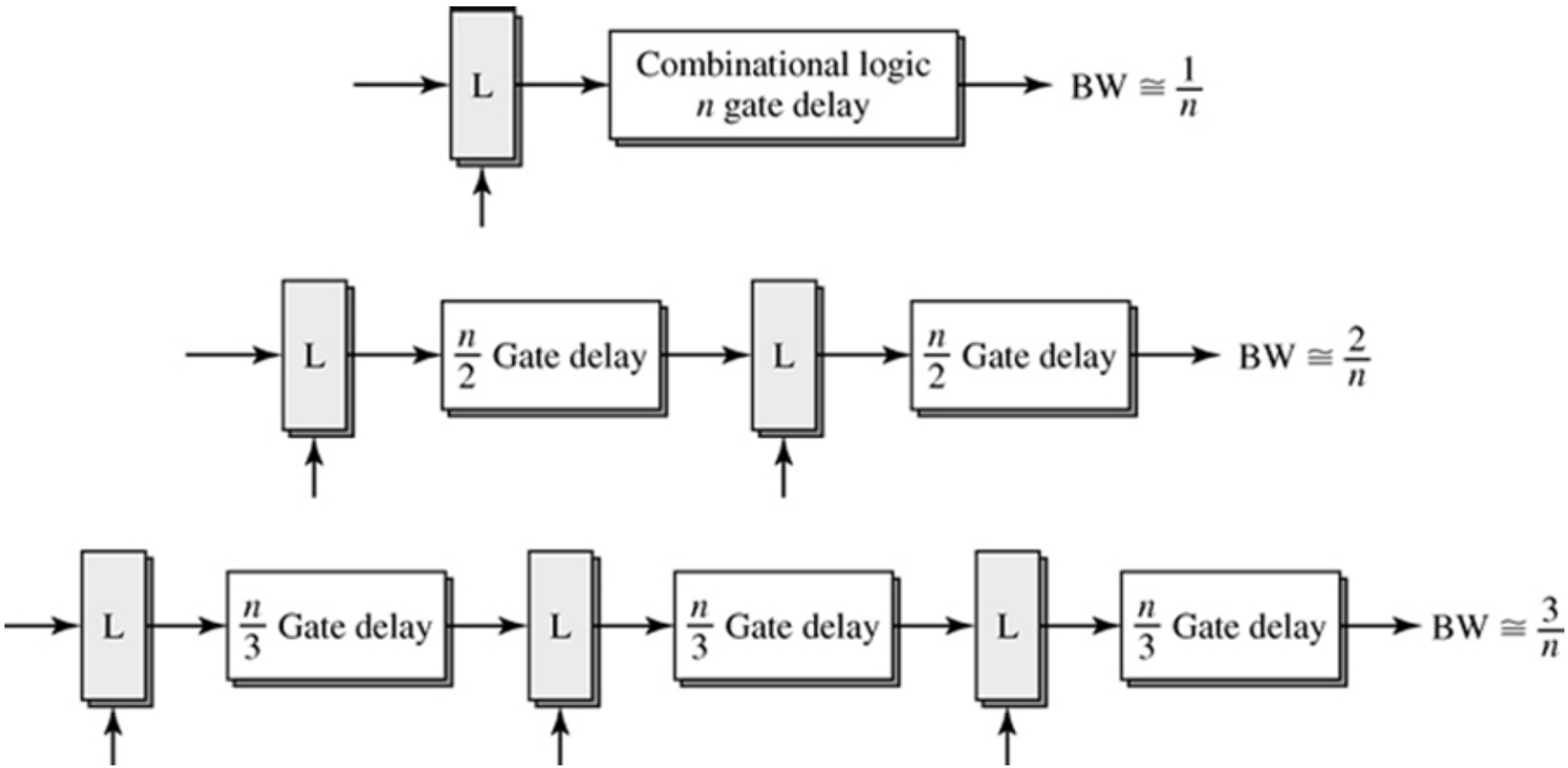

Figure 2.1: Potential *k*-Fold Increase of Throughput in a *k*-Stage Pipelined System.

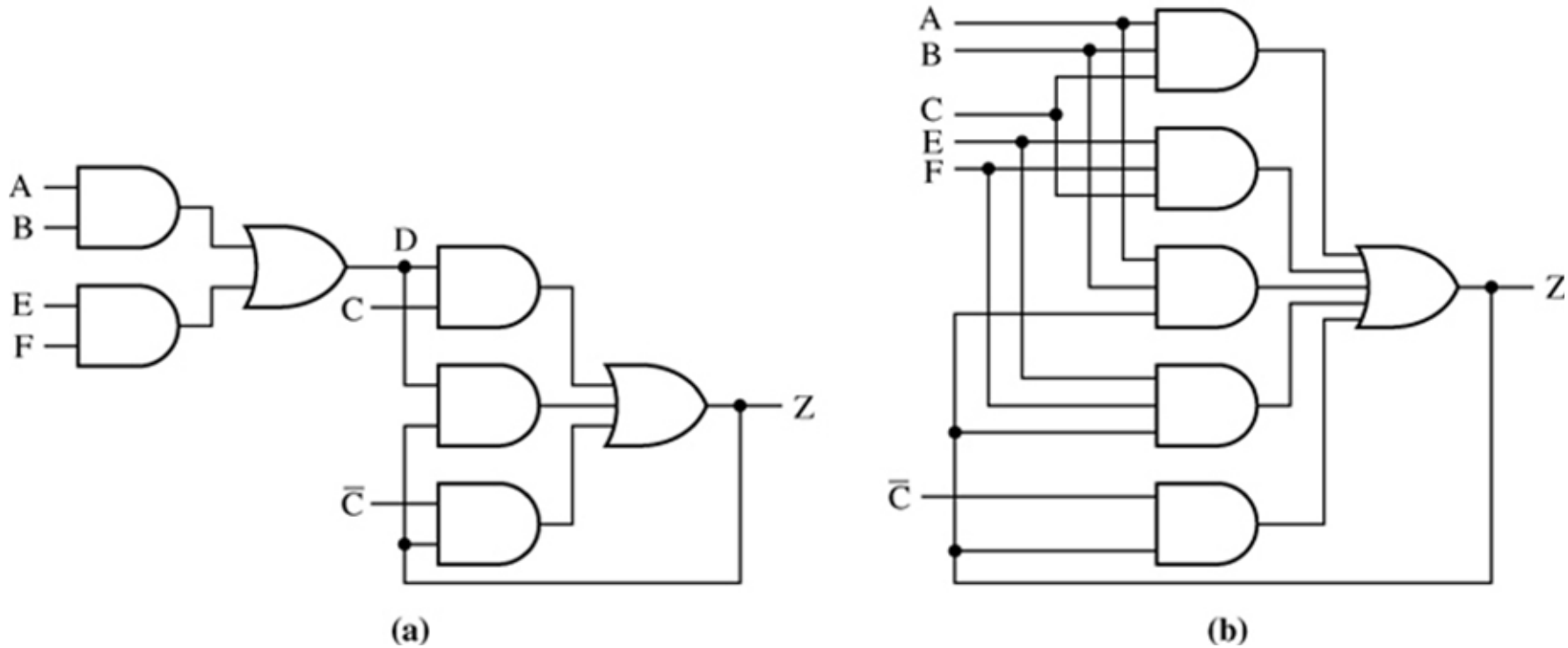

Figure 2.2: The Earle Latch and Its Incorporation into Logic Without Incurring Additional Gate Delay: (a) Earle Latch Following the Combinational Logic; (b) Earle Latch Integrated with the Combinational Logic.

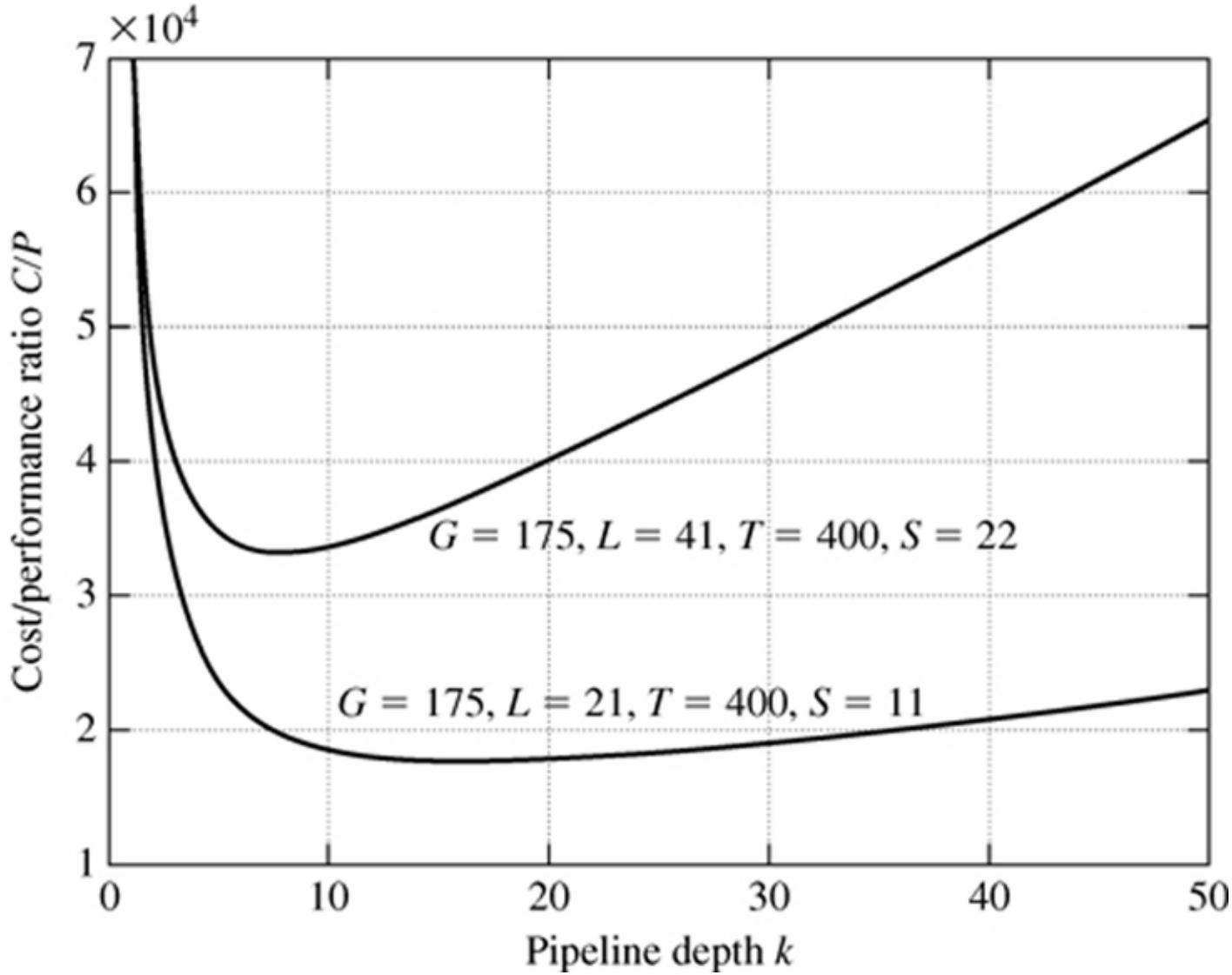

Figure 2.3: Cost/Performance Tradeoff Model for Pipelined Designs.

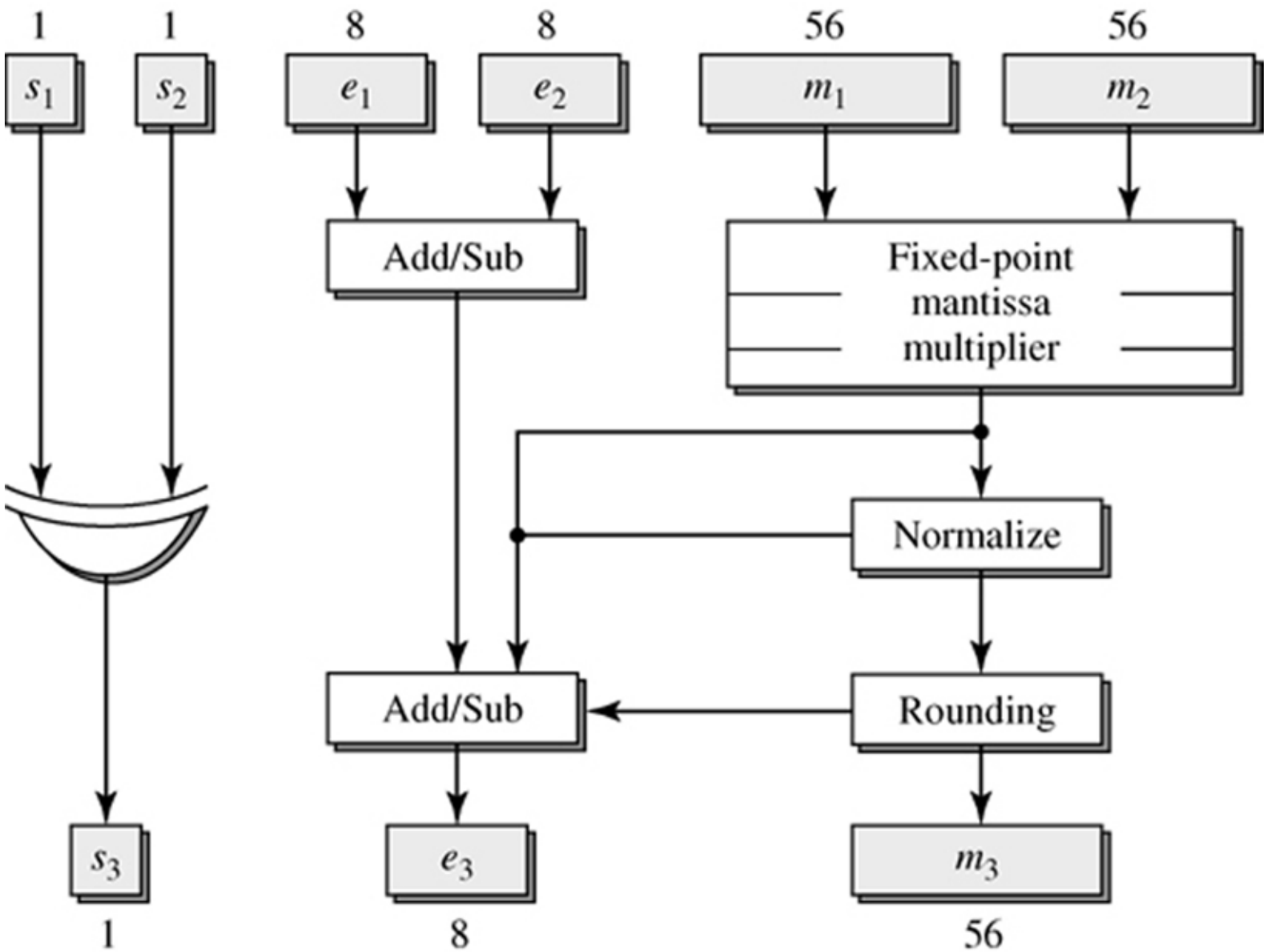

Figure 2.4: A Nonpipelined Floating-Point Multiplier. Waser and Flynn, 1982.

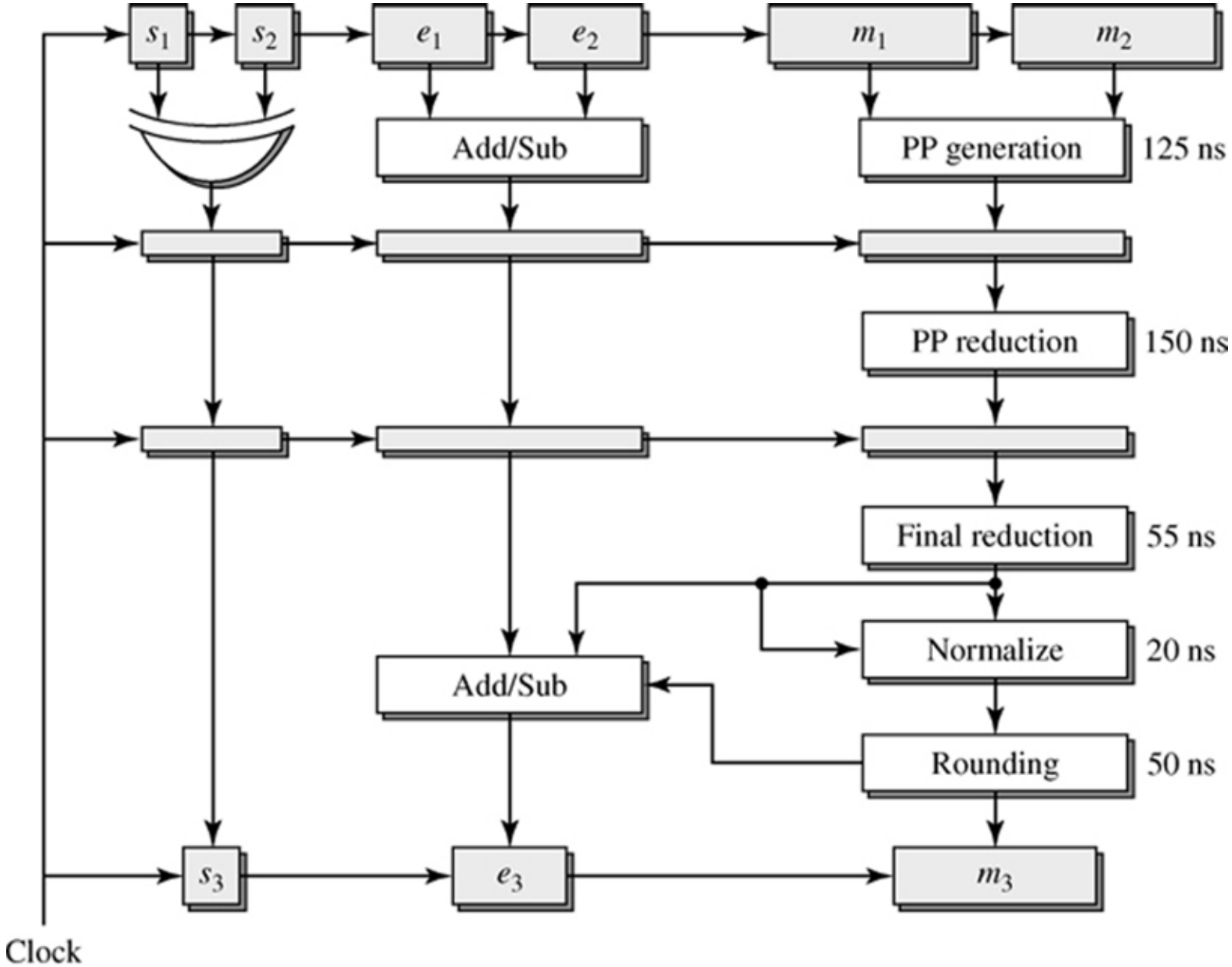

Figure 2.5: A Pipelined Floating-Point Multiplier.
Source: Waser and Flynn, 1982.

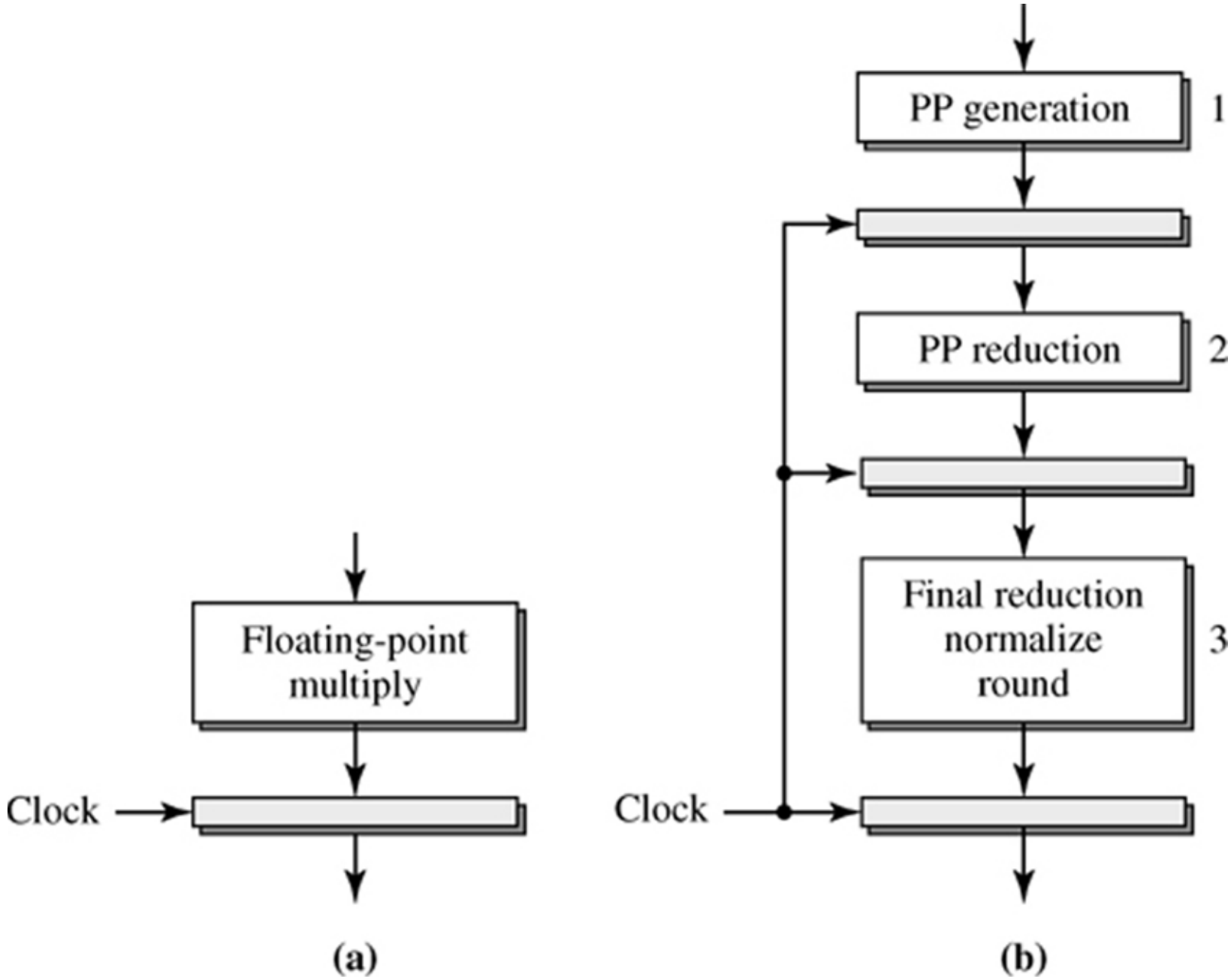

Figure 2.6: A Simple Illustration of Instruction Cycle vs. Machine Cycle.

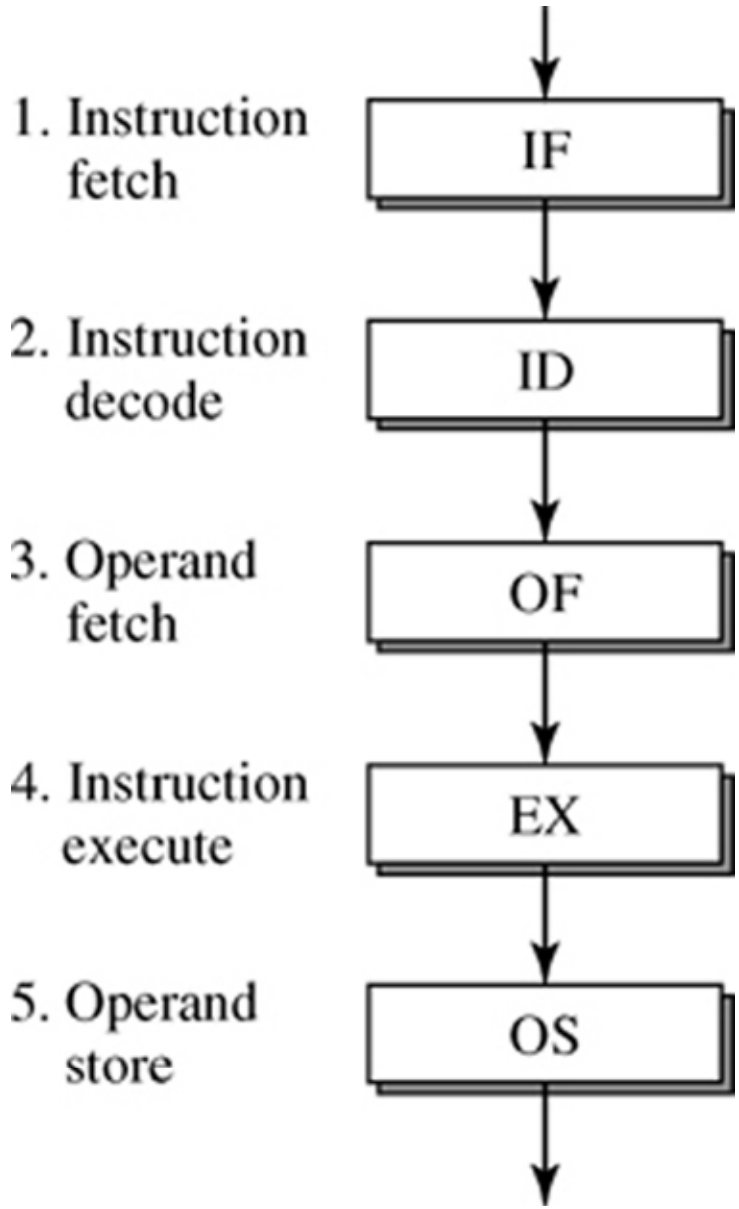

Figure 2.7: The Five-Stage GENERIC (GNR) Instruction Pipeline.

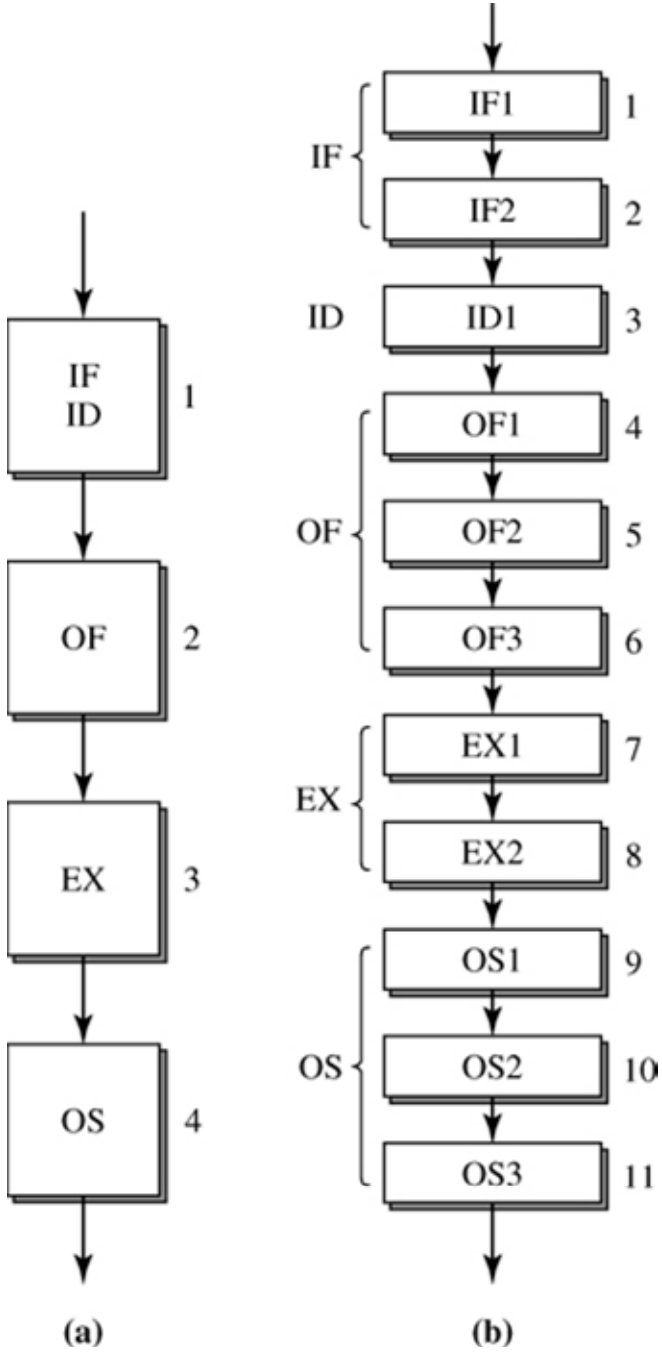

Figure 2.8: (a) A Four-Stage Instruciton Pipeline Example. (b) An 11-Stage Instruction Pipeline Example.

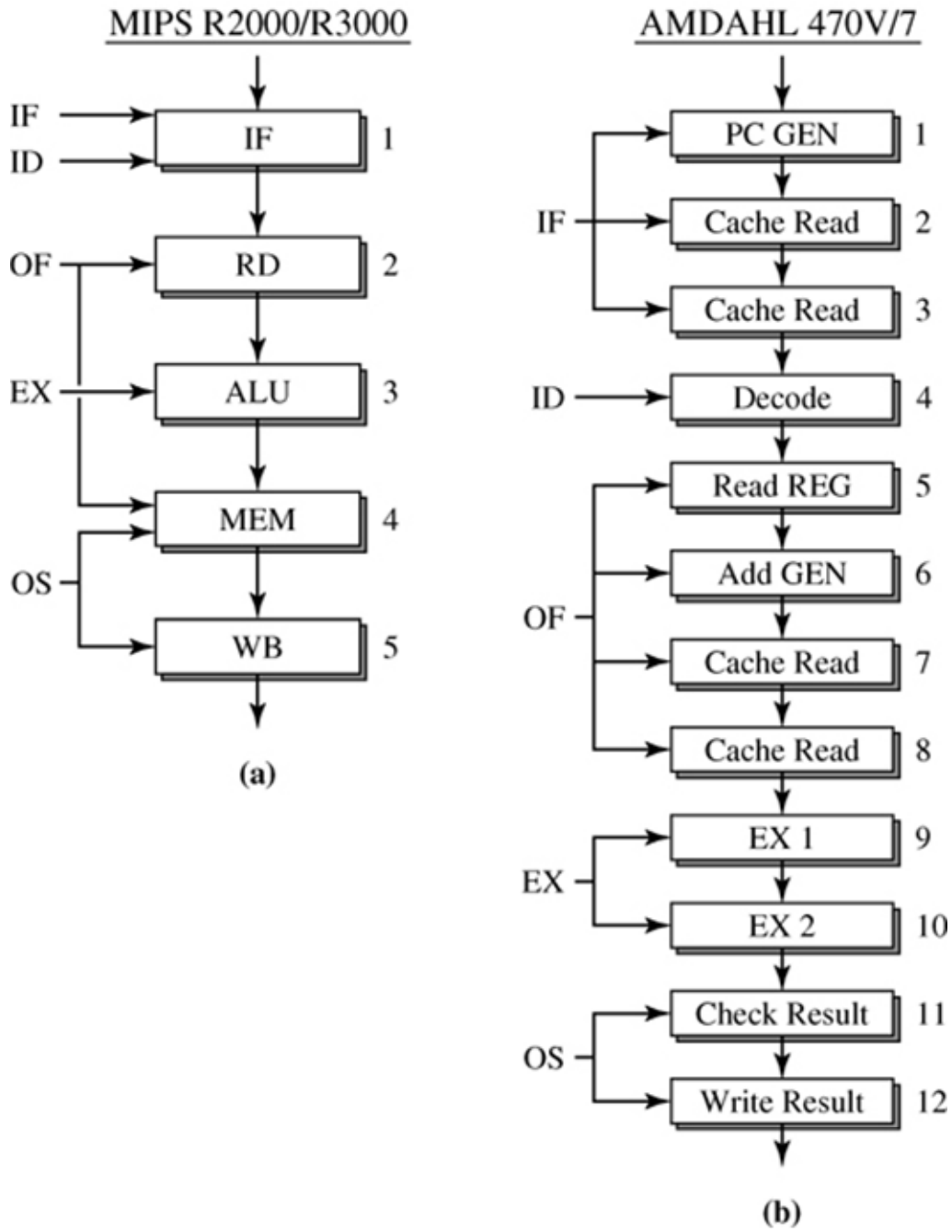

Figure 2.9: Two Commercial Instruction Pipelines: (a) MIPS R32000/R3000 Five-Stage Pipeline; (b) AMDAHL 470V/7 12-Stage Pipeline.

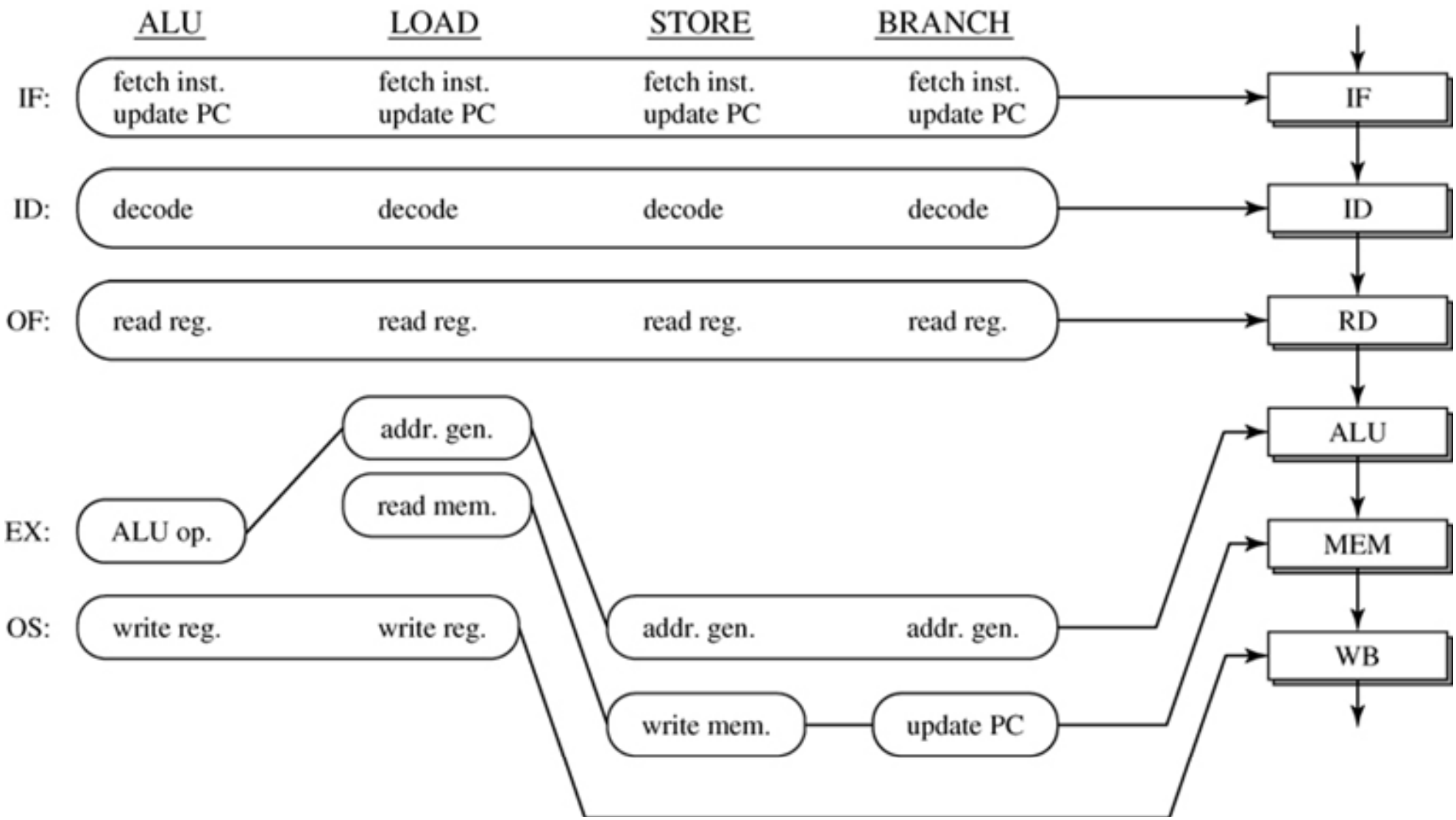

Figure 2.10: Unifying of LU, Load, Store, and Branch Instuction Types into a Six-Stage Instruction Pipeline, Henceforth Identified as the TYPICAL (TYP) Instruction Pipeline.

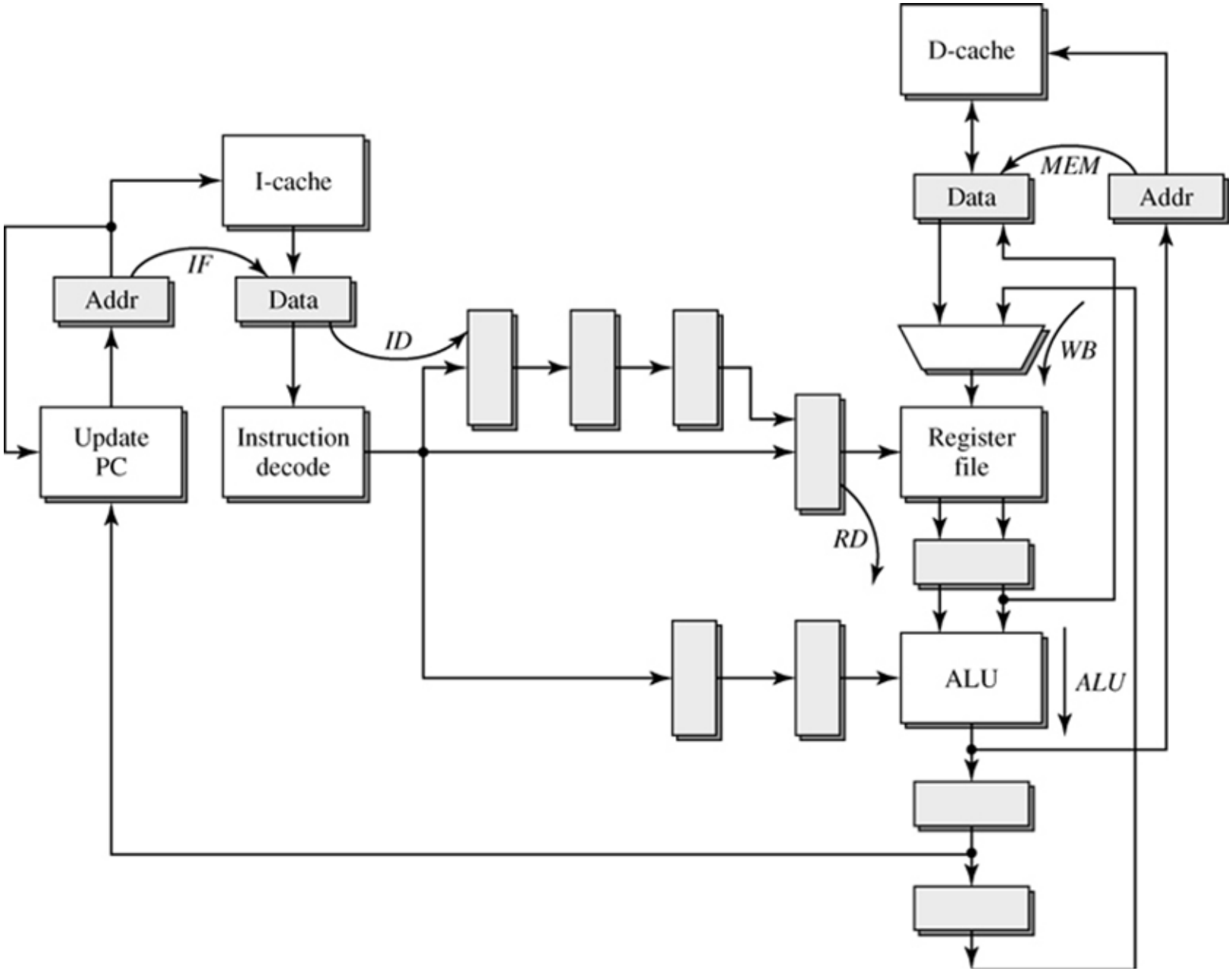

Figure 2.11: The Physical Organization of the Six-Stage TYP Instruction Pipeline.

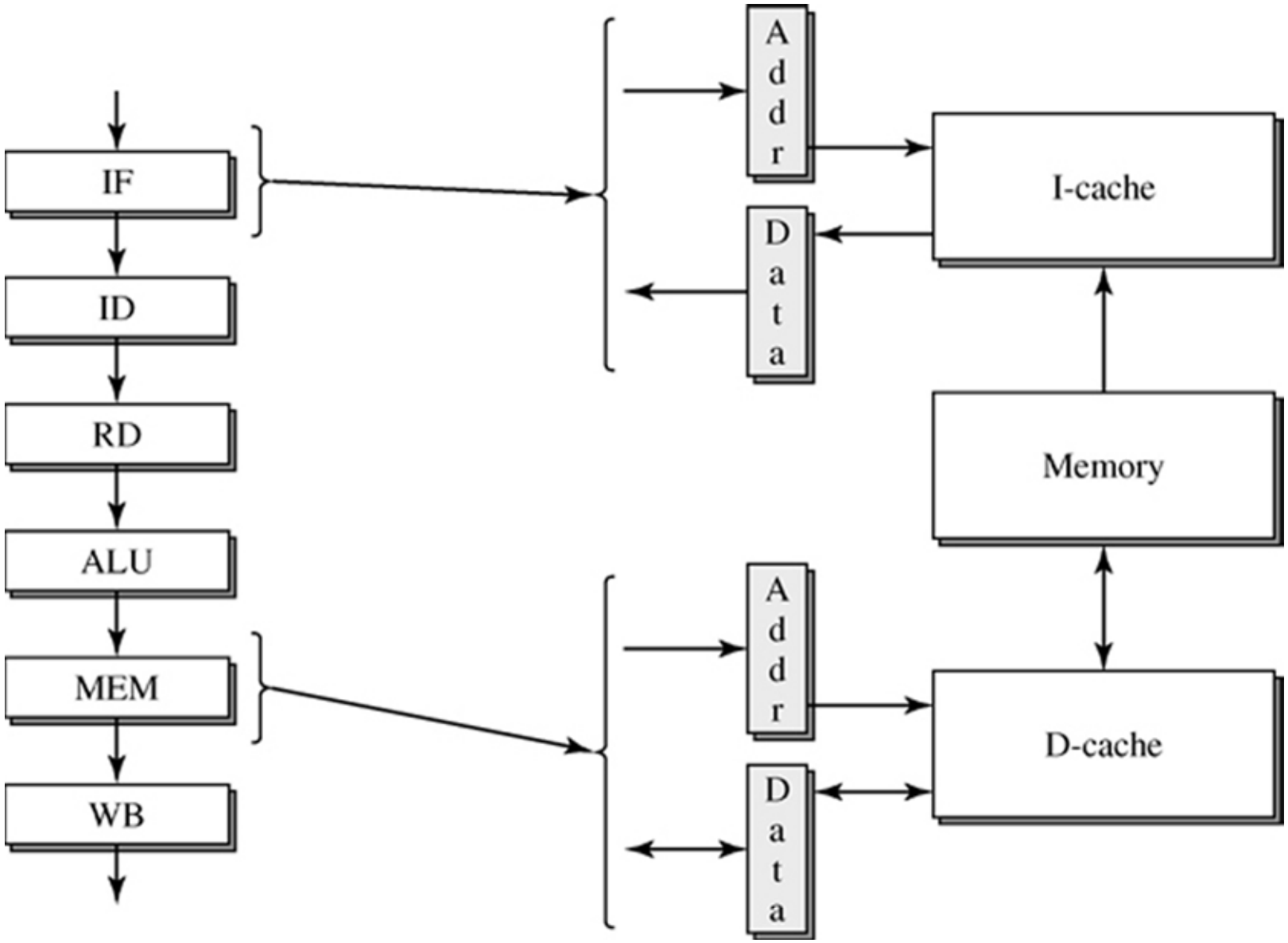

Figure 2.12: The Six-Stage TYP Instruction Pipeline's Interface to the Memory Subsystem.

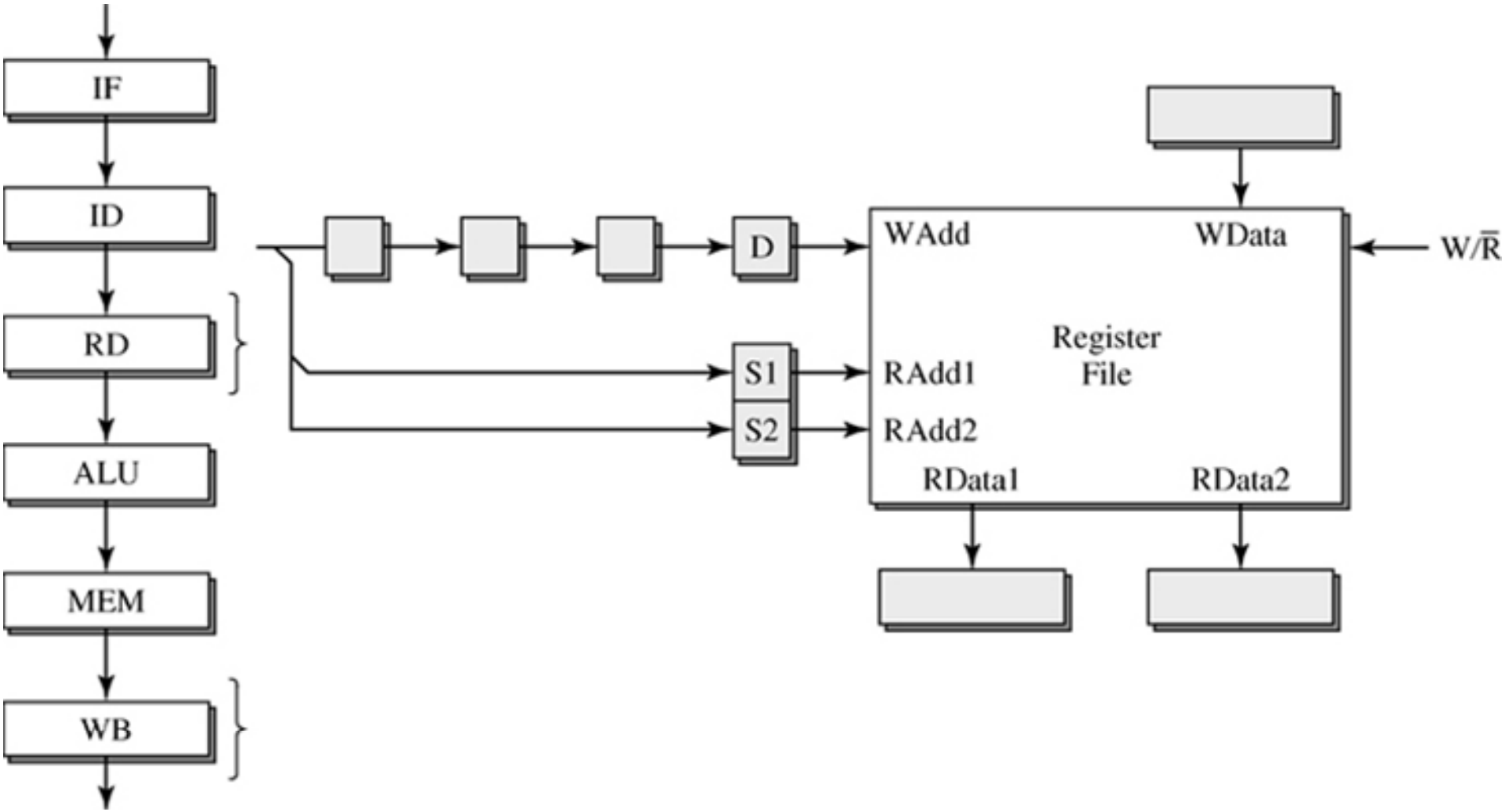

Figure 2.13: The Six-Stage TYP Instruction Pipeline's Interface to the Multiported Register File.

- True dependence

$$R_3 \leftarrow R_1 \; op \; R_2 \qquad \text{Read-after-write (RAW)}$$
$$R_5 \leftarrow R_3 \; op \; R_4$$

- Anti-dependence

$$R_3 \leftarrow R_1 \; op \; R_2 \qquad \text{Write-after-read (WAR)}$$
$$R_1 \leftarrow R_4 \; op \; R_5$$

- Output dependence

$$R_3 \leftarrow R_1 \; op \; R_2 \qquad \text{Write-after-write (WAW)}$$
$$R_5 \leftarrow R_3 \; op \; R_4$$
$$R_3 \leftarrow R_6 \; op \; R_7$$

Figure 2.14: Illustration of RAW, WAR, and WAW Data Dependences.

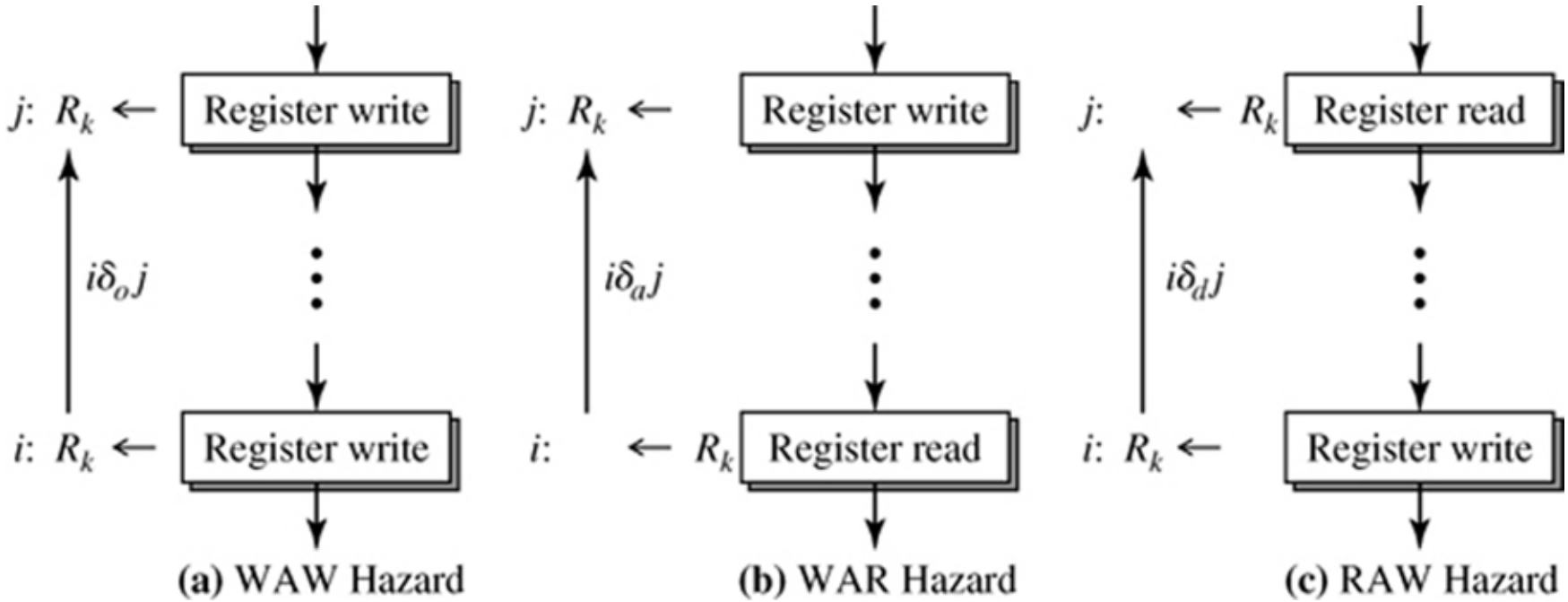

Figure 2.15: Necessary Conditions on the Pipeline Organization for the Occurrence of (a) WAW Hazards, (b) WAR Hazards, and (c) RAW Hazards.

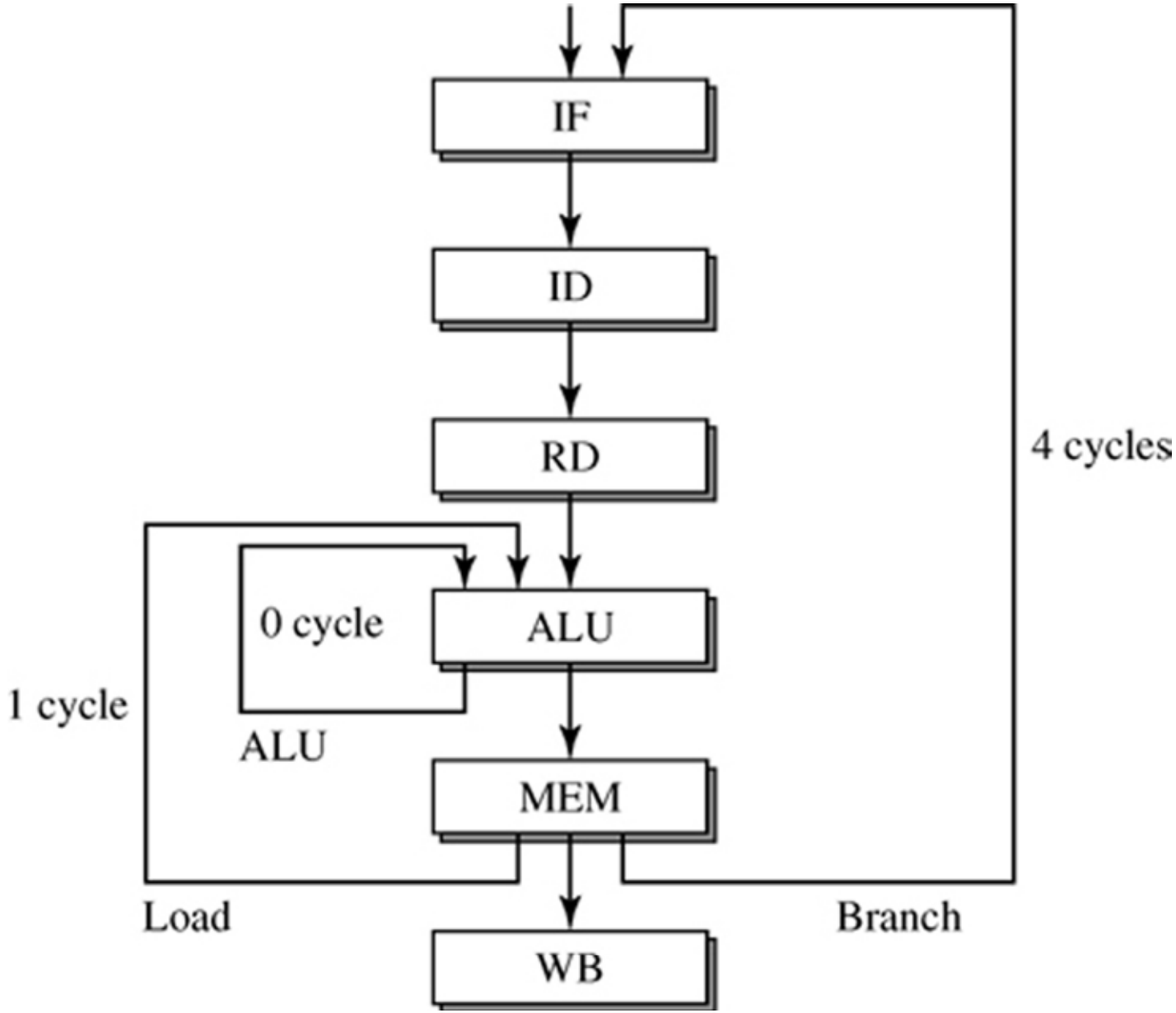

Figure 2.16: Incorporation of Forwarding Paths in the TYP Pipeline to Reduce ALU and Load Penalties.

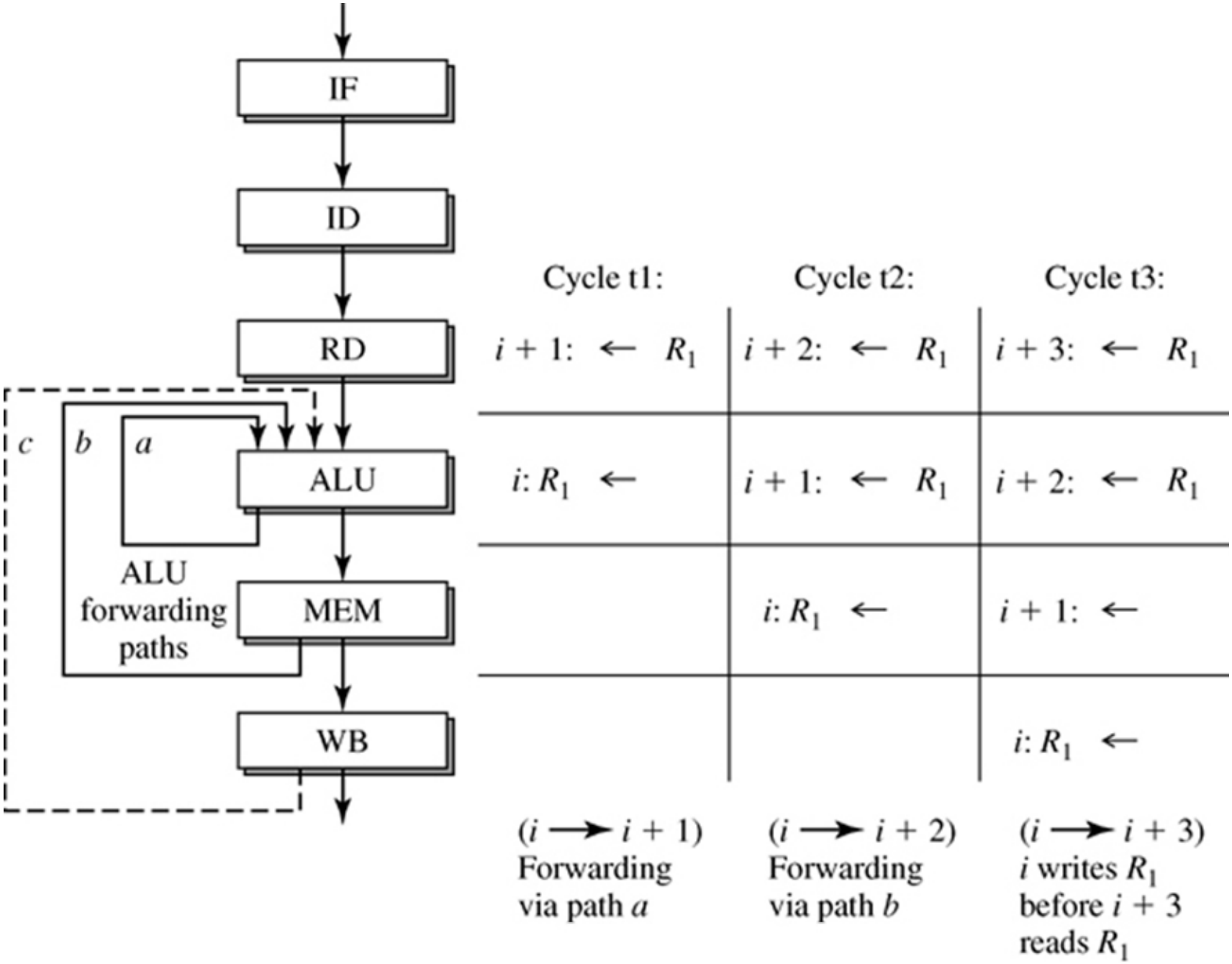

Figure 2.17: Forwarding Paths for Supporting Pipeline Hazards Due to an ALU Leading Instruction.

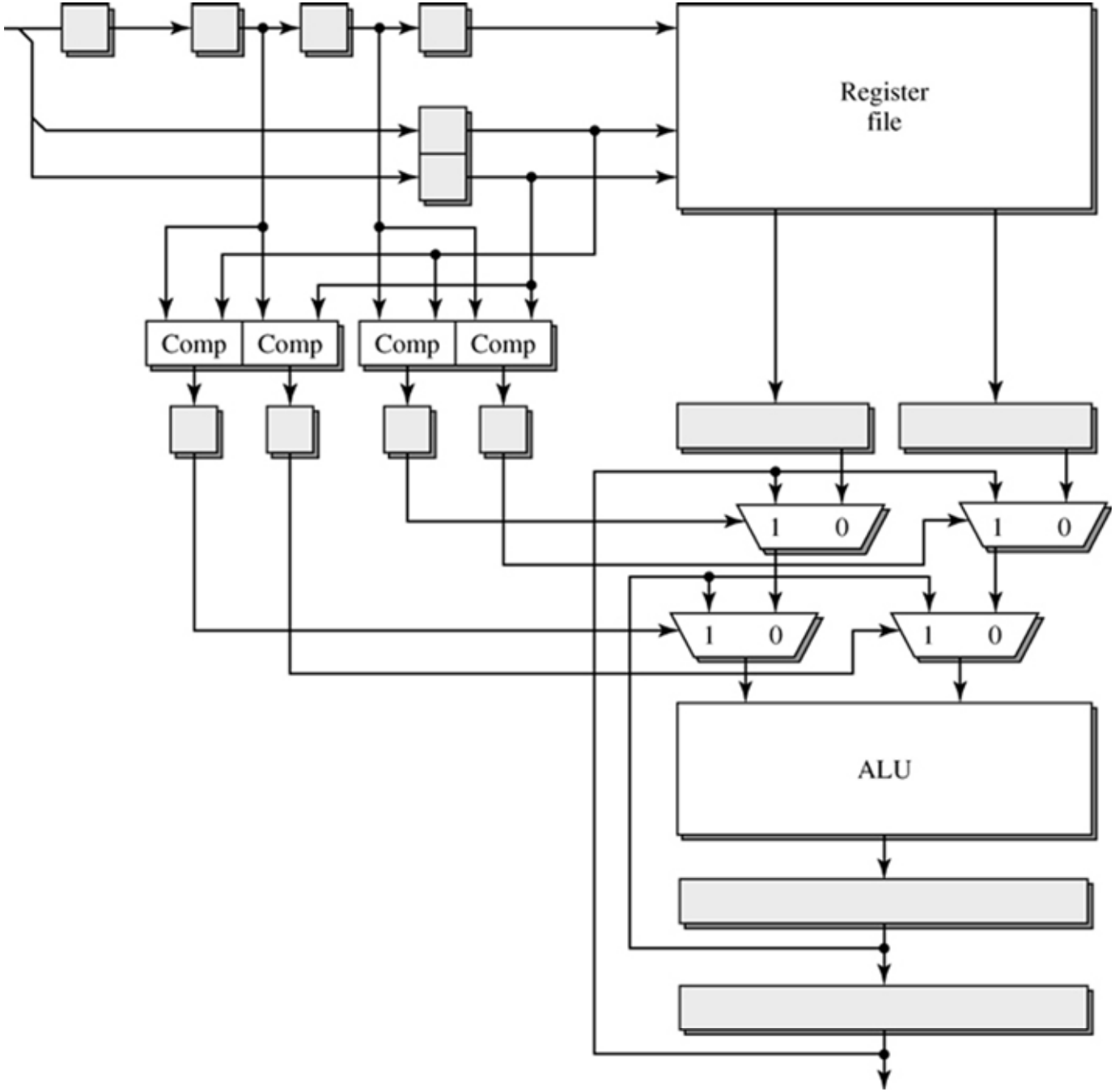

Figure 2.18: Implementation of Pipeline Interlock for RAW Hazards Involving a Leading ALU Instruction.

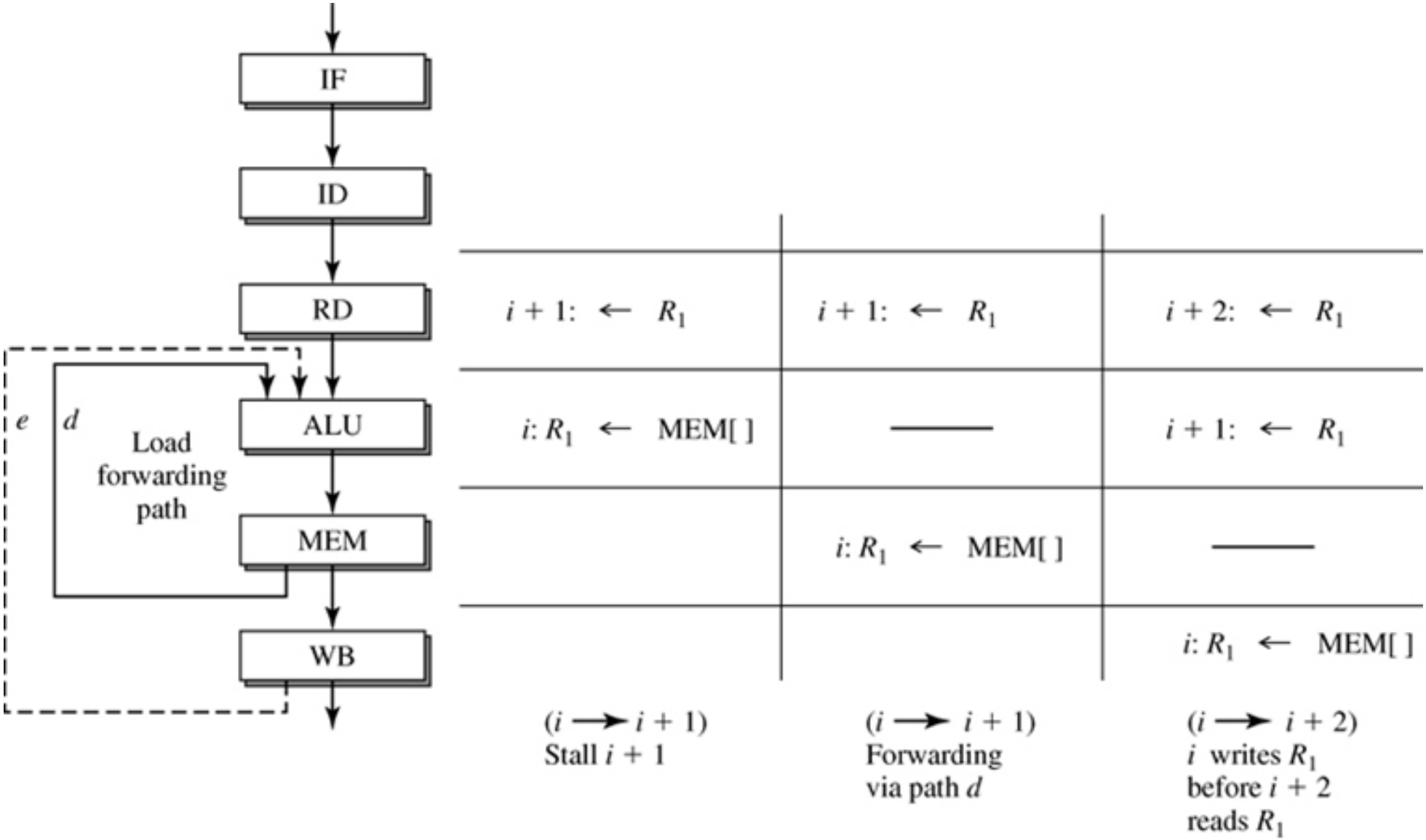

Figure 2.19: Forwarding Paths for Supporting Pipeline Hazards Due to a Leading Load Instruction.

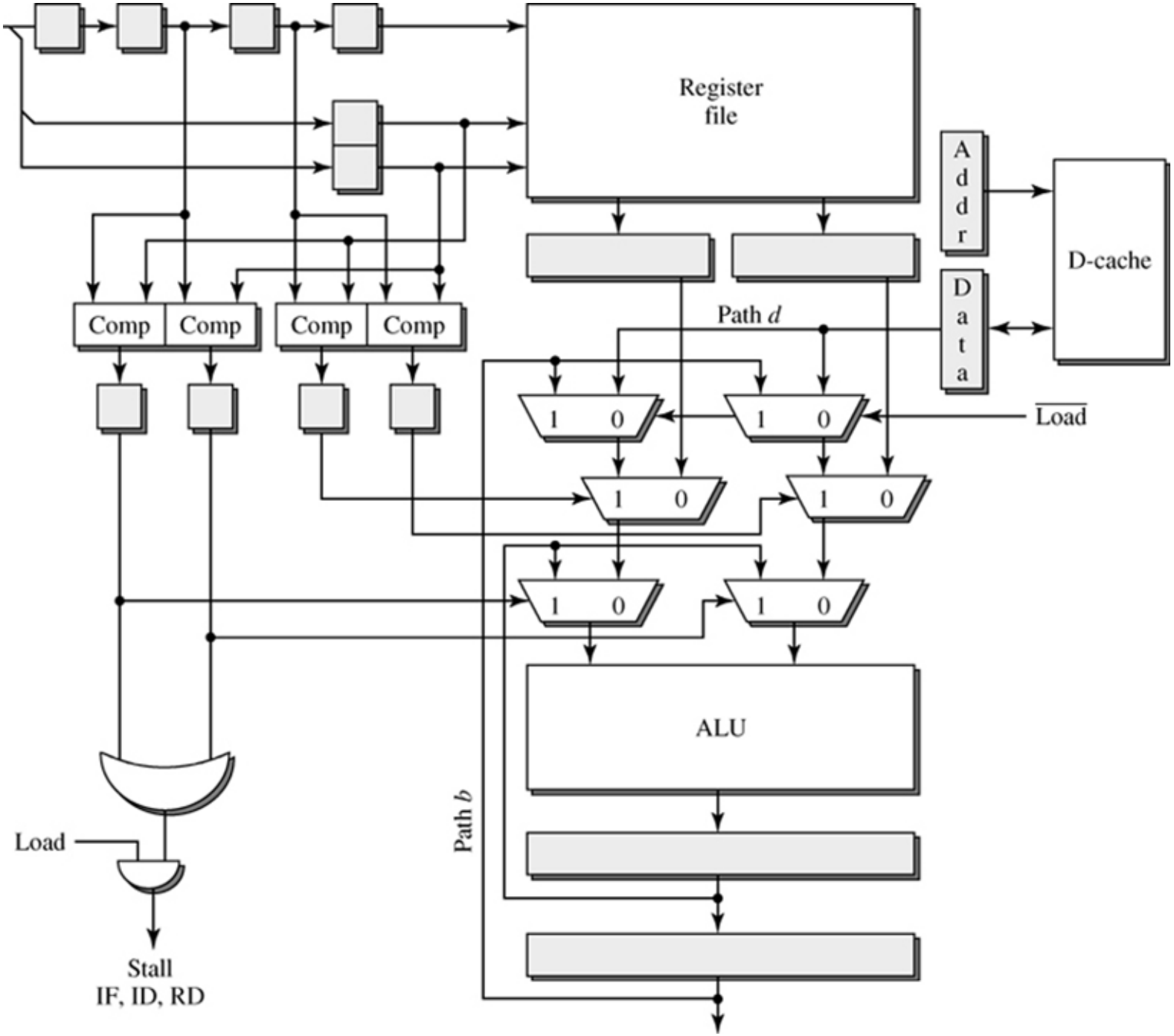

Figure 2.20: Implementation of Pipeline Interlock for RAW Hazards Involving ALU and Load Instructions.

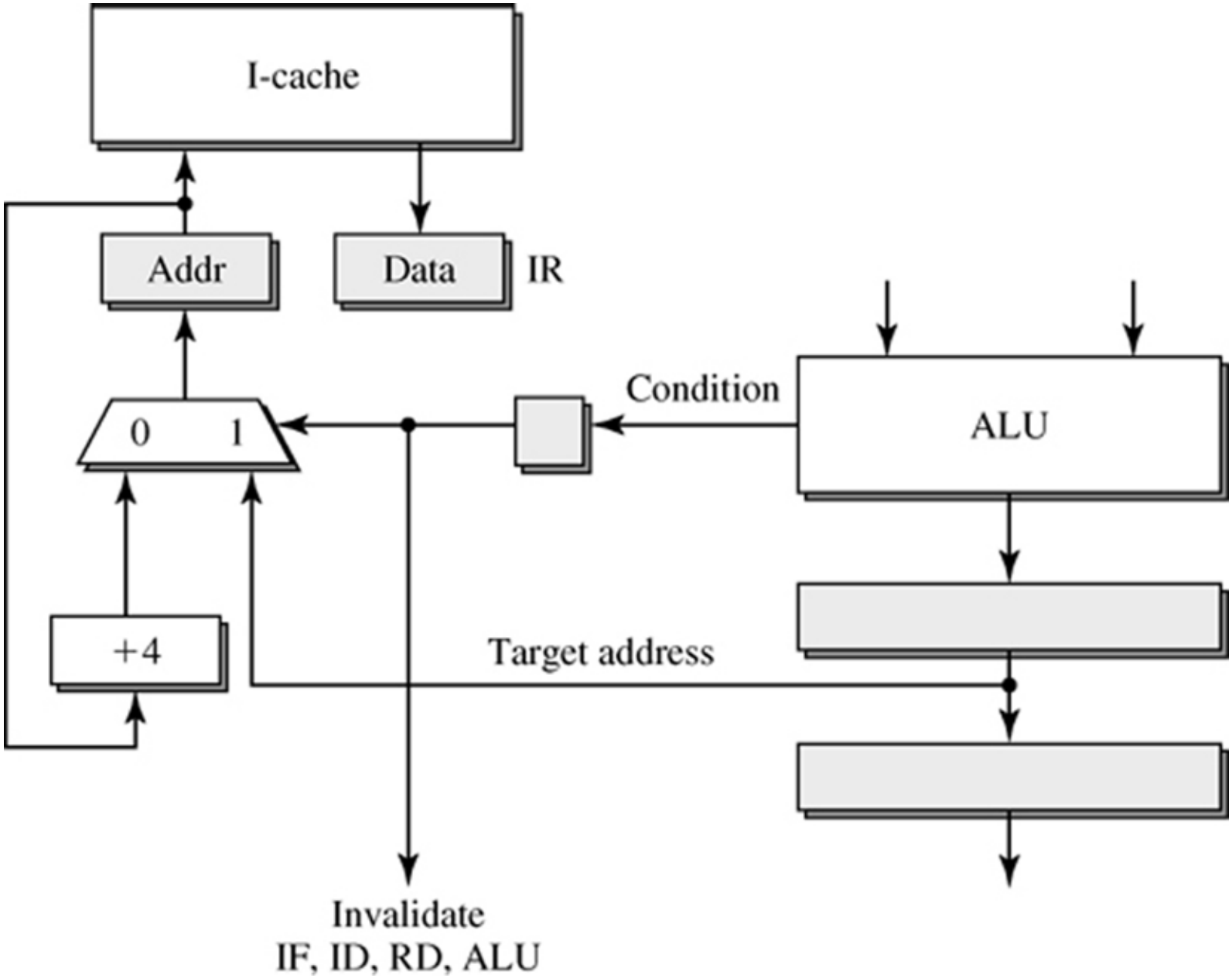

Figure 2.21: Implementation of Pipeline Interlock for Hazards Involving a Branch Instruction.

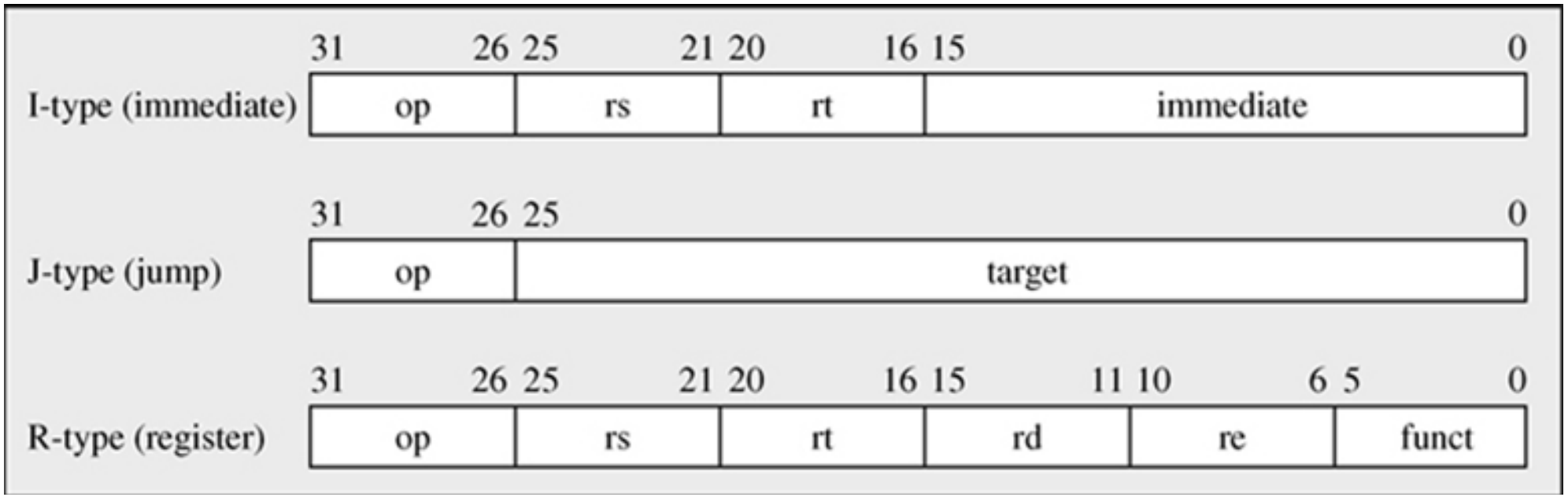

Figure 2.22: Instruction Formats Used in the MIPS Instruction Set Architecture.

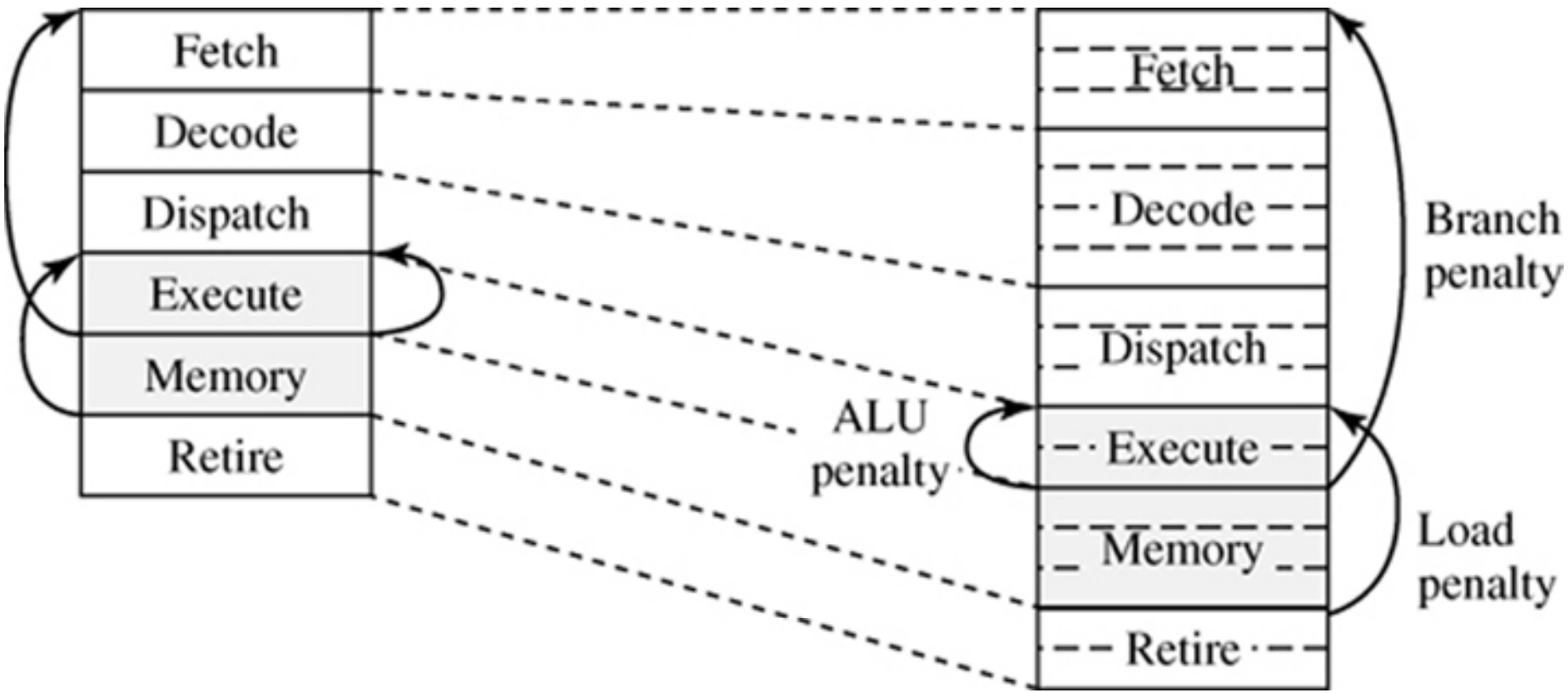

Figure 2.23: Impact on ALU, Load, and Branch Penalties with Increasing Pipeline Depth.

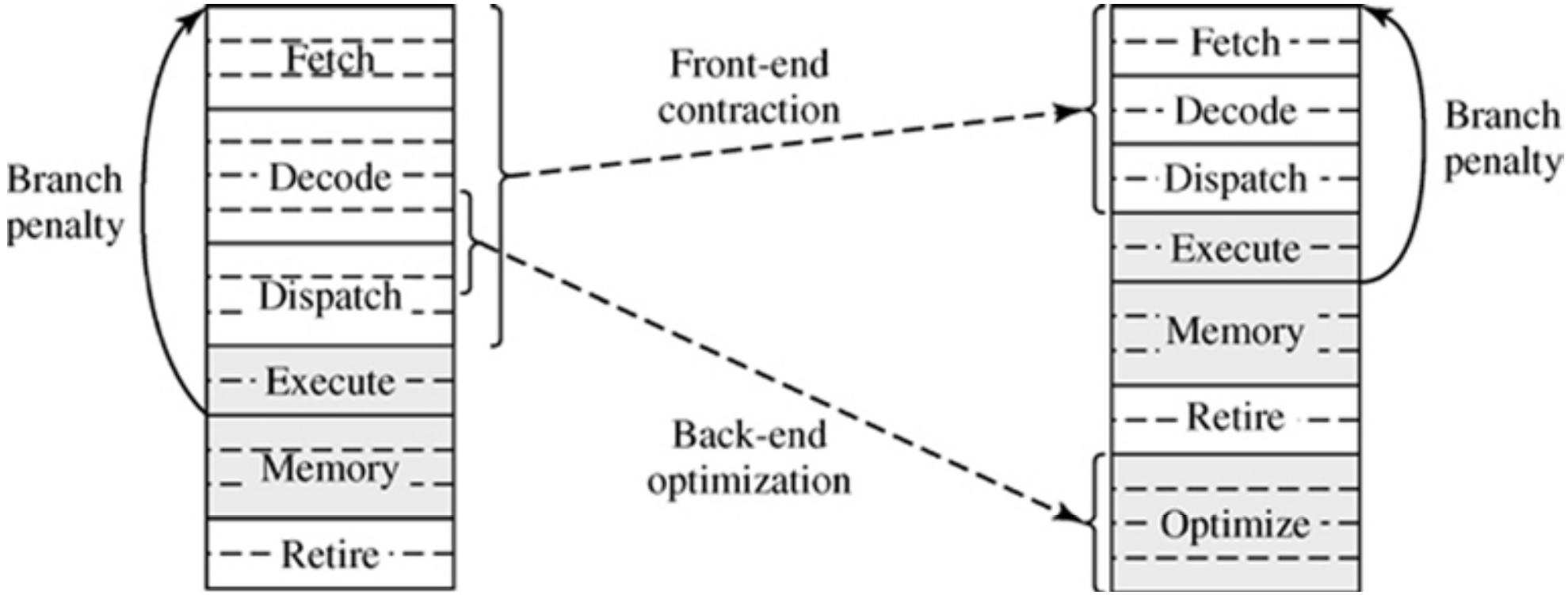

Figure 2.24: Mitigating the Branch Penalty Impact of Deep Pipelines.