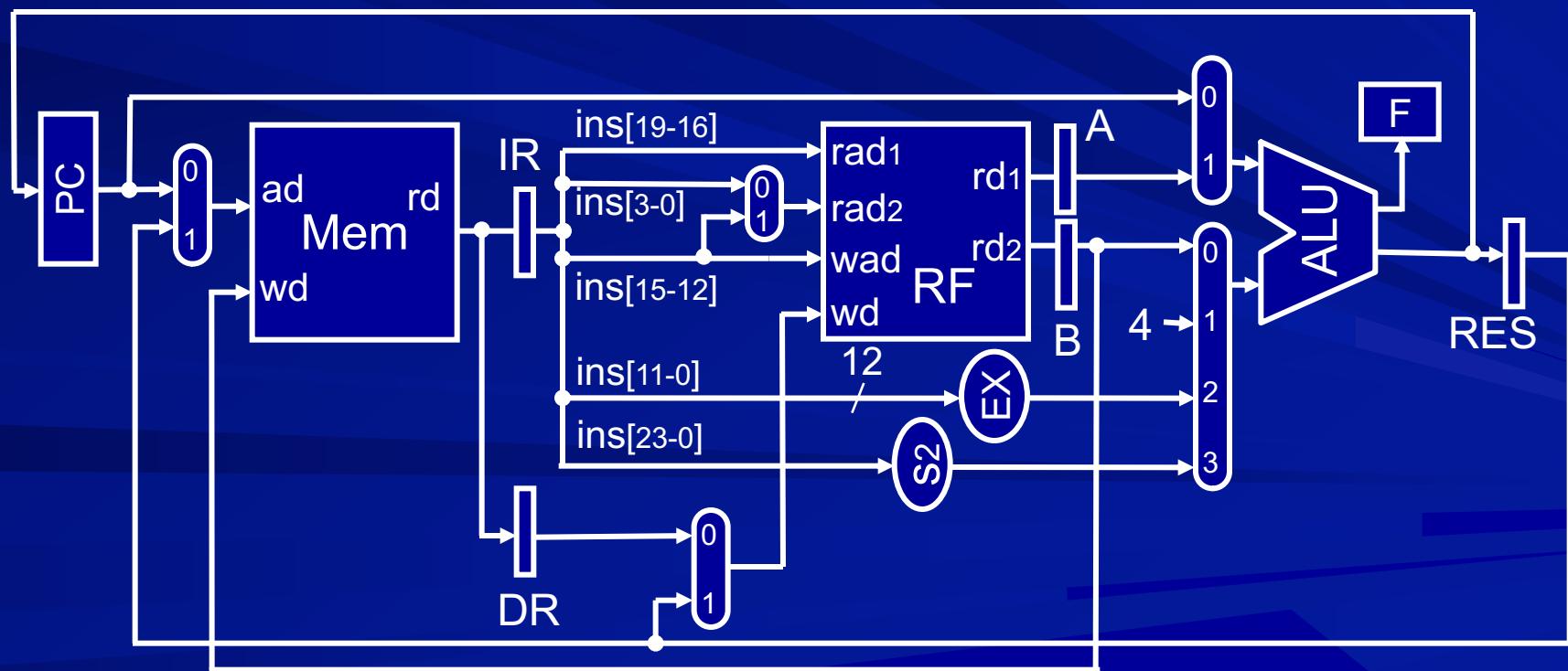
Eliminate adders



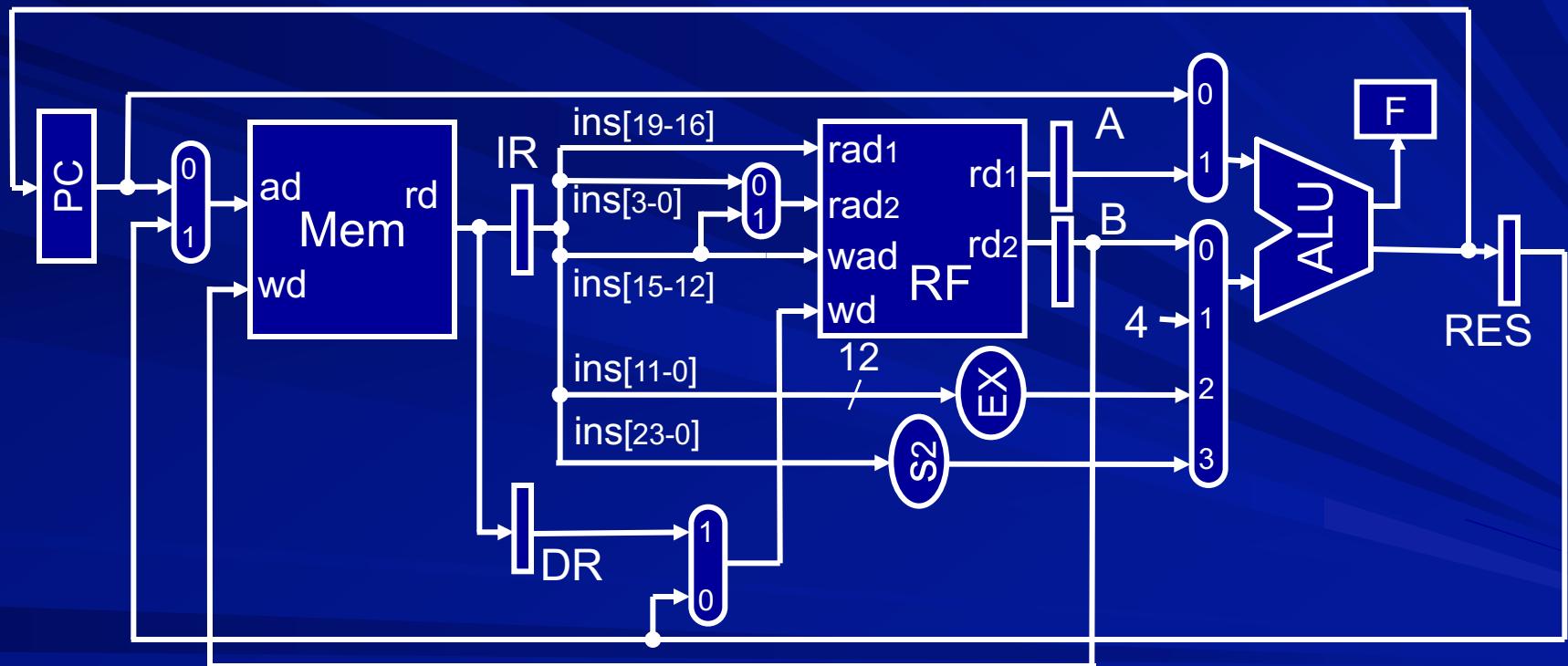
Eliminate adders



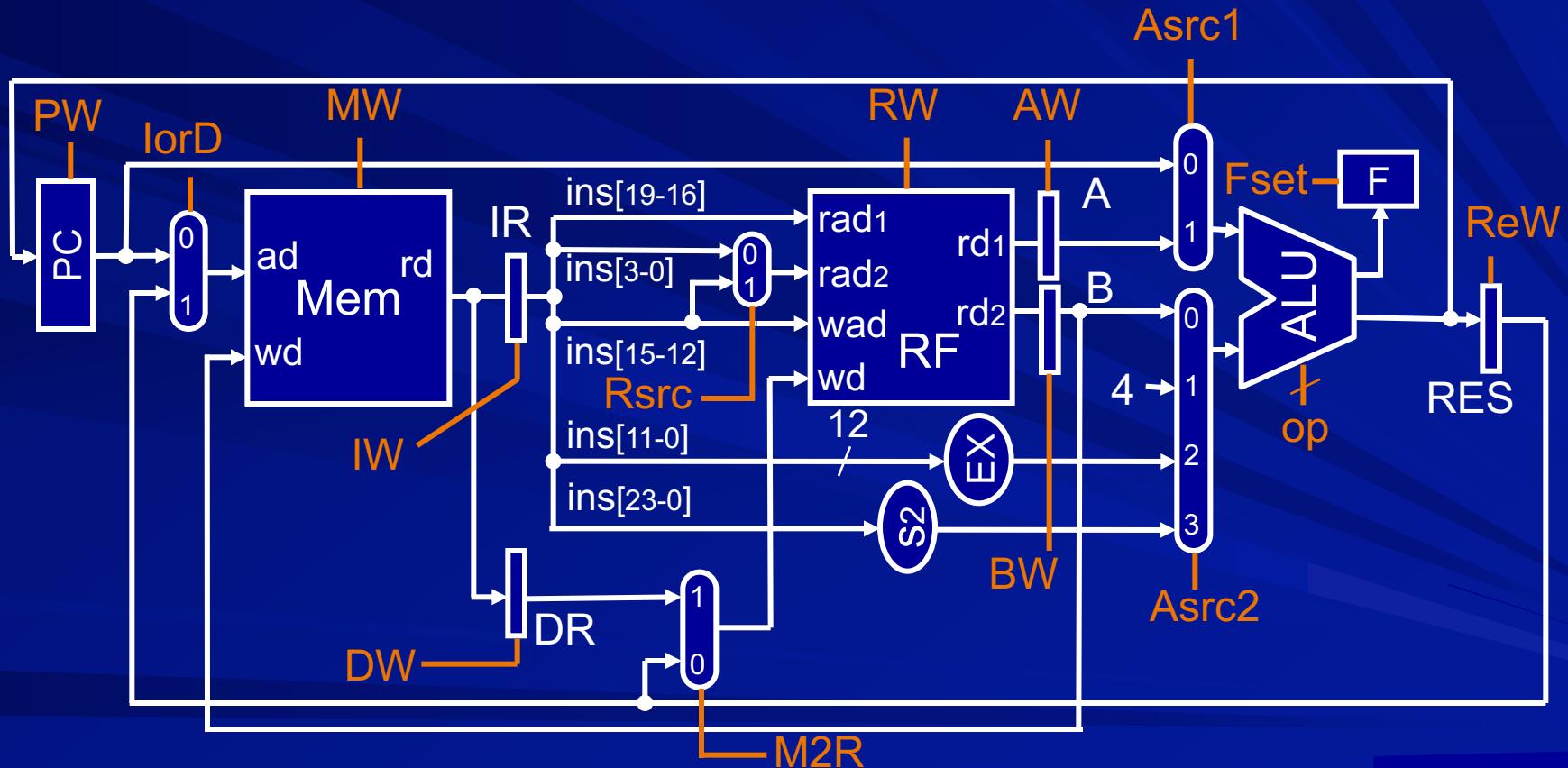
Eliminate adders



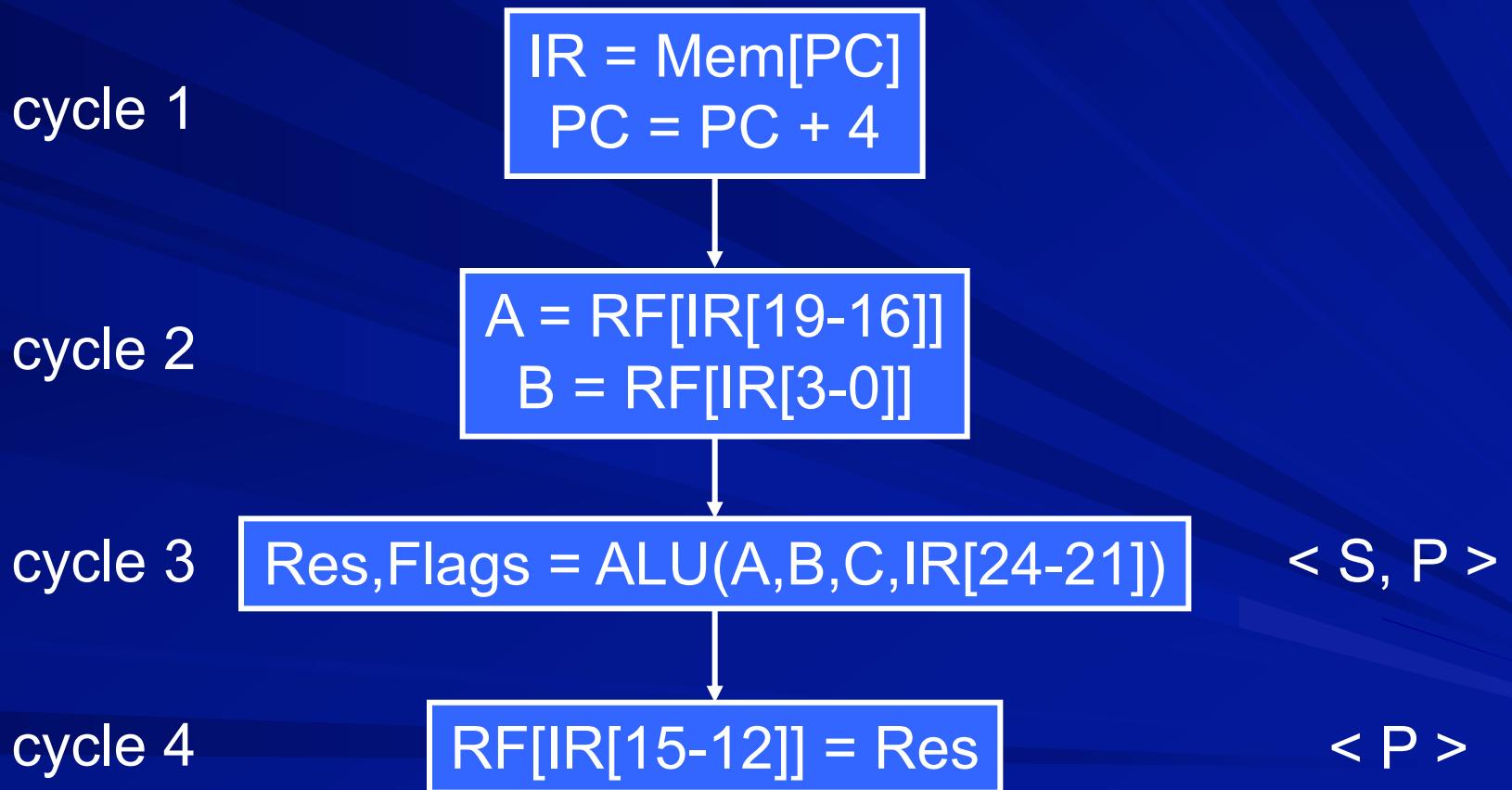
Multi-cycle Datapath



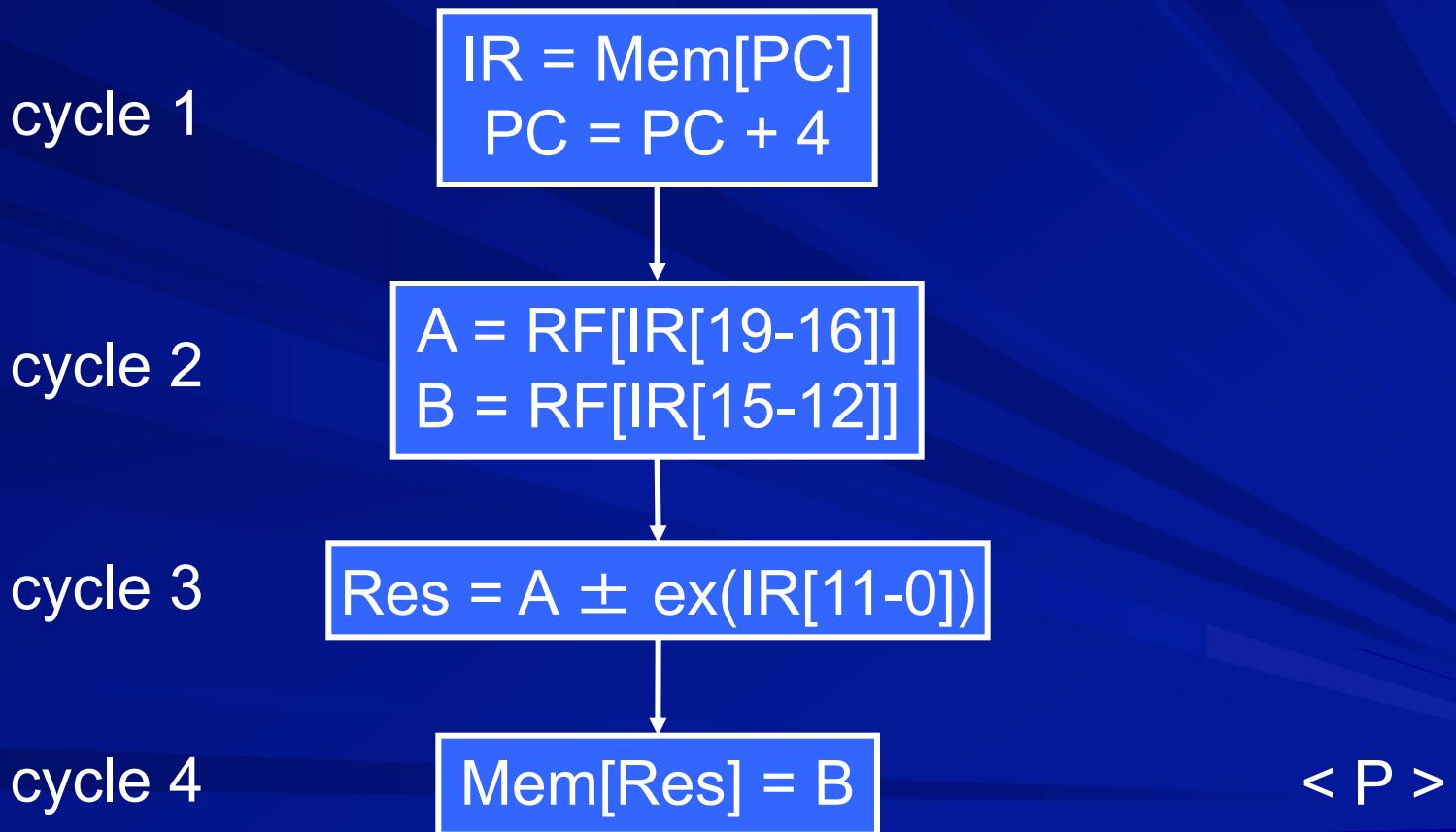
Control signals in multi-cycle datapath



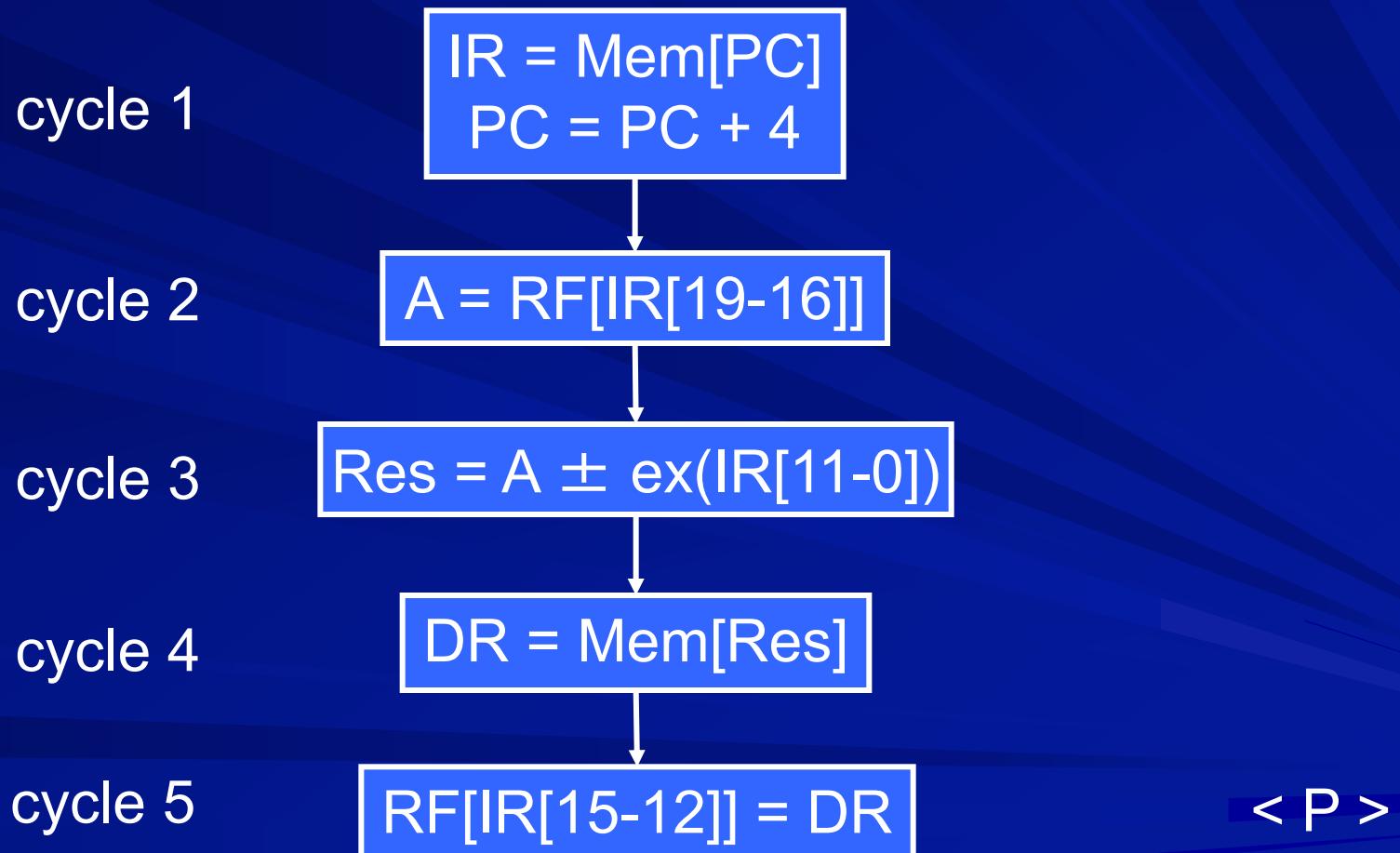
Break Instruction Execution into Cycles: DP instructions



Break Instruction Execution into Cycles: str instruction



Break Instruction Execution into Cycles: ldr instruction



Break Instruction Execution into Cycles: b instruction

cycle 1

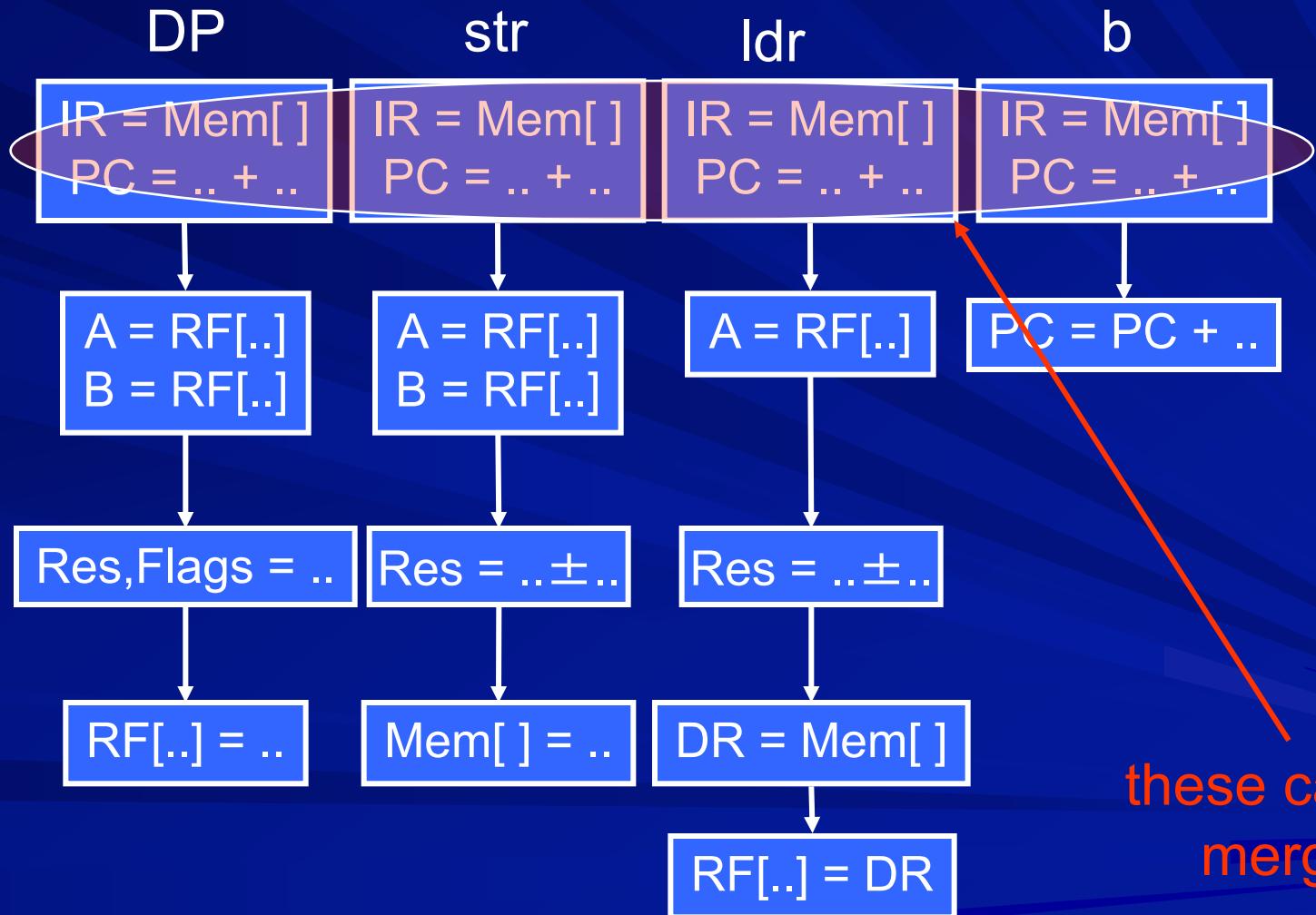
```
IR = Mem[PC]  
PC = PC + 4
```

cycle 2

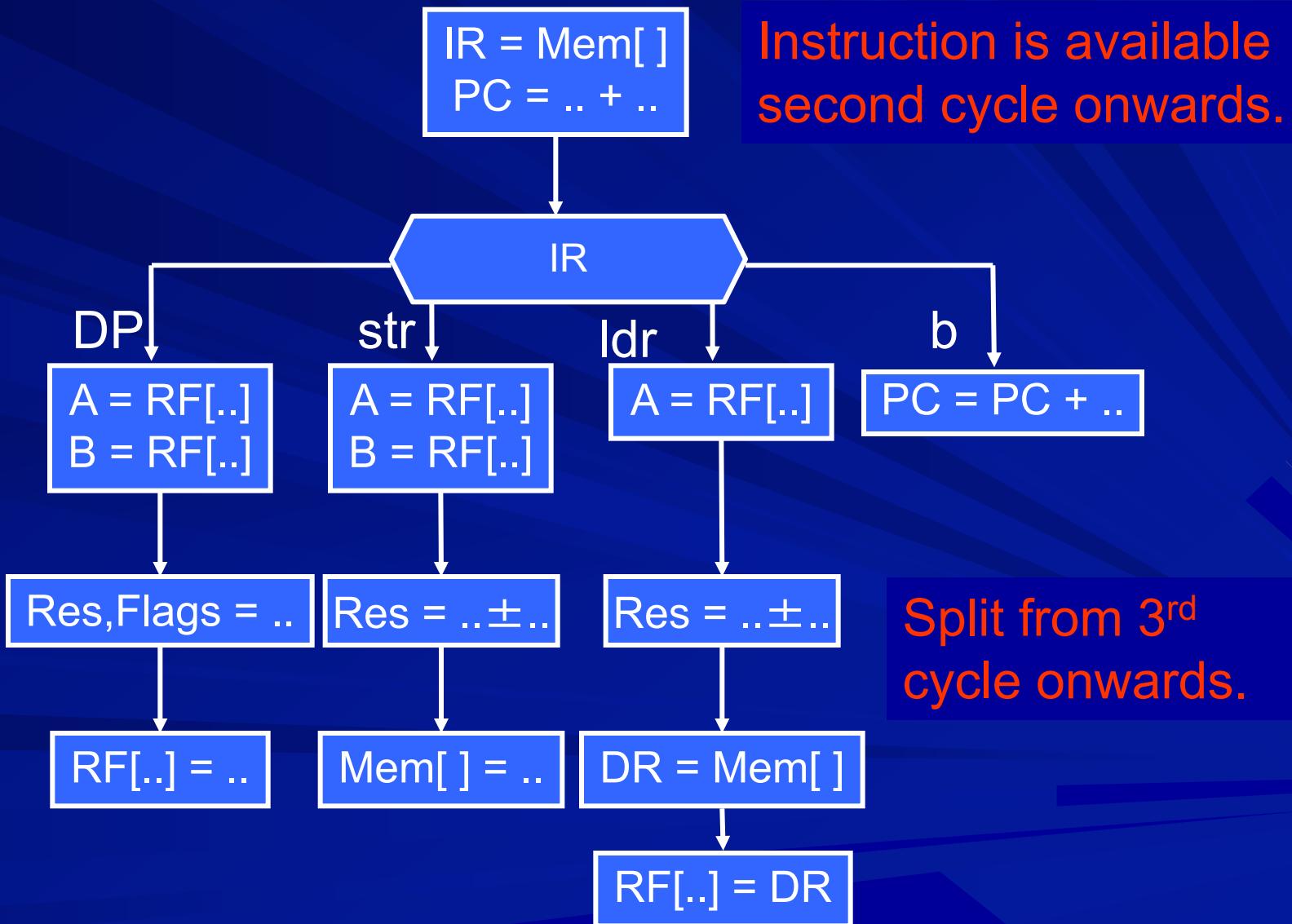
```
PC = PC + S2(IR[23-0]) + 4
```

< P >

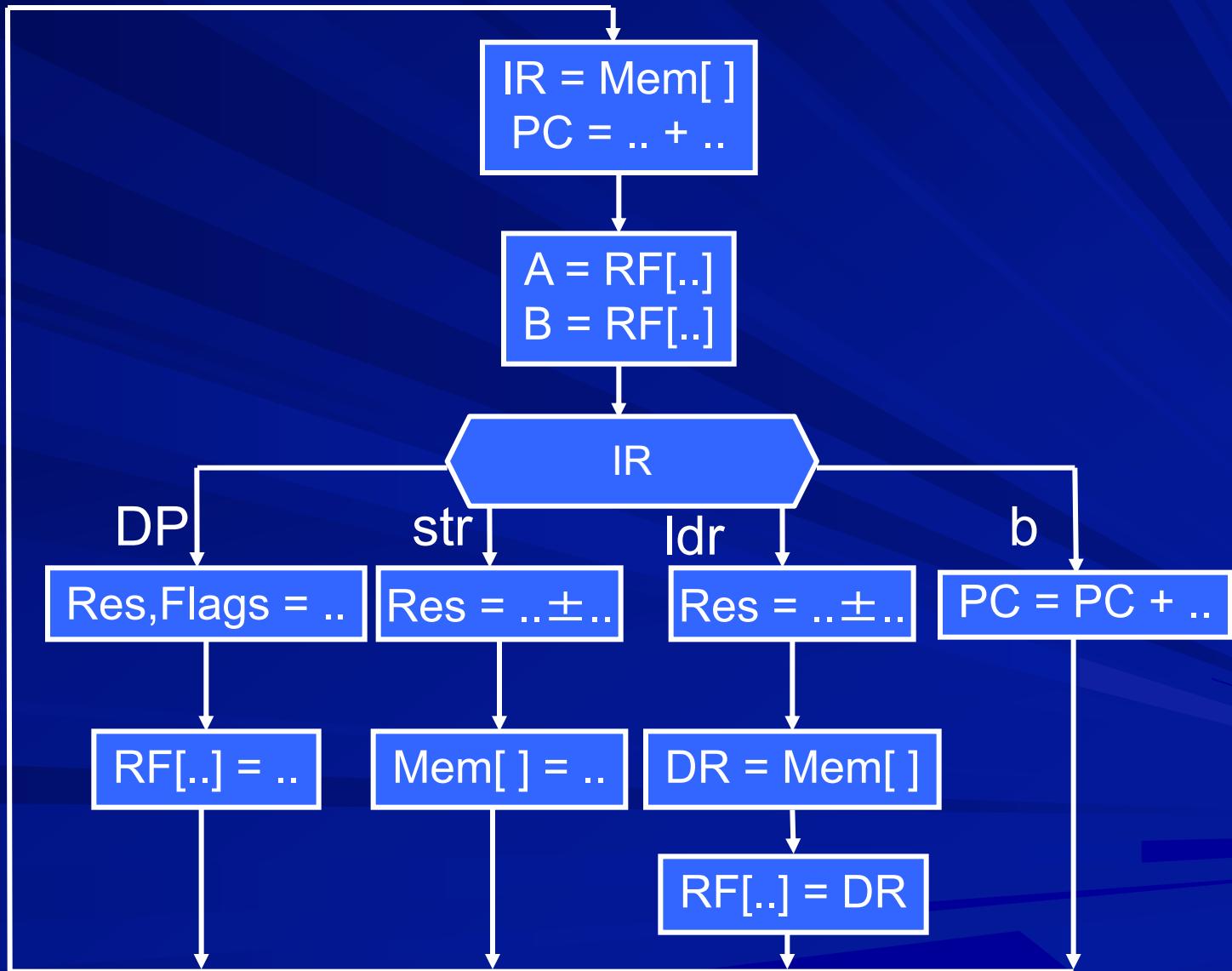
Put cycle sequences together



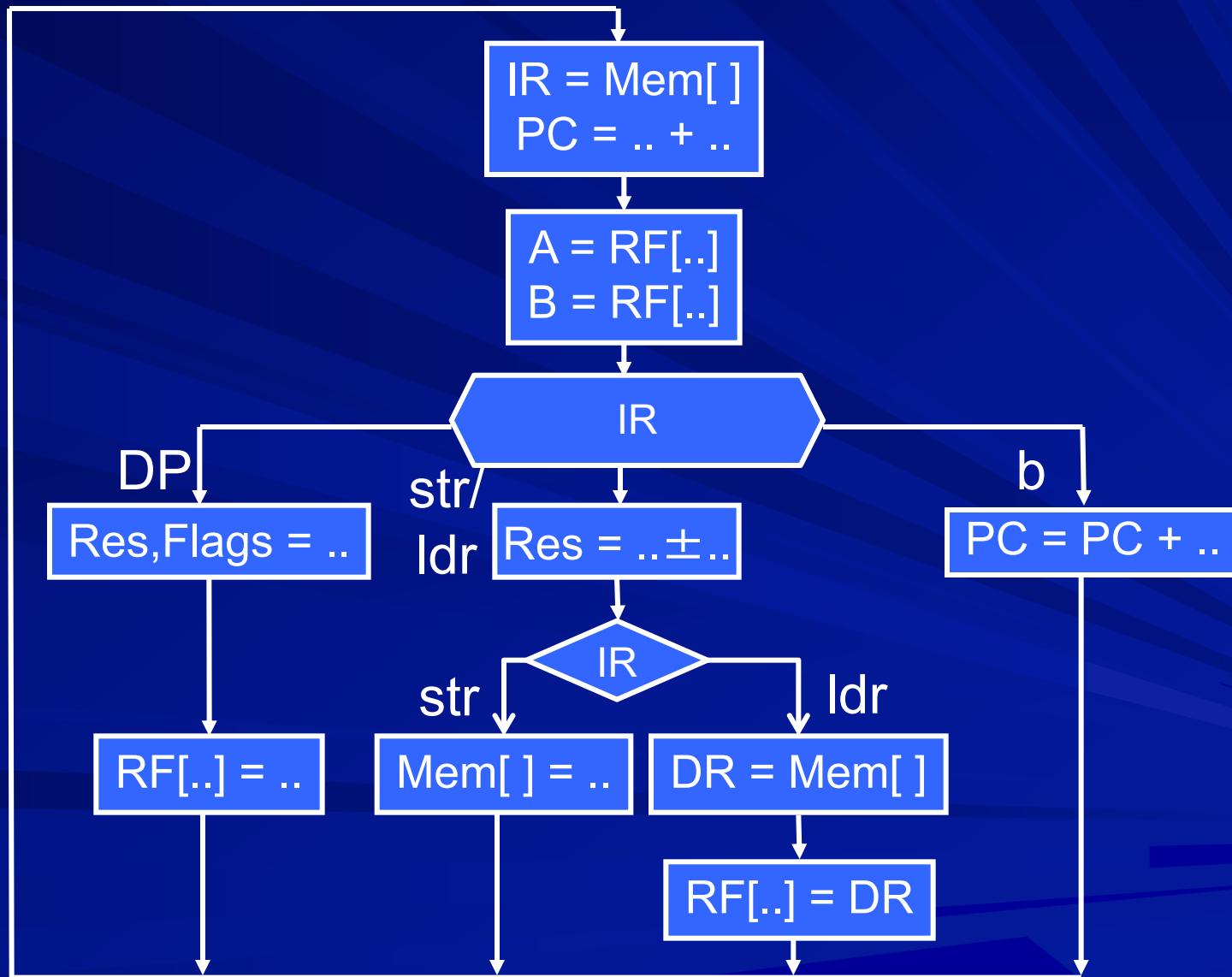
After merging fetch cycle



With a common decoding cycle



ldr, str can split after third cycle



Modified str actions

cycle 1

$$\begin{aligned} \text{IR} &= \text{Mem}[\text{PC}] \\ \text{PC} &= \text{PC} + 4 \end{aligned}$$

cycle 2

$$\begin{aligned} A &= \text{RF}[\text{IR}[19-16]] \\ B &= \text{RF}[\text{IR}[15-12]] \end{aligned}$$

cycle 3

$$\text{Res} = A \pm \text{ex}(\text{IR}[11-0])$$

cycle 4

$$\text{Mem}[\text{Res}] = B$$

$$\begin{aligned} \text{IR} &= \text{Mem}[\text{PC}] \\ \text{PC} &= \text{PC} + 4 \end{aligned}$$

$$\begin{aligned} A &= \text{RF}[\text{IR}[19-16]] \\ B &= \text{RF}[\text{IR}[3-0]] \end{aligned}$$

$$\begin{aligned} \text{Res} &= A \pm \text{ex}(\text{IR}[11-0]) \\ B &= \text{RF}[\text{IR}[15-12]] \end{aligned}$$

$$\text{Mem}[\text{Res}] = B$$

Modified ldr actions

cycle 1

$$\begin{aligned} \text{IR} &= \text{Mem}[\text{PC}] \\ \text{PC} &= \text{PC} + 4 \end{aligned}$$

cycle 2

$$A = \text{RF}[\text{IR}[19-16]]$$

cycle 3

$$\text{Res} = A \pm \text{ex}(\text{IR}[11-0])$$

cycle 4

$$\text{DR} = \text{Mem}[\text{Res}]$$

cycle 5

$$\text{RF}[\text{IR}[15-12]] = \text{DR}$$

$$\begin{aligned} \text{IR} &= \text{Mem}[\text{PC}] \\ \text{PC} &= \text{PC} + 4 \end{aligned}$$

$$\begin{aligned} A &= \text{RF}[\text{IR}[19-16]] \\ B &= \text{RF}[\text{IR}[3-0]] \end{aligned}$$

$$\begin{aligned} \text{Res} &= A \pm \text{ex}(\text{IR}[11-0]) \\ B &= \text{RF}[\text{IR}[15-12]] \end{aligned}$$

$$\text{DR} = \text{Mem}[\text{Res}]$$

$$\text{RF}[\text{IR}[15-12]] = \text{DR}$$

Modified b actions

cycle 1

```
IR = Mem[PC]  
PC = PC + 4
```

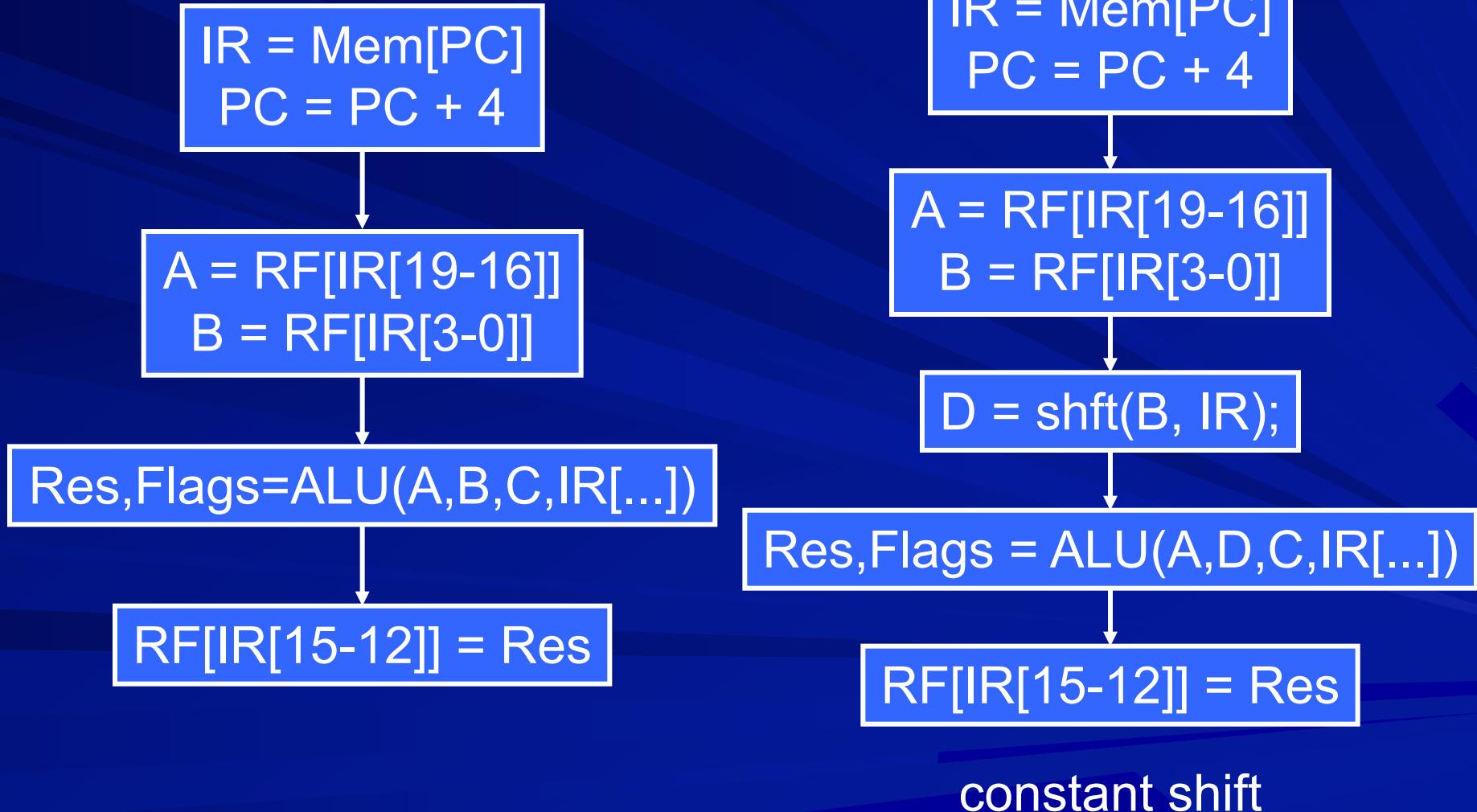
cycle 2

```
A = RF[IR[19-16]]  
B = RF[IR[3-0]]
```

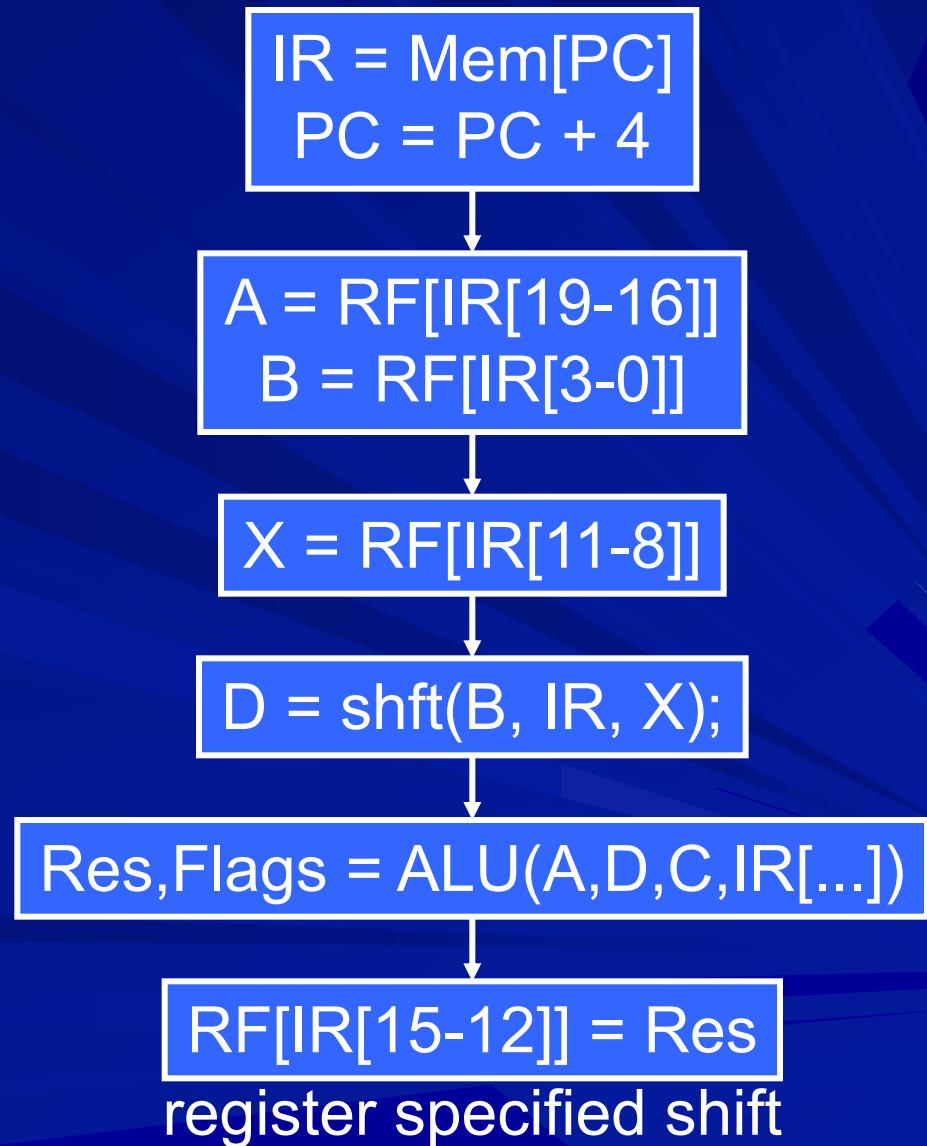
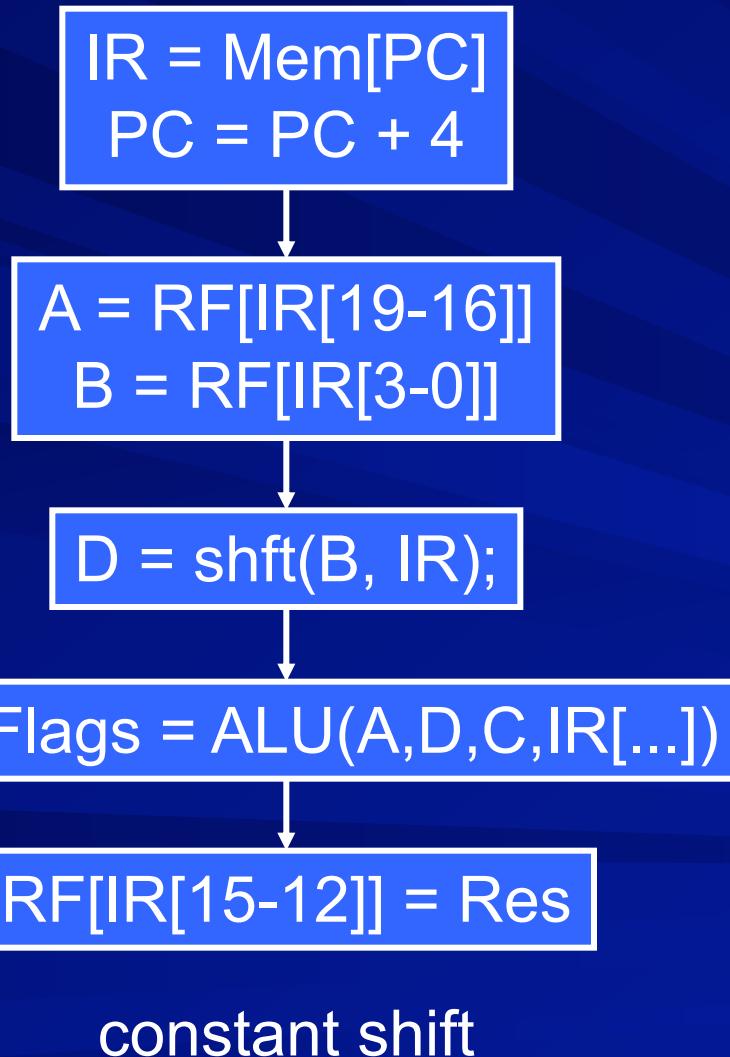
cycle 3

```
PC = PC + S2(IR[23-0]) + 4
```

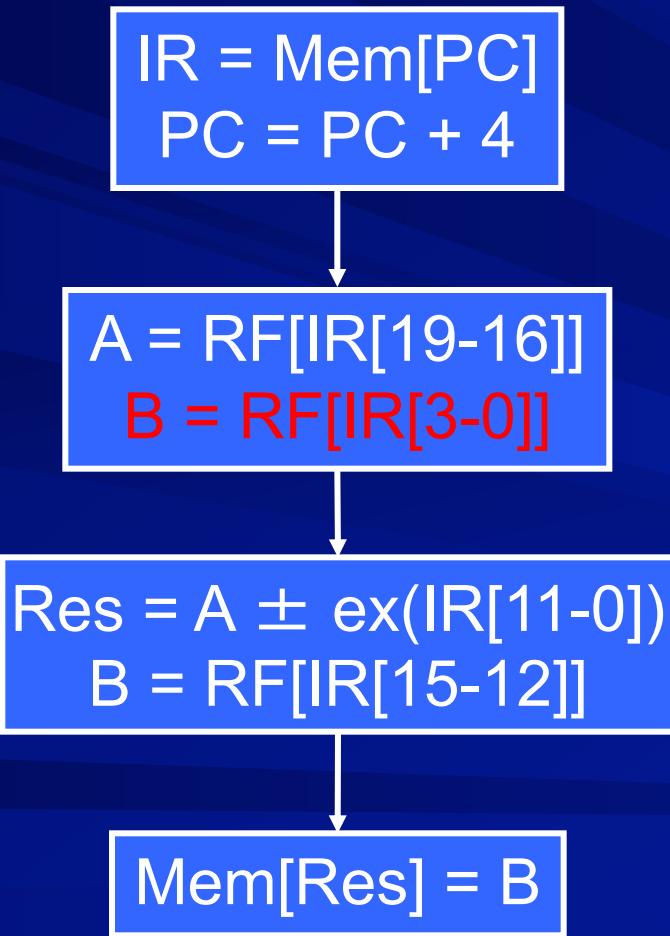
Other forms of DP instructions



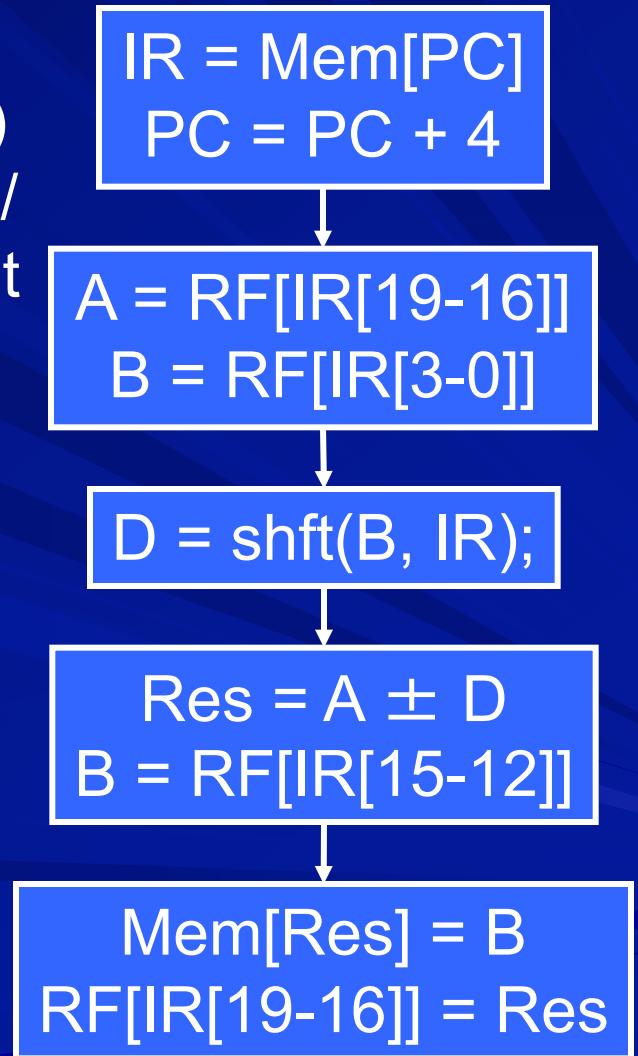
Other forms of DP instructions



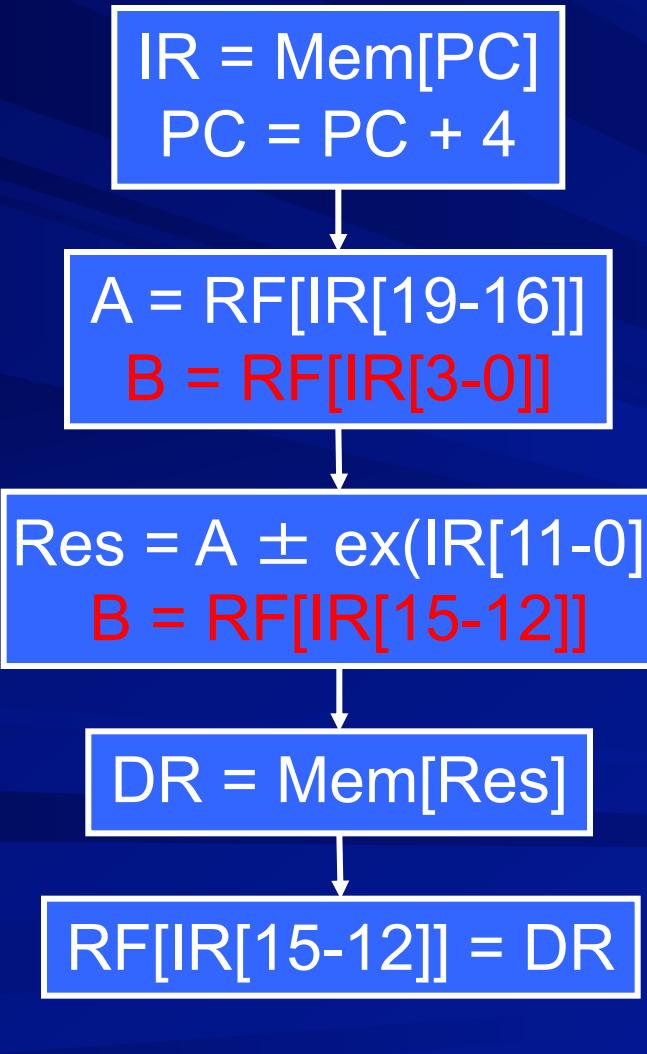
Other forms of str instruction



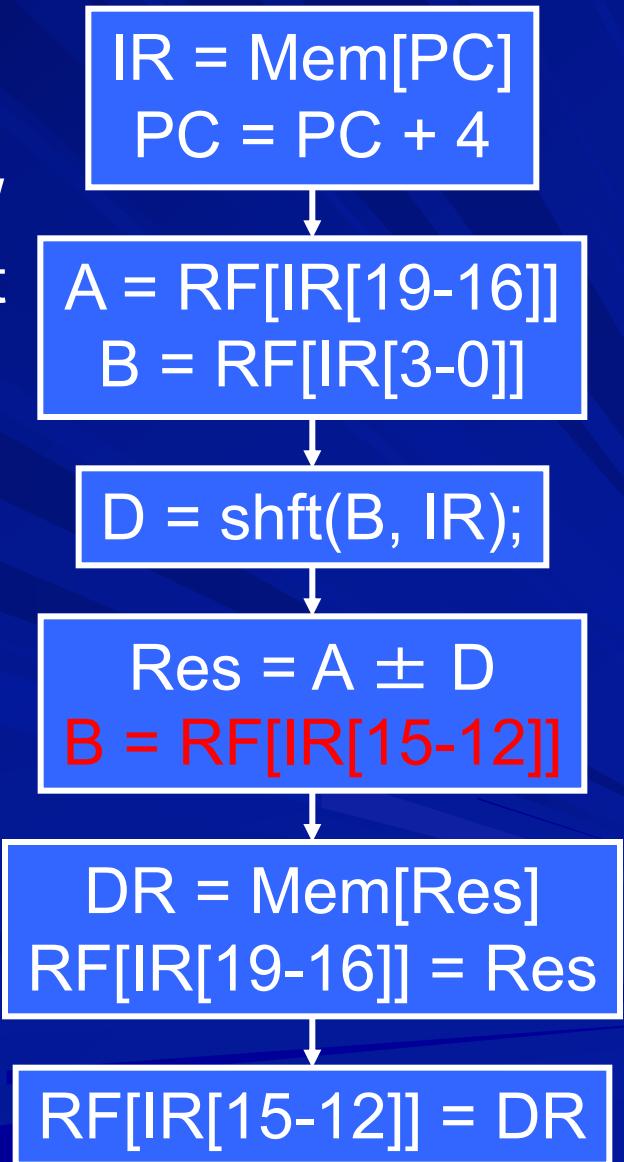
auto (pre)
increment/
decrement
register
offset



Other forms of ldr instruction



auto (pre)
increment/
decrement
register
offset



Thanks