Cache Simulation Report

1. Aim

To analyse the behaviour of a **2-way set-associative cache** (T1) and evaluate the impact of the **FIFO (First-In-First-Out) block replacement policy** (T2) on cache performance.

---

2. Introduction

Cache memory reduces access latency by storing frequently used data. This report investigates:

- **Set-Associative Cache Mapping (T1)**
- **FIFO Block Replacement Policy (T2)**
- **Cache Hits and Misses**
- **Performance Impact of FIFO Policy**

A **2-way set-associative cache** allows two blocks per set, improving efficiency. The **FIFO policy** replaces the block that was loaded first when new data needs to be placed.

---

3. System Parameters

Parameter	**Value**
Address Width	8 bits
Cache Size	32 bytes
Block Size	8 bytes
Associativity	2-way set-associative
Write Policy	Write-back
Write Miss Policy	Write-allocate
Replacement Policy	FIFO (First-In-First-Out)

---

4. Methodology

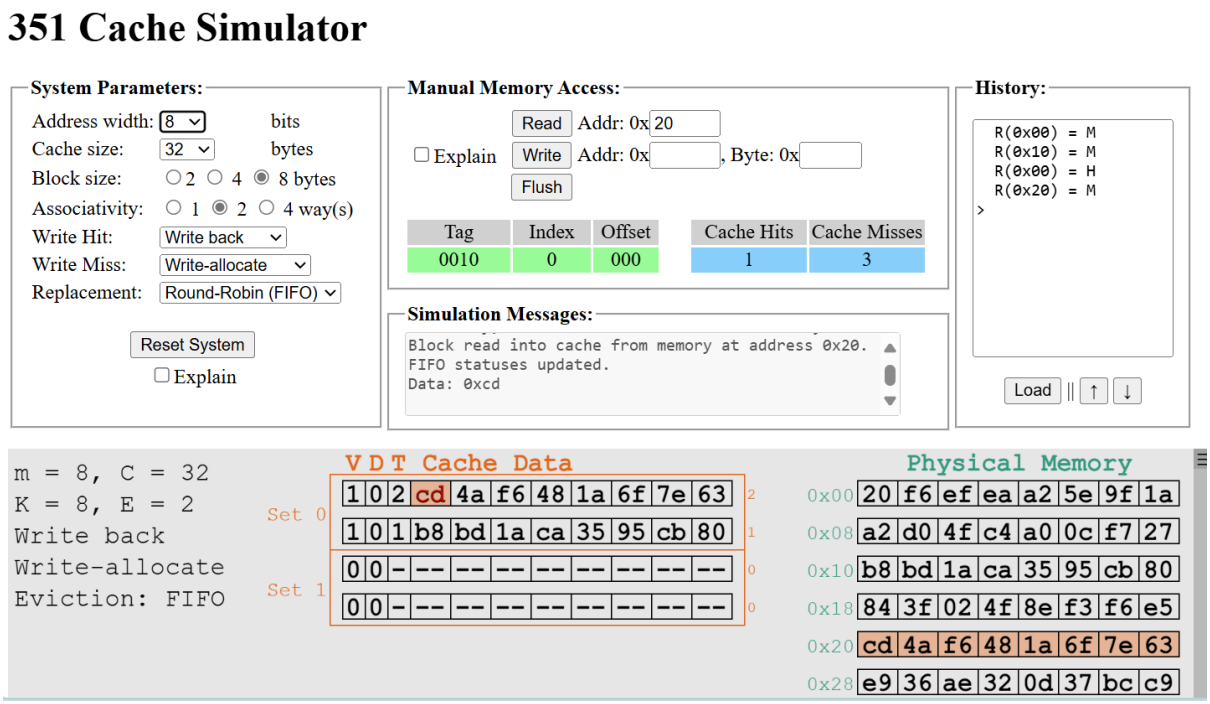
1. **Cache Accesses Performed:**
    - Read instructions executed: R(0x00), R(0x10), R(0x00), R(0x20)

- Cache hits and misses recorded.

2. **Address Breakdown:**
   - **Tag, Index, and Offset** extracted for each memory request.
3. **FIFO Block Replacement:**
   - Oldest block in the set is replaced when new data needs space.
4. **Cache State Analysis:**
   - Studied how data was stored and replaced using **FIFO**.

---

5. **Observations**



Cache Access Breakdown

Address	Tag	Index	Offset	Cache Hit/Miss
0x00	0000	0	000	Miss (M)
0x10	0001	0	000	Miss (M)
0x00	0000	0	000	Miss (M)
0x20	0010	0	000	Hit (H)

Set Associativity (T1)

- **Set 0 contains two blocks:**

    - **Block 1 (Tag = 0010)** → Loaded from **0x20**
    - **Block 2 (Tag = 0001)** → Loaded from **0x10**
- **Blocks are mapped to sets using index bits** to allow multiple blocks in a set.

FIFO Block Replacement (T2)

- **FIFO Eviction Occurred:**
    - **The oldest block (Tag = 0000 from 0x00) was replaced** when 0x20 was accessed.
    - **FIFO ensures blocks are evicted in order of arrival, without considering recent usage.**

Key Findings

- **Set Associativity (T1) reduced direct-mapped conflicts**, improving cache performance.
- **FIFO (T2) replaced the oldest block**, even if it was still in use, potentially causing unnecessary evictions.
- **Performance improved after initial cold misses, but FIFO may not always be optimal compared to LRU.**

---

6. Conclusion

- **Set Associativity (T1) enhances cache efficiency** by allowing multiple blocks in a single set.
- **FIFO (T2) is simple but can lead to suboptimal evictions**, as it does not track block usage frequency.
- **Cache performance improves after the initial cold misses**, but FIFO may not always be the best choice compared to LRU.

This analysis highlights **trade-offs in cache replacement strategies**, demonstrating the importance of selecting the right policy based on workload patterns.