National Institute of Technology Karnataka Surathkal
Department of Information Technology

IT 301 Parallel Computing

Scalar Processors – Instruction Pipeline

Dr. Geetha V

Assistant Professor

Dept of Information Technology

NITK Surathkal

Index

- 1: Instruction Pipeline - Introduction
- 2. Five Stage pipeline design
- 3: Issues of pipeline and Design Solutions
- 4: Performance evaluation of pipelining

1: Introduction

Course Plan: Theory:

Part A: Parallel Computer Architectures

Week 1,2,3: ***Introduction to Parallel Computer Architecture:***

Parallel Computing,

Parallel architecture,

bit level, instruction level , data level and task level parallelism.

Instruction level parallelisms: pipelining (Data and control instructions),

scalar processors and superscalar processors,

vector processors.

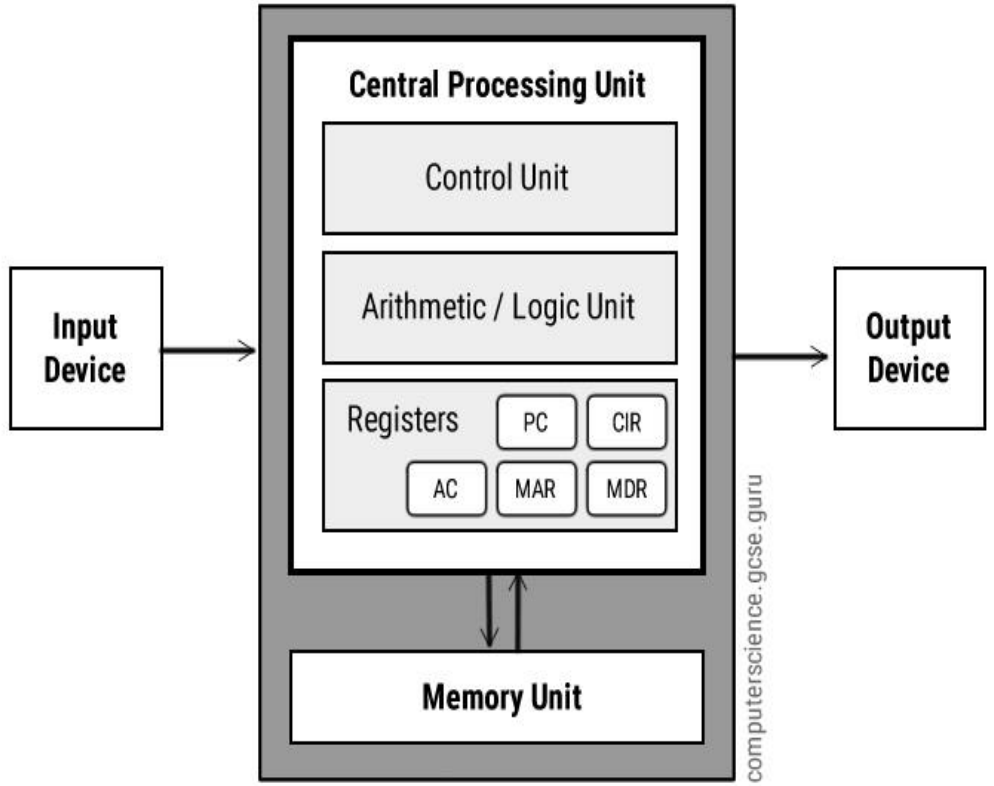
Parallel computers and computation.

1: Instruction Pipeline - Introduction

Features

- Established by John von Neumann in 1945.
- Stored Program Concept
- **PC :** Program Counter
- **CIR:** Current Instruction Register
- **AC:** Accumulator,
- **MAR:** Memory Address Register,
- **MDR:** Memory Data Register
- **Buses:** Address, Data and Control
- **Execution :** One Instruction at a Time

Von Neumann Architecture



https://www.computerscience.gcse.guru/theory/von-neumann-architecture

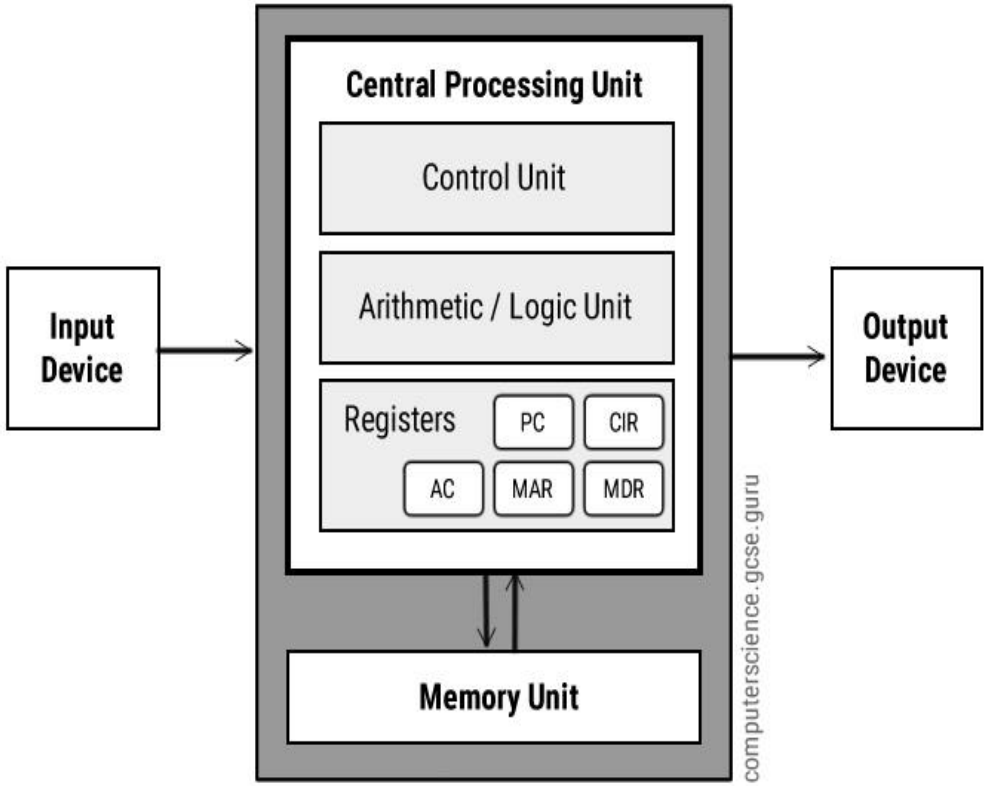
1: Instruction Pipeline - Introduction

Features of RISC

- **RISC:** Reduced Instruction Set Computer
- Fixed **instruction size**
- **Load/Store** Memory Operation
- Few **Addressing Modes**
- **Register to Register** operation

- These features leads towards **pipelined execution**

Von Neumann Architecture



https://www.computerscience.gcse.guru/theory/von-neumann-architecture

2. Five Stage Pipeline Design

- Fetch Stage: The address in program counter is sent to Instruction register and instruction is fetched from memory. PC is incremented.
- Decode Stage: The operands are fetched from register file; Sign extension done if necessary.
- Execution Stage: The instruction is executed.
- Memory Stage: The Load and Store instructions execution.
- Write Back Stage: The results are written back to register file.

Fetch (F)	Decode (D)	Execute (E)	Memory Access (M)	Write Back (W)

2. Five Stage Pipeline Design

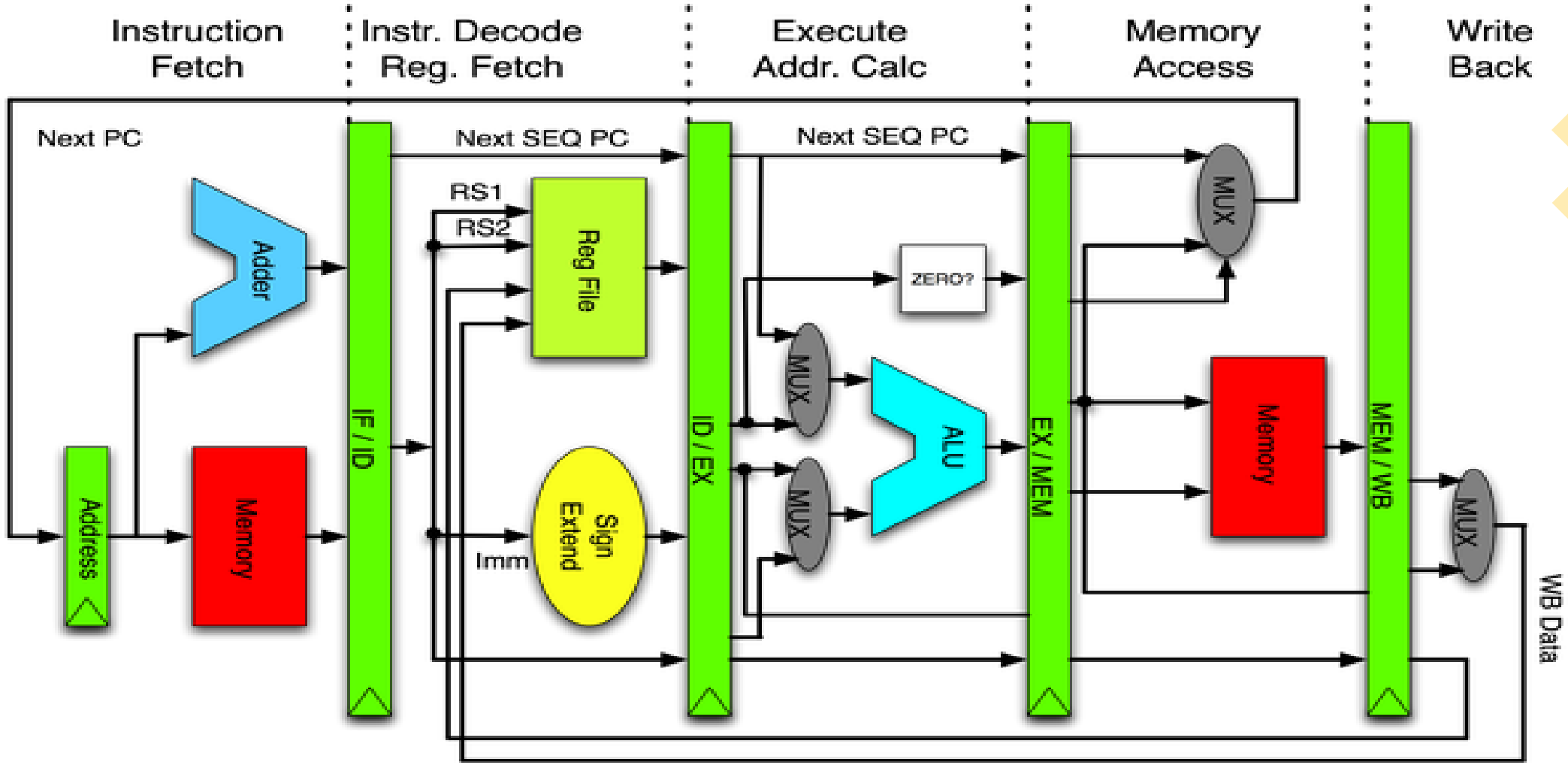


Image from Wikimedia

2. Five Stage Pipeline Design

	Instruction	(F)	(D)	(E)	(M)	(W)	
I_1	Move R5, #2000h	I_1					
I_2	Move R1, #15	I_2	I_1				
I_3	Move R2, #16	I_3	I_2	I_1			
I_4	Move R3, R1	I_4	I_3	I_2	I_1		
I_5	Add R1, R3	I_5	I_4	I_3	I_2	I_1	I1: Complete
I_6	Move R4, R2	I_6	I_5	I_4	I_3	I_2	I2: Complete
I_7	Add R2, R4	I_7	I_6	I_5	I_4	I_3	I3: Complete
I_8	Add R1, R2	I_8	I_7	I_6	I_5	I_4	I4: Complete
I_9	Store [R5], R1	I_9	I_8	I_7	I_6	I_5	I5: Complete
I_{10}			I_9	I_8	I_7	I_6	I6: Complete
I_{11}				I_9	I_8	I_7	I7: Complete
I_{12}					I_9	I_8	I8: Complete
I_{13}						I_9	I9: Complete

- Adding (2x+2y)
- x=15 y=16
- If no pipeline, then this program takes

9 x 5 = 45 cycles

- If 5 stage pipeline is used, then it takes

5 + 8 = 13 cycles

(first instruction takes 'n' stages/cycle) + (n-1) instructions

3: Issues of pipeline and Design Solutions

	Instruction	(F)	(D)	(E)	(M)	(W)	
I_1	Move R5, #2000h	I_1					
I_2	Move R1, #15	I_2	I_1				
I_3	Move R2, #16	I_3	I_2	I_1			
I_4	Move R3, R1	I_4	I_3	I_2	I_1		
I_5	Add R1, R3	I_5	I_4	I_3	I_2	I_1	I1: Complete
I_6	Move R4, R2	I_6	I_5	I_4	I_3	I_2	I2: Complete
I_7	Add R2, R4	I_7	I_6	I_5	I_4	I_3	I3: Complete
I_8	Add R1, R2	I_8	I_7	I_6	I_5	I_4	I4: Complete
I_9	Store [R5], R1	I_9	I_8	I_7	I_6	I_5	I5: Complete
I_{10}			I_9	I_8	I_7	I_6	I6: Complete
I_{11}				I_9	I_8	I_7	I7: Complete
I_{12}					I_9	I_8	I8: Complete
I_{13}						I_9	I9: Complete

- **Data Dependency**

Dependency of instructions due to operand values unavailability

- **Control Dependency**

Dependency due to unresolved decision on control instructions.

3: Issues of pipeline and Design Solutions

Cycle 1: I1 in fetch stage

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I1:Move R5, #2000h

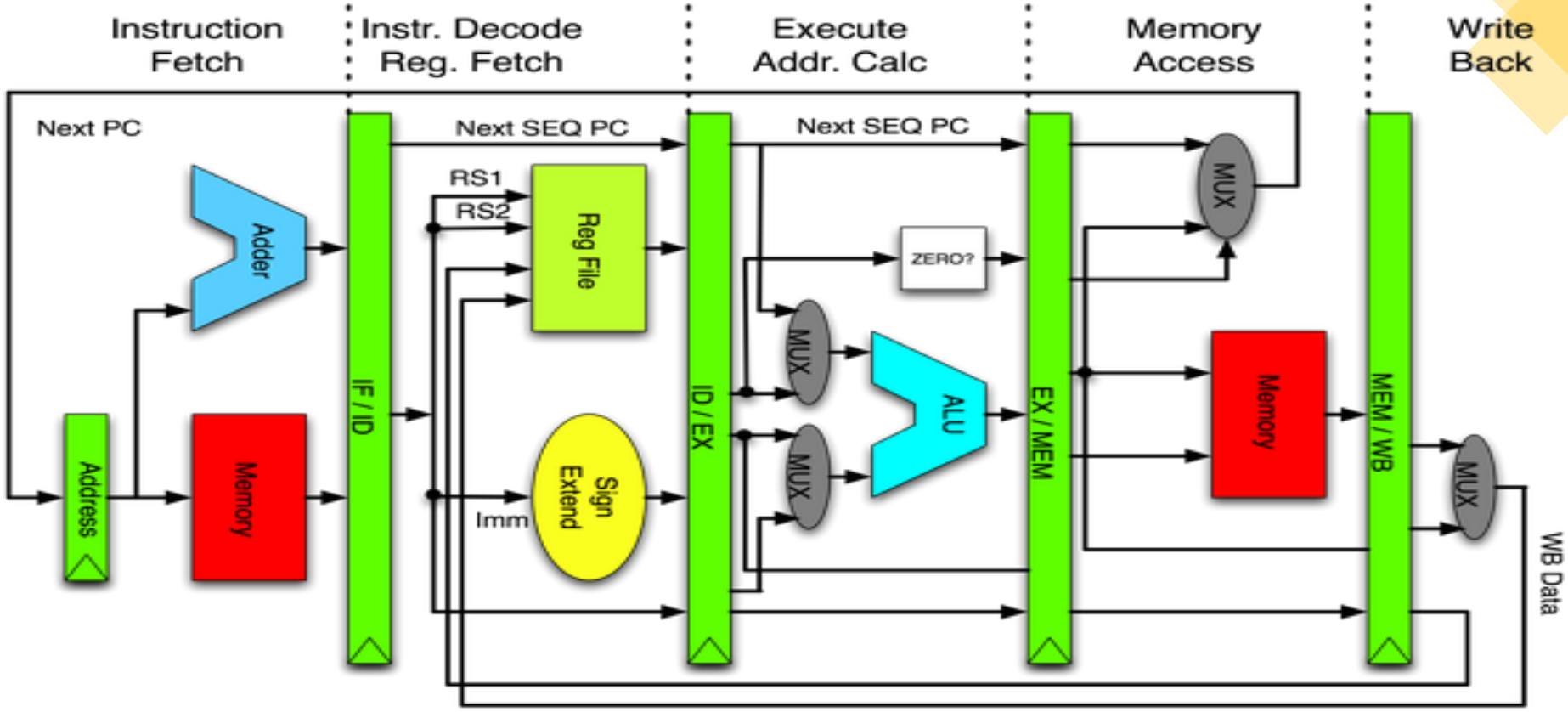


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 2: I1 in Decode stage, I2 in fetch stage

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I2: Move R1, #15

I1: Move R5, #2000h

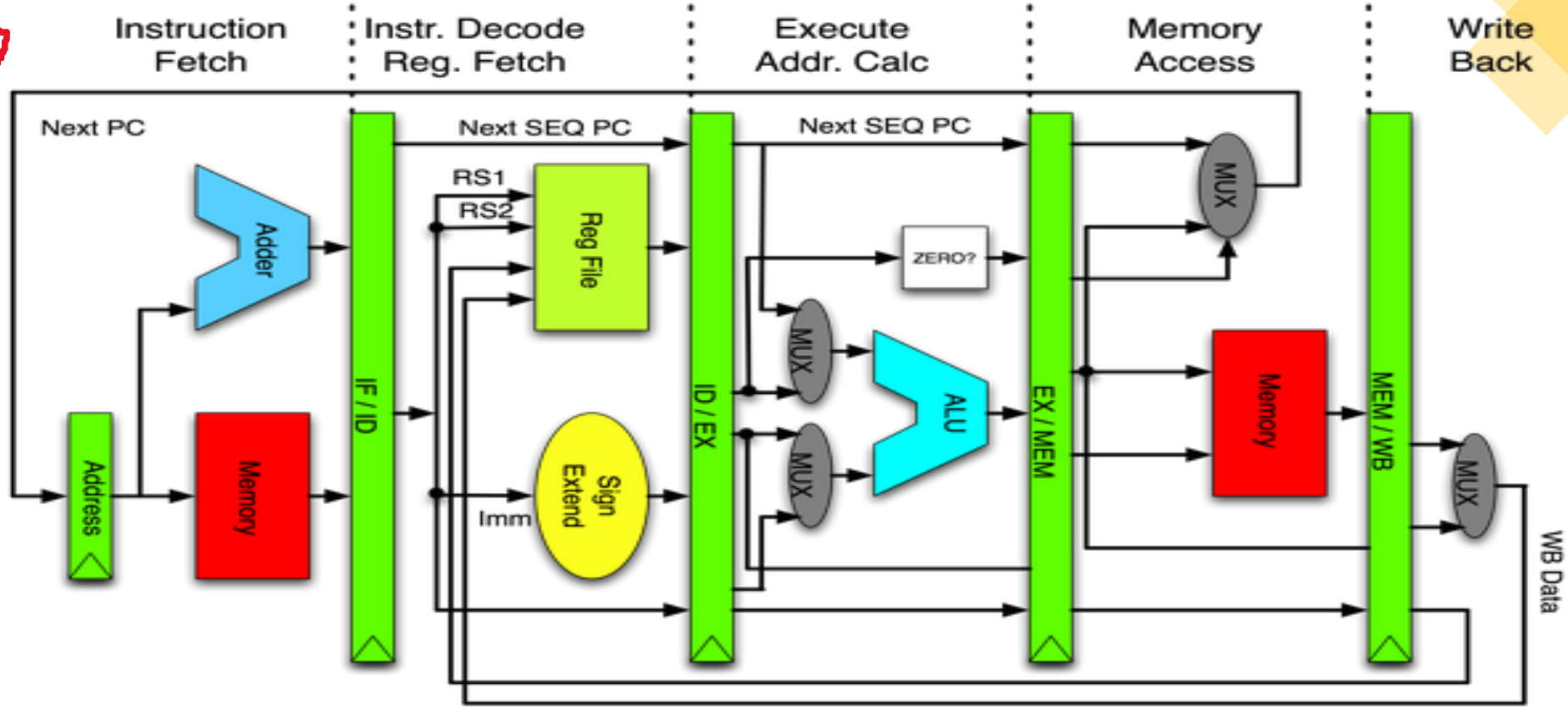


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 3: I1 in Execute stage, I2 in Decode stage, I3 in fetch stage

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

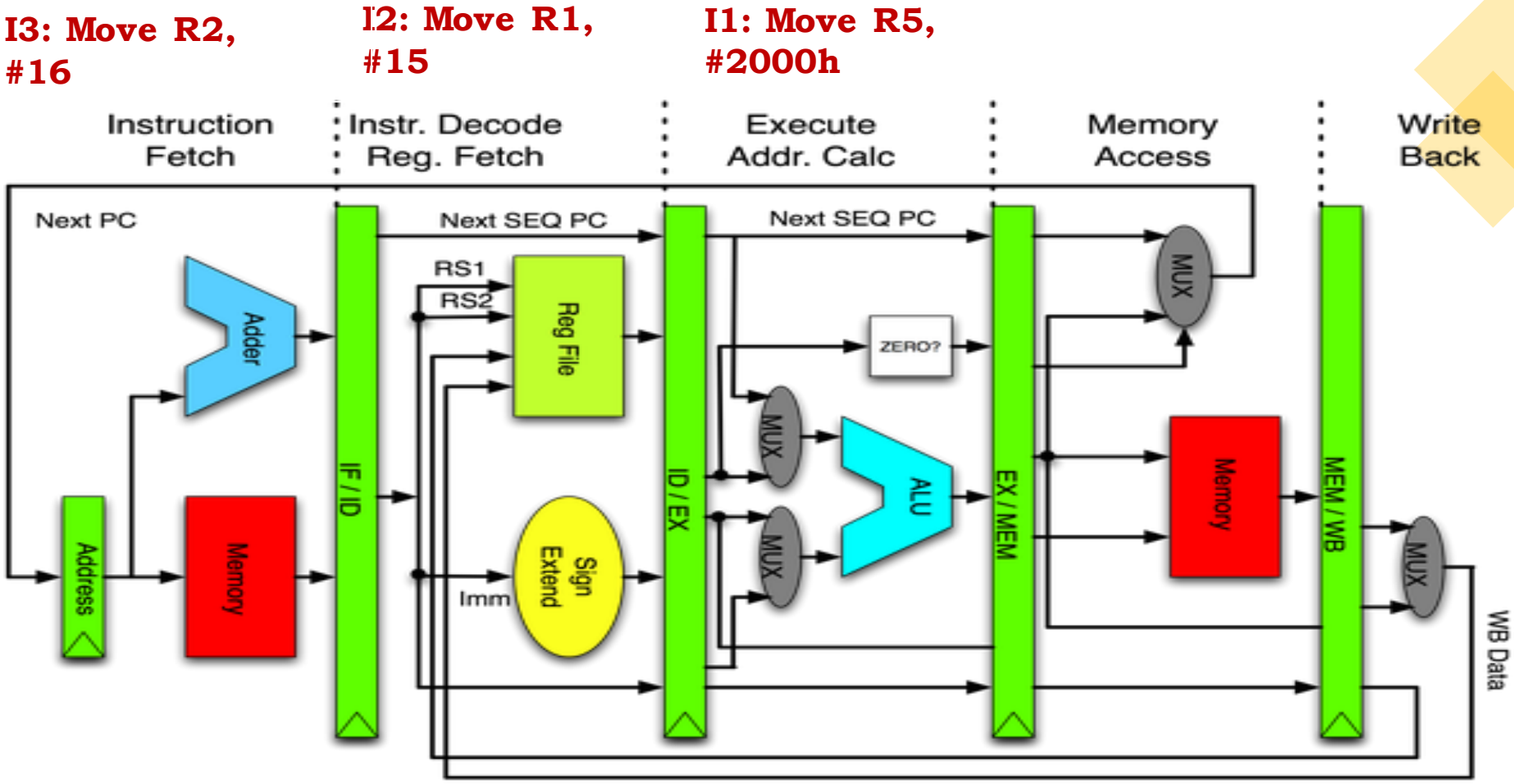


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 4: I1 in memory access stage, I2 in execute , I3 in Decode and I4 in fetch stage

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

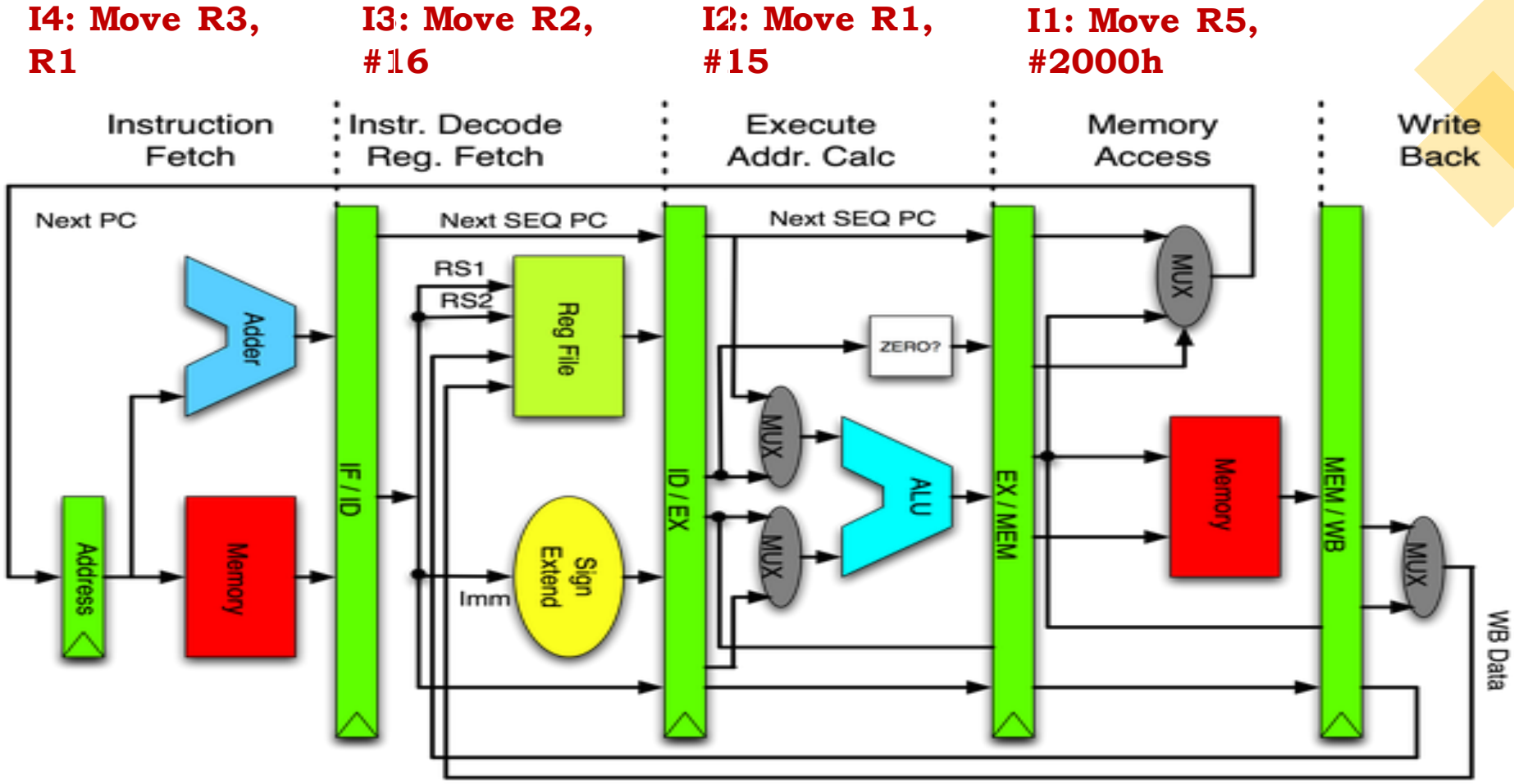


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 5: R1 data must be Written from I1 to Register file ; R1 Data must be read by Instruction 4.

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I5: Add R1, R3 | **I4: Move R3, R1** | **I3: Move R2, #16** | **I2: Move R1, #15** | **I5: Move R5, #2000h**

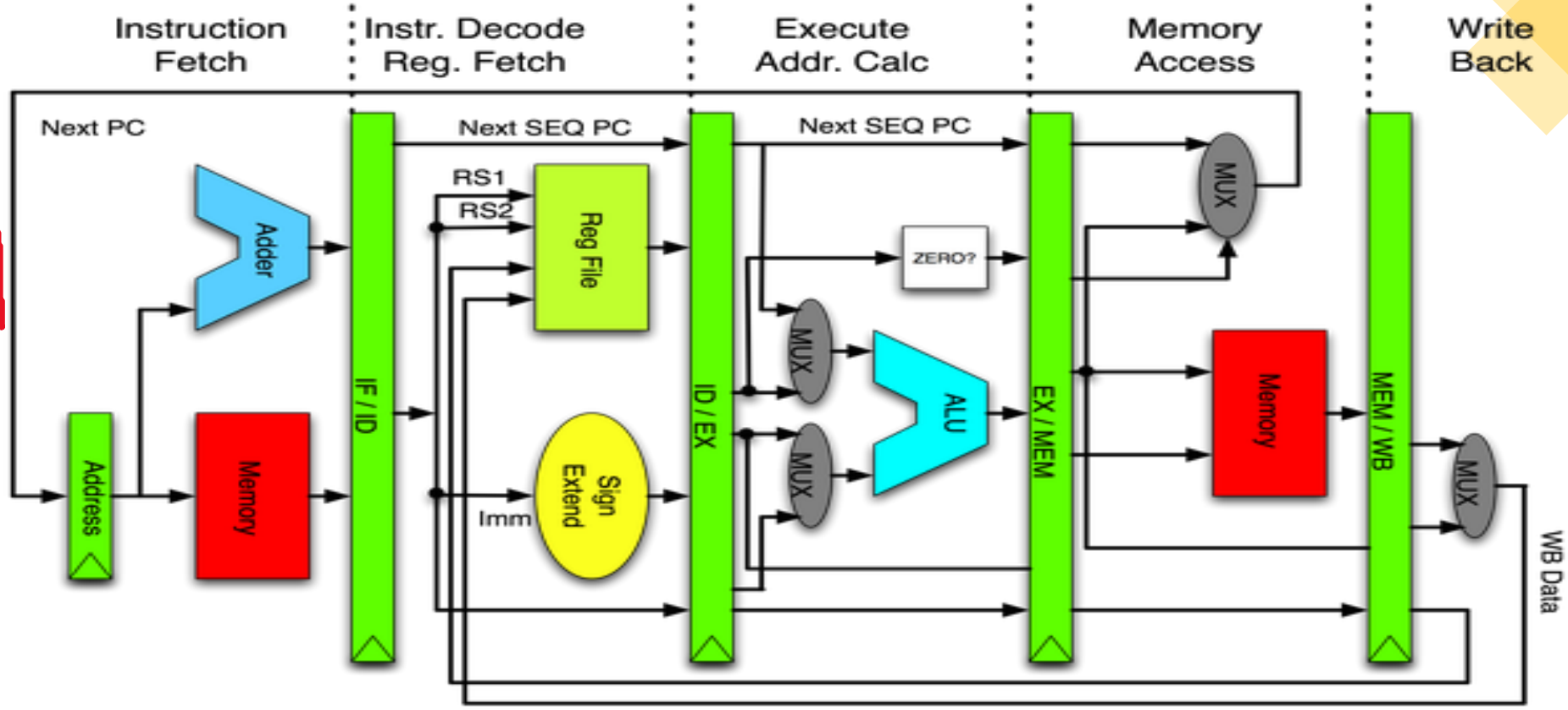


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 5: R1 data must be Written from I1 to Register file ; R1 Data must be read by Instruction 4.

Data Dependency Issue at Decode Stage: Solve by Read register content after write operation.

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I5: Add R1, R3 | **I4: Move R3, R1** | **I3: Move R2, #16** | **I2: Move R1, #15** | **I1: Move R5, #2000h**

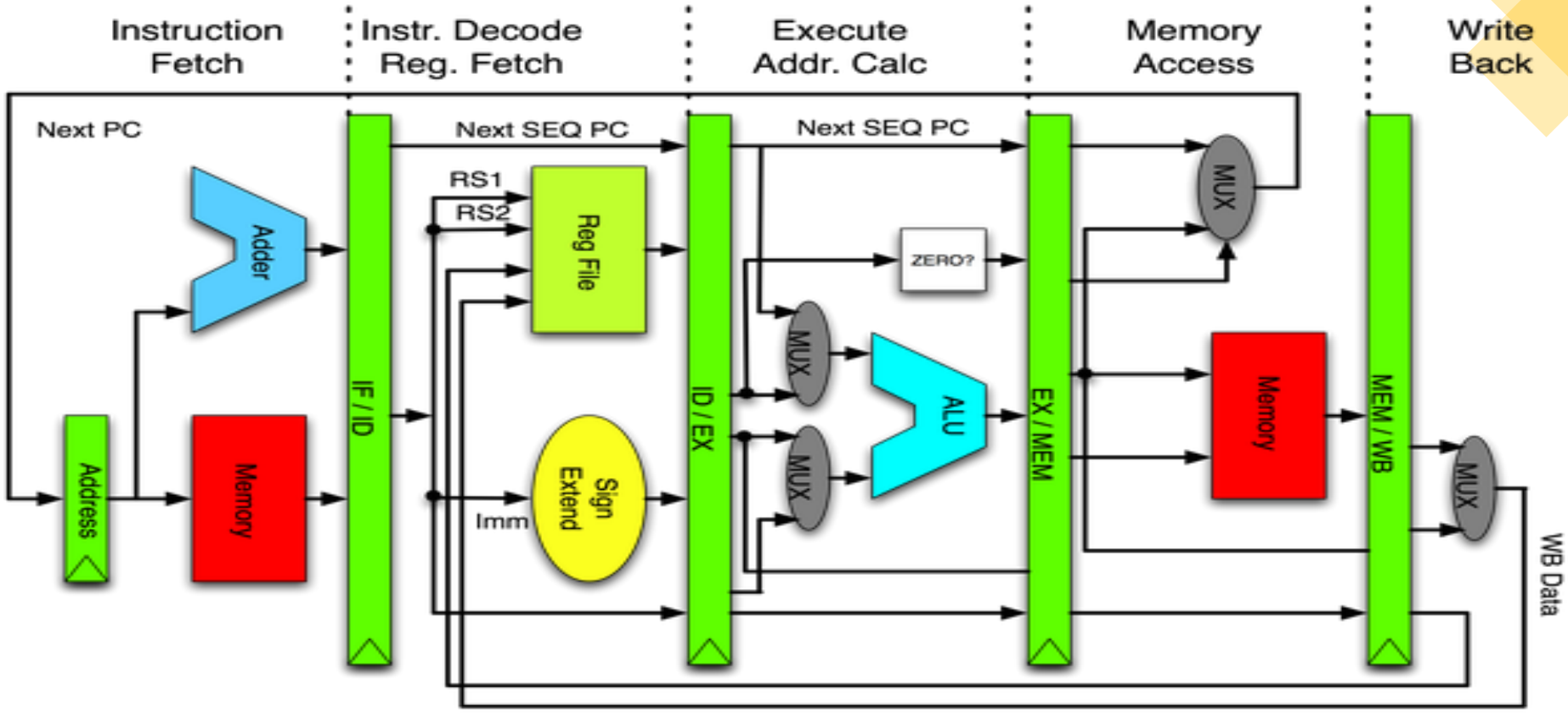


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 6: R3 data must be Written from I4 to Register file ; R3 Data must be read by Instruction 5.

Data Dependency Issue at Decode Stage: Since Data is not ready, R3 old value is read

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I6:Move R4, R2 | **I5: Add R1, R3** | **I4: Move R3, R1** | **I3: Move R2, #16** | **I2: Move R1, #15**

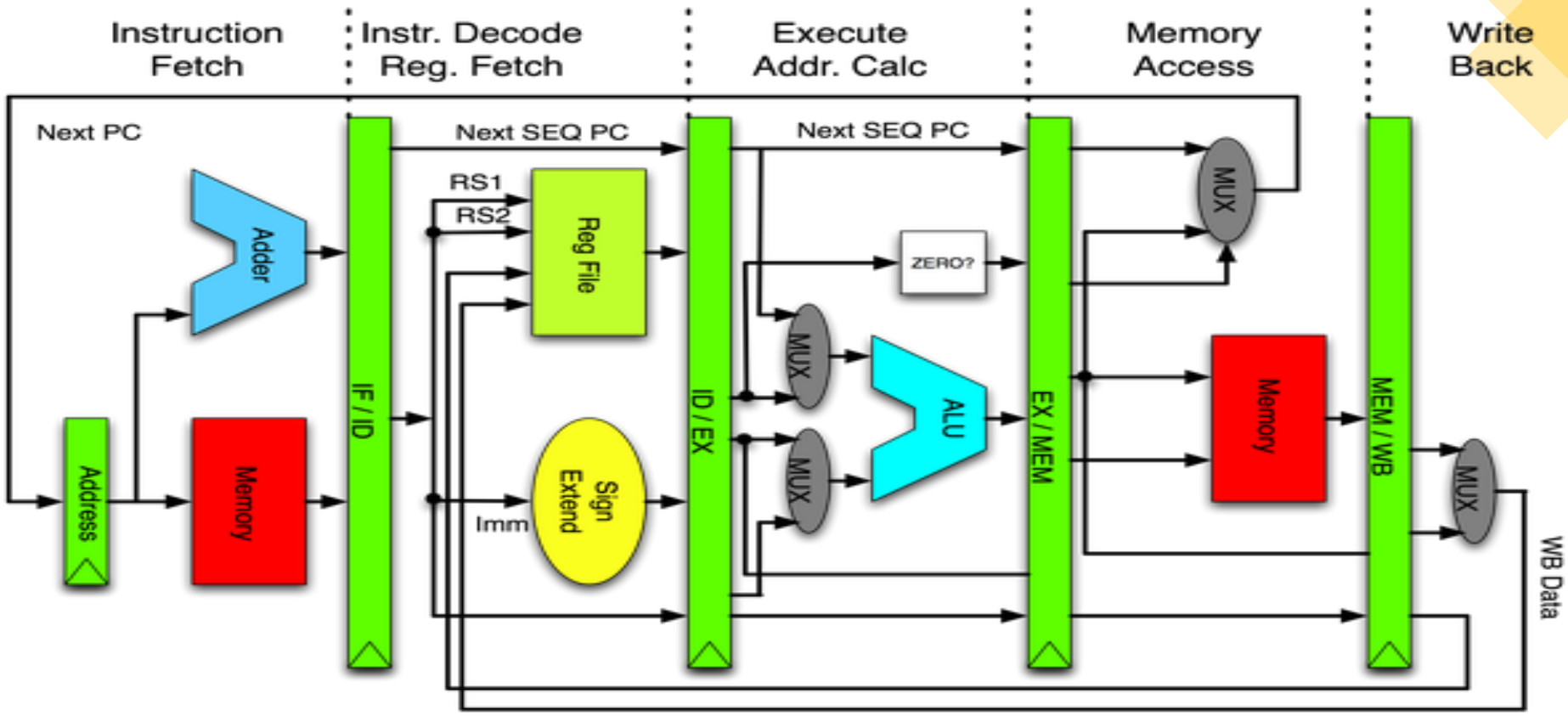


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 7: R3 data not ready due to instruction dependency

Data Dependency Issue at Execution stage : Stall the pipeline

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I7: Add R2, R4 | **I6:Move R4, R2** | **I5: Add R1, R3** | **I4: Move R3, R1** | **I3: Move R2, #16**

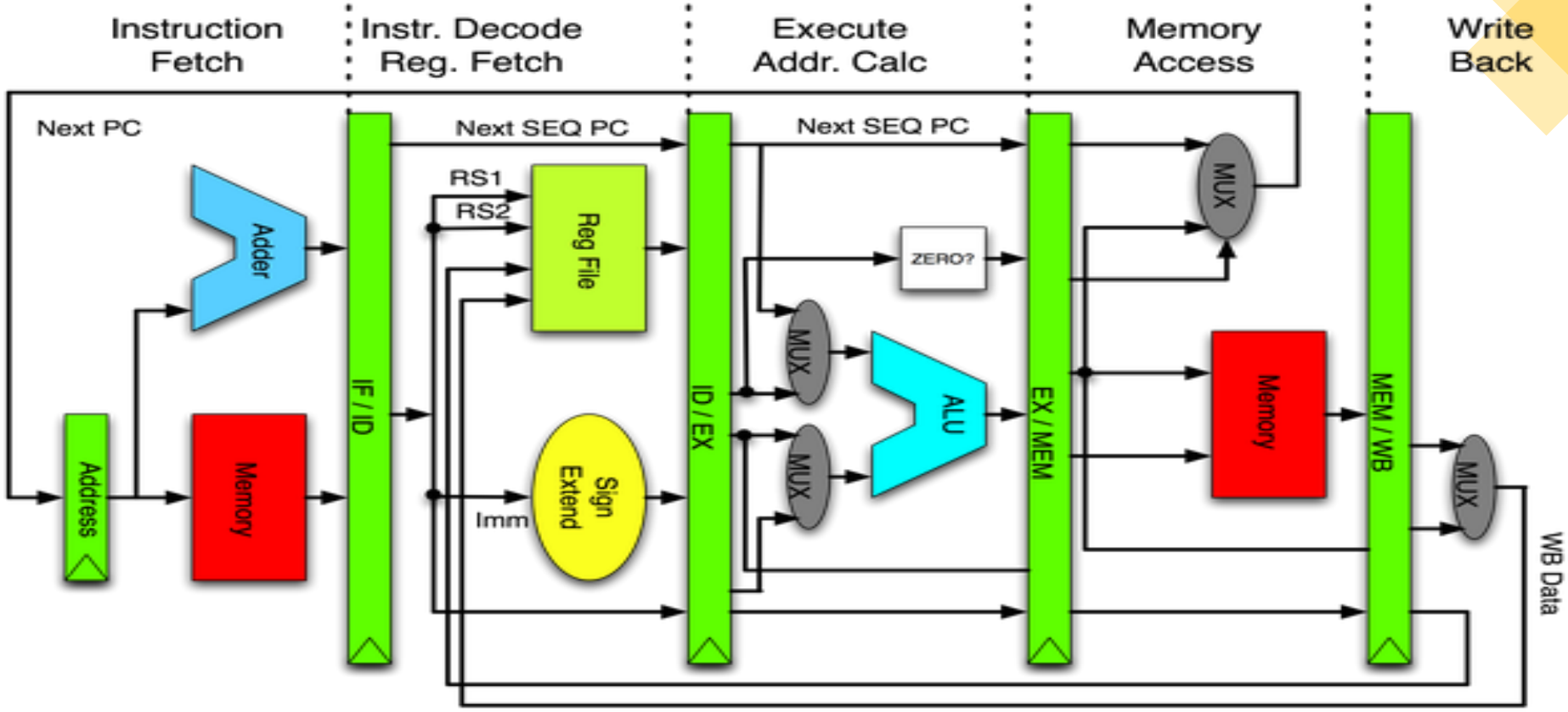


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 8: Pipeline stall: no new fetch instruction. Only write back happens

Data Dependency Issue at Execution stage : Stall the pipeline

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

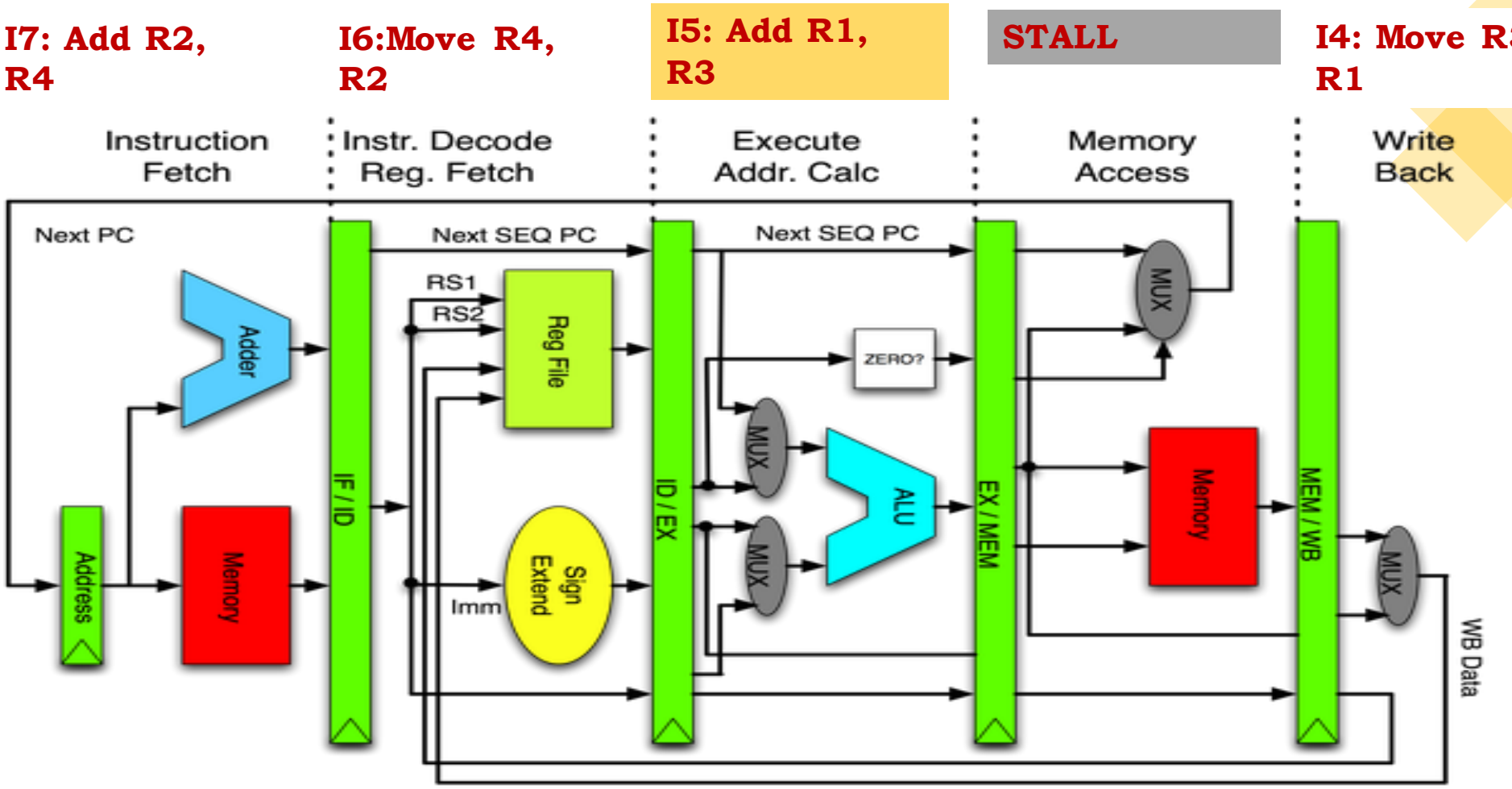


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 8: We need data of R3 at execution stage , Bu I4 is performing write back to register now.
Use the concept of Data Forwarding

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I7: Add R2, R4 | **I6:Move R4, R2** | **I5: Add R1, R3** | **STALL** | **I4: Move R3, R1**

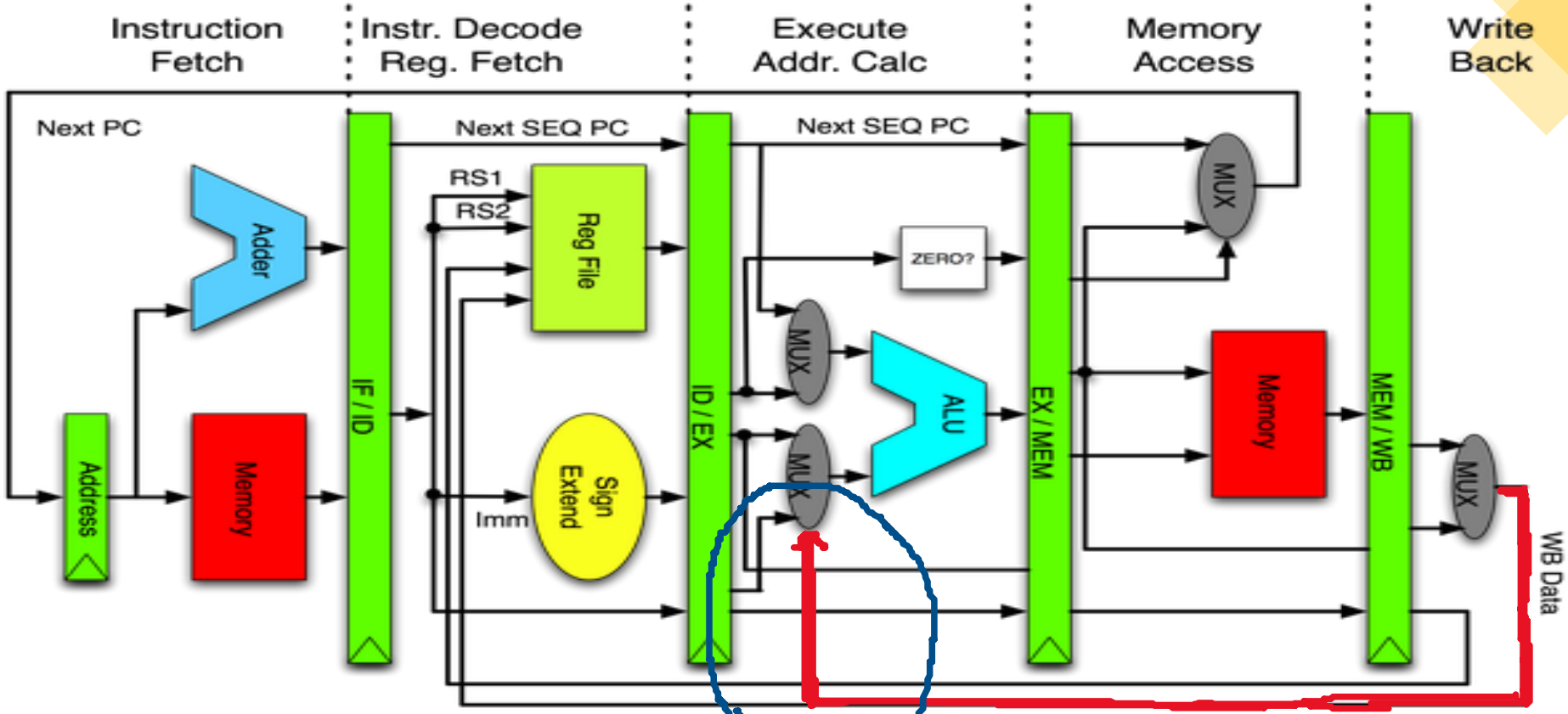


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 9: Nothing in Writeback stage

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I8: Add R1, R2

I7: Add R2, R4

I6:Move R4, R2

I5: Add R1, R3

STALL

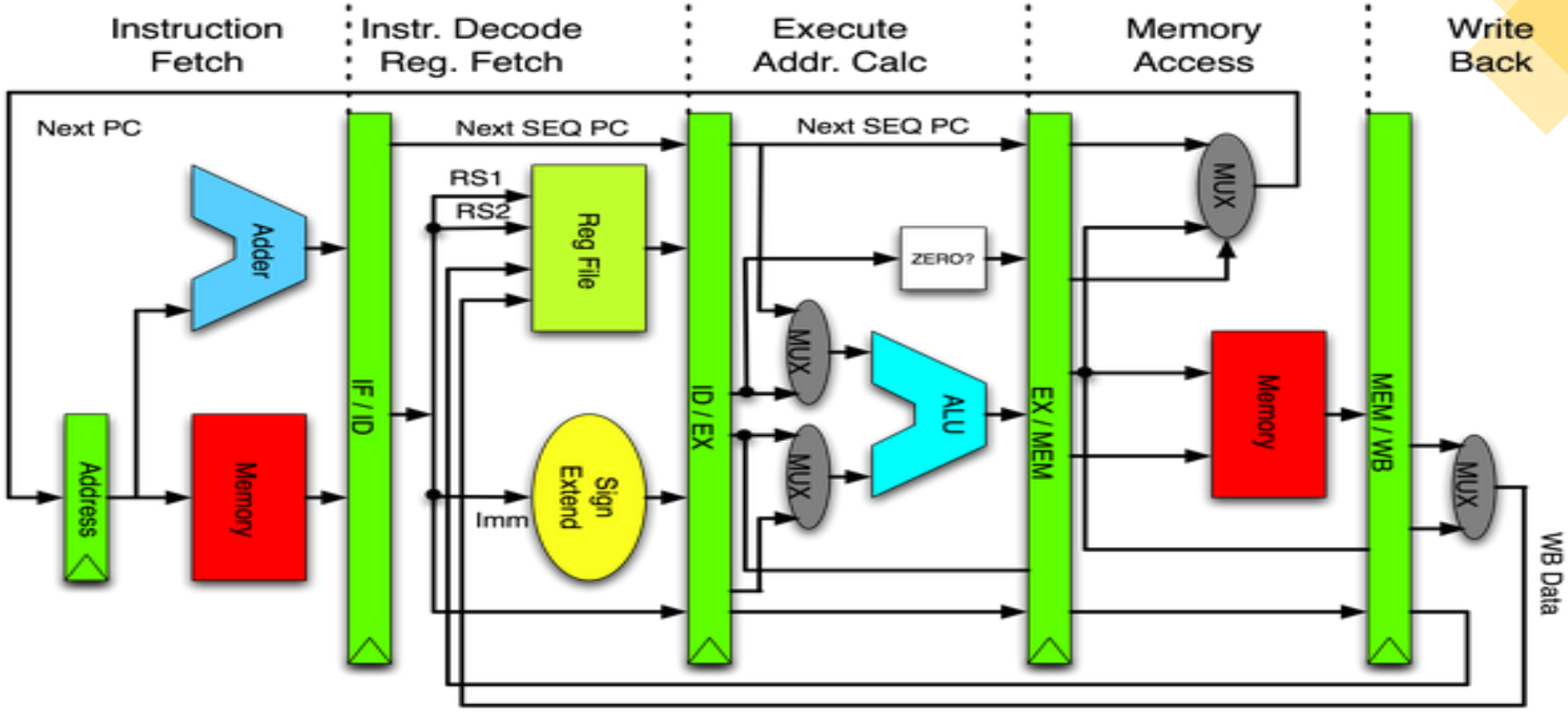


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 10: Data dependency at Execution stage; Wait

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I9: Store [R5],R1 **I8: Add R1, R2** **I7: Add R2, R4** **I6:Move R4, R2** **I5: Add R1, R3**

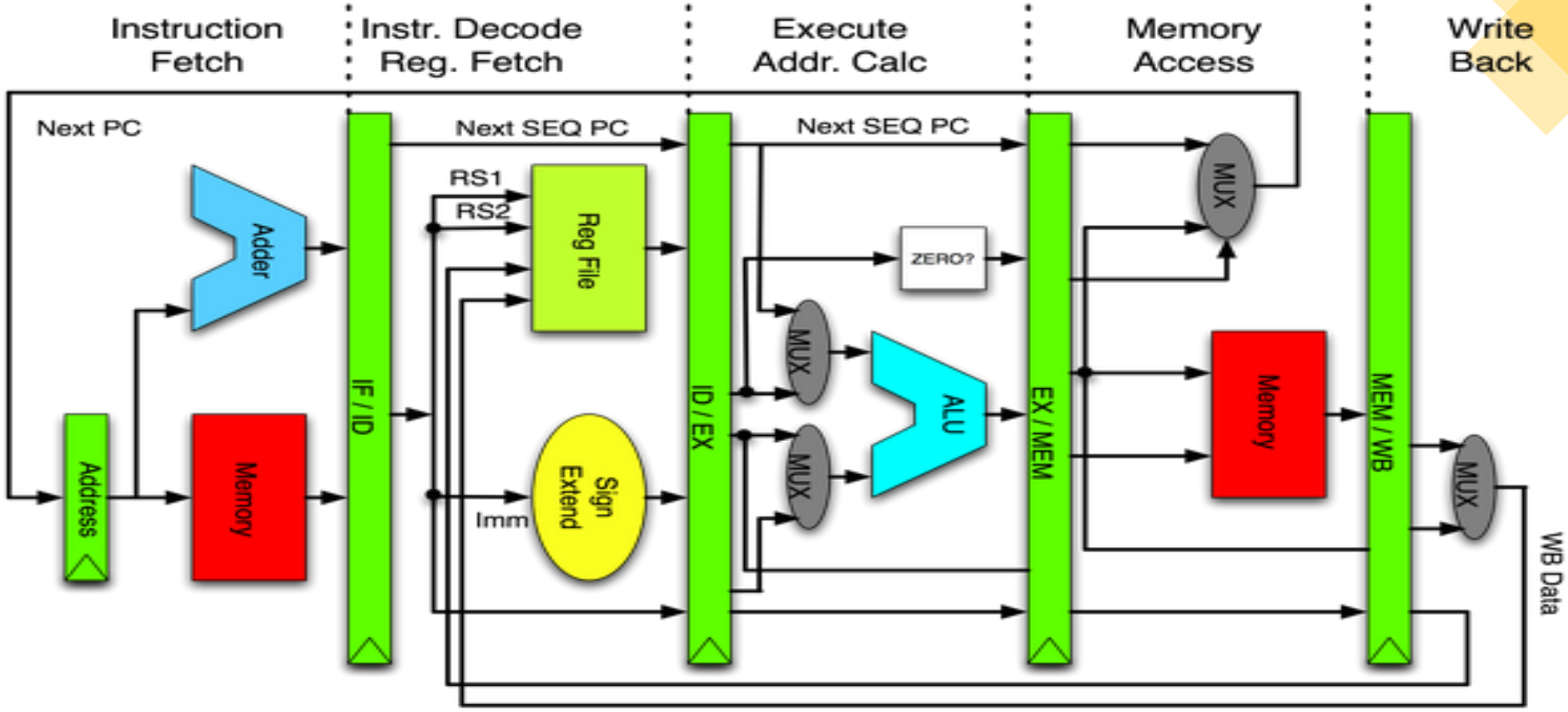


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 11: Data dependency at Execution stage; Data forwarding from writeback stage

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

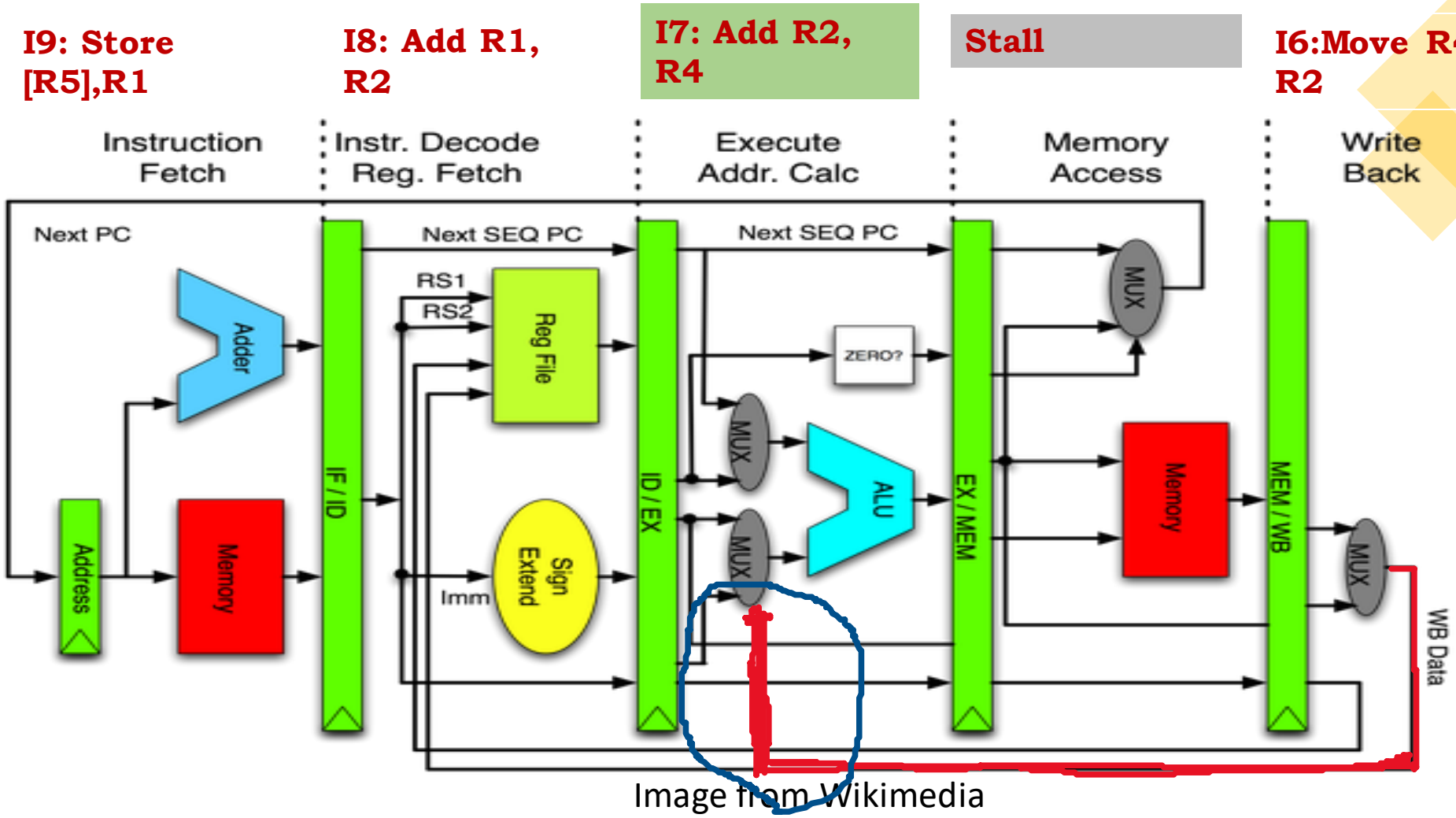


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 12: Data dependency at Execution stage; Wait

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I9: Store [R5], R1. | **I8: Add R1, R2** | **I7: Add R2, R4** | **Stall**

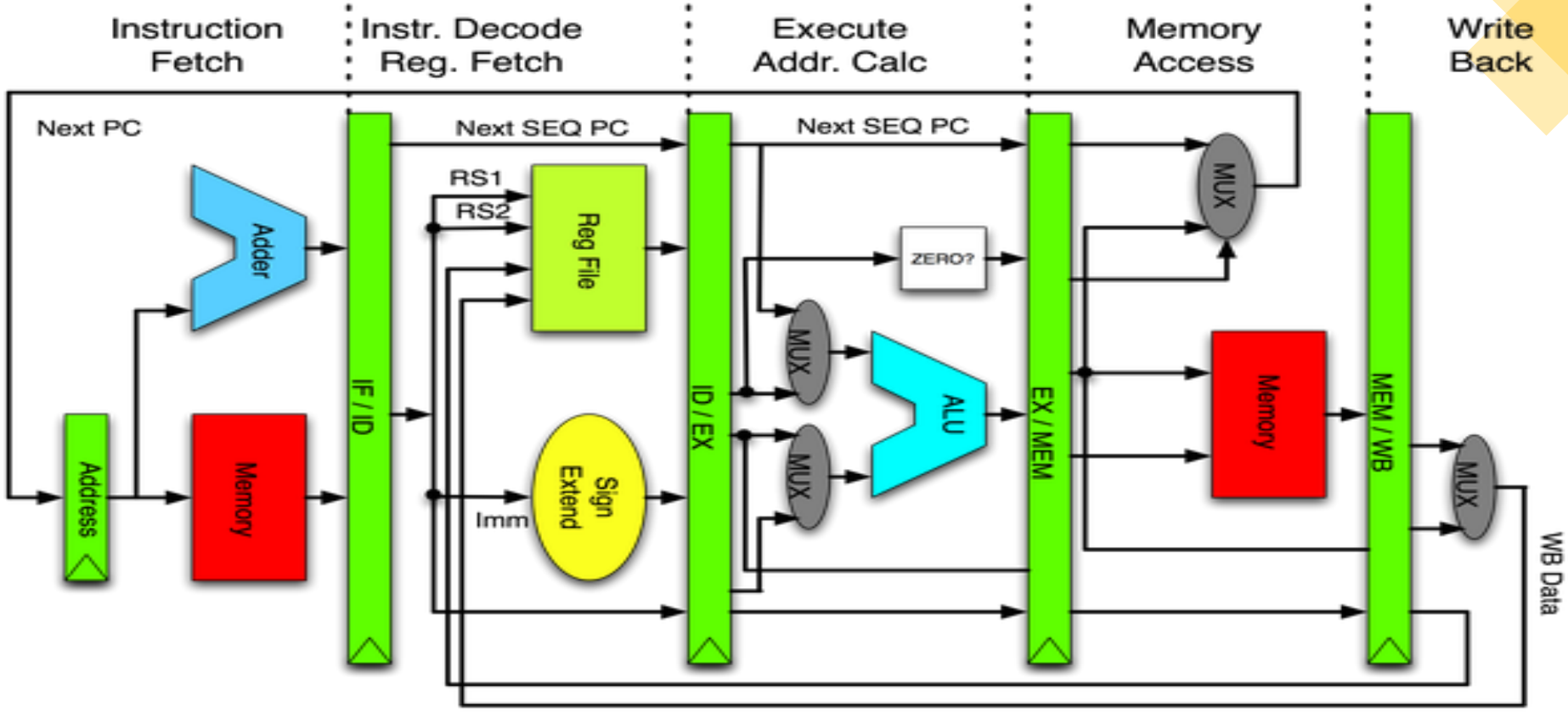


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 13: Data dependency at Execution stage; Data Forwarding

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I9: Store [R5], R1. | **I8: Add R1, R2** | **STALL** | **I7: Add R2, R4**

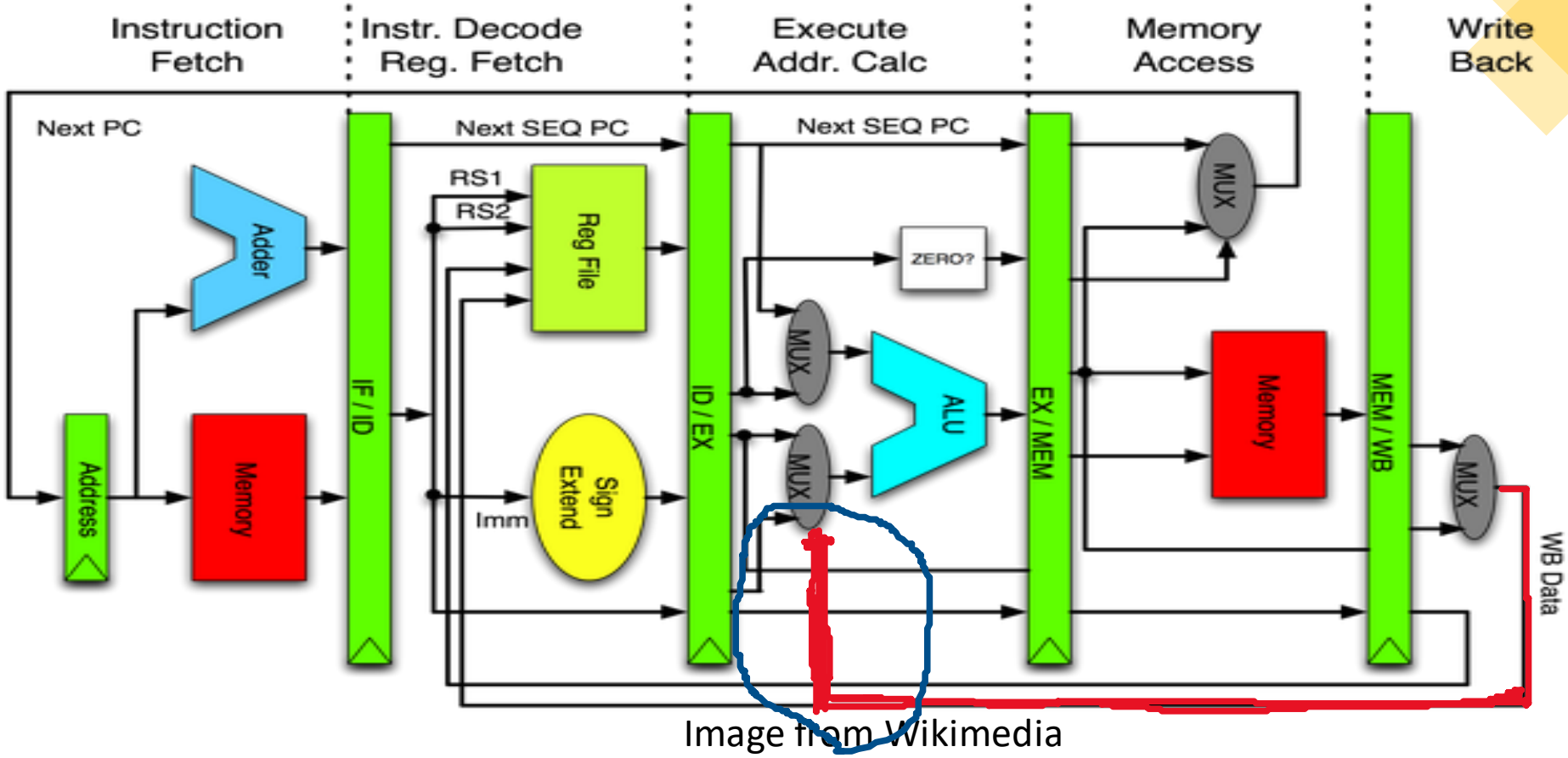


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 14: Data dependency at Execution stage; Stall Pipeline

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I9: Store [R5], R1.

I8: Add R1, R2

STALL

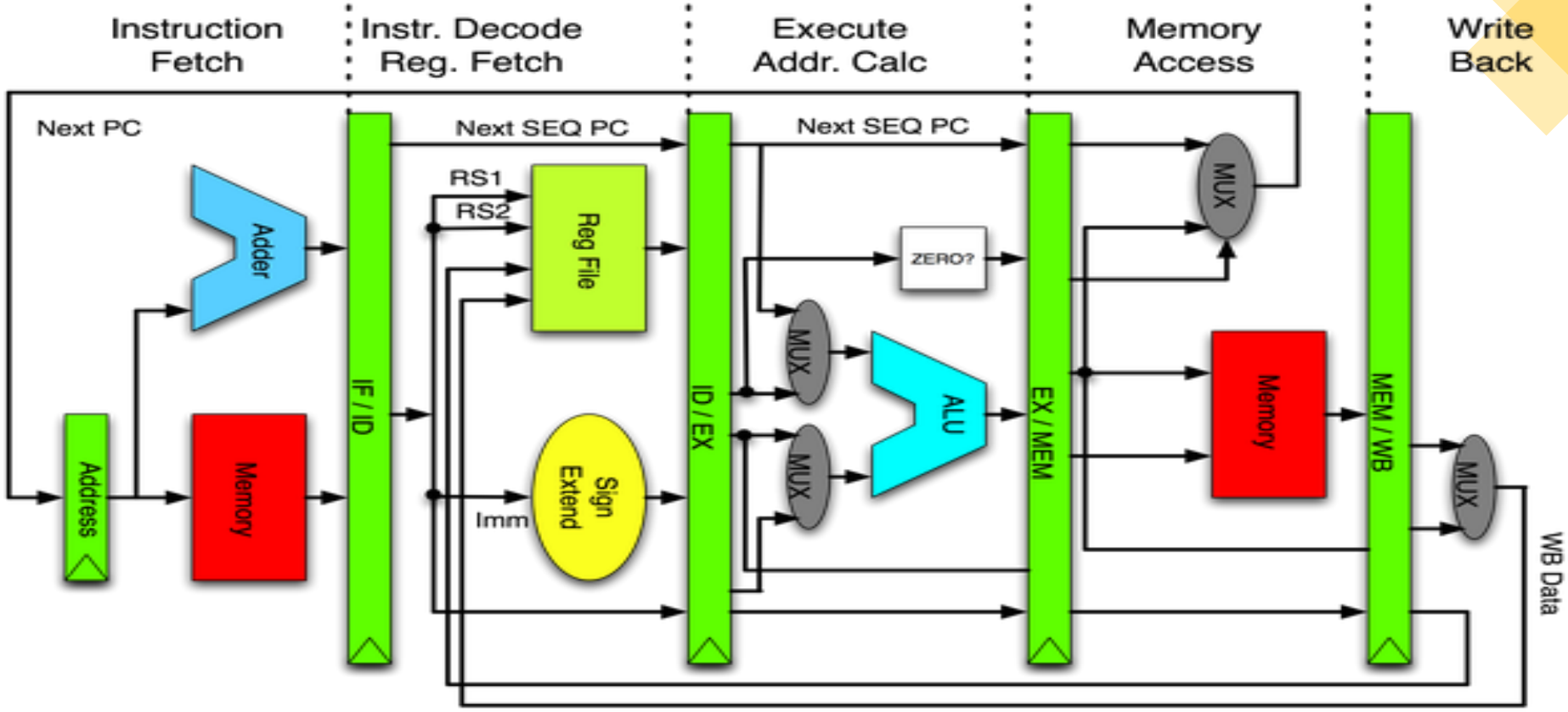


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 15: Data Dependency at Memory Write Stage : Use Data Forwarding ·

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

I9: Store [R5], R1.

I8: Add R1, R2

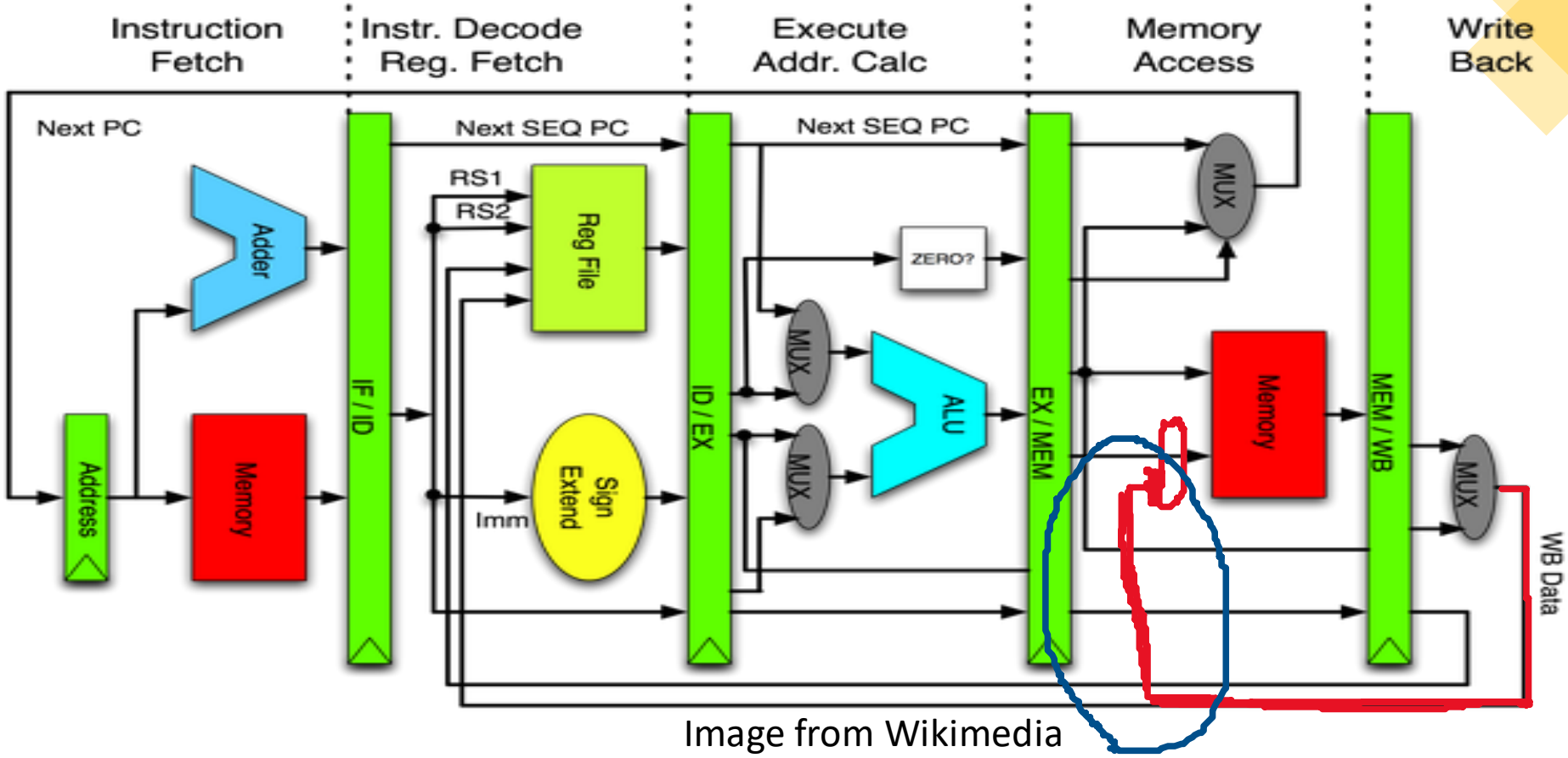


Image from Wikimedia

3: Issues of pipeline and Design Solutions

Cycle 16: Execution of Program Complete

Total Number of cycles taken = 16

I9: Store [R5], R1.

I_1	Move R5, #2000h
I_2	Move R1, #15
I_3	Move R2, #16
I_4	Move R3, R1
I_5	Add R1, R3
I_6	Move R4, R2
I_7	Add R2, R4
I_8	Add R1, R2
I_9	Store [R5], R1

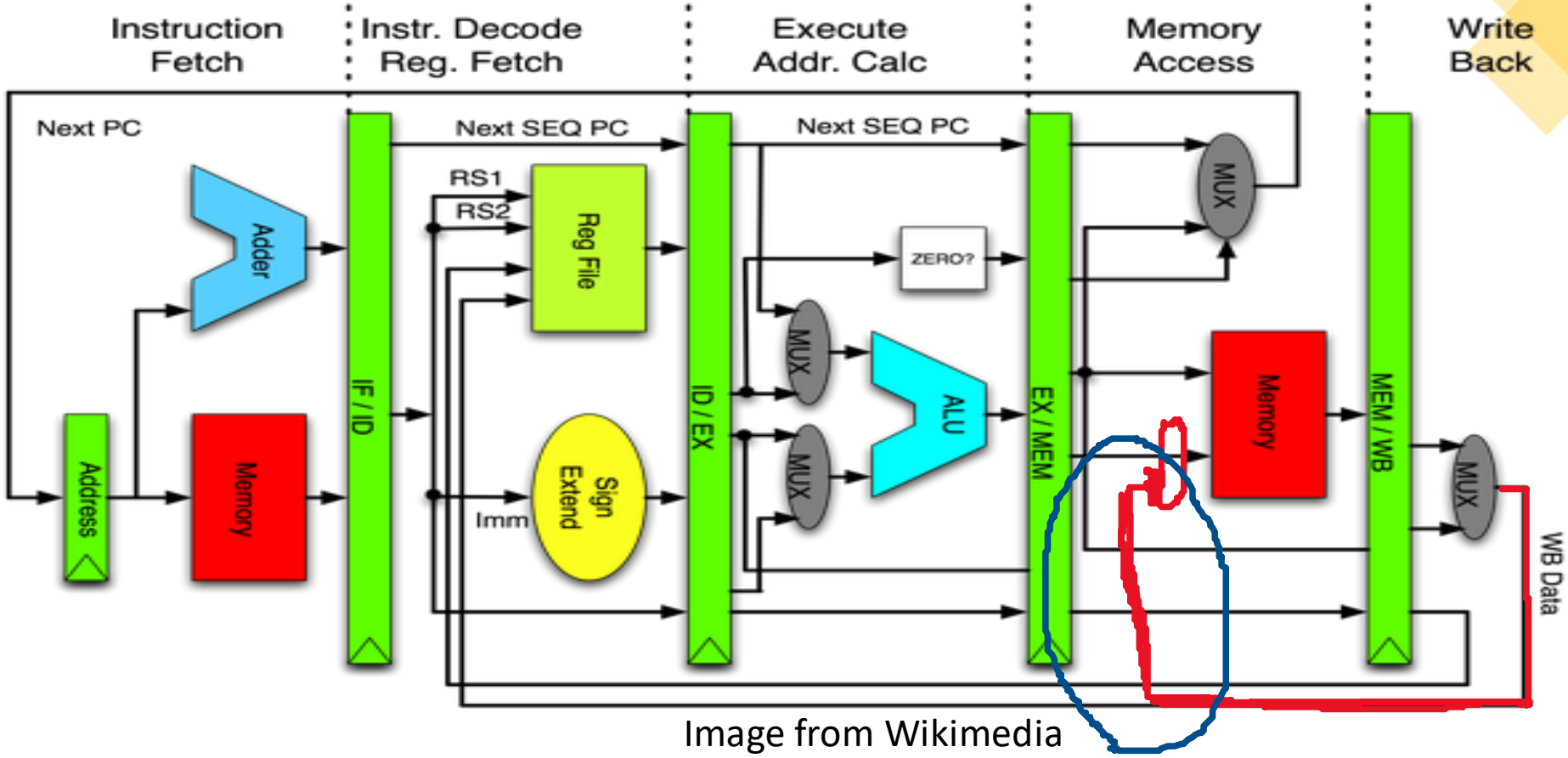


Image from Wikimedia

2. Five Stage Pipeline Design

	Instruction	(F)	(D)	(E)	(M)	(W)	
I_1	Move R5, #2000h	I_1					
I_2	Move R1, #15	I_2	I_1				
I_3	Move R2, #16	I_3	I_2	I_1			
I_4	Move R3, R1	I_4	I_3	I_2	I_1		
I_5	Add R1, R3	I_5	I_4	I_3	I_2	I_1	I1: Complete
I_6	Move R4, R2	I_6	I_5	I_4	I_3	I_2	I2: Complete
I_7	Add R2, R4	I_7	I_6	I_5	I_4	I_3	I3: Complete
I_8	Add R1, R2	I_8	I_7	I_6	I_5	I_4	I4: Complete
I_9	Store [R5], R1	I_9	I_8	I_7	I_6	I_5	I5: Complete
I_{10}			I_9	I_8	I_7	I_6	I6: Complete
I_{11}				I_9	I_8	I_7	I7: Complete
I_{12}					I_9	I_8	I8: Complete
I_{13}						I_9	I9: Complete

- If no pipeline, then this program takes

  9 x 5 = 45 cycles

- If 5 stage pipeline is used, then it takes

  5 + 8 = 13 cycles

(first instruction takes 'n' stages/cycle) + (n-1) instructions

- Actual Cycles taken due to **data dependency = 16 Cycles.**

Reference

Textbooks and/or Reference Books:

1. Professional CUDA C Programming – John Cheng, Max Grossman, Ty McKercher, 2014
2. Wilkinson, M. Allen, ”Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers”, Pearson Education, 1999
3. I. Foster, ”Designing and building parallel programs”, 2003
4. Parallel Programming in C using OpenMP and MPI – Micheal J Quinn, 2004
5. Introduction to Parallel Programming – Peter S Pacheco, Morgan Kaufmann Publishers, 2011
6. Advanced Computer Architectures: A design approach, Dezso Sima, Terence Fountain, Peter Kacsuk, 2002
7. Parallel Computer Architecture : A hardware/Software Approach, David E Culler, Jaswinder Pal Singh Anoop Gupta, 2011 8. Introduction to Parallel Computing, Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, Pearson, 2011

Thank You